INF1038: Relatório do 1.5
Grupo 2: Mariana Barreto, Thiago Levis, Paulo de Tarso

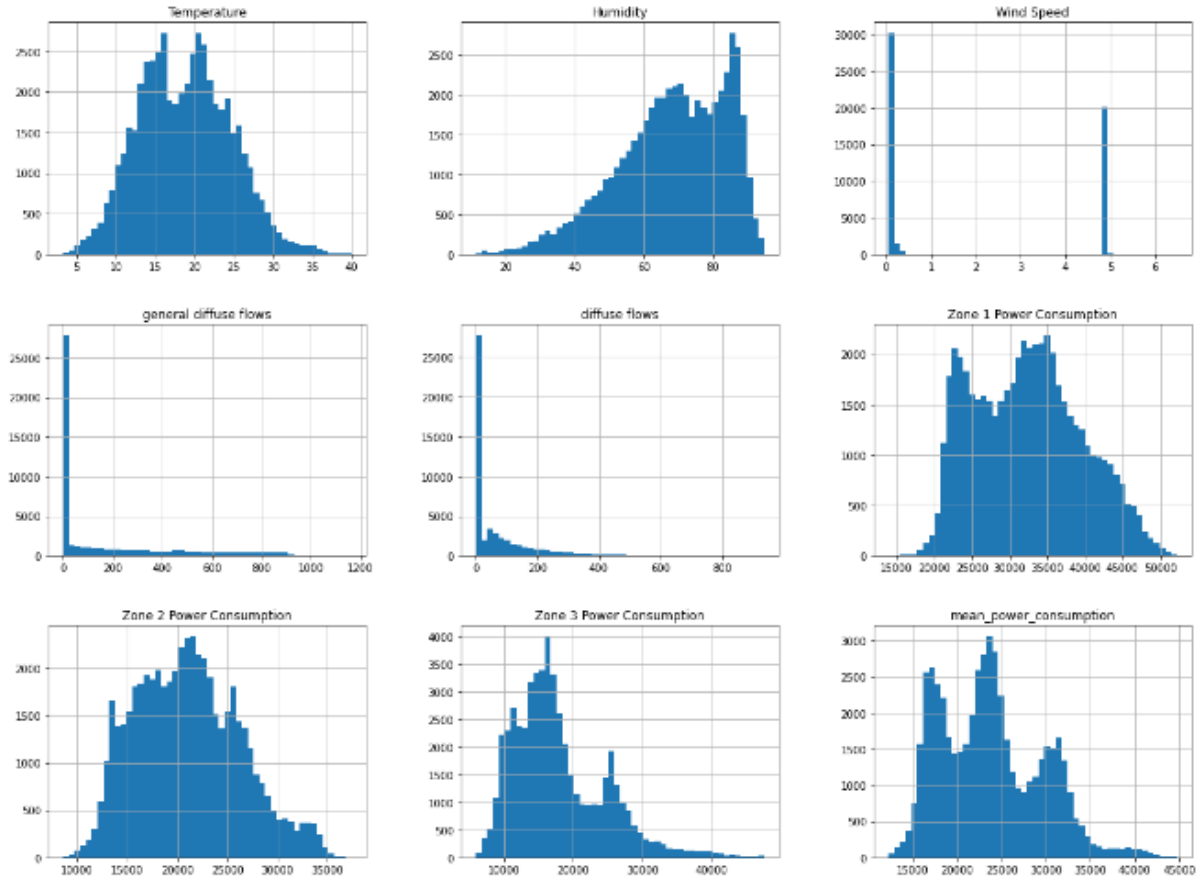
1. Modelo de Regressão (Tetuan City Power Consumption Dataset)

Para a análise de regressão, estamos utilizando um dataset sobre o consumo de energia na cidade de Tetuan[1]. Na tabela abaixo, mostramos todas as colunas com algumas informações sobre cada. Apenas a coluna DateTime que não está incluída. Ela informa o dia e a hora completa em que os outros valores foram registrados e está organizada de 10 em 10 minutos. No total, são 52416 linhas. O objetivo é tentar prever a quantidade de consumo de energia na cidade com base em outras variáveis.

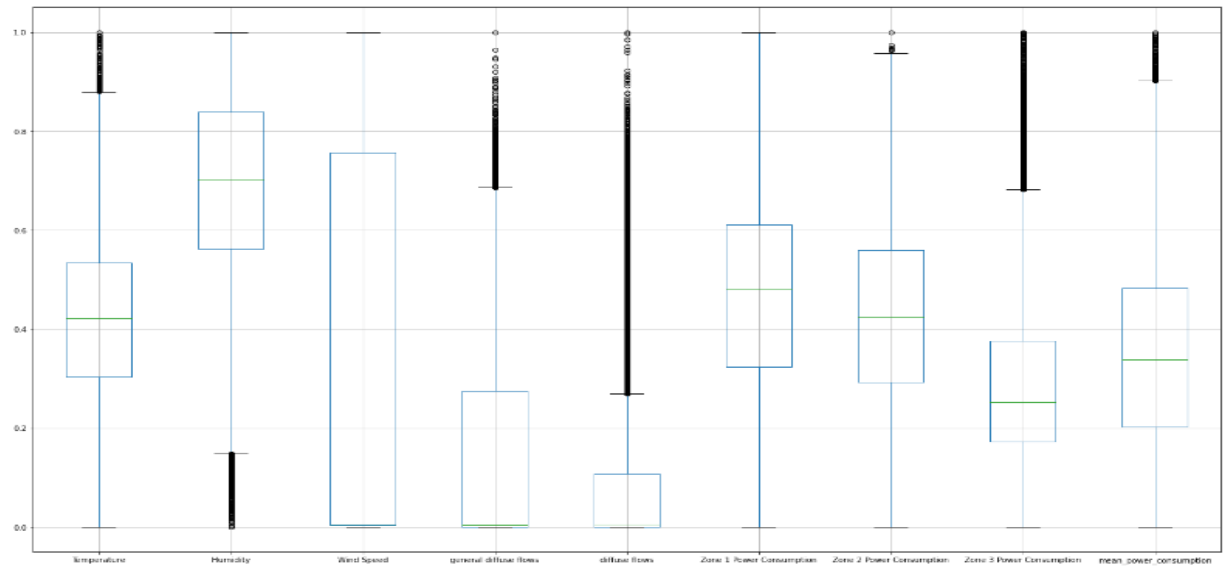
	Temperature	Humidity	Wind Speed	general diffuse flows	diffuse flows	Zone 1 Power Consumption	Zone 2 Power Consumption	Zone 3 Power Consumption
count	52416.000000	52416.000000	52416.000000	52416.000000	52416.000000	52416.000000	52416.000000	52416.000000
mean	18.810024	68.259518	1.959489	182.696614	75.028022	32344.970564	21042.509082	17835.406218
std	5.815476	15.551177	2.348862	264.400960	124.210949	7130.562564	5201.465892	6622.165099
min	3.247000	11.340000	0.050000	0.004000	0.011000	13895.696200	8560.081466	5935.174070
25%	14.410000	58.310000	0.078000	0.062000	0.122000	26310.668692	16980.766032	13129.326630
50%	18.780000	69.860000	0.086000	5.035500	4.456000	32265.920340	20823.168405	16415.117470
75%	22.890000	81.400000	4.915000	319.600000	101.000000	37309.018185	24713.717520	21624.100420
max	40.010000	94.800000	6.483000	1163.000000	936.000000	52204.395120	37408.860760	47598.326360

Aqui podemos observar que não existem dados faltando, ou zerados. Todos os valores também são positivos e não nulos. Não tem valores repetidos. Criamos uma nova coluna “Mean Power Consumption” para juntar os dados de consumo da energia nas três zonas, pois esse será nosso target. Com o histograma abaixo, dá para observar a variação de distribuição dos dados e de escala. Calculando a soma do consumo de energia de cada zona, descobrimos que a Zona 1 é a que mais consome.

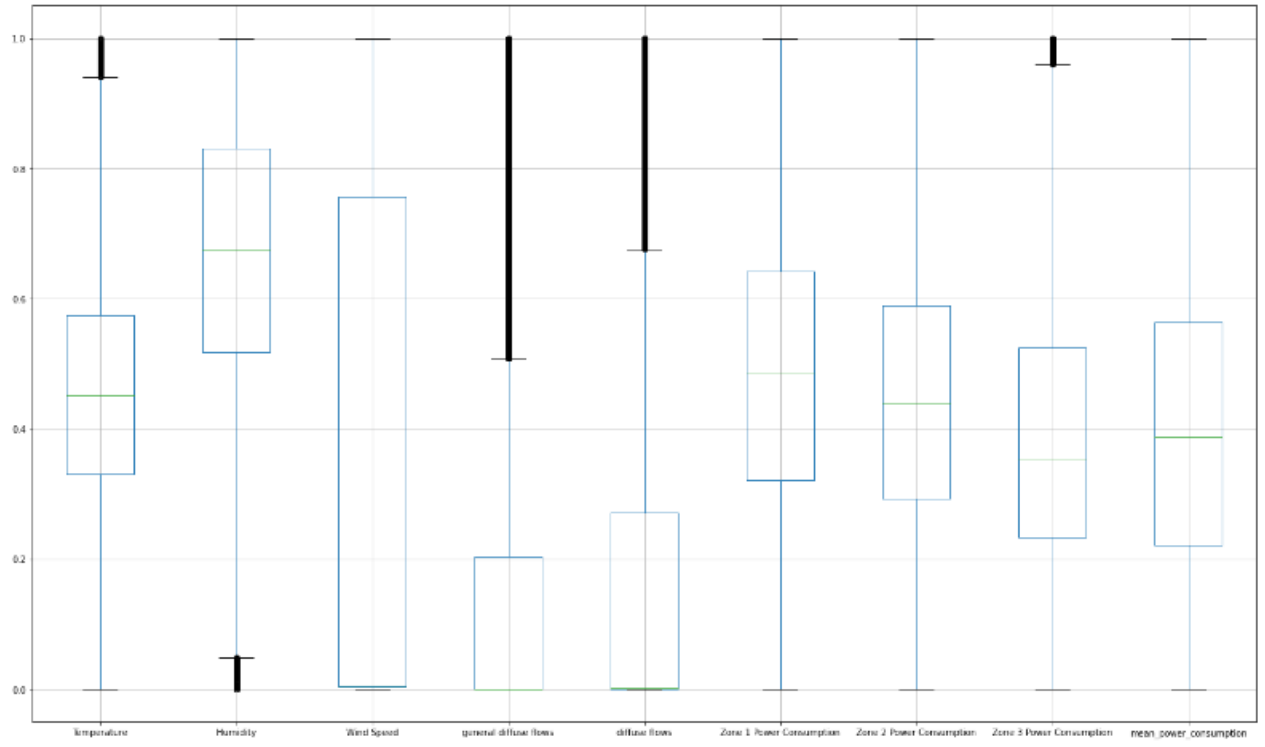
[1] http://archive.ics.uci.edu/ml/datasets/Power+consumption+of+Tetouan+city



Normalizamos os dados para lidar com a variação das escalas e fizemos um boxplot de cada coluna (exceto o de Date Time que removemos do dataframe por ser outro tipo de variável comparado aos outros). Assim como no modelo de regressão do Trabalho 1, existe uma grande quantidade de outliers na maioria das categorias. Isso também acontece no nosso target mean_power_consumption, o que pode afetar os resultados das análises.



Após remover outliers, conforme indicado na figura abaixo, ficamos com um total de 44349 linhas.



Em seguida, fizemos uma matriz de correlação para descobrir a conexão entre as variáveis.

Analisando as correlações, podemos destacar uma previsível forte similaridade entre os consumos de energia da Zona 1, 2 e 3, além do target de consumo de energia média (de 73 a 96%). Também é notável a esperada correlação (porém um tanto mais fraca) entre diffuse flows e general diffuse flows (60%). Das features, removemos apenas as colunas de consumo de energia de cada zona devido a proximidade dos dados.

Para fazer a análise do modelo utilizamos as mesmas métricas do Trabalho 1 como parâmetros principais: Mean squared error, Mean absolute error, Root mean squared error, e R2 score. O Intercept também está sendo usado novamente.

Regressão linear

Para treinamento, utilizamos de novo 70% dos dados e o resto para teste. Ao aplicar regressão linear tivemos os seguintes resultados:

Score = 0.187

MSE = 0.033

MAE = 0.150

RMSE = 0.181

R2 score = 0.187

Intercept = 0.283

Média dos scores de Negative MSE = -0.033

Desvio padrão dos scores de Negative MSE = 0.001

Negative MSE foi calculado usando Cross Validation de valor 7 (divide os dados em 7 subconjuntos, usa um para teste e o resto para calcular a acurácia do modelo). As métricas desse modelo foram em geral fracas, como pode ser observado pela figura abaixo. Em comparação com o Trabalho 1, os resultados ficaram piores, especialmente o R2 Score que era 0.614.

Lasso

Na sua aplicação, usando CV de valor 7, o score deu -0.127, o que significa que o Lasso Regression não foi muito eficiente.

Ridge

Com o Ridge Regression (ou regularização de Tikhonov) conseguimos:
lr mean score: 0.083
rg mean score: 0.086
rg alpha: 0.00025

O valor do CV foi 7 também. Os valores aqui pelo menos não estão negativos, mas ainda assim certamente o modelo não deu muito certo.

ElasticNet

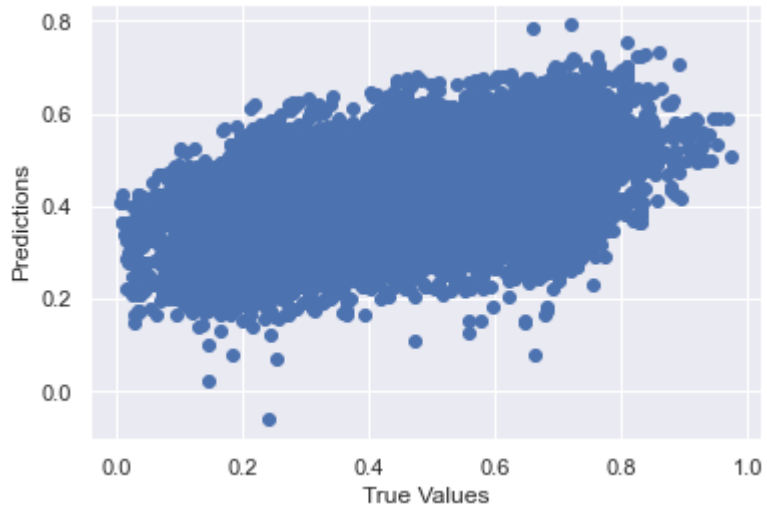
Os resultados nesse caso foram:

ElasticNet mean score: -0.151
encv.alpha_: 0.001
encv.l1_ratio: 0.1

O valor do CV foi 10. Exatamente como no Trabalho 1, o score do ElasticNet está mais próximo é parecido com o do Lasso, o que deve indicar maior influência do que o Ridge.

Polinomial 2 Grau

Ao aplicar uma polinomial de grau 2, obtivemos os seguintes resultados:
score: 0.23
MSE: 0.031
MAE: 0.142
RMSE: 0.175
R2 Score: 0.23



No dataset do Trabalho 1, o modelo de polinomial de grau 2 tinha dado os melhores resultados, mas nesse não está fazendo muita diferença.

Polinomial 4 Grau

Ao utilizar grau 4, chegamos a
score: 0.29
MSE: 0.029
MAE: 0.137
RMSE: 0.17
R2 Score: 0.29

Ao observar as informações acima podemos observar que os valores estão um pouco melhores, mas não o suficiente para afirmarmos que esse modelo é eficiente pros dados. Aumentando o grau do polinomial, o score também aumenta e o MSE, MAE e RMSE diminuem.

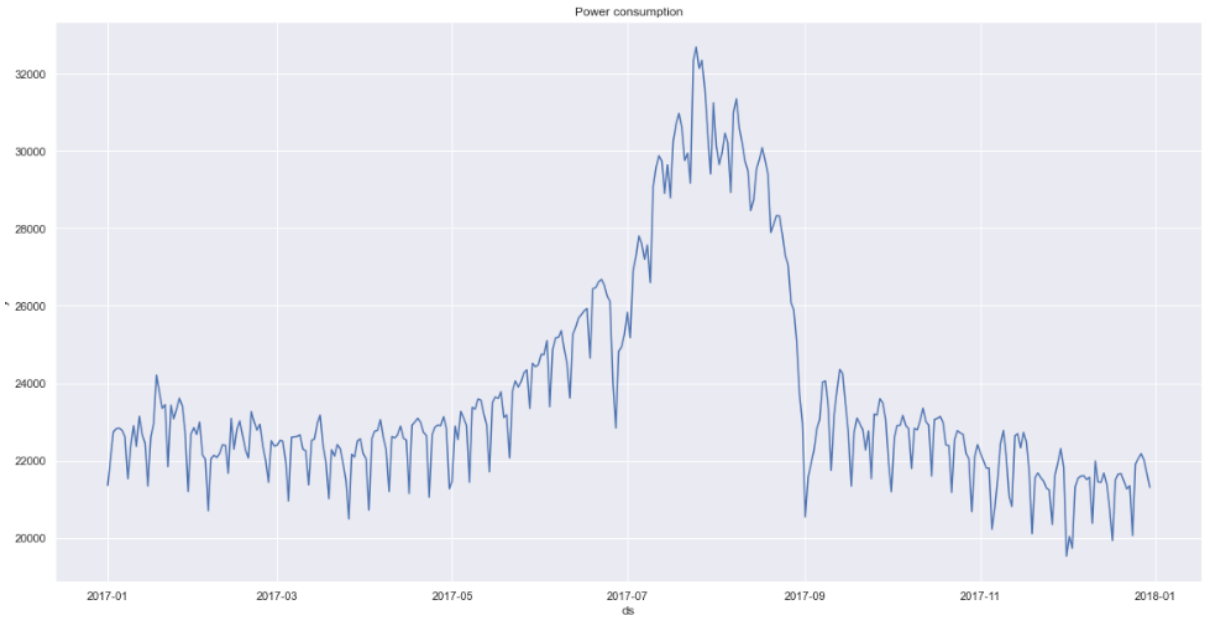
Polinomial 9 Grau
score: 0.39
MSE: 0.025
MAE: 0.120
RMSE: 0.157
R2 Score: 0.39

