Handwriting & Alzheimer's disease: Diagnosis with machine learning

Jacob Titcomb

Spring 2024

Contents

1 Introduction

1.1 Motivations

Neurodegenerative diseases like Alzheimer's and Parkinson's are currently-incurable conditions caused by deterioration of nerve cells. While Parkinson's disease manifests as both cognitive and motor decline, Alzheimer's is associated more closely with just cognitive decline. In the initial stages of Alzheimer's, ventromedial temporal lobe dysfunction leads to minor lapses in memory (Armstrong et al, 2013). With progression of the disease, amnesia—as well as mental faculties—worsen, indicative of "pathological involvement of more widespread neural systems" (Cilia et al., 2022). Identifying Alzheimer's early on benefits for the patient, such as potentially increasing quality of life by providing earlier access to care. And on the research side, early diagnosis gives more opportunities for pathologists to study the disease, searching for a potential cure.

The data for this project comes from a 2022 study by Cilia et al. which observes the handwriting of people with and without Alzheimer's disease, with the purpose of diagnosis. The data set is called DARWIN, or Diagnosis AlzheimeR WIth haNdwriting; in it, researchers recorded the kinematic and dynamic information of each participant when performing 25 writing tasks. Of the 174 participants, 89 of them had the diseas and 85 were healthy controls. These tasks come from a similar 2018 study by Impedovo et al., looking instead at individuals with Parkinson's disease. The handwriting tests can be categorized as follows: drawing simple geometric figures, copying semantically meaningful letters, words, or numbers, and writing dictated text (Cilia et al., 2022). Tasks increased with difficulty as the test continued, with the final tasks being to copy a paragraph. Participant movement was recorded using "cheap and widely used [sic] graphic tablets" (Cilia et al., 2022), with the goal of creating an easy, widely-accessible test for the disease based on motor function rather than mental faculties.

1.2 The Features

The data comes from a 2022 Italian study, in which researchers had 174 participants perform 25 handwritten tasks (Cilia et al., 2022). A tablet and writing utensil were used to record 18 measures per task, encompassing the time spent, writing speed, writing acceleration, writing pressure, and other features. In total, 450 features represent the feature space of this dataset.

Below we have the low-dimensional representations of the labeled data, using PCA, t-SNE, and MDS for dimension reduction. Below that, the correlogram indicates that most of the features are relatively uncorrelated, with similar tasks having some correlation.

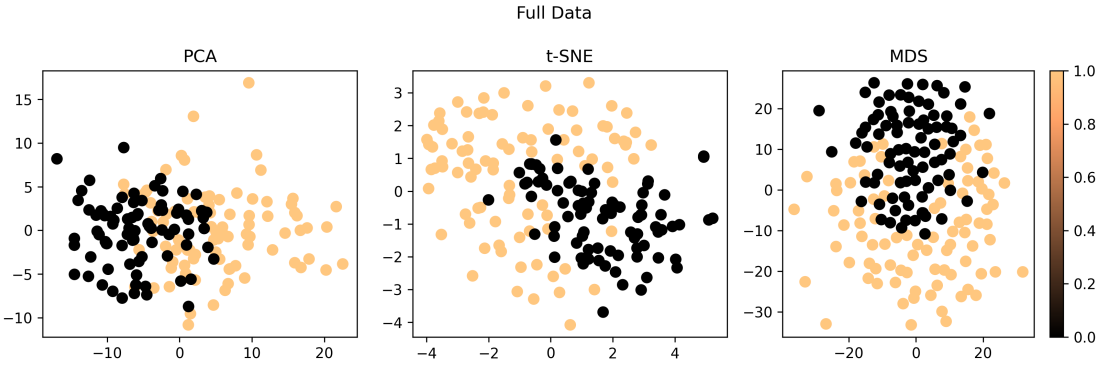
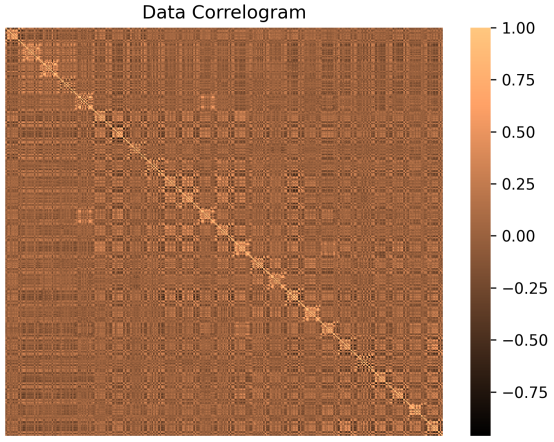


Figure 1: Low-dimensional representation of the data



1.3 Methods

All data processing and plotting were done using the following libraries in Python: NumPy, Pandas, Matplotlib, Seaborn, and SciPy. Model fitting and evaluation was done using scikit-learn and XGBoost. The work for this project was done in the associated Jupyter Notebook.

First, logarithm and inverse hyperbolic sine transformations were applied to features which, under the appropriate transformation, showed a decrease in skew. Once the appropriate transformations were applied, a 70-30 train-test split was performed. Both the train and test set were scaled and centered, having a mean of 0 and standard deviation of 1.

We fit the following models:

1. Nearest centroid (baseline)
2. Logistic regression
3. Decision tree
4. Random forest
5. Adaptive boosting (AdaBoost)
6. Extreme gradient boosting (XGBoost)
7. Support vector machine (SVM)
8. Neural network (MLP, for multilayer perceptron)
9. K-nearest neighbors
10. Linear discriminant analysis
11. Naïve bayes
12. Stacking
13. Voting

When hyperparameters needed to be set, a grid search was performed, using 10-fold cross validation repeated 5 times on the training data. We selected the optimal parameters based on the results.

We trained each model using 10-fold cross validation repeated 5 times. The following performance measures were calculated for the cross-validation and on the test set: accuracy, sensitivity, specificity, and F1 score. The area under the curve of the ROC curve was also calculated for the test set. Feature importance was calculated for each model; for tree-based models we used the typical feature importance, but for the others, permutation importance based on impurity was used.

Finally, we generated the following plots for each model applied on the test set: three low-dimensional representations (PCA, t-SNE, and MDS), the confusion matrix, and the ROC curve. For the low-dimensional representations, pink outlined points indicate misclassification.

Since the classes are about even (89 positive to 85 negative), our final analysis will use the accuracy as the guiding metric.

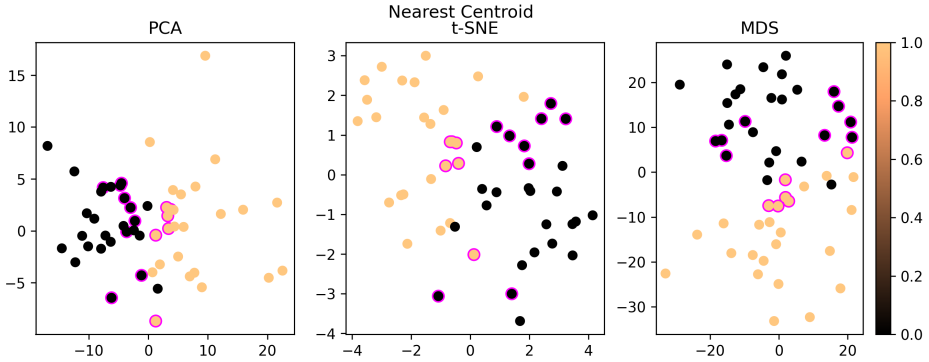
2 Model Fitting

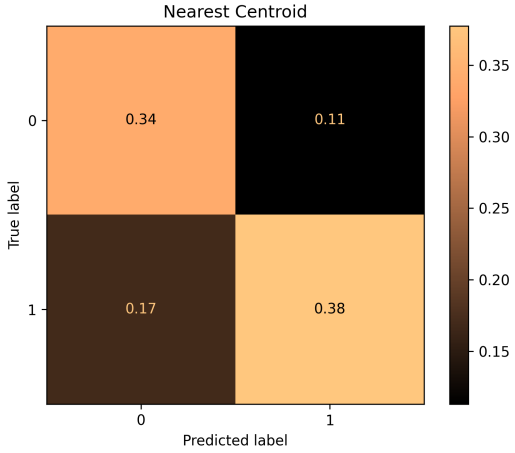
2.1 Nearest Centroid

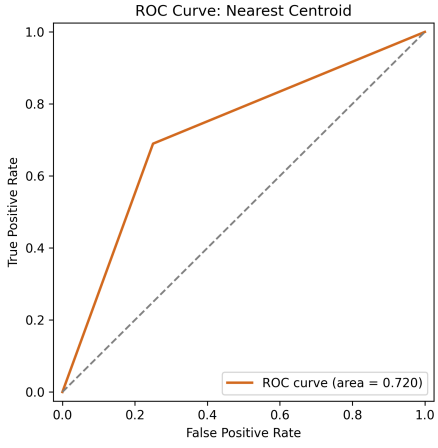
Rather than the majority rule classifier, we decided to use this model as the baseline. From the low-dimensional representation, we see that the misclassified points are on the border, indicating some non-complex behavior that can be modeled in this simple way.

Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.8232	0.7767	0.8690	0.8099	
Test	0.7170	0.6897	0.7500	0.7273	0.7198

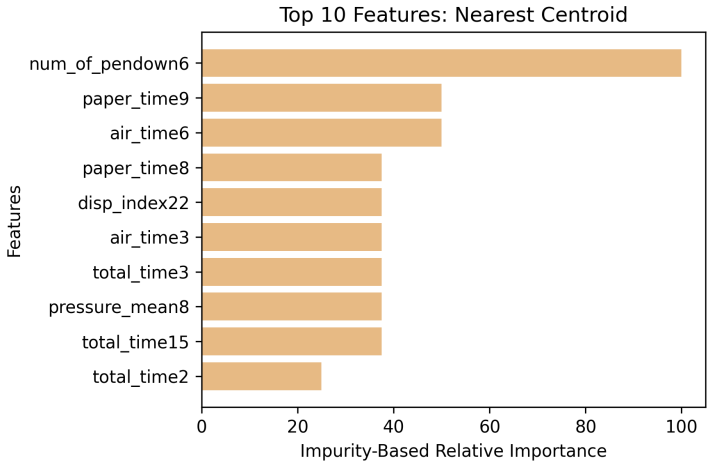
Table 1: Model performance measures







The top 10 features are seen below. Notably, the majority of these features are from the more simple tasks (lower task number).

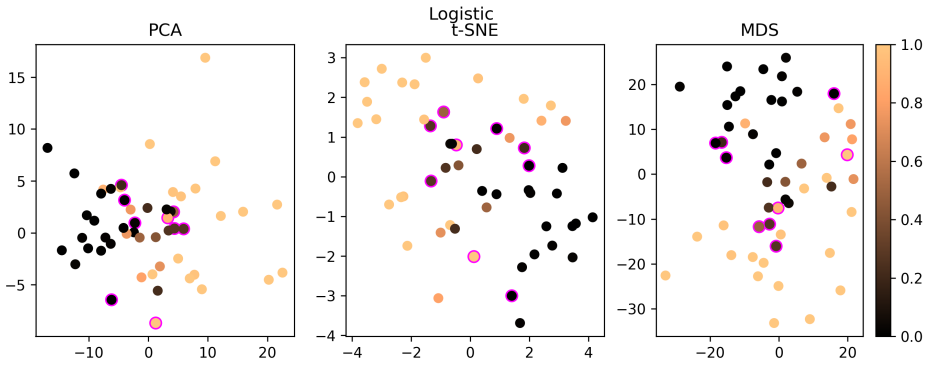


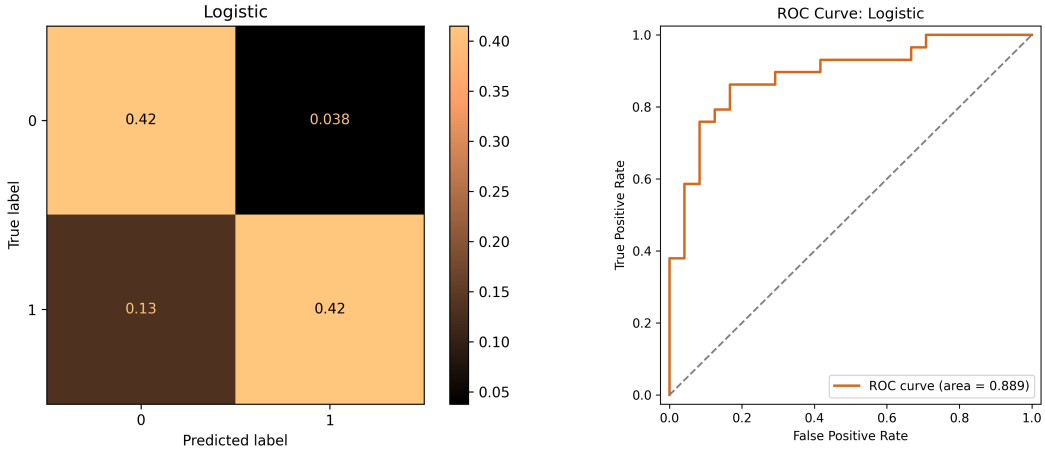
2.2 Logistic Regression

This model performs slightly better than the baseline on cross validation, but significantly better on the test measures. Visually, most of the misclassification occurs around the boundary.

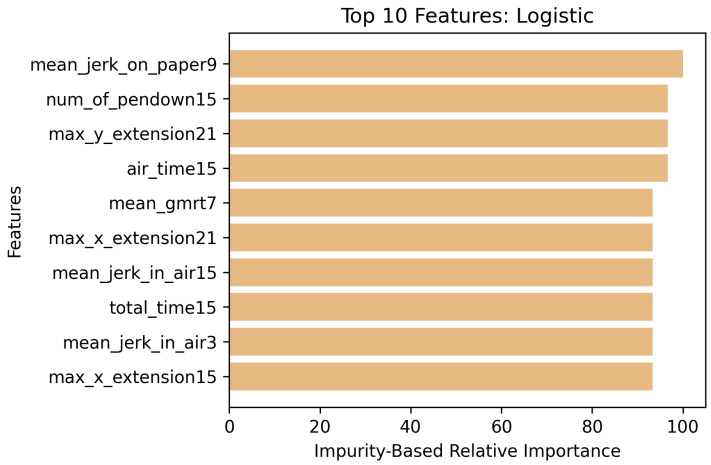
Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.8599	0.8533	0.8667	0.8527	
Test	0.8302	0.7586	0.9167	0.8302	0.8894

Table 2: Model performance measures





From the importance plot below, the complicated tasks tend to be more important in the model.

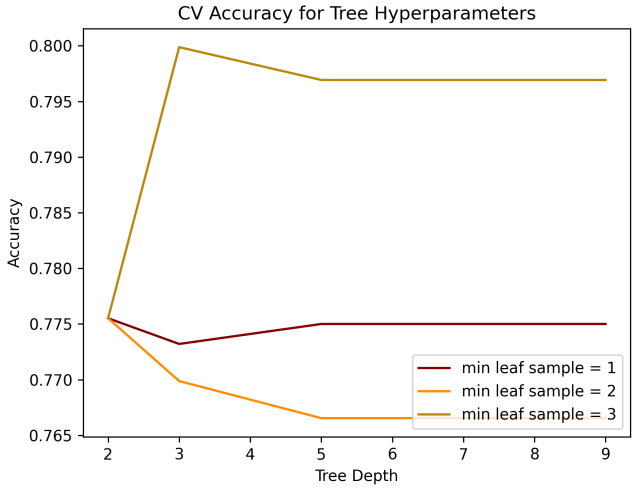


2.3 Decision Tree

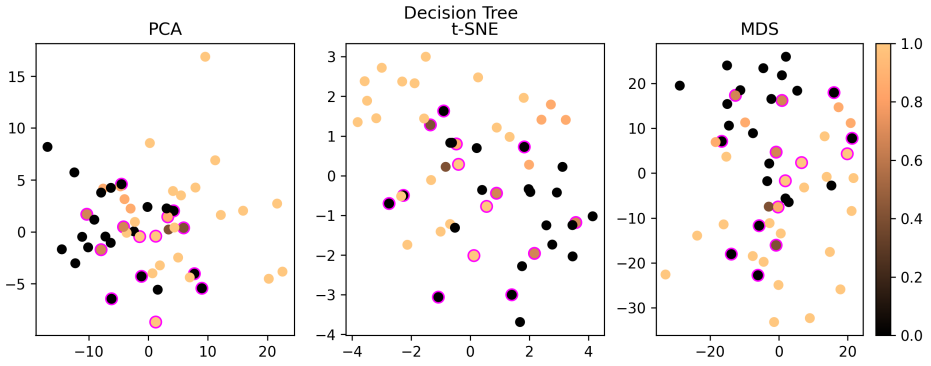
We used cross validation to find the optimal hyperparameters of $\alpha = 0$, maximum depth of 3, and minimum samples per leaf of 3. The model performs worse than the baseline on cross validation, but marginally better on the test measures.

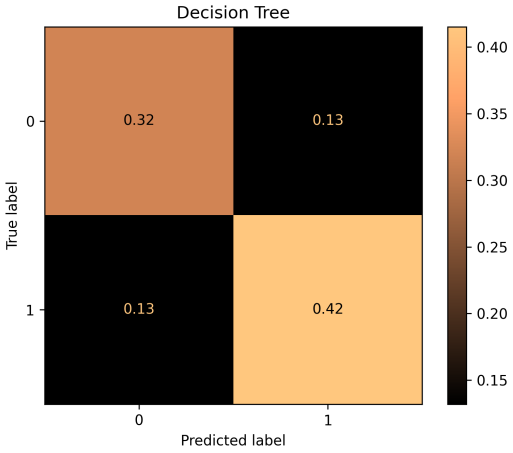
Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.7999	0.7967	0.8024	0.7954	
Test	0.7358	0.7586	0.7083	0.7586	0.7924

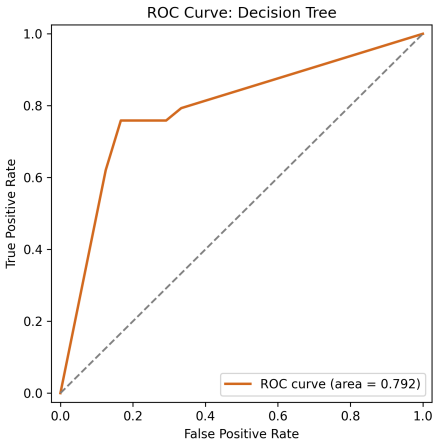
Table 3: Model performance measures



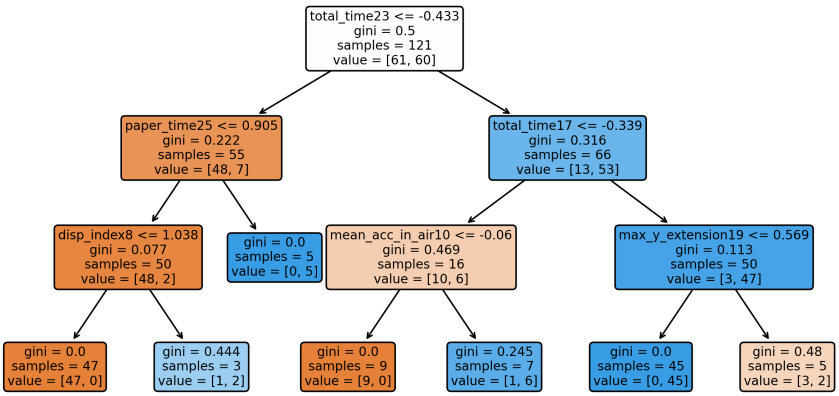
From the low-dimensional representations the incorrect classes seem to be more spread out, farther away from the visual decision boundary. The ROC curve also looks significantly worse than that of the logistic regression.

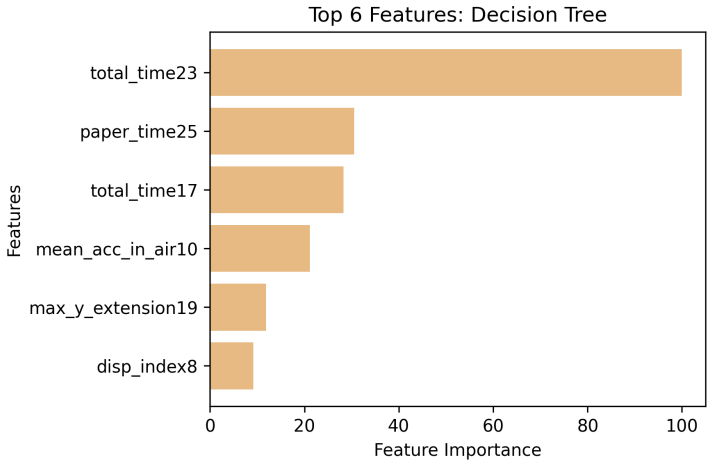






Below is the tree itself, with the feature importance below it. Interestingly, the tree is fairly shallow, and the most important features tend to be those relating to time taken. Unsurprisingly, most of the important features come from the more complex tasks.





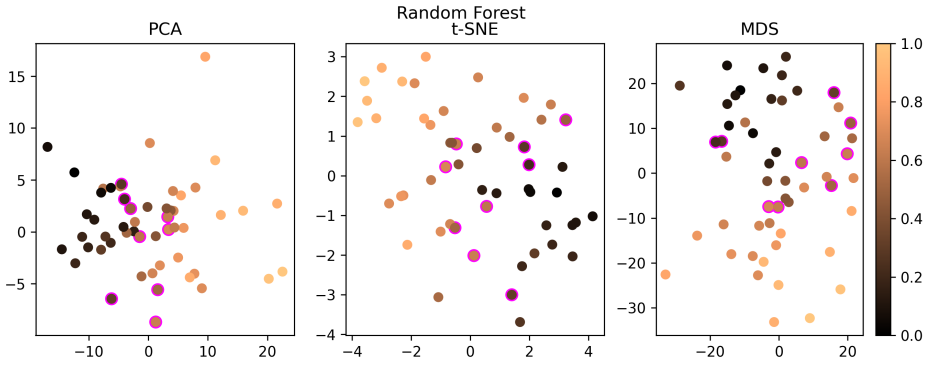
2.4 Random Forest

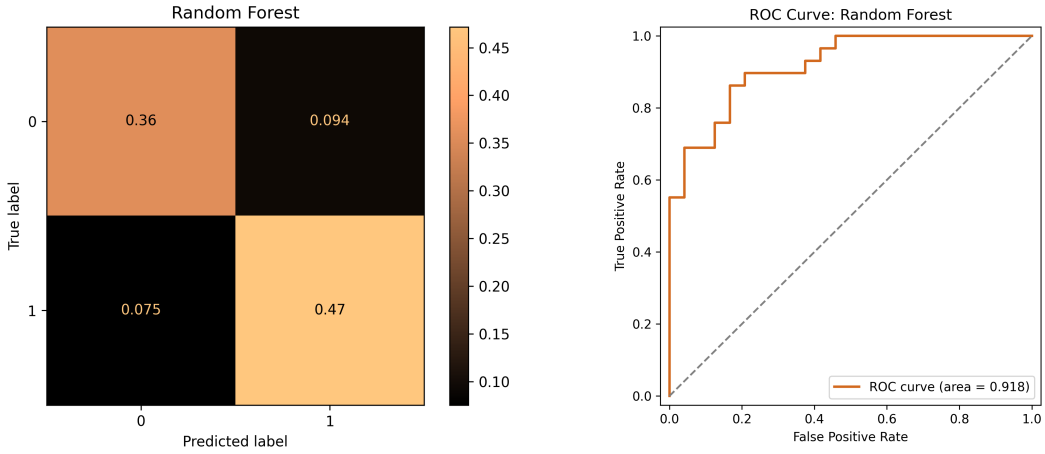
The optimal parameters for this model were determined to be a maximum depth of 5, minimum samples in a leaf of 1, minimum samples in a split of 5, and number of estimators of 163 (just over 36% of the 450 total).

The random forest model is our first to have a cross-validation accuracy above 90%. Visually, the few misclassified points tend to be near the visual boundary, and the ROC curve has a sharper angle, indicating better performance.

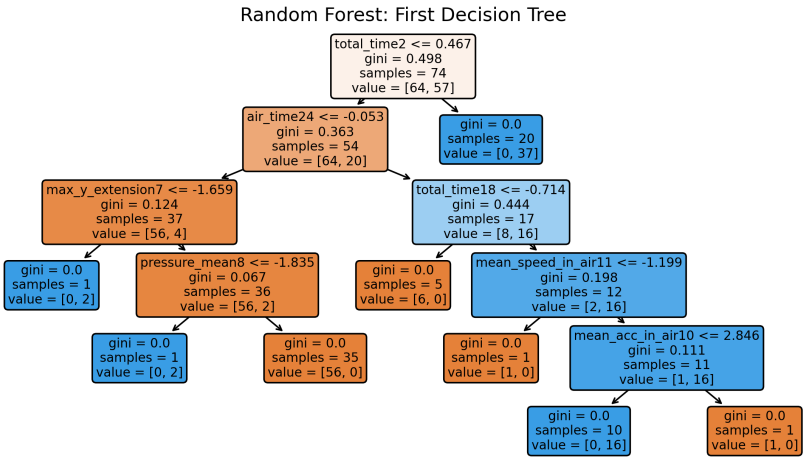
Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.9077	0.9033	0.9114	0.9042	
Test	0.8302	0.8621	0.7917	0.8475	0.9181

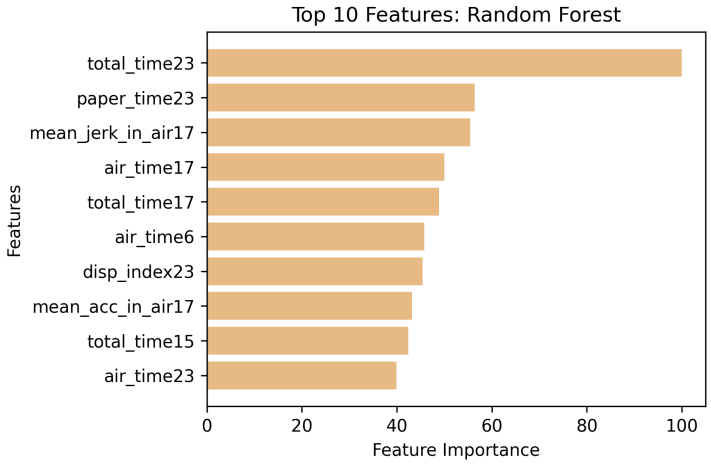
Table 4: Model performance measures





Below is the first decision tree of the forest, just to get an idea of what they look like. Similar to the decision tree model, the most important features tended to involve time, and favored almost exclusively the later tasks.





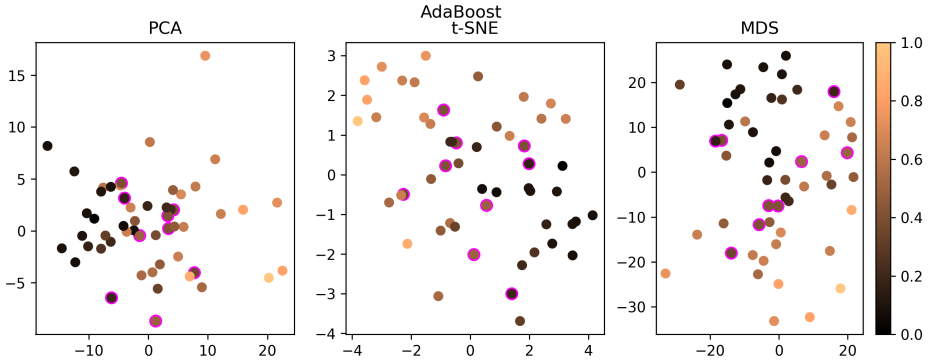
2.5 Adaptive Boosting (AdaBoost)

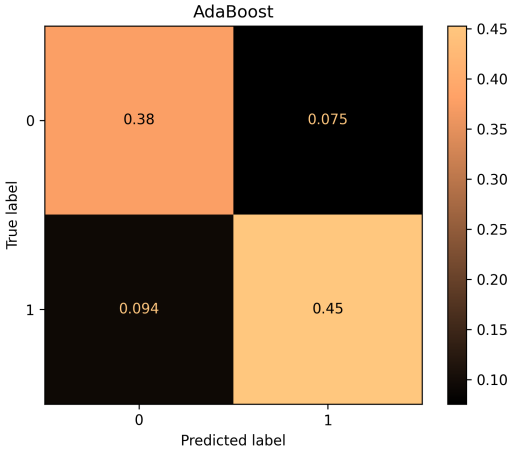
The optimal parameters for this model were determined to be a learning rate of 0.5 and number of estimators of 250 (about 56% of the 450 total).

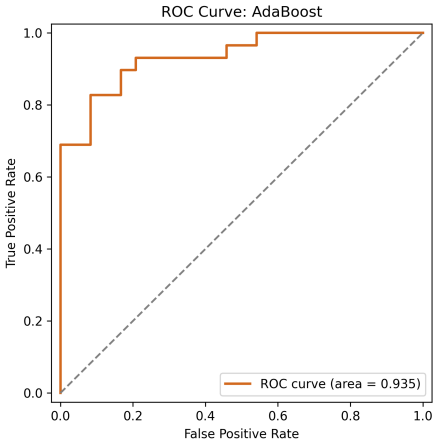
The AdaBoost model performs better on cross-validation accuracy and has a better ROC curve than the random forest model. Although the misclassified points appear to be a bit more spread out.

Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.9258	0.9067	0.9443	0.9217	
Test	0.8302	0.8276	0.8333	0.8421	0.9353

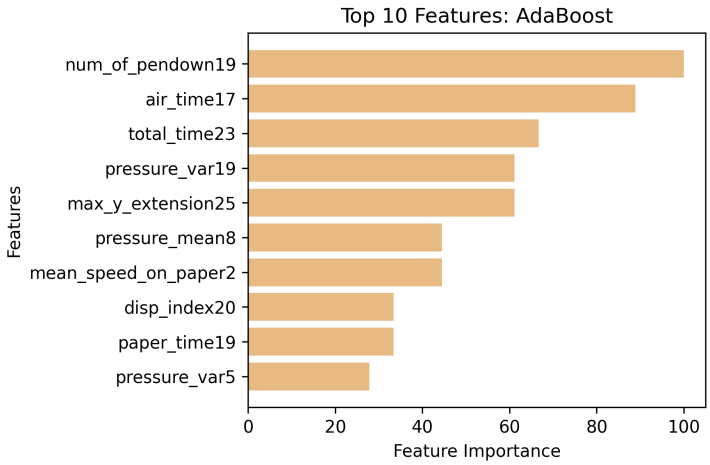
Table 5: Model performance measures







Below is the importance plot. Interestingly, the most important features come from a wider range of tasks and a wider diversity of measurement types.



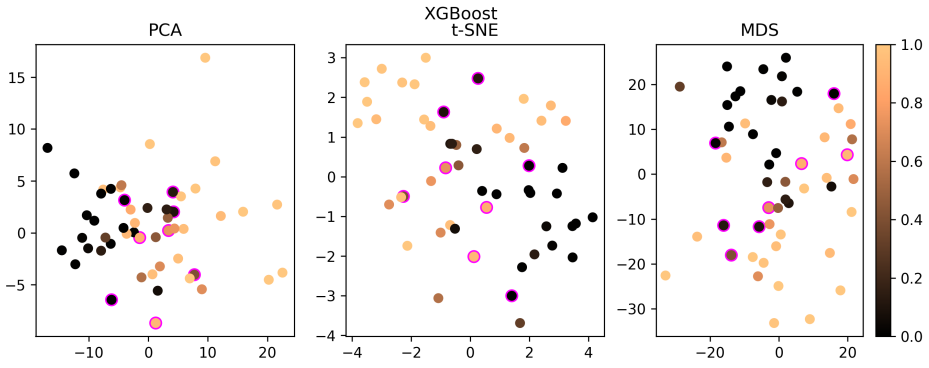
2.6 Extreme Gradient Boosting (XGBoost)

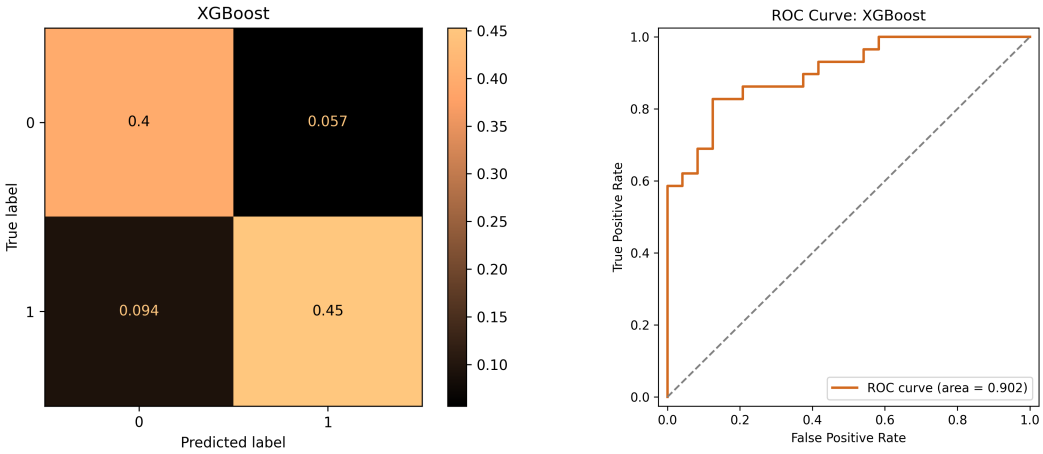
The optimized parameters were a learning rate of 0.215, maximum depth of 1 (very shallow), and number of estimators of 98.

The XGBoost model performs very similarly to the AdaBoost model. Notably, this model performs worse on cross validation but better on the test data. The area under the ROC curve is lower for this model than for AdaBoost.

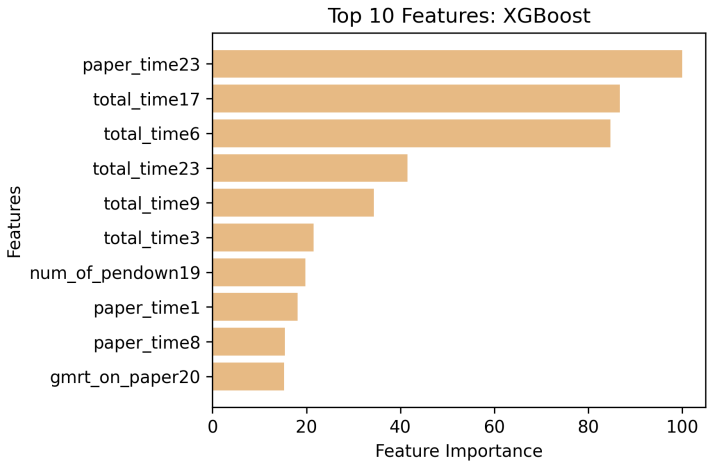
Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.9091	0.9033	0.9143	0.9065	
Test	0.8491	0.8276	0.8750	0.8571	0.9023

Table 6: Model performance measures





Below is the feature importance graph. Nearly all of the most important features have to do with time, though coming from a variety of the tasks.



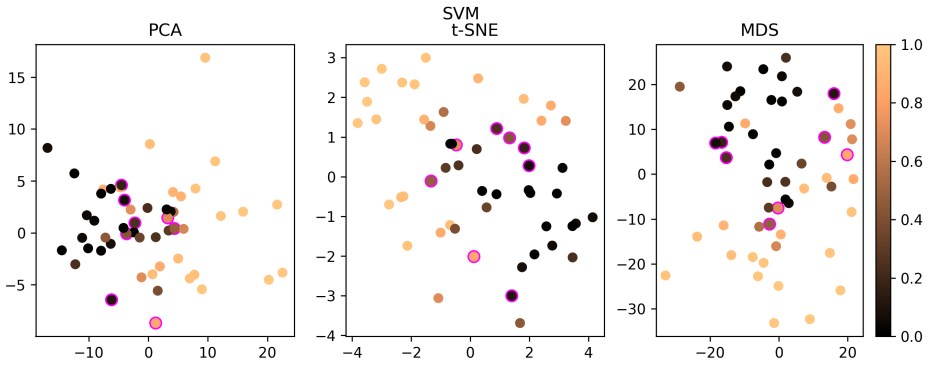
2.7 Support Vector Machine

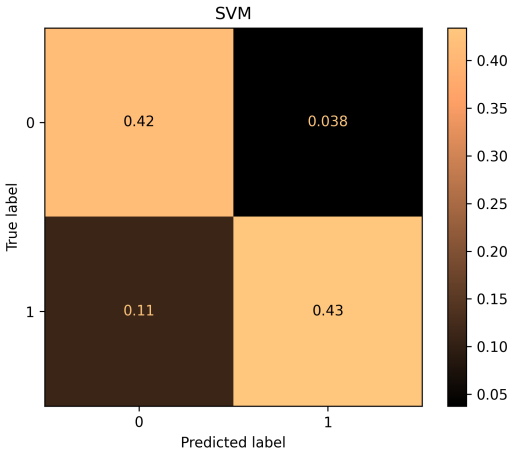
For the SVM model, we opted for a radial basis kernel. The optimal cost was 1.1 and optimal gamma was 'scale', the sklearn default parameter where `gamma = 1 / (n_features * X.var())`.

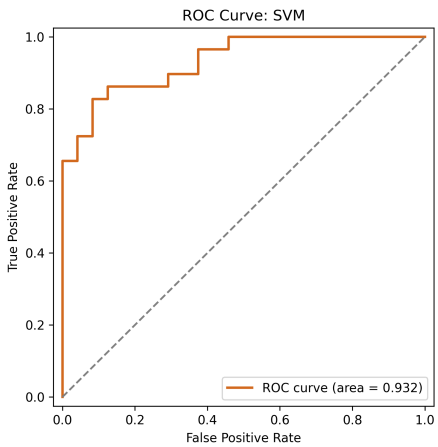
The SVM model performed very similar to the XGBoost model, even scoring higher on the area under the ROC curve. It tends to classify negative labels better than positive labels, relative to the XGBoost model.

Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.9012	0.9000	0.9019	0.8986	
Test	0.8491	0.7931	0.9167	0.8519	0.9325

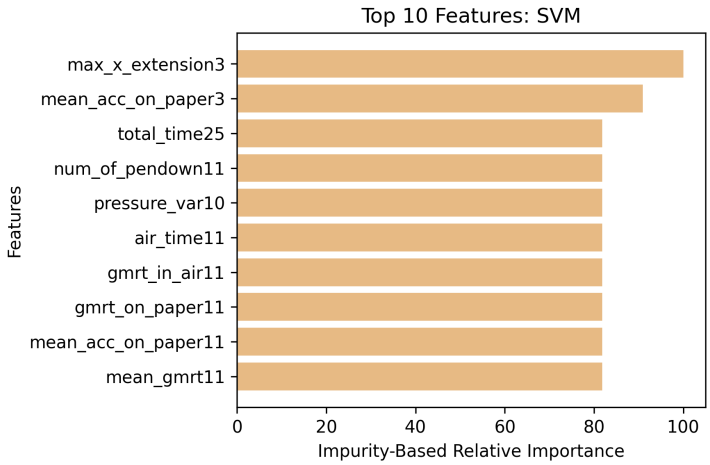
Table 7: Model performance measures







Interestingly, almost all the most important features come from the first 11 tasks.



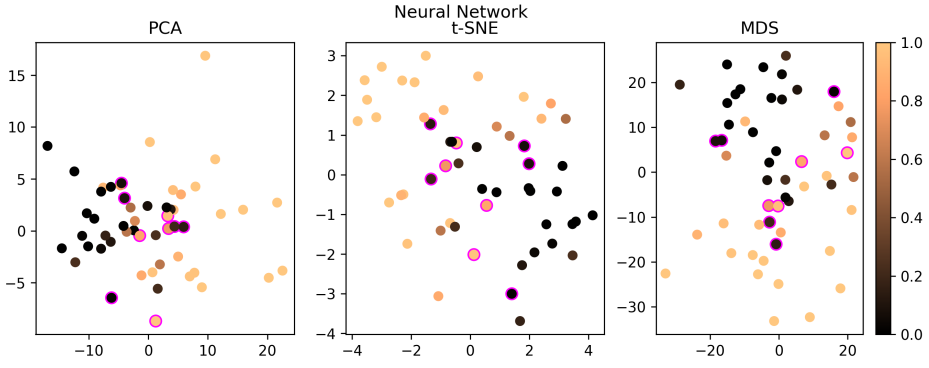
2.8 Neural Network

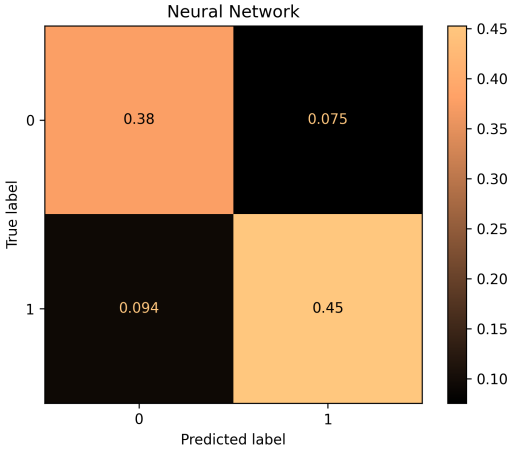
Initially, we wanted to construct the neural network using Keras or PyTorch. However, due to time constraints, we ended up constructing a multilayer perceptron model using the sklearn implementation. Our grid search for the optimal parameters was extensive. First, between an ADAM and LBFGS solver, ADAM was more optimal. Between sigmoid, tanh, and ReLU activations, ReLU was optimal. Between an adaptive and constant learning rate, constant was better. Between 1, 2, and 3 hidden layers, 3 was optimal. Finally, a thorough grid search found these optimal hyperparameters: α of 0, learning rate of 0.0001, and hidden layers of sizes 10, 40, and 50 nodes.

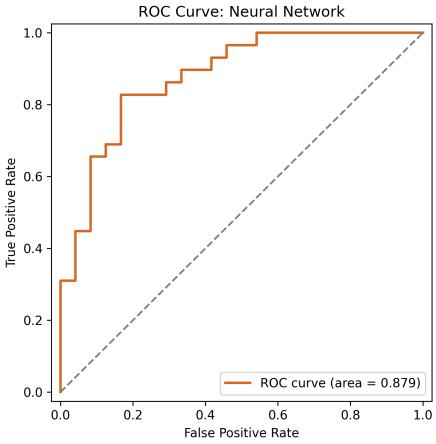
The training accuracy was fairly good, but the testing accuracy was less than other models. This result was expected since—though the feature space is high-dimensional—the full data set has only 174 observations, even less when accounting for a train-test split and cross validation.

Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.9178	0.9467	0.8890	0.9196	
Test	0.8302	0.8276	0.8333	0.8421	0.8793

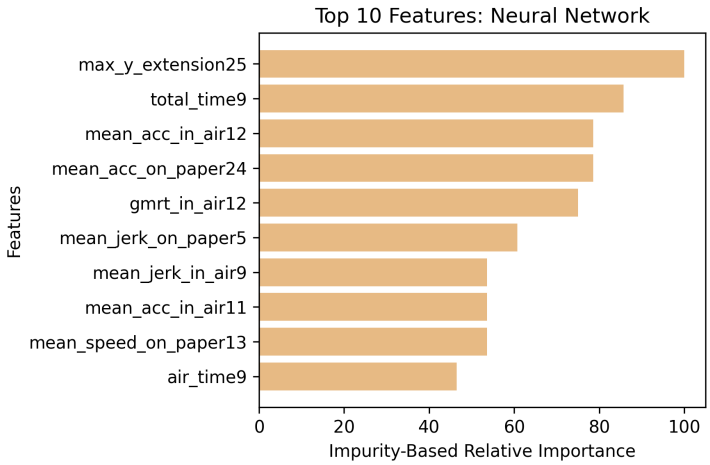
Table 8: Model performance measures







Below are the top 10 features, coming from a variety of measures and a wide set of tasks.

