Machine Learning Assignment

TISP variable selection

Question 1 (a)

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
from sklearn.preprocessing import StandardScaler
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
def TISP(X_Train, y_train, X_test, y_test, Lambda):
    X = X_Train
    Y = y_train
    X_test = X_test
    Y_test = y_test
    N, P = X.shape
    N_test = X_test.shape[0]
    W_old = np.zeros(P)
    train_errors_fr_iterations = []

    for i in range(100):

        gradient =  np.dot(X.T, Y - 1 / (1 + np.exp(-np.dot(X, W_old))))
        W_temp = W_old +  gradient
        W_new = W_temp * (np.abs(W_temp) > Lambda)
        W_old = W_new


        P_hat = 1 / (1 + np.exp(-np.dot(X, W_new)))
        Y_hat = 2 * (P_hat > 0.5) - 1
        Tr_table = pd.crosstab(Y, Y_hat)
        Tr_err = 1 - np.sum(np.diag(Tr_table)) / N
        train_errors_fr_iterations.append(Tr_err)

    var = np.sum(W_new != 0)
    P_hat = 1 / (1 + np.exp(-np.dot(X, W_new)))
    Y_hat = 2 * (P_hat > 0.5) - 1
    Tr_table = pd.crosstab(Y, Y_hat)
    Tr_err = 1 - np.sum(np.diag(Tr_table)) / N


    P_hat_test = 1 / (1 + np.exp(-np.dot(X_test, W_new)))
    Y_hat_test = 2 * (P_hat_test > 0.5) - 1
    Ts_table = pd.crosstab(Y_test, Y_hat_test)
    Ts_err = 1 - np.sum(np.diag(Ts_table)) / N_test

