Titanic report

group 27

20 April 2018

Kaggle Competition: Titanic: Machine Learning from Disaster

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

The aim of the competition is to predict who among the passengers and crew was more likely to survive than others. Kaggle provides two datasets: *train* and *test*. While both datasets reference to passengers details, only *train* dataset contains infromation if passenger survived or not. Our goal is to predict which passenger from *test* dataset survived the sinking of Titanic.

Preparation

As mentioned before, Kaggle has provided two seperate datasets: *train* and *test*. Both datasets contain details about passengers and their trip. The only defference between them is *Survived* column in *train* dataset that indicates if passenger has survived the Disaster. That dataset will be used to train our models.

Before attempting to perform predicions, we focused on given data and tried to retrieve more interesting facts based on Feature Engineering. Because the only column that differ is *Survival*, for further processing we decided to datane both datasets into big one. Such an approach allows us to perform more adequate data analyse as we have a full insight of traveling passengers.

Data exploration

To have a better insight into assignment, we had to explore given data. That's the most important step - we have to be aware of all the details to work efficiently. Whole dataned dataset contains 1309 records (passengers) with 12 variables. In this part we will take a closer look to every attribute.

In datasets we can distinguish several (12) columns:

- Survived - indicated if given passenged survived
- PassengerId - passenger index in dataset
- Pclass - the ticket class (1,2,3)
- Name - full name of passenger, including their title
- Sex - sex of passenger
- Age - age of passenger
- SibSp - number of siblings or spouses traveling with passenger
- Parch - number of parents or childern traveling with passenger
- Ticket - ticket number
- Fare - passenger fare

- Cabin - passenger's cabin number
- Embarked - port of embarkation (C = Cherbourg, Q - Queenstown, S = Southampton)

Lets make a closer look into several variables.

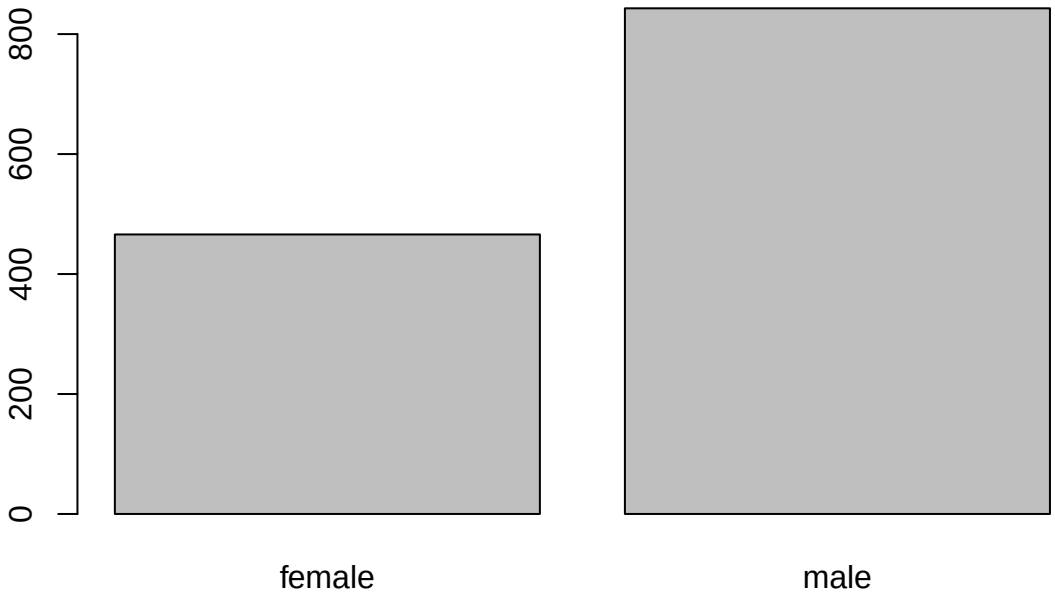
Name

In given dataset we can see that *Name* attribute contains string with passenger's name, surname and title.

example: *Allison, Master. Hudson Trevor*

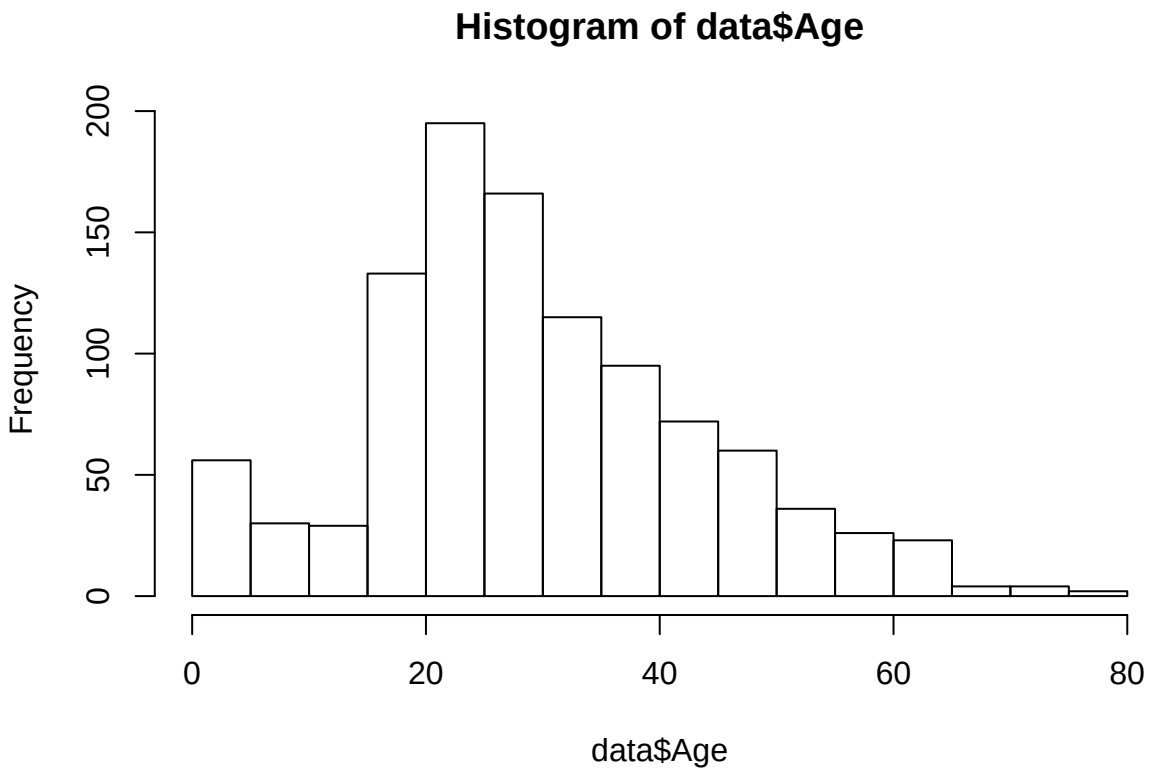
Fortunately, all rows in *Name* column follow the same string pattern (*surname*, *title first name*). Thanks to this fact, we will be able to retrieve more additional infrmation about passengers, like common surnames or titles.

Sex



Investigating Sex attribute we can see that there were 466 females and 843 males onboard. That gives us the first easy grouping of passengers.

Age



Regarding Age attribute, we can see that this variable varies up to 80 with mean around 23

Fare

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.000   7.896  14.454  33.295  31.275 512.329       1
```

Feature Engineering

After investigation of given dataset we can distingush columns that seem to be useful for further processing to retrieve even more data. In such an approach we are able to create additional columns with relevant variables that sould result in better prediciton accuracy.

Feature: Title

As mentioned before, Name column contains not only name and surname of passenger but also a title (like Sir., Mr., Mrs., ...). Following common pattern (*surname*, *title first name*) we can retrieve additional Column in our dataset that would group our passenger by Title. In addition, groups of unique similar titles were replaced by the tame variable (like 'Capt', 'Don', 'Major', 'Sir' => 'Sir').

As a result we obtained a fixed set of values:

```
##    Col     Dr   Lady Master   Miss   Mlle     Mr    Mrs     Ms    Rev
##      4      8      4     61    260      3    757    197      2      8
##    Sir
##      5
```

Feature: Family

Basing on variables *SibSp* (number of siblings or spouses), *Parch*(number of parents or child) and Surnames retrieved from *Name* variable we are able to group passengers by families. Assuming that during disaster, every person takes care about their relatives, we think that it can be a significant factor in predicitons.

Our assumptions:

- the number of relatives with who each passenger was traveling is aclculated as follows: *SibSp + Parch + 1* - result is family size
- if family size is less or equal 2 we assume that the value is not revelant and we mark sucha afamily as *n/a*

As a result we obtained Family attribute with 97 levels.

Feature: Deck

Analysing *Cabin* attribute we figured out that each cabin number consists of Deck Level and Room number (like C40 => Deck C, Room 40). Because Deck Level could play sigificant role in evacuation, we assumed thet it's a siginicant attribute. We decided to create a new attribute called Deck and we assigned relevant Deck Level to each passenger. Unfortunately, not every passenger had a Cabin number assigned, in such a case we marked Deck as 'U'.

Result:

```
##    A    B    C    D    E    F    G    T    U
##   22   65   94   46   41   21    5    1 1014
```

Feature: TicketType

Lokking into ticket numbers we can see that some tickets have common prefix that could refer to Ticket Type of place of purchase (example: STON/02 42342). We decided to retrieve that ticket prefix and create a new attribute for each passenger. If ticket didn't have any prefix, we marked TicketType as 'num'.

As a result we obtained TicketType factor with 51 levels.

Missing values

We have found that some records lack in Age attribute. In such a situationw e decided to use a Decision tree to predict missing Age values. As siginicant factors we marked attributes: Pclass, Sex, FamilySize, Embarked, Title, SibSp, Parch.

Also Fare column had some missing values. In such a case we replaces missing values with median of all ticket Fares.

Classification and evaluation

By analysing our data and engineering some additional features we have enriched our dataset.

Within all calumns we decided that only few of them play siginificant role in predictions.

Chosen factors: *Pclass, TicketType, Sex, Deck, Age, SibSp, Parch, Fare, Embarked, Title, FamilySize, FamilyID*

Creating a steup

To evaluate classifiers we will need to create a proper setup. In this case we decided to use *train* data from Kaggle as it contains *Survived* column. For evaluation purposes we decided to split the data for training and testing sets with ratio 70/30 (70% - training, 30% - testing). While splitting the data we based on random ordering.

For evaluation we decided to use two non-linear algorithms: k-Nearest Neighbour Classification and Conditional inference trees. Both classifiers were trained and tested with the same sets of data. For evaluation analysis we used Confusion Matrix.

Factors that we took into account:

- Accuracy - how well results were predicted
- 95 CI - confidence intervals, our final score should match into calculated intervals
- Kappa - accuracy through random predicitons
- F1 - model that takes recall and precision into account

Evaluation of k-Nearest Neighbour Classification

Accuracy : 0.6929
95% CI : (0.6338, 0.7477)
Kappa : 0.3338
F1 : 0.5638

Evaluation of Conditional inference trees

Accuracy : 0.809
95% CI : (0.7566, 0.8543)
Kappa : 0.5929
F1 : 0.7437

Kaggle Submission

For Kaggle competition we decided to use Conditional inference trees as it gives us higher results in included evaluation factors.

We have submited our Prediction in Kaggle system and obtained satisfactory result 0.82296 which is top 3% in leaderboard. This result also matches into expected Confidence Intervals calculateg during evaluation.