**Part 1:**

1. **sklearn.tree.DecisionTreeClassifier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Dataset | Optimal Parameter Setting | Accuracy | F1Score |
| 1. | c300\_d100 | DecisionTreeClassifier(criterion='entropy', max\_depth=20, max\_features='log2', min\_samples\_leaf=15, random\_state=0) | Accuracy: 0.585 | F1Score: 0.5990338164251208 |
| 2. | c300\_d1000 | DecisionTreeClassifier(criterion='entropy', max\_depth=15, max\_features='sqrt',  min\_samples\_leaf=10, min\_samples\_split=100,  random\_state=0) | Accuracy: 0.6135 | F1Score: 0.6058133605303416 |
| 3. | c300\_d5000 | DecisionTreeClassifier(criterion='entropy', max\_depth=10, max\_features='auto',  min\_samples\_leaf=10, min\_samples\_split=50,  random\_state=0) | Accuracy: 0.6589 | F1Score: 0.6581136614212689 |
| 4. | c500\_d100 | DecisionTreeClassifier(max\_depth=15, max\_features='sqrt', min\_samples\_leaf=10,  random\_state=0) | Accuracy: 0.61 | F1Score: 0.59375 |
| 5. | c500\_d1000 | DecisionTreeClassifier(criterion='entropy', max\_depth=40, max\_features='sqrt',  min\_samples\_leaf=20, min\_samples\_split=100,  random\_state=0) | Accuracy: 0.6745 | F1Score: 0.6775631500742942 |
| 6. | c500\_d5000 | DecisionTreeClassifier(criterion='entropy', max\_depth=160, max\_features='auto',  min\_samples\_leaf=20, min\_samples\_split=5,  random\_state=0) | Accuracy: 0.6902 | F1Score: 0.6971652003910068 |
| 7. | c1000\_d100 | DecisionTreeClassifier(criterion='entropy', max\_depth=40, max\_features='sqrt',  min\_samples\_leaf=20, min\_samples\_split=0.3,  random\_state=0) | Accuracy: 0.705 | F1Score: 0.6289308176100629 |
| 8. | c1000\_d1000 | DecisionTreeClassifier(criterion='entropy', max\_depth=40, max\_features='auto',  min\_samples\_leaf=20, min\_samples\_split=5,  random\_state=0) | Accuracy: 0.76 | F1Score: 0.7585513078470826 |
| 9. | c1000\_d5000 | DecisionTreeClassifier(criterion='entropy', max\_depth=40, max\_features='auto',  min\_samples\_leaf=15, min\_samples\_split=50,  random\_state=0) | Accuracy: 0.8255 | F1Score: 0.8251327788355547 |
| 10. | c1500\_d100 | DecisionTreeClassifier(max\_depth=15, max\_features='log2', min\_samples\_leaf=10,  min\_samples\_split=5, random\_state=0) | Accuracy: 0.735 | F1Score: 0.7389162561576353 |
| 11. | c1500\_d1000 | DecisionTreeClassifier(criterion='entropy', max\_depth=40, max\_features='auto',  min\_samples\_leaf=15, min\_samples\_split=10,  random\_state=0) | Accuracy: 0.889 | F1Score: 0.8906403940886699 |
| 12. | c1500\_d5000 | DecisionTreeClassifier(max\_depth=10, max\_features='sqrt', min\_samples\_leaf=15,  min\_samples\_split=10, random\_state=0) | Accuracy: 0.9188 | F1Score: 0.9188000000000001 |
| 13. | c1800\_d100 | DecisionTreeClassifier(criterion='entropy', max\_depth=15, max\_features='sqrt',  min\_samples\_split=0.1, random\_state=0) | Accuracy: 0.92 | F1Score: 0.9166666666666666 |
| 14. | c1800\_d1000 | DecisionTreeClassifier(max\_depth=40, max\_features='sqrt', min\_samples\_leaf=10,  min\_samples\_split=10, random\_state=0) | Accuracy: 0.9405 | F1Score: 0.940825459970164 |
| 15. | c1800\_d5000 | DecisionTreeClassifier(criterion='entropy', max\_depth=160, max\_features='sqrt',  min\_samples\_split=50, random\_state=0) | Accuracy: 0.9615 | F1Score: 0.9613492621222769 |

1. **sklearn.ensemble.BaggingClassifier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo. | Dataset | Optimal Parameters | Accuracy | FScore |
| 1. | c300\_d100 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False,  bootstrap\_features=True, max\_features=1, n\_estimators=50,n\_jobs=5, random\_state=0) | Accuracy: 0.695 | F1Score: 0.6995073891625615 |
| 2. | c300\_d1000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False,  bootstrap\_features=True, max\_features=0.5, n\_estimators=50,n\_jobs=5, random\_state=0) | Accuracy: 0.8545 | F1Score: 0.8534005037783374 |
| 3. | c300\_d5000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(),bootstrap\_features=True, max\_features=2, n\_estimators=100,n\_jobs=5, random\_state=0) | Accuracy: 0.8052 | F1Score: 0.8053168099140515 |
| 4. | c500\_d100 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False,max\_features=1, n\_estimators=100, n\_jobs=5, random\_state=0) | Accuracy: 0.775 | F1Score: 0.7804878048780488 |
| 5. | c500\_d1000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False, bootstrap\_features=True, max\_features=1, n\_estimators=100,n\_jobs=5, random\_state=0) | Accuracy: 0.8035 | F1Score: 0.8041853512705531 |
| 6. | c500\_d5000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), max\_features=10,n\_estimators=100, n\_jobs=5, random\_state=0) | Accuracy: 0.9225 | F1Score: 0.9234567901234569 |
| 7. | c1000\_d100 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), max\_features=1,n\_estimators=100, n\_jobs=5, random\_state=0) | Accuracy: 0.89 | F1Score: 0.8962264150943395 |
| 8. | c1000\_d1000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(),bootstrap\_features=True, max\_features=2, n\_estimators=100,n\_jobs=5, random\_state=0) | Accuracy: 0.9545 | F1Score: 0.9549281822684497 |
| 9. | c1000\_d5000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(),bootstrap\_features=True, max\_features=0.5, max\_samples=0.5,  n\_estimators=50, n\_jobs=5, random\_state=0) | Accuracy: 0.9873 | F1Score: 0.987324084239944 |
| 10. | c1500\_d100 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), max\_features=2,max\_samples=0.5, n\_estimators=100, n\_jobs=5, random\_state=0) | Accuracy: 0.99 | F1Score: 0.99 |
| 11. | c1500\_d1000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False,max\_features=10, max\_samples=0.5, n\_estimators=100, n\_jobs=5,  random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 12. | c1500\_d5000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(),bootstrap\_features=True, max\_features=0.5, n\_estimators=100,n\_jobs=5, random\_state=0) | Accuracy: 0.9992 | F1Score: 0.9991995197118271 |
| 13. | c1800\_d100 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), max\_features=0.5,n\_estimators=50, n\_jobs=5, random\_state=0) | Accuracy: 0.98 | F1Score: 0.9797979797979798 |
| 14. | c1800\_d1000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False, bootstrap\_features=True, max\_features=10, n\_estimators=50,n\_jobs=5, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 15. | c1800\_d5000 | BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False,  bootstrap\_features=True, max\_features=10, n\_estimators=100,  n\_jobs=5, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |

1. **sklearn.ensemble.RandomForestClassifier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Dataset | Optimal Parameter Setting | Accuracy | F1Score |
| 1. | c300\_d100 | RandomForestClassifier(bootstrap=False, criterion='entropy', max\_depth=20, max\_features='sqrt', min\_samples\_leaf=5,  n\_estimators=200, n\_jobs=5, random\_state=0) | Accuracy: 0.84 | F1Score: 0.8446601941747572 |
| 2. | c300\_d1000 | RandomForestClassifier(criterion='entropy', max\_depth=20, max\_features='sqrt',  min\_samples\_leaf=5, n\_estimators=200, n\_jobs=5,random\_state=0) | Accuracy: 0.8915 | F1Score: 0.8930507639231148 |
| 3. | c300\_d5000 | RandomForestClassifier(max\_depth=50, max\_features='log2', n\_estimators=200,  n\_jobs=5, random\_state=0) | Accuracy: 0.9003 | F1Score: 0.9013945208189101 |
| 4. | c500\_d100 | RandomForestClassifier(max\_depth=50, max\_features='sqrt', min\_samples\_leaf=2,  n\_estimators=200, n\_jobs=5, random\_state=0) | Accuracy: 0.9 | F1Score: 0.8989898989898989 |
| 5. | c500\_d1000 | RandomForestClassifier(max\_depth=50, max\_features='log2', min\_samples\_leaf=10,  n\_estimators=200, n\_jobs=5, random\_state=0) | Accuracy: 0.9595 | F1Score: 0.9599208312716478 |
| 6. | c500\_d5000 | RandomForestClassifier(bootstrap=False, criterion='entropy', max\_depth=20,  max\_features='log2', min\_samples\_leaf=10, n\_jobs=5, random\_state=0) | Accuracy: 0.9637 | F1Score: 0.9640487273447559 |
| 7. | c1000\_d100 | RandomForestClassifier(max\_depth=50, max\_features='sqrt', min\_samples\_leaf=5,  n\_estimators=200, n\_jobs=5, random\_state=0) | Accuracy: 0.985 | F1Score: 0.9849246231155778 |
| 8. | c1000\_d1000 | Params: RandomForestClassifier(max\_depth=5, max\_features='log2', min\_samples\_leaf=10,  n\_jobs=5, random\_state=0) | Accuracy: 0.9885 | F1Score: 0.9885401096163428 |
| 9. | c1000\_d5000 | RandomForestClassifier(criterion='entropy', max\_depth=50, max\_features='log2',  min\_samples\_leaf=10, n\_estimators=50, n\_jobs=5,random\_state=0) | Accuracy: 0.9957 | F1Score: 0.995704724802717 |
| 10. | c1500\_d100 | RandomForestClassifier(max\_depth=10, max\_features='log2', min\_samples\_leaf=5,  n\_jobs=5, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 11. | c1500\_d1000 | RandomForestClassifier(bootstrap=False, criterion='entropy', max\_depth=10,  max\_features='log2', n\_estimators=200, n\_jobs=5,random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 12. | c1500\_d5000 | RandomForestClassifier(bootstrap=False, criterion='entropy', max\_depth=20,  max\_features='log2', min\_samples\_leaf=5, n\_jobs=5,random\_state=0) | Accuracy: 0.9999 | F1Score: 0.9998999899989999 |
| 13. | c1800\_d100 | RandomForestClassifier(max\_depth=50, max\_features=0.1, min\_samples\_leaf=10,  n\_jobs=5, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 14. | c1800\_d1000 | RandomForestClassifier(max\_depth=50, max\_features='log2', min\_samples\_leaf=5,  n\_estimators=50, n\_jobs=5, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 15. | c1800\_d5000 | RandomForestClassifier(max\_depth=50, max\_features='sqrt', min\_samples\_leaf=2,  n\_estimators=50, n\_jobs=5, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |

1. **sklearn.ensemble.GradientBoostingClassifier**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Dataset | Optimal Parameter Setting | Accuracy | F1Score |
| 1. | c300\_d100 | GradientBoostingClassifier(max\_depth=10, max\_features='log2', n\_estimators=50,random\_state=0) | Accuracy: 0.71 | F1Score: 0.7040816326530612 |
| 2. | c300\_d1000 | GradientBoostingClassifier(criterion='squared\_error', max\_depth=10,max\_features='log2', min\_samples\_leaf=10,random\_state=0) | Accuracy: 0.88 | F1Score: 0.8815399802566634 |
| 3. | c300\_d5000 | GradientBoostingClassifier(criterion='mse', max\_depth=10, max\_features='sqrt',min\_samples\_leaf=2, random\_state=0) | Accuracy: 0.9432 | F1Score: 0.944116489571035 |
| 4. | c500\_d100 | GradientBoostingClassifier(max\_depth=10, max\_features='sqrt', min\_samples\_leaf=10, random\_state=0) | Accuracy: 0.92 | F1Score: 0.9207920792079208 |
| 5. | c500\_d1000 | GradientBoostingClassifier(loss='exponential', max\_depth=5, max\_features='log2',random\_state=0) | Accuracy: 0.9475 | F1Score: 0.9477351916376308 |
| 6. | c500\_d5000 | GradientBoostingClassifier(criterion='squared\_error', max\_depth=10, max\_features='sqrt', min\_samples\_leaf=10,random\_state=0) | Accuracy: 0.9698 | F1Score: 0.9700753071739991 |
| 7. | c1000\_d100 | GradientBoostingClassifier(criterion='squared\_error', loss='exponential',  max\_depth=1, max\_features='sqrt',  min\_samples\_leaf=10, random\_state=0) | Accuracy: 0.965 | F1Score: 0.9655172413793104 |
| 8. | c1000\_d1000 | GradientBoostingClassifier(criterion='mse', max\_depth=5, max\_features='log2',min\_samples\_leaf=5, random\_state=0) | Accuracy: 0.996 | F1Score: 0.9960039960039959 |
| 9. | c1000\_d5000 | GradientBoostingClassifier(criterion='squared\_error', max\_depth=5, max\_features='sqrt', min\_samples\_leaf=10,random\_state=0) | Accuracy: 0.9963 | F1Score: 0.9963018490754623 |
| 10. | c1500\_d100 | GradientBoostingClassifier(loss='exponential', max\_depth=5, max\_features='log2',min\_samples\_leaf=10, n\_estimators=50,random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 11. | c1500\_d1000 | GradientBoostingClassifier(criterion='mse', max\_depth=5, max\_features='log2',min\_samples\_leaf=5, n\_estimators=50, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 12. | c1500\_d5000 | GradientBoostingClassifier(criterion='squared\_error', loss='exponential',  max\_depth=5, max\_features='log2', min\_samples\_leaf=5,  random\_state=0) | Accuracy: 0.9997 | F1Score: 0.9996999699969997 |
| 13. | c1800\_d100 | GradientBoostingClassifier(criterion='mse', loss='exponential', max\_depth=5,max\_features='log2', min\_samples\_leaf=5,n\_estimators=50, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 14. | c1800\_d1000 | GradientBoostingClassifier(criterion='squared\_error', max\_depth=5,max\_features='log2', min\_samples\_leaf=10,  random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |
| 15. | c1800\_d5000 | GradientBoostingClassifier(criterion='squared\_error', loss='exponential',max\_depth=5, max\_features='sqrt',  min\_samples\_leaf=10, n\_estimators=50, random\_state=0) | Accuracy: 1.0 | F1Score: 1.0 |

5.

a. Best Accuracy/F1 Score among the four classifiers and why?

**Random Forest** yields the best overall generalization based on the above results. Random forest is a collection of decision trees with a single, aggregated result. Although for some datasets, the accuracy of Gradient Boosting Classifier is more than Random Forest, Gradient Boosting tends to overfit the data, which Random Forest tries to avoid by choosing only a random set of features from the input data when constructing forests/decision trees. Therefore, the performance of Gradient Boosting mainly depends on how well we tune the hyper parameters, but the random forest has the best overall performance.

b. Impact of **increasing train data**

|  |  |
| --- | --- |
| **Classifier** | **Observation** |
| **Decision Tree** | A slight change in accuracy and F1 scores |
| **Bagging** | A noticeable change in accuracy and F1 score |
| **Random Forest** | A change in accuracy and F1 score |
| **Gradient Boosting** | A change in accuracy and F1 score |

c. Impact of **increasing #features**

|  |  |
| --- | --- |
| **Classifier** | **Observation** |
| **Decision Tree** | A noticeable change in accuracy and F1 score |
| **Bagging** | A noticeable change in accuracy and F1 score |
| **Random Forest** | A noticeable change in accuracy and F1 score |
| **Gradient Boosting** | A noticeable change in accuracy and F1 score |

**Extra Credit: MNIST dataset**

Q: Which classifier among the four yields the best generalization accuracy on the MNIST dataset and why?

A:

Ran the classifiers with **10K size train set** and **1000 size test set**.

**Decision Tree**

Dataset: MNIST

Params: DecisionTreeClassifier(criterion='entropy', max\_depth=60, max\_features='auto',min\_samples\_split=7)

**Accuracy:** 0.769

**Bagging**

Dataset with Decision Tree(unstable): MNIST

Params: BaggingClassifier(base\_estimator=DecisionTreeClassifier(), bootstrap=False,max\_features=10, max\_samples=25, n\_estimators=100, random\_state=0)

**Accuracy:** 0.624

**Random** **Forest**

Dataset: MNIST

Params: RandomForestClassifier(bootstrap=False, max\_depth=20, max\_features='sqrt',min\_samples\_leaf=2, n\_estimators=500, n\_jobs=-1,

random\_state=0)

**Accuracy:** 0.963

**Gradient** **Boosting**

Dataset: MNIST

Params: GradientBoostingClassifier(criterion='mse', max\_depth=10, max\_features='log2',n\_estimators=50, random\_state=0)

**Accuracy:** 0.944

MNIST dataset gets highest generalization with **Random Forest classifier**. It is because Random Forest classifiers introduces a randomness into the iterations, by choosing a best feature out of randomly selected m features instead of trying to find best feature out of all the features. This randomness helps avoid overfitting and provides the ability to generalize better.