```
In [1]: ###Load data from CSV file

        import pandas as pd
```

```
In [2]: CycleCapacity=pd.read_csv('Cycle n Capacity.csv')
```

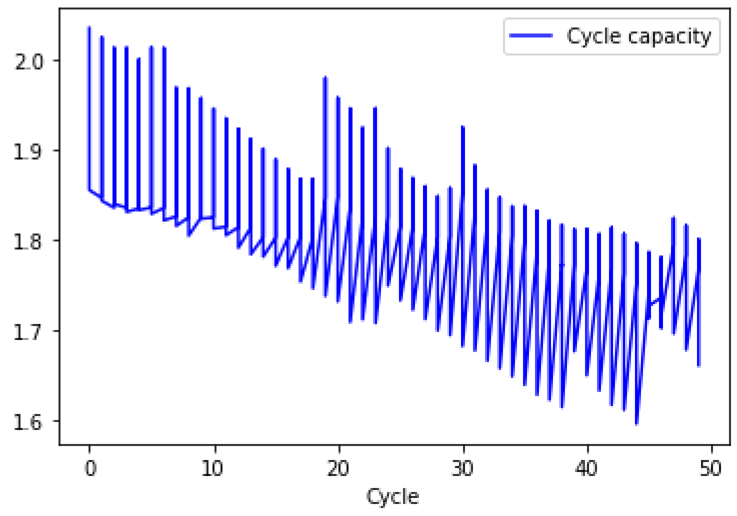
```
In [3]: CycleCapacity.shape
Out[3]: (636, 7)
```

```
In [4]: CycleCapacity.size
Out[4]: 4452
```

```
In [5]: CycleCapacity.count()
Out[5]: Cycle                   636
        Capacity(Ah)            636
        Voltage Measured(V)     636
        Current Measured        636
        Temperature Measured    636
        Time Measured(Sec)      636
        SampleId                636
        dtype: int64
```

```
In [6]: ###Distribution of the classes

        CycleC = CycleCapacity[1:200]
        CycleC.plot( x='Cycle', y='Capacity(Ah)', color='blue', label='Cycle capacity'
        )
Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x23fb058ad88>
```



In [7]:
```
CycleCapacity.dtypes
```

Out[7]:
```
Cycle                    int64
Capacity(Ah)           float64
Voltage Measured(V)    float64
Current Measured       float64
Temperature Measured   float64
Time Measured(Sec)     float64
SampleId                object
dtype: object
```

In [8]:
```
###Identifying Unwanted Rows

CycleCapacity.dtypes
CycleCapacity=CycleCapacity[pd.to_numeric(CycleCapacity['Cycle'],errors='coerc
e').notnull()]
CycleCapacity['Cycle'] = CycleCapacity['Cycle'].astype('int')
CycleCapacity.dtypes
```

Out[8]:
```
Cycle                    int32
Capacity(Ah)           float64
Voltage Measured(V)    float64
Current Measured       float64
Temperature Measured   float64
Time Measured(Sec)     float64
SampleId                object
dtype: object
```

In [9]:
```
CycleCapacity.columns
```

Out[9]:
```
Index(['Cycle', 'Capacity(Ah)', 'Voltage Measured(V)', 'Current Measured',
       'Temperature Measured', 'Time Measured(Sec)', 'SampleId'],
      dtype='object')
```

In [10]:
```
import numpy as np
CycleCapacity.columns
features = CycleCapacity[['Cycle', 'Capacity(Ah)', 'Voltage Measured(V)', 'Cur
rent Measured','Temperature Measured', 'Time Measured(Sec)', 'SampleId']]
x=np.asarray(features)
y=np.array(CycleCapacity['Capacity(Ah)'])
x[1:10]
```

Out[10]:
```
array([[0, 2.0353375910056, 2.47576775682819, -2.00943589187254,
        39.1629865323871, 3690.2340000000004, 'B0006'],
       [0, 1.89105229539079, 3.0621127090856803, -0.00143329876408607,
        37.3384784889293, 3690.2340000000004, 'B0007'],
       [0, 1.85500452079108, 3.0532303394443305, -0.00243341458390145,
        37.2056712984427, 3434.8909999999996, 'B0018'],
       [1, 1.84632724971993, 3.30024488712225, -0.000447552579117187,
        34.392136587188695, 3672.344, 'B0005'],
       [1, 2.02514024603141, 2.35152551190002, -2.01037485552297,
        39.2462026826329, 3672.344, 'B0006'],
       [1, 1.88063702768686, 3.07922623823352, -0.00323038524139762,
        37.1617386014679, 3672.344, 'B0007'],
       [1, 1.8431955317089999, 3.0882001701258304, -0.000910647531236055,
        37.1554752249802, 3425.485, 'B0018'],
       [2, 1.83534919422341, 3.3274510098686303, 0.00102601855936441,
        34.2327787157633, 3651.6409999999996, 'B0005'],
       [2, 2.01332637134546, 2.44047971510654, -2.00855888711011,
        38.999202460252796, 3651.6409999999996, 'B0006']], dtype=object)
```

In [11]:
```
###Divide Data into train/test data set

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=
4)
x_train.shape #508 * 7
x_test.shape #128 *7
```

Out[11]:
```
(128, 7)
```

In [12]:
```
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=
4)
y_train.shape # 508
y_test.shape #128
```

Out[12]:
```
(128,)
```

In [13]:
```
###Modeling


from sklearn import svm
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=
4)
classifier = svm.SVC(kernel='linear',gamma='auto',C=0)
classifier
#classifier.fit(x_train,y_train)
#y_predict = classifier.predict(x_test)
```

Out[13]:
```
SVC(C=0, break_ties=False, 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)
```

In [14]:
```
from sklearn.metrics import classification_report
#print(classification_report(y_test, y_predict))
```

In [ ]:

In [ ]:

In [ ]: