Projeto de conclusão de curso
Nanodegree Engenheiro de Machine Learning
Udacity

José Carlos Bezerra Filho

12/02/2018

1. Introdução

1.1. Sobre o problema

A formação de preço de um seguro é um desafio para as companhias de seguros ao redor do mundo, pois para fazer um preço é necessário entender qual o risco que cada cliente representa para a carteira da companhia. Em minha experiência de 7 anos como corretor de seguros eu aprendi que não existe um valor padrão para um seguro, principalmente de automóvel, algumas vezes um detalhe, que muitos acham pequeno, como o CEP pode agravar o prêmio em mais de 100%.

Ao longo dos anos as seguradoras ao redor do mundo têm investido quantias gigantes de dinheiro para evoluir seus departamentos de estatística, tudo isso para conseguir prever cada vez com mais precisão qual o risco que cada cliente representa para a sua carteira. Por ser um assunto com o qual trabalhei por muitos anos e sempre me deixou curioso eu escolhi fazer meu projeto baseado no desafio proposto pela Porto Seguro no Kaggle, onde o objetivo é encontrar qual a probabilidade de um determinado cliente usar o seguro no próximo ano. O desafio pode ser visualizado neste link: https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/data.

A grande questão é: como prever um cliente de alto risco? A resposta para essa pergunta é o motivo de as companhias de seguro ao redor do mundo investirem tanto dinheiro para melhorar seus modelos. Veja que para elas é interessante atrair cliente bons e afastar clientes ruins e para fazer isso é preciso classificar cada cliente de acordo com o seu perfil. As informações que elas utilizam são as mais variadas como: idade, sexo, CEP, CPF, etc. Prevendo quem são os clientes de baixo risco (clientes bons) e os clientes de alto risco (cliente ruins) no momento da cotação elas são capazes de oferecer vantagens como descontos e serviços extras para clientes de baixo risco e agravar o valor do prêmio para clientes de alto risco. Para este problema, como se trata de um problema de classificação, o ideal é utilizar um modelo de aprendizagem supervisionada.

1.2. Machine Learning

Para Arthur Samuel, Machine Leaning é: *"O campo de estudo que dá aos computadores a habilidade de aprender sem serem explicitamente programados".*

Os algoritimos de Machine Learning podem ser separados em dois grupos, são eles: aprendizagem supervisionada e aprendizagem não supervisionada. Por este problema se tratar de um problema de classificação vamos trabalhar com um modelo de aprendizagem supervisionada.

Imagine que eu coloque em suas mãos um objeto de metal fino e comprido, mais ou menos como um hashi, e peça para que você coloque

este objeto dentro de uma tomada, você faria isso? É logico que não, você deve ter pensado, e te pergunto, porque essa resposta?

Simples, aposto que na sua cabeça por um segundo você pensou que pela tomada passa energia elétrica e o metal é condutor de energia elétrica. A partir dessa informação você foi capaz de concluir que se fizesse o que eu lhe pedi a energia seria conduzida da tomada até a sua mão pelo objeto metálico e isso resultaria em um choque elétrico, o que seria desagradável, logo você tomou a decisão de não fazer. Podemos dizer então que você fez uma análise de dados presentes na sua memoria para tomar uma decisão.

Um algoritmo de Machine Learning do tipo aprendizagem supervisionada faz exatamente o que você neste caso, ele analisa dados passados para prever uma situação atual e tomar uma decisão.

1.3. Dados

A Porto Seguro disponibilizou três arquivos, são eles: train.csv, test.csv e sample_submission.csv.

O arquivo train.csv contem os dados dados para o treino do modelo, ele contém a coluna “target” que representa se aquele cliente usou ou não seguro, ou seja, é a coluna que devemos prever. O arquivo test.csv contem os dados para a predição e criação do arquivo de submissão, ele arquivo contém as mesmas colunas do arquivo train.csv com exceção da coluna “target” que é a coluna que deverá ser prevista.

Como os dados são reais eles foram reconfigurados e nome de cada variável alterada, levando em consideração a privacidade dos clientes e a segurança da empresa para não permitir que outros concorrentes saibam qual são as informações utilizadas por eles.

O que sabemos dos dados é o seguinte:

- Os campos com valores faltantes tem o valor -1.
- A coluna “id” representa a identificação do cliente representado naquela linha.
- A coluna “target” é binaria sendo 1 ou 0, o valor 1 representa que o cliente utilizou o seguro, enquanto o valor 0 representa que o cliente não utilizou o seguro.
- As colunas que contém “ind” no nome são colunas com dados referentes ao cliente.
- As colunas que contém “car” no nome são colunas com dados referentes ao carro.
- As colunas que contém “reg” no nome são colunas com dados referentes a região.
- As colunas que contém “calc” no nome são colunas com dados calculados pela seguradora.
- As colunas com “cat” no nome são colunas com dados categóricos.
- As colunas com “bin” no nome são colunas com dados binários.

ion.csv é um exemplo de como deve ser a submissão, ela deve ter duas colunas, uma com o “id” e outra com a probabilidade do cliente referente ao “id” da linha tem de usar o seguro no próximo ano.

O arquivo train.csv tem 595.212 linhas e 59 colunas, já o arquivo test.csv tem 892.816 linhas e 58 colunas (este arquico não tem a coluna target. Um exemplo dos arquivos pode ser visualizados nos links abaixo:

train.csv:https://docs.google.com/spreadsheets/d/1B8h1kBlmRVwN0TXxmL88mtpnUUNClCzHwWkZvh4i7CQ/edit?usp=sharing

test.csv:https://docs.google.com/spreadsheets/d/1tx6GymQBs7nG-YqpcAEyWRCXLohcntj_JAHu4UeGr3E/edit?usp=sharing

O arquivo sample_submission.csv é um exemplo de como deve ser a submissão, ela deve ter duas colunas, uma com o “id” e outra com a probabilidade do cliente referente ao “id” da linha tem de usar o seguro no próximo ano.

1.4. Métrica

A métrica proposta para avaliar a predição é o “Normalized Gini Coefficient”, podemos entender ele como:

$$Gini = 2 * AUC - 1$$

Onde significa represneta a Area Under the Receiver Operating Characteristic Curve (área sobre a curva de operação do receptor, em tradução livre). Essa curva representa o desempenho de um sistema de classificação binária. Ela é obtida através da representação gráfica dos pontos positivos verdadeiros e os pontos positivos falsos.

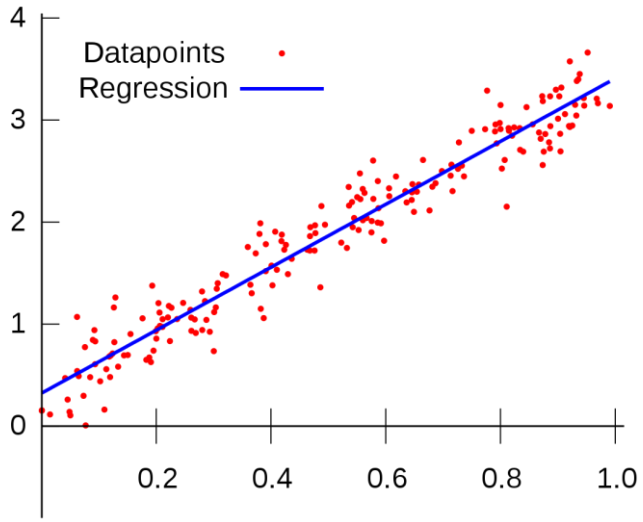
Acredito que essa métrica tenha sido escolhida para este desafio em específico por dois motivos, são eles:

- Predições aleatórias no AUC teriam um score de 0,5 enquanto no Gini seria 0, por isso tira a falsa impressão de uma em score diferente de 0 para uma predição totalmente aleatória.
- O Gini é usado em economia para avaliar diferenças sociais, sendo 0 uma pontuação totalmente igual e 1 uma situação totalmente desigual. Nesta caso como a intenção é diferenciar perfis de clientes esta métrica avalia o quanto o modelo foi capaz de diferenciar os perfis.

1.5. Modelos

1.5.1. Linear Regression

A regressão linear é um modelo de machine learning que tem como objetivo de estimar um resultado a partir da correlação entre uma ou mais vaiáveis independentes continuas. O nome regressão vem da ideia de que todos os valores regridem para a média. Através de um plot de dispersão é traçada uma linha em que se minimiza os erros e esta linha é usada para estimar o valor para novas entradas.



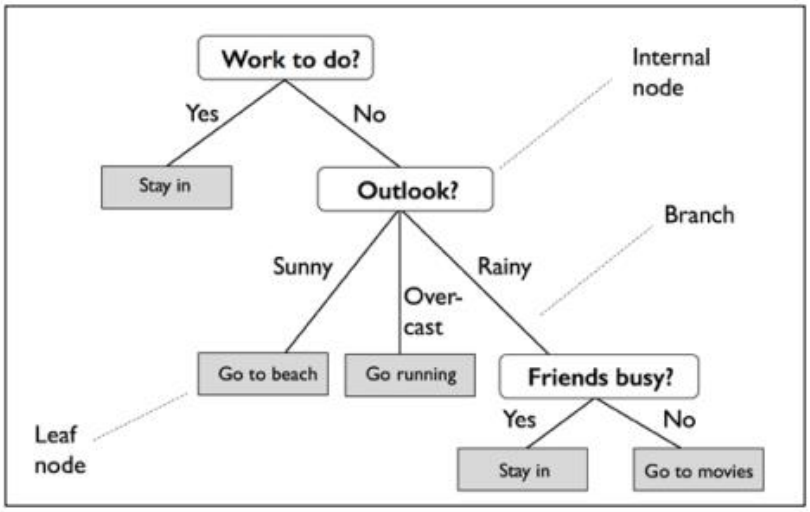
1.5.2. Logistic Regression

A regressão logística funciona de maneira muito parecida com a regressão comum, porém ela é otimizada para fazer a predição de dados categóricos e não contínuos.

