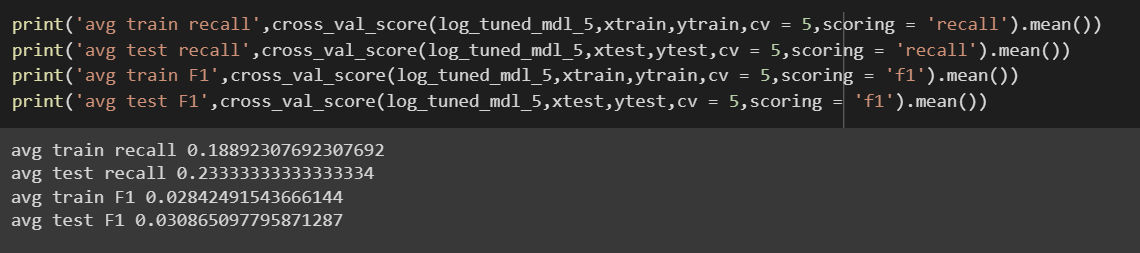
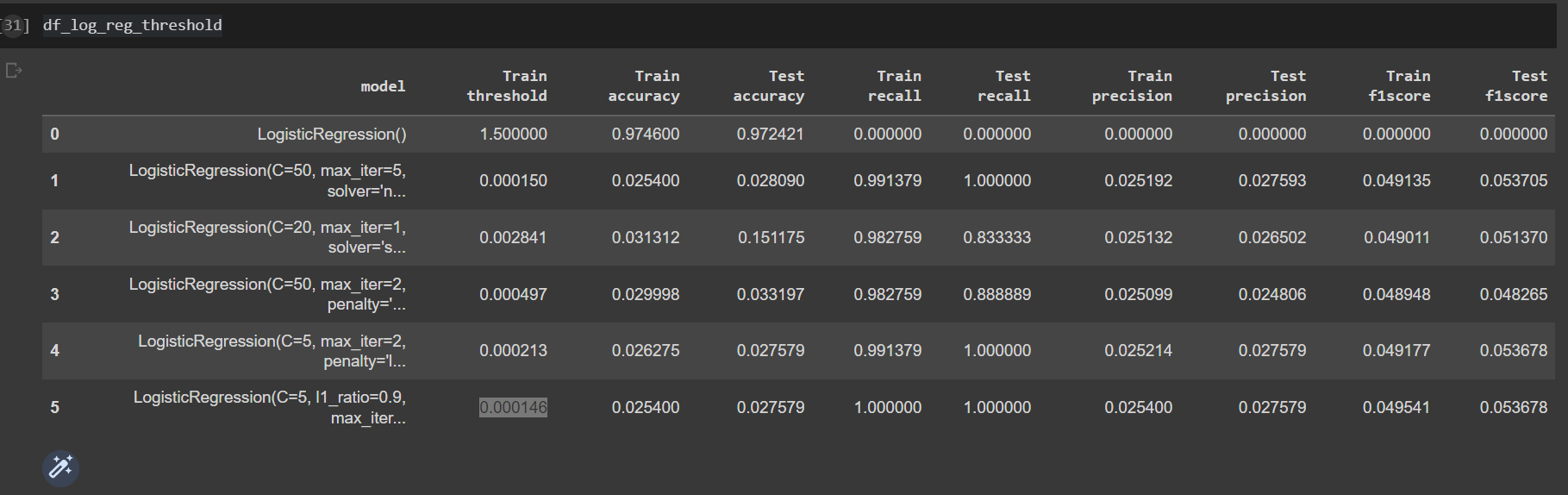
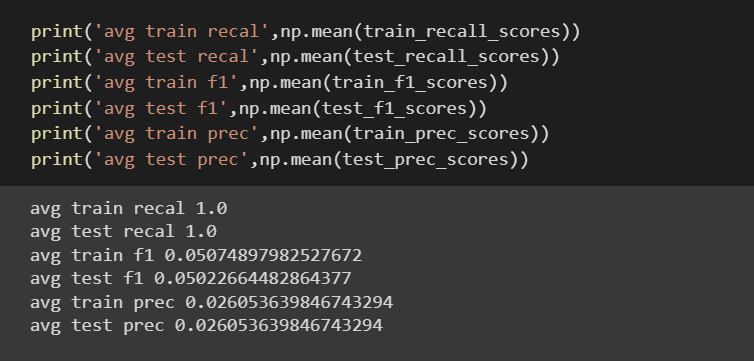
# Logistic Regression

|  |  |
| --- | --- |
| **Model** | **Tuned Parameters** |
| Log Model 1 | LogisticRegression(penalty='l2',max\_iter=5,C = 50,solver = 'newton-cg') |
| Log Model 2 | LogisticRegression(penalty='l2',max\_iter=1,C = 20,solver = 'sag') |
| Log Model 3 | LogisticRegression(penalty='none',max\_iter=2,C = 50,solver = 'sag') |
| Log Model 4 | LogisticRegression(penalty='l1',max\_iter=2,C = 5,solver = 'saga') |
| Log Model 5 | LogisticRegression(penalty='elasticnet',max\_iter=2,C = 5,solver = 'saga',l1\_ratio = 0.9) |



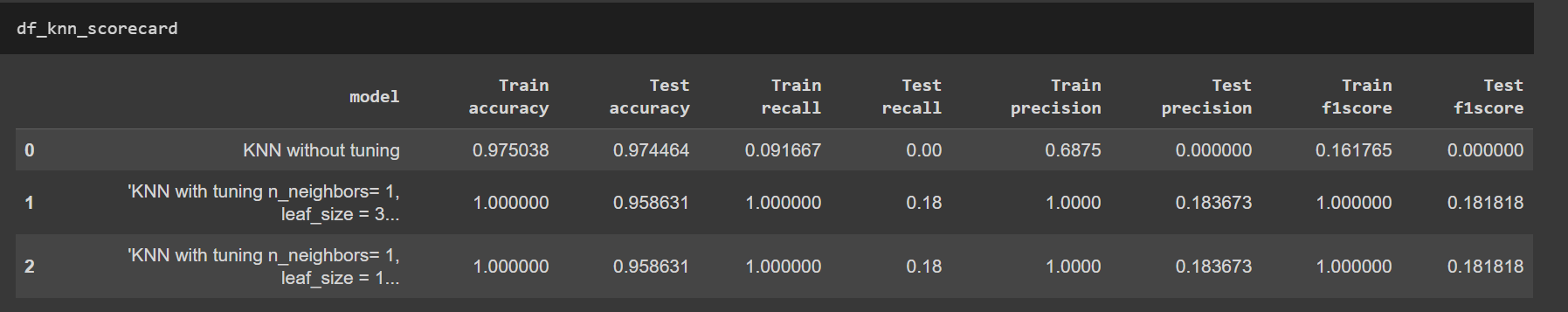


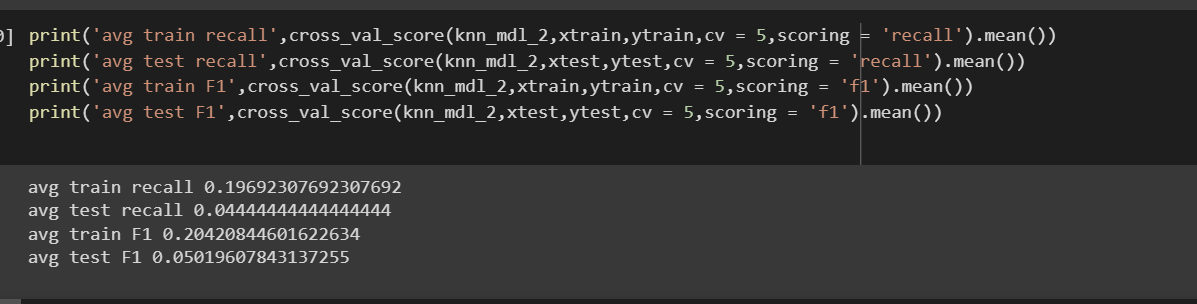


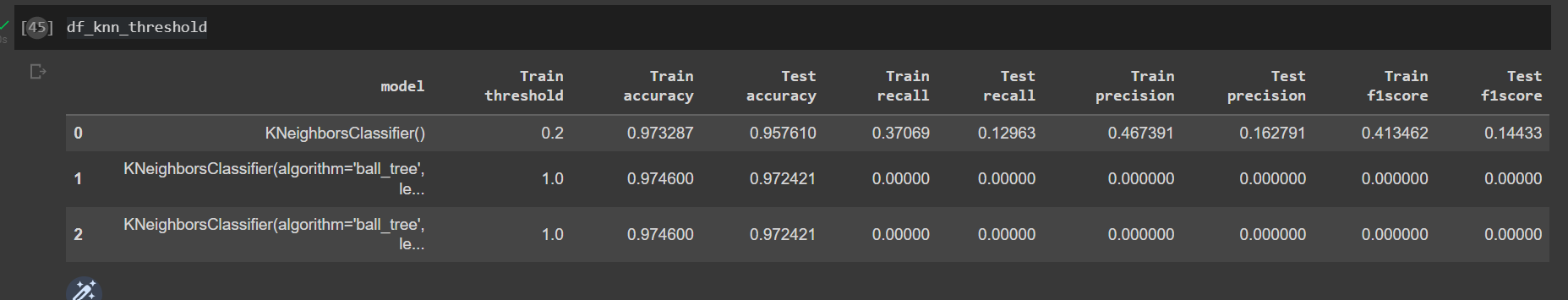


# KNN

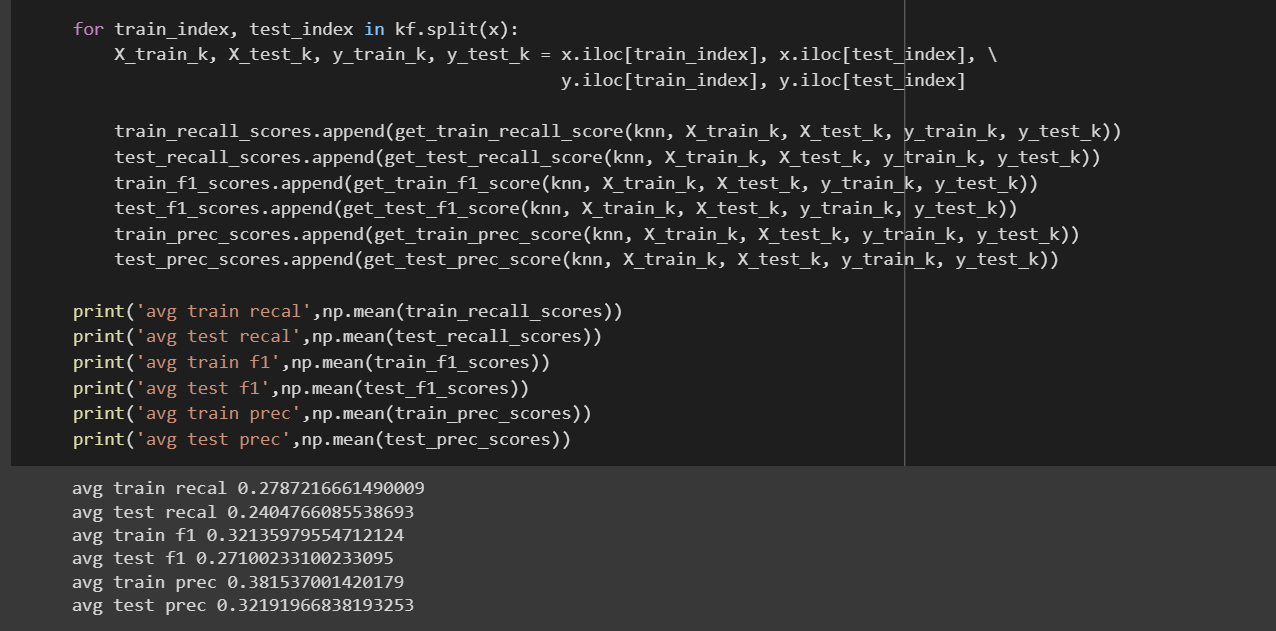
|  |  |
| --- | --- |
| **Model** | **Tuned Parameters** |
| KNN Model 1 | KNeighborsClassifier(n\_neighbors= 1, leaf\_size = 3, algorithm = 'ball\_tree', p =2 ,weights = 'uniform') |
| KNN Model 2 | KNeighborsClassifier(n\_neighbors= 1, leaf\_size = 1, algorithm = 'ball\_tree', p =2 ,weights = 'uniform') |





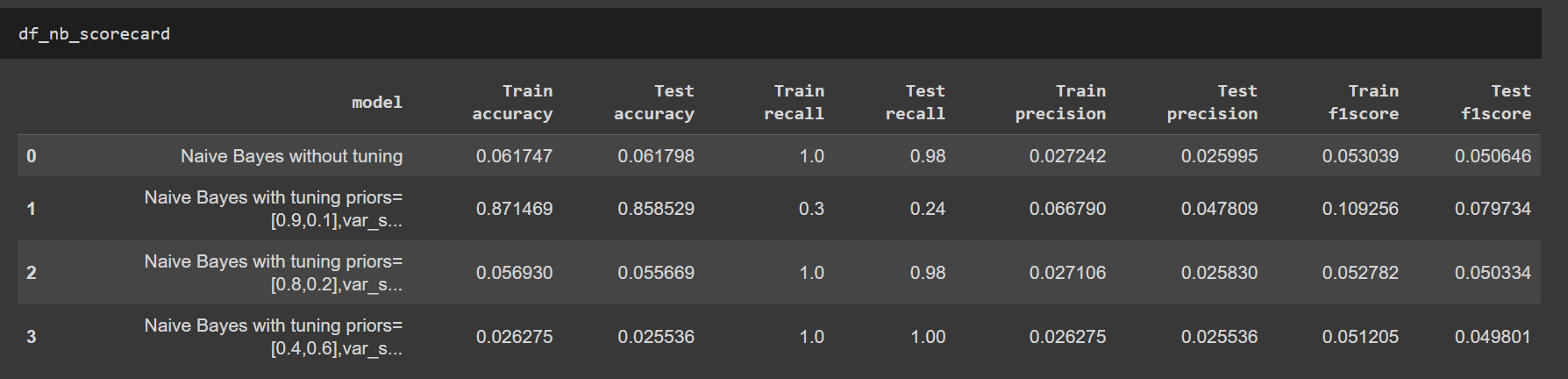


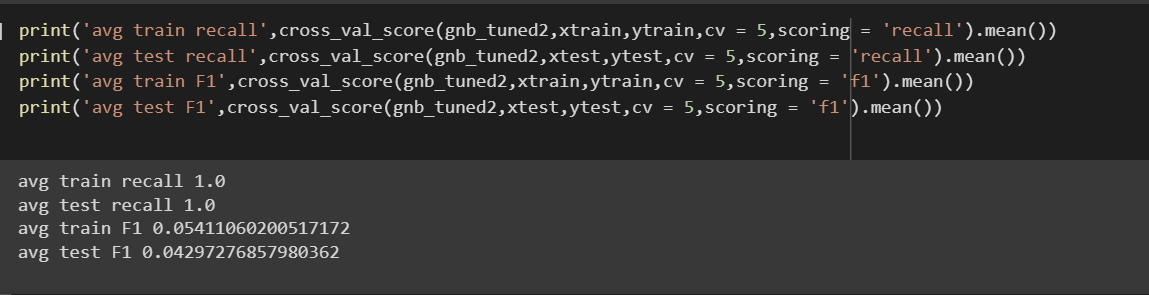
Threshold is checked for KNN without tuning model



# Naïve Bayes

|  |  |
| --- | --- |
| **Model** | **Tuned Parameters** |
| Naïve Bayes Model 1 | GaussianNB(priors=[0.9,0.1],var\_smoothing= 1e-05) |
| Naïve Bayes Model 2 | GaussianNB(priors=[0.8,0.2],var\_smoothing= 1e-07) |
| Naïve Bayes Model 3 | GaussianNB(priors=[0.4,0.6],var\_smoothing= 1) |

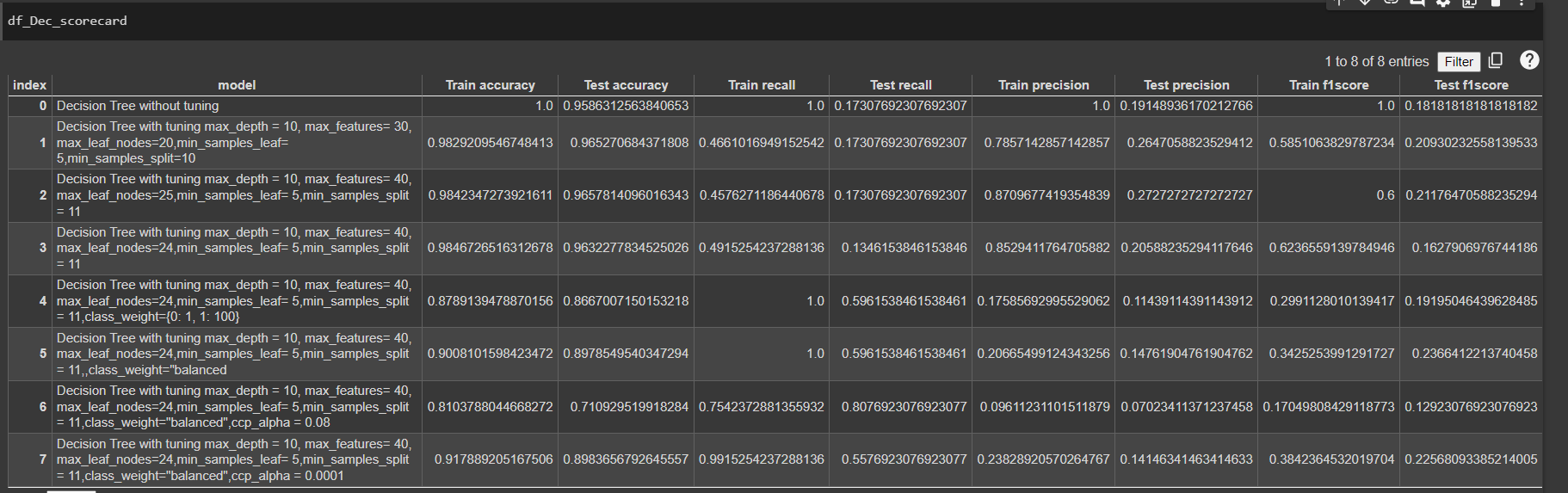


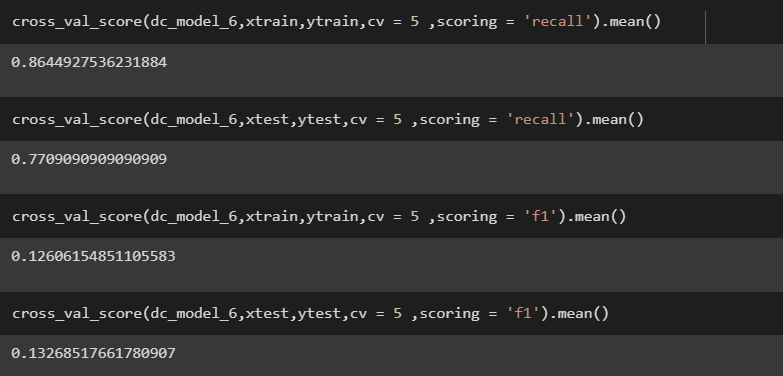


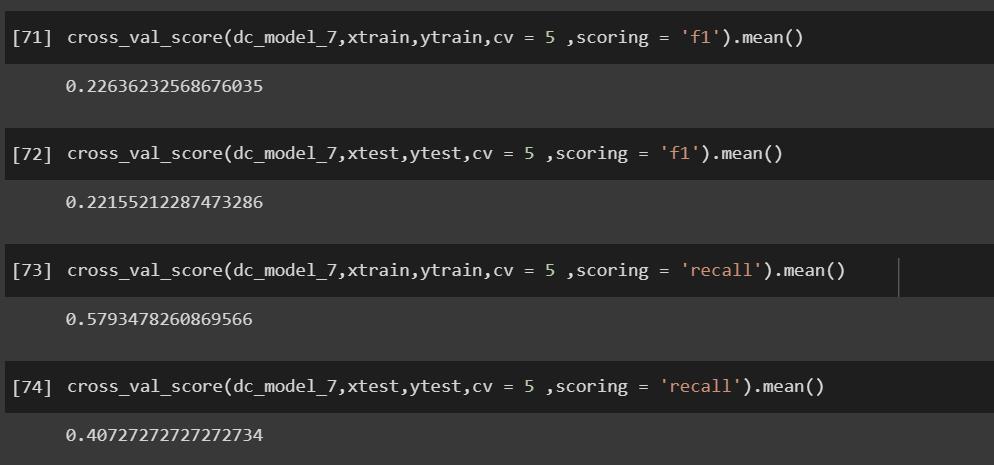
Did not check for Threshold here as recall is very high already

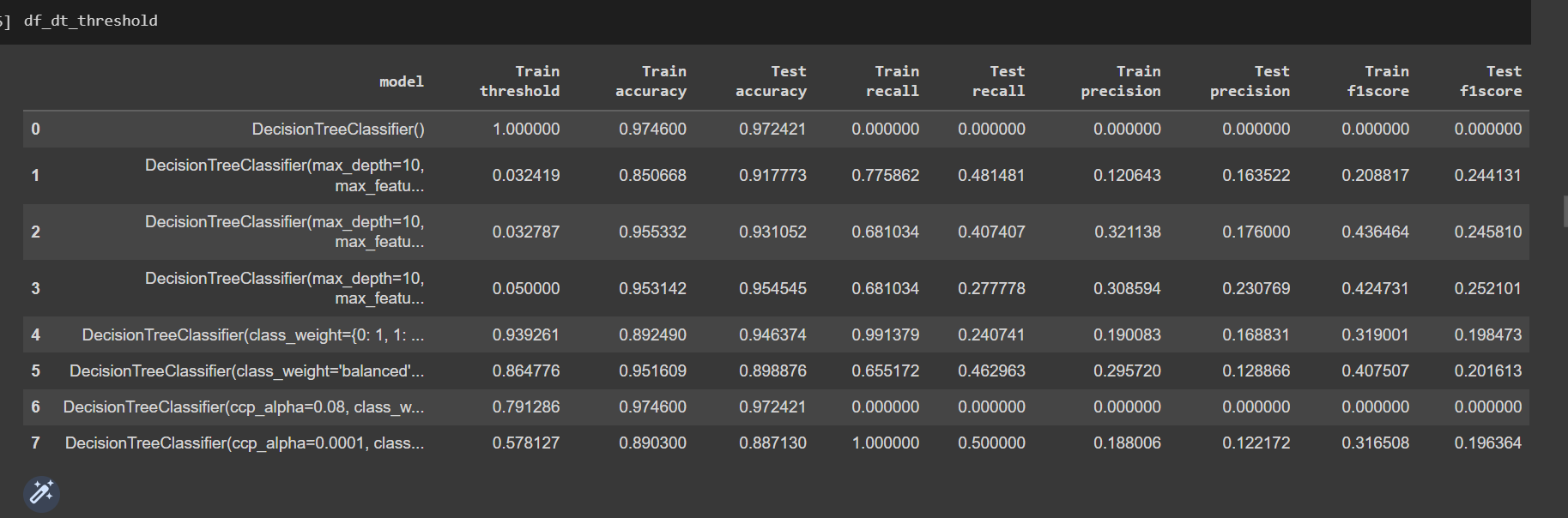
# Decision Tree

|  |  |
| --- | --- |
| **Model** | **Tuned Parameters** |
| DT Model 1 | DecisionTreeClassifier(max\_depth = 10, max\_features= 30, max\_leaf\_nodes=20,min\_samples\_leaf= 5,min\_samples\_split=10) |
| DT Model 2 | DecisionTreeClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=25,min\_samples\_leaf= 5,min\_samples\_split = 11) |
| DT Model 3 | DecisionTreeClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11) |
| DT Model 4 | DecisionTreeClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight={0: 1, 1: 100}) |
| DT Model 5 | DecisionTreeClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced") |
| DT Model 6 | DecisionTreeClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.08) |
| DT Model 7 | DecisionTreeClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.0001) |

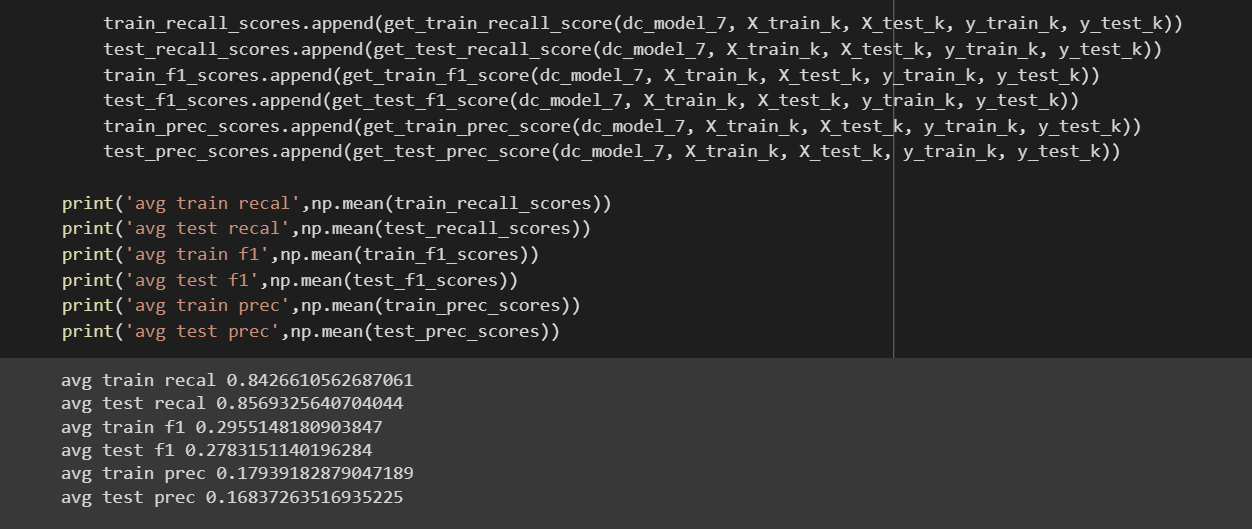






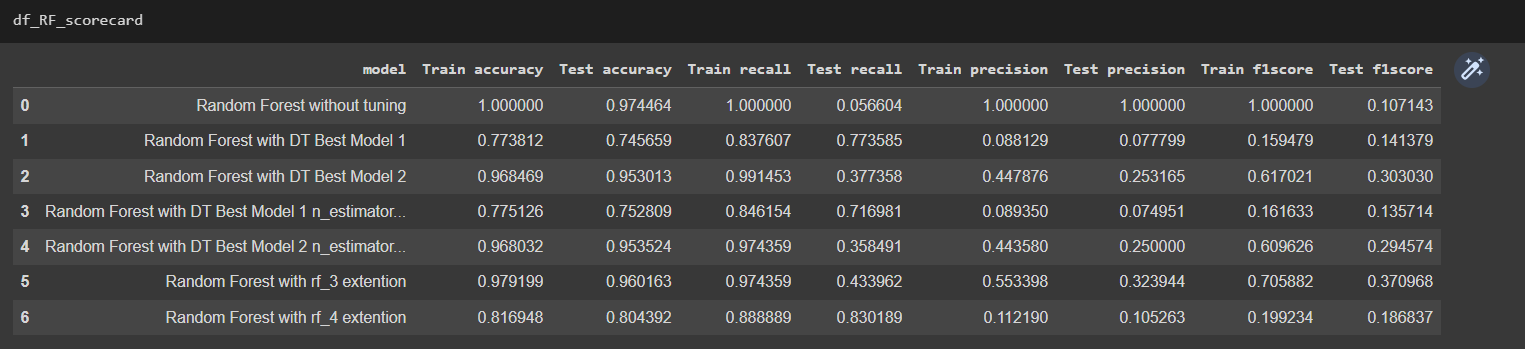


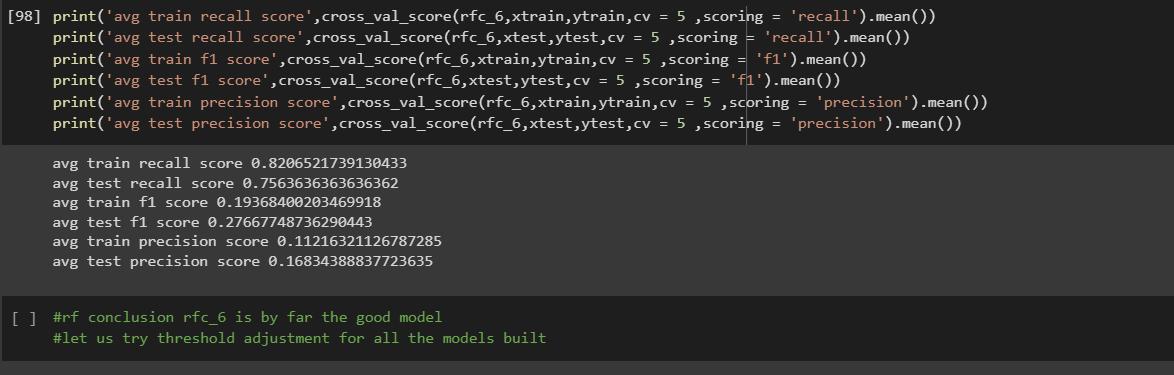
After threshold change model 7 worked better.

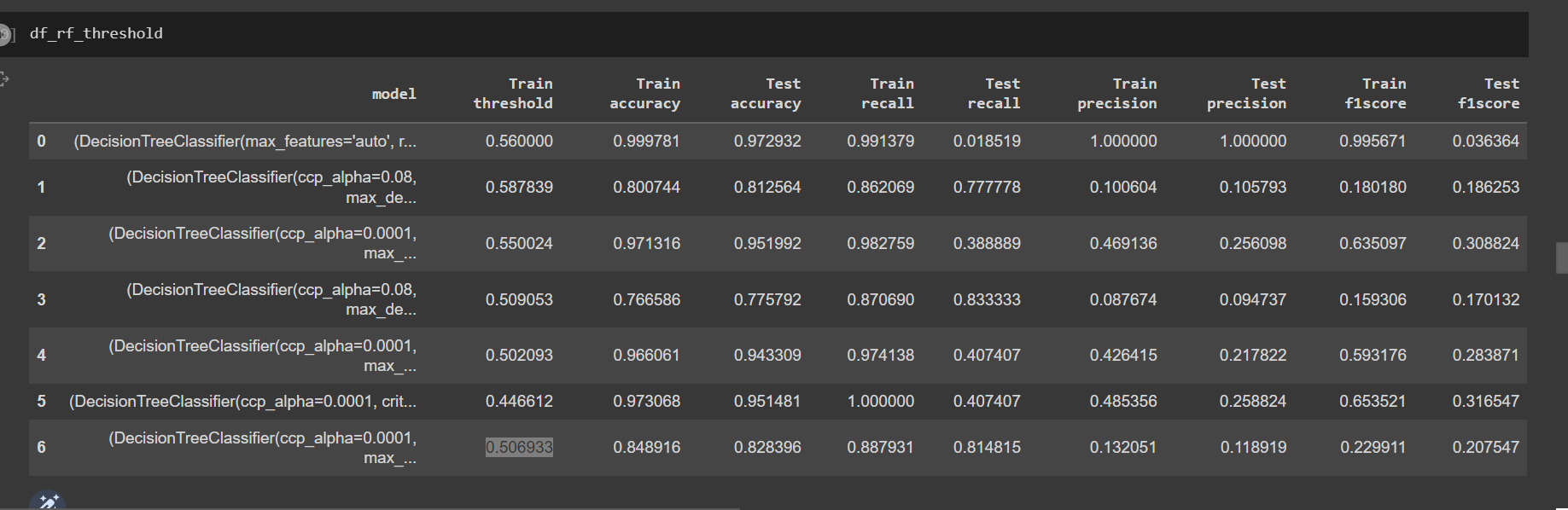


# Random Forest

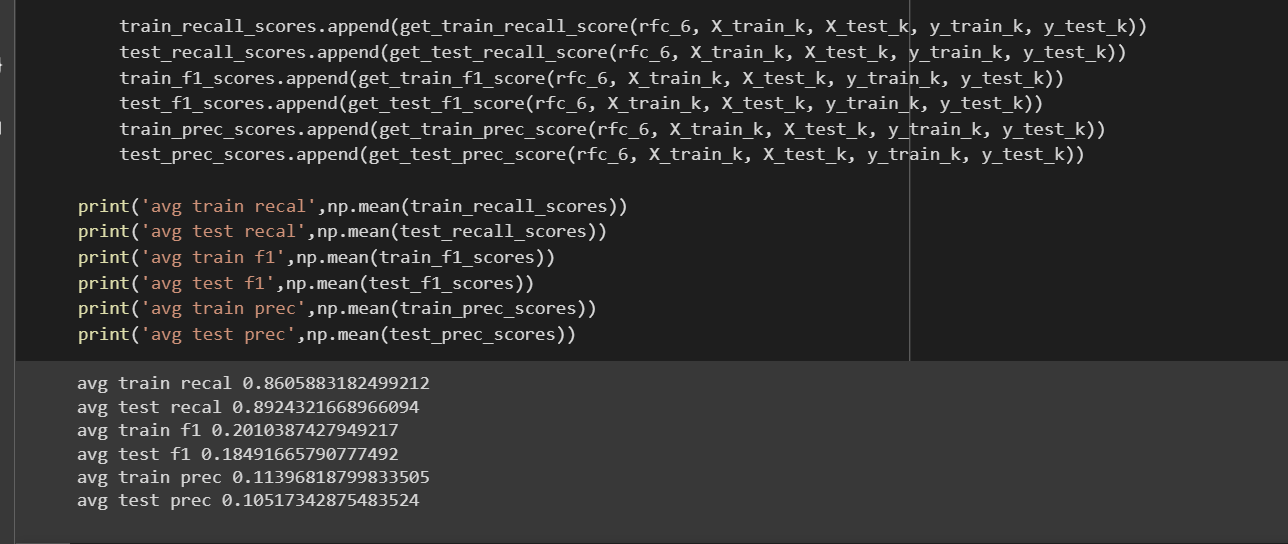
|  |  |
| --- | --- |
| **Model** | **Tuned Parameters** |
| RF Model 1 | RandomForestClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.08) |
| RF Model 2 | RandomForestClassifier(max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.0001) |
| RF Model 3 | RandomForestClassifier(n\_estimators = 37,max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.08) |
| RF Model 4 | RandomForestClassifier(n\_estimators = 30,max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.0001) |
| RF Model 5 | RandomForestClassifier(n\_estimators = 30,criterion = 'entropy',max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.0001) |
| RF Model 6 | RandomForestClassifier(n\_estimators = 30,min\_weight\_fraction\_leaf = 0.1,max\_depth = 10, max\_features= 40, max\_leaf\_nodes=24,min\_samples\_leaf= 5,min\_samples\_split = 11,class\_weight="balanced",ccp\_alpha = 0.0001) |



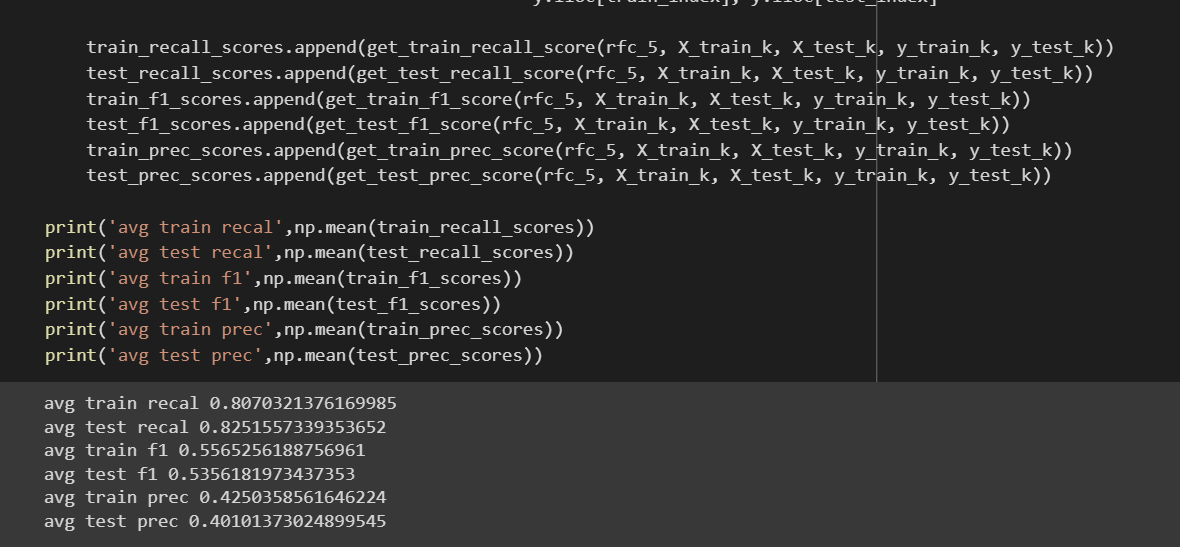




For rf\_threshold model 6



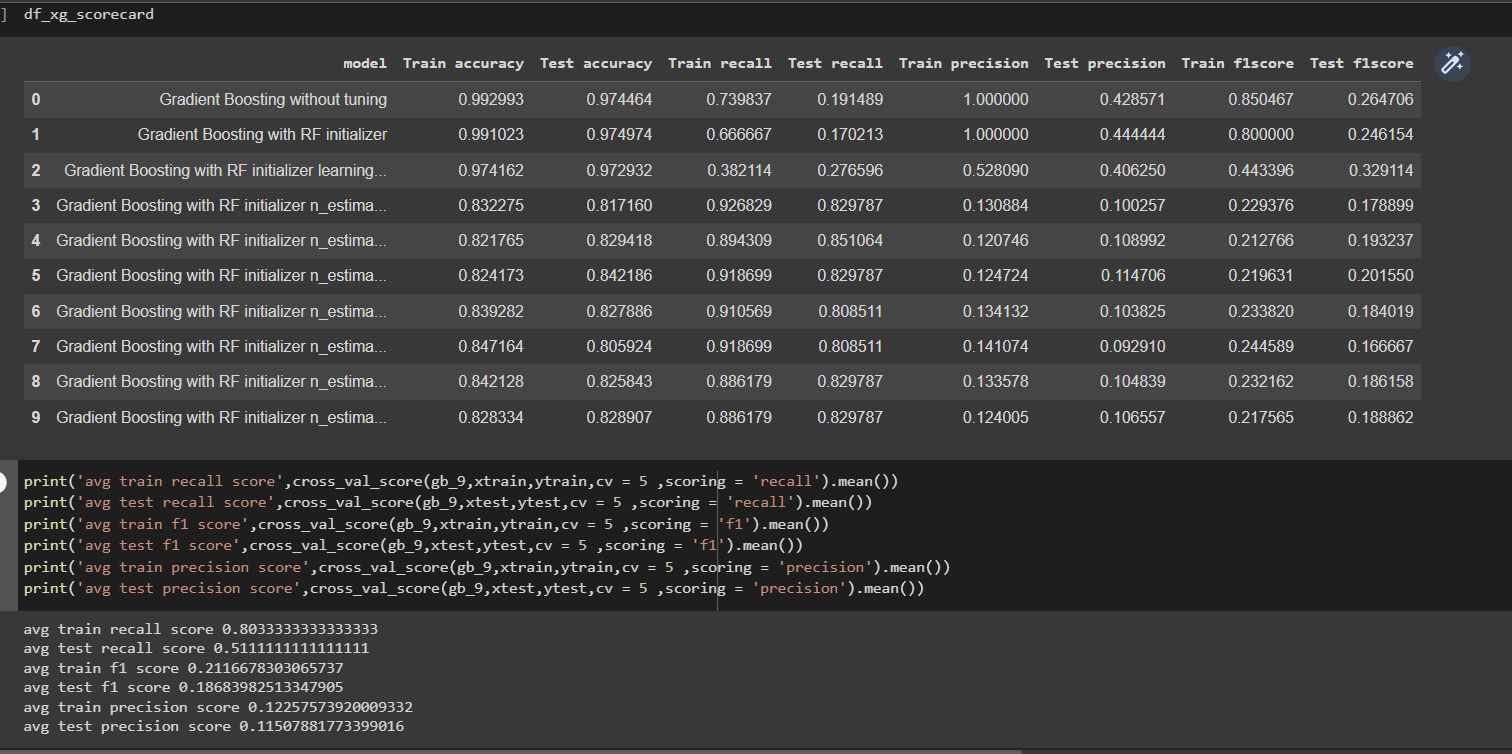
For threshold model 5

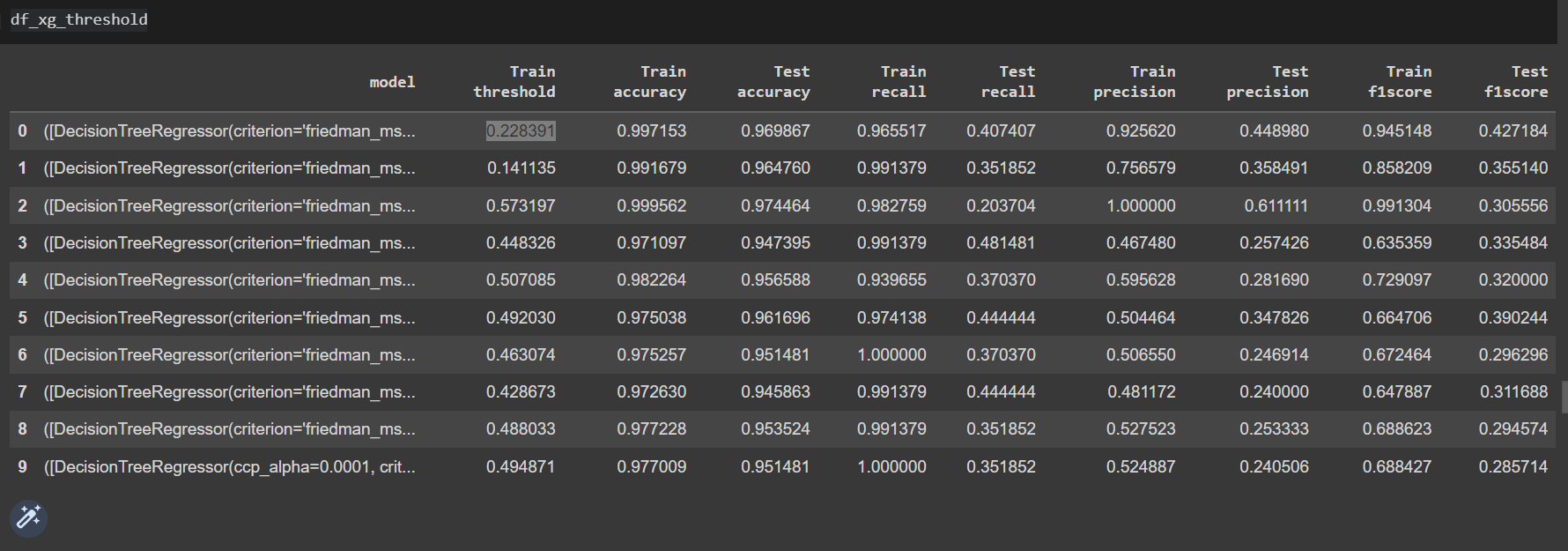


RF\_5 threshold adjustment gives good results

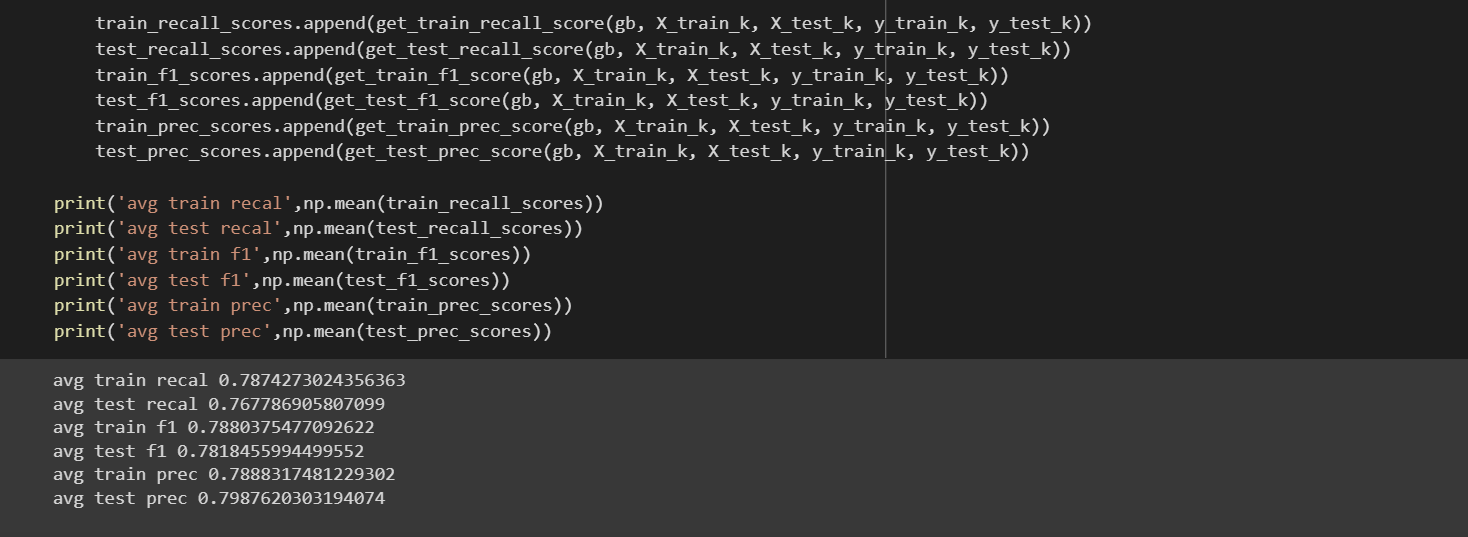
# Gradient Boosting

|  |  |
| --- | --- |
| **Model** | **Tuned Parameters** |
| GB Model 1 | GradientBoostingClassifier(init= rfc\_6) |
| GB Model 2 | GradientBoostingClassifier(learning\_rate= 0.01,loss = 'deviance',init= rfc\_5) |
| GB Model 3 | GradientBoostingClassifier(n\_estimators = 2,learning\_rate= 0.01,loss = 'deviance',init= rfc\_5) |
| GB Model 4 | GradientBoostingClassifier(n\_estimators = 2,learning\_rate= 0.01,min\_samples\_split = 4,loss = 'deviance',init= rfc\_5) |
| GB Model 5 | GradientBoostingClassifier(n\_estimators = 2,learning\_rate= 0.01,min\_samples\_split = 4,min\_samples\_leaf = 9,min\_weight\_fraction\_leaf = 0.1,loss = 'deviance',init= rfc\_5) |
| GB Model 6 | GradientBoostingClassifier(n\_estimators = 2,max\_depth = 16,learning\_rate= 0.01,min\_samples\_split = 4,min\_samples\_leaf = 9,min\_weight\_fraction\_leaf = 0.1,loss = 'deviance',init= rfc\_5) |
| GB Model 7 | GradientBoostingClassifier(n\_estimators = 2,max\_depth = 16,max\_features = 9,learning\_rate= 0.01,min\_samples\_split = 4,min\_samples\_leaf = 9,min\_weight\_fraction\_leaf = 0.1,loss = 'deviance',init= rfc\_5) |
| GB Model 8 | GradientBoostingClassifier(n\_estimators = 2,max\_depth = 16,max\_features = 9,max\_leaf\_nodes = 18,learning\_rate= 0.01,min\_samples\_split = 4,min\_samples\_leaf = 9,min\_weight\_fraction\_leaf = 0.1,loss = 'deviance',init= rfc\_5) |
| GB Model 9 | GradientBoostingClassifier(n\_estimators = 2,ccp\_alpha = 0.0001,tol = 0.2,max\_depth = 16,max\_features = 9,max\_leaf\_nodes = 18,learning\_rate= 0.01,min\_samples\_split = 4,min\_samples\_leaf = 9,min\_weight\_fraction\_leaf = 0.1,loss = 'deviance',init= rfc\_5) |



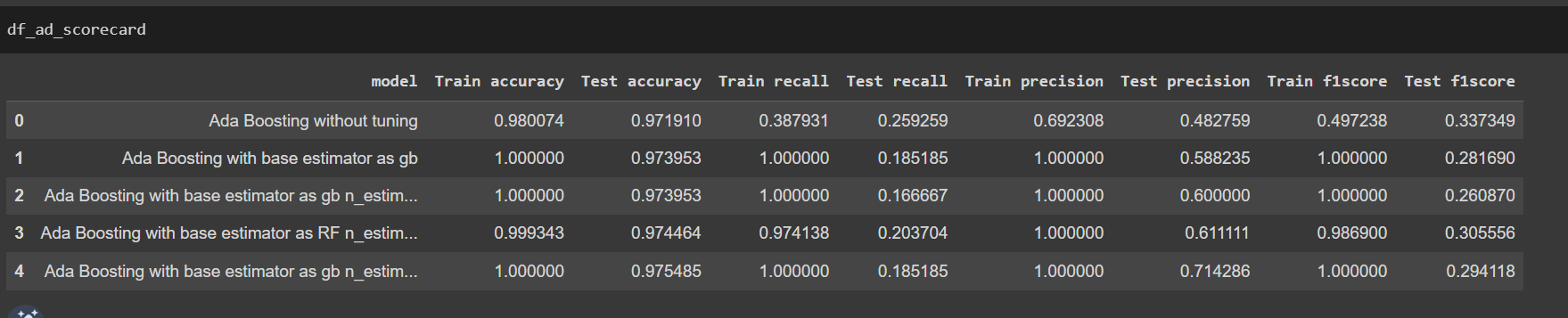


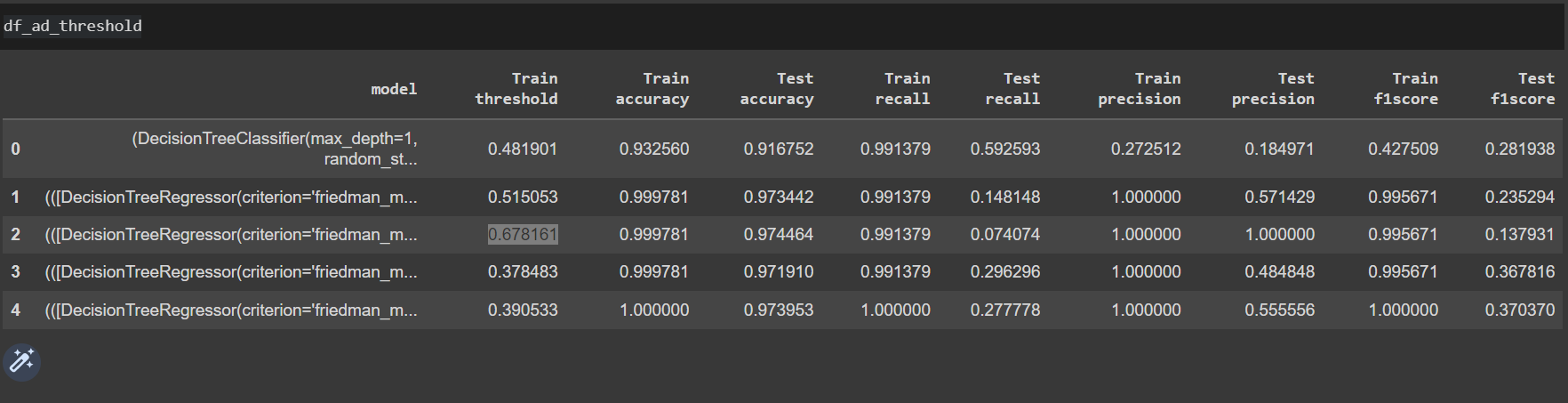
Gb with rfc\_5 initializer with default parameters gave good scores after threshold adjustment

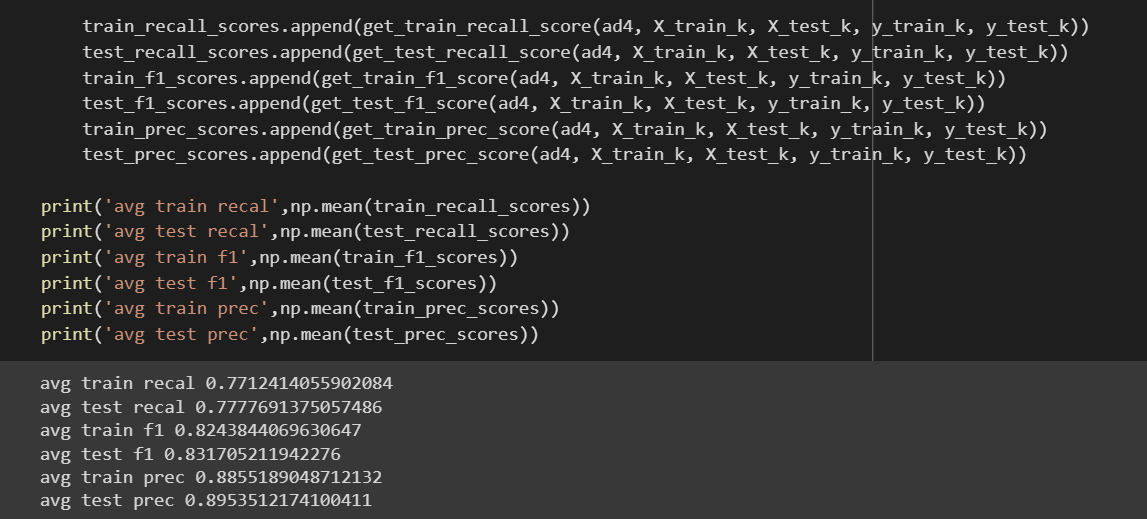


# Ada Boost

Parameters:







# Stacking Classifier

