lol

November 15, 2023

Analise estatistica - League Of Legends (LoL)

Brenno de Oliveira da Rosa - 2021029935 Eduardo Alves Carvalho - 2021017550 Lucas Luan Belarmino Barbosa - 2021017872

Neste bloco, trazemos as importações necessárias e importamos uma base de dados onde estão listados dados sobre o desempenho de varios jogadores norte-americanos (NA) em partidas do jogo League Of Legends, no site kaggle é possivel se encontrar os dados de outras regiões Chamamos os primeiros dados para verificar se a biblioteca havia sido importada:

[75]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression

lol = pd.read_csv('NAmatch.csv')
lol.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5760 entries, 0 to 5759
Data columns (total 22 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Unnamed: 0           5760 non-null   int64
 1   d_spell              5760 non-null   int64
 2   f_spell              5760 non-null   int64
 3   champion             5760 non-null   object
 4   side                 5760 non-null   object
 5   role                 5760 non-null   object
 6   assists              5760 non-null   int64
 7   damage_objectives    5760 non-null   int64
 8   damage_building      5760 non-null   int64
 9   damage_turrets       5760 non-null   int64
 10  deaths               5760 non-null   int64
 11  gold_earned          5760 non-null   int64
 12  kda                  5760 non-null   float64
 13  kills                5760 non-null   int64
 14  level                5760 non-null   int64
```

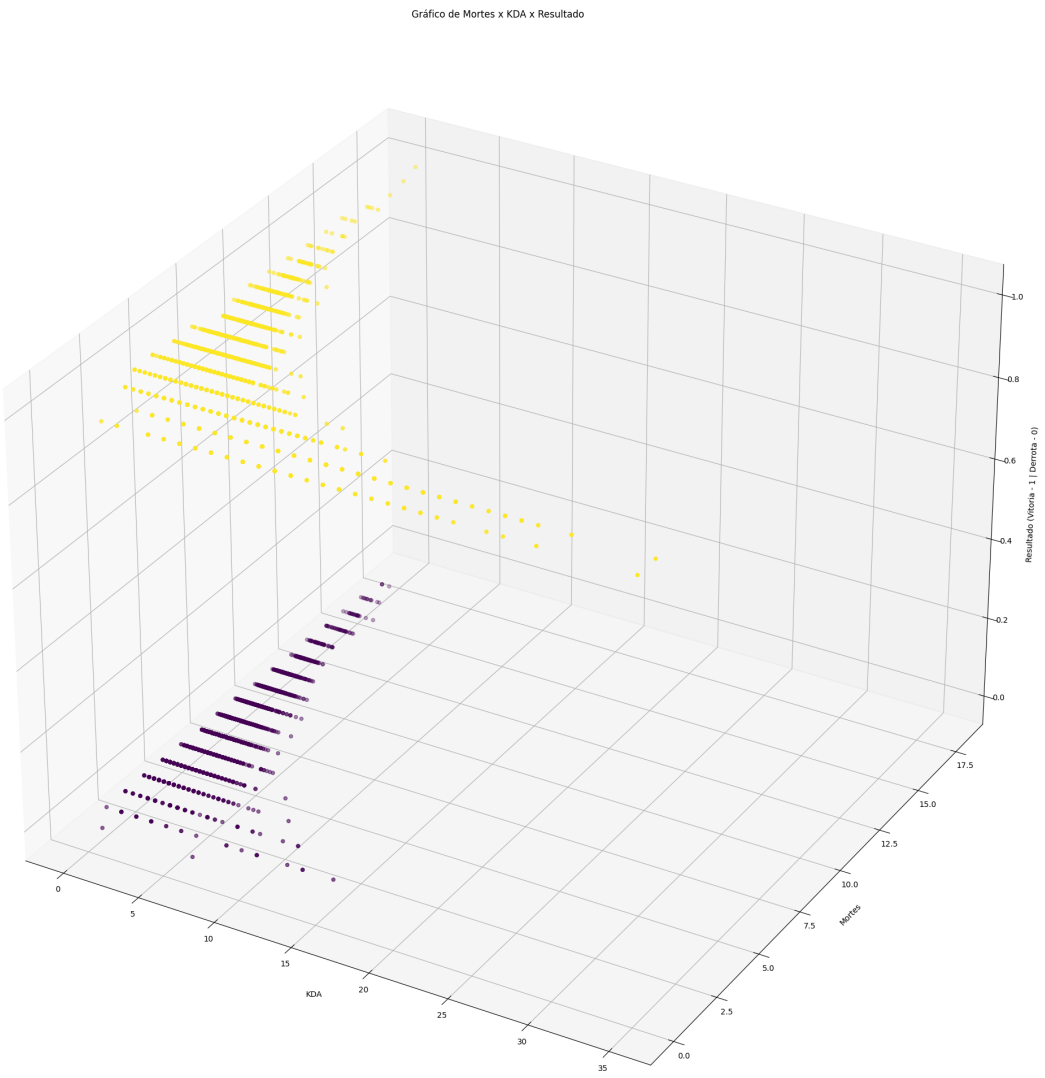
```
 15  time_cc                5760 non-null   int64
 16  damage_total           5760 non-null   int64
 17  damage_taken           5760 non-null   int64
 18  total_minions_killed   5760 non-null   int64
 19  turret_kills           5760 non-null   int64
 20  vision_score           5760 non-null   int64
 21  result                 5760 non-null   bool
dtypes: bool(1), float64(1), int64(17), object(3)
memory usage: 950.8+ KB
```

Utilizando os dados em questão, faremos um levantamento sobre as chances de vitória de um jogador base nos fatores de: numero de vezes em que um jogador é abatido durante uma partida e pelo seu KDA (do inglês Kills-Deaths-Assists), calculado pela soma do numero de abates e assistencias divididas pela quantidade de vezes que o player morreu em partida.

[73]:
```
x = lol[['kda', 'deaths']]
y = lol['result']

fig = plt.figure(figsize=(20, 40))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x['kda'], x['deaths'], y, c=y, cmap='viridis')
ax.set_xlabel('KDA', labelpad=10)
ax.set_ylabel('Mortes', labelpad=10)
ax.set_zlabel('Resultado (Vitoria - 1 | Derrota - 0)', labelpad=10)
ax.set_title('Gráfico de Mortes x KDA x Resultado')

plt.tight_layout()
plt.show()
```



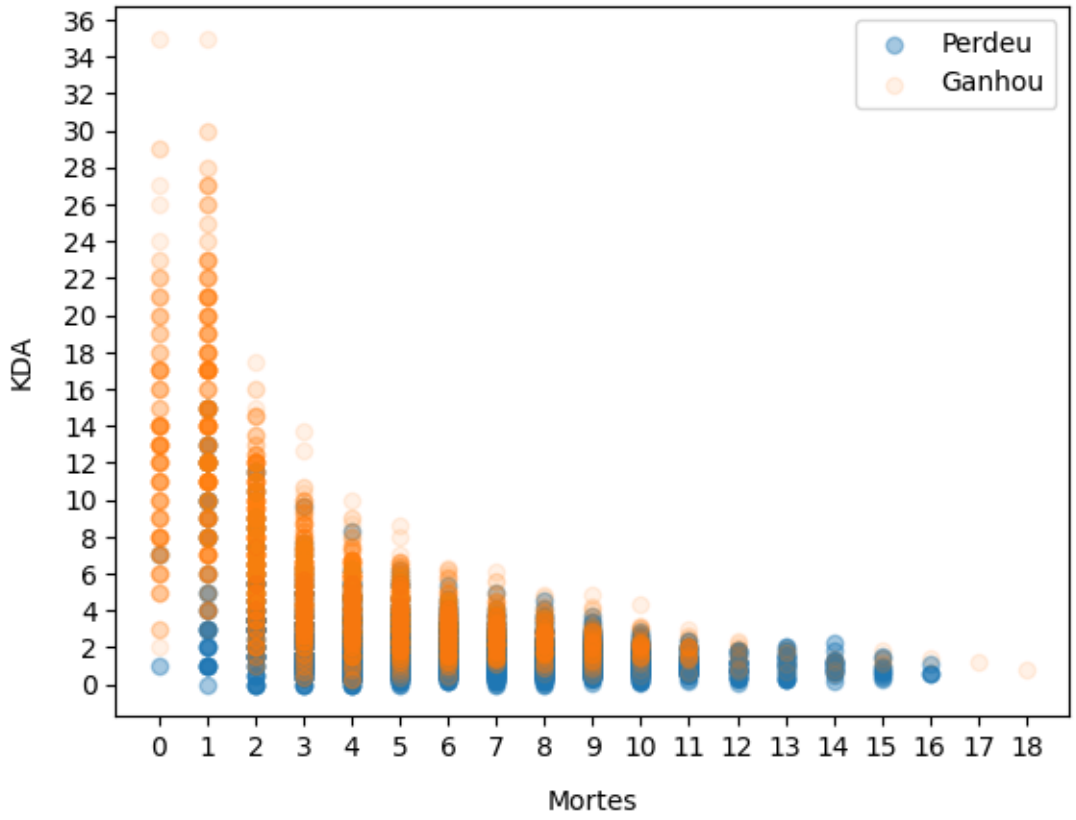
A imagem acima exibe gráficamente a distribuição de resultados (vitoria ou derrota) dos jogadores, juntamente aos seus valores de KDA e morte, por meio dela é possivel se notar alguns padrões como a maior concentração de pontos na faixa entre 5 a 15 mortes e 0 a 5 de KDA. Isso ocorre a diversos fatores, como tempo médio de jogo, composição de times além do desempenho individual e em equipe de cada jogador.

[82]:
```
lr = LogisticRegression()
scores = cross_val_score(lr, x, y, cv=10, scoring='accuracy') # validação␣
  ↪cruzada
print('Acurácia média:', scores.mean())

def plot_scatter():
    plt.scatter(x['deaths'][y == 0], x['kda'][y == 0], alpha=.4, label='Perdeu')
```

```
    plt.scatter(x['deaths'][y == 1], x['kda'][y == 1], alpha=.1, label='Ganhou')
    plt.legend()
    plt.xlabel('Mortes', labelpad=10)
    plt.ylabel('KDA', labelpad=10)
    plt.xticks(np.arange(0, max(x['deaths']) + 1, 1)) # Exibir os valores de 1␣
 ↪em 1 unidade no eixo x
    plt.yticks(np.arange(0, max(x['kda']) + 2, 2)) # Exibir os valores de 2 em␣
 ↪2 unidade no eixo y
plot_scatter()
```

```
Acurácia média: 0.7958333333333334
```



Pela regressão logística, obtivemos uma acurácia de 79.58%

O gráfico acima exibe gráficamente o modelo obtido, por meio dele observa-se ainda o comportamento de diminuição de resultados ao se aumentar o numero de mortes, o que já era esperado devido aos fatores externos, antes explicado. Além disso, como antes explicado os valores de KDA e de mortes são inversamente proporcionais, o que explica a queda nas areas de valores no gráfico. Como é percepitivel, quanto maior o numero de mortes e menor o KDA, mais provavel é que o jogador perca a partida.

[123]:
```
from sklearn.preprocessing import StandardScaler

df_jogadores = pd.DataFrame([[16, 2], # jogador 1
                             [3, 34], # jogador 2
                             [2, 17]]) # jogador 3
df_jogadores.columns = ['kda', 'deaths']
lr.fit(x,y)
print('Classificações:\n', lr.predict(df_jogadores))
print('Probabilidades:\n', lr.predict_proba(df_jogadores))
```

```
Classificações:
 [ True  True False]
Probabilidades:
 [[7.22763163e-06 9.99992772e-01]
 [4.13546795e-01 5.86453205e-01]
 [6.72962190e-01 3.27037810e-01]]
```

Nestes testes do modelo feitos acima, pode-se observar que as respostas são dadas através dos resultados do gráfico, então mesmo testando valores não apresentados no gráfico, o modelo em questão faz uma boa estimativa de qual será o resultado. Note que, quanto maior o numero de mortes do jogador, caso seu KDA seja superior a 1, isso indica que o mesmo obteve resultados positivos (abates e/ou assistencias) para seu time que superam a quantidade de mortes, sendo assim é, a alteração do KDA passa a ser mais volatil e de maior significância para o resultado conforme o numero de mortes aumenta.