

```

#TITANIC DATA SET
import numpy as np
import pandas as pd

titanic=pd.read_csv(r'/Users/sasidharbhagavatula/Desktop/titanicdata.csv')

del titanic['Name']
titanic.head()
del titanic['Ticket']
titanic.head()
del titanic['Fare']
titanic.head()
del titanic['Cabin']
titanic.head()

# Changing Value for "Male, Female" string values to numeric values , male=1 and female=2
def getNumber(str):
    if str=="male":
        return 1
    else:
        return 2
titanic["Gender"]=titanic["Sex"].apply(getNumber)
#We have created a new column called "Gender" and
#filling it with values 1,2 based on the values of sex column
titanic.head()

del titanic['Sex']
titanic.head()

titanic.isnull().sum()
### Fill the null values of the Age column.
#Fill mean Survived age(mean age of the survived people) in the column where the
#person has survived and mean not Survived age (mean age of the people who have
# not survived) in the column where person has not survived###
meanS= titanic[titanic.Survived==1].Age.mean()
meanS
titanic["age"]=np.where(pd.isnull(titanic.Age) & titanic["Survived"]==1 ,meanS,meanS)
titanic.head()

titanic.isnull().sum()

# Finding the mean age of "Not Survived" people
meanNS=titanic[titanic.Survived==0].Age.mean()
meanNS

titanic.age.fillna(meanNS,inplace=True)
titanic.head()

titanic.isnull().sum()

# delete Age
del titanic['Age']
titanic.head()

```

```

# Finding the number of people who have survived
# given that they have embarked or boarded from a particular port

survivedQ = titanic[titanic.Embarked == 'Q'][titanic.Survived == 1].shape[0]
survivedC = titanic[titanic.Embarked == 'C'][titanic.Survived == 1].shape[0]
survivedS = titanic[titanic.Embarked == 'S'][titanic.Survived == 1].shape[0]
print(survivedQ)
print(survivedC)
print(survivedS)

survivedQ = titanic[titanic.Embarked == 'Q'][titanic.Survived == 0].shape[0]
survivedC = titanic[titanic.Embarked == 'C'][titanic.Survived == 0].shape[0]
survivedS = titanic[titanic.Embarked == 'S'][titanic.Survived == 0].shape[0]
print(survivedQ)
print(survivedC)
print(survivedS)

titanic.dropna(inplace=True)
titanic.head()
titanic.isnull().sum()

def getEmb(str):
    if str=="S":
        return 1
    elif str=='Q':
        return 2
    else:
        return 3
titanic["Embark"]=titanic["Embarked"].apply(getEmb)
titanic.head()

del titanic['Embarked']
titanic.rename(columns={'Embark':'Embarked'}, inplace=True)
titanic.head()

#Drawing a pie chart for number of males and females aboard
import matplotlib.pyplot as plt
from matplotlib import style

males = (titanic['Gender'] == 1).sum()
#Summing up all the values of column gender with a
#condition for male and similiary for females
females = (titanic['Gender'] == 2).sum()
print(males)
print(females)
p = [males, females]
plt.pie(p,      #giving array
        labels = ['Male', 'Female'], #Correspdingly giving labels
        colors = ['green', 'yellow'], # Corresponding colors
        explode = (0.15, 0),         #How much the gap should me there between the pi
        startangle = 0) #what start angle should be given
plt.axis('equal')
plt.show()

```

```

MaleS=titanic[titanic.Gender==1][titanic.Survived==1].shape[0]
print(MaleS)
MaleN=titanic[titanic.Gender==1][titanic.Survived==0].shape[0]
print(MaleN)
FemaleS=titanic[titanic.Gender==2][titanic.Survived==1].shape[0]
print(FemaleS)
FemaleN=titanic[titanic.Gender==2][titanic.Survived==0].shape[0]
print(FemaleN)

chart=[MaleS, MaleN, FemaleS, FemaleN]
colors=['lightskyblue', 'yellowgreen', 'Yellow', 'Orange']
labels=["Survived Male", "Not Survived Male", "Survived Female", "Not Survived Female"]
explode=[0, 0.05, 0, 0.1]
plt.pie(chart, labels=labels, colors=colors, explode=explode, startangle=100, counterclock=True)
plt.axis("equal")
plt.show()

```

Python File Edit Search Source Run Debug Consoles Projects Tools View Help

Spyder (Python 3.13)

/Users/sasidharbhagavatula/Desktop/python/ML/titanic.py

```
temp.py x MLprog.py x untitled1.py* x titanic.py
titanic.head()
del titanic['Ticket']
titanic.head()
del titanic['Fare']
titanic.head()
del titanic['Cabin']
titanic.head()

# Changing Value for "Male, Female" string values to numeric values , mal
def getNumber(str):
    if str=="male":
        return 1
    else:
        return 2
titanic["Gender"]=titanic["Sex"].apply(getNumber)
#We have created a new column called "Gender" and
#filling it with values 1,2 based on the values of sex column
titanic.head()

del titanic['Sex']
titanic.head()

titanic.isnull().sum()
### Fill the null values of the Age column.
#Fill mean Survived age(mean age of the survived people) in the column wh
#person has survived and mean not Survived age (mean age of the people w
# in the column where person has not survived###
meanS= titanic[titanic.Survived==1].Age.mean()
meanNS=
titanic["age"]=np.where(pd.isnull(titanic.Age) & titanic["Survived"]==1 -
titanic.head()

titanic.isnull().sum()

# Finding the mean age of "Not Survived" people
meanNS=titanic[titanic.Survived==0].Age.mean()
meanNS

titanic.age.fillna(meanNS, inplace=True)
```

53 %

Category	Percentage
Not Survived Female	9.11%
Survived Male	12.26%
Survived Female	25.98%
Not Survived Male	52.64%

Help Variable Explorer Debugger Plots Files

x Console 1/A

IPython Console History

Update available: 3.13.5 -> 3.13.6

Index	PassengerId	Survived	Pclass	SibSp	Parch	Gender	age	Embarked
0	1	0	3	1	0	1	22	1
1	2	1	1	1	0	2	38	3
2	3	1	3	0	0	2	26	1
3	4	1	1	1	0	2	35	1
4	5	0	3	0	0	1	35	1
5	6	0	3	0	0	1	30.6262	2
6	7	0	1	0	0	1	54	1
7	8	0	3	3	1	1	2	1
8	9	1	3	0	2	2	27	1
9	10	1	2	1	0	2	14	3
10	11	1	3	1	1	2	4	1
11	12	1	1	0	0	2	58	1
12	13	0	3	0	0	1	20	1

Save and Close

Close