Project 2. Machine Learning (STA 5365). Aditya Ranjan Bhattacharya (arb17b) and Matthew Laird (mrl14b)

For Project 2 we used RandomForestClassifier model from sklearn[1] as our RandomForest model. In this case, we used 2 separate functions data_to_numpy() and label_to_numpy(), which loads the features and labels from the files to a numpy array. We load the Madelon dataset using these 2 functions.

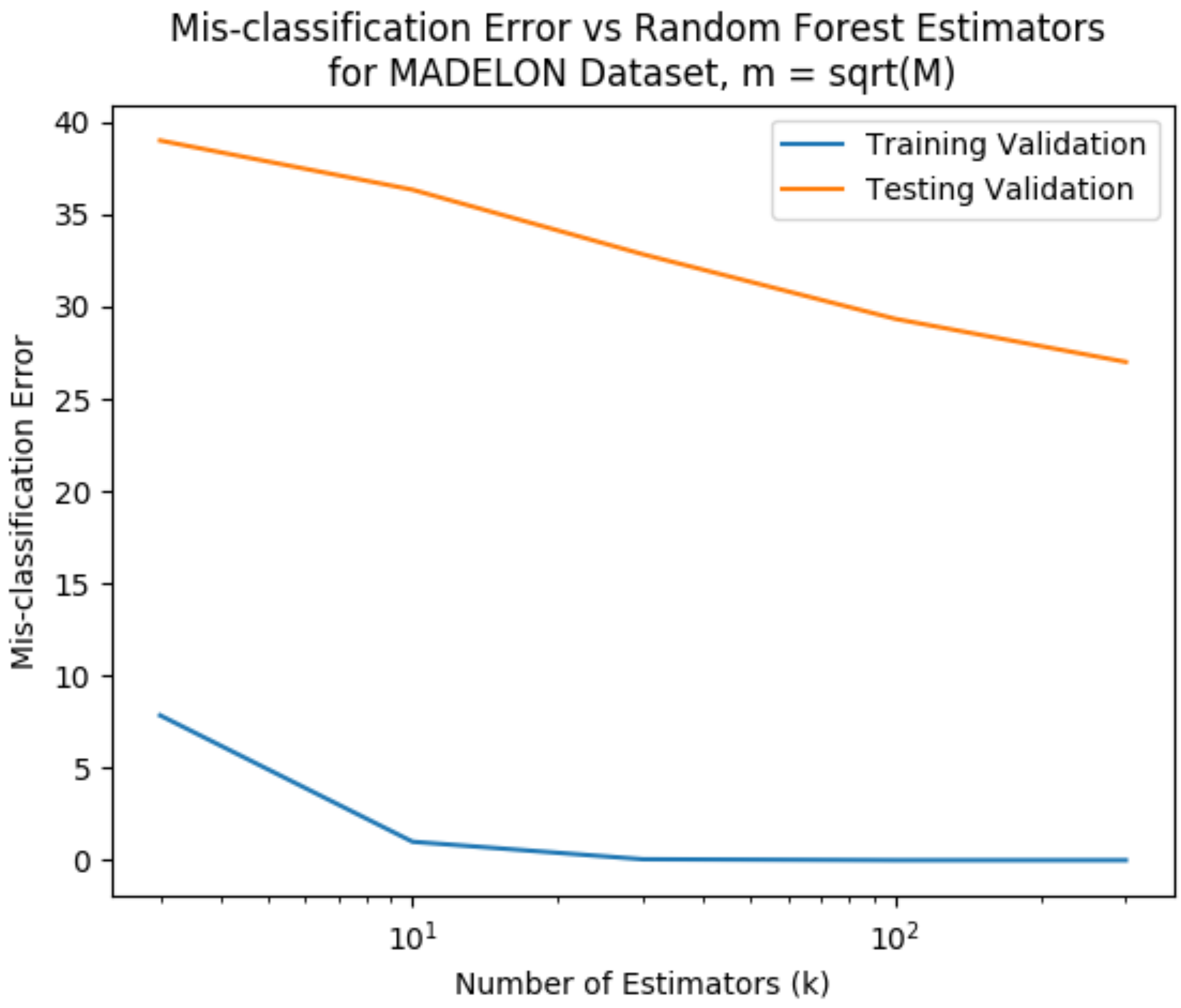
Once the dataset is loaded, we train 3 RandomForestClassifiers with the data, each having a subset of the features of the Madelon to consider. The first model considers a randomized sample of $\sqrt{M}$ features, where the total number of features of Madelon is M. (Since Madelon has 500 features, the first model considers $\sqrt{500}$ features. The second model considers randomized sample of $\lg M$ features, while the final model takes in all of the features of the dataset. For each of these models, we vary the number of trees (k) from 3,10,30,100,300.

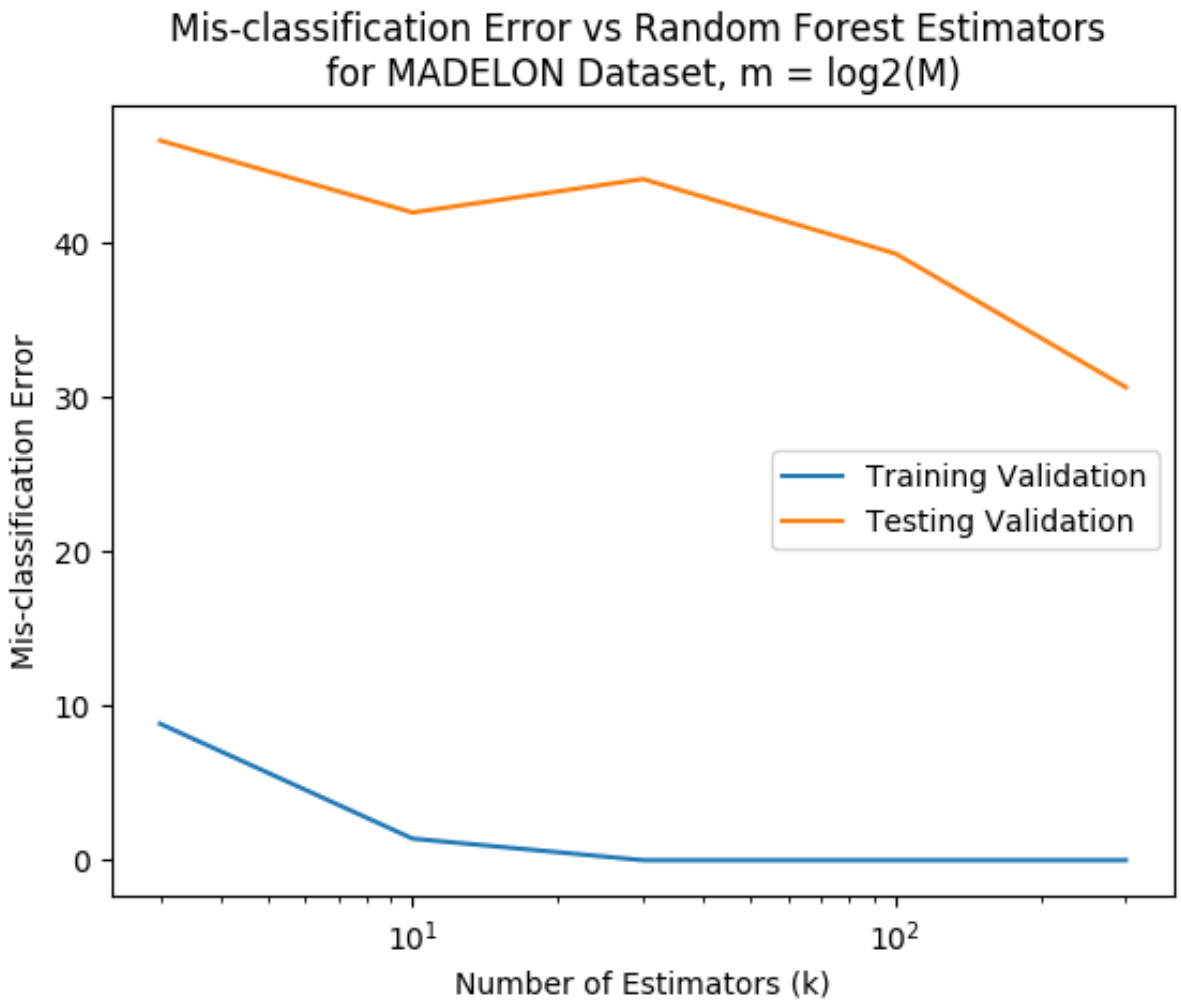
The training and testing validation for each of these models for a particular k, are calculated, and we plot them in a graph, as shown in the next 3 pages. The table below shows the mis-classification error of each model for each k.

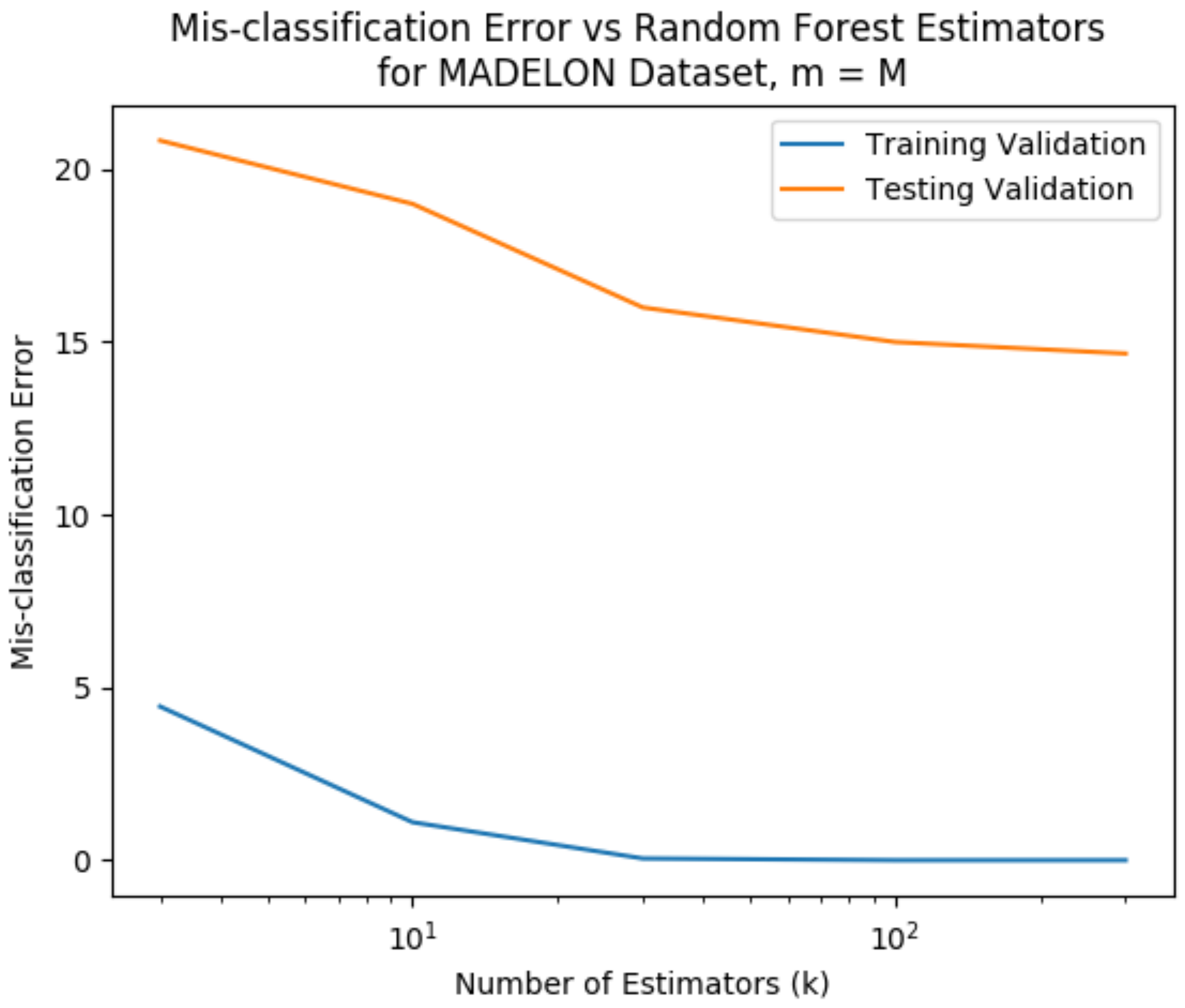
Min. Test Classification Error and Related Tree Depth For Datasets

Features Taken By Model (M = num_features)	Mis-Classification Error for K = 3	Mis-Classification Error for K =10	Mis-Classification Error for K = 30	Mis-Classification Error for K = 100	Mis-Classification Error for K = 300
sqrt(M)	39	36.33	32.83	29.33	27
lg(M)	46.67	42	44.17	39.33	30.67
M	20.83	19	16	15	14.67

[1] Lars Buitinck et. al. API design for machine learning software: experiences from the scikit-learn project, ECML PKDD Workshop: Languages for Data Mining and Machine Learning, 2013, pages (108-122)


Mis-classification Error vs Random Forest Estimators
for MADELON Dataset, m = sqrt(M)
Mis-classification Error
Number of Estimators (k)
Training Validation
Testing Validation
40
35
30
25
20
15
10
5
0
10^1
10^2


Mis-classification Error vs Random Forest Estimators
for MADELON Dataset, m = log2(M)
Mis-classification Error
0
10
20
30
40
10^1
10^2
Number of Estimators (k)
Training Validation
Testing Validation


Mis-classification Error vs Random Forest Estimators
for MADELON Dataset, m = M
Training Validation
Testing Validation
Mis-classification Error
0
5
10
15
20
10^1
10^2
Number of Estimators (k)

```
from sklearn.ensemble import RandomForestClassifier
import numpy
import matplotlib.pyplot as plt
from numpy import genfromtxt

def data_to_numpy(file, num_features):

    if "wilt" in file:
        #wilt is in .csv, hence we use numpy.genfromtxt which loads csv to a numpy array
faster and easily.
        X = genfromtxt(file, delimiter=',')
        print(X.shape)
        return X

    else:

        X = numpy.empty([0,num_features])

        with open(file, 'r') as f:
            content = f.readlines()

            for line in content:
                row = numpy.empty(0)

                for x in line.strip("\n,\'").split():
                    row = numpy.append(row, float(x))

                X = numpy.append(X, [row], 0)
            print(X.shape)

        return X
#Labels Extracted From .label files.
def label_to_numpy(file):

    y = numpy.empty(0)

    with open(file, 'r') as f:
        content = f.readlines()

        for line in content:
            y = numpy.append(y, int(line))

        print(y.shape)

    return y


k_list = [3,10,30,100,300]

scores_train = numpy.empty(0)
scores_test = numpy.empty(0)

X_train = data_to_numpy("../MADELON/madelon_train.data", 500)
y_train = label_to_numpy("../MADELON/madelon_train.labels")

X_test = data_to_numpy("../MADELON/madelon_valid.data", 500)
y_test = label_to_numpy("../MADELON/madelon_valid.labels")

print("m = sqrt(M)")

for k in k_list:

    model = RandomForestClassifier(n_estimators=k,
                                    criterion='gini',
                                    max_depth=None,
                                    min_samples_split=2,
                                    min_samples_leaf=1,
                                    min_weight_fraction_leaf=0.0,
                                    max_features='sqrt',
                                    max_leaf_nodes=None,
                                    min_impurity_decrease=0.0,
                                    min_impurity_split=None,
                                    bootstrap=True,
                                    oob_score=False,
                                    n_jobs=1,
                                    random_state=None,
                                    verbose=0,
                                    warm_start=False,
                                    class_weight=None)
    model.fit(X_train, y_train)

    training_scores = model.score(X_train, y_train)
    test_scores = model.score(X_test, y_test)

        #Error for DT with depth 1 will be at index 0. scores_train[0] gives efficiency of DT
with depth 1. Also, remember scores is accuracy
    scores_train = numpy.append(scores_train, (1 - training_scores)*100.00)
    scores_test = numpy.append(scores_test, (1 - test_scores)*100.00)

    print("For k = ", k, "mis-classification error is: ", (1 - test_scores)*100.00)

fig = plt.figure()
plt.semilogx(k_list, scores_train, label='Training Validation')
plt.semilogx(k_list, scores_test, label='Testing Validation')
plt.xlabel('Number of Estimators (k)')
plt.ylabel('Mis-classification Error')
plt.title("Mis-classification Error vs Random Forest Estimators \nfor MADELON Dataset, m =
sqrt(M)")
plt.legend()
fig.savefig("hw2_MADELON_sqrt.png")

scores_train = numpy.empty(0)
scores_test = numpy.empty(0)

print("m = log2(M)")

for k in k_list:

    model = RandomForestClassifier(n_estimators=k,
                                    criterion='gini',
                                    max_depth=None,
                                    min_samples_split=2,
                                    min_samples_leaf=1,
                                    min_weight_fraction_leaf=0.0,
                                    max_features='log2',
                                    max_leaf_nodes=None,
                                    min_impurity_decrease=0.0,
                                    min_impurity_split=None,
                                    bootstrap=True,
                                    oob_score=False,
                                    n_jobs=1,
                                    random_state=None,
                                    verbose=0,
                                    warm_start=False,
                                    class_weight=None)
    model.fit(X_train, y_train)

    training_scores = model.score(X_train, y_train)
    test_scores = model.score(X_test, y_test)

        #Error for DT with depth 1 will be at index 0. scores_train[0] gives efficiency of DT
with depth 1. Also, remember scores is accuracy
    scores_train = numpy.append(scores_train, (1 - training_scores)*100.00)
    scores_test = numpy.append(scores_test, (1 - test_scores)*100.00)

    print("For k = ", k, "mis-classification error is: ", (1 - test_scores)*100.00)

fig = plt.figure()
plt.semilogx(k_list, scores_train, label='Training Validation')
plt.semilogx(k_list, scores_test, label='Testing Validation')
plt.xlabel('Number of Estimators (k)')
plt.ylabel('Mis-classification Error')
plt.title("Mis-classification Error vs Random Forest Estimators \nfor MADELON Dataset, m =
log2(M)")
plt.legend()
fig.savefig("hw2_MADELON_log2.png")


scores_train = numpy.empty(0)
scores_test = numpy.empty(0)

print("m = M")

for k in k_list:

    model = RandomForestClassifier(n_estimators=k,
                                    criterion='gini',
                                    max_depth=None,
                                    min_samples_split=2,
                                    min_samples_leaf=1,
                                    min_weight_fraction_leaf=0.0,
                                    max_features=None,
                                    max_leaf_nodes=None,
                                    min_impurity_decrease=0.0,
                                    min_impurity_split=None,
                                    bootstrap=True,
                                    oob_score=False,
                                    n_jobs=1,
                                    random_state=None,
                                    verbose=0,
                                    warm_start=False,
                                    class_weight=None)
    model.fit(X_train, y_train)

    training_scores = model.score(X_train, y_train)
    test_scores = model.score(X_test, y_test)

        #Error for DT with depth 1 will be at index 0. scores_train[0] gives efficiency of DT
with depth 1. Also, remember scores is accuracy
    scores_train = numpy.append(scores_train, (1 - training_scores)*100.00)
    scores_test = numpy.append(scores_test, (1 - test_scores)*100.00)

    print("For k = ", k, "mis-classification error is: ", (1 - test_scores)*100.00)

fig = plt.figure()
plt.semilogx(k_list, scores_train, label='Training Validation')
plt.semilogx(k_list, scores_test, label='Testing Validation')
plt.xlabel('Number of Estimators (k)')
plt.ylabel('Mis-classification Error')
plt.title("Mis-classification Error vs Random Forest Estimators \nfor MADELON Dataset, m = M")
plt.legend()
fig.savefig("hw2_MADELON_M.png")
```