CMPS 242 Machine Learning - Exploration of the Titanic Dataset

Students: Viktor Jankov, Abhishek Grover, Ankit Gupta

Preprocessing

Preprocessing played a very important part of our project and we spent an entire day on this stage only. We brainstormed ideas and discussed which preprocessing steps we needed to take in order to clean up our data. As the saying goes “garbage in, garbage out”, so we made sure to clean our data first, and clean it in the best way possible so that we are able to run our models and achieve good results.

First we imported our data, extracted the labels and dropped irrelevant features. We decided to drop the following features: PassengerId , Name, Cabin and Ticket. These features are all unique, and we did not believe that they would add any predictive values to our model. For example the passenger name will not tell us anything about whether a passenger would survive or not.

After we dropped the features, we also dropped any instances with missing data. While there are several different ways to deal with missing data (mean imputation, using statistical models to allow for missing data etc.) we decided that it would be best to simply drop these instances.

Next in our preprocessing stage we encoded nominal feature values. The features Sex and Embarked both contained nominal values, so we performed a transformation for each. Males became 0 and females 1, and the three different embarkment stations took on the range from 0 to 2.

The final step in our preprocessing step was to standardize the data, and for this we used the Standard Scaler.

Basic exploratory statistics and visualizations

After our preprocessing stage was done, the first thing we did was run some very basic exploratory statistics [Figure 1] and visualization on our data. We did this in order to understand it better and find odd feature distributions that we could use later on in our models.

We ran a boxplot for each feature in order to see the distribution [Figure 2]. We did not see anything unusual, except that most of people embarked on only one port.

Next we plotted the survived vs died on the ship by making a pie chart [Figure 3]. This actually proved very useful because we saw that the distribution between survivors and deaths was not even so we had to use Stratified Cross Validation.

Analysis of Model Performance

We worked with total of seven models, and for each one we ran it on all the combinations of the seven features we had initially chosen. Then, for each one of these runs, we performed 10 fold stratified cross validation. At each fold, we used 1/10th as a validation set and 9/10ths as the train test. At each run, we calculated the accuracy and kept score so that we know which parameters worked the best.
In the end, when we had found the optimal parameters, we re-trained our model with the best parameters on the entire training set and ran it against the test set, in order to generate the csv file to submit to Kaggle.

Model Performance and Analysis

1. Random Forests was our best performing model with a Kaggle score of 79.4% [Figure 7] on the test data. Based on our validation set, we ran random forests with 10 trees, each of which had max depth of 5 and minimum samples split of 1. The max depth was specifically important, because when we didn't have max depth specified, our random forests would overfit our training data and our accuracy was lower. By pruning our tree, we managed to increase our accuracy on the test set and lower the variance. Figure 4 shows one tree from our forest.

2. Decision Trees had slightly lower accuracy than random forests, with Kaggle score of 78.4%. This makes sense as random forests are just a more sophisticated version of decision trees. In the case of decision trees as well, we had to prune our tree because with no pruning it overfit our training data and result in worse accuracies. In this case, we limit the depth of the tree, as well as the number of nodes.

3. Logistic Regression gave a Kaggle score of 75.5%. The accuracy we obtained was 80% on the training data when we only considered 'Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Embarked' as attributes. We get an accuracy of 77.9% when we just consider 'Sex' attribute, so we can conclude that probability of surviving was heavily dependent on 'Sex' attribute and other attributes must contribute very little. So, the weight obtained from the classifier should be max for 'Sex' attribute.

4. SVC gave a kaggle score of 77.5%. We obtained the best accuracy of 82.3% on training data using Pclass, Sex, Age, SibSp and Parch attributes. This indicates that most of the attributes points in higher dimension space are adequately separated from each other such that SVC was able to generate a hyperplane which could classify almost 82% points correctly.

5. k-Nearest neighbors, we get a kaggle scrore of 76.55% and an accuracy of 81.6% on training data for k=20 and when considering Pclass, Sex, Age, SibSp, Parch and Embarked attributes. This indicates that the samples must exist together closely in the 6-dimensional space where Pclass, Sex, Age, SibSp, Parch and Embarked are the dimensions.

6. For **Bernoulli Naive Bayes**, we get a kaggle score of 76.55% and the accuracy we obtained on training data was 77.9%. Interestingly, we obtained this accuracy for only 'Sex' attribute among all the combinations of attributes. This confirms the high dependency of survival rate on 'Sex' attribute. Since Naive Bayes performs well when the attributes are independent this lower accuracy shows us that the attributes are highly dependent on each other.

Conclusion:

We get the best accuracy of 79.4% and a rank of 1168 on kaggle using Random Forests. We also conclude from the accuracies we obtained over the training data by using different types of classifiers and all the possible combinations of attributes that the survival rate is heavily dependent on 'Sex' attribute.

Graphs and Figures:

Figure 1 - Basic Statistics

Pclass	Sex	Age	SibSp	Parch	Fare	Embarked	
count	7.120000e+02	7.120000e+02	7.120000e+02	7.120000e+02	7.120000e+02	7.120000e+02	7.120000e+02
mean	-1.587369e-16	-5.083325e-17	2.584933e-16	4.216353e-16	-1.372186e-17	-6.221615e-17	-1.640386e-16
std	1.000703e+00	1.000703e+00	1.000703e+00	1.000703e+00	1.000703e+00	1.000703e+00	1.000703e+00
min	-1.482983e+00	-7.561375e-01	-2.017717e+00	-5.527137e-01	-5.067874e-01	-6.534272e-01	-5.012257e-01
25%	-1.482983e+00	-7.561375e-01	-6.657639e-01	-5.527137e-01	-5.067874e-01	-5.012575e-01	-5.012257e-01
50%	-2.871914e-01	-7.561375e-01	-1.133826e-01	-5.527137e-01	-5.067874e-01	-3.576726e-01	-5.012257e-01
75%	9.085997e-01	1.322511e+00	5.770939e-01	5.225108e-01	6.647471e-01	-2.962586e-02	-5.012257e-01
max	9.085997e-01	1.322511e+00	3.477095e+00	4.823409e+00	6.522419e+00	9.031168e+00	3.336115e+00

Figure 2 - Feature boxplots

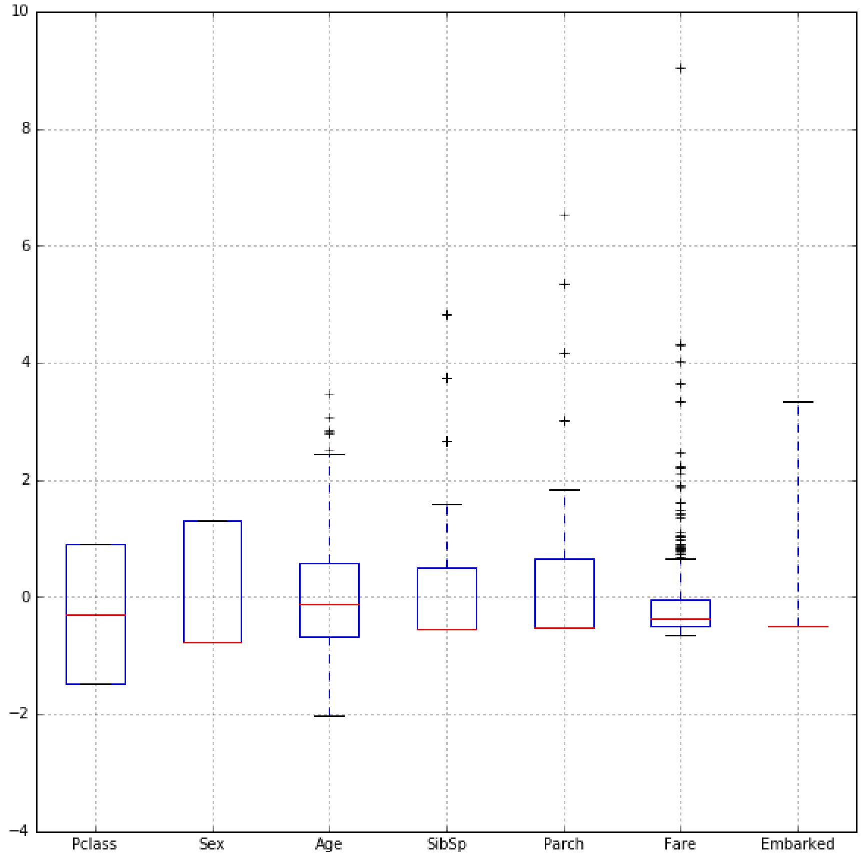


Figure 3 - Survivor vs Deaths

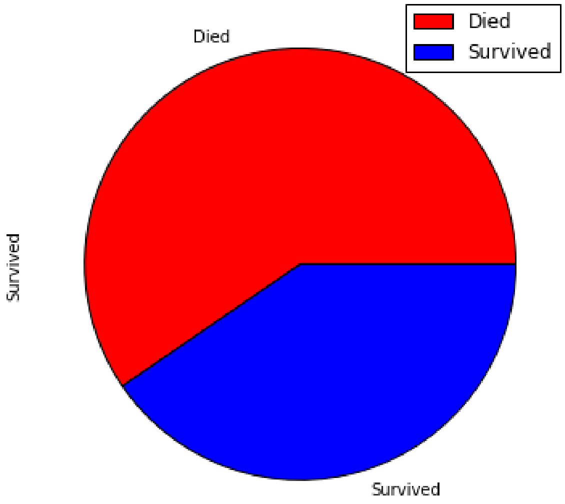


Figure 4 - Random Forest Single Tree

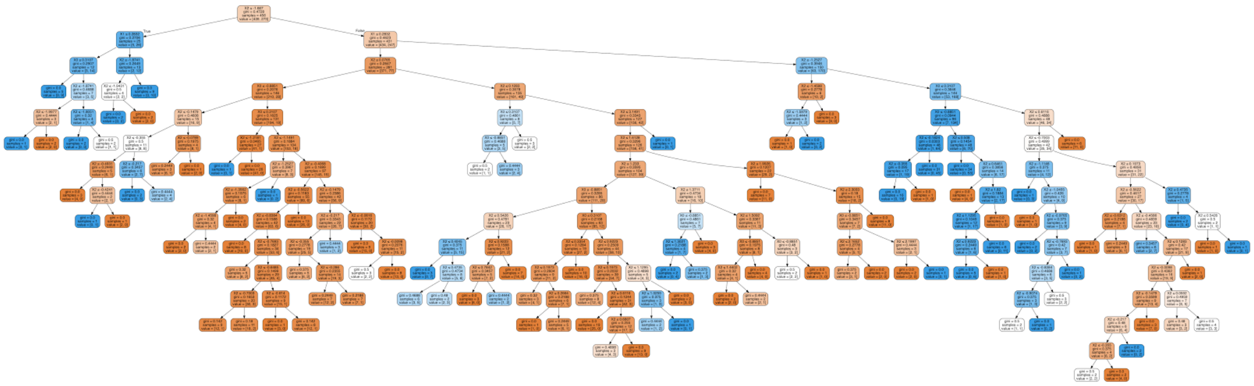


Figure 5 - Decision Tree Visualization

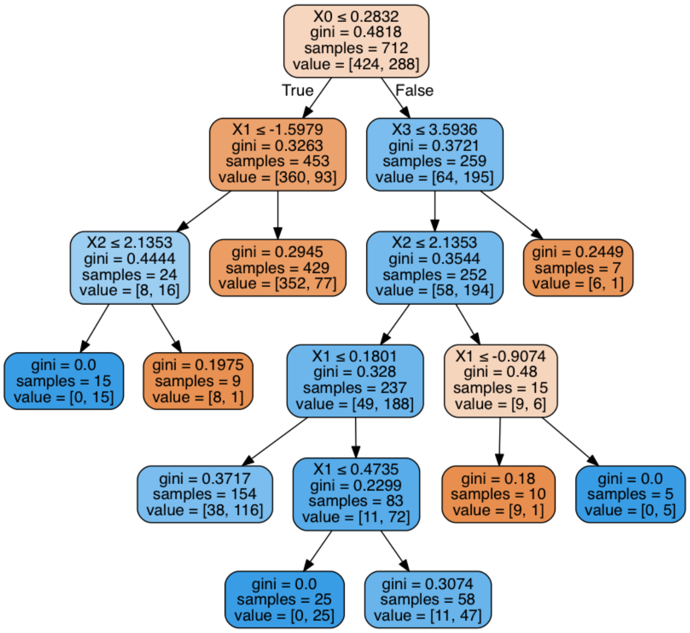
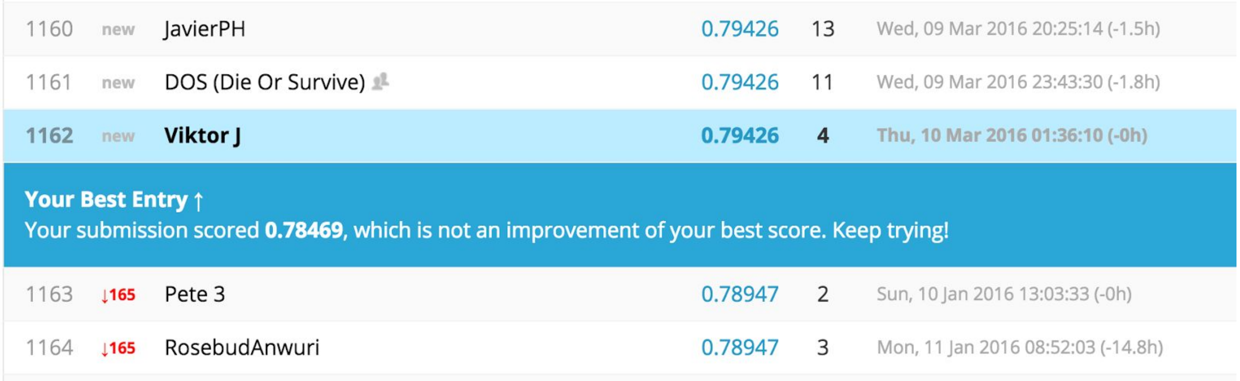


Figure 6 - Best Kaggle Score



1160	new	JavierPH	0.79426	13	Wed, 09 Mar 2016 20:25:14 (-1.5h)
1161	new	DOS (Die Or Survive)	0.79426	11	Wed, 09 Mar 2016 23:43:30 (-1.8h)
1162	new	**Viktor J**	**0.79426**	**4**	**Thu, 10 Mar 2016 01:36:10 (-0h)**

Your Best Entry ↑
Your submission scored **0.78469**, which is not an improvement of your best score. Keep trying!

1163	↓165	Pete 3	0.78947	2	Sun, 10 Jan 2016 13:03:33 (-0h)
1164	↓165	RosebudAnwuri	0.78947	3	Mon, 11 Jan 2016 08:52:03 (-14.8h)