    return {'W_hat': W_new,
            'Train_error': Tr_err,
            'Test_error': Ts_err,
            'Selected_Features': var,
            'Given_Lambda': Lambda,
            'Train_errors': train_errors_fr_iterations}
```

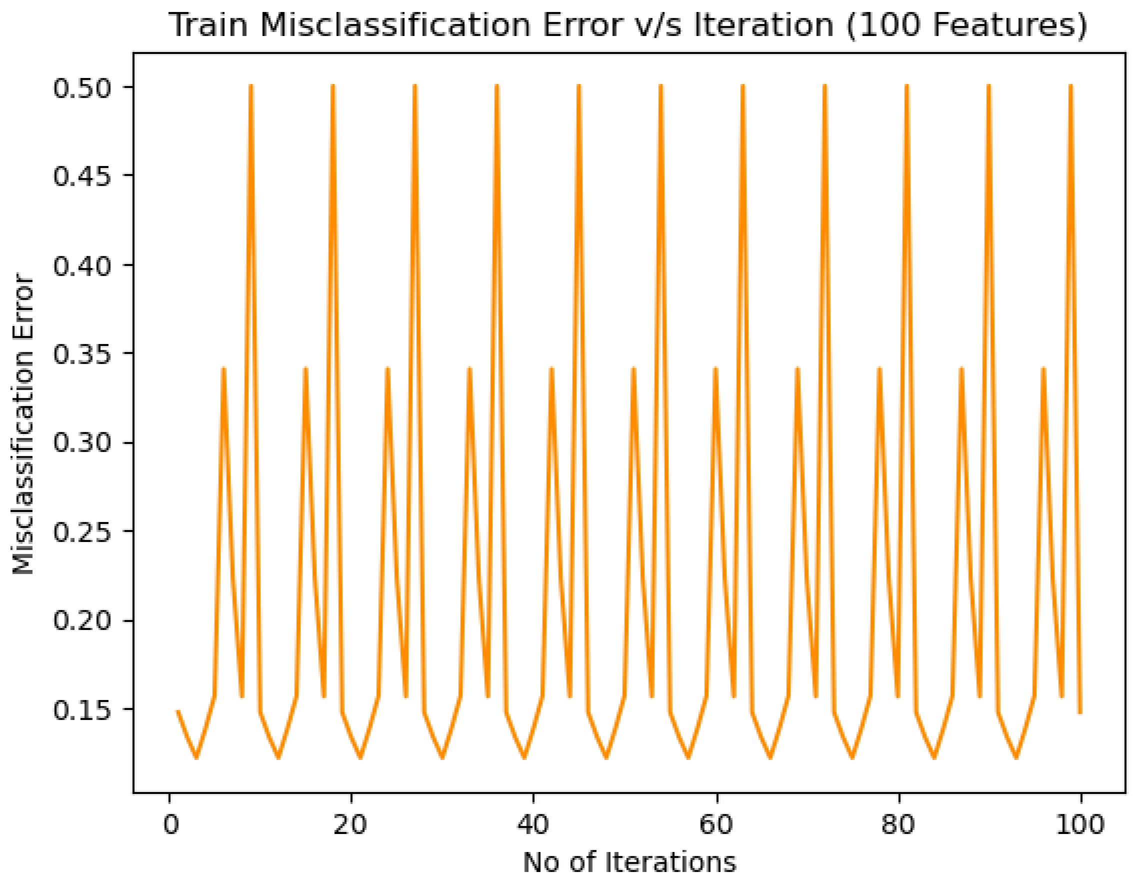
In [3]:

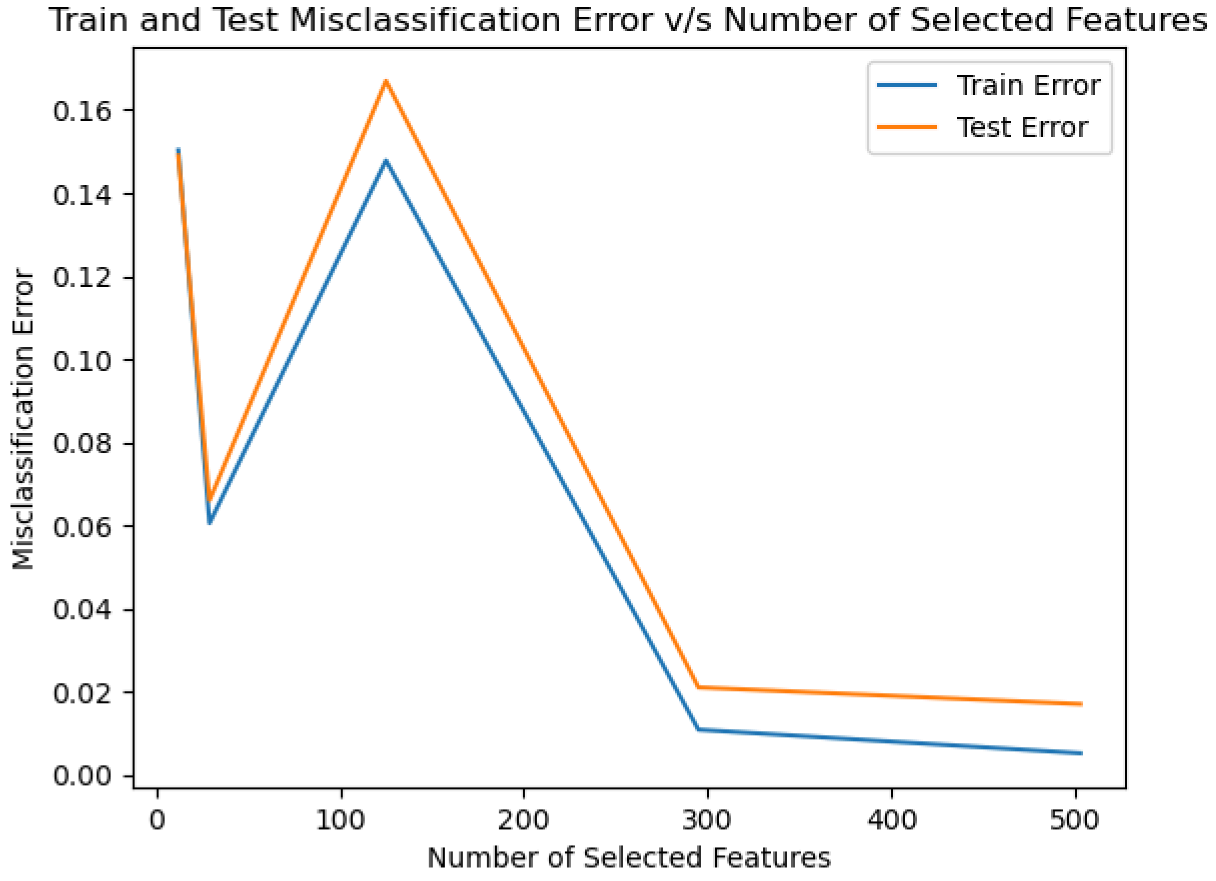
```
X_train = np.loadtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/Gisette/gisette_train.data")
y_train = np.loadtxt("//Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/Gisette/gisette_train.labels")

X_test = np.loadtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/Gisette/gisette_valid.data")
y_test = np.loadtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/Gisette/gisette_valid.labels")

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

y_train = (y_train + 1) // 2
y_test = (y_test + 1) // 2
```

In [4]:

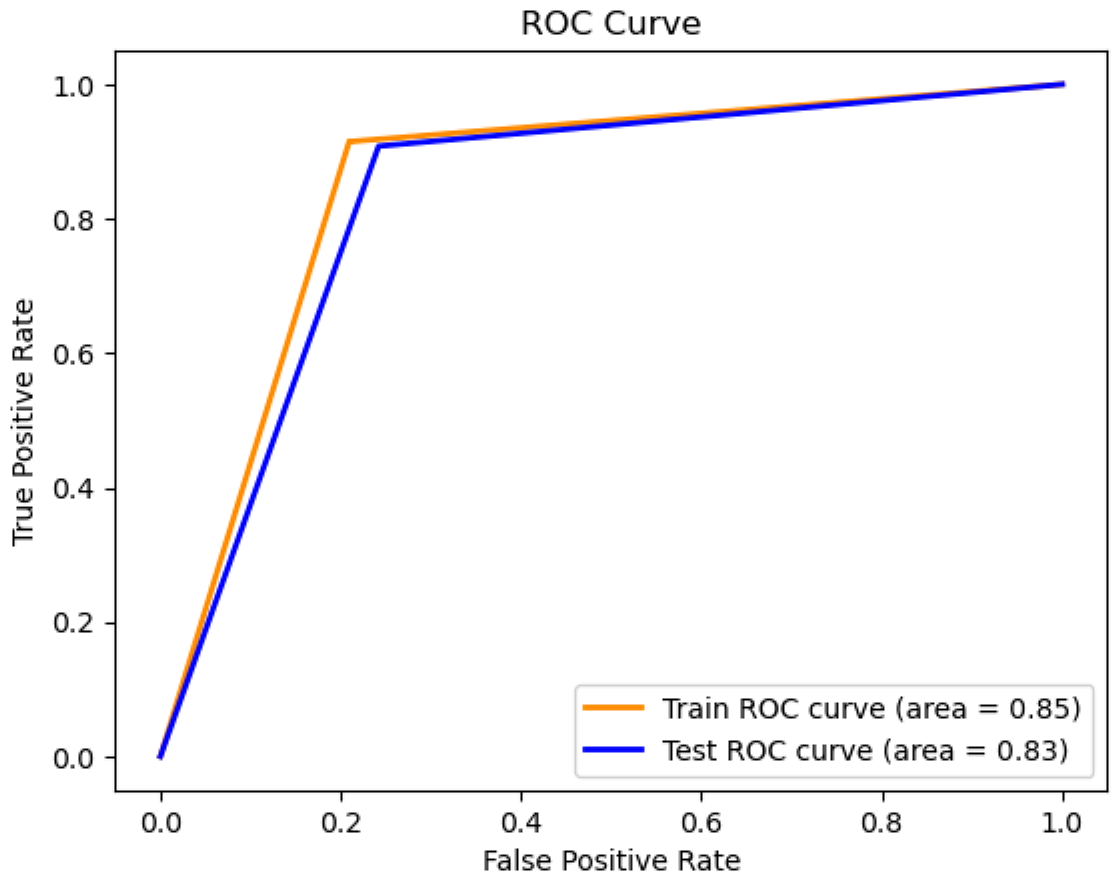
```
lambda_values =[1007.900,880.099,985,262.0, 201.500]
M = len(lambda_values)
TISP_var = np.zeros(M)
TISP_Tr_err = np.zeros(M)
TISP_Ts_err = np.zeros(M)
TISP_weights = []
train_errors_vs_iteration = []

for i in range(M):
    TISP_result = TISP(X_train, y_train, X_test, y_test, Lambda=lambda_values[i])
    TISP_var[i] = TISP_result['Selected_Features']
    TISP_Tr_err[i] = TISP_result['Train_error']
    TISP_Ts_err[i] = TISP_result['Test_error']
    TISP_weights.append(TISP_result['W_hat'])
    train_errors_vs_iteration.append(TISP_result['Train_errors'])

plt.plot(range(1, 101), train_errors_vs_iteration[2],color='darkorange')
plt.xlabel("No of Iterations")
plt.ylabel("Misclassification Error")
plt.title("Train Misclassification Error v/s Iteration (100 Features)")
plt.show()
plt.plot(TISP_var, TISP_Tr_err, label="Train Error")
plt.plot(TISP_var, TISP_Ts_err, label="Test Error")
plt.xlabel("Number of Selected Features")
plt.ylabel("Misclassification Error")
plt.title("Train and Test Misclassification Error v/s Number of Selected Features")
plt.legend()
plt.show()

results = pd.DataFrame({ "Given Lambda": lambda_values,
                        "Selected Features": TISP_var,
                        "Train Error": TISP_Tr_err,
                        "Test Error": TISP_Ts_err
})

print(results)

fpr_test, tpr_test, _ = roc_curve(y_test, 1
                                  / (1 + np.exp(-np.dot(X_test, TISP_weights[2]))))
roc_auc_test = auc(fpr_test, tpr_test)

fpr_train, tpr_train, _ = roc_curve(y_train, 1
                                    / (1 + np.exp(-np.dot(X_train, TISP_weights[2]))))
roc_auc_train = auc(fpr_train, tpr_train)

plt.figure()
plt.plot(fpr_train, tpr_train, color='darkorange', lw=2,
         label=f'Train ROC curve (area = {roc_auc_train:.2f})')
plt.plot(fpr_test, tpr_test, color='blue', lw=2,
         label=f'Test ROC curve (area = {roc_auc_test:.2f})')
```

```
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```





```
   Given Lambda  Selected Features  Train Error  Test Error
0      1007.900               12.0     0.150333       0.149
1       880.099               29.0     0.060500       0.066
2       985.000              125.0     0.147833       0.167
3       262.000              295.0     0.010833       0.021
4       201.500              503.0     0.005167       0.017
```



