In [1]:

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

In [2]:

```
data = pd.read_csv("room_occupancy.csv")
```

In [3]:

```
data
```

Out[3]:

| | date | Temperature | Humidity | Light | CO2 | HumidityRatio | Occupancy |
|---|---|---|---|---|---|---|---|
| **0** | 02-02-2015 14:19 | 23.7000 | 26.2720 | 585.200000 | 749.200000 | 0.004764 | 1 |
| **1** | 02-02-2015 14:19 | 23.7180 | 26.2900 | 578.400000 | 760.400000 | 0.004773 | 1 |
| **2** | 02-02-2015 14:21 | 23.7300 | 26.2300 | 572.666667 | 769.666667 | 0.004765 | 1 |
| **3** | 02-02-2015 14:22 | 23.7225 | 26.1250 | 493.750000 | 774.750000 | 0.004744 | 1 |
| **4** | 02-02-2015 14:23 | 23.7540 | 26.2000 | 488.600000 | 779.000000 | 0.004767 | 1 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **10803** | 10-02-2015 09:29 | 21.0500 | 36.0975 | 433.000000 | 787.250000 | 0.005579 | 1 |
| **10804** | 10-02-2015 09:29 | 21.0500 | 35.9950 | 433.000000 | 789.500000 | 0.005563 | 1 |
| **10805** | 10-02-2015 09:30 | 21.1000 | 36.0950 | 433.000000 | 798.500000 | 0.005596 | 1 |
| **10806** | 10-02-2015 09:32 | 21.1000 | 36.2600 | 433.000000 | 820.333333 | 0.005621 | 1 |
| **10807** | 10-02-2015 09:33 | 21.1000 | 36.2000 | 447.000000 | 821.000000 | 0.005612 | 1 |

10808 rows × 7 columns

In [4]:

```
data.head()
```

Out[4]:

| | date | Temperature | Humidity | Light | CO2 | HumidityRatio | Occupancy |
|---|---|---|---|---|---|---|---|
| **0** | 02-02-2015 14:19 | 23.7000 | 26.272 | 585.200000 | 749.200000 | 0.004764 | 1 |
| **1** | 02-02-2015 14:19 | 23.7180 | 26.290 | 578.400000 | 760.400000 | 0.004773 | 1 |
| **2** | 02-02-2015 14:21 | 23.7300 | 26.230 | 572.666667 | 769.666667 | 0.004765 | 1 |
| **3** | 02-02-2015 14:22 | 23.7225 | 26.125 | 493.750000 | 774.750000 | 0.004744 | 1 |
| **4** | 02-02-2015 14:23 | 23.7540 | 26.200 | 488.600000 | 779.000000 | 0.004767 | 1 |

In [5]:

```
data.tail()
```

Out[5]:

| | date | Temperature | Humidity | Light | CO2 | HumidityRatio | Occupancy |
|---|---|---|---|---|---|---|---|
| **10803** | 10-02-2015 09:29 | 21.05 | 36.0975 | 433.0 | 787.250000 | 0.005579 | 1 |
| **10804** | 10-02-2015 09:29 | 21.05 | 35.9950 | 433.0 | 789.500000 | 0.005563 | 1 |
| **10805** | 10-02-2015 09:30 | 21.10 | 36.0950 | 433.0 | 798.500000 | 0.005596 | 1 |
| **10806** | 10-02-2015 09:32 | 21.10 | 36.2600 | 433.0 | 820.333333 | 0.005621 | 1 |
| **10807** | 10-02-2015 09:33 | 21.10 | 36.2000 | 447.0 | 821.000000 | 0.005612 | 1 |

In [6]:

```
data.shape
```

Out[6]:

```
(10808, 7)
```

In [7]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10808 entries, 0 to 10807
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   date           10808 non-null  object 
 1   Temperature    10808 non-null  float64
 2   Humidity       10808 non-null  float64
 3   Light          10808 non-null  float64
 4   CO2            10808 non-null  float64
 5   HumidityRatio  10808 non-null  float64
 6   Occupancy      10808 non-null  int64  
dtypes: float64(5), int64(1), object(1)
memory usage: 591.2+ KB
```

In [8]:

```
data.describe()
```

Out[8]:

| | Temperature | Humidity | Light | CO2 | HumidityRatio | Occupancy |
|---|---|---|---|---|---|---|
| **count** | 10808.000000 | 10808.000000 | 10808.000000 | 10808.00000 | 10808.000000 | 10808.000000 |
| **mean** | 20.819992 | 25.638407 | 137.694088 | 634.00507 | 0.003903 | 0.249907 |
| **std** | 1.078410 | 4.953792 | 212.175483 | 312.81727 | 0.000803 | 0.432979 |
| **min** | 19.000000 | 16.745000 | 0.000000 | 412.75000 | 0.002674 | 0.000000 |
| **25%** | 20.000000 | 21.390000 | 0.000000 | 441.00000 | 0.003323 | 0.000000 |
| **50%** | 20.700000 | 25.680000 | 0.000000 | 464.00000 | 0.003805 | 0.000000 |
| **75%** | 21.500000 | 28.324167 | 413.541667 | 761.00000 | 0.004372 | 0.000000 |
| **max** | 24.408333 | 39.117500 | 1697.250000 | 2028.50000 | 0.006476 | 1.000000 |

In [9]:

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

Out[9]:

```
date             0
Temperature      0
Humidity         0
Light            0
CO2              0
HumidityRatio    0
Occupancy        0
dtype: int64
```

In [10]:

```
data.describe().columns
```

Out[10]:

```
Index(['Temperature', 'Humidity', 'Light', 'CO2', 'HumidityRatio',
       'Occupancy'],
      dtype='object')
```

In [11]:

```
data['Occupied'] = data['Occupancy'].replace({0: 'No', 1: 'Yes'})
```

In [12]:

```
data.head()
```

Out[12]:

| | date | Temperature | Humidity | Light | CO2 | HumidityRatio | Occupancy | Occupied |
|---|---|---|---|---|---|---|---|---|
| **0** | 02-02-2015 14:19 | 23.7000 | 26.272 | 585.200000 | 749.200000 | 0.004764 | 1 | Yes |
| **1** | 02-02-2015 14:19 | 23.7180 | 26.290 | 578.400000 | 760.400000 | 0.004773 | 1 | Yes |
| **2** | 02-02-2015 14:21 | 23.7300 | 26.230 | 572.666667 | 769.666667 | 0.004765 | 1 | Yes |
| **3** | 02-02-2015 14:22 | 23.7225 | 26.125 | 493.750000 | 774.750000 | 0.004744 | 1 | Yes |
| **4** | 02-02-2015 14:23 | 23.7540 | 26.200 | 488.600000 | 779.000000 | 0.004767 | 1 | Yes |

In [13]:

```
data['Occupied'].value_counts()
```

Out[13]:

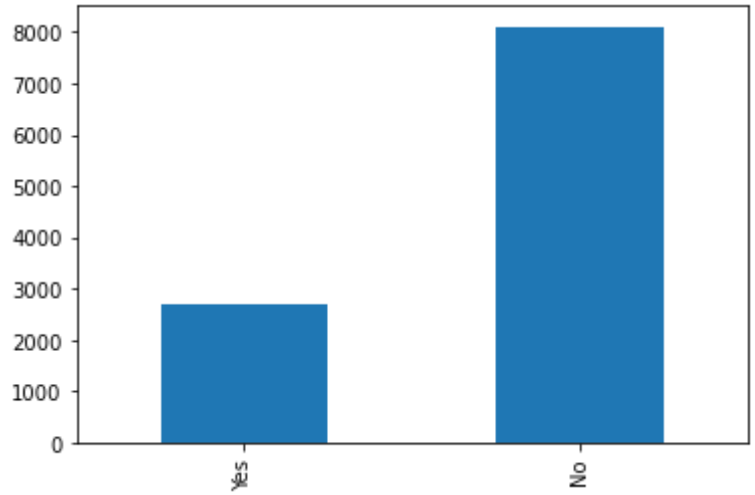
```
No     8107
Yes    2701
Name: Occupied, dtype: int64
```

In [23]:

```
data.Occupied.value_counts().sort_index(ascending=False).plot(kind='bar')
```

Out[23]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x5bc14ea340>
```




In [25]:

```
features=['Temperature', 'Humidity', 'Light', 'CO2', 'HumidityRatio']
data.groupby('Occupied')[features].mean()
```

Out[25]:

| Occupied | Temperature | Humidity | Light | CO2 | HumidityRatio |
|---|---|---|---|---|---|
| **No** | 20.449638 | 25.115111 | 25.595423 | 502.284955 | 0.003724 |
| **Yes** | 21.931603 | 27.209071 | 474.156093 | 1029.360484 | 0.004440 |

In [26]:

```
data_features = data[['Temperature', 'Humidity', 'Light', 'CO2', 'HumidityRatio', 'Occup
```

In [27]:

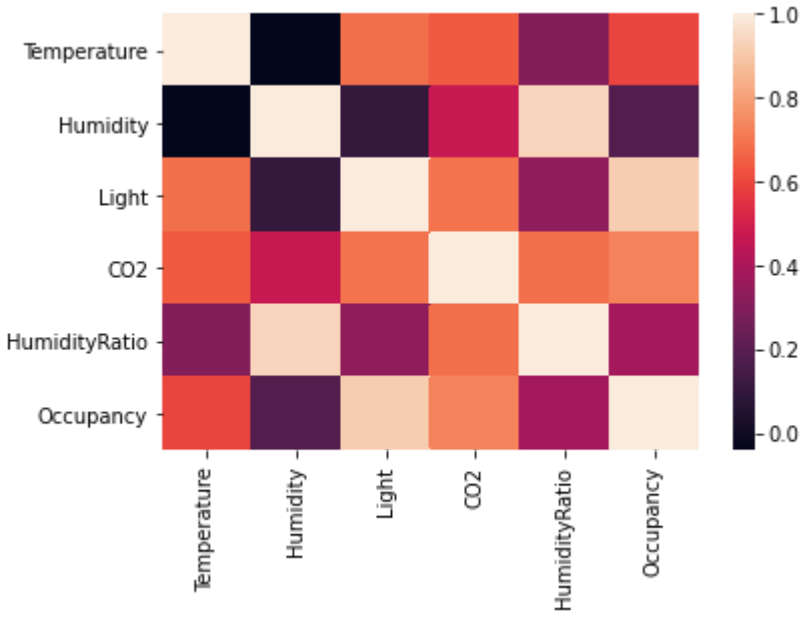
```
print(data_features.corr())
sns.heatmap(data_features.corr())
```

```
               Temperature  Humidity     Light       CO2  HumidityRatio  \
Temperature       1.000000 -0.040270  0.685543  0.640987       0.303167
Humidity         -0.040270  1.000000  0.104641  0.469475       0.938169
Light             0.685543  0.104641  1.000000  0.693588       0.335745
CO2               0.640987  0.469475  0.693588  1.000000       0.686438
HumidityRatio     0.303167  0.938169  0.335745  0.686438       1.000000
Occupancy         0.595005  0.183020  0.915363  0.729540       0.386346

               Occupancy
Temperature     0.595005
Humidity        0.183020
Light           0.915363
CO2             0.729540
HumidityRatio   0.386346
Occupancy       1.000000
```

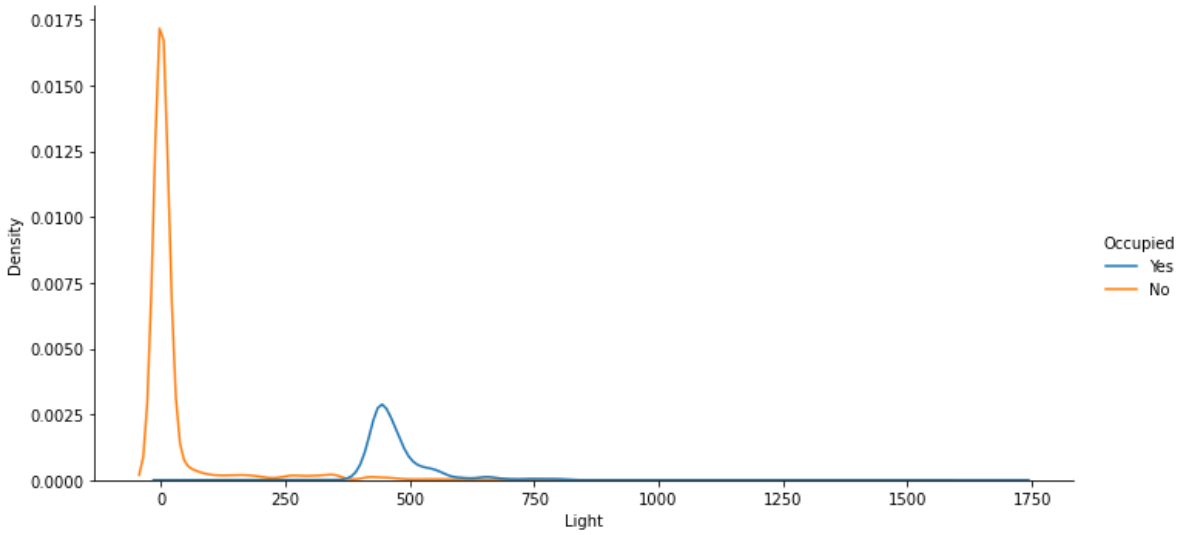
Out[27]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x5bc16af640>
```



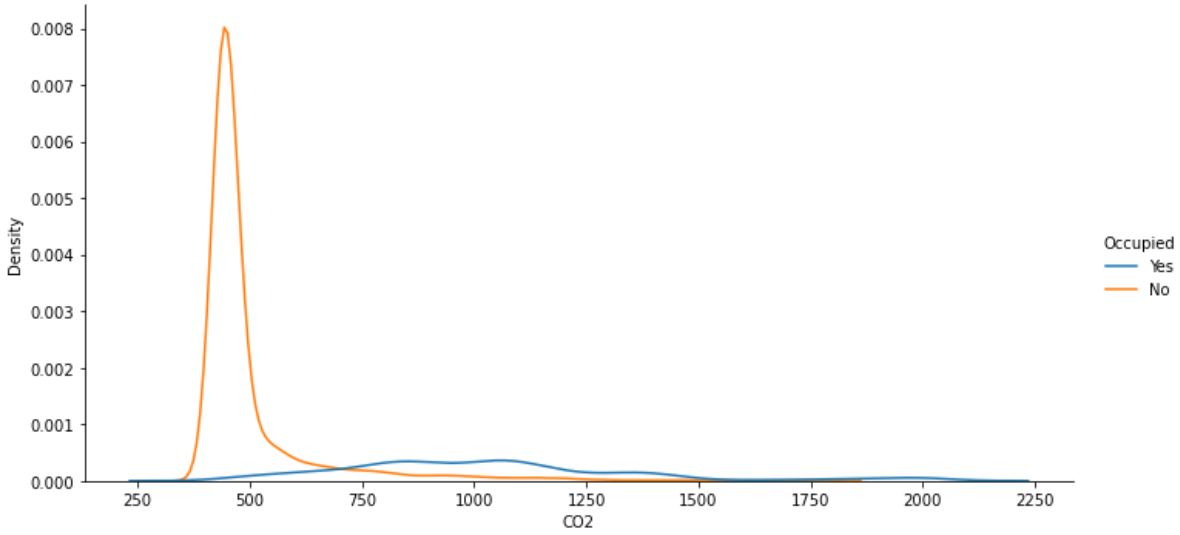
In [28]:

```
sns.displot(kind='kde', x='Light', hue='Occupied', data=data, aspect=2);
```




In [29]:

```
sns.displot(kind='kde', x='CO2', hue='Occupied', data=data, aspect=2);
```

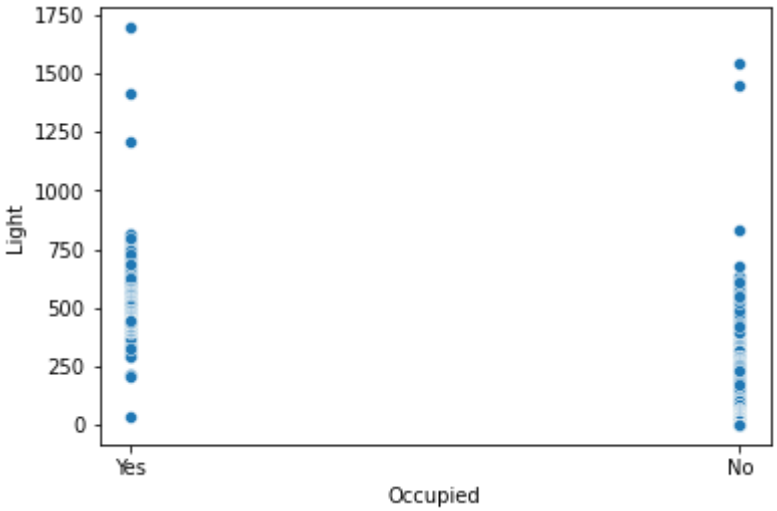


In [31]:

```
sns.scatterplot(y="Light",x="Occupied",data=data)
```

Out[31]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x5ba6cb8250>
```

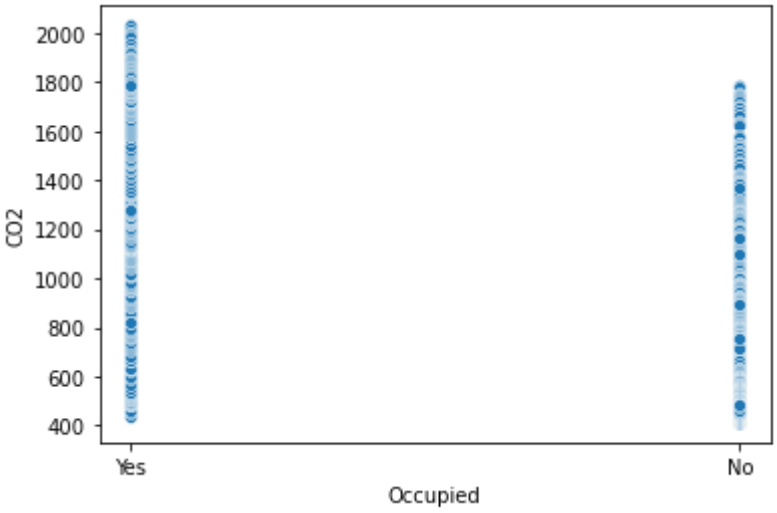



In [32]:

```
sns.scatterplot(y="CO2",x="Occupied",data=data)
```

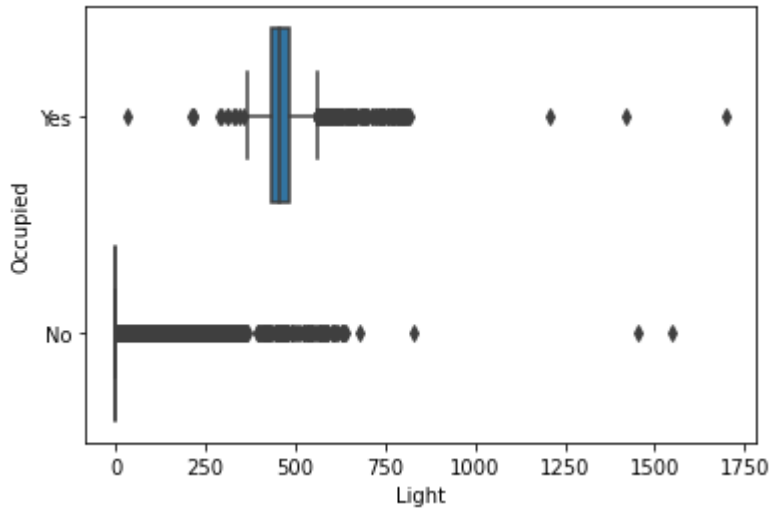
Out[32]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x5b926b2d30>
```



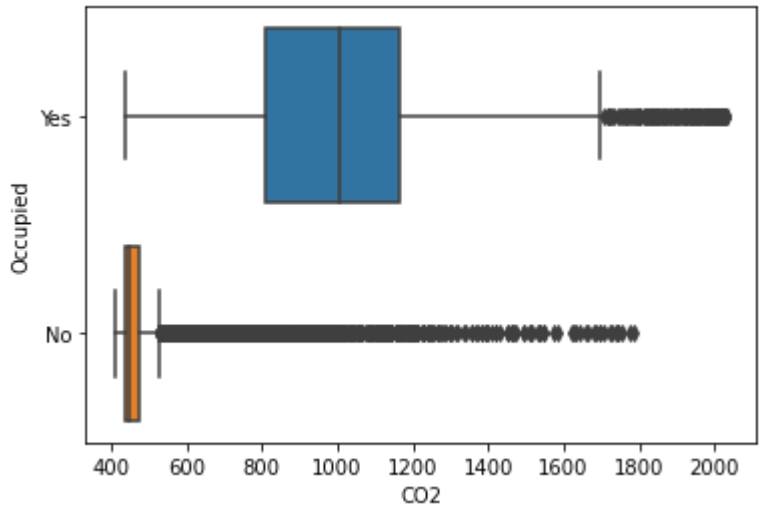
In [34]:

```
sns.boxplot(x='Light', y='Occupied', data=data);
```




In [35]:

```
sns.boxplot(x='CO2', y='Occupied', data=data);
```




In [37]:

```
x = data.drop(['date', 'Occupancy', 'Occupied'], axis = 1)
```

In [38]:

```
y = data.Occupancy
```

In [39]:

```
x.shape
```

Out[39]:

```
(10808, 5)
```

In [40]:

```
y.shape
```

Out[40]:

```
(10808,)
```

In [49]:

```
from sklearn.tree import DecisionTreeClassifier
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)
```

In [50]:

```
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```

Out[50]:

```
DecisionTreeClassifier()
```

In [51]:

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

In [52]:

```
print("Training Accuracy :", model.score(X_train, y_train))
print("Testing Accuracy :", model.score(X_test, y_test))
```

```
Training Accuracy : 1.0
Testing Accuracy : 0.9861239592969473
```

In [53]:

```
from sklearn.ensemble import RandomForestClassifier
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)
```

In [54]:

```
classifier= RandomForestClassifier(n_estimators= 10, criterion="entropy")
classifier.fit(X_train, y_train)
```

Out[54]:

```
RandomForestClassifier(criterion='entropy', n_estimators=10)
```

In [55]:

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

In [56]:

```
print("Training Accuracy :", classifier.score(X_train, y_train))
print("Testing Accuracy :", classifier.score(X_test, y_test))
```

```
Training Accuracy : 0.9987277353689568
Testing Accuracy : 0.9939870490286772
```

In [57]:

```
from sklearn.neighbors import KNeighborsClassifier
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)
```

In [58]:

```
model1= KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
model1.fit(X_train, y_train)
```

Out[58]:

```
KNeighborsClassifier()
```

In [59]:

```
y_pred = model1.predict(X_test)
```

In [60]:

```
print("Training Accuracy :", model1.score(X_train, y_train))
print("Testing Accuracy :", model1.score(X_test, y_test))
```

```
Training Accuracy : 0.9899375433726578
Testing Accuracy : 0.9893617021276596
```

In [61]:

```
from sklearn.naive_bayes import GaussianNB
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)
```

In [62]:

```
model2 = GaussianNB()
model2.fit(X_train, y_train)
```

Out[62]:

```
GaussianNB()
```

In [63]:

```
y_pred = model2.predict(X_test)
```

In [64]:

```
print("Training Accuracy :", model2.score(X_train, y_train))
print("Testing Accuracy :", model2.score(X_test, y_test))
```

```
Training Accuracy : 0.9686560259079343
Testing Accuracy : 0.96577243293247
```