A KNN-Based Model for Road Speed Inference

# High-Level Description of Our Proposal

First, we formulate this problem as univariate time series prediction from our understanding of the problem definition and task requirements. But the task is not like traditional univariate time series task because we are not given the date of the prediction target and the information (observation data points) of both before and after prediction are provided, for example, the speed of early morning (0:00 am - 5:55am) and noon (11:00am - 3:55pm).

It is reasonable to think of a potential solution that consider the features of time series in prediction file and use these features to find similar ones that belong to historical data. The instances (neighbors) with similar features are then used to infer the speed of interested time period in the prediction file. This general idea serves as the foundation of our solution. Since it is much like K-Nearest Neighbors (KNN) model, we name our solution KNN-based. The intuitive illustration is shown in the last page.

# Machine Learning Pipeline Overview

The team spent almost more than half of total time preprocessing the data. The critical steps in our **pipeline** includes:

1. extract historical speed of target road from given raw GPS files.
2. missing value imputation.
3. handle outliers.
4. training and validation datasets preparation.

The splitting of training and validation datasets in historical data is to find the best model/hyperparameters before we move on making final predictions on real datasets. Since we want our model to perform well on any given dates. Seven days of speed data are extracted individually and set as validation sets, which leaves the remaining training sets. After these manipulations, we have 7 folds of validation/training data. The seven validation sets are corresponded to and randomly sampled from each day of seven days of a week. The idea here is to develop seven models on these seven folds and get seven different performance of results, the final validation score is the average of them. So the candidate with best validation score is supposed to be the model performing the best on all seven days averagely.

1. KNN implementation (Pease refer to the final page of this report.)
2. smoothing the output of model.

For the purpose of making our model robust and more adaptive, we designed several import **hyperparameters**:

1. the degree of polynomial curve to fit the historical speed in order to cut off variance (noise).
2. the definition of similarity.
3. the number of K (neighbors)
4. the weights for K to average speed
5. the degree of polynomial curve to smooth final results.
6. the weights of data points when smoothing the final results. For example, it makes sense if more weights are assigned to the points near our prediction time period (like 5:55, 11:05 etc.).

These hyperparameters are searched in training/validation sets in a space we designed. And the optimal combination of them are chosen to make predictions in prediction file for each direction.

# Discussion

The proposed model is easy to deploy and could achieve decent prediction results as our validation results suggest. The main advantages of our model are:

1. Not like traditional time series prediction models only consider the information beforehand, our model could utilize the information after the time period of predictions, which renders more accurate results.

2. the proposed model is capable of predicting speed on days with accidental events, e.g., holidays, soccer games. Because some history days are likely to have similar events and could be used to make predictions.

3. Also, no mathematical formula is needed to make predictions because of the nonparametric nature of KNN. The trend of speed is stored implicitly inside the history.

4. The smoothing at the preprocessing stage is proved to be effective in decreasing noise/errors. And the extra steps of smoothing both before and after actual modelling improves the performance substantially.

The limitation is that it would be error-prone if the historical data is insufficient since there are few samples for model to search for. In our case, 30 days of data is somewhere in the middle. In real practice, Model would definitely behavior better if more and more historical data is incorporated.

# Graphic View of Proposed Model

