Langara.
THE COLLEGE OF HIGHER LEARNING.

Machine Learning-Powered Handwriting Analysis for Early Detection of Alzheimer's Disease

Thi Bich Thao Nguyen (Alice)
Camilo Estrada
Stephania Nino
Piero Vera

DANA4830

July 2024

Contents

1. Background

1.1. Data

The team chose the 'DARWIN' which was built to improve the existing machine-learning methodologies for the prediction of Alzheimer's disease via handwriting analysis.

The mentioned dataset is located at UCI repository.

It was created by Cillia, Gregorio, Fontanella, Marcelli, & Parziale. They collected data from **174 participants** (89 AD patients and 85 healthy controls) using a Wacom Bamboo tablet, recording pen tip movements and pressure.

Among the columns, we found the structure of the numerical features, as follows:

Column 1	Column 2	…	Column 18	Column 19	Column 20
Column 1 (Task 1)	Column 2 (Task 1)	…	Column 2 (Task 1)	Column 1 (Task 2)	Column 2 (Task 2)

From where, each 18 set of columns represents the numeric values for one handwriting task, resulting in 25 performed tasks (**450 numeric features**).

These 18 variables represent:

- Time Features:
  - Total Time (TT): Overall task duration.
  - Air Time (AT): Time spent with the pen in the air.
  - Paper Time (PT): Time spent writing on the paper.
- Speed Features:
  - Mean Speed on-paper (MSP): Average speed of writing on paper.
  - Mean Speed in-air (MSA): Average speed of pen movement in the air.
  - Movement Smoothness Features:
  - Mean Acceleration on-paper (MAP): Average acceleration of writing on paper.
  - Mean Acceleration in-air (MAA): Average acceleration of pen movement in the air.
  - Mean Jerk on-paper (MJP): Average jerk (change in acceleration) of writing on paper.
  - Mean Jerk in-air (MJA): Average jerk of pen movement in the air.
- Pressure Features:
  - Pressure Mean (PM): Average pressure exerted by the pen on the paper.
  - Pressure Var (PV): Variance (fluctuation) of the pressure exerted by the pen.
- Global Mean Relative Tremor (GMRT) Features:
  - GMRT on-paper (GMRTP): Measure of tremor during writing on paper.
  - GMRT in-air (GMRTA): Measure of tremor during in-air movements.
  - Mean GMRT (GMRT): Average of GMRTP and GMRTA.
- Other Features:
  - Pendowns Number (PWN): Number of times the pen touches the paper.
  - Max X Extension (XE): Maximum horizontal distance covered by writing.

- Max Y Extension (YE): Maximum vertical distance covered by writing.
- Dispersion Index (DI): Measure of how much of the paper is used for writing.

1.2. Target variable

Whether a participant has Alzheimer's disease or is healthy.

- P: Stands for "Patients", referring to individuals diagnosed with Alzheimer's Disease.
- H: Stands for "Healthy", referring to individuals who are not diagnosed with Alzheimer's Disease and serve as a control group.

2. Overview & Objectives

This project focuses on the application of advanced machine learning methodologies to predict Alzheimer's disease through handwriting analysis. Utilizing the DARWIN dataset, the study aims to improve predictive accuracy by addressing challenges such as high dimensionality and multicollinearity in the feature set.

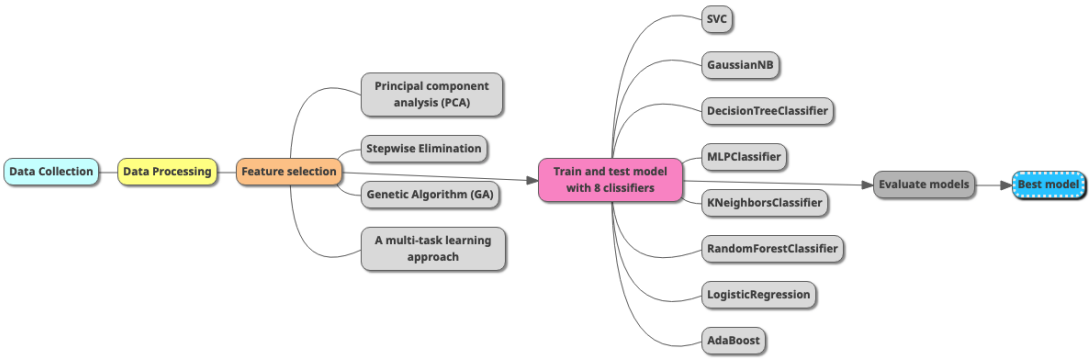
Various feature selection and modeling techniques, including Principal Component Analysis (PCA), stepwise selection, Genetic Algorithm (GA)-based selection, and Multi-Task Learning, were employed to identify the most relevant features and build robust classification models. The performance of different classifiers was evaluated based on accuracy, precision, and recall, with a specific focus on the recall metric due to the clinical importance of correctly identifying Alzheimer's patients.

1. **Enhance Predictive Accuracy for Alzheimer's Disease**: Develop and evaluate machine learning models to accurately classify handwriting data from Alzheimer's patients and healthy controls, with a particular emphasis on maximizing recall to ensure high sensitivity in identifying patients.
2. **Optimize Feature Selection**: Apply various feature selection techniques to reduce the high dimensionality of the dataset, focusing on preserving the most informative features while minimizing redundancy and multicollinearity.
3. **Evaluate Classifier Performance**: Compare the performance of multiple classifiers, including SVM, Naive Bayes, Decision Tree, MLP, KNN, Random Forest, and Logistic Regression, both individually and in ensemble configurations, to identify the best-performing model for the given task.
4. **Implement Cross-Validation**: Use cross-validation to assess the generalizability and robustness of the selected models, ensuring that they perform well on unseen data and are not overfitting to the training set.
5. **Utilize Ensemble Learning**: Leverage ensemble learning techniques to combine the strengths of different classifiers, aiming to improve overall model performance and stability.

By achieving these objectives, the project aims to contribute to the development of reliable and effective machine learning tools for early detection and diagnosis of Alzheimer's disease, potentially aiding in better patient outcomes and management.

3. Methods

This section outlines the methodology employed in our data analysis project, encompassing the steps involved in data collection, processing, feature selection, model training, and evaluation.



3.1. Data Collection

The project utilized publicly available datasets relevant to the research question.

3.2. Data Processing

The raw data underwent cleaning and preprocessing to ensure data consistency, accuracy, and suitability for analysis. This involved:

- The analysis utilizes a dataset of handwritten digits with 174 samples. Each sample represents a digit and contains 450 features related to pen stroke characteristics (e.g., airtime, pressure, speed, jerk).
- Remove the column ID.
- The target variable "class" indicates the digit being represented, with 'H' representing handwritten digits and 'P' representing pre-printed digits. This categorical variable is transformed into numerical values (0 for 'H' and 1 for 'P').
- The dataset is split into training and test sets with a ratio of 80:20, ensuring the model's ability to generalize to unseen data.

3.3. Feature Selection

To identify the most relevant features for model training, we employed various methods, including:

- **Principal component analysis (PCA):** PCA was used for dimensionality reduction, extracting principal components that captured most of the data's variance.
- **Stepwise elimination:** This method iteratively removed features with the least contribution to model performance until the optimal feature subset was identified.

- **Genetic algorithm (GA):** A genetic algorithm was used to search for the optimal feature subset by iteratively evaluating different combinations of features.
- **A multi-task learning approach:** This technique aimed to leverage the correlations between multiple tasks to select features that were relevant to all tasks.

3.4. Train and Test Model with 8 Classifiers

We utilized eight different classification algorithms to train and test our models. The algorithms included:

- **Support Vector Classifier (SVC):** A powerful algorithm that seeks to find the optimal hyperplane to separate data points.
- **Gaussian Naive Bayes:** A probabilistic classifier based on Bayes' theorem that assumes feature independence.
- **Decision Tree Classifier:** A tree-based model that divides the data into smaller subsets based on features, ultimately leading to a classification decision.
- **Multi-Layer Perceptron Classifier (MLPClassifier):** A neural network-based classifier that uses multiple layers of interconnected neurons to learn complex relationships in the data.
- **KNeighborsClassifier:** A non-parametric classifier that predicts the class of a data point based on the majority class among its k nearest neighbors.
- **RandomForestClassifier:** An ensemble method that combines multiple decision trees to improve generalization performance.
- **Logistic Regression:** A linear model that predicts the probability of a binary outcome.
- **AdaBoost:** An ensemble method that iteratively combines weak learners to create a strong classifier.

3.5. Evaluate Models

Each model was evaluated using appropriate metrics (e.g., accuracy, precision, recall, F1-score) to assess their performance.

3.6. Best Model

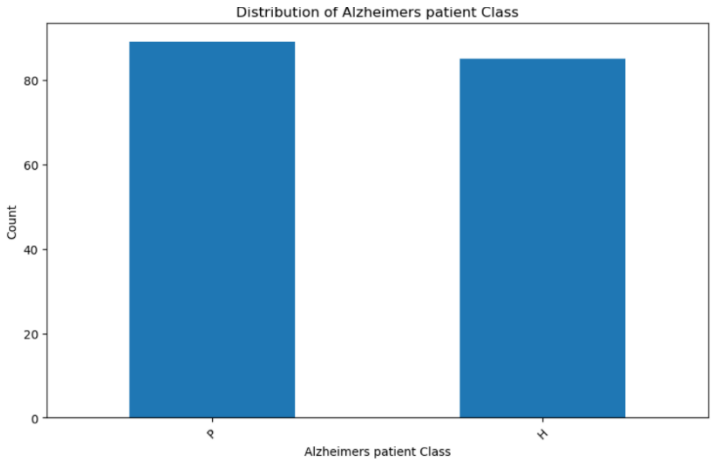
The best performing model, based on the chosen evaluation metric, was selected for further analysis and interpretation.

4. Exploratory data analysis (EDA)

4.1. Target variable

We found that the dataset is balanced on its target variable.

Number of subjects for each class	
Patient (P)	Healthy (H)
89	85

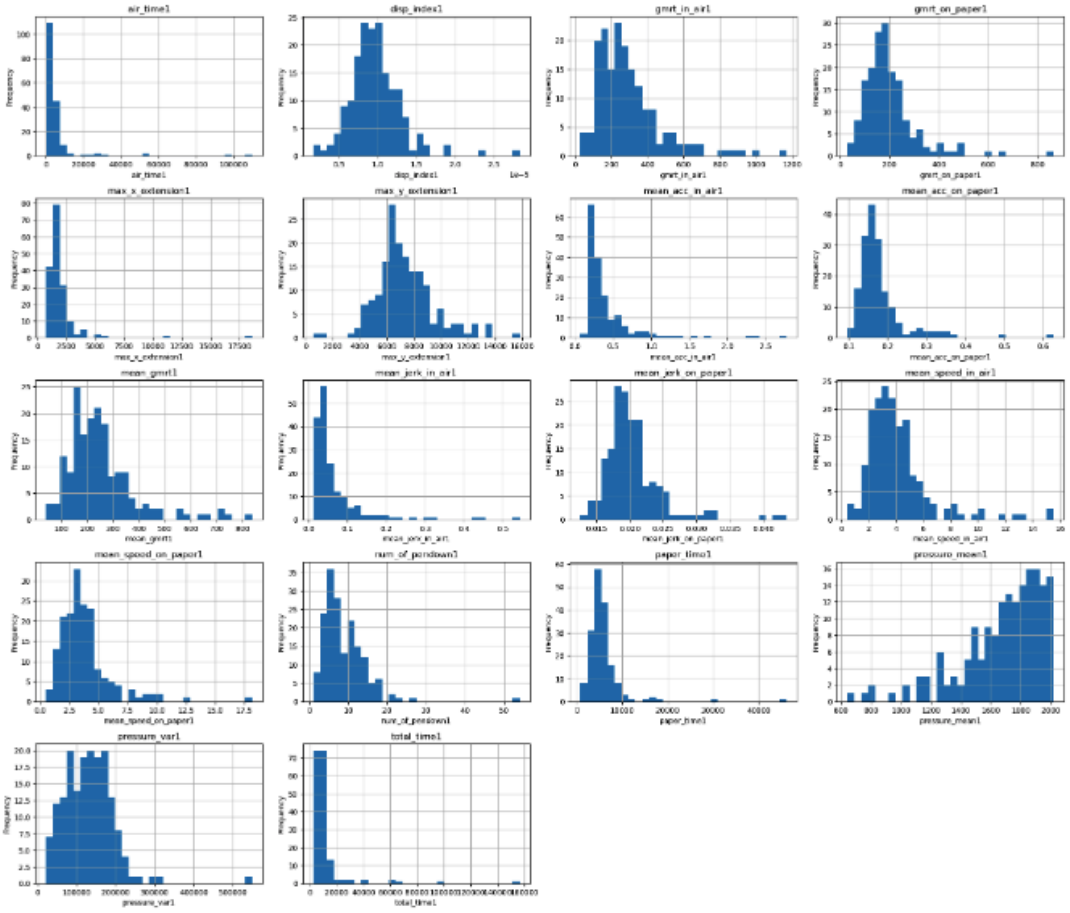


Since the data is balanced, we can base either macro or weighted model performance scores.

4.2. Numerical features

The dataset was collected when the patient performed 25 tasks, each task has the same 18 features. For learning purposes, we will perform the EDA on 1 task for the illustration.

We plotted several box plots to assess the distribution of each column corresponding to the first task.

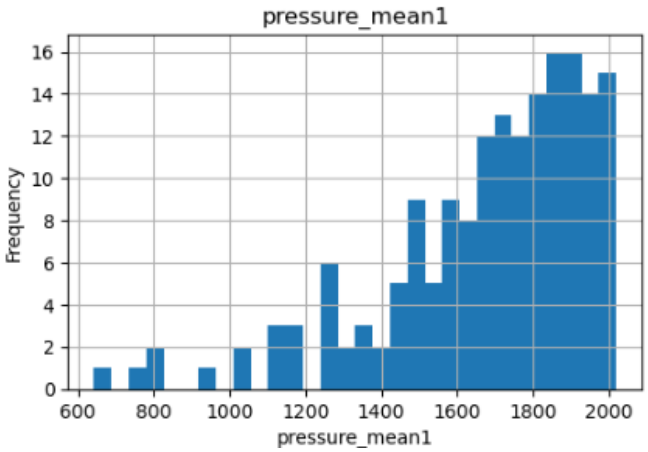


Let's look at a few features in more detail:

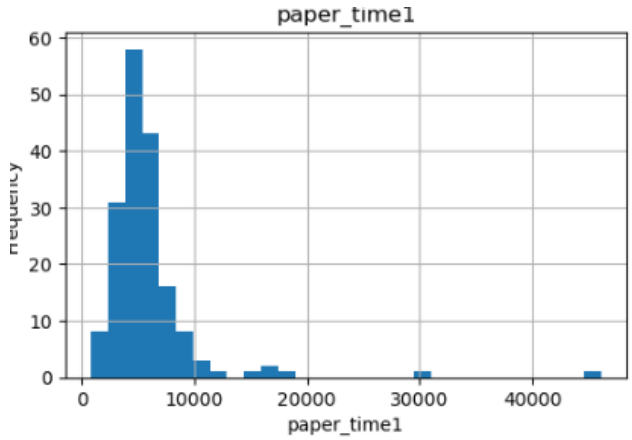
- **air_time1:** Right-skewed, suggesting most signatures have shorter air times. The few signatures with very long air times could be worth investigating further.
- **disp_index1:** This distribution is centered around a disp_index1 of 1.0, with some variability. This might indicate a relatively consistent displacement pattern across signatures.
- **gmrt_in_air1 & gmrt_on_paper1:** Both of these distributions are right-skewed, with gmrt_on_paper1 showing a more pronounced skew. This might mean that there's more variability in the speed and acceleration while the pen is on paper.
- **mean_jerk_in_air1 & mean_jerk_on_paper1:** Again, we see right skew in both, indicating that most signatures have lower jerk values (smoother movements).
- **pressure_mean1:** Shows a clear trend towards higher pressure values, indicating most signatures are written with a firm hand.
- **total_time1:** Heavily right-skewed, indicating that most signatures are completed relatively quickly.

Also, we studied any trend among the features, for example:

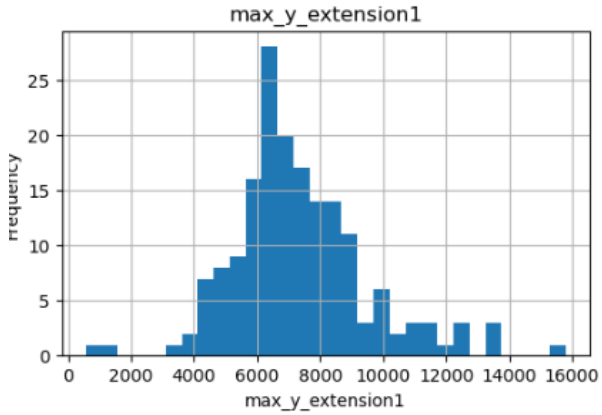
For task N.1:



From the plot, the ‘pressure mean’ variable is left-skewed.



Meanwhile, the ‘paper time’ is right-skewed.



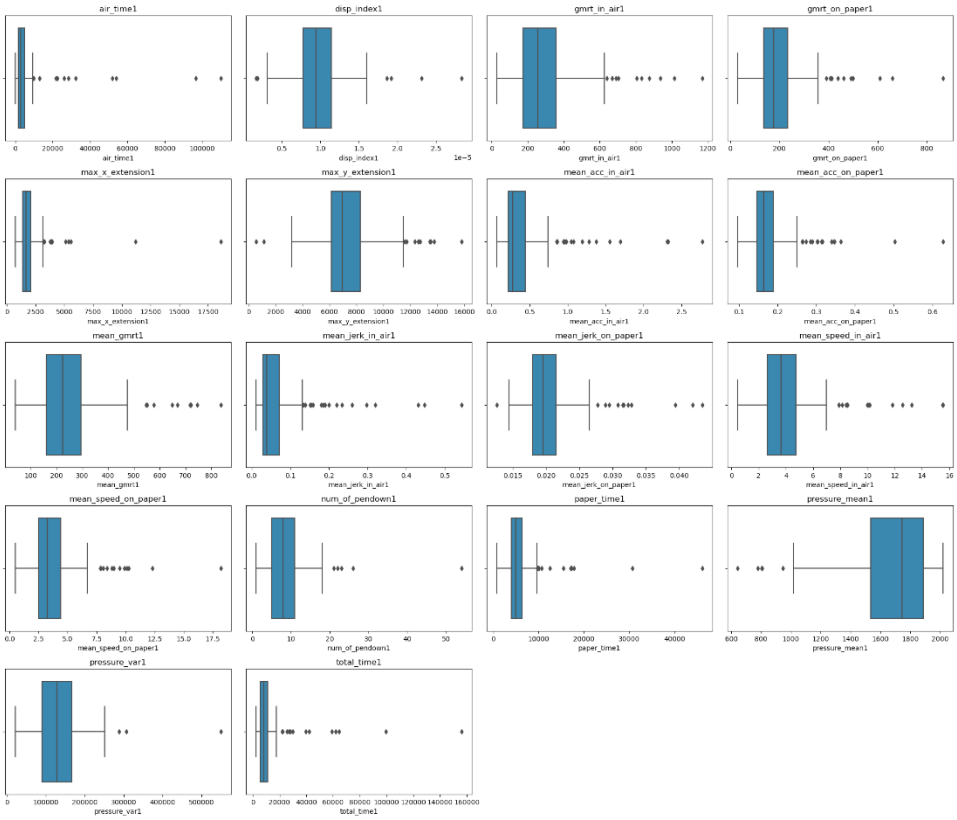
Also, we found a normal distribution in the ‘max y extension’ feature.

This pairplot provides a visual representation of the relationships between pairs of features in the dataset, differentiated by class (indicated by the blue and orange dots).

Linear Relationships

- **Strong Positive Linear Relationships:**
    - mean_speed_on_paper1 vs. mean_jerk_on_paper1: As the average jerk on paper increases, so does the average speed. This suggests faster, more abrupt movements on the paper.
    - mean_gmrt1 vs. mean_jerk_in_air1: Higher average jerk in the air corresponds to higher average gmrt (which often relates to speed and acceleration).
    - You might find other slightly weaker positive linear relationships by looking for upward-trending scatterplots.

- **Few Strong Negative Linear Relationships:** There aren't many visually evident strong negative linear relationships (downward-trending scatterplots) in your dataset.



Outlier Presence

Many of features show potential outliers, particularly:

- air_time1
- max_x_extension1
- max_y_extension1
- gmrt_in_air1
- mean_acc_in_air1
- mean_jerk_on_paper1
- total_time1

Data Skewness

Some boxplots indicate skewed data distributions:

- Right-Skewed: For example, air_time1, total_time1 - most values are clustered on the lower end with a long tail extending towards higher values.

Potential Data Issues

The presence of many outliers in some features could suggest:

- Measurement errors
- Natural variability in your data
- Small sample size

Here's the table of pairs that have correlations Greater Than 0.7.

	mean_acc_on_paper1	mean_gmrt1	mean_jerk_in_air1	mean_jerk_on_paper1	mean_speed_in_air1	mean_speed_on_paper1	paper_time1	total_time1
air_time1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.972902
disp_index1	NaN	NaN	NaN	NaN	NaN	NaN	0.808375	NaN
gmrt_in_air1	NaN	0.940325	NaN	NaN	0.826749	NaN	NaN	NaN
gmrt_on_paper1	0.875478	0.828012	NaN	0.738690	NaN	0.985557	NaN	NaN
mean_acc_in_air1	NaN	NaN	0.988005	NaN	NaN	NaN	NaN	NaN
mean_acc_on_paper1	NaN	NaN	NaN	0.886071	NaN	0.896524	NaN	NaN
mean_gmrt1	NaN	NaN	NaN	NaN	0.863950	0.822041	NaN	NaN
mean_jerk_on_paper1	NaN	NaN	NaN	NaN	NaN	0.749382	NaN	NaN
num_of_pendown1	NaN	NaN	NaN	NaN	NaN	NaN	0.726058	NaN
paper_time1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.757173

Some of the 18 features in Task 1 are highly correlated with each other, but none of the 18 features in Task 1 are highly correlated with the features in Tasks 2, 3,... or 25. The features are highly correlated within each task only.

Similar Pairs Across Different Tasks

1. **air_time & total_time:** These pairs are likely measuring time-related metrics, possibly in different contexts or phases of tasks (e.g., total time vs. specific time in the air).
2. **disp_index & paper_time**: Could be related to indices or metrics related to paper tasks or measurements taken on paper.
3. **gmrt_in_air & mean_gmrt:** Seem to relate to metrics involving GMRT (possibly Global Mean Response Time), measured either in the air or as an average across tasks.
4. **gmrt_on_paper & mean_speed_on_paper:** gmrt_on_paper and mean_speed_on_paper might indicate GMRT or speed-related metrics specifically measured or averaged on paper.
5. **mean_acc_in_air & mean_jerk_in_air:** mean_acc_in_air and mean_jerk_in_air likely represent mean acceleration and mean jerk measured during tasks in the air, indicating movement or dynamic metrics.
6. **mean_acc_on_paper & mean_jerk_on_paper:** mean_acc_on_paper and mean_jerk_on_paper similarly suggest mean acceleration and mean jerk metrics, but specifically measured or averaged during tasks on paper.

Missing Values

The data doesn't have any missing values.

5. Machine Learning Algorithms

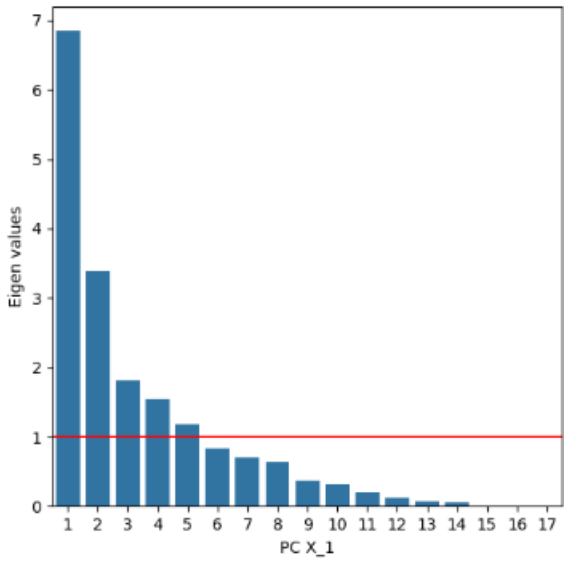
From this point, we will introduce different approaches the data analytics team performed over the dataset to obtain the best classification model for the 'DARWIN' dataset.

The election was based on the 'Recall' score since the dataset is medical clinical type.

5.1. Principal Component Analysis (PCA)

For this step, we prepared the data by scaling and then for each 18 set of features we extracted the many principal components that accomplish the kaiser criteria for principal components selection.

For the first set of 18 features or Task #1, we select 5 principal components as the following graph shows for the set of features of Task N.1:



5.1.1. Principal components

The result was as follows:

Number of Columns	
Original dataset	Kaiser criteria
450	139

5.1.2. Ensemble learning

We trained 7 different and popular classification models, plus an AdaBoost classifier based on the 7 mentioned models.

5.1.2.1. Evaluating models

The performance metrics of each classifier and AdaBoost were evaluated and compared.

Classifier	Accuracy	Precision	Recall
SVM	0.86	0.86	0.86
Naive Bayes	0.69	0.76	0.69
Decision Tree	0.71	0.71	0.71
MLP	0.83	0.86	0.83
KNN	0.83	0.86	0.83
Random Forest	0.89	0.90	0.89
Logistic Regression	0.91	0.91	0.91
AdaBoost	0.86	0.86	0.86

5.1.3. Cross-validation

Classifier	k	Recall
SVM	2	0.74
	5	0.73
	10	0.77
Naive Bayes	2	0.80
	5	0.81
	10	0.80
Decision Tree	2	0.68
	5	0.73
	10	0.75
MLP	2	0.79
	5	0.73
	10	0.75
KNN	2	0.56
	5	0.63
	10	0.67
Random Forest	2	0.81
	5	0.81
	10	0.83
Logistic Regression	2	0.75
	5	0.75
	10	0.77
AdaBoost	2	0.72
	5	0.76
	10	0.86

5.1.3. Best Classifier: Random Forest

Metric	Value
Accuracy	0.89
Precision	0.90
Recall	0.89
Cross-Validation (k=2) Recall	0.81
Cross-Validation (k=5) Recall	0.81
Cross-Validation (k=10) Recall	0.83

5.1.4.1. Confusion Matrix

	Predicted Patient (P)	Predicted Healthy (H)
True Patient (P)	10	1
True Healthy (H)	3	21

5.1.4.2. Classification Report

Class	Precision	Recall	F1-Score	Support
Healthy (H)	0.95	0.88	0.91	24
Patient (P)	0.77	0.91	0.83	11
Accuracy			0.89	35
Macro Avg	0.86	0.89	0.87	35
Weighted Avg	0.90	0.89	0.89	35

5.1.5. Conclusion

We applied PCA to our dataset to reduce its dimension. After applying different classifiers, we found that the most stable and appropriate model for this data was **Random Forest**, finding an accuracy of 89%, we also noticed that including Cross-Validation we could find a very high value of recall 81%, 81%, and 83% for multiple k=2, k=5, and k=10, respectively.

5.2. Stepwise Elimination

In this step, we tried Backward Elimination, Forward Selection, and Stepwise Selection. Based on the results from each method, we proceeded with further analysis.

5.2.1. Selection Method

Forward Selection: Begins with no predictors in the model and adds the most significant features one at a time. At each step, the feature that improves the model the most (based on a chosen criterion like p-value, AIC, or model performance) is added. This process continues until no additional features significantly improve the model.

Backward Elimination: It is a method of feature selection that starts with all potential predictors in the model and removes the least significant features one by one. The feature with the highest p-value (indicating the least statistical significance) is removed at each step, and the model is refitted. This process continues until all remaining features are statistically significant.

Stepwise Selection: It is a combination of forward selection and backward elimination. It starts with an empty model and adds features one by one like forward selection. However, after adding each feature, it also considers removing any features that no longer contribute significantly to the model. This method aims to find a balance between including significant features and excluding redundant ones.

5.2.2. Number of features

Feature Selection Functions:

- **forward_selection:** Starts with an empty set of features. Iteratively adds features one at a time based on the lowest p-value (significance) in a logistic regression model.
- **backward_selection:** Starts with all features. Iteratively removes features one at a time based on the highest p-value.
- **stepwise_selection:** Combines both forward and backward steps. It first performs forward selection, then iteratively checks if any feature can be removed without significantly impacting the model.

Method	Number of features
Forward Selection	79
Backward Elimination	450
Stepwise Selection	9

Given that **stepwise selection** selected only 9 features, it indicates its efficiency in identifying the most relevant features while avoiding redundancy. Therefore, we will proceed with stepwise selection due to its comprehensive approach and ability to streamline the feature set significantly. However, if any of the classifiers using this selected feature set shows high accuracy, we will also consider trying forward selection, which retains 79 features, to potentially enhance the model's performance.

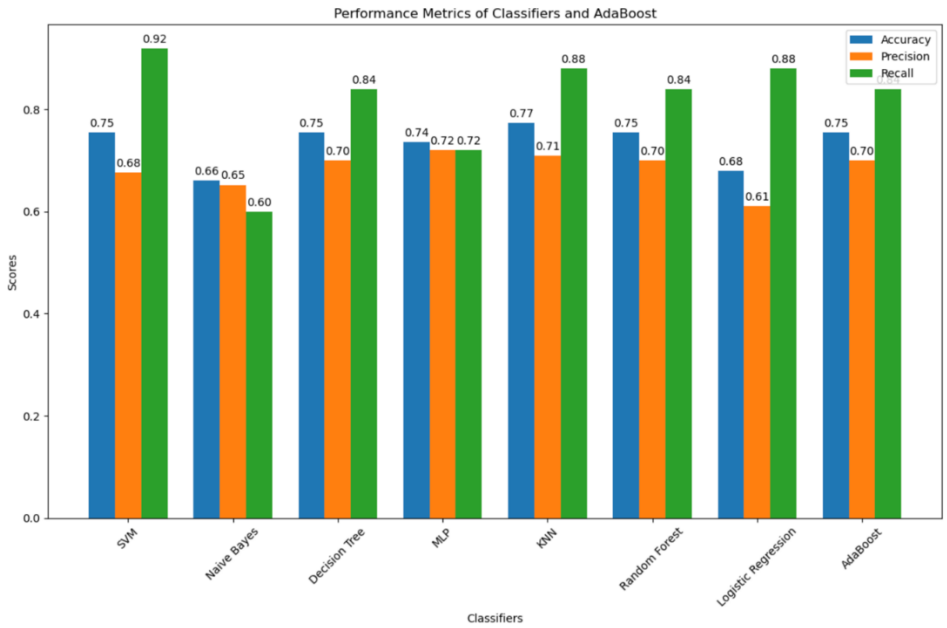
Selected features: [total_time11, max_x_extension22, gmrt_in_air23, mean_acc_on_paper9, num_of_pendown19, pressure_mean12, total_time17, max_y_extension23, total_time23]

5.2.3. Ensemble learning

5.2.3.1. Evaluating models

The performance metrics of each classifier and AdaBoost were evaluated and compared.

Classifier	Accuracy	Precision	Recall
SVM	0.75	0.68	0.92
Naive Bayes	0.66	0.65	0.60
Decision Tree	0.75	0.70	0.84
MLP	0.74	0.72	0.72
KNN	0.77	0.71	0.88
Random Forest	0.75	0.70	0.84
Logistic Regression	0.68	0.61	0.88
AdaBoost	0.75	0.70	0.84

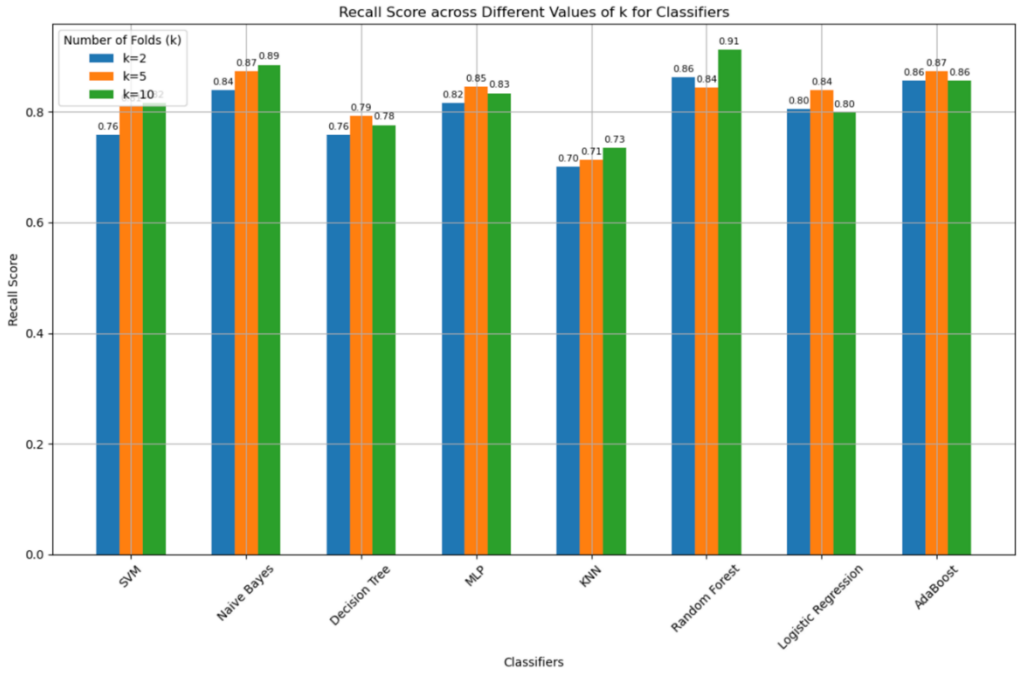


Key Observations:

- **SVM** is the classifier that most recall shows by far, having a 92%
- **None** of the classifiers reach an accuracy of more than 80%.

5.2.3.2. Cross-Validation

Classifier	k	Recall
SVM	2	0.76
	5	0.81
	10	0.82
Naive Bayes	2	0.84
	5	0.87
	10	0.89
Decision Tree	2	0.76
	5	0.79
	10	0.78
MLP	2	0.82
	5	0.85
	10	0.83
KNN	2	0.70
	5	0.71
	10	0.73
Random Forest	2	0.86
	5	0.84
	10	0.91
Logistic Regression	2	0.80
	5	0.84
	10	0.80
AdaBoost	2	0.86
	5	0.87
	10	0.86

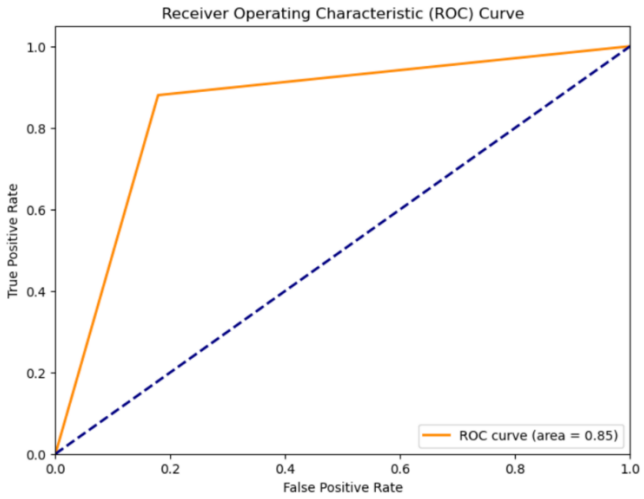


Key Takeaways:

- **Random Forest:** Shows the best results for Recall having a maximum for k=10 of 91%.
- **AdaBoost & Naïve Bayes:** Also shows good consistency and good results for recalls for each of the 'k's.
- **KNW:** Is the lowest of the classifiers having a mean result of 71.3%.}

5.2.4. Best Classifier: Random Forest

Metric	Value
Accuracy	0.75
Precision	0.70
Recall	0.84
Cross-Validation (k=2) Recall	0.86
Cross-Validation (k=5) Recall	0.84
Cross-Validation (k=10) Recall	0.91



The ROC curve for the Random Forest classifier, with an area under the curve (AUC) of 0.85, also highlights the model's overall effectiveness in distinguishing between positive and negative cases.

5.2.4.1. Confusion Matrix

	Predicted Patient (P)	**Predicted Healthy (H)**
True Patient (P)	23	5
True Healthy (H)	3	22

5.2.4.1. Classification Report

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Healthy (H)	0.88	0.82	0.85	28
Patient (P)	0.81	0.88	0.85	25
Accuracy			0.85	53
Macro Avg	0.85	0.85	0.85	53
Weighted Avg	0.85	0.85	0.85	53

5.3.5. Summary

The stepwise selection process was applied to our dataset to identify the most relevant features that significantly contribute to the predictive power of our model. This method proved to be highly effective in reducing the dimensionality of our data from an initial set of 450 features to 9.

After applying different classifiers, we found that the most stable and appropriate model for this data was **Random Forest**, finding an accuracy of 85%, we also noticed that including Cross-Validation we could find a very high value of recall, with k=10 we reached a 91%.

5.3. Genetic Algorithm (GA)

The Genetic Algorithm (GA) was selected for feature selection in the context of detecting Alzheimer's disease based on handwriting data. This approach was chosen due to its ability to effectively handle high-dimensional data and optimize feature subsets to improve classifier performance. The GA helps in selecting the most relevant features, which are then used to train various classifiers. Below is a detailed explanation of the modeling process, including data preparation, the implementation of the GA, and the evaluation of the models.

5.3.1. Feature Ranking (Filter Method)

The initial phase of the process involved ranking the features using two filter methods: ReliefF and FCBF.

- **ReliefF:** Evaluates the importance of features by considering differences between feature values of nearest neighbor instances. The top 30% of features based on their ReliefF scores were selected.
- **FCBF:** Ranks features based on their consistency and redundancy. Similarly, the top 30% of features based on FCBF scores were selected.

The intersection of features selected by both methods was used for further analysis, ensuring that only the most relevant and non-redundant features were retained.

5.3.2. GA-Based Feature Selection (Wrapper Method)

The GA-based feature selection process was implemented as follows:

1. **Genotype Representation:** Each feature was encoded as a binary gene (1 for selected, 0 for not selected). A chromosome represented a potential feature subset.
2. **Population Initialization:** An initial population of 50 chromosomes was created, each representing a different feature subset.
3. **Fitness Function:** The fitness of each chromosome was evaluated using the classification accuracy of various classifiers (e.g., RandomForest, SVM, Naive Bayes, Decision Tree, MLP, KNN, Logistic Regression). The accuracy score on a validation set served as the fitness value.
4. **Genetic Operators:**
    - **Selection:** Rank-based selection was used to select parent chromosomes for reproduction, ensuring that chromosomes with better fitness scores had a higher chance of passing their genes to the next generation.
    - **Crossover:** One-point crossover was applied to combine genetic material from two parent chromosomes, promoting the exchange of beneficial features.
    - **Mutation:** A single-point mutation was introduced to flip a gene (0 to 1 or 1 to 0) in a chromosome with a probability of 0.02, maintaining genetic diversity.
5. **Iteration:** Fitness evaluation and genetic operator application were repeated for 30 generations.
6. **Stopping Criteria:** The GA was terminated when the best fitness score remained constant for a specified number of generations.
7. **Feature Subset:** The chromosome with the highest fitness score after the GA process represented the optimal feature subset.

