project-1

March 7, 2024

[98]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression, LogisticRegression
import matplotlib.style as style
from sklearn.model_selection import train_test_split
data = pd.read_csv(r'D:/ANACONDA/diabetes.csv')
data.head()
```

[98]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

[99]:
```
data.shape
```

[99]:
```
(768, 9)
```

[100]:
```
data.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
```

[101]:
```
data.describe()
```

[101]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin
count	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479
std	3.369578	31.972618	19.355807	15.952218	115.244002
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000
75%	6.000000	140.250000	80.000000	32.000000	127.250000
max	17.000000	199.000000	122.000000	99.000000	846.000000

	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000
mean	31.992578	0.471876	33.240885	0.348958
std	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.078000	21.000000	0.000000
25%	27.300000	0.243750	24.000000	0.000000
50%	32.000000	0.372500	29.000000	0.000000
75%	36.600000	0.626250	41.000000	1.000000
max	67.100000	2.420000	81.000000	1.000000

[102]:
```
data.isnull()
```

[102]:

	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

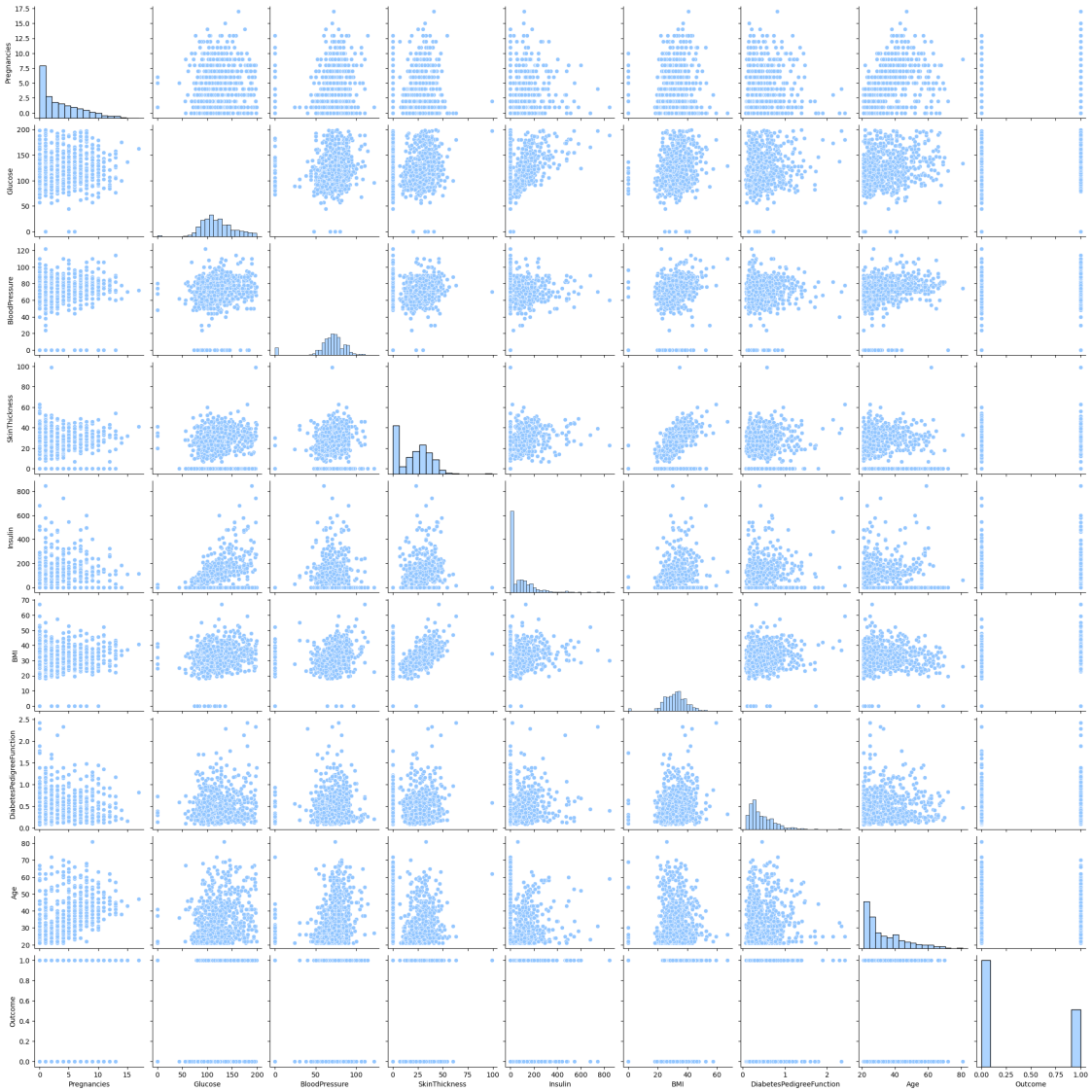
	DiabetesPedigreeFunction	Age	Outcome

