In [1]:
```
import os
import numpy as np
import pandas as pd

from sklearn import model_selection
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import plot_tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import cohen_kappa_score
import seaborn as sns
```

In [2]:
```
import rasterio as rio
import rasterio.plot
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from matplotlib_scalebar.scalebar import ScaleBar
import earthpy.plot as ep
from matplotlib.patches import Patch
```

In [4]:
```
os.chdir('E:/assignment/Random_forest')
```

In [5]:
```
data=pd.read_csv('sample_kolkata -veg.csv')
data
```

Out[5]:

	B2	B3	B4	B5	B6	B7	MNDWI	NDBI	NDVI	LULC	SYM
0	9335	8513	8199	10561	10483	9265	-0.103706	-0.003707	0.125906	Built-up	
1	9473	8620	8467	10446	10632	9463	-0.104509	0.008824	0.104637	Built-up	
2	9354	8313	8154	9530	10483	9590	-0.115450	0.047619	0.077810	Built-up	
3	9458	8556	8830	9389	9895	9221	-0.072571	0.026239	0.030682	Built-up	
4	9697	8681	8535	10033	11246	10498	-0.128720	0.057005	0.080676	Built-up	
...	...	...	...	...	...	...	...	...	...	...	
2495	8652	7693	6811	6163	5321	5188	0.182265	-0.073319	-0.049946	water	
2496	8568	7635	6746	6121	5299	5173	0.180609	-0.071979	-0.048574	water	
2497	8614	7643	6794	6132	5344	5213	0.177023	-0.068665	-0.051215	water	
2498	8750	7892	7019	6298	5446	5285	0.183386	-0.072548	-0.054141	water	
2499	8751	7879	7028	6220	5454	5268	0.181880	-0.065616	-0.060990	water	

2500 rows × 11 columns

In [6]:
```
x=data.drop(['LULC','SYMBOL'],axis=1)
x=x[:].values
y=data.iloc[:,-1]
y=y[:].values
```

In [7]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.30,stratify=y, ran
```

In [8]:
```
RF=RandomForestClassifier(random_state=10,n_jobs=1,max_depth=5,n_estimators=50, oob
RF.fit(x_train,y_train)
```

Out[8]:

RandomForestClassifier

```
RandomForestClassifier(max_depth=5, n_estimators=50, n_jobs=1, oob_score=T
rue,
                       random_state=10)
```

In [9]:
```
y_train_pred_RF=RF.predict(x_train)
y_test_pred_RF=RF.predict(x_test)
print('Train score: ', accuracy_score(y_train_pred_RF,y_train))
print('Test score: ', accuracy_score(y_test_pred_RF,y_test))
```

```
Train score:  0.9994285714285714
Test score:  0.9946666666666667
```

In [10]:
```
CM_RF=confusion_matrix(y_test,y_test_pred_RF)
CM_RF
```

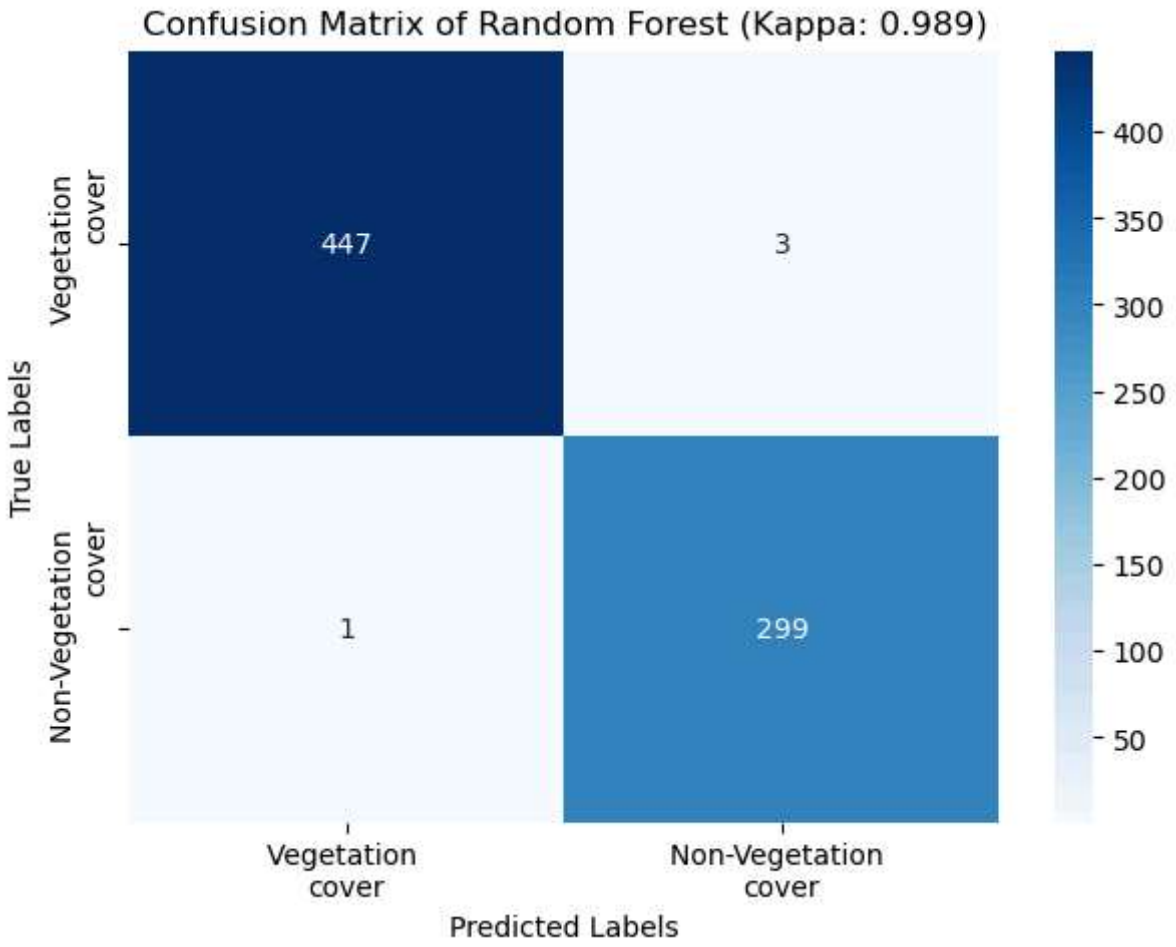
Out[10]:
```
array([[447,   3],
       [  1, 299]], dtype=int64)
```

In [11]:
```
print(classification_report(y_test,y_test_pred_RF))
```

```
              precision    recall  f1-score   support

           0       1.00      0.99      1.00       450
           1       0.99      1.00      0.99       300

    accuracy                           0.99       750
   macro avg       0.99      0.99      0.99       750
weighted avg       0.99      0.99      0.99       750
```

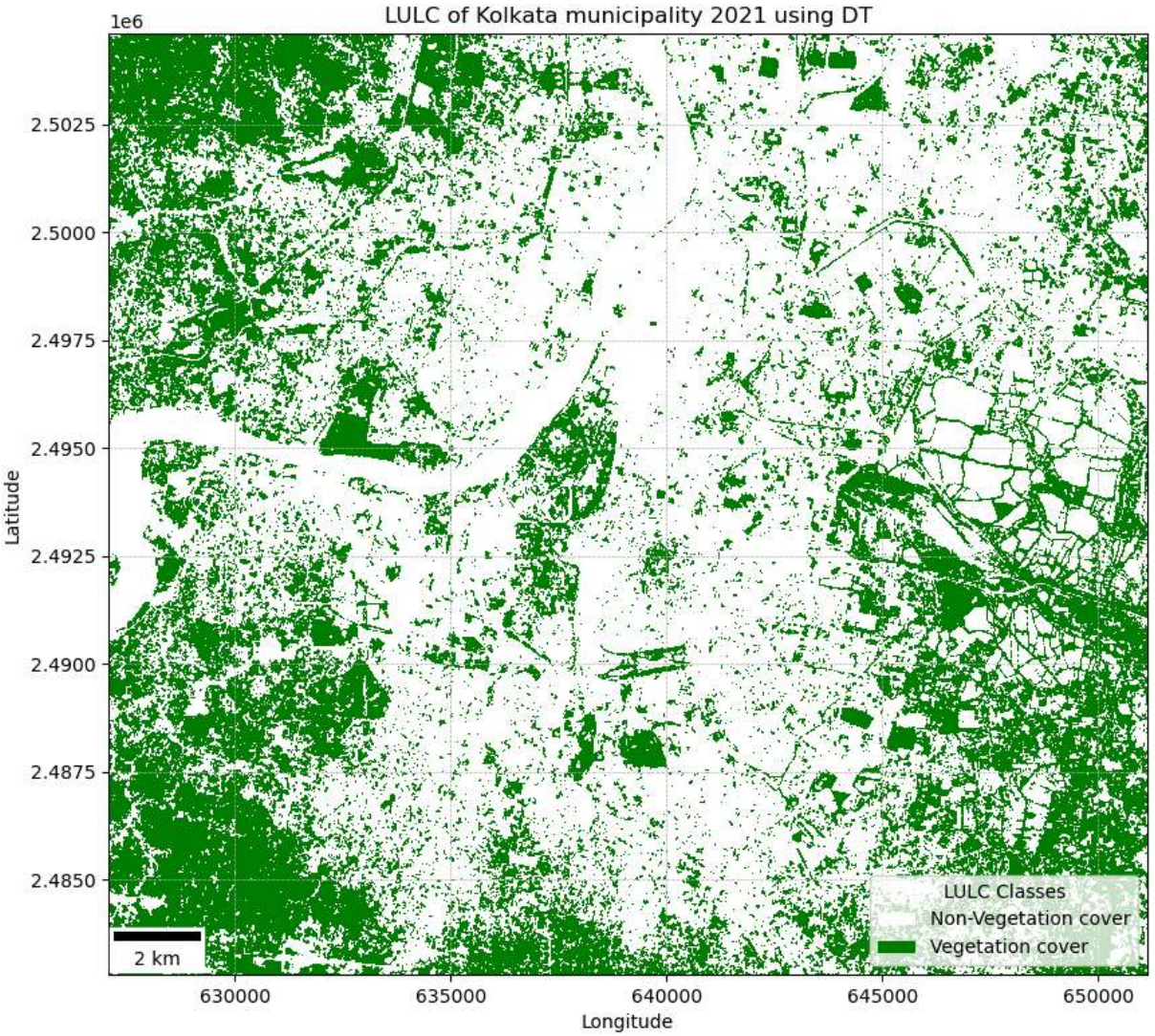
In [12]:
```
class_labels = {
    0: 'Vegetation \ncover',
    1: 'Non-Vegetation \ncover'}

# Compute confusion matrix
CM = confusion_matrix(y_test, y_test_pred_RF)

# Compute Cohen's Kappa coefficient
kappaCTB = cohen_kappa_score(y_test, y_test_pred_RF)
print(f"Cohen's Kappa Score: {kappaCTB:.3f}")

# Replace numeric labels with text labels
tick_labels = [class_labels[i] for i in sorted(class_labels.keys())]

# Plot confusion matrix using seaborn
plt.figure(figsize=(7, 5))
sns.heatmap(CM, annot=True, fmt='d', cmap='Blues', xticklabels=tick_labels, yticklabels=tick_labels)
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")
plt.title(f"Confusion Matrix of Random Forest (Kappa: {kappaCTB:.3f})")
plt.show()
```

```
Cohen's Kappa Score: 0.989
```



In [13]:
```
with rio.open('compkol.tif') as src:
    img = src.read()
    transform = src.transform
    crs = src.crs
    extent = rasterio.plot.plotting_extent(src)


Image = img.transpose([1, 2, 0])
new_Image = np.nan_to_num(Image)
Bands = new_Image.reshape(Image.shape[0]*Image.shape[1], Image.shape[2])


Pred_RF_Img = RF.predict(Bands)
Pred_RF_Img = Pred_RF_Img.reshape(Image.shape[0], Image.shape[1])


cmap = ListedColormap(['White', 'Green'])
class_labels = {
    0: "Non-Vegetation cover",
    1: "Vegetation cover"}


fig, ax = plt.subplots(figsize=(10, 12))

im = ax.imshow(Pred_RF_Img, cmap=cmap, extent=extent, origin="upper")
```

```
scalebar = ScaleBar(dx=1, units="m", location='lower left')
ax.add_artist(scalebar)


ax.set_xlabel("Longitude")
ax.set_ylabel("Latitude")
ax.set_title("LULC of Kolkata municipality 2021 using DT")
ax.grid(True, linestyle="--", linewidth=0.5)

legend_elements = [Patch(facecolor=cmap(i), label=label)
                   for i, label in class_labels.items()]
ax.legend(handles=legend_elements,
          loc='lower right',
          title="LULC Classes")

plt.show()
```



In [ ]:

In [ ]: