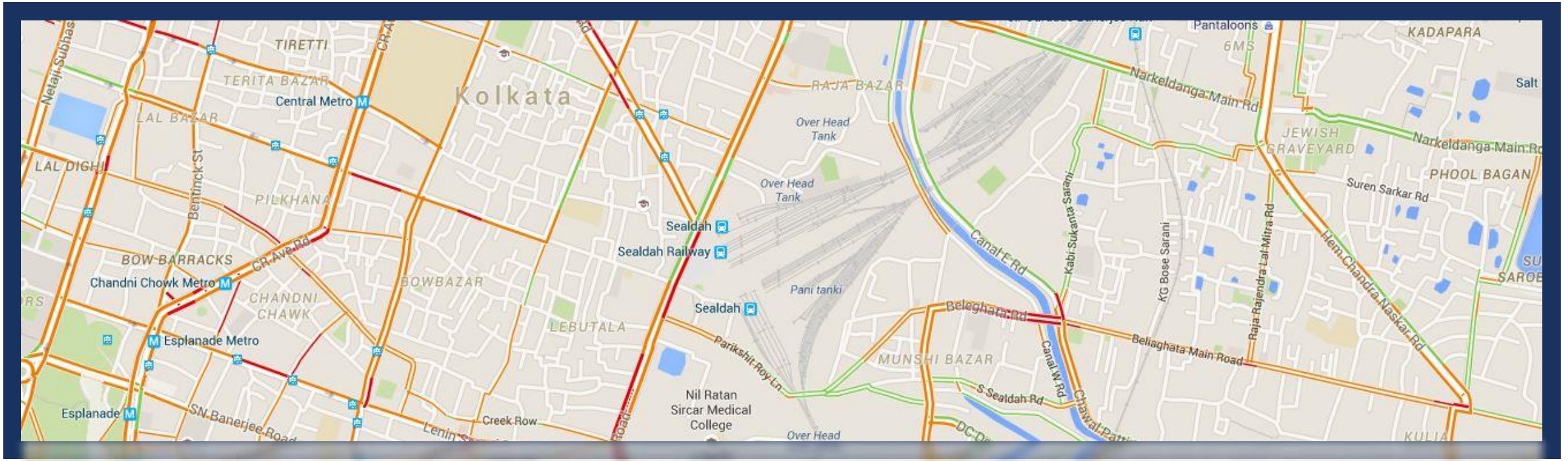
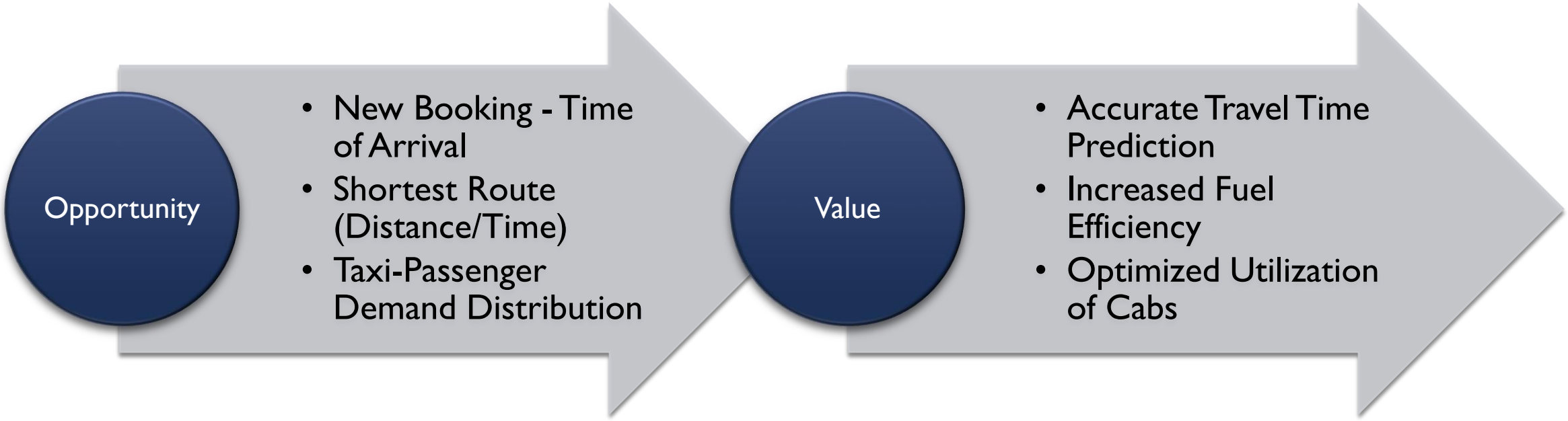
CAB TRAVEL TIME PREDICTION

- BASED ON HISTORICAL TRIP OBSERVATION



PROBLEM DEFINITION



Our project involves developing a model for predicting the travel time for a particular cab along different routes in a rectangular map of Porto. The output is based on several factors, like –

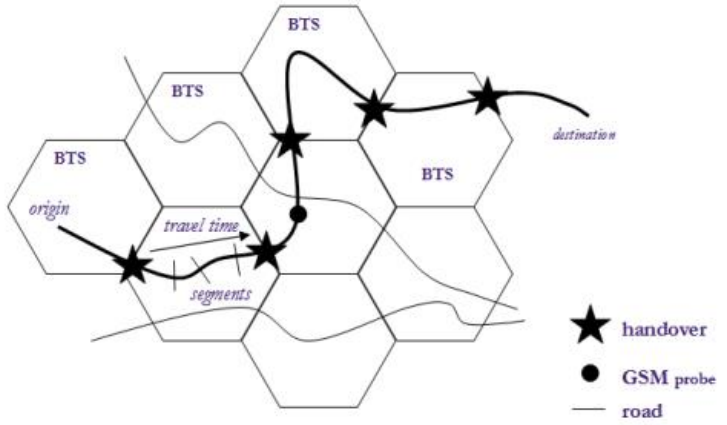
- day of the week,
- time of the day and other features

derived from Floating Car Data.

