Name: Joyce Le
ID#: 82549113
UCINet ID: lejy

CS 178 Homework 1

Problem 1

1)

In [104]:

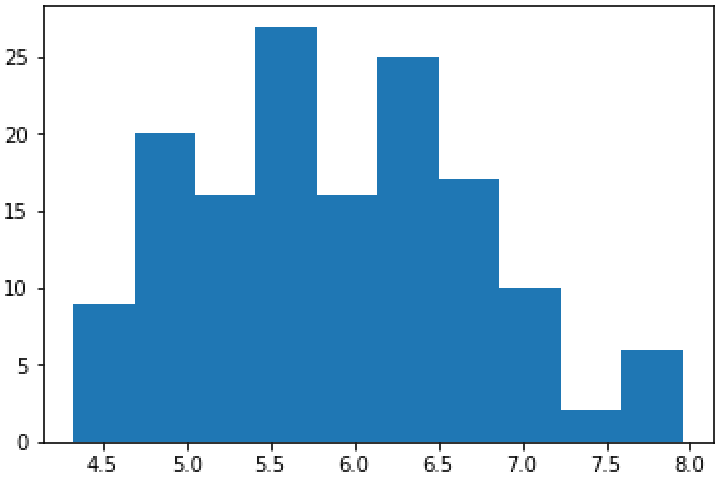
```
import numpy as np
import matplotlib.pyplot as plt

iris = np.genfromtxt("data/iris.txt",delimiter=None) # load the text file
Y = iris[:,-1] # target value (iris species) is the last column
X = iris[:,0:-1] # features are the other columns

print("Number of features=", X.shape[1])
print("Number of data points=", X.shape[0])
```

```
Number of features= 4
Number of data points= 148
```

2)

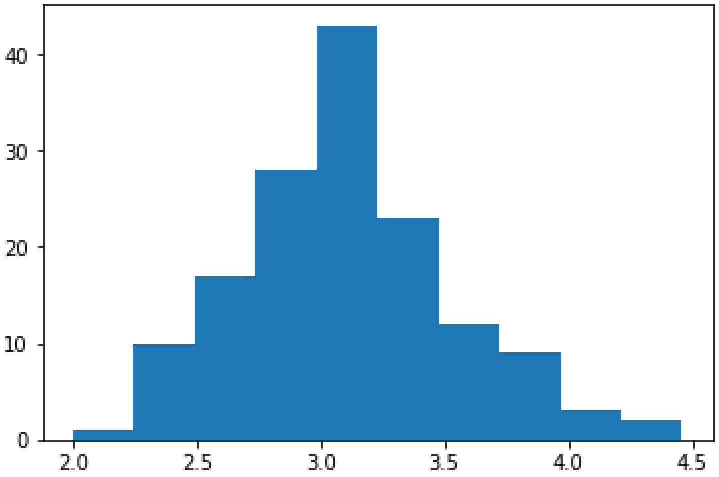
In [105]:
```
print("FEATURE 1\n")
print(plt.hist(X[:, 0]))
```

```
FEATURE 1

(array([ 9., 20., 16., 27., 16., 25., 17., 10.,  2.,  6.]), array([4.326557
9 , 4.68918546, 5.05181302, 5.41444058, 5.77706814,
       6.1396957 , 6.50232326, 6.86495082, 7.22757838, 7.59020594,
       7.9528335 ]), <a list of 10 Patch objects>)
```



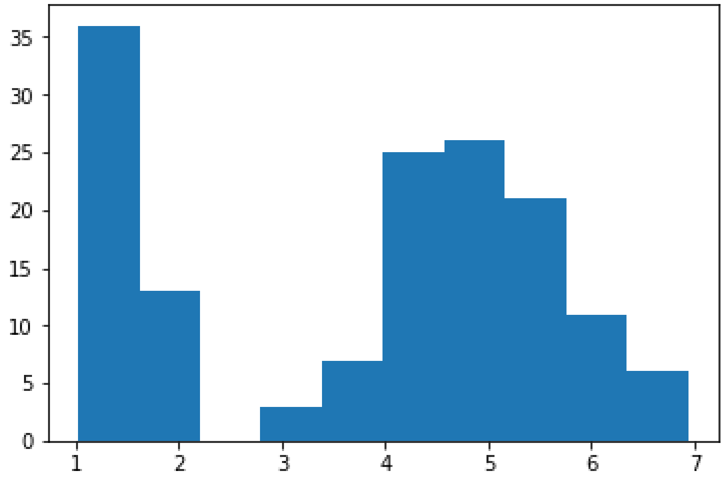
In [106]:
```
print("FEATURE 2\n")
print(plt.hist(X[:, 1]))
```

```
FEATURE 2

(array([ 1., 10., 17., 28., 43., 23., 12.,  9.,  3.,  2.]), array([2.002045
4 , 2.24764816, 2.49325092, 2.73885368, 2.98445644,
       3.2300592 , 3.47566196, 3.72126472, 3.96686748, 4.21247024,
       4.458073  ]), <a list of 10 Patch objects>)
```



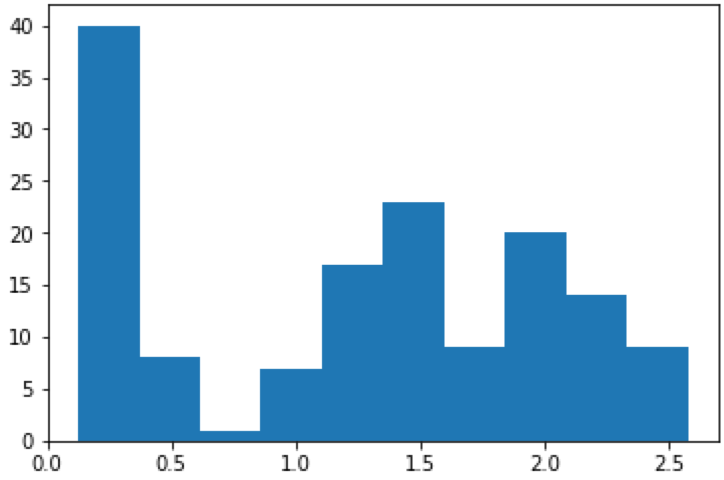
In [107]:
```
print("FEATURE 3\n")
print(plt.hist(X[:, 2]))
```

```
FEATURE 3

(array([36., 13.,  0.,  3.,  7., 25., 26., 21., 11.,  6.]), array([1.023831
6 , 1.61492121, 2.20601082, 2.79710043, 3.38819004,
       3.97927965, 4.57036926, 5.16145887, 5.75254848, 6.34363809,
       6.9347277 ]), <a list of 10 Patch objects>)
```



In [108]:
```
print("FEATURE 4\n")
print(plt.hist(X[:, 3]))
```

```
FEATURE 4

(array([40.,  8.,  1.,  7., 17., 23.,  9., 20., 14.,  9.]), array([0.125090
37, 0.37029368, 0.615497  , 0.86070031, 1.10590362,
       1.35110693, 1.59631025, 1.84151356, 2.08671687, 2.33192019,
       2.5771235 ]), <a list of 10 Patch objects>)
```



3)

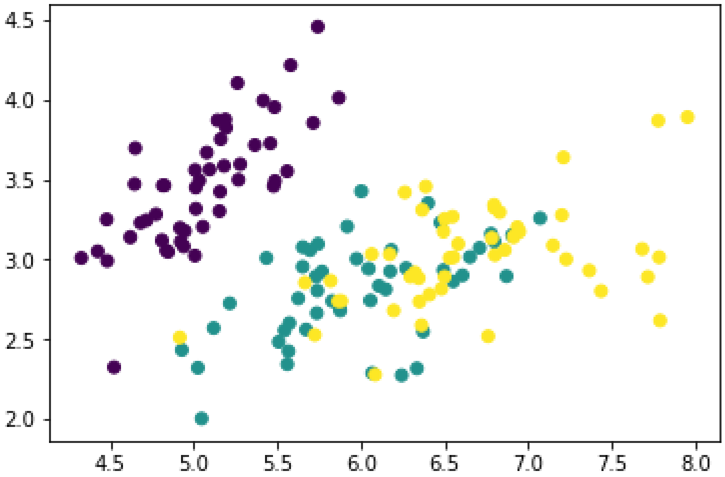
In [109]:
```
for i in range(4):
    print("Feature", i + 1, ":  mean =", np.mean(X[:, i]), ", std =", np.st
d(X[:, i]))
```

```
Feature 1 :  mean = 5.900103764189188 , std = 0.833402066774894
Feature 2 :  mean = 3.098930916891892 , std = 0.43629183800107685
Feature 3 :  mean = 3.8195548405405404 , std = 1.7540571093439352
Feature 4 :  mean = 1.2525554845945945 , std = 0.7587724570263247
```

4)

In [110]:
```
print("FEATURE (1,2)\n")
print(plt.scatter(X[:, 0], X[:, 1], c = iris[:, -1]))
```

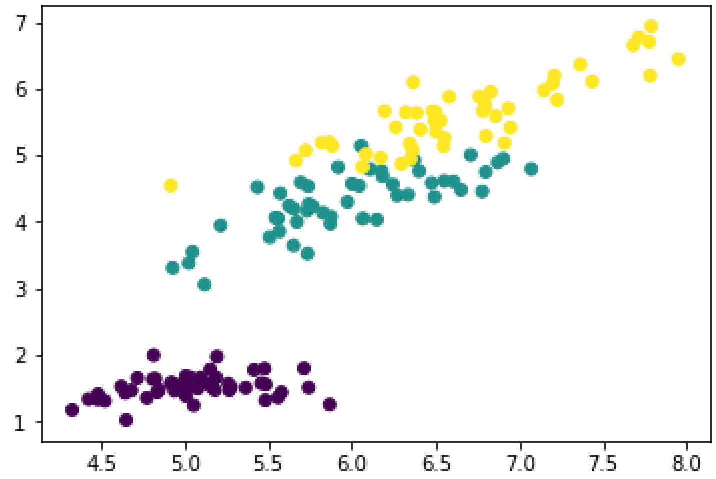
```
FEATURE (1,2)

<matplotlib.collections.PathCollection object at 0x0000016BA013DEF0>
```



In [111]:

```
print("FEATURE (1,3)\n")
print(plt.scatter(X[:, 0], X[:, 2], c = iris[:, -1]))
```

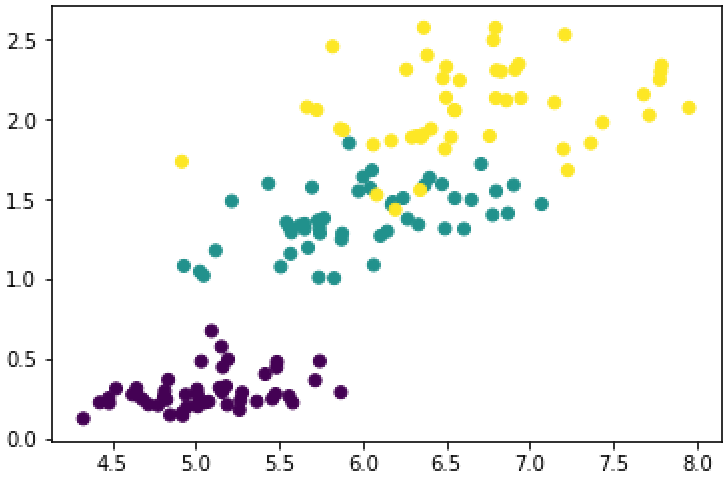
```
FEATURE (1,3)

<matplotlib.collections.PathCollection object at 0x0000016BA00AA668>
```



In [112]:

```
print("FEATURE (1,4)\n")
print(plt.scatter(X[:, 0], X[:, 3], c = iris[:, -1]))
```

```
FEATURE (1,4)

<matplotlib.collections.PathCollection object at 0x0000016BA03B6AC8>
```



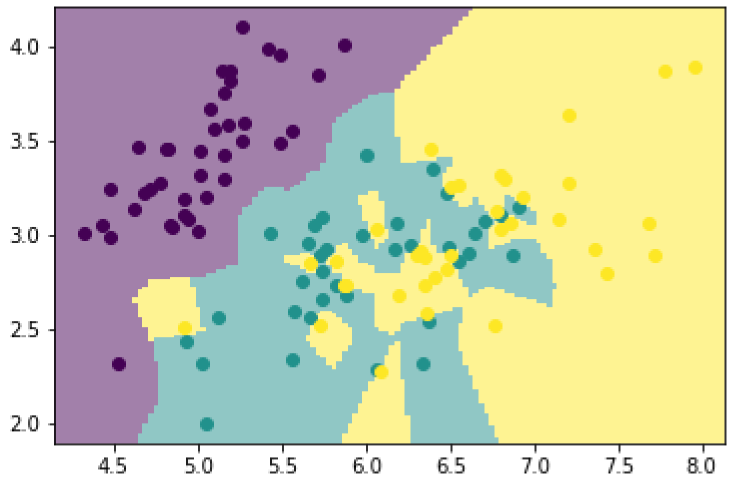
Problem 2

In [113]:
```
X_2col = iris[:, 0:-3]

import mltools as ml
# We'll use some data manipulation routines in the provided class code
# Make sure the "mltools" directory is in a directory on your Python path,
 e.g.,
# export PYTHONPATH=$\$${PYTHONPATH}:/path/to/parent/dir
# or add it to your path inside Python:
# import sys
# sys.path.append('/path/to/parent/dir/');

np.random.seed(0) # set the random number seed
X_2col,Y = ml.shuffleData(X_2col,Y); # shuffle data randomly
# (This is a good idea in case your data are ordered in some systematic wa
y.)

Xtr,Xva,Ytr,Yva = ml.splitData(X_2col,Y, 0.75); # split data into 75/25 tra
in/validation

knn = ml.knn.knnClassify() # create the object and train it
knn.train(Xtr, Ytr, 1) # where K is an integer, e.g. 1 for nearest neighbor
 prediction
YvaHat = knn.predict(Xva) # get estimates of y for each data point in Xva
```

1)

In [114]:
```
print("k = 1:")
ml.plotClassify2D( knn, Xtr, Ytr ); # make 2D classification plot with data
 (Xtr,Ytr)
```

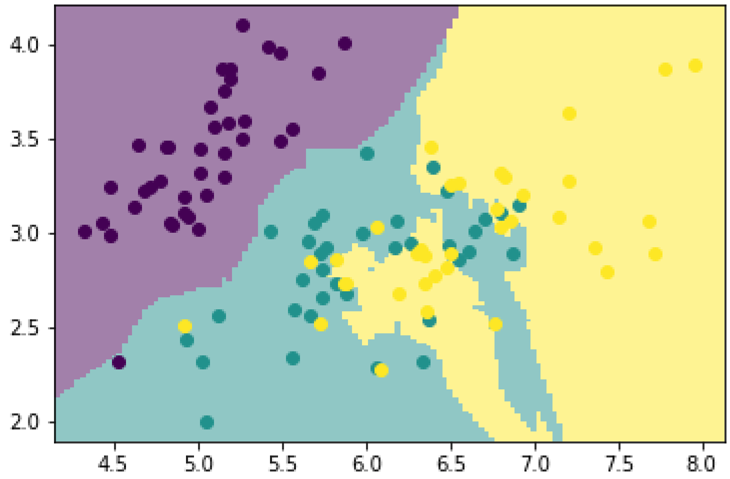
```
k = 1:
```



In [115]:

```
knn.train(Xtr, Ytr, 5) # where K is an integer, e.g. 1 for nearest neighbor prediction
print("k = 5:")
ml.plotClassify2D( knn, Xtr, Ytr ); # make 2D classification plot with data (Xtr,Ytr)
```

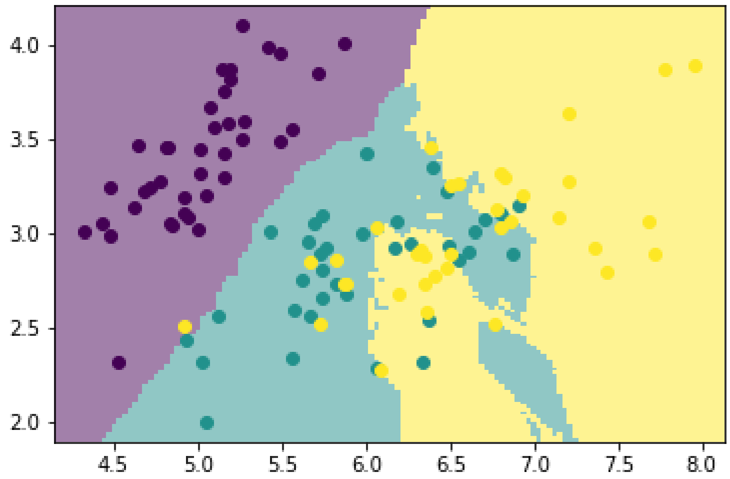
```
k = 5:
```



In [116]:

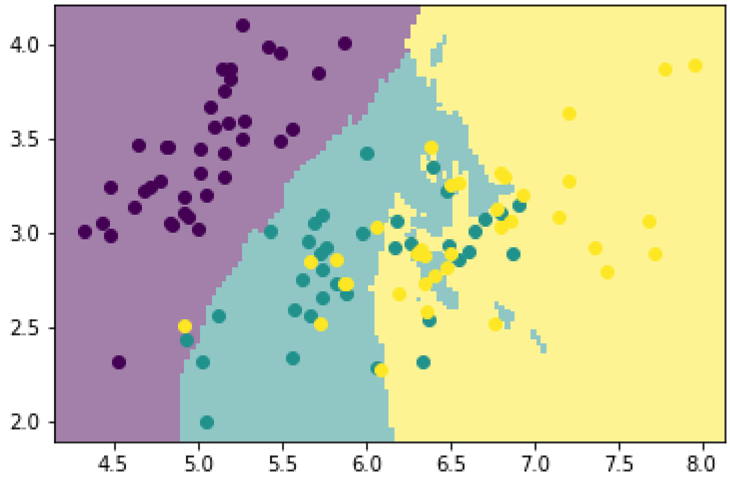
```
knn.train(Xtr, Ytr, 10) # where K is an integer, e.g. 1 for nearest neighbor prediction
print("k = 10:")
ml.plotClassify2D( knn, Xtr, Ytr ); # make 2D classification plot with data (Xtr,Ytr)
```

```
k = 10:
```



In [117]:
```
knn.train(Xtr, Ytr, 20) # where K is an integer, e.g. 1 for nearest neighbor prediction
print("k = 20:")
ml.plotClassify2D( knn, Xtr, Ytr ); # make 2D classification plot with data (Xtr,Ytr)
```

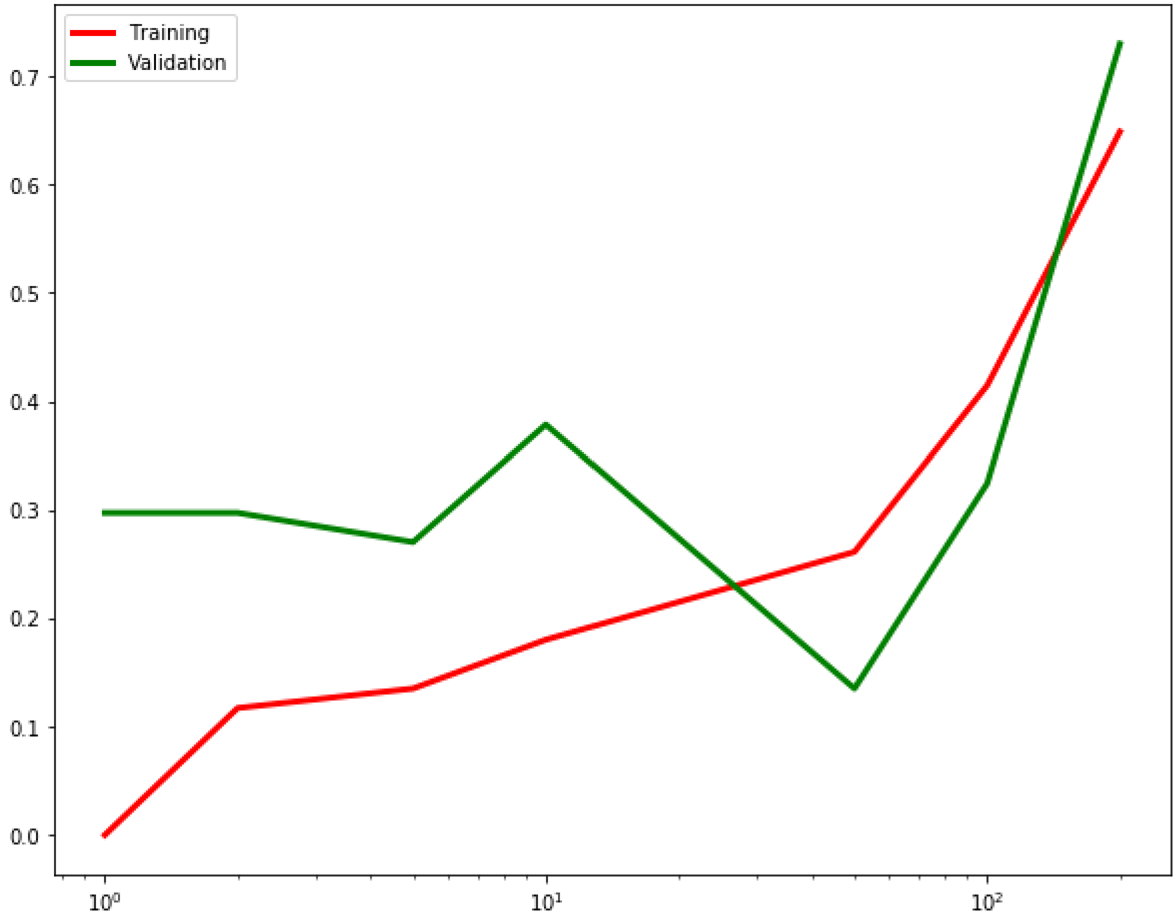
```
k = 20:
```



2)

In [118]:

```
K=[1,2,5,10,50,100,200];
errTrain = [None]*len(K)
errVal = [None]*len(K)
for i,k in enumerate(K):
    learner = ml.knn.knnClassify(Xtr, Ytr, k)
    Yhat = learner.predict(Xtr)
    errTrain[i] = np.mean(Yhat != Ytr)
    YvaHat = learner.predict(Xva)
    errVal[i] = np.mean(YvaHat != Yva)

fig, ax = plt.subplots(1, 1, figsize=(10, 8))
ax.semilogx(K, errTrain, 'r-', lw=3, label='Training')
ax.semilogx(K, errVal, 'g-', lw=3, label='Validation')

ax.legend()
plt.show()
```

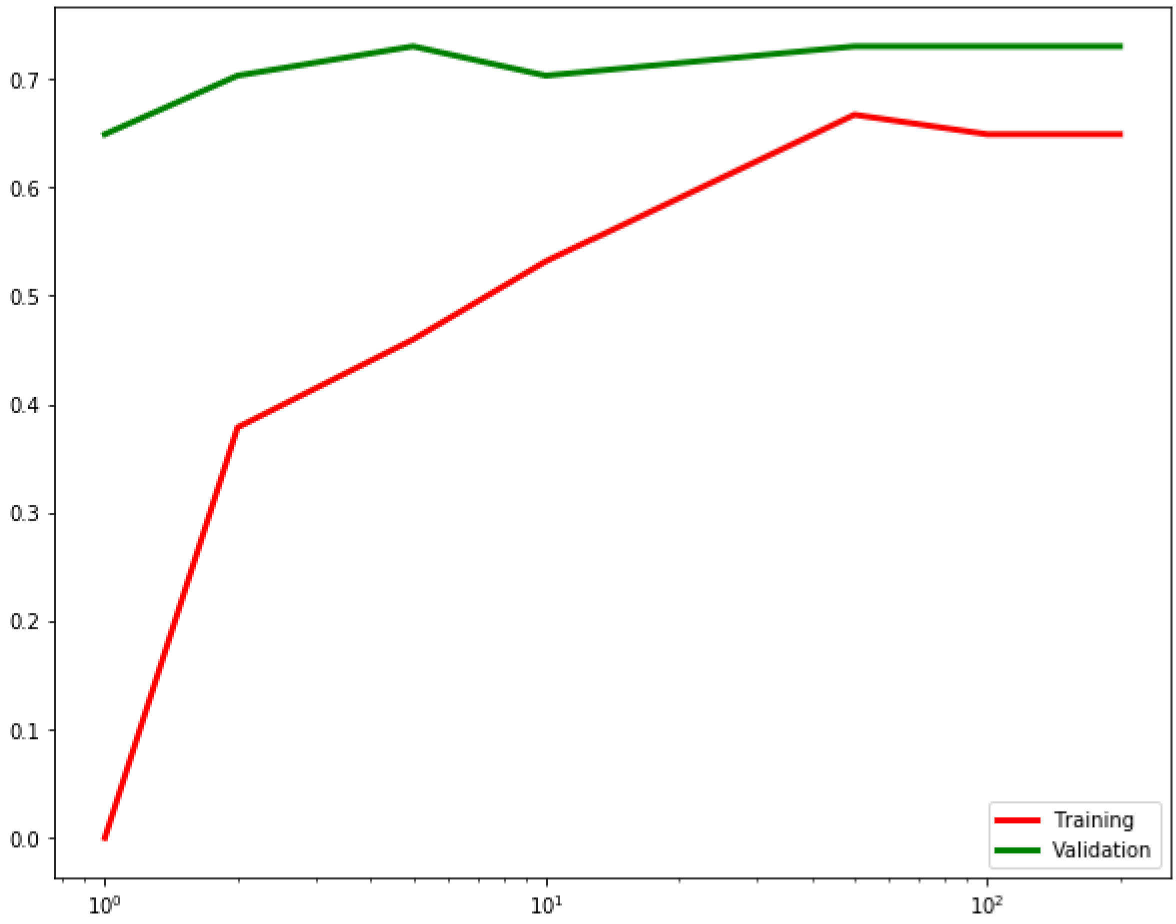


Based on this graph I would choose K=5.

3)

In [119]:

```
Xtr,Xva,Ytr,Yva = ml.splitData(X,Y, 0.75);
K=[1,2,5,10,50,100,200];
errTrain = [None]*len(K)
errVal = [None]*len(K)
for i,k in enumerate(K):
    learner = ml.knn.knnClassify(Xtr, Ytr, k)
    Yhat = learner.predict(Xtr)
    errTrain[i] = np.mean(Yhat != Ytr)
    YvaHat = learner.predict(Xva)
    errVal[i] = np.mean(YvaHat != Yva)

fig, ax = plt.subplots(1, 1, figsize=(10, 8))
ax.semilogx(K, errTrain, 'r-', lw=3, label='Training')
ax.semilogx(K, errVal, 'g-', lw=3, label='Validation')

ax.legend()
plt.show()
```



The graphs are very different because they now look like rough inverses of each other. I would not pick a different K.

Problem 3

1)

P(y=1) = 4/10
P(y=-1) = 6/10

P(x1=1 | y= -1) = 3/6
P(x2=1 | y= -1) = 5/6
P(x3=1 | y= -1) = 4/6
P(x4=1 | y= -1) = 5/6
P(x5=1 | y= -1) = 2/6

P(x1=1 | y=1) = 3/4
P(x2=1 | y=1) = 0
P(x3=1 | y=1) = 3/4
P(x4=1 | y=1) = 2/4
P(x5=1 | y=1) = 1/4

2)

For x = (0 0 0 0 0),
P(y= -1 | x) = (1- 3/6) *(1- 5/6)* (1- 4/6) *(1- 5/6)* (1- 2/6) *6/10* =
3/6 1/6 *2/6* 1/6 *4/6* 6/10 = 0.00185
P(y=1 | x) = (1- 3/4) *(1-0)* (1- 3/4) *(1- 2/4)* (1- 1/4) *4/10* =
1/4 1 *1/4* 2/4 *3/4* 4/10 = 0.00938
Since 0.00938 > 0.00185, y would be predicted as +1 for x.

For x = (1 1 0 1 0), since we know from above that P(x2=1 | y=1) = 0, we know that y would be predicted as -1. Since x2 has the value 1, and the probility of x2 being 1 while y is also 1 is 0, y has to be -1.

3)

P(y=1 | x=(0 0 0 0 0)) = 0.00185 (calculated above)
P(y=1 | x=(1 1 0 1 0)) = 0

4)

We should probably not use a joint Bayes classifier because there are too many x variables that determine y for it to be convenient. Being able to assume independence simplifies this problem dramatically.

5)

No, we would not need to retrain the model.

Problem 4: Statement of Collaboration

I started pretty late on this homework assignment and don't really have any close friends in this class, so I did not discuss this homework in person or online with anyone. The only collaborative help I got was through looking at Campuswire.