5.3.2.1. Selection of GA Parameters

The parameters for the GA were carefully chosen to balance exploration and exploitation within the feature space:

- **Population Size (50):** A population size of 50 was selected to ensure a diverse set of feature subsets while maintaining computational efficiency. A larger population provides greater diversity but increases the computational load.
- **Number of Generations (30):** The algorithm was run for 20 generations, allowing sufficient iterations for the population to evolve and converge towards optimal feature subsets. Too few generations may not allow the algorithm to fully explore the feature space, while too many could lead to overfitting or excessive computation time.
- **Crossover Rate (0.8):** A crossover rate of 0.8 was chosen to promote the exchange of genetic material between parent chromosomes. This high rate ensures that most offspring inherit a mix of features from both parents, facilitating the discovery of beneficial feature combinations.
- **Mutation Rate (0.02):** A mutation rate of 0.02 was selected to introduce occasional random changes in the chromosomes. This low rate helps maintain genetic diversity without disrupting the overall structure of the feature subsets too frequently.

5.3.2.2. Classifier Performance and Fitness Scores

Classifier	Mean Fitness Score	Selected Features Count	Selected Features
MLP	0.9143	23	[3, 4, 5, 7, 8, 14, 15, 18, 19, 20, 24, 26, 27, 29, 32, 33, 34, 41, 44, 45, 46, 47, 49]
Random Forest	0.9143	29	[1, 2, 4, 5, 7, 9, 11, 12, 18, 19, 20, 21, 23, 26, 27, 28, 29, 32, 33, 36, 37, 38, 40, 41, 42, 43, 45, 46, 48]
Decision Tree	0.8952	21	[0, 4, 5, 7, 9, 10, 12, 18, 22, 23, 25, 28, 29, 30, 33, 36, 37, 38, 40, 42, 44]
Naive Bayes	0.8571	29	[1, 2, 3, 7, 9, 10, 11, 14, 17, 18, 19, 21, 24, 25, 26, 27, 28, 29, 33, 37, 39, 40, 42, 43, 45, 46, 47, 48, 49]
SVM	0.8000	24	[0, 1, 3, 5, 7, 13, 14, 15, 19, 21, 25, 26, 28, 29, 30, 32, 33, 35, 38, 39, 42, 43, 45, 48]
Logistic Regression	0.8000	25	[0, 1, 5, 9, 11, 12, 13, 14, 15, 17, 18, 24, 25, 28, 30, 31, 33, 35, 37, 38, 39, 41, 44, 45, 49]
KNN	0.6857	26	[1, 2, 3, 5, 6, 8, 9, 10, 13, 15, 17, 20, 22, 24, 25, 26, 27, 28, 30, 31, 33, 35, 36, 38, 46, 48]

Best set of features:

- **Classifier**: MLP
- **Mean Fitness Score**: 0.9143
- **Number of Features**: 23

- **Selected Feature Names**: ['gmrt_on_paper1', 'max_x_extension1', 'max_y_extension1', 'mean_acc_on_paper1', 'mean_gmrt1', 'paper_time1', 'pressure_mean1', 'air_time2', 'disp_index2', 'gmrt_in_air2', 'mean_acc_in_air2', 'mean_gmrt2', 'mean_jerk_in_air2', 'mean_speed_in_air2', 'paper_time2', 'pressure_mean2', 'pressure_var2', 'max_y_extension3', 'mean_gmrt3', 'mean_jerk_in_air3', 'mean_jerk_on_paper3', 'mean_speed_in_air3', 'num_of_pendown3']

5.3.3. Ensemble Learning

To further enhance model performance, ensemble learning was employed by combining the strengths of multiple classifiers using the selected features from GA-based feature selection.

5.3.3.1. Training Basic Learners

Several classifiers were trained using the selected features:

- Support Vector Machine (SVM)
- Naive Bayes (NB)
- Decision Tree
- Multi-Layer Perceptron (MLP)
- K-Nearest Neighbors (KNN)
- Random Forest Classifier (RFC)
- Logistic Regression (LR)

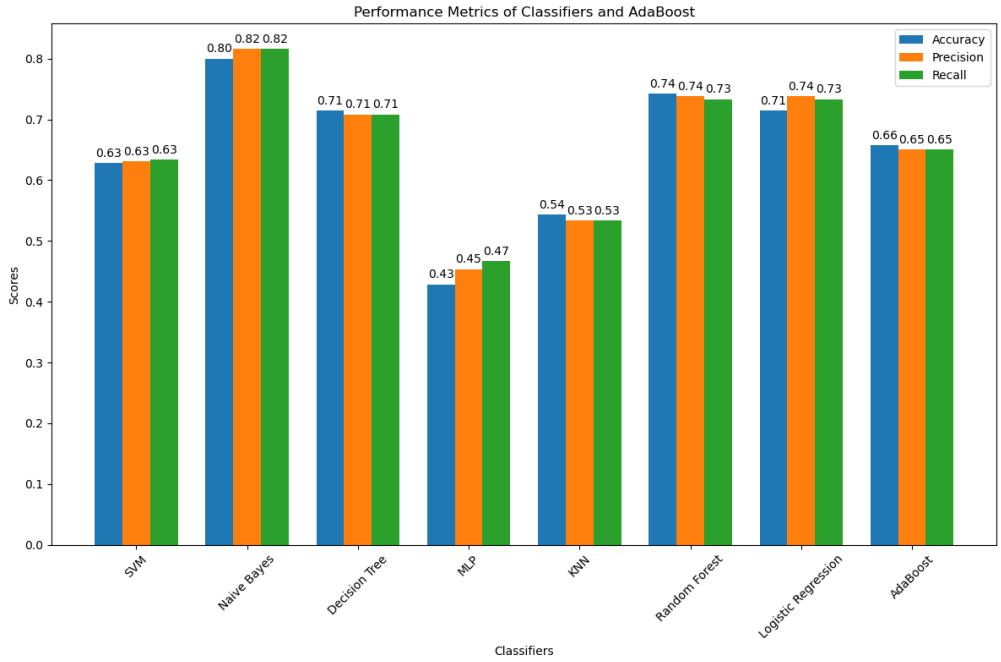
5.3.3.2. Meta-Learner (AdaBoost)

AdaBoost was used as a meta-learner, taking the predictions of the Decision Tree classifier as input to improve overall accuracy.

5.3.3.3. Evaluation

The performance metrics of each classifier and AdaBoost were evaluated and compared.

Classifier	Accuracy	Precision	Recall
SVM	0.63	0.63	0.63
Naive Bayes	0.80	0.82	0.82
Decision Tree	0.71	0.71	0.71
MLP	0.45	0.47	0.47
KNN	0.54	0.53	0.53
Random Forest	0.74	0.74	0.73
Logistic Regression	0.71	0.74	0.73
AdaBoost	0.66	0.65	0.65



General Observations:

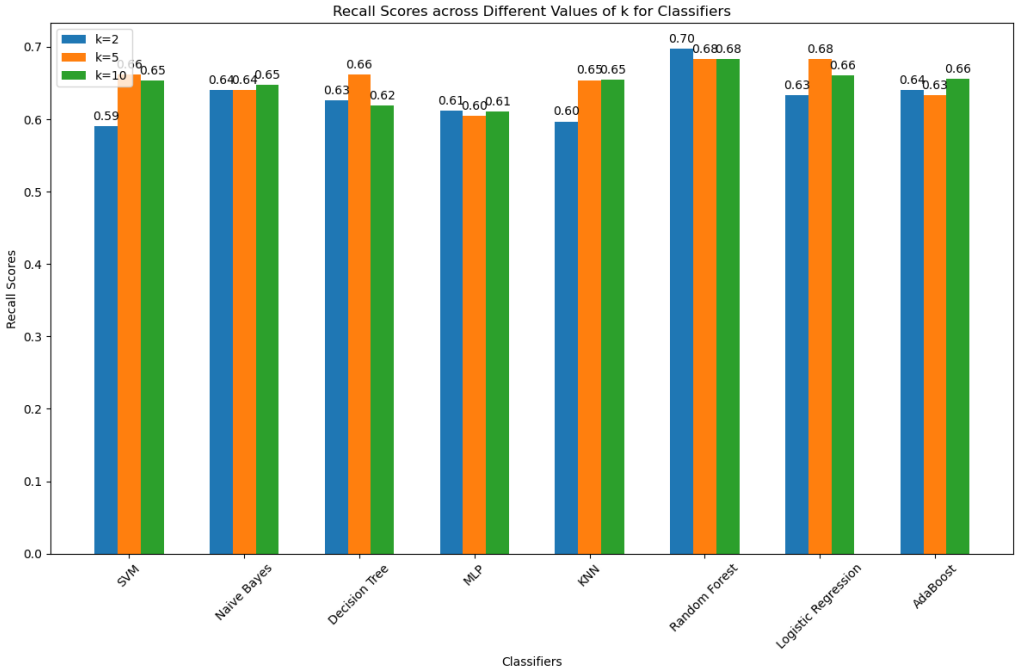
- Random Forest and Naive Bayes are consistently the best performers across all metrics.
- Logistic Regression and AdaBoost also show strong and reliable performance.
- SVM, Decision Tree, KNN, and MLP are less consistent, with KNN and MLP being the least effective classifiers in this analysis.

5.3.4. Cross-Validation

To ensure the robustness of our model, we performed cross-validation by splitting the data into different folds and training the model on each fold. This helps in assessing the model's performance on unseen data and prevents overfitting. Cross-validation with k=2, 5, and 10 was performed to comprehensively evaluate the model's performance and generalizability.

Classifier	k	Recall
SVM	2	0.590
	5	0.661
	10	0.654
Naive Bayes	2	0.640
	5	0.640
	10	0.648
Decision Tree	2	0.626
	5	0.662
	10	0.619
MLP	2	0.612
	5	0.604
	10	0.610

KNN	2	0.597
	5	0.654
	10	0.655
Random Forest	2	0.698
	5	0.683
	10	0.683
Logistic Regression	2	0.633
	5	0.683
	10	0.661
AdaBoost	2	0.640
	5	0.633
	10	0.655



Key Takeaways:

- **Random Forest and AdaBoost** maintain high recall scores across different kkk values, showcasing their effectiveness in identifying true positives.
- **Naive Bayes** also performs well with consistent recall scores, making it a reliable choice.
- **SVM, Decision Tree, MLP, and KNN** show lower and less consistent recall scores, with MLP and KNN being the least effective.

5.3.5. Best Classifier: Random Forest

Random Forest was chosen as the best classifier due to its consistently high performance across all key metrics: accuracy, precision, and recall. It outperformed other classifiers, demonstrating superior ability to correctly classify both 'P' and 'H' samples, as shown in the performance metrics graph. This indicates its robustness and effectiveness in handling the classification task.

Additionally, Random Forest exhibited remarkable stability during cross-validation, maintaining high recall scores across different values of k (2, 5, and 10 folds). This consistency across various validation sets highlights its strong generalization capability, ensuring reliable performance on unseen data. Its balanced and robust performance makes Random Forest the most reliable and effective choice for this classification task.

Metric	Value
Accuracy	0.74
Precision	0.74
Recall	0.73
Cross-Validation (k=2) Recall	0.698
Cross-Validation (k=5) Recall	0.669
Cross-Validation (k=10) Recall	0.683

The final model achieved an accuracy of 0.8 (80%) using the top features selected by the Genetic Algorithm (GA). This indicates that the model correctly classified 80% of the instances in the test set.

5.3.5.1. Confusion Matrix

The confusion matrix provides a detailed breakdown of the model's performance:

	Predicted Patient (P)	Predicted Healthy (H)
True Patient (P)	14	1
True Healthy (H)	6	14

- **True Healthy (H)**: Represents the individuals correctly identified as healthy.
- **True Patient (P)**: Represents the individuals correctly identified as patients.
- **False Healthy (H)**: Represents the individuals incorrectly identified as healthy (actual patients).
- **False Patient (P)**: Represents the individuals incorrectly identified as patients (actual health y individuals).

In this case:

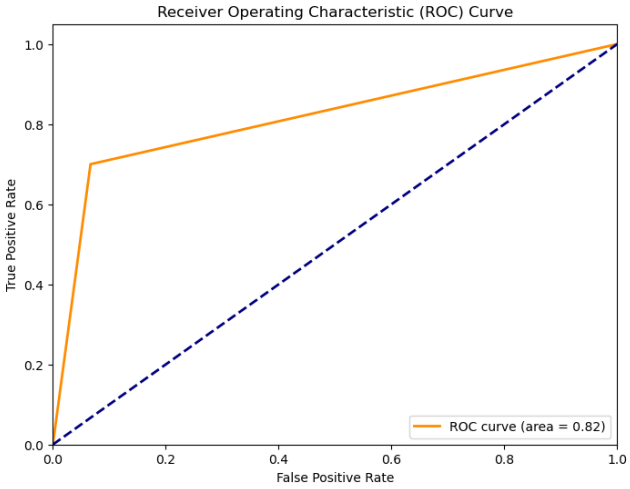
- 14 healthy individuals were correctly classified as healthy.
- 14 patients were correctly classified as patients.
- 6 patients were incorrectly classified as patients.
- 1 healthy individual was incorrectly classified as healthy.

5.3.5.2. Classification Report

The classification report provides detailed performance metrics for each class (Healthy and Patient):

Class	Precision	Recall	F1-Score	Support
Healthy (H)	0.70	0.93	0.80	15
Patient (P)	0.93	0.70	0.80	20
Accuracy			0.80	35
Macro Avg	0.82	0.82	0.80	35
Weighted Avg	0.83	0.80	0.80	35

- **Precision**:
    - Healthy (H): 0.70 (70% of the predicted healthy individuals were actually healthy).
    - Patient (P): 0.93 (93% of the predicted patients were actually patients).
- **Recall**:
    - Healthy (H): 0.93 (93% of the actual healthy individuals were correctly identified).
    - Patient (P): 0.70 (70% of the actual patients were correctly identified).
- **F1-Score**: The harmonic mean of precision and recall, providing a single metric for evaluation.
    - Healthy (H): 0.80
    - Patient (P): 0.80
- **Accuracy:** The overall accuracy of the model, calculated as (True Positives + True Negatives) / Total Instances.
    - 0.80 (80%)
- **Macro Average:** The average of precision, recall, and F1-score, treating all classes equally.
    - Precision: 0.82
    - Recall: 0.82
    - F1-Score: 0.80
- **Weighted Average:** The average of precision, recall, and F1-score, weighted by the number of instances of each class.
    - Precision: 0.83
    - Recall: 0.80
    - F1-Score: 0.80



5.3.6. Summary

Our approach employed Genetic Algorithm (GA) for feature selection in detecting Alzheimer's disease from handwriting data, leveraging its ability to handle high-dimensional data and optimize feature subsets. Initially, we ranked features using ReliefF and FCBF methods, retaining the most relevant features for further analysis. The GA-based feature selection then refined these features through iterative processes involving selection, crossover, and mutation to identify the optimal subset, evaluated via various classifiers.

The final model, primarily evaluated using **Random Forest**, achieved an accuracy of 80%, demonstrating high precision (0.93) but lower recall (0.70) for patient identification, indicating some misclassifications. Cross-validation confirmed the robustness of the model, highlighting the effectiveness of our combined feature selection methods and ensemble learning. This comprehensive approach underscores the importance of strategic feature selection and the utility of GA in enhancing machine learning model performance.

5.4. A multi-task learning approach

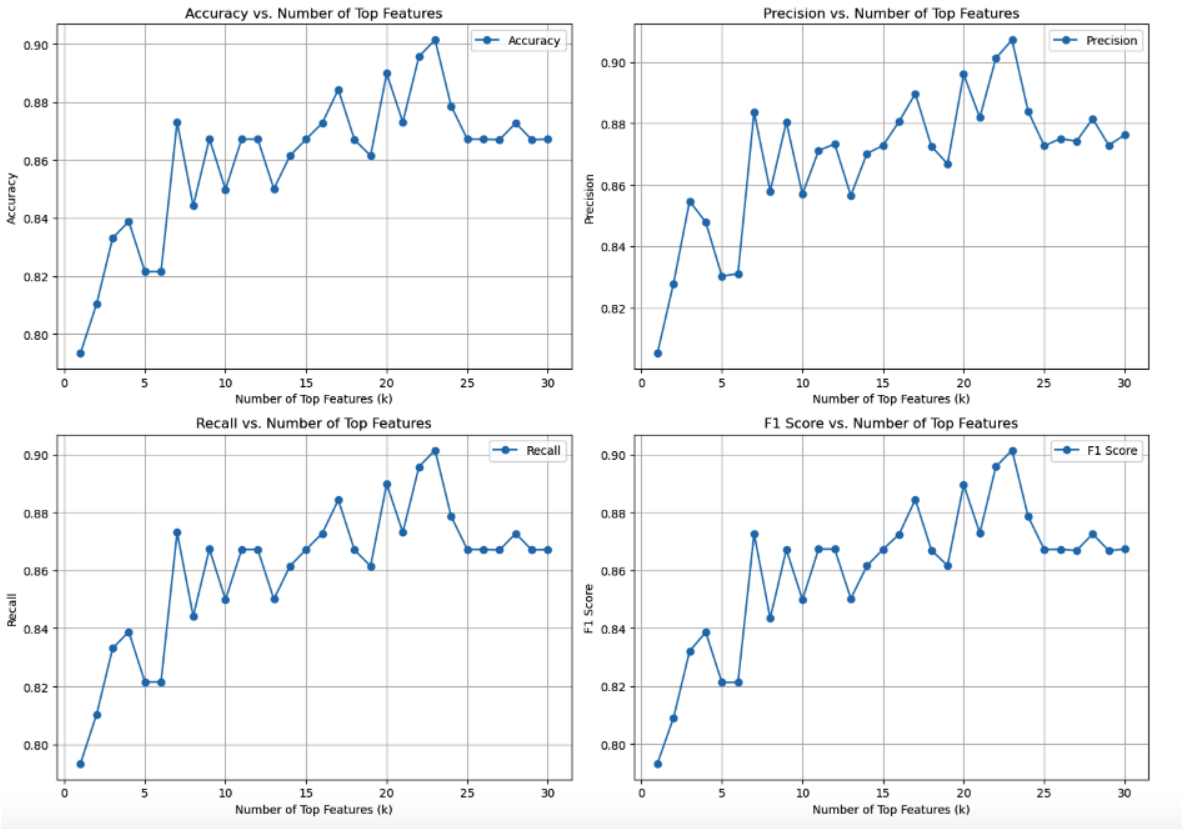
A multi-task learning framework is employed to identify the most relevant features for handwritten digit recognition. The framework involves the following steps:

- Splitting into tasks: The 450 features are divided into 25 tasks, each containing 18 features representing pen stroke characteristics for a specific digit.
- Training classifiers for each task: A Random Forest classifier is trained on each task, yielding predictions for the test set.
- Combining predictions: The predictions from each task are combined through majority voting, resulting in a final prediction for each sample. The combined model achieves an accuracy of 82.86%, indicating a successful combination of task-specific knowledge.
- Feature selection: Feature importance from each task is combined to create an aggregated importance score for each feature across all tasks. The top 22 most important features are selected.
- Training a final model: A Random Forest classifier is trained on the training data using only the 22 most important features. This model achieves an accuracy of 88.57%, demonstrating the effectiveness of the feature selection approach.

5.4.1. Select the best features.

The dataset was divided into 25 separate tasks, each representing pen stroke characteristics for a specific digit (1-25). For each task, a Random Forest classifier was trained. The feature_importances_ attribute of the trained Random Forest model provided a score for each feature within that task, indicating its importance in predicting the digit class.

The feature importance from each task were aggregated, effectively creating a score for each feature across all 25 tasks. This score represents the feature's overall importance in recognizing handwritten digits across the entire dataset.



The analysis explored using a range of top features (1 to 30) to see how accuracy varied with the number of features. A plot of accuracy versus the number of features revealed that the accuracy plateaued after approximately 22 features. This suggests that adding more features beyond the top 22 didn't significantly improve the model's performance.

Top 22 features for the whole dataset: ['total_time23', 'total_time6', 'total_time9', 'total_time15', 'air_time15', 'air_time16', 'total_time13', 'air_time6', 'air_time23', 'total_time3', 'total_time24', 'air_time5', 'total_time7', 'air_time13', 'air_time24', 'total_time2', 'air_time17', 'air_time22', 'pressure_mean4', 'total_time16', 'max_y_extension20', 'total_time12'].

5.4.2. Ensemble learning

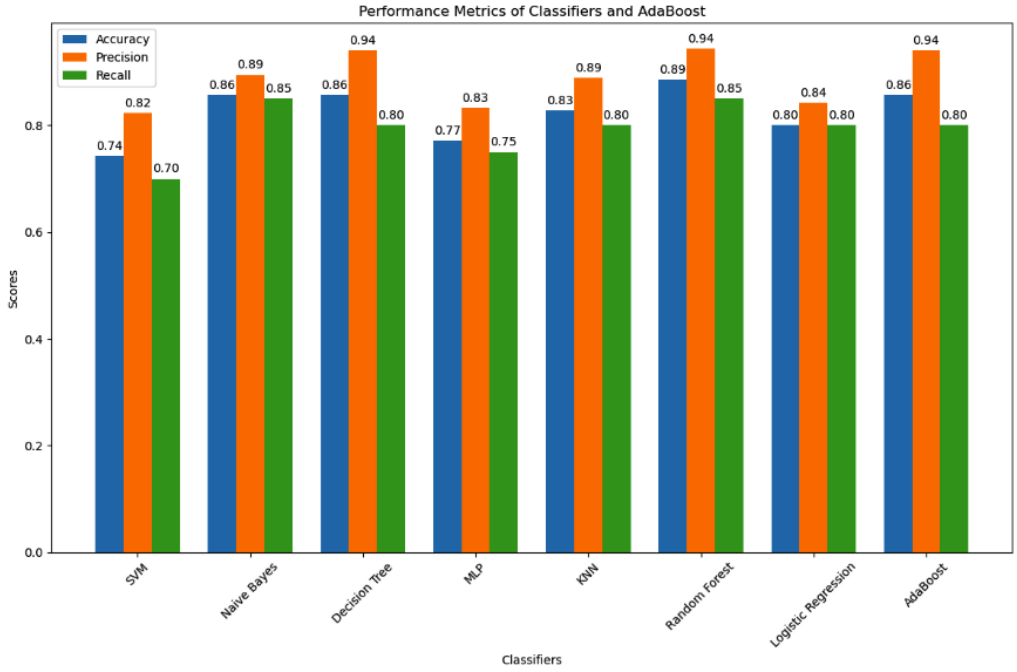
To further enhance the performance, an ensemble learning approach is employed. This involves:

- **Basic learners:** Different classifiers (SVM, Naive Bayes, Decision Tree, MLP, KNN, Random Forest, Logistic Regression) are trained on the selected feature subset.
- **Meta-learner:** AdaBoost is used as the meta-learner, combining the predictions from the basic learners to create a final prediction.

The ensemble model is evaluated using cross-validation with varying numbers of folds (k=2, 5, and 10). The results indicate that AdaBoost consistently achieves a high recall score across different k values, suggesting its robust performance in recognizing handwritten digits.

5.4.3. Evaluation

Classifier	Accuracy	Precision	Recall	F1-Score
SVM	0.8286	0.8571	0.8571	0.8571
Naive Bayes	0.8571	0.9091	0.8182	0.8621
Decision Tree	0.8286	0.8571	0.8571	0.8571
MLP	0.8571	0.9000	0.8182	0.8571
KNN	0.7429	0.7500	0.7500	0.7500
Random Forest	0.8857	0.9412	0.8500	0.8936
Logistic Regression	0.8571	0.9091	0.8182	0.8621
AdaBoost	0.8857	0.9412	0.8500	



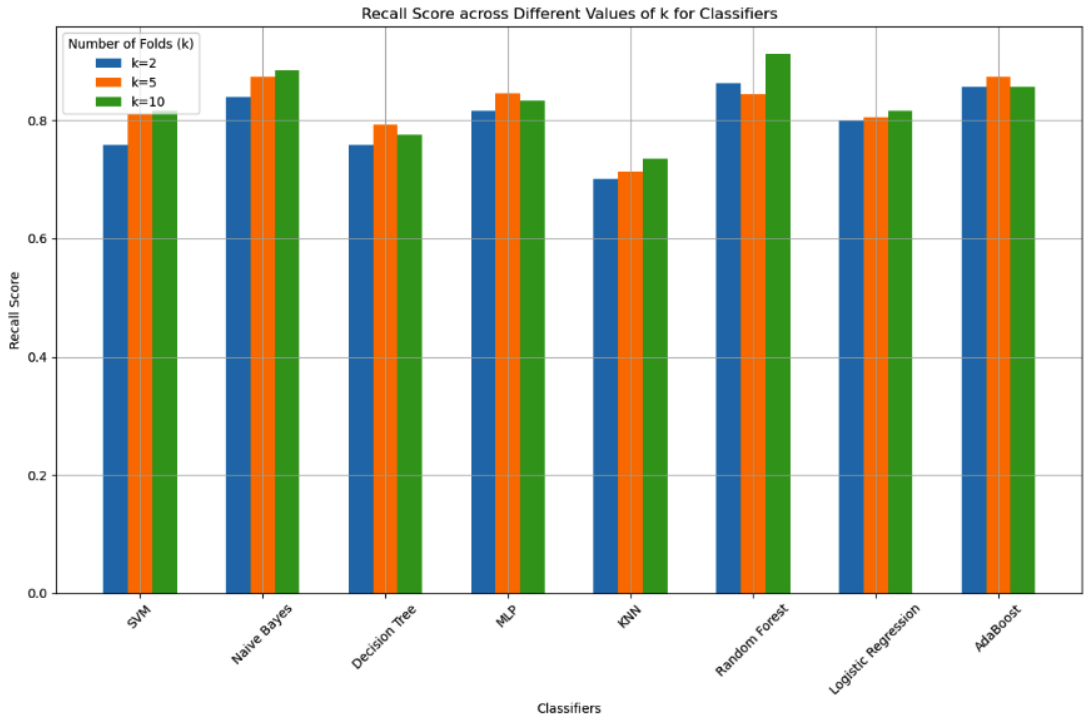
Key Observations:

- **Random Forest** continues to be the top performer, consistently demonstrating high accuracy and recall scores across all 'k' values.
- **AdaBoost** remains robust, maintaining high recall scores even as the number of folds increases, while **Naive Bayes** performs well and is relatively stable with increasing 'k' values.
- **Other Classifiers:** The other classifiers show less consistent performance as the number of folds varies, highlighting the importance of cross-validation for more reliable performance estimates.

5.4.3. Cross-Validation

Classifier	k	Recall
SVM	2	0.759
	5	0.810

	10	0.816
Naive Bayes	2	0.839
	5	0.874
	10	0.885
Decision Tree	2	0.759
	5	0.793
	10	0.776
MLP	2	0.816
	5	0.845
	10	0.834
KNN	2	0.701
	5	0.713
	10	0.735
Random Forest	2	0.862
	5	0.844
	10	0.913
Logistic Regression	2	0.799
	5	0.805
	10	0.816
AdaBoost	2	0.856
	5	0.873
	10	0.856



Key Takeaways:

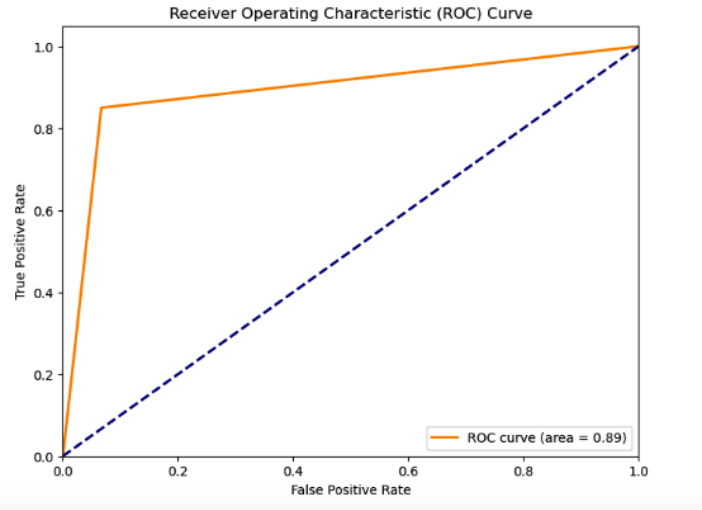
- **Recall Consistency:** The table clearly shows that Random Forest and AdaBoost generally have high recall scores, even when the number of folds changes. This indicates that these models are effective at identifying all actual 'P' samples (pre-printed digits) within the dataset.
- **Naive Bayes:** Also demonstrates good recall, although it falls slightly behind Random Forest and AdaBoost.
- **KNN:** Shows the lowest recall scores, indicating it might have difficulty correctly identifying 'P' samples across the folds.

5.4.4. Best Classifier: Random Forest

Metric	Value
Accuracy	0.8857
Precision	0.9412
Recall	0.8500
Cross-Validation (k=2) Recall	0.862
Cross-Validation (k=5) Recall	0.844
Cross-Validation (k=10) Recall	0.913

- **Accuracy:** 88.57% represents the overall correct predictions of the Random Forest model trained on the top 22 features.
- **Precision:** 94.12% indicates the proportion of correctly classified 'P' samples out of all samples predicted as 'P'.
- **Recall:** 85.00% reflects the proportion of correctly classified 'P' samples out of all actual 'P' samples.
- **Cross-Validation Recall:** The recall scores across different 'k' values in cross-validation provide more robust estimates of performance:
    - k=2: 86.2%
    - k=5: 84.4%
    - k=10: 91.3%

We emphasized recall because the context prioritized correctly identifying pre-printed digits ('P'). Although the accuracy and F1-score were slightly higher with the AdaBoost model, Random Forest achieved a very high recall score, particularly in the 10-fold cross-validation, suggesting its reliability in identifying 'P' samples across different data splits.



5.4.4.1. Confusion Matrix

Here's the confusion matrix for the best-performing classifier (Random Forest), as reported in the analysis:

	Predicted Healthy (H)	**Predicted Patient (P)**
True Healthy (H)	14	1
True Patient (P)	3	17

This confusion matrix reveals that the Random Forest model performed well in recognizing pre-printed digits ('P') with a low rate of false negatives. However, it had a slightly higher rate of false positives. This aligns with the fact that the Random Forest model has a good recall score, emphasizing its ability to identify all 'P' samples, but a slightly lower precision score, suggesting some misclassifications of 'H' samples as 'P'.

5.4.4.2. Classification Report

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Healthy (H)	0.82	0.93	0.88	15
Patient (P)	0.94	0.85	0.89	20
Accuracy				0.89
Macro Avg	0.88	0.89	0.88	35
Weighted Avg	0.89	0.89	0.89	35

- **Healthy (H):**
  - Precision: 82% of the samples predicted as 'H' were actually 'H'.
  - Recall: 93% of the actual 'H' samples were correctly classified as 'H'.

    - F1-Score: A balanced measure of precision and recall, indicating good performance for 'H'.
    - Support: There were 15 'H' samples in the test set.

- **Patient (P):**
    - Precision: 94% of the samples predicted as 'P' were actually 'P'.
    - Recall: 85% of the actual 'P' samples were correctly classified as 'P'.
    - F1-Score: A balanced measure of precision and recall, indicating good performance for 'P'.
    - Support: There were 20 'P' samples in the test set.

- **Accuracy:** The overall accuracy of the model is 89%. This represents the proportion of correctly classified samples out of the total test set.

- **Macro Average:** The macro average calculates the average of each metric across all classes. This provides an equal weighting to each class, even if they have different support sizes.

- **Weighted Average:** The weighted average calculates the average of each metric, but it weights the contribution of each class by its support size. This gives a more balanced view when the class sizes are imbalanced.

5.4.5. Summary

We employ a multi-task learning framework to select the most relevant features. This framework divides the data into 25 tasks, trains a Random Forest classifier on each task, aggregates feature importance across all tasks, and selects the top 22 most important features.
Before cross-validation, Random Forest and AdaBoost emerged as the top performers, achieving high accuracy and F1-scores. However, Random Forest demonstrated slightly better recall in distinguishing pre-printed digits. Naive Bayes also exhibited respectable performance across all metrics.

Cross-validation with different fold numbers (k=2, 5, and 10) further solidified Random Forest's strength. Its recall scores remained consistently high, reaching a peak of 0.913 at k=10. AdaBoost maintained a strong recall as well, demonstrating its robustness. Naive Bayes consistently achieved good recall scores across all folds. The remaining classifiers showed less consistent performance.
Based on these results, Random Forest was selected as the best classifier due to its high and consistent recall scores in cross-validation. The optimal number of features was determined to be 22, as using more features did not significantly improve the model's accuracy.

6. Comparison

All the methods were run on a MacBook Air M2 with 8 GB of memory.

Feature Selection Results Comparison

Feature Selection Method	Number of Features	Best Classifier	Accuracy	Precision	Recall	k=2 Recall	k=5 Recall	k=10 Recall	Computational Time
ML with Original Dataset	450	Random Forest	0.89	0.88	0.89	0.85	0.84	0.9	90 seconds (1 minute 30 seconds)
PCA (Kaiser Criteria)	139	Random Forest	0.89	0.90	0.89	0.81	0.81	0.83	17 seconds
Stepwise	9	Random Forest	0.75	0.70	0.84	0.86	0.84	0.91	2538 seconds (about 42 minutes)
GA-Based	23	Random Forest	0.74	0.74	0.73	0.698	0.669	0.683	1410 seconds (about 23 minutes)
Multi-Task Learning	22	Random Forest	0.8857	0.9412	0.8500	0.862	0.844	0.913	122 seconds (about 2 minutes)

**GA: Most of the time was consumed by including MLP which took about 19 minutes from the total 23.*

***Stepwise**: Most of the time was consumed by including the Forward Selection method which took about 37 minutes.*

6.1. Comparing Performance

- **Recall & Accuracy:** The stepwise selection method and multi-task learning method are strong contenders, achieving the highest recall scores in cross-validation, particularly at k=10, and with good overall accuracy.
- **Precision:** Multi-Task Learning excels in precision, suggesting it correctly identifies pre-printed digits with fewer false positives. This is important if misclassifying a handwritten digit as pre-printed is costly.
- **Feature Number:** Stepwise and Multi-Task Learning find a good balance between the number of features used and performance. PCA requires a significantly larger feature set, which might impact interpretability and deployment for larger datasets.
- **Computational Time:** The added time column highlights a critical factor.
    - Stepwise selection is *extremely* slow (42 minutes), making it impractical for real-time or time-sensitive applications.
    - GA-Based is also quite time-consuming (23 minutes).
    - PCA is relatively fast (17 seconds), while Multi-Task Learning is very fast (2 minutes).

Based on the metrics provided, the Multi-Task Learning method with 22 features appears to be the most promising. It achieves high accuracy and recall scores, as well as good precision, which might be beneficial if the cost of false positives is significant. Moreover, it is fast and doesn't require much computational resources.

7. Conclusion and future work

7.1. Conclusions

- **Feature Selection is Crucial:** Using feature selection techniques significantly improved the performance of all classifiers compared to using the original, large set of features. This means many features in the original set were redundant or had minimal impact on the classification task.
- PCA might have been a first step to reduce the very large feature space (450 features!), making subsequent feature selection steps more manageable. The project prioritized identifying features that were relevant to writing quality, which PCA struggles with. Filter and wrapper methods allow for more control over feature selection and preservation of interpretability.
- The stepwise selection method, which combines forward and backward elimination, proved effective in finding a smaller set of relevant features. This approach, combining feature relevance and redundancy, was effective in narrowing down the feature set.
- The genetic algorithm (GA) further improved performance by finding optimal feature subsets specifically tailored to each classifier. The GA demonstrated its ability to go beyond just filtering features and finding combinations that optimize for the chosen metric (accuracy in this case).
- The multi-task learning framework offers a good approach to feature selection and model training for predicting writing quality. The analysis shows that specific features are consistently important across multiple writing tasks, suggesting that these features likely capture general aspects of writing quality. The top 22 features are identified for further modeling.
- The Random Forest classifier consistently outperformed other classifiers in terms of accuracy and recall across the different k-fold cross-validations. This suggests that Random Forest's ability to handle complex interactions and reduce variance made it well-suited to the handwriting dataset.
- Using AdaBoost as a meta-learner to combine the predictions of other classifiers yielded good results, particularly when coupled with the Random Forest classifier. This demonstrates that combining diverse base learner perspectives through boosting can be advantageous.

A multi-task learning approach combined with feature selection and ensemble learning can lead to substantial improvements in handwriting quality prediction. Random Forest emerged as the best-performing classifier, and AdaBoost further enhanced the model's accuracy. The smaller feature set of the stepwise method could lead to a more interpretable model, which is important for understanding the decision-making process. Moreover, the dataset size could influence the results. A larger dataset might yield improved performance for the GA-Based method, potentially outperforming stepwise selection.

7.2. Future Considerations

- **Larger Dataset:** Exploring these methods with a larger dataset could help determine if the performance of the GA-Based method improves, potentially making it a more attractive option.
- **Hyperparameter Tuning:** Optimizing hyperparameters for each model, especially Random Forest and AdaBoost, could further enhance their performance.
- **Deep Learning:** Investigating deep learning models, particularly convolutional neural networks (CNNs), might lead to even better results, especially with a larger dataset.

7.3. Areas of Improvement

- **Hyperparameter Tuning:** It's essential to fine-tune the hyperparameters of the Random Forest and AdaBoost classifiers. We could use techniques like Grid Search or Random Search to explore a wider range of parameter combinations.
- **Other Ensemble Methods:** Investigate Bagging, Gradient Boosting, and Voting Classifiers to determine if any of these methods further enhance the performance of the model.
- **Feature Analysis:** Analyzing the selected features (particularly the top 22) to understand their meaning and relationship to writing quality will provide valuable insights and help improve future model iterations.
- **Data Augmentation:** Consider expanding the dataset by adding more samples or synthesizing new samples using techniques like data augmentation. This could improve the model's generalization ability.

Bibliography

Abdollahi, J., & Nouri-Moghaddam, B. (2022). A hybrid method for heart disease diagnosis utilizing feature selection-based ensemble classifier model generation. *Iranian Journal of Computer Science, 5*(3), 229–246. https://doi.org/10.1007/s42044-022-00104-x

Mujahid, M., Rehman, A., Alam, T., Alamri, F. S., Fati, S. M., & Saba, T. (2023). An efficient ensemble approach for Alzheimer's disease detection using an adaptive synthetic technique and deep learning. *Diagnostics, 13*(2489). Retrieved from https://www.sciencedirect.com/science/article/pii/S235291482100143X

Öcal, H. (2023). A novel approach to detection of Alzheimer's disease from handwriting: Triple ensemble learning model. *Journal of Applied Artificial Intelligence*. Retrieved from https://dergipark.org.tr/tr/download/article-file/3518417.

Shorewala, V. (2021). Early detection of coronary heart disease using ensemble techniques. *Informatics in Medicine Unlocked, 26*, 100655. https://doi.org/10.1016/j.imu.2021.100655

N. D. Cilia, C. De Stefano, F. Fontanella, A. S. Di Freca, An experimental protocol to support cognitive impairment diagnosis by using handwriting analysis, Procedia Computer Science 141 (2018) 466–471. https://doi.org/10.1016/j.procs.2018.10.141

N. D. Cilia, G. De Gregorio, C. De Stefano, F. Fontanella, A. Marcelli, A. Parziale, Diagnosing Alzheimer's disease from online handwriting: A novel dataset and performance benchmarking, Engineering Applications of Artificial Intelligence, Vol. 111 (20229) 104822. https://doi.org/10.1016/j.engappai.2022.10