

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Real-time prediction of Wikipedia articles' quality

Pedro Miguel Moás



Mestrado em Engenharia Informática e Computação

Supervisor: Carla Teixeira Lopes

July 18, 2022

Real-time prediction of Wikipedia articles' quality

Pedro Miguel Moás

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. Cristina Ribeiro

External Examiner: Prof. Lucie-Aimée Kaffé

Supervisor: Prof. Carla Teixeira Lopes

July 18, 2022

Abstract

Wikipedia is the largest and most globally well-known online encyclopedia, containing over 6.5 million English articles and averaging 5.9 user contributions per second. It is challenging to ensure article quality throughout the entire website, as vandalism and differing opinions may cause instability in its articles. Moreover, there is a significant quality gap between English Wikipedia and its other versions. Considering non-English articles make up over half of Wikipedia's traffic, that issue is concerning.

Despite its scale, Wikipedia contains many reliable articles. Wikipedia designed a quality scale that distinguishes the outstanding, well-written articles (Featured Articles and Good Articles) from the incomplete and incoherent ones (Starts and Stubs). Unfortunately, that assessment process is primarily manual, so many articles remain unassessed.

Our end goal is to provide Wikipedia users with a reliable and transparent tool for automatically assessing quality within Wikipedia. That way, readers will know beforehand if an article is worth reading, while editors may easily detect existing flaws in the articles they encounter. We thus propose creating an extension for the Google Chrome browser that uses machine learning to predict, in real-time, the quality of Wikipedia articles.

Initially, we review existing literature and compare approaches for automatically predicting quality, including article features, quality metrics, and machine learning algorithms. We then constructed a fully balanced dataset of 36,000 English articles and 145 features. The experimentation phase compared the performance of multiple classification and regression algorithms and studied how the number of classes and features affects the model's performance.

The six-class experiments achieved a classifier accuracy of 64% and a mean absolute error of 0.09 in regression methods, which matches or beats most state-of-the-art approaches. We concluded that using the entire feature set leads to the best performance, but the most important features are the ones related to the content of the article (e.g., number of characters and sections) and those measuring its revision history (e.g., number of reviews and contributors).

We also designed three non-English datasets and grouped them with the English one to analyze potential patterns. We discovered that the quality distribution varies between different languages and confirmed our assumption that numerous elements strongly correlate with article quality, namely character and word count. After assessing the effectiveness of our machine learning approach on those Wikipedia versions, we determined that our model produces, on average, slightly worse results than those obtained with the English dataset.

Based on the collected results, we designed a Chrome extension that applies the trained model in real-time, providing a quality score, a readability score, and some quality features that may be changed to improve article quality. Therefore, it is a unique solution that can assist the millions of readers and editors that regularly use Wikipedia.

Keywords: Wikipedia, Quality Assessment, Machine Learning, Chrome extension

Resumo

A Wikipédia é a maior enciclopédia online do mundo, contendo mais de 6,5 milhões de artigos ingleses e 5,9 contribuições por segundo. É difícil assegurar a qualidade do website inteiro, uma vez que vandalismo e opiniões divergentes podem causar instabilidade em alguns artigos. Além disso, a qualidade da Wikipédia inglesa é muito diferente das outras versões. Esse facto é preocupante, já que mais de metade do tráfego da Wikipédia é destinado a artigos não ingleses.

Apesar do seu tamanho, a Wikipédia contém bastantes artigos fiáveis. A Wikipédia definiu uma escala de qualidade que distingue os artigos bem escritos (*Featured Articles* e *Good Articles*) daqueles mais incompletos e incoerentes (*Starts* e *Stubs*). Infelizmente, esse processo de avaliação é maioritariamente manual, pelo que muitos artigos permanecem por avaliar.

Pretendemos fornecer aos utilizadores da Wikipédia uma ferramenta que os permita, de forma fiável e transparente, avaliar automaticamente a qualidade dos artigos que encontram. Desta forma, os leitores saberão de imediato se devem ler o artigo, enquanto os editores podem facilmente detectar falhas em artigos a melhorar. Propomos a criação de uma extensão do Google Chrome que usa aprendizagem computacional para prever, em tempo real, a qualidade dos artigos na Wikipédia.

Numa primeira fase, analisamos e comparamos as abordagens existentes para prever a qualidade dos artigos, assim como as *features*, métricas e algoritmos que utilizam. De seguida, construímos um *dataset* de 36.000 artigos ingleses e 145 *features*. Durante a fase de experimentação, comparámos o desempenho de vários algoritmos de classificação e regressão e analisamos o efeito que o número de *features* e níveis de qualidade têm no desempenho do modelo.

Os algoritmos de classificação atingiram precisão de 64% e os métodos de regressão obtiveram um erro médio absoluto de 0,09, correspondendo ou até ultrapassando os resultados do estado da arte atual. Concluímos também que a utilização de todas as *features* leva ao melhor desempenho, mas as mais importantes são as relacionadas com o conteúdo do artigo, como número de caracteres e secções, e as que analisam o seu histórico de edição, como número de revisões e editores.

Concebemos também três *datasets* não ingleses analisámo-los ao lado de um inglês. Descobrimos que a distribuição da qualidade varia consideravelmente entre as diferentes línguas e confirmámos que várias *features* se correlacionam fortemente com a qualidade do artigo, nomeadamente o número de caracteres e palavras. Depois de avaliarmos a eficácia do modelo de aprendizagem computacional nessas versões da Wikipédia, determinámos que a nossa abordagem produz, em média, resultados ligeiramente piores do que os obtidos com o *dataset* de artigos ingleses.

Através dos resultados obtidos, criámos uma extensão do Google Chrome que aplica o modelo de aprendizagem computacional em tempo real. Esta aplicação mostra a qualidade do artigo, o nível de inteligibilidade e alguns possíveis aspetos a melhorar. É uma solução única que poderá ajudar os milhões de leitores e editores que utilizam regularmente a Wikipédia.

Palavras-chave: Wikipédia, Avaliação de qualidade, Aprendizagem computacional, Extensão do Chrome

Agradecimentos

Estes últimos cinco anos foram uma longa e trabalhosa jornada que culmina na escrita desta dissertação. Aqui agradeço a todos aqueles que me ajudaram neste projeto tanto de forma profissional como pessoal. Este trabalho não seria possível sem a ajuda de todos aqueles que estiveram do meu lado durante o último ano, dando-me o ânimo e motivação que precisei para continuar a lutar.

Em primeiro lugar, agradeço à minha supervisora, Professora Carla Teixeira Lopes, por me guiar durante todo o processo de escrita e mostrar constante disponibilidade para me ajudar em qualquer momento. Agradeço também ao Tiago Devesas, por se ter disponibilizado para ajudar com o design gráfico da extensão criada neste projeto. Agradeço ainda à FEUP, Universidade do Porto, e todos os professores pelo ensino que tive durante os últimos cinco anos.

Agradeço também aos meus pais pelo apoio que me dão, quer através da mesada, por me fazerem o almoço ou pelas prendas de anos que me deram. Ainda não sei qual de vocês me dava banho quando eu era bebé, mas gosto imenso de vocês e sei que sentem o mesmo por mim. Obrigado irmão por me levares ao MöTAO ver a Kelly Bailey e por pores o Pepas Kuduro no baconal. Sempre me trouxeste até à diversão quando eu precisei dela.

Obrigado ao resto da minha família, desde os meus avós até aos meus tios e primos. Primos alemães, que infelizmente apenas vejo duas vezes por ano. Primos portugueses, que têm sempre um jogo novo para a Páscoa ou para o natal. Primos pirralhos, que já devem ter 16 anos mas que para mim sempre parecem ter 6.

Agradeço agora aos meus amigos. Àqueles que me deram força nos dias maus e energia nos dias bons. Àqueles que trazem diversão e alegria em casa, na praia, no cinema, no restaurante, no café, no bar, na discoteca. Àqueles que têm sempre uma piada para contar, uma aposta para fazer, um plano para combinar. Àqueles que me mantiveram em pé neste último ano.

Obrigado Doniel, por me dares a conhecer um lado deste mundo a que eu não chegaria sozinho. Que mesmo nos piores momentos conseguimos encontrar felicidade onde menos esperamos. Também agradeço a tua hospitalidade em leiden, se voltares lá e encontrares o comprimido avisa, antes que te espumes todo sozinho.

Obrigado Gugu, por toda a cultura e sabedoria que me deste ao longo dos últimos anos. Espero que consigas eventualmente encontrar a tua alma gémea de mãos claras.

Obrigado Guilherme, por me teres dado alojamento e refeição durante uns fins-de-semanazitos porreiros. Penso que isso perdoa não me teres convidado para o teu aniversário em 2020 e por me teres roubado a toalha no ano seguinte.

Obrigado Henrique, por me ensinares o que é que combina bem com açúcar e sugus de limão. Prometo que o assunto do forno ficou ultrapassado mais rápido do que o tempo que o prato demorou a cair.

Obrigado Johnny Milkers, por partilhares uma célula cerebral comigo. Prometo que vou tentar ganhar mais originalidade para ver se conseguimos aumentar esse número e finalmente descobrir quem matou a Sara.

Obrigado Luís, pela frequente disponibilização de alojamento e espaço recreativo, apesar de não me deixares comer em tua casa. Sem ti não saberia dançar 1D: wmyb. Continua a ser quem és, ignora os haters, nunca te esqueças do que aprendemos: “a partir de agora estão a valer bujas”.

Obrigado Pedro, pela tua *expertise* a utilizar expressões que envolvem pessoas idosas, partes do corpo e vegetais úteis para fazer estrugido. Tenho saudades dos teus planos a negrito no zap, mas a chamada ocasional às três da manhã é um bom substituto. Só gostava de fazê-lo contigo cá, mas sei que só vais descansar quando acabares com todos os terroristas ou finalmente te tornares um deles.

Obrigado Silva, pelos bons apetites que partilhamos em família, pelas festas em que ninguém sabe quem é que convidou certa pessoa, e pelos limites que tanto gostavas. Desde que vendeste a alma ao SNS24 que nunca mais te vi, mas não penses que me esquecerei de ti tão facilmente.

Obrigado Vasco, por aguentares. Sei que às vezes podes achar que não tens para onde ir, nem que fazer. Não sabes bem quem és aqui, neste mundo. Mas não te esqueças que, tal como tu, eu sou apenas um aprendiz de viajante, e sigo como uma ovelha que encontrou pastor. Quando perceberes isso, então verás que a vida se pode tornar... melhor. O único conselho que tenho é, agarra a vida. Ela é boa. Ela é para todos. Para toda a vida.

Obrigado Walex, por me tornares apenas no segundo nerd de 2000 da GB. Quando quiseres jogar catan manda mail, mas se for um daqueles jogos de cartas parvos dispenso. E olha, embora não tenhas vindo ter connosco na holanda, até és um gajo porreiro. Anseio pelo fim to teu jantar.

Obrigado a todos os outros membros da Gente Baril. Desde as opiniões arquitetónicas da Margarida, às *stories* progressivas da Lottie, à paixão por Robocar Poli que a Crol partilha comigo, aos vídeos brasileiros do Rôla, ao twister da Malu, à música do Branco e à dança da Rita, aos tweets do Vicente e às opiniões controversas do C. Geraldo.

Por fim, obrigado às pessoas que tive o prazer de conhecer ao longo dos últimos meses. Não vale a pena dizer o nome do sítio, pois ninguém deve conhecer, mas já que mudaram completamente a minha vida, digo os vossos nomes. Obrigado Alex, Carla, Catarina, Chelsia, Diana, Duarte, Filipa, Manuela, Moreira, Tavares, Rita e Zé, vocês deram-me aquilo que eu mais precisava quando eu não sabia o que era. Vimos e vivemos tanto em tão pouco tempo, só tenho pena de não vos ter conhecido mais cedo. É certo que partilhámos momentos bons e momentos menos agradáveis, mas temos de saber que viver traz de tudo, e eu estou grato por cada segundo que vivi com cada um de vocês.

Obrigado a todos aqueles que estiveram ao meu lado quando precisei e quando não precisei.

Obrigado. #GB #VLC

*“You can have data without information,
but you cannot have information without data.”*

Daniel Keys Moran

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	1
1.3	Objectives	3
1.4	Information Quality	4
1.5	Contributions	5
1.6	Document Structure	6
2	Machine Learning Background	7
2.1	Supervised Learning vs. Unsupervised Learning	7
2.2	Classification vs. Regression	9
2.3	Performance and Error Measures	9
2.4	Bias-Variance Trade-off and Overfitting	10
2.5	Supervised Learning Algorithms	12
2.5.1	Decision Trees	12
2.5.2	K-Nearest Neighbors	13
2.5.3	Support Vector Machines	13
2.5.4	Naive-Bayes	14
2.5.5	Neural Networks	15
2.5.6	Linear Regression	16
2.5.7	Logistic Regression	16
2.5.8	Ensemble Methods	16
2.6	Document Features	16
3	Literature Review	19
3.1	Background	19
3.1.1	Quality Assessment Methods	19
3.1.2	Wikipedia Network Graph	20
3.2	Methodology	21
3.2.1	Research Questions	21
3.2.2	Data Sources	22
3.2.3	Selection Process	22
3.2.4	Data Collection	24
3.3	Overview of Analyzed Articles	26
3.3.1	Year of Publication	26
3.3.2	Publication Venue and Number of Citations	26
3.3.3	Abstract and Keyword Analysis	27
3.3.4	Authors and Affiliations	28

3.4	Quality Features and Metrics	30
3.4.1	Content Features	32
3.4.2	Style Features	32
3.4.3	Readability Features	36
3.4.4	History Features	38
3.4.5	Network features	38
3.4.6	Stvilia's IQ Metrics	40
3.5	Machine Learning Approaches	41
3.5.1	Algorithms and Performance	41
3.5.2	Feature Importance	42
3.5.3	Datasets	43
3.6	Discussion	44
4	Methodology	47
4.1	MediaWiki Action API	47
4.1.1	Relevant Modules	47
4.1.2	Wikitext	48
4.2	Dataset Definition	48
4.2.1	Title Collection	48
4.2.2	Dataset Dimensioning	50
4.2.3	Article Collection and Aggregation	51
4.2.4	Network Graph Generation	51
4.2.5	Dataset Storage	51
4.3	Feature Computation	53
4.3.1	Wikitext Cleaning	53
4.3.2	Natural Language Processing Tools	55
4.3.3	Content Features	55
4.3.4	Style Features	55
4.3.5	Readability Features	57
4.3.6	History Features	57
4.3.7	Network Features	59
4.4	Experimentation	59
4.4.1	Algorithm Comparison	60
4.4.2	Quality Class Mapping	62
4.4.3	Feature Subset Optimisation	63
4.4.4	Training and Prediction Times	63
4.4.5	Multi-language Performance	64
5	Results	67
5.1	Dataset Analysis	67
5.2	Algorithm Performances	69
5.3	Analysis of Number of Classes	73
5.4	Feature Category Importance	76
5.5	Training and Prediction Times	79
5.5.1	Training Times	79
5.5.2	Model Prediction Times	81
5.5.3	Feature Computation Times	81
5.6	Multi-language Dataset Analysis	83
5.7	Multi-language Model Performance	87

5.8	Final Remarks	88
6	Chrome Extension Development	89
6.1	Architecture	89
6.2	Client	90
6.2.1	Quality Prediction	90
6.2.2	Feature Analysis	91
6.2.3	Readability Score	93
6.3	Server	94
7	Conclusion	95
7.1	Future Work	96
7.2	Limitations	96
A	Literature Review Results	99
A.1	Screening Phase - Abstract Scanning	99
A.2	Eligibility Phase - Full Text Assessment	104
A.3	Inclusion Phase - Relevance Scoring	106
B	MediaWiki Action API	107
B.1	MediaWiki Action API Endpoints	107
C	Feature Set Description	113
D	Machine Learning Results	119
D.1	Dataset Analysis	119
D.2	Algorithm Performance	120
D.3	Performance per Number of Classes	120
D.4	Feature Subset Performance	120
D.5	Training and Prediction Times	122
D.6	Multi-language dataset analysis	125
D.7	Multi-language model performance	127
E	Chrome Extension Mockups	129
	References	133

List of Figures

1.1	English Wikipedia growth since its creation	2
1.2	Top-10 largest Wikipedias by number of articles	2
1.3	Top-10 smallest Wikipedias (over 100k) by number of articles	3
2.1	Learning diagram for Supervised Learning	8
2.2	Learning diagram for Unsupervised Learning	9
2.3	Bias-Variance trade-off illustrated	11
2.4	Overfitting and Underfitting	12
2.5	Feature space partition and decision tree	13
2.6	Comparison between KNN decision boundaries using different values of K	14
2.7	Comparison between SVM margins using different values of C , on a binary clas- sification problem.	14
2.8	Simplified Neural Network diagram	15
2.9	Learning with the raw text	17
2.10	A comprehensive diagram of the learning process with a feature set	18
2.11	Usage of a model trained with a feature set	18
3.1	Diagrams of the different types of Quality Assessment Methods	20
3.2	Example of a Wikipedia Article's Graph	21
3.3	Identification phase: Titles assessed vs. Papers included	23
3.4	Article's Selection Process	25
3.5	Reviewed papers by year of publication	27
3.6	Word Cloud of article abstracts	30
3.7	Word Cloud of article keywords	30
3.8	Number of authors per affiliation's country	31
3.9	Collected Features: Count per Category	32
4.1	A portion of Wikipedia's <i>Island</i> article and its respective wikitext	49
4.2	Diagram representation of the information extracted from the graph	52
4.3	Dataset Features: Count per Category	53
4.4	<i>Frits Palm's</i> Wikipedia Page: Original Wikitext (April 2022)	54
4.5	<i>Frits Palm's</i> Wikipedia Page: Cleaned Wikitext (April 2022)	55
5.1	Average character and word count of each quality class	68
5.2	Average character per word, word per sentence, and character per sentence of each quality class	68
5.3	Average article age of each quality class	69
5.4	Average number of revisions of each quality class	70

5.5	Average grade level of each quality class, calculated from various readability formulas	70
5.6	Accuracy of classification models trained with all features and classes	71
5.7	F1-score of classification models trained with all features and classes, per article quality	72
5.8	MAE of regression models trained with all features and classes	72
5.9	MAE of regression models trained with all features and classes, per article quality	73
5.10	Accuracy of classification models trained with various numbers of classes (all features)	74
5.11	Avg_{Dist} of regression models trained with various numbers of classes (all features)	75
5.12	Accuracy of SVC models trained on feature sets with four categories	76
5.13	Accuracy of SVC models trained on feature sets with one category	77
5.14	Accuracy of SVC models trained on other feature sets	77
5.15	MAE of Random Forest regression models trained on feature sets with four categories	78
5.16	MAE of Random Forest regression models trained on feature sets with one category	78
5.17	MAE of Random Forest regression models trained on other feature sets	79
5.18	Algorithm training times with different numbers of classes	80
5.19	Algorithm training times with different feature subsets	80
5.20	Algorithm training times with six classes and all feature categories	81
5.21	Average algorithm prediction times across different feature subsets and number of classes	82
5.22	Average percentage of total time spent per feature computation task	82
5.23	Total prediction time of feature computation tasks related to each feature category	83
5.24	Boxplot of feature computation times per quality level	84
5.25	Quality distribution across datasets constructed from different Wikipedias	85
5.26	Comparison between the quality distribution of different Wikipedia versions	86
5.27	Comparison between Average Character Count (CC) and Word Count (CW) of different language datasets	87
5.28	Comparison between MSE of model predictions across different dataset languages	88
6.1	Chrome Extension Architecture	89
6.2	Chrome extension mockup: application layout	90
6.3	Chrome extension mockup: functionality in a non-Wikipedia page	90
6.4	Chrome extension mockup: quality prediction	91
6.5	Chrome extension mockup: feature analysis	92
6.6	Chrome extension mockup: readability score analysis	93
D.1	Comparison between Average Character Count of different language datasets	126
D.2	Comparison between Average Character Count-Word Count ratio of different language datasets	126
E.1	Chrome extension mockup: A+ prediction	129
E.2	Chrome extension mockup: A prediction	130
E.3	Chrome extension mockup: B prediction	130
E.4	Chrome extension mockup: C prediction	131
E.5	Chrome extension mockup: D prediction	131
E.6	Chrome extension mockup: F prediction	132

List of Tables

1.1	Wikipedia Quality Scale: English Articles per class	4
2.1	Performance measures for Binary Classifiers	10
2.2	Error metrics for Regression Models	10
3.1	Search queries submitted in each database	22
3.2	Inclusion Criteria	23
3.3	Exclusion Criteria	24
3.4	Relevance Score Questions	24
3.5	Average Relevance Scores	26
3.6	Bibliometric overview of reviewed publications	28
3.7	Authors and their Affiliations	29
3.8	List of Content Features mentioned in the assessed publications	33
3.9	List of Style Features mentioned in the assessed publications	34
3.10	List of Readability Features mentioned in the assessed publications	38
3.11	List of History Features mentioned in the assessed publications	39
3.12	List of Network Features mentioned in the assessed publications	40
3.13	Best Machine Learning algorithms from included publications	42
4.1	MediaWiki API: Important Query commands	48
4.2	Wikipedia categories of each quality class	49
4.3	Number of article titles gathered per quality class	50
4.4	Article quality distribution within the dataset	52
4.5	Computer Specifications	52
4.6	Intuition behind the Calculation of Content Features	56
4.7	Excluded Style Features	56
4.8	Intuition behind the Calculation of Style Features	57
4.9	Excluded History Features	58
4.10	Adapted History Features	58
4.11	Excluded Network Features	59
4.12	Training Experiments: Naming Convention	60
4.13	Model Experimentation: Used Machine Learning Algorithms	61
4.14	Regression Experimentation Class Mapping vs. Linear Class Mapping	62
4.15	Regression Experimentation: Quality-to-class Mapping	62
4.16	Classification Experimentation: Quality-to-class Mapping	62
4.17	Dataset sizes for different numbers of classes	63
4.18	Model Experimentation: Feature Subset Combinations	64
4.19	Feature Computation Steps and which Feature Categories use them	65
4.20	Model Experimentation: Quality Levels per Language	65

4.21	Model Experimentation: Multi-language Dataset Sizes	66
5.1	Confusion Matrix of SVM classifier trained with all features	72
5.2	Number of Classes: Most effective classifiers for each experiment	74
5.3	Accuracy of Random models vs. SVC models, by number of classes	75
5.4	Number of Classes: Most effective regressors for each experiment	75
5.5	Quality distribution across datasets constructed from different Wikipedias	84
6.1	Chrome extension letter grading system	91
6.2	Highlighted features and respective FA percentiles	92
6.3	Feature adjustments for computation of readability score	94
6.4	Available server endpoints	94
A.1	Literature Review: Publications assessed in the Screening Phase	99
A.2	Literature Review: Publications assessed in the Eligibility Phase	104
A.3	Literature Review: Publications assessed in the Inclusion Phase	106
B.1	Wikipedia version list and respective URLs	108
C.1	Dataset definition: List of used Content Features	114
C.2	Dataset definition: List of used Style Features	115
C.2	Dataset definition: List of used Style Features	116
C.3	Dataset definition: List of used Readability Features	117
C.4	Dataset definition: List of used History Features	117
C.5	Dataset definition: List of used Network Features	118
D.1	Average feature values for each quality level	119
D.2	Average readability values for each quality level	119
D.3	Performance metrics of classification algorithms (CSRHN6)	120
D.4	Performance metrics of regression algorithms (CSRHN6)	120
D.5	Accuracy of classifiers trained with various numbers of classes (all features)	121
D.6	MAE of regressors trained with various numbers of classes (all features)	121
D.7	Accuracy of classifiers trained with various feature subsets (all classes)	121
D.8	MAE of regressors trained with various feature subsets (all classes)	122
D.9	Algorithm training times across different feature sets and number of classes, in seconds	123
D.10	Model prediction times across different feature sets and number of classes in milliseconds	124
D.11	Feature computation time measurements per article quality (in seconds) and average percentages of total time spent	125
D.12	Comparison between Average Character Count (CC), Word Count (CW), and CC-CW ratio of different language datasets	125
D.13	Comparison between Average Character Count (CC), Word Count (CW), and CC-CW ratio of the A+ articles from different language datasets	125
D.14	MSE of model predictions across different dataset languages, with categories CSRH	127
D.15	MSE of model predictions across different dataset languages, with categories CRH	127
D.16	MSE of model predictions across different dataset languages, with categories CH	127

Chapter 1

Introduction

This first chapter explains the problem we aim to solve with this study and summarizes the structure of this document.

1.1 Context

Wikipedia is the largest and most well-known online encyclopedia globally and has kept its growing pace for years. It was designed in 2001 by Jimmy Wales [92], and two years later, he founded the non-profit Wikimedia Foundation¹ to fund and host the website. Figure 1.1 summarizes the growth of English Wikipedia since 2001. As of June 2022, it contains over 6.5 million English articles [95], with versions across a list of 314 active languages [91].

Not only is Wikipedia free, but it is also fully managed by human volunteers, averaging contributions at a rate of 5.9 edits per second during 2021 [87]. Its reader base is growing steadily, considering that, last year, Wikipedia totaled close to 260 billion page views across 2 billion unique devices [87].

Some studies even show that most search engines frequently include Wikipedia pages in their results. According to Vincent and Hecht [84], 80% of *common* (frequent) queries and 70% of *trending* queries (news/events) return results from Wikipedia on the first page when tested with search engines like Google, Bing, and DuckDuckGo.

1.2 Motivation

The fully collaborative aspect of Wikipedia also brings its own set of challenges. The lack of centralized authority over the editors makes it challenging to ensure quality throughout the website. Nonetheless, only 8.1% of edits are reverted [87], showing that vandalism and the so-called *revert wars* are relatively uncommon.

¹Wikimedia's website is available at <https://wikimediafoundation.org/>

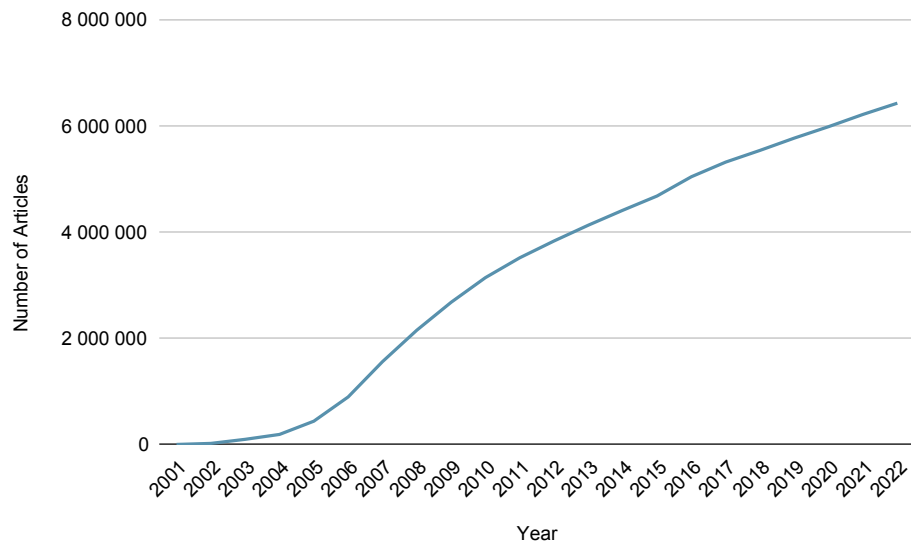


Figure 1.1: English Wikipedia growth since its creation (Data extracted from Wikipedia [95])

Another issue is the significant quality discrepancy between English Wikipedia and its other versions, which comprise around 57% of total Wikipedia traffic [87]. For starters, non-English Wikipedias cover a much smaller number of articles, as shown in Figures 1.2 and 1.3, which display the number of articles in the ten largest and smallest Wikipedia versions, only considering those with over 100,000 articles. Furthermore, they are often much more incomplete as well. Roy et al. [69] demonstrate that English articles from Wikipedia are usually longer than their translations. For instance, they determined that German articles are, on average, 30% shorter than English ones, and for Spanish articles, that value increases to 47%. Couto and Lopes [23] have also shown English articles rank higher on specific quality metrics, although only focused on health-related articles.

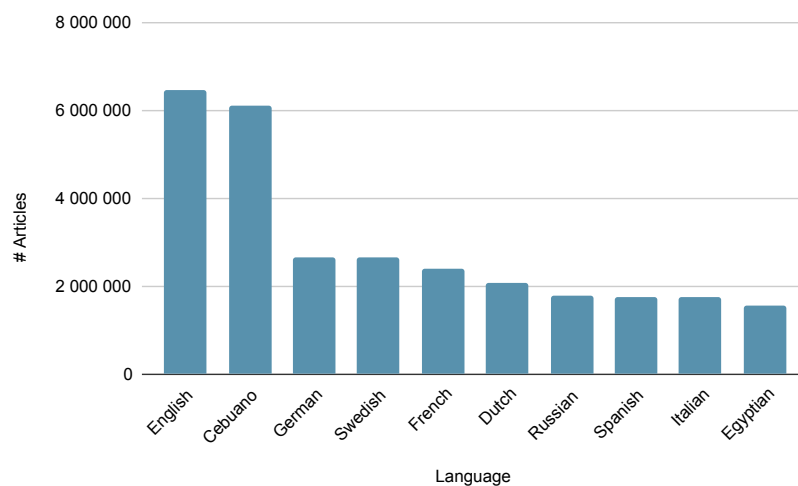


Figure 1.2: Top 10 largest Wikipedias by number of articles (Data extracted from Wikipedia [91])

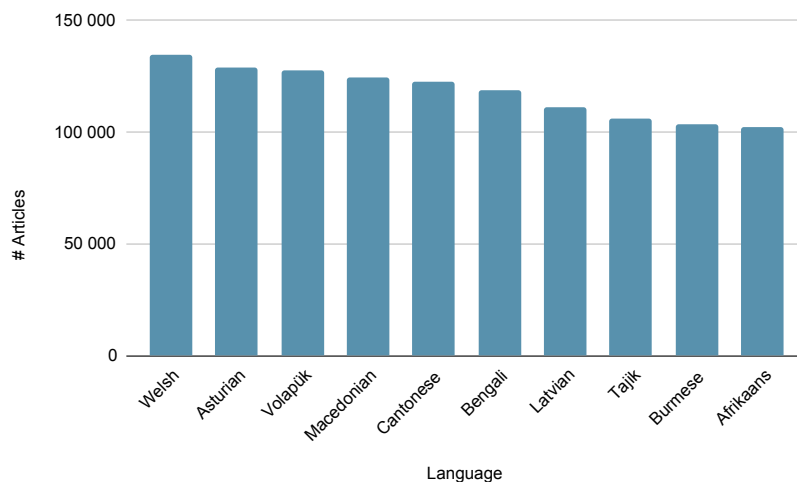


Figure 1.3: Top 10 smallest Wikipedias (over 100k) by number of articles (Data extracted from Wikipedia [91])

To better monitor and help maintain the quality of the website, the English Wikipedia uses a quality scale that aims to rate articles within one of 9 possible grades, which go from incomplete documents (Starts and Stubs) to comprehensive, well-written articles (Featured Articles) [93]. Table 1.1 lists the number of articles from each quality class across the English Wikipedia and allows us to conclude that quality varies considerably, even within the English articles. Notably, the majority of English Wikipedia consists of lower-quality articles, with Starts and Stubs accounting for more than 80% of its content. In comparison, the share of Featured Articles and Good Articles is around 0.6% [93].

Nonetheless, these values are not meant to be taken as official ratings but instead for internal use by the contributors. Besides, not every English article is rated, and the non-English versions of Wikipedia that also assess their content will have different quality scales, evaluated with other criteria.

1.3 Objectives

Our end goal is to provide Wikipedia readers and contributors with a more reliable and transparent assessment of article quality within Wikipedia. We aim to create a Machine Learning model that automatically predicts the quality of any Wikipedia article and apply that model in a Google Chrome extension so that any user can easily take advantage of it. This study is also heavily focused on investigating existing approaches for similar purposes, contributing to research in this field.

That way, Wikipedia readers will know beforehand if they face an excellent or a poor article. At the same time, contributors will quickly notice if an article is flawed and have insight into improving it. Chevalier et al. [20] propose a solution that displays five simple quality metrics to

Table 1.1: Wikipedia Quality Scale: English Articles per each class (Data extracted from Wikipedia [93])

Quality	Description	# Articles
FA	Featured Articles (FA) are considered the best of the best on Wikipedia. Must be well-written, comprehensive, well-researched, and rigorously follow Wikipedia's style norms. They are peer-reviewed from a list of candidates and require a thorough examination to attain this status.	7,650
FL	Featured Lists (FL) are equivalent to FA in terms of quality but usually focus on enumerating other Wikipedia articles.	2,194
A	These articles are detailed and well-structured and were also reviewed independently. Only minor adjustments must be considered FA, so few articles remain in this category for long.	2,284
GA	Good Articles (GA) are too examined by impartial reviewers and must be previously nominated for the category. They are well-written, verifiable, and address the underlying topic appropriately. Most readers would notice no issues in a GA, but it may require some editing before reaching FA status.	40,217
B	Decent quality article, but more serious researchers might desire a more extensive coverage of the topic.	155,896
C	C-Class articles cite a few reliable sources and have style and structure, so they are useful for casual readers. However, they do not meet all criteria for achieving B status, as they contain irrelevant material or lack important content.	460,378
Start	Readers will have trouble finding enough content in a Start article since they are quite incomplete. To improve quality, editors should add more content, cite reliable sources, and improve the article style.	2,168,809
Stub	The worst class of Wikipedia articles is likely an incomplete description of the topic. However, some larger articles can also fall into this category if deemed irrelevant or incomprehensible.	3,712,341
List	Contains a list of links to other Wikipedia articles, often within a specific category.	299,004
None	Unassessed article.	416,629
Total	—	7,265,402

the user, such as the number of links and word count. Still, it is up to the reader to analyze those measures by themselves, which is a step we aim to remove from the equation.

1.4 Information Quality

Before describing how we will approach this problem, we must first answer: *What is quality? And can it be objectively quantified?*. Information Quality (IQ) is an extraordinarily researched topic, and there exist numerous attempts to provide a way to calculate it. Lee et al. [50] attempt to systematize the measurement of Information Quality by breaking it down into 15 properties, each

with its own set of questions. Those properties include concepts such as *Accessibility*, *Believability*, *Interpretability*, *Objectivity*, *Reputation*, and *Timeliness*. Most of those are aspects we plan to assess in our work. However, it is not the case for properties that relate to the integrity of the information, like objectivity, as that would be more suitable for manual approaches. Regardless, there might be some correlation between some quality properties and others. Due to Wikipedia’s collaborative aspect, contributors will, in theory, cooperate to improve the article continuously, so it is reasonable to assume that well-written articles will be trustworthy more often than not.

Wikipedia has its own definition of quality, though. According to them [93], the most outstanding articles must be well-written, comprehensive, well-researched, and follow their style guidelines². These criteria are more subjective to evaluate but still relate to the IQ properties defined by Lee et al. [50], showing that different quality definitions often refer to the same ideas.

In the end, a quality article will be one that surpasses the reader’s expectations regarding how it presents its information. Therefore, in a way, there is no truly objective way to quantify that. However, there are undoubtedly measurable characteristics that people often relate to outstanding quality, and we plan to determine them and their correlation with excellence in written documents.

1.5 Contributions

Our main contribution is the creation of the Google Chrome extension we discussed. We made the software available through the website published in <https://up201705208.wixsite.com/wikiquality>, which briefly explains the context of the application and its uses.

We also wrote two scientific articles, both currently under review by different journals. We submitted the first article, “Automatic Quality Assessment of Wikipedia Articles - A Systematic Literature Review”, to ACM Computing Surveys³. That paper details the methodology and results of the systematic literature review we conducted for our study. We also submitted a research paper to the Journal of the Association for Information Science and Technology (JASIST)⁴, explaining the machine learning methodology, the results we achieved and the extension we created. That paper is entitled “Real-time prediction of Wikipedia articles’ quality”.

We produced a research dataset for each paper, detailing the data we generated throughout the study. The dataset containing the results of the review “Automatic Quality Assessment of Wikipedia Articles - A Systematic Literature Review” is temporarily available at <https://tinyurl.com/wikiquality-data-1>. The repository details the publications we reviewed during the state-of-the-art analysis. The dataset containing the results and other data produced for “Real-time prediction of Wikipedia articles’ quality” is available at <https://doi.org/10.25747/2rdd-rc08>.

We provide this project’s source code in the Github repository accessible from <https://github.com/MOAAAS/WikiQuality>.

²Wikipedia style guidelines available in https://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style/

³ACM Computing Surveys website available at <https://dl.acm.org/journal/csur>

⁴JASIST’s website is available at <https://asistdl.onlinelibrary.wiley.com/journal/23301643>

1.6 Document Structure

We divided this document into seven chapters. After the introduction in Chapter 1, Chapter 2 explains basic machine learning concepts relevant to this study, exemplifying some of the most common algorithms. Chapter 3 details a literature review that aims to collect previously used approaches for automatically predicting article quality in Wikipedia, including document features, datasets, and ML algorithms. We detail the methodology of our machine learning process in Chapter 4, then describe its results in Chapter 5. We dedicate Chapter 6 to explaining the Chrome extension we developed, its architecture, and its functionalities. Chapter 7 concludes our work by clarifying its limitations and possible research paths.

Chapter 2

Machine Learning Background

Machine Learning is a sub-field of Artificial Intelligence (AI) that, through the combination of data and statistical methods, trains models to make classifications or predictions while allowing the user to understand the underlying data [34].

This chapter discusses essential notions regarding Machine Learning, such as the distinction between Supervised and Unsupervised Learning, the concepts of Classification and Regression, and the dangers of Overfitting and Underfitting. We also describe some of the most common ML algorithms while detailing some typically used metrics to evaluate model performance. Finally, we explain the concept of article features, how they relate to Machine Learning, and how we can use them to predict quality in Wikipedia.

2.1 Supervised Learning vs. Unsupervised Learning

Machine learning algorithms are often distinguished by their category. When dealing with a classification or regression problem, that is, when our goal is to predict a value from a certain number of variables, the most common approach is *Supervised Learning*. Some machine learning classifiers also fall into other categories, such as *Unsupervised Learning*, which attempts to discover patterns in unlabelled data, *Semi-supervised*, which is a hybrid of both approaches, and *Reinforcement Learning*, that learns a model by developing the action policy that maximizes the cumulative reward (incentive).

Supervised Learning algorithms all have the same goal: to learn a function (the *model*) that estimates an output based on one or more inputs [46]. It does so by adapting the function in accordance to a set of labeled data (*training data*), intending to maximize the prediction performance against a collection of unlabelled data (*test data*). Using more specific notation, a training *dataset* of n samples and m variables will consist of a $n \times m$ matrix, usually called X , that represents the samples,

$$X = \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1m} \\ X_{21} & X_{22} & \dots & X_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \dots & X_{nm} \end{bmatrix}$$

An observation X_i can be considered as a vector of variables, that is,

$$X_i = \begin{pmatrix} X_{i1} \\ X_{i2} \\ \vdots \\ X_{in} \end{pmatrix}$$

and a vector of length n , usually called Y , which contains the correct label for each observation,

$$Y = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}$$

The testing data will have the same variables but will not include Y , as that is what the algorithm will try to predict. The learning process involves the application of specialized algorithms for estimating F , the function (model) that allows us to predict Y from X ,

$$F(X_i) = Y_i$$

where X_i is a random sample, and Y_i is its correct label. Figures 2.1 and 2.2 show a simplified comparison between Supervised and Unsupervised learning and what kind of data they take as input.

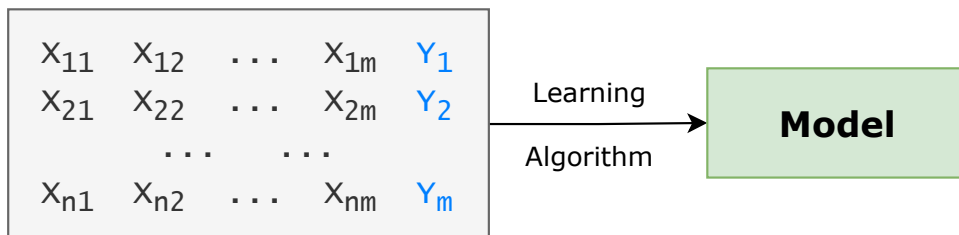


Figure 2.1: Learning diagram for Supervised Learning

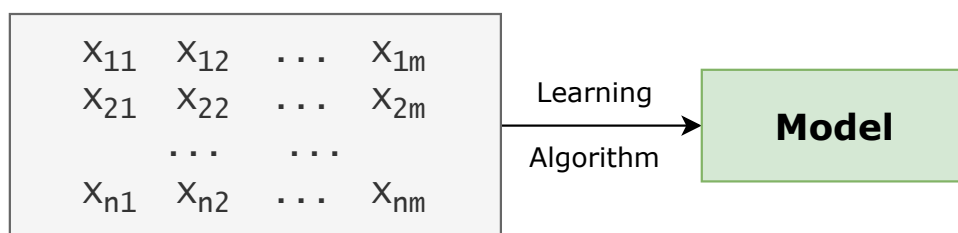


Figure 2.2: Learning diagram for Unsupervised Learning

2.2 Classification vs. Regression

Supervised Learning approaches can solve two types of tasks: *classification* and *regression*. The difference between the two is quite simple: in classification problems, F will output a class or label, that is, one of N possible values, whereas Regression problems have the goal of predicting a number. Mathematically, classification functions have a non-infinite, discrete range, while the range of regression functions is generally continuous.

This study aims to predict the quality of Wikipedia articles by analyzing previously assessed ones, so it is an application of Supervised Learning. Both classification and regression are possible solutions for this type of task. We can develop a Machine Learning model that predicts, for instance, an article's grade (A-F), which would be a classifier. Nonetheless, we could still apply a regression by having the function estimate a value for the quality (e.g., range of $[0 - 100]$).