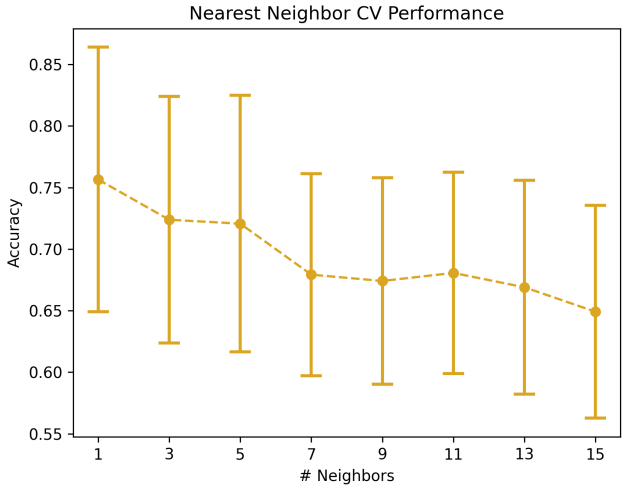


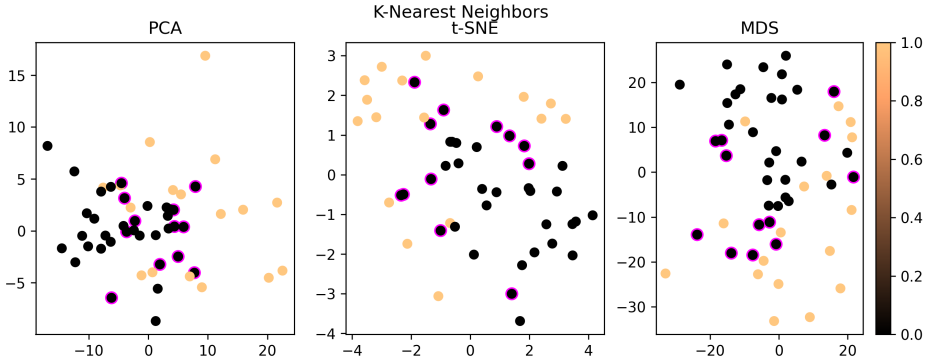
2.9 K-Nearest Neighbors

Using cross validation, we found that $k = 1$ is the optimal number of neighbors. Unfortunately, the accuracy for cross validation and on the test set are both unimpressive compared to the past few classifiers.

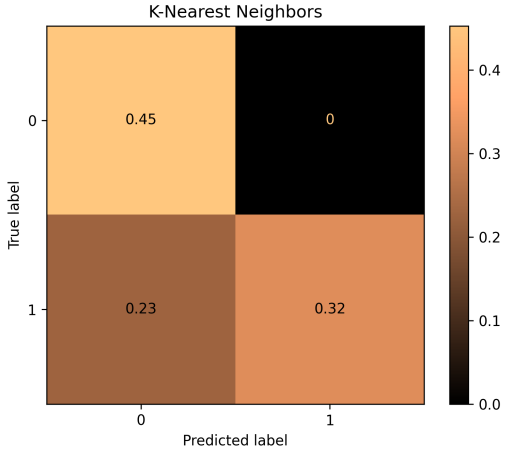
Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.7565	0.5100	1.0000	0.6462	
Test	0.7736	0.5862	1.0000	0.7391	0.7931

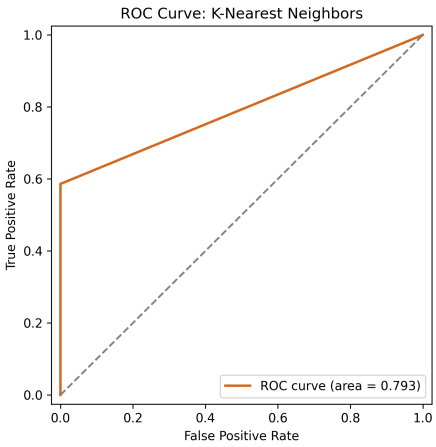
Table 9: Model performance measures

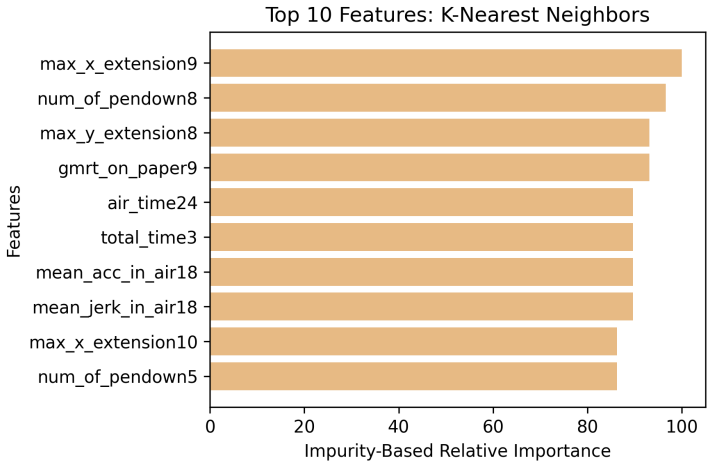




In the below two plots there are two things immediately noticeable: first, the false positive rate for this classifier is 0; second, the ROC curve is just a single angle because KNN with $k = 1$ is a hard classifier, only outputting 0 or 1. Then in the top 10 features, tasks 8 and 9 appear to be very important for this model.





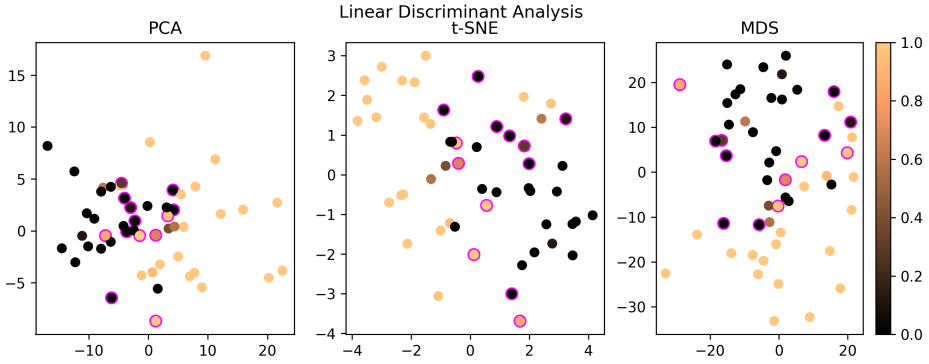


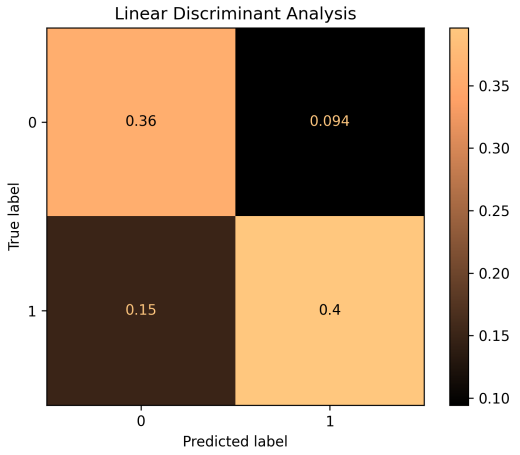
2.10 Linear Discriminant Analysis

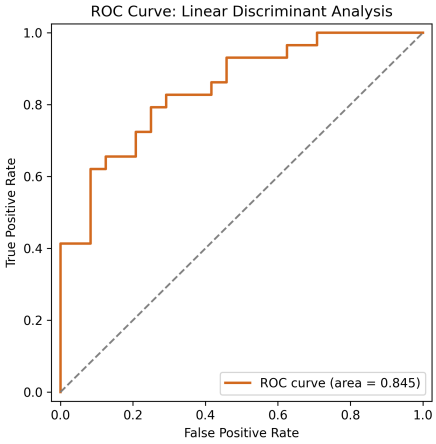
For this model, the accuracy is a little higher than the baseline for both cross validation and on the testing set. Though clearly by the confusion matrix and ROC curve, this model does not perform well comparatively.

Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.8309	0.7967	0.8638	0.8192	
Test	0.7547	0.7241	0.7917	0.7636	0.8448

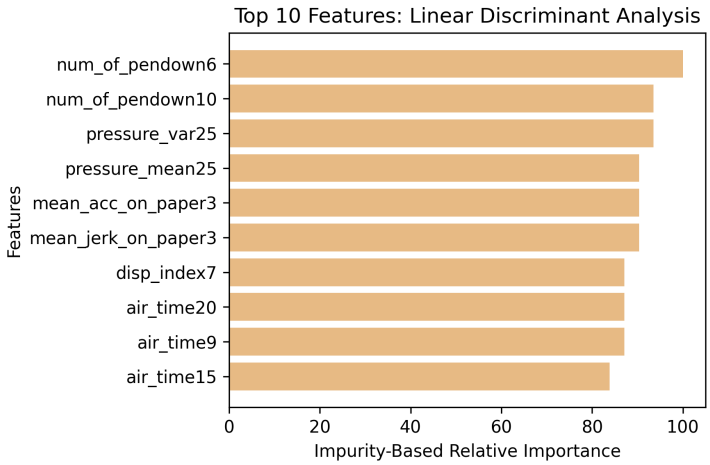
Table 10: Model performance measures







For the features, there is a variety of tasks and measurement types considered the most important. Though there is a slight preference for simpler tasks.

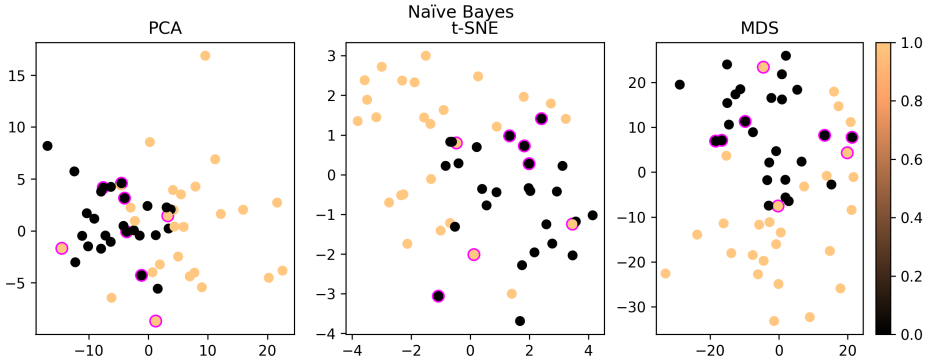


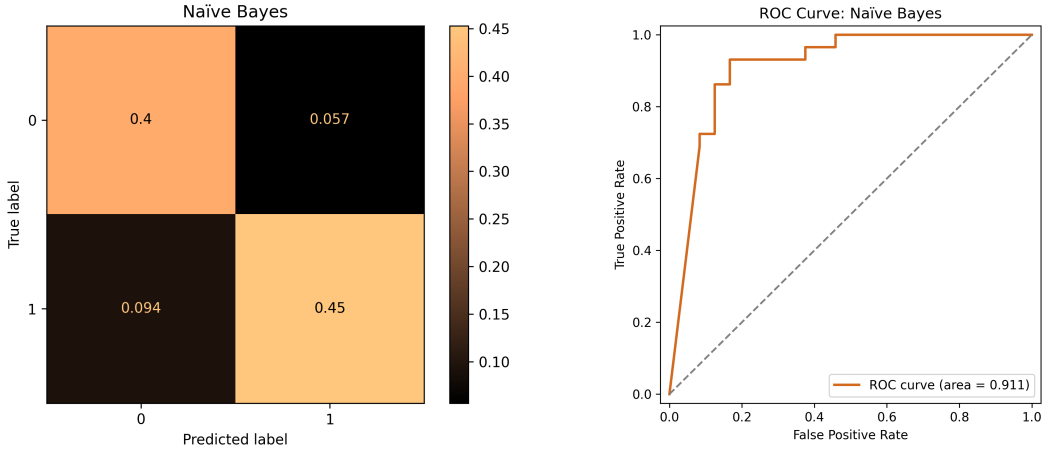
2.11 Naïve Bayes

This model performs surprisingly well on the testing accuracy, competing with some of the best models we have fitted so far. We suspect that the log-transformation might have normalized the data, allowing this simple model to take advantage of the Gaussian structure of the data.

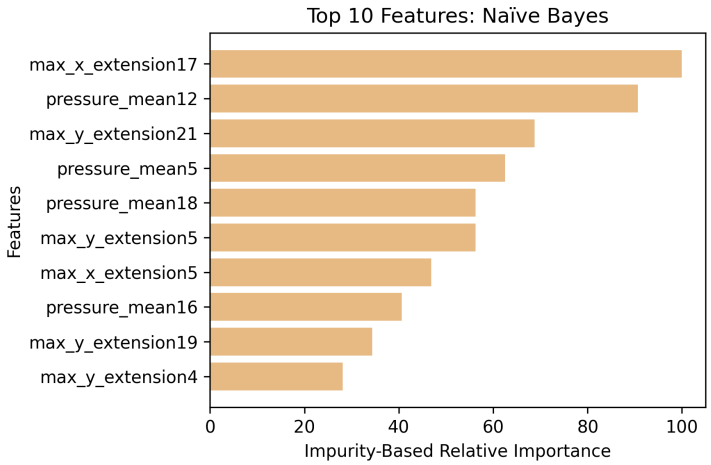
Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.8862	0.8533	0.9186	0.8766	
Test	0.8491	0.8276	0.8750	0.8571	0.9109

Table 11: Model performance measures





From the feature importance plot, the tasks with medium complexity seem to be the most important. This contrasts most of the other models which prefer either complex or simple questions.



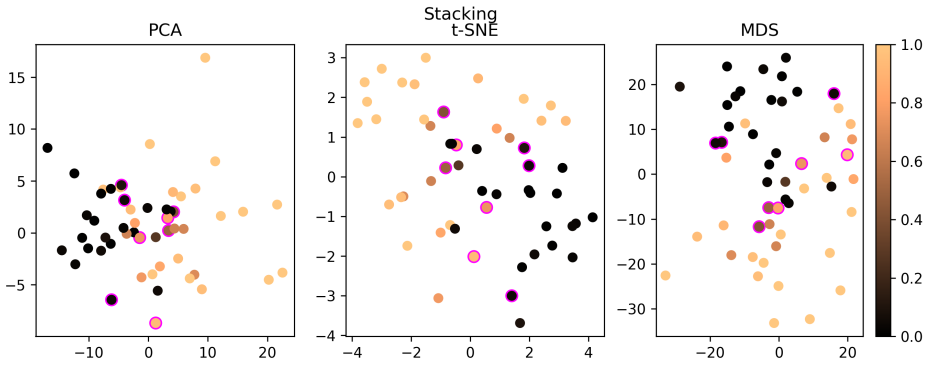
2.12 Stacking Model

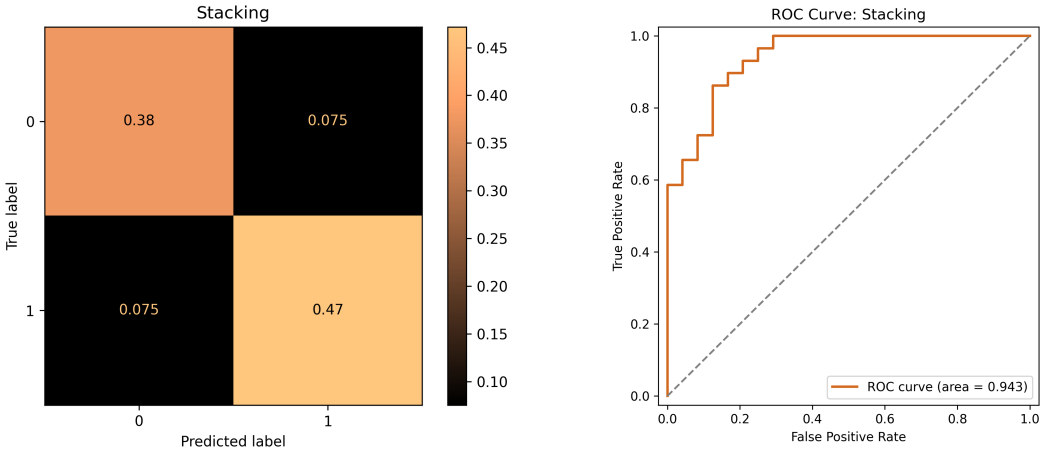
In order to construct a more powerful model from the existing models, we decided to fit two additional ensemble methods: a stacking model and a voting model. The stacking model is the more complex, which combines the predictions of the individual models to fit a meta-model to the predictions. Using the same hyperparameters for the existing models, we used a logistic regression as the final estimator. Note that we included all previous models except for the baseline.

This model has very good performance measures. Particularly, the F1 score and ROC AUC are very high compared to the other models.

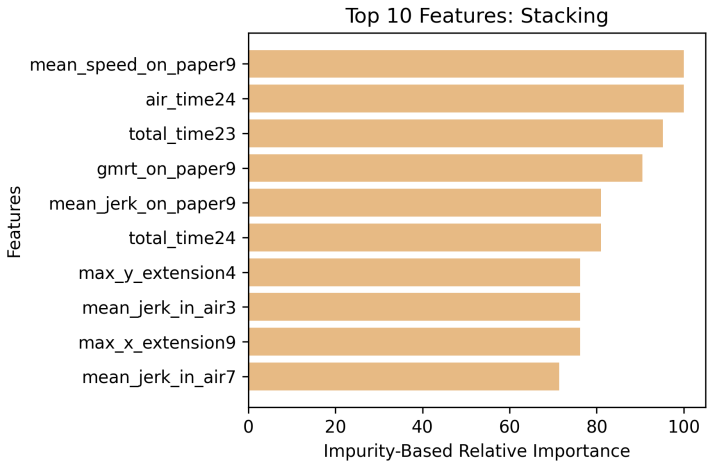
Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.92269	0.92667	0.91810	0.92163	
Test	0.84906	0.86207	0.83333	0.86207	0.94253

Table 12: Model performance measures





For the feature importance, while there is some variety in the type of measurement, tasks 9 and 24 stand out as showing high importance.



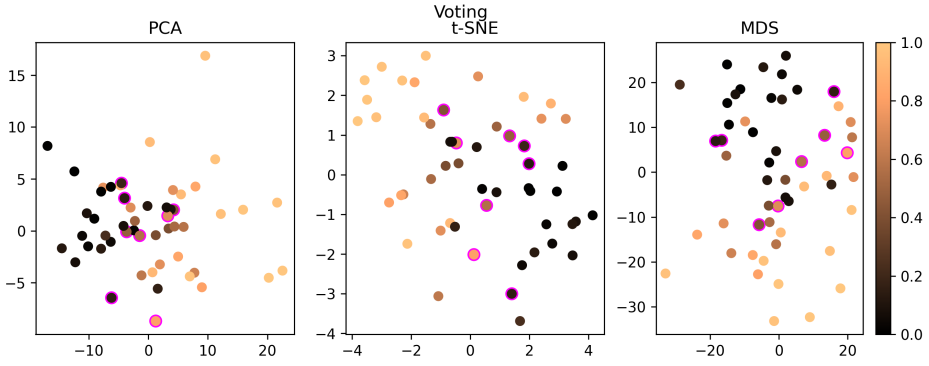
2.13 Voting Model

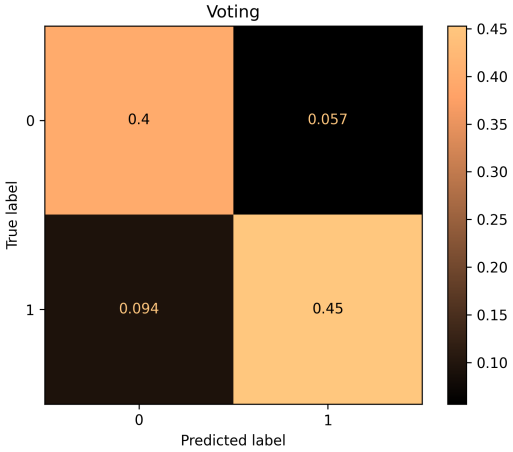
Our final model is a voting model, in which all the models (excluding baseline and stacking) generate predictions and vote on the label.

