The University of British Columbia

CPSC 340

Machine Learning and Data Mining

Homework #1

Student name:

ALI SIAHKOOHI

Student ID:

620192032

Instructor:

Professor Schmidt

Date Submitted:

September 23rd, 2016

Question #1,

Part 1.1

(1)

The desired statistical features are listed below:

minimum = 0.352000000000000
maximum = 4.862000000000000
mean = 1.324624999999998
median = 1.159000000000000
mode = 0.770000000000000

(2)

10% quantile: 0.501500000000000
25% quantile: 0.717000000000000
50% quantile: 1.159000000000000
75% quantile: 1.813500000000000
90% quantile: 2.317000000000000

(3)

Region 10 hast the highest mean: 1.566961538461540
Region 9 hast the lowest mean: 1.063211538461538

Region 8 hast the highest variance: 0.798801587858220
Region 9 hast the lowest variance: 0.322038875942685

(4)

Regions {2, 3} have the highest correlation: 0.987880705364545
Regions {1, 8} have the highest correlation: 0.709261207177526

Since the data is continuous, mode of data itself is not a good indication of most frequent data. In order to have a more meaningful sense of mode, I would bin the data. In other words, discretizing the range of data in to several bins would get us a better understanding of the mode! For instance, if we choose the number of bins to be 10, and if choose the mid point of each interval for discretization in that interval, we would get:

mode(10 bins) = 0.577500000000000

Part 1.2

(1)

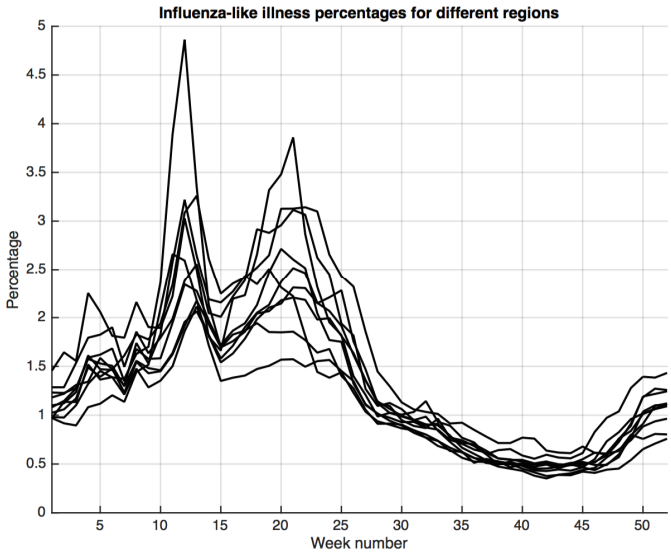

Figure 1 – 1 A plot containing the weeks on the x-axis and the percentages for each region on the y-axis.

(2)

