```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving indian_liver_patient.csv to indian_liver_patient.csv
```

```
data = pd.read_csv('indian_liver_patient.csv')
```

```
data.head()
```

| | Age | Gender | Total_Bilirubin | Direct_Bilirubin | Alkaline_Phosphotase | Alamine_Aminot |
|---|---|---|---|---|---|---|
| **0** | 65 | Female | 0.7 | 0.1 | 187 | |
| **1** | 62 | Male | 10.9 | 5.5 | 699 | |
| **2** | 62 | Male | 7.3 | 4.1 | 490 | |
| **3** | 58 | Male | 1.0 | 0.4 | 182 | |
| **4** | 72 | Male | 3.9 | 2.0 | 195 | |

```
data.tail()
```

| | Age | Gender | Total_Bilirubin | Direct_Bilirubin | Alkaline_Phosphotase | Alamine_Aminotransferase | As |
|---|---|---|---|---|---|---|---|
| **578** | 60 | Male | 0.5 | 0.1 | 500 | 20 | |
| **579** | 40 | Male | 0.6 | 0.1 | 98 | 35 | |
| **580** | 52 | Male | 0.8 | 0.2 | 245 | 48 | |
| **581** | 31 | Male | 1.3 | 0.5 | 184 | 29 | |
| **582** | 38 | Male | 1.0 | 0.3 | 216 | 21 | |

```
data.describe()
```

| | Age | Total_Bilirubin | Direct_Bilirubin | Alkaline_Phosphotase | Alamine_Aminotransferase | As |
|---|---|---|---|---|---|---|
| **count** | 583.000000 | 583.000000 | 583.000000 | 583.000000 | 583.000000 | |
| **mean** | 44.746141 | 3.298799 | 1.486106 | 290.576329 | 80.713551 | |
| **std** | 16.189833 | 6.209522 | 2.808498 | 242.937989 | 182.620356 | |
| **min** | 4.000000 | 0.400000 | 0.100000 | 63.000000 | 10.000000 | |
| **25%** | 33.000000 | 0.800000 | 0.200000 | 175.500000 | 23.000000 | |
| **50%** | 45.000000 | 1.000000 | 0.300000 | 208.000000 | 35.000000 | |

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 583 entries, 0 to 582
Data columns (total 11 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   Age                         583 non-null    int64  
 1   Gender                      583 non-null    object 
 2   Total_Bilirubin             583 non-null    float64
 3   Direct_Bilirubin            583 non-null    float64
 4   Alkaline_Phosphotase        583 non-null    int64  
 5   Alamine_Aminotransferase    583 non-null    int64  
 6   Aspartate_Aminotransferase  583 non-null    int64  
 7   Total_Protiens              583 non-null    float64
 8   Albumin                     583 non-null    float64
 9   Albumin_and_Globulin_Ratio  579 non-null    float64
 10  Dataset                     583 non-null    int64  
dtypes: float64(5), int64(5), object(1)
memory usage: 50.2+ KB
```

```
data.isnull().any()
```

```
Age                           False
Gender                        False
Total_Bilirubin               False
Direct_Bilirubin              False
Alkaline_Phosphotase          False
Alamine_Aminotransferase      False
Aspartate_Aminotransferase    False
Total_Protiens                False
Albumin                       False
Albumin_and_Globulin_Ratio     True
Dataset                       False
dtype: bool
```

```
data.isnull().sum()
```

```
Age                           0
Gender                        0
Total_Bilirubin               0
Direct_Bilirubin              0
Alkaline_Phosphotase          0
Alamine_Aminotransferase      0
Aspartate_Aminotransferase    0
Total_Protiens                0
Albumin                       0
Albumin_and_Globulin_Ratio    4
Dataset                       0
dtype: int64
```

```
data[data['Dataset']==1]
```

| | Age | Gender | Total_Bilirubin | Direct_Bilirubin | Alkaline_Phosphotase | Alamine_Aminotransferase | As |
|---|---|---|---|---|---|---|---|
| **0** | 65 | Female | 0.7 | 0.1 | 187 | 16 | |
| **1** | 62 | Male | 10.9 | 5.5 | 699 | 64 | |
| **2** | 62 | Male | 7.3 | 4.1 | 490 | 60 | |
| **3** | 58 | Male | 1.0 | 0.4 | 182 | 14 | |
| **4** | 72 | Male | 3.9 | 2.0 | 195 | 27 | |
| **...** | ... | ... | ... | ... | ... | ... | |
| **576** | 32 | Male | 15.0 | 8.2 | 289 | 58 | |
| **577** | 32 | Male | 12.7 | 8.4 | 190 | 28 | |
| **579** | 40 | Male | 0.6 | 0.1 | 98 | 35 | |
| **580** | 52 | Male | 0.8 | 0.2 | 245 | 48 | |
| **581** | 31 | Male | 1.3 | 0.5 | 184 | 29 | |

416 rows × 11 columns

