Relatório da Atividade Prática 2 de Linguagem da Programação:

Filipe Viana[1], Thomas Amorim[2], Reginaldo Santos[3]
Universidade Federal do Rio de Janeiro (UFRJ)

reginaldosantos1333@poli.ufrj.br,
thomas.jesus@poli.ufrj.br,
filipe.vianasilva@poli.ufrj.br

Resumo. O desafio "Cat in the Dat" desafia o cientista de dados a analisar uma base de dados que está repleto de variáveis categóricas (binária, nominal, ordinal e cíclica – dia, horas, minutos). O problema a ser solucionado nesse caso é fazer com que haja uma codificação que seja capaz de fazer com que todas essas variáveis possam ser lidas pelos algoritmos e então executadas de forma satisfatória, a fim de devolver um resultado.

Desenvolvimento:

```
#Todos os import's necessários para o funcionamento do código
from matplotlib import style
from IPython.display import display
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
import seaborn as sns
from scipy import sparse, stats
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.model_selection import train_test_split, StratifiedKFold, GridSearchCV
from sklearn.metrics import accuracy_score, roc_auc_score
from sklearn.linear_model import LogisticRegressionCV
from xgboost import XGBClassifier
```

Primeiramente, após certa pesquisa percebemos a necessidade de utilizar diversos programas que nos auxiliassem à analisar e demonstrar esses dados, através de gráficos – por exemplo, como o Pandas, Xgboost, Seaborn, entre outros.

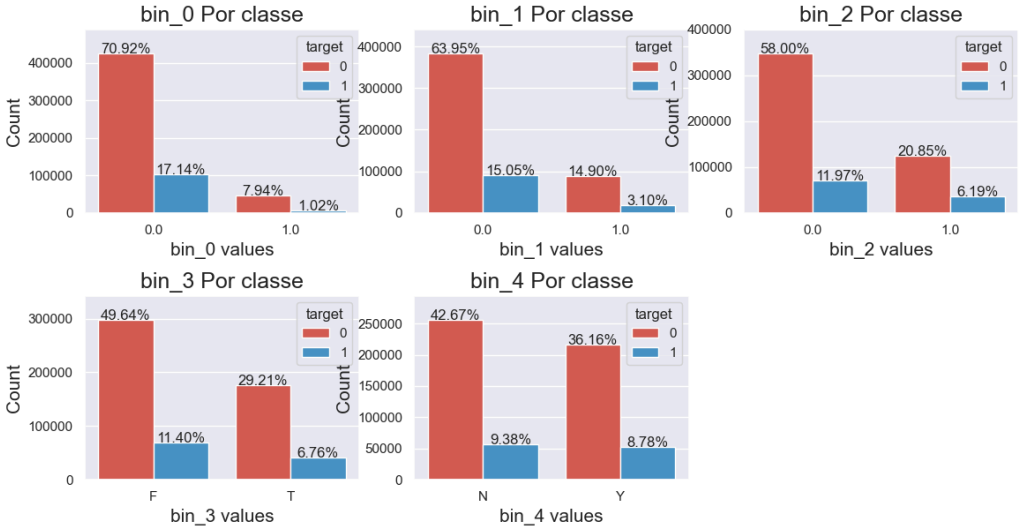
Logo em seguida, começamos o tratamento dos dados, procurando ter uma visão geral sobre os mesmos e entender a divisão dos dados no arquivo "train.csv", que serve como uma forma de treinar o algoritmo para aprender a se comportar com diferentes valores.

Algumas variáveis possuem níveis de cardinalidade maiores que as outras, sendo a nominal disparadamente a maior em tal quesito. Foi percebido também, que quanto mais alto o nível de cardinalidade de uma variável, maior será seu valor.

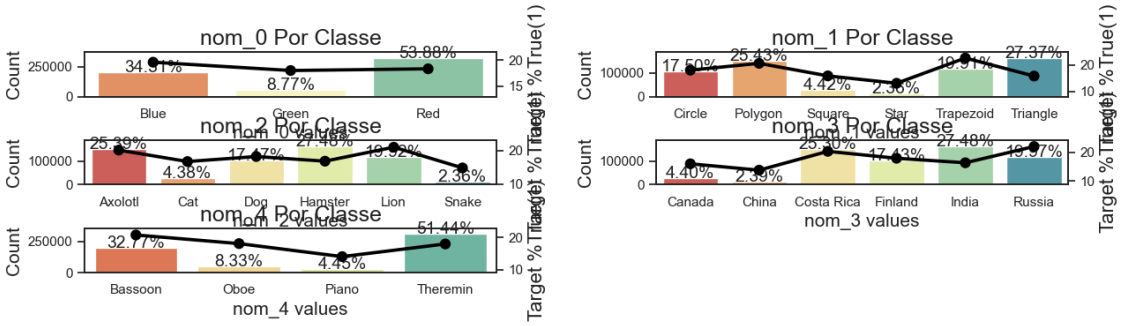
É importante se atentar que pelo número expressivo de valores únicos, e fatores ante mencionados: pode ser que alguns dados encontrados no treino não estejam presentes nos dados de teste e vice-versa. O problema desse modelo é que o código está treinado sobre um pré-determinado conjunto de classes, e avaliado com o que não foi visto. Isso é muito crítico para o desempenho do sistema.

No caso do problema criado pelo Kaggle, é importante observar que a variável nom_9 é bastante problemática, pois possuí valores únicos espalhados entre as bases de dados de treino e teste.

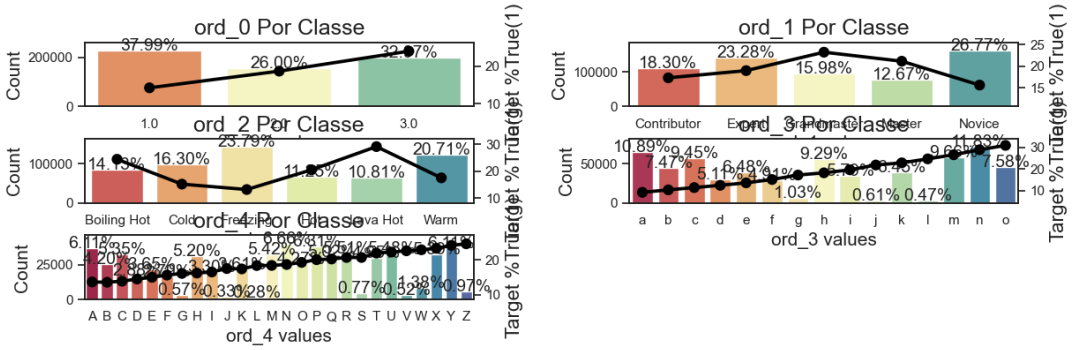
Os dados estavam bastante desbalanceados, enquanto cerca de 81% representavam"0", apenas 19% representavam "1". Nos quais os valores binários são divididos em alguns grupos, são eles: 0/1, Y/N e T/F. Todos estes valores são balanceados de forma cuidadosa, e apenas a variável bin_1 se destaca como de maior diferença entre as classes de variável target.



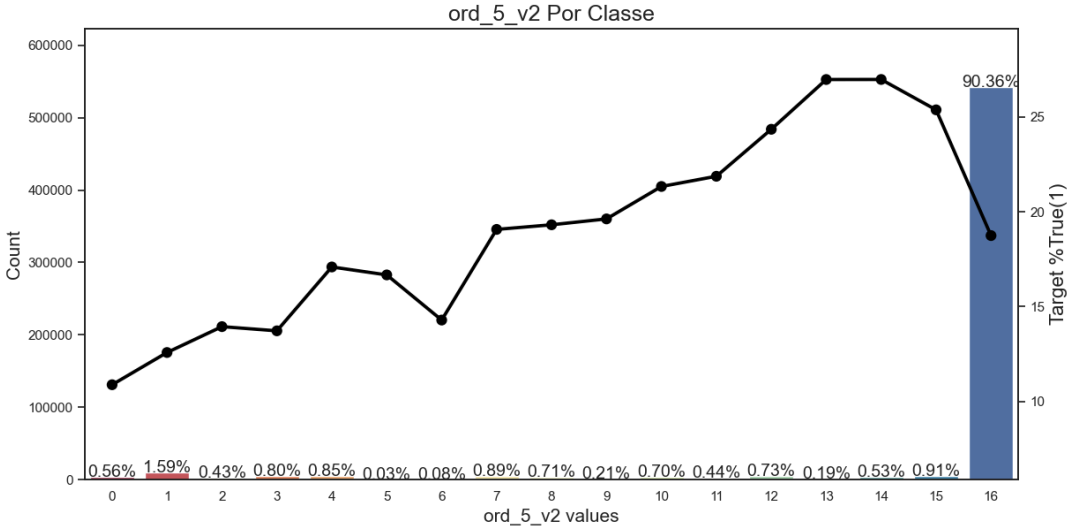
O gráfico abaixo mescla 2 estilos diferentes de plotagem, a plotagem de ponto com o gráfico de contagem de plotagem, com o objetivo de entender o comportamento de algumas das variáveis nominais e as ocorrências de valores "1" naquela variável.



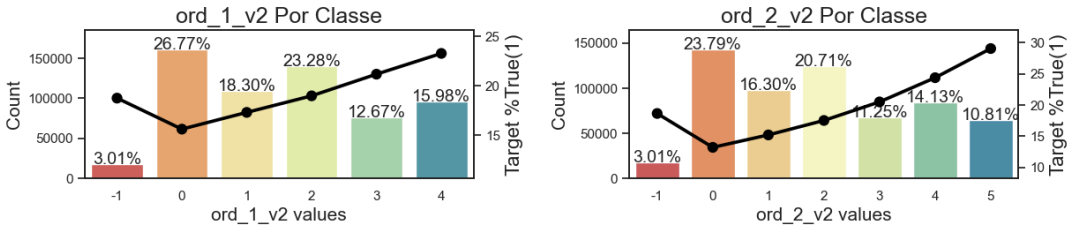
É interessante perceber que os valores True são maiores nas variáveis nominais que possuem menor ocorrência no grupo. Não foram feitas as plotagens de variáveis nominais de maiores que nom_4 pois serão de difícil visualização.



Por causa da organização das variáveis ordinais, podemos identificar uma ordem de importância entre os dados. O que pode ser exemplificado no “alfabeto” formado na ord_3 e ord_4. Entretanto, podemos perceber que tanto ord_1, como ord_2, parecem não seguir o mesmo esquema que os outros.



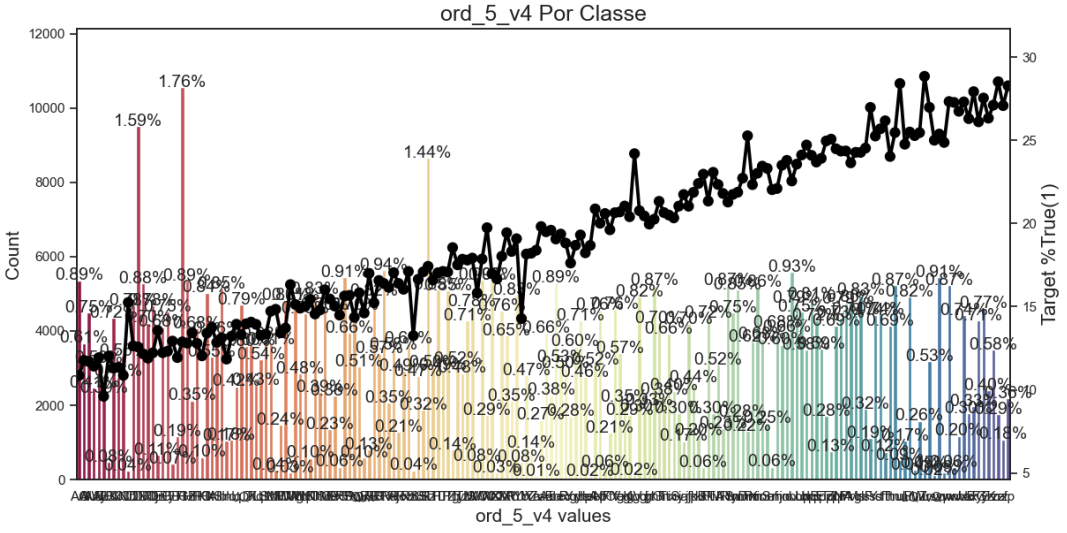
O motivo é que as variáveis não estão ordenadas pelo que de fato elas representam, e sim por ordem alfabética. Vamos visualizar então quando organizamos as variáveis ord_1 e ord_2 por uma ordem específica e para isso definimos algumas funções.



