Machine Learning CSC8635 Project Report – Peter Bevan (200731331)

Introduction and Exploratory Data Analysis:

The titanic was undoubtedly the most famous shipwrecking in recorded history and has been the subject of several movies and documentaries since. The huge 'unsinkable' passenger ship crashed into an iceberg and sank on its first voyage across the Atlantic. Such was the hubris about the hardiness of the ship, along with lacklustre safety regulations at the time, an insufficient number of lifeboats had been installed on the ship, and as a result a large proportion of the passengers died in the disaster; of the 2,224 passengers onboard, 1502 died (History.com, 2009). Data was collected about the passengers onboard which allows analysis of the passengers that survived and the passengers that died. The survivors were not completely chosen at random, for example, History.com states that passengers travelling first class were 44% more likely to survive. The clear interplay between certain facts about each passenger and survival mean the data collected about the passengers can be used to predict who survived and who died. This project will aim to implement machine learning models that use passenger information such as age, gender, embarkation location etc, as well as engineered features such as title and cabin letter to predict passenger survival to a high degree of accuracy. Four different machine learning models were experimented with: logistic regression, random forest classifier, multinomial naïve bayes and a multi-layer perceptron deep learning model.

The data was provided as a training set of 891 observations and a test set of 418 observations. The head of the training data is shown below to give a feel for the structure of the data and the nature of the variables. There are 12 variables in total, with 'Survived' being the response variable and the other 11 being the predictor variables when building a model for prediction of passenger survival.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

Figure 1: training data head

Descriptive information about the structure of the training dataset is shown in the table below, with the exception of 'name', 'cabin', 'sex' and 'embarked', as these are non-numerical. Some conclusions from this table are as follows:

- It's clear that there are 891 entries in the training dataset.
- 'Survived' is shown to be a binary variable, (0 for 'Survived' = FALSE and 1 for 'Survived' = TRUE), and the 'Survival' mean of 0.38 shows that 38% of people survived in the dataset.
- There are three classes (1^{st}-3^{rd})
- The average age was 29.69 and average fare was 32.2.
- The count values of each variable show that there's some values missing from 'Age' (177 missing).

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Table 1: Descriptive statistics for training data

The missing values were explored graphically as shown below using the 'missingno' library (white space indicates missing values). This also revealed many missing values in the 'Cabin' column and the 'Embarked' column that were not noticed as they were not included in the summary due to being categorical.

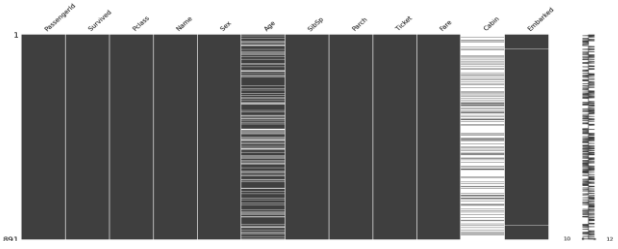
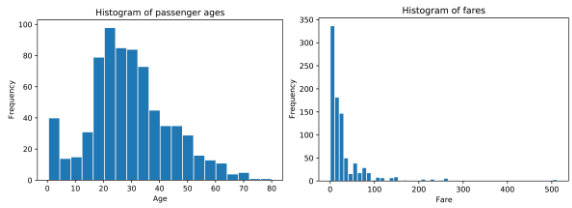


Figure 2: Missing value visualisation

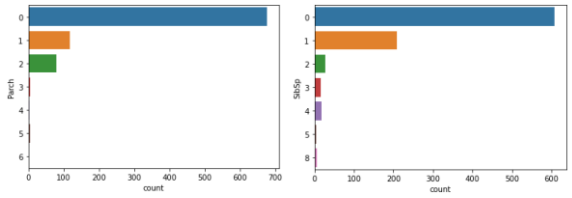
A histogram/bar chart was created for each variable in the training data to explore their distribution. These are displayed below except for those where each value is a unique value ('Name', 'PassengerId' and 'Ticket'). Passenger age is shown to have a roughly normal distribution judging by the bell shape of the histogram distribution. The most popular fare is shown to be less than 10, with the vast majority under 30 and then diminishing frequencies of more expensive tickets all the way up to over 500.



Figures 3 & 4: Age histogram (left), Fares histogram (right)

The bar 'Parch' plot below shows that the vast majority of passengers had no parents or children aboard, but if they did, most of these people had 1 or 2. There is then a very small amount of people who had between 3 and 6 parents/children aboard.

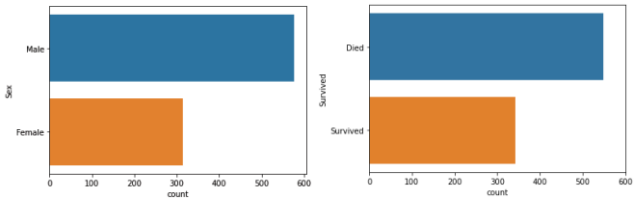
The 'SibSp' plot below shows that most passengers had no sibling or spouse aboard, and the majority that did, had just 1. There is a significant minority of people who had between 2 and 4 siblings/spouses aboard and then a very small number of people with 5-8.



Figures 5 & 6: Parents/children bar plot (left), sibling/spouse bar plot (right)

The 'Sex' bar plot below shows that there were almost twice the number of male passengers compared to female passengers.

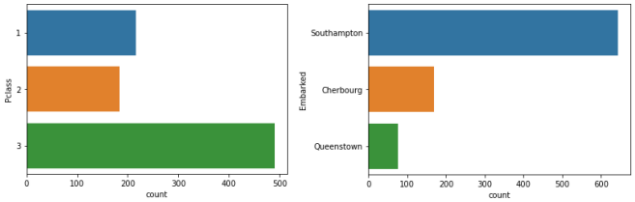
The 'Survived' bar plot on the right below shows that many more people died in the training set than survived (~350 to ~550).



Figures 7 & 8: Sex bar plot (left), survived bar plot (right)

The 'Pclass' bar plot below shows that the majority of passengers in the training set were in 3rd class (almost 500), while similar amounts were in 1st and 2nd class, with around 200 in each (slightly more in 1st class than 2nd)

Most of the passengers in the training set embarked from Southampton, with some from Cherbourg and the least from Queenstown.



Figures 9 & 10: Passenger class bar plot (left), embark location bar plot (right)

Following the exploration of individual variables, the correlation between each variable was visualised using a correlation matrix, utilising heatmap formatting to highlight the strength of negative and positive linear correlations

between pairs of variables. Some pre-processing was done prior to this to one hot encode the categorical variables 'Sex' and 'Embarked' in order to allow numerical comparison. One of the new dummy variables from each categorical variable was dropped to make this the baseline, to help reduce the dimensionality of the data (Cope, G). The strongest correlation is a negative correlation between fare and class with a coefficient of -0.55. This doesn't offer a great deal of insight as it is to be expected that the higher classes will cost more. Another strong correlation is a negative one between being male and survival with a negative correlation of -0.54. This suggests that there is good indication that being female increases the chance of survival. There is also a reasonable negative correlation between passenger class and survival (to be expected that higher class citizens are saved first) as well as a notable positive correlation between fare and survival (to be expected due to the previously mentioned link between fare and class). There are various other correlations between variables shown below, indicating that there is a chance that some of the variables are not providing independent information and we may be able to reduce the dimensionality of the data without losing too much of the total variation (this may reduce overfitting and increase interpretability).

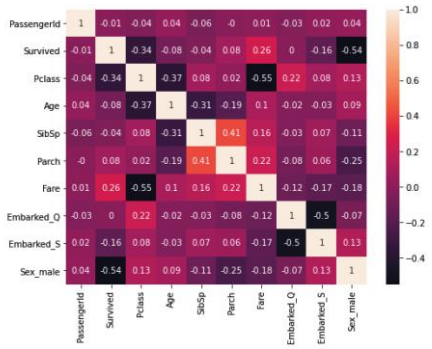


Figure 11: Correlation matrix heatmap of training data

Data Processing & Feature Engineering:

The two missing values for 'Embarked' were filled with the mode value (Southampton, S) since this variable is categorical and mean/median can't be used. To fill in the missing values of 'Fare', the median was used since the distribution is skewed. To fill in the missing values of 'Age' the mean was used since the distribution is roughly normal.

To extract a useful and comparable feature from the 'Cabin' feature, the first letter of each cabin entry was extracted and stored in a new feature called 'Cabin_letter'. Any individual with an N/A entry under 'Cabin' was extracted as 'Cabin_letter'=Unknown. This new feature would hopefully allow the models to predict more accurately based upon the type of cabin each passenger had (or didn't have).

Another feature that is not comparable as-is was the 'Name' feature since each name is unique. To extract common values between individuals, the titles of each passenger were extracted from their name and aggregated into groups of similar titles ('Master', 'Miss', 'Mrs', 'Honoured' and 'Noble'). These groups were chosen with the aim of splitting by groups most likely to have different chances of survival, which should hopefully increase the accuracy of the models.

The newly created 'Cabin_letter' and 'Title' features were one-hot encoded to represent these categorical variables numerically and allow the machine learning models to use the data to make better predictions. The first category in each was used as the baseline which allowed that category to be dropped and become implied by a 0 for all other categories of the same type.

A new binary feature 'Alone' was created using a combination of 'SibSp' and 'Parch'; if the sum of both of these features was greater than 0, 'Alone' was set to 1, otherwise it was set to 0, indicating whether the passenger had any family onboard (siblings, spouses, parents, children). This feature was deemed to be potentially useful since the models wouldn't know the relation between 'SibSp' (siblings and spouses) and 'Parch' (parents and children), and there may be a link between having any family on board at all and survival. 'SibSp' and 'Parch' were kept as they provide more specific detail to the model, and can be dropped in feature selection if necessary. Testing the models with and without the new 'Alone' feature seemed to show a slight improvement in prediction accuracy for some models and so the feature was kept, but it was trimmed off by some of the feature selection algorithms. Originally a feature 'Family' was created, however this was later removed to reduce model complexity and reduce the risk of overfitting. The feature was repeatedly being removed by the RFECV feature selection algorithm and so the decision was made comfortably.

Quantile binning was used as a discretization transformation on the 'Age' data to split it into 7 roughly equal sized groups, and the same on the 'Fare data' (6 groups). This mitigates irregularities in the distributions of these continuous numerical variables and may help the machine learning models to predict more accurately (Kuhn, M. & Johnson, K., 2019). These new bins could then also be one hot encoded if necessary, meaning all of the data could be formatted as one hot encoding rather than scaling the data (one hot encoding of all predictors was used later in the report for the multi-layer perceptron).

The missing entry visualisation was revisited to be sure that all missing values had been dealt with, and to visualise the structure of the data and features. The solid black bars show that all the data was complete and lay out how some of the features have been one hot encoded.

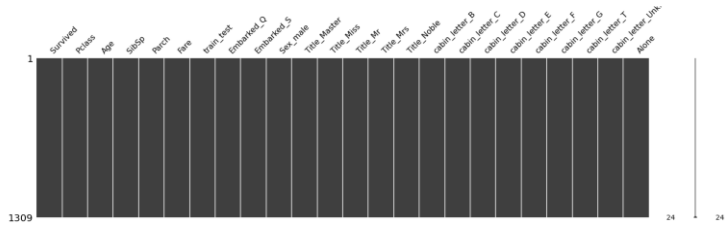


Figure 12: missingno revisited

The training and test data were further split into predictor (X) and response (y) variables. For the training data this will give the machine learning models labels to learn from, and in the case of the test data this will provide the ground truth to compare predictions to and calculate the final test error/accuracy. The test and train predictors were cloned, and the non-binary features were scaled to between 0 and 1. This range was chosen as it puts these features in the same range as the binary one hot encoded features (0-1). The original test and train data were also kept in case it was necessary to process it in a different way for different models.

Modelling methods:

Logistic Regression

The first model attempted was logistic regression, using Sklearn's LogisticRegression(). Feature selection was performed using the RFECV (recursive feature elimination cross validated) algorithm, in an attempt to reduce overfitting and increase interpretability. This algorithm performs feature ranking and recursively eliminates features, selecting the best subset of features based upon cross validated accuracy scores within the training set. The algorithm selected 19 features of the 22 features fed to it, which are shown below:

```
Optimal number of features: 19
Selected features: ['Pclass', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked_S',
'Sex_male', 'Title_Master', 'Title_Miss', 'Title_Mr', 'Title_Mrs', 'cabin_let
ter_B', 'cabin_letter_C', 'cabin_letter_D', 'cabin_letter_E', 'cabin_letter_
F', 'cabin_letter_G', 'cabin_letter_T', 'cabin_letter_Unknown']
```

The algorithm trimmed off 'Embarked_Q', 'Alone' and 'Title_Noble'. Perhaps the noble category wasn't much different to the baseline title category 'Honoured'. To select the best hyperparameters GridSearchCV was used to perform an exhaustive, cross validated search over the parameter values specified. The grid was defined using a selection of solvers, penalties, and c-values. The selected hyperparameters are shown in the output below along with the cross validated accuracy of the model on the training set.

```
Training error: 0.1661 using {'C': 1.0, 'penalty': 'l2', 'solver': 'newton-c
g'} (83.39% accuracy at predicting validation response variables)
```

5 fold cross validation was chosen to estimate the accuracy of the model at predicting unseen response variables rather than a test/validation split, as it has been shown to give better estimations due to all observations having the chance to appear in the training and test dataset. This is especially true on smaller datasets such as in this project. Cross validation also helps to reduce overfitting of the data to the test set to create a more generalisable model.

Random Forest Classifier

Feature selection was also performed using RFECV for the random forest classifier. The cross validated feature selection algorithm reduced the dimensionality to 9 features as shown below. This large reduction in the number of features, together with the intrinsic interpretability of decision trees and random forests make for an easily interpretable model. This interpretability should be considered alongside the accuracy scores when deciding on a champion model.

```
Optimal number of features: 9
Selected features: ['Pclass', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked_S',
'Sex_male', 'Title_Mr', 'cabin_letter_Unknown']
```

GridSearchCV was again used to tune the hyperparameters. The hyperparameters compiled into the grid for the algorithm to iterate through were as follows: number of trees in forest, number of features to consider when splitting, maximum depth of the tree, minimum samples to split internal node, minimum samples at a leaf node, and whether to use bootstrap samples or not. Realistic ranges of values were provided to the grid for each hyperparameter, and the model with the combination of these values that predicted most accurately was chosen. Due to the large number of hyperparameters being tuned and the 5-fold cross validation this calculation was computationally heavy. To mitigate this, RandomisedSearchCV was substituted for GridSearchCV (picks the best of random combinations of hyperparameters), and also the parameter n_jobs=-1 was passed to initiate parallel computing across all cores of the machine.

```
Training error: 0.164980227229929 using {'n_estimators': 800, 'min_samples_sp
lit': 10, 'min_samples_leaf': 2, 'max_features': 'sqrt', 'max_depth': 20, 'bo
otstrap': False} (83.5% accuracy at predicting training response variables)
```

As can be seen the cross validated training accuracy of the random forest classifier model was 83.5%.

Multinomial Naïve Bayes

A multinomial naïve Bayes model was chosen over Gaussian naïve Bayes model since all the data is either one hot encoded categorical or discretised continuous data (rather than GaussianNB suitable normally distributed continuous data) and so the former is more suitable. Feature selection was performed using the RFECV algorithm and yielded the 16 features shown below.

```
Optimal number of features: 16
Selected features: ['SibSp', 'Parch', 'Embarked_Q', 'Sex_male', 'Title_Maste
r', 'Title_Miss', 'Title_Mr', 'Title_Mrs', 'Title_Noble', 'cabin_letter_B',
'cabin_letter_C', 'cabin_letter_D', 'cabin_letter_E', 'cabin_letter_F', 'cabi
n_letter_G', 'cabin_letter_T']
```

Additive smoothing was tuned using GridSearchCV, which output an alpha value of 1 as optimal, with a cross validated training accuracy of 79.12%.

```
Training error: 0.20878162073943884 using {'alpha': 1} (79.12% accuracy at pr
edicting training response variables)
```

Multi-Layer Perceptron

A multi-layer perceptron (MLP) was chosen to test the effectiveness of deep learning at predicting the titanic dataset. The MLP was chosen in favour of a convolutional or recurrent neural network since the task is quite basic and is not an image recognition or sequence prediction problem. The Keras library for python (TensorFlow backend) was used and Keras tuner random search was utilised for selection of the optimal number of layers and neurons in each of these layers. The number of epochs to train the model was played with until a trade off between speed of training and model accuracy was identified at around 50 epochs. The number of trials was set at 5, and executions at each trial set to 3, to ensure a reliable idea of the training accuracies was obtained.

During iterative tuning of the deep learning model it was noted that the ‘accuracy’ (training accuracy) was around 10% higher than the ‘val_accuracy’ (validation accuracy), suggesting overfitting. To try to reduce this overfitting, the number of features was reduced to try and reduce the complexity of the model. The 8 features selected by the cross validated recursive feature elimination carried out on the random forest regression classifier were used as the selected features. This feature selection certainly reduced the complexity of the selected best models, with fewer nodes and layers being selected in the top 5 MLP models based on validation accuracy. When viewing the accuracy vs validation accuracy across each epoch and trial during tuning, there did seem to be a reduced difference between the two, down to around 5%, however the validation accuracy came down slightly. This slight validation accuracy reduction was accepted as a trade-off for the potentially increased generalisability of the model. The selected model validation accuracy and layer details are shown below.

```
Hyperparameters:
input_units: 224
n_layers: 2
dense_0_units: 208
dense_1_units: 64
dense_2_units: 144
Score: 0.8482587138811747
```

<u>Discussion and test set results:</u>

Care was taken not to allow the models to be exposed to the test data during the training and tuning process to ensure that the test data was truly unseen and gave a reliable metric for judging how well the models generalise. If a model had to be chosen without any test data, as is often the case in real world situations, the Random Forest classifier would

have been chosen, despite only having the 2nd highest cross-validated training accuracy (83.5% compared to 84.8% from the MLP). The simplicity and interpretability of the random forest model made it more favourable since it can quite easily be explained to decision makers and affect real change, with only a small difference in estimated accuracy. In the situation of advising someone whether to board the titanic, armed with one model, the random forest classifier would likely be the best combination of persuasive and accurate.

After tweaking the models based on validation using the training data, the final step was to use the models to predict the response variables of the test set, to find out how well they generalised to unseen data. The table below shows the training and test accuracy of each model (accuracy calculated as the percentage of predictions that matched the ground truth). There is not a great degree of variation between the test set results, with multi-layer perceptron scoring the highest by a margin of 0.5%.

	Model	Training Accuracy	Test Accuracy
0	Multi-Layer Perceptron	85.323383	77.511962
1	Multinomial Naive Bayes	79.121838	77.033493
2	Logistic Regression	82.491369	76.555024
3	Random Forest	83.726696	76.315789

Test accuracy percentage scores are just one of several ways to evaluate the performance of the models. To further understand the performance of each model, confusion matrixes were generated, which uncover the predicted values vs the ground truth values. As can be seen below, the MLP model had the lowest occurrence of false positive predictions for survival, but the highest occurrence of false negatives, suggesting the model is potentially biased towards predicting passengers dying, which is not necessarily an issue and can be accepted as part of the bias/variance trade off that is common for machine learning models, given the high accuracy of the model in comparison to other models.

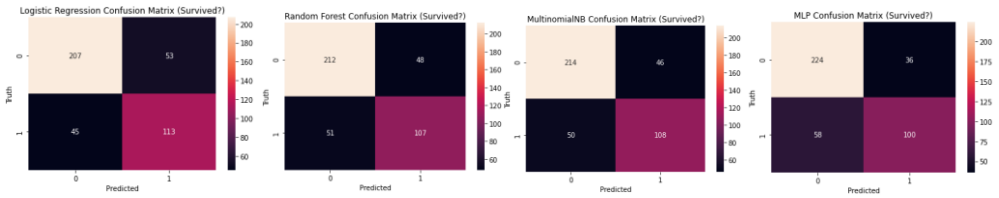


Figure 13: Confusion heatmap matrixes for each machine learning model, left to right: logistic regression, random forest, naive bayes, MLP

Despite best efforts during training and tuning, many of the models do still seem to be overfit to the training data, especially random forest and MLP, with a difference of 8% and 9% respectively from training validation accuracy to test accuracy. The feature selection and 5-fold cross validation were implemented to keep this minimised, but perhaps 10-fold cross validation would have reduced the overfitting further. Another cause of the overfitting is likely how small the dataset is, which can't be combatted given that this was a one-time event. This is potentially why there is more overfitting in the MLP and random forest as these are data hungry models, and lack of data has a greater impact on overfitting. The multinomial naïve Bayes model seems to generalise well, with similar cross validation accuracy and test accuracy, which is likely because of the simplicity of the model and its inherent ability to deal with small datasets.

Conclusion

Whichever model was chosen (random forest in this case as already mentioned), there was good test accuracy across all models (between 76% and 78%). This suggests that there is fairly consistent structure to the data, allowing predictions to be made accurately even with the small number of observations.

In the situation of being given the opportunity to board the titanic on its maiden voyage, it would almost never be a good idea, even if a futuristic machine learning model told you that you would likely survive due to your class, age, sex, etc. Even the passengers that survived likely suffered from mental and physical health issues following the trip, not to mention a thoroughly bad day on September 1st 1912. One lesson that can be learnt from this is that the prediction goal of a model is the first and foremost important feature in the model, rather than solely seeking accuracy at predicting anything. Predicting something with high accuracy that is not well aligned with any useful goals is useless, and is essentially a waste of time and resources, unless being used as an educational exercise like this. The main conclusions from the titanic disaster did not require machine learning to figure out; the design was poor, there weren't enough lifeboats and the safety regulations needed improving. Perhaps machine learning would have been slightly useful in interpreting the reasons for certain passengers surviving, highlighting discrimination, but there would surely be no aim to make deaths equal among citizens; rather an aim to prevent deaths entirely.

Bibliography

Cope, G., 2021. *Dummy Variable Trap in Regression Models*. [online] Algosome.com. Available at: <https://www.algosome.com/articles/dummy-variable-trap-regression.html#:~:text=The%20Dummy%20Variable%20trap%20is,%2Ffemale)%20as%20an%20example.> [Accessed 18 January 2021].

Kuhn, M. and Johnson, K., 2019. *Feature Engineering and Selection*. Chapman and Hall/CRC, pp.129-131.

Raschka, S. and Mirjalili, V., 2019. *Python Machine Learning*. 3rd ed. Birmingham: Packt Publishing, Limited, pp.100-103.

Pythonprogramming.net. 2019. *Python Programming Tutorials*. [online] Available at: <https://pythonprogramming.net/keras-tuner-optimizing-neural-network-tutorial/> [Accessed 1 January 2021].

HISTORY. 2009. *Titanic*. [online] Available at: <https://www.history.com/topics/early-20th-century-us/titanic> [Accessed 11 January 2021].

Scikit-yb.org. 2021. *Recursive Feature Elimination — Yellowbrick v1.2.1 documentation*. [online] Available at: <https://www.scikit-yb.org/en/latest/api/model_selection/rfecv.html> [Accessed 09 January 2021].

Scikit-learn.org. 2021. *Parameter estimation using grid search with cross-validation* — scikit-learn 0.24.1 documentation. [online] Available at: <https://scikit-learn.org/stable/auto_examples/model_selection/plot_grid_search_digits.html> [Accessed 09 January 2021].