```
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]
```

[103]:
```
plt.figure(figsize=(10,6))
sns.pairplot(data)
```

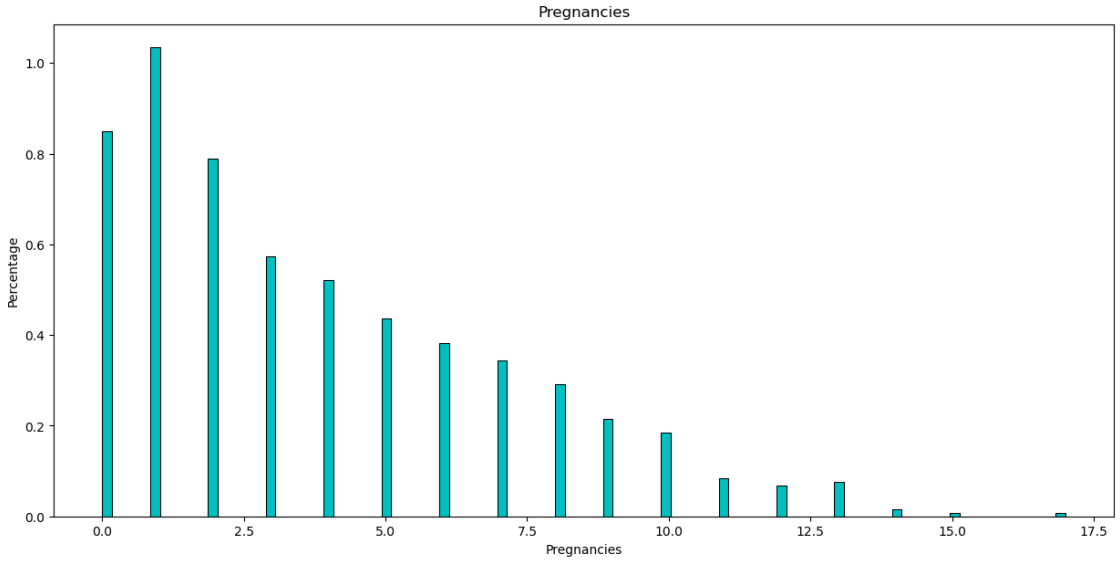
```
D:\ANACONDA\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure
layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

[103]:
```
<seaborn.axisgrid.PairGrid at 0x1a394ee9dd0>

<Figure size 1000x600 with 0 Axes>
```



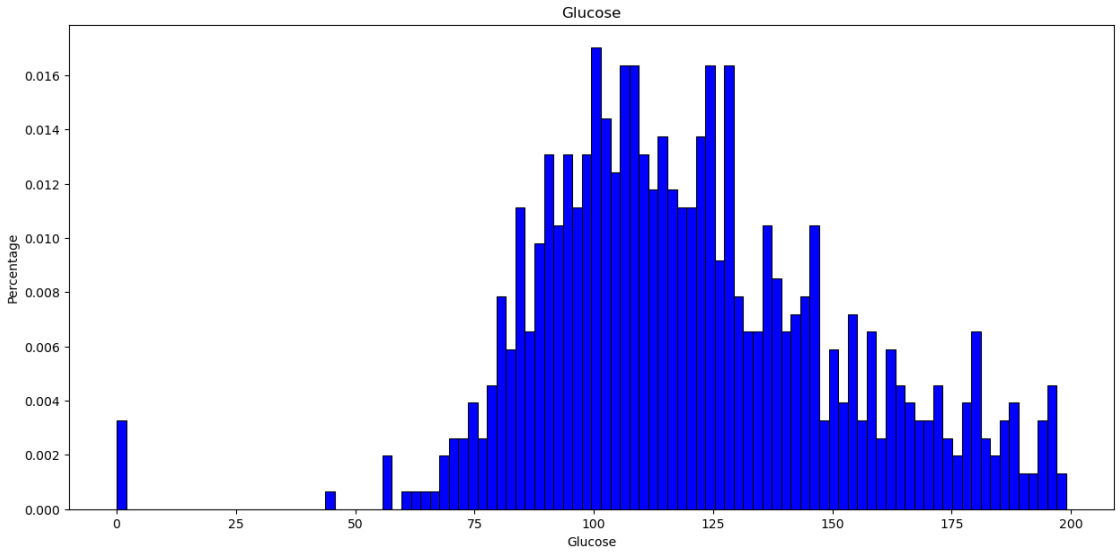
[104]:
```
#EDA
# Pregnancies
plt.figure(figsize=(15,7))
sns.histplot(data["Pregnancies"], facecolor='c', bins=100, stat="density");
plt.ylabel("Percentage")
plt.title("Pregnancies")
```

[104]: Text(0.5, 1.0, 'Pregnancies')



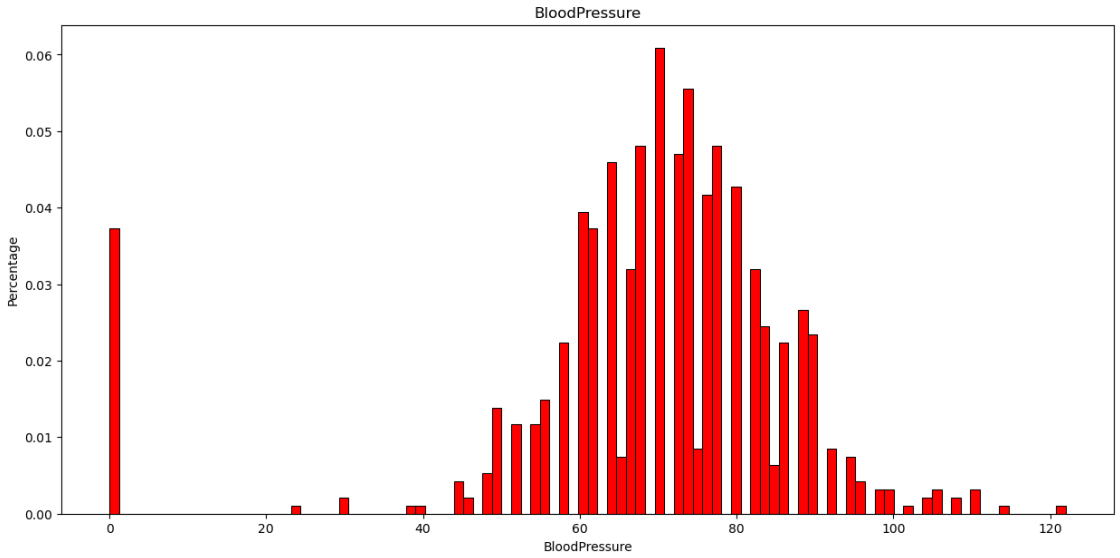
[105]:
```
# Glucose
plt.figure(figsize=(15,7))
sns.histplot(data["Glucose"], facecolor='blue', bins=100, stat="density");
plt.ylabel("Percentage")
plt.title("Glucose")
```

[105]: Text(0.5, 1.0, 'Glucose')



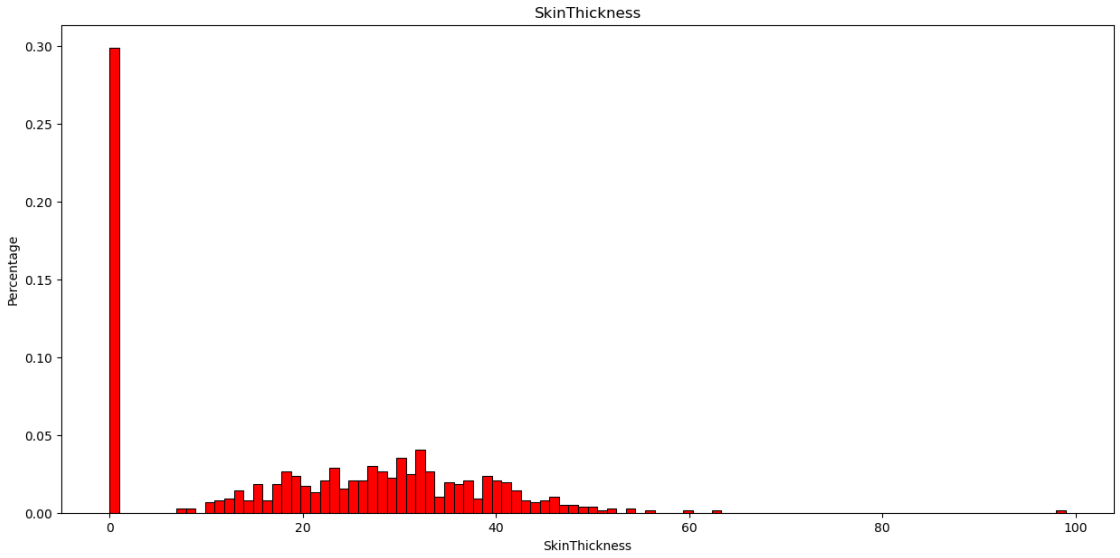
[106]:
```
# BloodPressure
plt.figure(figsize=(15,7))
sns.histplot(data["BloodPressure"], facecolor='red', bins=100, stat="density");
plt.ylabel("Percentage")
plt.title("BloodPressure")
```

[106]: Text(0.5, 1.0, 'BloodPressure')



[107]:
```
# SkinThickness
plt.figure(figsize=(15,7))
sns.histplot(data["SkinThickness"], facecolor='red', bins=100, stat="density");
plt.ylabel("Percentage")
plt.title("SkinThickness")
```

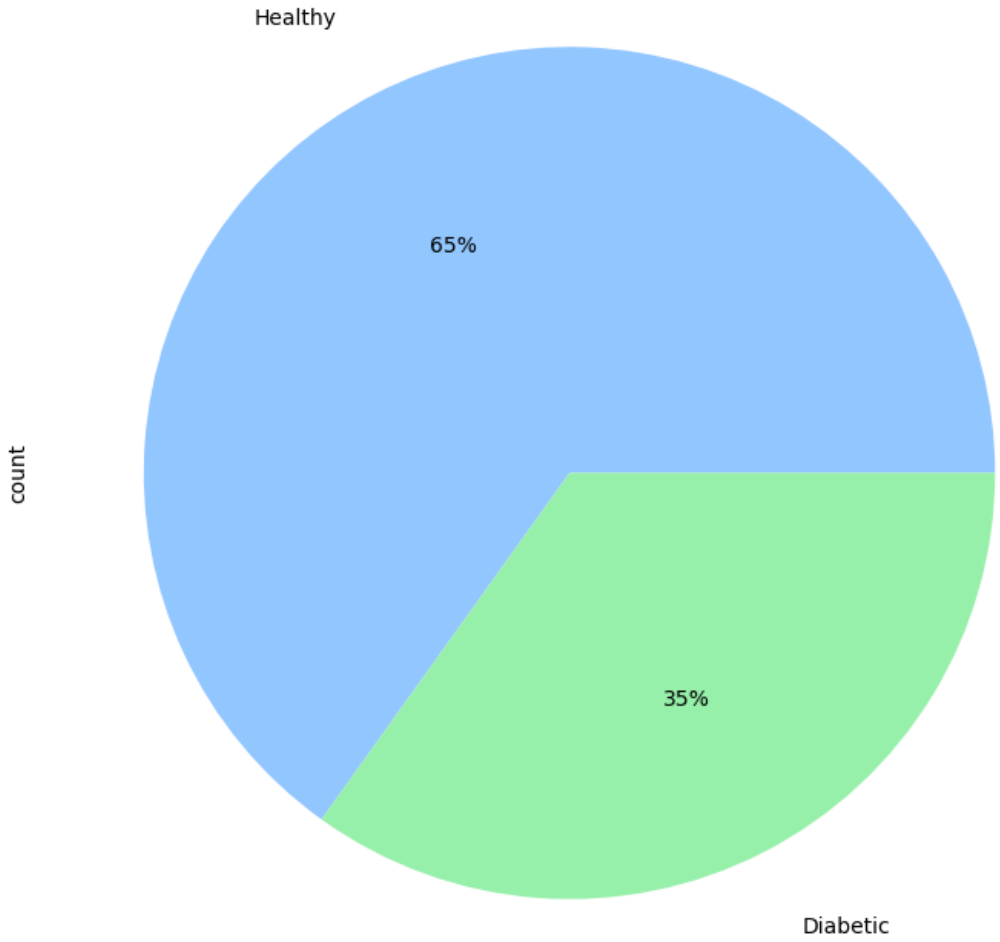
[107]: Text(0.5, 1.0, 'SkinThickness')



[108]:
```
style.available

style.use('seaborn-pastel')
labels = ["Healthy", "Diabetic"]
data['Outcome'].value_counts().plot(kind='pie',labels=labels,␣
 ↪subplots=True,autopct='%1.0f%%', labeldistance=1.2,figsize=(9,9))
```

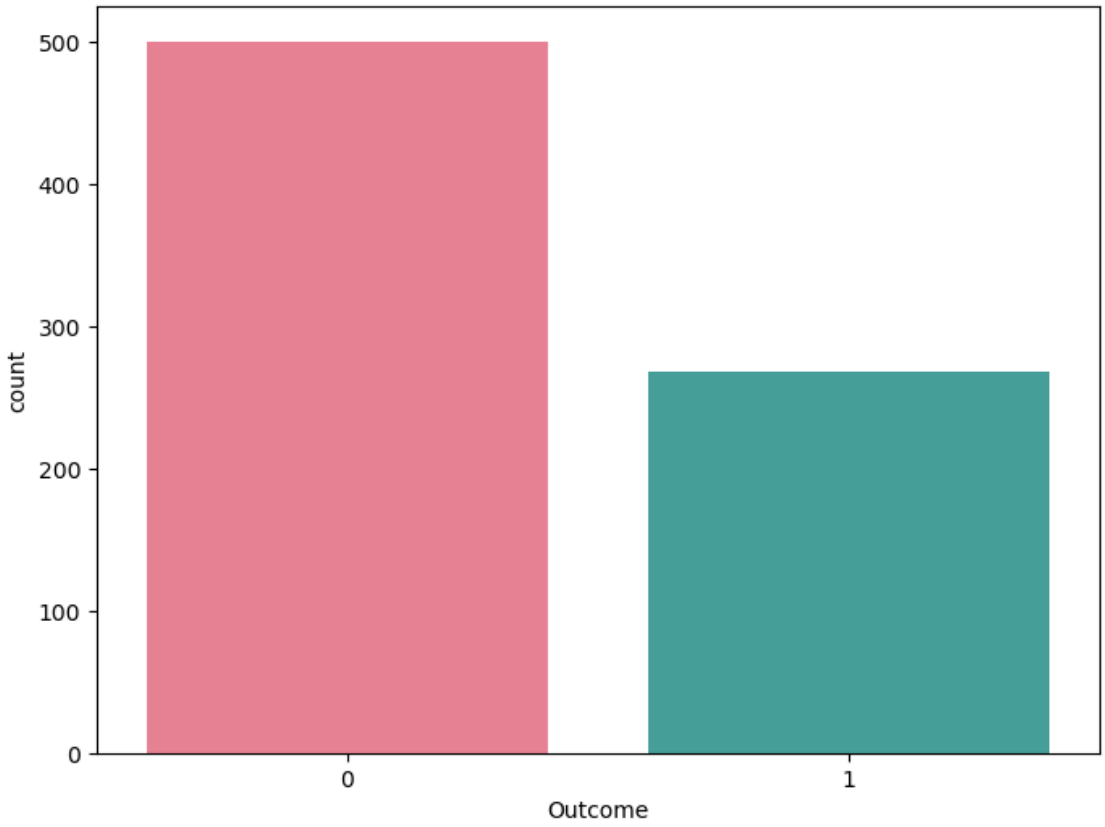
```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_6828\3929475367.py:3:
MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are
deprecated since 3.6, as they no longer correspond to the styles shipped by
seaborn. However, they will remain available as 'seaborn-v0_8-<style>'.
Alternatively, directly use the seaborn API instead.
  style.use('seaborn-pastel')
```

[108]:
```
array([<Axes: ylabel='count'>], dtype=object)
```



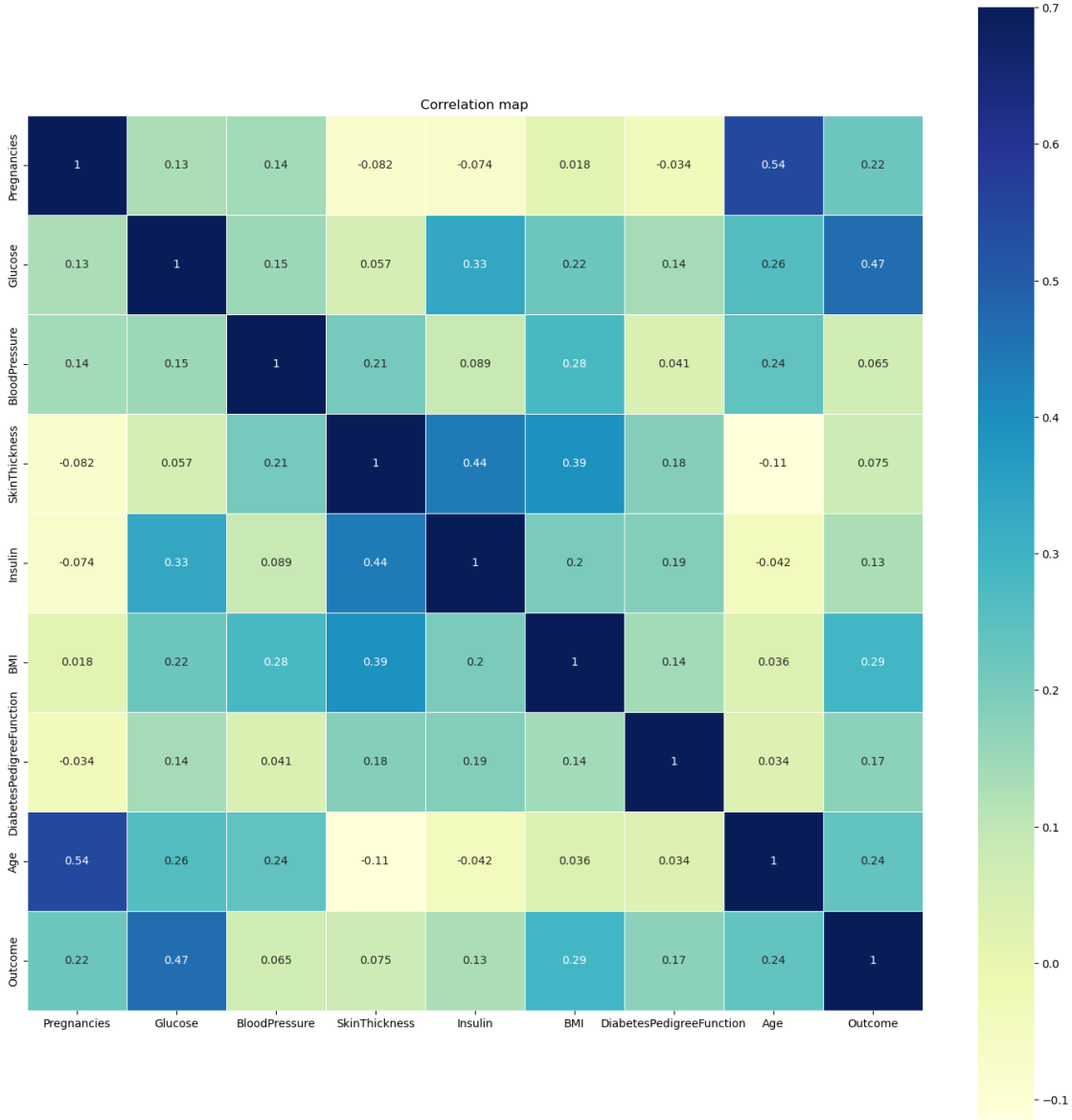
[109]:
```
from matplotlib.pyplot import figure, show

figure(figsize=(8,6))
ax = sns.countplot(x=data['Outcome'], data=data, palette="husl")
healthy, diabetics = data['Outcome'].value_counts().values
print("Sample of diabetic people: ",diabetics)
print("Sample of healthy people: ",healthy)
```

```
Sample of diabetic people:  268
Sample of healthy people:  500
```



[110]:
```
matrix = data.corr()
f, ax = plt.subplots(figsize=(18, 18))
sns.heatmap(matrix, vmax=.7, square=True, cmap="YlGnBu", annot=True,␣
 ↪linewidths=.5).set_title('Correlation map')
```

[110]: Text(0.5, 1.0, 'Correlation map')



[111]:
```
X = data.drop('Outcome', axis=1)
y = data['Outcome']
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[112]:
```
# Create Logistic Regression classifier object
log_model = LogisticRegression()
log_model.fit(X_train, y_train)
y_pred_3 = log_model.predict(X_test)
logistic = accuracy_score(y_pred_3, y_test)
```

```
logistic
```

```
D:\ANACONDA\Lib\site-packages\sklearn\linear_model\_logistic.py:460:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[112]: 0.7467532467532467

[ ]:

[ ]:

[ ]:

[ ]: