COMP4434 Big Data Analytics

Group 21

Topic One:

Prediction of Inflow / Outflow of Beijing's Taxi on Next Time Slot

Group 21

Tsui Kin Min	15076132d
Lam Tsun Fung	15067703d
Ng Tsz Kin	15066736d
Chan Hong Yu	15079132d
Cheung Chun Pan	14073743d
Ho Chong Yin	14074307d

CONTENTS

BACKGROUND

Ever since the advent of the World Wide Web, the volume of data that could be retrieved from the internet has been growing exponentially. This has motivated research in the multidisciplinary field of Data Analytics with the goal of developing systems that could take advantage of this gigantic scale of data for different uses. It is estimated that amount of data that is valuable to be analyzed in 2016 will be doubled by 2020 [1].

Among the many possible uses of Big Data analytics, making predictions is a very common one. This kind of systems typically involve using some past data to predict the state of future data that would occur under conditions similar to those when the past data were captured, in which a model of making the predictions, as well as the means to train that model, and to test it, and to evaluate it, would be defined. In this project, we design and implement such a system by applying different methods in Linear Regression and Neural Networks to the given dataset.

This dataset includes crowd flows and meteorological data in Beijing from 2015/11/1 to 2016/4/10. The city is partitioned into a 32 × 32 grid map based on the longitude and latitude. Time is divided into timeslots. The size of each timeslot is 30 minutes meaning there are 48 timeslots in a day.

Crowd inflows/outflows: for a grid (i,j) that lies at the i-th row and the j-th column, the inflow and outflow of the crowds are the total number of taxis that arrive and leave this grid during the timeslot, respectively.

Meteorological data: includes weather (a one-hot vector), temperature (a continuous value), and wind speed (a continuous value) of each day.

By developing models on making prediction of inflows/outflows of traffic data in Beijing, we hope to identify the relationship and provide results to further improve the current situation.

PROBLEM DEFINITION

The aim of this project is to design and test a model which applies the last inflows/outflows of a grid to predict the flows in next timeslot. Besides, meteorological data is combined to the flow data in advance to see whether it can improve the prediction accuracy or not.

(I) NATURE OF DATASETS

The dataset is collected from Beijing between 1st November, 2015 and 1st April, 2016 involving the inflow and outflow data of taxi in Beijing. Moreover, the meteorological data is collected which contains the information like the weather, windspeed and temperature. Each timeslot's data of the above flows and meteorological dataset is represented in a 2D array because the area of Beijing is divided into 32 x 32 grids based on the longitude and latitude.

In details, the weather data is classified into 17 different classes which are sunny, cloudy, overcast, rainy, sprinkle, moderate rain, heavy rain, rain storm, thunder storm, freezing rain, snowy, light snow, moderate snow, heavy snow, foggy, sandstorm and dusty.

Illustration:

	Date	Flow Data	Meteorological Data
Record Shape	(7220, 1) 1 timeslot equals to 30 minutes, 48 timeslots a day	(7220, 2, 32, 32) 2nd dimension: 0: inflow 1: outflow	Weather: (7220, 17) Temperature: (7220,) Windspeed: (7220,)

MODEL DESIGN AND CONSIDERATIONS

In this project, decision tree regression, linear regression, ridge linear regression and k neighbor regression models will be implemented and compared according to their results.

(I) REGRESSION MODELS

1. Linear regression
    - Ordinary least squares linear regression will be used.
2. Decision tree regression
    - Mean squared error decision trees regression will be used.
3. Ridge regression
    - Linear least squares Ridge regression with l2 regularization will be used.
4. K-nearest neighbors regression
    - K-nearest neighbors regression will be used with number of neighbors set to be 2.

Data Pre-processing:

- Remove data with uncompleted timeslots
- Adjust negative value

Splitting the training and testing dataset

- After the data pre-processing, the last 4 weeks are used for testing while the remaning are used for training.

Scores:

- The coefficient of determination R^2 for training and testing score and also the root mean square error will be calculated.

See [Appendix II]

(II) NEURAL NETWORK

Long Short-Term Memory

- Well suited for prediction/classification using data involving time series

Data Pre-processing:

- Remove data with uncompleted timeslots
- Data normalization (Squeeze between -1 and 1)
- Data shifting
    - Shift for different prediction range (1 day – 3 day)
    - Shifted data for acted as label (expected output)

Splitting the training and testing dataset

- After the data pre-processing, the last 300+ timeslot are used for testing while the remaining are used for training.

Scores:

- The coefficient of determination R^2 for training and testing score and also the root mean square error will be calculated.

SOLUTIONS USING REGRESSION MODELS

Different approaches for the prediction:

- Use the flow data of previous 1, 2 or 3 timeslot(s) together to predict the next flow data
- Use the flow data of 1 day before the current timeslot to predict the next flow data
- Use the flow data of 1 week before the current timeslot to predict the next flow data

(I) DATA PRE-PROCESSING

1. REMOVE UNCOMPLETE DATA

We would like to keep the data complete, so all flow and meteorological data in a day that is without 48 timeslots records will be discarded. This means up to 47 records will be abandoned in this situation.

Example:

2015110101 which is the 1st timeslot of the flow data on 1st November in 2015.

If the subsequent records are 2015110103, 2015110104 … 2015110148, it means the 2nd timeslot data (2015110102) is missing. Therefore, all the data in 1st November in 2015 will not be used for prediction. After we know the exact uncompleted date, we directly apply and select the related meteorological data.

After data pre-processing:

	Date	Flow Data	Meteorological Data
Record Shape	(6624, 1) 48 timeslots a day, 1 timeslot equals to 30 minutes	(6624, 2, 32, 32) 2nd dimension: 0: in-flow 1: out-flow	Weather: (6624, 17) Temperature: (6624,) Windspeed: (6624,)

See [Appendix I]

2. MAINTAIN POSITIVE RECORDS IN DATA

A simple process is to loop through the data and change the value of the negative records into 0 since we do not want any negative value existing in the dataset.

