TITANIC EDA

ABISHEK THEAGARAJAN

CONTENTS

THE DATASET

DATASET DESCRIBE THE SURVIVAL STATUS OF INDIVIDUAL PASSENGERS ON THE TITANIC

Data Dictionary

VARIABLE	DEFINITION	KEY	TYPE
Passengerid	ID of the obervation		int
Survived	If the passenger survived (**target**)	0 = No, 1 = Yes	int
Pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd	int
Name	Name of the passenger		string
Sex	Sex		string
Age	Age		float
SibSp	# of siblings / spouses aboard the Titanic		int
Parch	# of parents / children aboard the Titanic		int
Ticket	Ticket number		string
Fare	Passenger fare		float
Cabin	Cabin number		string
Embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton	string

THE DATASET IS COMPOSED OF 891 ROWS AND 12 FEATURES

DATA CLEANING

MISSING VALUES

Here we can see the percentage of missing values from the features that have some.

```python
null_list = []
null = data.isnull().sum()

for count, nulls in enumerate(null):
    if nulls > 0:
        null_list.append(null.index[count])
        print('{} = {:.2f}%'.format(null.index[count], nulls/len(data)*100))
```

✓ 0.1s

```
Age = 19.87%
Cabin = 77.10%
Embarked = 0.22%
```

Age: The age missing values are going to be filled it with the median for each Title posteriorly in the Feature Engineering topic.

Cabin: The Cabin has 77.10% of missing values. This is due to the fact that only the 1st class passengers have cabins, but the location of this cabin is important for the dataset, so here I will drop the missing values for the 1st class and define, later, an alternative value for the remaining missing values.

```python
# Finding the index from the data that will be droped.
index_1st_class = data[data['Pclass'] == 1 & data['Cabin'].isnull()].index

# Droping the first class Cabin and the Embarked missing value
data.drop(index_1st_class, inplace=True)
data.dropna(subset=['Embarked'], inplace = True)

# Filling the remaining Cabin missing values
data['Cabin'].fillna('Not_cabin', inplace=True)
```

Embarked: Since they are few examples, those missing values will have their row dropped.

Outliers

```python
data.describe()
```

✓ 0.1s

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	849.000000	849.000000	849.000000	686.000000	849.000000	849.000000	849.000000
mean	446.372203	0.378092	2.373380	29.202872	0.538280	0.398115	29.988226
std	257.903367	0.485197	0.802883	14.346565	1.122773	0.820892	45.976589
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	222.000000	0.000000	2.000000	20.000000	0.000000	0.000000	7.895800
50%	445.000000	0.000000	3.000000	28.000000	0.000000	0.000000	13.000000
75%	671.000000	1.000000	3.000000	37.750000	1.000000	0.000000	29.700000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

The only feature that look like having outliers that maybe need some treatment is the `Fare`, but if it's necessary it will be handled posteriorly.

FEATURE ENGINEERING

PERFORMED FEATURE ENGINEERING ON FEW FEATURES
TITLES, AGE, CABIN DECK, FAMILY SIZE

TITLES

There are many titles used in this, so I will reduce them all to Mrs. Miss, Mr. and Master, I have used a function that searches substrings

```
data['Title'] = data['Name'].str.extract(' ([A-Z-a-z]+)\.', expand=False)
```
0.1s

```
title_mapping = {'Mr': 0, 'Miss': 1, 'Mrs': 2, 'Master': 3,
                 'Dr': 3, 'Rev':3, 'Col':3}
```
0.7s

```
def replace_titles(x):
    title = x['Title']
    if title in ['Don', 'Major', 'Capt', 'Rev', 'Col']:
        return 'Mr'
    elif title in ['Countess', 'Mme']:
        return 'Mrs'
    elif title in ['Mlle', 'Ms']:
        return 'Miss'
    elif title in ['Lady', 'Sir']:
        return 'Master'
    elif title == 'Dr':
        if x['Sex'] == 'Male':
            return 'Mr'
        else:
            return 'Mrs'
    else:
        return title

data['Title'] = data.apply(replace_titles, axis=1)
```
0.1s

I then create a function to replace certain titles with the tittles we want

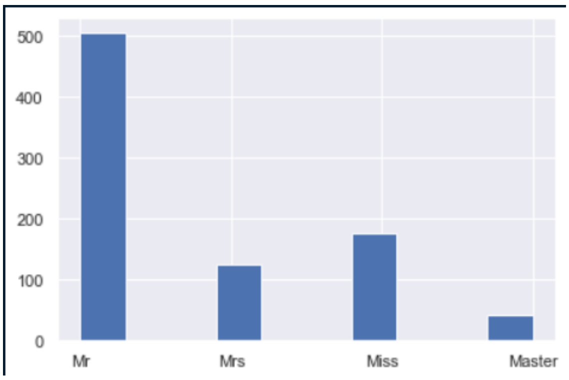
And then drop the names feature

```
# Drop the name feature
data.drop('Name', axis=1, inplace=True)
```
✓ 0.4s

```
data.head()
```
✓ 0.7s

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title
0	1	0	3	male	22.0	1	0	A/5 21171	7.2500	Not_cabin	S	Mr
1	2	1	1	female	38.0	1	0	PC 17599	71.2833	C85	C	Mrs
2	3	1	3	female	26.0	0	0	STON/O2. 3101282	7.9250	Not_cabin	S	Miss
3	4	1	1	female	35.0	1	0	113803	53.1000	C123	S	Mrs
4	5	0	3	male	35.0	0	0	373450	8.0500	Not_cabin	S	Mr

```
plt.hist(data['Title'])
```
✓ 0.1s



AGE

Here I will fill missing values with median age per title and then apply binning to group ages per category.

Feature vector map:

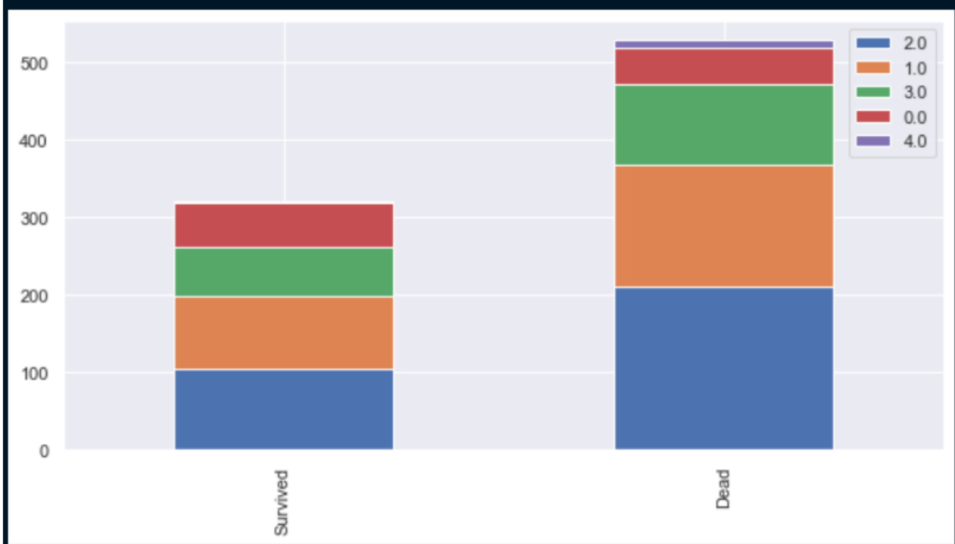
- child: 0
- young: 1
- adult: 2
- mid-age: 3
- senior: 4

```
# Filling the missing values with the median age per title
data['Age'].fillna(data.groupby('Title')['Age'].transform('median'), inplace=True)
```
✓ 0.6s

```
# Binning the age
data.loc[ data['Age'] <= 16, 'Age'] = 0
data.loc[ (data['Age'] > 16) & (data['Age'] <= 26), 'Age'] = 1
data.loc[ (data['Age'] > 26) & (data['Age'] <= 36), 'Age'] = 2
data.loc[ (data['Age'] > 36) & (data['Age'] <= 62), 'Age'] = 3
data.loc[ data['Age'] > 16, 'Age'] = 4
```
✓ 0.6s

```
def bar_chart(feature):
    survived = data[data['Survived']==1][feature].value_counts()
    dead = data[data['Survived']==0][feature].value_counts()
    df = pd.DataFrame([survived, dead])
    df.index = ['Survived', 'Dead']
    df.plot(kind='bar', stacked=True, figsize=(10,5))
```
✓ 0.6s

```
bar_chart('Age')
```
✓ 0.2s



I then create a function to display a stacked bar chart w.r.t. to the feature vector map.

CABIN DECK

Only first class passengers have cabins, so will simplify cabin numbers to just the first letter of the cabin number and mark unknown cabins as N

```
data['Cabin'] = data['Cabin'].str[:1]
```
✓ 0.6s

FAMILY SIZE

creating a new feature called family size

```
data['FamilySize'] = data['SibSp'] + data['Parch'] + 1
```
✓ 0.9s

IRRELEVANT VARIABLES FOR THE MODEL

We will drop features like:
ticket-not relevant to the model
SibSp & Parch - already considered in FamilySize

```
features_drop = ['Ticket', 'SibSp', 'Parch']
data.drop(features_drop, axis=1, inplace=True)
```
✓ 0.8s

```
data.head()
```
✓ 0.2s

	PassengerId	Survived	Pclass	Sex	Age	Fare	Cabin	Embarked	Title	FamilySize
0	1	0	3	male	1.0	7.2500	N	S	Mr	2
1	2	1	1	female	3.0	71.2833	C	C	Mrs	2
2	3	1	3	female	1.0	7.9250	N	S	Miss	1
3	4	1	1	female	2.0	53.1000	C	S	Mrs	2
4	5	0	3	male	2.0	8.0500	N	S	Mr	1

EXPLORATORY DATA ANALYSIS

```
data.info()
✓ 0.7s
<class 'pandas.core.frame.DataFrame'>
Int64Index: 849 entries, 0 to 890
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  849 non-null    int64
 1   Survived     849 non-null    int64
 2   Pclass       849 non-null    int64
 3   Sex          849 non-null    object
 4   Age          849 non-null    float64
 5   Fare         849 non-null    float64
 6   Cabin        849 non-null    object
 7   Embarked     849 non-null    object
 8   Title        849 non-null    object
 9   FamilySize   849 non-null    int64
dtypes: float64(2), int64(4), object(4)
```

- 1 for ID
- 1 target (Survived)
- 4 numerical
- 4 categorical

There are 10 attributes in total

And will perform EDA for each type of attribute.

EDA IN NUMERIC VARIABLES

```
data.hist(bins=20, figsize=(20,15));
✓ 0.9s
```

From this plot we can see that majority of passengers are from the 3rd class

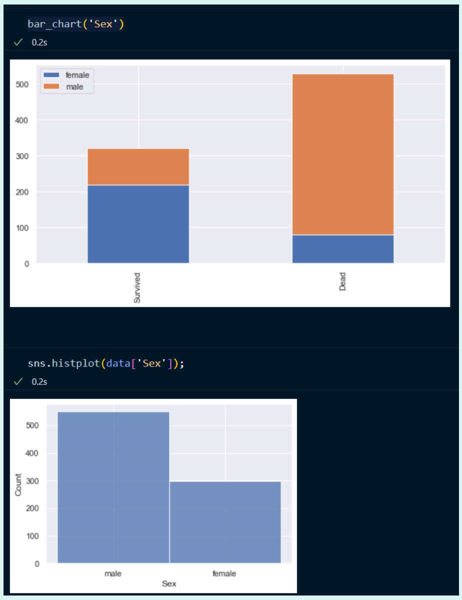
```python
data.describe(include=[object])
```
0.6s

	Sex	Cabin	Embarked	Title
count	849	849	849	849
unique	2	9	3	4
top	male	N	S	Mr
freq	550	647	623	505

Attributes under categoric variables are Sex, Cabin Embarked, Title

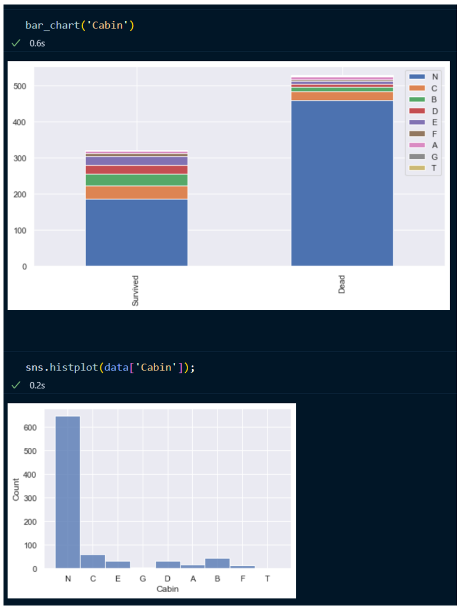
SEX

The dataset shows that males are more likely of dying



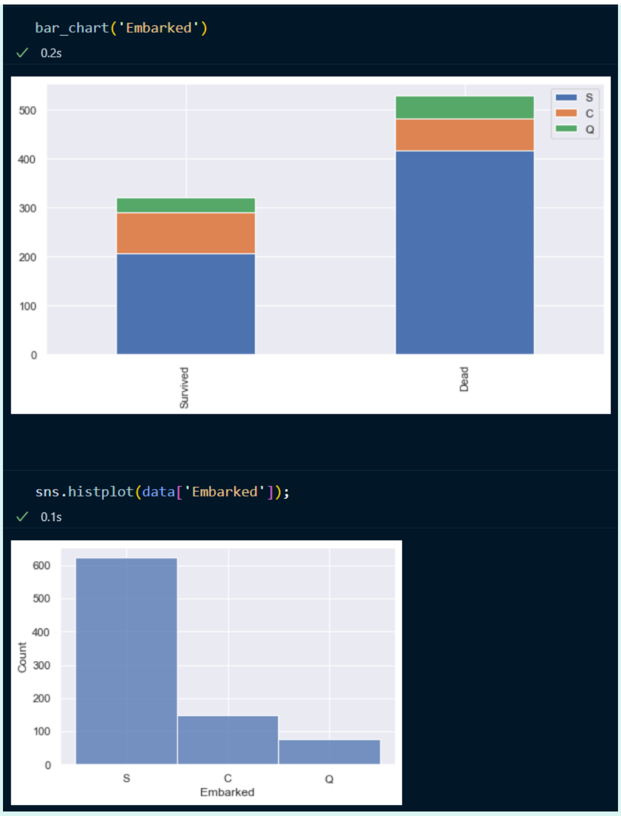
CABIN

Most of the passengers did not have a cabin.



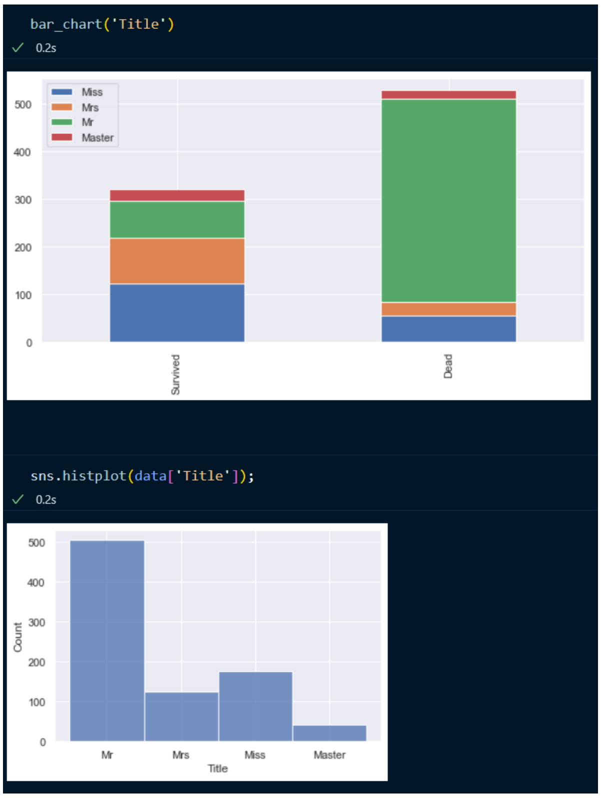
EMBARKED

From the plots we can see that most of the passengers had embarked from Southampton



TITLE

Just like from the Sex topic we can see that males had a larger chance of dying than females



HYPOTHESIS TESTING

HYPOTHESIS 1:

Being male increases the chances of the passenger dying.

Null Hypothesis: If the passenger is male it is more likely that he will die.

Alternate Hypothesis: There is no difference in the survival rate according to the sex of the passenger.

HYPOTHESIS 2:

Since some cabins where placed in the front of the ship, that has sinked first, the location of the cabin impact on the survival rate.

Null Hypothesis: If the cabin is placed in the front of the ship it is more likely that he will die.

Alternate Hypothesis: There is no difference in the survival rate according to the cabin of the passenger.

HYPOTHESIS 3:

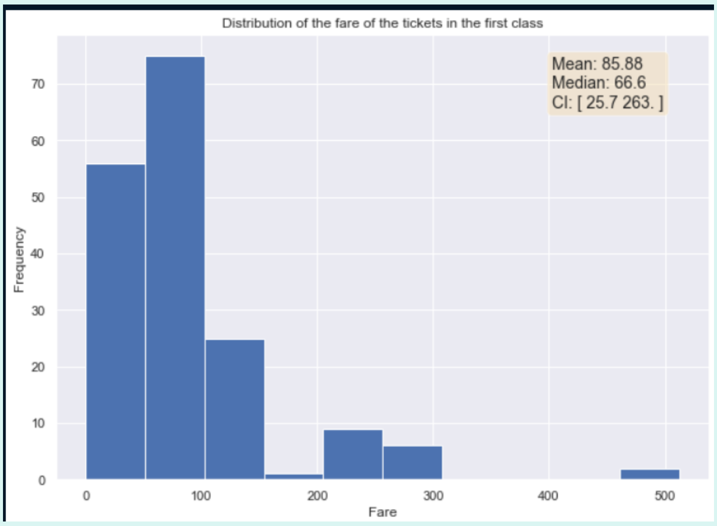
We can then make the hypothesis that rich people on the Titanic had a higher survival rate than the others.
(from the movie)

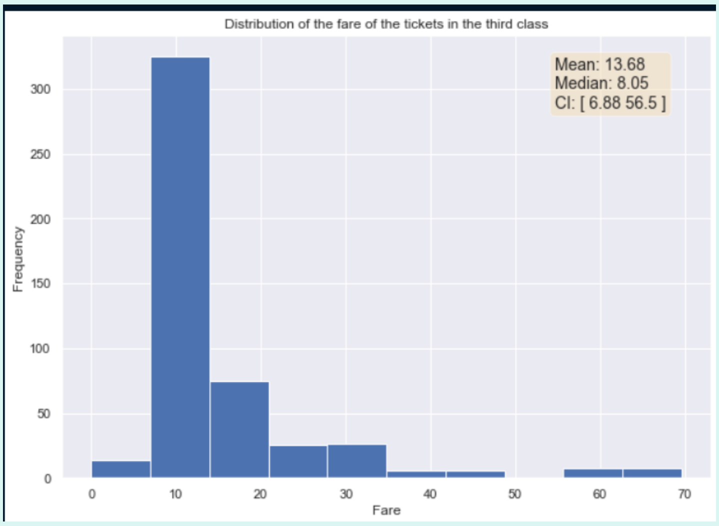
Null Hypothesis: The socio-economic class of the people didn't have an effect on the survival rate.

Alternate Hypothesis: The socio-economic class of the people affected their survival rate.

For the third hypothesis we only need 3 features:
Survived
Pclass
Fare

```python
# Distribution for rich:
first_fares = data["Fare"][data["Pclass"]==1]
first_mean = round(np.mean(first_fares), 2)
first_median = round(np.median(first_fares), 2)
first_conf = np.round(np.percentile(first_fares, [2.5, 97.5]), 2)

fig, ax = plt.subplots(figsize = (10, 7))

ax.hist(first_fares)
props = dict(boxstyle='round', facecolor='wheat', alpha=0.5)
ax.text(0.76, 0.95, f"Mean: {first_mean} \nMedian: {first_median} \nCI: {first_conf}", transform=ax.transAxes, fontsize=14,
        verticalalignment='top', bbox=props)
plt.xlabel("Fare")
plt.ylabel("Frequency")
plt.title("Distribution of the fare of the tickets in the first class")
plt.show()

# Distribution for Poor
third_fares = data["Fare"][data["Pclass"]==3]
third_mean = round(np.mean(third_fares), 2)
third_median = round(np.median(third_fares), 2)
third_conf = np.round(np.percentile(third_fares, [2.5, 97.5]), 2)

fig, ax = plt.subplots(figsize = (10, 7))

ax.hist(third_fares)
props = dict(boxstyle='round', facecolor='wheat', alpha=0.5)
ax.text(0.76, 0.95, f"Mean: {third_mean} \nMedian: {third_median} \nCI: {third_conf}", transform=ax.transAxes, fontsize=14,
        verticalalignment='top', bbox=props)
plt.xlabel("Fare")
plt.ylabel("Frequency")
plt.title("Distribution of the fare of the tickets in the third class")
plt.show()
```
0.3s

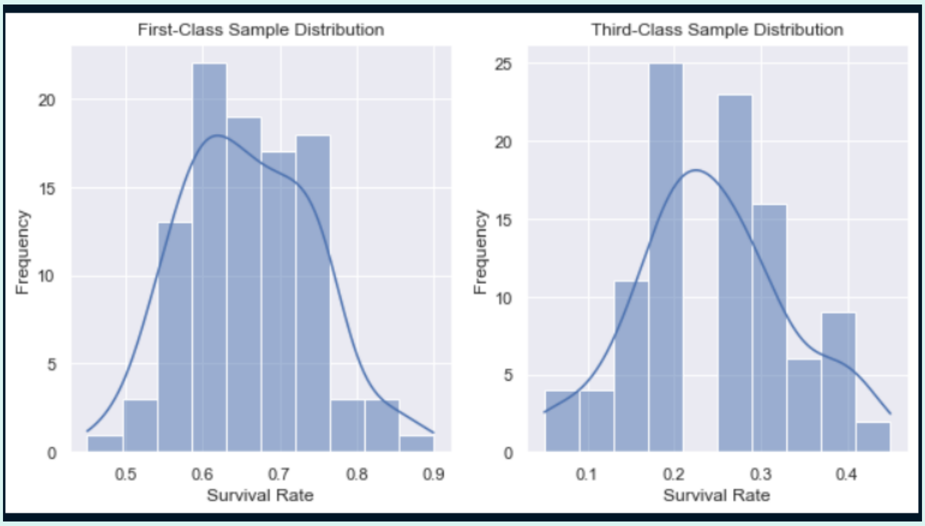




Taking a sample of 100 means from each population (first-class and third-class), using the central limit theorem, to ensure that our data is normally distributed.

```
first_class_sample = np.array([np.mean(data[data["Pclass"]==1].sample(20)["Survived"].values) for i in range(100)])
third_class_sample = np.array([np.mean(data[data["Pclass"]==3].sample(20)["Survived"].values) for i in range(100)])
```
✓ 0.2s

```
plt.subplots(1, 2, figsize = (10, 5))
plt.subplot(1,2, 1)
sns.histplot(first_class_sample, kde=True)
plt.title("First-Class Sample Distribution")
plt.xlabel("Survival Rate")
plt.ylabel("Frequency")

plt.subplot(1, 2, 2)
sns.histplot(third_class_sample, kde=True)
plt.title("Third-Class Sample Distribution")
plt.xlabel("Survival Rate")
plt.ylabel("Frequency")
plt.show()
```
✓ 0.3s




```
effect = np.mean(first_class_sample) - np.mean(third_class_sample)
sigma_first = np.std(first_class_sample)
sigma_third = np.std(third_class_sample)
sigma_difference = np.sqrt((sigma_first**2)/len(first_class_sample)  +  (sigma_third**2)/len(third_class_sample))
z_score = effect / sigma_difference
```
✓ 0.6s

```
z_score
```
✓ 0.9s

```
33.244710957508275
```

```
p_value = norm.sf(abs(z_score))*2
p_value
```
✓ 0.7s

```
2.434154273153322e-242
```

Since the p_value is so small we can reject the null hypothesis. The provided sample proves a significant correlation between the socio-economic class and the survival rate. We cannot establish causation between these two features but we can make a generalized induction that richer people had a better chance of survival at the ship.

CONCLUSION

In this report, I had conducted a simple exploratory analysis of the titanic dataset. After some feature engineering of the dataset, and the removal of duplicate and outlier values numerical and categorical variables were explored. From the results of the hypothesis testing conducted we cannot establish causation between these two features but we can make a generalized induction that richer people had a better chance of survival at the ship.

NEXT STEPS

The next step would be to thoroughly conduct analysis by investigating and after EDA is the creation of training and test splits for establishing a Machine Learning model.