Exploratory Data Analysis of Titanic data set

Fyodor Raevskiy[1]

[1]January 2022

***For correspondence:**
xboxraevskii@mail.ru (FMS);
@iwarnedyouaboutstairss
(Telegram)

[†]Project for Tinkoff Generation
[‡]The data set was taken from kaggle.com

Abstract In this research I want to analyse information about people on Titanic , we will understand who has survived and who has deceased , make some hypothesises on this topic and implement Logistic Regression which will predict a classification- survival or deceased.

Introduction

Hello!This is my first EDA so all calculations and conclusions I will do step by step and show you how I get them.We will be working with the Titanic Data Set from Kaggle. This is a very famous data set and very often is a student's first step in Exploratory Data Analysis and machine learning!

Firstly , we need to import some libraries that will help us in analysing data.

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

Reading the data

One of the most important first steps is to understand what data do we have. Let's start by reading in the "titanic train.csv" file into a pandas data frame. There are a lot of functions that generally return a pandas object, but in our case, we will use pandas.read-CSV() which is the most popular at newbies.

```
train = pd.read_csv("C:/datasets/train.csv")
```

How our data looks like

The easiest way to look on your data set is to show just a few of first lines of our data set.To do it let's print first 5 strings.

```
train.head()
```

What does each column mean?

Okay, now we have seen our information but I guess that reader needs some explanations about some of it.

Let's start with P-class. That column shows us in which class passenger was.

Then we can see this weird column which called parch. It shows number of brothers, sisters, step-brothers, stepsisters, spouses on board the Titanic.

Everyone understands what do fare and cabin mean but I guess not even a majority understood what does Embarked mean. 'Embarked' shows port of embarkation.Unexpectedly? 'C' is for Cherbourg , 'S' is for Southampton and 'Q' is for Queenstown.

Table 1. First 12 lines of Titanic data set.

Pass-ID	Surv	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embark
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
9	1	3	Johnson, Mrs. Oscar W	female	27.0	0	2	347742	11.1333	NaN	S
10	1	2	Nasser, Mrs. Nicholas	female	14.0	1	0	237736	30.0708	NaN	C
11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	G6	S
12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S

Source: https://www.kaggle.com/hesh97/titanicdataset-traincsv

Missing data

As we can see on our table , there are a lot of missing points about the cabin and the age of a passenger. And I would choose to rid of it but let me explain why. Getting rid of NaN objects in most cases caused simplicity. As data comes in many shapes and forms, we aim to find the easiest way of understanding statistics. We can use seaborn to create a simple heat map to see where we are losing information.

```
train.isnull()
```

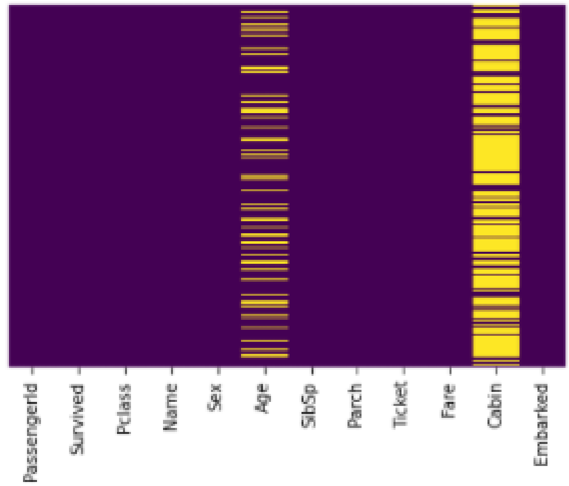
This function is very easy , it just checks our dataframe and shows True if this parameter is NaN and shows False if it's not.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	False	False	False	False	False	False	False	False	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True	False
887	False	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True	False
889	False	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True	False

For example , the first cell of Cabin is True , that means that we have no information about first passenger's cabin.
But as you already understood this is not the best way to remove missing data , cause it become more difficult if we have tons of information. So i offer you to use another method , method of visualisation . Let 's create a graphic that will show us which column has the most of missing data.

```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
#Some explanations: when I typed train.isnull() in brackets I said to sns library that it
    should take train.isnull() and whenever it's true sbs will display it in another color
     on graphic. The y axis is just all passengers but I typed yticklabels=False cause I
    do not want to see all of them.And the last two arguments are just for graph's
    appearance.
```

isNull graphic:



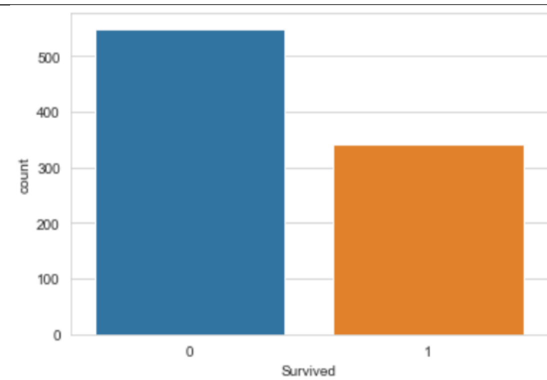
What we are supposed to do?

So, almost 20 percent of the data on the age of passengers is missing. But it seems to me that such a proportion is reasonable enough to replace this data with something sane. But we seem to have no information about the cabins at all , and therefore we will most likely get rid of this column or replace it with "Is there information about the cabin: 1 for yes and 0 for no"

Let’s continue visualising some more of the data.

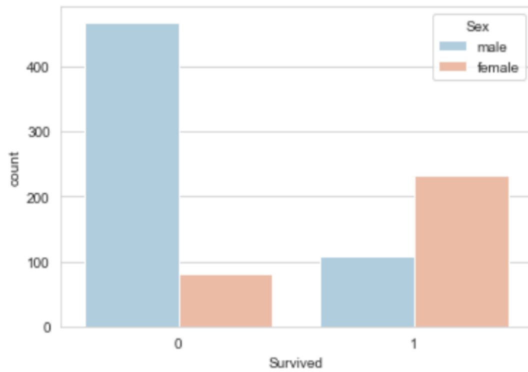
Let’s find out how many people survived on Titanic

```
sns.set_style('whitegrid') #it will create beatiful grid
sns.countplot(x='Survived',data=train)
#Some explanations : depending on Survived column i will create a graph that will show how
     many people died (how many 0 does 'survived' column have) and how many people survived
```



As you can see a lot of people did not survived , i would rather say the majority of passengers did not survived.Let's see did more men or women survive?

```
sns.countplot(x='Survived',hue='Sex',data=train,palette='RdBu_r')
```

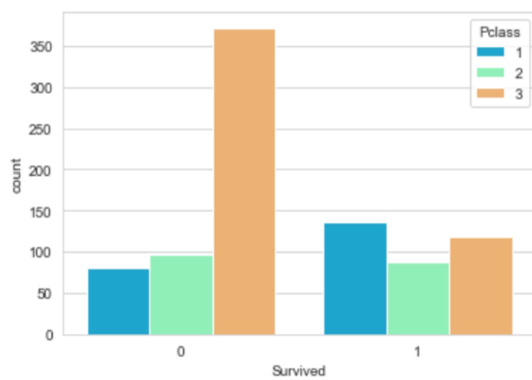


So most of men died and at least 80 women also died.Yes , it 's a pity , but there 's nothing we can do about it.

Comparing different classes

Let's see which class of passengers has survived the most and which the least.

```
sns.countplot(x='Survived',hue='Pclass',data=train,palette='rainbow')
#Some explanations: now I typed to sbs that it must take column 'Survived' and count,
     depending on column 'Pclass', how many passengers of each class have survived.
```

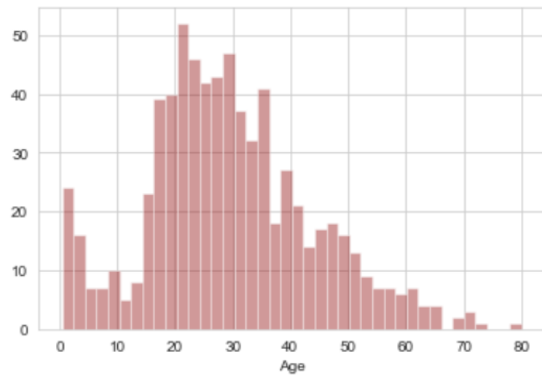


Pretty exiting!The passengers who died the most belonged to class three which is the lowest one. Another interesting observation is that nearly 60 per cents of class 2 passengers died , even third-class passengers survived more, although in proportion, of course, third-class passengers died more often.

What is the average age of Titanic passenger

Now I want to find out people of what age were on Titanic the most. Let's use function of seaborn that shows distribution of values.

```
sns.distplot(train['Age'].dropna(),kde=False,color='darkred',bins=40)
#Some explanations: I want transmit column 'Age' to sns but without NaN objects , to do it
     I need to type .dropna() after our data.The kda parameter is false cause i do not want
      to see kernel density estimation on our graph.
```

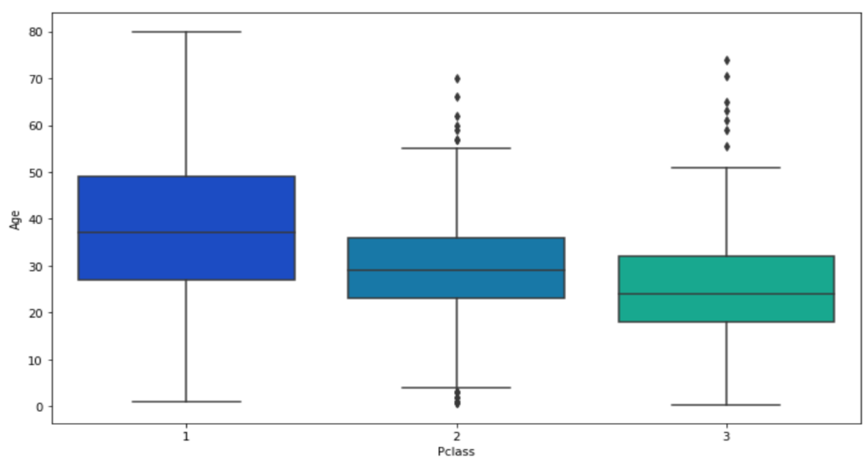


As you can see the average age was around 20-30 .

Data cleaning

We want to fill in missing age data instead of just dropping the missing age data rows. One way to do this is by filling in the mean age of all the passengers. But this is a too wild way of imputation so we will use another method: we will understand what is the average age of each class and only after that we will fill in the missing data.

```
plt.figure(figsize=(12, 7))
sns.boxplot(x='Pclass',y='Age',data=train,palette='winter')
#Some explanations: we will use .boxplot , the x axis will be 3 our classes and the y axis
    will be age.
```



This boxplot give us a lot of information . These black lines on each box is the average value of this class. So depending on this information we will replace every NaN value in 'Age' column. Let's take 36 as average age of 1st class passenger , 29 as average age of 2nd class passenger and 24 as average age of 3rd class passenger.

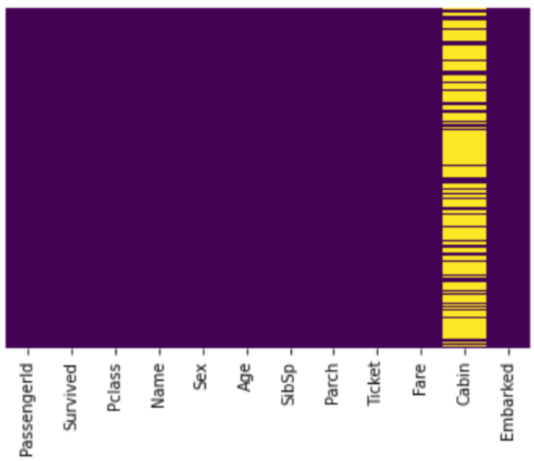
```
def impute_age(cols):
    Age = cols[0]
    Pclass = cols[1]

    if pd.isnull(Age):
        if Pclass == 1:
            return 36
        elif Pclass == 2:
            return 29
        else:
            return 24

    else:
        return Age
```

Now we will apply this function to our data set.

```
train['Age'] = train[['Age','Pclass']].apply(impute_age,axis=1)
#Some explanations: I used function impute-age by built-in function apply and transmit all
    the necessary values.
```

Let’s look at our heatmap of isNaN again.

```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```



Great! The last thing that we need to do is remove ’Cabin’ column at all because there is too little information about it. Also I will show how our data now look like.

```
train.drop('Cabin',axis=1,inplace=True) #removing 'cabin'
train.head()
```

Pass-ID	Surv	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S
2	1	1	Cumings, Mrs. John Bradley	female	38.0	1	0	PC 17599	71.2833	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S
4	1	1	Futrelle, Mrs. Jacques Heath	female	35.0	1	0	113803	53.1000	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S
6	0	3	Moran, Mr. James	male	24	0	0	330877	8.4583	Q
7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	S
8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	S
9	1	3	Johnson, Mrs. Oscar W	female	27.0	0	2	347742	11.1333	S
10	1	2	Nasser, Mrs. Nicholas	female	14.0	1	0	237736	30.0708	C
11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	S
12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	S

Conlusions and hypothesizing

We understood that:

I On Titanic, men died in the majority.

II Third-class passengers had almost no chance of survival.

III Young people were in greater danger than old people cause in most cases young people were 3rd class passengers.

And after this thoughts I want to make a hypothesis:if you are a passenger of the Titanic and want to survive with the greatest probability, you should be the little daughter of very rich parents.

Hypothesis testing

We start this analysis by adding a new column to the 'train data frame'. Use the Survived column to map to the new column with factors 0 for 'no' and 1 for 'yes' using the map method

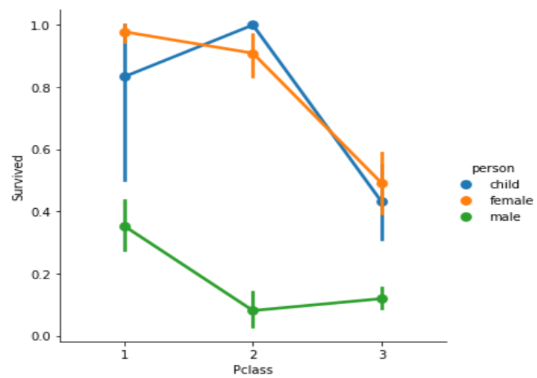
```
train['Survivor'] = train.Survived.map({0:'no', 1:'yes'})
```

Also let's add a 'Person' column which will contain three types : male , female or child.

```
# Function which will determine is this passenger a child
def whoIsPerson(passenger):
    age, sex = passenger

    if age < 16:
        return 'child'
    else:
        return sex


train['person'] = train[['Age', 'Sex']].apply(whoIsPerson, axis=1)
```

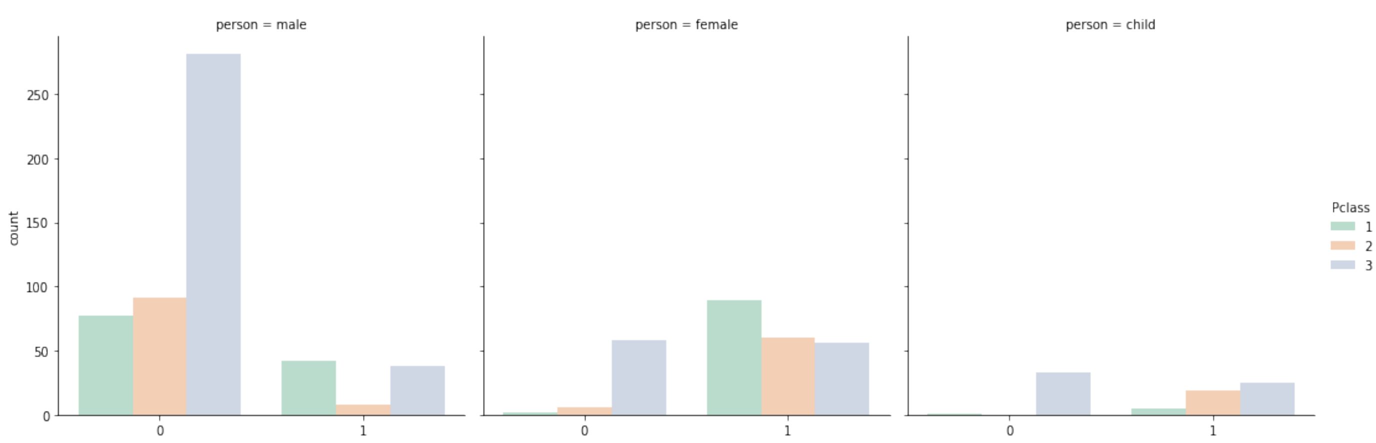
Now let's see how graph with information about class/gender and survival looks like.

```
sns.factorplot('Pclass','Survived', hue='person', data=train, order=range(1,4),
               hue_order = ['child','female','male'])
```



From the graph above, it is clear that being a man and even a third class greatly reduces the chances of survival

```
sns.factorplot('Survived', data=train, hue='Pclass', kind='count', palette='Pastel2',
     hue_order=range(1,4),
              col='person')
```



And last thing that we need to prove is correlation between class and survival (as I said you should be daughter of RICH parents who would probably buy seats at first class)

```
sns.lmplot('Age', 'Survived', data=train, hue='Sex')
sns.lmplot('Age', 'Survived', hue='Pclass', data=train, palette='winter', hue_order=range
    (1,4))
```

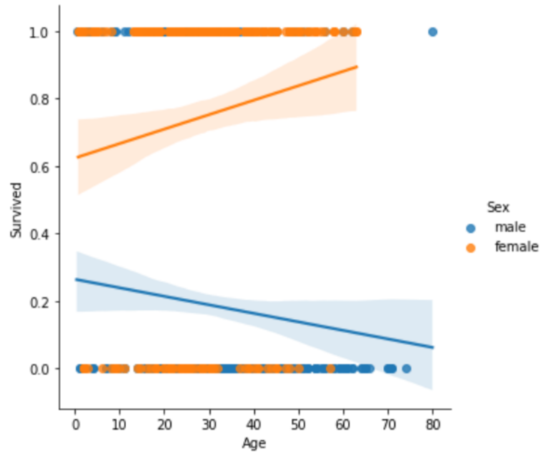


Figure 1. Age is X axis , Survived is Y axis grouped by sex

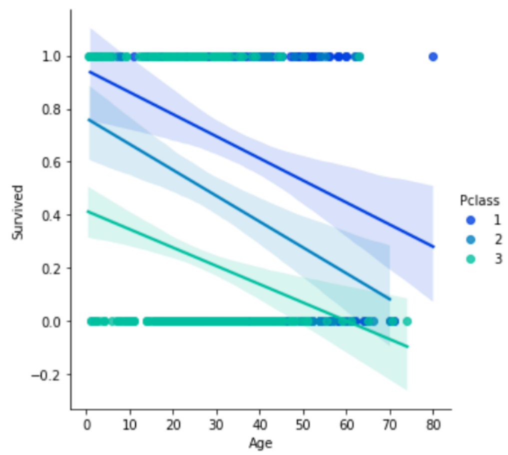


Figure 2. Age is X axis , Survived is Y axis grouped by class

As Graph 1 above showed : older women are more likely to survive than older men , but the second graph shows that the probability that a person will survive decreases with increasing age. I guess that all this information is enough to make a results review and the final conclusion.

Results , conclusion

I hypothesized about the survival factors on board the Titanic . Let me briefly remind you: *in order to survive, it is desirable to be a little girl of rich parents*. And as we learned and proved by statistics, children had more chances of survival than men, namely, *the chances of survival decrease with increasing age*.

Then we needed to confirm that women, on average, were saved more often than men in most cases. Graphs show that a woman of any age was saved more often than a man. It remains to prove the relationship between survival and class, but it was not difficult. As it turned out, almost all the passengers of the 3rd class could not escape, more than half of the second class were also unlucky, but the third class passengers showed the best "survival rates". So my hypothesis was confirmed, which is good news. However, in the course of work, I learned a lot of sad facts that we can prevent in the future.

The main conclusion that can be drawn based on my hypothesis is the following one: Passengers of any class, age or gender should have an equal chance of salvation.

The end

Links:

1. Code on GitHub: https://github.com/FyodoRaev/TitanicDat
2. EDA guides that i used : https://youtu.be/-o3AxdVcUtQ , https://youtu.be/Ea_KAcdv1vs
3. Pandas library documentation : https://pandas.pydata.org/pandas-docs/stable/user_guide/missing_data.html
4. Seaborn library documentation : https://seaborn.pydata.org/