```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from sklearn.metrics import confusion_matrix

df = pd.read_csv("/content/Diabetes.csv")

df.describe()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"Number of times pregnant\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
269.85223453356366,\n        \"min\": 0.0,\n        \"max\": 768.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
3.8450520833333335,\n          3.0,\n          768.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Plasma glucose concentration a 2
hours in an oral glucose tolerance test\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 243.73802348295857,\n
\"min\": 0.0,\n        \"max\": 768.0,\n        \"num_unique_values\":
8,\n        \"samples\": [\n          120.89453125,\n          117.0,\
n          768.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Diastolic blood pressure (mm Hg)\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 252.8525053581062,\n
\"min\": 0.0,\n        \"max\": 768.0,\n        \"num_unique_values\":
8,\n        \"samples\": [\n          69.10546875,\n          72.0,\n
768.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Triceps skin fold thickness (mm)\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 263.7684730531098,\n
\"min\": 0.0,\n        \"max\": 768.0,\n        \"num_unique_values\":
7,\n        \"samples\": [\n          768.0,\n
20.536458333333332,\n          32.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"2-Hour serum insulin (mu U/ml)\",\
n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 350.26059167945886,\n        \"min\": 0.0,\n        \"max\":
846.0,\n        \"num_unique_values\": 7,\n        \"samples\": [\n
768.0,\n          79.79947916666667,\n          127.25\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Body mass index (weight in
kg/(height in m)^2)\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 262.05117817552093,\n        \"min\":
0.0,\n        \"max\": 768.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          31.992578124999998,\n          32.0,\n
```

```
768.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Diabetes pedigree function\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 271.3005221658502,\n
\"min\": 0.078,\n        \"max\": 768.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
0.47187630208333325,\n          0.3725,\n          768.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Age (years)\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
260.1941178528413,\n        \"min\": 11.760231540678685,\n
\"max\": 768.0,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          33.240885416666664,\n          29.0,\n
768.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Class variable (0 or 1)\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 271.3865920388932,\n
\"min\": 0.0,\n        \"max\": 768.0,\n        \"num_unique_values\":
5,\n        \"samples\": [\n          0.3489583333333333,\n
1.0,\n          0.47695137724279896\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe"}
```


```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column
Non-Null Count  Dtype
---  ------
--------------  -----
 0   Number of times pregnant
768 non-null    int64
 1   Plasma glucose concentration a 2 hours in an oral glucose
tolerance test  768 non-null    int64
 2   Diastolic blood pressure (mm Hg)
768 non-null    int64
 3   Triceps skin fold thickness (mm)
768 non-null    int64
 4   2-Hour serum insulin (mu U/ml)
768 non-null    int64
 5   Body mass index (weight in kg/(height in m)^2)
768 non-null    float64
 6   Diabetes pedigree function
768 non-null    float64
 7   Age (years)
768 non-null    int64
 8   Class variable (0 or 1)
768 non-null    int64
```

```
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

```
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 768,\n  \"fields\": [\
n    {\n      \"column\": \"Number of times pregnant\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
3,\n        \"min\": 0,\n        \"max\": 17,\n
\"num_unique_values\": 17,\n        \"samples\": [\n          6,\n
1,\n          3\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Plasma glucose concentration a 2 hours in an oral glucose tolerance
test\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 31,\n        \"min\": 0,\n        \"max\": 199,\n
\"num_unique_values\": 136,\n        \"samples\": [\n          151,\n
101,\n          112\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Diastolic blood pressure (mm Hg)\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 19,\n        \"min\": 0,\n
\"max\": 122,\n        \"num_unique_values\": 47,\n
\"samples\": [\n          86,\n          46,\n          85\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Triceps skin fold thickness (mm)\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 15,\n        \"min\": 0,\n
\"max\": 99,\n        \"num_unique_values\": 51,\n        \"samples\":
[\n          7,\n          12,\n          48\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"2-Hour serum insulin (mu U/ml)\",\
n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 115,\n        \"min\": 0,\n        \"max\": 846,\n
\"num_unique_values\": 186,\n        \"samples\": [\n          52,\n
41,\n          183\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Body mass index (weight in kg/(height in m)^2)\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
7.884160320375446,\n        \"min\": 0.0,\n        \"max\": 67.1,\n
\"num_unique_values\": 248,\n        \"samples\": [\n          19.9,\n
31.0,\n          38.1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Diabetes pedigree function\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.3313285950127749,\n
\"min\": 0.078,\n        \"max\": 2.42,\n
\"num_unique_values\": 517,\n        \"samples\": [\n          1.731,\
n          0.426,\n          0.138\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Age (years)\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
11,\n        \"min\": 21,\n        \"max\": 81,\n
```

\"num_unique_values\": 52,\n   \"samples\": [\n   60,\n
47,\n   72\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Class variable (0 or 1)\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 0,\n
\"max\": 1,\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   0,\n   1\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"df"}

```
df.tail()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 5,\n \"fields\": [\n
{\n   \"column\": \"Number of times pregnant\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
3,\n   \"min\": 1,\n   \"max\": 10,\n
\"num_unique_values\": 4,\n   \"samples\": [\n   2,\n
1,\n   10\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Plasma glucose concentration a 2 hours in an oral glucose tolerance
test\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 14,\n   \"min\": 93,\n   \"max\": 126,\n
\"num_unique_values\": 5,\n   \"samples\": [\n   122,\n
93,\n   121\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Diastolic blood pressure (mm Hg)\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 5,\n   \"min\": 60,\n
\"max\": 76,\n   \"num_unique_values\": 4,\n   \"samples\":
[\n   70,\n   60,\n   76\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Triceps skin fold thickness
(mm)\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 17,\n   \"min\": 0,\n   \"max\": 48,\n
\"num_unique_values\": 5,\n   \"samples\": [\n   27,\n
31,\n   23\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"2-
Hour serum insulin (mu U/ml)\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 83,\n   \"min\": 0,\n
\"max\": 180,\n   \"num_unique_values\": 3,\n   \"samples\":
[\n   180,\n   0,\n   112\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Body mass index (weight in
kg/(height in m)^2)\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 3.9073008586491,\n   \"min\":
26.2,\n   \"max\": 36.8,\n   \"num_unique_values\": 5,\n
\"samples\": [\n   36.8,\n   30.4,\n   26.2\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Diabetes pedigree function\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0.07518643494673756,\n   \"min\": 0.171,\n   \"max\":

```
0.349,\n        \"num_unique_values\": 5,\n        \"samples\": [\n
0.34,\n          0.315,\n          0.245\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Age (years)\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
16,\n        \"min\": 23,\n        \"max\": 63,\n
\"num_unique_values\": 5,\n        \"samples\": [\n          27,\n
23,\n          30\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Class variable (0 or 1)\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0,\n        \"min\": 0,\n
\"max\": 1,\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          1,\n          0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe"}
```


```
num_instances, num_features = df.shape
print(f"\nNumber of instances: {num_instances}")
print(f"Number of features: {num_features}")


print(df.dtypes)
```

```
Number of instances: 768
Number of features: 9
Number of times pregnant
int64
Plasma glucose concentration a 2 hours in an oral glucose tolerance
test      int64
Diastolic blood pressure (mm Hg)
int64
Triceps skin fold thickness (mm)
int64
2-Hour serum insulin (mu U/ml)
int64
Body mass index (weight in kg/(height in m)^2)
float64
Diabetes pedigree function
float64
Age (years)
int64
Class variable (0 or 1)
int64
dtype: object
```

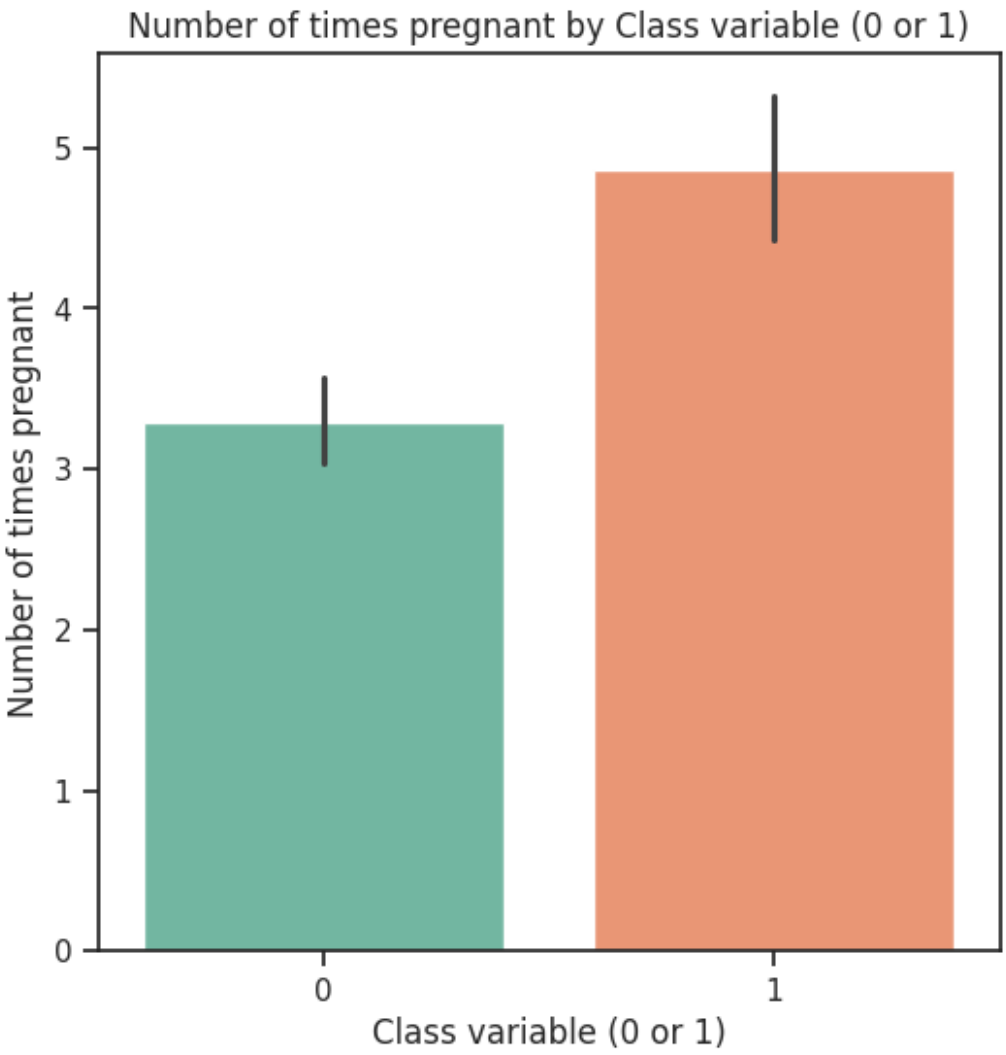
```
df['Class variable (0 or 1)'].value_counts()
```

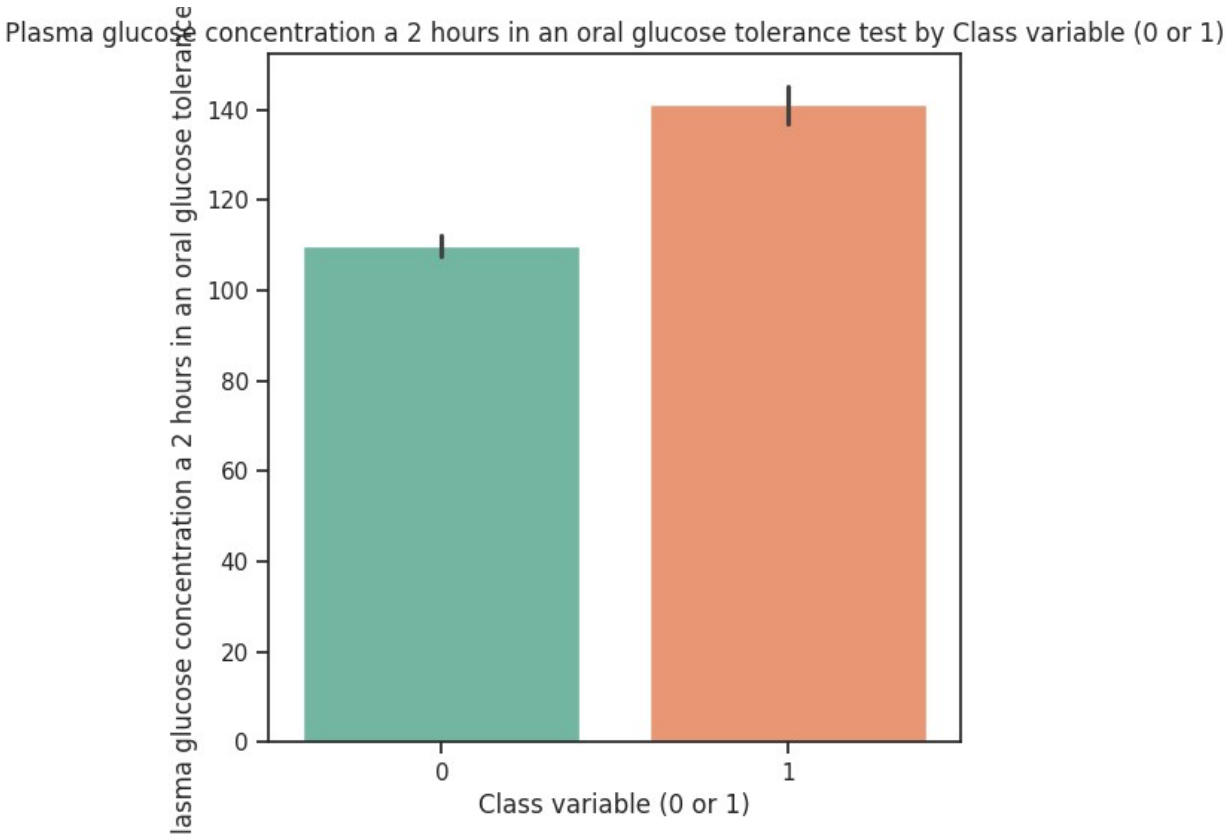
```
0    500
1    268
Name: Class variable (0 or 1), dtype: int64
```

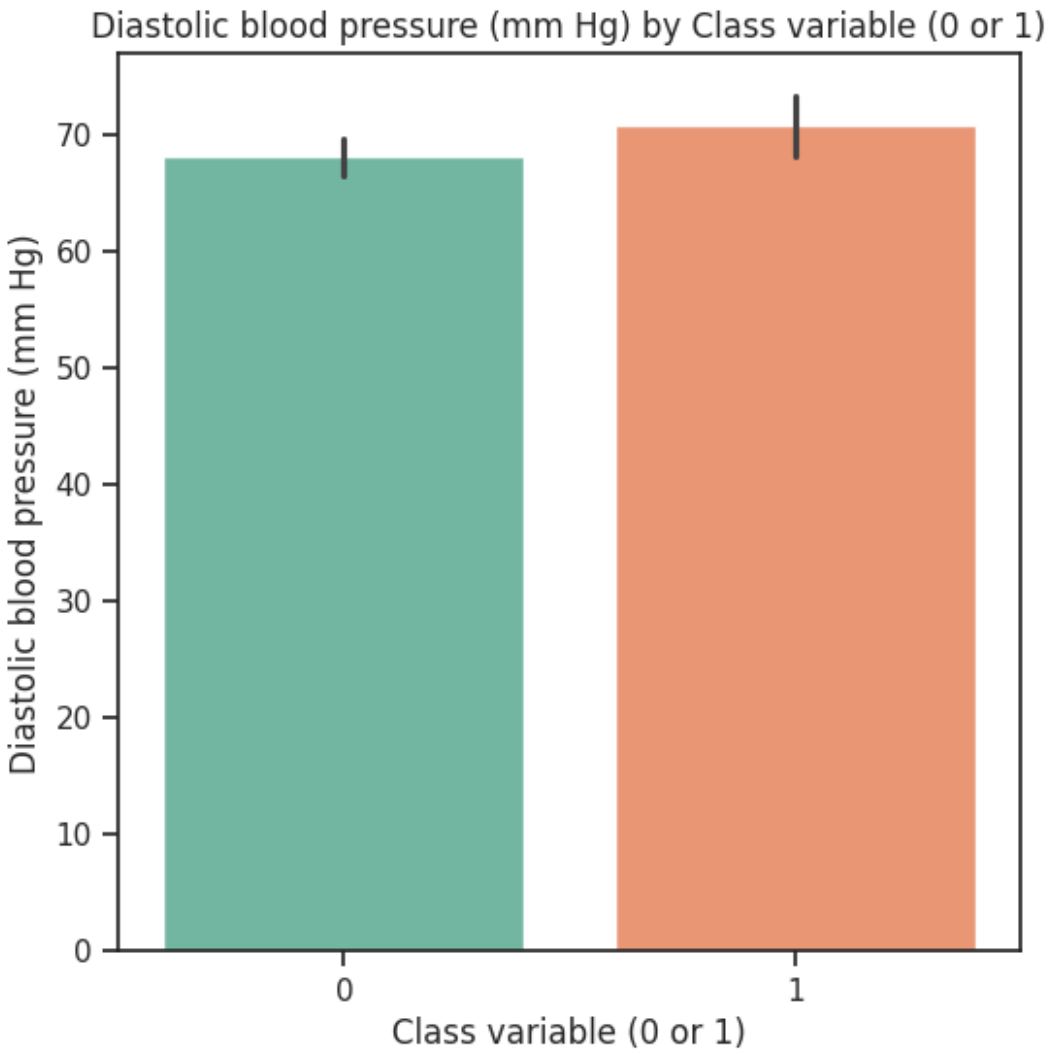
```
X = df.drop('Class variable (0 or 1)', axis=1)
y = df['Class variable (0 or 1)']
```

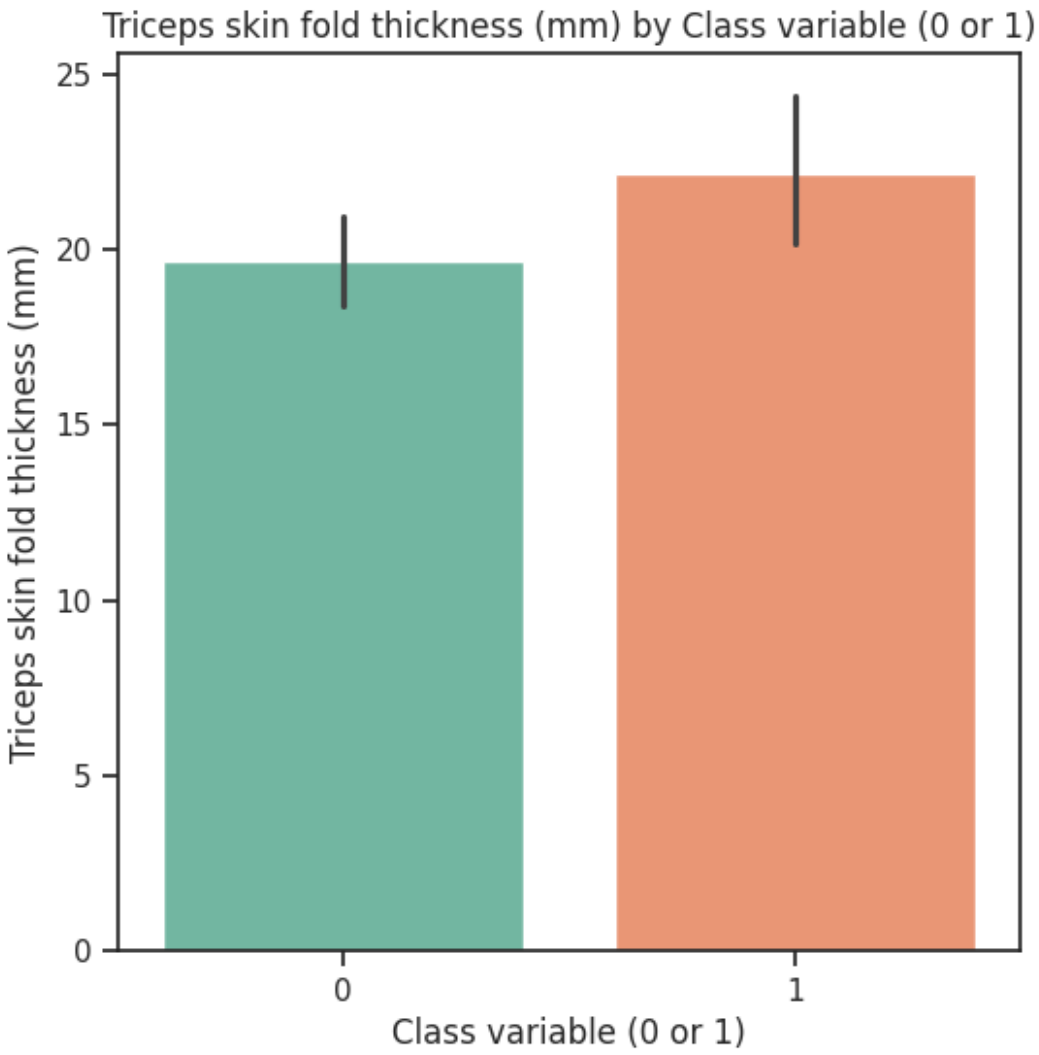
```
0      1
1      0
2      1
3      0
4      1
      ..
763    0
764    0
765    0
766    1
767    0
Name: Class variable (0 or 1), Length: 768, dtype: int64
```

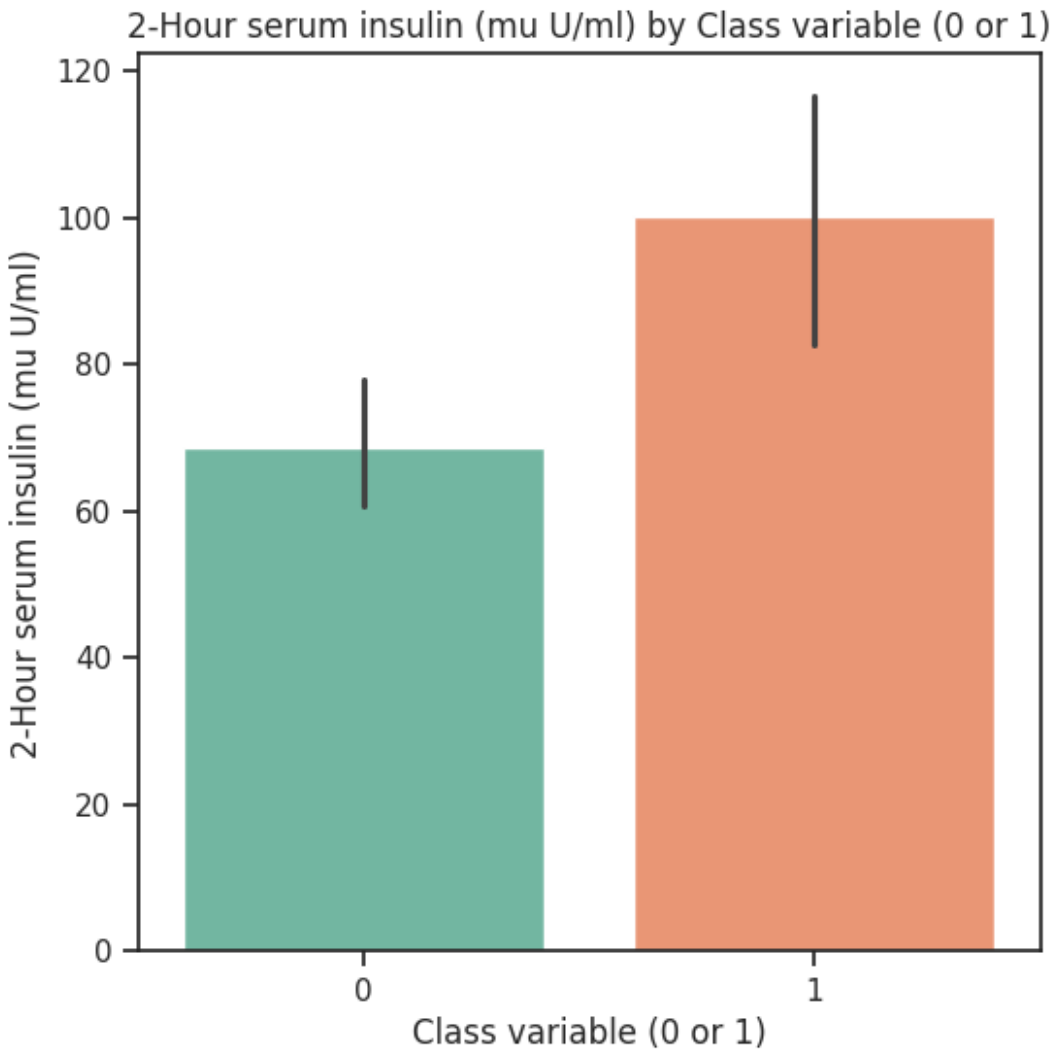
```
import seaborn as sns
import matplotlib.pyplot as plt

features = ['Number of times pregnant', 'Plasma glucose concentration
a 2 hours in an oral glucose tolerance test', 'Diastolic blood
pressure (mm Hg)', 'Triceps skin fold thickness (mm)', '2-Hour serum
insulin (mu U/ml)', 'Body mass index (weight in kg/(height in m)^2)',
'Diabetes pedigree function', 'Age (years)']
label = 'Class variable (0 or 1)'

for col in features:
    plt.figure(figsize=(6, 6))
    sns.barplot(x=label, y=col, hue=label, data=df, palette='Set2',
legend=False)
    plt.title(f'{col} by {label}')
    plt.show()
```

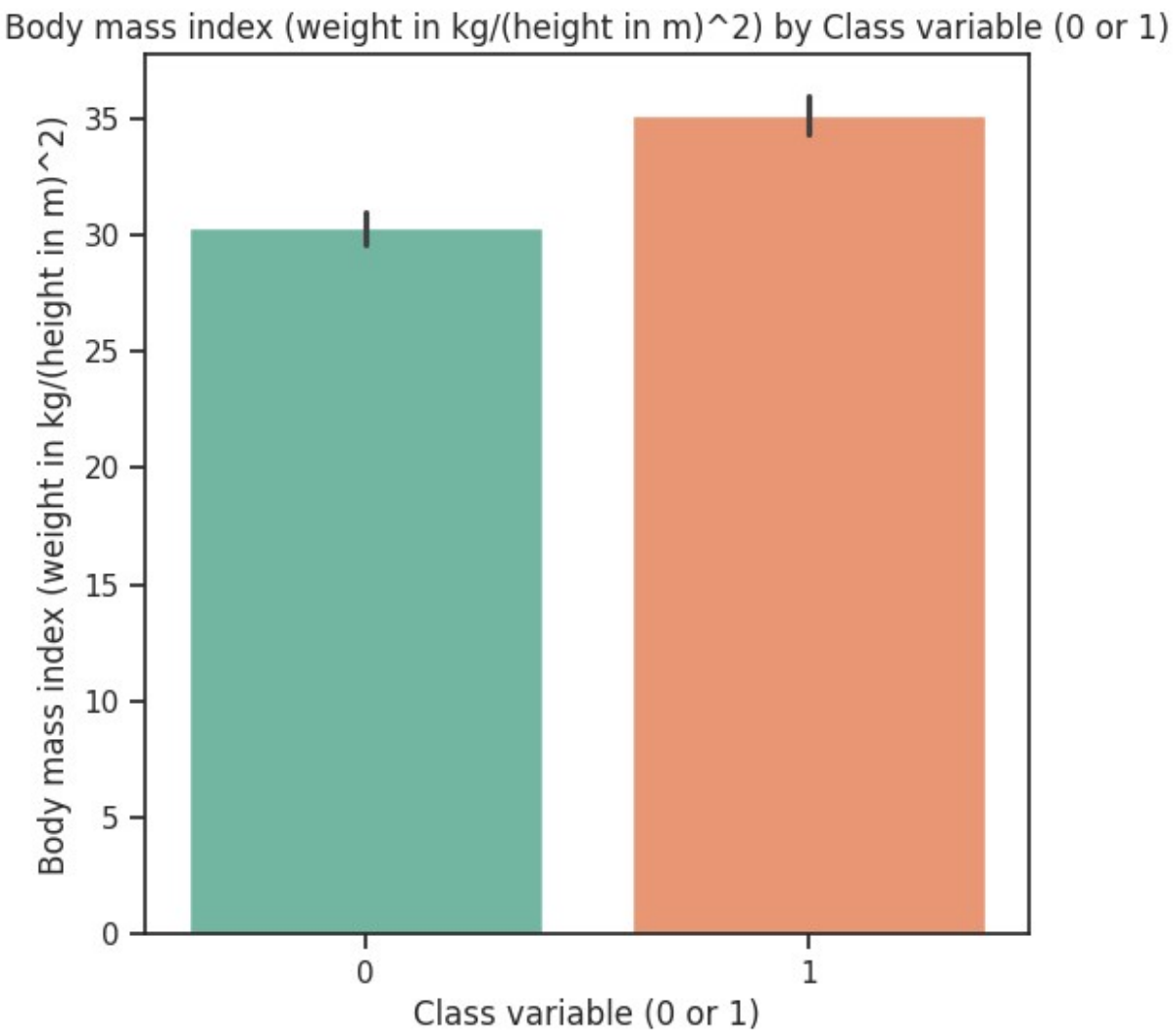

Number of times pregnant by Class variable (0 or 1)
Number of times pregnant
0
1
2
3
4
5
0
1
Class variable (0 or 1)

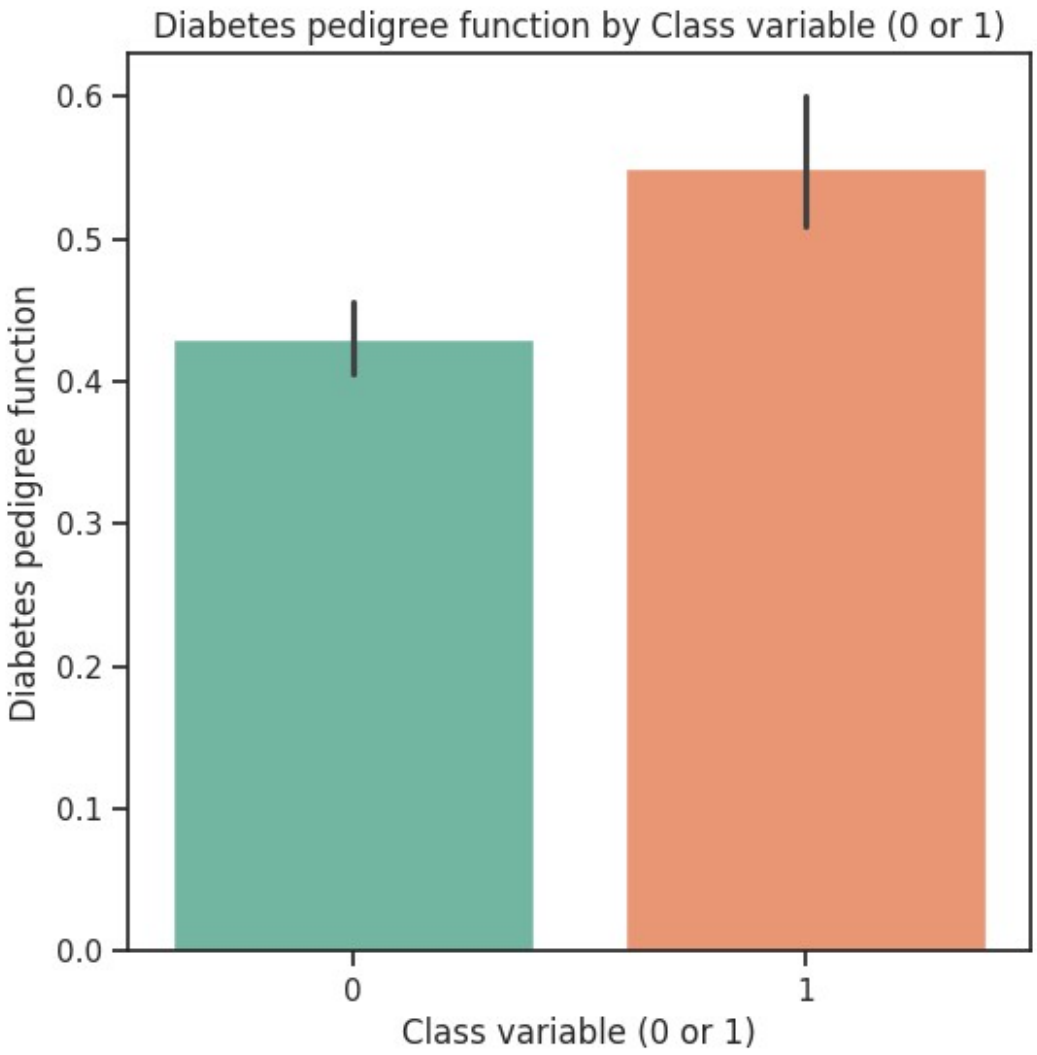

Plasma glucose concentration a 2 hours in an oral glucose tolerance test by Class variable (0 or 1)
lasma glucose concentration a 2 hours in an oral glucose tolerance
0
20
40
60
80
100
120
140
0
1
Class variable (0 or 1)

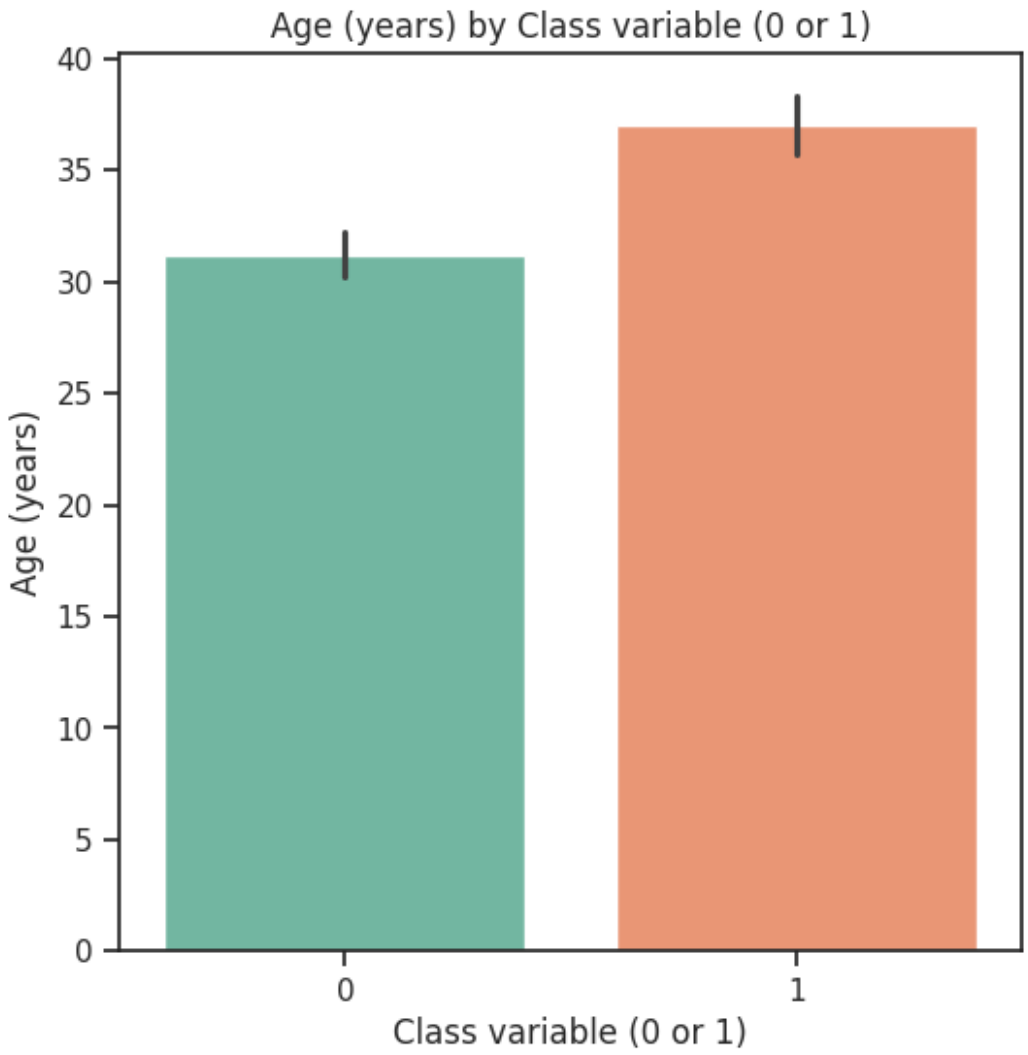

Diastolic blood pressure (mm Hg) by Class variable (0 or 1)
Diastolic blood pressure (mm Hg)
0
10
20
30
40
50
60
70
0
1
Class variable (0 or 1)


Triceps skin fold thickness (mm) by Class variable (0 or 1)
Triceps skin fold thickness (mm)
25
20
15
10
5
0
0
1
Class variable (0 or 1)


2-Hour serum insulin (mu U/ml) by Class variable (0 or 1)
120
100
80
60
40
20
0
2-Hour serum insulin (mu U/ml)
0
1
Class variable (0 or 1)


Body mass index (weight in kg/(height in m)^2) by Class variable (0 or 1)
Body mass index (weight in kg/(height in m)^2)
0
5
10
15
20
25
30
35
0
1
Class variable (0 or 1)


Diabetes pedigree function by Class variable (0 or 1)
Diabetes pedigree function
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0
1
Class variable (0 or 1)




```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

model = LogisticRegression(max_iter = 1000)

model.fit(X_train, y_train)

LogisticRegression(max_iter=1000)


y_pred = model.predict(X_test)

cm = confusion_matrix(y_test,y_pred)
print(cm)


accuracy = accuracy_score(y_test, y_pred)
print('Accuracy of the binary classifier = {:0.3f}'.format(accuracy))
```

```
print(classification_report(y_test, y_pred))
```

```
[[78 21]
 [18 37]]
Accuracy of the binary classifier = 0.747
              precision    recall  f1-score   support

           0       0.81      0.79      0.80        99
           1       0.64      0.67      0.65        55

    accuracy                           0.75       154
   macro avg       0.73      0.73      0.73       154
weighted avg       0.75      0.75      0.75       154
```