Reorganizando os dados dessa ordenadas, podemos observar que elas também têm o mesmo funcionamento das outras ordenadas.

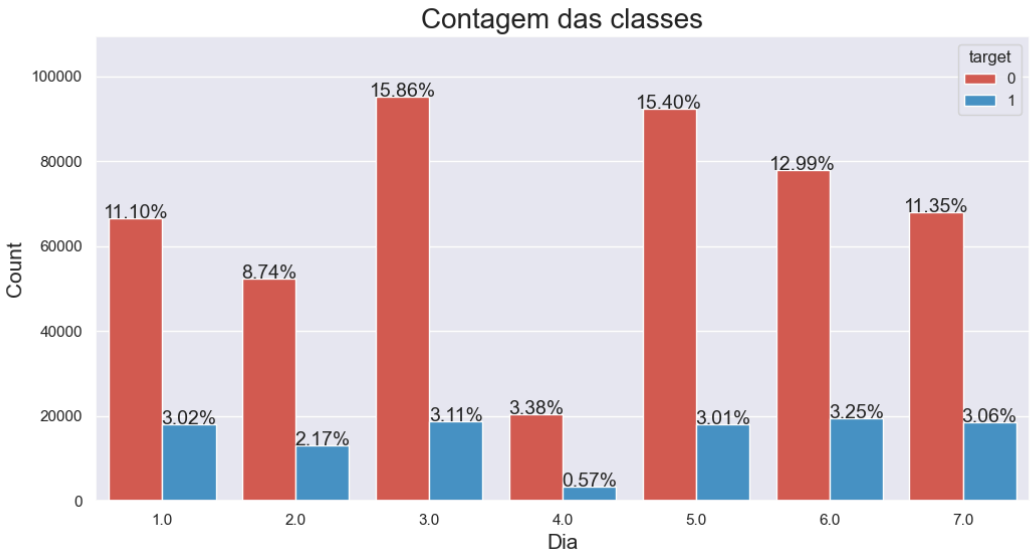
A ordenação_5 tem o comportamento mais aleatório em relação às outras variáveis ordinais. O que pode significar que uma junção de variáveis está provocando essa instabilidade, ou que a própria variável possuí esse ruído.

