May 2020 APEC RR20-01

Titanic
Machine Learning Study
from
Disaster

Emma Yiqin Cao[1*], Weitao Xie[1], Chunzhi Dong[1] and Jing Qiu[1]

[1]Department of Applied Economics and Statistics, University of Delaware

[*]Corresponding Author: Yiqin Cao (e-mail: caoemma@udel.edu)

APPLIED ECONOMICS & STATISTICS

APEC Research Reports

Department of Applied Economics and Statistics • College of Agriculture and Natural Resources • University of Delaware

ABSTRACT

Titanic

Machine learning Study From Disaster

Keywords: Machine Learning, Titanic, Survival rate, Prediction accuracy

Machine learning plays an important role in the data science field nowadays. They can be used for classification problems. In this project, we are interested in understanding what kinds of people were more likely to survive the sinking of Titanic using different machine learning methods. Different predictors of passenger information were provided, and the survival chance of different passengers was predicted based on their covariates using 5 different machine learning methods including Conventional Logistic Regression, Random Forest, K-Nearest Neighbor, Support Vector Machine and Gradient Boosting. Grid Search Cross-validation was used for calibrating the prediction accuracy of different methods. The SVM model performs the best for our data with nine predictors and the prediction accuracy is about 83%. The Random Forest model performs the best for our data with six predictors and the prediction accuracy is also about 83%. We used Python for the whole analysis including cleaning the data, visualization, validation, and modeling.

Acknowledgements

We thank Dr.Jing Qiu and Dr.Thomas Ilvento for their research assistance.

For additional information on this research course, contact:

Emma Yiqin Cao

E-mail: caoemma@udel.edu

Department of Applied Economics and Statistics
Townsend Hall, Newark, DE 19716

Suggested Citation for APEC Research Reports

Cao.Y., Xie.W., Dong.C., and J.Qiu. 2020. "Titanic Machine learning Study From Disaster" *Applied Economics & Statistics Research Report*, University of Delaware, RR20-01.

1 Introduction

The sinking of Titanic was one of the most tremendous marine accidents during human history, and because of the lack of lifeboats, 1502 out of 2224 passengers and crew were dead. Even though there are some factors of luck during survival, it seems some groups of people were more likely to survive than others. In order to explore useful messages in this tragedy, we applied machine learning methods to build different predictive models to find out what factors of people make the most effect on survival and also compared models to get a higher prediction accuracy.

The titanic dataset is a suitable dataset to get started with machine learning exploration with 10-15 predictors including numerical and categorical variables, and 891 data points which are enough for building machine learning models for beginners to study. Also, it is a meaningful dataset to interpret as real-world data with different aspects from passengers.

We chose conventional logistic regression as the first model to build as it is the best model to interpret. We chose a random forest model for the classification purpose in this project to avoid overfitting problems in the decision tree. We also used KNN for this classification project because it is one of the simplest classification algorithms and it is one of the most used learning algorithms. SVM works relatively well when there is a clear margin of separation between classes and because we have a binary variable with two clear classes, SVM is used in our project too. Gradient boosting is a popular boosting model, learns from the previous mistakes, and also as another ensemble learning besides a random forest, gives predictions with less error. Thus we added GBM as a boosting method. Grid Search CV is used for all model building processes because it determines the optimal values for a given model.

2 Data Preparation

2.1 Overview of the Titanic dataset

The titanic dataset is downloadable from Titanic machine learning competition on Kaggle website (https://www.kaggle.com/c/titanic/data). There are nine predictors including pclass(ticket class), sex, embarked (port of embarkation) as categorical variables and age, sibsp (number of siblings and spouses), parch (number of parents and children), ticket (number), fare (passenger fare), cabin (cabin number) as numeric variables. The outcome variable is “survival”, which is a binary variable with 0 (not survived) and 1 (survived). The original dataset was split into two groups,the training dataset (891 observations), and the testing dataset (419 observations). The 419 data points in the test dataset was not used in this project for model building since it didn’t have the survival outcome variable and is being used by Kaggle to test the accuracy rate from different teams. However, we used the testing data in our data preprocess for missing data imputation and make prediction for Kaggle competition. In our model building process, we split the

original training dataset into our new training dataset with 623 data points and testing dataset with 263 data points to validate the prediction accuracy of our model.

2.2 Data Preprocessing

2.2.1 Missing Data Imputation

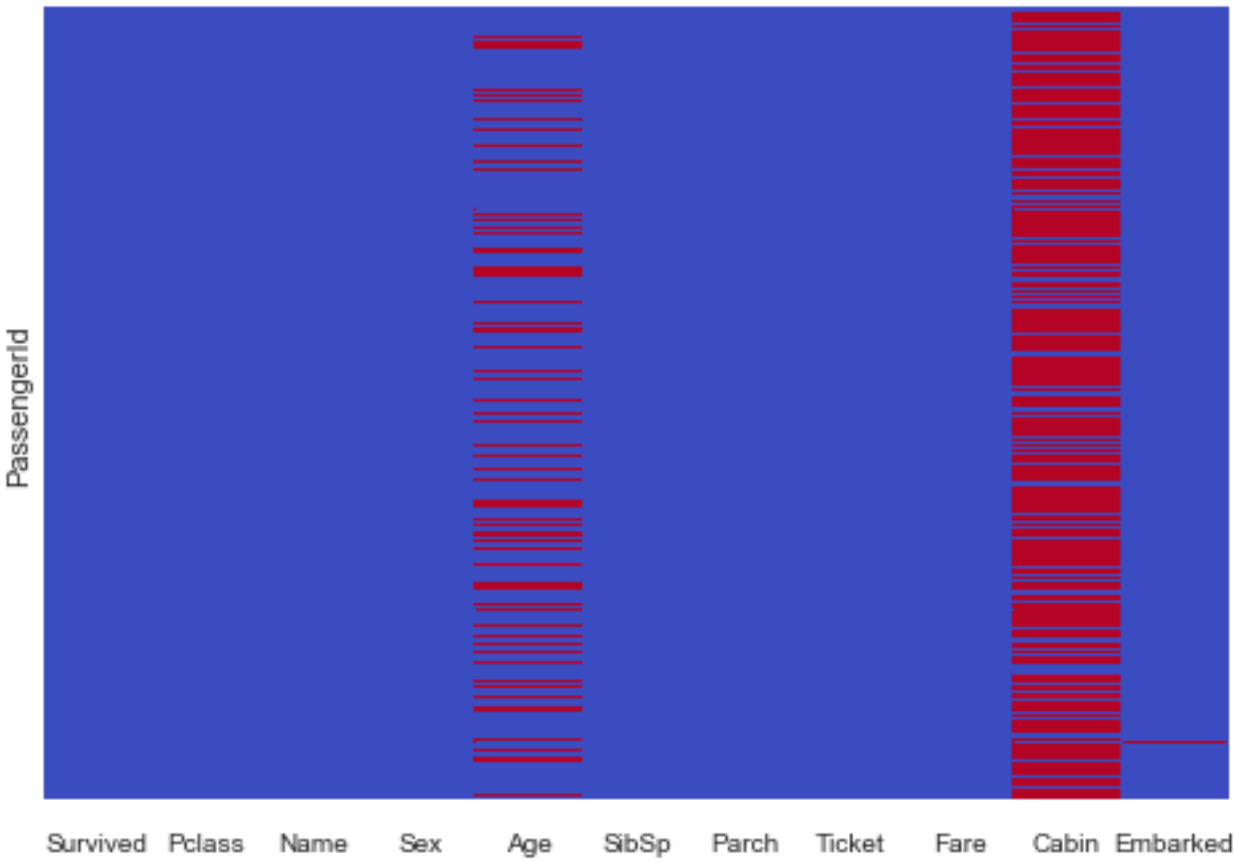


Figure 1. heatmap of missing data

First of all, we checked if there are any missing data points. The blue part in Figure 1 represents observed data points and the red part represents missing values. We found out there are missing values in predictors Cabin, Age, and Embarked in the training dataset, and Cabin, Age, and Fare in the test dataset. There are 687 missing values in Cabin in the training dataset and 327 missing datapoints in Cabin in the test dataset which occupies nearly 80 percent of the total data points. Thus the Cabin predictor is useless with so many missing values and we excluded it from model building in this project. Then we began to deal with the two missing points of Embarked in the training dataset. The two data points with missing Embarked value are as follows in table 1:

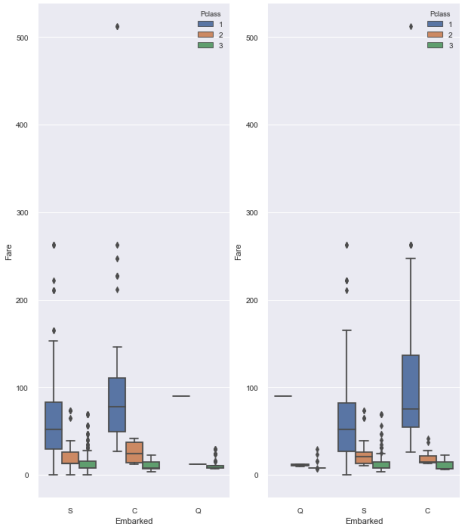
Table 1. two missing values of Embarked in the training dataset

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
PassengerId											
62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	B28	NaN
830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	B28	NaN

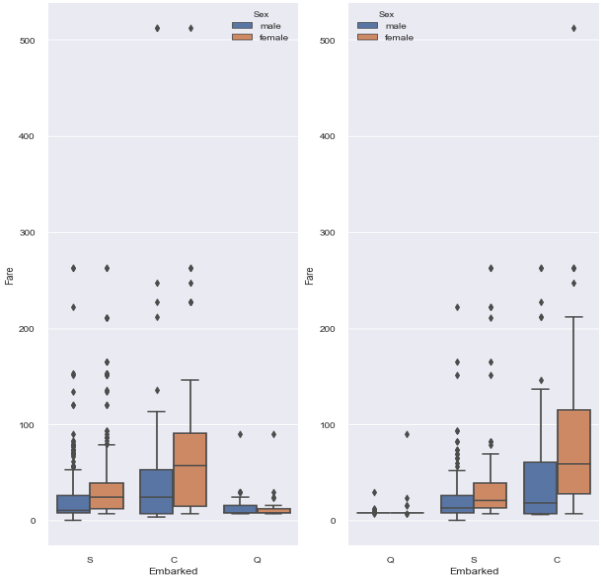
After observing the two missing data in the training set , we find that they share some common features: They were women that both came from the Pclass 1 and survived after this disaster. What's more, their tickets were the same and lived in the same Cabin. This informs us that maybe we should focus on those people who have the same conditions and got the most likely values for the Emarked. The two missing data might share these conditions: survived in disaster, female with the same fare lived in the same cabin at Pclass. We plot the box plot of passengers who lived in Pclass 1 and paid 80 for their tickers((Figure 2). Both training and testing data show that it's likely that these two missing data may belong to Embarked C. Also if we plot passengers who are female and paid 80 for their tickers, we will also get the same result, which reinforces our thoughts.

Figure 2.

(a) Pclass, Embarked and Fare



(b) Sex, Embarked and Fare



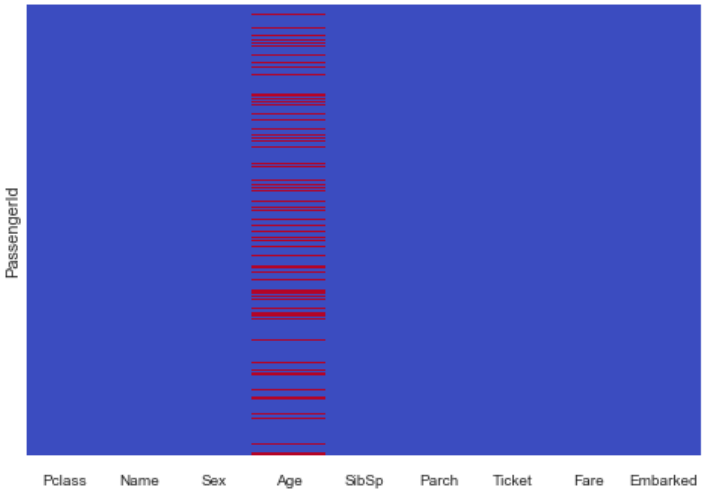
As a result, we filled the missing value of the two passengers in Embarked to "C".

Next, we dealt with the missing value of Fare in the test dataset. We continue the process as before and find out the same features of missing data. This time is much easier, we find out the fare of male living in

Pclass 3 embark from S, average the fare and fill the missing data. Then, we filled the missing value of passengers with fare equal to 12.

Finally, we began to work with the missing data in Age. From the plot (Figure 3), we can see that there are about one-third of missing data in the variable Age, and it's impossible to drop all of them. One way to replace is to find out the most related variable, which means the highest correlation one.

Figure 3. missing value heat map of the training dataset after imputation of Embarked



In order to utilize the high efficiency of the given predictors to get the most accurate assumption of the missing values, we used feature engineering to create some variables based on existing variables to achieve our goal of missing data imputation.

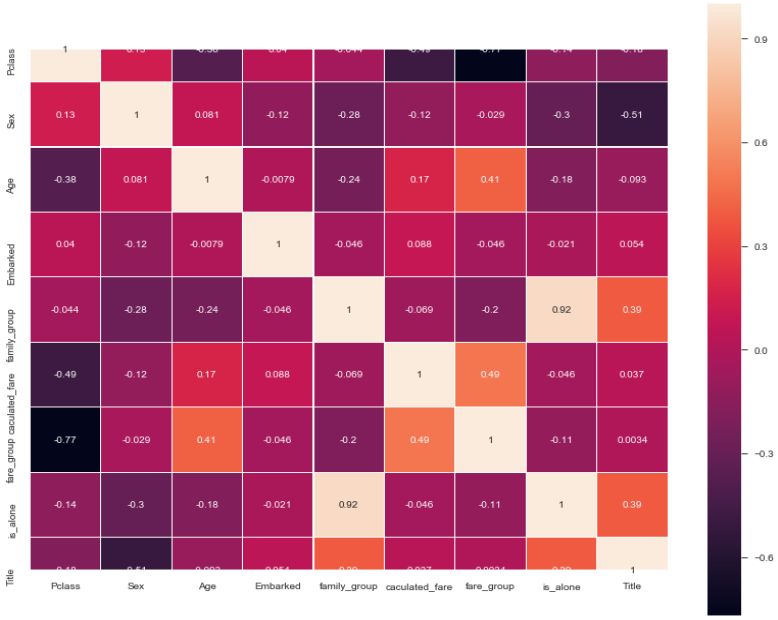
2.2.2 Feature Engineering

Firstly, we created a variable named "Family size" by summing "sibsp", "parch" and one (the passenger self) which explains the total family size of the passenger. Then we used the fare variable to divide the family size and gained the average fare price as a new variable named "calculated_fare" for each passenger, and convert the average price into five "fare_group". Next, we divided the family size into three groups and converted the variable into "family_group" as a categorical variable, and created an "is_alone" variable to indicate if the passenger is alone or not.

Lastly, we would like to extract some useful information from the name variable which could not play an important role in building models as itself. There are different kinds of titles in names which could be divided into different groups. The majority of titles are Mr., Miss., Mrs., and Master. The rest of the titles are Dr. and nobility titles (Jonkheer, colonel,etc.) of small numbers, thus we combined them into one group and used it with the other four groups to create the "Title" variable with five levels.

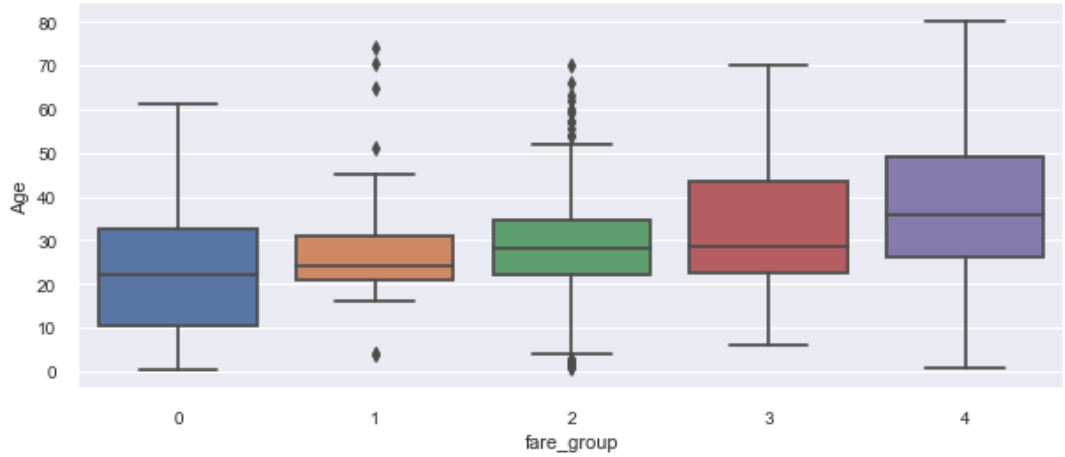
After feature engineering, we plot the heatmap to look for the highest correlation variable with “Age”. The heatmap (Figure 4) below shows the correlations between each variable, the highest one with Age is fare_group with 0.41. Fare_group shows a range of fares belonging to a certain group. Then we can use the average Age of a specified Fare_group to replace the missing data in Age.

Figure 4. a heatmap of predictors



From the box plot of Fare_group (Figure 5), we applied those means of each group to replace the missing values in the Age variable.

Figure 5. box plot of age in each Fare_group



After dealing with the missing values, we dropped some duplicated variables and there are nine predictors last, including Pclass, Sex, Embarked, Family_group, Fare_group, is_alone, and Title as categorical variables and Age with Calculated_fare as numerical variables.

3 Exploratory Data Analysis

3.1 Outcome variable:

We drew the histograms of the outcome variable and all the predictors. See Figures A.1-8 in the appendix. The histogram (figure A.1) of the survival or not comparison based on the training dataset (0 is not survived and 1 is survived) shows that the ratio of 0/1 is about 1.5 which is not an imbalanced dataset.

3.2 Pclass

Based on Figure A.2., we can clearly see that the first class is more likely to survive, and people from the third class are less likely to survive.

3.3 Sex

In figure A.3, 0, the blue bar presents the female and it shows that females are more likely to survive.

3.4 Embarked

In the figure A.4, 0 (blue bar) represents Southampton, 1 (brown bar) represents Cherbourg, 2 (green bar) represents Queenstown. There are more people dead compared to people who survived generally, however, from figure A.4 (b), we can see that people who embarked from Cherbourg are more likely to survive.

3.5 Title

In figure A.5, 0 (blue bar) is Mr., 1 (brown bar) is Miss, 2 (green bar) represents Mrs., 3 (red bar) represents Master, 4 (purple bar) is Dr & nobility titles. And we can draw the conclusion that compared to Mr. and Dr. with nobility titles, Mrs., Miss, and Master (boys younger than 18 years old) are more likely to survive. The conclusion gave us that females and children are more likely to survive.

3.6 Family_group

In figure A.6, 0 (blue bar) represents people with 0 or 1family member, 1 (brown bar) represents people with 2-4 family members, 2 (green bar) represents people with 4 members or more. From figure A.6 (b), we can see that people with 2-4 family members are more likely to survive compared to other groups.

3.7 Is_alone

From figure A.7, we can see that the not alone group (brown bar) has a higher survival rate than is_alone group. People who are alone are less likely to survive.

3.8 Age

The blue line in Figure A.8 represents the age distribution of not survived, and the brown line represents the age distribution of survived. We can see from the distribution that compared to the whole age group distribution, the center of the not survived distribution is more closely to the left. Which shows there are more young adults dead, and there is a small peak of survived from 0-10 shows more children or babies survived.

3.9 Conclusion

From the exploratory data analysis, we concluded that in general there are more people dead than survived. The first class is more likely to survive and females are more likely to survive. People who embarked from Cherbourg are more likely to survive. Females and children are more likely to survive than males and nobility titles. People with 2-4 family members are more likely to survive compared to other family groups People are not alone are more likely to survive and children are more likely to survive.

4 Exploratory Data Analysis

4.1 Introduction

Since our goal is to predict whether passengers could survive from the Titanic disaster, we use logistic regression, gradient boosting, support vector machine, k-nearest neighbors and random forests to build models. And we use cross validation and gridsearch to do the optimization. We will build our models with sklearn in Python 3.7. Scikit-learn(0.22.1) is a free machine learning package for Python, which provides lots of model including Clustering, Classification, Regression, Dimensionality reduction and Model selection. We use built-in methods fit() and predict() to train our model with data (Figure A.14). Our models are built like this example, but we use more methods to optimize model's performance.[4]

4.2 Logistic Regression

Regression analysis is a form of predictive modeling technique that investigates the relationship between a dependent (target) and independent variable (s) (predictor).

$$log\frac{p(x)}{1-p(x)} = \beta_0 + x\beta$$

Where, p=P(Y=1), Y is the response variable, x represents predictors, beta is the regression coefficient. In this case, whether people could survive is the response variable. And we consider those values larger than 0.5 as alive, otherwise as dead.

There are two hyper-parameters (C, penalty) for logistic regression in sklearn package (hyper-parameters are different from those we always use) (Figure A.15)

C:

- the regularization parameter, the inverse of regularization strength. Basically, this parameter tells the model how much you want to avoid being wrong.
- a lower C value allows for more error, which translates to higher bias.

Penalty: L1 & L2 regularization, also known as Lasso & Ridge

4.3 Gradient Boosting

It produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion as other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function. [5]

Gradient boosting is a type of machine learning boosting. It relies on the intuition that the best possible next model, when combined with previous models, minimizes the overall prediction error. The key idea is to set the target outcomes for this next model in order to minimize the error.

There are six hyper-parameters (n_estimators, learning_rate, max_depth, min_samples_leaf, max_features, n_estimators) for Gradient Boosting

- Learning_rate: shrinks the contribution of each tree. (Figure A.19)

4.4 Support Vector Machine

In machine learning, support vector machines (SVMs, also support-vector networks[1]) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space,

mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

There are three hyper-parameters (kernel, C, gamma) for SVM in sklearn package

Kernel

- Specifies which kernel should be used, “linear”, “poly”, and “rbf”.
- Rbf, or the radial basis function kernel, uses the distance between the input and some fixed point (either the origin or some of fixed point c) to make a classification assumption.
- Our dataset has lots of categorical data, it’s better to use the “rbf” kernel

Gamma:

- Defines how far the influence of a single training example reaches
- Low values meaning ‘far’ and high values meaning ‘close’ (Figure A.16)

4.5 KNN

KNN-K Nearest Neighbors is to decide how to classify a given observation x based on the class membership of the k nearest neighbors of x. It does this by computing the conditional probability of membership in class J as the fraction of the k nearest neighbors that belong to class J. x is then classified into the class that has the largest conditional probability associated with x. It’s a useful tool that can be assigned weights to the contributions of the neighbors so that the nearer neighbors contribute more to the average than the more distant ones.

There are two hyper-parameters (n_neighbors, weights) for KNN in sklearn package

- n_neighbors: the number of neighbors to use.
- Weights: function that weights the data when making a prediction. “Uniform” is an equal weighted function, while “distance” weights the points by the inverse of their distance. (Figure A.17)

4.6 Random Forest

Random Forests are an ensemble learning method for classification, regression, and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees’ habit of overfitting to their training set. Which means random forests are constructed by many binary trees that contain more accurate information. Then we will vote a class for each binary tree and combine the results of each binary tree to get a final class. Here I have a picture to help understand how random forests work. (Figure A.9)

There are six hyper-parameters (bootstrap, max_depth, max_features, min_samples_leaf, min_samples_split, n_estimators) for Random forest in sklearn package

- bootstrap: method for sampling data points without replacement
- max_depth: The maximum depth of the tree
- max_features:
    - the number of features to consider when looking for the best split
    - Some columns may so much better that every single tree we built always started with that column. We need to avoid this and take a different subset of columns.
- min_samples_leaf: stop training the tree further when a leaf node has the minimum number
- min_samples_split: The minimum number of samples required to split an internal node

n_estimators: The number of trees in the forest (Figure A.18)

4.7 Evaluation metrics

Evaluation metrics are designed to measure the quality of machine learning models. There are several types of metrics available for models, including accuracy, confusion matrix, recall and precision, F1 score. We mainly focus on accuracy.

4.8 GridSearchCv

GridSearchCv is a process of performing the hyperparameter in order to find the optimal values of a given model. The performance of models is entirely based on the hyperparameter we choose. Here is how this approach works: We first define a list of hyperparameters, the algorithm will automatically combine them into different groups, then iterate all those groups to find out the best performance of the models.[6]

For example, when we define hyperparameters as below, we have 288 combinations for our models. GridSearchCv will automate finding out the best one. (figure A.10)

4.9 Overcoming overfitting problem

Overfitting is a very general problem in machine learning. Overfitting means the model we trained has trained the data too well and the accuracy of the model is overestimated. (Figure A.11) The model has high accuracy with current data and it will decrease dramatically when using new data, meaning that the model can't generalize and infer from other data. This is because we keep using complex models to fit the pattern of data. One way to overcome this is by using training/testing sets and cross-validation.

The training and testing set approach simply divides the data set into training and testing sets. Using the training set to train the model and testing set to evaluate the performance of the model. In our case, we use

80% of data as a training set and 20% as a testing set. However, there is a shortage of this approach that we can't guarantee the split we divided is random. It's still possible that some important features are all in one set and cause overfitting. And also the model will rely too much on the training data, losing the ability of generalization.

One way to avoid this is by applying Cross-Validation, which is a special case of training and testing approach. We divide the data into K fold (Figure A.12), each time we select one fold as the test set and the remaining one as a train set, then repeat K time, and average all the accuracy as the final accuracy.

4.10 Model fitting

Scikit-learn is a python package which contains lots of models, we will import those models we mention above to fit our data. Take the random forest model as an example (Figure A.13) , we create the random forest instance first, then define the hyperparameter for the GridSearchCv approach and iterate these combinations with 10 times cross-validation. The algorithm will provide the best accuracy we define and the best hyperparameter we want.

After building models in the same steps, we got the accuracy for both training and testing data of all models. We found out the SVM achieved the best accuracy for both training and testing data.

5 Results

5.1 Model Accuracy Comparison

Here is a table of all model results comparison

Table 2: Model comparison of accuracy using GridSearchCV

Model	Accuracy for training data	Accuracy for testing data
Logistic ('C':0.1, 'penalty': 'l1')	0.81540	0.81343
Gradient Boosting ('learning_rate': 0.1, 'loss': 'deviance', 'max_depth': 4, 'max_features': 0.3, 'min_samples_leaf': 100, 'n_estimators': 300)	0.81059	0.79104
SVM ('C': 50, 'gamma': 0.01, 'kernel': 'rbf')	0.83146	0.82462

KNN ('algorithm': 'auto', 'n_neighbors': 4, 'weights': 'uniform')	0.82825	0.80970
Random Forest ('bootstrap': False, 'max_depth': 10, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_split': 10, 'n_estimators': 400)	0.83146	0.80597

5.2 Feature Importance

We would like to know if the feature importance will influence accuracy, so we get the feature importances of all the variables. Then remove those which are less than 0.05 and apply them back to models we choose. We find out the all the accuracy are decreasing, except the Random Forest. Random Forest achieve the highest accuracy.

Figure 6. Feature importance of all variables

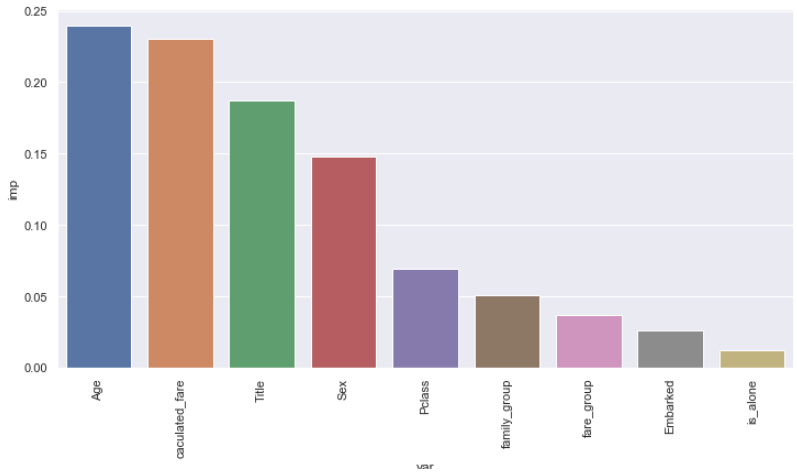
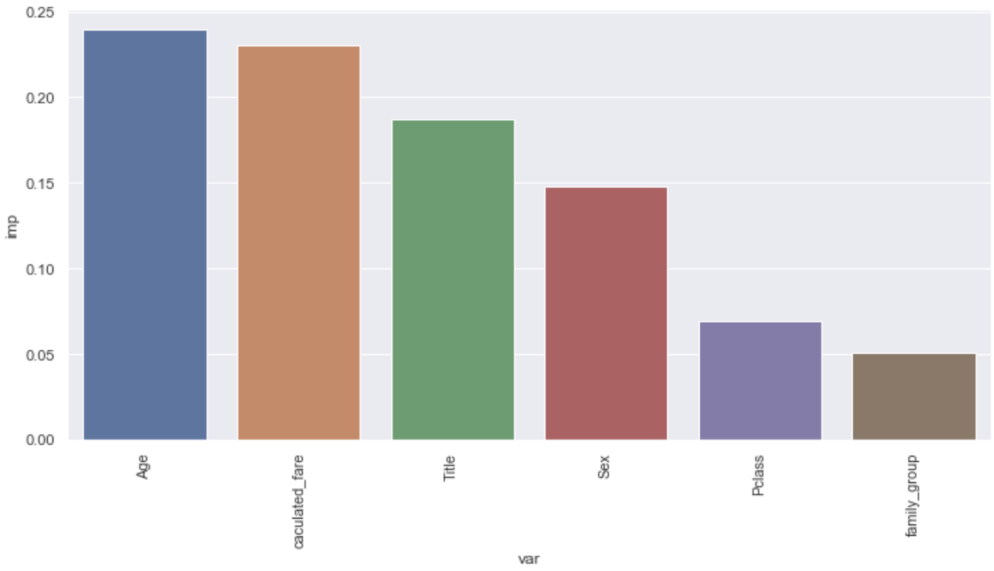


Figure 7: Feature importance with variables larger than 0.05



5.3 Model building after feature selection

We built the same five models using GridSearchCV and chose the best parameters as shown in Table 3 and got the accuracy. After comparison, we found out the accuracy of these models are decreasing except for the random forest.

Table 3: Model comparison of accuracy after feature selection using GridSearchCV

Model	Accuracy for training data	Accuracy for testing data
Logistic ('C': 1, 'penalty': 'l2')	0.80901	0.79477
Gradient Boosting ('learning_rate': 0.05, 'loss': 'deviance', 'max_depth': 4, 'max_features': 0.3, 'min_samples_leaf': 100, 'n_estimators': 100)	0.80419	0.80970
SVM ('C': 1000, 'gamma': 0.001, 'kernel': 'rbf')	0.81223	0.82089
KNN ('algorithm': 'auto', 'n_neighbors': 6, 'weights': 'distance')	0.74185	0.74253
Random Forest ('bootstrap': True, 'max_depth': 10,	0.83469	0.83955

'max_features': 'auto', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 200)		

6 Conclusion

6.1 Feature summary

From the feature importance (Figure 6) we can conclude that in the random forest model, both age and calculated fare are important factors to influence the survival variable. Title and sex are the second important variables, and are followed by Pclass and family group. We also used a logistic regression model to interpret the feature importance, and the result is a little bit different from the random forest. Sex makes the most effect on the survival variable and followed by Title, family_group, Age, and is_alone (Figure 8).

Figure 8: Feature importance in logistic regression

```
                  Overall
Pclass          2.6419378
Sex             8.8768070
Age             3.3111153
Embarked        0.9759918
family_group    3.6796976
caculated_fare 0.4291419
fare_group      2.1223125
is_alone        3.1575097
Title           4.2350102
```

In conclusion, Age, Title, Sex, and Family groups affect the survival rate mostly among nine predictors. From our exploratory data analysis, we conclude that in the Age variable, young children are more likely to survive due to the protection of Titanic. And young adults are less likely to survive. In the Title variable, male and nobility titles are less likely to survive compared to female and children titles: Mrs., Miss, and Master. Females are more likely to survive, and people who have a family group of 2-4 are more likely to survive compared to people with other family groups.

6.2 Overall summary

During this research process, we conducted several machine learning models and found out what factors make the most effect on survival or not on Titanic. Age, Title, Sex, and Family Group are the most important factors among all aspects of people. Among them, the Title and Family group are predictors we created from original variables, which means feature engineering plays an important role in machine

learning methods to get useful information. From both nine predictors and six predictors model building results, we can see that SVM and random forest performs better compared with other machine learning methods. Random Forest performs better after the feature selection using the random forest model. By comparing model performance, we learned that there is no perfect model, as statistics graduate students, summarizing and analyzing the characteristics from the dataset, and then fitting the corresponding models would be the procedure we should make a great effort on.

References

[1] VMware Hands-on Labs. (2020) “Launch Your Machine Learning Workloads in Minutes on VMware vSphere” [Online]. Available: https://docs.hol.vmware.com/HOL-2020/hol-2048-01-emt_html_en/

[2] Computer Science Engineering.(2017) “3.4.1 The Problem of Overfitting by Andrew Ng” Youtube. https://www.youtube.com/watch?v=OSd30QGMl88

[3] Krishni.(2018) “K-Fold Cross Validation” Medium [Online]. Available: https://medium.com/datadriveninvestor/k-fold-cross-validation-6b8518070833

[4] Wes McKinney and the Pandas Development Team.(2020) “pandas: powerful Python data analysis toolkit” Release 1.0.3 [Online]. Available: https://pandas.pydata.org/pandas-docs/stable/pandas.pdf

[5] Scott Hartshorn.(2017) “ Machine learning with Boosting: A Beginner’s Guide” Kindle Edition published by Amazon

[6] Stefan Janse. (2018) “Hands-On Machine Learning for Algorithmic Trading” Packt Publishing

Appendix

Code Github link:

https://github.com/fallenleaves404/668project

Figure A.1. histogram of Survived or not

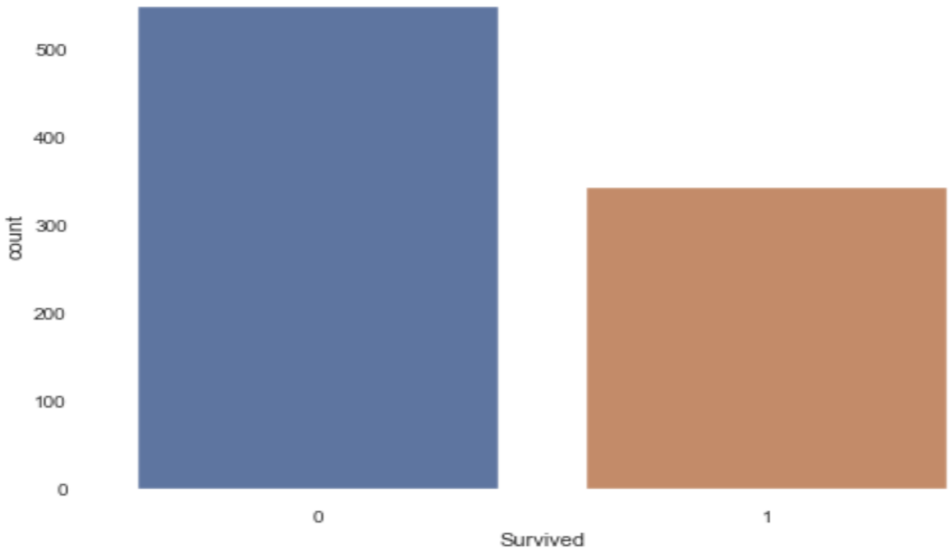


Figure A.2

(a) histogram of three classes **(b) histogram of three classes between survived or not**

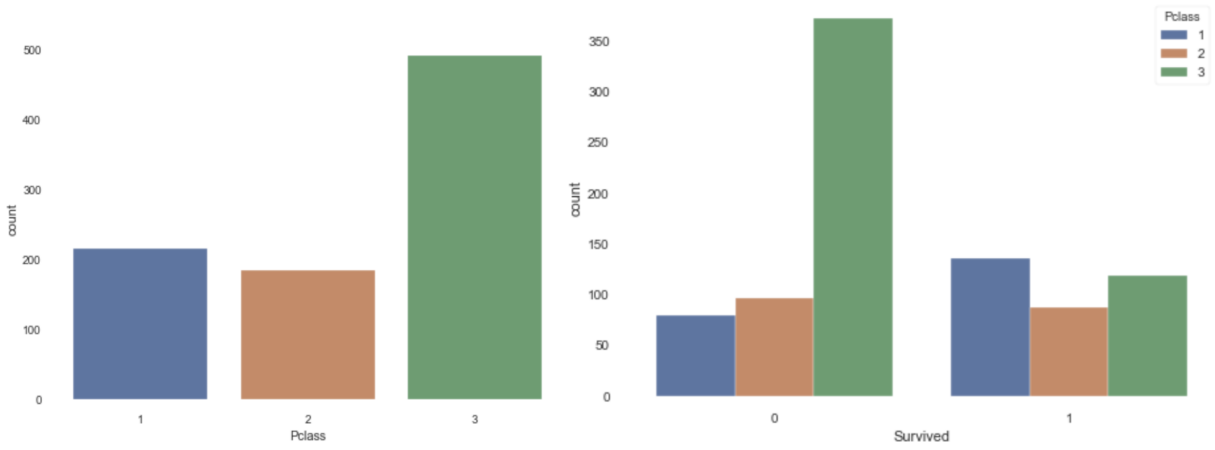


Figure A.3

(a) histogram of sex **(b) histogram of male and female between survived or not**

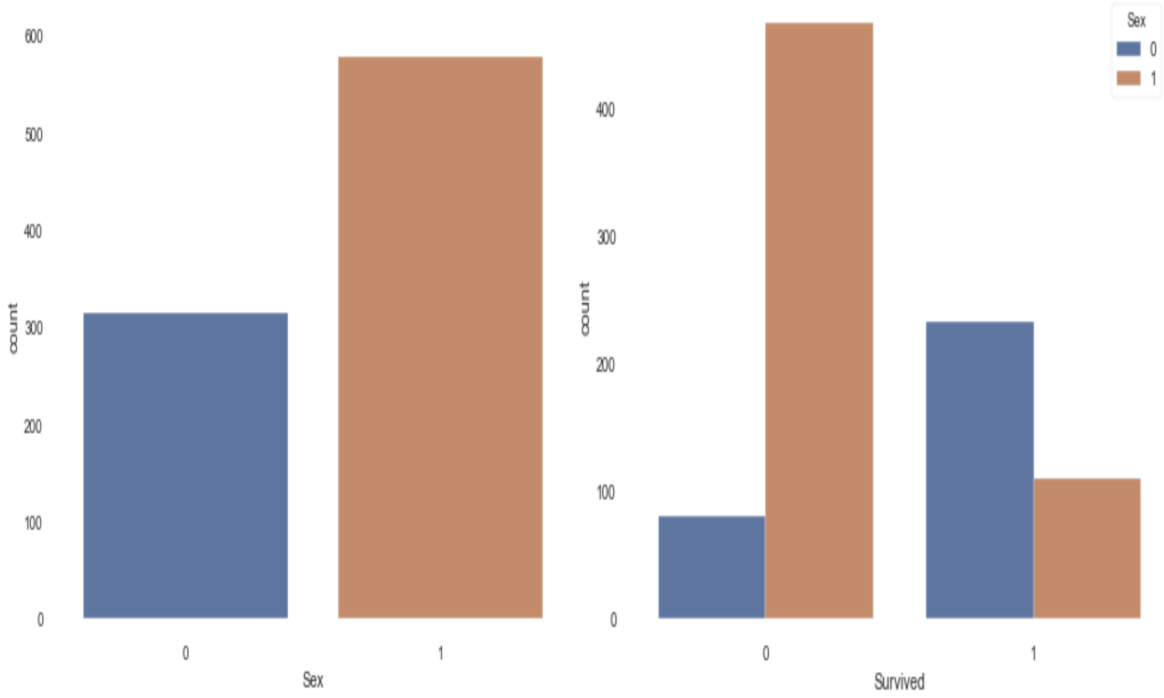


Figure A.4

(a) histogram of Embarked **(b) histogram of Embarked between survived or not**

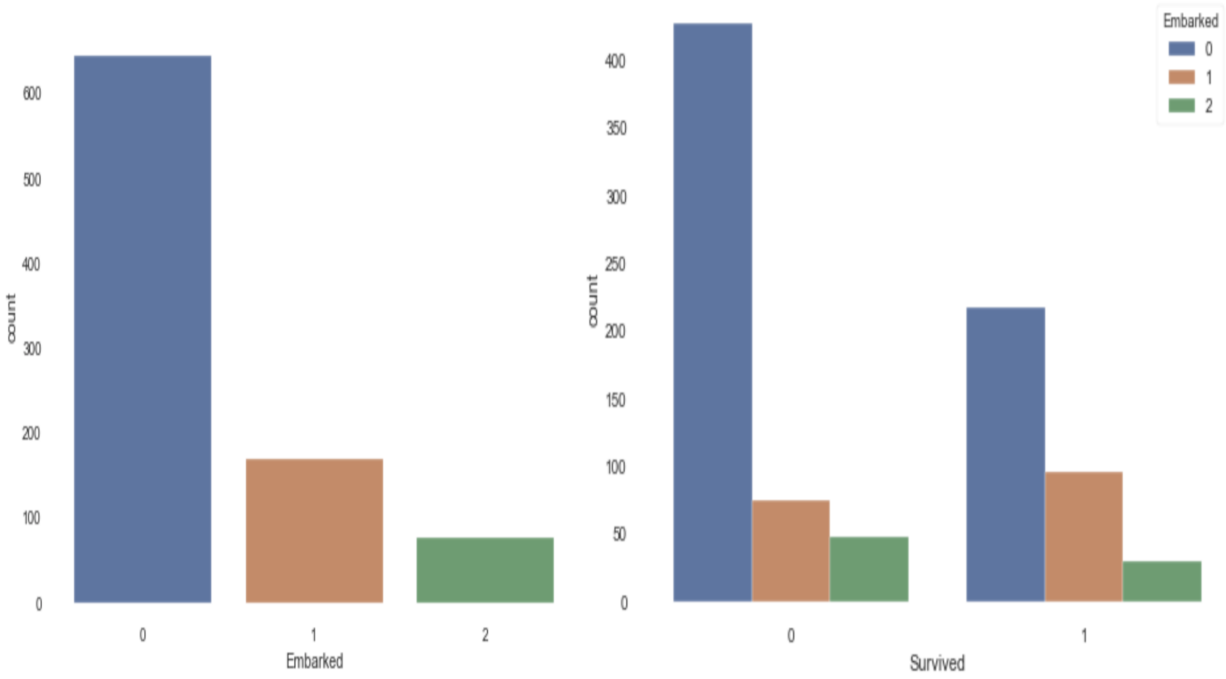


Figure A.5

(a) histogram of Titles **(b) histogram of Titles between survived or not**

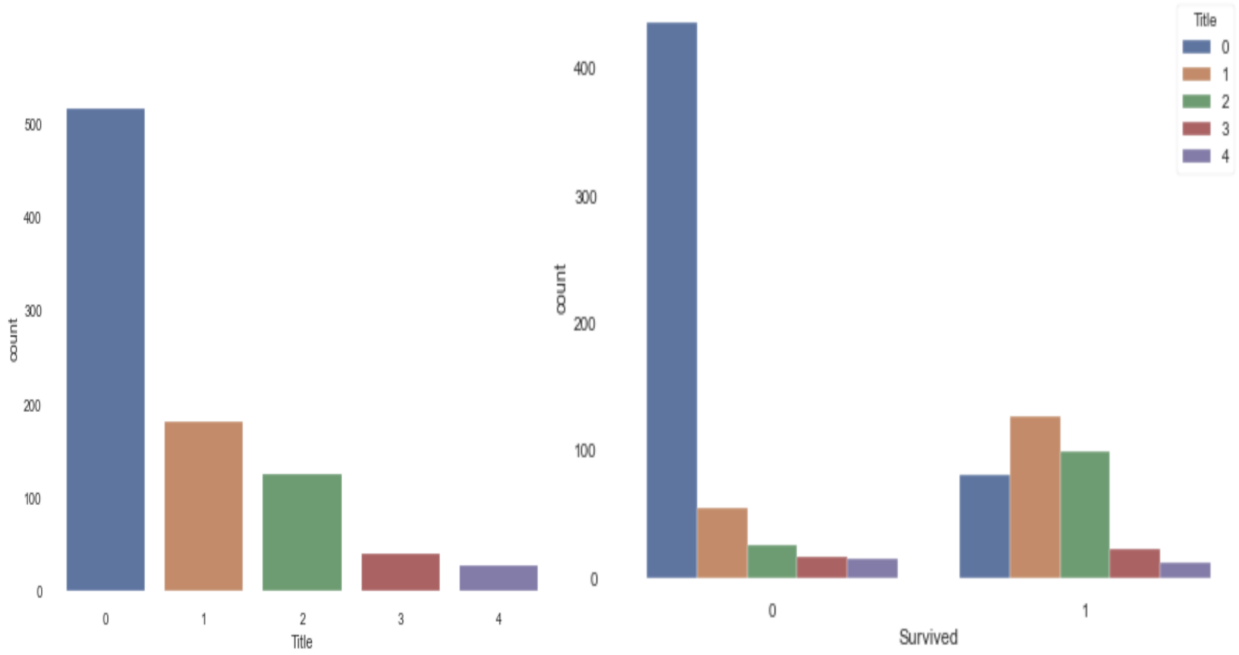


Figure A.6

(a) histogram of Family_groups **(b) histogram of Family_groups between survived or not**

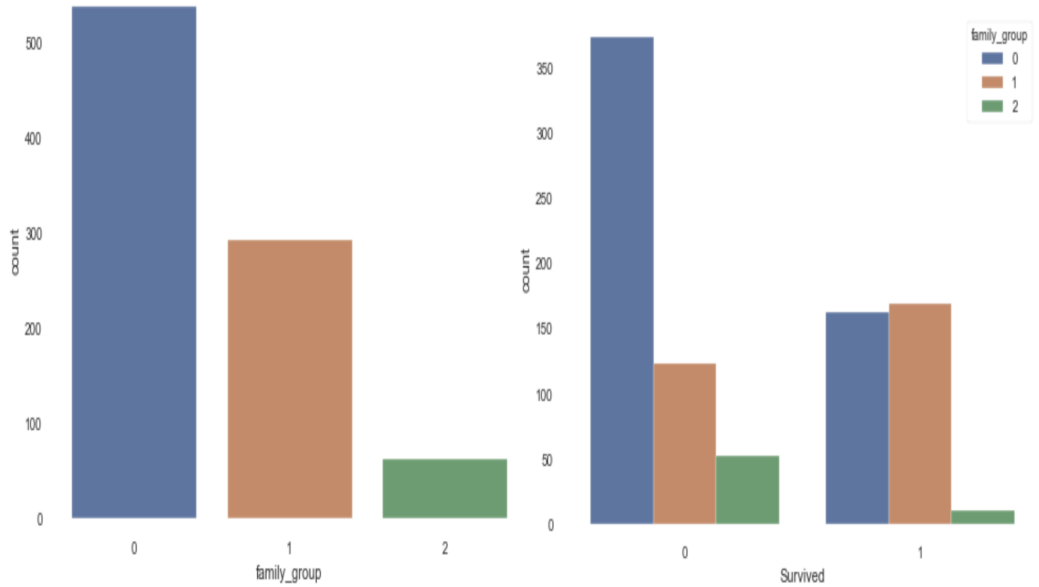


Figure A.7

(a) histogram of is_alone or not **(b) histogram of is_alone between survived or not**

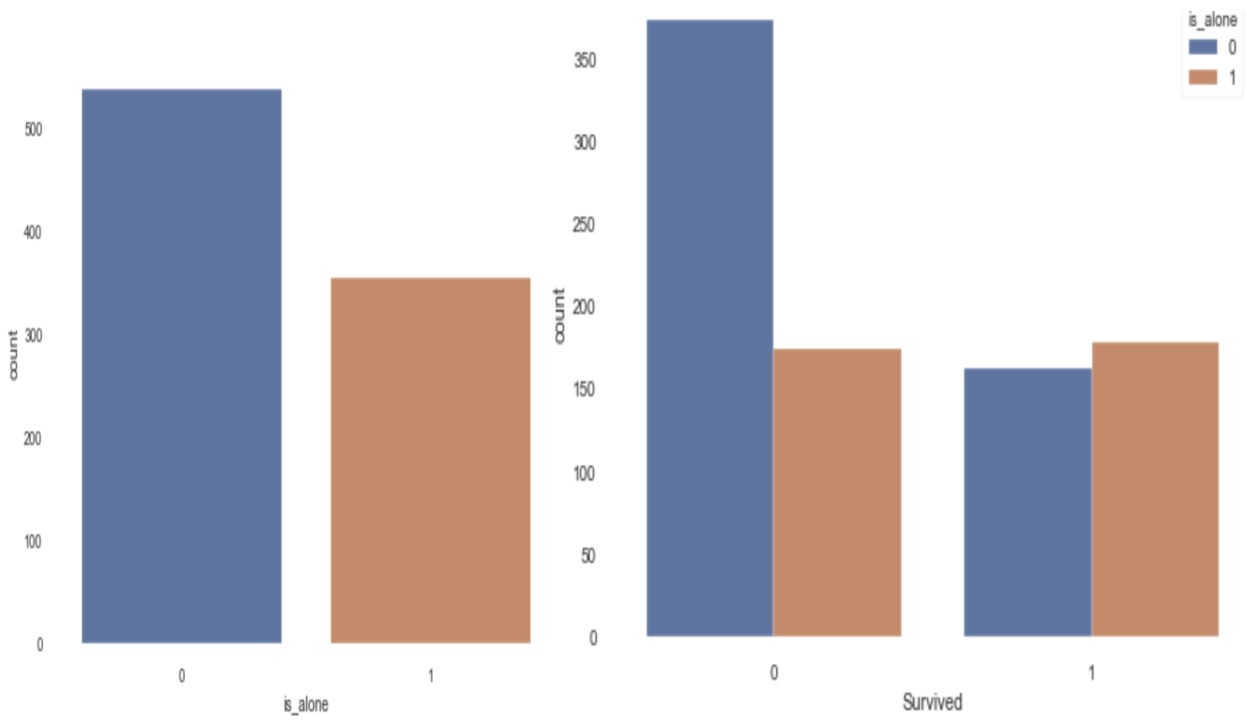
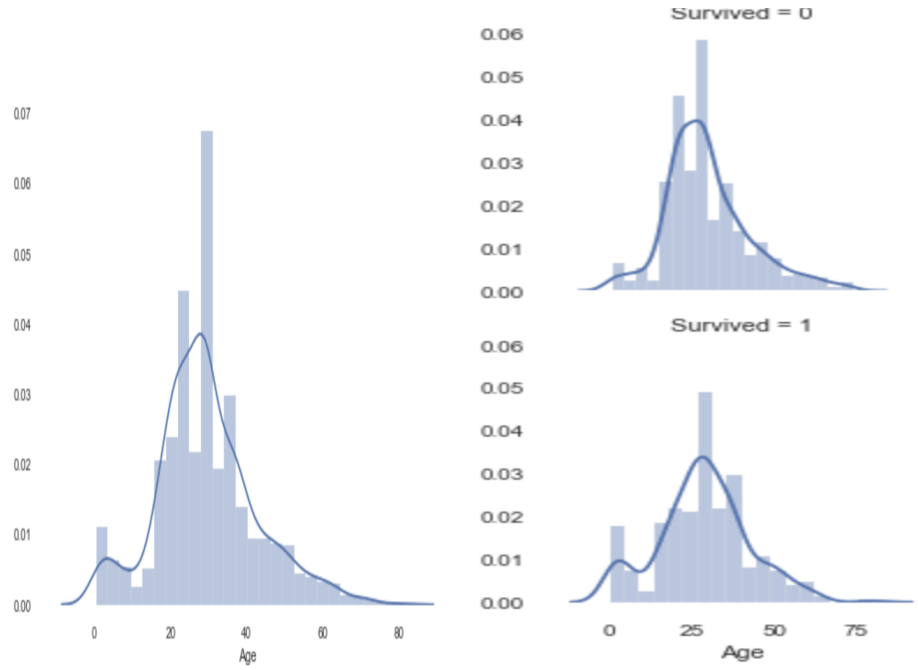


Figure A.8

(a) distribution of age **(b) distributions of age between survived or not**



(c) distributions of age between survived or not

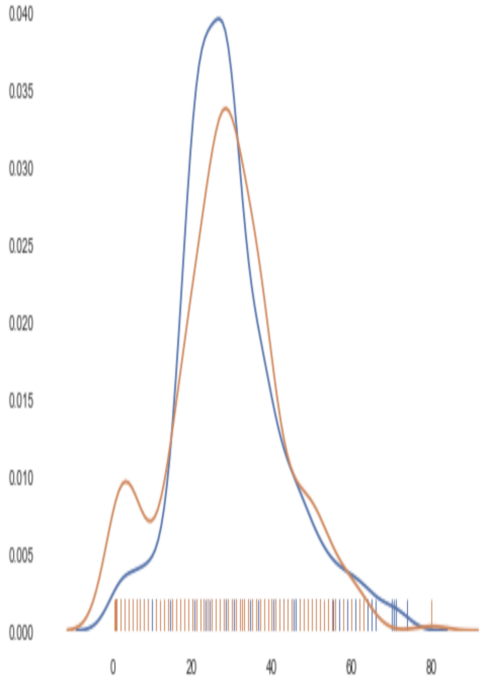


Figure A.9 : random forest [1]

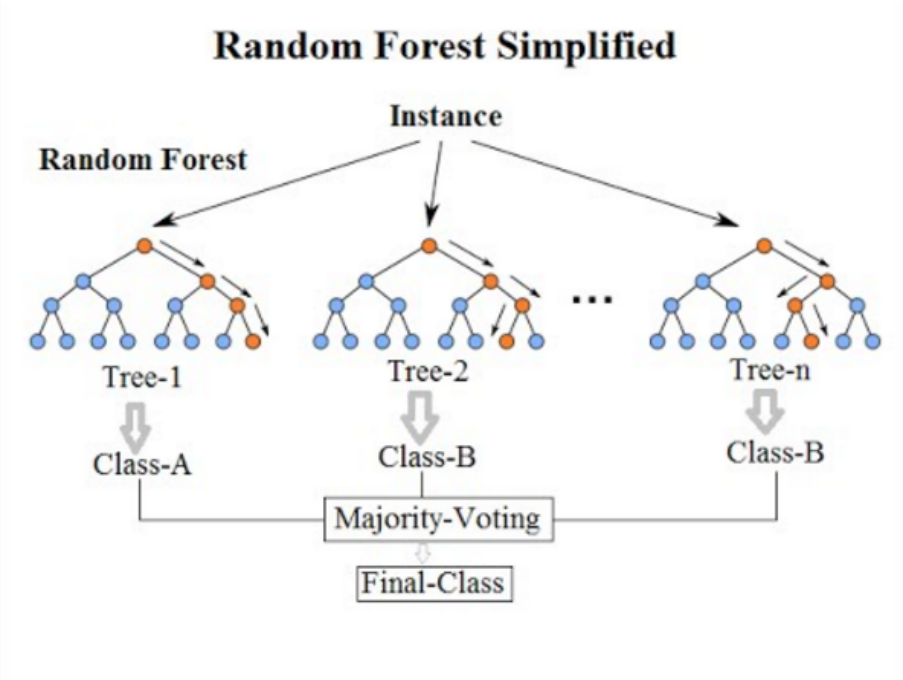


Figure A.10 hyperparameters of random forest

```
param_grid = {
    'bootstrap': [True],
    'max_depth': [80, 90, 100, 110],
    'max_features': [2, 3],
    'min_samples_leaf': [3, 4, 5],
    'min_samples_split': [8, 10, 12],
    'n_estimators': [100, 200, 300, 1000]
}
```

Figure A.11: Overfitting [2]

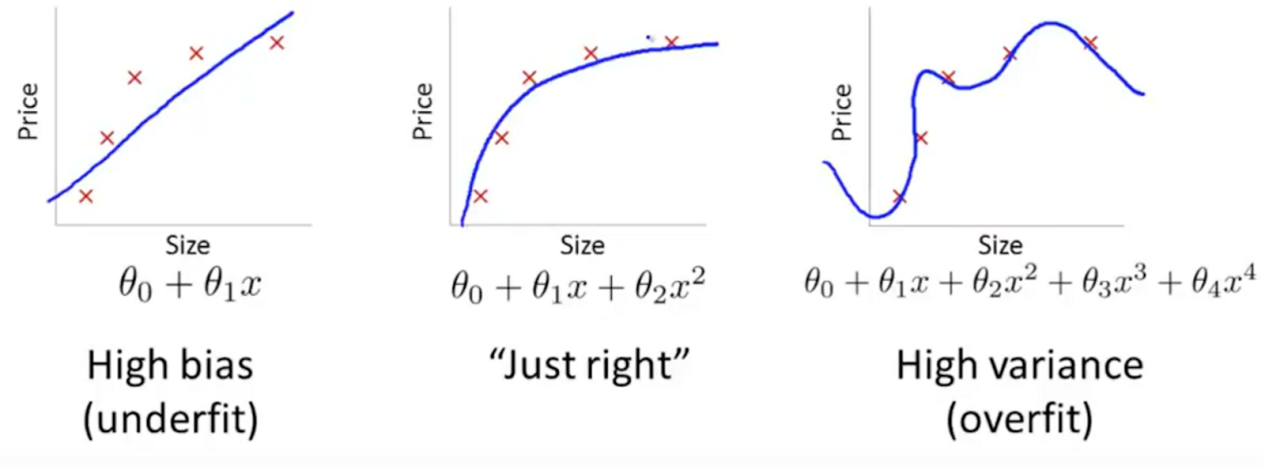


Figure A.12 K fold cross validation [3]

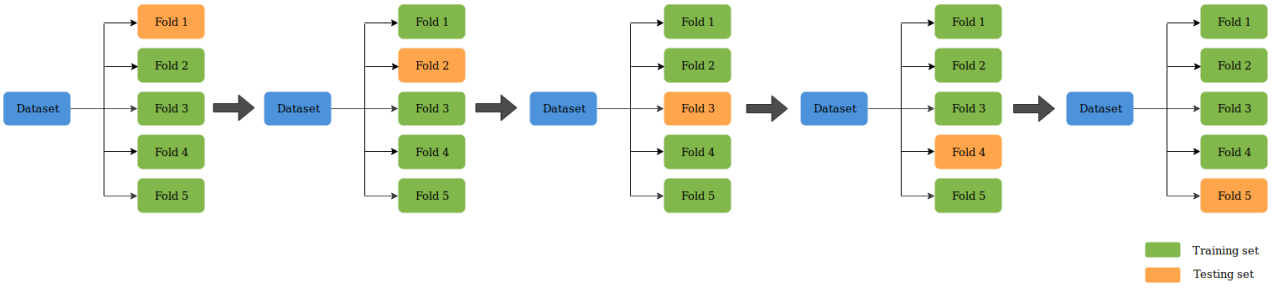


Figure A.13: GridSearchCV in Random Forest

```
RFC = RandomForestClassifier( random_state=0)


## Search grid for optimal parameters
rf_param_grid = {'bootstrap': [True, False],
 'max_depth': [10, 20, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [200, 400]}

gsRFC = GridSearchCV(RFC,param_grid = rf_param_grid, cv=10, scoring="accuracy",n_jobs= -1, verbose = 1)

gsRFC.fit(X_train,y_train)

RFC_best = gsRFC.best_estimator_

# Best score
gsRFC.best_score_
```

Figure A.14 Sklearn built-in methods fit() and predict()

```
>>> from sklearn.datasets import load_iris
>>> from sklearn.linear_model import LogisticRegression
>>> X, y = load_iris(return_X_y=True)
>>> clf = LogisticRegression(random_state=0).fit(X, y)
>>> clf.predict(X[:2, :])
array([0, 0])
>>> clf.predict_proba(X[:2, :])
array([[9.8...e-01, 1.8...e-02, 1.4...e-08],
       [9.7...e-01, 2.8...e-02, ...e-08]])
>>> clf.score(X, y)
0.97...
```

Figure A.15 hyper-parameters (C, penalty) for logistic regression

```
logistic_param = {
    "penalty":["l1","l2"],
    'C': [0.001,0.01,0.1,1,10,100,1000]
}
gsLog = GridSearchCV(LogisticRegression(), param_grid= logistic_param, cv = 10, scoring= "accuracy", n_job
s=-1)
gsLog.fit(X_train, y_train)
Logbest = gsLog.best_estimator_
```

Figure A.16 hyper-parameters (kernel, C, gamma) for SVM

```
svc_param_grid = {'kernel': ['rbf'],
                  'gamma': [ 0.001, 0.01, 0.1, 1],
                  'C': [1, 10, 50, 100,200,300, 1000]}
```

Figure A.17 hyper-parameters (n_neighbors, weights) for KNN

```
knn_param_grid ={
        'n_neighbors':num_1,

          'weights':['uniform', 'distance'],
```

Figure A.18 hyper-parameters for Random forest

```
rf_param_grid = {'bootstrap': [True, False],
 'max_depth': [10, 20, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [200, 400]}
```

Figure A.19 hyper-parameters for Gradient Boosting

```
gb_param_grid = {'loss' : ["deviance"],
              'n_estimators' : [100,200,300],
              'learning_rate': [0.1, 0.05, 0.01],
              'max_depth': [4, 8],
              'min_samples_leaf': [100,150],
              'max_features': [0.3, 0.1]
              }
```

The Department of Applied Economics and Statistics
College of Agriculture and Natural Resources
University of Delaware

The Department of Applied Economics and Statistics carries on an extensive and coordinated program of teaching, organized research, and public service in a wide variety of the following professional subject matter areas:

Subject Matter Areas

Agricultural Policy	Environmental and Resource Economics
Food and Agribusiness Management and Marketing	International Agricultural Trade
Natural Resource Management	Price and Demand Analysis
Rural and Community Development	Statistical Analysis and Research Methods

The department's research in these areas is part of the organized research program of the Delaware Agricultural Experiment Station, College of Agriculture and Natural Resources. Much of the research is in cooperation with industry partners, the USDA, and other State and Federal agencies. The combination of teaching, research, and service provides an efficient, effective, and productive use of resources invested in higher education and service to the public. Emphasis in research is on solving practical problems important to various segments of the economy.

The mission and goals of our department are to provide quality education to undergraduate and graduate students, foster free exchange of ideas, and engage in scholarly and outreach activities that generate new knowledge capital that could help inform policy and business decisions in the public and private sectors of the society. APEC has a strong record and tradition of productive programs and personnel who are engaged in innovative teaching, cutting-edge social science research, and public service in a wide variety of professional areas. The areas of expertise include: agricultural policy; environmental and resource economics; food and agribusiness marketing and management; international agricultural trade; natural resource management; operations research and decision analysis; rural and community development; and statistical analysis and research methods.

APEC Research Reports are published by the Department of Applied Economics and Statistics, College of Agriculture and Natural Resources of the University of Delaware.