Modelos de Agregação de Previsões

Aplicados à Previsão de Energia Eólica

Por

Liliana Sousa Oliveira

110414016@fep.up.pt

Dissertação de Mestrado em Análise de Dados e Sistemas de Apoio à Decisão.

Orientada por:

João Gama

João Moreira

Setembro de 2014

Agradecimentos

Em primeiro lugar gostaria de agradecer ao meu orientador Prof. João Gama pela ajuda e motivação ao longo da realização do trabalho.

Um grande agradecimento ao meu co-orientador Prof. João Moreira pela disponibilidade demonstrada desde o início.

Não me posso esquecer da empresa Prewind que me forneceu os dados para que fosse possível a realização deste trabalho.

Gostaria também de agradecer às minhas amigas: Carolina Vieira, Vera Silva, Maria João Lima e Paula Santos pela constante troca de ideias.

Por último, mas não menos importante, à minha família pelo apoio constante.

Palavras-chave: Métodos de Previsão, *Tracking the best expert*, *Weighted Majority Algorithm*, *Ensemble Learning*, Regressão.

Resumo

No domínio da inteligência artificial e considerando o permanente estado de mudança no ambiente social e económico, um grande número de trabalhos têm proposto a combinação de múltiplos modelos de previsão para a conceção de sistemas com alto desempenho na classificação de padrões. Este crescente interesse na combinação de múltiplos modelos provém do reconhecimento que a abordagem focada na escolha do melhor modelo individual tem sérios inconvenientes.

É neste sentido que surge este trabalho, tendo o objetivo de verificar se a combinação de modelos de previsão é capaz de prever com maior precisão a produção de energia eólica, em comparação com cada modelo individualmente. Para tal, é feita uma análise de métodos de integração e de modelos de atualização de pesos já existentes. Depois de uma breve descrição da base de dados fornecida, são analisados os resultados obtidos com os métodos de atualização de pesos utilizados na integração dos múltiplos modelos. Como existe um modelo que prevê melhor que os restantes em grande parte do período considerado (tendo o erro médio absoluto mais baixo), os modelos de atualização de pesos acabam por lhe atribuir um peso igual a 1 a partir de determinado período, sendo que a partir desse momento a previsão do conjunto é igual à previsão desse mesmo modelo. Contudo, foi observado que existe um mês (Maio) em que esse modelo não tem a melhor performance do conjunto de modelos. Deste modo, foram aplicados os modelos de atualização de pesos considerando apenas os dados do mesmo de Maio, sendo que já não foi escolhido o modelo referido anteriormente.

Por fim, serão apresentadas as conclusões a este trabalho e possíveis trabalhos futuros.

Abstract

In the field of artificial intelligence and considering the ongoing state of change in social and economic environment, a large number of papers have proposed the combination of multiple prediction models for designing systems with high performance in pattern classification. This growing interest in combining multiple models comes from the recognition that the approach focused on choosing the best individual model has serious drawbacks.

This is why this work arises, with the aim of verifying whether the combination of forecasting models is able to predict more accurately the production of wind energy compared with each model individually. This requires an analysis of integration methods and models for updating existing weights is done. After a brief description of the database provided, the results obtained with the methods of updating the weights used in the integration of multiple models are analyzed. As there is a model that best predicts the remaining large part of the period considered (having the lowest mean absolute error), the models for updating weights eventually assign it a weight equal to 1 after a certain period, and from that moment the weather is set equal to the prediction of the same model. However, it was observed that there is a Month (May) in this model, that is not the best performance of all models. Thus, the model update weights considering only data from the same May been applied, and has not already been chosen model referred to above.

Finally, the conclusions of this work and possible future works are presented.

Índice

Índice de Ilustrações

1. Introdução

Atualmente, a energia eólica é vista como uma das fontes de energia renovável mais promissora. Esta fonte de energia tem registado nos últimos anos uma evolução assinalável (Castro 2013).

A Prewind surgiu com base no trabalho desenvolvido por um projeto português de I&D financiado por um consórcio chamado EPREV, constituído por promotores de parques eólicos, com o objetivo principal de desenvolver modelos de previsão de potência de parques eólicos e operacionalizar um sistema de previsão da produção de parques eólicos. Este trabalho surge de uma proposta de tema feita pela Prewind, sendo que os dados foram fornecidos pela mesma. Foram assim fornecidas as previsões de 12 modelos de previsão e os valores reais observados durante o período de um ano. A escala temporal é a hora.

O objetivo é então criar uma combinação de modelos de previsão capaz de prever com maior precisão a produção de energia eólica. A previsão da energia eólica é importante na medida em que serve de apoio à gestão dos sistemas elétricos de energia e na gestão dos congestionamentos de rede. Quanto melhor for a previsão, maior é o controlo de grupos de reserva de forma a compensar a variabilidade do recurso eólico para definir estratégias de armazenamento de energia. Deste modo, torna-se de extrema importância reduzir o erro de previsão da produção de energia eólica.

Existindo já na literatura um vasto leque de trabalhos sobre a combinação de múltiplos modelos, assiste-se a um crescente interesse pelo tema. Este facto deve-se principalmente ao reconhecimento que a antiga abordagem focada na escolha do melhor modelo individual tem sérios inconvenientes, pois o facto de um modelo ter um menor erro médio de previsão, não significa que não haja períodos em que este não tem a melhor performance do conjunto.

Neste trabalho será feita primeiramente uma revisão bibliográfica sobre a combinação de múltiplos modelos. Começa-se por um estudo sobre a aprendizagem de múltiplos modelos que segue de perto o trabalho feito por Mendes-Moreira, Soares et al. (2012). Este capítulo é de extrema importância pois desenha as bases do modelo que

será desenvolvido, como por exemplo, o que é uma regressão, como pode ser calculado o erro de uma previsão, como funciona a aprendizagem e a integração de múltiplos modelos e quais os modelos de integração dinâmica. De seguida, é feita uma introdução à aprendizagem online para posteriormente serem apresentados os algoritmos "*weighted majority algorithm*" e "*tracking the best expert*".

No capítulo seguinte (secção 3) é feita uma análise aos dados fornecidos pela empresa Prewind. Serão analisados os dados reais e as previsões feitas pelos 12 modelos fornecidos (dados dizem respeito ao ano de 2012).

Depois de analisar os dados, serão apresentados os resultados da combinação dos múltiplos modelos aplicados à previsão de energia eólica. Os resultados são apresentados para dois modelos de atualização de pesos e comparadas performances.

Por último, serão feitas conclusões e sugeridos trabalhos futuros.

2. Estado da Arte

2.1. Introdução à Previsão de Energia Eólica

Devido ao crescimento da geração de energia eólica, existe uma necessidade cada vez maior de recorrer a métodos probabilísticos para prever este recurso energético. Ao longo dos últimos anos foram desenvolvidos modelos de Previsão Numérica do Tempo (PNT), que utilizam estimativas sobre o estado atual da atmosfera para um conjunto de modelos determinístico (Thorarinsdottir and Gneiting 2008).

A energia eólica apresenta uma dependência em relação à volatilidade do vento. Esta característica representa uma desvantagem da energia eólica comparada com a eletricidade convencional (Giebel 2003). A eletricidade produzida pela energia eólica é, portanto, muito volátil e variável, podendo sofrer grandes variações em menos de uma hora, de hora para hora, diariamente, semanalmente ou até sazonalmente. Sendo assim, foram já desenvolvidos muitos métodos de previsão de energia eólica para poder diminuir o erro de previsão da eletricidade produzida por esta fonte de energia. Devido ao facto da geração e consumo instantâneo de eletricidade dever manter uma certa estabilidade, a variabilidade da energia eólica pode representar um desafio quando grande quantidade de energia é incorporada no sistema. Deste modo, mudanças súbitas e rápidas de energia eólica (eventos de rampa) são um dos maiores desafios na previsão da energia eólica (Ferreira, Gama et al. 2010).

Giebel (2003) considera que os modelos de previsão de energia eólica podem ser classificados de uma forma geral, em modelos que utilizam Previsão Numérica do Tempo (PNT) ou não. Normalmente, modelos que utilizam PNT apresentam uma melhor performance para previsões 3-6 horas. Este autor ainda distingue as abordagens em físicas e estatísticas, sendo que alguns modelos usam uma combinação das duas. As abordagens físicas usam considerações físicas para chegar o mais próximo possível do valor da velocidade do vento, para depois ser usado um modelo estatístico para previsão da energia eólica. A abordagem estatística tenta encontrar relações entre a riqueza das variáveis estatísticas e dados de energia medidos online. Na abordagem estatística são usados muitas vezes técnicas recursivas como Redes Neuronais Artificiais (RNA). Para

os modelos da abordagem estatística são necessários dados mais antigos para fazer a aprendizagem, apresentando geralmente pouca precisão em previsões de muito curto prazo. No entanto, para horizontes temporais mais longos demonstra uma grande performance o que contrasta com os modelos físicos que apresentam um erro baixo no muito curto prazo mas ficam aquém para horizontes temporais mais longos.

2.2. Aprendizagem de Múltiplos Modelos para Regressão

O termo aprendizagem de múltiplos modelos refere-se ao processo que usa um conjunto de modelos, cada um deles obtido através da aplicação de um processo de aprendizagem a um dado problema. Este conjunto de modelos é integrado de alguma forma a fim de obter a previsão final. Esta definição não só tem em conta conjuntos no contexto de aprendizagem supervisionada (para ambos os problemas: classificação e regressão), mas também aprendizagem não supervisionada. Adicionalmente, não está feita a separação entre combinação e seleção, ao contrário do que muitas outras definições fazem. De acordo com esta, a seleção é um caso especial da combinação em que todos os pesos são zero, exceto para um deles. A maioria dos trabalhos que tem surgido sobre aprendizagem de múltiplos modelos focam em problemas de classificação. Infelizmente, a maioria das técnicas melhor sucedidas em classificação não podem ser diretamente aplicadas em regressão (Mendes-Moreira, Soares et al. 2012).

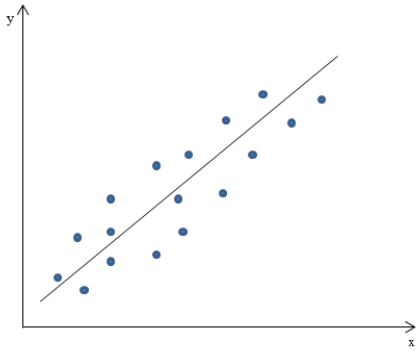
2.2.1. Regressão

O modelo de regressão linear é aquele que é usado para estudar a relação entre uma variável dependente e uma (modelo de regressão linear simples – Figura 1) ou mais (modelo de regressão linear múltiplo – Figura 2) variáveis independentes. A forma genérica de um modelo de regressão linear é (Greene 2012):

$$y = f(x_1, x_2, \dots, x_K) + \varepsilon = x_1\beta_1 + x_2\beta_2 + \dots + x_K\beta_K + \varepsilon,$$

onde y é a variável dependente ou explicada e x_1, x_2, …, x_K são as variáveis independentes ou explicativas. O termo ε é o termo de perturbação aleatória. Este termo representa a parte que não se consegue explicar de y e pode apresentar valores positivos ou negativos.

Exemplos:



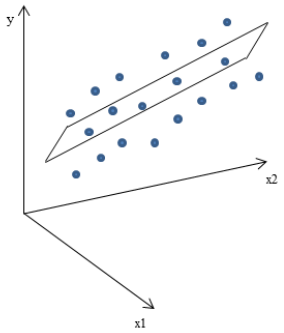


Figura 1 – Regressão Linear Simples **Figura 2 – Regressão Linear Múltipla**

Fonte: Elaboração Própria

Geman, Bienenstock et al. (1992) descrevem um típico problema de aprendizagem. Um problema de aprendizagem implica uma característica ou input x, um vetor de resposta y, sendo que o objetivo é prever y a partir de x. Os pares (x,y) obedecem a uma distribuição de probabilidade conjunta, P. Os dados de treino (x_1, y_1), …, (x_n, y_n) são um conjunto de pares valores observados de (x,y) que contêm a resposta y para cada valor de x.

O problema de aprendizagem consiste em construir uma função $\hat{f}(x)$ baseada nos dados treino (x_1, y_1), …, (x_n, y_n), em que $\hat{f}(x)$ aproxima o valor desejado de y.

2.2.2. Métricas de erro

No caso de problemas de regressão, o erro da hipótese *f* pode ser calculado pela distância entre o valor y_i conhecido e aquele predito pelo modelo, ou seja, $\hat{f}(x_i)$ (Monard and Baranauskas 2003). As medidas de erro mais conhecidas e usadas nesse caso são o erro quadrático médio (MSE- *mean squared error*) e a distância absoluta média (MAD- *mean absolute distance*, Lorena, Facelli et al. 2012).

$$mse(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

$$mad(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{f}\ (x_i)\right|$$

O MSE e o MAD são sempre não negativos. Para ambas as medidas, valores mais baixos correspondem a melhores modelos, ou seja, melhores aproximações dos rótulos verdadeiros dos objetos (Lorena, Facelli et al. 2012).

A função $\hat{f}$ é obtida correndo o algoritmo de indução em dados constituídos por um conjunto finito com n exemplos na forma $\{(x_1,y_1), \ldots, (x_n,y_n)\}$. A função $\hat{f}$ é denominada de modelo ou previsor. Dado que é impossível determinar o verdadeiro erro de um modelo $\hat{f}$ de acordo com a equação acima, o erro é estimado numa parte diferente dos dados, ou seja, nos exemplos para teste.

Existem outras funções de generalização do erro para previsões numéricas que também podem ser usadas em conjuntos de regressões. No entanto, a maioria dos trabalhos sobre múltiplos modelos usam o *mse* (Mendes-Moreira, Soares et al. 2012).

2.3. Processo de Aprendizagem de Múltiplos Modelos

Segundo Roli, Giacinto et al. (2001), o processo de aprendizagem de múltiplos modelos consiste num conjunto de modelos diferentes e numa função de combinação das previsões. O conjunto de modelos é obtido pelo processo denominado paradigma sobreprodução e escolha. Este processo contém duas fases: a fase da sobreprodução que consiste em gerar um conjunto de possíveis candidatos a modelos e a fase de escolha que seleciona o subconjunto de modelos que minimiza o erro de previsão.

Rooney, Patterson et al. (2004) descrevem o processo de aprendizagem de múltiplos modelos como sendo a solução de dois problemas: (1) geração do conjunto, ou seja, como gerar o conjunto de modelos e (2) integração do conjunto, isto é, como integrar as previsões dos modelos do conjunto de forma a obter a previsão final. Esta é uma abordagem direta (não tem poda) e pode ser vista como um caso particular da abordagem da sobreprodução e escolha. A poda também pode ser adicionada com sucesso a métodos diretos sem diminuir a precisão da previsão (Mendes-Moreira, Soares et al. 2012). A geração de conjunto pode ser classificada como *homogénea* no caso de todos os modelos terem como base o mesmo algoritmo de aprendizagem e

heterogénea se os modelos possuem como base diferentes algoritmos de aprendizagem. A integração de conjunto pode ser tratada com qualquer um destes dois mecanismos, seja com a combinação das previsões de modelos base (abordagem da combinação) ou pela previsão de um modelo base selecionado de acordo com um certo critério de forma a obter a previsão final (abordagem da seleção). Trabalhos empíricos e teóricos realizados mostram que o processo de aprendizagem conjunta só é eficaz se os modelos forem suficientemente diversos e precisos. No entanto, no caso de todos os modelos terem um erro de previsão muito baixo, então, a base de aprendizagem dos modelos deve ser muito semelhante, o que pode não garantir a diversidade. Na maioria dos casos existe um *trade-off* entre diversidade e precisão (Rooney, Patterson et al. 2004).

2.4. Geração do Conjunto

Como foi referido anteriormente, segundo Mendes-Moreira, Soares et al. (2012) a geração de um conjunto é a primeira fase no processo de aprendizagem conjunta. O objetivo é obter um conjunto de modelos:

$$F0 = \{\hat{f}_i, i = 1, \dots, K0\}.$$

Em conjuntos homogéneos, os modelos são gerados usando o mesmo algoritmo. Deste modo, a precisão e a diversidade dos modelos podem ser alcançadas através da manipulação da base de dados ou através do processo de geração de modelos. Já os conjuntos heterogéneos são obtidos quando mais que um algoritmo de aprendizagem é usado. Dada a natureza destes conjuntos, é esperado obter modelos com alguma diversidade. O problema prende-se com a falta de controlo sobre a diversidade da base de aprendizagem durante a fase de geração de conjunto. Esta dificuldade pode ser ultrapassada com o paradigma da sobreprodução e escolha, pois gerando um vasto conjunto de modelos, a probabilidade de ter um conjunto de modelos diversificado e preciso aumenta. Alguns autores defendem que conjuntos heterogéneos apresentam melhor performance que os conjuntos homogéneos. É de referir que os conjuntos heterogéneos podem ter conjuntos homogéneos como base de aprendizagem. A geração de conjunto pode ser classificada de acordo com a forma como a base de dados é manipulada ou o processo de modelagem de forma a gerar modelos diversos.

2.5. Poda do Conjunto

Mendes-Moreira, Soares et al. (2012) referem que os métodos de geração de modelos múltiplos criam conjuntos diversos, no entanto, não garantem o uso do menor conjunto capaz de maximizar a precisão. A poda do conjunto consiste em selecionar um subconjunto *F* de modelos gerados no passo anterior (geração de conjunto), F_0 (pilha de modelos). O objetivo desta fase é melhorar a habilidade preditiva e reduzir os custos. Esta é a fase de escolha na abordagem sobreprodução e escolha. Segundo Zou et al. 2002 e Hernández-Lobato et al. 2006, mesmo nas abordagens diretas, a adição da fase da poda não só reduz os custos computacionais mas também, em alguns casos, aumenta a precisão da previsão.

Os métodos para poda do conjunto podem ser classificados como baseados na partição ou baseados na procura. Os métodos baseados na partição dividem o conjunto de modelos em subgrupos usando um dado critério de partição e escolhem um modelo representativo de cada subgrupo. Esta abordagem assume que o conjunto de modelos contém um grande número de modelos similares e apenas um pequeno número deles não são redundantes. Todas as abordagens baseadas na partição têm em conta a geração dos subgrupos a partir de algoritmos de agrupamento (*clustering*). Na prática é óbvio que este método garante a diversidade, no entanto, é usada uma medida de avaliação da precisão diferente da que é usada para as abordagens baseadas na procura, que claramente favorece o desempenho do conjunto (Mendes-Moreira, Soares et al. 2012).

Os métodos para poda baseados na procura consistem em procurar um subconjunto de modelos, adicionando ou removendo modelos iterativamente do subconjunto candidato de acordo com uma dada medida de avaliação. Métodos baseados na procura podem ser classificados de acordo com: o objeto de avaliação, o algoritmo de procura e a medida de avaliação. Nos métodos baseados na procura é usada a mesma classificação de algoritmos que é utilizada na seleção de características (Aha and Bankert 1995): exponencial, aleatória e sequencial. Os algoritmos de procura exponencial procuram completar o espaço de *input*. Se o subconjunto de modelos selecionados da piscina de modelos possui *K* modelos, o espaço de procura tem $2^k - 1$ subconjuntos não vazios. A procura de algoritmos aleatória realiza uma procura heurística no espaço de *input*

usando métodos estocásticos, mediante um algoritmo evolucionário. Os algoritmos de procura sequencial mudam iterativamente uma solução adicionando e/ou removendo modelos (Mendes-Moreira, Soares et al. 2012).

Hernández-Lobato, Martínez-Muñoz et al. (2006) apresentam uma poda em conjuntos ordenados de regressões. Neste caso, a poda procede ordenando os modelos de previsão do conjunto original e selecionando um subconjunto para agregação. Deste modo, é construído um conjunto de modelos com tamanho máximo, incluindo primeiro os modelos que apresentam uma melhor performance quando agregados. Esta estratégia fornece uma solução semelhante à de extrair do conjunto original o subconjunto com o menor erro. Um dos métodos mais usados para construir um conjunto de modelos de previsão é o *bagging* (Breiman 1996). Alguns investigadores mostram que é possível selecionar subconjuntos de modelos que apresentam uma melhor performance que o conjunto completo (Zhou, Wu et al. 2002). É fácil de entender que isto acontece porque alguns modelos do conjunto têm um efeito negativo na previsão e por isso têm que ser removidos. No entanto, a identificação destes modelos é complicada. Hernández-Lobato, Martínez-Muñoz et al. (2006) apresentam um algoritmo para identificar o subconjunto ótimo do conjunto de modelos original. Nos métodos de *bagging* standard, a agregação dos modelos é determinada pelo processo de amostra *bootstrap*, que é um processo aleatório. Os modelos de regressão são agregados pela ordem em que são gerados pelas diferentes amostras *bootstrap*. No *bagging* ordenado, a agregação é adiada até que todos os modelos sejam gerados. Subconjuntos de tamanhos maiores são construídos incorporando, em cada iteração, o modelo de regressão que reduz o erro de treino do subconjunto. Chegando a um ponto o processo é interrompido e é encontrado o subconjunto final. Hernández-Lobato, Martínez-Muñoz et al. (2006) desenharam um algoritmo que constrói em cada passo, a melhor solução local. O algoritmo começa com um conjunto vazio, selecionando de seguida em cada iteração o modelo de previsão que quando incorporado, faz a maior redução do erro de treino do novo conjunto. O modelo de previsão selecionado na iteração *u* é o que minimiza a expressão:

$$s_u = arg\min_k u^{-2}(\sum_{i=1}^{u-1}\sum_{j=1}^{u-1} C_{s_i s_j} + 2\sum_{i=1}^{u-1} C_{s_i k} + C_{kk}),$$

onde, $k \in \{1, \ldots, N\}\{s_1, s_2, \ldots, s_{u-1}\}$, $\{s_1, s_2, \ldots, s_{u-1}\}$ são os índices dos modelos de previsão que já foram incorporados no conjunto podado na iteração u-1 e C_{ii} é o erro quadrático médio do membro do conjunto i.

2.6. Integração de Múltiplos Modelos

A última fase do processo de aprendizagem de múltiplos modelos é a integração de múltiplos modelos. Esta fase consiste em saber como é possível combinar as previsões dos vários modelos pertencentes ao conjunto obtido na fase anterior, de forma a obter uma única previsão (Mendes-Moreira, Soares et al. 2012). Em problemas de regressão, a integração de múltiplos modelos é dada por uma combinação linear dos previsores:

$$\hat{f}_{F(x)} = \sum_{i=1}^{K} [h_i(x) * \hat{f}_i(x)], \text{ onde } h_i(x) \text{ é a função de pesos.}$$

As abordagens de integração de múltiplos modelos podem ser divididas em funções de pesos constantes ou funções de peso não constantes (Merz 1998). A função de pesos constantes definida por Merz (Merz 1998) usa os dados de validação para estimar os parâmetros da função de pesos. Na função de pesos constantes tem-se que $h_i(x) = \alpha_i$. No entanto, existem outros métodos que usam apenas uma parte dos dados de validação para obter os pesos mais especializados para os dados de teste. Na função de pesos não constante, os pesos variam com o *input x*.

Os pesos h_i podem ser estimados globalmente ou dinamicamente. Quando os pesos são estimados globalmente, isto significa que é utilizado o mesmo conjunto de pesos para todos os dados de teste. Por outro lado, os pesos estimados dinamicamente são estimados de acordo com os dados de teste. Mendes-Moreira, Soares et al. (2012) descrevem alguns métodos para determinar os pesos. O método de integração básico (Perrone and Cooper 1993) calcula a média das previsões dos modelos do conjunto:

$$\hat{f}_{BEM}(x) = \frac{1}{K} \sum_{i=1}^{K} \hat{f}_i(x)$$

Deste modo, os pesos são dados por $h_i = 1/K$. Este método não depende dos modelos nem da base de dados, assumindo que os erros dos modelos são mutuamente independentes com média zero. Foi proposto pelos mesmos autores um método mais

complexo- método generalizado do conjunto, em que os pesos α_i são inversamente proporcionais ao erro nos dados de treino. Neste método, os erros também são estimados tendo em conta a correlação entre o erro e os modelos. No entanto, o método de integração generalizada sofre do problema da multicolineariedade. Um método que não sofre do problema da multicolineariedade é a mediana simples. Este método apresenta bons resultados usando MARS[1] (Friedman 1991) como base de aprendizagem (Mendes-Moreira, Soares et al. 2012).

Uma abordagem diferente referida por Mendes-Moreira, Soares et al. (2012) consiste em métodos que têm como objetivo evitar a multicolineariedade. O método *stacked regression* (Breiman 1996) é descrito como: dado o conjunto dos dados de aprendizagem *L* com M exemplos, o objetivo é obter os pesos α_i que minimiza:

$$\sum_{j=1}^{M}[f(x_j) - \sum_{i=1}^{K} x_i * \hat{f_i}(x_j)]^2,$$

usando os dados de treino.

Um resultado importante apresentado por Breiman (1996) é que, na maioria dos casos, muitos dos pesos α_i apresentam o valor zero, o que prova a importância da poda na fase de escolha dos modelos (Mendes-Moreira, Soares et al. 2012).

A determinação dos pesos a partir de métodos dinâmicos consiste em atribuir um peso a cada modelo de acordo com a sua precisão (Rooney, Patterson et al. 2004), sendo que a previsão final é baseada na média dos pesos das previsões dos modelos (Mendes-Moreira, Soares et al. 2012).

2.7. Métodos de integração dinâmica

Segundo Mendes-Moreira, Soares et al. (2012), nos métodos de integração dinâmicos, a seleção dos modelos é feito em tempo real. Dado um novo exemplo, ele escolhe os previsores que se espera fazerem a previsão combinada mais precisa. Nos métodos de integração dinâmica pode acontecer apenas serem usados alguns modelos, de forma a não usar modelos considerados imprecisos para um dado conjunto de dados teste (pós-

[1] MARS (*Multivariate Adaptive Regression Splines*) é um método de análise de regressão introduzido por Jerome Friedman em 1991. É uma técnica de regressão não paramétrica e pode ser vista como uma extensão de modelos lineares que automaticamente moldam interações entre variáveis.

poda). A abordagem dinâmica consiste nas seguintes fases (Mendes-Moreira, Soares et al. 2012), tendo por base que já é conhecido o conjunto de modelos:

(1) Dado um input x, é necessário encontrar uma base de dados similar (L_s) aos dados de teste (L_v), tal que $L_s \subseteq L_v$;
(2) Selecionar um subconjunto de modelos $F_x \subseteq F$ do conjunto de modelos de acordo com a precisão que tiveram na base de dados similar L_s. Esta fase é denominada de pós-poda;
(3) Obter as previsões $\hat{f}_i(x)$ para os valores de *input*, para cada $\hat{f}_i \in F_x$;
(4) Obter a previsão da combinação dos múltiplos modelos $\hat{f}_{F_x}$. Por vezes, na fase (2) é apenas selecionado um único modelo. Nesse caso, não é preciso fazer a combinação dos modelos. No entanto, quando é selecionado mais que um modelo é necessário escolher um método de integração.

O método normalmente utilizado para encontrar uma base de dados idêntica é o algoritmo dos k-vizinhos mais próximos com a distância euclidiana.

2.8. O erro de generalização do conjunto

Pode dizer-se que um conjunto bem-sucedido é aquele que tem modelos precisos e em que os erros são cometidos em partes diferentes dos dados. Contudo, para se compreender o erro de generalização do conjunto é necessário conhecer quais as características que os modelos devem possuir para reduzir o erro de generalização global. A decomposição do erro de generalização em regressões é simples. A maioria das decomposições feitas foram inicialmente propostas para conjuntos de redes neuronais, no entanto, elas não estão dependentes do algoritmo de indução usado. De forma a simplificar, *f*(x) pode ser representado por *f*.

A decomposição viés/variância para uma única rede neuronal:

$$E\left\{\left[\hat{f} - E(f)\right]^2\right\} = \left[E\left(\hat{f}\right) - E(f)\right]^2 + fE\left\{\left[\hat{f} - E\left(\hat{f}\right)\right]^2\right\}.$$

O primeiro termo do lado direito da equação é denominado de viés e representa a distância entre o valor esperado do previsor $\hat{f}$ e a média (desconhecida) da população. O segundo termo (variância) mede o quanto a variabilidade da previsão em torno da

média da previsão. Deste modo, a decomposição viés/variância pode ser escrita da seguinte forma (Mendes-Moreira, Soares et al. 2012):

$$mse(f) = viés(f)^2 + var(f).$$

2.9. Aprendizagem *Online*

Shalev-Shwartz (2007) referem que o processo de aprendizagem online ocorre numa sequência de iterações consecutivas. Em cada iteração, é dada uma pergunta e é requerida uma resposta. Para responder à pergunta, o algoritmo de aprendizagem usa um mecanismo de previsão, que serve de mapa desde o conjunto de perguntas ao conjunto de respostas admissíveis. Depois de fazer a sua previsão, o algoritmo de aprendizagem recebe a resposta correta. A qualidade da previsão é avaliada por uma função custo que mede a discrepância entre a resposta prevista e a correta. Para o algoritmo atingir o objetivo de prever com o menor custo possível, o algoritmo de aprendizagem deve atualizar as suas hipóteses em cada iteração para se tornar mais preciso na iteração seguinte. O algoritmo tenta deduzir informações a partir de exemplos anteriores de forma a melhorar as suas previsões presentes e futuras. No entanto, a aprendizagem é impossível se não existe uma correlação entre os exemplos passados e os exemplos presentes.

Auer and Gentile (2000) estudam a aprendizagem online no contexto de regressões lineares. A maior parte dos limites de desempenho para algoritmos online neste contexto assume uma taxa de aprendizagem constante. Para atingir estes limites, a taxa de aprendizagem deve ser otimizada com base em informação *à posteriori*. Estes autores introduziram novas técnicas para adaptar a taxa de aprendizagem à medida que a base de dados é progressivamente revelada. Essa nova técnica é diferente das conhecidas até agora, geralmente denominadas de *duplo truque*. Enquanto que o *duplo truque* faz reiniciar o algoritmo várias vezes usando uma taxa de aprendizagem constante para cada iteração, este novo método usa a informação revelada em cada iteração para atualizar o valor da taxa de aprendizagem suavemente.

2.9.1. *Weighted Majority Algorithm*

Littlestone and Warmuth (1994) estudam algoritmos de previsão online que aprendem seguindo os pressuposto seguintes. A aprendizagem acontece numa sequência de ensaios. Em cada ensaio, o algoritmo recebe um exemplo e faz uma previsão binária (problema de classificação). No fim do ensaio, o algoritmo recebe o valor correto para a previsão. O algoritmo é então avaliado de acordo com o número de vezes que faz uma previsão errada. É dado um conjunto de algoritmos de previsão com números de erro variados. O objetivo é construir um algoritmo mestre que usa as previsões do conjunto de modelos para fazer a sua própria previsão. O algoritmo mestre não deve cometer um número de erros maior que o do melhor previsor do conjunto. Este processo procede da seguinte forma em cada ensaio: o mesmo exemplo é dado a todos os algoritmos do conjunto de modelos. Cada algoritmo faz uma previsão e estas previsões são agrupadas para formar o exemplo que será dado ao algoritmo mestre. O algoritmo mestre faz então a sua previsão e recebe o valor verdadeiro, passando-o a todo o conjunto de modelos.

O *weighted majority alorithm* inicialmente associa um peso positivo a cada modelo (função) de previsão do conjunto. Geralmente, todos os pesos iniciais são iguais a um a não ser que seja especificado de outro modo. O *weighted majority alorithm* forma a sua previsão comparando o peso total $\boldsymbol{q_0}$ dos algoritmos do conjunto que preveem o valor 0 com o peso total $\boldsymbol{q_1}$ dos algoritmos que preveem o valor 1. Este prevê de acordo com a maioria total (arbitrariamente em caso de empate). Quando o *weighted majority alorithm* comete um erro, o peso desse algoritmo do conjunto que não está de acordo com o valor verdadeiro, é multiplicado por um valor fixo $\boldsymbol{\beta}$, sendo que $\boldsymbol{0 \leq \beta < 1}$ (Littlestone and Warmuth 1994).

Arora, Hazan et al. (2012) apresentam uma generalização do *weighted majority alorithm*. Na maioria dos casos, continua-se a ter um conjunto de modelos constituído por *n* peritos que fazem previsões. Os conjuntos de exemplos/resultados, nesta generalização, não precisam de ser dados binários e pode até ser um conjunto de valores infinitos. Pode-se então aplicar este algoritmo a problemas de regressão. Para motivar o algoritmo de atualização multiplicativa dos pesos, considera-se a estratégia de, em cada iteração, limitar-se a escolher um algoritmo aleatoriamente. A penalização esperada será a do

perito "médio". Vamos agora supor que alguns peritos claramente superam os seus concorrentes. Os peritos que têm uma performance claramente superior aos restantes são recompensados, aumentado a probabilidade de serem escolhidos na iteração seguinte.

Quando no início se desconhece o desempenho dos peritos, estes são selecionados uniformemente ao acaso para o conjunto de modelos ativos.

O conjunto de exemplos/resultados é representado por P. Assume-se que existe uma matriz M, em que M (*i*, *j*) é a penalização que o perito *i* sofre quando o resultado é $j \in$ P. É também assumido que para cada perito *i* e cada resultado *j*, M (*i*, *j*) está no intervalo [-*l*, ρ], onde $\boldsymbol{l \leq \rho \; e \; l > 0}$. O algoritmo de previsão será aleatório e é desejado que a penalização esperada não seja pior que a esperada para o melhor perito (Arora, Hazan et al. 2012).

Em cada tempo *t*, tem-se um peso w_i^t associado ao perito *i*. Inicialmente, $w_i^t = 1$ para todos os peritos (*i*). Em cada tempo *t*, é associada a distribuição $\boldsymbol{D^t} = \{p_1^t, p_2^t, \dots, p_n^t\}$ ao perito onde $p_i^t = \; w_i^t / \sum_k w_k^t$.

O valor esperado da penalização para o resultado $\boldsymbol{j^t \in P}$ é dado por:

$$\boldsymbol{\sum_i p_i^t \, \mathbf{M}(i, j) = \sum_i w_i^t \, \mathbf{M}(i, j^t) / \sum_i w_i^t},$$

que é representado por M ($\boldsymbol{D^t, j^t}$). A penalização total depois de T instantes é $\boldsymbol{\sum_{t=1}^{T} \mathbf{M}\,(D^t\,, j^t)}$.

2.9.1.1. Algoritmo de atualização multiplicativa dos pesos

No instante *t*, é escolhido um perito de acordo com a distribuição $\boldsymbol{D^t}$e usa-a para obter a sua previsão. Baseado no resultado $j^t \in \boldsymbol{P}$ no instante *t*, em *t*+1, o peso do perito *i* é atualizado da seguinte maneira:

$$w_i^{t+1} = \begin{cases} w_i^t (1-\epsilon)^{\boldsymbol{M}(i,j)/\boldsymbol{\rho}} & \text{if } \mathbf{M}(i, j^t) \geq 0 \\ w_i^t (1-\epsilon)^{\boldsymbol{M}(i,j)/\boldsymbol{\rho}} & \text{if } \mathbf{M}(i, j^t) < 0 \end{cases}$$

2.9.2. *Tracking the best expert*

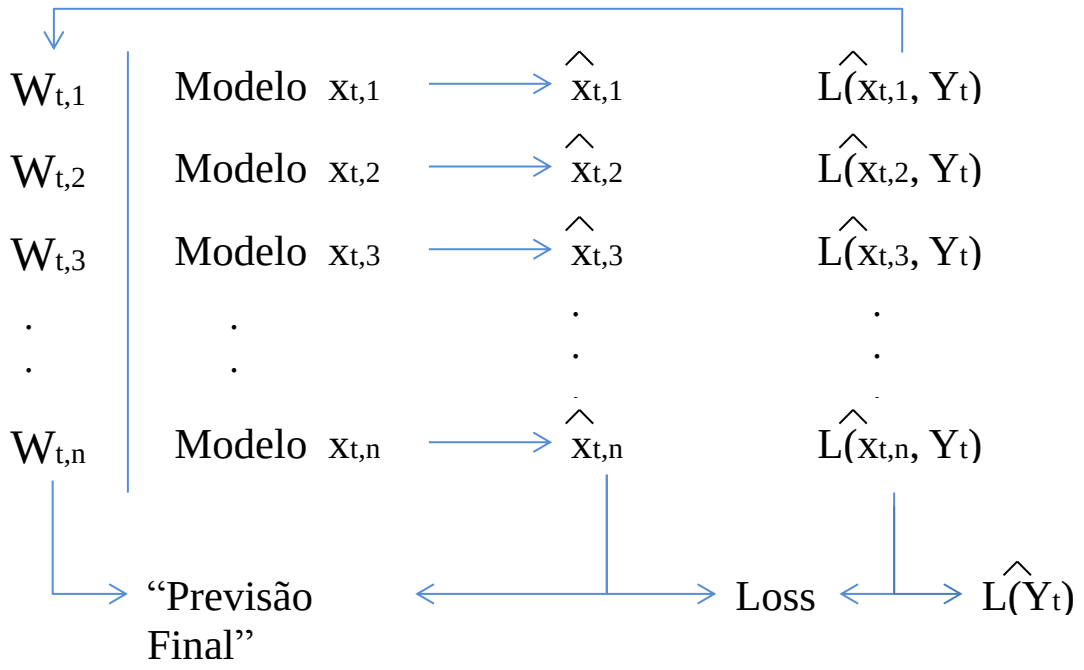


Figura 3- Esquema de funcionamento do modelo "*Tracking the Beste Expert*"

Fonte: Elaboração Própria

Considera-se o modelo de aprendizagem online que se segue (Herbster and Warmuth 1998 - Figura 3). A aprendizagem ocorre numa série de ensaios numerados 1,2,…, *n*. Em cada ensaio *t* o objetivo é prever o output y_t que é conhecido no fim do ensaio. No início do ensaio *t*, o algoritmo recebe uma n-tupla x_t. O elemento x_t,i da n-tupla x_t representa a previsão feita por um perito ε_i , do valor de y_t do ensaio *t*. Deste modo, o algoritmo produz uma previsão $\hat{f}_t$ baseada na corrente previsão do perito x_t, nas previsões passadas e nos valores observados. No final do ensaio, o algoritmo fica a conhecer o valor observado, ou seja, y_t. De seguida, o algoritmo mede o custo da previsão, isto é, a discrepância entre a previsão ($\hat{f}_t$) e o valor observado y_t. Desta forma, cada perito tem o seu custo de previsão. Um objetivo possível é minimizar o custo total do algoritmo sobre todos os ensaios numa sequência arbitrária de pares de resultados.

Sendo que não é tido em conta qualquer pressuposto sobre a relação entre a previsão dos peritos (x_t) e o valor observado (y_t), haverá sempre uma sequência de y_t que se encontra longe dos valores previstos $\hat{y}_t$ para cada algoritmo em particular. Deste

modo, um objetivo razoável é minimizar o custo de previsão numa sequência arbitrária de exemplos. Caso todos os peritos tenham custos de previsão elevados então este objetivo pode tornar-se fácil de alcançar. Este facto acontece uma vez que para todos os algoritmos, a perda adicional sobre o custo do melhor perito pode apresentar um valor reduzido.

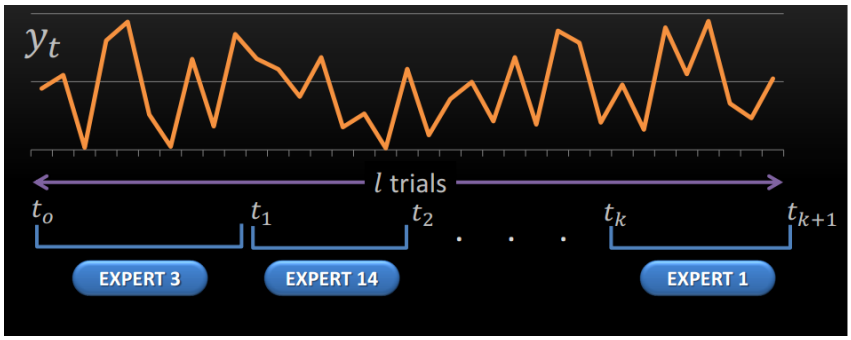


Figura 4 – Esquema de funcionamento do modelo *"Tracking the Best Expert"*

Fonte: http://cseweb.ucsd.edu/~kamalika/teaching/CSE291W11/mar2.pdf acedido em Dezembro de 2013.

Este quadro de peritos pode ser usado em várias questões. Por exemplo, é possível prever qual a probabilidade de chover ou se o mercado de ações vai cair ou subir. Outra maneira de trabalhar com os peritos é que eles podem ser sub-algoritmos, sendo que existe um algoritmo mestre que combina as previsões dos peritos (Figura 4). Este algoritmo mestre define um peso por perito, que representa a crença na previsão do perito, e em seguida, diminui ou aumenta o peso consoante a previsão do perito.

Trabalhos anteriores de Vovk (1998) e outros (Littlestone and Warmuth 1994; Haussler, Kivinen et al. 1998) produziram um algoritmo para o qual existe um limite superior para o custo adicional sobre o custo do melhor perito (Herbster and Warmuth 1998).

Em Herbster and Warmuth (1998), os algoritmos em que o seu custo é comparado ao do melhor perito são chamados de *peritos estáticos.*

Littlestone and Warmuth (1994) consideram que a sequência de exemplos é dividida em k+1 segmentos de tamanho e distribuição arbitrária. Cada segmento tem um perito associado. À sequência de segmentos e por conseguinte, à sequência de peritos, dá-se o nome de *partição*. O custo da partição é dado pela soma do custo total dos peritos

associados a cada segmento. A melhor *partição* de tamanho *k*, é a partição com *k+1* segmentos com o custo de previsão mais baixo. Herbster and Warmuth (1998) propõem o objetivo de melhorar a previsão da melhor *partição*. Este objetivo serve para modelar as previsões a situações da vida real em que a natureza dos exemplos se altera e passa a ser outro perito a prever melhor. Por exemplo, os padrões podem mudar e diferentes algoritmos podem prever melhor para diferentes segmentos da sequência *online* de padrões. Herbster and Warmuth (1998) procuram construir um algoritmo mestre capaz de controlar a performance da melhor sequência de peritos no sentido de incorrê-los a um custo adicional em relação ao custo da melhor partição de tamanho k. Se a sequência completa de exemplos foi determinada antes do tempo, é possível então calcular a melhor partição de um determinado tamanho e os peritos associados usando programação dinâmica. O algoritmo obtém os exemplos *online* e nunca produz a melhor *partição*. No entanto, este algoritmo limita o custo de previsão adicional em relação à melhor previsão offline da partição para uma sequência arbitrária de exemplos.

Quando se tem *l* ensaios, *k+1* segmentos e *n* peritos, então existem $\left(\frac{l-1}{k}\right) n(n-1)^k$ partições diferentes. Pode obter-se uma boa limitação para este problema expandindo o conjunto dos n peritos em $\left(\frac{l-1}{k}\right) n(n-1)^k = O((n^{k+1}\left(\frac{el}{k}\right)^k)$ *partições de peritos*. Cada *partição de peritos* representa uma partição singular da sequência de ensaios e prevê, em cada ensaio, como o perito associado ao segmento (ver Figura 4). Sendo assim, usando o algoritmo do *perito estático* obtém-se um limite de $cln\left(\frac{l-1}{k}\right) n(n-1)^k \leq c[(k+1)\log n + k\log\frac{l}{k} + k]$ do custo adicional do algoritmo sobre o custo da melhor *partição*. Deste modo, levantam-se dois problemas: o algoritmo é ineficiente uma vez que o número de *partições de peritos* é exponencial no número de *partições*; o limite no custo adicional cresce com o tamanho da sequência (Herbster and Warmuth 1998). É possível superar estes dois problemas. Em vez de ter um peso para cada uma das muitas *partições*, pode manter-se um único peso por perito como se faz no algoritmo do *perito estático*. Se for pretendido combinar as previsões dos *n* sub-algoritmos ou peritos, então o algoritmo mestres demora apenas mais *O(n)* de tempo adicional por ensaio sobre o tempo necessário para simular os *n* sub-algoritmos.

Herbster and Warmuth (1998) desenvolveram dois algoritmos principais: o algoritmo *fixed share* e o algoritmo *variable share*. Ambos os algoritmos são baseados no algoritmo *static expert* que mantém um peso de $e^{-n\,T_i}$ para cada perito (Littlestone and Warmuth 1994), onde T_i é o custo total de previsão passado do perito *i* nos ensaios passados. Em cada ensaio, as previsões dos peritos são combinadas usando os pesos atuais dos peritos. Quando o resultado de cada ensaio é recebido, é multiplicado o peso de cada perito *i* por e^{-nL_i}, onde L_i é o custo do perito *i* no ensaio atual. Este processo designa-se de atualização dos pesos. Herbster and Warmuth (1998) modificaram o algoritmo de *perito estático*, adicionando uma atualização. Por essa razão, cada perito partilha uma porção do seu peso depois da atualização dos pesos. A este processo dá-se o nome de *atualização da partilha*. Ambos os algoritmos (*partilha fixa* e *partilha variável*) primeiro procedem à atualização do custo e só depois da partilha. Na *atualização da partilha*, uma fração do peso de cada perito é adicionado ao peso de cada outro perito. No algoritmo de *partilha fixa*, os peritos partilham uma fração fixa do seu peso com os outros. Isto garante que o rácio do peso de qualquer perito sobre o peso total de todos os peritos pode ser limitado.

As funções de custo são as comuns, o erro quadrático, *relative entropy loss* e a *hellinger loss*.

- Erro quadrático $L(p,q) = (p-q)^2$
- *Relative entropy loss* $L_{ent}(p,q) = p\ ln\frac{p}{q} + (1-p)\ln\frac{1-p}{1-q}$
- *Hellinger loss* $L_{hel}(p,q) = \frac{1}{2}\left(\left(\sqrt{1-p}-\sqrt{1-q}\right)^2 + \left(\sqrt{p}-\sqrt{q}\right)^2\right)$

O algoritmo de *partilha fixa* obtém o custo adicional a partir de $O\left(c\left[(k+1)\log n + k\log\frac{l}{k} + k\right]\right)$, que é essencialmente o mesmo algoritmo usado pelo *perito estático* com partições-peritos exponenciais. Uma característica importante do algoritmo de *partilha fixa* é que continua a usar *O*(1) tempo por perito por ensaio. No entanto, o custo adicional deste algoritmo continua a depender do tamanho da sequência (Herbster and Warmuth 1998). Este algoritmo também partilha pesos após a atualização do custo de previsão, no entanto, a quantidade de cada partilha de perito está agora de acordo

com o custo de perito no ensaio corrente. Em particular, quando um perito não tem nenhum custo, não partilha peso.

Além do algoritmo de agregação de Vovk (1998) e o algoritmo da maioria ponderada (Littlestone and Warmuth 1994), que apenas usam a atualização do custo, um número considerável de trabalhos tem sido desenvolvido.

2.9.3. *Dynamic Weighted Majority*

Kolter and Maloof (2007) apresentam uma extensão ao *weighted majority alorithm* (Littlestone and Warmuth 1994), que não só tem em conta *drifting concepts* (Blum 1997), como também adiciona e remove peritos ou modelos. Por esse motivo, o *dynamic weighted majority algorithm* é mais habilitado para responder em problemas não estacionários.

Drifting concepts ocorre quando o rótulo das classes muda ao longo do tempo. Este conceito é importante em muitas aplicações que envolvem modelos de comportamento humano (Kolter and Maloof 2007). *Drifting concepts* é um problema de aprendizagem online em que o conceito muda ao longo do tempo. Por exemplo, se o problema em questão for o sistema de classificação de e-mails de um professor, o conceito de um "e-mail importante" vai mudar ao longo do tempo, por exemplo com a mudança de semestre (Kolter and Maloof 2007). Os conceitos podem mudar repentinamente ou gradualmente. O *dynamic weighted majority algorithm* mantem um conjunto ponderado de peritos ou modelos de previsão. Ele adiciona e remove peritos baseando-se na performance do algoritmo mestre. No caso do algoritmo mestre cometer erro, então ele adiciona um perito. Por outro lado, se um perito cometer erro, o *dynamic weighted majority algorithm* reduz o seu peso. Se depois de vários treinos um perito prevê mal, estando já com um peso muito reduzido, então este algoritmo elimina esse perito. Este método pode usar, geralmente, qualquer algoritmo online como base de aprendizagem. Também pode usar diferentes tipos de bases de aprendizagem, embora se tenha que implementar um controlo para determinar qual a base de aprendizagem a adicionar.

O algoritmo mantem um conjunto de *m* peritos (E), cada um com um peso (w_i), em que $i = 1, \dots, m$. O *input* do algoritmo são *n* exemplos para treino. Os parâmetros também incluem o número de classes (c) e um fator multiplicativo β que o *dynamic weighted majority algorithm* usa para fazer decrescer o peso de um perito no caso de ele prever incorretamente. Um valor típico para β é 0.5. O parâmetro θ é o limite a partir do qual se remove um perito que esteja a prever com baixa performance. O parâmetro p determina quantas vezes é que o *dynamic weighted majority algorithm* cria e remove peritos (Kolter and Maloof 2007).

Segundo (Kolter and Maloof 2007) o *dynamic weighted majority algorithm* funciona da seguinte maneira. O algoritmo começa por criar um conjunto que contém apenas um modelo com peso igual a um. Inicialmente, este modelo de previsão pode prever aleatoriamente ou utilizando experiência e conhecimentos já adquiridos. Desta forma, este algoritmo dá um exemplo singular e apresenta-o ao perito para ele fazer a previsão. Se a previsão do perito está errada, então o algoritmo diminui o peso desse modelo, multiplicando-o por β. Uma vez que apenas existe um perito no conjunto, a sua previsão é a previsão do algoritmo mestre. Deste modo, se a previsão do algoritmo mestre está incorreta então é criado um novo modelo com peso igual a um. Sendo assim, o *dynamic weighted majority algorithm* treina os peritos num novo exemplo. Depois de os treinar, o resultado deste algoritmo é a previsão global (ou previsão do algoritmo mestre). Quando já se tem múltiplos modelos de aprendizagem, o *dynamic weighted majority algorithm* obtem a previsão de cada membro do conjunto de modelos. Tendo em conta a performance das previsões, o *dynamic weighted majority algorithm* usa a previsão de cada modelo e os seus pesos para obter a soma dos pesos de cada classe. A classe com maior peso é a classe da previsão global.

2.9.4. Agregação de modelos especializados

Devaine, Gaillard et al. (2013) propõem um modelo de agregação de modelos especializados. Um conjunto de observações (ex: consumo de eletricidade hora a hora ou de meia em meia hora) $y_1, y_2, \dots, y_T \in [0, B]$, será previsto elemento a elemento a cada instante $t = 1, 2, \dots, T$. Um número finito *N* de métodos de previsão (peritos) está disponível. Antes de cada tempo *t*, alguns peritos fornecem uma previsão e outros não.

O primeiro grupo de modelos é designado de ativos enquanto o outro contém os inativos. As previsões dos peritos ativos são representadas por $f_{j,t} \in \mathbb{R}_+$, onde j é o índice do perito ativo considerado. Ao conjunto de peritos ativos em cada instante t é dada a seguinte representação: $E_t \subset \{1, \dots, N\}$ é assumido que é um conjunto não vazio. É assumido que os peritos conhecem o limite B e apenas produzem previsões para $f_{j,t} \in [0, B]$. Em cada tempo $t \geq 1$, uma sequência de regras de agregação produzem um vetor de pesos convexos $\boldsymbol{p}_t = (p_{1,t}, p_{2,t}, \dots, p_{N,t})$ baseado nas observações passadas $y_1, \dots, y_{t-1}$ e nas previsões passadas e do presente $f_{j,s}$, para todo o $s = 1, \dots, t$ e $j \in E_s$, sendo E_s o conjunto de peritos. Um vetor de pesos convexos é aquele em que $\boldsymbol{p}_t \in \mathbb{R}^N$, tal que $p_{j,t} \geq 0$ para todo o $j = 1, \dots, N$ e $p_{1,t} + \dots + p_{N,t} = 1$. O conjunto de todos estes vetores convexos é representado por X. A previsão final em t é obtida pela combinação linear dos peritos em E_t de acordo com os pesos dado pelas componentes do vetor $\boldsymbol{p}_t$. Mais precisamente, a previsão agregada em cada instante é dada por (Devaine, Gaillard et al. 2013):

$$\hat{y}_t = \sum_{j \in E_t} p_{j,t} f_{j,t}.$$

O valor observado y_t é revelado e o instante $t + 1$ começa.

Para medir a precisão da previsão pode ser utilizada uma função de erro vista anteriormente (secção 2.2.2.), como o erro quadrático ou o erro absoluto.

Devaine, Gaillard et al. (2013) propõem uma regra de agregação denominada de *regra de agregação especialista.*

A *regra de agregação especialista* depende do parâmetro $\eta > 0$. Esta regra começa com um vetor de pesos convexos uniforme ($\boldsymbol{w}_1$), $w_{i,1} = \frac{1}{N}$ para $i = 1, \dots, N$. Para cada instante $t = 1, 2, \dots, T$:

(1) Prever

$$\hat{y}_t = \frac{1}{\sum_{k \in E_t} w_{k,t}} \sum_{j \in E_t} w_{j,t} f_{j,t};$$

(2) Observar y_t e atualizar os vetor de pesos convexos $\boldsymbol{w}_{t+1}$ da seguinte maneira:

$$w_{i,t+1} = \begin{cases} w_{i,t} e^{-\eta l_t(\delta_i)} \dfrac{\sum_{i \in E_t} w_{j,t}}{\sum_{k \in E_t} w_{k,t} e^{-\eta l_t(\delta_t)}} & \text{if } i \in E_t \\ w_{i,t} & \text{if } i \notin E_t \end{cases}$$

2.10. Resumo Estado da Arte

Neste capítulo (secção 2) começou-se por identificar as dificuldades a que a previsão de de energia eólica está sujeita, nomeadamente o facto de depender de condições naturais tais como o vento. Deste modo, revelou-se de extrema importânica encontrar um método de previsão mais eficaz. A agregação de previsões de múltiplos modelos pode vir a ser um método eficaz. De forma a perceber-se como funciona a agregação de previsões de múltiplos modelos começou-se por esclarecer em que consiste uma regressão e quais as possíveis métricas de erro que podem ser utilizadas com a finalidade de medir a performance de uma previsão. De seguida seguiram os passos necessários para a integração de vários modelos. Com isso surgem as formas de actualização de pesos dos modelos em cada período, tal como o *weighted majority algorithm* e *tracking the best expert*. No capitulo seguinte (secção 3) serão aplicados alguns dos métodos aqui estudados à previsão de energia eólica. Para tal, em primeiro lugar serão analisados os dados fornecidos e de seguida efectuada a agregação das previsões.

3. Estudo de um caso

3.1. Introdução à Previsão de Energia Eólica

A electricidade gerada a partir da energia eólica irá desempenhar um papel importante no futuro em muitos países, no que diz respeito ao fornecimento de energia. Este facto implica integrar esta energia no sistema de fornecimento de energia eléctica já existente, que foi projectado principalmente para grandes unidades de combustíveis fosseis e centrais nucleares. Como a energia eólica tem características diferentes, esta integração leva a alguns desafios importantes do ponto de vista do sistema de energia eléctrica. A previsão da energia eólica tem um papel fundamental na luta contra este desafio, visto ser o pré-requisito para integração de grande parte da energia num sistema eléctrico (Boyle, 2007).

3.2. Trabalhos Relacionados

Catalão, Martins et al. (2008) apresentam uma ferramenta computacional para previsão da potência eólica baseada em redes neuronais. As Redes Neuronais Artificiais (RNA) conseguem encontrar uma relação entre os valores de entrada e os valores de saída, capaz de prever de forma melhor a potência eólica quando comparado com os métodos clássicos de regressão linear. A arquitetura utilizada consistiu numa rede neuronal com cinco unidades na camada escondida e uma unidade na camada de saída. Os resultados obtidos demonstraram que quanto menor o intervalo de tempo entre amostras de valores, melhor são os resultados obtidos. Esta ferramenta computacional permitiu ainda um tempo de computação aceitável.

Thorarinsdottir and Gneiting (2008) consideram uma base de dados com oito modelos diferentes e ainda observações da velocidade máxima do vento registada por cento e sete superfícies de observação das vias aéreas para desenvolver um conjunto de outputs de modelos usando regressão heteroscedástica censurada.