2.3 Performance and Error Measures

When training a Machine Learning model, we often evaluate its performance according to one or more metrics to efficiently determine how accurately it predicts the results. Here, we describe some of the most common performance measures used for Supervised Learning.

The simplest form of a classification task is *binary* classification, in which the output can belong to one of two classes (e.g., Bankruptcy prediction [57]), often called positive and negative class, but that is frequently not the case. Sometimes, multiple classes are available, making it a *multi-class* task. When it comes to predicting article quality in Wikipedia, both approaches are possible depending on how we define the problem. Most solutions follow a multi-class perspective, such as Dang and Ignat's [29]. Still, there are also binary classification projects, such as Blumenstock's [14], which develops a model that distinguishes between good and bad articles.

Performance metrics in binary classification are usually based on four basic values: number of examples correctly recognized as the positive class (*true positives*, tp), number of examples correctly recognized as the negative class (*true negatives*, tn), and examples that either were incorrectly assigned to the positive class (*false positives*, fp) or the negative class (*false negatives*, fn) [77]. In multi-class problems, the metrics instead consider the tp_i, tn_i, fp_i, fn_i for each class C_i . Table 2.1 explains some of the most common performance measures for classification tasks.

Table 2.1: Performance measures for Binary Classifiers

Measure	Formula	Objective
Accuracy	$\frac{tp+tn}{tp+tn+fp+fn}$	Overall effectiveness
Precision	$\frac{tp}{tp+fp}$	Accuracy of the positive labels
Recall	$\frac{tp}{tp+fn}$	Ability to identify positive examples
Specificity	$\frac{tn}{fp+tn}$	Ability to identify negative examples
F1-Score	$2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$	Harmonic mean of the precision and recall
AUC	$\frac{1}{2}(\text{recall} + \text{specificity})$	Probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance [35]

Regression methods generally use different metrics for assessing performance. Since regressors output a value directly, it is more effective to measure how different the predicted value is from the correct one. Botchkarev [16] determined that the most frequent metrics in research studies are Mean Square Error (MSE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). One metric which is also very common for regression analysis is the Coefficient of Determination (R^2), which measures the “proportion of variation of the mean $\bar{Y}$ explained by the regression” [19]. That is, R^2 shows how effectively the features explain the variation of the values. The formulas for each metric are described in Table 2.2, and consider the following values: n (number of data points), Y_i (correct value for point i) $\hat{Y}_i$ (predicted value for point i), and $\bar{Y}$ (mean of predicted values) [52].

Table 2.2: Error metrics for Regression Models

Measure	Formula	Ideal Value
MSE	$\frac{1}{n} \sum_i (Y_i - \hat{Y}_i)^2$	0
$RMSE$	$\sqrt{MSE}$	0
MAE	$\frac{1}{n} \sum_i Y_i - \hat{Y}_i $	0
$MAPE$	$\frac{100}{n} \sum_i \left \frac{Y_i - \hat{Y}_i}{Y_i} \right $	0 %
R^2	$1 - \frac{\sum_i (Y_i - \hat{Y}_i)^2}{\sum_i (Y_i - \bar{Y})^2}$	1

2.4 Bias-Variance Trade-off and Overfitting

In Machine Learning, we inevitably come across the terms *bias* and *variance*. They are important because they directly influence the reducible error, as described in Equation 2.1 [46]. The reducible error is the error within the estimation of the function, which can be reduced by improving the

learning method. However, even with a perfect function, there is also the error induced by the feature set (i.e., unmeasured input variables), which is irreducible.

$$\text{Error} = \text{Variance} + \text{Bias}^2 + \text{IrreducibleError} \quad (2.1)$$

The trade-off issue derives from the fact that the variance and bias, both non-negative values, are inversely correlated [61]. Such a conclusion is intuitive: simplifying the model's assumptions (i.e., increasing bias) will make the model less sensitive to fluctuations of the training data (i.e., variance decreases) and vice-versa. Overall, “the more complex we make the model $\hat{f}$, the lower the (squared) bias and the higher the variance.”, as described by Hastie et al. [42] and shown in Figure 2.3.

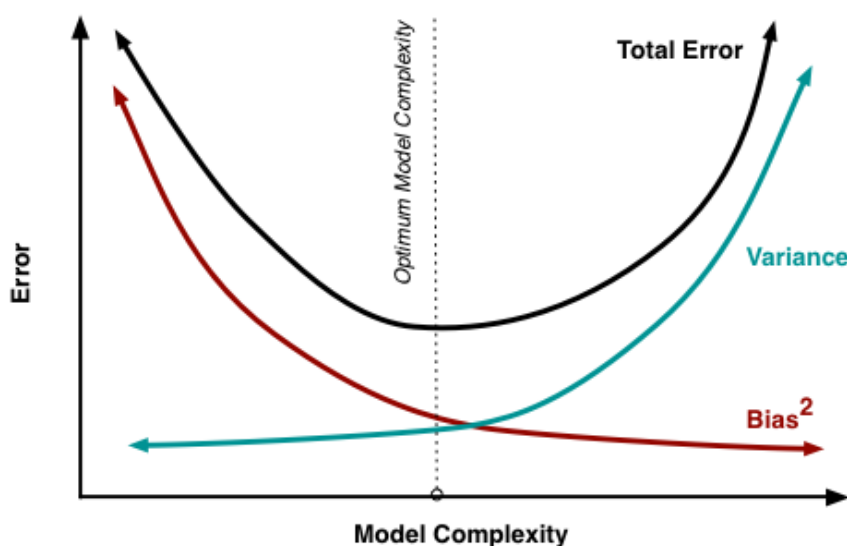


Figure 2.3: Bias-Variance trade-off illustrated [38]

Bias and variance are closely related to the problem of *overfitting* and *underfitting*. Over-decreasing the model's bias will make it follow the training data too closely, leading to a small training error. Although that may appear desirable, the function will unavoidably follow the noise in the training data and probably perform worse in real-world (testing) data. When that happens, we claim the model *overfit* the data. Conversely, if the bias is too large, the trained model will be too simple to predict any real-world information, again generalizing poorly. That phenomenon is called *underfitting* [42]. Figure 2.4 summarizes these concepts by comparing three different models trained on a simple dataset.

The goal is thus to select the algorithm and corresponding parameters that minimize the overall error. We want the optimal amount of bias and variance in our model so that their sum yields the least possible testing error.

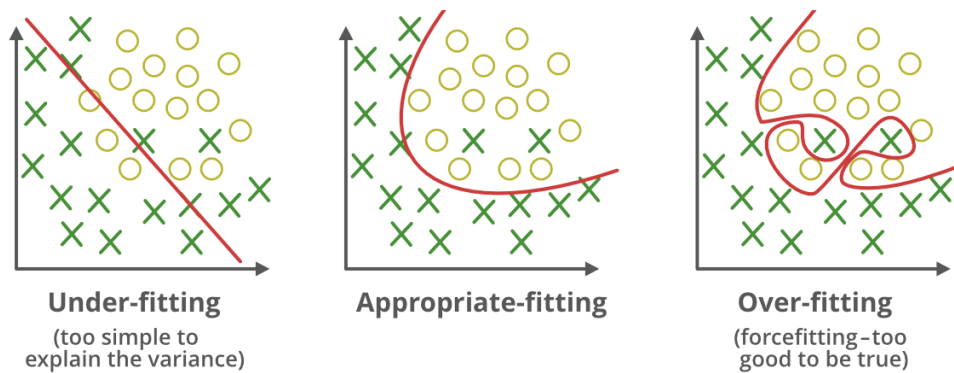


Figure 2.4: Overfitting and Underfitting (Adapted from GeeksForGeeks [60])

2.5 Supervised Learning Algorithms

There are many distinct ways to solve a Machine Learning problem. There exist numerous categories of ML algorithms, each with its pros and cons [75]. There is no silver bullet for these problems, though some methods may perform better than others in different scenarios. Therefore, it is often best to compare the accuracy of several models every time we approach a new problem. That process is called *model selection*, which Hastie et al. define as “estimating the performance of different models in order to choose the best one” [42].

This section lists some of the most commonly used algorithms for supervised learning, all of which can be applied to our goal. Nevertheless, only practical experimentation will determine which alternative produces the best results.

2.5.1 Decision Trees

Decision tree algorithms segment the feature space into a set of regions. We obtain the value of each region from the training data points that belong to that region (e.g., their mean or mode) [46]. They are called decision tree methods because we can represent all the splitting conditions within a tree diagram. Figure 2.5 displays a possible region partition and its respective decision tree diagram.

Classification and Regression Trees (CART) and C5.0 are two commonly used decision tree algorithms in Machine Learning, but their implementations do not differ significantly and often achieve similar results [4]. As implied by the acronym CART, we can use them to solve both classification and regression problems. Tree classifiers and regressors work similarly: the main difference is that, in regression approaches, the leaf nodes are not necessarily part of the training dataset. Instead, their values will be an interpolation of the values of multiple training items.

Although decision trees are simple to represent and interpret, they bring their limitations. According to Hastie et al., a significant problem “is their high variance. Often a small change in the data can result in a very different series of splits, making interpretation somewhat precarious” [42]. As we discussed, a high variance translates into a lower bias, so they can often overfit the training

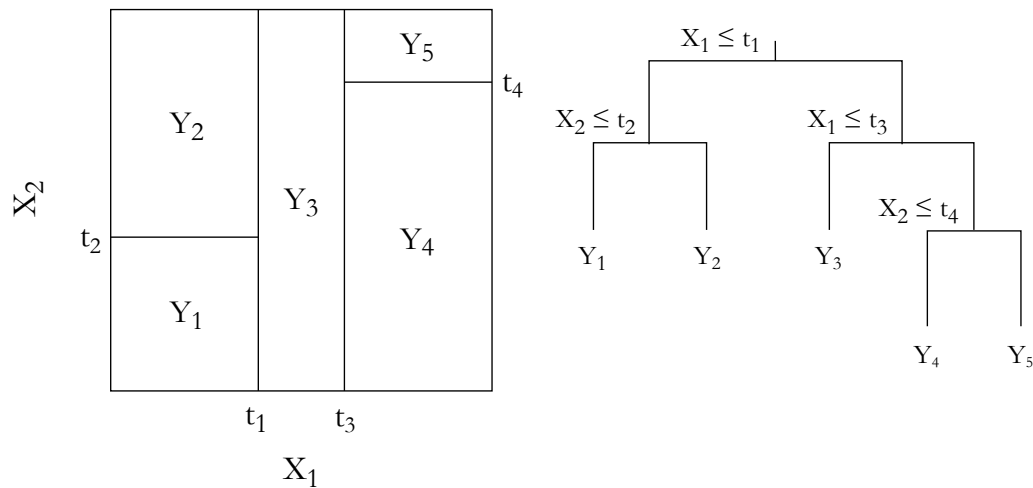


Figure 2.5: Feature space partition and respective decision tree. This dataset contains two variables (X_1 and X_2) and 5 possible labels (Y_1 to Y_5) (Adapted from James et al. [46])

data if we do not take appropriate precautions [75]. Using a larger sample size or pruning certain tree branches might mitigate that risk, improving generalization to unseen data [59].

2.5.2 K-Nearest Neighbors

The K-Nearest Neighbors (KNN) algorithm requires no training step. Instead, to predict the label for a specific data point X , it first calculates the K points in the training data closest to X , i.e., its neighborhood. The closeness metric may vary, but a standard metric is the Euclidean distance, as it is intuitive and straightforward to calculate [42].

The performance of KNN is heavily affected by noise and very dependent on the quality of the chosen K [75]. A low value will make the decision boundary overly flexible, creating a model with a very high variance and low bias. However, increasing the value of K too much makes the decision boundary close to linear [46]. Figure 2.6 shows this effect, comparing the results between a binary classifier using $K = 1$ and another with $K = 100$.

This algorithm can also be used for regression. After identifying the neighborhood of X , one can combine its values by calculating, for instance, their mean [5].

2.5.3 Support Vector Machines

The most basic form of Support Vector Machine (SVM) classifiers is targeted for binary classification. Its objective is to find an equation for a hyperplane that maximizes its distance to the nearest example (the margin) [2]. Still, we can extend the SVM to work with multi-classification and regression problems [46]. We can call them Support Vector Classifiers (SVC) in classification problems, whereas the term Support Vector Regression (SVR) is appropriate in regression.

In most cases, it is impossible to find an equation separating all the elements between the two classes. Hence, we use a cost parameter C , which allows for *some* margin to be violated by the training data. Increasing the cost makes the margin wider and more flexible [46]. It is crucial

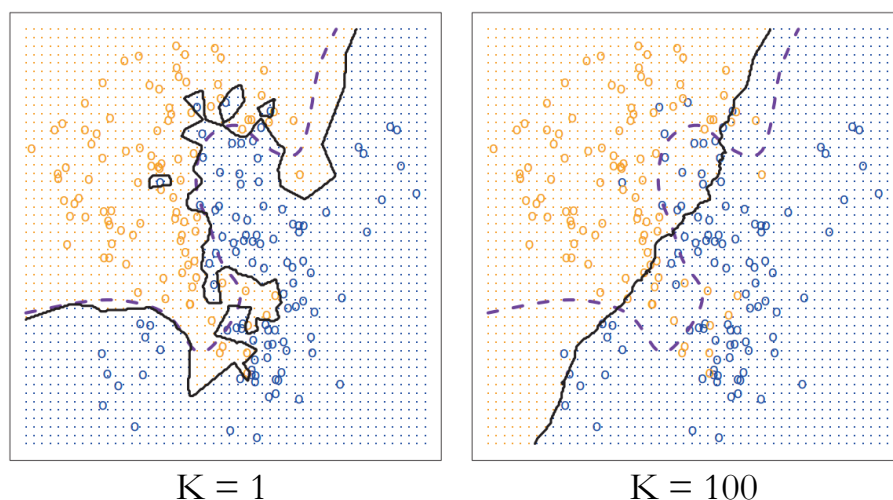


Figure 2.6: Comparison between KNN decision boundaries (solid black curves) using different values of K . The dashed line represents the optimal boundary. (Extracted from James et al. [46])

to tune this parameter during training, experimenting with different values to determine which produces better results. Figure 2.7 demonstrates the adverse effects occurring from improperly chosen cost parameters.

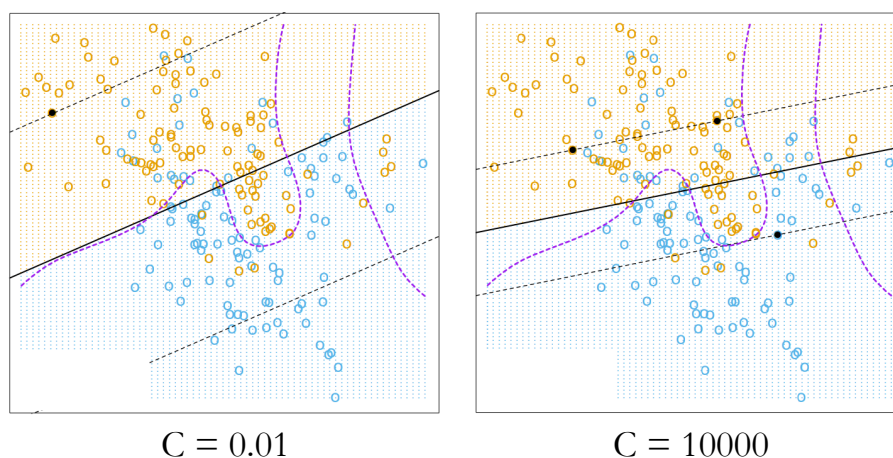


Figure 2.7: Comparison between SVM margins (dashed black lines) using different values of C on a binary classification problem. The dashed purple line represents the optimal boundary. (Extracted from Hastie et al. [42])

Although SVMs are robust against overfitting, their complexity leads to long training times, which is still undesirable [75].

2.5.4 Naive-Bayes

Naive-Bayes (NB) is a Bayesian Network classifier [75] that first calculates the conditional probability of each attribute X_i given each class y and then predicts the label with the highest posterior

probability [39]. For every X_i and y , the algorithm calculates $P(X_i|y)$ and uses the Bayes Theorem to compute $P(y|X)$ for each possible label, as displayed in Equation 2.2 [63].

$$P(y|x_1, \dots, x_n) = \frac{P(x_1|y) \times \dots \times P(x_n|y) \times P(y)}{P(x_1) \times \dots \times P(x_n)} \quad (2.2)$$

This algorithm makes a strong assumption, though: for each class C , each attribute must be independent, that is, be unrelated to the other features. Although the premise may seem unrealistic, NB often outperforms existing state-of-the-art approaches [42].

2.5.5 Neural Networks

Neural Networks (NNs), also called Artificial Neural Networks (ANNs), obtained their name from the way they mimic the human brain and how our neurons signal each other [33]. There are several types of ANNs, but Multi-layer Perceptrons (MLPs) are the most common. Generally, they are comprised of an input layer with I nodes (neurons), an output layer with O nodes, and zero or more hidden layers, each of them with its number of nodes [1]. Nodes from one layer will connect to nodes from the subsequent layer, and each connection will have an associated weight w_{ij} . I is usually the number of input features, and O is the number of classes in classification problems (in regression, we typically use only one output neuron for the result). Figure 2.8 displays a simplified diagram of an Artificial Neural Network.

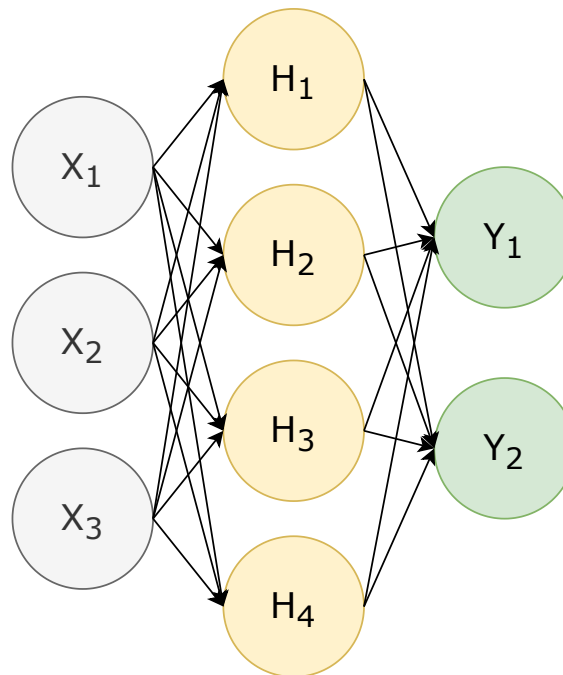


Figure 2.8: Diagram of Neural Network with three input variables, one hidden layer with four neurons, and two output classes. Each line represents a connection between two neurons, and its associated weight must be learned.

Learning algorithms aim to determine the values for all weights w_{ij} that minimize a pre-determined loss function, using a back-propagation algorithm [40]. Although NNs are robust to noise and can map non-linear functions, deeper networks are often slow to train. Additionally, they are frequently considered black boxes because their results are hard to interpret [75].

2.5.6 Linear Regression

Linear Regression is a regression algorithm based on the assumption that a set of variables X is linearly correlated to a quantitative response Y [46]. Mathematically, assuming an N -dimensional feature space, it estimates the values of $\beta_0, \beta_1, \dots, \beta_N$, that minimize the error ϵ in the equation:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n + \epsilon$$

Although it is resistant to overfitting and is an intuitive approach to complex problems, the linear assumption may be too powerful for many real-world cases. Thus, other methods may be more appropriate for practical applications [68].

2.5.7 Logistic Regression

Despite its name, Logistic Regression is mainly used to solve binary classification problems. Instead of predicting which outcome will occur, it uses regression to predict the probability of it happening [44]. We can also use it to solve problems with more than two outcomes, though, by computing the probability of each of them occurring.

2.5.8 Ensemble Methods

The intuition behind ensemble learning is to build a complex model by combining the best parts of simpler base models [42]. There are two main subcategories of ensemble algorithms: Boosting and Bagging. Boosting algorithms, such as AdaBoost and Gradient Boosting, sequentially apply different classifiers to improve the solution iteratively. On the other hand, bagging algorithms train distinct models in parallel, then combine their predictions [53]. A prevalent example of a bagging algorithm is the Random Forest (RF), where a set of decision trees is trained, and they all cast one vote for the predicted class. This algorithm outperforms all others in many classification problems, being fast and impervious to noise in the training data [75].

2.6 Document Features

Independently of the algorithm we pick, we must define a variable set for our training and test data so that the model can learn to map it to the label (the article quality). The variables will be the only information the model can use to predict the result, so, ideally, they should have *some* sort of correlation with the article quality, whether it be large, small, positive, or negative.

The most straightforward solution would be to use the entire raw text as a single variable, feeding it to the learning algorithm (Figure 2.9). However, although it could appear as complete as possible, such an approach would have many limitations. For starters, the raw text of Wikipedia articles is usually lengthy, and training a model with such a large dataset would be computationally expensive with long training times. Other issues may also occur. For instance, the entire text of an article will contain a lot of noise, as it is unlikely that every word will have a non-negligible impact on the quality. Also, other vital aspects that indicate the article is of good quality could be missing, for instance, who contributes to the article or their references [58]. Nonetheless, some authors have achieved promising results by using the raw text of Wikipedia articles, such as Dang & Ignat [29]. Unfortunately, their Neural Network-based method (LSTM [43]) has a predicting time “too long to be implemented as a real-time service”, so we must opt for a more suitable alternative.

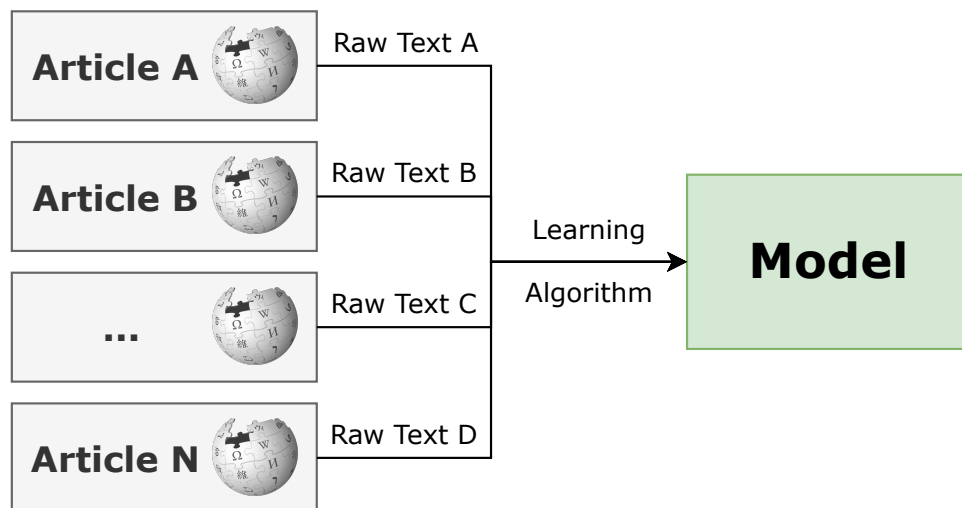


Figure 2.9: Learning with the raw text

A more appropriate approach to this problem is to use the concept of **Document Features**. Their goal is to transform a natural language document (*unstructured data*) into a set of features that may be used to represent the document as a whole [36]. In the case of Wikipedia articles, those can range from a simple character count to the number of words exceeding four syllables or the number of anonymous contributions per week. This solution significantly reduces the complexity of the variable set, which makes the learning process more efficient, improving performance.

Nonetheless, one must not make the tempting mistake of generating too many features, or the data will become increasingly sparse, which was one of the issues with training the model with the raw article text. According to Feldman (2007), “Problems relating to high feature dimensionality (i.e., the size and scale of possible combinations of feature values for data) are typically of much greater magnitude in text mining systems than in classic data mining systems.” [36]. For that reason, one of our goals will be to find the optimal feature set, i.e., the set of features that yields the best model performance.

Even with an optimal variable set, it is unlikely any model could predict article quality with total reliability for a couple of reasons. First, Machine Learning algorithms rarely find the function that optimally maps all sets of variables to their correct labels, only an estimation. Also, there are always implicit factors that cannot be quantified but still have an impact on the quality, such as the context of the article. These issues respectively relate to the concepts of reducible and irreducible error which [46] we described in Section 2.4. In a machine learning problem such as this study's, the goal is to minimize the irreducible error by defining the most appropriate input features and minimize the reducible error by selecting the best learning algorithm and respective parameters. Figures 2.10 and 2.11 summarize the machine learning process by illustrating how it would be applied to the problem of predicting Wikipedia quality.



Figure 2.10: A comprehensive diagram of the learning process with a feature set

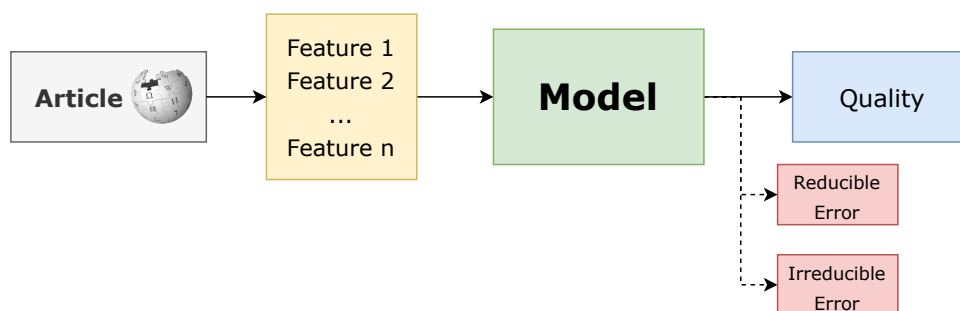


Figure 2.11: Usage of a model trained with a feature set

Still, even with the perfect feature set and the ideal ML algorithm, we would be dependent on the training data being correct. Whether we label it ourselves or trust Wikipedia's quality scale, it will always be subject to human error, often significant. Regardless, it is not a big issue. Real-life models do not usually reach 100% accuracy, as they are attempts to generalize all possible cases using a limited amount of examples and variables.

Chapter 3

Literature Review

We conducted a proper systematic literature review to assess the state-of-the-art within this topic. Our main objective was to examine existing features used to automatically measure Wikipedia article quality and analyze the application of machine learning algorithms for the same purpose.

3.1 Background

We first explore some concepts relating to quality assessment, explaining different possible methods for predicting the quality of Wikipedia articles.

3.1.1 Quality Assessment Methods

There are three general approaches for assessing the quality of an article: Manual, Semi-Automatic (Metric-based), and Automatic, summarized in Figure 3.1. We will explain the difference between them and detail the situations where they are most appropriate.

3.1.1.1 Manual Assessment

The crudest way to assess a Wikipedia article consists of a manual approach, analyzing the articles individually instead of following any specific algorithm or computing any metric. The assessment can be conducted with multiple reviewers, possibly volunteers, reducing human error during the evaluation and thus improving the method's accuracy.

Manual assessment requires no technical skills in automation, so it is common to find such approaches when measuring the quality of articles within a specific field (e.g., Health), as researchers do not always have an extensive programming background. However, this method is naturally time-consuming and extraordinarily difficult to scale up to the entirety of Wikipedia, so it is inadequate for real-time prediction and thus will be mostly disregarded during the literature review.

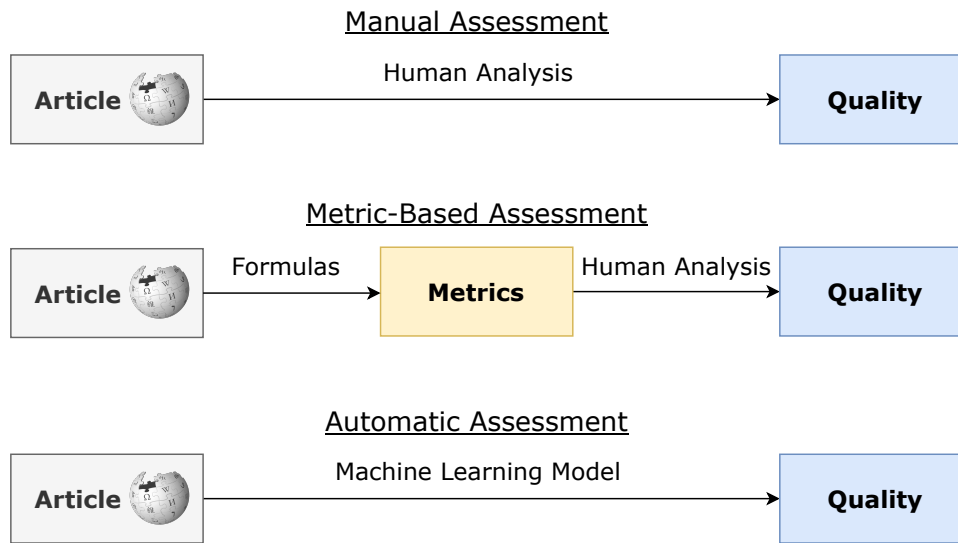


Figure 3.1: Diagrams of the different types of Quality Assessment Methods

3.1.1.2 Metric-based Assessment

There is also a middle-ground approach, which is a semi-automatic assessment of quality that uses predefined formulas to compute metrics that represent quality. Then the results are manually analyzed. These methods are straightforward to implement, providing some level of automation to the quality assessment process. Still, finding a balance between simplicity and accuracy is far from trivial. More straightforward formulas overlook essential quality factors, whereas complex formulas are more challenging to design and their results more complicated to analyze.

Unlike manual approaches, metric-based assessment is more appropriate for real-time quality prediction due to being easily automated. However, since these metrics aim to make the results more understandable, they tend to be too simplified. For this research, they may be somewhat insightful but are not wholly related to our goals, which intend to automate the process entirely.

3.1.1.3 Automatic Assessment

Automatic methods will be the main focus of this literature review, as they intend to fully automate the quality assessment process using Machine Learning or other equivalent approaches. Although much more complex to develop, they eliminate the need for human intervention, making the calculation of quality trivial for any user.

3.1.2 Wikipedia Network Graph

We can view Wikipedia as an “interconnected network of articles”, where each entry can link to other Wikipedia articles. For that reason, we can consider a collection of hyperlinked pages as a directed graph, where the nodes represent the articles and edges indicate the existence of a link between them [9].

Some of the collected features tend to be named according to graph terminology, so we define some crucial concepts below, based on the illustration in Figure 3.2.

- **In-degree:** Number of incoming edges. In our example, $In-degree_A = 2$
- **Out-degree:** Number of outgoing edges. In our example, $Out-degree_A = 3$
- **Assortativity:** Ratio between the (in/out)-degree of the node and the average (in/out)-degree of its neighbors, which are the nodes with outgoing edges to or incoming edges from the considered node. Thus, there are 4 types of Assortativity (in-in, in-out, out-in, out-out). In our example, $AssortativityInIn_A = \frac{2}{0.75} = 2.67$.
- **Local Clustering Coefficient (LCC):** Ratio between the number of edges between neighbors of a node, and the maximum possible number of edges between them. In our example, as the neighbors of A do not have any edges between them, $LCC_A = 0$.
- **Reciprocity:** The reciprocity of a graph is the ratio between the number of reciprocal edges and the total number of edges. In the example, $Reciprocity_A = \frac{1}{4} = 0.25$.

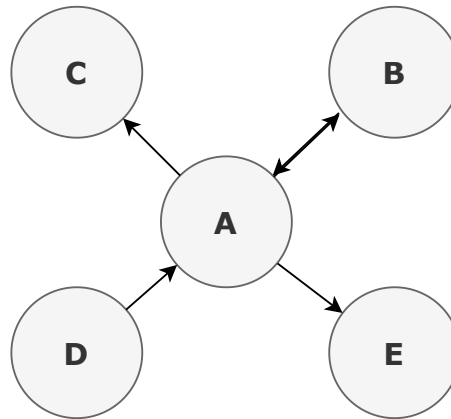


Figure 3.2: Example of a Wikipedia Article’s graph, in which the nodes represent articles, and the edges are the citations between them. In this case, article A cites articles B, C, and E, whereas only articles B and D cite article A. Articles B, C, D, and E do not cite each other.

3.2 Methodology

This section details the methodology used for our systematic literature review.

3.2.1 Research Questions

The systematic review had the goal of answering the following research questions:

RQ1. What are the most common features and metrics used to predict article quality in Wikipedia?

RQ2. To what extent are those features currently used, and how effective are they?

RQ3. Which machine learning approaches can be applied to predict article quality, and how effective are they?

3.2.2 Data Sources

This study included papers from the following online databases:

- Google Scholar
- ACM Digital Library
- Web of Science

We formulated the search queries according to the research questions, intending to scan for publications related to quality assessment within Wikipedia. The queries were deliberately made broad enough to avoid discarding potentially related results.

The search terms for each database are defined in Table 3.1. They were designed to be as similar as possible, to maintain a consistent publication search. Google Scholar's lack of advanced options made refining the search process more complex, as it returned around 18,000 results, most of them unrelated. Therefore, we included only the first 300 results. As for the other databases, the searches returned shorter lists of publications, so all the results could be properly inspected.

Table 3.1: Search queries submitted in each database

Database	Query
Google Scholar	Wikipedia (measuring OR measure) article quality
ACM	[Title: wikipedia assessing quality] AND [[Abstract: metric] OR [Abstract: measure]] AND [Full Text: wikipedia] AND [Full Text: assessing] AND [Full Text: quality] AND [Publication Date: (01/01/2000 TO 12/31/2021)]
Web of Science	wikipedia assessing quality (metric OR measure)

All searches were run on December 2021 and were restricted to a date range of 2000-2021. We picked a lower limit of 2000 because Wikipedia was launched in 2001 [92], so we would unlikely find related articles from an earlier year.

The search among all databases returned 18,520 results, 18,000 of which (Google Scholar) were excluded for the aforementioned reasons.

3.2.3 Selection Process

We guided our selection process according to the PRISMA statement [65, 66], which defines a set of guidelines for conducting systematic literature reviews. Our methodology is divided into 4 phases: Identification, Screening, Eligibility, and Inclusion.

Our Identification phase contained three database searches that returned 18,520 results, 18,000 of which were ignored due to Google Scholar producing many irrelevant results. Figure 3.3 summarizes the search process by comparing the number of results from each information source.

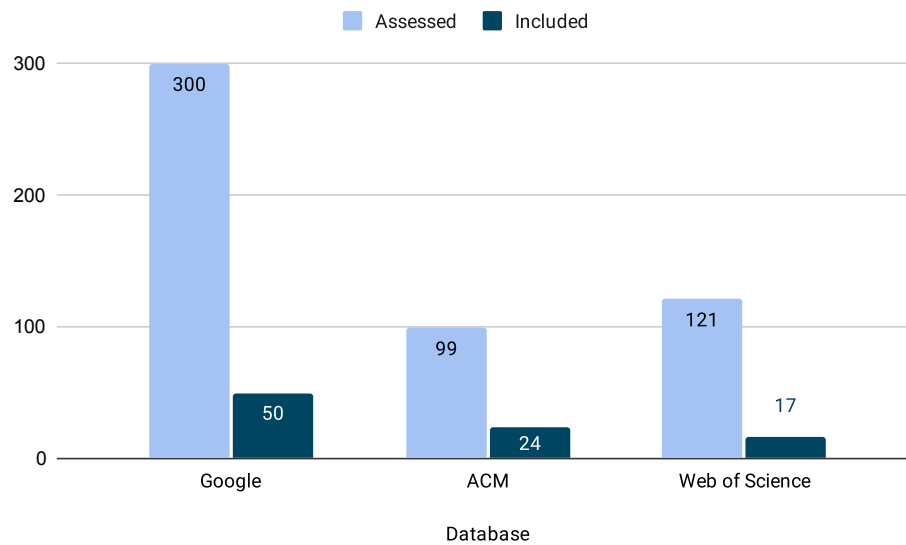


Figure 3.3: Identification phase: Titles assessed vs. Papers included

During the Screening phase, publication titles were first assessed for possible relevance within the research area. Any result that was at least marginally related to the quality assessment of Wikipedia was included in the next phase. From the 520 analyzed titles, 91 were considered relevant for our research. For instance, Nemoto et al.’s study [62] was one of the results our query returned, but we deemed its title unrelated to our research topic: Quality Assessment of Wikipedia. Before advancing to the next stage, results were compared for the existence of duplicates, and 21 repeated publications were excluded. Finally, we scanned the papers’ abstracts, now focusing on the research questions. Only studies that propose automatic methods for measuring Wikipedia quality were included. This phase excluded many publications experimenting with manual quality assessment approaches within specific science sub-fields (e.g., Health). Only 39 studies were included in the next step.

In the Eligibility phase, we considered the inclusion and exclusion criteria we defined. Publications not meeting any inclusion criteria or meeting any exclusion criteria are excluded. These criteria are described in Tables 3.2 and 3.3 and resulted in the exclusion of 17 papers.

Table 3.2: Inclusion Criteria

ID	Criteria
I1	Paper discusses possible features or metrics to assess information quality.
I2	Paper discusses machine learning approaches to predict information quality.

In the Inclusion phase, we assessed the 22 included publications and assigned them a relevance score based on four questions. Those questions are listed in Table 3.4, including the score range for each question. For each question, a 0 indicates a negative response, and the maximum score represents a solid affirmative response. This step was conducted by a single reviewer, which makes

Table 3.3: Exclusion Criteria

ID	Criteria
E1	Paper only discusses approaches to detect vandalism.
E2	Paper only discusses approaches to detect controversy.
E3	Paper discusses manual approaches to assess article quality, as opposed to automatic ones.
E4	Paper is in a language other than English.

it sensitive to human error. However, the assessment did not aim to include or exclude any study from the literature review but to give a better overview of which papers are more related to our research. Q4 is an exception to that goal, though, as its objective is to slightly incentivize the analysis of publications that study other versions of Wikipedia. This phase also served to extract from the papers all the necessary information.

Table 3.4: Relevance Score Questions

ID	Question	Score Range
Q1	Does the paper describe and compare possible features or metrics with detail?	[0, 3]
Q2	Does the paper describe and compare ML approaches with detail?	[0, 3]
Q3	Are the results, benefits and limitations well described?	[0, 3]
Q4	Does the paper bring focus to an article language that's not English?	[0, 1]

The process was designed to minimize the number of relevant publications that get excluded, so we were more lenient on the stages of the Screening phase (e.g., abstract scanning) when deciding whether to include the paper. On the other hand, the Eligibility phase was much stricter in enforcing the inclusion and exclusion criteria.

We included 22 publications in the final step of the selection process. Figure 3.4 outlines this process, detailing the number of publications included in each phase.

3.2.4 Data Collection

For every paper included at the end of the Screening phase, we first collected their title, keywords, abstract, year of publication, author names, journal/conference, and citation count. We immediately assigned them a number ID for easier reference during the review. When including or excluding a result, we also store the reason for doing so.

Most of the information was, however, extracted during the Inclusion phase. As defined in the research questions, our goal is to collect:

1. Features or metrics for measuring article quality in Wikipedia.
2. Machine Learning methods for automatically predicting quality in Wikipedia (datasets and algorithms)
3. Results of the application of both previous items.

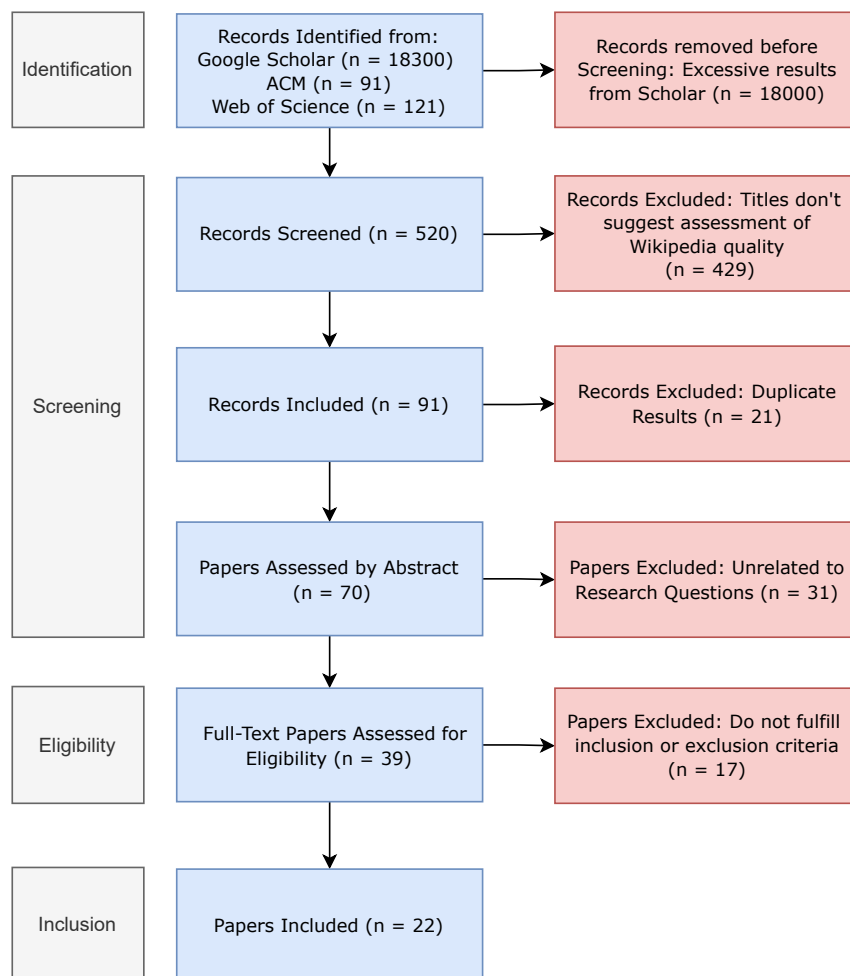


Figure 3.4: Article's Selection Process

These items somewhat relate to the relevance questions defined in Table 3.4. Table 3.5 shows the average scores for Q1, Q2, and Q3, ranging from 0 to 3, among the 22 included articles (in the case of Q4, the percentage of papers with a positive response is displayed instead). We notice that the proposal of features and metrics is often more complete than the comparison of distinct ML algorithms. Regardless, articles tend to analyze and discuss their results thoroughly.

