01 About the project

Ensembled Learning model to predict a person diabetic or not.

02 Models used

1. Logistic Regression
2. KNN
3. Decision Tree
4. Random Forest
5. SVM

03 Model Intuitions and their strengths

04 Project goals

Final Ensembled model should predict correctly and should give desired metrics.

Ensembled Learning

Ensemble learning is a machine learning technique that involves combining multiple models to improve the overall performance and accuracy of a prediction or classification task

Stacking

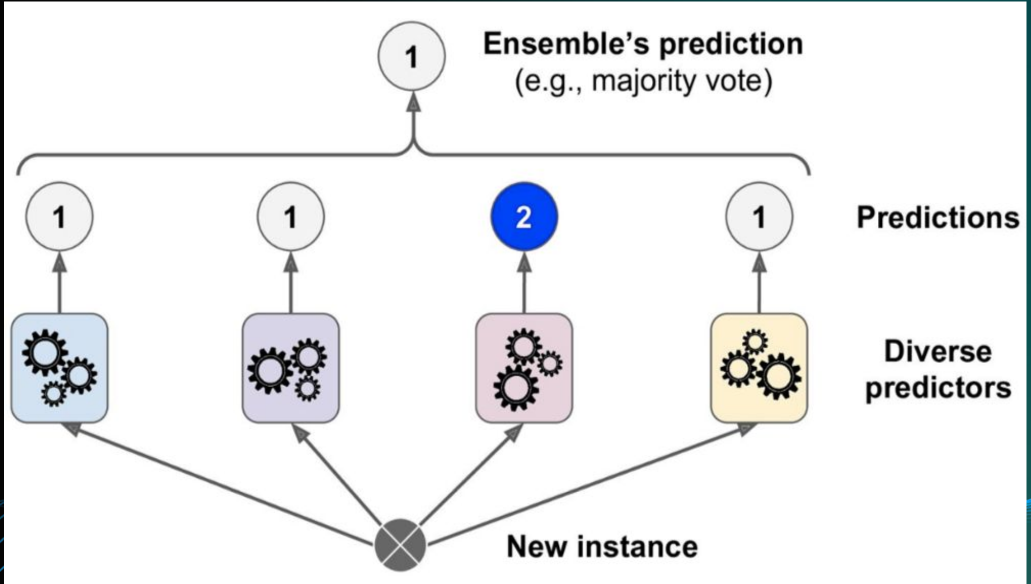


Figure 7-2. Hard voting classifier predictions

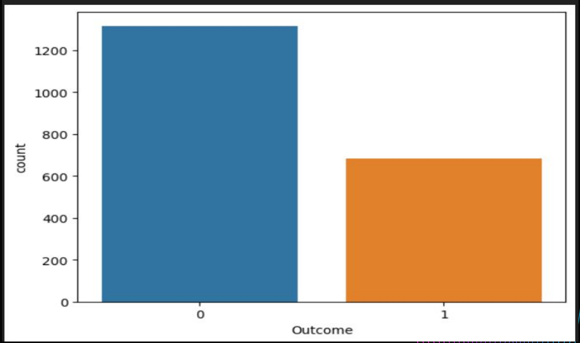
Problem statement

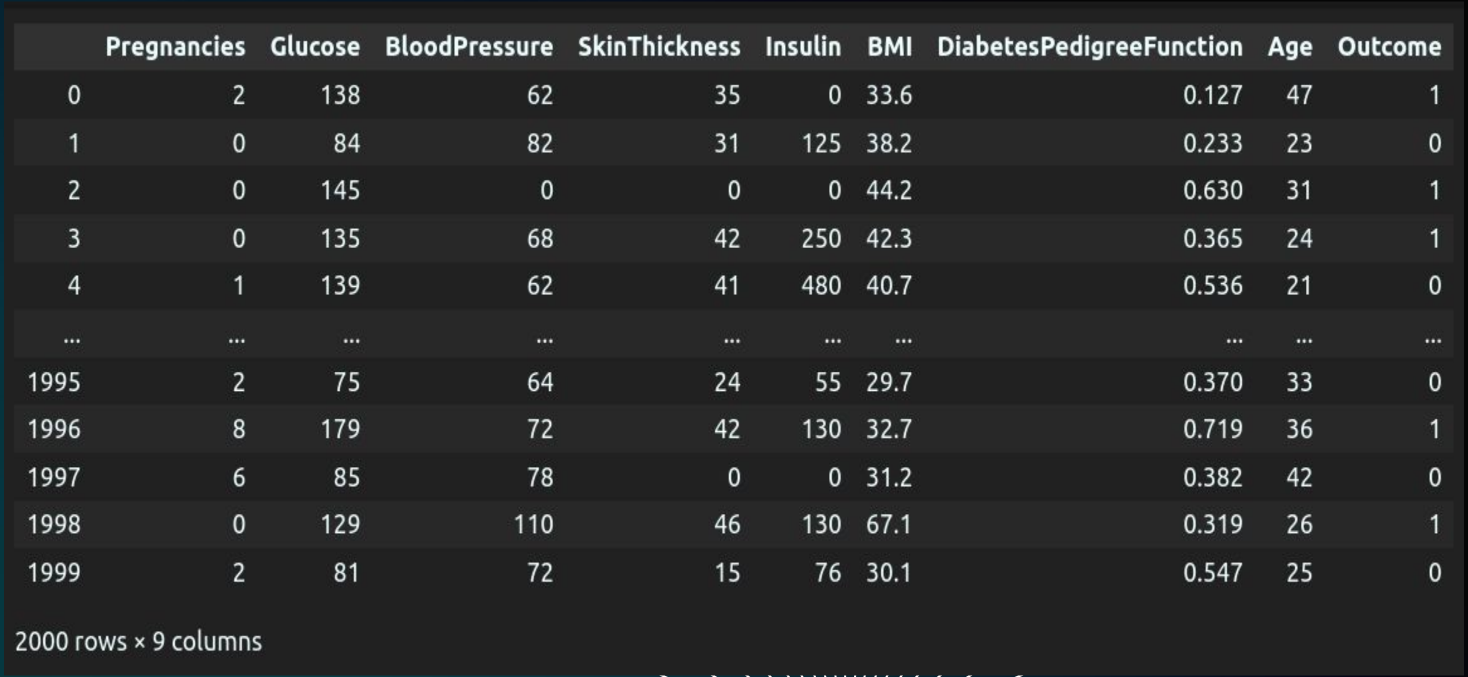
To detect Person has Diabetes or Not based on the following features in the dataset.

- No. of pregnancies
- Glucose
- Blood Pressure
- Skin Thickness
- Insulin
- Body Mass Index (BMI)
- Diabetes Pedigree Function
- Age.

If the person has diabetes output 1 otherwise 0.