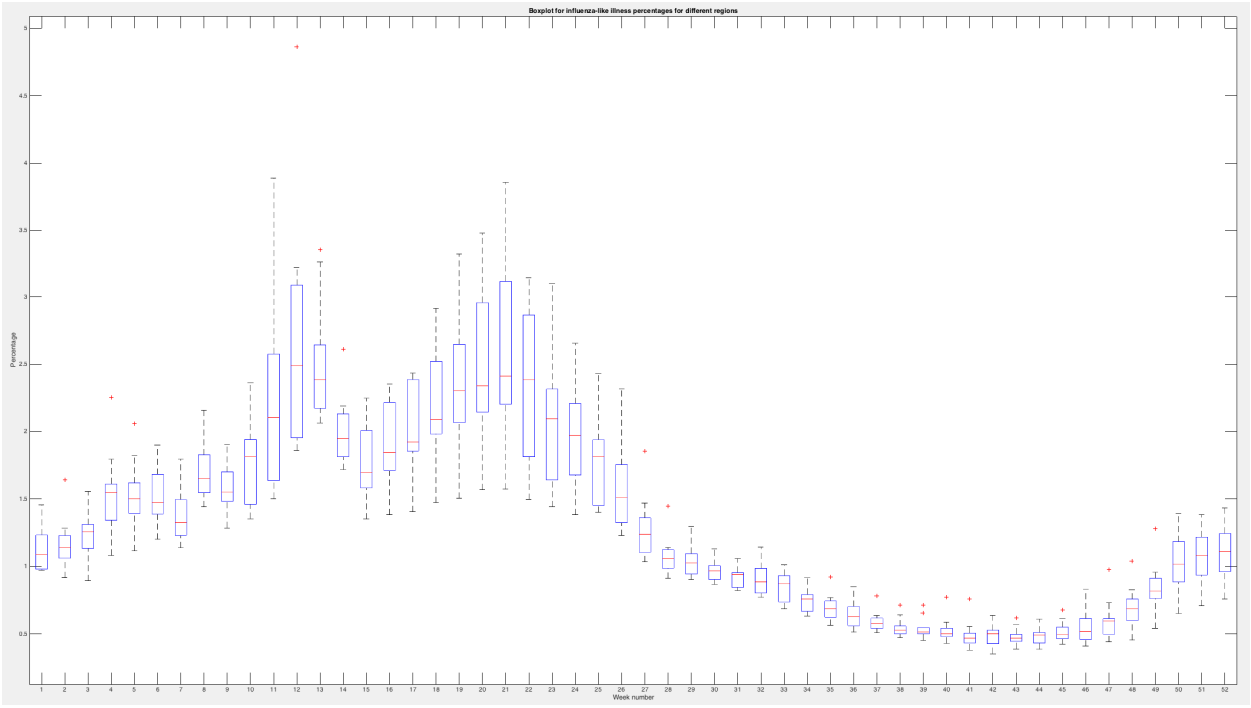

Figure 1 – 2 A boxplot grouping data by weeks, showing the distribution across regions for each week.

(3)

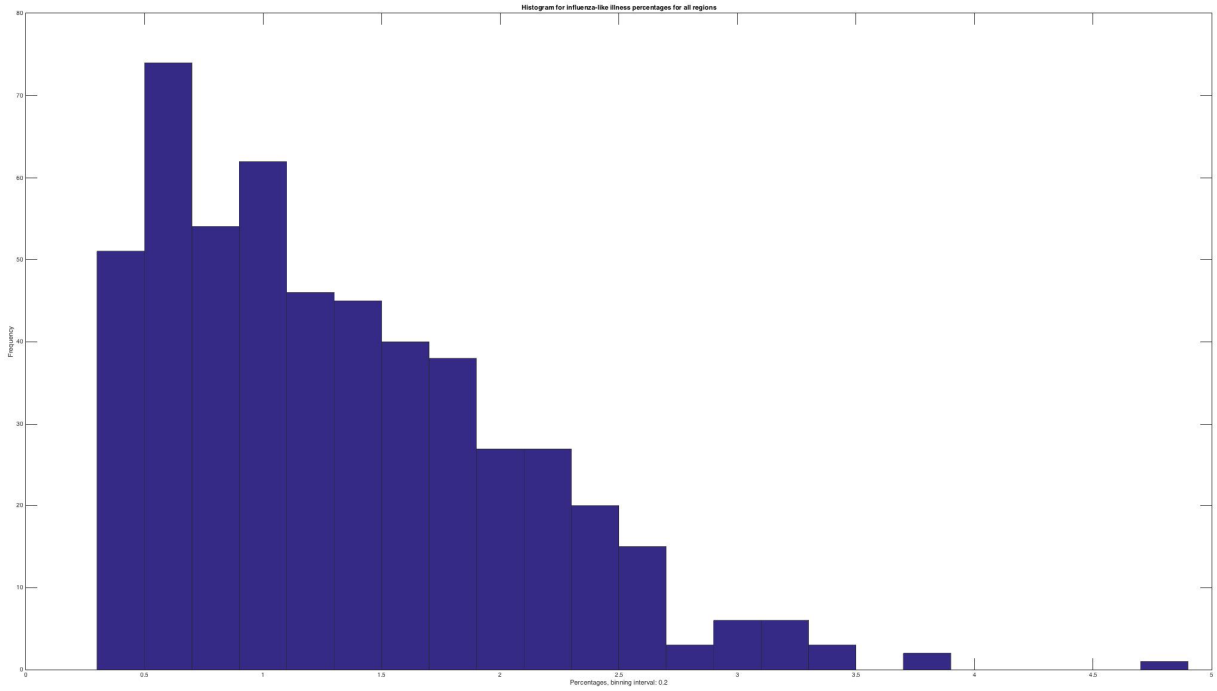

Figure 1 – 3 A histogram showing the distribution of all the values in the matrix X (with step size 0.2).

(4)

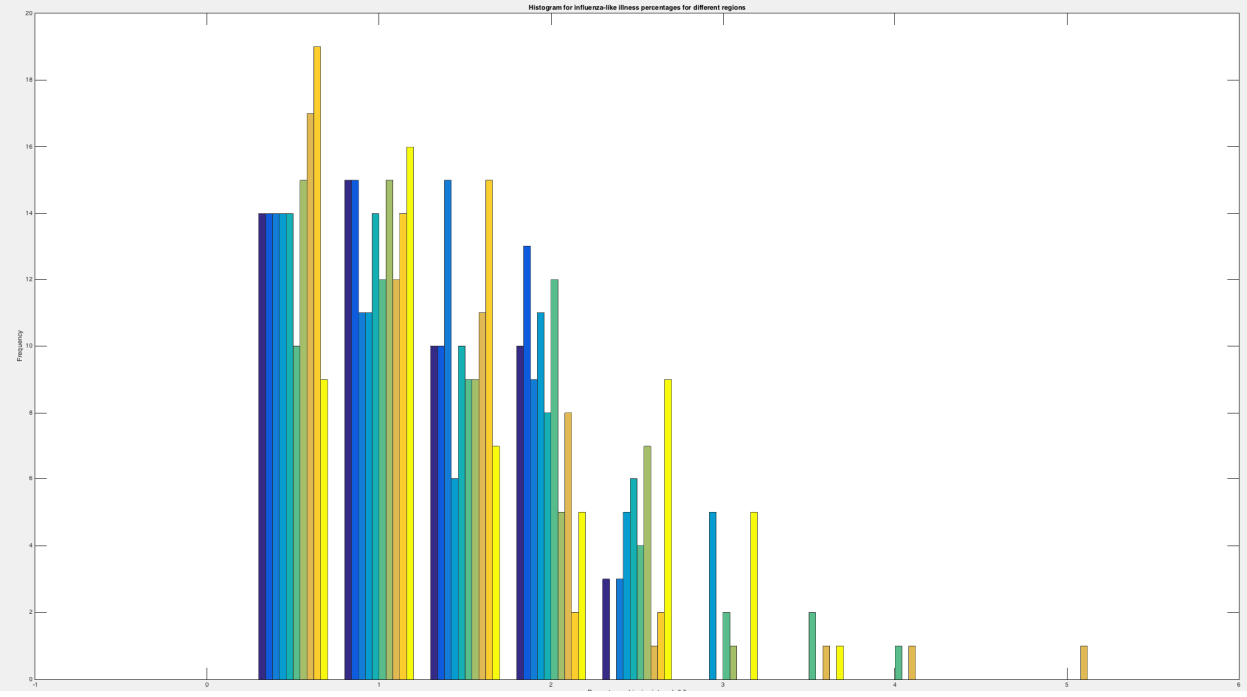

Figure 1 – 4 A single histogram showing the distribution of all the columns in X (with step size 0.5).

