EMS740U - Machine Learning and Artificial Intelligence for Engineering

Case Study - Artificial Intelligence in Air Traffic Management (ATM)

Group M

Team Members
Ilham Sadeque
Ahmad Serhal
Aila Sheri
Youssef Medhat Youssef Sherif
Myra Siddiqi

Table of Contents

Introduction 2

Data Preparation 2

Linear Regression 4

Neural Networks 6

Adaptive Neuro-Fuzzy Inference Systems (ANFIS) 8

Comparison 9

Conclusions 10

References 10

Introduction

An important part of monitoring airport operations is estimating the time it will take an aircraft to taxi from the gate to the runway. It helps in better scheduling, prevents delays, and pinpoints locations where traffic tends to get held up. To investigate how machine learning can help in resolving this issue, this coursework focuses on Manchester International Airport (MAN), the second busiest airport in the United Kingdom.

The concerned data has up to 25 features, including taxi time, taxi distance, and hour at the gate. Using this data, the most important components impacting taxi times is identified and then, machine learning models are created to produce precise predictions. Additionally, three models will be examined: Linear Regression (LR), Neural Networks (NN), and Adaptive Neuro-Fuzzy Inference Systems (ANFIS).

The workload has been divided in the following way:

- Ilham Sadeque: no contribution to any sections.
- Ahmad Serhal: Neural Networks, ANFIS, and respective sections in the report.
- Aila Sheri: Neural Networks, Comparison, and respective sections in the report.
- Youssef Medhat Youssef Sherif: Data Preparation, PCA and feature selection, Linear Regression, ANFIS, Neural Networks, and respective sections in the report.
- Myra Siddiqi: Linear Regression, Introduction, Conclusion, and respective sections in the report.

Data Preparation

Data Processing

In this section, the process of cleaning and reducing the dimensionality of the dataset is discussed.

The data that is used for this study is from March 2021 and the data is in the form cleaned.csv. This aircraft movement data is then snapped to the airport map and produces a file in the form of snapped.csv. Finally, a feature extraction exe file is used to extract the 25 features of the flight information and stores it in the form of features.exe. A data set with 1695 data points and 25 features is collected after running the feature extraction.

Principal Component Analysis (PCA)

PCA is a technique in machine learning that is used to reduce the dimensions of a dataset while retaining as much variance (information) as possible. PCA transforms the data into variables called principal components which are linearly uncorrelated variables.

To apply PCA, the first step is to standardise the dataset using Z-score, so each variable has a mean of 0 and standard deviation of 1.

$$z = \frac{x - \mu}{s}$$

μ is the mean of each feature and s being the standard deviation.

After standardising the dataset, the covariance matrix is computed. Covariance provides information on the relationship between the features of the dataset. The covariance matrix essentially measures the variance of an independent feature and how features vary together.

The eigenvalues and eigenvectors of the covariance matrix are then calculated. The eigenvectors provide information on the direction of the data's variance and the eigenvalues show the amount of variance in each eigenvector. The eigenvectors and eigenvalues are sorted in descending order.

The number of principal components is then chosen according to how much variance is retained. For the case of this study and the collected data, a variance of 88.53% is achieved using 12 principal components. These steps are implemented using scikit-learn library in this study.

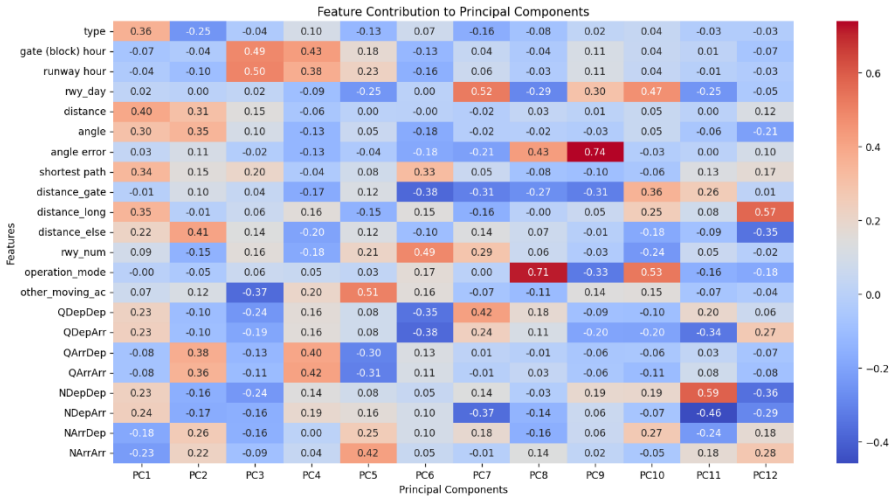

Figure 1: Feature Loading for Each Principal Component

In Figure 1, the contribution of each feature to each principal component is displayed. Principal component 1 (PC1) holds the most variance with 18.19% and the features that contribute most to it can be seen in the figure. The subsequent principal components such as PC2, PC3, and so on hold progressively less variance.

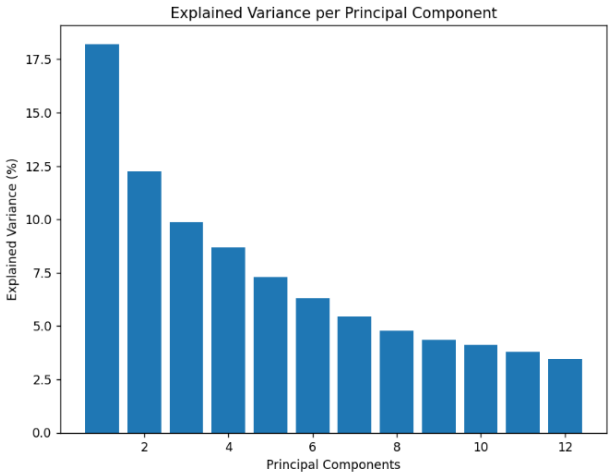

Figure 2: Variance per Principal Component

Feature Selection

Feature selection is crucial for machine learning models as reducing the number of features in a dataset ensures the models are trained on the most relevant features which improves accuracy and reduces overfitting.

PCA is used to select the most important features by selecting eigenvectors that correspond to the first m largest eigenvalues. The contribution of each feature is calculated by summing the absolute values of the eigenvectors across the top m selected eigenvalues. The contribution of each feature is ranked, and n features are chosen [1].

For the data set used in this project, the most important 15 features (largest (m=12) eigenvalues) are ranked as follows:

Feature	**Contribution Score**
QDepArr	2.4965657882579477
NDepArr	2.405597872931058
NDepDep	2.3999500666091547
distance_gate	2.348496286276258
operation_mode	2.2708196437305412
rwy_day	2.268658021853178
QDepDep	2.21106603792629
NArrDep	2.0397689565025754
angle error	2.027964845093177
distance_else	2.016969089081689
other_moving_ac	2.011235558066895
distance_long	1.9827915581781426
rwy_num	1.9648131750055478
QArrArr	1.7588705606888018
shortest path	1.7394935329211119

Prior to splitting the Data into Training and Testing sets, the data is filtered such that any instance of taxi time that is outside the 5^{th} and 95^{th} percentile is removed. This is done to ensure that the machine learning models are not influenced by any extreme values to maintain accuracy.

Linear Regression

Linear regression (LR), a fundamental machine learning model, makes predictions about a target variable by establishing a linear connection with one or more input features and a target variable. In this study, linear regression is applied to predict taxi times at Manchester international airport. The model is trained and tested on two datasets: 15 most significant features as discussed in the previous section, and the transformed PCA data with 12 principal components.

“scikit-learn” library is used to build the LR model in Python. The polynomial degree of the regression function is chosen to be 1. Using a second-degree polynomial function resulted in a better training model. However, the testing results were poor, which suggests that using a more complex model of 2 degrees overfits the data and fails to generalise to unseen data.

The model’s performance is evaluated using Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R^2) metrics.

MSE is used to measure the average squared difference between the predictions of the model and the actual values of the target.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_p)^2$$

Where n is the number of data points, y_i is the actual value and y_p is the predicted value.

RMSE is the square root of MSE, and it is used to measure the difference between the predicted values and the actual values. The unit of RMSE is the same as the target variable which would be minutes in this case. RMSE is used to check the accuracy of the model and the close RMSE value is to 0 the more accurate the model is.

MAE is used to find the average magnitude of error difference between the predicted value and the actual value. It has the same units as the target variable. MAE is robust against outliers unlike MSE and RMSE.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_P|$$

R-squared score (Coefficient of determination) explains how good of a fit the regression model is. The score is from 0 to 1. A score of 1 means the model perfectly explains all variations of the target variable. A score of 0 indicates that the model is not capturing any relationships in the data.

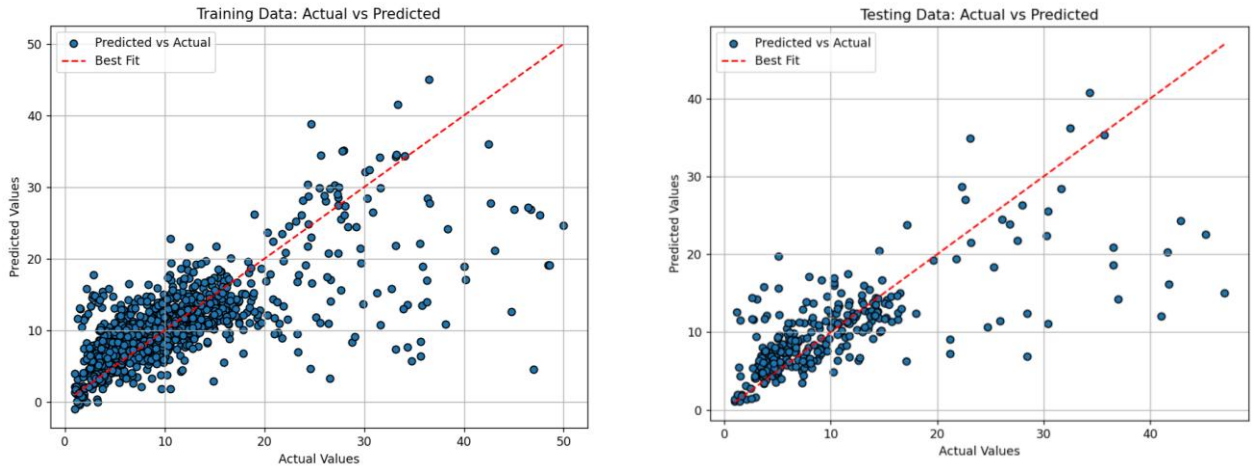

Figure 3: Actual vs Predicted Values for the Training and Testing Sets (15 Selected Features Dataset)

Model Performance Results for Linear Regression

Metric	PCA Dataset (Training)	PCA Dataset (Testing)	Top 15 features (Training)	Top 15 features (Testing)
MSE	27.3604	33.7434	29.1271	34.3824
RMSE (Minutes)	5.2307	5.8089	5.3970	5.8637
R-Squared	0.5578	0.5323	0.5292	0.5234
MAE (Minutes)	3.2036	3.4084	3.3155	3.5384

Regression function for the PCA dataset:

y = 11.5477 + (1.5757)*PC1 + (3.6029)*PC2 + (-1.4386)*PC3 + (4.4499)*PC4 + (-3.7601)*PC5 + (-1.9239)*PC6 + (-0.7386)*PC7 + (0.5224)*PC8 + (-1.2249)*PC9 + (-0.1834)*PC10 + (-.0796)*PC11 + (0.2071)*PC12

Regression function for the dataset with the 15 features:

y = 5.1899 + (1.7186)*QDepArr + (0.8133)*NDepArr + (-0.7338)*NDepDep + (0.0083)*distance_gate + (2.5094)*operation_mode + (-0.0188)*rwy_day + (3.0351)*QDepDep + (-0.2769)*NArrDep + (-6.2236)*angle error + (0.0036)*distance_else + (-0.5613)*other_moving_ac + (0.0055)*distance_long + (-5.1478)*rwy_num + (2.9389)*QArrArr + (0.0002)*shortest path

As seen from these results, The LR model with the PCA dataset performs slightly better than the dataset with the 15 selected features. However, using the LR model with physical features provides more insight on taxi time predictions compared to using principal components as features. For example, the coefficient for QDepArr is 1.7186. QDepArr refers to the number of other aircrafts that arrive at stand while the current aircraft is on the runway. The coefficient 1.7186 indicates that taxi time increases with higher values of QDepArr. In comparison, the coefficient for PC1 is 1.5757 which indicates larger values of PC1 cause higher taxi times. However, this lacks physical meaning and is not very useful when interpreting the model.

Determining the Significance of Features in the Model using T-tests

T-tests are conducted on the coefficients in the regression function with the 15 selected features to assess the significance of each feature in the model. The t-statistic is calculated by dividing the coefficient of the feature by its standard error. The p-value indicates whether the null hypothesis (that the coefficient is zero) can be rejected at a significance level of 0.05.

The t-tests and p-values are obtained using the model summary function from the "statsmodel" library.

Feature	Coefficient (Training)	p-value (Training)	Coefficient (Testing)	p-value (Testing)
QDepArr	1.7186	0.000	3.480	0.000
NDepArr	0.8133	0.007	0.6292	0.326
NDepDep	-0.7338	0.129	-1.3416	0.198
distance_gate	0.0083	0.043	-0.0102	0.238
operation_mode	2.5094	0.104	-7.0068	0.286
rwy_day	-0.0188	0.277	-0.0158	0.657
QDepDep	3.0351	0.000	2.7255	0.013
NArrDep	-0.2769	0.447	-1.0311	0.195
angle_error	-6.2236	0.265	-10.0953	0.121
distance_else	0.0036	0.000	0.0029	0.000
other_moving_ac	-0.5613	0.002	-0.4173	0.322
distance_long	0.0055	0.000	0.0044	0.000
rwy_num	-5.1478	0.000	-5.8724	0.000
QArrArr	2.9389	0.000	3.5698	0.003
shortest_path	0.0002	0.609	0.0019	0.059

From this table, it is evident that some coefficients have a p-value that is higher than 0.05 such as shortest_path which has a p-value higher than 0.05 across both the training and testing sets. This indicates that coefficients with a p-value of more than 0.05 may not have much of an effect when predicting taxi time. An area of improvement for this model is to evaluate the model after removing features that may not be statistically significant.

Neural Networks

A neural network (NN) model captures complex, nonlinear relationships between features and target variables. A backpropagation (BP) learning algorithm is chosen using "Pytorch" to perform the regression task of taxi time prediction.

The architecture of the neural network includes an input layer that processes PCA-reduced features and top 15 significant features as discussed before, with two hidden layers over 300 epochs, using the ReLU (Rectified Linear Unit) activation function that could assess the non-linear relationships in the data. Finally, a linear activation function is used in the output layer for taxi time prediction. The learning rate is set to 0.0001 with a batch size of 10 to achieve stable convergence.

The data is split into training and validation sets, with 80% comprising of the training set used, allowing it to learn the relationships between the taxi time and the input features. To test the model's predictive performance, 20% of the data is assessed for unbiased validation. The model's performance on the test set is evaluated and compared with the accuracy in the validation set. Backpropagation allows the model to shift errors backwards from the output layer to the input layers, aiming to minimize the loss function. Since Neural Network learns its internal parameters through a gradient descent algorithm, the data is scaled to ensure all features are equal.