Table 3.5: Average Relevance Scores

ID	Question	Average / %
Q1	Does the paper describe and compare possible features or metrics with detail?	2
Q2	Does the paper describe and compare ML approaches in detail?	1.18
Q3	Are the results, benefits, and limitations well described?	2
Q4	Does the paper bring focus to an article language that's not English?	18.2 %

All the information we collected is available in a research data repository¹. Appendix A also details most of the aforementioned data.

3.3 Overview of Analyzed Articles

This section provides an overview of the 22 included articles, analyzing attributes like their publication venues, citation count, authors, and affiliations. We also perform a keyword analysis using a word cloud method, where we attempt to discover patterns between the papers we collected.

3.3.1 Year of Publication

We thoroughly analyzed the different publications from our initial pool of 520, all published between the years 2000 and 2021. Stvilja's [79] study, from 2005, was the oldest of those 22, and there was not much interest in this subject until 2007. That can also be explained by the fact that Machine Learning was not as extensively researched at the time. Pugliese et al. [67] show us that the number of ML publications in 2006 was almost a third of the 2010s, the year in which the number started to boom. Figure 3.5 shows the trend for researching this topic has fluctuated, spiking around 2009, and recently it has remained steady, after a short break during 2012-2015.

3.3.2 Publication Venue and Number of Citations

Most of the analyzed papers were published in international conferences, and many were highly cited. Table 3.6 contains information regarding the number of citations and where the articles were published, allowing a better overview of the context of each study and their relevance to other studies.

¹While the paper awaits publication, the complete dataset is available at <https://drive.google.com/drive/folders/1tlojqreV98zmHWY1WCgmwuaQwBVcWca?usp=sharing>

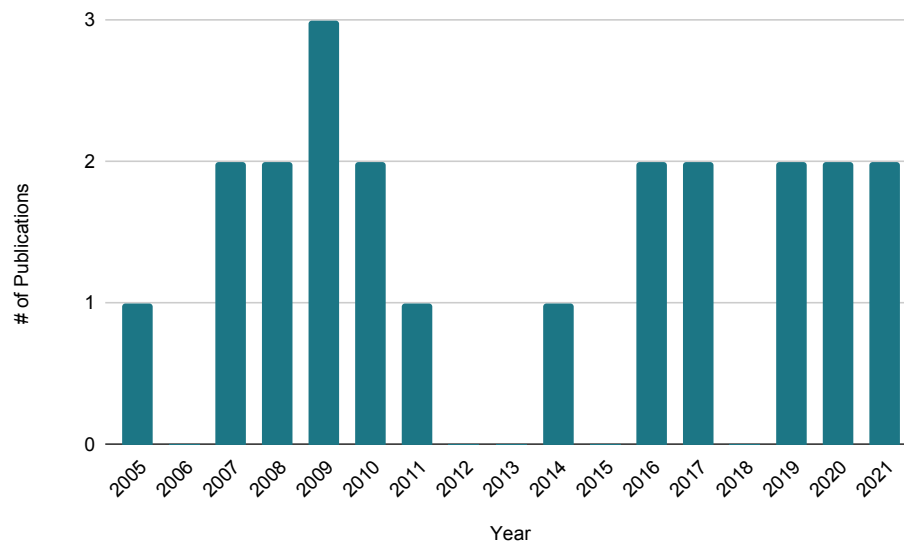


Figure 3.5: Reviewed papers by year of publication

We thus determine that 17 out of 22 articles were presented at a conference, which is a significant majority. Additionally, we can note that five papers were presented at the International World Wide Web Conference (*WWW*), 2 of them at the International Symposium on Wikis and Open Collaboration (*WikiSym*), and 2 of them at the Joint Conference on Digital Libraries (*JCDL*). Other conferences only appeared once, namely the International Symposium on Open Collaboration (*OpenSym*), the World Conference on Information Systems and Technologies (*WorldCIST*), the International Conference on Collaboration and Internet Computing (*CIC*), the ACM Symposium on Applied Computing (*SAC*), the ACM Conference on Hypertext and Hypermedia (*HT*), the International Conference on Management of Emergent Digital EcoSystems (*MEDES*), and the International Conference on Knowledge Management (*ICKM*). The number of citations varied wildly across papers and databases, but over half of the publications were cited more than 30 times when considering Google Scholar’s numbers. The most cited article was Stvilia et al.’s [80], achieving over 500 citations in Google Scholar.

3.3.3 Abstract and Keyword Analysis

We performed an abstract analysis using a word cloud method to obtain insight into the most common terms of both items. All abstracts were merged and later input to a word cloud generator². Figure 3.6 displays a word cloud of all the abstracts we included in our review. We can observe that the terms *Wikipedia articles*, *information quality*, and *quality assessment* are likely increasing the size of those words in the cloud. We also have a few other common terms, such as *feature/features* and *models*, as these papers deal in a Machine Learning context. We applied the same method to the keywords of all the papers, though we instead considered every keyword as

²We used the software provided by <https://www.wordclouds.com>

Table 3.6: Bibliometric overview of reviewed publications

Study	Year	Citations ¹	Type	Published In
Stvilia et al. [79]	2005	393 / X / 202	Conference	ICKM
Stein & Hess [78]	2007	104 / X / 43	Conference	HT '07
Stvilia et al. [80]	2007	569 / 192 / 287	Journal	JASIST
Blumenstock [14]	2008	353 / X / 208	Conference	WWW '08
Druck et al. [32]	2008	63 / X / 26	Workshop	WikiAI Workshop
Dalip et al. [27]	2009	142 / 40 / 84	Conference	JCDL '09
Halfaker et al. [41]	2009	125 / X / 57	Conference	WikiSym '09
Wöhner & Peters [100]	2009	136 / X / 70	Conference	WikiSym '09
Lipka & Stein [51]	2010	115 / X / 63	Conference	WWW '10
De La Calzada [18]	2010	76 / X / 42	Conference	WWW '10
Banerjee et al. [6]	2011	1 / X / 0	Conference	MEDES '11
Saengthongpattana [70]	2014	2 / 1 / 2	Conference	WorldCIST '14
Su & Liu [81]	2016	5 / 2 / 3	Conference	WWW '10
Dang & Ignat [28]	2016	55 / 26 / 41	Conference	JCDL '16
Dang & Ignat [29]	2017	34 / X / 18	Conference	OpenSym '17
Dang & Ignat [30]	2017	41 / 18 / 22	Conference	CIC
Bassani & Viviani [8]	2019	6 / 2 / 4	Conference	SAC '19
Velichety [83]	2019	2 / 0 / 0	Journal	SIGMIS Database
Wang & Li [85]	2020	22 / 8 / 10	Journal	JASIST
Antunes & Lopes [3]	2020	0 / 0 / 0	Conference	SIGIR 2020
Wang et al. [86]	2021	4 / 1 / 1	Journal	Journal of Information Science
Couto & Lopes [24]	2021	0 / 0 / 1	Conference	WWW '21

¹ (Google Scholar / Web of Science / Scopus), X indicates paper is unavailable

a phrase unless that keyword contained either *feature*, *quality*, or *Wikipedia*. The resulting cloud, shown in Figure 3.7, is much smaller but still shows *wikipedia*, *information quality*, and *feature* are terms that often appear in these papers. We can also observe the presence of ML terms, such as *machine learning*, *deep learning*, *models* and *RNN*, and algorithms, like *SVM* and *decision tree*.

3.3.4 Authors and Affiliations

We also decided to measure author presence across this research topic to determine which researchers and affiliations study this subject more often. Table 3.7 summarizes this information, displaying, for each author, their publications and affiliations. Most authors only took part in one of the reviewed papers, but some authors, such as Quang-Vinh Dang, Claudia-Lavinia Ignat, Ping Wang, and Xiaodan Li, show up in multiple of them, which indicates they have a strong presence in automatic quality assessment of Wikipedia.

We also grouped all the affiliation countries to discover possible patterns in that information. Figure 3.8 shows us that the USA is the most common country (40.5%), followed by Germany (11.9%). Many authors are affiliated with European countries (28.57%) and Asian ones (21.45%).

Table 3.7: Authors and their Affiliations

Study	Author Name	Affiliation	Country
[41]	Aaron Halfaker	Wikimedia Foundation, Inc	USA
[18]	Alex Dekhtyar	California Polytechnic	USA
[32]	Andrew McCallum	University of Massachusetts	USA
[41]	Aniket Kittur	Carnegie Mellon University	USA
[51]	Benno Stein	Bauhaus-Universität Weimar	Germany
[79, 80]	Besiki Stvilia	Florida State University	USA
[3, 24]	Carla Teixeira Lopes	University of Porto / INESC TEC	Portugal
[78]	Claudia Hess	University of Bamberg	Germany
[28, 29, 30]	Claudia-Lavinia Ignat	Inria Nancy - Grand-Est	France
[27]	Daniel Hasan Dalip	CEFET-MG	Brazil
[8]	Elias Bassani	University of Milano-Bicocca	Italy
[18]	Gabriel De La Calzada	California Polytechnic	USA
[32]	Gerome Miklau	University of Massachusetts	USA
[32]	Gregory Druck	University of Massachusetts	USA
[6]	Hameed Al-Qaheri	Kuwait University	Kuwait
[3]	Hélder Antunes	University of Porto / INESC TEC	Portugal
[41]	John Riedl	University of Minnesota	USA
[14]	Joshua E. Blumenstock	University of California, Berkeley	USA
[70]	Kanchana Saengthongpattana	NECTEC	Thailand
[78]	Klaus Stein	University of Bamberg	Germany
[79, 80]	Les Gasser	University of Illinois	USA
[79, 80]	Linda C. Smith	University of Illinois	USA
[24]	Luís Couto	University of Porto	Portugal
[27]	Marco Cristo	FUCAPI Research & Innovation	Brazil
[8]	Marco Viviani	University of Milano-Bicocca	Italy
[27]	Marcos André Gonçalves	UFMG	Brazil
[79, 80]	Michael B. Twidale	University of Illinois	USA
[6]	Nashwa El-Bendary	AASTMT	Egypt
[51]	Nedim Lipka	Adobe Systems Inc.	USA
[70]	Nuanwan Soonthornphisaj	Kasetsart University	Thailand
[27]	Pável Calado	Instituto Superior Técnico	Portugal
[81]	Pengyuan Liu	UAL, University of Singapore	Singapore
[85, 86]	Ping Wang	SIM, Wuhan University	China
[81]	Qi Su	Peking University	China
[28, 29, 30]	Quang-Vinh Dang	Industrial University, Ho Chi Minh	Vietnam
[100]	Ralf Peters	MLU (Halle-Wittenberg)	Germany
[86]	Renli Wu	SIM, Wuhan University	China
[41]	Robert Kraut	Carnegie Mellon University	USA
[6]	Soumya Banerjee	Birla Institute of Technology	India
[83]	Srikar Velichety	University of Memphis	USA
[100]	Thomas Wöhner	MLU (Halle-Wittenberg)	Germany
[85, 86]	Xiaodan Li	Duke University ECE Department	USA



Figure 3.6: Word Cloud of article abstracts

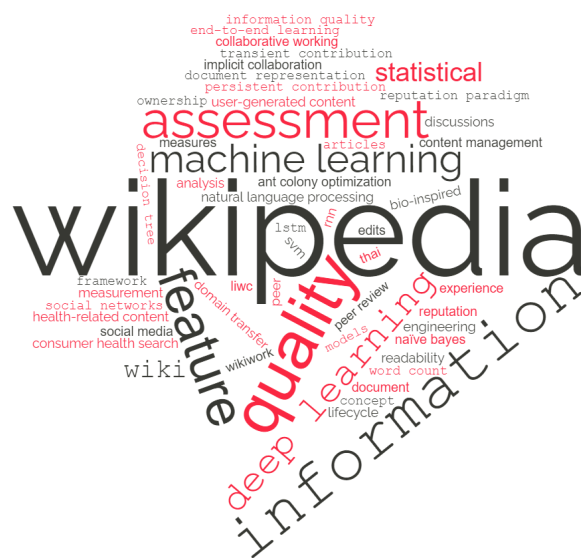


Figure 3.7: Word Cloud of article keywords

3.4 Quality Features and Metrics

All the article features collected from the reviewed publications were assigned a unique ID and are described here, each falling within one of the following categories:

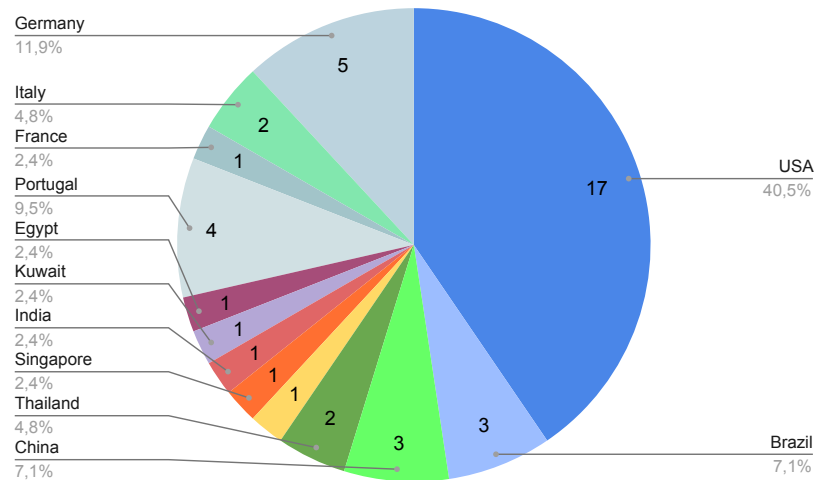


Figure 3.8: Number of authors per affiliation's country

- **Content features**, which relate to the length and structure of the article, taking into account factors such as the number of words, sections, or images.
- **Style features**, that measure how the authors write the articles, how long their phrases are, and what classes of words they use.
- **Readability features** estimate “the age or US grade level necessary to comprehend a text. (...) good articles should be well written, understandable, and free of unnecessary complexity” [27], by measuring the sentence and word complexity. They are characterized by their use of more complex formulas, which combine other types of features.
- **History features**, which analyze the revision history of an article and related factors, like the article's age and the number of contributions.
- **Network features** are a bit more complex, as they take into account the citations between Wikipedia articles to measure their popularity.

Most authors, such as Bassani and Viviani [7], consider Content, Style, and Readability to be all subcategories of *Text Features*, but we will distinguish them as different types in this review.

Throughout the 22 analyzed articles, we collected 165 distinct features. Figure 3.9 better displays the proportion of features per category. Bassani and Viviani's [8] and Dalip et al.'s [27] studies both also perform an extensive literature review of existing features, and therefore it is expected that they amass most of the collected ones.

For the computation of style and readability features, Dalip et al. [27] use the Diction software³.

³<http://www.gnu.org/software/diction/>

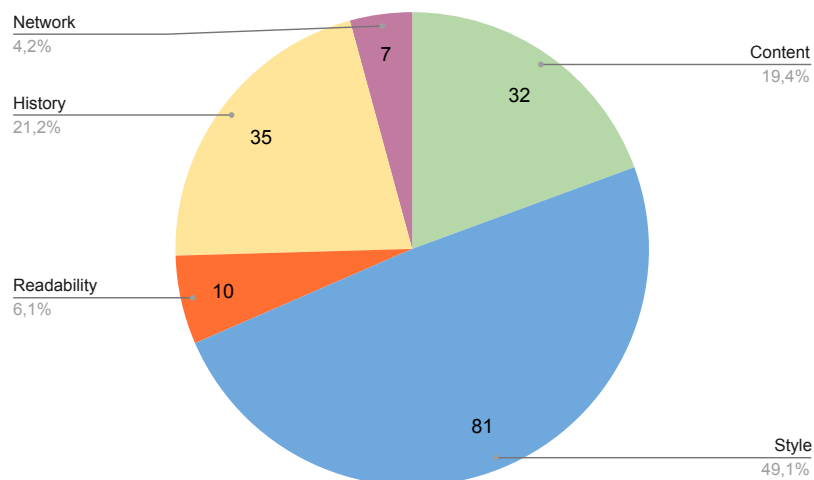


Figure 3.9: Collected Features: Count per Category

3.4.1 Content Features

Content features contain information regarding the length and structure of the article. They are one of the simplest to compute since they can be almost directly derived from the text. Table 3.8 lists all content features gathered from the literature review. As suggested by the table, we associated all the features we collected with a short identifier for easier reference. Every identifier starts with the letter respective to its category, while the other letters represent the feature's name.

Intuitively, there is a correlation between article length and quality. Good articles should not be too long and complex, overwhelming the reader, but not too simple either, as that could signify incomplete information. Also, according to Wikipedia [93], *stub* articles (drafts) are “usually very short”, which also demonstrates that correlation.

Features related to the article structure are also essential to represent quality. Well-written articles should have a clear organization, with a balanced division of the content in sections and paragraphs. Images improve the reading experience, and references tend to increase the credibility of an article, so they are both decent indicators of quality.

3.4.2 Style Features

Style features evaluate how contributors write their sentences, measuring the used word classes and sentence types. Table 3.9 provides the listing of all the style features. The intuition behind these features is more subtle, but the idea is that specific sentence structures are considered of better quality when presenting information about a topic.

Table 3.8: List of Content Features mentioned in the assessed publications

ID	Name	Proposed in
CC	Character Count	[8, 18, 27, 29, 30, 70, 80, 85, 86]
CW	Word Count	[8, 14, 27, 28, 51, 81, 83, 85, 86]
CSN	Sentence Count	[8, 27, 85, 86]
CS	Section Count	[8, 27, 29, 70, 86]
CMSL	Mean Section Length (Characters)	[8, 18, 27, 85]
CP	Paragraph Count	[8]
CMPL	Mean Paragraph Length (Words)	[8, 27, 85, 86]
CLSL	Largest Section Length	[8, 27, 85, 86]
CSSL	Shortest Section Length	[8, 27, 85, 86]
CSTDLSL	Standard Deviation of Section Length	[8, 27, 85, 86]
CLSSR	Longest-Shortest Section Ratio	[8]
CB	Subsection Count	[8, 27, 30, 85, 86]
CBPS	Subsection Count per Section	[8, 27, 85, 86]
CNL	Introduction Length (Characters)	[8, 27, 85, 86]
CNLTLR	Introduction Length-Text Length ratio	[8]
CR	Number of references	[30, 70, 83]
CCC	Citation Count	[8, 27, 85, 86]
CCCPC	Citation Count per Character	[8, 18, 27, 85, 86]
CCCPS	Citation Count per Section	[8, 27, 85, 86]
CEL	External Link Count	[8, 27, 70, 80, 85, 86]
CELPS	External Link Count per Section	[8, 27, 85, 86]
CELPC	External Link Count per Character	[8, 18, 27, 85, 86]
CIL	Internal Link Count	[30, 70, 80, 83]
CILPC	Internal Link Count per Character	[18]
CIBL	Internal Broken Link Count	[80]
CI	Image Count	[8, 27, 70, 80, 83]
CIPC	Images per Character	[8, 18, 30]
CIPS	Images per Section	[8, 27, 85, 86]
CH3	Number of L3 Headings	[30]
CIB	Number of Information Boxes	[30, 70]
CCT	Number of citation templates	[30]
CNCT	Number of non-citation templates	[30]

Table 3.9: List of Style Features mentioned in the assessed publications

ID	Name	Proposed in
SSLS	Largest Sentence Size (words)	[8, 27, 85, 86]
SSSS	Shortest Sentence Size (words)	[8]
SMSS	Mean Sentence Size	[8, 81]
SLSR	Large Sentence Rate (% sentences whose size is 10 words greater than SMSS)	[8, 27, 85, 86]
SSSR	Short Sentence Rate (% sentences whose size is 5 words shorter than SMSS)	[8, 27, 85, 86]
SQ	Question Count	[8, 27, 85, 86]
SQPS	Question Count per Sentence	[8]
SE	Exclamation Count	[8]
SEPS	Exclamation Count per Sentence	[8]
SS	Syllable Count	[8]
SMSPW	Mean # of Syllables per Word	[8]
SMCPW	Mean # of Characters per Word	[8]
SSTP	# of sentences starting with a pronoun	[8, 27, 85, 86]
SSTART	# of sentences starting with an article	[8, 27, 85, 86]
SSTC	# of sentences starting with a coordinate conjunction	[8, 27, 85, 86]
SSTSUB	# of sentences starting with a subordinate preposition or conjunction	[8, 27, 85, 86]
SSTI	# of sentences starting with an interrogative pronoun	[27, 85, 86]
SSTPRE	# of sentences starting with a preposition	[27, 85]
SSTD	# of sentences starting with a determiner	[8]
SSTADJ	# of sentences starting with an adjective	[8]
SSTN	# of sentences starting with a noun	[8]
SSTADV	# of sentences starting with an adverb	[8]
SSTPPS	# of sentences starting with a pronoun per Sentence	[8]
SSTARTPS	# of sentences starting with an article per Sentence	[8]
SSTCPS	# of sentences starting with a coordinate conjunction per Sentence	[8]
SSTSUBPS	# of sentences starting with a subordinate preposition or conjunction per Sentence	[8]
SSTDPS	# of sentences starting with a determiner per Sentence	[8]
SSTADJPS	# of sentences starting with an adjective per Sentence	[8]
SSTNPS	# of sentences starting with a noun per Sentence	[8]
SSTADVPS	# of sentences starting with an adverb per Sentence	[8]
SAV	Auxiliary verb count	[27, 85, 86]
SMV	Modal Verb Count	[8]

Table 3.9: List of Style Features mentioned in the assessed publications

ID	Name	Proposed in
SPV	Passive Voice Count	[8, 27, 85, 86]
STB	“To be” Verb Count	[8]
SUW	Unique Word Count	[8, 81]
SN	Noun Count	[8]
SUN	Unique Noun Count	[8]
SV	Verb Count	[8]
SUV	Unique Verb Count	[8]
SP	Pronoun Count	[8, 27, 85]
SUP	Unique Pronoun Count	[8]
SADJ	Adjective Count	[8]
SUADJ	Unique Adjective Count	[8]
SADV	Adverb Count	[8]
SUADV	Unique Adverb Count	[8]
SCC	Coordinating Conjunction Count	[8]
SUCC	Unique Coordinating Conjunction Count	[8]
SSUB	Subordinating Preposition or Conjunction Count	[8]
SUSUB	Unique Subordinating Preposition or Conjunction Count	[8]
SMVPW	Modal Verb Count per Word	[8]
SPVPW	Passive Voice Count per Word	[8]
STBPW	“To be” Verb Count per Word	[8, 27]
SUWPW	Unique Word Count per Word	[8]
SNPW	Noun Count per Word	[8]
SUNPW	Unique Noun Count per Word	[8]
SVPW	Verb Count per Word	[8]
SUVPW	Unique Verb Count per Word	[8]
SPPW	Pronoun Count per Word	[8]
SUPPW	Unique Pronoun Count per Word	[8]
SADJPW	Adjective Count per Word	[8]
SUADJPW	Unique Adjective Count per Word	[8]
SADVPW	Adverb Count per Word	[8]
SUADVPW	Unique Adverb Count per Word	[8]
SCCPW	Coordinating Conjunction Count per Word	[8, 27, 85, 86]
SUCCPW	Unique Coordinating Conjunction Count per Word	[8]
SSUBPW	Subordinating Preposition or Conjunction Count per Word	[8]
SUSUBPW	Unique Subordinating Preposition or Conjunction Count per Word	[8]
SPREPW	Preposition Count per Word	[27, 85, 86]

Table 3.9: List of Style Features mentioned in the assessed publications

ID	Name	Proposed in
SNOMPW	Nominalization Count per Word	[27, 85, 86]
STBPV	“To be” Verb Count per Verb	[8, 85, 86]
SMVPPV	Modal Verb Count per Passive Voice Count	[8]
SUNPUW	Unique Noun Count per Unique Word	[8]
SUVPUW	Unique Verb Count per Unique Word	[8]
SUPPUW	Unique Pronoun Count per Unique Word	[8]
SUADJPUW	Unique Adjective Count per Unique Word	[8]
SUADVPUW	Unique Adverb Count per Unique Word	[8]
SUCCPUW	Unique Coordinating Conjunction Count per Unique Word	[8]
SUSUBPUW	Unique Subordinating Preposition or Conjunction Count per Unique Word	[8]
SWL6	Words Larger than 6 Letters	[81]
SCHTRI	Top-m most discriminant character trigrams	[8, 51]
SPOSTRI	Top-n most discriminant POS trigrams	[8, 51]

A study by Bassani and Viviani [8] proposes a large amount of very specific, and using all of them could lead to high feature dimensionality. As with the other types of features, experiments must be done to determine the optimal feature set.

3.4.3 Readability Features

Readability features are metrics that assess how easily readable a Wikipedia article is. They are calculated using carefully designed formulas that combine word, syllable, and character count and are listed in Table 3.10. Although they were designed to output a readability score directly, they can still be considered features within the context of a Machine Learning approach. The formulas for each metric are described below, on Equations 3.1 to 3.9.

Automated Readability Index: Estimates readability by combining the average word length with the average sentence size [73]. It aims to estimate the US grade level required to read a text [48]. The formula scales linearly. A score of 1 indicates the text can easily be read by a kindergartener, while only college students will fully understand texts scoring 14 or above. The values in between correspond to grade levels between the first and twelfth grades.

$$ARI = 4.71 \frac{\text{characters}}{\text{words}} + 0.5 \frac{\text{words}}{\text{sentences}} - 21.43 \quad (3.1)$$

Coleman-Liau: Similarly to *ARI*, estimates readability by combining the average word length with the average sentence size. The scores also calculate the text’s grade level [21], therefore, although the results may vary marginally from *ARI*, the results will have the same meaning as that metric.

$$CL = 5.88 \frac{\text{characters}}{\text{words}} - 29.6 \frac{\text{sentences}}{\text{words}} - 15.8 \quad (3.2)$$

Flesch Reading Ease: Using the average sentence size and amount of syllables per word, computes a value between 0 and 100. A score below 60 indicates the text is more challenging to read than “Plain English” (grade level of 8th and 9th-grade students), whereas higher scores suggest better readability [37].

$$FRE = 206.835 - 1.015 \frac{\text{words}}{\text{sentences}} - 84.6 \frac{\text{syllables}}{\text{words}} \quad (3.3)$$

Flesch-Kincaid Score: It is equivalent to *FRE*, but instead aims to provide the US grade level of a text instead of a value between 0 and 100 [48].

$$FK = 0.39 \frac{\text{words}}{\text{sentences}} - 11.8 \frac{\text{syllables}}{\text{words}} - 15.59 \quad (3.4)$$

Gunning Fog Index: This index also calculates the grade level of a text, so the higher its value, the more complex the text is to read. It uses the concept of *complexwords*, which is the number of words with three or more syllables, excluding proper names and combinations of simpler terms (e.g., “housekeeper” / “butterfly”). Common suffixes, such as -ed, -es, and -ing, should not be counted as a syllable [17].

$$GFI = 0.4 \left(\frac{\text{words}}{\text{sentences}} + 100 \frac{\text{complexwords}}{\text{words}} \right) \quad (3.5)$$

Läsbarhetsindex (LIX): Very similar to *GFI*, but in this case, *complexwords* is the number of words with more than six letters [13]. The higher its value, the more difficult the text is to read.

$$LBI = \frac{\text{words}}{\text{sentences}} + 100 \frac{\text{complexwords}}{\text{words}} \quad (3.6)$$

Smog-Grading: Estimates the years of education a reader needs to fully comprehend a piece of text, that is, its grade level. It considers the average number of polysyllables per 30 sentences, so it is usually calculated by collecting a sample of 30 sentences and counting the words with three or more syllables [49]. The displayed formula is generalized for any number of sentences.

$$SG = 1.043 * \sqrt{30 \frac{\text{polysyllables}}{\text{sentences}}} + 3.1291 \quad (3.7)$$

Difficult Word Score (DWS): The DWS is calculated by counting the number of *difficult words*, which is a definition that varies between papers. For example, according to Dang & Ignat, “A word is considered difficult if it does not appear in a list of 3000 common English words that groups of fourth-grade American students could reliably understand.” [30].

Dale-Chall: Uses the concept of *difficult words* described above, as Dale & Chall [26] defined, combining it with the average sentence size to estimate readability. They detailed the list

of 3000 “simple” words⁴, but it does not include all regular verb conjugations, noun plurals, and possessives, which should also be considered simple.

$$DC = 0.1579 * (100 \frac{\text{difficult words}}{\text{words}}) + 0.0496 \frac{\text{words}}{\text{sentences}} \quad (3.8)$$

Linsear Write Formula: Combines the number of “easy” and “hard” words to determine the grade level of the text. In the formula, n_1 (easy words) is the number of words with two syllables or less, and n_2 (hard words) is the number of words with three syllables or more. Full names, numbers, and dates count as one easy word [25].

$$LWF = \begin{cases} \frac{n_1 + 3 * n_2}{\text{sentences} * 2}, & \text{if } \frac{n_1 + 3 * n_2}{\text{sentences}} > 20. \\ \frac{n_1 + 3 * n_2}{\text{sentences} * 2} - 1, & \text{otherwise.} \end{cases} \quad (3.9)$$

Table 3.10: List of Readability Features mentioned in the assessed publications

ID	Name	Proposed in
RARI	Automated Readability Index	[8, 27, 30, 83, 85, 86]
RCL	Coleman-Liau	[8, 27, 30, 85, 86]
RFE	Flesh reading ease	[3, 8, 27, 30, 80, 85, 86]
RFK	Flesh-Kincaid	[8, 27, 30, 80, 85, 86]
RGFI	Gunning Fog Index	[8, 27, 30, 85, 86]
RLBI	Lasbarhets Index	[8, 27, 85, 86]
RSG	Smog-Grading	[3, 8, 27, 30, 85, 86]
RDWS	Difficult Word Score	[3, 30]
RDC	Dale-Chall	[3, 8, 30]
RLWF	Linsear Write Formula	[30]

3.4.4 History Features

History features evaluate quality by analyzing an article’s revision history, authors, and contributions. Stable Wikipedia articles with trustworthy authors tend to be better quality than controversial articles with frequent edits. Table 3.11 lists all the collected history features, some of which make use of the following concepts:

- **Active revision:** Revision made by one of the most active 5% contributors
- **Occasional revision:** Revision made by a user that edited the article less than four times

3.4.5 Network features

Network features are based on the articles’ graph, in which nodes represent articles and the edges show the citations between them. High-quality articles tend to cite other sources and are more

⁴Available at <https://countwordsworth.com/download/DaleChallEasyWordList.txt>

Table 3.11: List of History Features mentioned in the assessed publications

ID	Name	Proposed in
HA	Age (days)	[80, 85]
HAPR	Age per review	[27, 85]
HRPD	Reviews per day	[27, 85]
HRPC	Reviews per contributor	[8, 27, 85, 86]
HRPCSD	Reviews per contributor standard deviation	[8, 27, 85, 86]
HD	Discussion Count	[8, 27, 85, 86]
HR	Review Count	[8, 27, 32, 70, 80, 83, 85, 86]
HC	Contributor Count	[8, 80, 83, 85]
HRC	Registered Contributor Count	[8, 86]
HAC	Anonymous Contributor Count	[8, 83, 85, 86]
HRCPC	Registered Contributor Count per Contributor	[8, 18]
HACPC	Anonymous Contributor Count per Contributor	[8, 18]
HRCPAC	Registered Contributor per Anonymous Contributor	[8, 18]
HAR	Anonymous review Count	[8, 27, 80]
HRR	Registered review Count	[8, 27, 32, 80]
HARPR	Anonymous review Count per Review	[8]
HRRPR	Registered review Count per Review	[8]
HRRPAR	Registered Review per Anonymous Review	[8]
HREV	Revert Count	[8, 80]
HREVPR	Revert Count per Review	[8, 18]
HRML	Recently Modified Lines (3 months)	[8, 27, 85]
HREC	Recent Review (3 months) Count	[8]
HRECPR	Recent Review (3 months) Count per Review	[8, 27, 85, 86]
HACT	Active Review Count	[8]
HACTPR	Active Review Count per Review	[8, 27, 86]
HOCCPR	Occasional Review Count per Review	[27, 85]
HPROB	ProbReview [45]	[8, 18, 27, 85]
HQAC	Number of FA/GA by Contributors	[29]
HSSLE	Seconds since Last Edit	[29]
HABPR	Per-author-based Page Rating	[78]
HEBPR	Per-edit-based Page Rating	[78]
HTC	Trensient Contribution	[100]
HPC	Persistent Contribution	[100]
HERV	# of Edits recently reverted for vandalism	[41]
HERNV	# of Edits recently reverted for non-vandalism	[41]

likely to be cited by others. Therefore, these features, listed in Table 3.12, may be strong indicators of quality.

Table 3.12: List of Network Features mentioned in the assessed publications

ID	Name	Proposed in
NPR	PageRank [64]	[8, 27, 85, 86]
NIN	In-degree	[8, 27, 85, 86]
NOUT	Out-degree	[8, 27, 85, 86]
NASS	Assortativity (in-in, in-out, out-in, out-out)	[8, 27, 85, 86]
NLCC	Local Clustering Coefficient	[8, 85, 86]
NREP	Reciprocity within Neighbors	[8, 27, 85, 86]
NT	# of Versions in other Languages	[8, 27, 85, 86]

3.4.6 Stvilia’s IQ Metrics

Stvilia et al. [79] also propose 7 Information Quality (IQ) metrics that, combining different article features, aim to evaluate Wikipedia quality more systematically. They were excluded from all the other categories because they incorporate characteristics from several classes of features. The definitions for each metric were taken from Stvilia et al.’s study [80] and their formulas are defined in Equations 3.10 to 3.16.

Authority: Authority is defined as “the degree of the reputation of an information object in a given community”. *Connectivity* corresponds to the number of articles with at least one contributor in common with the assessed article.

$$\text{Authority} = 0.2 \times \text{UniqueEditors} + 0.2 \times \text{Contributions} + 0.1 \times \text{Connectivity} + 0.3 \times \text{Reverts} + 0.2 \times \text{ExternalLinks} + 0.1 \times \text{RegisteredContributions} + 0.2 \times \text{AnonymousContributions} \quad (3.10)$$

Completeness: Authors define Completeness as “the granularity or precision of an information object’s model or content values”.

$$\text{Completeness} = 0,4 \times \text{InternalBrokenLinks} + 0,4 \times \text{InternalLinks} + 0,2 \times \text{ArticleLength} \quad (3.11)$$

Complexity: Complexity is defined as “the degree of cognitive complexity of an information object relative to a particular activity”.

$$\text{Complexity} = 0,5 \times \text{FleschreadingEase} - 0,5 \times \text{FleschKincaidgradelevel} \quad (3.12)$$

Informativeness: Measures the amount of information in a document. *InfoNoise* represents the ratio between the size of the information and the article, measuring the amount of *noise* in the document. *Diversity* refers to the ratio between editors and total edits.

$$Informativeness = 0,6 \times InfoNoise - 0,6 \times Diversity + 0,3 \times Images \quad (3.13)$$

Consistency: Consistency is defined as “the extent to which similar attributes or elements of an information object are consistently represented with the same structure, format, and precision”.

$$Consistency = 0,6 \times AdministratorsEditShare + 0,5 \times Age_{days} \quad (3.14)$$

Currency: Currency corresponds to “the age of an information object” in days.

$$Currency = CollectionDate - LastEditDate \quad (3.15)$$

Volatility: Volatility measures “the amount of time the information remains valid”.

$$Volatility = MedianRevertTime \quad (3.16)$$

These metrics are experimented with by Couto and Lopes [24], although not within a Machine Learning context. The application of ML algorithms to Stvilia’s metrics could be worthy of experimentation.

3.5 Machine Learning Approaches

This section details the ML algorithms and datasets we collected during the review. We also attempt to determine which categories of features lead bring the best performance to the results.

3.5.1 Algorithms and Performance

Among the 22 included articles, their authors conducted 47 different experiments with Supervised Machine Learning algorithms. The results often varied depending on the number of considered classes, as accuracy tends to increase when the number of labels is reduced.

Due to the nature of Machine Learning algorithms, it is unlikely that the best approach will be the same for every dataset. This concern relates to the *No Free Lunch* Theorem [99], which states that all optimization algorithms have the same performance when averaged across all possible problems. Regardless, we collected all the results to understand better which algorithms were experimented with, as that could be a starting point for future studies.

In Table 3.13, we list the methods that, in each study, yielded the best performance and the used number of classes used. The preferred performance measure was the accuracy of the model (chance of predicting the correct quality), as it is the simplest to understand and a great indicator of overall model effectiveness. However, authors often favor other metrics, so a different measure was listed in such cases.

Table 3.13: Best Machine Learning algorithms from included publications

Study	Best Method	Measure	Result	Classes
Dang & Ignat [29]	RNN + LTSM	Accuracy	68%	6
Dalip et al. [27]	SVR	MSE	0.82	6 ¹
Saengthongpattana [70]	Naive Bayes	AUC	99	2
Su & Liu [81]	SVM	F1-Score	0.857	2
Dang & Ignat [30]	Random Forest	Accuracy	64%	6
Bassani & Viviani [8]	Gradient Boosting	Accuracy	61.8%	7
Wang et al. [86]	Decision Tree	Accuracy	71.1%	3
Velichety [83]	C5.0	Accuracy	84.7%	4
Wang & Li [85]	Stacked LTSM	Accuracy	79.8%	2
Dang & Ignat [28]	Deep Learning	Accuracy	55.5%	6
Banerjee et al. [6]	Ant Colony Optimization	N/A ²	N/A ²	N/A ²
Blumenstock [14]	MLP	Accuracy	97.15%	2
Lipka & Stein [51]	SVM	F1-Score	0.964	2
De La Calzada [18]	SVM	Accuracy	92%	2
Stvilia et al. [79, 80]	C4.5	F1-Score	0.9	2

¹ This study followed a regression approach, and thus the model does not output a class, but these are the number of classes the regressor took into account when training.

² The authors did not provide quantifiable results in their paper

3.5.2 Feature Importance

As defined in RQ2, we aim to assess the effectiveness of the features we collect from our literature review. Since most studies have a different number of classes and performance measures, it is inappropriate to compare the performance between them directly. However, we grouped the publications with similar parameters and compared their effectiveness to determine the best methods. Such an analysis is also flawed, considering the datasets will vary between studies, but it is still a good assessment of relative performance.

Dang & Ignat's works [28, 29, 30] all measure their algorithm's accuracy and use six classes in their datasets. However, only one [30] uses features for their best-performing algorithm. Content and Readability features are used in that paper, but without other similar studies to compare with, this does not allow us to derive meaningful conclusions.

Su & Liu [81], Lipka & Stein [51], and Stvilia et al. [79] also conducted their experiments with similar parameters (2 classes, F1-score). Lipka's work showed the best performance using Content and Style features, but Stvilia's IQ metrics, which mix different categories of features, also had good results.

Dalip et al. [27] compared the performance of numerous feature subsets, with Content and Style features yielding the best performance when grouped. They also determined that measuring Character Count and Word Count does not improve performance when considering Style and other Content features. Wang et al. [85, 86] also do a similar study, and they too conclude that Content and Style features tend to improve results more than the others. However, they also infer that Network and History features significantly improve algorithm performance, although slightly less than the other categories.

Summarizing, Content and Style features seem to bring a more significant performance boost than the others. However, they are also the most prevalent among the studies, so such an analysis is slightly biased. As Wang et al. [85, 86] show in their studies, Network and History features are also good quality indicators and thus somewhat improve algorithm performance.

3.5.3 Datasets

We also opted to gather all the used datasets to understand better how the methodologies differ between authors. Below we list factors like class proportions, article languages, and dataset size.

- **Dang & Ignat [29]**: Combination of three datasets from Wikimedia Foundation⁵. Contains 29,502 English articles, 9,000 French articles, and 8085 Russian articles.
- **Dalip et al. [27]**: Full English Wikipedia, comprising 2,683,000 articles at the time (2009).
- **Saengthongpattana [70]**: Full Thai Wikipedia, comprising 85,000 articles at the time (2014).
- **Su & Liu [81]**: Approximately 12,000 articles, 30% featured, 70% low-quality.
- **Dang & Ignat [30]**: Approximately 20,000 Wikipedia articles, with a close to even distribution of quality.
- **Bassani & Viviani [8]**: 400 articles from each type of quality, for a total of 2800.
- **Wang et al. [86]**: 3,294 English articles. “The high-quality class includes FA and AC. The medium class includes GA and BC. The low quality includes ST and SB. These three classes are distributed equally”.
- **Velichety [83]**: Full English Wikipedia, comprising 4.7 million articles at the time (2019).
- **Wang & Li [85]**: 3,294 English articles. “The high-quality class includes FA, AC, and GA, and the low-quality class includes BC, SC, and SB”. Distributed equally.
- **Dang & Ignat [28]**: 30,000 English articles⁶.
- **Blumenstock [14]**: 11,067 articles (1,554 featured / 9,513 random).
- **Lipka & Stein [51]**: 760 English articles, with a 50/50 balancing between featured and non-featured.

⁵<https://wikimediafoundation.org/>

⁶Dataset available at <http://datasets.wikimedia.org/public-datasets/enwiki/>

- **De La Calzada [18]**: 96 articles.
- **Stvilia et al. [79, 80]**: 1,070 articles (236 featured / 834 random).

The datasets and algorithms vary significantly between studies, making it difficult to determine the best approach. Unfortunately, papers rarely describe the used datasets in substantial detail, making it more complex to replicate the work entirely, but it is not a critical issue. Regardless, most studies tend to collect articles from the English Wikipedia and often reduce the number of classes to simplify the problem. However, it would be interesting to see more publications experiment with the complete set of quality classes.

3.6 Discussion

This section discusses the results we obtained after conducting this literature review while attempting to answer the research questions we stated in Section 3.2:

- RQ1. What are the most common features and metrics used to predict article quality in Wikipedia?
- RQ2. To what extent are those features currently used, and how effective are they?
- RQ3. Which machine learning approaches can be applied to predict article quality, and how effective are they?

In this review, we collected 165 different features, and each of them factored in the text of an article, its revision history, or how it relates to other articles within Wikipedia. Even though the Style Features make up the largest group of identified features, a big part of them are only proposed in two studies (Dalip et al.'s [27] and Bassani and Viviani's [8]). Therefore, the Content Features, which consider only the length and structure of the article, are the ones proposed by the majority of the reviewed papers. The other features are frequently suggested but not as often as the previous ones, possibly due to their higher computational costs.

We also collected a group of seven IQ metrics that aim to assess quality more directly, without a need for ML algorithms. However, these metrics are less commonly used among the publications we reviewed. Besides, although they may be used with Machine Learning, they are better suited for more manual approaches.

Some papers also compare the subsets of feature categories regarding their effectiveness at predicting quality. By analyzing the different studies, we see that Content and Style features appear to be very common and the most effective, but History and Network features are sometimes considered very useful for predicting quality too. Stvilia's IQ Metrics and Readability Features also appear to generate promising results, but not so significantly as the others do, since those directly combine existing features in a specific way.

Many algorithms were experimented with in terms of Machine Learning approaches, totaling 39 distinct experiments. Some authors opted for Deep Learning methods, while others instead used classical statistical learning algorithms, achieving different results. Nevertheless, deciding

which algorithm is the most effective is not trivial, as that is essentially dependent on the dataset definitions, which were different across the different papers.

In summary, the quality assessment of Wikipedia is a significantly researched topic, for which there are multiple methods to approach it. The collected results will help any future work related to this research topic, and further experimentation may help develop better quality predictors.

Chapter 4

Methodology

This chapter explains in detail all the steps involved in developing our solution. Our end objective was to create a Google Chrome Extension that automatically predicts the quality of Wikipedia articles. We opted for using a machine learning approach, so we needed to define a dataset, automate the calculation of features and then compare the performance of different algorithms, maximizing the model's accuracy.

4.1 MediaWiki Action API

The Wikimedia Foundation manages the MediaWiki software, which powers numerous websites, including Wikipedia [55]. It provides an Action API (Application Programming Interface), allowing programmers to automatically fetch, create, edit, and obtain metadata about Wikipedia pages [56].

The API has several endpoints available, including URLs for MediaWiki and MetaWiki Commons¹, but we are mostly interested in the Wikipedia APIs. Each version of Wikipedia will have a different base URL. For example, the English Wikipedia has the following entry-point:

```
https://en.wikipedia.org/w/api.php
```

Instead, calls to the French API would require using its respective language code, `fr`, instead of `en`. For a more comprehensive listing, Appendix B shows in Table B.1 the language codes for the 314 active Wikipedia versions (April 2022 [91]).

4.1.1 Relevant Modules

Each Wikipedia API has its own set of modules or actions², each of which performs operations related to that module. The query command is the most relevant for our work, permitting us to

¹MetaWiki's website is available at <https://meta.wikimedia.org/>

²Full documentation is available at <https://en.wikipedia.org/w/api.php>

fetch data from Wikipedia. It includes several sub-modules, such as `categories` and `revisions`, which allow us to obtain information about Wikipedia categories and edits. Each API method may contain a variable number of parameters, all of which we must pass through the called URL. Table 4.1 shows the commands of the query module which are most relevant to this study.

Table 4.1: MediaWiki API: Important Query commands

Command	Description
<code>categorymembers</code>	Returns a list of all pages within the category specified by <code>cmtitle</code>
<code>revisions</code>	If <code>rvlimit</code> is specified, returns the <code>rvlimit</code> most recent revisions of the page specified by <code>titles</code> . Otherwise, returns the latest revision of all the pages specified by <code>titles</code> , separated by “ ”
<code>langlinkscount</code>	Obtains the number of versions of the pages specified by <code>titles</code>
<code>random</code>	Returns one or more random Wikipedia pages

4.1.2 Wikitext

Wikitext is the markup language that MediaWiki software uses to format pages [97]. It has a well-defined syntax and a vast range of tags, frequently called templates, each with its purpose. Editors must use these templates to write the article structure, its references and internal links, tables, styling, and every other element present in the article. Figure 4.1 shows a comparison between a Wikipedia page and its respective wikitext.

When requesting a Wikipedia article through the API, we may choose to obtain the page’s HTML or its raw wikitext. We opted for the wikitext, as its structure is simpler to interpret.

4.2 Dataset Definition

This section details the process we followed to construct a balanced dataset for training the machine learning model.

4.2.1 Title Collection

Our first step was to collect a list of all the titles of each quality class from English Wikipedia to simplify the creation of our dataset. We opted not to use any other Wikipedia version because different editions may have specific quality scale standards, leading to inconsistencies in the labeled data. Besides, there are over 300 different Wikipedias, and training a model for each would be an extraordinarily complex task. Therefore, we only considered the English Wikipedia, as it is the largest one and has the most substantial user base [91], so they will be more active in managing content assessment.

An **island** (or isle) is an isolated piece of habitat that is surrounded by a dramatically different habitat, such as water.^[1] Very small islands such as emergent land features on **atolls** can be called **islets**, **skerries**, **cays** or **keys**. An **island in a river** or a **lake island** may be called an **eyot** or **ait**, and a small island off the coast may be called a **holm**. Sedimentary islands in the **Ganges delta** are called **chars**. A grouping of geographically or geologically related islands, such as the **Philippines**, is referred to as an **archipelago**.

There are two main types of islands in the sea: continental and oceanic. There are also **artificial islands**, which are man-made.

An '''island''' (or isle) is an isolated piece of habitat that is surrounded by a dramatically different habitat, such as water.<ref>{{cite book|title=Encyclopedia of Islands|editor1-last=Gillespie|editor1-first=Rosemary G.|editor2-last=Clague|editor2-first=David A.|publisher=University of California|volume=|edition=|year=2009|chapter=|pages=xxxi|isbn=9780520256491|url={{Google books|plainurl=yes|id=g9ZogGs_fz8C|page=xxxi}}}</ref> Very small islands such as emergent land features on [[atoll]]s can be called [[islet]]s, [[skerry|skerries]], [[cay]]s or keys. An [[river island|island in a river]] or a [[lake island]] may be called an [[ait|eyot or ait]], and a small island off the coast may be called a [[holm (island)|holm]]. Sedimentary islands in the [[Ganges delta]] are called [[List of islands of Bangladesh|chars]]. A grouping of geographically or geologically related islands, such as the [[Philippines]], is referred to as an [[archipelago]].

There are two main types of islands in the sea: continental and oceanic. There are also [[artificial island]]s, which are man-made.

Figure 4.1: A portion of Wikipedia’s *Island* article and its respective wikitext

Wikipedia provides a category page for each quality class, where they list the pages assessed as such. We took advantage of MediaWiki’s API to fetch the elements of each category, each of which is shown in Table 4.2. We should note that most of them do not contain the articles directly but instead list subcategories that detail the pages of that quality. For those cases, we had first to collect every subcategory and then obtain all the pages among them.

Table 4.2: Wikipedia categories of each quality class

Class	Category	Lists Subcategories?
FA	Category:Featured_articles	No
FL	Category:FL-Class_articles	Yes
A	Category:A-Class_articles	Yes
GA	Category:Wikipedia_good_articles	Yes
B	Category:B-Class_articles	Yes
C	Category:C-Class_articles	Yes
Start	Category:Start-Class_article	Yes
Stub	Category:All_stub_articles	No

After obtaining the list of titles for each one of the eight classes, we remove duplicate occurrences and save each list to a file. Some articles belong to multiple WikiProjects and so may be assessed differently. 229,756 titles suffered from that problem, primarily mid-low quality articles when we last collected the titles (May 2022). We did not add them to our final list to avoid

inconsistencies in the dataset. The entire process took over two hours and collected almost 5 million different article titles. Table 4.3 shows the number of results per quality level before and after removing the titles with multiple assessments. The table also demonstrates the staggering discrepancy between the number of excellent and sub-par articles on Wikipedia.

Table 4.3: Number of article titles gathered per quality class (May 2022)

Class	# of Titles (initial)	# of Titles (final)
FA	6,091	6,086
FL	3,966	3,964
A	959	131
GA	36,301	35,363
B	116,377	88,393
C	442,089	361,405
Start	1,950,433	1,744,635
Stub	2,335,646	2,189,785
Total	4,891,862	4,429,767

4.2.2 Dataset Dimensioning

Owning all the Wikipedia titles and the quality of their respective pages allowed us to create a randomized dataset much more effortlessly. The initial step was determining which labels we would consider and how many articles of each quality would be in the dataset.

We decided not to include the FL and A-class items in our experiments for distinct reasons. First, Featured Lists represent a different type of content than usual. Although they attain that status due to their comprehensiveness and structure, their goal is to list other items as thoroughly as possible and often do not contain much more content (e.g., see revision <https://en.wikipedia.org/w/?oldid=843704571>). Regarding the A-class tier, they correspond to a tiny portion of Wikipedia (see Table 4.3). They also do not follow the linear quality path since being a GA (Good Article) is not a requirement for being A-class. Furthermore, some parts of Wikipedia omit that tier [93], which was another reason for not considering it for our dataset.

As for the numbers, we resolved that 6,000 articles per quality class would be appropriate. We arrived at that number because we wanted to keep the dataset balanced, and the low supply of FA's does not allow us to expand it further. That decision was not mindless, though: Even if balanced datasets generally improve accuracy across the board, it is not necessarily beneficial for every real-world case. For instance, if end-users check B and C-class articles much more often, it is less crucial for the model to guess FA's and Stubs as accurately, so we could design a dataset with more mid-tier pages. Regardless, we opted for balancing it because the results we obtain will be more coherent. Besides, we cannot make assumptions regarding which articles readers view more frequently and should try to maximize accuracy on every quality level.

4.2.3 Article Collection and Aggregation

Collecting the articles was a straightforward process. We selected 6,000 random titles for each class and used MediaWiki’s API to obtain their contents. We thus fetched the information of 36,000 distinct English articles and afterward computed the 145 features for each one. We explain the process of feature computation in further detail in Section 4.3.

4.2.4 Network Graph Generation

Before feeding the page contents to the feature calculator, we must construct a network graph of each article we assess. We do so because network features consider some graph features, such as the in-degree and out-degree. We decided to build only one graph per level of quality, with the network data of all 6,000 articles in each class. We could have pre-built the entire graph for the 36,000 articles, but we determined it would not significantly impact the build time and would further increase memory usage.

The first step is obtaining the links between all Wikipedia articles. Wikimedia provides SQL dumps of Wikipedia page links³, but they were too large and computationally expensive to parse (55 GB, as of May 2022). Instead, we used WebGraph framework I, a library that eases the manipulation of large Web graphs [15]. The Laboratory for Web Algorithms provides datasets⁴ of several networks, including Wikipedia, compatible with the WebGraph framework. We downloaded the `enwiki-2022` dataset, which contains 6,492,490 nodes and 159,047,205 arcs.

Using the WebGraph framework, we quickly assembled a program that parses the provided dataset into two CSV files, `nodes.csv` and `edges.csv`, so we can process them later more efficiently. The node file maps each title to a numeric ID, and the edge file lists all the arcs as a pair of IDs, maintaining disk usage at acceptable levels (around 2.5 GB).

Having the list of nodes and edges, the dataset builder constructs the graphs using that information. We heavily simplified the graph’s representation to avoid running out of memory. We first extracted the adjacent neighbors of all the 6,000 articles, distinguishing between in and out arcs. After that, we computed the in and out-degree of all the neighbors. Figure 4.2 shows a diagram representing how the program structures the information extracted from the graph.

The `enwiki-2022` dataset was produced the same week we ran the experiments described in this document, so the values are up-to-date. However, when we randomly select the 36,000 articles to include in our dataset, we sometimes encounter titles that are not present in `enwiki-2022`. Those occurrences are rare, so we skip the missing items and do not include them in the dataset.

4.2.5 Dataset Storage

After calculating each article’s entire set of features, our final step was to partition our data into a training set and a testing set [42]. We decided on a 70/30 split, 70% for training and 30% for testing data, as previously done by multiple authors [54, 76]. Table 4.4 shows the final distribution

³English Wiki dumps are available at <https://dumps.wikimedia.org/enwiki/latest/>

⁴We used the datasets available at <https://law.di.unimi.it/datasets.php>

Moon	1	1	title $\rightarrow$ Moon out $\rightarrow$ o_{11}, o_{12} in $\rightarrow$ i_{11}, i_{12}	o_{11}	in_deg $\rightarrow$ 4 out_deg $\rightarrow$ 2
Guitar	2	2	title $\rightarrow$ Guitar out $\rightarrow$ o_{21}, o_{22} in $\rightarrow$ i_{21}, i_{22}	o_{12}	in_deg $\rightarrow$ 7 out_deg $\rightarrow$ 12
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Figure 4.2: Diagram representation of the information extracted from the graph, applied to two hypothetical articles, Moon and Guitar. The table on the left maps the titles of the main articles (the 6,000 items of the same quality whose features we want to calculate) to their assigned identifiers. The middle table stores information about the main articles, while the right table only stores information about their immediate neighbors.

of articles in our dataset. We initially picked 6,000 titles from each level of quality, but we did not compensate for missing items during the generation of network graphs. Fortunately, that step showed an insignificant number of missing results, so we consider our dataset practically balanced.

Table 4.4: Article quality distribution within the dataset

Class	Training	Testing	Total
FA	4,200	1,800	6,000
GA	4,200	1,800	6,000
B	4,197	1,798	5,995
C	4,199	1,799	5,998
Start	4,196	1,797	5,993
Stub	4,198	1,798	5,996
Total	25,190	10,792	35,982

It took 16 hours to fully construct the dataset, the most considerable portion of which was spent on computing the features (around 13 hours). Extracting the information from the network graphs took about 2 hours, not including the usage of the WebGraph framework, which pre-produces a list of nodes and edges and does not take long to run. The API calls for obtaining the contents of the Wikipedia pages did not make up much of the build time, totaling around 1 hour. For reference, Table 4.5 details the specifications of the machine we used to run the dataset builder and the remaining experiments.

Table 4.5: Computer Specifications

Operating System	Windows 10 Education 64-bit
Processor	Intel Core i7-4770 CPU @ 3.40GHz (8 CPUs)
Memory (RAM)	16,384 MB
Graphics Card	NVIDIA GeForce GTX 1070

4.3 Feature Computation

This section comprehensively describes the calculation of the features for each Wikipedia article. We kept in our feature set most of the features collected in our literature review (see Section 3.4) and added a few original ones. Figure 4.3 shows the distribution of categories in our dataset, but Appendix C exhaustively lists the features we included.

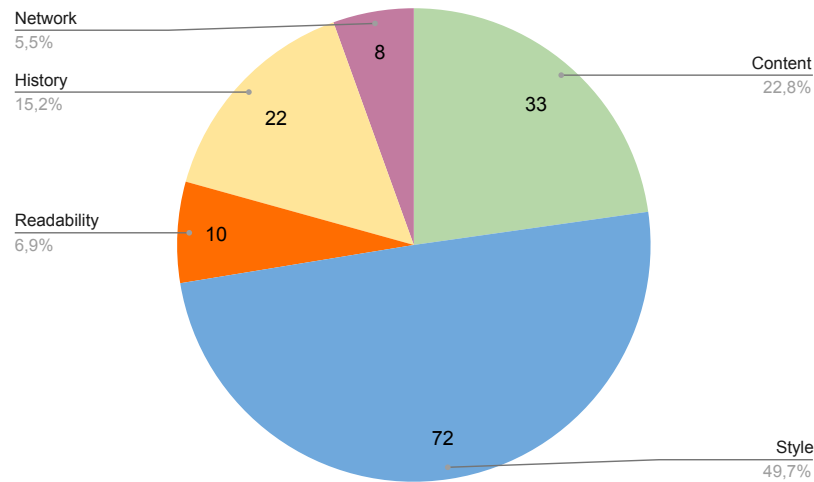


Figure 4.3: Dataset Features: Count per Category

4.3.1 Wikitext Cleaning

Some features required us to isolate the text of the article from all the formatting tags and templates. Therefore, we clean the article’s wikitext before extracting its features, obtaining the filtered plain text.

During the cleaning process, we remove references, footnotes, and other sections without meaningful text. As we need those to compute certain features, we still preserve the original wikitext. We also remove many non-textual templates, like {chess diagram} and {location map}, and ignore formatting tags, such as <small> and <center>, keeping the text inside them.

The filtering process also removes some textual elements, the most common example being the tables. It was a difficult decision, but we ultimately determined to do so because structured content cannot be easily converted into plain text elements (i.e., sentences and paragraphs). We still include them in features that consider the entire wikitext, but the table’s contents will not count for features like Character Count and Word Count. If the other features disregard the contents of the tables entirely, model performance could take a hit, but it may not be that meaningful. Our dataset averages 0.727 tables per article against 12,724 characters and 2,051 words per page, so this issue should not significantly impact our results.

To exemplify the process mentioned above, we demonstrate in Figures 4.4 and 4.5 the wikitext of a small Wikipedia page⁵, and the resulting text, after being cleaned.

```
{{Short description|Politician}}
{{Infobox officeholder
|name           = Frits Palm
|image          =
|alt            =
|caption        =
|office         = Member of the [[House of Representatives (Netherlands)|House of
Representatives]]
|term_start1    = 23 May 2002
|term_end1      = 30 January 2003
|birth_date     = {{Birth date|1947|01|30|df=yes}}
|birth_place    = [[Singapore]]
|death_date     = {{Death date and age|2011|03|07|1947|01|30|df=yes}}
|death_place    = [[Delft]], Netherlands
|party          = [[Pim Fortuyn List]]
|alma_mater     = [[Erasmus University Rotterdam]]
|religion       =
|website        =
}}
'''Frederik Albert "Frits" Palm''' (born 1947 – died 2011) was a Dutch economist,
civil servant and former politician for the [[Pim Fortuyn List]] in the [[House
of Representatives (Netherlands)|House of Representatives]] from 2002 to 2003.

Before he was elected to parliament, Palm was the director of the Cabinet of
Antillean Affairs. In parliament he focused on Foreign Affairs, European
Affairs, Defense and Kingdom Relations. Palm was also second vice-president of
the House of Representatives.<ref>{{Cite web|url=https://www.parlement.com/id/
vg9fgopy3mdw/f_a_frits_palm|title=Drs. F.A. (Frits) Palm|website=www.parlement.
com}}</ref><ref>{{Cite web|url=http://www.biografischportaal.nl/persoon
/41594161|title=Drs. Frits Palm|website=www.biografischportaal.nl}}</ref><ref
>{{Cite web|url=https://www.parlementairemonitor.nl/9353000/1/j9vvij5epmjley0/
vgbwk0xpkxzt?ctx=vg9fgopvu7zh|title=Weisglas in de Volkskrant: "Politiek
vergrofd door LPF" – Parlementaire monitor|website=www.parlementairemonitor.nl
}}</ref>
```

Figure 4.4: *Frits Palm's* Wikipedia Page: Original Wikitext (April 2022)

Overall, our wikitext cleaner is not flawless, but empirical findings show almost perfect results. Wikitext has a complex syntax and countless tags, so it is difficult to determine all the edge cases. To ensure maximum reliability, we collected a total of 39 different articles from diverse fields (biology, chemistry, history, technology, and religion, among others) and then designed unit tests to always produce correct results for those pages. The tests do not detect all the flaws but give us confidence that the process works for most cases.

⁵We show results for the Wikipedia page https://en.wikipedia.org/wiki/Frits_Palm, as of April 2022.

```

Frederik Albert "Frits" Palm (born 1947 - died 2011) was a Dutch economist, civil
servant and former politician for the Pim Fortuyn List in the House of
Representatives from 2002 to 2003.

Before he was elected to parliament, Palm was the director of the Cabinet of
Antillean Affairs. In parliament he focused on Foreign Affairs, European
Affairs, Defense and Kingdom Relations. Palm was also second vice-president of
the House of Representatives.

```

Figure 4.5: *Frits Palm*'s Wikipedia Page: Cleaned Wikitext (April 2022)

4.3.2 Natural Language Processing Tools

Several feature categories demand a specialized analysis of the plain text we extract from the wikitext (i.e., Natural Language Processing, NLP). We used NLTK (Natural Language Toolkit) [11,12], a library that allows Python programs to process natural language data. It provides features like tokenization for isolating sentences and words, and POS (part-of-speech) tagging for identifying word classes. We also relied on the `syllables` python package⁶ to estimate the number of syllables in each word.

Although these tools are powerful, they are mainly designed to work for English texts. It is possible to use them for a few other Western languages, but they will not perform as accurately. We will take this issue into account during experimentation.

4.3.3 Content Features

Content features analyze the length and structure of the article and can be all obtained from the wikitext. It requires both the plain text and the original wikitext to calculate the features entirely.

We kept all the features listed in Table 3.8 of Chapter 3, except for CIBL (Internal Broken Link Count). Although a potentially decent indicator of quality, we figured it would be difficult to determine the existence of every broken link on the page. We also added two new features: CTB (Number of Tables) and CUCT (Number of Unique Templates).

Some features are likely to be more relevant when determining quality, but all should have a non-negligible impact on how good an article is. Grouping the content features by common themes, we can explain the intuition behind each and how to obtain them, and we describe that information in Table 4.6.

4.3.4 Style Features

Style features evaluate how contributors write their sentences, measuring the used word classes and sentence types. They can all be derived from the article's plain text.

⁶The `syllables` package is available at <https://pypi.org/project/syllables/>

Table 4.6: Intuition behind the Calculation of Content Features

Group	Intuition	Obtained from
Basic Elements	Characters, words, and sentence count must not be too low or too high. Otherwise, the text will not have a proper flow.	Plain text
Sections	The distribution of sections, subsections, and paragraphs must be well-defined to allow for a smooth article structure.	Plain text
References	Good articles cite multiple trustworthy sources, so a low citation and reference count may indicate poor quality writing.	Wikitext
Links	Both internal and external links can have some correlation with quality, for reasons similar to the ones mentioned in the previous row.	Wikitext
Images	The count and distribution of images is important for improving the reader's experience.	Wikitext
Templates	The varied usage of Wikitext templates (including tables) may indicate a richer article.	Wikitext

Some features listed in our Literature Review (see Table 3.9) were excluded from our dataset. For most of them, we decided that calculating them was too complex for the benefit they might bring. Furthermore, style features make up almost half of our set, so skipping some of them helps mitigate the issue of feature dimensionality. Table 4.7 details the excluded style features, briefly describing the main reason for doing so.

Table 4.7: Excluded Style Features

Feature(s)	Reason for Exclusion
SSTPRE SPREPW	Features relating to subordinate prepositions and conjunctions are very similar to these counting prepositions
SAV	We already have features for measuring modal verbs. Auxiliary verbs are similar, so we deemed it unnecessary to spend time determining if a verb was auxiliary.
SPV SPVPW SMVPPV	Detecting passive voice usage is too complex for the benefit it might bring.
SNOMPW	Detecting nominalizations is too complex for the benefit it might bring.
SCHTRI SPOSTRI	Measuring the frequency of character and part-of-speech trigrams is too complex for the benefit it might bring.

Similar to the previous section, we grouped the features by common themes, explaining in Table 4.8 the intuition behind them.

These features are also relatively fast to compute. However, the NLP process with the NLTK tool sometimes takes a while (up to a couple of seconds for very long articles, which potentially amounts to hours with 36,000 articles).

Table 4.8: Intuition behind the Calculation of Style Features

Group	Intuition	Obtained from
Sentences	Better written texts generally have varied sentence lengths since repeating sizes causes a monotonous feeling to the reader.	Sentences
Sentence Structures	Interrogations and Exclamations are often less preferred in an expository text like a Wikipedia article.	Sentences
Syllables	A text with many long words makes it more difficult to read, but too many simple words may indicate a document is too simple to be of quality.	Syllable Estimation
Word Classes	These features are a simple way to retain a footprint of the entire text, so we measure every word class and their average per sentence.	POS Tagging

4.3.5 Readability Features

Readability features assess how easy a text is to read, using carefully designed formulas that combine word, syllable, and character count. We did not exclude any features from the Literature Review (see Table 3.10).

Applying the formulas was a swift and straightforward process, considering they only require values we had previously determined with the assistance of NLP tools (e.g., number of syllables and number of sentences). Calculating the number of difficult words was an exception, though, for we had also to collect the list of easy words⁷ initially proposed by Dale & Chall [26].

4.3.6 History Features

History features measure an article’s revision history and analyze its authors and contributions. Stable articles with trustworthy authors tend to be better quality than controversial articles with frequent edits. All the features within this category aim to measure articles based on this principle. To fully calculate these, we need all the revisions of a page, which we obtained through the MediaWiki API. For each revision, we request its author, comment, and length.

We removed and adapted some features we collected during the Literature Review (see Table 3.11). Tables 4.9 and 4.10 explain which features differ from our survey and why we made those decisions. We also added an extra feature, HMCL (Mean Comment Length), which should help incorporate some information about the revision discussion into our feature set.

Measuring history features takes significant time because we must first obtain the entire revision history of its respective article. Pages with more contributions will take longer to acquire. For example, the United States article⁸ has the second-largest history between Wikipedia articles, reaching 43,824 revisions (May 2022) [94]. During experimentation, fetching all its contributions

⁷Text version is available at <http://countwordsworth.com/download/DaleChallEasyWordList.txt>

⁸Wikipedia article is available at https://en.wikipedia.org/wiki/United_States

Table 4.9: Excluded History Features

Feature(s)	Reason for Exclusion
HD	Would need to fetch the talk page for each article, heavily increasing dataset build times.
HPROB	The recursive formula is too complex to apply in real-time.
HQAC	Obtaining all the articles authored by each contributor requires high computational effort.
HAR HRR HARPR HRRPR HRRPAR	We already measure the number of anonymous and registered contributors in other features, so these would not add much value.
HABPR HEBPR HTC HPC	These metrics have very specific formulas and are difficult to compute in real-time.
HERV HERNV	It would be feasible to check the comments that mention vandalism, but revert-related features already serve a similar purpose.

Table 4.10: Adapted History Features

Feature	Reason for Adapting
HRML	Requesting access to the content of all the revisions would be very computationally expensive, so we used the difference between revision sizes instead of the modified lines.

took around 60 seconds, which is considerable. However, most articles rarely have that many revisions. Analyzing our dataset, 95.1% of the items have under 2,500 contributions, and 77.1% have under 500. Moreover, considering only the Stubs, Starts, and C-class articles, which comprise the vast majority of Wikipedia, 92.8% of them have under 500 revisions.

4.3.7 Network Features

Network features consider Wikipedia as a graph, where its nodes represent articles and the edges are the citations between them. The most complex part of its calculation is the Network Graph generation, which takes a substantial amount of time. As mentioned in Section 4.2.4, it contains all the titles of the dataset and their in and out-neighbors. It also stores the in/out-degree of the neighbors and the number of reciprocal edges of the main articles. We can almost instantly compute all the features from the graph, except for the NT (Number of Translations), which we obtain directly from the MediaWiki API.

We excluded two network features from the Literature Review results displayed in Table 3.12. The features and motives for exclusion are described in Table 4.11.

Table 4.11: Excluded Network Features

Feature	Reason for Exclusion
NPR	Computing the PageRank of one article requires the calculation of the PageRank of its neighbors and so on, which we considered unfeasible time-wise.
LCC	Calculating it would require us to compute edges of all neighbors, which would have been extremely time-costly.

Among all the categories we use, this is the least appropriate for real-time quality assessment because it requires a pre-computation of a network graph, which is a lengthy process. Experimentation must be conducted to assess if these features (and others) are suitable for real-time usage.

4.4 Experimentation

This phase of our study was extensive, as thorough experimentation is essential for developing an accurate machine learning model. This section details the experiments we conducted to answer the following questions:

- EQ1. Which ML algorithm yields the best performance?
- EQ2. How does model performance vary with the number of classes?
- EQ3. Which subset of features produces the best performance?
- EQ4. How long does each model take to train and test?
- EQ5. How effective is our model at predicting the quality of non-English articles?

Each experiment and respective model will have an assigned identifier with the following format:

`<CATEGORIES><CLASSES>/<ALGORITHM>_<TASK>`

We also explain this naming convention in Table 4.12. By default, our model will train with the six classes and all the described features (e.g., CSRHN6/tree_c).

Table 4.12: Training Experiments: Naming Convention

Element	Description	Example
<CATEGORIES>	Feature categories included in the dataset. Each category is represented by its initial letter. Possible categories: Content, Style, Readability, History, Network.	CSRHN
<CLASSES>	Number of output classes considered for training the model. If it is less than 6, multiple levels of quality may have the same class.	6
<ALGORITHM>	Algorithm used to train the model.	tree
<TASK>	Whether the chosen algorithm solves classification (c) or regression (r) tasks.	c

4.4.1 Algorithm Comparison

We selected 16 different machine learning algorithms for our experiments, eight for classification and eight for regression. Choosing the best learning algorithm also significantly impacts the model efficacy, so we designed several experiments to confidently determine the most effective approach. All experiments compare results between all 16 algorithms, but we analyzed CSRHN6/* models in more detail, assessing multiple metrics and statistics, which should reveal a summary of each algorithm’s overall performance. Although classifiers and regressors typically use distinct metrics, we also aim to understand the differences in their performance. Regardless, we expect regression to be a more suitable strategy due to the linear property of article quality.

Besides determining which algorithm generates the most promising results, we seek to calculate the best-performing hyper-parameters. There are several parameter optimization approaches, but we combined manual search and grid search, which do not necessarily give the optimal values but are still widely used in the current state-of-the-art [10]. We did not repeat this process for other experiments, as we observed that it was not improving results significantly.

We trained the models using *scikit-learn*, a Python library that provides efficient tools for classification and regression [72]. Table 4.13 lists the different algorithms we experimented with, along with the hyper-parameters we found to be most suitable (unmentioned parameters were left to *scikit-learn*’s default values). Many of these algorithms require the normalization of the input data to achieve maximum performance [47], so we invariably applied z-score scaling before training, which normalizes the features to have a mean value of 0 and a standard deviation of 1 [74]. It is a helpful generic method for when we don’t know the ranges of the input set.

Table 4.13: Model Experimentation: Used Machine Learning Algorithms

ID	Algorithm	Parameters
tree_c	Decision Tree Classifier	criterion = entropy max_depth = 15
tree_r	Decision Tree Regressor	criterion = squared_error max_depth = 15
forest_c	Random Forest Classifier	n_estimators = 150 criterion = entropy max_depth = 15
forest_r	Random Forest Regressor	n_estimators = 150 criterion = squared_error max_depth = 15
ada_c	Ada Boost Classifier	n_estimators = 150 learning_rate = 0.1 base_estimator = tree_c
ada_r	Ada Boost Regressor	n_estimators = 150 learning_rate = 0.1 base_estimator = tree_r
gboost_c	Gradient Boost Classifier	n_estimators = 150 learning_rate = 0.01
gboost_r	Gradient Boost Regressor	n_estimators = 150 learning_rate = 0.01
svc_c	Support Vector Classifier	kernel = rbf C = 2
svr_r	Support Vector Regression	kernel = rbf C = 2
mlp_c	MLP Classifier	hidden_layer_sizes = (100) alpha = 0.1 max_iter = 10,000
mlp_r	MLP Regressor	hidden_layer_sizes = (100) alpha = 0.1 max_iter = 10,000
knn_c	K-Nearest Neighbors	n_neighbors = 10
gnb_c	Gaussian Naive-Bayes	<i>No parameters were changed</i>
logreg_c	Logistic Regression	max_iter = 10,000
linreg_r	Linear Regression	<i>No parameters were changed</i>

4.4.2 Quality Class Mapping

Most regression algorithms do not work well with numerical labels due to the nature of their purposes. Therefore, we first converted the quality labels to numerical values. At first, a direct linear mapping seemed intuitive, but we considered it was not necessarily the best approach. Teblunthuis’s study [82] disproves the assumption that “different levels of quality are ‘evenly spaced’ from one another”. He used ordinal regression models to design a better calibrated and more accurate measure of quality for Wikipedia content. We detail in Table 4.14 the control quality scale for our regression experiments, based on Teblunthuis’s results, comparing it with a linear scale. This mapping does not affect classification experiments because those algorithms do not consider the values of each label.

Table 4.14: Regression Experimentation Class Mapping vs. Linear Class Mapping

Class	FA	GA	B	C	Start	Stub
Default	1.0	0.9	0.7	0.6	0.4	0.0
Linear	1.0	0.8	0.6	0.4	0.2	0.0

Although the dataset distinguishes between six distinct levels of quality, we also decided to study the model performance on a smaller number of labels. We thus conducted four different experiments for every algorithm, varying the number of classes in each one, but keeping all the features. Table 4.15 shows the quality-to-class mapping for each regression experiment. We maintained a value range of $[0, 1]$ to allow us to compare the results of different models more reliably.

Table 4.15: Regression Experimentation: Quality-to-class Mapping

# Classes	FA	GA	B	C	Start	Stub
6	1.0	0.9	0.7	0.6	0.4	0.0
5	1.0	0.9	0.6	0.6	0.4	0.0
3	1.0	1.0	0.6	0.6	0.0	0.0
2	1.0	1.0	0.0	0.0	0.0	0.0

Although classification experiments do not look at the value numbers, they will be based on the regression mapping, meaning quality levels with equal value will be considered the same class. Table 4.16 clarifies which classes we merged together for the classification experiments.

Table 4.16: Classification Experimentation: Quality-to-class Mapping. Table cells with identical color indicate the quality levels are considered the same class.

# Classes	FA	GA	B	C	Start	Stub
6	1	2	3	4	5	6
5	1	2	3	3	4	5
3	1	1	2	2	3	3
2	1	1	2	2	2	2

It is worth noting that the 5-class and 2-class mappings lead to unbalanced data. For instance, merging the B and C tiers in the 5-class experiment will effectively double the support of that label, and therefore ML algorithms would give more weight to those examples. Having imbalanced data would be inconsistent with the rest of our work, so we randomly removed excess rows from every label until the classes were distributed evenly. Table 4.17 shows the final dataset sizes for the different experiments.

Table 4.17: Dataset sizes for different numbers of classes

# Classes	Training	Testing	Total
6	25,190	10,792	35,982
5	20,980	8,985	29,965
3	25,190	10,792	35,982
2	16,800	7,200	24,000

4.4.3 Feature Subset Optimisation

The dataset we built contains 145 unique features. It is possible that using all of them leads to the most significant performance boost. However, as we intend to develop a tool that predicts quality in real-time, we must also focus on reducing unnecessary computational effort. Therefore, features that heavily increase run times but only improve the model effectiveness in a negligible way may be ignored.

We trained 14 extra models for each previously listed algorithm, varying the feature set in each run, and keeping all six classes. Table 4.18 lists those experiments, showing the different combinations of feature categories we tested, each with its combination ID. The table also includes the control group, which contains all the features.

Our goal was to test the impact of each category on the algorithm’s performance, so we trained models with feature sets with four and one categories, comparing them to the ones trained with CSRHN. We designed the remaining combinations for potential use in a real-time application as we likely cannot rely on network features and wanted to analyze the impact of removing other categories.

4.4.4 Training and Prediction Times

We recorded the elapsed time for training each model in every experiment. Those findings do not significantly affect designing a real-time application since the model is only trained once. Nevertheless, we considered it pertinent to compare the relative speed of training algorithms.

However, measuring prediction time is especially relevant when applying machine learning to a real-time scenario. We must measure the times of each article individually since we wish to simulate real-world usage of the model. We collected these results in two distinct moments:

- **Feature Computation Time:** We measured this information in a separate dataset. To maximize the accuracy of our results and minimize the impact caused by outliers, we measured

Table 4.18: Model Experimentation: Feature Subset Combinations

ID	C	S	R	H	N	Features
CSRHN	×	×	×	×	×	145
SRHN		×	×	×	×	112
CRHN	×		×	×	×	73
CSHN	×	×		×	×	135
CSRN	×	×	×		×	123
CSRH	×	×	×	×		137
C	×					33
S		×				72
R			×			10
H				×		22
N					×	8
CSR	×	×	×			115
CSH	×	×		×		127
CRH	×		×	×		65
CH	×			×		55

the feature computation times of 500 random articles of each quality and then calculated their average.

- **Model Prediction:** During the model testing phase, we collected the time taken to apply the model. We select 50 random rows from the dataset and measure the average time to predict each of them, repeating that process for each trained model.

We also determined that a significant majority of computational effort is spent calculating the article’s features. Consequently, we measured the time to compute each step, including pre-computation tasks, like word and sentence tokenization. Table 4.19 details every step we considered for these measurements and which feature categories they relate to. The final rows show the steps involved in calculating the features, while the first ones relate to the initial pre-computation required for computing the features. We excluded network features because, as we previously mentioned, calculating the entire network graph for every article is unfeasible for a real-time application.

The experiments mentioned in the previous sections strongly relate to this analysis. The algorithm, number of classes, and feature set should all somewhat impact the training time, whereas only the feature set should significantly affect prediction time.

4.4.5 Multi-language Performance

As we mentioned in Chapter 1, quality varies across different Wikipedia versions. We designed experiments assessing that discrepancy of quality and model effectiveness in distinct Wikipedias.

We constructed four new datasets, each respective to a different language. They all distinguish six different levels of quality, which helps maintain consistency with the rest of our work. We

Table 4.19: Feature Computation Steps and which Feature Categories use them

Step	C	S	R	H
Fetch Wikitext	×	×	×	
Clean Wikitext	×	×	×	
Tokenizer	×	×	×	
Syllable Estimator		×	×	
Revision History				×
Content Features	×			
Style Features ¹		×		
Readability Features			×	
History Features				×

¹This task includes POS-tagging

did not expand beyond that because few Wikipedia versions consider more than two levels of quality (equivalent to Featured and Good articles). Even with the same number of classes, it is not guaranteed that distinct quality scales have identical criteria. Still, we considered them to be similar enough for this experiment.

Each Wikipedia version has a different naming convention for its quality levels. To avoid confusion, we normalized the classes into letter grades, ranging from A+ to F. Table 4.20 displays the mapping between the classes and their respective letters.

Table 4.20: Model Experimentation: Quality Levels per Language

Standard	A+	A	B	C	D	F
English [93]	FA	GA	B	C	Start	Stub
French [90]	AdQ	BA	A	B	BD	Ébauche
Portuguese [89]	6	5	4	3	2	1
Russian [88]	MC	XC	I	II	III	IV

We did not find any Wikipedia version that provides a list of articles for every quality level, aside from the English one. We instead used MediaWiki’s API to obtain random articles, determining their quality from the discussion page. Although that approach produces an unbalanced dataset, it summarizes the quality distribution within each Wikipedia. We discarded some titles throughout the article collection, some for being unassessed and others for missing a discussion page, so the dataset sizes may differ between them. We disclose the dimensions of each dataset in Table 4.21. They all contain around 10,000 articles, but the size difference between them should not significantly affect our results.

Apart from considering diverse languages, we also experimented with three different feature sets for each one:

- **CSRH**: As our control group, this feature set includes all but Network features. We excluded those because we do not have access to WebGraph datasets for other Wikipedia versions.

Table 4.21: Model Experimentation: Multi-language Dataset Sizes

Language	# Articles
English	11,096
French	11,195
Portuguese	10,525
Russian	10,341

- **CRH:** Style features are mostly inaccurate outside the English domain. Over 75% of style features depend on the POS tagging from NLTK, which was designed to work with English sentences only, so we figured we should test model effectiveness without any style feature.
- **CH:** Although readability features may somewhat work for other languages, most were specifically designed to assess English texts, and they may yield less precise results in different situations [22]. Hence, we concluded it would be appropriate to include only Content and History features in one of the experiments.

We will compare the performance of *forest_r* models on those category sets and Wikipedia versions. We selected the Random Forest Regressors because they performed best on the standard dataset.

Chapter 5

Results

This chapter lists and discusses the main findings of our machine learning experiments. We start by analyzing some relevant properties of the dataset, then examine the results from every experiment we conducted, such as performance per number of classes, feature importance, and comparison of training and prediction times. Finally, we analyze non-English datasets and assess our model’s performance on them.

Since our result data is extensive, we summarize the essential information using tables and graphs, but we show the collected data in more detail in Appendix D. The full dataset is available in a research data repository¹.

5.1 Dataset Analysis

We began by examining key features and their correlation with article quality by aggregating their values from our dataset. Figures 5.1 and 5.2 show relevant statistics involving the number of characters, words, and sentences of different quality classes. The immediate conclusion is that better texts tend to be longer than low-quality ones, both in character and word count. We can also see that word length and sentence size are similar throughout every type of article, but Stubs tend to have smaller sentences than other articles.

Many other features are strongly associated with good quality too. However, most of them correlate with character count (e.g., Citation Count, Verb Count, Introduction Length), so it’s also natural that higher values show up in better articles.

