International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 10 | Oct 2020 www.irjet.net p-ISSN: 2395-0072

Predicting the Survival of Titanic Passengers using Different Machine Learning Models

Hitesh Pujari[1], Rohit Singh Adhikari[2], Plaban Patra[3]

[1]*Hitesh Pujari, Asst. Professor, Chinmaya Degree College, Haridwar, Uttarakhand, India*
[2]*Rohit Singh Adhikari, Aspiring Data Scientist, Bengaluru, Karnataka*
[3]*Plaban Patra, Aspiring Data Scientist, Bengaluru, Karnataka*

Abstract *- The deadliest maritime disaster occurred over a century ago called sinking of "The Titanic". Many individuals of all ages and gender were present there on a fateful night, including an outsized number of men, women, and children on board where more than 1,500 people lost their lives in this deadliest marine disaster. In this research article, firstly we will explore previously unknown or hidden information by applying exploratory data analysis on the available dataset from Kaggle. We have implemented various machine learning algorithms like Logistic Regression, Decision Tree, SVM and Random Forest to predict the survival of passengers. Then, the results of applied machine learning models are compared and analyzed on an accuracy basis. We also analyzed the confusion matrix to identify the false positive and false negative of different models and accepted the best outcome. Finally the prediction made using ensemble technique through a voting classifier.*

Key Words: **Machine Learning, Feature Engineering, Logistic Regression, Decision Tree, Support Vector Machine, Random Forest, Voting Classifier.**

1. INTRODUCTION

Machine learning algorithms are applied to make a prediction for passengers survived after sinking of Titanic. Different features like name, title, age, sex, class will be used to make the predictions. Predictive analysis is a procedure that includes the utilization of computational methods to find out the important and useful patterns in large data. Using machine learning algorithms, survival is predicted on different combinations of features. The target is to perform exploratory data analytics on the available dataset and to understand the effect of every field on the survival of passengers by applying analytics between every field of the dataset with the "Survival" field. Different algorithms are compared based on their accuracy and therefore the best performing model is recommended for predictions.

2. DATA DESCRIPTION

1. survival : Survival of passenger (0 = No; 1 = Yes)
2. pclass : Passenger Class (1=First, 2=Second,3=Third)
3. name : Name
4. sex : Sex (Male/Female)
5. age : Age of passengers in years
6. sibsp : number of siblings and spouses traveling
7. parch : number of parents and children traveling
8. ticket : Ticket Number
9. fare : Passenger Fare
10. cabin : Cabin number
11. Embarked : Port of Embarkation

(C = Cherbourg, Q = Queenstown, S = Southampton)

3. PROCESS FLOW

Exploratory Data Analysis (EDA) in Python is the first step in your data analysis process developed by "John Tukey" in the 1970s. Exploratory data analysis is a data exploration technique to understand the various aspects of data sets. It is basically used to filter data from redundancies.

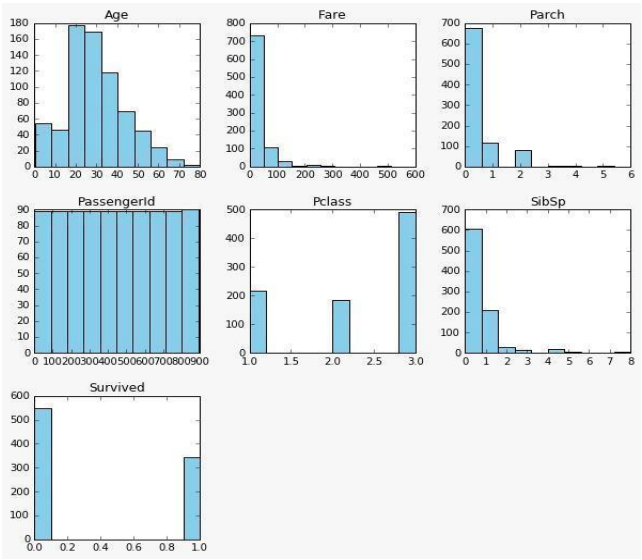
Yes – it's finally a time for Exploratory Data Analysis! There is a true saying "A picture is worth a thousand words".

Explore and Visualize Dataset

titanic.describe()

	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

Examine plots to identify any patterns or insights



4. METHODOLOGY

4.1 Feature Engineering

Feature engineering is the most important part of data analytics process. It deals with, selecting the features that are used in training and making predictions.

Let's create new features from our dataset:

- **title**: reflecting a persons title (Mr., Mrs. etc)
- **mother**: reflecting if a person is a mother or not

Create a new feature called **"Mother"** if

- the person is female
- has more than one child
- is over 18
- title is not Miss

4.2 Examine New DataFrame (Updated with 2 New Features)

```
# We added in 2 new features
titanic.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title	Mother
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	Mr	Not Mother
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	Mrs	Not Mother
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	Miss	Not Mother

4.3 Imputing Missing Values

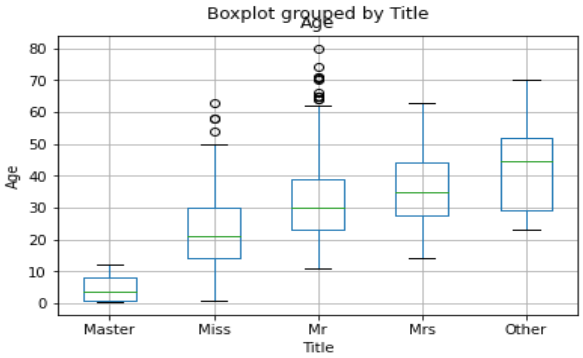
Missing Values in dataset:

- age (~20% missing)
- embarked (0.2% missing)
- cabin (~77% missing)

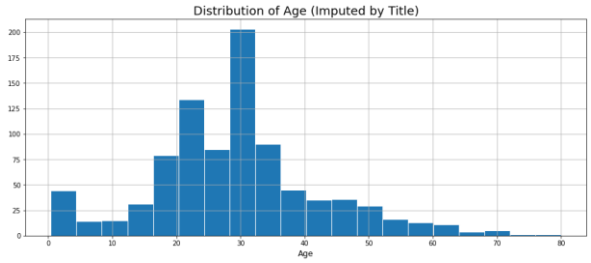
A. Data Imputation (1): Age

Data Imputation: Age by Passenger Title

Missing Age Imputation: The "Title" of each passenger tells a more realistic story on how we can impute their missing age.



Plot the Age Distribution Again



B. Data Imputation (2): Embarked

We have two missing values for Embarked. Let's impute it with the most occurring embarked station (S)

titanic.Embarked.value_counts()

S 644

C 168

Q 77

Impute missing 'Embarked' variable with the most frequent value: (S)

C. Data Imputation (3): Cabin

77% missing values from Cabin

titanic.Cabin.isnull().sum() / len(titanic)

0.7710437710437711

So, We Drop Cabin Feature

titanic.drop(columns=['Cabin'], inplace=True)

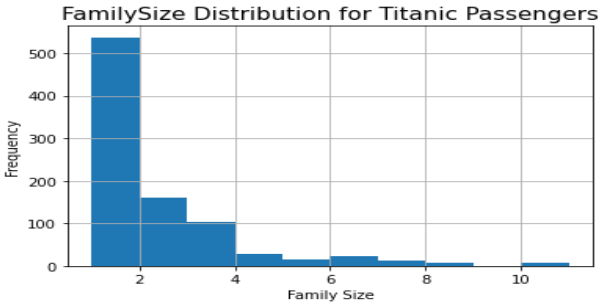
4.4 Feature Engineering Techniques

There are a lot of different types of numerical data, example:

- counts
- price
- percentages
- measurements

Numeric Features are about:

- Scale: the range of values
- Distribution: the probability of taking on a particular value



4.4.1 Dealing with Counts

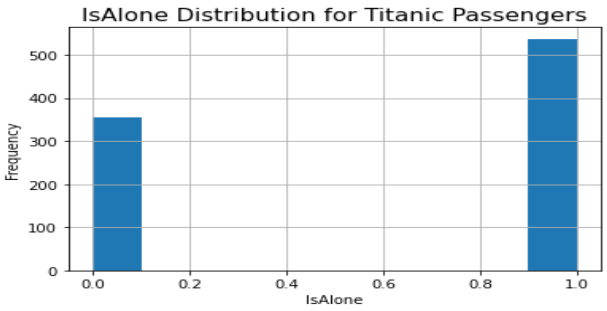
a. Binarization

Raw family size may not be a robust measurement.

With domain knowledge, we can say that passengers are broken up into two family sizes:

1. Traveling Alone
2. Traveling with Family

Create new feature:"IsAlone"



b. Quantization or Binning

We group the counts into bins, and get rid of the actual count values. Quantization maps a continuous number to a discrete one.

Binning helps solve the skewness problem.

Fixed-width Binning: With domain knowledge, we can safely bin our passengers into different age groups.

titanic.head()

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Title	Mother	FamilySize	IsAlone	AgeGroup
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S	Mr	Not Mother	2	0	adult
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C	Mrs	Not Mother	2	0	adult
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S	Miss	Not Mother	1	1	adult
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S	Mrs	Not Mother	2	0	adult
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S	Mr	Not Mother	1	1	adult

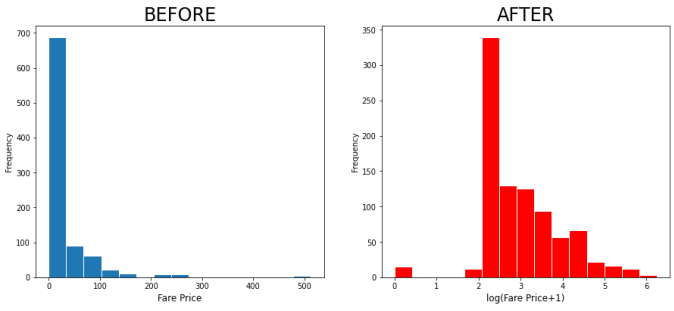
c. Power Transformations

Transformations are useful tools to apply to non-normal data.

*Power Transformations **DOES CHANGE** the distribution of data and tries to make it more "normal".*

Example: Log Transformation

The log transformation is a powerful tool for dealing with heavy right-skewed distributions.



Techniques we will use so far:

- Binning continuous variables (e.g. Age)

- Create new features out of existing variables (e.g. Title)

- Label encoding for non numeric features (e.g. Sex)

- One hot encoding for categorical features (e.g. Pclass)

4.5 Feature Scaling

Feature scaling always divides the feature by a constant and it does not change the distribution of our data. ***Let's see feature scaling in action on 'Fare Price'***

- Code
- Text

a. Standardization (Variance Scaling)

df_scale['Standardization'] = StandardScaler().fit_transform(titanic[['Fare']])

b. Categorical Feature Engineering Techniques Dummy Encoding

Use one Dummy encoding where you want each value/category of the feature to be **unique**.

One Hot Encoding fixes the problem of different categorical values have some numeric association to it.

Apply Dummy Encoding to 'AgeGroup' feature

titanic_dummyage = pd.get_dummies(titanic[['Mother','AgeGroup','Sex', 'Title', 'Embarked', 'FareGroup']])

titanic_dataframe = pd.concat([titanic, titanic_dummyage], axis=1)

titanic_dataframe.iloc[:10,10:20]

titanic_dataframe.Survived.value_counts()

0 549

1 342

Name: Survived, dtype: int64

5. BUILDING MACHINE LEARNING MODELS

Various machine learning models are implemented to validate and predict survival.

5.1 Logistic Regression

Logistic regression is one of the most popular supervised machine learning algorithms for binary classification but it can used for multiclass classification also. This is because it is a simple algorithm that performs very well on a wide range of problems.

Logistic regression is the technique which works best when a dependent variable is binary or categorical.

From sklearn.linear_model import LogisticRegression

From sklearn.metrics import accuracy_score, confusion_matrix, classification_report

#logistic_regression = LogisticRegression(class_weight='balanced')

logistic_regression = LogisticRegression(multi_class="ovr", solver="lbfgs", C=10)

logistic_regression = logistic_regression.fit(X_train, y_train)

y_pred_lr = logistic_regression.predict(X_test)

print(accuracy_score(y_pred_lr, y_test))

print(confusion_matrix(y_pred_lr, y_test))

Accuracy: 0.8181818181818182

Confusion Matrix :

[[141 37]

[23 129]]

print(classification_report(y_pred_lr, y_test))

precision recall f1-score support

0 0.86 0.79 0.82 178

1 0.78 0.85 0.81 152

accuracy 0.82 330

macro avg 0.82 0.82 0.82 330

weighted avg 0.82 0.82 0.82 330

5.2 Decision Tree

Decision tree is a supervised learning algorithm. This is generally used in problems based on classification. It is suitable for both categorical and continuous input and output variables.

Decision Tree classifier utilizes a tree structure to model relationships among the features and the potential outcomes.

From sklearn.tree import DecisionTreeClassifier

```
decision_tree = DecisionTreeClassifier(criterion='entropy')

decision_tree = decision_tree.fit(X_train, y_train)

y_pred_dt = decision_tree.predict(X_test)

print(accuracy_score(y_pred_dt, y_test))

print(confusion_matrix(y_pred_dt, y_test))
```

Accuracy: 0.8151515151515152

Confusion Matrix :

```
[[139 36]

[ 25 130]]
```

5.3 Support Vector Machine

Support Vector Machine (SVM) falls in supervised machine learning algorithm. This algorithm is used to solve both classification and regression problems.

```
from sklearn.svm import SVC

from sklearn.model_selection import cross_val_score

svm_svc = SVC(kernel='linear', gamma='auto')

svm_svc = svm_svc.fit(X_train, y_train, sample_weight=None)

y_pred_svc = svm_svc.predict(X_test)

print(accuracy_score(y_pred_svc, y_test))

print(confusion_matrix(y_pred_svc, y_test))

#scores = cross_val_score(svm_svc, X, y, cv=5)

#print(scores)

#print("Accuracy: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
```

Accuracy: 0.8121212121212121

Confusion Matrix :

```
[[139 37]

[ 25 129]]
```

5.4 Random Forest

It is supervised classification algorithm and ensemble of Decision trees is a Random Forest. The algorithm basically makes forest with large number of trees. The higher the number of trees in the forest gives the higher accuracy results.

Random Forest creates several trees, sometimes thousands, and calculates the best possible model for a given dataset. Instead of considering all features while splitting a node, Random Forest algorithm selects the best feature out of a subset of all features. This trades a higher bias for lower variance, which yields a much better model.

```
from sklearn.ensemble import RandomForestClassifier

random_forest = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini', max_depth=None, max_features='auto', max_leaf_nodes=None,

min_impurity_decrease=0.0, min_impurity_split=None,

min_samples_leaf=1, min_samples_split=2,

min_weight_fraction_leaf=0.0, n_estimators=400, n_jobs=1,

oob_score=False, random_state=None, verbose=0,

warm_start=False)

random_forest = random_forest.fit(X_train, y_train)

y_pred_rf = random_forest.predict(X_test)

print(accuracy_score(y_pred_rf, y_test))

print(confusion_matrix(y_pred_rf, y_test))
```

Accuracy: 0.8242424242424242

Confusion Matrix :

```
[[139 33]

[ 25 133]]
```

6. ENSEMBLE LEARNING

Ensemble Learning is a method of collection of several models working together on a single set. It is a technique which create multiple models and then combine them to produce improved results.

It is applied to both regression as well as classification.

Ensemble learning for classification creates multiple classifiers i.e. multiple classification models such as logistic, decision trees, KNN, SVM etc.

Ensemble Learning Techniques/Methods :

- **Bagging / Bootstrap Aggregating** : Building Multiple Models of same type from different subsamples of the training dataset.
- **Boosting** : Building Multiple Models of same type each of which learns to fix the prediction errors of a prior model in the chain.
- **Stacking :** It involves combining the predictions from multiple machine learning models.
- **Voting and Averaging :** Building multiple models of different types and simple statistics are used to combine predictions.

Voting used in classification whereas Averaging used in regression.

Voting Classifier using Sklearn

Voting is an ensemble machine learning algorithm.

We use voting ensembles when all models in the ensemble have generally the same good performance. It simply aggregates the findings of each classifier passed into Voting Classifier and predicts the output class based on the highest majority of voting.

We can train our model using diverse algorithms and then ensemble them to predict the final output. We can use a Logistic Regression, SVM Classifier, Random Forest Classifier, etc.; models are participated together and selected upon best performance by voting using the ***VotingClassifier*** Class from ***sklearn.ensemble***. The accuracy of the ***VotingClassifier*** is generally higher than the individual classifiers. It can be used for both classification or regression.

In Classification Voting Ensemble method, Predictions are made on the majority vote of participating models.

There are two approaches to the majority vote prediction for classification :

Hard Voting. It involves summing the predictions for each class label and predicting the class label with the most votes. We can say it predicts the class with the largest sum of votes from models.

Soft Voting. It involves summing the predicted probabilities (or probability-like scores) for each class label and predicting the class label with the largest probability. Similarly, it predicts the class with the largest summed probability from models.

A voting ensemble may be considered a meta-model, a model of models.

```
# implement voting of best three algorithms

from sklearn.ensemble import VotingClassifier

from sklearn.model_selection import cross_val_score

from sklearn.model_selection import cross_val_predict

voting_clf = VotingClassifier (estimators=[('lr', logistic_regression), ('rf', random_forest)], voting='soft')

voting_clf = voting_clf.fit(X_train, y_train)

y_pred_ens = voting_clf.predict(X_test)

print(accuracy_score(y_pred_ens, y_test))

print(confusion_matrix(y_pred_ens, y_test))

#voting_cv_scores = cross_val_score(voting_clf, X_train, y_train, cv=3, scoring="accuracy" )
```

Accuracy: 0.8454545454545455

Confusion:

[[144 31]

[20 135]]

7. MODEL EVALUATION

The accuracy of the model is evaluated using "confusion matrix".

A. Confusion Matrix

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. A confusion matrix is a table layout that allows to visualize the correctness and the performance of an algorithm. A confusion matrix is a method to verify how accurately the classification model works.

For a binary classification problem, we would have a 2 x 2 matrix as shown below with 4 values:

		Actual Values: Positive (1)	Actual Values: Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Positive Predictive Value: It gives the performance measure of the statistical test. It is a ratio true positive (event that makes true prediction and subject result is also true) and the sum of true positive and false positive (event that makes false prediction and subject result is also false).

Negative Predicted Value: It is the ratio of true negatives (the event which makes negative prediction and result is also false) and sum of true negative and false negative (event that makes false prediction and subject result is positive).

B. Classification Accuracy :

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

It gives the measure of percentage of correct prediction done by the model/algorithm. The best value is "1.0" and the worst value is "0.0".

Confusion Matrix		Target Positive	Target Negative		
Model	Positive	a	b	Positive Predictive Value	a/(a+b)
	Negative	c	d	Negative Predictive Value	d/(c+d)
		Sensitivity a/(a+c)	Specificity d/(b+d)	Accuracy= (a+d)/(a+b+c+d)	

SN	Model	True Positive	True Negative	False Positive	False Negative	Accuracy
1.	Logistic Regression	141	129	37	23	0.8181
2.	Decision Tree	139	130	36	25	0.8151
3.	SVM	139	129	37	25	0.8121
4.	Random Forest	139	133	33	25	0.8242

8. PREDICTION

Here we can choose any of the models to predict survival of test sample. Since we have evaluated all models by using confusion matrix we will predict by using model which has highest accuracy.

We performed prediction on dataset by using logistic regression, SVC and Random Forest. Additionally, the final prediction is made through best voting outcome of algorithms using ensemble technique. As it is very much clear from above table, and we predicted that Random Forest model has the highest accuracy

9. CONCLUSION

It would be interesting to play more with dataset and introducing more attributes which might lead to better results. Various other machine learning techniques like Naive Bayes, K-NN classification can be used to solve the problem.

10. REFERENCES

[1] Kaggle, Titanic: Machine Learning form Disaster [Online]. Available: http://www.kaggle.com/

[2] Prediction of Survivors in Titanic Dataset: A Comparitive Study using Machine Learning Algorithms, Tryambak Chatterlee, IJERMT-2017.

[3] Eric Lam, Chongxuan Tang, "Titanic Machine Learning From Disaster", LamTang-Titanic Machine Learning From Disaster, 2012.

[4] Analyzing Titanic disaster using machine learning algorithms-Computing, Communication and Automation (ICCCA), 2017 International Conference on 21 December 2017, IEEE.

[5] Dr. Neeraj Bhargava, Girja Sharma, Decision Tree Analysis on J48 Algorithm for Data Mining. Volume 3, Issue 6, June 2013.

[6] Dr.Prabha Shreeraj Nair, Analyzing Titanic Disaster using Machine Learning Algorithms, | Volume - 2 | Issue – 1

[7]Yogesh Kakde,Shefali Agrawal,Predicting Survival on Titanic by Applying Exploratory Data Analytics and Machine Learning Techniques,Volume 179 – No.44, May 2018

[8]https://towardsdatascience.com/predicting-the-survival-of-titanic-passengers-Niklas Donges

[9]https://www.analyticsvidhya.com/machine-learning

[10]Wikipedia. Logistic Regression [Online]. Available: https://en.wikipedia.org/wiki/Logistic_regression