Final Report

Introduction & Background

Literature Review
Flight delay is a critical factor in both traveler experience and airline operations. Previous studies have considered specific areas of the flight delay problem such as delay propagation and airline delay using various machine learning methods [1]. One study has shown that applying neural networks is a feasible approach for predicting whether a flight has been delayed [2]. Another study showed that neural networks overfitted delay prediction when using time-series data and did not perform as well as random-forest classifiers [3]. As such, our project will focus on the problem of predicting delays for single flights and use neural networks trained on important factors to prevent overfitting.

Dataset
We will merge two datasets, which both span 2018-2024 and detail each flight's scheduled and actual departure/arrival times, origin and destination locations, various delay types (in minutes), and other flight procedure statistics.

Dataset Links
Flight Delay, Flight Status Prediction

Problem Definition

Problem
Flight delays originate from multiple factors including weather, increased airport traffic, or technical failures. We will predict whether flights are delayed or not.

Motivation
Delays are inconvenient and potentially costly for both airlines and consumers. Understanding causes and predictors of delays can aid future mitigation and circumvention at both the individual and corporate level.

Methods

To preprocess our data, we used the following methods: data cleaning, data augmentation, and rescaling. To begin, we first converted all of our data in csv files, as some of the data was initially in a parquet file. We converted the data by reading the parquet file in chunks and writing it into a new csv.

We cleaned our data using functions from the pandas library, including dropna() which removes missing values and drop_duplicates() which removes duplicate data. At this stage, we also dropped data points and features that represented canceled or diverted flights, since we are not interested in studying these cases. We also narrowed our data points down to the 2019-2022 date range.

We augmented the clean datasets using merge() from the pandas library to broaden the set of features available for analysis. The pandas library allowed us to conduct an inner join of the data using common features. Some examples of common features include flight date, origin city, destination city, air time, etc. By evaluating these features together, we were able to identify which data points represented the same flights.

After augmenting the datasets, measurable (as opposed to categorical) data was rescaled to ensure features are not considered disproportionally, since features may have different relative scales. Rescaling

was especially important since one of the models we used was Principal Component Analysis (PCA), which requires distance calculations. We used StandardScaler from the scikit-learn library for rescaling.

Before we used the data for PCA, we also one-hot encoded the categorical features because PCA only takes numerical input. This was done with the get_dummies function in the pandas library.

The machine learning algorithms we are using for this project are PCA (unsupervised), Support Vector Machine (supervised), and neural networks (supervised).

First, we used the PCA algorithm to find the most significant features in our data. After data pre-processing and one-hot encoding, we had 794 features in our data. We used the fit method in scikit-learn's PCA class to fit the data and calculate the z-space. We plotted the features against their explained cumulative variance and used the elbow method to determine that we should take the 14 most significant features in z-space. We then transformed our data to only have those 14 most significant features using the transform method in scikit-learn's PCA class.

After performing PCA, we used SVM and the reduced feature set from PCA to create a discrete classification model to predict whether a flight will be delayed or not. SVM is a supervised learning algorithm that classifies data into categories using either an optimal line or a hyperplane, and we used scikit-learn's SVC class for binary classifications for SVM.

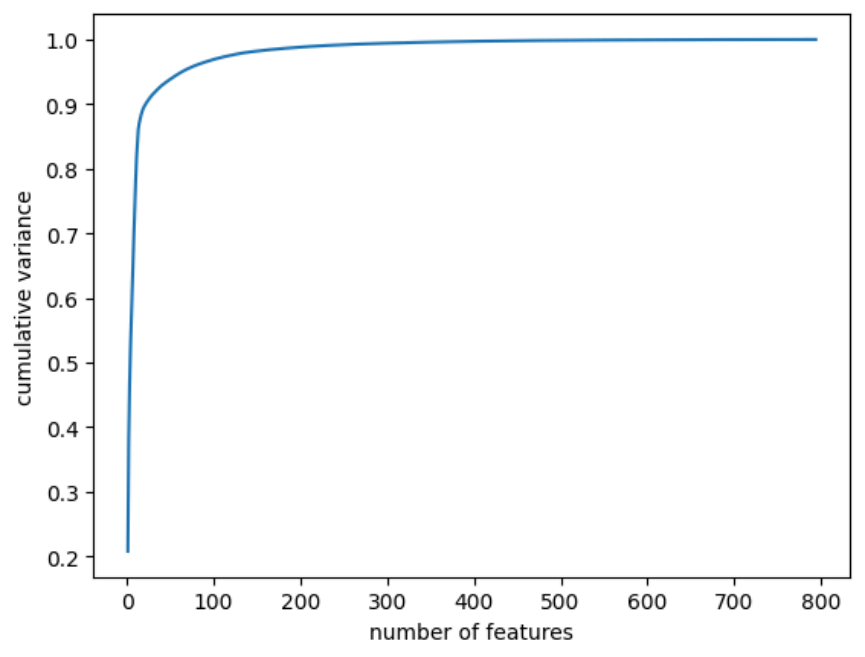
Due to the overwhelming size of the final dataset that we collected, we trained our SVM classifier on a small subset (1%, or roughly 200,000) datapoints in order to train it in a reasonable amount of time. However, we were still able to achieve reasonable metrics using this limited training dataset. When constructing the training dataset, we used an equal amount of training data from every year of flight delay data (2019 to 2022) in order to minimize bias.

For our third machine learning algorithm, we used NN and the reduced feature set from PCA to make another discrete classification model. This was accomplished using scikit-learn's train_test_split to split the data into training and testing sets. We used a 60/20/20 split for training, validation, and testing. To build and train the neural network, we used TensorFlow and Keras. The architecture of our neural network included 3 dense layers using ReLU as the activation function. Each layer contained 32 neurons. In addition to these three layers, we used one final layer that used softmax as an activation function.

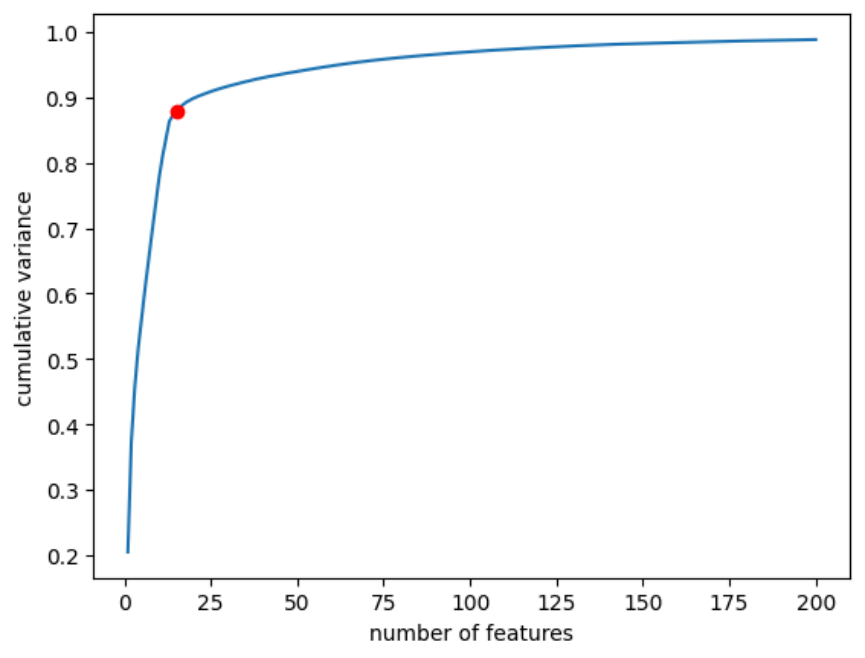
We also used accuracy_score, precision_score, recall_score, and confusion_matrix from scikit-learn for model evaluation metrics.

Results and Discussion

For PCA, we graphed the features from most significant to least significant in z-space and plotted their cumulative variance.



