

Aircraft Delay Prediction

Tyler Cady, Samreen Farooqui, Uzma Khan, Akash Ratheesh, Maxime Remy

Flight delays have significant economic and logical implications, negatively affecting passengers, air-lines, and airports. According to the U.S. Federal Aviation Administration (FAA), flight delays cost the U.S. economy an estimated \$33 billion annually[1]. Predicting delays can help to mitigate the repercussions caused by them such as missed connecting flights and rescheduling conflicts for airlines. However, predicting such delays can get challenging due to their correlation with several other events and factors such as weather, security, and chained delay events caused by previous delays[2]. In recent years, machine-learning techniques have become increasingly popular and promising when predicting such correlated events. In previous work[3], the authors have explored several supervised learning techniques (Random Forest, KNN, Gradient Boosted Trees, etc) and found them to be very effective in predicting delays with high accuracy. In a similar spirit, other works have looked into various other machine-learning techniques including hybrid models [4, 5, 6] that use clustering-based methods along-side regression and other statistical techniques to predict delays reliably. The goal of our project is to develop machine-learning models that can predict flight delays by leveraging historical flight data and weather forecasts to compare the performance of these models. Accurately predicting these flight delays will allow airport and carrier personnel to anticipate issues and enable airports to optimize resources when readjusting schedules.

For the flight delay problem, we have rich statistical data available regarding factors leading to the delays. The Bureau of Transportation Statistics has been collecting and publishing [7] datasets that include historical flight delays and cancellations dating back to 2003. The dataset consists of several data points including airport name, carrier, cause of delay (eg. weather, carrier delay), etc. This will be our primary dataset source as it is freely and readily available. Additionally, the historical weather data, specifically near the vicinity of the airport is also available through the Aviation Weather Center [8].

Due to the large amount of data available, data preprocessing is crucial to extract meaningful data and facilitate faster training. When analyzing flight and weather data, the main preprocessing methods required are data cleansing, dimensionality reduction, normalization, and feature engineering. Normalizing the various features is required as weather data and flight data will be at different scales. Similarly, data cleansing for weather and flight data is required to account for missing data and other inconsistencies. Additionally, previous works [2] have shown that binning the delay time data points, i.e. making delay groups based on time windows, tends to provide more accurate predictions.

There are a number of machine learning techniques that could be applied to predict flight delays. We have identified four machine-learning techniques we would like to explore for our project. Previous works[3] indicate that

Introduction

Contributions Table

Data Pre-Processing

Dbscan

Random Forest

SVM

Neural Network

Results

Milestones

Aircraft Delay Prediction

CS7641 Project - Group 23

supervised learning tends to work better for such flight delay predictions. We plan to implement supervised learning techniques like Random Forest, Support Vector Machines (SVM), and train a fully connected neural network model to classify whether a flight is delayed or not. Regarding unsupervised learning, we would like to implement a clustering-based technique, in our case DBSCAN to train a model. To evaluate and compare the models, we will use quantitative metrics such as accuracy, recall, precision, and F1 score.

References

- [1] M. Ball, C. Barnhart, M. Dresner, M. Hansen, K. Neels, A. Odoni, E. Peterson, L. Sherry, A. Trani, and B. Zou, "Total delay impact study: a comprehensive assessment of the costs and impacts of flight delay in the united states," 2010.
- [2] J. Chen and M. Li, "Chained predictions of flight delay using machine learning," in AIAA Scitech 2019 forum, 2019, p. 1661.
- [3] Y. Tang, "Airline flight delay prediction using machine learning models," in Proceedings of the 2021 5th International Conference on E-Business and Internet, 2021, pp. 151–154.
- [4] M. Dai, "A hybrid machine learning-based model for predicting flight delay through aviation big data," Scientific Reports, vol. 14, no. 1, p. 4603, 2024.
- [5] M. Mamdouh, M. Ezzat, and H. A. Hefny, "A novel intelligent approach for flight delay prediction," Journal of Big Data, vol. 10, no. 1, p. 179, 2023.
- [6] R. K. Jha, S. B. Jha, V. Pandey, and R. F. Babiceanu, "Flight delay prediction using hybrid machine learning approach: A case study of major airlines in the united states," arXiv preprint arXiv:2409.00607, 2024.
- [7] DOT. Bureau of Transportation Statistics (BTS), Airline On-Time Statistics and Delay Causes, year = 2024, url = https://www.transtats.bts.gov/ot_delay/ot_delaycause1.asp.
- [8] N. W. Service. Aviation Weather Center, year = 2024, url = <https://aviationweather.gov/>.

Aircraft Delay Prediction

Proposal Contribution

Name	Contributions
Tyler Cady	Slides and Video
Samreeen Farooqui	Report - Introduction, problem, and motivation
Uzma Khan	Report - Potential results and discussion
Akash Ratheesh	Report - Literature review and methods
Maxime Remy	Slides for Video

Midterm Contribution

Name	Contributions
Tyler Cady	Data pre-processing, model coding, and report
Samreeen Farooqui	Results evaluation, analysis, and report
Uzma Khan	Data pre-processing, model coding, and report
Akash Ratheesh	Data sourcing, results evaluation, analysis, cleaning, and report
Maxime Remy	Data sourcing, results evaluation, analysis, cleaning, and report

Final Report

Name	Contributions
Tyler Cady	Slides, video, cleaning, and report
Samreeen Farooqui	Random Forests and report

[Introduction](#)[Contributions Table](#)[Data Pre-Processing](#)[Dbscan](#)[Random Forest](#)[SVM](#)[Neural Network](#)[Results](#)[Milestones](#)

Aircraft Delay Prediction

CS7641 Project - Group 23

Name	Contributions
Uzma Khan	Model comparisons and report
Akash Ratheesh	SVM and report
Maxime Remy	Neural Network, slides, and report

🕒 Introduction

📄 Contributions Table

🔍 **Data Pre-Processing**

🔗 Dbscan

🌲 Random Forest

🌀 SVM

🧠 Neural Network

📊 Results

🏆 Milestones

Aircraft Delay Prediction

CS7641 Project - Group 23

Data Pre-Processing

The first step in applying the proposed Machine Learning (ML) algorithms is to acquire and process a dataset encapsulating our problem. In this case, the dataset needs to be comprehensive and include flights and information about them (airlines, dates, destination, departure, etc.) and indicate whether they arrived at their destination delayed (this will act as our label for supervised ML algorithms). The data has been acquired through Kaggle [1,2] and includes a comprehensive set of features and labels that can be leveraged.

Here are the features that are available in the dataset.

Feature	Description
FL_DATE	Flight Date (yyyymmdd)
AIRLINE_CODE	Unique Carrier Code. When the same code has been used by multiple carriers, a numeric suffix is used for earlier users, for example, PA, PA(1), PA(2). Use this field for analysis across a range of years.
DOT_CODE	An identification number assigned by US DOT to identify a unique airline (carrier). A unique airline (carrier) is defined as one holding and reporting under the same DOT certificate regardless of its Code, Name, or holding company/corporation.
FL_NUMBER	Flight Number
ORIGIN	Origin Airport
ORIGIN_CITY	Origin Airport, City Name
DEST	Destination Airport
DEST_CITY	Destination Airport, City Name
CRS_DEP_TIME	CRS Departure Time (local time: hhmm)
DEP_TIME	Actual Departure Time (local time: hhmm)
DEP_DELAY	Difference in minutes between scheduled and actual departure time. Early departures show negative numbers.
TAXI_OUT	Taxi Out Time, in Minutes

Feature	Description
WHEELS_OFF	Wheels Off Time (local time: hhmm)
WHEELS_ON	Wheels On Time (local time: hhmm)
TAXI_IN	Taxi In Time, in Minutes
CRS_ARR_TIME	CRS Arrival Time (local time: hhmm)
ARR_TIME	Actual Arrival Time (local time: hhmm)
ARR_DELAY	Difference in minutes between scheduled and actual arrival time. Early arrivals show negative numbers.
CANCELLED	Cancelled Flight Indicator (1=Yes)
CANCELLATION_CODE	Specifies The Reason For Cancellation
DIVERTED	Diverted Flight Indicator (1=Yes)
CRS_ELAPSED_TIME	CRS Elapsed Time of Flight, in Minutes
ELAPSED_TIME	Elapsed Time of Flight, in Minutes
AIR_TIME	Flight Time, in Minutes
DISTANCE	Distance between airports (miles)
DELAY_DUE_CARRIER	Carrier Delay, in Minutes
DELAY_DUE_WEATHER	Weather Delay, in Minutes
DELAY_DUE_NAS	National Air System Delay, in Minutes
DELAY_DUE_SECURITY	Security Delay, in Minutes
DELAY_DUE_LATE_AIRCRAFT	Late Aircraft Delay, in Minutes

Among those features: FL DATE, DOT CODE, FL NUMBER, ORIGIN, DEST, CRS DEP TIME, CRS ARR TIME, CRS ELAPSED TIME, DISTANCE, DEP DELAY, DELAY DUE WEATHER, CANCELLED, DIVERTED, DELAY DUE NAS, DELAY DUE CARRIER, DELAY DUE LATE AIRCRAFT, DELAY DUE SECURITY, AIR TIME has been selected. The others have been deemed either redundant or too noisy to be used. It has been decided that the labels for the initial tests will be combined into two DELAYED and ON TIME. This is so we can apply the properties of binary classification in our project. Depending on the results, this label might be split up again. Furthermore, as part of the clean-up, the dataset of NaN values is cleaned (for example, DELAY labels were empty in case of an on-time flight).

References

- [1] P. Zelazko. Flight Delay and Cancellation Dataset (2019-2023), year = 2023, url = <https://www.kaggle.com/datasets/patrickzel/flight-delay-and-cancellation-dataset-2019-2023?resource=download>,.
- [2] On-Time : Reporting Carrier On-Time Performance (1987-present), year = 2024, url = https://www.transtats.bts.gov/DL_SelectFields.aspx?gnoyr_VQ=FGJ&QO_fu146_anzr=b0-gvzr

[Introduction](#)[Contributions Table](#)[Data Pre-Processing](#)[Dbscan](#)[Random Forest](#)[SVM](#)[Neural Network](#)[Results](#)[Milestones](#)

Aircraft Delay Prediction

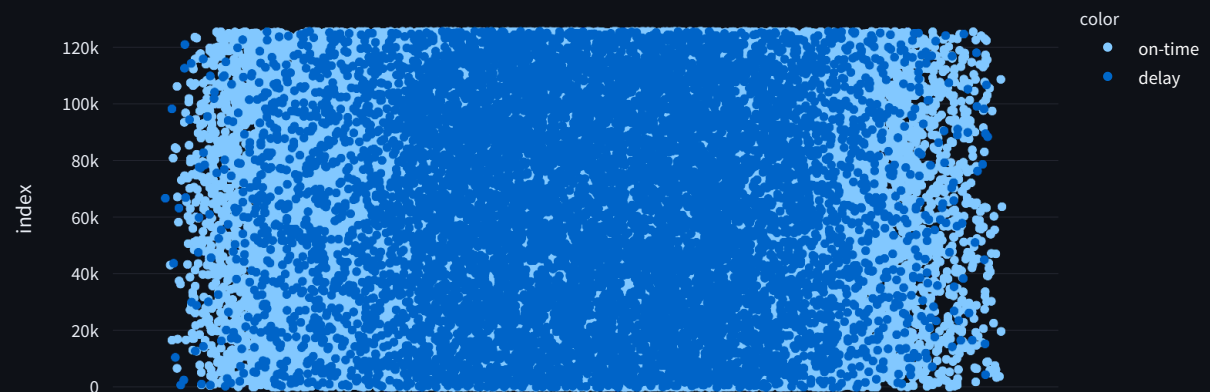
CS7641 Project - Group 23

Method 1: DBSCAN

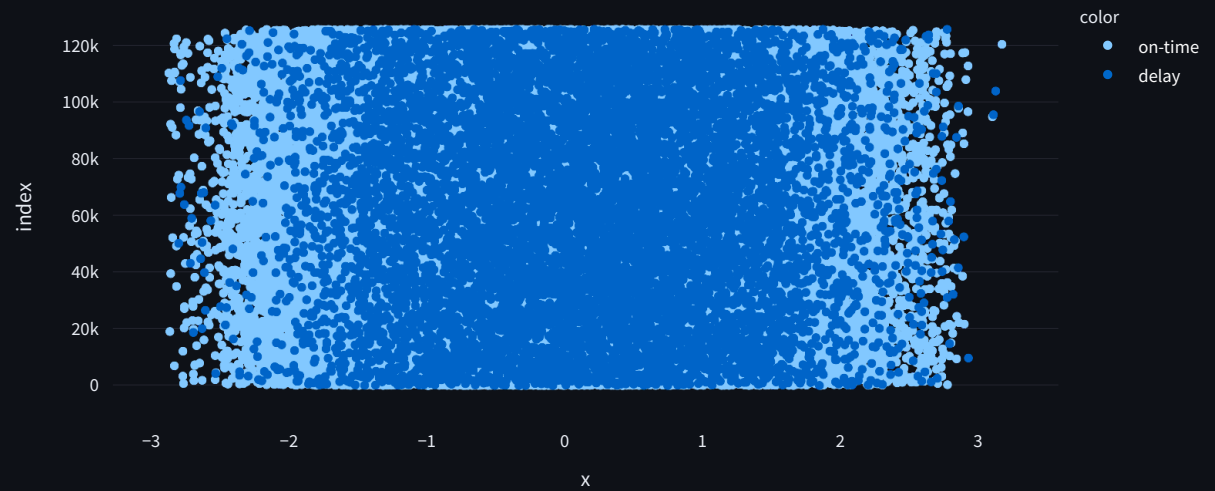
DBSCAN is typically used as a preprocessing method to identify clusters or groups of correlated data points. However, in specific cases such as binary classification (whether a flight is late or not), it is also a viable method to classify the data into clusters. In our implementation, we use DBSCAN to cluster the data points into several clusters. Then using the true labels, we can identify the clusters that predominantly contains delayed flights and vice versa. When a new data is available, to perform inference, we try to identify the closest cluster to the given points. If the points belongs to a cluster which was earlier classified as 'delayed', we can predict that the current flight would be delayed too. In our initial experimentation, we tried performing DBSCAN on the entire data set (all airports, from year 2019-2023), but this resulted in a very low accuracy. Subsequently, we tried only Year 2023 data, but we were having similar issues. We identified that when considering all the airports, there are several issues preventing us from getting good accuracy. The data contains a large amount of noise in the data and, there is an imbalance between the number of delayed flight data and non-delayed flight data. Also, the features with higher variances in our cases does not help much in separating classes, which results in bad or improper clustering. Due to this reason, we have opted to fit our current model with data only from Atlanta Airport (From year 2019-2023).

PCA was also applied to see if the problem could have its dimension reduced and be visualized. However, even with a requirement of retaining only 0.9 of the variance, most are still being kept. This indicates that the data set can not be simplified using the variance as our metric. This is further proven when visualizing the data against the label using the two features with the highest variance.

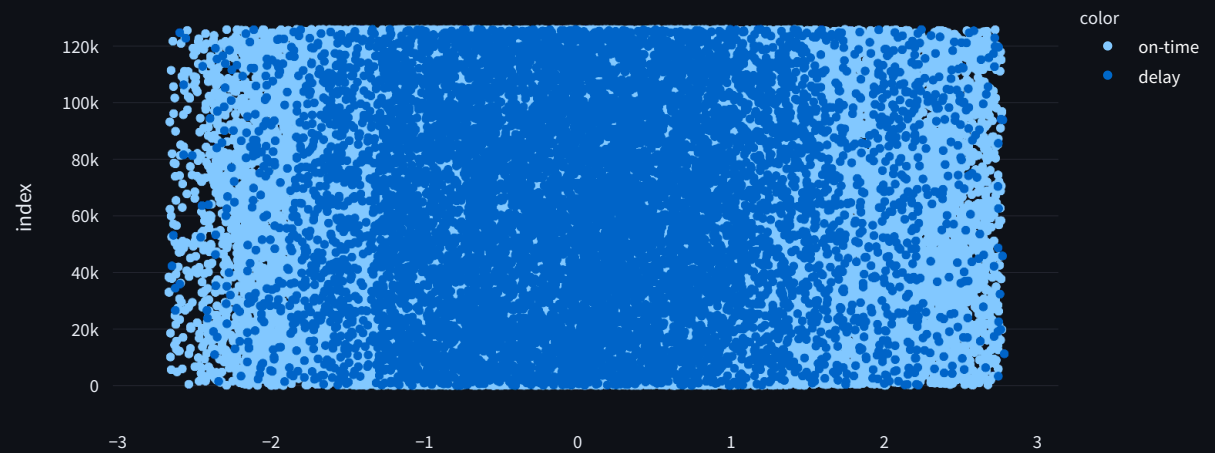
PCA 0th and 1th strongest variance features

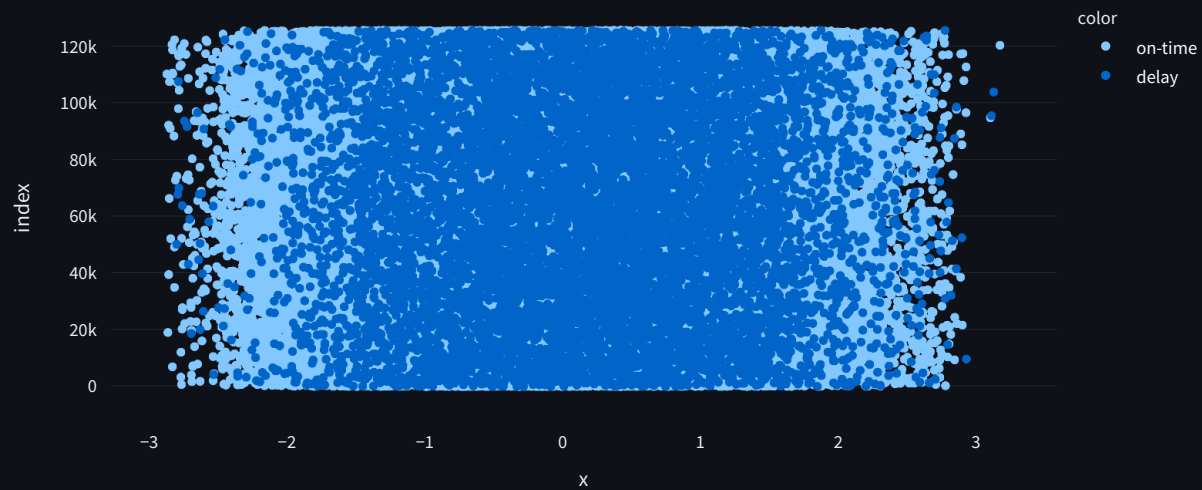
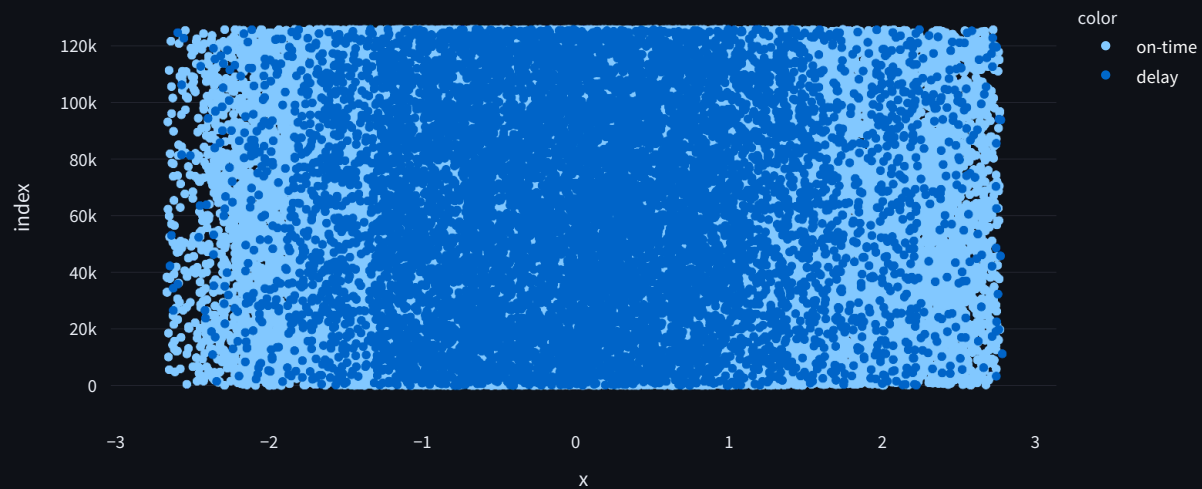


PCA 0th and 2th strongest variance features

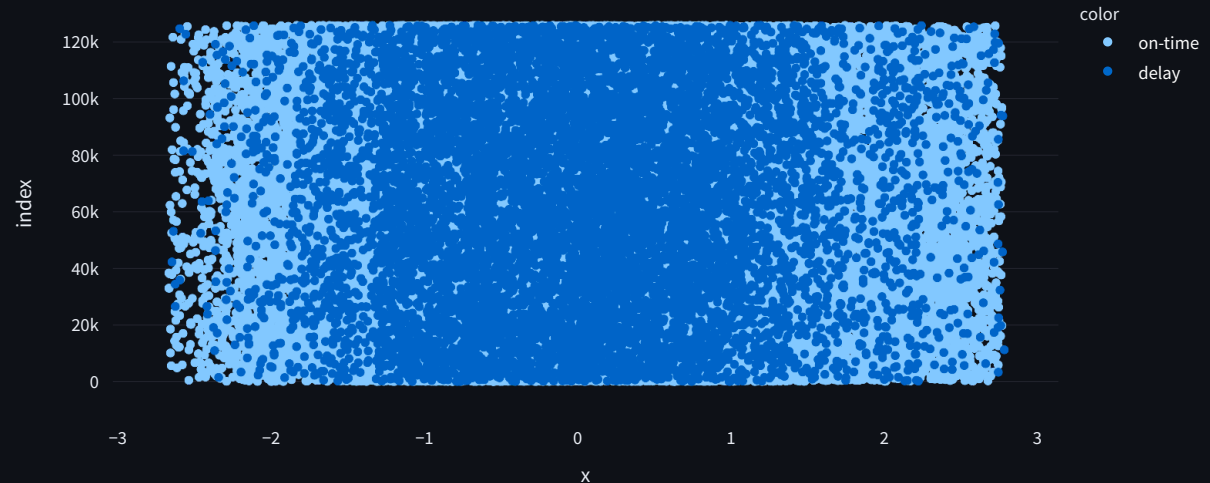


PCA 0th and 3th strongest variance features



PCA 1th and 2th strongest variance features**PCA 1th and 3th strongest variance features**

PCA 2th and 3th strongest variance features



Furthermore, a quick survey through the dataset indicates a heavy imbalance between positive labels (delayed) and negative labels (no-delays). To compensate for it, the testing dataset has been tweaked to ensure parity between the two labels. This parity will ensure that metrics such as F1-scores do not suffer from an overwhelming higher number of FN compared to TP and FP.

Results

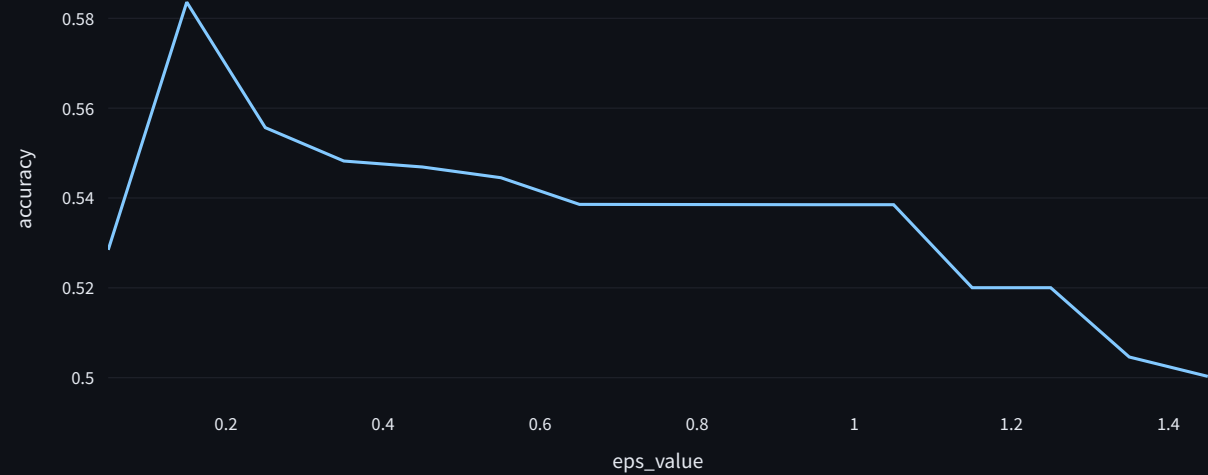
After training our DBSCAN model on our ATL training data (composed of 80% of our total ATL data) and then using our DBSCAN model with our ATL test data (composed of 20% of our total ATL data), we further generated both supervised and unsupervised metrics to help us validate how well our model worked and if this data set can actually be leveraged for supervised learning to predict future delays. For supervised metrics, we used given labeled data that determined the actual delay in parallel with a generalization/assignment of each cluster based on their mean delay value to utilize a confusion matrix to present us with an F1-score among other metrics.

For unsupervised metrics, we calculated a Silhouette score, which helps us to indicate the quality of the clustering. This is done through the generic formula of $SS = \text{mean}[(b-a)/\max(a,b) \forall \text{ datapoints}]$, where a is the mean distance between a data point to all other data points in the same cluster and b is the mean distance between a data point and

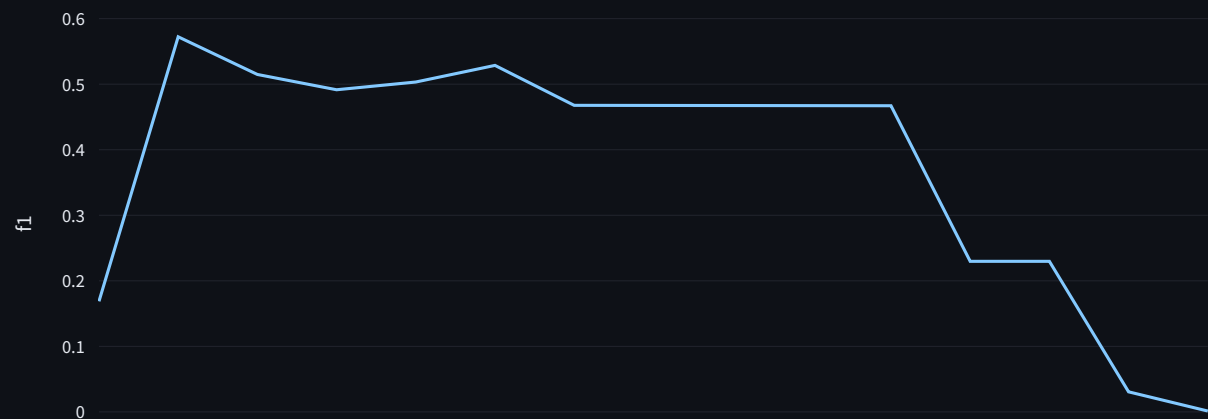
all other points in the next nearest cluster. Thus, a value of -1 indicates the worst clustering, a value of 1 indicates the best clustering, and a value of 0 means for a quite indifferent clustering.

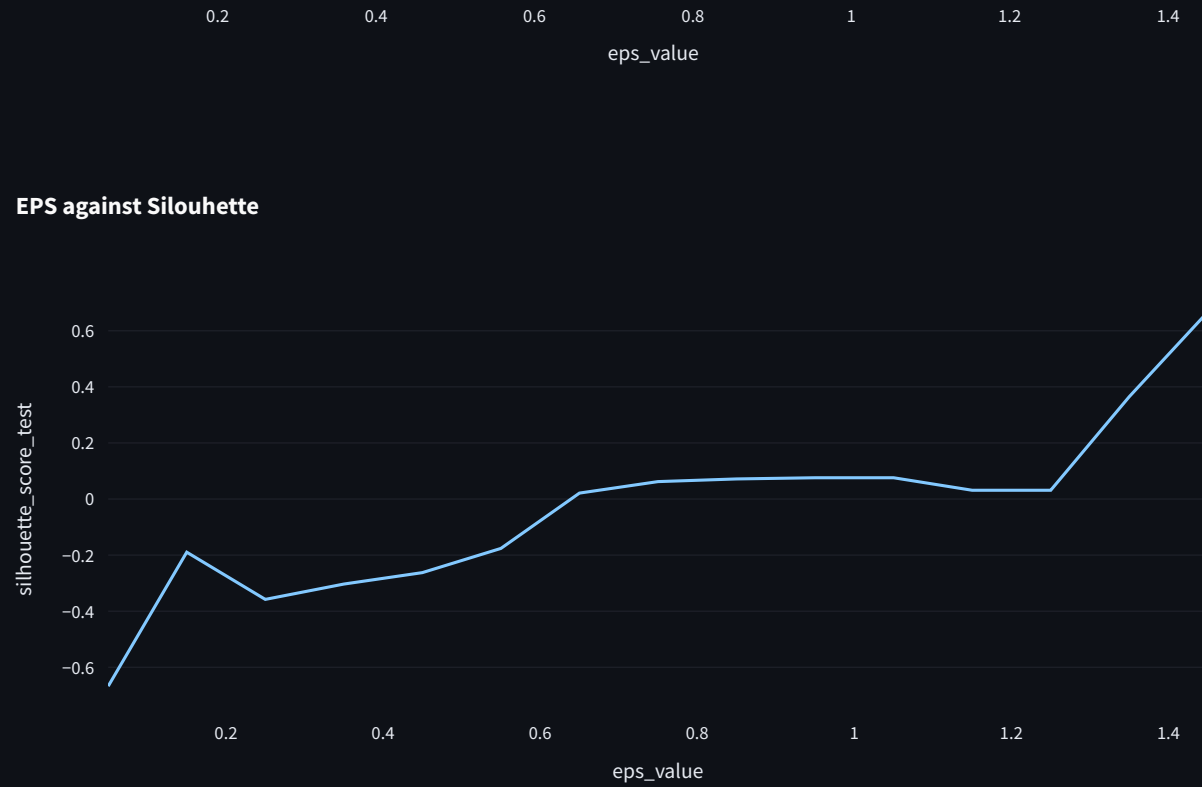
Using these metrics as our evaluation we sweep over the eps distance parameter of the DBSCAN and got the following results:

EPS against Accuracy



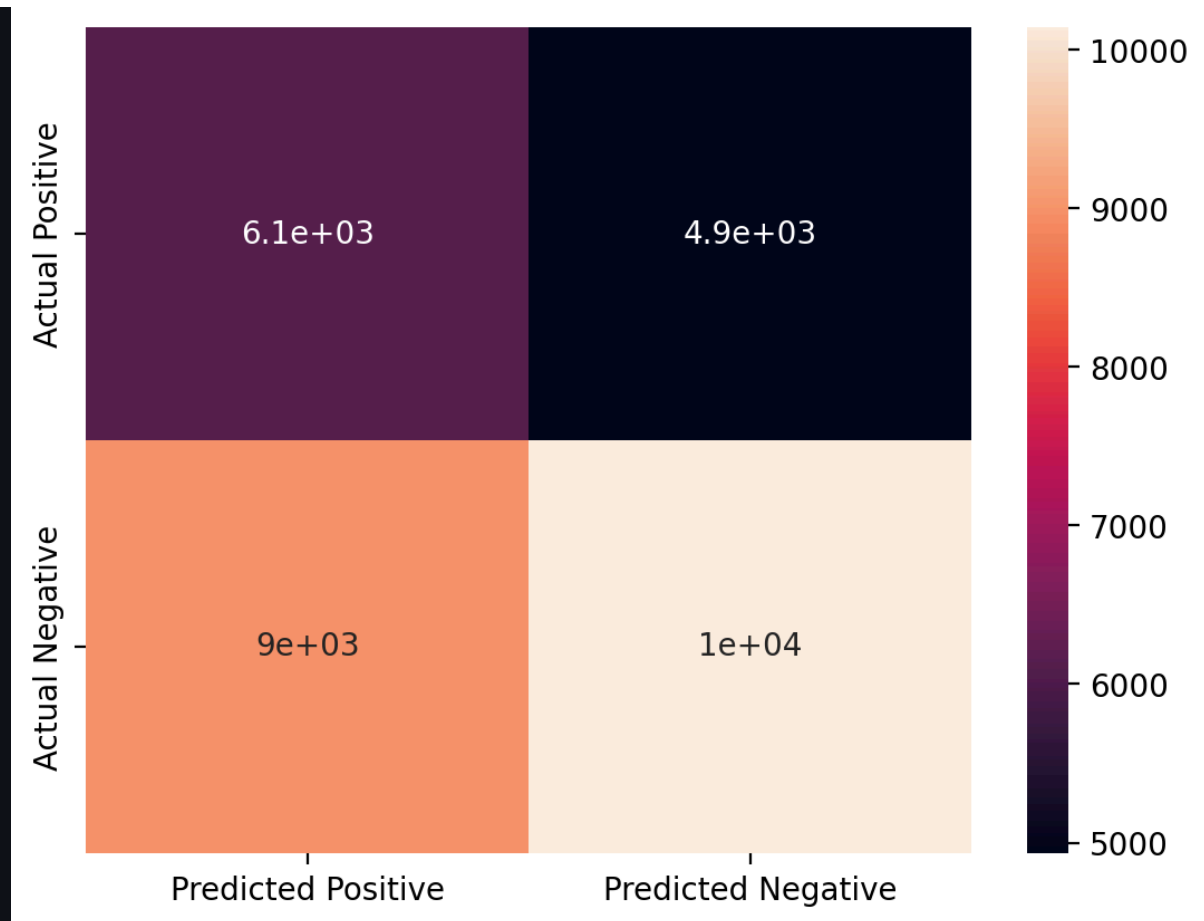
EPS against F1-score



EPS against Silouhette

Based on this curves the most promising parameter is $\text{eps} = 0.95$. Using this parameter we get the following performances:

Confusion Matrix (n = 30,150)



The above confusion matrix can provide us with the following values for our metrics (note: 0 is the worst value and 1 is the best value for all of the following):

Metrics

	Metric
Accuracy (TP)	0.553
Accuracy (TN)	0.530
Balanced Accuracy	0.539
Precision	0.552
Recall (Sensitivity)	0.405
Specificity	0.673
F1-Score	0.467

Unfortunately, as demonstrated in the PCA's visualization, the dataset does not seem to form cluster around specific labels resulting in disappointing results. The accuracy score value of 0.539 tells us that we have a model that guess correctly half the time. In other world its performance is around the same as randomly picking a label for a new data point. The precision value, 0.552, tells a similar grim story where the model is not particularly precised resulting in intertwined labels prediction. The recall value, 0.405, is even worse it is because it depends on the number of FN to calculate its scores. Based on the confusion matrix it is clear that the model is biased towards assigning a negative label even though the testing dataset has been balanced so an equal amount of positive and negative labels exist within. The specificity, 0.673, is the best metrics for the same reason. Since the number of TN is its numerator it benefits from the bias. And finally, the F1-Score, 0.467, has a dependency on the number of FN which results in the negative skewing of its score. Thus based on the overall score, it can be concluded that a clustering approach to predict flight delay will not work. This is due to the fact that labels do not seem to have natural clusters DBSCAN could leverage to assign a label.

For the best eps value prediction wise, we calculated a Silhouette score of 0.067222 for our model. This value means that our data set have entangled data and cannot be clustered in a satisfying manner. This score can be increased by raising the eps-value as seen in the figure below. However, this results in clusters where no meaningful prediction can be done as the accuracy seems to converge to 0.5 which means the result is the same as coin toss. This most likely due to the fact that at this point the labels are perfectly mixed within the formed clusters.

Next Steps

So overall, the DBSCAN indicates that the dataset cannot leverage a clustering approach to predict the flight delays. It seems that issue stems from the data being deeply entangled with each others. Due to this results it is confirmed the other approaches will need to leverage another method to classify correctly the delays. We are currently working on

implementing supervised learning techniques such as Random Forest, SVM, and a neural network model. These supervised learning techniques generally work well in identifying hidden correlations and predicting classes better.

[Introduction](#)[Contributions Table](#)[Data Pre-Processing](#)[Dbscan](#)[Random Forest](#)[SVM](#)[Neural Network](#)[Results](#)[Milestones](#)

Aircraft Delay Prediction

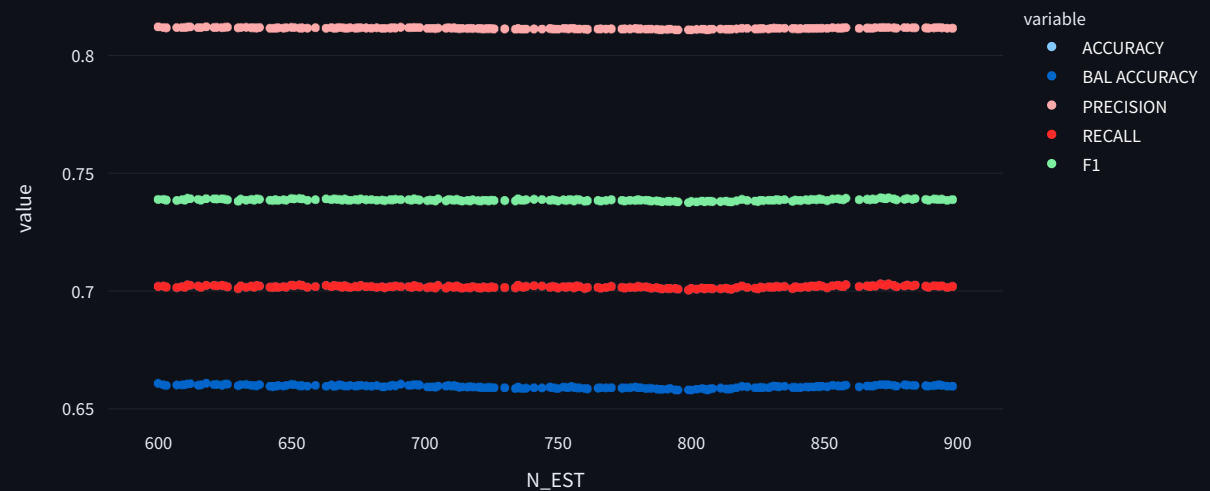
CS7641 Project - Group 23

Method 2: Random Forest

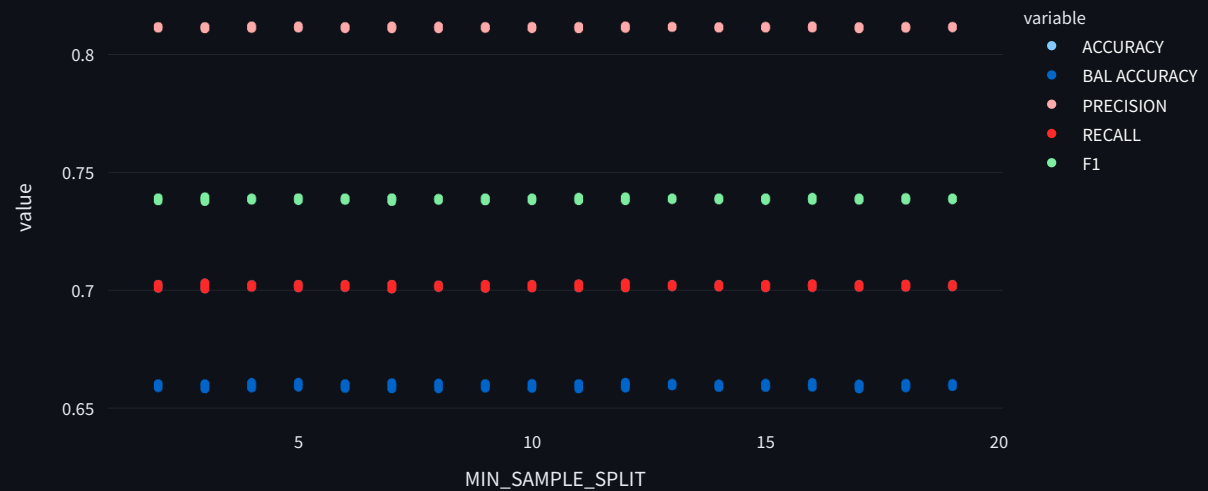
Random Forest classification was a model we implemented for predicting flight delays at Atlanta Airport (ATL) using data from 2019-2023. This machine learning approach is specifically well-suited for flight delay prediction due to its ability to handle complex, non-linear relationships between various flight attributes and delay outcomes. Random Forest operates by constructing multiple decision trees that are each trained on different bootstrap samples of the data, where each tree independently learns patterns that contribute to flight delays. The final prediction is determined by aggregating the predictions of all trees, which makes it robust against overfitting and capable of capturing intricate patterns in flight operations. In our implementation, we focused on ATL data to ensure consistency and minimize the noise that might arise from different airports' varying operational characteristics, weather patterns, and traffic volumes.

Using Weights and Biases, many different hyperparameters were tested, by setting a number of iterations from 75-300 and by ranges of numbers for each hyperparameter to randomly be chosen for. The following are the results from the random search.

Number of Estimator vs Performance



Min Sample Split vs Performance



Min Sample Leaf vs Performance



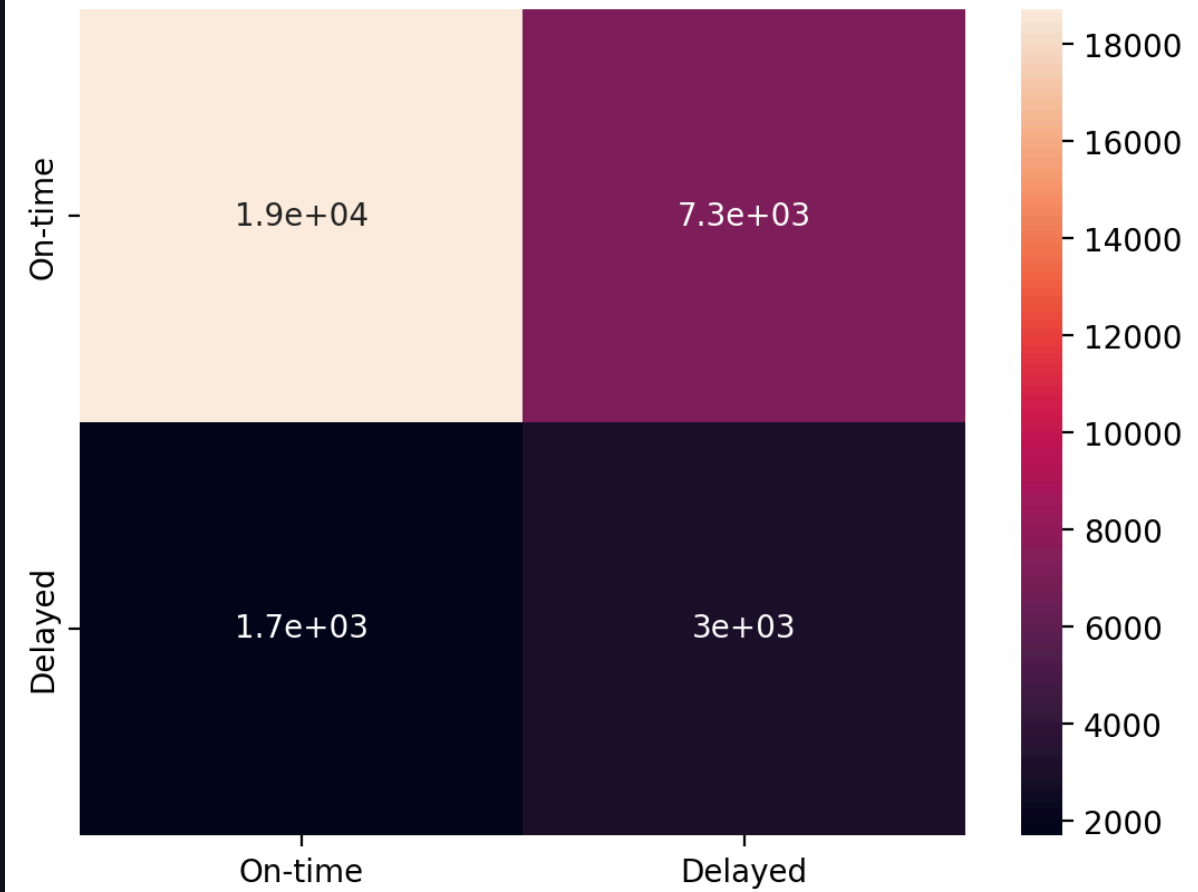
Through extensive experimentation involving 300 iterations of hyperparameter tuning, we identified the optimal configuration for our Random Forest model: 874 estimators (decision trees), a maximum depth of 15, minimum

samples split of 12, and minimum samples leaf of 4. These parameters were chosen after observing their consistent performance on the charts on Weights and Biases. The model achieved an accuracy of 0.701 and an F1 score of 0.73, demonstrating balanced performance between precision and recall in delay prediction. The balanced accuracy of 0.66 shows that the model maintains reasonable performance across both delayed and non-delayed flight classifications. The precision consistently remained above 0.81, indicating a strong reliability in positive delay predictions. The precision of 0.81 indicates that when our model predicts a flight will be delayed, it is correct approximately 81% of the time, making it a reliable tool for airlines and passengers to anticipate potential delays. Below is the summary of the classification's performance and its confusion matrix.

Metrics

	Metric
Accuracy (TP)	0.700
Balanced Accuracy	0.650
Precision	0.810
Recall (Sensitivity)	0.700
F1-Score	0.730

Confusion Matrix (n = 30,714)

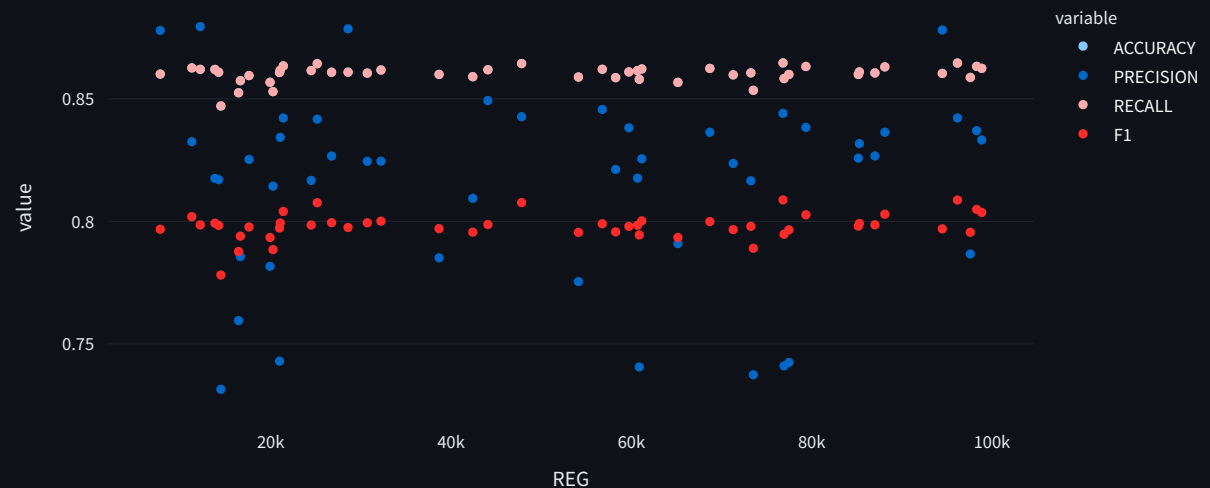


Method 3: SVM Classifier

Support Vector Machines (SVM) is a very popular tool in supervised machine learning techniques where the decision boundaries are estimated using support vectors based on the classes and features of the data. We are using the scikit-learn's SVM classifier to classify our 2019-2023 Atlanta flight data. The hyperparameters for all the runs were found using random grid search. For the classifier we tried linear, RBF and polynomial kernels where RBF (best) and linear tend to give better results. Initially, we performed the classification using binary class - one to indicate delayed, cancelled and diverted flight and one class to indicate nondelay. We were able to accuracy of 72% but the balanced accuracy and the F1 score were not in the acceptable range.

We also tried multiclass where we have delayed, cancelled, diverted and nondelay as the class labels. Training the SVM classifier with a multiclass gave us a better result with an accuracy of ~80% and weighted F1 score of 0.78. Since our data set is very large and contains a lot of noise, we explored using the stochastic gradient version of the SVM classifier. This gave us an accuracy of ~82% and F1 score of 0.79. Finally, we tried training the classifier with data transformed using Nystroem[1] method. The Nystroem method makes an approximation of the RBF kernel using the data set and this can be used to train a linear SVM. This gave us the best results. The below image shows one of the random runs we performed to tune the hyper parameters.

Regularizer vs Performance



Aircraft Delay Prediction

CS7641 Project - Group 23

Number of Component vs Performance



Based on this curves the most promising parameter is regularizer = 56400 and number of components = 1741. Using this parameter we get the following performances:

Confusion Matrix (n = 30,714)









Metrics

	Metric
Accuracy (TP)	0.862
Precision (Weighted)	0.827
Recall (Weighted)	0.861
F1-Score (Weighted)	0.801

Furthermore, we also performed the same training with random search on the entire dataset (for all airport, 2019-2023). Since we have around 3 million data points, we used PACE to train and tune for the hyperparameters. The best parameter we found were regularizer = 41800 and number of components = 584. Below are the results for the entire dataset.

Metrics

	Metric
Accuracy	0.825
Precision (Weighted)	0.821
Recall (Weighted)	0.825
F1-Score (Weighted)	0.755

 Introduction Contributions Table Data Pre-Processing Dbscan Random Forest SVM **Neural Network** Results Milestones

Aircraft Delay Prediction

CS7641 Project - Group 23

Method 4: Neural Network

Neural Network is a technique relying on 'neurons'. They consist of a linear combination of inputs fed to activation function which will transform, scale, and if needed add non-linearity to the output. These neurons are then connected to each other forming a neural network. Another popular technique used in conjunction with neural network is convolutional neural network (CNN). This is useful to simplify and highlight useful information from a large amount of features such as pictures. However, in this case the amount of features is limited, thus CNN is not used.

For this project, the architecture consists of six fully connected layers based on pytorch, as described below:

```
neural_classifier = nn.Sequential(  
    nn.Linear(entry_nb, nb_neurons),  
    nn.SiLU(SiLU_slope),  
  
    nn.Dropout(p=dropout_prob),  
    nn.Linear(nb_neurons, nb_neurons),  
    nn.SiLU(SiLU_slope),  
  
    nn.Dropout(p=dropout_prob),  
    nn.Linear(nb_neurons, nb_neurons),  
    nn.SiLU(SiLU_slope),  
  
    nn.Dropout(p=dropout_prob),  
    nn.Linear(nb_neurons, nb_neurons),  
    nn.SiLU(SiLU_slope),  
  
    nn.Dropout(p=dropout_prob),  
    nn.Linear(nb_neurons, halved_neurons),  
    nn.SiLU(SiLU_slope),  
  
    nn.Linear(halved_neurons, 4) # multiclass output  
)
```

This architecture uses batched descent to train six layers has been selected as a lower amount of layers results in an underfitting. They are both fully connected except for the second who has half of the number to reduce the amount of hyper parameters that needs to be swept through. For the activation function both ReLU and SiLU has been tested.

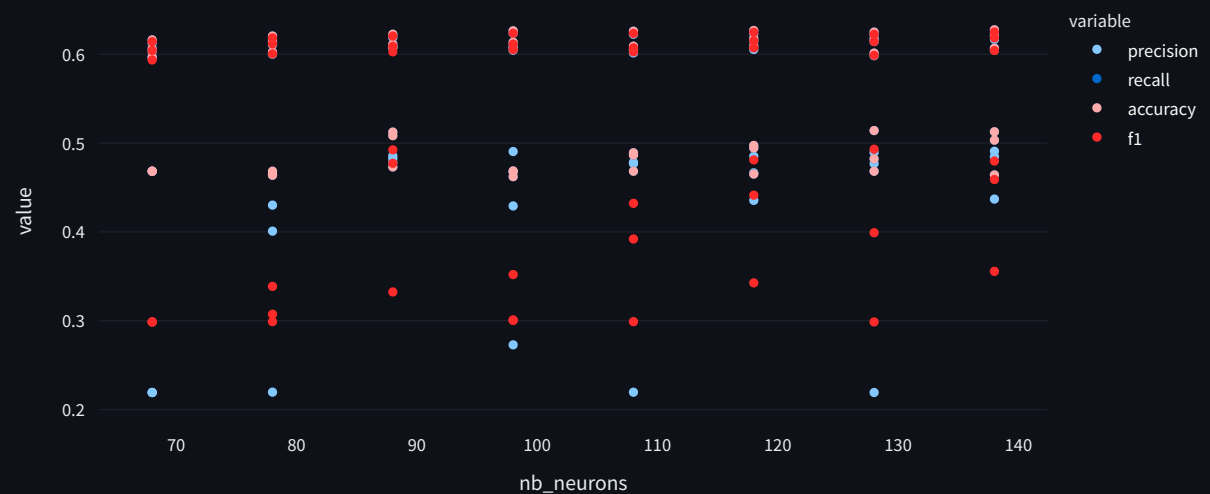
SiLU is selected due to no noticeable decrease in the training speed and some minor increase in the performance of the model. Dropout is also added to avoid overfitting between each layers. The last layers has four outputs for the four possible labels to classify: on-time, delayed, cancelled, diverted. Softmax requires to be added the output to normalize it such as its outputs sum results in one.

The function is not added as the loss function `nn.CrossEntropyLoss` already applies it its calculation. This loss function has been selected as it is a good starting function for multi classes classifying use cases. It is used in conjunction with the Adam optimizer as it demonstrates the best performance compared to Stochastic Gradient Descent (SGD). In this case there is a important imbalance between the classes of the data set so class weight has been added to the loss function. However, based on initial results it seems it is not enough. Thus the data set has been balanced so that the most two important classes (on-time, delayed) have the same amount of data points.

The hyper parameters swept through are the number of neurons in the layers, the learning rate, and the SiLU negative slope. Both learning rate and SiLU negative slope does not seem to have a noticeable impact on the performance of the model compared to the number of neurons as seen in the figures below:

Results of number of neurons sweep

NB Neurons against Performance

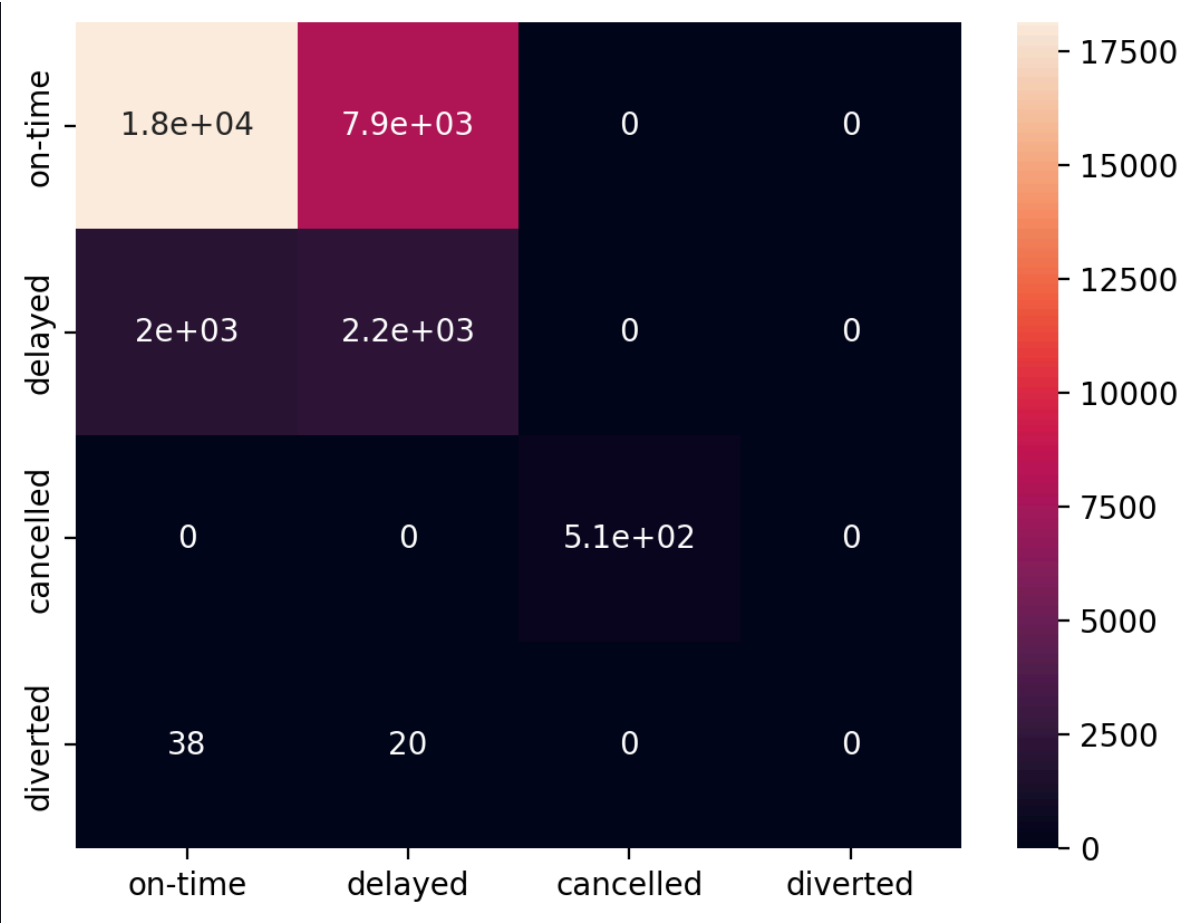


Learning rate against Performance



Using the learning rate 0.001, 138 neurons for our architecture and batches of 10 percent of the data size, we get the following confusion matrix and metrics:

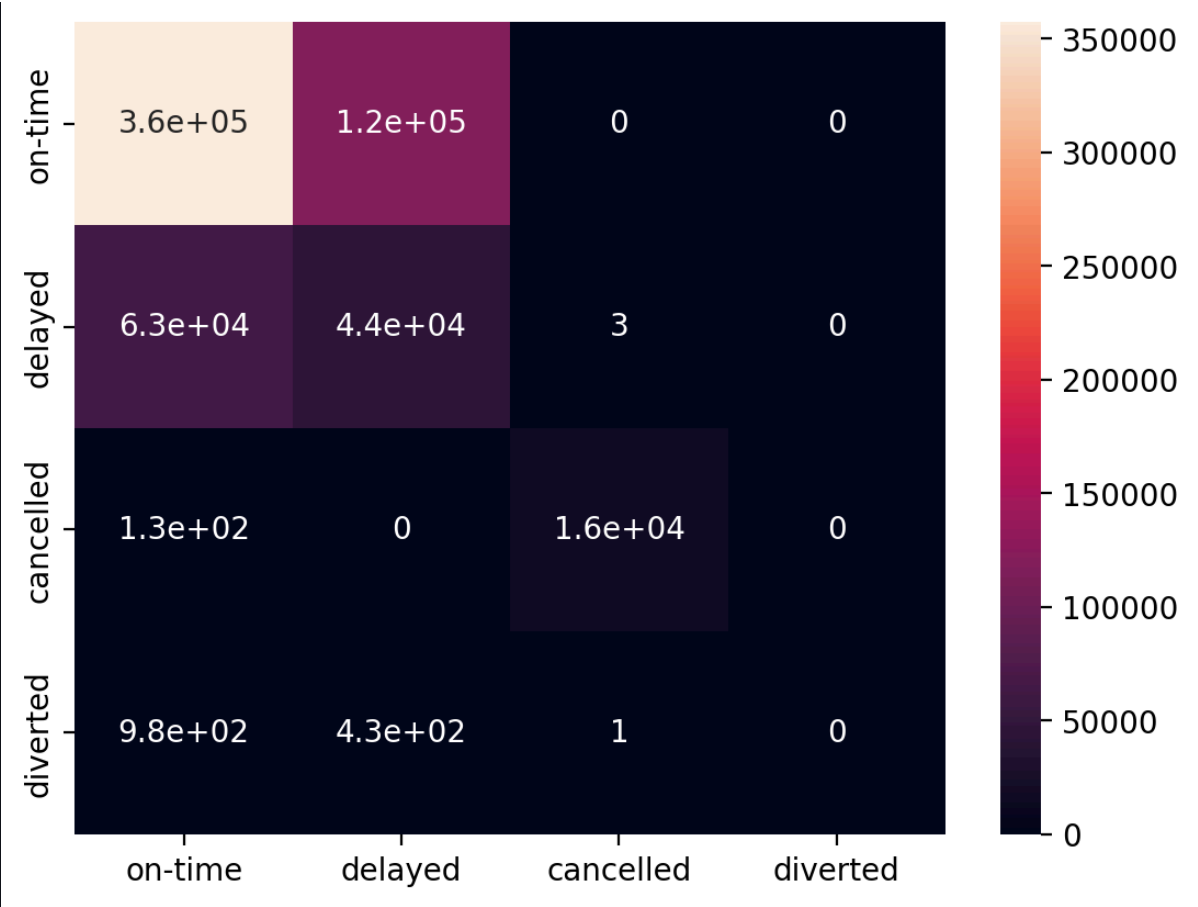
Confusion maxtrix



Metrics

	accuracy	precision	recall	f1
weighted	0.677	0.807	0.677	0.723

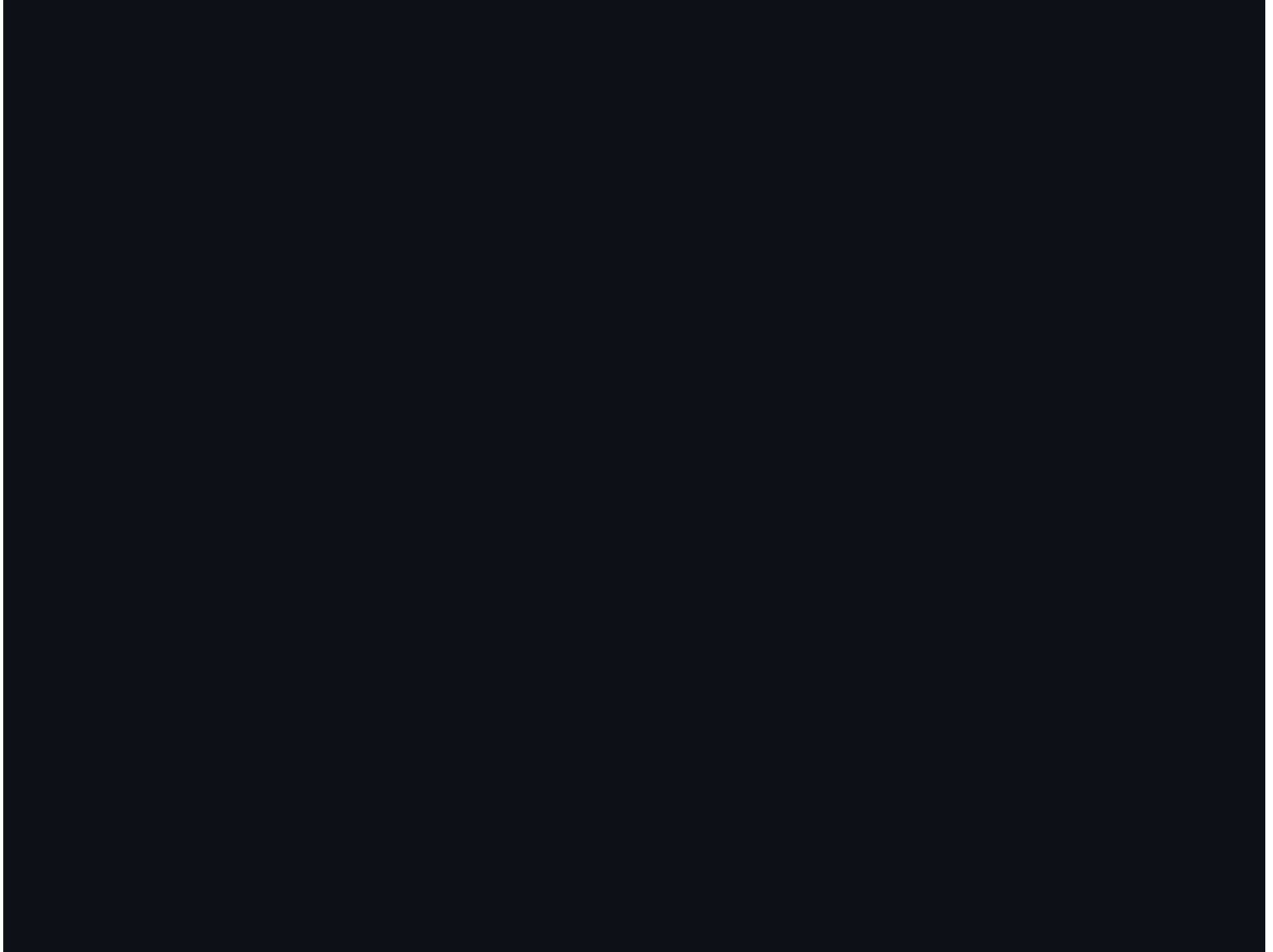
Confusion maxtrix









Metrics

	accuracy	precision	recall	f1
weighted	0.694	0.747	0.694	0.715

It seems that the issue lies in the network being unable to distinguish effectively against the on-time and delayed class. Several architectures with dozens more layer also have been tested to try and trigger overfitting. However, the model accuracy stagnates with the same value as the proposed architecture without any improvement instead of overfitting.



 Introduction Contributions Table Data Pre-Processing Dbscan Random Forest SVM Neural Network **Results** Milestones

Aircraft Delay Prediction

CS7641 Project - Group 23

Final Results

Our machine learning investigation into flight delay prediction revealed significant variations across four different approaches, with Support Vector Machines (SVM) emerging as the most effective method. The SVM model achieved the highest accuracy with 86.2% and a 0.801 weighted F1 score for the ATL data, which showed its ability to distinguish between multiple flight statuses including delayed, non-delayed, cancelled, and diverted flights through multilabel classification. By leveraging the Nystroem method and RBF kernel approximation, the SVM approach proved to be most effective in modeling the complex relationship between attributes that predict flight delay and the delay itself, outperforming other techniques by a considerable margin. This particular method gave room to experiment with different classifiers, highlighting flexibility as another strength.

Random Forest presented the second most promising approach, achieving a 70.1% accuracy and 0.73 F1 score. Its strength lied in constructing multiple decision trees that were able to capture the complex, nonlinear relationships between the flight attributes and delay outcomes and not being prone to overfitting. The model's precision of 0.81 indicates a strong reliability in positive delay predictions, meaning it is particularly good at predicting that a delay will indeed occur. Unlike SVM, it utilized a binary classification of delayed and not delayed, which may have played a part in the high positive delay prediction, but could be thought of as less specific. Overall, the method demonstrated a balanced performance across delayed and non-delayed flight classifications, highlighting its fit for this particular problem of predicting flight delays.

In contrast, the unsupervised DBSCAN clustering method proved least effective, and did not necessarily provide insight into whether a flight would be delayed. Although computationally efficient, its accuracy of 53.9% and F1 score of 0.467 revealed the fundamental limitations in using clustering techniques for flight delay prediction. The method struggled primarily because the flight delay data lacked natural, separable clusters that could be leveraged for meaningful classification, even after the data was made to be balanced. This approach showed how important it is to select appropriate machine learning techniques that align with the specific characteristics of the dataset (in this case, it required a method that accounted for lots of noise and complex relationships).

The neural network approach resulted in significant challenges, but still performed better than DBSCAN. Despite extensive experimentation with various architectures, learning rates, and preprocessing techniques, the model's performance remained stagnant. Between running it with the Atlanta dataset and the dataset with all airports, the performances remained similar, with accuracy of 67.7% and F1 score of 0.723 and accuracy of 69.4% and F1 score of 0.715 respectively. Like SVM and unlike Random Forest, it employed a multilabel classification. The neural network's inability to effectively separate similar classes highlighted the complex nature of flight delay prediction and the potential limitations of such learning approaches when confronted with nuanced, multiclass classification tasks.

These methods revealed several critical insights into machine learning applied to flight delay prediction. Supervised learning techniques significantly outperformed unsupervised methods, with SVM and Random Forest demonstrating the most promising results. One interesting thing to note is that, both Neural Networks and SVM seem to be good at predicting cancelled flights despite the overwhelmingly low amount of samples present in the dataset. The results emphasized the importance of sophisticated feature engineering, robust data preprocessing, and careful model selection. Class imbalance emerged as a consistent challenge across methods, suggesting that any extension of this work should focus on developing more advanced techniques to address this fundamental issue.

Next Steps

Potential next steps would be a multi-pronged approach to improve upon our current flight delay prediction models. This includes further feature engineering to identify more relevant predictive attributes, advanced data preprocessing techniques, and continued refinement of SVM and Random Forest models. Exploring ensemble methods that combine multiple approaches could potentially yield even more accurate predictions. These steps would be key to developing a highly reliable predictive model that can provide actionable insights for airlines, airports, and passengers in managing and anticipating flight delays, as we originally intended.

Aircraft Delay Prediction

📄 Introduction

📊 Contributions Table

🔍 Data Pre-Processing

🔗 Dbscan

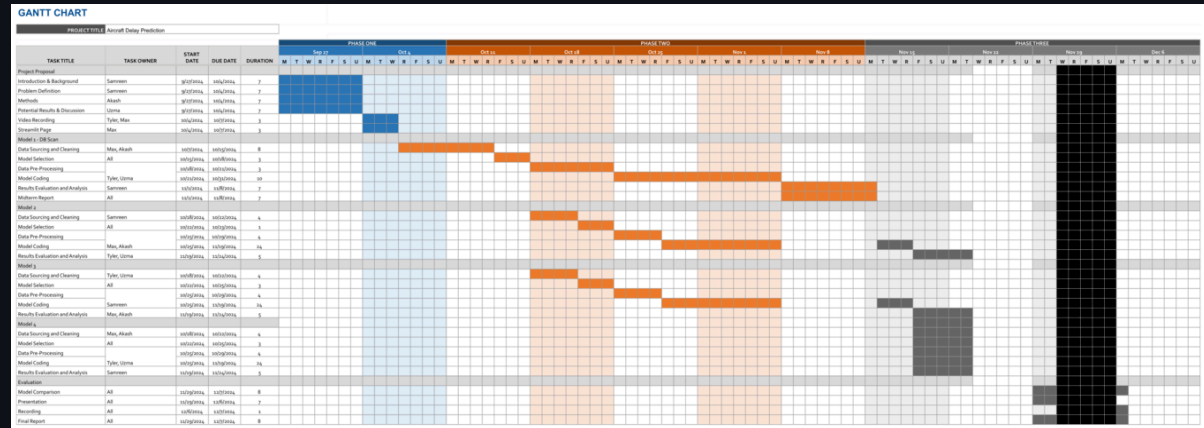
🌲 Random Forest

🔑 SVM

🧠 Neural Network

📈 Results

📅 Milestones



Gantt Chart

Download higher resolution image

Aircraft Delay Prediction

CS7641 Project - Group 23