(II) FEATURE EXTRACTION

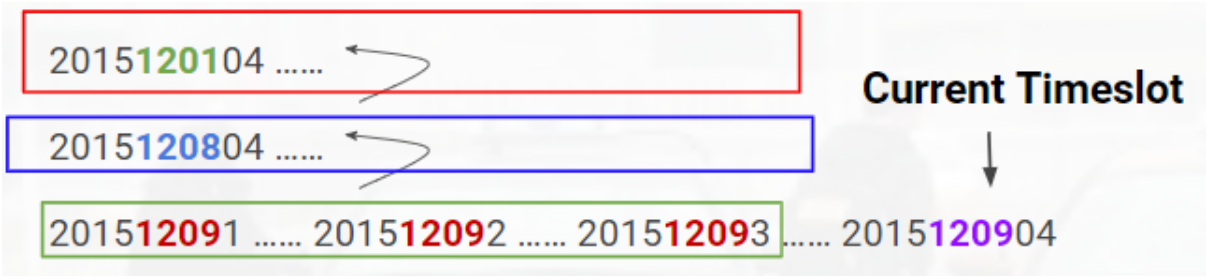


Figure 1- Timeslot showcase

In the above figure, the timeslot in purple is the current timeslot while the data in red is the previous 3 timeslots, blue is 1 day before the current timeslot and green is 1 week before the current timeslot if no data have uncompleted timeslots.

If we want to extract the data of previous 1, 2 and 3 timeslots before the current timeslot, we need to ensure that these previous timeslots exist with respect to the current timeslot (the green

box). The amount of dataset will be dropped slightly after the feature extraction since only some of the data is selected for prediction.

Besides, using the 1 day before and 1 week before timeslots before the current timeslot will further decrease the number of the dataset. For the feature of 1 day, the data of 48 timeslots (48 timeslots a day) before the current timeslot will be extracted (the blue box). Similarly, the data of 336 timeslots (7*48 = 336 timeslots a week) before the current timeslot will be extracted for the feature of 1 week (red box).

Inside the program, “closeness”, “period” and “trend” are variables related to the feature extraction.

- *closeness = 1, period = 0, trend =0* : we want to extract 1st timeslot before the current timeslot for prediction

- *closeness = 0, period = 1, trend =0*: we want to extract 1 day before the current timeslot for prediction

- *closeness = 0, period = 0, trend =1*: we want to extract 1 week before the current timeslot for prediction

For example (“*closeness = 1, period = 0, trend =0”*):

The quantity of the dataset will be decreased into 6605. It means the flow data of 30 minutes before the current timeslot will be extracted for the prediction. As shown in the following table:

	Date	Flow Data	Meteorological Data
Record Shape	(6605, 1)	(6605, 2, 32, 32)	Weather: (6605, 17)
			Temperature: (6605,)
			Windspeed: (6605,)

(III)SPLIT THE TRAINING AND TESTING DATASET

After the data pre-processing, the last 4 weeks of the dataset is used as testing dataset and remain data is used for the training dataset. Therefore, the last 1344 timeslots data (7*4*48=1344) are the testing data.

For example:

If the whole dataset has 6605 records, the dataset will be divided into 5261 training data and 1344 testing data.

(IV)RESHAPE THE DATASET

To fit in the dataset into the linear regression array, the original 4D Python numpy array should be reshaped into a 2D array. The Python function numpy.reshape() is used for the reshape work.

For example (“*closeness = 1, period = 0, trend =0”*):

If the whole dataset has 6605 records, the dataset will be divided into 5261 training data and 1344 testing data.

- Training data: reshape (5261, 2, 32, 32) into (5261, 2x32x32) = (5261, 2048)
- Testing data: reshape (1344, 2, 32, 32) into (1344, 2x32x32) = (1344, 2048)

Within the 2048 records, the first 1024 records are the in-flow data of the 32x32 grid and the following 1024 records are the out-flow data.

(V) COMBINE THE METEOROLOGICAL DATA

The meteorological data, which contains weather, wind speed and temperature data, will be combined with the flow data to check whether any improvement on the regression results can be found. Python function called numpy.hstack() is implemented to combine the meteorological data and flow data together. Before the combination, the meteorological data will be divided into training and testing dataset as well.

For example ("*closeness = 1, period = 0, trend =0"*):

The shape of the training data of meteorological after combination:

	Meteorological Data	Combined Meteorological Data
Record Shape	Weather: (5261, 17)	(5261,19)
	Temperature: (5261,)	
	Windspeed: (5261,)	

The shape of the training dataset after combination:

	Reshaped Flow Data	Combined Meteorological Data	X Training Dataset
Record Shape	(5261, 2048)	(5261,19)	(5261,2067)

SOLUTIONS USING LSTM

Approaches for the prediction:

- Use one day's in-flow and out-flow to predict that of next day, day after and two day after

(I) DATA PRE-PROCESSING

1. REMOVE UNCOMPLETE DATA

It follows exactly the same methodology purposed in the previous section. However, data from weather dataset will not be included since there is no meaningful representation of text/one-hot encoded data to fit with other numeric data.