Devido à instabilidade a que a produção de energia eólica está sujeita, há ainda estudos que se concentram na previsão de eventos de rampa (alterações súbitas e grandes) da energia eólica. Ao se conseguir um maior controlo nos eventos de rampa de

energia eólica, é possível obter-se melhores previsões. Ferreira, Gama et al. (2010) apresentam uma revisão bibliográfica sobre a previsão de rampas de energia eólica. Quando se trata de um problema de previsão de eventos de rampa, há dois fatores principais a ter em conta: o objetivo da previsão e o horizonte temporal da previsão. O objetivo da previsão pode ser prever a velocidade do vento ou prever a produção de energia eólica. No caso da velocidade do vento, é necessário utilizar posteriormente modelos de turbinas para converter a velocidade do vento em energia eólica. Já no caso de o objetivo ser prever a produção de energia eólica, esta obtém-se diretamente (Negnevitsky and Johnson 2008). Quanto ao outro fator importante (horizonte temporal), sabe-se que a precisão das previsões diminui com o tempo, sendo que é muito difícil conseguir prever eventos de rampa com confiança para horizontes temporais mais longos que 48 horas (Ferreira, Gama et al. 2010). Quanto a modelos de previsão de eventos de rampa de energia eólica já desenvolvidos, tem-se o de Zheng and Kusiak (2009) que combina seleção de características com cinco algoritmos de *data-mining* para 10 a 60 minutos à frente. Estes autores usam a média, desvio-padrão, velocidade máxima e mínima do vento, o poder do parque eólico e a taxa de eventos de rampa do parque. Como um grande número de peritos pode baixar a precisão de previsão, é usado um algoritmo de árvores de decisão para selecionar os peritos com melhores performances. A seleção de características é usada para treinar cinco algoritmos *data-mining*: *Perceptron* Multicamadas, *Support Vector Machine* (SVM), Floresta Aleatória, Árvores de Decisão e Classificação e *Pace Regression*. No horizonte temporal analisado, verificou-se que o algoritmo SVM apresenta as previsões com a maior precisão. Bossavy, Girard et al. (2010) conseguem identificar eventos de rampa mapeando a série de energia eólica num sinal. Esta metodologia pode ser usada para detetar eventos diferentes eventos de rampa e apenas requer um único parâmetro.

3.3. Dados utilizados

Os dados utilizados para a realização deste trabalho foram fornecidos pela empresa PreWind. Nos dados constam os dados reais de produção de energia eólica de um aerogerador e os dados previstos por vários modelos de previsão. Os modelos de previsão são: ARMA1, PCM1, PCM2, PCM3, PCM4, PCM5, PCM6, PCM7, PCM8, PCM9, PCM10 E PCM11. Os valores estão apresentados de hora a hora, sendo que os

modelos fazem uma previsão de 24 horas, começando às 14h de cada dia. Os dados correspondem ao ano de 2012, sendo que foram eliminados alguns dias a seguir à ocorrência de mudança de hora. Deste modo, não há dados para os dias 24 e 25 de Março de 2012 e para os dias 26, 27 e 28 de Outubro de 2012. Cada modelo e os dados reais apresentam 8582 valores (horas).

A figura 5 representa os dados reais de produção de energia eólica. Pela análise do gráfico é possível observar que os valores apresentam uma grande variação e que existe uma redução dos mesmos entre os meses de abril e agosto, onde a produção diminui bastante (devido à diminuição do vento).

O valor máximo registado foi de 862 MW e o valor mínimo de -4 MW. Este valor negativo de produção de energia eólica diz respeito a um momento em que o aerogerador não está a produzir energia, no entanto, precisa de energia para rodar a nacela [anexo A]. Existem horas em que a produção de energia eólica regista valor nulo. Este facto acontece quando não há vento suficiente para que o aerogerador funcione ou quando por algum motivo o aerogerador teve que ser desligado (por exemplo, uma avaria). A mediana encontra-se nos 249 MW, o que significa que 50% das observações registam valores iguais ou inferiores a este valor. A produção de energia eólica apresenta ainda uma média de 300.6 MG

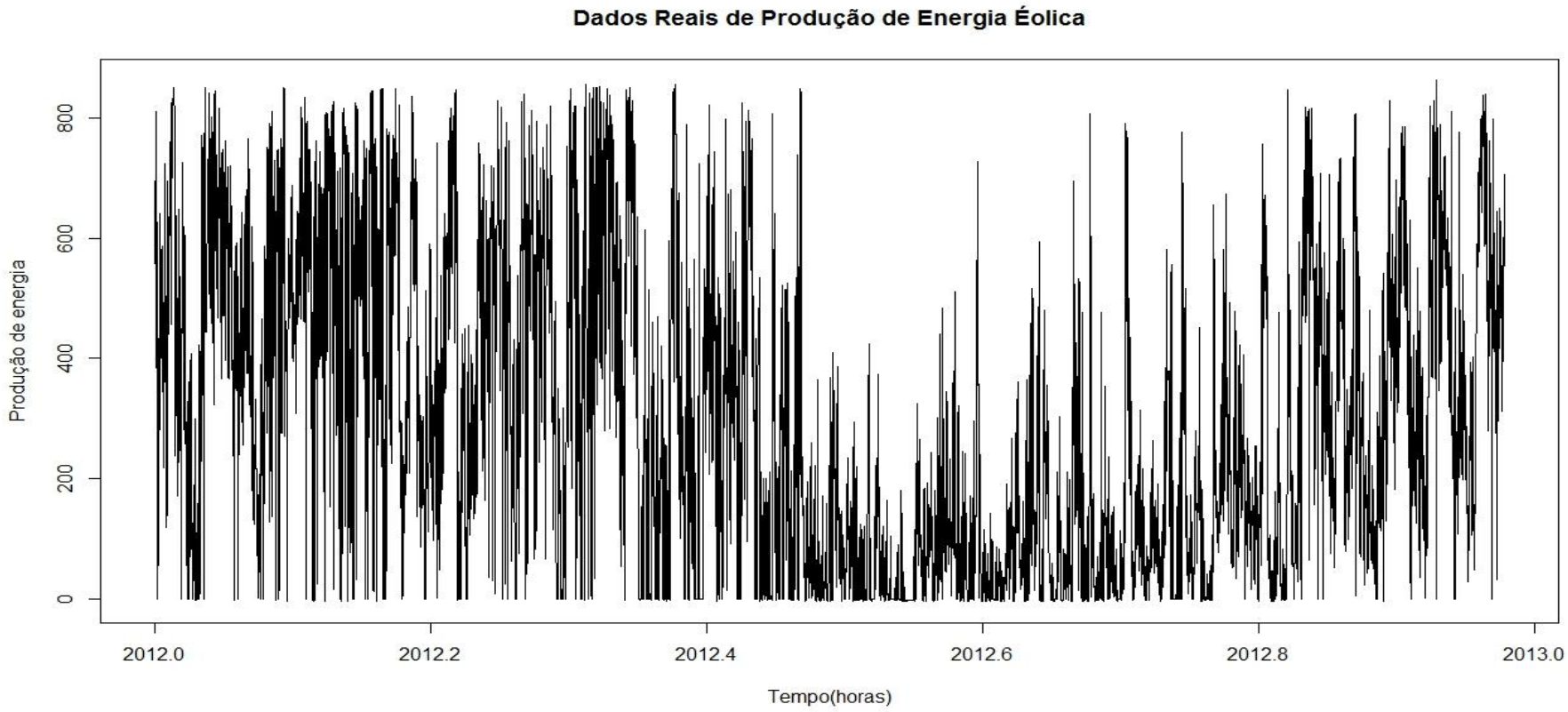


Figura 5 – Gráfico que representa a evolução dos dados reais de Produção de Energia Eólica ao longo do ano de 2012

De forma a obter-se o erro médio absoluto de cada um dos modelos foram usados os comandos no software R, apresentados de seguida.

Leitura do ficheiro "Tese.csv", onde os dados fornecidos que se apresentam organizados consoante demonstrado na Figura 6:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	DadosReais	ARMA1	PCM1	PCM2	PCM3	PCM4	PCM5	PCM6	PCM7	PCM8	PCM9	PCM10	PCM11
2	667	441,006	572,848	782,568	583,026	583,565	580,512	581,819	762,287	578,124	574,521	574,38	576,034
3	609	447,974	549,007	709,096	569,522	570,75	563,549	561,956	702,75	566,267	561,925	558,693	559,749
4	557	415,357	540,825	772,729	559,08	560,128	550,4	545,527	754,099	556,115	551,635	548,399	547,127

Figura 6 – Estrutura dos dados no ficheiro excel

Assim sendo, na primeira coluna do ficheiro encontram-se os dados reais, sendo que nas colunas seguintes se encontram as previsões de cada um dos modelos fornecidos.

A leitura dos dados realizou-se com o seguinte comando (Figura 7):

```
> dados<-read.csv("Tese.csv", sep=";", dec=",")
```

Figura 7 – Leitura dos dados utilizados

Para se calcular o erro médio absoluto para cada um dos modelos, é criado um vetor, onde serão guardados os resultados (por exemplo, para o modelo ARMA1, o vetor denomina-se "erroARMA1"). O primeiro passo é subtrair a coluna onde se encontra a previsão do modelo à coluna onde se encontram os dados reais. É feito o módulo desse resultado. Para finalizar calcula-se a média de forma a obter-se o erro médio absoluto. Deste modo, cada vetor irá conter apenas um resultado. A Figura 8 apresenta os comandos descritos anteriormente inseridos no R.

```
> erroARMA1<-mean(abs(dados[,1]-dados[,2]))
> erroPCM1<-mean(abs(dados[,1]-dados[,3]))
> erroPCM2<-mean(abs(dados[,1]-dados[,4]))
> erroPCM3<-mean(abs(dados[,1]-dados[,5]))
> erroPCM4<-mean(abs(dados[,1]-dados[,6]))
> erroPCM5<-mean(abs(dados[,1]-dados[,7]))
> erroPCM6<-mean(abs(dados[,1]-dados[,8]))
> erroPCM7<-mean(abs(dados[,1]-dados[,9]))
> erroPCM8<-mean(abs(dados[,1]-dados[,10]))
> erroPCM9<-mean(abs(dados[,1]-dados[,11]))
> erroPCM10<-mean(abs(dados[,1]-dados[,12]))
> erroPCM11<-mean(abs(dados[,1]-dados[,13]))
```

Figura 8 – Comandos utilizados para o cálculo do erro médio absoluto

De seguida e de forma a visualizar os resultados, basta inserir o nome de cada vetor que contém o erro absoluto de cada um dos modelos (Figura 9). É possível observar-se que o modelo que detém o erro médio absoluto mais baixo é o PCM1 com um erro de 121.1351MW. Logo a seguir encontram-se os modelos PCM4, PCM5, PCM6, PCM9, PCM10 e PCM11 com valores muito próximos a rondar os 123 MW.

Verifica-se ainda que o modelo ARMA1 apresenta o maior erro médio absoluto com um valor de 171.1076MW, o que representa cerca de 50MW acima da melhor performance.

```
> erroARMA1
[1] 171.1076
> erroPCM1
[1] 121.1351
> erroPCM2
[1] 151.3196
> erroPCM3
[1] 157.7261
> erroPCM4
[1] 123.6072
> erroPCM5
[1] 123.3624
> erroPCM6
[1] 123.3
> erroPCM7
[1] 150.1105
> erroPCM8
[1] 157.6353
> erroPCM9
[1] 123.5386
> erroPCM10
[1] 123.4407
> erroPCM11
[1] 123.4655
```

Figura 9 – Visualização dos erros

3.4. Métodos de integração de múltiplos modelos

Serão apresentados resultados para dois modelos de previsão. O primeiro consiste na atualização dos pesos seguindo a fórmula de atualização de pesos proposta por Auer et al. (2002). No segundo é usada a fórmula de atualização de pesos de Charles (2013). Os modelos serão comparados ao nível da performance da previsão por eles realizada. Para que os modelos façam as atualizações dos pesos em cada período utilizando os dados fornecidos, foram usados comandos em comum apresentados na Figura 10.

A primeira etapa consiste na criação de uma função (a função "*updates*"), que depende dos parâmetros: mode, alpha e eta. Estes definem-se como:

- mode – modelo a utilizar (1 no caso do modelo Auer et al. (2002) e 2 no caso do modelo Charles (2013));

- alpha – representa a parte da atualização que depende do erro do modelo de assimilação;

- eta – taxa de aprendizagem

```
> updates <- function(file, mode = 2, alpha = 0.2, eta = 0.2) {
+ dados<-read.csv(file, sep=";", dec=",")
+
+ nexpert=ncol(dados)- 1
+
+ n=nrow(dados)-1
+
+ prev <- c()
+ erro.ens <- c()
+ erros <- NULL
+
+ W<-matrix ( 0 , nrow=n, ncol=nexpert)
+
+ W[1,]<-rep(1/nexpert, nexpert)
+
+ for(t in 1:n) {
+ prev <- c(prev, sum(W[t,]*dados[t,2:13]))
+ erros <- rbind(erros,abs(dados[t,2:13] - dados[t,1]))
+ erro.ens <- c(erro.ens, abs(dados[t,1]-prev[t]))
```

Figura 10 – Parte comum dos comandos para os modelos de atualização de pesos

Dentro da função, é usado um comando para leitura dos dados que é um ficheiro de formato csv. É definido o número de modelos como sendo "ncol(dados)-1", pois cada coluna do ficheiro representa um modelo com exceção da coluna dos dados reais. O número de períodos é definido como "nrow(dados)-1" pois a primeira linha do ficheiro serve para definir o que representa cada coluna. São criados vetores para as previsões ("prev"), para o erro do conjunto ("erro.ens") e para o erro individual ("erros") de cada modelo para que possam ser adicionados dados à medida que a função corre. É criada a matriz para os pesos, que vai ter como número de linhas o número de períodos ("n") e como número de colunas igual ao número de modelos ("nexpert").

Considera-se que no primeiro período todos os modelos têm um peso igual, ou seja, a primeira linha do vetor de pesos é igual a 1 a dividir pelo número de modelos.

Em cada momento *t*, é calculada a previsão do conjunto, os erros de cada modelo e o erro do conjunto. A previsão do conjunto é dada por $Y_t = w_{t,1}\beta_{t,1} + w_{t,2}\beta_{t,2} + \cdots + w_{t,i}\beta_{t,i}$, sendo Y_t a previsão do conjunto, $w_{t,i}$ é o peso atribuído ao modelo *i* no momento *t* e $\beta_{t,i}$ é a previsão do modelo *i* no momento *t*. Sabe-se que W[t,] é um vetor que representa os pesos dos modelos para o período *t*, deste modo, basta

multiplicá-lo pela respetiva linha do ficheiro para obter as previsões do modelo de assimilação. É necessários ter em conta que a primeira coluna é ocupada com os dados reais, e portanto, esta não é considerada na elaboração do modelo de assimilação ("dados[t, 2:13]"). Para o cálculo do erro de cada modelo de previsão em cada momento *t*, como foi visto anteriormente (secção 3.3.), basta calcular o módulo da diferença entre os dados reais e cada modelo, no entanto, de modo a simplificar e fazer o cálculo para todos os modelos num só comando é usado a função "rbind", que faz para cada linha a função introduzida. O valor do erro médio absoluto do conjunto em cada período *t*, é dado pelo módulo da diferença entre os dados reais e a previsão do modelo de assimilação.

Esta é a programação em comum aos dois modelos. É de referir que foi estipulado o valor de 0.2 para o valor de alpha e 0.2 à taxa de aprendizagem para ambos os modelo

3.4.1. Atualização dos pesos de Auer et al. (2002)

Como já referido, foi utilizada a fórmula de atualização de pesos proposta em Auer et al. (2002). Deste modo, tem-se que em cada instante *t*, o algoritmo recebe as previsões dos modelos num vetor $x_t = (x_{t,1}, \ldots, x_{t,n})$ e um valor observado y_t. O algoritmo prevê a cada instante *t*, a combinação linear de $\hat{y} = w_t * x_t$, sendo w_t o vetor de pesos para o momento *t*. Depois de recebido o valor correto de y_t, o algoritmo associa um erro de $l_t = \frac{1}{2} |y_t - \hat{y}_t|$.

Este algoritmo atualiza os pesos, de acordo com: $w_{t+1,i} = w_{t,i} \exp\{-\eta \frac{1}{2} |y_t - x_{t,i}|/W_{t,i}\}$, sendo η a taxa de aprendizagem e $W_{t,i}$ o termo de normalização, de forma a transformar w_{t+1} num vector normal.

Para programar este modelo em R, para além da parte comum aos dois modelos referida acima, foram utilizados os seguintes comandos representados na Figura 11. Assim sendo, para cada modelo, o peso no momento seguinte será igual ao peso do momento com uma atualização.

```
+ if (mode == 1) {
+ for (i in 1:nexpert) {
+ cat("t=",t,";i=",i,";W[t,i]=",W[t,i],";erros[t,i]=",erros[t,i],
+ ";prev[t]=",prev[t],";erro.ens[t]=",erro.ens[t],"\n")
+
+  W[t+1,i] <- W[t,i]*(exp(-eta*(1/2)*erros[t,i]))
+
+ }
+ W[t+1,] <- W[t+1,]/sum(W[t+1,])
+ }
```

Figura 11 – Comandos para a atualização de pesos de Auer et al. (2002)

Os resultados obtidos com este modelo de atualização de pesos foram os que se podem observar na Figura 12:

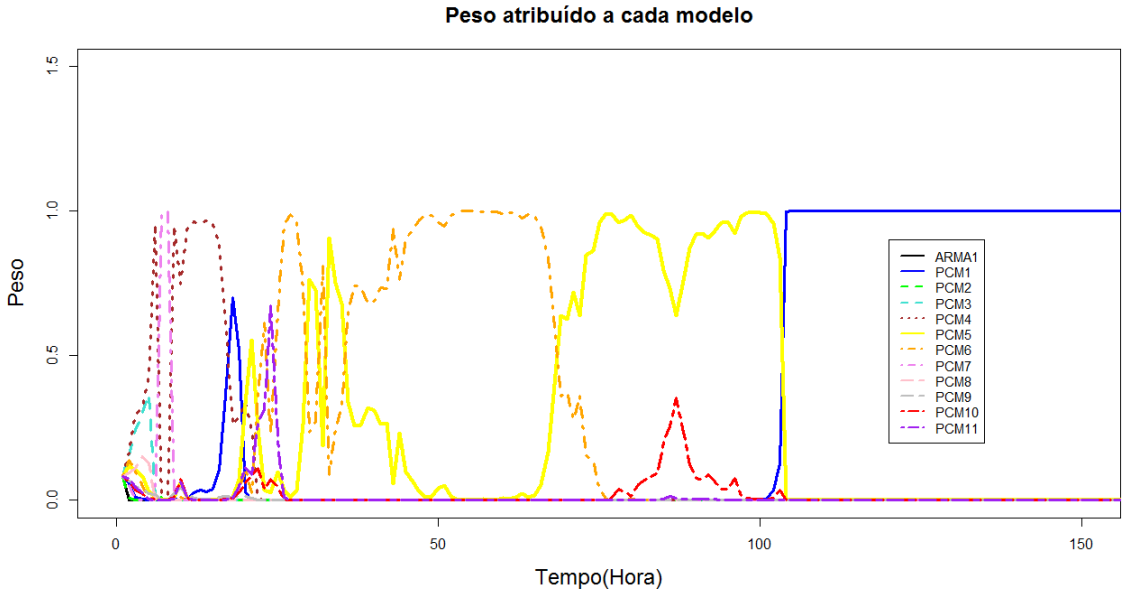


Figura 12 – Peso atribuído a cada modelo com a atualização de pesos de Auer et al (2002)

Através do gráfico que representa o peso atribuído à previsão de cada modelo, é possível verificar-se que a partir de determinado período existe um modelo com o peso total do conjunto, ou seja, a previsão do modelo é a previsão do modelo de assimilação. O modelo escolhido é o PCM1, que como referido na análise dos dados é o que apresenta o menos erro médio absoluto.

Verifica-se que nas primeiras atualizações, existe uma grande variabilidade dos pesos dos modelos. Observa-se que a partir da 25ª hora, é dada uma grade importância à previsão do modelo PCM 6. No entanto, mais à frente ganha importância o modelo PCM 5.

O gráfico representa os resultados até ao momento *t*=150, pois a partir desse momento o modelo de assimilação escolhe o modelo PCM1 com peso igual a 1, sendo assim, analisando só até ao período 150, temos uma melhor perceção daquilo que se passa nos primeiros momentos.

Pode concluir-se igualmente, que a partir de determinado momento, o erro do conjunto é igual ao erro do modelo PCM1 (Figura 13).

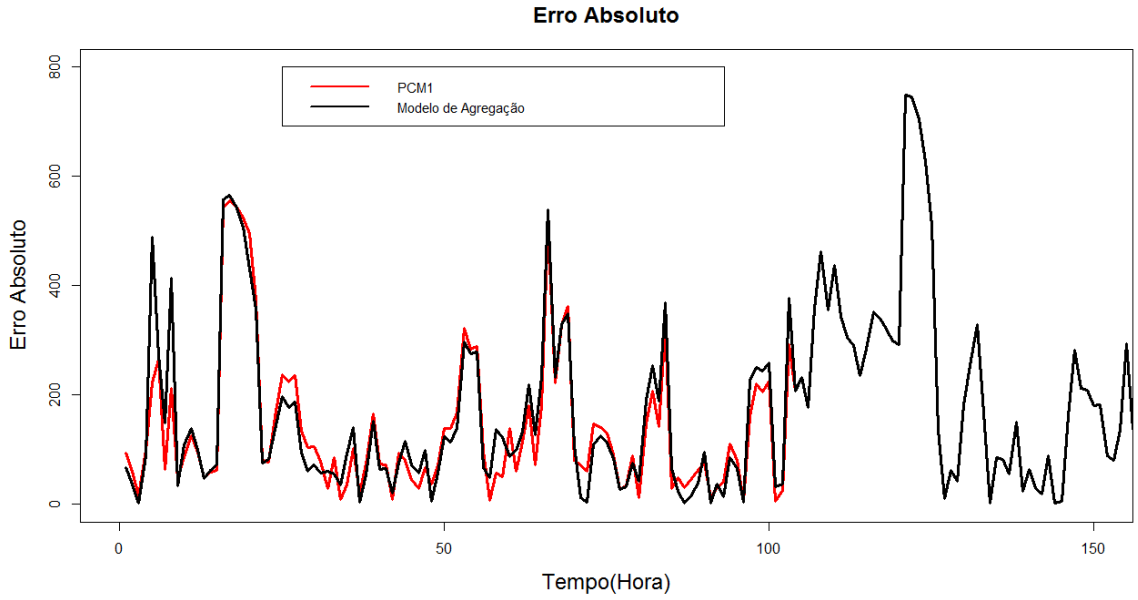


Figura 13 – Evolução do erro absoluto do modelo de agregação (atualização de pesos de Auer et al.2002) e do melhor modelo individual nas primeiras 150 horas.

O gráfico que compara o erro absoluto com o erro do melhor modelo individual mostra que nos primeiros momentos, a performance dos dois modelos apesar de não ser igual, mostra-se muito semelhante. Pode observar-se que o período onde diverge mais a performance do modelo de agregação e do melhor modelo individual, é por volta quando *t* ronda os valores entre 10 e 15. Nesse período o modelo de agregação apresenta um erro superior ao do modelo individual.

Calculando-se o erro médio absoluto, obtém-se os resultados observados na Figura 14:

```
> mean(erro.ens)
[1] 121.226
```

Figura 14 – Resultado do Cálculo do erro do modelo de assimilação

O erro é de 121.226 KW, o que representa um valor ligeiramente maior que o do melhor perito (PCM1). Este facto pode ser justificado pelo tempo que o modelo demora a perceber que o modelo PCM1 apresenta, em média, previsões melhores que o dos outros modelos.

3.4.2. Atualização dos pesos de Charles (2013)

Como já referido, foi utilizada a fórmula de atualização de pesos proposta em Charles (2013). Deste modo, tem-se que em cada instante *t*, o algoritmo recebe as previsões dos modelos num vetor $\mathbf{x_t} = (x_{t,1}, \ldots, x_{t,n})$ e um valor observado $\mathbf{y_t}$. O algoritmo prevê a cada instante *t*, a combinação linear de $\hat{y} = w_t * x_t$. Depois de recebido o valor correcto de y_t, o algoritmo associa um erro de $l_t = |Y_t - \hat{y}_t|$.

Este algoritmo atualiza os pesos, de acordo com: $w_{t+1,i} = \alpha * \frac{W_{t+1}}{N} + (1-\alpha) * w_{t,i} \exp\{-\eta\,|y_t - x_{t,i}|\}$, sendo η a taxa de aprendizagem e α representa a parte da actualização dos pesos que depende do erro do modelo de assimilação.

Para programar este modelo em R, para além da parte comum aos dois modelos referida acima (secção 3.4.), foram utilizados os seguintes comandos apresentados na Figura 15:

```
+ if (mode == 2) {
+ for (i in 1:nexpert) {
+ cat("t=",t,";i=",i,";W[t,i]=",W[t,i],";erros[t,i]=",erros[t,i],
+ ";prev[t]=",prev[t],";erro.ens[t]=",erro.ens[t],"\n")
+
+ W[t+1,i] <- alpha*(sum(W[t+1,i])/nexpert)+(1-alpha)*W[t,i]*exp(-eta*erros[t,i])
+ }
+ W[t+1,] <- W[t+1,]/sum(W[t+1,])
+ }
+ }
```

Figura 15 – Comandos para a atualização de pesos de Charles (2013)

Os resultados obtidos com este modelo de atualização dos pesos foram os representados na Figura 16:

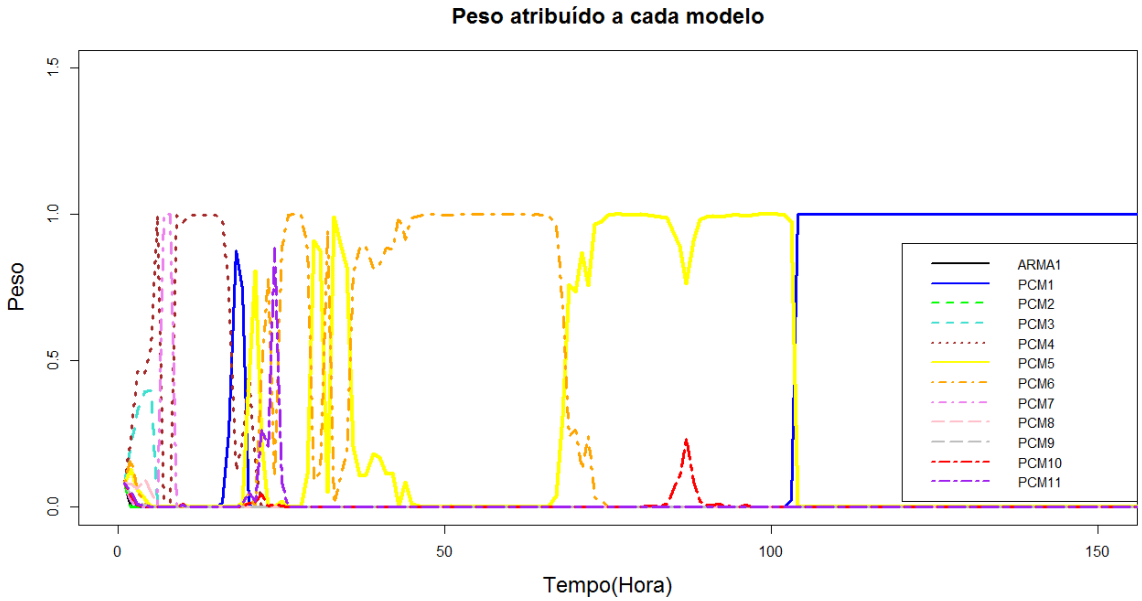


Figura 16 - Peso atribuído a cada modelo com a atualização de pesos de Charles (2013)

Através do gráfico que representa o peso atribuído à previsão de cada modelo, é possível verificar-se que tal como no modelo de atualização de pesos de Auer et al (2002), a partir de determinado período, o modelo PCM1 fica com o total do peso atribuído aos modelos. Como visto anteriormente (secção 3.3.), este modelo é o que apresenta o menor erro médio absoluto.

À semelhança do que aconteceu na análise dos resultados do modelo de Auer et al (2002), foi usado um gráfico onde são apenas visíveis os resultados até ao período t=150, de forma a poder-se observar melhor o que acontece nos primeiros momentos, pois a partir daí o PCM1 tem o peso total do conjunto.

Quanto à evolução dos pesos atribuídos a cada modelo, verifica-se resultados idênticos aos observados no modelo de atualização de pesos de Auer et al (2002).

Pode concluir-se igualmente, que a partir de determinado momento, o erro do conjunto é igual ao erro do modelo PCM1 (Figura 17).

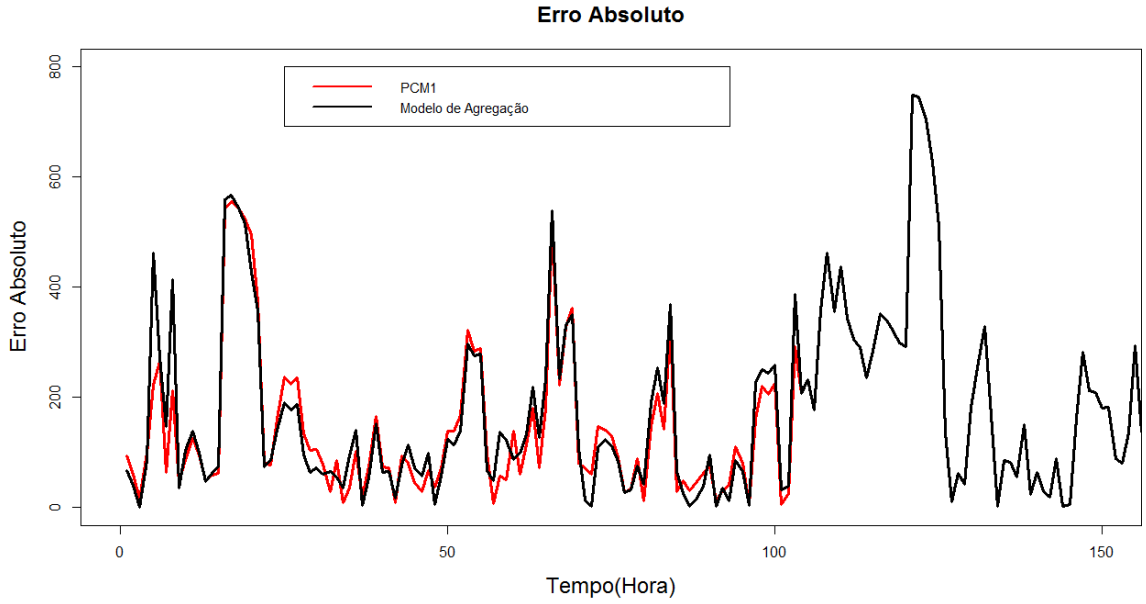


Figura 17 - Evolução do erro absoluto do modelo de agregação (atualização de pesos de Charles (2013) e do melhor modelo individual nas primeiras 150 horas.

Ao observar-se o gráfico representativo da evolução do erro do modelo de agregação com a atualização de Charles (2013) e do erro do melhor modelo individual (PCM1), verifica-se em primeiro lugar resultados extremamente parecidos com o modelo de atualização de pesos de Auer et al (2002). É de salientar, que tal como o anterior (secção 3.4.1.) nos valores de *t* em torno de 10 e 15, o modelo de agregação apresenta um erro superior. No entanto, observa-se também que à semelhança do modelo anterior (secção 3.4.1.), em torno de *t*=25, o modelo PCM1 tem um erro superior ao do modelo de agregação.

Calculando-se o erro médio absoluto, obtém-se o *output* representado na Figura 18:

```
> mean(erro.ens)
[1] 121.2257
```

Figura 18 - Resultado do Cálculo do erro do modelo de assimilação

O erro é de 121.2257 KW, o que representa um valor ligeiramente maior que o do melhor perito (PCM1). No entanto, este erro é ligeiramente inferior ao verificado pelo modelo de atualização de pesos de Auer et al (2002). Este erro confirma que os resultados dos dois modelos de agregação apesar de idênticos, não são iguais.

3.4.3. Verificação

Ao observarmos o gráfico do erro absoluto médio mensal[2] (Figura 19), verifica-se que o modelo PCM1 tem valores de erro mais baixos na maioria do tempo. No entanto, verifica-se que existe um período em que esse facto não se verifica.

No gráfico, o erro do modelo PCM1, está assinalado a cor (vermelho), ao contrário dos outros modelos de forma a realçar a sua evolução, para uma melhor comparação com os restantes.

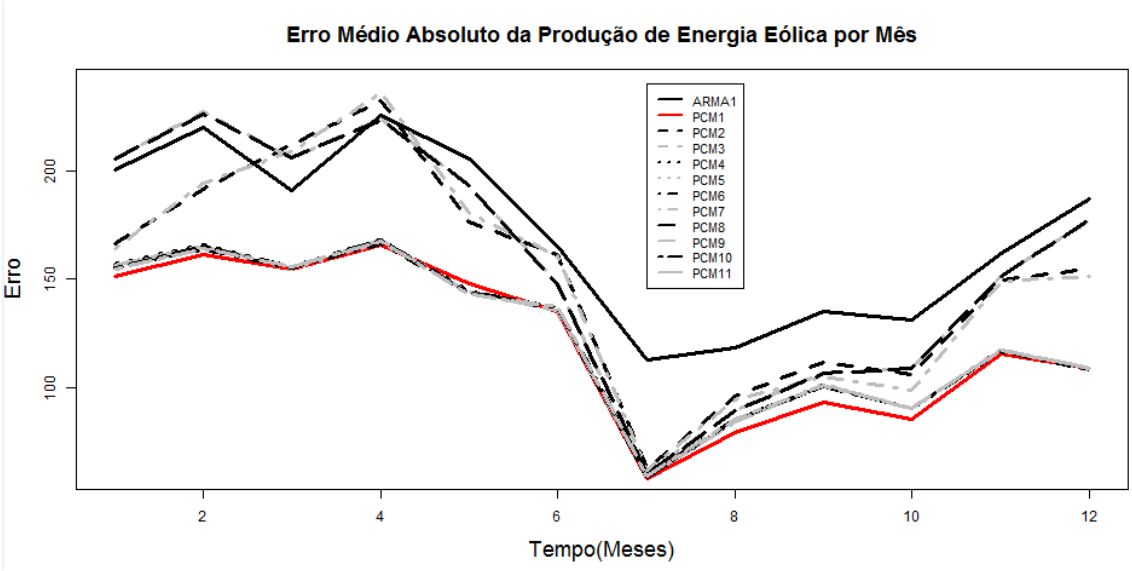


Figura 19 - Erro Médio Absoluto da Produção de Energia Eólica por Mês

No mês de Maio (Mês 5 dos dados), existem modelos com um erro médio absoluto de previsão mais baixo que o do modelo PCM1. Neste sentido, é necessário verificar se utilizando apenas os dados de esse mês, o modelo escolhido seria outro.

É de salientar que nos dois primeiros meses, o modelo PCM1 é claramente o modelo com o erro médio absoluto mais baixo. Resultados voltam a ser inequívocos nos meses 8, 9 e 10.

[2] Foram utilizados os dados mensais de forma a facilitar a leitura do gráfico.

Durante o mês de Maio, os modelos apresentam os erros médios absolutos representados na Figura 20:

```
> erroARMA1
[1] 205.3439
> erroPCM1
[1] 147.8458
> erroPCM2
[1] 176.5256
> erroPCM3
[1] 193.2433
> erroPCM4
[1] 144.322
> erroPCM5
[1] 143.9233
> erroPCM6
[1] 144.1503
> erroPCM7
[1] 180.544
> erroPCM8
[1] 193.1236
> erroPCM9
[1] 143.8981
> erroPCM10
[1] 143.251
> erroPCM11
[1] 143.099
```

Figura 20 – Visualização dos erros do mês de Maio

Deste modo, observa-se que o modelo de previsão que apresenta o erro médio absoluto mais baixo durante o mês de Maio é o PCM11, seguido pelos modelos PCM10, PCM9 e PCM5.

Verifica-se, portanto, que o modelo de previsão PCM1 não é o melhor perito ao longo do mês de maio, tendo um erro de 147.8458KW a comparar com um erro de 143.099KW do melhor modelo (PCM11).

Deste modo, procede-se à análise dos resultados dos modelos de atualização de pesos analisados.

3.4.3.1. Verificação com a atualização dos pesos de Auer et al. (2002)

Os resultados obtidos com este modelo de atualização de pesos foram os da Figura 21:

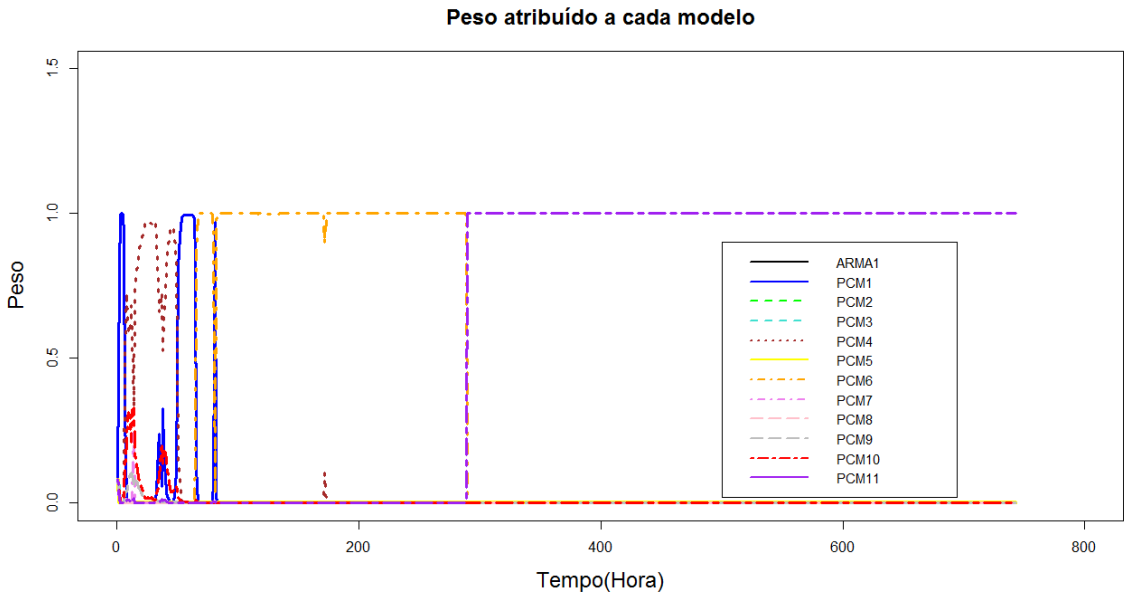


Figura 21 – Peso atribuído a cada modelo durante o mês de Maio com a atualização de pesos de Auer et al. (2002)

Verifica-se que a partir de determinado momento, é atribuído ao modelo PCM11 peso igual a 1. Este modelo, como foi visto anteriormente é o que apresenta menor erro médio absoluto durante o período considerado.

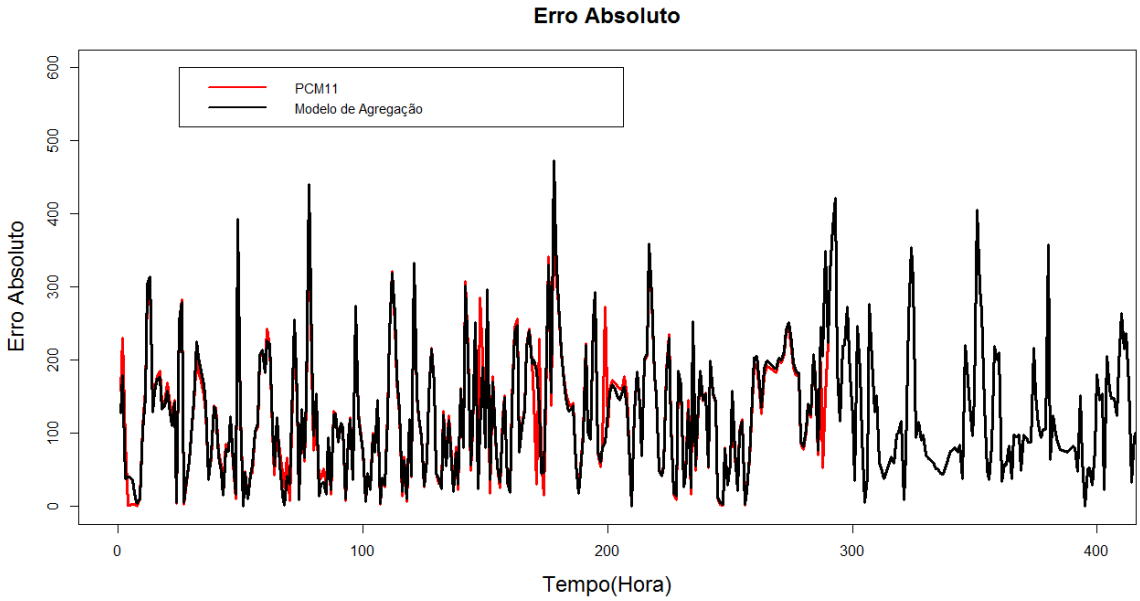


Figura 22 – Erro absoluto modelo durante o mês de Maio com a atualização de pesos de Auer et al. (2002)

Fazendo uma comparação da evolução do erro absoluto do modelo de agregação com o melhor modelo individual durante o período considerado (Figura 22), para além de observar que a partir de determinado momento o erro é igual, é possível concluir-se também que existem dois períodos com maiores divergências de resultados. O primeiro é quando o *t* está entre 150 e 200. Esse é o período em que o modelo PCM6 é dominante. O segundo período localiza-se já perto do t=300, onde o modelo PCM11 apresenta melhores resultados. Curiosamente coincide com o período em que o modelo de agregação passa a dar peso 1 ao melhor modelo individual do período considerado.

O erro médio absoluto do conjunto no período considerado é apresentado na Figura 23:

```
> mean(erro.ens)
[1] 143.2293
```

Figura 23 – Cálculo do erro médio absoluto

Verifica-se que, mais uma vez, o erro médio absoluto do modelo de assimilação é ligeiramente superior ao melhor modelo individual.

3.4.3.2. Verificação com atualização dos pesos de Charles (2013)

Os resultados obtidos com este modelo de atualização de pesos são os observados na Figura 24:

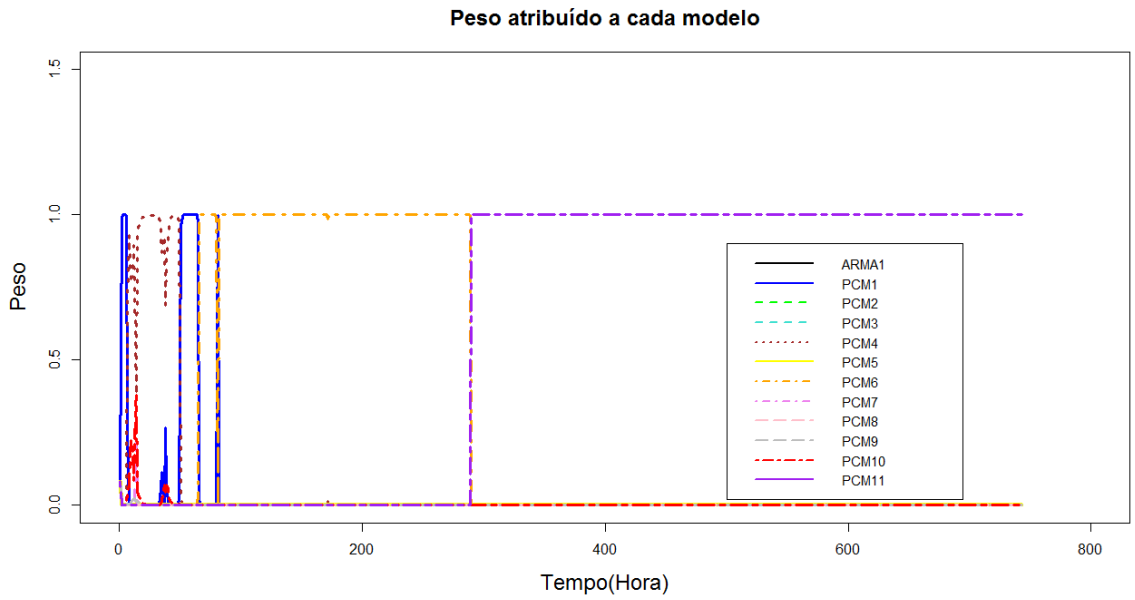


Figura 24 - Peso atribuído a cada modelo durante o mês de Maio com a atualização de pesos de Charles (2013)

Verifica-se mais uma vez, resultados idênticos aos do modelo de atualização de pesos de Auer et al (2002), sendo o modelo PCM11, o escolhido a partir de determinado momento. No entanto, notam-se ligeiras diferenças por volta do período 50, sendo que o modelo de Auer et al. (2002) dá um peso superior ao modelo PCM10 e inferior ao PCM4 em relação ao modelo de Charles (2013). Já perto do período *t*=200, o modelo de Auer et al (2002) atribuí um peso maior ao modelo PCM4, e menor ao modelo PCM6, em relação ao de Charles (2013). Por outro lado, observa-se que perto de *t*=300, ambos os modelos passam a dar peso 1 ao modelo PCM11 (melhor modelo individual no perí odo considerado) e peso 0 ao modelo PCM6.

Na Figura 25, observa-se mais uma vez resultados muito semelhantes em termos de erro absoluto ao observado no modelo anterior (secção 3.4.3). Este modelo de agregação com a atualização de pesos de Charles (2013) também escolhe o modelo PCM11 em detrimento do modelo PCM6 perto de *t*=300.

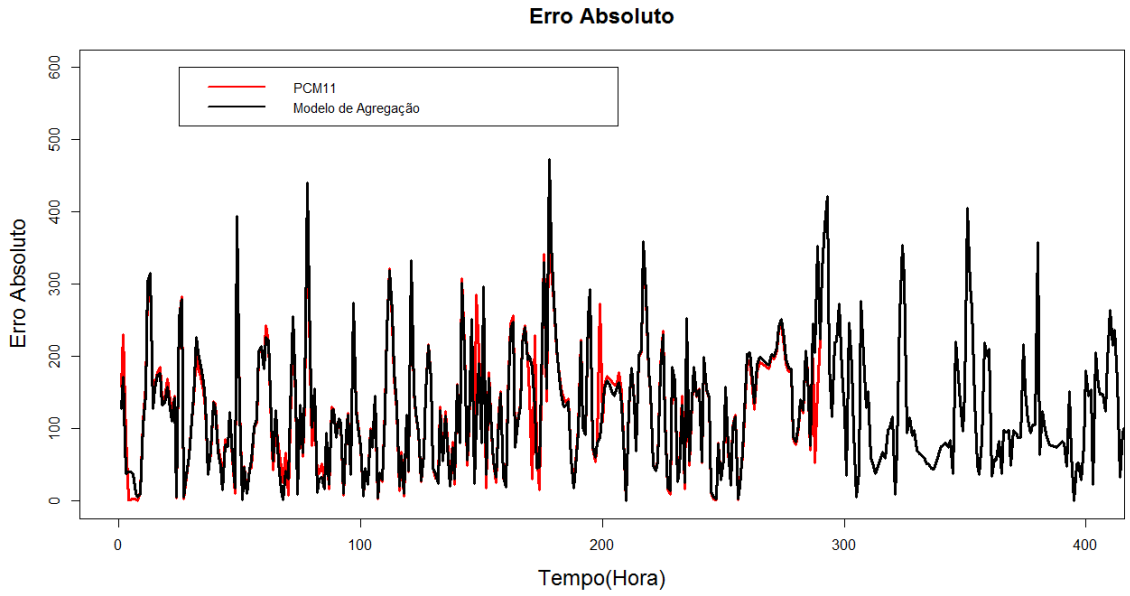


Figura 25 - Erro absoluto modelo durante o mês de Maio com a actualização de pesos de Charles (2013)

Quanto ao erro do conjunto, esta apresenta o resultado da Figura 26:

```
> mean(erro.ens)
[1] 143.2181
```

Figura 26 – Cálculo do erro médio absoluto

O erro do conjunto com o modelo de atualização de Charles (2013), é superior ao do melhor modelo individual (PCM11), no entanto ligeiramente inferior ao do modelo Auer et al. (2002).

Com esta verificação foi possível verificar-se que a agregação de previsões de múltiplos modelos com as atualizações de pesos de Auer et al. (2002) e de Charles (2013) acabam por escolher após os primeiros períodos, o melhor modelo individual.

4. Conclusão

O trabalho realizado teve como objetivo aplicar um modelo de assimilação, capaz de combinar as previsões dos vários modelos à previsão de energia eólica. O objetivo seria provar que a combinação de múltiplos modelos obtém um erro médio absoluto inferior ao do melhor modelo individual. No entanto, observou-se que a combinação dos modelos, para esta base de dados apresenta um erro médio absoluto ligeiramente superior ao do melhor perito individual. Esta situação pode ser justificada com o facto de um dos modelos de previsão (PCM1) apresentar ao longo do período dos dados um erro médio absoluto inferior. Contudo, verificou-se que no mês de Maio, tal não se verificava. Deste modo, de forma a averiguar-se que os modelos estavam de facto a atualizar os pesos, estes foram aplicados apenas para os dados do mês de Maio, sendo que foi observado que o modelo escolhido já não foi o modelo PCM1, mas sim o modelo com a melhor performance durante esse período de tempo (PCM11). Ambos os modelos de atualização de pesos, acabam por escolher sempre o melhor modelo e atribuem-lhe peso igual a 1. No entanto, o número de períodos que esta atualização demora, prejudica o erro médio absoluto do conjunto.

A vantagem de um modelo de agregação, neste caso em particular que ele acaba sempre por escolher o melhor perito individual é a identificação automática do melhor modelo individual do período considerado.

Um inconveniente dos modelos que representa uma limitação a este trabalho é que como existe um modelo que prevê na maioria dos períodos melhor que os restantes, o modelo de agregação de previsões acaba sempre por escolher a previsão desse mesmo. O tempo que a atualização dos pesos demora a dar peso 1 a esse modelo piora ligeiramente a previsão em relação ao melhor modelo individual. É de referir também que a atualização de pesos funciona muito lentamente, pois caso contrário no mês de Maio, o PCM1 não teria peso igual a 1.

Por fim, pode concluir-se que um modelo de assimilação com atualização dos pesos não é benéfico no caso em que haja um modelo dominante, ou seja, um modelo que tem constantemente um erro inferior aos restantes, pois quando isso acontece, é

atribuído peso 1 a esse modelo, no entanto o tempo que o modelo demora a dar essa importância ao melhor modelo individual prejudica a performance do modelo de assimilação.

Um trabalho futuro a ser realizado é a criação de uma fórmula de atualização dos pesos dos modelos que se adeque da melhor forma aos dados de previsão de energia eólica e consequentemente seja capaz de atualizar de forma mais rápida os pesos dos modelos.

Referências

Aha, D. W. and R. L. Bankert (1995). A Comparative Evaluation of Sequential Feature Selection Algorithms. In Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics, Springer-Verlag:199-206

Arora, S., E. Hazan and S. Kale (2012). "The multiplicative weights update method: a meta algorithm and applications." Princeton.

Auer, P. and C. Gentile (2000). Adaptive and Self-Confident On-Line Learning Algorithms. Proceedings of the Thirteenth Annual Conference on Computational Learning Theory, Morgan Kaufmann Publishers Inc.**:** 107-117.

Blum, A. (1997). "Empirical Support for Winnow and Weighted-MajorityAlgorithms: Results on a Calendar Scheduling Domain." Mach. Learn. **26**(1): 5-23.

Bossavy, A., R. Girard and G. Kariniotakis (2010). "Forecasting Uncertainty Related to Ramps of Wind Power Production." Proceedings of the European Wind Energy Conference & Exhibition **Warsaw**(Poland): April.

Boyle, G. (2007). Renewable Electrecity and the Grid. Earthscan: 1-27.

Breiman, L. (1996). "Bagging predictors " Machine Learning **24**(no.2): 123-140.

Breiman, L. (1996). "Stacked regressions " Machine Learning **24**: 49-64.

Castro, R. M. G. (2003). Introdução à Energia Eólica Energias Renováveis e Produção Descentralizada. Universidade Técnica de Lisboa Instituto Superior Técnico

Catalão, J. P. S., N. M. S. Martins and V.M.F. Mendes (2008) "Redes Neuronais Artificiais para Previsão da Potência Eólica ".

Devaine, M., P. Gaillard, Y. Goude and G. Stoltz (2013). "Forecasting electricity consumption by aggregating specialized experts." Mach. Learn. **90**(2): 231-260.

Ferreira, C., J. Gama, L. Matias, A. Botterud and J. Wang (2010). "A Survey on Wind Power Ramp Forecasting " Argonne National Laboratory **Decision and Information Sciences Division**

Friedman, J. H. (1991). "Multivariate adaptative regression splines " The Annals of Statistics 19 **1**: 1-141.

Geman, S., E. Bienenstock, and R. Doursat (1992). "Neural networks and the bias/variance dilemma." Neural Comput. **4**(1): 1-58.

Giebel, G. (2003). "The State-Of-The-Art in Short-Term Prediction of Wind Power." Project ANEMOS.

Greene, W. H. (2012). Econometric analysis. 5th Edition. 7-10

Haussler, D., J. Kivinen and M. K. Warmuth (1998). "Sequential prediction of individual sequences under general loss functions." IEEE Transactions on Information Theory.

Herbster, M. and M. K. Warmuth (1998). "Tracking the Best Expert." (NN): 1-29.

Hernández-Lobato, D., G. Martínez-Muñoz and A. Suarez (2006). Pruning in Ordered Regression Bagging Ensembles IJCNN'06**:** 1266-1273.

Kolter, J. Z. and M. A. Maloof (2007). "Dynamic Weighted Majority: An Ensemble Method for Drifting Concepts." J. Mach. Learn. Res. **8**: 2755-2790.

Littlestone, N. and M. K. Warmuth (1994). "The weighted majority algorithm." Information and Computation **108(2)**: 212-261.

Gama, J., A. Carvalho, M. Oliveira, A. C. Lorena and K. Faceli (2012). Extração de Conhecimento de Dados. Edições Sílabo.

Mendes-Moreira, J., C. Soares, A. Jorge and J. Sousa (2012). "Ensemble approaches for regression: A survey." ACM Comput. Surv. **45**(1): 1-40.

Merz, C. J. (1998). Classification and regression by combining models, University of California, Irvine**:** 207.

Monard, M. C. and J. A. Baranauskas (2003). Conceitos de aprendizado de máquina. In: Resende 89-114.

Negnevitsky, M. and P. Johnson (2008). Very Short Term Wind Power Prediction: A Data Mining Approach. Pittsburgh, Pa.

Perrone, M. P. and L. N. Cooper (1993). "When Networks Disagree: Ensemble Methods for Hybrid Neural Networks." Chapman and Hall: 126-142.

Roli, F., G. Giacinto and G. Vernazza (2001). Methods for Designing Multiple Classifier Systems. Proceedings of the Second International Workshop on Multiple Classifier Systems, Springer-Verlag**:** 78-87.

Rooney, N., D. Patterson, S. Anand and A. Tsymbal (2004). Dynamic Integration of Regression Models. n Proceedings of the 5 th International Multiple Classifier Systems Workshop. LNCS, Trinity College Dublin, Department of Computer Science**:** 164-173.

Shalev-Shwartz, S. (2007). Online Learning: Theory, Algorithms, and Applications. Submitted to the Senate of the Hebrew University

Thorarinsdottir, T. L. and T. Gneiting (2008). "Probabilistic Forecasts of Wind Speed: Ensemble Model Output Statistics using Heteroskedastic Censored Regression." Technical Report no. 546 **Department of Statistics, University of Washington**.

Vovk, V. (1998). "A game of prediction with expert advice." Journal of Computer and System Sciences.

Zheng, H. and A. Kusiak (2009). "Prediction of Wind Farm Power Ramp Rates: A Data-Mining Approach." Journal of Solar Energy Engineering **131(3)**.

Zhou, Z.-H., J. Wu and W. Tang (2002). "Ensembling neural networks: Many could be better than all." Artificial Intelligence **137**(no.1-2): 239-263.

WEBSITES

http://cseweb.ucsd.edu/~kamalika/teaching/CSE291W11/mar2.pdf acedido em Dezembro de 2013

http://www.prewind.eu/ acedido em Setembro de 2014.

