```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from collections import Counter
from PIL import Image
import os
from os import path
```

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

```
Mounted at /content/drive
```

```
result=pd.read_csv('/content/drive/My Drive/GSCM/BankChurners_PCA.csv')
```

```
from imblearn.over_sampling import SMOTE
```

```
# Split target & features
X = result.drop('Attrition_Flag', axis=1)
y = result['Attrition_Flag']
```

```
# Dealing with imbalanced dataset
from imblearn.over_sampling import SMOTE

# Upsampling with SMOTE algorithm
smote = SMOTE(random_state=66)

X_smote, y_smote = smote.fit_resample(X, y)

print(f'''Shape of X before SMOTE: {X.shape}
Shape of X after SMOTE: {X_smote.shape}''')

print('\nBalance of positive and negative classes (%):')
y_smote.value_counts(normalize=True) * 100
```

```
Shape of X before SMOTE: (10127, 11)
Shape of X after SMOTE: (17000, 11)

Balance of positive and negative classes (%):
0    50.0
1    50.0
Name: Attrition_Flag, dtype: float64
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_smote, y_smote, train_size=0.8, random_state=69)
```

## Visualize Model Score

```
from sklearn.ensemble import RandomForestClassifier ,AdaBoostClassifier,BaggingClassifier,ExtraTreesClassifier,GradientBoostingClassifier
from sklearn.model_selection import GridSearchCV, cross_val_score, StratifiedKFold, learning_curve ,KFold
from sklearn.metrics import roc_curve,accuracy_score,f1_score,auc,confusion_matrix,roc_auc_score,plot_confusion_matrix
from xgboost.sklearn import XGBClassifier
from sklearn.metrics import classification_report,confusion_matrix,accuracy_score
from sklearn.tree import DecisionTreeClassifier
```

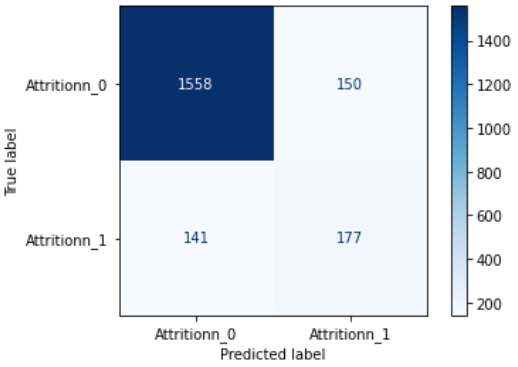
## DecisionTreeClassifier

```
dt = DecisionTreeClassifier(random_state = 0)
dt.fit(X_train,y_train)
print("Accuracy: %.2f%%" % ((dt.score(X_test,y_test))*100.0))
```

```
Accuracy: 85.64%
```

```
plot_confusion_matrix(dt,
                                      X_test,  y_test,
                                      cmap=plt.cm.Blues,
                                      display_labels  =  ['Attritionn_0','Attritionn_1'])
plt.grid(False)
plt.show();
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is de
  warnings.warn(msg, category=FutureWarning)
```




## GradientBoostingClassifier

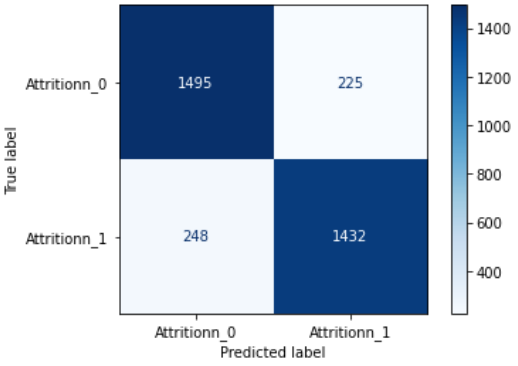
```
from  sklearn.ensemble  import  GradientBoostingClassifier

gb  =  GradientBoostingClassifier(random_state  =  0)
gb.fit(X_train,y_train)
print("Accuracy:  %.2f%%"  %  ((gb.score(X_test,y_test))*100.0))
```

```
Accuracy: 86.09%
```

```
plot_confusion_matrix(gb,
                                      X_test,  y_test,
                                      cmap=plt.cm.Blues,
                                      display_labels  =  ['Attritionn_0','Attritionn_1'])
plt.grid(False)
plt.show();
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is de
  warnings.warn(msg, category=FutureWarning)
```




## Random Forest

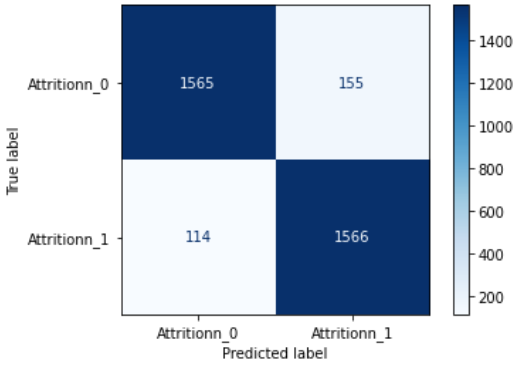
```
from  sklearn.ensemble  import  RandomForestClassifier

RFC  =  RandomForestClassifier(random_state  =  0)
RFC.fit(X_train,y_train)
print("Accuracy:  %.2f%%"  %  ((RFC.score(X_test,y_test))*100.0))
```

```
Accuracy: 92.09%
```

```
plot_confusion_matrix(RFC,
                                      X_test,  y_test,
                                      cmap=plt.cm.Blues,
                                      display_labels  =  ['Attritionn_0','Attritionn_1'])
plt.grid(False)
plt.show();
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is de
  warnings.warn(msg, category=FutureWarning)
```

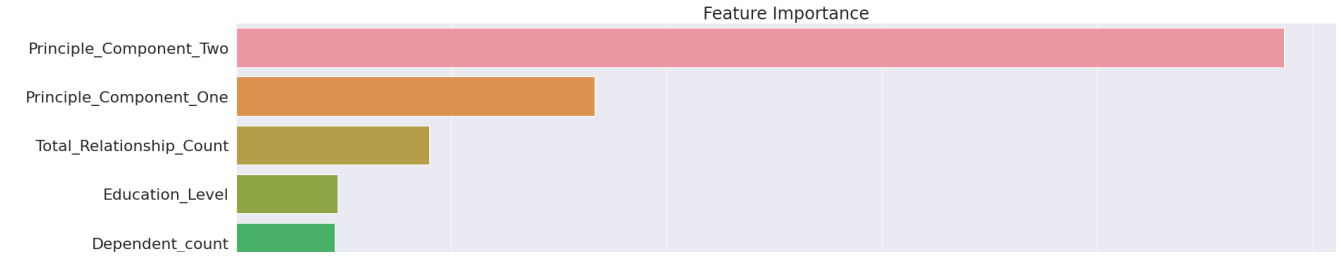



```
feature  =  pd.Series(RFC.feature_importances_,  index  =  X_train.columns).sort_values(ascending  =  False)
print(feature)
```

```
Principle_Component_Two     0.486513
Principle_Component_One     0.166600
Total_Relationship_Count    0.089794
Education_Level             0.047248
Dependent_count             0.045809
Income_Category             0.041943
Customer_Age_cat            0.032230
Marital_Status              0.029365
Months_on_book_cat          0.026667
Gender                      0.025580
Card_Category               0.008251
dtype: float64
```

```
import  seaborn  as  sns
plt.figure(figsize  =  (28,14))
sns.set(font_scale=2)
sns.barplot(x  =  feature,  y  =  feature.index)
plt.title("Feature  Importance")

plt.xlabel('Score')
plt.ylabel('Features')
plt.show()
```




## adaboost

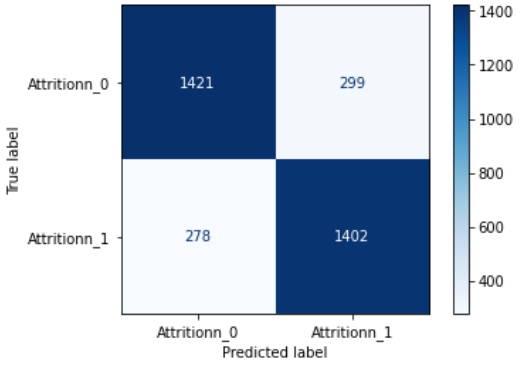
```
from sklearn.ensemble import AdaBoostClassifier

ada = AdaBoostClassifier(random_state = 0)
ada.fit(X_train,y_train)
print("Accuracy: %.2f%%" % ((ada.score(X_test,y_test))*100.0))
```

```
Accuracy: 83.03%
```

```
plot_confusion_matrix(ada,
                                        X_test, y_test,
                                        cmap=plt.cm.Blues,
                                        display_labels = ['Attritionn_0','Attritionn_1'])
plt.grid(False)
plt.show();
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is de
  warnings.warn(msg, category=FutureWarning)
```

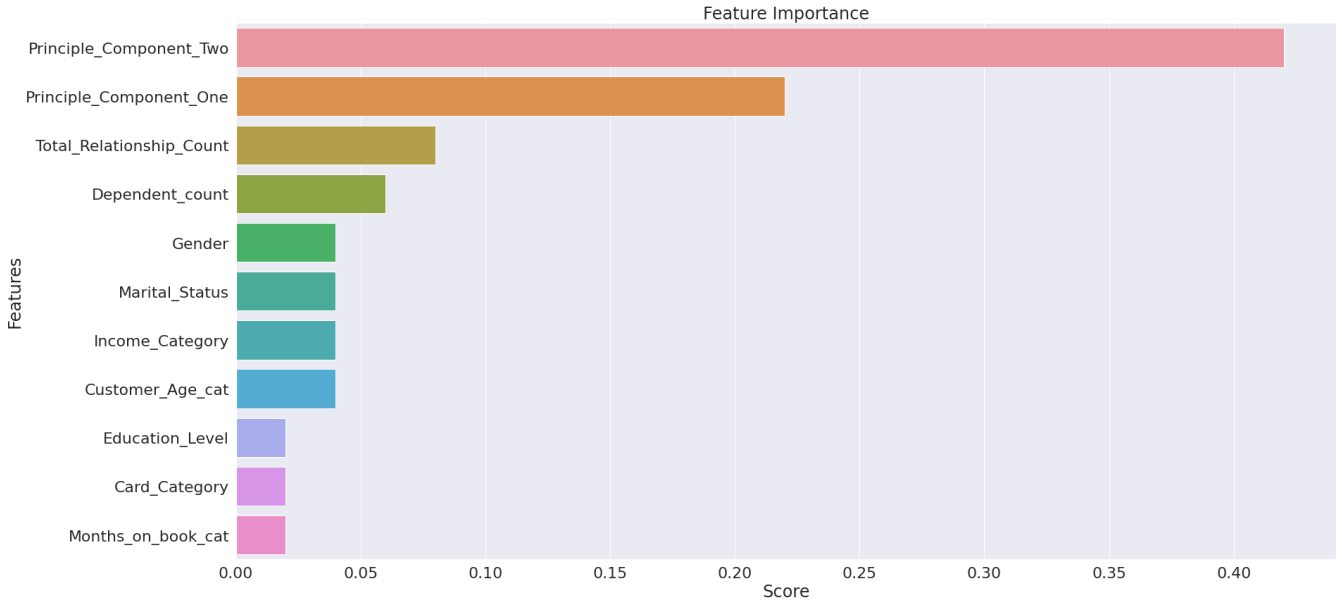



```
feature = pd.Series(ada.feature_importances_, index = X_train.columns).sort_values(ascending = False)
print(feature)
```

```
Principle_Component_Two     0.42
Principle_Component_One     0.22
Total_Relationship_Count    0.08
Dependent_count             0.06
Gender                      0.04
Marital_Status              0.04
Income_Category             0.04
Customer_Age_cat            0.04
Education_Level             0.02
Card_Category               0.02
Months_on_book_cat          0.02
dtype: float64
```

```
import seaborn as sns
plt.figure(figsize = (28,14))
sns.set(font_scale=2)
sns.barplot(x = feature, y = feature.index)
plt.title("Feature Importance")

plt.xlabel('Score')
plt.ylabel('Features')
plt.show()
```



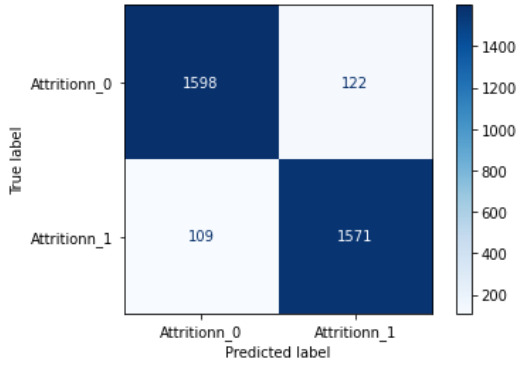

## ExtraTree Classifier

```
extraT = ExtraTreesClassifier(random_state = 0)
extraT.fit(X_train,y_train)
print("Accuracy: %.2f%%" % ((extraT.score(X_test,y_test))*100.0))
```

```
Accuracy: 93.21%
```

```
plot_confusion_matrix(extraT,
                                        X_test, y_test,
                                        cmap=plt.cm.Blues,
                                        display_labels = ['Attritionn_0','Attritionn_1'])
plt.grid(False)
plt.show();
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_confusion_matrix is de
  warnings.warn(msg, category=FutureWarning)
```




## Feature Importance

```
feature = pd.Series(extraT.feature_importances_, index = X_train.columns).sort_values(ascending = False)
print(feature)
```

```
Principle_Component_Two      0.408123
Principle_Component_One      0.129841
```

```
Total_Relationship_Count    0.107655
Education_Level             0.067629
Dependent_count             0.065919
Income_Category             0.057044
Marital_Status              0.044225
Customer_Age_cat            0.043823
Months_on_book_cat          0.036693
Gender                      0.025425
Card_Category               0.013623
dtype: float64
```

```
plt.figure(figsize  =  (28,14))
sns.set(font_scale=2)
sns.barplot(x  =  feature,  y  =  feature.index)
plt.title("Feature  Importance")

plt.xlabel('Score')
plt.ylabel('Features')
plt.show()
```