Nanodegree Engenheiro de Machine Learning

Projeto Capstone Junho/2018

Aluno: Thiago Meireles Grabe

Definição

Visão Geral do Projeto

Em engenharia mecânica o termo confiabilidade é relacionado com a disponibilidade de um equipamento ou sistema em operar em seu estado ótimo, ou seja, em pleno trabalho sem gerar perdas na produção ou operação em que ele foi projetado. Num contexto mais específico, o estudo de confiabilidade pode associar-se à análise de manutenção, determinando instantes ótimos de tempo em que a mesma deve ser realizada. Tal análise tem como objetivo evitar que a falha ocorra, ampliando o tempo de funcionamento da máquina e buscando reduzir os custos.

O tempo de vida útil remanescente (*Remaining Useful Life – RUL*) de determinado sistema, máquina ou componente é definido como a extensão do tempo atual até o final de sua vida útil. Assim, desenvolver um plano de prognóstico de falha que possa monitorar e administrar adequadamente o sistema, estimando o RUL de maneira precisa, é de grande valia. Este plano visa permitir que as ações de manutenção sejam executadas antes que a falha ocorra, impactando positivamente a disponibilidade do sistema, evitando custos envolvidos com manutenção corretiva e relacionados com a indisponibilidade (BEZERRA SOUTO MAIOR *et al.*, 2016).

Tendo em vista a importância de termos uma manutenção preventiva e preditiva nos diversos setores industriais, alguns componentes mecânicos tem sido alvo de estudos importantes utilizando métodos computacionais ou *Data-driven* para reconhecimento de padrões e predição de quando um componente ou sistema irá falhar. Nesse aspecto, itens como rolamentos (ELFORJANI, 2016), filtros analógicos (HU *et al.*, 2015) e Turbinas aeroespaciais (FORNLÖF *et al.*, 2016).

Nesse aspecto, o meu interesse pessoal no assunto se dá pela minha formação acadêmica em engenharia mecânica. As análises convencionais de dados para ativos importantes como os citados acima ainda são precárias e baseadas em planilhas extensas de Excel. Essas planilhas são eficientes até certo ponto, pois quando se pensa em escalabilidade, confiabilidade de modelos e mesmo cenários flexíveis não podemos nos prender à uma ferramenta essencialmente de manipulação de dados manual. Com isso, ao estabelecer um modelo para determinação de RUL em um componente ou sistema mecânico terei uma experiência em manipulação de dados e estabelecer um modelo de predição ou classificação para um *dataset* específico.

Descrição do problema

O problema a ser tratado neste projeto foi apresentado como uma competição internacional chamada *PHM08 Prognostics Data Challenge Dataset* (PROGNOSTICS AND HEALTH MANAGEMENT SOCIETY., [S.d.]). Neste evento o desafio era um *set* de Turbinas aeroespaciais do mesmo tipo. Cada Turbina tem um início de degradação e desgaste diferente, ou seja, os pontos de operação inicial e final não são conhecidos para os Turbinas. Sabe-se apenas que os Turbinas iniciam a operação em condições de operar normalmente e a degradação ocorre ao longo do tempo.

Através desse cenário, o objetivo é encontrar o RUL de cada Turbina aeroespacial e predizer caso o Turbina irá falhar e quando isso ocorrerá após os testes feitos. Essa métrica deve ser feita através dos números de ciclos restantes para cada Turbina após o teste realizado.

Métricas de avaliação

A métrica de avaliação é simples e objetiva. Para o cálculo da RUL, utilizamos a equação abaixo:

$$s = \begin{cases} \sum_{i=1}^{n} e^{-\left(\frac{d}{a_1}\right)} - 1 \text{ for } d < 0 \\ \sum_{i=1}^{n} e^{\left(\frac{d}{a_2}\right)} - 1 \text{ for } d \geq 0, \end{cases}$$

where,
s is the computed score,
n is the number of UUTs,
d = (Estimated RUL – True RUL),
$a_1 = 13$, and $a_2 = 10$.

Para essa avaliação deve-se obter o tempo remanescente para cada linha do dataset e comparar com os dados reais para a obteção do parâmetro “d”. A partir disso, temos uma algebra relativamente simples para se obter o resultado. Quanto mais próximo de zero o valor de “s”, melhor foi a estimativa do modelo criado.

O modelo da métrica apresentado tem o intuíto de penalizar mais as predições atrasadas do que as predições que ocorrem antes da suposta quebra da turbina conforme a figura 01:

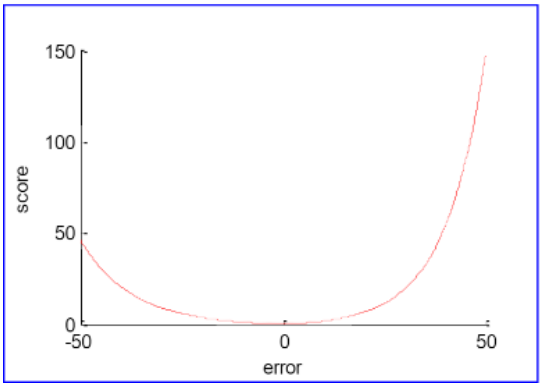

Figura 01: *Score* em funçaõ dos erros associados à predição

Análise do Problema

Exploração dos dados

Os dados para o projeto estão separados por dados de treino e de teste. Os arquivos são de formato *.txt* com os respectivos nomes:

 Dados para treino: *Train.txt*  Dados para teste: *Test.txt*

Em ambos arquivos, vemos 26 colunas de dados ou *features* que descrevem a condição da turbina em um determinado ponto ou uma referência para o teste executado no equipamento. Para o conjunto de dados de treino são oferecidas 45.918 linhas de dados e para o conjunto de teste 29.820 linhas de dados.

Há duas colunas iniciais de referência para os testes. Nelas há informações da turbina em teste ("*Unit Number"*) e outra que detalha o número de ciclos da determinada turbina (*"Cycle Number*).

As três próximas colunas trazem informações do teste em si, ou seja, os parâmetros de operação das turbinas durante o teste: "*Setting 01"*, "*Setting 02"* e "*Setting 03".* Há ainda 21 colunas que são dados de sensores que mostram as condições das turbinas durante a operação do teste.

Cada coluna de dados apresentam as seguintes características:

- **'unit':** Unidade amostral da turbina (*int64*)
- **'cycle_num':** Ciclo atual da unidade amostral (*int64*)
- **'setting1':** Parâmetro um de operação da turbina (*float64*)
- **'setting2':** Parâmetro dois de operação da turbina (*float64*)
- **'setting3':** Parâmetro três de operação da turbina (*float64*)
- **'sensor1':** Dado do sensor (*float64*)
- **'sensor2':** Dado do sensor (*float64*)
- **'sensor3':** Dado do sensor (*float64*)
- **'sensor4':** Dado do sensor (*float64*)
- **'sensor5':** Dado do sensor (*float64*)
- **'sensor6':** Dado do sensor (*float64*)
- **'sensor7':** Dado do sensor (*float64*)
- **'sensor8':** Dado do sensor (*float64*)
- **'sensor9':** Dado do sensor (*float64*)
- **'sensor10':** Dado do sensor (*float64*)
- **'sensor11':** Dado do sensor (*float64*)
- **'sensor12':** Dado do sensor (*float64*)
- **'sensor13':** Dado do sensor (*float64*)
- **'sensor14':** Dado do sensor (*float64*)
- **'sensor15':** Dado do sensor (*float64*)
- **'sensor16':** Dado do sensor (*float64*)
- **'sensor17':** Dado do sensor (*int64*)
- **'sensor18':** Dado do sensor (*int64*)
- **'sensor19':** Dado do sensor (*float64*)
- **'sensor20':** Dado do sensor (*float64*)
- **'sensor21':** Dado do sensor (*float64*)

Visualização Exploratória

Através de uma análise gráfica básica é possível entender como os dados estão distribuídos. Podemos observar a distribuição dos últimos ciclos de cada unidade através da figura 2 abaixo:

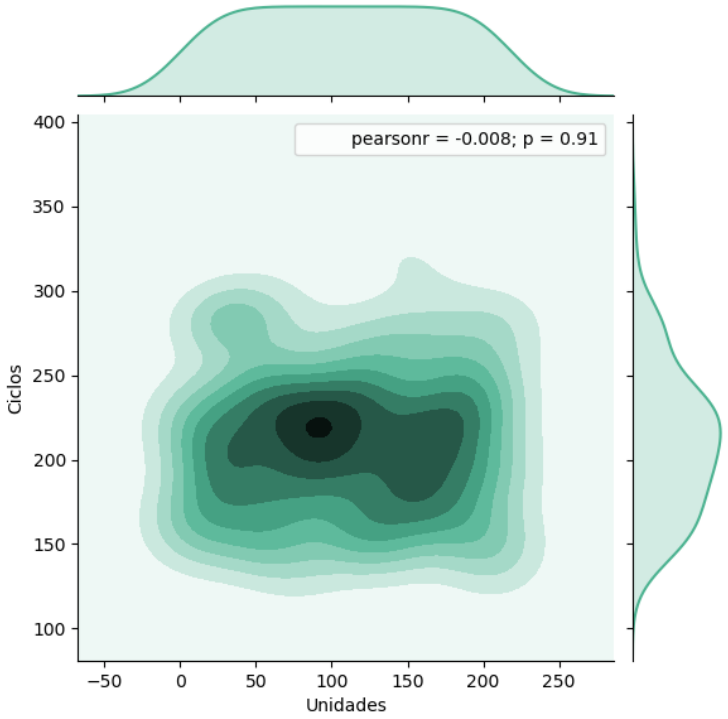

Figura 02: Mapa de correlação entre os ciclos e as unidades

Neste gráfico observamos que as turbinas apresentam o seu fim de vida (ciclo máximo) na região central do gráfico, entre 200 e 230 ciclos. Além disso, o coeficiente da correlação de Pearson apresentado foi de 0.91 o que indica uma boa correlação entre as *features* apresentadas.

Os dados apresentam três colunas de condição de operação. Essas condições podem representar alguma alteração na vida útil das turbinas e será discutido mais a frente. Contudo, faz-se necessário uma análise do comportamento dessas condições de operação em função dos ciclos conforme figura 3 para o parâmetro 1, figura 4 para o parâmetro 2 e figura 5 para o parâmetro 5.

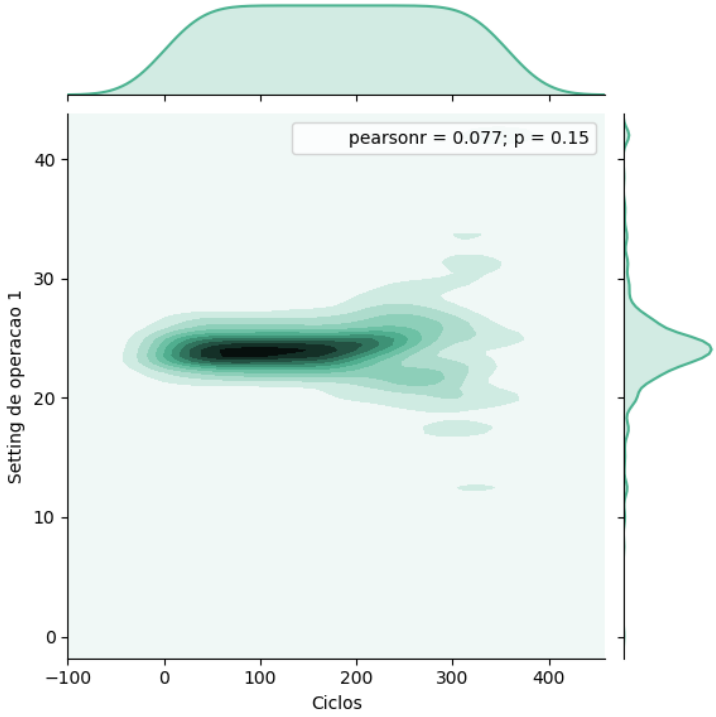

Figura 03: Correlação entre parâmetro 1 e número de ciclos

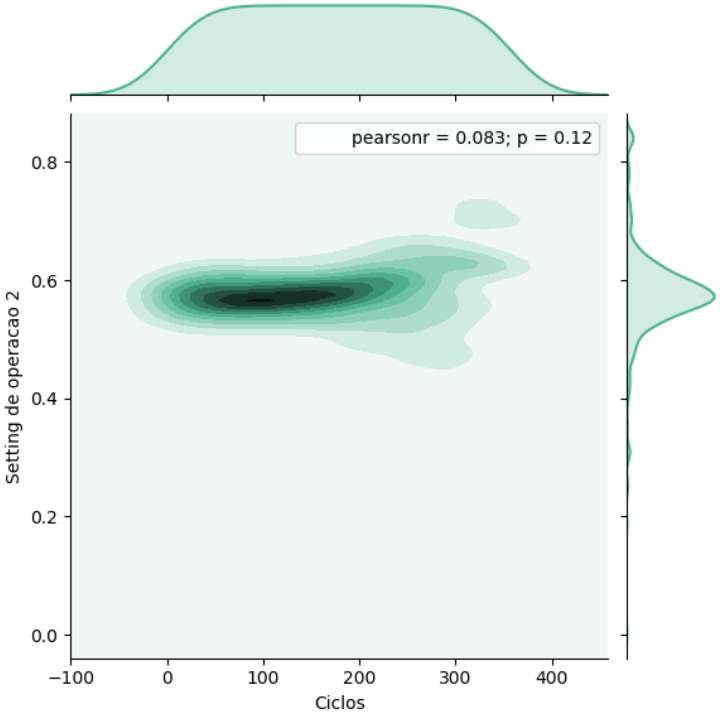

Figura 04: Correlação entre parâmetro 2 e número de ciclos

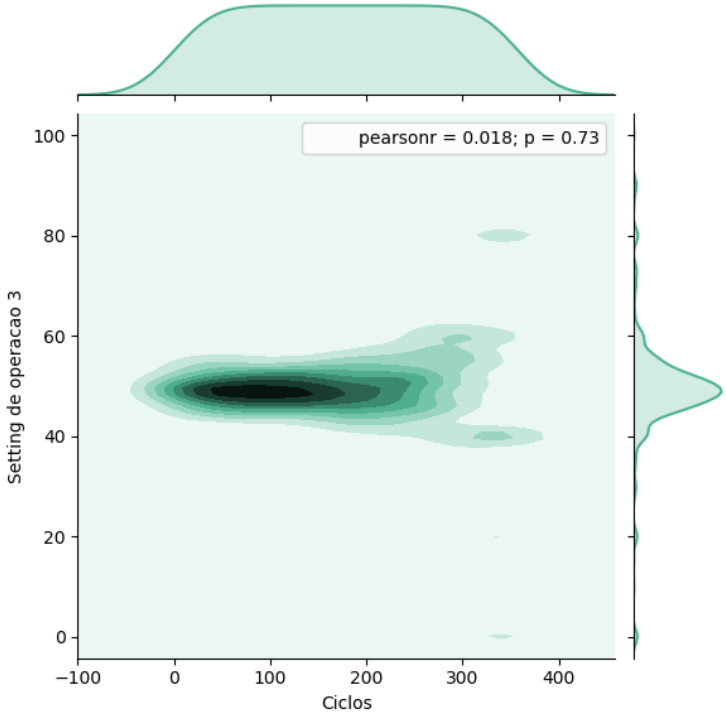

Figura 05: Correlação entre parâmetro 3 e número de ciclos

Através dos gráficos acima pode-se observar que os parâmetros 1 e 2 apresentam correlação desprezível com o número de ciclos (análise pela correlação de Pearson), mas o parâmetro 3 já se relaciona com um bom coeficiente. Essas análises indicam que podemos utilizar esses três parâmetros em algum processamento dos dados para extrair informações através de agrupamentos ou na redução de *features* para melhorar o modelo de predição.

Apersar de grande importância para a degradação precoce ou não das turbinas, os parâmetros de operação não devem representar o ponto chave para a predição dos ciclos

restantes de cada turbina. Se faz necessário uma análise dos dados dos sensores e entender o comportamento de cada sensor. Para essa análise, é proposto um mapa de correlação para uma visão, ainda que geral, de como os sensores podem ou não apresentar uma importânica na análise dos dados.

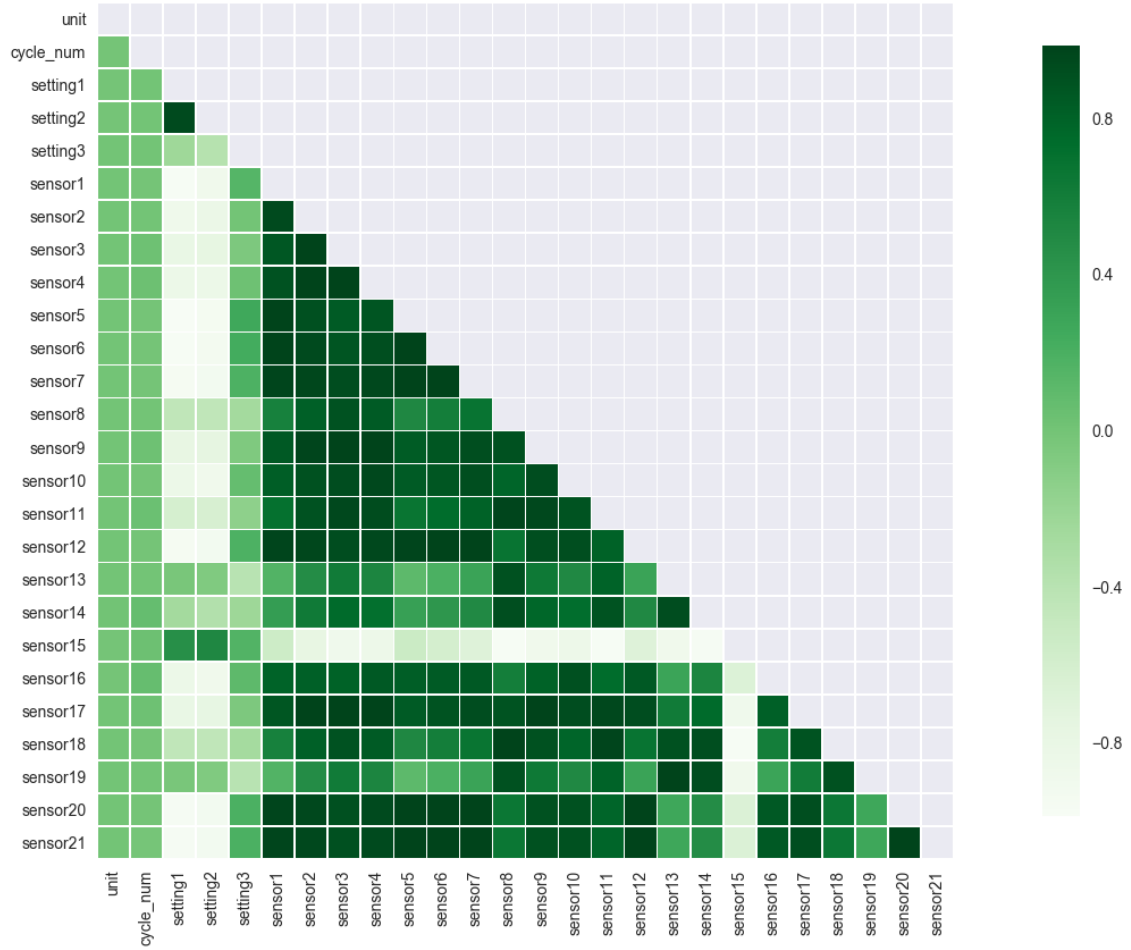

Figura 06: Correlação entre dados

A visão que a figura 6 traz é uma alta correlação entre os dados dos sensores, exceto para o sensor 15 que apresenta alguma correlação somente com os parâmetros de operação. Essa visão é importante para o processamento dos dados, como por exemplo, uma identificação de sensores que podem ou não ser ignorados no modelo.

A partir dessas visualizações e algumas análises dos dados, na etapa de pré processamento dos dados será possível a manipulação de algumas *features* e ter uma redução de variáveis ou mesmo um agrupamento de dados, como os parâmetros de operações bem como sensores.

Algoritmos e técnicas

Algumas técnicas e algoritmos foram utilizados tanto no processamento dos dados quanto na criação do modelo de predição. A fonte para todos os algoritmos e técnicas, exceto quando citado explicitamente, é do artigo publicado em 2011 fonte da biblioteca Scikit-learn (PEDREGOSA, F. AND VAROQUAUX, G. AND GRAMFORT, A. AND MICHEL *et al.*, 2011).

No processamento dos dados foi utilizado um classificador para extrair informação da importância de um determinado *feature* para o modelo. Neste sentido, foi utilizado o *ExtraTreeClassifier* que atua como árvore de decisão na qual dois vetores são passados no algoritmo (X=*features*, Y=*Target*). Em síntese, um subconjunto aleatório de *features* é usado, mas em vez de procurar os limites mais discriminativos, os limites são desenhados aleatoriamente para cada *feature* e o melhor desses limites, gerados aleatoriamente, é escolhido como a regra de divisão/classificação. Isso geralmente permite reduzir um pouco mais a variação do modelo, em detrimento de um aumento ligeiramente maior de viés.

Parâmetros principais do *ExtraTreeClassifier:*

- *n_estimators* – é o número de árvores na floresta de decisão.
- *max_features* – significa o número de características a serem consideradas.
- *max_depth* – profundidade da árvore de decisão.

Para a avaliação do classificador foi utilizado:

- *feature_importances_*: retorna uma lista com as características mais importantes do conjunto avaliado.

Ao entender como o resultado deste classificador, um agrupamento de *features* foi pensado para otimizarmos os resultados do modelo. Com isso, uma técnica de *clustering* chamada *Kmeans* foi empregado para agrupar os dados de operação, conforme melhor explicado na etapa de processamento dos dados. A técnica de *clustering* ou agrupamento que permite uma compreensão dados estudados no sentido de como aqueles valores estão distribuídos. Ao indicar um grupo de dados de treino é retornado um vetor com o número de diferentes grupos para aqueles dados informados. Em suma, ele procura pela menor distância entre os pontos centrais de cada grupo e o ponto estudado até atingir um limiar adequado.

Parâmetros principais do *Kmeans*:

- *n_clusters* – número de grupos a se criar ou número de centróides.
- *n_init* – número de vezes que o algoritmo irá avaliar em diferentes centróides. O resultado é o melhor entre as vezes iniciadas.
- *max_iter* – máximo número de iterações em uma avaliação do algoritmo.

Para avaliar o agrupamento de dados proposto pleo *Kmeans*, foi verificado:

- *silhouette_score* – esta métrica avalia a distância média entre os pontos de um mesmo grupo e a distância média entre os pontos de outros grupos que aquele ponto não faz parte em relação ao centróide.

Na etapa implementar os modelos de *machine learning* alguns algoritmos foram escolhidos como *benchmarking* para se ter uma base de como o modelo comporta aos dados. Foram utilizados:

- → *LinearRegression;*
- → *KneighborsRegressor;*
- → *SupportVectorRegression;*
- → *ExtremeGradientBoosting.*

LinearRegression é um modelo de regressão linear simples na qual um valor alvo é observado com a combinação linear das variáveis de entrada. Esse modelo ajusta um modelo linear com coeficientes para minimizar a soma residual de quadrados entre as respostas observadas no conjunto de dados e as respostas preditas pela aproximação linear.

KneighborsRegressor tem como objetivo encontrar um número pré-definido de amostras de treinamento mais próximas da distância do novo ponto e prever o rótulo a partir delas. O número de amostras pode ser uma constante definida pelo usuário (*k-neighbor neighbor learning*) ou variar com base na densidade local de pontos (aprendizado de vizinho baseado em raio). A distância pode, em geral, ser qualquer medida métrica: a distância euclidiana padrão é a escolha mais comum. Os métodos baseados em *Kneighbors* são conhecidos como métodos de aprendizado de máquina não generalizados, já que eles simplesmente “lembram” todos os seus dados de treinamento.

SupportVector é um método que procura encontrar um hiperplano de separação entre os dados e suas classes. Esse hiperplano busca maximizar a distância entre os pontos mais próximos em relação a cada uma das classes. Para implementar uma regressão, essa técnica possui algumas diferenças básicas de uma classificação, por exemplo, a saída é um número real mas a ideia de identificar o hiperplano e minimar o erro associado.

ExtremeGradientBoosting é uma técnica de aprendizado de máquina para problemas de regressão e classificação, que produz um modelo de previsão na forma de um conjunto de modelos de previsão fracos, geralmente árvores de decisão. Ele constrói o modelo em um modo de estágio, como outros métodos de otimização, e os generaliza ao permitir a otimização de uma função de perda arbitrariamente diferenciável. Para o trabalho em questão, foi utilizado a biblioteca *XGBoost* (“XGBoost Documents”, [S.d.]).

Modelos de Referência

Para a criação de modelos de referência ou *benchmark* foram utilizadas duas fontes. A primeira fonte é o próprio resultado do desafio *PHM08 Prognostics Data Challenge Dataset*. Os três primeiros colocados tiveram pontuações abaixo dos mil pontos de acordo com a métrica específica da competição. Sendo assim, modelos próximos a mil pontos representam resultados incríveis comparados aos vencedores do desafio.

Para referência ao projeto a ser desenvolvido, o trabalho descrito por RAMASSO; SAXENA, 2014) traz alguns pontos para análise dos dados e então entendermos melhor o que cada coluna representa em um modelo de predição de falhas em turbinas.

Tendo em vista o caráter do projeto, uma segunda fonte de referência é o próprio modelo escolhido através de métricas padrões de regressão e que, através de uma exploração melhor dos dados e aperfeiçoamento do modelo, foi possível refinar o resultado.

Sendo assim, o objetivo principal deste projeto é obter um modelo que após o pré-processamento dos dados seja melhorado significativamente tanto nas métricas convencionais de *machine learning* quanto na métrica proposta pela competição *PHM08 Prognostics Data Challenge Dataset.*

Metodologia

Pré-Processamento dos Dados

No pré-processamento dos dados o trabalho junto aos parâmetros de operação foi realizado em primeiro lugar. Ao analisar a figura 6 pode-se compreender uma correlação considerável entre os parâmetros 1 e 2 de operação (*setting1* e *setting2*). Com a intenção de analisar especificamente esses parâmetros, a figura 7 esclarece alguns pontos importantes.

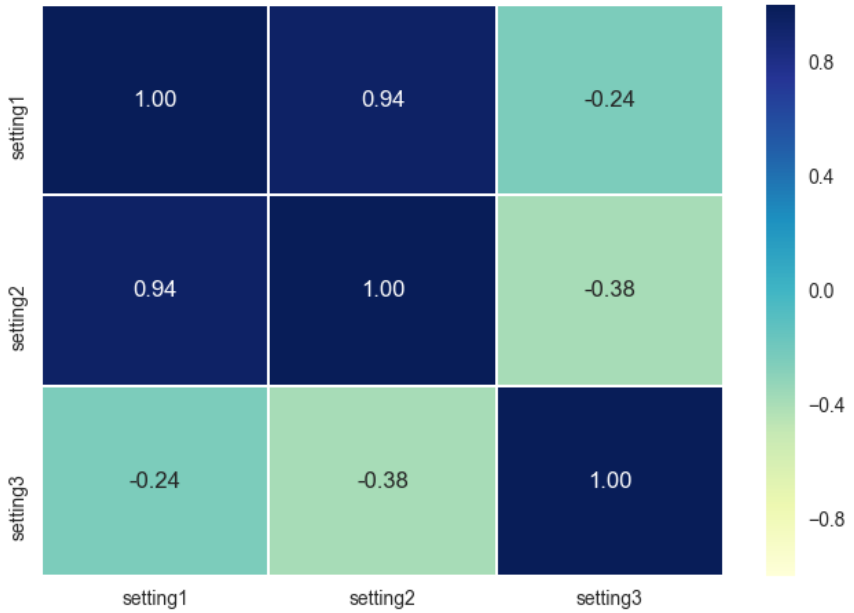

Figura 07: Correlação entre os parâmetros de operação

Pela figura 7 fica evidenciado a alta correlação entre os dois parâmetros citados acima. A partir desta análise, fora criado um atributo em substituição aos dois parâmetros *setting1* e *setting2* através da média entre esses dois valores.

Ainda sobre os parâmetros de operação, fora realizado um estudo sobre a importância desses parâmetros sobre os ciclos de cada turbina. A figura 8 mostra a importância de cada *setting* através de uma análise utilizando *ExtraTreeClassifier* para identificar e quantificar a imporância de cada *setting*.

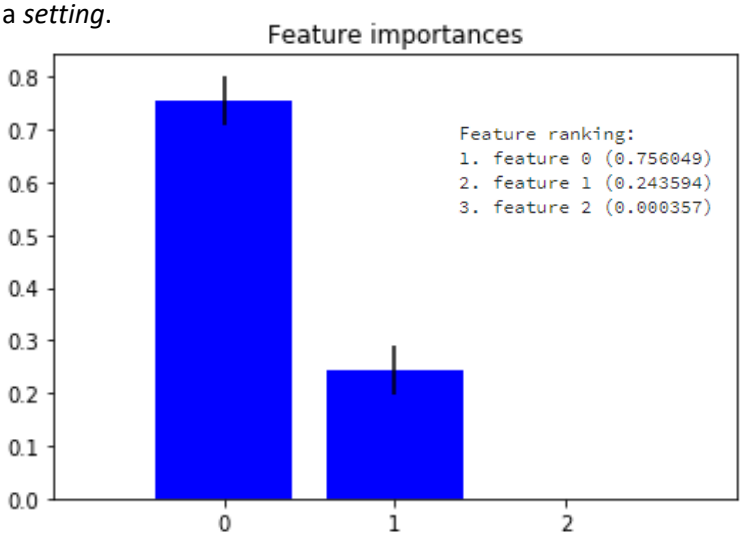

Figura 08: Importância dos três parâmetros de operação (0 = *setting1*, 1 = *setting2* e 3 = *setting3*)

Mesmo a figura 6, através da matriz de correlação, apontar que *setting3* apresenta alguma correlação com alguns sensores, através da análise acima percebe-se que em relação à coluna *target* esse atributo não tem relevância. Com isso, essa coluna foi excluída dos dados, pois não carrega relevância.

Equipamentos mecânicos como motores de combustão interna, turbinas, bombas e ventiladores possuem modos de operação para situações diversas. Ao pensarmos em modos de operação e a degradação ou desgaste gradual que ocorre nesses equipamentos, uma bomba operando em estado estacionário e em valores nominais de operação possui uma degradação, a priori, menor. Já uma mesma bomba operando em uma condição extrema de operação como alta velocidade pode-se entender uma degradação diferente. Seguindo esse raciocínio, uma turbina deve apresentar degradação diferente em modos de operações diferentes, seja em modo estacionário ou em partida.

Sendo assim, os atributos que carregam os parâmetros de operação foram agrupados para se verificar se há de alguma forma grupos de operações pelos dados apresentados. Utilzando uma técnica de *clustering* foram percebidos 6 grupos ou modos de operação das turbinas em estudo. Esse agrupamento foi realizado utilizando o algoritmo *Kmeans* e pode ser observado na figura 9a com os *clusters* bem divididos em função dos três parâmetros de operação. Já na figura 9b tem-se a pontuação *silh_score* para os grupos testados.

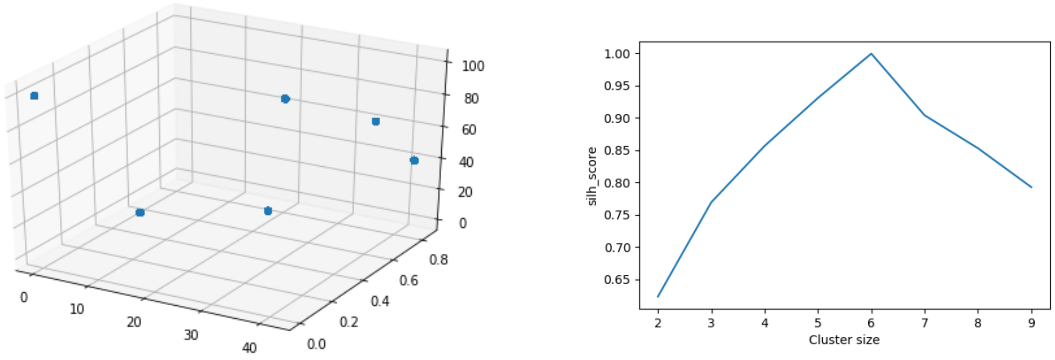

Figura 09: a) *Clusters* em função dos parâmetros de operação. B) Pontuação *silh_score* para os grupos testados

Com esse processemanto sobre os dados fora ainda criado um atributo especificando qual modo de operação, entre 1 e 6, aquela unidade naquele determinado ciclo se encontra.

Feito este trabalho sobre os dados de operação das turbinas o próximo passo é entender e analisar os dados de todos os sensores. Para isso, percebeu-se que os sensores apresentam intervalos de medições diversos e para uma melhor visualização e entendimento de padrões nos dados foi feito a normalização de cada linha do conjunto de dados levando em consideração não somente o dado em si, mas também o modo de operação. A equação 1 abaixo exemplifica a normalização dos dados:

$$N\left(x^{(m,d)}\right) = \frac{x^{(m,d)} - \mu^{(m,d)}}{\sigma^{(m,d)}}$$

A normalização dos dados permitiu extrair algumas informações importantes dos dados. Verificou-se que alguns sensores não possuíam variações dentre os modos de operação. Com isso, a variância deste sensor para o modo de operação é igual a zero. A partir desta informação, esses dados de sensores foram removidos do conjunto devido a variância ser igual a zero.

Por fim, ao normalizar os dados, a visualização dos dados dos sensores em um mesmo gráfico foi possível. Tentou-se agrupar os dados conforme o comportamento de cada sensor. As figuras 10 e 11 mostram esses comportamentos.

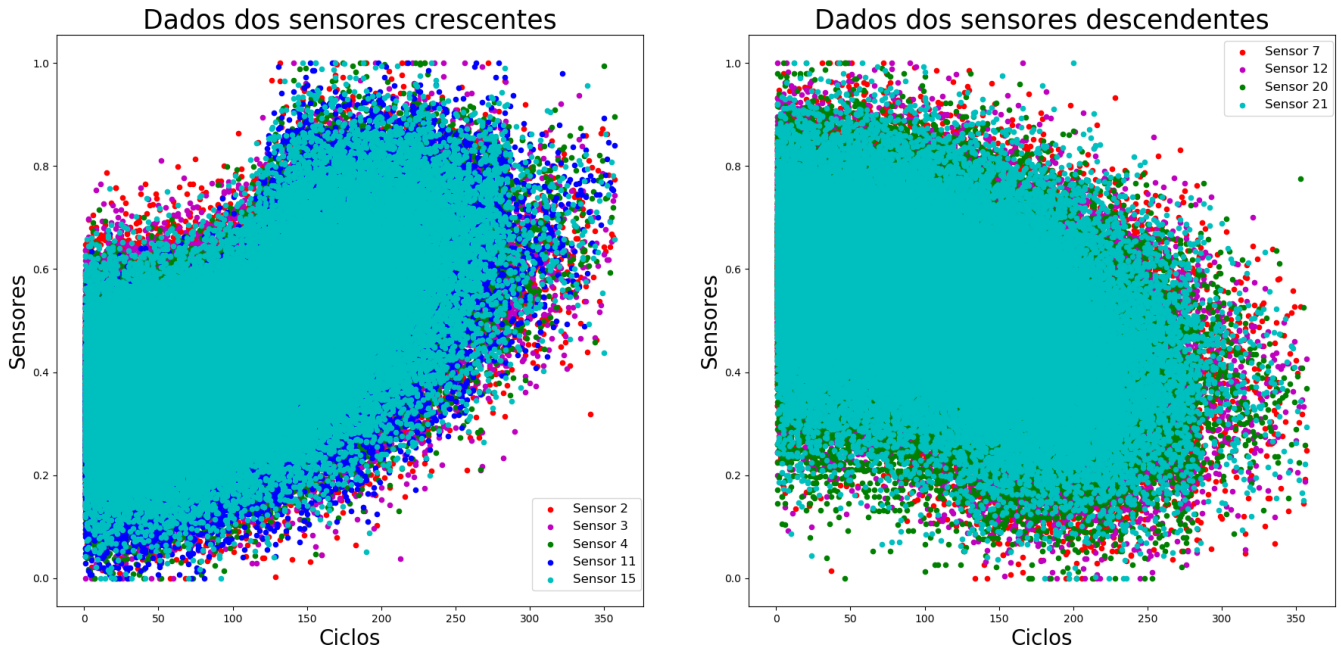

Figura 10: a) Sensores com tendência de crescimento. B) Sensores com tendência decrescente.

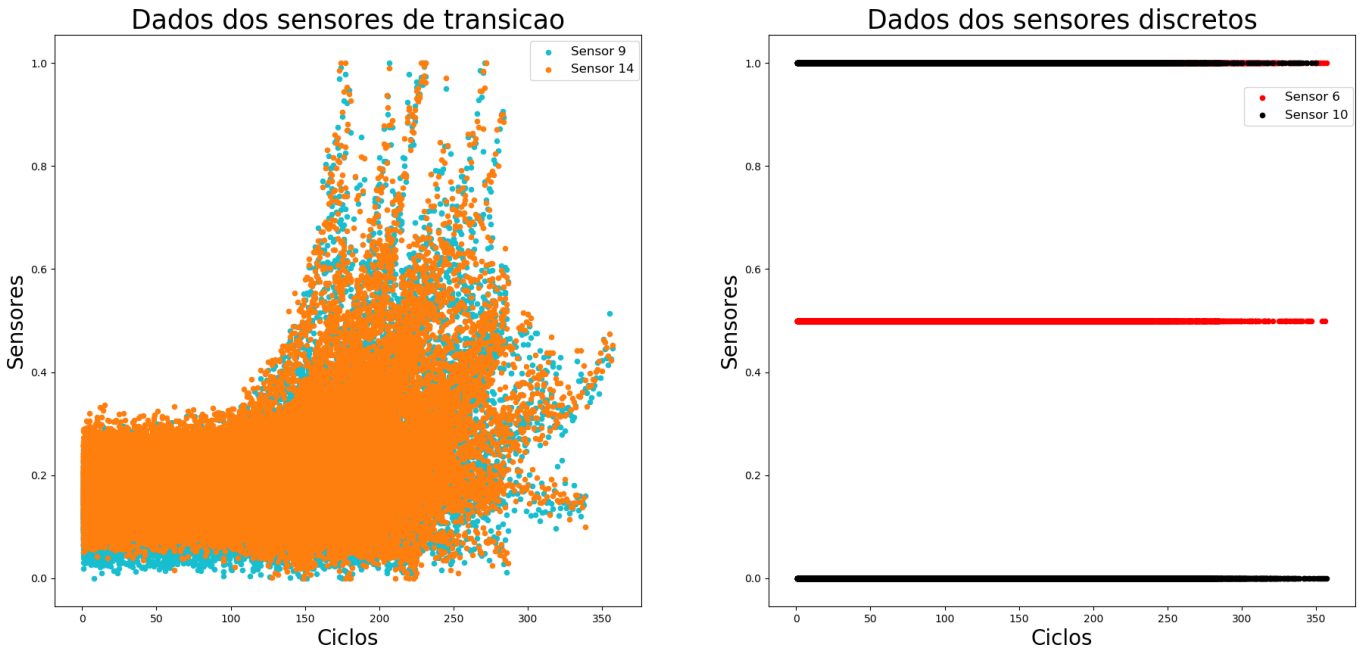

Figura 11: a) Sensores de transição. B) Sensores discretos.

Ao observar o comportamento dos sensores 9 e 14 ao chegar a 120 ciclos aproximadamente percebe-se um aumento dos valores para esses sensores. Com isso, um novo atributo binário foi criado para ciclos abaixo de 120 com valor 0 e para valores de ciclos acima de 120 com valor 1.

Implementação e Avaliação do modelo

A figura abaixo exemplifica a implementação estabelecida para o projeto:

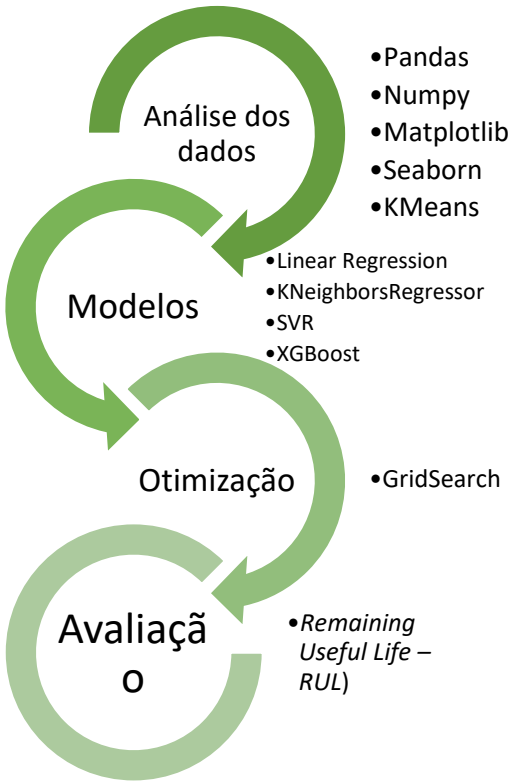

Figura 12: Etapas de implementação do projeto

A etapa de análise de dados já fora detalhada no capítulo anterior de pré-processamento dos dados.

Para os modelos aplicados no estudo em questão decidiu-se, conforme citado no capítulo de Algoritmo e técnicas, os seguintes modelos de regressão para realizar a predição da coluna alvo:

- → *LinearRegression;*
- → *KneighborsRegressor;*
- → *SupportVectorRegression;*
- → *ExtremeGradientBoosting.*

Para avaliar os algoritmos de regressão acima, foi utilizado as métricas *Explained Variance*, *Mean Square Error* e *R2 Score*. Essas pontuações foram novamente extraídas do artigo publicado em 2011 fonte da biblioteca Scikit-learn (PEDREGOSA, F. AND VAROQUAUX, G. AND GRAMFORT, A. AND MICHEL *et al.*, 2011).

Foi realizado um *loop* simples para que cada modelo fosse testado dividindo os dados de treino através do algoritmo de *train test split* no qual 70% dos dados de treino foi efetivamente utilizado no treino do modelo e o restante como conjunto de dados para a validação deste modelo. O resultado para o *dataset* sem nenhum pre processamento está evidenciado na tabela abaixo:

Modelo utilizado	Pontuação R2 (treinamento)	Explained Variance Score (treinamento)	Mean Squared Error (treinamento)	Pontuação R2 (validação)	Explained Variance Score (validação)	Mean Squared Error (validação)
DecisionTreeRegressor	1.000	1.000	0.000	0.302	0.302	3210.105
SVR	0.052	0.057	4469.888	0.031	0.034	4455.077
KNeighborsRegressor	0.731	0.731	1269.463	0.577	0.578	1942.402
LinearRegression	0.577	0.577	1996.470	0.570	0.570	1975.900
XGBRegressor	0.649	0.649	1655.760	0.620	0.620	1746.215

Tabela 01 – Resultado para os modelos sem nenhum pré-processamento nos dados.

Através dos dados apresentados na tabela 1, percebe-se um *overfitting* grande do modelo *Decision Tree* devido aos valores perfeitos nos dados de treino e resultado bem abaixo nos dados de teste. Pode-se tentar entender esse resultado como o modelo em questão não apresentou um aprendizado completo sobre os dados resultado no sobreajuste.

Já para *Support Vector Regression* os resultados foram extremamente baixos para as pontuações de *R2 Score* e *Explained Variance*, e erro associado muito alto definido pelo *mean squared error*. Esse fato pode ser entendido pelo fato de termos muitos ruídos nos dados dos sensores o que dificulta esse modelo em aprender com os dados.

Linear Regression usualmente é um modelo simples mas que apresenta resultados contundentes. Nos dados deste projeto este modelo apresentou pontuação mediana mas bem razoáveis ao compararmos as pontuações de treino e teste.

Por fim, *K Neighbors* e *XGBoost* apresentaram boas pontuações e baixos erros e não indicaram sobreajuste nos dados. Com isso, deverão ser melhor compreendidos após o trabalho de análise de dados.

Tendo a tabela 1 como modelo padrão a melhorar toda a etapa de pré-processamento de dados foi realizada como análise dos dados das turbinas. Em seguida, os mesmos modelos foram testados nos dados trabalhados para se ter uma visão melhor sobre a melhora que essas análises e processamento efetuaram nos dados. A tabela 2 mostra os resultados para esta etapa:

Modelo utilizado	Pontuação R2 (treinamento)	Explained Variance Score (treinamento)	Mean Squared Error (treinamento)	Pontuação R2 (validação)	Explained Variance Score (validação)	Mean Squared Error (validação)
DecisionTreeRegressor	1.000	1.000	0.000	0.683	0.683	0.011
SVR	0.853	0.853	0.005	0.834	0.834	0.006
KNeighborsRegressor	0.908	0.908	0.003	0.851	0.851	0.005
LinearRegression	0.787	0.787	0.008	0.788	0.788	0.008
XGBRegressor	0.832	0.832	0.006	0.824	0.824	0.006

Tabela 02 – Resultado para os modelos após o pré-processamento nos dados.

Analisando a tabela 2 e seus resultados percebemos o mesmo comportamento para o modelo *Decision Tree* mas uma melhora considerável para o modelo *Support Vector Machine* mas ainda abaixo dos demais. Observa-se ainda que os melhores modelos estão entre *k Neighbors* e *XGBoost*.

Após esta etapa de avaliação dos modelos após o pré-processamento dos dados, os modelos que apresentaram melhores pontuações foram o *XGBoost* e *KNeighborsRegressor* em comparação com o restante dos modelos.

Refinamento

O primeiro ponto da etapa de refinamento foi escolher o modelo final. Apesar dos dois modelos, *XGBoost* e *KNeighborsRegressor*, apresentarem ótimos resultados nas métricas estatísticas convencionais, uma avaliação na métrica da competição é necessária para se tomar uma decisão assertiva.

Sendo assim, aplicando a métrica de avaliação da competição encontra-se os seguintes valores:

Modelo utilizado / Dataset	Treino	Teste
KNeighborsRegressor	790,778.35	26,545,793,045,829.88
XGBRegressor	1,047,353.54	3,30,475,759.05

Tabela 03 – Resultado para os modelos escolhidos utilizando a métrica da competição

Tendo em vista o alto sobre ajuste do modelo *KNeighborsRegressor*, o modelo escolhido foi o *XGBRegressor* devido aos bons resultados nas métricas padrão e um melhor resultado na métrica da competição.

O refinamento do modelo escolhido foi realizado através de uma técnica chamada *Grid Search*. Através desta técnica é possível testar parâmetros e através de uma métrica estabelecida encontrar a melhor combinação destes parâmetros.

Para o modelo escolhido foi testado os seguintes parâmetros:

- → n_esteimators = 100,150 e 200;
- → leaning_rate = 0.05, 0.07 e 0.4;
- → colsample_bytree = 0.5, 0.7 e 1;
- → max_depth = 5, 7 e 10;
- → subsample = 0.6, 0.7 e 0.8;
- → gamma = 0, 0.1 e 0.3

Para avaliar estes parâmetros o resultado da métrica *R2 score* foi definida como padrão para definir qual valor desses parâmetros acima serão considerados. A partir desta análise, os resultados foram:

- → n_esteimators = 200;
- → leaning_rate = 0.07;
- → colsample_bytree = 0.7;
- → max_depth = 10;
- → subsample = 0.8;
- → gamma = 0

Com esses parâmetros o resultado do modelo está apresentado na tabela 4 para as métricas padrão e tabela 5 para a métrica da competição:

Modelo utilizado	Pontuação R2 (treinamento)	Explained Variance Score (treinamento)	Mean Squared Error (treinamento)	Pontuação R2 (validação)	Explained Variance Score (validação)	Mean Squared Error (validação)
XGBRegressor	0.976	0.976	0.001	0.901	0.901	0.004

Tabela 04 – Resultado para os modelos após o pré-processamento nos dados.

Modelo utilizado / Dataset	Treino	Teste
XGBRegressor	135,297.47	598,144,934.58

Tabela 05 – Resultado para o modelo escolhido utilizando a métrica da competição

Observando os resultados apresentados nas tabelas acima percebe-se uma melhora considerável para o modelo após o *GridSearch* ter sido aplicado otimizando os parâmetros do modelo.

Modelo utilizado / Dataset	Treino	Teste
XGBRegressor	774.11%	180.99%

Tabela 06 – Percentual de melhora do modelo

Mesmo sendo notória a melhora nos modelos, pode-se perceber um possível sobre ajuste ou *overfitting* dos dados devido à alta discrepância entre o resultado para os dados de teste comparado aos dados de treino. Sendo assim, para o modelo *XGBoost* foi utilizado um parâmetro chamado *Early Stop* que, através de uma métrica pré-estabelecida, ele avalia em qual

etapa da árvore de decisão essa métrica não apresenta uma melhora. Essa técnica é utilizada para diminuir ou prevenir o sobre ajuste. Para esta técnica, a métrica escolhida foi o *Root Mean Squared Error (rmse).*

Ao implementar essa técnica obtém-se o gráfico da figura 13 abaixo:

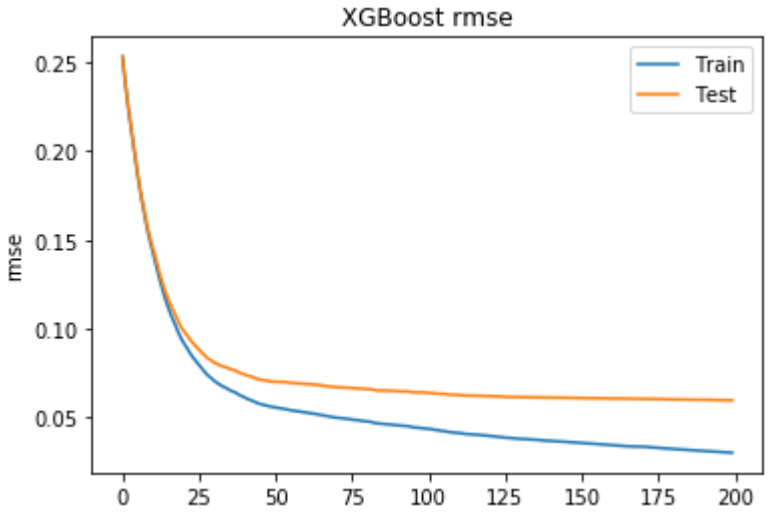

Figura 13: Erro associado ao sobre ajuste do modelo

Analisando a figura 13, percebe-se que não é possível afirmar que houve sobre ajuste dos dados, mas a diferença demonstrada na tabela 5 deve ser entendida como um ponto de melhoria para futuros trabalhos.

Resultados

Avaliação e Validação do Modelo

O modelo escolhido foi o *XGBoost* e os parâmetros detalhados na etapa de Implementação. Pode-se ainda acrescentar ao modelo escolhido:

- Através das tabelas 1 e 2 é possível entender a importância do processamento dos dados neste estudo;
- Conforme discutido, o modelo de *KneighborsRegression* apresentou ótimos resultados *standard,* mas não quando utilizado a métrica da competição. Por este motivo o *XGBoost* foi o escolhido para o problema em questão;
- O refinamento do modelo final melhora sensivelmente o resultado do trabalho, sendo ponto essencial para o resultado;
- Mesmo com ótimos resultados nas métricas padrões, a métrica da competição se mostrou ainda abaixo dos vencedores da competição;
- Investigando um possível sobre ajuste do modelo não foi possível afirmar este sobre ajuste de acordo com a figura 13;
- O modelo teve uma validação cruzada com 70% dos dados para treino e o restante para validação do modelo.

Para verificar o quão robusto o modelo é pode-se entender que:

- Após obter-se os ciclos previstos é criada uma coluna D que é a diferença entre o previsto e o real. Essa variável apresenta valores consistentes de média igual a -0.203. Essa média próxima a zero evidencia um modelo ajustado, porém como a métrica da competição penaliza mais valores de D maior que zero pode-se compreender os altos valores nos resultados.
- Os valores de resultado contidos na tabela 4 torna evidente bons valores para o modelo.

Justificativa

Comparando com o modelo de referência o modelo final apresenta melhora significativa e pode-se observar nas tabelas 1 e 2, contudo a métrica da competição tem um caráter peculiar de penalizar mais os falsos negativos do que os falsos positivos. Neste caso, significa que valores de ciclos previstos maiores do que o ciclo real possui uma penalização maior do que o contrário.

Tendo em vista essa característica podemos afirmar a melhora do modelo em relação ao *benchmark* do projeto, mas há ainda pontos a se melhorar, pois os vencedores do desafio obtiveram métricas melhores com os dados.

Conclusão

Reflexão

O trabalho foi extremamente desafiador. Por ser um conjunto de dados da NASA sem muitos projetos como referência pode-se dizer que aumentou o grau do desafio. Mas através de algumas técnicas foi possível descobrir *insights* importantes nos dados e concluir o projeto.

O conjunto de dados é interessante para a minha área de atuação para mesclar conhecimentos de engenharia à ciência de dados e *Machine Learning* o que abre um campo de aplicação dessas técnicas.

A biblioteca *XGBoost* mostrou-se muito poderosa além do desafio em encontrar os parâmetros corretos para o problema. Uma próxima abordagem poderia ser feita utilizando conceitos de *DeepLearning* para aumentar o crescimento com o projeto.

Aperfeiçoamento

Os seguintes pontos de aperfeiçoamento são propostos:

- Utilização do filtro de Kalman nos dados dos sensores para eliminar ruídos;
- Tentar compreender a função física de cada sensor ou relacioná-la melhor com os parâmetros de operação. Isso pode acarretar *insights* importantes para a análise dos dados;
- Utilização de técnicas e bibliotecas como DeepLearning *e TensorFlow* para expandir o aprendizado e melhorar as respostas.

Bibliografia

BEZERRA SOUTO MAIOR, C. *et al.* Remaining Useful Life Estimation by Empirical Mode Decomposition and Support Vector Machine. *IEEE Latin America Transactions*, v. 14, n. 11, p. 4603–4610, 2016. Disponível em: <http://ieeexplore.ieee.org/document/7795836/>.

ELFORJANI, M. Estimation of Remaining Useful Life of Slow Speed Bearings Using Acoustic Emission Signals. *Journal of Nondestructive Evaluation*, v. 35, n. 4, p. 1–16, 2016.

FORNLÖF, V. *et al.* RUL estimation and maintenance optimization for aircraft engines: a system of system approach. *International Journal of Systems Assurance Engineering and Management*, v. 7, n. 4, p. 450–461, 2016.

HU, Z. *et al.* Incipient Fault Diagnostics and Remaining Useful Life Prediction of Analog Filters. *Journal of Electronic Testing: Theory and Applications (JETTA)*, v. 31, n. 5–6, p. 461–477, 2015. Disponível em: <http://dx.doi.org/10.1007/s10836-015-5543-3>.

PEDREGOSA, F. AND VAROQUAUX, G. AND GRAMFORT, A. AND MICHEL, V. *et al.* Scikit-learn: Machine Learning in {P}ython. *Journal of Machine Learning Research*, v. 12, p. 2825--2830, 2011. Disponível em: <http://scikit-learn.org/stable/about.html#citing-scikit-learn>. Acesso em: 5 jun. 2018.

PROGNOSTICS AND HEALTH MANAGEMENT SOCIETY. *International journal of prognostics and health management.* [S.l.]: Prognostics and Health Management Society, [S.d.]. Disponível em: <http://www.phmsociety.org/>. Acesso em: 9 jun. 2018.

RAMASSO, E.; SAXENA, A. Performance Benchmarking and Analysis of Prognostic Methods for CMAPSS Datasets. *International Journal of Prognostics and Health Management*, n. ISSN2153-2648, p. 1–15, 2014.

XGBoost Documents. Disponível em: <https://xgboost.readthedocs.io/en/latest//>. Acesso em: 5 jun. 2018.