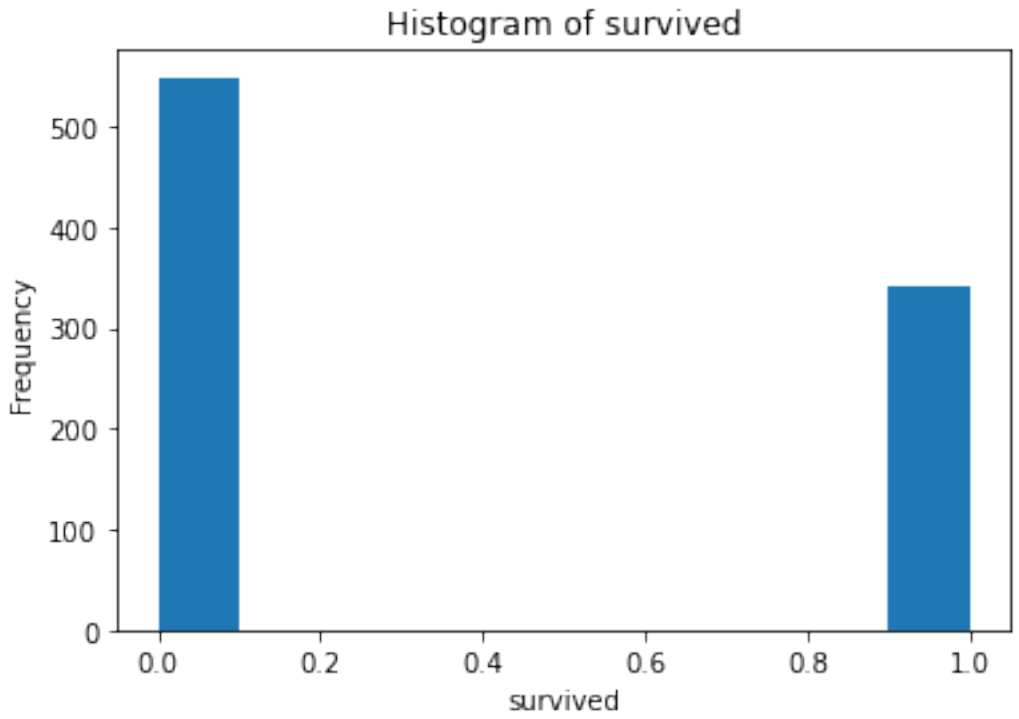
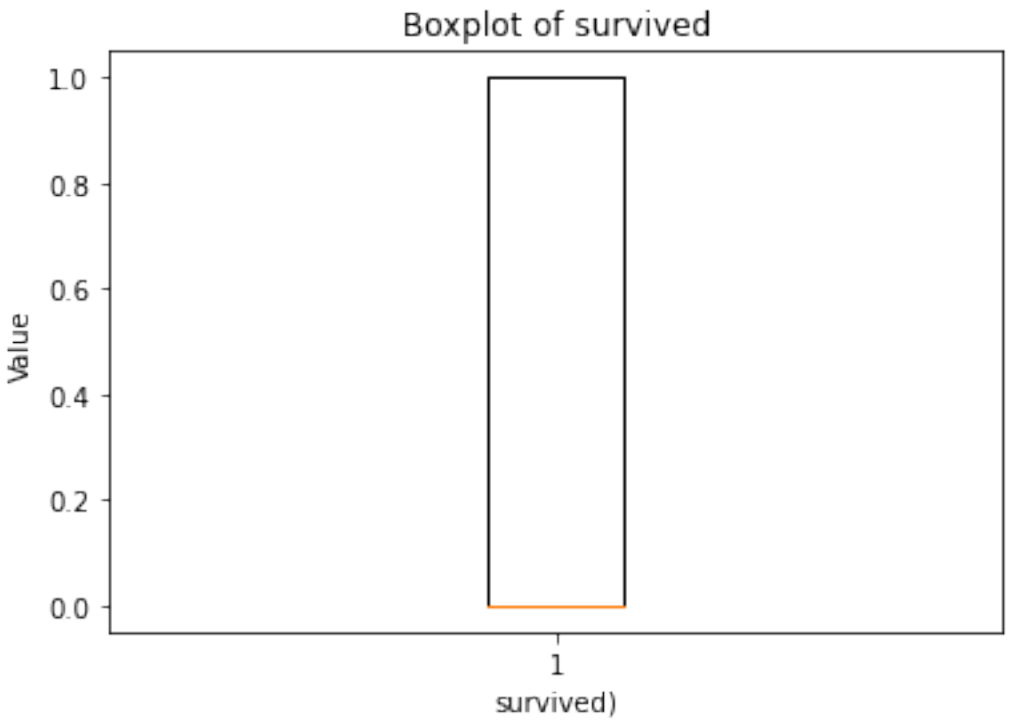
```
#Load the Dataset
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

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

```
     survived  pclass     sex   age  sibsp  parch     fare embarked
class  \
0           0       3    male  22.0      1      0   7.2500        S
Third
1           1       1  female  38.0      1      0  71.2833        C
First
2           1       3  female  26.0      0      0   7.9250        S
Third
3           1       1  female  35.0      1      0  53.1000        S
First
4           0       3    male  35.0      0      0   8.0500        S
Third
..        ...     ...     ...   ...    ...    ...      ...       ...
...
886         0       2    male  27.0      0      0  13.0000        S
Second
887         1       1  female  19.0      0      0  30.0000        S
First
888         0       3  female   NaN      1      2  23.4500        S
Third
889         1       1    male  26.0      0      0  30.0000        C
First
890         0       3    male  32.0      0      0   7.7500        Q
Third

       who  adult_male deck  embark_town alive  alone
0      man        True  NaN  Southampton    no  False
1    woman       False    C    Cherbourg   yes  False
2    woman       False  NaN  Southampton   yes   True
3    woman       False    C  Southampton   yes  False
4      man        True  NaN  Southampton    no   True
..     ...         ...  ...          ...   ...    ...
886    man        True  NaN  Southampton    no   True
887  woman       False    B  Southampton   yes   True
888  woman       False  NaN  Southampton    no  False
889    man        True    C    Cherbourg   yes   True
890    man        True  NaN   Queenstown    no   True

[891 rows x 15 columns]
```

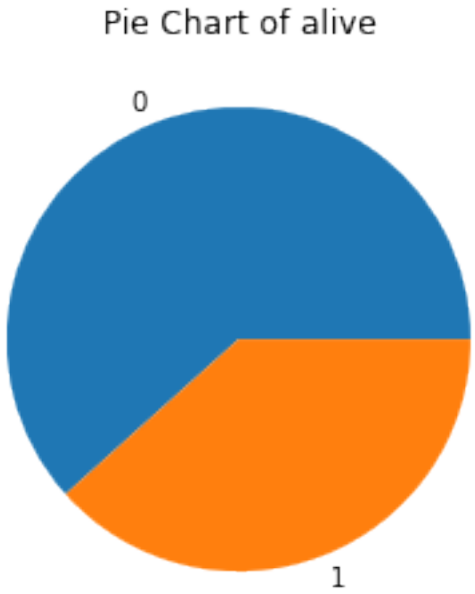
```
#Perform Below Visualizations.
#● Univariate Analysis(Histogram, box plot, Pie chart)
#● Bi - Variate Analysis(Scatter plot, Line chart, Bar chart)
#● Multi - Variate Analysis(Tree map, 3D Scatter plot, Heat map)

# Univariate analysis
# Histogram
plt.hist(df['survived'], bins=10)
plt.title('Histogram of survived')
plt.xlabel('survived')
plt.ylabel('Frequency')
plt.show()
```



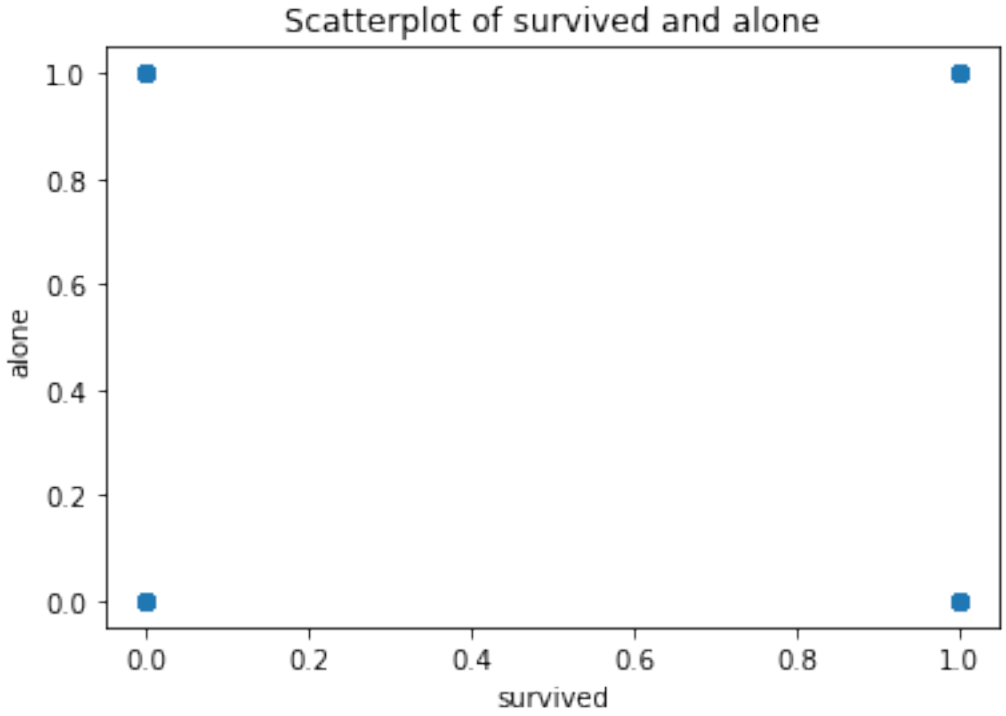

```
# Boxplot
plt.boxplot(df['survived'])
plt.title('Boxplot of survived')
plt.xlabel('survived)')
plt.ylabel('Value')
plt.show()
```



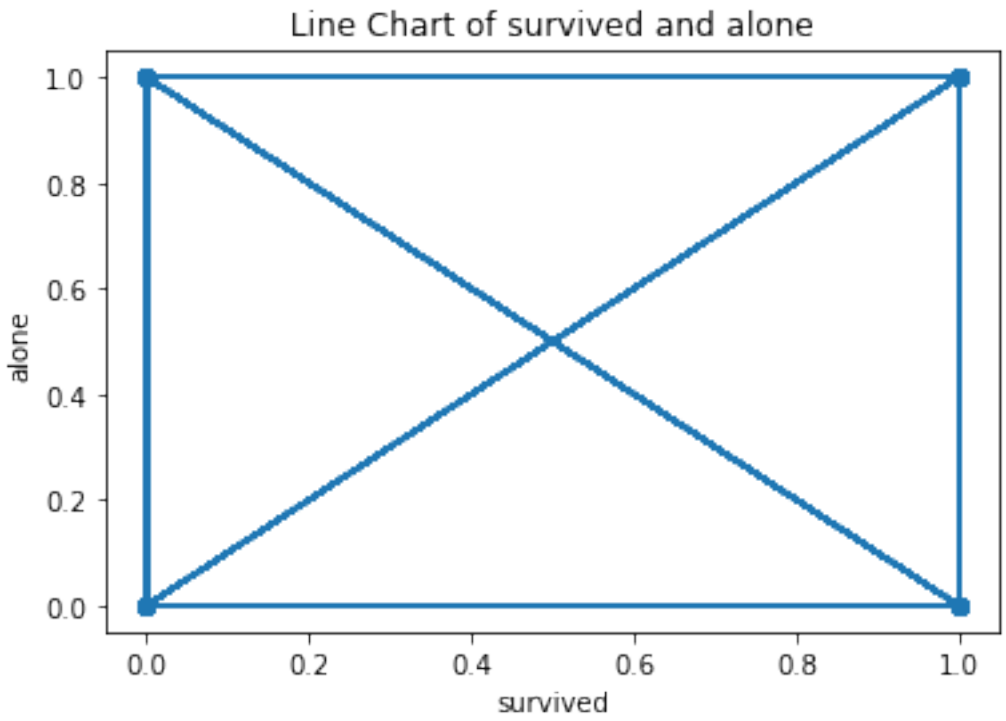

```
#Pie Chart
plt.pie(df['survived'].value_counts(), labels=df['survived'].unique())
plt.title('Pie Chart of alive')
plt.show()
```



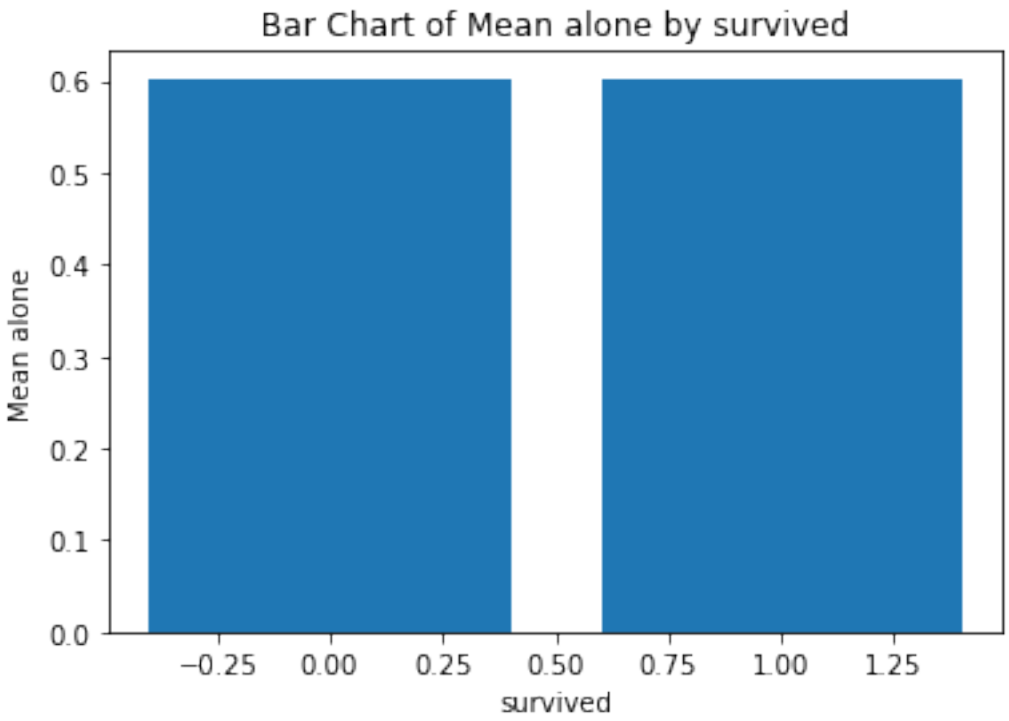
```
# Bivariate analysis
# Scatterplot
plt.scatter(df['survived'], df['alone'])
plt.title('Scatterplot of survived and alone')
plt.xlabel('survived')
plt.ylabel('alone')
plt.show()
```



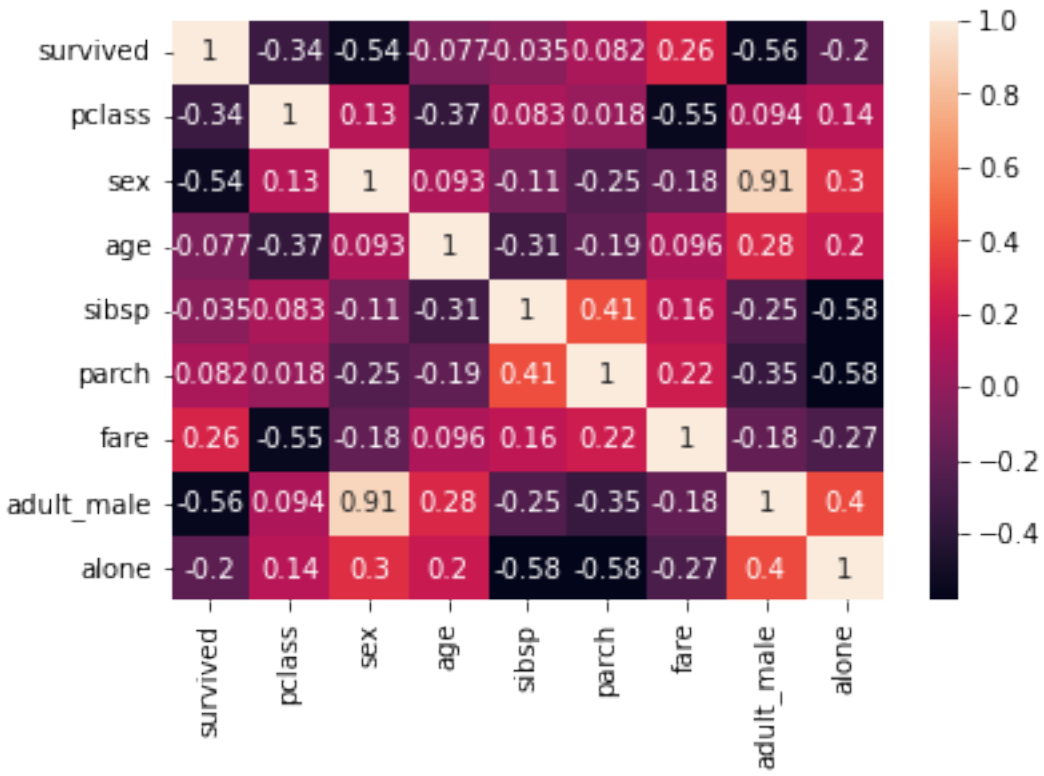

```
# Line chart
plt.plot(df['survived'], df['alone'], 'o-')
plt.title('Line Chart of survived and alone')
plt.xlabel('survived')
plt.ylabel('alone')
plt.show()
```



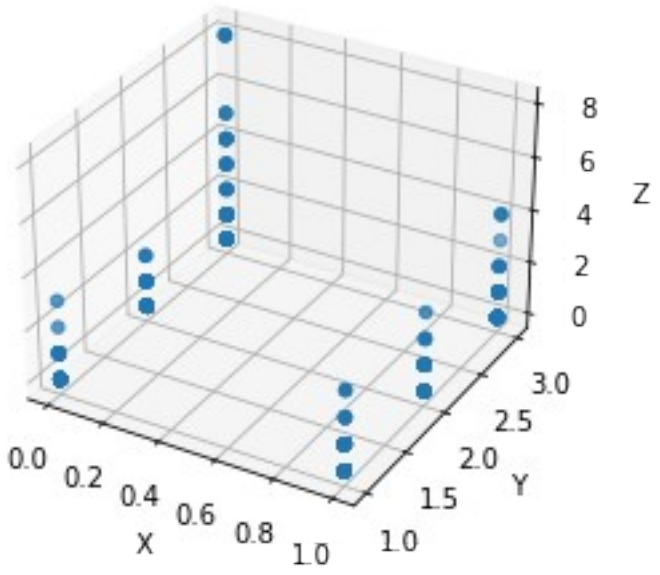

```
# Bar chart
plt.bar(df['survived'].unique(), df['alone'].mean(), align='center')
plt.title('Bar Chart of Mean alone by survived')
plt.xlabel('survived')
plt.ylabel('Mean alone')
plt.show()
```



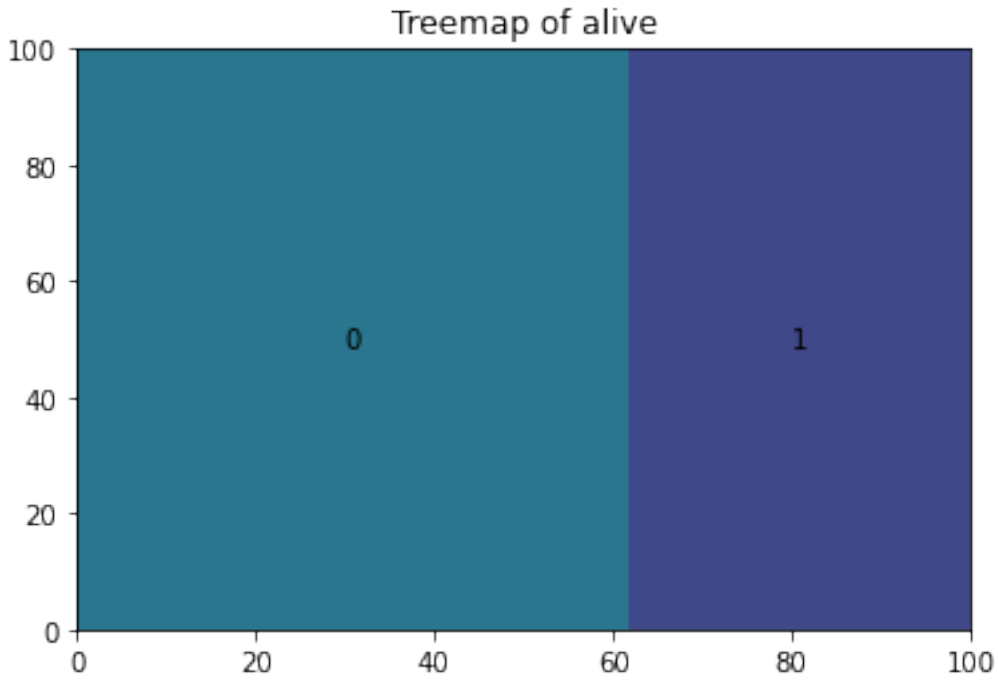

```
# Multivariate analysis
# Heatmap
import seaborn as sns
df['sex'] = df['sex'].astype('category').cat.codes
sns.heatmap(df.corr(), annot=True)
plt.show()
```




```
# Multivariate analysis
# 3D scatterplot
from mpl_toolkits.mplot3d import Axes3D
x = df['survived']
y = df['pclass']
z = df['sibsp']
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(x, y, z)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.show()
```




```
# Treemap
import squarify
plt.figure()
squarify.plot(df['survived'].value_counts(),
label=df['survived'].unique())
plt.title('Treemap of alive')
plt.show()
```



```
# 4.Perform descriptive statistics on the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    int8
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    object
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    object
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), float64(2), int64(4), int8(1), object(6)
memory usage: 86.3+ KB
```

```
df.describe()
```

```
         survived      pclass         sex         age       sibsp
parch  \
count  891.000000  891.000000  891.000000  714.000000  891.000000
891.000000
mean     0.383838    2.308642    0.647587   29.699118    0.523008
0.381594
std      0.486592    0.836071    0.477990   14.526497    1.102743
0.806057
min      0.000000    1.000000    0.000000    0.420000    0.000000
0.000000
25%      0.000000    2.000000    0.000000   20.125000    0.000000
0.000000
50%      0.000000    3.000000    1.000000   28.000000    0.000000
0.000000
75%      1.000000    3.000000    1.000000   38.000000    1.000000
0.000000
max      1.000000    3.000000    1.000000   80.000000    8.000000
6.000000

              fare
count  891.000000
mean    32.204208
```

```
std     49.693429
min      0.000000
25%      7.910400
50%     14.454200
75%     31.000000
max    512.329200
```

```
# 4.Handle the Missing values.
df.isnull().sum()
```

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

```
df = df.dropna()
```

```
#6.Find the outliers and replace the outliers
target_column = 'survived'
Q1 = df[target_column].quantile(0.25)
Q3 = df[target_column].quantile(0.75)
IQR = Q3 - Q1
```

```
IQR
```

```
1.0
```

```
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
```

```
lower_bound
```

```
-1.5
```

```
upper_bound
```

```
2.5
```

```
outliers = df[(df[target_column] < lower_bound) | (df[target_column] >
upper_bound)]
```

```
median_value = df[target_column].median()
df.loc[(df[target_column] < lower_bound) | (df[target_column] >
upper_bound), target_column] = median_value

print(df)
```

```
     survived  pclass  sex   age  sibsp  parch     fare embarked
class  \
1           1       1    0  38.0      1      0  71.2833        C
First
3           1       1    0  35.0      1      0  53.1000        S
First
6           0       1    1  54.0      0      0  51.8625        S
First
10          1       3    0   4.0      1      1  16.7000        S
Third
11          1       1    0  58.0      0      0  26.5500        S
First
..        ...     ...  ...   ...    ...    ...      ...      ...    ..
.
871         1       1    0  47.0      1      1  52.5542        S
First
872         0       1    1  33.0      0      0   5.0000        S
First
879         1       1    0  56.0      0      1  83.1583        C
First
887         1       1    0  19.0      0      0  30.0000        S
First
889         1       1    1  26.0      0      0  30.0000        C
First

       who  adult_male deck  embark_town alive  alone
1    woman       False    C    Cherbourg   yes  False
3    woman       False    C  Southampton   yes  False
6      man        True    E  Southampton    no   True
10   child       False    G  Southampton   yes  False
11   woman       False    C  Southampton   yes   True
..     ...         ...  ...          ...   ...    ...
871  woman       False    D  Southampton   yes  False
872    man        True    B  Southampton    no   True
879  woman       False    C    Cherbourg   yes  False
887  woman       False    B  Southampton   yes   True
889    man        True    C    Cherbourg   yes   True

[182 rows x 15 columns]
```

```
#7.Check for Categorical columns and perform encoding
from sklearn.preprocessing import LabelEncoder
df.dtypes
```

```
survived         int64
pclass           int64
sex               int8
age            float64
sibsp            int64
parch            int64
fare           float64
embarked        object
class           object
who             object
adult_male        bool
deck            object
embark_town     object
alive           object
alone             bool
dtype: object
```

```
categorical_columns = df.select_dtypes(include=['object']).columns
df_encoded = pd.get_dummies(df, columns=categorical_columns)

print(df_encoded)
```

```
     survived  pclass  sex   age  sibsp  parch     fare  adult_male
alone  \
1           1       1    0  38.0      1      0  71.2833       False
False
3           1       1    0  35.0      1      0  53.1000       False
False
6           0       1    1  54.0      0      0  51.8625        True
True
10          1       3    0   4.0      1      1  16.7000       False
False
11          1       1    0  58.0      0      0  26.5500       False
True
..        ...     ...  ...   ...    ...    ...      ...         ...
...
871         1       1    0  47.0      1      1  52.5542       False
False
872         0       1    1  33.0      0      0   5.0000        True
True
879         1       1    0  56.0      0      1  83.1583       False
False
887         1       1    0  19.0      0      0  30.0000       False
True
889         1       1    1  26.0      0      0  30.0000        True
True

     embarked_C  ...  deck_C  deck_D  deck_E  deck_F  deck_G  \
1             1  ...       1       0       0       0       0
3             0  ...       1       0       0       0       0
```

```
6              0  ...       0       0       1       0       0
10             0  ...       0       0       0       0       1
11             0  ...       1       0       0       0       0
..           ...  ...     ...     ...     ...     ...     ...
871            0  ...       0       1       0       0       0
872            0  ...       0       0       0       0       0
879            1  ...       1       0       0       0       0
887            0  ...       0       0       0       0       0
889            1  ...       1       0       0       0       0

     embark_town_Cherbourg  embark_town_Queenstown
embark_town_Southampton  \
1                        1                       0
0
3                        0                       0
1
6                        0                       0
1
10                       0                       0
1
11                       0                       0
1
..                     ...                     ...
...
871                      0                       0
1
872                      0                       0
1
879                      1                       0
0
887                      0                       0
1
889                      1                       0
0

     alive_no  alive_yes
1           0          1
3           0          1
6           1          0
10          0          1
11          0          1
..        ...        ...
871         0          1
872         1          0
879         0          1
887         0          1
889         0          1

[182 rows x 30 columns]
```

```
categorical_columns
```

```
Index(['embarked', 'class', 'who', 'deck', 'embark_town', 'alive'],
dtype='object')
```

```
# 8.Split the data into dependent and independent variables.
dependent_variable = 'alive'
independent_variables = df.drop(dependent_variable, axis=1)
dependent_variable = df[dependent_variable]

print(independent_variables)
```

```
     survived  pclass  sex   age  sibsp  parch     fare embarked
class  \
1           1       1    0  38.0      1      0  71.2833        C
First
3           1       1    0  35.0      1      0  53.1000        S
First
6           0       1    1  54.0      0      0  51.8625        S
First
10          1       3    0   4.0      1      1  16.7000        S
Third
11          1       1    0  58.0      0      0  26.5500        S
First
..        ...     ...  ...   ...    ...    ...      ...      ...    ..
.
871         1       1    0  47.0      1      1  52.5542        S
First
872         0       1    1  33.0      0      0   5.0000        S
First
879         1       1    0  56.0      0      1  83.1583        C
First
887         1       1    0  19.0      0      0  30.0000        S
First
889         1       1    1  26.0      0      0  30.0000        C
First

       who  adult_male deck  embark_town  alone
1    woman       False    C    Cherbourg  False
3    woman       False    C  Southampton  False
6      man        True    E  Southampton   True
10   child       False    G  Southampton  False
11   woman       False    C  Southampton   True
..     ...         ...  ...          ...    ...
871  woman       False    D  Southampton  False
872    man        True    B  Southampton   True
879  woman       False    C    Cherbourg  False
887  woman       False    B  Southampton   True
889    man        True    C    Cherbourg   True
```

```
[182 rows x 14 columns]
```

```
print(dependent_variable)
```

```
1      yes
3      yes
6       no
10     yes
11     yes
      ...
871    yes
872     no
879    yes
887    yes
889    yes
Name: alive, Length: 182, dtype: object
```

```
# 9.Scale the independent variables
from sklearn.preprocessing import StandardScaler
columns_to_scale = ['survived','pclass','sibsp']
scaler = StandardScaler()
df[columns_to_scale] = scaler.fit_transform(df[columns_to_scale])
```

```
C:\Users\sabih\AppData\Local\Temp\ipykernel_22192\2276414521.py:5:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df[columns_to_scale] = scaler.fit_transform(df[columns_to_scale])
```

```
print(df)
```

```
     survived    pclass  sex   age     sibsp  parch     fare embarked
class  \
1    0.692586 -0.373420    0  38.0  0.828576      0  71.2833        C
First
3    0.692586 -0.373420    0  35.0  0.828576      0  53.1000        S
First
6   -1.443865 -0.373420    1  54.0 -0.726072      0  51.8625        S
First
10   0.692586  3.510145    0   4.0  0.828576      1  16.7000        S
Third
11   0.692586 -0.373420    0  58.0 -0.726072      0  26.5500        S
First
..        ...       ...  ...   ...       ...    ...      ...      ...
...
```

```
871  0.692586 -0.373420    0  47.0  0.828576      1  52.5542        S
First
872 -1.443865 -0.373420    1  33.0 -0.726072      0   5.0000        S
First
879  0.692586 -0.373420    0  56.0 -0.726072      1  83.1583        C
First
887  0.692586 -0.373420    0  19.0 -0.726072      0  30.0000        S
First
889  0.692586 -0.373420    1  26.0 -0.726072      0  30.0000        C
First

       who  adult_male deck  embark_town alive  alone
1    woman       False    C    Cherbourg   yes  False
3    woman       False    C  Southampton   yes  False
6      man        True    E  Southampton    no   True
10   child       False    G  Southampton   yes  False
11   woman       False    C  Southampton   yes   True
..     ...         ...  ...          ...   ...    ...
871  woman       False    D  Southampton   yes  False
872    man        True    B  Southampton    no   True
879  woman       False    C    Cherbourg   yes  False
887  woman       False    B  Southampton   yes   True
889    man        True    C    Cherbourg   yes   True

[182 rows x 15 columns]
```

```
#10.Split the data into training and testing
from sklearn.model_selection import train_test_split
X = df.drop('survived', axis=1)
y = df['survived']
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.25, random_state=42)

X_train
```

```
      pclass  sex   age     sibsp  parch      fare embarked  class
who  \
581 -0.37342    0  39.0  0.828576      1  110.8833        C  First
woman
224 -0.37342    1  38.0  0.828576      0   90.0000        S  First
man
583 -0.37342    1  36.0 -0.726072      0   40.1250        C  First
man
724 -0.37342    1  27.0  0.828576      0   53.1000        S  First
man
370 -0.37342    1  25.0  0.828576      0   55.4417        C  First
man
..       ...  ...   ...       ...    ...       ...      ...    ...
...
523 -0.37342    0  44.0 -0.726072      1   57.9792        C  First
```

```
woman
92  -0.37342    1  46.0  0.828576      0   61.1750        S  First
man
462 -0.37342    1  47.0 -0.726072      0   38.5000        S  First
man
879 -0.37342    0  56.0 -0.726072      1   83.1583        C  First
woman
512 -0.37342    1  36.0 -0.726072      0   26.2875        S  First
man

     adult_male deck  embark_town alive  alone
581       False    C    Cherbourg   yes  False
224        True    C  Southampton   yes  False
583        True    A    Cherbourg    no   True
724        True    E  Southampton   yes  False
370        True    E    Cherbourg   yes  False
..          ...  ...          ...   ...    ...
523       False    B    Cherbourg   yes  False
92         True    E  Southampton    no  False
462        True    E  Southampton    no   True
879       False    C    Cherbourg   yes  False
512        True    E  Southampton   yes   True

[136 rows x 14 columns]

X_test

       pclass  sex    age     sibsp  parch      fare embarked   class
who  \
118 -0.373420    1  24.00 -0.726072      1  247.5208        C   First
man
251  3.510145    0  29.00  0.828576      1   10.4625        S   Third
woman
742 -0.373420    0  21.00  2.383223      2  262.3750        C   First
woman
496 -0.373420    0  54.00  0.828576      0   78.2667        C   First
woman
712 -0.373420    1  48.00  0.828576      0   52.0000        S   First
man
96  -0.373420    1  71.00 -0.726072      0   34.6542        C   First
man
139 -0.373420    1  24.00 -0.726072      0   79.2000        C   First
man
337 -0.373420    0  41.00 -0.726072      0  134.5000        C   First
woman
572 -0.373420    1  36.00 -0.726072      0   26.3875        S   First
man
487 -0.373420    1  58.00 -0.726072      0   29.7000        C   First
man
486 -0.373420    0  35.00  0.828576      0   90.0000        S   First
```

```
woman
765 -0.373420    0  51.00  0.828576      0   77.9583        S   First
woman
340  1.568363    1   2.00  0.828576      1   26.0000        S  Second
child
550 -0.373420    1  17.00 -0.726072      2  110.8833        C   First
man
262 -0.373420    1  52.00  0.828576      1   79.6500        S   First
man
97  -0.373420    1  23.00 -0.726072      1   63.3583        C   First
man
291 -0.373420    0  19.00  0.828576      0   91.0792        C   First
woman
627 -0.373420    0  21.00 -0.726072      0   77.9583        S   First
woman
492 -0.373420    1  55.00 -0.726072      0   30.5000        S   First
man
307 -0.373420    0  17.00  0.828576      0  108.9000        C   First
woman
857 -0.373420    1  51.00 -0.726072      0   26.5500        S   First
man
599 -0.373420    1  49.00  0.828576      0   56.9292        C   First
man
707 -0.373420    1  42.00 -0.726072      0   26.2875        S   First
man
183  1.568363    1   1.00  2.383223      1   39.0000        S  Second
child
54  -0.373420    1  65.00 -0.726072      1   61.9792        C   First
man
609 -0.373420    0  40.00 -0.726072      0  153.4625        S   First
woman
318 -0.373420    0  31.00 -0.726072      2  164.8667        S   First
woman
110 -0.373420    1  47.00 -0.726072      0   52.0000        S   First
man
789 -0.373420    1  46.00 -0.726072      0   79.2000        C   First
man
701 -0.373420    1  35.00 -0.726072      0   26.2875        S   First
man
835 -0.373420    0  39.00  0.828576      1   83.1583        C   First
woman
305 -0.373420    1   0.92  0.828576      2  151.5500        S   First
child
456 -0.373420    1  65.00 -0.726072      0   26.5500        S   First
man
430 -0.373420    1  28.00 -0.726072      0   26.5500        S   First
man
332 -0.373420    1  38.00 -0.726072      1  153.4625        S   First
man
```

```
177 -0.373420    0  50.00 -0.726072      0   28.7125        C   First
woman
585 -0.373420    0  18.00 -0.726072      2   79.6500        S   First
woman
331 -0.373420    1  45.50 -0.726072      0   28.5000        S   First
man
336 -0.373420    1  29.00  0.828576      0   66.6000        S   First
man
193  1.568363    1   3.00  0.828576      1   26.0000        S  Second
child
75   3.510145    1  25.00 -0.726072      0    7.6500        S   Third
man
248 -0.373420    1  37.00  0.828576      1   52.5542        S   First
man
625 -0.373420    1  61.00 -0.726072      0   32.3208        S   First
man
473  1.568363    0  23.00 -0.726072      0   13.7917        C  Second
woman
571 -0.373420    0  53.00  2.383223      0   51.4792        S   First
woman
438 -0.373420    1  64.00  0.828576      4  263.0000        S   First
man
```

```
     adult_male deck  embark_town alive  alone
118        True    B    Cherbourg    no  False
251       False    G  Southampton    no  False
742       False    B    Cherbourg   yes  False
496       False    D    Cherbourg   yes  False
712        True    C  Southampton   yes  False
96         True    A    Cherbourg    no   True
139        True    B    Cherbourg    no   True
337       False    E    Cherbourg   yes   True
572        True    E  Southampton   yes   True
487        True    B    Cherbourg    no   True
486       False    C  Southampton   yes  False
765       False    D  Southampton   yes  False
340       False    F  Southampton   yes  False
550        True    C    Cherbourg   yes  False
262        True    E  Southampton    no  False
97         True    D    Cherbourg   yes  False
291       False    B    Cherbourg   yes  False
627       False    D  Southampton   yes   True
492        True    C  Southampton    no   True
307       False    C    Cherbourg   yes  False
857        True    E  Southampton   yes   True
599        True    A    Cherbourg   yes  False
707        True    E  Southampton   yes   True
183       False    F  Southampton   yes  False
54         True    B    Cherbourg    no  False
```

```
609       False    C  Southampton   yes   True
318       False    C  Southampton   yes  False
110        True    C  Southampton    no   True
789        True    B    Cherbourg    no   True
701        True    E  Southampton   yes   True
835       False    E    Cherbourg   yes  False
305       False    C  Southampton   yes  False
456        True    E  Southampton    no   True
430        True    C  Southampton   yes   True
332        True    C  Southampton    no  False
177       False    C    Cherbourg    no   True
585       False    E  Southampton   yes  False
331        True    C  Southampton    no   True
336        True    C  Southampton    no  False
193       False    F  Southampton   yes  False
75         True    F  Southampton    no   True
248        True    D  Southampton   yes  False
625        True    D  Southampton    no   True
473       False    D    Cherbourg   yes   True
571       False    C  Southampton   yes  False
438        True    C  Southampton    no  False
```

y_train

```
581    0.692586
224    0.692586
583   -1.443865
724    0.692586
370    0.692586
         ...
523    0.692586
92    -1.443865
462   -1.443865
879    0.692586
512    0.692586
Name: survived, Length: 136, dtype: float64
```

y_test

```
118   -1.443865
251   -1.443865
742    0.692586
496    0.692586
712    0.692586
96    -1.443865
139   -1.443865
337    0.692586
572    0.692586
487   -1.443865
486    0.692586
```

```
765    0.692586
340    0.692586
550    0.692586
262   -1.443865
97     0.692586
291    0.692586
627    0.692586
492   -1.443865
307    0.692586
857    0.692586
599    0.692586
707    0.692586
183    0.692586
54    -1.443865
609    0.692586
318    0.692586
110   -1.443865
789   -1.443865
701    0.692586
835    0.692586
305    0.692586
456   -1.443865
430    0.692586
332   -1.443865
177   -1.443865
585    0.692586
331   -1.443865
336   -1.443865
193    0.692586
75    -1.443865
248    0.692586
625   -1.443865
473    0.692586
571    0.692586
438   -1.443865
Name: survived, dtype: float64
```