Relatório de preparação dos dados

Alunos:
André Luiz Pereira da Silva (alps2@cin.ufpe.br)

Para o preparo dos dados, foi criada uma função “load_data”, como pode ser visto a seguir:

```python
from sklearn.model_selection import train_test_split
import pandas

RANDOM_SEED = 1337

def load_data(file_path: str, positive_label_multiplication: int = 2) -> pandas.DataFrame:
    """ Receives a file path for the dataset training, testing and validation datasets. """

    # Loading data from csv file
    df = pandas.read_csv(file_path)

    # Selecting useful features
    useful_features = [
        "Bidder_Tendency",
        "Bidding_Ratio",
        "Successive_Outbidding",
        "Last_Bidding",
        "Auction_Bids",
        "Starting_Price_Average",
        "Early_Bidding",
        "Winning_Ratio",
        "Auction_Duration",
        "Class"
    ]

    df = df[useful_features]

    # Augmenting positive label data
    positive_labels = df[df["Class"] == 1]

    dfs_to_concat = [df]
    for _ in range(positive_label_multiplication):
        dfs_to_concat.append(positive_labels)

    df = pandas.concat(dfs_to_concat)
    df = df.sample(frac=1)

    # Separating features and labels
    columns = list(df.columns)
    features = columns[:len(columns)-1]
    label = columns[len(columns)-1:]

    X = df[features]
    y = df[label]

    # Creating training, testing and validation datasets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state = RANDOM_SEED)
    X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train, test_size = 0.125, random_state = RANDOM_SEED)

    return X_train, y_train, X_test, y_test, X_valid, y_valid
```

Durante o EDA, checamos o dataset para verificar casos específicos. Embora o dataset já esteja em ótimo estado (sem valores nulos ou duplicados, valores numéricos consistentes e sem outliers), ainda é preciso tomar algumas medidas para que os dados possam ser consumidos por um classificador. Cada passo da função será explicado neste relatório.

1. Feature selection

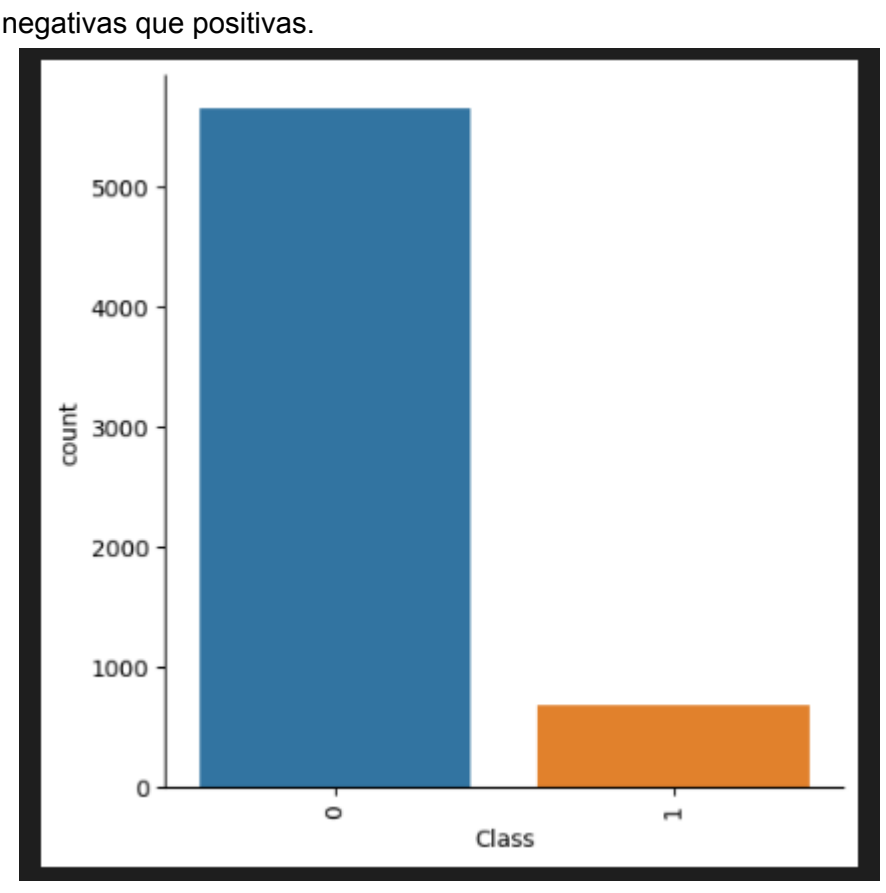
A primeira etapa da função é uma seleção de features úteis. Foi determinado que os atributos Record_ID, Auction_ID e Bidder_ID representam identificadores dos registros e não seriam relevantes para a classificação, dessa forma mantendo apenas os atributos:

- Bidder_Tendency
- Bidding_Ratio
- Successive_Outbidding
- Last_Bidding
- Auction_Bids
- Starting_Price_Average
- Early_Bidding
- Winning_Ratio
- Auction_Duration
- Class

Embora o Bidder_ID pudesse ser uma informação útil (já que identifica uma conta, e assumindo que uma conta que já foi marcada com comportamento fraudulento anteriormente, provavelmente vai infringir as regras da plataforma novamente), ainda assim foi cortado, pois criar uma conta nova na plataforma é simples. Dessa forma, indivíduos desejando burlar as regras sempre teriam Bidder_ID’s novos quando necessário.

2. Data Augmentation

Como visto no relatório de análise exploratória, o dataset está desbalanceado, com muito mais labels negativas que positivas.



Distribuição dos rótulos no dataset ShillBidding

Isso pode ser muito ruim para o treinamento do modelo, que pode acabar enviesado a classificar sempre de forma negativa, já que, estatisticamente falando, esse rótulo ocorre mais vezes.

Para sanar isso, incluímos a possibilidade de multiplicar os registros de labels positivas presentes no dataset, por meio do atributo “positive_label_multiplication“. Caso o valor passado seja 0, não será aplicado data augmentation no dataset. Por padrão, multiplicamos esses rótulos 3 vezes.

Com isso, passamos um sério risco de gerarmos modelos sofrendo de overfitting, já que o treinamento estaria muito contido no universo inicial de dados com classificação positiva. Contudo, como vimos no relatório, a maior parte dos atributos numéricos possui uma discrepância de valor muito grande entre rótulos positivos e negativos, nos levando a crer que vale a pena o risco.

3. Training, testing and validation sets

Por último, separamos o dataset completo em 3 conjuntos menores, de treinamento, teste e validação (seguindo a escala “70% - 20% - 10%”, respectivamente). Utilizamos uma semente aleatória fixa, para garantir reprodutibilidade dos resultados.