

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Value estimation of NFTs through Machine Learning

Mário Mesquita



Master in Informatics and Computing Engineering

Supervisor: Rui Maranhão

Second Supervisor: José Silva

Jan 31, 2023

Value estimation of NFTs through Machine Learning

Mário Mesquita

Master in Informatics and Computing Engineering

Jan 31, 2023

Resumo

Ao longo dos últimos anos, o mercado de arte digital tem visto um rápido crescimento com o aparecimento de Tokens Não-Fungíveis (NFTs). Assentes na segurança e transparência da blockchain, estes agem como registos de propriedade e tornaram possível a criação de escassez em ficheiros digitais, promovendo uma plataforma segura para a sua aquisição e transferência.

NFTs são uma nova classe de ativos na interseção de criptomoedas e arte tradicional, e o mercado ainda é jovem e volátil, muitas vezes visto como polémico e confuso. A sua avaliação depende de fatores subjetivos, como a arte ou a utilidade, e existem grandes discrepâncias de preço entre diferentes NFTs, mesmo considerando tokens da mesma coleção. Esta falta de diretrizes claras, junto com o baixo volume de transações, dificulta a tarefa de determinar o seu preço.

Os donos de NFTs querem perceber quanto vale o seu portfólio, querem saber quando é uma boa altura para comprar ou vender um token, desejam poder utilizar os seus ativos digitais como colateral para empréstimos. Para permitir estas funcionalidades, é necessária uma forma automatizada de calcular o preço dos NFTs, com confiabilidade suficiente para permitir que plataformas e protocolos a integrem nos seus serviços. Este é o caso da Anchorage Digital, uma plataforma líder no mercado de assets digitais para instituições, com quem este trabalho foi desenvolvido.

Começando por familiarizar com a estimação do valor de ações, criptomoedas e arte, neste trabalho exploramos o uso de Machine Learning para determinar o valor de NFTs. A pesquisa inclui uma análise extensa de dados relativos a NFTs, com um foco em coleções *blue-chip* na blockchain Ethereum e, em particular, na coleção BAYC. Segue-se o desenvolvimento e avaliação de vários modelos, explorando diferentes configurações e parâmetros. Abordamos este problema de duas formas: modelando-o como uma Regressão Múltipla, tentando encontrar formas de determinar o preço dos NFTs em função do seu histórico de vendas e metadados; mas também explorando modelos que consideram a natureza sequencial e temporal dos dados.

Os resultados demonstram o potencial de técnicas baseadas Machine Learning para resolver esta tarefa complexa, em particular de Recurrent Neural Networks, fazendo uso de dados de várias fontes para inferir padrões e prever movimentos de preços. Enquanto um método intuitivo que utiliza o último preço de venda da coleção como a predição alcança um valor de MAPE de 21.69% para a coleção BAYC, os nossos modelos foram capazes de fazer uma melhoria significativa e reduzir este valor para 9.42%. Debatemos sobre quais os fatores importantes para este cálculo e identificamos múltiplas sugestões e próximos passos, para que investigações futuras possam partir do nosso trabalho e experiências para construir modelos ainda mais precisos.

Este estudo é uma contribuição importante para investigadores deste tema, investidores que procuram entender melhor o que está por detrás do valor de NFTs, e para o número crescente de plataformas que dependem do cálculo do valor de NFTs para os seus serviços. O trabalho apresenta-se como um passo para a frente em tornar o processo de valorização de NFTs mais transparente e promover um ecossistema de NFTs mais saudável.

Keywords: NFT; bem digital; estimação de valor; machine learning; deep learning

Abstract

Over the last years, the digital art market has seen a rapid growth with the emergence of Non-Fungible Tokens (NFTs). Backed by the safety and transparency of blockchain technology, these act as records of ownership in the digital world and have made it possible to create scarcity in digital files, promoting a safe platform for acquiring and transferring them.

NFTs are a new asset class that are at the intersection of cryptocurrencies and traditional Art, and the market is still young and volatile, often seen as controversial and misunderstood. Their valuation relies on many subjective factors, such as the art, the utility, or the creator, and there are huge price discrepancies between NFTs, even considering tokens of the same collection. This lack of clear guidelines, together with the low amount of transaction volume, makes it difficult to determine their price.

The owners of NFTs want to understand how much their portfolio is worth, want to know when is a good time to buy or sell a particular token, want to be able to leverage their digital assets as collateral for loans. To unlock these use cases, an automated way of calculating the price of NFTs is necessary, with enough reliability that crypto platforms and protocols can trust it to work and enhance their services. This is the case of Anchorage Digital, a Premier Digital Asset platform for institutions, with whom this work was developed with.

Building on top of previous research on the topics of stock, cryptocurrency and art price prediction, in this work we explored the usage of Machine Learning algorithms to determine the value of NFTs. The research includes a thorough analysis and visualization of NFT market data, with a focus on blue-chip collections on the Ethereum blockchain, and in particular BAYC, followed by the development and evaluation of multiple models, exploring different configurations and carefully fine-tuning parameters. We tackled this problem on two sides: modeling it as a Multiple Regression and finding ways of determining the price of NFTs as a function of its sales history and metadata, using Linear regression and Dense Neural Networks; but also exploring models that consider the time-series sequential nature of the data, such as Recurrent Neural Networks.

The results demonstrate the potential of Machine Learning to solve this complex task, in particular of Recurrent Neural Networks, leveraging data from multiple sources to infer patterns and predict price movements. While an intuitive method that uses the last sale value of the collection presents a MAPE of 21.69% for the BAYC collection, our models were able to make a significant improvement and lower this error to 9.42%. We provide insights on what factors are important for this calculation and multiple suggestions and next steps are identified, so that future research can leverage our work and experiences to build even more accurate models.

The findings of this study are an important contribution for researchers on this topic, investors looking to better understand what drives the value of NFTs, and the rapidly growing number of platforms that rely on NFT pricing for their services. This work is a step forward in making NFT pricing more transparent and promoting a healthier ecosystem.

Keywords: NFT; digital asset; asset valuation; machine learning; deep learning

Acknowledgements

A first thank you to Rui Maranhão and José Silva for accepting the task of being my supervisors and helping me navigate this process. Thank you for all the conversations and advice over the development of this project.

Thank you to Anchorage Digital for on-boarding me and giving me the opportunity to understand this problem first-hand and learn more about the industry. Understanding how much this work could impact the company and the NFT market as a whole gave me motivation to continue and do better. I had a great experience over the last months and everyone I crossed paths with was very kind and interested in knowing more about and helping me with my thesis. A special thank you to Nelson, who was there for me on a weekly basis to hear about my progress and concerns, and always ready to question me and give suggestions.

Last but not least, thank you to my family for always being supportive, caring, and feeding me delicious meals and deserts on a daily basis; thank you to my pets for being cute and erasing the stress out of my life; thank you *baba*, *chiquitita21* and *guigui* for being amazing friends and always being there for anything life throws my way; thank you to *The Boys* for the constant stream of League of Legends and always being there to decompress and take my mind of life's worries; thank you *meteor* and *clair* for being my Machine Learning advisors and giving me very insightful advice; thank you to everyone who over these past months showed interest in my work and gave me the opportunity to present it and exchange ideas.

Mário Mesquita

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	2
1.3	Problem	3
1.4	Objectives	4
1.5	Document Structure	5
2	State of the Art	6
2.1	Blockchain	6
2.1.1	Ethereum	6
2.1.2	Smart Contracts	7
2.1.3	NFTs	8
2.2	Machine Learning	9
2.2.1	Regression	9
2.2.2	ARIMA	9
2.2.3	Deep Learning	10
2.3	Existing Research	13
2.3.1	Value estimation of Stocks	13
2.3.2	Value estimation of Cryptocurrencies	14
2.3.3	Value estimation of Art	15
2.3.4	Value estimation of NFTs	16
2.4	Existing Projects	18
2.4.1	nftfi	18
2.4.2	arcade.xyz	18
2.4.3	Upshot	19
2.4.4	NFTBank	19
2.4.5	Zharta	20
2.4.6	Anchorage	20
2.5	NFT Pricing Strategies	21
2.5.1	Last Sale Price	21
2.5.2	Sales History	22
2.5.3	Floor Price	22
2.5.4	Machine Learning	22
3	Methodology	26
3.1	Problem Formulation	26
3.1.1	Time Series	26
3.1.2	Multiple Regression	26

3.1.3	Assumptions	26
3.2	Dataset	27
3.2.1	Data Gathering	27
3.2.2	Gathered Data	29
3.2.3	Data Augmentation	29
3.2.4	Feature Engineering	29
3.3	Evaluation	31
3.3.1	Approach	31
3.3.2	Metrics	31
3.4	Proposed Approach	32
3.4.1	Tech Stack	32
3.4.2	Data Preparation	33
3.4.3	Models	34
3.4.4	Features	37
4	Results	40
4.1	Data Visualization	40
4.2	Initial Models	48
4.2.1	Baseline	48
4.2.2	Linear	48
4.2.3	Dense	50
4.2.4	Multi-step Dense	50
4.2.5	Convolutional Neural Network	53
4.2.6	Recurrent Neural Network	56
4.2.7	Performance	58
4.3	Experiments	58
4.3.1	Shuffling the data	59
4.3.2	Linear model features	61
4.3.3	RNN model features	64
4.3.4	Window sizes	64
4.3.5	RNN models	66
4.3.6	Loss function	68
4.4	Final Model	70
4.5	Other collections	74
4.5.1	CryptoPunks	74
4.5.2	Doodles	74
4.5.3	Azuki	74
4.5.4	Anime Metaverse	74
4.5.5	Pixelmon	79
4.6	Discussion	79
5	Conclusion	82
5.1	Conclusion	82
5.2	Future Work	83
	References	85

List of Figures

2.1	BAYC and CryptoPunks NFTs	7
2.2	Recurrent Neural Network	11
2.3	Bidirectional Recurrent Neural Network	12
2.4	Long-Short Term Memory	13
2.5	Gated Recurrent Unit	13
2.6	nftfi Price Information	18
2.7	arcade.xyz Price Information	19
2.8	Zharta's supported NFTs	21
3.1	Example Data Window	33
3.2	Single Time Step Window	35
3.3	Multi-step Dense Model	36
3.4	CNN Data Window	37
3.5	LSTM Data Window	38
3.6	Input Features Dataframe - Part 1	38
3.7	Input Features Dataframe - Part 2	39
4.1	Sales of Blue-Chip Collections	41
4.2	Sales of Blue-Chip Collections (log10)	41
4.3	Sales of Blue-Chip Collections between 14/01/21 and 20/12/22	42
4.4	Sales of Blue-Chip Collections between 14/01/21 and 20/12/22 (log10)	42
4.5	BAYC - Total Sales without removing outliers	43
4.6	BAYC - Total Sales	43
4.7	BAYC - Average Sales value per day without removing outliers	44
4.8	BAYC - Average Sales value per day	45
4.9	BAYC - Tokens with most sales	45
4.10	BAYC - Sales Statistics	46
4.11	BAYC - Fur Trait	46
4.12	BAYC - All Traits	47
4.13	BAYC - Rarity Scores	47
4.14	Baseline model predictions on 3 separate windows	49
4.15	Baseline model predictions for the BAYC dataset	49
4.16	Linear model metrics during training	50
4.17	Linear model weights	51
4.18	Linear model predictions on 3 separate windows	51
4.19	Linear model predictions for the BAYC dataset	52
4.20	Dense model metrics during training	52
4.21	Dense model predictions on 3 separate windows	53

4.22	Dense model predictions for the BAYC dataset	53
4.23	Multi-step Dense model metrics during training	54
4.24	Multi-step Dense model predictions on 3 separate windows	54
4.25	Multi-step Dense model predictions for the BAYC dataset	55
4.26	Convolutional model metrics during training	55
4.27	Convolutional model predictions on 3 separate windows	56
4.28	Convolutional model predictions for the BAYC dataset	56
4.29	LSTM model metrics during training	57
4.30	LSTM model predictions on 3 separate windows	57
4.31	LSTM model predictions for the BAYC dataset	58
4.32	Final MAPE of the initial models	59
4.33	MAPE results with no shuffling	60
4.34	MAPE results after full data shuffle	60
4.35	MAPE results while shuffling data in each set	61
4.36	Linear model weights for different configurations	62
4.37	Linear model val and test MAPE for different configurations	63
4.38	LSTM model val and test MAPE for different configurations	65
4.39	MAPE results for different window sizes	67
4.40	MAPE results for different RNN models	69
4.41	LSTM model val and test MAPE for different configurations	70
4.42	MAPE results when using MAPE as the loss function	71
4.43	Loss function over epochs for the final LSTM models	72
4.44	MAPE results for the final LSTM models	73
4.45	CryptoPunks - MAPE results	75
4.46	Doodles - MAPE results	76
4.47	Azuki - MAPE results	77
4.48	Anime Metaverse - MAPE results	78
4.49	Pixelmon - MAPE results	80

List of Tables

1.1	NFT collections' statistics, as of 23/01/2023	2
3.1	Dataset Features Explanation	39
4.1	Performance of the initial models	58
4.2	Linear Model configurations	61
4.3	Window sizes of each configuration	66
4.4	Final LSTM configurations	71

Listings

3.1	Models' metrics	32
3.2	Linear model	34
3.3	Single-input Dense model	34
3.4	Multi-step Dense model	35
3.5	Convolutional Neural Network	36
3.6	Convolutional Neural Network	37
4.1	RNN models	66

Abbreviations and Symbols

ARIMA	Autoregressive Integrated Moving Average
BAYC	Bored Ape Yacht Club
dApp	Decentralized Application
DeFi	Decentralized Finance
DL	Deep Learning
DNN	Deep Neural Networks
ERC	Ethereum request for comment
ETH	Ether
GRU	Gated Recurrent Unit
LSTM	Long-short Term Memory
MAE	Mean Absolute Error
ML	Machine Learning
NFT	Non-fungible token
NLP	Natural Language Processing
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
USD	United States Dollar

Chapter 1

Introduction

1.1 Context

Non-fungible tokens (NFTs) are digital assets that are used to represent a wide variety of items, including artwork, collectibles, real estate, and more. They act as records of ownership, enabling authentication of possessions, property and identity in the digital world. "The NBA is using them to sell highlight videos. Artists are using them to sell digital works. Musicians are using them to retain royalties. Twitter's CEO is even using them to sell tweets." (García-Hodges and Dasrath, 2021). Their use has been made possible by the emergence of blockchain technology, which provides a secure and transparent platform for the creation, ownership, and transfer of digital assets, through smart contracts. Each NFT is unique, non-fungible, meaning that they cannot be exchanged for other tokens on a one-to-one basis, unlike fungible tokens such as Bitcoin (Clark, 2022).

NFTs have recently gained significant attention in the art and collectibles market, as these allow for the creation of unique digital assets that can be bought and sold like traditional physical objects. They are financial instruments, making it easier to sell digital files by creating scarcity (Smee, 2021). Even though the market is still young and volatile, it is rapidly growing and has proven to be a potentially lucrative investment opportunity to both individuals and institutions, looking to expose themselves to a new market and diversify their portfolio. In 2022, NFTs' market capitalization peaked at \$41.5B, with total sales reaching \$55.5B, up 175% from \$20.2B in 2021, or 390x from 2020. There were around 45M transactions, 3M unique buyers and 2.7M unique sellers. About 85k NFT collections were launched in 2022, and the month with most trading volume reached \$17.4B in value, with about 7,700 collections having over \$100k on their own (Lian, 2022). Table 1.1 displays statistics about some of the most well-known and established NFT collections today.

However, despite their popularity, NFTs are still seen as controversial and often misunderstood. Unlike stocks or cryptocurrencies, traded every second and with rich price histories, NFTs belong to an asset class defined as low-velocity, similarly to physical collectibles or art (Upshot,

2021). Their value depends on a mixture of subjective factors such as the utility, the art, the creator and the community. This lack of sales data, clear guidelines, and defined standards, makes it difficult to determine a price.

Collection	Launch date	Collection size	Floor price (ETH)	Total Volume (ETH)
CryptoPunks	Jun/2017	10,000	66.49	1,095,153
BAYC	Apr/2021	10,000	73.00	711,169
VeeFriends	May/2021	10,255	5.40	59,566
Doodles	Oct/2021	10,000	7.70	155,518
CloneX	Nov/2021	20,000	5.59	247,133
Azuki	Jan/2022	10,000	15.49	274,271
World of Women	Nov/2021	10,000	1.70	74,345

Table 1.1: NFT collections' statistics, as of 23/01/2023

1.2 Motivation

The motivation for this thesis comes from the recent explosion in interest in NFTs and the resulting growth of the NFT market. While it has been expanding rapidly, there is still a significant lack of understanding of how these assets are valued. This leads to speculation and uncertainty in the market, and makes it difficult for buyers and sellers to make informed decisions. For the creators of NFTs, it can affect the value of their work and their potential profitability.

Determining the value of an NFT is the basis of many other financial use cases besides investing, such as lending and leasing, that you would see in the real world with physical assets. An accurate and automated process will lead to a healthier and stabler NFT ecosystem, promoting more transparency over the factors that influence the pricing decision and unlocking new use cases for NFTs.

This thesis is being developed alongside [Anchorage Digital](https://anchorage.com)¹, a leading U.S.-based crypto platform. Having started with a base proposition of secure, accessible custody, Anchorage has now become a platform of many offerings integrated with the same security-by-design, including: staking, governance, trading, financing, and custom integrations (Mónica and Mccauley, 2022). While the main focus so far has been cryptocurrencies, such as Bitcoin or Ether, the growing interest in NFTs means institutions have begun to look for partners to aid in acquiring and keeping their

¹anchorage.com

tokens safe. In 2021, Anchorage Digital helped Visa navigate the process of buying CryptoPunk #7610, becoming the "first publicly-traded institution to actually own an NFT." (Pimentel, 2021).

The pricing of NFTs is the backbone of many use cases, such as calculating how much a user's portfolio is worth, influencing not only the value seen on a dashboard, but critical components such as invoicing and how much Anchorage charges for custody fees. It is a current priority and a reliable way to estimate their value is key in allowing Anchorage to scale its services to more clients.

1.3 Problem

Determining the value of an NFT is the basis for many financial use cases, such as knowing what is a good entry point to invest, or being able to use it as collateral for a loan. However, this task is complex and there are no clear guidelines on how to do it. Every day we see NFTs trading from prices ranging a couple dollars to thousands or millions, with considerable price differences even among similar tokens, and the reasons are often not understood. The value of each NFT is dependant on multiple factors, objective and subjective ones, such as:

- **Utility** — What is the utility behind the NFT? What will you gain from owning it?
- **Trading history** — What was the last sale price? For what price are similar NFTs trading? Tokens in the same collection will usually go for identical prices.
- **Creator** — Who created the NFT? The name of famous personalities or artists brings value to the token.
- **Rarity** — How rare is this NFT compared to similar ones? NFTs often have a rarity score and rarer ones tend to fetch higher prices.
- **Scarcity** — How many similar NFTs will be issued? Factors such as collection size influence the value for its exclusivity.
- **Art** — How much do you like the art? Most NFTs have a visual representation and it may influence a buyers valuation based on their personal art taste.
- **Community** — Who else owns similar NFTs? Active communities or famous personalities may considerably raise the price of a collection as a whole.
- **Chain** — In what blockchain does the NFT reside? Chains like Ethereum or Solana are more popular and may drive the value up because they have more active wallets.
- **Market** — How is the overall NFT / crypto / stock market performing? Trends in these markets may affect a particular NFT valuation.
- **Trivia** — What is the history behind the NFT? If it was previously owned by a personality or has any story or meme associated with it the value may be affected.

Unlike stocks and cryptocurrencies that are liquid and high-velocity assets, with transactions every second and a rich trading history, NFT sales are much scarcer. Even in famous collections such as BAYC, many tokens have little to no sales at all. These are more easily comparable to physical collectibles or art. When an investor wants to buy an NFT, he is often faced with a scenario where the NFT hasn't been sold before, or the last sale was months ago. According to a report by [Artsy \(2019\)](#): "The lack of access to artworks' prices was the most frequently mentioned roadblock for collectors when trying to buy art.". To make an informed decision, he must take all of the above mentioned factors into account and come up with a valuation. In alternative, similarly to the art market, he can outsource this valuation to an expert, at a cost. This process is time-consuming and constitutes a high entry barrier for potential investors, especially in a fast-paced market where things can change very quickly.

Being able to price your NFT is also important for unlocking many other financial use cases. Taking the example of a loan protocol that allows you to use your NFTs as collateral, how much should your assets be valued at? If the value is too low, you may not be able to borrow as much as you require. If the value is too big, the lending protocol might be taking unnecessary risk. An ideal protocol would be automated and self-service, where you could at any time ask for an appraisal and get your loan immediately, taking advantage of the speed and efficiency that other Decentralized Finance (DeFi) services have created for the crypto markets.

At Anchorage Digital, a company at the forefront of crypto services, NFT pricing is relying on the opinion of experts that, for each token, set a fixed price and update it as required. This process is manual, time-consuming and not scalable. There is a clear need for an improved, automated, NFT pricing solution.

1.4 Objectives

The main goal of this thesis is to examine the factors that influence the pricing of NFTs and to use machine learning techniques to automate the pricing process. As it is being developed alongside Anchorage Digital, the objective is also to understand how it can be integrated into their ecosystem.

The focus of this work will be on blue-chip NFT collections on the Ethereum blockchain. These are the most established and stable tokens, considered a safer long-term investment.

To achieve this, the following specific objectives will be pursued:

1. Study existing research on NFTs and the factors that influence their pricing. This will involve familiarizing with Blockchain technologies, examining the characteristics of NFTs and studying the current state of the NFT market.
2. Conduct a literature review of existing research about the pricing of art, stocks and cryptocurrencies, understanding what knowledge and techniques can be transferable to the pricing of NFTs.

3. Understand the importance of NFT pricing for Anchorage Digital and how it is being used across its services. This will involve getting to know Anchorage's product, getting in touch with engineers and digging into the Pricing Service's code base.
4. Identify and gather data about NFTs and the factors that influence their pricing, so that we can use it to build our ML models. This will involve interacting with a variety of sources, including NFT marketplaces and social medias.
5. Develop machine learning models for NFT pricing, by selecting relevant features from the data set and testing different architectures.
6. Evaluate the performance of the ML models, assessing their accuracy and reliability, in order to understand what the best pricing method is.
7. Draw conclusions about the factors that influence NFT pricing and the potential of machine learning techniques to automate the pricing process.

1.5 Document Structure

After the introduction and contextualizing the problem and goals of the research, this work starts by analysing the State of the Art about Blockchain, NFTs, Machine Learning models and NFT valuation, on Chapter 2. Next, in Chapter 3 we discuss the strategy and investigation methods we decided to use and justify them, defining the problem as a Machine Learning task, going over the data we used, and mentioning on a high level the models we tackled the problem with. Chapter 4 presents the work that was developed and its results, together with data and charts to help understand them. In the same chapter, we analyse the results and discuss them, interpreting what they represent. Finally, Chapter 5 concludes by recalling our initial objectives and how they were met, perspectives of future work and improvements, and some limitations of the developed work.

Chapter 2

State of the Art

This chapter goes through existing literature and state of the art about the areas relevant to the problem at hand. The goal is to understand these topics, acquire context about the problem, and learn what knowledge from other works that tackle similar problems can be transferred to this research.

2.1 Blockchain

A blockchain is a distributed and decentralized database that allows the storage of data with security without the need for a trusted third party. These characteristics have brought a lot of attention to them and to the potential that they have to revolutionize many industries, such as healthcare, finance or supply chain management.

This concept was first introduced by Satoshi Nakamoto in 2008 ([Nakamoto, 2008](#)), whose identity remains a mystery. In the paper, the author presents "peer-to-peer version of electronic cash", the digital currency Bitcoin, and the concept of a blockchain is used to track the ownership of these coins. It not only shows the benefits of such a system, but also makes apparent many other potential uses, such as tracking supply chain logistics or managing medical records.

Blockchains store data in blocks, linked to each other through cryptographic hash functions. They are maintained by a group of participating nodes, without any single entity controlling the whole network, hence being called decentralized. They are by design secure and resilient, as they remove the possibility of a single point of failure, making them resistant to modification by malicious actors.

2.1.1 Ethereum

One of the most relevant Blockchains today is [Ethereum](#)¹, which not only provides security, scalability and decentralization for its native currency, Ether, but also serves as a software platform. It natively supports Smart Contracts, which are the essential tool behind decentralized applications (dApps) and make it possible to create tokens such as NFTs.

¹ethereum.org/en

Ethereum was developed by Vitalik Buterin (Buterin, 2014), building upon the original concepts proposed by Bitcoin. It allows the creation of self-executing contracts with the conditions and terms of agreement being explicitly and transparently written in lines of code. This makes it possible to automate and enforce the performance of a contract, making sure all parties follow the pre-defined rules.

It is the most active and relevant NFT platform, where the majority of widely-known and established NFT projects reside, such as Bored Ape Yacht Club (BAYC ²) and CryptoPunks ³, displayed in Figure 2.1. In 2022, Ethereum had 95% of NFTs trading volume, 47% of transactions and 71% of protocols, consolidating itself as the most widely used platform for NFTs (Lian, 2022). In Q1, Ethereum achieved 105.58 million total transactions, with 1.45 million smart contracts created and, on average, 329,900 daily active addresses (Alchemy, 2022).

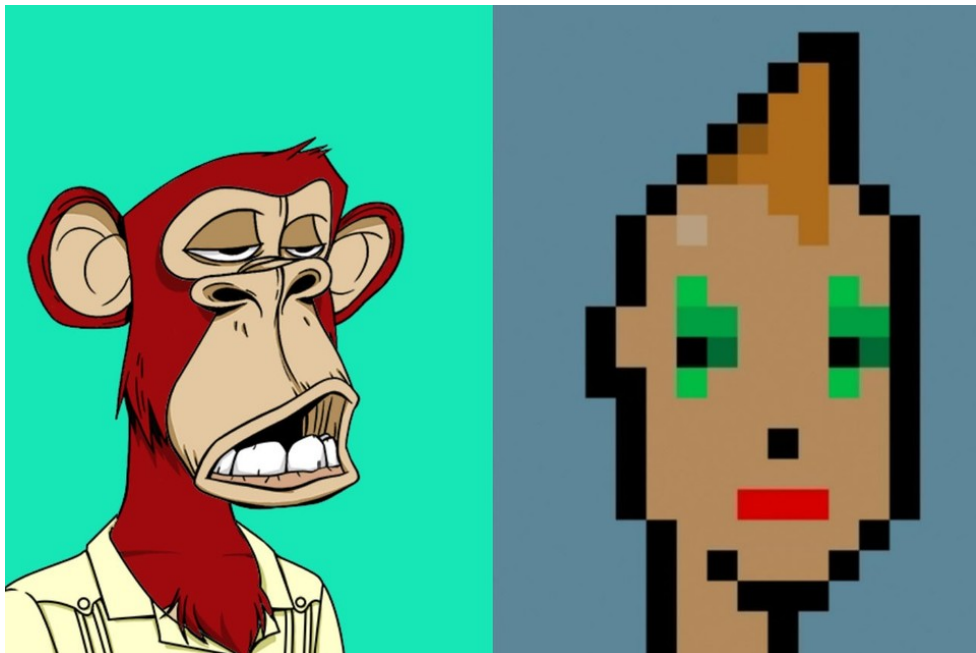


Figure 2.1: BAYC and CryptoPunks NFTs

2.1.2 Smart Contracts

Smart contracts (Frankenfield, 2022) (IBM, 2022) are the key component in dApps that allow them to be considered decentralized. These are computer programs stored on a blockchain that automatically execute when certain conditions are met, without the need for a third party. Instead of having someone mediating an agreement between a buyer and a seller (which most times introduces overhead such as time and fees), the rules of the agreement are encoded in the contract and anyone can use and trust it to work. They increase the efficiency and transparency of transactions, as well as reducing the risks of fraud or errors.

²boredapeyachtclub.com

³cryptopunks.app

Smart contracts are permanently stored on the blockchain and can not be altered. All of the tokens that reside in the Ethereum blockchain, fungible or non-fungible, are defined through a Smart Contract.

Smart contracts vary from blockchain to blockchain, in terms of implementation and features. Relevant blockchains that support smart contracts are Ethereum, Solana and Cardano. For example, on Ethereum, these can be defined through the Solidity or Viper programming languages.

2.1.3 NFTs

NFT stands for Non-Fungible Token (Clark, 2022), and these are crypto tokens stored on a blockchain through a smart contract. "Non-fungible" means that each token is unique and can not be replaced by any other token. This is in contrast with other tokens such as Bitcoin, where a Bitcoin is exactly the same as any other Bitcoin and they can be traded without any difference. A good example to help in understanding this is Art. There is only one Mona Lisa, painted by Leonardo da Vinci, and even if someone else replicates it perfectly, it is not the original one and they will have different values.

Built on top of the decentralization of blockchains, NFTs allow for representing identity and ownership in the digital world, providing a layer of security and trust that was not possible before. This creates a new class of assets, digital ones, whose value is dependant on many factors such as their utility. For example, an NFT could represent a physical house, and the owner of the NFT would be the rightful owner of the house. This can be automated through Smart Contracts and is actually already happening.

On the Ethereum blockchain, they are typically based on the ERC-721 standard (Entriiken et al., 2018), which defines a common interface and functionality for NFTs to adopt. Put simply, each token is unique and can not be interchanged. In some cases they may adopt the ERC-1155 standard, becoming semi-fungible tokens, which are preferred for some use cases, mostly related to gaming.

As mentioned before, this work focuses on NFTs that are considered 'blue-chip'. These are defined by being widely known and established, having a much stabler (and higher) price than other tokens and considered a safer long-term investment. With these tokens there is a high level of confidence that their value will not disappear suddenly, as is frequent in many new NFT collections that are still finding their market and users. Examples include the Bored Ape Yacht Club or CryptoPunks. This stability makes it more appealing to companies/institutions, that can afford paying more expensive NFTs and avoid risky assets, and due to the interests of Anchorage Digital and its clients, they will be the focus.

Currently, OpenSea is by far the most popular and widely used NFT marketplace, where users can browse, buy and sell all sorts of tokens. It was founded in 2017 by Devin Finzerand and Alex Atallah, and has since then grown to have over 1 million users, with 80 million different types of NFTs available and more than \$2 billion in trading volume as of early 2022 (Wise, 2022). Besides being a marketplace, OpenSea also provides an API for developers to build their own applications on top of the data related to NFTs, such as collection info and transactions history. Querying this

data directly from the blockchain is very costly and runs into performance and bandwidth issues, since blockchains do not provide a query language like regular databases (DeHive, 2022). The way you would do it would be to go through each block in sequence trying to find the data you need, but this is very non efficient. As such, indexing services of this data like OpenSea provides are of utmost importance.

2.2 Machine Learning

The purpose of this work is to explore Machine Learning and apply it to the NFT pricing problem. As such, in this section we present a brief overview about the algorithms and techniques that will be considered, providing the necessary context on how they could be applicable.

2.2.1 Regression

This can be seen as a Regression problem, a type of Machine Learning algorithm that falls under the category of supervised learning, where the model is trained with both input features and output labels. This type of technique investigates the relationship between the input features (the independent variables) and the outcome values (dependant variables). As we take the output labels, this allows the model to, over training, establish a relationship among the variables, by estimating how they affect each other. This relationship can then be leveraged to predict the outcome of new input data that hasn't been seen before.

One of the most common Regression techniques is Linear regression, that assumes a linear relationship between the independent and dependent variables. This can be considered a Simple Linear Regression, if we are using one independent variable, or a Multiple Linear Regression, in case there are more independent variables.

2.2.2 ARIMA

To predict future trends in time series data, ARIMA is one of the most common approaches (Hayes, 2022). It stands for Autoregressive Integrated Moving Average, and is a statistical analysis model that predicts future values based on past ones using linear regression. As we will see in the next sections, it is often applied to stock and cryptocurrencies' price prediction.

Its nomenclature can be composed in 3 separate terms:

- Autoregression (AR): meaning that the model uses relationships between current and past values of the data
- Integrated (I): meaning that the time series data is stationary, by differencing the raw observations
- Moving Average (MA): the model's forecast depends on a linear window of past values

It is typically denoted by three parameters (p, d, q) , that control how the model works. p corresponds to the number of "lag observations" in the model, d is the number of times the observations are differenced to make the data stationary, q is the size of the moving average window. These can take the value 0, meaning that component is not included in the model. As such, the ARIMA model can become ARMA ($d=0$), AR ($d=0$ and $q=0$) or MA ($p=0$ and $d=0$).

While they can be very accurate under the appropriate conditions, one of the short-comings of this model is that the (p, d, q) parameters need to be defined manually, and finding the most accurate combination can take some time in a trial-and-error process. Additionally, the performance of this model highly depends on the amount and reliability of the historical data. It is necessary that the data is continuous, in regular time intervals, so that it can apply the moving average component. Finally, this is most suitable for univariate time series data, where we only have one variable. Other variations, such as the VARMA family, attempt to target models with more variables.

2.2.3 Deep Learning

Deep Learning is a subfield of Machine Learning that takes inspiration in the structure and functioning of the human brain, the neurons, to create Artificial Neural Networks (Brownlee, 2019). These consist on multiple layers of nodes connected to each other through weights. The idea behind this structure is that the data fed into the input layer will be progressively refined, each layer building on top of each other, to reach the final prediction/classification in the output layer(s). These networks require large amounts of computing power and training data, but have proved themselves to outperform traditional ML algorithms in many scenarios.

There are many DL algorithms and new architectures are constantly being researched. These can broadly be split into categories according to their constituent parts and the problems they solve, such as:

- **Multi-layer Perceptron** — MLPs are a fully connected class of neural networks, composed of simple nodes in 3 layers: input, hidden and output. These are widely used for prediction and classification tasks.
- **Convolutional Neural Networks** — CNNs introduce convolution and pooling layers, capable of applying filters and down/up-sampling the data. These are especially adequate for image processing and object detection.
- **Recurrent Neural Networks** — RNNs introduce memory cells, with connections from a node to itself. This means a node is capable of using previous outputs as inputs for next steps, effectively having memory and taking into account previous occurrences for future processing. Example usages include time-series analysis and natural language processing.
- **Generative Adversarial Networks** — GANs are networks that generate new data examples similar to the training data. They are composed of 2 main components: a generator, that learns how to generate data; a discriminator, capable of distinguishing between real and fake (generated) data.

2.2.3.1 Recurrent Neural Networks

Due to the nature of the problem, based on time-series data, in this project we will be focusing on Recurrent Neural Networks (IBM, 2020) to tackle the problem. These are a class of neural networks that introduce memory cells, allowing previous outputs to be used as inputs across different timesteps, as shown in Figure 2.2. These are the perfect scenario for sequential data, where the size of the input doesn't have to match the size of the output and you want to capture relationships between the different elements. The weights in each cell are shared across timesteps, not introducing additional overhead. They are trained through backpropagation through time (BPTT), which is a variant of the typical backpropagation algorithm.

They are widely used in problems with sequential input data, such as prediction problems, machine translation, speech recognition, video tagging, among others. This is the case of our problem, where we will be focusing on time-series NFT sales data, as explained in Chapter 3.

However, base RNNs suffer from problems such as having trouble capturing long-term dependencies in the input, losing the connection between the data over time epochs. Other architectures have been developed to improve on such shortcomings, presented in the next sections.

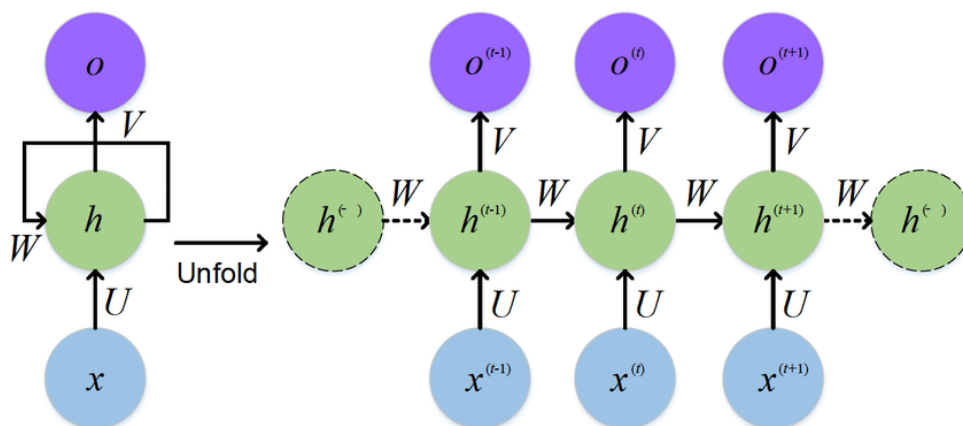


Figure 2.2: Recurrent Neural Network

2.2.3.2 Bidirectional Recurrent Neural Networks

BRNNs are a variant that take the capability of using information from other timesteps even further. Not only can they draw from previous inputs to make predictions about the current state, like the base RNNs, but they can also pull in future data to improve the accuracy. In the case of NLP, analysing a full sentence in sequence word by word, the last words can be used as context to improve the analysis of the first ones. This architecture is displayed in Figure 2.3.

2.2.3.3 Long Short-Term Memory

LSTMs (Hochreiter and Schmidhuber, 1997) were developed as a means to solve the vanishing gradient problem, addressing the problem of long-term dependencies raised above. In this archi-

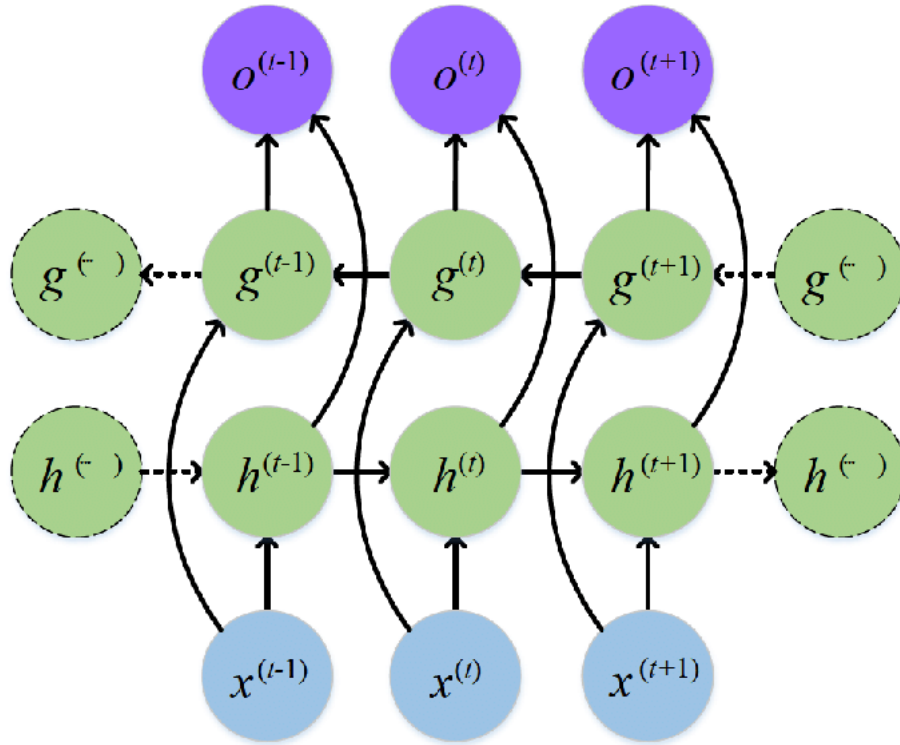


Figure 2.3: Bidirectional Recurrent Neural Network

ture, each cell has a state composed of 3 gates, as shown in figure 2.4: input gate, output gate and forget gate. The forget gate is responsible for forgetting information from previous states, deciding what part of it should be kept in the current one (from none to all of it). The input gate takes new input and combines them with the previous state, deciding which relevant new information can be added. The output gate takes all previously calculate values in account to determine the value of the next hidden state.

As opposed to base RNNs, where the cell state is similar to a conveyor belt and runs straight in a linear way, LSTMs introduce the ability to add or remove information at each step through the gates. This makes it possible to retain information for longer, and be more selective about what parts are important or can be forgotten. LSTMs have increased performance than base RNNs for most tasks.

2.2.3.4 Gated Recurrent Unit

The GRU variant (Cho et al., 2014) is similar to the LSTM, in that it address the short-term memory problem of base RNNs. However, instead of a "cell state" to handle information, it uses hidden states. Moreover, it only contains two gates, reset and update, instead of the LSTM's three. This makes the GRU architecture simpler and more adequate for problems with smaller datasets. Its architecture is present in Figure 2.5.

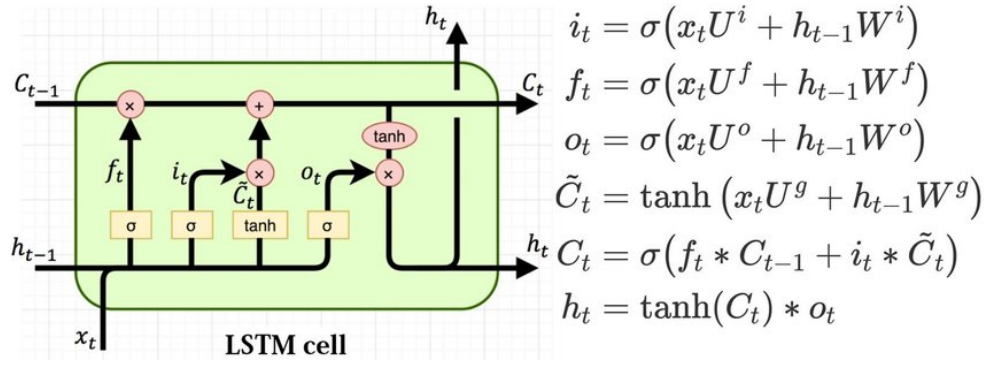


Figure 2.4: Long-Short Term Memory

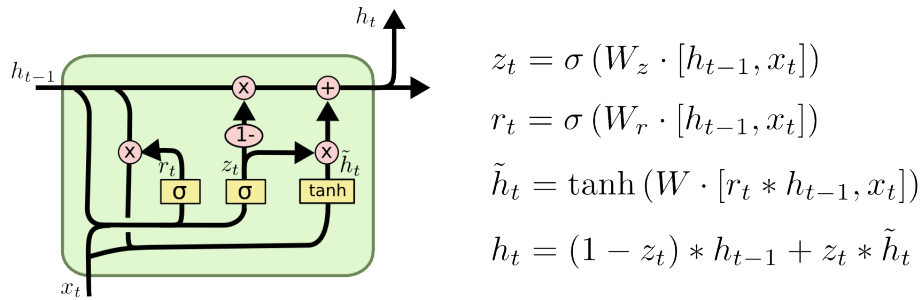


Figure 2.5: Gated Recurrent Unit

2.3 Existing Research

2.3.1 Value estimation of Stocks

The prediction of stock market prices is a topic with a lot of attention and research, as there is a big financial incentive to find better solutions. Even though the asset class is different from NFTs, in its liquidity and velocity, the problem can be defined in similar terms, as a time-series forecasting considering the historic prices. For this reason, in this section we study solutions to tackling this problem.

[Ariyo et al. \(2014b\)](#) explore the use of Autoregressive Integrated Moving Average (ARIMA) models for stock prediction, modeling the problem as a time series prediction and experimenting with multiple parameters to find the best fit. Their results demonstrate the potential of these techniques for short-term predictions, competing with other already existing techniques such as those based on Artificial Neural Networks.

In another article, [Ariyo et al. \(2014a\)](#) go further in comparing ARIMA and Artificial Neural Networks models for stock prediction. With data from the New York Stock Exchange (NYSE), they analyse the forecasting performance of these models and, even though both of them achieve satisfactory results, propose the superiority of neural networks over ARIMA.

Another attempt at using traditional Machine Learning using Support Vector Machines was made by [Xiao et al. \(2020\)](#). They start by analysing the shortcomings of current stock market

forecasting methods and standard SVM models. Then they propose a model designated ARI-MA-LS-SVM, combining a cumulative auto-regressive moving average, with a least squares support vector machine component. They conclude this combination of models achieves higher performance than the single models.

A study on sentiment analysis is made by [Jin et al. \(2020\)](#), by using a Convolutional Neural Network over comments made by users to identify bullish/bearish trends. They add this sentiment to the data and adopt a LSTM model to predict prices, due to its advantages of analysing time-based relationships in data. The paper concludes that the addition of investors' emotional tendency is effective in improving the prediction results.

[Yu and Yan \(2020\)](#) identify the problem of stock prediction as being nonlinear and time-dependent, due to the complex, incomplete and sometimes fuzzy information of financial data. As such, they study the application of deep neural networks (DNN) to stock price prediction, comparing the performance of multiple models, including: an ARIMA linear model, a Support Vector Regression ML model, a deep Multilayer Perceptron, a deep LSTM model with no PSR (phase-space reconstruction), and a deep LSTM model with PSR. This comparison is made for multiple stock indices across different time periods, and suggests that the DNN-based model has a higher prediction capacity than other models.

Due to the volatile nature of the stock market, as a nonlinear time series prediction, [Lu et al. \(2021\)](#) argue that simple models will have trouble predicting it. Their paper proposes a CNN-BiLSTM-AM architecture, combining Convolutional Neural Networks (CNN), bi-directional long short-term memory units (BiLSTM), and an attention mechanism (AM). The CNN component is used to extract input features, the BiLSTM makes the prediction, the AM is used as a memory component so that the model can capture the influence of predictions across different time steps. By comparing it to other methods, including Multilayer Perceptrons (MLP), CNNs, RNNs, LSTMs, BiLSTM, CNN-LSTM, CNN-BiLSTM and BiLSTM-AM, the results show the proposed model achieves the best performance, with a MAE value of 21.952 and a RMSE value of 31.694.

As a summary, the pricing of stocks is a difficult task due to the volatility and non-linearity of markets, often modeled as a time-series prediction problem. Multiple Machine Learning techniques have been attempted at solving this problem, but it seems to be the consensus that Deep Learning models have an edge at this prediction task.

2.3.2 Value estimation of Cryptocurrencies

Predicting Cryptocurrencies' prices stands to be a similar problem to predicting the value of stocks, as both asset classes are similar. However, cryptocurrencies share the characteristic with NFTs of being built on top of a blockchain, an exciting recent technology that is still full of uncertainty among investors, and characterized by a high volatility.

In an attempt to predict changes in Bitcoin and Ethereum prices, [Abraham et al. \(2018\)](#) utilize Twitter and Google Trends data to build a linear model that is capable of accurately predicting the direction of price changes. By analysing tweets, the authors point out that twitter sentiment is not as relevant as one would think, since, overall, the sentiment tends to be positive among investors

that have a bias towards their investments. Rather than sentiment, the authors suggest that tweet volume is more important, as well as the search volume in Google Trends. This model provides a way for investors to make better and informed decisions, and could be an important input to a price prediction algorithm.

Patel et al. (2020) tackle the problem of cryptocurrencies' price prediction through a Deep Learning approach, focusing on two coins, Litecoin and Monero. They build on top of previous research, recognizing that Machine Learning algorithms have proved to be more effective than traditional time-series models, such as ARIMA, SARIMA, ARCH or GARCH. The paper proposes a hybrid model based on both LSTM and GRU architectures. Results indicate an improved performance in comparison with a traditional LSTM model, with smaller prediction errors. They highlight the use of sentiment data as a future improvement.

Jay et al. (2020) propose a stochastic neural network model, inspired on the random walk theory that is often applied to stock prices. The model introduces randomness in the layers to simulate market volatility. The authors train Multi Layer Perceptron (MLP) and Long Short-Term Memory (LSTM) models, over data related to Bitcoin, Ethereum, and Litecoin. When compared to deterministic versions, the stochastic models achieved best performance, with average relative improvement reaching 7.41%.

In *Machine Learning Algorithm for Cryptocurrencies Price Prediction* (Awotunde et al., 2021), a LSTM model is applied to the problem of cryptocurrency price prediction, based on factors such as: available price, close price, high price, low price, volume and market cap. Results show a superior performance of the proposed model when compared to a simpler LSTM, or traditional price prediction methods such as ARIMA, ARCH, or GARCH. The author was able to achieve an accuracy of 67.43%.

All in all, there seems to be a clear preference for using Machine Learning models for the prediction of cryptocurrency prices, with particular usage of LSTMs. Using sentiment data from Twitter and Google Trends may also be interesting areas to explore.

2.3.3 Value estimation of Art

Similarly to NFTs, the traditional Art market is characterized by the uniqueness of each individual piece of art and for its low velocity, meaning transactions are scarce and there is not a lot of data about past sales. It can go decades between sales, and it becomes difficult to establish any pattern about the change in value. This job usually falls back to professional art appraisers, that must develop a presale price for each artwork that goes into auction, based on factors such as the artist, country of origin, age of the art piece, dimensions, rarity, signature, materials, subject, and condition (Bailey, 2020).

As such, the Value estimation of Art presents itself as a similar problem to the value estimation of NFTs. In this section we analyse previous attempts at this problem, in the hopes of learning useful insights to apply to our work.

In an attempt to understand the uncertainty of the art market, Beckert and Rössel (2013) study the factors behind the pricing of art. They argue the value of a art work lies in the assessment that

experts in the field make of it, such as gallery owners, curators, critics, art dealers, journalists and collectors. The economic value buyers perceive is based on the reputation these entities confer the artist. To test this approach, they analyse two datasets containing data on art prices and information from the biographies of artists.

Contrary to what one might assume, [Aubry et al. \(2019a\)](#) demonstrate that the visual features of an artwork are not as important in determining its price. They show that models based on numeric and textual data perform considerably better than those based exclusively on images. Similarly, [Bailey \(2020\)](#) argues that signals of reputation and prestige of the artist need to be considered in the pricing formula. He also discusses the importance of techniques that rely on mining textual data from biographies and social media presence.

In *Painting2Auction* ([Worth, 2020](#)), the author highlights the problem of pre-auction predictions of painting prices. He proposes new models based on Deep Learning for image-to-price prediction that take into account the RGB data of the input image, but also market data such as prices and images of recently-auctioned paintings. Through a CNN, the similarity between the input painting and previous auctioned ones is compared. This similarity is then fed, together with prices data, to an LSTM model to predict the input painting's price. Results demonstrate a higher performance when compared to a naive CNN.

Biased Auctioneers ([Aubry et al., 2019b](#)) explored the application of a neural network to generate price predictions for art, relying on both visual and non-visual characteristics. While the aim was not to come up with the best possible art predictor algorithm, the paper sheds light on the potential and key factors of price prediction. The dataset they use contains information about: the artist, the artist's nationality, birth and death years, the artist style, the artwork creation year, width, height, materials, title, markings, the auction house, the auction house's location, the auction's month and year. They argue that while it is unlikely that ML models completely replace human judgement, they will likely become important tools for investors and intermediaries.

As a summary, multiple attempts have been made at automating the prediction of art prices, with a preference towards Machine Learning and Deep Learning techniques. There is a general consensus that, while visual features may be relevant, more importance should be given to non-visual features.

2.3.4 Value estimation of NFTs

As the main goal of this thesis is the value estimation of NFTs, in this section we take a look at existing research around this topic. Additionally, we analyse already launched projects that involve NFT pricing and review how they perform this task.

[Lommers \(2022\)](#) argues that NFTs present a difficult pricing problem due to there being numerous idiosyncrasies in the market that have to be considered. These include their unique nature, limited data availability, low liquidity, high dimensionality of features, and the volatility of NFT prices. Their analysis is that while Machine Learning algorithms are usually adequate for such a problem, there has to be a significant understanding of the model and an appropriate feature selection in order to prevent overfitting. Since the amount of available data is small, and due to

the curse of dimensionality in NFT metadata features, it is suggested to first perform a step of dimensionality reduction by feature selection and/or transformation. They identify the prices of Collection-based NFTs to move in sync with each other, where investors mainly use the traits to compare the value of NFTs within the same collection.

In an attempt to identify behavioral price determinants of NFTs, [Horky et al. \(2022\)](#) used a LASSO regression text mining approach to extract insights out of art NFTs descriptions from SuperRare. Results indicate that file size and market size are positively correlated with increased expected values, the gas price is not very relevant, and that the file format can also influence the valuation, where mp4 files tend to be more valuable.

Considering the social media communities that are built and evolve alongside NFT projects, [Luo et al. \(2022\)](#) explore how their online presence impact the NFT price moves, through experiments involving the number of tweets and extracted word features from social media activity. They find that terms related to the general market and related to NFTs have both a positive contribution towards NFT pricing moves.

An analysis of the NFT market structure and evolution is made in *Mapping the NFT revolution* ([Nadini et al., 2021](#)). Statistics about collections and prices are presented, as well as an investigation about the traders and NFT networks. Finally, a simple ML algorithm is developed to estimate NFT pricing based on transaction history, and it is shown that incorporating visual features improves accuracy.

[Dowling \(2022\)](#) explores relationships between NFTs and cryptocurrencies, as the former market has emerged from the later. Different indexes suggest varied levels of correlation, with some finding a connection and others indicating NFTs are a new market. Overall, two main conclusions are drawn from this study: NFT pricing seems to be distant from cryptocurrency pricing in terms of volatility transmission, meaning that they are low-correlated and can be interesting assets for an investor that wants to diversify its portfolio; despite low volatility transmission, it is suggested there is some co-movement between the two sets of markets and there may be value in applying understanding of cryptocurrency pricing to NFTs.

[Mekacher et al. \(2022\)](#) study rarity patterns in NFT collections and how they drive price dynamics. By analysing over 3.7M transactions involving 1.4M NFTs, across 410 collections, they find that, on average, NFTs of higher rarity: sell for higher prices, are less traded, achieve higher ROI (return on investment), and carry less risk.

Through a two-stage ML approach, [Zheng \(2022\)](#) incorporates both visual and non-visual information of NFTs to predict their price. The study finds that the impact of visual features is less significant when compared with non-visual data such as total number of bids in history and the collection the NFT comes from.

As a summary, the pricing of NFTs is a complex topic due to the lack of sales, high dimensionality of features, and high volatility. Previous studies have been performed at identifying what factors matter for the pricing of NFTs, indicating that:

- Non-visual features are more impactful than visual ones

- There is a correlation between the NFT market and the overall crypto market
- Rarer NFTs drive higher prices
- Social media presence and activity positively influences price movements

2.4 Existing Projects

In this section we go over already launched projects that involve NFT pricing and analyse how they perform this task, in an attempt to understand what is the State of The Art outside of research.

2.4.1 nftfi

nftfi⁴ is a platform for borrowing and lending NFTs, where you can list your assets and get loans in the form of wETH or DAI. A borrower will set their desired terms, and lenders will make offers that the borrower can accept or not. As such, in this platform it is the users that decide on the price of each NFT and this appraisal is not automated.

As a way of helping the decision, they include 2 automated NFT valuations, using Upshot (2.4.3) and NFTBank (2.4.4). These provide price suggestions to the user when the asset is supported, as can be seen in Figure 2.6.

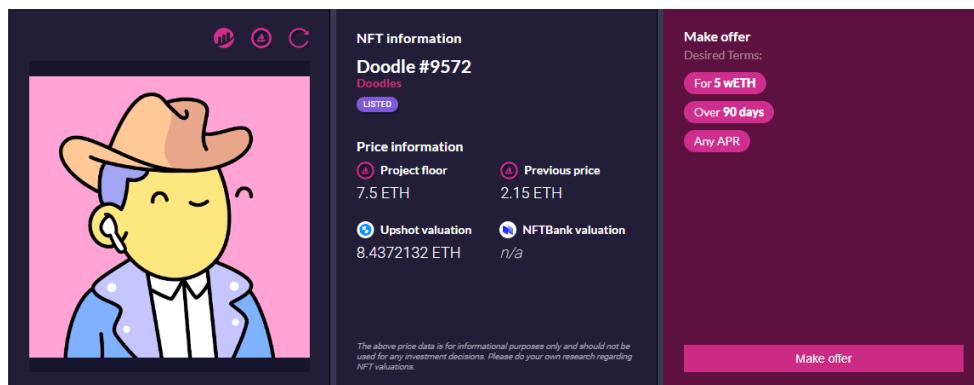


Figure 2.6: nftfi Price Information

2.4.2 arcade.xyz

arcade.xyz⁵ is another platform for borrowing and lending NFTs, working in the same way as nftfi. Borrowers list their NFTs and set desired terms, lenders can accept them or counter offer, making the appraisal's decision lie on the users' hands.

As a price suggestion, they use NFTBank's API, as can be seen in Figure 2.7.

It is important to note how, in both systems presented, the valuation of NFTs is determined by the users and the agreement they reach. The problem with this approach is that both users will

⁴nftfi.com

⁵arcade.xyz

always try to make the situation more favorable to them, turning the process into a negotiation. Such a system would greatly benefit from an algorithm capable of attributing a value to NFTs, making it more fair for both parties. This would even enable automating loan requests, without the need for a lender to accept terms.

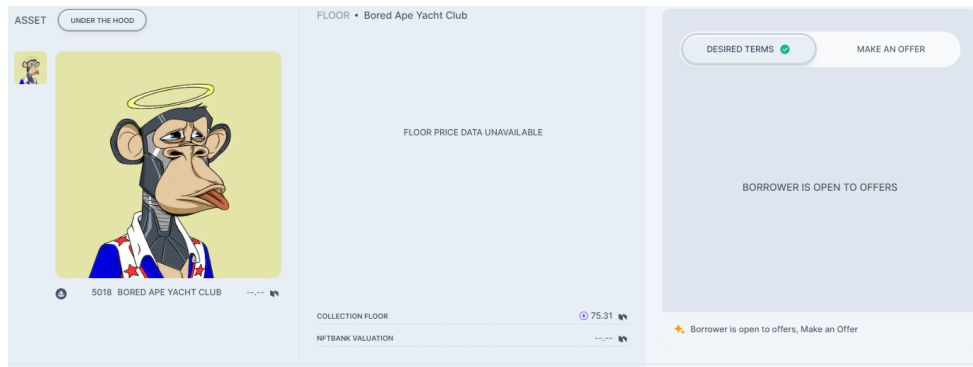


Figure 2.7: arcade.xyz Price Information

2.4.3 Upshot

Upshot⁶ is a blockchain-based protocol that performs NFT appraisals using Machine Learning models, repriced every hour. It provides an API to get near real-time prices for a considerable amount of NFTs, currently around 12,500 collections (32,000,000 NFTs). Besides pricing, it also provides other types of insights and analytics.

The pricing mechanism is not fully revealed, but some insights into the algorithm are shared through a series of blog posts ([Upshot.xyz, 2021a](https://upshot.xyz/blog/2021-01-15-upshot-protocol)). They "leverage machine learning models that ingest historical sales data and NFT metadata to construct features based on this information to generate accurate, reliable pricings" ([Upshot.xyz, 2021b](https://upshot.xyz/blog/2021-01-15-upshot-protocol)). This has proven to be more effective than the more basic techniques of using the last sales / floor price or using moving averages.

They also calculate error bounds around the predicted pricing to measure to which extent the prediction deviates from actual sales. These bounds capture both the model uncertainty and the market randomness. Through Shapley Values, Upshot is able to determine which variables in the model are important, things such as "rare traits" or "previous price" ([Upshot.xyz, 2021c](https://upshot.xyz/blog/2021-01-15-upshot-protocol)). This analysis helps understanding what are the main drivers for NFT pricing. There is also a "memory" component incorporated into the model, taking into account the time-dependent nature of markets ([Upshot.xyz, 2022](https://upshot.xyz/blog/2021-01-15-upshot-protocol)).

2.4.4 NFTBank

NFTBank⁷ is a portfolio manager service powered by Machine Learning models for NFT price appraisals, covering 1,800+ projects. They expose a public API that can be used to interact with

⁶upshot.xyz

⁷nftbank.ai

their resources.

The NFT pricing algorithm is based on a ML model that analyses NFT metadata traits and sales history, applying different methods to improve accuracy and prevent edge cases such as wash trading. It works better with collections with richer sales data and low volatility, so called "blue-chip".

Through a series of blog posts, they reveal some details about the inner workings of their algorithms. They identify that, due to the uniqueness of each project, the ideal solution would be to develop a model for each project that suits it. However, considering the growth of NFTs, this is not practical, and so it becomes necessary to explore a generic solution. They propose the implementation of a baseline model that can be applied to all NFT projects, even with reduced accuracy. Then, individual models can be created for specific collections over time to improve on the baseline solution (Ryker, 2021).

They tackle the problem of low volume through Trait Reconstruction (NFTBank.ai, 2022), by tracking and processing price information of "similar" items. This means finding the similarity between different NFTs of the same collection, such as the ones with overlapping traits. This involves determining which traits significantly impact price, and through the combination of individual weights we can reconstruct a price.

2.4.5 Zharta

Zharta⁸ is a real-time NFT loan platform for borrowers and lenders, meaning that users can use their NFTs as collateral for loans. In this app, borrowers can connect their wallet and pick the assets to collateralize. Then, their pricing algorithm will calculate how much they are valued and provide a quote in real-time.

Not much is revealed about the pricing mechanism, but it is revealed it is based on AI price discovery mechanisms (Zharta.io, 2022a). Additionally, in a Q&A (Zharta.io, 2022b), it is highlighted that the price analysis depends on the NFT, and that there is a focus on collectibles and profile picture (PFP) NFTs. The machine learning algorithm considers factors such as hype, trends, market sentiment, on-chain metadata and rarity.

Overall, it is still under development and not much is revealed about how it works. By going through the platform, we can see that only a few collections are supported, shown in Figure 2.8, indicating that the ML model is not generic, but that there is instead a specific model for each one.

2.4.6 Anchorage

Anchorage Digital⁹ prides itself on being a trailblazer platform with strong regulatory protections, proven security, and efficient, personalized support (Mónica and Mccauley, 2022). As such, there is a strong concern in every new feature, every new tool, every partner, to make sure regulatory compliance is maintained.

⁸zharta.io

⁹anchorage.com



Figure 2.8: Zharta's supported NFTs

This is the case for the Pricing Service, that determines the price of each crypto asset. In the case of cryptocurrencies, such as Bitcoin and Ether, Anchorage has partnered with 3rd party services that provide APIs to fetch the pricing information. These 3rd parties have been carefully reviewed and approved to make sure they are regulatory compliant and provide accurate and reliable data.

As for NFTs, the current solution relies on fixed pricing. For each token in custody, an expert will set a price manually in code and update it as needed. This valuation relies mostly on the floor price of the collection, and is dependant on the expertise of who performs the appraisal. This solution is far from ideal, as it requires manual updates and is not scalable. However, the NFT pricing APIs presented in the previous section are still in its infancy stage and their reliability is yet to be asserted, with the source code and inner workings being kept a secret. As such, there is no clear alternative for a better NFT pricing solution.

2.5 NFT Pricing Strategies

Having analysed existing research around price prediction topics, and given the context of NFTs, in this section we describe possible implementations of the NFT pricing algorithm, with different levels of complexity and trade offs. As discussed before, unlike stocks or cryptocurrencies, NFTs are not traded often and so don't have as much historical data. For this reason, it becomes necessary to explore other factors, such as the NFTs' metadata, to aid the decision process.

2.5.1 Last Sale Price

Using the last sale price is the simplest way of pricing an NFT. If you want to determine the value of a token, just use the value it was last traded at. However, many NFTs are selling for the first time and don't have any sales history, rendering this method impossible.

Additionally, the last sale might not actually reflect the true market value of the asset, since the last buyer might have priced it wrong. He might have been an inexperienced buyer, might have been particularly fond of the art of the NFT, etc, and so bought it for 3x the value. It is hard to distinguish market value from personal value.

Sometimes it is also hard to determine if the last sale of an NFT was an actual sale, or a transfer between wallets, sometimes even from the same owner, so the value we see might not be reliable.

This method also leaves the pricing algorithm open to manipulation, being very easy to control the valuation of an NFT simply by its last sale price.

2.5.2 Sales History

A step beyond relying only on the last price is to use the entire price history to drive the valuation. Considering an individual NFT, it is better because it helps averaging out outlier prices from different buyers and sellers' behaviours. This can be done with a Moving Average approach, that considers a window of X past sales and averages out the result. Even better, we can use an Exponentially Weighted Moving Average (EWMA), that gives bigger weights/importance to more recent sales, considering the time based nature of markets.

However, this still doesn't solve the problem of NFTs without any price history, and, even though harder, market manipulation is still possible.

2.5.3 Floor Price

Another approach to pricing is to use the floor price of the collection as the valuation for every token. This value represents the current lowest Offer price of a collection, the cheapest NFT for sale. Bids and offers depend on the marketplace, but considering that OpenSea controls the largest part of NFT trading volume, we could use the floor price from there. Alternatively, we could consider other marketplaces, such as X2Y2 and LooksRare, and use an average floor price, or the minimum of all the values for a conservative estimate. The floor price of each marketplace can be gathered from the APIs they provide.

As such, all of the NFTs of a collection would be valued at the same price. An NFT might be rarer or "prettier" than those in the floor, but this method provides a "conservative" estimate based on existing Offers.

It solves the problem of missing prices for individual NFTs, since we use the whole collections' floor, and it is also more robust against market manipulation.

However, this method assumes that the value of every NFT in the collection is the same, which is definitely not true. For example, as of 15th of January 2023, the floor price of CryptoPunks is 66.19 ETH. However, if you search for NFTs with the Alien trait, the floor price is a whopping 7,300 ETH, a 11029% difference.

2.5.4 Machine Learning

ML allows us to incorporate more data in the decision than what was possible before, such as the whole sales history of a collection to infer about each specific NFT, or using the NFT's metadata in the process. These factors are very important, as an NFT buyer would most certainly consider the tokens' art, rarity, owner, utility, and other characteristics, when he wants to make an investment.

By building a model trained on all of this data, the problem of having no sales history is solved. It can also be argued that such a method can better capture market behaviour, as it is trained over the whole sales history.

However, a Machine Learning algorithm takes more development work, as it requires data acquisition and preparation, figuring out which factors play out a big role in NFT pricing (or implement a way to infer them), tweaking parameters, interpreting results, etc. The resulting model, while hopefully robust and reliable, can also be unpredictable and sometimes hard to explain, unlike a simpler approach such as using floor price.

2.5.4.1 Token-level model

This ML model can be defined considering different scopes, each one with their own advantages and disadvantages. As each NFT is unique, it can be argued that they should be treated separately, just as it happens with stocks. The ideal solution would therefore be building a Machine Learning model for each individual NFT, trained and adapted to each particular combination of traits, meta-data, sales history, owners history, etc. However, considering the amount of data available for each NFT, that is often non-existent, it's easy to see this approach would not be possible. Even among the most famous collections, such as BAYC, the tokens with more sales data have at maximum around 50 sales. This is not nearly enough data to train a ML model.

2.5.4.2 Collection-level model

If we increase the scope a level, building a model that is trained to predict the prices of NFTs of an entire collection, the task becomes easier. We would be able to leverage the entire sales history of all tokens and, even if each token individually has only a few sales, in total we have a respectable amount of data, even if still not as rich as ideal. It makes sense to consider this scope, as the prices of NFTs in the same collection tend to move together, as if the collection has a "base price" and then each token adds some more value depending on their rarity or art. The model would be able to learn this collection base price and understand, from each NFT's metadata, what added value they have.

Among NFTs of the same collection, all of them (in most cases) share the same utility, have similar art, the creators and artists are the same, etc. Many of these subjective factors are shared, and don't need to be used as input to our model. In this case, we need only to find the distinguishing features between tokens of the same collection, such as their price history, ownership history, their metadata, among others. In this case, each collection would have their own Machine Learning model, that was trained and adapted to the specific circumstances of that collection. However, this approach sacrifices scalability, as for each new collection we want to calculate the price we need to train a new model.

2.5.4.3 Generic ML model

The way to solve this problem is to study a more general model that is capable of estimating the value of every NFT, regardless of its age, fame, if it's purely art or has some other utility, if it's a 1/1 or part of a collection, etc. The data concerns would definitely be solved, as we would be able to leverage Sales from a vast amount of collections and find relationships between them. However,

considering all of the differences between NFTs, along with the unpredictable, volatile, and often not understood factors behind the pricing of tokens, a "one-size-fits-all" model is very hard to accomplish. It would require coming up with features that identify the uniqueness of each project, so that the model can learn their differences, such as:

- The type of NFT, if it is a unique piece of art, a membership token, a part of a collection of similar tokens.
- Utility: is it a profile picture, does it unlock something in a videogame, does it give you access to exclusive websites, clubs or parties, does it provide monthly rewards.
- Quality of the associated media type, such as art or sound. This is very subjective and depends on each person's personal taste, the style of art, the colors.
- Social media activity: number of followers, frequency of posts, type of language and words used, engagement.
- Artist or creator reputation: are they famous or anonymous? are they respected or irrelevant?
- The timing of the project in the market.
- The philosophy of the project and its founders .

But how would you quantify something you don't understand, or subjective factors such as the utility of a token, or the quality of the art? Let's consider the case of BAYC: their art is often considered not to be very high-quality; their utility has grown from being just a profile picture to giving access to exclusive communities and events, but is still arguably less impressive than many other projects. So why is it valued so highly? A possible explanation would be due to the timing of the project, as well as the founders' philosophy, and how they grew and kept building this brand and community that is now widely known and respected. But how would you make your ML model understand this?

One could argue some of these factors, especially the more subjective ones, are already priced in the current market value, and that considering the price history and objective characteristics would be enough. We could even limit our scope to blue-chip collections, removing a lot of uncertainty and volatility in the data to help the model learn. If successful, this would be the "ideal" solution, as it is scalable and more desirable from a business standpoint. However, it doesn't change the fact that testing this approach would require collecting a vast amount of representative data from different NFTs, and coming up with meaningful features that could identify each project and their value.

2.5.4.4 Machine Learning Techniques

Building on top of previous research, we have seen that there is a general consensus for techniques applied to time-series price prediction problems, such as the one we are targeting in this paper. The

most common approaches rely either on statistical analysis models, such as ARIMA (2.2.2), or in deep learning models, with a tendency towards RNNs and, more specifically, LSTMs (2.2.3.3). However, unlike the previously identified problems, this one presents itself as an irregular time series, where the data samples are not separated on regular intervals, and are instead dictated by sales that can happen at any time of the day. Weerakody et al. (2021) studies irregular time series data handling and argues that this characteristic makes statistical models less applicable, with a preference for RNNs models.

When considering visual features of an art work or NFT, the ideal solution is to use a Convolutional Neural Network (CNN) to extract relevant features that can be used in the following steps. However, both in the case of art and NFTs, there is a strong indication that non-visual features are more relevant than visual ones, such as information related to the artist and its fame. Data about social media presence and activity has also been found to influence the pricing of assets, with many attempts made at considering sentiment analysis from Twitter or Google Trends in the model.

Given these insights, in this work we will focus our effort

Chapter 3

Methodology

3.1 Problem Formulation

3.1.1 Time Series

Due to the time-dependent structure of the data, the sales of NFTs, one would be inclined to model this problem as a typical time-series. This has been the case for the prediction of prices for stocks and cryptocurrencies, but the problem we are targeting is not as simple. While the price prediction of a single NFT token can be perfectly modeled as a time-series problem, only considering that token's past sales, when we increase the scope to a whole collection, with multiple tokens, we need to introduce non-time dependent factors to distinguish between them, such as token ID or traits information. Instead of a single "price" for each day, we have sparse data, at irregular intervals, from the sales of each NFT. Some days have zero sales, some days have multiple ones for multiple tokens. For this reason, it becomes harder to apply traditional statistical time-series models, such as ARIMA, that require data points to have a continuous fixed granularity, such as daily or every 1 hour (VERMA, 2022). However, the data is still sequential and time based, so it can benefit from time series Deep Learning models that can handle more complex sequential data.

