

APRESENTAR UMA ANÁLISE ADICIONAL COM ALGUM DOS ALGORITMOS DE MACHINE LEARNING VISTOS AO LONGO DA DISCIPLINA.

COMO SUGESTÃO, ESCOLHA UMA DAS ANÁLISES MAIS DISSEMINADAS COMO: ANÁLISE DE CLUSTER HIERÁRQUICA, ANÁLISE DE CLUSTER NÃO HIERÁRQUICA (EX: K-MEANS), PCA (ANÁLISE DE COMPONENTES PRINCIPAIS) E ANÁLISE DE REGRESSÃO (MÚLTIPLA OU LOGÍSTICA).
OBS: APROFUNDAR E DETALHAR AS ESCOLHAS METODOLÓGICAS.

MATH_BIALUZ / DIEGO / GABRIEL

MODELO ESCOLHIDO É O ANÁLISE DE CLUSTER NÃO HIERÁRQUICA (EX: K-MEANS)

Em nosso código fizemos uma seleção de variáveis, junto com o nosso dataframe para ter a Preparação dos dados e assim a escolha do número de clusters ($K = 3$) onde a aplicação do algoritmo K-means foi um sucesso no nosso colab que disponibilizarei no próximo slide bem com sua análise dos clusters obtendo assim uma tomada de decisão nas apostas.

```
+ Código + Texto Copiar para o Drive
```

```
[1] # KNN
vizinhos = 3
```

```
# Definições de Parâmetros e dados de entrada e resultado para treinamento
knn_dados = KNeighborsClassifier(n_neighbors=vizinhos, metric="minkowski", p=2)
knn_dados.fit(x_dados_treinamento, y_dados_treinamento)
```

```
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

```
[14] # Previsões
previsoes_knn = knn_dados.predict(x_dados_teste)
```

```
[15] # taxa de acerto
accuracy_score(y_dados_teste, previsoes_knn)
```

0.601063829787234

- `df1["Previsões"] = previsoes_knn:`
Adiciona uma coluna chamada "Previsões" ao DataFrame `df1` com os valores das previsões do algoritmo KNN.
- `stake = 1:` Define o valor da aposta.
- `win_Back = stake * (df1.Odd_H_FT - 1):`
Calcula o lucro em caso de vitória (Back) considerando as odds de casa (Odd_H_FT).
- `lose_Back = -stake:` Define o prejuízo em caso de derrota (Back).

ACESSE O LINK PARA EXECUTAR A
APLICAÇÃO COM O MODELO K-MEANS

https://colab.research.google.com/drive/1-In3oH55uAE-uWYFfXzf_Sl2l_G1WHaO#scrollTo=Az_qDY_l4RIh

```
[21] df1["Previsões"] = previsoes_knn

stake = 1
win_Back = stake * (df1.Odd_H_FT - 1)
lose_Back = -stake

df1.loc[(df1['Previsões'] == 1) & (df1['BackHome'] == 1), 'Profit'] = win_Back
df1.loc[(df1['Previsões'] == 1) & (df1['BackHome'] == 0), 'Profit'] = lose_Back
df1.loc[(df1['Previsões'] == 0) & (df1['BackHome'] == 1), 'Profit'] = 0
df1.loc[(df1['Previsões'] == 0) & (df1['BackHome'] == 0), 'Profit'] = 0

filtro = df1.Previsões == 1
df0 = df1[filtro]

# Ajustando o Índice
df0.reset_index(inplace=True, drop=True)
df0.index = df0.index.set_names(['Nº'])
df0 = df0.rename(index=lambda x: x + 1)

df0['Profit_acu'] = df0.Profit.cumsum()
df0.Profit_acu.plot()
df0.Profit_acu.tail(1)
```