```
pip install numpy
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in c:\users\sairam\appdata\roaming\python\python312\site-packages (2.1.2)
Note: you may need to restart the kernel to use updated packages.
```

```
pip install pandas
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in c:\users\sairam\appdata\roaming\python\python312\site-packages (2.2.3)
Requirement already satisfied: numpy>=1.26.0 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas) (2.1.2)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas) (2
Requirement already satisfied: pytz>=2020.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas) (2024.2)
Requirement already satisfied: six>=1.5 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from python-dateutil>=2.8.2->
Note: you may need to restart the kernel to use updated packages.
```

```
pip install seaborn
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: seaborn in c:\users\sairam\appdata\roaming\python\python312\site-packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from seaborn) (2.
Requirement already satisfied: pandas>=1.2 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from seaborn) (2.2.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from seaborn)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.
Requirement already satisfied: cycler>=0.10 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=
Requirement already satisfied: fonttools>=4.22.0 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6
Requirement already satisfied: packaging>=20.0 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1
Requirement already satisfied: pillow>=8 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.
Requirement already satisfied: python-dateutil>=2.7 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib!=
Requirement already satisfied: pytz>=2020.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas>=1.2->seaborn
Requirement already satisfied: tzdata>=2022.7 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas>=1.2->seabo
Requirement already satisfied: six>=1.5 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from python-dateutil>=2.7->ma
Note: you may need to restart the kernel to use updated packages.
```

```
pip install statsmodels
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: statsmodels in c:\users\sairam\appdata\roaming\python\python312\site-packages (0.14.4)
Requirement already satisfied: numpy<3,>=1.22.3 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from statsmodels) (2.
Requirement already satisfied: scipy!=1.9.2,>=1.8 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from statsmodels) (
Requirement already satisfied: pandas!=2.1.0,>=1.4 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from statsmodels)
Requirement already satisfied: patsy>=0.5.6 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from statsmodels) (0.5.6)
Requirement already satisfied: packaging>=21.3 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from statsmodels) (24.
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas!=2.
Requirement already satisfied: pytz>=2020.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas!=2.1.0,>=1.4-
Requirement already satisfied: tzdata>=2022.7 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas!=2.1.0,>=1.
Requirement already satisfied: six in c:\users\sairam\appdata\roaming\python\python312\site-packages (from patsy>=0.5.6->statsmodels) (1
Note: you may need to restart the kernel to use updated packages.
```

```
pip install matplotlib
```

```
Defaulting to user installation because normal site-packages is not writeableNote: you may need to restart the kernel to use updated pac

Requirement already satisfied: matplotlib in c:\users\sairam\appdata\roaming\python\python312\site-packages (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (1.3
Requirement already satisfied: cycler>=0.10 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (4.
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (1.
Requirement already satisfied: numpy>=1.23 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (2.1.2)
Requirement already satisfied: packaging>=20.0 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (24.1
Requirement already satisfied: pillow>=8 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib) (3.2
Requirement already satisfied: python-dateutil>=2.7 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib)
Requirement already satisfied: six>=1.5 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from python-dateutil>=2.7->ma
```

```
pip install sklearn
```

```
Defaulting to user installation because normal site-packages is not writeableNote: you may need to restart the kernel to use updated pac
  error: subprocess-exited-with-error

  × Getting requirements to build wheel did not run successfully.
```

```
      │ exit code: 1
      └─> [15 lines of output]
          The 'sklearn' PyPI package is deprecated, use 'scikit-learn'
          rather than 'sklearn' for pip commands.

          Here is how to fix this error in the main use cases:
          - use 'pip install scikit-learn' rather than 'pip install sklearn'
          - replace 'sklearn' by 'scikit-learn' in your pip requirements files
            (requirements.txt, setup.py, setup.cfg, Pipfile, etc ...)
          - if the 'sklearn' package is used by one of your dependencies,
            it would be great if you take some time to track which package uses
            'sklearn' instead of 'scikit-learn' and report it to their issue tracker
          - as a last resort, set the environment variable
            SKLEARN_ALLOW_DEPRECATED_SKLEARN_PACKAGE_INSTALL=True to avoid this error

          More information is available at
          https://github.com/scikit-learn/sklearn-pypi-package
          [end of output]

    note: This error originates from a subprocess, and is likely not a problem with pip.
  error: subprocess-exited-with-error

  × Getting requirements to build wheel did not run successfully.
  │ exit code: 1
  └─> See above for output.

  note: This error originates from a subprocess, and is likely not a problem with pip.

  Collecting sklearn
    Using cached sklearn-0.0.post12.tar.gz (2.6 kB)
    Installing build dependencies: started
    Installing build dependencies: finished with status 'done'
    Getting requirements to build wheel: started
    Getting requirements to build wheel: finished with status 'error'
```

```
import sklearn
print(sklearn.__version__)
```

```
1.5.2
```

```
pip install missingno
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting missingno
  Downloading missingno-0.5.2-py3-none-any.whl.metadata (639 bytes)
Requirement already satisfied: numpy in c:\users\sairam\appdata\roaming\python\python312\site-packages (from missingno) (2.1.2)
Requirement already satisfied: matplotlib in c:\users\sairam\appdata\roaming\python\python312\site-packages (from missingno) (3.9.2)
Requirement already satisfied: scipy in c:\users\sairam\appdata\roaming\python\python312\site-packages (from missingno) (1.14.1)
Requirement already satisfied: seaborn in c:\users\sairam\appdata\roaming\python\python312\site-packages (from missingno) (0.13.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->miss
Requirement already satisfied: cycler>=0.10 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->missingn
Requirement already satisfied: fonttools>=4.22.0 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->mis
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->mis
Requirement already satisfied: packaging>=20.0 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->missi
Requirement already satisfied: pillow>=8 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->missingno)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->miss
Requirement already satisfied: python-dateutil>=2.7 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from matplotlib->
Requirement already satisfied: pandas>=1.2 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from seaborn->missingno) (
Requirement already satisfied: pytz>=2020.1 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas>=1.2->seaborn
Requirement already satisfied: tzdata>=2022.7 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from pandas>=1.2->seabo
Requirement already satisfied: six>=1.5 in c:\users\sairam\appdata\roaming\python\python312\site-packages (from python-dateutil>=2.7->ma
Downloading missingno-0.5.2-py3-none-any.whl (8.7 kB)
Installing collected packages: missingno
Successfully installed missingno-0.5.2
Note: you may need to restart the kernel to use updated packages.
```

```
pip install xgboost
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting xgboost
  Downloading xgboost-2.1.1-py3-none-win_amd64.whl.metadata (2.1 kB)
Requirement already satisfied: numpy in c:\users\sairam\appdata\roaming\python\python312\site-packages (from xgboost) (2.1.2)
Requirement already satisfied: scipy in c:\users\sairam\appdata\roaming\python\python312\site-packages (from xgboost) (1.14.1)
Downloading xgboost-2.1.1-py3-none-win_amd64.whl (124.9 MB)
   ---------------------------------------- 0.0/124.9 MB ? eta -:--:--
    --------------------------------------- 2.1/124.9 MB 9.8 MB/s eta 0:00:13
    --------------------------------------- 2.9/124.9 MB 7.6 MB/s eta 0:00:17
   - -------------------------------------- 3.7/124.9 MB 6.6 MB/s eta 0:00:19
```

```
- ---------------------------------------- 4.5/124.9 MB 5.5 MB/s eta 0:00:22
- ---------------------------------------- 5.0/124.9 MB 5.0 MB/s eta 0:00:25
- ---------------------------------------- 5.8/124.9 MB 4.5 MB/s eta 0:00:27
-- ---------------------------------------- 6.8/124.9 MB 4.6 MB/s eta 0:00:26
-- ---------------------------------------- 8.1/124.9 MB 4.8 MB/s eta 0:00:25
-- ---------------------------------------- 9.2/124.9 MB 4.8 MB/s eta 0:00:24
--- ---------------------------------------- 9.7/124.9 MB 4.7 MB/s eta 0:00:25
--- ---------------------------------------- 10.7/124.9 MB 4.6 MB/s eta 0:00:26
--- ---------------------------------------- 11.8/124.9 MB 4.6 MB/s eta 0:00:25
---- ---------------------------------------- 13.4/124.9 MB 4.8 MB/s eta 0:00:24
---- ---------------------------------------- 14.7/124.9 MB 5.0 MB/s eta 0:00:22
----- ---------------------------------------- 15.7/124.9 MB 5.0 MB/s eta 0:00:22
----- ---------------------------------------- 16.5/124.9 MB 4.9 MB/s eta 0:00:23
----- ---------------------------------------- 17.6/124.9 MB 4.9 MB/s eta 0:00:22
----- ---------------------------------------- 18.6/124.9 MB 4.9 MB/s eta 0:00:22
------ ---------------------------------------- 19.4/124.9 MB 4.9 MB/s eta 0:00:22
------ ---------------------------------------- 20.2/124.9 MB 4.8 MB/s eta 0:00:22
------ ---------------------------------------- 21.5/124.9 MB 4.9 MB/s eta 0:00:22
------- ---------------------------------------- 23.1/124.9 MB 5.0 MB/s eta 0:00:21
------- ---------------------------------------- 24.9/124.9 MB 5.1 MB/s eta 0:00:20
-------- ---------------------------------------- 25.7/124.9 MB 5.1 MB/s eta 0:00:20
-------- ---------------------------------------- 26.7/124.9 MB 5.1 MB/s eta 0:00:20
-------- ---------------------------------------- 28.0/124.9 MB 5.1 MB/s eta 0:00:19
--------- ---------------------------------------- 29.9/124.9 MB 5.3 MB/s eta 0:00:19
--------- ---------------------------------------- 31.2/124.9 MB 5.3 MB/s eta 0:00:18
---------- ---------------------------------------- 32.0/124.9 MB 5.2 MB/s eta 0:00:18
---------- ---------------------------------------- 32.8/124.9 MB 5.2 MB/s eta 0:00:18
---------- ---------------------------------------- 34.1/124.9 MB 5.2 MB/s eta 0:00:18
---------- ---------------------------------------- 35.7/124.9 MB 5.3 MB/s eta 0:00:17
----------- ---------------------------------------- 37.2/124.9 MB 5.4 MB/s eta 0:00:17
------------ ---------------------------------------- 38.8/124.9 MB 5.4 MB/s eta 0:00:16
------------ ---------------------------------------- 39.6/124.9 MB 5.4 MB/s eta 0:00:16
------------ ---------------------------------------- 40.6/124.9 MB 5.4 MB/s eta 0:00:16
------------ ---------------------------------------- 41.7/124.9 MB 5.4 MB/s eta 0:00:16
------------ ---------------------------------------- 43.5/124.9 MB 5.4 MB/s eta 0:00:15
------------- ---------------------------------------- 45.1/124.9 MB 5.5 MB/s eta 0:00:15
------------- ---------------------------------------- 45.9/124.9 MB 5.5 MB/s eta 0:00:15
------------- ---------------------------------------- 46.7/124.9 MB 5.4 MB/s eta 0:00:15
-------------- ---------------------------------------- 47.7/124.9 MB 5.4 MB/s eta 0:00:15
-------------- ---------------------------------------- 49.3/124.9 MB 5.5 MB/s eta 0:00:14
-------------- ---------------------------------------- 51.1/124.9 MB 5.6 MB/s eta 0:00:14
-------------- ---------------------------------------- 52.2/124.9 MB 5.5 MB/s eta 0:00:14
--------------- ---------------------------------------- 53.5/124.9 MB 5.6 MB/s eta 0:00:13
--------------- ---------------------------------------- 54.5/124.9 MB 5.6 MB/s eta 0:00:13
--------------- ---------------------------------------- 55.6/124.9 MB 5.5 MB/s eta 0:00:13
---------------- ---------------------------------------- 56.6/124.9 MB 5.5 MB/s eta 0:00:13
---------------- ---------------------------------------- 58.2/124.9 MB 5.6 MB/s eta 0:00:12
---------------- ---------------------------------------- 59.8/124.9 MB 5.6 MB/s eta 0:00:12
```

```
import numpy as np
import pandas as pd
import seaborn as sns
import statsmodels.api as sm
import matplotlib.pyplot as plt
from sklearn.preprocessing import scale, StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.metrics import confusion_matrix, accuracy_score, mean_squared_error, r2_score, roc_auc_score, roc_curve, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.model_selection import KFold
import warnings
warnings.simplefilter(action='ignore')
sns.set()
plt.style.use("ggplot")
%matplotlib inline
```

```
# read the dataset from dir
df = pd.read_csv("C:\Sunaina\Desktop\Dieseas Prediction app\dataset\diabetes.csv")
```

```
df.head()
```

```
# Supervised or Unsupervised
# ans: Supervised->target column
```

```
# Regreesion or classification
# ans: classification

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
```

```
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
# independent feature->'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
#        'BMI', 'DiabetesPedigreeFunction', 'Age'
# dependent feature-> outcome

# descriptive statistics of the dataset
df.describe()
```

```
# (row, columns)
df.shape
```

```
(768, 9)
```

```
# distribution of outcome variable
df.Outcome.value_counts()*100/len(df)
```

```
Outcome
0    65.104167
1    34.895833
Name: count, dtype: float64
```

```
df['Outcome'].value_counts()*100/len(df)
```

```
Outcome
0    65.104167
1    34.895833
Name: count, dtype: float64
```

```
# plot the hist of the age variable
plt.figure(figsize=(8,7))
plt.xlabel('Age', fontsize=10)
plt.ylabel('Count', fontsize=10)
df['Age'].hist(edgecolor="black")
```

```
df['Age'].max()
```

```
np.int64(81)
```

```
df['Age'].min()
```

```
np.int64(21)
```

```
print("MAX AGE: "+str(df['Age'].max()))
print("MIN AGE: "+str(df['Age'].min()))
```

```
MAX AGE: 81
MIN AGE: 21
```

```
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
# density graph
# 4*2=8
# columns=2 figure
# having 4 row

# [0,0], [0,1]
# [1,0], [1,1]
# [2,0], [2,1]
# [3,0], [3,1]

fig,ax = plt.subplots(4,2, figsize=(20,20))
sns.distplot(df.Pregnancies, bins=20, ax=ax[0,0], color="red")
sns.distplot(df.Glucose, bins=20, ax=ax[0,1], color="red")
sns.distplot(df.BloodPressure, bins=20, ax=ax[1,0], color="red")
sns.distplot(df.SkinThickness, bins=20, ax=ax[1,1], color="red")
sns.distplot(df.Insulin, bins=20, ax=ax[2,0], color="red")
sns.distplot(df.BMI, bins=20, ax=ax[2,1], color="red")
sns.distplot(df.DiabetesPedigreeFunction, bins=20, ax=ax[3,0], color="red")
sns.distplot(df.Age, bins=20, ax=ax[3,1], color="red")
```

```
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
df.groupby("Outcome").agg({'Pregnancies':'mean'})
```

```
df.groupby("Outcome").agg({'Pregnancies':'max'})
```

```
df.groupby("Outcome").agg({'Glucose':'mean'})
```

```
df.groupby("Outcome").agg({'Glucose':'max'})
```

```
# Homework
# 'BloodPressure', 'SkinThickness', 'Insulin',
#        'BMI', 'DiabetesPedigreeFunction', 'Age'
#     groupby-> mean/max


# 0>healthy
# 1>diabetes

f,ax = plt.subplots(1,2, figsize=(18,8))
df['Outcome'].value_counts().plot.pie(explode=[0,0.1],autopct = "%1.1f%%", ax=ax[0], shadow=True)
ax[0].set_title('target')
ax[0].set_ylabel('')
sns.countplot(x = 'Outcome', data=df, ax=ax[0])
ax[1].set_title('Outcome')
plt.show()
```

```
df.corr()
```

```
f,ax = plt.subplots(figsize=[20,15])
sns.heatmap(df.corr(), annot=True, fmt = '.2f', ax=ax, cmap='magma')
ax.set_title("Correlation Matrix", fontsize=20)
plt.show()
```

```
# EDA Part Completed
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

```
df[['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age']] = df[['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age']].replace(0, np.nan)
```

```
# Data preprocessing Part
df.isnull().sum()
```

```
Pregnancies                111
Glucose                      5
BloodPressure               35
SkinThickness              227
Insulin                    374
BMI                         11
DiabetesPedigreeFunction     0
Age                          0
Outcome                      0
dtype: int64
```

```
df.head()
```

```
import missingno as msno
msno.bar(df, color="orange")
```

```
#median
def median_target(var):
    temp = df[df[var].notnull()]
    temp = temp[[var, 'Outcome']].groupby(['Outcome'])[[var]].median().reset_index()
    return temp
```

```
columns = df.columns
columns = columns.drop("Outcome")
for i in columns:
    median_target(i)
    df.loc[(df['Outcome'] == 0 ) & (df[i].isnull()), i] = median_target(i)[i][0]
    df.loc[(df['Outcome'] == 1 ) & (df[i].isnull()), i] = median_target(i)[i][1]
```

```
df.head()
```

```
df.isnull().sum()
```

```
Pregnancies                0
Glucose                    0
BloodPressure              0
SkinThickness              0
Insulin                    0
```

```
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

```
# pair plot
p = sns.pairplot(df, hue="Outcome")
```

```
# Outlier Detection
# IQR+Q1
# 50%
# 24.65->25%+50%
# 24.65->25%
for feature in df:
    Q1 = df[feature].quantile(0.25)
    Q3 = df[feature].quantile(0.75)
    IQR = Q3-Q1
    lower = Q1-1.5*IQR
    upper = Q3+1.5*IQR
    if df[(df[feature]>upper)].any(axis=None):
        print(feature, "yes")
    else:
        print(feature, "no")
```

```
Pregnancies yes
Glucose no
BloodPressure yes
SkinThickness yes
Insulin yes
BMI yes
DiabetesPedigreeFunction yes
Age yes
Outcome no
```

```
plt.figure(figsize=(8,7))
sns.boxplot(x= df["Insulin"], color="red")
```

```
Q1 = df.Insulin.quantile(0.25)
Q3 = df.Insulin.quantile(0.75)
IQR = Q3-Q1
lower = Q1-1.5*IQR
upper = Q3+1.5*IQR
df.loc[df['Insulin']>upper, "Insulin"] = upper
```

```
plt.figure(figsize=(8,7))
sns.boxplot(x= df["Insulin"], color="red")
```

```
# LOF
# local outlier factor
from sklearn.neighbors import LocalOutlierFactor
lof = LocalOutlierFactor(n_neighbors=10)
lof.fit_predict(df)
```

```
array([ 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1, -1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
```

```
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1, -1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
-1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1, -1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1, -1,  1,  1,  1, -1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, -1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
 1,  1,  1])
```

```
df.head()
```

```
plt.figure(figsize=(8,7))
sns.boxplot(x= df["Pregnancies"], color="red")
```

```
df_scores = lof.negative_outlier_factor_
np.sort(df_scores)[0:20]
```

```
array([-3.06509976, -2.38250393, -2.15557018, -2.11501347, -2.08356175,
       -1.95386655, -1.83559384, -1.74974237, -1.7330214 , -1.71017168,
       -1.70215105, -1.68722889, -1.64294601, -1.64180205, -1.61181746,
       -1.61067772, -1.60925053, -1.60214364, -1.59998552, -1.58761193])
```

```
thresold = np.sort(df_scores)[7]
```

```
thresold
```

```
np.float64(-1.7497423670960557)
```

```
outlier = df_scores>thresold
```

```
df = df[outlier]
```

```
df.head()
```

```
df.shape
```

```
(760, 9)
```

```
plt.figure(figsize=(8,7))
sns.boxplot(x= df["Pregnancies"], color="red")
```

```
# Feature Enginnering
NewBMI = pd.Series(["Underweight","Normal", "Overweight","Obesity 1", "Obesity 2", "Obesity 3"], dtype = "category")
```

```
NewBMI
```

```
0    Underweight
1         Normal
2     Overweight
3      Obesity 1
4      Obesity 2
5      Obesity 3
dtype: category
Categories (6, object): ['Normal', 'Obesity 1', 'Obesity 2', 'Obesity 3', 'Overweight', 'Underweight']
```

```
df['NewBMI'] = NewBMI
df.loc[df["BMI"]<18.5, "NewBMI"] = NewBMI[0]
df.loc[(df["BMI"]>18.5) & df["BMI"]<=24.9, "NewBMI"] = NewBMI[1]
df.loc[(df["BMI"]>24.9) & df["BMI"]<=29.9, "NewBMI"] = NewBMI[2]
df.loc[(df["BMI"]>29.9) & df["BMI"]<=34.9, "NewBMI"] = NewBMI[3]
df.loc[(df["BMI"]>34.9) & df["BMI"]<=39.9, "NewBMI"] = NewBMI[4]
df.loc[df["BMI"]>39.9, "NewBMI"] = NewBMI[5]
```

```
df.head()
```

```
# if insulin>=16 & insuline<=166->normal
def set_insuline(row):
    if row["Insulin"]>=16 and row["Insulin"]<=166:
        return "Normal"
    else:
        return "Abnormal"
```

```
df = df.assign(NewInsulinScore=df.apply(set_insuline, axis=1))
```

```
df.head()
```

```
# Some intervals were determined according to the glucose variable and these were assigned categorical variables.
NewGlucose = pd.Series(["Low", "Normal", "Overweight", "Secret", "High"], dtype = "category")
df["NewGlucose"] = NewGlucose
df.loc[df["Glucose"] <= 70, "NewGlucose"] = NewGlucose[0]
df.loc[(df["Glucose"] > 70) & (df["Glucose"] <= 99), "NewGlucose"] = NewGlucose[1]
df.loc[(df["Glucose"] > 99) & (df["Glucose"] <= 126), "NewGlucose"] = NewGlucose[2]
df.loc[df["Glucose"] > 126 ,"NewGlucose"] = NewGlucose[3]
```

```
df.head()
```

```
# One hot encoding
df = pd.get_dummies(df, columns = ["NewBMI", "NewInsulinScore", "NewGlucose"], drop_first=True)
```

```
df.head()
```

```
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome', 'NewBMI_Obesity 1',
       'NewBMI_Obesity 2', 'NewBMI_Obesity 3', 'NewBMI_Overweight',
       'NewBMI_Underweight', 'NewInsulinScore_Normal', 'NewGlucose_Low',
       'NewGlucose_Normal', 'NewGlucose_Overweight', 'NewGlucose_Secret'],
      dtype='object')
```

```
categorical_df = df[['NewBMI_Obesity 1',
       'NewBMI_Obesity 2', 'NewBMI_Obesity 3', 'NewBMI_Overweight',
       'NewBMI_Underweight', 'NewInsulinScore_Normal', 'NewGlucose_Low',
       'NewGlucose_Normal', 'NewGlucose_Overweight', 'NewGlucose_Secret']]
```

```
categorical_df.head()
```

```
y=df['Outcome']
X=df.drop(['Outcome','NewBMI_Obesity 1',
       'NewBMI_Obesity 2', 'NewBMI_Obesity 3', 'NewBMI_Overweight',
       'NewBMI_Underweight', 'NewInsulinScore_Normal', 'NewGlucose_Low',
       'NewGlucose_Normal', 'NewGlucose_Overweight', 'NewGlucose_Secret'], axis=1)


cols = X.columns
index = X.index


X.head()
```

```
from sklearn.preprocessing import RobustScaler
transformer = RobustScaler().fit(X)
X=transformer.transform(X)
X=pd.DataFrame(X, columns = cols, index = index)


X.head()
```

```
X = pd.concat([X, categorical_df], axis=1)


X.head()
```

```
X_train, X_test, y_train , y_test = train_test_split(X,y, test_size=0.2, random_state=0)


scaler =StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)


# Machine Learning Algo


# Logistic Regreesion


log_reg = LogisticRegression()
log_reg.fit(X_train, y_train)
```

```
y_pred = log_reg.predict(X_test)


accuracy_score(y_train, log_reg.predict(X_train))
```

```
0.8470394736842105
```

```
log_reg_acc = accuracy_score(y_test, log_reg.predict(X_test))


confusion_matrix(y_test, y_pred)
```

```
array([[88, 10],
       [ 6, 48]])
```

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

```
              precision    recall  f1-score   support

           0       0.94      0.90      0.92        98
           1       0.83      0.89      0.86        54

    accuracy                           0.89       152
   macro avg       0.88      0.89      0.89       152
```

```
weighted avg       0.90      0.89      0.90       152
```

```
# KNN
```

```
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print("Train Accuracy :", accuracy_score(y_train, knn.predict(X_train)))
knn_acc = accuracy_score(y_test, knn.predict(X_test))
print("Test Accuracy :", accuracy_score(y_test, knn.predict(X_test)))
print("Confusion matrix:", confusion_matrix(y_test, y_pred))
print("Classification report: ", classification_report(y_test, y_pred))
```

```
Train Accuracy : 0.875
Test Accuracy : 0.881578947368421
Confusion matrix: [[88 10]
 [ 8 46]]
Classification report:                precision    recall  f1-score   support

           0       0.92      0.90      0.91        98
           1       0.82      0.85      0.84        54

    accuracy                           0.88       152
   macro avg       0.87      0.87      0.87       152
weighted avg       0.88      0.88      0.88       152
```

```
# SVM
svc = SVC(probability=True)
parameter = {
    "gamma":[0.0001, 0.001, 0.01, 0.1],
    'C': [0.01, 0.05,0.5, 0.01, 1, 10, 15, 20]
}
grid_search = GridSearchCV(svc, parameter)
grid_search.fit(X_train, y_train)
```

```
# best_parameter
grid_search.best_params_
```

```
{'C': 10, 'gamma': 0.01}
```

```
grid_search.best_score_
```

```
np.float64(0.8618208914781196)
```

```
svc = SVC(C=10, gamma = 0.01, probability=True)
svc.fit(X_train, y_train)
y_pred = svc.predict(X_test)
print(accuracy_score(y_train, svc.predict(X_train)))
svc_acc = accuracy_score(y_test, svc.predict(X_test))
print(accuracy_score(y_test, svc.predict(X_test)))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
0.875
0.9078947368421053
[[90  8]
 [ 6 48]]
              precision    recall  f1-score   support

           0       0.94      0.92      0.93        98
           1       0.86      0.89      0.87        54

    accuracy                           0.91       152
   macro avg       0.90      0.90      0.90       152
weighted avg       0.91      0.91      0.91       152
```

```
# Decision Tree
```

```
DT = DecisionTreeClassifier()
DT.fit(X_train, y_train)
```

```
y_pred = DT.predict(X_test)
print(accuracy_score(y_train, DT.predict(X_train)))

print(accuracy_score(y_test, DT.predict(X_test)))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
1.0
0.8486842105263158
[[85 13]
 [10 44]]
              precision    recall  f1-score   support

           0       0.89      0.87      0.88        98
           1       0.77      0.81      0.79        54

    accuracy                           0.85       152
   macro avg       0.83      0.84      0.84       152
weighted avg       0.85      0.85      0.85       152
```

```
# hyperparameter tuning of dt
grid_param = {
    'criterion':['gini','entropy'],
    'max_depth' :  [3,5,7,10],
    'splitter' : ['best','radom'],
    'min_samples_leaf':[1,2,3,5,7],
    'min_samples_split':[1,2,3,5,7],
    'max_features':['auto','sqrt','log2']
}
grid_search_dt = GridSearchCV(DT, grid_param, cv=50, n_jobs=-1, verbose = 1)
grid_search_dt.fit(X_train, y_train)
```

```
grid_search_dt.best_params_
```

```
{'criterion': 'entropy',
 'max_depth': 7,
 'max_features': 'sqrt',
 'min_samples_leaf': 5,
 'min_samples_split': 3,
 'splitter': 'best'}
```

```
grid_search_dt.best_score_
```

```
np.float64(0.8743589743589745)
```

```
DT = grid_search_dt.best_estimator_
y_pred = DT.predict(X_test)
print("Train Accuracy:", accuracy_score(y_train, DT.predict(X_train)))
dt_acc = accuracy_score(y_test, DT.predict(X_test))
print("Test Accuracy:", accuracy_score(y_test, DT.predict(X_test)))
print("Confusion Matrix:",confusion_matrix(y_test, y_pred))
print("Claffication Report:", classification_report(y_test, y_pred))
```

```
Train Accuracy: 0.9095394736842105
Test Accuracy: 0.8947368421052632
Confusion Matrix: [[92  6]
 [10 44]]
Claffication Report:               precision    recall  f1-score   support

           0       0.90      0.94      0.92        98
           1       0.88      0.81      0.85        54

    accuracy                           0.89       152
   macro avg       0.89      0.88      0.88       152
weighted avg       0.89      0.89      0.89       152
```

```
rand_clf = RandomForestClassifier(criterion = 'entropy', max_depth = 15, max_features = 0.75, min_samples_leaf = 2, min_samples_split = 3, n
rand_clf.fit(X_train, y_train)
```

```
y_pred = rand_clf.predict(X_test)
```

```
y_pred = rand_clf.predict(X_test)
print(accuracy_score(y_train, rand_clf.predict(X_train)))
rand_acc = accuracy_score(y_test, rand_clf.predict(X_test))
print(accuracy_score(y_test, rand_clf.predict(X_test)))
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
0.9950657894736842
0.8947368421052632
[[89  9]
 [ 7 47]]
              precision    recall  f1-score   support

           0       0.93      0.91      0.92        98
           1       0.84      0.87      0.85        54

    accuracy                           0.89       152
   macro avg       0.88      0.89      0.89       152
weighted avg       0.90      0.89      0.90       152
```

```
gbc = GradientBoostingClassifier()

parameters = {
    'loss': ['deviance', 'exponential'],
    'learning_rate': [0.001, 0.1, 1, 10],
    'n_estimators': [100, 150, 180, 200]
}

grid_search_gbc = GridSearchCV(gbc, parameters, cv = 10, n_jobs = -1, verbose = 1)
grid_search_gbc.fit(X_train, y_train)
```

```
grid_search_gbc.best_params_
```

```
{'learning_rate': 0.1, 'loss': 'exponential', 'n_estimators': 150}
```

```
grid_search_gbc.best_score_
```

```
np.float64(0.888032786885246)
```

```
gbc = GradientBoostingClassifier(learning_rate = 0.1, loss = 'exponential', n_estimators = 150)
gbc.fit(X_train, y_train)
```

```
gbc = grid_search_gbc.best_estimator_
y_pred = gbc.predict(X_test)
print("Train Accuracy :", accuracy_score(y_train, gbc.predict(X_train)))
gbc_acc = accuracy_score(y_test, gbc.predict(X_test))
print("Test Accuracy:", accuracy_score(y_test, gbc.predict(X_test)))
print("Confusion Matrix:", confusion_matrix(y_test, y_pred))
print("Classification Report:", classification_report(y_test, y_pred))
```

```
Train Accuracy : 0.9983552631578947
Test Accuracy: 0.9144736842105263
Confusion Matrix: [[91  7]
 [ 6 48]]
Classification Report:               precision    recall  f1-score   support

           0       0.94      0.93      0.93        98
           1       0.87      0.89      0.88        54

    accuracy                           0.91       152
   macro avg       0.91      0.91      0.91       152
weighted avg       0.91      0.91      0.91       152
```

```
from xgboost import XGBClassifier
xgb = XGBClassifier(objective = 'binary:logistic', learning_rate = 0.01, max_depth = 10, n_estimators = 180)

xgb.fit(X_train, y_train)
```

```
y_pred = xgb.predict(X_test)
print("Train Accuracy :", accuracy_score(y_train, xgb.predict(X_train)))
xgb_acc = accuracy_score(y_test, xgb.predict(X_test))
print("Test Accuracy: ",accuracy_score(y_test, xgb.predict(X_test)))
print("Confusion Matrix:", confusion_matrix(y_test, y_pred))
print("Classification Report:", classification_report(y_test, y_pred))
```

```
Train Accuracy : 0.9736842105263158
Test Accuracy:  0.881578947368421
Confusion Matrix: [[89  9]
 [ 9 45]]
Classification Report:               precision    recall  f1-score   support

           0       0.91      0.91      0.91        98
           1       0.83      0.83      0.83        54

    accuracy                           0.88       152
   macro avg       0.87      0.87      0.87       152
weighted avg       0.88      0.88      0.88       152
```

```
# Model Comparison
models = pd.DataFrame({
    'Model': ['Logistic Regression', 'KNN', 'SVM', 'Decision Tree Classifier', 'Random Forest Classifier', 'Gradient Boosting Classifier', '
    'Score': [100*round(log_reg_acc,4), 100*round(knn_acc,4), 100*round(svc_acc,4), 100*round(dt_acc,4), 100*round(rand_acc,4),
              100*round(gbc_acc,4), 100*round(xgb_acc,4)]
})
models.sort_values(by = 'Score', ascending = False)
```

```
import pickle
model = gbc
pickle.dump(model, open("diabetes.pkl",'wb'))
```

```
from sklearn import metrics
plt.figure(figsize=(8,5))
models = [
{
    'label': 'LR',
    'model': log_reg,
},
{
    'label': 'DT',
    'model': DT,
},
{
    'label': 'SVM',
    'model': svc,
},
{
    'label': 'KNN',
    'model': knn,
},
{
    'label': 'XGBoost',
    'model': xgb,
},
{
    'label': 'RF',
    'model': rand_clf,
},
{
    'label': 'GBDT',
    'model': gbc,
}
]
for m in models:
    model = m['model']
    model.fit(X_train, y_train)
    y_pred=model.predict(X_test)
    fpr1, tpr1, thresholds = metrics.roc_curve(y_test, model.predict_proba(X_test)[:,1])
    auc = metrics.roc_auc_score(y_test,model.predict(X_test))
    plt.plot(fpr1, tpr1, label='%s - ROC (area = %0.2f)' % (m['label'], auc))

plt.plot([0, 1], [0, 1],'r--')
```

```
plt.xlim([-0.01, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('1 - Specificity (False Positive Rate)', fontsize=12)
plt.ylabel('Sensitivity (True Positive Rate)', fontsize=12)
plt.title('ROC - Diabetes Prediction', fontsize=12)
plt.legend(loc="lower right", fontsize=12)
plt.savefig("roc_diabetes.jpeg", format='jpeg', dpi=400, bbox_inches='tight')
plt.show()
```

```
from sklearn import metrics
import numpy as np
import matplotlib.pyplot as plt
models = [
{
    'label': 'LR',
    'model': log_reg,
},
{
    'label': 'DT',
    'model': DT,
},
{
    'label': 'SVM',
    'model': svc,
},
{
    'label': 'KNN',
    'model': knn,
},
{
    'label': 'XGBoost',
    'model': xgb,
},
{
    'label': 'RF',
    'model': rand_clf,
},
{
    'label': 'GBDT',
    'model': gbc,
}
]

means_roc = []
means_accuracy = [100*round(log_reg_acc,4), 100*round(dt_acc,4), 100*round(svc_acc,4), 100*round(knn_acc,4), 100*round(xgb_acc,4),
                  100*round(rand_acc,4), 100*round(gbc_acc,4)]

for m in models:
    model = m['model']
    model.fit(X_train, y_train)
    y_pred=model.predict(X_test)
    fpr1, tpr1, thresholds = metrics.roc_curve(y_test, model.predict_proba(X_test)[:,1])
    auc = metrics.roc_auc_score(y_test,model.predict(X_test))
    auc = 100*round(auc,4)
    means_roc.append(auc)

print(means_accuracy)
print(means_roc)

# data to plot
n_groups = 7
means_accuracy = tuple(means_accuracy)
means_roc = tuple(means_roc)

# create plot
fig, ax = plt.subplots(figsize=(8,5))
index = np.arange(n_groups)
bar_width = 0.35
opacity = 0.8

rects1 = plt.bar(index, means_accuracy, bar_width,
alpha=opacity,
color='mediumpurple',
label='Accuracy (%)')
```

```
rects2 = plt.bar(index + bar_width, means_roc, bar_width,
alpha=opacity,
color='rebeccapurple',
label='ROC (%)')

plt.xlim([-1, 8])
plt.ylim([60, 95])

plt.title('Performance Evaluation - Diabetes Prediction', fontsize=12)
plt.xticks(index, ('   LR', '   DT', '   SVM', '   KNN', 'XGBoost' , '   RF', '   GBDT'), rotation=40, ha='center', fontsize=12)
plt.legend(loc="upper right", fontsize=10)
plt.savefig("PE_diabetes.jpeg", format='jpeg', dpi=400, bbox_inches='tight')
plt.show()
```