REPORT

Approach:

The intuition behind the approach we have followed is, the probability for a particular vector depends on its own distribution as well as influenced by neighboring points. The more common values are more likely to occur. Based on the idea, we first found out relative frequency of each of the vector which is simply number of times observations have repeated / total number of observations in the file.

Second, to consider the effects of neighbors, we find 20 closest neighbors using k-nearest neighbor. We then define probability of vector mv as sum of relative frequency and weighted average of relative frequencies of its neighbors.

Relative Frequency = # of repetitions of observations/ # total observations

Probability Vector = Relative Frequency + [$\sum$ (Relative Frequency/distance)]/N (N=20)

Implementation details:

We implemented the code in python (included the ipython notebook used in submission). We first read the speed, flow and occupancy files for each of the zones. Next, we grouped them together based on vectors and found out frequency of each vector which was then used to calculate relative frequency. The next step was to find k nearest neighbors. We used knn function from python library to find 20 nearest neighbors for each point and their distances. Using the distances, weighted relative frequency was calculated for the neighbors which was summed up with relative frequency of point to calculate the final probability. The vectors were then sorted and final report was made on 0.1 percent of the data.

Intermediate Results:

We initially plotted all the vectors in to the 3D space to visualize the data. We got the figure as shown above. Based on the figure we came up with the idea of relative frequencies and also how probability distribution of the vector is affected by neighboring point. If a vector is far off from cluster of values, most probably, it is an outlier. But we couldn't simply apply clustering as the number of points was too large. So we grouped all vectors with same value and generated relative frequencies based on those counts. Negative and improbable vectors were left as it is, as their probability was almost 0. Vector with missing values were removed before processing as we did not want to corrupt the data.

Challenges:

The major challenge during implementation was the size of input files which caused frequent memory errors. To overcome the difficulty, we made batches of data and processed them batch by batch and then finally merged the results.

Results and Observations:

- Based on the results we found that data is quite heavily skewed towards 0,0,0 vector. It would be interesting to figure out why.
- Probability distributions are affected by their neighbors. It would be interesting to put a penalty term to adjust importance of weighted sum of relative frequencies of neighbors to give it more or less importance.
- Probabilty thus calculated for each point is based on its immediate neighbors. We can perform subsequent iterations to update the probability value for dispersing the total effect.