REPORT

Project 2: Evaluation of Tree-Based Classifiers and Their Ensembles

Comparative Analysis: Record the classification accuracy and F1 scores for each testset and classifier in a table. You can arrange all your results in a table shown above (Table 11). Make sure you have two tables: one for classification accuracy and one for F1 Score.

TABLE 1: CLASSIFICATION ACCURACY RESULTS

Dataset	Decision Tree	Bagging	Random Forest	Gradient Boosting
c300_d100	0.5350	0.8000	0.8450	0.8050
c300_d1000	0.6610	0.8855	0.8925	0.9075
c300_d5000	0.7793	0.9353	0.9305	0.9342
c500_d100	0.5200	0.8900	0.9000	0.9050
c500_d1000	0.6760	0.8880	0.9520	0.9545
c500_d5000	0.7883	0.9505	0.9700	0.9725
c1000_d100	0.7450	0.9100	1.0000	1.0000
c1000_d1000	0.7945	0.9495	0.9975	0.9945
c1000_d5000	0.8597	0.9784	0.9980	0.9981
c1500_d100	0.8450	0.9800	1.0000	1.0000
c1500_d1000	0.9140	0.9885	1.0000	1.0000
c1500_d5000	0.9531	0.9941	1.0000	1.0000
c1800_d100	0.9500	0.9850	1.0000	1.0000
c1800_d1000	0.9755	0.9950	1.0000	1.0000
c1800_d5000	0.9892	0.9988	1.0000	1.0000

TABLE 2: F1 SCORE RESULTS OF EACH CLASSIFIER

Dataset	Decision Tree	Bagging	Random Forest	Gradient Boosting
c300_d100	0.5507	0.8020	0.8473	0.8116
c300_d1000	0.6558	0.8902	0.8942	0.9084
c300_d5000	0.7874	0.9374	0.9317	0.9351
c500_d100	0.5102	0.8962	0.8958	0.9055
c500_d1000	0.6667	0.8861	0.9520	0.9547
c500_d5000	0.7981	0.9508	0.9703	0.9727
c1000_d100	0.7437	0.9082	1.0000	1.0000
c1000_d1000	0.7966	0.9503	0.9975	0.9945
c1000_d5000	0.8646	0.9784	0.9980	0.9981
c1500_d100	0.8442	0.9804	1.0000	1.0000
c1500_d1000	0.9159	0.9885	1.0000	1.0000
c1500_d5000	0.9536	0.9941	1.0000	1.0000
c1800_d100	0.9510	0.9849	1.0000	1.0000
c1800_d1000	0.9758	0.9950	1.0000	1.0000
c1800_d5000	0.9892	0.9988	1.0000	1.0000

1. **Which classifier achieves the best overall generalization accuracy/F1 score? Explain why.**

   Gradient Boosting achieves the highest overall accuracy and F1 score across datasets (≥ 0.99 for larger clause sizes) as it iteratively minimizes residual errors and learns complex relationships between the features and target values.

   Random Forest performs better as well, however it shows slightly less accuracy but faster training and greater robustness to noise. Meanwhile, Bagging improves over

single trees but lacks the adaptive learning mechanism of boosting and Decision Tree performs weakest, showing high variance and sensitivity to data size.

Thus, **Gradient Boosting** achieves the best generalization accuracy and F1 score

2. **How does increasing the training data size impact accuracy/F1 score for each classifier?**

   As the number of training examples increases, that is from 100 → 500 →1000 → 5000, the accuracy and F1 score of all classifiers consistently improve. However, the rate of improvement differs across models.

   In decision tree, Accuracy and F1 scores improve significantly with more data. As a single tree tends to overfit small datasets, Larger training sets provide more representative patterns, helping it generalize better.

   Bagging with the Decision Tree shows moderate improvement. With more data, each tree in the ensemble sees diverse samples, reducing variance and improving stability. Performance becomes more reliable.

   In Random Forest, the model shows high and stable performance even with smaller data and improves slightly with large data. Random sampling and feature randomness make it robust to limited data, but more data still enhances generalization.

   Gradient Boosting shows strong performance on small data and becomes nearly perfect (≥0.99) as examples increase. Sequential learning allows it to learn from residual errors effectively, and more data reduces noise, improving convergence and accuracy.

   Thus the overall trend each classifiers are:
   As more training examples are added, higher accuracy and F1 scores are shown for all models.

   Performance gains are largest for simpler models (Decision Tree) and smaller for ensemble methods that already generalize well.

   With enough data (≥ 1000 examples), Random Forest and Gradient Boosting achieve near-perfect classification.

3. **How does increasing the number of features (clauses) affect classifier performance?**

   As the number of features increases, the overall performance of all classifiers generally improves because the models gain access to richer information for learning decision boundaries.
   Decision Trees show a noticeable rise in accuracy at first, as more features allow deeper and more informative splits, but they can also become prone to overfitting when the feature space becomes too large.
   Bagging classifiers benefit from additional features since each tree in the ensemble learns from different subsets of the data, reducing variance and improving stability.
   Random Forests handle the increase in clauses even more effectively by randomly selecting subsets of features at each split, which helps capture important relationships while avoiding overfitting; their performance improves steadily and then plateaus once enough clauses are included.
   Gradient Boosting, on the other hand, consistently achieves the best results as feature count grows—it effectively learns complex patterns and refines predictions through iterative error correction.
   Overall, increasing the number of clauses enhances model accuracy and F1 scores across all classifiers, with Gradient Boosting and Random Forests showing the strongest and most stable improvements.

MNIST EXPERIMENT

Evaluate the four classifiers used earlier—Decision Trees, Bagging, Random Forest, and Gradient Boosting—on the MNIST dataset. Report their classification accuracy (do not compute F1 scores).

Here are the MNIST experiment results with all classifiers and their test accuracies:

Classifier	**Test Accuracy**	**Time (min)**
Decision Tree	0.8443	0.1
Bagging	0.9539	8.0
Random Forest	0.9676	0.3
Gradient Boosting	0.9651	37.9

From the accuracy summary above, we can see that Random Forest is best classifier for MNIST dataset. Decision Tree is fastest to train but shows lowest accuracy (0.8443). Bagging shows significant boost in accuracy (0.9539) at the cost of longer training time (8 min). Random Forest shows the best trade-off between accuracy (0.9676) and training time (0.3 min). Gradient Boosting shows high accuracy (0.9651), but very long training time (37.9 min) and not as high as Random Forest.

Thus, Random Forest achieved the highest accuracy with minimal training time, making it the most efficient classifier for this MNIST experiment.

Which classifier achieves the highest classification accuracy on MNIST? Provide a brief explanation for its superior performance.

Random Forest is the classifier with the highest classification accuracy on MNIST. Here are the reasons why Random Forest performs well:

1. Ensemble Effect:

Random Forest builds many decision trees and aggregates their predictions (usually by majority vote). This reduces the chance of making mistakes that a single tree might make, improving overall accuracy.

2. Reduction of Overfitting:

A single decision tree can overfit the training data, capturing noise instead of general patterns. Random Forest mitigates overfitting by averaging predictions across trees, making it generalize better to unseen MNIST images.

3. Random Feature Selection:

Each tree considers only a random subset of features at each split.
This ensures that the trees are diverse, capturing different aspects of the data, which improves the ensemble's predictive power.

4. Bagging (Bootstrap Aggregating):

Trees are trained on different random samples of the dataset.
This further increases diversity among trees, helping the model handle variability in MNIST digits more effectively.

In contrast:

Decision Tree: Fast but prone to overfitting and therefore shows lower accuracy.

Bagging: Better than a single tree but doesn’t use feature randomness → slightly lower accuracy than Random Forest.
Gradient Boosting: Very accurate but slower, can overfit if not tuned properly.