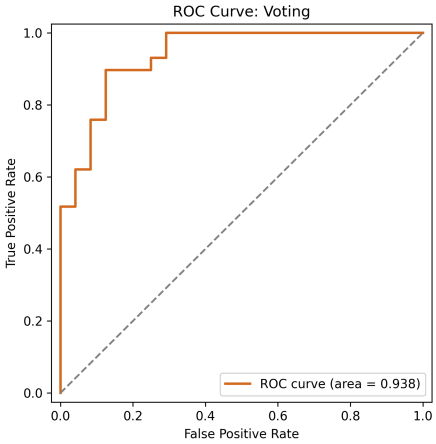
The performance measures are still very good compared to the other models, just not as good as the stacking model.

Type	Accuracy	Sensitivity	Specificity	F1	ROC AUC
CV	0.91115	0.90000	0.92143	0.90675	
Test	0.84906	0.82759	0.87500	0.85714	0.93822

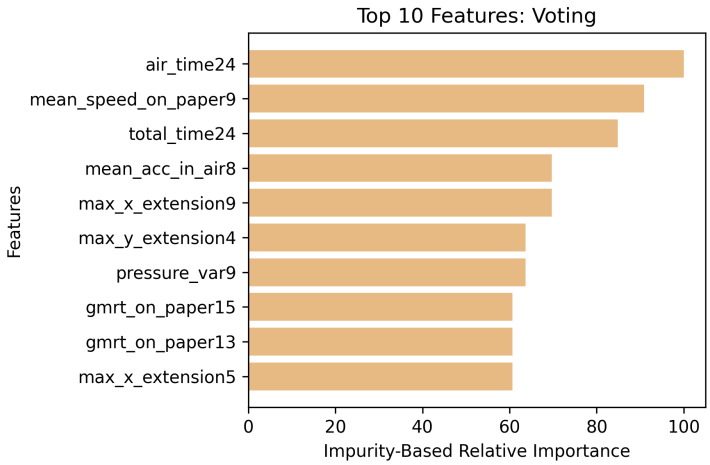
Table 13: Model performance measures







While the top 10 features show more variety than the stacking model, task 9 still appears to be very important.



3 Conclusions

3.1 Comparing Models

In the first table below, we have the cross-validation performance, then the following table has the test sample performance. Between the two, there is not a consistent top-performing model. If anything, the consistent low-performing models are the decision tree and the baseline (nearest centroid). But AdaBoost and MLP (the neural network) perform well in cross validation, yet only middle of the pack for the test sample. Alternatively, XGBoost, SVM, and naïve Bayes (the top three performers on the test set) also are middle of the pack for cross validation. If we were going for top performer on F1 score and/or ROC AUC, the clear winner would be the stacking model. Ultimately, letting test sample accuracy guide our decision, we would have XGBoost as our preferred overall model.

Model	Accuracy	Sensitivity	Specificity	F1 Score
AdaB	0.92577	0.90667	0.94429	0.92165
stack	0.92269	0.92667	0.91810	0.92163
MLP	0.91782	0.94667	0.88905	0.91964
vote	0.91115	0.90000	0.92143	0.90675
XGB	0.90910	0.90333	0.91429	0.90650
RF	0.90769	0.90333	0.91143	0.90421
SVM	0.90115	0.90000	0.90190	0.89859
NB	0.88615	0.85333	0.91857	0.87659
LR	0.85987	0.85333	0.86667	0.85273
LDA	0.83090	0.79667	0.86381	0.81919
NC	0.82321	0.77667	0.86905	0.80992
DT	0.79987	0.79667	0.80238	0.79539
KNN	0.75654	0.51000	1.00000	0.64623

Table 14: Ranked cross-validation performance measures

Model	Accuracy	Sensitivity	Specificity	F1 Score	ROC AUC
XGB	0.84906	0.82759	0.87500	0.85714	0.90230
SVM	0.84906	0.79310	0.91667	0.85185	0.93247
NB	0.84906	0.82759	0.87500	0.85714	0.91092
stack	0.84906	0.86207	0.83333	0.86207	0.94253
vote	0.84906	0.82759	0.87500	0.85714	0.93822
LR	0.83019	0.75862	0.91667	0.83019	0.88937
RF	0.83019	0.86207	0.79167	0.84746	0.91810
AdaB	0.83019	0.82759	0.83333	0.84211	0.93534
MLP	0.83019	0.82759	0.83333	0.84211	0.87931
KNN	0.77358	0.58621	1.00000	0.73913	0.79310
LDA	0.75472	0.72414	0.79167	0.76364	0.84483
DT	0.73585	0.75862	0.70833	0.75862	0.79239
NC	0.71698	0.68966	0.75000	0.72727	0.71983

Table 15: Ranked test sample performance measures

For clarity, the top 6 performing models by test data measure are seen in the table below. Interestingly, the stacking model performs very well on the F1 score and ROC AUC, but not as well on the accuracy. Our preferred model, XGBoost, performs very well on accuracy and F1 score, but does not even make the top 6 for the ROC AUC. Other consistently high-performing models are SVM and Naïve Bayes, which make up the top 3 accuracies, and perform well on the other measures. Finally, the two combined models (stacking and voting) perform the best on AUC ROC, which speaks to how the models are trained—they seek to fit a meta-model to the predictions from other models, so their ROC performance will generally be higher by design.

Ranking	Accuracy	F1 Score	ROC AUC
1	XGB	stack	stack
2	SVM	XGB	vote
3	NB	NB	AdaB
4	stack	vote	SVM
5	vote	SVM	RF
6	LR	RF	NB

Table 16: Top performing models on test sample

3.2 Important Features

For a feature space of 450 vectors, choosing the top performing features is a difficult task. However, from analyzing the top 10 features for each model, we can gain an understanding of some which are consistently of high importance. First are features having to do with time. In fact, our preferred model, XGBoost, has 5 of the top 10 just being "total time" for certain tasks, and 3 of the 10 are "paper time," which has to do with time spent with pen to paper (as opposed to the writing utensil being in the air). One thing to notice is that taking longer to do a task might be associated with having to wait, process, and even plan future movement.

Consistently, tasks 9, 11, 23, and 24 show high importance across multiple models. Task 9 involves writing two letters in cursive. Task 11 involves writing a word. Task 23 involves writing a dictated telephone number. And task 24 is drawing a clock face. With each of these tasks, more thinking is required compared to other tasks, in addition to more complex motor components. This is one of many possible explanations for why these tasks are more valuable when diagnosing Alzheimer's disease.

3.3 Limitations & Future Work

In all studies, one's analysis is only as good as their tools of measurement; as such, testing conditions and limitations on technology sensitivity might have impacted how the information was recorded. As the test was administered in Italian, there are potential language biases. In particular, dictation tasks and paragraph copying might have variability. And more importantly, the ability to write using the Latin alphabet and Arabic numerals might not be the same as writing in a character- or script-based language, like Chinese or Arabic.

There are many possibilities for future work in this setting. Using this data, one could try to construct a combined classifier which takes the best classifier *for each task*, much like how the researchers did in the original study. Doing so would require much more work in the name of robustness, especially in constructing a repeated train-test split in addition to repeated cross validation.

Future work in this area might look to diagnose Alzheimer's disease based purely on images of handwriting alone. While such a project would require much more computational power, the applicability of such a classifier would go beyond this current data, which is based on kinetic measures of writing and not the writing itself.

4 Bibliography

Armstrong, Melissa J., Irene Litvan, Anthony E. Lang, Thomas H. Bak, Kailash P. Bhatia, Barbara Borroni, Adam L. Boxer et al. "Criteria for the diagnosis of corticobasal degeneration." Neurology 80, no. 5 (2013): 496-503. https://doi.org/10.1212/WNL.0b013e31827f0fd1.

Cilia, Nicole D., Giuseppe De Gregorio, Claudio De Stefano, Francesco Fontanella, Angelo Marcelli, and Antonio Parziale. "Diagnosing Alzheimer's Disease from on-Line Handwriting: A Novel Dataset and Performance Benchmarking." Engineering Applications of Artificial Intelligence 111 (May 2022): 104822. https://doi.org/10.1016/j.engappai.2022.104822.

Impedovo, Donato, Giuseppe Pirlo, and Gennaro Vessio. "Dynamic Handwriting Analysis for Supporting Earlier Parkinson's Disease Diagnosis." Information 9, no. 10 (October 3, 2018): 247. https://doi.org/10.3390/info9100247.