1.5.3. Decision Tree

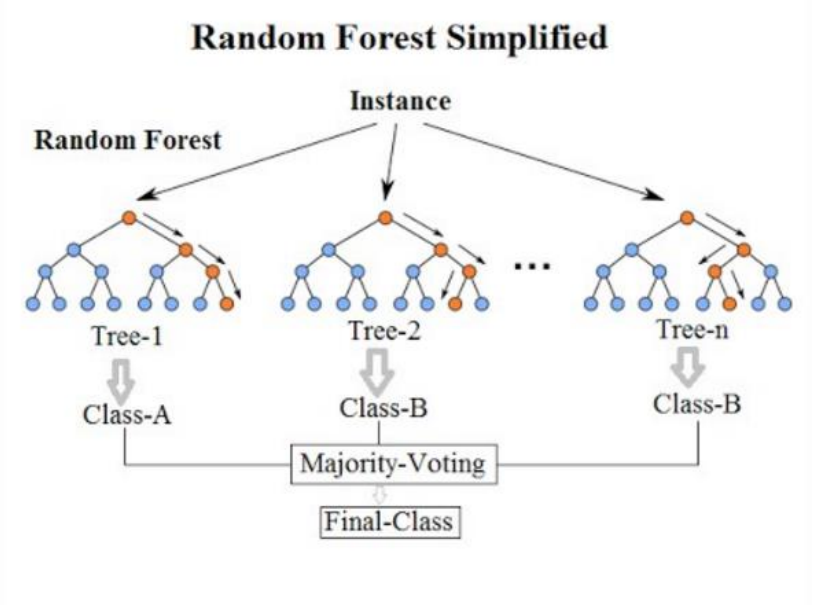
As pessoas usam árvores de decisão todos os dias para tomar decisões corriqueiras, na verdade podemos dizer que todas as decisões da vida de alguém faz parte de uma árvore de decisão.

Como exemplo, vamos considerar o exemplo abaixo em que uma pessoa decide qual atividade realizar em um dia se baseando em uma arvore de decisão.



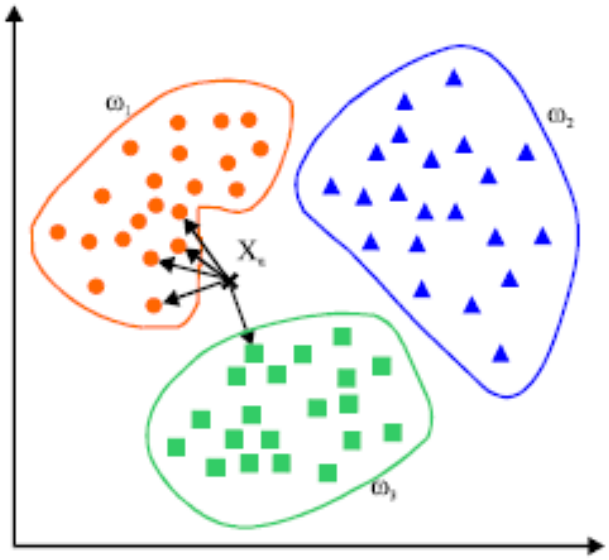
1.5.4. Random Forest

O random forest é uma floresta de arvores de decisão, ou seja, ele cria diversas arvores de decisão. Ele toma a decisão a partir da analise do resultado dessas arvores, veja uma demonstração abaixo.



1.5.5. KNN

O KNN funciona projetando os pontos em um plano e quando entramos com uma linha para predição o que ele faz é projetar este ponto no plano e analisar os vizinho mais próximos deste pronto, o que forma a vizinhança deste ponto. A partir da analise dos resultados da vizinhança ele faz a predição do ponto desejado. Veja uma demonstração abaixo.



No exemplo da imagem o ponto seria atribuído a vizinhança laranja, pois ele tem mais vizinhos próximos dentro desta vizinhança.

Para esse projeto o modelo em questão se mostrou muito ineficaz, pois devido a grande quantidade de registros e de variáveis ele foi muito lento, levando até 3 horas para rodar.

1.5.6. Naive Bayes

Este modelo funciona fazendo combinações de probabilidade. Ele calcula a probabilidade de cada variável em relação ao target. Para a predição ele faz a combinação das probabilidades de cada variável da predição e toma uma decisão.

1.5.7. XGB

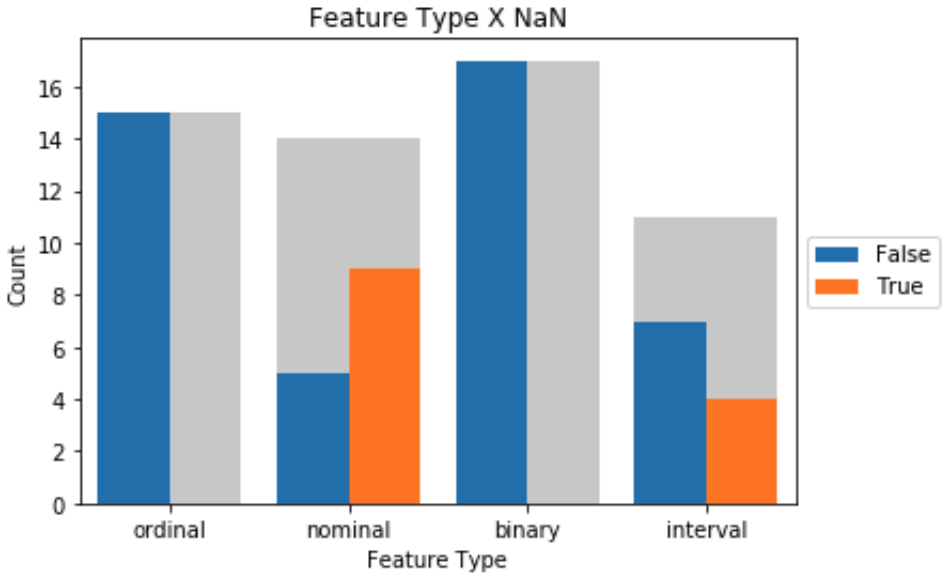
Da mesma forma que o random forest é uma combinação de arvores de decisão este modelo é uma combinação de random forests.

2. Projeto

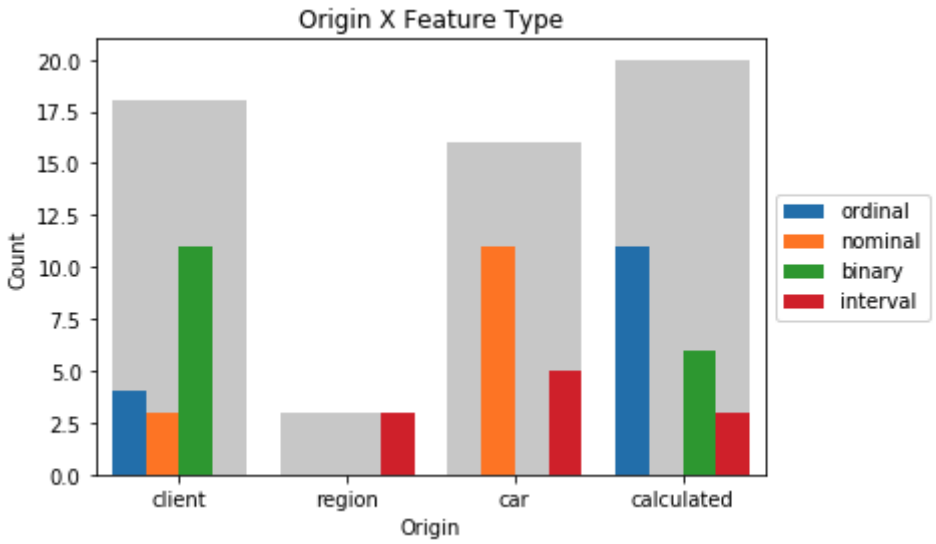
2.1. Entendendo os dados

A primeira coisa que fiz após importar os dados para um DataFrame foi fazer uma função para mapear os dados. Com essa função eu consegui saber o nome da coluna, a origem (cliente, calculado, etc), o tipo dos dados (binário, categórico), o tipo dos dados (int, float), se a coluna tem valores faltando, quantos valores estão faltando e qual a porcentagem de valores faltando. O resultado deste função foi um DataFrame onde foi possível visualizar melhor os dados de uma forma analítica.

O meu segundo passo foi retirar o “target” e o “id” deste DataFrame, e fazer um gráfico para analisar como estava a distribuição das colunas com dados faltando de acordo com o seu tipo. Sendo azul para colunas sem dados faltando e laranja para colunas com dados faltando.



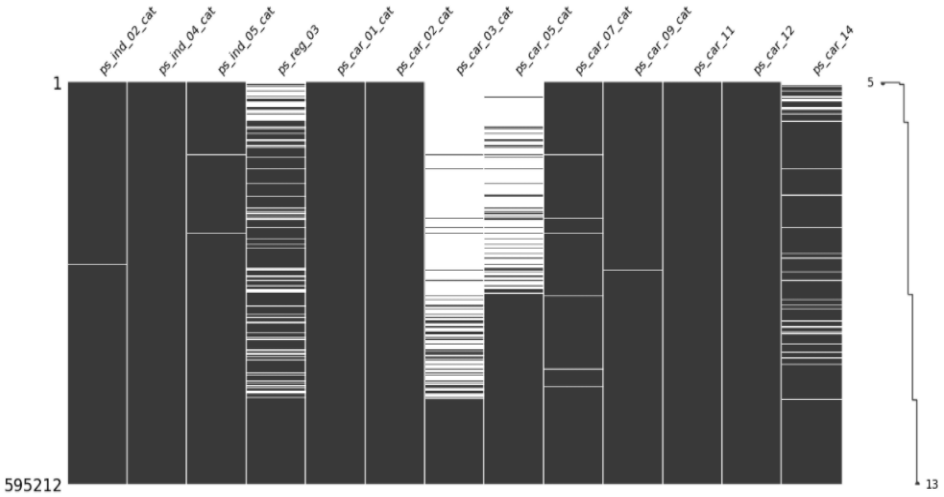
Após isso eu fiz um outro gráfico analisando qual o tipo de dados para cada tipo de origem.



