```
import numpy as np #linear algebra library of Python
import pandas as pd # build on top of numpy for data analysis, data manipulation and d
import matplotlib.pyplot as plt #plotting library of Python
```

Now let's mount Google drive so that we can upload the diabetes.csv file. You can find the code in the 'Code s

```
from google.colab import drive
drive.mount('/content/gdrive')
```


First thing that we do is take a look at the shape of the dataframe (df.shape) and take a look at first 5 lines th

```
df=pd.read_csv('/content/gdrive/My Drive/Colab Notebooks/creditcard.csv') #import file
df.head() #shows first 5 lines including column namesdf.shape # number of rows and col
```



```
df.shape
```


Let's create numpy arrays, one for the features (X) and one for the label (y)

```
X=df.drop('Class', 1).values #drop 'Outcome' column but you keep the index column
y=df['Class'].values
```

```
from sklearn.model_selection import train_test_split #method to split training and tes
X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.2, random_state=42
```

```
pip install bayesian-optimization
```

```
import xgboost as xgb
from xgboost.sklearn import XGBClassifier
import bayes_opt
from bayes_opt import BayesianOptimization
from sklearn.model_selection import cross_val_score
```

```
pbounds = {'n_estimators': (50, 1000), 'eta': (0.01, 3), 'max_depth': (1,32), 'gamma':
model_tuning = XGBClassifier(n_jobs=-1)

def xgboostcv(eta, n_estimators, max_depth, gamma, min_child_weight, subsample, colsam
    return np.mean(cross_val_score(model_tuning, X_train, y_train, cv=5, scoring='accu

optimizer = BayesianOptimization(
    f=xgboostcv,
    pbounds=pbounds,
    random_state=1)
```

```
optimizer.maximize(
    init_points=2,
    n_iter=3)
print(optimizer.max)
```

```
model = XGBClassifier(eta=2, n_estimators=138, max_depth=10, min_child_weight=5, gamma
model.fit(X_train, y_train)
```

```
from sklearn import metrics
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve

y_pred_proba=model.predict_proba(X_test)[:,1]
fpr, tpr, thresholds=roc_curve(y_test, y_pred_proba)
print(roc_auc_score(y_test, y_pred_proba)) # ROC score

plt.plot([0,1], [0,1], 'k--')
plt.plot(fpr, tpr, label='RF')
plt.xlabel('fpr')
```

```
plt.ylabel('tpr')
plt.title('ROC curve')
plt.show()
```