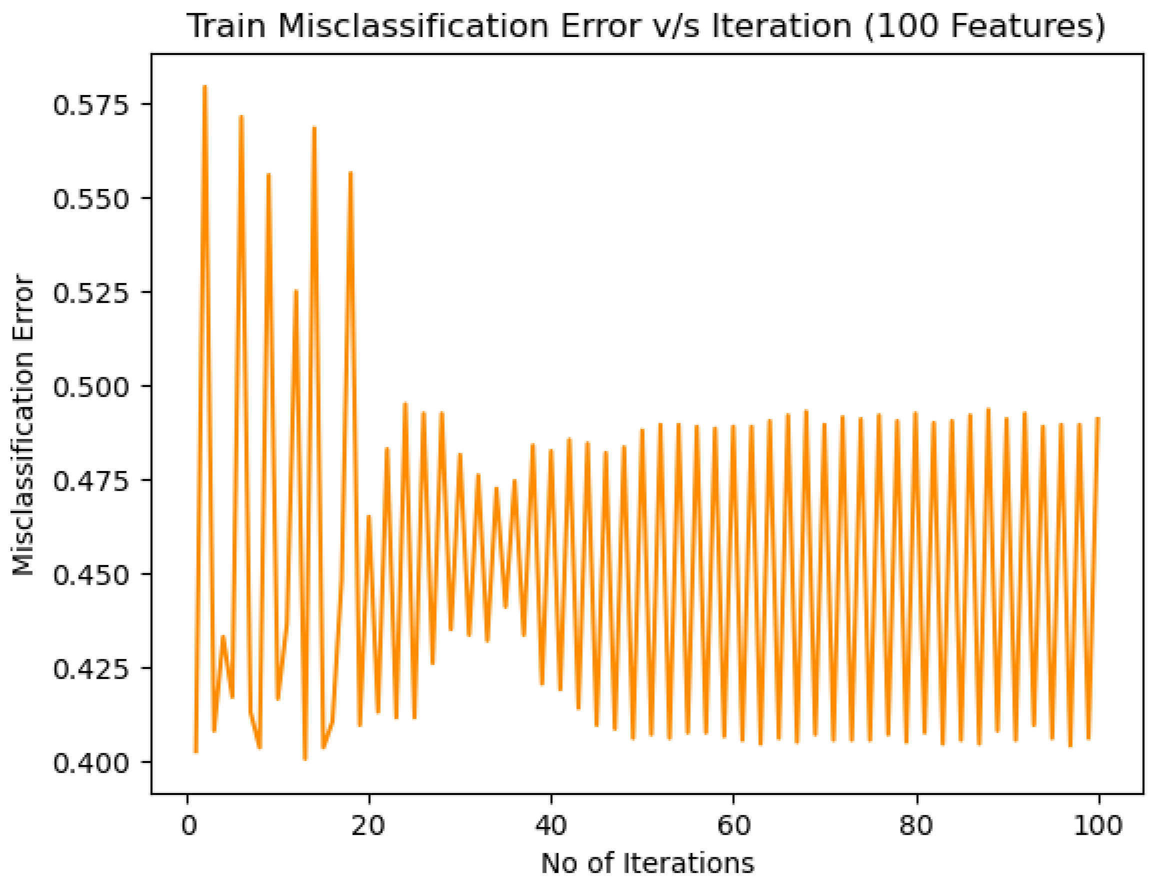
Question 1 (b)

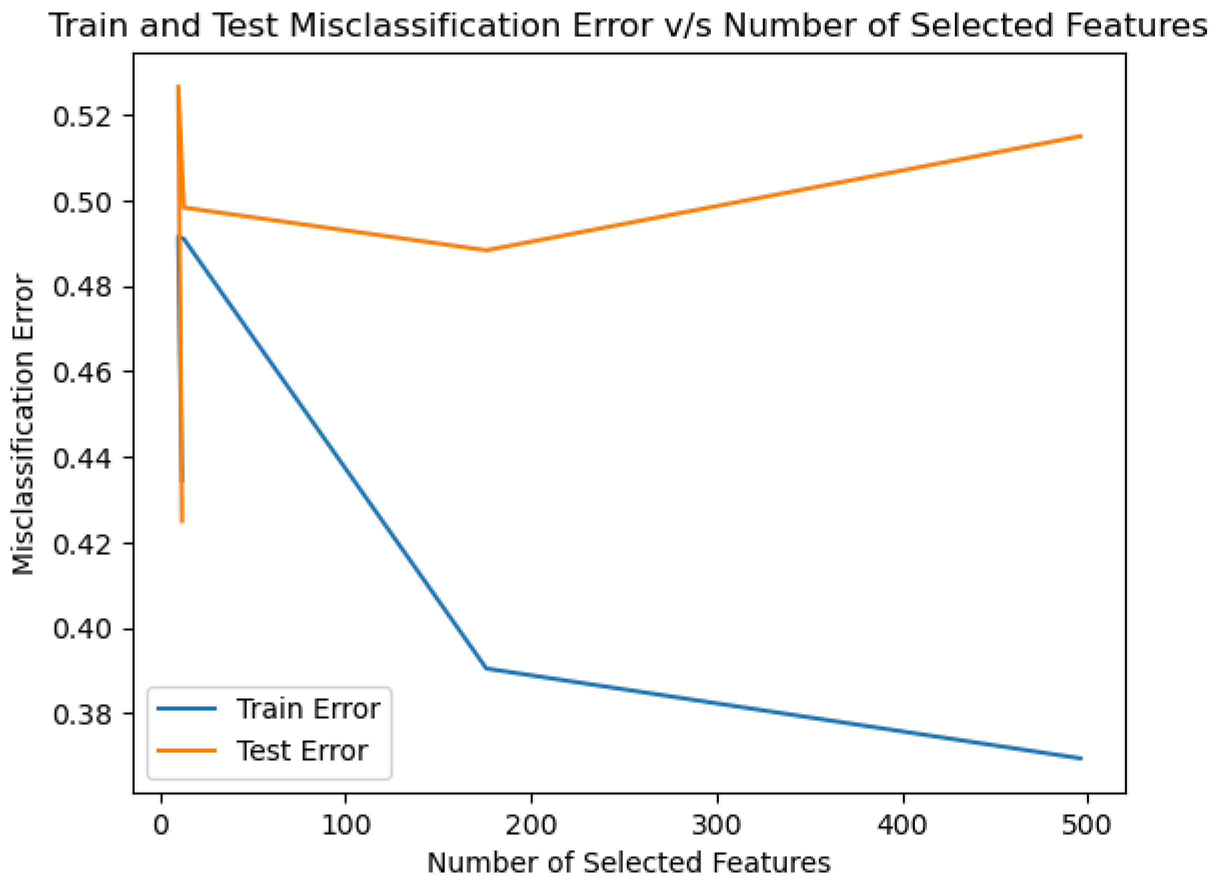
In [5]:
```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
```