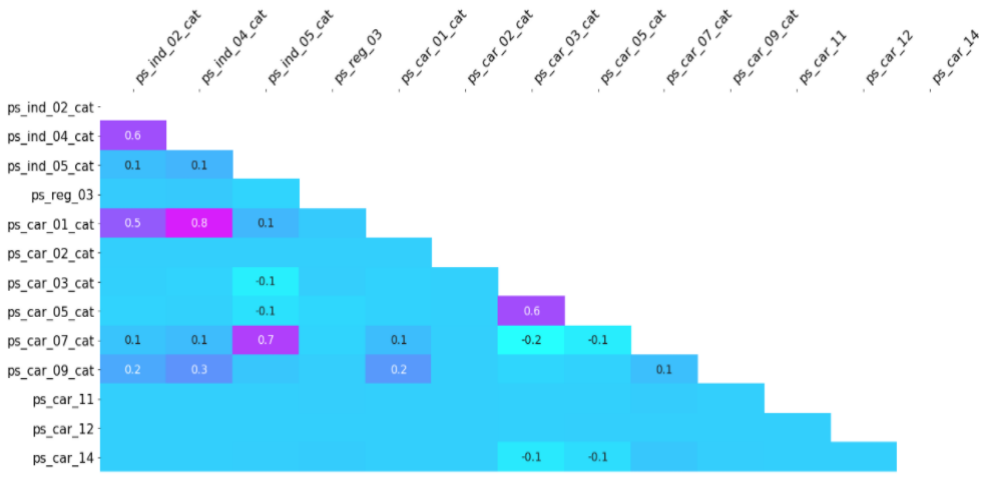
2.2. Entendendo as colunas com valores faltando

Para entender melhor as colunas com valores faltando eu comecei descobrindo quais são essas colunas, são elas: 'ps_ind_02_cat', 'ps_ind_04_cat', 'ps_ind_05_cat', 'ps_reg_03', 'ps_car_01_cat', 'ps_car_02_cat', 'ps_car_03_cat', 'ps_car_05_cat', 'ps_car_07_cat', 'ps_car_09_cat', 'ps_car_11', 'ps_car_12', 'ps_car_14'.

Com isso feito eu usei a biblioteca missingno para primeiro entender como estava a distribuição destes dados e quanto é a presença de cada um, o resultado foi:

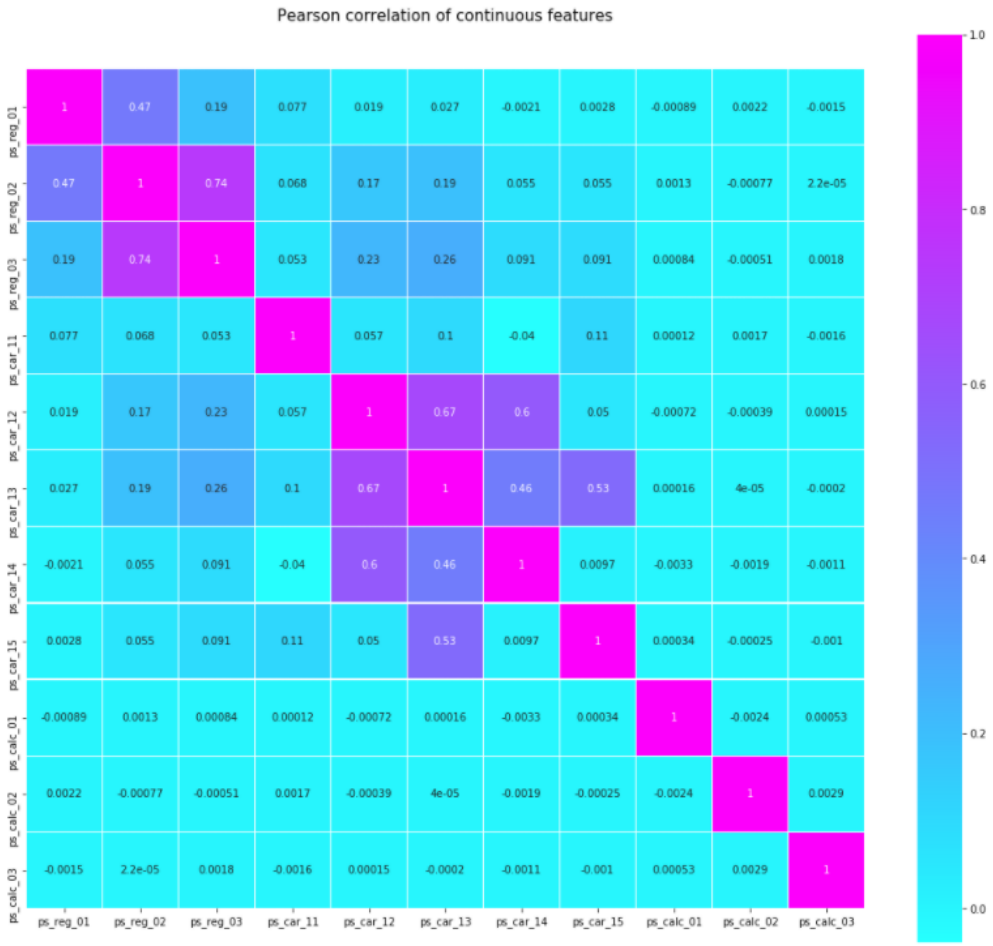


Após isso eu tentei entender como era a relação de dados faltando de uma coluna com outra:



2.3. Correlação entre os dados do tipo intervalo

Aqui eu fiz um mapa de calor da correlação entre os dados de intervalor, o resultado foi:



2.4. Importância de cada coluna

Agora para entender a importância de cada coluna na predição eu utilizei o Random Forest para conseguir essa informação. Após isso eu fiz um merge com a mapeamento dos dados para conseguir ter uma informação a mais sobre cada coluna.

2.5. Função de Cross Validation

Para avaliar os modelos eu criei uma função de Cross Validation que me retornava o score Gini de cada modelo, isso porque a métrica de avaliação da competição é o Gini. Eu fiz apenas com o Decision Tree e com o Naive Bayes porque os outros modelos eram muito demorados para rodar, principalmente o KNN.

Eu cheguei a conclusão de que a melhor maneira de avaliar os meus modelos era utilizar um processamento em nuvem para ser mais rápido e submeter meus modelos no Kaggle para ter um score mais preciso e foi isso que fiz. Ao todo foram mais de 80 tentativas que mais para frente irei demonstrar neste relatório.

2.6. Feature Engineering

2.6.1. Colunas com dados faltando

A coluna 'ps_car_05_cat' foi removida porque tinha 44,78% de dados faltando e a importância dela era de apenas 0,01. A coluna 'ps_car_03_cat' foi removida porque tinha 69,09% dos dados faltando. A coluna 'ps_ind_02_cat' foi removida porque tinha dados faltando e a importância era baixa.

As colunas categóricas eu utilizei Regressão Logística para prever os valores faltantes. O input foram as colunas categóricas que não tinham dados faltando. Como as colunas em questão tinham haver com o carro e com o individuo e estes dois estão diretamente relacionados eu achei prudente utilizar esta abordagem.

A coluna 'ps_reg_03' tem uma importância muito elevada e uma correlação alta com a coluna 'ps_reg_02', por isso eu utilizei uma Regressão Linear, usando a 'ps_reg_02' como input para prever os valores que estavam faltando da 'ps_reg_03'.

A coluna 'ps_car_12' tem uma alta correlação com a coluna 'ps_car_13', por isso usei uma Regressão Linear para prever os dados que estavam faltando com a coluna 'ps_car_13' como input.

As demais colunas como tinham um numero muito baixo de dados faltando eu substitui os campos com dados faltando pela média da coluna.

2.6.2. Nova coluna calculada

Uma das alterações com a qual consegui o maior aumento no meu score foi a criação de uma coluna calculada. Nesta coluna eu coloquei a soma de todas as colunas calculada que não eram binárias de cada linha.

Fiz isso porque pensei em criar um campo com um score para o cliente a partir das colunas calculadas.

2.6.3. Informações importantes para o revisor

Eu fiz inúmeros testes utilizando o OneHotEncoder e tratando das dummy's, mas todos os testes que fiz dessa forma apenas reduziram meu score ao invés de aumentar. Por isso na versão final eu faço apenas o Label Encoder que não muda em nada, mas deixei ali apenas para mostrar que tenho noção do que é que sei como usar.

2.7. Avaliando alguns modelos

Nessa parte o que eu fiz foi avaliar alguns modelos. Além dos modelos propostos na minha proposta de projeto eu resolvi experimentar o XGBClassifier, isso porque durante a minha pesquisa eu conclui que ele poderia atender bem a minha expectativa. Para essa avaliação eu utilizei uma função de Cross Validation. Os scores foram:

Decision Tree – 0,01209
Naive Bayes – 0,21898
Random Forest – 0,08044
XGB – 0,25848
KNN – 0.05001

Com essas informações eu decidi por seguir adiante usando o XGBClassifier.