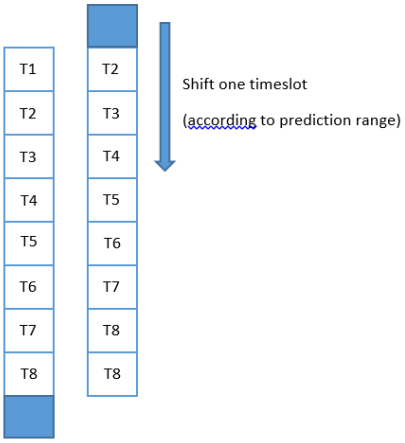
2. DATA NORMALIZATION

In order to represent the data in the same scale, Min-Max Normalization is used to fit all data under same scale between –1 and 1. The normalization process follows the below calculation logic:

$$x'' = 2\frac{x - \min x}{\max x - \min x} - 1$$

3. DATA SHIFTING

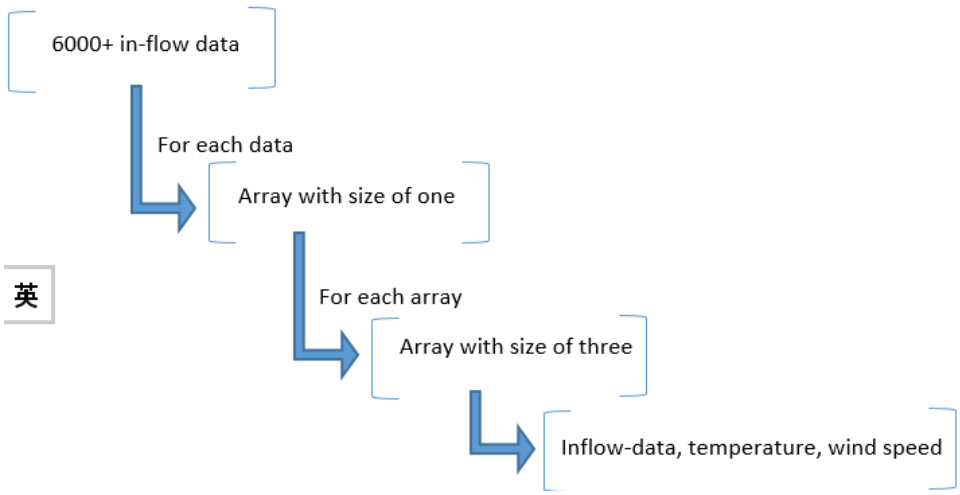
To successfully predict in-flow or out-flow data, label (expected output) should be mapped to each data (timeslot). The idea can be illustrated as below:



For each timeslot, it is expected to predict the behavior (in-flow/out-flow) of next timeslot.

4. RESHAPE THE DATASET

To integrate the dataset of temperature and windspeed which can potentially boost the prediction performance of the model, these data will also be normalized and appended after the in-flow data. The shape of data in Numpy can be expressed as below:



Shape of train data	Shape of test data
(6300, 1, 3)	(318, 1, 3)

PERFORMANCE EVALUATION (REGRESSION MODELS)

(I) REGRESSION MODELS (INFLOW&OUTFLOW PREDICTION)

The followings are the results from decision tree, linear, ridge and K neighbor regression.

1. USING ONLY 30 MINUTES BEFORE CURRENT TIMESLOT DATA

The previous 1st timeslot should be existed so the number of data is dropped to 6605.

closeness=1, period=0, trend =0 →

Shape: X_train(5261, 2048), y_train(5261, 2048), X_test (1344, 2048), y_test(1344, 2048)

```
Decision Tree Regression: Inflow and Outflow Training score: 1.0
Decision Tree Regression: Inflow and Outflow Test score: 0.8281183769205361
Decision Tree Regression: Inflow and Outflow RMSE: 27.215816777512686
Decision Tree Regression: explained_variance_score: 0.5310128015762121

Linear Regression: Inflow and Outflow Training score: 0.9635973160810887
Linear Regression: Inflow and Outflow Test score: 0.8860775239299133
Linear Regression: Inflow and Outflow RMSE: 22.15702038742074
Linear Regression: explained_variance_score: 0.6476168459806397

Ridge Linear Regression: Inflow and Outflow Training score: 0.9636028645232535
Ridge Linear Regression: Inflow and Outflow Test score: 0.8863696400092473
Ridge Linear Regression: Inflow and Outflow RMSE: 22.128595020597025
Ridge Linear Regression: explained_variance_score: 0.6485874815333054

K Neighbour Regression: Inflow and Outflow Training score: 0.9747242894505799
K Neighbour Regression: Inflow and Outflow Test score: 0.9075428941749503
K Neighbour Regression: Inflow and Outflow RMSE: 19.960747310111657
K Neighbour Regression: explained_variance_score: 0.6936658947185682
```

Same case but combined with meteorological data (**linear regression only**)→

Shape: X_train(5261, 2067), y_train(5261, 2048), X_test (1344, 2067), y_test(1344, 2048)

```
Inflow and Outflow Training score: 0.9637486445775288
Inflow and Outflow Test score: 0.8853472779224086
(1344, 2048)
Inflow and Outflow RMSE: 22.227920486892 59
==========================
```

2. USING ONLY 60 MINUTES BEFORE CURRENT TIMESLOT DATA

The previous 2nd timeslots should be existed so the number of data is dropped to 6586.

closeness=2, period=0, trend =0 →

Shape: X_train(5242, 2048), y_train(5242, 2048), X_test (1344, 2048), y_test(1344, 2048)

```
Decision Tree Regression: Inflow and Outflow Training score: 1.0
Decision Tree Regression: Inflow and Outflow Test score: 0.770830671105723
Decision Tree Regression: Inflow and Outflow RMSE: 31.432084051575206
Decision Tree Regression: explained_variance_score: 0.4912768806592899

Linear Regression: Inflow and Outflow Training score: 0.9417869864046675
Linear Regression: Inflow and Outflow Test score: 0.826157494714651
Linear Regression: Inflow and Outflow RMSE: 27.376184348526923
Linear Regression: explained_variance_score: 0.5945072900545746

Ridge Linear Regression: Inflow and Outflow Training score: 0.9417945008675132
Ridge Linear Regression: Inflow and Outflow Test score: 0.8265557806908633
Ridge Linear Regression: Inflow and Outflow RMSE: 27.344805938903296
Ridge Linear Regression: explained_variance_score: 0.5956329883120197

K Neighbour Regression: Inflow and Outflow Training score: 0.9686644957498292
K Neighbour Regression: Inflow and Outflow Test score: 0.864008443841083
K Neighbour Regression: Inflow and Outflow RMSE: 24.213128251529028
K Neighbour Regression: explained variance score: 0.658413212331826
```

Same case but combined with meteorological data (**linear regression only**)→

Shape: X_train(5242, 2067), y_train(5242, 2048), X_test (1344, 2067), y_test(1344, 2048)

```
Inflow and Outflow Training score: 0.9420447882671324
Inflow and Outflow Test score: 0.8246134115191645
(1344, 2048)
Inflow and Outflow RMSE: 27.497494314829392
==========================================
```

Comparing to the results of using 30 minutes before current timeslot for prediction, the training score, testing score and RMSE of this situation become worse.

