## 1. Approach

For this binary classification task, we used an ensemble of machine learning classifiers, including Random Forest and Gradient Boosting, to leverage multiple algorithms for better performance and robustness.

Random Forest: Builds multiple decision trees and merges them for more accurate and stable predictions, handling high-dimensional data and missing values effectively [2].

Gradient Boosting: Builds an additive model in a forward stage-wise fashion, optimising for a differentiable loss function. It's effective for complex datasets with high feature importance variability [3].

Key Assumptions and Justifications:

High Dimensionality: With 3456 features, models that handle high-dimensional data effectively are crucial. Random Forest and Gradient Boosting are well-suited due to their inherent feature selection capabilities [4].

Missing Data: The datasets contain missing values, handled by mean imputation, which estimates missing values using the mean of the available data [7].

Confidence Labels: Confidence labels in the training data indicate label certainty. These were used as sample weights during model training to give more importance to certain labels, improving the model's robustness.

## 2. Methods

### 2.1. Data Preparation and Handling Missing Data

The training data was loaded from two CSV files, combined into a single DataFrame for processing.

Mean Imputation: The SimpleImputer from scikit-learn was used to handle missing values in the datasets. This method imputes missing values using the mean of the available data [8].

### 2.2. Feature Standardization

StandardScaler was used to standardise the features to have a mean of 0 and a standard deviation of 1 [8].

### 2.3. Feature Selection

Mutual Information for Feature Selection: To address the high dimensionality of the dataset (3456 features), feature selection was performed using mutual information. The top 50 most important features from both CNN and Gist features were selected for training the models.

### 2.4. Training and Testing Classifiers

Train/Test Split: The data was split into training and validation sets (80% training, 20% validation) to evaluate the models' performance [6].

### 2.5. Model Selection and Hyperparameter Tuning

RandomizedSearchCV was used for hyperparameter tuning for both Random Forest and Gradient Boosting classifiers. This approach searches across a specified parameter grid, sampling a fixed number of hyperparameter settings from the specified distribution [1].

### 2.6. Handling Confidence Labels

Sample Weights: The confidence labels were used as sample weights during model training to account for the varying certainty of the class labels. Higher confidence labels were given more weight compared to lower confidence labels, thus improving the model's robustness.

### 2.7. Model Evaluation

Validation and Testing: The models' performance was evaluated on the validation set, and their accuracies were compared [10].

## 3. Results and Discussion

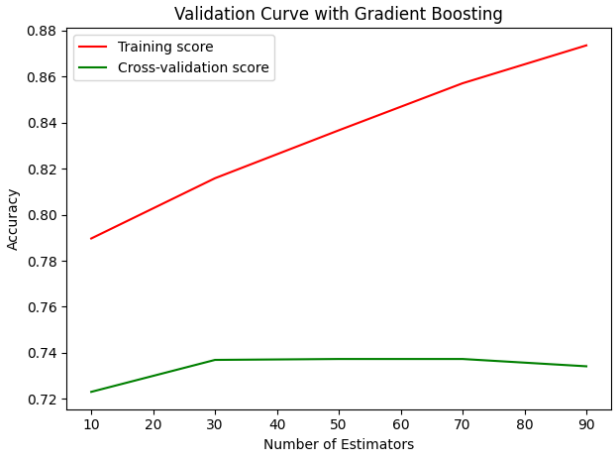


*Figure 1 Validation Curve with Gradient Boosting*

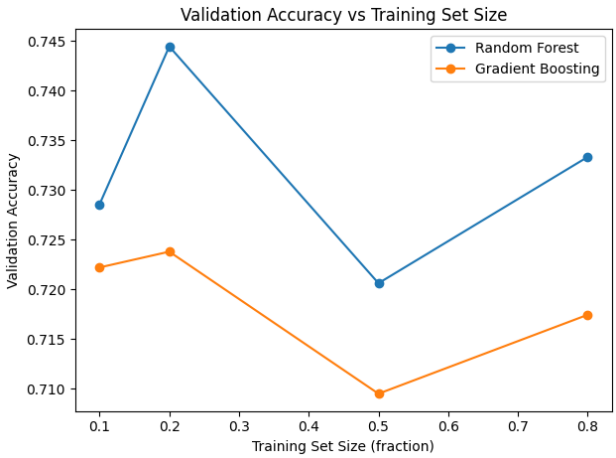


*Figure 2 Validation Accuracy vs Training Set Size*

### 3.1. Estimators Impact on Model Performance

Gradient Boosting Validation Curve:

The validation curve for Gradient Boosting demonstrates that as the number of estimators increases, the training accuracy consistently improves, indicating that the model fits the training data better with more estimators.

