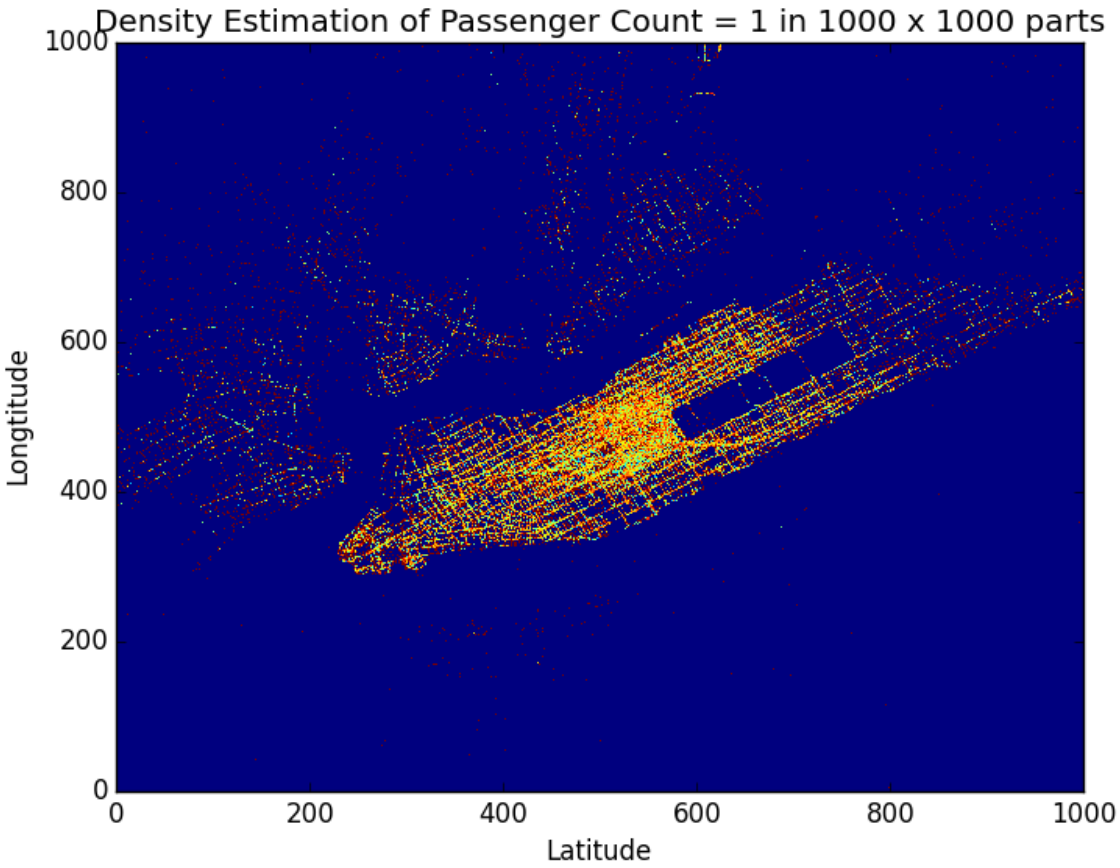
CS5785 Homework2

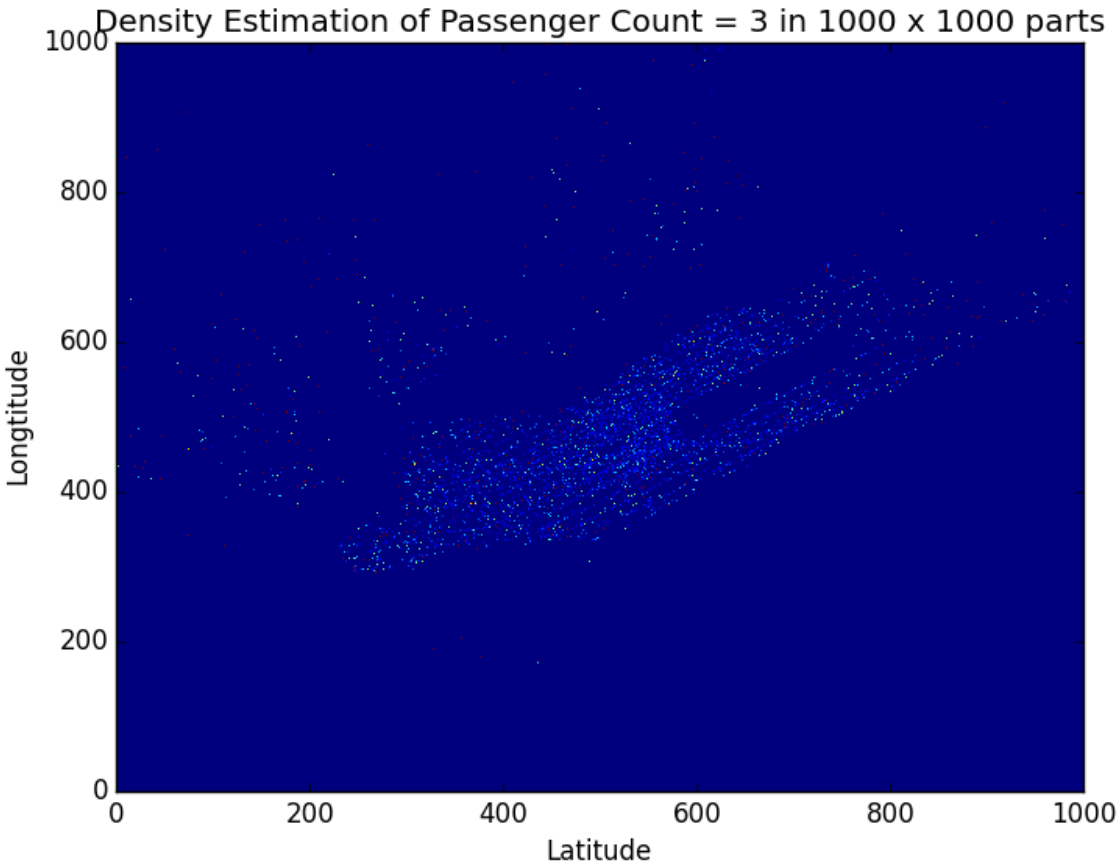
Author: Siyi Fan, Wen Li

1 Programming Exercises

1. K-nearest-neighbour

(a) Discretization is used in this problem to do the density estimation. All the longitude and latitude is divided into 1000 parts from the minimum to the maximum. And then, within each part, the total numbers of passengers with both 1 and 3 are counted, after which the total numbers are divided by the number of elements in the part. It gives the probability of the passengers conditioned on the latitude and longitude. Since there are many extreme large outliers in the dataset and in order to show the graph more clear, a outlier filter is given by limit the latitude from 40 to 45 and the longitude from -75 to -70. And we can see the beautiful map of New York City. It is obvious that 1 passengers has much more probability than 3 passenger.





(b) The k nearest neighbor is implemented using sklearn library. And below is the result:
Training Data: Train_Data.csv Test Data: Top 10000 lines of Trip_Data_1.csv
Root Mean Square Error: 448.23450213
Mean Absolute Error: 279.0433
Correlation Coefficient: [[1 0.68720671], [0.68720671 1]]

(c) Below is the result:
Training Data: Train_Data.csv Test Data: Top 10000 lines of Trip_Data_1.csv
Root Mean Square Error: 442.744574219
Mean Absolute Error: 301.378
Correlation Coefficient: [[1 0.61642068] [0.61642068 1]]

(d) The way to do the scale is using the normalization method, which is removing the mean from the data and then being divided by the standard deviation. The result improves and below is the result:
Training Data: Train_Data.csv Test Data: Top 10000 lines of Trip_Data_1.csv
Root Mean Square Error: 409.410793458
Mean Absolute Error: 261.1406
Correlation Coefficient: [[1 0.69130892][0.69130892 1]]

(e) We run the model with k from 5 to 20 and for each group of neighbours, we find the median of trip time as the prediction. By comparing the mean absolute error, we find that with k = 20, we got the optimal model. Below is the evaluation:
k = [5, 20]
*** **********
5 208.1771
6 206.224
7 204.9629
8 203.81435
9 202.2108
10 199.6945
11 197.4752
12 196.61865
13 195.7583
14 195.7804
15 195.4448
16 195.76
17 196.7749
18 196.95635
19 196.6028
20 195.2892

Optimal K: 20

(f) We can split both the latitude and longitude into N parts, and in each parts, we can use the highest probability to represents the approximation value of that part. Now, we divide the data into NxN parts. The approximation value Y = max(P(X)) where X is the test data, P(X) is the set of probability of the all the labels in the same part of X. It can ease the calculation of training the model.

2. Decision Tree

(a) ['good quality dog food bought several vitality canned dog food products found good quality product looks like stew processed meat smells better labrador finicky appreciates product better', 1],
['advertised product arrived labeled jumbo salted peanuts peanuts actually small sized unsalted sure error vendor intended represent product jumbo', 0]
['delight says confection around centuries light pillowy citrus gelatin nuts case filberts cut tiny squares liberally coated powdered sugar tiny mouthful heaven chewy flavorful highly

recommend yummy treat familiar story c lewis lion witch wardrobe treat seduces edmund selling brother sisters witch', 1]
['cough medicine looking secret ingredient robitussin believe found got addition root beer extract ordered good made cherry soda flavor medicinal', 0]
['great taffy great taffy great price wide assortment yummy taffy delivery quick taffy lover deal', 1]

*From the previews pair of review label, we notice that the the review with the label 0 is indeed negative review. However, the negative review tends to go into detail and tries to write the review in a really sarcastic way, hence the words in negative review are pretty scattered; And the reviews with label 1 look like mediocre or somehow positive. And for the positive reviews, the word 'great' appears many times.

(b) top 30 of all (in 40000 reviews):

```
[('the', 134788), ('i', 118305), ('and', 92210), ('a', 87870), ('it', 75521), ('to', 72204), ('of', 56730), ('is',
52321), ('this', 48064), ('br', 44188), ('for', 41243), ('in', 38325), ('my', 33442), ('that', 31413), ('but', 27984),
('you', 26257), ('not', 25910), ('with', 25561), ('have', 24611), ('are', 23362), ('they', 22931), ('s', 22316), ('was',
21950), ('t', 21581), ('as', 19787), ('on', 19315), ('like', 18788), ('so', 18302), ('these', 18101)]
```

top 30 of all positive (in 40000 reviews):

```
[('the', 95464), ('i', 85094), ('and', 71133), ('a', 66782), ('it', 55095), ('to', 52976), ('of', 40672), ('is', 39724),
('this', 35240), ('br', 31608), ('for', 31597), ('in', 28253), ('my', 25996), ('that', 22030), ('you', 20328), ('with',
19277), ('have', 18785), ('but', 18724), ('are', 18398), ('they', 17104), ('s', 16709), ('great', 15404), ('not',
15276), ('as', 14535), ('on', 14429), ('t', 14273), ('these', 14171), ('good', 14125), ('was', 13977)]
```

top 30 of all negative (in 40000 reviews):

```
[('the', 39324), ('i', 33211), ('a', 21088), ('and', 21077), ('it', 20426), ('to', 19228), ('of', 16058), ('this',
12824), ('is', 12597), ('br', 12580), ('not', 10634), ('in', 10072), ('for', 9646), ('that', 9383), ('but', 9260),
('was', 7973), ('my', 7446), ('t', 7308), ('with', 6284), ('you', 5929), ('they', 5827), ('have', 5826), ('s', 5607),
('like', 5570), ('as', 5252), ('are', 4964), ('on', 4886), ('so', 4614), ('be', 4519)]
```

* Conclusion is hard to be drawn in these three sets. Because there are so many stop words and those words weigh out other useful words. But we can still see that for negative reviews there are a lot of 'but's

(c) top 30 of all (in 40000 reviews):

```
[('br', 44188), ('like', 18788), ('good', 17797), ('great', 17016), ('coffee', 13238), ('taste', 13127), ('product',
12895), ('one', 12476), ('flavor', 11209), ('love', 10702), ('tea', 10264), ('would', 9003), ('food', 8654), ('get',
7840), ('best', 7769), ('amazon', 7500), ('really', 7451), ('much', 6742), ('dog', 6668), ('time', 6150), ('use', 6096),
('price', 6086), ('ve', 6062), ('little', 6038), ('also', 5932), ('buy', 5766), ('tried', 5541), ('better', 5484),
('make', 5380)]
```

top 30 of all positive (in 40000 reviews):

```
[('br', 31608), ('great', 15404), ('good', 14125), ('like', 13218), ('coffee', 9792), ('love', 9509), ('one', 9254),
('product', 8981), ('taste', 8863), ('tea', 8422), ('flavor', 8143), ('best', 7093), ('food', 6568), ('would', 5824),
('get', 5770), ('amazon', 5756), ('really', 5600), ('dog', 5007), ('use', 4914), ('price', 4875), ('ve', 4848), ('much',
4781), ('little', 4727), ('time', 4686), ('also', 4646), ('make', 4242), ('find', 4213), ('tried', 4187), ('well',
4084)]
```

top 30 of all negative (in 40000 reviews):

```
[('br', 12580), ('like', 5570), ('taste', 4264), ('product', 3914), ('good', 3672), ('coffee', 3446), ('one', 3222),
('would', 3179), ('flavor', 3066), ('food', 2086), ('get', 2070), ('much', 1961), ('really', 1851), ('tea', 1842),
('amazon', 1744), ('even', 1693), ('buy', 1691), ('dog', 1661), ('great', 1612), ('better', 1539), ('m', 1475), ('time',
1464), ('first', 1374), ('bad', 1368), ('tried', 1354), ('made', 1315), ('little', 1311), ('sugar', 1294), ('also',
1286)]
```

(d)

```
def entropy(data):
    elist = []
    freqList = []
    freqList.append(0)
    freqList.append(0)
    #compute the sum
    sumFreq = len(data)
    if sumFreq == 0:
        return 0
    #compute the freqList
    for item in data:
        if item[1] == 1:
            freqList[1] += 1
        if item[1] == 0:
            freqList[0] += 1

    for f in freqList:
        c = f / sumFreq
        if c == 0:
            continue
        print c
        elist.append(-c * math.log(c ,2))
    return sum(elist)
```

```
def information_gain(data, word):
    if len(data) == 0:
        return 0
    subset_entropy = 0.0

    inData = []
    notInData = []
    for item in data:
        wordList = item[0]
        if word in wordList:
            inData.append(item)
        else:
            notInData.append(item)
    dataAll = [inData, notInData]
    for tmpData in dataAll:
        val_prob        = len(tmpData) / len(data)
        subset_entropy += val_prob * entropy(tmpData)

    return (entropy(data) - subset_entropy)
```

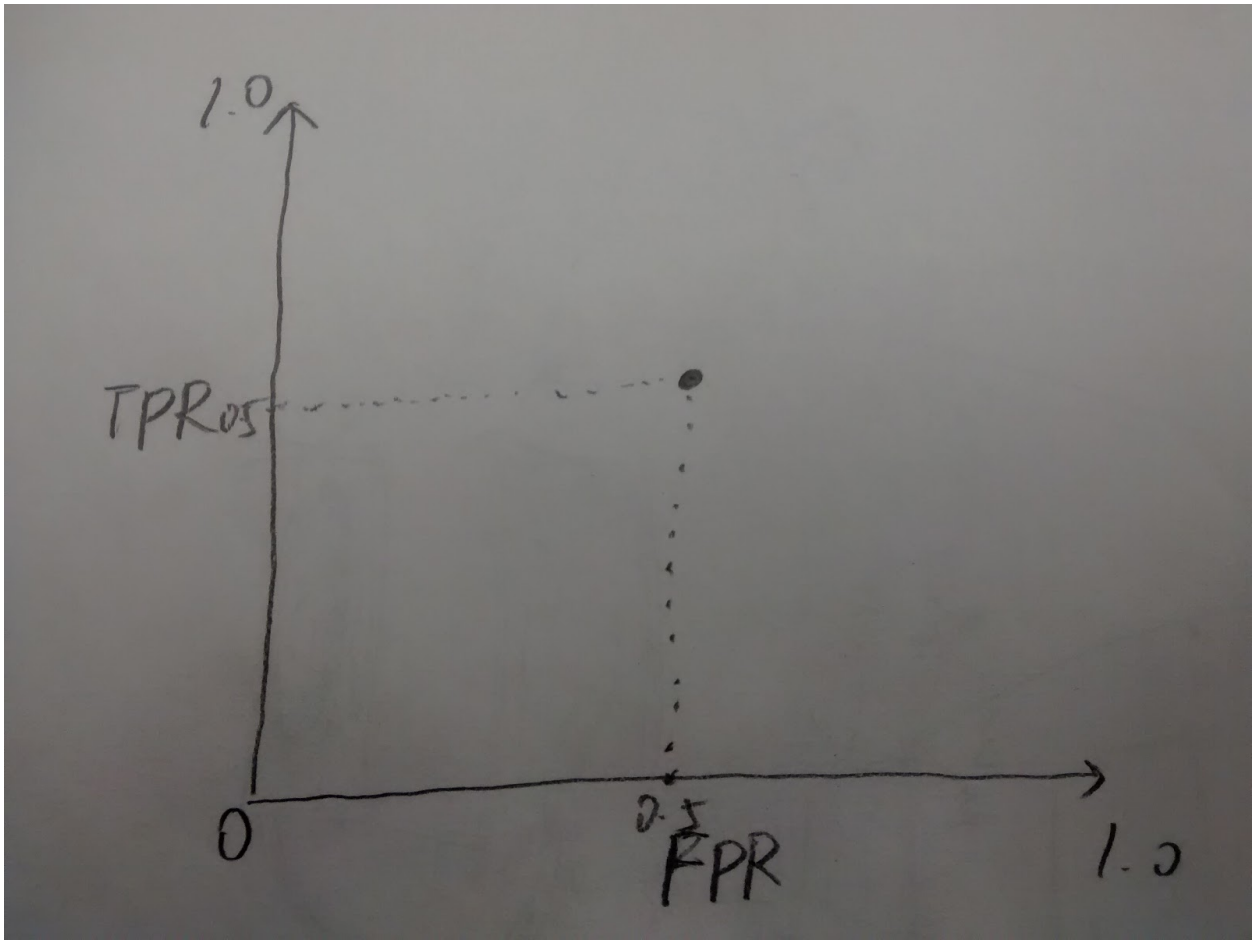
(e) all in code

(f) Accuracy: in 50000 reviews(79.75%)
Take-Away: Most error cases are originally zero but predicted as 1. The reason behind is that the positive reviews are easier to identify because of the obvious words such as “great” “love” etc. However for negative reviews the words are more scattered. This may be improved by assigning weight value of words or use not only binary implementation.

2 Written Exercises

1. ROC Curves

(a) TPR = TP/P = 0.5
FNR = FN/N = 0.5



(b) TPR = TP/P = 30/(30+10) = 0.75
FNR = FN/N = 10/(10+30) = 0.25

(c) TPR = p/1 = p
FNR = (1-p)/1 = 1-p

(d) D is the best. If we assume the parameter p equals 0, then D has the TPR = 1 and FNR = 0

2. Statistics

(a) If head is 1 and tail is 0, then X = {0|1}. P(x=0) = P(x=1) = 0.5

(b)

$$E[X] = \sum_i^n x_i p_i = 1 \cdot P(x=1) + 0 \cdot P(x=0) = 1 \cdot 0.5 + 0 \cdot 0.5 = 0.5$$

$$E[Y] = \sum_i^n y_i p_i = 1 \cdot P(y=1) + 2 \cdot P(y=2) + 3 \cdot P(y=3) + 4 \cdot P(y=4) + 5 \cdot P(y=5) + 6 \cdot P(=6) = (1+2+3+4+5+6) \cdot \left(\frac{1}{6}\right) = \frac{7}{2}$$

(c) Y is the random variable of the value two times of rolls can sum to.

$$E[Y]=E[X_1+X_2]=E[X_1]+E[X_2]=2E[X_1]=2 \cdot \frac{7}{2}=7$$

(d)

$$E[X_1+X_2]=\sum_i^n \sum_j^m (x_{1i}+x_{2j}) p_{1i} p_{2j} = \sum_i^n \sum_j^m x_{1i} p_{1i} p_{2i} + \sum_i^n \sum_j^m x_{2j} p_{1i} p_{2j} \; = E[X_1] \sum_j^m p_{2j} + E[X_2] \sum_i^n p_{1i}$$

Since $\sum_i^n p_{1i}=1$ and $\sum_j^m p_{2j}=1$,

$$E[X_1] \sum_j^m p_{2j} + E[X_2] \sum_i^n p_{1i} = E[X_1]+E[X_2]$$

(e)

$$Var[X]=E[X^2]-(E[X])^2=\sum_{i=1}^6 \frac{i^2}{6} - \left(\sum_{i=1}^6 \frac{i}{6}\right)^2 = \frac{(1+4+9+16+25+36)}{6} - \left(\frac{7}{2}\right)^2 = \frac{35}{12}$$

(f) $Var[a+X]=E[(a+X)^2]-(E[a+X])^2=E(a^2+2aX+X^2)-(E[a]+E[X])^2$

$=E[a^2]+E[2aX]+E[X^2]-(E[a])^2-2E[a]E[X]-(E[X])^2$

$=a^2+2aE[X]+E[X^2]-a^2-2aE[X]-(E[X])^2$

$=E[X^2]-(E[X])^2=Var[X]$