

Taxi Destination Prediction Model

Team Members-

SHIVANI REDDY SURUSANI

SREEPRIYANKA G R

I.INTRODUCTION.

Taxi ridership has made transportation more convenient and comfortable for riders and is evolving rapidly. New competitors and technologies are changing the way traditional taxi services do business. While this evolution has created new efficiencies, it has also created new problems. One major shift is the widespread adoption of electronic dispatch systems. These electronic dispatch systems make it easy to see where a taxi has been, but not necessarily where it is going. This makes it extremely difficult for dispatchers to decide about the taxi to contact.

II.MOTIVATION

During periods of high demand, there is often a taxi whose current ride will end near or exactly at a requested pick up location from a new rider. If a dispatcher knew approximately where their taxi drivers would be ending their current rides, they would be able to identify which taxi to assign to each pickup request. But, in most cases, taxi drivers operating with an electronic dispatch system do not indicate the final destination of their current ride. To improve the efficiency of electronic taxi dispatching systems it is important to be able to predict the final destination of a taxi while it is in service.

III.PROJECT APPROACH

The taxi destination prediction model predicts the destinations of taxi rides based on their initial trajectories and meta-information of each ride. This prediction model could help in dispatching taxis accurately and efficiently. The data consists of all the complete trajectories of 442 taxis running in the city of Porto (Portugal) from the year 2013-07-01 to 2014-06-30. The training dataset maintains 1.7 million data points, each representing a complete taxi ride constituting the following data attribute:

- the complete taxi ride
- a sequence of GPS positions (latitude and longitude) measured every 15 seconds.

The metadata of the dataset possesses the client ID, if the client booked the taxi by phone. If the taxi was booked at the taxi stand, a taxi stand ID is used. In other cases wherein we lack the client identification, the taxi ID and the time at which the taxi ride commences is used.

Having had a basic overview about the nature of the environment of the data set, the implementation of the model is as follows:

1.FEATURES OF THE DATA

The data set, derived from kaggle(<https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/data>), consisted of the following:

Meta-data

- 1.Unique Caller ID
- 2.Unique Stand ID
- 3.Taxi Number
- 4.Week of the year
- 5.Day of the week
- 6.1/4th of the day
- 7.Latitude and Longitude

The above given metadata sample holds the following properties:

A) TRIP_ID: is a string which is an unique identifier for each trip;

B) CALL_TYPE: data value which is in the form of a char contains either of three below mentioned values to demand the taxi booking service.

1. A -dispatched from the central;
2. B-if the taxi is obtained at the taxi stand by directly approaching the driver;

3. 'C' -if picked up otherwise.

C)ORIGIN_CALL: is an integer which is a unique identifier of the phone number which was used to demand the service.If CALL_TYPE='A' , it identifies the customer CALL_TYPE. It assumes NULL otherwise

D)ORIGIN_STAND: an integer which is an unique identifier for the taxi stand.If CALL_TYPE='B',it identifies the starting point of the taxi stand ,else it assumes a NULL value otherwise.

E)TAXI_ID: an integer that is an unique identifier for the taxi driver

F)TIMESTAMP: a time stamp in seconds that identifies the trip's start.

G)DAYTYPE: as the name suggests is a char daytype of the trip's start. It holds three values:

1. B - trips on a holiday or extending holidays, special holidays etc.
2. C - trip on the day before type B day;
3. A- trip on a normal day, weekend or workday.

H)MISSING_DATA: a boolean set to FALSE when the GPS data is a complete stream and TRUE otherwise if locations are missing

I)POLYLINE:is a list of GPS string coordinates.Brackets [and] are used for representing the beginning and the end of the string .We could also identify the data as [LONGITUDE, LATITUDE]. This contains a pair of coordinates for

every 15 seconds of trip. The last item in the list corresponds to the trip's destination while the first one in the list contains the start.

	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard
1	TRIP_ID	CALL_TYPE	ORIGIN_CALL	ORIGIN_STAND	TAXI_ID	TIMESTAMP	DAY_TYPE	MISSING_DATA	POLYLINE
2	1372636858620000589	C			20000589	1372636858	A	False	[[-8.618643,41.141412],[-8.618499,41.141376],[-8.620326,41.14251],[-8.622153,41.1
3	1372637303620000596	B		7	20000596	1372637303	A	False	[[-8.639847,41.159826],[-8.640351,41.159871],[-8.642196,41.160114],[-8.644455,41.1
4	1372636951620000320	C			20000320	1372636951	A	False	[[-8.612964,41.140359],[-8.613378,41.14035],[-8.614215,41.140278],[-8.614773,41.1
5	1372636854620000520	C			20000520	1372636854	A	False	[[-8.574678,41.151951],[-8.574705,41.151942],[-8.574696,41.151933],[-8.574666,41.1
6	1372637091620000337	C			20000337	1372637091	A	False	[[-8.645994,41.18049],[-8.645949,41.180517],[-8.646048,41.180049],[-8.646804,41.1
7	1372636965620000231	C			20000231	1372636965	A	False	[[-8.615502,41.140674],[-8.614854,41.140926],[-8.613351,41.14152],[-8.609976,41.1
8	1372637210620000456	C			20000456	1372637210	A	False	[[-8.57952,41.145948],[-8.580942,41.145039],[-8.582706,41.145021],[-8.584092,41.1
9	1372637299620000011	C			20000011	1372637299	A	False	[[-8.617563,41.146182],[-8.617527,41.145849],[-8.616978,41.144832],[-8.615754,41.1
10	1372637274620000403	C			20000403	1372637274	A	False	[[-8.611794,41.140557],[-8.611785,41.140575],[-8.612081,41.140566],[-8.612622,41.1
11	1372637905620000320	C			20000320	1372637905	A	False	[[-8.615907,41.140557],[-8.614449,41.141088],[-8.613522,41.14143],[-8.609904,41.1
12	1372636875620000233	C			20000233	1372636875	A	False	[[-8.619894,41.148009],[-8.620164,41.14773],[-8.62065,41.148513],[-8.62092,41.150
13	1372637984620000520	C			20000520	1372637984	A	False	[[-8.56242,41.168403],[-8.562429,41.168358],[-8.562348,41.167953],[-8.564571,41.1
14	1372637343620000571	A	31508		20000571	1372637343	A	False	[[-8.618868,41.155101],[-8.6175,41.154912],[-8.615079,41.154525],[-8.613468,41.15
15	1372638595620000233	C			20000233	1372638595	A	False	[[-8.608716,41.153499],[-8.607627,41.153481],[-8.606502,41.153472],[-8.606493,41.1
16	1372638151620000231	C			20000231	1372638151	A	False	[[-8.612208,41.14053],[-8.612235,41.140521],[-8.614035,41.140323],[-8.614809,41.1
17	1372637610620000497	B		13	20000497	1372637610	A	False	[[-8.585145,41.164857],[-8.584146,41.164704],[-8.583147,41.164750],[-8.627931,41.1
18	1372638481620000403	B		28	20000403	1372638481	A	False	[[-8.584263,41.163156],[-8.584695,41.163003],[-8.585595,41.162652],[-8.585487,41.1
19	1372639135620000570	A	33100		20000570	1372639135	A	False	[[-8.666757,41.174055],[-8.666784,41.174064],[-8.666649,41.174073],[-8.666325,41.1
20	1372637482620000005	C			20000005	1372637482	A	False	[[-8.599239,41.149188],[-8.598681,41.149296],[-8.597943,41.150583],[-8.596962,41.1

We used Mean-shift algorithm on the destination points to extract the classes for the destination point.

We performed no pre processing on the data set used in this model.

II. TECHNIQUE

The technique used in the model is a Feed Forward Neural Network whose principle is based as follows:

A feedforward neural network is an artificial neural network whose connections between the node units does not form a cycle. The information in the model only moves in a single forward direction , as the name suggests, from the input nodes to the output nodes through a series of hidden layers .

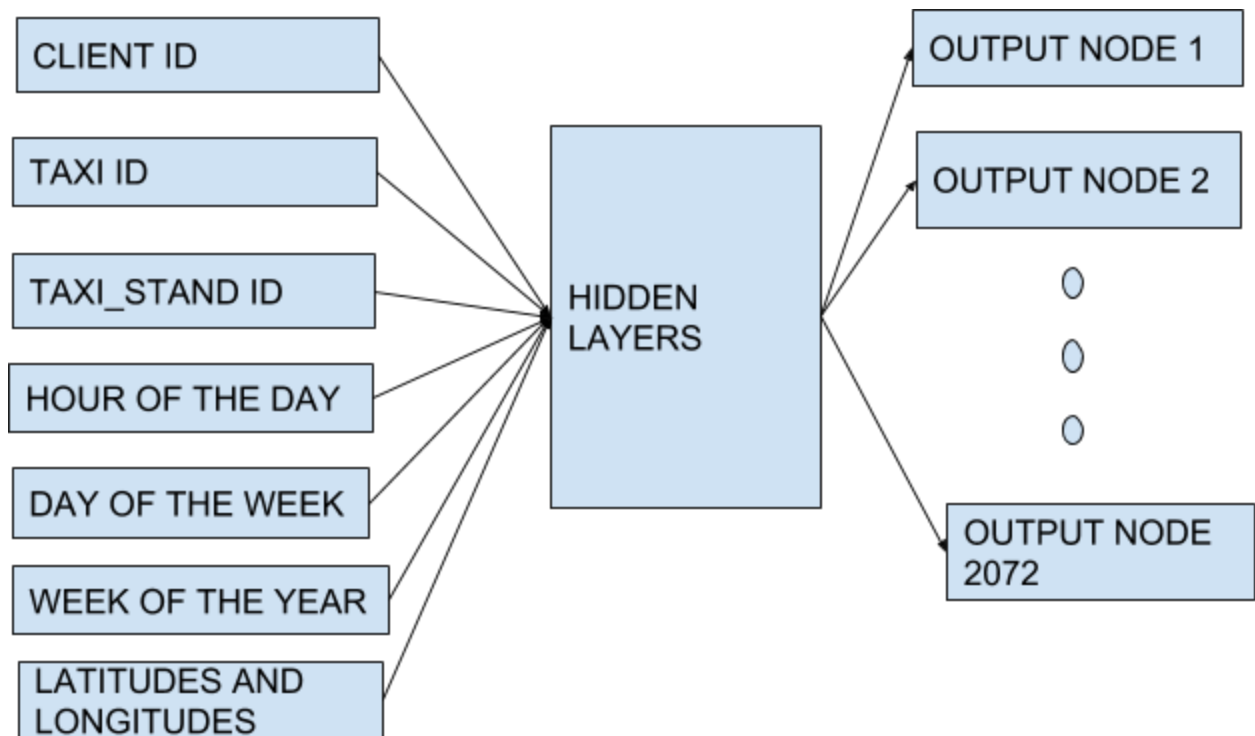
Applying this principle to our model,

- the input layer are the metadata associated with the taxi along with the first five and last five (longitude,latitude) coordinates.
- The hidden layer in this model is a single layer consisting of 500 nodes
- The output layer nodes represents the clusters from the destination data points.

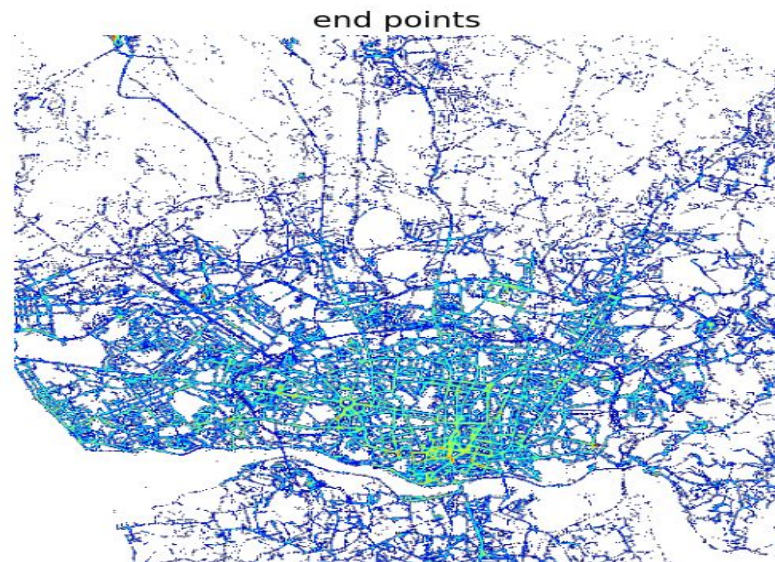
In the input layer, the data as mentioned above is fed. The hidden layer consisting of 500 nodes .The neural network activation function applied is “tanh” which transforms the inputs into the layer to its output. Tanh function is an appropriate

function as its derivative can be computed from its original function value. This allows us to compute once and re-use a value later on to get the derivative. The activation function for the output layer is a softmax, which converts the raw scores to probabilities. The output layer predicts the probabilities of the destination point being in the each cluster.

The schematic diagram of the neural network created is as follows:



The visualisation of the destination clusters is as follows:

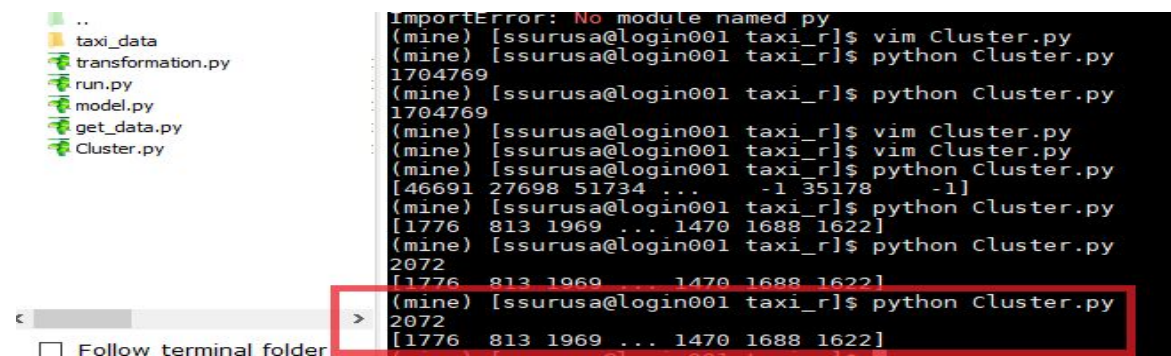


III.CODE DESCRIPTION

1.get_data.py

This file contains the logic to fetch the columns of the dataset, which is stored in the form of pickle files.

2.cluster.py



```
ImportError: No module named py
(mine) [ssurusa@login001 taxi_rl]$ vim Cluster.py
(mine) [ssurusa@login001 taxi_rl]$ python Cluster.py
1704769
(mine) [ssurusa@login001 taxi_rl]$ python Cluster.py
1704769
(mine) [ssurusa@login001 taxi_rl]$ vim Cluster.py
(mine) [ssurusa@login001 taxi_rl]$ vim Cluster.py
(mine) [ssurusa@login001 taxi_rl]$ python Cluster.py
[46691 27698 51734 ... -1 35178 -1]
(mine) [ssurusa@login001 taxi_rl]$ python Cluster.py
[1776 813 1969 ... 1470 1688 1622]
(mine) [ssurusa@login001 taxi_rl]$ python Cluster.py
2072
[1776 813 1969 ... 1470 1688 1622]
(mine) [ssurusa@login001 taxi_rl]$ python Cluster.py
2072
[1776 813 1969 ... 1470 1688 1622]
```

This file creates the number of clusters based on the destination latitudes and longitudes.

3.transformation.py

This file fetches the data from the get_data.py and transforms it such that it is usable by the neural network model.

4.model.py

```
for i in xrange(0, num_passes):
    # Forward propagation
    zz1 = X.dot(W1) + b1
    aal = np.tanh(zz1)
    zz2 = aal.dot(W2) + b2
    expo_scor = np.exp(zz2)
    probabs = expo_scor / np.sum(expo_scor, axis=1, keepdims=True)
    # Backpropagation
    delta3 = probabs
    delta3[range(n_example), y] -= 1
    dW2 = (aal.T).dot(delta3)
    db2 = np.sum(delta3, axis=0, keepdims=True)
    delta2 = delta3.dot(W2.T) * (1 - np.power(aal, 2))
    dW1 = np.dot(X.T, delta2)
    db1 = np.sum(delta2, axis=0)
    dW2 += r_lam * W2
    dW1 += r_lam * W1
    # Gradient descent parameter update
    W1 += -learn_rate * dW1
    b1 += -learn_rate * db1
    W2 += -learn_rate * dW2
    b2 += -learn_rate * db2
    # Assign new parameters to the model
    model = {'W1': W1, 'b1': b1, 'W2': W2, 'b2': b2}
print "Loss after iteration %i: %f" % (i, calculate_loss(model))
return model
```

This file builds the neural network for predicting the destination of each rides.

5.Script.py

Generates the visualization of the clusters based on the destination points.

6.mean_shift_clustering.py

The mean shift function is used in this file to generate the clusters.

IV. DEPENDENCIES

We have the following dependencies in the project:

1. Theano.

Theano is a numerical ,general GPU-accelerated math library for python.

2.Numpy

Fundamental package for scientific computing in python.

3.HDBSCAN

Hierarchical Density-Based Spatial Clustering of Applications with Noise.

Performs a database scan over varying epsilon values and integrates the result to find a clusters.

4.Pickle

It is used for serializing and de-serializing structure of python objects.

5.h5py

A Pythonic interface to the HDF5 binary data format.

V. GENERATING THE SOLUTION.

The sequence of files to be run to generate the model is as follows:

1. Download all files from the below link into data folder:
<https://drive.google.com/open?id=1N6rbJ7kjLudIPzeZ9zF6jFWvFo4B43eE>
2. Fetch the data through the *fetch_data.py*
3. Create the clusters using the *cluster.py*
4. Run the *transformation.py* to convert the data into suitable form to feed into the network.
5. Create the network by executing the *model.py*
6. To create the visualization, run *script.py*

IV. REFERENCES.

1.Artificial Neural Networks Applied to Taxi Destination Prediction

**Alexandre de Brébisson¹ , Étienne Simon² , Alex Auvolat³ , Pascal Vincent¹⁴,
and Yoshua Bengio¹⁴(arXiv:1508.00021v2 [cs.LG] 21 Sep 2015).**

- 2.<https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/>
- 3.[https://en.wikipedia.org/wiki/Theano_\(software\)](https://en.wikipedia.org/wiki/Theano_(software))
- 4.<http://www.numpy.org/>
- 5.<https://pythontips.com/2013/08/02/what-is-pickle-in-python/>
- 6.<https://pypi.org/project/hdbscan/>
- 7.https://www.google.com/search?ei=ky7ZWv95jZCOBKHqLg&q=what+is+h5py+&oq=what+is+h5py+&gs_l=psy-ab.3...11923.13308.0.13967.6.6.0.0.0.0.0..0.0....0...1c.1.64.psy-ab..6.0.0....0._wYazmxnwD8
- 8.<http://www.wildml.com/2015/09/implementing-a-neural-network-from-scratch/>

