Group ID - MSc in Data Analytics

Author: Olena Pleshan
e-mail: sbs24043@student.cct.ie
Student ID: sbs24043
GitHub Link: https://github.com/CCT-Dublin/ca1-sbs24043.git

List of Figures:

Abstract
The study explores tourism trends in Ireland using two datasets: air and sea travel data from 2010 to 2023, and air passenger transport data. Objectives include identifying primary travel routes, analyzing route fluctuations, assessing flight occupancy rates and validating whether international arrival/departure airports can be predicted using factors such as year, month, number of flights, total capacity and number of passengers. Descriptive and exploratory analyses were conducted, alongside functional programming for code efficiency. Machine learning models such as Support Vector Machines, Decision Trees, and Ensemble methods were used for predictions. Results show a growing trend in tourism, with insights into flight occupancy and international airport predictability. Ensemble models demonstrated the best performance without data preprocessing.

Introduction
This study attempts to answer several questions regarding tourism in Ireland, leveraging two datasets:

1. Air and Sea Travel from 2010 to 2023, sourced from the Irish Central Statistics Office. This dataset offers a comprehensive, top-level overview. Source: Irish Central Statistics Office https://data.cso.ie/table/ASA02
2. Air passenger transport between Ireland's primary airports and their main partner airports (routes data), obtained from Eurostat https://ec.europa.eu/eurostat/databrowser/view/avia_par_ie/default/table This dataset furnishes data encompassing passenger numbers, flight capacity, and flight frequency between all six Irish airports and airports worldwide, with a granularity extending to monthly intervals. Source: Eurostat

By analysing these datasets, the report aims to achieve the following objectives:

1. Determine the primary travel route into Ireland for tourism purposes.
2. Determine the route experiencing the most significant percentage increase and decrease since 2010.
3. Ascertain the most lucrative months for operating flights on the primary travel route (using Air Passenger Transport dataset)
4. Assess occupancy rates for the primary travel route.

Additionally, two hypotheses are proposed for investigation:

Hypothesis 1: Is there variability in flight occupancy among Irish airports and it depends on the route and month of travel?
Hypothesis 2: Is it possible to identify the domestic airport based on factors including airport pairs, passenger numbers, flight frequencies, total flight capacity, year, and month?

Materials and Methods
Descriptive analysis was used for ordering, manipulating, and interpreting raw data from various sources. Data visualization techniques were used to provide insights into the datasets.

Exploratory and statistical analysis was used, as the dataset itself does not contain any notion of the relationships between the data and different variables. It helped to find connections and generate graphs to identify suitable methods for data preprocessing, and further hypotheses validation.

Feature Engineering was used as a technique of formulating the most appropriate features given the data, the model and the task (Zheng, 2018).

Functional programming was used to create repeated code which could be used across the analysis notebooks repeatedly. The functions aim to be robust and take inputs and help avoid repeated code, ensure scalability, traceability and predictability. This afforded the ability to rerun the same analysis for the same dataset but with different encoders, scalers and different ML models. Python libraries were used to create graphs and understand how the data can further be explored to answer the questions posed.

Python programming language was used for this project, because it has a vast ecosystem of libraries specifically designed for data analytics, such as NumPy, pandas, SciPy, matplotlib, seaborn, and scikit-learn. These libraries provide tools for data manipulation, analysis, visualization, and machine learning, and have a large support community through StackOverflow.com and other resources.

Support Vector Machines with different kernels, Decision Trees, Ensemble and Boosting models were used and compared. GridSearchCV was used to identify the best combination of hyperparameters for a given model, and reports were built to provide insights into confusion matrices and classification reports.

Finally, KDD (Knowledge Discovery in Databases) was used as a paradigm for extracting useful knowledge from large volumes of data. KDD encompasses various stages such as data preprocessing, data mining, interpretation, and evaluation to uncover patterns, trends, and relationships within the data, which was done throughout this study (Fayyad, 1996).

Results / Discussion

The number of tourists arriving into Ireland has exhibited a growing trend from 2010 till 2023, with the majority of tourists coming from Continental Europe and Cross Channel (UK) routes. Since 2014, travelling from Continental Europe has surpassed Cross Channel Tourism. In 2023 tourism to Ireland surpassed the pre-pandemic levels of the year 2019. These results are demonstrated in Figure 1. With cumulative records and events charts ranking high on Tufte's empirical measure of graphical performance (Iversen, 1988), this study will aim to provide visualizations that conform to this principle and provide data-rich time-series graphs.

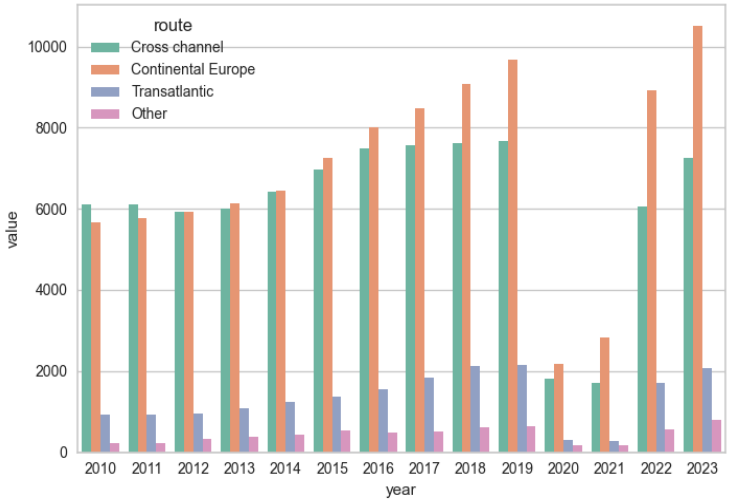

Figure 1. Route of Travel for tourists into Ireland

The rate of change in the number of tourists arriving into Ireland is showing a yearly increase (Figure 2), with Transatlantic and Other (non-European) routes of travel growing faster than Continental Europe and Cross Channel. Post-pandemic recovery highly screws the data, hence the graph below applies Tufte's principle of small multiples (Bell, 2019), emphasizing differences by repeating a graph with pre-pandemic and post-pandemic graphs in separate frames.