Now comparing the age of different Wikipedia pages, Figure 5.3 shows that we can find recent and old articles of every quality. However, better articles are, on average, older than worse ones. More specifically, featured articles are about five and a half years older than stubs, but the correlation is not as direct as seen in the box plot. We did the same analysis for the number of revisions,

¹While the paper awaits publication, the entire dataset is available at <https://doi.org/10.25747/2rdd-rc08>

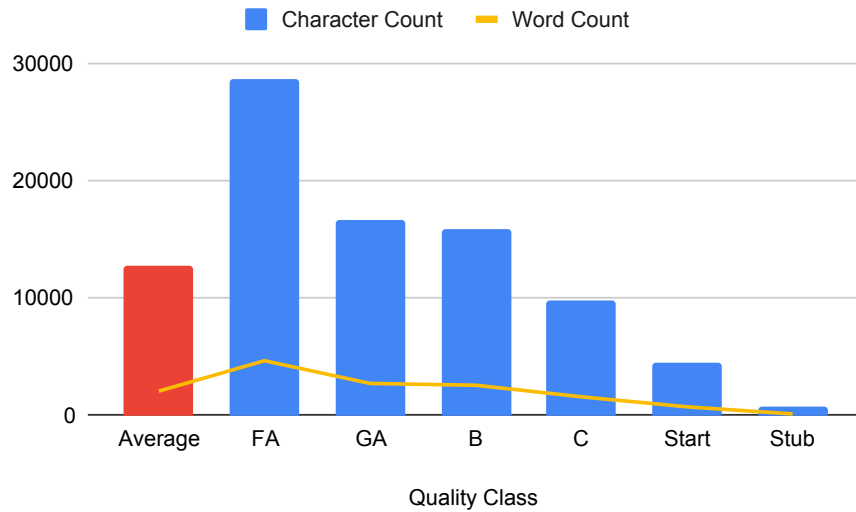


Figure 5.1: Average character and word count of each quality class

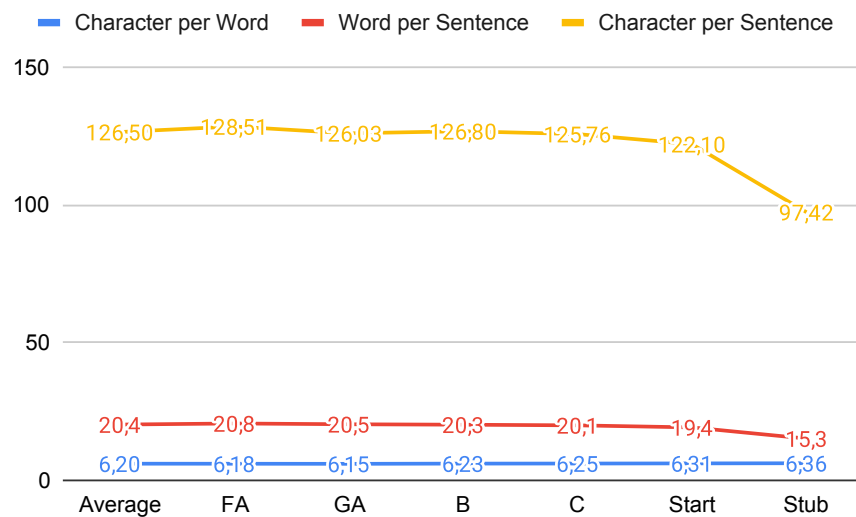


Figure 5.2: Average character per word, word per sentence, and character per sentence of each quality class

as shown in Figure 5.4. We can conclude that Wikipedia articles with more contributions tend to be of better quality, as would be expected. In fact, stub articles only average 28 revisions, while FA's have 1,462.

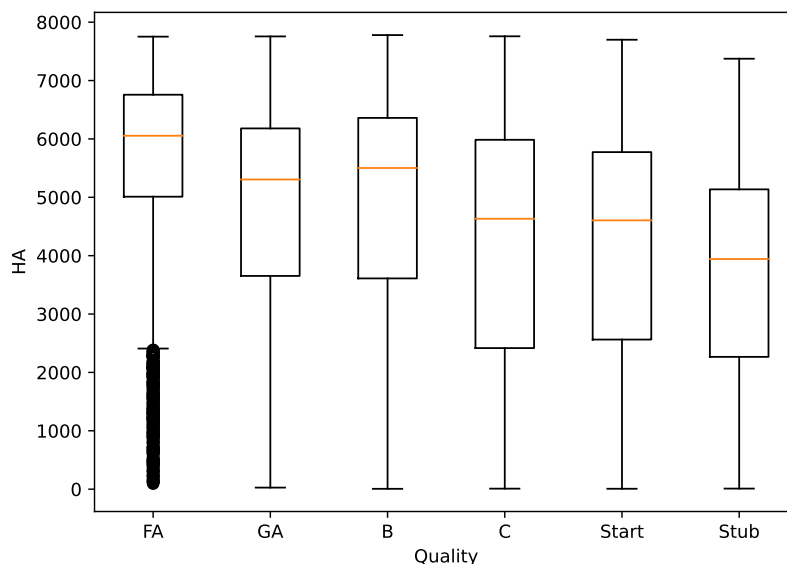


Figure 5.3: Average article age of each quality class

We also assessed the readability of our Wikipedia dataset. We compared the average grade level of each quality based on the readability formulas we applied. Figure 5.5 suggests no correlation between Wikipedia quality and readability, as the values only slightly increase or decrease between levels. Besides, we noticed that some formulas return significantly higher values than others. For example, the average Gunning Fog Index is 17.48, whereas the average value for Coleman-Liau is 12.56 among all articles.

5.2 Algorithm Performances

We collected multiple performance metrics from our experimentations. This section discusses the most relevant findings of the experiments with all features and classes (CSRHN6), but Appendix Section D.2 lists these results more extensively.

Figure 5.6 summarizes the accuracy of all the classifiers we trained with CSRHN6 parameters. We found accuracy to be a suitable metric, as our dataset is nearly perfectly balanced. Additionally, precision, recall, and f1-score values are almost identical, so any metric would have been appropriate. The best-performing model was the Support Vector Classifier, achieving 64% accuracy. Most of the other algorithms reached similar values, ranging from 60% to 62%. Only KNN, Decision Trees, and Gaussian Naive-Bayes performed relatively poorly, the last of which showed the worst results (45%).

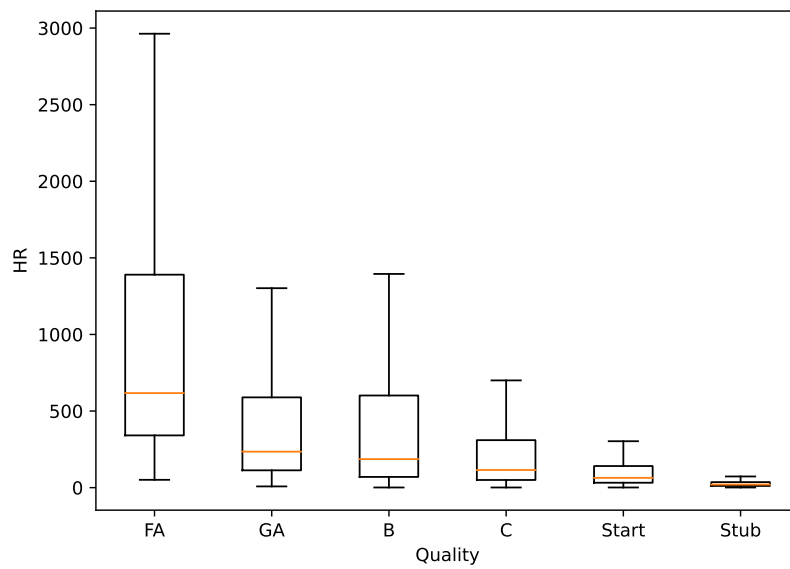


Figure 5.4: Average number of revisions of each quality class

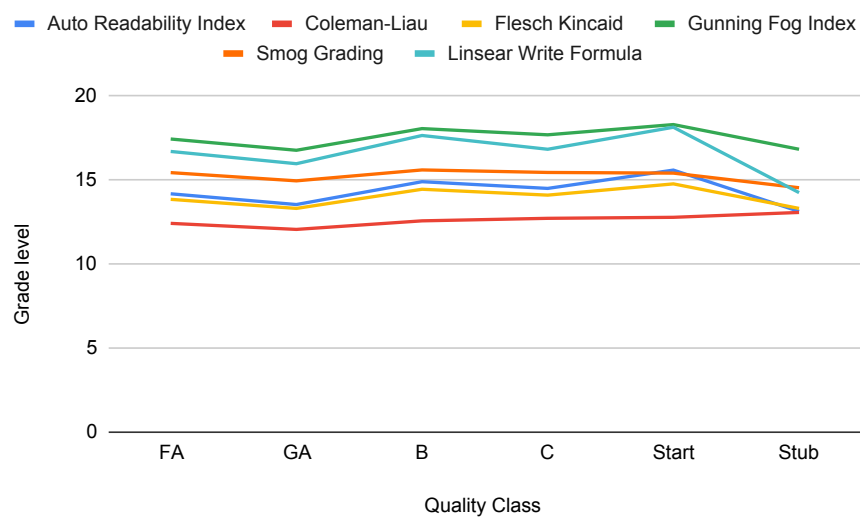


Figure 5.5: Average grade level of each quality class, calculated from various readability formulas

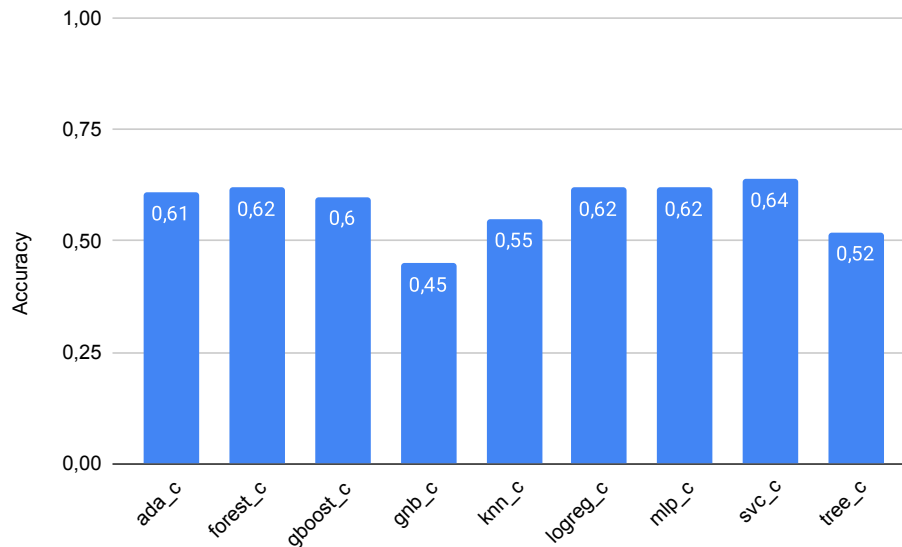


Figure 5.6: Accuracy of classification models trained with all features and classes (higher is better)

Then, we decided to analyze algorithm performance on a per-class basis because we suspected the models were performing worse on certain quality levels. We compared F1 scores, which harmonically reflect both precision and recall values. Figure 5.7 suggests all algorithms have more difficulty predicting the quality of mid-tier articles than the extreme ones, likely because B and C levels are very similar quality-wise.

To further understand where the models fail more often, we inspected the confusion matrix of CSRHN6/svc_c. A confusion matrix is a table that allows easy visualization of an algorithm's performance by specifying how many instances of each class were correctly predicted or predicted to be one of the others. Table 5.1 permits us to note a couple of observations:

- When the model predicts the wrong class, it tends to pick quality levels closer to the correct one.
- The model has more difficulty correctly predicting mid-tier articles (B and C-class) than the others.

Although we are currently analyzing the confusion matrix of the svc_c model, these points remain valid for all of the other classifiers.

We have also compared the performance of the multiple regression algorithms. Since accuracy is not applicable in these methods, we opted for showing the MAE of the predictions in Figure 5.8, as it is considered an appropriate indicator of the average error [98]. In this case, Random Forest has shown the best results (0.09), but almost all others display MAE values around 0.1. These findings indicate that the forest_r model will, on average, produce quality values off by 9%. We find that number acceptable for real-time usage, considering most quality levels are spaced out by 10% or more.

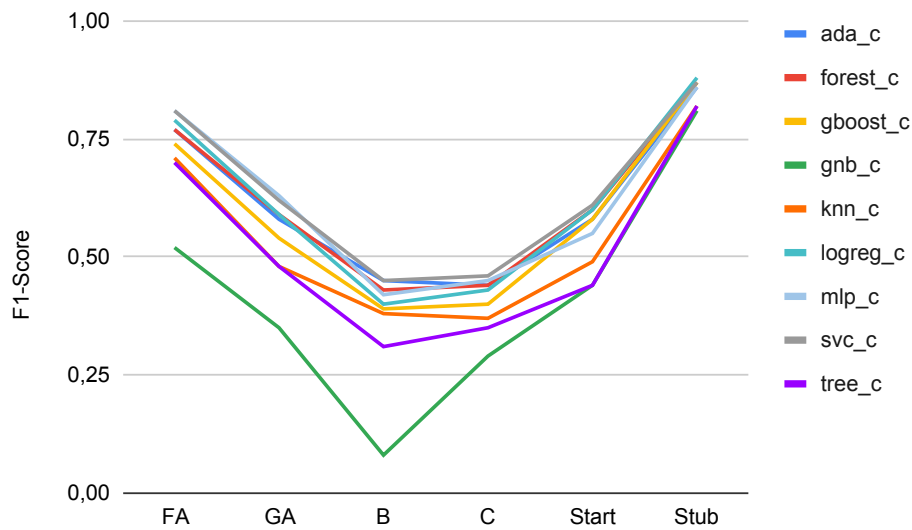


Figure 5.7: F1-score of classification models trained with all features and classes, per article quality (higher is better)

Table 5.1: Confusion Matrix of SVM classifier trained with all features

	pred:FA	pred:GA	pred:B	pred:C	pred:Start	pred:Stub
true:FA	1520	232	47	1	0	0
true:GA	325	1112	239	108	15	1
true:B	94	271	748	503	162	20
true:C	26	136	379	808	420	30
true:Start	1	49	114	326	1145	162
true:Stub	0	1	9	5	224	1559

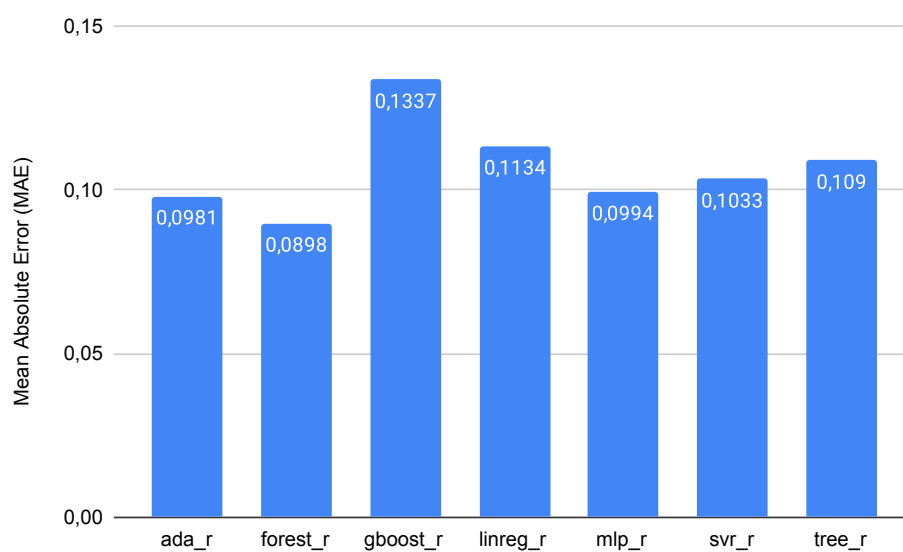


Figure 5.8: MAE of regression models trained with all features and classes (lower is better)

We performed a per-class analysis of the regression results, similarly to the classification algorithms. We compared the MAE of the model on each level of quality, summarizing our findings in Figure 5.9, which were not as consistent as the classification one. However, we can still extract the following conclusions:

- Gradient Boosting shows a unique pattern compared to the other algorithms, achieving the lowest error rate for mid-tier articles.
- Most algorithms peak the MAE on the Start articles, probably because its quality value (0.4) is more distant from its adjacent levels (Stub - 0.0, C - 0.6). Conversely, the error bottoms at featured articles because its only adjacent level (GA) is only 0.1 points lower.
- Otherwise, the error rates usually do not vary significantly throughout the multiple classes.

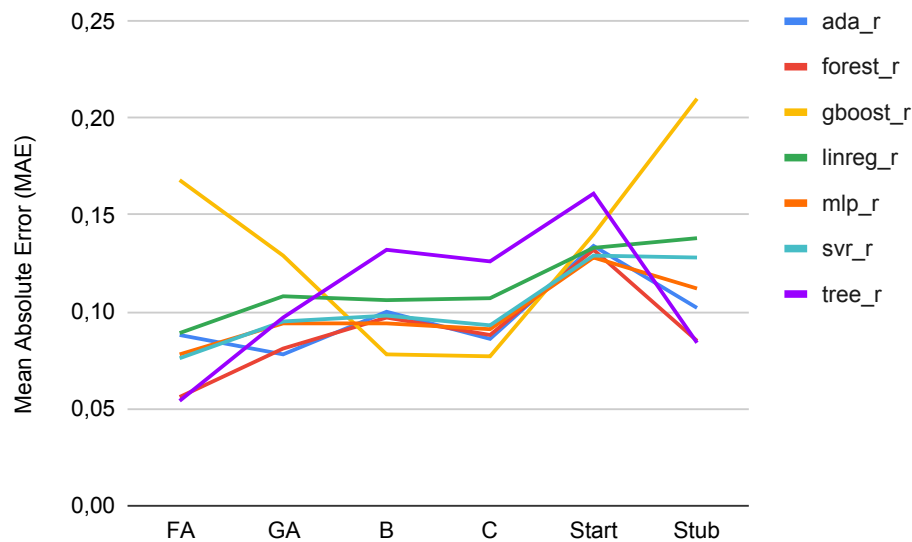


Figure 5.9: MAE of regression models trained with all features and classes, per article quality (lower is better)

5.3 Analysis of Number of Classes

Here we observe and analyze how reducing the number of quality classes impacts the performance of the models. Similar to the previous section, Appendix Section D.3 fully details the numbers behind this analysis.

We can first notice that model accuracy invariably increases as the number of classes decreases, as shown in Figure 5.10. Additionally, Support Vector Classifiers were consistently the best-performing models (*svc_c*). However, other algorithms achieved equal accuracy in some experiments.

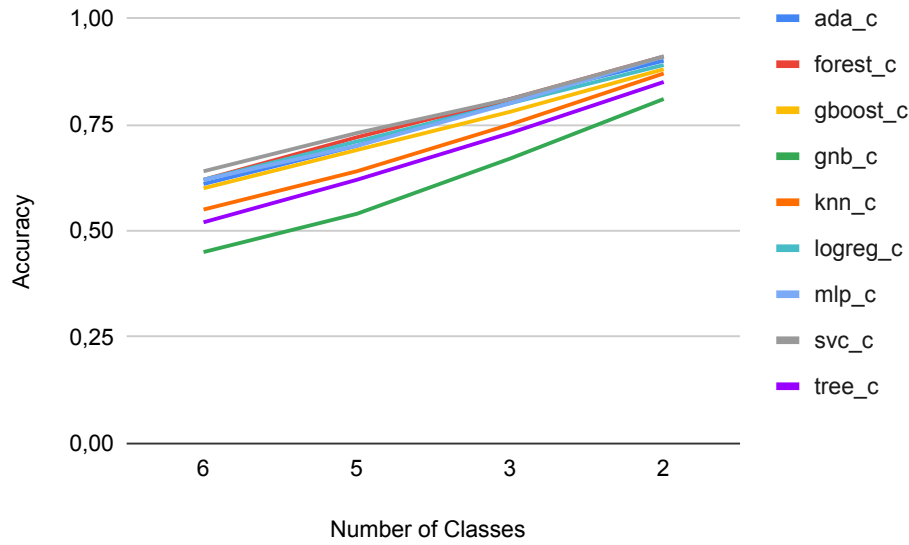


Figure 5.10: Accuracy of classification models trained with various numbers of classes (all features)

Table 5.2 lists the most effective algorithms for each number of quality levels, showing how performance improves when the number of labels decreases. The most notable improvement is the transition from 6 to 5 classes. Merging the B and C qualities leads to an accuracy increase of 0.09 while only decreasing the number of classes by 16.67%. For comparison, reducing from 3 to 2 quality levels boosts accuracy by 0.10 but decreases the number of classes by 33.33%. Table 5.3 shows another way to observe that phenomenon, comparing the effectiveness of each SVC model with a purely random one.

Table 5.2: Number of Classes: Most effective classifiers for each experiment

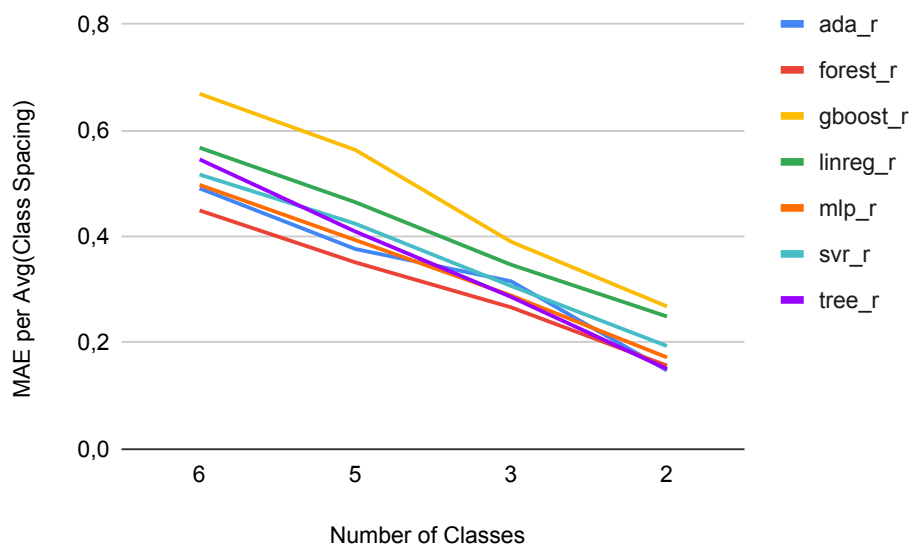
# Classes	Best Models	Accuracy
6	<i>svc_c</i>	64%
5	<i>svc_c</i>	73%
3	<i>ada_c</i> <i>forest_c</i> <i>svc_c</i>	81%
2	<i>forest_c</i> <i>mlp_c</i> <i>svc_c</i>	91%

Regarding regression algorithms, directly comparing the mean error does not give great insight into performance improvement, as fewer classes lead to more spaced-out levels of quality and may lead to a larger error. For that reason, we compared the values of $AvgDist = \frac{MAE}{AvgSpacing}$, where $AvgSpacing = \frac{1}{NumClasses-1}$. In other words, $AvgSpacing$ is the average distance between two levels of quality, and $AvgDist$ represents how many classes away will the model prediction be from the correct one. Figure 5.11 demonstrates the relationship between the number of classes and the

Table 5.3: Accuracy of Random models vs. SVC models, by number of classes

# Classes	Random	SVC	Difference
6	16.67%	64%	+47.33%
5	20.00%	73%	+53.00%
3	33.33%	81%	+47.67%
2	50.00%	91%	+41.00%

Avg_{Dist} of each regression model. For instance, a prediction from a *gboost_r* model trained with three classes will be, on average, around 0.39 quality levels away from the correct one.

Figure 5.11: Avg_{Dist} of regression models trained with various numbers of classes (all features)

For the sake of consistency, we show in Table 5.4 the best-performing algorithm for each number of quality levels.

Table 5.4: Number of Classes: Most effective regressors for each experiment

# Classes	Best Model	MAE	Avg_{Dist}
6	<i>forest_r</i>	0.0898	0.4490
5	<i>forest_r</i>	0.0878	0.3512
3	<i>forest_r</i>	0.1333	0.2666
2	<i>ada_r</i>	0.1479	0.1479

These results, both classification and regression ones, are based on a balanced dataset. If we had trained and tested the models with a different class distribution, accuracy and error reports would not have been the same. Regardless, our approach makes the results more reliable and coherent.

5.4 Feature Category Importance

Subsequently, we collected and compared the performance metrics of models trained with various feature sets. This section summarizes that investigation, while Appendix Section D.4 discloses the results in more detail.

Overall, the patterns do not vary significantly between algorithms. That is, if a particular algorithm performs worse with a specific feature set, other algorithms will follow suit. As a result, we decided to focus this analysis solely on the *svc_c*, which is the best-performing algorithm in every feature set.

We first examined how the model performed when removing just one feature category. Figure 5.12 suggests that content and history features are more important than the others, but removing one of the categories generally does not significantly hurt performance. That chart also indicates that network features are not too crucial for accurately predicting an article’s quality, making our machine learning approach more appropriate for real-time assessment.

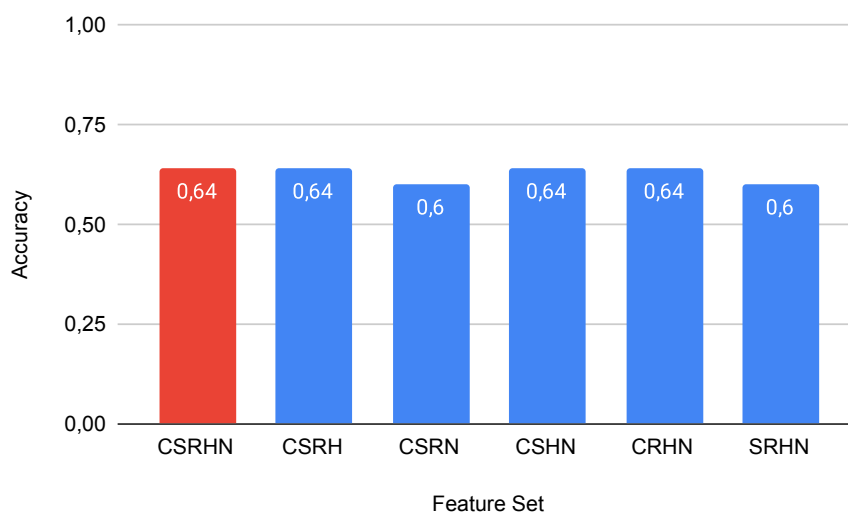


Figure 5.12: Accuracy of SVC models trained on feature sets with four categories

Afterward, we aimed to assess model effectiveness when trained with only one feature category to determine which were the most “independent” features. Looking at Figure 5.13, we can observe that the model still performs adequately when simply using content features. However, using only history, network, and readability features leads to much poorer performances, indicating these features may not be as strong as the others by themselves.

Finally, we also analyze how the model performs with medium-sized feature sets for potential real-time application. Figure 5.14 indicates that performance does not drop after removing both network and readability features. Although the history category does not lead to an enormous accuracy boost, it is still a meaningful increase for the purposes of this study.

Regression algorithms follow roughly the same pattern, as shown by Figures 5.15, 5.16, and 5.17, which show the effectiveness of the *forest_r* models.

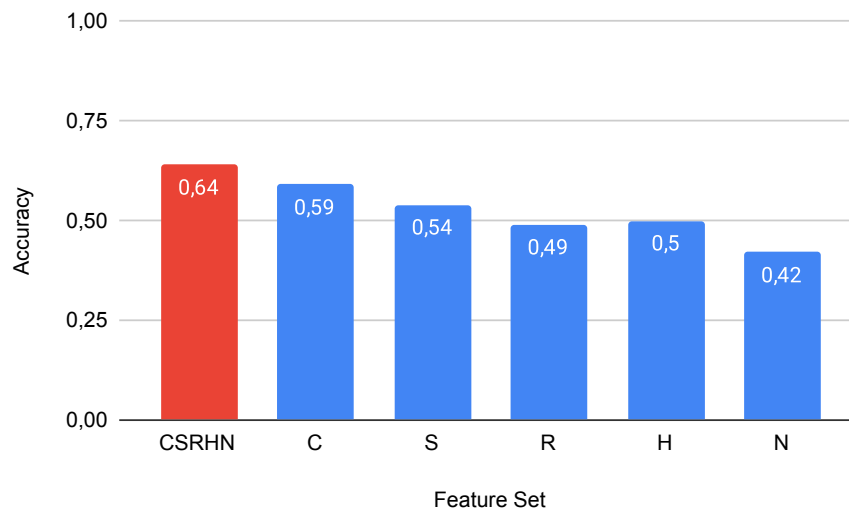


Figure 5.13: Accuracy of SVC models trained on feature sets with one category

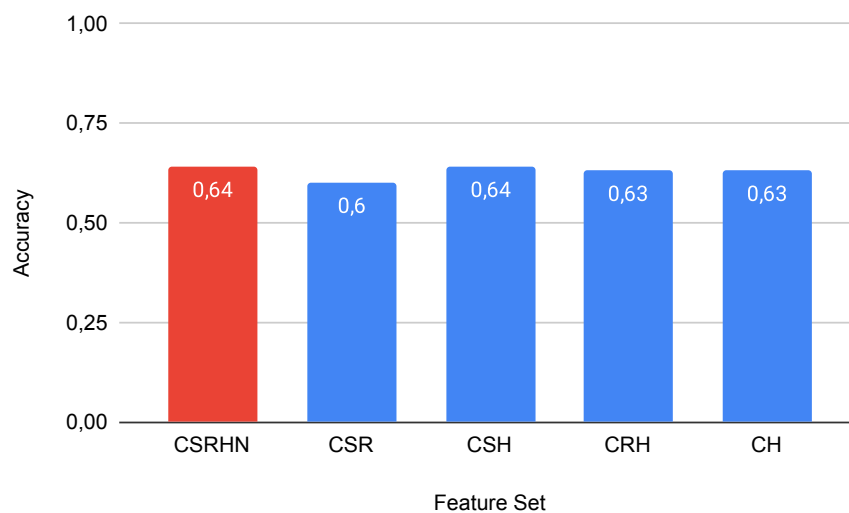


Figure 5.14: Accuracy of SVC models trained on other feature sets

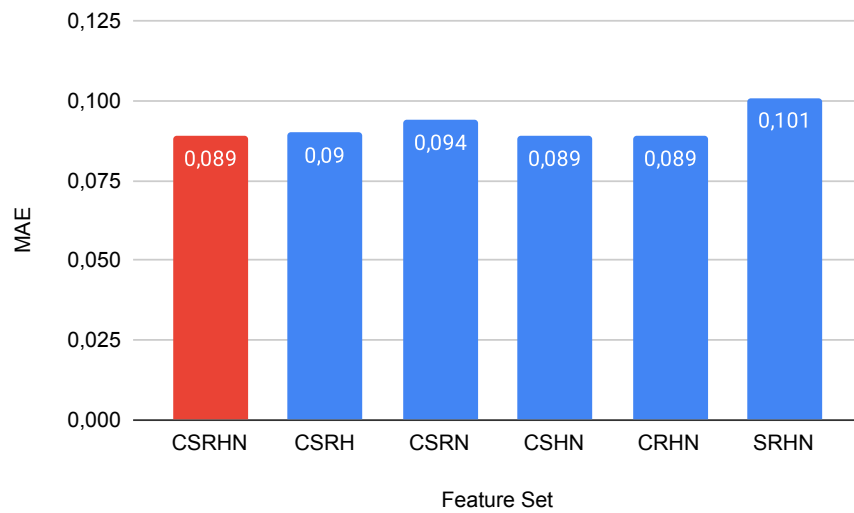


Figure 5.15: MAE of Random Forest regression models trained on feature sets with four categories

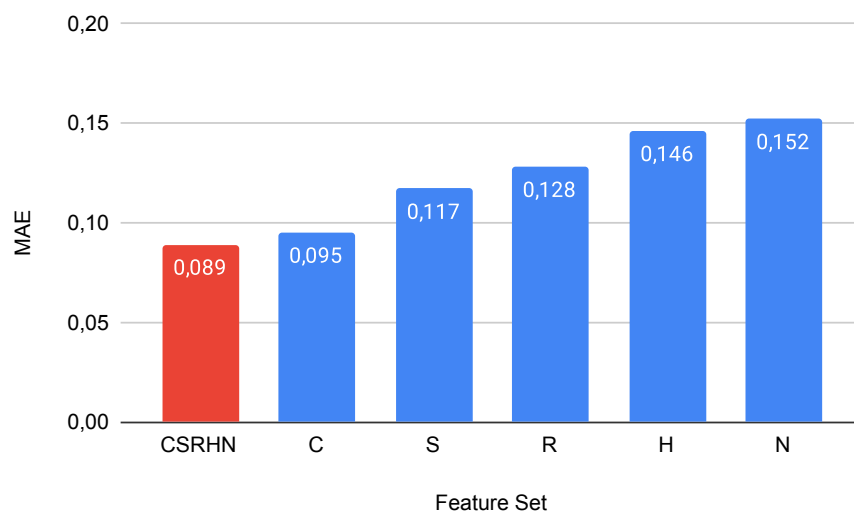


Figure 5.16: MAE of Random Forest regression models trained on feature sets with one category

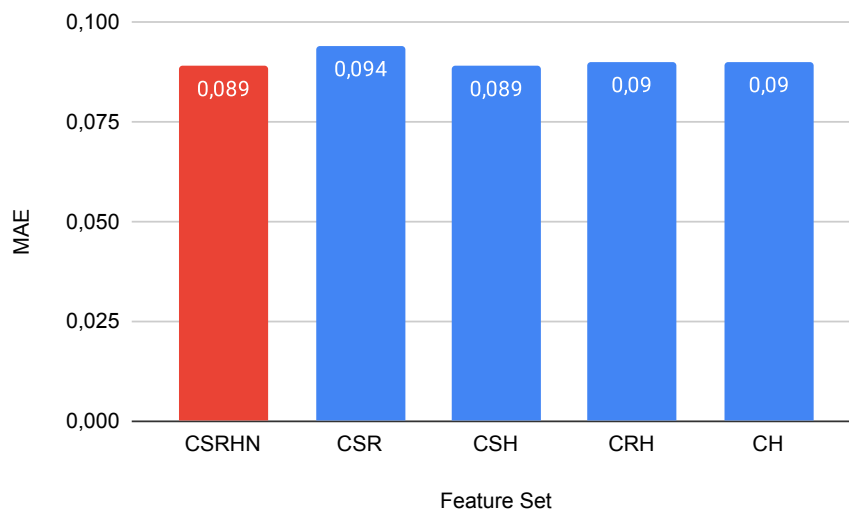


Figure 5.17: MAE of Random Forest regression models trained on other feature sets

Overall, content features are certainly the most relevant and work great without the aid of other categories. Still, pairing them with style and history creates an even more robust feature set for predicting the quality of Wikipedia articles.

5.5 Training and Prediction Times

This section examines and compares training times and prediction times for distinct models. For the complete results, refer to Appendix Section D.5.

5.5.1 Training Times

We gathered the training times of each model and arranged them to observe better which algorithms train faster in certain circumstances. We first aimed to evaluate the effect of the number of classes on the model training time. Figure 5.18 better displays that relationship, showing that, for most algorithms, the number of labels has no significant impact on the algorithm's training time². However, some methods (*gboost_c*, *svc_c*, *forest_c*, and *logreg_c*) appear to train faster as the number of classes decreases

Afterward, we performed a similar analysis over different feature subsets. Figure 5.19 shows the training times for every algorithm and used feature categories. The horizontal axis is sorted in descending order of feature count, demonstrating that increasing the input data's dimensions boosts training time significantly. We used the naming convention we defined in Chapter 4 and the category subsets listed in Table 4.18.

²For this analysis, we did not balance the 5-class and 2-class datasets as mentioned in Section 4.4.2, since reducing the number of rows would likely have an unwanted impact on the observed training time.

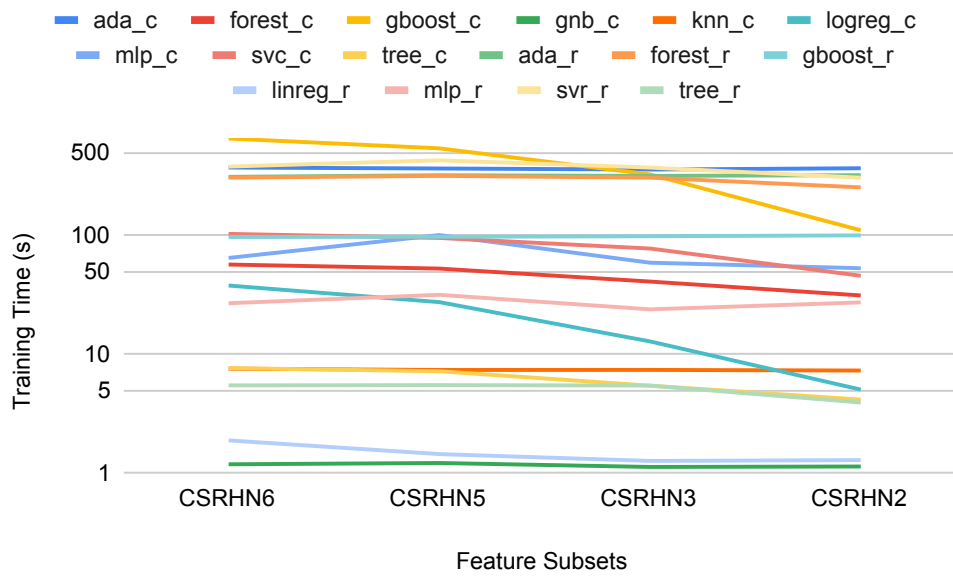


Figure 5.18: Algorithm training times with different numbers of classes (Refer to Appendix Table D.9 for the full list of measurements)

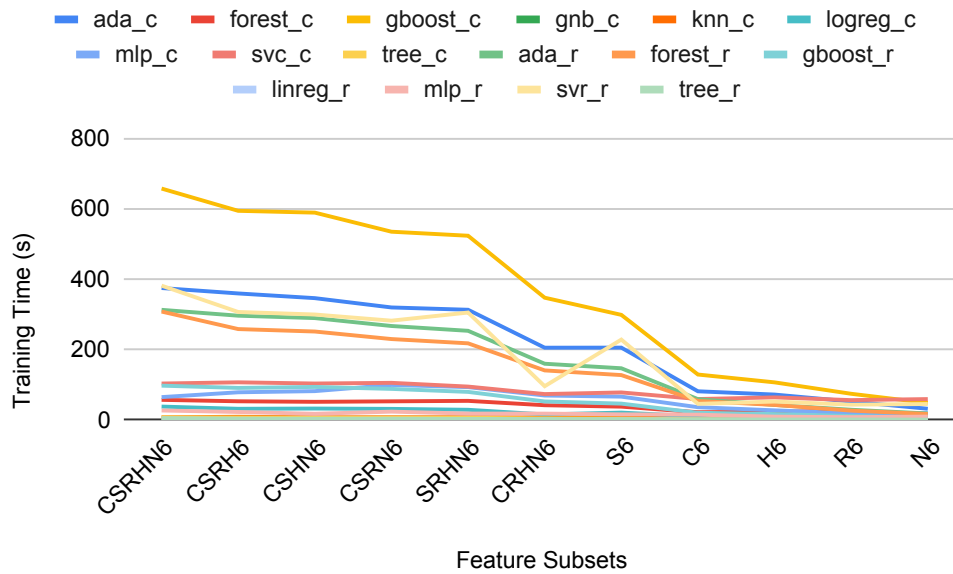


Figure 5.19: Algorithm training times with different feature subsets (Refer to Appendix Table D.9 for the full list of measurements)

Since all the algorithms appear to roughly share the correlation between classes/features and training time, we considered any combination to be a good relative indicator of an algorithm's training time. Therefore, we have compared all the measurements for CSRHN6 to visualize better which algorithms are generally slower to train. Figure 5.20 shows that ensemble methods (e.g., Gradient Boosting, Ada Boost, Random Forest) require more computational effort and thus are slower to train. That observation is not surprising, considering ensemble algorithms combine multiple methods to predict new examples [31]. All other algorithms took around 90 seconds or less to train, except for SVR, which was slower than some ensemble methods.

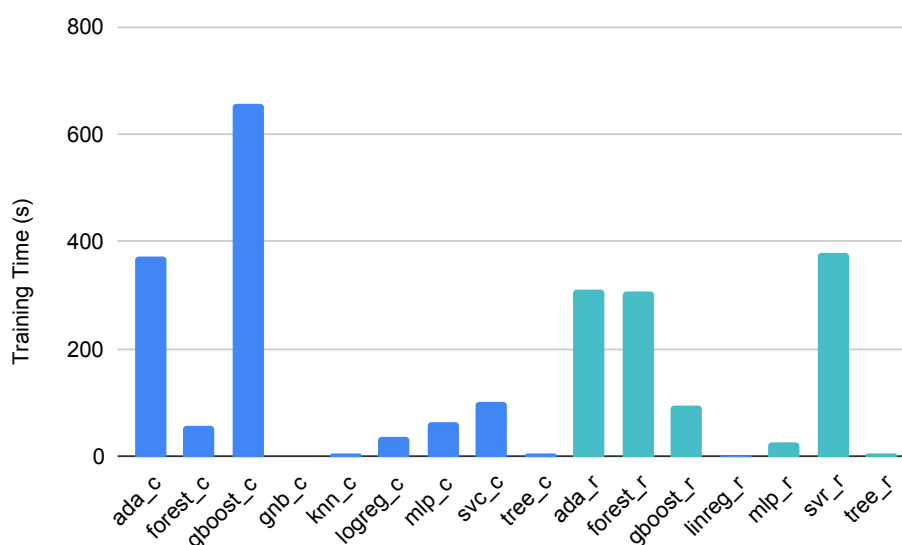


Figure 5.20: Algorithm training times with six classes and all feature categories

5.5.2 Model Prediction Times

Upon further inspection, models almost instantaneously convert the features into a quality value. We measured no value above 25 milliseconds, and most are much lower. The number of classes and features also appear to have no significant impact on the prediction time of most models, as indicated by the small standard deviation values shown in Figure 5.21. The most notable exception is the KNN model, which scans all the training data every time it calculates a prediction [46], so datasets with more features will take longer to predict.

5.5.3 Feature Computation Times

Finally, we aggregated the collected measurements to determine which factors most influence the time to calculate the features. Figure 5.22 shows that calls to the API significantly slow the process, as fetching the wikitext and the revisions account for over 75% of the time spent. NLP processing (tokenizer, syllable estimator, and POS tagger) also takes up a considerable portion of computer time, totaling an average of 15%. The other tasks are executed relatively instantly.