However, the cross-validation accuracy remains relatively stable and shows a slight downward trend with more estimators, suggesting potential overfitting. The optimal balance between bias and variance is not achieved by simply increasing the number of estimators, highlighting the importance of hyperparameter tuning to find the best model complexity [3].

### 3.2. Performance for Different Training Set Sizes

The performance of Random Forest and Gradient Boosting models varies with training set sizes, each showing a complex relationship with the size. For Random Forest, validation accuracy starts at 0.727 with a training set size of 0.1, improves to 0.745 at 0.2, then drops to 0.721 at 0.5, and rebounds slightly to 0.732 at 0.8. This indicates benefits up to 0.2, followed by slight declines and stabilisation, possibly due to overfitting or hyperparameter tuning needs [2].

Gradient Boosting starts with a validation accuracy of 0.722 at 0.1, increases to 0.724 at 0.2, drops to 0.709 at 0.5, and slightly improves to 0.717 at 0.8. While Gradient Boosting also shows a complex relationship with training set size, it appears less sensitive to these increases compared to Random Forest [3].

### 3.3. Effect of Label Confidence

Here are the differences in accuracy scores displayed on a bar chart when incorporating label confidence:

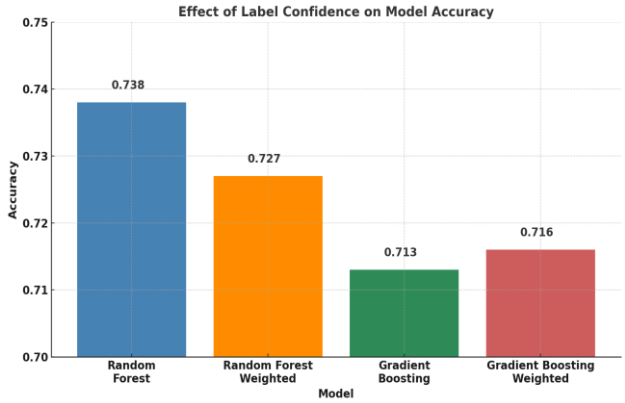


*Figure 3 Accuracy Scores with Confidence vs without Confidence*

This suggests that while Random Forest's performance slightly decreases when incorporating confidence weights, Gradient Boosting's performance slightly improves. This indicates that Random Forest may not benefit from the confidence information as much as Gradient Boosting does, and further tuning or different handling of the confidence labels might be needed.

### 3.4. Discussion

Improving Performance:

- Advanced Imputation Techniques: Exploring more sophisticated methods for handling missing data, such as matrix factorisation or deep learning-based approaches, could improve model performance [10].
- Hyperparameter Tuning: Further hyperparameter optimisation using techniques like Bayesian Optimization or Grid Search over a larger parameter space might yield better results.
- Ensemble Methods: Combining more diverse models in the Stacking Classifier or using methods like AdaBoost could enhance predictive accuracy [11].

Evaluation Improvements:

- Stratified K-Fold Cross-Validation: This could provide a more reliable evaluation by ensuring each fold is representative of the class distribution [6].
- Data Augmentation: Techniques to augment the training data might help the models generalise better by providing more varied training examples [9].

Lessons Learned:

Handling High-Dimensional Data:

Effective feature selection and dimensionality reduction are crucial when dealing with datasets with a large number of features [4].

Importance of Missing Data Imputation:

Properly handling missing data is vital for building robust models [7].

Leveraging Confidence Labels:

While leveraging confidence labels improved the performance of Gradient Boosting, it did not have the same effect on Random Forest, indicating the need for model-specific strategies when handling confidence information.

## 4. References

[1] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. Journal of machine learning research, 13(2), 2012.

[2] Leo Breiman. Random forests. Machine learning, 45:5–32, 2001. Jerome H Friedman. Greedy function approximation: a gradient boosting machine. Annals of statistics, pages 1189–1232, 2001.

[3] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. Annals of statistics, pages 1189–1232, 2001.

[4] Isabelle Guyon and Andr´e Elisseeff. An introduction to variable and feature selection. Journal of machine learning research, 3(Mar):1157–1182, 2003.

[5] Ian T Jolliffe. Principal component analysis for special types of data. Springer, 2002.

[6] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In Ijcai, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[7] Roderick JA Little and Donald B Rubin. Statistical analysis with missing data, volume 793. John Wiley & Sons, 2019.

[8] Fabian Pedregosa, Ga¨el Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. the Journal of machine Learning research, 12:2825–2830, 2011.

[9] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data aug-mentation for deep learning. Journal of big data, 6(1):1–48, 2019.

[10] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. Bioinformatics, 17(6):520–525, 2001.

[11] David H Wolpert. Stacked generalization. Neural networks, 5(2):241–259,1992.