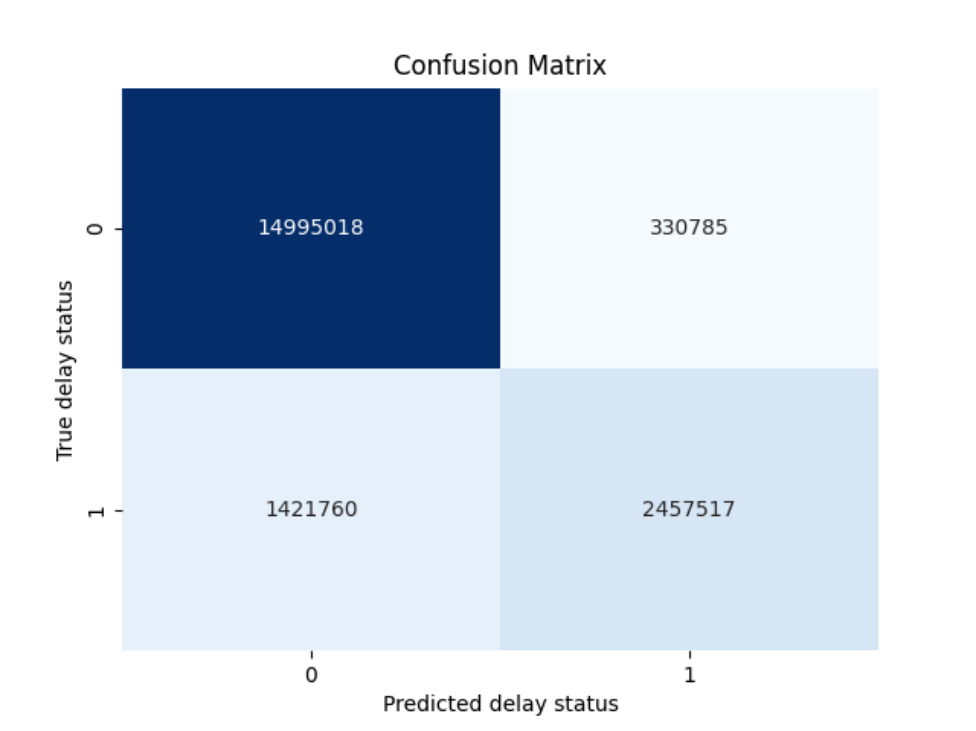
Zooming in on the most significant features for a better visualization, we used the elbow method to determine that we should keep 14 of the features in z-space to balance information retained and complexity of the data. The point on the graph below marks where we determined the “elbow” to be.



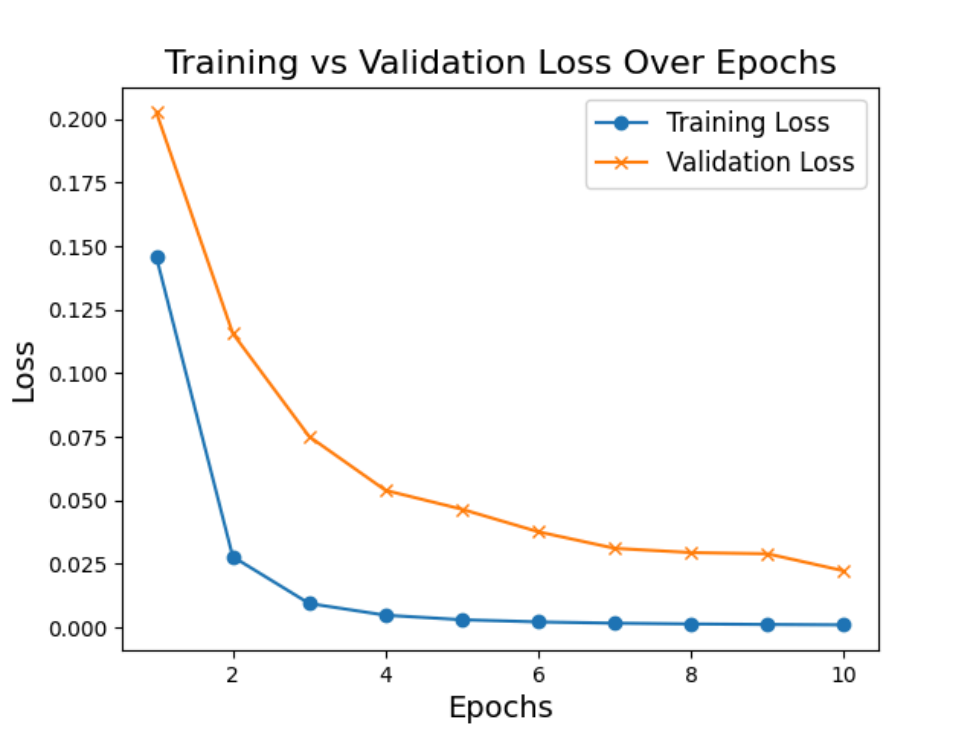
As a result, the 14 most significant features in z-space explained 88.43% of the variance in the dataset. The PCA model did pretty well at reducing the feature set size from 794 features down to 14 features and still being able to maintain a high explained variance.

After PCA, we ran other ML algorithms on data with this reduced feature set to reduce the complexity and speed up computation.

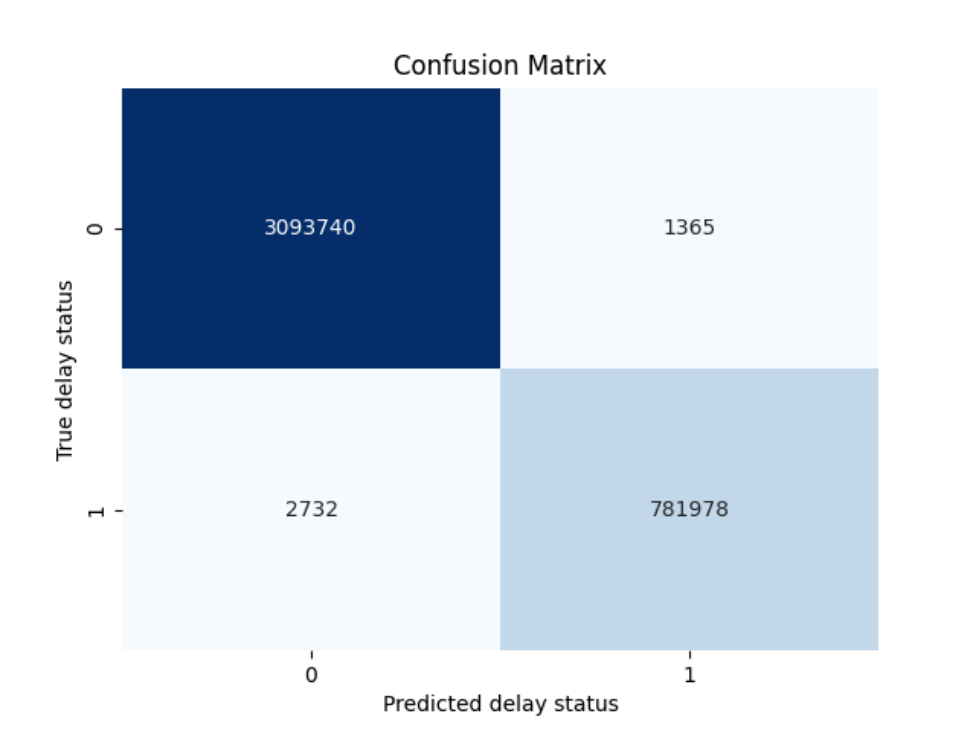
Our SVM classifier achieved an overall accuracy of 0.9087, with a precision of 0.8813 and a recall of 0.633, resulting in an overall F1 score of 0.737. We also generated a confusion matrix to visualize the predictions.



For our neural network, we created the graph below to show the training loss and validation loss after each epoch. Since training loss was approaching 0 after 10 epochs, we decided to train the neural net for 10 epochs.



Our NN achieved an overall accuracy of 0.9989, with a precision of 0.9983 and a recall of 0.9965, resulting in an overall F1 score of 0.9974. We also generated a confusion matrix to visualize the predictions.



Originally, we planned to use neural networks to do continuous prediction of flight delay. However, we decided to use NN to do discrete prediction instead in order to better compare its performance with our SVM classifier.

Since we needed numerical and not categorical data for SVM and PCA, after one-hot encoding our data, we ended up with 794 features. The strength of our first machine learning model, PCA, was that it was able to drastically reduce the amount of features in our dataset down to 14 features while still maintaining a high percentage of the variance in the data. This helped save training time for both SVM and NN since they didn't need to process data with so many features. A tradeoff with PCA is that we used the elbow method to determine how many features to reduce the dataset down to and ended up with an 88.43% explained variance. If we had tried to maintain a higher explained variance by keeping more features, we might have been able to achieve better results in our other models, particularly SVM.

The strength of our second machine learning model, SVM, was that SVMs are better at generalization compared to other classifiers. This was beneficial to our task because we had a large amount of varied data and in order to achieve high performance, we would need the model to be able to perform well even with unseen data. A tradeoff that we made for our SVM model is that SVMs generally take a long time to train on large datasets. In order to train the SVM in a reasonable amount of time, we reduced the training dataset to a random 1% of the total dataset. Thus, performance suffered as the model was not able to train on a larger amount of the dataset. Additionally, using nonlinear kernels for the model would have increased training time even more, though they may improve model performance.

The strength of our third machine learning model, neural networks, was that we were able to model nonlinear relationships easily using ReLU activation in the dense layers. This allows us to better capture relationships between the columns of our data, which could be a reason the performance of the neural network classifier was better compared to the SVM classifier. A tradeoff that we had with our neural network is that it is difficult to interpret the results and identify features that are most relevant to making the final prediction. Additionally, we spent a considerable amount of time tuning hyperparameters for the neural network.

In comparing our discrete classification models (SVM and NN), the neural net outperformed the SVM on all metrics (accuracy, precision, and recall). While accuracy was a key indicator of performance (and neural net outperformed SVM by 9.02 percentage points on this), because our data was pretty imbalanced and there were significantly more flights that were not delayed than delayed, we also needed to place

emphasis on precision and recall. For precision, the neural net outperformed SVM by 11.7 percentage points meaning that it predicted a lower proportion of false positives. Minimizing false positives could be important in not falsely alarming customers and maintaining airlines' reputations. For precision, neural net outperformed SVM by 25.95 percentage points meaning that it predicted a considerably lower proportion of false negatives as well. Minimizing false negatives could also be important by ensuring that relevant stakeholders are alerted about potential flight delays whenever they happen. Either way, minimizing both false positives and false negatives leads to a better prediction system for delayed flights and so overall, NN performed better.

For next steps, we noticed that there were significantly more flights that were not delayed compared to flights that were delayed in our data. We could attempt to fix this by balancing the distribution of delayed vs non delayed flights in our training data for future training. Additionally, while we achieved a pretty high accuracy, precision, and recall with the neural net, our SVM did not fare as well, with a pretty low recall. We could try to improve our SVM in the future by trying to use nonlinear SVMs, though training time would be negatively impacted.

By training accurate and precise models, we hope that airlines and travelers can be better informed about whether a flight will be delayed or not. This could have significant economic and environmental impacts by saving travelers time at the airport.

References

[1] L. Carvalho et al., "On the relevance of data science for flight delay research: a systematic review," Transport Reviews, vol. 41, no. 4, pp. 1–30, Dec. 2020, doi: https://doi.org/10.1080/01441647.2020.1861123.

[2] Y. J. Kim, S. Choi, S. Briceno, and D. Mavris, "A deep learning approach to flight delay prediction," IEEE Xplore, Sep. 01, 2016. https://ieeexplore.ieee.org/abstract/document/7778092

[3] G. Gui, F. Liu, J. Sun, J. Yang, Z. Zhou, and D. Zhao, "Flight Delay Prediction Based on Aviation Big Data and Machine Learning," IEEE Transactions on Vehicular Technology, vol. 69, no. 1, pp. 140–150, Jan. 2020, doi: https://doi.org/10.1109/tvt.2019.2954094.

Additional Deliverables

Gantt Chart: Spreadsheet

Member Contributions

	Member	Contribution
0	Amy	Methods brainstorm; Methods final edits; Literature review; Literature review slide; Title, intro, and methods slide recording; Combined and uploaded proposal video; Ran PCA on the datasets; Calculated PCA metrics; Created PCA visualizations; Wrote PCA sections in methods and results/discussion; Updated the GitHub README to explain all files and directories; Worked on results and discussion section for final report; Final presentation recordings for PCA and model comparison
1	Josh	Proposal outline; Literature review; Results and discussion; Editing; Batch-processed PCA data annotation script; Ran SVM on data; SVM metrics and visualization; SVM methods and discussion; Wrote neural network training, evaluation, and plotting code; Wrote neural network portions of report; Final presentation slides and recordings
2	Rose	Proposal outline; Draft for Results and Discussion; Results and Discussion slide recording; PACE documentation; Cleaned original datasets as part of data preprocessing; Edited methods section of report; Final video editing

	Member	Contribution
3	Sophie	Proposal outline; Research functions/libraries for models and data processing; Methods section and slide; Repository and Streamlit setup; Updated Streamlit with content; Streamlit, Datasets, and PACE documentation; Converted data to csv format; Augmented and rescaled datasets to be used for PCA; Summarized preprocessing methods for 'Methods' section of report; Initial SVM code; Final presentation slides; Recording for intro-data preprocessing slides; Final report Streamlit and upload