# Problem1.1

1. Use X.shape to get the number of features and the number of data points. Report both numbers, mentioning which number is which. (5 points)

In [256]:

```
import numpy as np
import matplotlib.pyplot as plt
nych = np.genfromtxt ("nyc_housing.txt", delimiter = None)
Y = nych[:,-1]
X = nych[:,0:-1]
```

In [257]:

```
X.shape
```

Out[257]:

```
(300, 3)
```

300 = Number of data points (row), 3 = Number of features (col)

# Problem1.2

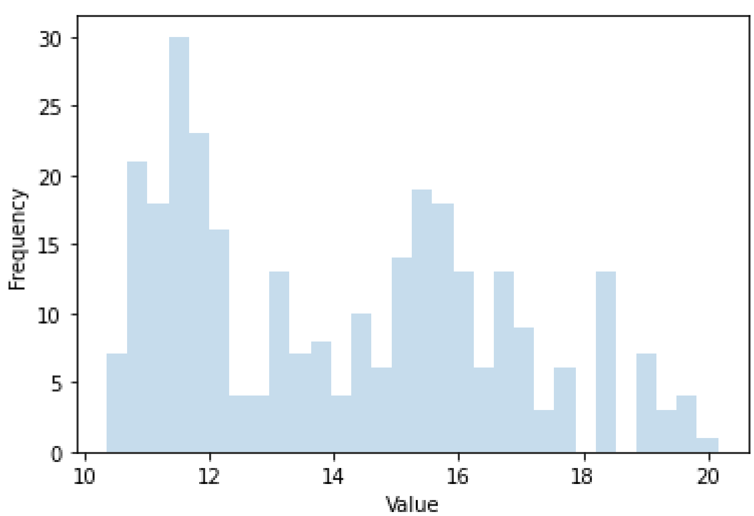
For each feature, plot a histogram ( plt.hist ) of the data values. (5 points)

In [89]:

```
#First histogram
#feature1 = [i[0] for i in X]

feature1 = nych[:,0]
plt.hist(feature1, bins = 30, alpha = 0.25)
plt.xlabel('Value')
plt.ylabel('Frequency')
```

Out[89]:

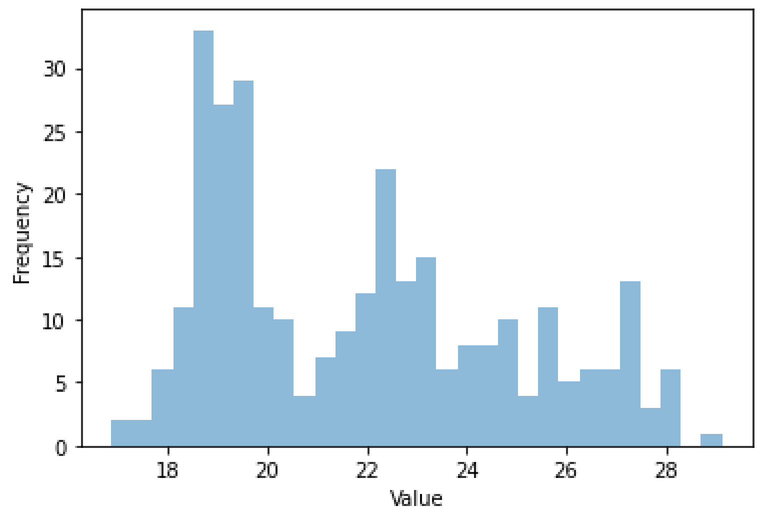
```
Text(0, 0.5, 'Frequency')
```



In [88]:

```
#Second Histogram
feature2 = [i[1] for i in X]
plt.hist(feature2, bins = 30, alpha = 0.5)
plt.xlabel('Value')
plt.ylabel('Frequency')
```

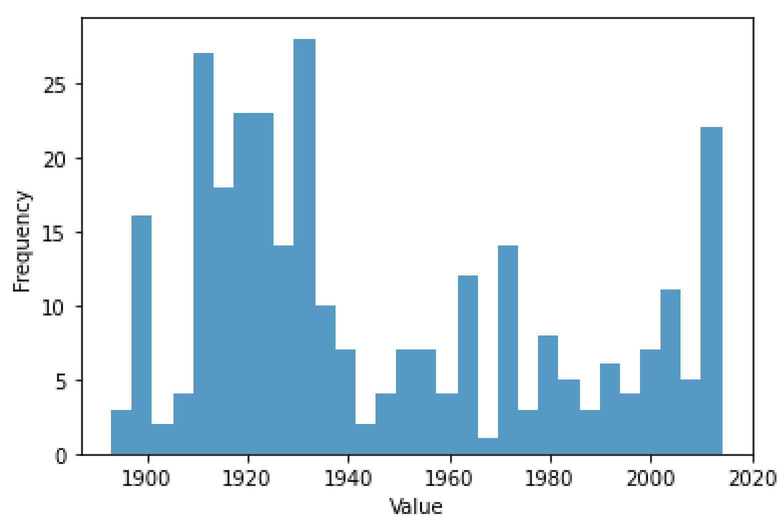
Out[88]: Text(0, 0.5, 'Frequency')




In [90]:

```
#Thrid Histogram
feature3 = [i[2] for i in X]
plt.hist(feature3, bins = 30, alpha = 0.75)
plt.xlabel('Value')
plt.ylabel('Frequency')
```

Out[90]: Text(0, 0.5, 'Frequency')




# Problem1.3

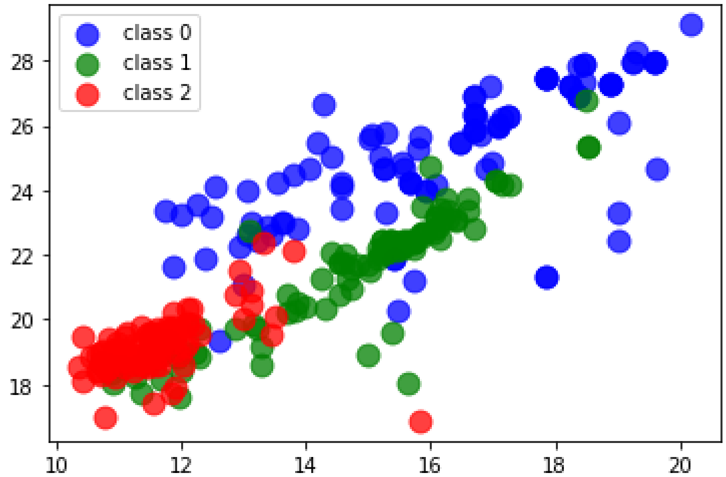
In [263]:
```
a = X[:, 0:1]

print("feature 1 \n mean = " , np.mean(a) , "and standard deivation = "


b = X[:, 1:2]
print("feature 2 \n mean =", np.mean(b), " and standard deivation = ",

c = X[:, 2:3]
print("feature 3 \n mean =", np.mean(c), " and standard deivation =", n
```

```
feature 1
 mean =  14.118392438424483 and standard deivation =  2.569090284260317

feature 2
 mean = 21.907116176170856  and standard deivation =  2.9785784999947165

feature 3
 mean = 1946.3533333333332  and standard deivation = 35.39889577687731
```

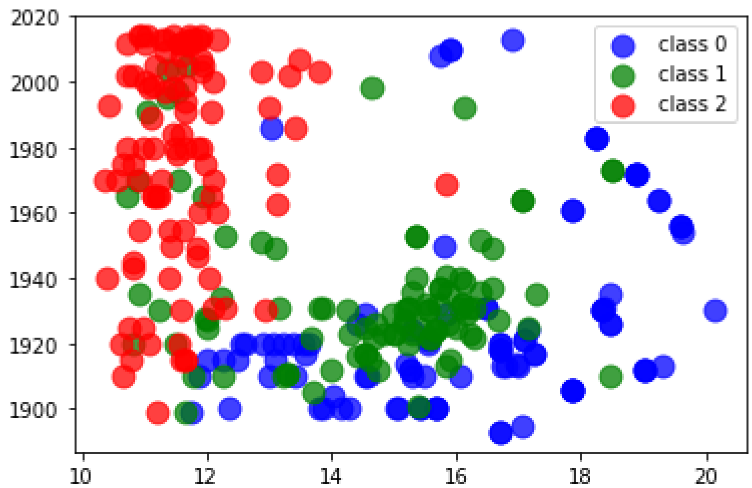
# Problem1.4

For each pair of features (1,2), (1,3), and (2,3), plot a scatterplot (see plt.plot or plt.scatter ) of the feature values, colored according to their target value (class). (For example, plot all data points with y = 0 as blue, y = 1 as green, and y = 2 as red.) (5 points)

In [93]:
```
colors = ['blue', 'green', 'red']

for i, c in enumerate(np.unique(nych[:, -1])):
    mask = np.where(nych[:, -1] == c)[0]    # Finding the right points
    plt.scatter(nych[mask, 0], nych[mask, 1], s=120, c=colors[i], alpha

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

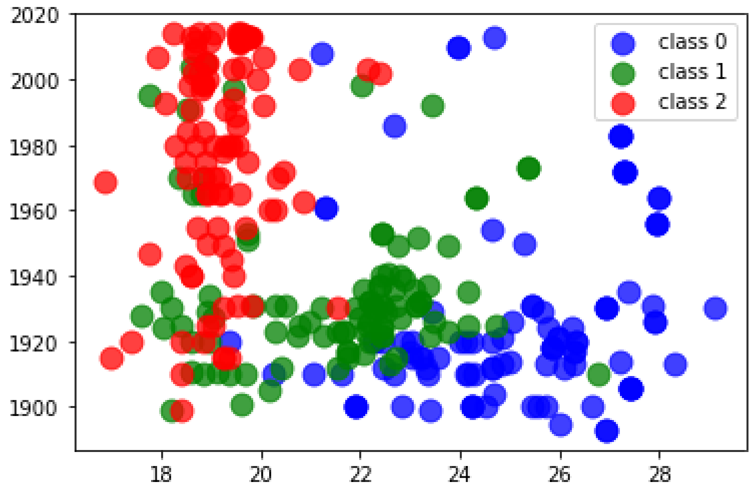



In [94]:
```
colors = ['blue', 'green', 'red']

for i, c in enumerate(np.unique(nych[:, -1])):
    mask = np.where(nych[:, -1] == c)[0]    # Finding the right points
    plt.scatter(nych[mask, 0], nych[mask, 2], s=120, c=colors[i], alpha

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



In [95]:

```python
colors = ['blue', 'green', 'red']

for i, c in enumerate(np.unique(nych[:, -1])):
    mask = np.where(nych[:, -1] == c)[0]    # Finding the right points
    plt.scatter(nych[mask, 1], nych[mask, 2], s=120, c=colors[i], alpha

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



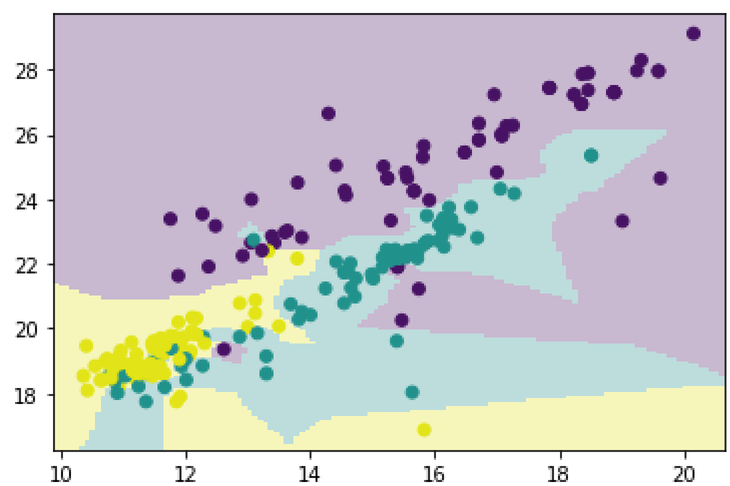

# Problem 2: k-nearest-neighbor predictions (25 points)

# Problem 2.1

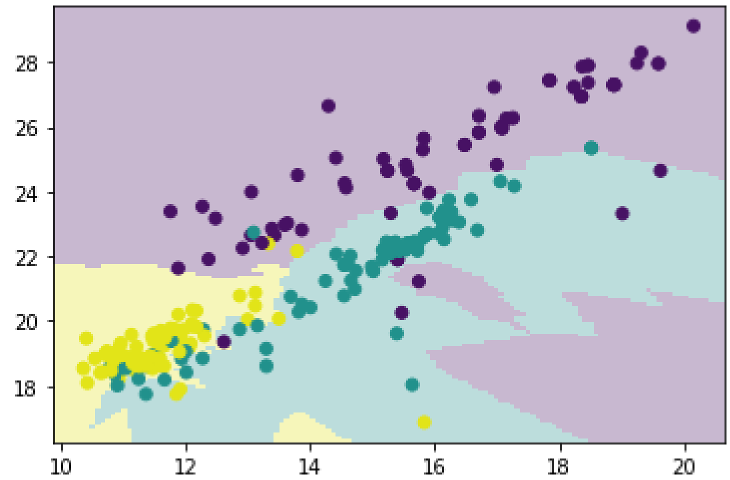
In [155]:

```
nych = np.genfromtxt("nyc_housing.txt",delimiter=None) # load the data
Y = nych[:,-1]
X = nych[:,0:-1]
# Note: indexing with ":" indicates all values (in this case, all rows)
 # indexing with a value ("0", "1", "-1", etc.) extracts only that valu
 # indexing rows/columns with a range ("1:-1") extracts any row/column

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 p
 # 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,Y = ml.shuffleData(X,Y); # shuffle data randomly
 # (This is a good idea in case your data are ordered in some systemati

Xtr,Xva,Ytr,Yva = ml.splitData(X,Y, 0.75); # split data into 75/25 trai

K=[1,5,10,50]
knn = ml.knn.knnClassify()
for i,k in enumerate(K):
    print("K = "+str(k) )

    knn.train(Xtr, Ytr, k)

    Yva = knn.predict ( Xva)
    knn = ml.knn.knnClassify()
    knn.train(Xtr[:, :2], Ytr, k)

    ml.plotClassify2D(knn, Xtr[:, :2], Ytr)
    plt.show()
```

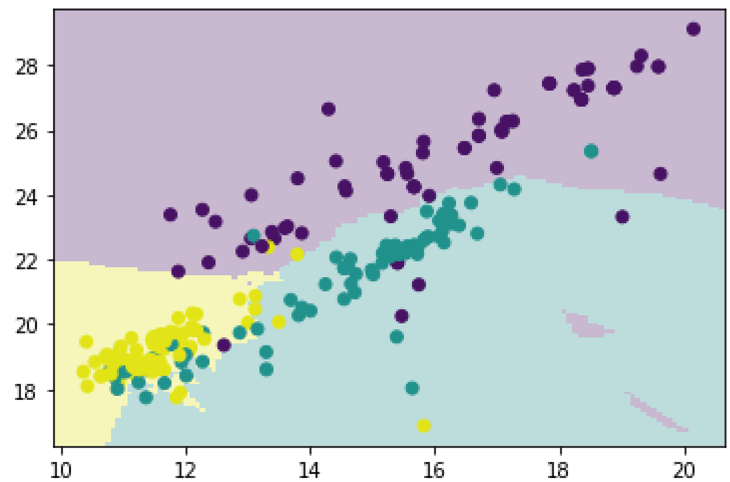
K = 1



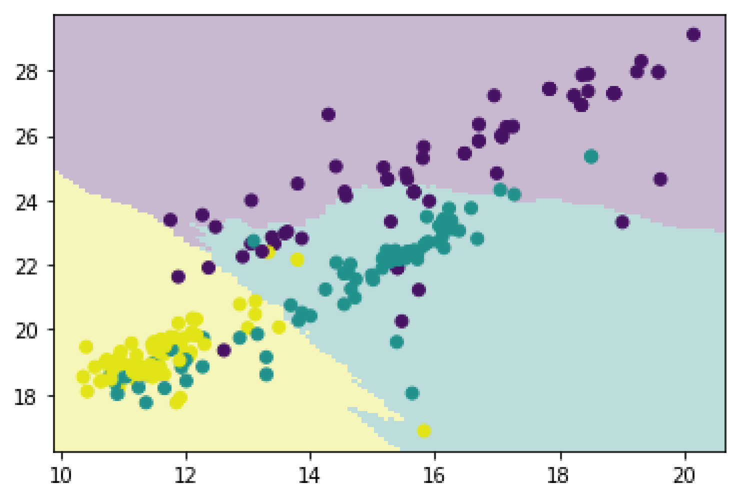

K = 5




K = 10



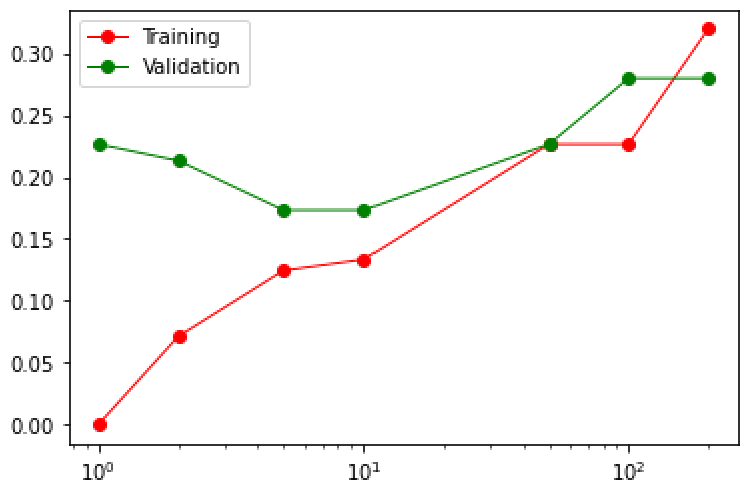

K = 50




# Problem 2.2

Again using only the first two features, compute the error rate (number of misclassifications) on both the training and validation data as a function of K = [1, 2, 5, 10, 50, 100, 200]. You can do this most easily with a for-loop:

In [217]:
```
#with training error in red and validation error in green.
Y = nych [:, -1 ]        #only the first two features
X = nych [:, 0:2 ]       #only the first two features
np.random.seed ( 0 )
X,Y = ml.shuffleData ( X , Y );   # shuffle data randomly
Xtr,Xva,Ytr,Yva = ml.splitData ( X , Y , 0.75 )   # split data into 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 ( ) # create the object and train it
    learner.train( Xtr , Ytr , k)    # where K is an integer, e.g. 1 fc
    errTrain[i] = learner.err(Xtr, Ytr)
    errVal [i]  = learner.err(Xva, Yva)

plt.semilogx(K, errTrain, color='red', lw = 1, marker = 'o', label='Tra
plt.semilogx(K, errVal, color='green', lw = 1, marker = 'o', label='Val
# Adding a legend to the plot that will use the labels from the 'label'
plt.legend()

# Controlling the axis.
ax.semilogx(.8, 200)
ax.semilogx(0, 1)
plt.show ()
# pratcie one
```



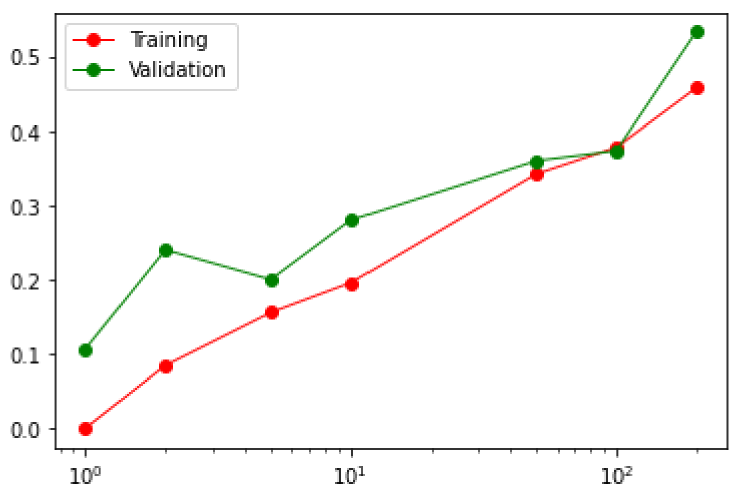

Based on these plots, I would like to recommend value of 10(between 10^(0.5) and 10^(1)). Becasue this value is the lowest

# Problem2.3

Again using only the first two features, compute the error rate (number of misclassifications) on both the training and validation data as a function of K = [1, 2, 5, 10, 50, 100, 200]. You can do this most easily with a for-loop:

In [215]:

```
Y = nych [:, - 1 ]
X = nych [:, 0 : -1 ]
np.random.seed ( 0 )
X,Y = ml.shuffleData ( X , Y );          # shuffle data randomly
Xtr,Xva,Ytr,Yva = ml.splitData ( X , Y , 0.75 )  # split data into 75/2

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 (   )       # create the object and tra
    learner.train( Xtr , Ytr , k)            # where K is an integer, e.
    errTrain[i] = learner.err(Xtr, Ytr)      # to calculate the training
    errVal [i]  = learner.err(Xva, Yva)      # to calculate the training

plt.semilogx(K, errTrain, color='red',   lw = 1, marker = 'o', label='T
plt.semilogx(K, errVal,   color='green', lw = 1, marker = 'o', label='V


plt.legend()   # Adding a legend to the plot that will use the labels f


ax.semilogx(.8, 200) # Controlling the axis.
ax.semilogx(0, 1)
plt.show ()

# pratcie one
```




Based on these plots, I would like to recommend value of 5. Becasue this value is the lowest

# Problem3 Naïve Bayes Classifiers (35 points)

# Problem3.1

Compute all the probabilities necessary for a naïve Bayes classifier, i.e., the class probability p( y) and all the individual feature probabilities p(xi j y), for each class y and feature xi . (7 points)

In [137]:
```
from IPython.display import Image
Image( filename = 'page/178-hw1-9.jpg')
```

Out[137]:

1. Compute all the probabilities necessary for a naïve Bayes classifier, i.e., the class probability $p(y)$ and all the individual feature probabilities $p(x_i|y)$, for each class $y$ and feature $x_i$. *(7 points)*

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

$P(x_1=1 \mid y=1) = \frac{3}{4}$ , $P(x_1=0 \mid y=1) = \frac{1}{4}$

$P(x_2=1 \mid y=1) = \frac{0}{4} = 0$ , $P(x_2=0 \mid y=1) = \frac{4}{4} = 1$

$P(x_3=1 \mid y=1) = \frac{3}{4}$ , $P(x_3=0 \mid y=1) = \frac{1}{4}$

$P(x_4=1 \mid y=1) = \frac{2}{4} = \frac{1}{2}$ , $P(x_4=0 \mid y=1) = \frac{2}{4} = \frac{1}{2}$

$P(x_5=1 \mid y=1) = \frac{1}{4}$ , $P(x_5=0 \mid y=1) = \frac{3}{4}$

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

$P(x_1=1 \mid y=-1) = \frac{3}{6} = \frac{1}{2}$ , $P(x_1=0 \mid y=-1) = \frac{3}{6} = \frac{1}{2}$

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

$P(x_3=1 \mid y=-1) = \frac{4}{6} = \frac{2}{3}$ , $P(x_3=0 \mid y=-1) = \frac{2}{6} = \frac{1}{3}$

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

$P(x_5=1 \mid y=-1) = \frac{2}{6} = \frac{1}{3}$ , $P(x_5=0 \mid y=-1) = \frac{4}{6} = \frac{2}{3}$

# Problem3.2

In [139]:
```python
from IPython.display import Image
Image( filename = 'page/178-hw1-10.jpg')
```

Out[139]:

2. Which class would be predicted for $\underline{x} = (0\ 0\ 0\ 0\ 0)$? What about for $\underline{x} = (1\ 1\ 0\ 1\ 0)$? *(7 points)*

$x = (0\,0\,0\,0\,0)$

$P(x|y=1),\ \frac{1}{4} \times 1 \times \frac{1}{4} \times \frac{2}{4} \times \frac{3}{4} = \frac{6}{4^4} = \frac{6}{256} = \frac{3}{128}$

$P(x|y=-1),\ \frac{3}{6} \times \frac{1}{6} \times \frac{2}{6} \times \frac{1}{6} \times \frac{4}{6} = \frac{24}{6^5} = \frac{24}{7776} = \frac{12}{3888} = \frac{6}{1944} = \frac{3}{972} = \frac{1}{324}$

$P(y=1)\,P(x|y=1) = \frac{4}{10} \times \frac{3}{128} = 0.009375$

$P(y=-1)\,P(x|y=-1) = \frac{6}{10} \times \frac{1}{324} = 0.0018518518\overline{5} = 0.001852$

$P(y=1)\,p(x|y=1) > P(y=-1)\,P(y=-1)\,P(x|y=-1)$

$\therefore$ Predicted as "Read".

$x_1 = 1\ \&\ y=1$ $\qquad x_2 = 0,\ y=1$

$x = (1\,1\,0\,1\,0)$

$P(x|y=1) = \frac{3}{4} \times \frac{0}{4} \times \frac{1}{4} \times \frac{2}{4} \times \frac{3}{4} = \frac{0}{1024}$

$P(x|y=-1) = \frac{3}{6} \times \frac{5}{6} \times \frac{2}{6} \times \frac{5}{6} \times \frac{4}{6} = \frac{600}{7776}$

$P(y=1)\,P(x|y=1) = \frac{4}{10} \times \frac{0}{1024}$

$P(y=-1)\,P(x|y=-1) = \frac{6}{10} \times \frac{600}{7776} = 0.0462962962\overline{9}$

$\approx 0.04630$

$P(x|y=1) < P(x|y=-1)$

$\therefore$ Predict "Discard"

# Problem3.3

In [146]:
```
from IPython.display import Image
Image( filename = 'page/178-hw1-11.jpg')
```

Out[146]:

3. Compute the posterior probability that $y = +1$ given the observation $\underline{x} = (0\ 0\ 0\ 0\ 0)$. Also compute the posterior probability that $y = +1$ given the observation $\underline{x} = (1\ 1\ 0\ 1\ 0)$. *(7 points)*

$x = (0\ 0\ 0\ 0\ 0)$

$$P(y=1|x) = \frac{P(y=1)\,P(x|y=1)}{P(y=1)\cdot P(x|y=1) + P(y=-1)\,P(x|y=-1)}$$

$$= \frac{\frac{4}{10} \times \frac{24}{1024}}{\frac{4}{10} \times \frac{24}{1024} + \frac{6}{10} \times \frac{24}{7776}} \approx 0.8351$$

$x = (1\ 1\ 0\ 1\ 0)$

$$P(y=1|x) = \frac{P(y=1)\,P(x|y=1)}{P(y=1)\cdot P(x|y=1) + P(y=-1)\,P(x|y=-1)}$$

$$= \frac{\frac{4}{10} \times 0}{\frac{4}{10} \times 0 + \frac{6}{10} \times \frac{600}{7776}} = 0.00$$

# Problem3.4

In [143]:
```
from IPython.display import Image
Image( filename = 'page/178-hw1-12.jpg')
```

Out[143]:

4. Why should we probably not use a "joint" Bayes classifier (using the joint probability of the features $x$, as opposed to the conditional independencies assumed by naïve Bayes) for these data? *(7 points)*

We can think about two chararitst of "Joint Bayes". One the depedent and antoher is total number.
FIrst, case of "Joint Bayes" need to be dependent. But this problems (email) event is not dependent not independent.
Second, for the "Joint Bayes", we need 32(2^5) parameters. But, in this case, we just have 10 parameters.

# Problem3.5

In [145]:

```
from IPython.display import Image
Image( filename = 'page/178-hw1-13.jpg')
```

Out[145]:

5. Suppose that before we make our predictions, we lose access to my address book, so that we cannot tell whether the email author is known. Do we need to re-train the model to classify based solely on the other four features? If so, how? If not, what changes about how our trained parameters are used? Hint: what parameters do I need for a naïve Bayes model over only features $x_2, \ldots, x_5$? Do I need to re-calculate any new parameter values in our new setting? What, if anything, changes about the parameters or the way they are used? *(7 points)*

No, we don't have to re-train the model. Becasue this is independency event, it will not change the probabilies.

For example, probability of $P(x_1 = 1 \mid y = 1)$ will not be changed even if $P(x_n = 1 \mid y = 1)$ is changed.

# Problem 4: Gaussian Bayes Classifiers (15 points)

Now, using the NYC Housing data, we will explore a classifier based on Bayes rule. Again, we'll use only the first two features of NYC Housing, shuffled and split in to training and validation sets as before.

1. Splitting your training data by class, compute the empirical mean vector and covariance matrix of the data in each class. (You can use mean and cov for this.) (5 points)

In [ ]:

```

```

# Problem 5: Statement of Collaboration (5 points)

- Piazza question @116 "How to use err() "
- Pual Sung : talk about basic principle of "k-nearest-neighbor predictions"
- Paul Sung : discuss about probability for naive bayes