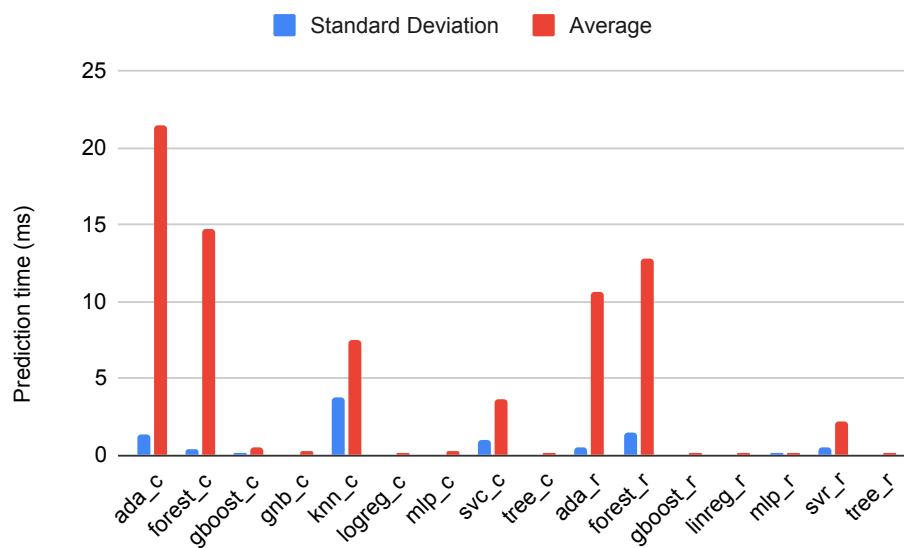


Figure 5.21: Average algorithm prediction times across different feature subsets and number of classes

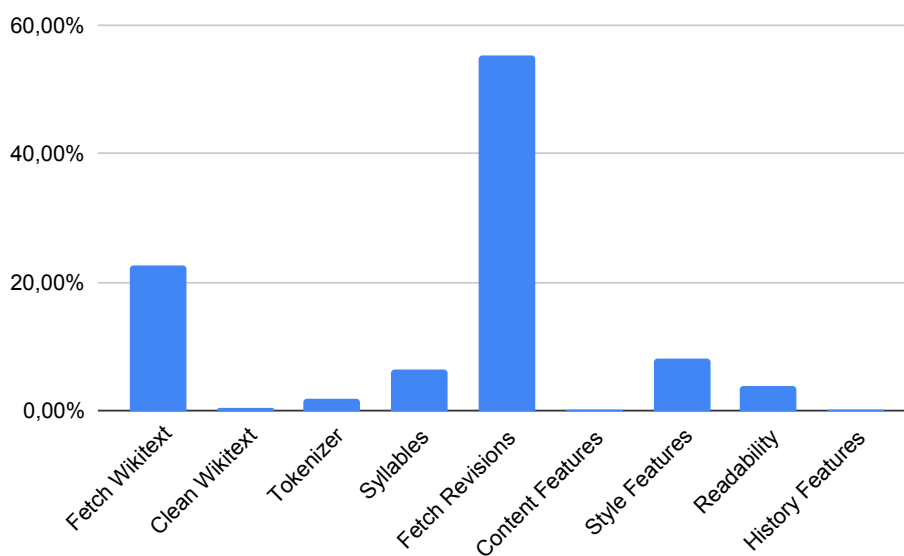


Figure 5.22: Average percentage of total time spent per feature computation task

Afterward, we combined the results with the task requirements for each feature category to determine which ones have the most significant impact on increasing computation time. We did that by adding up the tasks involved in each category, representing the results in the graph in Figure 5.23. We conclude that history features are responsible for most of the time spent, as we must collect the article’s entire revision history from MediaWiki’s API. Contrarily, if we only used content features, prediction times would drastically decrease since they are the fastest to compute.

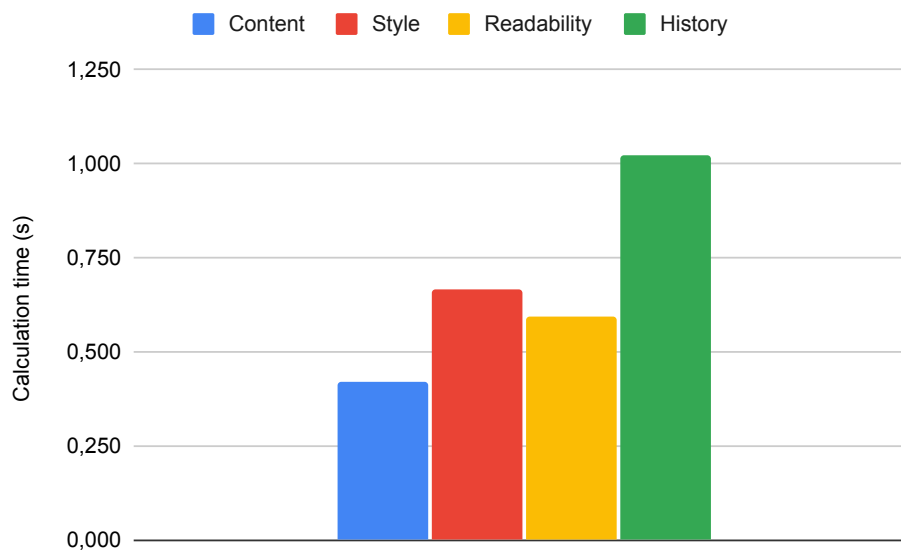


Figure 5.23: Total prediction time of feature computation tasks related to each feature category

Figure 5.24 displays a box plot of all the recorded times per quality level. Higher-quality articles take longer to predict, but even B and C-class can sometimes reach more extreme values (9-10 seconds).

Regardless, the program predicts the quality of most articles in 4 seconds or less, which indicates our approach is perfectly suitable for real-time quality assessment. Besides, Starts and Stubs show very low values for their measurements, likely due to their lower character and revision count. Therefore, most real-time predictions will have even shorter computation times since most Wikipedia articles are of poor quality.

5.6 Multi-language Dataset Analysis

After constructing a dataset for every selected language, we aggregated the articles of each quality to analyze how their distribution varies across the different Wikipedia versions. We illustrate that information in the pie charts represented in Figure 5.25 and detail their respective numbers in Table 5.5.

We can notice that low-quality articles are the most prevalent, regardless of the Wikipedia version. Although different Wikipedias show different concentrations of good articles, we should note

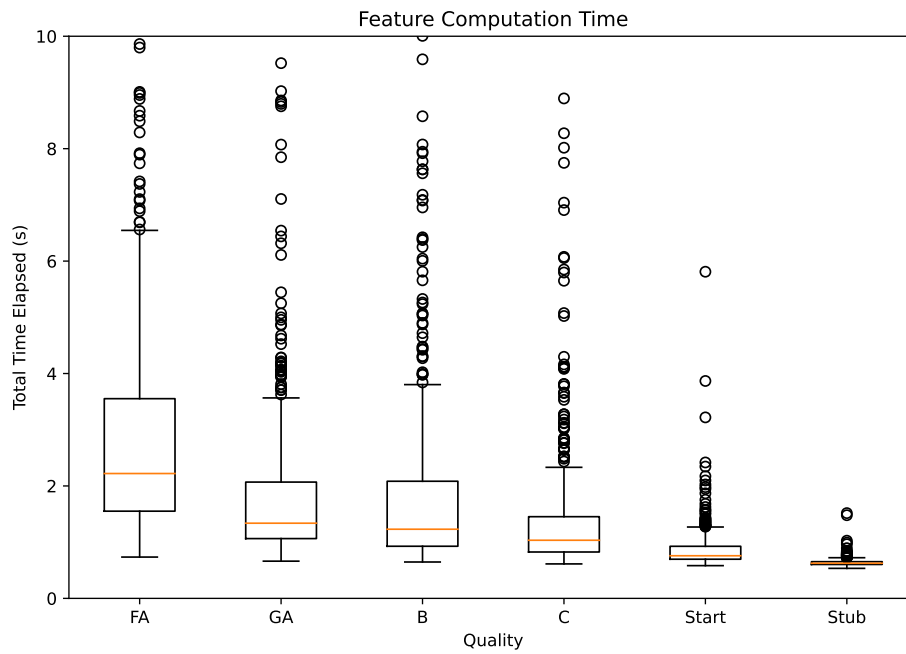


Figure 5.24: Boxplot of feature computation times per quality level

Table 5.5: Quality distribution across datasets constructed from different Wikipedias

Language	A+	A	B	C	D	F	Total
English	13	48	199	725	3,860	6,251	11,096
French	11	30	18	334	3,292	7,510	11,195
Portuguese	77	83	55	255	1,852	8,203	10,525
Russian	50	100	284	745	3,048	6,114	10,341

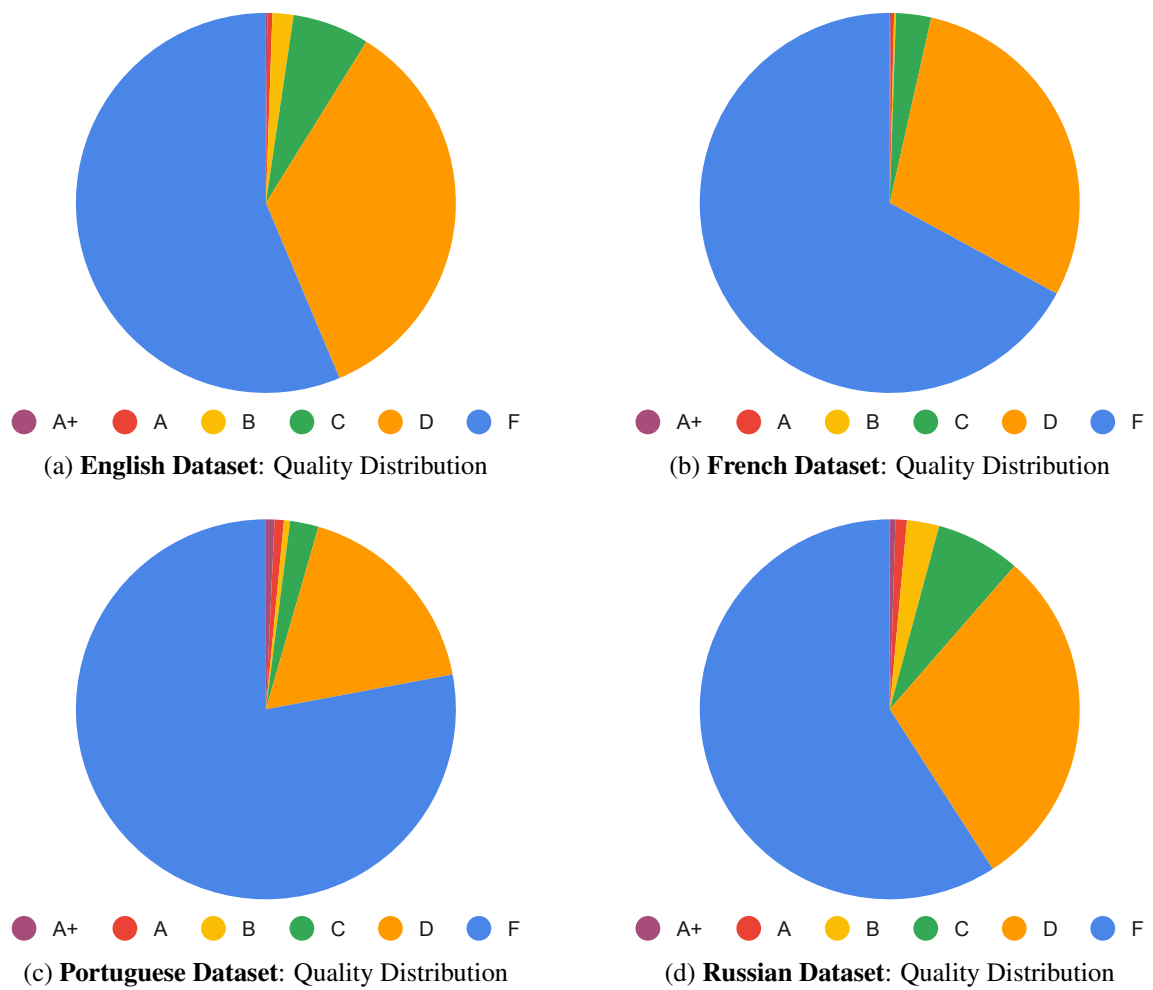


Figure 5.25: Quality distribution across datasets constructed from different Wikipedias

that reviewers and their standards likely vary between versions, so featured articles are not necessarily equally good in every language. Also, top-tier articles are generally a minuscule portion of Wikipedia (around 0.1% for Featured Articles in the English version [93]), so even a sample of 10,000 items may not necessarily represent those distributions perfectly accurately.

Regardless, we can still consider these statistics to be a reliable portrayal of the average Wikipedia article and can observe the following patterns:

- As quality decreases, article frequency generally increases exponentially, regardless of the Wikipedia version. The chart in Figure 5.26 visually demonstrates that phenomenon.
- French Wikipedia has the lowest number of above-average articles, with only 59 (0.5%) ranking B or higher.
- Compared to other versions, Portuguese Wikipedia contains more high and low-tier articles and fewer mid-tier ones.

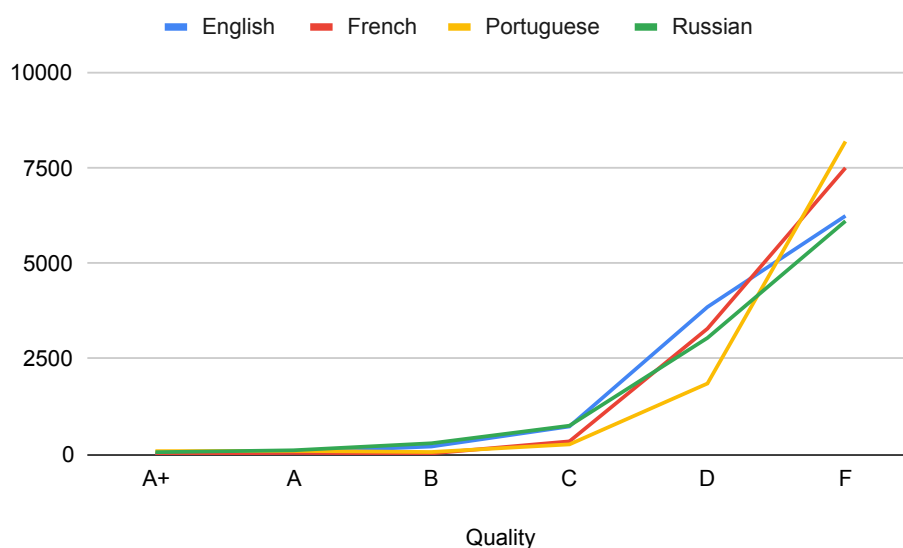


Figure 5.26: Comparison between the quality distribution of different Wikipedia versions

We also analyzed and compared other relevant statistics between Wikipedia translations, detailing the full results in Appendix Section D.6. For instance, as shown in Figure 5.27, Portuguese articles are generally much shorter than their peers, averaging roughly half as many characters and words as Russian pages. These numbers are not surprising, considering Portuguese Wikipedia has many more poor-quality articles, so we also analyzed text lengths of A+ articles only. Naturally, this analysis is not as reliable as the previous one, considering these samples contain very few items. Nonetheless, we concluded that well-rated articles usually are much longer than the average. We also noticed that A+ articles use fewer characters per word than average, which correlates with a more straightforward and concise text.

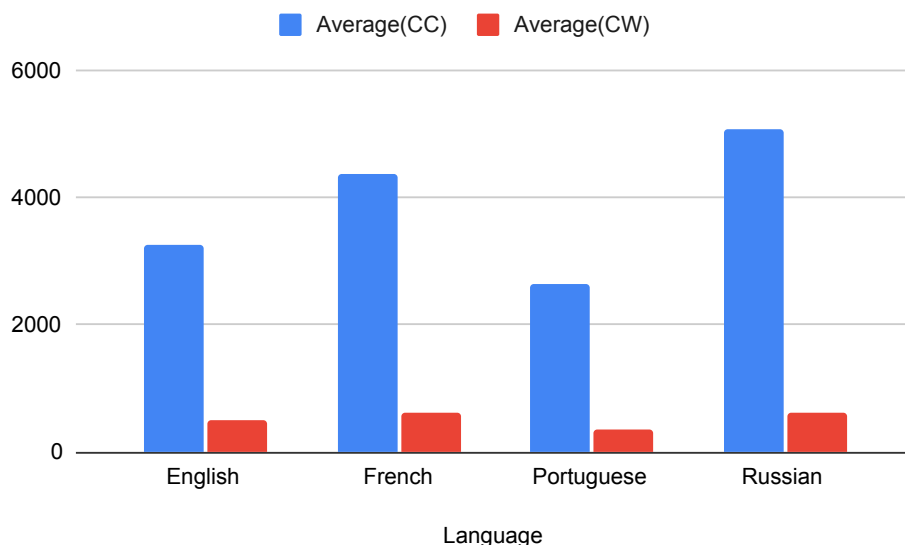


Figure 5.27: Comparison between Average Character Count (CC) and Word Count (CW) of different language datasets

5.7 Multi-language Model Performance

We used the dataset described in the previous section and assessed the performance of three models: *CSRH6/forest_r*, *CRH6/forest_r*, and *CH6/forest_r*. Figure 5.28 compares the Mean Squared Error (MSE) for the predictions of each model in every language. We do not assess the MAE as we had previously done because these datasets are much different from the training data, so we want to give more weight to larger errors. Nonetheless, we detail all the extracted results (including the MAE values) in Appendix Section D.7.

The results show that we can maintain the prediction error at an acceptable rate, even in other languages. The MSE is slightly higher than we had previously observed, looking solely at the English results, likely due to the drastic differences in article quality distribution between the two English datasets.

We also noticed that, on average, the model fares better against Portuguese articles. The relatively lower abundance of mid-tier Portuguese articles may explain that phenomenon, as it is more trivial to distinguish between articles that are much more distinct quality-wise. The minor differences between results of other versions likely occur because the models were trained with an English dataset, so variations are expected.

We can also observe that, except for the English version, using fewer features decreases model performance. That outcome is surprising, considering that style and readability features are supposed to be much more accurate for English texts. Regardless, most machine learning models are typically known as black boxes. That is, even though they can reach outstanding levels of accuracy, it is rarely clear how the model arrives at its prediction value, so unexpected results often occur [71].

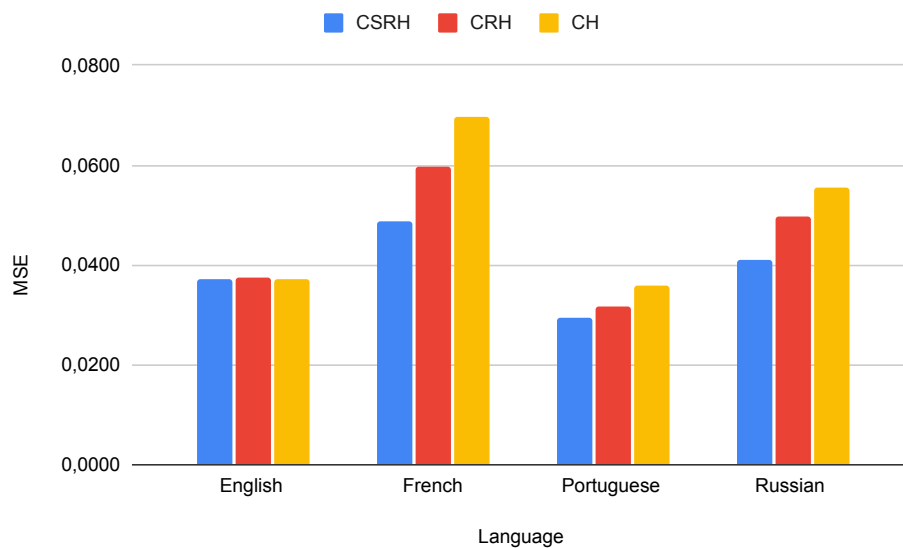


Figure 5.28: Comparison between MSE of model predictions across different dataset languages

5.8 Final Remarks

We have successfully trained well-performing models for automatically predicting Wikipedia quality. Our solution matches or beats most state-of-the-art approaches, comparing our results with those we collected during the literature review (see Table 3.13). Some results cannot be effectively compared with ours, though:

- Dang & Ignat’s [29]: Although they achieve accuracy values of 68%, they mention that “the predicting time of RNN-LSTM is too long to be implemented as a real-time service”, which would serve no purpose for this project. Regardless, 68% is not too far off our results.
- Dalip et al. [27]: It is not entirely clear how to interpret the MSE of their approach, so it is not easy to compare their results with ours.
- Blumenstock’s [14]: This study designs an ML solution that can distinguish a featured article from a random one at an accuracy of 97%. Even though this model only uses text length to make its prediction, we have no information about the random article distribution. For instance, it would be elementary to distinguish between a featured article and a stub. Besides, our 2-class models also considered good articles as part of the better tier, making our study even more different from Blumenstock’s.

Analyzing the performance of our models and the prediction times of each category, we believe that a CSRH feature set (all but network) is the most appropriate for real-time implementation. Although history features significantly increased prediction time, it is still manageable, and the performance boost makes up for that issue. Additionally, we have shown that keeping the style and readability features improves performance for predicting non-English articles.

Chapter 6

Chrome Extension Development

This chapter details the development process of the Google Chrome extension that applies the model we trained, explaining its architecture and functionalities. The extension is available for download in <https://up201705208.wixsite.com/wikiquality>.

6.1 Architecture

We opted for a straightforward client-server architecture for this application. As suggested by Figure 6.1, when scanning a Wikipedia page, the browser will collect the article's title and request a quality assessment from our server. The server, hosted in Heroku, will use MediaWiki's API to obtain the respective wikitext and revision history. Using a previously trained model, it converts the features into a quality value, returning it to the user.

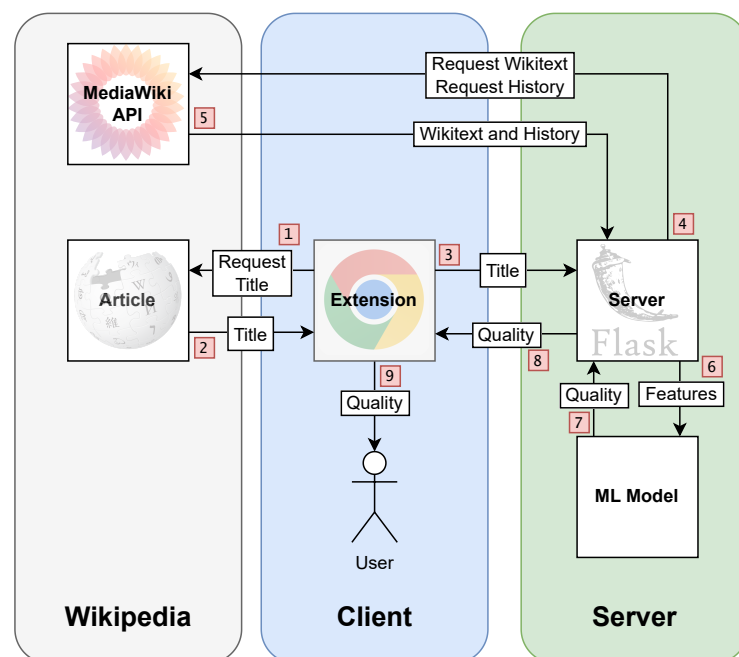


Figure 6.1: Chrome Extension Architecture

6.2 Client

This section describes the client part of the application, its features, and respective mockups. We have also included additional mockups in Appendix E.

The application layout is straightforward. When the user opens a Wikipedia page, they can open the extension popup and assess its quality by pressing the *Scan* button, as suggested by the mockup in Figure 6.2. If the current browser tab is not from Wikipedia, user interaction will be disabled, as shown in Figure 6.3

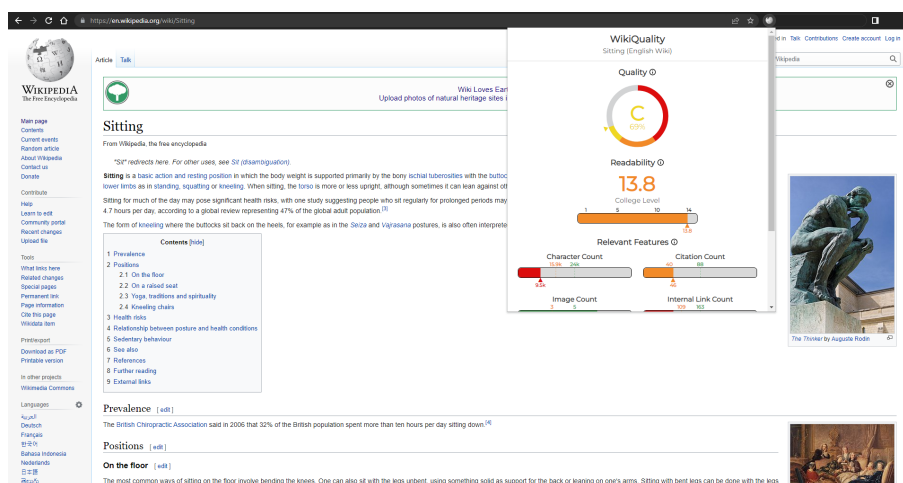


Figure 6.2: Chrome extension mockup: application layout

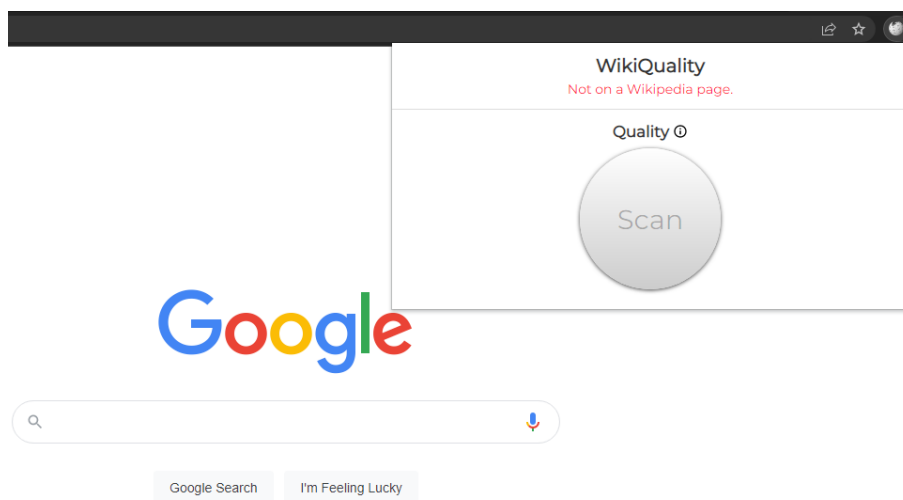


Figure 6.3: Chrome extension mockup: functionality in a non-Wikipedia page

6.2.1 Quality Prediction

Quality prediction of Wikipedia articles entirely summarizes the topic of our study, so this feature will naturally be the central part of this extension. However, the evaluation process is simple:

the extension obtains the URL from the current tab, extracts the article’s title, and requests an evaluation from the server. We then convert the quality prediction to a letter grade and display it to the user, as shown in Figure 6.4.

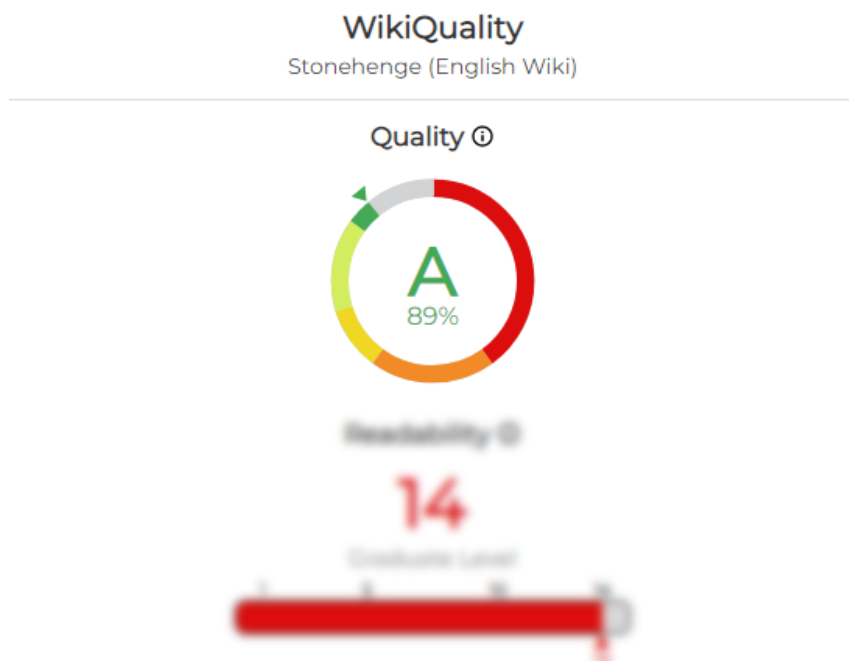


Figure 6.4: Chrome extension mockup: quality prediction

The quality levels displayed in the quality meter do not directly match the classes we used during the ML training phase. To make it more intuitive to a regular user, we slightly adjusted the block sizes to make the scale similar to the grading system usually seen in schools. Table 6.1 details the mapping we used to convert a quality value to a letter.

Table 6.1: Chrome extension letter grading system

Letter	Range	Range (%)
A+	[0.95, 1.00]	[95%, 100%]
A	[0.85, 0.95[	[85%, 95%[
B	[0.70, 0.85[	[70%, 85%[
C	[0.60, 0.70[	[60%, 70%[
D	[0.40, 0.60[	[40%, 60%[
F	[0.00, 0.40[	[0%, 40%[

6.2.2 Feature Analysis

Apart from computing the article quality and displaying it to the user, the extension also summarizes the values of relevant features, providing readers and editors with a more detailed explanation of where the article needs improvement. We first selected eight features that we previously determined to be heavily correlated with quality. Then, for each feature, we processed the dataset

to obtain the values corresponding to the 25% and 50% percentiles of FA's, providing us with quality thresholds where most featured articles fit. Table 6.2 lists the values we obtained for each highlighted feature.

Table 6.2: Highlighted features and respective FA percentiles

ID	Feature Name	P25	P50
CC	Character Count	15,934	24,002
CW	Word Count	2,595	3,908
CSN	Sentence Count	125	187
CCC	Citation Count	40	88
CI	Image Count	3	5
CIL	Internal Link Count	109	163
HA	Age (days)	5,010	6,054
HR	Revision Count	341	617

The percentiles for every feature are pre-calculated, so the extension can immediately determine if the computed value is inside the optimal range. As suggested by the mockup in Figure 6.5, the extension displays a pointer and a meter where P25 and P50 are highlighted. Values above P50 will be displayed as green and values above P25 are displayed as orange, while results outside that range are shown as red.

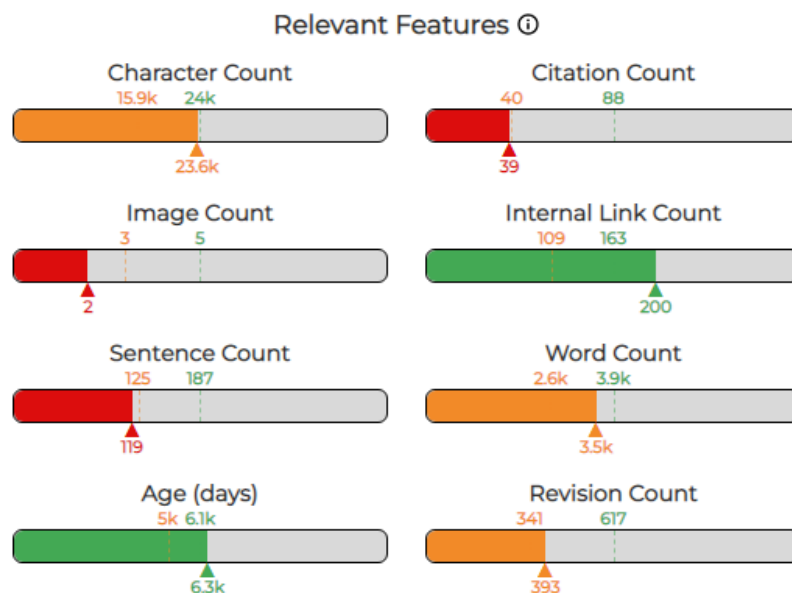


Figure 6.5: Chrome extension mockup: feature analysis

6.2.3 Readability Score

We concluded in Chapter 5 that readability features do not strongly correlate with quality because, on average, the numbers do not significantly vary between classes. However, that does not mean that the values do not differ between different Wikipedia articles. We thus decided to give Wikipedia users some insight into how easily read an article is by combining the results of multiple readability features. The mockup in Figure 6.6 better displays that functionality, showing the computed readability score of a Wikipedia article.

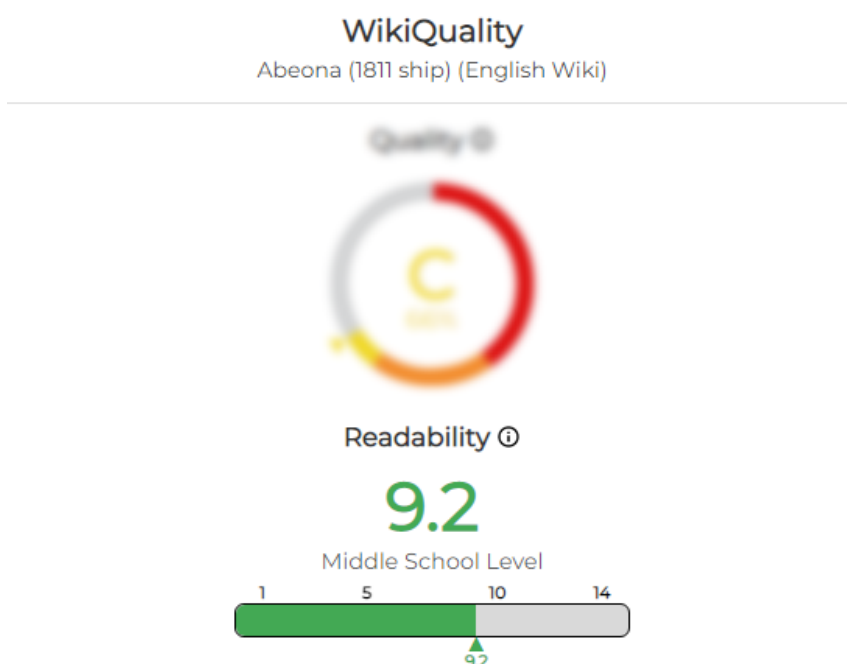


Figure 6.6: Chrome extension mockup: readability score analysis

We calculated the readability score based on six readability features, all of which aim to estimate the grade level required to fully comprehend the article's text. However, the same article may produce very different grade levels depending on the formula, so we first normalized the features to reduce the gap between various formulas. The readability score will then be the average of the adjusted readability values. Mathematically,

$$Score = \frac{\sum(R_i + Adj_i)}{|R|} \quad (6.1)$$

where R_i represents the value of readability feature i , and Adj_i is the average difference between Flesch-Kincaid's grade level (RFKI) and that feature, rounded to the nearest half. This formula produces a result that gives equal importance to all the used features while avoiding significant variance between them. Table 6.3 lists the Adj_i values for every readability feature we used. We calculated these values of Adj_i based on the analysis of our dataset, as detailed in Appendix Table D.2.

Table 6.3: Feature adjustments for computation of readability score

Feature i	RARI	RCL	RFK	RGFI	RSG	RLWF
Adj_i	-0.5	1.5	0	-3.5	-1.5	-2.5

6.3 Server

We designed a REST API that accepts basic requests related to the analysis of Wikipedia articles. Its main functionality is to evaluate the quality of Wikipedia articles, and so that endpoint will be the most important. More precisely, it is the only request that the extension uses, but we also found the other features potentially helpful too. Table 6.4 lists the endpoint URLs of every method we defined.

Table 6.4: Available server endpoints

Endpoint URL	Arguments	Description
/evaluate/<title>[/<lang>]	<title>: Article title <lang>: Version code	Calculates the features and evaluates the quality of the specified Wikipedia article, from 0 to 1. Default <lang>: en.
/wikitext/<title>	<title>: Article title	Obtains the raw wikitext of the specified Wikipedia article.
/plain/<title>	<title>: Article title	Obtains the plain text of the specified Wikipedia article.
/features/<title>	<title>: Article title	Calculates the features of the specified Wikipedia article.

We used the `CSRH6/forest_r` model, as it achieved the best performance in most experiments. As previously discussed, we excluded network features from the application as it would not be computationally feasible to compute them in real-time. The other categories lead to measurable performance gains without significantly hurting run times. Although we also considered our classifier results very good for six classes, we thought regression to be a more appropriate and reliable method.

Chapter 7

Conclusion

Wikipedia is a vast encyclopedia with an article for almost every topic, and it is continuously improving thanks to hundreds of thousands of volunteering editors. Nonetheless, there is still a significant quality disparity between the millions of existing Wikipedia articles. Less than 10% of the English content is considered “useful to a casual reader”, and around 0.1% have achieved featured status, exemplifying “Wikipedia’s very best work” [93]. Therefore, it is difficult for regular users to efficiently determine the quality of the article they are reading and for editors to discover areas to improve.

There exists a vast amount of literature surrounding this topic, from manual approaches assessing the overall quality of specific Wikipedia categories to machine learning solutions that automatically predict article quality with acceptable accuracy. Across the 22 publications we included in our systematic literature review, we gathered 165 distinct article features, compared the results of 47 different experiments, and concluded the best ML models achieve an accuracy of 60 to 65%, using six levels of quality. However, very few studies experiment with regression approaches, which would be more appropriate in predicting a linear value, like a quality level. Besides, the works that propose using regression algorithms consider Wikipedia’s quality scale is evenly spaced, which we determined was an incorrect assumption. We also did not find any papers presenting practical applications designed for this purpose, which would have been valuable for every Wikipedia user.

We developed our machine learning solution based on the existing work on the research subject. We constructed a dataset of around 36,000 English articles, equally split between six different quality levels. The feature set combined elements collected during the literature review and some original ones, totaling 145 distinct features across five categories. We conducted various experiments and trained 288 models, comparing the performance of several classification and regression algorithms, how reducing the number of classes affects model effectiveness, and which features better assist in predicting article quality accurately.

We achieved a classifier accuracy of 64% and a Mean Absolute Error (MAE) of 0.09 in regression methods. Although these are the numbers relative to our best results, other algorithms had similar performance. Decreasing the number of classes naturally improved the model's accuracy, but we considered the 5-class experiments to produce the most significant relative improvement, with an accuracy of 73%. Regarding feature importance, we found content features to be much more effective than the rest, but mixing them with the other categories led to a significant performance boost, especially history features, so we did not exclude them immediately. We only opted to exclude network features, as their calculation requires substantial computational effort for the benefit it generates, making them inappropriate for real-time usage. Finally, we tested our model in randomly generated datasets of non-English Wikipedia articles. The results varied, as it proved to be less effective in some versions and better in others, but, on average, we can expect this solution to perform slightly worse in non-English Wikipedias.

Our investigation also included an analysis of the constructed Wikipedia datasets. We confirmed our assumption that numerous elements strongly correlate with article quality, namely character/word count, number of reviews, and citations. We then discovered that quality distribution varies across different Wikipedia versions. For instance, Portuguese articles are much less likely to be mid-tier than their peers are, as the percentage of high and low-quality entries is much more significant. These conclusions further prove the need for a straightforward method to detect how good an article is.

We studied the collected results to design a Google Chrome extension that applies the trained model in real-time. It provides a quality value, readability score, and relevant features that may need improvement, being helpful to any Wikipedia user.

Overall, we have built a reliable application based on an effective machine learning model that matches or beats most state-of-the-art approaches. Furthermore, it is a unique solution that aims to assist both Wikipedia readers and editors on a regular basis.

7.1 Future Work

We consider this study has potential for future research on a few paths. The set of features could be further tuned by assessing their importance individually instead of grouping them in categories. For instance, some style features are probably much more relevant than others, so upon further experiment, that category could see a severe reduction without significantly affecting model performance. Authors may also wish to experiment with additional features, such as the ones we excluded from the literature review, to determine if they improve the results.

7.2 Limitations

Although this work has various benefits, it also comes with several limitations. First, the wikitext cleaner we designed is not flawless. We created one from scratch because we found no existing solutions that could specifically perform what the project required. However, it is still probable

we overlooked various wikitext keywords and templates. This issue is exacerbated in non-English articles, where some keywords are translated to the respective language, a factor that we did not consider. Regardless, it is unlikely that this scenario significantly impacts our English model's effectiveness, as it relates to only a few edge cases. For the remaining languages, there may be a slight impact on the performance, depending on the importance of features related to those keywords (e.g., image count), but the other features should help mitigate that issue.

There is also a limitation concerning the subject of this investigation. To sum up, article features are not enough to measure quality. Downscaling entire articles into a few numbers guarantees the loss of critical information, which is why studies like ours never achieve accuracy values much higher than 90%, even with two classes. The article's context, the text's integrity, and the dimension of its theme are crucial to determining the quality and are virtually impossible to summarize for an approach like ours. Not only that, but Wikipedia's quality scale is subject to human error in numerous stages. For starters, reviewers assess articles within their respective WikiProjects [96]. Editors of different WikiProjects may unintentionally apply their distinct biases in their evaluations, effectively leading to varying standards for the same level of quality. This observation indicates that quality assessment may be inconsistent even within the English Wikipedia. Additionally, we assume that the evaluations are up-to-date, which can never be a guarantee.

Regardless, quality is a subjective property, even if there are some indicators we can associate with it. If we cannot say that Wikipedia has a uniformized scale, it is not unreasonable to say that our solution attempts to create one.

Appendix A

Literature Review Results

This appendix details the publications we reviewed during the analysis of the state of the art ¹

A.1 Screening Phase - Abstract Scanning

Table A.1 lists the publications that were assessed in the of the Literature Review process, where we assess the paper abstracts. We include the assigned IDs, the paper URL, and whether they were included in the following phase.

Table A.1: Literature Review: Publications assessed in the Screening Phase

ID	Title (URL)	Included
1	An end-to-end learning solution for assessing the quality of Wikipedia articles (https://doi.org/10.1145/3125433.3125448)	Yes
2	Automatic quality assessment of content created collaboratively by web communities: A case study of wikipedia (https://doi.org/10.1145/1555400.1555449)	Yes
3	Seeking Health Information Online: Does Wikipedia Matter? (https://doi.org/10.1197/jamia.M3059)	No
4	WikipediaViz: Conveying article quality for casual Wikipedia readers (http://dx.doi.org/10.1109/PACIFICVIS.2010.5429611)	Yes
5	Assessing the quality of Thai Wikipedia articles using concept and statistical features (http://dx.doi.org/10.1007/978-3-319-05951-8_49)	Yes

¹This information is also available in the data repository we provide at <https://drive.google.com/drive/folders/1t1ojqrEV98zmHWY1WCgmwuaQwBVcWca?usp=sharing>

Table A.1: Literature Review: Publications assessed in the Screening Phase

ID	Title (URL)	Included
6	A psycho-lexical approach to the assessment of information quality on wikipedia (http://dx.doi.org/10.1109/WI-IAT.2015.23)	Yes
7	Measuring article quality in Wikipedia using the collaboration network (https://doi.org/10.1145/2808797.2808895)	Yes
8	Accuracy and readability of cardiovascular entries on Wikipedia: Are they reliable learning resources for medical students? (http://dx.doi.org/10.1136/bmjopen-2015-008187)	No
9	Measuring quality of collaboratively edited documents: The case of Wikipedia (https://doi.org/10.1109/CIC.2016.044)	Yes
10	Relative quality and popularity evaluation of multilingual Wikipedia articles (http://dx.doi.org/10.20944/preprints201709.0130.v1)	No
11	Readability and quality of Wikipedia pages on neurosurgical topics (http://dx.doi.org/10.1016/j.clineuro.2018.01.021)	No
12	Automatically assessing the quality of Wikipedia contents (https://doi.org/10.1145/3297280.3297357)	Yes
13	A deep learning-based quality assessment model of collaboratively edited documents: A case study of Wikipedia (http://dx.doi.org/10.1177/0165551519877646)	Yes
14	Quality assessment of peer-produced content in knowledge repositories using big data and social networks: The case of implicit collaboration in Wikipedia (https://doi.org/10.1145/3371041.3371045)	Yes
15	Assessing the quality of information on Wikipedia: A deep-learning approach (http://dx.doi.org/10.1002/asi.24210)	Yes
16	Assessing the quality of health-related Wikipedia articles with generic and specific metrics (https://doi.org/10.1145/3442442.3452355)	Yes
17	Measuring Wikipedia Article Quality in One Dimension by Extending ORES with Ordinal Regression (https://doi.org/10.1145/3479986.3479991)	Yes
18	Automating assessment of collaborative writing quality in multiple stages: The case of wiki (https://doi.org/10.1145/2883851.2883963)	Yes

Table A.1: Literature Review: Publications assessed in the Screening Phase

ID	Title (URL)	Included
19	Quality assessment of Wikipedia articles without feature engineering (https://doi.org/10.1145/2910896.2910917)	Yes
20	"A ""quick and dirty"" website data quality indicator" (https://doi.org/10.1145/1458527.1458538)	No
21	Measuring the quality of web content using factual information (https://doi.org/10.1145/2184305.2184308)	Yes
22	Web content classification using distributions of subjective quality evaluations (https://doi.org/10.1145/2994132)	No
23	On measuring the lexical quality of the web (https://doi.org/10.1145/2184305.2184307)	No
24	Interactive quality analytics of user-generated content: An integrated toolkit for the case of Wikipedia (https://doi.org/10.1145/3150973)	No
25	Measuring the quality of edits to Wikipedia (https://doi.org/10.1145/2641580.2641621)	Yes
26	Measuring article quality in wikipedia: Models and evaluation (https://doi.org/10.1145/1321440.1321476)	Yes
27	Topic quality metrics based on distributed word representations (https://doi.org/10.1145/2911451.2914720)	No
28	Proposal and Comparison of Health Specific Features for the Automatic Assessment of Readability (https://doi.org/10.1145/3397271.3401187)	Yes
29	Exploring wiki: Measuring the quality of social media using ant colony metaphor (https://doi.org/10.1145/2077489.2077545)	Yes
30	So you know you're getting the best possible information: A tool that increases Wikipedia credibility (https://doi.org/10.1145/1518701.1518929)	No
31	Do you know your IQ? A research agenda for information quality in systems (https://doi.org/10.1145/1710115.1710121)	No
32	Reliability of user-generated data: The case of biographical data in Wikipedia (https://doi.org/10.1145/2641580.2641618)	No
33	Size matters: Word count as a measure of quality on Wikipedia (https://doi.org/10.1145/1367497.1367673)	Yes
34	Measuring Wikipedia (http://eprints.rclis.org/6207/)	No
35	Statistical measure of quality in Wikipedia (https://doi.org/10.1145/1964858.1964876)	No

Table A.1: Literature Review: Publications assessed in the Screening Phase

ID	Title (URL)	Included
36	Cooperation and quality in Wikipedia (https://doi.org/10.1145/1296951.1296968)	Yes
37	Who does what: Collaboration patterns in the Wikipedia and their impact on article quality (https://doi.org/10.1145/1985347.1985352)	No
38	"A jury of your peers: quality, experience and ownership in Wikipedia" (https://doi.org/10.1145/1641309.1641332)	Yes
39	Does it matter who contributes - A study on featured articles in the german Wikipedia (https://doi.org/10.1145/1286240.1286290)	Yes
40	Tell me more: An actionable quality model for wikipedia (https://doi.org/10.1145/2491055.2491063)	No
41	Assessing the quality of Wikipedia articles with lifecycle based metrics (https://doi.org/10.1145/1641309.1641333)	Yes
42	Measuring author contributions to the Wikipedia (https://doi.org/10.1145/1822258.1822279)	Yes
43	On ranking controversies in Wikipedia: Models and evaluation (https://doi.org/10.1145/1341531.1341556)	No
44	Don't bite the newbies: How reverts affect the quantity and quality of Wikipedia work (https://doi.org/10.1145/2038558.2038585)	No
45	Identifying featured articles in Wikipedia: Writing style matters (https://doi.org/10.1145/1772690.1772847)	Yes
46	Determinants of Wikipedia quality: The roles of global and local contribution inequality (https://doi.org/10.1145/1718918.1718963)	No
47	Information quality assessment of community generated content: A user study of Wikipedia (https://doi.org/10.1177/2F0165551511416065)	No
48	Learning to predict the quality of contributions to wikipedia (https://www.aaai.org/Papers/Workshops/2008/WS-08-15/WS08-15-002.pdf)	Yes
49	Trust in wikipedia: How users trust information from an unknown source (https://doi.org/10.1145/1772938.1772944)	Yes
50	Information quality work organization in Wikipedia (https://doi.org/10.1002/asi.20813)	No

Table A.1: Literature Review: Publications assessed in the Screening Phase

ID	Title (URL)	Included
51	Assigning trust to Wikipedia content (https://doi.org/10.1145/1822258.1822293)	Yes
52	On measuring the quality of wikipedia articles (https://doi.org/10.1145/1772938.1772943)	Yes
53	Detecting Wikipedia vandalism using wikitrust (https://www.researchgate.net/publication/221159945_Detecting_wikipedia_vandalism_using_WikiTrust_Lab_report_for_PAN_at_CLEF_2010)	Yes
54	Measuring qualities of articles contributed by online communities (https://doi.org/10.1109/WI.2006.115)	No
55	Automatically classifying edit categories in Wikipedia revisions (https://www.semanticscholar.org/paper/Automatically-Classifying-Edit-Categories-in-Daxenberger-Gurevych/66c587532b5ba17115d3385307c2af761f091630)	No
56	Can you ever trust a Wiki? Impacting perceived trustworthiness in Wikipedia (https://doi.org/10.1145/1460563.1460639)	Yes
57	Estimating the quality of articles in Russian Wikipedia using the logical-linguistic model of fact extraction (http://dx.doi.org/10.1007/978-3-319-59336-4_3)	Yes
58	"Disinformation on the web: Impact, characteristics, and detection of Wikipedia hoaxes" (https://doi.org/10.1145/2872427.2883085)	No
59	Detecting wikipedia vandalism with active learning and statistical language models (https://doi.org/10.1145/1772938.1772942)	No
60	Content and collaboration: An affiliation network approach to information quality in online peer production communities (https://doi.org/10.1287/isre.2016.0622)	No
61	Pagerank on wikipedia: Towards general importance scores for entities (http://dx.doi.org/10.1007/978-3-319-47602-5_41)	No
62	Quality of information sources about mental disorders: a comparison of Wikipedia with centrally controlled web and printed sources (https://doi.org/10.1017/S003329171100287X)	No
63	On the measurability of information quality (https://doi.org/10.1002/asi.21447)	Yes

Table A.1: Literature Review: Publications assessed in the Screening Phase

ID	Title (URL)	Included
64	Information quality in a community-based encyclopedia (https://doi.org/10.1142/9789812701527_0009)	Yes
65	Computational trust in Web content quality: a comparative evaluation on the Wikipedia project (http://informatica.si/index.php/informatica/article/viewFile/137/129)	Yes
66	Quality of patient health information on the Internet: reviewing a complex and evolving landscape (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3920473/)	No
67	Reputation and reliability in collective goods: The case of the online encyclopedia Wikipedia (http://dx.doi.org/10.1177/1043463109336804)	No
68	Identifying controversial articles in Wikipedia: A comparative study (https://doi.org/10.1145/2462932.2462942)	Yes
69	A framework for information quality assessment (https://doi.org/10.1002/asi.20652)	Yes
70	"What's on Wikipedia, and what's not...?: Assessing completeness of information" (https://doi.org/10.1177/2F0894439308321890)	No

A.2 Eligibility Phase - Full Text Assessment

Table A.2 lists the publications that were assessed in the Eligibility Phase of the Literature Review process, where we read the full text of each paper. For each ID, we indicate whether we included the article in the following phase, and a short explanation for the exclusions.

Table A.2: Literature Review: Publications assessed in the Eligibility Phase

ID	Included	Reason
1	Yes	—
2	Yes	—
4	No	Focus on quality visualization instead of automatic assessment.
5	Yes	—
6	Yes	—
7	No	No discussion of ML algorithms, features or metrics.
9	Yes	—
12	Yes	—
13	Yes	—
14	Yes	—

Table A.2: Literature Review: Publications assessed in the Eligibility Phase

ID	Included	Reason
15	Yes	—
16	Yes	—
17	No	Does not discuss article features, focuses on disproving the theory «different levels of quality are “evenly spaced” from one another».
18	No	Does not explain used features and used algorithms.
19	Yes	—
21	No	Manual measurement of quality (number of facts).
25	No	Measures quality of edits, not the articles. Does not propose any metrics or algorithms, and validates it with human raters.
26	No	Uses the metrics to directly output the quality and not for ML.
28	Yes	—
29	Yes	—
33	Yes	—
36	No	Little emphasis is given to predicting Wikipedia quality, more to evaluate correlation between a few criteria (e.g # of edits) and quality.
38	Yes	—
39	Yes	—
41	Yes	—
42	No	Measures quality of the contributions, not the articles.
45	Yes	—
48	Yes	—
49	No	Uses volunteers to manually assess trustworthiness of articles
51	No	Focus of the paper is on computing trust of edits, not articles.
52	Yes	—
53	No	The study proposes metrics evaluating for user edits, not articles.
56	No	Unrelated to our research goals, since its focused visualization of trustworthiness, not measurement.
57	No	Based on fact extraction and less on quality metrics.
63	No	Manual study of existing Wikipedia articles.
64	Yes	—
65	No	Discusses subjective trustworthiness ideals, not quality measures.
68	No	The goal is to predict article controversy (e.g revert wars).
69	No	Does not propose any feature metric, and is very similar to study 64.

A.3 Inclusion Phase - Relevance Scoring

Table A.3 lists the relevance scores for all of the included papers. The four questions and their score ranges can be summarized as:

- Q1. Discussion of Features ([0 – 3])
- Q2. Discussion of ML approaches ([0 – 3])
- Q3. Discussion of Results ([0 – 3])
- Q4. Article Language other than English ([0 – 1])

Table A.3: Literature Review: Publications assessed in the Inclusion Phase

ID	Q1	Q2	Q3	Q4	Total
1	1	2	2	1	6
2	3	1	3	1	8
5	2	2	2	1	7
6	2	1	3	0	6
9	3	3	3	0	9
12	3	3	3	0	9
13	3	2	3	0	8
14	3	2	2	0	7
15	3	2	3	0	8
16	3	0	2	0	5
19	1	1	2	0	4
28	2	0	2	0	4
29	0	2	1	0	3
33	1	2	2	0	5
38	1	0	1	0	2
39	1	0	1	1	3
41	2	0	2	0	4
45	1	1	2	0	4
48	2	0	1	0	3
52	2	1	2	0	5
64	3	1	2	0	6
69	2	0	0	0	2

Appendix B

MediaWiki Action API

This appendix provides secondary details regarding MediaWiki's Action API.

B.1 MediaWiki Action API Endpoints

Table [B.1](#) displays the language codes of all Wikipedia versions. The base URL for each one will always have the following format: `https://<lang>.wikipedia.org/w/api.php`, where `<lang>` is the language code for the respective Wikipedia.

Table B.1: Wikipedia version list and respective URLs (Data Extracted from Wikipedia [91])

Language	Code	Language	Code
English	en	Belarusian	be
Cebuano	ceb	Croatian	hr
German	de	Greek	el
Swedish	sv	Simple English	simple
French	fr	Lithuanian	lt
Dutch	nl	Azerbaijani	az
Russian	ru	Galician	gl
Spanish	es	Slovene	sl
Italian	it	Urdu	ur
Egyptian Arabic	arz	Norwegian (Nynorsk)	nn
Polish	pl	Georgian	ka
Japanese	ja	Hindi	hi
Vietnamese	vi	Thai	th
Mandarin	zh	Tamil	ta
Waray	war	Uzbek	uz
Arabic	ar	Latin	la
Ukrainian	uk	Welsh	cy
Portuguese	pt	Asturian	ast
Persian	fa	Macedonian	mk
Catalan	ca	Volapük	vo
Serbian	sr	Cantonese	zh-yue
Indonesian	id	Bengali	bn
Korean	ko	Latvian	lv
Norwegian (Bokmål)	no	Tajik	tg
Finnish	fi	Burmese	my
Hungarian	hu	Afrikaans	af
Czech	cs	Malagasy	mg
Turkish	tr	Bosnian	bs
Chechen	ce	Occitan	oc
Serbo-Croatian	sh	Albanian	sq
Southern Min	zh-min-nan	Marathi	mr
Romanian	ro	Low German	nds
Tatar	tt	Kyrgyz	ky
Basque	eu	Malayalam	ml
Malay	ms	Belarusian (Tarašk)	be-tarask
Esperanto	eo	Telugu	te
Hebrew	he	Newar	new
Armenian	hy	Breton	br
Bulgarian	bg	Swahili	sw
Danish	da	Venetian	vec
South Azerbaijani	azb	Javanese	jv
Slovak	sk	Piedmontese	pms
Kazakh	kk	Western Punjabi	pnb
Estonian	et	Haitian Creole	ht
Minangkabau	min	Sundanese	su

Table B.1: Wikipedia version list and respective URLs (Data Extracted from Wikipedia [91])

Language	Code	Language	Code
Luxembourgish	lb	Amharic	am
Bashkir	ba	Neapolitan	nap
Irish	ga	Sakha	sah
Silesian	szl	Pashto	ps
Icelandic	is	Upper Sorbian	hsb
Kurdish (Kurmanji)	ku	Banyumasan	map-bms
Lombard	lmo	Faroese	fo
Chuvash	cv	Maithili	mai
West Frisian	fy	Limburgish	li
Tagalog	tl	Mazanderani	mzn
Wu	wuu	Balinese	ban
Aragonese	an	Emilian-Romagnol	eml
Kurdish (Sorani)	ckb	Gorontalo	gor
Scots	sco	Acehnese	ace
Zazaki	diq	Central Bikol	bcl
Punjabi	pa	Sanskrit	sa
Yoruba	yo	Walloon	wa
Nepali	ne	Classical Chinese	zh-classical
Bavarian	bar	Ligurian	lij
Ido	io	Shan	shn
Gujarati	gu	Zulu	zu
Alemannic	als	Hill Mari	mrj
Kannada	kn	Meadow Mari	mhr
Sicilian	scn	Assamese	as
Bishnupriya	bpy	Fiji Hindi	hif
Interlingua	ia	Meitei	mni
Quechua	qu	Western Armenian	hyw
Kotava	avk	Hakka	hak
Mongolian	mn	Tarantino	roa-tara
Navajo	nv	Khmer	km
Mingrelian	xmf	Kapampangan	pam
Crimean Tatar	crh	Interlingue	ie
Sinhalese	si	Northern Sotho	nso
Ladin	lld	Rusyn	rue
Samogitian	bat-smg	Somali	so
Odia	or	Shona	sn
Scottish Gaelic	gd	Bhojpuri	bh
North Frisian	frf	Northern Sami	se
Eastern Min	cdo	West Flemish	vls
Ossetian	os	Dutch Low Saxon	nds-nl
Yiddish	yi	Erzya	myv
Ilokano	ilo	Māori	mi
Sindhi	sd	Nahuatl	nah
Buginese	bug	Sardinian	sc
Hausa	ha	Santali	sat

Table B.1: Wikipedia version list and respective URLs (Data Extracted from Wikipedia [91])

Language	Code	Language	Code
Veps	vep	Kinyarwanda	rw
Gan	gan	Dhivehi	dv
Kabyle	kab	Ripuarian	ksh
Gilaki	glk	Gagauz	gag
Turkmen	tk	Buryat	bxr
Võro	fiu-vro	Avar	av
Corsican	co	Palatinate German	pfl
Tibetan	bo	Pangasinan	pag
Abkhazian	ab	Pali	pi
Franco-Provençal	frp	Hawaiian	haw
Komi	kv	Awadhi	awa
Kashubian	csb	Tayal	tay
Picard	pcd	Papiamento	pap
Cornish	kw	Karachay-Balkar	krc
Manx	gv	Kalmyk	xal
Uyghur	ug	Sakizaya	szy
Moroccan Arabic	ary	Ingush	inh
Udmurt	udm	Zhuang	za
Aymara	ay	Karakalpak	kaa
Saraiki	skr	Pennsylvania German	pdn
Norman	nrm	Atikamekw	atj
Zeelandic	zea	Twi	tw
Guarani	gn	Tongan	to
Maltese	mt	Tumbuka	tum
Banjar	bjn	Aramaic	arc
Lezgian	lez	Dagbani	dag
Inari Sami	smn	Tok Pisin	tpi
Lingua Franca Nova	lfn	Kabiye	kbp
Saterland Frisian	stq	Jamaican Patois	jam
Lao	lo	Nauruan	na
Mirandese	mwl	Moksha	mdf
Igbo	ig	Wolof	wo
Livvi-Karelian	olo	Kabardian	kbd
Romansh	rm	Tulu	tcy
Friulian	fur	Novial	nov
Ladino	lad	Nias	nia
Anglo-Saxon	ang	Kikuyu	ki
Konkani	gom	Tetum	tet
Komi-Permyak	koi	Luganda	lg
Extremaduran	ext	Bislama	bi
Tuvan	tyv	Lojban	jbo
Lower Sorbian	dsb	Aromanian	roa-rup
Doteli	dti	Fijian	fj
Lingala	ln	Kongo	kg
Chavacano	cbk-zam	Lak	lbe

Table B.1: Wikipedia version list and respective URLs (Data Extracted from Wikipedia [91])

Language	Code	Language	Code
Xhosa	xh	Kirundi	rn
Tahitian	ty	Inuktitut	iu
N’Ko	nqo	Akan	ak
Mon	mnw	Swazi	ss
Kashmiri	ks	Chamorro	ch
Old Church Slavonic	cu	Pontic Greek	pnt
Shilha	shi	Adyghe	ady
Seediq	trv	Inupiaq	ik
Sranan Tongo	srn	Fula	ff
Oromo	om	Ewe	ee
Samoan	sm	Dinka	din
Guianan Creole	gcr	Sango	sg
Altai	alt	Greenlandic	kl
Cherokee	chr	Paiwan	pwn
Latgalian	ltg	Tigrinya	ti
Norfolk	pih	Dzongkha	dz
Chewa	ny	Cree	cr
Gothic	got	Ndonga	ng
Madurese	mad	Choctaw	cho
Sotho	st	Kuanyama	kj
Amis	ami	Marshallese	mh
Romani	rmy	Hiri Motu	ho
Tswana	tn	Nuosu	ii
Bambara	bm	Northern Luri	lrc
Tsonga	ts	Afar	aa
Gungbe	guw	Muscogee	mus
Cheyenne	chy	Herero	hz
Venda	ve	Kanuri	kr

Appendix C

Feature Set Description

This appendix lists the contents of our model’s feature set, describing each feature along with its assigned identifier. Every identifier starts with the letter respective to its category. We show the features in Tables C.1 to C.5, each detailing a different category.

Table C.1: Dataset definition: List of used Content Features. *Length* attributes refer to text length in number of characters.

ID	Description
CC	Character Count
CW	Word Count
CSN	Sentence Count
CS	Section Count
CMSL	Mean Section Length
CP	Paragraph Count
CMPL	Mean Paragraph Length
CLSL	Largest Section Length
CSSL	Shortest Section Length
CSTDSL	Standard Deviation of Section Length
CLSSR	Longest-Shortest Section Ratio
CB	Subsection Count
CBPS	Subsection Count per Section
CNL	Introduction Length (Characters)
CNLTLR	Introduction Length-Text Length Ratio
CR	Reference Count
CCC	Citation Count
CCCPC	Citation Count per Character
CCCPS	Citation Count per Section
CEL	External Link Count
CELPS	External Link Count per Section
CELPC	External Link Count per Character
CIL	Internal Link Count
CILPC	Internal Link Count per Character
CI	Image Count
CIPC	Images per Character
CIPS	Images per Section
CH3	Number of L3 Headings (Sub-subsections)
CIB	Number of Information Boxes
CCT	Number of Citation Templates
CNCT	Number of Non-Citation Templates
CUCT	Number of Unique Templates
CTB	Number of Tables

Table C.2: Dataset definition: List of used Style Features

ID	Description
SSLS	Largest Sentence Size (in words)
SSSS	Shortest Sentence Size (in words)
SMSS	Mean Sentence Size (in words)
SLSR	Large Sentence Rate (% sentences whose size is 10 words greater than SMSS)
SSSR	Short Sentence Rate (% sentences whose size is 5 words shorter than SMSS)
SQ	Question Count
SQPS	Question Count per Sentence
SE	Exclamation Count
SEPS	Exclamation Count per Sentence
SS	Syllable Count
SMSPW	Mean # of Syllables per Word
SMCPW	Mean # of Characters per Word
SSTP	# of sentences starting with a pronoun
SSTART	# of sentences starting with an article
SSTC	# of sentences starting with a coordinate conjunction
SSTSUB	# of sentences starting with a subordinate preposition or conjunction
SSTI	# of sentences starting with an interrogative pronoun
SSTD	# of sentences starting with a determiner
SSTADJ	# of sentences starting with an adjective
SSTN	# of sentences starting with a noun
SSTADV	# of sentences starting with an adverb
SSTPPS	SSTP per Sentence
SSTARTPS	SSTART per Sentence
SSTCPS	SSTC per Sentence
SSTSUBPS	SSTSUB per Sentence
SSTDPS	SSTD per Sentence
SSTADJPS	SSTADJ per Sentence
SSTNPS	SSTN per Sentence
SSTADVPS	SSTADV per Sentence
SMV	Modal Verb Count
STB	“To be” Verb Count
SUW	Unique Word Count
SN	Noun Count
SUN	Unique Noun Count
SV	Verb Count
SUV	Unique Verb Count
SP	Pronoun Count

Table C.2: Dataset definition: List of used Style Features

ID	Description
SUP	Unique Pronoun Count
SADJ	Adjective Count
SUADJ	Unique Adjective Count
SADV	Adverb Count
SUADV	Unique Adverb Count
SCC	Coordinating Conjunction Count
SUCC	Unique Coordinating Conjunction Count
SSUB	Subordinating Preposition or Conjunction Count
SUSUB	Unique Subordinating Preposition or Conjunction Count
STBPV	“To be” Verb Count per Verb
SWL6	Number of Words Larger than 6 Letters
SMVPW	SMV per Word
STBPW	STB per Word
SUWPW	SUW per Word
SNPW	SN per Word
SUNPW	SUN per Word
SVPW	SV per Word
SUVPW	SUV per Word
SPPW	SP per Word
SUPPW	SUP per Word
SADJPW	SADJ per Word
SUADJPW	SUADJ per Word
SADPW	SAD per Word
SUADPW	SUAD per Word
SCCPW	SCC per Word
SUCCPW	SUCC per Word
SSUBPW	SSUB per Word
SUSUBPW	SUSUB per Word
SUNPUW	SUN per Unique Word
SUVPW	SUV per Unique Word
SUPPUW	SUP per Unique Word
SUADJPW	SUADJ per Unique Word
SUADVPW	SUADV per Unique Word
SUCCPW	SUCC per Unique Word
SUSUBPW	SUSUB per Unique Word

Table C.3: Dataset definition: List of used Readability Features

ID	Description
RARI	Automated Readability Index
RCL	Coleman-Liau
RFE	Flesh reading ease
RFK	Flesh-Kincaid
RGFI	Gunning Fog Index
RLBI	Lasbarhets Index
RSG	Smog-Grading
RDWS	Difficult Word Score
RDC	Dale-Chall
RLWF	Linsear Write Formula

Table C.4: Dataset definition: List of used History Features

ID	Description
HA	Age (days)
HAPR	Age per Revision
HRPD	Revisions per Day
HRPC	Revisions per Contributor
HRPCSD	Revisions per Contributor Standard Deviation
HR	Revision Count
HC	Contributor Count
HRC	Registered Contributor Count
HAC	Anonymous Contributor Count
HRPCPC	Registered Contributor Count per Contributor
HACPC	Anonymous Contributor Count per Contributor
HRCPAC	Registered Contributor per Anonymous Contributor
HREV	Revert Count
HREVPR	Revert Count per Revision
HRML	Recent Revision Diff (Difference between text size now and 3 months before)
HREC	Recent Revision (3 months) Count
HRECPR	Recent Revision (3 months) Count per Revision
HACT	Active Revision Count
HACTPR	Active Revision Count per Revision
HOCCPR	Occasional Revision Count per Revision
HSSLE	Seconds since Last Edit
HMCL	Mean Comment Length (characters)

Table C.5: Dataset definition: List of used Network Features

ID	Description
NIN	In-degree
NOUT	Out-degree
NASSinin	Assortativity In-In
NASSinout	Assortativity In-Out
NASSoutin	Assortativity Out-In
NASSoutout	Assortativity Out-Out
NREP	Reciprocity within Neighbors
NT	# of Versions in other Languages

Appendix D

Machine Learning Results

This appendix lists all the results we collected during our model experimentation.

D.1 Dataset Analysis

This section shows average values for some features we assessed during the analysis of our dataset. Table D.2 lists the average values for several readability formulas we used in the dataset. For a more reliable comparison, we only considered formulas that aim to evaluate the grade level of a text. We also detail the mean readability of each quality level. Table D.1 displays information relative to features from other categories.

Table D.1: Average feature values for each quality level

Feature	Avg.	FA	GA	B	C	Start	Stub
CC (# Characters)	12724.4	28730.4	16659.5	15936.1	9816.9	4507.8	680.0
CW (# Words)	2051.34	4647.72	2709.95	2556.40	1569.62	714.88	106.89
CSN (# Sentences)	100.59	223.56	132.19	125.68	78.06	36.92	6.98
CNL (Intro Length)	1033.2	2019.65	1443.13	1003.06	741.07	610.51	380.71
SV (# Verbs)	283.71	671.97	383.89	342.56	205.33	86.6	11.53
CCC (# Citations)	50.04	111.71	79.74	58.77	33.47	13.71	2.8
HA (Age)	4541.3	5622.6	4790.8	4855.9	4186.8	4140.4	3650.4
HR (# Revisions)	557.16	1462.03	714.71	658.77	337.75	140.66	28.2
NIN (In-degree)	157.25	436.5	194.96	180.71	82.88	41.9	6.31

Table D.2: Average readability values for each quality level

Readability Formula	Avg.	FA	GA	B	C	Start	Stub
Auto Readability Index	14.27	14.14	13.5	14.86	14.46	15.55	13.09
Coleman-Liau	12.56	12.38	12.02	12.53	12.68	12.74	13.03
Flesch Kincaid	13.93	13.81	13.27	14.41	14.06	14.73	13.27
Gunning Fog Index	17.48	17.40	16.73	18.02	17.65	18.26	16.79
Smog Grading	15.19	15.40	14.91	15.56	15.41	15.37	14.50
Linsear Write Formula	16.55	16.66	15.93	17.61	16.79	18.10	14.21

D.2 Algorithm Performance

Table D.3 compares the performance of different classification algorithms, using all the features and classes (CSRHN6). Table D.4 performs the same analysis for regression methods.

Table D.3: Performance metrics of classification algorithms (CSRHN6)

Algorithm	Accuracy	Precision	Recall	F1-Score
ada_c	0.61	0.62	0.61	0.62
forest_c	0.62	0.62	0.62	0.62
gboost_c	0.60	0.59	0.60	0.59
gnb_c	0.45	0.44	0.45	0.42
knn_c	0.55	0.55	0.55	0.54
logreg_c	0.62	0.61	0.62	0.62
mlp_c	0.62	0.62	0.62	0.62
svc_c	0.64	0.63	0.64	0.64
tree_c	0.52	0.52	0.52	0.52

Table D.4: Performance metrics of regression algorithms (CSRHN6)

Algorithm	MSE	RMSE	MAE	R ² Score
ada_r	0.0163	0.1278	0.0981	0.8515
forest_r	0.0159	0.1262	0.0898	0.8553
gboost_r	0.0254	0.1594	0.1337	0.7691
linreg_r	0.0216	0.1471	0.1134	0.8033
mlp_r	0.0179	0.1337	0.0994	0.8375
svr_r	0.0185	0.1361	0.1033	0.8315
tree_r	0.0297	0.1724	0.1090	0.7299

D.3 Performance per Number of Classes

Table D.5 shows the accuracy of classification models trained with different numbers of classes. Table D.6 displays the equivalent results for regressors. Refer to Tables 4.16 and 4.15 for information about each experiment.

D.4 Feature Subset Performance

This section details the results of our experiments with various subsets of feature categories. We kept all of the six classes throughout all of these tests. Table D.7 shows accuracy values for the classifiers, while Table D.8 displays the Mean Absolute Errors of each regressor. Column headers will use the naming convention we previously defined (i.e., each letter corresponds to a category's initial).

Table D.5: Accuracy of classifiers trained with various numbers of classes (all features)

Algorithm	6 classes	5 classes	3 classes	2 classes
ada_c	0.61	0.70	0.81	0.91
forest_c	0.62	0.72	0.81	0.91
gboost_c	0.60	0.68	0.78	0.88
gnb_c	0.45	0.46	0.67	0.81
knn_c	0.55	0.63	0.75	0.88
logreg_c	0.62	0.71	0.80	0.90
mlp_c	0.62	0.70	0.80	0.91
svc_c	0.64	0.73	0.81	0.91
tree_c	0.52	0.62	0.73	0.86

Table D.6: MAE of regressors trained with various numbers of classes (all features)

Algorithm	6 classes	5 classes	3 classes	2 classes
ada_r	0.0981	0.1010	0.1578	0.1546
forest_r	0.0898	0.0915	0.1333	0.1513
gboost_r	0.1337	0.1368	0.1950	0.2498
linreg_r	0.1134	0.1179	0.1734	0.2359
mlp_r	0.0994	0.1003	0.1444	0.1663
svr_r	0.1033	0.1061	0.1534	0.1804
tree_r	0.1090	0.1060	0.1431	0.1484

Table D.7: Accuracy of classifiers trained with various feature subsets (all classes)

Algorithm	CSRHN	CSRH	CSRN	CSHN	CRHN	SRHN
ada_c	0.61	0.60	0.58	0.61	0.61	0.57
forest_c	0.62	0.62	0.60	0.62	0.63	0.59
gboost_c	0.60	0.60	0.57	0.60	0.60	0.56
gnb_c	0.45	0.46	0.46	0.44	0.41	0.45
knn_c	0.55	0.55	0.53	0.55	0.56	0.51
logreg_c	0.62	0.62	0.58	0.62	0.60	0.59
mlp_c	0.62	0.61	0.58	0.61	0.63	0.59
svc_c	0.64	0.64	0.60	0.64	0.64	0.60
tree_c	0.52	0.52	0.50	0.52	0.52	0.49

Algorithm	C	S	R	H	N	CSR	CSH	CRH	CH
ada_c	0.56	0.52	0.47	0.48	0.41	0.58	0.60	0.61	0.60
forest_c	0.59	0.53	0.49	0.49	0.44	0.60	0.62	0.63	0.62
gboost_c	0.57	0.52	0.49	0.47	0.43	0.57	0.60	0.60	0.59
gnb_c	0.42	0.46	0.30	0.34	0.27	0.47	0.45	0.42	0.41
knn_c	0.55	0.48	0.43	0.44	0.39	0.53	0.55	0.56	0.57
logreg_c	0.55	0.53	0.48	0.47	0.40	0.58	0.62	0.60	0.60
mlp_c	0.60	0.52	0.50	0.50	0.43	0.58	0.61	0.63	0.62
svc_c	0.59	0.54	0.49	0.50	0.42	0.60	0.64	0.63	0.63
tree_c	0.52	0.44	0.43	0.40	0.37	0.49	0.52	0.53	0.52

Table D.8: MAE of regressors trained with various feature subsets (all classes)

Algorithm	CSRHN	CSRH	CSRN	CSHN	CRHN	SRHN
ada_r	0.098	0.098	0.101	0.098	0.099	0.108
forest_r	0.089	0.090	0.094	0.089	0.089	0.101
gboost_r	0.133	0.133	0.135	0.133	0.134	0.141
linreg_r	0.113	0.114	0.119	0.113	0.120	0.120
mlp_r	0.099	0.100	0.105	0.098	0.098	0.109
svr_r	0.103	0.103	0.106	0.103	0.099	0.112
tree_r	0.109	0.108	0.113	0.111	0.107	0.127

Algorithm	C	S	R	H	N	CSR	CSH	CRH	CH
ada_r	0.108	0.126	0.138	0.158	0.165	0.103	0.099	0.100	0.102
forest_r	0.095	0.117	0.128	0.146	0.152	0.094	0.089	0.090	0.090
gboost_r	0.136	0.149	0.153	0.175	0.178	0.135	0.133	0.134	0.134
linreg_r	0.133	0.129	0.188	0.170	0.228	0.120	0.114	0.121	0.124
mlp_r	0.110	0.119	0.132	0.148	0.165	0.104	0.098	0.097	0.101
svr_r	0.102	0.124	0.137	0.147	0.163	0.106	0.103	0.099	0.099
tree_r	0.112	0.139	0.143	0.171	0.173	0.113	0.110	0.109	0.109

D.5 Training and Prediction Times

This section contains results relating to time measurements of the training and prediction steps.

Table D.9 lists training times for all the algorithms we studied in this project, varying the used classes and feature categories. At the same time, Table D.10 shows the prediction times for the same algorithms, feature subsets, and classes. Column headers will have the format <CATEGORIES><CLASSES>, similarly to our previously defined naming convention, with some differences:

- If <CATEGORIES> is displayed as *[all]*, the model trained with all the available features.
- If <CLASSES> is omitted, six (all) classes were considered

Table D.11 lists the measured times of every task of feature computation, separated by article quality. We also show the average percentage of the time spent on each task among the six classes. Since these are averages of ratios, the total value for that column has no significant meaning.

Table D.9: Algorithm training times across different feature sets and number of classes, in seconds

Algorithm	[all]	[all]5	[all]3	[all]2	CSRH	CSRN	CSHN	CRHN	SRHN
ada_c	374.1	367.1	359.8	368.9	358.8	319.1	345.5	205.0	312.5
forest_c	57.09	52.79	41.05	31.37	53.04	52.90	51.51	41.87	54.25
gboost_c	656.2	542.4	327.5	110.9	593.1	533.9	588.0	346.7	522.5
gnb_c	1.190	1.220	1.130	1.140	1.120	1.180	1.120	1.110	1.070
knn_c	7.590	7.400	7.410	7.320	9.130	7.680	7.510	7.120	7.650
logreg_c	38.01	27.56	12.87	5.060	31.73	31.50	32.43	15.74	28.92
mlp_c	64.87	101.0	59.17	53.15	78.56	99.38	81.84	69.41	93.16
svc_c	103.4	95.56	77.86	45.93	106.8	105.3	103.4	73.49	94.73
tree_c	7.680	7.220	5.460	4.170	7.010	6.500	6.410	4.500	6.200
ada_r	312.3	322.1	318.0	322.6	295.7	266.5	288.9	159.5	252.9
forest_r	307.4	318.1	307.5	254.5	257.8	229.5	250.8	140.6	217.3
gboost_r	97.35	97.76	98.77	100.5	90.96	88.17	93.74	52.55	79.65
linreg_r	1.890	1.450	1.270	1.290	1.170	1.420	1.420	1.020	1.200
mlp_r	26.98	31.75	23.96	27.40	22.70	23.67	17.62	18.04	18.29
svr_r	381.0	430.4	373.9	307.0	306.2	281.6	299.1	95.76	304.6
tree_r	5.490	5.520	5.470	3.940	4.050	3.770	3.680	2.780	3.240

Algorithm	C	S	R	H	N	CSR	CSH	CRH	CH
ada_c	81.36	205.2	52.31	72.10	31.84	306.5	328.9	193.0	149.6
forest_c	22.00	37.84	26.62	23.10	13.57	47.41	50.70	41.03	33.15
gboost_c	128.9	298.1	74.33	106.7	48.27	481.8	526.2	295.7	225.8
gnb_c	0.930	1.010	0.870	0.970	0.930	1.150	1.170	1.000	1.010
knn_c	6.740	7.370	1.910	6.780	1.890	7.480	7.950	7.990	6.740
logreg_c	10.33	20.95	4.770	8.370	4.420	27.92	32.13	15.65	15.83
mlp_c	36.25	66.23	18.84	27.37	19.44	72.29	63.72	50.89	53.45
svc_c	59.41	78.12	56.33	64.10	59.30	82.99	95.30	65.24	63.48
tree_c	2.150	4.040	1.590	1.910	1.160	5.900	6.150	3.870	3.270
ada_r	59.05	146.6	29.03	42.71	18.39	247.3	253.8	130.7	100.7
forest_r	54.09	127.3	26.57	42.63	17.00	219.3	224.8	123.1	112.3
gboost_r	20.46	46.39	11.18	16.79	7.210	74.66	78.78	44.83	34.38
linreg_r	0.830	0.970	0.740	0.820	0.760	1.100	1.280	1.430	1.670
mlp_r	15.02	16.02	8.080	9.500	9.620	26.29	18.51	21.82	14.60
svr_r	45.77	227.9	43.47	52.27	45.31	289.2	343.4	92.49	75.67
tree_r	1.470	2.480	1.430	1.300	1.040	3.440	3.780	2.700	2.360

Table D.10: Model prediction times across different feature sets and number of classes in milliseconds

Algorithm	[all]	[all]5	[all]3	[all]2	CSRH	CSRN	CSHN	CRHN	SRHN
ada_c	20.54	20.40	19.92	18.66	21.55	21.03	21.70	20.81	20.54
forest_c	14.54	14.06	14.64	15.72	14.58	14.58	14.63	15.27	14.83
gboost_c	0.574	0.490	0.476	0.238	0.426	0.482	0.432	0.626	0.482
gnb_c	0.300	0.242	0.202	0.132	0.288	0.328	0.278	0.258	0.270
knn_c	11.28	8.997	12.44	7.138	11.14	10.59	11.06	6.475	9.684
logreg_c	0.102	0.072	0.110	0.076	0.094	0.088	0.084	0.080	0.090
mlp_c	0.246	0.287	0.252	0.230	0.264	0.276	0.378	0.156	0.288
svc_c	4.738	2.972	3.272	1.018	4.696	4.734	5.088	3.578	4.238
tree_c	0.088	0.116	0.078	0.080	0.076	0.072	0.078	0.074	0.078
ada_r	10.81	9.812	10.93	9.861	11.10	11.15	11.07	10.61	11.23
forest_r	17.88	11.51	13.43	10.95	11.89	11.89	12.68	12.82	11.99
gboost_r	0.172	0.174	0.160	0.170	0.152	0.168	0.166	0.158	0.176
linreg_r	0.076	0.062	0.076	0.064	0.062	0.078	0.098	0.070	0.094
mlp_r	0.220	0.230	0.236	0.216	0.228	0.236	0.216	0.102	0.214
svr_r	2.692	1.984	3.834	2.028	2.546	2.316	2.428	1.642	1.994
tree_r	0.090	0.072	0.088	0.092	0.086	0.098	0.136	0.090	0.094

Algorithm	C	S	R	H	N	CSR	CSH	CRH	CH
ada_c	22.15	24.00	23.12	22.17	22.44	21.12	21.08	23.33	22.46
forest_c	15.01	14.82	14.69	14.79	14.42	15.25	14.23	14.38	14.26
gboost_c	0.550	0.418	0.448	0.512	0.442	0.482	0.485	0.470	0.474
gnb_c	0.318	0.298	0.256	0.282	0.278	0.272	0.346	0.328	0.282
knn_c	3.467	6.428	0.948	2.474	0.756	10.17	11.33	6.006	5.177
logreg_c	0.090	0.080	0.084	0.076	0.080	0.084	0.096	0.084	0.122
mlp_c	0.148	0.174	0.170	0.142	0.170	0.252	0.262	0.158	0.140
svc_c	2.928	3.584	2.682	2.862	2.826	3.692	4.824	3.712	3.212
tree_c	0.078	0.072	0.144	0.082	0.076	0.078	0.078	0.072	0.076
ada_r	10.85	10.48	10.56	11.01	10.73	10.60	10.23	10.21	9.818
forest_r	12.11	11.85	13.16	12.57	12.48	13.27	12.24	13.51	12.69
gboost_r	0.150	0.158	0.164	0.160	0.190	0.162	0.169	0.168	0.169
linreg_r	0.064	0.076	0.070	0.060	0.082	0.062	0.088	0.096	0.172
mlp_r	0.094	0.094	0.090	0.098	0.086	0.320	0.264	0.098	0.088
svr_r	1.626	2.202	1.874	2.012	2.086	2.403	2.414	2.010	1.698
tree_r	0.090	0.138	0.110	0.098	0.134	0.114	0.106	0.102	0.096

Table D.11: Feature computation time measurements per article quality (in seconds) and average percentages of total time spent

Task	FA	GA	B	C	Start	Stub	Avg(% of total)
Fetch Wikitext	0.458	0.407	0.400	0.366	0.323	0.300	22.56%
Clean Wikitext	0.016	0.012	0.012	0.009	0.005	0.002	00.57%
Tokenizer	0.062	0.041	0.040	0.029	0.014	0.002	1.84%
Syllables	0.239	0.144	0.140	0.094	0.042	0.007	6.50%
Revision	2.302	1.210	1.054	0.841	0.411	0.315	55.39%
Content	0.003	0.003	0.003	0.002	0.001	0.000	0.12%
Style	0.289	0.174	0.171	0.119	0.055	0.010	8.04%
Readability	0.140	0.084	0.084	0.054	0.022	0.004	3.77%
History	0.004	0.002	0.002	0.002	0.001	0.000	0.11%
Total	3.513	2.077	1.906	1.516	0.874	0.640	N/A

D.6 Multi-language dataset analysis

This section details all results regarding our analysis of the datasets we constructed from different Wikipedia versions. Tables D.12 and D.13 show relevant values we calculated during this examination, while Figures D.1 and D.2 display that information graphically. We also compare data obtained from the complete dataset with its A+ portion.

