

Exploratory Data Analysis of Titanic data set

Fyodor Raevskiy¹

¹January 2022

*For correspondence:

xboxraevskii@mail.ru (FMS);

[@iwarnedyouaboutstairss](https://www.instagram.com/iwarnedyouaboutstairss)

(Telegram)

[†]Project for Tinkoff Generation

[‡]The data set was taken from [kaggle.com](https://www.kaggle.com)

Abstract I am writing this report to analyze information about people who died and survived on Titanic, make some hypotheses on this topic and try to prove it by statistics.

Introduction

This is my first EDA so all calculations and conclusions I will do step by step and show you how I got them. I used the Titanic Data Set from Kaggle. This is a very famous data set and it is frequently a student's first step in Exploratory Data Analysis and machine learning.

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

```
14 1 import pandas as pd
15 2 import numpy as np
16 3 import matplotlib.pyplot as plt
17 4 import seaborn as sns
18 5 %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.

```
24 1 train = pd.read_csv("C:/datasets/train.csv")
```

How our data looks like

The easiest way to look on the data set is to show just a few of the first lines of the table. To begin with, let's print first 12 strings.

```
28 1 train.head()
```

What does each column mean?

After seeing all the information, the reader will need some explanations.

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

There is a column which is called parch. It shows number of brothers, sisters, stepbrothers, step-sisters, spouses on the board of Titanic.

'Embarked' shows port of embarkation. '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	Embarked
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>

36 Missing data

37 According to the table 1, there are a lot of missing points about the cabin and the age of a passenger.
 38 Moreover, I would like to remove them and I will explain why. Getting rid of NaN objects in most
 39 cases caused by simplicity. As data comes in many shapes and forms, we aim to find the easiest
 40 way of understanding statistics. We can use seaborn to create a simple heat map to see where the
 41 information is missing.

```
42 1 train.isnull()
```

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

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
	0	False	False	False	False	False	False	False	False	False	True	False
	1	False	False	False	False	False	False	False	False	False	False	False
	2	False	False	False	False	False	False	False	False	False	True	False
	3	False	False	False	False	False	False	False	False	False	False	False
	4	False	False	False	False	False	False	False	False	False	True	False

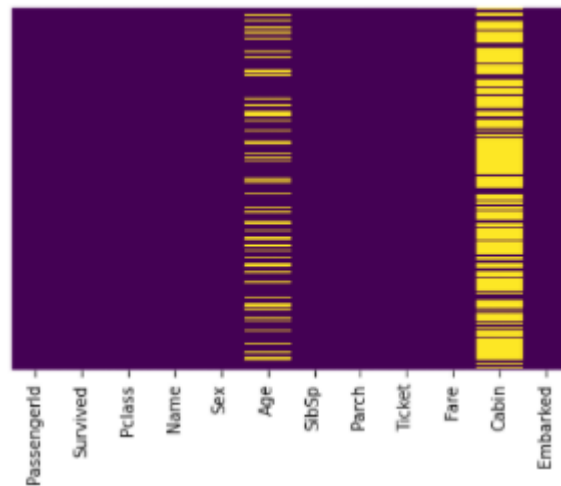
	886	False	False	False	False	False	False	False	False	False	True	False
	887	False	False	False	False	False	False	False	False	False	False	False
	888	False	False	False	False	True	False	False	False	False	True	False
	889	False	False	False	False	False	False	False	False	False	False	False
	890	False	False	False	False	False	False	False	False	False	True	False

45
 46 For example , the first cell of Cabin is True , that means that we have no information about first
 47 passenger's cabin.

48 However, this is not the best way to remove missing data because it becomes more complicated
 49 , if there is a lot of information. So, I suggest using method of visualisation . Let 's create a graph
 50 that will show which column has the most of the missing data.

```
51 1 sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
52 2 #Some explanations: when I typed train.isnull() in brackets I meant that it should take
53 train.isnull() and whenever it's true, sns will display it in yellow color on a graph.
54 The y axis is all passengers but I typed yticklabels=False because I do not want to
55 see all of them.And the last two arguments are for graph's appearance.
```

56 *isNull graphic:*



57

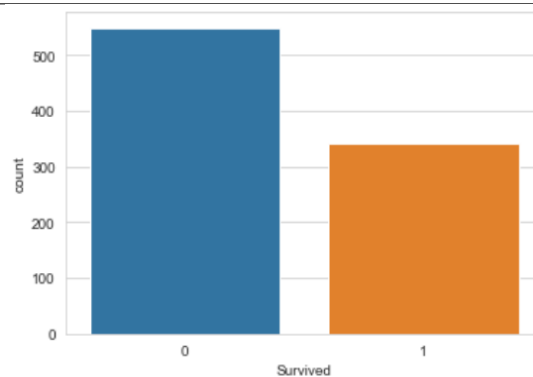
58 **What we are supposed to do?**

59 Therefore, almost 20 percent of the age of passengers is missing. But it seems to me that such a
60 proportion is reasonable enough to replace this data with something sensible. But there is a lack
61 of information about the cabins, and I will most likely get rid of this column or replace it with "Is
62 there information about the cabin: 1 for yes and 0 for no".

63 **Let's continue visualising some more of the data.**

64 Let's find out how many people survived on Titanic

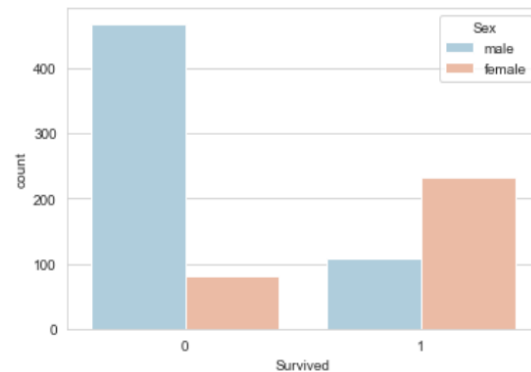
```
65 1 sns.set_style('whitegrid') #it will create beautiful grid
66 2 sns.countplot(x='Survived',data=train)
67 3 #Some explanations : depending on Survived column I will create a graph that will show how
68 4 many people died (how many 0 does 'survived' column have) and how many people survived
```



69

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

```
72 1 sns.countplot(x='Survived',hue='Sex',data=train,palette='RdBu_r')
```



73

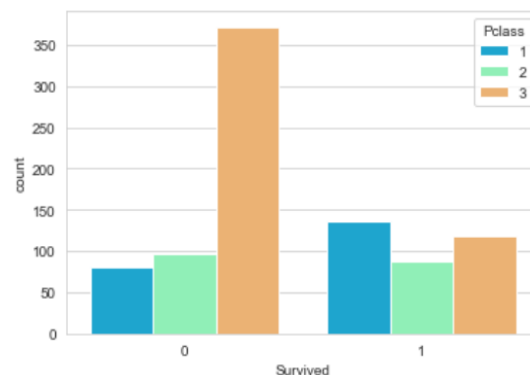
74 So most of men died and at least 80 women also died.Yes , it's a pity , but there's nothing we can
75 do about it.

76 Comparing different classes

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

```
78 1 sns.countplot(x='Survived',hue='Pclass',data=train,palette='rainbow')
```

```
79 2 #Some explanations: now I typed to sbs that it must take column 'Survived' and count,  
80     depending on column 'Pclass', how many passengers of each class have survived.
```



81

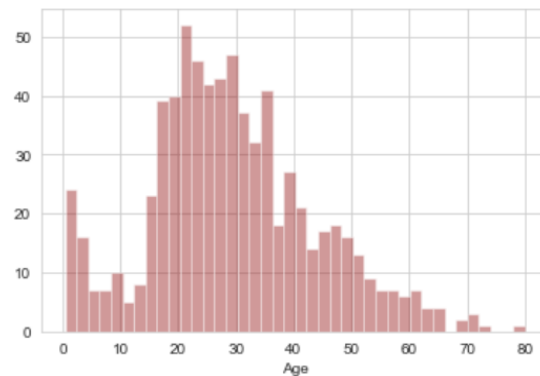
82 Pretty exiting!The passengers who died the most belonged to class three which is the lowest one.
83 Another interesting observation is that nearly 60 per cents of class 2 passengers died , even third-
84 class passengers survived more, although in proportion, of course, third-class passengers died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

86 What is the average age of Titanic passenger

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

```
89 1 sns.distplot(train['Age'].dropna(),kde=False,color='darkred',bins=40)
```

```
90 2 #Some explanations: I want transmit column 'Age' to sns but without NaN objects , to do it  
91     I need to type .dropna() after our data.The kda parameter is false cause I do not want  
92     to see kernel density estimation on our graph.
```



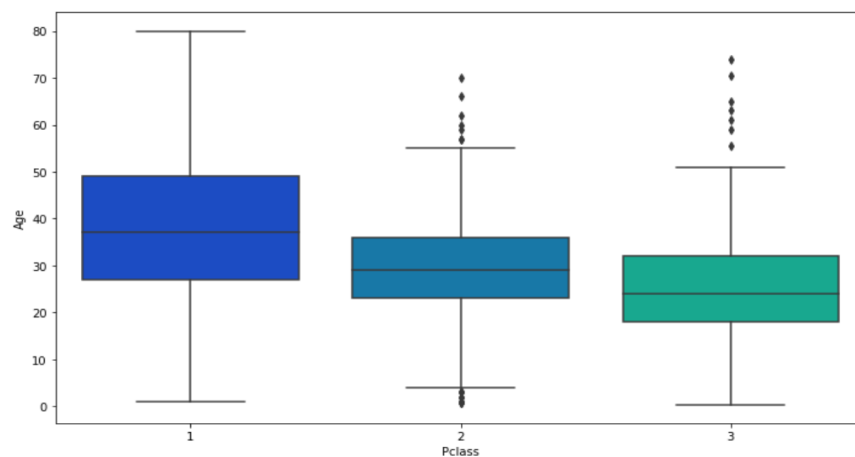
93

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

95 Data cleaning

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

```
100 1 plt.figure(figsize=(12, 7))
101 2 sns.boxplot(x='Pclass',y='Age',data=train,palette='winter')
102 3 #Some explanations: we will use .boxplot , the x axis will be 3 our classes and the y axis
103 4 will be age.
```



104

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

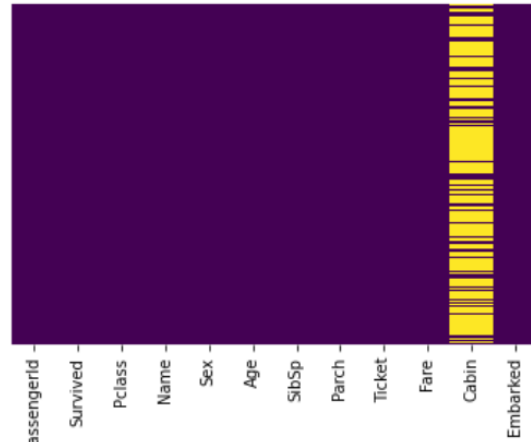
```
109 1 def impute_age(cols):
110 2     Age = cols[0]
111 3     Pclass = cols[1]
112 4
113 5     if pd.isnull(Age):
114 6         if Pclass == 1:
115 7             return 36
116 8         elif Pclass == 2:
117 9             return 29
118 10        else:
119 11            return 24
120 12
121 13    else:
122 14        return Age
```

123 Now we will apply this function to our data set.

```
124 1 train['Age'] = train[['Age', 'Pclass']].apply(impute_age,axis=1)
125 2 #Some explanations: I used function impute-age by built-in function apply and transmit all
126     the necessary values.
```

127 Let's look at our heatmap of isNaN again.

```
128 1 sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```



129
130 Great! The last thing that we need to do is remove 'Cabin' column at all because there is too little
131 information about it. Also I will show how our data now look like.

```
132 1 train.drop('Cabin',axis=1,inplace=True) #removing 'cabin'
133 2 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

135 Conlusions and hypothesizing

136 We understood that:

137 I On Titanic, men died in the majority.

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

139 III Young people were in greater danger than old people cause in most cases young people were
140 3rd class passengers.

141 After these thoughts I want to make a hypothesis:among the survivors , most of all children and
142 elderly women , could this mean that human morality contributed to the salvation of these people
143 ?Perhaps, quite others would have survived, if not for the nobility of people of the 3rd class..

144 Hypothesis testing

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

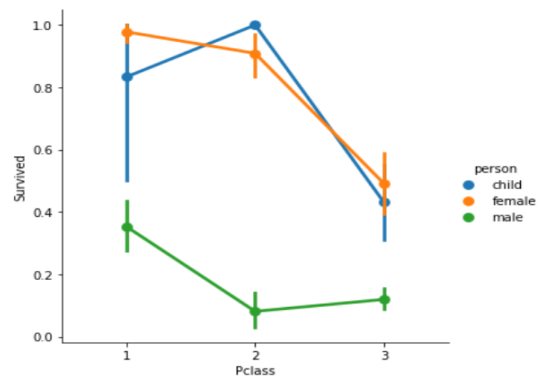
```
147 1 train['Survivor'] = train.Survived.map({0:'no', 1:'yes'})
```

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

```
149 1 # Function which will determine is this passenger a child
150 2 def whoIsPerson(passenger):
151 3     age, sex = passenger
152 4
153 5     if age < 16:
154 6         return 'child'
155 7     else:
156 8         return sex
157 9
158 10
159 11 train['person'] = train[['Age', 'Sex']].apply(whoIsPerson, axis=1)
```

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

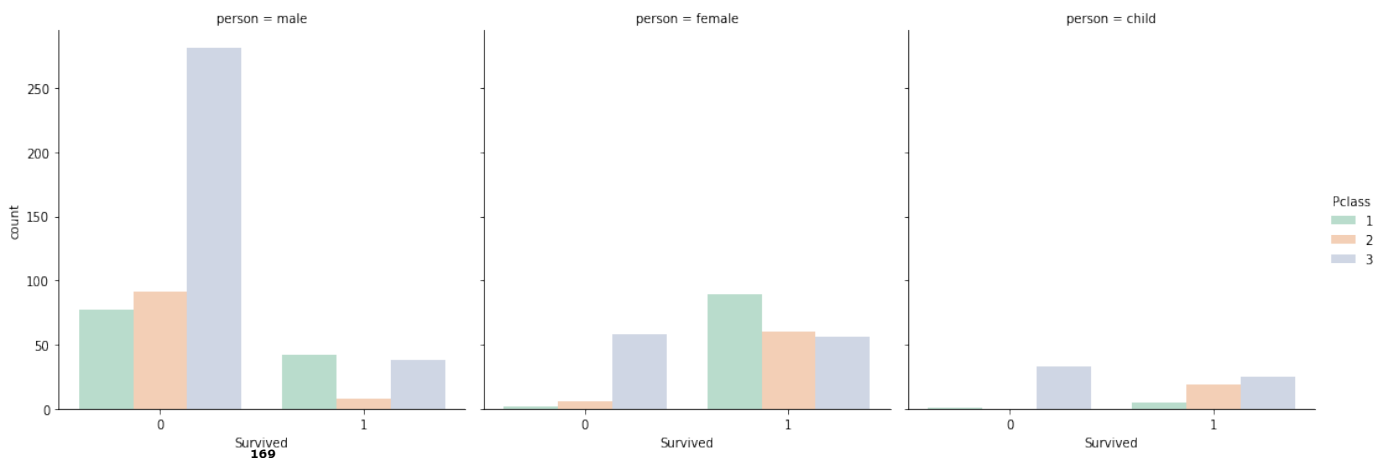
```
161 1 sns.factorplot('Pclass', 'Survived', hue='person', data=train, order=range(1,4),
162 2               hue_order = ['child', 'female', 'male'])
```



163

164 The graph above confirms the assumption that men of the 3rd class survived much less often than
165 passengers of the 1st class.

```
166 1 sns.factorplot('Survived', data=train, hue='Pclass', kind='count', palette='Pastel2',
167 2               hue_order=range(1,4),
168 3               col='person')
```



170 Now I will show the correlation between the survival rate of a passenger and his age, gender
 171 and class

```
172 1 sns.lmplot('Age', 'Survived', data=train, hue='Sex')
173 2 sns.lmplot('Age', 'Survived', hue='Pclass', data=train, palette='winter', hue_order=range
174 (1,4))
```

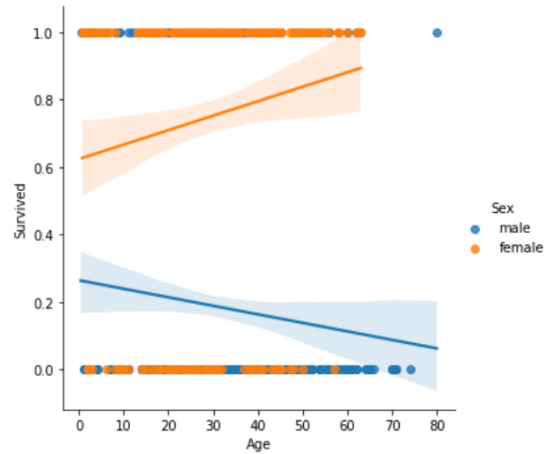


Figure 1. Age is X axis , Survived is Y axis grouped by sex

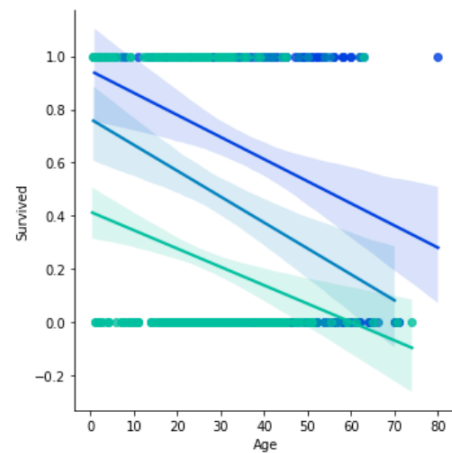


Figure 2. Age is X axis , Survived is Y axis grouped by class

175

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

179 **Results , conclusion**

180 To conclude, I assumed that it was morality that prevented more of the 3rd class from surviving. My
181 arguments are fully supported by the data because we found out that elderly women and children
182 were rescued much more often. How important is this fact? Today, predictions and instructions for
183 rescue services must view morality as a very important factor in human behaviour. For example,
184 when designing features or machine learning algorithms, our goal is usually to isolate the factors
185 of variation that explain the observed data. Accordingly, my hypothesis can help analysts of rescue
186 services in preventing dangerous events, which, in my opinion, is important in the modern world.

187 **The end**

188 Links:

- 189 1. Code on GitHub: <https://github.com/FyodoRaev/TitanicData>
- 190 2. Pandas library documentation : [https://pandas.pydata.](https://pandas.pydata.org/pandas-docs/stable/user_guide/missing_data.html)
- 191 [org/pandas-docs/stable/user_guide/missing_data.html](https://pandas.pydata.org/pandas-docs/stable/user_guide/missing_data.html)
- 192 3. Seaborn library documentation : [https://seaborn.](https://seaborn.pydata.org/)
- 193 [pydata.org/](https://seaborn.pydata.org/)