Project Report

Ye Zhang

In this month, I have built HMM, Decision Tree, SVM and DBN models.

K-means clustering + HMM

First, I use K-means clustering for the waiting time to discretize the waiting time value and produce about 25 clusters and I use "time of day+day of week" as the feature for office 548. The hidden node(waiting time) has 25 possible values, and the observation(time) has 330 possible values. Then I use MLE to calculate necessary possibilities and Viterbi algorithm to predict the hidden waiting time. In figure 1, the horizontal axis denotes the time in which 0-65 denote "Monday 7:00-Monday 17:50", 66-131 denotes "Tuesday 7:00-Tuesday 17:50", 132-197 denotes "Wednesday 7:00-Wednesday 17:50", 189-263 denotes "Thursday 7:00-Thursday 17:50" and 264-329 denotes "Friday 7:00-Friday 17:50". I obtain the average predicting error that is above 0.6 hour. Then I try to manually decrease the frequencies of overly high counts (waiting time 0 to waiting time 0) by ⅙, ⅓, ⅔ and ⅚, the corresponding results are shown figure 2,3,4 and 5. From these results, we can see that if we modify the overly high counts, we can obtain better trends in waiting time, but the error will increase.

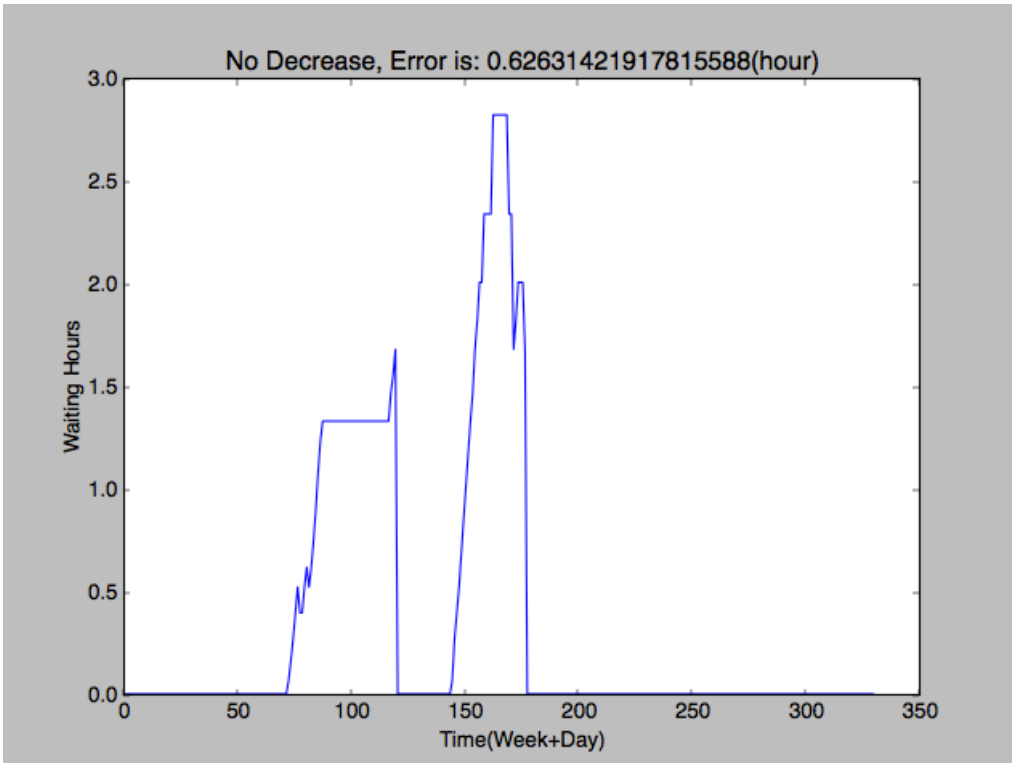

Figure 1

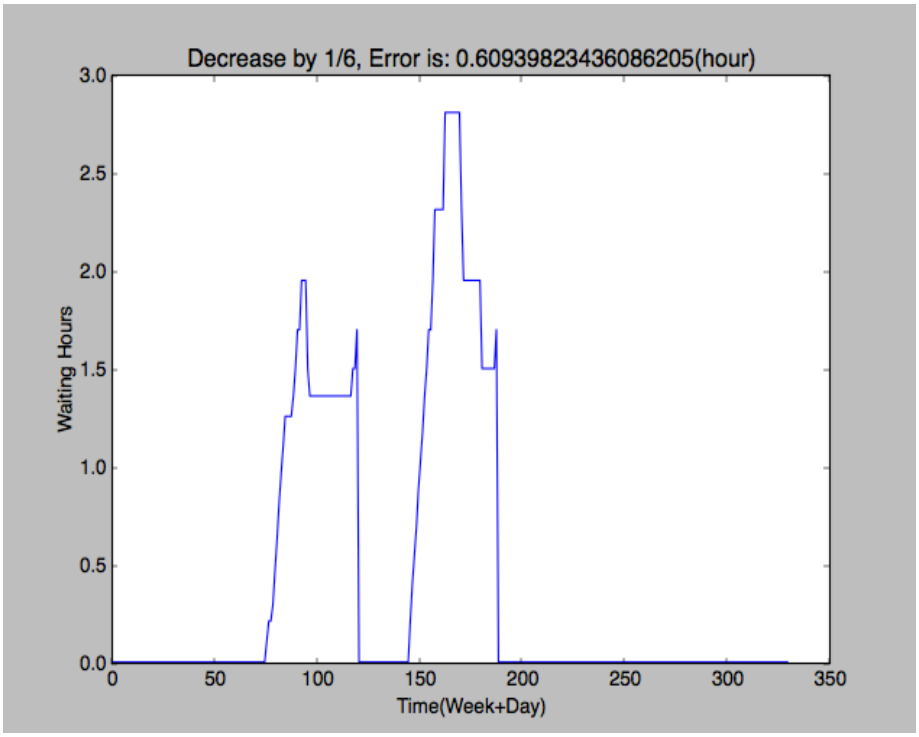

Figure 2

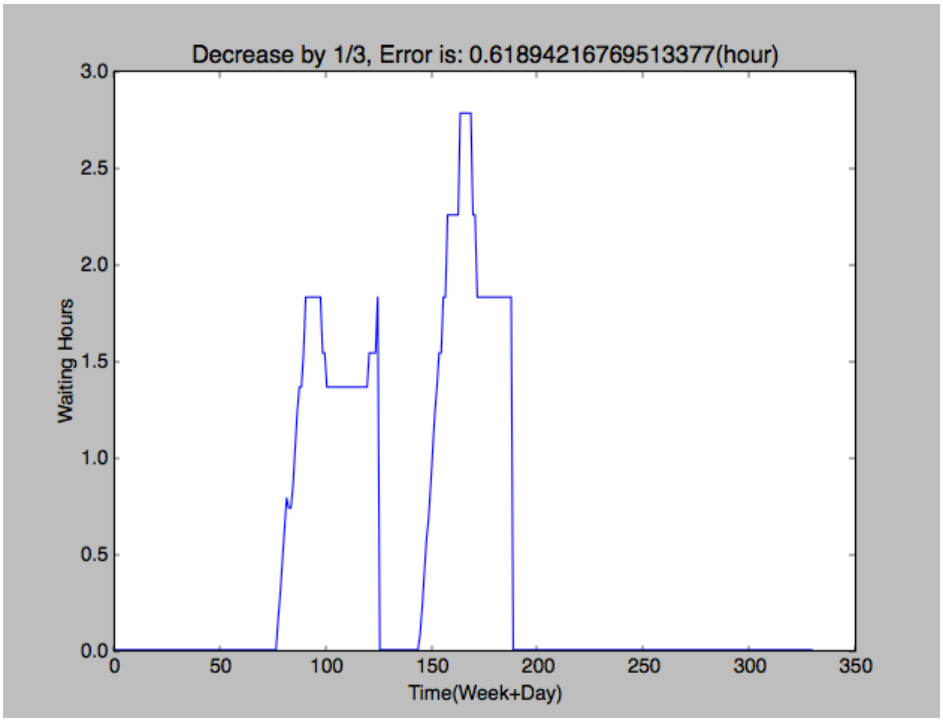

Figure 3

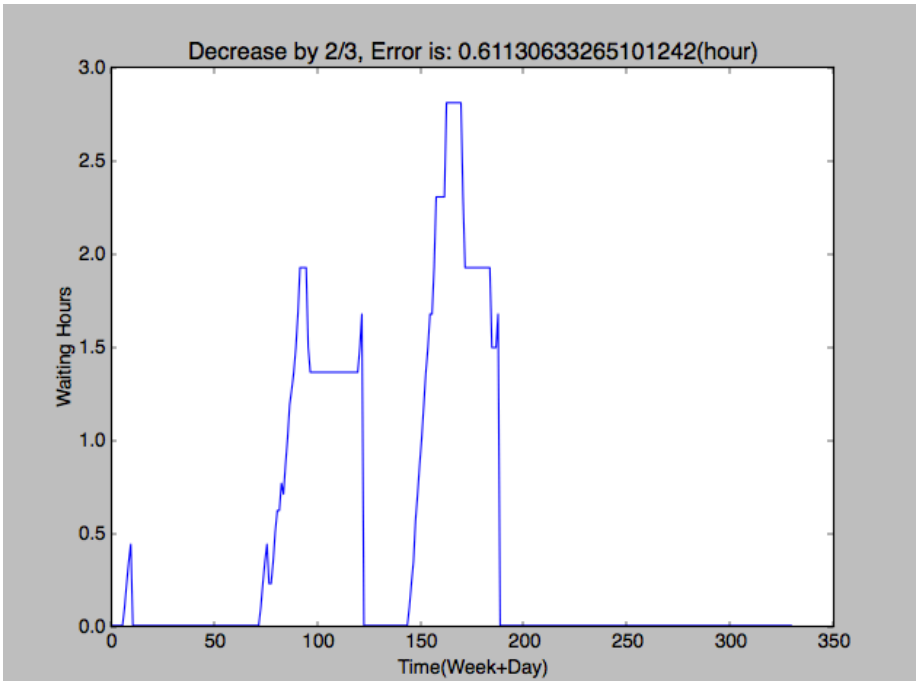

Figure 4

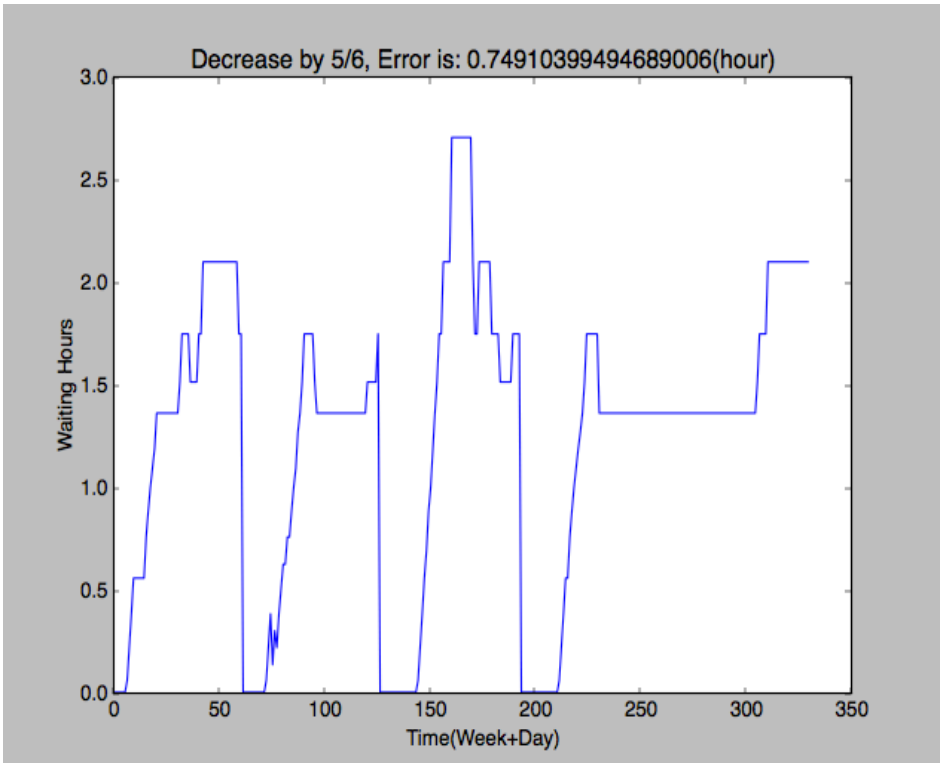

Figure 5

Due to the lack of training data, so many states might cause severe over fitting which pseudo count cannot help avoid. So I decrease the number of possible hidden states and assume some states("busy", "occupied", "empty" and so on)for waiting time instead of numerical values to do prediction. This time, the curve of waiting time against time in figure 6 seems much better which indicates that HMM with enough training data(compared to the size of parameters) could predict the trend in the waiting time. However, the error percentage is about 61.4% which is pretty high. Then I try to decrease the number of hidden states to 3. This time, the error rate is around 50% which is much better.

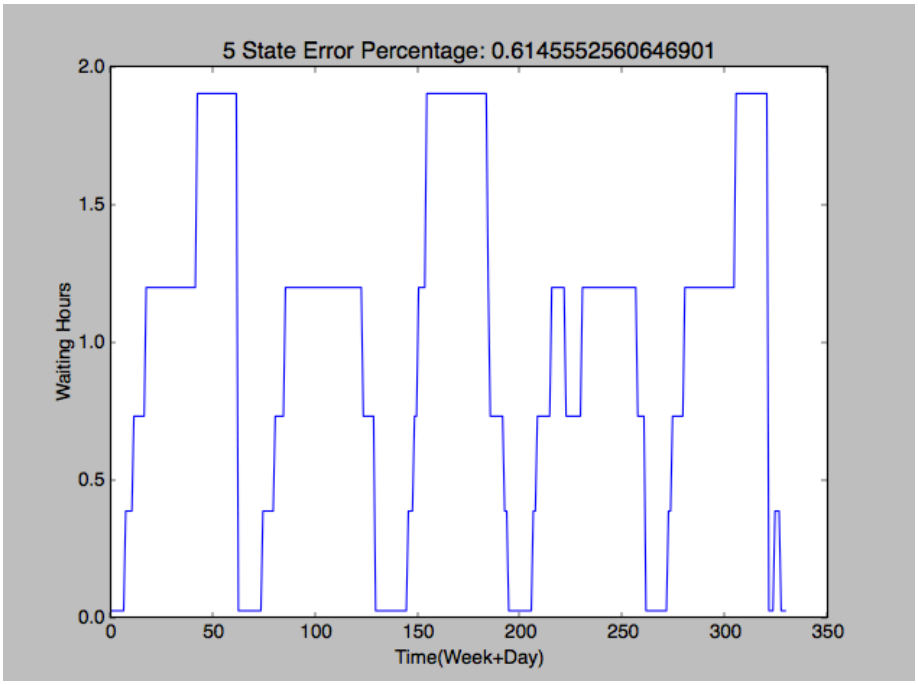

Figure 6

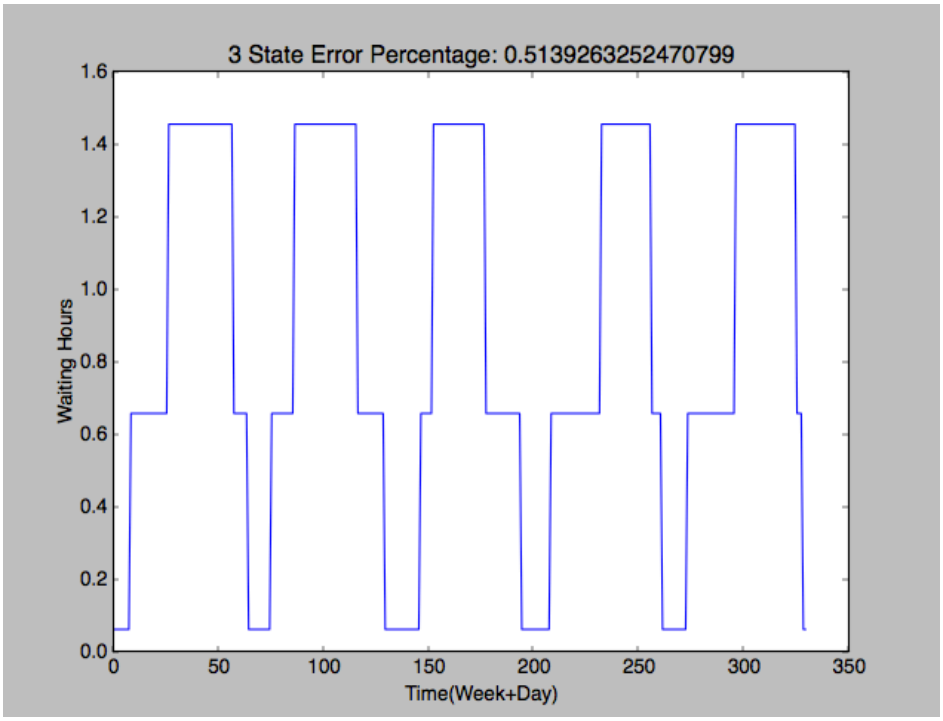

Figure 7

K-means Clustering + Decision Tree and SVM

I also try decision tree and SVM combined with K-means clustering(5 clusters). The features include previous waiting time and "weekday + time of day". Then I compute the correctly classified labels in test data. The accuracy of SVM is 61.9% and the accuracy of Decision tree is 63.3%. I haven't tried more clusters yet.

Equal-height histogram + Dynamic Bayesian Network

I decide to use equal-height histogram instead of K-means clustering to discretize the continuous waiting time values. First, I observe that there are many days when all waiting time value equal 0 which led Maybe these days are holiday, so I remove all these days from the dataset for office 548. Then I build an equal-height histogram over all waiting time values for office 548 which has buckets with different width but equal frequencies. Another change is I treat "Week" and "Time of day" as two variables. So there are three variables. The hidden node is still "waiting time", and the two observation nodes are "week' and "time of day" which both are children of "waiting time". Then I use MLE to obtain the necessary parameters of DBN and use Viterbi algorithm to calculate most probable hidden path given the observation sequence in the testing data. Note that this step is different from HMM. In the above HMM process, I manually create a continuous sequence from the earliest time to latest time in a week. This sequence denotes a single complete possible time sequence. But here in DBN, I directly use the time sequence in

testing data. First, I split the whole test data into several sequences. Then I calculate the average error hour over all the test sequences. The following is the result.

Number of Bucket	10	15	20
Error (Hour)	0.602	0.654	0.671

Next step

Next step I will check the method I have used to do prediction. In Viterbi algorithm, I assume all the nodes in a sequence are observed, because we know the time value in each slice and do inference on waiting time sequence. But in this project, we might aim to predict the future waiting time given the current time and the previous waiting time. I'll consider how to this. Maybe applying forward-backward algorithm or only forward algorithm can achieve this goal.

Since I use different methods to prepossessing data and different metric to calculate different models' errors, next step, I will analyze the results I have obtained so far and consider how to measure each model's accuracy in a better way and improve the current result. Also, I'll check the details in each method I have used to avoid mistake.