Com grau 9, podemos notar pela figura e pelos valores melhores que os dados estão mais próximos e organizados, o modelo está ficando melhor. Não testamos com grau maior devido a demora de execução. Provavelmente deve ter um grau

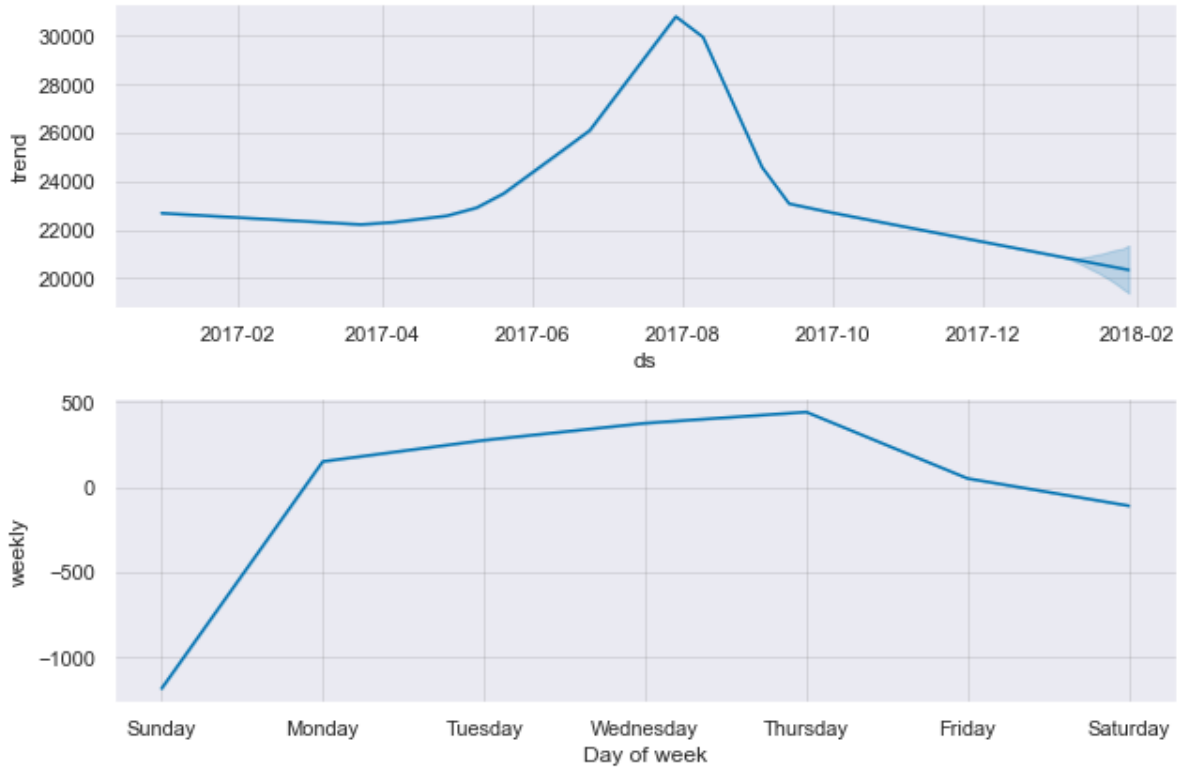
maior no qual atinge uma boa porcentagem no score, outro grau maior que cause overfitting e aí o valor começa a decair.

Time Series

Uma diferença em relação ao Trabalho 1 é que decidimos implementar uma série temporal dessa vez e conferir os resultados, especialmente considerando que os outros modelos se saíram pior nesse dataset. Esse modelo se baseia na predição de valores futuros considerando os previamente observados. Utilizando a biblioteca open-source Prophet, do Facebook, construímos o modelo usando as colunas DateTime e Zone 1 Power Consumption, que será o target. Fizemos testes com os valores das outras zonas e os resultados não ficaram muito diferentes. Somando os valores de consumo de energia de cada dia, ficamos com uma tabela com uma coluna das datas e outra do consumo. Com isso, criamos um gráfico para observar o padrão temporal.



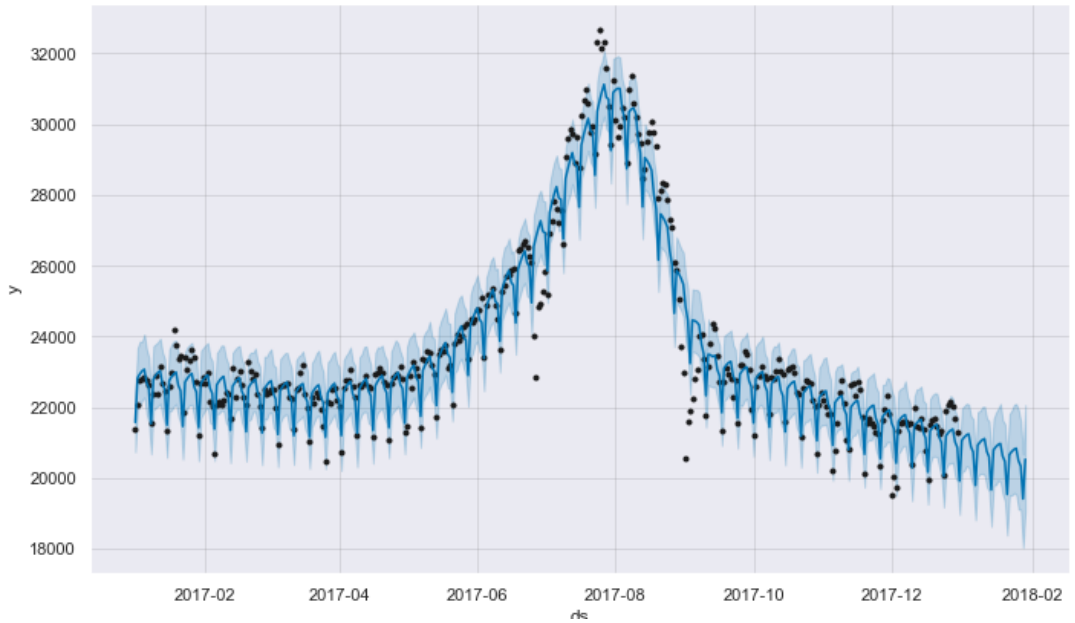
Podemos observar um pico no consumo entre os meses de julho e setembro. Usando as funções do Prophet, estendemos os dados de consumo prevendo os valores dos próximos 30 dias. Ou seja, estamos incluindo dados de janeiro de 2018. O forecast nos mostra o valor previsto e uma margem inferior e outra superior. Depois, fizemos o plot para ver como esses valores ficam posicionados em relação ao gráfico anterior. Também pudemos observar o padrão de consumo de energia mensalmente e semanalmente de forma simplificada. Basta observar na primeira figura abaixo como o formato do gráfico é próximo ao de cima.



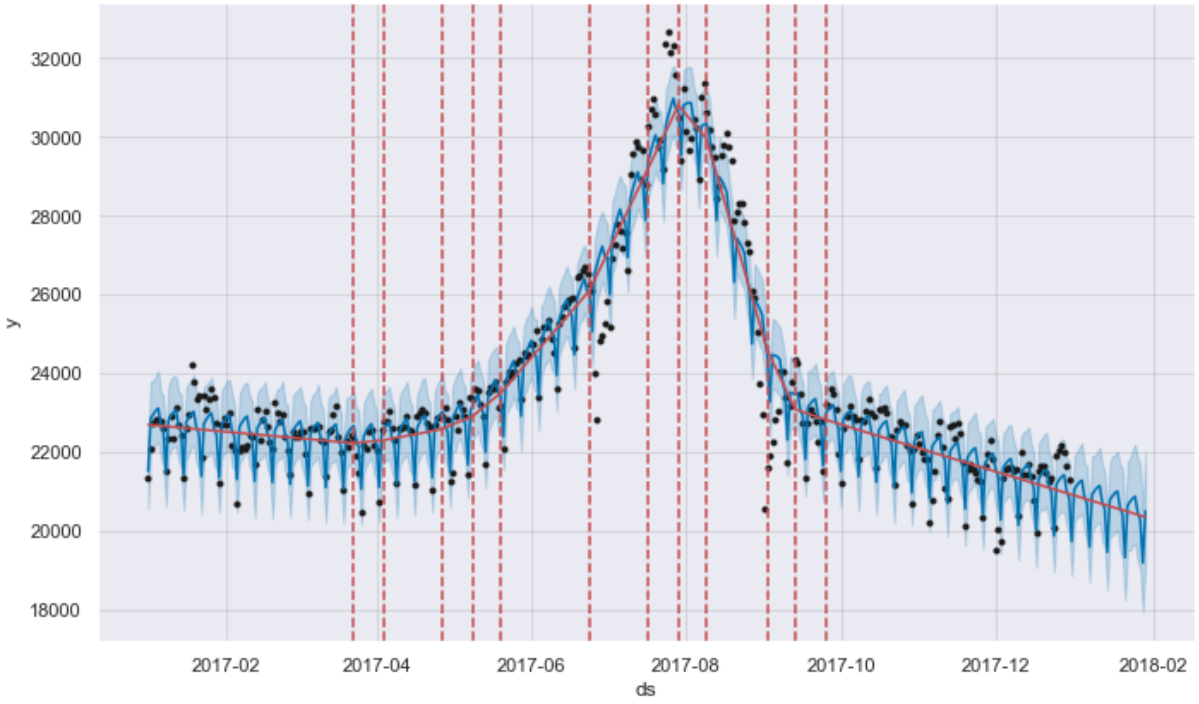
Contudo, a predição não está muito efetiva. Isso ocorre porque a sazonalidade do consumo não foi considerada. É esperado que alguns meses do ano apresentem um gasto menor de energia. O padrão semanal até que faz um certo sentido, pois dias de trabalho certamente exigem mais energia do que finais de semana.

A sazonalidade padrão do Prophet é incluída somando seu impacto ao trend dos dados para chegar nas predições[2]. Então ajustamos para sazonalidade multiplicativa para os valores previstos possivelmente acompanharem melhor o trend. Infelizmente, não fez grande diferença. Por isso, nem colocamos a figura do plot anterior. No notebook, pode-se perceber como é semelhante a essa da sazonalidade multiplicativa.

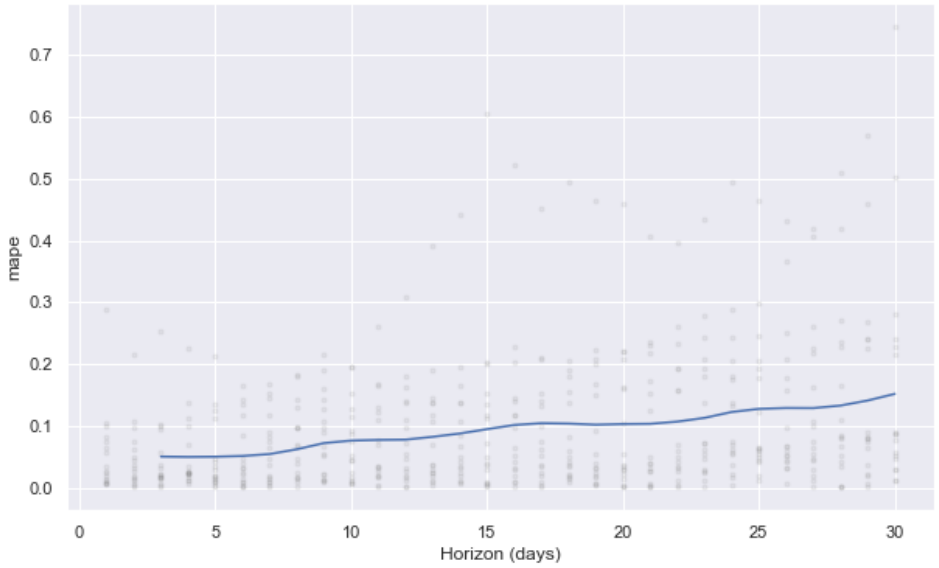
[2] https://facebook.github.io/prophet/docs/multiplicative_seasonality.html



Verificamos também os changepoints, os pontos onde a taxa de crescimento do gráfico muda, para a adaptação correta do modelo.



Por fim, utilizamos Cross Validation e não tivemos bons resultados. A figura abaixo demonstra baixa coverage, utilizando a métrica do Mean Absolute Percentage Error (mape).



Conclusão

Em termos gerais, o modelo do polinomial de grau 9 foi o que teve a melhor performance, mas ainda não foi satisfatório. A série temporal teve um coverage baixo. Comparado com o Trabalho 1, todos os modelos tiveram menos eficiência. Não encontramos nenhum caso de overfitting.

2. Modelo de Classificação (in-vehicle-coupon-recommendation)

Esse trabalho foi feito em cima de um dataset que dada determinadas respostas de um usuário, indica se ele aceitou ou rejeitou um cupom. Para isso foi necessária uma análise exploratória dos dados, assim como uma limpeza para as avaliações feitas dos modelos.

- **Análise prévia e limpeza dos dados**

O primeiro passo foi analisar e limpar os dados, procurando por valores duplicados, valores nulos e observando também a distribuição de cada atributo. Em relação aos valores duplicados, eles foram rapidamente removidos (eram cerca de 74 tuplas).

Já os valores nulos, dois casos aconteceram. O primeiro foi o atributo “car” que era constituído praticamente de valores nulos (12502 de 12610). Nesse caso, o atributo foi desconsiderado, pois pouca coisa poderia ser aproveitada dele. Já o restante das colunas, o grupo achou mais interessante imputar os valores com o valor mais frequente daquele determinado atributo.

Ao analisar a distribuição, o grupo notou que diversos atributos eram nominais (as classes não possuem ordem), enquanto outros eram ordinais (as classes possuem ordem).

Apenas alguns não eram classes, mas sim números inteiros, como foi o caso da temperatura.

Um detalhe interessante que surgiu nessa etapa foi em relação à coluna "toCoupon_GEQ5min", que indica o valor 1 no dataset inteiro, isso foi visto ao notar que na distribuição (*value_counts*) havia apenas um único valor. Assim, como a coluna não é um atributo que pode agregar para o modelo, ela foi retirada.

Antes então de fazer o mapeamento para transformar o texto das classes em valores inteiros, foi gerada a matriz de correlação, para ver se algum outro atributo poderia ser removido dessa análise.

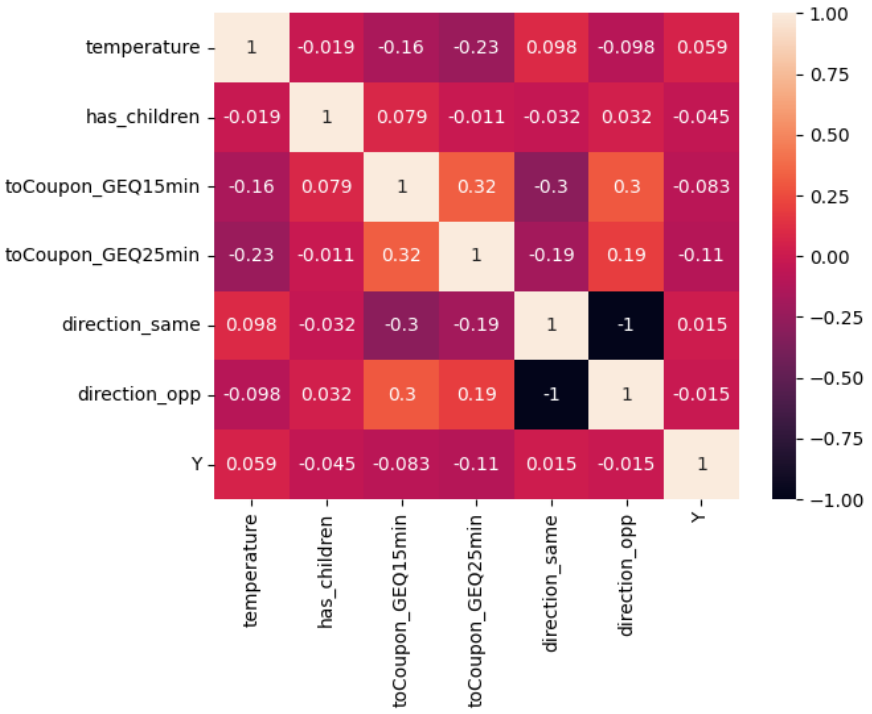


Figura 1: Matriz de Correlação

Como é possível notar, direction_same e direction_opp são altamente correlacionados (um é o oposto do outro). E, realmente, é possível notar que esse é o caso pela própria documentação do dataset. O primeiro indica se o restaurante está na mesma direção do destino e o segundo indica se o restaurante está na direção oposta do destino[3]. Por isso, a coluna "direction_opp" foi removida. Pela matriz de correlação indicada acima, não há outros casos de alta correlação entre dois atributos.

O mapeamento dos atributos ordinais (age, income, Bar, CoffeeHouse, CarryAway, RestaurantLessThan20, Restaurant20To50) e dos cardinais (destination, passanger, weather, coupon, gender, maritalStatus, education, occupation) foi feito, em que no caso dos ordinais foi preciso fazer de forma manual qual seria a ordem das classes. No caso dos

[3] https://archive.ics.uci.edu/ml/datasets/in-vehicle+coupon+recommendation

cardinais bastava avaliar quantas eram as classes (*value_counts*, função do pandas) e atribuir um número a partir de 0 para elas.

- **Análise exploratória dos dados**

Antes de passar para a avaliação dos modelos, foi feita também uma análise em cima do dataset entregue, verificando o que é mais provável para o usuário (aceitar ou não o cupom) de acordo com cada classe de cada coluna. Nesse momento, o grupo descobriu fatos interessantes sobre o dataset, como o fato de que é mais provável que um usuário aceite um cupom se ele não está indo para nenhum lugar urgentemente. Isso faz sentido, porque ele não tem a responsabilidade de estar em algum lugar e o tempo de aceitar o cupom não o atrapalharia.

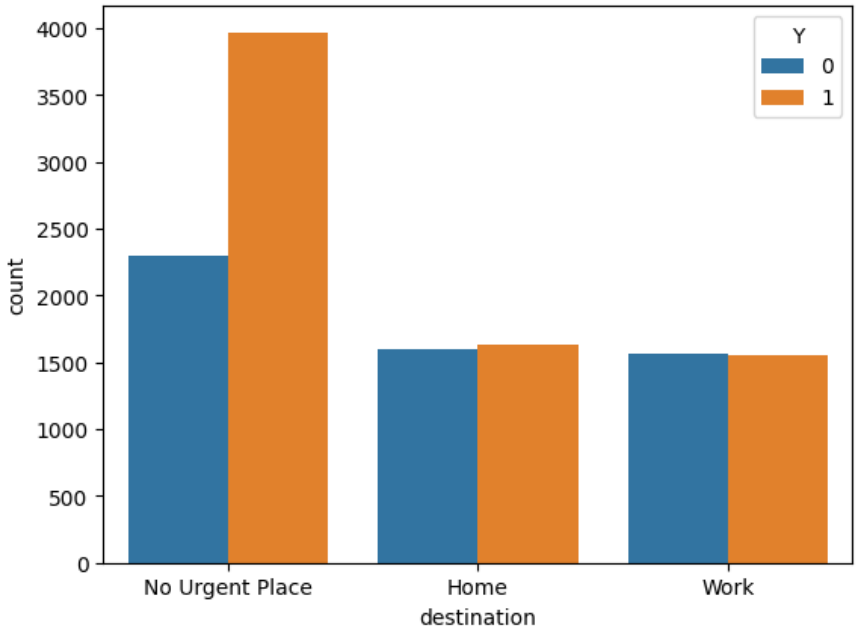


Figura 2: Gráfico que indica se o usuário aceitou ou não o cupom de acordo com cada classe da coluna 'destination'

Ou, por exemplo, que há uma maior probabilidade do usuário aceitar o cupom se ele estiver em um horário que não seja nos extremos do dia (nem muito cedo, nem muito tarde). O usuário também tem mais chance de aceitar um cupom com validade maior (1 dia em vez de 2 horas) e se o dia não está chovendo ou nevando.

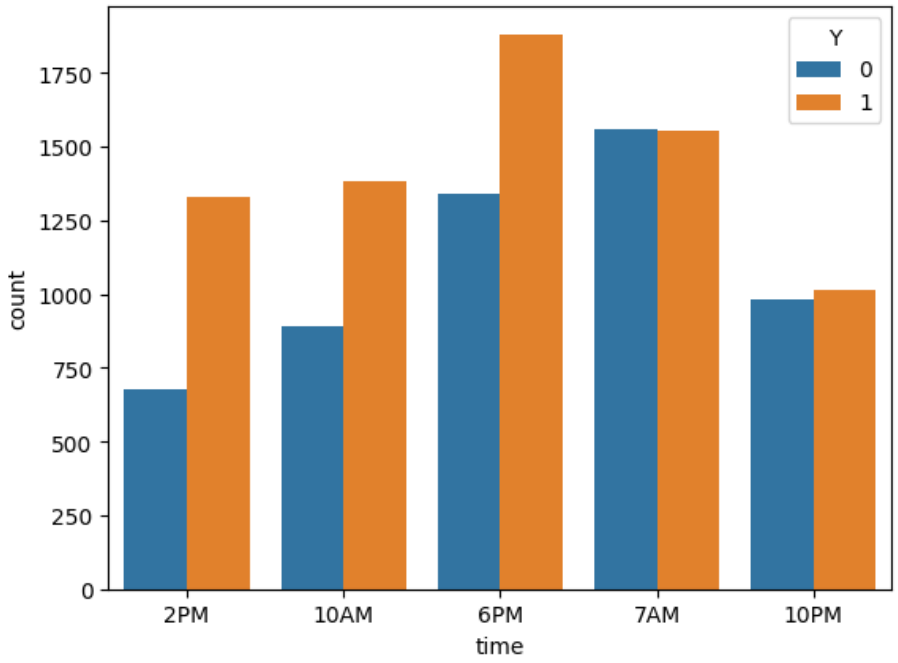


Figura 3: Gráfico que indica se o usuário aceitou ou não o cupom de acordo com cada classe da coluna 'time'

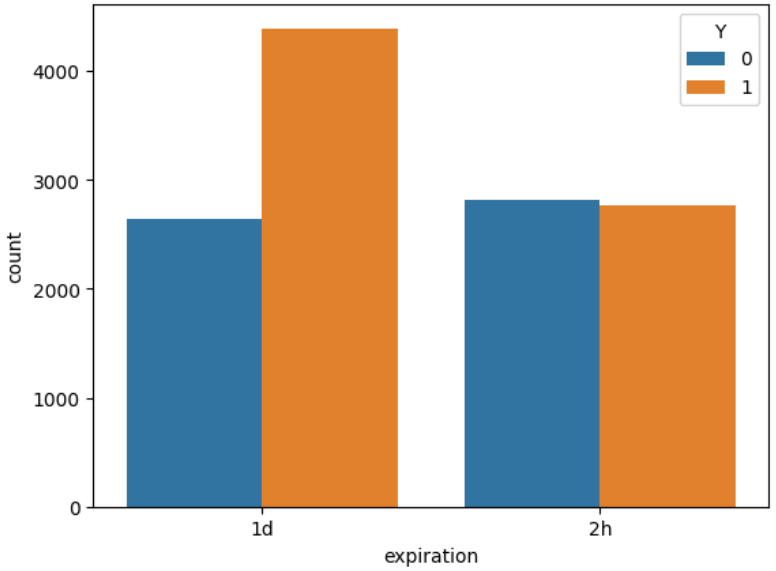


Figura 4: Gráfico que indica se o usuário aceitou ou não o cupom de acordo com cada classe da coluna 'expiration'

