Problem 1

In [1]:
```
import numpy as np
```

In [2]:
```
import matplotlib.pyplot as plt
```

In [3]:
```
iris = np.genfromtxt("data/iris.txt",delimiter=None) # load the text file
```

In [4]:
```
Y = iris[:,-1] # target value (iris species) is the last column
```

In [5]:
```
X = iris[:,0:-1] # features are the other columns
```

Problem 1.1

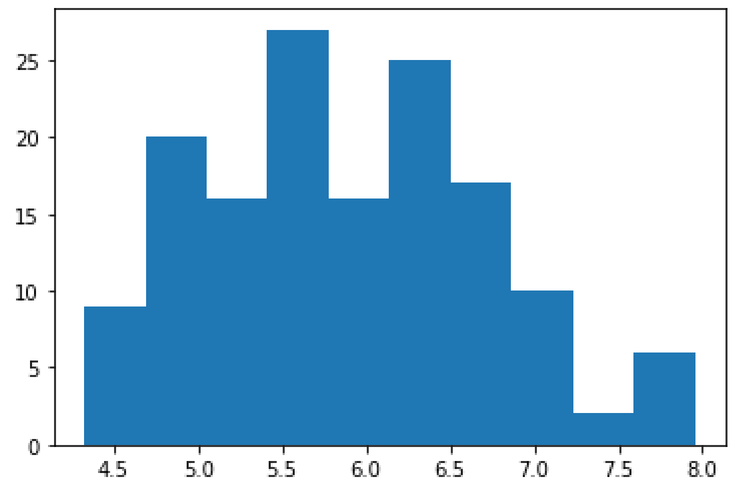
In [6]:
```
X.shape
```

Out[6]:
```
(148, 4)
```

There are 148 data points which is the number of rows and 4 features which is the number of columns of X.

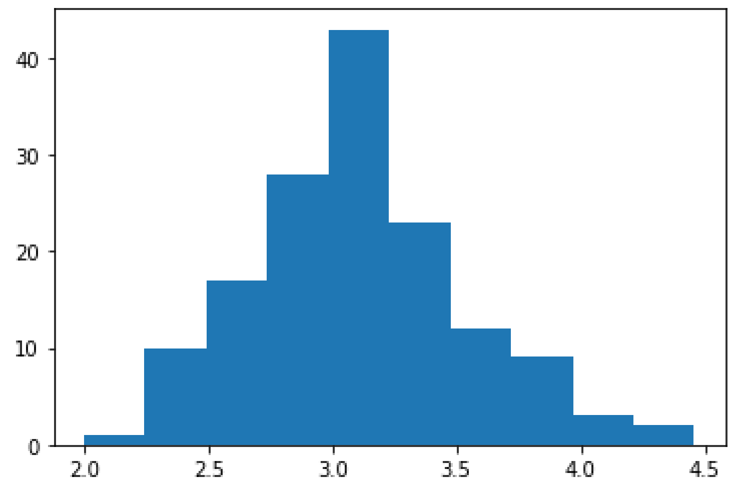
Problem 1.2

In [7]:
```
plt.hist(X[:,0])
plt.show()
```



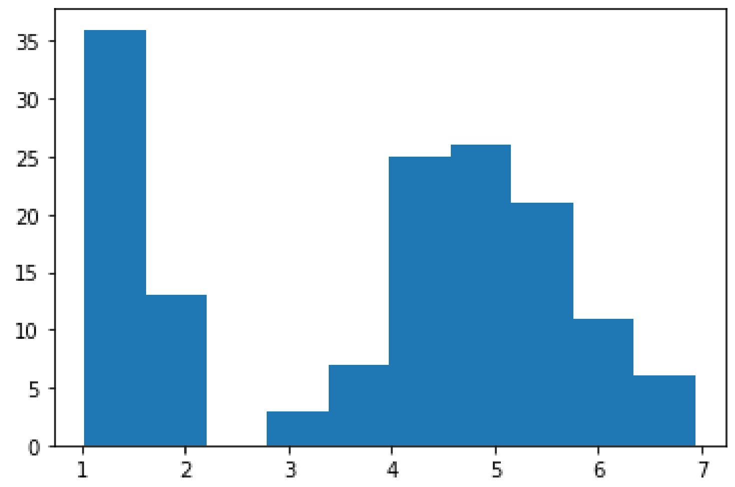
Histagram of the first feature

In [8]:
```
plt.hist(X[:,1])
plt.show()
```



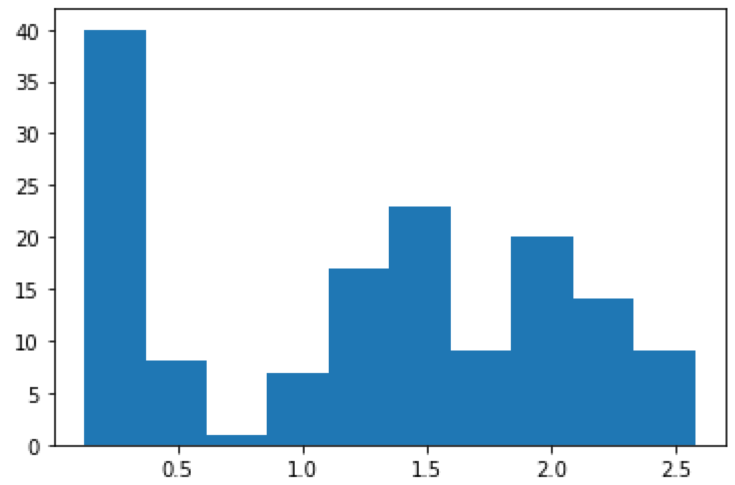
Histagram of the second feature

In [9]:
```
plt.hist(X[:,2])
plt.show()
```



Histagram of the third feature

In [10]:
```
plt.hist(X[:,3])
plt.show()
```



Histagram of the fourth feature

Problem 1.3

In [11]:
```
print("The mean of the first feature is", np.mean(X[:,0]), ", and the standard deviation of it is", np.std(X[:,0]))
```

```
The mean of the first feature is 5.900103764189188 , and the standard deviation of it is 0.833402066774894
```

In [12]:
```
print("The mean of the second feature is", np.mean(X[:,1]), ", and the standard deviation of it is", np.std(X[:,1]))
```

```
The mean of the second feature is 3.098930916891892 , and the standard deviation of it is 0.43629183800107685
```

In [13]:
```
print("The mean of the third feature is", np.mean(X[:,2]), ", and the standard deviation of it is", np.std(X[:,2]))
```

```
The mean of the third feature is 3.8195548405405404 , and the standard deviation of it is 1.7540571093439352
```

In [14]:
```
print("The mean of the fourth feature is", np.mean(X[:,3]), ", and the standard deviation of it is", np.std(X[:,3]))
```

```
The mean of the fourth feature is 1.2525554845945945 , and the standard deviation of it is 0.7587724570263247
```

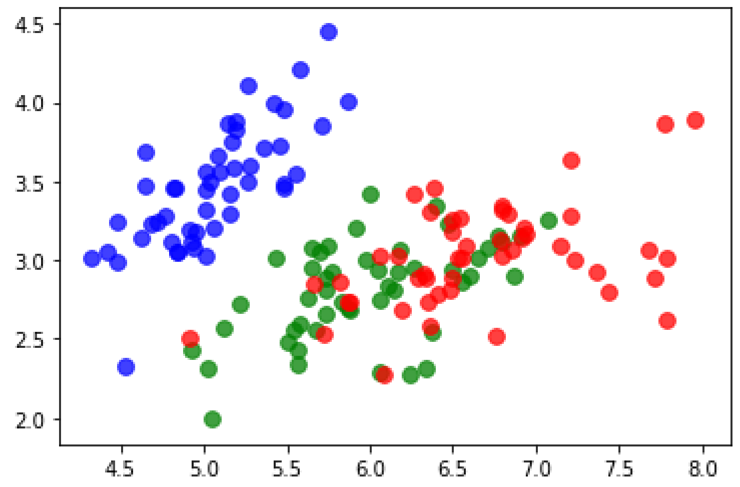
Problem 1.4

Feature (1,2)

In [15]:
```
colors = ['blue', 'green', 'red']

for i, c in enumerate(np.unique(iris[:, -1])):
    mask = iris[:, -1] == c  # Finding the right points
    plt.scatter(iris[mask, 0], iris[mask, 1], s=60, c=colors[i], alpha=0.75, label='class %d' % i)

plt.show()
```

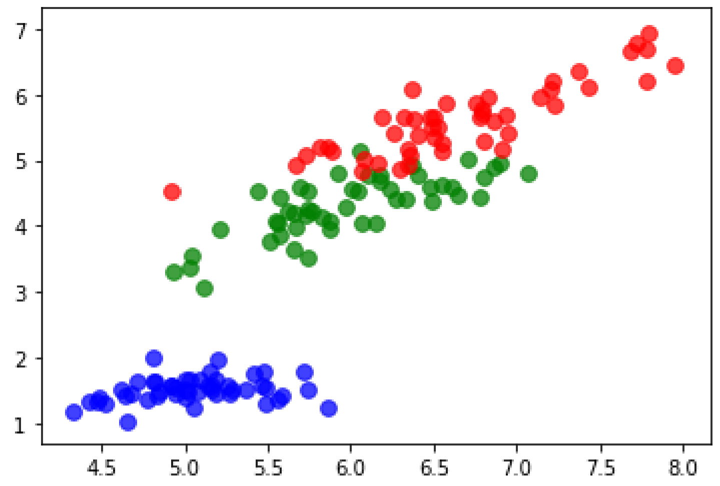


Feature (1,3)

In [16]:
```
colors = ['blue', 'green', 'red']

for i, c in enumerate(np.unique(iris[:, -1])):
    mask = iris[:, -1] == c  # Finding the right points
    plt.scatter(iris[mask, 0], iris[mask, 2], s=60, c=colors[i], alpha=0.75, label='class %d' % i)

plt.show()
```

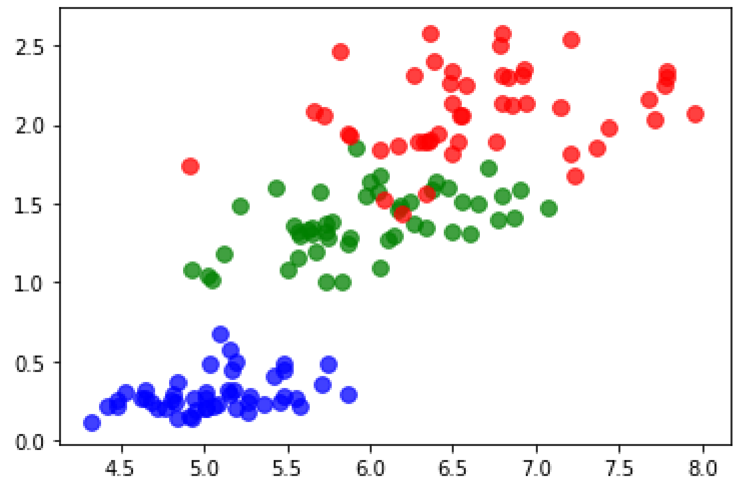


Feature (1,4)

In [17]:
```
colors = ['blue', 'green', 'red']

for i, c in enumerate(np.unique(iris[:, -1])):
    mask = iris[:, -1] == c  # Finding the right points
    plt.scatter(iris[mask, 0], iris[mask, 3], s=60, c=colors[i], alpha=0.75, label='class %d' % i)

plt.show()
```



Problem 2

In [18]:

```
import mltools as ml
import numpy as np
```

In [19]:

```
iris = np.genfromtxt("data/iris.txt",delimiter=None) # load the text file
Y = iris[:,-1]
X = iris[:,0:-1]
```

In [20]:

```
np.random.seed(0) # set the random number seed
```

In [21]:

```
X,Y = ml.shuffleData(X,Y);
```

In [22]:

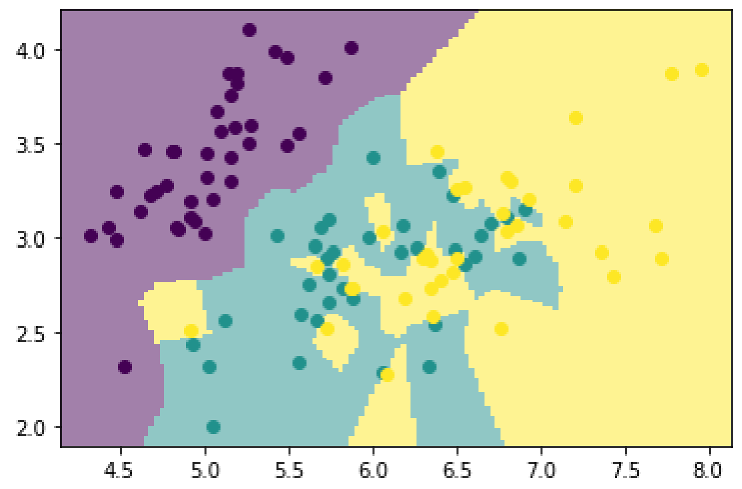
```
Xtr,Xva,Ytr,Yva = ml.splitData(X,Y, 0.75);
```

Problem 2.1

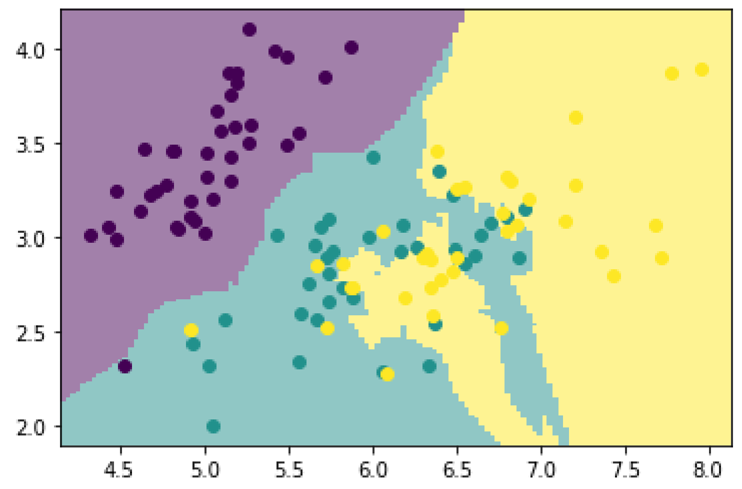
In [23]:

```
import matplotlib.pyplot as plt
%matplotlib inline
K = [1,5,10,50]
for i in K:
    knn = ml.knn.knnClassify()
    print("K = ",i)
    knn.train(Xtr[:, :2], Ytr, K=i)
    ml.plotClassify2D(knn, Xtr[:, :2], Ytr)
    plt.show()
```

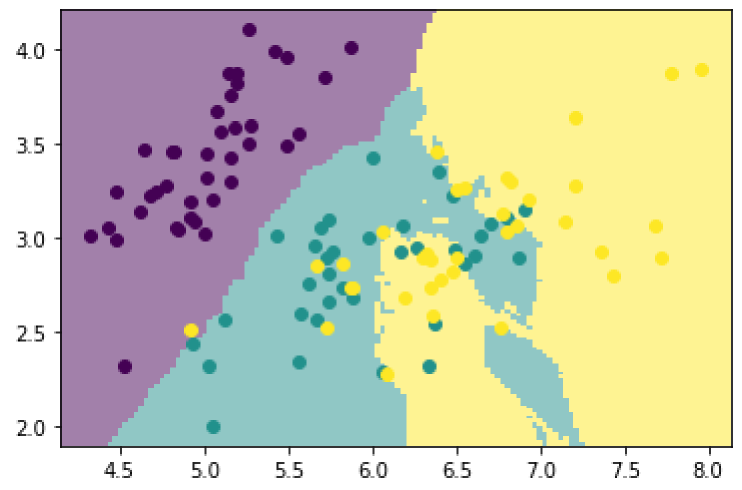
```
K =  1
```



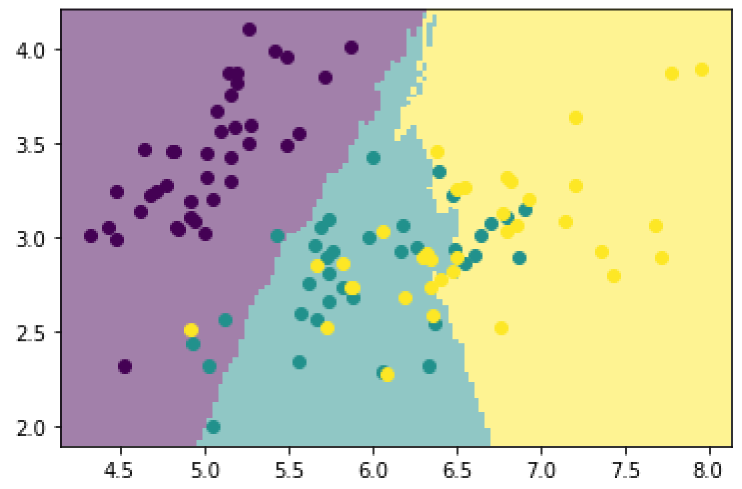
```
K =  5
```



```
K =  10
```



```
K =  50
```

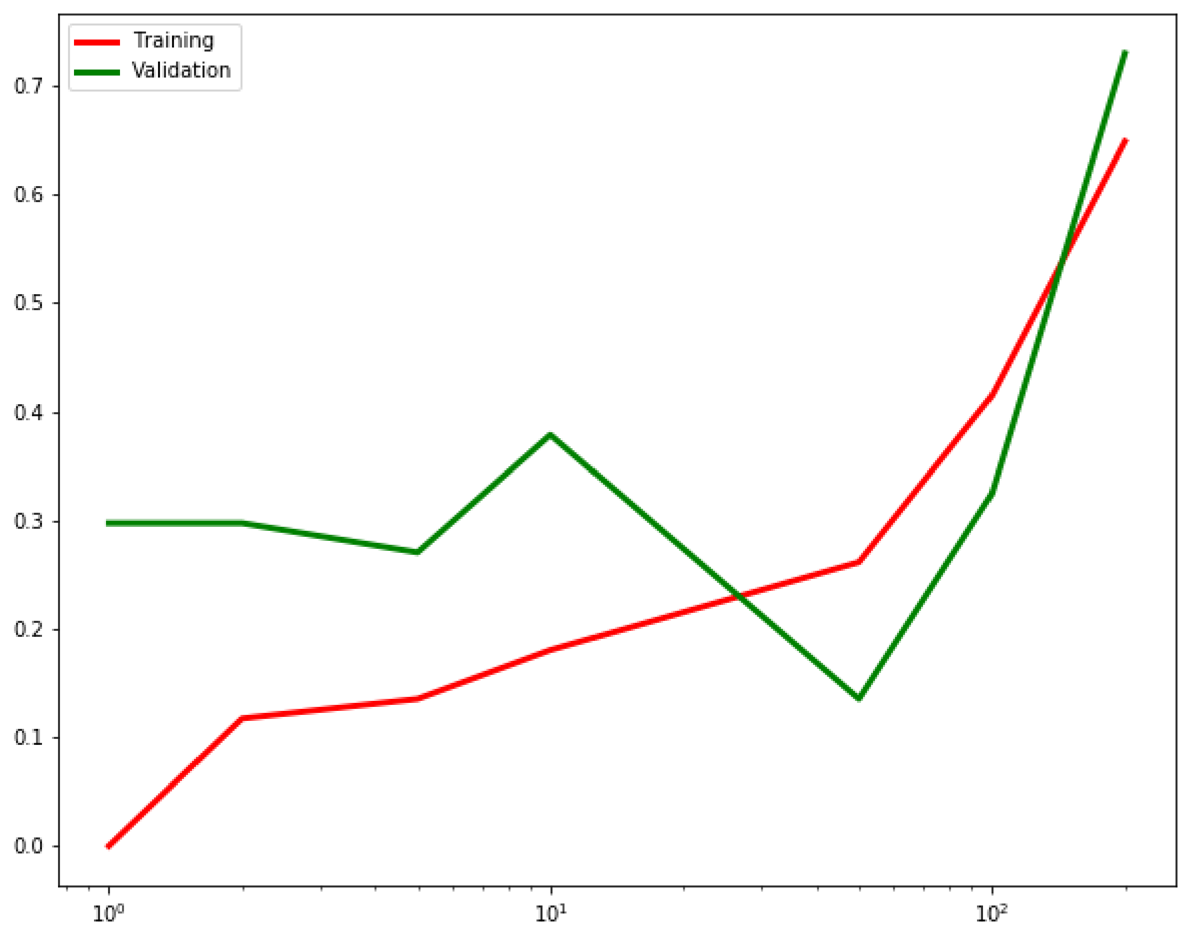


Problem 2.2

In [24]:

```
K=[1,2,5,10,50,100,200];
errTrain = np.zeros((len(K),))
errVa = np.zeros((len(K),))
for i,k in enumerate(K):
    learner = ml.knn.knnClassify( Xtr[:, :2], Ytr, K=k );
    print("K = ",k)
    Yhat = learner.predict(Xtr[:, :2])
    Yvahat = learner.predict(Xva[:, :2])
    errTrain[i] = (np.sum(Yhat != Ytr))/len(Ytr)
    errVa[i] = (np.sum(Yvahat != Yva))/len(Yva)
    print("Training error is", errTrain[i])
    print("Validation error is", errVa[i])
fig, ax = plt.subplots(1, 1, figsize=(10, 8))
ax.semilogx(K, errTrain, 'r-', lw=3, label='Training')
ax.semilogx(K, errVa, 'g-', lw=3, label='Validation')
ax.legend()
plt.show()
```

```
K =  1
Training error is 0.0
Validation error is 0.2972972972972973
K =  2
Training error is 0.11711711711711711
Validation error is 0.2972972972972973
K =  5
Training error is 0.13513513513513514
Validation error is 0.2702702702702703
K =  10
Training error is 0.18018018018018017
Validation error is 0.3783783783783784
K =  50
Training error is 0.26126126126126126
Validation error is 0.13513513513513514
K =  100
Training error is 0.4144144144144144
Validation error is 0.32432432432432434
K =  200
Training error is 0.6486486486486487
Validation error is 0.7297297297297297
```



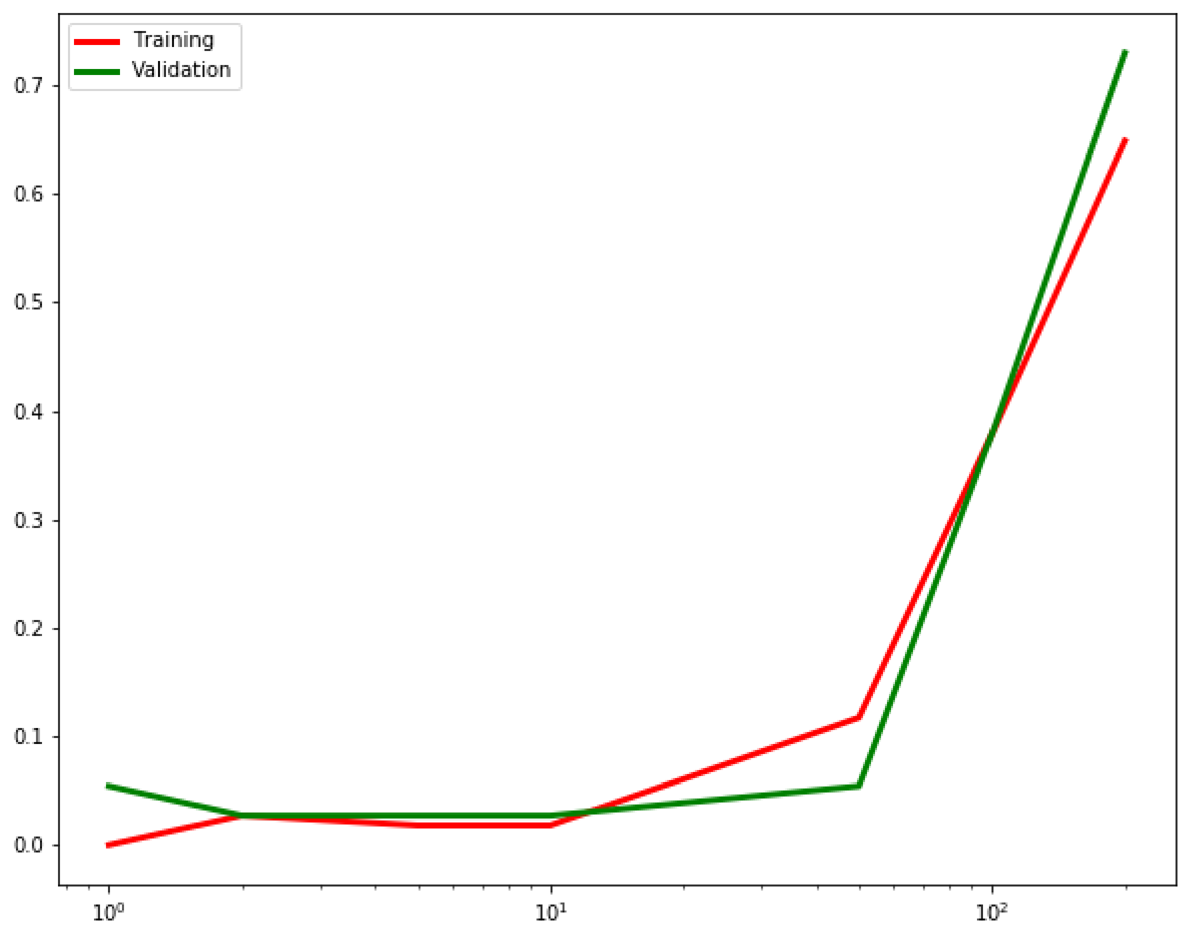
In my opinion, I recommend K = 50

Problem 2.3

In [25]:

```
K=[1,2,5,10,50,100,200];
errTrain = np.zeros((len(K),))
errVa = np.zeros((len(K),))
for i,k in enumerate(K):
    learner = ml.knn.knnClassify( Xtr, Ytr, K=k );
    print("K = ",k)
    Yhat = learner.predict(Xtr)
    Yvahat = learner.predict(Xva)
    errTrain[i] = (np.sum(Yhat != Ytr))/len(Ytr)
    errVa[i] = (np.sum(Yvahat != Yva))/len(Yva)
    print("Training error is", errTrain[i])
    print("Validation error is", errVa[i])
fig, ax = plt.subplots(1, 1, figsize=(10, 8))
ax.semilogx(K, errTrain, 'r-', lw=3, label='Training')
ax.semilogx(K, errVa, 'g-', lw=3, label='Validation')
ax.legend()
plt.show()
```

```
K =  1
Training error is 0.0
Validation error is 0.05405405405405406
K =  2
Training error is 0.02702702702702703
Validation error is 0.02702702702702703
K =  5
Training error is 0.018018018018018018
Validation error is 0.02702702702702703
K =  10
Training error is 0.018018018018018018
Validation error is 0.02702702702702703
K =  50
Training error is 0.11711711711711711
Validation error is 0.05405405405405406
K =  100
Training error is 0.3783783783783784
Validation error is 0.3783783783783784
K =  200
Training error is 0.6486486486486487
Validation error is 0.7297297297297297
```



These two plots are very different. Now the recommendation of K is 5 or 10.

Problem 3

3.1: $P(y=1) = \frac{4}{10} = \frac{2}{5}$

$P(y=-1) = \frac{6}{10} = \frac{3}{5}$

$P(x_1=0|y=1) = \frac{1}{4}$ $\quad P(x_1=1|y=1) = \frac{3}{4}$

$P(x_1=0|y=-1) = \frac{1}{2}$ $\quad P(x_1=1|y=-1) = \frac{1}{2}$

$P(x_2=0|y=1) = 1$ $\quad P(x_2=1|y=1) = 0$

$P(x_2=0|y=-1) = \frac{1}{6}$ $\quad P(x_2=1|y=-1) = \frac{5}{6}$

$P(x_3=0|y=1) = \frac{1}{4}$ $\quad P(x_3=1|y=1) = \frac{3}{4}$

$P(x_3=0|y=-1) = \frac{1}{3}$ $\quad P(x_3=1|y=-1) = \frac{2}{3}$

$P(x_4=0|y=1) = \frac{1}{2}$ $\quad P(x_4=1|y=1) = \frac{1}{2}$

$P(x_4=0|y=-1) = \frac{1}{6}$ $\quad P(x_4=1|y=-1) = \frac{5}{6}$

$P(x_5=0|y=1) = \frac{3}{4}$ $\quad P(x_5=1|y=1) = \frac{1}{4}$

$P(x_5=0|y=-1) = \frac{2}{3}$ $\quad P(x_5=1|y=-1) = \frac{1}{3}$

3.2 The predicted class for $x=(0,0,0,0,0)$ is $+1$

The predicted class for $x=(1,1,0,1,0)$ is -1

3.3 $$P(Y=+1|X=(0,0,0,0,0)) = \frac{P(X=(0,0,0,0,0)|Y=1)\cdot P(Y=1)}{P(X=(0,0,0,0,0)|Y=1)\cdot P(Y=1) + P(X=(0,0,0,0,0)|Y=-1)\cdot P(Y=-1)}$$

$$= \frac{\frac{1}{4}\cdot 1\cdot \frac{1}{4}\cdot \frac{1}{2}\cdot \frac{3}{4}\cdot \frac{2}{5}}{\frac{1}{4}\cdot 1\cdot \frac{1}{4}\cdot \frac{1}{2}\cdot \frac{3}{4}\cdot \frac{2}{5} + \frac{1}{2}\cdot \frac{1}{6}\cdot \frac{1}{3}\cdot \frac{1}{6}\cdot \frac{2}{3}\cdot \frac{3}{5}}$$

$$= 0.835$$

$P(Y=+1|X=(1,1,0,1,0)) = 0$ because $P(X_2=1|Y=1) = 0$

3.4

Because there are too many possibilities when we use joint probability. If we want to use join probability, we must collect all data which is hardly to do. If we lose one probability, the result will be inaccurate.

3.5

Yes, we need to re-train our model.

The formula will be different.

$P(X = (X_2, X_3, X_4, X_5) \mid Y = 1)$

$$= P(X_2 \mid Y = 1) * P(X_3 \mid Y = 1) * P(X_4 \mid Y = 1) * P(X_5 \mid Y = 1)$$

Problem 4

I have followed the academic honesty guidelines posted on the course website.

Bingchen Lu