In [1]:
```
# Instalando pacotes necessários
!pip install imblearn
```

```
Requirement already satisfied: imblearn in /Users/rpaganini/opt/an
aconda3/lib/python3.8/site-packages (0.0)
Requirement already satisfied: imbalanced-learn in /Users/rpaganin
i/opt/anaconda3/lib/python3.8/site-packages (from imblearn) (0.7.0
)
Requirement already satisfied: scikit-learn>=0.23 in /Users/rpagan
ini/opt/anaconda3/lib/python3.8/site-packages (from imbalanced-lea
rn->imblearn) (0.23.1)
Requirement already satisfied: scipy>=0.19.1 in /Users/rpaganini/o
pt/anaconda3/lib/python3.8/site-packages (from imbalanced-learn->i
mblearn) (1.5.0)
Requirement already satisfied: numpy>=1.13.3 in /Users/rpaganini/o
pt/anaconda3/lib/python3.8/site-packages (from imbalanced-learn->i
mblearn) (1.18.5)
Requirement already satisfied: joblib>=0.11 in /Users/rpaganini/op
t/anaconda3/lib/python3.8/site-packages (from imbalanced-learn->im
blearn) (0.16.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /Users/rpag
anini/opt/anaconda3/lib/python3.8/site-packages (from scikit-learn
>=0.23->imbalanced-learn->imblearn) (2.1.0)
```

In [220]:
```
# Importanto bibliotecas necessárias
import pandas as pd
from statistics import mean, median, mode
import seaborn as sb
from scipy import stats
import numpy as np
from sklearn import preprocessing
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, confusion_matrix, class
ification_report
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from imblearn.over_sampling import SMOTE
from sklearn.linear_model import LogisticRegression
import joblib
```

In [3]:
```
# Carregando o arquivo para um dataset
dataset = pd.read_csv('diabetes.csv')
```

In [4]:
```
# Visualizando o dataset
dataset.head()
```

Out[4]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeFun |
|---|---|---|---|---|---|---|---|
| **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 | |

In [5]:
```
# Visualizando os tipos das variáveis
dataset.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
```

In [6]:
```
# Definindo o tipo correto da variável TARGET.
dataset.Outcome = dataset.Outcome.astype('category')
dataset.Pregnancies = dataset.Pregnancies.astype('object')
dataset.Age = dataset.Age.astype('object')
```

In [7]:
```
dataset.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    object  
 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    object  
 8   Outcome                   768 non-null    category
dtypes: category(1), float64(2), int64(4), object(2)
memory usage: 49.0+ KB
```

In [8]:
```
# Verificando valores faltantes
dataset.isna().sum()
```

Out[8]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [9]:
```
dataset.describe()
```

Out[9]:

| | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigre |
|---|---|---|---|---|---|---|
| **count** | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 768.000000 | 7 |
| **mean** | 120.894531 | 69.105469 | 20.536458 | 79.799479 | 31.992578 | |
| **std** | 31.972618 | 19.355807 | 15.952218 | 115.244002 | 7.884160 | |
| **min** | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | |
| **25%** | 99.000000 | 62.000000 | 0.000000 | 0.000000 | 27.300000 | |
| **50%** | 117.000000 | 72.000000 | 23.000000 | 30.500000 | 32.000000 | |
| **75%** | 140.250000 | 80.000000 | 32.000000 | 127.250000 | 36.600000 | |
| **max** | 199.000000 | 122.000000 | 99.000000 | 846.000000 | 67.100000 | |

In [10]:
```
# Algumas variáveis como: Glucose, BloodPressure, SkinThickness, In
sulin, BMI, DiabetesPedigreeFunction, Age não
# podem conter valores 0 pois são índices de análises sanguíneas pa
ra determinar a diabetes.
```

In [11]:
```
dataset.Glucose.isin([0]).sum()
```

Out[11]: 5

In [12]:
```
dataset.SkinThickness.isin([0]).sum()
```

Out[12]: 227

In [13]:
```
dataset.BloodPressure.isin([0]).sum()
```

Out[13]: 35

In [14]:
```
dataset.Insulin.isin([0]).sum()
```

Out[14]: 374

In [15]:
```
dataset.BMI.isin([0]).sum()
```

Out[15]: 11

In [16]:
```
dataset.DiabetesPedigreeFunction.isin([0]).sum()
```

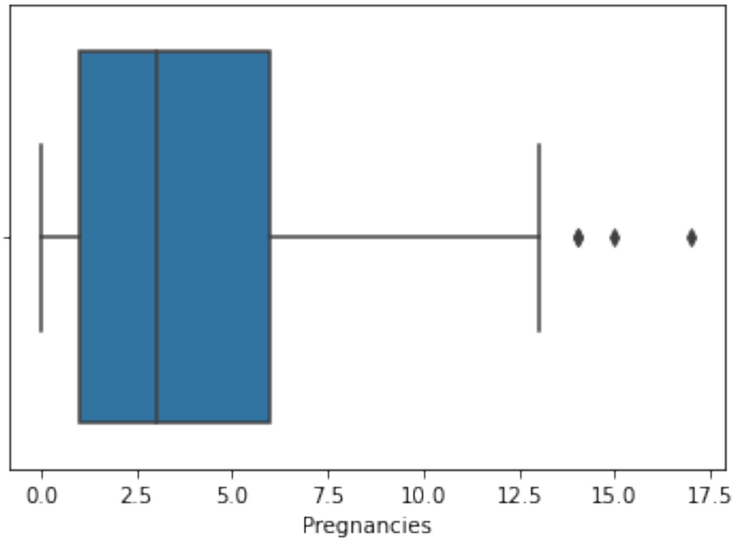
Out[16]: 0

In [17]:
```
# Tratando valores Outliers. Estes valores podem comprometer o algoritmo de ML, portanto devem ser tratados.

# Verificando Outliers individualmente nas variáveis e aplicando a técnica IQR para removê-los.

# Variável 'Pregnancies'
sb.boxplot(dataset.Pregnancies)
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf69ddf700>




In [18]:
```
# Técnica IQR para detectar Outliers aplicada na variável 'Pregnancies'
Q1_Pregnancies = dataset.Pregnancies.quantile(0.25)
Q3_Pregnancies = dataset.Pregnancies.quantile(0.75)
```

In [19]:
```
Q1_Pregnancies , Q3_Pregnancies
```

Out[19]: (1.0, 6.0)

In [20]:
```
IQR_Pregnancies = Q3_Pregnancies - Q1_Pregnancies
```

In [21]:
```
lower_limit_Pregnancies = Q1_Pregnancies - 1.5 * IQR_Pregnancies
upper_limit_Pregnancies = Q3_Pregnancies + 1.5 * IQR_Pregnancies
```

In [22]:
```
lower_limit_Pregnancies , upper_limit_Pregnancies
```

Out[22]: (-6.5, 13.5)

In [23]:
```
dataset[(dataset.Pregnancies < lower_limit_Pregnancies) | (dataset.
Pregnancies > upper_limit_Pregnancies)]
```

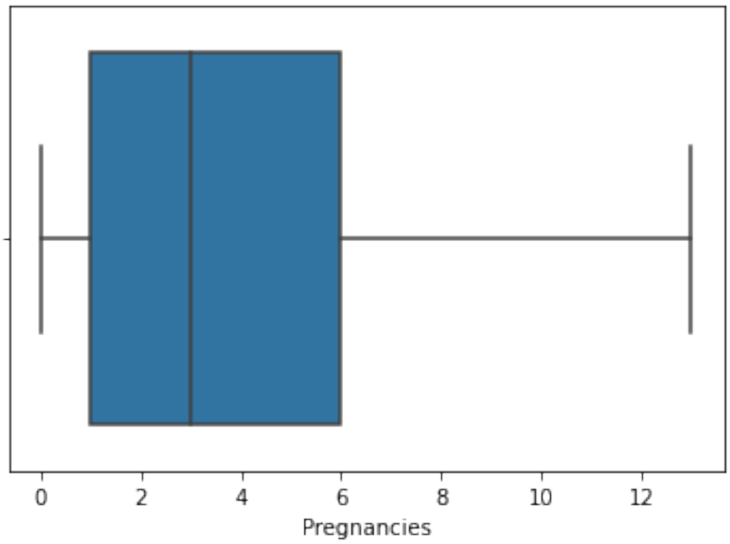
Out[23]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **88** | 15 | 136 | 70 | 32 | 110 | 37.1 | |
| **159** | 17 | 163 | 72 | 41 | 114 | 40.9 | |
| **298** | 14 | 100 | 78 | 25 | 184 | 36.6 | |
| **455** | 14 | 175 | 62 | 30 | 0 | 33.6 | |

In [24]:
```
dataset = dataset[(dataset.Pregnancies > lower_limit_Pregnancies) &
(dataset.Pregnancies < upper_limit_Pregnancies)]
```

In [25]:
```
sb.boxplot(dataset.Pregnancies)
```

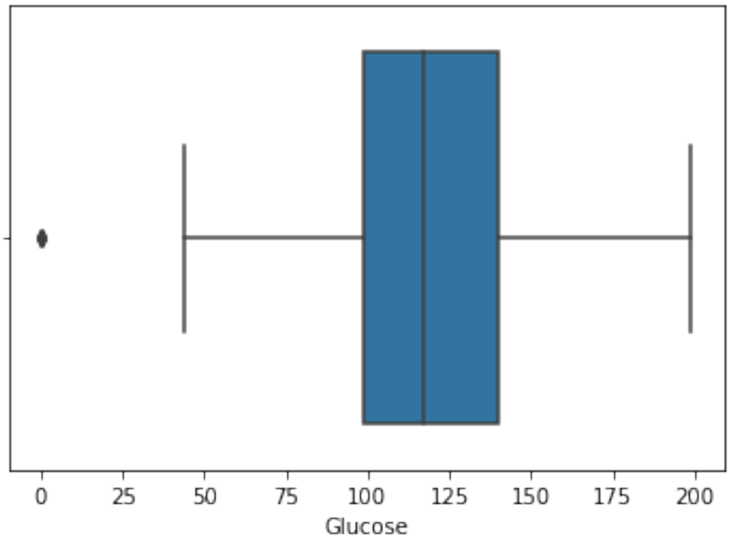
Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf69dd8280>




In [26]:
```
# Perfect !!! Outliers da variável 'Pregnancie' removidos com suces
so !!
# Aplicando a técnica de detecção e remocação de Outliers (IQR) nas
demais variáveis.
```

In [27]:
```
# Variável 'Glucose'
sb.boxplot(dataset.Glucose)
```

Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf69f71160>




In [28]:
```
Q1_Glucose = dataset.Glucose.quantile(.25)
Q3_Glucose = dataset.Glucose.quantile(.75)

IQR_Glucose = Q3_Glucose - Q1_Glucose

lower_limit_Glucose = Q1_Glucose - 1.5 * IQR_Glucose
upper_limit_Glucose = Q3_Glucose + 1.5 * IQR_Glucose
```

In [29]:
```
dataset[(dataset.Glucose < lower_limit_Glucose) | (dataset.Glucose
> upper_limit_Glucose)]
```

Out[29]:

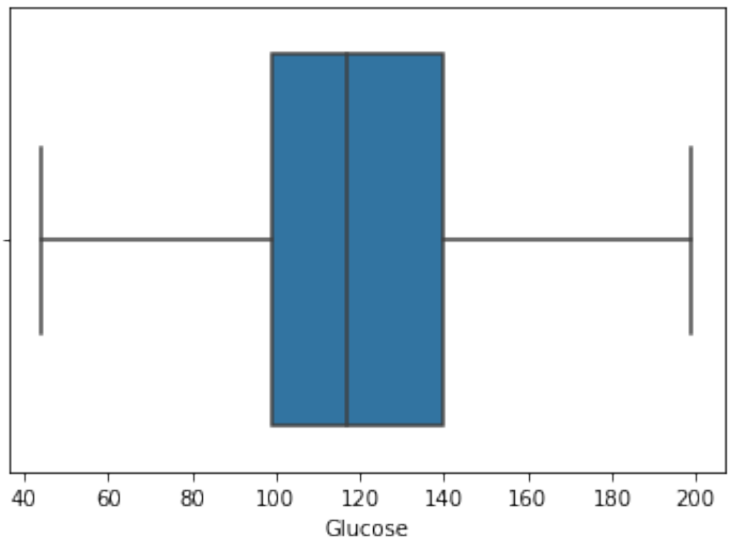
| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **75** | 1 | 0 | 48 | 20 | 0 | 24.7 | |
| **182** | 1 | 0 | 74 | 20 | 23 | 27.7 | |
| **342** | 1 | 0 | 68 | 35 | 0 | 32.0 | |
| **349** | 5 | 0 | 80 | 32 | 0 | 41.0 | |
| **502** | 6 | 0 | 68 | 41 | 0 | 39.0 | |

In [30]:
```
# Observe que o Outlier da variável 'Glucose'. Não existem exames c
omo este com valores 0. Isso indica claramente
# que os valores nesta variável não foram digitados. Teremos que ab
ordar esse problema de outra forma. Aplicarei
# uma técnica para preencher os valores 0 desta variável com a médi
a da mesma.
```

In [31]:
```
dataset.Glucose = dataset.Glucose.replace([0],[mean(dataset.Glucose
)])
```

In [32]:
```
sb.boxplot(dataset.Glucose)
```

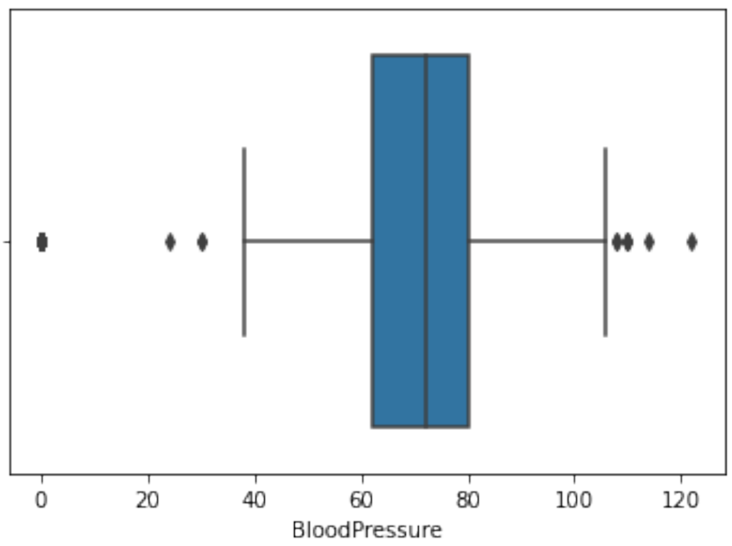
Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a042340>




In [33]:
```
# Mais um problema resolvido. Vamos partir para a próxima variável.
'BloodPressure'
```

In [34]:
```
sb.boxplot(dataset.BloodPressure)
```

Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a1043a0>



In [35]:
```
Q1_BloodPressure = dataset.BloodPressure.quantile(.25)
Q3_BloodPressure = dataset.BloodPressure.quantile(.75)

IQR_BloodPressure = Q3_BloodPressure - Q1_BloodPressure

lower_limit_BloodPressure = Q1_BloodPressure - 1.5 * IQR_BloodPress
ure
upper_limit_BloodPressure = Q3_BloodPressure + 1.5 * IQR_BloodPress
ure
```

In [36]:
```
dataset[(dataset.BloodPressure < lower_limit_BloodPressure) | (data
set.BloodPressure > upper_limit_BloodPressure)]
```

Out[36]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **7** | 10 | 115.0 | 0 | 0 | 0 | 35.3 | |
| **15** | 7 | 100.0 | 0 | 0 | 0 | 30.0 | |
| **18** | 1 | 103.0 | 30 | 38 | 83 | 43.3 | |
| **43** | 9 | 171.0 | 110 | 24 | 240 | 45.4 | |
| **49** | 7 | 105.0 | 0 | 0 | 0 | 0.0 | |
| **60** | 2 | 84.0 | 0 | 0 | 0 | 0.0 | |
| **78** | 0 | 131.0 | 0 | 0 | 0 | 43.2 | |
| **81** | 2 | 74.0 | 0 | 0 | 0 | 0.0 | |
| **84** | 5 | 137.0 | 108 | 0 | 0 | 48.8 | |
| **106** | 1 | 96.0 | 122 | 0 | 0 | 22.4 | |
| **125** | 1 | 88.0 | 30 | 42 | 99 | 55.0 | |
| **172** | 2 | 87.0 | 0 | 23 | 0 | 28.9 | |
| **177** | 0 | 129.0 | 110 | 46 | 130 | 67.1 | |
| **193** | 11 | 135.0 | 0 | 0 | 0 | 52.3 | |
| **222** | 7 | 119.0 | 0 | 0 | 0 | 25.2 | |
| **261** | 3 | 141.0 | 0 | 0 | 0 | 30.0 | |
| **266** | 0 | 138.0 | 0 | 0 | 0 | 36.3 | |
| **269** | 2 | 146.0 | 0 | 0 | 0 | 27.5 | |
| **300** | 0 | 167.0 | 0 | 0 | 0 | 32.3 | |
| **332** | 1 | 180.0 | 0 | 0 | 0 | 43.3 | |
| **336** | 0 | 117.0 | 0 | 0 | 0 | 33.8 | |
| **347** | 3 | 116.0 | 0 | 0 | 0 | 23.5 | |
| **357** | 13 | 129.0 | 0 | 30 | 0 | 39.9 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **362** | 5 | 103.0 | 108 | 37 | 0 | 39.2 |
| **426** | 0 | 94.0 | 0 | 0 | 0 | 0.0 |
| **430** | 2 | 99.0 | 0 | 0 | 0 | 22.2 |
| **435** | 0 | 141.0 | 0 | 0 | 0 | 42.4 |
| **453** | 2 | 119.0 | 0 | 0 | 0 | 19.6 |
| **468** | 8 | 120.0 | 0 | 0 | 0 | 30.0 |
| **484** | 0 | 145.0 | 0 | 0 | 0 | 44.2 |
| **494** | 3 | 80.0 | 0 | 0 | 0 | 0.0 |
| **522** | 6 | 114.0 | 0 | 0 | 0 | 0.0 |
| **533** | 6 | 91.0 | 0 | 0 | 0 | 29.8 |
| **535** | 4 | 132.0 | 0 | 0 | 0 | 32.9 |
| **549** | 4 | 189.0 | 110 | 31 | 0 | 28.5 |
| **589** | 0 | 73.0 | 0 | 0 | 0 | 21.1 |
| **597** | 1 | 89.0 | 24 | 19 | 25 | 27.8 |
| **601** | 6 | 96.0 | 0 | 0 | 0 | 23.7 |
| **604** | 4 | 183.0 | 0 | 0 | 0 | 28.4 |
| **619** | 0 | 119.0 | 0 | 0 | 0 | 32.4 |
| **643** | 4 | 90.0 | 0 | 0 | 0 | 28.0 |
| **691** | 13 | 158.0 | 114 | 0 | 0 | 42.3 |
| **697** | 0 | 99.0 | 0 | 0 | 0 | 25.0 |
| **703** | 2 | 129.0 | 0 | 0 | 0 | 38.5 |
| **706** | 10 | 115.0 | 0 | 0 | 0 | 0.0 |

In [37]:
```
# Observem acima que a variável 'BloodPressure' também possui valor
es 0 e vários Outliers. Teremos que aplicar as
# duas técnicas (IQR e Preencher valores 0) Let's do it !
```

In [38]:
```
dataset.BloodPressure = dataset.BloodPressure.replace([0],[mean(dat
aset.BloodPressure)])
```

In [39]:
```
Q1_BloodPressure = dataset.BloodPressure.quantile(.25)
Q3_BloodPressure = dataset.BloodPressure.quantile(.75)

IQR_BloodPressure = Q3_BloodPressure - Q1_BloodPressure

lower_limit_BloodPressure = Q1_BloodPressure - 1.5 * IQR_BloodPress
ure
upper_limit_BloodPressure = Q3_BloodPressure + 1.5 * IQR_BloodPress
ure
```

In [40]:
```
dataset[(dataset.BloodPressure < lower_limit_BloodPressure) | (data
set.BloodPressure > upper_limit_BloodPressure)]
```

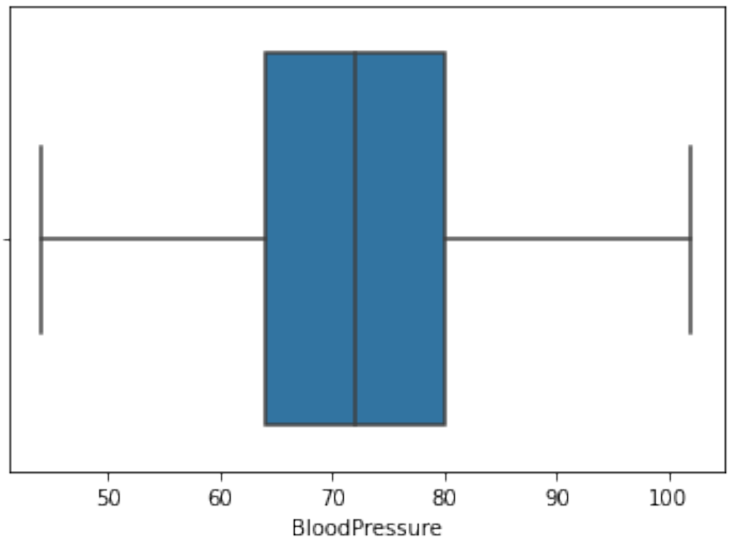
Out[40]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **18** | 1 | 103.0 | 30.0 | 38 | 83 | 43.3 | |
| **43** | 9 | 171.0 | 110.0 | 24 | 240 | 45.4 | |
| **84** | 5 | 137.0 | 108.0 | 0 | 0 | 48.8 | |
| **106** | 1 | 96.0 | 122.0 | 0 | 0 | 22.4 | |
| **125** | 1 | 88.0 | 30.0 | 42 | 99 | 55.0 | |
| **177** | 0 | 129.0 | 110.0 | 46 | 130 | 67.1 | |
| **362** | 5 | 103.0 | 108.0 | 37 | 0 | 39.2 | |
| **549** | 4 | 189.0 | 110.0 | 31 | 0 | 28.5 | |
| **597** | 1 | 89.0 | 24.0 | 19 | 25 | 27.8 | |
| **599** | 1 | 109.0 | 38.0 | 18 | 120 | 23.1 | |
| **658** | 11 | 127.0 | 106.0 | 0 | 0 | 39.0 | |
| **662** | 8 | 167.0 | 106.0 | 46 | 231 | 37.6 | |
| **672** | 10 | 68.0 | 106.0 | 23 | 49 | 35.5 | |
| **691** | 13 | 158.0 | 114.0 | 0 | 0 | 42.3 | |

In [41]:
```
dataset = dataset[(dataset.BloodPressure > lower_limit_BloodPressur
e) & (dataset.BloodPressure < upper_limit_BloodPressure)]
```

In [42]:
```
sb.boxplot(dataset.BloodPressure)
```

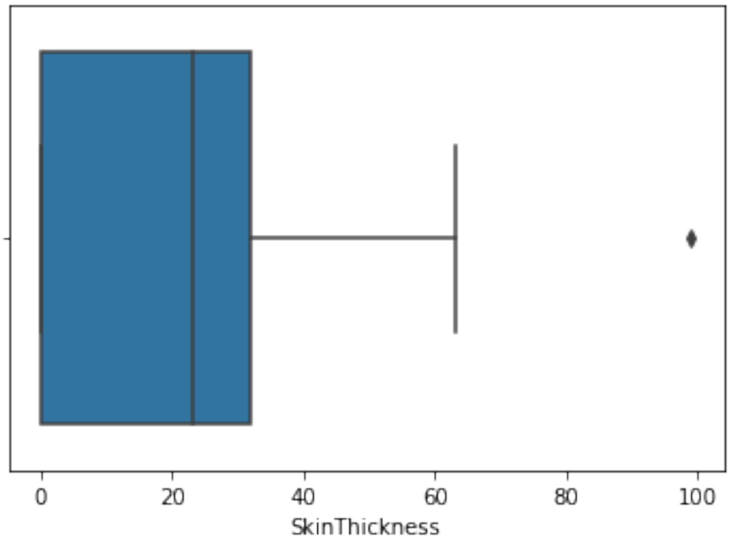
Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a1ccfd0>



In [43]:
```
# Variável 'SkinThickness'
```

In [44]:
```
sb.boxplot(dataset.SkinThickness)
```

Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a2a8190>




In [45]:
```
Q1_SkinThickness = dataset.SkinThickness.quantile(.25)
Q3_SkinThickness = dataset.SkinThickness.quantile(.75)

IQR_SkinThickness = Q3_SkinThickness - Q1_SkinThickness

lower_limit_SkinThickness = Q1_SkinThickness - 1.5 * IQR_SkinThickn
ess
upper_limit_SkinThickness = Q3_SkinThickness + 1.5 * IQR_SkinThickn
ess
```

In [46]:
```
dataset[(dataset.SkinThickness < lower_limit_SkinThickness) | (data
set.SkinThickness > upper_limit_SkinThickness)]
```

Out[46]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **579** | 2 | 197.0 | 70.0 | 99 | 0 | 34.7 | |

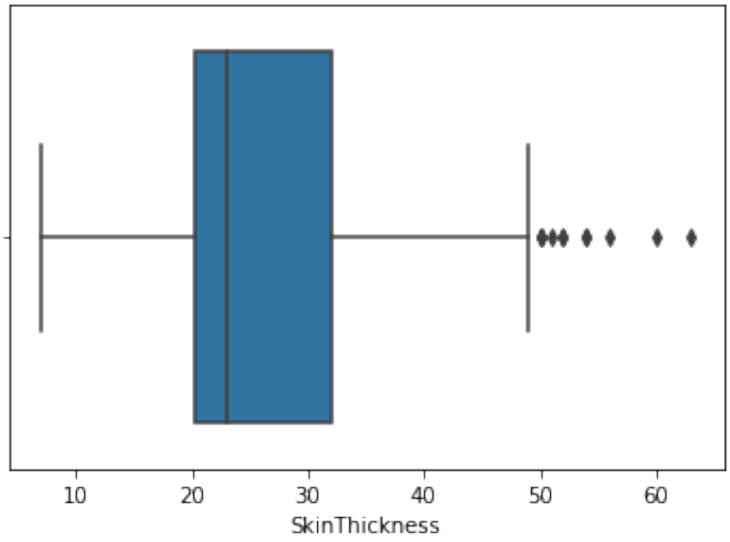
In [47]:
```
dataset = dataset[(dataset.SkinThickness > lower_limit_SkinThicknes
s) & (dataset.SkinThickness < upper_limit_SkinThickness)]
```

In [48]:
```
# Mesmo não sendo um outlier, esta variável também possui valores 0
. Devemos tratar esses valores
```

In [49]:
```
dataset.SkinThickness = dataset.SkinThickness.replace([0],[mean(dat
aset.SkinThickness)])
```

In [50]:
```
sb.boxplot(dataset.SkinThickness)
```

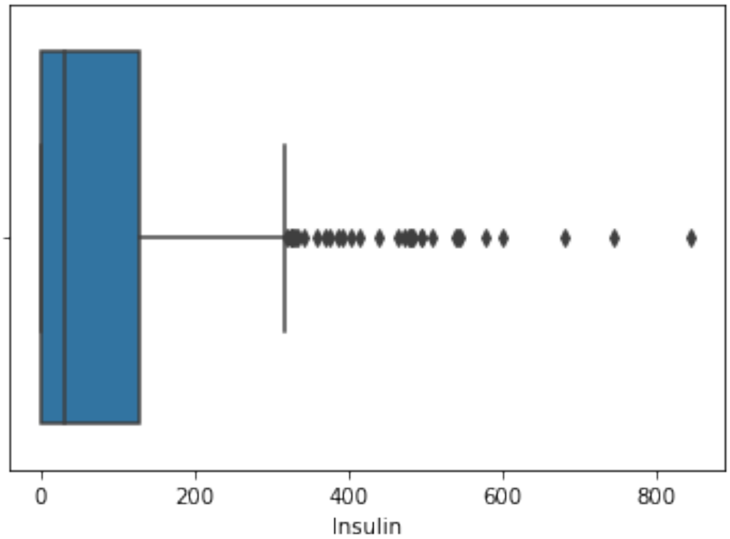
Out[50]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a365550>
```




In [51]:
```
# Next ;-) variável 'Insulin'
```

In [52]:
```
sb.boxplot(dataset.Insulin)
```

Out[52]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a428a90>
```




In [53]:
```
Q1_Insulin = dataset.Insulin.quantile(.25)
Q3_Insulin = dataset.Insulin.quantile(.75)

IQR_Insulin = Q3_Insulin - Q1_Insulin

lower_limit_Insulin = Q1_Insulin - 1.5 * IQR_Insulin
upper_limit_Insulin = Q3_Insulin + 1.5 * IQR_Insulin
```

In [54]:
```
dataset[(dataset.Insulin < lower_limit_Insulin) | (dataset.Insulin
> upper_limit_Insulin)]
```

Out[54]:

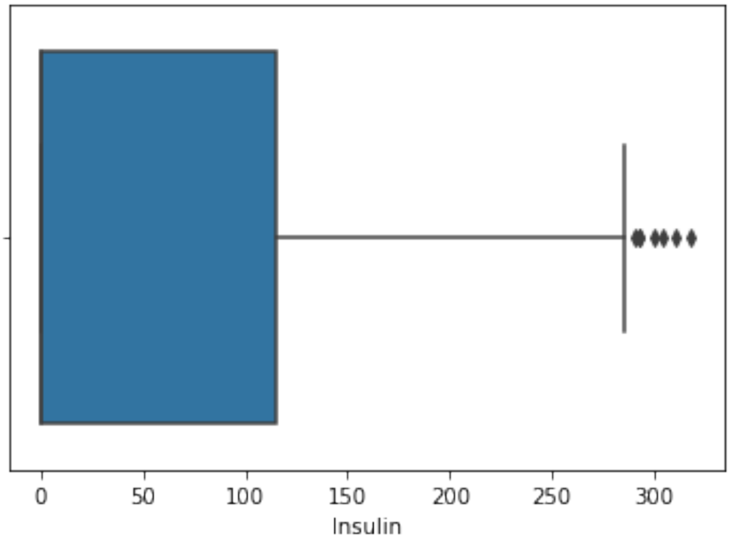
| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **8** | 2 | 197.0 | 70.0 | 45.0 | 543 | 30.5 | |
| **13** | 1 | 189.0 | 60.0 | 23.0 | 846 | 30.1 | |
| **54** | 7 | 150.0 | 66.0 | 42.0 | 342 | 34.7 | |
| **111** | 8 | 155.0 | 62.0 | 26.0 | 495 | 34.0 | |
| **139** | 5 | 105.0 | 72.0 | 29.0 | 325 | 36.9 | |
| **153** | 1 | 153.0 | 82.0 | 42.0 | 485 | 40.6 | |
| **186** | 8 | 181.0 | 68.0 | 36.0 | 495 | 30.1 | |
| **220** | 0 | 177.0 | 60.0 | 29.0 | 478 | 34.6 | |
| **228** | 4 | 197.0 | 70.0 | 39.0 | 744 | 36.7 | |
| **231** | 6 | 134.0 | 80.0 | 37.0 | 370 | 46.2 | |
| **247** | 0 | 165.0 | 90.0 | 33.0 | 680 | 52.3 | |
| **248** | 9 | 124.0 | 70.0 | 33.0 | 402 | 35.4 | |
| **258** | 1 | 193.0 | 50.0 | 16.0 | 375 | 25.9 | |
| **286** | 5 | 155.0 | 84.0 | 44.0 | 545 | 38.7 | |
| **296** | 2 | 146.0 | 70.0 | 38.0 | 360 | 28.0 | |
| **360** | 5 | 189.0 | 64.0 | 33.0 | 325 | 31.2 | |
| **370** | 3 | 173.0 | 82.0 | 48.0 | 465 | 38.4 | |
| **375** | 12 | 140.0 | 82.0 | 43.0 | 325 | 39.2 | |
| **392** | 1 | 131.0 | 64.0 | 14.0 | 415 | 23.7 | |
| **409** | 1 | 172.0 | 68.0 | 49.0 | 579 | 42.4 | |
| **415** | 3 | 173.0 | 84.0 | 33.0 | 474 | 35.7 | |
| **480** | 3 | 158.0 | 70.0 | 30.0 | 328 | 35.5 | |
| **486** | 1 | 139.0 | 62.0 | 41.0 | 480 | 40.7 | |
| **519** | 6 | 129.0 | 90.0 | 7.0 | 326 | 19.6 | |
| **574** | 1 | 143.0 | 86.0 | 30.0 | 330 | 30.1 | |
| **584** | 8 | 124.0 | 76.0 | 24.0 | 600 | 28.7 | |
| **612** | 7 | 168.0 | 88.0 | 42.0 | 321 | 38.2 | |
| **645** | 2 | 157.0 | 74.0 | 35.0 | 440 | 39.4 | |
| **655** | 2 | 155.0 | 52.0 | 27.0 | 540 | 38.7 | |
| **695** | 7 | 142.0 | 90.0 | 24.0 | 480 | 30.4 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| **707** | 2 | 127.0 | 46.0 | 21.0 | 335 | 34.4 |
| **710** | 3 | 158.0 | 64.0 | 13.0 | 387 | 31.2 |
| **715** | 7 | 187.0 | 50.0 | 33.0 | 392 | 33.9 |
| **753** | 0 | 181.0 | 88.0 | 44.0 | 510 | 43.3 |

In [55]:
```
dataset = dataset[(dataset.Insulin > lower_limit_Insulin) & (dataset.Insulin < upper_limit_Insulin)]
```

In [56]:
```
sb.boxplot(dataset.Insulin)
```

Out[56]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a1f3400>




In [57]:
```
dataset[(dataset.Insulin < lower_limit_Insulin) | (dataset.Insulin > upper_limit_Insulin)]
```

Out[57]:

| Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeFunc |
|---|---|---|---|---|---|---|

In [58]:
```
# Observem que ainda temos valores Outliers. Para resolver esse problema precisaremos alterar o fator da técnica de
# detecção e remoção de Outliers IQR
```

In [59]:
```
lower_limit_Insulin = Q1_Insulin - 1 * IQR_Insulin
upper_limit_Insulin = Q3_Insulin + 1 * IQR_Insulin
```

In [60]:
```
dataset[(dataset.Insulin < lower_limit_Insulin) | (dataset.Insulin
> upper_limit_Insulin)]
```

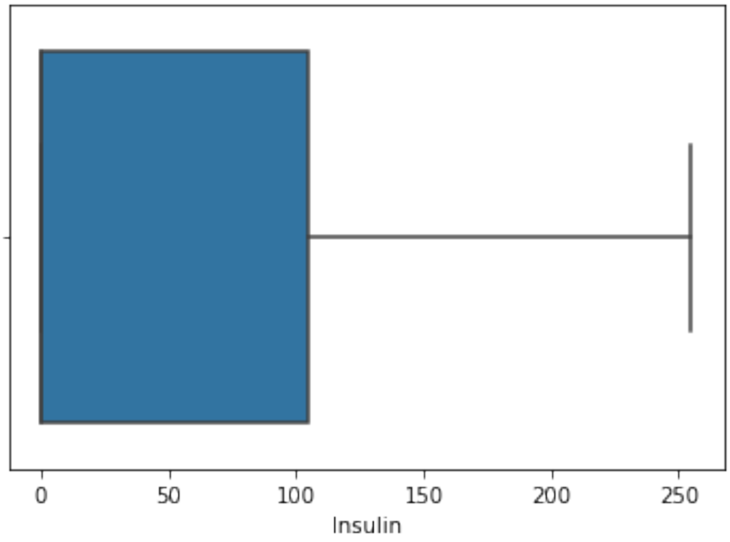
Out[60]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **53** | 8 | 176.0 | 90.0 | 34.0 | 300 | 33.7 | |
| **56** | 7 | 187.0 | 68.0 | 39.0 | 304 | 37.7 | |
| **73** | 4 | 129.0 | 86.0 | 20.0 | 270 | 35.1 | |
| **144** | 4 | 154.0 | 62.0 | 31.0 | 284 | 32.8 | |
| **162** | 0 | 114.0 | 80.0 | 34.0 | 285 | 44.2 | |
| **199** | 4 | 148.0 | 60.0 | 27.0 | 318 | 30.9 | |
| **206** | 8 | 196.0 | 76.0 | 29.0 | 280 | 37.5 | |
| **215** | 12 | 151.0 | 70.0 | 40.0 | 271 | 41.8 | |
| **254** | 12 | 92.0 | 62.0 | 7.0 | 258 | 27.6 | |
| **279** | 2 | 108.0 | 62.0 | 10.0 | 278 | 25.3 | |
| **364** | 4 | 147.0 | 74.0 | 25.0 | 293 | 34.9 | |
| **388** | 5 | 144.0 | 82.0 | 26.0 | 285 | 32.0 | |
| **395** | 2 | 127.0 | 58.0 | 24.0 | 275 | 27.7 | |
| **412** | 1 | 143.0 | 84.0 | 23.0 | 310 | 42.4 | |
| **425** | 4 | 184.0 | 78.0 | 39.0 | 277 | 37.0 | |
| **487** | 0 | 173.0 | 78.0 | 32.0 | 265 | 46.5 | |
| **561** | 0 | 198.0 | 66.0 | 32.0 | 274 | 41.3 | |
| **606** | 1 | 181.0 | 78.0 | 42.0 | 293 | 40.0 | |
| **608** | 0 | 152.0 | 82.0 | 39.0 | 272 | 41.5 | |
| **679** | 2 | 101.0 | 58.0 | 17.0 | 265 | 24.2 | |
| **713** | 0 | 134.0 | 58.0 | 20.0 | 291 | 26.4 | |

In [61]:
```
dataset = dataset[(dataset.Insulin > lower_limit_Insulin) & (datase
t.Insulin < upper_limit_Insulin)]
```

In [62]:
```
sb.boxplot(dataset.Insulin)
```

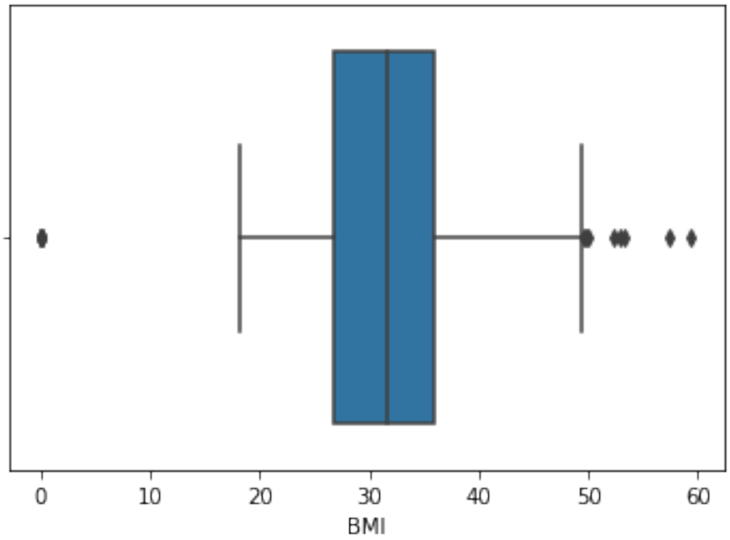
Out[62]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a6b0400>




In [63]:
```
# Próxima variável 'BMI'
```

In [64]:
```
sb.boxplot(dataset.BMI)
```

Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a886d30>




In [65]:
```
Q1_BMI = dataset.BMI.quantile(.25)
Q3_BMI = dataset.BMI.quantile(.75)

IQR_BMI = Q3_BMI - Q1_BMI

lower_limit_BMI = Q1_BMI - 1.5 * IQR_BMI
upper_limit_BMI = Q3_BMI + 1.5 * IQR_BMI
```

In [66]:
```
dataset[(dataset.BMI < lower_limit_BMI) | (dataset.BMI > upper_limi
t_BMI)]
```

Out[66]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **9** | 8 | 125.0 | 96.000000 | 20.323056 | 0 | 0.0 | |
| **49** | 7 | 105.0 | 69.098168 | 20.323056 | 0 | 0.0 | |
| **60** | 2 | 84.0 | 69.098168 | 20.323056 | 0 | 0.0 | |
| **81** | 2 | 74.0 | 69.098168 | 20.323056 | 0 | 0.0 | |
| **99** | 1 | 122.0 | 90.000000 | 51.000000 | 220 | 49.7 | |
| **120** | 0 | 162.0 | 76.000000 | 56.000000 | 100 | 53.2 | |
| **145** | 0 | 102.0 | 75.000000 | 23.000000 | 0 | 0.0 | |
| **155** | 7 | 152.0 | 88.000000 | 44.000000 | 0 | 50.0 | |
| **193** | 11 | 135.0 | 69.098168 | 20.323056 | 0 | 52.3 | |
| **303** | 5 | 115.0 | 98.000000 | 20.323056 | 0 | 52.9 | |
| **371** | 0 | 118.0 | 64.000000 | 23.000000 | 89 | 0.0 | |
| **426** | 0 | 94.0 | 69.098168 | 20.323056 | 0 | 0.0 | |
| **445** | 0 | 180.0 | 78.000000 | 63.000000 | 14 | 59.4 | |
| **494** | 3 | 80.0 | 69.098168 | 20.323056 | 0 | 0.0 | |
| **522** | 6 | 114.0 | 69.098168 | 20.323056 | 0 | 0.0 | |
| **673** | 3 | 123.0 | 100.000000 | 35.000000 | 240 | 57.3 | |
| **681** | 0 | 162.0 | 76.000000 | 36.000000 | 0 | 49.6 | |
| **684** | 5 | 136.0 | 82.000000 | 20.323056 | 0 | 0.0 | |
| **706** | 10 | 115.0 | 69.098168 | 20.323056 | 0 | 0.0 | |

In [67]:
```
# Observem que esta variável também possui valores 0. Teremos que t
ratar esses valores da mesma forma como foi na
# variáveis 'BloodPressure' e 'Glucose'
```

In [68]:
```
dataset.BMI = dataset.BMI.replace([0],[mean(dataset.BMI)])
```

In [69]:
```
dataset[(dataset.BMI < lower_limit_BMI) | (dataset.BMI > upper_limi
t_BMI)]
```

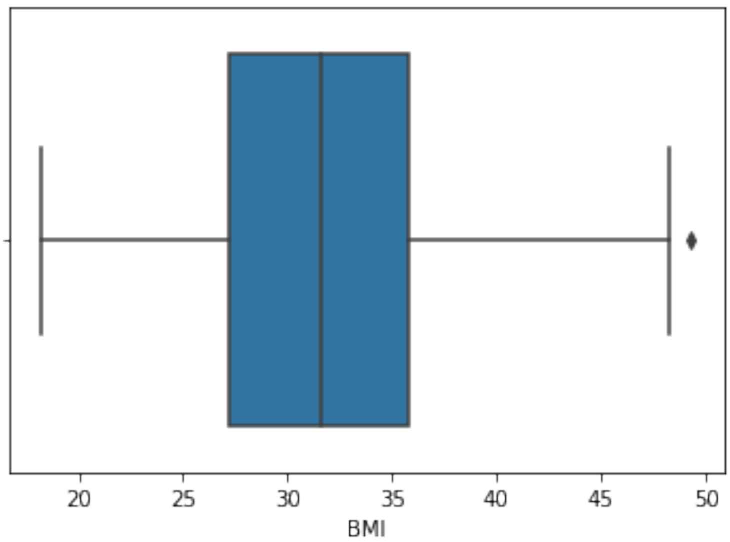
Out[69]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **99** | 1 | 122.0 | 90.000000 | 51.000000 | 220 | 49.7 | |
| **120** | 0 | 162.0 | 76.000000 | 56.000000 | 100 | 53.2 | |
| **155** | 7 | 152.0 | 88.000000 | 44.000000 | 0 | 50.0 | |
| **193** | 11 | 135.0 | 69.098168 | 20.323056 | 0 | 52.3 | |
| **303** | 5 | 115.0 | 98.000000 | 20.323056 | 0 | 52.9 | |
| **445** | 0 | 180.0 | 78.000000 | 63.000000 | 14 | 59.4 | |
| **673** | 3 | 123.0 | 100.000000 | 35.000000 | 240 | 57.3 | |
| **681** | 0 | 162.0 | 76.000000 | 36.000000 | 0 | 49.6 | |

In [70]:
```
dataset = dataset[(dataset.BMI > lower_limit_BMI) & (dataset.BMI <
upper_limit_BMI)]
```

In [71]:
```
sb.boxplot(dataset.BMI)
```

Out[71]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a698a30>




In [72]:
```
# Ainda temos Outliers.Devemos alterar o valor do fator.
lower_limit_BMI = Q1_BMI - 1.25 * IQR_BMI
upper_limit_BMI = Q3_BMI + 1.25 * IQR_BMI
```

In [73]:
```
dataset[(dataset.BMI < lower_limit_BMI) | (dataset.BMI > upper_limit_BMI)]
```

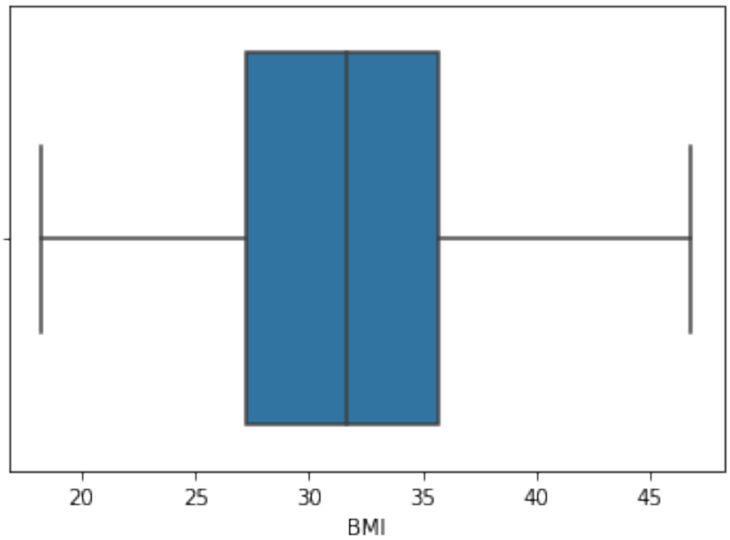
Out[73]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **154** | 8 | 188.0 | 78.0 | 20.323056 | 0 | 47.9 | |
| **335** | 0 | 165.0 | 76.0 | 43.000000 | 255 | 47.9 | |
| **378** | 4 | 156.0 | 75.0 | 20.323056 | 0 | 48.3 | |
| **746** | 1 | 147.0 | 94.0 | 41.000000 | 0 | 49.3 | |

In [74]:
```
dataset = dataset[(dataset.BMI > lower_limit_BMI) & (dataset.BMI < upper_limit_BMI)]
```

In [75]:
```
sb.boxplot(dataset.BMI)
```

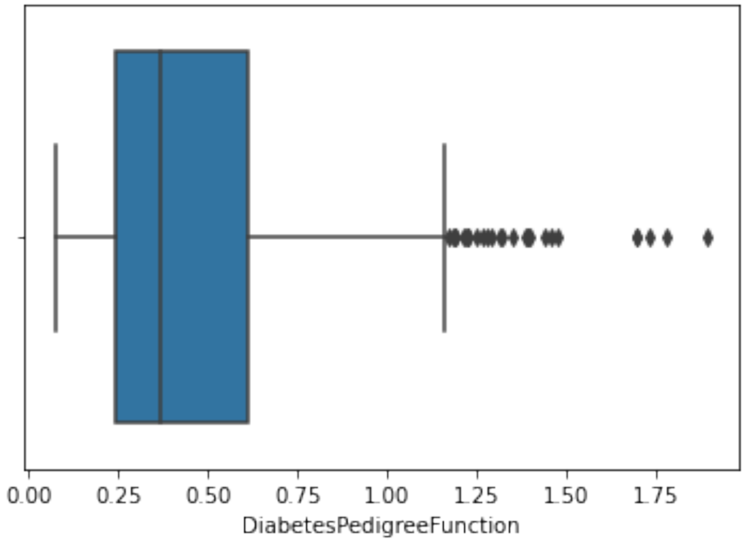
Out[75]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6a9f7e80>




In [76]:
```
# Perfect !!! Próxima variável 'DiabetesPedigreeFunction'
```

In [77]:
```
sb.boxplot(dataset.DiabetesPedigreeFunction)
```

Out[77]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6aac5c40>




In [78]:
```
Q1_DiabetesPedigreeFunction = dataset.DiabetesPedigreeFunction.quan
tile(.25)
Q3_DiabetesPedigreeFunction = dataset.DiabetesPedigreeFunction.quan
tile(.75)