```
data['Dataset'].unique()
```

```
array([1, 2])
```

```
data.isnull().sum()
```

```
Age                         0
Gender                      0
Total_Bilirubin             0
Direct_Bilirubin            0
Alkaline_Phosphotase        0
Alamine_Aminotransferase    0
Aspartate_Aminotransferase  0
Total_Protiens              0
Albumin                     0
Albumin_and_Globulin_Ratio  4
Dataset                     0
dtype: int64
```

```
#mode imputation
#data['Albumin_and_Globuin_Ratio'] = data.fillna(data['Albumin_and_Globulin_ratio'].mpde()[0])
```

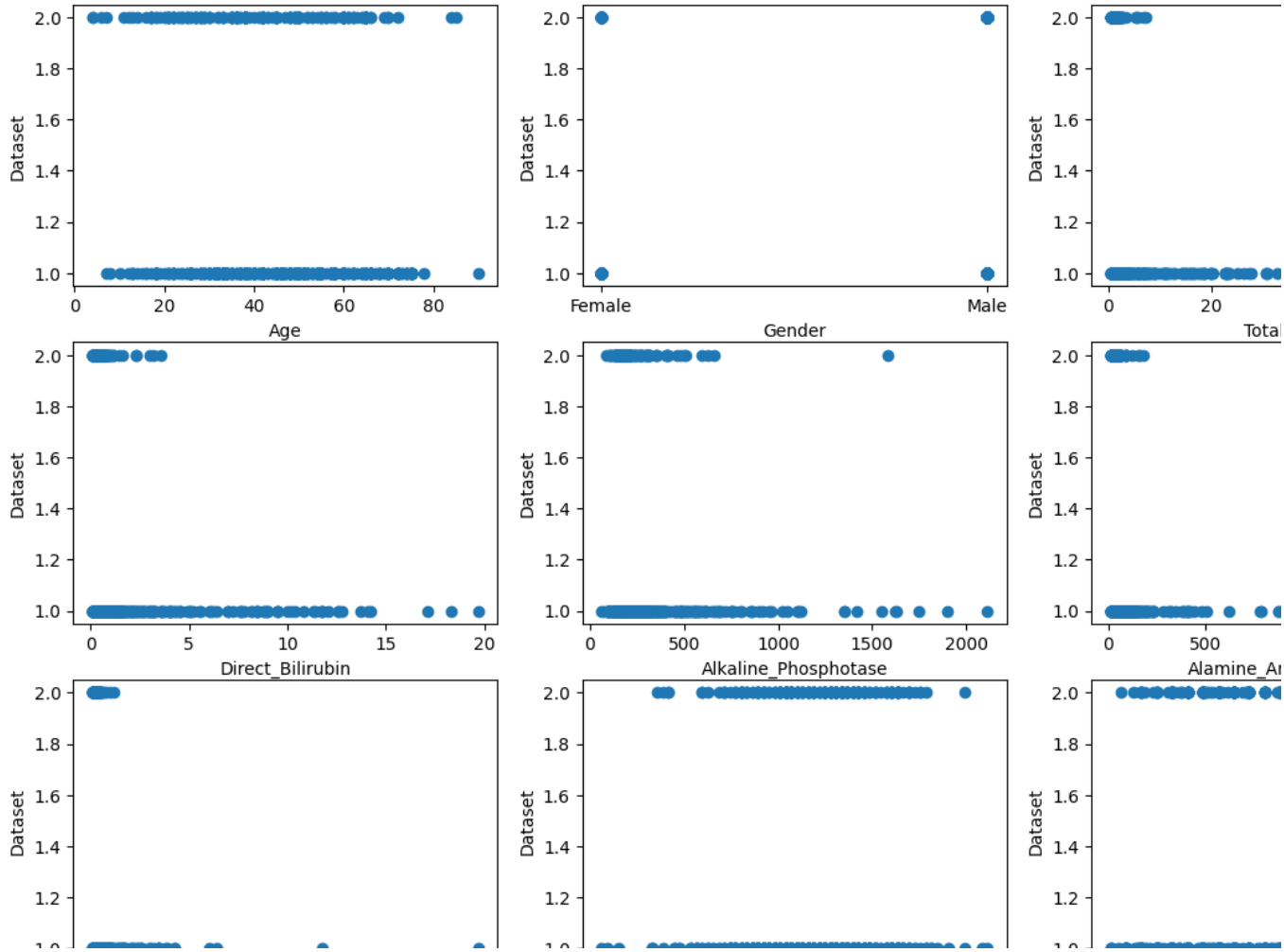
```
data_1 = data.dropna()
```

```
data_1.isnull().sum()
```

```
Age                         0
Gender                      0
Total_Bilirubin             0
Direct_Bilirubin            0
Alkaline_Phosphotase        0
Alamine_Aminotransferase    0
Aspartate_Aminotransferase  0
Total_Protiens              0
Albumin                     0
Albumin_and_Globulin_Ratio  0
Dataset                     0
dtype: int64
```

```
plt.figure(figsize=(15,10))
plt.subplot(3,3,1)
plt.scatter(data_1['Age'], data_1['Dataset'])
plt.ylabel('Dataset')
plt.xlabel('Age')

plt.subplot(3,3,2)
plt.scatter(data_1['Gender'],data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Gender')

plt.subplot(3,3,3)
plt.scatter(data_1['Total_Bilirubin'], data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Total_Bilirubin')

plt.subplot(3,3,4)
plt.scatter(data_1['Direct_Bilirubin'], data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Direct_Bilirubin')

plt.subplot(3,3,5)
plt.scatter(data_1['Alkaline_Phosphotase'], data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Alkaline_Phosphotase')

plt.subplot(3,3,6)
plt.scatter(data_1['Alamine_Aminotransferase'],data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Alamine_Aminotransferase')

plt.subplot(3,3,7)
plt.scatter(data_1['Aspartate_Aminotransferase'], data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Aspartate_Aminotransferase')

plt.subplot(3,3,8)
plt.scatter(data_1['Total_Protiens'], data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Total_Protiens')

plt.subplot(3,3,9)
plt.scatter(data_1['Albumin_and_Globulin_Ratio'], data_1['Dataset'],)
plt.ylabel('Dataset')
plt.xlabel('Albumin_and_Globulin_Ratio')
```

```
Text(0.5, 0, 'Albumin_and_Globulin_Ratio')
```

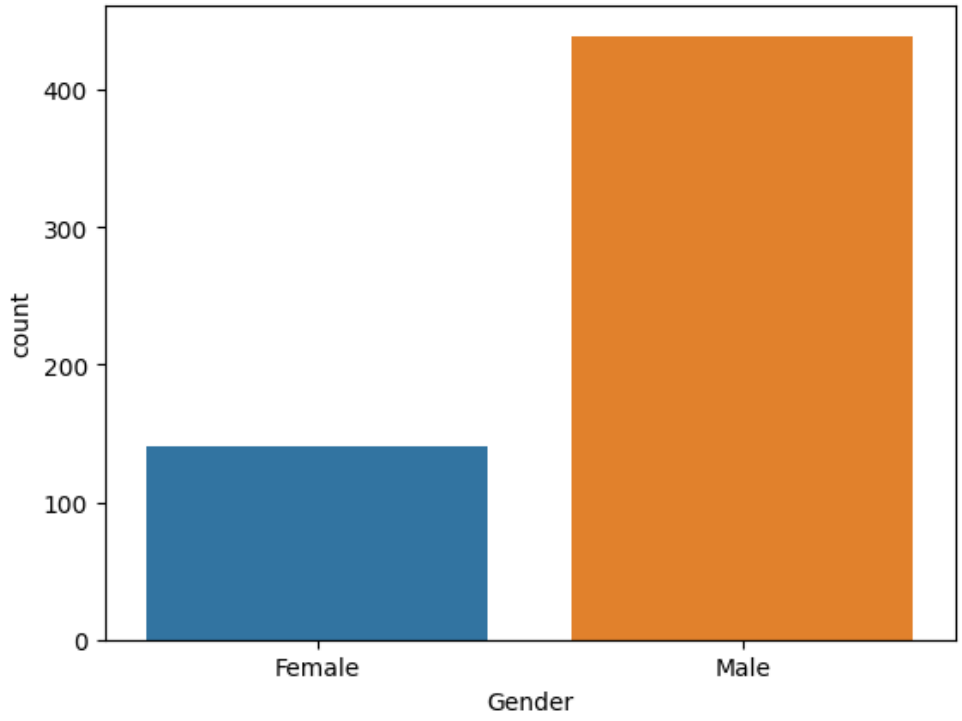



```
sns.countplot(data=data_1, x = 'Dataset')
LD,NLD=data_1['Dataset'].value_counts()
print("liver disease patinets:",LD)
print("Noo-liver disease patinets:",NLD)
```

```
liver disease patinets: 414
Noo-liver disease patinets: 165
```


```
sns.countplot(data_1, x = 'Gender', label='Count')
m,f=data_1['Gender'].value_counts()
print("No of Males:",m)
print("No of females:",f)
```

```
No of Males: 439
No of females: 140
```




```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
data_1['Gender'] = le.fit_transform(data_1['Gender'])
data_1.head()
```

```
<ipython-input-19-063e78f52f00>:3: 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.
  data_1['Gender'] = le.fit_transform(data_1['Gender'])
```

| | Age | Gender | Total_Bilirubin | Direct_Bilirubin | Alkaline_Phosphotase | Alamine_Aminotransferase | Aspar |
|---|---|---|---|---|---|---|---|
| **0** | 65 | 0 | 0.7 | 0.1 | 187 | 16 | |
| **1** | 62 | 1 | 10.9 | 5.5 | 699 | 64 | |
| **2** | 62 | 1 | 7.3 | 4.1 | 490 | 60 | |
| **3** | 58 | 1 | 1.0 | 0.4 | 182 | 14 | |
| **4** | 72 | 1 | 3.9 | 2.0 | 195 | 27 | |

```
x=data_1.iloc[:,0:-1]
y=data_1.iloc[:,-1]
```

```
x=data_1.iloc[:,0:-1]
y=data_1.iloc[:,-1]
```

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.3)
```

```
xtrain.shape
```

```
(405, 10)
```

```
xtest.shape
```

```
(174, 10)
```

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
```

```
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
```

```
svm=SVC()
RFmodel=RandomForestClassifier()
KNNmodel=KNeighborsClassifier()
```

```
from sklearn.svm import SVC
svm=SVC()
```

```
svm.fit(xtrain, ytrain)
```

```
▾ SVC
SVC()
```

```
SVCpred=svm.predict(xtest)
```

```
from sklearn.metrics import accuracy_score,confusion_matrix
SVCaccuracy=accuracy_score(SVCpred,ytest)
SVCaccuracy
```

```
0.7011494252873564
```

```
SVCcm=confusion_matrix(SVCpred, ytest)
SVCcm
```

```
array([[122,  52],
       [  0,   0]])
```

```
from sklearn.ensemble import RandomForestClassifier
RFmodel=RandomForestClassifier()
```

```
#Random Forest Classifier model
RFmodel.fit(xtrain, ytrain)
```

▾ RandomForestClassifier

RandomForestClassifier()

```
RFpred=RFmodel.predict(xtest)
```

```
RFaccuracy=accuracy_score(RFpred, ytest)
RFaccuracy
```

```
0.7011494252873564
```

```
RFcm=confusion_matrix(RFpred, ytest)
RFcm
```

```
array([[112,  42],
       [ 10,  10]])
```

```
from sklearn.neighbors import KNeighborsClassifier
KNN = KNeighborsClassifier()
```

```
KNN.fit(xtrain, ytrain)
```

▾ KNeighborsClassifier

KNeighborsClassifier()

```
KNNpred=KNN.predict(xtest)
```

```
KNNaccuracy=accuracy_score(KNNpred, ytest)
KNNaccuracy
```

```
0.6149425287356322
```

```
KNNcm=confusion_matrix(KNNpred, ytest)
KNNcm
```

```
array([[92, 37],
       [30, 15]])
```

```
print("Support Vector Machine Algorithm accuracy score : {value:.2f} %".format(value=SVCaccuracy*100))
print("Random Forest Algorithm Accuracy SCore : {value:.2f} %".format(value=RFaccuracy*100))
print("K-Nearest Neighbors Algorithm accuracy score : {value:.2f} %".format(value=KNNaccuracy*100))
```

```
Support Vector Machine Algorithm accuracy score : 70.11 %
Random Forest Algorithm Accuracy SCore : 70.11 %
K-Nearest Neighbors Algorithm accuracy score : 61.49 %
```

```
import pickle
pickle.dump(svm, open('liver_analysis_1 .pk1','wb'))
```