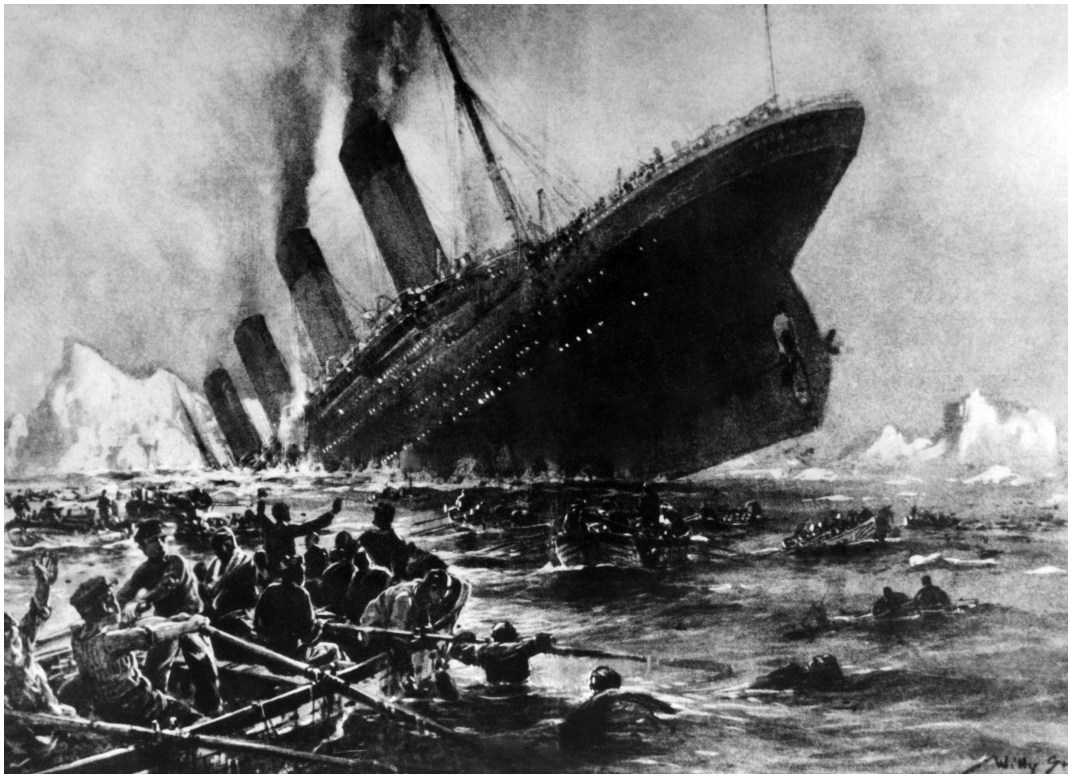
RMS Titanic: Machine Learning from Disaster (draft version)

Myrthe Boone

December 25, 2018



1 Preface

In this paper we will have a look at the passengers on board of the Titanic. We will try to analyse what sort of people were most likely to survive the disaster using machine learning techniques. Furthermore, we will make a prediction on a part of the dataset whether those passengers have survived or not with the help of particular algorithms. These algorithms are developed using the other part of our dataset. We will try to make our predictions as accurate as possible. The goal of this paper is not to make predictions about the future or about disasters in general. The results of our research may teach us something about the circumstances during the time that the Titanic sank. The passengers all played a different part in society back in those days. It teaches us something about the civilization.

Moreover, this paper is written because I wanted to learn something about machine learning and programming using Python. I want to study engineering at TU Eindhoven. It will come in handy if I already know a thing or two about programming in Python. Python is a programming language that is becoming more and more popular for things like data analysis and I am certain that I will use it more often in the future.

I would like to give a special thanks to the following people. My father, who has helped me learn programming in Python and has taught me the basics of machine learning. Thank you for believing in me. Likewise, I would like to thank my supervisor mr. Kampwart for being enthusiastic and keeping me motivated. Lastly, I wanted to give thanks to DataCamp for providing me with courses on programming in Python and to Kaggle.com for the dataset of the Titanic.

Contents

2 Introduction

In the year 1912 on the 15th of April one of the most infamous ships in history would crash into an iceberg and sink in the North Atlantic Ocean. During its maiden voyage from Southhampton to New York City on the 14th of April at 11:40 p.m. ship's time, the lookout sounded the alarm when a massive clump of solid ice caught his attention. The first mate had seen the iceberg before the lookout did and tried to turn the ship around. Unfortunately, he was too late. Forty seconds later at a high speed the Titanic collided with a huge rock made of ice with a weight of 30 million kilograms. The collision caused a series of holes along the side of the hull.[1] Six of the watertight compartments were filled with water, whereas the ship could only sail on with a maximum of four compartments flooded. Consequently, the Titanic was doomed to sink. The crew understood they needed to act fast. They deployed the evacuation program. The ship carried twenty lifeboats. In principle the protocol "women and children first" was followed. However, this was not true for everyone on board. The chance of being saved was dependent on the class in which one travelled and the place where one found itself during the evacuation. Around 2:20 a.m. parts of the Titanic broke off and sunk with one thousand people still on board. On deck were some of the richest people in the world, including millionaires, movie stars, school teachers and immigrants, who were hoping to find a new life in New York City. A life that they would, therefore, never find. Two hours after the ship sank, the liner RMS Carpathia arrived and saved an estimated 705 people.[2] The sinking of the RMS Titanic killed 1502 out of the 2224 people on board, crew members as well as passengers.[3]

The RMS Titanic was the largest ship on water during that time and it was the second of three ocean liners operated by the White Star Line .[4] The ship consisted of nine decks, the boat deck, seven decks labelled from A to G which carried the passengers and the Orlop Deck which was below the waterline. The liner had a height of 175 feet and a breadth of 92 feet.[5]

The Titanic may be one of the most iconic ships in history, its story known the world over.[6] The tragedy has led to better safety regulations for ships and inspired numerous expeditions, movies, books, plays and characters.

So many passengers have lost their lives due to the fact that there were not enough lifeboats. Luck played a part in surviving this disaster. Moreover, some groups had an advantage compared to other groups. For instance, the "women and children first" policy left a relatively larger number of men aboard. In the same way as children and teenagers

[1] http://www.bbc.co.uk/history/titanic (consulted on the 5th of August, 2018).

[2] https://en.wikipedia.org/wiki/RMS_Titanic#Maiden_voyage (consulted on the 5th of August, 2018).

[3] https://www.kaggle.com/c/titanic (consulted on the 5th of August, 2018).

[4] https://en.wikipedia.org/wiki/RMS_Titanic#Maiden_voyage (consulted on the 5th of August, 2018).

[5] https://www.encyclopedia-titanica.org/titanic/ (consulted on the 5th of August, 2018).

[6] http://www.bbc.co.uk/history/titanic (consulted on the 5th of August, 2018).

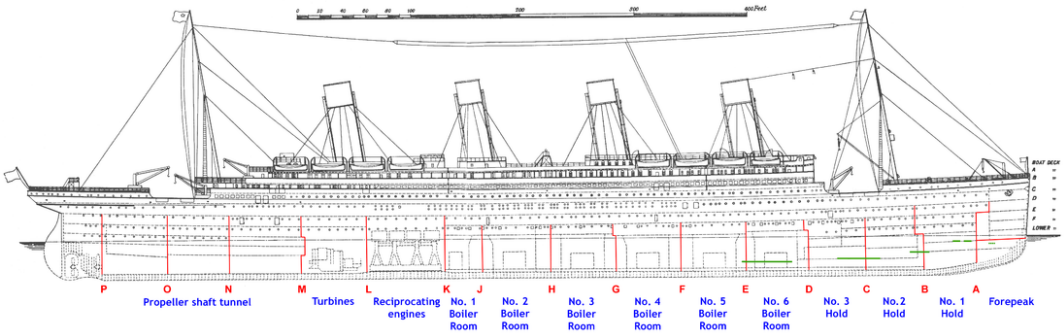


Figure 1: Profile of RMS Titanic with the decks indicated

had an advantage because of this principle. Similarly, speculations can be made regarding the advantage of the elderly aboard the Titanic. On the one hand it seems logical that the seniors were helped to the lifeboats because of a policy similar to the one about women and children. Older people are not as physically fit as the rest of the passengers, therefore they need to be assisted. On the other hand however, were the elderly the ones left behind as a result of their physical condition. They would have had more trouble climbing from the lowest deck to the boat deck. Finally, some people travelling first class might have had a better chance at surviving as well. The passengers were able to choose between three classes, varying in price and comfort. There was also a correlation between these three classes and wealth and social class. Most of the people travelling first class were, for example, businessmen, politicians and bankers. Second class travellers included professors, authors and tourists, members of the middle class. Emigrant workers moving to the United States and Canada travelled third class. In general, people travelling first class were closer to the boat deck and had, therefore, more chance to escape the flooding of the cabins (see Figure 2 and Figure 3). They could get to the life boats faster than people whose cabins were on one of the lower decks. The price paid for a ticket is correlated with class. Tickets for travelling first class were in general more expensive than tickets for travelling second or third class.

In this paper we will take a look at what people were more likely to survive the demise of the Titanic with the help of machine learning. We will predict the chances of survival of certain groups of passengers. In addition, we will see if the expectations that children, women and rich people were indeed benefited are correct.

2.1 Machine Learning

For the past 15 years, scientists have tried to make computers learn new things from given data with the help of machine learning. The definition of machine learning given by an professor at Stanford University is as follows: "Machine learning is the science of getting

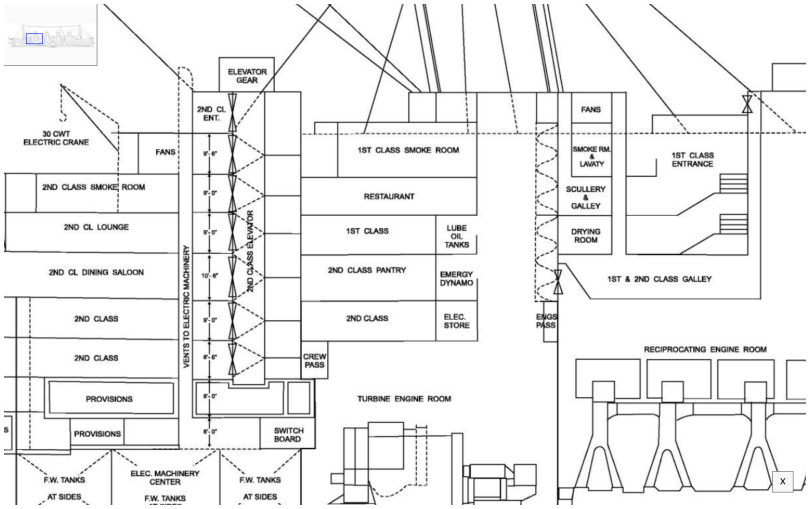


Figure 2: Deckplan of the Titanic

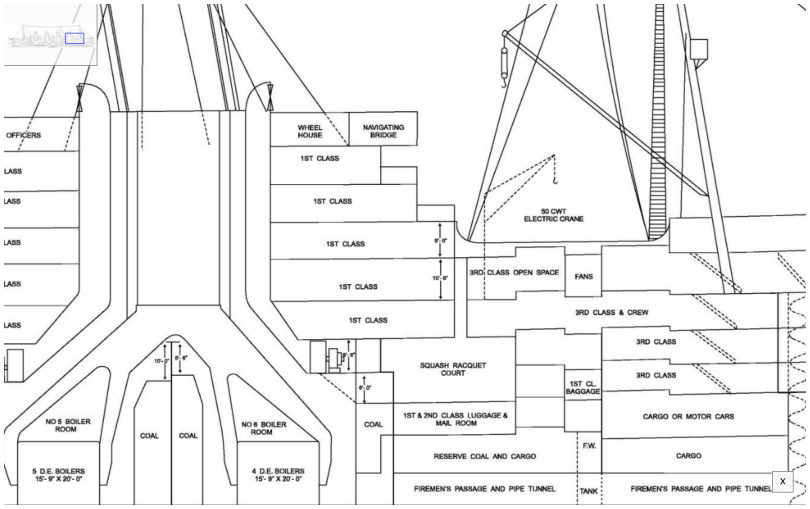


Figure 3: Deckplan of the Titanic

computers to act without being explicitly programmed."[7] It consists of giving computers the ability to learn and make decisions from data. These machine learning techniques are used to build predictive models. To illustrate, we will discuss some examples.

Spam emails are sent to everyone who has an emailaccount. Whether the email is from a lottery telling you you have won a $1-million prize or from an unknown travel-agency offering you a trip to an exclusive resort for very little money. It does not matter what the email looks like, your computer is able to distinguish the spam from the usual emails and places the spam in the spam folder of your account. The computer can detect the elements of spam, find patterns and compares the found patterns to new mail. Spam tends to have characteristic elements such as spelling mistakes, an originating address in Nigeria or claims that it needs your bank information. Furthermore, huge tech giants such as Google, Netflix and Spotify use machine learning. The algorithms of these firms offer recommendations and suggestions based on previous user searches, exactly because they can recognise a pattern in these searches.[8] Maybe one of the best known examples is AlphaGo. The computer programm developed by Google DeepMind in London to play the the boardgame Go.[9] In October 2015, AlphaGo became the first computer Go program to beat a human professional Go player. It was trained on moves of expert players from recorded historical games, a database of around 30 million moves. The algorithm used these moves to mimic human play by attempting to match these moves. Moreover, machine learning is making a breakthrough in the medical field as well. AI pioneer Regina Barzilay carried out research and is now teaching machines to hunt down cancer. Experienced doctors have only a limited amount of patients' experience. Curing cancer is now more a trial-and-error process. With the help of machine learning people can be diagnosed faster and can be cured with the appropriate treatment.[10]

A lot of different machine learning techniques exist. In this paper we will discuss two examples.

2.2 Different types of Machine Learning

Machine learning can be divided in roughly three categories: reinforcement, unsupervised and supervised learning. The latter two will be discussed and these can also be divided in subgroups. We have to ask ourselves the questions how does the computer know it is getting better or not, and how does it know how to improve? The different answers to these questions have made these different types of machine learning techniques exist, see

[7] Quote created by Stanford University on the course of Machine Learning, taught by: Andrew Ng, Co-founder, Coursera; Adjunct Professor, Stanford University; formerly head of Baidu AI Group/Google Brain. https://www.coursera.org/learn/machine-learning (consulted on the 6th of August, 2018).

[8] https://www.redpixie.com/blog/examples-of-machine-learning(consulted on the 6th of August, 2018).

[9] https://deepmind.com/blog/alphago-zero-learning-scratch/(consulted on the 6th of August, 2018).

[10] New Scientist Weekly, 21 July 2018, I teach machines to hunt down cancer, Interview by Chelsea Whyte

Figure 4.

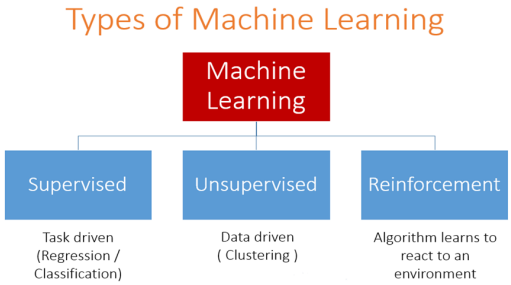


Figure 4: An illustration of the different types of machine learning

Unsupervised learning This is a version of machine learning where the computer has to uncover hidden patterns from unlabeled data. Correct responses are not provided. The algorithm has to identify similarities between the inputs. This way the inputs that have something in common are categorised together.[11]

For instance, grouping customers in categories based on buying behaviour without knowing in advance what these categories might be.

Supervised learning The majority of machine learning uses supervised learning. Whereas unsupervised learning has to make decisions from data that is not labeled (the correct responses are not provided), supervised machine learning deals with labeled data. The correct answers are already provided in a training set of examples. The algorithm generalises to respond correctly to all possible inputs, based on this training. The computer is provided with a specific input combined with the correct output or prediction. This way, the machine is trained to see the connections between the input and the right output. When a computer has had enough training or has been provided with enough data points, it will make less mistakes with every try. Eventually the computer is able to produce the right output based on a given input. [12]

The Titanic task is a perfect example of supervised learning. We already know who has survived the disaster and who has not. This way we can train our computer on the complete dataset. Consequently, the computer learns to connect particular variables to the fact if someone has survived or not. Given a new person, of whom we don't know if he or she has survived it, the computer can make a prediction. We can produce the chances of survival for particular variables, e.g. gender, class etc. Picking the right variables is crucial for producing a model. Moreover, choosing how to process your data is important. We will put a lot of effort in choosing the right variables and how to process the data. This

[11] Machine Learning, An Algorithmic Perspective second edition by Stephen Marsland, 2015 by Taylor & Francis Group.

[12] https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/(consulted on the 26th of August, 2018).

will take up a lot of time and is part of the trial-and-error procedure. A dataset consists of datapoints. These are samples described using predictor variables and a target variable. Organised in a table with rows and columns. The goal is to predict the target variable, in this case 1 or 0 representing survived or not survived respectively in our Titanic dataset, given the predictor variables, such as class, gender, age, siblings etc.

We can specify two different types of supervised learning:

- **Classification**: the target variable consists of categories. Predicting survival on the Titanic is a classification problem. We have to classify, based on our predictor variables, if a person belongs to the class of survived (1) or not survived (0). This is a special case of a classification problem called binary classification. For the Titanic problem we use labelled data. Consequently, we use supervised machine learning.
- **Regression**: the target variable is continuous. For instance, a dataset containing housing price data like the year the house was built, number of bedrooms, acreage. There is a price associated with each house. The goal is to predict the price of a house, given these variables. For the reason that a price is a continuous variable, this problem is an example of regression.

2.3 Algorithms

To train our computer on the dataset we use two different algorithms. Because we approach our problem in two different ways, the results will be more trustworthy. Training our model on the data using an algorithm is called 'fitting' a model to the data. Fitting means minimizing the classification mistakes that we make. We split our data into a training and test set. We fit our model to the training data and predict on the test set.

2.3.1 KNearestNeighbors

To begin with, we will use the so-called KNearestNeighbors algorithm. It predicts a label of a datapoint by looking at the 'k' closest labelled data points. KNN takes a majority vote on what label an undecided point has to have. For instance, when we want to decide if a dot on this map is a blue square or a red triangle, we can choose our 'k' as 3 (see Figure 5). With choosing our 'k', we create a set of decision boundaries. Our computer will look at the three closest datapoints to classify our undecided point. If two of those three are blue squares, it classifies our undecided point as a blue square. If two of those three points are red triangles, it classifies our undecided point as a red triangle. The trick is to choose the right value for 'k'. Choosing a too large value for 'k', will lead to underfitting therefore creating a smoother decision boundary. This way we will have a less complex model, because our algorithm generalizes too much and uses too little information. On the other side, choosing a too small value for 'k' will lead to overfitting. Consequently, our model will be more complex and will have a more erratic pattern. We use 'too much' information

and our model becomes less reliable. These problems of overfitting and underfitting are very common in the world of machine learning. They also occur using other algorithms. Finding the right 'k' is a combination of using other algorithms to find it and a trial-and-error procedure.[13]

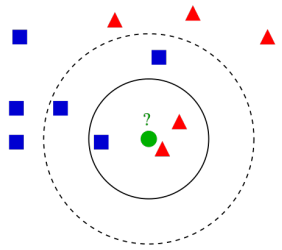


Figure 5: Illustration of the algorithm called KNearestNeighbors

2.3.2 Logistic regression

Second, we use an algorithm called logistic regression (logreg). The name may be misleading because logreg is commonly used for classification problems. It outputs probabilities. For example, if the dataset consists of n different classes, the algorithm calculates the chance that one specific case is classified as belonging to one of these n classes. In our case, we see $n = 2$. Therefore, we are dealing with a binary classification problem.[14] This implies the following: if we find $p > 0.5$, the variable is classified as 1, the passenger has survived the disaster; when we see $p < 0.5$, it is classified as 0, the passenger has not survived.

To explain the principle of logistic regression, we will have a look at a linear function first:

$$y = ax + b \tag{1}$$

In this case there is only one predictor variable. But we have more than one predictor variable in our dataset of the Titanic. a and b are the parameters of our model. We want to fit a line to the data. Fitting, in this case, consists of choosing a slope a and an intercept b. Our Titanic dataset has more than one feature, because we have more than one predictor variable. Using linear regression, our line will look something like this, where each x represents a different predictor variable.

$$y = a_1x_1 + a_2x_2 + \cdots + a_nx_n + b + \varepsilon_i \tag{2}$$

[13] DataCamp courses on Supervised Learning with scikitlearn: https://www.datacamp.com/courses/q:supervised (consulted on the 13th of February, 2018).

[14] https://www.statisticssolutions.com/what-is-logistic-regression/(consulted on the 5th of September, 2018).

By calculating the vertical distance between each data point and the line, we can get an impression of how accurate our model is. This distance is called the residual (ε). One option is to minimze the sum of the residuals. However, this will not work because large positive values will cancel out large negative values. Consequently, shifting the line upwards will always reduce the sum of the residuals. This is because the positive values will be ∞ and the negative values will be $-\infty$. As a result of this, the sum of the residuals will be zero. So, to make sure that our line is as close to the actual data as possible, we calculate the sum of squared residuals (see Figure 6 and see Equation 3). This is called OLS, which stands for Ordinary Least Squares. When we call fit on our logistic regression model in scitkitlearn, it performs this OLS under the hood. Scikitlearn is a popular machine learning library for Python, which we will use to train our computer (see Footnote 13).

$$\sum_{i=1}^{N} \varepsilon_i^2 \tag{3}$$

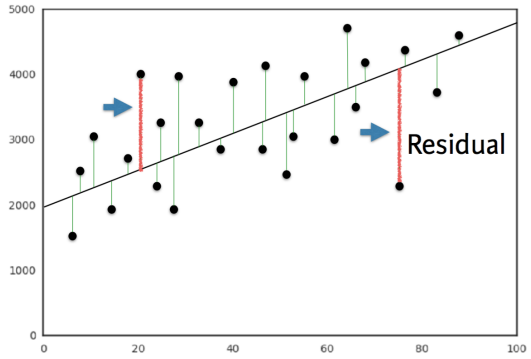


Figure 6: Ordinary Least Squares: Minimize sum of squares of residuals

The red lines in the illustration (see Figure 6) represent ε_i^2. The equations mentioned earlier are used most commonly for linear regression. We will use logistic regression, because our target variable is not continuous: our variable is either 0 or 1. The logistic function $\varsigma(t)$ is defined as follows:

$$\sigma_t = \frac{e^t}{1 + e^t} \tag{4}$$

Because we have three variables(i.e. age, gender and class), t in this case is of the form:

$$y = a_1 x_1 + a_2 x_2 + a_3 x_3 + b + \varepsilon_i \tag{5}$$

As the name already tells us suggests, logistic regression is based on the logistic function. This is a sigmoid function (see Figure 7), which takes any real input t ($t \in \mathbb{R}$), and outputs a value between zero and one, a probability.

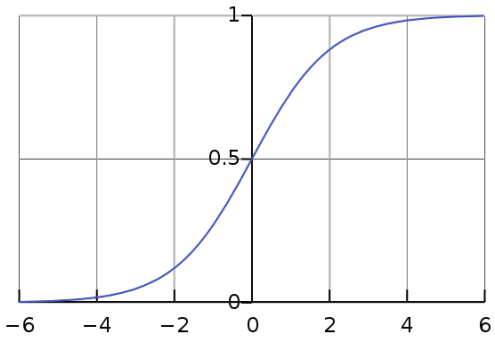


Figure 7: The logistic function

The same principle applies to logistic regression regarding the underfitting and overfitting problem. Adding more independent variables to our model will increase the amount of explained variance. Our model will be more complex and will have a more erratic pattern, as mentioned earlier. Using too little independent variables will result in underfitting, where our model is too 'simple'.

After using these two algorithms, we can measure model performance. To do this, we can use metrics such as accuracy. Accuracy is the fraction of correct predictions, think of the fraction of cases where the model correctly predicts that someone survived. How these metrics work, will be explained later on.

To sum up, we follow this procedure: We split our dataset into a training set and test set. Then we fit or train the classifier to the training set. Subsequently, we predict on the test set and print the prediction. In the end, we compare our predictions to the known labels and compute the metric of accuracy.

2.4 Main questions and sub-questions

This research and information leads us to the following main question and sub-questions:

Main question

Is it possible to make an accurate prediction whether the passengers on board of the Titanic survived the disaster or not using the information about gender, class, age and fare given in the dataset?

Sub-questions

- *What is the influence of gender on the chance of surviving after the Titanic had sunk?*
- *What is the influence of fare on the chance of surviving after the Titanic had sunk?*
- *What is the influence of class on the chance of surviving after the Titanic had sunk?*
- *What is the influence of age on the chance of surviving after the Titanic had sunk?*
- *Is there a monotonous relationship between age and survival rate?*

These questions lead to the following hypotheses:

- **Main question** : Yes this is possible, with the help of machine learning using the algorithms KNearestNeighbours and logistic regression.
- **Sub-questions** :
  - The survival rate of women is higher than the survival rate of men.
  - The survival rate of passengers who paid a higher fare is higher than those who paid less.
  - The survival rate of passengers who were travelling in a higher class is higher than those travelling in a lower class.
  - The survival rate of children and elderly is higher than the survival rate of the adults.
  - The relationship between age and survival rate is not monotonous.

3 Preparation

3.1 A first look at the dataset

The adventure begins with importing a couple of packages. We will use other packages as well. These will be imported along the way.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

The dataset is downloaded from Kaggle[15] as `csv_file`. Thereafter, the data is read into a dataframe by using pandas `pd.read_csv`.

```
data = pd.read_csv('titanic.csv')
```

Before we get started with our algorithms, we will have a look at our dataset. First we perform some numerical EDA. EDA stands for Exploratory Data Analysis. This will help us analyse our dataset and get a first impression of the information. It is not necessary to build a dataframe, because all the information is already organised in a table.

Using the `.head()` method, we can see the first five rows of our dataset in Table 1. A couple of questions come to mind. Which variables play a role by determining the probability of surviving the Titanic? Moreover, `Sex` for example is not a numeric value. How do we convert this in a way that our computer can deal with this variable?

```
data.head()
```

We see a lot of columns. `Pass` gives us the PassengerId. `Surv` shows us a 0 or 1, which stands for not survived and survived respectively. `SibSp` represents the number of siblings and `Parch` represents the number of parents of the passenger on board. `Emb` tells us the port of embarkation: `C` stands for Cherbourg, `Q` for Queenstown and `S` for Southampton. With the `.describe()` method we can see the static data. The mean, standarddeviation etcetera are given in Table 2.

```
data.describe()
```

It is also possible to search for particular passengers in the dataset. Such as passengers with a particular name or with a particular age of, for example, eighty years old.

[15] https://www.kaggle.com/c/titanic (consulted on the 18th of January 2018)

Table 1: Head of the dataframe

	Pass	Surv	Class	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emb
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/ O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Table 2: Description of the dataframe

	Pass	Surv	Class	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	8.910000e+02	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	-1.832252e+18	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	3.682066e+18	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	-9.223372e+18	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	6.000000e+00	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	2.400000e+01	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	3.500000e+01	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	8.000000e+01	8.000000	6.000000	512.329200

```
data[data.Name == 'Braund, Mr. Owen Harris']
```

```
   PassengerId  Survived  Pclass                      Name   Sex   Age  SibSp  \
0            1         0       3  Braund, Mr. Owen Harris  male  22.0      1

   Parch     Ticket  Fare Cabin Embarked
0      0  A/5 21171  7.25   NaN        S
```

```
data[data.Age == 80]
```

```
     PassengerId  Survived  Pclass                                  Name  \
630          631         1       1  Barkworth, Mr. Algernon Henry Wilson

      Sex   Age  SibSp  Parch Ticket  Fare Cabin Embarked
630  male  80.0      0      0  27042  30.0   A23        S
```

It is also possible to see who has paid more than 400 dollars for his or her ticket. We see that it is easy to make a selection in our dataset using the > sign.

```
data[data.Fare > 400]
```

```
     PassengerId  Survived  Pclass                                Name  \
258          259         1       1                   Ward, Miss. Anna
679          680         1       1  Cardeza, Mr. Thomas Drake Martinez
737          738         1       1             Lesurer, Mr. Gustave J

        Sex  Age  SibSp  Parch    Ticket      Fare        Cabin Embarked
258  female   35      0      0  PC 17755  512.3292          NaN        C
679    male   36      0      1  PC 17755  512.3292  B51 B53 B55        C
737    male   35      0      0  PC 17755  512.3292         B101        C
```

Next we perform some visual EDA. We do this in order to have a look at possible correlation between variables and at how our data is distributed. We can make a couple of plots, such as the Seaborn's binary countplot or a scatter matrix. We do this using the `matplotlib.pyplot` and `seaborn` packages. Scatterplots are not the best choice to illustrate some of our variables. We have plotted these figures just to take a look at possible correlation, not at causality.

Let's start with plotting `Age` against `Survived`. The result is Figure 8. `Survived` is not a continuous variable, so we see two strokes of dots. Looking at the plot, we can conclude that there was someone of eighty who has survived. However, it is not possible to draw more conclusions from this plot because the distribution for instance is not visible.

```
    plt.scatter(data.Age,data.Survived)
    plt.xlabel('Age')
    plt.ylabel('Survived')
    plt.savefig ('AgeSurvived.png')
```

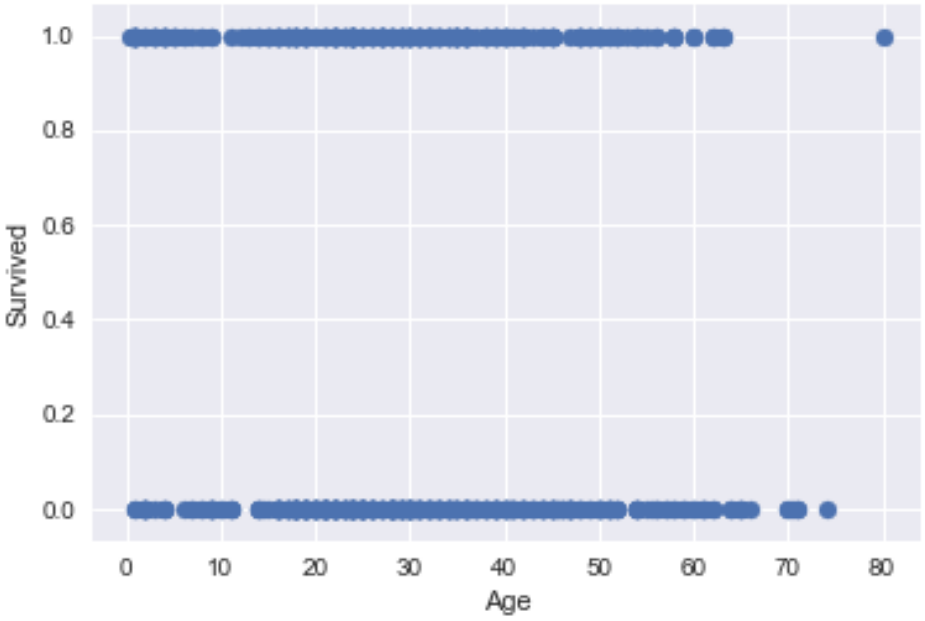


Figure 8: Plot of Age against Survived

Now we have a look at the relationship between class and price paid for a ticket in Figure 9. It is likely that we will see some correlation. The line relating to first class has higher values than the ones relating to second and third class. Below we write the same code for plotting the scatter plots. We will not, however, show the code everytime because this would make it less readable.

A couple of values stand out. We see that a passenger or more passengers travelling first class have paid more than 500 pounds for their ticketprice.

After we have plotted `Fare` against `Survived`, we take a look at Figure 10.

Between `Fare` and `Age` we can conclude that passengers younger than ten years have not paid a lot for their ticket as opposed to other passengers (see Figure 11). People who paid more for their tickets were older. But not everyone who was older, has paid more for their tickets.

If we plot a scatter matrix, we get Figure 12.

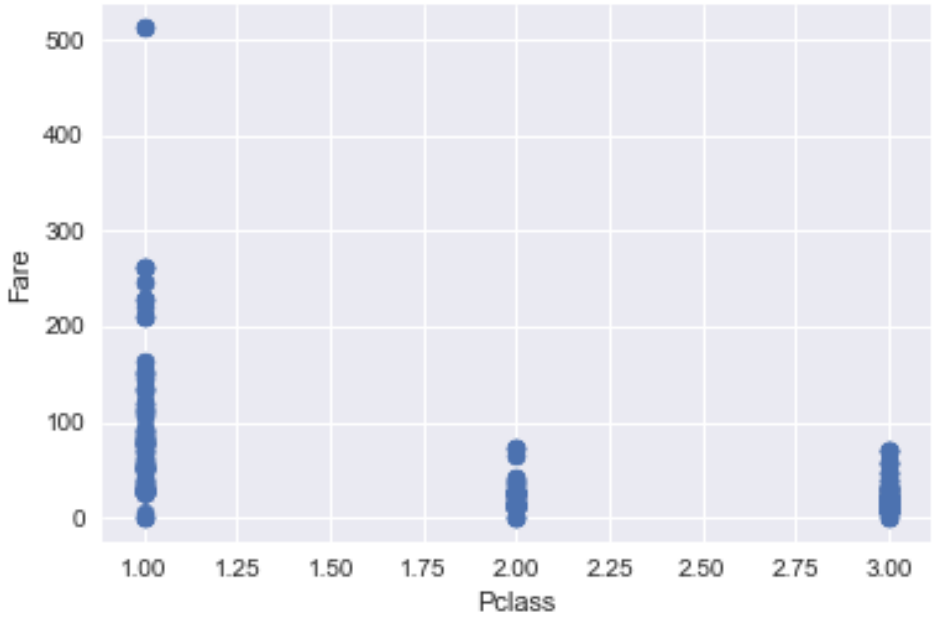


Figure 9: Plot of Class against Fare

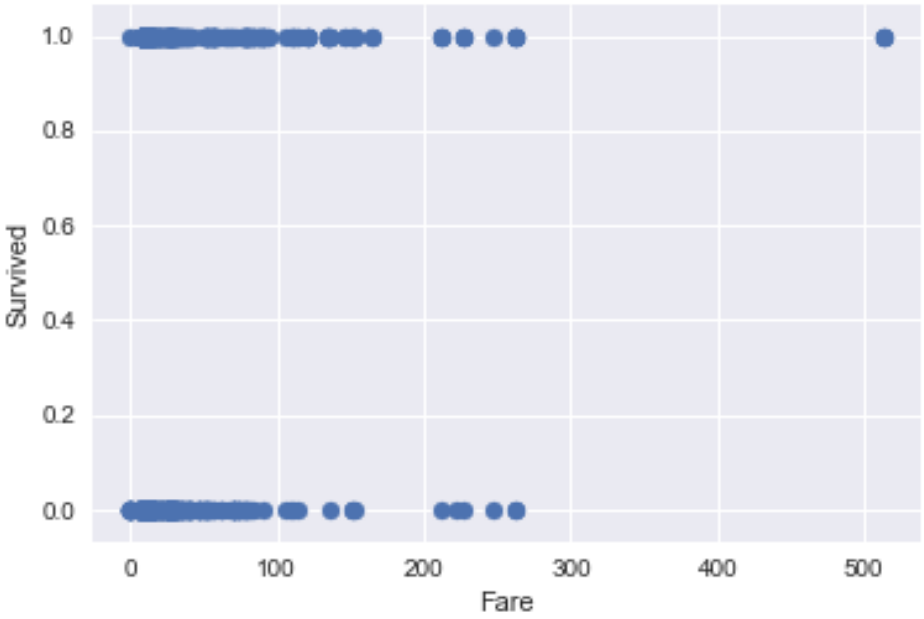


Figure 10: Plot of Fare against Survived

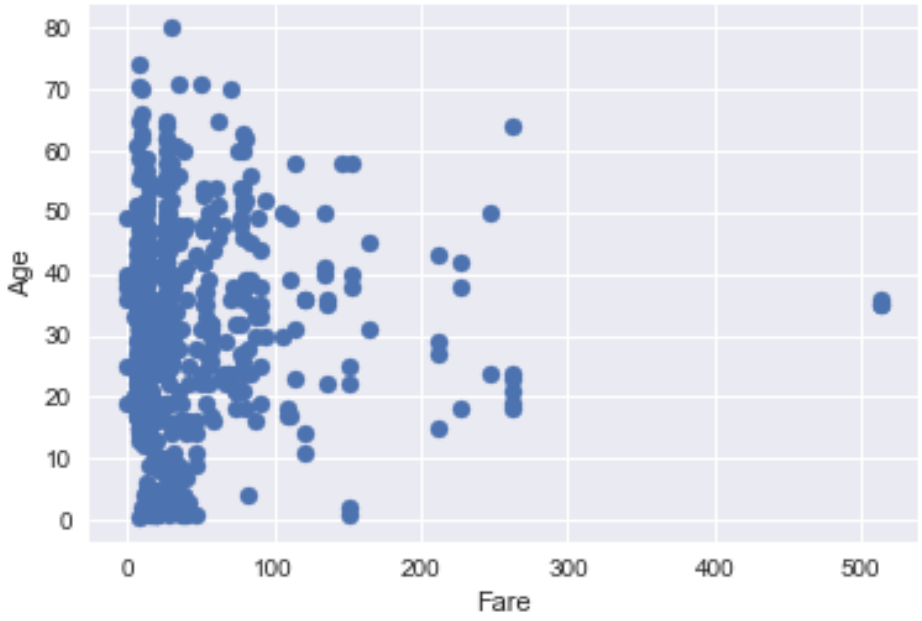


Figure 11: Plot of Fare against Age

```
from pandas.plotting import scatter_matrix

axs = scatter_matrix(P_titanic[['Pclass','Fare','Age']], alpha=0.2, figsize=(10, 10), diagonal='hist')
plt.savefig('scatter.png')
```

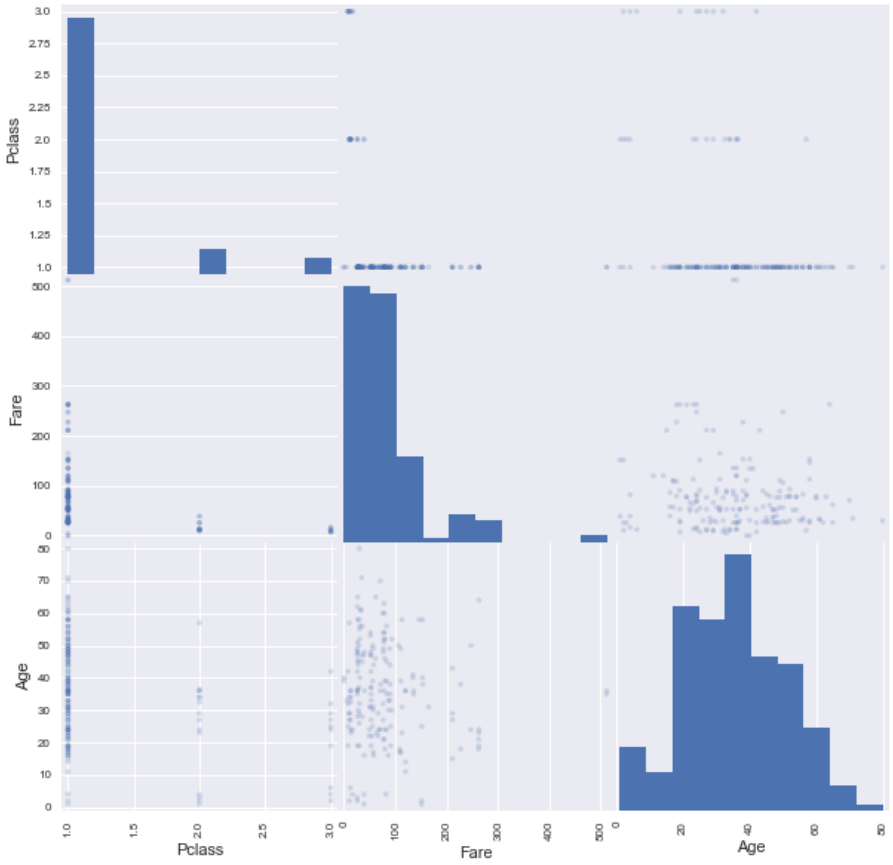


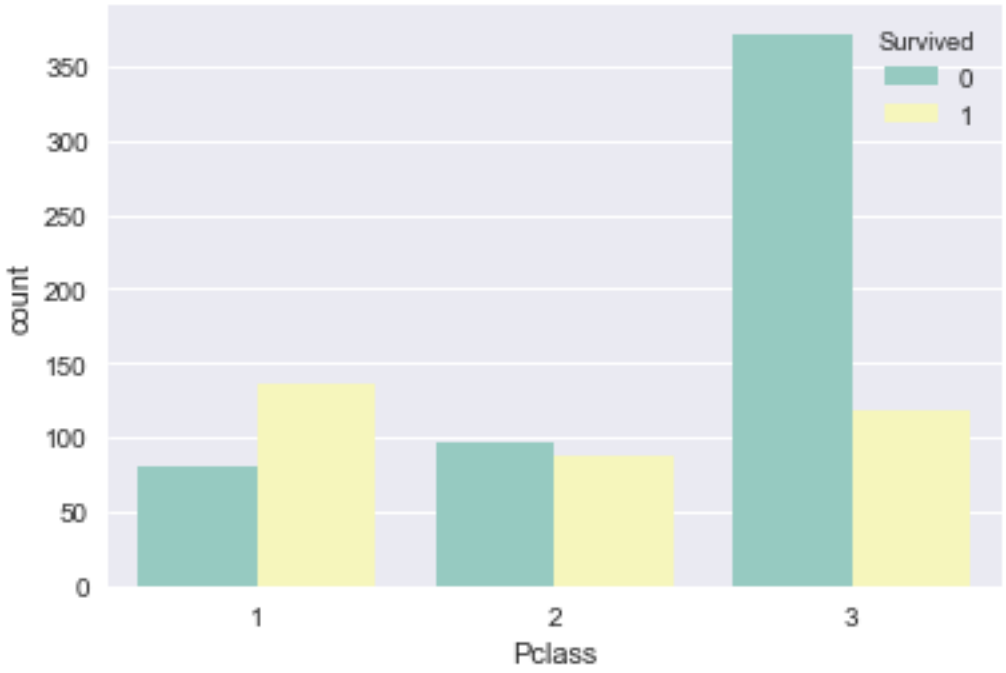
Figure 12: Scatter matrix with histograms on the diagonal

The scatter matrix plots all the combinations of our variables in the scatter plots. This gives us a nice overview. On the diagonal we see a histogram that represents the relative distribution of the variables. Looking at the histogram for `Age` for example, it shows how

many people of each particular age group were on board of the Titanic.

We will now plot a `binary Seaborn Counplot`. Plotting `Class` against `Survived`, we can see that there were more people in the third than in the first class. This makes it difficult to compare them to eachother and to draw a conclusion. One option is to calculate a percentage. In general, we cannot draw a conclusion regarding survival probabilities. In the third class, more passengers died than survived. In the first class, more people survived than perished. The plot only shows us one variable. This is another reason why we cannot be sure about the influence of class on the chance of survival. The effect of first class on the chance of survival can be different for a woman than for a man for example. This is because the variables have an influence on eachother as well. We will have a further look at this problem in the

```
    sns.set(style="darkgrid")
    ax = sns.countplot(x="Pclass",hue="Survived", data=data, palette="Set3")
```

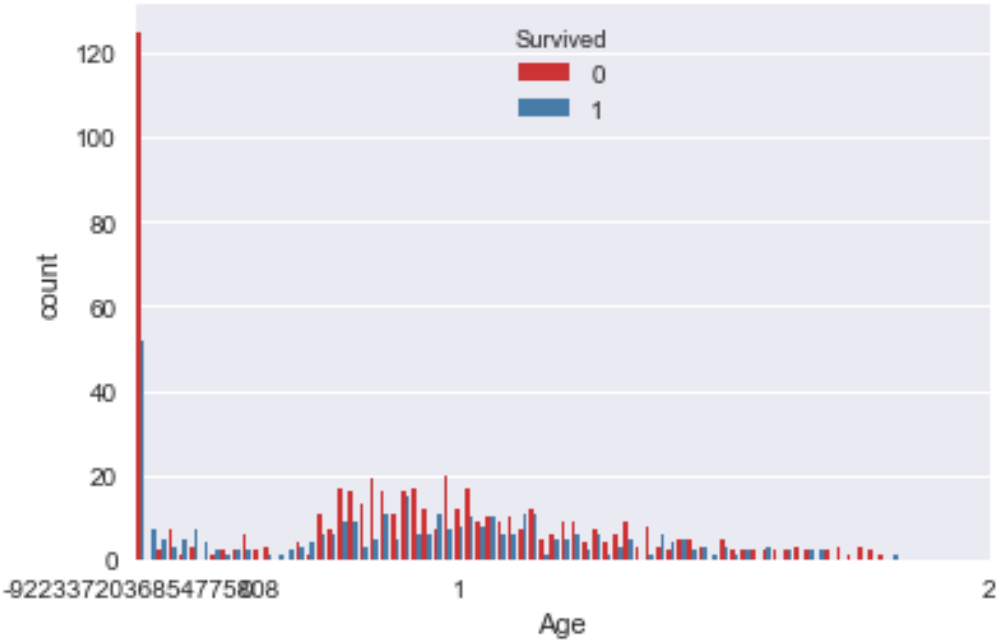


Here we see a plot with `Age` against `Survived`. We can see some blue points for the passengers of a younger age. Furthermore, a lot of people of middle age have not survived. This is caused to a great extent by the fact that there were more passengers of middle age on board.

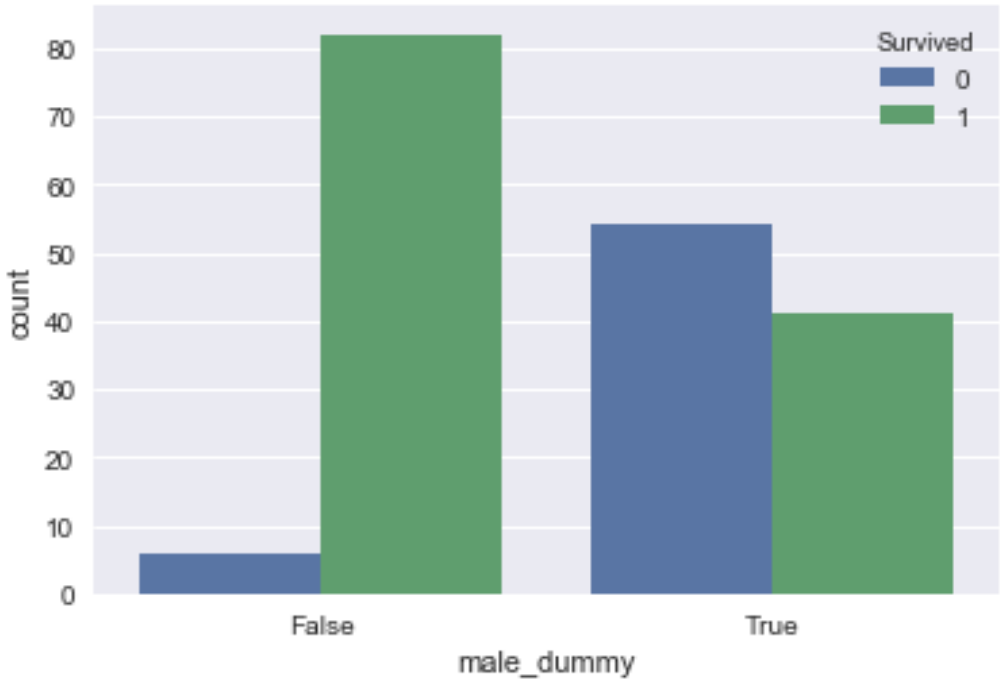
```
data.Age = data.Age.astype('int')
```

```
1 type(data.Age[0])
```

```
numpy.int64
```



Here we see a plot of our `male_dummy`. `False` represents in this case the women on board of the Titanic. We see that there were more women who have survived than women who did not. `True` stands in this case for the men on board. We see that more men have perished than survived. Could this mean that the "women and children first" policy was helpful?



For more plots, take a look at the

3.2 Preprocessing techniques

Now we have explored our dataset and have seen what it looks like, we will adjust a couple of things. This adjusting will be done using the so-called "preprocessing techniques". As mentioned, the package `scikitlearn` cannot work with non-numerical values like the values of `Sex`. We have to come up with a solution. In addition to this, our dataset is not complete. We still miss values of particular passengers. We write the following code:

```
1    df_cleaned = data.dropna()
2    df_cleaned['male_dummy'] = (df_cleaned.Sex == 'male')
3    X = df_cleaned[['Age','male_dummy', 'Pclass', 'SibSp', 'Fare']]
4    y = df_cleaned[['Survived']]
```

We "clean" our dataset for the first time to make it more suitable for the packages we will be using. All rows with missing values, these are called NaNs (this is short for Not a Number), are deleted. We delete these by using `.dropna()`. There are other ways than deleting rows to handle this problem. Such as, replacing the NaNs with the mean or interpolating. However, the choice was made this time to delete these rows. Furthermore, we see that the problem of the `Sex` column not being a numeric value is handled. The

Table 3: Head of the cleaned dataframe

	Pass	Surv	Class	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emb	male_-dummy
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	False
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	False
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S	True
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	G6	S	False
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S	False

Table 4: Head of P_titanic

	Pclass	Fare	Age	male_dummy
1	1	71.2833	38.0	False
3	1	53.1000	35.0	False
6	1	51.8625	54.0	True
10	3	16.7000	4.0	False
11	1	26.5500	58.0	False

values in the `Sex` column are changed into a boolean. A boolean is a datatype with only two possible values, i.e. `True or False`. Males are given a `True` (1) and the females are given a `False` (0). Next we have added a couple of variables to `X`: `Age`, `male_dummy`, `Pclass`, `SibSp` and `Fare`. These are all numeric values and therefore easy to use. Here we see the cleaned dataframe in Table 3 with the new added column `male_dummy`.

In this paper we will only have a look at the variables `Age`, `Sex`, `Class` and `Fare`. For a more accessible dataset, we will delete the columns with data we will not use when making a prediction. This new dataset is called `P_titanic`. See Table 4.

```
P_titanic = df_cleaned[['Pclass', 'Fare', 'Age', 'male_dummy']]
```

```
P_titanic.head()
```

The corresponding column with the information about who has survived and who has not survived is called `q_titanic` and is given in Table 5.

Table 5: Head of q_titanic

	Survived
1	1
3	1
6	0
10	1
11	1

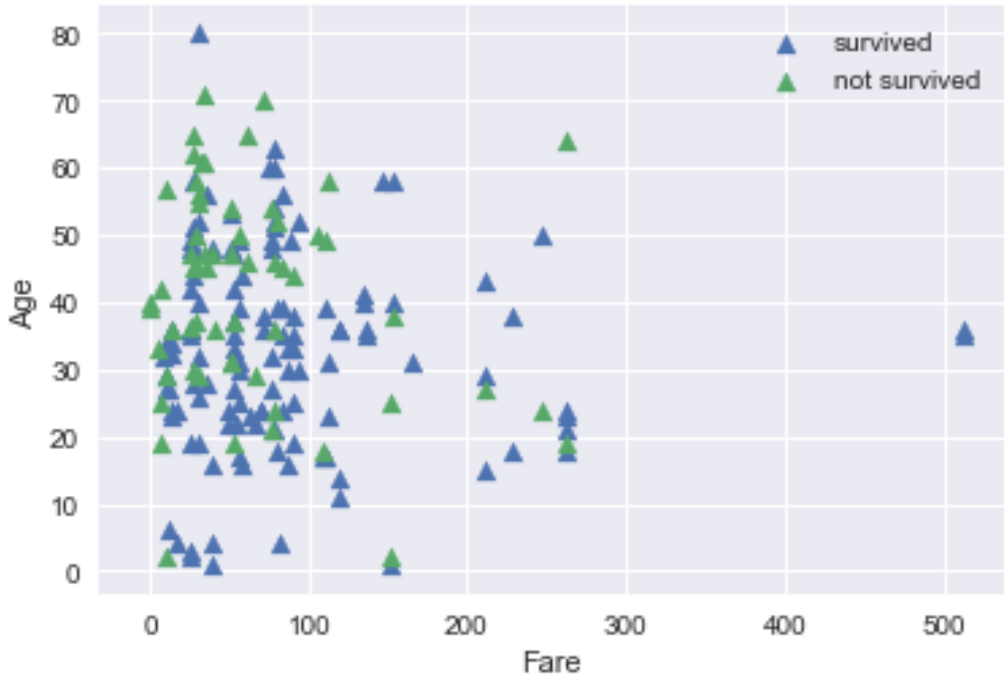
```
    q_titanic = df_cleaned.Survived
```

```
q_titanic.head()
```

We see that numbers 2, 4, 5 etcetera are missing. This makes sense because we have deleted these rows with missing values earlier.

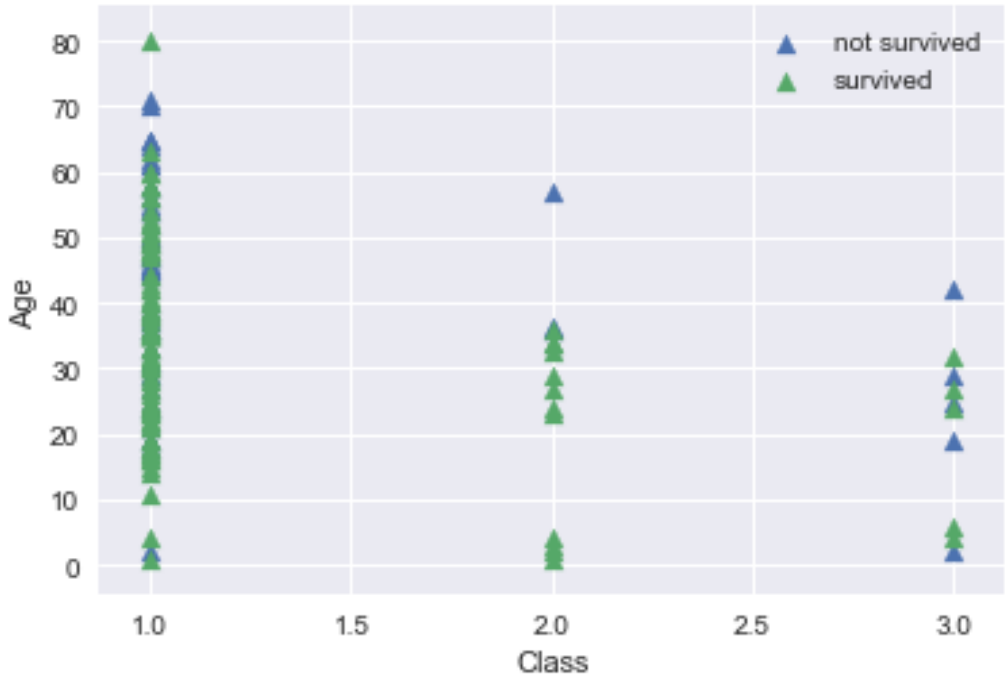
Using this data it is possible to make a graphic illustration of a prediction. We select the data concerning three of our variables, which includes `Survived` in any case.

```
    survived = df_cleaned[df_cleaned.Survived == 1]
    not_survived = df_cleaned[df_cleaned.Survived == 0]

    plt.scatter(survived.Fare, survived.Age, marker='^', label = 'survived')
    plt.scatter(not_survived.Fare, not_survived.Age, marker='^', label = 'not survived')
    plt.xlabel('Fare')
    plt.ylabel('Age')
    plt.legend()
```



Here we see one of the first graphic illustrations of the relation between `Fare`, `Age` and `Survived`. The relation is not very clear but we see that the higher the fare the more people survived and the higher the age the less people survived. However, this figure is not very accurate, because of the fact that only three variables were used. It is not possible to draw a reliable conclusion from this plot.

```
    survived = df_cleaned[df_cleaned.Survived == 1]
    not_survived = df_cleaned[df_cleaned.Survived == 0]


    plt.scatter(not_survived.Pclass, not_survived.Age, marker='^', label = 'not survived')
    plt.scatter(survived.Pclass, survived.Age, marker='^', label = 'survived')
    plt.xlabel('Class')
    plt.ylabel('Age')
    plt.legend()
```



It is very inconvenient to plot discrete variables such as `Class` and `Age`. It is harder to distinguish how many blue and how many green triangles there are in the plot. In the we will plot other variables against eachother.

3.3 The first algorithm: KNearestNeighbors

One way to approach our problem is using the algorithm called KNearestNeighbors. We import the classifier from the library `sklearn.neighbours`.

```
    from sklearn.neighbors import KNeighborsClassifier
```

To start with we can choose 6 as our number of neighbors, just to explore how the algorithm works and to see how reliable the results are. In KNN finding the value of k is not easy. A small value of k means that noise will have a higher influence on the result and a large value makes it computationally expensive. We will not spend a lot of time on finding the right k for the reason that the emphasis of this paper is on getting a general idea of how the algorithms work.

```
    knn = KNeighborsClassifier(n_neighbors=6)
```

We split our data into a training set and a test set. The arguments give us information about how much of our data we use as a test set and how much of our data we use as a training set. This and the parameters will be varied to see which values gives the best prediction. We find that our model performance is dependent on the way our data is split. If we choose our test size to be 0.2 and we compute our accuracy score, we get the following:

```
from sklearn.model_selection import train_test_split
P_titanic_train, P_titanic_test, q_titanic_train, q_titanic_test = \
    train_test_split(P_titanic,q_titanic, test_size=0.2, random_state=42)
```

We fit our classifier on the training set and consequently predict on the test set.

```
 knn.fit(P_titanic_train, q_titanic_train)
 prediction= knn.predict(P_titanic_test)
```

If we compute our accuracy score, which is the fraction of correct predictions, we find the following value:

```
knn.score(P_titanic_test, q_titanic_test)
```

```
0.7027027027027027
```

If we `print` our prediction, this is what it looks like:

```
print('Prediction{}'.format(prediction))
```

Table 6: Head of the test set

	Pclass	Fare	Age	male_dummy
118	1	247.5208	24.0	True
251	3	10.4625	29.0	False
742	1	262.3750	21.0	False
544	1	106.4250	50.0	True
712	1	52.0000	48.0	True

```
Prediction[1 1 1 0 1 0 1 1 1 1 0 1 0 1 1 1 1 0 \
1 1 1 1 0 1 0 1 1 1 1 0 1 1 0 1 1 1 1]
```

This is a prediction for the first 38 passengers with his or her specific characteristics. If we take a look at the head of our `P_titanic_test` (Table 6), we can see for whom the algorithm has predicted that he or she has survived. The third '1' corresponds with the passenger number 742 on the list.

So number 742 has, according to our model, survived the disaster. The `PassengerID` of this passenger is 743, because the ID is one higher than the row number.

```
df_cleaned[df_cleaned.PassengerId == 743]
```

```
     PassengerId  Survived  Pclass                                      Name  \
742          743         1       1  Ryerson, Miss. Susan Parker "Suzette"

        Sex   Age  SibSp  Parch    Ticket     Fare            Cabin Embarked  \
742  female  21.0      2      2  PC 17608  262.375  B57 B59 B63 B66        C

     male_dummy
742       False
```

Miss Ryerson has survived! Congratulations Suzette!

Back to varying our test size. If we choose a value of 0.4 for our test size, we get a different outcome.

```
P_titanic_train, P_titanic_test, q_titanic_train, q_titanic_test = \
    train_test_split(P_titanic,q_titanic, test_size=0.4, random_state=42)
knn.fit(P_titanic_train, q_titanic_train)
prediction= knn.predict(P_titanic_test)
knn.score(P_titanic_test, q_titanic_test)
```

```
0.6756756756756757
```

```
print('Prediction{}'.format(prediction))
```

```
Prediction[1 1 1 0 0 0 1 1 1 0 0 1 1 1 1 1 1 1 1 1 /
1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1
1 1 1 1 0 1 0 0 1 0 1 1 1 1 1 0 1 0 0 1 1 0 1 0 1 /
0 1 1 0 1 1 1 1 1 1 0 0 1]
```

A larger test set gives us consequently a lower accuracy score, because we have a smaller training set. However, the accuracy score is not always reliable. See the for an explanation. It is not always obvious which size gives the best result. In the future we will use a test size of 0.2 for KNN and one of 0.25 for logistic regression.

Now we will use a couple of methods to make our model better and more reliable. To prevent that our results are influenced by one particular way of splitting our data, we perform a technique called *cross-validation*. We ask ourselves the questions: Do we see these results because we have accidentally chosen a very specific part of the data as our test set? Or is this a representative result of our entire dataset? This uncertainty can influence the reliability of our outcome. Using cross-validation we split our data into k folds and let our computer perform the algorithm k times on k different but equally large selections of our data of test and training sets. To illustrate, if we choose k is 5 we perform 5-fold cross-validation (see Figure 13) . Note well, we are not gaining more accuracy with this technique for we are not using more data. The dataset stays the same.

We use five different parts of our data as test set and the rest of the data as training set.

First we split our data into five groups. We hold out the first fold as a test set, fit our model on the remaining four groups and we then predict on the first fold. In the next fold we use the second block as test set and fit on the remaining data and so on. Working with more folds is more computationally expensive and thus taking the computer longer to perform the cross-validation.

To get an idea about how this cross-validation works, we will perform cv with 5 folds.

```
from sklearn.model_selection import cross_val_score
cv_scores = cross_val_score(knn, P_titanic, q_titanic, cv=5, scoring='roc_auc')
print(cv_scores)
```

```
[0.41666667 0.48833333 0.53833333 0.5    /
    0.50694444]
```

Here we see five values of R^2 from which we can compute statistics of interest such as the mean or median. R^2 is a statistical measure of how close the datapoints are to the

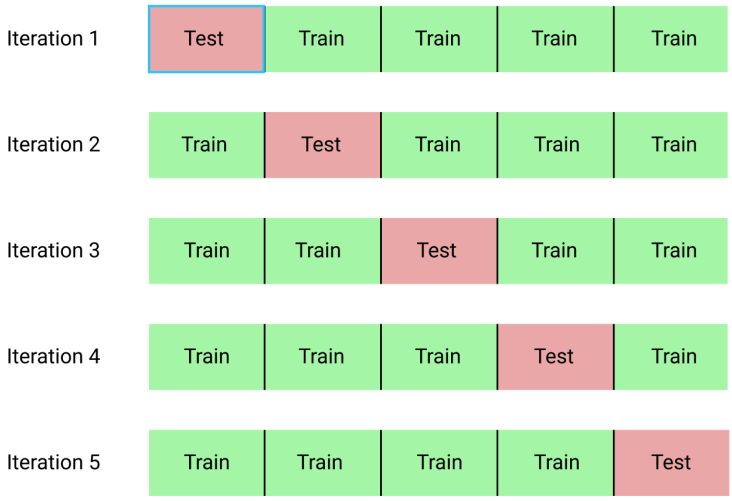


Figure 13: 5-fold cross-validation

fitted regression line. It is also known as the coefficient of determination or the coefficient of multiple determination for multiple regression.[16] It is the percentage of the response variable variation that is explained by a linear or *logistic?* model. 0% indicates that the model explains none of the variability of the response data around its mean, whereas 100% indicates that the model explains all the variability of the response data around its mean.

```
1 cv_scores.mean()
```

```
0.4900555555555556
```

To get an idea what the influence is of different sizes of cross-validation on our score, we will perform other cross-validations in the

Another way to find out how well our model performs is the so-called confusion matrix. The confusion matrix is a table with four different combinations of predicted and actual values. The name stems from the fact that it makes it easy to see if the system is confusing two classes.[17] These four different combinations are: true positive (TP), true negative (TN), false positive (FP) and false negative (FN). This table has two dimensions: "actual" and "predicted". TP indicates that the algorithm predicted positive and that it was right.

[16] https://www.datasciencecentral.com/profiles/blogs/regression-analysis-how-do-i-interpret-r-squared-and-ass (consulted on the 10th of December, 2018)

[17] https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62 (consulted on the 2nd of December, 2018)

So this is a correct prediction that the passenger has survived. TN says that the algorithm predicted negative (so the passenger did not survive) and that the prediction was true. FP: the computer predicted positive but it is false. FN means that the algorithm predicted negative but was not right. For an example for the prediction of spam emails in a confusion matrix, see Figure 14.

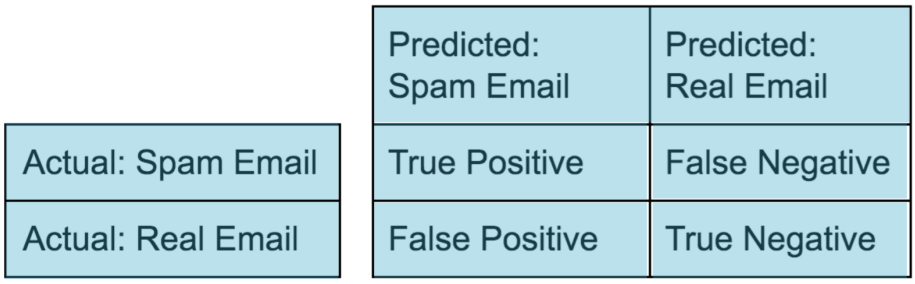


Figure 14: The confusion matrix

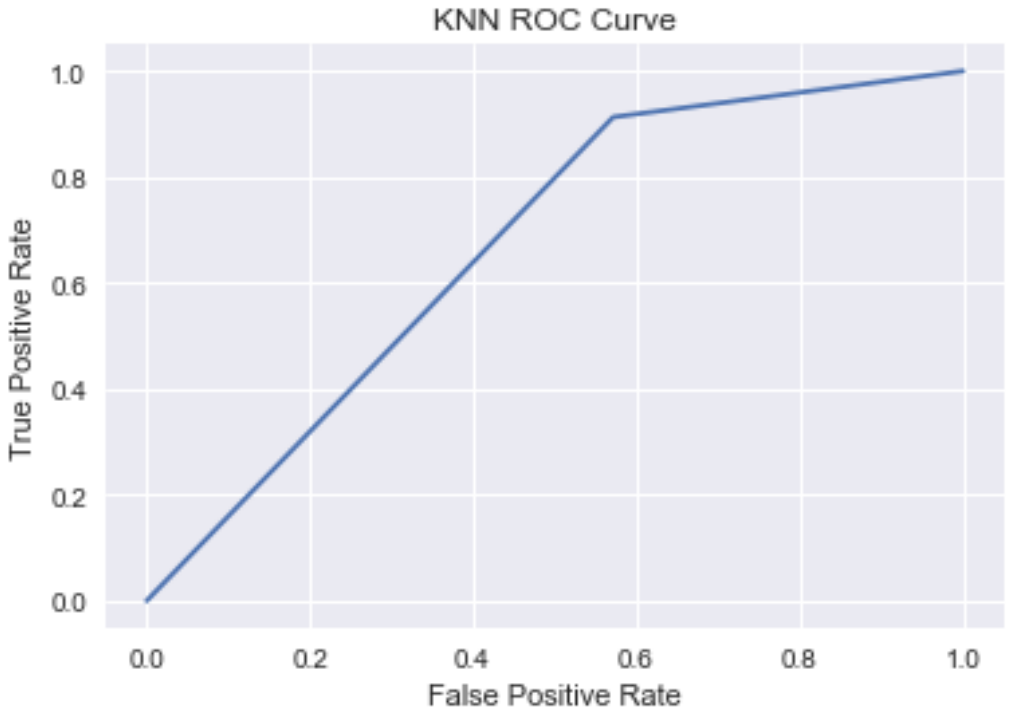
So accuracy can be described as follows:

$$accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{6}$$

We want our values on the diagonal to be as high as possible. A high number of values off the diagonal indicate problem areas. There are a lot of metrics that work with the classes in the confusion matrix in order to measure our model performance. A very popular metric for classifcation is the ROC (i.e. receiver operating characteristic) Curve and especially the area under this curve (AUC). This curve has to do with the threshold we set for our model. Using the logistic regression model, we have set our threshold at $p = 0.5$ ($p < 0.5$ indicates that the passenger has not survived and $p > 0.5$ that he has survived). We have also set a threshold for our KNN model. So, what happens if we vary this threshold? What happens to our True Positive and False Positive rates? When $p = 0$, the model predicts 1 for all the data, which means the True Positive rate is equal to our False Positive rate which is equal to 1. When we set $p = 1$, the model predicts 0 for all the data. Both True and False Positive rates are 0. If we vary the threshold, we get a series of different True Positive and False Positive rates. The series of points we get when trying all possible thresholds is called the ROC curve. If we plot the ROC curve for our predictions with KNN, we get the following:

```
from sklearn.model_selection import train_test_split
P_titanic_train, P_titanic_test, q_titanic_train, q_titanic_test = \
train_test_split(P_titanic,q_titanic, test_size=0.2, random_state=42)
prediction= knn.predict(P_titanic_test)
```

```
from sklearn.metrics import roc_curve, auc
false_positive_rate, true_positive_rate, thresholds = roc_curve(q_titanic_test, prediction)
plt.plot(false_positive_rate, true_positive_rate, label='KNN')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('KNN ROC Curve')
plt.show()
```



The larger the area under the ROC Curve, the better our model is. One way to understand this, is the following. We would have a great model if we had a model which produced an ROC Curve that had a single point in the upper left corner representing a True Positive rate of 1 and a False Positive Rate of 0. The ROC Curve is in the case of Figure 15, the red line. The area under this curve (the light blue square) is at it's maximum. Therefore AUC is another popular metric for classification models.

To compute our AUC score, we program the following code:

```
roc_auc = auc(false_positive_rate, true_positive_rate)
roc_auc
```

```
0.6708074534161491
```

If the AUC is greater than 0.5, it means that our model is better than just random guessing.

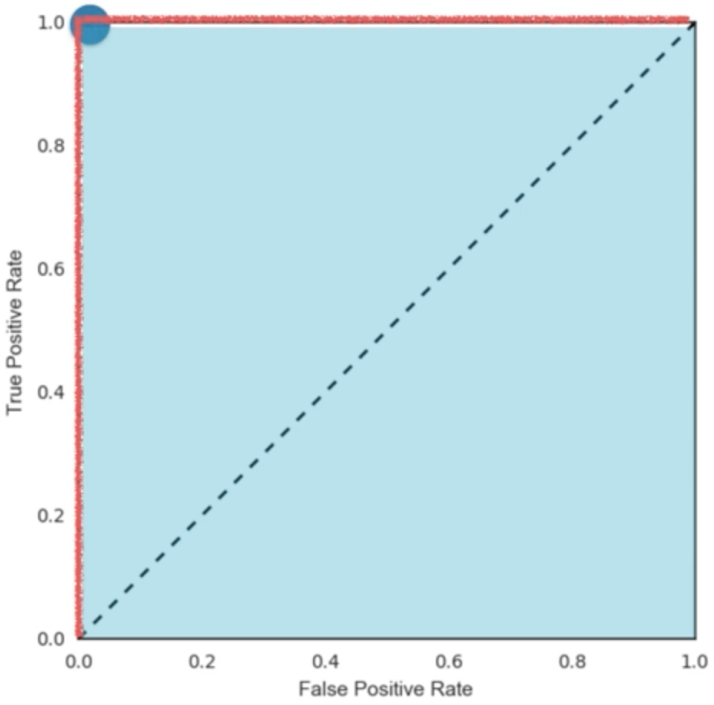


Figure 15: AUC

3.4 The second algorithm: Logistic Regression

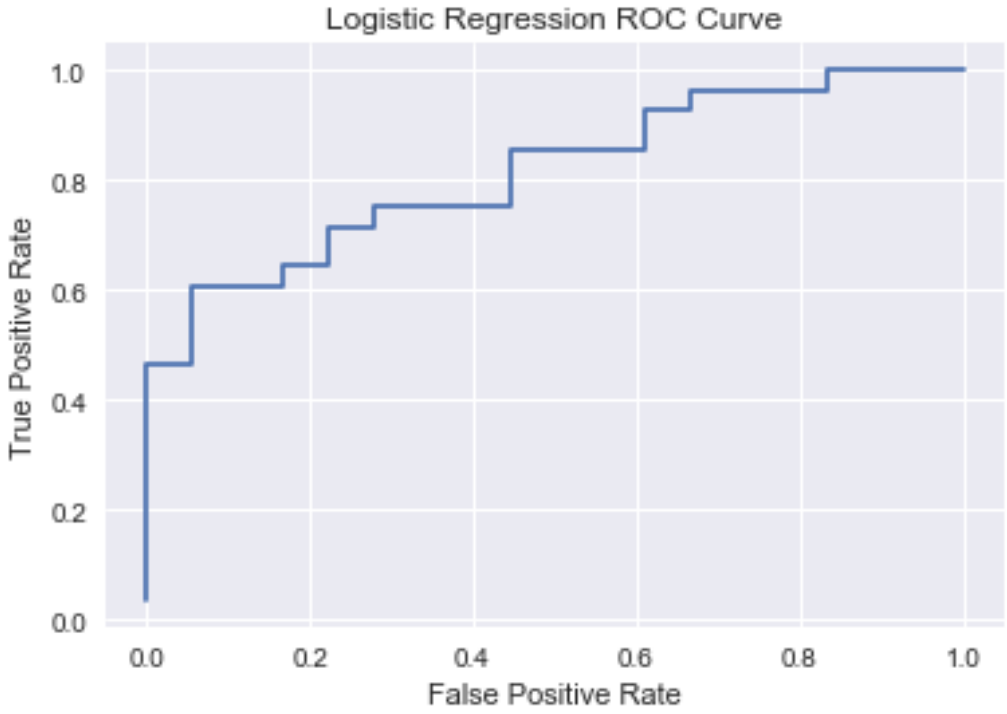
Another way to approach our problem is by using the algorithm logistic regression (logreg for short). This is the algorithm that outputs probablities, which is exactly what we need in order to answer our main- and subquestions. We follow almost the same procedure as we did with KNearestNeighbors. We import the regressor from the library `sklearn.linear_model`. Thereafter, we split our dataset into training and test set, perform k-fold cross-validation, fit our regressor to the training set and predict on the test set. We choose 0.25 for our test size.

```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
from sklearn.model_selection import train_test_split
P_titanic_train, P_titanic_test, q_titanic_train, q_titanic_test = \
train_test_split(P_titanic,q_titanic, test_size=0.25, random_state=42)
from sklearn.model_selection import cross_val_score
cv_scores = cross_val_score(logreg, P_titanic, q_titanic, cv=5, scoring='roc_auc')
logreg.fit(P_titanic_train, q_titanic_train)
ylog_pred = logreg.predict(P_titanic)
print(cv_scores)
```

```
[0.86666667 0.80333333 0.74666667 0.73263889 0.92361111]
```

Here we see our cross-validation scores. These show us how well our model performs and give us an indication about the fitting proces of our model. We see that these scores of R^2 are much higher than the ones we found using the algorithm KNearestNeighbors. This algorithm might be more helpful than KNN.

```
from sklearn.metrics import roc_curve, auc
y_pred_prob=logreg.predict_proba(P_titanic_test)[:,1]
false_positive_rate, true_positive_rate, thresholds = roc_curve(q_titanic_test, y_pred_prob)
plt.plot(false_positive_rate, true_positive_rate, label='LogisticRegression')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Logistic Regression ROC Curve')
plt.show()
```




```
roc_auc = auc(false_positive_rate, true_positive_rate)
roc_auc
```

```
0.8154761904761904
```

Our AUC score is also higher than the one we calculated while using KNN.

```
    print('Prediction{}'.format(ylog_pred))
```

```
Prediction[1 1 0 1 1 0 0 1 1 0 0 1 0 1 0 0 0 1 0 1 1 0 /
1 0 1 0 1 0 0 1 1 1 1 1 1 0 1
 1 0 1 0 0 1 0 1 0 0 1 1 0 1 1 0 1 1 1 /
 1 1 1 1 1 1 1 1 1 0 1 0 1 0 1 1 1 1
 1 1 0 1 1 1 1 1 0 0 0 1 1 1 0 0 0 0 0 0 1 /
 1 1 0 0 1 1 1 1 0 0 1 1 1 0 1 1
 0 1 1 1 1 0 1 1 0 1 0 1 0 1 1 0 0 1 0 0 1 0 0 /
0 0 0 1 0 1 0 0 0 1 0 0 1 0
 0 1 1 0 1 1 0 1 0 0 1 1 1 1 1 1 1 0 0 1 1 0 1 /
 1 1 1 1 0 1 0 1 0 1 1 0]
```

Here we see our first prediction using logistic regression. Once again this is the prediction for a fraction of 0.25 of our dataset. If we print our coefficients we get the following.

```
1 logreg.coef_
```

```
array([[ 0.07374214,  0.00377371, -0.00684224, -2.0694906 ]])
```

These coefficients correspond to the four columns of `P_titanic`, which are `Pclass`, `Fare`, `Age` and `male_dummy` respectively (as seen in Table 4). One can interpret the coefficients as follows: The higher your class, the higher your chance of survival. Same goes for `Fare`, because 0.00377371 is a positive number. We see that the coefficient corresponding to `Age` is negative, which indicates that the higher your age the lower your chance of surviving. In the case of `male_dummy`, the coefficient is negative as well which indicates that the chance of surviving decreases when `male_dummy` equals one.

If we take a look at the coefficient corresponding to `Pclass` we see something counterintuitive. The positive coefficient 0.07374214 suggests that the higher the class (in this case 3 is a higher class than 1), the higher the chance of survival. One might expect that the chance of survival is highest when travelling first class.

This paradox is resolved once we observe that the higher the fare, the higher the chance of survival. We have seen that plotting `Fare` against `Pclass` gives us a positive correlation. The coefficient of `Pclass` gives the effect of class on the chance of survival with a **given** fare, age and gender. This situation does not necessarily arise because there belongs a certain value of Fare to the first class: these two variables are positively correlated. When travelling first class instead of second class, two things change: the class and the price paid for a ticket (`Fare`). If we want to calculate the overall chance of surviving when travelling first class, we will have to take the effect of Fare into account as well.