IQR_DiabetesPedigreeFunction = Q3_DiabetesPedigreeFunction - Q1_Dia
betesPedigreeFunction

lower_limit_DiabetesPedigreeFunction = Q1_DiabetesPedigreeFunction
- 1.5 * IQR_DiabetesPedigreeFunction
upper_limit_DiabetesPedigreeFunction = Q3_DiabetesPedigreeFunction
+ 1.5 * IQR_DiabetesPedigreeFunction
```

In [79]:
```
dataset[(dataset.DiabetesPedigreeFunction < lower_limit_DiabetesPed
igreeFunction) | (dataset.DiabetesPedigreeFunction > upper_limit_Di
abetesPedigreeFunction)]
```

Out[79]:

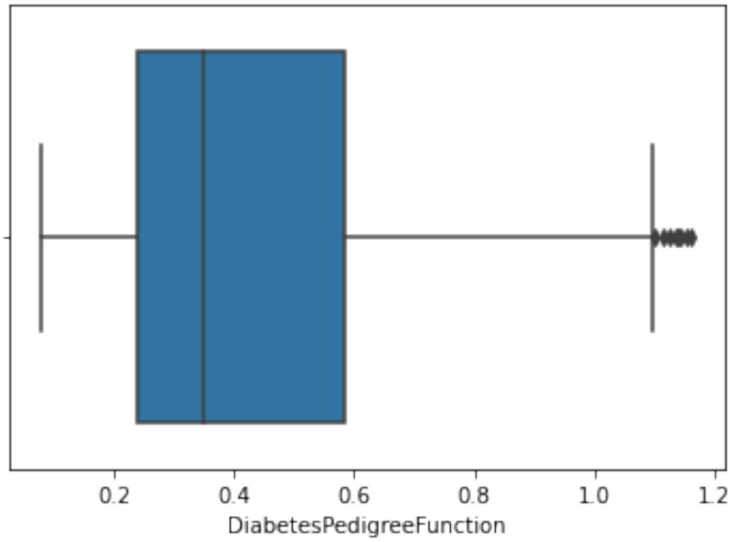
| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedi |
|---|---|---|---|---|---|---|---|
| **12** | 10 | 139.0 | 80.0 | 20.323056 | 0 | 27.100000 | |
| **39** | 4 | 111.0 | 72.0 | 47.000000 | 207 | 37.100000 | |
| **45** | 0 | 180.0 | 66.0 | 39.000000 | 0 | 42.000000 | |
| **58** | 0 | 146.0 | 82.0 | 20.323056 | 0 | 40.500000 | |
| **100** | 1 | 163.0 | 72.0 | 20.323056 | 0 | 39.000000 | |
| **147** | 2 | 106.0 | 64.0 | 35.000000 | 119 | 30.500000 | |
| **152** | 9 | 156.0 | 86.0 | 28.000000 | 155 | 34.300000 | |
| **187** | 1 | 128.0 | 98.0 | 41.000000 | 58 | 32.000000 | |
| **218** | 5 | 85.0 | 74.0 | 22.000000 | 0 | 29.000000 | |
| **243** | 6 | 119.0 | 50.0 | 22.000000 | 176 | 27.100000 | |
| **245** | 9 | 184.0 | 85.0 | 15.000000 | 0 | 30.000000 | |
| **259** | 11 | 155.0 | 76.0 | 28.000000 | 150 | 33.300000 | |
| **292** | 2 | 128.0 | 78.0 | 37.000000 | 182 | 43.300000 | |
| **308** | 0 | 128.0 | 68.0 | 19.000000 | 180 | 30.500000 | |
| **330** | 8 | 118.0 | 72.0 | 19.000000 | 0 | 23.100000 | |
| **371** | 0 | 118.0 | 64.0 | 23.000000 | 89 | 31.530246 | |
| **383** | 1 | 90.0 | 62.0 | 18.000000 | 59 | 25.100000 | |
| **408** | 8 | 197.0 | 74.0 | 20.323056 | 0 | 25.900000 | |
| **534** | 1 | 77.0 | 56.0 | 30.000000 | 56 | 33.300000 | |
| **593** | 2 | 82.0 | 52.0 | 22.000000 | 115 | 28.500000 | |
| **618** | 9 | 112.0 | 82.0 | 24.000000 | 0 | 28.200000 | |
| **621** | 2 | 92.0 | 76.0 | 20.000000 | 0 | 24.200000 | |
| **622** | 6 | 183.0 | 94.0 | 20.323056 | 0 | 40.800000 | |
| **659** | 3 | 80.0 | 82.0 | 31.000000 | 70 | 34.200000 | |
| **661** | 1 | 199.0 | 76.0 | 43.000000 | 0 | 42.900000 | |
| **744** | 13 | 153.0 | 88.0 | 37.000000 | 140 | 40.600000 | |
| **750** | 4 | 136.0 | 70.0 | 20.323056 | 0 | 31.200000 | |

In [80]:

```
dataset = dataset[(dataset.DiabetesPedigreeFunction > lower_limit_DiabetesPedigreeFunction) & (dataset.DiabetesPedigreeFunction < upper_limit_DiabetesPedigreeFunction)]
```

In [81]:
```
sb.boxplot(dataset.DiabetesPedigreeFunction)
```

Out[81]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6ab9f400>




In [82]:
```
# Ainda temos Outliers.Devemos alterar o valor do fator.
lower_limit_DiabetesPedigreeFunction = Q1_DiabetesPedigreeFunction
- 1.20 * IQR_DiabetesPedigreeFunction
upper_limit_DiabetesPedigreeFunction = Q3_DiabetesPedigreeFunction
+ 1.20 * IQR_DiabetesPedigreeFunction
```

In [83]:
```
dataset[(dataset.DiabetesPedigreeFunction < lower_limit_DiabetesPed
igreeFunction) | (dataset.DiabetesPedigreeFunction > upper_limit_Di
abetesPedigreeFunction)]
```

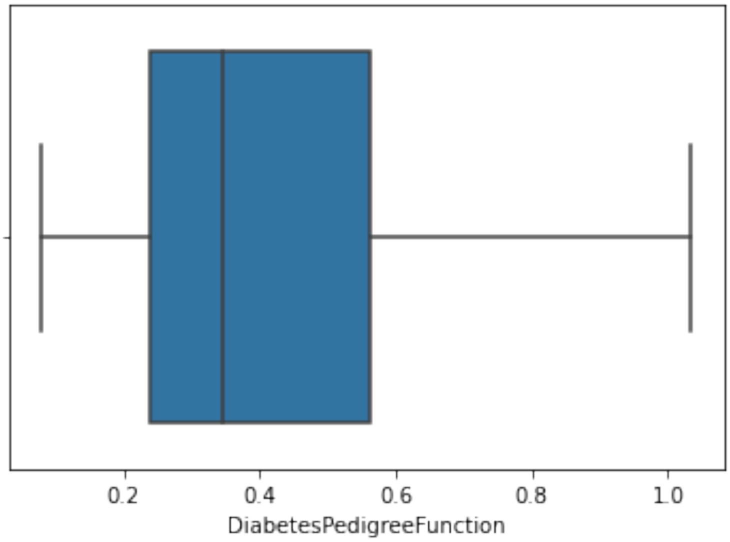
Out[83]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **131** | 9 | 122.0 | 56.0 | 20.323056 | 0 | 33.3 | |
| **267** | 2 | 128.0 | 64.0 | 42.000000 | 0 | 40.0 | |
| **270** | 10 | 101.0 | 86.0 | 37.000000 | 0 | 45.6 | |
| **314** | 7 | 109.0 | 80.0 | 31.000000 | 0 | 35.9 | |
| **416** | 1 | 97.0 | 68.0 | 21.000000 | 0 | 27.2 | |
| **434** | 1 | 90.0 | 68.0 | 8.000000 | 0 | 24.5 | |
| **493** | 4 | 125.0 | 70.0 | 18.000000 | 122 | 28.9 | |
| **588** | 3 | 176.0 | 86.0 | 27.000000 | 156 | 33.3 | |
| **657** | 1 | 120.0 | 80.0 | 48.000000 | 200 | 38.9 | |
| **747** | 1 | 81.0 | 74.0 | 41.000000 | 57 | 46.3 | |
| **755** | 1 | 128.0 | 88.0 | 39.000000 | 110 | 36.5 | |

In [84]:
```
dataset = dataset[(dataset.DiabetesPedigreeFunction > lower_limit_D
iabetesPedigreeFunction) & (dataset.DiabetesPedigreeFunction < uppe
r_limit_DiabetesPedigreeFunction)]
```

In [85]:
```
sb.boxplot(dataset.DiabetesPedigreeFunction)
```

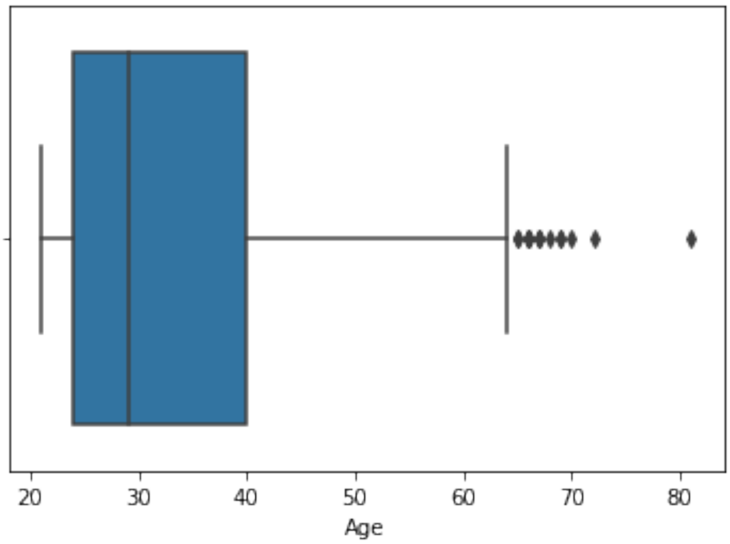
Out[85]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6ac64970>




In [86]:
```
# Última variável para analisar 'Age'
```

In [87]:
```
sb.boxplot(dataset.Age)
```

Out[87]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6ad206a0>



In [88]:
```
Q1_Age = dataset.Age.quantile(.25)
Q3_Age = dataset.Age.quantile(.75)

IQR_Age = Q3_Age - Q1_Age

lower_limit_Age = Q1_Age - 1.5 * IQR_Age
upper_limit_Age = Q3_Age + 1.5 * IQR_Age
```

In [89]:
```
dataset[(dataset.Age < lower_limit_Age) | (dataset.Age > upper_limi
t_Age)]
```

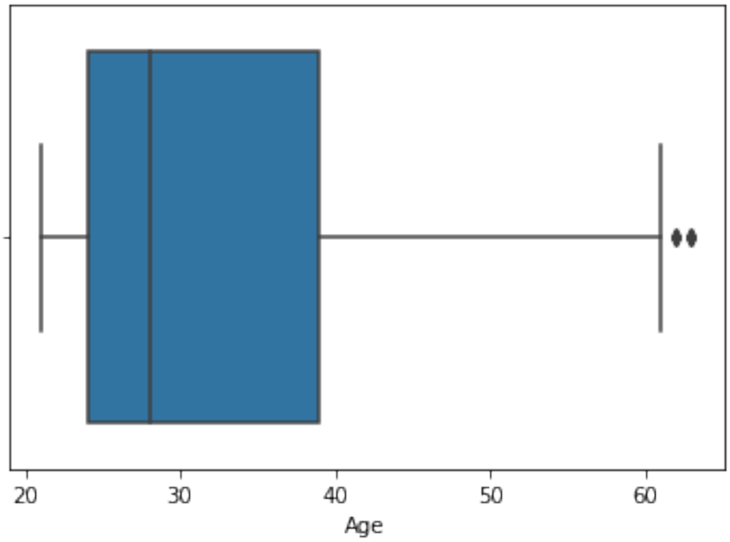
Out[89]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedi |
|---|---|---|---|---|---|---|---|
| **123** | 5 | 132.0 | 80.000000 | 20.323056 | 0 | 26.800000 | |
| **148** | 5 | 147.0 | 78.000000 | 20.323056 | 0 | 33.700000 | |
| **221** | 2 | 158.0 | 90.000000 | 20.323056 | 0 | 31.600000 | |
| **294** | 0 | 161.0 | 50.000000 | 20.323056 | 0 | 21.900000 | |
| **363** | 4 | 146.0 | 78.000000 | 20.323056 | 0 | 38.500000 | |
| **453** | 2 | 119.0 | 69.098168 | 20.323056 | 0 | 19.600000 | |
| **459** | 9 | 134.0 | 74.000000 | 33.000000 | 60 | 25.900000 | |
| **489** | 8 | 194.0 | 80.000000 | 20.323056 | 0 | 26.100000 | |
| **495** | 6 | 166.0 | 74.000000 | 20.323056 | 0 | 26.600000 | |
| **537** | 0 | 57.0 | 60.000000 | 20.323056 | 0 | 21.700000 | |
| **552** | 6 | 114.0 | 88.000000 | 20.323056 | 0 | 27.800000 | |
| **666** | 4 | 145.0 | 82.000000 | 18.000000 | 0 | 32.500000 | |
| **674** | 8 | 91.0 | 82.000000 | 20.323056 | 0 | 35.600000 | |
| **684** | 5 | 136.0 | 82.000000 | 20.323056 | 0 | 31.530246 | |
| **759** | 6 | 190.0 | 92.000000 | 20.323056 | 0 | 35.500000 | |

In [90]:
```
dataset = dataset[(dataset.Age > lower_limit_Age) & (dataset.Age <
upper_limit_Age)]
```

In [91]:
```
sb.boxplot(dataset.Age)
```

Out[91]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6ade2250>




In [92]:
```
# Ainda temos Outliers.Devemos alterar o valor do fator.
lower_limit_Age = Q1_Age - 1.25 * IQR_Age
upper_limit_Age = Q3_Age + 1.25 * IQR_Age
```

In [93]:
```
dataset[(dataset.Age < lower_limit_Age) | (dataset.Age > upper_limi
t_Age)]
```

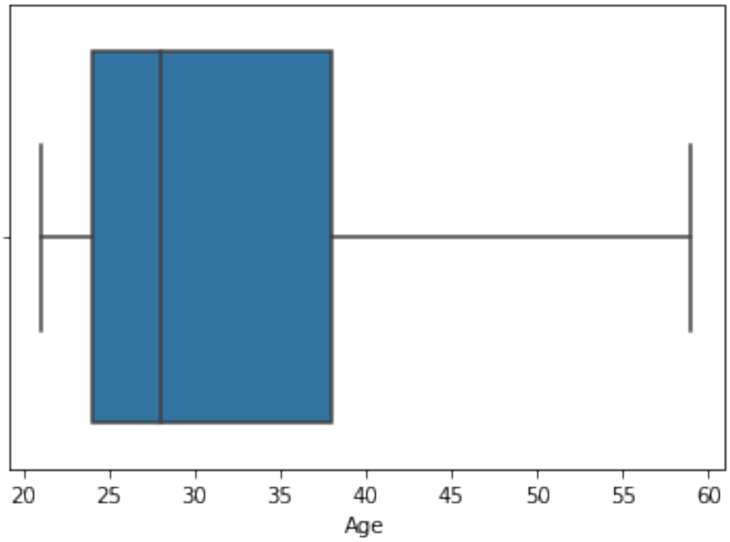
Out[93]:

| | Pregnancies | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigreeF |
|---|---|---|---|---|---|---|---|
| **115** | 4 | 146.0 | 92.0 | 20.323056 | 0 | 31.2 | |
| **129** | 0 | 105.0 | 84.0 | 20.323056 | 0 | 27.9 | |
| **223** | 7 | 142.0 | 60.0 | 33.000000 | 190 | 28.8 | |
| **263** | 3 | 142.0 | 80.0 | 15.000000 | 0 | 32.4 | |
| **361** | 5 | 158.0 | 70.0 | 20.323056 | 0 | 29.8 | |
| **456** | 1 | 135.0 | 54.0 | 20.323056 | 0 | 26.7 | |
| **479** | 4 | 132.0 | 86.0 | 31.000000 | 0 | 28.0 | |
| **582** | 12 | 121.0 | 78.0 | 17.000000 | 0 | 26.5 | |
| **763** | 10 | 101.0 | 76.0 | 48.000000 | 180 | 32.9 | |

In [94]:
```
dataset = dataset[(dataset.Age > lower_limit_Age) & (dataset.Age <
upper_limit_Age)]
```

In [95]:
```
sb.boxplot(dataset.Age)
```

Out[95]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6aea7610>




In [96]:
```
# Vamos verificar como ficaram as medidas estatísticas após a remoç
ão dos outliers e valores 0
dataset.describe()
```

Out[96]:

| | Glucose | BloodPressure | SkinThickness | Insulin | BMI | DiabetesPedigre |
|---|---|---|---|---|---|---|
| **count** | 613.000000 | 613.000000 | 613.000000 | 613.000000 | 613.000000 | 6 |
| **mean** | 116.621339 | 71.365807 | 25.835280 | 56.424144 | 31.683968 | |
| **std** | 27.747708 | 10.598459 | 8.881398 | 69.399805 | 6.175456 | |
| **min** | 44.000000 | 44.000000 | 8.000000 | 0.000000 | 18.200000 | |
| **25%** | 97.000000 | 64.000000 | 20.323056 | 0.000000 | 27.300000 | |
| **50%** | 112.000000 | 70.000000 | 22.000000 | 0.000000 | 31.600000 | |
| **75%** | 131.000000 | 78.000000 | 32.000000 | 105.000000 | 35.700000 | |
| **max** | 196.000000 | 102.000000 | 60.000000 | 250.000000 | 46.800000 | |

In [97]:
```
# Agora vamos verificar se as classes da variável target (Outcome)
estão balanceadas.
# Mas antes é necessário separar o dataset em variáveis preditoras
e variável target.

x = dataset.drop(columns='Outcome')
y = dataset.Outcome
```

In [98]:
```
x.shape , y.shape
```

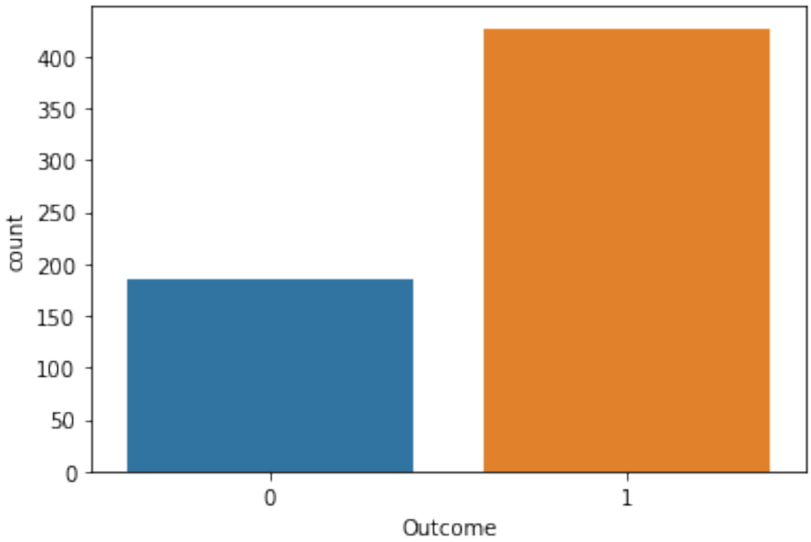
Out[98]: ((613, 8), (613,))

In [99]:
```
# Verificando se as classes da variável Outcome estão balanceadas
y.value_counts()
```

Out[99]:
```
0    427
1    186
Name: Outcome, dtype: int64
```

In [100]:
```
# Constatando o desbalanceamento das classes graficamente
sb.countplot(x=y, data=y)
```

Out[100]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6af6e580>




In [101]:
```
# Aplicando o algoritmo SMOTE para balancear as classes.

# Instanciando o SMOTE
smt = SMOTE()
```

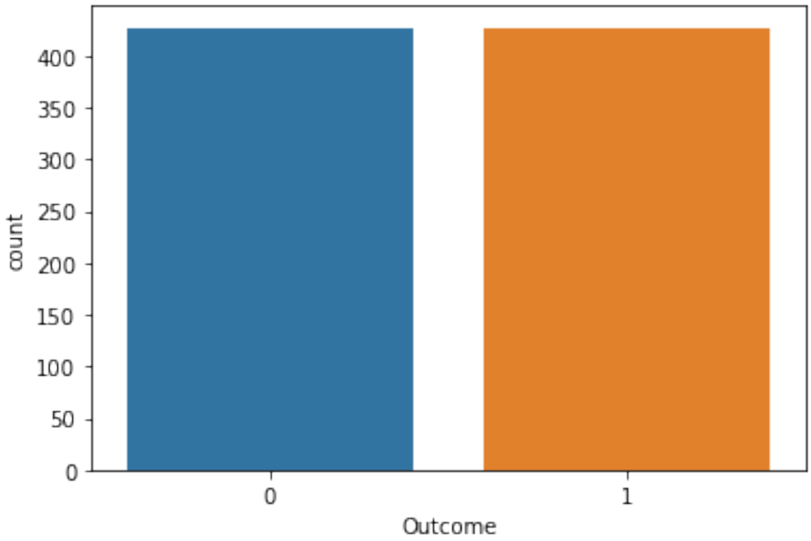
In [102]:
```
x,y = smt.fit_sample(x,y)
```

In [103]:
```
# Verificando o balanceamento após a execução do SMOTE
y.value_counts()
```

Out[103]:
```
1    427
0    427
Name: Outcome, dtype: int64
```

In [104]:
```
# Verificando graficamente
sb.countplot(x=y, data=y)
```

Out[104]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdf6af758b0>




In [105]:
```
# O último passo é normalizar as variáveis de medida. Variáveis com diferentes escalas podem comprometer o desempenho
# de um algoritmo de ML. Para isso utilizarei o StandardScaler() do pacote SciKit-Learn

# Instanciando o StandardScaler()
scaler = preprocessing.StandardScaler().fit(x)
```

In [106]:
```
x = scaler.transform(x)
```

In [107]:
```
# Dividindo os dados de Treino e dados de Teste para o modelo de ML
```

In [108]:
```
xTrain, xTest, yTrain, yTest = train_test_split(x,y)
```

In [109]:
```
xTrain.shape , xTest.shape
```

Out[109]: ((640, 8), (214, 8))

In [110]:
```
yTrain.shape , yTest.shape
```

Out[110]: ((640,), (214,))

In [111]:
```
# Vamos agora começar a construir o modelo de ML
# Primeiramente vamos construir modelos de base utilizando diversos algoritmos de ML
```

In [112]:
```
####################################### Random Forest ######################################
```

In [113]:
```
# Instânciando o classificador
clf_rf = RandomForestClassifier()
```

In [114]:
```
# Treinando o modelo base
modelo_rf = clf_rf.fit(xTrain,yTrain)
```

In [115]:
```
# Fazendo previsões com o modelo treinado
pred_rf = modelo_rf.predict(xTest)
```

In [116]:
```
# Verificando a acurácia do modelo
rf_accuracy = accuracy_score(yTest, pred_rf)
```

In [117]:
```
# Relatório de classificação
print(classification_report(yTest, pred_rf))
```

```
              precision    recall  f1-score   support

           0       0.85      0.82      0.83       117
           1       0.79      0.82      0.81        97

    accuracy                           0.82       214
   macro avg       0.82      0.82      0.82       214
weighted avg       0.82      0.82      0.82       214
```

In [118]:
```
# Matriz de cofusão
print(pd.crosstab(yTest, pred_rf, rownames=['Real'], colnames=['Pre
dito'], margins=True))
```

```
Predito    0    1  All
Real
0         96   21  117
1         17   80   97
All      113  101  214
```

In [119]:
```
######################################## SVM ######################
#################
```

In [120]:
```
# Instanciando o classificador
clf_svm = svm.SVC()
```

In [121]:
```
# Treinando o modelo base
modelo_svm = clf_svm.fit(xTrain, yTrain)
```

In [122]:
```
# Fazendo previsões com o modelo treinado
pred_svm = modelo_svm.predict(xTest)
```

In [123]:
```
# Verificando a acurácia do modelo
svm_accuracy = accuracy_score(yTest, pred_svm)
```

In [124]:
```
# Relatório de classificação
print(classification_report(yTest, pred_svm))
```

```
              precision    recall  f1-score   support

           0       0.83      0.79      0.81       117
           1       0.76      0.80      0.78        97

    accuracy                           0.80       214
   macro avg       0.80      0.80      0.80       214
weighted avg       0.80      0.80      0.80       214
```

In [125]:
```
# Matriz de cofusão
print(pd.crosstab(yTest, pred_svm, rownames=['Real'], colnames=['Pr
edito'], margins=True))
```

```
Predito    0    1  All
Real
0         93   24  117
1         19   78   97
All      112  102  214
```

In [126]:
```
######################################## KNN ######################
##################
```

In [127]:
```
# Instanciando o classificador
clf_knn = KNeighborsClassifier()
```

In [128]:
```
# Treinando o modelo base
modelo_knn = clf_knn.fit(xTrain, yTrain)
```

In [129]:
```
# Fazendo previsões com o modelo treinado
pred_knn = modelo_knn.predict(xTest)
```

In [130]:
```
# Verificando a acurácia do modelo
knn_accuracy = accuracy_score(yTest, pred_knn)
```

In [131]:
```
# Relatório de classificação
print(classification_report(yTest, pred_knn))
```

```
              precision    recall  f1-score   support

           0       0.88      0.72      0.79       117
           1       0.72      0.88      0.79        97

    accuracy                           0.79       214
   macro avg       0.80      0.80      0.79       214
weighted avg       0.80      0.79      0.79       214
```

In [132]:
```
# Matriz de cofusão
print(pd.crosstab(yTest, pred_knn, rownames=['Real'], colnames=['Pr
edito'], margins=True))
```

```
Predito   0    1  All
Real
0        84   33  117
1        12   85   97
All      96  118  214
```

In [133]:
```
######################################## Logistic Regression ######
##################################
```

In [134]:
```
# Instanciando o classificador
clf_lr = LogisticRegression()
```

In [135]:
```
# Treinando o modelo base
modelo_lr = clf_lr.fit(xTrain, yTrain)
```

In [136]:
```
# Fazendo previsões com o modelo treinado
pred_lr = modelo_lr.predict(xTest)
```

In [137]:
```
# Verificando a acurácia do modelo
lr_accuracy = accuracy_score(yTest, pred_lr)
```

In [138]:
```
# Relatório de classificação
print(classification_report(yTest, pred_lr))
```

```
              precision    recall  f1-score   support

           0       0.81      0.75      0.78       117
           1       0.73      0.79      0.76        97

    accuracy                           0.77       214
   macro avg       0.77      0.77      0.77       214
weighted avg       0.77      0.77      0.77       214
```

In [139]:
```
# Matriz de cofusão
print(pd.crosstab(yTest, pred_lr, rownames=['Real'], colnames=['Pre
dito'], margins=True))
```

```
Predito    0    1  All
Real
0         88   29  117
1         20   77   97
All      108  106  214
```

In [172]:
```
# Criando uma tabela com os algoritmos utilizados e os respectivos
scores.
obj = {'Model':['RF','SVM', 'KNN', 'LR'], 'Score':[round(rf_accurac
y * 100),
                                                   round(svm_accura
cy * 100),
                                                   round(knn_accura
cy * 100),
                                                   round(lr_accurac
y * 100)]}
```

In [173]:
```
scores = pd.DataFrame(data=obj)
```

In [174]:
```
scores.head()
```

Out[174]:

| | Model | Score |
|---|---|---|
| **0** | RF | 82.0 |
| **1** | SVM | 80.0 |
| **2** | KNN | 79.0 |
| **3** | LR | 77.0 |

In [175]:
```
# Podemos observar acima que o algoritmo BASE Random Forest apresentou uma maior accurácia bem como um menor índice
# de erros (comparando as matrizes de confusão). Portanto, utilizarei o RF para criar o modelo definitivo.
# Primeiramente vou realizar testes com os hiperparâmetros do RF através do grid e verificar se ocorre alguma
# melhora na acurácia do modelo.
```

In [184]:
```
# Criando as listas de valores dos hiperparâmetros
valores = [1,2,3,4,5,6,7,8,9,10]
criterion = ['gini', 'entropy']

grid_params_rf = [{'criterion' : criterion,
                   'min_samples_leaf': valores,
                   'max_depth': valores,
                   'min_samples_split': valores}]
```

In [185]:
```
grid_rf = GridSearchCV(estimator=clf_rf,
                       param_grid=grid_params_rf,
                       scoring='accuracy',
                       cv=10,)
```

In [ ]:
```
grid_rf.fit(xTrain, yTrain)
```

In [187]:
```
grid_rf.best_params_
```

Out[187]:
```
{'criterion': 'entropy',
 'max_depth': 10,
 'min_samples_leaf': 1,
 'min_samples_split': 3}
```

In [188]:
```
grid_rf.best_score_
```

Out[188]:
```
0.8484375
```

In [189]:
```
# Observe que melhoramos nosso modelo através dos testes com os hiperparâmetros do algoritmo Random Forest.
# Agora vamos criar o modelo definitivo.

# Instanciando o classificador com os hiperparâmetros
classifier = RandomForestClassifier(criterion='entropy',max_depth=10, min_samples_leaf=1,min_samples_split=3)
```

In [210]:
```
# Treinando o algoritmo com o novo classificador
modelo = classifier.fit(x, y)
```

In [211]:
```
# Fazendo predições
predict = modelo.predict(xTest)
```

In [212]:
```
# Verificando a acurácia do modelo
accuracy = accuracy_score(yTest, predict)
```

In [213]:
```
accuracy
```

Out[213]:
```
0.9579439252336449
```

In [214]:
```
# Relatório de classificação
print(classification_report(yTest, predict))
```

```
              precision    recall  f1-score   support

           0       0.99      0.93      0.96       117
           1       0.92      0.99      0.96        97

    accuracy                           0.96       214
   macro avg       0.96      0.96      0.96       214
weighted avg       0.96      0.96      0.96       214
```

In [215]:
```
# Matriz de cofusão
print(pd.crosstab(yTest, predict, rownames=['Real'], colnames=['Pre
dito'], margins=True))
```

```
Predito    0    1  All
Real
0        109    8  117
1          1   96   97
All      110  104  214
```

In [216]:
```
# O modelo melhorou quando eu o treinei com todos os dados do datas
et. Quanto mais dados o dataset tiver,
# mais amostras ele terá e consequentemente sua accurácia tente a s
ubir pois ele aprenderá mais
# Com bases muito pequenas, dificilmente um modelo ficará acima da
faixa dos 80 - 90%.
```

In [221]:
```
# Vamos salvar esse modelo treinado para o disco com o joblib
joblib.dump(modelo,'model_rf.joblib')
```

Out[221]:
```
['model_rf.joblib']
```