A mathematical essay on logistic regression

Gautham Govind A
Dept. of Electrical Engineering
Indian Institute of Technology Madras
ee19b022@smail.iitm.ac.in

***Abstract*—The objective of this assignment is to explore the mathematical formalism behind logistic regression and then to use it in a real-life application. In this assignment, as a real-life application, logistic regression is used to formally identify the factors which could have been used to predict the chance of survival of an individual in the historical sinking of The Titanic. Data visualization, cleaning and modelling is done using Python. The analysis enables us to arrive at the conclusion that although luck played a major role in predicting survival, other factors like socioeconomic status and gender does indeed have an impact on the chance of survival. This is a reworked version of the original assignment with improvements to the Modelling section in the form of enhancements to the vanilla logistic regression model.**

***Index Terms*—logistic regression, python, visualization, predictive modelling**

I. Introduction

The sinking of Titanic is one of the most tragic incidents in the history of humanity. On April 15, 1912, during her maiden voyage, the widely considered “unsinkable” RMS Titanic sank after colliding with an iceberg, resulting in the death of 1502 out of the total 2224 individuals (including passengers and the crew). Although there definitely was some element of luck involved in determining who survived and who did not, it is necessary to examine if any other factors(such as socioeconomic status) played a role in determining the survival of an individual. This can help in bringing out involuntary biases which may play a role in imposing disadvantages on certain sections of the society.

In this assignment, the goal is to make use of logistic regression to tackle the above mentioned problem. Logistic regression is a popular mathematical model for estimating how a binary variable is influenced by a variety of known factors. By performing logistic regression analysis, we will be able to get an idea of how each feature influences the target variable. This model can then be used for predicting the target variable from the known factors for cases in which the actual value of the target variable is unknown.

In this particular problem setting, the goal is to make use of a logistic regression model for the particular case of predicting whether an individual will survive the sinking of Titanic, provided we know certain characteristics of the individual, including but not limited to socioeconomic status and gender. By making use of data for which we know if an individual survived or not, we then build a model for the data for which we do not know this. The accuracy of the model can then be used to assess how good the identified relationships are.

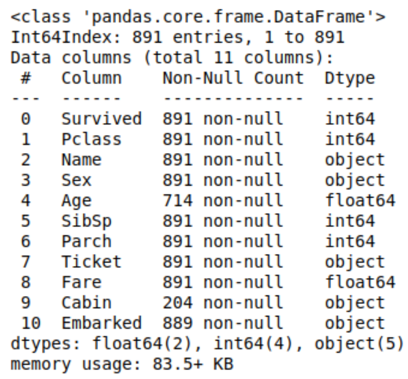
Section II gives an overview of the various techniques used for data cleaning, feature engineering and an initial exploratory analysis. A lot of insights can be gained just by making qualitative observations from the given data. Section III gives a short description of the mathematical formalism behind logistic regression. Section IV sates the various results that are obtained by applying logistic regression in this particular case. Section V gives a summary of the major conclusions drawn from the analysis.

II. Exploratory Data Analysis

In this section, we describe the process of data cleaning, feature engineering and data visualization.

A. Data Cleaning

We are given two datasets: a training dataset and a test dataset. The training dataset consists of 891 rows and 11 columns, whereas the test dataset consists of 418 rows and 10 columns. A brief overview of the training dataset is presented in Figure 1 and of the test dataset is presented in Figure 2. We see that we know the ground truth values for whether the individual survived or not in the training dataset whereas we do not know this for the test dataset. We will keep the test dataset aside and use it only after we have completed building the model.

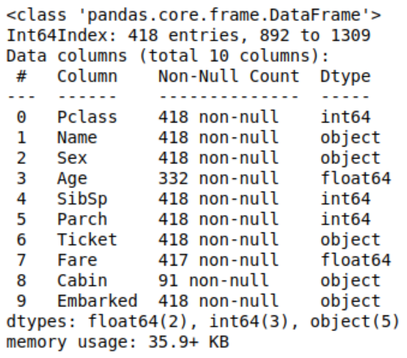


```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Name      891 non-null    object
 3   Sex       891 non-null    object
 4   Age       714 non-null    float64
 5   SibSp     891 non-null    int64
 6   Parch     891 non-null    int64
 7   Ticket    891 non-null    object
 8   Fare      891 non-null    float64
 9   Cabin     204 non-null    object
 10  Embarked  889 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 83.5+ KB
```

Fig. 1. Summary of the training dataset

Our next task is to account for the missing values in the dataset. We see that there are missing values in three columns: ”Age”, ”Cabin” and ”Embarked”. Each column is dealt with separately and the procedure is described below.

The values in the column ”Age” are plotted in a boxplot in Figure 3. From the plot, it can be seen that the distribution **is**



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 418 entries, 892 to 1309
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Pclass    418 non-null    int64
 1   Name      418 non-null    object
 2   Sex       418 non-null    object
 3   Age       332 non-null    float64
 4   SibSp     418 non-null    int64
 5   Parch     418 non-null    int64
 6   Ticket    418 non-null    object
 7   Fare      417 non-null    float64
 8   Cabin     91 non-null     object
 9   Embarked  418 non-null    object
dtypes: float64(2), int64(3), object(5)
memory usage: 35.9+ KB
```

Fig. 2. Summary of the test dataset

skewed to the right. In such scenarios, it is better to impute the missing values **using median as compared to mean.** Hence the missing values are replaced with the computed median, which happens to be 28.

We see that we only have **202 non-null values for column ”Cabin” out of the total 891, which is merely 22.67%.** Since this is so low, we drop this column for predictive modelling. This is further supported by the fact that most of the datapoints in the test set also do not have cabin values. However, we will use the available values to see if there is any correlation between cabin number and chance of survival.

Finally, for the column ”Embarked” we have only 2 missing values. Since this is just 2 datapoints, we drop these rows. After handling the missing values, we end up with a dataset which is summarized by Figure 4.

We also note that the feature ”Fare” has a missing value in the test set, but not in the training set. We perform mean imputing for this feature in the test set.

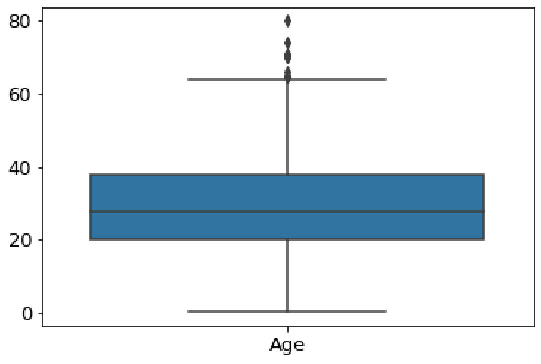
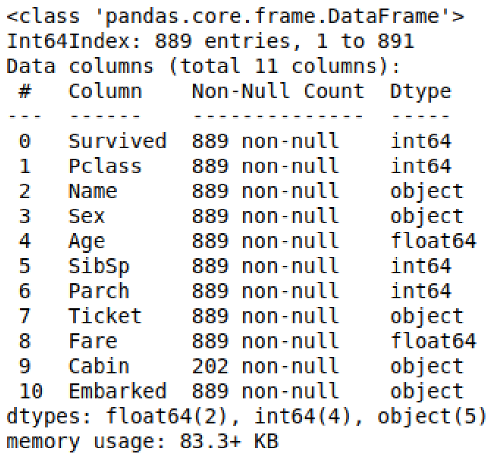


Fig. 3. Distribution of age

B. *Data Visualization/ Qualitative Analysis*

To get a qualitative picture, we make plots of relevant features against the survival rate, which is the number of individuals who survived divided by the total number of individuals.



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  889 non-null    int64
 1   Pclass    889 non-null    int64
 2   Name      889 non-null    object
 3   Sex       889 non-null    object
 4   Age       889 non-null    float64
 5   SibSp     889 non-null    int64
 6   Parch     889 non-null    int64
 7   Ticket    889 non-null    object
 8   Fare      889 non-null    float64
 9   Cabin     202 non-null    object
 10  Embarked  889 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 83.3+ KB
```

Fig. 4. Summary of the processed training dataset

First, we make a barplot of passenger class, given by the column ”Pclass”. We obtain the plot given in Figure 5. From the plot, it is evident that a higher fraction of passengers belonging to a higher passenger class survived as compared to passengers belonging to a lower passenger class. This could be **due to the fact that higher class tickets were often taken by people who were economically and socially privileged thereby giving them an advantage over the lesser privileged sections in times of such crises.** This also reflects an inherent bias present in the society.

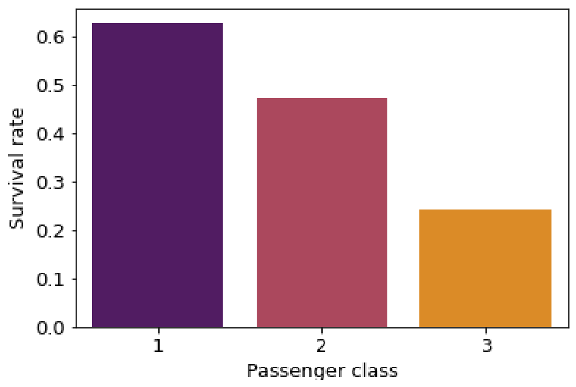


Fig. 5. Survival rate v/s Passenger class

Next, we make a barplot of gender, given by the column ”Sex”. We obtain the plot given in Figure 6. From the plot, it is evident that the **survival rate is higher for females as compared to males.** This could be because of the social norm of the time, which stipulated giving preference to women and children during times of disasters.

Next, we make a kdeplot (kernel density estimate plot) of age, given by the column ”Age”. We obtain the plot given in Figure 7. From the plot, we make two observations:

- There was a **greater chance for children to survive.** Again, this could be because priority was given to women

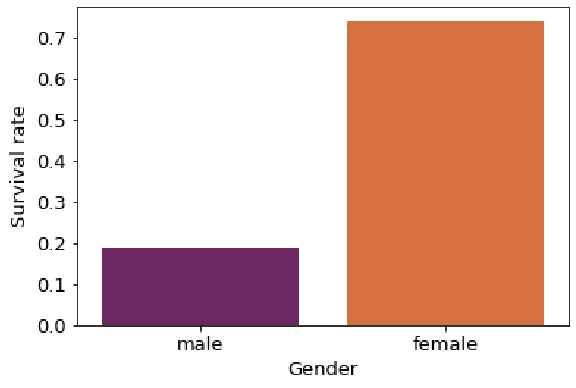


Fig. 6. Survival rate v/s Gender

and children during evacuation.

- **A greater fraction of people in the age group 20-40 did not survive** compared to other age groups; this could be because they stayed back to aid younger and older individuals during evacuation.

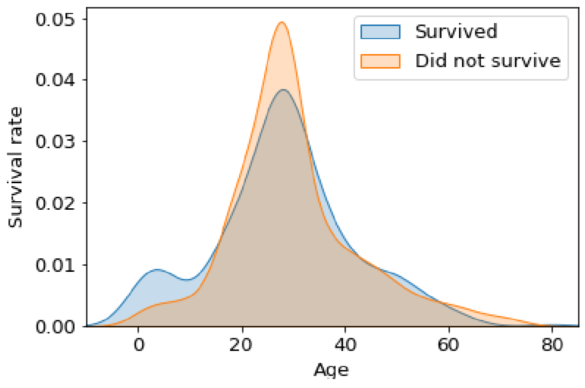


Fig. 7. Survival rate v/s Age

Next, we make a kdeplot (kernel density estimate plot) of fare, given by the column ”Fare”. We obtain the plot given in Figure 8. It can be seen that, in general, **people who paid a higher fare had a higher chance for survival.** This could again be associated with higher social and economic status and the associated privilege, as generally only the richer sections can afford to pay a higher fare.

Next, we explore if there is any relationship between cabin and survival rate. Since considering each cabin separately is impractical, we consider only the cabin group, which is given by the alphabet preceding the number. Note that we do this only for the datapoints for which the cabin information is available. Rest of the datapoints are assigned the cabin group ”U”. We obtain the plot given in Figure 9. Due to the limited data available, it is not possible to make conclusive statements regarding the impact of cabin group on survival rate. However, if more data is available, it might be possible to correlate the

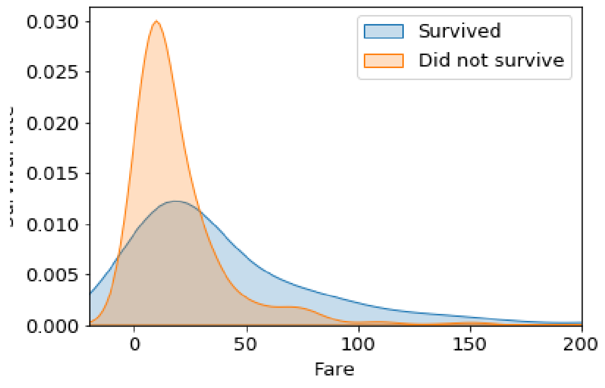


Fig. 8. Survival rate v/s Fare

cabin group with survival rate as we intuitively expect certain cabin groups to have more access to lifeboats and other such similar factors.

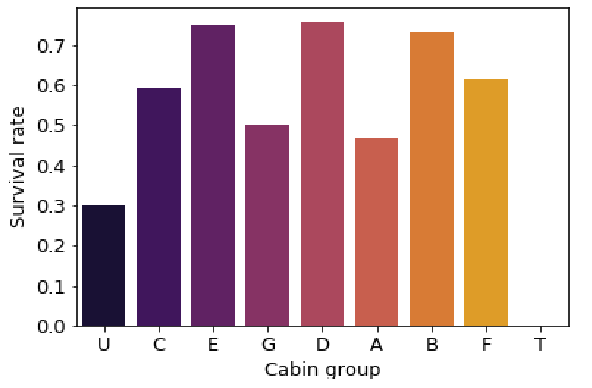


Fig. 9. Survival rate v/s Cabin group

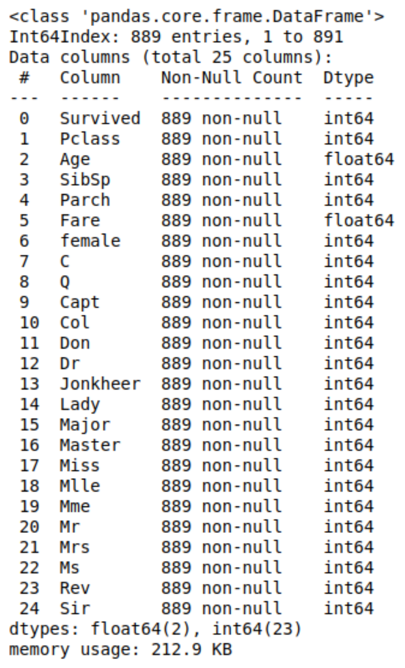
C. Feature Extraction

In this section, we look at manipulating some of the existing features so as to make them more useful.

We do not expect the name of an individual to directly have any influence on his/her survival chance. However, **the honorifics associated with the name might have an impact on survival chance.** Hence, we extract the honorifics and store them in a separate column titled ”title”. We expect this feature to capture the socioeconomic status of an individual.

The categorical features we have now are ”Pclass”, ”Sex”, ”Embarked” and ”title”. Of these, ”Pclass” which denotes the passenger class has a numbering by default; as we have seen before, higher the class of the passenger, higher is the chance of survival. Hence, we maintain the existing numerical ordering for this feature.

For the other features, we implement one-hot encoding. We cannot employ ordinal ordering as this would enforce an ordering of classes which does not naturally exist. Although in some applications doing one-hot encoding would blow up the number of features, in this case, since we only have a limited number of features, one-hot encoding is feasible. After performing one-hot encoding, we end up with the dataset summarized in Figure 10.



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 889 entries, 1 to 891
Data columns (total 25 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  889 non-null    int64
 1   Pclass    889 non-null    int64
 2   Age       889 non-null    float64
 3   SibSp     889 non-null    int64
 4   Parch     889 non-null    int64
 5   Fare      889 non-null    float64
 6   female    889 non-null    int64
 7   C         889 non-null    int64
 8   Q         889 non-null    int64
 9   Capt      889 non-null    int64
 10  Col       889 non-null    int64
 11  Don       889 non-null    int64
 12  Dr        889 non-null    int64
 13  Jonkheer  889 non-null    int64
 14  Lady      889 non-null    int64
 15  Major     889 non-null    int64
 16  Master    889 non-null    int64
 17  Miss      889 non-null    int64
 18  Mlle      889 non-null    int64
 19  Mme       889 non-null    int64
 20  Mr        889 non-null    int64
 21  Mrs       889 non-null    int64
 22  Ms        889 non-null    int64
 23  Rev       889 non-null    int64
 24  Sir       889 non-null    int64
dtypes: float64(2), int64(23)
memory usage: 212.9 KB
```

Fig. 10. Summary of dataset after feature extraction

III. Model: Logistic Regression

In this section, we will give a brief overview of the mathematical formalism behind the logistic regression model.

Logistic Regression (also called Logit Regression) is a model commonly used to estimate the probability that an instance belongs to a particular class out of K given classes. The most common version is the one with K = 2, in which case the classification is binary. If the estimated probability is greater than 50%, then the model predicts that the instance belongs to that class (called the positive class, labeled "1"), and otherwise it predicts that it does not (i.e., it belongs to the negative class, labeled "0").

Logistic Regression is a linear model for classification, which means the decision surfaces are linear functions of the input vector x. The simplest case is to model $y(\mathbf{x}) = \mathbf{w^T x} + w_0$ so that y is a real number. But for a classification problem, we seek to obtain probabilities that lie in (0,1) range. Hence, we apply a non-linear function $f()$ such that

$$y(\mathbf{x}) = f(\mathbf{w^T x} + w_0)$$

returns the posterior probabilities. Decision surfaces correspond to $y(x) =$ constant , which implies $\mathbf{w^T x} + w_0 =$ constant for one to one f, which makes the decision boundary linear. An example of such a decision boundary is given in Figure 11.

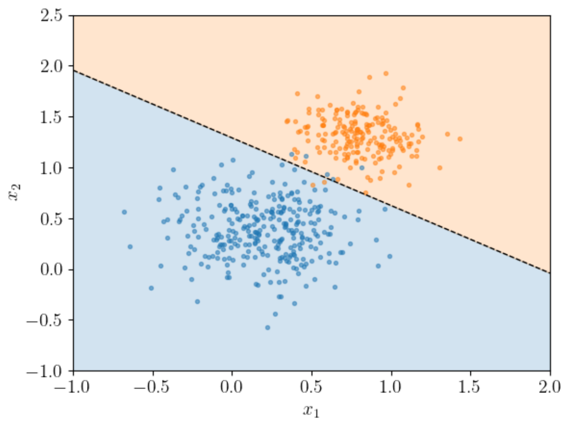


Fig. 11. Linear decision boundary of logistic regression

The non-linear function $f()$ used in logistic regression is a sigmoid function (also known as logistic function) that outputs a number between 0 and 1:

$$\sigma(u) = \frac{1}{1 + e^{-u}}$$

$$\hat{p} = \sigma(\mathbf{w^T x} + w_0)$$

The estimated probabilities can easily be converted into a binary classifier by predicting

$$\hat{y} = \begin{cases} C1, & \text{if } \hat{p} < \text{Threshold} \\ C2, & \text{otherwise} \end{cases}$$

where $C1$ and $C2$ are the two classes. The threshold is usually set to 0.5.

Parameter Estimation

The parameters can be estimated using Maximum Likelihood Estimation (MLE). Let the number of data points be M and i be an integer such that $1 \leq i \leq M$; i is used to index a datapoint in the dataset. Let y_i denote the ground truth corresponding to datapoint x_i. Then the Likelihood, given by $L(\mathbf{w})$ is:

$$\begin{aligned} L(\mathbf{w}) &= P(y_1, y_2, ..., y_m | x_1, x_2, ..., x_m) \\ &= \Pi_{i=1}^{M} P(Y_i = y_i | X_i x_i, \mathbf{w}) \\ &= \Pi_{i=1}^{M} \sigma(y_i \mathbf{w^T} x_i) \\ \log L(\mathbf{w}) &= \sum_{i=1}^{M} \log \sigma(y_i \mathbf{w^T} x_i) \end{aligned}$$

We could maximise log-likelihood or minimise negative log-likelihood to estimate the parameters. Or equivalently, minimise Empirical Logistic Loss function, $\hat{R}(\mathbf{w})$ defined as:

$$\hat{R}(\mathbf{w}) \triangleq -\log L(\mathbf{w}) = \sum_{i=1}^{M} \log\left(1 + \exp(-y_i \mathbf{w^T} x_i)\right)$$

The goal is to minimize $\hat{R}(\mathbf{w})$; however there exists no closed form solution. So we make use of Gradient Descent. The steps involved in the gradient descent algorithm for logistic regression are briefly outlined below:

- Initialize $\mathbf{w_t} = \mathbf{w_0}$ randomly.
- Repeat till convergence:

$$\begin{aligned} \mathbf{w_{t+1}} &= \mathbf{w_t} - \eta \nabla \hat{R}(\mathbf{w_t}) \\ &= \mathbf{w_t} - \eta \sum_{i=1}^{M} \sigma(-y_i \mathbf{w_t^T} x_i)(-y_i x_i) \end{aligned}$$

where η is the learning rate and t is the iteration number.

Regularised Logistic Regression

In order for the model to generalise well and to prevent over-fitting, a penalty function could be added to the loss function. We choose the $L2$ norm and hence this is termed Ridge regression. For the $L2$ norm penalty function, the empirical loss function becomes:

$$\hat{R}(\mathbf{w}) = \sum_{i=1}^{M} \log\left(1 + \exp(-y_i \mathbf{w}^{\mathbf{T}} x_i)\right) + \frac{\lambda}{2} ||\mathbf{w}||^2$$

where λ denotes the regularization parameter, which is treated as a hyper-parameter. The modified expression of $\hat{R}(\mathbf{w})$ is then used for performing gradient descent.

IV. Modelling

In this section, we discuss the application of the logistic regression model to our problem.

The logistic regression is trained on the training set using gradient descent. The training process is abstracted out through the sklearn library. The only **hyper-parameter present in the model is** λ**,** which is the **regularization parameter.** The **best value of** λ **is found out using cross-validation,** which involves breaking up the training data into multiple sets, evaluating different values of λ and choosing the most suitable value.

The predictive model for a classification problem can be evaluated using multiple metrics. A short description of the most commonly used metrics is given below:

- **Accuracy:** Accuracy is simply the **ratio of number of correct predictions to total number of predictions.** Although this seems like a very good metric intuitively, accuracy fails on classification problems with a skewed class distribution because of the intuitions developed by practitioners on datasets with an equal class distribution.
- **Precision:** Precision is the **ratio of true positives to the total positive predictions.** Precision is typically used when the cost of false positive is high. For instance, email spam detection.
- **Recall:** Precision is the **ratio of true positives to the total positive ground truths.** Recall is typically used when the cost of false negative is high. For instance, in fraud detection or sick patient detection.

TABLE I
Evaluation of Logistic Regression model

Metric	Score
Accuracy	0.831
Precision	0.791
Recall	0.759
F1 Score	0.775

- **F1-score:** F1-score is simply a **harmonic average of precision and recall.** F1 Score is typically used if we need to seek a balance between Precision and Recall and there is an uneven class distribution.

The confusion matrix for the training set is shown in Figure 12. The values for the evaluation metrics are given in Table I. From this analysis, it is clear that we can **predict the survival chance of an individual to a reasonable extent from the parameters used.** This indicates that there indeed exists some correlation between survival chance and factors like socioeconomic status and gender. Predictions were also made for the entries in the test set. However, the quality of predictions could not be evaluated as labels were not available.

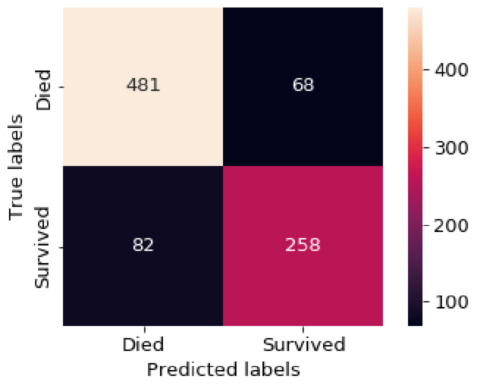


Fig. 12. Confusion matrix for the training set

ROC curve is another common tool used with binary classifiers. A Receiver Operating Characteristic curve, or ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The ROC curve for the logistic regression model is given in Figure 13.

The dotted line represents the ROC curve of a purely random classifier; a good classifier stays as far away from that line as possible (toward the top-left corner). One way to compare classifiers is to measure the area under the curve (AUC). A perfect classifier will have a ROC AUC equal to 1, whereas a purely random classifier will have a ROC AUC equal to 0.5. **The area under the ROC curve of our logistic regression classifier is 0.876.**

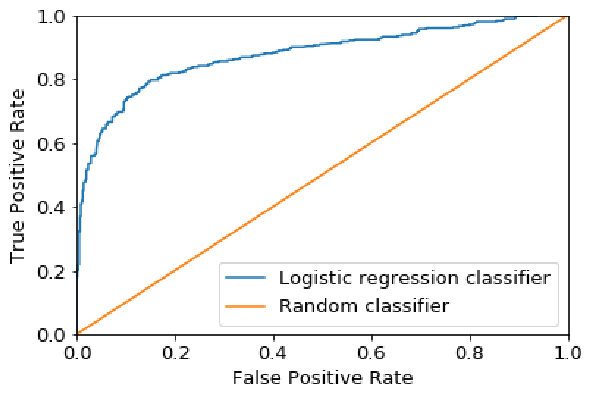


Fig. 13. ROC curve

TABLE II
EVALUATION OF KERNEL-BASED LOGISTIC REGRESSION MODEL

Metric	Score
Accuracy	0.901
Precision	0.854
Recall	0.894
F1 Score	0.873

Kernel-based logistic regression

Although the vanilla logistic regression model works reasonably well, it still suffers from that logistic regression assumes linearity in the conditional probability space. To capture non-linear relationships, it is necessary to incorporate kernels into the vanilla logistic regression model. We approximate a Gaussian kernel using Nystroem approximation.

The Nystroem method is a general method for low-rank approximations of kernels. It achieves this by essentially subsampling the data on which the kernel is evaluated. Usually, Nystroem uses the RBF/Gaussian kernel, but it can use any kernel function or a pre-computed kernel matrix. The number of samples used, which is also th,e dimensionality of the features computed, is given by a parameter.

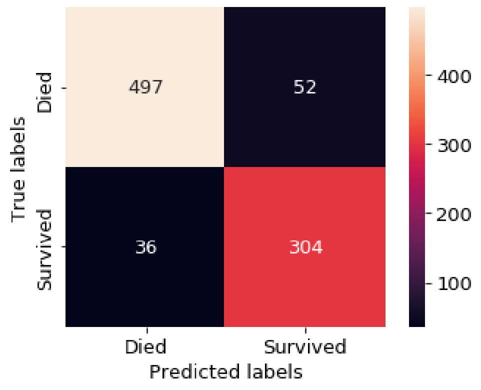


Fig. 14. Confusion matrix for the kernel-based model

The model built using this method yields the confusion matrix shown in Figure 14. The evaluation metrics for this model is presented in Table II. It can be seen that this model **clearly outperforms the vanilla logistic regression model.** This can be attributed to the use of non-linearity in the model. The ROC curve is also given in Figure 15.

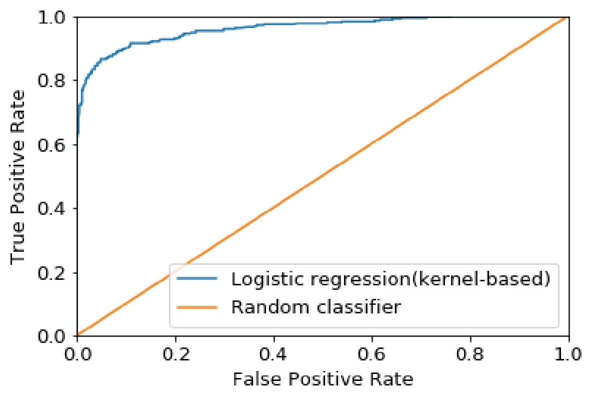


Fig. 15. ROC curve for kernel-based model

Thus, by making use of a kernel approximation, we were able to drastically enhance the performance of the logistic regression model.

V. CONCLUSIONS

From the logistic regression model, we were able to arrive at the conclusion that although luck did play a role in determining who survived and who did not, it also depended on a variety of other factors. In particular, it was found that **passengers in first class, women, children and people who paid a higher fare were all more likely to survive.** We were also able to make predictions of whether an individual will survive or not for cases where the ground truth was unknown. This analysis helps bring out biases which society as a whole may voluntarily or involuntarily hold against sections of the population. Closely reflecting on such analyses and focusing on their implications can help better everyone's lives.

VI. AVENUES FOR FURTHER RESEARCH

Although logistic regression is a powerful model, it is only a linear model. Making use of non-linear models, either by considering non-linear models or by using kernels for introducing non-linearity is an avenue worth exploring. Collecting more features (like proximity to lifeboats) could also help improve modelling.

REFERENCES

[1] A. Geron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media, Inc., 2nd ed., 2019.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[3] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[4] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference* (Stéfan van der Walt and Jarrod Millman, eds.), pp. 56 – 61, 2010.

[5] "Logistic regression." https://en.wikipedia.org/wiki/Logistic_regression.