CPS844
Professor Cherie Ding
March 30th, 2021

Nicolas de la Guardia
Alifiyah Hussain

Titanic Data Mining

Titanic - Machine Learning from Disaster is a Kaggle exercise designed as an introduction to dataset analysis, machine learning and feature engineering. The goal is: given a dataset with ~1400 Titanic passengers and multiple attributes, to predict which survived and which did not.

There are 12 attributes in the dataset:

- PassangerId, Name, Pclass, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked, Survived

Description of attributes:

- Survival : 0 = No, 1 = Yes (nominal, class)
- Pclass: Ticket class: 1 = 1st, 2 = 2nd, 3 = 3rd (nominal)
- Sex = Male or Female (nominal)
- Age = Age in years (numeric)
- Sibsp: # of siblings / spouses aboard the Titanic (numeric)
- Parch: # of parents / children aboard the Titanic (numeric)
- Ticket: Ticket number (numeric)
- Fare: Passenger fare (numeric)
- Cabin: Cabin number (nominal)
- Embarked: Port of Embarkation: C = Cherbourg, Q = Queenstown, S = Southampton (nominal)

Our approach to this problem was the following:

1. Separate ~20% of total instances into a different dataset (test set). This will only be used for testing at the very end once the model is fully trained.
2. The other ~80% of data is our training set, which will be fed to the model and cross validated. This step is the bulk and is where the model is trained. Training the model also implies choosing the best hyperparameters for the algorithm (max depth, bag size etc) and feature engineering. The latter consists of choosing/discarding attributes based on how much they benefit the model. Some attributes can be noise and should be discarded, or maybe there's information inside the noise which could be extracted from it. The idea is to optimize our data to be able to get the best predictions form the classifier.
3. After all optimization is done, the model should be tested on data it has never seen. This is where the initial test set comes into play. This test set should only be used once and the accuracy returned should be considered final.

1. Testing the Algorithms

OneR

With original dataset:
=== Stratified 10 fold cross-validation ===
Correctly Classified Instances 613 68.7991 %
Incorrectly Classified Instances 278 31.2009 %

The rule created was created using only the ticket attribute, the tickets vary considerably and therefore do not form an effective generalization, some instances in the training data may not even possess an attribute value that could be present in the test set, therefore the accuracy of this rule is fairly low.

Running it with the ADABoost M1 resulted in slightly less accuracy, which makes sense because if the data is overfitted and nonrepresentative of the test set, then increasing weight of misclassified instances for the next iterations does not help.

Correctly Classified Instances 611 68.5746 %
Incorrectly Classified Instances 280 31.4254 %

Modified dataset:
After removing the ticket attribute, the rule produced by 1R used the sex attribute, this gave the results:
Correctly Classified Instances 701 78.6756 %
Incorrectly Classified Instances 190 21.3244 %

This is a 10% increase and it makes sense because more women survived than men, and it clearly creates a better generalization for the test set
With the AdaBoost the 1R rule produced a rule using the passenger ID this lowered the accuracy:
Correctly Classified Instances 659 73.9618 %
Incorrectly Classified Instances 232 26.0382 %

This also makes sense because the algorithm tried to discretize the values to make sense of them as they are numeric, however this intuitively does not make sense because the quantitative differences between passenger IDs do not actually have any meaning, thus making a rule off of this attribute is not effective.

Logistic Regression

=== Stratified 10 fold cross-validation ===

Correctly Classified Instances 660 74.0741 %
Incorrectly Classified Instances 231 25.9259 %

This took a very long time, so not a feasible approach, we change the cross validation to 5 and then 3 folds, **3 folds** actually increased the accuracy:
Correctly Classified Instances 689 77.3288 %
Incorrectly Classified Instances 202 22.6712 %

However, this still took a significant amount of time, and a lot of it can be accredited to the Ticket and Cabin attributes. **Removing Cabin**:
Correctly Classified Instances 704 79.0123 %
Incorrectly Classified Instances 187 20.9877 %

Removing Ticket and Cabin:
Correctly Classified Instances 709 79.5735 %
Incorrectly Classified Instances 182 20.4265 %

Any other combinations/removals of attributes resulted in less accuracy, even after using the selected attribute subsets suggested by CfsSubsetEval with the Greedy method and the BestFirst

Naive Bayes

=== Stratified 10 fold cross-validation ===
Correctly Classified Instances 710 79.6857 %
Incorrectly Classified Instances 181 20.3143 %

=== Confusion Matrix ===

```
  a   b   <-- classified as
 498  51 |   a = 0
 130 212 |   b = 1
```

Observation: Here we can see that class b (survived) is misclassified almost 40% of the time, since Naive Bayes is sensitive to redundant data, below is a series of steps in attempt to try to remove ones perceived to be redundant:

Only without attribute Embarked:
Correctly Classified Instances 712 79.9102 %
Incorrectly Classified Instances 179 20.0898 %

Only without attribute Passenger ID:
Correctly Classified Instances 713 80.0224 %
Incorrectly Classified Instances 178 19.9776 %

Without both aforementioned attributes:

Correctly Classified Instances	713	80.0224 %
Incorrectly Classified Instances	178	19.9776 %

Without Siblings:

Correctly Classified Instances	715	80.2469 %
Incorrectly Classified Instances	176	19.7531 %

Without Parent:

Correctly Classified Instances	717	80.4714 %
Incorrectly Classified Instances	174	19.5286 %

Observation: We removed Embarked and Passenger ID because it seemed like redundant information in that there are other indicators such as Fare that conclude a person was on this ship, it only slightly increased accuracy. Then siblings and parents also seemed redundant, removing those also only resulted in slight increases. It can be concluded that these attributes may simply be irrelevant as opposed to redundant as irrelevance does not decrease the accuracy in the naive bayes algorithm.

J48

Model with original dataset:

```
J48 pruned tree
------------------
Sex = male
|   Parch <= 0: 0 (484.0/80.0)
|   Parch > 0
|   |   Age <= 3: 1 (19.69/6.27)
|   |   Age > 3: 0 (73.31/15.58)
Sex = female: 1 (314.0/81.0)
Number of Leaves  :     4
Size of the tree :      7
```

Accuracy with original dataset:

Correctly Classified Instances	690	77.4411 %
Incorrectly Classified Instances	201	22.5589 %

(using AdaBoost did not increase accuracy here)

Accuracy with modified dataset:

Without ticket

Correctly Classified Instances	713	80.0224 %

Incorrectly Classified Instances 178 19.9776 %

Without ticket and passenger ID because of reasons described in 1R:
Correctly Classified Instances 717 80.4714 %
Incorrectly Classified Instances 174 19.5286 %

Observation: 0.40% increase
Resulting tree:

```
Sex = male
|  Parch <= 0: 0 (484.0/80.0)
|  Parch > 0
|  |  Age <= 3: 1 (19.69/6.27)
|  |  Age > 3: 0 (73.31/15.58)
Sex = female
|  Pclass = 1: 1 (94.0/3.0)
|  Pclass = 2: 1 (76.0/6.0)
|  Pclass = 3
|  |  Fare <= 23.25
|  |  |  Embarked = S
|  |  |  |  Parch <= 0: 0 (45.0/20.0)
|  |  |  |  Parch > 0
|  |  |  |  |  Parch <= 1
|  |  |  |  |  |  SibSp <= 2: 1 (10.0/3.0)
|  |  |  |  |  |  SibSp > 2: 0 (2.0)
|  |  |  |  |  Parch > 1
|  |  |  |  |  |  Age <= 28: 1 (4.0)
|  |  |  |  |  |  Age > 28: 0 (2.0)
|  |  |  Embarked = C
|  |  |  |  Fare <= 15.2458
|  |  |  |  |  Fare <= 13.8625: 1 (6.0)
|  |  |  |  |  Fare > 13.8625: 0 (10.0/2.0)
|  |  |  |  Fare > 15.2458: 1 (7.0)
|  |  |  Embarked = Q
|  |  |  |  Parch <= 0
|  |  |  |  |  Fare <= 7.65: 0 (2.0)
|  |  |  |  |  Fare > 7.65: 1 (27.0/4.0)
|  |  |  |  Parch > 0: 0 (2.0)
|  |  Fare > 23.25: 0 (27.0/3.0)
```

Here, once again looking at this tree, it seems that **Embarked** is overfitting the tree, making it less general, **removing** this value gives:
Correctly Classified Instances 721 80.9203 %
Incorrectly Classified Instances 170 19.0797 %

Observation: a roughly 0.5% increase in accuracy
Resulting tree:
Sex = male
| Parch <= 0: 0 (484.0/80.0)
| Parch > 0
| | Age <= 3: 1 (19.69/6.27)
| | Age > 3: 0 (73.31/15.58)
Sex = female
| Pclass = 1: 1 (94.0/3.0)
| Pclass = 2: 1 (76.0/6.0)
| Pclass = 3
| | Fare <= 23.25: 1 (117.0/48.0)
| | Fare > 23.25: 0 (27.0/3.0)

Here, it seemed as though PClass and Fare would be redundant, it seemed this would ruin the model, however removing either/or reduced the accuracy, and same happened when attempting to remove any other attribute.. Thus 80.9% is the highest accuracy for J48 in this report.

Random Forest

=== Stratified 10 fold cross-validation ===

Correctly Classified Instances	684	76.7677 %
Incorrectly Classified Instances	207	23.2323 %

Run time very long, made it 3 fold:

Correctly Classified Instances	713	80.0224 %
Incorrectly Classified Instances	178	19.9776 %

Removed ticket:

Correctly Classified Instances	730	81.9304 %
Incorrectly Classified Instances	161	18.0696 %

Removed ticket, passengerID, Cabin:

Correctly Classified Instances	735	82.4916 %
Incorrectly Classified Instances	156	17.5084 %

Removed ticket, passengerID, siblings:

Correctly Classified Instances	738	82.8283 %
Incorrectly Classified Instances	153	17.1717 %

2. Analysis of Attributes:

There are 12 attributes which include:

- PassangerId, Name, Pclass, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked, Survived

Initial Ranking of Attributes

- Using InfoGainAttributeEval with Ranker search method

```
=== Attribute selection 5 fold cross-validation (stratified), seed: 1 ===

average merit      average rank  attribute
 0.961 +- 0          1   +- 0       3 Name
 0.846 +- 0.005      2   +- 0       8 Ticket
 0.218 +- 0.005      3   +- 0       4 Sex
 0.097 +- 0.006      4   +- 0       9 Fare
 0.084 +- 0.008      5   +- 0       2 Pclass
 0.03  +- 0.003      6.6 +- 0.49   10 Cabin
 0.029 +- 0.005      6.6 +- 0.8     6 SibSp
 0.021 +- 0.006      8.4 +- 1.02   11 Embarked
 0.016 +- 0.009      9   +- 0.89    7 Parch
 0.014 +- 0.007      9.4 +- 0.49    5 Age
 0     +- 0         11   +- 0       1 PassengerId
```

- The ranker values the Name and Ticket attributes as the most significant.

Feature Engineering

In order to keep the relevant attributes and get rid of the noise, we noticed some attributes could be either split into different parts or some info could be extracted from them. Sometimes this simplified data could be more relevant than the whole attribute.

To check for relevant attributes, we used a bottom up approach mixed with bidirectional depending on observation. We walk through the process of elimination below.

Missing Values

Age has some missing values

- We set missing ages to the average of all ages = 29.6 ~= 30

Cabin has a large amount of missing values, we either delete it or try to extract some information from it. Before deleting it we split it into two columns, letter and number, it might allow the algorithm to make better predictions since maybe the letter or numeral are more important independently

- Ex. C103 -> C and 103
- It didnt seem to make a difference for the accuracy, any of the 2 attributes, or any of them combined. This is most likely due to the large amount of missing data

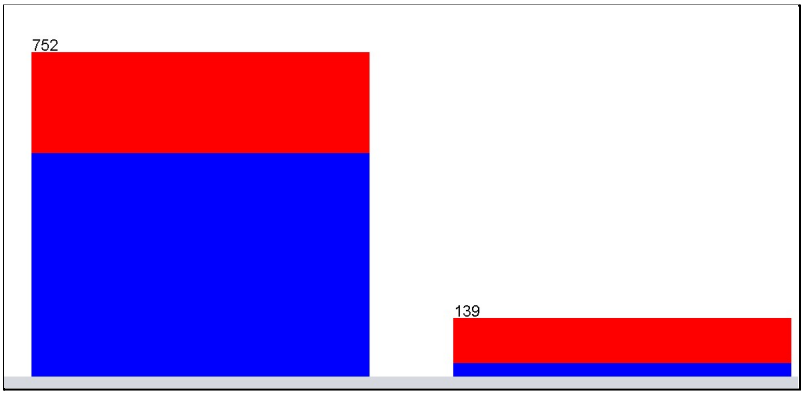
We also combine Parch and SibSp into a new column Company which determines the family size for each instance. We did this in order to see if size would increase their chances of survival or act as a hindrance since they have to look out for each other as opposed to simply saving themselves.

- Parch (2) + SibSp (1) = Company (3)
- Company seemed to make no significant difference compared to the separate attributes.

Attribute Testing

Name attribute decreases accuracy as far as we can tell, but we noticed some names had an **alias** in parentheses. We made a new boolean column indicating whether the name of that instance had an alias or not (1/0)

- It resulted to not have much impact on the score, even though the graph would seem to indicate otherwise. This might be due to the small amount of aliases present in the dataset, making most instances alias = 0.

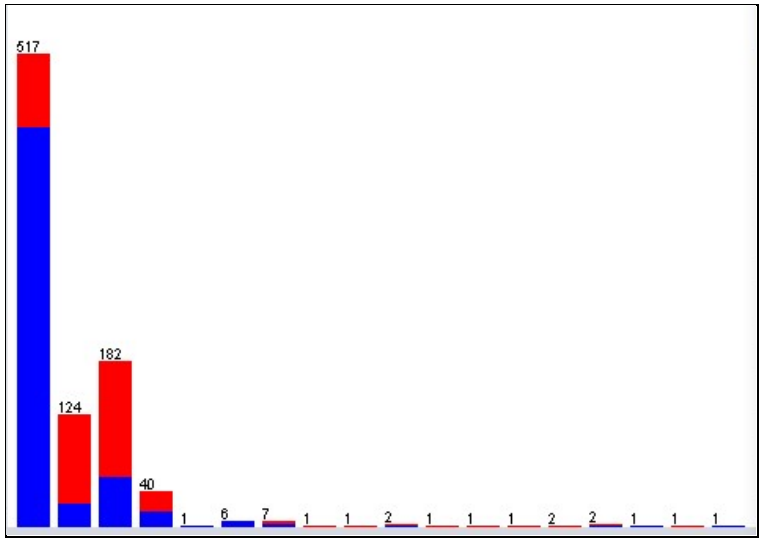


No alias (left), alias(right)

Red = Survived
Blue = Not survived

    - Percentage of death seems much higher when there is no alias, possibly correlating the presence of an alias with the importance of the person and therefore survival rate.

- Another attribute derived from the Name attribute is Title. Using regex we extracted the titles (Mr, Mrs, Dr. etc.) from the name attribute. This leads to an interesting graph.



Bars order: Mr, Mrs, Miss, Master…

We can see that only ~20% of the Mr's survived

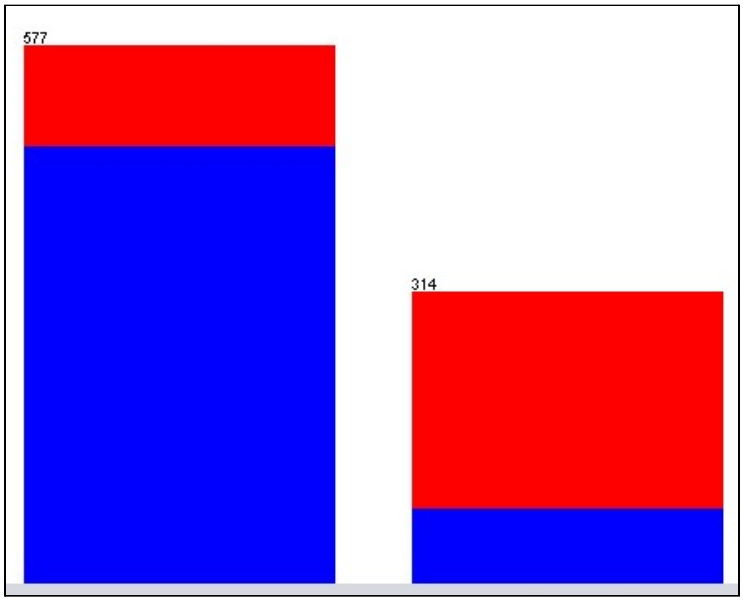
~80% of the Mrs' and ~70% of the Misses survived

After removing **Name** and using feature selection (CfsSubsetEval algorithm with BestFirst search method) on the remaining attributes we get the two most important attributes:

- Ticket
- Sex

Sex makes sense. Statistically, more women survived because women and kids were prioritized for lifeboats.

Sex/Survival Graph



Red = Survived 1 = Survived
Blue = Survived 0 = Died

Men (577), around 30% survived
Women (314), around 70% survived

On the other hand, Ticket being in the top 2 most important attributes seemed a bit odd. There seems to be no apparent pattern in the tickets or hierarchy in the numbers, the format is

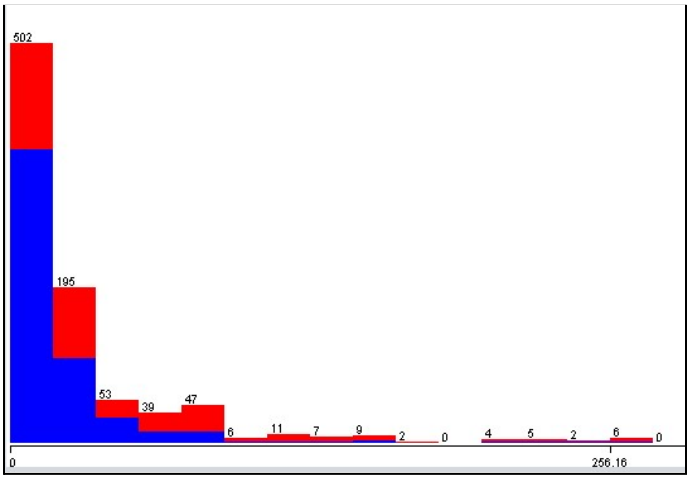
not even standard. Since there are so many tickets, it seems like our algorithm is overfitting to our training set.

- Examples of tickets are: (A/5 21171, PC 17599, STON/O2. , 3101282, 113803, 373450, 330877, PP 9549)
- If the ticket attribute is removed, the accuracy in some algorithms increases by as much as 10%. This most likely means that this attribute is just noise.

Removed "Ticket" Attribute

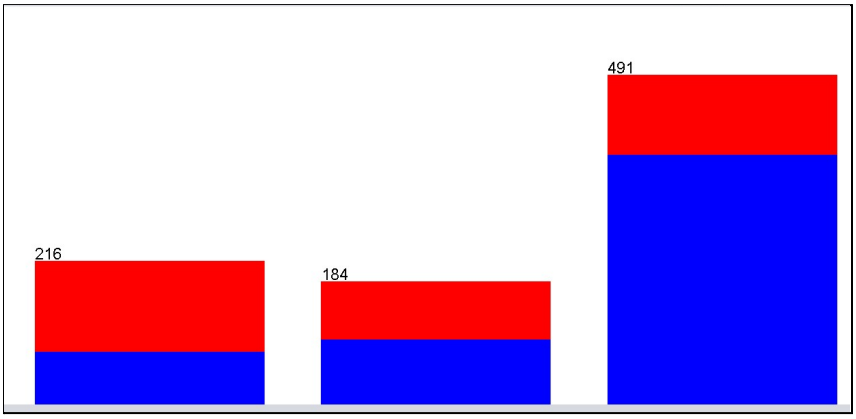
- After running feature selection (InfoGainAttributeEval with Ranker) again on the remaining attributes, these are the highest ranked
    - Title 0.257 (graph shown earlier)
    - Sex 0.218
    - Fare 0.097
    - Pclass 0.084
    - Alias 0.083 (shown earlier as well)
    - Company 0.061

Fare/Survival Graph

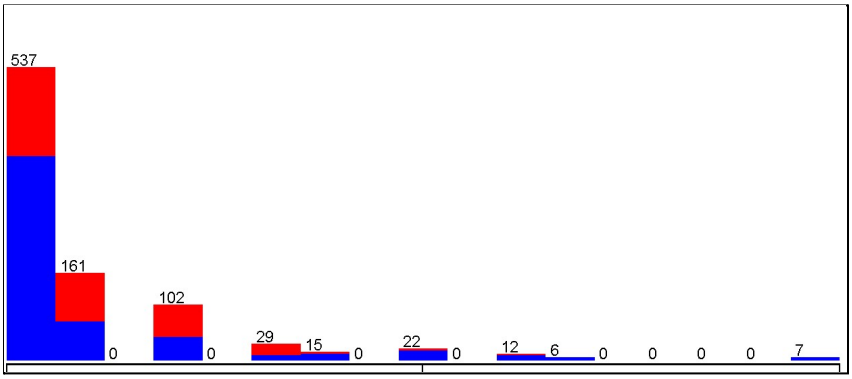


- Even though it is not normalized, we can see the % of survivability increasing as we climb to the more expensive fares. Initially (rough estimate) it seems around 40% survival, then 50%, then 60% etc, increasing as the fares cost rise.
    - That trend seems general except in one *fare* range, between $35-55 (the third bar in the graph) where survivability decreases slightly.

Pclass / Survival Graph



Company / Survival Graph



After removing Name and Ticket, we continued tweaking the attributes and considering how much they affect the algorithm. We mapped the variance in a spreadsheet where we were measuring the accuracy when removing/adding one attribute at a time under the same algorithm and same conditions. The image below corresponds to the spreadsheet with the accuracy for the filled in attributes on that row. We did not see any significant change besides removing Name and Ticket.

Model	Passenger	Pclass	Name	Title	Alias	Sex	Age	SibSp	Parch	Company	Ticket	Fare	Cabin	Cletter	Cnum	Embarked	Accuracy
RandomForest																	73.8496
5 Folds																	79.0123
0 Max depth																	79.798
																	79.349
																	81.7059
																	80.4714
																	81.1448
																	81.3692
																	80.0224
																	81.59%
																	82.4916
																	81.9304
																	78.5634
																	82.0426
																	81.0325
																	81.93
																	82.1549

Comparing Algorithms

In part 1 our highest accuracies came from the tree models, J48 and Random Forest, with Random Forest producing the higher accuracy, by about 2.5%. This seems to make sense because these models filter noise and irrelevant data in a stepwise fashion and that makes sense with our dataset because not everything is particularly important or highly correlated to each other. It is a dataset which requires more of a logical sequencing "if this exists, then this". For part 2 we did more feature engineering with the selected attributes we concluded were giving us better accuracies in part 1 and ran it with the Random Forest algorithm, this gave us the best result, the process went as follows:

Hyperparameters:

- Cross-validation 10 folds
- Unlimited depth

Accuracy with modified dataset:

- Removed Ticket attribute
    - Accuracy: 82.3793 %
- Removed Ticket, PassengerId
    - Accuracy: 83.165 %
    - Hyperparam:
        - Max Depth: changed to 12
            - Accuracy: 83.7262 %
    - Changed cross-validation 10 -> 8 fold
        - Accuracy: 83.9506 %
    - Combined Parch and SibSp -> Company
        - Accuracy: **84.0629 %**

It can be seen that the feature engineering and selecting the relevant attributes, filling in and removing missing values significantly improves accuracy.

Conclusion and Further Thoughts

After finishing this project and learning more in depth about the models, feature engineering and our data set, we have come up with possible attributes which could help find a stronger correlation., These could be:

- Instead of setting missing values in Age to the average of ALL passengers, we could use more precise averages. Ex. If a female passenger is missing Age, set it to the average of all females, or better yet, the average of all Miss or Mrs. (title attribute). This could give a more precise approximation.
- Rather than only having family size, we could have another attribute which determines if the person is alone. If Company == 0, isAlone = True.
- Discretizing the age attribute into Children, Teenage, Adult and Elder.
- Splitting the Fare attribute into those paying average or higher than average