(5)

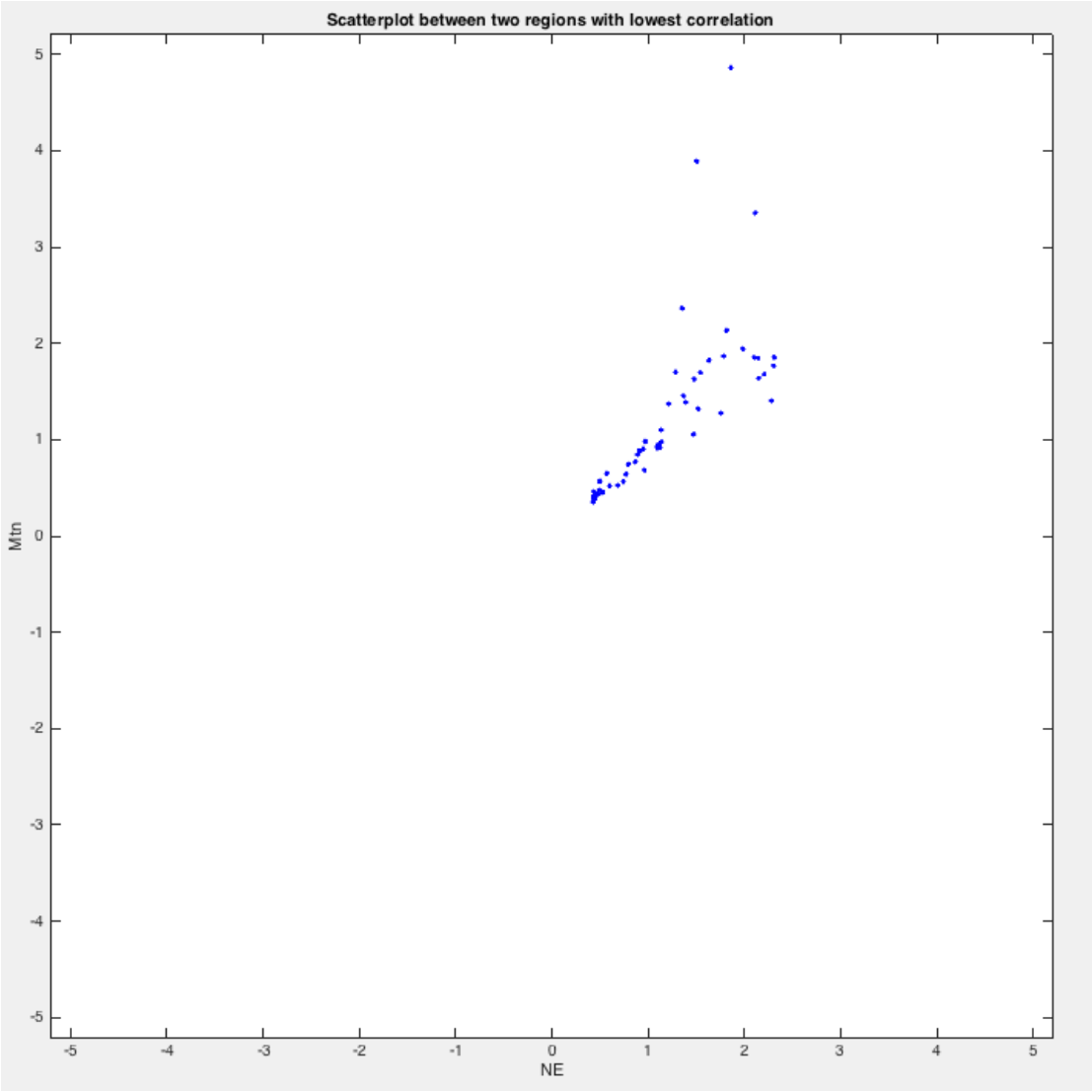

Figure 1 – 5 A scatterplot between the two regions with lowest correlation.

(6)

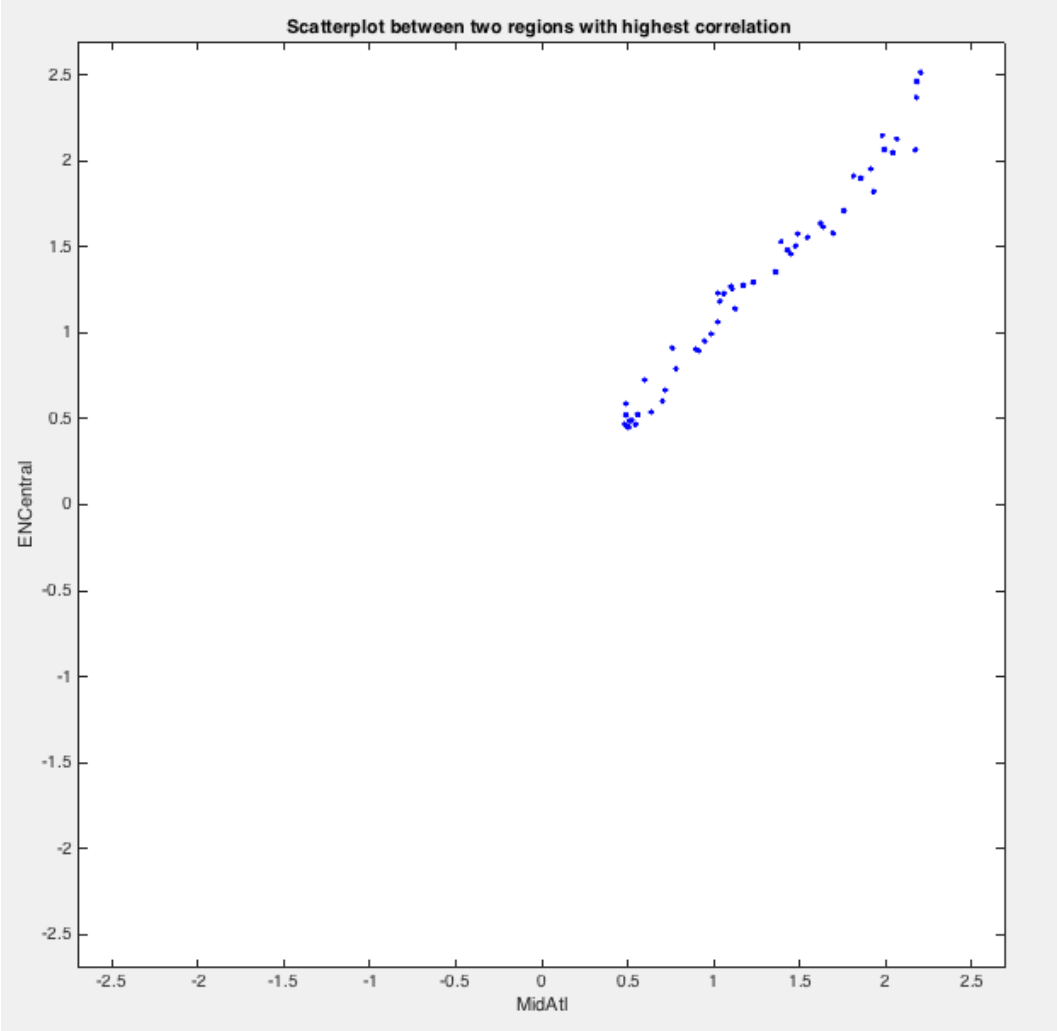

Figure 1 – 6 A scatterplot between the two regions with highest correlation.

 # Question #2

2.1
decisionStump.m is uploaded!
the new error rate is : 0.2525

2.2

predict_ifelse.m is uploaded!

Classification accuracy is not a good score (criteria) for learning. It works when simple rules apply well to the problem. On the other hand, Information gain, which is based on Entropy, can have more complicated rules and better in general for problems which simple rules don't apply.

2.3

$$\text{Cost of fitting} = O(nd\log n) + \sum_{i=1}^{m-1} 2^i O(\frac{n}{2^i} d\log\frac{n}{2^i}) \le O(nd\log n) + (m-1)O(nd\log n) = O(mnd\log n)$$

Question #3

3.1
As expected, the training error gets small and smaller, while the testing error for decision trees with depth 9 or higher remain approximately constant. In that situation, the decision tree has become over trained!

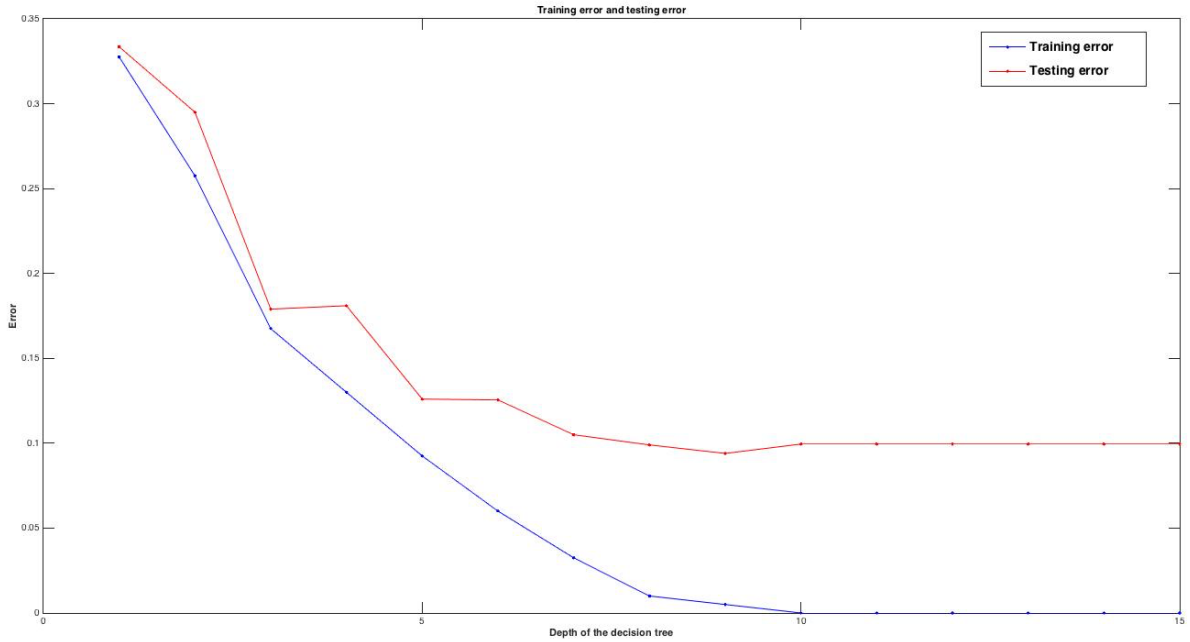

Figure 2 – 1 Training error and testing error versus depth of the decision tree.

3.2

Below are the results of testing different depths for minimizing validation error:

```
Validation error with depth-1 decision tree: 353.000
Validation error with depth-2 decision tree: 250.000
Validation error with depth-3 decision tree: 155.000
Validation error with depth-4 decision tree: 146.000
Validation error with depth-5 decision tree: 121.000
Validation error with depth-6 decision tree: 107.000
Validation error with depth-7 decision tree: 91.000
Validation error with depth-8 decision tree: 57.000
Validation error with depth-9 decision tree: 57.000
Validation error with depth-10 decision tree: 55.000
Validation error with depth-11 decision tree: 57.000
Validation error with depth-12 decision tree: 58.000
Validation error with depth-13 decision tree: 57.000
Validation error with depth-14 decision tree: 57.000
Validation error with depth-15 decision tree: 57.000
```

So a decision tree with depth 10 would have the minimum validation error.

