Aula 7

March 27, 2025

0.1 KDD

Base de dados ->descobrir padrões em dados e esse padrões ajudarem noa tomada de decisão -> Descobrir conhecimento em dados (KDD)(insights)

Etapas do processo de KDD: - Dados estruturados - Pré-processamento - EDA - Análise Exploratória dos Dados - Aplica as técnicas e modelos e - Obtém insights ou gráficos

Etapa 1: Importando as bibliotecas

[2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import scipy
```

[3]:
```
# Carregando a base de dados
df = pd.read_csv('diabetes.csv')
```

[4]:
```
print(df.head(10))
```

```
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1
5            5      116             74              0        0  25.6
6            3       78             50             32       88  31.0
7           10      115              0              0        0  35.3
8            2      197             70             45      543  30.5
9            8      125             96              0        0   0.0

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
5                     0.201   30        0
```

```
6                    0.248   26        1
7                    0.134   29        0
8                    0.158   53        1
9                    0.232   54        1
```

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

[6]:
```
df.isnull().sum()
```

[6]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

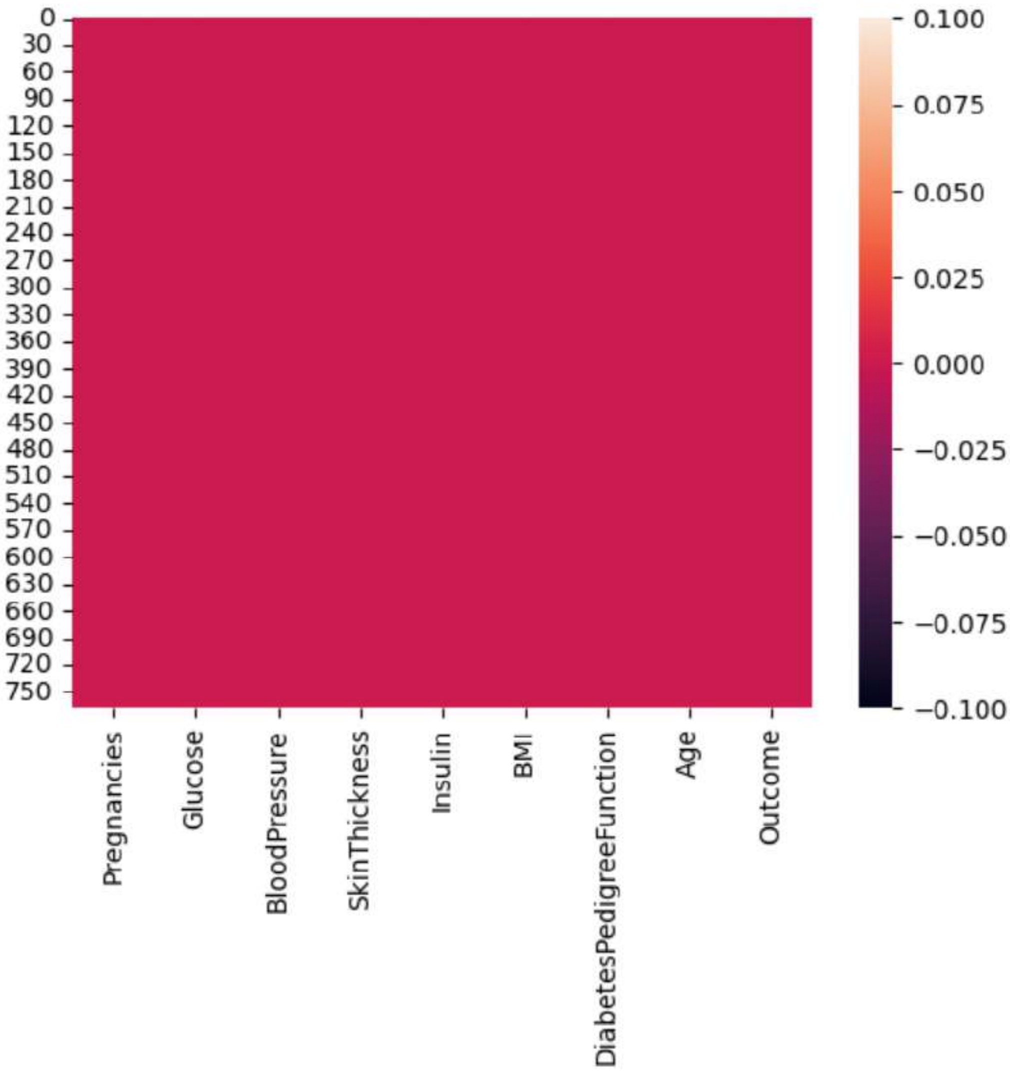
[8]:
```
df.isnull()
```

[8]:
```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin    BMI  \
0          False    False          False          False    False  False
1          False    False          False          False    False  False
2          False    False          False          False    False  False
3          False    False          False          False    False  False
4          False    False          False          False    False  False
..           …        …             …             …       …      …
763        False    False          False          False    False  False
764        False    False          False          False    False  False
```

