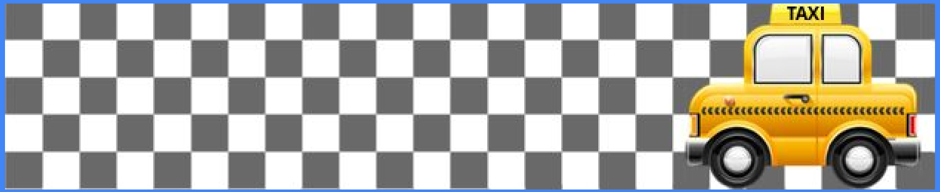
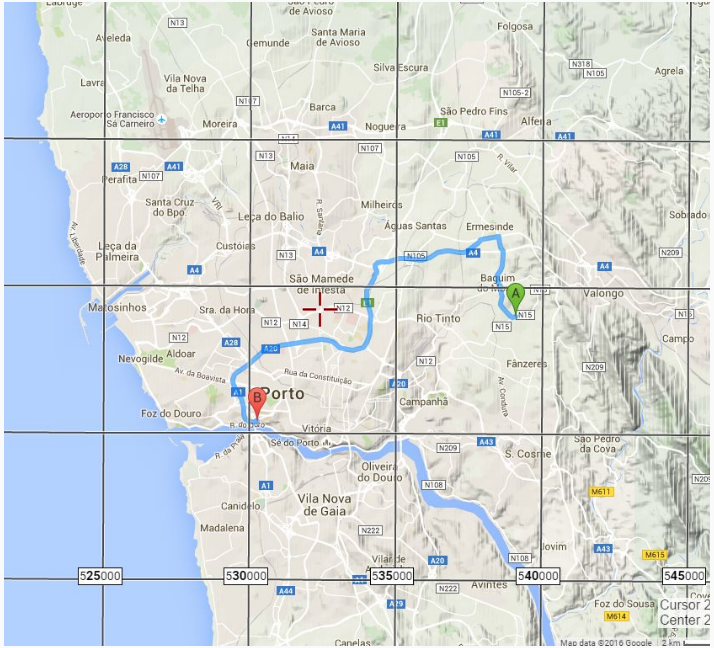
Taxi Trajectory Analysis

Prateek Agrawal, Paul Cherian, Dong Hyuk Kim, Danny Lee



Introduction



- Electronic dispatch systems (one-to-one) have replaced VHR-radio sytem (one-to-many).
- Taxis do not enter their drop-off location
- Predicting the drop-off location of taxis in service using their spatial trajectories

Data Description

- A year's worth of 1.7 million observations of taxi trips (7/1/2014~6/30/2015) of 442 taxis operating in Porto, Portugal.

	TRIP_ID	CALL_TYPE	ORIGIN_CALL	ORIGIN_STAND	TAXI_ID	TIMESTAMP	DAY_TYPE	MISSING_DATA	POLYLINE
0	1372636858620000589	C	NaN	NaN	20000589	1372636858	A	False	[[-8.618643,41.141412], [-8.618499,41.141376], [...
1	1372637303620000596	B	NaN	7	20000596	1372637303	A	False	[[-8.639847,41.159826], [-8.640351,41.159871], [...
2	1372636951620000320	C	NaN	NaN	20000320	1372636951	A	False	[[-8.612964,41.140359], [-8.613378,41.14035], [-...
3	1372636854620000520	C	NaN	NaN	20000520	1372636854	A	False	[[-8.574678,41.151951], [-8.574705,41.151942], [...
4	1372637091620000337	C	NaN	NaN	20000337	1372637091	A	False	[[-8.645994,41.18049], [-8.645949,41.180517], [-...

- TRIP_ID is the unique ID for all trips
- TAXI_ID is the unique taxi numbers
- TIMESTAMP is the date and time of the trip
- POLYLINE includes GPS coordinates of the taxi trip that are taken in 15 second intervals

Data Pre-Processing

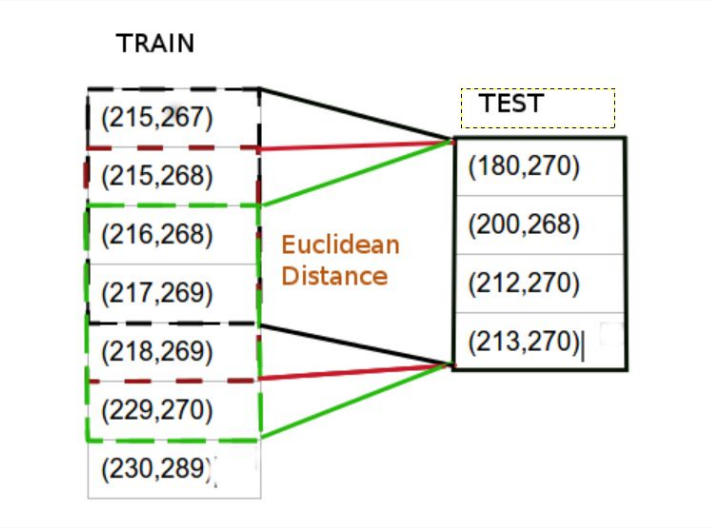
- Divided the area of the city of Porto, Portugal into a grid in latitutude and longitudes to bin different locations (grid resolution: 500m).
- Converted all latitude and longitude from degree to meter to plot them in 2D space.
- Timestamp is converted to date time format from posix format.
- Last grid as end destination grid.
- 1st Label as starting grid

Methods Used

- KNN clustering
- Neural Networks

K-NEAREST NEIGHBOURS

- Find K nearest trips that have similar trajectories.
- Use N latest coordinates from test set.
- Create a sliding window to calculate similarity across train coordinates
- Sum of euclidean distance as similarity metric



Multiprocessing

- Training data huge: 1.7 million records
- Use *multiprocessing* package in python to distribute the workload
- Creates seperate processes for calculating similarity with each training data point.
- Used vectorized operations

Parameter Tuning

- Did 5 fold cross validation on subset of training data to tune parameters
- The evaluation metric is Mean Haversine Distance

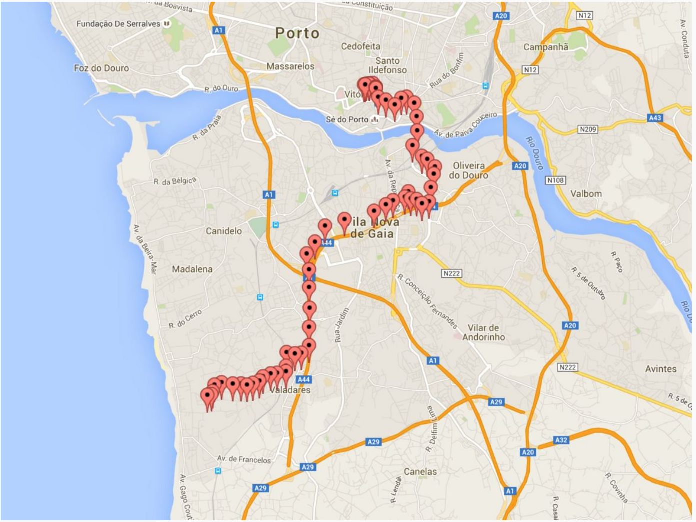
$$a = sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + cos\left(\phi_1\right) cos\left(\phi_2\right) sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)$$

$$d = 2 \cdot r \cdot atan\left(\sqrt{\frac{a}{1-a}}\right)$$

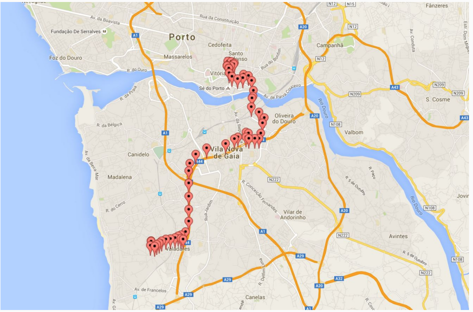
Parameter Set (K,N)	Mean Harvensine Distance
(3,5)	2.621
(5,5)	2.582
(5,3)	2.653
(5,7)	2.581
(7,7)	2.618

Results

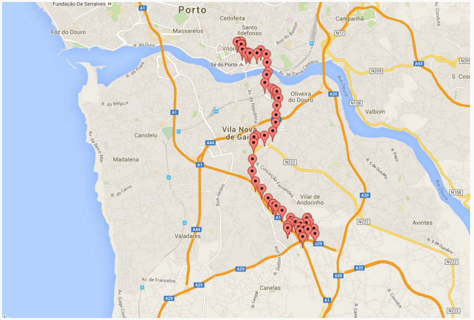
TEST OBSERVATION



Neighbour 1



Neighbour 10



Artificial Neural Network (ANN)

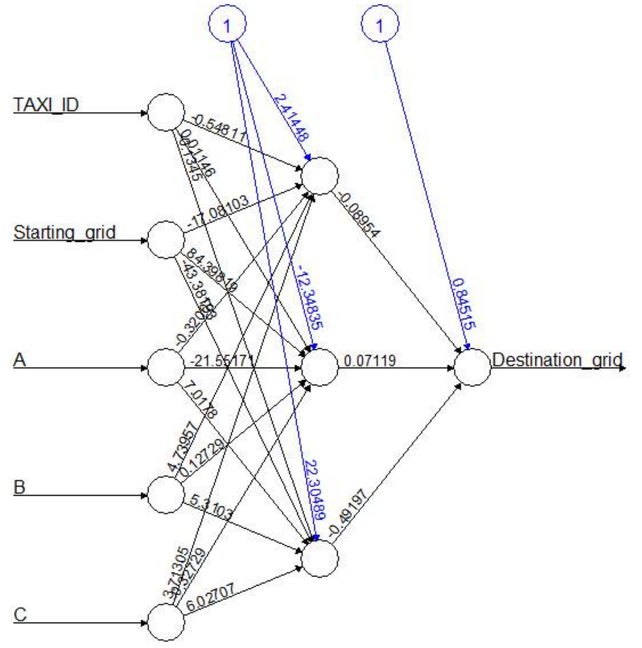
- Modeling functional relationships between covariates and response variable
- Used the package neuralnet (R)
    - Trained a multi-layer perceptrons model to classify each trip on basis of destination grid
    - We used 3 nodes i
    - Used Backpropogation algorithm to reduce the errors
    -
    - Supervised Learning Al $o(\mathbf{x}) = f\left(w_0 + \sum_{i=1}^{n} w_i x_i\right) = f\left(w_0 + \mathbf{w}^T\mathbf{x}\right)$

Approach

- Conversion of categorical variable into dummy variables.
- Encoded labels with value between 0 and n-1
- Normalization
- Random split of data into a train and a test set

Graphical Representation of the Model

- We trained the model based on label-encoded Taxi-ID's, Starting grid label and CALL_TYPE.
- The output of the analysis predicts the label in which the trip ends
- Performed cross-validation with the test data, we got an RMSE of 0.092



Conclusions

- Testing the Neural Network on kaggle test data, we got the mean harvensine distance of 2.591
- The KNN algorithm gave a mean harvensine distance of 2.623
- Would have put us in top 25%

Questions?