FLOATING CAR DATA ESTIMATION

Floating car data are position fixes of vehicles traversing city streets throughout the day.

- Advantages
  - Most Inexpensive Data
  - Generally very accurate (GPS)
- Disadvantages
  - FCD is usually sampled infrequently
  - High Density of data is required to get meaningful travel time predictions.
- Use
  - Trajectories of Single Vehicles,
  - Location and Time of Lane Changes,
  - Traffic Density (vehicles per kilometer),
  - Traffic Flow (vehicles per hour),
  - Vehicle Speed
  - Length and Position of Traffic Jams



DATA DESCRIPTION

- The dataset contained 9 features and a total of 17,10,760 observations of various parameters of taxi movement on Porto, Portugal.
- Features :
  1. **TRIP_ID:** (String) It contains an unique identifier for each trip.
  2. **CALL_TYPE:** (Char) It identifies the way used to demand this service.
  3. **ORIGIN_CALL:** (Integer) It contains an unique identifier for each phone number which was used to demand, at least, one service. It identifies the trip's customer if CALL_TYPE='A'. Otherwise, it assumes a NULL value.
  4. **ORIGIN_STAND:** (Integer): It contains an unique identifier for the taxi stand.
  5. **TAXI_ID: (Integer):** It contains an unique identifier for the taxi driver that performed each trip.

DATA DESCRIPTION

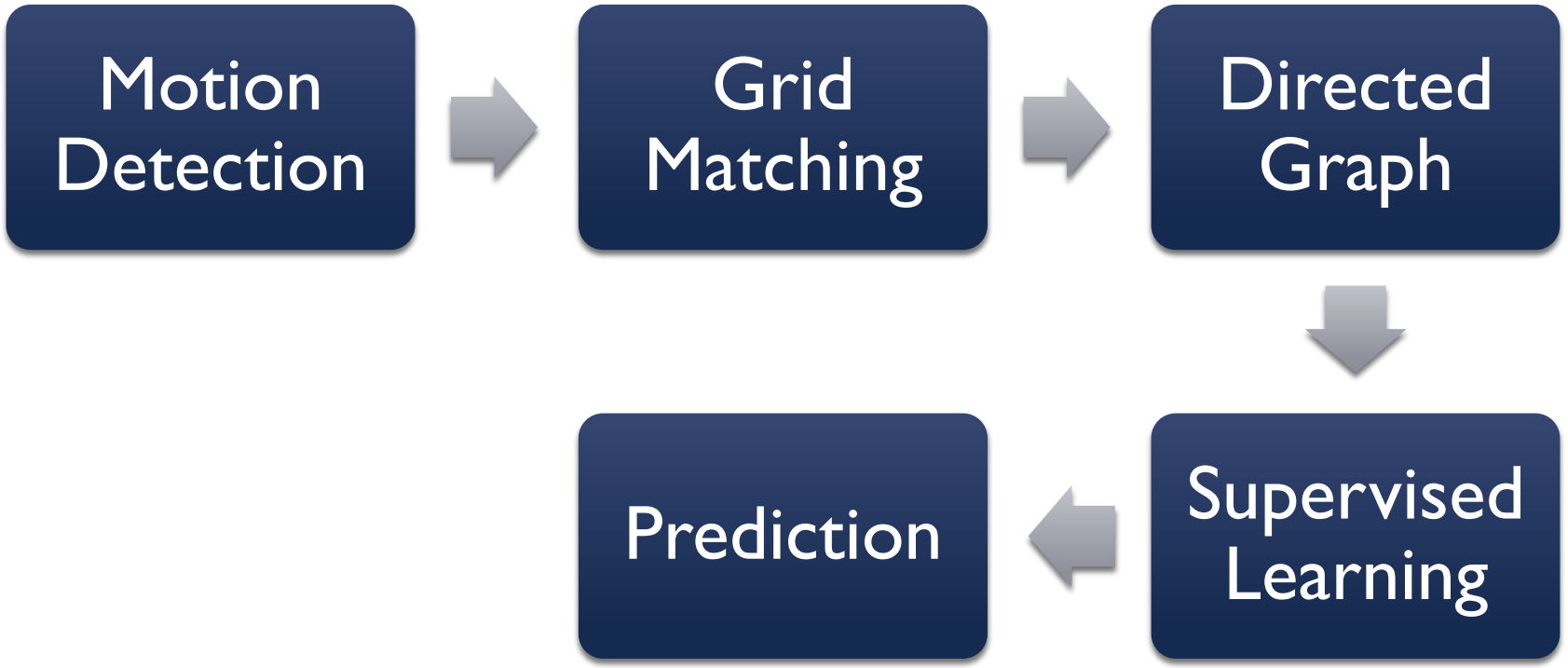
- The dataset contained 9 features and a total of 17,10,760 observations of various parameters of taxi movement on Porto, Portugal.
- Features :

6. **TIMESTAMP**: (Integer) Unix Timestamp (in seconds). It identifies the trip's start.
7. **DAYTYPE:** (Char) It identifies the type of the day in which the trip starts. (Holiday, Day before Holiday, Weekday)
8. **MISSING_DATA:** (Boolean) It is FALSE when the GPS data stream is complete and TRUE whenever one (or more) locations are missing.
9. **POLYLINE**: (String): It contains a list of GPS coordinates (i.e. WGS84 format) mapped as a string. Each pair of coordinates is of the form [LONGITUDE, LATITUDE].

- For Prediction of Time we have considered Timestamp and Polyline.
- The rows having missing value are not considered in our calculations.

