UFOP - Universidade Federal de Ouro Preto

Decom - Departamento de Ciência da Computação

Modelos Básicos para IPMSMs

Autor: Bianca Barreto Leme

Matrícula: 24.1.4008

Professor: Rodrigo César Pedrosa Silva

Ouro Preto - MG
11 de setembro de 2025

Sumário

1 Introdução

Os Motores Síncronos de Ímã Interno Permanente (IPMSMs) têm sido amplamente utilizados por serem uma alternativa menos agressiva ao meio ambiente, quando comparados aos motores de carros comuns.

Em fase de testes (Finite Element Analysis), os motores devem ser submetidos a várias condições de velocidade e torque. Fazer estes testes fisicamente é custoso e pode levar dias. Por essa razão, utilizar "motores virtuais" e prever suas perdas através de modelos de IA pode ser muito mais viável, pois este método não tem grandes custos e levam por volta de algumas horas.

Estudando IPMSMs podemos criar modelos de inteligência artificial que nos auxiliem a predizer as principais causas de perdas em ferro de determinado motor: perda por histerese e perda por eddy current.

Nesse contexto, este estudo tem como objetivo principal encontrar o melhor modelo de IA possível para predizer as perdas em histerese e eddy current de 3 motores IPMSMs analisados: 2D, Nabla e V.

2 Fundamentos

Nesta seção do trabalho, serão descritos os métodos, algoritmos, procedimentos e métricas utilizadas para atingir o objetivo.

2.1 Bases de dados

As bases de dados consistem em estados de cada parâmetro dos motores em diferentes condições, e se dividem em duas partes: a primeira destinada para o treinamento dos modelos (train) e a segunda para a testagem de suas acurácias (test).

Destes datasets, os parâmetros se categorizam em outros dois grupos: Features (X) – variáveis preditoras ou independentes – e Targets (y) – variáveis alvo ou dependentes.

- Features:
  - Variáveis geométricas (Xgeom);
  - Velocidade do motor (N);
  - Corrente no eixo direto (Id);
  - Corrente no eixo em quadratura (Iq);
- Targets:
  - Perda por histerese (hysteresis);
  - Perda por eddy current (joule).

Em resumo, a base de dados se separa em 4:

- X train: features para treino;
- y train: targets para treino;
- X test: features para teste;
- y test: targets para teste.

Esses termos serão utilizados ao longo do artigo para melhor comunicação.

2.2 Motores analisados

Neste trabalho, foram analisados 3 IPMSMs distintos:

- 2D

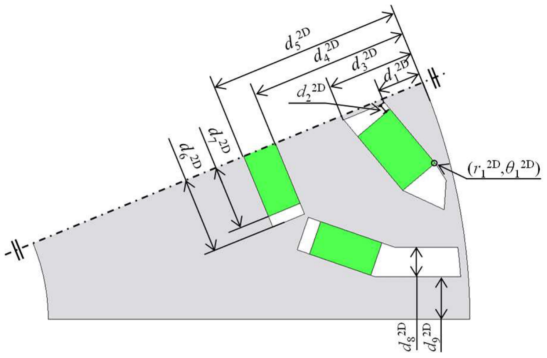


Figura 1: Diagrama do motor 2D.

- Nabla

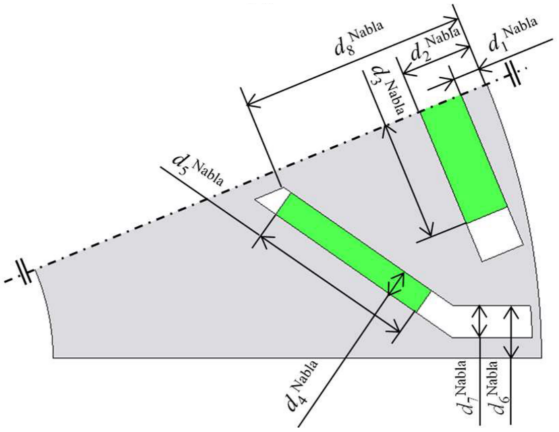


Figura 2: Diagrama do motor Nabla.

