In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.model_selection import LeaveOneOut
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import mean_squared_error
```

In [2]:
```
df = pd.read_csv('Boston.csv')
df_copy = df.copy()
df_copy = df_copy.drop('Unnamed: 0', 1)
# crim = per capita crime rate by town
# zn = proportion of residential land zoned for lots over 25,000 sq.ft.
# INDUS - proportion of non-retail business acres per town.
# CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
# NOX - nitric oxides concentration (parts per 10 million)
# RM - average number of rooms per dwelling
# AGE - proportion of owner-occupied units built prior to 1940
# DIS - weighted distances to five Boston employment centres
# RAD - index of accessibility to radial highways
# TAX - full-value property-tax rate per $10,000
# PTRATIO - pupil-teacher ratio by town
# B - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
# LSTAT - % lower status of the population
# MEDV - Median value of owner-occupied homes in $1000's

medv_median = df_copy['medv'].median()
df_copy.loc[df_copy['medv'] >= medv_median, 'medv01'] = 1
df_copy.loc[df_copy['medv'] < medv_median, 'medv01'] = 0
df_copy = df_copy.drop('medv', 1)
df_copy.head()
```

```
C:\Users\ADAMYA~1\AppData\Local\Temp/ipykernel_19012/3080220384.py:3: FutureWarning: In a future version of pandas all ar
guments of DataFrame.drop except for the argument 'labels' will be keyword-only
  df_copy = df_copy.drop('Unnamed: 0', 1)
```

```
C:\Users\ADAMYA~1\AppData\Local\Temp/ipykernel_19012/3080220384.py:22: FutureWarning: In a future version of pandas all a
rguments of DataFrame.drop except for the argument 'labels' will be keyword-only
  df_copy = df_copy.drop('medv', 1)
```

Out[2]:

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv01
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	1.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	1.0
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	1.0
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	1.0
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	1.0

25 Trees

In [3]:

```
y = df_copy['medv01']
X = df_copy.drop('medv01', 1)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)

clf = RandomForestClassifier(n_estimators=25)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_test, y_pred).ravel()
print("Number of True Positives: " + repr(num_TP))
print("Number of False Negatives: " + repr(num_FN))
print("Number of False Positives: " + repr(num_FP))
print("Number of True Negatives: " + repr(num_TN))
```

```
Number of True Positives: 42
Number of False Negatives: 4
Number of False Positives: 6
Number of True Negatives: 50
C:\Users\ADAMYA~1\AppData\Local\Temp/ipykernel_19012/111937910.py:2: FutureWarning: In a future version of pandas all arg
uments of DataFrame.drop except for the argument 'labels' will be keyword-only
  X = df_copy.drop('medv01', 1)
```

500 Trees

In [4]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)

clf = RandomForestClassifier(n_estimators=500)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

num_TN, num_FP, num_FN, num_TP = confusion_matrix(y_test, y_pred).ravel()
print("Number of True Positives: " + repr(num_TP))
print("Number of False Negatives: " + repr(num_FN))
print("Number of False Positives: " + repr(num_FP))
print("Number of True Negatives: " + repr(num_TN))
```

```
Number of True Positives: 45
Number of False Negatives: 7
Number of False Positives: 10
Number of True Negatives: 40
```

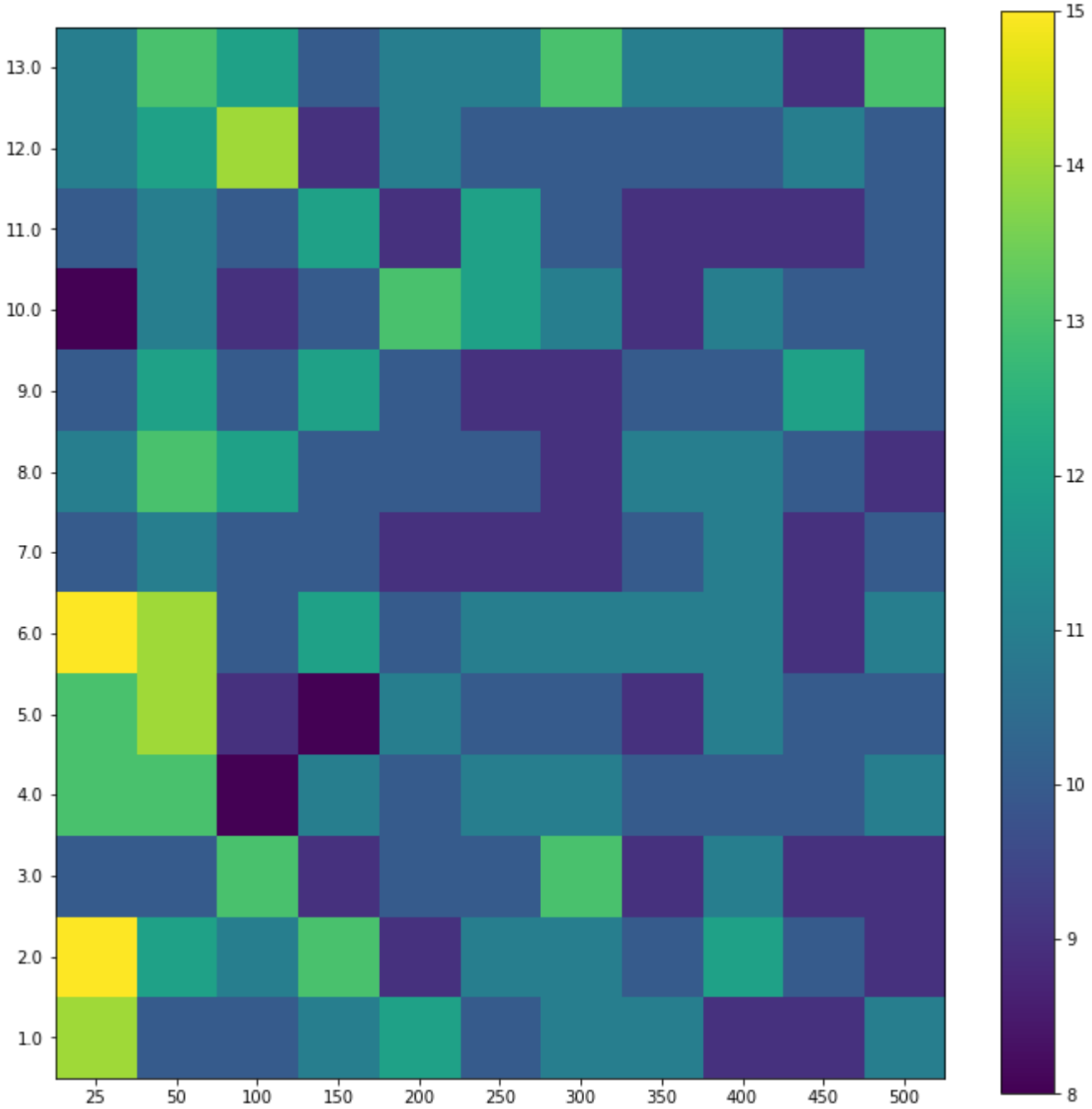
Different Number of Trees and Predictors

In [5]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)

test_error_array = []
num_trees_list = [25, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500]
for i in range(13):
    test_errors = []
    for trees in (num_trees_list):
        y_curr = y
        X_curr = X.iloc[:,:(i+1)]

        clf = RandomForestClassifier(n_estimators=trees)
        clf.fit(X_train, y_train)
        y_pred = clf.predict(X_test)
        curr_error = np.sum((y_test - y_pred)**2)

        test_errors.append(curr_error)

    test_error_array.append(test_errors)
```

In [12]:
```
fig = plt.figure(figsize=(12, 12))
plt.imshow(test_error_array)
```

```
ax = plt.gca()
ax.invert_yaxis()
ax.set_yticklabels(np.linspace(1,13,13))
ax.set_xticklabels(num_trees_list)
ax.set_xticks(np.linspace(0, 10, 11))
ax.set_yticks(np.linspace(0, 12, 13))
plt.colorbar()
plt.show()
```

```
C:\Users\ADAMYA~1\AppData\Local\Temp/ipykernel_19012/633484942.py:5: UserWarning: FixedFormatter should only be used toge
ther with FixedLocator
  ax.set_yticklabels(np.linspace(1,13,13))
C:\Users\ADAMYA~1\AppData\Local\Temp/ipykernel_19012/633484942.py:6: UserWarning: FixedFormatter should only be used toge
ther with FixedLocator
  ax.set_xticklabels(num_trees_list)
```


13.0
12.0
11.0
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
25
50
100
150
200
250
300
350
400
450
500
15
14
13
12
11
10
9
8

In [ ]:

In [ ]: