1- Análise exploratória dos dados utilizando estatística descritiva e inferencial, considerando uma, duas e/ou mais variáveis;

Importação das libs

In [35]:
```
import pandas as pd
import sweetviz as sv
import numpy as np
import sklearn
import matplotlib.pyplot as plt
```

Chamando o arquivo do excel

In [3]:
```
df = pd.read_excel('teste_smarkio_Lbs.xls','Análise_ML')
```

Verificando quantas linhas tem dado nulo

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
Pred_class          0
probabilidade       0
status              0
True_class        462
dtype: int64
```

Tratando os dados da coluna True_class(Todos que o true_class for NaN, considerar o valor da coluna Pred_class)

In [16]:
```
values = {'True_class': df['Pred_class']}
```

In [6]:
```
df.fillna(value=values,inplace = True)
```

Verificando quantas linhas tem dado nulo(depois de rodar o comando para tratar os dados)

In [83]:
```
df.isnull().sum()
```

Out[83]:
```
Pred_class       0
probabilidade    0
status           0
True_class       0
dtype: int64
```

Trazendo todas as informações do DataFrame

In [8]:
```
df
```

Out[8]:

	Pred_class	probabilidade	status	True_class
0	2	0.079892	approved	0.0
1	2	0.379377	approved	74.0
2	2	0.379377	approved	74.0
3	2	0.420930	approved	74.0
4	2	0.607437	approved	2.0
...	...	...	...	...
638	60	0.543772	revision	60.0
639	60	0.553846	revision	60.0
640	77	0.606065	revision	77.0
641	84	0.561842	revision	84.0
642	96	0.340740	revision	96.0

643 rows × 4 columns

Agrupando a coluna status pela contagem dos valores da coluna probabilidade

In [119...
```
status_group_probabi = df.groupby('status')['probabilidade'].value_counts()
```

In [120...
```
status_group_probabi
```

Out[120...
```
status    probabilidade
approved  1.000000        56
          0.432421        14
          0.426136        11
          0.057740         8
          0.288653         5
                          ..
revision  0.740292         1
          0.752448         1
          0.784920         1
          0.817525         1
          0.909148         1
Name: probabilidade, Length: 492, dtype: int64
```

In [96]:
```
status = df['status']
```

Agrupando a coluna status com as informações da coluna True_class

In [11]:
```
status_group_true =df.groupby('status')['True_class'].value_counts()
```

In [98]:
```
status_group_true
```

Out[98]:
```
status    True_class
approved  74.0          78
          2.0           58
          3.0           57
          0.0           54
          77.0          26
                        ..
revision  81.0           1
          84.0           1
          86.0           1
          113.0          1
          114.0          1
Name: True_class, Length: 91, dtype: int64
```

Fazendo a contagem do total da coluna status, agrupado por cada status.

In [13]:
```
status_group_status = df.groupby('status')['status'].count()
```

In [14]:
```
status_group_status
```

Out[14]:
```
status
approved    600
revision     43
Name: status, dtype: int64
```

Utilizando o comando describe()

In [15]:
```
df.describe()
```

Out[15]:

	Pred_class	probabilidade	True_class
count	643.000000	643.000000	643.000000
mean	52.712286	0.622436	48.251944
std	37.602068	0.266811	38.542269
min	2.000000	0.043858	0.000000
25%	12.000000	0.408017	3.000000
50%	59.000000	0.616809	55.000000
75%	81.000000	0.870083	77.000000
max	118.000000	1.000000	118.000000

2 -Calcule o desempenho do modelo de classificação utilizando pelo menos três métricas;

In [40]:
```
accuracy = sklearn.metrics.accuracy_score(df['True_class'], df['Pred_class'])
```

In [115...
```
accuracy
```

Out[115...
```
0.71850699844479
```

In [37]:
```
from sklearn.metrics import classification_report
```

In [39]:
```
print(classification_report,(df['True_class'], (df['Pred_class'])))
```

```
<function classification_report at 0x0000023FC15249D0> (0        0.0
1       74.0
2       74.0
3       74.0
4        2.0
        ...
638     60.0
639     60.0
640     77.0
641     84.0
642     96.0
Name: True_class, Length: 643, dtype: float64, 0       2
1       2
2       2
3       2
4       2
       ..
638    60
639    60
640    77
641    84
642    96
Name: Pred_class, Length: 643, dtype: int64)
```

In [60]:
```
>>> from sklearn.metrics import confusion_matrix
```

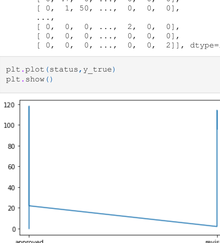
In [61]:
```
y_true = df['True_class']
y_pred = df['Pred_class']
```

confusion_matrix

In [62]:
```
confusion_matrix(y_true, y_pred)
```

Out[62]:
```
array([[ 0,  2,  2, ...,  0,  0,  0],
       [ 0, 47,  0, ...,  0,  0,  0],
       [ 0,  1, 50, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  2,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  0],
       [ 0,  0,  0, ...,  0,  0,  2]], dtype=int64)
```

In [128...
```
plt.plot(status,y_true)
plt.show()
```



In [129...
```
plt.plot(status,y_pred)
plt.show()
```