Project final

May 29, 2022

0.1 Feature Engineering

0.1.1 Data preparation

[125]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf
import keras as ks
import seaborn as sn
from sklearn.svm import SVC
from scipy.io import loadmat
from pandas import read_csv
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score,␣
↪classification_report
from sklearn.decomposition import PCA
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler
from keras.layers import Dense, Conv1D, Dropout, Flatten, MaxPooling1D, LSTM,␣
↪Embedding
```

[2]:
```
pine = loadmat('Indian_pines.mat')['indian_pines']
pine_gt = loadmat('Indian_pines_gt.mat')['indian_pines_gt']
```

[3]:
```
pine
```

[3]:
```
array([[[3172, 4142, 4506, …, 1020, 1020, 1005],
        [2580, 4266, 4502, …, 1029, 1020, 1000],
        [3687, 4266, 4421, …, 1030, 1016, 1009],
        …,
        [2570, 3890, 4320, …, 1021, 1015, 1025],
        [3170, 4130, 4320, …, 1024, 1020, 1011],
        [3172, 3890, 4316, …, 1034, 1016, 1015]],

       [[2576, 4388, 4334, …, 1030, 1006, 1015],
```

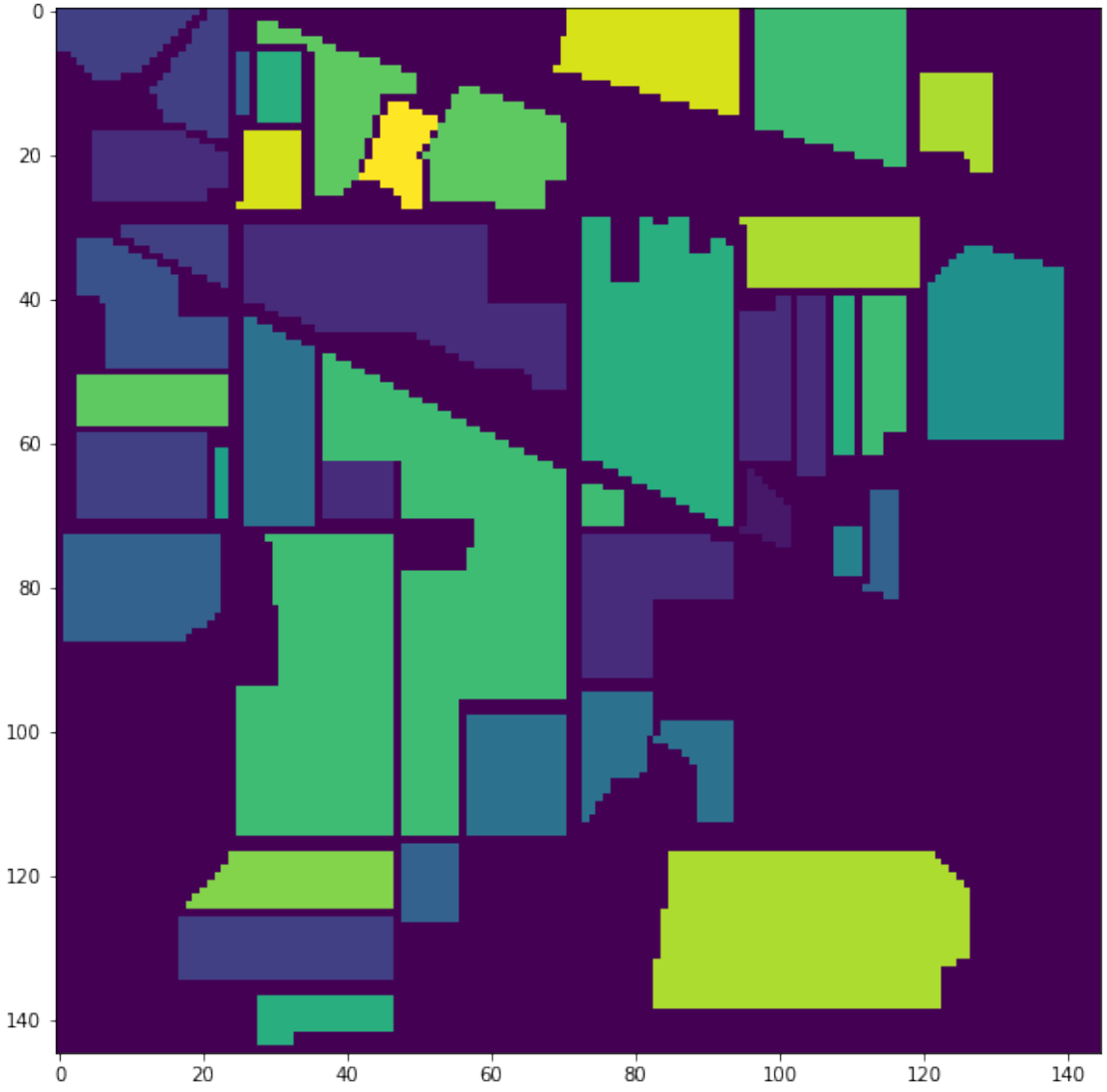
```
 [2747, 4264, 4592, …, 1039, 1015, 1020],
 [2750, 4268, 4423, …, 1026, 1015, 1020],
 …,
 [3859, 4512, 4605, …, 1035, 1015,  996],
 [3686, 4264, 4690, …, 1012, 1020, 1014],
 [2744, 4268, 4597, …, 1019, 1016, 1010]],

[[2744, 4146, 4416, …, 1029, 1025, 1010],
 [2576, 4389, 4416, …, 1021, 1011, 1000],
 [2744, 4273, 4420, …, 1033, 1010, 1014],
 …,
 [2570, 4266, 4509, …, 1025, 1010, 1005],
 [2576, 4262, 4496, …, 1029, 1020, 1005],
 [2742, 4142, 4230, …, 1025, 1011, 1010]],

…,

[[3324, 3728, 4002, …, 1004, 1004, 1000],
 [2983, 3604, 3829, …, 1013, 1008,  995],
 [2988, 3612, 3913, …, 1001, 1004, 1003],
 …,
 [2564, 4115, 4103, …, 1005, 1013, 1009],
 [2730, 4111, 4103, …, 1013, 1004, 1004],
 [3156, 3991, 4103, …, 1014, 1000, 1009]],

[[3161, 3731, 3834, …, 1000, 1000, 1009],
 [2727, 3742, 4011, …,  991, 1003, 1000],
 [2988, 4114, 4011, …, 1008, 1013, 1004],
 …,
 [3156, 3858, 4016, …, 1004, 1003, 1009],
 [3159, 3858, 4100, …, 1000, 1000,  995],
 [2561, 3866, 4003, …, 1008, 1000, 1003]],

[[2979, 3728, 3732, …, 1004, 1000,  995],
 [2977, 3728, 3741, …, 1009,  990, 1013],
 [2814, 3728, 3914, …, 1009, 1003, 1019],
 …,
 [3153, 3864, 4282, …, 1008, 1000, 1009],
 [3155, 4104, 4106, …, 1005, 1003, 1004],
 [3323, 3860, 4197, …, 1004, 1000, 1000]]], dtype=uint16)
```

[4]:
```
pine_gt
```

[4]:
```
array([[3, 3, 3, …, 0, 0, 0],
       [3, 3, 3, …, 0, 0, 0],
       [3, 3, 3, …, 0, 0, 0],
       …,
```

```
       [0, 0, 0, …, 0, 0, 0],
       [0, 0, 0, …, 0, 0, 0],
       [0, 0, 0, …, 0, 0, 0]], dtype=uint8)
```

[8]:
```
fig, (ax1) = plt.subplots(1, figsize=(20,10))
img= ax1.imshow(pine_gt) #.astype(int),aspect='auto',cmap=plt.cm.bone)
```



[5]:
```
pine_gt[:,144] # 0 labellar boş alanı temsil ediyor.
```

[5]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=uint8)
```

[14]:
```
np.unique(pine_gt)
```

[14]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16],
      dtype=uint8)
```

[6]:
```
pine_gt.shape
```

[6]:
```
(145, 145)
```

[35]:
```
pine_220 = pine.reshape((145*145,220))
y_1 = pine_gt.reshape(145*145)

labeler = LabelBinarizer()
y_17 = labeler.fit_transform(y_1)
yy_17=y_17.astype(np.float32)

print(yy_17.shape)
print(pine_220.shape)
```

```
(21025, 17)
(21025, 220)
```

0.1.2 Adding point coordinates to data

[24]:
```
Pine_222 =[]
for i in range (145):
    for j in range (145):
        x = pine[i,j].tolist()
        x.extend([i,j])
        # k = pine_gt[i,j]
        # x.extend([i,j,k])
        Pine_222.append(x)
```

[25]:
```
pine_222 = np.asarray(Pine_222).astype(np.float32)
pine_222.shape
```

[25]:
```
(21025, 222)
```

0.1.3 Desizing the data with PCA

[27]:
```
model_pca = PCA(n_components=3)
pine_3 = model_pca.fit_transform(pine_220)
pine_3.shape
```

[27]: (21025, 3)

0.1.4 Add locations to Pca applied data

[32]:
```
Pine_3 = pine_3.reshape(145,145,3)

Pine_5 =[]
for i in range (145):
    for j in range (145):
        x = Pine_3[i,j].tolist()
        x.extend([i,j])
        # k = pine_gt[i,j]
        # x.extend([i,j,k])
        Pine_5.append(x)
```

[33]:
```
pine_5 = np.asarray(Pine_5).astype(np.float32)
pine_5.shape
```

[33]: (21025, 5)

[36]:
```
plants = ['Free space', 'Alfalfa','Corn-notill',␣
 ↪'Corn-mintill','Corn','Grass-pasture','Grass-trees','Grass-pasture-mowed','Hay-windrowed','
'Soybean-clean', 'Wheat','Woods','Buildings Grass Trees Drives','Stone Steel␣
 ↪Towers']
```

0.2 Classical Classification Machine Learning Methods

0.2.1 Splitting the data

[39]:
```
pine_3_Train,  pine_3_Test, pine_3_Y_Train, pine_3_Y_Test =␣
 ↪train_test_split(pine_3, y_1, test_size = 0.25, random_state = 0)
```

[40]:
```
pine_5_Train,  pine_5_Test, pine_5_Y_Train, pine_5_Y_Test =␣
 ↪train_test_split(pine_5, y_1, test_size = 0.25, random_state = 0)
```

[41]:
```
pine_220_Train,  pine_220_Test, pine_220_Y_Train, pine_220_Y_Test =␣
 ↪train_test_split(pine_220, y_1, test_size = 0.25, random_state = 0)
```

[42]:
```
pine_222_Train,  pine_222_Test, pine_222_Y_Train, pine_222_Y_Test =␣
 ↪train_test_split(pine_222, y_1, test_size = 0.25, random_state = 0)
```

0.2.2 1. SVM

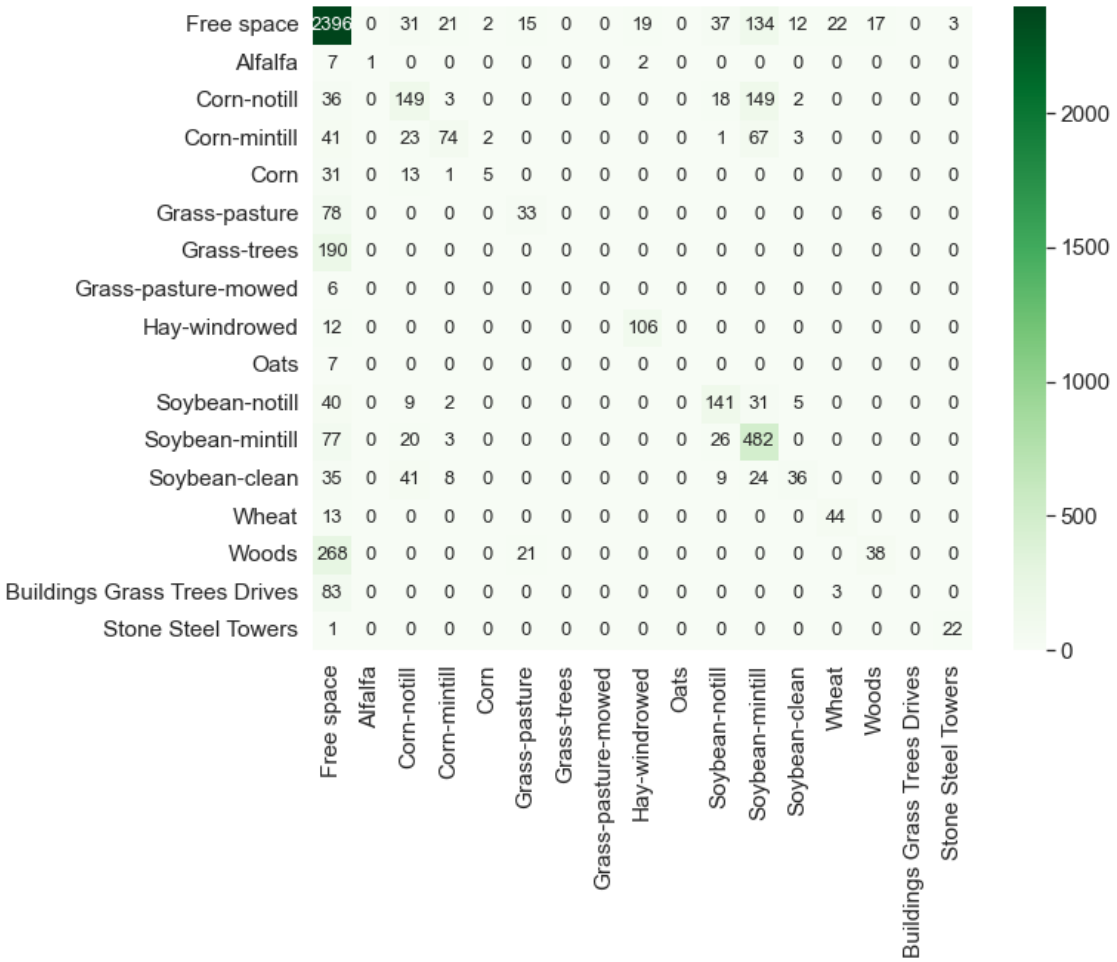
[92]:
```
def SVM(x_train,x_test,y_train,y_test):
    svm_model =  SVC(C = 100, kernel = 'rbf', cache_size = 10*1024)
    svm_model.fit(x_train, y_train)
    y_pred = svm_model.predict(x_test)
    svm_acc = accuracy_score(y_test,y_pred)
```

```
    cm_svm = confusion_matrix(y_test, y_pred)

    return {'SVM_Acc': svm_acc, 'SVM_Pred': y_pred, 'SVM_cm': cm_svm}
```

[93]:
```
def CM(cm):
    df_cm = pd.DataFrame(cm[0:17, 0:17], columns= plants[0:17], index= plants[0:17])
    plt.figure(figsize = (10,8))
    sn.set(font_scale=1.4) #for label size
    sn.heatmap(df_cm, cmap="Greens", annot=True,annot_kws={"size": 13}, fmt='d')
    plot = plt.savefig('cmap.png', dpi=300)
    return plot
```

[96]:
```
SVM_Outputs = SVM(pine_3_Train,  pine_3_Test, pine_3_Y_Train, pine_3_Y_Test)

print(f'SVM_Acc = {SVM_Outputs.get("SVM_Acc")}')
CM(SVM_Outputs.get("SVM_cm"))
```

```
SVM_Acc = 0.6709149705155031
```

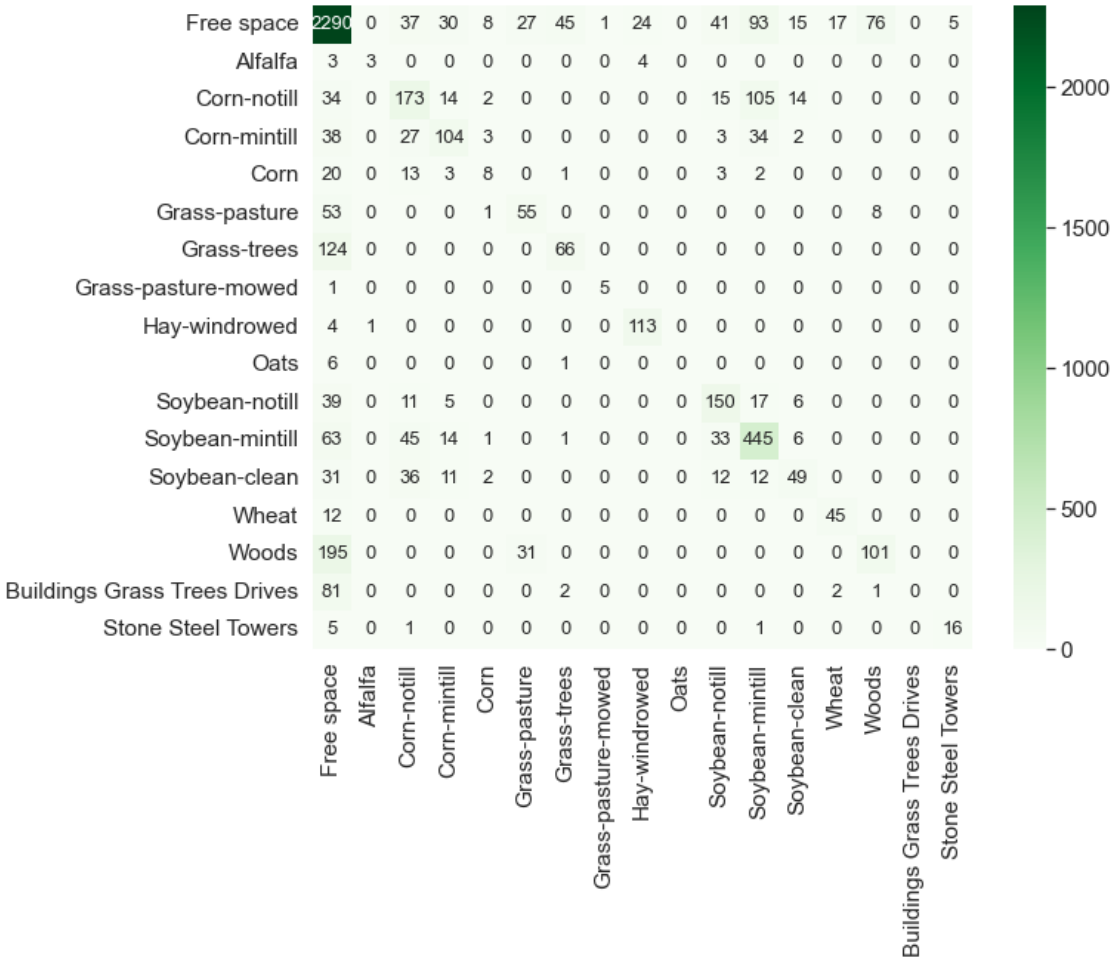


0.2.3 2. KNN

[91]:
```
def KNN(x_train,x_test,y_train,y_test,n):
    scaler = StandardScaler()
    scaler.fit(x_train)
    knn_train = scaler.transform(x_train)
    knn_test = scaler.transform(x_test)
    classifier_knn = KNeighborsClassifier(n_neighbors=n)
    classifier_knn.fit(knn_train, y_train)
    knn_pred = classifier_knn.predict(knn_test)
    cm_knn = confusion_matrix(y_test, knn_pred)
    knn_acc = accuracy_score(y_test,knn_pred)
    return {'KNN_Acc': knn_acc, 'KNN_Pred': knn_pred, 'KNN_cm': cm_knn}
```

[90]:
```
KNN_Outputs = KNN(pine_3_Train,  pine_3_Test, pine_3_Y_Train, pine_3_Y_Test,14)

print(f'KNN_Acc = {KNN_Outputs.get("KNN_Acc")}')
CM(KNN_Outputs.get("KNN_cm"))
```

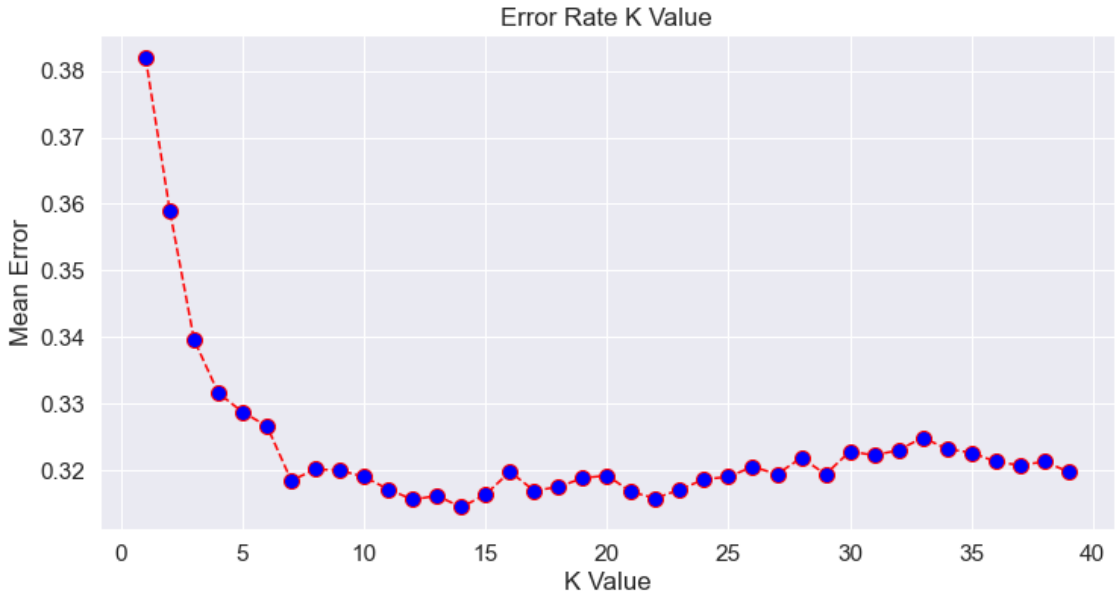
```
KNN_Acc = 0.6891763363134867
```



[100]:
```
def knn_error(x_train,x_test,y_train,y_test):
    KNN_error = []

    for i in range(1, 40):
        model_error = KNeighborsClassifier(n_neighbors=i)
        model_error.fit(x_train, y_train)
        pred_i = model_error.predict(x_test)
        KNN_error.append(np.mean(pred_i != y_test))

    plt.figure(figsize=(12, 6))
    plt.plot(range(1, 40), KNN_error, color='red', linestyle='dashed',␣
↪marker='o',
             markerfacecolor='blue', markersize=10)
    plt.title('Error Rate K Value')
    plt.xlabel('K Value')
    plot_error = plt.ylabel('Mean Error')
    return plot_error
```

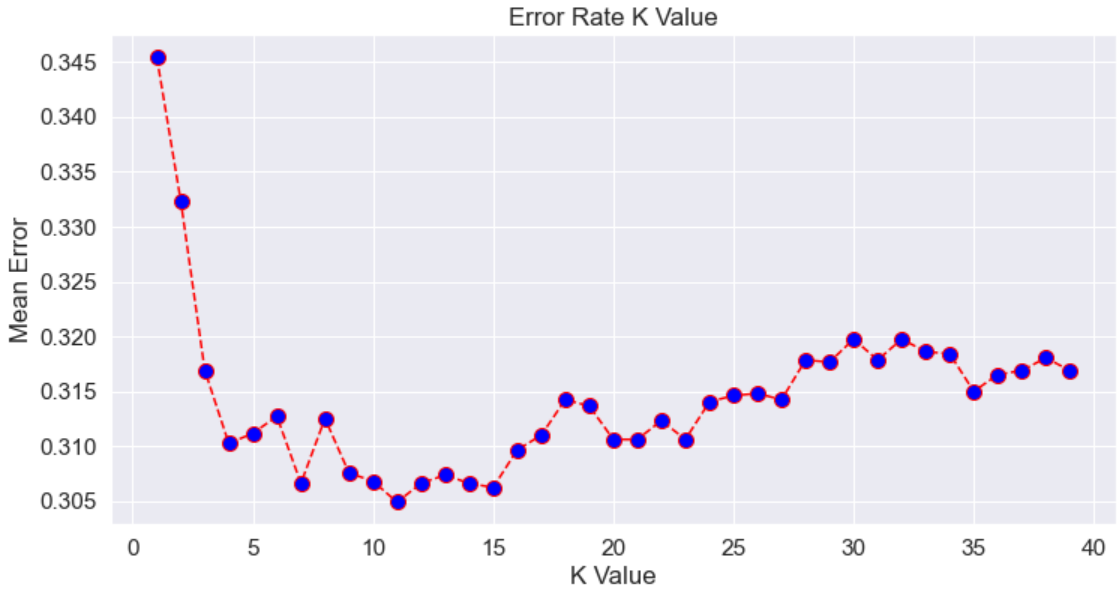
[101]:
```
knn_error(pine_3_Train,  pine_3_Test, pine_3_Y_Train, pine_3_Y_Test)
```

[101]: Text(0, 0.5, 'Mean Error')



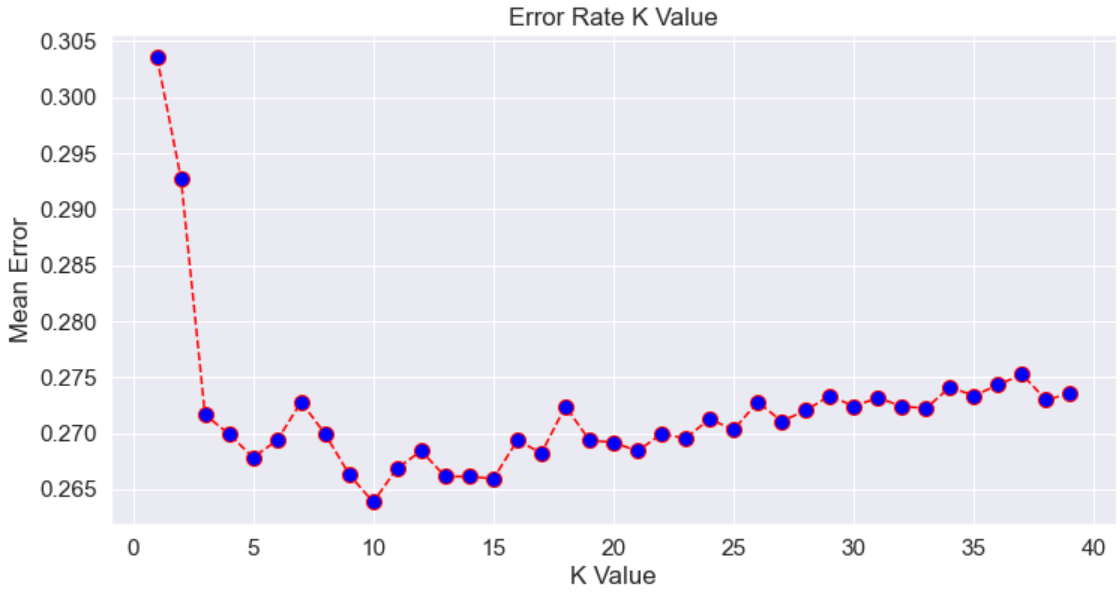
[161]:
```
knn_error(pine_5_Train,  pine_5_Test, pine_5_Y_Train, pine_5_Y_Test)
```

[161]: Text(0, 0.5, 'Mean Error')



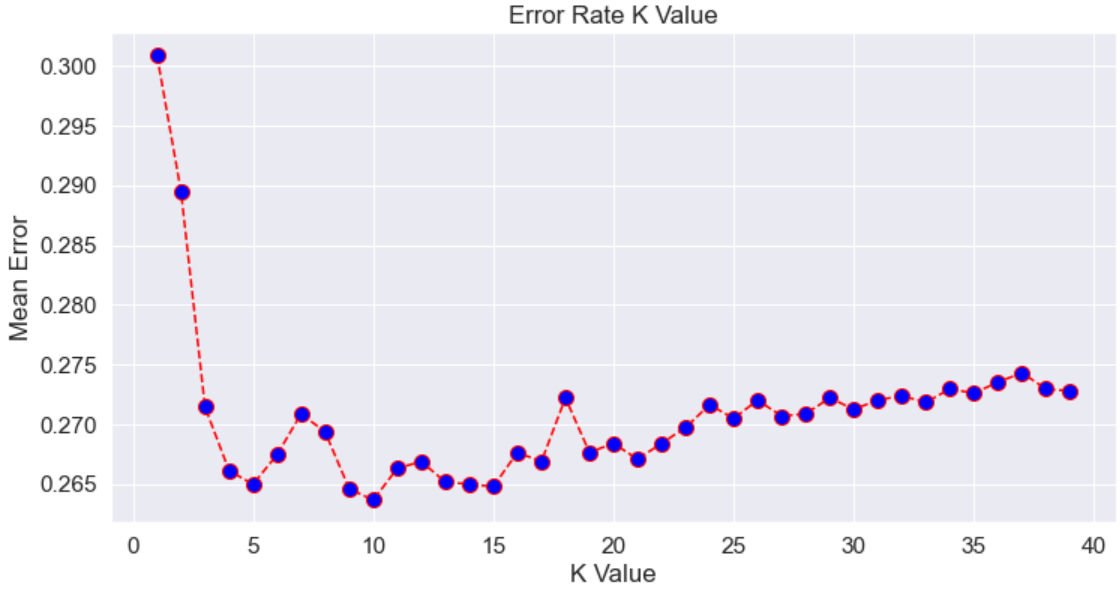
[162]:
```
knn_error(pine_220_Train,  pine_220_Test, pine_220_Y_Train, pine_220_Y_Test)
```

[162]: Text(0, 0.5, 'Mean Error')



[164]:
```
knn_error(pine_222_Train,  pine_222_Test, pine_222_Y_Train, pine_222_Y_Test)
```

[164]: Text(0, 0.5, 'Mean Error')

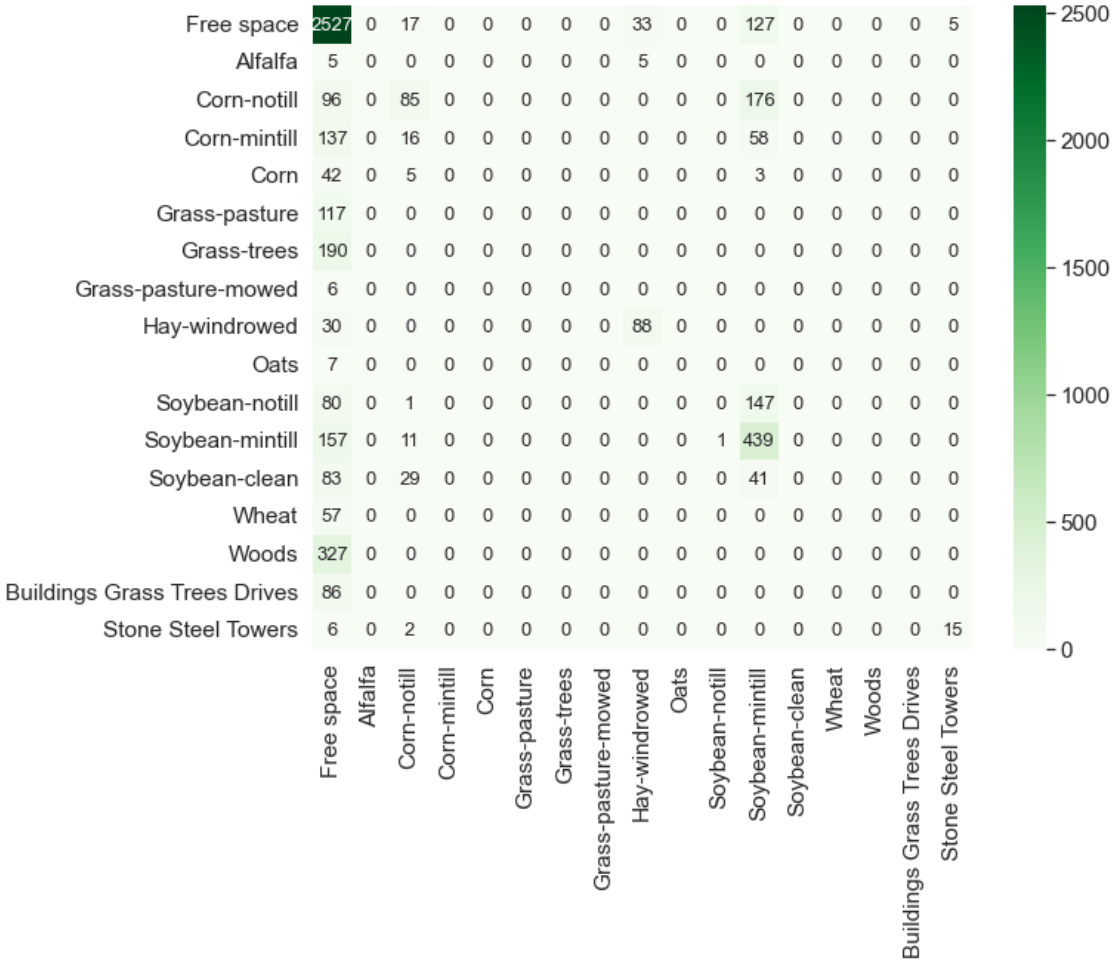


0.2.4 3. Logistic Regression

[115]:
```
def LR(x_train,x_test,y_train,y_test, iteration):
    lr_model =  LogisticRegression(solver='liblinear', penalty='l2',C= 1.0,␣
 ↪random_state=0, max_iter = iteration)
    lr_model.fit(x_train,y_train)
    lr_model = LogisticRegression(solver='liblinear', random_state=0, max_iter␣
 ↪= iteration).fit(x_train, y_train)
    lr_proba = lr_model.predict_proba(x_test)
    y_pred = lr_model.predict(x_test)
    cm_lr = confusion_matrix(y_test, lr_model.predict(x_test))
    lr_acc = accuracy_score(y_test,y_pred)

    return {'LR_Acc': lr_acc, 'LR_Pred': y_pred, 'LR_cm': cm_lr, 'LR_proba':␣
 ↪lr_proba}
```

[116]:
```
LR_Outputs = LR(pine_3_Train,  pine_3_Test, pine_3_Y_Train, pine_3_Y_Test,100)

print(f'LR_Acc = {LR_Outputs.get("LR_Acc")}')
CM(LR_Outputs.get("LR_cm"))
```

```
LR_Acc = 0.5999619554879209
```



0.3 Neural Network Models

0.3.1 Splitting the data

[118]:
```
sc = StandardScaler()

pine_3_sc = sc.fit_transform(pine_3)
pine_5_sc = sc.fit_transform(pine_5)
pine_220_sc = sc.fit_transform(pine_220)
pine_222_sc = sc.fit_transform(pine_222)

pine_3_train, pine_3_test, pine_3_y_train, pine_3_y_test =␣
 ↪train_test_split(pine_3_sc, y_17)
pine_5_train, pine_5_test, pine_5_y_train, pine_5_y_test =␣
 ↪train_test_split(pine_5_sc, y_17)
pine_220_train, pine_220_test, pine_220_y_train, pine_220_y_test =␣
 ↪train_test_split(pine_220_sc, y_17)
```

```
pine_222_train, pine_222_test, pine_222_y_train, pine_222_y_test =␣
↪train_test_split(pine_222_sc, y_17)
```

0.3.2 Standard Neural Network

[121]:
```
def NN(x_train,x_test,y_train,y_test, dimension):
    nn_model = ks.models.Sequential()
    nn_model.add(ks.layers.Dense(64, activation='relu', input_dim=dimension))
    nn_model.add(ks.layers.Dense(17,activation='sigmoid'))
    nn_model.compile(optimizer='adam',
                  loss='binary_crossentropy',
                  metrics=['binary_accuracy'])
    nn_model.fit(x_train,y_train,epochs=50,batch_size=100)
    y_pred =  nn_model.predict(x_test)
    yy_pred = np.argmax(y_pred,axis=1)
    yy_test = np.argmax(y_test,axis=1)
    nn_acc = accuracy_score(yy_test,yy_pred)
    cm_nn = confusion_matrix(yy_test,yy_pred)
    return {'NN_Acc': nn_acc, 'NN_Pred': y_pred, 'NN_cm': cm_nn}
```

[124]:
```
NN_Outputs = NN(pine_3_train, pine_3_test, pine_3_y_train, pine_3_y_test, 3)