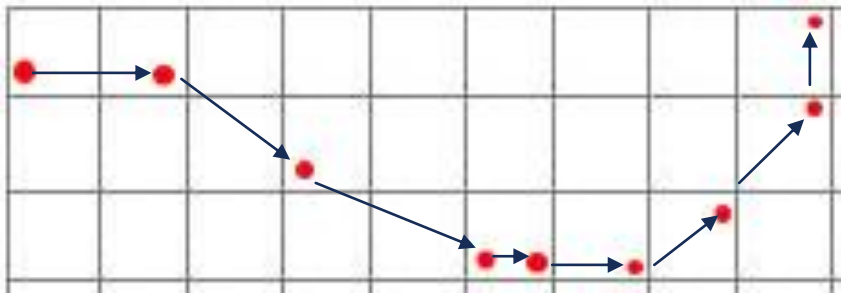
PREDICTION ARCHITECTURE

- Brief overview of the various processing steps required



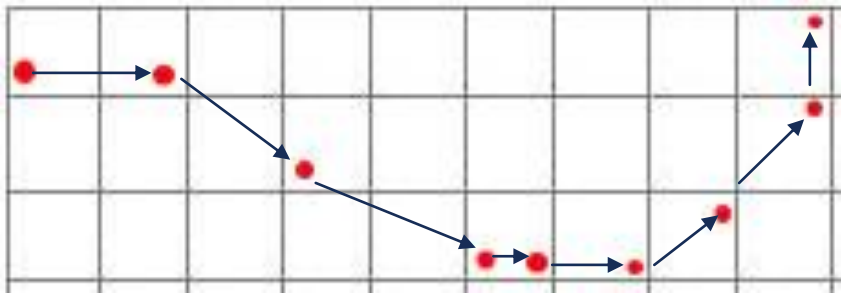
PREDICTION ARCHITECTURE

- Motion Detection
  - Since the incoming streams of traffic data come from taxi cabs, delivery vehicles, or other commercial fleet vehicles, there will always be a certain amount of ambiguity between a slowdown in traffic and a commercial stop by the vehicle. (i.e. for a taxi customer, or a delivery vehicle dropping off packages). Therefore, any further processing must clean out all unwanted stops that are in the GPS logs. The most common and intuitive technique, and that which is used in this project is to track the number of consecutive GPS points that are less than a specified distance D_{max} from each other.



PREDICTION ARCHITECTURE

- Grid Matching
  - Grid Matching is a widely studied problem in transportation research is perhaps the most computationally difficult and important subcomponent.
  - Input: $\{x_l , y_l , t_l\}$
  - Output: $\{i_l , t_l\}$



PREDICTION ARCHITECTURE

- Several distance measure:

  Manhattan Distance, Block Distance, Minkowski distance, Haversine Distance.

- For our calculation we have used Haversine distance which is defined as:
- For any two points on a sphere, the haversine of the central angle between them is given by

$$\operatorname{hav}\left(\frac{d}{r}\right) = \operatorname{hav}(\phi_2 - \phi_1) + \cos(\phi_1)\cos(\phi_2)\operatorname{hav}(\lambda_2 - \lambda_1)$$

- Where hav is the haversine function:

$$\operatorname{hav}(\theta) = \sin^2\left(\frac{\theta}{2}\right) = \frac{1 - \cos(\theta)}{2}$$

d : the distance between the two points (along a great circle of the sphere; see spherical distance),

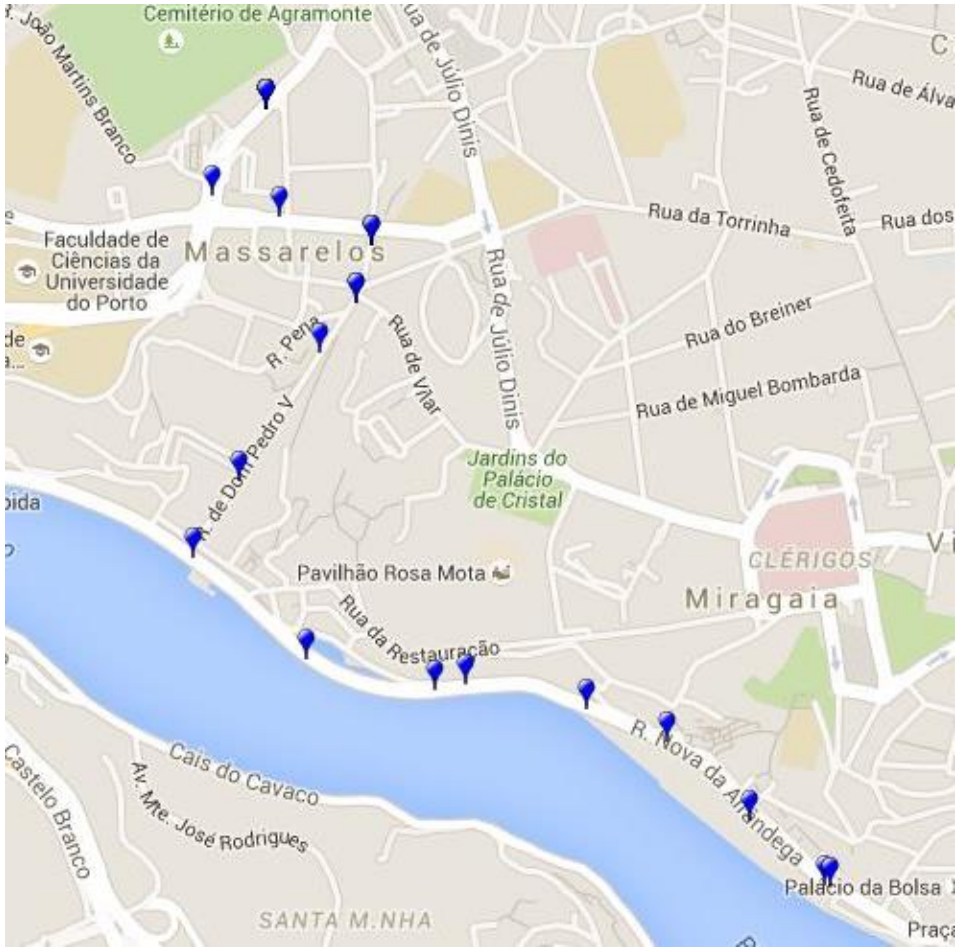
r : the radius of the sphere,

$\emptyset_1, \emptyset_2$: latitude of point 1 and latitude of point 2

λ_1, λ_2 : longitude of point 1 and longitude of point 2

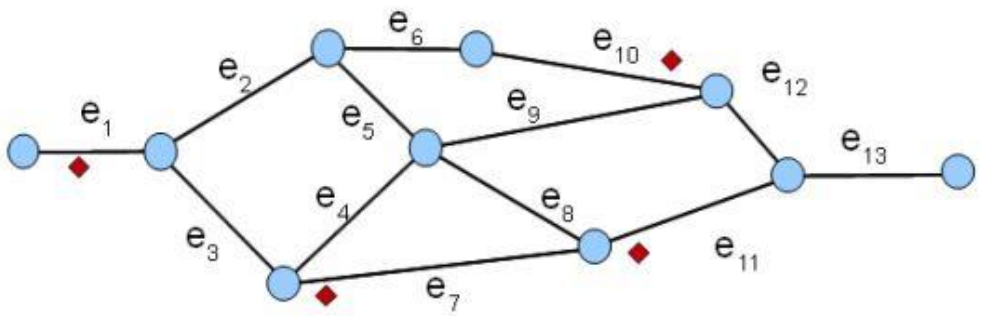
PREDICTION ARCHITECTURE

- Directed Graph
  - After Grid Matching of polyline covered by each training observation, we get a series of nodes and edges. Thus a directed graph is constructed using the grid points as nodes and velocity or time as edges.
  - Input: $\{i_1 , t_1\}$
  - Process:
    - $e_{1,2} = t_2 - t_1$
    - $e_{1,2}$ = (Haversine distance) / $(t_2 - t_1)$
  - Output: Sparse Matrix M = $\{e_{i,j}\}$ for all i , j



PREDICTION ARCHITECTURE

- Supervised Learning
  - Decision Tree Classifier, Decision Tree Regression, Random Forest Regression, Historical Mean
  - Process:
    - The whole day is categorized in x units.
    - The timestamp of each edge and the time/velocity is fed into the system.
    - The output gives us a predicted value for a corresponding timestamp value.
  - Output: Sparse Matrix M = $\{e_{i,j}\}$ for all i, j at that timestamp.



PREDICTION ARCHITECTURE

- Decision Tree Classification:
  - Classification trees work just like regression trees, only they try to predict a discrete category (the class), rather than a numerical value. The response variable Y is categorical, so we can use information theory to measure how much we learn about it from knowing the value of another discrete variable A:

    $I[Y \mid A] = \sum_a P(A = a) * I[Y \mid A = a]$

    Where, $I[Y \mid A = a] = H[Y] - H[Y \mid A = a]$
  - The definitions of entropy H[Y] and conditional entropy H[Y |A = a].

    $H[Y \mid A = a] = - \sum_k p_{ak} \log_2 p_{ak}$

PREDICTION ARCHITECTURE

- Decision Tree Regression:
  - If the target is a continuous value, then for node *m*, representing a region R_m with Nm criterion, a common criterion is to minimize the Mean Square Error:

$$m_c = \frac{1}{N_m} \sum_{i \in N_m} y_i$$

$$H(x) = \frac{1}{N_m} \sum_{i \in N_m} (y_i - m_c)^2$$

  - Complexity : $O(n_{features} \cdot n_{samples} \cdot \log(n_{samples}))$

PREDICTION ARCHITECTURE

- Random Forest Regression:
  - Here, each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set.
  - Disadvantage: In addition, when splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. Instead, the split that is picked is the best split among a random subset of the features. As a result of this randomness, the bias of the forest usually slightly increases (with respect to the bias of a single non-random tree)
  - Advantage: But, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding an overall better model.

PREDICTION ARCHITECTURE

- Gradient Tree Boosting:
  - Gradient Tree Boosting or Gradient Boosted Regression Trees (GBRT) is a generalization of boosting to arbitrary differentiable loss functions.

$$F_m(x) = F_{m-1}(x) + h(x) = y$$

  - Find h, such that residual $y - F_{m-1}(x)$ is minimized.

$$F_m(x) = F_{m-1}(x) + \gamma_m \sum_{i=1}^{n} \nabla_F \left(L\left(y_i, F_{m-1}(x_i)\right)\right)$$

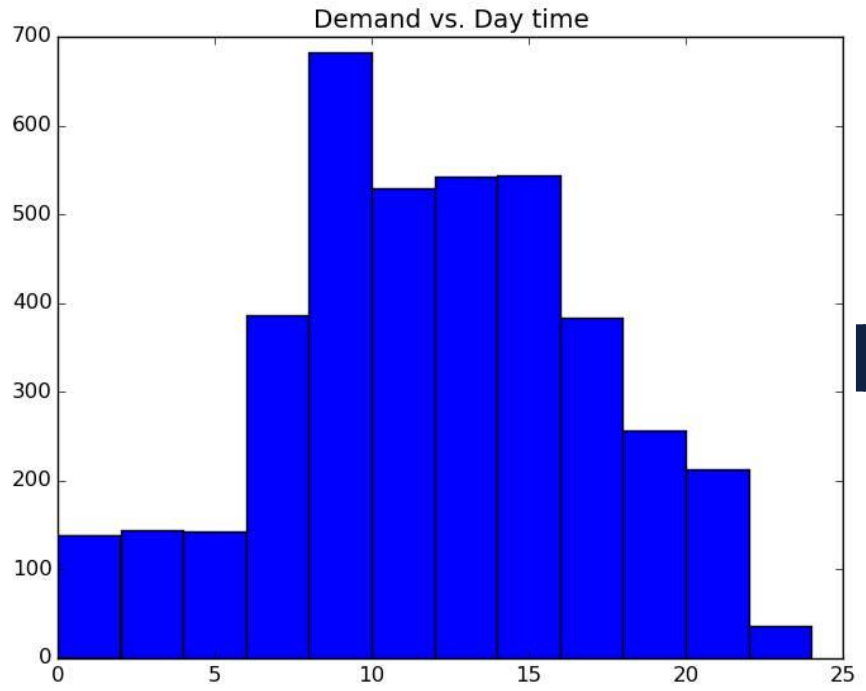
  - Loss Functions: -
    - Regression: LS, LAS, HUBER, QUANTILE
    - Classification: Binomial Deviance, Multinomial Deviance, Exponential Loss.

PREDICTION ARCHITECTURE

- Dijkstra's algorithm is an algorithm for finding the shortest paths between nodes in a graph

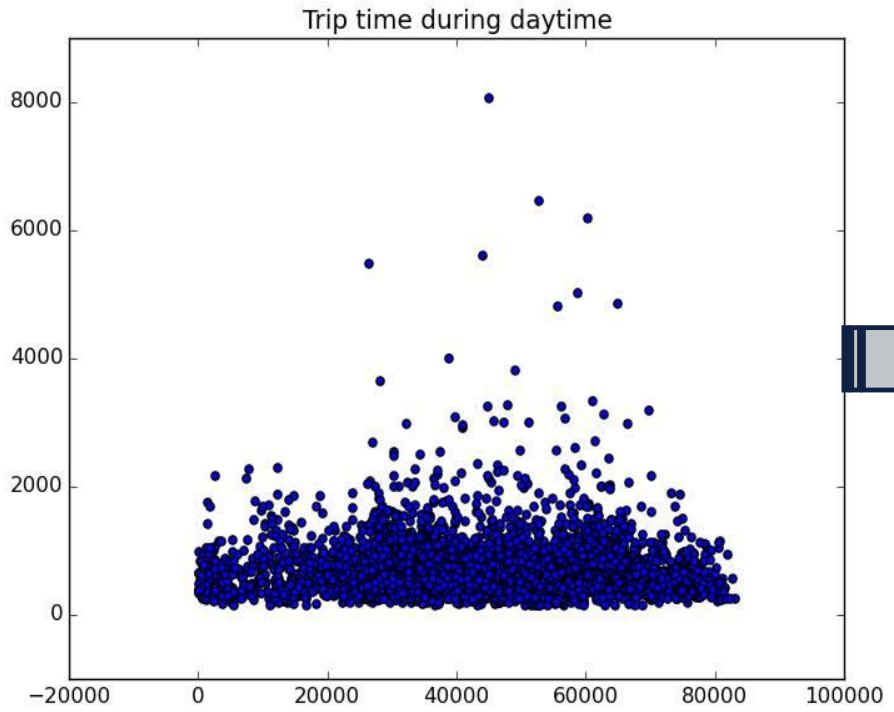
1. Assign to every node a tentative distance value.
2. Set the initial node as current. Mark all other nodes unvisited. Create a set of all the unvisited nodes called the *unvisited set*.
3. For the current node, consider all of its unvisited neighbors and calculate their *tentative* distances. Compare the newly calculated *tentative* distance to the current assigned value and assign the smaller one.
4. When we are done considering all of the neighbors of the current node, mark the current node as visited and remove it from the *unvisited set*. A visited node will never be checked again.
5. If the destination node has been marked visited (when planning a route between two specific nodes) or if the smallest tentative distance among the nodes in the *unvisited set* is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.
6. Otherwise, select the unvisited node that is marked with the smallest tentative distance, set it as the new "current node", and go back to step 3.

DATA INFERENCE



- The **Demand distribution** of taxi along the daytime is shown here.
  1. Highest peak is at the range of 9 to 10 A.M. (Office Hours)
  2. A steady distribution thereafter till 6 P.M.
  3. Can be utilized to model the value of charge/Km during the course of the day.
  4. Can be used to optimize the model of resource utilization and thereby profit generation.

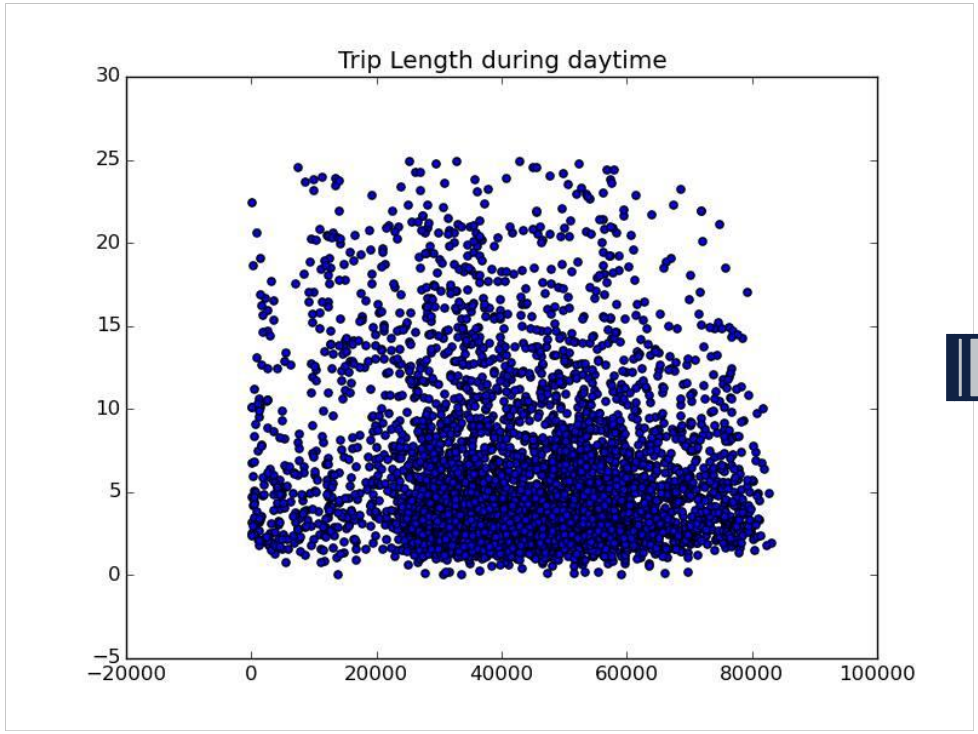
DATA INFERENCE



Individual trip time during daytime is shown here.

- Travelling time of maximum trips

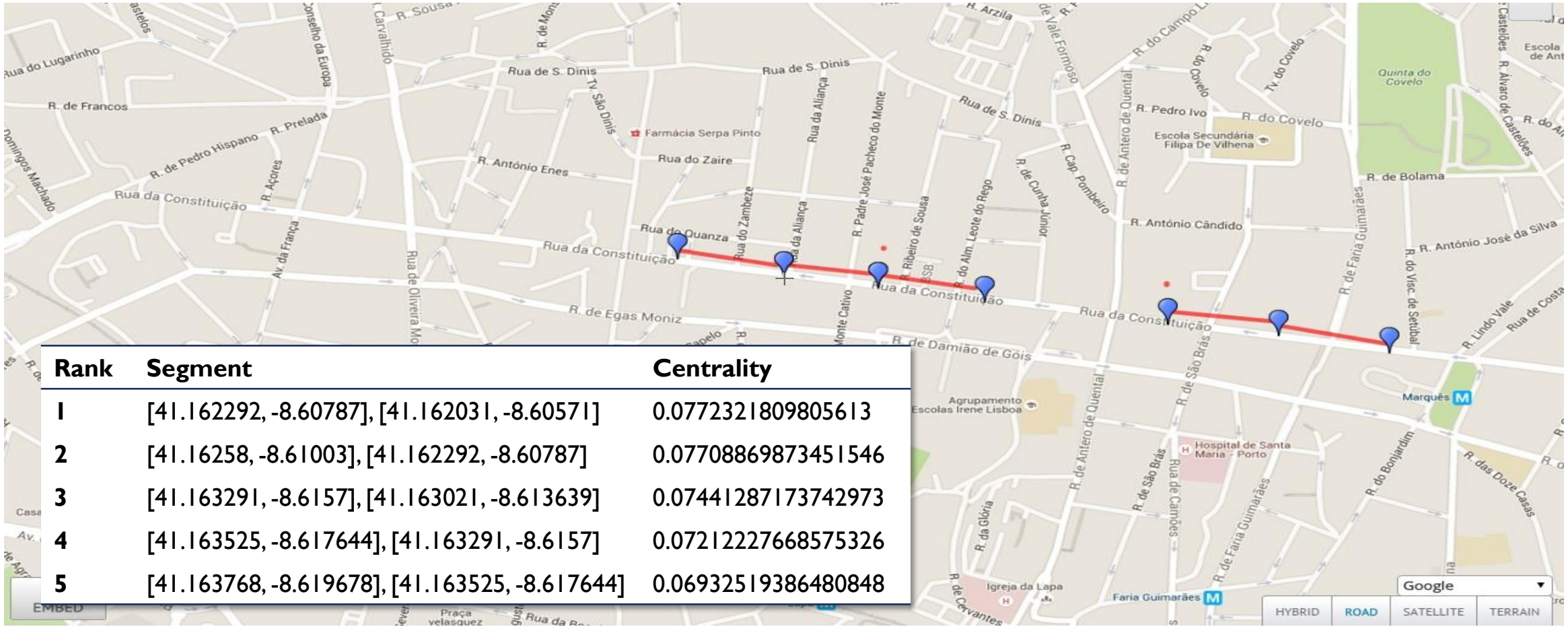
DATA INFERENCE



- The trip time shows a cluster of points in the time domain of 26000 to 82000 corresponding to 7 A.M. and 11 P.M. respectively.
- Most of the trip lengths are within 10 km.
- This is in conformity with the previous plot of travelling time where most of the travelling time are within half an hour.

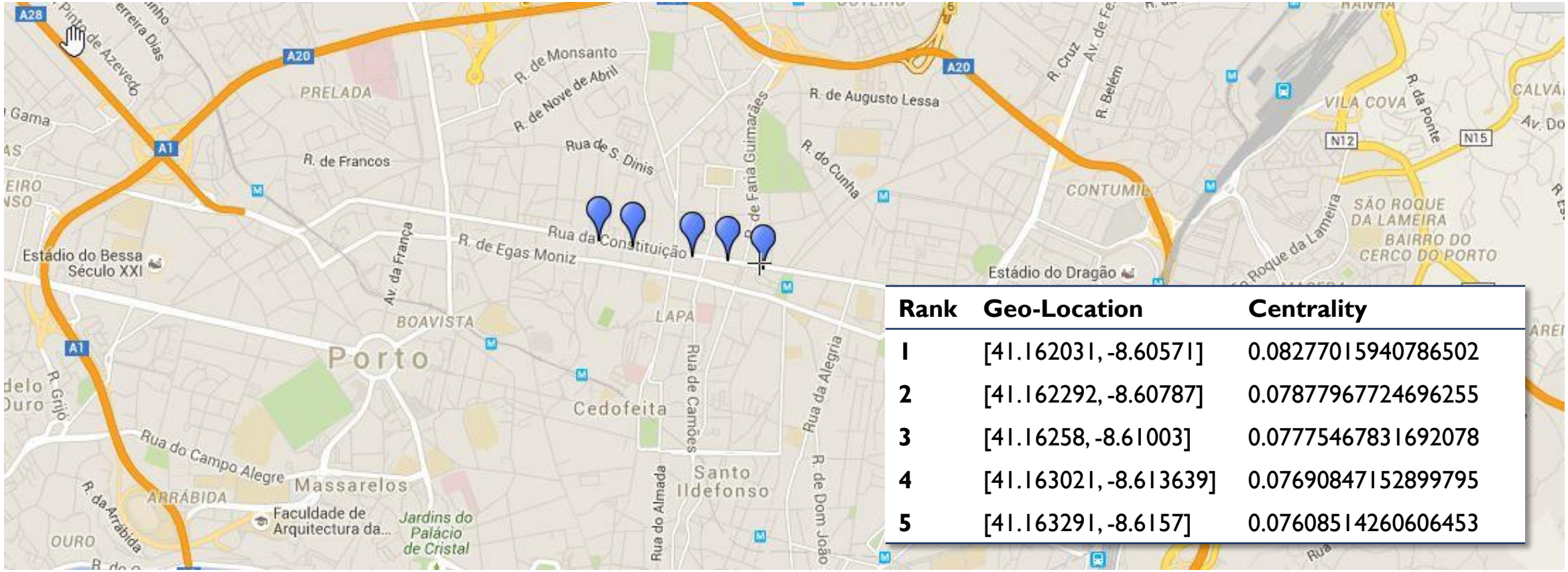
DATA INFERENCE

DATA INFERENCE



Rank	Segment	Centrality
1	[41.162292, -8.60787], [41.162031, -8.60571]	0.0772321809805613
2	[41.16258, -8.61003], [41.162292, -8.60787]	0.07708869873451546
3	[41.163291, -8.6157], [41.163021, -8.613639]	0.07441287173742973
4	[41.163525, -8.617644], [41.163291, -8.6157]	0.072122276685753 26
5	[41.163768, -8.619678], [41.163525, -8.617644]	0.06932519386480848

DATA INFERENCE



Rank	Geo-Location	Centrality
1	[41.162031, -8.60571]	0.08277015940786502
2	[41.162292, -8.60787]	0.07877967724696255
3	[41.16258, -8.61003]	0.077754678316920 78
4	[41.163021, -8.613639]	0.07690847152899795
5	[41.163291, -8.6157]	0.07608514260606453

PERFORMANCE EVALUATION

- Mean Absolute Percentage Error
  - The MAPE (Mean Absolute Percent Error) measures the size of the error in percentage terms. It is calculated as the average of the unsigned percentage error.

$$\left(\frac{1}{n}\sum\frac{|Actual - Forecast|}{|Actual|}\right)*100$$

- Mean Absolute Error:
  - The MAE measures the average magnitude of the errors in a set of forecasts, without considering their direction. It measures accuracy for continuous variables.

$$\frac{1}{n}\sum_{i=1}^{n}|f_i - y_i|$$

- Mean Percentage Error:
  - MPE is the computed average of percentage errors by which forecasts of a model differ from actual values of the quantity being forecast.

$$\frac{100\%}{n}\sum_{t=1}^{n}\frac{a_t - f_t}{a_t}$$

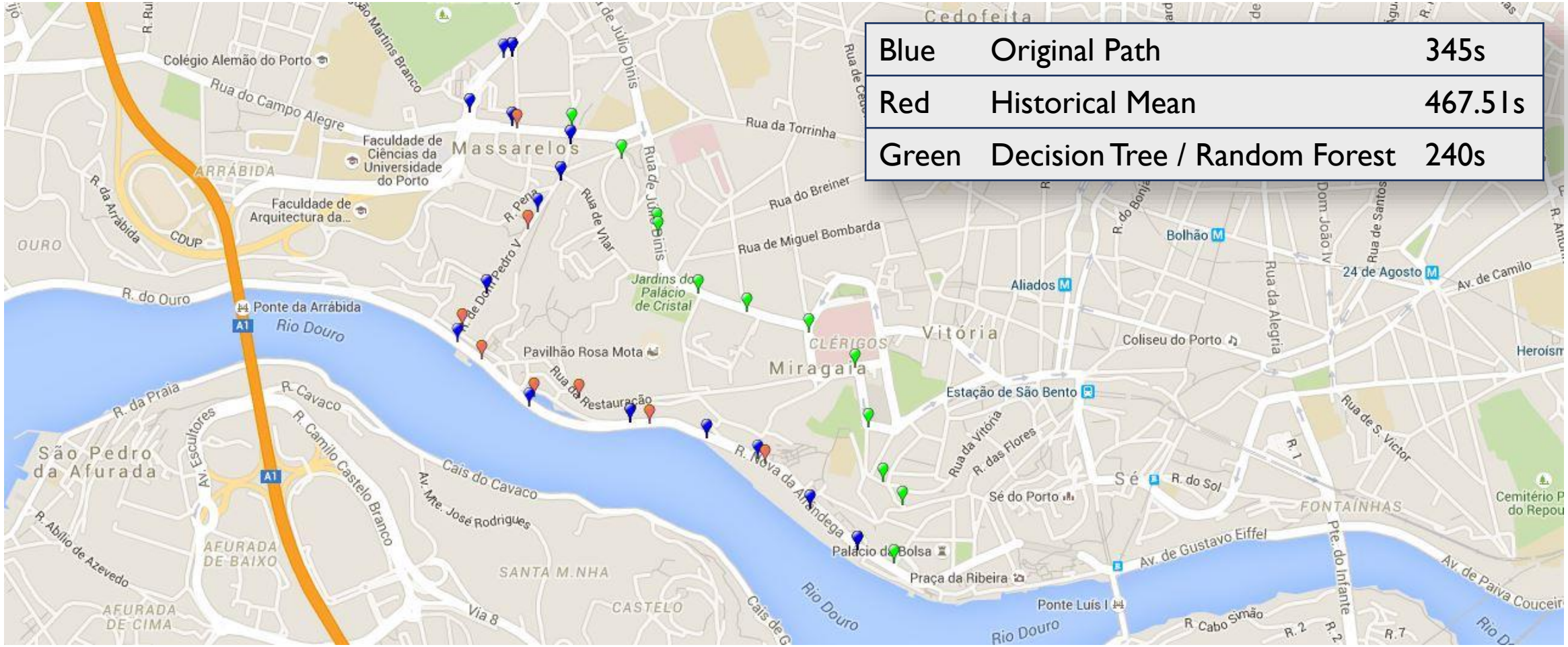
HYPOTHESIS TESTING (P-VALUE CALCULATION)

- The Percentage Error is estimated using the Null Hypothesis:
  - H_0 : PE <= 0
  - H_1 : PE > 0
- Here, the $\overline{PE}$ = mean Percentage Error ($\frac{\Delta t}{t}$) for the selected trips
- $t = \frac{\overline{PE}}{\frac{S}{\sqrt{n}}}$
- $t > t_a$ for a = critical value (one-sided) implies we reject H_0 in favor of H_1

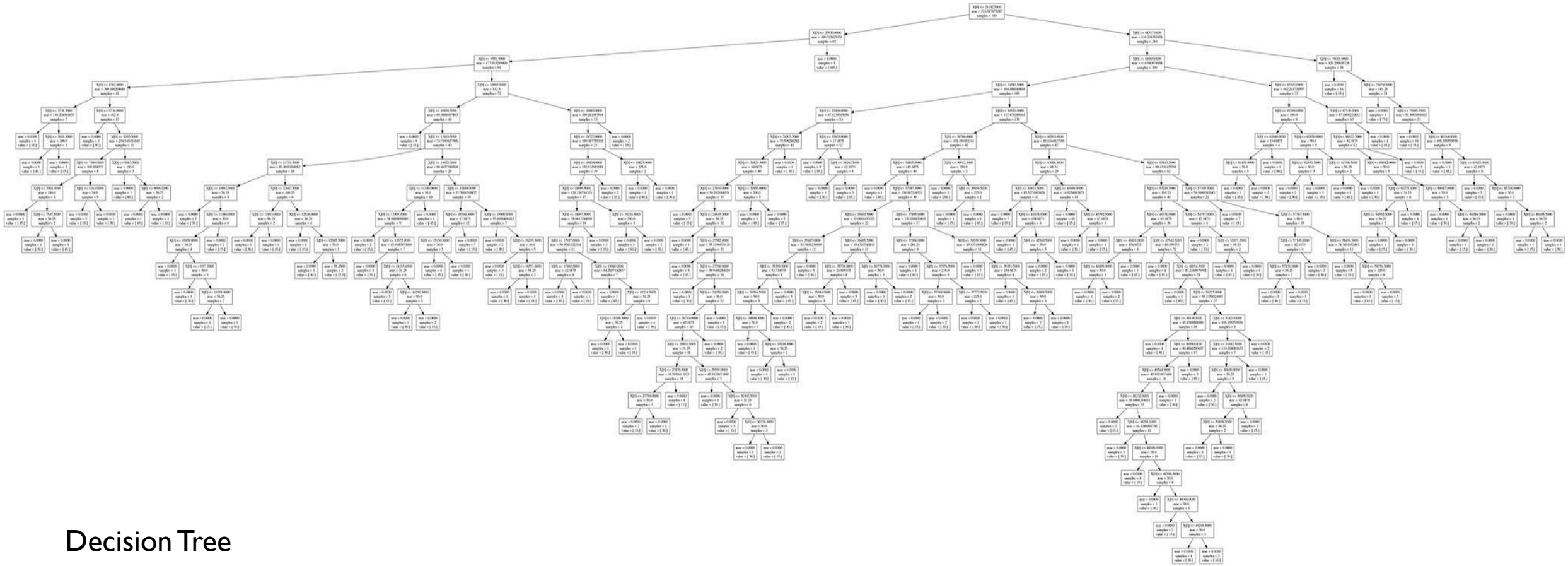
PERFORMANCE EVALUATION

	MAPE	**MAE**	**MPE**	**P-Value**
DT(R)	0.3713	252.00s	0.3556	0.999781858
DT(C)	0.3729	253.33s	0.3365	0.999182999
RF(C)	0.3692	253.33s	0.3933	0.999430984
Historical Mean	0.5899	274.24s	-0.3664	0.043007171

PERFORMANCE EVALUATION



PERFORMANCE EVALUATION



Decision Tree

FUTURE VALUE ADDITION

- Apply an incremental ARIMA model to each segment for a specific time of the day (Data needed)
- Use Map Matching Technique to position the standard points on the road accurately.
- Real Time Traffic Map (travelling route management, traffic signal optimization etc.)
- Optimization of our approach in order to reduce model error.
- To make this prediction real time and make better visualization of prediction.
- Implement it locally.
- Consolidate the Features in an Application

REFERENCES

- http://arxiv.org/pdf/1012.4249.pdf
- http://arxiv.org/pdf/1509.05257.pdf
- https://en.wikipedia.org/wiki/Haversine_formula
- http://toc.proceedings.com/04373webtoc.pdf
- http://www.ivt.ethz.ch/oev/ped2012/vpl/publications/reports/ab379.pdf
- https://www.kaggle.com/c/pkdd-15-taxi-trip-time-prediction-ii
- http://www.galitshmueli.com/data-mining-project/predicting-customer-cancellation-cab-bookings-yourcabscom
- http://scikit-learn.org/stable/modules/ensemble.html
- https://www.scribblemaps.com/create/#id=srin&lat=41.1619705&lng=-8.605703899999 98&z=14&t=road

THANK YOU