Selecting the number of hidden layer nodes is another key part of using the NN model, and it required employing various configurations to find the balance needed between overfitting and adjusting to model dimensionality. Two hidden layers with 50 and 25 neurons respectively achieved the perfect balance and convergence level. This is a critical part of determining the model's ability to learn and generalize, with the number of neurons per layer being sensitive to memorizing the training data to generalize and learn patterns.

To monitor the model's performance, the evaluation metrics MAE, MSE, RMSE, and R-squared values are used.

Model Performance Results for Neural Networks

Metric	PCA dataset (Training)	PCA dataset (Testing)	Top 15 features (Training)	Top 15 features (Testing)
MSE	18.21	26.74	20.03	28.37
RMSE (Minutes)	4.27	5.17	4.48	5.94
R-Squared	0.71	0.63	0.68	0.61
MAE (Minutes)	2.66	3.04	2.64	3.07

Overall, the model's ability to achieve an R^2 value of 0.63 in testing PCA-reduced data validates its capability to be applied to real world taxi time predictions at Manchester International Airport. In contrast, when using the top 15 features data, the training R^2 was 0.68, which dropped to 0.61 on the test set. While this is a modest decline, this indicates that although the model can still explain over half of the variance in the training data with the top 15 features, its ability to generalize to unseen data is notably lower without the PCA reduced data. This was likely due to model becoming less effective in identifying and establishing relationships between relevant features within the testing data. While testing performance is lower than training performance, a reasonable level of generalization can be inferred from the results. However, it can be deduced from the higher MSE and RMSE for testing data that techniques such as adding regularization techniques, refining feature selection and fine tuning hyperparameters could improve generalization and model efficiency in the future.

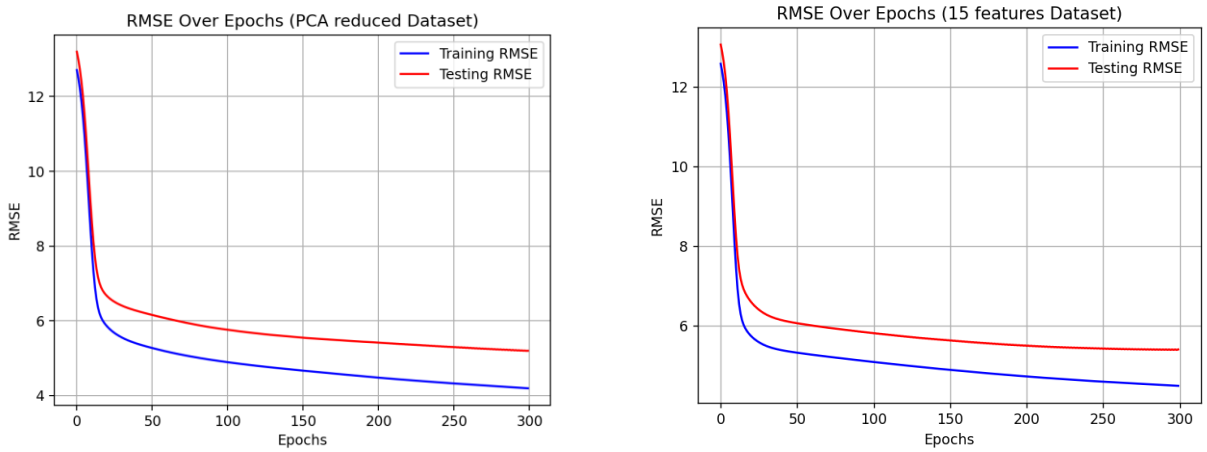

Figure 4. Training vs. Testing RMSE Plots for PCA-Reduced Data and 15 Features Data

From the graphs and the table of metrics, the PCA reduced data performs better than the dataset with the most significant 15 features. The testing RMSE is 5.17 for the PCA reduced data set which was smaller compared to the 5.94 RMSE for the top 15 features dataset. Moreover, there is a smaller gap between the training and testing metrics for the PCA reduced data compared to the dataset with 15 features which suggests that using the PCA reduced data improves the model's generalization ability which is likely due to reducing the variance of irrelevant features.

A one sample t test is used on the NN model on both datasets to statistically test the residuals (actual – predicted) of the testing datasets to show whether the mean difference between the actual and predicted values is different from zero. A p-value greater than the significance level of 0.05 indicates that the mean residual is not different from zero and the model is unbiased. The 'scipy.stats.ttest_1samp' library is used for the test to compute the p-value. A p-value of 0.8469 for the testing set of the PCA reduced data model and a p-value of 0.8773 for the testing set for the model with the top 15 features are obtained. Since $p > 0.05$, the mean residual is not significantly different than zero.

Adaptive Neuro-Fuzzy Inference Systems (ANFIS)

Fuzzy Logic Systems (FIS) are used to model systems that are complex to be modelled mathematically. This includes modelling systems with uncertainty and vagueness. As such, the rule structure of an FIS is inspired by human reasoning in inferring a result based on a fed input(s). These systems, however, have drawbacks. For instance, designing an FIS requires many parameters (such as fuzzy rules, fuzzy sets, membership functions…) that need to be chosen, and since FIS are untrainable, tuning such parameters is done manually which is a time-consuming procedure. On the other hand, neural network models are computation-based structures. They are trainable and work well when there is a large pair of input-output data for the intended model. However, they fail to handle uncertainties and vagueness in the target system. As such, ANFIS comes up to combine both systems to have an adaptive FIS with the ability to learn.

In this coursework, the Fuzzy Logic Toolbox of MATLAB is used to build, train, and evaluate the ANFIS model. To generate the initial FIS, subtractive clustering with a cluster influence range of 0.7 is used. In fact, many cluster influence range values were tested. It turns out that increasing it makes the training process faster and increases the testing R2 score but decreases the training one. For best balance between

the R2 scores, 0.7 was chosen. Using the initial FIS, the model can be trained and later evaluated to check the performance. For the training and testing, two types of data were used: PCA data with 12 principal components, and original data with features reduction to 15.

The following two figures show the plots comparing the predicted and actual testing outputs with the PCA and the 15 Features data respectively.

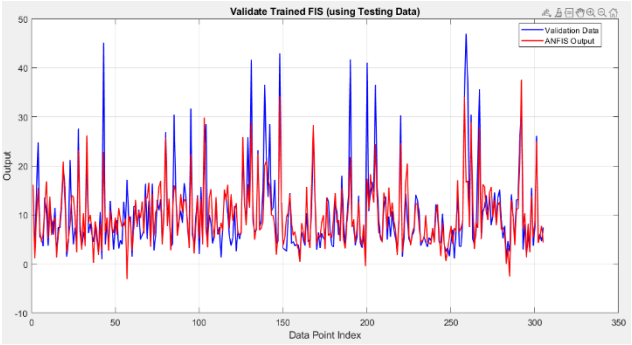

Figure 5 - Validation Plot for the FIS Trained on the PCA Data

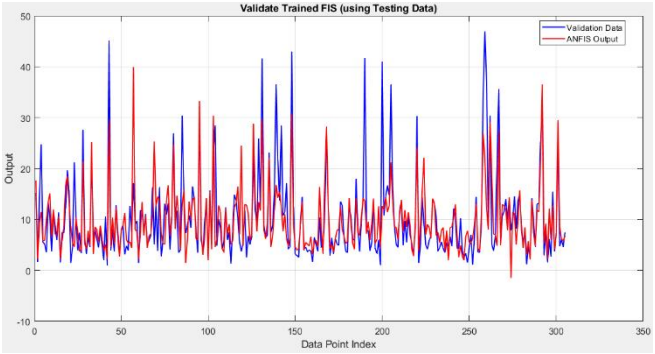

Figure 6 - Validation Plot for the FIS Trained on the 15 Features Data

From the above figures and the below table, it can be observed that ANFIS trained on the PCA data performs better than the one trained on the 15 features data. However, as mentioned previously, using PCA data replaces the direct physical interpretation of the results with respect to the original features.

Model Performance Results for ANFIS

Metric	PCA Dataset (Training)	PCA Dataset (Testing)	Top 15 features (Training)	Top 15 features (Testing)
MSE	21.9913	26.0503	22.2904	33.6721
RMSE (Minutes)	4.6895	5.1039	4.7213	5.8028
R-Squared	0.6445	0.6392	0.6397	0.5336
MAE (Minutes)	2.9892	3.0471	2.7747	3.2948

Comparison

In this study, three predictive modelling techniques—Neural Networks (NN), Linear Regression (LR), and Adaptive Neuro-Fuzzy Inference System (ANFIS)—were employed and assessed to predict taxi times at the Manchester International Airport using 15 top features and 12 PCA reduced components. This section analyses the best model performance by comparing the unique methodologies presented by each approach and their various advantages and drawbacks.

The results for this study showed that the models' performances are data dependent. With regards to the PCA-reduced data, NN and ANFIS outperformed LR when comparing their RMSE and R^2 values. Overall, NN was able to achieve the lowest RMSE value followed by ANFIS and LR. NN was able to perform well showing strong predictive capability as well. For non-linear and elaborate models, PCA-reduced data worked best to enhance performance overall. For this reason, a model such as LR that lacks the flexibility to model complex non-linear relationships was not able to perform the best. When using the 15 features data, NN and ANFIS reduced in their generalization capability overall with higher RMSE and

lower R^2 values, exhibiting higher error. This is likely because models such as NN and ANFIS can learn noise or develop spurious relationships during the training phase if the data isn't reduced or transformed as seen in the PCA data, and this lowers their ability to generalize to new sets of data.

In general, LR being a quick baseline model was the most interpretable and transparent, as each feature in relation to taxi time could be understood directly. While NN and ANFIS provide better predictive performance, it came at the cost of lowering their transparency. For this reason, they are often called 'black box models'[2]. In ANFIS, the fuzzy rules and membership functions can be examined, helping it serve as a better model than NN for transparency, but a worse model than LR due to its increased complexity.

Comparison Results for all Three Models

Model	Dataset	Testing RMSE (minutes)	Testing R^2	Transparency
LR	PCA	5.81	0.53	High
LR	Top 15 features	5.87	0.52	High
NN	PCA	5.17	0.63	Low
NN	Top 15 features	5.94	0.61	Low
ANFIS	PCA	5.10	0.64	Medium
ANFIS	Top 15 features	5.80	0.53	Medium

Conclusions

The results in this report showed that the best model for forecasting taxi times at MAN Airport was the Neural Networks (NN) on the PCA dataset with a testing R2 of 0.63 and the RMSE of 5.17 minutes. This is because it showed how well it generalises to new data and captures intricate, non-linear relationships. NN performed slightly less well on the top 15 features dataset (with an R2 of 0.61 and an RMSE of 5.94 minutes), but it was still more accurate than the other models. Linear Regression (LR) showed a clear understanding of how various characteristics affected taxi times and therefore, it was the most easy and comprehensible model. However, its predictions were not very accurate because it was unable to use the non-linear data. The Adaptive Neuro-Fuzzy Inference System (ANFIS) performed well on the PCA dataset with an RMSE of 5.10 minute, making it a good model for predicting taxi times as it is easier to interpret than NN.

To make a model accurate for real life operations at an actual airport, there needs to be further research on models that can both be interpretable and accurate. Furthermore, for very unpredictable situations that constantly change, the models need to be able to use adaptive learning strategies which would improve the model performance. These future developments would help air traffic control and minimise delays.

References

[1] F. Song, Z. Guo and D. Mei, "Feature Selection Using Principal Component Analysis," 2010 International Conference on System Science, Engineering Design and Manufacturing Informatization, Yichang, China, 2010, pp. 27-30, doi: 10.1109/ICSEM.2010.14.

[2] Lipton, Z. C. (2018). The mythos of model interpretability. Queue, 16(3), 31–57. doi:10.1145/3236386.3241340