**Tree Based Classifiers and Their Ensemble Methods**

**Table 1: Classification Accuracy**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | **DecisionTree** | **Bagging** | **RandomForest** | **GradientBoosting** |
| c300\_d100 | 0.65 | 0.81 | 0.83 | 0.79 |
| c300\_d1000 | 0.6685 | 0.8745 | 0.877 | 0.9945 |
| c300\_d5000 | 0.7796 | 0.9351 | 0.9249 | 0.9994 |
| c500\_d100 | 0.705 | 0.855 | 0.885 | 0.87 |
| c500\_d1000 | 0.693 | 0.9295 | 0.9525 | 0.998 |
| c500\_d5000 | 0.7898 | 0.9609 | 0.9578 | 0.9998 |
| c1000\_d100 | 0.685 | 0.94 | 0.98 | 0.985 |
| c1000\_d1000 | 0.7955 | 0.975 | 0.9925 | 0.9985 |
| c1000\_d5000 | 0.8546 | 0.9894 | 0.9953 | 0.9999 |
| c1500\_d100 | 0.855 | 1.0 | 1.0 | 1.0 |
| c1500\_d1000 | 0.925 | 0.996 | 1.0 | 1.0 |
| c1500\_d5000 | 0.9544 | 0.9982 | 0.9997 | 1.0 |
| c1800\_d100 | 0.915 | 0.995 | 1.0 | 0.995 |
| c1800\_d1000 | 0.971 | 0.9985 | 1.0 | 1.0 |
| c1800\_d5000 | 0.9844 | 0.9999 | 1.0 | 1.0 |

**Table 2: F1 Score**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | **DecisionTree** | **Bagging** | **RandomForest** | **GradientBoosting** |
| c300\_d100 | 0.649123 | 0.809981 | 0.829932 | 0.789916 |
| c300\_d1000 | 0.668463 | 0.874491 | 0.876999 | 0.994500 |
| c300\_d5000 | 0.779281 | 0.935050 | 0.924839 | 0.999400 |
| c500\_d100 | 0.704816 | 0.854560 | 0.884974 | 0.869883 |
| c500\_d1000 | 0.692969 | 0.929495 | 0.952497 | 0.998000 |
| c500\_d5000 | 0.789346 | 0.960899 | 0.957798 | 0.999800 |
| c1000\_d100 | 0.684044 | 0.940000 | 0.980000 | 0.984999 |
| c1000\_d1000 | 0.794935 | 0.974999 | 0.992500 | 0.998500 |
| c1000\_d5000 | 0.854305 | 0.989400 | 0.995300 | 0.999900 |
| c1500\_d100 | 0.854909 | 1.000000 | 1.000000 | 1.000000 |
| c1500\_d1000 | 0.924978 | 0.995999 | 1.000000 | 1.000000 |
| c1500\_d5000 | 0.954393 | 0.998200 | 0.999700 | 1.000000 |
| c1800\_d100 | 0.914742 | 0.994999 | 1.000000 | 0.994999 |
| c1800\_d1000 | 0.970997 | 0.998500 | 1.000000 | 1.000000 |
| c1800\_d5000 | 0.984399 | 0.999900 | 1.000000 | 1.000000 |

**Analysis of Synthetic Datasets**

1. **Best Overall Generalization:** Gradient Boosting and Random Forest generalized the best, with both achieving some perfect scores on the larger datasets. In this experiment the Gradient Boosting performed slightly better on average, with some exceptions. This is because of their ability to combine multiple weak learners and correct the errors made by previous models.
2. **Impact of Training Data Size:** Increasing the data size improves each of the models F1 scores and accuracy, with Gradient Boosting and Random Forest achieving perfect or almost perfect scores on the larger datasets. The Decision Trees had improvements too but they plateaud at a lower level than the ensemble methods, so the impact of training data size on them wasn’t as much.
3. **Effect of Number of Features (Clauses):** Increasing the number of features (clauses) impacts the classifiers differently. For Decision Trees the performance fluctuates and doesn't show a clear increasing or decreasing trend as the number of clauses increases. However, with ensemble methods, especially Random Forest and Gradient Boosting, the performance tends to increase , especially with larger sample sizes. This suggests that ensemble methods are better at handling higher-dimensional data.

**MNIST Dataset Experiment**

The classification accuracy of each classifier on the MNIST dataset is as follows:

* Decision Tree: 0.8757
* Bagging: 0.9426
* Random Forest: 0.9697
* Gradient Boosting: 0.9458

**Analysis of MNIST Dataset**

Random Forest achieves the highest classification accuracy on the MNIST dataset (0.9697). This performance can be attributed to Random Forest's ability to reduce overfitting by averaging multiple decision trees trained on different subsets of the data and features. The diversity in the random sampling helps create a more generalizable model compared to individual Decision Trees or Bagging. While Gradient Boosting also performs well, Random Forest's architecture seems particularly effective for the high-dimensional and complex nature of the MNIST dataset.

**References**

[1] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830. Retrieved from <https://scikit-learn.org/>

[2] McKinney, W. (2010). *Data Structures for Statistical Computing in Python*. Proceedings of the 9th Python in Science Conference, 51–56. Retrieved from https://pandas.pydata.org/

[3] van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). *The NumPy Array: A Structure for Efficient Numerical Computation*. Computing in Science & Engineering, 13(2), 22–30. Retrieved from <https://numpy.org/>

[4] The Python Software Foundation. (2024). *OS — Miscellaneous operating system interfaces*. Python 3 Documentation. Retrieved from <https://docs.python.org/3/library/os.html>

[5] Buitinck, L., Fortiana, J., Holger, H., et al. (2013). *API design for machine learning software: experiences from the scikit-learn project*. arXiv preprint arXiv:1309.0238. Retrieved from <https://arxiv.org/abs/1309.0238>

[6] FetchOpenML. (n.d.). *OpenML: Machine Learning Datasets and Experimentation Platform*. Retrieved from <https://www.openml.org/>

[7] Joblib Development Team. (2024). *Joblib: Running Python functions as pipeline jobs*. Retrieved from <https://joblib.readthedocs.io/>