Project Paper

RMS Titanic

——*Who are most likely to survive*

Zhiyang Wang, Ziyu Chen

16:137:550:01
Fundamentals of Analytics and Discovery Informatics
Instructor: Pottenger William

May 14th, 2014

Introduction

RMS Titanic was a famous Olympic class ocean liner, which was completed in 1912 by the Harland and Wolff shipyard in Belfast with Thomas Andrews as her naval architect [1] and operated by the White Star Line. During her maiden voyage, she traveled from Southampton, UK to New York City, US, via Cherbourg in France and Queenstown (now Cobh) in Ireland. Unfortunately, she collided with an iceberg just during her maiden voyage in North Atlantic Ocean on 15 April 1912, which caused two thirds of the passengers died. She carried more than 2,200 passengers and crew, more than 1500 people died in this shipwreck tragedy, there was not lack of aristocrats and millionaires in the passenger list. During that time, RMS Titanic was known as *unsinkable liner*, no one can image it will sink, so this tragedy caused a great sensation in the international community. This tragedy also led to better safety regulations for ships. [2]

As we see from the movie, there are only a few people who were lucky enough to get on a lifeboat, which was a fatal mistake that led to the tragedy, they didn't prepare enough lifeboats for all the passengers and crew. The lifeboats that Titanic had only covered around one-third of her total capacity. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class. [3]

By the time, there have been several expeditions getting the site to explore the mystery of RMS Titanic, they hoped to find treasure or to find the cause of the sinking. However, this project will find out the people with what kinds of attributes will survive when sinking occurs, and then make prediction for people on board who will survive the tragedy. In fact, during short period after RMS Titanic sank, some of the people who didn't get on the lifeboat are still alive, they were just floating in the Atlantic Ocean, but it was too cold, the cold water of the Atlantic Ocean made them gradually frozen, some of them died, some of them were picked up and rescued by the lifeboats which came back to search people who still alive. So the people who had good physical stamina may have a certain probability to survive. Roughly, from the statistic data we can conclude that people from first class cabins are most likely to survive, whereas people from third class cabins are hard to survive. Women are most likely to survive, whereas men are hard to survive. And passengers are more likely to survive than crews. [4] The above is only the rough analysis, in this project we are going to dig more information from the data set, and giving a more accurate prediction for people who are most likely to survive.

In this proposal, the concept, type of learning and a list of attributes of this project will be introduced, then the method of data obtaining and data cleaning will also be introduced. We will also discuss several algorithms which will perform in this project.

❖ Concept Learning

This project seeks to analyze the attributes of the people who survived when RMS Titanic sinking occurs using Data Mining techniques. The concept to be learned is the people who are most likely to survive the tragedy. This project will perform supervised learning to create several numeric and nominal prediction models to predict people who will survive the tragedy.

To make prediction, we should analyze the training data set first, then find the common attributes of people who are most likely to survive the tragedy. The common attributes can be age group (such as 15 to 25), cabin class, even ticket price. We can give a decision tree or other rules with other algorithms which can help to find out the people who are most likely to survive. Before doing this, we should make two simple assumption that the health status of the people who have been on board are normal and all the people are willing to survive. Because a sick young man may run slower than a healthy old woman, and when they escape from the cabin the young man may be submerged by water. Also if a person is willing to die, he will

become an outlier of the dataset, since most people have a desire to survive. With these two assumptions, we can make full use of our statistic data to learn the concept.

Motivation

RMS Titanic is a famous shipwreck tragedy in history, and nowadays transportation becomes more and more convenient, but tragedies happened frequently, like South Korea Shipwreck led to almost three hundred people died. So it's necessary to find out people with what features are most likely to survive when tragedy comes.

At the same time, as machine learning algorithms become more and more popular in data analytics field, it's a good chance for us to perform machine learning algorithms in data analysis. There are many kinds of machine learning algorithms, supervised or unsupervised, and for a certain algorithm, how to set proper parameters is also a challenge for us. So it's important for us to try different algorithms with different parameters in experiments, also try to evaluate the model using ROC curve of other metrics, then we can find out the optimal model with proper parameters. After built the model, how to evaluate the performance is also a problem for us, there are so many metrics such as TP rate, FP rate, Recall and Precision, and if we should do cross validation to make it better. All in all, we should find out the metric that is the most suitable to evaluate the performance.

In terms of analysis tools, there are many choices for us, such as Weka, Python even R language. Weka is a powerful tool that we learn in class, which encapsulates many algorithms in itself, such as random forests, OneR, SVM, and logistic regression, etc. What we need to do is to make a .arff file, then put it into Weka, by clicking a proper classifier we can get the result. It's very convenient and straight forward. We can get the precision, or TP rate, FP rate by one click to evaluate the performance. It's very easy to use. However, as a beginner, we are looking forward to get to know each algorithm, we need to know how it works and its working principles so that we could know the algorithms better. In this case, highly encapsulated algorithms of Weka will not benefit us, since all we need to do with Weka is to make a .arff file and throw it in, then we will get a model, but we won't know how does the model is built and how does it is learned. On the other hand, if we can implement the algorithms by ourselves, we could definitely learn the algorithms better. Another consideration is that in job market, employers from data analytic companies are not familiar with Weka, they tend to use Python or R language more often. So from the employment point of view, maybe choosing python or R is more suitable for us.

After considering that, we downloaded Python and studied on it for so long time, then we found Python is also very powerful and easy to use. At the beginning we are truly beginners on Python, after long time study, we were more and more familiar with it. There are many useful packages in Python, such as Scikit, Numpy, and Matplotlib, etc. What we need to do is to download and install them, then if we want to call some functions, we can look up the references of the package to determine how to write the code. One of the advantages of Python is that Python can make the result visualization, which means it can give us chart to express the result, comparing with plain numeric result, chart is more intuitive and easy to make sense.

Literature Review

In order to gather more background information and previous work done in the area of data mining of RMS Titanic, and also prove the novelty of this application specifically, a literature search has been completed.

The initial search was done using Wikipedia, which provide us a wealth of information about the history background of RMS Titanic. Within the preliminary understanding of RMS Titanic, Google Scholar and Citeseer website come in handy, Google Scholar provides many relevant papers of RMS Titanic, for

example, the chapter of *A Comparison of Several Approaches to Missing Attribute Values* in Data Mining gives us some insight about how to deal with missing values, even though we didn't apply exactly what says in the paper. Instead of using the most common value for attribute age, we used median value to do the replacement, and as for the attribute job and cabin, we used the method 9 mentioned in the paper, we treated them as special values. The impression deepest is a website named encyclopedia-titanica [5], which is a truly useful website, it provides a wealth information about the history, culture and people of RMS Titanic. There is a passenger list and a crew list provides personal information of the passengers and crews, including their age, sex, job, ticket, etc.

To determine which machine learning algorithms should be adopted, we have searched for so many paper, finally we decided to adopt Support Vector Machine, Random Forests, Logistic Regression and Neural Network, all of which are supervised machine learning algorithms. We have also tried unsupervised machine learning algorithms, such as k-means algorithm, however after reading some papers about k-means algorithm (*Constrained k-means clustering with background knowledge* [6]), we found it's hard to perform it on our project, because if we divide the people who are on board into two clusters, we still cannot say that people in cluster one will die and people in cluster two will survive, it won't be accurate. On the other hand, even we divide people into two clusters, we still don't know which cluster should be labeled as died and which cluster should be labeled as survived. So we have to discard unsupervised learning methods.