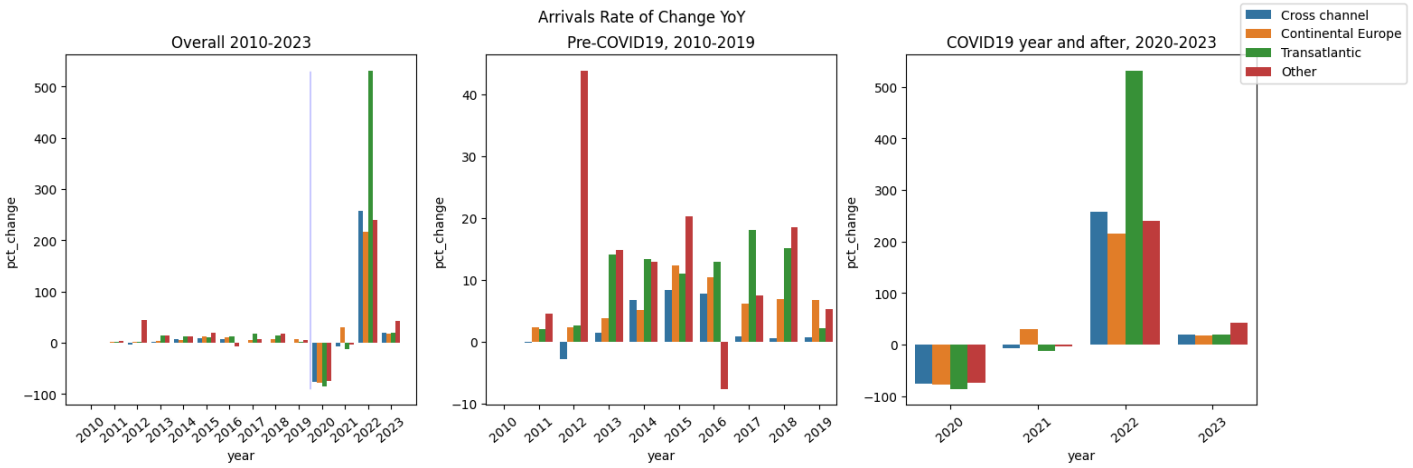

Figure 2. Rate of change of arrivals for tourism into Ireland Year over Year

The second dataset was transformed to analyze passenger arrivals, capacity, and flight numbers at Irish airports, focusing on European destinations. Results are detailed in the Statistics and Machine Learning Sections. Profitability scenarios were investigated using occupancy rates, revealing probabilities of flights being profitable or incurring losses. This analysis was preceded by extensive data preparation.

Data Preparation: Feature Engineering

The second dataset contains monthly data on passenger arrivals at six Irish airports from global sources. Metrics include seat availability, passengers, and flight numbers, each as a separate row. The objective was to consolidate insights on passenger arrivals,

capacity, and flights for all airports, particularly focusing on European destinations. Feature engineering involved transforming the dataset by separating Transport and measurement categories into distinct columns, splitting airport pairs, and adding a continent-based lookup. After merging three datasets, 40 empty rows out of 47,411 (0.08%) were handled with SimpleImputer. The final dataset was filtered to include only European international airports and the top 3 Irish airports.

Statistics

Statistical information about the dataset is outlined in the table below:

index	year	month	num_passangers	total_capacity	num_flights	pct_occupancy
count	47376.0	47376.0	47376.0	47376.0	47376.0	47376.0
mean	2013.0183637284701	6.503123944613306	8759.915062478893	11506.86790780142	70.30091185410335	0.767134688449848
std	5.853811015022934	3.386602620917335	10740.775633929368	14021.9381519355	83.65780105320536	0.1503205287112916
min	2003.0	1.0	0.0	6.0	1.0	0.0
25%	2008.0	4.0	2834.75	3828.0	23.0	0.689
50%	2013.0	7.0	5235.0	6756.0	42.0	0.787
75%	2018.0	9.0	10099.25	13134.0	83.0	0.878
max	2023.0	12.0	100056.0	134374.0	651.0	1.066

Table 1. Statistical information about the transformed Air passenger transport dataset

To identify the distribution of our main numeric columns, “fitter” python library was used to try to fit different distributions to the data, and it was identified that the data follows lognormal distribution, as shown in Figure 3, and number of passengers was found to be exponentially / cauchy distributed, as shown in Figure 4.

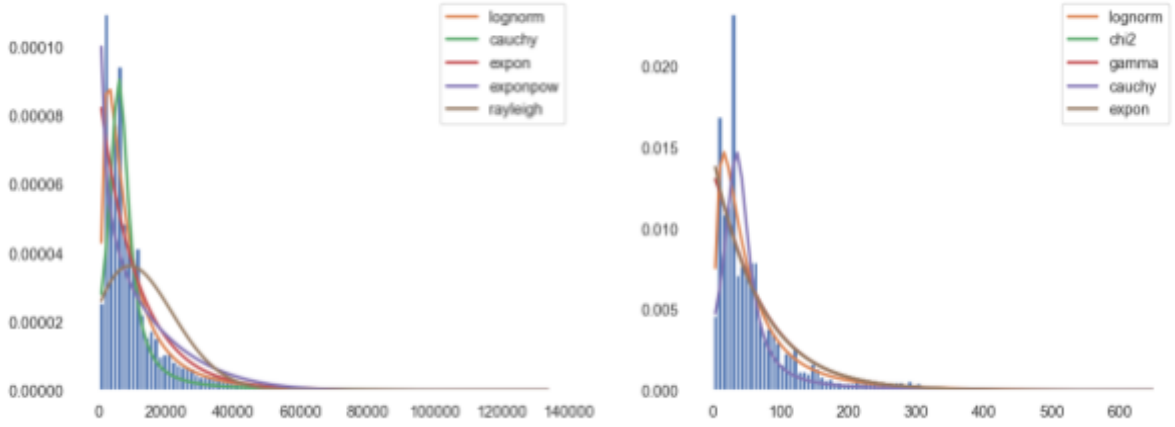

Figure 3. Visualizing how data can be fitted to distributions. Total capacity is presented to the left and number of flights to the right

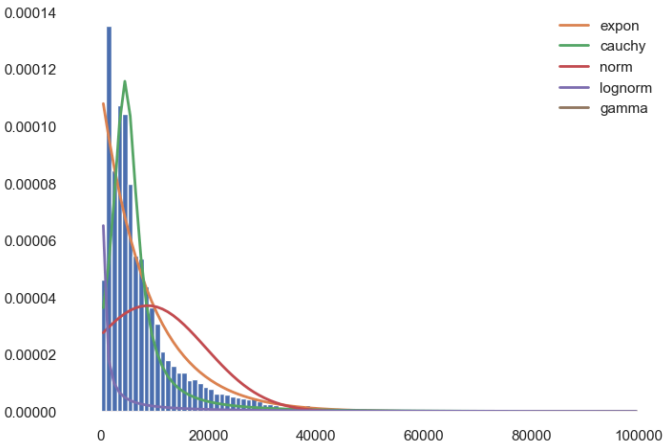

Figure 4. Visualizing number of passengers fitted to different distributions

According to Crow 1988, a log-normal (or lognormal) distribution is a continuous probability distribution of a random variable whose logarithm is normally distributed. Although the normal distribution is continuous, it is often used to approximate discrete distributions. As shown in the graph, the log-normal distribution is a right skewed continuous probability distribution, meaning it has a long tail towards the right. It is used for modelling in diverse areas, ranging from business to oceanography (Kundu, 2017).

Therefore to better comprehend the data distribution, it is worth doing a log transformation. The log transformation is often used to reduce skewness of a measurement variable (West, 2021). The log transform is shown in the graphs below, and visualizes how lognorm is related to normal distribution:

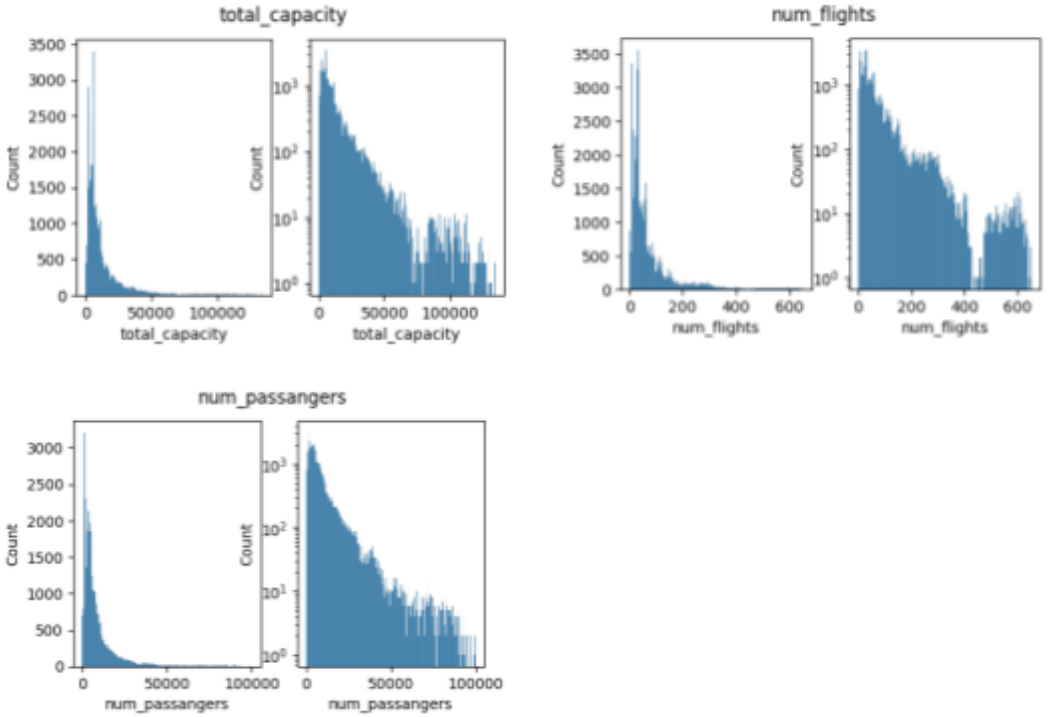

Figure 5. Variable distributions to the left with their Log Transform graphs to the right

Distributions provide the basis for forecasting future outcomes and building predictive models (James, 2013). Hence, understanding data distribution will help with data preprocessing (for example, imputing), as well as ML model evaluation.

Before diving into further the analysis, occupancy rates were calculated by dividing the number of passengers arriving from each airport into Ireland per each month by the total capacity of the flights, and added as a separate column on the dataframe. The graph below (Figure 6) shows that occupancy rates across different airports follow normal distribution:

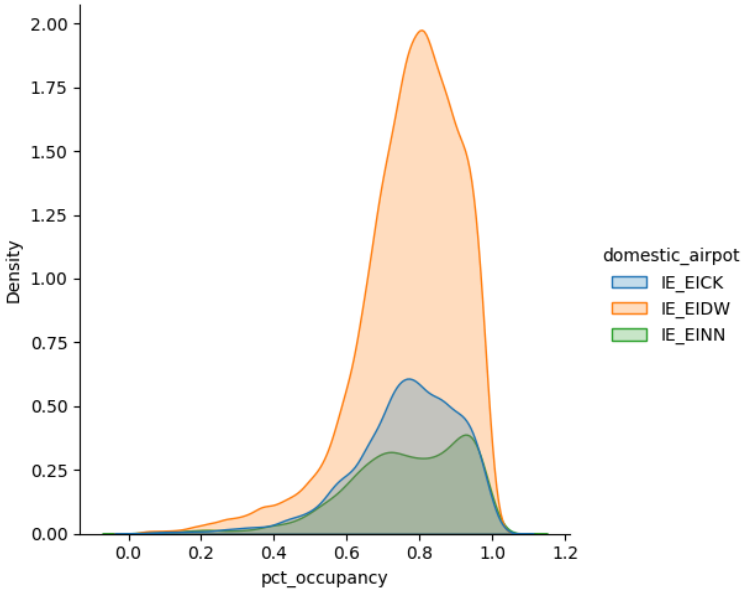

Figure 6. Occupancy rate displot

Additionally, monthly occupancy rates were plotted per each airport in **Figure 7**. The boxplot shows outliers on the lower end, and show that some months occupancy rates are normally distributed, while there is skewness for low and high seasons

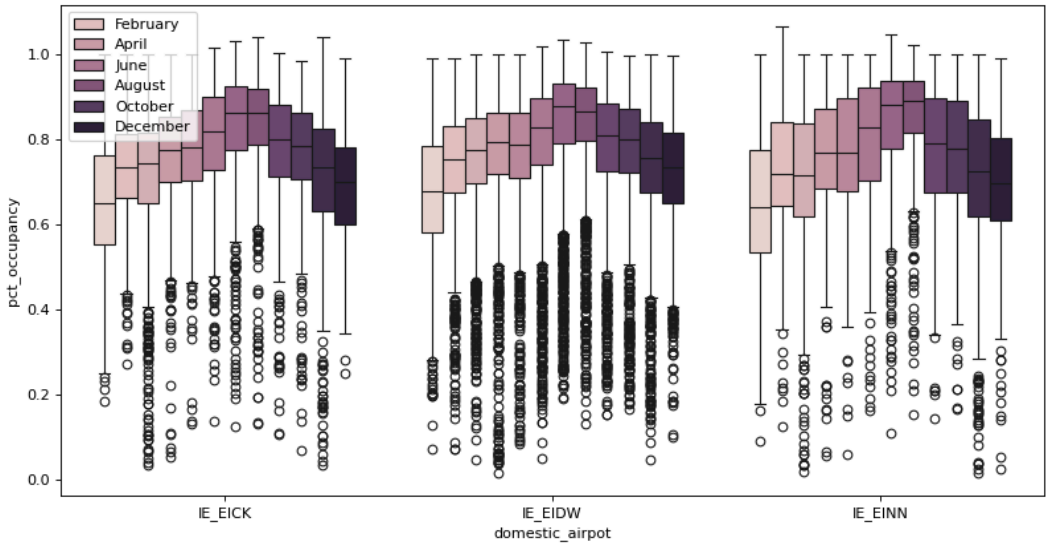

Figure 7. Occupancy rates by month for top 3 Irish airports

In a hypothetical scenario, with a 75% occupancy rate, profitability for each destination, airport, and month can be calculated. Using a binomial distribution, probabilities of flights being profitable or operating at a loss was determined. For flights to/from Ireland, the probability of profitability over a year is 0.309353, and for losses, it's 0.398029. Seasonal analysis showed profitability probabilities of 0.690647 in January (low season) and 0.804925 in July. This is summarized in Table 1.

time	is_profitable	probabilities
year	1	0.601971
year	0	0.398029
january	0	0.690647
january	1	0.309353
july	1	0.804925
july	0	0.195075

Table 1. Profitability probabilities with a 75% occupancy rate margin

Hence, for 100 flights arriving at the Dublin / Shannon / Cork airport, below are the results for Cumulative Distribution Function (CDF), Probability Mass Function (PMF), and Survival Function (SF) are:

	Year Avg	January	July
Probability of up to 30 of those 100 flights are operating at a loss	0.0277	0.0	0.9959
Probability exactly up to 30 of those 100 flights are operating at a loss	0.011	0.0	0.0038
Probability of over 30 of those 100 flights are operating at a loss	0.9723	1.0	0.0041

Table 2. Summary for binomial distribution scenarios

Another case of a binomial distribution in this dataset is whether or not a person is arriving for a particular country.

country_code	probabilities
United Kingdom of Great Britain & Northern Ireland	0.3245660289806163
Spain	0.18615933011326485
Poland	0.07143911750437662
Germany	0.06289679610217039
France	0.06264369028284575
Italy	0.056105123283626164
Portugal	0.03480205015713653
Netherlands	0.028474404674020796
Ireland	0.024677817384151358
Belgium	0.022104574887684292

Therefore, Among 1000 random traveller going through the passport control on any given day, the at least 300 of them are coming from a flight from United Kingdom of Great Britain & Northern Ireland is 0.051312 and the probability that at least 100 of them are coming from a flight from Poland 0.999654.

These features are further used by machine learning models to validate hypothesis 2, and the analysis above helps to understand how the scale and distribution of the data can impact the outcomes of the model.

Finally, figure 8 addressed one of the goals of this analysis, which is to identify the top 5 countries with heaviest flight traffic between Ireland and Europe.

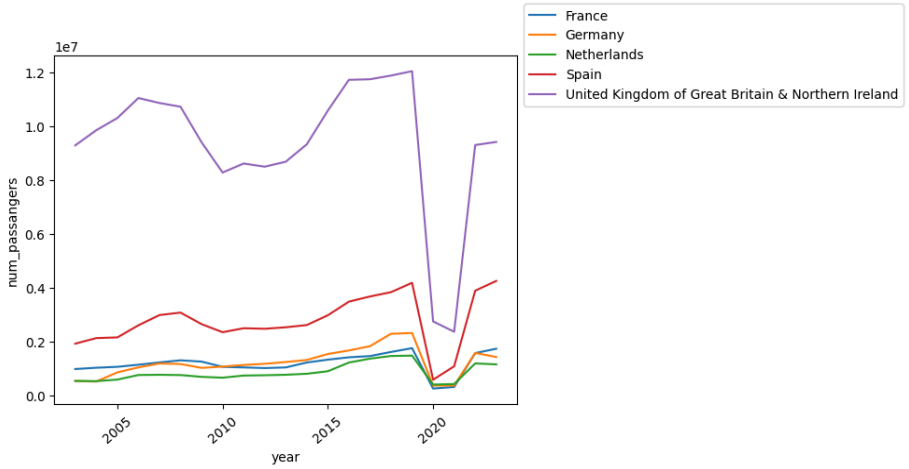

Figure 8. Top 5 countries by passenger traffic between Ireland and Europe

Data Preparation: Preprocessing for Machine Learning

The dataset contains categorical, as well as numerical values. According to Scikit learn documentation, depending on the machine learning model, the data may need to be processed differently. For example, tree based models require little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Some tree and algorithm combinations implement algorithms to impute missing values. (scikit-learn. (n.d.))

Considering that this analysis will need to repeatedly try different algorithms or different / same data, utility functions have been developed. For data preparation purposes, one of such functions visualizes the impact of different scalers from the scikit-learn library on the data distribution. By graphing the various scalers together, it provides insight into how each scaler transforms the data.

Scaling

Feature scaling is one of the important transformations that need to be applied, since machine learning algorithms do not perform well when the numerical features have very different scales (Géron, 2019). Various algorithms are available for data scaling. The jointplot below showcases the distribution of passenger numbers and flight counts, illustrating how different scalers can affect both the shape and actual values of the data.

In the figure below, it's clear that most scalers maintain the shape of the distribution, whereas quantile transformers and L2 normalization do change it. QuantileTransformer, for instance, applies non-linear transformations that compress the distances between marginal outliers and inliers (scikit-learn. (n.d.)) In some instances during this study the model could not be built or the build was timing out due to skipped data scaling step.

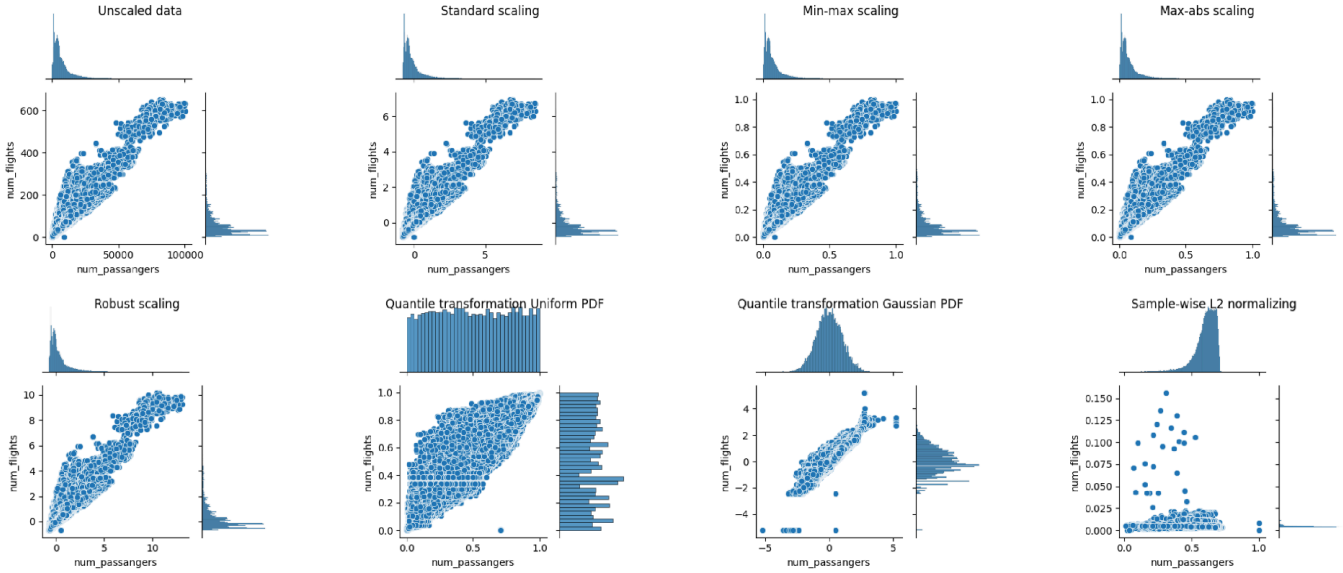

Figure 9. Visualizing effect of different scalers on data distribution

Encoding

In the utility function, Ordinal Encoder was applied as a default encoding method for most of the initial testing to encode categorical labels such as direction and airport. In ordinal encoding, an integer is assigned to each category, provided the number of existing categories are known, and it implies an order to the variable that may not actually exist (Potdar, 2017).

Although there is no natural order to the categorical variables in the Air Passenger Transport dataset, a comparison between outputs generated using ordinal and one-hot encoding for exploratory purposes was done.

Therefore, in addition to Ordinal Encoder, OneHot encoding was attempted to prevent assigning weights to international airports. Due to computational limitations, sampling was necessary for OneHot encoding. As a result, a sample of 6000 rows from the initial dataset was selected, leading to a rather sparse dataset with 7681 features derived from the 6 input columns.

Label Encoder was employed to encode the categories corresponding to domestic airports. The label names were stored in an array and used while graphing the decision matrix during each run, as well as in classification reports.

Data Splitting

Splitting the data into separate train and test datasets allows us to first fit the model to the training data, and then use the test dataset to evaluate generalization performance (Deisenroth, 2020). In the `split_encode_scale` the choice was made to first split the data, do encoding and scaling, and provide optional inputs to the function which will enable to change the data split. The default split ratio between train and test data is 80% vs 20% respectively, however, the utility function allows for adjustments, and a 70/30 split is used on the model where overfitting is suspected.

Machine Learning

Validating the hypothesis of whether it is possible to predict Irish domestic airport based on the international airport, month, year, number of passengers and total capacity is a supervised classification problem. The dataset at hand has labeled data and the hypothesis requires to see whether it is possible to predict or classify future observations based on known patterns in the data.

In order to repeatedly run different classifiers on data that had been prepared using different encoders and scalers, 3 utility functions were created: split_encode_scale, run_CFL, run_GridSearch (ML Utility Functions section), which have been applied throughout the Machine Learning section of the notebook for experimentation. The table below presents a summary of the experiments run in the Machine Learning section of the Jupyter notebook.

Encoder	**Scaler**	**Balanced**	**Support Vector RBF**	**Decision Tree**	**Random Forest Classifier**	**Hist Gradient Boosting Classifier**
Ordinal	MinMax	N	0.598	-	-	-
		Y 6000 samples	0.51 0.53 Gridsearch	0.68	0.65 0.65 GridSearch	0.73 0.73 Gridsearch
	Standard		0.49	0.68	0.66	0.74
	Quantile		0.49	0.80, **0.69** Gridsearch	0.72	0.75 **0.80** Gridsearch
Sampled 12.000 samples						
OneHot	Standard	Y	**0.69**	0.71	0.69	0.72 0.68 GridSearch
Sampled 30.000 samples						
None	None	N	requires preprocessing	requires encoding	requires encoding	**0.69** **0.73** balanced class weight **0.95** Gridsearch

Table 3: Overview of ML model performance on Air Travel Dataset with and without Pre-Processing

Support Vector Machines

Classification is one of the main areas of application for support vector machines. (Basic Statistical Analysis of SVMs, 2008). In Support Vector Machines (SVM), feature scaling or normalization are not strictly required, but are highly recommended, as it can significantly improve model performance and convergence speed (scikit-learn. (n.d.)).

Hence for the first pass of the support vector machine training experiment, MixMax Scaling was applied. The SVC model showed a 0.60 pct accuracy. Below is the confusion matrix for the run:

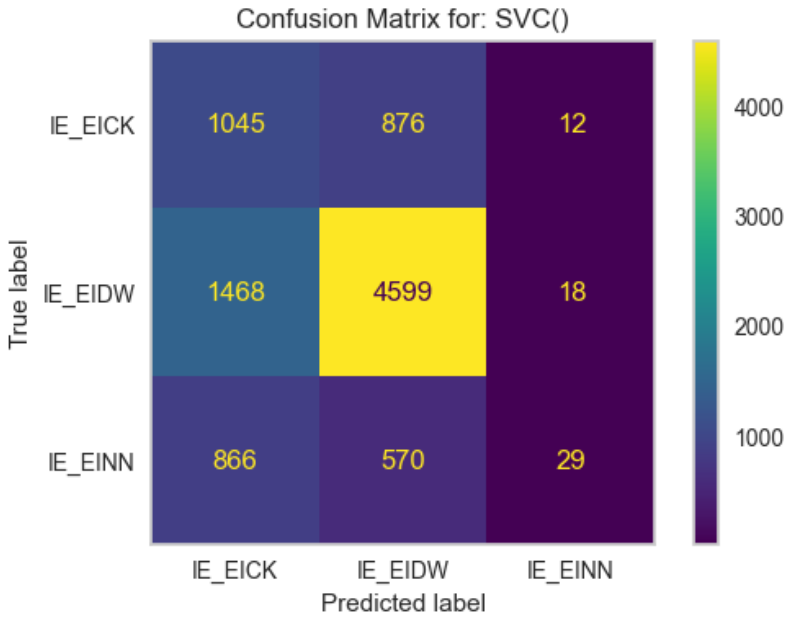

Figure 10: Confusion Matrix for SVC with RBF kernel

The confusion matrix indicated skewed predictions towards the IE_EIDW category, with IE_EINN having a recall of only 0.02. The dataset is imbalance, as reflected in the count of domestic airport instances and hence why the model gives a lot of true positives for the label with overweighting number of rows, while scoring lowest for the airport with the least observations:

IE_EIDW 30402

IE_EICK 9934

IE_EINN 7075

Dataset Balancing

To address dataset imbalance, the NearMiss algorithm was used, however, that resulted in degraded model performance with an accuracy of 51%. Class-imbalanced classification can be improved by utilizing ‘near-miss’ instances (Tanimoto, 2022). It aims to balance class distribution by randomly eliminating majority class examples. The main challenge with undersampling is information loss, hence methods that have proved to enhance SVM performance, such as NearMiss (Bao, 2016), should be used.

As a result of the balancing the SVC performed at 0.51 accuracy which is degraded performance. Hyperparameter tuning, including kernel selection, did not find parameters to improve the model, contrary to the initial assumption that this will improve the model due to dataset balancing. Best parameters were {'C': 10, 'gamma': 1, 'kernel': 'rbf'} and the model accuracy was estimated at 0.53.

One of the reasons behind the degraded performance could be information loss, however, other issues such as inappropriate data encoding, and data scaling could be other factors impacting the model.

OneHot encoding as well as sampling the data, improved the performance of the SVC by about 10%, below is the confusion matrix that demonstrated an increased rate in precision for the category IE_EINN, and an improved recall of 0.42 (up from 0.02). This demonstrates that balancing the sample, as well as choosing a more appropriate encoding and scaling method can have a double digit percent model improvement.

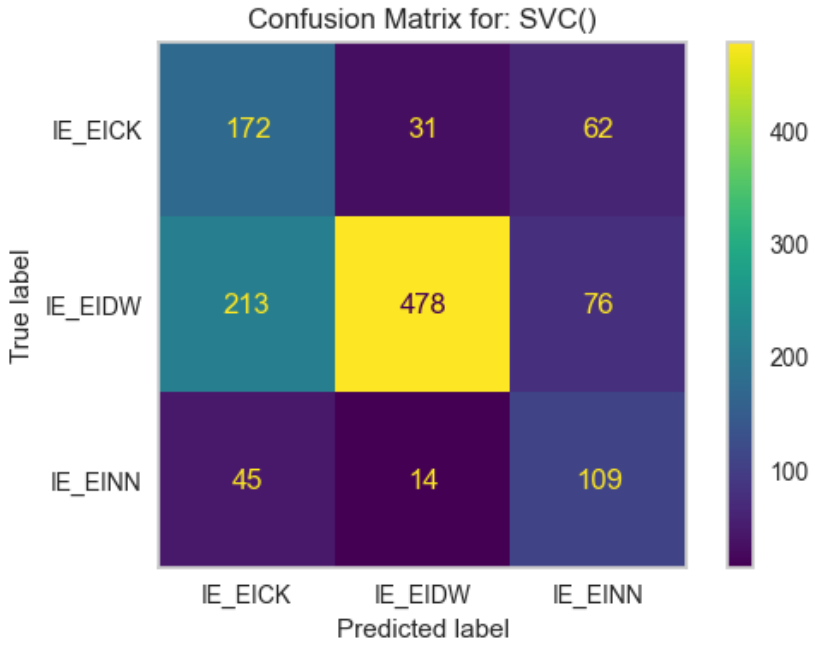

Figure 11: Confusion Matrix for SCV with RBF kernel on a Balanced Dataset after NearMiss Rebalancing

GridSearchCV works by defining a grid of hyperparameters and then systematically training and evaluating a machine learning model for each hyperparameter combination (scikit-learn. (n.d.)). It has been demonstrated that while model performance can achieve high scores when fitted, GridSearch cross validation may produce not only better results, but also worse (Cawley, 2010).

Using GridSearchCV hyperparameter tuning did not find model parameters to raise the score. The assumption that variables, and their errors, are Gaussian distributed is commonplace in areas such as numerical weather prediction and modeling (Goodliff et al., 2022). Since the features of this dataset are lognorm and exponentially distributed, the Support Vector Machine, according to Goodliff et al. may need to be used with other cost functions if this model was to be further improved.

As demonstrated by Ahmad et al., 2022, comparing GridSearchCV outputs for different models can help finetune and select best hyperparameters for poorly performing models, hence the rest of this study will be applying GridSearchCV after each model training and evaluation with defaults / randomly chosen hyperparameter values.

Decision Trees & Random Forests

The two other types of ML models used for this study are Decision Tree Classifier and Random Forest. Random Forests are ensembles of Decision Trees that combine multiple base learners to make more robust predictions (Breiman, 2001). Ensembles of Decision Trees are less prone to overfitting than an individual Decision Tree. Random Forest achieves increased classification performance by aggregating the predictions of multiple trees (Breiman, 2001). It yields accurate and precise results even in the cases of a large number of instances. Moreover, Random Forest is effective in handling missing values in datasets, thereby overcoming the overfitting problem associated with missing data (Cutler et al., 2007). The experiments for this study showed that Decision Tree and Random Forest Classifier showed comparable performance with 0.65 and 0.65 accuracy respectively.

Random Forest is a collection of numerous Decision Trees. Therefore, Random Forest has a simple structure with strong anti-noise capability and can overcome the interpretation capability disadvantage of a single Decision Tree (Lahouar, 2015). Hence, the expectation is that with proper hyperparameter tuning and feature selection, the Random Forest will show better predictive capabilities, which will be explored later in this report.

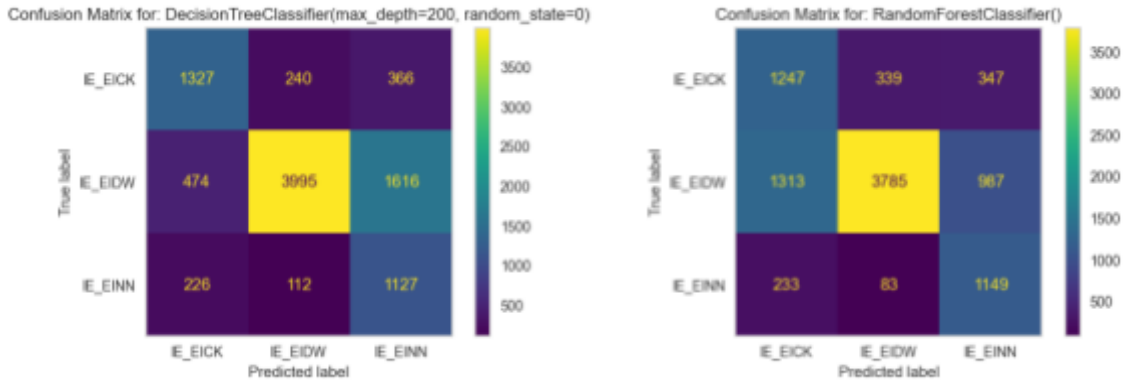

Figure 12: Decision Tree and Random Forest Showing Comparable Results

HistGradientBoostingClassifier

Another Ensemble model for tried on the dataset - HistGradientBoostingClassifier. Boosting, which leverages the concept of ensemble supervised learning to strengthen the detection capabilities. By combining several learners into a strong model, boosting effectively reduces bias and variance in the prediction process (Bentéjac, 2021). Those simple models are called weak learners or base estimators. It aggregates all predictions from its constituent learners in a sequential manner (Saied, 2023).

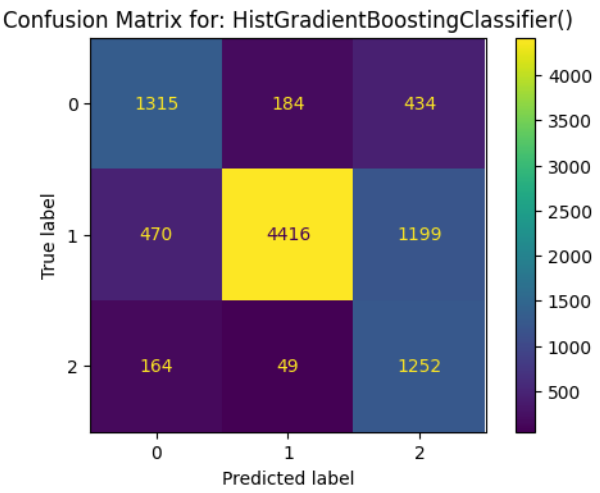

Figure 13: Hist Gradient Boosting

HistGradientBoostingClassifier resulted in a 0.73 accuracy - best score achieved thus far, and hyperparameter tuning with GridSearchCV suggested 500 iterations for the boosting process to achieve an optimal model.
All results for hyperparameter tuning for the 3 models run for this study, can be found in the `cross-val` folder of the referenced repository.

Impact of Scalers

In terms of the effect of scalers, there was no discernible difference in performance of SVC, Decision Tree and Random Forest when employing data scaled with MinMax, Standard Scaler, and Quantile Transformer. Further details can be found in the "Repeat With Other Scalers" section of the notebook. However, it's noteworthy that the HistGradient Boosting Classifier achieved an accuracy of 0.80 after resampling and applying quantile transformation. Additionally, it performed optimally with 300 boosting iterations, as determined by GridSearchCV. These outcomes are depicted in the plotted classification report below.

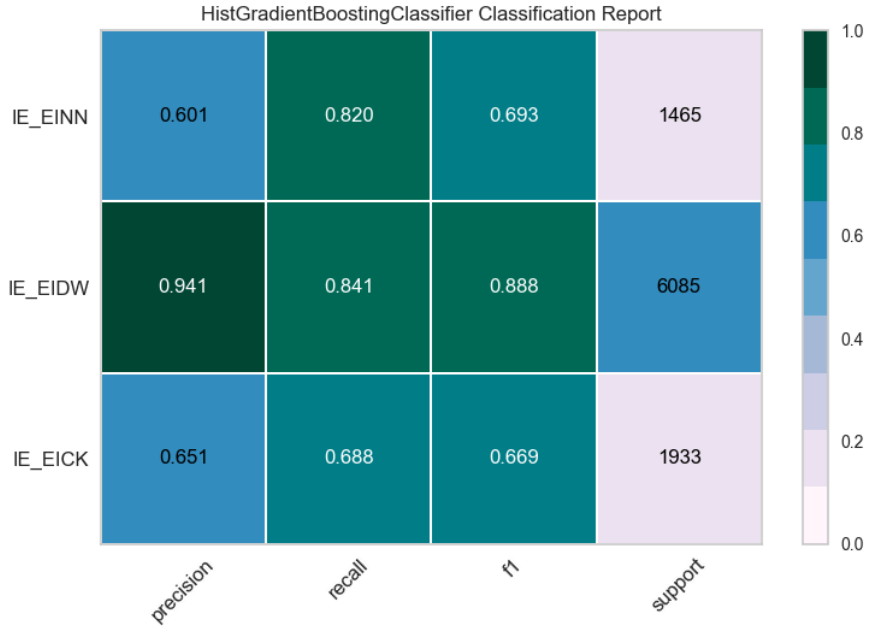

Figure 14: Classification report for HistGradientBoosting Classifier encoded with quantile encoder on a balanced sampled dataset

Running One-Hot encoding and using Standard helped achieve a 0.69 accuracy score for the SVC, while ensemble and boosting models exhibited the same performance as before.

Tree-based models are typically considered insensitive to feature scaling (Breiman et al., 1984). Our experiments corroborate this notion, as we observed only minor improvements in performance after feature scaling was applied. This could be attributed to the phenomenon where features with larger scale dominate those with smaller scale, potentially overshadowing any benefits of scaling (Cutler et al., 2007, Friedman et al., 2001). Interestingly, in our study, we noted that HistGradient Boosting exhibited much better performance when trained on non-processed features.

For datasets with categorical features, using the native categorical support is often better than relying on one-hot encoding, because one-hot encoding requires more tree depth to achieve equivalent splits (scikit-learn, n.d.). It is also usually better to rely on the native categorical support rather than to treat categorical features as continuous (ordinal), which happens for ordinal-encoded categorical data, since categories are nominal quantities where order does not matter (scikit-learn, n.d.). Hence, when training HistGradientBoosting Classifier on non-processed dataset, the model scored at 0.95 (Figure 15 A). While this model exhibited the best performance, it warrants careful consideration due to the potential for overfitting when presented with new data. So, an alternative approach would involve testing this model through GridSearchCV with a train/test data split of 70% for training and 30% for testing. The outcomes showed a decline in performance, with an accuracy of 0.73 observed when GridSearch was conducted on the re-split data, as illustrated in **Figure 14B**.

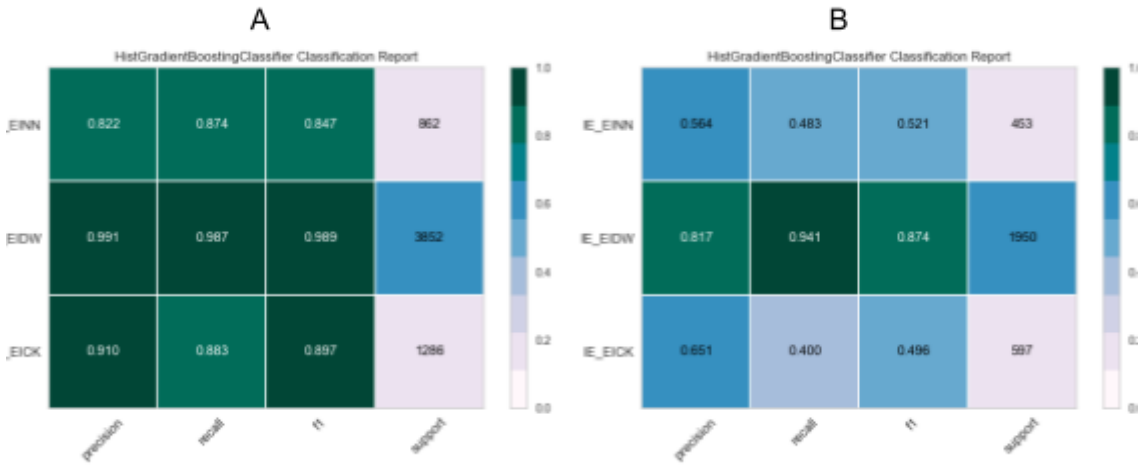

Figure 15: Classification report for HistGradientBoosting Classifier without data preparation and in-built encoder for 80/20 vs 70/30 train/test data split

Permutation Importances

Finally, permutation importances were utilized as a "model inspection technique to gauge the contribution of each feature to the performance of a fitted model" (scikit-learn.org, n.d.). This method involves randomly shuffling the values of a single feature and observing the consequent decline in the model's score. By disrupting the

relationship between the feature and the target variable, it can help assess the degree to which the model depends on each feature (Altmann, 2010).

Permutation importances were not run for SVCs. Figure 15 demonstrates how the different features in the dataset gain a different permutation importance score, depending on the scaler used. Unfortunately, it was not possible to run this for non-processed data fed to the HistGradientBoosting Classifier, as some values could not be properly handled without pre-processing. The graphs demonstrate that with different scalers some features tend to gain a very different importance score. As such, 'month' feature acquired a heavy weights after Quantile Transformer was applied.

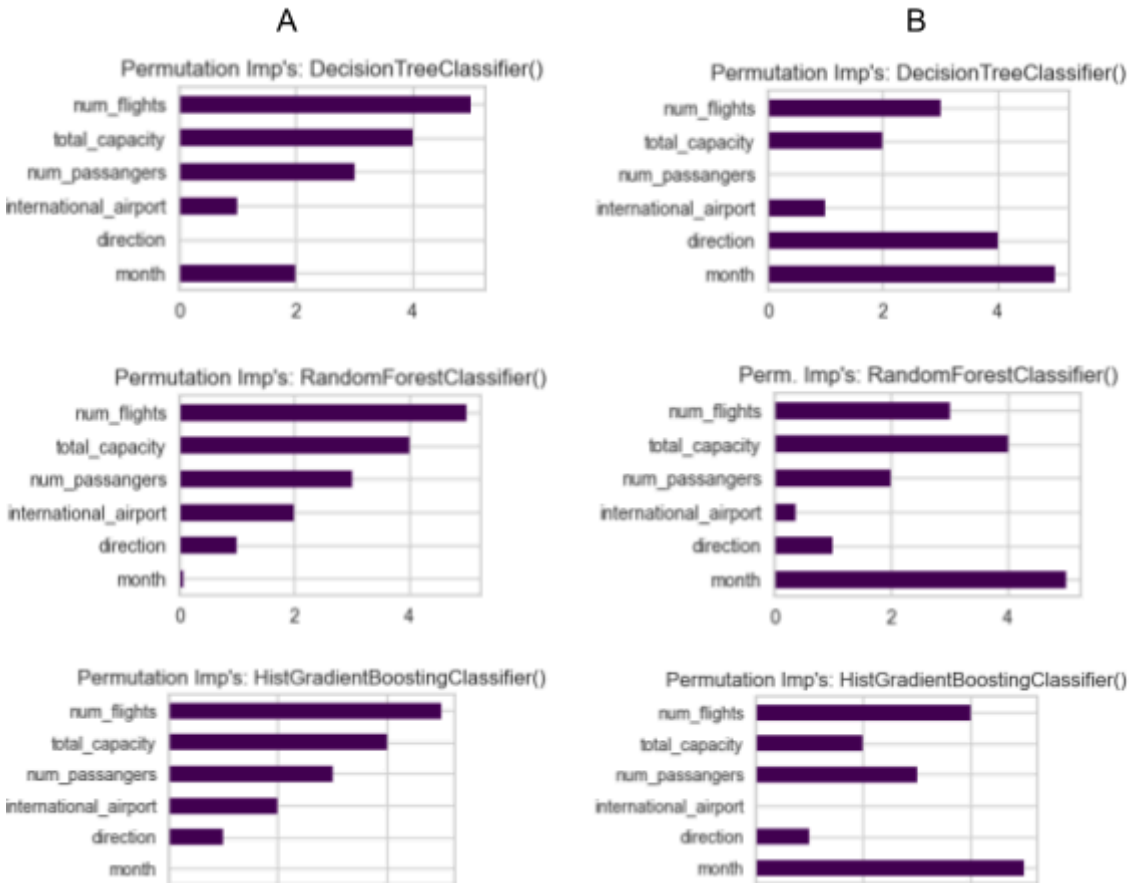

Figure 16: Permutation Importances (A) on resampled data with Standard Scaling to the left vs (B) resampled with Quantile Transformer scaling to the right

A study conducted by Caruana (2006) compared 10 different binary classifiers, including SVM, Neural Networks, KNN, Logistic Regression, Naive Bayes, Random Forests, Decision Trees, Bagged Decision Trees, Boosted Decision Trees, and Bootstrapped Decision Trees, across eleven distinct datasets. The study evaluated the performance of these classifiers using eight different performance metrics. It was found that Boosted Decision Trees ranked first, followed by Random Forests, Bagged Decision Trees, and SVM, respectively, though the outcome may vary depending on the number of classes being classified. This study's findings were consistent with the results obtained in a comparison of Support Vector Machines, Decision Trees, Random

Forests, and HistGradientBoosting Classifier. Additionally, this study highlighted the influence of data preprocessing on the performance of different classifiers.

Conclusions

The conclusions drawn from this study include several insights into tourism dynamics in Ireland. Analysis of Air and Sea Travel from 2010 to 2023 dataset indicates a consistent upward trend in tourist arrivals to Ireland, with 2023 surpassing pre-pandemic levels seen in 2019. Continental Europe appears to be the main route of tourist arrivals, overtaking Cross Channel routes since 2014. This suggests evolving preferences in travel destinations among tourists. By assessing flight occupancy rates, the study provides insights into the demand for air travel to Ireland, potentially facilitating planning for airlines and tourism authorities. The use of machine learning models shows their effectiveness in predicting tourism patterns. HistBoosting Classifier demonstrated better performance without extensive data preprocessing, suggesting that efficient model selection can yield good results without data preparation.

References

Altmann, A., Toloşi, L., Sander, O. and Lengauer, T. (2010). Permutation importance: a corrected feature importance measure. Bioinformatics, 26(10), pp.1340–1347. doi:https://doi.org/10.1093/bioinformatics/btq134.

Bao, L., Cao, J., Li, J. and Zhang, Y. (2016). Boosted Near-miss Under-sampling on SVM ensembles for concept detection in large-scale imbalanced datasets. *Neurocomputing*, 172, pp.198–206. doi:https://doi.org/10.1016/j.neucom.2014.05.096.

Basic Statistical Analysis of SVMs. (2008). *Information science and statistics*, pp.202–237. doi:https://doi.org/10.1007/978-0-387-77242-4_6.

Bell, P. (2019). How Pew Research Center uses small multiple charts. [online] Pew Research Center: Decoded. Available at: https://medium.com/pew-research-center-decoded/how-pew-research-center-uses-small-multiple-charts-2531bfc06419 [Accessed 30 Mar. 2024].

Bentéjac, C., Csörgő, A., Martínez, G.: A comparative analysis of gradient boosting algorithms. Springer, Netherlands 54(3), 1937–1967 (2021)

Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

Cawley GC, Talbot NLC (2010) On over-tting in model selection and subsequent selection bias in performance evaluation. The Journal of Machine Learning Research https://www.jmlr.org/papers/volume11/cawley10a/cawley10a.pdf

Caruana, R. and Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. Proceedings of the 23rd international conference on Machine learning - ICML '06. doi:https://doi.org/10.1145/1143844.1143865.

Crow, E.L. (1988). Lognormal distributions : theory and applications. New York, Ny: Dekker.

Cutler, D. R., Edwards Jr, T. C., Beard, K. H., Cutler, A., Hess, K. T., Gibson, J., & Lawler, J. J. (2007). Random forests for classification in ecology. Ecology, 88(11), 2783-2792.

Iversen, I.H. (1988). TACTICS OF GRAPHIC DESIGN: A REVIEW OF TUFTE'S THE VISUAL DISPLAY OF QUANTITATIVE INFORMATION. Journal of the Experimental Analysis of Behavior, 49(1), pp.171–189. doi:https://doi.org/10.1901/jeab.1988.49-171.

M. Goodliff, Fletcher, S.J., Kliewer, A., Jones, A.S. and Forsythe, J.M. (2022). Non‐Gaussian Detection Using Machine Learning With Data Assimilation Applications. Earth and space science (Hoboken, N.J.), 9(4). doi:https://doi.org/10.1029/2021ea001908.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn and TensorFlow concepts, tools, and techniques to build intelligent systems*. 2nd ed. O'Reilly Media, Inc.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of statistics, 29(5), 1189-1232.

Huang, N., Lu, G. and Xu, D. (2016). A Permutation Importance-Based Feature Selection Method for Short-Term Electricity Load Forecasting Using Random Forest. Energies, 9(10), p.767. doi:https://doi.org/10.3390/en9100767.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). An Introduction to Statistical Learning. New York, Ny Springer New York.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. AI magazine, 17(3), 37-37.

Kundu, D. and Ganguly, A. (2017). Analysis of Step-Stress Models. Academic Press.

Lahouar, A. and Ben Hadj Slama, J. (2015). Day-ahead load forecast using random forest and expert input selection. Energy Conversion and Management, 103, pp.1040–1051. doi:https://doi.org/10.1016/j.enconman.2015.07.041.

Marc Peter Deisenroth, A Aldo Faisal and Cheng Soon Ong (2020). *Mathematics for machine learning*. Cambridge, United Kingdom ; New York, Ny: Cambridge University Press.

Potdar, K., S., T. and D., C. (2017). A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers. *International Journal of Computer Applications*, 175(4), pp.7–9. doi:https://doi.org/10.5120/ijca2017915495.

Saied, M., Guirguis, S. and Madbouly, M. (2023). A Comparative Study of Using Boosting-Based Machine Learning Algorithms for IoT Network Intrusion Detection. International Journal of Computational Intelligence Systems, 16(1). doi:https://doi.org/10.1007/s44196-023-00355-x.

scikit-learn. (n.d.). sklearn.ensemble.HistGradientBoostingClassifier. [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.HistGradientBoostingClassifier.html [Accessed 29 Mar. 2024]

scikit-learn.org. (n.d.). 4.2. Permutation feature importance — scikit-learn 0.23.1 documentation. [online] Available at: https://scikit-learn.org/stable/modules/permutation_importance.html [Accessed 1 Apr. 2024]

scikit-learn. (n.d.). 1.11. Ensembles: Gradient boosting, random forests, bagging, voting, stacking. [online] Available at: https://scikit-learn.org/stable/modules/ensemble.html#categorical-support-gbdt.

Tanimoto, A., Yamada, S., Takenouchi, T., Sugiyama, M. and Kashima, H. (2022). Improving imbalanced classification using near-miss instances. *Expert Systems with Applications*, 201, p.117130. doi:https://doi.org/10.1016/j.eswa.2022.117130.

West, R.M. (2021). Best practice in statistics: The use of log transformation. Annals of Clinical Biochemistry: International Journal of Laboratory Medicine, 59(3), pp.162–165. doi:https://doi.org/10.1177/00045632211050531.

Zheng, A. and Casari, A. (2018). *Feature Engineering for Machine Learning Principles and Techniques for Data Scientists*. Oreilly & Associates Inc Wiesbaden Divibib Gmbh.