```
765        False    False            False            False    False  False
766        False    False            False            False    False  False
767        False    False            False            False    False  False

     DiabetesPedigreeFunction    Age  Outcome
0                       False  False    False
1                       False  False    False
2                       False  False    False
3                       False  False    False
4                       False  False    False
..                        ...    ...      ...
763                     False  False    False
764                     False  False    False
765                     False  False    False
766                     False  False    False
767                     False  False    False

[768 rows x 9 columns]
```

[7]:
```
sns.heatmap(df.isnull())
```

[7]: <AxesSubplot: >



[9]:
```
df.nunique()
```

[9]:
```
Pregnancies                  17
Glucose                     136
BloodPressure                47
SkinThickness                51
Insulin                     186
BMI                         248
DiabetesPedigreeFunction    517
Age                          52
Outcome                       2
dtype: int64
```

[11]:
```
df['Pregnancies'].unique()
```

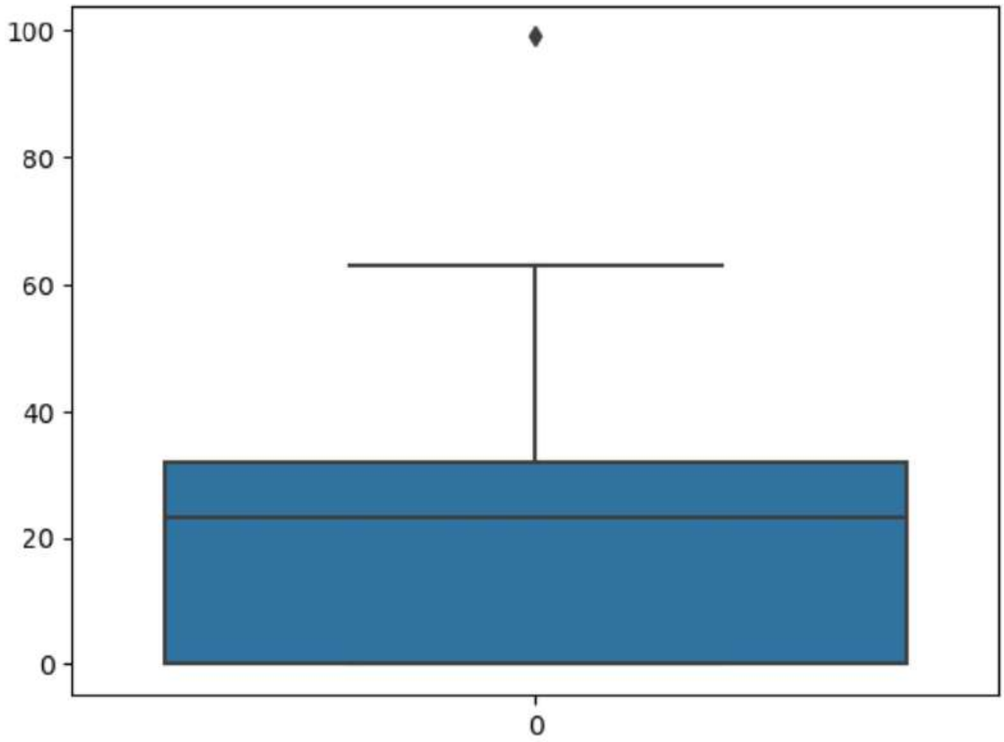
[11]:
```
array([ 6,  1,  8,  0,  5,  3, 10,  2,  4,  7,  9, 11, 13, 15, 17, 12, 14])
```

[12]:
```
df.describe().T
```

[12]:
```
                          count        mean         std     min       25%  \
Pregnancies               768.0    3.845052    3.369578   0.000   1.00000
Glucose                   768.0  120.894531   31.972618   0.000  99.00000
BloodPressure             768.0   69.105469   19.355807   0.000  62.00000
SkinThickness             768.0   20.536458   15.952218   0.000   0.00000
Insulin                   768.0   79.799479  115.244002   0.000   0.00000
BMI                       768.0   31.992578    7.884160   0.000  27.30000
DiabetesPedigreeFunction  768.0    0.471876    0.331329   0.078   0.24375
Age                       768.0   33.240885   11.760232  21.000  24.00000
Outcome                   768.0    0.348958    0.476951   0.000   0.00000

                               50%        75%     max
Pregnancies                 3.0000    6.00000   17.00
Glucose                   117.0000  140.25000  199.00
BloodPressure              72.0000   80.00000  122.00
SkinThickness              23.0000   32.00000   99.00
Insulin                    30.5000  127.25000  846.00
BMI                        32.0000   36.60000   67.10
DiabetesPedigreeFunction    0.3725    0.62625    2.42
Age                        29.0000   41.00000   81.00
Outcome                     0.0000    1.00000    1.00
```

[30]:
```
sns.boxplot(df['SkinThickness'])
```

[30]: <AxesSubplot: >



[17]:
```
Q1, Q3 = np.percentile(df['Pregnancies'], [25, 75])
```

[18]:
```
    Q1
```

[18]: 1.0

[19]:
```
Q3
```

[19]: 6.0

[21]:
```
# INTERVALO INTERQUARTIL
IRQ = Q3-Q1
```

[23]:
```
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)
```

[24]:
```
limite_inferior
```

[24]: -6.5

[25]:
```
limite_superior
```

[25]: 13.5

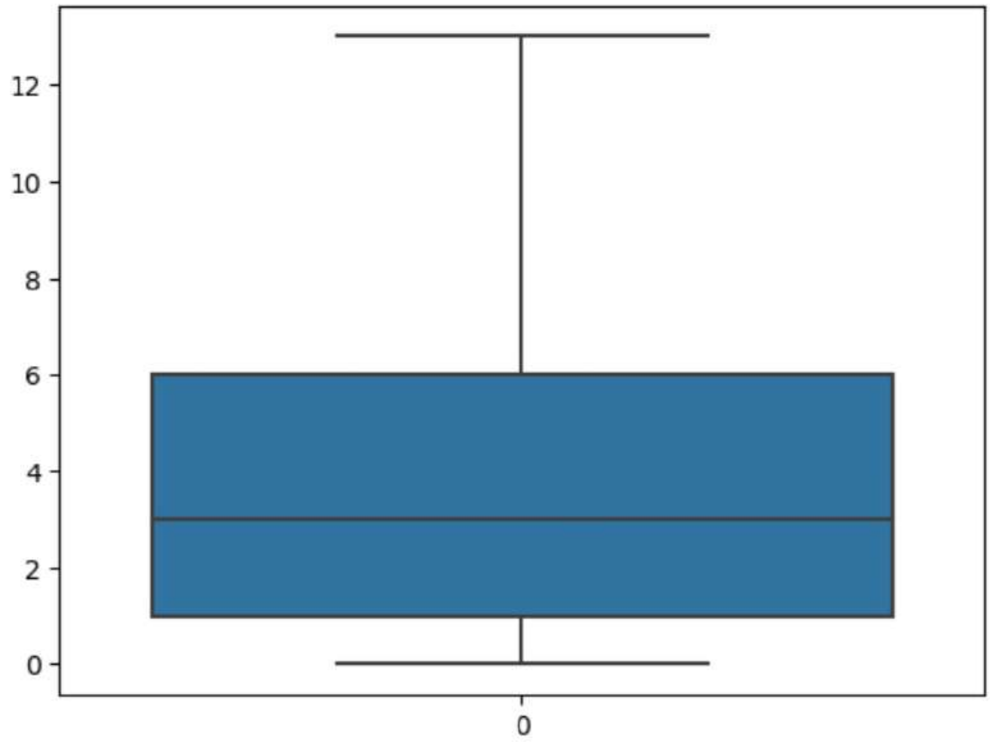
[26]:
```
df_limpo =
 ↪df[(df['Pregnancies']>=limite_inferior)&(df['Pregnancies']<=limite_superior)]
```

[27]:
```
sns
```

[27]:
```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Outcome
0                       0.627   50        1
1                       0.351   31        0
2                       0.672   32        1
3                       0.167   21        0
4                       2.288   33        1
..                        ...  ...      ...
763                     0.171   63        0
764                     0.340   27        0
765                     0.245   30        0
766                     0.349   47        1
767                     0.315   23        0

[764 rows x 9 columns]
```

[28]:
```
sns.boxplot(df_limpo['Pregnancies'])
```

[28]:
```
<AxesSubplot: >
```

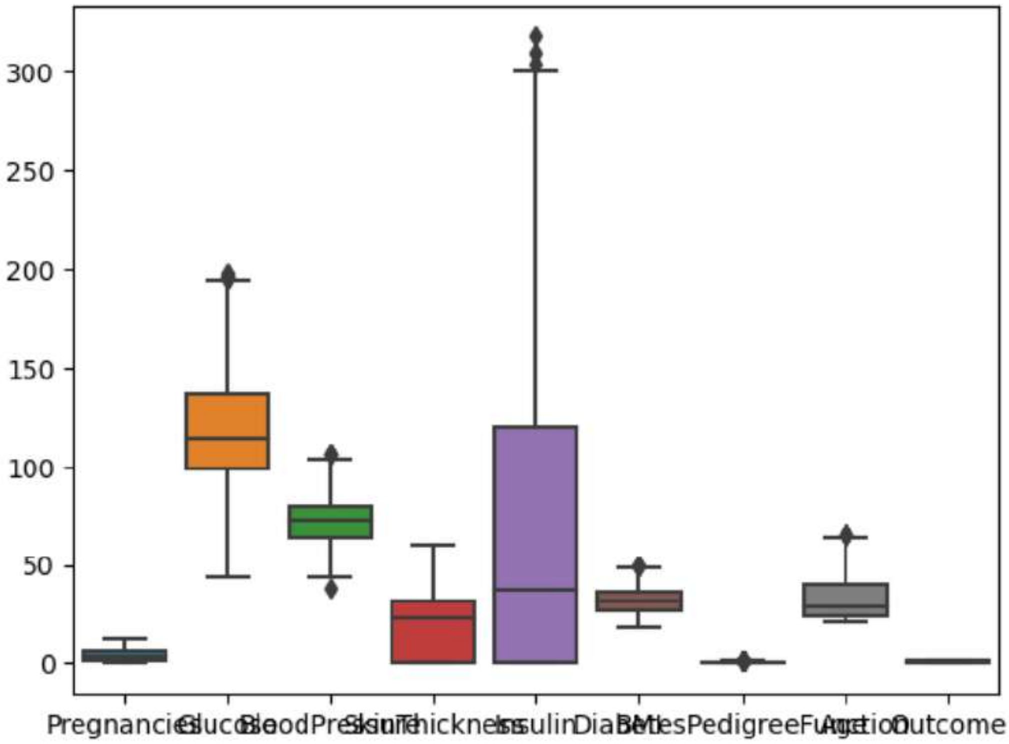


[31]:
```
Q1, Q3 = np.percentile(df['Insulin'], [25, 75])
IRQ = Q3-Q1
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)
df_limpo =␣
 ↪df_limpo[(df_limpo['Insulin']>=limite_inferior)&(df_limpo['Insulin']<=limite_superior)]

Q1, Q3 = np.percentile(df['Age'], [25, 75])
IRQ = Q3-Q1
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)

df_limpo =␣
 ↪df_limpo[(df_limpo['Age']>=limite_inferior)&(df_limpo['Age']<=limite_superior)]

Q1, Q3 = np.percentile(df['Glucose'], [25, 75])
IRQ = Q3-Q1
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)
df_limpo =␣
 ↪df_limpo[(df_limpo['Glucose']>=limite_inferior)&(df_limpo['Glucose']<=limite_superior)]
```

```
Q1, Q3 = np.percentile(df['BloodPressure'], [25, 75])
IRQ = Q3-Q1
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)
df_limpo =␣
 ↪df_limpo[(df_limpo['BloodPressure']>=limite_inferior)&(df_limpo['BloodPressure']<=limite_su

Q1, Q3 = np.percentile(df['SkinThickness'], [25, 75])
IRQ = Q3-Q1
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)
df_limpo =␣
 ↪df_limpo[(df_limpo['SkinThickness']>=limite_inferior)&(df_limpo['SkinThickness']<=limite_su

Q1, Q3 = np.percentile(df['BMI'], [25, 75])
IRQ = Q3-Q1
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)
df_limpo =␣
 ↪df_limpo[(df_limpo['BMI']>=limite_inferior)&(df_limpo['BMI']<=limite_superior)]

Q1, Q3 = np.percentile(df['DiabetesPedigreeFunction'], [25, 75])
IRQ = Q3-Q1
limite_inferior = Q1-(1.5*IRQ)
limite_superior = Q3+(1.5*IRQ)
df_limpo =␣
 ↪df_limpo[(df_limpo['DiabetesPedigreeFunction']>=limite_inferior)&(df_limpo['DiabetesPedigre
```

[32]:
```
sns.boxplot(df_limpo)
```

[32]:
```
<AxesSubplot: >
```



[33]:
```
df_limpo
```

[33]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
5	5	116	74	0	0	25.6	
..	…	…	…	…	…	…	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

	DiabetesPedigreeFunction	Age	Outcome
0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
5	0.201	30	0
..	…	…	…

```
763                          0.171   63        0
764                          0.340   27        0
765                          0.245   30        0
766                          0.349   47        1
767                          0.315   23        0

[639 rows x 9 columns]
```

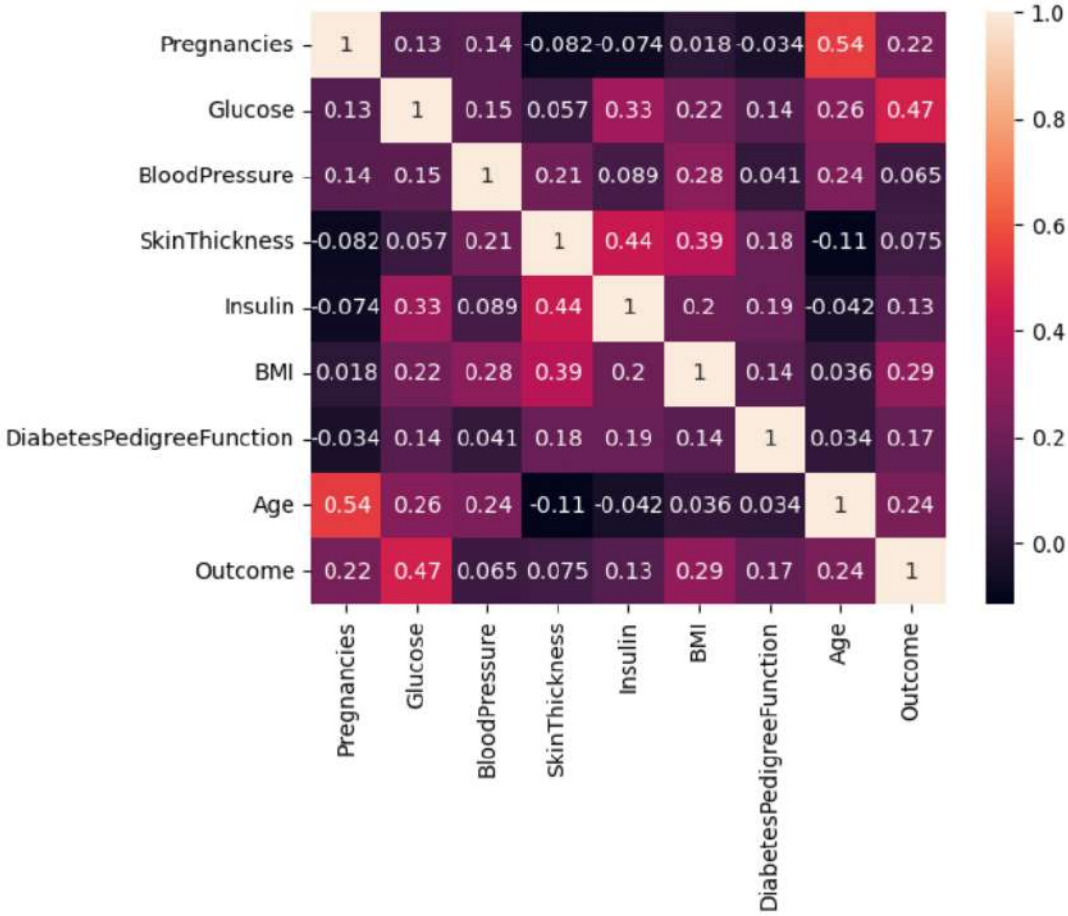
[34]:
```
correlação = df.corr()
```

[35]:
```
correlação
```

[35]:
```
                          Pregnancies   Glucose  BloodPressure  SkinThickness  \
Pregnancies                  1.000000  0.129459       0.141282      -0.081672
Glucose                      0.129459  1.000000       0.152590       0.057328
BloodPressure                0.141282  0.152590       1.000000       0.207371
SkinThickness               -0.081672  0.057328       0.207371       1.000000
Insulin                     -0.073535  0.331357       0.088933       0.436783
BMI                          0.017683  0.221071       0.281805       0.392573
DiabetesPedigreeFunction    -0.033523  0.137337       0.041265       0.183928
Age                          0.544341  0.263514       0.239528      -0.113970
Outcome                      0.221898  0.466581       0.065068       0.074752

                           Insulin       BMI  DiabetesPedigreeFunction  \
Pregnancies              -0.073535  0.017683                 -0.033523
Glucose                   0.331357  0.221071                  0.137337
BloodPressure             0.088933  0.281805                  0.041265
SkinThickness             0.436783  0.392573                  0.183928
Insulin                   1.000000  0.197859                  0.185071
BMI                       0.197859  1.000000                  0.140647
DiabetesPedigreeFunction  0.185071  0.140647                  1.000000
Age                      -0.042163  0.036242                  0.033561
Outcome                   0.130548  0.292695                  0.173844

                               Age   Outcome
Pregnancies               0.544341  0.221898
Glucose                   0.263514  0.466581
BloodPressure             0.239528  0.065068
SkinThickness            -0.113970  0.074752
Insulin                  -0.042163  0.130548
BMI                       0.036242  0.292695
DiabetesPedigreeFunction  0.033561  0.173844
Age                       1.000000  0.238356
Outcome                   0.238356  1.000000
```

[38]:
```
sns.heatmap(df.corr(), annot=True)
```

[38]: <AxesSubplot: >

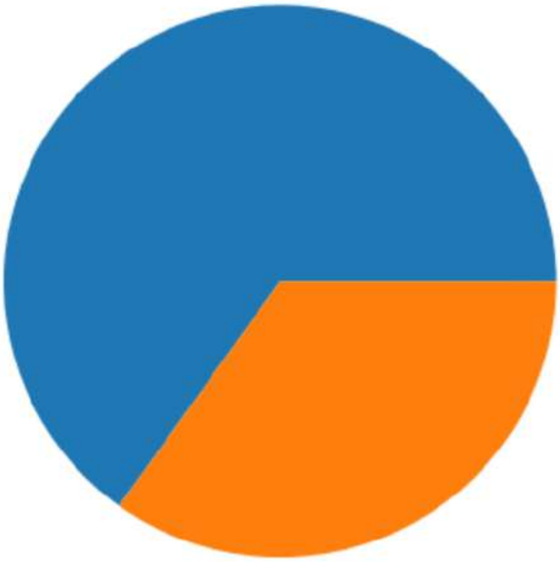


[41]:
```
correlação['Age'].sort_values(ascending=False)
```

[41]:
```
Age                         1.000000
Pregnancies                 0.544341
Glucose                     0.263514
BloodPressure               0.239528
Outcome                     0.238356
BMI                         0.036242
DiabetesPedigreeFunction    0.033561
Insulin                    -0.042163
SkinThickness              -0.113970
Name: Age, dtype: float64
```

[42]:
```
plt.pie(df.Outcome.value_counts())
```

[42]: ([<matplotlib.patches.Wedge at 0x7fc9e8062bd0>,
  <matplotlib.patches.Wedge at 0x7fc9e804eb50>],
 [Text(-0.5025943242672991, 0.9784676515931925, ''),
  Text(0.5025944158780503, -0.9784676045369114, '')])



[45]:
```
sns.pairplot(df, hue='Outcome')
```

[45]: <seaborn.axisgrid.PairGrid at 0x7fc9e076df50>

[46]:
```
from sklearn.preprocessing import MinMaxScaler
```

[47]:
```
escala = MinMaxScaler()
```

[48]:
```
# X armazeno as variáveis independentes e Y a variável alvo que é a varável␣
↪dependente
X = df.drop(columns=['Outcome'])
Y=df.Outcome
```

[49]:
```
escala.fit(X)
Xescalar = escala.transform(X)
```

[51]:
```
Xescalar
```

[51]:
```
array([[0.35294118, 0.74371859, 0.59016393, …, 0.50074516, 0.23441503,
        0.48333333],
       [0.05882353, 0.42713568, 0.54098361, …, 0.39642325, 0.11656704,
        0.16666667],
       [0.47058824, 0.91959799, 0.52459016, …, 0.34724292, 0.25362938,
        0.18333333],
       …,
       [0.29411765, 0.6080402 , 0.59016393, …, 0.390462  , 0.07130658,
        0.15      ],
       [0.05882353, 0.63316583, 0.49180328, …, 0.4485842 , 0.11571307,
        0.43333333],
       [0.05882353, 0.46733668, 0.57377049, …, 0.45305514, 0.10119556,
        0.03333333]])
```

[52]:
```
df.describe().T
```

[52]:
```
                          count        mean         std     min       25%  \
Pregnancies               768.0    3.845052    3.369578   0.000   1.00000
Glucose                   768.0  120.894531   31.972618   0.000  99.00000
BloodPressure             768.0   69.105469   19.355807   0.000  62.00000
SkinThickness             768.0   20.536458   15.952218   0.000   0.00000
Insulin                   768.0   79.799479  115.244002   0.000   0.00000
BMI                       768.0   31.992578    7.884160   0.000  27.30000
DiabetesPedigreeFunction  768.0    0.471876    0.331329   0.078   0.24375
Age                       768.0   33.240885   11.760232  21.000  24.00000
Outcome                   768.0    0.348958    0.476951   0.000   0.00000

                               50%        75%     max
Pregnancies                 3.0000    6.00000   17.00
Glucose                   117.0000  140.25000  199.00
BloodPressure              72.0000   80.00000  122.00
SkinThickness              23.0000   32.00000   99.00
Insulin                    30.5000  127.25000  846.00
BMI                        32.0000   36.60000   67.10
DiabetesPedigreeFunction    0.3725    0.62625    2.42
Age                        29.0000   41.00000   81.00
Outcome                     0.0000    1.00000    1.00
```

[53]:
```
df_titanic=pd.read_csv('titanic.csv')
```

[54]:
```
df_titanic.head()
```

[54]:
```
   survived  pclass     sex   age  sibsp  parch     fare embarked  class  \
0         0       3    male  22.0      1      0   7.2500        S  Third
1         1       1  female  38.0      1      0  71.2833        C  First
2         1       3  female  26.0      0      0   7.9250        S  Third
3         1       1  female  35.0      1      0  53.1000        S  First
```

```
4         0       3    male  35.0      0      0   8.0500        S  Third

     who  adult_male deck  embark_town alive  alone
0    man        True  NaN  Southampton    no  False
1  woman       False    C    Cherbourg   yes  False
2  woman       False  NaN  Southampton   yes   True
3  woman       False    C  Southampton   yes  False
4    man        True  NaN  Southampton    no   True
```

[57]:
```
df_titanic.duplicated()
```

[57]:
```
0      False
1      False
2      False
3      False
4      False
       …
886     True
887    False
888    False
889    False
890    False
Length: 891, dtype: bool
```

[58]:
```
df_titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    object
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    object
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), float64(2), int64(4), object(7)
memory usage: 92.4+ KB
```

[59]:
```python
cat_col = [col for col in df_titanic.columns if df_titanic[col].dtype ==
 ↪'object']
print(cat_col)
```

```
['sex', 'embarked', 'class', 'who', 'deck', 'embark_town', 'alive']
```

[60]:
```python
num_col = [col for col in df_titanic.columns if df_titanic[col].dtype !=
 ↪'object']
print(num_col)
```

```
['survived', 'pclass', 'age', 'sibsp', 'parch', 'fare', 'adult_male', 'alone']
```

[61]:
```python
df_titanic[cat_col].nunique()
```

[61]:
```
sex            2
embarked       3
class          3
who            3
deck           7
embark_town    3
alive          2
dtype: int64
```

[62]:
```python
df_titanic['deck'].unique()[:50]
```

[62]:
```
array([nan, 'C', 'E', 'G', 'D', 'A', 'B', 'F'], dtype=object)
```

[63]:
```python
df_titanic.isnull().sum()
```

[63]:
```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

[64]:
```python
177/890
```

[64]:
```
0.19887640449438201
```

[65]:
```
2/890
```

[65]: 0.0022471910112359553

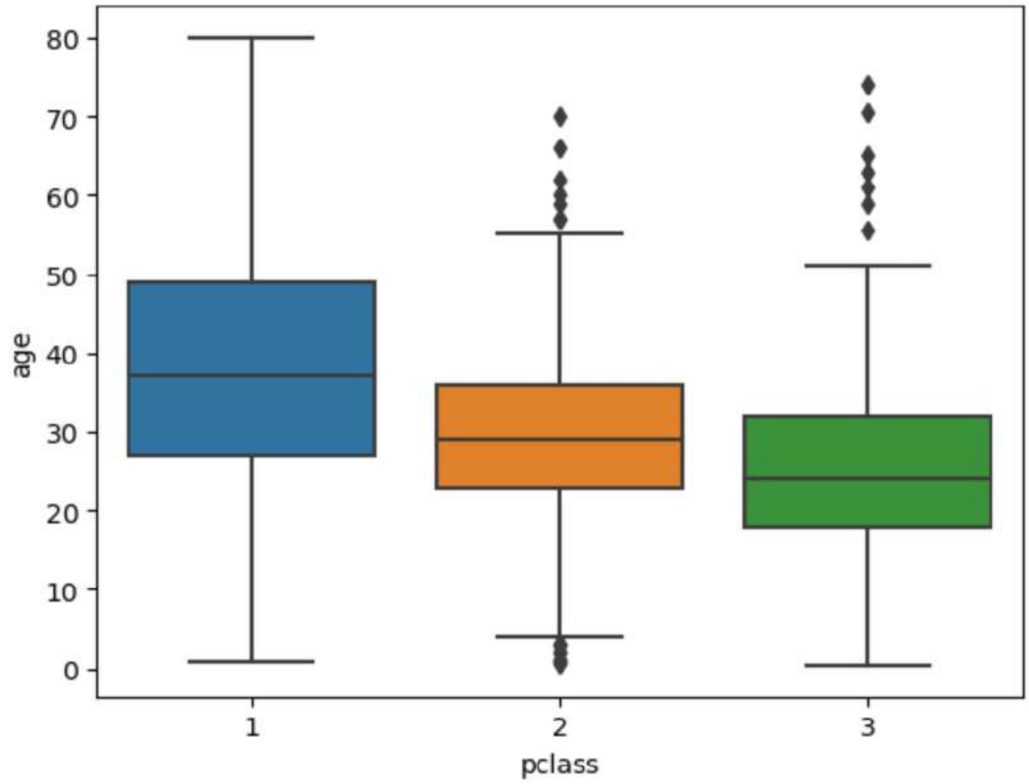
[66]:
```
df_titanic_2=df_titanic.drop(columns='deck')
df_titanic_2.dropna(subset=['embark_town'],axis=0,inplace=True)
```

[67]:
```
df_titanic_2.shape
```

[67]: (889, 14)

[73]:
```
sns.boxplot(data=df_titanic_2, x='pclass', y='age')
```

[73]: <AxesSubplot: xlabel='pclass', ylabel='age'>



0.2 Exercício

Substitua os valores faltantes do atributo age da base de dados pela mediana presentes nos dados da primeira, segunda e terceira classes da base de dados titanic, conforme explicado durante a aula.