- V

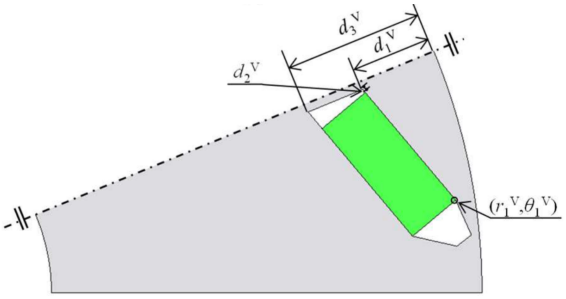


Figura 3: Diagrama do motor V.

2.3 Métricas de Avaliação de Modelos de Regressão

Para a análise da eficácia de cada modelo na predição dos atributos, foram definidas três métricas...

2.3.1 Mean Absolute Percentage Error (MAPE)

O *Mean Absolute Percentage Error* (MAPE) mede o erro percentual médio entre os valores reais y_i e os valores previstos $\hat{y}_i$:

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

2.3.2 Mean Squared Error (MSE)

O *Mean Squared Error* (MSE) mede o erro quadrático médio, penalizando mais fortemente desvios grandes:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

2.3.3 Coeficiente de Determinação (R^2)

O coeficiente de determinação, ou R^2, mede a proporção da variância dos dados reais que é explicada pelo modelo:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

onde $\bar{y}$ é a média dos valores reais.

2.4 Modelos Utilizados

Nesta seção, serão apresentados os diferentes modelos de aprendizado de máquina utilizados para a predição.

2.4.1 Regressão Linear

A regressão linear busca modelar a relação entre variáveis de entrada $x_1, x_2, \ldots, x_p$ e uma variável de saída y, assumindo que essa relação é aproximadamente linear. O modelo é definido por:

$$\hat{y} = \beta_0 + \sum_{i=1}^{p} \beta_i x_i,$$

onde β_0 é o intercepto e os coeficientes β_i representam o peso de cada variável preditora.

Os parâmetros $\boldsymbol{\beta}$ são estimados pelo método dos mínimos quadrados ordinários (MQO), que minimiza o erro quadrático médio:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{j=1}^{n} \left(y_j - \beta_0 - \sum_{i=1}^{p} \beta_i x_{ij} \right)^2.$$

2.4.2 Regression Trees

Uma árvore de regressão divide o espaço das variáveis preditoras em regiões disjuntas $R_1, R_2, \ldots, R_M$. O modelo pode ser escrito como:

$$\hat{y}(x) = \sum_{m=1}^{M} c_m \cdot \mathbb{1}(x \in R_m),$$

onde $\mathbb{1}(\cdot)$ é a função indicadora, que vale 1 se x pertence à região R_m e 0 caso contrário. O valor c_m é geralmente a média dos valores de saída y dos pontos de treino naquela região.

2.4.3 Random Forests

O modelo de Random Forests combina B árvores de regressão, cada uma construída a partir de amostras bootstrap dos dados de treino e subconjuntos aleatórios das variáveis. A previsão é a média das árvores:

$$\hat{y}(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x),$$

onde $T_b(x)$ representa a previsão da b-ésima árvore.

2.4.4 XGBoost

O XGBoost é baseado em *gradient boosting*, que combina árvores de forma aditiva para corrigir erros sucessivos. O modelo após K iterações é:

$$\hat{y}^{(K)}(x) = \sum_{k=1}^{K} f_k(x), \quad f_k \in \mathcal{F},$$

onde cada f_k é uma árvore de regressão e $\mathcal{F}$ é o espaço de funções possíveis.

A cada passo, adiciona-se uma nova árvore $f_k(x)$ para minimizar uma função de perda regularizada:

$$\mathcal{L}^{(K)} = \sum_{j=1}^{n} l\big(y_j, \hat{y}^{(K-1)}(x_j) + f_K(x_j)\big) + \Omega(f_K),$$

onde $l(\cdot)$ é a função de perda (ex.: erro quadrático) e $\Omega(f)$ controla a complexidade do modelo.

2.4.5 CatBoost

O CatBoost também segue o princípio do *gradient boosting*, mas introduz técnicas específicas para lidar com variáveis categóricas e reduzir o viés preditivo. Sua formulação geral é semelhante ao XGBoost:

$$\hat{y}^{(K)}(x) = \sum_{k=1}^{K} f_k(x),$$

mas a diferença está no processo de construção das árvores $f_k(x)$, que utiliza estratégias de ordenação para codificação de variáveis categóricas e um esquema de regularização que evita o sobreajuste.

3 Metodologia

Para atingir os objetivos deste trabalho, foram definidos 2 experimentos...

3.1 Experimento 1: Identificar os Modelos mais Promissores

Para esta primeira fase, os hiperparâmetros utilizados foram os já estabelecidos pela linguagem Python, sem nenhuma alteração feita pela pesquisa. Os modelos de aprendizado foram alimentados com as bases de dados destinadas para treino, X train e y train.

Por mais que as IAs com target em hysteresis e as com target em *eddy current* foram treinadas separadamente, para um modelo de predição para perda de histerese foram utilizados as mesmas configurações e hiperparâmetros para a predição de *eddy current*, neste mesmo modelo.

Após essa fase, os dados de X test são fornecidos aos modelos já treinados que retornam, então, os resultados preditos – y pred. A acurácia dos modelos é avaliada pela semelhança entre os valores reais em y test e os valores preditos em y pred. Essa acurácia é refletida nas métricas MSE, MAPE e R^2 score, que foram coletadas ao longo da pesquisa.

Esse processo foi executado com todos os modelos descritos na seção 2.4.

3.2 Experimento 2: Identificar o melhor conjunto de hiper parâmetros para os modelos mais promissores

Após o primeiro experimento, foi observado que os dois modelos mais promissores para predição de perdas são XGBoost e CatBoost. Desse modo, há maior chance de que, quando aplicados métodos de hyperparameter tuning, esses modelos retornem resultados mais satisfatórios. Assim, foram determinados os vetores de hiperparâmetros para cada modelo de aprendizado:

- Para CatBoost: número de iterações, taxa de aprendizado, profundidade das árvores, regularização L2, força aleatória, temperatura do *bagging*, profundidade máxima das árvores, regularização L1 (*reg_ alpha*) e L2 (*reg_ lambda*).
- Para XGBoost: número de árvores (*n_ estimators*), taxa de aprendizado, parâmetro de regularização de divisão (*gamma*), profundidade máxima das árvores, regularização L1 (*reg_ alpha*) e L2 (*reg_ lambda*).

Para o alcançe de melhores resultados com menor gasto de processamento, foi utilizado para ambos os modelos o método Randomized Search, que seleciona combinações aletórias entre os valores dos hiperparâmetros, treinando o modelo com cada uma dessas combinações. Em seguida, cada um desses modelos é avaliado segundo a métrica neg MAE (erro médio absoluto negativo) e o melhor conjunto de hiperparâmetros é utilizado para retreinar o modelo final, que retornará o melhor resultado possível.

Também em ambos modelos, foi empregada a técninca de validação cruzada repetida (RepeatedKFold), a fim de obter uma avaliação mais robusta do desempenho.

4 Resultados

Nesta seção, os resultados para cada motor serão apresentados em 2 formatos: tabelas expondo os valores das métricas e gráficos de comparação entre os valores preditos e os valores reais.

4.1 Métricas

As tabelas a seguir apresentam as métricas de desempenho obtidas para cada motor analisado utilizando diferentes métodos de regressão e duas variáveis *target*: hysteresis e joule. As métricas consideradas foram o coeficiente de determinação (R^2 Score), o erro quadrático médio (MSE) e o erro percentual absoluto médio (MAPE).

4.1.1 Métricas para o Motor 2D

Tabela 1: Métricas para o motor 2D

method	variable	score	mse	mape
linear	hysteresis	0.877371	0.125016	0.914328
	joule	0.725346	0.316331	1.71494
reg_tree	hysteresis	0.969112	0.031489	0.549346
	joule	0.822426	0.20452	0.439176
rand_for	hysteresis	0.978109	0.022318	0.315941
	joule	0.831141	0.194482	0.312343
catboost	hysteresis	0.982187	0.018159	0.120402
	joule	0.834822	0.190243	0.192682
xgboost	hysteresis	0.980641	0.019735	0.191011
	joule	0.833086	0.192242	0.247595
cat_rand	hysteresis	0.981849	0.018504	0.161930
	joule	0.833280	0.192019	0.196973
xgb_rand	hysteresis	0.980901	0.019471	0.178209
	joule	0.833522	0.191739	0.217464

Nos resultados obtidos para o motor 2D, observa-se que os métodos regressão linear, árvore de regressão e *random forests* obtiveram os piores retornos, com altos valores de erro (tanto em MSE quanto MAPE). Entre estes, o modelo de regressão linear apresentou o pior desempenho, com *score* de 0.87 para predição de perda por histerese e 0.72 para a predição de perda por *eddy current*. Os modelos de árvores de regressão e florestas aleatórias, apesar de terem taxas de erros altas, não retornaram scores insatisfatórios.

Para os modelos *CatBoost* e *XGBoost*, os resultados foram consideravelmente positivos, com *scores* para perda por histerese em 0.98 e para perda por *eddy current* em 0.83. Os *scores* obtidos pelo método *random forests* não se manteve muito distante, porém apresentou MAPEs maiores: enquanto *CatBoost* e *XGBoost* tiveram um máximo de 0.24, *random forests* apresentou um mínimo de 0.312.

Em geral, na aplicação do método *Randomized Search* sob os modelos *CatBoost* e *XGBoost*, os valores de R^2 score, MSE e MAPE se mantiveram similares aos anteriores, com uma variação substancial nos resultados.

Importante notar que, enquanto a perda por histerese foi prevista com um *score* entre 0.87 e 0.98, todos os modelos falharam em prever a perda por *eddy current* com um *score* acima de 0.84.

Por fim, as versões híbridas (*cat_ rand* e *xgb_ rand*) mantiveram resultados competitivos, próximos aos melhores modelos, evidenciando estabilidade do desempenho. De forma geral, o CatBoost mostrou ser a abordagem mais adequada para o problema em questão, seguido de perto pelos métodos baseados em *boosting* e *bagging*.

4.1.2 Métricas para o Motor Nabla

Tabela 2: Métricas para o motor Nabla

method	variable	score	mse	mape
linear	hysteresis	0.885856	0.111261	4.135917
	joule	0.851045	0.145358	1.901598
reg_tree	hysteresis	0.970236	0.029012	2.19469
	joule	0.951621	0.04721	0.954882
rand_for	hysteresis	0.988997	0.010725	0.571856
	joule	0.981698	0.01786	0.50522
catboost	hysteresis	0.995904	0.003992	0.828559
	joule	0.991773	0.008028	0.260392
xgboost	hysteresis	0.992065	0.007735	0.73543
	joule	0.985918	0.013741	0.424453
cat_rand	hysteresis	0.995244	0.004636	0.578424
	joule	0.991611	0.008186	0.268995
xgb_rand	hysteresis	0.993701	0.006139	0.891864
	joule	0.987984	0.011726	0.298276

Assim como foi notado para o motor anterior, os métodos regressão linear, árvore de regressão e *random forests* obtiveram os piores retornos, com altos valores de erro (tanto em MSE quanto MAPE). Entre estes, o modelo de regressão linear apresentou o pior desempenho, com *score* de 0.88 para predição de perda por histerese e 0.85 para a predição de perda por *eddy current* e taxas de MAPE exorbitantes: 4.13 para predição de perda por histerese e 1.9 para predição de perda por *eddy current*. Os modelos de árvores de regressão e florestas aleatórias, apesar de terem taxas de erros altas (especialmente para o MAPE na predição de perda por histerese pela árvore de regressão), não retornaram *scores* insatisfatórios.

Para os modelos *CatBoost* e *XGBoost*, os resultados foram consideravelmente positivos, com *scores* para perda por histerese em 0.99 e para perda por *eddy current* em aproximadamente 0.99. Os resultados obtidos pelo método *random forests* não se mantiveram muito distantes, contudo em alguns casos ainda inferiores.

Na aplicação do método *Randomized Search* sob os modelos *CatBoost* e *XGBoost*, apesar de não modificar expressivamente os resultados de *score* e também apresentar melhoras nas taxas de erro, apresentou alguns resultados conflitantes nas taxas de erros: por exemplo, enquanto houve piora em MSE para o *target hysteresis* no método *CatBoost*, houve melhora em MAPE para o mesmo cenário.

Por fim, as versões híbridas (*cat_rand* e *xgb_rand*) mantiveram resultados competitivos, próximos aos melhores modelos. De forma geral, o CatBoost mostrou ser a abordagem mais adequada para o problema em questão, seguido de perto pelos métodos baseados em *boosting* e *bagging*.

4.1.3 Métricas para o Motor V

Tabela 3: Métricas para o motor V

method	variable	score	mse	mape
linear	hysteresis	0.898586	0.103019	1.468986
	joule	0.867551	0.134683	1.692969
reg_tree	hysteresis	0.972356	0.028081	0.687815
	joule	0.972245	0.028223	0.905085
rand_for	hysteresis	0.989882	0.010278	0.57334
	joule	0.989422	0.010756	0.535394
catboost	hysteresis	0.998258	0.00177	0.1907
	joule	0.997135	0.002913	0.318281
xgboost	hysteresis	0.995228	0.004847	0.428569
	joule	0.993864	0.006239	0.448335
cat_rand	hysteresis	0.997204	0.002841	0.263157
	joule	0.996858	0.003195	0.189069
xgb_rand	hysteresis	0.995912	0.004153	0.315962
	joule	0.994688	0.005402	0.331223

Assim como foi notado para o motor anterior, os métodos regressão linear, árvore de regressão e *random forests* obtiveram os piores retornos, com altos valores de erro (tanto em MSE quanto MAPE). Entre estes, o modelo de regressão linear apresentou o pior desempenho, com *score* de 0.89 para predição de perda por histerese e 0.86 para a predição de perda por *eddy current*. Os modelos de árvores de regressão e florestas aleatórias, apesar de terem taxas de erros altas (especialmente para o MAPE na predição de perda por histerese pela árvore de regressão), não retornaram *scores* insatisfatórios.

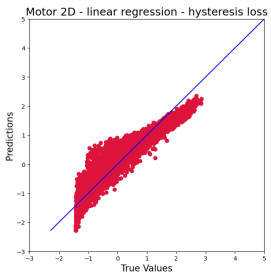
Para os modelos *CatBoost* e *XGBoost*, os resultados foram consideravelmente positivos, com *scores* para perda por histerese em 0.99 e para perda por *eddy current* em aproximadamente 0.99. Os resultados obtidos pelo método *random forests* não se mantiveram muito distantes, contudo obteve taxas de erro maiores.

Na aplicação do método *Randomized Search* sob os modelos *CatBoost* e *XGBoost*, apesar de não modificar expressivamente os resultados de *score*, apresentou alguns resultados conflitantes nas taxas de erros.

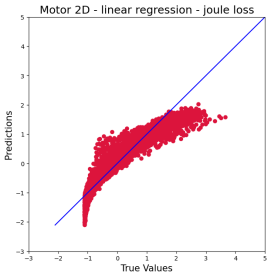
Por fim, as versões híbridas (*cat_rand* e *xgb_rand*) mantiveram resultados competitivos, próximos aos melhores modelos. De forma geral, o CatBoost mostrou ser a abordagem mais adequada para o problema em questão, seguido de perto pelos métodos baseados em *boosting* e *bagging*.

4.2 Gráficos

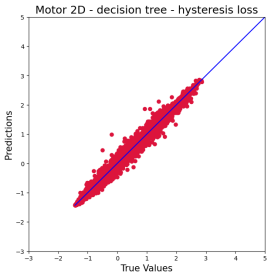
4.2.1 Gráficos para o Motor 2D



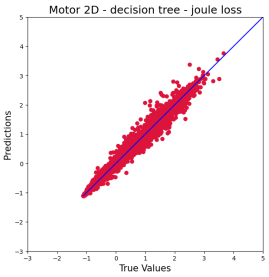
(a) Linear Hysteresis



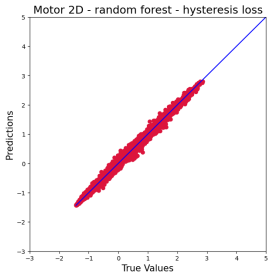
(b) Linear Joule



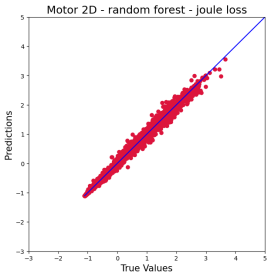
(c) Reg Tree Hysteresis



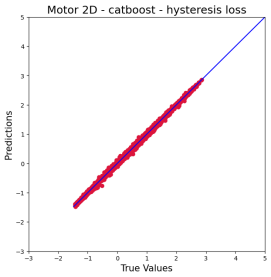
(d) Reg Tree Joule



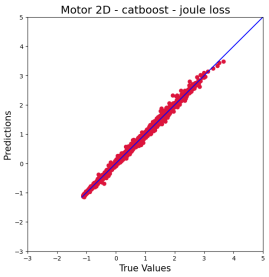
(e) Rand Forests Hysteresis



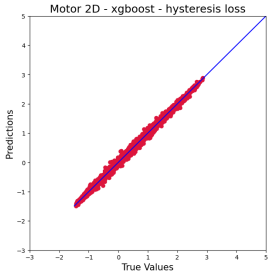
(f) Rand Forests Joule



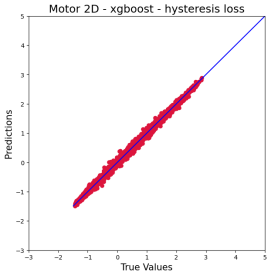
(g) CatBoost Hysteresis



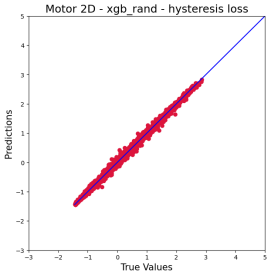
(h) CatBoost Joule



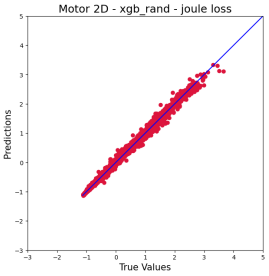
(i) XGBoost Hysteresis



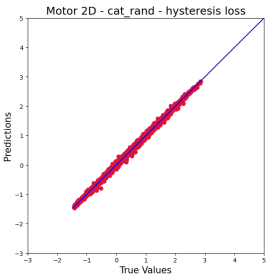
(j) XGBoost Joule



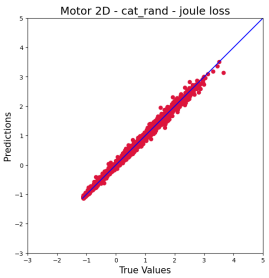
(k) XGB Rand Hysteresis



(l) XGB Rand Joule



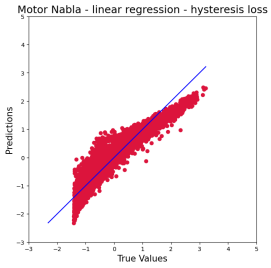
(m) Cat Rand Hysteresis



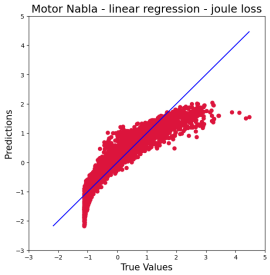
(n) Cat Rand Joule

Figura 4: Comparação entre resultados para o motor 2D.

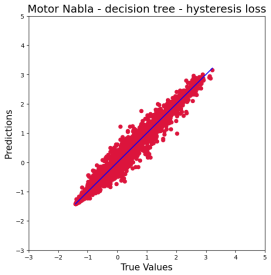
4.2.2 Gráficos para o Motor Nabla



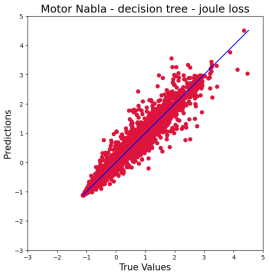
(a) Linear Hysteresis



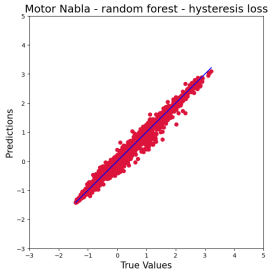
(b) Linear Joule



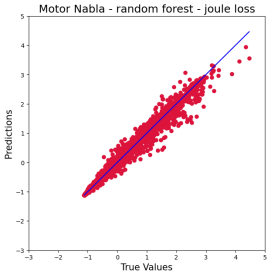
(c) Reg Tree Hysteresis



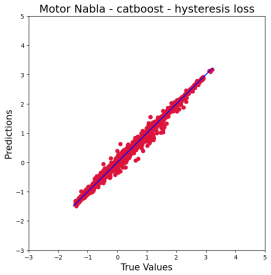
(d) Reg Tree Joule



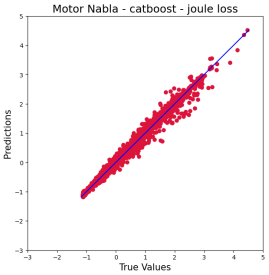
(e) Rand Forests Hysteresis



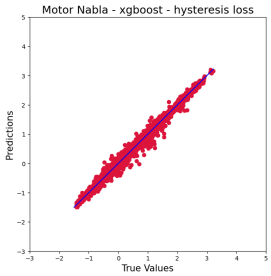
(f) Rand Forests Joule



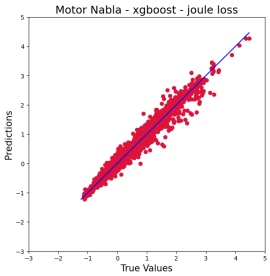
(g) CatBoost Hysteresis



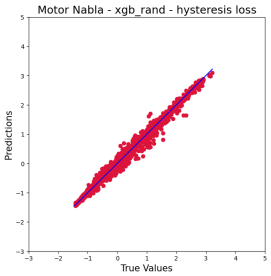
(h) CatBoost Joule



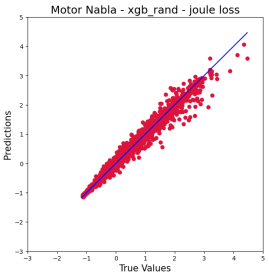
(i) XGBoost Hysteresis



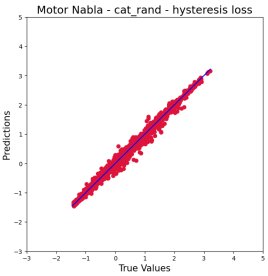
(j) XGBoost Joule



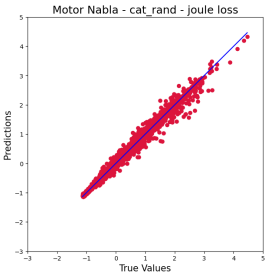
(k) XGB Rand Hysteresis



(l) XGB Rand Joule



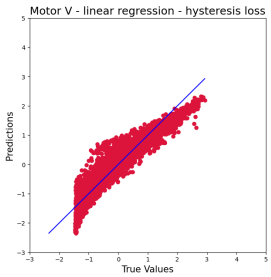
(m) Cat Rand Hysteresis



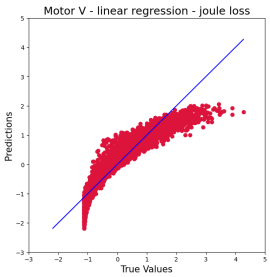
(n) Cat Rand Joule

Figura 5: Comparação entre resultados para o motor Nabla.

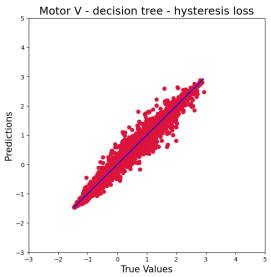
4.2.3 Gráficos para o Motor V



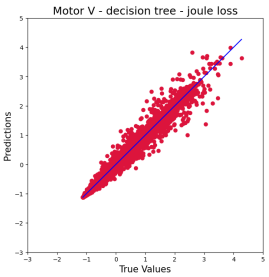
(a) Linear Hysteresis



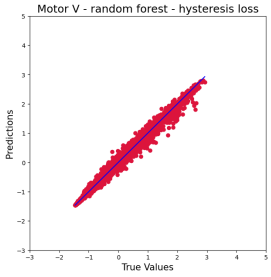
(b) Linear Joule



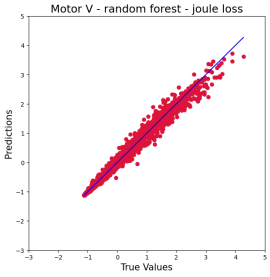
(c) Reg Tree Hysteresis



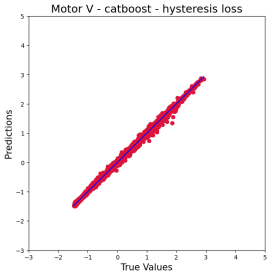
(d) Reg Tree Joule



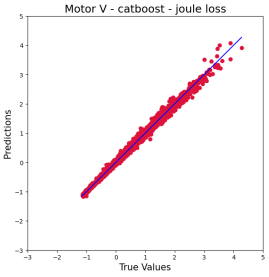
(e) Rand Forests Hysteresis



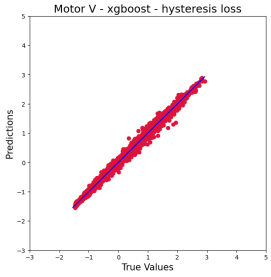
(f) Rand Forests Joule



(g) CatBoost Hysteresis

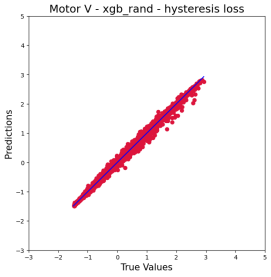


(h) CatBoost Joule

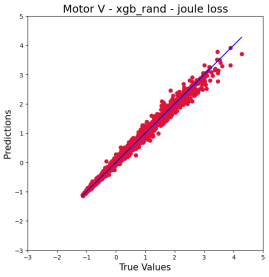


(i) XGBoost Hysteresis

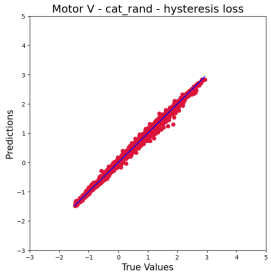
(j) XGBoost Joule



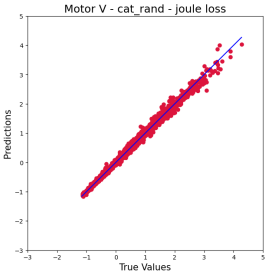
(k) XGB Rand Hysteresis



(l) XGB Rand Joule



(m) Cat Rand Hysteresis



(n) Cat Rand Joule

Figura 6: Comparação entre resultados para o motor V.

4.3 Observações

No geral, a predição da perda por histerese retornou resultados similares ou ligeiramente melhores quando comparados à predição de perda por joule. Isso só não ocorreu no motor 2D, onde todos os modelos tiveram desempenho expressivamente pior na predição de joule.

Como esperado, o modelo de regressão linear provou-se como o menos eficaz para a predição de valores dos *targets*. Isso se deve, muito provavelmente, ao fato de que a regressão linear é um modelo extremamente simples e rudimentar.

Dentre os modelos observados na primeira fase experimental, destacam-se 2: *CatBoost* e *XGBoost*. Contudo, observa-se que os resultados obtidos com *CatBoost* foram ainda mais satisfatórios.

Já na segunda fase experimental, o comportamento da aplicação do método *Randomized Search* foi errático. Em alguns casos, houve uma melhora substancial nos resultados, quando comparados àqueles obtidos com modelos sem ajustes nos hiperparâmetros. Em outros, o comportamento foi contrário, e uma leve piora foi registrada. Isso pode ser explicado pela utilização dos mesmos hiperparâmetros e métodos para modelos em diferentes conjuntos de dados.

Nota-se que o motor 2D obteve os melhores valores para MAPE dentre todos os outros motores. Entretanto, seu coeficiente de determinação e MSE não refletiram valores tão positivos quanto os demais. Na comparação entre valores preditos e observados, é notável que os resultados de 2D também se sobressaíram.

5 Conclusão

Tendo em mãos os resultados inesperados em alguns dos modelos, como valores piores na predição de perda por *eddy current*, vale pontuar a importância de melhor análise e tratamento destes comportamentos em trabalhos futuros.

Em relação à escolha dos modelos de aprendizado, é evidente que *CatBoost* e *XGBoost* são promissores. Para este tipo de problema e de base de dados, o uso dos modelos rudimentares como *Linear Regression*, *Regression Tree* e *Random Forest* não provou gerar resultados positivos.

Além disso, o estudo de hiperparâmetros e melhores configurações pode ser vantajoso para obter modelos mais precisos.

6 Referências