Data Preprocessing

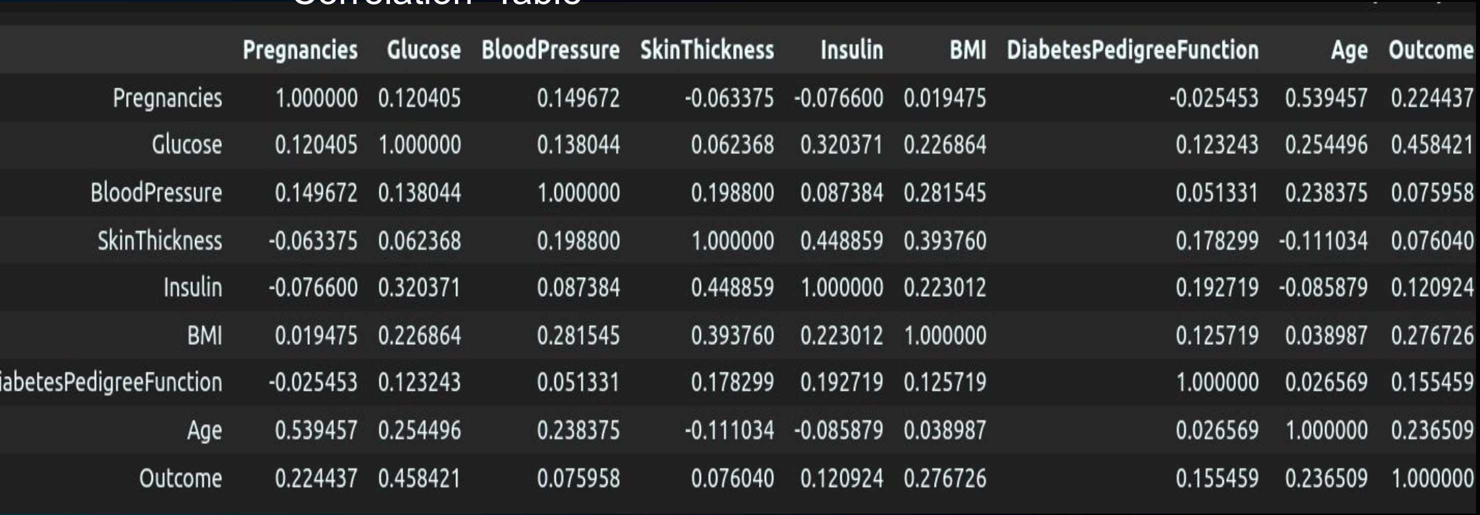




	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	2	138	62	35	0	33.6	0.127	47	1
1	0	84	82	31	125	38.2	0.233	23	0
2	0	145	0	0	0	44.2	0.630	31	1
3	0	135	68	42	250	42.3	0.365	24	1
4	1	139	62	41	480	40.7	0.536	21	0
...	...	...	...	...	...	...	...	...	...
1995	2	75	64	24	55	29.7	0.370	33	0
1996	8	179	72	42	130	32.7	0.719	36	1
1997	6	85	78	0	0	31.2	0.382	42	0
1998	0	129	110	46	130	67.1	0.319	26	1
1999	2	81	72	15	76	30.1	0.547	25	0

2000 rows × 9 columns

Correlation Table

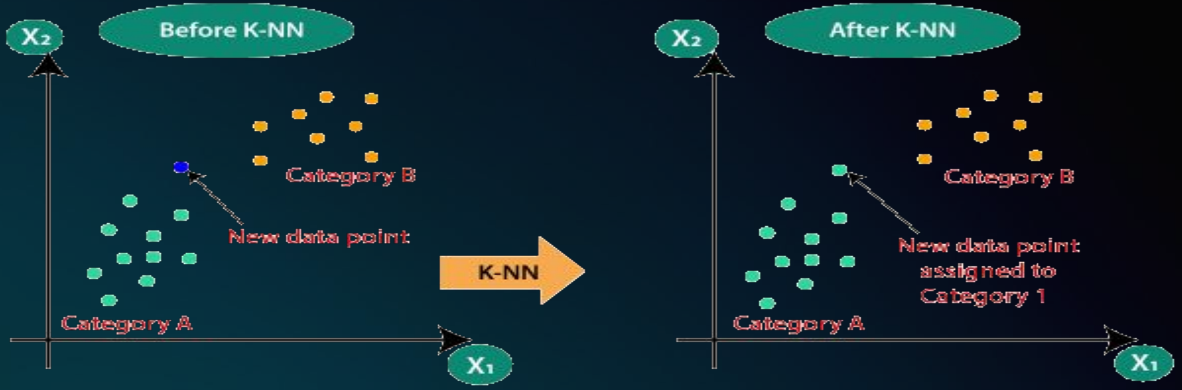


	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.120405	0.149672	-0.063375	-0.076600	0.019475	-0.025453	0.539457	0.224437
Glucose	0.120405	1.000000	0.138044	0.062368	0.320371	0.226864	0.123243	0.254496	0.458421
BloodPressure	0.149672	0.138044	1.000000	0.198800	0.087384	0.281545	0.051331	0.238375	0.075958
SkinThickness	-0.063375	0.062368	0.198800	1.000000	0.448859	0.393760	0.178299	-0.111034	0.076040
Insulin	-0.076600	0.320371	0.087384	0.448859	1.000000	0.223012	0.192719	-0.085879	0.120924
BMI	0.019475	0.226864	0.281545	0.393760	0.223012	1.000000	0.125719	0.038987	0.276726
iabetesPedigreeFunction	-0.025453	0.123243	0.051331	0.178299	0.192719	0.125719	1.000000	0.026569	0.155459
Age	0.539457	0.254496	0.238375	-0.111034	-0.085879	0.038987	0.026569	1.000000	0.236509
Outcome	0.224437	0.458421	0.075958	0.076040	0.120924	0.276726	0.155459	0.236509	1.000000

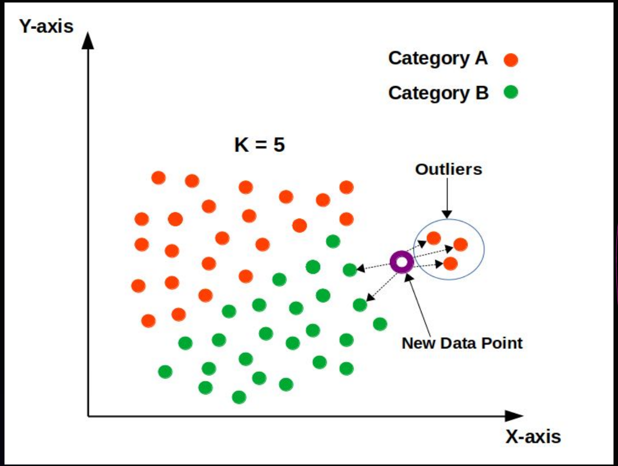
Logistic Regression

This algorithm comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

KNN model



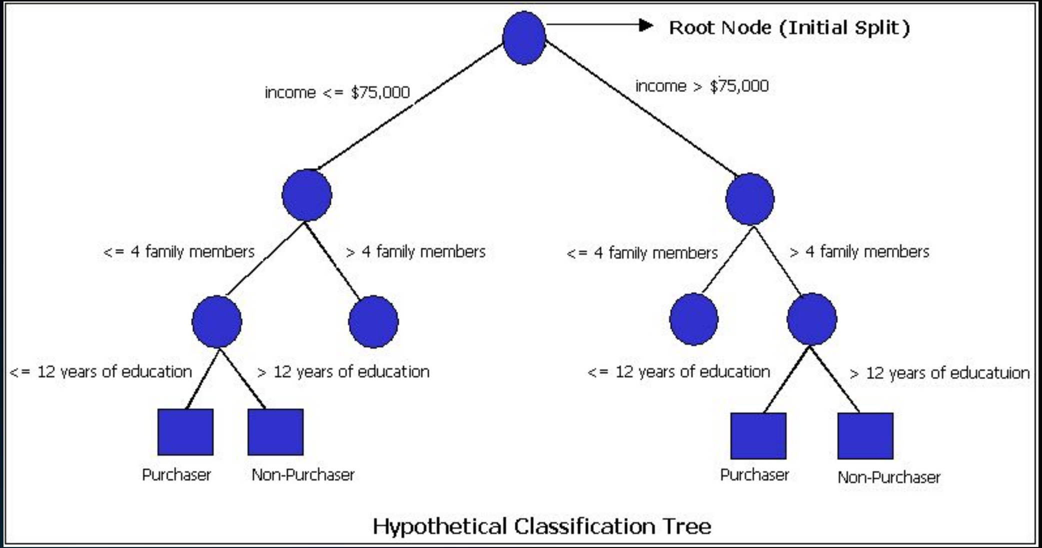
Before K-NN
X₂
X₁
Category B
New data point
Category A
K-NN
After K-NN
X₂
X₁
Category B
New data point assigned to Category 1
Category A



Y-axis
Category A ●
Category B ●
K = 5
Outliers
New Data Point
X-axis

Decision Trees

- Graphical Representation of all the possible solutions to a decision.
- Decisions are based on conditions.
- Compared to other algorithms decision trees requires less effort for data preparation during preprocessing.
- A Decision tree model is very intuitive

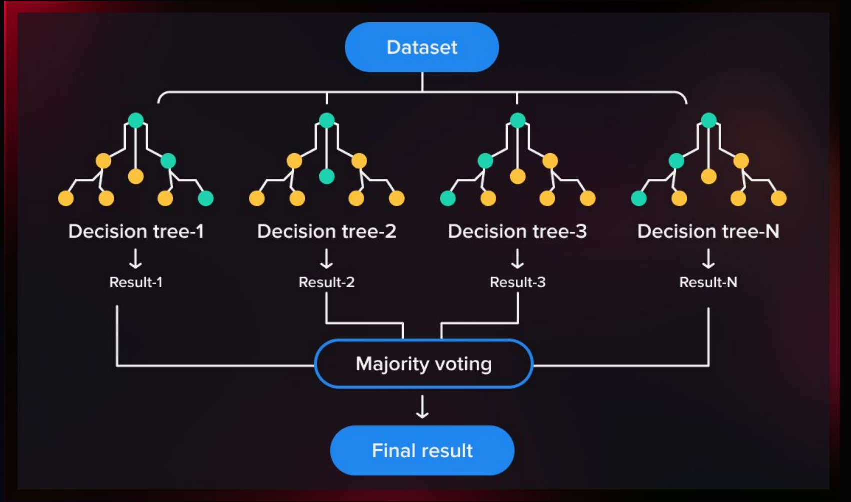


Random Forest

- Builds with multiple decision trees and merges them together.
- More accurate and stable prediction.
- Solves the problem of Overfitting.

Trained with the “Bagging” method.

Bagging

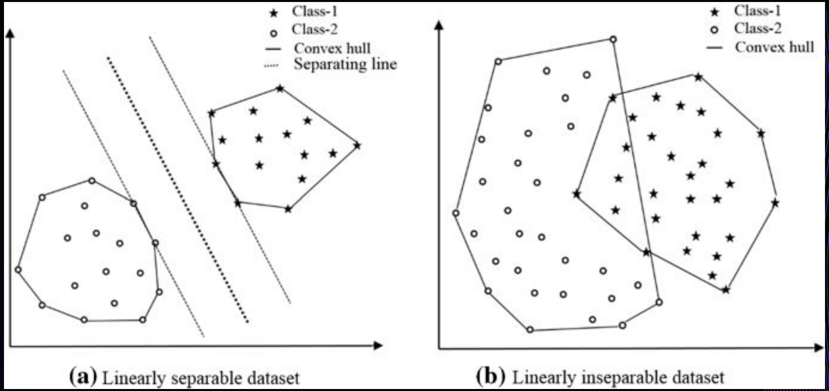


Bagging, short for "Bootstrap Aggregating," is a machine learning ensemble technique that involves training multiple models on different subsets of the training data, and then combining their predictions to produce a final output. The idea behind bagging is to reduce the variance of the model and improve its generalization performance.

