

Traffic Management

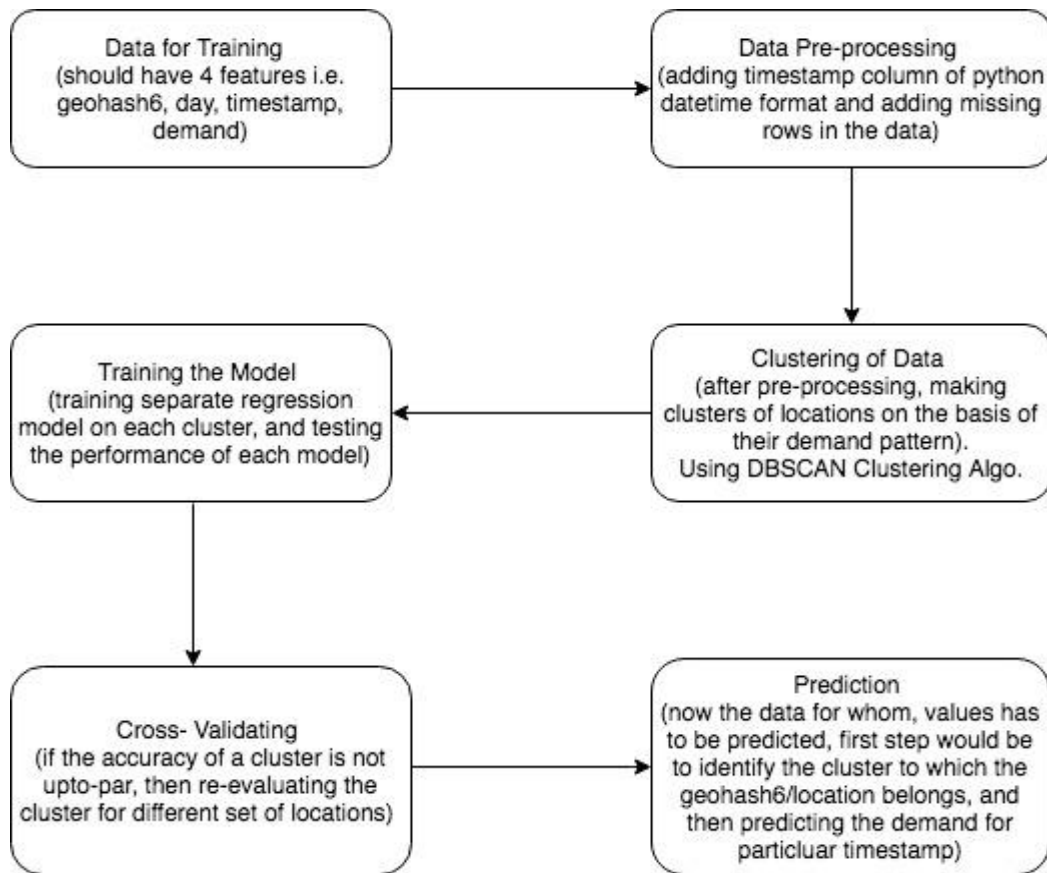
- I have selected the problem “Traffic Management”, and here in this document, i am providing my complete approach for this problem statement.
- The code file “main.py” contains all the methods for the solution logic, the file is present in the folder (the code is in Python Language).
- I have added a flow chart of the complete solution logic first and then have added the detailed explanation for each step in the flow-chart.
- At the end, i have also explained about the prediction for $T+1$ to $T+5$ timestamps (which simply means the “demand-value” that each location would have for these 5 different timestamps).

Insights from Data




- Total number of unique locations (geohash6) = 1329
- Total number of timestamps for each location for each day should be = 96 (4 timestamps in each hour)
- Total number of unique days in training dataset = 61
- Average number of sample for each location = 2956.6333 (in the training set), but for each location it should be 5856 rows (as 61 days and 96 timestamps per day), so the missing one are assumed to be the timestamps for which the location has 0 demand on a particular day (this is important for feature selection)
- Minimum demand in training dataset = 0.0
- Maximum demand in training dataset = 1.0
- Average demand from the training dataset = 0.1050
- Day 39 has maximum number of sample in training dataset = 76097 rows

Solution Flowchart



Detailed & Technical Explanation

STEP 1 (Data Preprocessing)

- 
- The given timestamp feature in the data is in string format.
 - So, adding a feature of timestamps in python datetime format.
 - Sorting the data first based on days then on the basis of new python format timestamps.
 - Also, adding missing row for geohash6/locations, as this will help us in our clustering
 - The missing rows are those timestamps of the day for which the data is not present in the dataset, so the demand on these timestamps would be 0 for these.
 - This, by doing this we will have same & equal number of data for each geohash6/location.

STEP 2 (Clustering the Data)

- In this, clustering the location data based on their demand patterns
- This is done, rather than brute force method in which we would be creating a separate model for each location/geohash6 and then would make predictions using that.
- As from preprocessing of the data, i found that there is correlation between different locations demand pattern (which validates our clustering assumption)
- Clustering on the basis of correlation among the demand feature of location, hence by keeping a threshold for correlation score we can make cluster of locations having similar demand trend.
- Right now, clustering is done on the basis of correlation only.

STEP 3 (Training the Model)

- Training the Elastic Net regression model on individual clusters (as this algorithm takes care of overfitting also by regularization).
- Optimal parameters for Elastic Net can be found by using ElasticNetCV(), by making pairs of different values and then choosing the optimal one.
- Trained model is then stored in pickle file, so that it can be loaded for prediction and cross-validation in future

STEP 4 (Cross-Validating)

- From our data of Training set of 61 days, model is trained on 55 days and rest 6 days are holded out for measuring the accuracy of our model and clusters.
- Using same RMSE for validating the performance.
- If the accuracy of a cluster is upto the desired level, the model is selected for the cluster.
- If the accuracy of a cluster is not upto par, then we have two options either change the size of the cluster or making separate model of each location in the cluster.
- So for this, i have selected a threshold (if number of locations in the clusters < 10) the we can separate model for each location.
- Otherwise, we will reallocate the cluster based on different threshold of correlation scores.

STEP 5 (Prediction in Future)

- For this, first step would be to read a single sample of geohash6/location and load its either clustered model or its individual model (depending on what is there for that location)
- Simply, predicting the demand for that location.

Hence this is my solution for the “Traffic Management” problem, the code file is “main.py” (present in the same repo), which contains some of the code for this problem,.

THANKS.