print(f'NN_Acc = {NN_Outputs.get("NN_Acc")}')
```

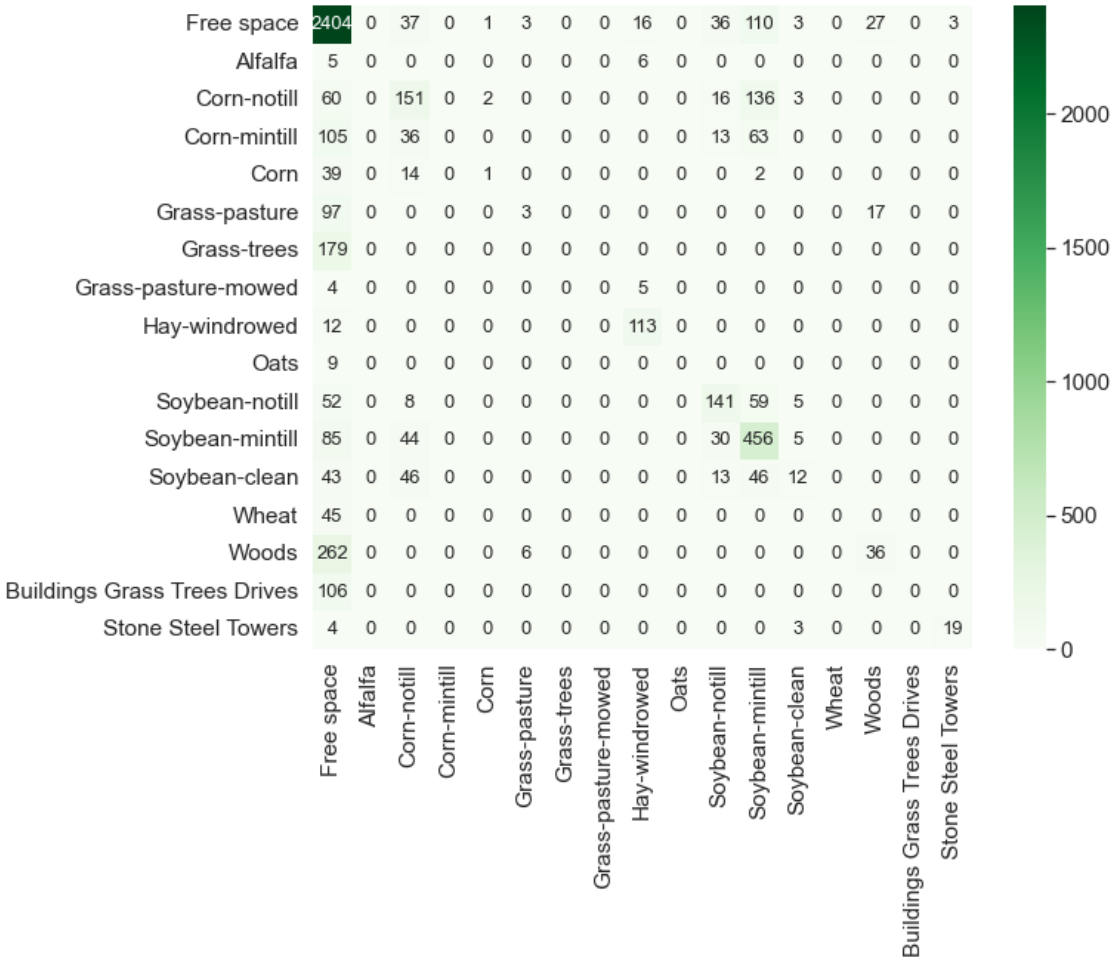
```
Epoch 1/50
158/158 [==============================] - 0s 1ms/step - loss: 0.4111 -
binary_accuracy: 0.8988
Epoch 2/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1576 -
binary_accuracy: 0.9550
Epoch 3/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1314 -
binary_accuracy: 0.9559
Epoch 4/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1222 -
binary_accuracy: 0.9563
Epoch 5/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1167 -
binary_accuracy: 0.9566
Epoch 6/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1130 -
binary_accuracy: 0.9571
Epoch 7/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1103 -
binary_accuracy: 0.9577
Epoch 8/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1083 -
binary_accuracy: 0.9582
```

```
Epoch 9/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1068 -
binary_accuracy: 0.9587
Epoch 10/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1055 -
binary_accuracy: 0.9593
Epoch 11/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1045 -
binary_accuracy: 0.9594
Epoch 12/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1037 -
binary_accuracy: 0.9596
Epoch 13/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1029 -
binary_accuracy: 0.9598
Epoch 14/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1023 -
binary_accuracy: 0.9599
Epoch 15/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1017 -
binary_accuracy: 0.9600
Epoch 16/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1011 -
binary_accuracy: 0.9600
Epoch 17/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1006 -
binary_accuracy: 0.9603
Epoch 18/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1002 -
binary_accuracy: 0.9603
Epoch 19/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0997 -
binary_accuracy: 0.9603
Epoch 20/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0994 -
binary_accuracy: 0.9604
Epoch 21/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0990 -
binary_accuracy: 0.9605
Epoch 22/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0987 -
binary_accuracy: 0.9606
Epoch 23/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0984 -
binary_accuracy: 0.9607
Epoch 24/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0981 -
binary_accuracy: 0.9606
```

```
Epoch 25/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0977 -
binary_accuracy: 0.9608
Epoch 26/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0975 -
binary_accuracy: 0.9610
Epoch 27/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0973 -
binary_accuracy: 0.9608
Epoch 28/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0970 -
binary_accuracy: 0.9609
Epoch 29/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0968 -
binary_accuracy: 0.9609
Epoch 30/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0966 -
binary_accuracy: 0.9611
Epoch 31/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0964 -
binary_accuracy: 0.9611
Epoch 32/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0961 -
binary_accuracy: 0.9612
Epoch 33/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0959 -
binary_accuracy: 0.9613
Epoch 34/50
158/158 [==============================] - 0s 2ms/step - loss: 0.0957 -
binary_accuracy: 0.9613
Epoch 35/50
158/158 [==============================] - 0s 943us/step - loss: 0.0955 -
binary_accuracy: 0.9614
Epoch 36/50
158/158 [==============================] - 0s 805us/step - loss: 0.0953 -
binary_accuracy: 0.9613
Epoch 37/50
158/158 [==============================] - 0s 858us/step - loss: 0.0952 -
binary_accuracy: 0.9613
Epoch 38/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0950 -
binary_accuracy: 0.9617
Epoch 39/50
158/158 [==============================] - 0s 840us/step - loss: 0.0948 -
binary_accuracy: 0.9615
Epoch 40/50
158/158 [==============================] - 0s 897us/step - loss: 0.0946 -
binary_accuracy: 0.9615
```

```
Epoch 41/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0945 -
binary_accuracy: 0.9618
Epoch 42/50
158/158 [==============================] - 0s 847us/step - loss: 0.0943 -
binary_accuracy: 0.9618
Epoch 43/50
158/158 [==============================] - 0s 868us/step - loss: 0.0942 -
binary_accuracy: 0.9616
Epoch 44/50
158/158 [==============================] - 0s 886us/step - loss: 0.0940 -
binary_accuracy: 0.9617
Epoch 45/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0938 -
binary_accuracy: 0.9618
Epoch 46/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0937 -
binary_accuracy: 0.9618
Epoch 47/50
158/158 [==============================] - 0s 931us/step - loss: 0.0935 -
binary_accuracy: 0.9617
Epoch 48/50
158/158 [==============================] - 0s 903us/step - loss: 0.0934 -
binary_accuracy: 0.9618
Epoch 49/50
158/158 [==============================] - 0s 929us/step - loss: 0.0932 -
binary_accuracy: 0.9618
Epoch 50/50
158/158 [==============================] - 0s 920us/step - loss: 0.0931 -
binary_accuracy: 0.9618
NN_Acc = 0.6372455773254708
```

[123]:
```
CM(NN_Outputs.get("NN_cm"))
```



0.3.3 Convolutional Neural Network

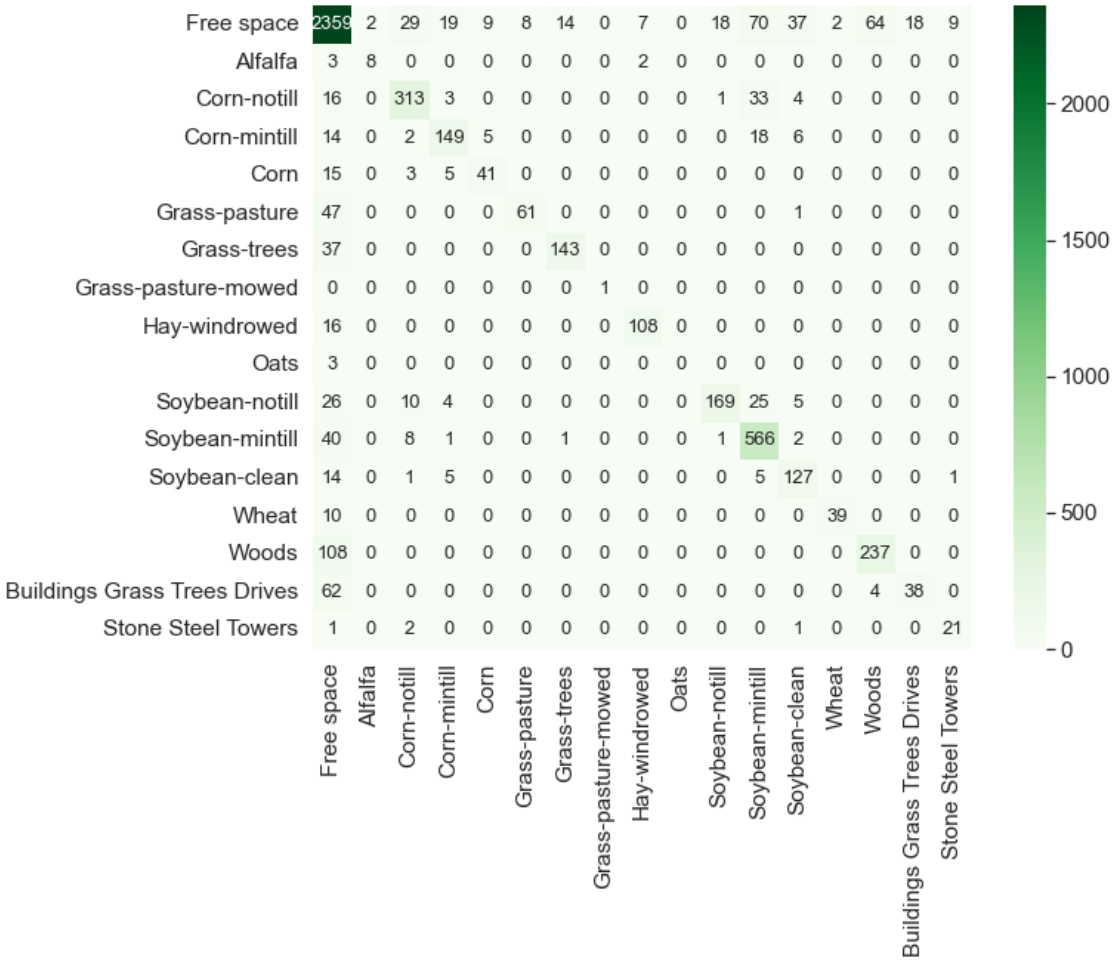
[126]:
```
def CNN(x_train,x_test,y_train,y_test, m,n):
    model_cnn = ks.models.Sequential()
    model_cnn.add(Conv1D(64, 3, padding="same",␣
 ↪activation='relu',input_shape=(m,n)))
    model_cnn.add(MaxPooling1D(pool_size=2))
    model_cnn.add(Flatten())
    model_cnn.add(Dense(128, activation = 'relu'))
    model_cnn.add(Dense(17, activation = 'sigmoid'))
    model_cnn.compile(optimizer='adam',
                  loss='binary_crossentropy',
                  metrics=['binary_accuracy'])
    model_cnn.fit(x_train, y_train, batch_size=64, epochs=10)
    y_pred = model_cnn.predict(x_test)
    ycc_pred = np.argmax(y_pred,axis=1)
    ycc_test = np.argmax(y_test,axis=1)
```