SVM

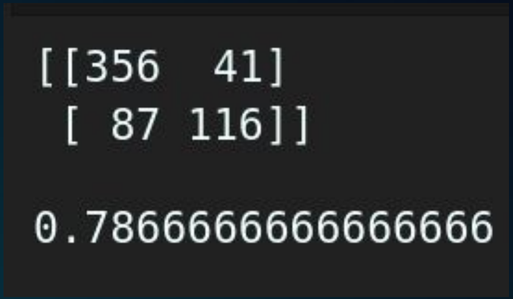
Works well with exact Linearly separable data. It performs well for dataset with higher dimensions also.

SVM does not perform very well when the data set has more noise i.e. target classes are overlapping



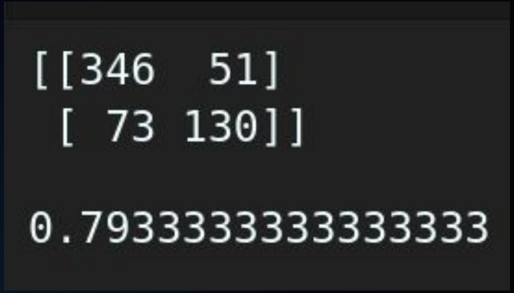
Confusion matrix

Logistic Regression



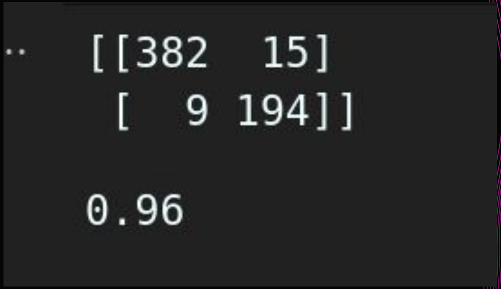
```
[[356  41]
 [ 87 116]]

0.7866666666666666
```

KNN



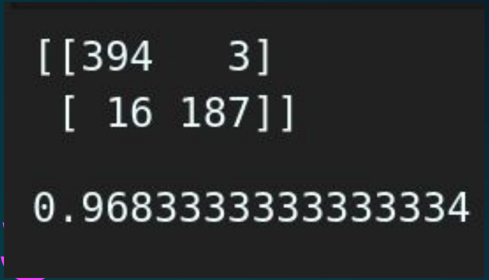
```
[[346  51]
 [ 73 130]]

0.7933333333333333
```

Decision Tree



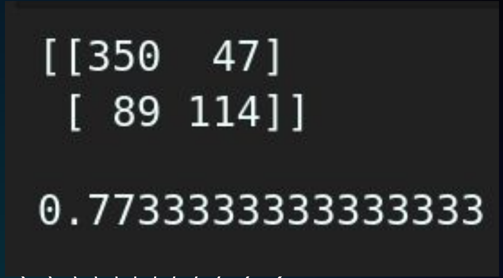
```
[[382  15]
 [  9 194]]

0.96
```

Random Forest

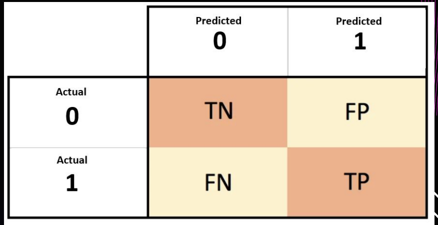


```
[[394   3]
 [ 16 187]]

0.9683333333333334
```

SVM

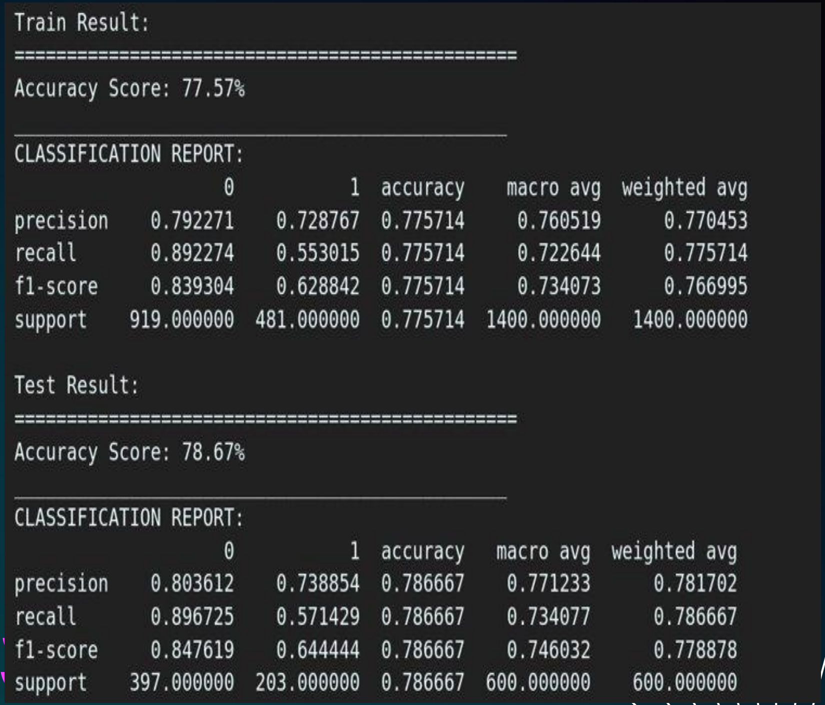


```
[[350  47]
 [ 89 114]]

0.7733333333333333
```



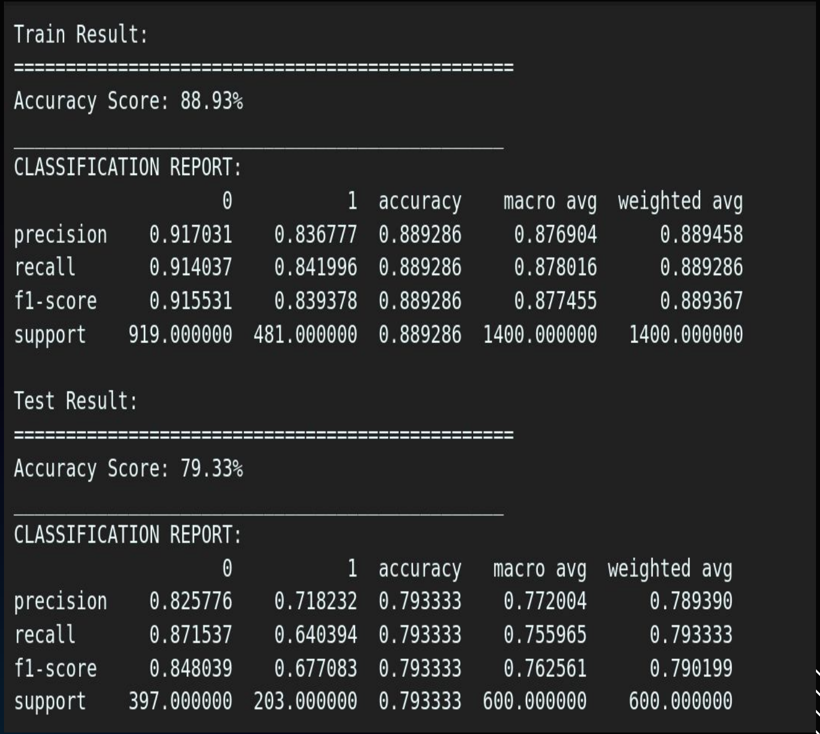
	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

Logistic regression



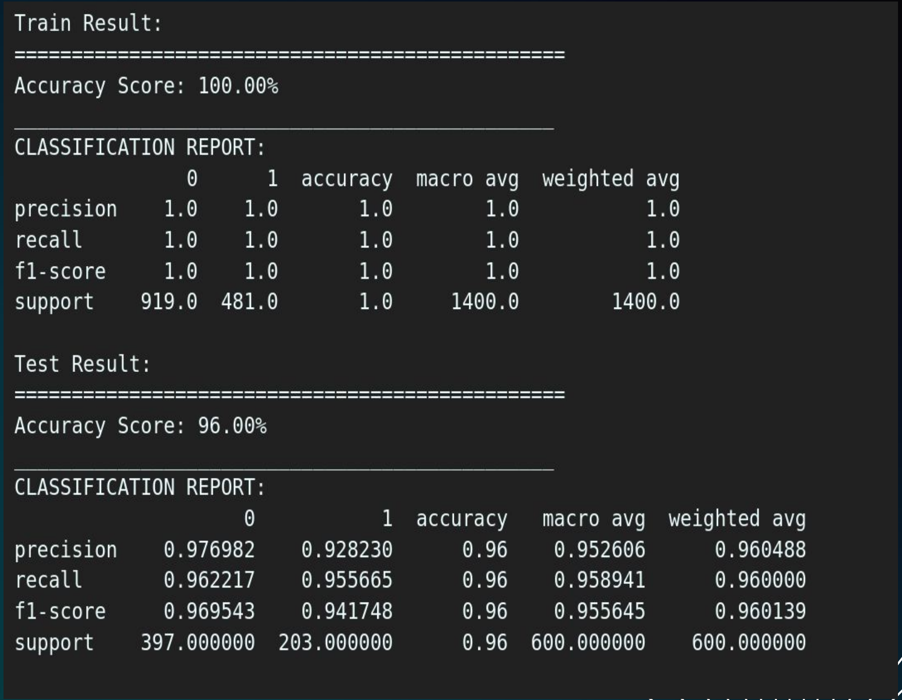
```
Train Result:
================================================
Accuracy Score: 77.57%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy    macro avg  weighted avg
precision    0.792271    0.728767  0.775714     0.760519      0.770453
recall       0.892274    0.553015  0.775714     0.722644      0.775714
f1-score     0.839304    0.628842  0.775714     0.734073      0.766995
support    919.000000  481.000000  0.775714  1400.000000   1400.000000

Test Result:
================================================
Accuracy Score: 78.67%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy   macro avg  weighted avg
precision    0.803612    0.738854  0.786667    0.771233      0.781702
recall       0.896725    0.571429  0.786667    0.734077      0.786667
f1-score     0.847619    0.644444  0.786667    0.746032      0.778878
support    397.000000  203.000000  0.786667  600.000000    600.000000
```

KNN



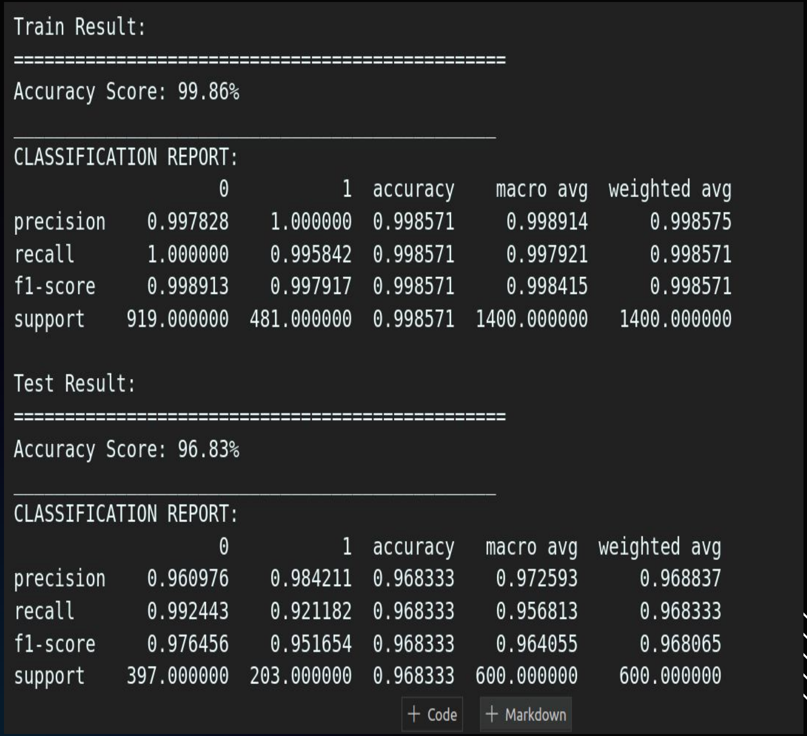
```
Train Result:
================================================
Accuracy Score: 88.93%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy    macro avg  weighted avg
precision    0.917031    0.836777  0.889286     0.876904      0.889458
recall       0.914037    0.841996  0.889286     0.878016      0.889286
f1-score     0.915531    0.839378  0.889286     0.877455      0.889367
support    919.000000  481.000000  0.889286  1400.000000   1400.000000

Test Result:
================================================
Accuracy Score: 79.33%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy   macro avg  weighted avg
precision    0.825776    0.718232  0.793333    0.772004      0.789390
recall       0.871537    0.640394  0.793333    0.755965      0.793333
f1-score     0.848039    0.677083  0.793333    0.762561      0.790199
support    397.000000  203.000000  0.793333  600.000000    600.000000
```

Decision Tree



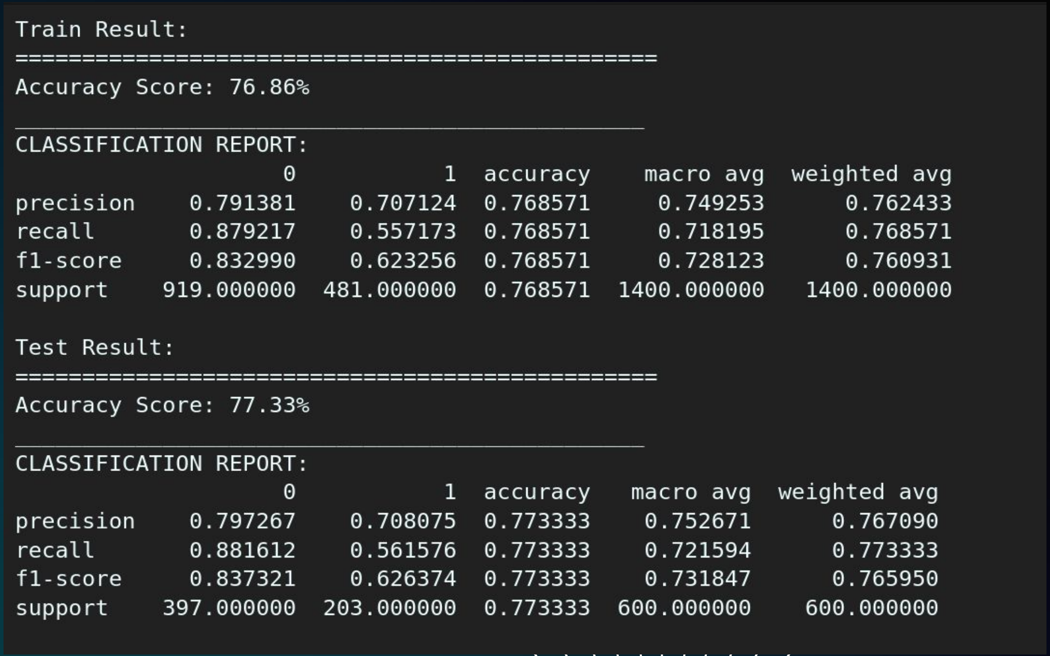
```
Train Result:
================================================
Accuracy Score: 100.00%
_______________________________________________
CLASSIFICATION REPORT:
               0      1  accuracy  macro avg  weighted avg
precision    1.0    1.0       1.0        1.0           1.0
recall       1.0    1.0       1.0        1.0           1.0
f1-score     1.0    1.0       1.0        1.0           1.0
support    919.0  481.0       1.0     1400.0        1400.0

Test Result:
================================================
Accuracy Score: 96.00%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy   macro avg  weighted avg
precision    0.976982    0.928230      0.96    0.952606      0.960488
recall       0.962217    0.955665      0.96    0.958941      0.960000
f1-score     0.969543    0.941748      0.96    0.955645      0.960139
support    397.000000  203.000000      0.96  600.000000    600.000000
```

Random Forest



```
Train Result:
================================================
Accuracy Score: 99.86%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy    macro avg  weighted avg
precision    0.997828    1.000000  0.998571     0.998914      0.998575
recall       1.000000    0.995842  0.998571     0.997921      0.998571
f1-score     0.998913    0.997917  0.998571     0.998415      0.998571
support    919.000000  481.000000  0.998571  1400.000000   1400.000000

Test Result:
================================================
Accuracy Score: 96.83%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy   macro avg  weighted avg
precision    0.960976    0.984211  0.968333    0.972593      0.968837
recall       0.992443    0.921182  0.968333    0.956813      0.968333
f1-score     0.976456    0.951654  0.968333    0.964055      0.968065
support    397.000000  203.000000  0.968333  600.000000    600.000000
```

SVM



```
Train Result:
================================================
Accuracy Score: 76.86%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy    macro avg  weighted avg
precision    0.791381    0.707124  0.768571     0.749253      0.762433
recall       0.879217    0.557173  0.768571     0.718195      0.768571
f1-score     0.832990    0.623256  0.768571     0.728123      0.760931
support    919.000000  481.000000  0.768571  1400.000000   1400.000000

Test Result:
================================================
Accuracy Score: 77.33%
_______________________________________________
CLASSIFICATION REPORT:
                    0           1  accuracy   macro avg  weighted avg
precision    0.797267    0.708075  0.773333    0.752671      0.767090
recall       0.881612    0.561576  0.773333    0.721594      0.773333
f1-score     0.837321    0.626374  0.773333    0.731847      0.765950
support    397.000000  203.000000  0.773333  600.000000    600.000000
```

Thank you