Nanodegree Engenheiro de Machine Learning

Modelo de previsão de vendas da Rossmann

Projeto final

Fabio Shindi Uchidate
01 de agosto de 2018

I. Definição

Visão geral do projeto

Para melhorar os lucros, atualmente as empresas buscam trabalhar com o menor estoque e alocação de recursos possíveis. O processo é puxado pelo cliente, e não mais empurrado como acontecia antes. O que acarretava imensos estoques e pouca eficiência. Esse novo processo vem se aprimorando por meio da logística e da previsão de vendas. Através do cruzamento de dados de pesquisas de mercado, tendências e histórico de vendas, as empresas têm assim definido uma melhor estratégia de demanda e sobrevivido em meio a forte concorrência.

Com a Rossmann não é diferente, a rede alemã de drogarias opera mais de 3.000 drogarias em 7 países europeus e sabe que suas vendas podem ser influenciadas por vários fatores, como promoções, proximidade de concorrentes, feriados, sazonalidade, localidade, entre outros. Seus gerentes de drogarias têm a tarefa de prever suas vendas diárias com até seis semanas de antecedência, porém com milhares de gerentes individuais prevendo vendas com base em suas circunstâncias únicas, a precisão dos resultados pode ser bastante variada.

A área de supermercados, indústrias de alimentos, bem como outros setores enfrentam desafios parecidos. Eu, como Engenheiro de Alimentos, tendo trabalhado na área de consultoria empresarial com foco em resultados, tive o interesse de vasculhar e entender melhor os dados da Rossmann.

Descrição do problema

Em sua primeira competição no Kaggle, a Rossmann propôs o desafio de prever 6 semanas de vendas diárias para 1.115 lojas localizadas em toda a Alemanha. Previsões de vendas confiáveis permitem que os gerentes de loja criem agendas de pessoal eficazes que aumentam a produtividade e a motivação. Ao ajudar a Rossmann a criar um modelo de previsão robusto, os gerentes de loja poderão manter o foco naquilo que é mais importante para eles: seus clientes e suas equipes.

Embora a competição já esteja encerrada, o desafio continua válido. Sempre podemos melhorar algo existente.
(https://www.kaggle.com/c/rossmann-store-sales)

Métricas

De forma que o desempenho do modelo será avaliado utilizando o RMSPE (Root Mean Squared Predction Error), que é a raiz quadrada do erro quadrático médio, implantarei a função RMSPE no projeto, de forma que o avaliador da Udacity possa executar o código. Quanto mais próximo de zero, melhor é o ajuste entre as unidades.

O RMSPE tem a seguinte fórmula:

$$\text{RMSPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2},$$

onde y_i denota as vendas de uma única loja em um único dia e $\hat{y}_i$ denota a previsão correspondente. Dados de dias e lojas que não venderam, são ignorados na pontuação.

II. Análise

Exploração de dados

O projeto usa um total de três arquivos de dados, "train.csv" para treinamento do modelo, "store.csv" para informações da loja Rossmann e "test.csv" para prever o atributo "Sales".

Os registros históricos de vendas das 1115 lojas fornecidas pela Rossmann são as seguintes:

Nome da coluna	Significado
Id	Identificação (Store, Date)
Store	Identificação que representa uma tupla (Store, Date), dentro do conjunto de testes
Sales	Volume de vendas diárias *(→ NOSSA PREVISAO)*
Customers	Quantidade de clientes em um determinado dia
Open	Indicador se loja estava aberta: 0 = fechada, 1 = aberta
StateHoliday	Indica um feriado estadual. Normalmente todas as lojas estão fechadas nos feriados estaduais. a = Feriado público, b = Páscoa, c = Natal, 0 = não feriado
SchoolHoliday	Indica se a (Store, Date) foi afetada pelo fechamento das escolas
Storetype	Diferencia entre 4 modelos de lojas diferentes: a, b, c, d
Assortment	Descreve um nível de funcionamento: a = básico, b = extra, c = estendido
CompetitionDistance	Distância em metros até a loja concorrente mais próxima
CompetitionOpenSince [Month/Year]	Indica o ano e mês aproximado do momento em que o concorrente mais próximo foi aberto
Promo	Indica se uma loja está executando uma promoção nesse dia
Promo2	Promo2 é uma promoção contínua e consecutiva para algumas lojas: 0 = loja não está participando, 1 = loja está participando
Promo2Since[Year/Week]	Descreve o ano e a semana em que a loja começou a participar do Promo2

PromoInterval	Descreve os meses em que o Promo2 começou e reiniciou. Por exemplo, "fev, mai, ago, nov" significa que uma rodada de promoção começou em fevereiro, outra em maio, outra em agosto, e mais outra em novembro de qualquer ano para aquela loja

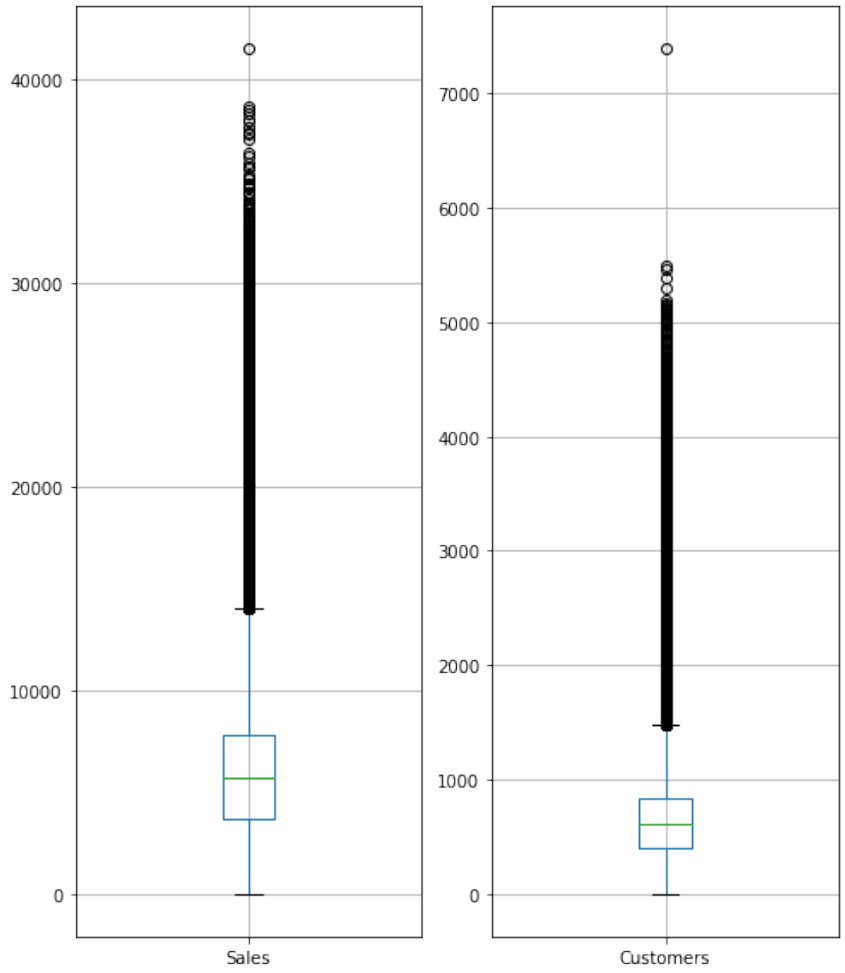
Investigando preliminarmente a partir de uma amostra de dados, pode-se ver que alguns dos atributos se relacionam a tempo, como 'StateHoliday', 'SchoolHoliday', 'Promo', entre outros. E ainda que alguns campos de alguns atributos como "Promo2", "Open" contém valores ausentes.

Para recursos binários (os valores podem ser apenas 1 ou 0), então é razoável preencher os valores ausentes. Um exemplo é o caso dos atributos Open e Sales, onde caso Open esteja ausente, podemos preenchê-lo conforme o atributo Sales, ou seja, se Sales > 0, significa que a loja está aberta, sendo o inverso também verdadeiro.

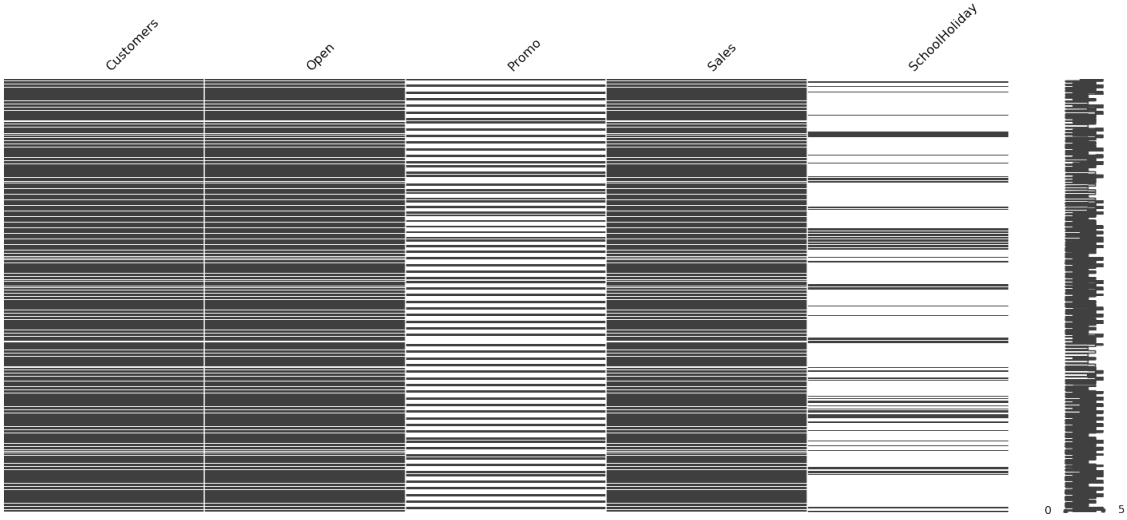
Já em relação aos outliers, podemos analisar através dos quartis e da variância, principalmente nos atributos 'Customers' e 'Sales'. Podendo estas duas características serem diretamente descartadas após o teste, uma vez que podemos prever o valor de 'Sales' com os dados restantes.

