

Big Data in Practice(MidTerm)

-Akshat Gaur(agaaur)

Introduction:

In this task, we are supposed to solve a Kaggle competition, Titanic: Machine Learning from disaster. In this competition, we are supposed to predict the survival of the passengers from drowning given some information about them.

Data Set: Train and test

Train data is used to build the model that will later be used to perform prediction. We are provided with the survival information (ground truth) for the samples.

Test data will be used to perform prediction. Once we train our model using train data, we predict the survival of the passengers in the test data. To get the estimate of our model, we compare the result with true labels for these samples and calculate the accuracy.

Feature Set:

The feature list provides the information about the passenger which may or may not help in predicting the survival of the passenger. Using these feature list and some new features generated by feature modeling we can predict the chances of survival the sinking of titanic.

Some information related to the features provided:

Data Dictionary

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / 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

Variable Notes

pclass: A proxy for socio-economic status (SES)

1st = Upper

2nd = Middle

3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...

Sibling = brother, sister, stepbrother, stepsister

Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way...

Parent = mother, father

Child = daughter, son, stepdaughter, stepson

Some children travelled only with a nanny, therefore parch=0 for them.

Methods & approach:

The two approaches that were used to train the model are: Random Forest and KNN. This task was to be accomplished over two iterations by using Random Forest and KNN.

First Iteration (Baseline):

In first iteration, we were supposed to perform prediction of survival of passenger by directly using the train data as it is to get a baseline accuracy. In this iteration, no feature modelling was involved. The only modification that was done in this iteration was to deal empty values. For the algorithm I used, it considers blanks to be empty string and classify all of them as same. This is not a appropriate approach but for baseline code to work we can just use this. Also, numeric features were replaced with zero values.

Second Iteration (Feature Engineering):

In this iteration, some feature engineering was performed on the train data set to improve the results. Some approaches that were followed to do feature modelling include:

1. **Filling NaN:** In Some samples, few of the features had a blank value assigned to them. Filling the values with appropriate value gives better information about the data set.
2. **Feature Selection:** Some features provided redundant information or were of no use so they were dropped while creating the model for prediction.
3. **Feature Construction:** Some of the features provided in the train data did not provide much insight about the data but we could still use those features to create new features that provide much more information for predicting survival.
4. **Feature Modification:** Some features were transformed as using them as it is of less worth. They did not provide much information that could have helped in predicting survival of passengers.

KNN:

In KNN, to calculate the distance of a sample form other samples I used Euclidean Distance. To make this approach work efficiently I took care of two most important things. Firstly, I normalized my numeric features so that they do not have more effect on my result compared to other features. Also, I one hot encoded categorical features for my KNN to handle them. For the first iteration, I considered if strings are equal distance is zero else 1 and also did not normalize the data set. So, effect of feature engineering is more in KNN compared to Random Forest.

Random Forest:

In Random Forest, I selected 6 features (sq. root number of features) to generate a tree and randomly picked these 6 features. Also, I used to $2/3^{\text{rd}}$ of the train set to create each tree. Since we have many features in both the iterations (10 in first and 13 in the second) so creating 21 trees can give enough trees for reducing overfitting errors. Effect of feature engineering is less on Random Forest as it already handles it by choosing few of the features randomly and also takes care of overfitting. Also, it can somewhat handle categorical features. Still we can see some improvement as some of the features had NaN values or were sparse so filling them gave better insight.

Features selected:

Some of the features in the data set do not provide any useful insight about the data or they cannot be used as it is and some of them also provided redundant information. So, in our problem of predicting survival chances of the passengers on Titanic we had to perform feature selection. Some approaches that were followed to do feature modelling include:

1. **Filling NaN:** In some samples, few of the features had a blank value assigned to them. Filling the values with appropriate values gives better information about the data set. For feature Age, which is a numeric continuous feature, we used the median value to replace the missing value. For feature Cabin, which is a sparse feature and we do not have much information to fill the blanks, we filled all the blank values with a new value 'cabinless'. For feature Embarked, which is a categorical feature, I filled the missing value with the mode value as the chances of boarding from the mode value are higher for the passenger.
2. **Feature Selection:** Some features provided redundant information or were of no use so they were dropped while creating the model for prediction. Feature "passenger ID" is of no use as all the passengers will have a unique passenger ID so this feature can be surely dropped. Similarly, a feature like "Name" is unique for the passenger so we drop that feature. Features "SibSp" and "Parch" are dropped as they provide redundant information about the passenger i.e. whether the passenger had any family member on the ship. So, we can create a new feature from it and drop these two features.
3. **Feature Construction:** Some of the features provided in the train data did not provide much insight about the data but we could still use those features to create new features that provide much more information for predicting survival. A feature like Name is not useful as all the passengers will have different names and so it cannot help in prediction if used as it is. But creating a new feature "Family name" by extracting the last name of each passenger gives much more information as if a person survived there are higher chances of survival of his family as well. Also, creating "prefix" from the Name feature gives us better information as girls/ladies or people from higher society are given higher preference. I also created a feature called "Family" which was a categorical feature with values 0/1. This feature was created from features "SibSp" and "Parch". These two features can be combined to get better results.
4. **Feature Modification:** Some features were transformed as using them as is is of less worth. A feature like Ticket which had alphanumeric values was converted into only numeric by dropping the alphabets from the beginning. It improved the results as considering ticket as a string is of no use as it will be different for all the passengers but converting it into numeric can provide some information about the location of the passenger on the Titanic and help in improving prediction. Categorical features like Embarked, Pclass and Sex were dummy coded/one-hot encoded to convert them into numeric data so that KNN can perform more efficiently on them. Also, feature normalization was done on features like Age and Fare so that it does not have an advantage over other features while calculating Euclidean Distance.

Results

Iteration 1: Baseline

KNN

Accuracy: 0.626794258373

Confusion Matrix

```
[[241 19]
```

```
[137 21]]
```

Precision Score: 0.595017214754

Recall Score: 0.626794258373

Misclassification rate: 0.373205741627

F1 score: 0.550099243549

Random Forest

Accuracy: 0.763157894737

Confusion Matrix

```
[[211 49]
```

```
[ 50 108]]
```

Precision Score: 0.762869542991

Recall Score: 0.763157894737

Misclassification rate: 0.23684210526

F1 score: 0.763009250574

Iteration 2: Feature Modelling

KNN

Accuracy: 0.774641148325

Confusion Matrix

```
[[233 27]
```

```
[ 69 89]]
```

Precision Score: 0.763541248958

Recall Score: 0.774641148325

Misclassification rate: 0.225358851675

F1 score: 0.761862423133

Random Forest

Accuracy: 0.799043062201

Confusion Matrix

```
[[230 30]
```

```
[ 54 104]]
```

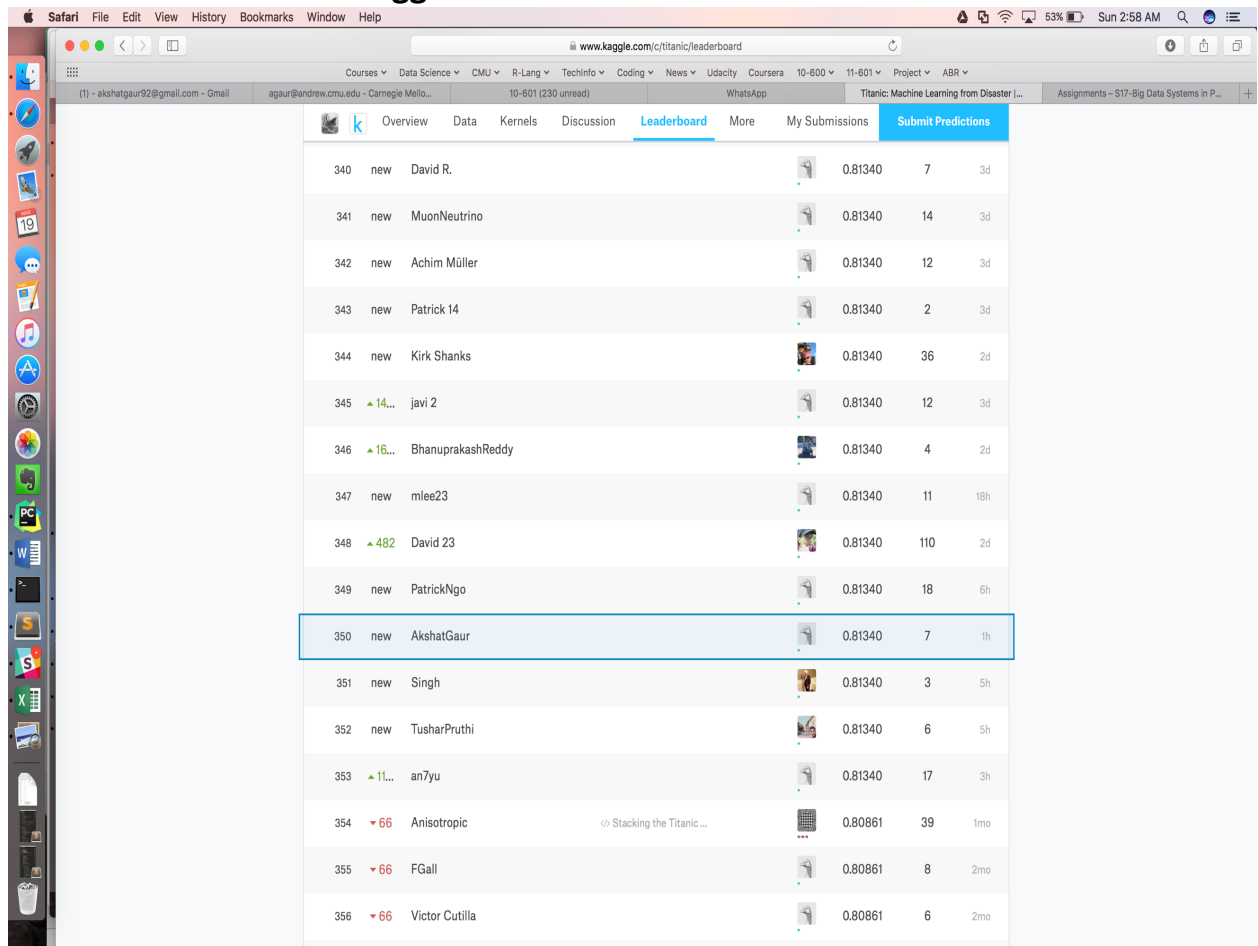
Precision Score: 0.797105851563

Recall Score: 0.799043062201

Misclassification rate: 0.200956937799

F1 score: 0.795217431536

Screenshot of rank on kaggle



340	new	David R.		0.81340	7	3d		
341	new	MuonNeutrino		0.81340	14	3d		
342	new	Achim Müller		0.81340	12	3d		
343	new	Patrick 14		0.81340	2	3d		
344	new	Kirk Shanks		0.81340	36	2d		
345	▲14...	javi 2		0.81340	12	3d		
346	▲16...	BhanuprakashReddy		0.81340	4	2d		
347	new	mlee23		0.81340	11	18h		
348	▲482	David 23		0.81340	110	2d		
349	new	PatrickNgo		0.81340	18	6h		
350	new	AkshatGaur		0.81340	7	1h		
351	new	Singh		0.81340	3	5h		
352	new	TusharPruthi		0.81340	6	5h		
353	▲11...	an7yu		0.81340	17	3h		
354	▼66	Anisotropic	⌂ Stacking the Titanic ...	0.80861	39	1mo		
355	▼66	FGall		0.80861	8	2mo		
356	▼66	Victor Cutilla		0.80861	6	2mo		

Conclusion:

On performing feature engineering the accuracy of predicting survival increased. The results became more accurate. This shows that some features in the data set provided were not very relevant and some of them need to be transformed in order to provide more precise label. Doing some feature engineering to improve the data/information we have can get better insights about the data which can help in providing more accurate predictions. This we saw in this Titanic data as well.

KNN:

For KNN the most important part is to perform normalization of all the features. If we do not do that some features have more effect on the result compared to others as we were calculating Euclidean Distance for numeric features and 0/1 value for categorical/string features. Also, performing one hot encoding improves the results as we convert categorical features into numeric value which can be better handled by KNN. This is why we see high increase in accuracy in second iteration. Moreover, KNN considers all features which in our case results into worse results as we have many features which do not provide much help in prediction.

Random Forest:

The Random forest classifier performs better than KNN in first iteration as KNN cannot properly handle Categorical data and also because feature normalization was not done. We get a good accuracy score for random forest for the first iteration itself as it can handle categorical features. It gives better result than KNN because we choose features randomly out of over feature set and considers best (with max information gain) to perform division of the tree. It also overcomes the problem of overfitting train data by creating multiple trees. Performing feature modelling on train data did improve our results as some of the features had empty values and some of the features were not providing any information. We also added new few features to get more information for predicting. These modelling methods improved the accuracy of the prediction to some extent.

How to run the code on Aspen

The Complete Process for Random Forest is divided into three parts.

1. Preprocessing of Data

In preprocessing I have created train file for each mapper function. These train files have 2/3rd of train data and 6 features.

These train files should be pushed to hdfs directory. Along with these train file we also need to push test file to hdfs for mapper.

2. Random Forest over MapReduce

In this part, multiple trees are created over each mapper node.

These mappers also have the test data and so perform the prediction using the tree created.

In reducer, the predictions given by all the mappers are used to find the best predicted label.

3. Accuracy and other metric calculation

Finally calculate the accuracy by calculating the misclassification and also calculate other metrics.

1. Preprocessing of Data:

a. Execute python script CreateInputFiles.py

```
python CreateInputFiles.py <datasource_file> <Num of trees> <hdfs directory path>
```

```
python CreateInputFiles.py train.csv test.csv 21 hdfs://aspen.local/user/agaur/Titanic/
```

b. After this push all the trainFile_<i></i>.csv to hdfs to the path provided above.

Push the test file to the same hdfs directory.

2. Random Forest over MapReduce

a. Execute RandomForest.py script as follows

```
python RandomForest.py <input file for mapper> -r hadoop > result
```

This input file for mapper is created by previous script so it will be below value only.

```
python RandomForest.py mapper_input_file.txt -r hadoop > result
```

3. Accuracy and other metrics

a. Execute accuracy.py script as follows

```
python accuracy.py <predictions> <true_labels>
```

```
python accuracy.py result labels
```

The Complete Process for KNN is divided into three parts.

1. Preprocessing:

a. Execute Main.py

```
python main.py <train filepath> <test filepath>
```

2. KNN over MapReduce:

a. Execute KNN_MR.py

```
python KNN.py --test_path <test file path> --output_path <outfilepath> --k <value of K>
```

3. Accuracy and other metrics

a. Execute accuracy.py script as follows

```
python accuracy.py <predictions> <true_labels>
```

```
python accuracy.py result labels
```