In the paper of *Introduction to Support Vector Machines* [7], we found the fundamental theory of SVM, including the history, mathematical properties, kernel functions, and how to implement SVM. After reading this paper we knew how to build the hyperplane, and what the support vectors are. If we only have 2 or 3 features, we can draw the hyperplane in a 2-D or 3-D space, it will be very intuitive. From this paper, we also found if we choose different parameters of the kernel functions will lead to different results, such as the parameter σ in RBF kernel function, different value of σ will lead to different result accuracy. So we have tried different parameters in RBF kernel function. And we also found linear kernel function, which is a special case of polynomial kernel functions. After several experiments and comparisons, we found the optimal kernel function with proper parameter whose accuracy is the highest.

In the paper of Random Forests [8], we found theoretical background for random forests, forests using the random selection of features at each node to determine the split, out-of-bag estimates, and applications of random forests. Each decision tree in the random forest just like an expert in a narrow field (since we select m out of M features for each decision tree to learn), so there are lots of experts who are proficient in different fields in the random forest, and for each new input, each expert will approach the problem from different angles then vote to get the result.

From this paper, we also knew that select different number of features at each node will influence the result, and different number of decision trees in the forest will also influence the result. So we choose different number of features and decision trees for experiments. After several experiments and comparisons, we found the optimal number of features at each node and number of decision trees in the forest whose accuracy is the highest.

From the *book Logistic Regression: A Self-Learning Text* by David G. Klein Baum, Mitchel Klein [9], we learned the basics about Logistic Regression. It is a model that outputs binary result based multiple input features. The way it makes decision is to use a sigmoid function, whose result is also ranging from zero to one, to compute the result of an instance with all the input features, the result it regard as the probability of the instance to be "1", in our case, the probability of a people to be survived. And by setting a threshold, normally 50%, it will classify all the instance either to be "0" or "1".

From the paper *Theory of the back propagation neural network* [10] by Hecht-Nielsen, we learned a basic back propagation learning method for the neural network. To do back propagation, we need to calculate the error rate at the output layer first, use the predict value minus the actual value, and then use it to calculate the error rate for the second last layer, and use the second last layer to calculate the third last layer, and go on until it reaches the second layer (the input layer doesn't have an error rate). Using the error rate we could get the derivative of the cost function, which could help us to do gradient decent to find the best weights for each connection.

From the paper *A Comparison of Several Approaches to Missing Attribute Values in Data Mining* [11], we learned several useful ways to deal with missing values. It provides up to 9 ways to deal with missing values, including use the most common value to replace the missing value and treat the missing value as a new specific value. Normally, when we have a lot of regular instance, and the missing values are rare, we could use the most common values to replace the missing values, since it has the most possibility to occur, it's a little similar to OneR. And in our case, when there are a lot of values are missing in one feature, it is inaccurate to find known attribute value as the missing value, instead, we treat it as a new value just like other values.

Approach

❖ Evaluation

➢ Evaluation metric

We will output the result in term of accuracy, precision, recall. But the final metric will be use precision. The purpose of this project is to learn who is most likely to survive, so each person we predicted survived, we hope we can predict it right, and this is exactly what does precision stands for. We are using accuracy to give us a general estimate of the model, since it will give us feedback of how many instance the classifier predicted right, but we are not using accuracy as our result of our test data because it will yield misleading result if the data set is unbalanced. Since our data set is unbalanced, you can easily score accuracy around 65% by predicting all the people are dead, but in practice the classifier will have 100% recognition rate for the dead class while have a 0% recognition rate for the survived class, this is not what we want. We are not using recall because we are focusing on get every instance that is predicted survived to be right, instead of trying to find out all the survived instance.

➢ Performance Evaluation

Because of the limited data size, we used 10 fold cross validation on our training set to calculate the precision. That means we split our training set into two subsets, the one subset consists of 80 percent instances which are randomly chosen is the training subset, we train our model on this subset. And then we ran our trained model on the rest instance subset to calculate the accuracy. Iterate the above steps 10 times, the mean of ten results will be our final output accuracy of the cross validation.

Because we don't have the actual result in our test data, the only way we can test our model on the test data is to upload the predict results to the kaggle host, and it will feedback the accuracy of the performance on our test data.

❖ Data Acquisition

➢ Collection

The collection of our data is quiet straight forward. Since our project is from kaggle.com, the host already provides the whole data we need on the website. So, we download the training data which contains 891 instance and the test data which contains 419 instance directly from the website.

➢ Preparation

To prepare the data for our model, we first split it into two dataset, a training set consists of 891 instance and a testing set consists of 419 instance. And each instance have 10 features: Pclass, Name, Sex, Age, Sibsp, Parch, Ticket, Fare, Cabin, and Embarked.

After splitting the data, we dealt with the missing values. There are a bunch of missing values in attribute age, cabin, and few in embarked. Following is the method we apply to deal with missing values:

Name	Type	Range	Description
Pclass	Integer	0,1,2	The passenger's class on the boat.
Name	String		Passenger's name
Age	Integer	0-100	Passenger's age
Sex	String		Passenger's sex
Sibsp	Integer	0-10	How many siblings of the passenger are one the boat
Parch	Integer	0-10	How many children or parents of the passenger are one the boat
Ticket	String		Passenger's ticket number
Fare	Float	0-400	How many does the passenger spend on cabin and class
Cabin	String		Passenger's room number
Embarked	Character	Q,S,C	Where the Passenger embarked.

- For attribute age, our method is quiet straight forward, we calculate the median value according to the missing instance's gender, and replace the blank with this value. [12]
- For attribute fare, we believe it is correlated to at least three attributes, class, cabin, and embarked position. Since the information about cabin is extremely limited, we use the other two attributes to estimate the missing fare. What we do is that we first split all passengers based on their class into 3 groups, and within each group, we split them again based on the embarked position into three groups. And we calculate the average fare of each group, and replace the missing fare using the correlated median value. [8]
- For the attribute cabin, since there are a lot of missing values in this attribute, and the format of existing values varies a lot, so our strategy here is to treat missing value as a specific value. But after applying it to the model, it didn't work well, so we decided to discard this attribute. [11]

Besides the missing value, the attribute ticket has chaos format. Some of them are purely numbers, some of them are made of characters, and some of them are the combination. And it seems that the number and the characters are used randomly, we can't find any relation between ticket and any other attribute, so we decided to discard this attribute too.

The following is the list our data:

➢ Input

We are using python to build our model, and we used the data structure of dataframe from the library Pandas, so the format of our input data could be very casual. The final format we use is a csv file, with the first raw being the attribute name, and the instances are located under the attribute name. Our program will directly load the data from the csv file.

➢ Processing

We built up our model by executing python code. The program will load data from a csv file, and based on the instructions, it will fit the appropriate data to the model. Our program was coded in 64-bit windows or mac osx operating system, and it is single- threaded.

➢ Output and interpretation

After the program is executed, it will output our predict value in a csv file as well as ROC curve, confusion matrix, and most importantly the accuracy and precision of our model.

➢ Storage

The predicting result will be stored in a csv file. All the results in the transition phase will not be stored.

Use of algorithms

❖ Supervised learning algorithms

Supervised learning is the machine learning task of inferring a function from labeled training data. [13] In supervised learning, each instance contains a pair of features and desired result, which also means before training the machine, we should put the results into the training data set, whereas, unsupervised learning doesn't have to do that. Here, we adopt Logistic Regression, Support Vector Machine (SVM) and Random Forests to make classification and prediction.

➢ Support Vector Machine

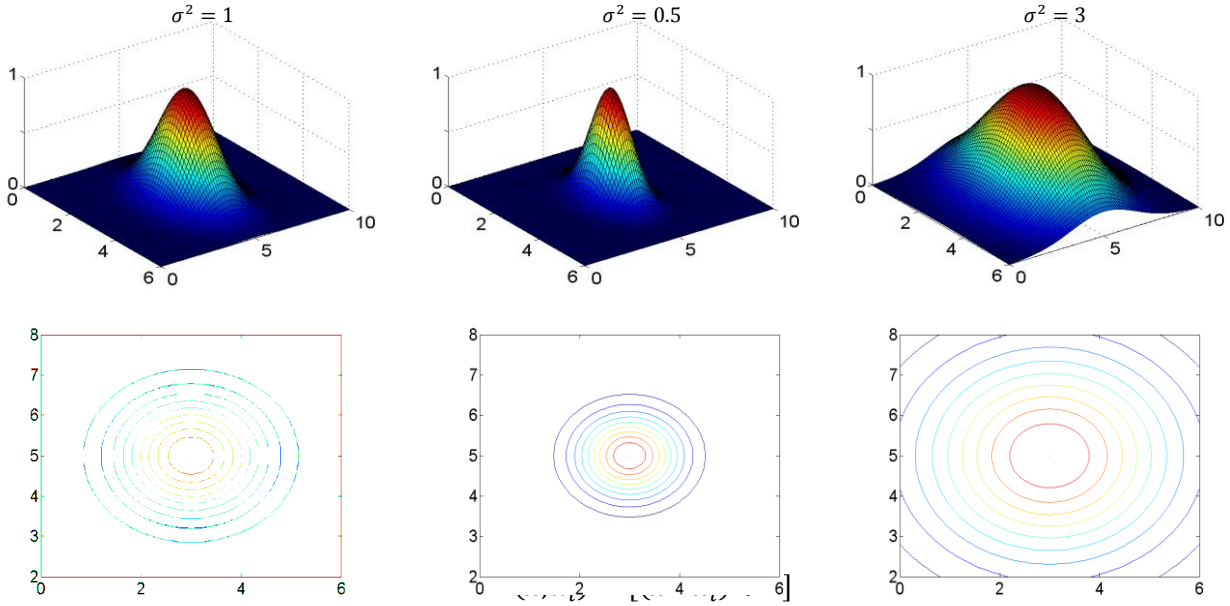
Support vector machine is a kind of supervised learning method with associated learning algorithms. Support vector machine can be used for classification and regression analysis. [14] Given a set of training data, each instance has been marked as belonging one certain category, an SVM training algorithm can build a model that classify new instance into one category, making it a non-probabilistic binary linear classifier. [15] An SVM model represents the instances as points in space, mapped so that the instances of separate categories are divided by a clear gap that is as wide as possible. New instances such as test data are then mapped into the same space and make a prediction for which category it belongs to, based on which side of gap the fall on.

Kernel function plays a key role in support vector machine. Usually, the vectors in low dimensional space is difficult to classify, the solution is mapping these vectors into a high dimensional space. However, attendant problem is that runtime complexity increased. However, kernel function can solve this problem, which means that if we choose kernel function properly, then we can get classification function in high dimensional space. In SVM, adopting different kernel functions will lead to different results. In this paper, we will use SVM to predict people who will survive with two different kernel functions —radial basis function (rbf) and linear kernel function.

Radial basis function (RBF), as known as Gaussian kernel function:

$$K(x, x_i) = \exp\left\{-\frac{\|x - x_i\|^2}{2\sigma^2}\right\}$$

The following figure shows the influence of different values of parameter σ.



We can get p order polynomial classifier from that, if we set p as 1, then we get linear kernel function.

➢ Random Forests

Random forests are an ensemble learning method for classification (and regression) that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. [8] The forests are built in random manner and there are lots of decision trees in the forests, but decision tree is independent of each other. After building the forest, when new sample coming, each decision tree in the forest will respectively makes decision to determine which category the sample belongs to, the final result depends on which class has the most votes.

During the process of establishing each decision tree, there are two significant points —Sampling and splitting completely. First is the two processes of random sampling, random forest will take samples from each row and each column. For column sampling, random forest adopts the method of sampling with replacement, which means there may exist duplicate samples in the set of obtained samples. Assuming that the number of input sample set is N, so the number of sampling samples is also N, which can avoid overfitting problem due to every tree won't contain all the input samples during training process. For row sampling, selecting m features from M features, where m << M. After sampling, random forest will build decision trees by completely splitting method, which can ensure a leaf node cannot be split anymore, or all the samples belongs to the same category. For general decision tree algorithms, there is a significant process which is pruning. However, that is not necessary for random forests, since the above two random sampling processes ensure the randomness of decision trees, even not pruning, it still won't appear overfitting problem.

We can just assume that each decision tree is an expert in a narrow field (since we select m out of M features for each decision tree to learn), so there are lots of experts who are proficient in different fields in the random forest, and for each new input, each expert will approach the problem from different angles then vote to get the result.

➢ Logistic Regression

Logistic Regression is a type of regression that predicts the probability of occurrence of an event by fitting data to a logistic function. Like many forms of regression analysis, it makes use of several predictor variables that may be either numerical or categorical. For instance, the probability that a person has a heart attack within a specified time period might be predicted from knowledge of the person's age, sex and body mass index. This regression is quite used in several scenarios such as prediction of customer's propensity to purchase a product or cease a subscription in marketing applications and many others.

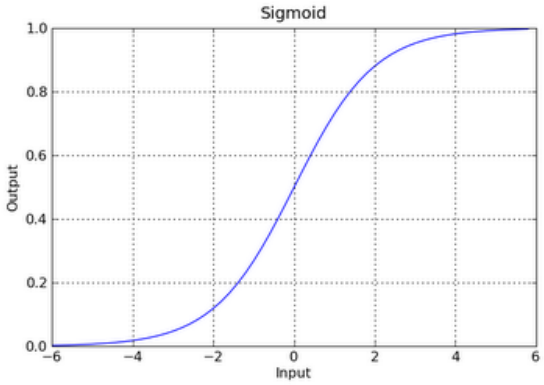
The logistic regression hypothesis is defined as:

$$h_\theta(x) = g(\theta^T x)$$

Where the function g is the sigmoid function. It is defined as:

$$g(z) = \frac{1}{1 + e^{-z}}$$

The sigmoid function has special properties that can result values in the range [0, 1]. So you have large positive values of X, the sigmoid should be close to 1, while for large negative values, the sigmoid should be close to 0.



The variable z equals to input vector x multiply the vector θ. And the sigmoid function will return us a output range from 0, 1, which is the possibility of the instance to be 1. With a decision boundary, then we classify the instance to be 1 or 0.

Given X, to find the best z, we need the cost function and gradient for logistic regression, they are given as below:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}[-y^{(i)}\log(h_\theta(x^{(i)})) - (1 - y^{(i)})\log(1 - h_\theta(x^{(i)}))]$$

And the gradient of the cost is a vector theta where the j element is defined as follows:

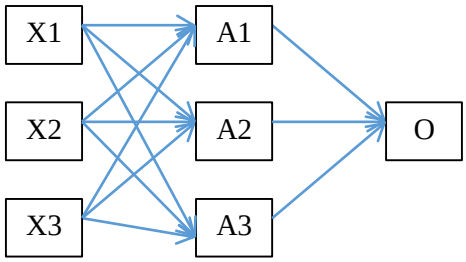
$$\frac{\varphi J(\theta)}{\varphi \theta_j} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

With the cost function, after applying method like gradient decent, the model could learn the best θ [16], and then give the best z.

➢ Neural network

ANN (Artificial Neural Network) is an artificial network that consists of many simple connected function units, which is an abstract and a simulation of basic features of human or creatures' neural network.

Figure shows the basic structure of a neural network:



The first layer is the input layer consist of input vector X, the second layer is the hidden layer consists of some functional units, normally sigmoid function, it process the inputs and output its result to the next layer. The third lawyer is the output layer, which use the result from hidden layer as inputs, after processing the inputs it will output the final result.

To determine the weight on each connection, method like back propagation is applied. Back propagation neural network is to calculate the error at the output layer at first:

$$\delta^3 = \text{output} - a^3, \text{ where } a^3 \text{ is the actual value.}$$

And then calculate δ^2:

$$\delta^2 = (\theta^2)^T * \delta^3 .* g'(z^2), \text{ where } g'(z^i) = a^i .* (1 - a^i)$$

And it is proved that [17]

$$\frac{\partial}{\partial \theta_{ij}^l} J(\theta) = a_j^l * \delta_i^l$$

So, by applying the gradient decent, we can update the weights on every connection.

Result

❖ Support Vector Machine

Features Selection: C(Pclass) + C(Sex) + Age + SibSp + Parch + C(Embarked)

➢ Rbf Kernel

Radial basis function (RBF), as known as Gaussian kernel function:

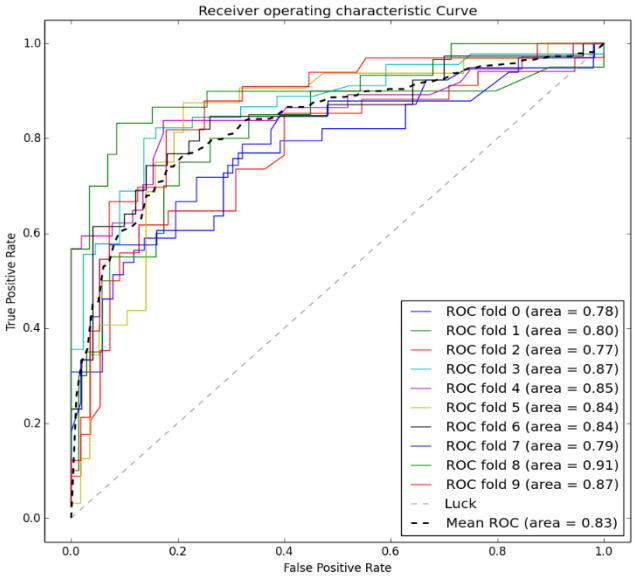
$$K(x, x_i) = \exp\left\{-\frac{\|x - x_i\|^2}{2\sigma^2}\right\}$$

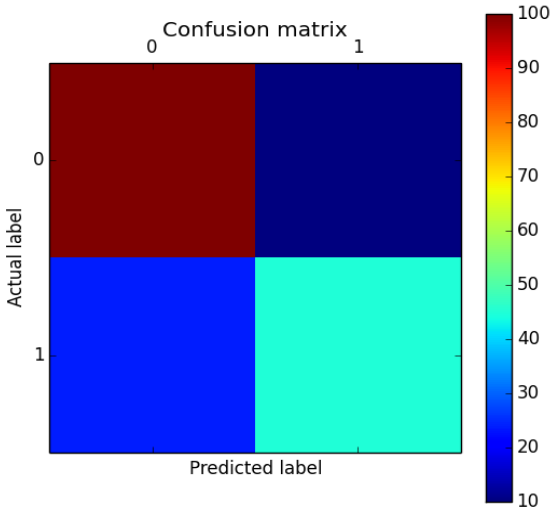
Here, as we introduced in Algorithm section, factor σ^2 will influence the performance of SVM, so we choose different value for σ^2, and then we can see different results as following:

Output representation:

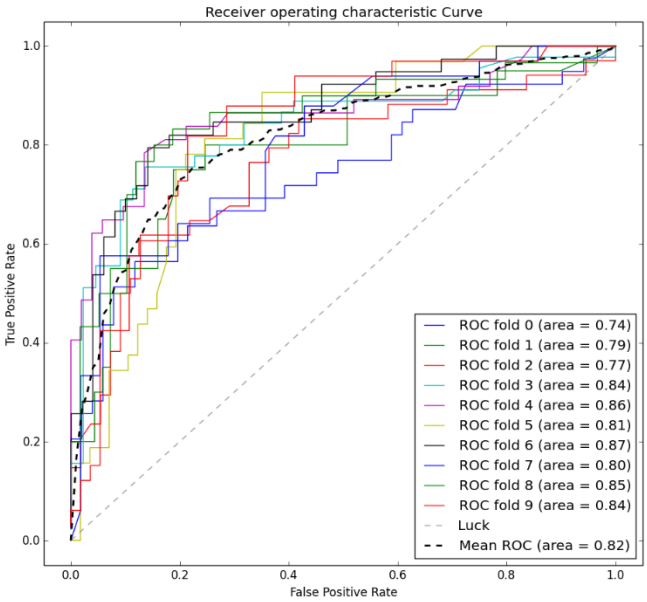
svm.SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=2, gamma=1.0, kernel='rbf', max_iter=-1, probability=True, random_state=None, shrinking=True, tol=0.001, verbose=False)

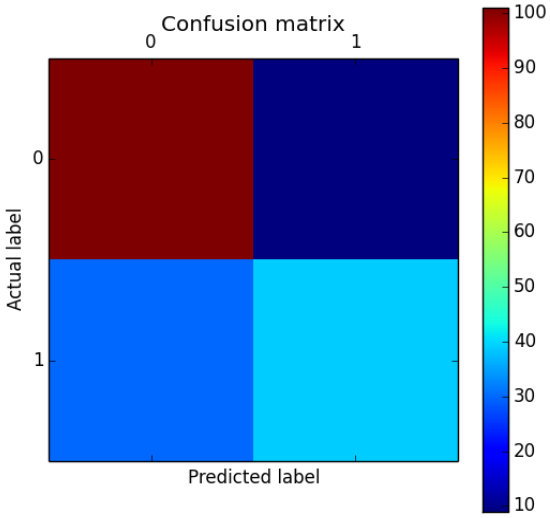
Cross Validation:



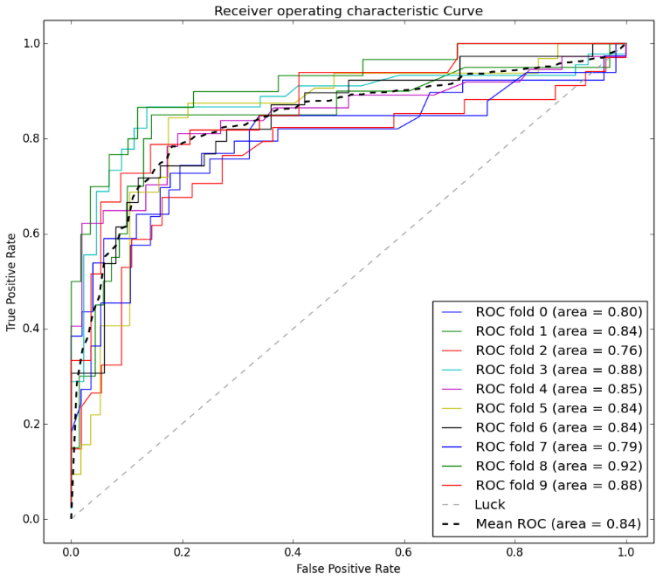


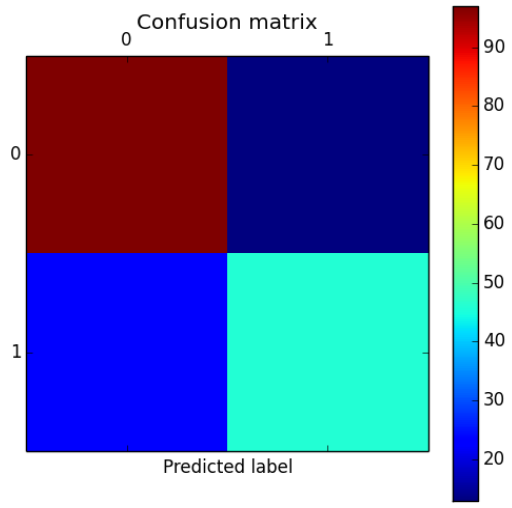
$\sigma^2 = 1.0$ $AUC = 0.83$





$\sigma^2 = 0.5$ $AUC = 0.82$





$\sigma^2 = 3.0$ $AUC = 0.84$

	Mean AUC	Mean Accuracy
$\sigma^2 = 1.0$	0.8310	78.68%
$\sigma^2 = 0.5$	0.8163	76.89%
$\sigma^2 = 3.0$	0.8390	80.25%

Training Data:

$\sigma^2 = 1.0$

Pre / Actual	Died	Survived
Died	530	19
Survived	65	277

	TP Rate	FP Rate	Precision
Died	96.54%	19.01%	89.08%
Survived	80.99%	3.46%	93.58%

$\sigma^2 = 0.5$

Pre / Actual	Died	Survived
Died	537	12
Survived	49	293

	TP Rate	FP Rate	Precision
Died	97.81%	14.33%	91.64%
Survived	85.67%	2.19%	96.07%

$\sigma^2 = 3.0$

Pre / Actual	Died	Survived
Died	499	50
Survived	87	255

	TP Rate	FP Rate	Precision
Died	90.89%	25.44%	85.15%
Survived	74.56%	9.11%	83.61%

Test Data:

$$\sigma^2 = 1.0$$

Actual \ Pre	Died	Survived
Died	126	13
Survived	28	56

	TP Rate	FP Rate	Precision
Died	90.65%	33.33%	81.82%
Survived	66.67%	9.35%	81.16%

$$\sigma^2 = 0.5$$

Actual \ Pre	Died	Survived
Died	121	18
Survived	31	53

	TP Rate	FP Rate	Precision
Died	87.05%	36.90%	79.61%
Survived	63.10%	12.95%	74.65%

$$\sigma^2 = 3.0$$

Actual \ Pre	Died	Survived
Died	122	17
Survived	24	60

	TP Rate	FP Rate	Precision
Died	87.77%	28.57%	83.56%
Survived	71.43%	12.23%	77.92%

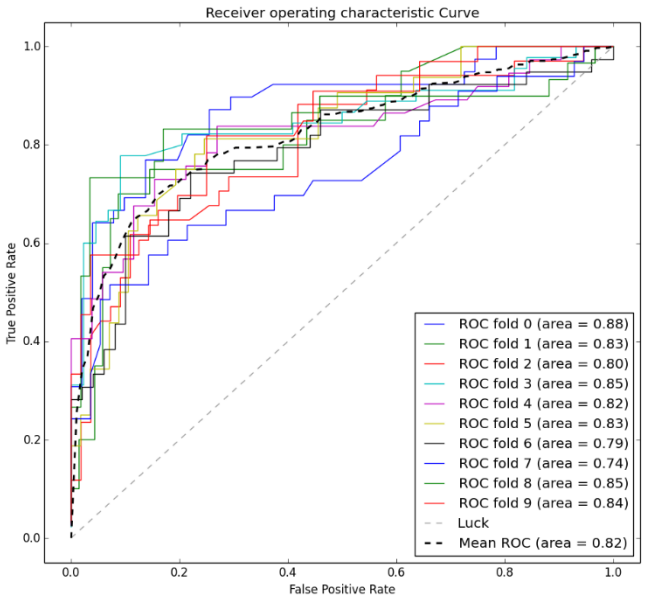
➢ Linear Kernel

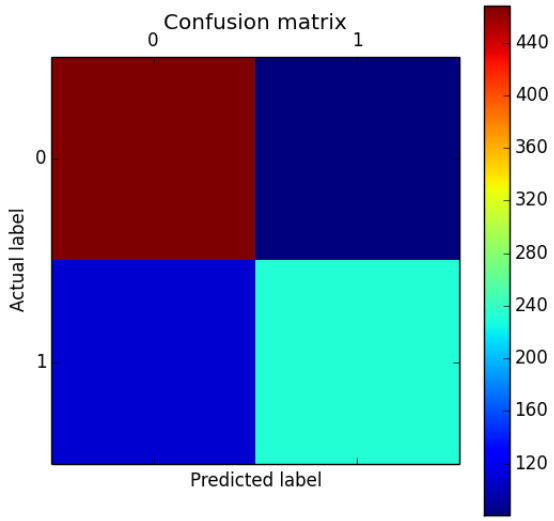
Linear kernel is a special case of polynominal kernel function, the order of which is one. Let's see the result below:

Output representation:

svm.SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=1, gamma=1.0, kernel='linear', max_iter=-1, probability=True, random_state=None, shrinking=True, tol=0.001, verbose=False)

Cross Validation:





Mean AUC	Mean Accuracy
0.8216	78.67%

Training Data:

Pre / Actual	Died	Survived
Died	468	81
Survived	109	233

	TP Rate	FP Rate	Precision
Died	85.25%	31.87%	81.11%
Survived	68.13%	14.75%	74.20%

Test Data:

Pre / Actual	Died	Survived
Died	115	24
Survived	25	59

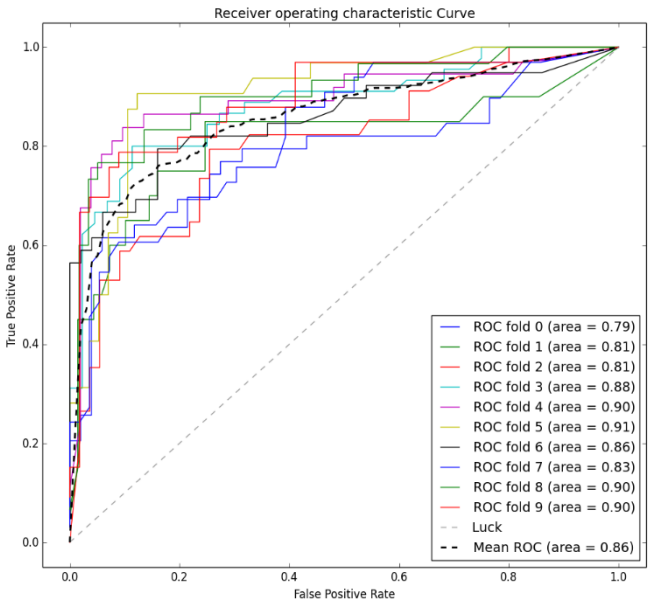
	TP Rate	FP Rate	Precision
Died	82.73%	29.76%	82.14%
Survived	70.24%	17.27%	71.08%

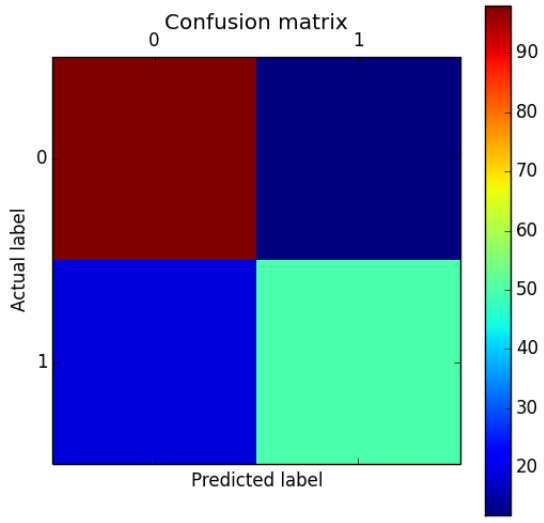
❖ Random Forests

Features Selection: C(Pclass) + C(Sex) + Age + SibSp + Parch + Fare + C(Embarked)

A. Output representation: Random forest of 100 trees, each constructed while considering 4 random features.

Cross Validation:





Mean AUC	Mean Accuracy
0.8572	80.93%

Training Data:

Pre / Actual	Died	Survived
Died	540	9
Survived	10	332

	TP Rate	FP Rate	Precision
Died	98.36%	2.92%	98.18%
Survived	97.08%	1.64%	97.36%

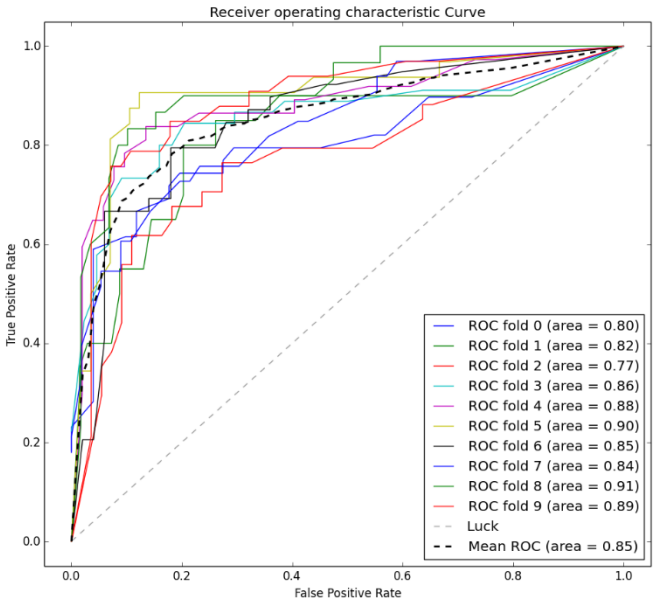
Test Data:

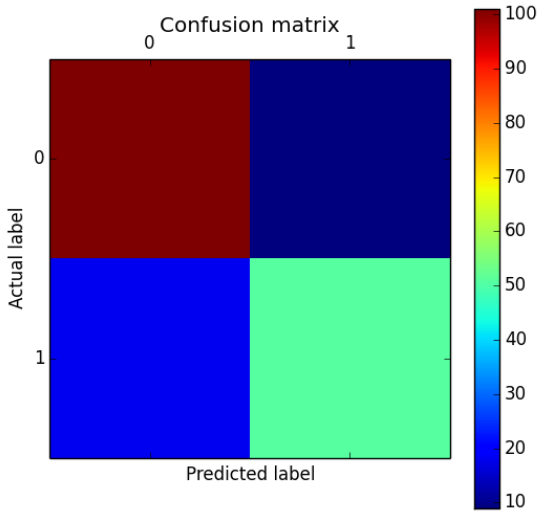
Pre / Actual	Died	Survived
Died	98	12
Survived	19	50

	TP Rate	FP Rate	Precision
Died	89.09%	27.54%	83.76%
Survived	72.46%	10.91%	80.65%

B. Output representation: Random forest of 20 trees, each constructed while considering 4 random features.

Cross Validation:





20 trees, 4 features/tree

Mean AUC	Mean Accuracy
0.8534	80.58%

Training Data:

Pre / Actual	Died	Survived
Died	535	14
Survived	9	333

	TP Rate	FP Rate	Precision
Died	97.45%	2.63%	98.35%
Survived	97.37%	2.55%	95.97%

Test Data:

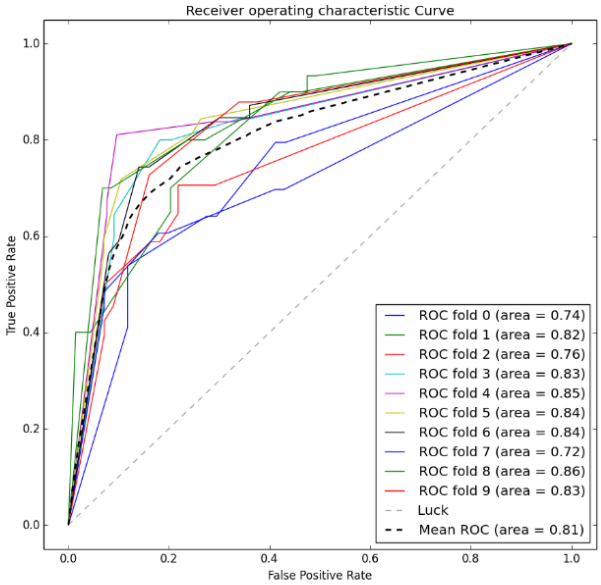
Pre / Actual	Died	Survived
Died	101	9
Survived	18	51

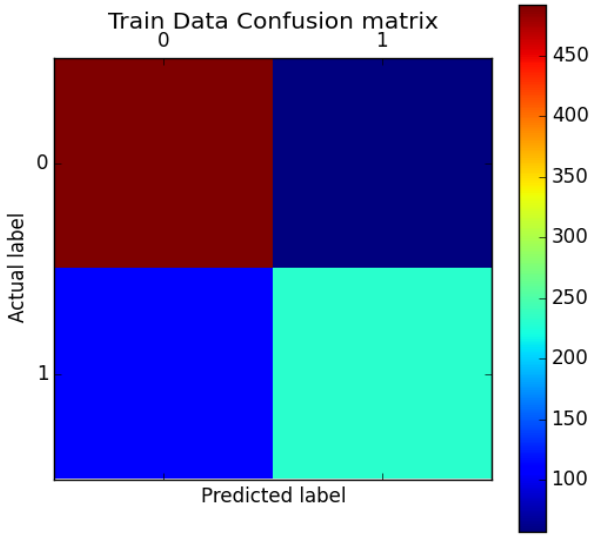
	TP Rate	FP Rate	Precision
Died	91.82%	26.09%	84.87%
Survived	73.91%	8.18%	85.00%

❖ Neural Network

Features Selection: C(Pclass) + C(Sex) + Age + SibSp + Parch + C(Embarked)

Cross Validation:





Mean AUC	Mean Accuracy
0.8168	76.71%

Training Data:

Pre / Actual	Died	Survived
Died	440	109
Survived	98	242

	TP Rate	FP Rate	Precision
Died	80.15%	28.82%	81.78%
Survived	71.18%	19.85%	68.95%

Test Data:

Pre / Actual	Died	Survived
Died	88	26
Survived	20	43

	TP Rate	FP Rate	Precision
Died	77.19%	31.75%	81.48%
Survived	68.25%	22.81%	62.32%

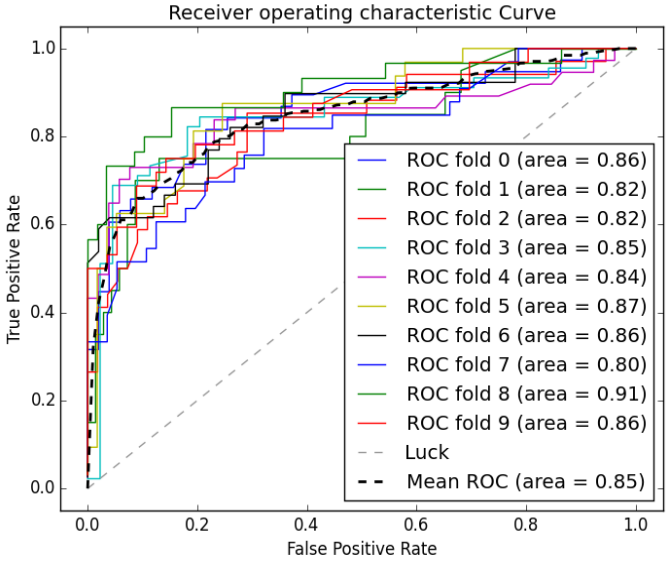
Output representation:

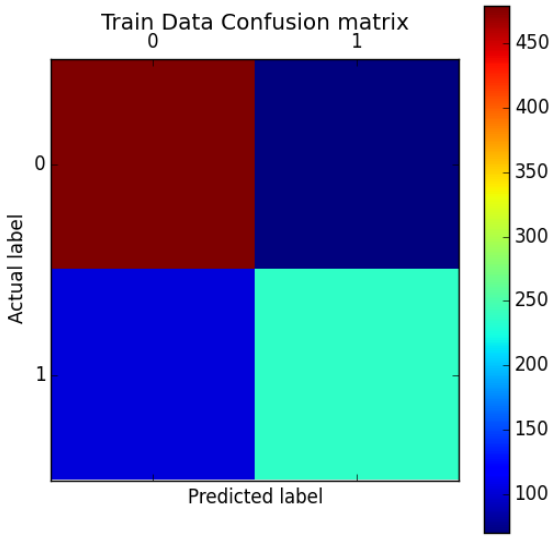
```
>>> net.params
array([  2.24036524e-01,   3.52130791e-01,  -3.56303526e-02,
        -7.11817859e-02,   5.51283432e-02,   7.59782946e-02,
         1.07084137e-01,  -3.56347799e-02,  -7.76302447e-02,
         6.10702231e-01,   1.01214238e+00,  -2.89770560e-02,
         2.65258997e-01,  -9.77128419e-02,   1.24605537e-01,
        -1.50927121e-01,  -7.11648838e-02,  -8.53748668e-02,
         3.09720897e-01,   5.36233800e-01,  -1.68492077e-02,
         1.13085090e-01,   6.86349507e-02,   1.05509222e-01,
        -1.09682509e-01,   5.50670502e-02,   7.85166190e-02,
        -4.25666641e-01,  -7.32334153e-01,   2.26973136e-02,
        -1.75309712e-01,  -5.10763461e-03,  -8.94445068e-02,
         9.66972643e-02,   7.60568770e-02,   9.35394586e-02,
        -1.22944410e-01,  -1.45527465e-01,  -5.71752622e-04,
        -1.57707070e-02,  -1.36283725e-01,  -1.78950802e-01,
         1.33873486e-01,   1.07115485e-01,   1.26487780e-01,
        -2.50992964e-01,  -3.85656535e-01,   1.06263799e-02,
        -6.36107260e-02,  -1.17673048e-01,  -1.85189636e-01,
         1.89162380e-01,   3.49163423e-01,   3.92050593e-01,
        -7.73090118e-02,  -2.75148623e-01,  -2.82922279e-01,
        -2.48654713e-01,  -5.82902639e-02,   3.02540373e-01,
         3.23080210e-01,   3.88097770e-01])
```

❖ Logistic Regression

Features Selection: C(Pclass) + C(Sex) + Age + SibSp + Parch + C(Embarked)

Cross Validation:





Mean AUC	Mean Accuracy
0.8467	79.41%

Training Data:

Actual \ Pre	Died	Survived
Died	479	70
Survived	103	237

	TP Rate	FP Rate	Precision
Died	87.25%	30.29%	82.30%
Survived	69.71%	12.75%	77.20%

Test Data:

Actual \ Pre	Died	Survived
Died	98	12
Survived	19	50

	TP Rate	FP Rate	Precision
Died	81.90%	42.47%	73.50%
Survived	57.53%	18.10%	68.85%

Output representation:

Coefficients: [2.15955893, -0.66918462, -1.92782039, -2.79994969, 0.26797427, -0.52929524, -0.04446271, -0.32114044, -0.08315593, 0.00421118]

Discussion

❖ Support Vector Machine

➢ Rbf Kernel

When $\sigma^2 = 3.0$, mean AUC is 0.8390, and the mean accuracy is 80.25%, which performs better than $\sigma^2 = 0.5$ and $\sigma^2 = 1.0$. And from precision of training data and test data, we can see that the results of precision seems reversed, in training data, model with $\sigma^2 = 0.5$ performs better than the other two models, however, in test data, model with $\sigma^2 = 3.0$ performs better than the other two models. The reason of this circumstance maybe that for training data, we put all the data into learning, so smaller σ^2 can make the model more accurate, but in terms of test data, the model will predict some unknown value, so bigger σ^2 maybe better. And I guess that smaller σ^2may lead to overfitting. So we should choose $\sigma^2 = 3.0$ for the model of rbf kernel.

➢ Linear Kernel

There is no parameter that we can tune in linear kernel model, the precision of both survived and died in training data and test data are less than that of rbf kernel ($\sigma^2 = 3.0$, the best choice), maybe that linear kernel is too simple to build the hyperplane, so the result lacks of accuracy.

❖ Random Forests

We performs two different random forests, the first contains 100 decision trees, each constructed while considering 4 random features. Another contains 20 decision trees, each constructed while considering 4 random features. The mean AUC and mean accuracy of these two models are much closed, so we can't say which one is better depends on these two metrics. So let's see the precision of training data and test data, in training data, the forest with 100 decision tree performs a little better than the forest with 20 decision trees, however, in test data, the forest with 20 decision tree performs a little better than the forest with 100 decision trees. I think the reason is that too much decision trees may lead to overfitting. So we should choose the model of 20 decision trees, each constructed while considering 4 random features.

❖ Neural Network

As we can see, neural network performs really badly in our case. We are trying to figure it out, the following is what we found:

1. The missing values do matter. I think there are problems about how we deal with missing values. When we deal with missing values, the highest accuracy we get is 71%. But after we drop all the missing values, we could get the accuracy around 75%. So, I guess how we deal with missing values have great influence on the Neural network.
2. As we can see, the classifier predicted most of instance dead, we thought the reason is that maybe our layer is not enough so that our algorithm can perform well, but after we change the number of function units and even add a layer, the result didn't improve significantly.
3. The result varies a lot. It varies from 60% - 70%. It shouldn't be, I'm still trying to figure this out.
4. All the outputs are around 0.5

There are still many problems, but the precision is good, so I think neural network could work in our case.

❖ Logistic Regression

As we can see, the precision drops a lot from training data to test data, I guess we overfitted a little bit. There is not much we can do with Logistic Regression, Only thing I can think of is to change the threshold,

in our case, 51% works better than the original 50%. But other than that, our optimizing choice is really limited.

❖ Overall

For overall discussion of all the algorithms, we just choose each algorithm with optimal parameters to analyze,

Model	Mean AUC	Mean Accuracy
SVM (RBF, $\sigma^2 = 3.0$)	0.8390	80.25%
Random Forest (20 trees, 4features)	0.8534	80.58%
Neural Network	0.8168	76.71%
Logistic Regression (threshold = 51%)	0.8467	79.41%

Model		SVM	Random Forest	Neural Network	Logistic Regression
Precision	Died	83.56%	84.87%	81.48%	73.50%
	Survived	77.92%	85.00%	62.32%	68.85%

From the above table, random forest performs the best among all the algorithms, which has the largest mean AUC and the highest mean accuracy, the precision of test data is also the highest.

Conclusion and Future work

❖ Conclusion

We built four classification models: Logistic Regression, Neural network, Random Forest, SVM. And the Random Forest performed best in terms of precision, which is most important to us, accuracy, mean AUC so far. It has the precision of 84.87% and 85.00%. And by doing this project, we learned about how to clean our data, and also, we learned the importance of the data cleaning during the optimization, it really has huge influence on the classifier performance. Also, we learned about above four algorithms very well since we implement them by ourselves. Finally, we learned using python to do data mining, which I think is the one of the basic requirement for a data scientist. In a word, this project give us a valuable experience of how to do a data mining project.

❖ Future work

First all of, there are still improvement on how we do data cleaning. As we discussed, the missing value have huge influence on some algorithms like neural network. We may figure out a way to calculate the miss values in feature age more accurately. And for the feature ticket, cabin, we didn't use them in our model because of the chaos format. Maybe we could also figure out a way to apply them in the future. Moreover, besides the data the kaggle host give us, we could search more useful features from the internet, we already found some features that may be very helpful, like the job of the passenger, and whether the passenger is a crew or not. Maybe in the future, we could merge them with our existing data set. In a word, our preparation of data and feature selection could be improved a lot.

As for the optimization of our models, we could figure out the optimized number of function units in the hidden layer. As for SVM, we could give a try for polynomial kernel or other kernel. And more importantly, we may try other more advanced learning algorithm instead of only doing gradient decent, like conjugate gradient or BFGS. And also, we could always try more algorithms on this dataset, like k-nearest neighbor.

Besides the optimization, since we already get some model, maybe we could do something in practice, maybe apply our model on other sink boat dataset, and do some predictions on the passenger of that boat.

Appendix A: Work log

Date	Duration(hours)	Task
Mar,11-Mar,27	Around 40	Learned and installed Python, including libraries like pandas, skitlearn, matplotlib
Mar 28-April 4	Around 20	Learned the basics of algorithms like Logistic Regression
April 4-April 7	Around 10	Data cleaning
April 8 – April 13	Around 10	Implemented Logistic Regression
April 8 – April 14	Around 10	Implemented SVM
April 14 – April 22	Around 15	Implemented Neural network
April 15 – April 19	Around 10	Implemented Random Forest
April 20 - April 23	Around 10	Tried to implement K-means
April 23 – April 24	Around 5	Finished implementation of Logistic Regression and Random Forest.
April 25 – May 4	Around 40	Tried to figure out the problem of Neural network and optimize it.
May 2 – May 3	Around 5	Visualized the result and discussion
May 4 – May 10	Around 20	Wrote final project paper

References

[1] Wikipedia, "Titanic," [Online]. Available: https://en.wikipedia.org/wiki/Titanic.

[2] C. C. Page, ""Titanic Ship Listing".," 12 April 2012.. [Online]. Available: http://www.chriscunard.com/titanic.php.

[3] A. Gill, Titanic: The Real Story of the Construction of the World's Most Famous Ship, Transworld, 2012.

[4] WikiPedia, "Titanic passengers list," [Online]. Available: https://en.wikipedia.org/wiki/List_of_Titanic_passengers.

[5] encyclopedia-titanica, "titanic-passenger-list," [Online]. Available: http://www.encyclopedia-titanica.org/titanic-passenger-list/.

[6] Wagstaff, Kiri , Cardie, Claire, Rogers, Seth and Schr, "Constrained k-means clustering with background knowledge," in *ICML*, 2001, pp. 577--584.

[7] Boswell and Dustin, Introduction to support vector machines, August, 2002.

[8] Breiman and Leo, "Random forests," *Machine learning*, vol. 45, pp. 5--32, 2001.

[9] Kleinbaum, D. G. a. Klein and Mitchel, Logistic regression: a self-learning text, Springer, 2010.

[10] Hecht-Nielsen and Robert, "Theory of the backpropagation neural network," in *Neural Networks, 1989. IJCNN., International Joint Conference on*, IEEE, 1989, pp. 593--605.

[11] Grzymala-Busse, J. W. a. Hu and Ming, "A comparison of several approaches to missing attribute values in data mining," in *Rough sets and current trends in computing*, Springer, 2001, pp. 378--385.

[12] Kaggle, "Getting started," Kaggle, [Online]. Available: http://www.kaggle.com/c/titanic-gettingStarted/details/getting-started. [Accessed 30 04 2014].

[13] M. Mehryar , R. Afshin and Talwalk, Foundations of machine learning, MIT Press, 2012.

[14] Wikipedia, "Support Vector Machine," [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine. [Accessed 09 05 2014].

[15] WIkipedia, "Support Vector_Machine," [Online]. Available: https://en.wikipedia.org/wiki/Support_vector_machine. [Accessed 02 05 2014].

[16] Baldi and Pierre, "Gradient descent learning algorithm overview: A general dynamical systems perspective," *Neural Networks, IEEE Transactions on*, vol. 6, pp. 182--195, 1995.

[17] Hecht-Nielsen and Robert, "Theory of the backpropagation neural network," in *Neural Networks, 1989. IJCNN., International Joint Conference on*, IEEE, 1989, pp. 593--605.

[18] L. BREIMAN, "Random Forests," *Machine Learning,* vol. 45, pp. 5-32, 2001.