Visualização exploratória

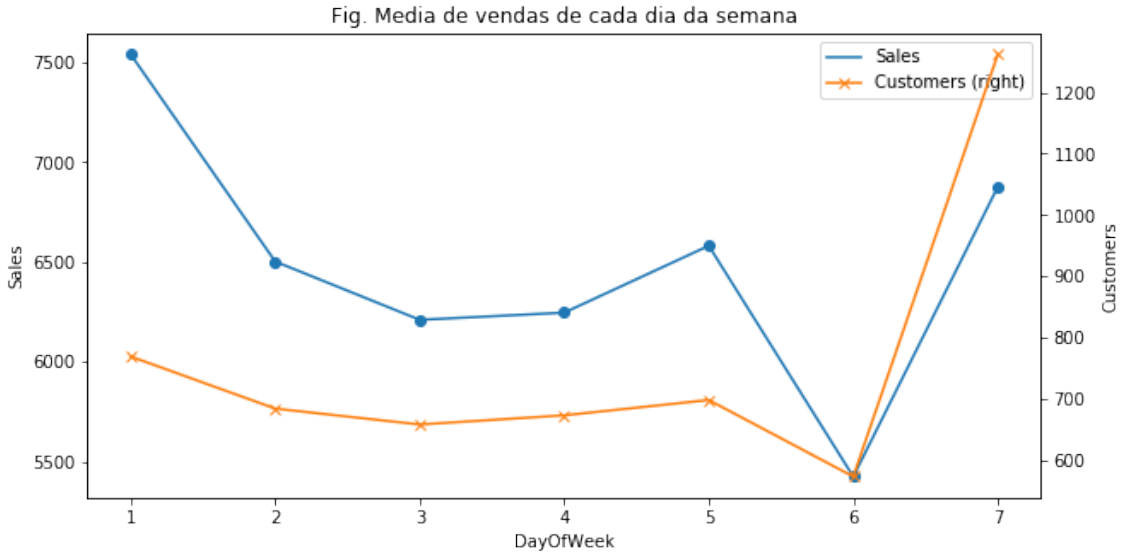
Fig. Gráfico BoxPlot dos atributos "Sales" e "Customers"



Apesar do descarte de dados com vendas zeradas, podemos verificar através do diagrama boxplot acima, que ainda há outros outliers nos atributos ‘Sales’e ‘Customers’.



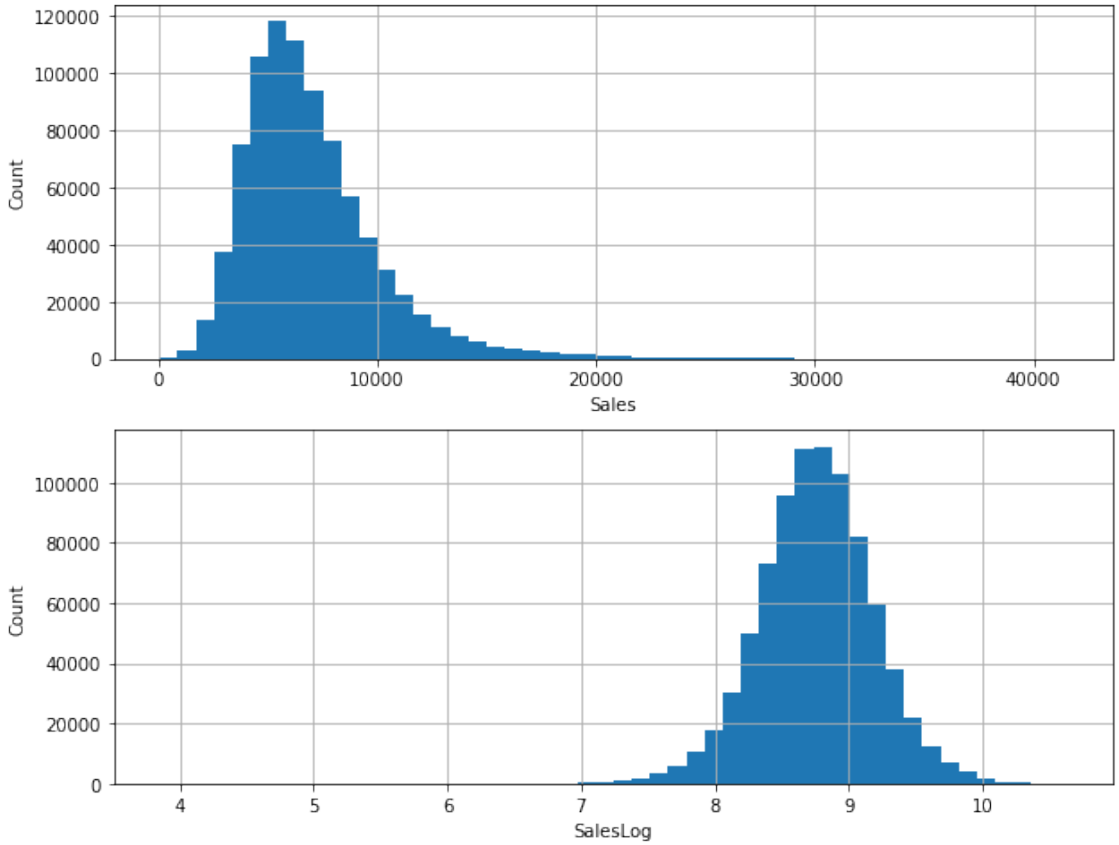
A figura acima mostra a ausência ou não de valores em determinados atributos, podemos visualizar que pode haver ou não correlação entre eles.



O gráfico de linhas acima nos mostra a distribuição das vendas no decorrer da semana. Nota-se que as vendas aumentam nos fins de semana, especialmente nos domingos e decrescem no decorrer da semana.

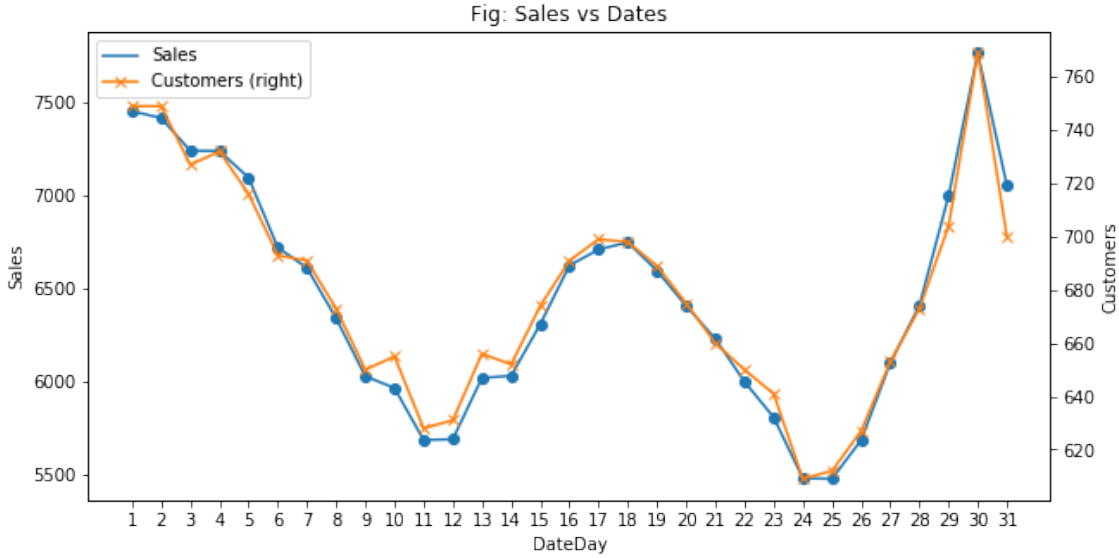
É interessante notar que o maior movimento de clientes acontece nos sábados, porém é nos domingos onde as drogarias mais vendem financeiramente, sendo que os sábados (eixo x – número 7) são os dias com o menor ticket médio, ocorrendo o oposto nos domingos (eixo x – número 1).

Fig. Distribuição do atributo "Sales" and "SalesLog"



A distribuição das vendas tem uma inclinação relativamente grande conforme mostra a gráfico de barras acima. Ao normalizarmos através da função log, a distribuição segue próximo a normal.

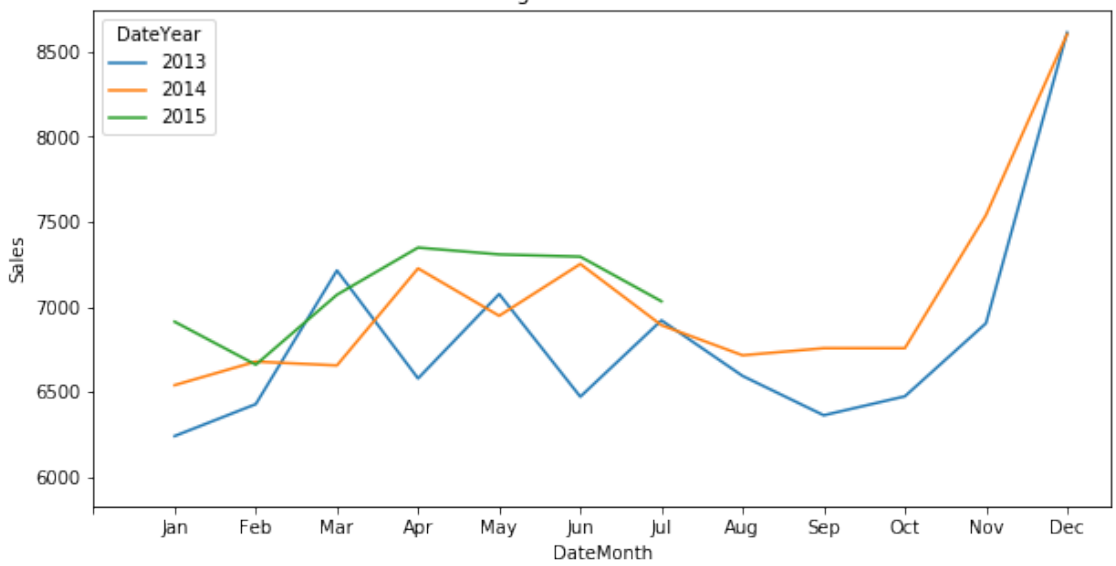
Fig: Sales vs Dates



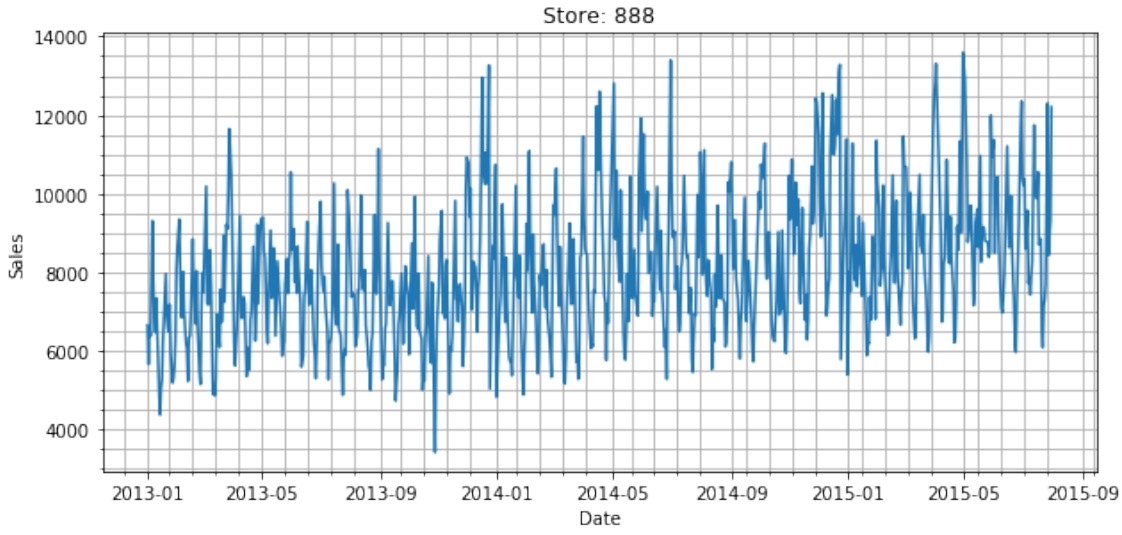
O gráfico acima mostra a distribuição das vendas no decorrer do mês.