É importante ressaltar que esse não foi a única avaliação que fiz, sempre que eu tentava algo novo em relação aos dados ou alterações nos modelos eu rodava um novo script na nuvem para testar. Outro detalhe importante é que cheguei a conclusão de que a melhor maneira de obter um score correto era gerar um arquivo de submissão para cada tentativa e submeter no Kaggle para ter o score exato que eu receberia na competição. Para ver o que foi feito em cada avaliação é só entrar no link: https://www.floydhub.com/josecbf/projects/final-project-udacity

Abaixo a minha planilha de acompanhamento das minhas tentativas:

Date	Submission	Version	Model	Treatment	Improviments in model	Score
01/02/2018	0	0	Naive Bayes	Yes	No	0,21657
01/02/2018	1	0	Decision Tree	No	No	0,01731
01/02/2018	1	0	Decision Tree	Yes	No	0,01731
01/02/2018	1	0	Naive Bayes	No	No	0,22358

01/02/2018	1	0	Naive Bayes	Yes	No	0,22358
01/02/2018	1	0	Random Forest	No	No	0,07062
01/02/2018	1	0	Random Forest	Yes	No	0,07062
02/02/2018	2	0	Random Forest	No	Yes	0,25652
02/02/2018	2	0	Random Forest	Yes	Yes	0,25010
02/02/2018	2	1	Naive Bayes	Yes	Yes	0,22523
02/02/2018	2	1	Decision Tree	Yes	Yes	0,19566
04/02/2018	2	1	Random Forest	Yes	Yes	0,24910
05/02/2018	3	0	Random Forest	No	Yes	0,25623
05/02/2018	3	0	Random Forest	Yes	Yes	0,24923
05/02/2018	3	0	KNN	No	Yes	0,03883
05/02/2018	3	0	KNN	Yes	Yes	0,03605
05/02/2018	3	1	Random Forest	Yes	Yes	0,25383
05/02/2018	3	1	Random Forest	No	Yes	0,25383
05/02/2018	3	2	Random Forest	Yes	Yes	0,25715
05/02/2018	3	2	Random Forest	No	Yes	0,25715
05/02/2018	3	3	Random Forest	Yes	Yes	0,25709
05/02/2018	3	3	Random Forest	No	Yes	0,25709
06/02/2018	3	3	Random Forest	Yes	Yes	0,25647
06/02/2018	3	3	Random Forest	No	Yes	0,25647
06/02/2018	4	0	Random Forest	Yes	Yes	0,23658
06/02/2018	4	0	XGB	Yes	Yes	0,24823
06/02/2018	6	0	Random Forest	Yes	Yes	0,23391
06/02/2018	6	0	XGB	Yes	Yes	0,25409
06/02/2016	6	1	Random Forest	Yes	Yes	0,25630
06/02/2016	6	2	Random Forest	Yes	Yes	0,26098
09/02/2016	6	2	XGB	Yes	Yes	0,27806
09/02/2018	8	2	XGB	Yes	Yes	0,24513
09/02/2018	8	2	Random Forest	Yes	Yes	0,22287
09/02/2018	8	3	XGB	Yes	Yes	0,24513
09/02/2018	8	3	Random Forest	Yes	Yes	0,22287
09/02/2018	8	5	XGB	Yes	Yes	0,27686
09/02/2018	8	5	Random Forest	Yes	Yes	0,25611
09/02/2018	8	6	XGB	Yes	Yes	0,27686
09/02/2018	8	6	Random Forest	Yes	Yes	0,25128
10/02/2018	9	0	XGB	Yes	Yes	0,27760
10/02/2018	9	0	Random Forest	Yes	Yes	0,24907
10/02/2018	10	0	XGB	Yes	Yes	0,27679
10/02/2018	10	0	Random Forest	Yes	Yes	0,25657
10/02/2018	10	1	XGB	Yes	Yes	0,27806
10/02/2018	10	1	Random Forest	Yes	Yes	0,25925
10/02/2018	10	2	XGB	Yes	Yes	0,27783

10/02/2018	10	2	Random Forest	Yes	Yes	0,25882
10/02/2018	10	4	XGB	Yes	Yes	0,27823
10/02/2018	10	4	Random Forest	Yes	Yes	0,25870
10/02/2018	10	5	XGB	Yes	Yes	0,27871
10/02/2018	10	5	Random Forest	Yes	Yes	0,25875
10/02/2018	10	6	XGB	Yes	Yes	0,27772
10/02/2018	10	6	Random Forest	Yes	Yes	0,25819
10/02/2018	11	0	XGB	Yes	Yes	0,26531
10/02/2018	11	0	Random Forest	Yes	Yes	0,24852
11/02/2018	11	1	XGB	Yes	Yes	0,26487
11/02/2018	11	1	Random Forest	Yes	Yes	0,24884
11/02/2018	11	2	XGB	Yes	Yes	0,26694
11/02/2018	11	2	Random Forest	Yes	Yes	0,25243
11/02/2018	11	3	XGB	Yes	Yes	0,27623
11/02/2018	11	3	Random Forest	Yes	Yes	0,25079
11/02/2018	11	4	XGB	Yes	Yes	0,27327
11/02/2018	11	4	Random Forest	Yes	Yes	0,25617
11/02/2018	11	5	XGB	Yes	Yes	0,27468
11/02/2018	11	6	XGB	Yes	Yes	0,27365
11/02/2018	12	0	XGB	Yes	Yes	0,20186
11/02/2018	12	1	XGB	Yes	Yes	0,23725
11/02/2018	12	2	XGB	Yes	Yes	0,25289
11/02/2018	12	3	XGB	Yes	Yes	0,23385
11/02/2018	12	4	XGB	Yes	Yes	0,25666
11/02/2018	12	5	XGB	Yes	Yes	0,27676
11/02/2018	12	6	XGB	Yes	Yes	0,27676
11/02/2018	12	7	XGB	Yes	Yes	0,27676
11/02/2018	12	8	XGB	Yes	Yes	0,27693
11/02/2018	12	9	XGB	Yes	Yes	0,27716
11/02/2018	13	0	XGB	Yes	Yes	0,27867
11/02/2018	13	1	XGB	Yes	Yes	0,27871
11/02/2018	13	2	XGB	Yes	Yes	0,27851
12/02/2018	13	4	XGB	Yes	Yes	0,27625
12/02/2018	14	0	XGB	Yes	Yes	0,27861
12/02/2018	14	1	XGB	Yes	Yes	0,27442
12/02/2018	14	1	XGB	Yes	Yes	0,27625
12/02/2018	14	1	Random Forest	Yes	Yes	0,24227
12/02/2018	14	2	XGB	Yes	Yes	0,25951
12/02/2018	14	2	XGB	Yes	Yes	0,26088
12/02/2018	14	3	XGB	Yes	Yes	0,27488
12/02/2018	14	3	XGB	Yes	Yes	0,27690

12/02/2018	15	0	XGB	Yes	Yes	0,27871
12/02/2018	15	0	XGB	Yes	Yes	0,27867
12/02/2018	15	0	XGB	Yes	Yes	0,27720
12/02/2018	15	0	XGB	Yes	Yes	0,27840
12/02/2018	15	0	XGB	Yes	Yes	0,27748
12/02/2018	15	0	XGB	Yes	Yes	0,27446
12/02/2018	15	0	XGB	Yes	Yes	0,27368
12/02/2018	15	1	XGB	Yes	Yes	0,27871
12/02/2018	15	1	XGB	Yes	Yes	0,22218
12/02/2018	15	1	XGB	Yes	Yes	0,27647
12/02/2018	15	1	XGB	Yes	Yes	0,27812
12/02/2018	15	1	XGB	Yes	Yes	0,27500
12/02/2018	15	1	XGB	Yes	Yes	0,27713
12/02/2018	15	1	XGB	Yes	Yes	0,27634
12/02/2018	15	1	Naive Bayes	Yes	Yes	0,22221
12/02/2018	16	0	XGB	Yes	Yes	0,28206

2.8. Melhorando o modelo

Para melhorar o meu modelo eu usei Grid Search através de uma função. Vale ressaltar que o Grid Search completo que utilizei não está no notebook, pois ele levou mais de 10 horas para rodar e fiz ele na nuvem, como isso custa e não é barato eu optei por não reproduzir no notebook. Porém os parâmetros ideais encontrados foram:

```
params['learning_rate'] = 0.02
params['n_estimators'] = 1000
params['max_depth'] = 4
params['subsample'] = 0.9
params['colsample_bytree'] = 0.9
```

2.9. Submissão Kaggle

Para fazer a submissão eu criei uma função que faz o treino e a predição e salva a predição junto com o “id” em um arquivo csv no formato para fazer a submissão no Kaggle.

3. Conclusão

Após todo este trabalho minha submissão final ficou um score de 0,28206 o que na minha opinião foi bom levando em consideração que essa foi a minha primeira participação e que o primeiro colocado está com um score 0,29698. A impostancia de cada coluna no modelo final pode ser vista neste link:https://docs.google.com/spreadsheets/d/1GsFEQNtra2LkCGu2pCdDoASH4i-LzP0zc4W7rJLGS2w/edit?usp=sharing

Acredito que o grande segredo está no Feature Engeneering, pois foi quando trabalhei com isso que o meu score começou a subir. Acredito que

ainda tenho muito a estudar e aprender, porém para começo fiquei satisfeito com o meu score. Para melhorar meu score pretendo fazer 3 coisas, são elas:

- Melhorar o tratamento das colunas, porém para isso precisei me aprofundar um pouco mais.
- Tentar usar o modelo LigthGBM
- Usar a técnica de Staking

4. Referencias

4.1. Bibliotecas

- pandas
- numpy
- time
- sklearn.tree - DecisionTreeClassifier
- sklearn.ensemble - RandomForestClassifier
- sklearn.naive_bayes - GaussianNB
- sklearn.neighbors - KNeighborsClassifier
- xgboost import XGBClassifier
- sklearn.model_selection - train_test_split
- sklearn.model_selection - StratifiedKFold
- sklearn.model_selection - cross_val_score
- sklearn.linear_model - LinearRegression
- sklearn.linear_model - LogisticRegression
- sklearn.preprocessing - StandardScaler
- xgboost - XGBClassifier
- sklearn.preprocessing - LabelEncoder
- sklearn.grid_search - GridSearchCV
- seaborn
- matplotlib.pyplot
- missingno
- IPython.display - set_matplotlib_formats

4.2. Repositórios

- https://github.com/felipeeeantunes/udacity_live
- https://github.com/xiaozhouwang/kaggle-porto-seguro
- https://github.com/dryguz/Porto/blob/master/script.py
- https://github.com/wkirgsn/porto-seguro