USECASE UNIT II

DIABETES DATA-SET

BY ADIEL DELGADO

CONTENT:

- INTRODUCTION
- DATASET
- EDA
- DATA TRANSFORMATION
- COMPARASION

INTRODUCTION

Why did I choose this dataset?

Diabetes is among the top 10 diseases that cause deaths in the world, this because the body produces a lot of glucose that cannot be regulated by the body.

I chose this Dataset because I believe that a model that can predict whether a person has diabetes can save lives.

DATASET

FEATURES

Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age, **Outcome**

VALUES

768 Values in the Dataset

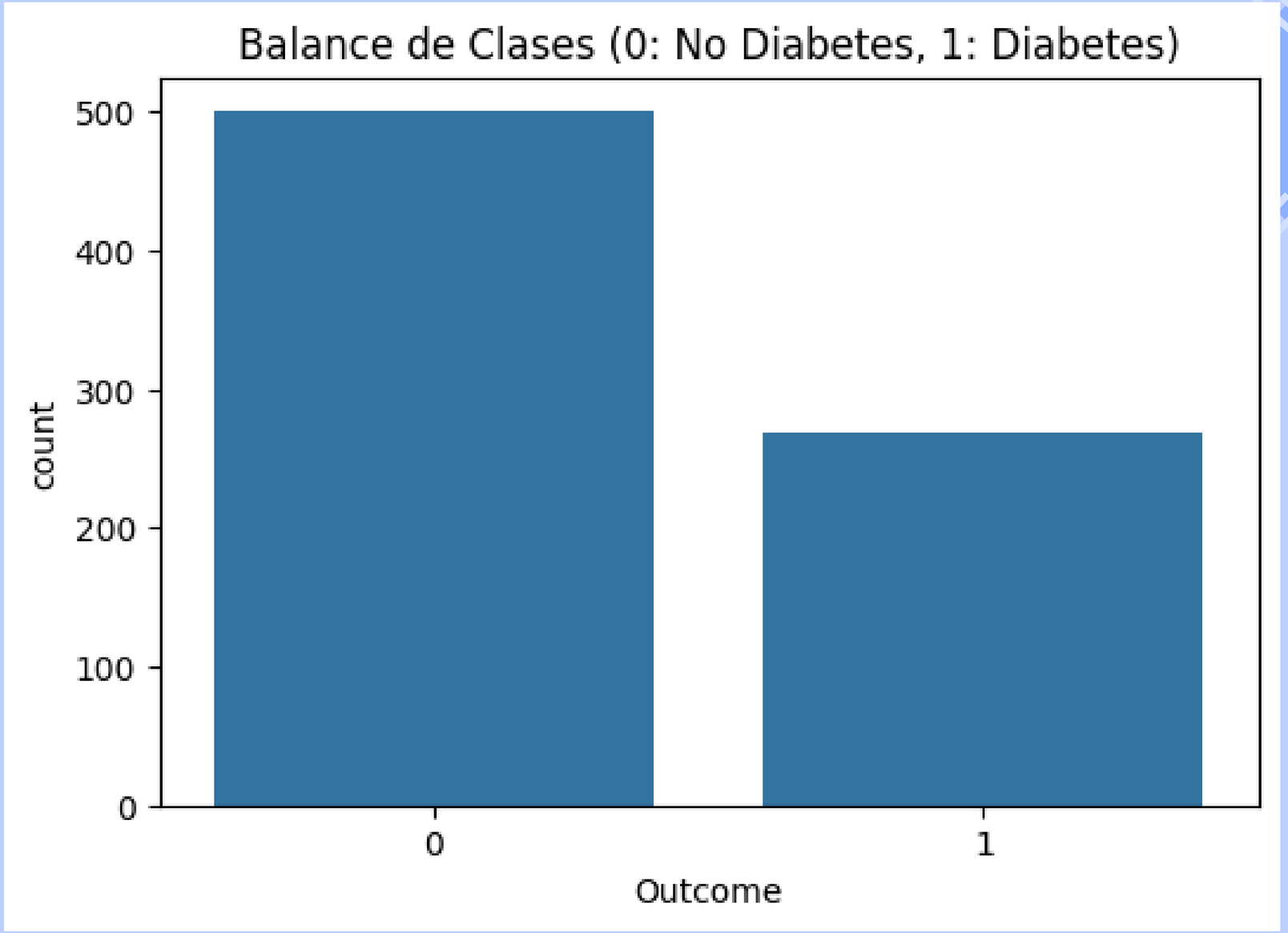
EDA

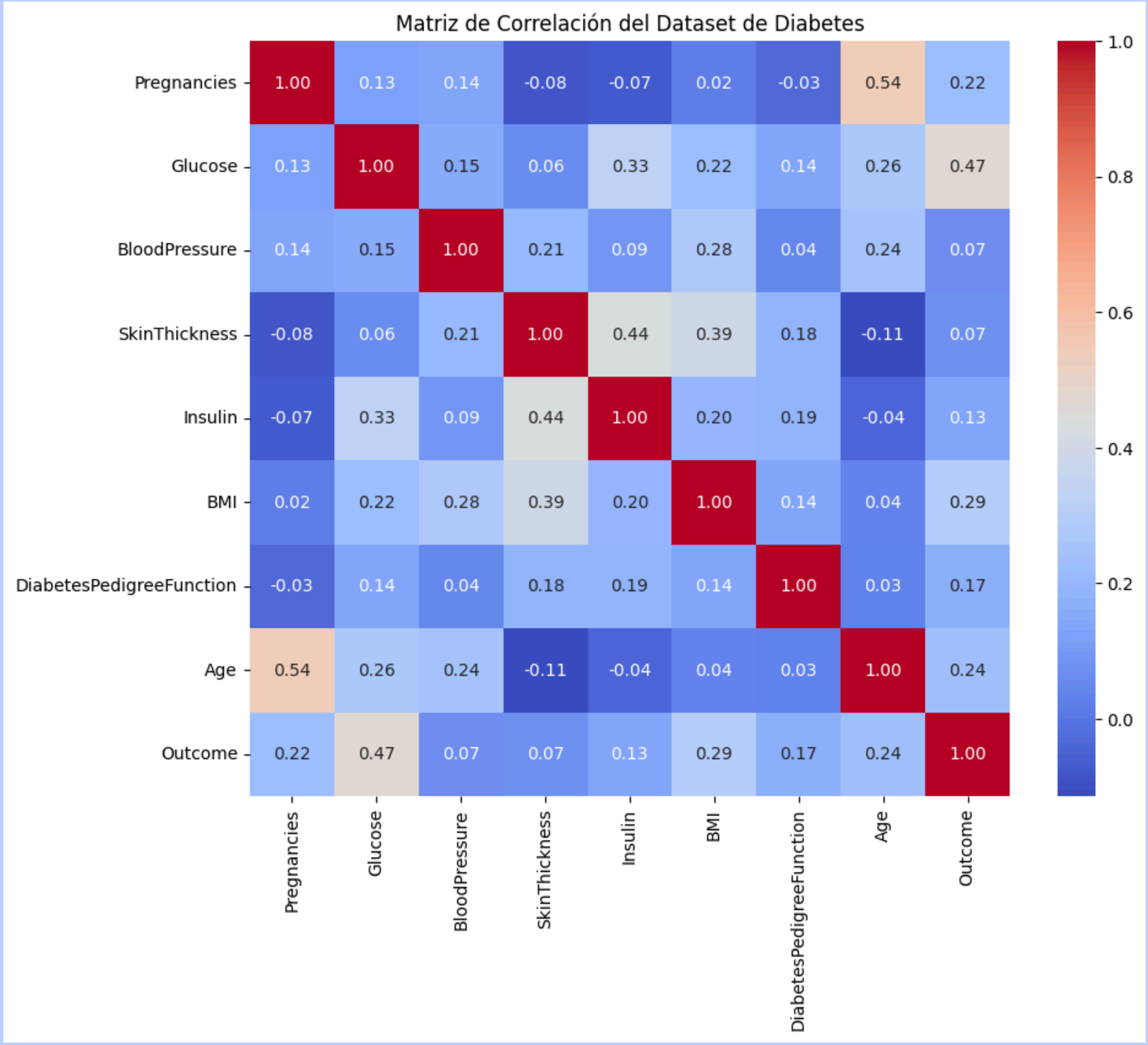
1st
Outcome Balance

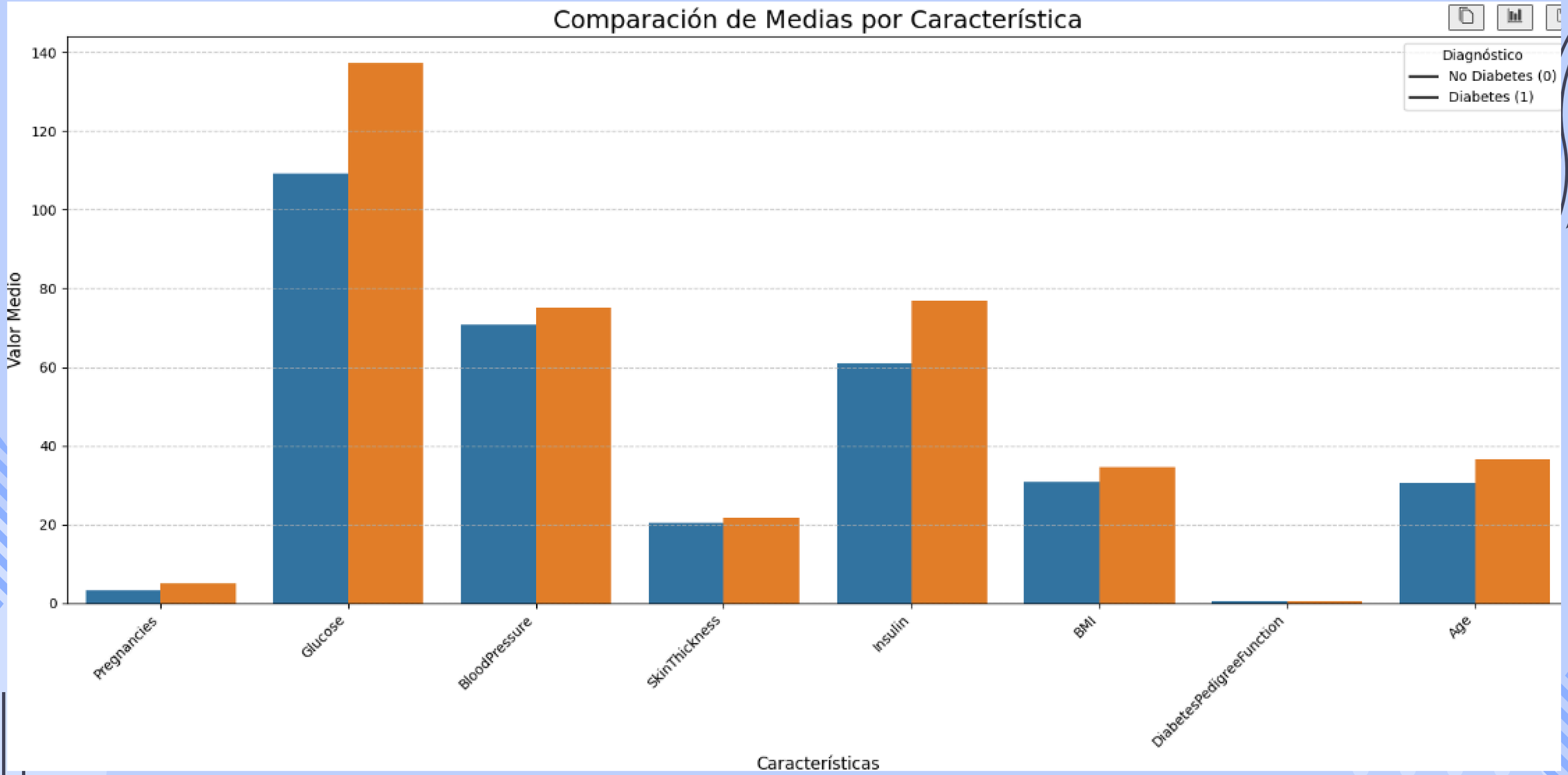
2nd
Correlation Heatmap

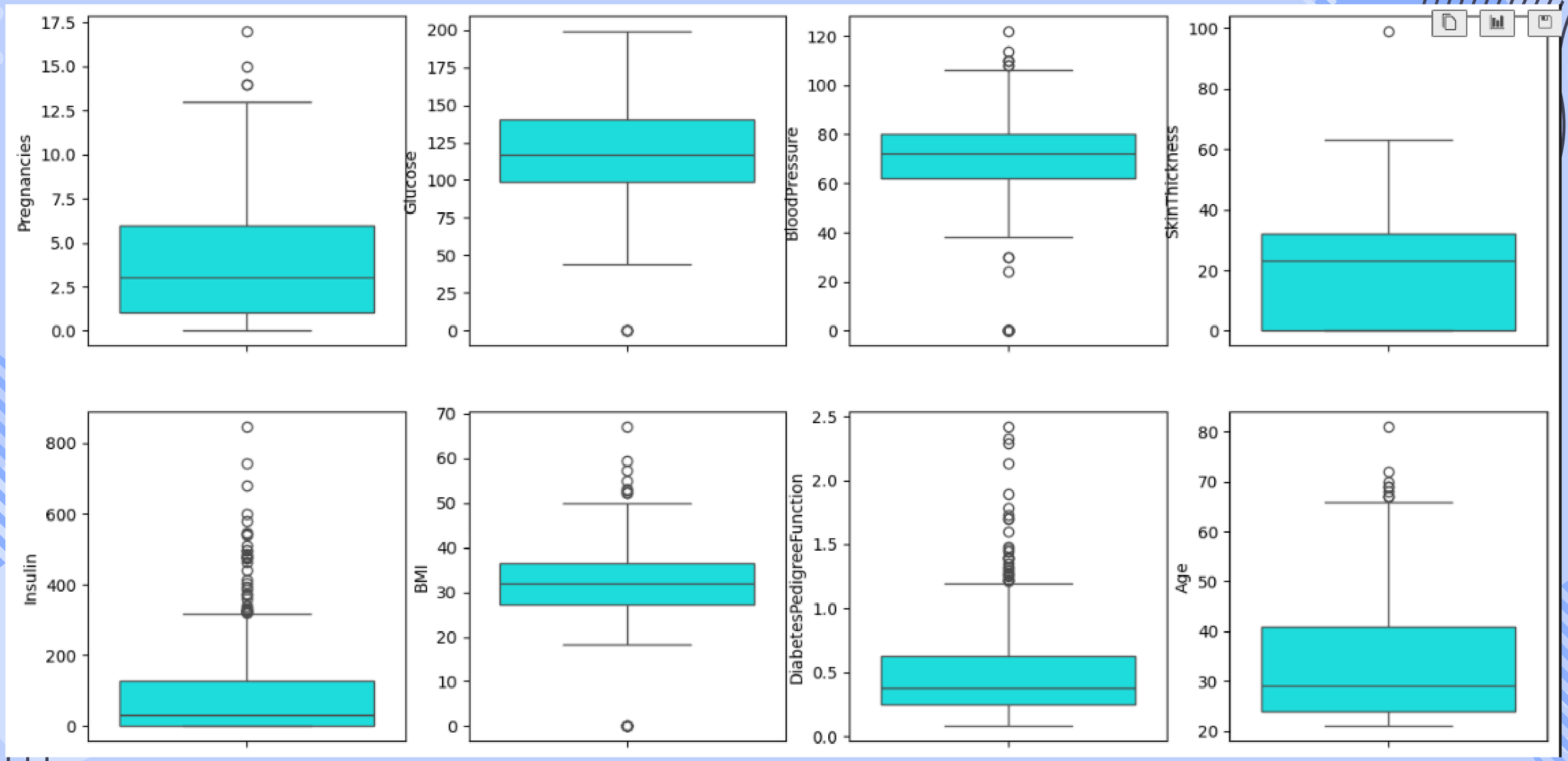
3rd
Mean Depending of the Outcome

4th
Outliers


Balance de Clases (0: No Diabetes, 1: Diabetes)
count
500
400
300
200
100
0
0
1
Outcome


Matriz de Correlación del Dataset de Diabetes
Pregnancies 1.00 0.13 0.14 -0.08 -0.07 0.02 -0.03 0.54 0.22
Glucose 0.13 1.00 0.15 0.06 0.33 0.22 0.14 0.26 0.47
BloodPressure 0.14 0.15 1.00 0.21 0.09 0.28 0.04 0.24 0.07
SkinThickness -0.08 0.06 0.21 1.00 0.44 0.39 0.18 -0.11 0.07
Insulin -0.07 0.33 0.09 0.44 1.00 0.20 0.19 -0.04 0.13
BMI 0.02 0.22 0.28 0.39 0.20 1.00 0.14 0.04 0.29
DiabetesPedigreeFunction -0.03 0.14 0.04 0.18 0.19 0.14 1.00 0.03 0.17
Age 0.54 0.26 0.24 -0.11 -0.04 0.04 0.03 1.00 0.24
Outcome 0.22 0.47 0.07 0.07 0.13 0.29 0.17 0.24 1.00
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Outcome
1.0
0.8
0.6
0.4
0.2
0.0


Comparación de Medias por Característica
Diagnóstico
No Diabetes (0)
Diabetes (1)
Valor Medio
0
20
40
60
80
100
120
140
Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Características


Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age

DATA TRANSFORMATION

```python
def countoutlier(df, col):
    iqr = df[col].quantile(0.75) - df[col].quantile(0.25) #Inter-quartile range
    ul = (1.5 * iqr) + df[col].quantile(0.75) #upper limit
    ll = df[col].quantile(0.25) - (1.5 * iqr) #lower limit
    print(f'There are {(df[col] > ul).sum()} values greater than upper limit in {col} column')
    print(f'There are {(df[col] < ll).sum()} values less than lower limit in {col} column')
    print('')
```

```python
for i in df.columns:
    countoutlier(df, i)
```

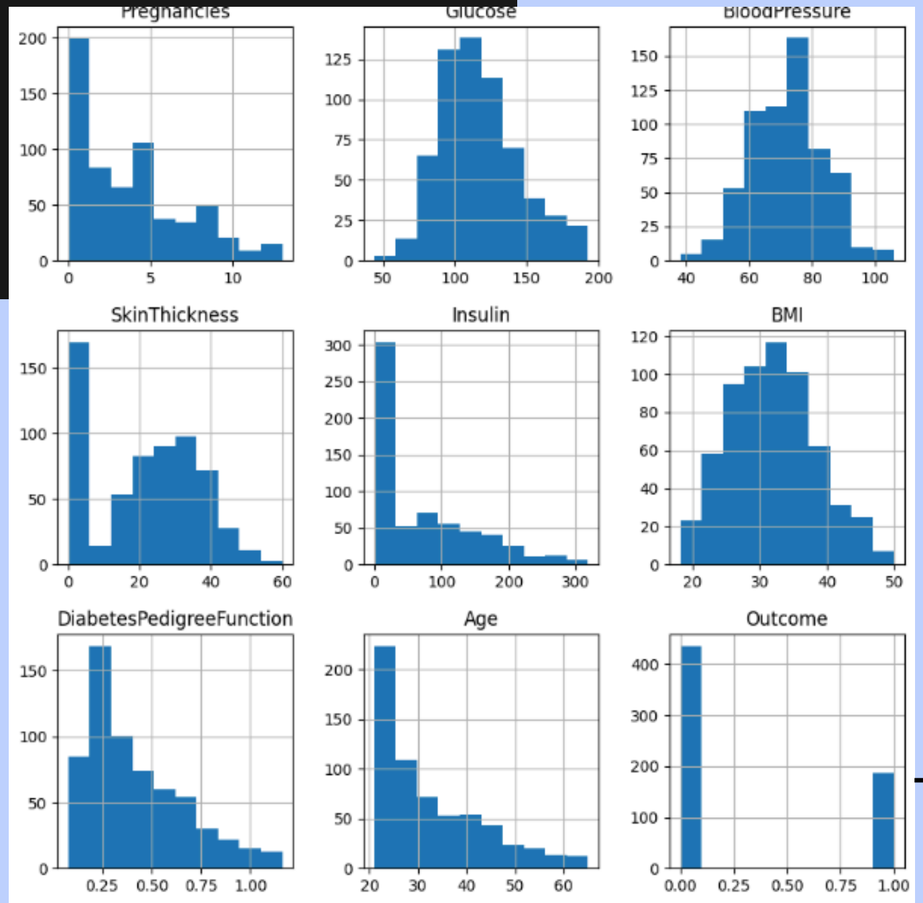
✓ 0.0s

```
There are 4 values greater than upper limit in Pregnancies column
There are 0 values less than lower limit in Pregnancies column

There are 0 values greater than upper limit in Glucose column
There are 5 values less than lower limit in Glucose column

There are 7 values greater than upper limit in BloodPressure column
There are 38 values less than lower limit in BloodPressure column

There are 1 values greater than upper limit in SkinThickness column
There are 0 values less than lower limit in SkinThickness column

There are 34 values greater than upper limit in Insulin column
There are 0 values less than lower limit in Insulin column

There are 8 values greater than upper limit in BMI column
There are 11 values less than lower limit in BMI column

There are 29 values greater than upper limit in DiabetesPedigreeFunction column
There are 0 values less than lower limit in DiabetesPedigreeFunction column

There are 9 values greater than upper limit in Age column
There are 0 values less than lower limit in Age column

There are 0 values greater than upper limit in Outcome column
There are 0 values less than lower limit in Outcome column
```

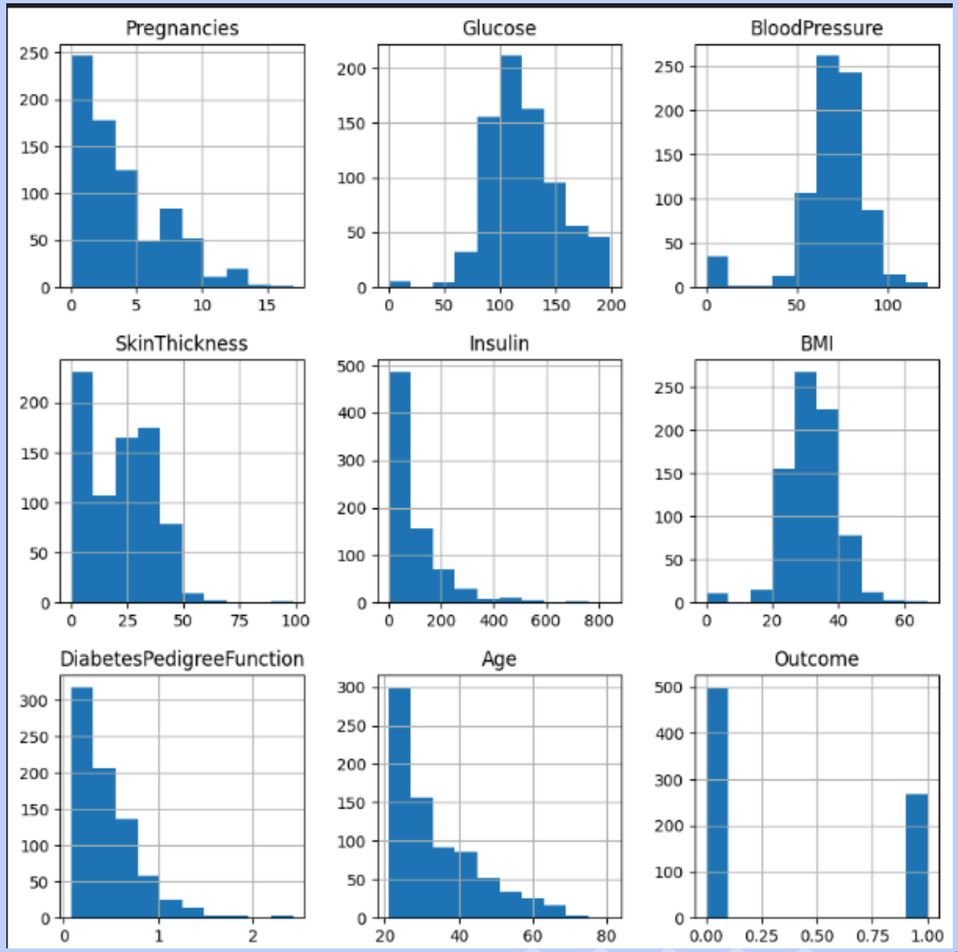
```python
def treating(feature):
    global df
    limit_1 = df[feature].quantile(0.25)
    limit_3 = df[feature].quantile(0.75)
    iqr = limit_3 - limit_1
    lower_limit = limit_1 - 1.5 * iqr
    upper_limit = limit_3 + 1.5 * iqr

    df = df[(df[feature] > lower_limit) & (df[feature] < upper_limit)]
```

✓ 0.0s

```python
treating("Insulin")
treating("Age")
treating("Glucose")
treating("BMI")
treating("Pregnancies")
treating("BloodPressure")
treating("SkinThickness")
treating("DiabetesPedigreeFunction")
```



Treated



OG

ML MODELS

I decided to use these three models

DECISION TREE MODEL IMPLEMENTATION WITH HYPERPARAMETER TUNING

El compromiso con nuestro planeta es un pilar fundamental de la empresa y de este proyecto.

LOGISTIC REGRESSION

El uso de la más última tecnología es fundamental para el desarrollo óptimo de nuestros objetivos.

RANDOM FOREST WITH HYPERPARAMETER TUNING

La eficiencia, tanto en cuestión de tiempo como en costo, es muy importante para nosotros.

Model	Normal	HyperParameter Tunning	SMOTE
Decision Tree	Train = 93% Test = 67%	Train = 85% Test = 71%	Train = 84% Test = 76%
Logistic Regression	-	-	Train = 76% Test = 75%
Random Forest	Train = 95% Test = 75%	Train = 99% Test = 74%	Train = 98% Test = 81%

Q&A