MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Ouro Preto
Escola de Minas
Curso de Graduação em Engenharia de Controle e Automação

VITOR REIS ROSA

ESTUDO DE DESEMPENHO DE ALGORITMOS DE *MACHINE LEARNING* APLICADO À PROGNÓSTICOS E MONITORAMENTO DE CONDIÇÃO

OURO PRETO - MG
DEZEMBRO DE 2019

VITOR REIS ROSA
vitor.reisr@hotmail.com

ESTUDO DE DESEMPENHO DE ALGORITMOS DE *MACHINE LEARNING* APLICADO À PROGNÓSTICOS E MONITORAMENTO DE CONDIÇÃO

Monografia a apresentada ao curso de graduação em Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como requisito para a obtenção do título de Engenheiro de Controle e Automação.

Professor Orientador: Dra. Luciana Gomes Castanheira

OURO PRETO – MG
DEZEMBRO DE 2019

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

R788e Rosa, Vitor Reis .
Estudo de desempenho de algoritmos de machine learning aplicado à prognósticos e monitoramento de condição. [manuscrito] / Vitor Reis Rosa. Vitor Rosa. - 2019.
68 f.: il.: color., gráf., tab.. + Equação.

Orientadora: Profa. Dra. Luciana Castanheira.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Escola de Minas.

1. Machine learning. 2. Remaining useful life (RUL) . 3. Engenharia - Estimativas . 4. Condition monitoring (CM). 5. Aprendizado do computador . I. Rosa, Vitor. II. Castanheira, Luciana. III. Universidade Federal de Ouro Preto. IV. Título.

CDU 681.5

Bibliotecário(a) Responsável: Maristela Sanches Lima Mesquita - CRB:1716

Monografia intitulada Estudo de desempenho de algoritmos de Machine Learning aplicados à prognóstico e monitoramento de condição defendida e aprovada, em 11 de dezembro de 2019, pela comissão avaliadora constituída pelos professores:

Profa. Dra. Luciana Gomes Castanheira– Orientadora

Prof. Dr. José Alberto Naves Cocota Júnior – Professor Convidado

Prof. André Almeida Santos –Professor Convidado

AGRADECIMENTOS

Agradeço primeiramente a minha família, considero-vos incompreensivelmente sensacionais. Particularmente peculiares, estão sempre a tentar me ajudar de maneira tão incessante que eventualmente o fazem de forma até involuntária. Realmente será difícil passar adiante o que vocês fazem por mim.

À Eterna República Pif-Paf, seus moradores e ex-alunos, suas dificuldades e glórias, suas virtudes e injustiças... se tudo funcionasse perfeitamente não seria tão emocionante! Constituí outra família, superei meus medos, me diverti como nunca. Depois do que por aqui passei, sinto-me preparado para qualquer desafio.

Aos meus amigos, presentes e distantes, por constituírem minhas melhores memórias. É de vocês que vem minha motivação.

À UFOP e os grandes mestres, pelo ensino e relações de qualidade. Não é só uma Universidade.

À Israel Electric Co. em especial Möshe e Netzah, por me apresentarem ao mundo dos dados.

À Paderborn Universität, em especial Sr. Kennedy, pela confiança e paciência.

"A crise é a melhor benção que pode acontecer a pessoas e países porque a crise traz progresso, a criatividade nasce da angústia e o dia lindo vem do ventre da tempestade escura. É na crise que surge a invenção, a descoberta, a reflexão e as grandes estratégias do "marketing" do amor. Quem supera a crise, supera a si mesmo, sem ficar superado. E quem pendura no gancho da crise seus fracassos e lamúrias, violenta seu próprio talento e tem mais respeito a problemas que soluções. A crise é uma farsa, a não ser a crise de incompetência, pois o problema de pessoas e países é de auto gerência. Sem crises não há desafios, sem desafios, a vida é rotina que chama o túmulo. Sem crise ninguém tem méritos. É só na crise que você mostra que é bom, pois sem crise toda vento é carícia. Por isso, falar da crise é promovê-la e calar na crise é exaltar o conformismo. Em vez disso, trabalhe duro, desinflacione a crise de você mesmo e acabe de uma vez com a única crise ameaçadora que é a da tragédia de não saber por onde começar. "

Maurício Góis

RESUMO

ROSA, Vitor Reis: **Estudo de desempenho de algoritmos de *machine learning* aplicado à prognósticos e monitoramento de condição,** 2019. (Graduação em Engenharia de Controle e Automação). Escola de Minas, Universidade Federal de Ouro Preto.

Neste trabalho será realizada uma abordagem original para um problema recorrente no campo da manutenção preditiva. O objetivo visa determinar com a maior precisão possível a vida útil remanescente de motores de turbina submetidos a estresse constante até que atinjam a condição de falha. Dados provenientes de simulações de leituras de sensores instalados nos motores são o alvo principal deste estudo e serão devidamente trabalhados através de técnicas avançadas de estatística, análise e tratamento de dados. Os dados foram disponibilizados pelo Centro de Prognósticos e Excelência da NASA Ames e posteriormente utilizados em uma competição de prognósticos e monitoramento de condição denominada *PHM08 Data Challenge*.

Palavras-chave: *Machine learning* para manutenção preditiva, prognósticos, tempo de vida útil remanescente, monitoramento de condição.

ABSTRACT

This work presents an original approach to a recurring problem in predictive maintenance field. The objective is to determine as accurately as possible the remaining useful life of turbofan engines subjected to constant stress until they reach the failure condition. Data from simulations of sensor readings installed on the engines are the main target of this study and will be properly worked through advanced techniques of statistics, analysis and data processing. The data were made available by Center for Prognosis and Excellence of NASA Ames and later used in a condition monitoring and prognostics competition named the PHM08 Data Challenge.

Key-words: *Machine learning for predictive maintenance, PHM08, RUL prediction, condition monitoring.*

LISTA DE FIGURAS

Figura 1 - Exemplo de *clustering* 9
Figura 2 - Exemplo de teste do *training set* 18
Figura 3 - Exemplo de teste do *test set* 19
Figura 4 - Exemplo de estrutura dos dados do *original training set* 22
Figura 5 - Diagrama de blocos das etapas do trabalho 25
Figura 6 - Todos os sensores crus do *original training set* 29
Figura 7 - Todas as condições de operação do *original training set* 30
Figura 8 - Condições de operação visualização 3D 31
Figura 9 - Condições de operação, cores definidas por *cluster* 33
Figura 10 - Todos os sensores crus, cores definidas por cluster 35
Figura 11 - Todos os sensores após a padronização personalizada 38
Figura 12 - Sensores do grupo A 39
Figura 13 - Sensores do grupo B 39
Figura 14 - Sensores do grupo C 39
Figura 15 - Explicação do *violin plot* 40
Figura 16 - *Violin plot* após padronização e primeira seleção de sensores 41
Figura 17 - *Violin plot* após remoção de *outliers* e seleção completa de sensores 42
Figura 18 - Grupo de sensores A após filtragem 43
Figura 19 - Grupo de sensores B após filtragem 43
Figura 20 - Grupo de sensores C após filtragem 43
Figura 21 - Matriz de correlação dos sensores do grupo A 44
Figura 22 - Matriz de correlação dos sensores do grupo B 44
Figura 23 - Matriz de correlação dos sensores do grupo C 45
Figura 24 - Sensor A após redução de dimensionalidade 46
Figura 25 - Sensor B após redução de dimensionalidade 46
Figura 26 - Sensor C após redução de dimensionalidade 47
Figura 27 - Sensor A após transformação logarítmica 48
Figura 28 - Sensor B após transformação logarítmica 49
Figura 29 - Sensor C após transformação logarítmica 49
Figura 30 - Exemplo de criação de testes artificiais a partir de testes originais do *training set* 51

Figura 31 - Exemplo da estrutura final de dados do *artificial training set* antes da etapa de *feature extraction* 53
Figura 32 - Código de extração automática de *features* 54
Figura 33 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *Extreme Gradient Boosting* 56
Figura 34 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *Decision Tree* 57
Figura 35 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *SVM* 58
Figura 36 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *Random Forest* 59
Figura 37 - Python função de avaliação de desempenho 61

LISTA DE EQUAÇÕES

Equação 1 - Equação do algoritmo *K-Means clustering*..... 10
Equação 2 - Equação da pontuação do teste Silhouette Analysis..... 10
Equação 3 – Matriz de transformação PCA..... 12
Equação 4 - Transformação PCA 12
Equação 5 – Z Score..... 15
Equação 6 - Métrica oficial da competição no qual os dados pertencem 21
Equação 7 – Padronização 35

LISTA DE TABELAS

Tabela 1 - 20 melhores pontuações *PHM08 Data Challenge* 20
Tabela 2 - Exemplo explicativo da não linearidade da equação de avaliação 21
Tabela 3 - 20 melhores pontuações PHM08 Data Challenge + pontuação final do trabalho .. 63

SUMÁRIO

1 INTRODUÇÃO 1
1.1 Contextualização 1
1.2 Importância do Trabalho 2
1.3 Objetivos 3
1.3.1 Geral 3
1.3.2 Específico 3
1.4 Estrutura do Trabalho 4
2 REFERENCIAL TEÓRICO 5
2.1 *Data Science* 5
2.1.1 *Data Science* Aplicado à Manutenção 6
2.2 *Machine Learning* 7
2.3 *Supervised Learning* 7
2.4 *Unsupervised Learning* 8
2.4.1 *Clustering* 8
2.4.2 *Silhouette Analysis* 10
2.4.3 Redução de Dimensionalidade 11
2.5 Padronização 13
2.6 *Outliers* 14
2.7 *Standard Score* 14
2.8 Transformações 15
2.9 *Feature Engineering* 16
3 APROFUNDAMENTO DE CASO 17
3.1 Origem dos Dados 17
3.2 Desmembrando o Problema 17
3.3 Estrutura dos Dados 22
4 METODOLOGIA 23

4.1 Estratégia 24
4.2 *Original Training Set* 27
4.2.1 Visualização de Dados 27
4.2.2 *Clustering* 31
4.2.3 Padronização 35
4.2.4 Filtragem 38
4.2.5 PCA (*Principal Component Analysis*) 44
4.2.6 Transformação Logarítmica 47
4.3 *Artificial Training Set* 50
4.3.1 *Feature Extraction and Selection* 52
4.3.2 Treinamento dos Modelos Preditivos 54
5. RESULTADOS E DISCUSSÕES 60
5.1 Pontuação Final 62
6 CONSIDERAÇÕES FINAIS 63

1 INTRODUÇÃO

1.1 Contextualização

A evolução não acontece de forma planejada, é um processo dinâmico e constante. A humanidade passa por incessantes transformações sociais, políticas, econômicas, tecnológicas, etc. É da natureza do ser humano, e fortemente catalisada pelos modelos capitalistas da sociedade atual, que constantemente busquem por novos desafios, e esta "demanda" traz à tona algumas das características mais nobres da espécie: a capacidade de se adaptar, mudar, melhorar, criar soluções criativas e inéditas, sair da zona de conforto.

A tecnologia da informação, incorporada no meio pessoal e comercial, a princípio pela terceira revolução industrial, fez com que hoje a maioria das pessoas estejam conectados através dos mais variados dispositivos eletrônicos. Entretanto o pico de recursos tecnológicos relacionadas a esta área tem sido tão expressivo nos últimos anos que inovações são criadas e se tornam obsoletas em intervalos de tempo cada vez menores. O futuro tem chegado cada vez mais rápido e como sempre, trazendo consigo novas possibilidades, desafios e problemas. É inegável que chegamos à quarta revolução industrial, denominada por muitos como a era dos dados, e com isso expressões recentes como "ciência de dados", "*big data*", "*machine learning*" ganham cada vez mais expressividade no cenário econômico global.

A ciência de dados, por definição, é um campo multidisciplinar que utiliza métodos estatísticos, processos, algoritmos e sistemas científicos para extrair padrões e informações relevantes de dados estruturados e não estruturados. É razoável de entender o porquê de esta área ter sido afetada tão positivamente com a evolução tecnológica recente. Problemas recorrentes como a limitação de *hardware* para armazenamento de dados e processamento de informações tem sido constantemente superado, e isto afeta diretamente o desempenho dos algoritmos baseados inteligência artificial.

Neste trabalho, a ciência e o tratamento de dados serão utilizados para absorver informações relevantes da distribuição de dados fornecidos por sensores instalados em motores de turbina e comparar a eficiência de alguns dos algoritmos mais populares da atualidade em um caso nada incomum, mas específico da manutenção preditiva.

1.2 Importância do Trabalho

Com crescente desenvolvimento tecnológico somado às políticas capitalistas das maiores industrias globais, a competitividade entre estas se tornou inevitavelmente uma corrida pelos meios de produção mais eficientes. Esta eficiência é um custo-benefício não linear que envolve duas variáveis fundamentais: a quantidade de recursos utilizados e o tempo da produção. Este cenário tem contribuído de forma significativa para o aumento da relevância da manutenção na cadeia produtiva, tornando cada vez mais atrativo a implementação de métodos que visam otimizar estrategicamente sua execução.

A ABNT 5452-1994 *apud* Xenos (1998, p.18) denomina manutenção como: “combinação de ações técnicas e administrativas, incluindo as de supervisão, destinadas a manter ou recolocar um item no estado no qual possa desempenhar uma função requerida”. Já a NBR 5462 de novembro de 1994 vai além e trata a respeito dos principais conceitos e terminologias que envolvem a confiabilidade da manutenção, e a classifica em três tipos:

- Manutenção corretiva: “Manutenção efetuada após a ocorrência de uma pane destinada a recolocar um item em condições de executar uma função requerida”. Dentre todos os três tipos esta é a forma mais óbvia e mais primária de manutenção, pode-se resumi-la pelo ciclo "quebra-repara", ou seja, o reparo dos equipamentos acontece após a condição de falha. Geralmente constitui a forma mais cara de manutenção quando encarada do ponto de vista total do sistema.
- Manutenção preventiva: “Manutenção efetuada em intervalos predeterminados, ou de acordo com critérios prescritos, destinada a reduzir a probabilidade de falha ou a degradação do funcionamento de um item”. Neste caso as ações de manutenção realizadas de forma prévia, com a finalidade de evitar que os equipamentos falhem durante o processo de produção. Todas as atividades destinadas a prevenção de falhas, panes, quebras, etc., são classificadas como manutenção preventiva.
- Manutenção preditiva: “Manutenção que permite garantir uma qualidade de serviço desejada, com base na aplicação sistemática de técnicas de análise, utilizando-se de meios de supervisão centralizados ou de amostragem, para reduzir ao mínimo a manutenção preventiva e diminuir a manutenção corretiva”.

A partir do que foi apresentado, destaca-se a manutenção preditiva por ser o foco deste estudo, pois possui a capacidade de demonstrar as condições reais de funcionamento do equipamento em função das condições de operação e de vida útil.

Este tipo de manutenção tem o objetivo de predizer a situação do equipamento e encontrar falhas em estágio inicial, quando ainda não são prejudiciais ao equipamento e/ou processo de produção. Uma vez que a falha foi identificada em estágio inicial, pode-se planejar e programar ações para eliminar essa falha.

Dessa forma, os custos e o tempo desprendidos através da manutenção preditiva são infinitamente menores do que quando tratamos o equipamento com manutenção corretiva ou preventiva. O investimento é maior, tendo em vista que demanda equipamentos de análise mais sofisticados e profissionais especialistas em interpretar e lidar com os indicadores.

Para Moro (2007), a manutenção preditiva tem inúmeros objetivos diferentes das manutenções corretiva e preventiva, que tem em vista reduzirem os custos envolvidos e aumentar a produtividade, tais como:

- Prever com esmero a necessidade de intervenção no equipamento;
- Eliminar o tempo gasto em desmontagens desnecessárias;
- Lograr da máxima vida útil dos equipamentos;
- Mitigar progressão de danos;
- Aumentar o grau de confiança no desempenho do equipamento.

1.3 Objetivos

1.3.1 Geral

Realizar um estudo baseado na implementação de diferentes algoritmos de *machine learning* com o objetivo de avaliar seus respectivos desempenhos em um caso específico da manutenção preditiva através de uma análise comparativa.

1.3.2 Específico

- Realizar um estudo teórico sobre recursos específicos baseados em modelos estatísticos e algoritmos de *machine learning*, bem como as razões que justifiquem suas implementações.
- Desenvolvimento de uma intensa fase de visualização e pré-processamento de dados.
- Prever com a maior precisão possível o tempo de vida útil remanescente dos motores.

- Análise comparativa dos resultados da implementação de algoritmos baseados em regressão após o devido tratamento dos dados.

1.4 Estrutura do Trabalho

O estudo de será dividido em cinco capítulos e será redigido de acordo com as normas vigentes da ABNT e do curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto, apresentando-se de acordo com a estrutura a seguir.

- Capítulo 1: Contextualização e justificativa do caso a ser estudado seguido pelos objetivos geral e específico.
- Capítulo 2: Fundamentação teórica de todos os procedimentos a serem realizados neste trabalho bem como o embasamento estatístico que justifique as circunstâncias nas quais determinado recurso é utilizado e suas vantagens para o caso.
- Capítulo 3: Análise detalhada do problema e da estrutura dos dados a serem trabalhados.
- Capítulo 4: Apresentar-se-á neste, todo o procedimento metodológico adotado no estudo de forma cronológica.
- Capítulo 5: Resultados e discussões.
- Capítulo 6: Conclusões e recomendações.

2 REFERENCIAL TEÓRICO

2.1 *Data Science*

Segundo Hyashi (1998), a definição de *Data Science* é: "um conceito para unificar estatísticas, análise de dados, *machine learning* e seus métodos relacionados, a fim de entender e analisar fenômenos reais envolvendo dados". Entretanto uma definição mais atualizada e esclarecedora é comumente encontrada na internet: "*data science* é um campo multidisciplinar que utiliza métodos, processos, algoritmos e sistemas científicos para extrair conhecimento e padrões de dados estruturados e não estruturados.

Independente da definição, o termo *data science* é relativamente novo e surgiu para designar especificamente uma nova profissão que deve dar sentido aos vastos volumes de dados que são armazenados a cada segundo. O ramo da tecnologia relacionado a informática e estatística jamais fora tão abençoado por um pico tecnológico como é o caso para a revolução dos dados na qual mergulha-se cada vez mais a sociedade do século XXI.

Nos dias de hoje é seguro afirmar que a ciência de dados contribui ou poderia contribuir significativamente para a maioria dos setores do mercado de trabalho, se utilizado de maneira apropriada. Lidar com dados pode ser em primeira impressão uma tarefa relativamente simples, cujo não demanda extremo conhecimento cognitivo para analisar uma porção de dados e deduzir resultados lógicos. É razoavelmente simples analisar uma planilha com 365 valores que representam as médias diárias de temperaturas durante um ano e concluir que o ano é divido em quatro estações. Entretanto talvez não seja tão simples analisar uma planilha com dados que representam medições por minuto dos últimos dez anos de fornecimento energético de um país e deduzir a demanda energética da semana seguinte.

Utilizar grandes volume de dados juntamente com os expressivos e incansáveis recursos de processamentos disponíveis atualmente é uma ferramenta poderosa e seus resultados podem trazer incalculáveis otimizações. Não é de se espantar que ramos econômicos tão diferentes entre si aderem cada vez mais aos benefícios que o tratamento de dados pode trazer. Alguns exemplos de aplicação: detecção de fraudes, análise riscos, prognósticos na área da saúde, pesquisas na internet, publicidade segmentada, reconhecimento de imagens, reconhecimento

de fala, planejamento de rotas aéreas, realidade aumentada e pôr fim a aplicação alvo deste trabalho: a manutenção.

2.1.1 *Data Science* Aplicado à Manutenção

Quando o impacto de uma falha não pode ser proporcionado (quando envolve fatalidades), ou quando seu prejuízo se refere a valores exorbitantes, é imprescindível que o equipamento em questão seja submetido a manutenção preventiva, que envolve inspeção e reparo periódicos, geralmente programados com base no tempo de serviço. O desafio do planejamento adequado cresce de forma diretamente proporcional a complexidade do sistema no qual a máquina está inserida. Em um sistema constituído por diversas máquinas e componentes operando simultaneamente, sob condições e configurações específicas, é comum que todos estes fatores estejam interligados e de alguma forma tenham sua parcela de contribuição para a degradação da condição de saúde de determinados equipamentos. É extremamente complicado de entender o momento em que a manutenção deve ser executada para que os componentes não sejam substituídos prematuramente quando o sistema como um todo é complexo o suficiente, e é neste caso que a manutenção preventiva “evolui” para manutenção preditiva.

Neste modelo de manutenção, dados são adquiridos em tempo real de equipamentos. Estes dados serão processados e inseridos em modelos matemáticos capazes de transforma-los em informações relevantes para a definição do estado de saúde do equipamento, e consequentemente a otimização de seu tempo de operação.

2.2 *Machine Learning*

Machine learning consiste no estudo científico de algoritmos e modelos estatísticos que sistemas computacionais utilizam para executar tarefas específicas sem usar instruções explícitas pré-definidas pelo usuário, baseando-se em padrões e inferência. Pode ser interpretado como subconjunto de inteligência artificial e uma ferramenta robusta para o campo da análise e processamento de dados. Os algoritmos de *machine learning* constroem modelos matemáticos baseado em dados de amostragem, comumente conhecidos como "*training set*", para fazer previsões ou tomar decisões sem instruções explicitas definidas pelo programador para executar determinada tarefa. Os algoritmos de machine learning realizam seus processos de "aprendizagem" baseados em três métodos distintas, cujo são classificados como:

- *Supervised learning*
- *Unsupervised learning*
- *Reinforcement Learning*

Neste trabalho serão realizados procedimentos envolvendo algoritmos de *supervised* e *unsupervised learning*, uma explicação simplificada de seus princípios de funcionamento é disposta nos tópicos abaixo.

2.3 *Supervised Learning*

Supervised learning é um subconjunto de *machine learning* e pode ser definido como uma função que mapeia a relação entre uma ou mais entradas para uma determinada saída. O objetivo é conseguir, através da inserção do *training set*, mapear essa relação de forma mais próxima possível da realidade, possibilitando a realização mais eficaz de previsões quando submetidos a entradas desconhecidas.

Está técnica é nomeada de tal forma pois o processo de "aprendizado" do algoritmo após a inserção do *training set* simula um professor supervisionando o processo de aprendizado de um aluno. Conhecemos as respostas corretas, o algoritmo faz iterativamente previsões nos dados de treinamento e é corrigido pelo professor, que sabe as respostas corretas. O desempenho do treinamento é avaliado comparando as previsões realizadas pelo algoritmo com os valores reais, usualmente chamados de rótulos.

A maioria dos problemas relacionados a machine learning utilizam métodos baseados em supervised learning como solução. Os problemas de *supervised learning* podem ser agrupados em dois tipos principais:

- Classificação: Ocorre quando a variável de saída é uma categoria, como "aprovado" ou “reprovado”, “falha” ou "normal", "infectado" ou "saudável", etc. As previsões neste caso são limitas às categorias conhecidas.
- Regressão: Ocorre quando a variável de saída é um valor real, como temperatura, valor do dólar, altura, etc. As previsões neste caso podem ser infinitas.

Destaca-se para este trabalho os métodos relacionados a regressão, tendo em vista que como o objetivo é realizar previsões no domínio do tempo, estas serão valores reais.

2.4 *Unsupervised Learning*

Unsupervised learning é um subconjunto de *machine learning*. Também comumente conhecido como auto-organização, seu objetivo é encontrar padrões anteriormente desconhecidos nos conjuntos de dados sem rótulos pré-existentes. Também é possível modelar densidades de probabilidade de determinadas entradas. As aplicações deste método são múltiplas, apesar de serem utilizadas em menor escala que o *supervised learning*.

Destaca-se para este trabalho dois tipos de algoritmos utilizados na fase de pré-processamento dos dados, são eles:

- Clustering
- Redução de dimensionalidade

2.4.1 *Clustering*

De forma simplificada, o objetivo da técnica *clustering* é encontrar diferentes grupos dentro dos conjuntos de dados. Em teoria, os elementos de dados que estão no mesmo grupo devem ter propriedades e/ou características semelhantes, enquanto os elementos de dados em grupos diferentes devem ter propriedades e/ou características altamente diferentes. Pode-se observar abaixo através de um exemplo simbólico uma ilustração de seu princípio de funcionamento.

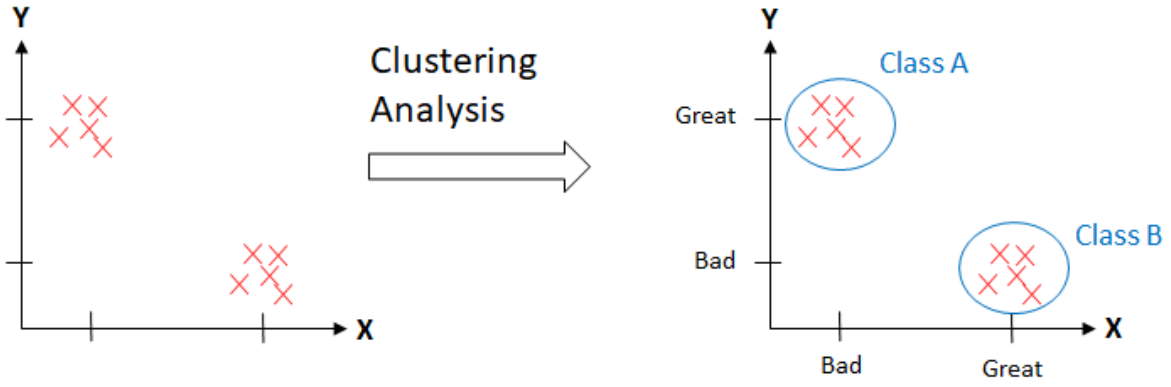

Figura 1 - Exemplo de *clustering*
Fonte: www.towardsdatascience.com/unsupervised-machine-learning-clustering-analysis-d40f2b34ae7e

Neste trabalho será utilizado o algoritmo *K-Means Clustering*. Este algoritmo separa os dados em grupos adequados com base nas informações que seu modelo matemático embutido já possui. Os dados são separados em K diferentes *clusters*. O número K geralmente é definido por meio de técnicas de visualização onde é possível observar separações suficientemente claras entre a distribuição de dados. Cada *cluster* possui um centro, chamado de centroide, e o cluster de cada elemento do conjunto de dados é determinado com base na distância euclidiana entre suas coordenadas e o centroide mais próximo.

O algoritmo *K-means* minimiza iterativamente as distâncias entre cada ponto de dados e seu centroide, a fim de encontrar a melhor solução para todos elementos. O procedimento é realizado da seguinte maneira:

1. K pontos aleatórios do conjunto de dados são escolhidos para serem centroides;
2. As distâncias entre todos os elementos e os centroides K são calculadas e armazenadas;
3. Com base nos cálculos da distância, cada elemento é atribuído ao *cluster* mais próximo;
4. Novas posições do centroide do *cluster* são atualizadas: semelhante à descoberta de uma média nas coordenadas dos elementos;
5. Se as coordenadas dos centroides forem alteradas, o processo será repetido da etapa 2, até que o novo centroide calculado permaneça o mesmo, o que indica que os elementos e os centroides dos clusters estão agora definidos.

Encontrar as distâncias mínimas entre todos os elementos do conjunto de dados implica que os elementos foram separados de forma que os *clusters* sejam os mais compactos possíveis, com a menor variação dentro deles. Em outras palavras, nenhuma outra iteração poderia ter uma distância média menor entre os centroides e os elementos encontrados neles.

O algoritmo *K-means* definido acima visa minimizar uma função objetivo, que neste caso é a função quadrada do erro, disponibilizada abaixo:

$$J = \sum_{i=1}^{k} \sum_{j=1}^{n} (||\boldsymbol{x_i} - \boldsymbol{v_j}||)^2 = \mathbf{1}$$

Equação 1 - Equação do algoritmo *K-Means clustering*

Cujo,

|| X*i* - V*j* || é a distância euclidiana entre os pontos X*i* e um centroide V*j* iterado em todos os pontos K no *iésimo* cluster, para todos os N clusters.

2.4.2 *Silhouette Analysis*

Para objetivos menos técnicos, o número K de *clusters* a ser configurado previamente para executar o algoritmo *K-means* é definido apenas utilizando técnicas de visualização, onde observando as separações visíveis nas distribuições de dados, detecta-se qual o número ideal de *clusters* a ser configurado. Entretanto, para fins de monografia, é preciso uma motivação matemática para determinar a quantidade ideal de *clusters*. Para isso, um teste chamado *Silhouette Analysis* será realizado. Esse teste retorna uma "pontuação" que determina o melhor número de *clusters* para ajustar-se ao algoritmo.

O *silhouette score* (Rousseeuw 1987), em essência, é a diferença normalizada da distância, capturando o quão perto um elemento está de outros elementos em seu próprio *cluster*, em comparação com os elementos do *cluster* mais próximo.

A fórmula para o *silhouette score* S(*i*) para um único ponto *i* pode ser observada abaixo:

$$\boldsymbol{s(i)} = \frac{\boldsymbol{b(i) - a(i)}}{\boldsymbol{max\{\, a(i), b(i) \,\}}}$$

Equação 2 - Equação da pontuação do teste Silhouette Analysis

Cujo,

$-1 \leq s(i) \leq +1$;

a(*i*) é a distância média entre *i* e todos os outros pontos em seu próprio *cluster*.

b(*i*) é a distância entre *i* o centroide de seu *cluster* mais próximo.

Em direção ao extremo positivo de S(i) = 1, tem-se que b(i) ≫ a(i) para que o ponto i esteja bem em cima de todos os outros pontos em seu *cluster* (a(i) = 0) ou o *cluster* mais próximo fica infinitamente distante (b(i) → ∞). Portanto, S(i) = 1 representa um cluster de boa qualidade.

Movendo-se para o extremo negativo de S(i) = -1, agora tem-se que a(i) ≫ b(i), de modo que o ponto i esteja no topo de todos os pontos em seu *cluster* mais próximo (b(i) = 0) ou todos os outros pontos em seu próprio *cluster* designado estão infinitamente distantes (a(i) → ∞). No último caso, o ponto i provavelmente deveria ter sido atribuído ao seu vizinho mais próximo, portanto, S(i) = -1 representa um *cluster* de má qualidade.

A fórmula no esquema acima representa o *silhouette score* de um único elemento em uma projeção que contém vários elementos, mas pode-se encontrar facilmente uma média do *silhouette score* em todos os pontos. Para fins de melhor visualização, costuma-se redimensionar a pontuação entre 0 e 1, onde 0 representa um *cluster* ruim e 1 representa um *cluster* perfeito.

2.4.3 Redução de Dimensionalidade

Redução da dimensionalidade é uma técnica de *unsupervised learning*. Seu objetivo como o nome sugere consiste na redução da dimensão dos parâmetros de um conjunto de dados comumente conhecidos como *features*. Cada *feature* geralmente é representado por uma coluna. A redução da dimensionalidade tem como o objetivo diminuir este número de colunas mantendo a maior quantidade possível de informações relevantes e originais. Neste trabalho implementar-se-á um algoritmo bastante usual nesta área chamado PCA (*principal component analysis*).

PCA é um procedimento estatístico que usa uma transformação ortogonal para converter um conjunto de observações de variáveis possivelmente correlacionadas (entidades que assumem vários valores numéricos) em um conjunto de valores de variáveis linearmente não correlacionadas chamadas componentes principais. Essa transformação é definida de tal forma que o primeiro componente principal tenha a maior variação possível (ou seja, é responsável por grande parte da variabilidade nos dados possível), e cada componente subsequente, por sua vez, possui a maior variação possível sob a restrição que é ortogonal aos componentes anteriores. Os vetores resultantes (cada um sendo uma combinação linear das variáveis e contendo N observações) são um conjunto de bases ortogonais não correlacionadas. O PCA é sensível à escala relativa das variáveis originais.

Hotelling (1933) inicialmente desenvolveu o PCA para explicar a estrutura de variância-covariância de um conjunto de variáveis combinando linearmente as variáveis originais. A técnica PCA pode ser responsável por a maior parte da variação das variáveis P originais via K componentes principais não correlacionados, onde K ≤ P. Para melhor entendimento: seja x = x_1, x_2…, x_p um conjunto de variáveis originais com uma matriz de variância-covariância ∑. Por meio do PCA, um conjunto de combinações lineares não correlacionadas pode ser obtido na seguinte matriz:

$$\boldsymbol{Y = A^T * x}$$

Equação 3 – Matriz de transformação PCA

Onde Y = (Y_1, Y_2…, Y_3)^T, Y_1 é chamado o primeiro componente principal, Y_2 é chamado o segundo componente principal e assim por diante; A = (a_{mn}) p × p e A é uma matriz ortogonal com A^TA = I. Portanto, X também pode ser expresso da seguinte maneira:

$$\boldsymbol{X = AY = \sum_{j=1}^{p} A_j Y_j}$$

Equação 4 - Transformação PCA

No qual A_n = [a_{1n}, a_{2n},…, a_{pn}]^T é o j-ésimo vetor próprio de ∑.
Consequentemente, as variáveis secundárias têm as seguintes características (Timm, 2002):

1. $\boldsymbol{\sum_{j=1}^{p} a_{ij}^2 = 1 \qquad \forall i = 1, \dots, p}$

2. $\boldsymbol{\sum_{k=1}^{p} a_{ik} a_{jk} = 0 \qquad \forall i, j \mid i \neq j}$

3. Cada variável secundária pode ser obtida a partir de uma combinação linear de variáveis originais.
4. A primeira variável secundária cobre o desvio máximo existente nas variáveis originais. Var($Z_1 = p_1'Y$) = $p_1'\Sigma p_1$ seria maximizado sujeito à restrição de que p_1' p_1. Foi demonstrado que o vetor característico associado à maior raiz da equação a seguir é a solução perfeita para p_1 e a maior raiz λ_1 é a variação de Z_1.

$$\boldsymbol{|\Sigma - \lambda I| = 0}$$

5. A variável secundária K-ésima cobre o desvio máximo que não é coberto pelo (K-1)-ésimo. Se a solução do item 4 for expressa como $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_p)$ tal que $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p$, K-ésimo componente seria o vetor característico associado a λ_k.
6. As variáveis secundárias são independentes.

2.5 Padronização

A padronização de dados é uma etapa fundamental na área de processamento de dados que converte a estrutura de um conjunto disperso de dados em um formato comum a toda a distribuição de dados. A padronização de dados permite que o usuário analise e utilize os dados de maneira mais clara e consistente. Normalmente, dados são adquiridos e armazenados em *databases*. Suas estruturas são dispostas de maneiras específicas cujo geralmente são desconhecidas pelo analista. Além disso, os conjuntos de dados que podem ser ou não semanticamente relacionados, podem ser armazenados e representados de maneira diferente, dificultando o trabalho do analista em interpretar a comparar as informações presente nos dados.

Existem duas categorias principais de casos de uso na padronização de dados: mapeamento *source-to-target* e reconciliação complexa. É comum dividir o primeiro em duas subcategorias, obtendo então três casos de aplicação:

1. Mapeamento simples de fontes externas: tem como objetivo manipular os dados cujo influencias externas afetam sua organização, mapeando valores-chave e os envolvendo em cálculos para a obtenção de um *output* de menor variabilidade.
2. Mapeamento simples de fontes internas: esta aplicação envolve o tratamento de conjuntos de dados internos baseados em definições inconsistentes e os transforma organizando-os de maneira confiável e diminuindo sua variabilidade.
3. Reconciliação complexa: esta categoria consiste no cálculo de métricas complexas cujo auxiliam na definição de uma semântica com base na lógica específica de cada caso.

Em todos os casos, a padronização utiliza fortemente estatísticas extraídas da distribuição de dados, em especial a média e o desvio padrão. Destaca-se para este trabalho o método descrito pelo item 1 acima.

2.6 *Outliers*

No ramo da estatística, *outlier* é um elemento do conjunto de dados que difere significativamente de outras observações. Sua ocorrência pode ser influenciada por diversos fatores, entretanto destaca-se as principais e mais comuns causas:

1. Variabilidade na medição
2. Erros de transformações
3. Erros experimentais

É importante entender que a maneira no qual se lida com os *outliers* depende das características dos mesmos e também da distribuição dos dados. O objetivo em questão também deve ser levado em consideração no momento de definir uma estratégia.

A julgar de forma generalizada, a forma mais adequada de lidar com *outliers* gerados devido ao item 1 citado acima, seria descartá-las ou utilizar ferramentas estatísticas robustas para valores discrepantes. Entretanto a respeito das causas 2 e 3, essas indicam que a distribuição possui alta assimetria, devendo-se ter muito cuidado ao usar técnicas ou intuições que assumem tal distribuição como normal.

Destaca-se para este trabalho apenas a definição teórica do termo *outlier* devido ao fato de que uma abordagem original será implementada para o objetivo do trabalho, não os classificando exclusivamente como causado por algum item específico citado acima.

2.7 *Standard Score*

Standard score, também conhecido como *Z-Score*, é definido no campo da estatística como o número fracionário assinado de desvios padrão pelo qual o valor de uma observação ou um elemento de dados está acima do valor médio do que está sendo observado ou medido. Valores observados acima da média têm pontuações positivos, enquanto valores abaixo da média têm escores padrão negativos. A computação de um *Z-Score* requer o conhecimento da média e do desvio padrão da totalidade da distribuição de dados no qual o elemento para o qual se deseja calcular esta pontuação pertence. A pontuação pode ser calculada por meio da equação abaixo:

$$\boldsymbol{Z} = \frac{\boldsymbol{x} - \mu}{\boldsymbol{\sigma}}$$

Equação 5 – Z Score

Cujo,

Z é a pontuação calculada para um determinado elemento presente em uma distribuição de dados

x é o elemento em questão

μ é a média da distribuição de dados

σ é o desvio padrão da distribuição de dados

As aplicações para o cálculo do Z-Score são inúmeras, entretanto destaca-se para este trabalho sua utilização para a detecção de elementos considerados *outliers* em uma distribuição de dados.

2.8 Transformações

Na área da estatística, a transformação de dados é a aplicação de uma função matemática determinística em cada elemento de um conjunto de dados, ou seja, cada elemento z_x é substituído pelo valor transformado $y_x = f(z_x)$, onde f é a função de transformação. Estas transformações geralmente são aplicadas para que os dados possam apresentar uma distribuição mais propícia a suposições de um procedimento de inferência estatística a ser aplicado ou para melhorar a interoperabilidade ou a aparência das visualizações geradas.

Na maioria dos casos, a função usada para transformar os dados é invertível e contínua. A função a ser aplicada também depende da estrutura da distribuição de dados e do objetivo em questão. Destaca-se para este trabalho o uso da transformação logarítmica.

Comumente chamada como “transformada log”, esta pode e geralmente é utilizada para tornar distribuições altamente ruidosas menos ruidosas. No caso deste trabalho este procedimento será importante para tornar a conexão das variáveis de entrada com as de saída mais linear com o objetivo de tornar a distribuição mais “interpretável” para o próximo passo. Em geral, existem duas circunstâncias em que a transformação logarítmica pode ser implementada:

1. Quando se enfrenta um problema de “mudança relativa”. A escala logarítmica informa sobre alterações relativas (multiplicativa), enquanto a escala linear informa sobre alterações absolutas (aditivo).
2. Quando for possível provar que a variação dos dados Y é proporcional ao valor esperado de Y^2.

No caso do item 1, faz total sentido sua aplicação para este trabalho tendo em vista que os algoritmos posteriormente treinados avaliarão relações não lineares entre os parâmetros de

entrada e seus respectivos rótulos. O item 2 também é amplamente atendido para o caso da distribuição de dados em questão, entretanto uma visualização que confirme esta proporcionalidade será evidenciada no capítulo 4 (Metodologia).

2.9 *Feature Engineering*

Feature engineering é uma técnica de processamento de dados que consiste na "tradução" de uma distribuição de dados em características pertinentes extraídas da mesma. Este processo utiliza o conhecimento do domínio dos dados para criar *features* a partir dos dados com o objetivo de adquirir informações nas quais os algoritmos terão mais facilidade em "interpretar" para a criação dos modelos preditivos. *Feature engineering* é fundamental para a aplicação do *machine learning*, e é considerado por muitos a etapa mais difícil e que requer maiores conhecimentos técnico-estatísticos da parte de pré-processamento de dados. A necessidade de engenharia manual de recursos pode ser evitada pelo aprendizado automatizado de recursos. A quantidade de *features* que podem ser extraídas de uma distribuição de dados é infinita e não existe receita para a execução deste procedimento. Algumas *features* são consideradas melhores que outras quando descrevem (ou traduzem) de maneira mais precisa possível os dados nos quais estas foram extraídas. A estrutura dos dados, o objetivo em questão e a modelo matemático base dos algoritmos de previsão são os fatores mais relevantes para a extração e seleção das melhores *features*.

Existem diversos métodos para a extração de *features* de um conjunto de dados, entretanto eles se resumem em manuais e automáticas. O primeiro com o próprio nome sugere, consiste na extração manual das *features* dos dados. Este método requer um conhecimento avançado de estatística pois o analista deve conhecer seus dados suficiente bem a ponto de, através de uma análise crítica, definir quais as *features* que melhor traduzirão os dados e depois extraí-las. Obviamente para realizar este procedimento de forma manual, o profissional deve ter pleno conhecimento de que informações determinadas *features* trazem dos dados.

Realizar o processo de *feature engineering* de forma automática tem se tornado cada vez mais popular pela facilidade de sua implementação. Ferramentas e bibliotecas robustas para este objetivo tem sido desenvolvida e aperfeiçoada constantemente e seus princípios de funcionamento são simples e na maioria das vezes extremamente semelhante: insere-se o conjunto de dados e as ferramentas "traduzem" os dados nas *features* automaticamente.

Certamente para fins de desempenho, o método manual atinge resultados mais precisos, pois extrai recursos personalizados dos dados de acordo com o objetivo em questão, entretanto é extremamente difícil de ser implementado da maneira correta.
Destaca-se para este trabalho o método de extração de *features* automático através de uma biblioteca exclusiva do Python para dados cujo a estrutura é baseada em sequencias temporais, chamada "*tsfresh"*.

3 APROFUNDAMENTO DE CASO

3.1 Origem dos Dados

Antes de iniciar uma profunda análise no campo do processamento de dados, é importante primeiro entender a origem dos mesmos.
Os dados foram fornecidos pelo Centro de Excelência em Prognósticos (PCoE) da NASA, unidade Ames – Lowa (Estados Unidos). Como pontuado no resumo deste trabalho, os dados são provenientes de simulações de sensores instalados em motores de turbina submetidos a estresse constante. A principal aplicação destes dados foi a sua utilização em uma competição de prognósticos e tratamento de dados denominada PHM08 *Data Challenge*. Esta competição teve início e fim no ano de 2008 e seus melhores resultados foram apresentados e premiados na Primeira Conferência Internacional de Prognósticos e Gestão de Saúde em outubro de 2008, Denver – Estados Unidos. Entretanto o desafio permanece aberto para os pesquisadores desenvolverem e compararem seus esforços com os vencedores do desafio em 2008. Os resultados podem ser avaliados submetendo as previsões no site oficial da NASA.

3.2 Desmembrando o Problema

A seguir são dispostas informações cruciais para o entendimento e interpretação do problema:

- Os dados consistem em diversas séries temporais contendo múltiplas variáveis.
- Os dados são divididos em dois conjuntos de dados chamados *training set* e *test set*.
- Cada série temporal, cujo será referido a parir daqui como "teste" é referente a um motor diferente, porém idêntico. Ou seja, os dados podem ser considerados de uma frota de motores do mesmo tipo.

- Todos os testes referem-se a motores usados, entretanto o tempo de operação antes do início de cada teste não é revelado.
- Os testes se iniciam com diferentes níveis de desgaste e variação de fabricação, nos quais também são desconhecidos pelo usuário. Este desgaste e variação são considerados normais para os padrões do equipamento, isso significa que não são considerados em condição de falha, ou seja, todos os testes se iniciam quando os equipamentos estão em condições normais de saúde.
- O motor está operando normalmente no início do seu respectivo teste e começa a se degradar em algum momento durante a série temporal.
- No *training set*, a degradação cresce em magnitude até que um limite predefinido seja alcançado além do qual não é preferível operar o motor, atingindo a condição de falha.
- No *test set*, a série temporal termina algum tempo antes da degradação completa, ou seja, neste conjunto de dados teste termina antes de atingir a condição de falha.

Abaixo, pode-se observar duas figuras exemplo para o melhor entendimento entre a diferença do *training* e *test set.*

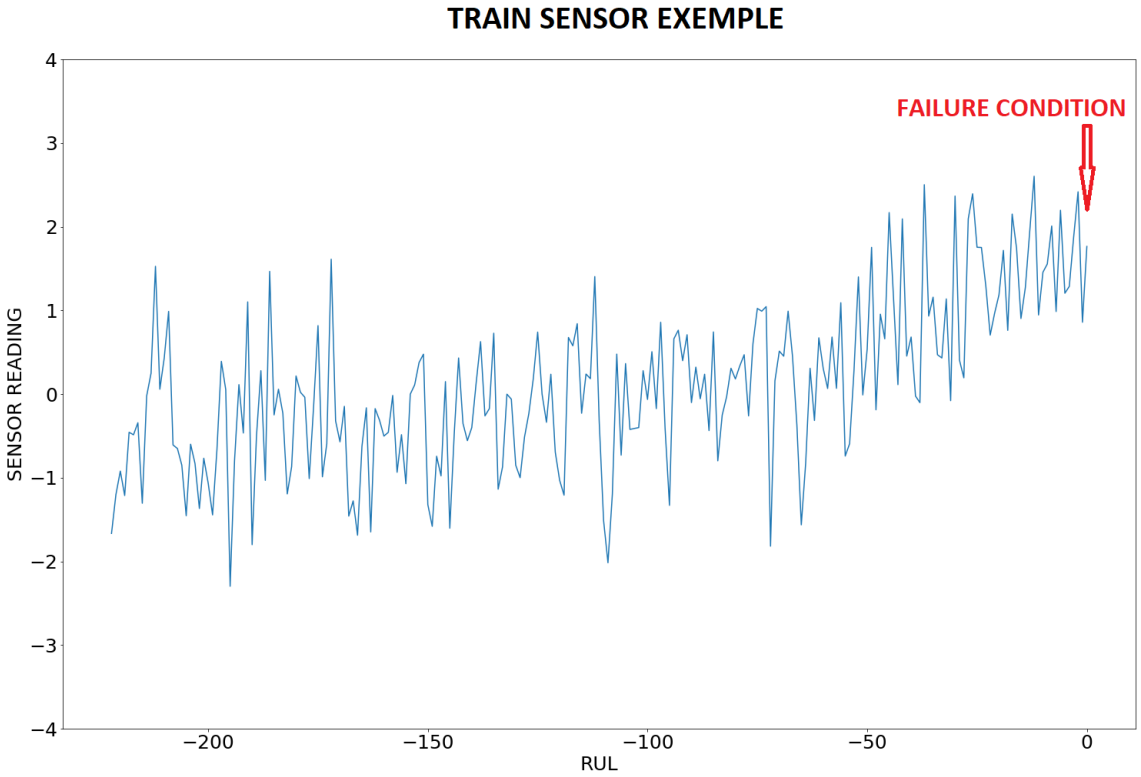

Figura 2 - Exemplo de teste do *training set*

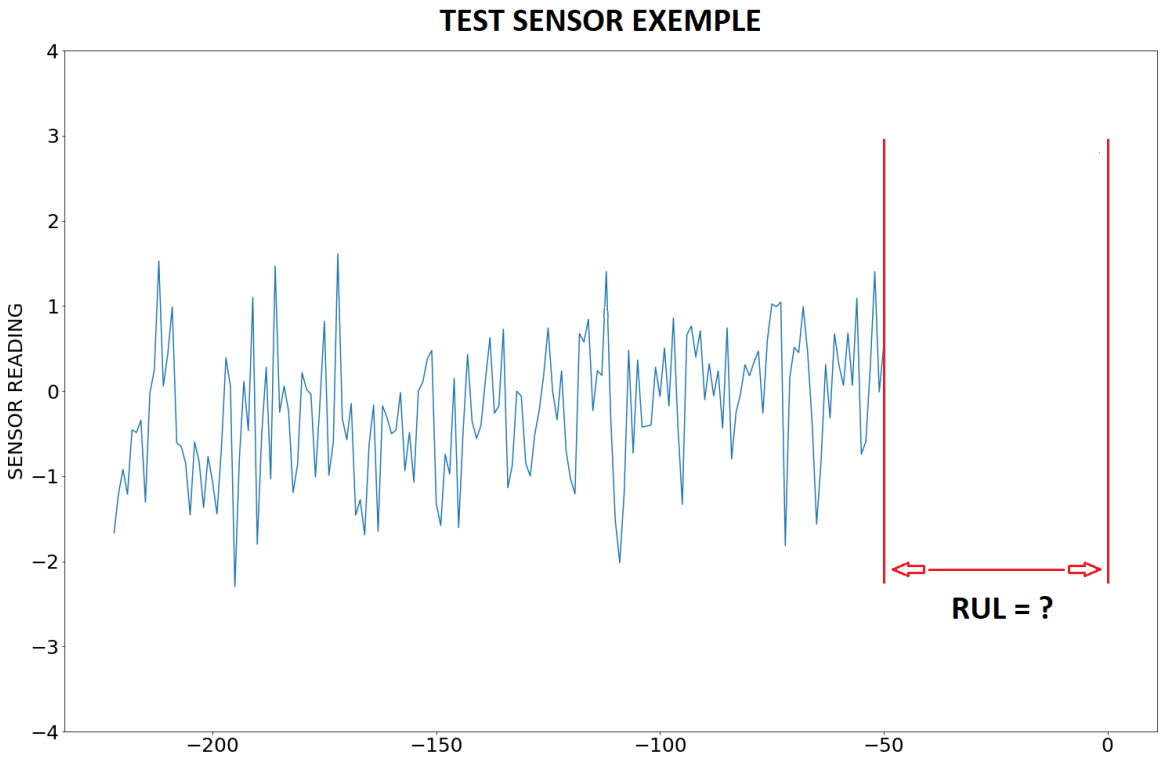

Figura 3 - Exemplo de teste do *test set*

Observa-se que na figura 3 o trecho da série temporal descrito por "*RUL = ?*" é exatamente o que objetiva-se prever ao final do trabalho.

- Existem também três configurações operacionais que afetam substancialmente o desempenho do motor. Essas configurações também estão incluídas nos dados.
- Os dados dos sensores são altamente ruidosos
- Ambos *training* e *test set* são compostos por 218 testes cada.
- O objetivo da competição no qual os dados foram utilizados é prever o número de ciclos operacionais em que o motor continuará operando adequadamente após o último ciclo lido pelos sensores no *test set.*

Para a competição, espera-se que os usuários treinem seus algoritmos utilizando dados do *training set* contidos arquivo "*train.txt*" e testem no *test set* cujo está contido no arquivo "*test.txt*". Os resultados obtidos são 218 *RULs* referentes aos 218 testes contidos no *test set*.

Os *RULs* verdadeiros (respostas consideradas perfeitas) não são revelados pelos fornecedores dos dados. Os usuários podem submeter suas previsões no site oficial da NASA e obter um feedback agregado de sua pontuação. Os resultados devem ser submetidos como um vetor coluna em extensão *".txt"* contendo 218 valores numéricos.

Uma tabela das 20 melhores pontuações obtidas durante a competição é disponibilizada a baixo como referência.

N°	Score
1	436,841
2	512,426
3	737,769
4	809,757
5	908,588
6	975,586
7	1049,57
8	1051,88
9	1075,16
10	1083,91
11	1127,95
12	1139,83
13	1219,61
14	1263,02
15	1557,61
16	1808,75
17	1966,38
18	2065,47
19	2399,88
20	2430,42

Tabela 1 - 20 melhores pontuações *PHM08 Data Challenge*

Como se pode perceber observando a tabela 1 acima, quanto menor o valor da pontuação adquirida, melhor é o resultado da solução apresentada.

A pontuação final é calculada a partir do somatório das pontuações individuais de cada teste, de forma que se a previsão de um determinado teste for zero, significa que este obteve uma previsão perfeita. No caso de a pontuação final for zero, significa que a pontuação individual de cada teste foi zero, e consequentemente o somatório das mesmas também será zero, o que significa que todos os testes obtiveram previsões perfeitas.

Pode-se entender em maiores detalhes como a pontuação é calculada observando a equação abaixo.

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

Equação 6 - Métrica oficial da competição no qual os dados pertencem

Cujo,

s é a pontuação final,

n é o número de testes

d = (RUL previsto – RUL verdadeiro),

a_1= 13,

a_2 = 10.

A equação 6 é uma função assimétrica que penaliza mais as previsões atrasadas que as antecipadas. É extremamente importante entender porque essa função apresenta este comportamento. Um breve exemplo que explique melhor esta relação é disponibilizada a seguir.

Considere-se duas previsões realizada para um determinado teste.

Teste Exemplo	
RUL verdadeiro	10
Previsão #1	8
Previsão #2	12

Tabela 2 - Exemplo explicativo da não linearidade da equação de avaliação

Neste exemplo, o RUL verdadeiro do teste exemplo é 10 e tem-se duas previsões diferentes para o mesmo teste. A previsão #1 sugere que o equipamento atingirá a condição de falha antes que isso efetivamente aconteça, adiantando a pausa da operação do equipamento desnecessariamente. Neste caso, o "prejuízo" se resume a dois ciclos operacionais desperdiçados no equipamento. Já a previsão #2 sugere que o equipamento atingirá a condução de falha dois ciclos operacionais depois do que este efetivamente atingiria. Neste caso o equipamento atingiria a condição de falha.

Comparando as consequências da previsão #1 (adiantada) em relação a previsão #2 (atrasada), e levando em consideração conceitos básicos da manutenção, é de razoável compreensão que previsões atrasadas sejam mais penalizadas que adiantadas em relação ao cálculo da pontuação que determina a eficiência das previsões.

3.3 Estrutura dos Dados

Para entender melhor como está disposta a estrutura dos dados, a planilha a seguir deve ser observada.

```
In [12]:
1 print(df_train.shape)
2 df_train[215:230]
```

```
(45918, 27)
```

Out[12]:

	UnitNumber	Cycle	OpSet1	OpSet2	OpSet3	SensorMeasure1	SensorMeasure2	SensorMeasure3	SensorMeasure4	SensorMeasure5	...	SensorMeasure13
215	1	216	20.0065	0.7000	0.0	491.19	608.06	1495.47	1276.75	9.35	...	2388.23
216	1	217	35.0031	0.8412	60.0	449.44	556.79	1378.59	1135.83	5.48	...	2388.13
217	1	218	41.9993	0.8400	40.0	445.00	550.25	1366.57	1146.32	3.91	...	2387.86
218	1	219	35.0073	0.8400	60.0	449.44	556.39	1381.50	1149.74	5.48	...	2388.11
219	1	220	35.0027	0.8400	60.0	449.44	556.65	1375.43	1149.45	5.48	...	2388.15
220	1	221	0.0008	0.0000	100.0	518.67	643.88	1593.26	1425.82	14.62	...	2388.28
221	1	222	20.0003	0.7001	0.0	491.19	608.00	1494.99	1270.18	9.35	...	2388.37
222	1	223	34.9992	0.8400	60.0	449.44	556.60	1385.96	1150.47	5.48	...	2388.18
223	2	1	0.0017	0.0006	100.0	518.67	641.59	1578.61	1401.71	14.62	...	2387.98
224	2	2	35.0026	0.8405	60.0	449.44	554.68	1357.66	1117.74	5.48	...	2388.07
225	2	3	42.0011	0.8400	40.0	445.00	548.63	1347.31	1118.90	3.91	...	2388.05
226	2	4	42.0003	0.8407	40.0	445.00	548.88	1350.73	1125.61	3.91	...	2388.05
227	2	5	20.0023	0.7000	0.0	491.19	606.99	1477.32	1237.14	9.35	...	2388.01
228	2	6	35.0044	0.8419	60.0	449.44	554.45	1357.29	1124.18	5.48	...	2388.03
229	2	7	24.9989	0.6217	80.0	462.54	536.41	1253.28	1048.21	7.05	...	2028.30

Figura 4 - Exemplo de estrutura dos dados do *original training set*

Esta é a planilha referente aos dados do *training set*. Observa-se que existem 45918 observações, que são descritas pelo exato número de linhas. Existem também 26 colunas, são elas:

- "UnitNumber": Informa qual o teste que as observações percetem. Os testes vão de 1 até 218.
- "Cycle": Informa o o ciclo no qual aquela observação pertence. Observe que o número de ciclos é resetado toda vez que que a coluna "UnitNumber" muda, isso significa que o teste anterior acabou e começou um novo. Cada teste possui um número particular de ciclos. Esta coluna representa o domínio do tempo.
- "OpSet1", "OpSet2" e "OpSet3": Descrevem as condições de operação que aquela observação possui. Como dito anteriormente, na descrição dos dados não é informado exatamente o que são essas condições de operação, podem ser condições específicas de operação no qual aquela observação foi submetida, como pressão, temperatura, etc.

- “SensorMeasure1 ~ SensorMeasure21”: Descreve as informações geradas pelas leituras dos sensores naquela observação. São 21 sensores no total.

Resumindo: Existem 45918 observações, cada observação é referente a um “UnitNumber” especifico cujo naquele momento está em um determinado momento descrito por “Cycle”. Para cada uma dessas observações tem-se três “OpSet” que descrevem a condição que as observações forem feitas e finalmente os 21 sensores que realizam 21 observações de perspectivas diferentes.

Para o *test set*, o número de observações é consideravelmente menor que no *training set* que acaba de ser exibido. Isso é razoável de entender porque os testes não chegam a condição de falha, então tendem a ter menos ciclos de operação.

4 METODOLOGIA

Neste capítulo será apresentado toda a metodologia empregada para resolver o problema exposto no capítulo 1. Para tal, todas as etapas realizadas serão descritas seguindo fielmente sua ordem cronológica. O embasamento teórico de todos os procedimentos pode ser conferido no capítulo 2.

Como na maioria de trabalhos que envolvem o tratamento maciço de dados com o objetivo de extrair informações e padrões úteis para as finalidades em questão, este também pode ser dividido em duas partes principais: pré-processamento de dados e desenvolvimento dos modelos preditivos. Na fase de pré-processamento são realizadas etapas que vão transformar os dados extremamente ruidosos e inconsistentes em informações concretas com padrões bem definidos para a otimizar interpretação e desempenho dos algoritmos. As etapas são, respectivamente: visualização, *clustering*, padronização, filtragem, normalização e *feature extraction.* Na fase de desenvolvimento dos modelos preditivos os dados adquiridos após a fase de pré-processamento são inseridos em algoritmos baseados em regressão e estes farão a previsão dos tempos de vida útil remanescente dos motores. Os resultados são avaliados submetendo as previsões no site oficial da NASA.

4.1 Estratégia

O objetivo final é testar diferentes algoritmos com o intuito de realizar uma análise comparativa de seus desempenhos para o problema em questão. Para isso, como em todo e qualquer problema relacionado ao tratamento de dados, é necessária uma parte de pré-processamento de dados para que as informações inseridas nos algoritmos sejam melhor interpretadas por eles e que os resultados sejam mais precisos. Um fluxograma que descreve todas as etapas do processo é disponibilizado abaixo.

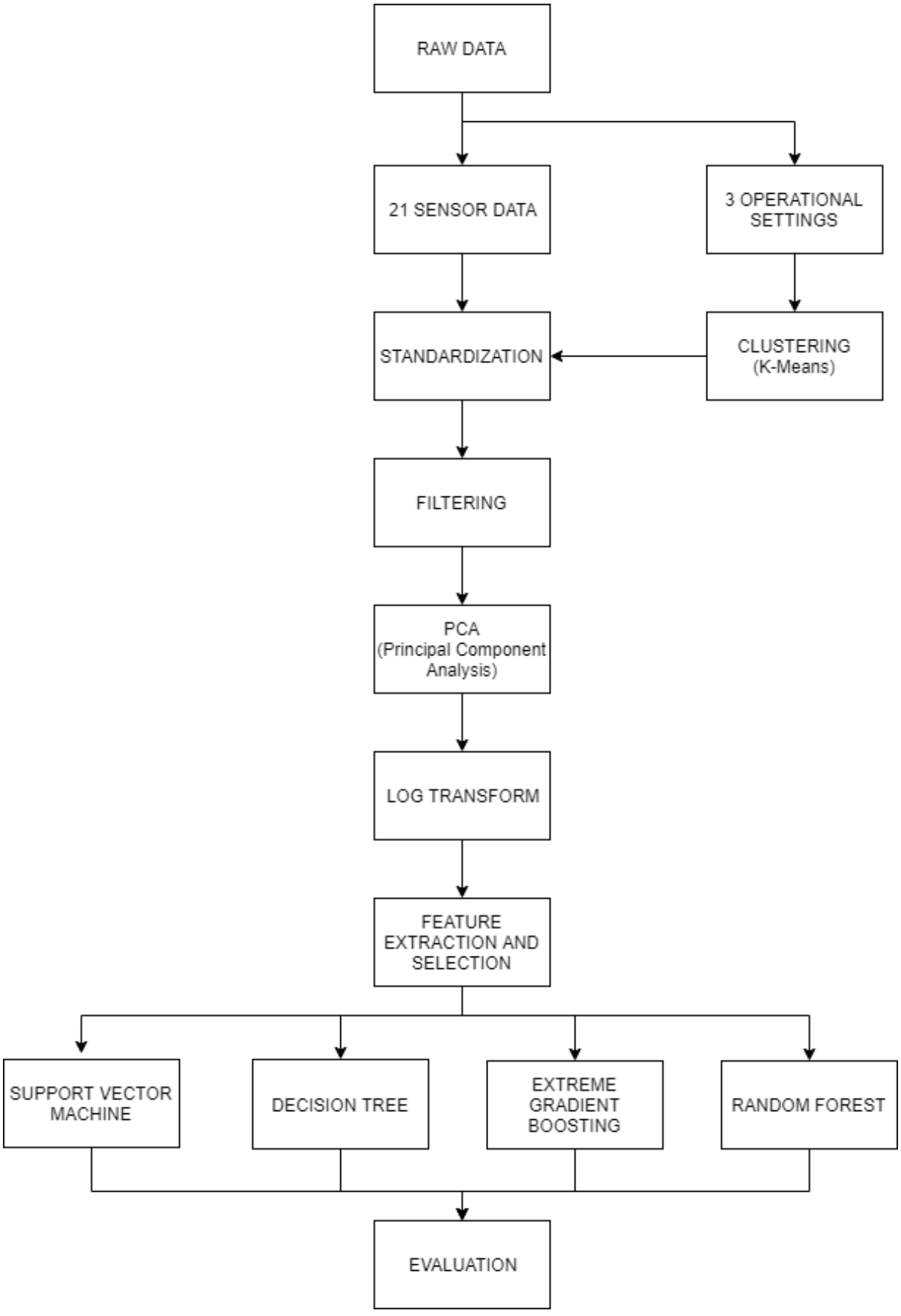

Figura 5 - Diagrama de blocos das etapas do trabalho

Cada etapa é de fundamental importância para o resultado final e para que a próxima etapa faça sentido. É interessante pontuar que no campo da análise e tratamento de dados, não existe o certo ou errado, mas inúmeras maneiras de atingir um resultado. Cada conjunto de dados possui suas particularidades, cada trabalho possui objetivos diferentes. Neste campo qualifica-se e quantifica-se a eficiência de soluções baseadas em seus desempenhos, que por sua vez são descritos por métricas que se adequam melhor em cada caso. No caso deste trabalho, a métrica de avaliação já foi definida pela equação 6 no capítulo 3 e será utilizada para avaliar o desempenho final.

Como parte da estratégia deste trabalho, este será dividido em três partes principais:

1. *Original training set*
2. *Artificial training set*
3. *Test set*

Na parte 1, realizar-se-á todo o estudo e tratamento dos dados. Todas as etapas realizadas nesta parte serão explicadas e os resultados de cada serão exibidos em forma de gráficos.

Também é importante ressaltar que, caso os dados permitam a criação de modelo preditivo perfeito, estes serão descritos pelos sensores do *training set*, tendo em vista que todos os seus testes são finalizados com o RUL = 0, ou seja, atingem a condição de falha. Por isso, algumas informações extraídas deste conjunto de dados serão salvas para futuros cálculos nas partes 2 e 3. Na parte 2, uma técnica que visa tornar o *training set* mais parecido com os dados nos quais os modelos serão submetidos para avaliação é realizado. Vários testes "artificiais" serão criados a partir dos testes "originais" do *training set*. Posteriormente todos as etapas realizadas na parte 1 também serão realizadas nesta parte em cima dos novos dados aqui confeccionados, denominados agora como *artificial training set*. Informações calculadas e salvas na parte 1 serão a base dos cálculos realizados nesta parte. O treinamento dos modelos preditivos, bem como as previsões e avaliações de seus desempenhos para uma porção limitada do artificial training set também serão realizadas nesta parte e servirão de base para a previsão final. Uma explicação mais detalhada deste procedimento será realizada posteriormente.

Na parte 3, todos as etapas de processamento realizadas na parte 2 serão repetidas para o conjunto de dados oficial para treinamento, o *test set*. Os dados devidamente processados serão submetidos às previsões de acordo com o melhor modelo treinado na parte 2. As previsões realizadas serão submetidas à avaliação oficial da competição disponível no site da NASA, que por sua vez emitirá a pontuação final que determina a eficiência de todo o trabalho realizado.

Como o *test set* contém dados truncados, ou seja, testes cujo não atingem o fim de sua vida útil, os testes do *training set* serão cortados em intervalos de tempo pré-determinados de forma a gerar testes “artificiais” a partir de testes “originais”. Conjunto de treinamento em momentos aleatórios durante a vida útil de uma unidade,

Vários testes artificiais serão criados a partir dos testes originais do *training set* com o objetivo de tornar um conjunto de dados para o treinamento mais semelhante aos que os modelos serão avaliados.

4.2 *Original Training Set*

4.2.1 Visualização de Dados

Algumas etapas do processamento de dados não realizam efetivas mudanças na distribuição de dados, entretanto como mencionado no tópico acima, não existem receitas pré-definidas para soluções de problemas no campo da ciência de dados. Manipular os dados e dispô-los de forma a exibir informações mais claras, limpas e interpretáveis é fundamental para entendimento dos padrões e estrutura dos dados, tornando mais suscetível o surgimento das ideias.

Sabendo que, apesar de os testes do training set possuírem tempos operacionais diferentes e consequentemente vidas úteis diferentes, todos terminam em um mesmo ponto: a condição de falha (RUL = 0). Então uma manipulação é realizada de forma a criar uma nova coluna em que descreve o RUL teste, nesta os testes começam com RUL = -X e vão até 0, simbolizando que os testes terminam com sua degradação máxima. Observa-se abaixo as leituras dos 21 sensores cujo o eixo horizontal é descrito pelo RUL dos testes.

Observação: todos os testes são exibidos separados por sensores.

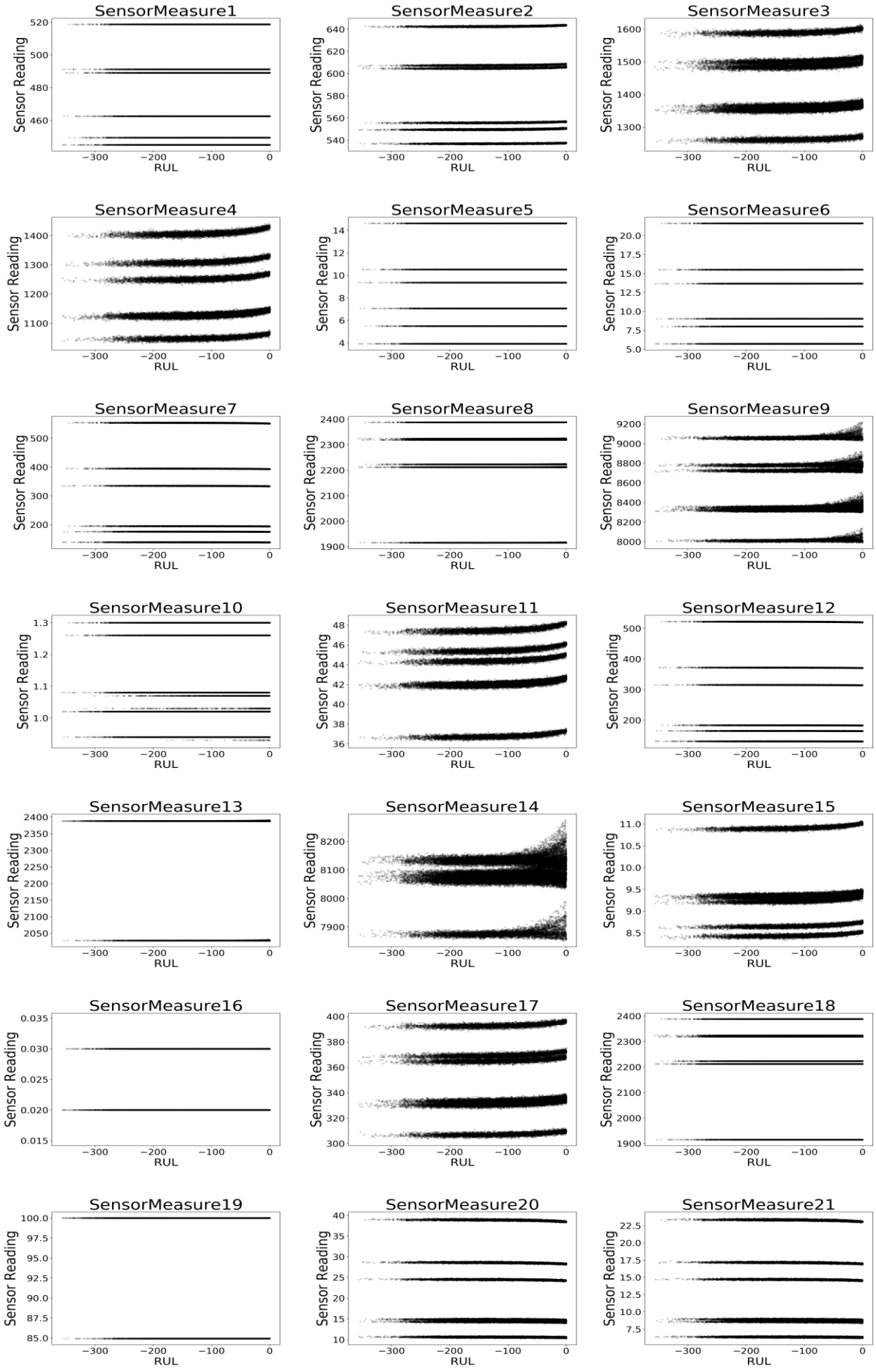
SensorMeasure1
SensorMeasure2
SensorMeasure3
SensorMeasure4
SensorMeasure5
SensorMeasure6
SensorMeasure7
SensorMeasure8
SensorMeasure9
SensorMeasure10
SensorMeasure11
SensorMeasure12
SensorMeasure13
SensorMeasure14
SensorMeasure15
SensorMeasure16
SensorMeasure17
SensorMeasure18
SensorMeasure19
SensorMeasure20
SensorMeasure21
Sensor Reading
RUL

Figura 6 - Todos os sensores "crus" do *original training set*

Observando os gráficos superficialmente já é possível inferir algumas informações:

1. A distribuição dos dados apresenta comportamentos diferentes e isto é visível em todos os sensores. As separações são muito claras, entretanto apenas com esta visualização é quase impossível entender por que isso acontece.
2. Na maioria dos sensores a distribuição dos dados tende a apresentar distúrbios característicos a medida que as amostras se aproximam do fim do teste.
3. Existem alguns sensores que parecem não apresentarem padrões concretos.

É extremamente improvável que em um sistema real onde todos os sensores instalados em equipamentos com a finalidade de adquirir dados em tempo real, sejam relevantes e exibam informações pertinentes. Os motivos pelos quais sensores são eventualmente descartados podem ser numerosos. Aparentemente, estes dados, mesmo provenientes de simulações, também simulam sensores inconsistentes, então considera-se que o exposto no item 3 acima é relativamente normal.

O item 2 representa um ponto positivo, pois o comportamento da distribuição de dados apresenta mudança em seus padrões à medida que se aproxima da condição de falha, relacionar esta mudança de comportamento com os níveis de degradação é o objetivo.

A informação mais importante gerada pelos gráficos acima certamente foi descrita pelo item 1. A separação evidente na distribuição de dados é um fator que deve ser estudado, caso contrário um algoritmo que classifique à qual comportamento os dados pertencem deve ser treinado antes de submetê-los aos algoritmos preditivos. Esta técnica é uma alternativa que pode ser implementada em último caso, entretanto definitivamente não seria o ideal pelo simples fato de que um algoritmo a mais seria envolvido, trazendo consigo suas possíveis taxas de erro e demandas extras de processamento computacional. Entender o que está influenciando esta enorme separação na distribuição de dados e traçar uma estratégia para lidar com esta situação é a prioridade no momento.

Observando a planilha dos dados percebe-se que ainda existem duas informações a serem analisadas: o número de ciclos de cada teste e as condições de operação. Um pouco de senso crítico é fundamental, não é razoável que a distribuição dos dados seja afetada de maneira tão intensa devido diferença entre o número de ciclos dos testes. Através de algumas linhas de código, obtém-se as seguintes informações:

- Existem 218 testes

- 114 testes possuem um tempo operacional exclusivo (quantidade de ciclos que só o teste em questão possui)
- O teste com menor tempo de operação possui 128 ciclos
- O teste com maior tempo de operação possui 357 ciclos

Analisando os números acima de forma rudimentar, é improvável que a separação comportamental dos dados, que é definida em 5 separações diferentes (dependendo de cada sensor), seja influenciada pelo número de ciclos de cada teste.

Restou-se o último conjunto de variáveis a ser analisado: as condições de operação.

Abaixo observa-se os gráficos cujo o eixo horizontal agora é definido pelo número de ciclos de cada teste, tendo em vista que as condições de operação, como o nome sugere, são variáveis externas e não faz sentido relacioná-las com o RUL dos testes.

Observação: novamente todos os testes são exibidos separados por condição de operação.

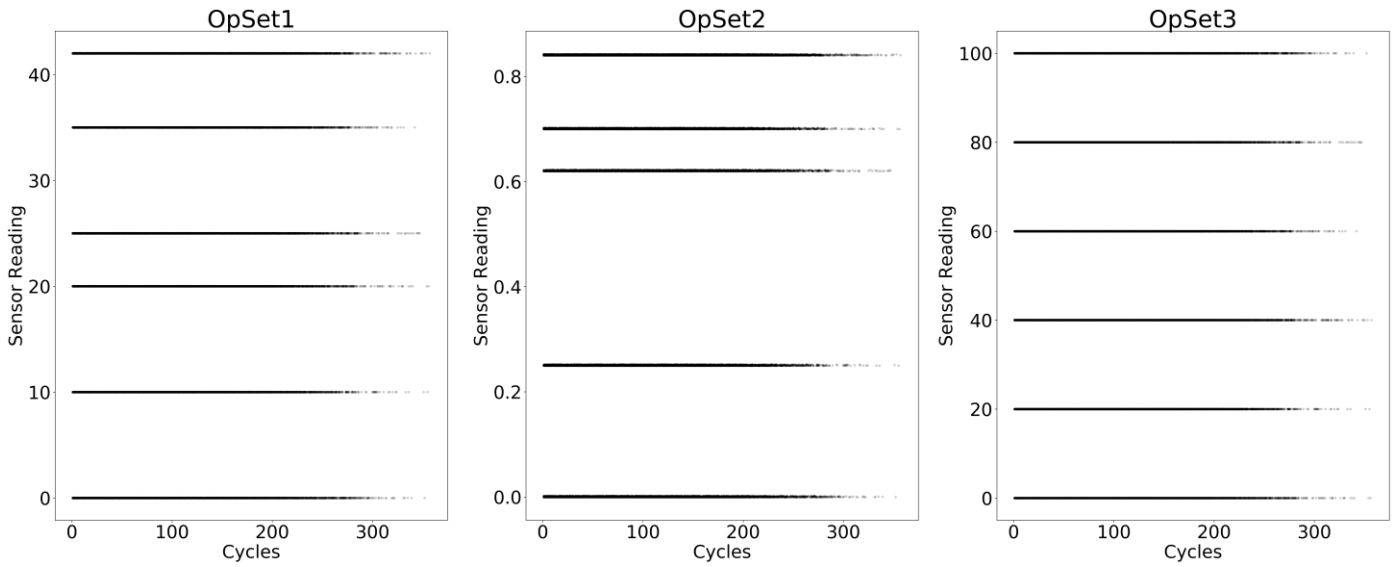

Figura 7 - Todas as condições de operação do *original training set*

Podemos observar que as condições de operação também possuem separações claras na sua distribuição de dados. Um pouco de conhecimento técnico, intuição e senso crítico é fundamental para tomar decisões de como se proceder. Apesar de não se ter conhecimento sobre o que as condições de operação representam, é razoável acreditar que essas têm potencial para influenciar as leituras dos sensores. Imagina-se por via de exemplo que essas três variáveis descrevam condições de pressão, temperatura e altitude. Componentes submetidos a estresse, e consequentemente suas curvas de degradação podem ser altamente influenciadas por esses tipos de variáveis. Somando o argumento exposto com o fato de

existirem apenas 3 colunas de dados que representem as condições de operação, talvez seja interessante gerar uma visualização 3D.

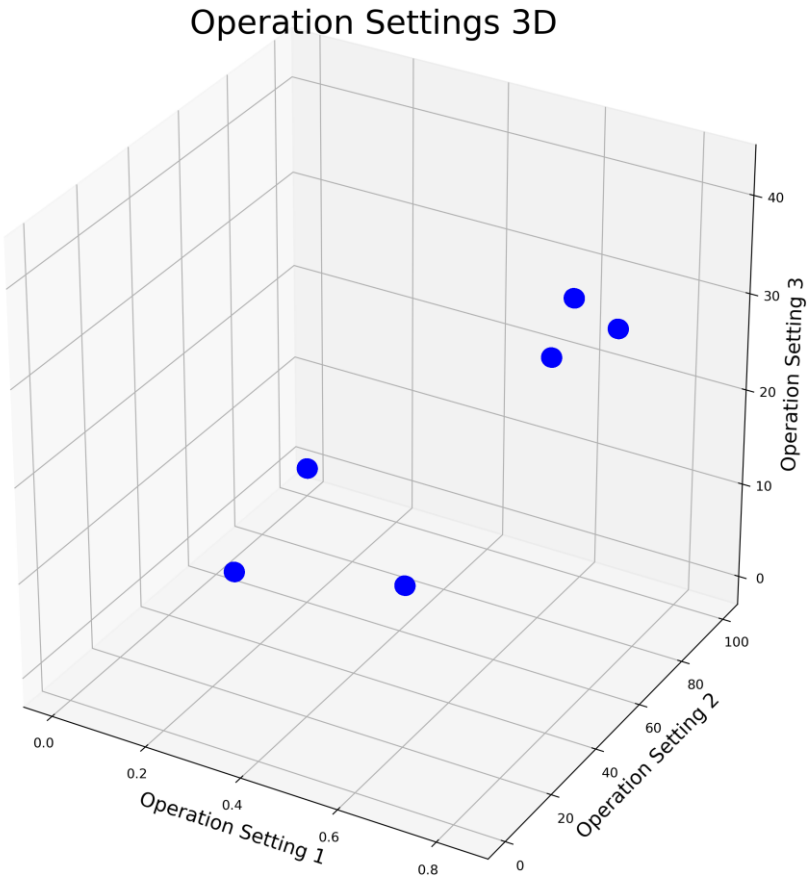

Figura 8 - Condições de operação visualização 3D

Apesar de ainda não significar muito, observa-se que as condições de operação estão integralmente distribuídas (a olho nu) em seis grupos distintos. Essa definitivamente é uma informação sugestiva tendo em vista que se aproxima do número comportamentos distintos exibidos pela distribuição dos dados.

4.2.2 *Clustering*

Para definir como se proceder a partir daqui, é importante relembrar o objetivo momentâneo: entender o que influencia as separações na distribuição de dados. Levando em consideração que no caso das condições de operações, essas são descritas por três colunas sem rótulos anexados, o problema em questão se encaixa em um caso de *unsupervised learning*. Sabendo

disto e após analisar a figura 8, a próxima etapa é submeter esses dados ao processo de *clustering*. Este procedimento irá classificar os diferentes grupos de dados em diferentes *clusters*, assim, cada observação presente do *training set* terá o seu *cluster* correspondente.

Para isso, será implementado um algoritmo chamado K-Means Clustering. Este, por sua vez, demanda dois parâmetros de entrada:

1. Os dados nos quais se objetiva calcular os clusters
2. O número de clusters a serem calculados

Para o item 1, os dados serão as três colunas que descrevem as condições de operação.

No caso do item 2, para objetivos menos técnicos, este número poderia ser facilmente definido como 6, tendo em vista que a visualização da figura 8 mostra que os dados são separados em 6 *clusters* diferentes. Entretanto, para finalidade deste trabalho, é preciso uma motivação matemática para determinar a quantidade ideal de *clusters*.

Para isso, um teste chamado *Silhouette Analysis* é realizado. Esse teste demanda como entrada a distribuição de dados a ser analisada e os números de clusters a serem testados. Sua saída é uma "pontuação" que determina a eficiência de N clusters para os dados inseridos. No caso deste trabalho, os cálculos foram feitos de forma a redimensionar a pontuação para limites de 0 a 1.

0 representa que o número N de *clusters* para os dados inseridos não os agrupou de forma eficiente. 1 representa que o número N de *clusters* agrupou os dados inseridos de forma perfeita.

Abaixo são exibidos os *silhouette scores* N clusters. Por questões de limitação de processamento computacional, o intervalo de N foi limitado entre 2 e 9.

```
For n_clusters = 2 The average silhouette_score is : 0.5777667217847624
For n_clusters = 3 The average silhouette_score is : 0.7228561018284831
For n_clusters = 4 The average silhouette_score is : 0.7882728525424852
For n_clusters = 5 The average silhouette_score is : 0.9194220366343638
For n_clusters = 6 The average silhouette_score is : 0.9998558910028272
For n_clusters = 7 The average silhouette_score is : 0.8940266135519652
For n_clusters = 8 The average silhouette_score is : 0.8307691370710419
For n_clusters = 9 The average silhouette_score is : 0.7700881538862213
```

Observa-se que a pontuação para N = 6 clusters foi quase 1, indicando que a implementação do algoritmo de *clustering* para os dados em questão, configurados para agrupá-los em 6 clusters diferentes será quase perfeito, confirmando a importância da visualização 3D da figura 8.

Feito isto, os dados das condições de operação são submetidos ao algoritmo *K-Means Clustering* e duas saídas são geradas:

1. As coordenadas centrais dos *clusters* calculados, denominados de centroides
2. Uma coluna extra que descreve a qual *cluster* cada observação presente no *training set* pertence.

Os valores descritos pelo item 1 acima são imediatamente salvos para cálculos futuros.

Com a coluna de dados descrita pelo item 2, uma nova técnica de visualização que simula o "ganho" de uma dimensão extra nos gráficos se torna possível.

Abaixo pode-se observar a mesma visualização exibida pela figura 7, entretanto para os gráficos a seguir, os dados serão exibidos separados por cores, que serão definidas de acordo com seus respectivos clusters.

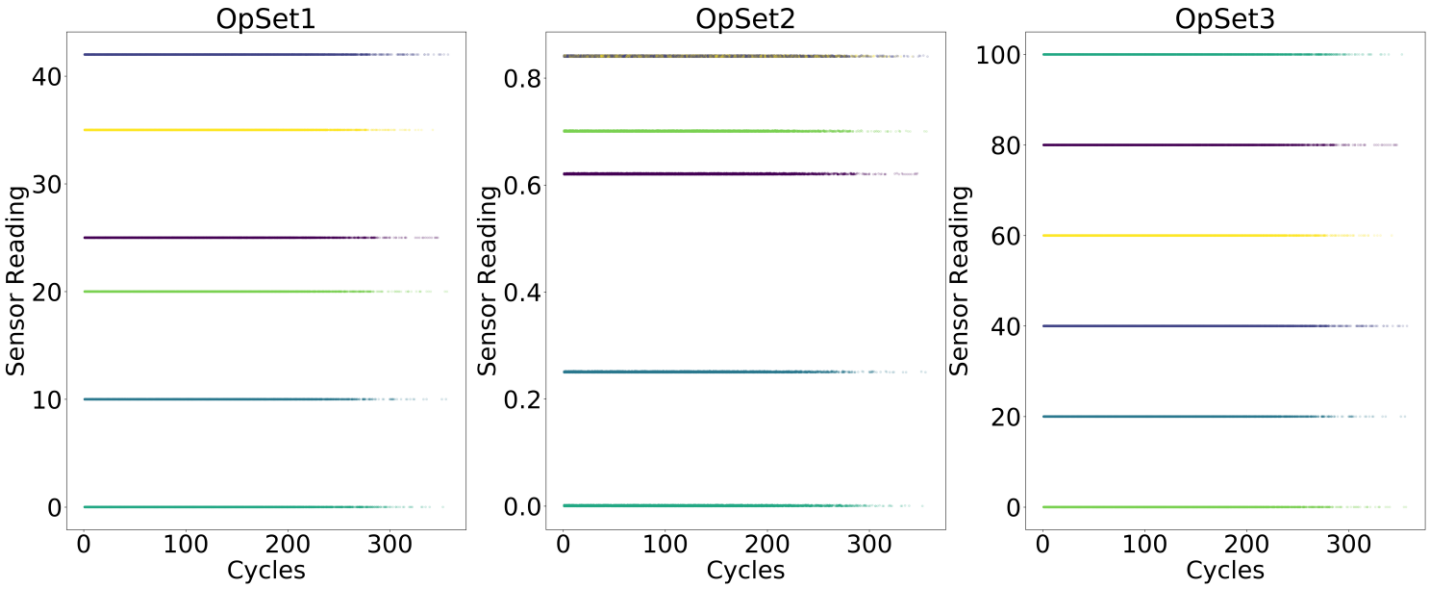

Figura 9 - Condições de operação, cores definidas por *cluster*

Interessante. As separações da distribuição dos dados das condições de operação são inquestionavelmente causadas pelos *clusters* embutidos nos dados. A mesma técnica é utilizada a seguir para separar por cor os dados dos sensores baseados nos *clusters* calculados.

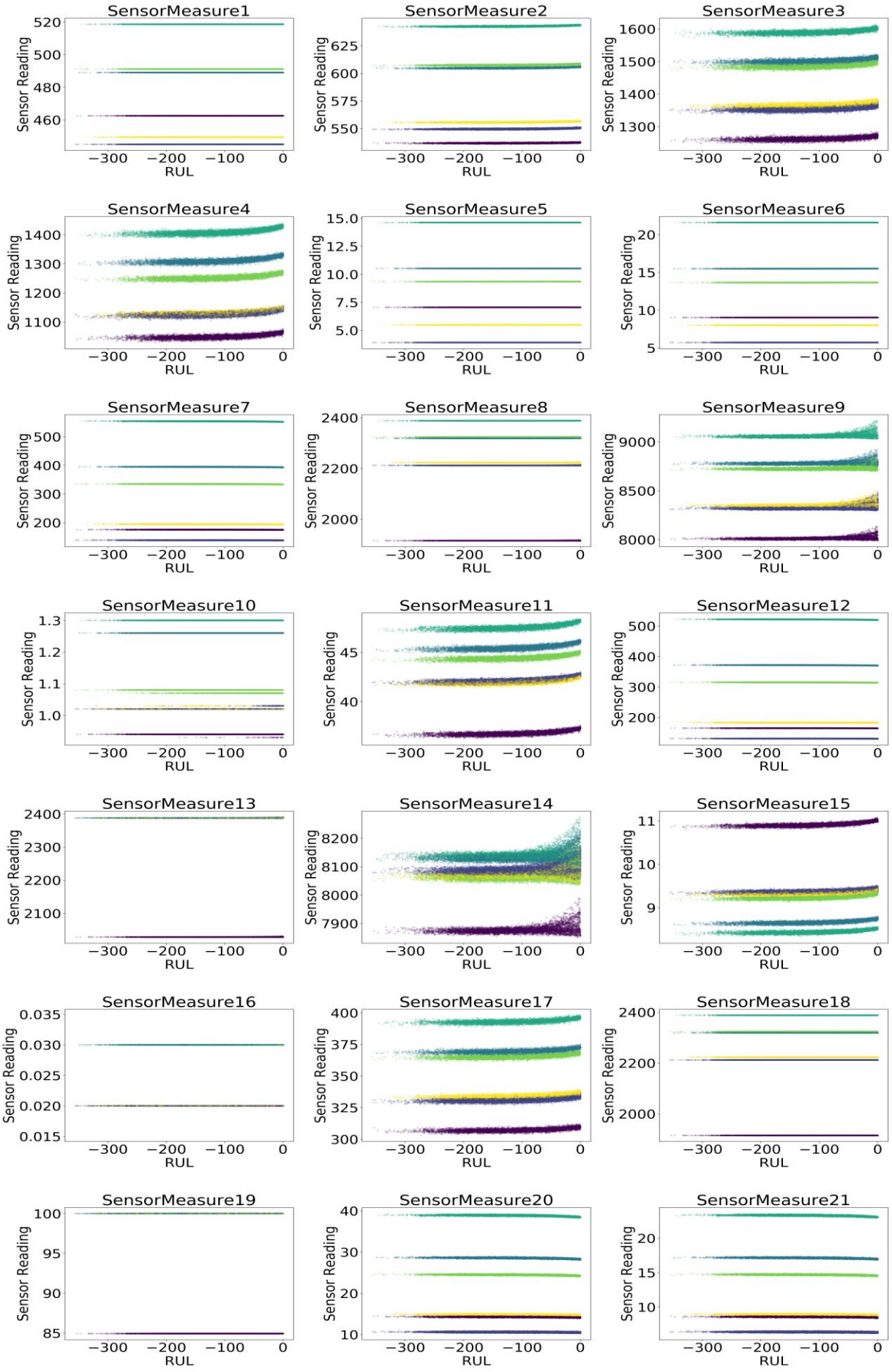
SensorMeasure1
SensorMeasure2
SensorMeasure3
SensorMeasure4
SensorMeasure5
SensorMeasure6
SensorMeasure7
SensorMeasure8
SensorMeasure9
SensorMeasure10
SensorMeasure11
SensorMeasure12
SensorMeasure13
SensorMeasure14
SensorMeasure15
SensorMeasure16
SensorMeasure17
SensorMeasure18
SensorMeasure19
SensorMeasure20
SensorMeasure21
Sensor Reading
RUL

Figura 10 - Todos os sensores crus, cores definidas por cluster

Excelente, de fato neste ponto pode-se afirmar com propriedade que as separações das distribuições de dados ocorrem devido ao efeito das condições de operação. Até os sensores cujo não apresentam padrões claros são afetados. Como pontuado no tópico 4.2.1, existe a possibilidade de se treinar um algoritmo extra que classificaria os dados em relação ao seu respectivo *cluster* para posteriormente treinar os modelos preditivos, entretanto não é a melhor opção. Uma técnica de padronização personalizada se encaixa bem neste caso e será implementada na próxima etapa.

4.2.3 Padronização

Detectado a causa das grandes separações nas distribuições de dados: as condições de operação, o objetivo neste momento é definir uma estratégia para lidar com este problema. A solução implementada foi uma técnica de padronização personalizada para este caso. A finalidade desta técnica é eliminar completamente o efeito das condições de operação de forma a extinguir a grande diferença comportamental da distribuição dos dados dos sensores. Este procedimento consiste em realizar a padronização separada por *cluster*, e pode ser resumida em duas etapas:

1. Separar os conjuntos de dados por *cluster*
2. Realizar o cálculo da padronização de acordo com a formula abaixo

$$\boldsymbol{S(new) = \frac{S(old) - S(mean)}{S(dev)}}$$

Equação 7 – Padronização

Cujo:

S(new): Valor calculado após a padronização

S(old): Valor original

S(mean): Média do sensor

S(dev): Standard Deviation do sensor

É importante ressaltar que as variáveis *S(new)* e *S(old)* descrevem as observações e são padronizadas individualmente, ou seja cada observação original é submetida individualmente à equação 7 e tem seu valor padronizado calculado.

Como os dados foram separados por cluster, tem-se agora 6 conjuntos de dados constituídos por 21 sensores cada. Neste caso, as variáveis *S(mean) e S(dev)* são estatísticas calculadas para cada sensor e para cada cluster, isso significa que cada uma dessas variáveis assumirão 126 valores diferentes (6 conjunto de dados * 21 sensores) ao longo do processo de padronização de todas as observações.

Feito isso, os dados agora padronizados são reagrupados de forma que a planilha retome suas dimensões originais.

Observação: É extremamente importante que no processo de reagrupamento, este aconteça respeitando rigorosamente os índices antigos (antes da separação por *cluster*), caso contrário a ordem cronológica dos testes será afetada.

A visualização dos sensores após a padronização personalizada pode ser conferida na figura 11 abaixo:

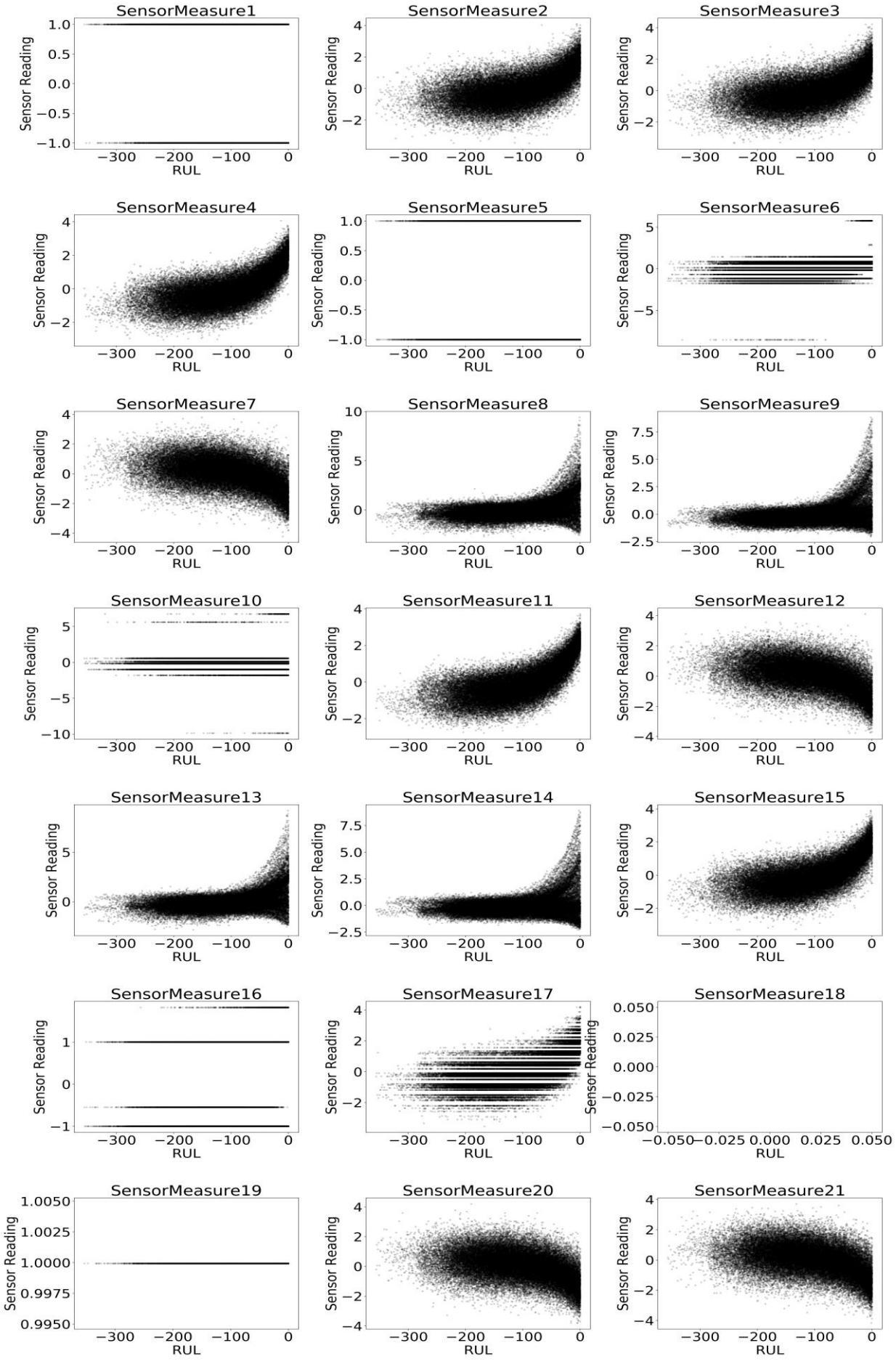
SensorMeasure1
SensorMeasure2
SensorMeasure3
SensorMeasure4
SensorMeasure5
SensorMeasure6
SensorMeasure7
SensorMeasure8
SensorMeasure9
SensorMeasure10
SensorMeasure11
SensorMeasure12
SensorMeasure13
SensorMeasure14
SensorMeasure15
SensorMeasure16
SensorMeasure17
SensorMeasure18
SensorMeasure19
SensorMeasure20
SensorMeasure21
Sensor Reading
RUL

Figura 11 - Todos os sensores após a padronização personalizada

Interessante, o efeito das condições de operação foi completamente removido e alguns padrões já começaram a se destacar. Observa-se que os gráficos em branco correspondem aos sensores que apresentam desvio padrão zero quando separados por *clusters*. Isso acontece pois quando a equação da padronização é ativada, esta realiza uma divisão pelo desvio padrão. Na maioria das linguagens de programação, como também é o caso do Python, a divisão por zero resulta em *NaN values* (valores em branco), consequentemente gerando os gráficos "vazios".

4.2.4 Filtragem

Após a realização da padronização personalizada nos sensores, as colunas de dados referentes as condições de operação podem ser descartadas. Os únicos dados relevantes a partir deste ponto são os sensores, cujo são distribuídos de acordo com seus respectivos testes e números de ciclos.

Dando continuidade ao processamento dos dados, após analisar figura 11, pode-se inferir que alguns sensores apresentam comportamentos semelhantes entre si, são eles:

- Grupo A: 2, 3, 4, 11, 15, 17
- Grupo B: 7, 12, 20, 21
- Grupo C: 8, 9, 13, 14
- Grupo D: 1, 5, 6, 10, 16, 18, 19.

O grupo D representa o grupo de sensores vazios e/ou sem padrões definidos, e não apresentam informações relevantes. Caso sejam utilizados para alimentar os modelos preditivos, a probabilidade de reduzirem seu desempenho é muito alta, então serão descartados.

A seguir uma visualização contendo os sensores remanescentes separados por grupo é exibida.

Grupo A

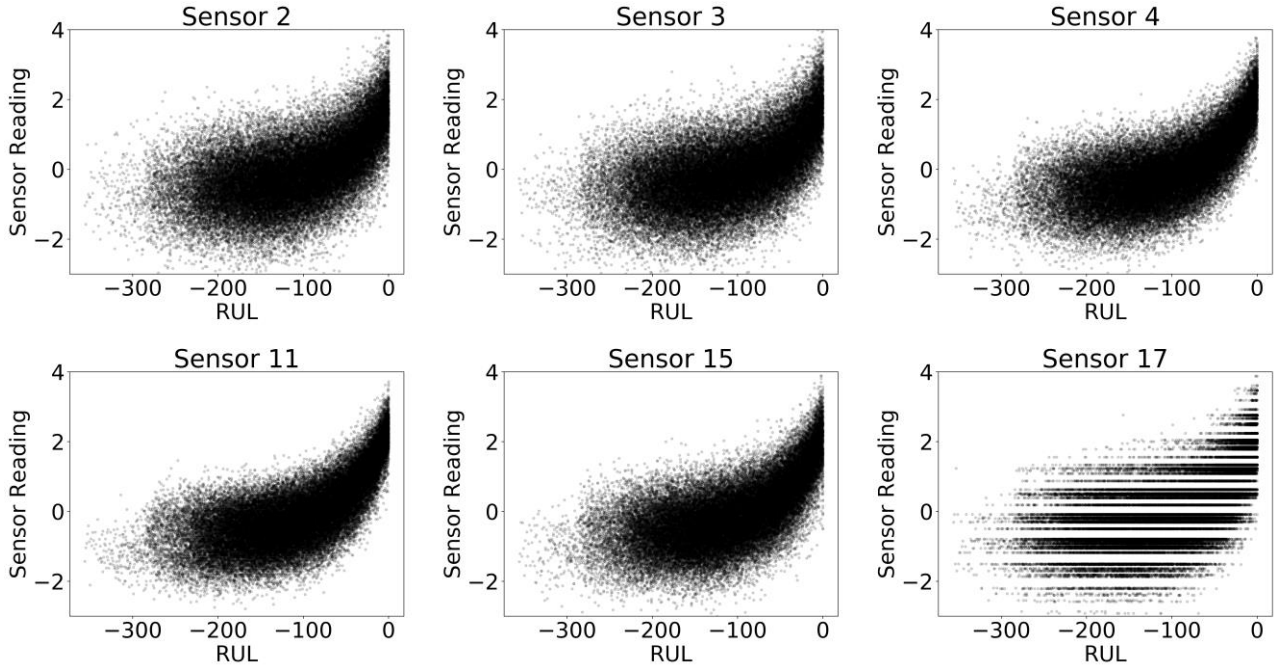

Figura 12 - Sensores do grupo A

Grupo B

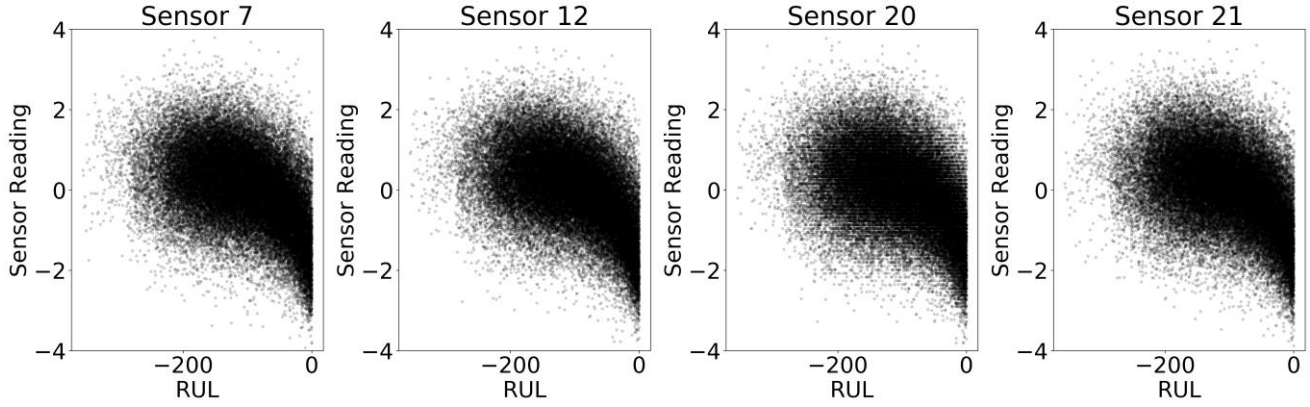

Figura 13 - Sensores do grupo B

Grupo C

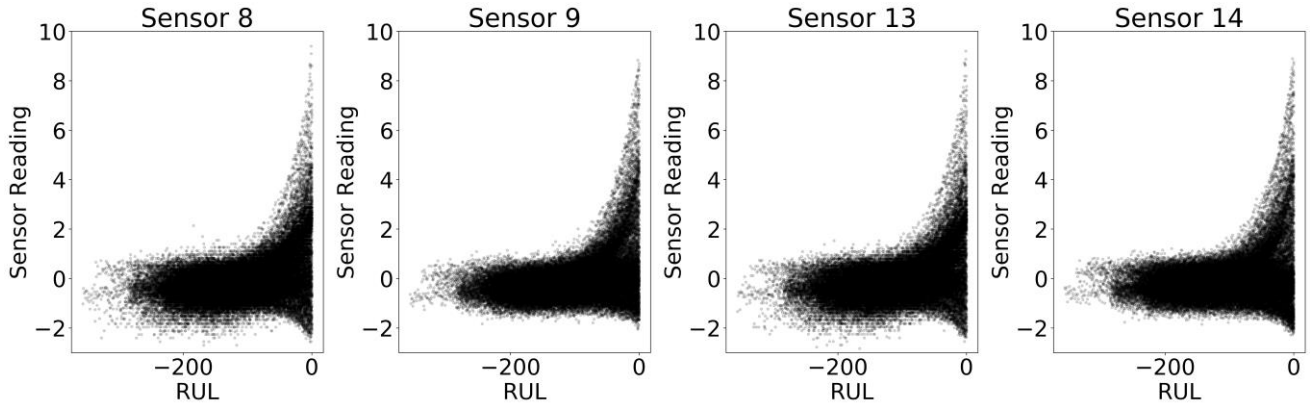

Figura 14 - Sensores do grupo C

Interessante, os sensores são bastante semelhantes se separados por grupos. O sensor 17 apresenta um comportamento peculiar. Com o objetivo de interpretar os sensores de forma mais eficiente, um método que visa entender melhor a variabilidade dos dados será executado e consiste na construção de um gráfico específico chamado *violin plot*.

Violin plot é um gráfico utilizado para visualizar a distribuição dos dados e sua densidade de probabilidade. Através da imagem exemplo abaixo é possível entender melhor como ele funciona.

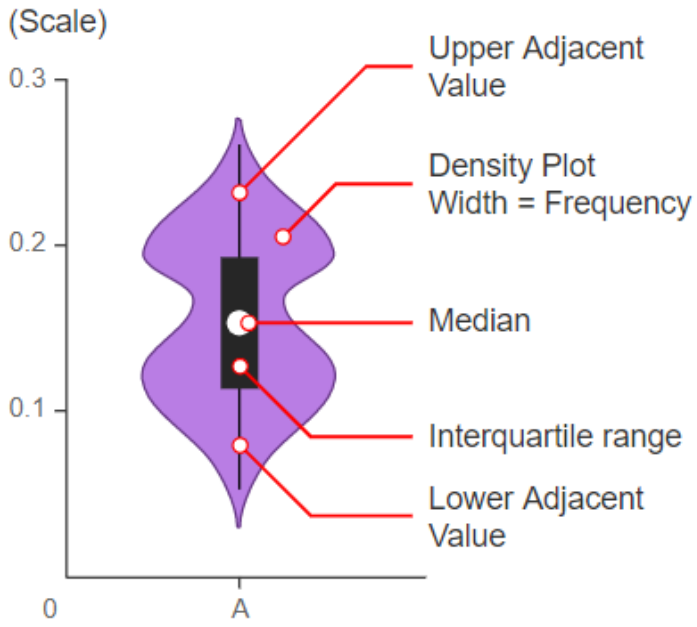

Figura 15 - Explicação do *violin plot*
Fonte: www.datavizcatalogue.com/methods/violin_plot.html

A seguir segue o *violin plot* para os sensores selecionados até este ponto.

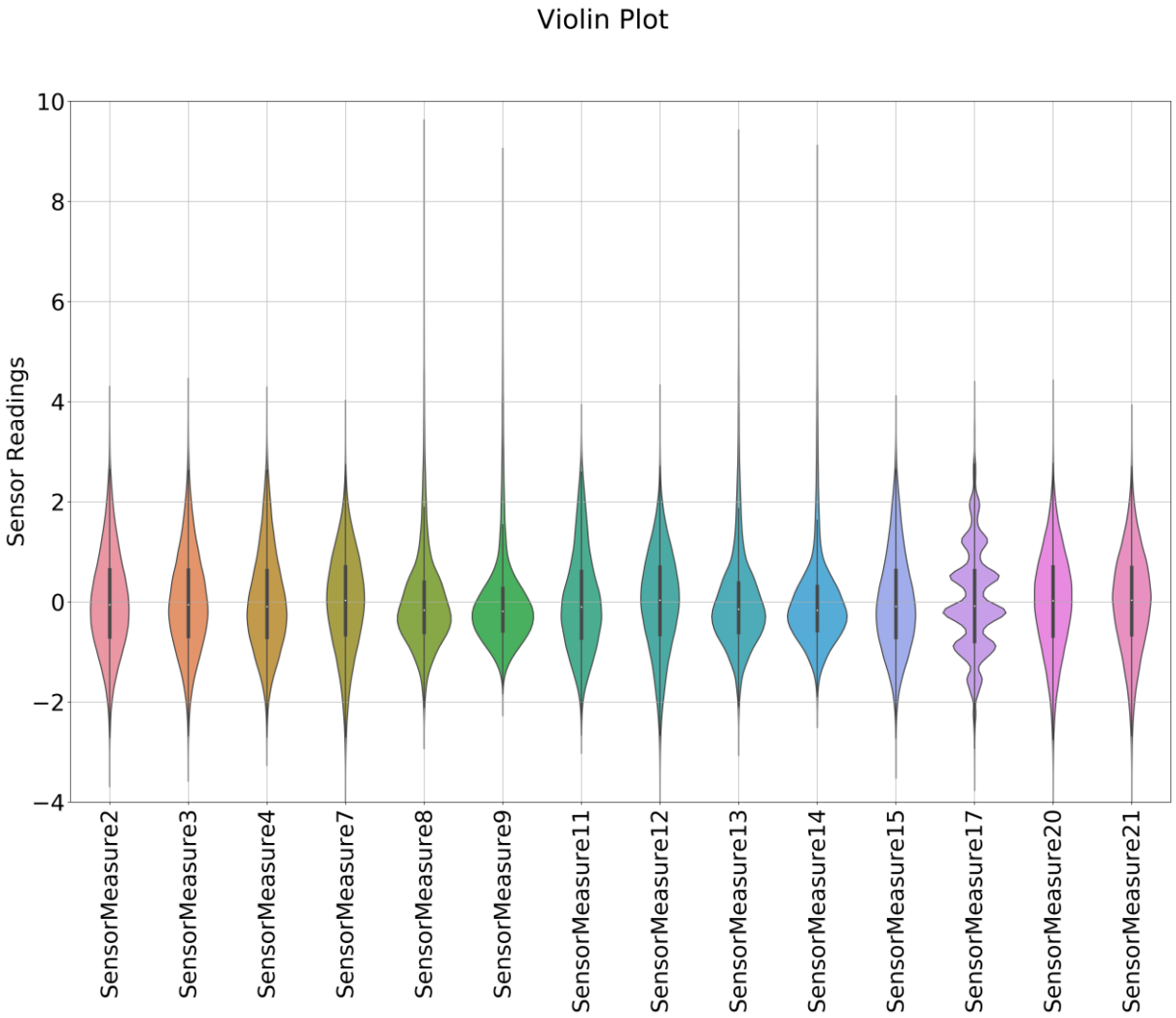

Figura 16 - *Violin plot* após padronização e primeira seleção de sensores

Observando a figura 16 acima, é possível inferir algumas informações:

1. O sensor 17 apresenta sua variação de densidade comprometida
2. Os sensores do grupo C (8, 9, 13, 14) apresentam altos valores adjacentes superiores e estão relativamente bastante afastados da média da distribuição dos dados

Levando em consideração que o grupo A é o que possui maior número de sensores, e que apenas o de número 17 se difere em relação aos demais, este também será descartado.

A respeito do item 2 pontuado acima, o comportamento dos valores adjacentes superiores pode significar a existência de *outliers*. Para lidar com este problema uma abordagem baseada no modelo matemático *Z-Score* será implementada.

Z-Score, de forma resumida, pode ser entendido como o número sinalizado de desvios padrões pelo qual o valor de um elemento está acima do valor médio da distribuição de dados a qual pertence. Na maioria dos casos envolvendo sua utilização, as pontuações calculadas entre o intervalo -3 e 3 definem as observações como normais. Os elementos cujo apresentam

Z-Scores acima e abaixo dos limites mencionados são considerados *outliers*. Entretanto para este caso, com o objetivo de conservar o máximo possível as informações originais dos sensores, os limites serão definidos de -4 a 4. É importante ressaltar também que os sensores foram submetidos ao processo de remoção dos *outliers* separados por grupo, para manter a confiabilidade da pontuação calculada, já que para o cálculo desta são envolvidas estatísticas de toda a distribuição de dados, conforme explicado no tópico 2.7.

Observa-se abaixo o novo *violin plot* sem a presença do sensor 17 e após a remoção dos *outliers*. Os sensores são exibidos lado a lado separados por seus respectivos grupos.

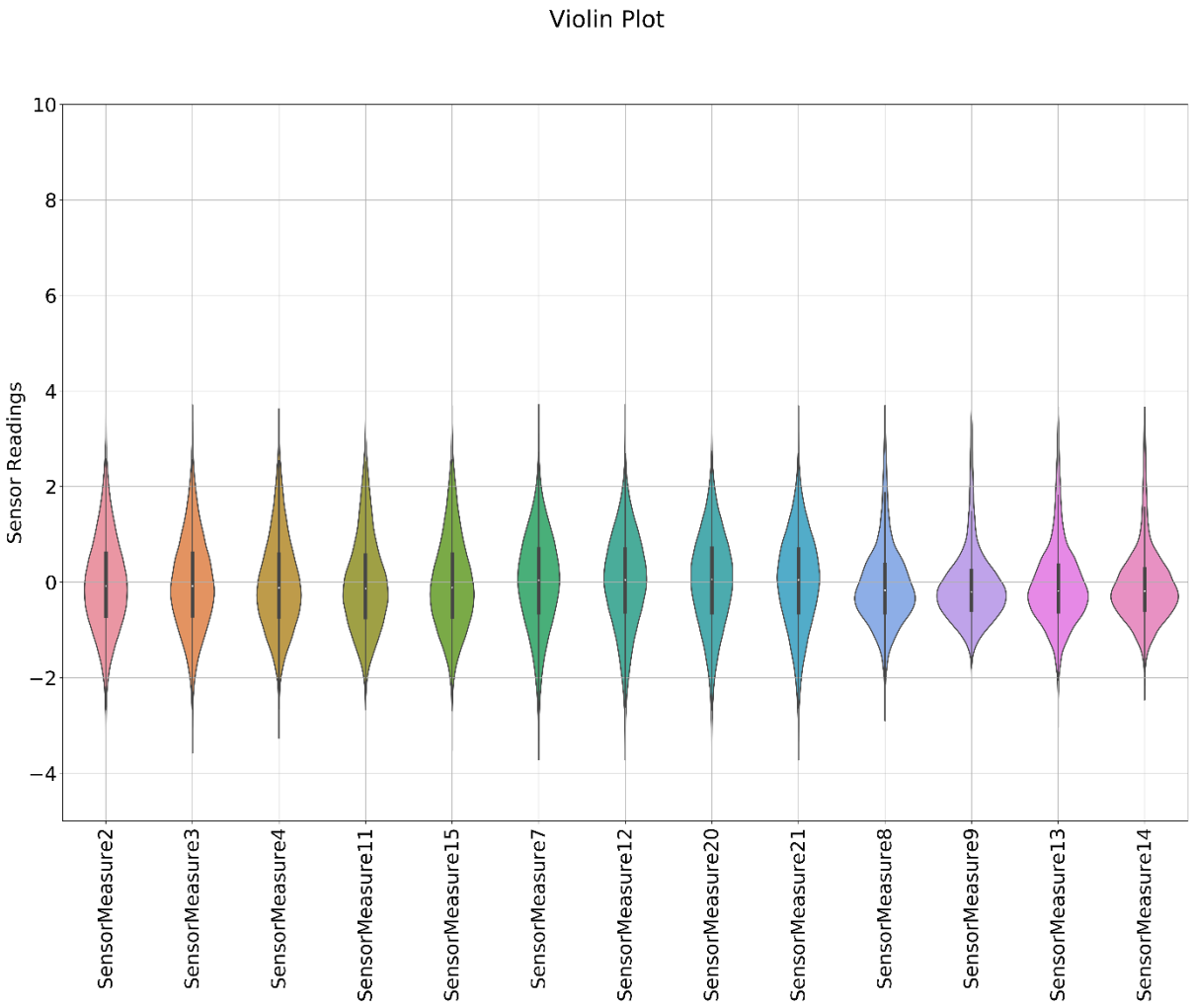

Figura 17 - *Violin plot* após remoção de *outliers* e seleção completa de sensores

É de crucial importância ressaltar que 903 observações foram removidas. Este número corresponde a aproximadamente 2% do conjunto original de dados. Também é importante entender que como estas observações foram removidas, isto significa que o número de ciclos operacionais da maioria dos testes do *original training set* foi reduzido. Apesar disto,

considera-se que o RUL de cada teste permanece o mesmo definido pela descrição original dos dados: zero.

Observa-se na figura a baixo a nova visualização dos sensores após a finalização do processo filtragem.

Grupo A

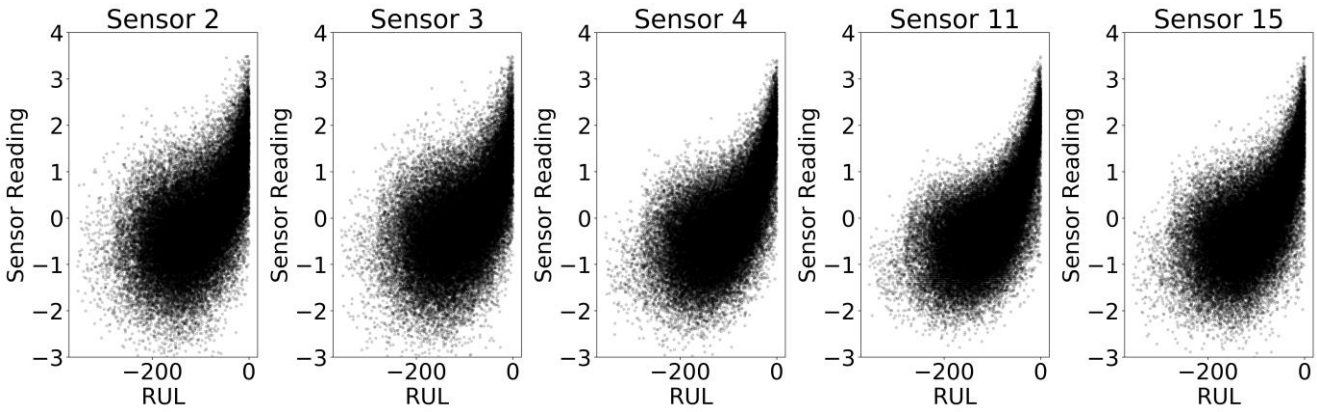

Figura 18 - Grupo de sensores A após filtragem

Grupo B

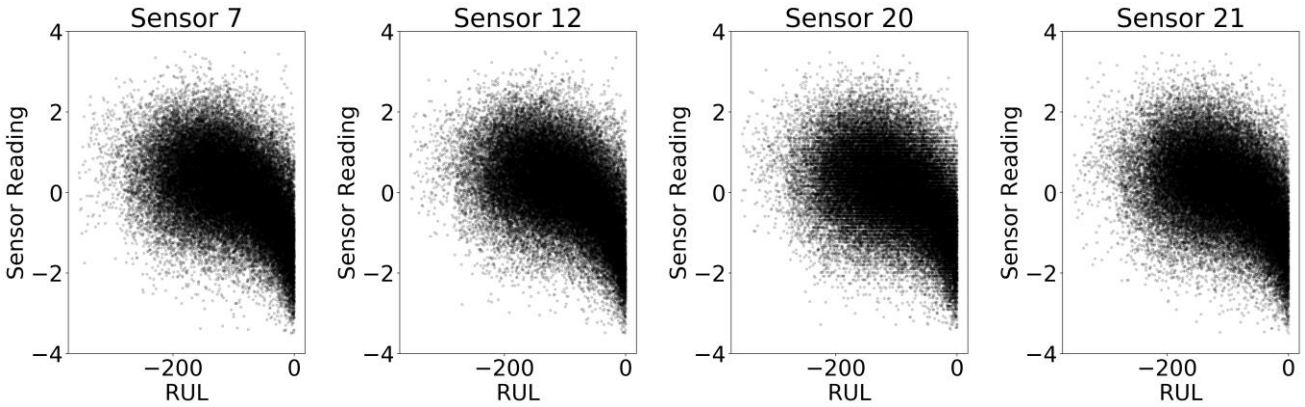

Figura 19 - Grupo de sensores B após filtragem

Grupo C

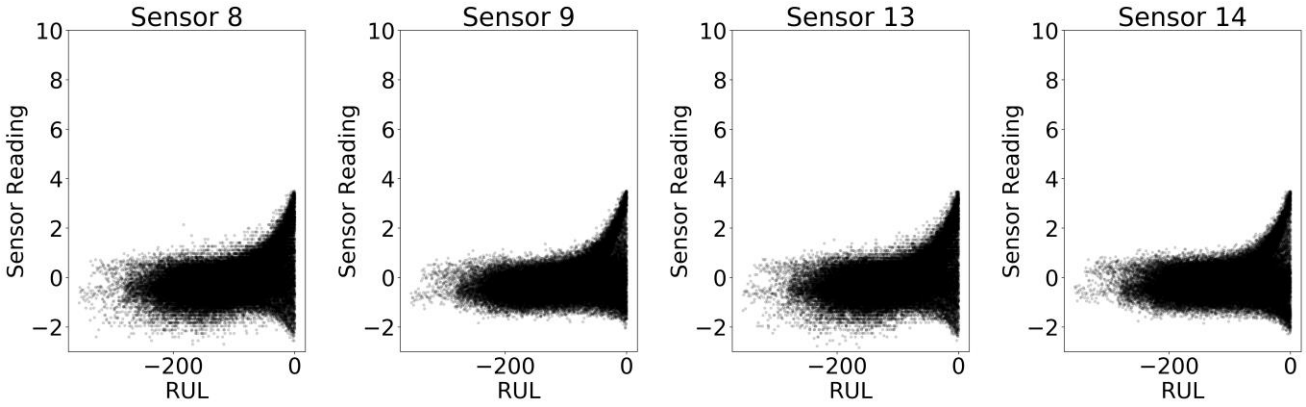

Figura 20 - Grupo de sensores C após filtragem

Interessante, podemos observar que o grupo de sensores C foi o mais afetado pelo processo de remoção dos *outliers*. A partir deste ponto este são os sensores que seguirão às próximas etapas de processamento.

4.2.5 PCA (*Principal Component Analysis*)

Nesta fase do processamento de dados, os sensores foram reduzidos até que todos estejam em condições de trazer informações pertinentes. Entretanto, observa-se que os sensores, separados por grupo, apresentam comportamentos extremamente semelhantes. Por este motivo, talvez seja interessante realizar uma técnica de redução de dimensionalidade para evitar a necessidade de processamento computacional em dados redundantes. Abaixo é exibido a matriz de correlação dos sensores separados por grupo e suas respectivas médias de correlação.

Observação: 1 representa que os sensores comparados são idênticos e 0 representa que os mesmos são totalmente diferentes.

SENSOR A

```
Correlation average
 0.7054330233244711
```

	SensorMeasure2	SensorMeasure3	SensorMeasure4	SensorMeasure11	SensorMeasure15
SensorMeasure2	1	0.49	0.6	0.64	0.59
SensorMeasure3	0.49	1	0.58	0.62	0.57
SensorMeasure4	0.6	0.58	1	0.76	0.71
SensorMeasure11	0.64	0.62	0.76	1	0.76
SensorMeasure15	0.59	0.57	0.71	0.76	1

Figura 21 - Matriz de correlação dos sensores do grupo A

SENSOR B

```
Correlation average
 0.5301073177337383
```

	SensorMeasure7	SensorMeasure12	SensorMeasure20	SensorMeasure21
SensorMeasure7	1	0.42	0.36	0.36
SensorMeasure12	0.42	1	0.38	0.38
SensorMeasure20	0.36	0.38	1	0.33
SensorMeasure21	0.36	0.38	0.33	1

Figura 22 - Matriz de correlação dos sensores do grupo B

SENSOR C

Correlation average
0.7536189219627256

	SensorMeasure8	SensorMeasure9	SensorMeasure13	SensorMeasure14
SensorMeasure8	1	0.57	0.92	0.52
SensorMeasure9	0.57	1	0.58	0.91
SensorMeasure13	0.92	0.58	1	0.52
SensorMeasure14	0.52	0.91	0.52	1

Figura 23 - Matriz de correlação dos sensores do grupo C

Pode-se perceber que os grupos de sensores são extremamente similares entre si, principalmente os grupos A e C, cujo a correlação ultrapassa 70%. De acordo com os números apresentados, considera-se que a motivação matemática para definir os sensores como redundantes foi alcançada, justificando a utilização da técnica de redução de dimensionalidade.

O algoritmo a ser utilizado chama-se PCA (*Principal Component Analysis*), e se encaixa na categoria de *unsupervised learning*, por não utilizar rótulos anexados aos dados. Este algoritmo tem como entrada um conjunto de dados e a dimensão (descrita por um número inteiro) no qual se objetiva que os dados sejam convertidos. De forma simplificada, este algoritmo analisa as entradas, no caso os sensores, retira as informações originais de cada um e "cria" um novo sensor que será a "tradução" dos sensores previamente inseridos. Também é importante ressaltar que este algoritmo foi implementado três vezes, cada uma sendo alimentada por um grupo diferente de sensores. Para cada vez que esta técnica é utilizada, uma matriz de transformação é confeccionada e multiplicada aos dados originais de forma a convertê-los para a dimensão pré-definida. As três matrizes de transformação são salvas para futuros cálculos nas partes 2 e 3 deste capítulo.

Observa-se abaixo o resultado dos após a implementação do algoritmo PCA.

Figura 24 - Sensor A após redução de dimensionalidade

Figura 25 - Sensor B após redução de dimensionalidade

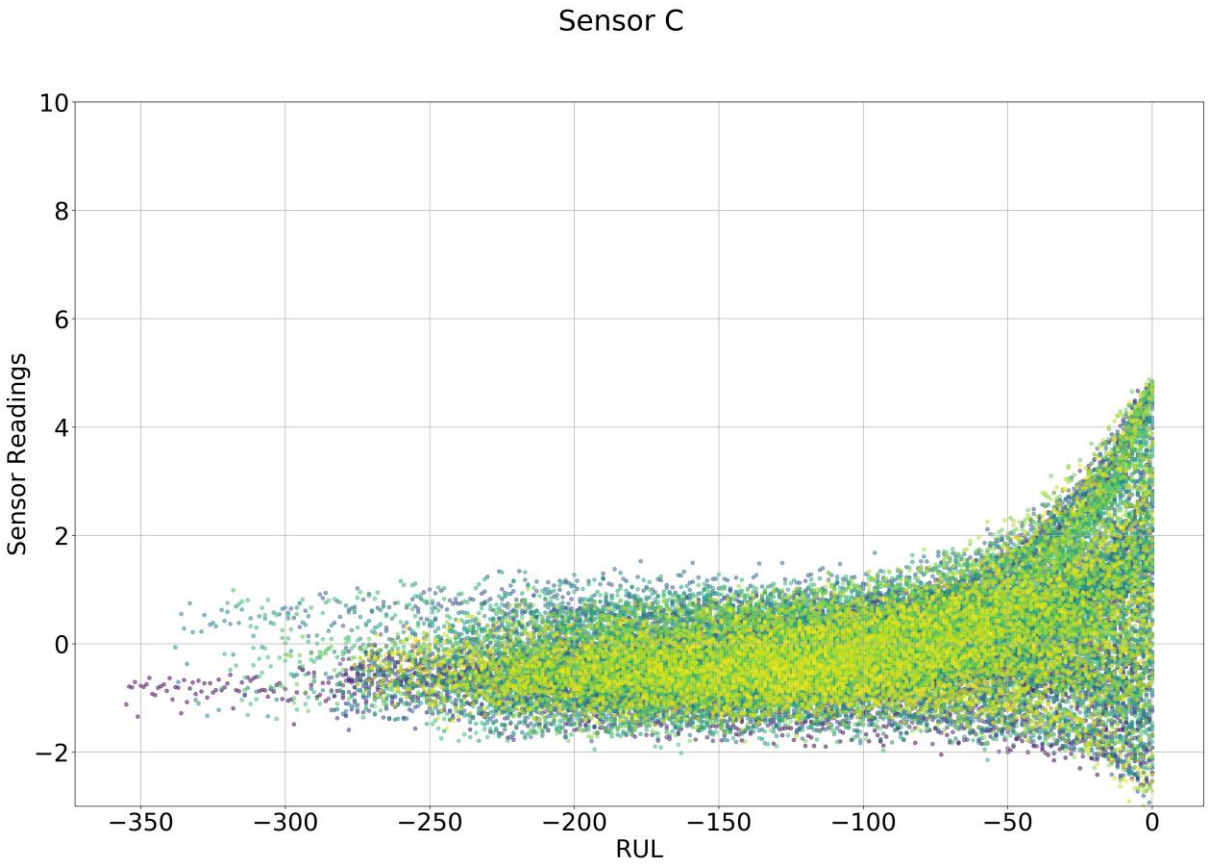

Figura 26 - Sensor C após redução de dimensionalidade

Interessante, os grupos de sensores A e C apresentaram um comportamento semelhante ao anterior, é visível que as tendências de perturbação permanecem muito parecidas a medida que o eixo horizontal se aproxima de 0. Entretanto o grupo de sensores B mudou bastante, a curva cujo antes da redução de dimensionalidade tendia para baixo a medida que o RUL se aproximava de 0 agora tende para cima, isso mostra que haviam mais informações ocultas nos sensores do que se é capaz de perceber apenas através das visualizações.

4.2.6 Transformação Logarítmica

Neste ponto, os sensores foram reduzidos a três: A, B e C, de forma que todos parecem apresentar fortes padrões à medida que os dados se aproximam da condição de falha (eixo horizontal = 0). Entretanto a distribuição dos dados ainda está relativamente "barulhenta" para seguir para a próxima etapa: *feature extraction*. Por isso, uma técnica de "suavização" dos dados adequada para este caso é implementada, a transformação logarítmica.

Em geral, a transformação logarítmica é utilizada para tornar distribuições altamente ruidosas menos ruidosas. Existem duas condições importantes cujo devem ser atendidas para justificar

a utilização correta desta técnica, e podem ser conferidas no tópico 2.8 deste trabalho. No caso da distribuição de dados em questão, as condições são rigorosamente atendidas.

Abaixo pode-se conferir a visualização dos três sensores após a realização da transformação logarítmica. É importante observar que os limites verticais dos gráficos abaixo são exatamente os mesmo aos exibidos pelas figuras 24, 25 e 26, tornando mais visível a percepção dos resultados.

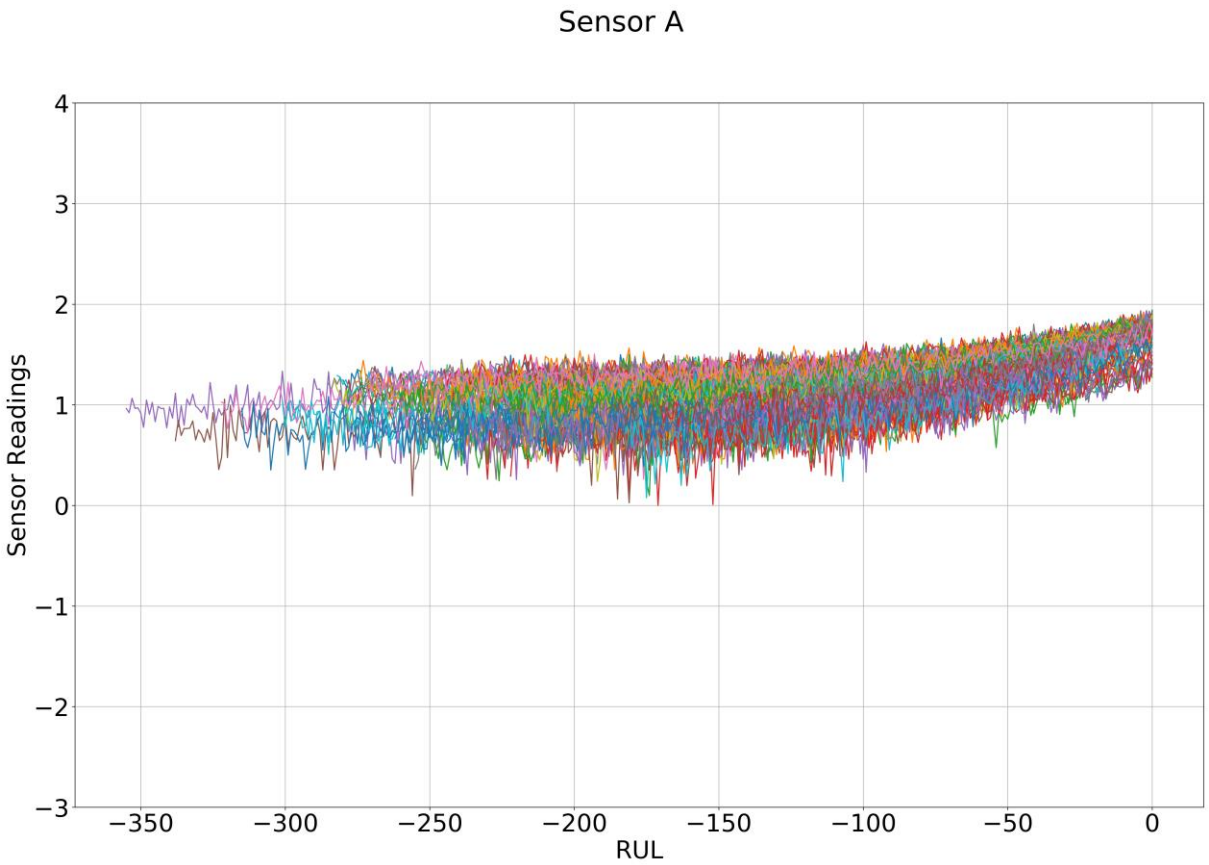

Figura 27 - Sensor A após transformação logarítmica

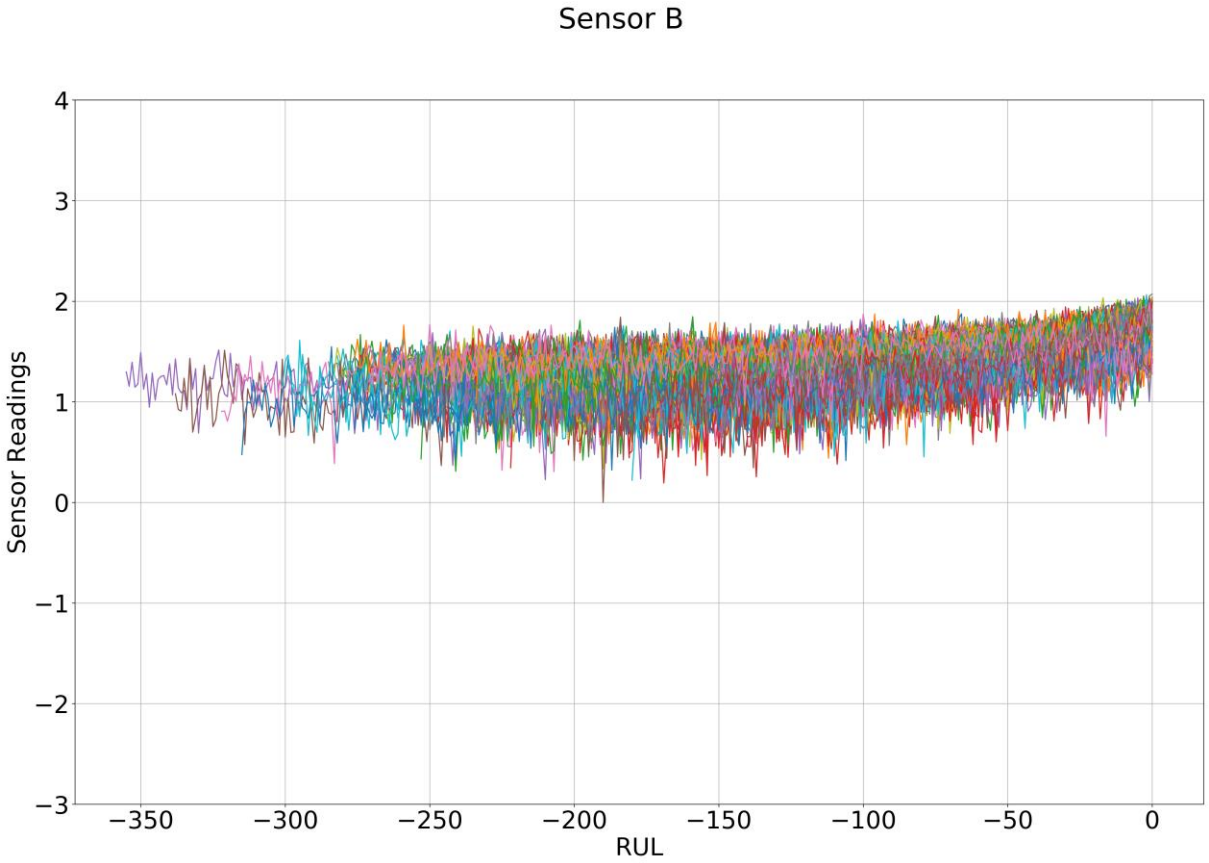

Figura 28 - Sensor B após transformação logarítmica

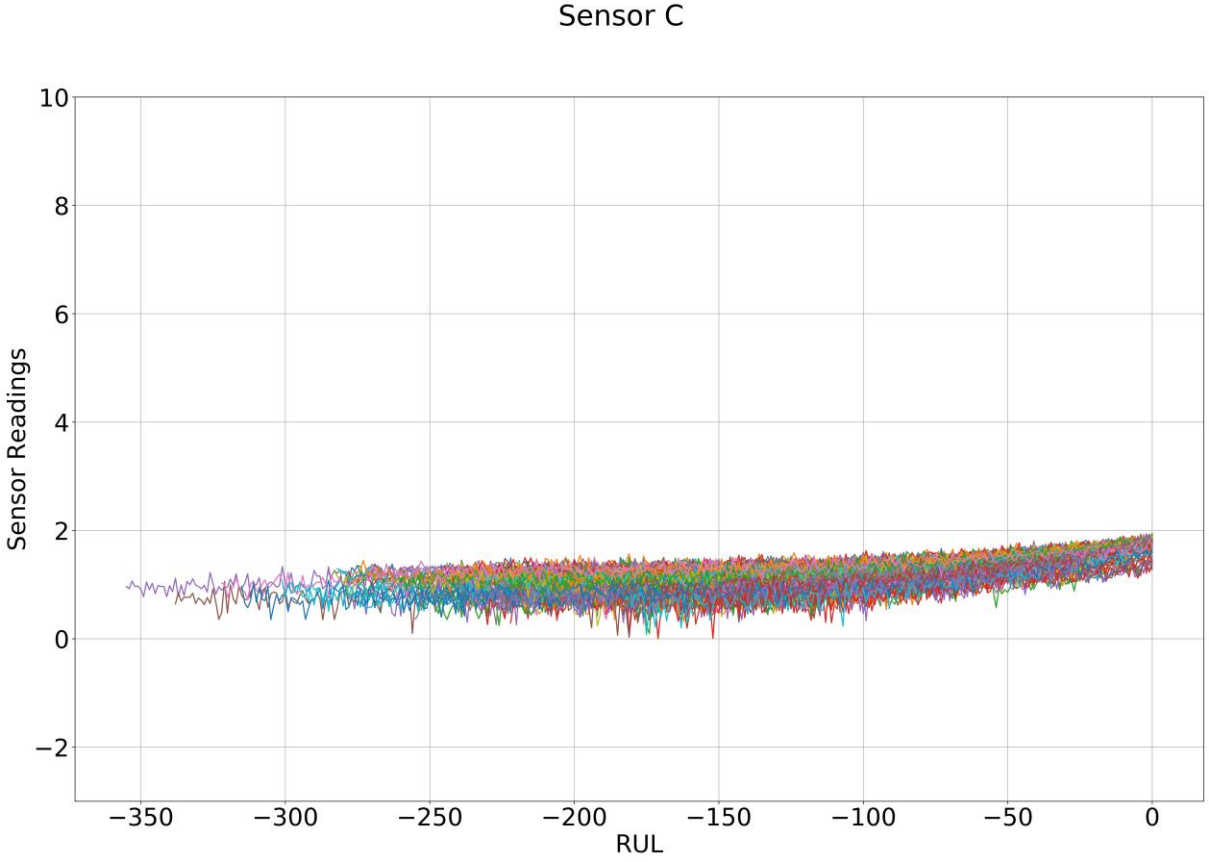

Figura 29 - Sensor C após transformação logarítmica

Excelente, é perceptível que os dados se tornaram menos ruidosos após esta etapa. Neste ponto do trabalho, os dados estão suficientemente prontos para o último passo: *feature extraction*. Entretanto esta etapa será realizada na parte 2, pois o objetivo é extrair as *features* do *artificial training set*.

Antes de prosseguir, é importante lembrar as estatísticas que foram extraídas do *original training set* até este ponto:

1. Centroides dos *clusters* de condições de operação
2. Dados dos sensores "crus" separados por *clusters*
3. As matrizes de transformação geradas pelo algoritmo PCA para a sintetização dos sensores unificados A, B e C.

4.3 *Artificial Training Set*

Como já mencionado, o *test set* é constituído por dados truncados, cujo são descritos por testes nos quais as trajetórias dos sensores não atingem o fim de sua vida útil. Por isso, a confecção de testes "artificiais" é criada a partir dos testes "originais" do training set. Estes novos testes serão "cortados" em momentos estrategicamente definidos durante a vida útil de um teste. O objetivo é tornar os dados nos quais os modelos preditivos serão treinados mais parecidos com os dados nos quais estes serão avaliados.

Os testes serão cortados em momentos aleatórios, em períodos pré-estabelecidos, com base na porcentagem do tempo de operação de cada teste. Os intervalos pré-estabelecidos são:

- 80% - 100%
- 60% - 80%
- 40% - 60%
- 20% - 40%

O objetivo é cobrir todos os intervalos de tempo de tempos de operação dos testes do *test set*. Como o menor teste (com menos ciclos operacionais) do *test set* possui aproximadamente 20% do tempo operacional do menor teste do *training set*, os intervalos são definidos da maneira descrita a cima com o objetivo de cobrir todos os intervalos de tempo de operação dos testes do *test set*.

Pode-se entender melhor como os novos testes são criados a partir dos originais observando a figura abaixo.

Figura 30 - Exemplo de criação de testes artificiais a partir de testes originais do *training set*

Considerando que para cada teste são criados quarto novos testes artificiais, o novo training set, agora chamado de artificial training set, é constituído por 218 * 4 = 1090 testes. Desta maneira, além de os dados para o treinamento dos modelos preditivos serem mais parecidos com os dados no qual os modelos serão testados, possui-se também cinco vezes mais dados que o original, facilitando o trabalho dos algoritmos de regressão. É importante ressaltar, que o rótulo dos testes do *original training set* são sempre 0, tendo em vista que atingem a condição de falha. Entretanto os testes artificiais possuem seus respectivos rótulos exatamente iguais a quantidade de ciclos operacionais que lhes foram retirados.

Após a criação do nosso novo conjunto de dados, todos as etapas realizadas nos tópicos 4.2.2 a 4.2.6 são executadas novamente na mesma ordem, porém utilizando as estatísticas extraídas do original training set como base de cálculos quando necessário. Feito isso, nosso conjunto de dados está pronto para a última etapa do processamento de dados.

4.3.1 *Feature Extraction and Selection*

Neste ponto do trabalho tem-se 1090 curvas que representam os testes do *artificial training set*. Tais curvas como observou-se até este ponto, são resultados de um intenso processamento de dados. O rótulo de cada curva é o seu respectivo RUL, ou seja, para cada curva X, tem-se anexado a mesma o seu rótulo correspondente Y.

Os modelos preditivos baseados em regressão são treinados de maneira a entender que para um determinado X (curva), esta possui seu Y (rótulo) correspondente. Entretanto uma técnica de crucial importância para a eficiência deste "treinamento" é realizada: *feature extraction*. *Feature extraction,* ou rudemente traduzida para extração de recursos, consiste na "tradução", "conversão" e/ou "representação" de um conjunto de dados em características e propriedade extraídas destes de maneira a representa-los de maneira mais "interpretável" para os algoritmos. No caso deste trabalho, é praticamente impossível treinar modelos preditivos sem a realização desta etapa, pois os testes possuem tamanhos (quantidade de ciclos operacionais) diferentes. É importante que as *features*, cujo agora representam os dados dos testes, sejam as mesmas e estejam organizadas na mesma ordem para todos os testes.

Para a realização desta etapa, utilizou-se uma ferramenta desenvolvida recentemente (2016) para a linguagem de programação Python, chamada "tsfresh".

Segundo a própria definição da empresa desenvolvedora da ferramenta: "tsfresh é um pacote disponível para o Python. Este pacote calcula automaticamente um grande número de características de séries temporais, os chamados *features*. Além disso, o pacote contém métodos para avaliar o poder explicativo e a importância de tais características para tarefas de regressão ou classificação".

Em outras palavras, este pacote oferece uma maneira de, através de poucas linhas de código, traduzir os dados, cujo são sequencias temporais separadas por testes. Este também seleciona os recursos a serem extraídos de forma que apenas sejam calculadas apenas as *features* altamente relacionadas com variável *target*: o rótulo (RUL).

Para entender como este procedimento é realizado, a imagem abaixo resume uma passagem da estrutura de dados que será processada.

```
In [158]:
1 print(df_log.shape)
2 df_log[210:225]
```

```
(151672, 5)
```

Out[158]:

	unit	sensor_a	sensor_b	sensor_c	time
210	1	1.799095	1.622845	1.303958	-7
211	1	1.749212	1.862630	1.334270	-6
212	1	1.830144	1.566030	1.356571	-5
213	1	1.815952	1.602227	1.254192	-4
214	1	1.832062	1.686856	1.455061	-3
215	1	1.793654	1.667007	1.373121	-2
216	1	1.788823	1.918250	1.736545	-1
217	1	1.729667	1.905519	1.675885	0
218	2	0.561578	0.831363	0.837566	-158
219	2	0.290943	1.055592	1.289266	-157
220	2	0.673623	1.345844	1.392860	-156
221	2	0.761037	1.359169	1.345048	-155
222	2	0.392930	0.919922	1.045771	-154
223	2	0.533605	1.216733	1.223463	-153
224	2	0.684692	0.543274	1.304868	-152

Figura 31 - Exemplo da estrutura final de dados do *artificial training set* antes da etapa de *feature extraction*

"df_log" armazena o conjunto de dados processados até este ponto do trabalho. Pode-se observar que os dados possuem 151.672 observações, cujo são descritos pelo número de linhas. Existem também 5 colunas:

- Coluna 1 (unit): Descreve qual o teste que a observação pertence
- Colunas 2, 3 e 4 (sensores): As informações dos sensores devidamente tratados
- Coluna 5 (time): representa o domínio do tempo de cada teste

As observações deste conjunto de dados estão distribuídas em 1090 testes. É importante pontuar que existe um vetor-coluna descrito pela variável "label" no qual possui 1090 valores, cujo descrevem o RUL de cada teste.

Exposto o acima, o processo de extração e seleção das *features* acontece da seguinte maneira:

```
In [148]:
from tsfresh import extract_relevant_features
features_filtered_direct = extract_relevant_features(df_log, labels,
                                                     column_id='unit', column_sort='time')

Feature Extraction: 100%|██████████| 10/10 [13:32<00:00, 67.54s/it]
WARNING:tsfresh.feature_selection.relevance:Infered regression as machine learning task

In [149]:
print('features shape: ', features_filtered_direct.shape)
print('labels shape: ', labels.shape)
hide()

features shape:  (1090, 1113)
labels shape:  (1090,)
```

Figura 32 - Código de extração automática de *features*

Pode-se observar que, como "tsfresh" é um pacote especializado em extrair *features* de séries temporais, é importante especificar com clareza não só o conjunto de dados no qual se deseja extrair os recursos ("df_log"), mas também a coluna que representa os diferentes testes ("unit") e a que representa o domínio do tempo ("time"). Um parâmetro extra utilizado neste caso para realizar a extração e seleção simultaneamente é a variável *target* ("label").

Apesar de simples, a execução deste código requer bastante processamento computacional, observa-se que o processo dividido em 10 partes leva em média 13 minutos e 32 segundos por parte, totalizando um tempo de execução de mais de duas horas. Este código foi executado pelo notebook Dell Vostro V14T-5470-A50.

Pode-se perceber que a ferramenta extraiu 1113 *features* de cada um dos 1090 testes. Todas as *features* aptas a serem calculadas podem ser conferidas no campo de referências bibliográficas deste trabalho.

Após isto, os dados estão prontos para o treinamento dos modelos preditivos.

4.3.2 Treinamento dos Modelos Preditivos

Como todos os rótulos do conjunto de dados a serem inseridos nos modelos são conhecidos, um procedimento muito comum em trabalhos de *machine learning* é utilizado: a separação do conjunto de dados de treinamento no que se chama de *training* e *validation set.*

No caso deste trabalho, 80% dos dados serão atribuídos ao *training set* e 20% ao *validation set*. Desta maneira pode-se treinar os modelos preditivos na grande maioria dos dados e testá-los nos dados restante. O objetivo é testar a eficiência dos algoritmos quando submetidos às previsões no próprio conjunto de dados para treinamento.

Como mencionado no tópico 4.1, os modelos preditivos são treinados nos seguintes algoritmos de regressão:

1. *Extreme Gradient Boosting*
2. *Decision Tree*
3. *Support Vector Machine (Radial Kernel)*
4. *Random Forest*

A seguir pode-se observar através dos gráficos de pontos as diferenças entre os valores reais (pontos em azul) e os valores previstos (pontos em vermelho)

Extreme Gradient Boosting

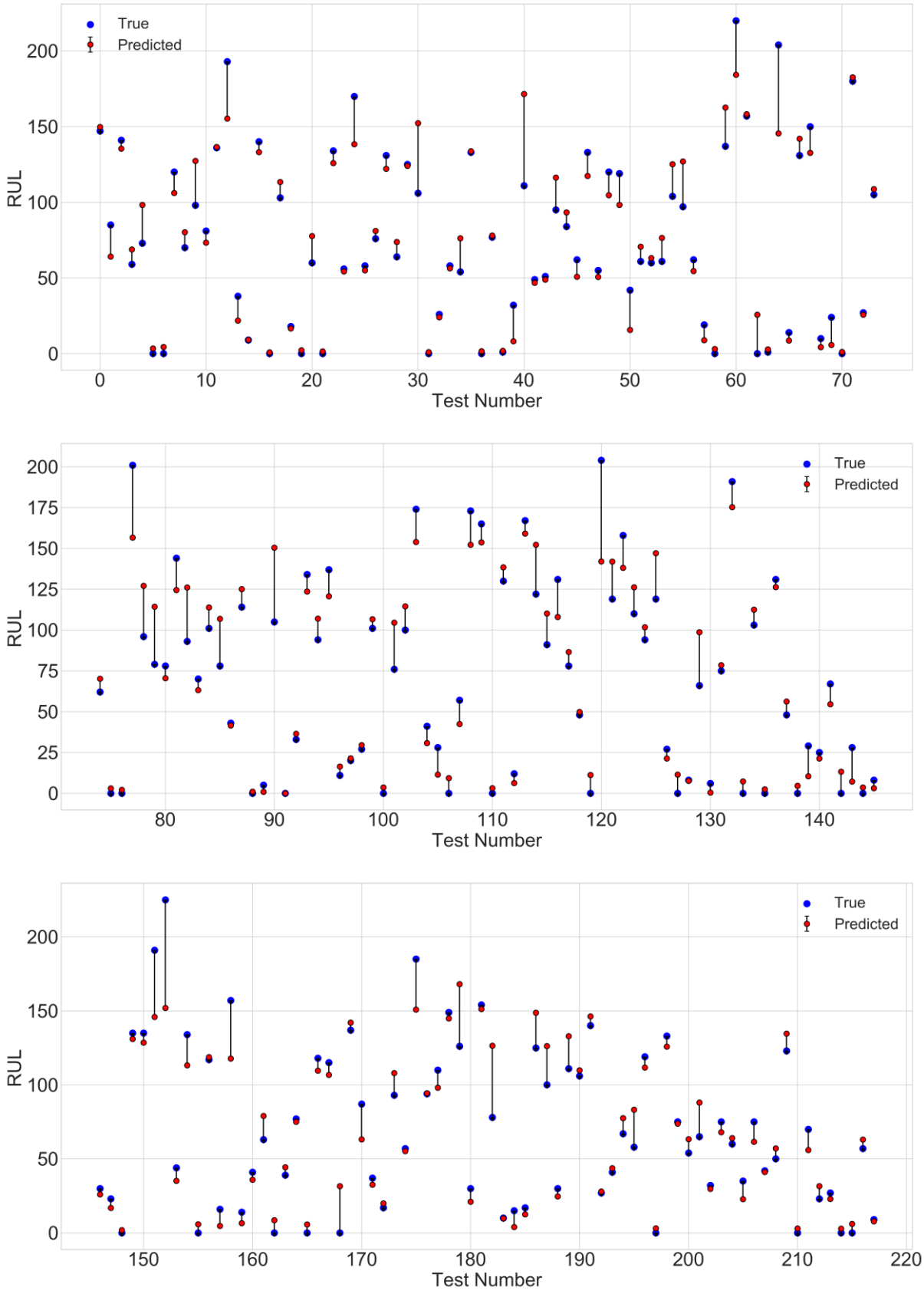

Figura 33 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *Extreme Gradient Boosting*

Decision Tree

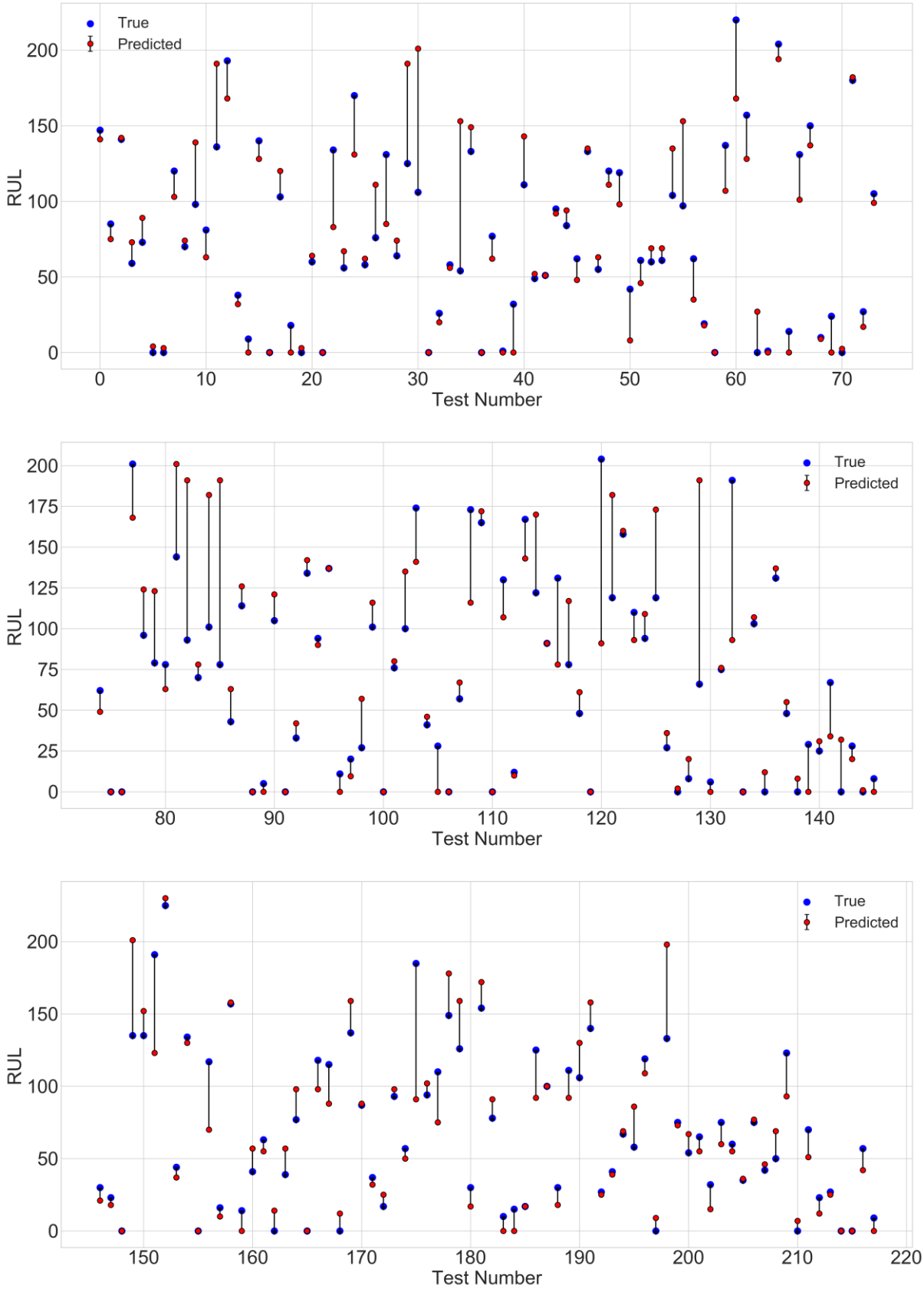

Figura 34 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *Decision Tree*

SVM (Support Vector Machine) – Radial Kernel

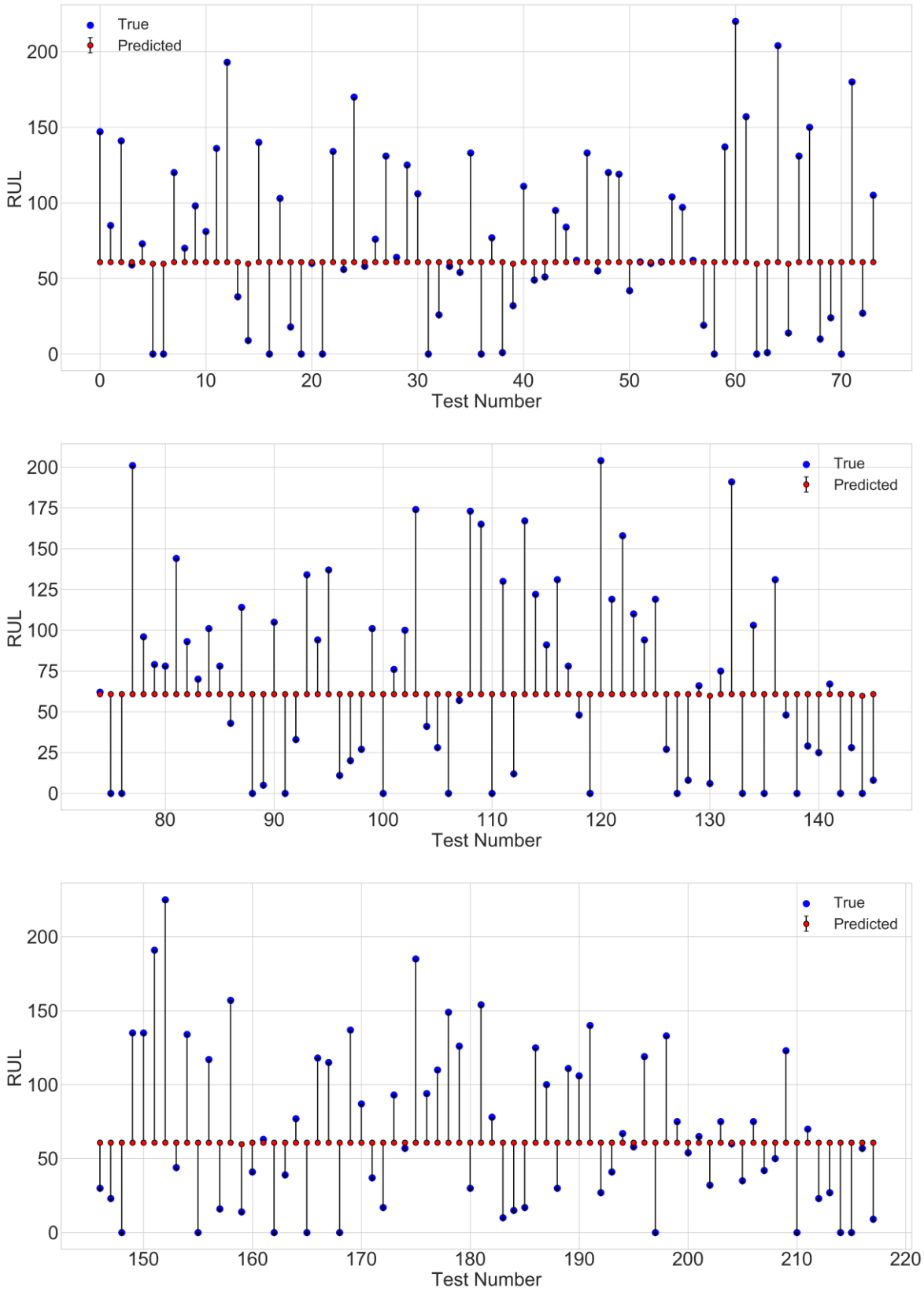

Figura 35 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *SVM*

Random Forest

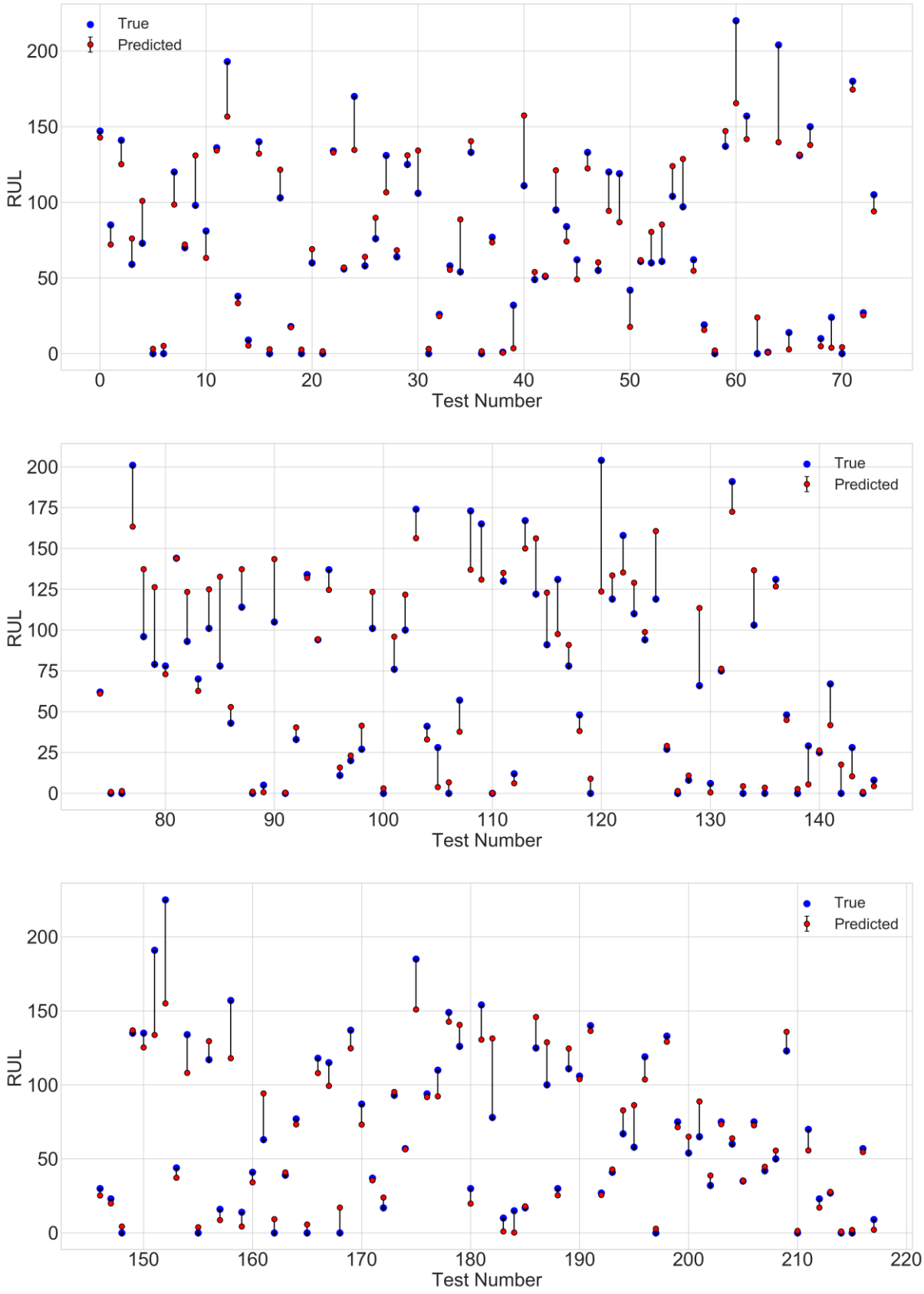

Figura 36 - Gráfico de pontos valores reais vs. previstos. Algoritmo: *Random Forest*

5. RESULTADOS E DISCUSSÕES

Observando superficialmente os gráficos exibidos no tópico 4.3.2, pode-se inferir algumas informações:

- O algoritmo *Support Vector Machine* não funcionou corretamente. Aparentemente ao realizar todas as previsões idênticas, este considerou que a maneira mais eficiente para isto foi utilizar a média dos RULs dos dados de treinamento inseridos em seu modelo. Realmente as aplicações do Support Vector Machine geralmente são mais voltadas para classificações. Considera-se que a aplicação deste algoritmo foi falha.
- O algoritmo Decision Tree realizou previsões personalizadas para cada teste, considera-se então, que sua aplicação superou o SVM para este caso. Entretanto é possível observar que a distância entre valores os valores reais e previstos é consideravelmente maior que as observadas nos algoritmos *Extreme Gradient Boosting* e *Random Forest*.
- Os algoritmos Extreme *Gradient Boosting* e *Random Forest* aparentemente obtiveram os melhores desempenhos. Entretanto é impossível quantificar numericamente qual o melhor apenas com os gráficos de pontos exibidos.
- É possível observar também que os testes cujo possuem RULs menores foram previstos de forma mais precisa. É razoável entender o motivo deste comportamento pois sabe-se que estes são mais semelhantes aos testes do *original training set*, tendo em vista que lhes foram retirados poucos ciclos operacionais. Neste caso suas *features* traduzem as informações mais precisamente que um teste cujo sua vida útil foi cortada em 80%, por exemplo.

Para avaliar e comparar de forma precisa o desempenho dos modelos treinados, o melhor método certamente é utilizar a própria métrica de avaliação evidenciada na descrição original dos dados. Uma função que utiliza a equação 6 descrita no capítulo 3.2 foi confeccionada para isso:

```python
def evaluation(predicted_rul, true_rul):
    #a1 = 13
    #a2 = 10
    negative = 0
    for i in range(0, len(predicted_rul)):
        if (predicted_rul[i] < 0):
            predicted_rul[i] = 0
            negative = negative + 1
    d = predicted_rul - true_rul
    pred = pd.DataFrame(predicted_rul)
    true = pd.DataFrame(true_rul)
    result = pd.concat([true, pred], axis=1)
    result.columns = (['true', 'pred'])
    result['diff'] = result['true'] - result['pred']
    score = 0
    for i in range(0, len(d)):
        if (d[i]<0):
            x = np.exp(-(d[i]/13))-1
        else:
            x = np.exp(d[i]/10)-1
        score = score + x
    score_per_test = score/len(predicted_rul)
    print('Negative predictions: ', negative)
    print('Score: ', score)
    print('AVG Score per test: ', score_per_test)
    return(result)
```

Figura 37 - Python função de avaliação de desempenho

É importante ressaltar que esta função conta a quantidade de previsões negativas que fora realizada, entretanto para o cálculo da pontuação, esta considera as previsões negativas como 0, pois não existe RUL negativo.

Seguem os resultados:

Extreme Gradient Boosting

```
Amount of predicted values:  (218)
Negative predictions:  1
Score:  1994.471324620693
AVG Score per test:  9.148951030370151
```

Decision Tree

```
Amount of predicted values:  (218)
Negative predictions:  0
Score:  418150.8883642604
AVG Score per test:  1918.1233411204605
```

Support Vector Machine

```
Amount of predicted values:  (218)
Negative predictions:  0
Score:  836859.1518818449
AVG Score per test:  3838.8034489992883
```

Random Forest

```
Amount of predicted values:  (218)
Negative predictions:  0
Score:  2650.6762174544174
AVG Score per test:  12.159065217680814
```

De fato, pode-se perceber que os algoritmos *Decision Tree* e *Support Vector Machine* obtiveram pontuações catastróficas, talvez realmente não seja a mais aplicação utiliza-los em casos de regressão. Os algoritmos *Extreme Gradient Boosting* e *Random Forest* obtiveram pontuações razoáveis, certamente o suficiente para afirmar que o método funciona.

5.1 Pontuação Final

Como mencionado no capítulo 4.1, a última etapa a ser realizada é a execução das previsões para o *test set*. Para isso, outro modelo baseado no algoritmo cujo obteve o melhor desempenho (*Extreme Gradient Boosting*) foi treinado utilizando desta vez 100% dos dados obtidos após toda a fase de processamento do artificial training set.

Após o treinamento do modelo, o *test set* é submetido a uma fase de processamento de dados exatamente idêntica à que fora submetido o *artificial training set*. As previsões para os 218 testes foram realizadas e submetidas a avaliação no site oficial da competição (NASA.gov).

Uma pontuação de 2448,02 foi obtida e resume o desempenho de todo o trabalho realizado. Esta pontuação classificaria o trabalho um pouco abaixo do vigésimo colocado, como exibido pela tabela abaixo.

N°	Score
1	436,841
2	512,426
3	737,769
4	809,757
5	908,588
6	975,586
7	1049,57
8	1051,88
9	1075,16
10	1083,91
11	1127,95
12	1139,83
13	1219,61
14	1263,02
15	1557,61
16	1808,75
17	1966,38
18	2065,47
19	2399,88
20	2430,42
Vitor	**2448,02**

Tabela 3 - 20 melhores pontuações PHM08 Data Challenge + pontuação final do trabalho

Apesar a competição na qual os dados originalmente pertencem ter sido realizada a um tempo consideravelmente grande. Considera-se que, pelo pela escala e nível da mesma, o trabalho foi bem-sucedido. O resultado final mostra que certamente a parte de pré-processamento de dados e/ou a seleção dos algoritmos utilizados não foi a melhor possível. Entretanto um método original foi implementado e o objetivo do trabalho, cujo se resume em realizar uma análise comparativa de desempenho de algoritmos de regressão foi concluída com sucesso.

6 CONSIDERAÇÕES FINAIS

Ao longo deste trabalho, diversas técnicas foram utilizadas com o mesmo objetivo: melhorar o desempenho das previsões. Muitas foram malsucedidas, algumas funcionaram bem. Destaca-se para trabalhos futuros a implementação de algum algoritmo baseado em redes neurais recorrentes. Acredita-se que a implementação de um algoritmo capaz de prever o comportamento dos sensores pode ser a ideia chave para alcançar melhores resultados,

principalmente com relação aos testes cujo apresentam tempos de vida útil remanescente maiores, pois estes foram os que os algoritmos obtiveram maior dificuldades em realizar as previsões. Acredita-se fielmente que a implementação correta de um algoritmo a nível *deep learning* para a finalidade pontuada acima, possa levar a eficiência do trabalho para outro patamar. Uma dica seria tentar implementar o algoritmo LSTM (*Long Short-Term Memory*).

Um pacote robusto da linguagem de programação Python, também fora desenvolvido recentemente e tem sido extremamente bem referenciado na comunidade de cientistas de dados, seu nome é FeatureTools. Este pacote tem como objetivo extrair e selecionar recursos de certos conjuntos de dados de forma extremamente eficiente. Os parâmetros de suas funções permitem selecionar de forma semiautomática as *features* a serem extraídas baseadas nas estruturas dos dados nos objetivos em questão. Caso o tempo e a limitação de processamento computacional não sejam um problema, talvez seja interessante reconsiderar a etapa de redução de dimensionalidade dos sensores. Técnicas que foram amplamente testadas e que não obtiveram bons resultados são os procedimentos de filtragem baseado em suavização de curvas. Foram testados: IIR-Filter (*Infinite Impulse Response*), Savitzky-Golay-Filter e regressão polinomial.

REFERÊNCIAS BIBLIOGRÁFICAS

ROUSSEEUW, Peter J. **Silhouettes: a graphical aid to the interpretation and validation of cluster analysis**. Journal of Computational and Applied Mathematics, North-Holland, p. 53-65, 13 jun. 1986.

KARDEK, A; NASCIF, J. & BARONI, T**. Gestão da Manutenção e Técnicas Preditivas.** Rio de Janeiro: Ed. Qualitymark: ABRAMAN, 2002.

LORE IO INC. **What is Data Standardization?**. [*S. l.*], 2019. Disponível em: https://www.getlore.io/knowledgecenter/data-standardization#:~:targetText=Data%20Standardization%20is%20a%20data,it‘s%20loaded%20into%20target%20systems. Acesso em: 1 dez. 2019.

RAFFAELI, Carlos. **#1 Introdução: A Quarta Revolução Industrial**. [*S. l.*], 5 jan. 2016. Disponível em: https://medium.com/@carlosraffaeli/1-introdu%C3%A7%C3%A3o-a-quarta-revolu%C3%A7%C3%A3o-industrial-a42058aba533. Acesso em: 24 nov. 2019.

WIKIPEDIA, THE FREE ENCYCLOPEDIA. **Unsupervised learning**. [*S. l.*], 1 dez. 2019. Disponível em: https://en.wikipedia.org/wiki/Unsupervised_learning#:~:targetText=Unsupervised%20learning%20is%20a%20type,probability%20densities%20of%20given%20inputs. Acesso em: 25 nov. 2019.

PRESS, Gil. **A Very Short History Of Data Science**. [*S. l.*], 28 maio 2013. Disponível em: https://www.forbes.com/sites/gilpress/2013/05/28/a-very-short-history-of-data-science/#2af4ab255cfc. Acesso em: 23 nov. 2019.

WARD, Rachel. **Some mathematics for k-means clustering**. K-means clustering, Berlin, 2015. Disponível em: https://www3.math.tu-berlin.de/numerik/csa2015/Slides/Ward.pdf. Acesso em: 25 nov. 2019.

LAKSHMANAN, Swetha. **Outlier Detection and Treatment: A Beginner's Guide**. [*S. l.*], 8 maio 2019. Disponível em: https://medium.com/@swethalakshmanan14/outlier-detection-and-treatment-a-beginners-guide-c44af0699754. Acesso em: 2 dez. 2019.

Harnad, Stevan (2008), "**The Annotation Game: On Turing (1950) on Computing, Machinery, and Intelligence**", in Epstein, Robert; Peters, Grace (eds.), The Turing Test Sourcebook: Philosophical and Methodological Issues in the Quest for the Thinking Computer, Kluwer.
Stuart J. Russell, Peter Norvig (2010) **Artificial Intelligence: A Modern Approach**, Third Edition, Prentice Hall ISBN 9780136042594.

M.R. Smith and T. Martinez (2011). "**Improving Classification Accuracy by Identifying and Removing Instances that Should Be Misclassified**". Proceedings of International Joint Conference on Neural Networks (IJCNN 2011). pp. 2690–2697. CiteSeerX 10.1.1.221.1371. doi:10.1109/IJCNN.2011.6033571.

Hastie, Trevor; Tibshirani, Robert (2009). **The Elements of Statistical Learning: Data mining, Inference, and Prediction**. New York: Springer. pp. 485–586. doi:10.1007/978-0-387-84858-7_14. ISBN 978-0-387-84857-0.

Duda, Richard O.; Hart, Peter E.; Stork, David G. (2001). "**Unsupervised Learning and Clustering**". Pattern classification (2nd ed.). Wiley. ISBN 0-471-05669-3.

Roweis, S. T.; Saul, L. K. (2000). "**Nonlinear Dimensionality Reduction by Locally Linear Embedding**". Science. 290 (5500): 2323–2326. Bibcode:2000Sci...290.2323R. CiteSeerX 10.1.1.111.3313. doi:10.1126/science.290.5500.2323. PMID 11125150.

Rico-Sulayes, Antonio (2017). "**Reducing Vector Space Dimensionality in Automatic Classification for Authorship Attribution**". Revista Ingeniería Electrónica, Automática y Comunicaciones. 38 (3): 26–35.

WIKIPEDIA, THE FREE ENCYCLOPEDIA. **Data transformation (statistics)**. [*S. l.*], 30 nov. 2019. Disponível em: https://en.wikipedia.org/wiki/Data_transformation_(statistics). Acesso em: 29 nov. 2019.

ZUMEL, Nina; MOUNT, John. **Log Transformations for Skewed and Wide Distributions**. [*S. l.*], 27 maio 2013. Disponível em: https://www.r-statistics.com/2013/05/log-transformations-for-skewed-and-wide-distributions-from-practical-data-science-with-r/. Acesso em: 27 nov. 2019.

CHRIST, Maximilian. **Tsfresh**. [*S. l.*], 2016. Disponível em: https://tsfresh.readthedocs.io/en/latest/. Acesso em: 5 nov. 2019.

WIKIPEDIA, THE FREE ENCYCLOPEDIA. **Feature engineering**. [*S. l.*], 2 dez. 2019. Disponível em: https://en.wikipedia.org/wiki/Feature_engineering#:~:targetText=Feature%20engineering%20is%20the%20process,is%20both%20difficult%20and%20expensive. Acesso em: 5 dez. 2019.