3.1.2 Multiple Regression

The present problem is best formulated as a Multiple Regression, where we have a set of independent variables that we use to predict a single dependent variable. In this case, we want to use the sales history of NFT collections, together with general information about them and each individual NFT, to predict the current value of tokens. Some of the input variables we will use are time dependent, but the collections and tokens' information is static and so it may make more sense to consider a non-time series approach.

3.1.3 Assumptions

In this work we are considering that an NFT's value corresponds to the value that it is sold for. We believe this assumption to make sense, as it implies that both a buyer and a seller agreed on a

price, the buyer was willing to pay the agreed amount and the seller was willing to letting go of the token for that same value.

We are focusing on blue-chip collections in the Ethereum blockchain, as these have stabler and more rich sales history, that will provide us more meaningful data to train our models.

We have opted for targeting non-visual features of NFTs, as these were found to be more relevant to determine the price.

3.2 Dataset

This section covers our strategy with respect to the data we used for this project, going over what data we needed, where we could find it, and how to prepare it for our ML models.

3.2.1 Data Gathering

Most of the data we needed to fetch related to NFTs, their collections, and past transactions, is recorded on the Ethereum blockchain. Metadata is usually stored off-chain, but a link pointing towards it is on-chain. While this information is publicly available to anyone and could be extracted directly from it, this method is very non-efficient. Unlike traditional databases, such as SQL, Ethereum does not provide a query language to fetch exactly the data we need. The way to solve this task would be to go through each block on the blockchain in sequence, checking if each one has data related to our use case. However, considering the enormous amount of data being added to the chain everyday, and that we would need to go back years in the past, this method would be very costly and unfeasible.

The alternative is to use centralized services that index this NFT data over time and provide an accessible and queryable API, giving us the information we need in a much easier manner. These services include NFT marketplaces such as LooksRare and OpenSea that, since their founding, have been collecting this data. Both of them provide a free to use API that is capable of giving us all the blockchain information we need. OpenSea is the clear preference, as it is the most established NFT marketplace, and would be the ideal choice. However, for some endpoints, it requires an API key, and currently this can only be acquired by filling a form and waiting for an approval that can take a long time.

We started our data gathering process with the [LooksRare API](https://looksrare.dev/reference/)¹, as it was free to use and easy to get started, with no API key required. Through its endpoints, we were able to fetch the following data:

- **/collections:** generic information about a collection, such as its name, symbol, owner and social media links.
- **/collections/stats:** statistics related to a collection, such as the current floor price, the total supply, the number of owners.

¹looksrare.dev/reference/

- **/tokens:** information about specific NFT tokens and its metadata. This includes detailed data about each token's traits and the number of times they occur.
- **/events:** activity related to a token, such as Listings, Transfers, Mints, Sales, etc. Through its filters we were able to fetch the whole Sales history for each token.

LooksRare has a rate limit of 120 requests per minute. Considering most collections have around 10,000 tokens, if we want to make an API call for each token separately it will take, in theory, around 84 minutes (10,000 / 120) to go through each token in a collection. However, since the endpoints for token data and Sales are different, and each request has a limited number of Sales that it can return at a time, this means we actually need to do a lot more calls. Depending on the collection and amount of events to fetch, our data collection was taking from 2 hours to 4 hours. In our implementation, we made sure our script would sleep for some time whenever it exceeded the rate limit, making sure it could continue later.

However, after gathering the data and starting to visualize it, we noticed that the earliest Sales we had were from January 2022. This is a clear problem, as we would only have a couple months worth of data to train our models and it would likely not be enough. After some investigation we realized that the LooksRare platform only started in January 2022, meaning that they only started indexing data from then. Through their API we would not be able to get historic data prior to that, missing out on the NFT boom that happened in 2021.

As such, it became imperative to consider [OpenSea's API](#)². Their platform has been around since 2017 and has all of the available data from the blockchain about the NFT projects we are targeting. Luckily, after some time we were able to acquire an API key and use it. Similarly to LooksRare, OpenSea provides the following endpoints to fetch data:

- **/assets:** information about specific NFT tokens and its metadata. This includes detailed data about each token's traits and the number of times they occur.
- **/events:** activity related to a token, such as Listings, Transfers, Mints, Sales, etc. Through its filters we were able to fetch the whole Sales history for each token.

Instead of looping through each token individually, OpenSea's API provides a way of fetching data in 'pages', grouping information from different tokens in the same response and reducing the total number of requests required. As such, we updated our script to fetch both the NFTs info and the Sales data from OpenSea, making the whole process more efficient and now being able to gather a larger amount of historical data.

We decided to continue to use LooksRare's API for gathering generic information about collections and statistics. This is because OpenSea has strict policies around NFTs and wallets that were hacked or compromised, and we noticed it sometimes hides them from the results, such as the total number of tokens. As such, we collect this information first from LooksRare, and then gather the sales and individual data from OpenSea.

²docs.opensea.io/reference/api-overview

3.2.2 Gathered Data

For each NFT collection, we started by gathering general collection details, including the name of the collection, symbol, address of the owner, floor price, total supply and count of owners.

We then collected a full list of traits, storing all of the possible values and the number of occurrences of each, to allow us to determine which appear more or less frequently. For each NFT, we kept a list of all of the traits that it was composed of. And, finally, we gathered the full list of transactions, including the timestamp, the token id, and the sales price in both ETH and USD.

We also gathered data about the price of Ether over time, with the intent of using it as a feature that will help the model understand the overall crypto market state, as there have been indications there is a correlation with the NFT market.

3.2.3 Data Augmentation

As there was a concern that the amount of available data for each collection wouldn't be enough to train the data-hungry Machine Learning models, especially Deep Learning ones, some thought was given to Data Augmentation. The idea here would be to artificially create more data points that would help the models train and become more reliable. Such a technique would have to be studied very carefully, making sure that the new data that was created made sense according to some realistic factor.

A possible approach to this would be to create sales based on tokens' rarity. For each sale that happened, we could check the token's rarity score and then create new sales for all tokens with similar rarity, accepting some threshold. This augmentation method is assuming that tokens with similar rarity are priced at the same value, that in most cases seems to prove true.

However, this approach was not implemented in the end. These concerns mainly came up when we only had the LooksRare API to use and the data was very limited, once we got access to OpenSea the data we gathered proved to be enough.

3.2.4 Feature Engineering

In order to better represent the problem at hand and help the models learn, we made an effort at feature engineering, by selecting and transforming the raw data we gathered to create new features. Below we detail them, explaining why they are important and how we accomplished them.

NFT sales can happen individually or in **bundles**, where multiple tokens are sold in one transaction. In the first case it is straightforward to know how much a token was valued at, its sale value. However, in the case of bundles there is no way to understand how much each token was priced at individually, as we only have the collective price of the bundle for 2 or more tokens. As such, considering that we already have a scarce amount of data and want to maximize it as much as possible, in the case of bundles we are averaging the price and considering that each NFT in that bundle was sold for the same amount.

Considering that sales can happen in **multiple cryptocurrencies**, such as Ether (ETH), Wrapped Ether (wETH), DAI, USDC, or even APE, we needed to find a way to convert all of these values to the same scale, and as such decided to use US dollars. However, taking into account that the US dollar value of each of these coins changes over time, we had to find a way to incorporate their historical prices at each sales timestamp in the calculation. For this purpose we used investing.com³, a financial markets platform that allows users to export historical prices for each cryptocurrency in a CSV format. For each sale, we check the currency, the timestamp, and fetch from this exported data the corresponding historical price.

To tackle the problem of identifying how unique a certain NFT is in a collection, we introduced in our model a calculation of **Rarity**. This is a frequently mentioned and important concept among investors, as it has been proven that rarer tokens fetch higher prices and are more stable (Mekacher et al., 2022). However, NFTs do not possess a 'rarity' trait, this is a concept that needs be calculated from the frequency and importance of all the other traits it is made of. Multiple methods have been proposed to achieve this, such as Rarity Ranking, Average Trait Rarity, Statistical Rarity, and Rarity score (Khritonenkova, 2022). All of them work in different ways, but Rarity Score provides the most objective valuation since it considers the rarities of all traits, and we ended up using this one. It works by calculating the *RarityScore* (3.1) for each attribute of an NFT and then summing up all of the values. This will give us the final Rarity of an NFT, where an higher value corresponds to a rarer NFT.

To account for market trends and patterns, we introduced **moving average** (MA) features in our model. MA is a technical indicator that is often used by investors and traders to determine the trend direction of securities. By considering a window of past data and averaging out the price, it helps in smoothing out the price, diminishing the impacts of random, short-term volatility. Simpler Moving Averages use a simple arithmetic average of the prices, but we decided to use Exponential Moving Average, giving more importance to recent prices rather than older ones. Since in our case we have multiple tokens with multiple sales per day, we considered the MA of the whole collection, by taking, for each day, the average of the sales price, and building a MA over those values. We considered windows of 3 sizes, 3-days, 7-days and 30-days. Likewise, we also built features considering the Standard Deviation (STD) of the sales price, for the same 3 windows.

We also introduced 2 extra variables in the data, for each data point of sales: the **Last Token Price** and the **Last Collection Price**. The first variable keeps track of the previous sale price of the NFT that was sold, the second keeps track of the last sale price of any NFT in the same collection as the token that was sold.

$$RarityScore = CollectionSize / NTokensWithTrait \quad (3.1)$$

³investing.com/crypto/currencies

3.3 Evaluation

This section goes over our reasoning behind evaluating the models we are going to train and analysing their performance.

3.3.1 Approach

Our testing method will rely on using Back Testing, by making use of historical data to train and test our model. We start by splitting the data into training and testing, keeping the chronological order of sales, and then analyse how accurate the predictions of the model trained using the 1st part are for the 2nd part of the data. For example, if we have 1 year of sales data, we can use the 11 first months to train. Then, for the remaining 1 month, for each actual sales price, we will pick the date and the NFT and make our model predict its value. We can then compare this prediction with the actual sale and check how accurate it was.

Another approach would be using Forward testing, also named Paper Trading, where the model is placed in a real-time environment making predictions, comparing these with the actual transaction values of NFTs that are happening. This is relevant because it means putting the model to work in a real scenario, as a normal user would do. However, its evaluation speed is constrained by the speed of the blockchain transactions, and would take a long time by itself.

For this reason, Back Testing comes as a way to still use real data in a faster and more available manner. We only need to gather the historic data and use it to train the model, without having to wait for new transactions to finish.

3.3.2 Metrics

This problem comes down to a regression problem and we want to measure how far our price predictions are from the real world. To accomplish this, we will use both the Mean Absolute Percentage Error (MAPE), defined in Formula 3.2, and the Root Mean Squared Error (RMSE), defined in Formula 3.3, where N is the number of timestamps, A_t is the actual value, F_t is the forecast value.

MAPE measures the relative error between actual and predicted values, as a percentage. RMSE considers the difference between actual and predicted price values for all timestamps and gets an absolute measure of error.

Our Deep Learning models used Mean Squared Error (MSE) as the loss function, calculated as defined in Formula 3.4. We implemented an Early Stopping mechanism based on the loss of the valuation dataset, so that the models stop training if this value doesn't improve over 3 epochs.

The implementation of these metrics and loss function can be seen in Listing 3.1.

$$MAPE = \frac{1}{N} * \sum_{t=1}^N \left| \frac{A_t - F_t}{A_t} \right| \quad (3.2)$$

$$RMSE = \sqrt{\frac{1}{N} * \sum_{t=1}^N (At - Ft)^2} \quad (3.3)$$

$$MSE = \frac{1}{N} * \sum_{t=1}^N (At - Ft)^2 \quad (3.4)$$

```

1 early_stopping = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=3,
2 mode='min')
3 model.compile(loss=MeanSquaredError(),
4               optimizer=Adam(),
5               metrics=[RootMeanSquaredError(), mean_absolute_percentage_error])

```

Listing 3.1: Models' metrics

3.4 Proposed Approach

We decided to tackle this problem with a focus on collection level, by designing models that are specific for each NFT collection. We believe this to be a good compromise between scalability and applicability of the models for NFT prediction, allowing us to test their effectiveness and infer next steps and improvements.

As mentioned, this problem can be seen as a Multiple Regression with a time-series component. For this reason, we explored models of increased complexity, starting by taking all of the inputs as being independent, and then proceeding to include a memory component in our model to learn from the sequential data.

3.4.1 Tech Stack

We chose to use **Python**⁴ for development, for its simplicity and ease of use. Our work will involve a lot of data operations and Machine Learning algorithms, and Python has many ready-to-use packages for these tasks. Using it will allow us to fetch data, prepare and save it, visualize it through charts, and train Machine Learning models, all in the same coding language.

As our development environment we chose to use **Google Colab**⁵, as it allows us to run our code in the cloud at any time and keep our scripts organized. With it, we can easily make use of Google Drive space to store and import datasets between our data fetching and ML models' scripts.

We used **TensorFlow**⁶ as our machine learning platform, a free and open-source software library, originally developed by Google, with easy integrations with Python. More specifically, we used **Keras**⁷, a deep learning API built on top of TensorFlow. To create visualizations and plots,

⁴python.org

⁵research.google.com/colaboratory/faq.html

⁶tensorflow.org

⁷keras.io

we went with [Matplotlib](#)⁸. We also made extensive use of [Pandas](#)⁹ for data manipulation and analysis.

3.4.2 Data Preparation

We split our data into training, validation, and test sets, using a 70, 20, 10 split. The training data is the largest and will be used to fit the model and learn the best parameters; the validation data is used for hyper-parameter tuning and selecting the best model under those under consideration, still having an impact on the outcome of the model; the test data is used after we have chosen the final model to evaluate its performance and ensure it is general enough for new data points.

We decided to keep the data ordered by the date of the sales, keeping its sequenced nature of consecutive samples. This makes it so that the validation and test valuations are more close to reality, by using data that was collected after the model has finished training.

Before training the neural networks, we scale the features through normalization. This means we subtract the mean and divide by the standard deviation of each feature. These values are calculated using the training data, to ensure that the models do not have any access to validation and test sets. The goal of this is to change the values of different features to a common scale, without distorting the differences in ranges of the values.

Additionally, to take advantage of the sequenced nature of the data, some of the models we develop will take the input data in the form of [windows of consecutive samples](#)¹⁰. This means the model will ingest multiple time steps at a time for the prediction of a single output. For example, a model that would make predictions given the past six transactions can be seen in Figure 3.1.

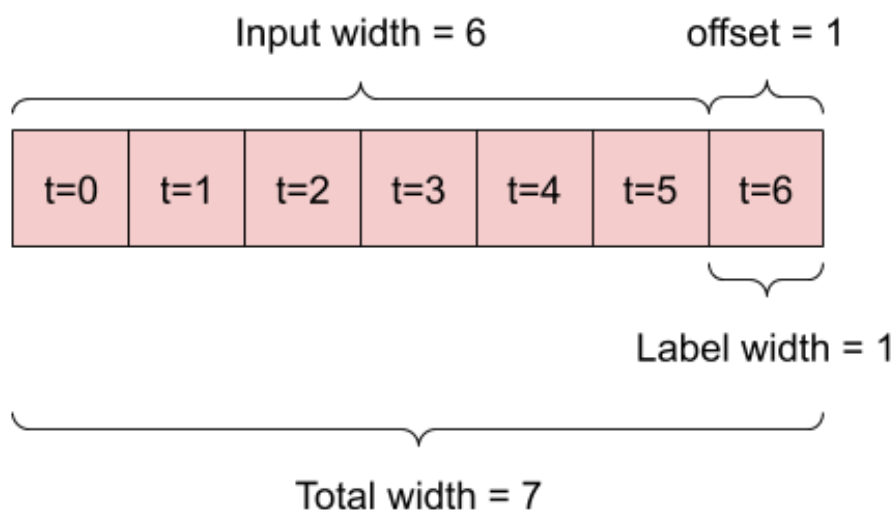


Figure 3.1: Example Data Window

⁸matplotlib.org

⁹pandas.pydata.org

¹⁰tensorflow.org/tutorials/structured_data/time_series

3.4.3 Models

Here we discuss the models we used to tackle this problem on a high-level view. It is worth noting that the parameters used in this section are examples, as there was effort put into searching for the best set of parameters and configurations, as discussed in Chapter 4.

3.4.3.1 Baseline

We started by creating a simple model to use as a performance baseline to compare with the following, more complex models. Since we want to predict the price of NFTs given the sales history, we started by developing a model that simply returns the previous sale of any NFT of the same collection as output, as if the value hasn't changed. This model does not require any training and should be a good baseline estimate, as it is intuitive that sequential sales should not be much different from each other.

3.4.3.2 Linear model

We then considered this problem as Multiple Linear Regression, by training linear transformation models that try to find a relationship of the type $y = mx + b$ between the input data and the output, where m is a matrix. These models do not yet take into account the sequential nature of the data and take time steps as independent (3.2).

```
1 linear = Sequential([  
2     Dense(units=1)  
3 ])
```

Listing 3.2: Linear model

3.4.3.3 Dense model

Increasing the complexity of the approach, while still ingesting single input steps, we experimented with deeper models, by stacking a few Dense layers (3.3).

The workings of this model and the previous Linear model are displayed in Figure 3.2. In both cases, the models take a single sample of the data and feed it into the trainable model to make a prediction, taking samples independently from each other.

```
1 dense = Sequential([  
2     Dense(units=64, activation='relu'),  
3     Dense(units=64, activation='relu'),  
4     Dense(units=1)  
5 ])
```

Listing 3.3: Single-input Dense model

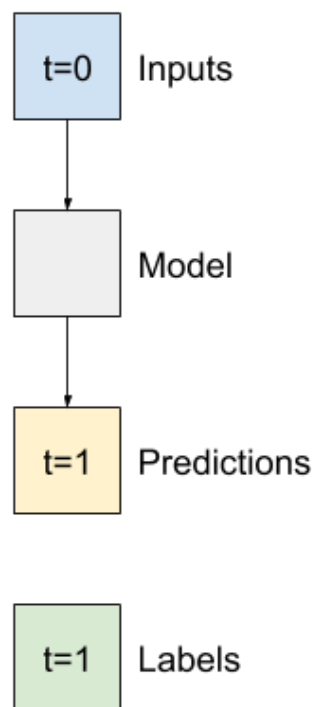


Figure 3.2: Single Time Step Window

3.4.3.4 Multi-step Dense model

To take advantage of the continuous nature of the inputs, we then explored feeding the model with multiple time steps at a time to make the predictions. This allows the model to gain context of the previous inputs and use them to make better decisions. In this case we will be giving the model a window of multiple samples, and, as such, need to introduce a Flatten layer in the model to collapse the window into a single array (3.4).

In Figure 3.3 we can see how a model that takes windows of size 3 would work, concatenating all of the samples and feeding it into the trainable model to make a prediction.

```

1 multi_step_dense = Sequential([
2     # Shape: (time, features) => (time*features)
3     Flatten(),
4     Dense(units=32, activation='relu'),
5     Dense(units=32, activation='relu'),
6     Dense(units=1),
7     # Add back the time dimension.
8     # Shape: (outputs) => (1, outputs)
9     Reshape([1, -1]),
10 ])
  
```

Listing 3.4: Multi-step Dense model

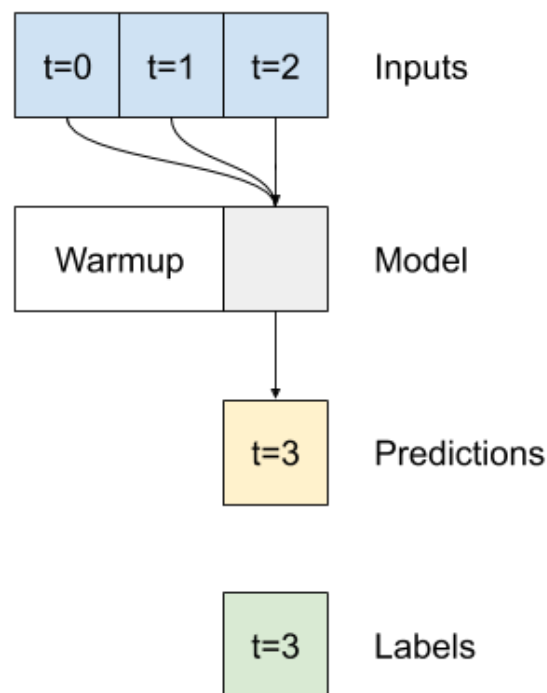


Figure 3.3: Multi-step Dense Model

3.4.3.5 Convolutional Neural Network

We also tested the efficacy of Convolutional Neural Networks to this problem, as defined in Listing 3.5.

Figure 3.4 displays the workings of such a model, that applies the convolutional operation to a sliding window of inputs. In this case the kernel is of size 3 and the model learns 3 time steps at a time.

```

1 conv_model = Sequential([
2     Conv1D(filters=32,
3         kernel_size=(3,),
4         activation='relu'),
5     Dense(units=32, activation='relu'),
6     Dense(units=1),
7 ])
  
```

Listing 3.5: Convolutional Neural Network

3.4.3.6 Recurrent Neural Networks

RNNs are the most well-suited models to time series data, as they process each sample step by step and keep an internal state between timesteps. This allows the model to learn useful insights over time, as if it had a memory component. We experimented mostly with LSTMs, such as in the examples in Listing 3.6. These have a *return_sequences* parameter that controls whether the

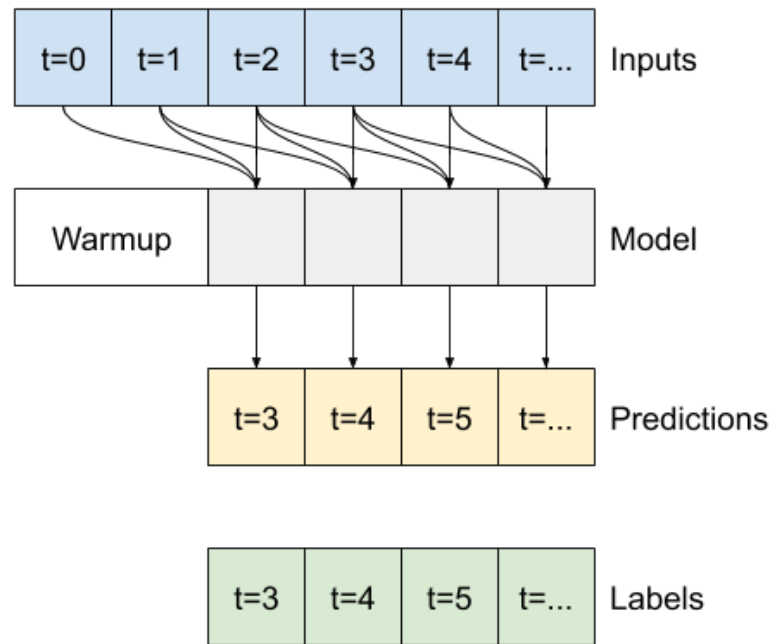


Figure 3.4: CNN Data Window

model should return output for each input time step, or only for the last one after going through the whole window of data.

Figure 3.5 helps in visualizing how the model trains when it takes as input a window of 24 samples of data, when *return_sequences* is set to False. It takes all of the samples through its internal state and arrives at a single prediction in the end.

```

1 lstm_model = Sequential([
2     LSTM(32, activation='relu'),
3     Dense(units=1)
4 ])

```

Listing 3.6: Convolutional Neural Network

3.4.4 Features

As discussed before in the Feature Engineering step, we created some additional variables from the raw data with the intent of making the model learn. The final set of features we considered in our dataset can be seen in the data samples presented in Figures 3.6 and 3.7. The meaning of each feature is explained in Table 3.1.

As we trained our models, we experimented with different combinations of these features to find the most optimal prediction model.

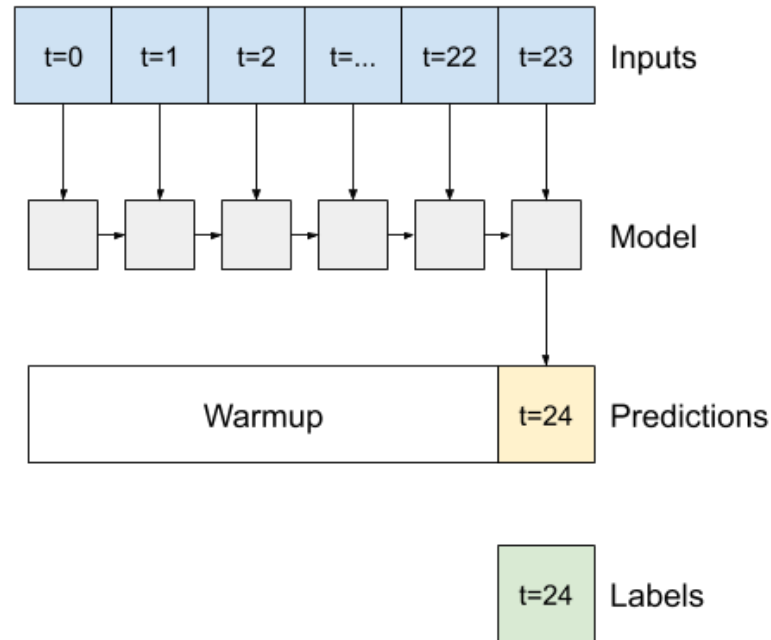


Figure 3.5: LSTM Data Window

	collection	token_id	rarity	timestamp	price	last_token_price	last_collection_price
27020	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	2032	339.061461	1671527111	7152.99	7165.09	7140.89
27021	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	15783	278.269060	1671530375	7152.99	7165.09	7152.99
20004	0x32973908faee0bf825a343000fe412ebe56f802a	3784	92.321660	1671531071	422.40	0.00	847.73
9742	0x68cd21d362c2dae66909afd810e391fa152b2379	3784	1679.921600	1671531071	422.40	0.00	847.73
27022	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	3178	571.695786	1671532091	7805.35	0.00	7152.99
9743	0x68cd21d362c2dae66909afd810e391fa152b2379	2115	640.686068	1671532439	375.20	387.30	375.20
20005	0x32973908faee0bf825a343000fe412ebe56f802a	8521	158.330533	1671532439	375.20	387.30	422.40
20006	0x32973908faee0bf825a343000fe412ebe56f802a	2115	234.349845	1671532439	375.20	387.30	375.20
27023	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	13954	452.680324	1671533819	7152.99	6890.02	7805.35
27024	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	1575	804.835107	1671534539	7152.99	7165.09	7152.99
27025	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	18685	134.524465	1671534635	7165.09	7165.09	7152.99
27026	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	2427	240.900111	1671535427	7152.99	7165.09	7165.09
27029	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	3571	662.102561	1671535523	7152.99	7165.09	7152.99
27028	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	18343	230.286314	1671535523	7152.99	7165.09	7165.09
27027	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	18685	134.524465	1671535523	7165.09	7165.09	7152.99
27030	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	17832	292.580852	1671535823	7152.99	7165.09	7152.99
27031	0x49cF6f5d44E70224e2E23fDcdd2C053F30aDA28B	14826	348.800235	1671536051	7152.99	7165.09	7152.99

Figure 3.6: Input Features Dataframe - Part 1

	eth_price	ewma_3	std_3	ewma_7	std_7	ewma_30	std_30
27020	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27021	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
20004	1217.20	446.617717	66.279699	494.494225	125.695371	595.895789	219.727226
9742	1217.20	447.020629	49.019232	484.358427	118.723556	584.914758	224.239523
27022	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
9743	1217.20	447.020629	49.019232	484.358427	118.723556	584.914758	224.239523
20005	1217.20	446.617717	66.279699	494.494225	125.695371	595.895789	219.727226
20006	1217.20	446.617717	66.279699	494.494225	125.695371	595.895789	219.727226
27023	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27024	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27025	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27026	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27029	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27028	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27027	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27030	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539
27031	1217.20	7259.560363	263.522903	7599.985746	863.963581	9360.055660	1959.512539

Figure 3.7: Input Features Dataframe - Part 2

collection	The collection that the sold NFT belongs to
token_id	ID of the token that was sold
rarity	Rarity Score of the token that was sold
timestamp	Unix timestamp of when the sale happened
price	Value in USD the NFT was sold for
last_token_price	Last sales value for the same token, in USD
last_collection_price	Last sales value for any token of the same collection, in USD
eth_price	Price of Ether in USD at given timestamp
ewma_3	Exponential MA window of the avg collection price over past 3 days
std_3	Standard Deviation of the avg collection price over past 3 days
ewma_7	Exponential MA window of the avg collection price over past 7 days
std_7	Standard Deviation of the avg collection price over past 7 days
ewma_30	Exponential MA window of the avg collection price over past 30 days
std_30	Standard Deviation of the avg collection price over past 30 days

Table 3.1: Dataset Features Explanation

Chapter 4

Results

4.1 Data Visualization

In this section we take a look at the input data that we fed into our ML models, visualizing it through tables and charts, to better understand what we are working with.

As mentioned before, we collected Sales data related to multiple Blue-Chip NFT collections. These sales can be seen in Figure 4.1, displaying the whole sales history, in US dollars, for BAYC, CryptoPunks, Azuki, MAYC, Doodles and World of Women. For visualization purposes, we averaged the sales of each collection per day, as a collection might have many sales on the same day and this would clutter the chart. Additionally, we removed outlier sales both on the lower and higher bounds. These outliers are discussed in further detail below, for BAYC, as seen on Figure 4.5. The same data can be seen on a logarithmic scale on Figure 4.2. We see that most collections started appearing in 2021, with the sole exception of CryptoPunks, having originated much earlier in 2017. In Figure 4.3 we can take a closer look at the sales history starting in 2021, and the same data is shown on a logarithmic scale in Figure 4.4. We see that the Punks' price was more or less stable from 2017 to 2020, but in 2021 there was an NFT boom and prices rose quickly to the high valuations of 6, sometimes 7 digits. These charts shed some light over the enormous volatility of the NFT market, with prices changing immensely upwards and downwards just in the span of 1 year. The appearance of the other collections coincided with this boom, and we see that they too quickly rose in valuation. Overall, we notice a trend between different collections' prices, where they seem to rise and fall in sync, meaning that when one collection drops the others drop as well. This may be an indicator of external factors that are affecting the whole NFT market. For our purposes, this characteristic may prove itself useful, as we can infer patterns between different collections, leveraging the sales history of all of these collections on the prediction of individual ones.

Taking a closer look at a single collection, Bored Ape Yacht Club, we can see how disperse the sales are. Figure 4.5 displays all the sales related to BAYC, while Figure 4.6 displays the same data but removing outlier sales in the upper and lower bounds, for visibility purposes. Even for the same day, there are multiple sales with very different price points, often double or triple from each

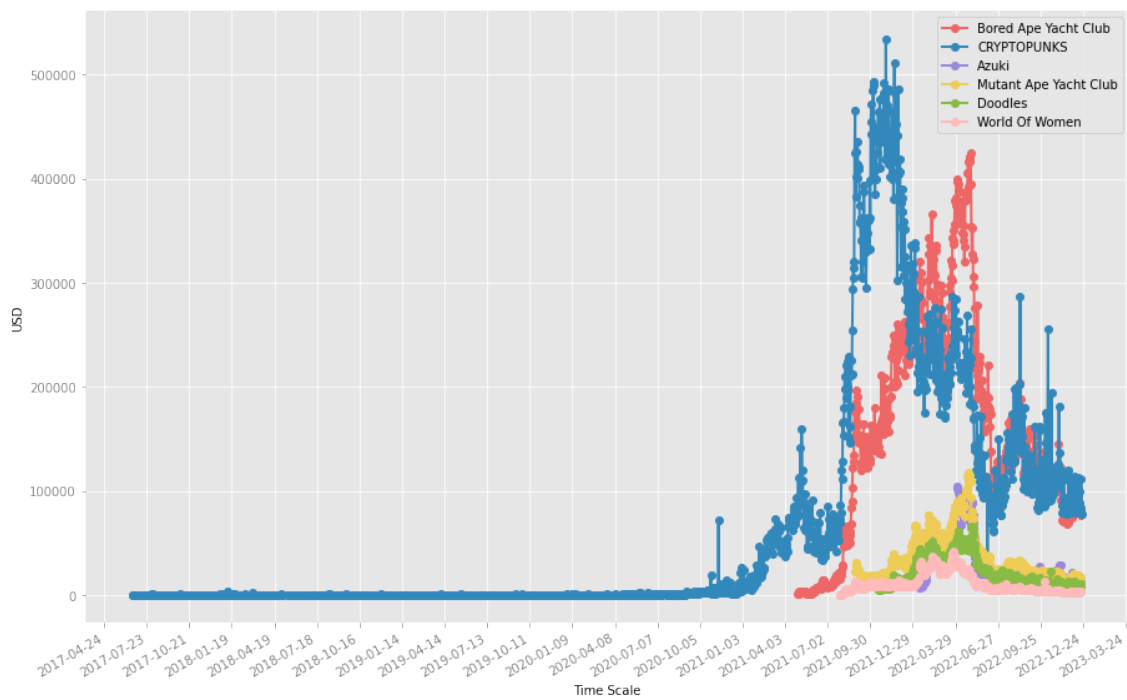


Figure 4.1: Sales of Blue-Chip Collections

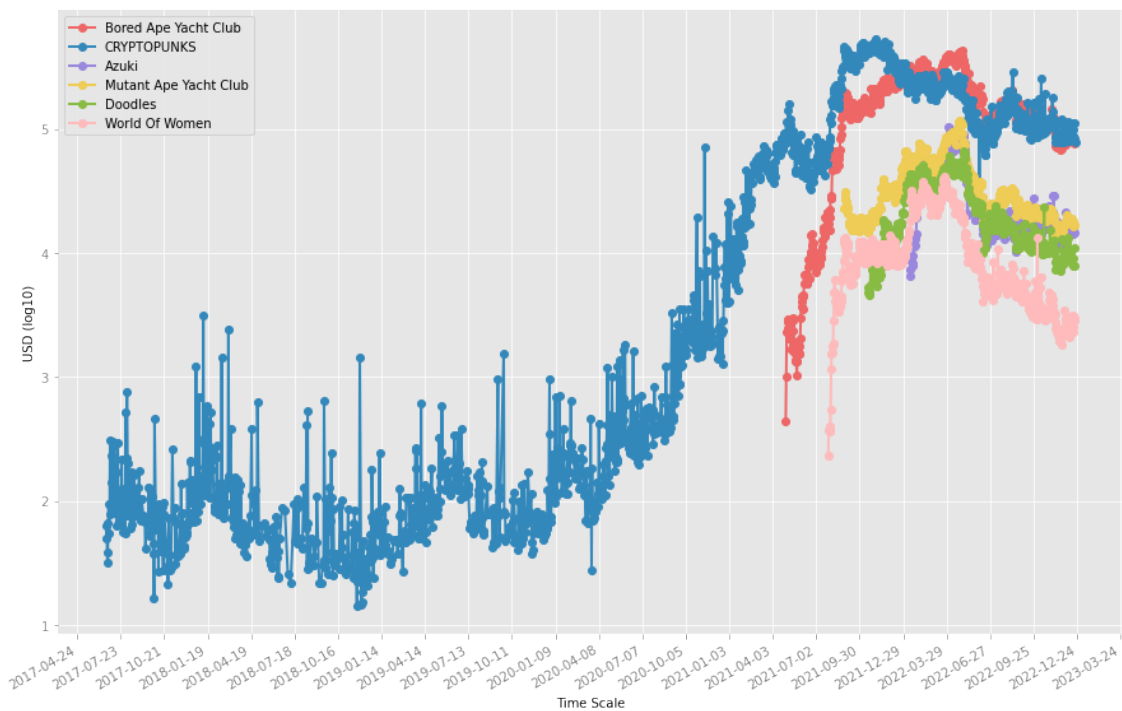


Figure 4.2: Sales of Blue-Chip Collections (log10)

other. These outlier sales with high prices can be justified by the uniqueness, rarity, or personal preference of the underlying token, making it more valuable to the buyer than the others. The outlier sales on the lower bound are often credited to hacked wallets, where the attackers manage

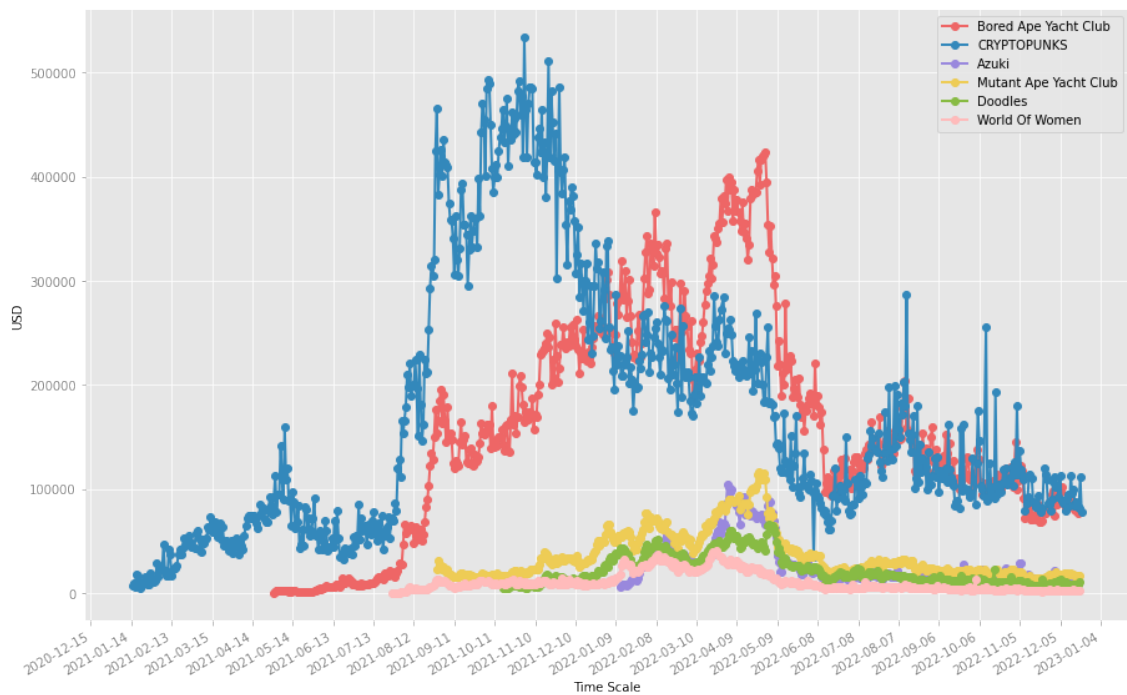


Figure 4.3: Sales of Blue-Chip Collections between 14/01/21 and 20/12/22

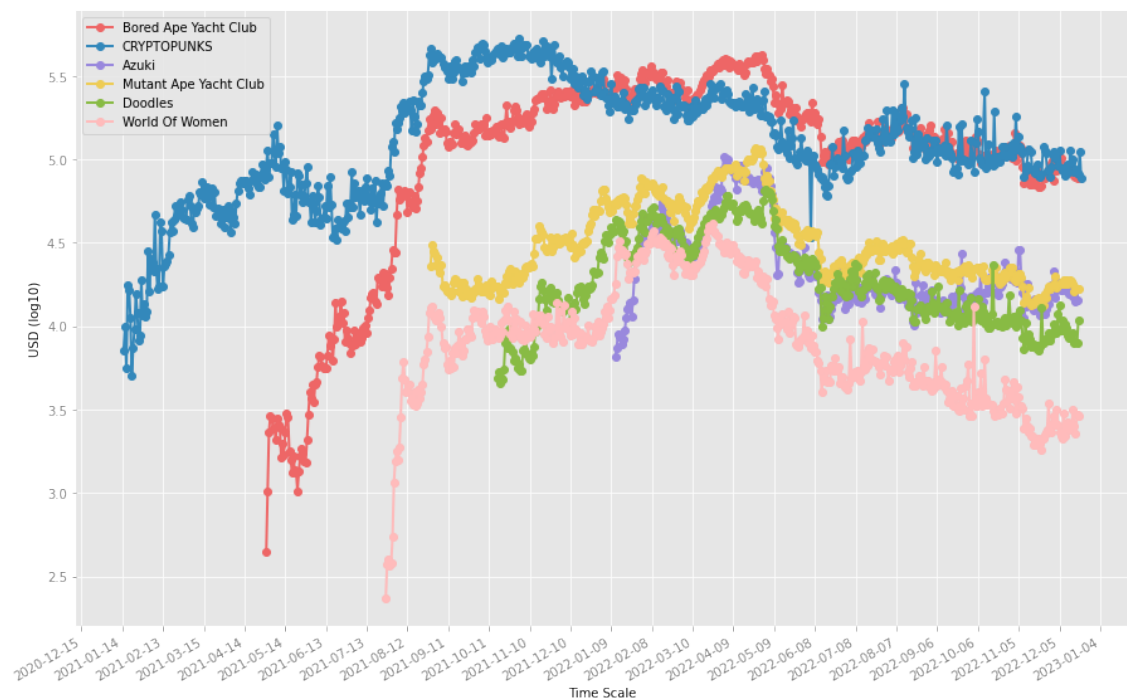


Figure 4.4: Sales of Blue-Chip Collections between 14/01/21 and 20/12/22 (log10)

to sell those tokens to themselves at low or zero prices. We can clearly see the problem of NFT valuation here, from how volatile the prices are and how different they can be in the same day between different tokens.

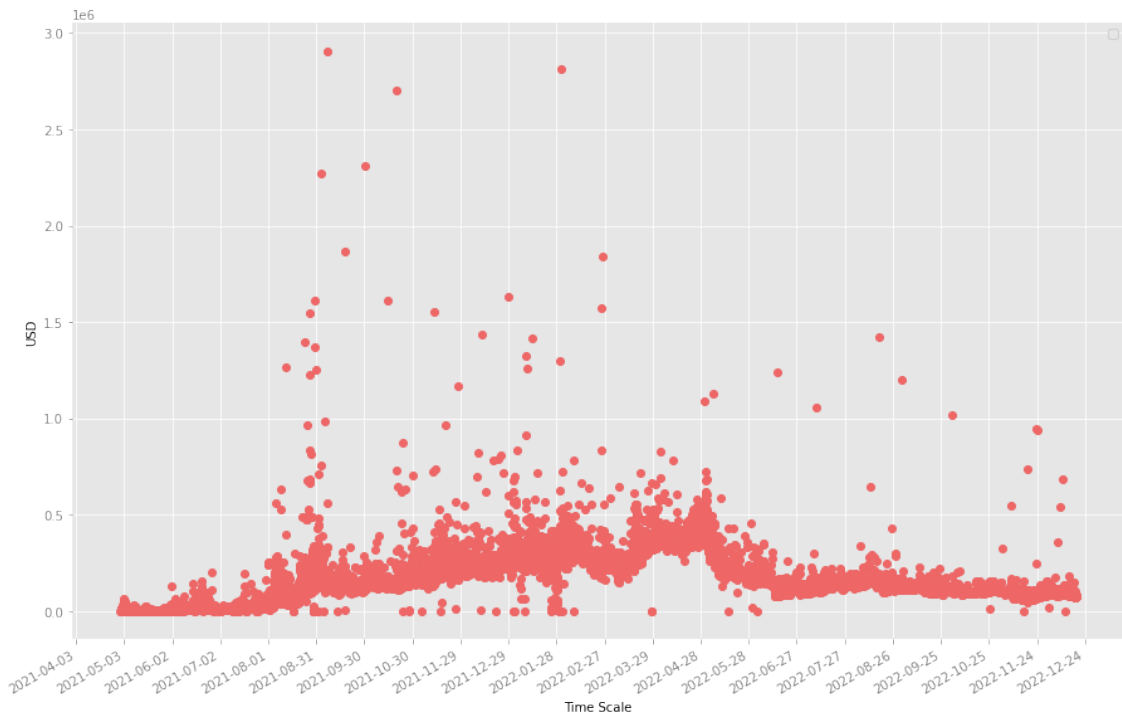


Figure 4.5: BAYC - Total Sales without removing outliers

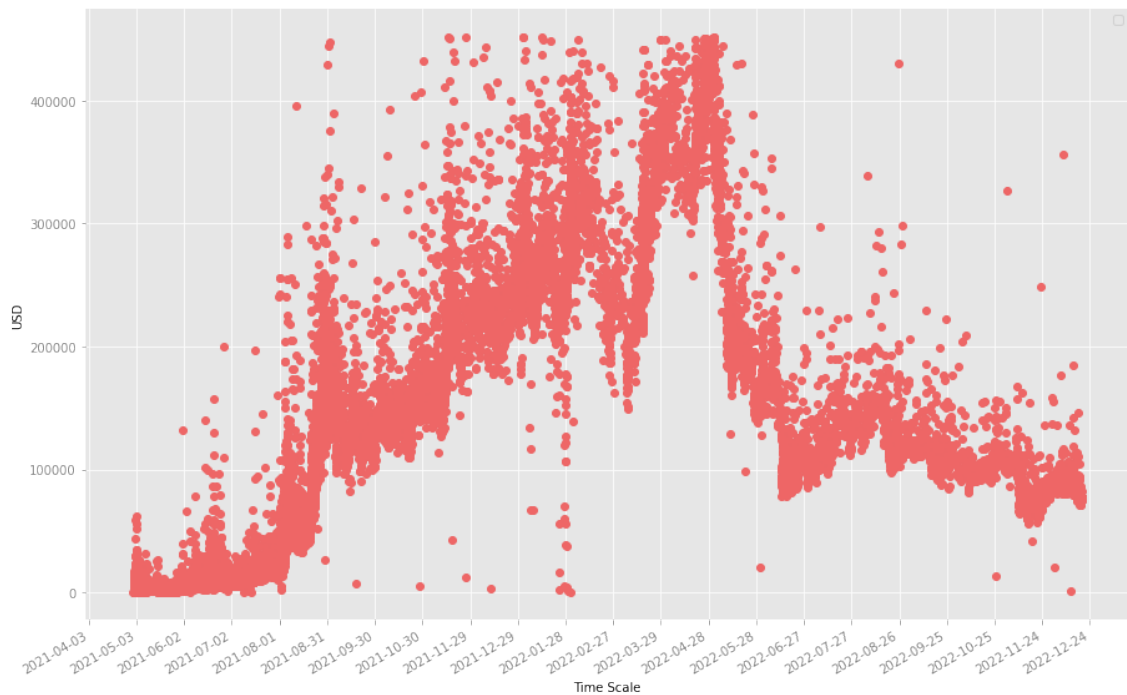


Figure 4.6: BAYC - Total Sales

To look at the price trend, we averaged the sales of the collection per day. Figure 4.7 displays the trend when using the unfiltered data, without removing outliers. Figure 4.8 displays the trend after removing the outliers. While still very volatile, we can see that in the second chart the price

change makes more sense on a day-to-day basis, without sudden up or down movements. We see the steady increase in price over time since its creation in April 2021, with a big drop in April 2022 coinciding with the big drops in every major stock and crypto market, whose instability still remains to this day and lead the price to trend downwards.

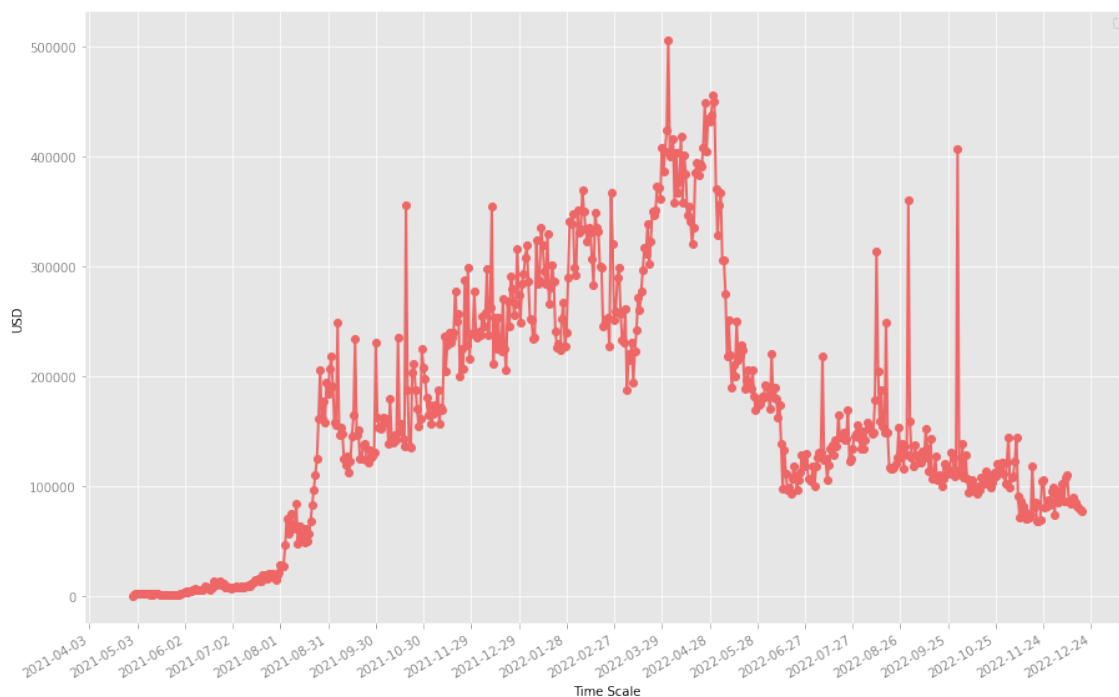


Figure 4.7: BAYC - Average Sales value per day without removing outliers

In Figure 4.9 we display the sales of the 10 BAYC tokens with most sales, in an attempt to understand what is the sale volume per token and how spaced in time they are. The same data is displayed in the table present in Figure 4.10, with the addition of information about the Maximum sale of each token, the Minimum sale, the Average sale price, and the Average difference in days between sales of that token.

We can see that there is not a lot of data about each token, the one with most sales, token 8069, has 47, over the span of 2 years. This value drops to 38 for token 2193, and then quickly decreases to smaller values, with the average number of sales per token being 3 (2.9461). Even for the token with most sales, the average length of time between sales is 12 days, with the value quickly rising to 15, 17, and more days. The average of the average length of time between sales for all tokens of the collection is of 153 days. This highlights one of the problems of NFT pricing in comparison with the pricing of stocks or cryptocurrencies, since each token has a very scarce amount of data and it becomes hard to infer a value with such a big time gap.

Besides sales data, we also take a look at the metadata that we will use in the models, in particular the traits of NFTs. Figure 4.11 shows the *Fur* trait for BAYC, with all its possible values and the number of tokens that have it. We can see the different rarity between values of the same trait, with *Brown* fur being the most common value with 1,370 tokens, and *Solid Gold*

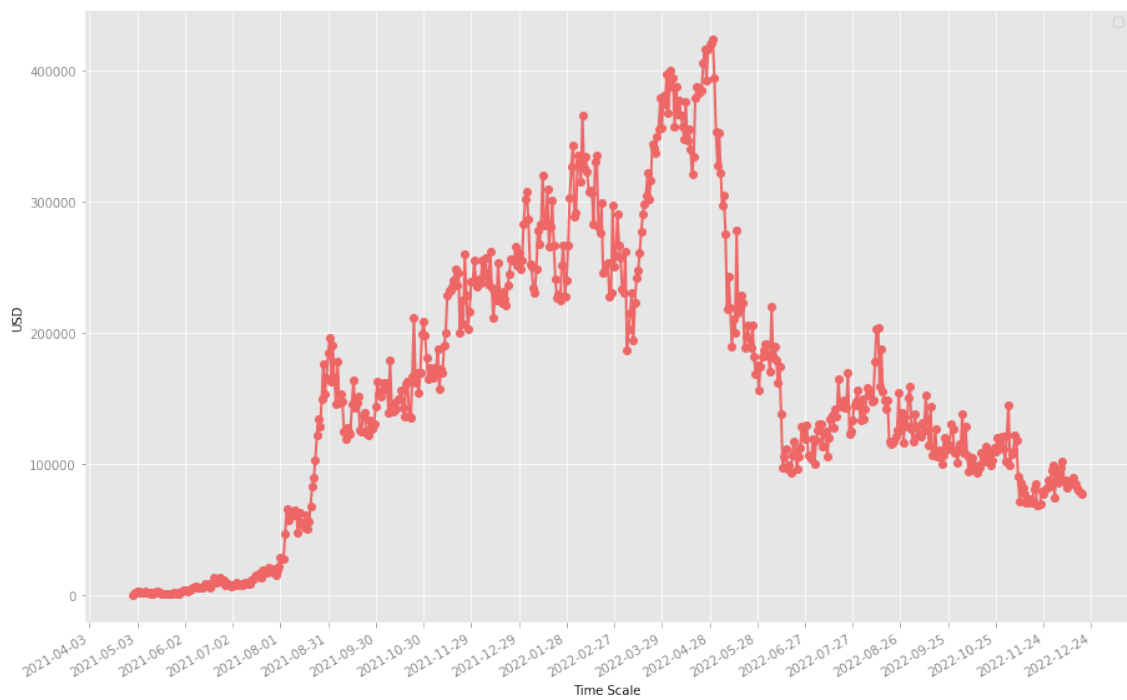


Figure 4.8: BAYC - Average Sales value per day

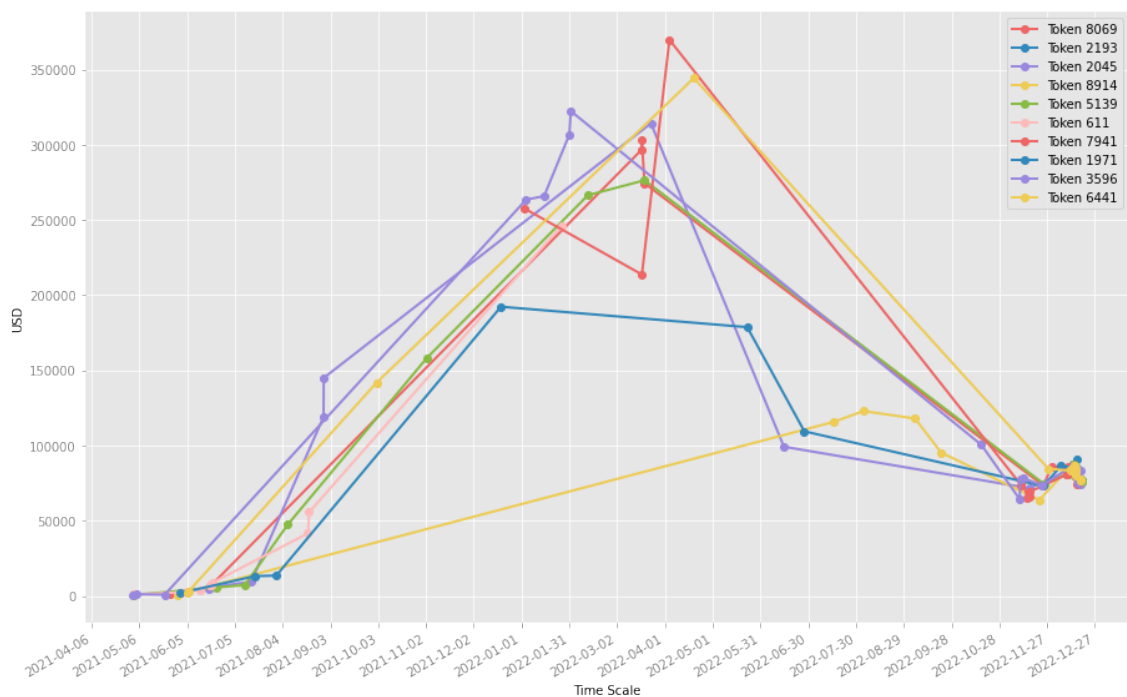


Figure 4.9: BAYC - Tokens with most sales

being the most rare with 46. If we look at the floor value of these tokens, as of 16/01/23, *Brown fur* is 74.6 ETH and *Solid Gold* is 760 ETH, meaning the rarity of this trait in particular heavily influences the valuation. In Figure 4.12 we can see all the traits of BAYC and that most Traits

Token ID	Total Sales	Max Sale (\$)	Min Sale (\$)	Avg Sale (\$)	Avg Sales Time Diff (days)
8069	47	303446.09	1054.99	90792.631915	12.326531
2193	38	91175.04	75135.48	81820.058947	15.100000
2045	33	314238.42	4904.97	82772.087879	17.257143
8914	32	122974.08	955.95	83782.739687	17.764706
5139	31	276378.00	796.59	88991.581935	18.303030
611	26	246141.00	1041.29	15184.988077	21.571429
7941	26	369715.50	65474.70	101342.500000	21.571429
1971	25	192341.66	1963.64	80707.287600	22.370370
3596	24	322391.53	736.16	106332.225000	23.230769
6441	24	344323.11	2208.34	89531.878750	23.230769

Figure 4.10: BAYC - Sales Statistics

follow this pattern of having a few values that are much rarer than the others.

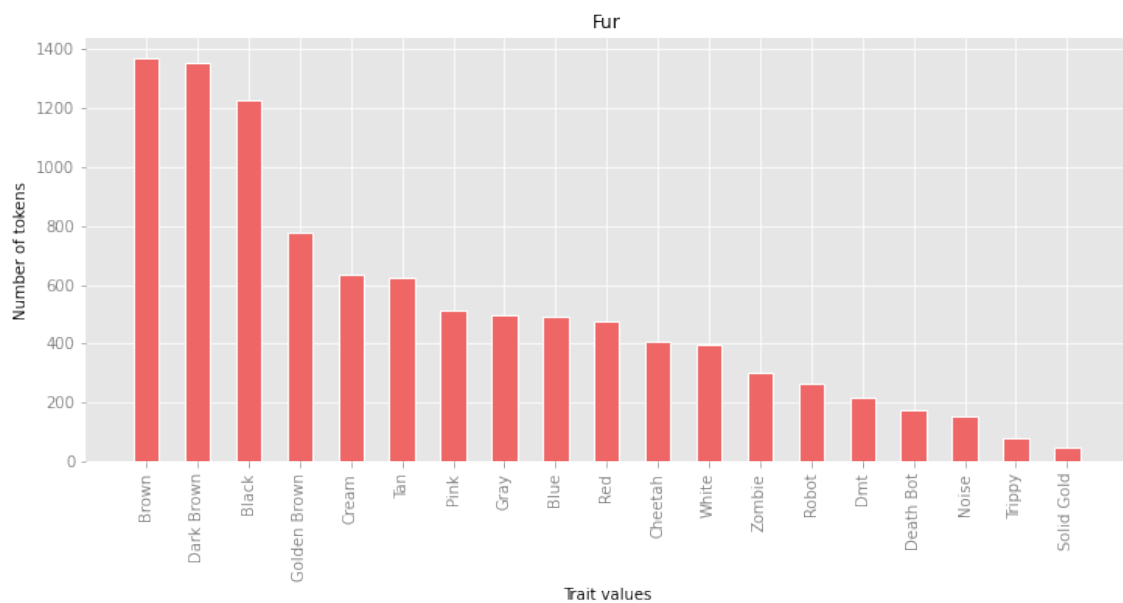


Figure 4.11: BAYC - Fur Trait

Based on these traits, we calculated a Rarity Score following the method described in Section 3.2. We can see the distribution of this score in Figure 4.13. Most NFTs fall in the 100-300 range, but it is shown that a small amount, the ones with rarer individual traits, get a higher score.

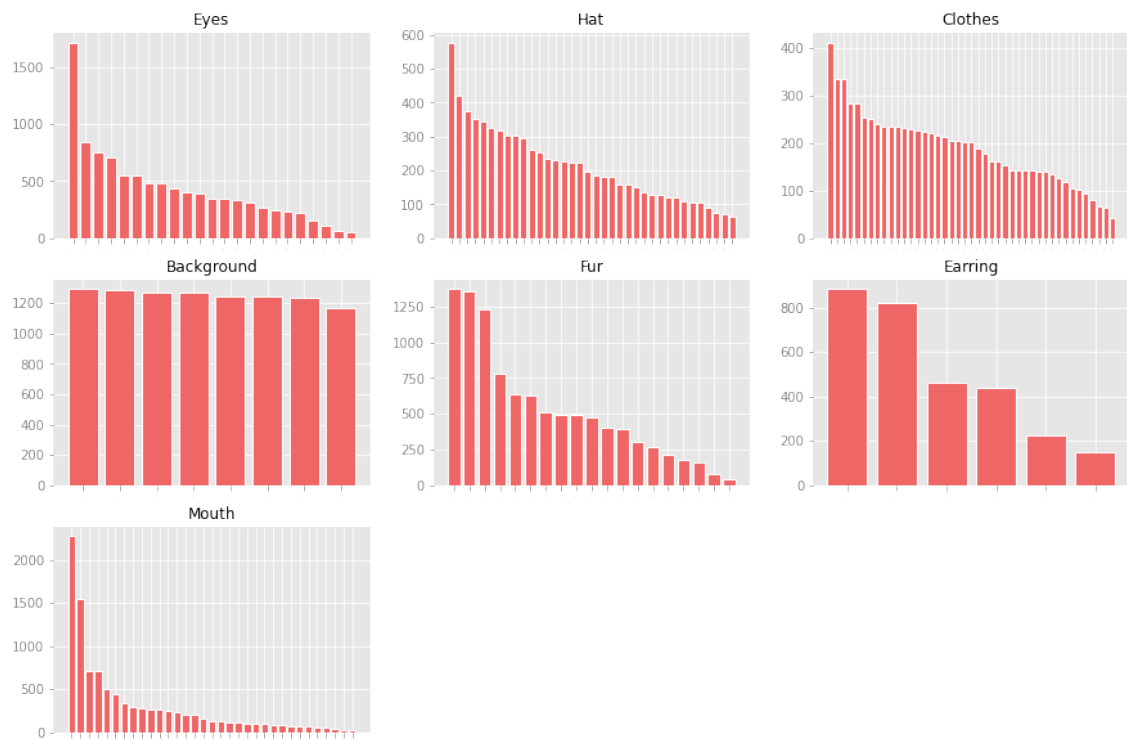


Figure 4.12: BAYC - All Traits

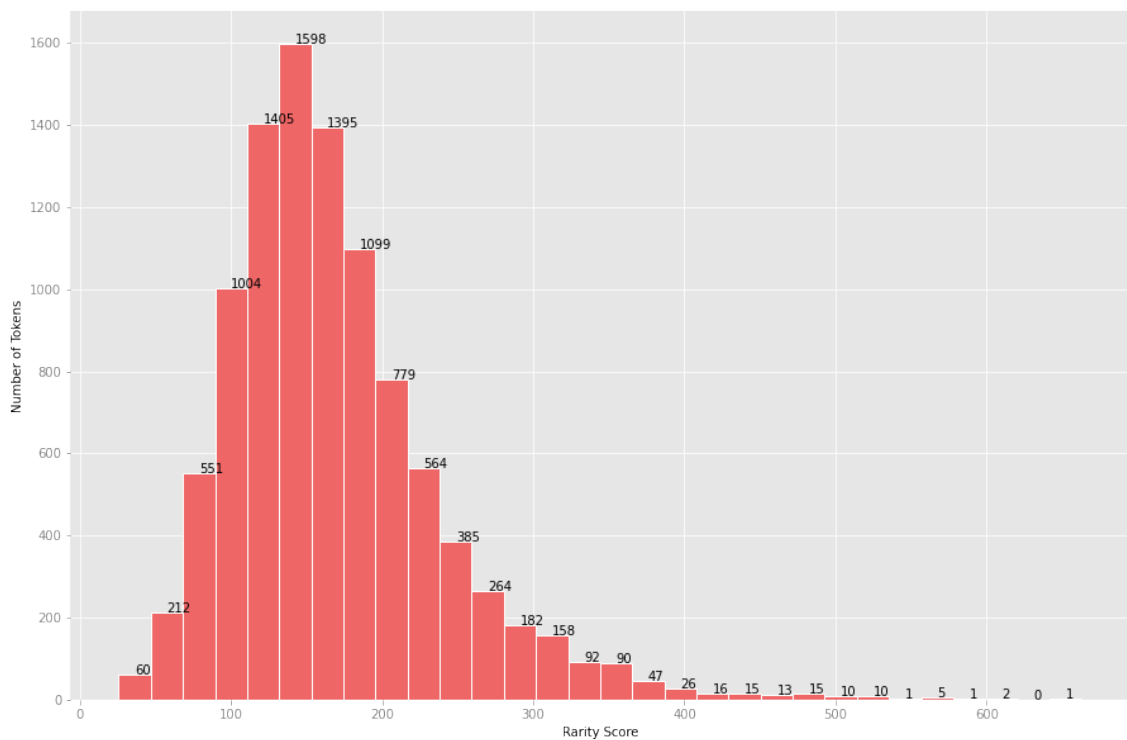


Figure 4.13: BAYC - Rarity Scores

4.2 Initial Models

Here we show the obtained results for the initial models discussed in section 3.4.3, applied over the data of Bored Ape Yacht Club. As input we used the features described in Table 3.1, removing the *collection* and *token_id* columns, since we are targeting a collection-based model. These models were chosen as a starting point for further improvements.

4.2.1 Baseline

In Figure 4.14 we display the predictions of the Baseline model for 3 separate windows of size 24. The green circles represent the real values, and the orange crosses represent the predictions of the model. Figure 4.15 displays the entire BAYC sales history as a scatter plot. The red dots are the real values, the blue dots are the predictions of the baseline model for the validation data, and the purple dots are the predictions for the test data.

We can see that, as mentioned before, what this model does is use the last prediction as the current prediction. As such, the predictions in the charts appear to be shifted 1 step to the right. While a simplification of the problem, this model was able to achieve a MAPE value in validation of 29.4261, and in test of 21.6949. This means that, on average, predictions are 21.6949% away from the real values. This number in itself is interesting to interpret and shows how difficult this prediction problem is, where even consecutive sales are so different from each other. It is also interesting to notice how the MAPE is lower for the test set than the validation set, which at first might not be expected. This is due to the data used for the test set being more recent and more stable, with fewer upwards and downwards movements when compared with the other time periods.

4.2.2 Linear

This is a trainable model, as defined in Listing 3.2, and in Figure 4.16 we can see the evolution of the loss function and the Mean Absolute Percentage Error over the training epochs. Figure 4.17 displays the weights attributed to each input feature. Figure 4.18 shows predictions using this model for 3 separate windows of size 24, and Figure 4.19 displays the predictions for the entire BAYC collection as a scatter plot.

This model achieved a val MAPE of 23.84% and test MAPE of 24.33%. We see how the loss function tends to stabilize logarithmically over the epochs. The train value quickly decreases and doesn't change much, but the validation loss takes around 100 epochs.

Since this is a linear model, the target is predicted as a weighted sum of the feature inputs. By taking a look at the Linear model weights, we can see that it found a strong correlation with the exponential moving average of size 3, which makes sense, as this takes into consideration the last 3 days of sales and these are close in time. The value for *ewma_3* is of around 0.9, which means that for an increase in this feature of 1 unit, the price will increase by 0.9 (the values are normalized). The rarity seems to have a positive effect on the price, as well as the last token

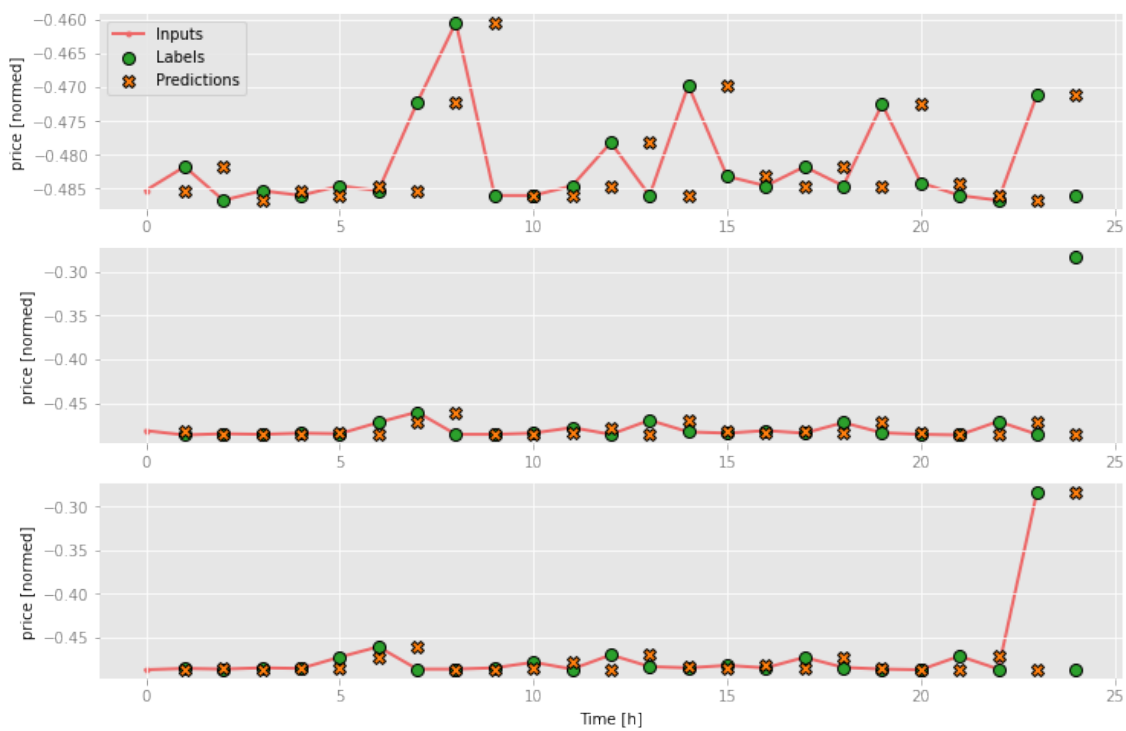


Figure 4.14: Baseline model predictions on 3 separate windows

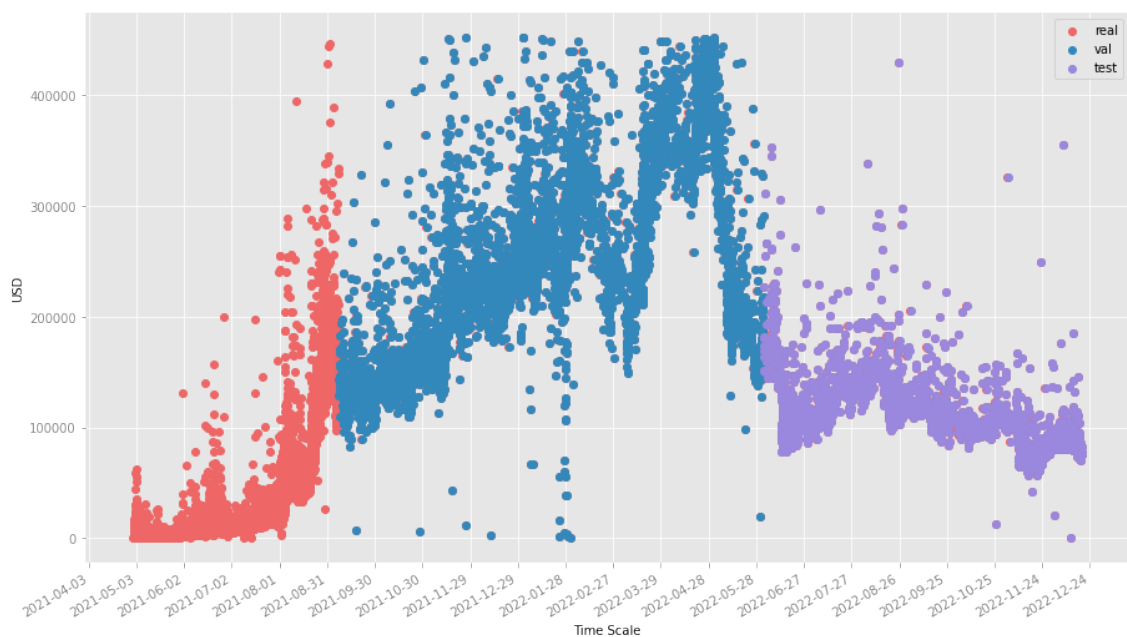


Figure 4.15: Baseline model predictions for the BAYC dataset

price. On the other hand, the current price of ETH doesn't seem to influence the prediction, nor the last collection price. These might be explained by the relationship with other variables, where arguably the moving averages already include information about the last collection price, and the timestamp relates to the ETH price. Interestingly, the model found that the moving average of size

7 has a negative impact on the result. These weights will be explored further in another section.

Looking at the predictions for the whole dataset, we can see how it seems to follow the volatility of the data, with a lot of upwards and downwards movements, but still with a considerable distance between predictions and real sales.

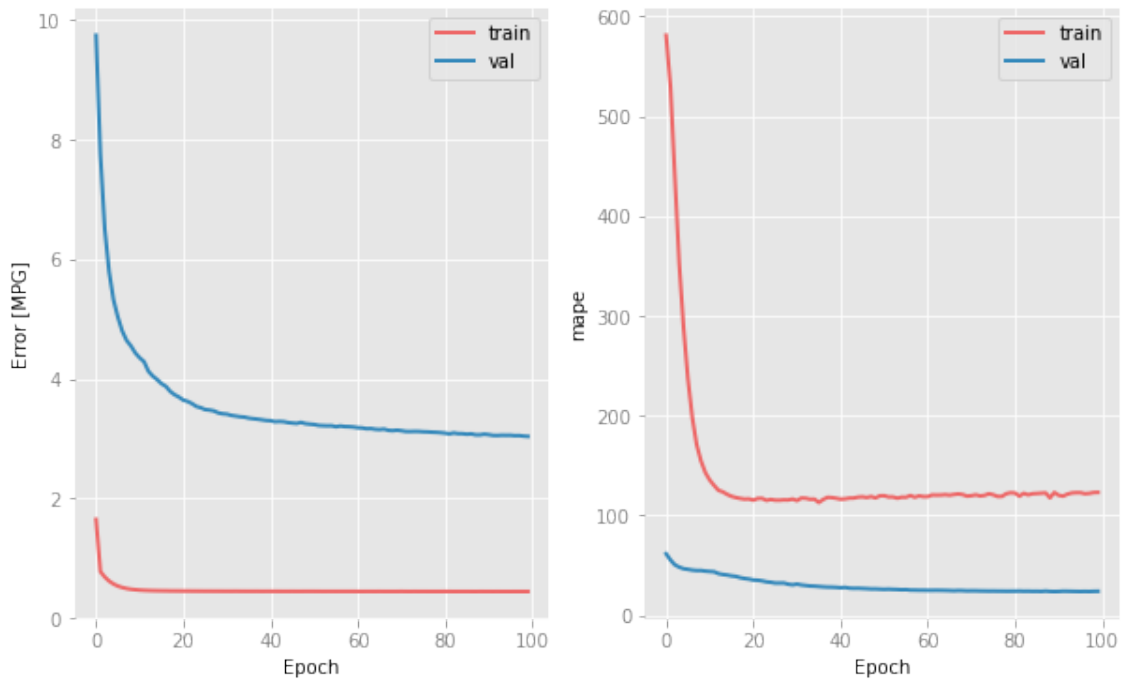


Figure 4.16: Linear model metrics during training

4.2.3 Dense

In Figure 4.20 we can see the evolution of the loss function and the Mean Absolute Percentage Error over the training epochs for the dense model defined in Listing 3.3. Figure 4.18 shows predictions using this dense model for 3 separate windows of size 24, and Figure 4.19 displays the predictions for the entire BAYC collection as a scatter plot.

This model achieved a val MAPE of 28.5605, and a test MAPE of 45.47. While more complex than the linear model, it is interesting to note how the model seems to have worse performance. Looking at the predictions for the whole dataset, the model is less dynamic and doesn't react as well to up and down movements, providing a more stable prediction that doesn't follow the volatility of the real data.

4.2.4 Multi-step Dense

The loss and MAPE values over training are present in Figure 4.23. Figure 4.24 shows predictions using this model for 3 separate windows of size 24, and Figure 4.25 displays the validation and test predictions as a scatter plot.

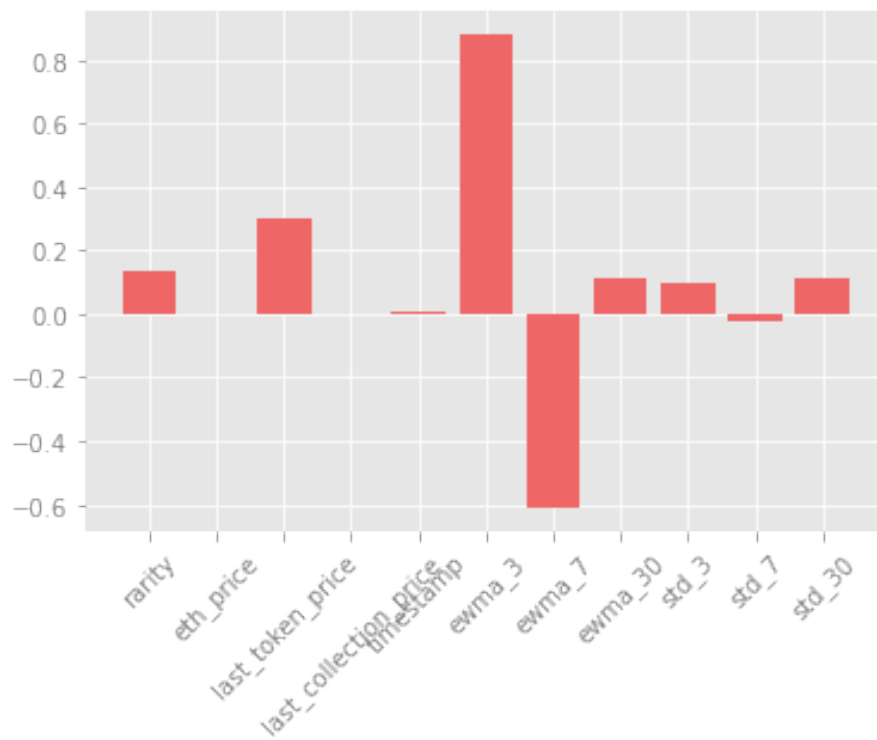


Figure 4.17: Linear model weights

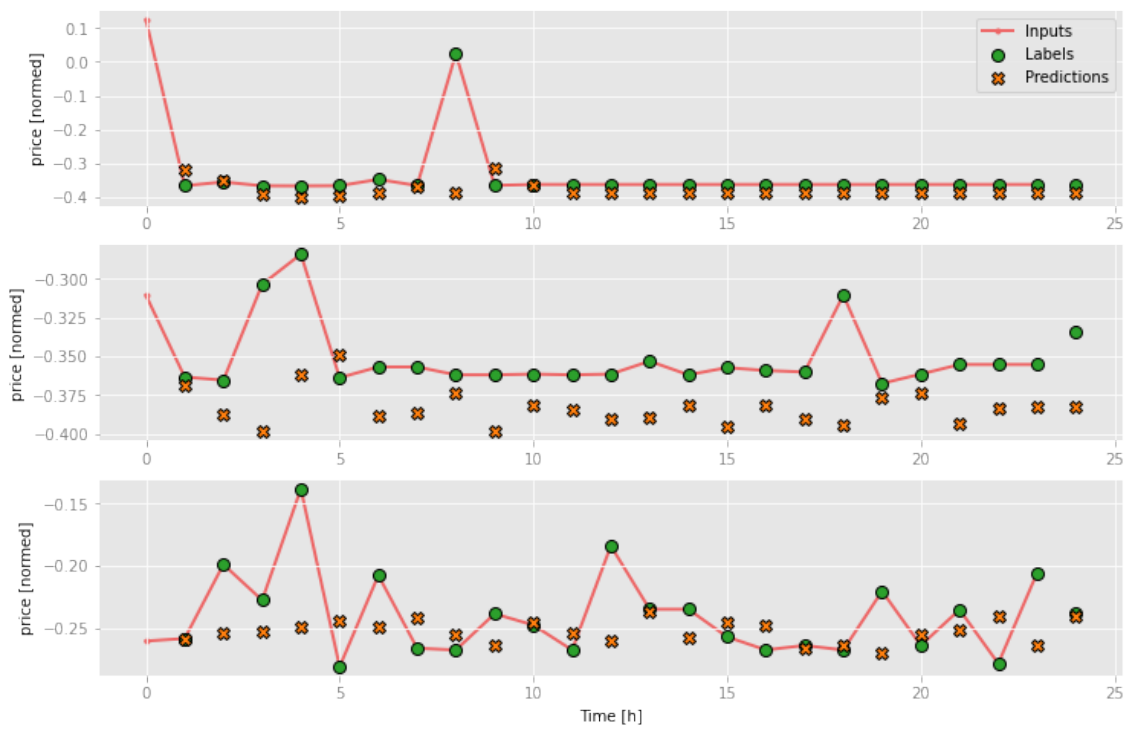


Figure 4.18: Linear model predictions on 3 separate windows

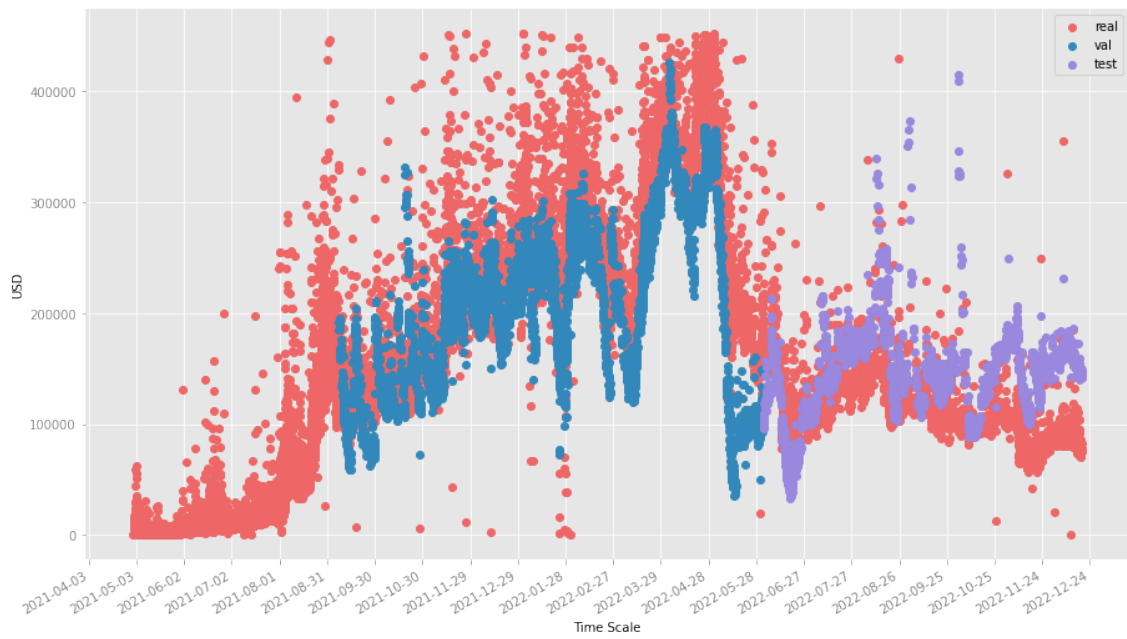


Figure 4.19: Linear model predictions for the BAYC dataset

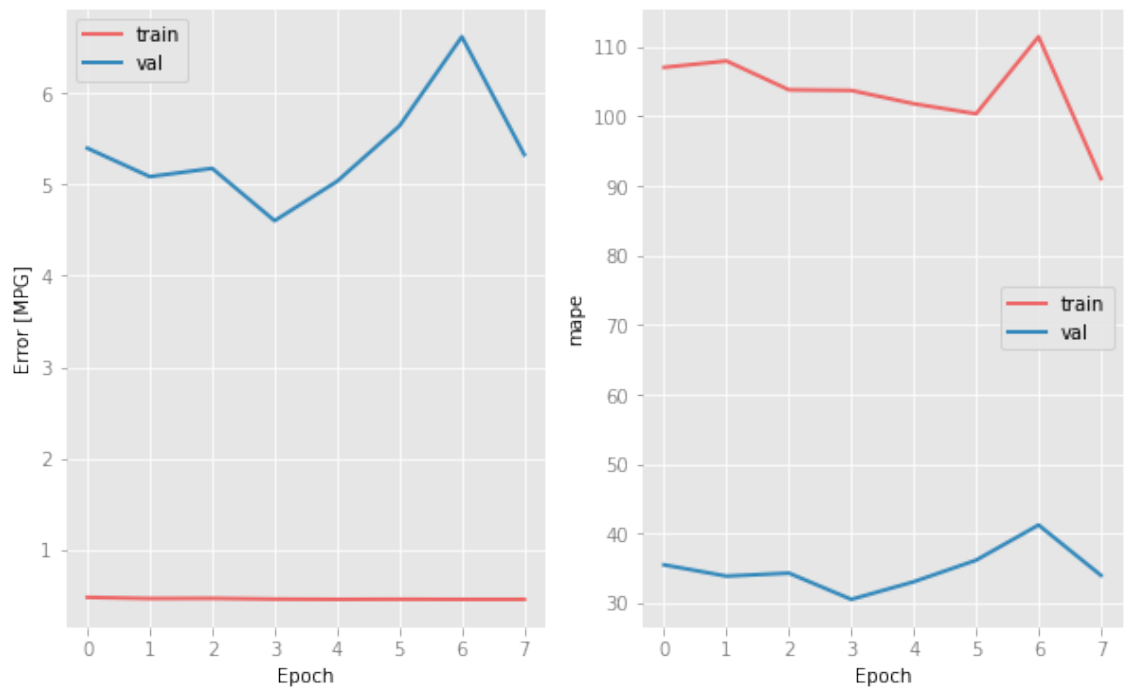


Figure 4.20: Dense model metrics during training

Comparing with the Dense model, adding the windowing component does not seem to improve results. The model achieved a val MAPE of 56.90% and test MAPE of 52.38%.

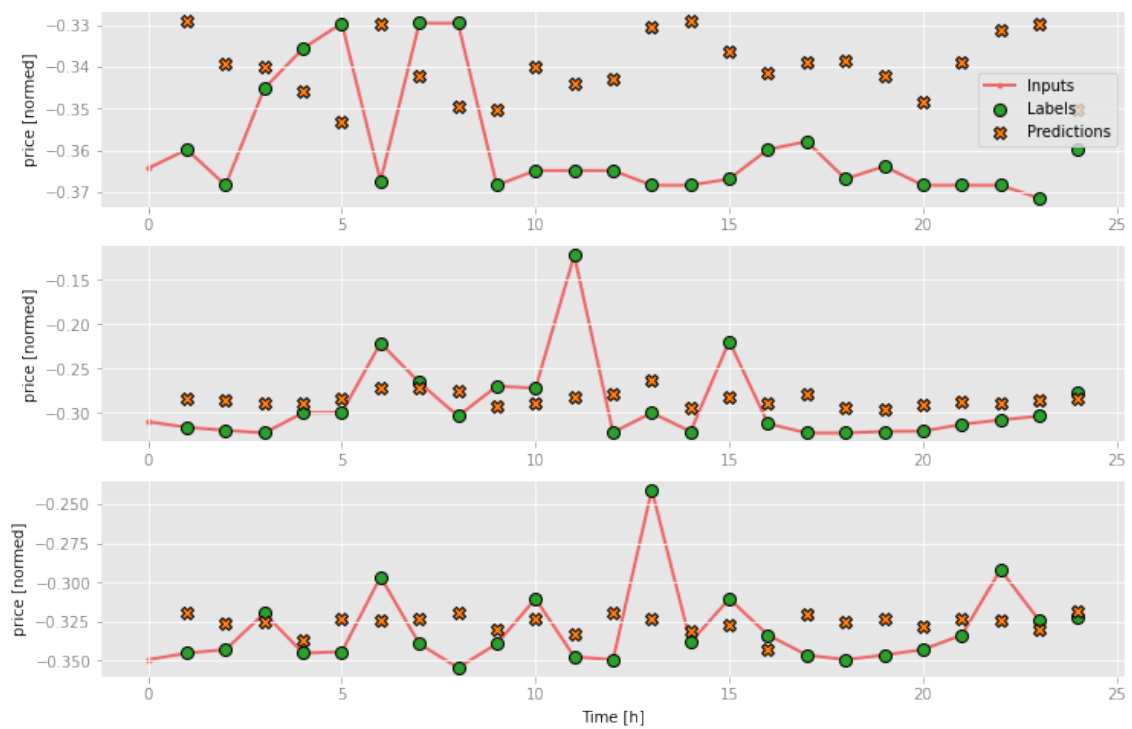


Figure 4.21: Dense model predictions on 3 separate windows

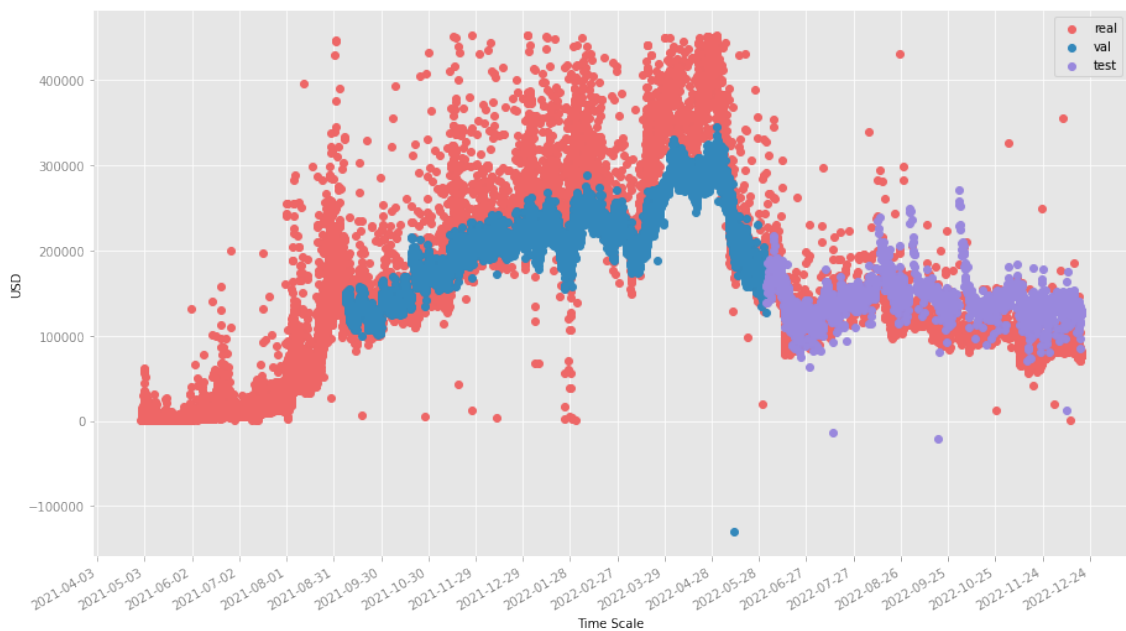


Figure 4.22: Dense model predictions for the BAYC dataset

4.2.5 Convolutional Neural Network

Training loss and MAPE can be seen in figure 4.26, predictions for 3 windows of data in Figure 4.27, and predictions for the whole dataset in Figure 4.28.

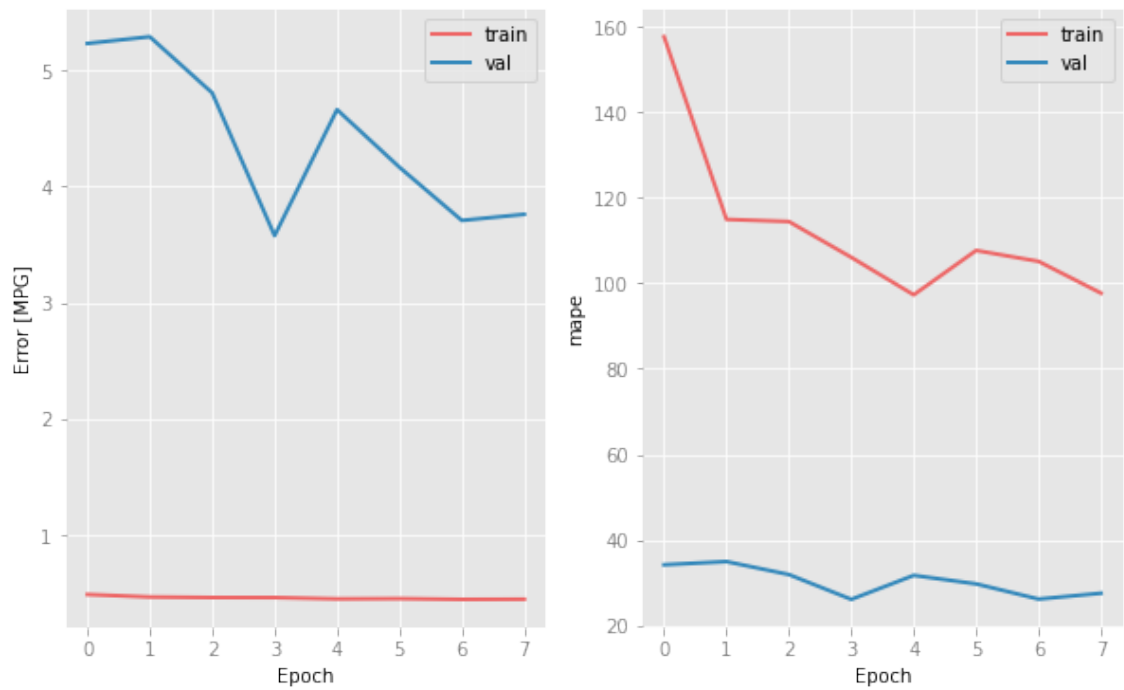


Figure 4.23: Multi-step Dense model metrics during training

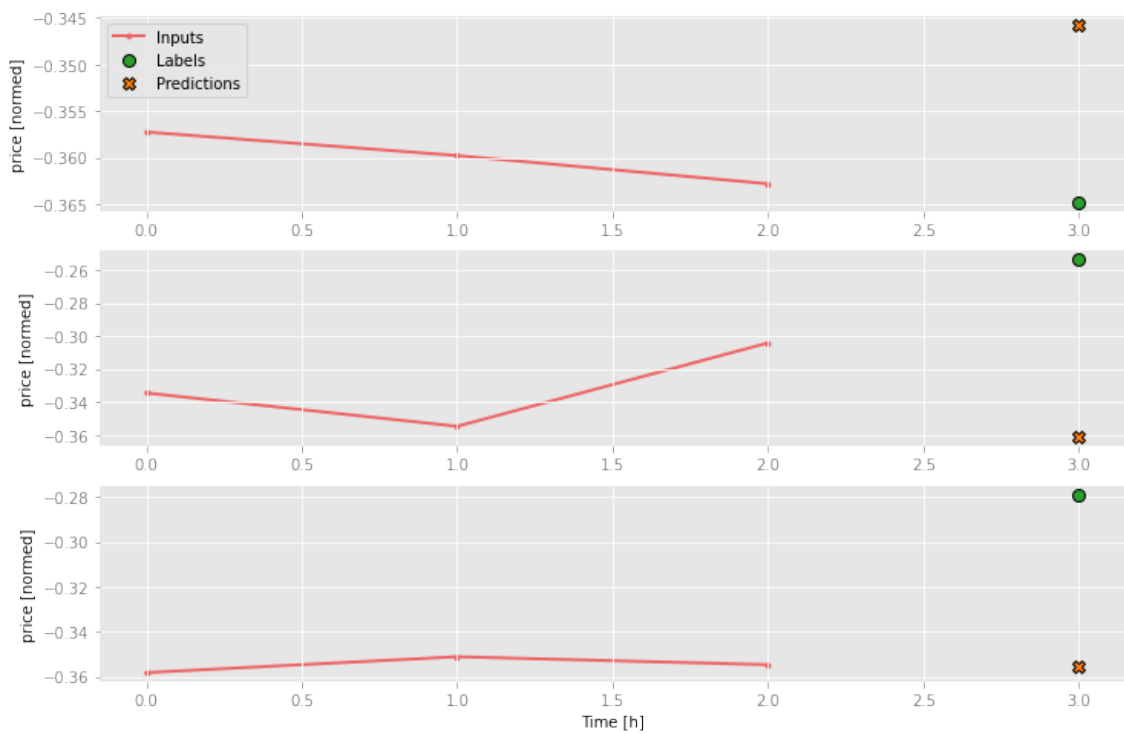


Figure 4.24: Multi-step Dense model predictions on 3 separate windows

The convolutional model achieved a val MAPE of 130.84% and test MAPE of 32.53%. This extremely high value for the validation set seems to indicate this model is not suitable for this

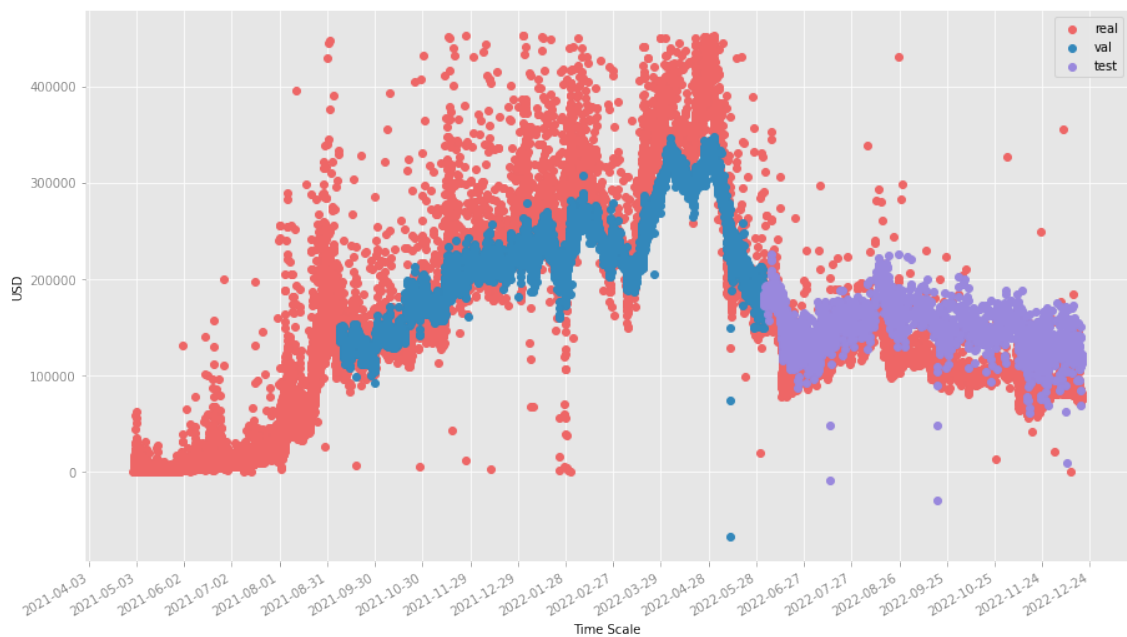


Figure 4.25: Multi-step Dense model predictions for the BAYC dataset

problem. If we look at the loss function over time, we see that the validation loss decreases and increases again, finishing at a higher value than the initial one. Looking at the predictions for the whole dataset, we see how far the validation results are from the real data.

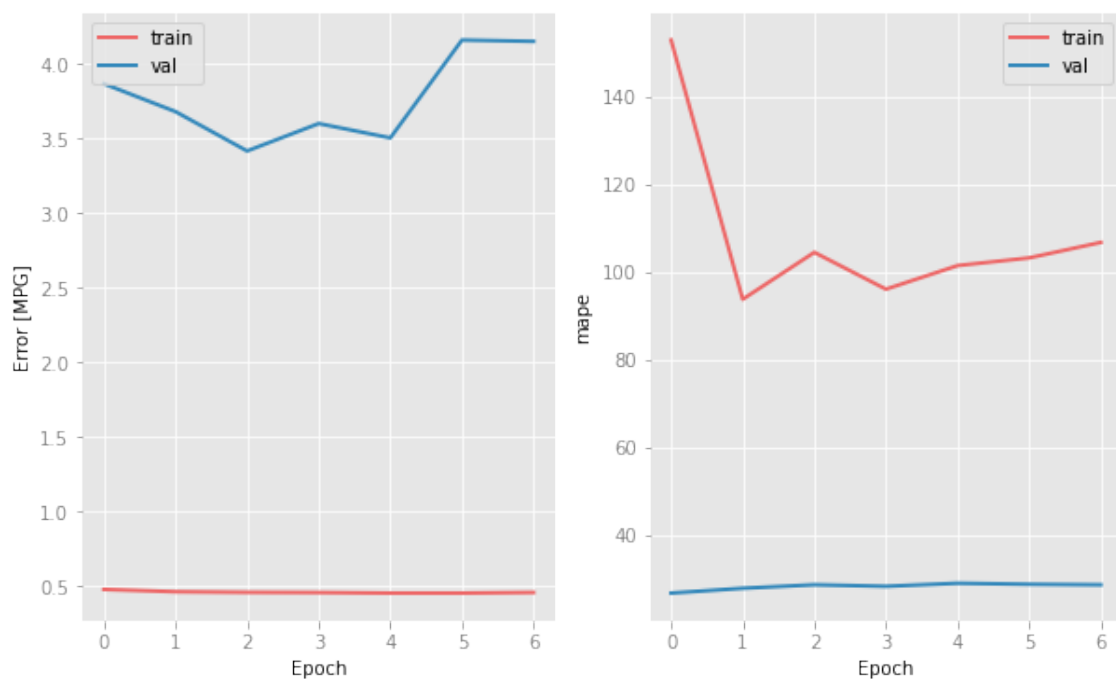


Figure 4.26: Convolutional model metrics during training

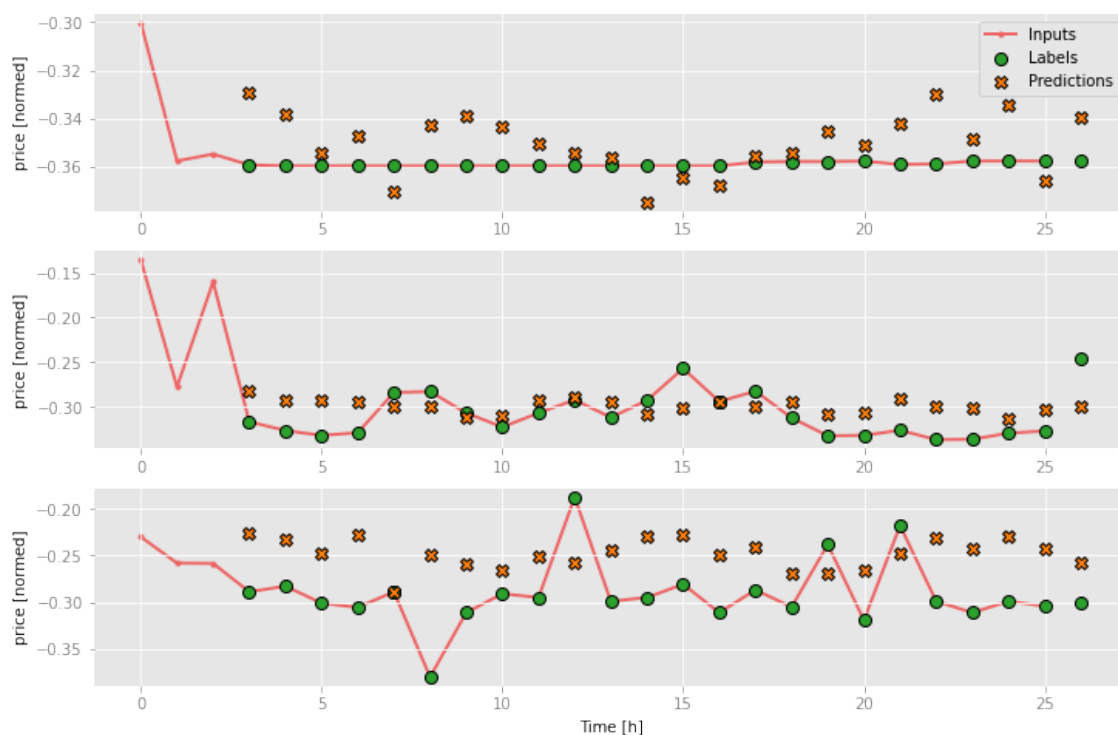


Figure 4.27: Convolutional model predictions on 3 separate windows

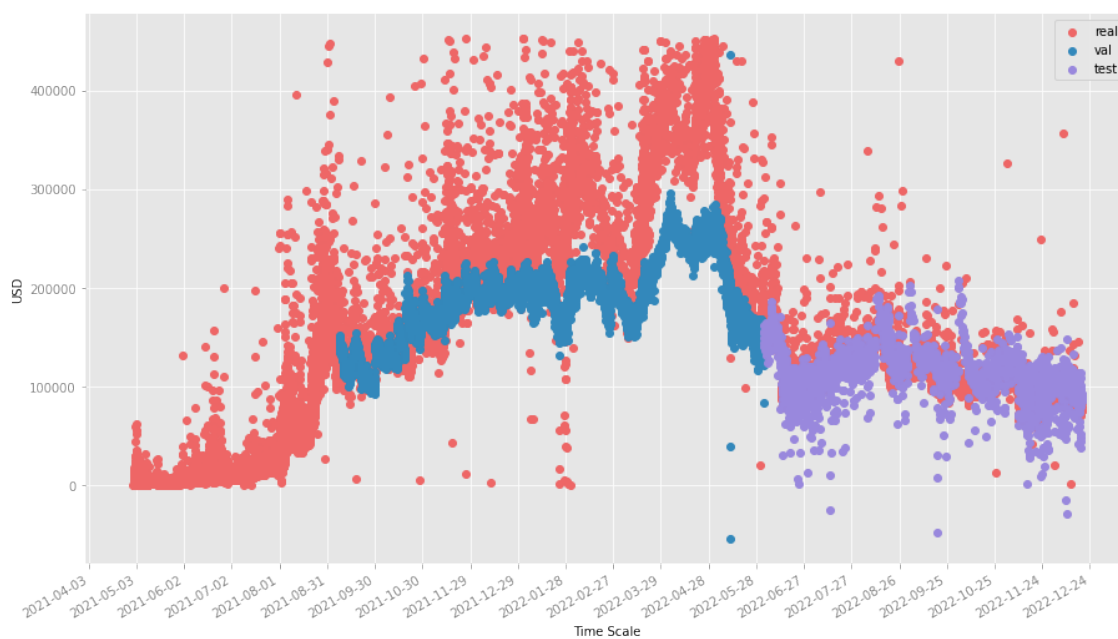


Figure 4.28: Convolutional model predictions for the BAYC dataset

4.2.6 Recurrent Neural Network

Training loss and MAPE can be seen in figure 4.29, predictions for 3 windows of data in Figure 4.30, and predictions for the whole dataset in Figure 4.31.

The val MAPE was 22.998%, the best one among the models in this section. The test MAPE was 31.49%.

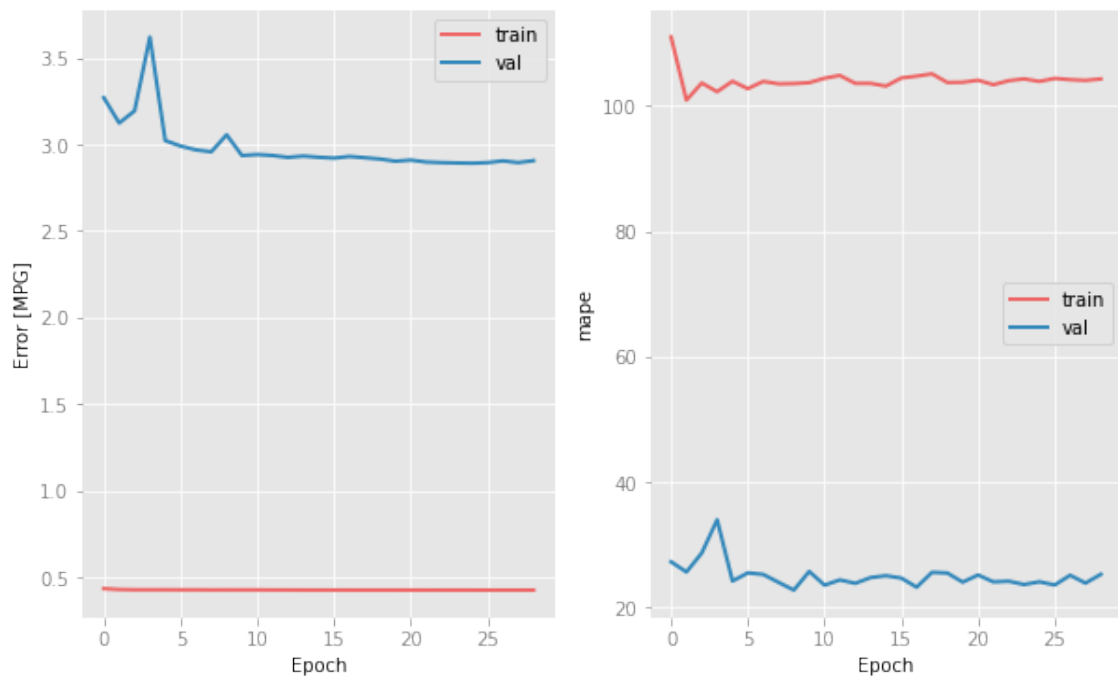


Figure 4.29: LSTM model metrics during training

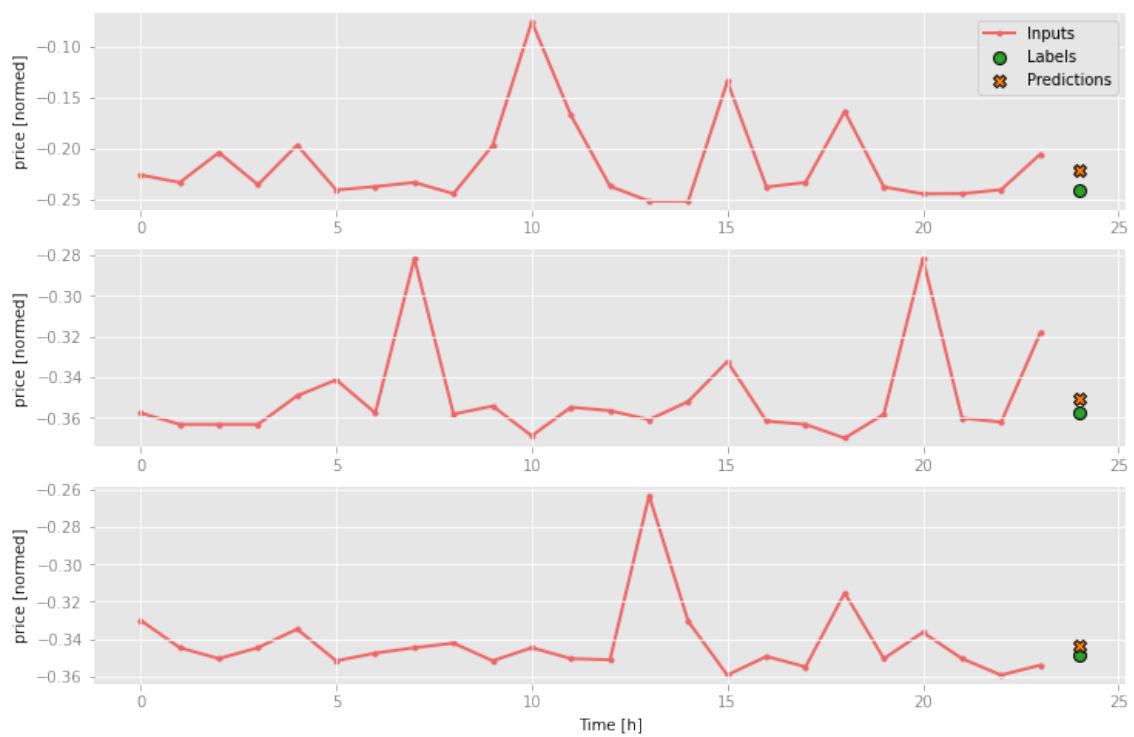


Figure 4.30: LSTM model predictions on 3 separate windows

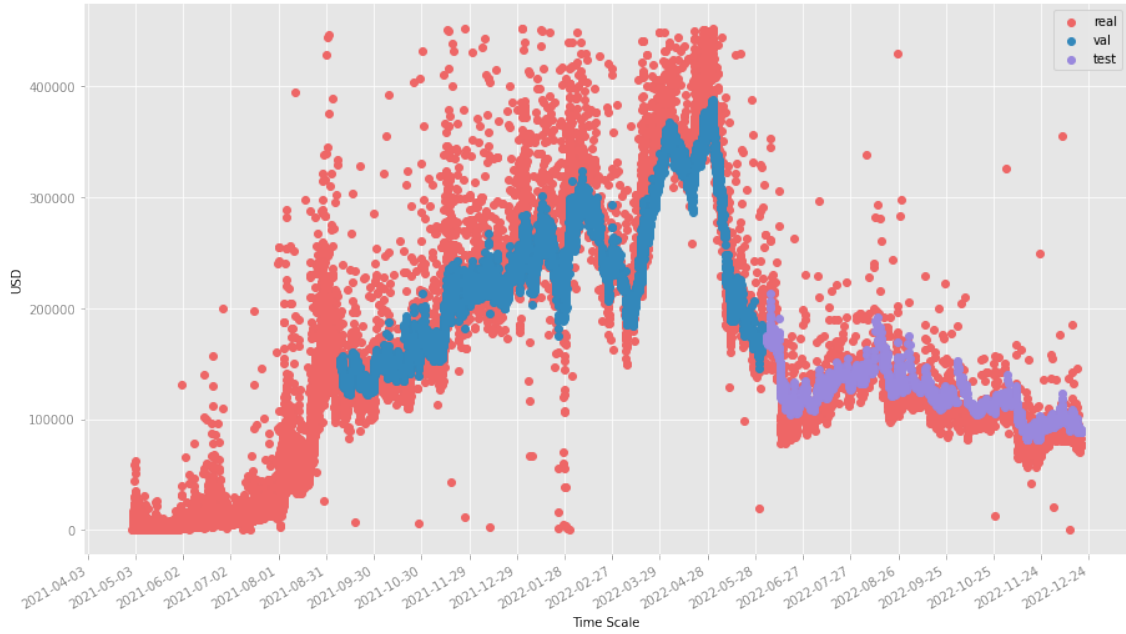


Figure 4.31: LSTM model predictions for the BAYC dataset

4.2.7 Performance

In table 4.1 we display the values that the above models achieved during validation and test steps, including the loss function, RMSE, and MAPE. The MAPE values can be more easily compared in Figure 4.32.

We can see that, from these initial models, none of them was able to achieve a better performance than the baseline model. The linear model was the one with lower test MAPE and the LSTM model the one with lower val MAPE. If we look at the RMSE value, the LSTM model performed the best.

Model	val_loss	val_rmse	val_mape	test_loss	test_rmse	test_mape
Baseline	5.6127	2.3691	29.4261	1.9456	1.3949	21.6949
Linear	3.0409	1.7438	23.8430	1.0731	1.0359	24.3256
Dense	4.2338	2.0576	28.5605	1.3078	1.1436	45.4663
Multi-step Dense	10.5130	3.2424	56.9044	1.9009	1.3787	52.3829
Conv	0.4551	0.6746	130.8435	5.1231	2.2634	32.5293
LSTM	3.0927	1.7586	22.9977	1.0730	1.0358	31.4859

Table 4.1: Performance of the initial models

4.3 Experiments

In an attempt to improve upon the initial results, we experimented with multiple aspects of the models to understand how they influence the results and how to increase the models' accuracy. In

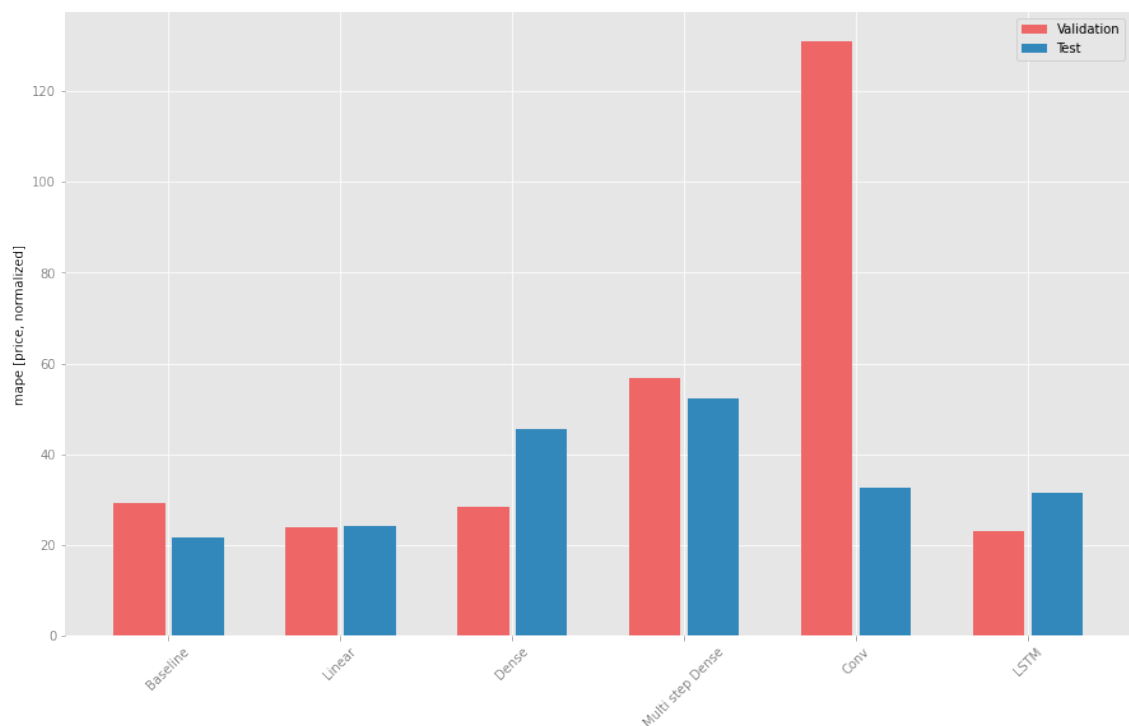


Figure 4.32: Final MAPE of the initial models

this section we still focus on the Bored Ape Yacht Club collection.

4.3.1 Shuffling the data

Here we considered shuffling the input data, instead of keeping the chronological order, while maintaining the same set of input features as before.

Figure 4.33 displays the resulting MAPE values for the models without any shuffling of the data. In Figure 4.34 we display the MAPE for the models after shuffling the whole dataset before splitting into train, validation and test. In Figure 4.35 we show results when we shuffle the data after it has been split into the 3 sets, so there are 3 independent shuffles happening.

We can see that shuffling the entire dataset highly increases the MAPE results of the models to above 100%. On the other hand, there is a benefit in shuffling the datasets after they have been split in train, val, and test. This can be explained by the high volatility of the data, with initial prices in April 2021 being very low, and peak prices in Fall 2021 being extremely high. By shuffling the data in its entirety, the model is faced with extreme changes from input row to row, and has trouble learning. On the other hand, by first grouping the data in 3 separate sets, we reduce the volatility of each one. The validation and test sets will not contain the initial sudden increases and decreases that are typical of a project launch, and will be much stabler. Shuffling this data will instead result in the model becoming more robust to small changes and increasing the prediction accuracy.

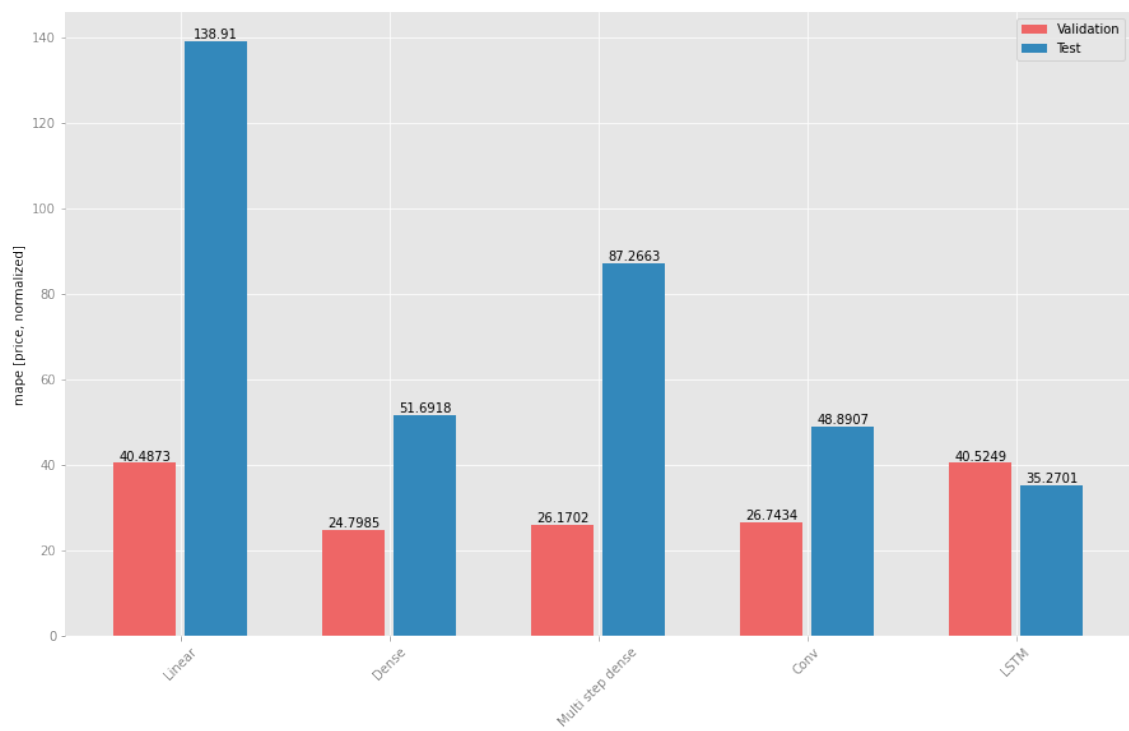


Figure 4.33: MAPE results with no shuffling

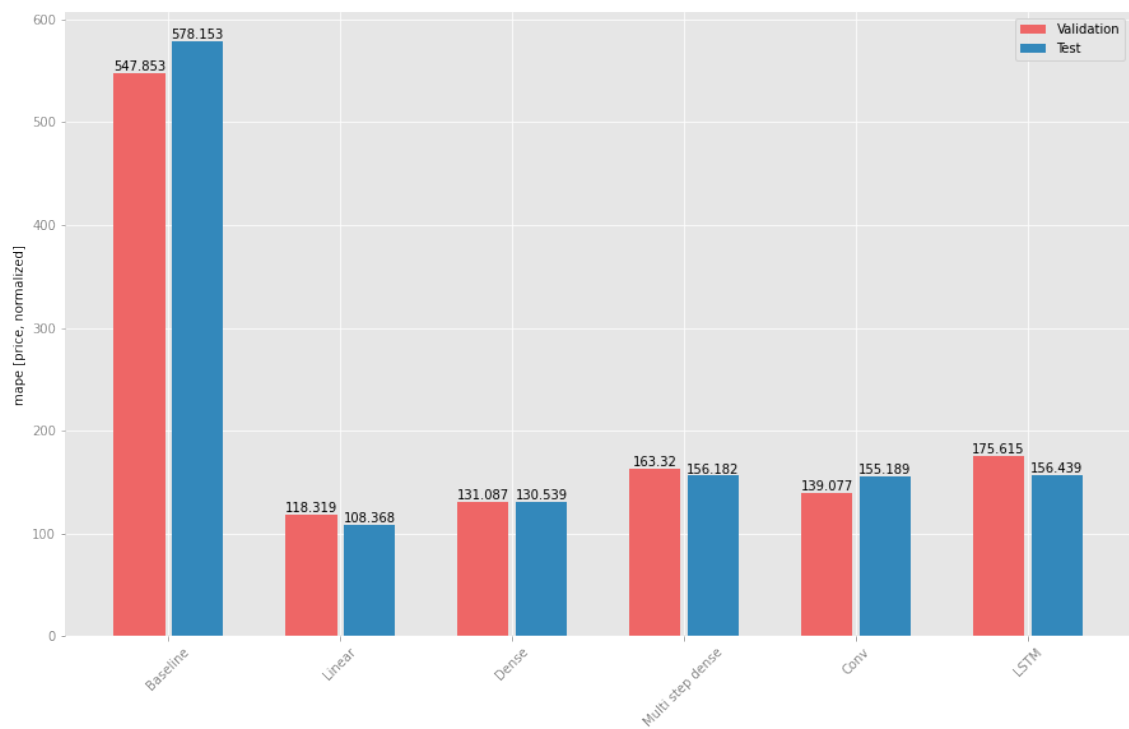


Figure 4.34: MAPE results after full data shuffle

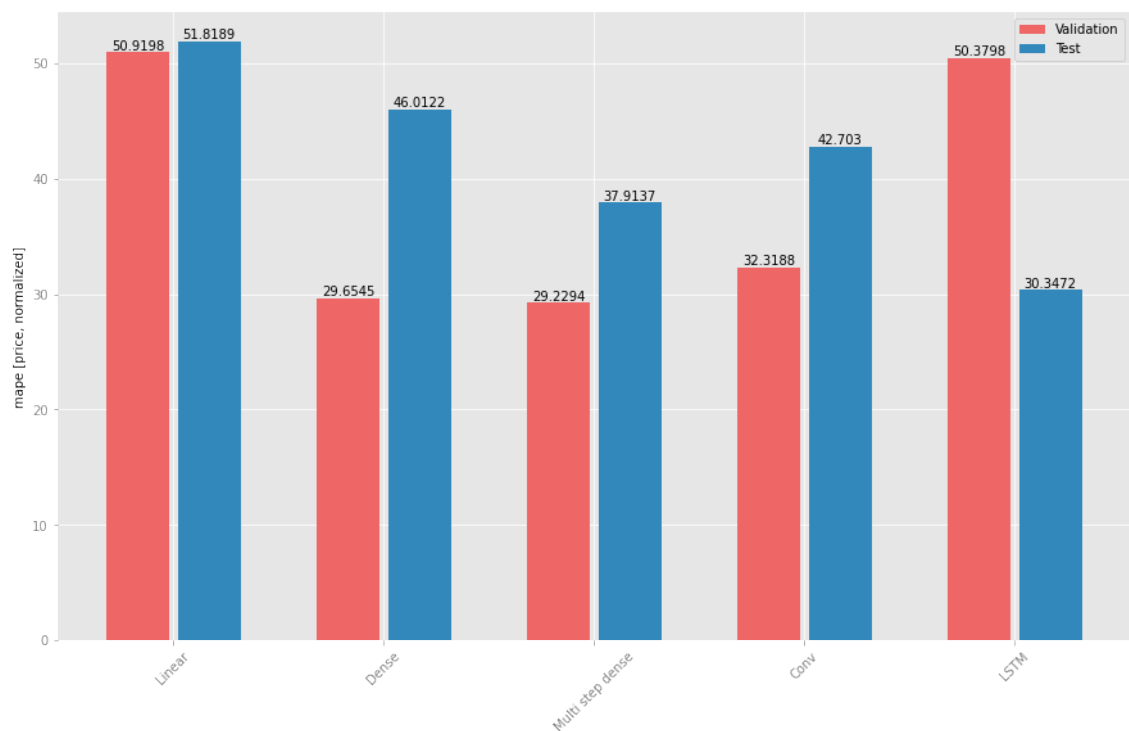


Figure 4.35: MAPE results while shuffling data in each set

4.3.2 Linear model features

Testing the linearity of the problem, we decided to assess the performance of the Linear model when ran with different combinations of input features. As such, we analysed the following 15 configurations present in Table 4.2. Figure 4.36 shows the resulting linear weights attributed to each input feature on the different configurations. Figure 4.37 displays the final MAPE value for the validation and test sets of each configuration.

Input features	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
rarity	X	X	X	X	X	X	X			X	X	X		X	
timestamp	X	X	X	X	X	X									
last_token_price		X	X												
last_collection_price	X	X	X						X		X				X
eth_price			X											X	
ewma_3	X		X			X	X	X				X			X
ewma_7	X		X		X							X	X		
ewma_30	X		X												
std_3			X			X									
std_7			X		X										
std_30			X												

Table 4.2: Linear Model configurations

Looking at the linear models' weights, we can see that there is a general consensus that the

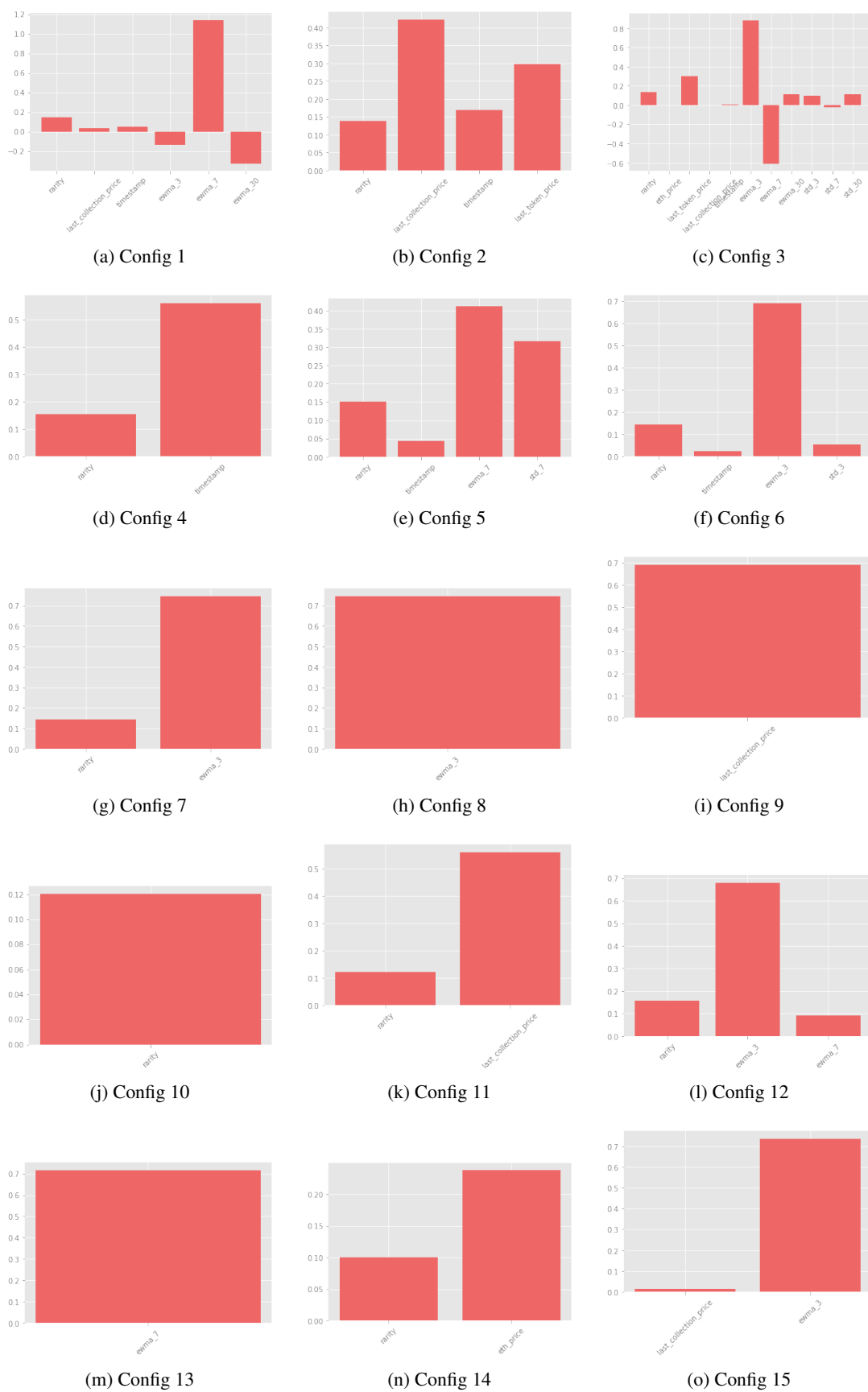


Figure 4.36: Linear model weights for different configurations

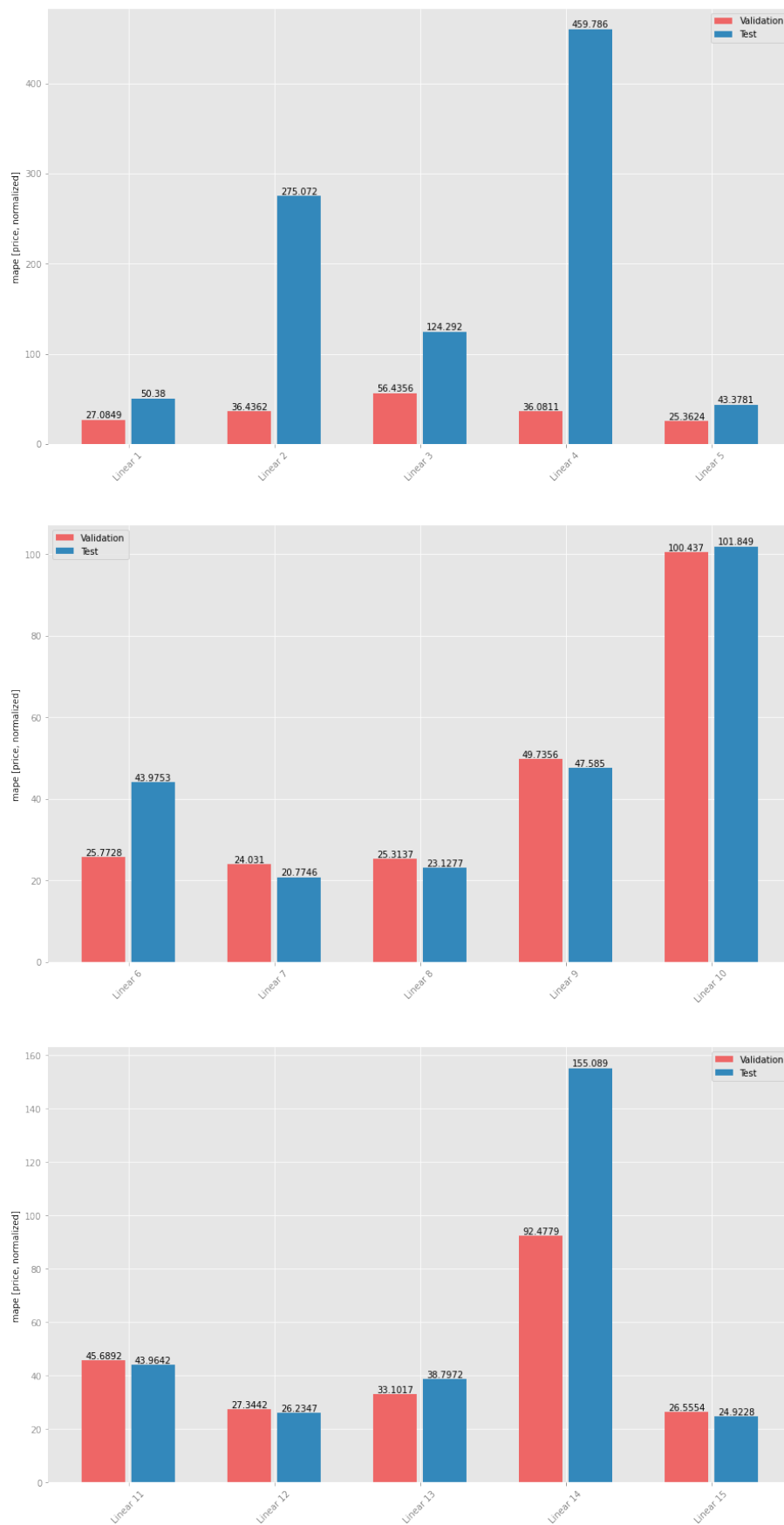


Figure 4.37: Linear model val and test MAPE for different configurations

moving average features positively affect the price prediction and are the most impactful ones. Similarly, the rarity is always seen as a positive factor in the model, meaning that higher rarities drive higher prices, as we had initially assumed. When considering multiple moving average windows, one of them tends to be the most predominantly positive one, while the others balance it out with negative weights.

The best performance was achieved by Configuration 7, in both validation and test sets, 24% and 20.8%, respectively. There seems to be a preference for simpler models that consider fewer features, where at least one of them is a moving average feature. By comparing Configurations 7 and 8, we can see that the inclusion of the rarity in the calculation helps the model achieve a lower MAPE, meaning this feature is impactful. However, as seen on Configuration 10, rarity by itself is not enough, and a variable that somehow communicates recent price movements is necessary to complement it.

Configuration 14 looked at the relationship between the price and the current value of ETH. Analysing the results, this doesn't seem to be a good predictor of the prices. Likewise, using only the last collection price by itself doesn't achieve good results.

It is worth noticing that, by only changing the input features, configuration 7 already achieves better performance than the Baseline model.

4.3.3 RNN model features

Similarly to before, we tested the performance of the LSTM model when ran with different input features. For the same 15 configurations present in Table 4.2, and for a window of size 24, we tested the base LSTM defined in Listing 3.6 and achieved the MAPE results present in Figure 4.38. As opposed to the linear model, the *price* is also present in the input features' windows.

We can see that the LSTM model achieves best performance when met with a reduced number of features. The best results are seen in Configuration 13, 15 and 7, with MAPE values of 16.7%, 17.3%, and 17.4%, respectively. These results indicate a preference for the usage of exponential moving average features (*ewma_3* and *ewma_7*), as well as *rarity* and *last_collection_price*. On the other hand, including the *timestamp* seems to make it harder for the model to learn. Comparing Config 7 and 8, there is an indication that including the rarity score improves accuracy, as opposed to only using the moving average.

4.3.4 Window sizes

Considering the sequential nature of the data and the use of windowing on the input rows, we experimented with multiple sizes of windows to see how it affected the models' performance. Having identified that the LSTM model behaves better when using fewer columns, in this experiment we used as input features the *rarity*, *price*, *ewma_3*, and *ewma_7*. In Table 4.3 we see the different window sizes we tried. We applied these windows to the Multi-step Dense model and to the LSTM model, and the resulting MAPE for validation and test sets can be seen in Figure 4.39.

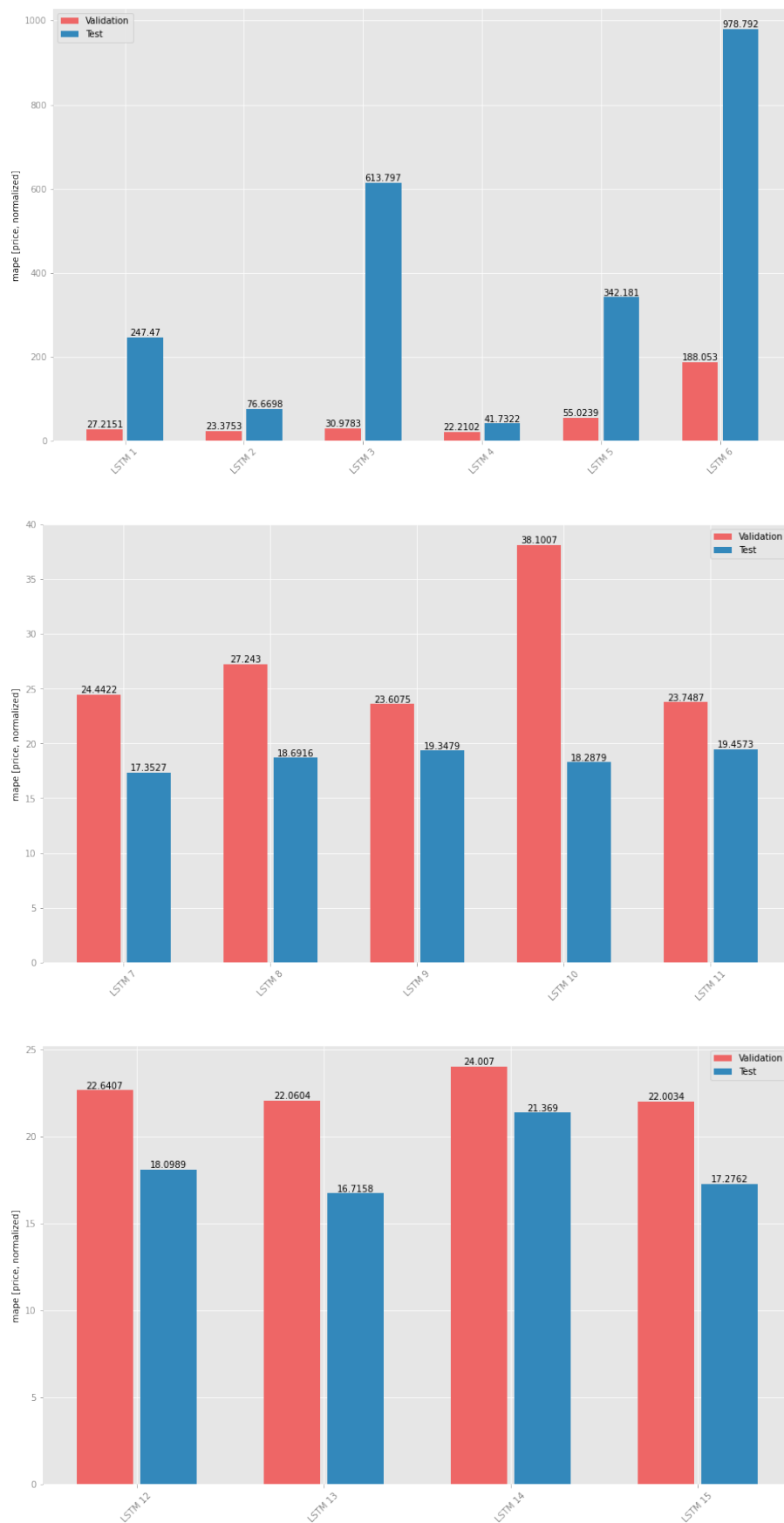


Figure 4.38: LSTM model val and test MAPE for different configurations

The LSTM model displays better results when fed a window of size 3, going as low as 13.98% MAPE for the test set. Sizes 7 and 24 also achieve acceptable results, while windows of size 1 or higher than 50 seem to have diminishing performance.

As for the Multistep Dense model, the MAPE result steadily decreases from size 1 until size 50, indicating that it benefits from higher window sizes. The best performance is achieved at size 50 with a value of 15.94%.

Config	1	2	3	4	5	6	7
Window Size	1	3	7	12	24	50	100

Table 4.3: Window sizes of each configuration

4.3.5 RNN models

Given the identified potential in Chapter 2 for the use of Recurrent Neural Networks to tackle similar problems, we decided to experiment using them with different architectures and parameters. As such, we used both LSTMs and GRUs, with different numbers of hidden layers, while stacking them on top of each other. We used as input features the *rarity*, *price*, *ewma_3*, and *ewma_7* columns. The window size is set at 24. The attempted configurations can be found in Listing 4.1. The resulting MAPE values can be seen in Figure 4.40.

We find that the best results were achieved by the simple LSTM with 32 hidden layers, and by the GRU with 100 hidden layers, with similar MAPEs of around 15.7%. In both cases, there doesn't seem to be any benefit in stacking multiple RNN layers in sequence, as it only increases the error value.

```

1 # LSTM 1
2 model = Sequential([
3     LSTM(32, activation='relu'),
4     Dense(units=1)
5 ])
6
7 # LSTM 2
8 model = Sequential([
9     LSTM(100, activation='relu'),
10    Dense(units=1)
11 ])
12
13 # LSTM 3
14 model = Sequential([
15     LSTM(32, activation='relu', return_sequences=True),
16     LSTM(32, activation='relu'),
17     Dense(units=1)
18 ])
19
20 # LSTM 4
21 model = Sequential([

```

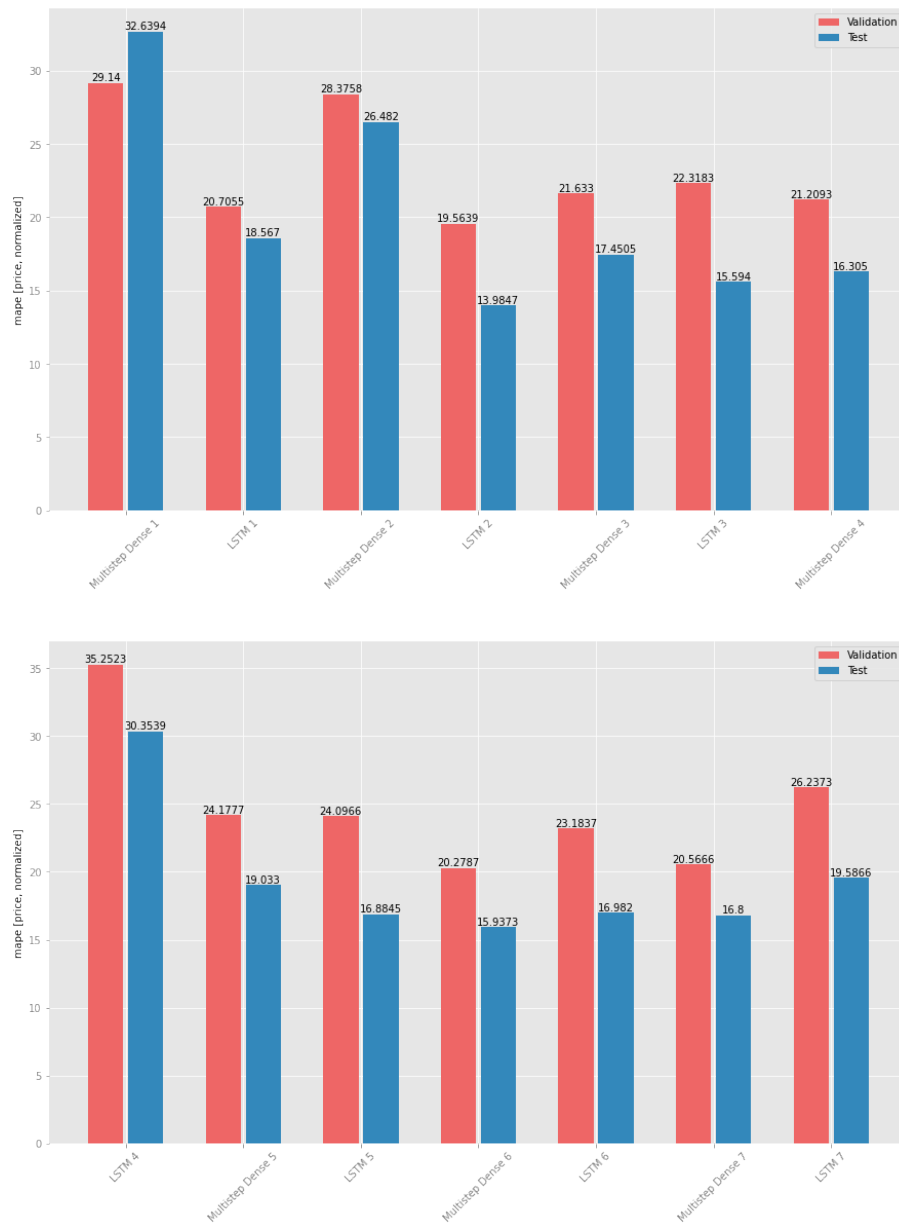


Figure 4.39: MAPE results for different window sizes

```

22     LSTM(100, activation='relu', return_sequences=True),
23     LSTM(100, activation='relu'),
24     Dense(units=1)
25 ])
26
27 # GRU 1
28 model = Sequential([
29     GRU(32, activation='relu'),
30     Dense(units=1)
31 ])
32
33 # GRU 2
34 model = Sequential([
35     GRU(100, activation='relu'),
36     Dense(units=1)
37 ])
38
39 # GRU 3
40 model = Sequential([
41     GRU(32, activation='relu', return_sequences=True),
42     GRU(32, activation='relu'),
43     Dense(units=1)
44 ])
45
46 # GRU 4
47 model = Sequential([
48     GRU(100, activation='relu', return_sequences=True),
49     GRU(100, activation='relu'),
50     Dense(units=1)
51 ])

```

Listing 4.1: RNN models

4.3.6 Loss function

So far we have been using the Mean Squared Error (MSE) as the loss function in the models, and considering MAPE and RMSE as metrics. In this part we attempt to use MAPE as the loss function and see if it improves overall performance. Similarly to previous experiments, the input features we used here were the *rarity*, *price*, *ewma_3*, and *ewma_7*.

Figure 4.41 shows the evolution of MAPE over the training epochs for multiple models, and Figure 4.42 displays the final MAPE achieved in validation and test sets.

We can see how the Linear and LSTM models have a more steady learning, where the test MAPE logarithmically decreases until it stabilizes. The other models seem to alternate between high and low val MAPE values, but with a tendency to decrease over time.

Comparing with previous experiments, we can see that, except for the Dense one, all the models are able to achieve a lower test MAPE result, going as low as 16.4%. However, there seems to be a big difference between validation and test sets, since the val MAPE is still as high

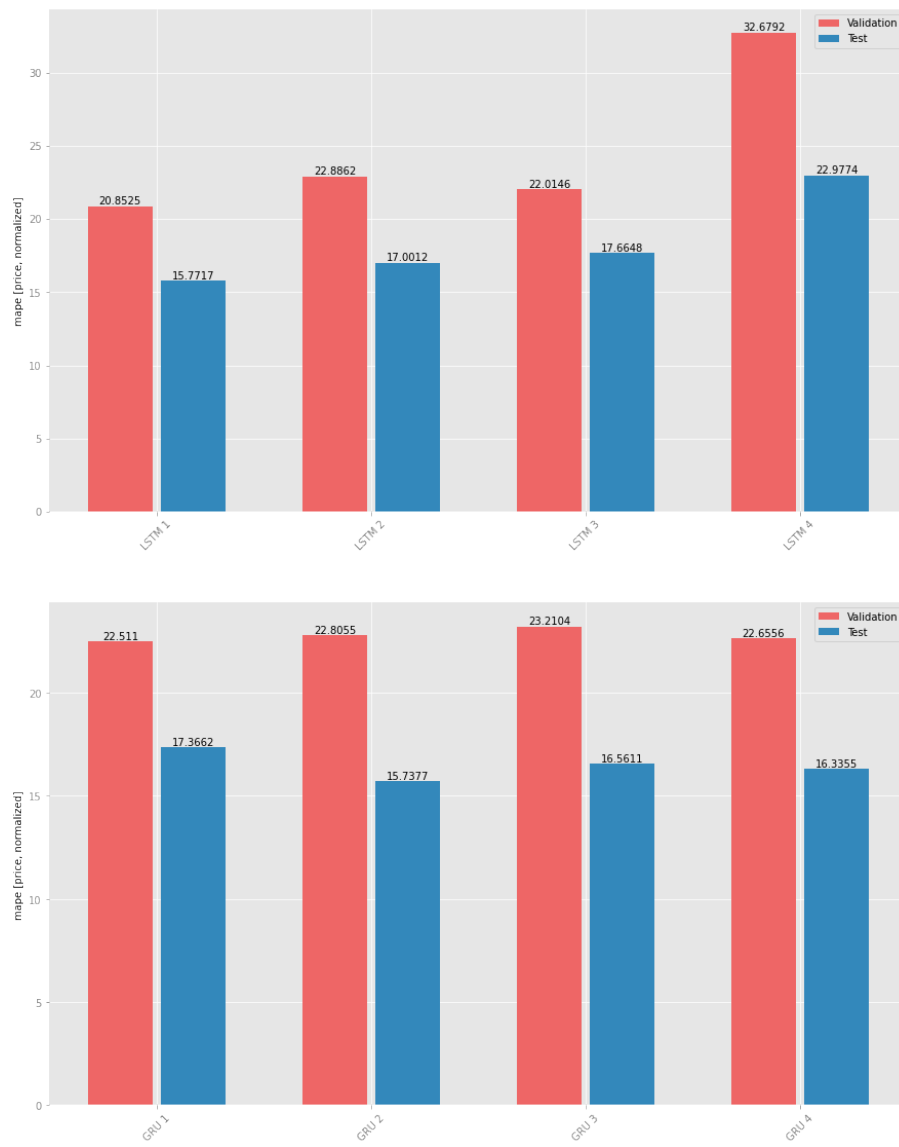


Figure 4.40: MAPE results for different RNN models

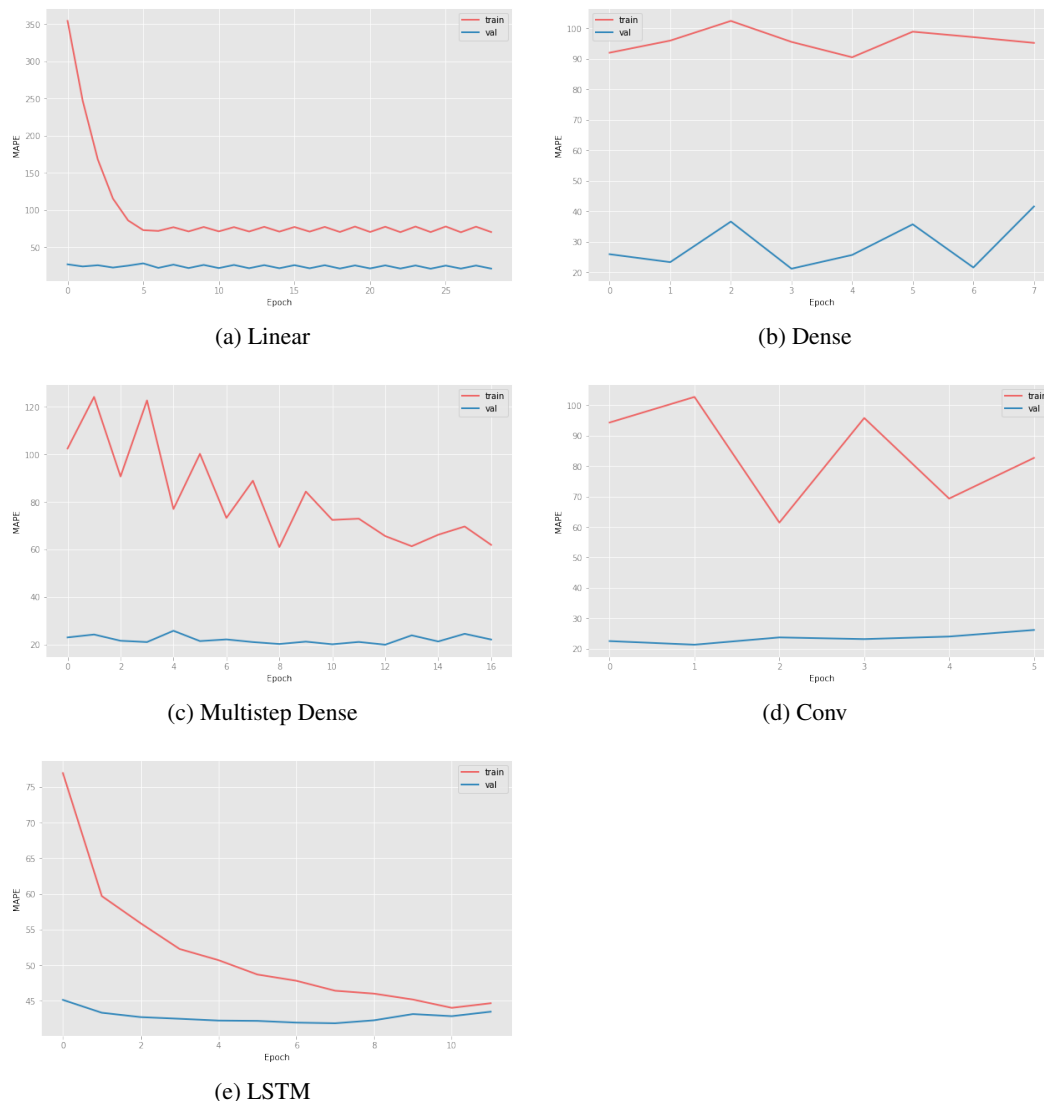


Figure 4.41: LSTM model val and test MAPE for different configurations

as before. This is not necessarily a problem, as what we want to achieve is the prediction of actual prices and these are more closely related to the test dataset. These results seem to indicate that using the MAPE as a loss function can yield better prediction accuracy than using the MSE.

4.4 Final Model

After having experimented with multiple architectures, parameters and configurations, in this section we present the final models with which we were able to achieve the best prediction accuracy.

Building on top of previous experiments with LSTM's parameters, we combined the different indicators of best performance to achieve the most accurate model. As such, we tested the following 8 configurations present in Table 4.4, while using as input features the *rarity*, *price*, *ewma_3*,

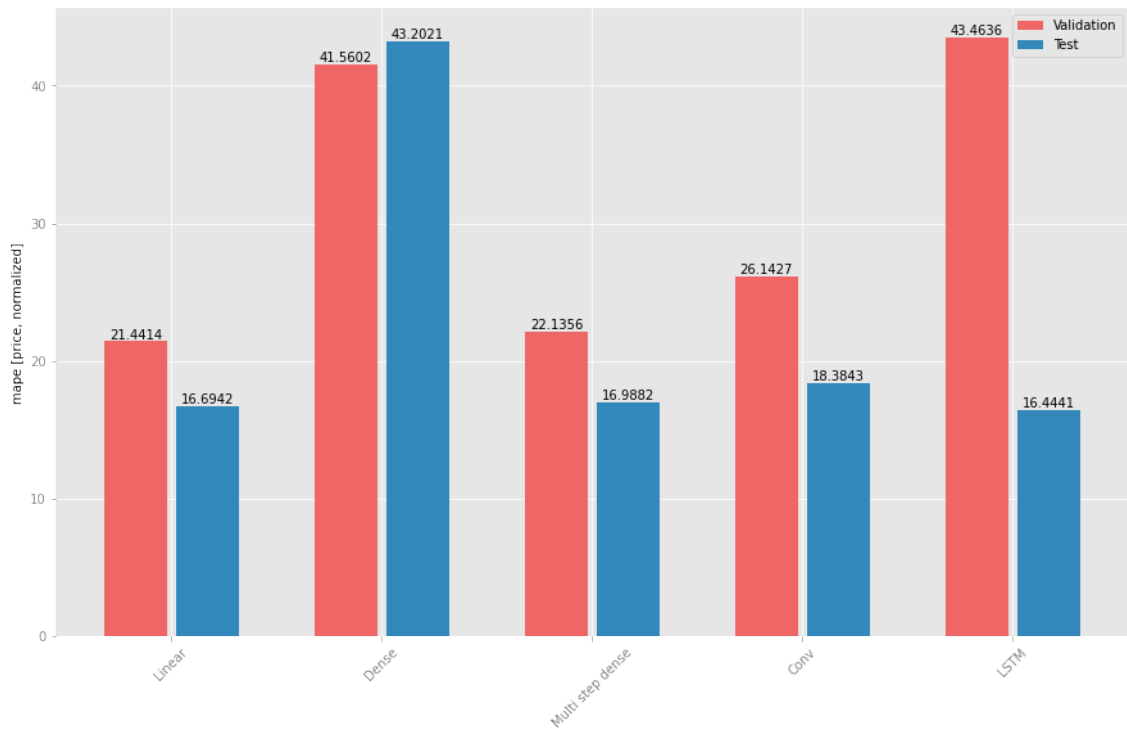


Figure 4.42: MAPE results when using MAPE as the loss function

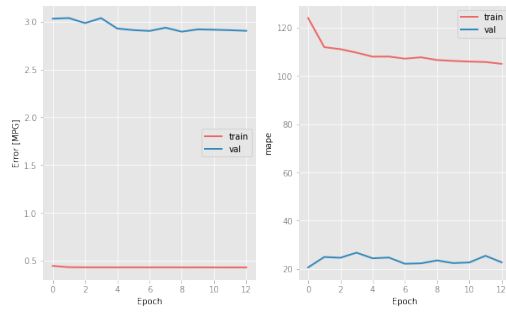
and *ewma_7*, as well as shuffling the data of train, val, and test in separate. We can see the evolution of the loss function over the epochs in Figure 4.43. It is worth nothing that the first 3 configs used MSE as the loss function, and as such display both that value and the MAPE, while the following configurations only have the MAPE value. We achieved the final MAPE results shown in Figure 4.44.

We can quickly see that Configuration 7 achieved the lowest MAPE result so far, for both validation and test sets. The test MAPE was a staggering 9.42%, which we consider to be a really good value. Looking at the loss over the epochs, it steadily decreases until it converges on the target value.

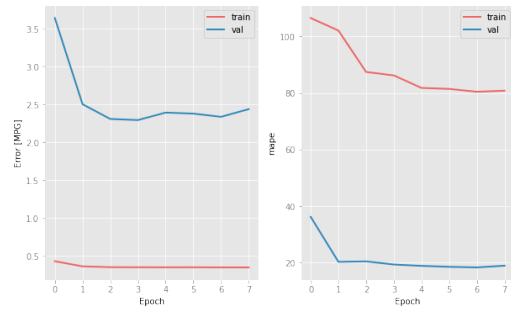
Overall, and comparing with other attempted models, all of these configurations achieved good results. Looking at the other configurations, we see that the model benefits from using the MAPE as the loss function, achieving lower error values in the test set. The best results were achieved when the number of hidden layers was increased, and with a preference for a small window size of 3.

Parameters	C1	C2	C3	C4	C5	C6	C7	C8
window_size	24	3	3	3	3	24	3	3
hidden_layers	32	32	100	32	100	100	150	200
mape_as_loss	False	False	False	True	True	True	True	True

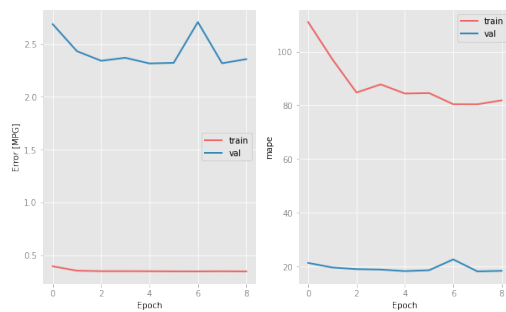
Table 4.4: Final LSTM configurations



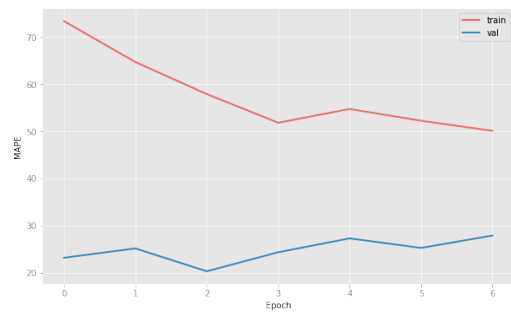
(a) Config 1



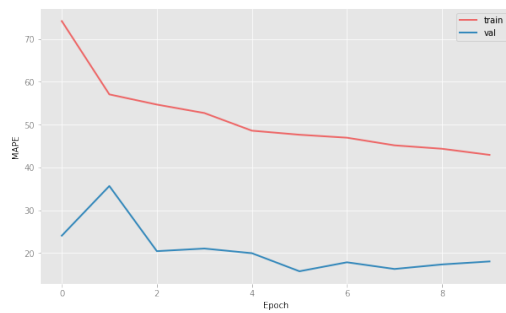
(b) Config 2



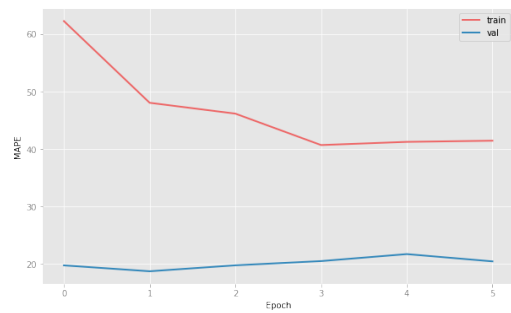
(c) Config 3



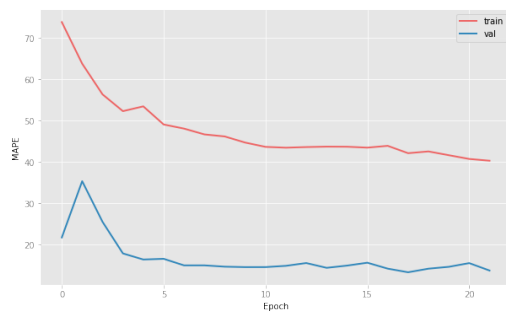
(d) Config 4



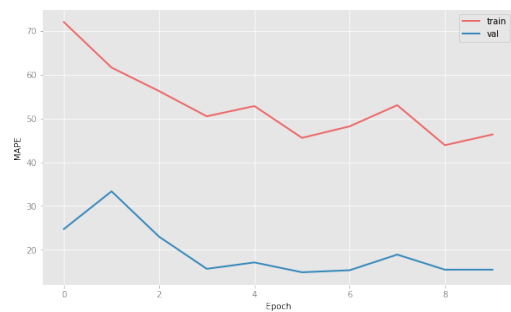
(e) Config 5



(f) Config 6



(g) Config 7



(h) Config 8

Figure 4.43: Loss function over epochs for the final LSTM models

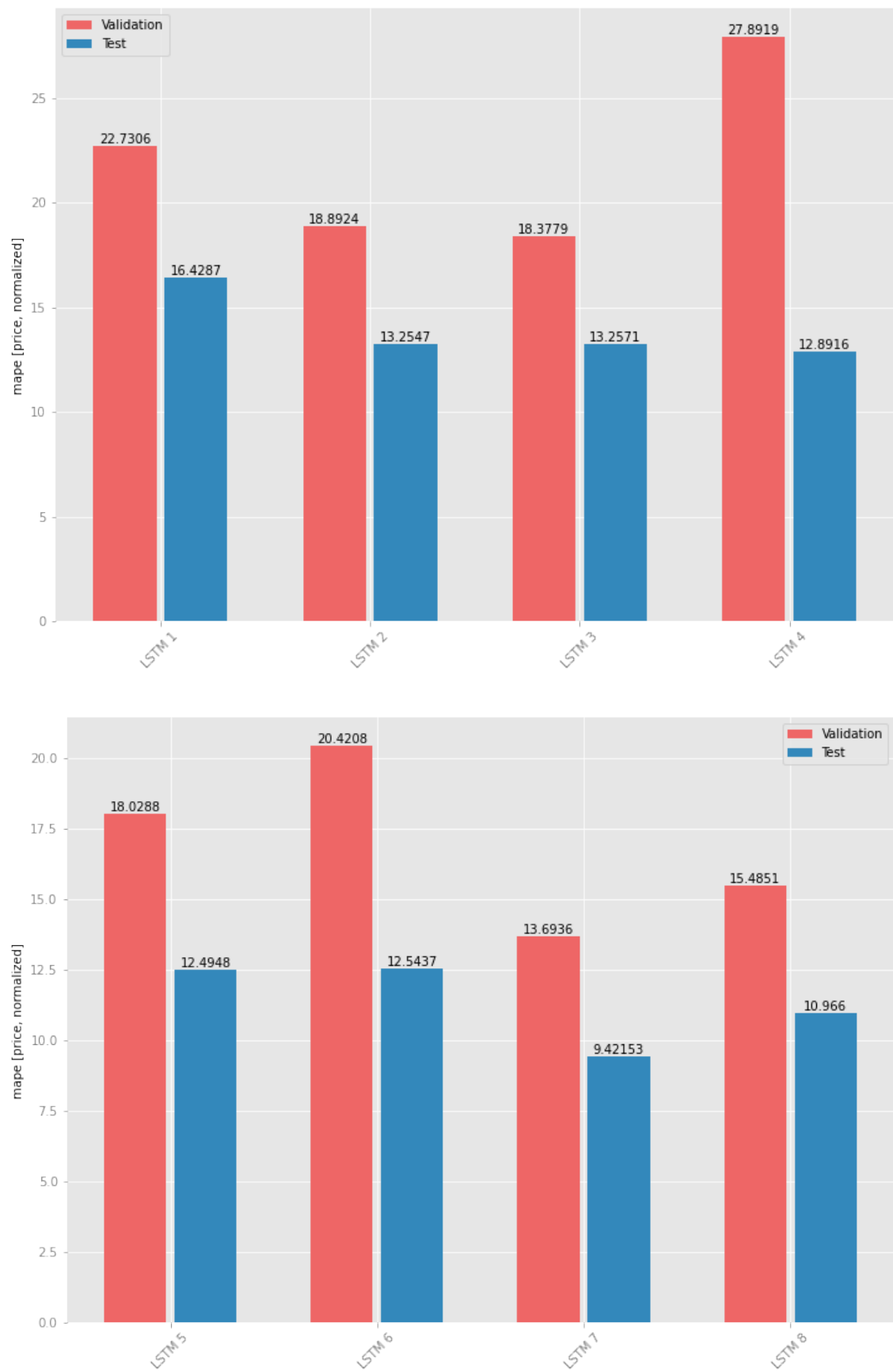


Figure 4.44: MAPE results for the final LSTM models

4.5 Other collections

As an attempt to study the transferability of the model we fine-tuned for the BAYC collection, we applied it to other NFT collections and display the results below.

4.5.1 CryptoPunks

Figure 4.45 displays the achieved MAPE results by the LSTM models' configurations present in Table 4.4.

We can see that the best result it could achieve was 57.3551%, which is a very high error and isn't of much use. Further fine-tuning should be needed to adjust the models to this collection and achieve better results.

4.5.2 Doodles

Figure 4.46 displays the achieved MAPE results by the LSTM models' configurations present in Table 4.4.

In this case, the models were able to achieve very low MAPE values, with Configuration 5 reaching 5.09%. This is great news, as it suggests that there is benefit in applying the learned model from BAYC to other collections with even better results. However, we see that the val MAPE is still considerably high, which might be a bit worrying. This may suggest that the test data used was a period of particular low volatility, and begs the question if it would be reproducible in the future. In any case, all of the attempted configurations achieved a very low MAPE.

4.5.3 Azuki

Figure 4.47 displays the achieved MAPE results by the LSTM models' configurations present in Table 4.4.

The same models used on BAYC were able to achieve a 22.63% MAPE result for the Azuki dataset. While this value is not ideal, it is still a promising result since we spent no effort at fine-tuning it to this specific collection. Surely with some effort this MAPE can be further reduced.

4.5.4 Anime Metaverse

Figure 4.48 displays the achieved MAPE results by the LSTM models' configurations present in Table 4.4.

In this case, the best test MAPE the models achieved was 15.18%. While not as good as the results for BAYC or Doodles, we consider this to be an acceptable result. It is interesting to note that Anime Metaverse is a recent collection and not considered to be blue-chip, with much fewer amounts of data available, 9743 rows to be exact. This seems to suggest the applicability of the same LSTM prediction models to other collections that might not be as well known, increasing the coverage of these models.