```
    cm_cnn = confusion_matrix(ycc_test,ycc_pred)
    cnn_acc = accuracy_score(ycc_test,ycc_pred)
    return {'CNN_Acc': cnn_acc, 'CNN_cm': cm_cnn}
```

[128]:
```
CNN_Outputs = CNN(pine_222_train, pine_222_test, pine_222_y_train,␣
↪pine_222_y_test, 222,1)
print(f'CNN_Acc = {CNN_Outputs.get("CNN_Acc")}')
```

```
Epoch 1/10
247/247 [==============================] - 4s 16ms/step - loss: 0.1123 -
binary_accuracy: 0.9567
Epoch 2/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0869 -
binary_accuracy: 0.9641
Epoch 3/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0749 -
binary_accuracy: 0.9686
Epoch 4/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0679 -
binary_accuracy: 0.9716
Epoch 5/10
247/247 [==============================] - 4s 17ms/step - loss: 0.0625 -
binary_accuracy: 0.9740
Epoch 6/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0596 -
binary_accuracy: 0.9755
Epoch 7/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0548 -
binary_accuracy: 0.9775
Epoch 8/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0522 -
binary_accuracy: 0.9786
Epoch 9/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0493 -
binary_accuracy: 0.9801
Epoch 10/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0461 -
binary_accuracy: 0.9814
CNN_Acc = 0.8331748145330036
```

[129]:
```
CM(CNN_Outputs.get("CNN_cm"))
```



0.4 ANALYSIS

0.4.1 Accuracy Scores

SVM Scores

[131]:
```
SVM_Outputs_3 = SVM(pine_3_Train,  pine_3_Test, pine_3_Y_Train, pine_3_Y_Test)
SVM_3 = SVM_Outputs_3.get("SVM_Acc")
```

[133]:
```
SVM_Outputs_5 = SVM(pine_5_Train,  pine_5_Test, pine_5_Y_Train, pine_5_Y_Test)
SVM_5 = SVM_Outputs_5.get("SVM_Acc")
```

[135]:
```
SVM_Outputs_220 = SVM(pine_220_Train,  pine_220_Test, pine_220_Y_Train,␣
 ↪pine_220_Y_Test)
SVM_220 = SVM_Outputs_220.get("SVM_Acc")
```

[136]:
```
SVM_Outputs_222 = SVM(pine_222_Train,  pine_222_Test, pine_222_Y_Train,␣
 ↪pine_222_Y_Test)
SVM_222 = SVM_Outputs_222.get("SVM_Acc")
```

KNN Scores

[137]:
```
KNN_Outputs_3 = KNN(pine_3_Train,  pine_3_Test, pine_3_Y_Train,␣
↪pine_3_Y_Test,14)
KNN_3 = KNN_Outputs_3.get("KNN_Acc")
```

[163]:
```
KNN_Outputs_5 = KNN(pine_5_Train,  pine_5_Test, pine_5_Y_Train,␣
↪pine_5_Y_Test,11)
KNN_5 = KNN_Outputs_5.get("KNN_Acc")
```

[165]:
```
KNN_Outputs_220 = KNN(pine_220_Train,  pine_220_Test, pine_220_Y_Train,␣
↪pine_220_Y_Test,10)
KNN_220 = KNN_Outputs_220.get("KNN_Acc")
```

[166]:
```
KNN_Outputs_222 = KNN(pine_222_Train,  pine_222_Test, pine_222_Y_Train,␣
↪pine_222_Y_Test,10)
KNN_222 = KNN_Outputs_222.get("KNN_Acc")
```

LR Scores

[143]:
```
LR_Outputs_3 = LR(pine_3_Train,  pine_3_Test, pine_3_Y_Train, pine_3_Y_Test,100)
LR_3 = LR_Outputs_3.get("LR_Acc")
```

[144]:
```
LR_Outputs_5 = LR(pine_5_Train,  pine_5_Test, pine_5_Y_Train, pine_5_Y_Test,100)
LR_5 = LR_Outputs_5.get("LR_Acc")
```

[146]:
```
LR_Outputs_220 = LR(pine_220_Train,  pine_220_Test, pine_220_Y_Train,␣
↪pine_220_Y_Test,220)
LR_220 = LR_Outputs_220.get("LR_Acc")
```

[147]:
```
LR_Outputs_222 = LR(pine_222_Train,  pine_222_Test, pine_222_Y_Train,␣
↪pine_222_Y_Test,222)
LR_222 = LR_Outputs_222.get("LR_Acc")
```

NN Scores

[148]:
```
NN_Outputs_3 = NN(pine_3_train, pine_3_test, pine_3_y_train, pine_3_y_test, 3)
NN_3 = NN_Outputs_3.get("NN_Acc")
```

```
Epoch 1/50
158/158 [==============================] - 0s 1ms/step - loss: 0.3828 -
binary_accuracy: 0.9169
Epoch 2/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1525 -
binary_accuracy: 0.9551
Epoch 3/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1306 -
binary_accuracy: 0.9558
Epoch 4/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1225 -
binary_accuracy: 0.9561
```

```
Epoch 5/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1175 -
binary_accuracy: 0.9563
Epoch 6/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1140 -
binary_accuracy: 0.9567
Epoch 7/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1114 -
binary_accuracy: 0.9570
Epoch 8/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1093 -
binary_accuracy: 0.9576
Epoch 9/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1076 -
binary_accuracy: 0.9583
Epoch 10/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1063 -
binary_accuracy: 0.9587
Epoch 11/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1052 -
binary_accuracy: 0.9590
Epoch 12/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1043 -
binary_accuracy: 0.9593
Epoch 13/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1035 -
binary_accuracy: 0.9593
Epoch 14/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1028 -
binary_accuracy: 0.9598
Epoch 15/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1022 -
binary_accuracy: 0.9598
Epoch 16/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1016 -
binary_accuracy: 0.9599
Epoch 17/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1012 -
binary_accuracy: 0.9598
Epoch 18/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1007 -
binary_accuracy: 0.9601
Epoch 19/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1004 -
binary_accuracy: 0.9600
Epoch 20/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1000 -
binary_accuracy: 0.9602
```

```
Epoch 21/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0996 -
binary_accuracy: 0.9603
Epoch 22/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0993 -
binary_accuracy: 0.9603
Epoch 23/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0990 -
binary_accuracy: 0.9603
Epoch 24/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0987 -
binary_accuracy: 0.9604
Epoch 25/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0983 -
binary_accuracy: 0.9604
Epoch 26/50
158/158 [==============================] - 0s 2ms/step - loss: 0.0981 -
binary_accuracy: 0.9606
Epoch 27/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0979 -
binary_accuracy: 0.9606
Epoch 28/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0976 -
binary_accuracy: 0.9605
Epoch 29/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0974 -
binary_accuracy: 0.9606
Epoch 30/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0971 -
binary_accuracy: 0.9607
Epoch 31/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0969 -
binary_accuracy: 0.9606
Epoch 32/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0967 -
binary_accuracy: 0.9607
Epoch 33/50
158/158 [==============================] - 0s 979us/step - loss: 0.0965 -
binary_accuracy: 0.9608
Epoch 34/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0963 -
binary_accuracy: 0.9609
Epoch 35/50
158/158 [==============================] - 0s 955us/step - loss: 0.0961 -
binary_accuracy: 0.9609
Epoch 36/50
158/158 [==============================] - 0s 949us/step - loss: 0.0959 -
binary_accuracy: 0.9609
```

```
Epoch 37/50
158/158 [==============================] - 0s 944us/step - loss: 0.0957 -
binary_accuracy: 0.9609
Epoch 38/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0955 -
binary_accuracy: 0.9610
Epoch 39/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0954 -
binary_accuracy: 0.9612
Epoch 40/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0951 -
binary_accuracy: 0.9612
Epoch 41/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0950 -
binary_accuracy: 0.9611
Epoch 42/50
158/158 [==============================] - 0s 996us/step - loss: 0.0948 -
binary_accuracy: 0.9613
Epoch 43/50
158/158 [==============================] - 0s 936us/step - loss: 0.0947 -
binary_accuracy: 0.9612
Epoch 44/50
158/158 [==============================] - 0s 943us/step - loss: 0.0945 -
binary_accuracy: 0.9614
Epoch 45/50
158/158 [==============================] - 0s 931us/step - loss: 0.0943 -
binary_accuracy: 0.9615
Epoch 46/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0942 -
binary_accuracy: 0.9614
Epoch 47/50
158/158 [==============================] - 0s 955us/step - loss: 0.0940 -
binary_accuracy: 0.9615
Epoch 48/50
158/158 [==============================] - 0s 936us/step - loss: 0.0939 -
binary_accuracy: 0.9615
Epoch 49/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0937 -
binary_accuracy: 0.9615
Epoch 50/50
158/158 [==============================] - 0s 940us/step - loss: 0.0935 -
binary_accuracy: 0.9617
```

[149]:
```
NN_Outputs_5 = NN(pine_5_train, pine_5_test, pine_5_y_train, pine_5_y_test, 5)
NN_5 = NN_Outputs_5.get("NN_Acc")
```

