by. Null Science

16 April 2022

DATA CLEANSING & EXPLORATORY DATA ANALYSIS

Learning Python #6

by. Null Science

16 April 2022

NULL SCIENCE TEAM

Leader : Fahmi Hamzah

Member :

1. Aditya Hermawan
2. Fara Rizkhi Karunia
3. Nafiatul Risa
4. Salsabilla Atasyaputri Setyawan

by. Null Science

16 April 2022

BACKGROUND

General Overview

In this presentation, we will share about simple data cleansing and exploratory data analysis using Titanic.csv data.

History

The Titanic.csv file contains data for 891 Titanic passengers. Each row represents one person. The columns describe different attributes about the person including whether they survived, their age, their passenger-class, their sex, and the fare they paid.

by. Null Science

16 April 2022

DATA CLEANSING

Import several libraries.

- pandas = microsoft excel - data Tabular
- numpy = numerical computation
- matpotlib, seaborn = Visualization

Source Code

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

DATA CLEANSING

Beautiful Grateful Chill

Make a data frame than load a dataset

Source Code :

```
[2] from google.colab import files
    files.upload()
```

Pilih File Tidak ada file yang dipilih Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Titanic.csv to Titanic.csv
{'Titanic.csv': b'PassengerId,Survived,Pclass,Name,Sex,Age,SibSp,Parch,Ticket,Fare,Cabin,Embarked\r\n1,0,3,"Braund, Mr. Owen Harris",male,22,1,0,A/5 21171,7.25,,S
```

```
[3] df = pd.read_csv("Titanic.csv")
```

Learning Python #6

DATA CLEANSING

Show Dataset

```
[4]  df.head() #--> 5 top
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
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

```
[5]  df.tail() #--> 5 Bottom
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

DATA CLEANSING

Show Information Dataset (Garbage in Garbage Out)

```
[6]  df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

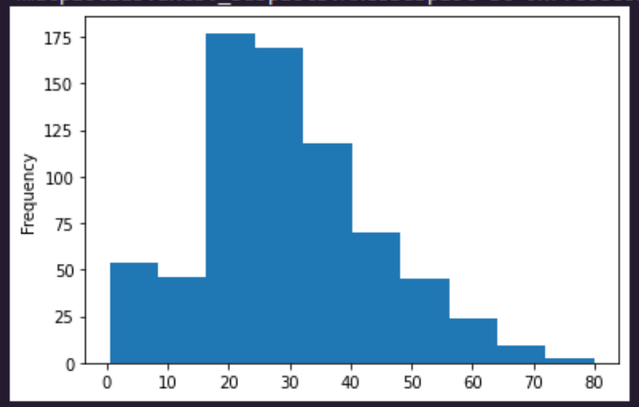
- Your analysis is as good as your data

DATA CLEANSING

Column Age

```
[7]  #df.Age.plot(kind='hist'); #--> histogram
     df['Age'].plot(kind='hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc0ec6d7750>
```




```
[8]  df['Age'].value_counts() #data count checking
```

```
24.00    30
22.00    27
18.00    26
19.00    25
28.00    25
         ..
36.50     1
55.50     1
0.92      1
23.50     1
74.00     1
Name: Age, Length: 88, dtype: int64
```

Show Visualization

DATA CLEANSING

Column Age

Fast Insight Taking

```
[9]  df['Age'].describe()

count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

Column Age distribution skewness/kurtosis
--> inputation median

```
[10]  val = df.Age.median()
      df['Age'] = df.Age.fillna(val)
```

Checking Column

```
[11]  df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

DATA CLEANSING

Column Cabin

```
[12] df.Cabin.value_counts()

     B96 B98        4
     G6             4
     C23 C25 C27    4
     C22 C26        3
     F33            3
                   ..
     E34            1
     C7             1
     C54            1
     E36            1
     C148           1
     Name: Cabin, Length: 147, dtype: int64
```

Entry data 891, Cabin has only 204 data.

Too much data unique. Not Really informative for knowing Survived data.

Drop column Cabin

```
[13] df.drop('Cabin', axis=1, inplace = True) #--> True (permanent)
```

Data Bayes --> not know it's data real or fake.

```
[14] df.info()

     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 891 entries, 0 to 890
     Data columns (total 11 columns):
      #   Column       Non-Null Count  Dtype
     ---  ------       --------------  -----
      0   PassengerId  891 non-null    int64
      1   Survived     891 non-null    int64
      2   Pclass       891 non-null    int64
      3   Name         891 non-null    object
      4   Sex          891 non-null    object
      5   Age          891 non-null    float64
      6   SibSp        891 non-null    int64
      7   Parch        891 non-null    int64
      8   Ticket       891 non-null    object
      9   Fare         891 non-null    float64
      10  Embarked     889 non-null    object
     dtypes: float64(2), int64(5), object(4)
     memory usage: 76.7+ KB
```

Checking Data

DATA CLEANSING

Learning Python #6

Column Embarked

891 entry data, has 889 data

```
[15] df.Embarked[df.Embarked.isnull()]

     61     NaN
     829    NaN
     Name: Embarked, dtype: object
```

```
[16] df.Embarked.isnull()

     0      False
     1      False
     2      False
     3      False
     4      False
            ...
     886    False
     887    False
     888    False
     889    False
     890    False
     Name: Embarked, Length: 891, dtype: bool
```

Show Proporsi Data, column Embarked as a Data Categoric (having class)

```
[17] df.Embarked.value_counts() #Proporsi Data

     S    644
     C    168
     Q     77
     Name: Embarked, dtype: int64
```

Imputation on column Embarked. as Data Categorical we are using mode (modus)

```
[18] val = df.Embarked.mode().values[0]
     df['Embarked'] = df.Embarked.fillna(val)
```

Data Proportion

```
[19] df.Embarked.value_counts()

     S    646
     C    168
     Q     77
     Name: Embarked, dtype: int64
```

```
[20] df.info()

     <class 'pandas.core.frame.DataFrame'>
     RangeIndex: 891 entries, 0 to 890
     Data columns (total 11 columns):
      #   Column       Non-Null Count  Dtype
     ---  ------       --------------  -----
      0   PassengerId  891 non-null    int64
      1   Survived     891 non-null    int64
      2   Pclass       891 non-null    int64
      3   Name         891 non-null    object
      4   Sex          891 non-null    object
      5   Age          891 non-null    float64
      6   SibSp        891 non-null    int64
      7   Parch        891 non-null    int64
      8   Ticket       891 non-null    object
      9   Fare         891 non-null    float64
      10  Embarked     891 non-null    object
     dtypes: float64(2), int64(5), object(4)
     memory usage: 76.7+ KB
```

DATA CLEANSING

1 Beautiful

Treat Yourself 2 Grateful

3 Chill

Column SibSp and Column Parch

Doing data manipulation, not change a data value but for easiest read by machine.

```
[23] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Embarked     891 non-null    object
 11  Alone        891 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Column SibSp (Sibling Spouse) --> the number of relatives or spouses carried by Passenger.
Column Parch (Parent Chidren) --> number of parents or number of children carried by Passenger.

```
[21] #add new Column as Alone
     df['Alone'] = df['SibSp']+df['Parch']


[22] df['Alone'][df['Alone']>0]='With Family'
     df['Alone'][df['Alone']==0]='Without Family'

     /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
     A value is trying to be set on a copy of a slice from a DataFrame

     See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
       """Entry point for launching an IPython kernel.
     /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
     A value is trying to be set on a copy of a slice from a DataFrame

     See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
[24] df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	Alone
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	S	With Family
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C	With Family
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	S	Without Family
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	S	With Family
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	S	Without Family

by. Null Science

16 Aprill 2022

RELATION BETWEEN COLUMN SEX AND COLUMN SURVIVED

1

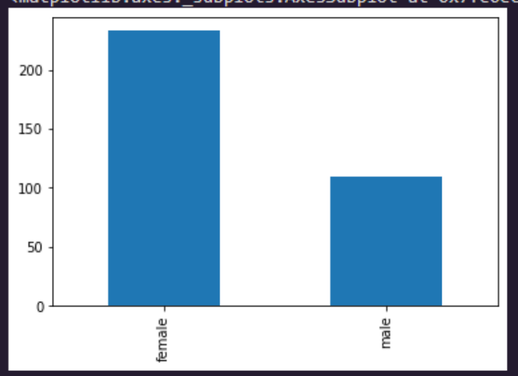
```
[25] df.Sex[df['Survived']==1].value_counts()

female    233
male      109
Name: Sex, dtype: int64
```

2

```
[26] df.Sex[df['Survived']==1].value_counts().plot(kind='bar')

<matplotlib.axes._subplots.AxesSubplot at 0x7fc0ec5d6c90>
```



by. Null Science

16 Aprill 2022

EXPLORATORY DATA ANALYSIS

Let's Start

by. Null Science

16 Aprill 2022

```python
#import library required
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.neighbors import KNeighborsClassifier
```

SHOW DATA

```
df = pd.read_csv("Titanic.csv", index_col="PassengerId")
df.head()
```

PassengerId	Survived	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 (Florence Briggs Th...	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 (Lily May Peel)	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

by. Null Science

16 Aprill 2022

DROP UNNEEDED DATA (NAME, TICKET, CABIN

```
#import data and drop data
df.drop(columns = ["Name", "Ticket", "Cabin"], inplace = True)
df.head()
```

PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
1	0	3	male	22.0	1	0	7.2500	S
2	1	1	female	38.0	1	0	71.2833	C
3	1	3	female	26.0	0	0	7.9250	S
4	1	1	female	35.0	1	0	53.1000	S
5	0	3	male	35.0	0	0	8.0500	S

by. Null Science

TARGET

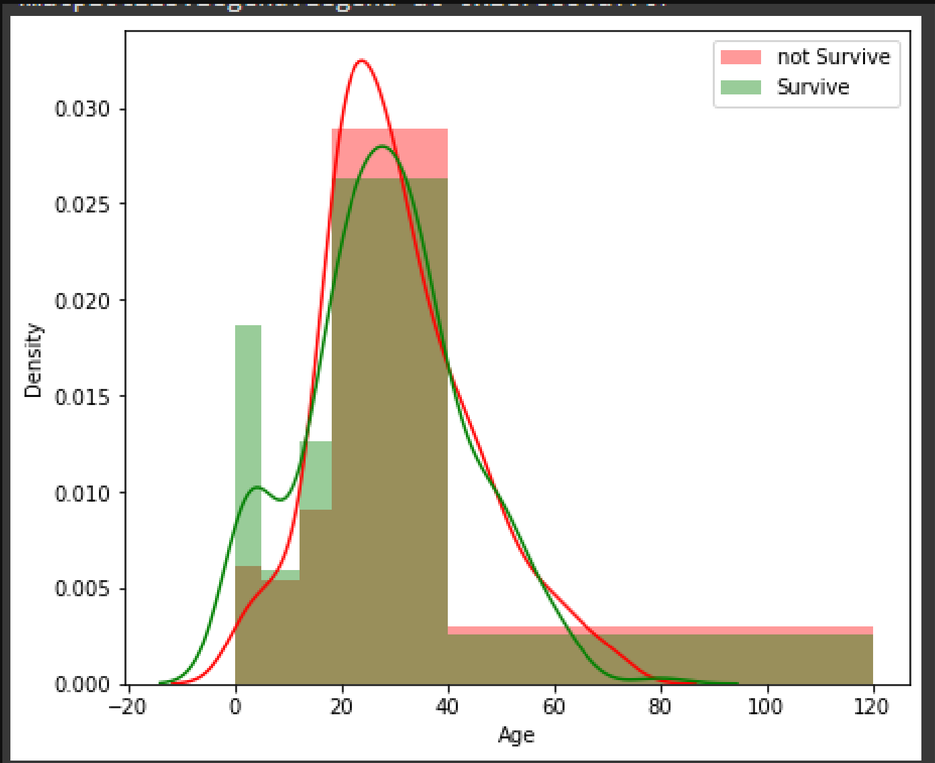
16 Aprill 2022

```
#target
df.Survived.value_counts()
```

```
0    549
1    342
Name: Survived, dtype: int64
```

The insight obtained from the data the number of Survived is 342 and those who are not 549

```
#numeric vs target
plt.figure(figsize=(7, 6))
sns.distplot(df.Age[df.Survived == 0], bins= [0, 5, 12, 18, 40, 120], color="r", label = "not Survive")
sns.distplot(df.Age[df.Survived == 1], bins= [0, 5, 12, 18, 40, 120], color="g", label = "Survive")
plt.legend()
```

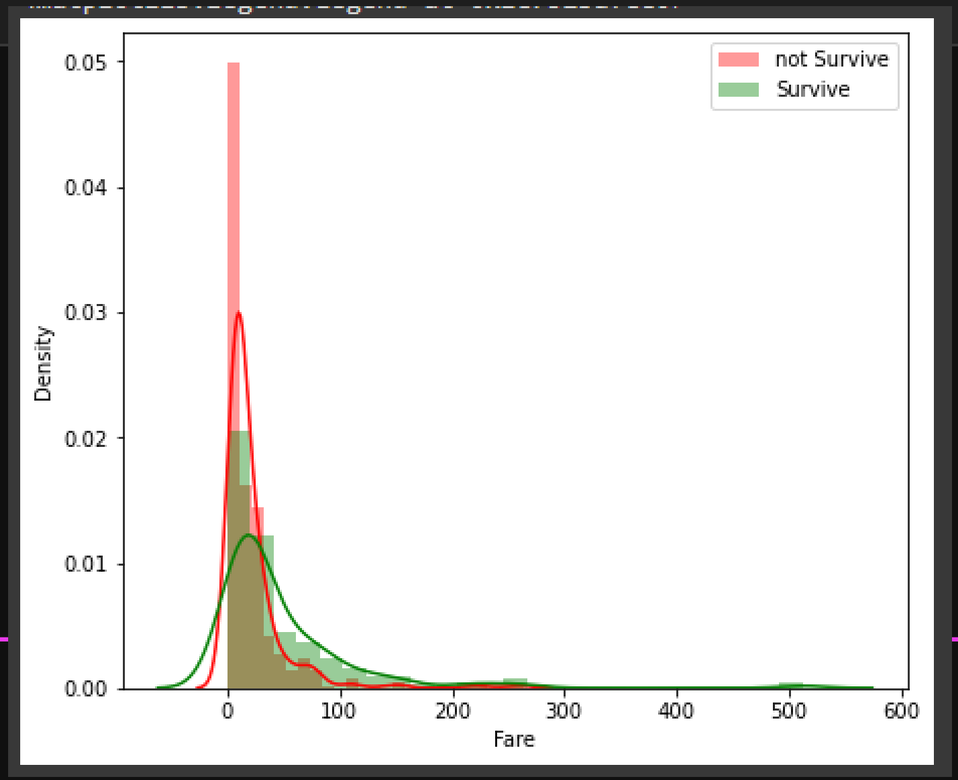


by. Null Science

16 Aprill 2022

The insight obtained from the data is that infants are more likely to survive and adults (20 - 40) are more likely to be unsafe. you could say that rescue is prioritized for children

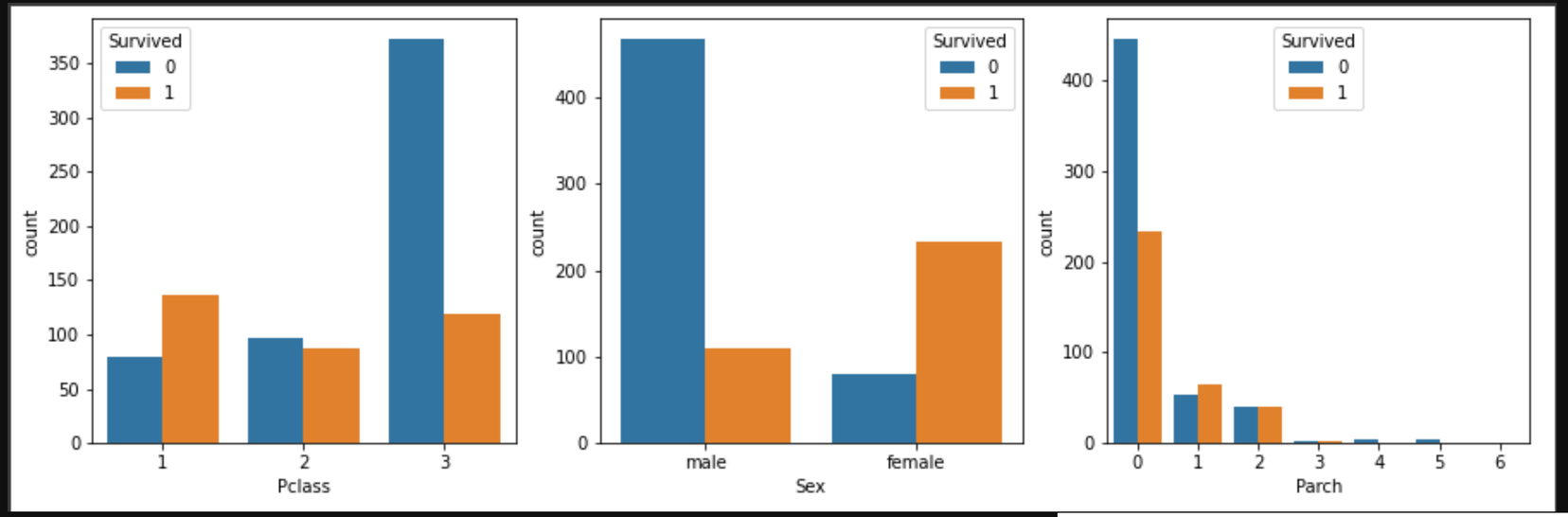
```
plt.figure(figsize=(7, 6))
sns.distplot(df.Fare[df.Survived == 0], bins= 25, color="r", label = "not Survive")
sns.distplot(df.Fare[df.Survived == 1], bins= 25, color="g", label = "Survive")
plt.legend()
```



by. Null Science

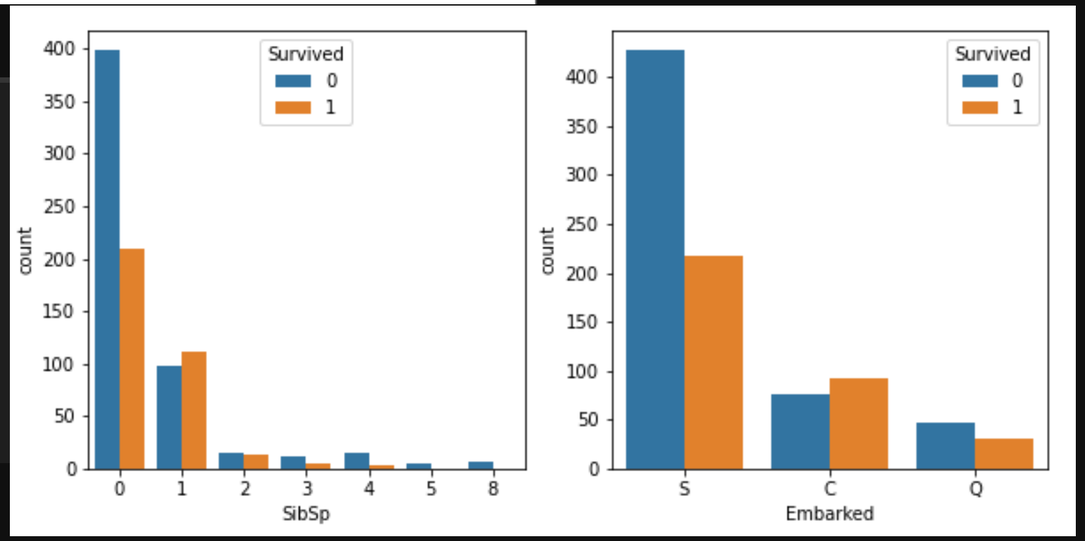
16 Aprill 2022

The insight that comes from the data is that 25 dollars are more likely to not survive, and those over 25 dollars are more likely to survive. you could say the facilities obtained are different



```
#catagoric / ordinal vs target
cat_var = ["Pclass", "Sex", "Parch", "SibSp", "Embarked"]

fig, axes = plt.subplots(2, 3, figsize = (15, 10))
for cat, ax in zip(cat_var, axes.flatten()):
    sns.countplot(cat, data=df, hue="Survived", ax=ax)
```



THE INSIGHT OBTAINED FROM THE DATA ARE:

- Class P: the class with the best quality (1) has higher security than the simple class (3). because it is possible that the facilities received are different from the level
- Sex: men are less safe than women because men may put women's safety first
- Parch: can be seen if not bringing children. increased self-safety
- SibSp: being alone tends to be unsafe and in pairs increases safety
- Embarked: city departures (S) tend to be unsafe and their safety is higher than that of cities (c, and d). it is possible that most of the departures come from class (3)

by. Null Science

16 Aprill 2022

THANK YOU!