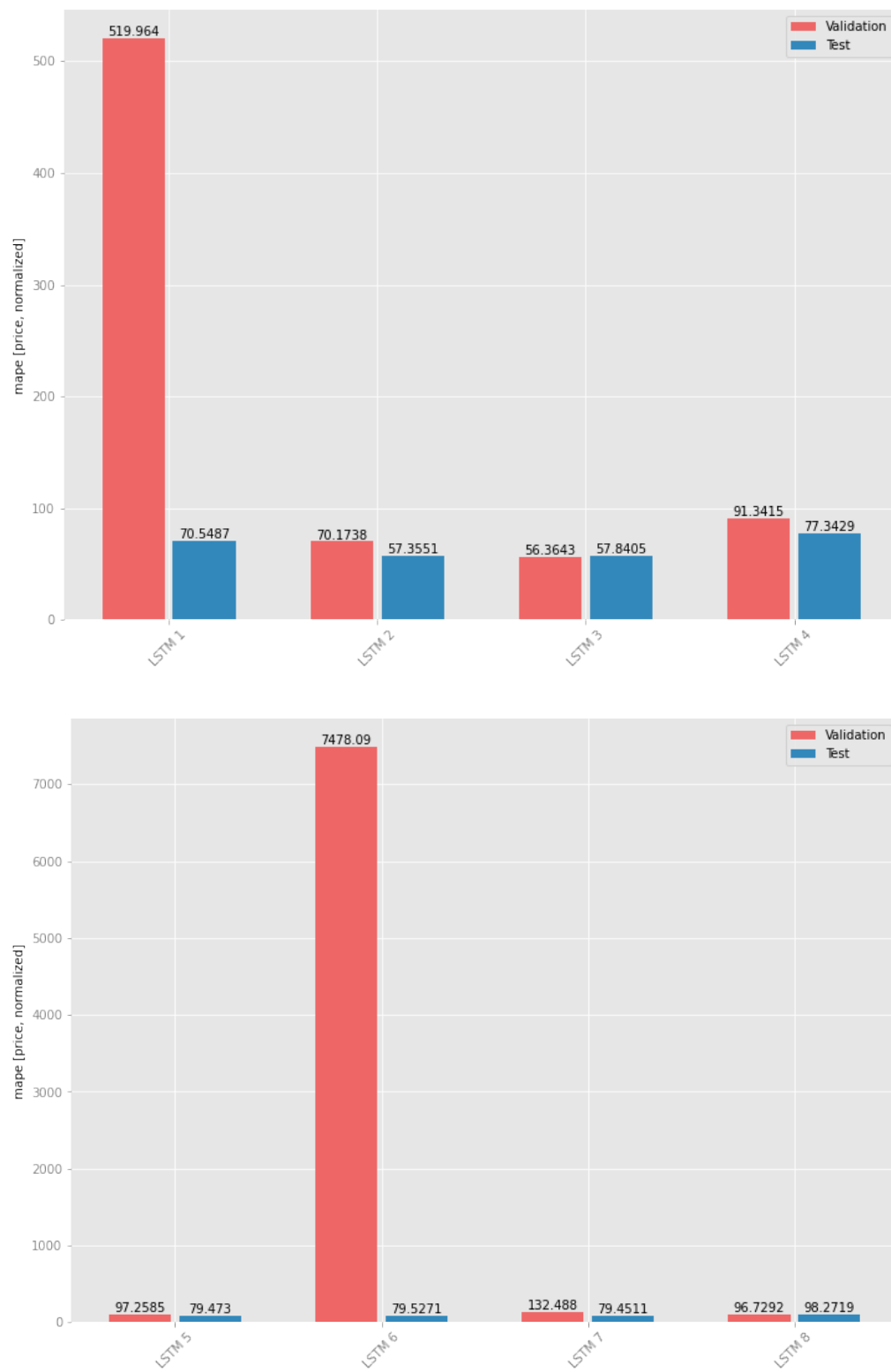


Figure 4.45: CryptoPunks - MAPE results

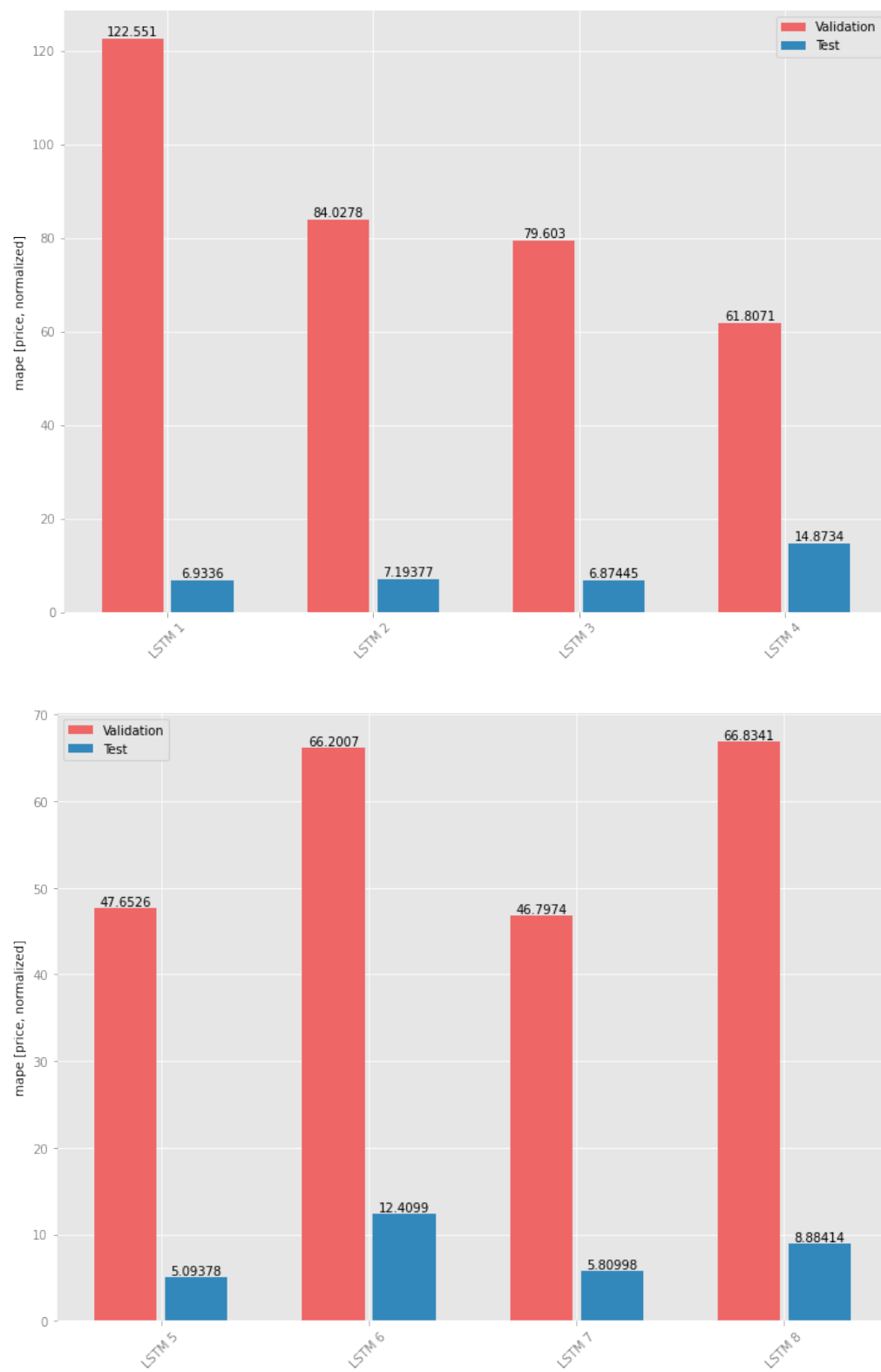


Figure 4.46: Doodles - MAPE results

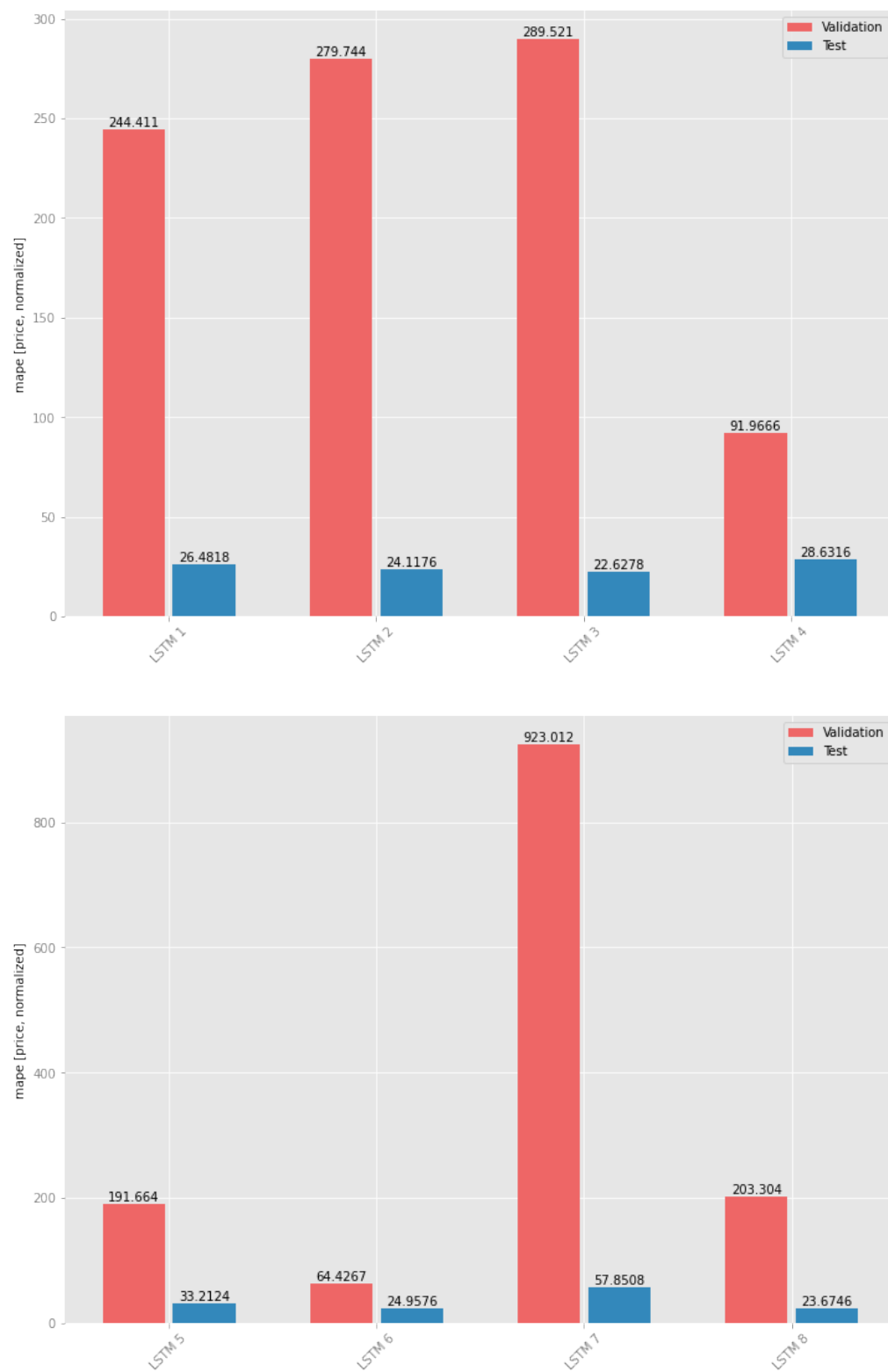


Figure 4.47: Azuki - MAPE results

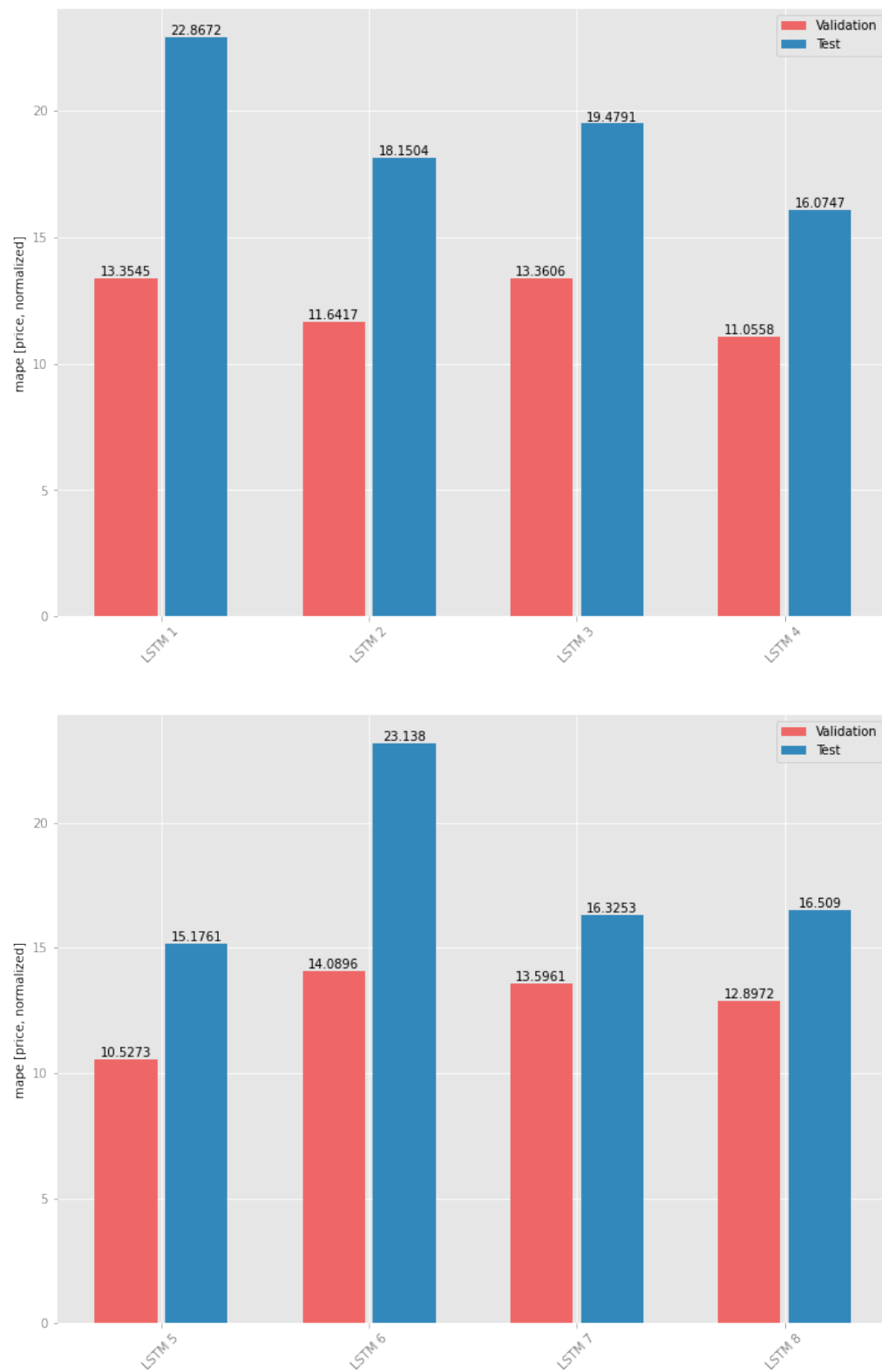


Figure 4.48: Anime Metaverse - MAPE results

4.5.5 Pixelmon

Figure 4.49 displays the achieved MAPE results by the LSTM models' configurations present in Table 4.4.

For Pixelmon, the best test MAPE the models achieved was 12.65%. Like Anime Metaverse, this collection is recent and not considered blue-chip.

4.6 Discussion

Considering the Results and insights presented above in this Chapter, here we elaborate over what was achieved, present interesting points of discussion, and mention short-comings of our work.

Overall, we focused on the price prediction for the Bored Ape Yacht Club collection, trying out multiple models, configurations, and choices of parameters. This effort allowed us to achieve a great MAPE result of 9.42%, meaning that our predictions are, on average, 9.42% away from the real values. We believe this to be a great MAPE value, especially considering the volatility and amount of outliers in the data. However, it is worth noting that it took considerable optimization and fine-tuning, and this way of creating a pricing model is expensive and time-consuming.

We used as a comparison basis a Baseline model that considered the last sale of the collection as the prediction, achieving a MAPE of 21.69%. This value is unexpectedly high and comes to show how volatile the price of these NFT collections is, where even consecutive sales differ, on average, by 22%. We were able to improve upon this value with many different models, including Linear Regression, Dense Neural Network models, Dense NN models that considered the input as a timeseries, CNNs, and RNNs. The most promising models proved to be the ones based on Recurrent Neural Networks, especially LSTMs, and so these were the ones with which we explored further.

It is worth noting how the train MAPE often is the highest one, followed by the val MAPE and then the test MAPE. This seems to be counter-intuitive, as one would expect the model to adapt to the train data and have the lowest error there. This is explained by how we kept the sequential order of the data, in chronological order, and that the train and val sets are larger than the train one, on a 70%, 20%, 10% split, respectively. As such, and due to the high volatility of the data, it makes sense that the model displays a larger MAPE value for the bigger windows of data. Additionally, the initial part of the data catches the launch of the project, that is always extremely volatile and may take some time until it stabilizes. The test set is able to capture a part of the data where it is already more stable. What is important to note is that, after training, on the remaining 10% of data used for test the model provides accurate predictions.

As mentioned before, by including metadata and rarity information of NFTs in the process, this problem isn't a straightforward timeseries forecasting. We opted for experimenting with Deep Learning timeseries models, but it is important to understand what is happening behind the scenes. When we use windows of data as input to predict the next step, these would typically be composed of data that is time-dependent, such as the price of the NFT. However, we have included the Rarity

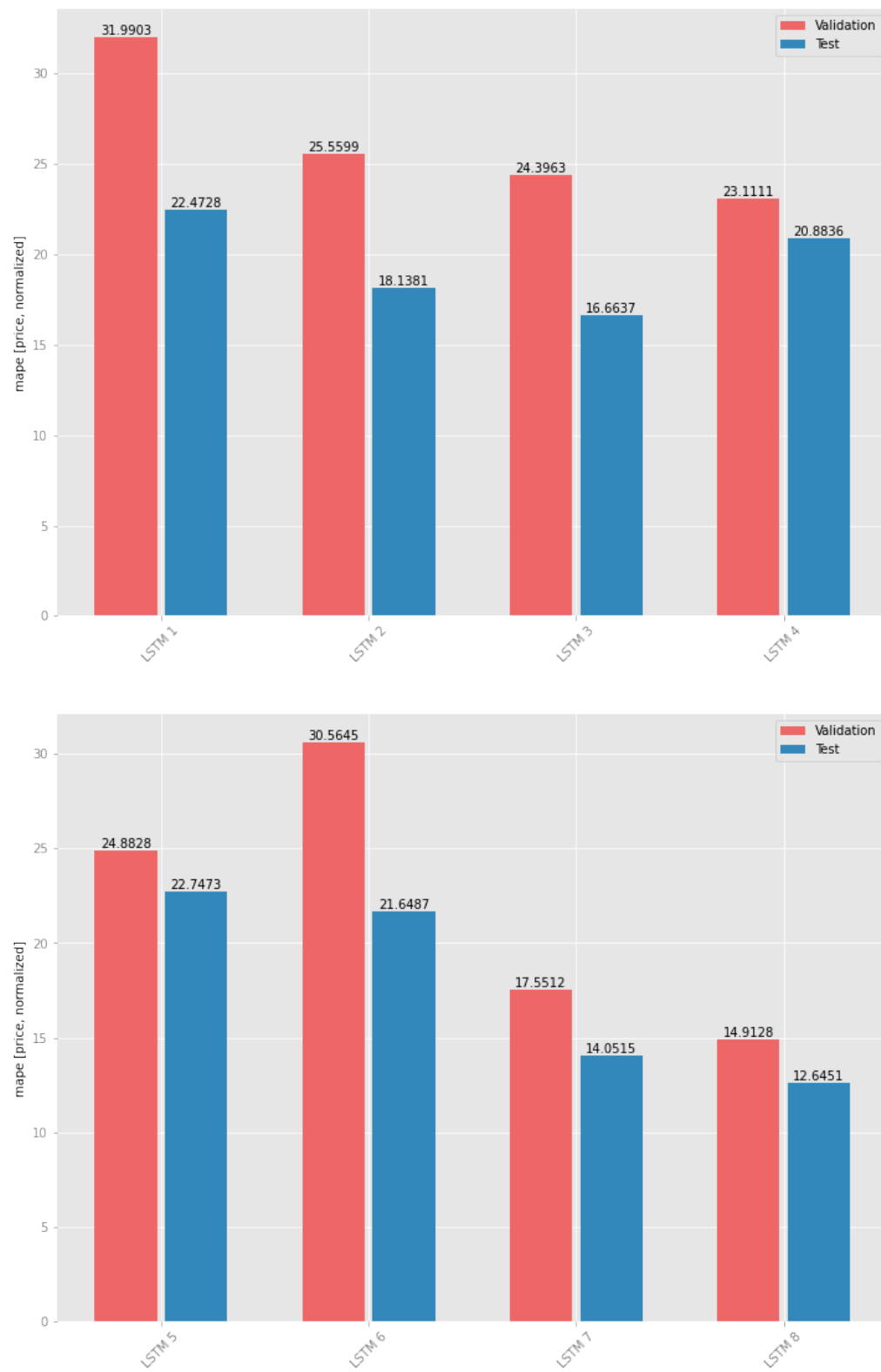


Figure 4.49: Pixelmon - MAPE results

Score in them, and this value is fixed for each NFT token, not dependent on time. As such, the model might be learning relationships between the sequence of tokens that it finds on the data windows. For example, that if Token 1, 2 and 3 were sold in sequence, then the next sale will be higher/lower. However, this factor shouldn't matter in the decision.

Considering the traits of the NFTs, the Rarity Score allows us to get a ranking of NFTs by the frequency of the traits they are composed of, and we have seen by the results that including it in the model helps it achieve a higher accuracy. However, in reality it is often the case that this ranking is not followed, with specific traits values that are valued more than others, or by combinations of trait values that by itself may not be rare, but when combined they are. We are assuming that the less frequent traits will always be more valuable, as well as assuming that different traits with the same frequency are priced the same. But this is definitely not the case, as the price dynamics are subjective and dictated by visual taste and personal preferences. As such, this method of incorporating the NFTs traits' data has flaws and fails to capture important scenarios. A better way to incorporate the traits data should be explored.

We faced concerns about the amount of available data, as it is not very significant and such Deep Learning models typically require a vast amount of data to train. However, it is worth noting that this is a problem that everyone tackling the pricing of NFTs faces. If there is not enough data about each NFT collection or token, other options should be considered to make use of what we have, such as decomposing NFTs in its traits and making a trait-based prediction model, or trying to use data across different collections and find patterns, or study ways to augment the data.

As for input data, we considered creating new features from the raw data in a process of feature engineering. In the end, we ended up not using many of them, as they were found to not improve the models' accuracy. A better study on these should be made, trying to come up different features that are not related to each other. For example, we tried to use features such as *timestamp* and *eth_price* together, but these are co-related as the ETH price is time-dependent.

We tested the applicability of the developed models to other collections, with satisfactory results. While for CryptoPunks the results do not look promising, we were able to achieve very low MAPE values for Doodles, as well as satisfactory results for Azuki. Without any effort at fine-tuning, it is interesting to see how the same model is able to provide acceptable results. This was also applied to non-blue chip collections, Anime Metaverse and Pixelmon. These are characterized by being more recent, without as much sales data, and it is interesting to see that it still achieves a low MAPE value.

Chapter 5

Conclusion

5.1 Conclusion

NFTs have seen a significant growth in the last few years and are quickly becoming mainstream and being used for a multitude of use cases. Their potential as collectibles has been widely recognized and has created a new asset class at the intersection of cryptocurrencies and traditional art. As the NFT market continues to evolve, it is important to develop tools to help understand and predict the prices of NFTs, unlocking key financial use cases in the digital world. Basing the decision on a single factor, such as the last trading price, is short-sighted and doesn't provide owners justice to the value of their NFTs. Existing platforms for NFT liquidity, such as arcade.xyz or nftfi.com would benefit from a valuation algorithm, promoting fairness between users and enabling the possibility of automating the lending process.

In this thesis, we explored the use of Machine Learning techniques for the prediction of NFT values, focusing on understanding the factors behind NFT price movements, how to model this problem, and how to automate the prediction task with a reliable measure of accuracy. By analysing the sales history of NFTs, together with their metadata, with a focus on blue-chip collections such as Bored Ape Yacht Club, we have shown that there are multiple factors that have a significant impact on the pricing, such as the rarity, uniqueness, and previous sales prices.

Unlike stock or cryptocurrencies' price prediction, this problem is not a straightforward time-series forecasting, as it is necessary to incorporate non-time dependant data in the model such as the metadata and rarity of individual NFTs. This characteristic makes it harder to use typical time-series solutions such as ARIMA or Prophet models, that have been widely applied to other prediction problems. Additionally, due to the novelty of NFTs, the amount of data available is much reduced than other predictions problems, making the data-hungry Neural Network models have more trouble learning.

By training and evaluating multiple machine learning algorithms, we have shown that these can be applied to accurately predict NFT prices. While an intuitive method that uses the last sale value of the collection presents a MAPE of 21.69% for the BAYC collection, our models were able to lower this error to 9.42%. Our results indicate that Recurrent Neural Network based

models provide the highest accuracy for NFT price prediction, taking advantage of the sequential nature of the data. However, these models require careful and thorough fine tuning of the parameters to achieve a desirable level of accuracy, and the reality of one collection is not always fully transferable to others, each requiring their own fine-tuning to achieve good prediction accuracy.

This work demonstrates the feasibility of Machine Learning for this task, and it serves as a foundation for future investigation on the topic of NFT pricing. Its results are an important contribution for researchers that decide to tackle this problem, for investors looking to better understand what drives the value of NFTs, and for the rapidly growing number of platforms that rely on NFT pricing for their services and need to develop their own models. such is the case of Anchorage Digital, and we hope that the extensive analysis of state of the art and attempted techniques Overall, this work is a step forward in making NFT pricing more understandable, transparent, and promoting a healthier NFT ecosystem.

5.2 Future Work

Going forward with the pricing of NFTs, we believe this task would benefit from including additional information in the models about the collections and the tokens themselves. Some interesting factors to incorporate could be the floor price of the collection at each time step, that signals how much the current owners are willing to sell it for, or the total number of bids of each NFT, that gives a signal of how popular a specific token is.

As identified in Chapter 4, the incorporation of the Traits' data by calculation of a Rarity Score has flaws and fails to capture important relationships between NFTs in the same collection and their values. A better way of encoding the traits data should be explored, that accounts for each trait Type and possible values, allowing the model to understand the importance of each individual trait value and how the combination of an NFT's traits results in the price. Such a solution would allow the model to decompose each NFT in its parts, and be able to use sales data from different NFTs that share some of the same traits, and not necessarily all, to learn. This would significantly help with the problem of not having enough historical data, leveraging similarities between tokens. It would also greatly help in prediction the price of tokens with no sales history at all, since we would be able to deduce it by the combination of its traits individually.

Multiple studies in Chapter 2 have found that market sentiment data, gathered from social medias such as twitter, are a helpful factor for market movements and price trends of financial assets. This could be achieved by analysing social media posts, the language and key words used, the number of followers and evolution over time, engagement, among other factors. This would help identify situations of increased activity and *hype*, where prices tend to go up, as well as negative emotions that tend to drive the price down.

While it was suggested that visual features are not as relevant for this prediction task, incorporating them in the decision process, together with the other already identified features, could

increase the models performance by some amount. These visual features can be identified by using a Convolutional Neural Network on the NFT image, converting it into an input vector that can be fed into the predictor model.

Since NFTs are highly volatile, a model like the ones we developed should be updated regularly with new data to ensure its accuracy is maintained. It should be trained over time, automatically incorporating data from new sales, twitter updates and followers, etc. Prices can suddenly increase or decrease and it should be ready to deal with such situations.

It would also be interesting to further study in which situations the current models fail to capture the real price. Does it happen in situations where the price suddenly rises/drops? Is it related with inaccurate modeling of the rarity of the NFT? Or is the problem somewhere else?

As for additional methods to try, one could attempt to use [AutoML](#)¹ to automate the processes of selecting regression algorithm, hyperparameter tuning of models, iterative modeling, and model evaluation. This would allow us to try many combinations without requiring manual effort, and quickly understand what configurations work for the data that we have.

Another approach to tackle the NFT pricing problem could be based on [XGBoost](#)², an open-source implementation of the gradient boosted trees algorithm. Has it has proven to be effective for multiple regression problems, it could be interesting to apply it to this one as well.

We also suggest experimenting with [Shapley Values](#)³ to better understand which features play out a significant role in the Deep Learning models. This will promote an higher explainability of the model, and allow to extract useful insights to improve the accuracy even further.

As a step forward, it would be interesting to dive deeper in creating a generic pricing model that is capable of calculating the value of any NFT. Such a model should be able to harness the data from different collections and extract insights between them, solving the problem of insufficient data. Additionally, it would be a more scalable solution, not requiring to train and fine-tune a new model for each collection, that is an expensive and time-consuming task. However, this presents itself as a very challenging problem, as there are huge discrepancies between values of different collections, each one has their own set of Traits and values, and there's many subjective factors to consider. For this reason, we would suggest the creation of such a generic model as a Baseline solution for the pricing of NFTs and to allow for new collections to be automatically included. However, on top of this model one can develop specific, collection-level models that can achieve higher accuracy and replace the usage of the general one when they are implemented.

¹automl.org

²xgboost.readthedocs.io/en/stable

³christophm.github.io/interpretable-ml-book/shapley.html

References

- Jethin Abraham, Daniel Higdon, John Nelson, and Juan Ibarra. Stochastic neural networks for cryptocurrency price prediction. *SMU Data Science Review*, 1(3), 2018. URL <https://scholar.smu.edu/datasciencereview/vol1/iss3/1>.
- Alchemy. Ethereum statistics (2022), 2022. Available at <https://www.alchemy.com/overviews/ethereum-statistics>, last accessed in February 20, 2023.
- Adebiyi A. Ariyo, Adewumi O. Adewumi, and Charles K. Ayo. Comparison of arima and artificial neural networks models for stock price prediction. *Journal of Applied Mathematics*, 2014: 7, 2014a. doi: <https://doi.org/10.1155/2014/614342>. URL <https://www.hindawi.com/journals/jam/2014/614342/>.
- Adebiyi A. Ariyo, Adewumi O. Adewumi, and Charles K. Ayo. Stock price prediction using the arima model. In *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, pages 106–112, 2014b. doi: 10.1109/UKSim.2014.67.
- Artsy. 4 key challenges preventing the art industry from attracting new buyers, 2019. Available at <https://www.artsy.net/article/artsy-editorial-key-challenges-preventing-art-industry-attracting-new-buyers>, last accessed in February 20, 2023.
- Mathieu Aubry, Roman Kraeussl, Gustavo Manso, and Christophe Spaenjers. Machines and masterpieces: Predicting prices in the art auction market. *Journal of Finance, Forthcoming*, 2019a. doi: <https://doi.org/10.2139/ssrn.3347175>.
- Mathieu Aubry, Roman Kraeussl, Gustavo Manso, and Christophe Spaenjers. Biased auctioneers. *Journal of Finance, Forthcoming*, 2019b. doi: <https://dx.doi.org/10.2139/ssrn.3347175>.
- Joseph Bamidele Awotunde, Roseline Oluwaseun Ogundokun, Rasheed Gbenga Jimoh, Sanjay Misra, and Taye Oladele Aro. *Machine Learning Algorithm for Cryptocurrencies Price Prediction*, pages 421–447. Springer International Publishing, Cham, 2021. ISBN 978-3-030-72236-4. doi: 10.1007/978-3-030-72236-4_17. URL https://doi.org/10.1007/978-3-030-72236-4_17.
- Jason Bailey. Can machine learning predict the price of art at auction?, 2020. Available at <https://hdrs.mitpress.mit.edu/pub/lvdc2z91/release/3>, last accessed in February 20, 2023.
- Jens Beckert and Jörg Rössel. The price of art. *European Societies*, 15(2):178–195, 2013. doi: 10.1080/14616696.2013.767923. URL <https://doi.org/10.1080/14616696.2013.767923>.

Jason Brownlee. What is deep learning?, 2019. Available at <https://machinelearningmastery.com/what-is-deep-learning/>, last accessed in February 20, 2023.

Vitalik Buterin. A next-generation smart contract and decentralized application platform, 2014. Available at <https://ethereum.org/en/whitepaper/>, last accessed in February 20, 2023.

Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL <http://arxiv.org/abs/1406.1078>.

Mitchell Clark. Nfts, explained, 2022. Available at <https://www.theverge.com/22310188/nft-explainer-what-is-blockchain-crypto-art-faq>, last accessed in February 20, 2023.

DeHive. How to solve the blockchain data query issue and what the graph service has to say, 2022. Available at <https://coinmarketcap.com/alexandria/article/how-to-solve-the-blockchain-data-query-issue-and-what-the-graph-service-has-t>, last accessed in February 20, 2023.

Michael Dowling. Is non-fungible token pricing driven by cryptocurrencies? *Finance Research Letters*, 44:102097, 2022. ISSN 1544-6123. doi: <https://doi.org/10.1016/j.frl.2021.102097>. URL <https://www.sciencedirect.com/science/article/pii/S1544612321001781>.

William Entriken, Dieter Shirley, Jacob Evans, and Nastassia Sachs. Eip-721: Non-fungible token standard, 2018. Available at <https://eips.ethereum.org/EIPS/eip-721>, last accessed in July 2022.

Jake Frankenfield. Smart contracts, 2022. Available at <https://www.investopedia.com/terms/s/smart-contracts.asp>, last accessed in February 20, 2023.

Ahiza García-Hodges and Diana Dasrath. What’s an nft? behind the boom for digital collectibles, 2021. Available at <https://www.nbcnews.com/tech/tech-news/nft-boom-digital-collectibles-rcna430>, last accessed in February 20, 2023.

Adam Hayes. Autoregressive integrated moving average (arima) prediction model, 2022. Available at <https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp>, last accessed in February 20, 2023.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. URL <https://www.bioinf.jku.at/publications/older/2604.pdf>.

Florian Horky, Carolina Rachel, and Jarko Fidrmuc. Price determinants of non-fungible tokens in the digital art market. *Finance Research Letters*, 48:103007, 2022. ISSN 1544-6123. doi: <https://doi.org/10.1016/j.frl.2022.103007>. URL <https://www.sciencedirect.com/science/article/pii/S1544612322002495>.

- IBM. What are recurrent neural networks?, 2020. Available at <https://www.ibm.com/cloud/learn/recurrent-neural-networks>, last accessed in February 20, 2023.
- IBM. What are smart contracts on blockchain?, 2022. Available at <https://www.ibm.com/uk-en/topics/smart-contracts>, last accessed in February 20, 2023.
- Patel Jay, Vasu Kalariya, Pushpendra Parmar, Sudeep Tanwar, Neeraj Kumar, and Mamoun Alazab. Stochastic neural networks for cryptocurrency price prediction. *IEEE Access*, 8:82804–82818, 2020. doi: 10.1109/ACCESS.2020.2990659.
- Zhigang Jin, Yang Yang, and Yuhong Liu. Stock closing price prediction based on sentiment analysis and lstm. *Neural Comput & Applic*, 32:9713–9729, 2020. doi: <https://doi.org/10.1007/s00521-019-04504-2>. URL <https://link.springer.com/article/10.1007/s00521-019-04504-2>.
- Daria Khritonenkova. Nft rarity greatly impacts market value - but how do you check it?, 2022. Available at <https://bdc.consulting/insights/nft/nft-rarity-greatly-impacts-market-value>, last accessed in February 20, 2023.
- Anndy Lian. 27 stats about nfts in 2022 – who are the big winners?, 2022. Available at <https://cryptoslate.com/27-stats-about-nfts-in-2022-who-are-the-big-winners/>, last accessed in February 20, 2023.
- Kristof Lommers. A framework for asset pricing models in nfts, 2022. Available at SSRN: <https://ssrn.com/abstract=4248399> or <http://dx.doi.org/10.2139/ssrn.4248399>.
- Wenjie Lu, Jiazheng Li, Jingyang Wang, and Lele Qin. A cnn-bilstm-am method for stock price prediction. *Neural Comput & Applic*, 33:4741–4753, 2021. doi: <https://doi.org/10.1007/s00521-020-05532-z>. URL <https://link.springer.com/article/10.1007/s00521-020-05532-z>.
- Junliang Luo, Yongzheng Jia, and Xue Liu. Understanding nft price moves through social media keywords analysis, 2022. URL <https://arxiv.org/abs/2209.07706>.
- Amin Mekacher, Alberto Bracci, Matthieu Nadini, Mauro Martino, Laura Alessandretti, Luca Maria Aiello, and Andrea Baronchelli. Heterogeneous rarity patterns drive price dynamics in nft collections. *Sci Rep*, 12:13890, 2022. doi: <https://doi.org/10.1038/s41598-022-17922-5>. URL <https://www.nature.com/articles/s41598-022-17922-5>.
- Diogo Mónica and Nathan Mccauley. Anchorage digital started with custody. five years later—custody is just the start., 2022. Available at <https://www.anchorage.com/insights/anchorage-digital-started-with-custody-five-years-later-custody-is-just-the>, last accessed in February 20, 2023.
- M. Nadini, L. Alessandretti, and F. Di Giacinto et al. Mapping the nft revolution: market trends, trade networks, and visual features. *Scientific Reports*, 11, 2021. doi: <https://doi.org/10.1038/s41598-021-00053-8>.

Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. May 2008. URL <http://www.bitcoin.org/bitcoin.pdf>.

NFTBank.ai. How to accurately estimate the price of individual nfts, 2022. Available at <https://medium.com/nftbank-ai/how-to-accurately-estimate-the-price-of-individual-nfts-part-2-ac73e88aa3ad>, last accessed in February 20, 2023.

Mohil Maheshkumar Patel, Sudeep Tanwar, Rajesh Gupta, and Neeraj Kumar. A deep learning-based cryptocurrency price prediction scheme for financial institutions. *Journal of Information Security and Applications*, 55:102583, 2020. ISSN 2214-2126. doi: <https://doi.org/10.1016/j.jisa.2020.102583>. URL <https://www.sciencedirect.com/science/article/pii/S2214212620307535>.

Benjamin Pimentel. <https://www.protocol.com/fintech/crypto-nft-anchorage-blockchain-banking>, 2021. Available at <https://www.protocol.com/fintech/crypto-nft-anchorage-blockchain-banking>, last accessed in February 20, 2023.

Ryker. How to value items in nft projects? — part 1, 2021. Available at <https://medium.com/nftbank-ai/how-to-value-items-in-nft-projects-part-1-a87be8bcb21d>, last accessed in February 20, 2023.

Sebastian Smee. Will nfts transform the art world? are they even art?, 2021. Available at <https://www.washingtonpost.com/arts-entertainment/2021/12/18/nft-art-faq/>, last accessed in February 20, 2023.

Upshot. An efficient price mechanism for nfts - upshothq, 2021. Available at <https://mirror.xyz/0x82FE4757D134a56BFC7968A0f0d1635345053104/rH7FgAjOaoWfGHFvtf-v0JrV0sZyJ31tpLRp6ZqXB-4>, last accessed in February 20, 2023.

Upshot.xyz. How real-time nft pricing can increase nft adoption, 2021a. Available at <https://mirror.xyz/0x82FE4757D134a56BFC7968A0f0d1635345053104/5DXSmgQR1irC37Ke8s7cSxdT27TyN8sWvj-0oL3WzXM>, last accessed in February 20, 2023.

Upshot.xyz. Taking nft pricing to the next level with machine learning, 2021b. Available at <https://mirror.xyz/0x82FE4757D134a56BFC7968A0f0d1635345053104/dHNbc5bE8hnXB3xtjZInZ7TApKrE-S8QcexXrcj2QPo>, last accessed in February 20, 2023.

Upshot.xyz. Determining what matters for nft valuation, 2021c. Available at https://mirror.xyz/0x82FE4757D134a56BFC7968A0f0d1635345053104/hYImuHwz3_Pu-pYxVioEbjguRKST-eeN_8eBKNgzQls, last accessed in February 20, 2023.

Upshot.xyz. Explaining nft pricing with machine learning, 2022. Available at <https://mirror.xyz/0x82FE4757D134a56BFC7968A0f0d1635345053104/GFYBIKWQxYJQ18OUI4dKf-ha2xpYe1MsetJEzg-W3Jg>, last accessed in February 20, 2023.

- YUGESH VERMA. 5 conditions when the arima model should be avoided, 2022. Available at <https://analyticsindiamag.com/5-conditions-when-the-arima-model-should-be-avoided/>, last accessed in February 20, 2023.
- Philip B. Weerakody, Kok Wai Wong, Guanjin Wang, and Wendell Ela. A review of irregular time series data handling with gated recurrent neural networks. *Neurocomputing*, 441:161–178, 2021. ISSN 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2021.02.046>. URL <https://www.sciencedirect.com/science/article/pii/S0925231221003003>.
- Jason Wise. Opensea statistics 2023: Users, revenue & market size, 2022. Available at <https://earthweb.com/opensea-statistics/>, last accessed in February 20, 2023.
- Tom Worth. Painting2auction: Art price prediction with a siamese cnn and lstm. 2020. URL http://cs230.stanford.edu/projects_fall_2020/reports/55539436.pdf.
- Chenglin Xiao, Weili Xia, and Jijiao Jiang. Stock price forecast based on combined model of ari-ma-ls-svm. *Neural Comput & Applic*, 32:5379–5388, 2020. doi: <https://doi.org/10.1007/s00521-019-04698-5>. URL <https://link.springer.com/article/10.1007/s00521-019-04698-5>.
- Pengfei Yu and Xuesong Yan. Stock price prediction based on deep neural networks. *Neural Comput & Applic*, 32:1609–1628, 2020. doi: <https://doi.org/10.1007/s00521-019-04212-x>. URL <https://link.springer.com/article/10.1007/s00521-019-04212-x>.
- Zharta.io. Zharta in a nutshell, 2022a. Available at <https://medium.com/@ZhartaFinance/what-can-you-do-with-zharta-6a1e14dfe2a6>, last accessed in February 20, 2023.
- Zharta.io. Meet the team — diogo pires — cto & co-founder, 2022b. Available at <https://medium.com/@ZhartaFinance/meet-the-team-diogo-pires-cto-co-founder-548ce8aaf835>, last accessed in February 20, 2023.
- Xi Zheng. Multimodal learning for improved nft price prediction, 2022. Available at <https://conferences.computer.org/icebe/2022/icebe2022-proceedings/Multimodal%20Learning%20for%20Improved%20NFT%20Price%20Prediction.pdf>.