Individual Project 2

Classification with Deep Learning

Goal

In this project, you will be asked to finish a sequence classification task using deep learning. A trajectory data set with five taxi drivers' daily driving trajectories in 6 months will be provided, and the task is to build a model to predict which driver a trajectory belongs to. A trajectory to be classified includes all GPS records of a driver in a day. During the test, we will use data from 5 drivers in 5 days, i.e. there will be 25 labels to be evaluated. You can do anything to preprocess the data before input the data to the neural network, such as extracting features, getting sub-trajectory based on the status, and so on. This project should be completed in Python 3. Keras, Pytorch, and Tensorflow are recommended, but you can make your decision to use other tools like MxNet.

Introduction & Proposal

Since the trajectory data is a time series dta consisting: longitude, latitude, timestamp and status. The first idead come to my mind is building a Recurrent neural network using Long short-term memory (LSTM) architecture.

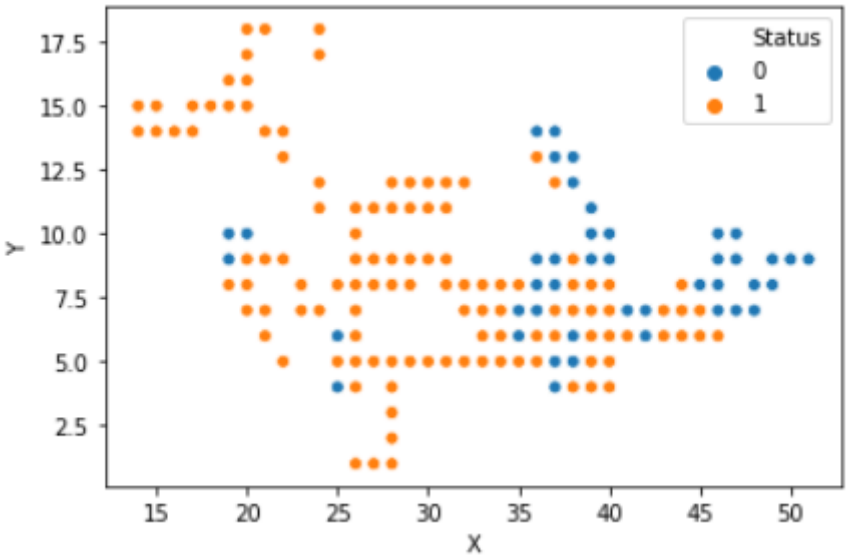
Methodology

a. Data processing

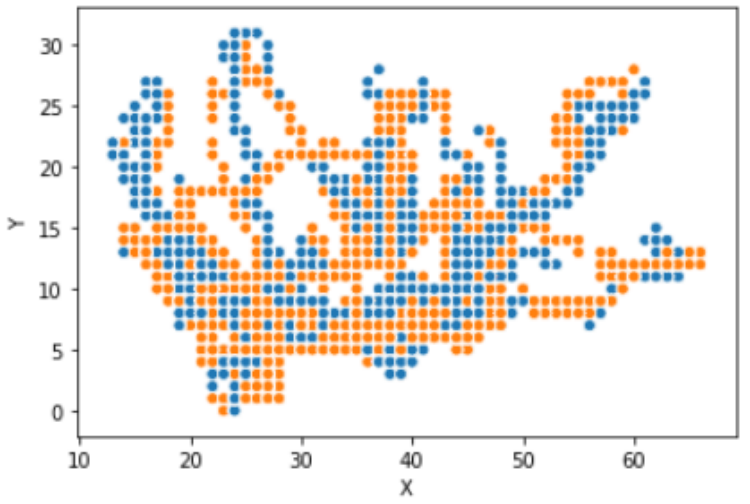
After some data exploration, besides some error data (some latitude is over 2000, which makes no sense), I tried to divide the map of Shenzhen into small grids, as professor Li's tip. I choose the top left of the map to be (113.660000, 26.870000), and the bottom right of the map to be (115.350000, 22.470000), and grid size to be [0.01, 0.01]

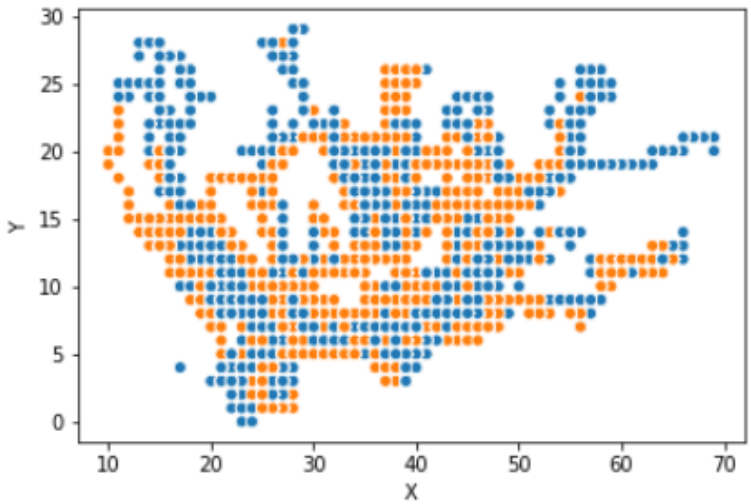
Then just for curious, I tried to plot the root for each traj:

1. the trace of traj 0 on 2016_07_01:

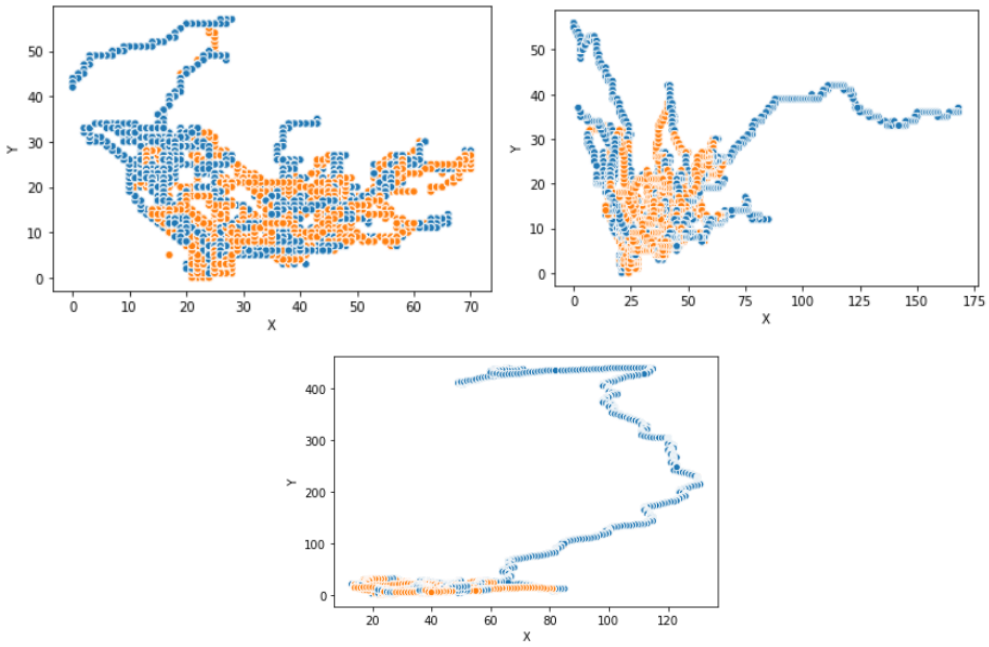


2. the trace of traj 0, 1 for whole data set:





3. then the trace of traj 2, 3, 4 for whole data set are very interesting:



After futher exploration, I found out that, driver 3, 4 probably use this taxi for personal use.

b. Feature generation

To start with, I first seperated the data by each trajectory , and processesed the timestemp to , 'X', 'Y','No.', 'status', 'Hour', Minute', 'Second', for example, top 5 of traj 0 on 2016_07_02 is:

	X	Y	No.	Status	Hour	Min	Second
	46	7	1222	0	0	8	51
	46	7	1222	0	0	9	1
	46	7	1222	1	0	9	12
	46	7	1222	1	0	9	22
	46	7	1222	1	0	9	31
	46	7	1222	1	0	9	43

Then used sequence.pad_sequences(x_train, maxlen=5000) method to pad the input of each traj data to 5000 records (fill the 0 if some day data is not enough)

c. Network structure

My Recurrent neural network is looks that this:

```
def baseline_model():
  model = Sequential()
  model.add(LSTM(100, input_shape = (5000,3), activation='relu',
return_sequences=True))
  model.add(Dropout(0.2))
  model.add(LSTM(64, activation='relu', return_sequences=True))
  model.add(Dropout(0.2))
  model.add(LSTM(32, activation='relu', kernel_regularizer = l2(0.01)))
  model.add(Dense(5, activation='softmax'))
    # Compile model
  adam =  keras.optimizers.Adam(lr=0.00001)
  model.compile(loss='categorical_crossentropy', optimizer = adam, metrics=
['accuracy'])
  return model
model = baseline_model()
```

d. Training & validation process

```
model.fit(x_train, one_hot_labels,  epochs=10, verbose=1, validation_split=0.4)
```

Evaluation & Results

a. Training & validation results

I used the train_test_split from sklearn.model_selection, to saperate the dataset into trainning (50%) and validation (40%) and testing (10%), also stratify by the label, then used keras.utils.to_categorical(y, num_classes=5) to change the label into OneHotEncoding

b. Performance comparing to your baselines (maybe different network structure)

Baseline model:

put the original data, with only separation of the time into the network, ('Long', 'Lat', 'Status','Hour','Minute', 'Second')

The performance of both model are bad, the loss dropped quickly but the accuracy didn't go up.

I also tried to use convolutional neural network, by plotting all the traj data day by day and then let the network to learn the scatterplot, however, that seems will never work, since the figure will be vague on the third or even the second layer.

c. Hyperparameter (learning rate, dropout, activation)

learning rate = 00001

dropout = 0.2

activation = relu (for LSTM network) and softmax (for output layer)

Conclusion

Network structure really need to be rebuild.