In [6]:
```
X_train = np.loadtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/MADELON/madelon_train.data")
y_train = np.loadtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/MADELON/madelon_train.labels")

X_test = np.loadtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/MADELON/madelon_valid.data")
y_test = np.loadtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_4/MADELON/madelon_valid.labels")
```

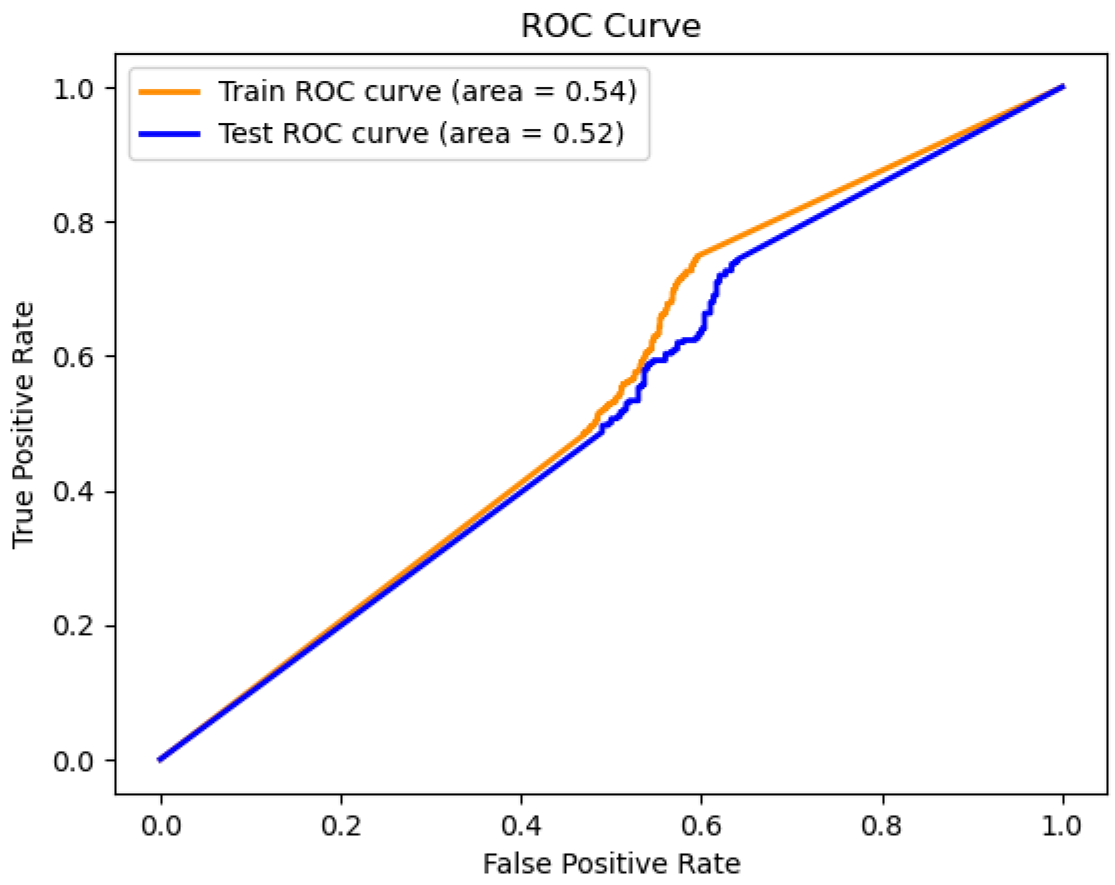
In [7]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)


y_train = (y_train + 1) // 2
y_test = (y_test + 1) // 2
```

In [8]:
```
lambda_values =[160.50,119.0,83.19,42.30,1]
M = len(lambda_values)
TISP_var = np.zeros(M)
TISP_Tr_err = np.zeros(M)
TISP_Ts_err = np.zeros(M)
TISP_weights = []
train_errors_vs_iteration = []