Anexos

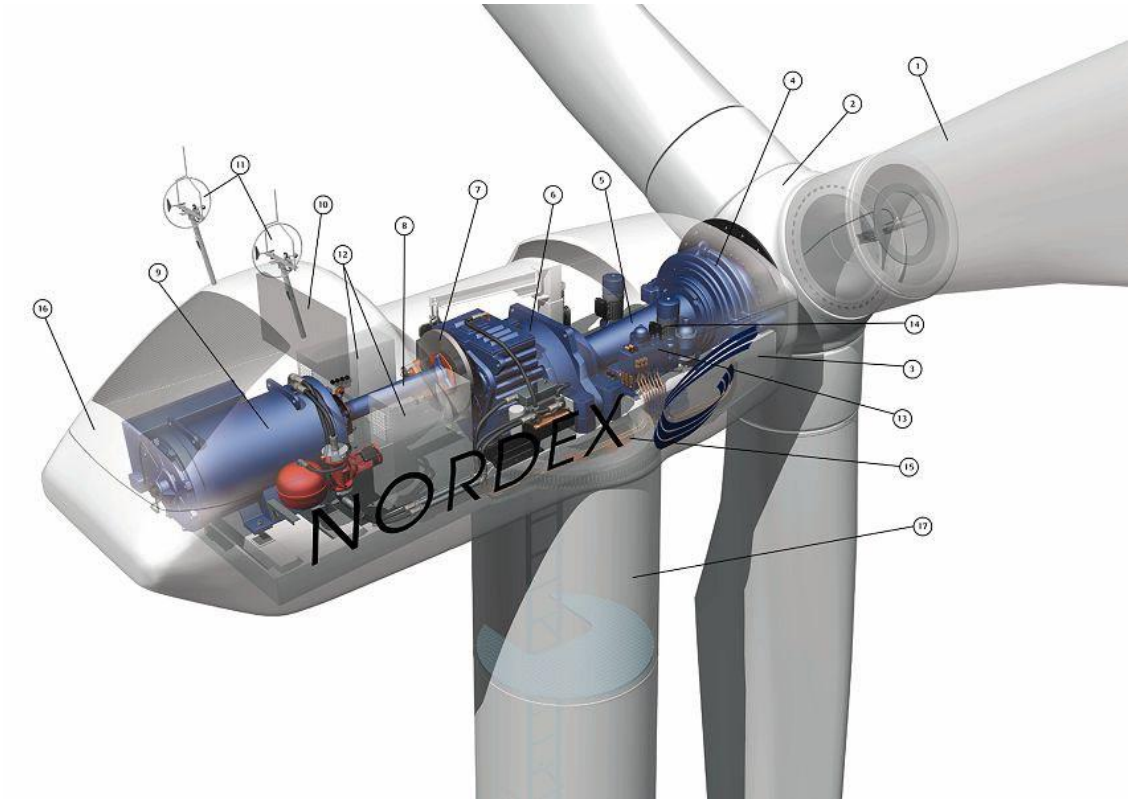
Anexo A: Características elétricas do aerogerador

Segundo Castro (2003), as turbinas eólicas são projetadas para gerarem a máxima potência para uma determinada velocidade do vento. Devido à lei de variação cúbica da potência com a velocidade do vento, para velocidades abaixo de um certo valor (normalmente, cerca de 5m/s, mas depende do local) não interessa extrair energia. Pela mesma razão, para valores superiores à velocidade do vento nominal não é económico aumentar a potência, pois isso obrigaria a robustecer a construção. Apenas se conseguiria tomar partido do aumento de investimento durante poucas horas do ano. Deste modo, a turbina é regulada para funcionar a uma velocidade constante (Castro 2003).

Quando a velocidade do vento se torna perigosamente elevada (superior a cerca de 25-30 m/s), a turbina é desligada por razões de segurança (Castro 2003).

Componentes do Sistema

A figura que se segue mostra as principais componentes de uma turbina eólica do tipo mais comum, isto é, de eixo horizontal e diretamente ligada à rede elétrica (Castro 2003).



Legenda: 1-pás do rotor; 2-cubo do rotor; 3-cabina; 4-chumaceira do rotor; 5-veio do rotor; 6-caixa de velocidades; 7- travão de disco; 8-veio do gerador; 9- gerador; 10-radiador de arrefecimento; 11-anemómetro e sensor de direção; 12-sistema de controlo; 13-sistema de hidráulica; 14-mecanismo de orientação direcional; 15-chumaceira do mecanismo de orientação direcional; 16-cobertura da cabina; 17-torre.

Figura 27- Esquema de uma turbina eólica típica [Nordex]

Fonte: Castro 2013

É possível observar que o sistema de conversão de energia eólica divide-se em três partes: rotor, cabina e torre (Castro 2003).

Rotor

Segundo Castro (2003), a vida útil do rotor está relacionada com os esforços a que fica sujeito e com as condições ambientais em que se insere. A seleção dos materiais usados na construção das pás das turbinas é uma operação delicada: atualmente, a escolha faz-se entre a madeira, os compostos sintéticos e metais. A madeira é o material de fabrico de pás de pequena dimensão (da ordem dos 5m de comprimento). Mais recentemente, a madeira passou a ser empregue em técnicas avançadas de fabrico de materiais compósitos de madeira laminada. Atualmente, há alguns fabricantes a usar estes materiais em turbinas 40m de diâmetro.

Os compósitos sintéticos constituem os materiais mais usados nas pás das turbinas eólicas, nomeadamente, plásticos reforçados com fibras de vidro. Estes materiais são relativamente baratos, robustos, resistem bem à fadiga e são facilmente moldáveis, o que é uma vantagem na fase de fabrico. No grupo dos metais, o aço tem sido usado principalmente nas turbinas de maiores dimensões. Contudo, é um material denso, o que o torna pesado. A tendência atual aponta para o desenvolvimento na direção de novos materiais compósitos híbridos, por forma a tirar partido das melhores características de cada um dos componentes, designadamente sob o ponto de vista do peso, robustez e resistência à fadiga (Castro 2003).

Cabina

Na cabina estão alojados entre outros equipamentos, o veio principal, o travão de disco, a caixa de velocidades (quando existe), o gerador e o mecanismo de orientação direcional. O mecanismo de orientação direcional, constituído essencialmente por um motor que em face de informação recebida de um sensor de direção do vento, roda a nacela e o rotor até que a turbina fique adequadamente posicionada, pois é estritamente necessário que o rotor fique alinhado com a direção do vento, de modo a extrair a máxima energia possível. No cimo da cabina está montado um anemómetro e o respetivo sensor de direção. As medidas de velocidade do vento são usadas pelo sistema de controlo para efetuar o controlo da turbina, nomeadamente, a entrada em funcionamento a partir da velocidade de aproximadamente 5m/s, e a paragem, para ventos superiores a cerca de 25m/s (Castro 2003).

Torre

A torre suporte a *nacelle* e eleva o rotor até uma cota em que a velocidade do vento é maior e menos perturbada do que junto ao solo. As torres modernas podem ter cinquenta e mais metros de altura, pelo que a estrutura tem de ser dimensionada para suportar cargas significativas, bem como para resistir a uma exposição em condições naturais ao longo da sua vida útil, estimada em cerca de 20 anos.

Anexo B: Comandos para a obtenção do erro médio absoluto da previsão de energia eólica por dia:

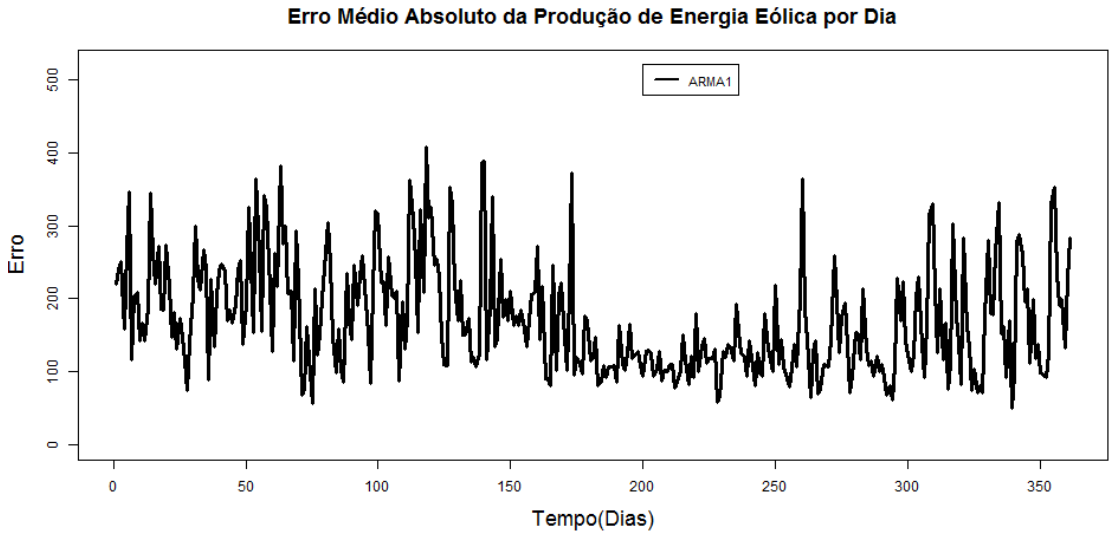
```
Dados<-read.csv("DadosDia.csv", sep=";", dec=",")
media1<-tapply(Dados[,2],Dados$dia, mean)
media2<-tapply(Dados[,3],Dados$dia, mean)
media3<-tapply(Dados[,4],Dados$dia, mean)
media4<-tapply(Dados[,5],Dados$dia, mean)
media5<-tapply(Dados[,6],Dados$dia, mean)
media6<-tapply(Dados[,7],Dados$dia, mean)
media7<-tapply(Dados[,8],Dados$dia, mean)
media8<-tapply(Dados[,9],Dados$dia, mean)
media9<-tapply(Dados[,10],Dados$dia, mean)
media10<-tapply(Dados[,11],Dados$dia, mean)
media11<-tapply(Dados[,12],Dados$dia, mean)
media12<-tapply(Dados[,13],Dados$dia, mean)
tiff(filename="ErroMédio.tiff", width = 960, height = 480)
```

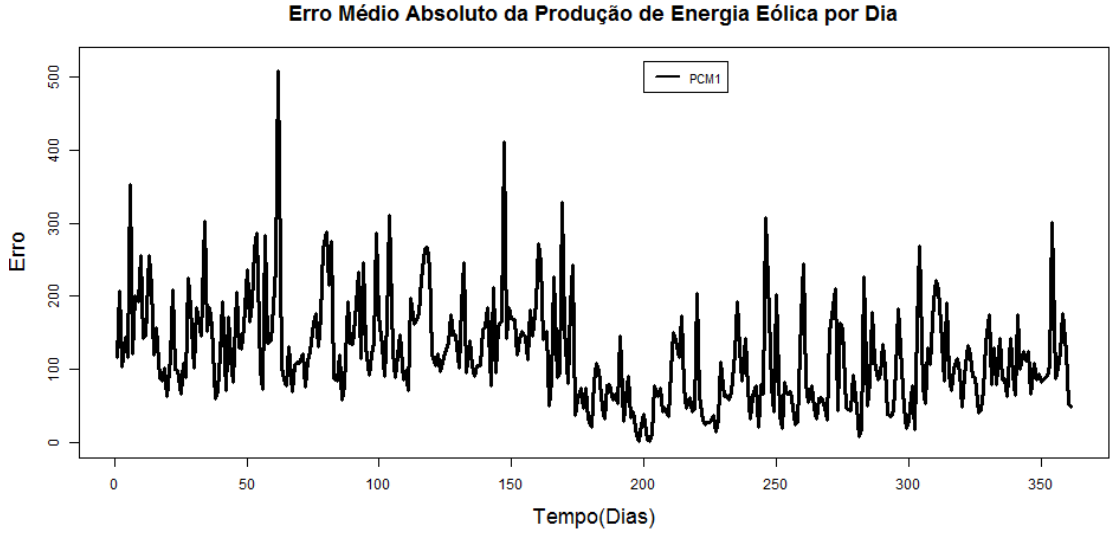
Anexo C: Comandos para a obtenção dos gráficos do erro médio absoluto da previsão de energia eólica por dia:

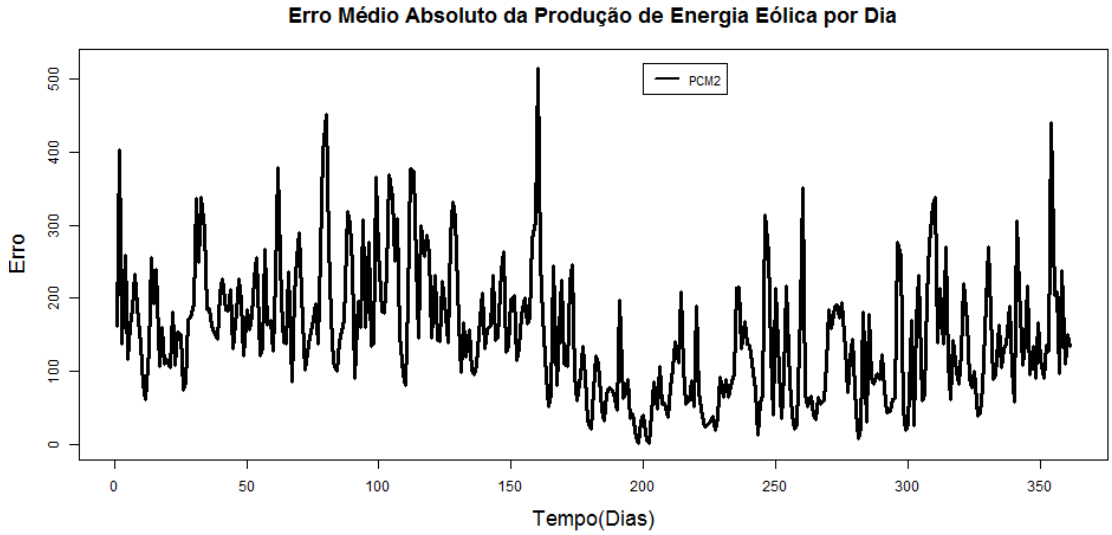
```
plot(media1, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media2, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media3, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media4, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media5, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media6, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media7, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media8, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media9, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media10, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

plot(media11, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)
```

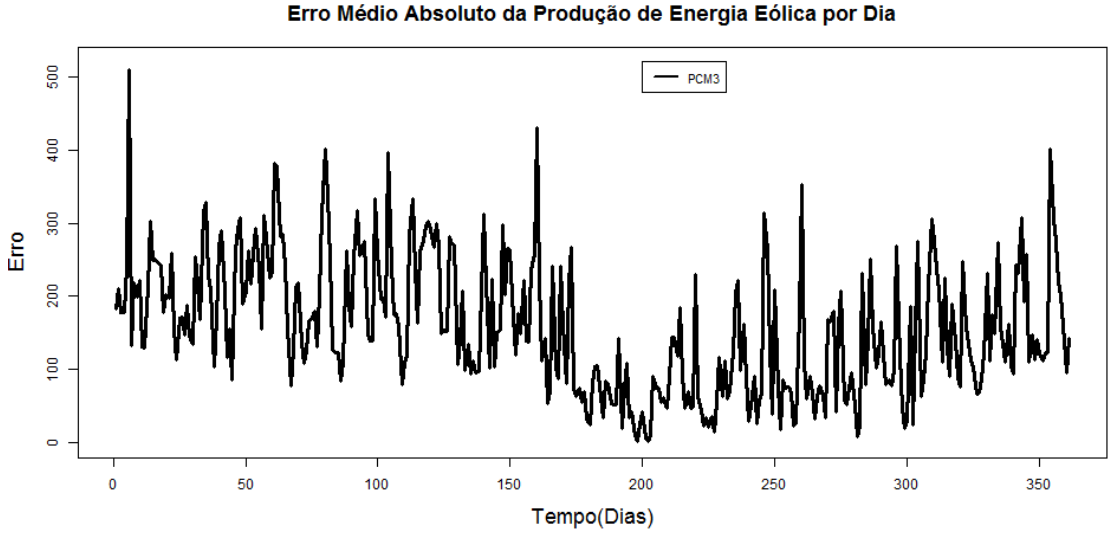
plot(media12, type="l", xlab="Tempo(Dias)", ylab="Erro", main="Erro Médio Absoluto da Produção de Energia Eólica por Dia",ylim=c(0, 520), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

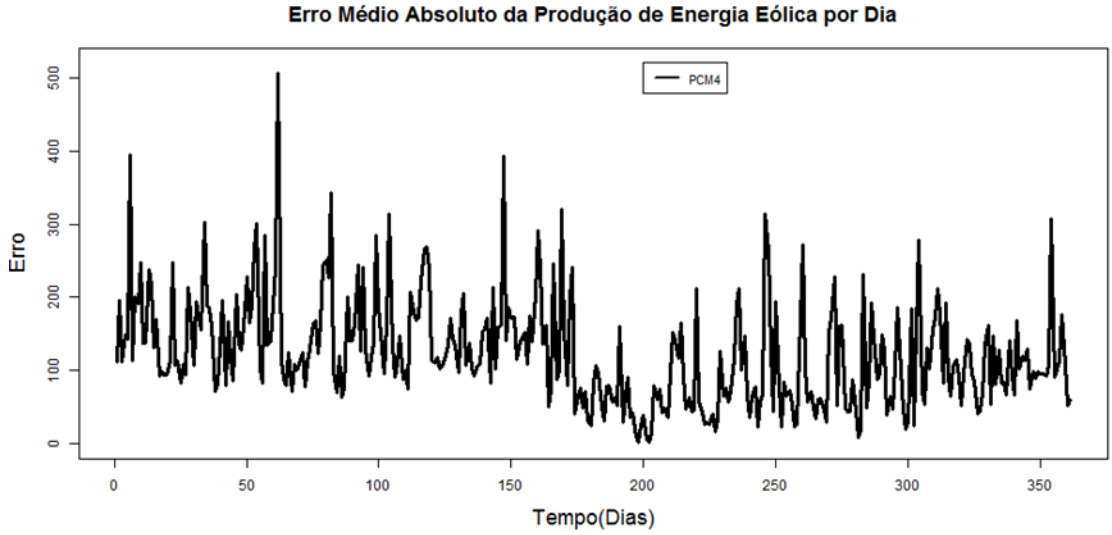
Anexo D: Gráficos do erro médio absoluto da previsão de energia eólica por dia:

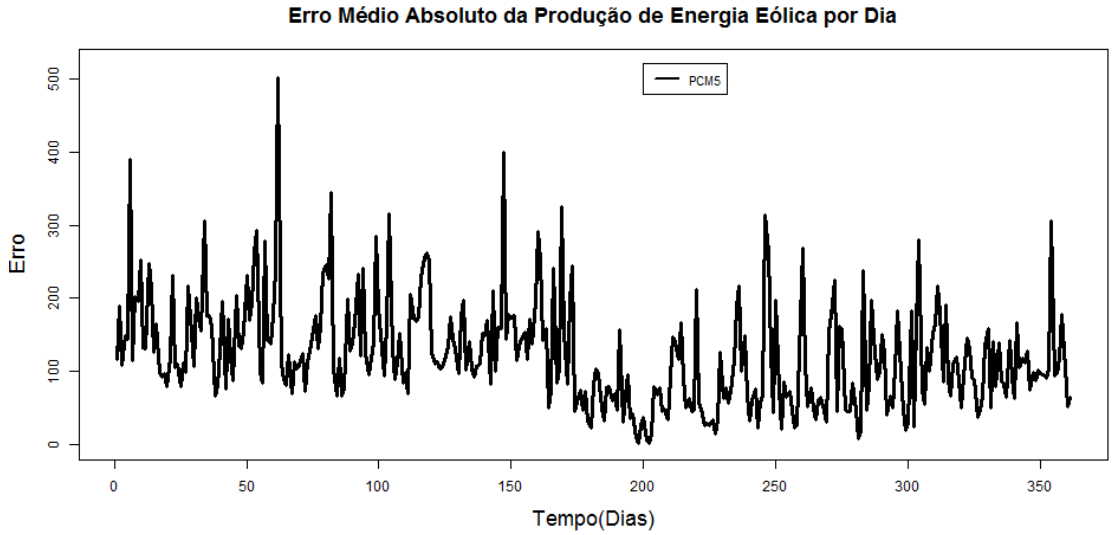


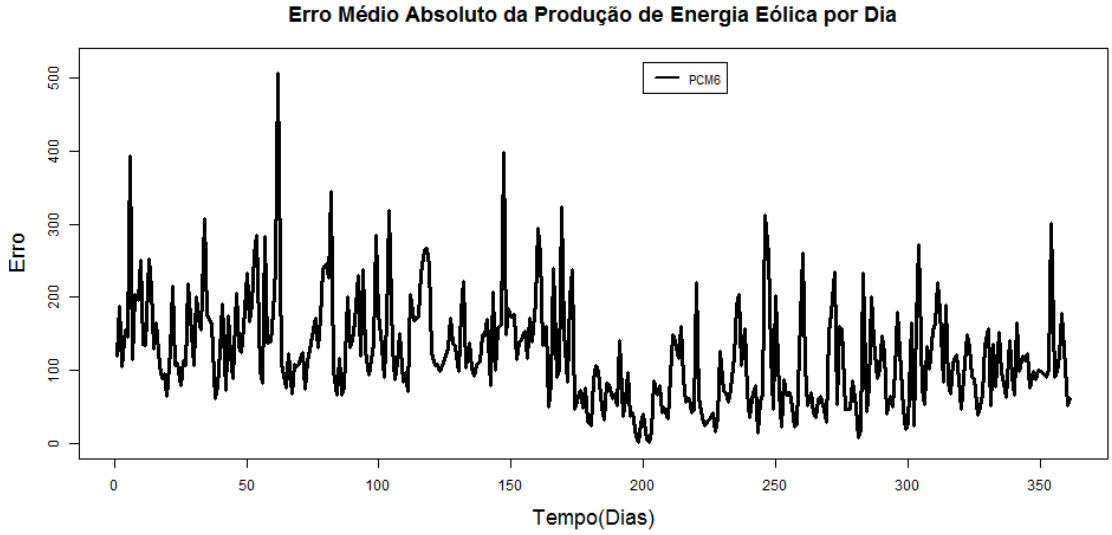


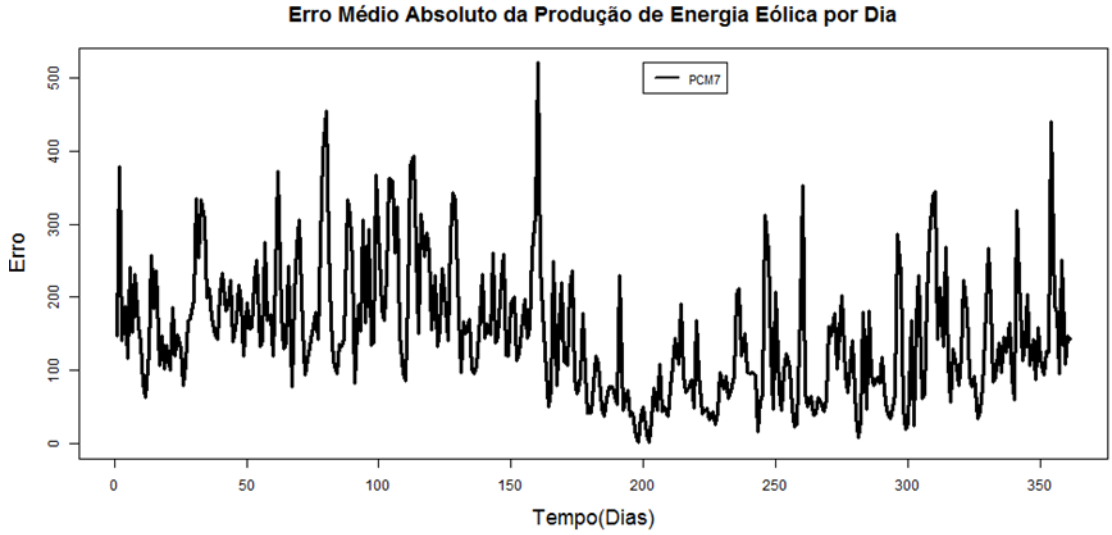

Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM2
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)

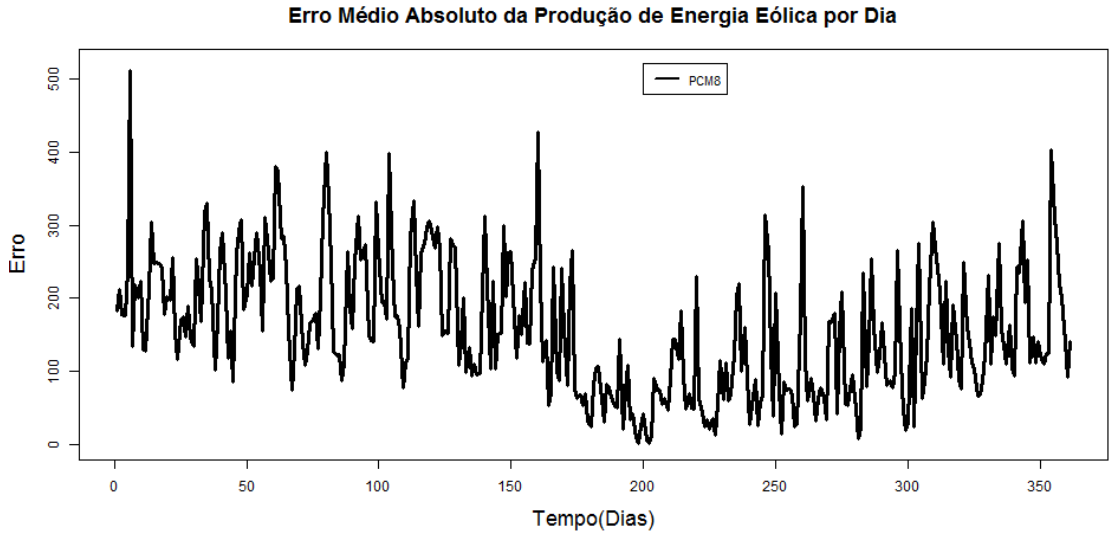

Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM3
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)

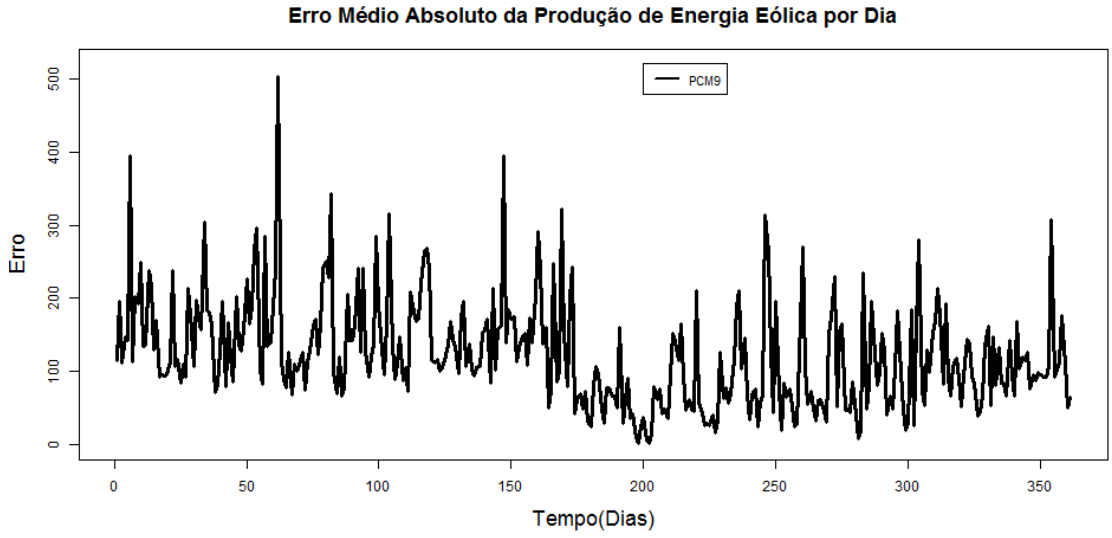

Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM4
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)

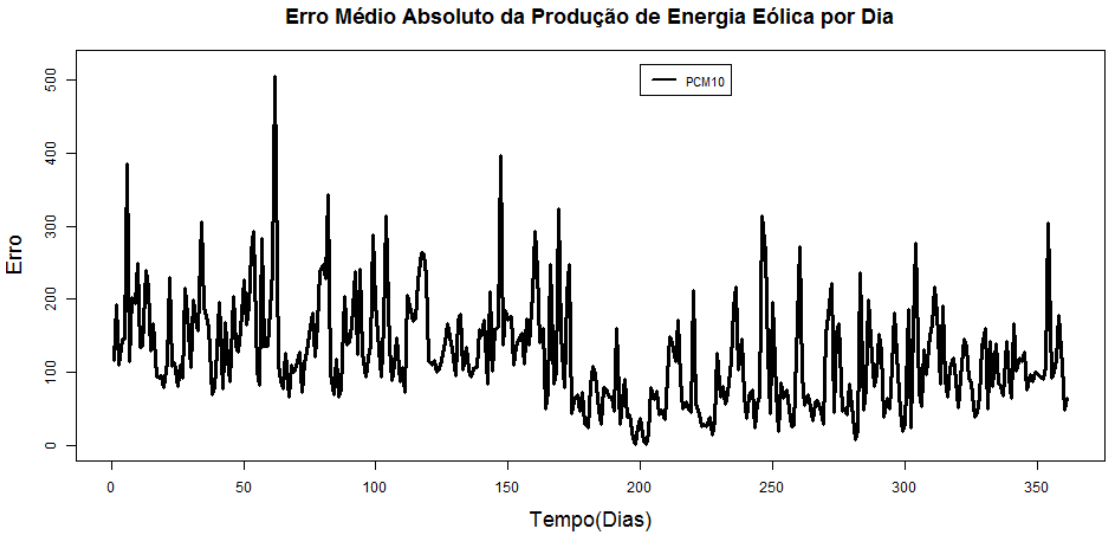

Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM5
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)

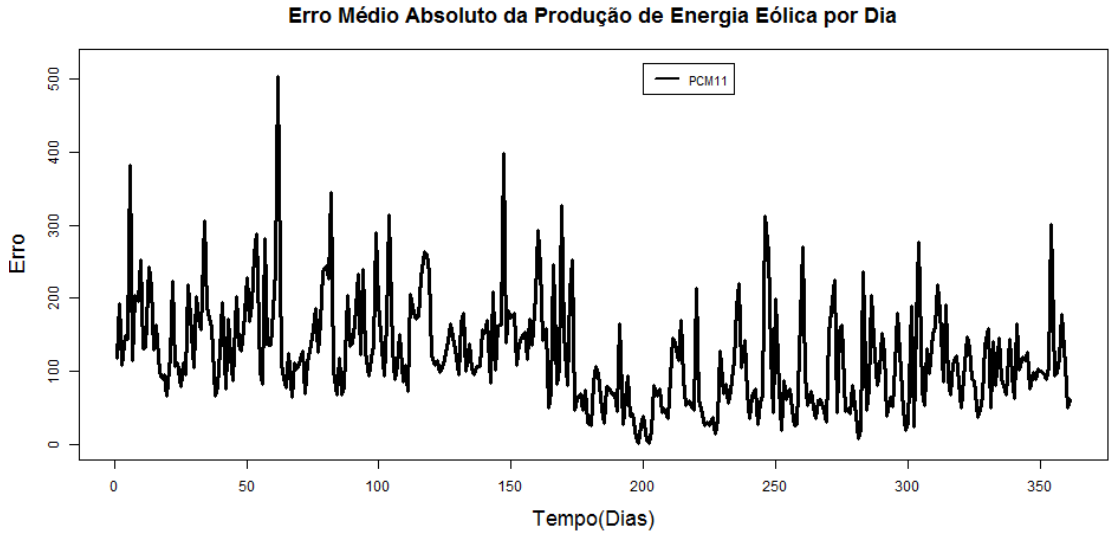

Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM6
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)


Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM7
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)


Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM8
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)


Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM9
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)


Erro Médio Absoluto da Produção de Energia Eólica por Dia
PCM10
Erro
0
100
200
300
400
500
0
50
100
150
200
250
300
350
Tempo(Dias)



Anexo E: Função em R para a Previsão e actualização dos pesos

```
updates <- function(file, mode = 1, alpha = 0.2, eta = 0.2) {
dados<-read.csv(file, sep=";", dec=",")
        nexpert=ncol(dados)- 1
        n=nrow(dados)-1
        prev <- c()
        erro.ens <- c()
        erros <- NULL
        W<-matrix ( 0 , nrow=n, ncol=nexpert)
        W[1,]<-rep(1/nexpert, nexpert)
        for(t in 1:n) {
                prev <- c(prev, sum(W[t,]*dados[t,2:13]))
                erros <- rbind(erros,abs(dados[t,2:13] - dados[t,1]))
                erro.ens <- c(erro.ens, abs(dados[t,1]-prev[t]))
# Loss Update (Tracking the best of many experts)
                if (mode == 1) {
                        for (i in 1:nexpert) {
```

```
cat("t=",t,";i=",i,";W[t,i]=",W[t,i],";erros[t,i]=",erros[t,i],";prev[t]=",prev[t],";erro.ens[t]
=",erro.ens[t],"\n")

                                W[t+1,i] <- alpha*(sum(W[t+1,i])/nexpert)+(1-
alpha)*W[t,i]*exp(-eta*erros[t,i])

                        }

                        W[t+1,] <- W[t+1,]/sum(W[t+1,])

                }

# Loss Update (Adaptative and Self Confidence Online Learning Algorithms)

                if (mode == 2) {

                        for (i in 1:nexpert) {

cat("t=",t,";i=",i,";W[t,i]=",W[t,i],";erros[t,i]=",erros[t,i],";

prev[t]=",prev[t],";erro.ens[t]=",erro.ens[t],"\n")

                                W[t+1,i] <- W[t,i]*(exp(-eta*(1/2)*erros[t,i]))

                        }

                        W[t+1,] <- W[t+1,]/sum(W[t+1,])

                }

        }

}

updates("Tese.csv")
```

Anexo F: Comandos em R para gráficos de representação dos resultados

Gráficos representativos dos pesos de cada modelo

```
plot(W[,1], type="l", xlab="Tempo(Hora)", ylab="Peso", main="Peso atribuído a cada modelo",xlim= c(0,150) , ylim=c(0, 1.5), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

lines(W[,2], col="blue",lty=1,lwd = 3)

lines(W[,3], col="green",lty=2,lwd = 3)

lines(W[,4], col="turquoise",lty=2,lwd = 3)

lines(W[,5], col="brown",lty=3,lwd = 3)

lines(W[,6], col="yellow",lty=1,lwd = 4)

lines(W[,7], col="orange",lty=4,lwd = 3)

lines(W[,8], col="violet",lty=4,lwd = 3)

lines(W[,9], col="pink",lty=5,lwd = 3)

lines(W[,10], col="grey",lty=5,lwd = 3)

lines(W[,11], col="red",lty=6,lwd = 3)

lines(W[,12], col="purple",lty=6,lwd = 3)


legend(x=120,y=0.9,c("ARMA1","PCM1","PCM2","PCM3","PCM4","PCM5","PCM6","PCM7","PCM8","PCM9","PCM10","PCM11"),col=c("black","blue","green","turquoise","brown","yellow","orange","violet","pink","grey","red", "purple"), lty=c(1,1,2,2,3,1,4,4,5,5,6), cex=0.9, lwd = 2)
```

Gráficos representativos dos pesos de cada modelo na verificação

```
plot(W[,1], type="l", xlab="Tempo(Hora)", ylab="Peso", main="Peso atribuído a cada modelo",xlim= c(0,800) , ylim=c(0, 1.5), cex.main = 1.5, col="black",lty=1,cex.lab = 1.5,lwd = 3)

lines(W[,2], col="blue",lty=1,lwd = 3)

lines(W[,3], col="green",lty=2,lwd = 3)

lines(W[,4], col="turquoise",lty=2,lwd = 3)

lines(W[,5], col="brown",lty=3,lwd = 3)

lines(W[,6], col="yellow",lty=1,lwd = 4)

lines(W[,7], col="orange",lty=4,lwd = 3)

lines(W[,8], col="violet",lty=4,lwd = 3)

lines(W[,9], col="pink",lty=5,lwd = 3)

lines(W[,10], col="grey",lty=5,lwd = 3)

lines(W[,11], col="red",lty=6,lwd = 3)

lines(W[,12], col="purple",lty=6,lwd = 3)


legend(x=500,y=0.9,c("ARMA1","PCM1","PCM2","PCM3","PCM4","PCM5","PCM6","PCM7","PCM8","PCM9","PCM10","PCM11"),col=c("black","blue","green","turquoise","brown","yellow","orange","violet","pink","grey","red", "purple"), lty=c(1,1,2,2,3,1,4,4,5,5,6), cex=0.9, lwd = 2)
```

Anexo G: Comandos R para gráficos de comparação do erro absoluto do modelo de agregação com o melhor modelo individual

Gráfico para total dos dados:

```
plot(erros[,2], type="l", xlab="Tempo(Hora)", ylab="Erro Absoluto", main="Erro Absoluto",xlim= c(0,150) , ylim=c(0, 800), cex.main = 1.5, col="red",lty=1,cex.lab = 1.5,lwd = 3)

lines(erro.ens[], col="black",lty=1,lwd = 3)
```

legend(x=25,y=800, c("PCM1","Modelo de Agregação"), col=c("red","black"), lty=c(1,1), cex=0.9, lwd = 2)

Gráfico para o mês 5 (Maio):

plot(erros[,12], type="l", xlab="Tempo(Hora)", ylab="Erro Absoluto", main="Erro Absoluto",xlim= c(0,400) , ylim=c(0, 600), cex.main = 1.5, col="red",lty=1,cex.lab = 1.5,lwd = 3)

lines(erro.ens[], col="black",lty=1,lwd = 3)

legend(x=25,y=600, c("PCM11","Modelo de Agregação"), col=c("red","black"), lty=c(1,1), cex=0.9, lwd = 2)