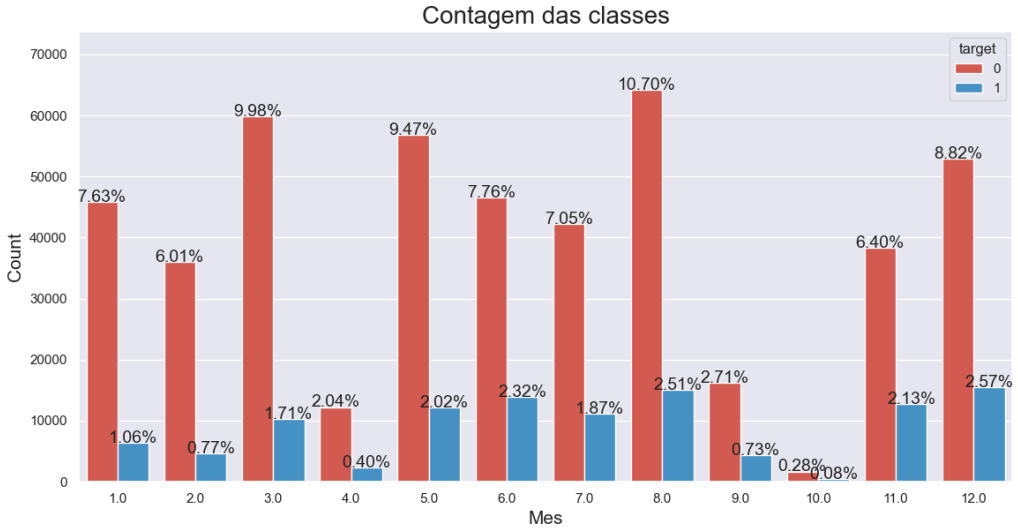
Podemos então testar a primeira suposição, e realizar a separação da *ord_5* em novas duas variáveis (através de listas). Onde a primeira coluna diz respeito ao primeiro caractere de cada amostra, e a segunda coluna ao segundo caractere de cada amostra.



Quando ocorre essa separação, uma dessas novas variáveis – a que pode ser vista acima - mostrou um padrão muito mais linear (como as ordenadas anteriores) que a original. Ao passo que, a outra variável criada apresenta apenas ruído, o que era a responsável por prejudicar o entendimento da ord_5.

Vamos analisar agora os dados de Dia e Mês, que também foram disponibilizados nos arquivos enviados pelo Kaggle.





A distribuição de valores 0 e 1 não apresentam muito desbalanceamento, permanecendo dentro do esperado. Podemos nos atentar também, que tanto no Dia – como no Mês – 4, ocorre uma baixa considerável de 1 e 0 comparado ao restante dos dados.

Com todos os dados acessados e organizados, será feito o pré-processamento deles para alcançarmos os resultados desejados. Dessa forma, usufruiremos de 2 principais técnicas

de tratamento de variáveis categóricas. O Label Encoder e o One Hot Encoder disponíveis no Scikit Learn e no Pandas, respectivamente.

Sobre o tratamento com as variáveis binárias: como as variáveis bin_0, bin_1 e bin_2 já tem valores próprios, será preciso alterar apenas as variáveis restantes – bin_3 e bin_4.

Já ao se tratar dos meses do ano, seguindo algumas indicações nas referências que conferimos, decidimos dividir os dados em quatro trimestres. Que serão referenciados como: *Q1*, *Q2*, *Q3*, *Q4* (*quarters* – equivalente a trimestres em português).

"Ao usar o método do "One Hot Encoder" em toda a matriz, a quantidade de memória utilizada é muito grande. Para minimizar isso podemos realizar a criação de uma matriz esparsa, que justamente é especializada no armazenamento de uma matriz que possui grandes quantidades de zeros com valores esparsados ao longo de sua dimensão."
-Lobdata

Através de todos esses passos, a pontuação mais confiável foi verificada pelo método da Regressão Logística, com o valor de 78,19% na métrica AUC (traduzida do inglês livremente, como "área sob a curva") que é utilizada para avaliação do método.

Referências:

https://pt.stackoverflow.com/
https://www.kaggle.com/warkingleo2000/first-step-on-kaggle
https://www.lobdata.com.br/2019/11/04/categorical-feature-encoding-challenge/
https://www.kaggle.com/learn/intro-to-machine-learning

https://www.kaggle.com/dansbecker/how-models-work
https://www.kaggle.com/c/cat-in-the-dat
https://www.kaggle.com/kabure/eda-feat-engineering-encode-conquer
https://www.kaggle.com/adaubas/2nd-place-solution-categorical-fe-callenge
https://medium.com/bio-data-blog/entenda-o-que-%C3%A9-auc-e-roc-nos-modelos-de-machine-learning-8191fb4df772