Table D.12: Comparison between Average Character Count (CC), Word Count (CW), and CC-CW ratio of different language datasets

Language	Avg(CC)	Avg(CW)	Avg(CC / CW)
English	3,248	496.1	6.619
French	4,370	630.8	7.352
Portuguese	2,634	358.1	8.718
Russian	5,076	615.0	8.928

Table D.13: Comparison between Average Character Count (CC), Word Count (CW), and CC-CW ratio of the A+ articles from different language datasets

Language	Avg(CC)	Avg(CW)	Avg(CC / CW)
English	31,621	5,081	6.224
French	54,964	8,553	6.443
Portuguese	40,021	6,109	6.560
Russian	63,039	8,127	7.730

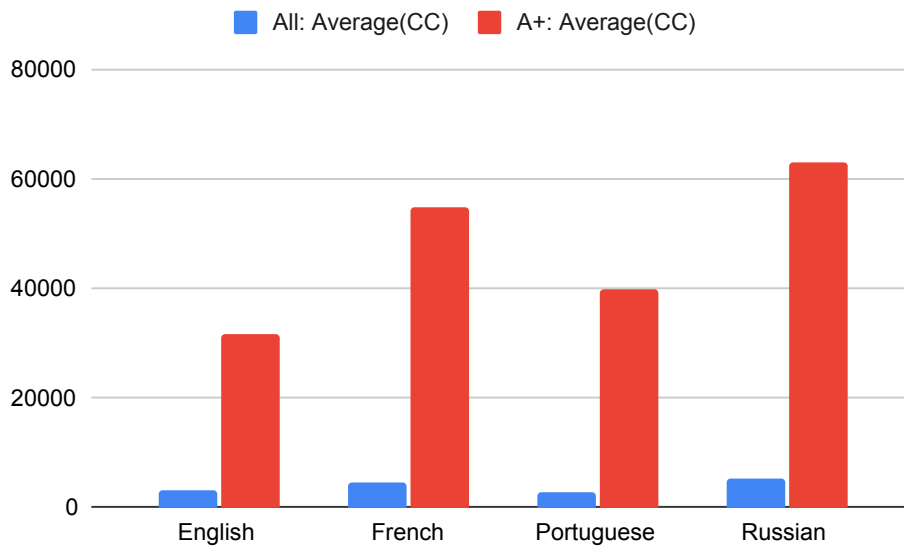


Figure D.1: Comparison between Average Character Count of different language datasets

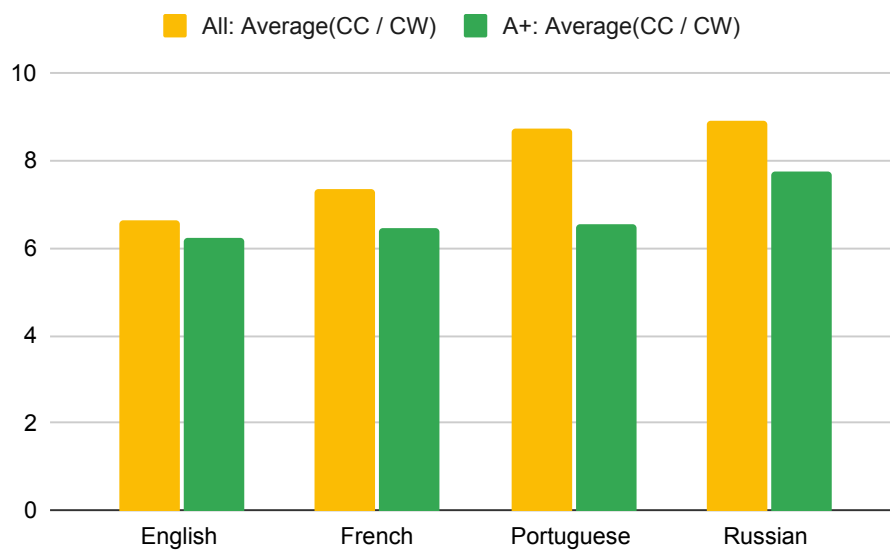


Figure D.2: Comparison between Average Character Count-Word Count ratio of different language datasets

D.7 Multi-language model performance

This section details results concerning the performance of different models when predicting the quality of articles in datasets of different Wikipedia versions. Tables D.14, D.15, and D.16 show the results respective to models *CSRH6/forest_r*, *CRH6/forest_r*, and *CH6/forest_r*.

Table D.14: MSE of model predictions across different dataset languages, with categories CSRH

Language	MSE	RMSE	MAE
English	0.0373	0.1931	0.1318
French	0.0487	0.2207	0.1800
Portuguese	0.0294	0.1714	0.1319
Russian	0.0411	0.2026	0.1812

Table D.15: MSE of model predictions across different dataset languages, with categories CRH

Language	MSE	RMSE	MAE
English	0.0375	0.1936	0.1322
French	0.0596	0.2441	0.1924
Portuguese	0.0316	0.1777	0.1295
Russian	0.0498	0.2231	0.1871

Table D.16: MSE of model predictions across different dataset languages, with categories CH

Language	MSE	RMSE	MAE
English	0.0373	0.1931	0.1317
French	0.0697	0.2640	0.2092
Portuguese	0.0360	0.1898	0.1329
Russian	0.0556	0.2359	0.1918

Appendix E

Chrome Extension Mockups

This appendix shows multiple examples of the chrome extension in use. Figures E.1 to E.6 show all the six different prediction results, from A+ to F.

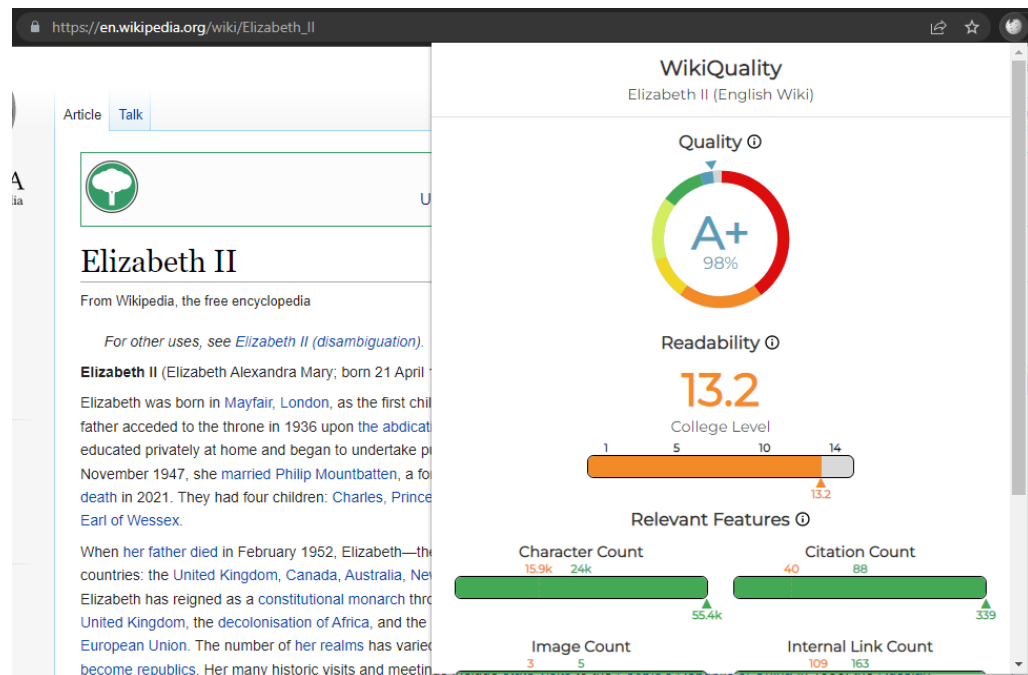


Figure E.1: Chrome extension mockup: A+ prediction

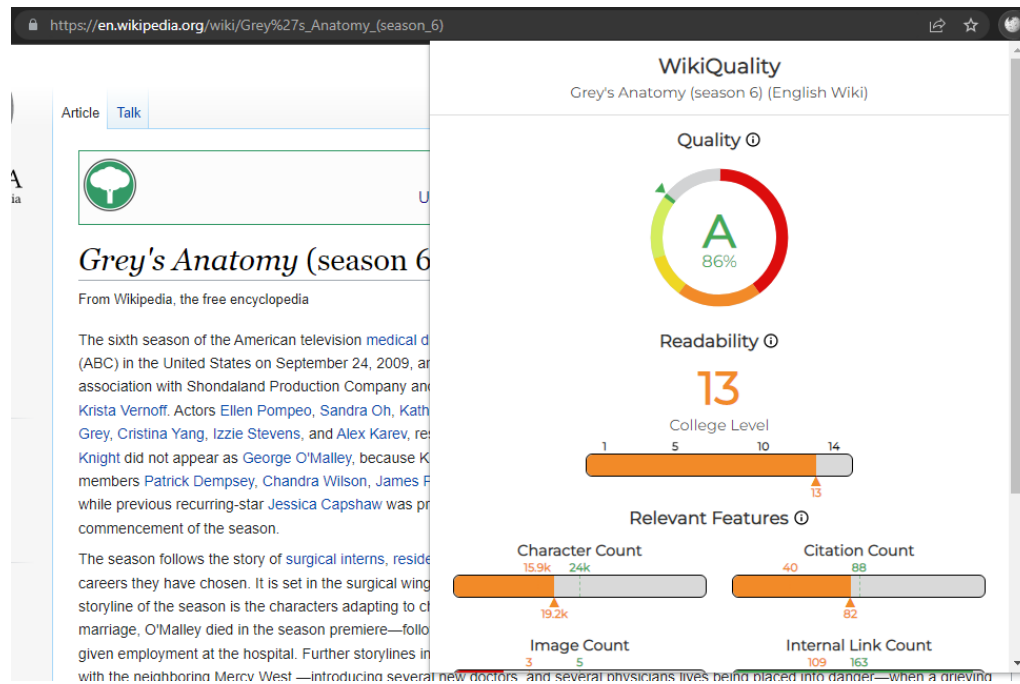


Figure E.2: Chrome extension mockup: A prediction

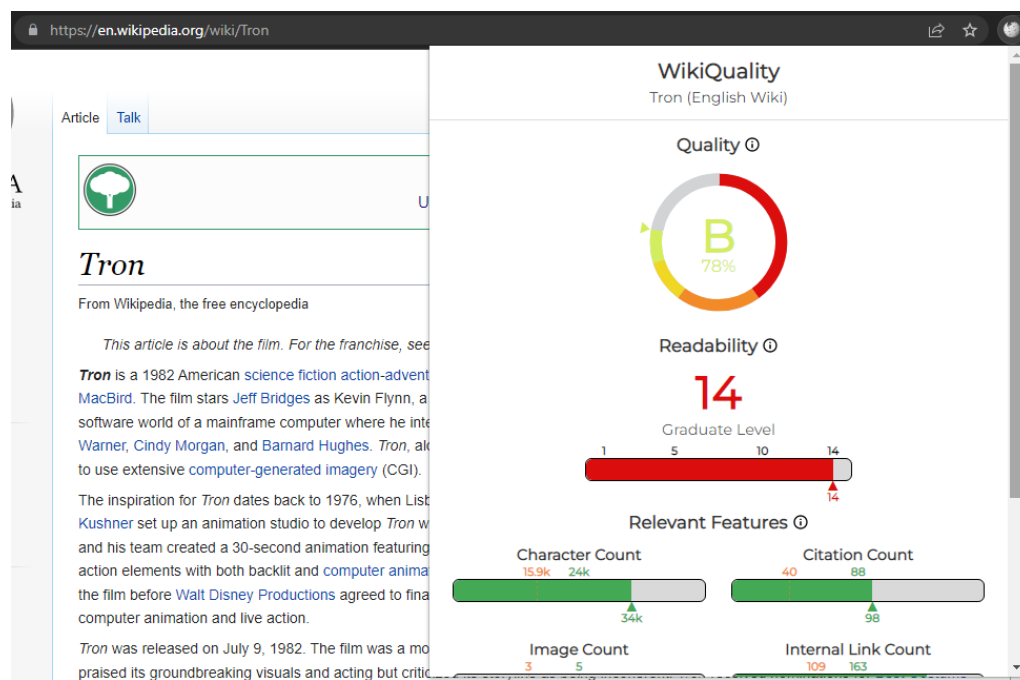


Figure E.3: Chrome extension mockup: B prediction

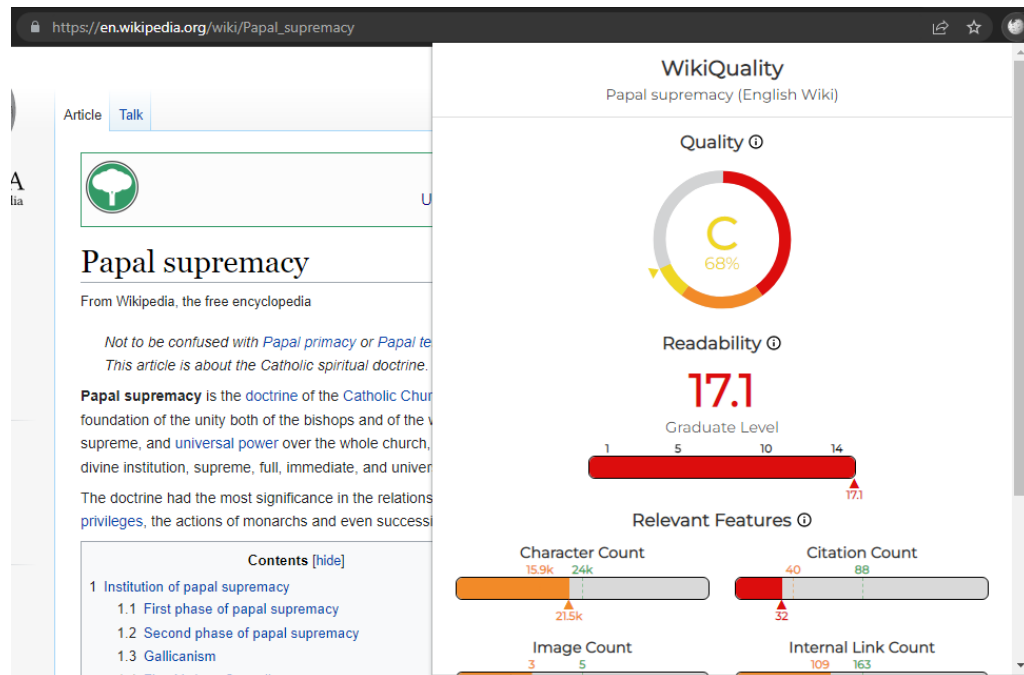


Figure E.4: Chrome extension mockup: C prediction

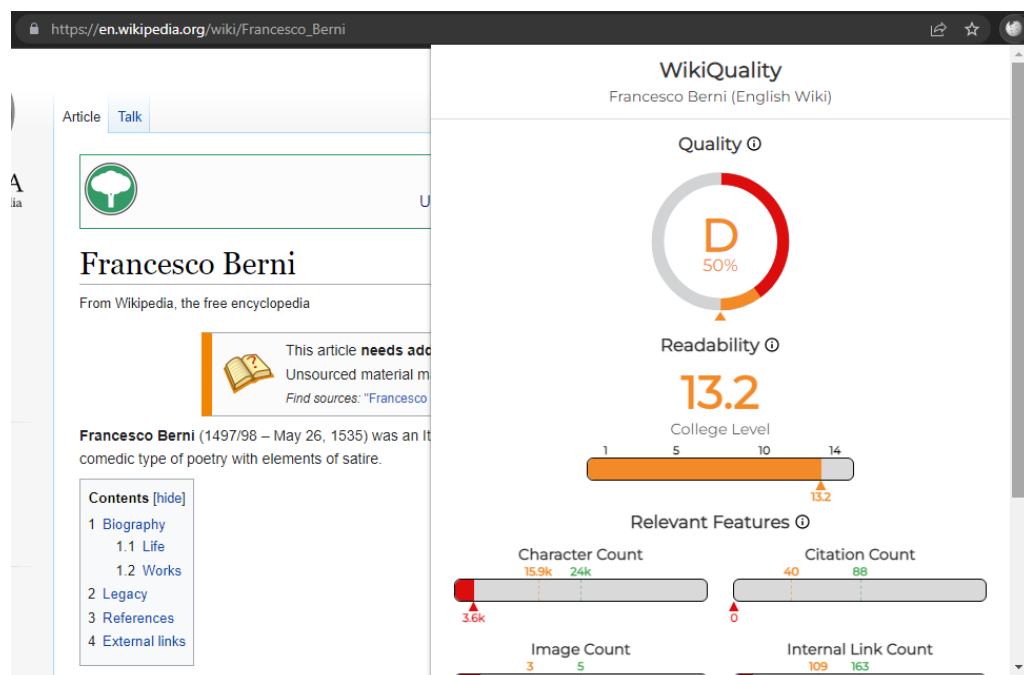


Figure E.5: Chrome extension mockup: D prediction

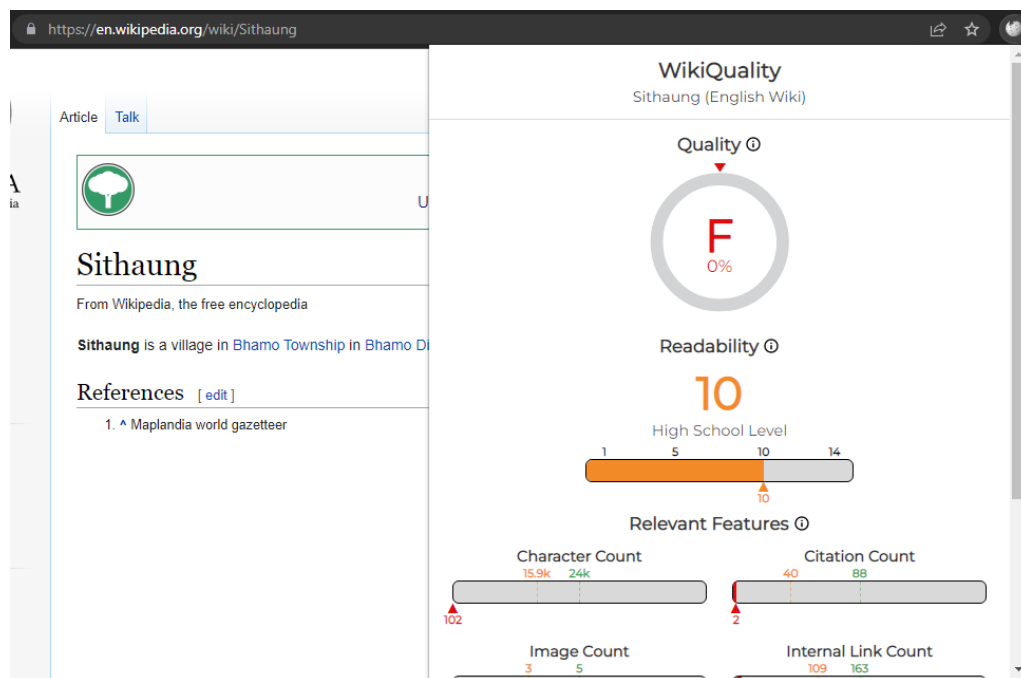


Figure E.6: Chrome extension mockup: F prediction

References

- [1] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4:938, 2018.
- [2] Mohamed Aly. Survey on multiclass classification methods. *Neural Netw*, 19(1-9):2, 2005.
- [3] Hélder Antunes and Carla Teixeira Lopes. Proposal and comparison of health specific features for the automatic assessment of readability. In *SIGIR 2020 - Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1973–1976, Online, China, 7 2020. Association for Computing Machinery, Inc.
- [4] M. Balamurugan and S. Kannan. Performance analysis of cart and c5.0 using sampling techniques. In *2016 IEEE International Conference on Advances in Computer Applications, ICACA 2016*, pages 72–75, Coimbatore, India, 3 2017. Institute of Electrical and Electronics Engineers Inc.
- [5] Tao Ban, Ruibin Zhang, Shaoning Pang, Abdolhossein Sarrafzadeh, and Daisuke Inoue. Referential nn regression for financial time series forecasting. In *International Conference on Neural Information Processing (ICONIP 2013)*, volume 8226 LNCS, pages 601–608, Daegu, South Korea, 2013. Springer, Berlin, Heidelberg.
- [6] Soumya Banerjee, Nashwa El-Bendary, and Hameed Al-Qaheri. Exploring wiki: Measuring the quality of social media using ant colony metaphor. In *Proceedings of the International Conference on Management of Emergent Digital EcoSystems, MEDES'11*, pages 305–312, San Francisco, United States, 2011. Association for Computing Machinery, Inc.
- [7] Elias Bassani and Marco Viviani. Feature analysis for assessing the quality of wikipedia articles through supervised classification, 12 2018.
- [8] Elias Bassani and Marco Viviani. Automatically assessing the quality of wikipedia contents. In *Proceedings of the ACM Symposium on Applied Computing*, volume Part F147772, pages 804–807, New York, United States, 2019. Association for Computing Machinery, Inc.
- [9] F Bellomi and R Bonato. Network analysis for wikipedia. In *Proceedings of Wikimania 2005*, Frankfurt, Germany, 2005.
- [10] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization yoshua bengio. *Journal of Machine Learning Research*, 13:281–305, 2012.
- [11] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python*. O'Reilly, 1 edition, 2009.

- [12] Steven Bird, Ewan Klein, and Edward Loper. Nltk: Natural language toolkit. <https://www.nltk.org/>, 2022. Accessed: 2022-05-10.
- [13] Carl Hugo Bjornsson. *Lesbarkeit durch Lix*. Pedagogiskt centrum i Stockholm. Pedagogiskt Centrum, 1968.
- [14] Joshua E. Blumenstock. Size matters: Word count as a measure of quality on wikipedia. In *Proceeding of the 17th International Conference on World Wide Web 2008, WWW'08*, pages 1095–1096, Beijing, China, 2008. Association for Computing Machinery, Inc.
- [15] P. Boldi and S. Vigna. The webgraph framework i: compression techniques. In *The 2004 World Wide Web Conference (WWW04)*, page 602, New York, United States, 2004. Association for Computing Machinery (ACM).
- [16] Alexei Botchkarev. Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology. *Interdisciplinary Journal of Information, Knowledge, and Management*, 14:45–76, 2019.
- [17] Carl Brucker. Arkansas tech writing: 12th edition. Technical report, Arkansas Tech University, 2009.
- [18] Gabriel De La Calzada and Alex Dekhtyar. On measuring the quality of wikipedia articles. In *Proceedings of the 4th Workshop on Information Credibility, WICOW '10*, pages 11–18, Raleigh, United States, 2010. Association for Computing Machinery, Inc.
- [19] A. Colin Cameron and Frank A.G. Windmeijer. An r-squared measure of goodness of fit for some common nonlinear regression models. *Journal of Econometrics*, 77:329–342, 4 1997.
- [20] Fanny Chevalier, Stéphane Huot, and Jean Daniel Fekete. Wikipediaviz: Conveying article quality for casual wikipedia readers. In *IEEE Pacific Visualization Symposium 2010, PacificVis 2010 - Proceedings*, pages 49–56, 2010.
- [21] Meri Coleman and T. L. Liau. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60:283, 4 1975.
- [22] Alfonso Contreras, Rafael García-Alonso, Marta Echenique, and Fedora Daye-Contreras. The sol formulas for converting smog readability scores between health education materials written in spanish, english, and french. *Journal of Health Communication*, 4:21–29, 2010.
- [23] Luís Couto and Carla Teixeira Lopes. Assessing the quality of health-related wikipedia articles with generic and specific metrics. In *The Web Conference 2021 - Companion of the World Wide Web Conference, WWW 2021*, pages 640–647, Online, Spain, 4 2021. Association for Computing Machinery, Inc.
- [24] Luís Couto and Carla Teixeira Lopes. Assessing the quality of health-related wikipedia articles with generic and specific metrics. In *The Web Conference 2021 - Companion of the World Wide Web Conference, WWW 2021*, pages 640–647, Ljubljana, Slovenia, 4 2021. Association for Computing Machinery.
- [25] CSUN. Readability helps the level. <http://www.csun.edu/~vcecn006/read1.html>, 11 2006. Accessed: 2022-04-12.

- [26] Edgar Dale and Jeanne S Chall. A formula for predicting readability: Instructions. *Educational Research Bulletin*, 27:37–54, 1948.
- [27] Daniel Hasan Dalip, Marcos André Gonçalves, Marco Cristo, and Pável Calado. Automatic quality assessment of content created collaboratively by web communities: A case study of wikipedia. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pages 295–304, Austin, United States, 2009. Association for Computing Machinery, Inc.
- [28] Quang Vinh Dang and Claudia Lavinia Ignat. Quality assessment of wikipedia articles without feature engineering. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, volume 2016-September, pages 27–30, Newark, United States, 9 2016. Association for Computing Machinery, Inc.
- [29] Quang Vinh Dang and Claudia Lavinia Ignat. An end-to-end learning solution for assessing the quality of wikipedia articles. In *Proceedings of the 13th International Symposium on Open Collaboration, OpenSym 2017*, pages 1–10, Galway, Ireland, 8 2017. Association for Computing Machinery, Inc.
- [30] Quang Vinh Dang and Claudia Lavinia Ignat. Measuring quality of collaboratively edited documents: The case of wikipedia. In *Proceedings - 2016 IEEE 2nd International Conference on Collaboration and Internet Computing, IEEE CIC 2016*, pages 266–275, Pittsburgh, United States, 1 2017. Institute of Electrical and Electronics Engineers Inc.
- [31] Thomas G. Dietterich. Ensemble methods in machine learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 1857 LNCS, pages 1–15, Berlin, Heidelberg, 2000. Springer, Berlin, Heidelberg.
- [32] G Druck, G Miklau, and A McCallum. Learning to predict the quality of contributions to wikipedia. *WikiAI*, 8:7–12, 2008.
- [33] IBM Cloud Education. What are neural networks? <https://www.ibm.com/cloud/learn/neural-networks>, 2020. Accessed: 2022-04-26.
- [34] IBM Cloud Education. What is machine learning? <https://www.ibm.com/cloud/learn/machine-learning>, 2020. Accessed: 2022-02-14.
- [35] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 2005.
- [36] Ronen Feldman and James Sanger. *The text mining handbook : advanced approaches in analyzing unstructured data*. Cambridge University Press, 2007.
- [37] Rudolf Flesch. How to write plain english. <https://pages.stern.nyu.edu/~wstarbuc/Writing/Flesch.htm>, 1979. Accessed: 2022-04-11.
- [38] Fortmann-Roe. Understanding the bias-variance tradeoff. <https://scott.fortmann-roe.com/docs/BiasVariance.html>, 2012. Accessed: 2022-04-23.
- [39] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine Learning* 1997 29:2, 29:131–163, 1997.
- [40] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, illustrated edition, 2016.

- [41] A Halfaker, A Kittur, R Kraut, J Riedl Proceedings of the 5th, and undefined 2009. A jury of your peers: quality, experience and ownership in wikipedia. In *WikiSym '09: Proceedings of the 5th International Symposium on Wikis and Open Collaboration*, pages 1–10, Orlando, United States, 2009. Association for Computing Machinery, Inc.
- [42] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer New York, 2 edition, 2009.
- [43] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 11 1997.
- [44] David W. Hosmer, Stanley Lemeshow, and Rodney X. Sturdivant. *Applied Logistic Regression*. Wiley, 3 edition, 1989.
- [45] Meiqun Hu, Ee Peng Lim, Aixin Sun, Hady W. Lauw, and Ba Quy Vuong. Measuring article quality in wikipedia: Models and evaluation. In *International Conference on Information and Knowledge Management, Proceedings*, pages 243–252, Lisbon, Portugal, 2007. Association for Computing Machinery.
- [46] Gareth (Gareth Michael) James. *An introduction to statistical learning: with applications in R*. Springer Science & Business Media, 2013.
- [47] Jun-Mo Jo. Effectiveness of normalization pre-processing of big data to the machine learning performance. *Journal of the KIECS*, 14:547–552, 2019.
- [48] J. Peter Kincaid, Robert P. Fishburne, Richard L. Rogers, and Brad S. Chissom. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. Technical report, Institute for Simulation and Training, University of Central Florida, 1975.
- [49] G. Harry Mc Laughlin. Smog grading-a new readability formula. *Journal of Reading*, 12:639–646, 1969.
- [50] Yang W Lee, Diane M Strong, Beverly K Kahn, and Richard Y Wang. Aimq: a methodology for information quality assessment. *Information & Management*, 40:133–146, 2002.
- [51] Nedim Lipka and Benno Stein. Identifying featured articles in wikipedia: Writing style matters. In *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, pages 1147–1148, Raleigh, United States, 2010. Association for Computing Machinery, Inc.
- [52] Shcherbakov Maxim Vladimirovich, Brebels Adriaan, Natalia Lvovna Shcherbakova, Anton Pavlovich Tyukov, Timur Alexandrovich Janovsky, and Valeriy Anatol'evich Kamaev. A survey of forecast error measures. *World Applied Sciences Journal*, 24:171–176, 2013.
- [53] Andreas Mayr, H. Binder, O. Gefeller, and M. Schmid. The evolution of boosting algorithms: From machine learning to statistical modelling. *Methods of Information in Medicine*, 53:419–427, 1 2014.
- [54] Ramesh Medar, Vijay S. Rajpurohit, and B. Rashmi. Impact of training and testing data splits on accuracy of time series forecasting in machine learning. In *2017 International Conference on Computing, Communication, Control and Automation, ICCUBEA 2017*, pages 1–6, Pune, India, 9 2018. Institute of Electrical and Electronics Engineers Inc.

- [55] MediaWiki. Mediawiki. <https://www.mediawiki.org/wiki/MediaWiki>, 2020. Accessed: 2022-04-27.
- [56] MediaWiki. Mediawiki api. https://www.mediawiki.org/wiki/API:Main_page, 2022. Accessed: 2022-04-27.
- [57] Jae H Min and Chulwoo Jeong. A binary classification method for bankruptcy prediction. *Expert Systems With Applications*, 36:5256–5263, 2009.
- [58] Dunja Mladenić. Feature selection for dimensionality reduction. In *Subspace, Latent Structure and Feature Selection*, volume 3940 LNCS, pages 84–102, Berlin, Heidelberg, 2006. Springer, Berlin, Heidelberg.
- [59] Anthony J. Myles, Robert N. Feudale, Yang Liu, Nathaniel A. Woody, and Steven D. Brown. An introduction to decision tree modeling. *Journal of Chemometrics*, 18:275–285, 6 2004.
- [60] Dewang Nautiyal. ML | underfitting and overfitting. <https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/>, 2021. Accessed: 2022-04-23.
- [61] Brady Neal, Sarthak Mittal, Aristide Baratin, Vinayak Tantia, Matthew Scicluna, and Simon Lacoste-Julien. A modern take on the bias-variance tradeoff in neural networks, 2018.
- [62] Keiichi Nemoto, Peter A. Gloor, and Robert Laubacher. Social capital increases efficiency of collaboration among wikipedia editors. In *HT 2011 - Proceedings of the 22nd ACM Conference on Hypertext and Hypermedia*, pages 231–240, Eindhoven, The Netherlands, 2011. Association for Computing Machinery.
- [63] Encyclopedia of Mathematics. Bayes formula. https://encyclopediaofmath.org/index.php?title=Bayes_formula, 2020. Accessed: 2022-04-26.
- [64] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.
- [65] Matthew J. Page, Joanne E. McKenzie, Patrick M. Bossuyt, Isabelle Boutron, Tammy C. Hoffmann, Cynthia D. Mulrow, Larissa Shamseer, Jennifer M. Tetzlaff, Elie A. Akl, Sue E. Brennan, Roger Chou, Julie Glanville, Jeremy M. Grimshaw, Asbjørn Hróbjartsson, Manoj M. Lalu, Tianjing Li, Elizabeth W. Loder, Evan Mayo-Wilson, Steve McDonald, Luke A. McGuinness, Lesley A. Stewart, James Thomas, Andrea C. Tricco, Vivian A. Welch, Penny Whiting, and David Moher. The prisma 2020 statement: an updated guideline for reporting systematic reviews. *BMJ*, 372, 3 2021.
- [66] Matthew J. Page, David Moher, Patrick M. Bossuyt, Isabelle Boutron, Tammy C. Hoffmann, Cynthia D. Mulrow, Larissa Shamseer, Jennifer M. Tetzlaff, Elie A. Akl, Sue E. Brennan, Roger Chou, Julie Glanville, Jeremy M. Grimshaw, Asbjørn Hróbjartsson, Manoj M. Lalu, Tianjing Li, Elizabeth W. Loder, Evan Mayo-Wilson, Steve McDonald, Luke A. McGuinness, Lesley A. Stewart, James Thomas, Andrea C. Tricco, Vivian A. Welch, Penny Whiting, and Joanne E. McKenzie. Prisma 2020 explanation and elaboration: updated guidance and exemplars for reporting systematic reviews. *BMJ*, 372, 3 2021.

- [67] Raffaele Pugliese, Stefano Regondi, and Riccardo Marini. Machine learning-based approach: Global trends, research directions, and regulatory standpoints. *Data Science and Management*, 4:19–29, 12 2021.
- [68] Susmita Ray. A quick review of machine learning algorithms. In *Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Perspectives and Prospects, COMITCon 2019*, pages 35–39, Faridabad, India, 2 2019. Institute of Electrical and Electronics Engineers Inc.
- [69] Dwaipayan Roy, Sumit Bhatia, and Prateek Jain. Information asymmetry in wikipedia across different languages: A statistical analysis. *Journal of the Association for Information Science and Technology*, 73:347–361, 3 2022.
- [70] Kanchana Saengthongpattana and Nuanwan Soonthornphisaj. Assessing the quality of thai wikipedia articles using concept and statistical features. *Advances in Intelligent Systems and Computing*, 275 AISC:513–523, 2014.
- [71] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models, 8 2017.
- [72] scikit learn. Scikit-learn: Machine learning in python. <https://scikit-learn.org/stable/>, 2022. Accessed: 2022-05-11.
- [73] RJ Senter and Edgar A Smith. Automated readability index. Technical report, Cincinnati Univ OH, 1967.
- [74] Luai Al Shalabi, Ziad Shaaban, and Basel Kasasbeh. Data mining: A preprocessing engine. *Journal of Computer Science*, 2:735–739, 9 2006.
- [75] Amanpreet Singh. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1310–1315, New Delhi, India, 2016.
- [76] Vikash Singh, Michael Pencina, Andrew J. Einstein, Joanna X. Liang, Daniel S. Berman, and Piotr Slomka. Impact of train/test sample regimen on performance estimate stability of machine learning in cardiovascular imaging. *Scientific Reports 2021 11:1*, 11:1–8, 7 2021.
- [77] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45:427–437, 2009.
- [78] Klaus Stein and Claudia Hess. Does it matter who contributes - a study on featured articles in the german wikipedia. In *Hypertext 2007: Proceedings of the Eighteenth ACM Conference on Hypertext and Hypermedia, HT'07*, pages 171–174, Manchester, United Kingdom, 2007. Association for Computing Machinery, Inc.
- [79] B Stvilia, MB Twidale, and L Gasser Culture. Information quality in a community-based encyclopedia. In *Knowledge Management: Nurturing Culture, Innovation, and Technology - Proceedings of the 2005 International Conference on Knowledge Management*, pages 101–113, North Carolina, United States, 2005. World Scientific.
- [80] Besiki Stvilia, Les Gasser, Michael B. Twidale, and Linda C. Smith. A framework for information quality assessment. *Journal of the American Society for Information Science and Technology*, 58:1720–1733, 10 2007.

- [81] Qi Su and Pengyuan Liu. A psycho-lexical approach to the assessment of information quality on wikipedia. In *Proceedings - 2015 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT 2015*, pages 184–187, Singapore, 2 2016. Institute of Electrical and Electronics Engineers Inc.
- [82] Nathan Teblunthuis. Measuring wikipedia article quality in one dimension by extending ores with ordinal regression. In *OpenSym 2021: 17th International Symposium on Open Collaboration*, pages 1–10, Online, Spain, 9 2021. Association for Computing Machinery (ACM).
- [83] Srikar Velichety. Quality assessment of peer-produced content in knowledge repositories using big data and social networks: The case of implicit collaboration in wikipedia. *Data Base for Advances in Information Systems*, 50:28–51, 11 2019.
- [84] Nicholas Vincent and Brent Hecht. A deeper investigation of the importance of wikipedia links to search engine results. *Proceedings of the ACM on Human-Computer Interaction*, 5:18, 2021.
- [85] Ping Wang and Xiaodan Li. Assessing the quality of information on wikipedia: A deep-learning approach. *Journal of the Association for Information Science and Technology*, 71:16–28, 1 2020.
- [86] Ping Wang, Xiaodan Li, and Renli Wu. A deep learning-based quality assessment model of collaboratively edited documents: A case study of wikipedia. *Journal of Information Science*, 47:176–191, 4 2021.
- [87] Wikimedia. Wikistats - statistics for wikimedia projects. <https://stats.wikimedia.org>, 2022. Accessed: 2022-01-04.
- [88] Wikipedia. Rossija/ocenki - vikipedija. <https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B5%D0%BA%D1%82:%D0%A0%D0%BE%D1%81%D1%81%D0%B8%D1%8F/%D0%9E%D1%86%D0%B5%D0%BD%D0%BA%D0%B8>, 2013. Accessed: 2022-05-17.
- [89] Wikipedia. Escala de avaliação – wikipédia. https://pt.wikipedia.org/wiki/Predefini%C3%A7%C3%A3o:Escala_de_avaliao%C3%A7%C3%A3o, 2019. Accessed: 2022-05-17.
- [90] Wikipedia. Évaluation/avancement - wikipédia. <https://fr.wikipedia.org/wiki/Projet:%C3%89valuation/Avancement>, 2021. Accessed: 2022-05-17.
- [91] Wikipedia. List of wikipedias. https://meta.wikimedia.org/wiki/List_of_Wikipedias, 2022. Accessed: 2022-04-27.
- [92] Wikipedia. Wikipedia. <https://en.wikipedia.org/wiki/Wikipedia>, 2022. Accessed: 2022-01-05.
- [93] Wikipedia. Wikipedia: Content assessment. https://en.wikipedia.org/wiki/Wikipedia:Content_assessment, 2022. Accessed: 2022-02-16.
- [94] Wikipedia. Wikipedia: Pages with the most revisions. https://en.wikipedia.org/wiki/Wikipedia:Database_reports/Pages_with_the_most_revisions, 2022. Accessed: 2022-05-10.

- [95] Wikipedia. Wikipedia: Size of wikipedia. https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia, 2022. Accessed: 2022-06-30.
- [96] Wikipedia. Wikipedia: Wikiproject. <https://en.wikipedia.org/wiki/Wikipedia:WikiProject>, 2022. Accessed: 2022-06-07.
- [97] Wikipedia. Wikitext. <https://en.wikipedia.org/wiki/Help:Wikitext>, 2022. Accessed: 2022-05-05.
- [98] Cort J Willmott and Kenji Matsuura. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate Research*, 30:79–82, 2005.
- [99] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, 1:67, 1997.
- [100] Thomas Wöhner and Ralf Peters. Assessing the quality of wikipedia articles with lifecycle based metrics. In *Proceedings of the 5th International Symposium on Wikis and Open Collaboration, WiKiSym 2009*, pages 1–10, Orlando, United States, 2009. Association for Computing Machinery.