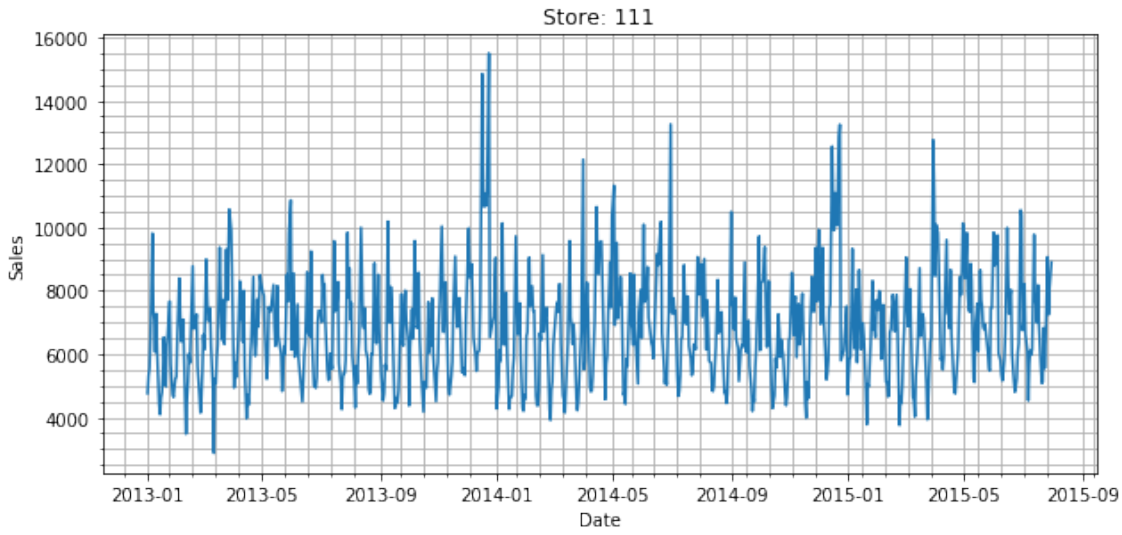
Nota-se que há correlação natural entre os atributos ‘Customers’ e ‘Sales’ como já esperado, pois quanto mais clientes mais se vende. E uma variação nas vendas no decorrer do mês, com picos próximos do meio e início/fim dos meses.

Fig: Vendas Anuais



O gráfico mostra o volume médio de vendas para cada mês no período de janeiro/13 a julho/2015. Pode-se visualizar picos de consumo nos finais de ano e padrões de venda diferente no decorrer no ano comparando-se os diferentes anos.





O gráfico acima representa a distribuição de vendas de 2 lojas de mesmo modelo. Pode-se observar que ocorrem tendências de venda semelhantes em ambas as lojas.

Algoritmos e Técnicas

A partir da visualização exploratória acima, nota-se que as vendas estão intimamente relacionadas a data, tendo dias específicos como fins de semana e inverno (hemisfério norte/Europa) onde vende-se mais, bem como meio da semana onde vende-se menos.

Há outros atributos ainda não visualizados, que certamente influencia nas vendas. Tais como distância dos concorrentes, férias escolares e promoções.

Os valores omissos ou os discrepantes (outliers) também afetam nossa previsão de vendas. Para melhor prevermos, os outliers do atributo 'Sales' e 'Customers' foram descartados através do Método Tukey. Consideramos esta como a primeira etapa do pré-processamento.

A segunda etapa do pré-processamento, consiste processamento de valores ausentes e a realização de alterações apropriadas em recursos importantes ou a geração de novos recursos parciais de recursos antigos para facilitar o treinamento do modelo.

Na terceira etapa, os dados de treinamento são divididos aleatoriamente em conjuntos de treinamento e de testes utilizando-se do cross_validation.train_test_split e iniciado o treinamento através dos algoritmos XGBoost e DecisionTree.

XGBoost

O excelente desempenho do Gradient Boosting e a eficiente implementação do XGBoost fazem com que ele tenha um bom desempenho neste projeto. O primeiro lugar na competição de Kaggle utilizou deste algoritmo para prever os resultados. O XGBoost personaliza uma classe de matriz de dados, a DMatrix, que é pré-processada no início do treinamento para melhorar a eficiência de cada iteração. O DMatrix é uma estrutura de dados interna usada pelo XGBoost que é otimizada tanto para a eficiência da memória quanto para a velocidade de treinamento.

Os principais parâmetros que precisam ser depurados são os seguintes:

- eta: controla a taxa de aprendizagem através de um fator entre 0 e 1. O parâmetro é utilizado para evitar overfitting, tornando o processo de reforço mais conservador. Menores valores desse parâmetro implicam em maior valor para nrounds: baixo valor eta significa modelo mais robusto para overfitting, mas mais lento para calcular. Padrão: 0,3
- nthread: número de encadeamentos a serem usados para carregar dados de uma matriz numpy. Se -1, usa encadeamentos máximos disponíveis no sistema.
- subsample: proporção da subamostra da instância de treinamento. Configurando-o para 0,5 significa que o xgboost coletou aleatoriamente metade das instâncias de dados para cultivar árvores e isso evitará overfitting. Isso torna o cálculo mais curto (porque menos dados para analisar). É aconselhável usar este parâmetro com eta e aumentar nround.
- colsample_bytree: proporção de subamostras de colunas ao construir cada árvore. Padrão: 1

- DecisionTree Regression: O modelo de regressão da árvore de decisão treina o modelo aprendendo as regras de decisão a partir dos dados, corre mais rápido e aprende as regras de acordo com as características do jogo, conseguindo assim uma melhor previsão. Os parâmetros que precisam ser depurados aqui são principalmente min_samples_split, max_depth e min_samples_leaf. O método usa grid_search para selecionar os parâmetros ótimos.

