BIMM 185 Lab Report Week 6

Lihui Lu
5/16/2017

Introduction

Since in week5 I have already done finding all the orthologs by using the BiDirectional Model, last week I mostly concentrated on the mini project. Last week we started a mini project on using the Bayesian model to predict if two genes belong to the same operon base on the distance in between.

Mini-projects steps

The mini-project is done in separate steps. At first, the operon set information was downloaded from online database and was parsed using Python. Then the distance within each operon between adjacent genes(h1) and distance of the border of the operon and adjacent genes(h0) are calculated. A Probability Density Function was then computed using out data and the posteriors (the probability of 2 gene being in the same operon giving the distance) are calculated and plotted using the PDFs.

Some import terms and calculations

p(dist=x|h0) - calculated using the distribution of distance of adjacent genes within the same operon
p(dist=x|h1) - calculated using the distribution of distance of adjacent genes with the same direction which one of them is at the border of an operon and the other is next to it.
p(h1) = 0.6
p(h0) = 1 - p(h0) = 0.4

$$p(h1|dist = x) = \frac{p(dist = x|h1) * p(h1)}{\sum_{k=0,1} p(dist = x|hk) * p(hk)}$$

Problems met during the project

1. At first, I was querying all the directions from the database regardless of which organism the genes belong to. The resulted h0 contained a lot of negative values, meaning that the genes overlap with each other. Then I changed the query command to only query directons from E.coli. This has improved the h0 set result to only 9 negative values out of 205 values.

   The sql statement querying all the directons in the database:

   ```
   select genes.gene_id, strand, left_position, right_position from genes inner join(select gene_id, min(left_position) as left_position, max(right_position) as right_position from exons group by gene_id) position on position.gene_id = genes.gene_id order by left_position;
   ```

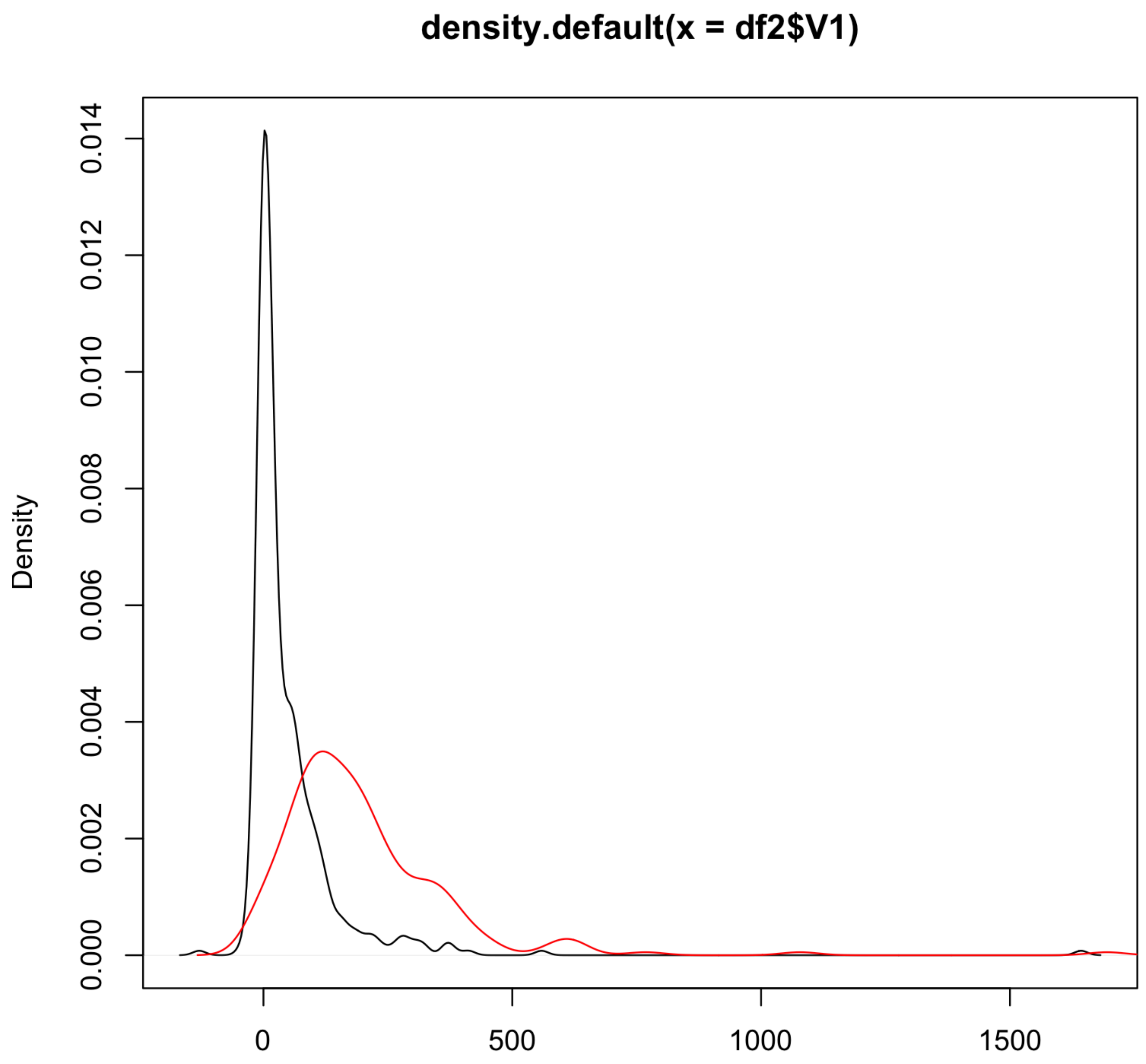
   Then I added one more constrain to gain only directons in E.coli:

   ```
   select genes.gene_id, strand, left_position, right_position from genes inner join(select gene_id, min(left_position) as left_position, max(right_position) as right_position from exons group by gene_id) position on position.gene_id = genes.gene_id where genes.genome_id = 1 order by left_position;
   ```

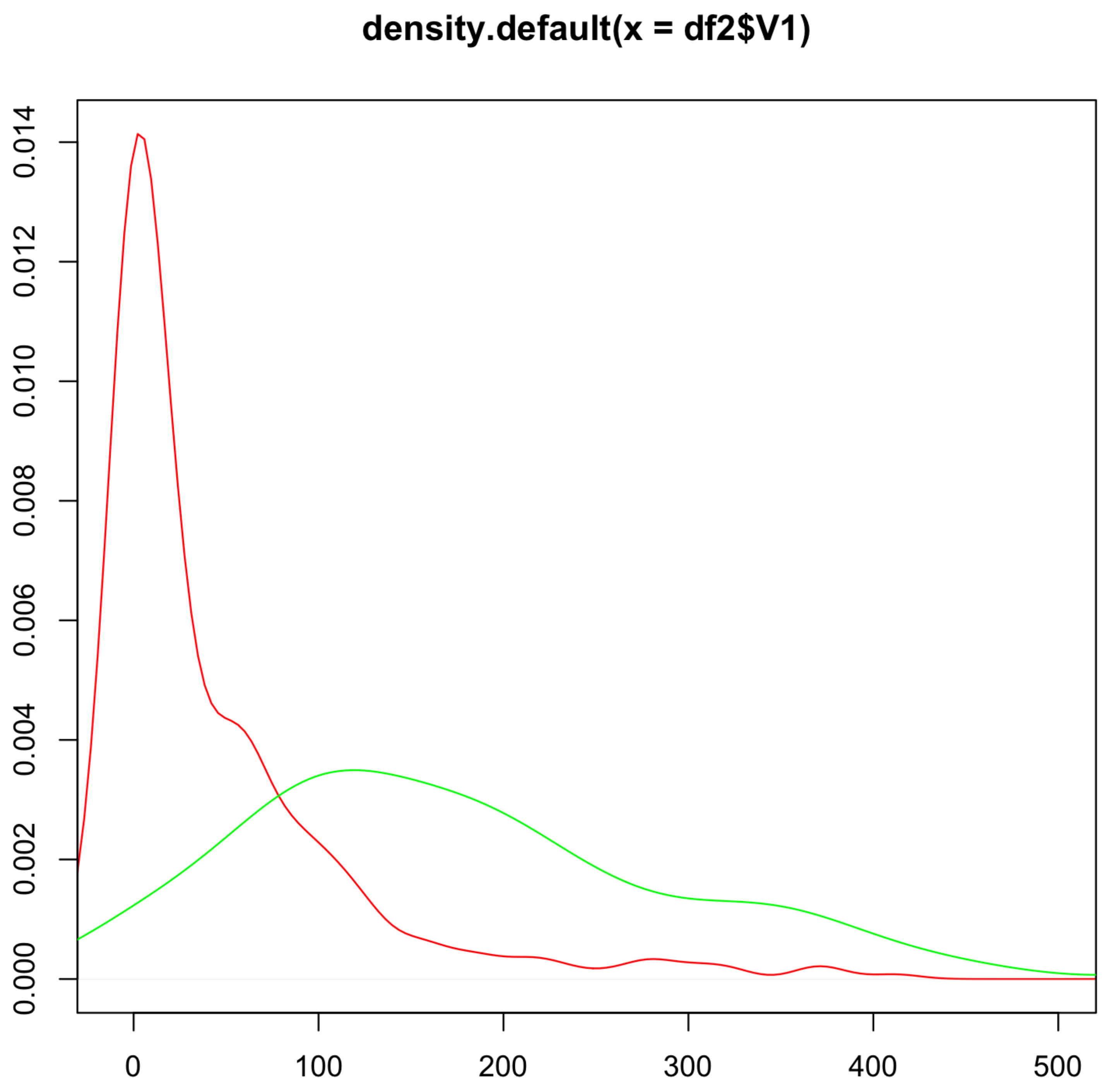
2. While calculating the h0 set, I have found some duplicated values. For example, operon (hemA,prfA,prmC) and operon (ychQ,ychA,kdsA) are found next to each other. Thus, the distance between prmC and ychQ was calculated twice in the result set. I think that we should remove this kind of duplication in order to improve our training dataset. However, I didn't think of a good way to detect this kind of duplication and thus decided to leave it in the result set for now.

Results and plots

p(dist=x|h0) and p(dist=x|h1) are calculated using scipy.state.gaussian_kde(data) function and plotted using both R and python.



A Zoomed plot in the region(-10,600) was also done to visualize the 2 separated peeks better. It can be observed that the peek of the PDF for h1(red) is at around distance = [10,20] and the peek has value 0,014 while the peek of the PDF for h0(green) is at around distance = [90,120] and the peek has value 0.004. The PDF function for h1 is much denser than PDF function for h0.

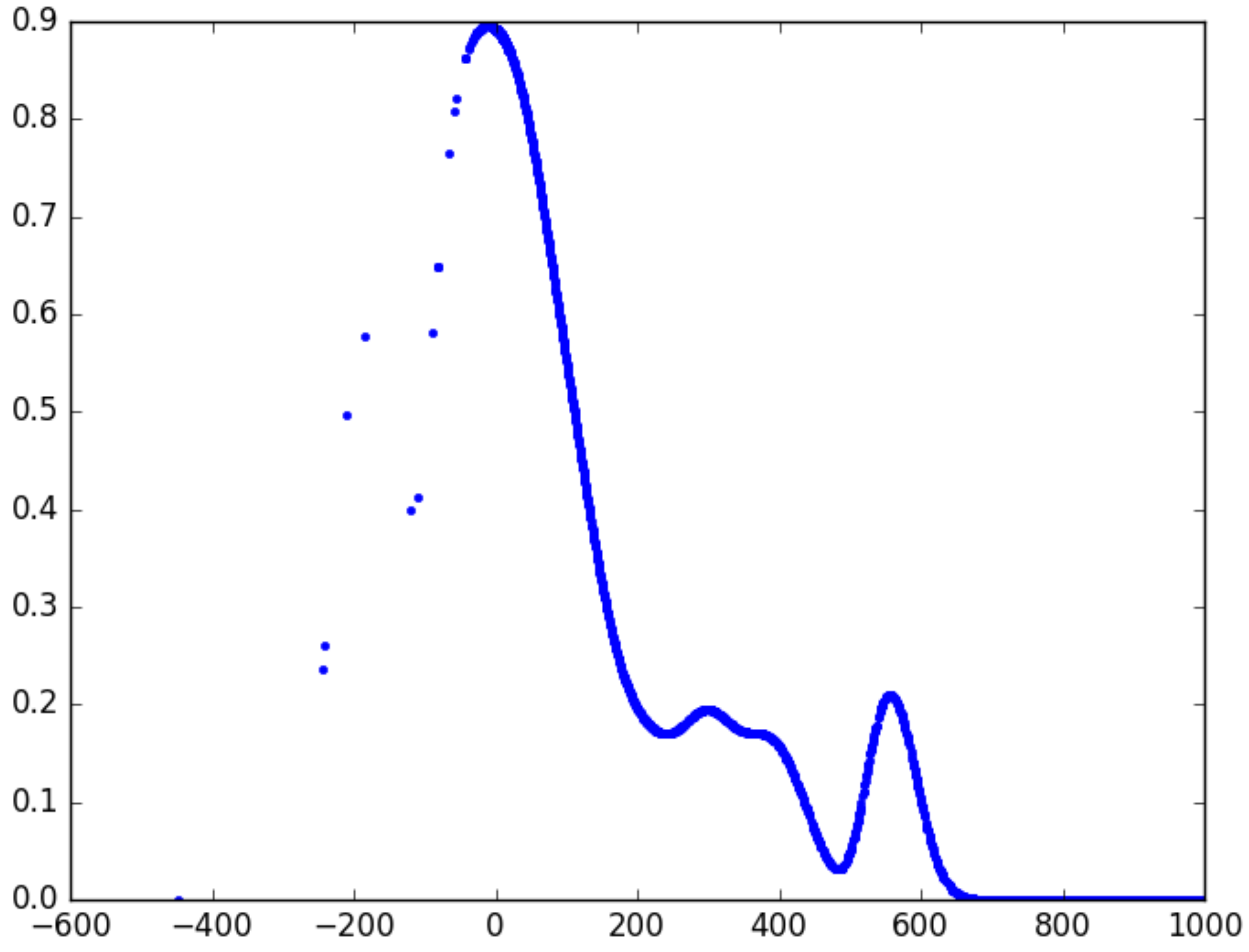


Then the posterior p(h1|dist=x) is calculated using all gene pairs in the same directon in E.coli and plotted below. The gens are queried from the sql database and iterated through using a forloop. The code can be found below:

```
#The directons are queried from the sql database using the sql statement above

kde0, kde1 = pdf_calculate(h0, h1)
posts = []
distances = []
directon = []
```

```
direction = ""
for d in directons:
        if direction == "":
            directon.append((d[3],d[4]))
            direction = d[2]
            continue
        elif direction == d[2]:
            directon.append((d[3],d[4]))
        else:
            for i in range(len(directon)):
                for j in range(i+1, len(directon)):
                    distance = directon[j][0] - directon[i][1]
                    #print(distance,kde1(distance),kde0(distance))
                    if kde1(distance) * 0.6 == 0:
                        posts.append(0)
                    else:
                        pos = kde1(distance) * 0.6 / (kde1(distance) * 0.6 + kd
e0(distance) * 0.4)
                        posts.append(pos[0])


                    distances.append(distance)

            directon = [(d[3],d[4])]
            direction = d[2]
```

The probability of the 2 gene being in the same directon given their distance are around [0,100] is very high. However, 2 peeks are observed from the posterior plot, which requires further investigation.



Source code: all source code can be found in the git repository https://github.com/luna5124/BIMM185/tree/master/Week6

Discussion

Last week we have learned how to utilize the database and other available sources. We have also learned the process of starting a project with a hypothesis to designing a model and prove that model. It is very important that we understand the data that we are processing and the data that we are going to generate to make the best use of those data. It is also important to discover some special cases as well as problems during computation process and use biological knowledge to solve those problems.