Question 2

1 Team

Team Members	*Shuyang Ye*	*Ning Shen*
Student ID	*96481163*	*70533633*
CS ID	*l8n8s*	*i0c1p*

Kaggle Team Name	*123321*

2 Solution Summary

To predict the future path of ego cars, we first decide which machine learning model to be adopted. Considering the training set only cover the track from last 1 second, and the task is to predict 3 second forward, AR model is not suitable. On the other hand, KNN is a good choice. We just need to find the similar cases. Here, "similar" means the tracks in last 1 second are close enough in regard to Uclidean distance. "Others" cars and "agent" car are considered seperately. First, we calculated the similarity of "others" between training set intersection and test set intersection. Second, we calculated the similarity betweem "agent" cars. Then, we pick k nearest intersections(examples) based on the overall distance and average the corresponding "agent" car's track in 3 second to be our prediction.
Hyperparameters are the k from KNN and a weight w which is used to balance the influence from "others" and "agent". We use validation set finding that k=6 and w=1 can yield best validation error. To conclude, in our model, comparing "agent" car is adequate for predicting the future track.

3 Experiments

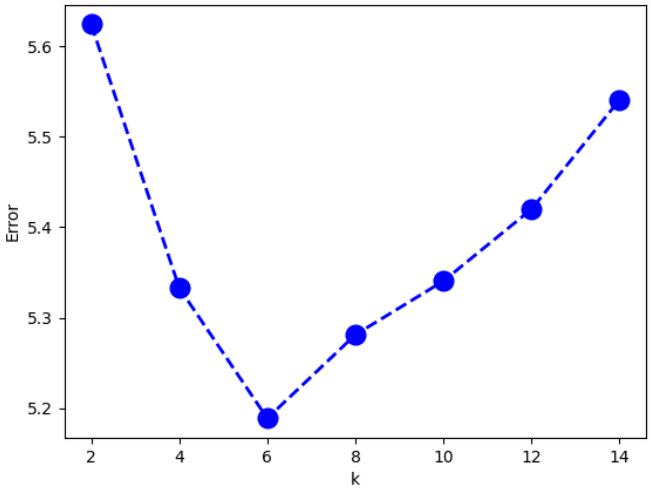
The main experiment is to find the best hyperparameters. As we explained in the Summary, the similarity between two intersections comes from two different sources: "others" and "agent". During the comparison, we realize "agent" must pair with "agent", "others" pairs with "others". For "others", we establish the distance matrix of size 9*9. Each element d_{ij} is the distance between the track of car_i from intersection 1 and of car_j from intersection 2. For the intersection with less than 9 "others", the corresponding d_{ij} will be filled with np.inf. Once the distance matrix is completed, we apply the greedy search algorithm to find the similarity. We first find the smallest d_{ij} in the matrix, which means car_i and car_j are paired. Then we delete row i and column j and search for the smallest $d_{i'j'}$ from the new matrix. Repeating above steps until there is no available d_{ij}. Averaging all the d_{ij} found yields the similarity (s_1) of "others". For "agent" part, we directly compute Uclidean distance between two "agent" cars (s_2). The total similarity is weighted by w: $s = (1 - w) * s_1 + w * s_2$. For each example from validation set, we sort the total similarity s to find k nearest neighbors, in other word, k most similar intersections from training set. The averaged y from these intersection is the predicted track y_{pred}. The validation error is evaluated by the different between y_{pred} and y_{val}. Figure 1 shows how does the hyperparameters impact the validation error.
The experiment result reveals the fact that there is no need to include "others" into the similarity. Thus, we have done the feature selection: only the data of "agent" car is selected.

4 Result

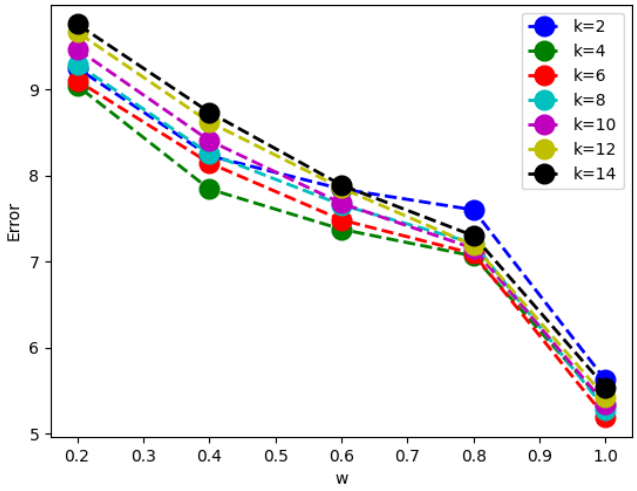
Team name	Score
123321	0.54513

5 Conclusion

We used KNN as our model. An experiment of hyperparameters choosing is done which also help us to select feature. We found similar "agent" cars tend to have the close track in 3 second.



(a) Error vs k(w=1)



(b) Error vs w

Figure 1: hyperparameters scan