CS 4342 Term Project Report

Veronica Gurnawan, Lucas Varella and Rakesh Veetekat

Project Summary

We decided to do our term project based on the Kaggle competition *Titanic: Machine Learning from Disaster*, which predicts survivorship on the Titanic based on the provided dataset through machine learning algorithms used for predictive analysis. From research, this was a popular competition on Kaggle and had a very interesting goal (predicting who is most likely to survive on the *Titanic* based on a few factors). While we did not submit to the Kaggle competition, we did use it as inspiration for our project.

The data was split into a training set and a test set, both offered through Kaggle (and provided along with our report and source code). The only difference between the two is that the training set contains the class variable denoting if the passenger survived or not, while the other does not. Since this is a predictive analysis problem, we also wanted to estimate the accuracy of our models in addition to training the models. This was accomplished by splitting the training set given to us by Kaggle into a training set and validation set (80/20 split respectively).

For this project, given its predictive nature, we decided to use the Random Forest, KNN (*k* nearest neighbors, k = 3) and linear SVM (support vector machine) classifiers.

Data

The structure for the dataset (training + test) is as follows:

Variable	**Definition**	**Key**
survival* *only the test set	Whether or not the passenger survives	0 = No, 1 = Yes
pclass	Class of the ticket (first, second, etc.)	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex of passenger	
Age	Age of passenger (years)	
sibsp	# of siblings and spouses aboard the Titanic	
parch	# of parents/children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	

embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Preprocessing Steps

As evident from our source code file, we had a number of preprocessing and data analysis techniques done in order to learn as much as possible about our data set and the problem given to us.

Our data analysis included examining these relationships:

1. Gender vs. Survived (comparing the women who survived versus the men who survive)
2. Class vs. Survived (comparing who survived across the classes)
3. Sibling/Spouse vs. Survived (examining the number of siblings/spouses aboard on the *Titanic* and its relation to survivors)
4. Parent/Child vs. Survived (examining the number of parents/children aboard and its relation to survivors)

Once we completed our initial data analysis, we went on to preprocessing and some more data analysis. Here were our steps and some of our findings:

1. Using .describe() on the 'Age' column in the training set, we determined that there were 100 missing values (unaccounted for passengers). We also determined the distribution of ages aboard the *Titanic.*
2. We discretized the age column into: Missing Values, Baby, Toddler, Kid, Teen, YA, and Adult. We also created a histogram to visualize the amount of surviving people in each age bucket. We found that while the median age on the ship was about 30 years old, with 75% of people being around 38 years old, the majority age demographic of surviving passengers were babies and toddlers. Seniors survived the least of all the demographics.
3. Using dummies, we converted the Pclass, Sex, SibSp, Parch and Age_Bins (categorical variables) into dummy/indicator variables.
4. We then split the training set into training and validation sets (80/20 split). This will be used later to validate our accuracy for each model.

Machine Learning Algorithms

Random Forest Classifier

The first machine learning technique we examined was a random forest classifier, which is an ensemble tree-based learning algorithm composed of decision trees that are randomly selected from a subset of the training dataset. The classifier collects the decisions from each of the trees to determine its prediction for the class of each test observation.

We created our random forest classifier with a max depth of 5 levels and 100 estimators. The max depth refers to the maximum depth for each tree, and the number of estimators indicates how many trees should populate the random forest. The classifier was run on both the

validation and testing datasets, and both the accuracy and validation score were recorded after applying the model to each dataset. The accuracy of running the model on the validation set was 80.4% with a validation score of 0.75, and the accuracy of running the model on the testing set was 82.9% with a validation score of 0.82.

KNN

The second technique we used was a K-Nearest Neighbors (KNN) classifier. The KNN method determines class membership of the input by splitting the training dataset into clusters of closely associated observations. The number of clusters to be formed is a hyperparameter and can be tuned for performance. An input is classified by the majority class membership among its nearest observations, if it were put among the original training set.

We trained our classifier with 3 nearest neighbors. The classifier was run on both the validation and testing datasets, and both the accuracy and validation score were recorded after applying the model to each dataset. The accuracy of running the model on the validation set was 80.4% with a validation score of 0.73, and the accuracy of running the model on the testing set was 83.1% with a validation score of 0.82.

Linear SVM

The third and last technique we used was a Linear Support Vector Machine (SVM) classifier. The SVM method determines class membership (between two classes) of the input by determining a boundary line in the feature space that separates observations of one class from observations belonging to the other class. A linear SVM exclusively involves a linear boundary line, but other SVMs allow many forms of nonlinear boundary lines. The boundary line itself is determined using specific points from each class that seem like they would be close to an optimal boundary line (called “support vectors”). The boundary line is then traced to be equidistant to all support vectors, as much as possible, so that it may be optimal. An input is classified by whether it falls on one side of the boundary line, or the other.

We trained our classifier with 3 nearest neighbors. The classifier was run on both the validation and testing datasets, and both the accuracy and validation score were recorded after applying the model to each dataset. The accuracy of running the model on the validation set was 76.5% with a validation score of 0.78, and the accuracy of running the model on the testing set was 80.4% with a validation score of 0.79.

Conclusion

As was noted in our source code, we found that Random Forest was the best at predicting survivorship, followed by linear SVM and then KNN. While KNN had the best from our calculations, upon checking the accuracy through Kaggle, it was actually not very accurate. This could be because Random Forest is an ensemble method (which generally performs well since it’s an aggregation of methods), whereas the KNN we implemented was quite basic.

By discretizing the Age column into categories, we were able to improve our models through more accurate readings of the data.