```
In [ ]: !jupyter-nbconvert --to-PDFviaHTML
```

ANÁLISE DE MICRODADOS DO ENEM 2019

PREVENDO NOTAS DOS PARTICIPANTES COM USO DE MACHINE LEARNING

O Exame Nacional do Ensino Médio, criado em 1998, é a avaliação que facilita a população, teoricamente de forma indiscriminada, o acesso às universidades públicas localizadas em todo o território nacional, e também utilizado como critério de avaliação para o sistema ProUni, que permite a ingressão em universidades particulares.

Um grande questionamento é levantado durante muito tempo sobre como a renda e fatores socioecônomicos facilitam aqueles que possuem uma renda salarial maior do que sobre outros na ingressão de universidades. Iremos então tratar os dados e investigar se realmente esses fatores alteram sua nota principal.

Depois disso, iremos utilizar parte dos dados como treinamento da máquina de aprendizado e verificar se conseguimos prever a nota média dos participantes levando em conta os parâmetros como renda, acesso a escola pública ou privada, entre outros.

Para darmos início à análise dos dados do Enem de 2019, disponibilizados no link do governo, veremos as colunas que podem ser trabalhadas na análise:

Coluna	Importância
TP_FAIXA_ETARIA	Separado em 20 classificações
TP_SEXO	2 classificações
TP_ESTADO_CIVIL	5 classificações
TP_COR_RACA	6 classificações
TP_NACIONALIDADE	5 classificações
TP_ST_CONCLUSAO	Situação de conclusão do Ensino Médio (4 classificações)
TP_ESCOLA	Tipo de escola do Ensino Médio (2 classificações)
TP_DEPENDENCIA_ADM_ESC	Dependência administrativa da Escola (4 classificações)
TP_ENSINO	Tipo de instituição que concluiu ou concluirá o Ensino Médio (3 classificações)
TP_LOCALIZACAO_ESC	Localização da Escola (2 classificações)
NU_NOTA_CN	Nota da prova de Ciências da Natureza
NU_NOTA_CH	Nota da prova de Ciências Humanas
NU_NOTA_LC	Nota da prova de Linguagens e Códigos
NU_NOTA_MT	Nota da prova de Matemática
NU_NOTA_REDACAO	Nota da prova de Redação
Q001,Q002,Q003,Q004,Q005,Q006,Q022,Q024,Q025	Dados categóricos de questionário sócioeconômico

As colunas que serão retiradas do dataset (não listadas acima) são colunas qualitativas nominais, ou seja, seriam valores que não se encaixariam no cálculo e predição das notas médias dos participantes. Algumas delas são: Número da inscrição (um ID que diferencia um aluno dos demais, Código do município, entre outros.

ANÁLISE E TRATAMENTO DOS DADOS

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
        import plotly.express as px
```

Começamos então a leitura do arquivo, já retirando os dados nulos (uma vez que indicam os alunos que não participaram da prova) e escolhendo somente as colunas que utilizaremos:

```
In [2]: df_enem = pd.read_csv("./data/MICRODADOS_ENEM_2019.csv",
                              encoding='latin-1',
                              sep=';',
                              #nrows foi utilizado para análises iniciais (por conta do peso total do dataset)
                              nrows=100_000,
                              usecols=['TP_FAIXA_ETARIA', 'TP_SEXO','TP_ESTADO_CIVIL',
                                       'TP_COR_RACA', 'TP_NACIONALIDADE','TP_ST_CONCLUSAO',
                                       'TP_ESCOLA','TP_ENSINO', 'TP_DEPENDENCIA_ADM_ESC',
                                       'TP_LOCALIZACAO_ESC', 'NU_NOTA_CN', 'NU_NOTA_CH',
                                       'NU_NOTA_LC', 'NU_NOTA_MT', 'NU_NOTA_REDACAO',
                                       'Q001', 'Q002', 'Q003', 'Q004', 'Q005', 'Q006',
                                       'Q022', 'Q024', 'Q025'] ).dropna()
```

```
In [3]: df_enem.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 17384 entries, 26 to 99997
Data columns (total 24 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   TP_FAIXA_ETARIA         17384 non-null  int64  
 1   TP_SEXO                 17384 non-null  object 
 2   TP_ESTADO_CIVIL         17384 non-null  int64  
 3   TP_COR_RACA             17384 non-null  int64  
 4   TP_NACIONALIDADE        17384 non-null  int64  
 5   TP_ST_CONCLUSAO         17384 non-null  int64  
 6   TP_ESCOLA               17384 non-null  int64  
 7   TP_ENSINO               17384 non-null  float64
 8   TP_DEPENDENCIA_ADM_ESC  17384 non-null  float64
 9   TP_LOCALIZACAO_ESC      17384 non-null  float64
 10  NU_NOTA_CN              17384 non-null  float64
 11  NU_NOTA_CH              17384 non-null  float64
 12  NU_NOTA_LC              17384 non-null  float64
 13  NU_NOTA_MT              17384 non-null  float64
 14  NU_NOTA_REDACAO         17384 non-null  float64
 15  Q001                    17384 non-null  object 
 16  Q002                    17384 non-null  object 
 17  Q003                    17384 non-null  object 
 18  Q004                    17384 non-null  object 
 19  Q005                    17384 non-null  int64  
 20  Q006                    17384 non-null  object 
 21  Q022                    17384 non-null  object 
 22  Q024                    17384 non-null  object 
 23  Q025                    17384 non-null  object 
dtypes: float64(8), int64(7), object(9)
memory usage: 3.3+ MB
```

```
In [4]: # Transformando variáveis categócias 'string' para numéricas.

        df_enem_mod = pd.get_dummies(df_enem, columns=['TP_SEXO'], drop_first=True)
        df_enem_mod = df_enem_mod.replace({'A':1,'B':2,'C':3,'D':4,'E':5,'F':6,'G':7,'H':8,'I':9,'J':10,'K':11,'L':12,'
        df_enem_mod[['Q001','Q002','Q003','Q004','Q022','Q025']].head(2)
```

Out[4]:

	Q001	Q002	Q003	Q004	Q022	Q025
26	5	5	2	2	3	2
28	5	2	2	2	4	1

```
In [5]: df_enem_mod['NOTA_TOTAL'] = df_enem_mod[['NU_NOTA_CN','NU_NOTA_LC','NU_NOTA_MT','NU_NOTA_CH','NU_NOTA_REDACAO']

        df_enem_mod[['NU_NOTA_CN','NU_NOTA_LC','NU_NOTA_MT','NU_NOTA_CH','NU_NOTA_REDACAO','NOTA_TOTAL']].head(2)
```

Out[5]:

	NU_NOTA_CN	NU_NOTA_LC	NU_NOTA_MT	NU_NOTA_CH	NU_NOTA_REDACAO	NOTA_TOTAL
26	618.2	636.3	713.7	744.7	900.0	722.58
28	430.4	515.9	394.1	466.8	580.0	477.44

Os dados sócioeconômicos que estão presentes no dataframe são importantes também para a análise e aprendizado de máquina, e podemos ver a importância na seguinte investigação:

A coluna 'Q006' separa a renda salarial dos participantes em categorias que vão de A a Q (ou 1 a 17, nos nossos dados já tratados), da menor para a maior renda. Verificando a média da nota dos participantes de cada categoria, podemos ter uma noção de como a renda mensal familiar afeta no desempenho dos inscritos.

```
In [6]: df_renda = pd.DataFrame(columns=[['Renda', 'Nota média']])

        for i in range (17):
            df_enem_renda = df_enem_mod[df_enem_mod['Q006'] == i+1]
            df_renda.loc[i+1]=[f"De {998*(i-1)+0.01} até {998*(i)}",round(df_enem_renda['NOTA_TOTAL'].mean(),2)]
        df_renda = df_renda.replace(['De -997.99 até 0'],['Nenhuma renda'])
        df_renda
```

Out[6]:

	Renda	Nota média
1	Nenhuma renda	485.88
2	De 0.01 até 998	505.57
3	De 998.01 até 1996	521.14
4	De 1996.01 até 2994	540.53
5	De 2994.01 até 3992	541.92
6	De 3992.01 até 4990	560.09
7	De 4990.01 até 5988	562.76
8	De 5988.01 até 6986	582.09
9	De 6986.01 até 7984	588.77
10	De 7984.01 até 8982	595.97
11	De 8982.01 até 9980	610.98
12	De 9980.01 até 10978	622.13
13	De 10978.01 até 11976	624.48
14	De 11976.01 até 12974	637.93
15	De 12974.01 até 13972	644.45
16	De 13972.01 até 14970	664.88
17	De 14970.01 até 15968	674.94

Nesta tabela percebemos como existe uma correlação entre o aumento da faixa salarial para o rendimento nas provas do ENEM. Faremos um gráfico que desenhará de uma forma mais clara o movimento das notas médias agrupadas por variáveis categóricas econômicas.

ANALISANDO GRÁFICOS

```
In [7]: df = df_enem_mod.copy()
        for i in range (17):
            df_aux = df[df['Q006'] == i+1]
            df.loc[df['Q006'] == i+1, 'NOTA_TOTAL'] = round(df_aux['NOTA_TOTAL'].mean(),2)
        df['COUNT'] = 1
```

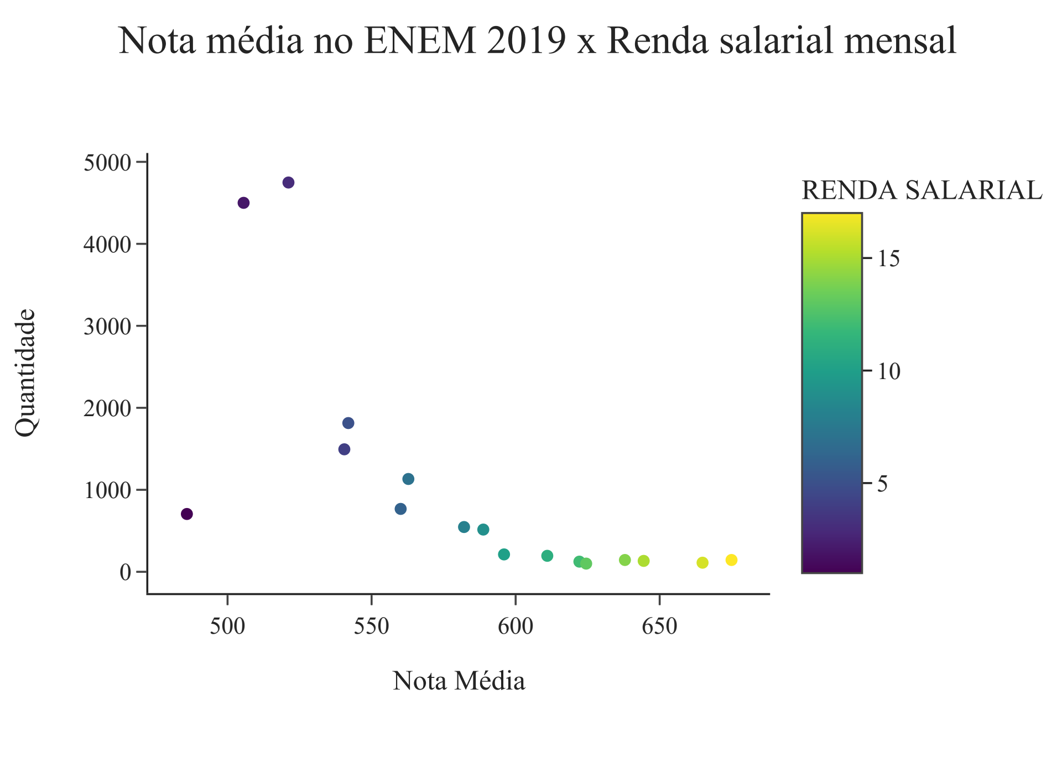
```
In [8]: df_agrupado = df.groupby(['NOTA_TOTAL'], as_index=False)['COUNT'].sum()
        df_agrupado['RENDA SALARIAL'] = df_agrupado.index + 1
```

```
In [9]: df_agrupado.head(2)
```

Out[9]:

	NOTA_TOTAL	COUNT	RENDA SALARIAL
0	485.88	705	1
1	505.57	4500	2

```
In [10]: fig_renda = px.scatter(df_agrupado, x="NOTA_TOTAL", y="COUNT", color="RENDA SALARIAL", template='simple_white',
                                height= 400, width= 550, title = 'Nota média no ENEM 2019 x Renda salarial mensal')
         fig_renda.update_layout(font_family='Rockwell',
                                 title=dict(xanchor='center',
                                            x=.5,
                                            font_size=20),
                                 legend=dict(y=1, yanchor='top',
                                             x=1, xanchor="right"))
         fig_renda.update_xaxes(title='Nota Média')
         fig_renda.update_yaxes(title='Quantidade')
         fig_renda.show()
```



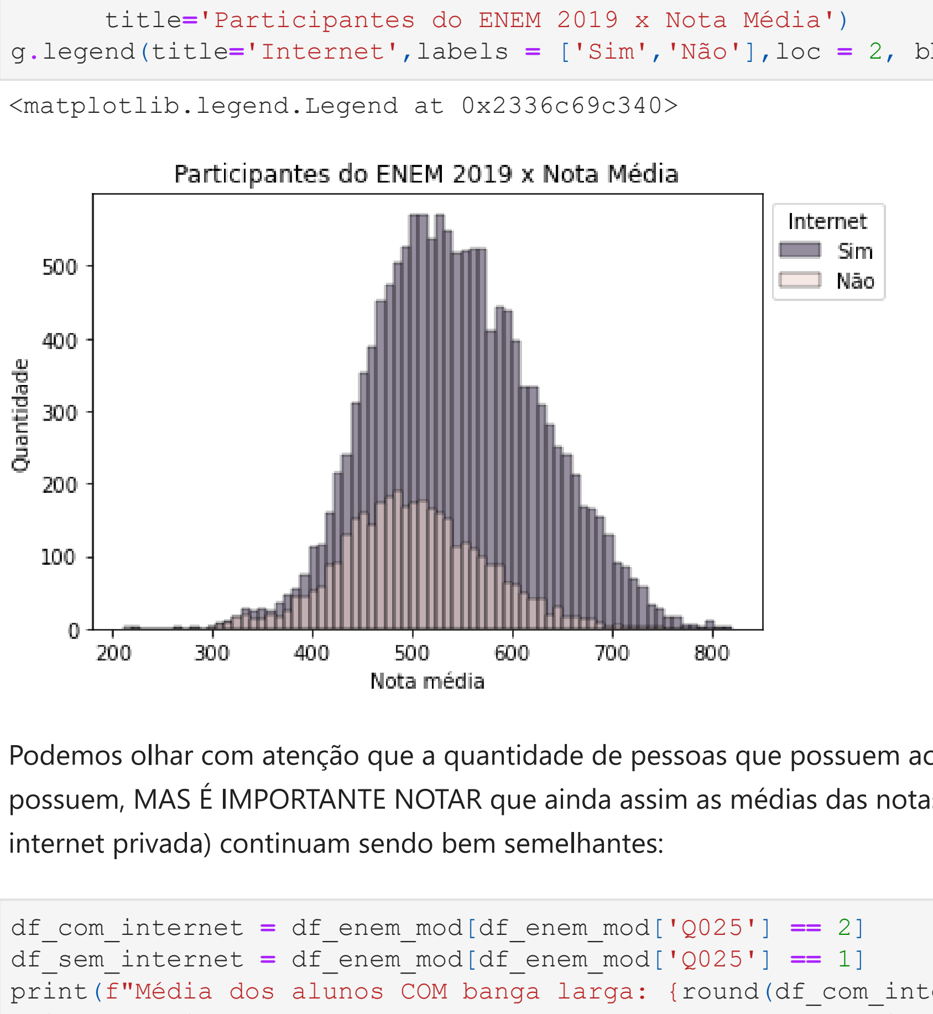
Nele interpretamos um descrescimento exponencial da quantidade da população inserida em uma renda salarial, ou seja, quanto mais você ganha, menor será o grupo em que você pertence na sociedade brasileira.

A matéria de 2021 feita pela BBC Brasil, podendo ser verificada no link do site BBC, nos traz os dados retirados do IBGE (Instituto Brasileiro de Geografia e Estatística) e comprovam que a grande maioria da população não recebe mais que R$3,5 mill.

Outro gráfico que podemos construir é um histograma para verificação dos alunos que possuem acesso a internet.

```
In [11]: g = sns.histplot(data=df_enem_mod, x="NOTA_TOTAL", hue="Q025")
         g.set(xlabel='Nota média',ylabel='Quantidade',
               title='Participantes do ENEM 2019 x Nota Média')
         g.legend(title='Internet',labels = ['Sim','Não'],loc = 2, bbox_to_anchor = (1,1))
```

```
Out[11]: <matplotlib.legend.Legend at 0x2336c69c340>
```



Podemos olhar com atenção que a quantidade de pessoas que possuem acesso a banda larga é muito maior do que os que não possuem, MAS É IMPORTANTE NOTAR que ainda assim as médias das notas na avaliação entre os dois grupos (com e sem acesso a internet privada) continuam sendo bem semelhantes:

```
In [12]: df_com_internet = df_enem_mod[df_enem_mod['Q025'] == 2]
         df_sem_internet = df_enem_mod[df_enem_mod['Q025'] == 1]
         print(f"Média dos alunos COM banga larga: {round(df_com_internet['NOTA_TOTAL'].mean(),2)}")
         print(f"Média dos alunos SEM banga larga: {round(df_sem_internet['NOTA_TOTAL'].mean(),2)}")
```

```
Média dos alunos COM banga larga: 543.99
Média dos alunos SEM banga larga: 501.88
```

```
In [13]: x = len(df_sem_internet)/len(df_enem_mod)
         print("Porcentagem dos desconectados:",round(x,4))
```

```
Porcentagem dos desconectados: 0.2088
```

De acordo com G1, no link, 20% da população brasileira com mais de 16 anos NÃO possui acesso a internet, o que corresponde a porcentagem que calculamos acima.

MACHINE LEARNING: PREVENDO A NOTA MÉDIA

Agora iremos treinar nosso modelo, onde primeiramente separaremos os dados em uma variável preditora e outra variável alvo.

```
In [14]: X = df_enem_mod.iloc[:,:-1]
         y = df_enem_mod.iloc[:,-1]
         X.head(2)
```

Out[14]:

	TP_FAIXA_ETARIA	TP_ESTADO_CIVIL	TP_COR_RACA	TP_NACIONALIDADE	TP_ST_CONCLUSAO	TP_ESCOLA	TP_ENSINO	TP_DEPENDENCIA_AI
26	2	1	1	1	2	3	1.0	
28	3	1	1	1	2	2	1.0	

2 rows × 24 columns

```
In [15]: # Train Test Split:
         from sklearn.model_selection import train_test_split
         X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=0)
```

```
In [16]: from sklearn.metrics import mean_squared_error, r2_score
```

Utilizaremos o algoritmo Random Forest, onde ele basicamente irá criar árvores de decisão aleatórias, formando uma floresta, e cada árvore será utilizada na escolha do resultado final, em uma espécie de votação.

```
In [17]: # Random Forest Regressor:
         from sklearn.ensemble import RandomForestRegressor
         RandomForestRegressor = RandomForestRegressor()
         RandomForestRegressor = RandomForestRegressor.fit(X_train, y_train)
```

```
In [18]: # Predição:
         y_pred = RandomForestRegressor.predict(X_test)
```

```
In [19]: # Acurácia e Desvios:
         print("Acurácia", r2_score(y_test, y_pred))
         print("Desvio médio quadrático", mean_squared_error(y_test, y_pred))
```

```
Acurácia 0.9945106519468225
Desvio médio quadrático 34.154896823445476
```

```
In [20]: resultado = RandomForestRegressor.score(X_test, y_test)
         print("Acurácia:",resultado)
```

```
Acurácia: 0.9945106519468225
```

Aqui está uma tabela com 10 dados, contendo a nota média ORIGINAL e a nota média PREVISTA.

```
In [21]: previsao = RandomForestRegressor.predict(X_test[400:410])
         aux=0

         for k,v in y_test[400:410].items():
             print(round(v,2), '-->', round(previsao[aux],2))
             aux+=1
```

```
614.52 --> 612.07
575.88 --> 578.42
572.92 --> 573.75
405.08 --> 407.03
692.02 --> 697.66
517.9 --> 515.9
568.66 --> 569.6
747.14 --> 747.18
665.38 --> 666.35
469.18 --> 471.17
```