```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df=pd.read_csv('bstc.csv')
df.head()
```

```
   Sample code number  Clump Thickness  Uniformity of Cell Size  \
0             1000025                5                        1
1             1002945                5                        4
2             1015425                3                        1
3             1016277                6                        8
4             1017023                4                        1

   Uniformity of Cell Shape  Marginal Adhesion  Single Epithelial Cell
Size  \
0                         1                  1
2
1                         4                  5
7
2                         1                  1
2
3                         8                  1
3
4                         1                  3
2

   Bare Nuclei  Bland Chromatin  Normal Nucleoli  Mitoses  Class
0          1.0                3                1        1      0
1         10.0                3                2        1      0
2          2.0                3                1        1      0
3          4.0                3                7        1      0
4          1.0                3                1        1      0
```

```
df.tail()
```

```
     Sample code number  Clump Thickness  Uniformity of Cell Size  \
694              776715                3                        1
695              841769                2                        1
696              888820                5                       10
697              897471                4                        8
698              897471                4                        8

     Uniformity of Cell Shape  Marginal Adhesion  Single Epithelial
Cell Size  \
694                         1                  1
3
695                         1                  1
2
696                        10                  3
7
```

```
697                           6                  4
3
698                           8                  5
4

     Bare Nuclei  Bland Chromatin  Normal Nucleoli  Mitoses  Class
694          2.0                1                1        1      0
695          1.0                1                1        1      0
696          3.0                8               10        2      1
697          4.0               10                6        1      1
698          5.0               10                4        1      1
```

```
df.size
```

```
7689
```

```
df.count()
```

```
Sample code number             699
Clump Thickness                699
Uniformity of Cell Size        699
Uniformity of Cell Shape       699
Marginal Adhesion              699
Single Epithelial Cell Size    699
Bare Nuclei                    683
Bland Chromatin                699
Normal Nucleoli                699
Mitoses                        699
Class                          699
dtype: int64
```

```
df.fillna(method='bfill',inplace=True)
```

```
df.count()
```

```
Sample code number             699
Clump Thickness                699
Uniformity of Cell Size        699
Uniformity of Cell Shape       699
Marginal Adhesion              699
Single Epithelial Cell Size    699
Bare Nuclei                    699
Bland Chromatin                699
Normal Nucleoli                699
Mitoses                        699
Class                          699
dtype: int64
```

```
df['Class'].value_counts()
```

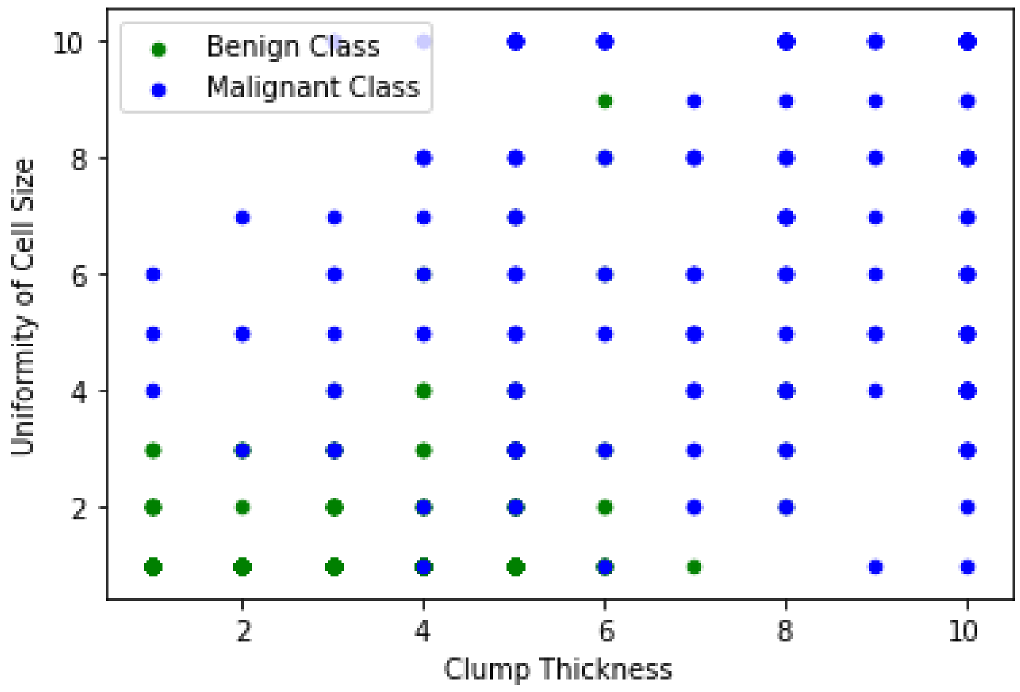
```
0    458
1    241
Name: Class, dtype: int64
```

```
benign=df[df['Class']==0]
malignant=df[df['Class']==1]
```

```
axes=benign.plot(kind='scatter',x='Clump Thickness',y='Uniformity of
Cell Size',color='green',label='Benign Class')
```

```
malignant.plot(kind='scatter',x='Clump Thickness',y='Uniformity of
Cell Size',color='blue',label='Malignant Class',ax=axes)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x24d12874f48>
```



```
df.dtypes
```

```
Sample code number               int64
Clump Thickness                  int64
Uniformity of Cell Size          int64
Uniformity of Cell Shape         int64
Marginal Adhesion                int64
Single Epithelial Cell Size      int64
Bare Nuclei                    float64
Bland Chromatin                  int64
Normal Nucleoli                  int64
Mitoses                          int64
```

```
Class                              int64
dtype: object
```

```
df.columns
```

```
Index(['Sample code number', 'Clump Thickness', 'Uniformity of Cell
Size',
       'Uniformity of Cell Shape', 'Marginal Adhesion',
       'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland
Chromatin',
       'Normal Nucleoli', 'Mitoses', 'Class'],
      dtype='object')
```

```
 feature= df[['Clump Thickness','Uniformity of Cell Size','Uniformity
of Cell Shape','Marginal Adhesion','Single Epithelial Cell Size','Bare
Nuclei','Bland Chromatin',
              'Normal Nucleoli','Mitoses']]
```

```
 x = np.asarray(feature)
 y= np.asarray(df['Class'])
```

```
x
```

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

```
y
```

```
array([0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1,
1,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1,
1,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0,
1,
       0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0,
1,
       0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0,
0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1,
0,
       0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
1,
       0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
0,
       0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1,
0,
       0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0,
0,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1,
1,
       1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1,
1,
       1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0,
0,
       1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1,
1,
       1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
0,
       0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0,
0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1,
0,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0,
0,
       0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
1,
       0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0,
0,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1,
1,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1,
0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
0,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1],
dtype=int64)
```

```python
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=2)
```

```
x_train.shape
```

```
(559, 9)
```

```
x_test.shape
```

```
(140, 9)
```

```
y_train.shape
```

```
(559,)
```

```
y_test.shape
```

```
(140,)
```

```
from sklearn import svm
```

```
model=svm.SVC(kernel='linear',gamma='auto',C=0.3)
```

```
model.fit(x_train,y_train)
```

```
SVC(C=0.3, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto',
kernel='linear',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

```
y_predict =model.predict(x_test)
```

```
from sklearn.metrics import classification_report
```

```
print(classification_report(y_test,y_predict))
#precision of our prediction is 98% for benign and 91% for malignant
```

```
              precision    recall  f1-score   support

           0       0.98      0.96      0.97        99
           1       0.91      0.95      0.93        41

    accuracy                           0.96       140
   macro avg       0.94      0.96      0.95       140
weighted avg       0.96      0.96      0.96       140
```