3. USING ONLY 90 MINUTES BEFORE CURRENT TIMESLOT DATA

The previous 3rd timeslots should be existed so the number of data is dropped to 6586.

closeness=3, period=0, trend =0 →

Shape: X_train(5242, 2067), y_train(5242, 2048), X_test (1344, 2048), y_test(1344, 2048)

```
Decision Tree Regression: Inflow and Outflow Training score: 1.0
Decision Tree Regression: Inflow and Outflow Test score: 0.7236839199428139
Decision Tree Regression: Inflow and Outflow RMSE: 34.52204128804589
Decision Tree Regression: explained_variance_score: 0.44767351181889414

Linear Regression: Inflow and Outflow Training score: 0.9290310573653734
Linear Regression: Inflow and Outflow Test score: 0.7743703308833662
Linear Regression: Inflow and Outflow RMSE: 31.195466988764903
Linear Regression: explained_variance_score: 0.5534000477976266

Ridge Linear Regression: Inflow and Outflow Training score: 0.9290308188853005
Ridge Linear Regression: Inflow and Outflow Test score: 0.7749033807886653
Ridge Linear Regression: Inflow and Outflow RMSE: 31.158595566326515
Ridge Linear Regression: explained_variance_score: 0.5545771094111578

K Neighbour Regression: Inflow and Outflow Training score: 0.9658989472631747
K Neighbour Regression: Inflow and Outflow Test score: 0.8428442272908266
K Neighbour Regression: Inflow and Outflow RMSE: 26.035058460999725
K Neighbour Regression: explained variance score: 0.6443264800230422
```

Same case but combined with meteorological data (**linear regression only**)→.

Shape: X_train(5242, 2067), y_train(5242, 2048), X_test (1344, 2067), y_test(1344, 2048)

```
Inflow and Outflow Training score: 0.9295462429792793
Inflow and Outflow Test score: 0.7699610230961691
(1344, 2048)
Inflow and Outflow RMSE: 31.49880673266707
==========================================
```

Comparing to the results of using 30 or 60 minutes before current timeslot for prediction, the training score, testing score and RMSE of this situation become worse.

4. USING ONLY 1 DAY BEFORE CURRENT TIMESLOT DATA

The previous 48th timeslots should be existed so the number of data is dropped to 5712.

closeness=0, period=1, trend =0 →

Shape: X_train(4368, 2048), y_train(4368, 2048), X_test (1344, 2048), y_test(1344, 2048)

```
Decision Tree Regression: in_out - Training score: 1.0
Decision Tree Regression: in_out - Test score: 0.3842189989053642
Decision Tree Regression: in_out - RMSE: 51.68312296804715
Decision Tree Regression: in_out - explained_variance_score: 0.16378333140314377

Linear Regression: in_out - Training score: 0.8821558097548734
Linear Regression: in_out - Test score: 0.41983076542822667
Linear Regression: in_out - RMSE: 50.16640189416522
Linear Regression: in_out - explained_variance_score: 0.21719676328234666

Ridge Linear Regression: in_out - Training score: 0.8821419184264951
Ridge Linear Regression: in_out - score: 0.42148553769837993
Ridge Linear Regression: in_out - RMSE: 50.094807914372225
Ridge Linear Regression: in_out - explained_variance_score: 0.22023580514414076

K Neighbour Regression: in_out - Training score: 0.9493655635925918
K Neighbour Regression: in_out - Test score: 0.6147971481535864
K Neighbour Regression: in_out - RMSE: 40.87712587378161
K Neighbour Regression: in_out - explained_variance_score: 0.4593105019696442
```

Same case but combined with meteorological data (**linear regression only**)→.

Shape: X_train(4368, 2067), y_train(4368, 2048), X_test (1344, 2067), y_test(1344, 2048)

```
Inflow and Outflow Training score: 0.8859499789542449
Inflow and Outflow Test score: 0.340308244281002 65
(1344, 2048)
Inflow and Outflow RMSE: 53.49413062048518
==========================================
```

Comparing to the results of using 30, 60 or 90 minutes before current timeslot for prediction, the training score, testing score and RMSE of this situation become worse.

5. USING ONLY 1 WEEK BEFORE CURRENT TIMESLOT DATA

The previous 336th timeslots should be existed so the number of data is dropped to 5520.

closeness=0, period=0, trend =1 →

Shape: X_train(4176, 2048), y_train(4176, 2048), X_test (1344, 2048), y_test(1344, 2048)

```
Decision Tree Regression: Inflow and Outflow Training score: 1.0
Decision Tree Regression: Inflow and Outflow Test score: 0.3387316053251865
Decision Tree Regression: Inflow and Outflow RMSE: 54.17642108210954
Decision Tree Regression: explained_variance_score: 0.13527445383259182

Linear Regression: Inflow and Outflow Training score: 0.8745860170867664
Linear Regression: Inflow and Outflow Test score: 0.2227757132261677
Linear Regression: Inflow and Outflow RMSE: 58.73468046835249
Linear Regression: explained_variance_score: 0.03117305506514579

Ridge Linear Regression: Inflow and Outflow Training score: 0.8745660449955903
Ridge Linear Regression: Inflow and Outflow Test score: 0.22553935426078894
Ridge Linear Regression: Inflow and Outflow RMSE: 58.63016358175088
Ridge Linear Regression: explained_variance_score: 0.03547890123042837

K Neighbour Regression: Inflow and Outflow Training score: 0.9433750611102841
K Neighbour Regression: Inflow and Outflow Test score: 0.5492756245640277
K Neighbour Regression: Inflow and Outflow RMSE: 44.727739690089685
K Neighbour Regression: explained_variance_score: 0.4079282980619953
```

Same case but combined with meteorological data (**linear regression only**)→.

Shape: X_train(4176, 2067), y_train(4176, 2048), X_test (1344, 2067), y_test(1344, 2048)

```
Inflow and Outflow Training score: 0.877926929429636
Inflow and Outflow Test score: 0.1570856243193501
(1344, 2048)
Inflow and Outflow RMSE: 61.16643362885011
========================================
```

Comparing to the results of using 30 minutes, 60 minutes, 90 minutes, 1 day before current timeslot for prediction, the training score, testing score and RMSE of this situation become worse.

6. USING COMBINATION OF DATA (WITHOUT METEOROLOGICAL DATA)

The previous 336th, 48th, 3rd, 2nd, 1st timeslots should be existed so the number of data is dropped greatly into 4848.

closeness=3, period=1, trend =1 →

Shape: X_train(3504, 2048), y_train(3504, 2048), X_test (1344, 2048), y_test(1344, 20)

```
---------------1---------------

Decision Tree Regression: in_out - Training score: 1.0
Decision Tree Regression: in_out - Test score: 0.8382078034837872
Decision Tree Regression: in_out - RMSE: 26.910568691583887
Decision Tree Regression: in_out - explained_variance_score: 0.5362036298617504

Linear Regression: in_out - Training score: 0.9760379283730628
Linear Regression: in_out - Test score: 0.8239401926541429
Linear Regression: in_out - RMSE: 28.07205468672764
Linear Regression: in_out - explained_variance_score: 0.45535033542260617

Ridge Linear Regression: in_out - Training score: 0.9760053773661589
Ridge Linear Regression: in_out - score: 0.8270186187915326
Ridge Linear Regression: in_out - RMSE: 27.825550832840015
Ridge Linear Regression: in_out - explained_variance_score: 0.4627240225778315

K Neighbour Regression: in_out - Training score: 0.9750301942302313
K Neighbour Regression: in_out - Test score: 0.9082099636354579
K Neighbour Regression: in_out - RMSE: 20.26944517136544
K Neighbour Regression: in_out - explained_variance_score: 0.6925360060693508

---------------2---------------

Decision Tree Regression: in_out - Training score: 1.0
Decision Tree Regression: in_out - Test score: 0.762126293116222
Decision Tree Regression: in_out - RMSE: 32.63001777078044
Decision Tree Regression: in_out - explained_variance_score: 0.47137177237181455

Linear Regression: in_out - Training score: 0.9621968823314277
Linear Regression: in_out - Test score: 0.7356771625282321
Linear Regression: in_out - RMSE: 34.39627670178915
Linear Regression: in_out - explained_variance_score: 0.377101292187748

Ridge Linear Regression: in_out - Training score: 0.9621459752452219
Ridge Linear Regression: in_out - score: 0.739716104494417
Ridge Linear Regression: in_out - RMSE: 34.132471719731726
Ridge Linear Regression: in_out - explained_variance_score: 0.3851972777516013

K Neighbour Regression: in_out - Training score: 0.9716499290238819
K Neighbour Regression: in_out - Test score: 0.8753499249120151
K Neighbour Regression: in_out - RMSE: 23.620569387940403
K Neighbour Regression: in_out - explained_variance_score: 0.6673799795370978
```

```
---------------3---------------
Decision Tree Regression: in_out - Training score: 1.0
Decision Tree Regression: in_out - Test score: 0.7393813275958052
Decision Tree Regression: in_out - RMSE: 34.15441524238071
Decision Tree Regression: in_out - explained_variance_score: 0.4590162307306836

Linear Regression: in_out - Training score: 0.953687709886586
Linear Regression: in_out - Test score: 0.6645241660040501
Linear Regression: in_out - RMSE: 38.75026802218306
Linear Regression: in_out - explained_variance_score: 0.3187449228740261

Ridge Linear Regression: in_out - Training score: 0.9536395029065875
Ridge Linear Regression: in_out - score: 0.6687189300252844
Ridge Linear Regression: in_out - RMSE: 38.5072407486229
Ridge Linear Regression: in_out - explained_variance_score: 0.32697073248054287

K Neighbour Regression: in_out - Training score: 0.9668741372879989
K Neighbour Regression: in_out - Test score: 0.8535376697323459
K Neighbour Regression: in_out - RMSE: 25.603955276375164
K Neighbour Regression: in_out - explained_variance_score: 0.6525739353580893
```

```
---------------1+2---------------
Decision Tree Regression: in - Training score: 1.0
Decision Tree Regression: in - Test score: 0.827983194221825
Decision Tree Regression: in - RMSE: 27.737475933284802
Decision Tree Regression: in - explained_variance_score: 0.528532681887594

Linear Regression: in - Training score: 0.9769356028523094
Linear Regression: in - Test score: 0.8294908725156764
Linear Regression: in - RMSE: 27.615652872721352
Linear Regression: in - explained_variance_score: 0.4548731376306517

Ridge Linear Regression: in - Training score: 0.976935082349825
Ridge Linear Regression: in - score: 0.829819972697881
Ridge Linear Regression: in - RMSE: 27.588989473200787
Ridge Linear Regression: in - explained_variance_score: 0.4557364557138486

K Neighbour Regression: in - Training score: 0.9773854495589686
K Neighbour Regression: in - Test score: 0.9124362817798013
K Neighbour Regression: in - RMSE: 19.789899847956306
K Neighbour Regression: in - explained_variance_score: 0.6983171699788995
```

```
---------------1+2+3+day+week---------------
Decision Tree Regression: in - Training score: 1.0
Decision Tree Regression: in - Test score: 0.8148770864113901
Decision Tree Regression: in - RMSE: 28.774751876664656
Decision Tree Regression: in - explained_variance_score: 0.5146556776825065

Linear Regression: in - Training score: 1.0
Linear Regression: in - Test score: 0.6206970862045862
Linear Regression: in - RMSE: 41.188368392352466
Linear Regression: in - explained_variance_score: -0.07916193059657521

Ridge Linear Regression: in - Training score: 0.9999999997389181
Ridge Linear Regression: in - score: 0.6207906218000587
Ridge Linear Regression: in - RMSE: 41.1832895807174
Ridge Linear Regression: in - explained_variance_score: -0.07891581684330931

K Neighbour Regression: in - Training score: 0.9767696342859462
K Neighbour Regression: in - Test score: 0.8983577192814547
K Neighbour Regression: in - RMSE: 21.321546394951852
K Neighbour Regression: in - explained variance score: 0.6819416463416276
```

These are the results of the experiment on combining multiple timeslots for prediction. The first three results marked with '1', '2' and '3' are prediction using 1st, 2nd and 3rd timeslot before current timeslot respectively. It also shows that using "30 minutes before" data for prediction is better than using "60 minutes before" and "90 minutes before" data

Besides, the results of the combination of multiple timeslots for prediction indicate that using "1+2" is the best, following with "1+2+3+day+week". However, using only 30 minutes before current timeslot data has better prediction results than "1+2"

(II) REGRESSION MODELS (IN-IN AND OUT-OUT)

The following experiments are based on using only 1 timeslot of flow data before the current flow data for prediction. Moreover, the prediction is tested on using in-flow to predict in-flow and out-flow to predict out-flow.

The previous 1st timeslot should be existed so the number of data is dropped to 6605. But, the second dimension of the data is decreased to the half because only in-flow or out-flow is used.

closeness=1, period=0, trend =0 →

Shape: X_train(5261, 1024), y_train(5261, 1024), X_test (1344, 1024), y_test(1344, 1024)

1. PREVIOUS IN-FLOW DATA PREDICT NEXT IN-FLOW DATA

```
---------------1---------------
Decision Tree Regression: in - Training score: 1.0
Decision Tree Regression: in - Test score: 0.8196560353747634
Decision Tree Regression: in - RMSE: 27.86667055211243
Decision Tree Regression: in - explained_variance_score: 0.5234161315545127

Linear Regression: in - Training score: 0.951143750218836
Linear Regression: in - Test score: 0.9153154937162574
Linear Regression: in - RMSE: 19.095729620940375
Linear Regression: in - explained_variance_score: 0.7434017484515594

Ridge Linear Regression: in - Training score: 0.9511437399707656
Ridge Linear Regression: in - score: 0.9153219805806243
Ridge Linear Regression: in - RMSE: 19.094998236973495
Ridge Linear Regression: in - explained_variance_score: 0.7434206000540888

K Neighbour Regression: in - Training score: 0.9756926181290023
K Neighbour Regression: in - Test score: 0.9081451412629159
K Neighbour Regression: in - RMSE: 19.887735879540557
K Neighbour Regression: in - explained_variance_score: 0.6963914429574021
```

The training and testing scores are both high (> 0.9) as well as the RMSE is lower than using "30 minutes before" for previous in/out-flow predicting next in/out-flow

2. PREVIOUS OUT-FLOW DATA PREDICT NEXT OUT-FLOW DATA

```
---------------1---------------
Decision Tree Regression: out - Training score: 1.0
Decision Tree Regression: out - Test score: 0.7999071694905441
Decision Tree Regression: out - RMSE: 29.37613458886863
Decision Tree Regression: out - explained_variance_score: 0.5065201921858324

Linear Regression: out - Training score: 0.950798571058046
Linear Regression: out - Test score: 0.9140609258146969
Linear Regression: out - RMSE: 19.251927643754545
Linear Regression: out - explained_variance_score: 0.7388950708955376

Ridge Linear Regression: out - Training score: 0.9507985616872799
Ridge Linear Regression: out - score: 0.9140674623300455
Ridge Linear Regression: out - RMSE: 19.251195480215006
Ridge Linear Regression: out - explained_variance_score: 0.7389140010093712

K Neighbour Regression: out - Training score: 0.9740807295697094
K Neighbour Regression: out - Test score: 0.9069213187314721
K Neighbour Regression: out - RMSE: 20.035675822675127
K Neighbour Regression: out - explained_variance_score: 0.6915794565020456
```

Similarly, the training and testing scores are both high (> 0.9) as well as the RMSE is lower than using "30 minutes before" for previous in/out-flow predicting next in/out-flow

(III) EVALUATION ON LINEAR REGRESSION

Only 1 timeslot before current	In/out-flow to in/out-flow	In-In / Out-Out
Train & Test Scores	0.96 / 0.88	Both > 0.9
RMSE	~22.15	~19.09 / ~19.25

From the above results, combining the meteorological data leads to the increasing the training score slightly, but the testing score is slightly decreased and the RMSE is slightly increased. It shows that combining the meteorological data with flow data may not help the prediction of next timeslot flow data a lot.

Besides, combining multiple diverse timeslots of flow data together for prediction may not be a good choice because the it leads to overfitting sometimes which training score is too high and testing score is low. It is interesting that the combination of using “30 minutes before” and “60 minutes before” together has an acceptable testing score (~0.75) and a high training score. However, using “30 minutes before” flow data for prediction is still better than the results of combination.

In addition, the experiment of previous in-flow predicting next in-flow data and previous out-flow predicting next out-flow data have a better prediction results than using 30 minutes before current timeslot dataset for prediction. Thus, in-flow and out-flow are more related independently and a separately prediction will slightly improve the prediction results.

To conclude, using the 30 minutes before the current timeslot data without combining meteorological data to predict the flow data in the next timeslot will be the best option for previous in/out-flow predicting next in/out-flow data. In order to a slightly better result, using in-flow data predicting next in-flow data or out-flow data predicting next out-flow data will be helpful. [**Appendix III**]

PERFORMANCE EVALUATION (LSTM)

The results of using LSTM to predict in-flow data based on different prediction range with different measurement metrics are as below:

Prediction Range	Training Time	R2 Score on Train Set	R2 Score on Test Set	MSE
One Timeslot	209.73s	0.629	0.608	0.048656
Two Timeslot	212.25s	0.530	0.526	0.058840
Three Timeslot	186.00s	0.444	0.459	0.067278

MSE here is particularly small since the value of expected output and actual output are squeezed within –1 and 0. Decimal number will always get smaller after square operation which is the main reason that MSE is so small.

The actual result of prediction [**Appendix III**].

is very similar to the original data which, however, perform especially bad in extreme case (inflow-data between the actual and predict result has great difference). The reason may be due to the lack of extreme data (nature of normal distribution).

CONCLUSION OF EVALUATION

The linear regression is the standard of performance evaluation which the final values of training score, testing score and RMSE (lowest value: ~22 / ~19) are reasonable.

Ridge and K neighbor regression sometimes have a slightly better results than linear regression. The LSTM model has the results of extremely low mean square error. However, our group is

not familiar with these models. We just tried to implement the model and check the results. These models with a better performance will be used as a reference for the future development.

FUTURE DEVELOPMENT

Implement training boosting algorithm to speed up the execution and training time

- XgBoost
- AdaBoost

Apply neural network on analysis

- Try convolutional neural network or ResNet
- More neurons and neural network layers, the more sophisticated the model is, the higher the accuracy will be

Capturing both spatial and temporal dependencies

Crowd flows of a location are often affected by those of a nearby location, especially in short terms; for example, the outflow of location A at time T could have a significant effect on the inflow of location B nearby at T+1; taking the region in consideration as a whole, the in- and outflows of one location (*grid* in our case) could influence those of every other location in some (more or less significant) way. In this system, we have applied the LSTM model to capture the temporal factor, i.e. predicting flows of one timeslot based on which of the previous timeslots, but not the spatial ones described above. There have been attempts at simultaneously capturing spatial and temporal dependencies in predicting crowd flows; for example, W.Jin et al. [2] have proposed a deep-learning approach called STRCNs, which combines Convolutional Neural Network (CNN) and LSTM structures. In the future, we could consider adopting these approaches to improve the performance of our system.

REFERENCE

[1]"BIG DATA ANALYTICS PREDICTIONS AND ITS ROLE IN FUTURE", *flatworld*. [Online]. Available: https://www.flatworldsolutions.com/data-management/articles/big-data-analytics-predictions-and-future.php. [Accessed: 20- Apr- 2019]

[2] W. Jin, Y. Lin, Z. Wu et al. "Spatio-Temporal Recurrent Convolutional Networks for Citywide Short-term Crowd Flows Prediction," *ACM International Conference Proceeding Series*, Mar. 2018, pp. 28-35.

APPENDIX

(I) DATA PRE-PROCESSING

```
def remove_incomplete_days(data, date, T=48):
    # remove a certain day which has not 48 timestamps
    days = []  # available days: some day only contain some segs
    days_incomplete = []
    i = 0
    while i < len(date):
        if int(date[i][8:]) != 1:
            i += 1
        elif i+T-1 < len(date) and int(date[i+T-1][8:]) == T:
            days.append(date[i][:8])
            i += T
        else:
            days_incomplete.append(date[i][:8])
            i += 1
    print("incomplete days: ", days_incomplete)
    days = set(days)
    idx = []
    for i, t in enumerate(date):
        if t[:8] in days:
            idx.append(i)

    data = data[idx]
    date = [date[i] for i in idx]
    return data, date

data, date = remove_incomplete_days(data, date, T=48)
```

```
incomplete days:  [b'20151101', b'20151103', b'20151109', b'20151110', b'20151112', b'20151121', b'20151124', b'20151128', b'20151201', b'20151203', b'20151206', b'20151210', b'20151211', b'20151212', b'20151217', b'20151227', b'20151229', b'20160111', b'20160127', b'20160323', b'20160410']
```

```
# negative values of flow data means 0 number of flow
data[data < 0] = 0
data.shape
```

```
(6624, 2, 32, 32)
```

(II) REGRESSION MODELS

```
# Prepare Data Model
# Decision Tree Regression
from sklearn.tree import DecisionTreeRegressor
dtr_model = DecisionTreeRegressor(random_state=0)
# Linear Regression
lr_model = LinearRegression()
# Ridge Regression
from sklearn.linear_model import Ridge
ridge_model = Ridge(alpha=1.0)
# K Neighbour Regression
from sklearn.neighbors import KNeighborsRegressor
neigh_model = KNeighborsRegressor(n_neighbors=2)
```

(III) MATPLOTLIB GRAPH

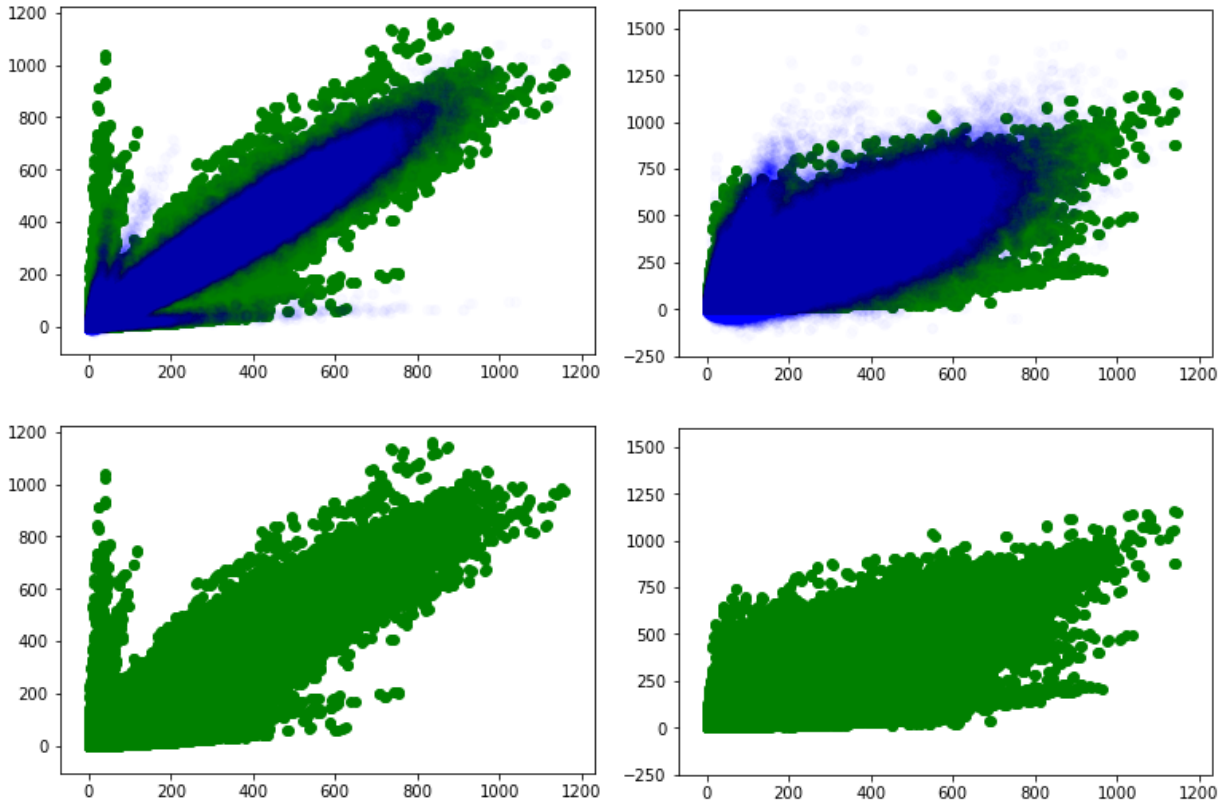


Figure 2 - "30 minutes before" only (left), "1 week before" only (right)

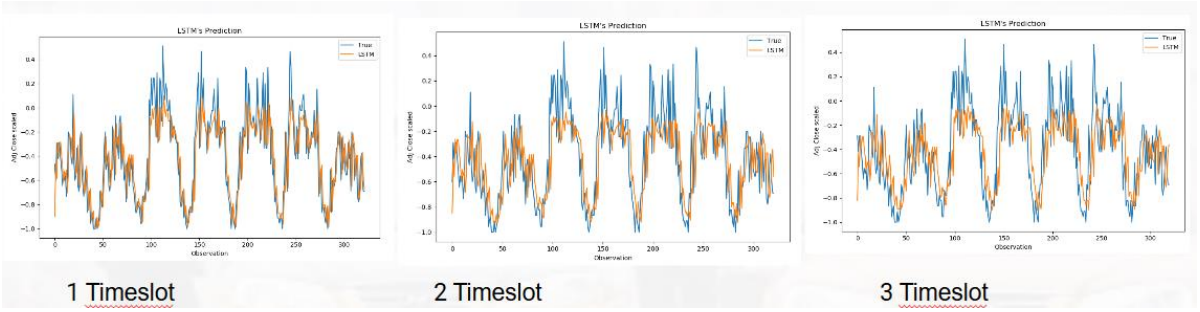


Figure 3 - Results of the LSTM Model

(IV) DOCUMENTED SOURCE FILES AND USER MANUAL

1. Download Anaconda
2. Inside Anaconda prompt, type:

   - conda create – n comp4434

   - conda activate comp443

   - conda install jupyter notebook

   - conda install -c anaconda h5py

   - conda install -c anaconda pandas

   - conda install -c anaconda scikit-learn

   - conda install -c conda-forge xgboost / pip install xgboost

   - pip install matplotlib
3. Unzip the attached zip files and place them to a certain directory like “C:/”
4. Inside the Anaconda prompt, move to related directory and type:

   - conda activate comp443

   - jupyter notebook
5. Click on the main.ipynb (Jupyter Notebook script), click and press “Ctrl+Enter” to execute the import box to check the import setting

6. No import setting error then simply adjust the variables setting and choose “Restart and Run all” in the Kernel to check the results

(IV) CONTRIBUTIONS OF TEAM MEMBERS

Student Id	Student Name	Role(s)
15076132d	Tsui Kin Min	Project leader
15067703d	Lam Tsun Fung	Program Writer
15066736d	Ng Tsz Kin	Program Writer
15079132d	Chan Hong Yu	Quality Controller
14073743d	Cheung Chun Pan	Technical Writer
14074307d	Ho Chong Yin	Technical Writer

	Task Name			
Student Name	Programming	Documentation	Presentation	Model Design
Tsui Kin Min	V	V	V	V
Lam Tsun Fung	V	V	V	V
Ng Tsz Kin	V	V	V	V
Chan Hong Yu	V	V		V
Cheung Chun Pan	V	V	V	
Ho Chong Yin	V	V		