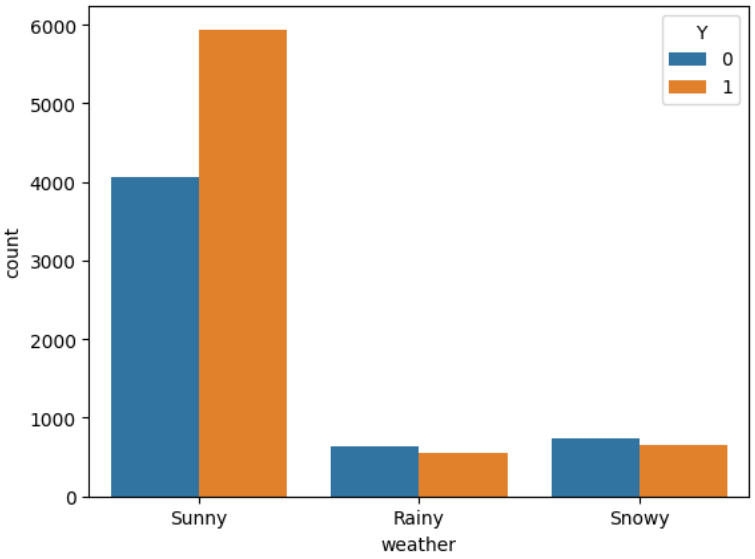


Figura 5: Gráfico que indica se o usuário aceitou ou não o cupom de acordo com cada classe da coluna 'weather'

Todos esses aspectos são importantes no caso da empresa que está organizando essas pesquisas. Dessa forma, é possível eventualmente descartar possibilidades de tentativa de entrega do cupom. Dependendo do caso, o esforço de tentar oferecer um cupom em uma clima chuvoso não compensa. São decisões de negócio que podem ser observadas pelos dados coletados.

- **Avaliação dos modelos**

Essa foi a parte que mais reproduziu o que foi feito no trabalho anterior (G1 - Modelo de Classificação). A diferença maior foi em relação ao uso do *cross_validate* e tuning de hiperparâmetros. No trabalho passado, a *cross_validate* do SKLearn não foi utilizada porque os dados de treinamento tinham um tratamento diferente dos dados de teste, o TF-IDF não poderia ser feito para o X como um todo porque ele estaria calculando em cima dos dados de teste e os dados de teste em cima dos de treinamento, causando um possível *data leakage*. Por isso, não era possível passar X e Y na *cross_validate* como o esperado. Pelo mesmo motivo também não era possível usar o *GridSearchCV*.

Como essa não foi a abordagem tomada pela maioria, foi entendido que mesmo com esse problema daria sim para utilizar ambas as funções e que esse pormenor poderia ser desprezado. Um caso similar também ocorreu nesse *dataset*, porém foi tomada a abordagem de manter o uso do *cross_validate* e do *GridSearchCV* mesmo com essa possibilidade de *data leakage*.

No dataset de recomendação de cupons, o que acontece é em relação à imputação dos dados faltantes. É recomendado que essa imputação seja feita apenas nos dados de

treinamento, porque os dados de teste poderiam influenciar qual seria a classe mais frequente. No entanto, ao usar a função pronta do SKLearn *cross_validate* e *GridSearchCV*, a separação entre dado de teste e treinamento (usando o *StratifiedKFold*) é feita dentro do método.

MODELO	**ACURÁCIA**	**NEG LOG LOSS**	**F1-SCORE MACRO**	**F1-SCORE WEIGHTED**
Regressão Logística	62.96%	-0.64	60.34%	61.72%
RandomForest	76.35%	-0.51	75.57%	76.16%
KNN	67.12%	-1.61	65.71%	66.65%
SVM	67.84%	-0.60	65.57%	66.76%

Acima é possível observar nossos resultados para os modelos avaliados. O tuning de hiperparâmetros foi feito apenas no que havia dado o melhor resultado previamente por motivos de performance (tivemos dificuldade de rodar o *GridSearchCV* nos nossos computadores pessoais). No caso acima, o RandomForest antes tinha obtido uma acurácia com cerca de 75%, porém após o tuning subiu para aproximadamente 76%.

- **Conclusão**

É possível observar, analisando o último trabalho feito para um modelo de classificação, que a acurácia não é tão boa quanto esperado. No entanto isso acontece porque esse era o tipo de resultado esperado para o dataset proposto. Esse não é o caso para o de recomendação de cupons. Uma diferença, por exemplo, é que os dados não são tão linearmente separáveis quanto o das descrições e títulos dos vídeos do Youtube. A análise de texto aumentava a dimensionalidade de tal forma que a regressão logística se superou na avaliação dos modelos. No caso aqui, os dados não são linearmente separáveis como antes e, por isso, modelos não-lineares obtiveram melhor resultado.

Um detalhe a ser notado também é a questão da possibilidade do *data leakage* por ter feito a imputação dos dados depois da separação do dataset em treinamento e teste. Isso foi discutido nos tópicos anteriores, mas é importante deixar claro que o grupo tem noção dessa possibilidade, mas entendeu que esse seria um erro “desprezível” para o trabalho aqui proposto.

Por fim, para trabalhos futuros, seria interessante corrigir essa questão da imputação e também utilizar mecanismos como o RFE e PCA para escolher os atributos de acordo com a sua importância. Isso seria feito mais para conhecer as duas técnicas do que a

necessidade de utilizá-las para esse dataset específico. Como último desejo, o grupo também esperava ter adentrado um pouco mais no atributo de ocupação, já que ele era o que mais possuía classes diferentes e as classes poderiam ter alta correlação entre si.