```
**Kirti Vichare
Roll No: 1233**
```

Kirti Vichare Roll No: 1233

```
import  numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

```
dataset = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/breast_cancer.csv')
```

```
dataset.head()
```

	Sample code number	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Blan Chromat:
0	1000025	5	1	1	1	2	1	
1	1002945	5	4	4	5	7	10	
2	1015425	3	1	1	1	2	2	
3	1016277	6	8	8	1	3	4	
4	1017023	4	1	1	3	2	1	

```
dataset.sample(5)
```

Sample code	Clump Thickness	Uniformity of Cell	Uniformity of Cell	Marginal Adhesion	Single Epithelial	Bare Nuclei	B Chrom

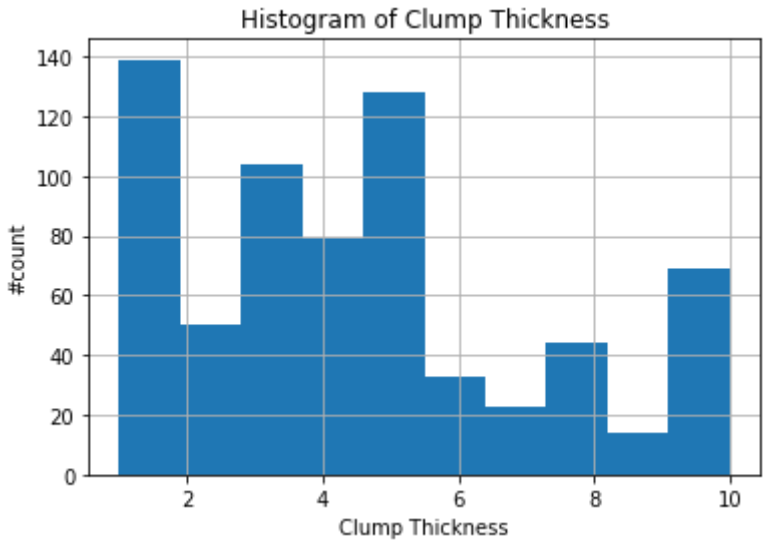
```
dataset.Class.value_counts
```

```
<bound method IndexOpsMixin.value_counts of 0      2
1      2
2      2
3      2
4      2
      ..
678    2
679    2
680    4
681    4
682    4
Name: Class, Length: 683, dtype: int64>
```

```
dataset['Clump Thickness'].unique()
```

```
array([ 5,  3,  6,  4,  8,  1,  2,  7, 10,  9])
```

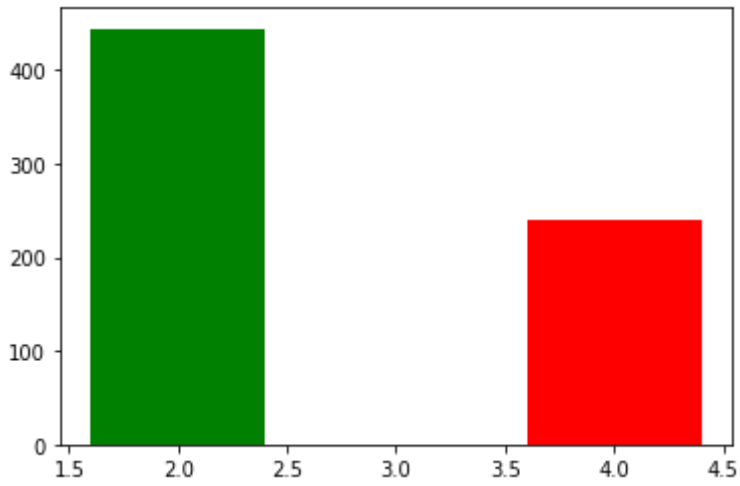
```
dataset['Clump Thickness'].hist(bins=10)
dataset['Clump Thickness'].value_counts(sort=False)
plt.xlabel("Clump Thickness")
plt.ylabel("#count")
plt.title('Histogram of Clump Thickness')
plt.show()
```




```
#display graph of class
import seaborn as sns
sns.countplot(dataset.Class.value_counts(),data = dataset)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f2fca5c72d0>
```


```
plt.bar(dataset.Class.unique(),dataset.Class.value_counts(),color=['green','red'])
plt.show()
```



Importing Dataset

```
X = dataset.iloc[:,1:-1].values
y = dataset.iloc[:, -1].values
```

```
#divdie data into train test split
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2,random_state=0)

#display X_train and X_test recorded counts
print(X_train.shape[0])
print(X_test.shape[0])
```

```
546
137
```

Feature Scaling

```
#Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)
```

```
X_train
```

```
array([[ 1.98839518, -0.69781134, -0.74152574, ...,  0.61907387,
         0.34532102, -0.33863738],
       [-1.22468404, -0.69781134, -0.74152574, ..., -0.18860673,
        -0.62157783, -0.33863738],
       [ 0.20335117, -0.69781134, -0.74152574, ..., -0.18860673,
        -0.62157783, -0.33863738],
       ...,
       [-1.22468404, -0.69781134, -0.74152574, ..., -0.99628733,
        -0.62157783, -0.33863738],
       [-0.51066644, -0.69781134, -0.74152574, ..., -0.59244703,
        -0.62157783, -0.33863738],
       [ 1.98839518,  1.90512627,  1.27779124, ...,  1.42675446,
         1.31221987, -0.33863738]])
```

Training Logistic Regression

```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression(solver = 'liblinear',random_state = 0)
logreg.fit(X_train, y_train)
```

```
LogisticRegression(random_state=0, solver='liblinear')
```

Confusion MAtrix

```
y_pred = logreg.predict(X_test)
```

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
print(cm)
```

```
[[84  3]
 [ 3 47]]
```

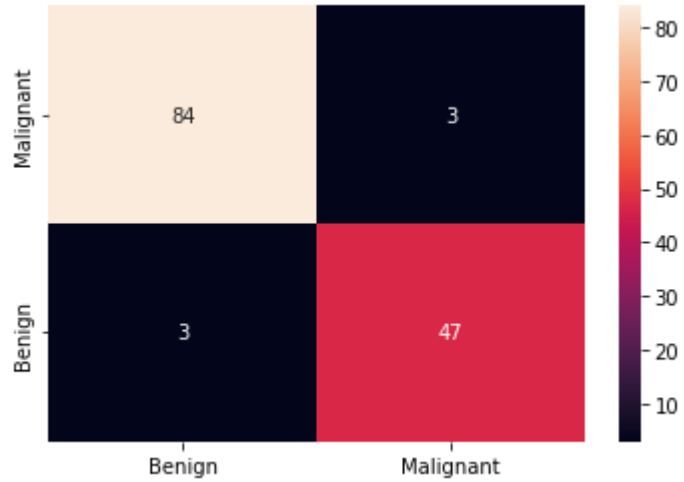
```
from sklearn.metrics import accuracy_score
print("Accuracy", accuracy_score(y_test, y_pred))
```

```
Accuracy 0.9562043795620438
```

```
import seaborn as sns
```

```
sns.heatmap(confusion_matrix(y_test,y_pred),annot = True,xticklabels={"Malignant","Benign"
            yticklabels = {"Benign" , "Malignant"})
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2fca374810>
```




```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = logreg,X = X_train, y = y_train,cv = 10)
print ("Accuracy:",accuracies.std()*100)
print("satndard deviation:",accuracies.std()*100)
```

```
Accuracy: 1.8669995719100532
satndard deviation: 1.8669995719100532
```

✓ 0s completed at 4:15 PM