If we switch the training and validation set, the results are different. Maybe the reason is the features are not completely IID. Below are the results if we switch the training and validation set:

```
Validation error with depth-1 decision tree: 287.000
Validation error with depth-2 decision tree: 193.000
Validation error with depth-3 decision tree: 201.000
Validation error with depth-4 decision tree: 149.000
Validation error with depth-5 decision tree: 113.000
Validation error with depth-6 decision tree: 79.000
Validation error with depth-7 decision tree: 59.000
Validation error with depth-8 decision tree: 52.000
Validation error with depth-9 decision tree: 51.000
Validation error with depth-10 decision tree: 53.000
Validation error with depth-11 decision tree: 52.000
Validation error with depth-12 decision tree: 52.000
Validation error with depth-13 decision tree: 51.000
Validation error with depth-14 decision tree: 51.000
Validation error with depth-15 decision tree: 51.000
```

As it mentioned before, in this situation a depth 9 decision tree has the minimum validation error. We can use more data in order to get a lower validation error by dividing the data to more than 2 subsets. It's worth mentioning that dividing the data set to so many subsets could cause increase in optimization bias.

#Question 4

4.1

As the problem states, we have:

$$\begin{cases} P(T=1|D=1)=0.99 \\ P(T=0|D=0)=0.99, \\ P(D=1)=0.001 \end{cases} \tag{4-1}$$

$$\begin{aligned} &P(T=1|D=0)=1-P(T=0|D=0)=1-0.99 \\ &\Rightarrow P(T=1|D=0)=0.01 \\ &P(D=0)=1-P(D=1)=1-0.001 \\ &\Rightarrow P(D=0)=0.999 \\ &P(D=1|T=1)=\frac{P(T=1|D=1)P(D=1)}{P(T=1)}=\frac{P(T=1|D=1)P(D=1)}{P(T=1|D=1)P(D=1)+P(T=1|D=0)P(D=0)} \\ &\Rightarrow P(D=1|T=1)=\frac{0.99\times 0.001}{0.99\times 0.001+0.01\times 0.999}=0.0902 \end{aligned} \tag{4-2}$$

4.2

(a)

$$\begin{aligned} &p(y=1)=\frac{6}{10}, \\ &p(y=0)=\frac{4}{10}. \end{aligned} \tag{4-3}$$

(b)

$$\begin{aligned} &p(x_1=1|y=1)=\frac{3}{6} \\ &p(x_2=1|y=1)=\frac{4}{6} \\ &p(x_1=1|y=0)=\frac{4}{4} \\ &p(x_2=1|y=0)=\frac{1}{6} \end{aligned} \tag{4-4}$$

(c)

$$
\begin{aligned}
&p(y=1 \mid x_1=1, x_2=1) \propto p(x_1=1, x_2=1 \mid y=1)p(y=1) \\
&\approx p(x_1=1 \mid y=1)p(x_2=1 \mid y=1)p(y=1) = \frac{3}{6}\frac{4}{6}\frac{6}{10} = \frac{1}{5} \\
&p(y=0 \mid x_1=1, x_2=1) \propto p(x_1=1, x_2=1 \mid y=0)p(y=0) \\
&\approx p(x_1=1 \mid y=0)p(x_2=1 \mid y=0)p(y=0) = \frac{4}{4}\frac{1}{6}\frac{4}{10} = \frac{1}{15} \\
&\Rightarrow p(y=1 \mid x_1=1, x_2=1) \geq 3p(y=0 \mid x_1=1, x_2=1) \\
&\Rightarrow x_1=1, x_2=1 \rightarrow y=1.
\end{aligned}
\tag{4 – 5}
$$

From the calculations above, we can see that y=1 is the most likely label.

4.3

The obtained errors are as below:

```
Validation error with decision tree: 0.36
Validation error with naive Bayes: 0.19
```

In addition to the naiveBayes.m function that I have handed in, the following is the code used to compute p_xy separately.

```
for c = 1:k
    ind = find(c==y);
    for j = 1:d
        tmp = sum(1==X(ind, j));
        a = zeros(1);a(1) = tmp(1);
        p_xy(j, 1, c) = a/length(ind);
    end
end
p_xy(:, 2, :) = 1 - p_xy(:, 1, :);
```

4.4

It is obvious from the predict function in naiveBayes.m code that that if we don't consider the features binary, and consider they can have c discrete values, the computational complexity of ther predict function is:

O(tdck).