```
Epoch 1/50
158/158 [==============================] - 0s 1ms/step - loss: 0.3759 -
```

```
binary_accuracy: 0.9020
Epoch 2/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1457 -
binary_accuracy: 0.9554
Epoch 3/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1201 -
binary_accuracy: 0.9572
Epoch 4/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1086 -
binary_accuracy: 0.9590
Epoch 5/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1012 -
binary_accuracy: 0.9595
Epoch 6/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0958 -
binary_accuracy: 0.9603
Epoch 7/50
158/158 [==============================] - 0s 936us/step - loss: 0.0917 -
binary_accuracy: 0.9617
Epoch 8/50
158/158 [==============================] - 0s 930us/step - loss: 0.0883 -
binary_accuracy: 0.9629
Epoch 9/50
158/158 [==============================] - 0s 930us/step - loss: 0.0854 -
binary_accuracy: 0.9641
Epoch 10/50
158/158 [==============================] - 0s 936us/step - loss: 0.0829 -
binary_accuracy: 0.9652
Epoch 11/50
158/158 [==============================] - 0s 936us/step - loss: 0.0807 -
binary_accuracy: 0.9661
Epoch 12/50
158/158 [==============================] - 0s 934us/step - loss: 0.0787 -
binary_accuracy: 0.9668
Epoch 13/50
158/158 [==============================] - 0s 967us/step - loss: 0.0769 -
binary_accuracy: 0.9677
Epoch 14/50
158/158 [==============================] - 0s 957us/step - loss: 0.0753 -
binary_accuracy: 0.9684
Epoch 15/50
158/158 [==============================] - 0s 943us/step - loss: 0.0738 -
binary_accuracy: 0.9691
Epoch 16/50
158/158 [==============================] - 0s 943us/step - loss: 0.0723 -
binary_accuracy: 0.9698
Epoch 17/50
158/158 [==============================] - 0s 936us/step - loss: 0.0711 -
```

```
binary_accuracy: 0.9706
Epoch 18/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0697 -
binary_accuracy: 0.9710
Epoch 19/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0686 -
binary_accuracy: 0.9716
Epoch 20/50
158/158 [==============================] - 0s 998us/step - loss: 0.0674 -
binary_accuracy: 0.9722
Epoch 21/50
158/158 [==============================] - 0s 991us/step - loss: 0.0662 -
binary_accuracy: 0.9727
Epoch 22/50
158/158 [==============================] - 0s 943us/step - loss: 0.0652 -
binary_accuracy: 0.9732
Epoch 23/50
158/158 [==============================] - 0s 974us/step - loss: 0.0643 -
binary_accuracy: 0.9738
Epoch 24/50
158/158 [==============================] - 0s 988us/step - loss: 0.0634 -
binary_accuracy: 0.9742
Epoch 25/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0625 -
binary_accuracy: 0.9745
Epoch 26/50
158/158 [==============================] - 0s 976us/step - loss: 0.0617 -
binary_accuracy: 0.9749
Epoch 27/50
158/158 [==============================] - 0s 943us/step - loss: 0.0609 -
binary_accuracy: 0.9754
Epoch 28/50
158/158 [==============================] - 0s 943us/step - loss: 0.0602 -
binary_accuracy: 0.9758
Epoch 29/50
158/158 [==============================] - 0s 968us/step - loss: 0.0595 -
binary_accuracy: 0.9763
Epoch 30/50
158/158 [==============================] - 0s 985us/step - loss: 0.0589 -
binary_accuracy: 0.9764
Epoch 31/50
158/158 [==============================] - 0s 946us/step - loss: 0.0582 -
binary_accuracy: 0.9768
Epoch 32/50
158/158 [==============================] - 0s 946us/step - loss: 0.0576 -
binary_accuracy: 0.9771
Epoch 33/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0570 -
```

```
binary_accuracy: 0.9774
Epoch 34/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0566 -
binary_accuracy: 0.9776
Epoch 35/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0559 -
binary_accuracy: 0.9781
Epoch 36/50
158/158 [==============================] - 0s 956us/step - loss: 0.0555 -
binary_accuracy: 0.9782
Epoch 37/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0550 -
binary_accuracy: 0.9784
Epoch 38/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0545 -
binary_accuracy: 0.9785
Epoch 39/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0542 -
binary_accuracy: 0.9788
Epoch 40/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0537 -
binary_accuracy: 0.9788
Epoch 41/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0533 -
binary_accuracy: 0.9790
Epoch 42/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0529 -
binary_accuracy: 0.9791
Epoch 43/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0526 -
binary_accuracy: 0.9793
Epoch 44/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0522 -
binary_accuracy: 0.9796
Epoch 45/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0519 -
binary_accuracy: 0.9796
Epoch 46/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0516 -
binary_accuracy: 0.9798
Epoch 47/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0513 -
binary_accuracy: 0.9800
Epoch 48/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0510 -
binary_accuracy: 0.9800
Epoch 49/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0506 -
```

```
binary_accuracy: 0.9801
Epoch 50/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0504 -
binary_accuracy: 0.9802
```

[150]:
```
NN_Outputs_220 = NN(pine_220_train, pine_220_test, pine_220_y_train,␣
↪pine_220_y_test, 220)
NN_220 = NN_Outputs_220.get("NN_Acc")
```

```
Epoch 1/50
158/158 [==============================] - 1s 1ms/step - loss: 0.1987 -
binary_accuracy: 0.9258
Epoch 2/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1116 -
binary_accuracy: 0.9581
Epoch 3/50
158/158 [==============================] - 0s 2ms/step - loss: 0.1016 -
binary_accuracy: 0.9598
Epoch 4/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0959 -
binary_accuracy: 0.9613
Epoch 5/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0916 -
binary_accuracy: 0.9629
Epoch 6/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0880 -
binary_accuracy: 0.9645
Epoch 7/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0851 -
binary_accuracy: 0.9656
Epoch 8/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0829 -
binary_accuracy: 0.9664
Epoch 9/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0804 -
binary_accuracy: 0.9672
Epoch 10/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0788 -
binary_accuracy: 0.9678
Epoch 11/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0773 -
binary_accuracy: 0.9686
Epoch 12/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0762 -
binary_accuracy: 0.9688
Epoch 13/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0746 -
binary_accuracy: 0.9697
```

```
Epoch 14/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0733 -
binary_accuracy: 0.9701
Epoch 15/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0721 -
binary_accuracy: 0.9706
Epoch 16/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0712 -
binary_accuracy: 0.9708
Epoch 17/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0705 -
binary_accuracy: 0.9712
Epoch 18/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0693 -
binary_accuracy: 0.9719
Epoch 19/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0685 -
binary_accuracy: 0.9724
Epoch 20/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0678 -
binary_accuracy: 0.9723
Epoch 21/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0666 -
binary_accuracy: 0.9728
Epoch 22/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0664 -
binary_accuracy: 0.9727
Epoch 23/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0655 -
binary_accuracy: 0.9733
Epoch 24/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0647 -
binary_accuracy: 0.9736
Epoch 25/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0647 -
binary_accuracy: 0.9734
Epoch 26/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0641 -
binary_accuracy: 0.9735
Epoch 27/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0633 -
binary_accuracy: 0.9739
Epoch 28/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0621 -
binary_accuracy: 0.9746
Epoch 29/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0619 -
binary_accuracy: 0.9746
```

```
Epoch 30/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0613 -
binary_accuracy: 0.9748
Epoch 31/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0608 -
binary_accuracy: 0.9752
Epoch 32/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0606 -
binary_accuracy: 0.9751
Epoch 33/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0597 -
binary_accuracy: 0.9759
Epoch 34/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0593 -
binary_accuracy: 0.9759
Epoch 35/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0588 -
binary_accuracy: 0.9757
Epoch 36/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0586 -
binary_accuracy: 0.9760
Epoch 37/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0580 -
binary_accuracy: 0.9765
Epoch 38/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0581 -
binary_accuracy: 0.9764
Epoch 39/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0576 -
binary_accuracy: 0.9765
Epoch 40/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0573 -
binary_accuracy: 0.9764
Epoch 41/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0564 -
binary_accuracy: 0.9770
Epoch 42/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0564 -
binary_accuracy: 0.9768
Epoch 43/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0559 -
binary_accuracy: 0.9771
Epoch 44/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0557 -
binary_accuracy: 0.9770
Epoch 45/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0554 -
binary_accuracy: 0.9773
```

```
Epoch 46/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0553 -
binary_accuracy: 0.9772
Epoch 47/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0547 -
binary_accuracy: 0.9775
Epoch 48/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0545 -
binary_accuracy: 0.9779
Epoch 49/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0546 -
binary_accuracy: 0.9776
Epoch 50/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0541 -
binary_accuracy: 0.9780
```

[151]:
```
NN_Outputs_222 = NN(pine_222_train, pine_222_test, pine_222_y_train,␣
↪pine_222_y_test, 222)
NN_222 = NN_Outputs_222.get("NN_Acc")
```

```
Epoch 1/50
158/158 [==============================] - 1s 1ms/step - loss: 0.1908 -
binary_accuracy: 0.9310
Epoch 2/50
158/158 [==============================] - 0s 1ms/step - loss: 0.1097 -
binary_accuracy: 0.9581
Epoch 3/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0976 -
binary_accuracy: 0.9608
Epoch 4/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0905 -
binary_accuracy: 0.9626
Epoch 5/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0847 -
binary_accuracy: 0.9645
Epoch 6/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0802 -
binary_accuracy: 0.9664
Epoch 7/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0765 -
binary_accuracy: 0.9677
Epoch 8/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0735 -
binary_accuracy: 0.9694
Epoch 9/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0707 -
binary_accuracy: 0.9703
Epoch 10/50
```

```
158/158 [==============================] - 0s 1ms/step - loss: 0.0683 -
binary_accuracy: 0.9713
Epoch 11/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0659 -
binary_accuracy: 0.9727
Epoch 12/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0638 -
binary_accuracy: 0.9735
Epoch 13/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0619 -
binary_accuracy: 0.9746
Epoch 14/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0602 -
binary_accuracy: 0.9753
Epoch 15/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0585 -
binary_accuracy: 0.9761
Epoch 16/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0573 -
binary_accuracy: 0.9765
Epoch 17/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0557 -
binary_accuracy: 0.9772
Epoch 18/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0547 -
binary_accuracy: 0.9777
Epoch 19/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0536 -
binary_accuracy: 0.9782
Epoch 20/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0523 -
binary_accuracy: 0.9787
Epoch 21/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0516 -
binary_accuracy: 0.9790
Epoch 22/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0505 -
binary_accuracy: 0.9794
Epoch 23/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0498 -
binary_accuracy: 0.9796
Epoch 24/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0494 -
binary_accuracy: 0.9795
Epoch 25/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0481 -
binary_accuracy: 0.9804
Epoch 26/50
```

```
158/158 [==============================] - 0s 1ms/step - loss: 0.0471 -
binary_accuracy: 0.9808
Epoch 27/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0467 -
binary_accuracy: 0.9812
Epoch 28/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0460 -
binary_accuracy: 0.9814
Epoch 29/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0456 -
binary_accuracy: 0.9815
Epoch 30/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0447 -
binary_accuracy: 0.9820
Epoch 31/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0442 -
binary_accuracy: 0.9822
Epoch 32/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0438 -
binary_accuracy: 0.9824
Epoch 33/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0433 -
binary_accuracy: 0.9827
Epoch 34/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0428 -
binary_accuracy: 0.9828
Epoch 35/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0420 -
binary_accuracy: 0.9831
Epoch 36/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0416 -
binary_accuracy: 0.9834
Epoch 37/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0413 -
binary_accuracy: 0.9833
Epoch 38/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0410 -
binary_accuracy: 0.9836
Epoch 39/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0401 -
binary_accuracy: 0.9839
Epoch 40/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0403 -
binary_accuracy: 0.9839
Epoch 41/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0394 -
binary_accuracy: 0.9843
Epoch 42/50
```

```
158/158 [==============================] - 0s 1ms/step - loss: 0.0390 -
binary_accuracy: 0.9845
Epoch 43/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0387 -
binary_accuracy: 0.9846
Epoch 44/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0385 -
binary_accuracy: 0.9847
Epoch 45/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0380 -
binary_accuracy: 0.9850
Epoch 46/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0379 -
binary_accuracy: 0.9851
Epoch 47/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0372 -
binary_accuracy: 0.9854
Epoch 48/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0370 -
binary_accuracy: 0.9855
Epoch 49/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0364 -
binary_accuracy: 0.9857
Epoch 50/50
158/158 [==============================] - 0s 1ms/step - loss: 0.0367 -
binary_accuracy: 0.9855
```

CNN Scores

[152]:
```
CNN_Outputs_3 = CNN(pine_3_train, pine_3_test, pine_3_y_train, pine_3_y_test,␣
↪3,1)
CNN_3 = CNN_Outputs_3.get("CNN_Acc")
print(CNN_3)
```

```
Epoch 1/10
247/247 [==============================] - 1s 1ms/step - loss: 0.2039 -
binary_accuracy: 0.9441
Epoch 2/10
247/247 [==============================] - 0s 1ms/step - loss: 0.1191 -
binary_accuracy: 0.9568
Epoch 3/10
247/247 [==============================] - 0s 1ms/step - loss: 0.1099 -
binary_accuracy: 0.9582
Epoch 4/10
247/247 [==============================] - 0s 1ms/step - loss: 0.1057 -
binary_accuracy: 0.9589
Epoch 5/10
247/247 [==============================] - 0s 1ms/step - loss: 0.1037 -
binary_accuracy: 0.9591
```

```
Epoch 6/10
247/247 [==============================] - 0s 1ms/step - loss: 0.1024 -
binary_accuracy: 0.9593
Epoch 7/10
247/247 [==============================] - 0s 1ms/step - loss: 0.1009 -
binary_accuracy: 0.9595
Epoch 8/10
247/247 [==============================] - 0s 1ms/step - loss: 0.1001 -
binary_accuracy: 0.9598
Epoch 9/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0992 -
binary_accuracy: 0.9601
Epoch 10/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0984 -
binary_accuracy: 0.9602
{0.6355335742819098}
```

[153]:
```
CNN_Outputs_5 = CNN(pine_5_train, pine_5_test, pine_5_y_train, pine_5_y_test,␣
↪5,1)
CNN_5 = CNN_Outputs_5.get("CNN_Acc")
print(CNN_5)
```

```
Epoch 1/10
247/247 [==============================] - 1s 2ms/step - loss: 0.1824 -
binary_accuracy: 0.9505
Epoch 2/10
247/247 [==============================] - 0s 2ms/step - loss: 0.1074 -
binary_accuracy: 0.9595
Epoch 3/10
247/247 [==============================] - 0s 2ms/step - loss: 0.0946 -
binary_accuracy: 0.9612
Epoch 4/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0869 -
binary_accuracy: 0.9634
Epoch 5/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0804 -
binary_accuracy: 0.9666
Epoch 6/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0758 -
binary_accuracy: 0.9687
Epoch 7/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0715 -
binary_accuracy: 0.9705
Epoch 8/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0681 -
binary_accuracy: 0.9723
Epoch 9/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0649 -
```

```
binary_accuracy: 0.9734
Epoch 10/10
247/247 [==============================] - 0s 1ms/step - loss: 0.0623 -
binary_accuracy: 0.9746
{0.7955107475746623}
```

[160]:
```
CNN_Outputs_220 = CNN(pine_220_train, pine_220_test, pine_220_y_train,␣
↪pine_220_y_test, 220,1)
CNN_220 = CNN_Outputs_220.get("CNN_Acc")
print(CNN_220)
```

```
Epoch 1/10
247/247 [==============================] - 4s 16ms/step - loss: 0.1154 -
binary_accuracy: 0.9559
Epoch 2/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0924 -
binary_accuracy: 0.9626
Epoch 3/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0847 -
binary_accuracy: 0.9649
Epoch 4/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0795 -
binary_accuracy: 0.9672
Epoch 5/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0754 -
binary_accuracy: 0.9688
Epoch 6/10
247/247 [==============================] - 4s 18ms/step - loss: 0.0725 -
binary_accuracy: 0.9699
Epoch 7/10
247/247 [==============================] - 5s 19ms/step - loss: 0.0690 -
binary_accuracy: 0.9715
Epoch 8/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0672 -
binary_accuracy: 0.9725
Epoch 9/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0646 -
binary_accuracy: 0.9735
Epoch 10/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0628 -
binary_accuracy: 0.9741
0.7490964428381206
```

[158]:
```
CNN_Outputs_222 = CNN(pine_222_train, pine_222_test, pine_222_y_train,␣
↪pine_222_y_test, 222,1)
CNN_222 = CNN_Outputs_222.get("CNN_Acc")
print(CNN_222)
```

```
Epoch 1/10
247/247 [==============================] - 4s 15ms/step - loss: 0.1125 -
binary_accuracy: 0.9575
Epoch 2/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0848 -
binary_accuracy: 0.9648
Epoch 3/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0749 -
binary_accuracy: 0.9687
Epoch 4/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0673 -
binary_accuracy: 0.9719
Epoch 5/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0621 -
binary_accuracy: 0.9745
Epoch 6/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0573 -
binary_accuracy: 0.9764
Epoch 7/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0534 -
binary_accuracy: 0.9781
Epoch 8/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0496 -
binary_accuracy: 0.9796
Epoch 9/10
247/247 [==============================] - 4s 16ms/step - loss: 0.0476 -
binary_accuracy: 0.9806
Epoch 10/10
247/247 [==============================] - 4s 15ms/step - loss: 0.0441 -
binary_accuracy: 0.9823
0.8152938938558113
```

0.4.2 Creating the Accuracy Scores Table

[159]:
```
data = [['SVM', SVM_3, SVM_5, SVM_220, SVM_222], ['KNN', KNN_3, KNN_5, KNN_220␣
↪, KNN_222], ['LR', LR_3 , LR_5, LR_220 , LR_222],
        ['NN', NN_3 , NN_5, NN_220 , NN_222], ['CNN', CNN_3, CNN_5, CNN_220,␣
↪CNN_222]]

df = pd.DataFrame(data, columns = ['Models', 'Pine_3', 'Pine_5', 'Pine_220',␣
↪'Pine_222'])
df
```

[159]:

	Models	Pine_3	Pine_5	Pine_220	\
0	SVM	{0.6709149705155031}	{0.7487159977173293}	{0.7863800646756706}	
1	KNN	{0.6891763363134867}	{0.900703823473464}	{0.695263458246148}	
2	LR	{0.5999619554879209}	{0.6237397755373787}	{0.7715427049648088}	

```
3     NN  {0.6313486779532053}   {0.848202396804261}   {0.780483165303405}
4    CNN  {0.6355335742819098}  {0.7955107475746623}  {0.7513791135628686}

                Pine_222
0  {0.7934182994103101}
1  {0.7334981928856762}
2  {0.7930378542895188}
3  {0.8548601864181092}
4              0.815294
```

[ ]: