UNIVERSIDADE FEDERAL DO TOCANTINS
CIÊNCIA DA COMPUTAÇÃO
APRENDIZADO DE MÁQUINA

NATHAN MACHADO DOS SANTOS

Trabalho Final
Técnicas de aprendizado de máquina

Palmas, Junho de 2023

1. Introdução

O dataset escolhido por mim para ser analisado posteriormente foi o "Stellar Classification Dataset", que se trata de um dataset para análise de dados espaciais, com o intuito de classificar os objetos em 3 conjuntos: estrelas, galáxias e quasares, e que pode ser encontrado no seguinte link: https://www.kaggle.com/datasets/fedesoriano/stellar-classification-dataset-sdss

O dataset consiste em 100 mil linhas de observações feitas pelo SDSS (Sloan Digital Sky Survey). Existem 17 colunas de características e uma coluna de classe que identifica o objeto como estrela, galáxia ou quasar.

Dicionário de dados:

- obj_ID = Identificador do objeto, valor único que identifica o objeto no catálogo.
- alpha = Ângulo de Ascensão Reto (no momento J2000)
- delta = Ângulo de Declinação (no momento J2000)
- u = Filtro ultravioleta no sistema fotométrico
- g = Filtro verde no sistema fotométrico
- r = Filtro vermelho no sistema fotométrico
- i = Filtro de infravermelho próximo no sistema fotométrico
- z = Filtro de infravermelho no sistema fotométrico
- run_ID = Identificador de execução usado para identificar a varredura específica
- rereun_ID = Identificador de reexecução para especificar como a imagem foi processada
- cam_col = Coluna da câmera para identificar a linha de varredura dentro da execução
- field_ID = Número de campo para identificar cada campo
- spec_obj_ID = Identificador exclusivo usado para objetos espectroscópicos ópticos (isso significa que duas observações diferentes com o mesmo spec_obj_ID devem compartilhar a classe de saída)

● class = classe do objeto (galáxia, estrela ou quasar)

● redshift = valor de desvio para o vermelho baseado no aumento do comprimento de onda

● plate = ID do prato, identifica cada prato no SDSS

● MJD = Data Juliana Modificada, usada para indicar quando um determinado dado do SDSS foi coletado

● fiber_ID = ID da fibra que apontou a luz para o plano focal em cada observação

2. Metodologia

Como pode-se observar pelo dicionário de dados dado na introdução, há muitas colunas presentes no dataset, e grande parte delas não é relevante para ajudar na classificação dos objetos espaciais. Assim sendo, o primeiro passo adotado foi um estudo dos dados para ver como eles estão correlatos.

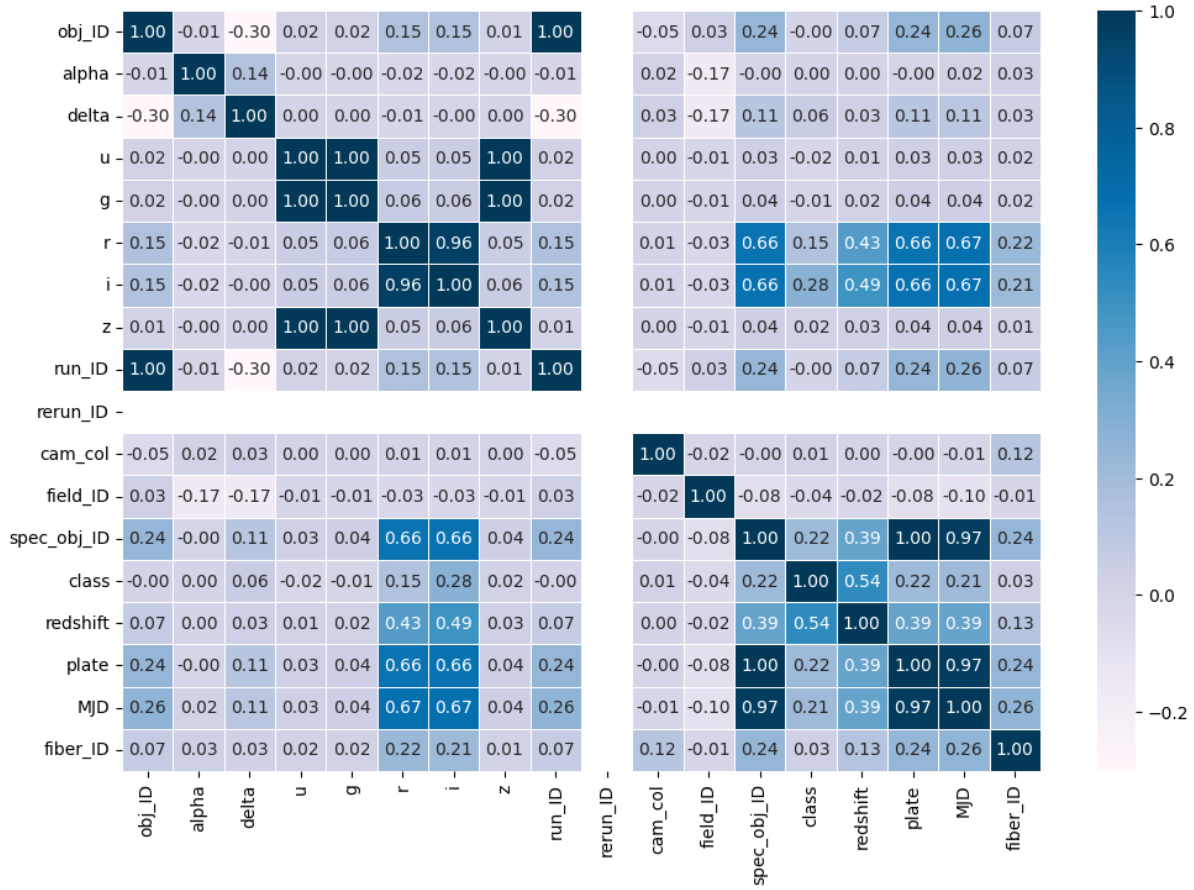

Figura 1. Gráfico de correlação

Dado o gráfico de correlação acima, gerado por um heatmap do módulo seaborn, é possível observar quais valores têm correlação com a classe do objeto estelar, e pode-se fazer uma limpeza das colunas que pouco, nada ou negativamente influenciam na classificação. As colunas removidas foram as seguintes:

'obj_ID', 'alpha', 'delta', 'run_ID', 'rerun_ID', 'cam_col', 'field_ID', 'fiber_ID'

Isto posto, prosseguimos para a detecção e remoção de outliers, onde foi utilizado o método interquartil, como demonstrado pelo trecho de código abaixo.

```
# iterar pelas colunas do DataFrame
for col in df.columns:
    # calcular o limite superior e inferior para detecção de outliers
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lim_sup = Q3 + 1.5 * IQR
    lim_inf = Q1 - 1.5 * IQR
    #dropa outliers
    df.drop(df[df[col]<lim_inf].index, inplace=True)
    df.drop(df[df[col]>lim_sup].index, inplace=True)
```

Figura 2. Interquartil para detecção de outliers

O modelo utilizado foi o MLPClassifier, utilizando-se de duas camadas ocultas, sendo a primeira com 10 neurônios e a segunda com 20 neurônios, o número máximo de interações foi definido como 100.

```
#classifica as colunas
x = df[["u", "g", "r", "i", "z", "spec_obj_ID", "redshift", "plate", "MJD"]]
y = df['class']

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.5, random_state=12)
✓ 0.0s


#Escala os valores
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

#Treina o multilayer perceptron classificador e realiza os teste
clf = MLPClassifier(
    hidden_layer_sizes=(10, 20),
    random_state=5,
    learning_rate_init=0.01,
    max_iter=100,
)

clf.fit(X_train_scaled, y_train)

y_pred = clf.predict(X_test_scaled)
```

Figura 3. Código do Multilayer Perceptron Classificador

3. Conclusão

Dado o modelo acima, foi possível adquirir bons resultados, com uma acurácia de cerca de 97% e um R2 Score de 0.79, como demonstrado na Figura 3. Foi gerado também uma matriz de confusão para melhor visualização dos resultados, que pode ser observada na Figura 4.

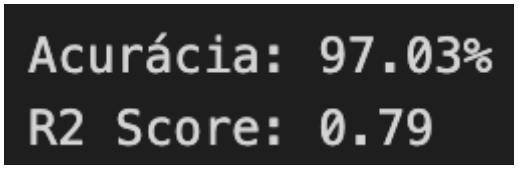

Figura 3. Acurácia e R2 Score

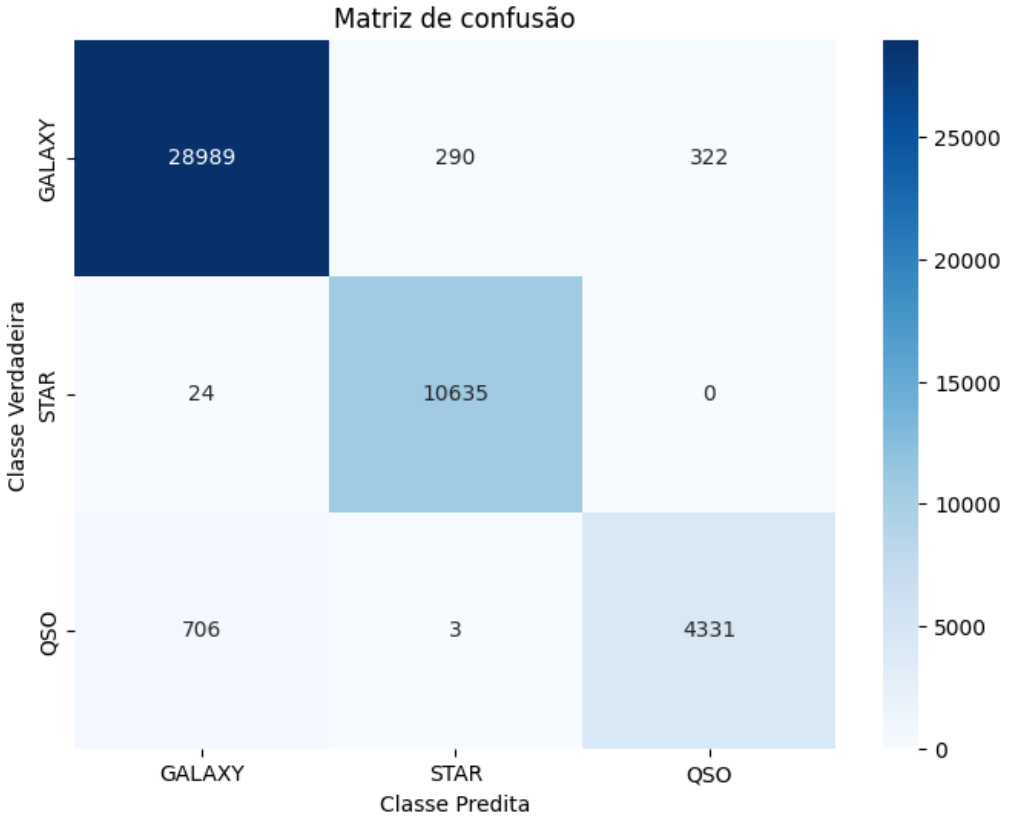

Figura 4. Matriz de Confusão