Benchmark

O projeto usa o RMSPE para índice de avaliação. Quanto menor a pontuação, melhor o desempenho do modelo.

Considerei o benchmark como 0.10021, sendo este o melhor resultado obtido na já encerrada competição.

III. Metodologia

Pré-processamento de dados

A primeira etapa do pré-processamento foi concatenar os conjuntos de treinamento e teste fornecidos pela Rossmann, e rotulá-los sua origem através de uma nova coluna 'isTrain'. Posteriormente foi dado o início a identificação dos atributos e variáveis-alvo que serão utilizadas na predição.

Na segunda etapa foi lidar com os dados anormais de lojas que estavam abertas, sem vender absolutamente nada.

Agora numa terceira etapa, o atributo ‘Date’ foi destrinchado em outros atributos para uma análise mais aprofundada, temos ‘DateDay’, ‘DateMonth’, ‘DateWeek’, ‘DateYear’, ‘DateDayOfYear’.

Na quarta etapa criou-se o atributo 'CompetitionSinceOpen' da união dos atributos 'CompetitionOpenSinceYear' e 'CompetitionOpenSinceMonth', e ainda dividiu-se o atributo 'PromoInterval' em 4 diferentes colunas, uma vez que este atributo contemplava meses, e poderia conter até 4 meses em cada campo.

Numa quinta etapa adicionou-se os novos atributos criados a lista de atributos a serem utilizados na predição.

Após obtermos nosso conjunto para treinamento, lidamos então com os outliers numa sexta-etapa. Os outliers geralmente causam um grande desvio nos resultados de previsão do modelo. Utilizei do Método Tukey para descartar os dados fora do intervalo de 1,5 quartis.

O processamento é o seguinte:
- Quantidade de outliers no atributo ‘Sales’ = 26701
- Quantidade de outliers no atributo ‘Customers’ = 38095
- Quantidade de outliers comuns = 18985
- 1,90198062454 % de dados descartados

Numa sétima etapa, a função de avaliação RMSPE é implementada e novos atributos são criados e adicionados a lista de atributos para predição 'SalesPerDay', 'CustomersPerDay', 'SalesPerCustomersPerDay'.

Implementação

Os dados de treinamento primeiro são divididos aleatoriamente em um conjunto de treinamento e um conjunto de teste utilizando cross_validation.train_test_split.

E então treinado o modelo XGBoost com os seguintes parâmetros: *'bst:max_depth':12, 'bst:eta':0.0095, 'subsample':0.8, 'colsample_bytree':0.7, 'silent':1, 'objective':'reg:linear', 'nthread':6, 'seed':1*

A melhor pontuação local foi obtida após o treinamento para 4.000 rodadas.

Em seguida o modelo de previsão de regressão da árvore de decisão foi treinado, e utilizando-se do grid_search foram selecionados os melhores parâmetros. O resultado obtido foi: {'min_samples_split': 30, 'max_depth': None, 'min_samples_leaf': 8}

Refinamento

No treinamento do modelo de regressão da árvore de decisão, a pontuação local inicial é de 0,2543, que é reduzida para 0,2260 após o Grid_Search.

XGBoost do índice de público é 0,16088, modelo de regressão árvore de decisão é 0,19176, os resultados finais previstos de integração dos dois modelos é multiplicado por um fator de 0,65 e 0,35 respectivamente, obtendo um resultado final de 0,15952 .

O princípio do Ensemble é aplicado aqui: em teoria, o efeito de múltiplos modelos é sempre melhor que o de um único modelo, especialmente quando as diferenças entre os modelos são grandes.

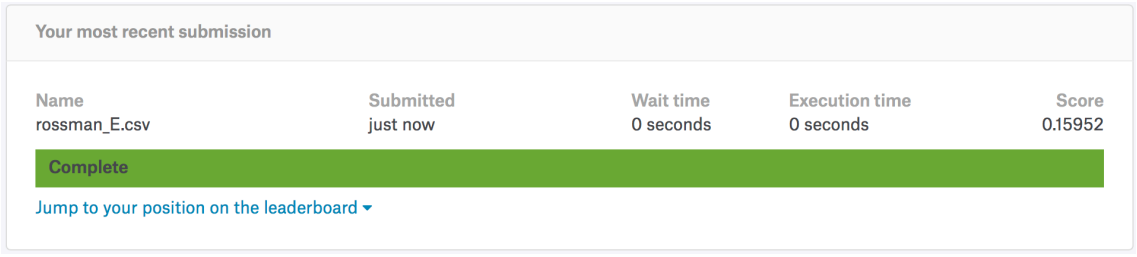
IV. Resultados

Modelo de avaliação e validação

O modelo final de predição foi calculado levando-se em conta, a profundidade de cada um dos algoritmos. Como o XGBoost é mais elaborado, merece um maior peso no modelo final de predição.

Calculo:
0,65*(predição XGBoost) e 0,35*(predição Árvore de Decisão).

O desempenho final é mostrado no Kaggle.



Pode-se visualizar os indicadores Score Privado e Score Público, onde o Score Público é maior do que o Score Privado, indicando assim que o modelo teve desempenho razoável e confiável.

Embora não tenha superado o benchmark, o que já era esperado, pois meu objetivo ao definir o benchmark era obter um valor de referência.

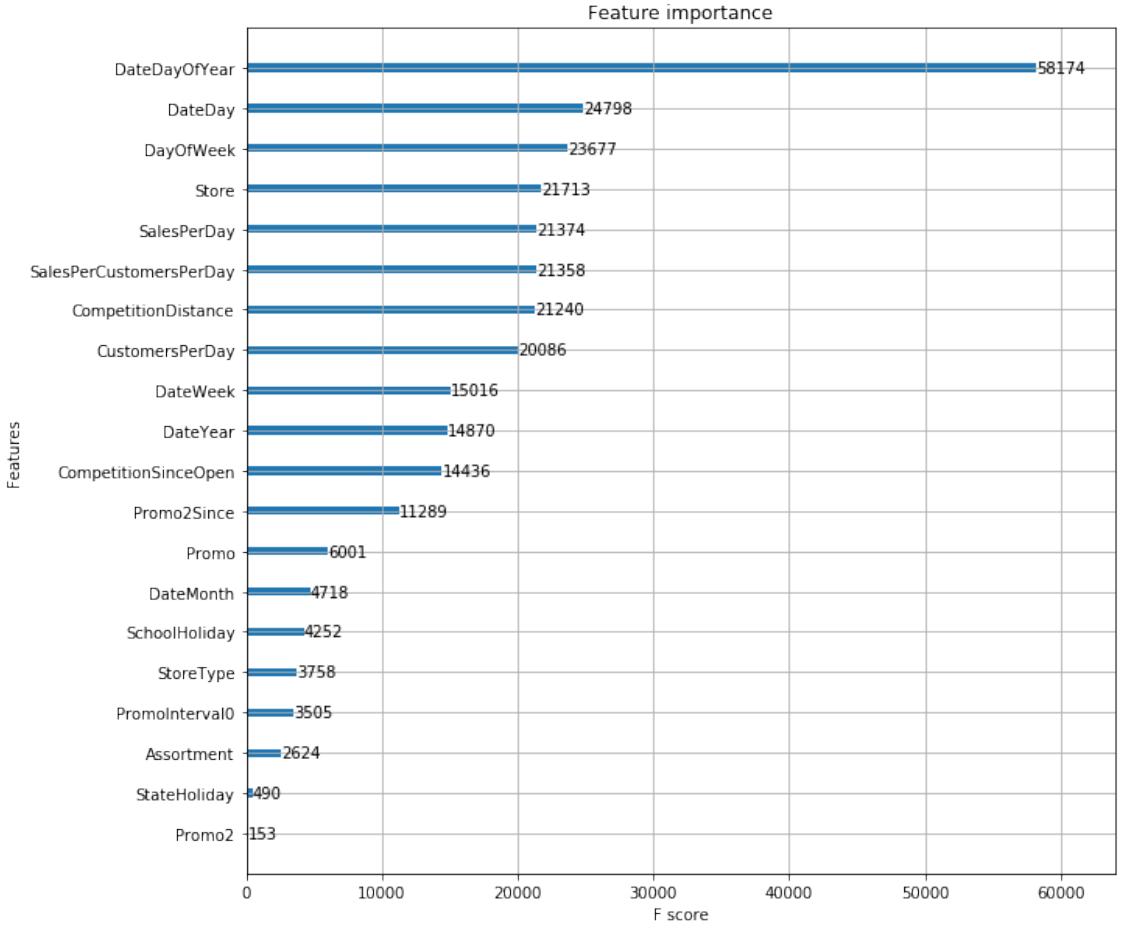
Justificativa

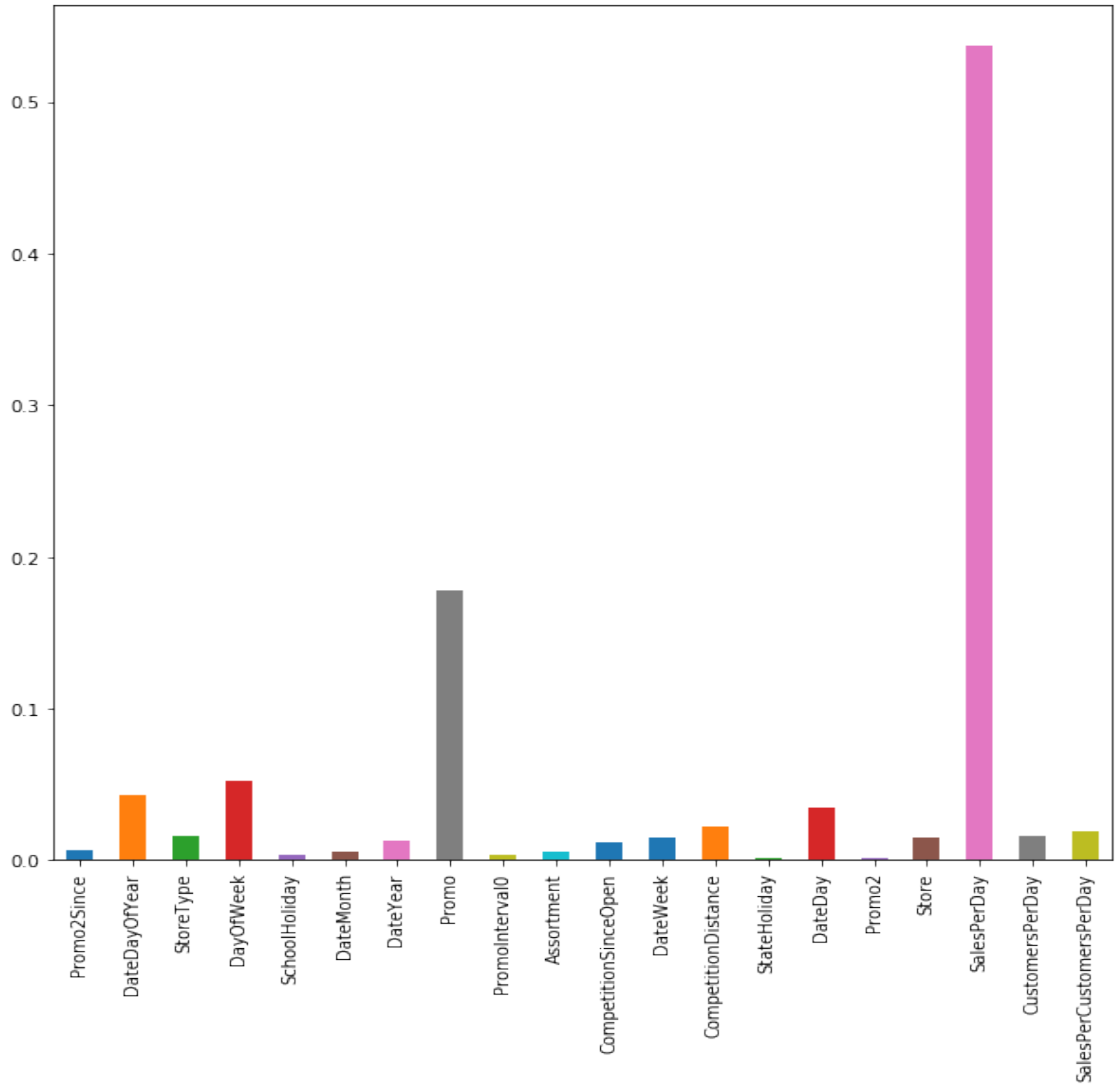
A união de diferentes algoritmos para predição pode ser um ótimo caminho, conforme o desempenho obtido do modelo Ensemble final.

Através da união dos modelos XGBoost e DecisionTree Regression por meio de diferentes pesos obtive um melhor resultado do que as análises individuais de cada um dos modelos de referência.

V. Conclusão

Forma livre de visualização





Nas duas figuras acima, são apresentados a importância dos atributos nos diferentes algoritmos utilizados, o primeiro trata-se do XGBoost e o segundo a Regressão da árvore de decisão.

Nota-se que não há consistência absoluta nos dados, uma vez que são apresentados diferentes atributos e ordenações como mais importantes. As quatro características mais importantes no XGBoost foram 'DateDayOfYear', 'DateDay', 'DayOfWeek' e 'CompetitionDistance', enquanto que na regressão da Árvore de Decisão foram SalesPerDay, Promo, DayOfWeek e DateDayOfYear.

É interessante notar como os atributos 'Dia do ano' e 'Dia da semana', possuem uma grande correlação nas vendas, o que de certa forma era esperado. Por exemplo, espera-se que nos fins de semana venda-se mais, e que determinados dias do ano também, tais como feriados. Outro atributo que se destaca são as promoções, que dá um impulso nas vendas da Rossmann.

Exemplos testados

Por meio do conjunto de treinamento fornecido pela Rossmann, o qual utilizamos para ajuste do modelo, selecionamos algumas amostras e comparamos nossa Previsão com o Real.

- Amostra 10 = Previsão: 10248.43, Real: 10457.00
- Amostra 35 = Previsão: 12315.87, Real: 12422.00
- Amostra 70 = Previsão: 11545.00, Real: 11496.27
- Amostra 500 = Previsão: 7175.64, Real: 6626.00
- Amostra 1000 = Previsão: 7149.21, Real: 7724.00

Pode-se notar que as amostras testadas apresentam previsões razoáveis comparadas ao dado real.

Reflexão

Todo o processo do projeto desde as primeiras aulas até o relatório final, é algo

No processo de realização deste projeto, é interessante observar que vários fatores podem ter diferentes efeitos no resultado final, pois os dois modelos possuem diferentes características de classificação, mas os resultados são todos iguais. Muito razoável.

O processo de aprendizagem e teste

Melhorias

Embora o modelo final tenha tido um resultado razoável, devido a minha pouca experiência com Machine Learning, podemos ver no ranking do Kaggle que os vencedores da competição obtiveram predições muito melhores.

Analisando detalhadamente possíveis melhorias que pode ser efetuada, temos:

- Pré-tratamento: pode-se buscar correlações entre atributos com criação de outros novos atributos a serem estudados, pode-se estipular a influência que uma nova loja concorrente traz e ainda correlacionar-se dados externos, como demanda, mercado com dados que temos da Rossmann.

- Seleção de modelo: Apenas o modelo de árvore de decisão e o modelo XGBoost são usados aqui. Podemos tentar utilizar outros modelos.

Documento de referência

- [《Beating Kaggle the easy way》 By Ying Dong](https://www.ke.tu-darmstadt.de/lehre/arbeiten/studien/2015/Dong_Ying.pdf)
- A Simple XGBoost Tutorial Using the Iris Dataset (https://www.kdnuggets.com/2017/03/simple-xgboost-tutorial-iris-dataset.html)
- Decision Tree e Random Forest (http://carlosbaia.com/2016/12/24/decision-tree-e-random-forest/)
- [Understanding XGBoost Model on Otto Dataset](https://www.kaggle.com/tqchen/otto-group-product-classification-challenge/understanding-xgboost-model-on-otto-data)
- [Github: ROSSMANN-KAGGLE](https://github.com/JohanManders/ROSSMANN-KAGGLE)