for i in range(M):
    TISP_result = TISP(X_train, y_train, X_test, y_test, Lambda=lambda_values[i])
    TISP_var[i] = TISP_result['Selected_Features']
    TISP_Tr_err[i] = TISP_result['Train_error']
    TISP_Ts_err[i] = TISP_result['Test_error']
    TISP_weights.append(TISP_result['W_hat'])
    train_errors_vs_iteration.append(TISP_result['Train_errors'])

plt.plot(range(1, 101), train_errors_vs_iteration[2],color='darkorange')
plt.xlabel("No of Iterations")
plt.ylabel("Misclassification Error")
plt.title("Train Misclassification Error v/s Iteration (100 Features)")
plt.show()
plt.plot(TISP_var, TISP_Tr_err, label="Train Error")
plt.plot(TISP_var, TISP_Ts_err, label="Test Error")
plt.xlabel("Number of Selected Features")
plt.ylabel("Misclassification Error")
plt.title("Train and Test Misclassification Error v/s Number of Selected Features")
plt.legend()
plt.show()

results = pd.DataFrame({ "Given Lambda": lambda_values,
                        "Selected Features": TISP_var,
                        "Train Error": TISP_Tr_err,
                        "Test Error": TISP_Ts_err
})

print(results)

fpr_test, tpr_test, _ = roc_curve(y_test, 1
                                  / (1 + np.exp(-np.dot(X_test, TISP_weights[2]))))
roc_auc_test = auc(fpr_test, tpr_test)

fpr_train, tpr_train, _ = roc_curve(y_train, 1
                                    / (1 + np.exp(-np.dot(X_train, TISP_weights[2]))))
roc_auc_train = auc(fpr_train, tpr_train)

plt.figure()
plt.plot(fpr_train, tpr_train, color='darkorange', lw=2,
         label=f'Train ROC curve (area = {roc_auc_train:.2f})')
plt.plot(fpr_test, tpr_test, color='blue', lw=2,
         label=f'Test ROC curve (area = {roc_auc_test:.2f})')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```





```
   Given Lambda  Selected Features  Train Error  Test Error
0        160.50               12.0       0.4345    0.425000
1        119.00               10.0       0.4915    0.526667
2         83.19               13.0       0.4910    0.498333
3         42.30              176.0       0.3905    0.488333
4          1.00              496.0       0.3695    0.515000
```



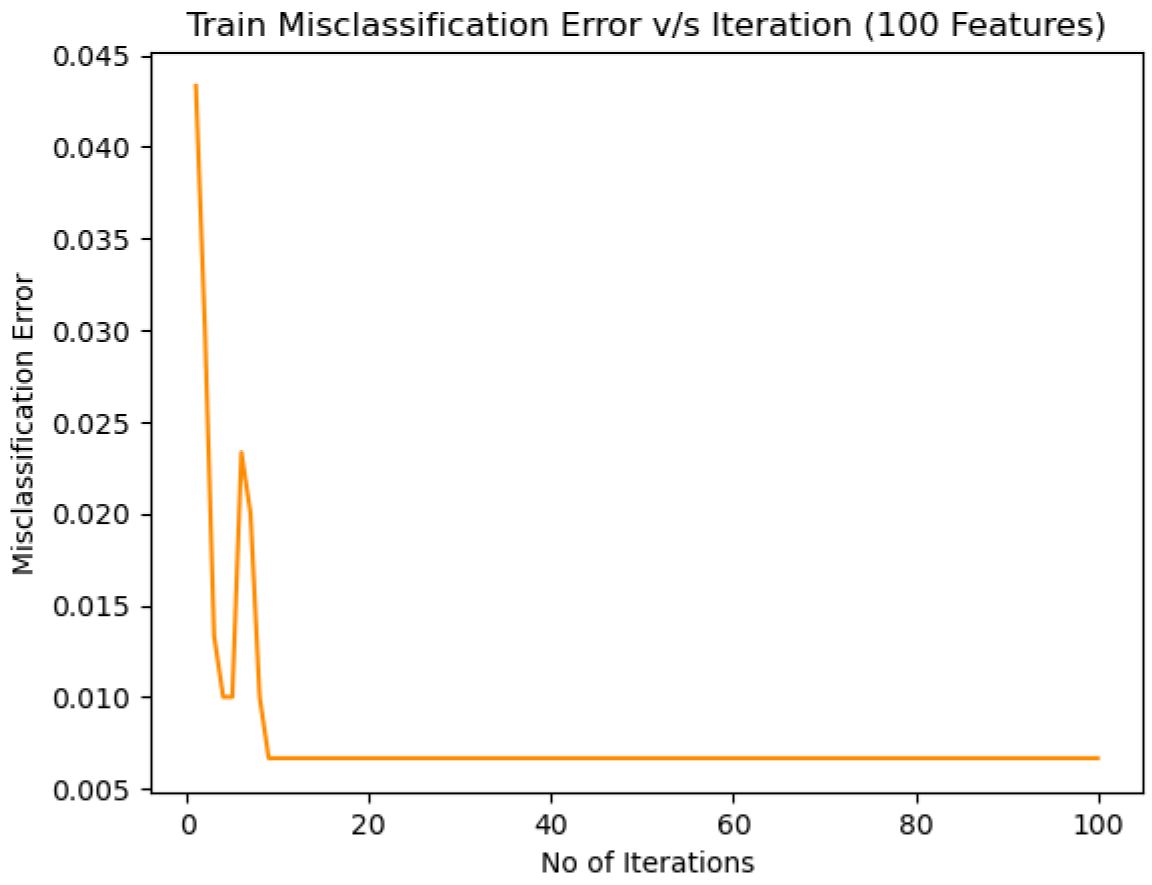
Question 1 (c)

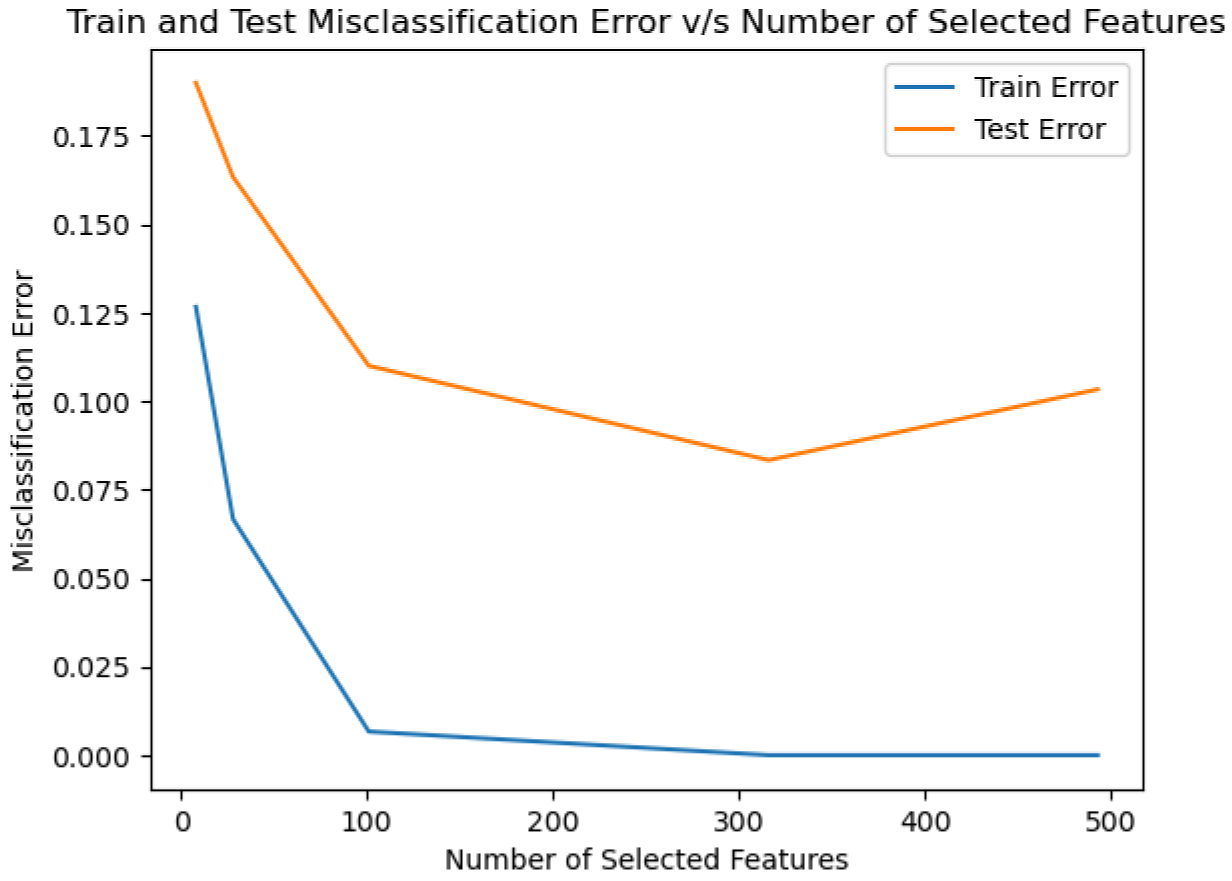
In [9]:

```
X_train = np.genfromtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_3/dexter/dexter_train.csv", delimiter=',')
y_train = np.genfromtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_3/dexter/dexter_train.labels")

X_test = np.genfromtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_3/dexter/dexter_valid.csv", delimiter=',')
y_test = np.genfromtxt("/Users/gaganullas19/Documents/Spring2024/AppliedMachineLearning/Homework_3/dexter/dexter_valid.labels")

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Converting -1 to 0 and keeping 1 as 1
y_train = (y_train + 1) // 2
y_test = (y_test + 1) // 2
```

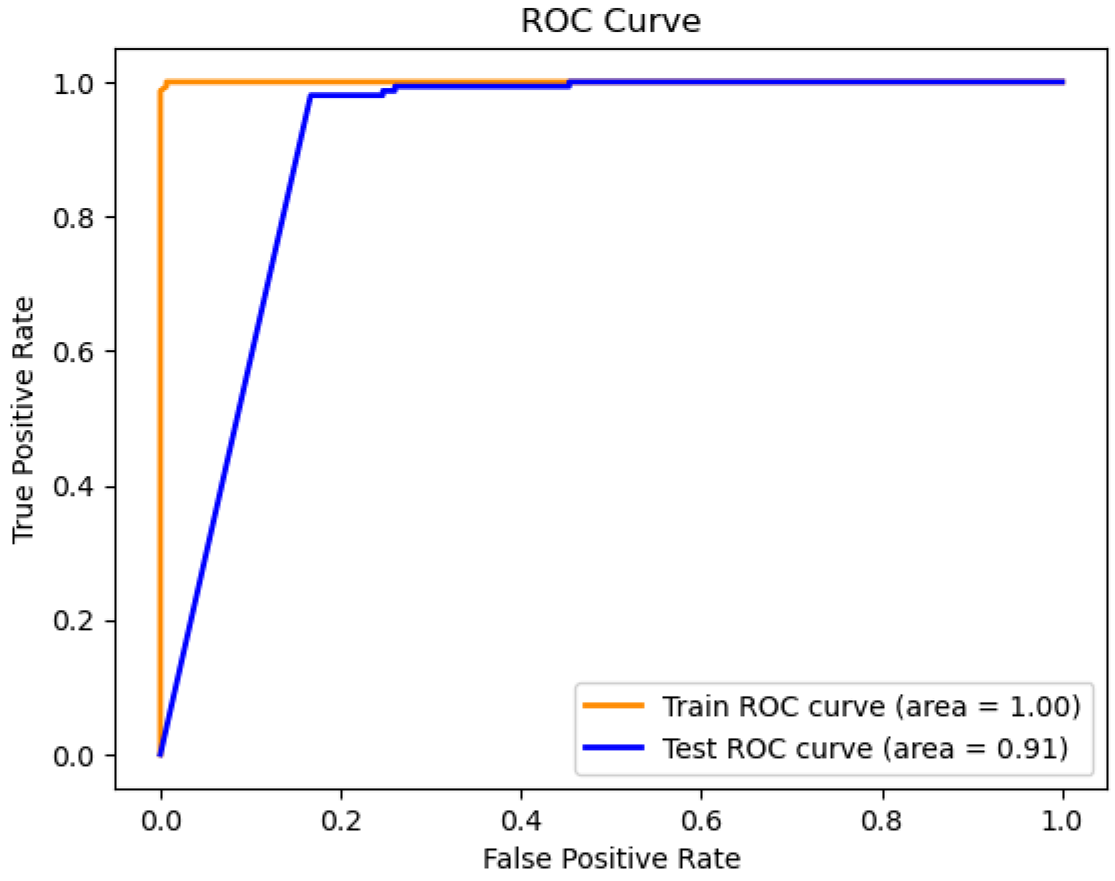
In [10]:

```
lambda_values =[32.001, 25,20.6,16.999,15.135]
M = len(lambda_values)
TISP_var = np.zeros(M)
TISP_Tr_err = np.zeros(M)
TISP_Ts_err = np.zeros(M)
TISP_weights = []
train_errors_vs_iteration = []

for i in range(M):
    TISP_result = TISP(X_train, y_train, X_test, y_test, Lambda=lambda_values[i])
    TISP_var[i] = TISP_result['Selected_Features']
    TISP_Tr_err[i] = TISP_result['Train_error']
    TISP_Ts_err[i] = TISP_result['Test_error']
    TISP_weights.append(TISP_result['W_hat'])
    train_errors_vs_iteration.append(TISP_result['Train_errors'])

plt.plot(range(1, 101), train_errors_vs_iteration[2],color='darkorange')
plt.xlabel("No of Iterations")
plt.ylabel("Misclassification Error")
plt.title("Train Misclassification Error v/s Iteration (100 Features)")
plt.show()
plt.plot(TISP_var, TISP_Tr_err, label="Train Error")
plt.plot(TISP_var, TISP_Ts_err, label="Test Error")
plt.xlabel("Number of Selected Features")
plt.ylabel("Misclassification Error")
plt.title("Train and Test Misclassification Error v/s Number of Selected Features")
plt.legend()
plt.show()

results = pd.DataFrame({ "Given Lambda": lambda_values,
                        "Selected Features": TISP_var,
                        "Train Error": TISP_Tr_err,
                        "Test Error": TISP_Ts_err
})

print(results)

fpr_test, tpr_test, _ = roc_curve(y_test, 1 /
                                  (1 + np.exp(-np.dot(X_test, TISP_weights[2]))))
roc_auc_test = auc(fpr_test, tpr_test)

fpr_train, tpr_train, _ = roc_curve(y_train, 1 /
                                    (1 + np.exp(-np.dot(X_train, TISP_weights[2]))))
roc_auc_train = auc(fpr_train, tpr_train)

plt.figure()
plt.plot(fpr_train, tpr_train, color='darkorange', lw=2,
         label=f'Train ROC curve (area = {roc_auc_train:.2f})')
plt.plot(fpr_test, tpr_test, color='blue', lw=2,
         label=f'Test ROC curve (area = {roc_auc_test:.2f})')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```





```
   Given Lambda  Selected Features  Train Error  Test Error
0        32.001                8.0     0.126667    0.190000
1        25.000               28.0     0.066667    0.163333
2        20.600              101.0     0.006667    0.110000
3        16.999              316.0     0.000000    0.083333
4        15.135              493.0     0.000000    0.103333
```


ROC Curve
Train ROC curve (area = 1.00)
Test ROC curve (area = 0.91)
True Positive Rate
False Positive Rate