Análise de Dados Exploratória -Python

A resolução das fichas de trabalho deverá fazer parte integrante de um único relatório que deverá:
- conter a resolução de TODAS as fichas de trabalho;
- ser submetido em formato pdf, através da plataforma Nónio, dentro do prazo indicado nessa plataforma;
- ser acompanhado pela submissão separada de um ficheiro notebook no formato .ipynb contendo o código desenvolvido;
- seguir o modelo disponível na plataforma Nónio;
- ser realizado individualmente;
- incluir uma análise SWOT;
- ser assinado digitalmente.

Para a realização do presente trabalho considere o dataset que lhe foi atribuído no site da unidade curricular.

Data wrangling (munging) and Exploration

Aquisição de Dados

1. Recorra à biblioteca Pandas para ler o conteúdo do dataset.
   df = pd.read_csv(path)
2. Visualize as 15 primeiras linhas.
   df.head())
3. Obtenha a dimensão do dataframe.
   df.shape
4. Remova a primeira linha do dataframe se esta contiver os headers.
5. Adicione os mesmos headers que retirou anteriormente ao dataframe. Caso o dataframe não tenha à partida os headers, deverá criar os mesmos. Se o dataframe for constituído por um número elevado de colunas poderá reduzi-las para um valor inferior.
   headers = ["column1","column2"]
   df.columns = ["column1","column2"] ou
   df = pd.read_csv(filename, names = headers)
6. Apresente o nome das colunas.
   df.columns
7. Renomeie uma das colunas (à escolha) para um nome mais sugestivo.
   df.rename(columns={'column_old': 'column_new'}, inplace=True)
8. Renomeia a primeira coluna para "Indice" (caso esta coluna se trate de um índice).
   df.index.name = 'Indice'

Estatística Descritiva

9. Obtenha um sumário da estatística descritiva para cada coluna (média, desvio padrão, quartis), incluindo as colunas do tipo "object". Descreva o significado de cada uma das linhas do sumário obtido.
   df.describe(include = "all")
10. Obtenha um sumário da estatística descritiva somente para variáveis categóricas.
    df.describe(include=['object'])
11. Obtenha um sumário da estatística descritiva somente para uma coluna (à escolha), respeitante a uma variável de escala.

df[['column1', 'column2']].describe()

12. Obtenha os valores máximos e mínimos da coluna anterior.
    df['column'].max()
13. Apresente a frequência de cada um dos valores possíveis de uma coluna (à escolha), respeitante a uma variável categórica.
    df['column'].value_counts()
14. Crie uma dataframe contendo cada um dos valores possíveis da coluna da alínea anterior.
    df['column'].value_counts().to_frame()
15. Apresente graficamente o histograma de uma coluna à escolha.
    plt.pyplot.hist(df["column"]) plt.pyplot.xlabel("column") plt.pyplot.ylabel("count") plt.pyplot.title("Histogram")
16. Apresente as categorias existentes numa coluna (à escolha), respeitante a uma variável categórica.
    df['column'].unique()
17. Apresente em caixa de bigodes (boxplot) a relação entre duas colunas (à escolha), uma respeitante a uma variável categórica, e outra respeitante a uma variável de escala.
    sns.boxplot(x="column1", y="column2", data=df)

Missing Values

18. Substitua numa coluna (à escolha) respeitante a uma variável de escala, um determinado valor (à escolha) por NaN. Estes valores serão tratados seguidamente como "missing values".
    df.replace('?', np.NaN, inplace = True)
19. Obtenha o número de "missing values" presentes na coluna em causa.
    missing_data=df.isnull()
    print (missing_data[column].value_counts()))
20. Descreva as várias formas de lidar com "missing values".
21. Substitua os “missing values” presentes na coluna em causa, pela média da mesma.
    avg_norm_loss = df["column"].astype("float").mean(axis=0)
    df["column"].replace(np.nan, avg_norm_loss, inplace=True)
22. Elimine os “missing values” da coluna em causa.
    df.dropna(subset=["column"], axis=0, inplace=True) df.reset_index(drop=True, inplace=True)

Normalização

23. Caracterize os tipos de dados utilizados nos dataframes do Pandas.
24. Obtenha o tipo de dados de cada coluna.
    df.dtypes, df.info()
25. Converta, se necessário, os tipos das colunas para o formato adequado.
    df[["column1", "column2"]] = df[["column1", "column2"]].astype("float")
26. Normalize uma coluna (à escolha), respeitante a uma variável de escala, através da criação de uma nova coluna que resulte da primeira, mas cujos valores variem entre 0 e 1.
    df['column'] = df['column']/df['column'].max()

Binning

27. Crie uma nova coluna que resulte da divisão de uma coluna (à escolha), respeitante a uma variável de escala, em três categorias "Baixa", "Media", "Elevada". Os pontos de corte são à escolha.

bins = np.linspace(min(df["column"]), max(df["column"]), 4) group_names = ['Low', 'Medium', 'High'] df[' column_new '] = pd.cut(df[' column_old '], bins, labels=group_names, include_lowest=True)

29. Compare a nova coluna com aquela que lhe deu origem mostrando simultaneamente as primeiras 5 linhas de cada uma.
    df[['column_old','column_new']].head(5)
28. Apresente graficamente o histograma da nova coluna.
    plt.pyplot.hist(df["column"])
    plt.pyplot.xlabel("column")
    plt.pyplot.ylabel("count")
    plt.pyplot.title("Histogram")

Estatística Inferencial

29. Obtenha o coeficiente de correlação entre as diferentes colunas (features) numéricas do dataset.
    df.corr()
30. Para as duas "features" mais fortemente correlacionadas produza o respetivo scatterplot.
    sns.regplot(x="column1", y="column2", data=df)
    plt.ylim(0,)
31. Apresente uma "pivot table" considerando uma coluna (à escolha), respeitante a uma variável de escala, em função de duas colunas (à escolha), ambas respeitantes a variáveis categóricas.
    df_new = df[['column1','column2','column3']]
    grouped = df_new.groupby(['column1','column2'],as_index=False).mean()
    grouped_pivot = grouped.pivot(index='column1',columns='column2')
    grouped_pivot
32. Apresente graficamente a "pivot table" da alínea anterior no formato de um "heatmap".
    plt.pcolor(grouped_pivot, cmap='RdBu')
    plt.colorbar()
    plt.show()
33. Apresente a média da variável de escala anterior para cada grupo de uma das variáveis categóricas anteriores.
    df_groups = df[['column1','column2']]
    grouped_by_mean = df_groups.groupby(['column1'],as_index=False).mean()
34. Realize o teste one-way ANOVA para os grupos de uma das variáveis categóricas anteriores, considerando a variável de escala anterior como variável dependente.
    grouped = df_groups.groupby(['column1'])
    df_groups['column1'].unique()
    f_val, p_val = stats.f_oneway(grouped.get_group('group1')['Age'],
    grouped.get_group('Group2')['Age'], ...)