

INDIANA UNIVERSITY BLOOMINGTON

CSCI B 565

DATA MINING

Data Analytics for IU Bus System

Group Name : RedMiners

Authors:

JAYASANKAR,
SIDDHARTH
NAGARAJAN, GANESH
MADHAVAN,
SARVOTHAMAN

Supervisor:

DR. DALKILIC,
MEHMET

December 18, 2015

Abstract

University Bus systems are widely used for shuttling students, faculty and patrons in and out of University campuses and still remains the easiest way to reach the University Campuses. Indiana University operates its bus system under the name IU Bus[1]

The project that this paper reports on attempts to develop an uniform framework and processes for analyzing IU Bus data and thus along the process hopes to establish a general framework. Any such framework should be able to transform practical nuances into presentable data analytics. The areas focused in this paper consider various aspects of discussion with the key stake holders, developing glossary, developing metrics for measuring performance, data pipe lining, data visualization, Model creation and model verification.

This project uses MySql as its preferred database for storing operational data, Neo4j graphical database for holding aggregated data, Tableau Public for visualizing the data and R for model creation and verification.

Contents

4 | 1
Introduction

5 | 2
Problem Description

5 | 3
Data Management

3.1 Source Data Model 5
3.2 Base Data Model 7
3.3 Intermediate Model 8
3.4 Data Warehouse Model 10
3.5 Data Preparation 11

13 | 4
Exploratory Analysis

16 | 5
Reporting

18 | 6
Visualization

23 | 7
Statistical Model

27 | 8
Concluding remarks

27

	9	Future Work
27	10	References

1 Introduction

University Bus systems are critical for shuttling students in and out of university campuses. These bus systems are not only used at Indiana University, but also at other university systems like University Transit Service, University of Virginia[2], The Bus, Boston University system [3]. All these universities have similar kind of schedules for different terms like Spring, Fall and special trips during late nights. All these services have an live tracking system which records the bus statuses, Geo-locations of buses and their stops and is presented real-time to the users. We also consider Cintra, Neves as one of the approaches for performing prediction. We also consider Cintra, Neves as one of the approaches for prediction[4]

IU Bus system has four routes A,B,E and X and has about 52 stops including the minor and major stops. These busses are scheduled about one bus in every 15 minutes for use of its passengers. IU Bus System commits itself to keep up the time as per the schedule[5], however as with any transportation system, the bus gains or looses speed during the course of the travel.

Thus, primary interest of this study is to understand the time difference between the schedule time and the actual time, $t_{schedule} - t_{actual}$. An model was created in Neo4j depicting the acutal bus system, and the ability of the graph database serves in the shortest paths problem and going from place A to place B.

In addition to the above said studies, an statistical model was developed in R to predict the arrival status of the bus at a given stop. The statuses can be On-Time, Delayed and Early, an detailed explanation is given in the corresponding sections. Other than this, a reporting solution was developed in Tableau and is hosted on cloud at location given below. Here, detailed reporting of status of each trips for Spring 2015 for Route A is available. This includes whether the bus arrival status and the seconds by which the bus is early / on-time.

Following are the locations of the project artifacts:

1. The database schema is part of this report and the corresponding SQL is also attached with the report submission
2. Neo4j Cloud Database : <http://iubus.sb02.stations.graphenedb>.

com:24789/browser/

3. Tableau Public : <https://public.tableau.com/profile/ganesh.nagarajan#!/vizhome/RouteA/IUBus>
4. GitHub Project URL : <https://github.com/sarvothaman/nefarious-octo-rutabaga>

All the codes, data extracts, models, templates related to the project are available at the Github location specified above.

2 Problem Description

The main objectives that were considered are

1. Find out how early/late each trip of each route has run in the past runs.
2. Use the both GPS and schedule data at hand to try improve the performance and perception of the IU transportation system

Our project uses a subset of data available to create a framework which can then be employed to perform a variety of activities. The data used

1. is from the spring semester of 2015
2. contains data from Monday through Thursday for all regular routes (does not include nightowls, weekend routes etc)

3 Data Management

3.1 Source Data Model

The source data was available in Micorosft Access DB which was imported into MySQL database facilitate manipulation. The data model into which the data was imported is below:

DBMAP_StopID
ID : int(11)
Index : double
Stop : varchar(255)

DBMAP_RouteID
ID : varchar(150)
Index : varchar(150)
Route ID : varchar(150)
Field3 : varchar(150)

DBMAP_ScheduleData
ID : int(11)
Route : varchar(255)
Time : time
Stop : varchar(255)

DBMAP_WeatherData
ID : int(11)
EDT : datetime
MinTemp : double
Precipitation : varchar(255)
Events : varchar(255)

DBMAP_WorkRecord
ID : int(11)
Clock In : time
Clock Out : time
Driver : varchar(150)
Shift Type : varchar(150)
daynum : int(11)
Date : date
Route : varchar(150)
Bus : int(11)

DBMAP_GPSData
BUS_ID : int(11)
LAT : float
LONGI : float
x : int(11)
y : int(11)
timestamp : varchar(100)

DBMAP_IntervalData
id : int(10)
from_id : int(10)
to_id : int(10)
time_id : int(10)
bus_id : int(10)
route_id : int(10)
when_time : timestamp

The tables that are prefixed 'DBMAP_' contain data from DoubleMap. This includes but not limited to schedules, work record (shifts and drivers), Time and Location for each trip for each route with bus that was run in that particular route.

IU_SPRING_Drivers
Driver : varchar(30)
Driver ID : int(11)

IU_SPRING_Buses
Bus ID : smallint(6)

IU_SPRING_Dates
Date : datetime

IU_SPRING_Schedules
Schedule Key : int(11)
Schedule ID : varchar(10)
Start Times : time

IU_SPRING_ScheduleNames
Schedule Name : varchar(255)

NOAA_Weather
STATION_NAME : varchar(150)
DATE : date
PRCP : int(11)
Measurement Flag : varchar(150)
SNOW : int(11)
TMIN : float
FOG : int(11)
RAIN : int(11)
HEAVY_FOG : int(11)
THUNDER : int(11)

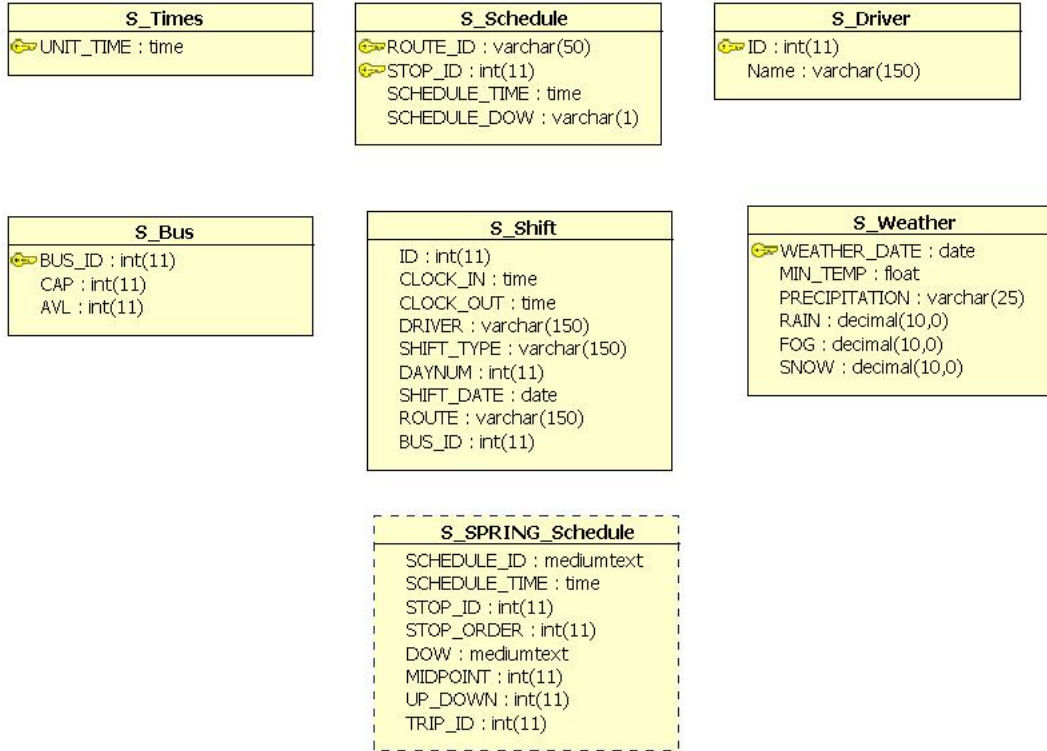
IU_SPRING_Shifts
Shift ID : int(11)
Date : datetime
Schedule Name : varchar(10)
Driver : varchar(30)
Bus ID : smallint(6)

IU_SPRING_Trips
Trip ID : int(11)
Shift ID : int(11)
Schedule ID : varchar(150)
Starting Time : time
Inbound : int(11)
Outbound : int(11)
To Mall : int(11)
From Mall : int(11)
Comments : varchar(500)

The tables that are prefixed ‘IU_’ contain data maintained by IU Bus System. This included schedules, trips and their overview such as inbound and outbound passenger count. The table prefixed ‘NOAA_’ was obtained from NOAA[6] for the Spring Term 2015

3.2 Base Data Model

The base tables contain the schedule data from both IU and DBMAP but in the format required to calculate the variation of actual trip time from schedule time.



3.3 Intermediate Model

Since one of the primary objective was to find how late/early do trips run the tables below are created to be populated with trip/schedule times (The method is explained in next section ‘Data Preparation ’)

INTER_FIRST_STOP
SCHEDULE_ID : mediumtext
SCHEDULE_TIME : time
STOP_ID : int(11)
STOP_ORDER : double
DOW : mediumtext
MIDPOINT : int(11)
UP_DOWN : bigint(20)
TRIP_ID : int(11)

INTER_LAST_STOP
SCHEDULE_ID : mediumtext
SCHEDULE_TIME : time
STOP_ID : int(11)
STOP_ORDER : double
DOW : mediumtext
MIDPOINT : int(11)
UP_DOWN : bigint(20)
TRIP_ID : int(11)

INTER_TRIPS_START_END
SCHEDULE_ID : mediumtext
TRIP_ID : int(11)
Date : date
Inbound : int(11)
outbound : int(11)
trip_start_time : time
trip_end_time : time

The above tables are intermediate tables that are used in calculating **scheduled** start and end time of each trip

INTER_SPRING_TRIPS
schedule name : varchar(10) dt : date Starting Time : time Inbound : int(11) outbound : int(11)

INTER_RUN_STOP_TRIPS
Route : varchar(150) Date : date TripID : longtext from_id : int(11) to_id : int(11) sec_time : int(11) when_time : timestamp stop_order : longtext bus_id : int(11) id : mediumtext integration_key : mediumtext travel_flag : bigint(20)

INTER_RUN_START_TRIPS
Route : varchar(150) Date : date TripID : longtext from_id : int(11) to_id : int(11) sec_time : int(11) when_time : timestamp stop_order : longtext bus_id : int(11) id : mediumtext integration_key : mediumtext travel_flag : bigint(20)

INTER_RUNS_START_END
start_time : time Date : date Route : varchar(6) TripID : double end_time : time

3.4 Data Warehouse Model

The data model by far developed is an OLTP model, which is used for operational efficiency. In order to facilitate a better analytic needs, these abstract entities are to be converted to a dimension and fact model. However, instead of traditional OLAP model, we have considered that the dimension to be the schedule and run to be facts. They reside in the model as shown below.

W_RUNS_F	W_SHED_D	F_TRIPS_COMPARED
id : mediumtext	integration_key : varchar(35)	SCHEDULE_ID : mediumtext
integration_key : varchar(35)	schedule_id : mediumtext	TRIP_ID : int(11)
Route : varchar(150)	trip_id : int(11)	Date : date
time_date : date	DOW : mediumtext	Inbound : int(11)
trip_id : longtext	STOP1_NAME : varchar(255)	outbound : int(11)
STOP1_NAME : varchar(255)	STOP1_TIME : time	schedule_start : time
STOP2_NAME : varchar(255)	STOP1_ID : int(11)	actual_start : time
STOP2_TIME : time	STOP2_NAME : varchar(255)	schedule_end : time
TRAVEL2_TIME : int(11)	STOP2_TIME : time	actual_end : time
STOP3_NAME : varchar(255)	STOP2_ID : int(11)	
STOP3_TIME : time	STOP3_NAME : varchar(255)	
TRAVEL3_TIME : int(11)	STOP3_TIME : time	
STOP4_NAME : varchar(255)	STOP3_ID : int(11)	
STOP4_TIME : time	STOP4_NAME : varchar(255)	
TRAVEL4_TIME : int(11)	STOP4_TIME : time	
STOP5_NAME : varchar(255)	STOP4_ID : int(11)	
STOP5_TIME : time	STOP5_NAME : varchar(255)	
TRAVEL5_TIME : int(11)	STOP5_TIME : time	
STOP6_NAME : varchar(255)	STOP5_ID : int(11)	
STOP6_TIME : time	STOP6_NAME : varchar(255)	
TRAVEL6_TIME : int(11)	STOP6_TIME : time	
STOP7_NAME : varchar(255)	STOP6_ID : int(11)	
STOP7_TIME : time	STOP7_NAME : varchar(255)	
TRAVEL7_TIME : int(11)	STOP7_TIME : time	
STOP8_NAME : varchar(255)	STOP7_ID : int(11)	
STOP8_TIME : time	STOP8_NAME : varchar(255)	
TRAVEL8_TIME : int(11)	STOP8_TIME : time	
STOP9_NAME : varchar(255)	STOP8_ID : int(11)	
STOP9_TIME : time	STOP9_NAME : varchar(255)	
TRAVEL9_TIME : int(11)	STOP9_TIME : time	
STOP10_NAME : varchar(255)	STOP9_ID : int(11)	
STOP10_TIME : time	STOP10_NAME : varchar(255)	
TRAVEL10_TIME : int(11)	STOP10_TIME : time	
STOP11_NAME : varchar(255)	STOP10_ID : int(11)	

It can be seen that instead of a one-row per stop approach, this data model now has all trip data in one row. The data model is designed in such a way that it can hold upto 30 stops and a select statement can be used to query the data from these tables and calculate variance.

3.5 Data Preparation

This section details the process that was undertaken to convert the data obtained from IU and DBMAP into actionable format. The format of data that is considered actionable is the flattened table format noted in the Data Warehouse model section. The data preparation involved the following steps:

1. Generating a TripID for each trip. This is generated by looking at

the stop ID and whenever the originating stop occurs, the trip id is incremented.

2. Consolidating different fields such as RouteID, Busid, stopid, time from multiple tables

The above two steps are performed for both schedule data and actual trip data. Now TripId can be further used to compare scheduled and actual time. As discussed above, the data is now available in OLTP tables with the trip id and stop order. Now as per the data model the data has to be flattened to the new structure.

Route	Stop	Time
Route A	Stop 1	8:00
Route A	Stop 2	8:05
Route A	Stop 3	8:10
Route B	Stop 1	9:00
Route B	Stop 2	9:15

Route	Stop1	Stop 1 Time	Stop 2	Stop 2 Time
Route A	Stop 1	8:00	Stop 2	8:05
Route B	Stop 1	9:00	Stop 2	9:15

Multiple rows per Trip to single row per trip -
Flattening Hierarchy

As shown in the above the transformation is made by repeated joins to the same table. For the schedule table,

`concat(DOW,concat('~',concat(trip_id,concat('~',Schedule_ID)))) 'integration_key'`

is used as an integration key. This is nothing but the concatenation of day of week, trip id, and the route if concatenated to create an unique key for every trip. Thus the data is available on the schedule dimension as below,

integration_key	schedule_id	trip_id	DOW	STOP1_NAME	STOP1_TIME	STOP1_ID	STOP2_NAME	STOP2_TIME	STOP2_ID	STOP3_NAME	STOP3_TIME	STOP3_ID	STOP
M~1~M-A1.1	M-A1.1	1	M	Stadium (A)	07:25:00	67	Well's Library	07:32:00	1	3rd & Jordan	07:34:00	6	IMU
M~2~M-A1.1	M-A1.1	2	M	Stadium (A)	07:49:00	67	Well's Library	07:56:00	1	3rd & Jordan	07:58:00	6	IMU
M~3~M-A1.1	M-A1.1	3	M	Stadium (A)	08:20:00	67	Well's Library	08:27:00	1	3rd & Jordan	08:29:00	6	IMU
M~4~M-A1.1	M-A1.1	4	M	Stadium (A)	08:55:00	67	Well's Library	09:02:00	1	3rd & Jordan	09:04:00	6	IMU
M~5~M-A1.1	M-A1.1	5	M	Stadium (A)	09:47:00	67	Well's Library	09:54:00	1	3rd & Jordan	09:56:00	6	IMU
M~6~M-A1.1	M-A1.1	6	M	Stadium (A)	10:29:00	67	Well's Library	10:36:00	1	3rd & Jordan	10:38:00	6	IMU
M~7~M-A1.1	M-A1.1	7	M	Stadium (A)	11:07:00	67	Well's Library	11:14:00	1	3rd & Jordan	11:16:00	6	IMU

Once the data to schedule is loaded, now the data to the actual run can be loaded. However to this new table, the unique key is the concatenation of data and the other components as described above.

id	integration_key	RECON_KEY	Route	time_date	trip_id	STOP1_NAME	STOP2_NAME	STOP2_TIME	TRAVEL2_TIME	STOP1_TIME
2015-01-13~T~2~T-B1.1	T~2~T-B1.1	T~3~T-B1.1	T-B1.1	2015-01-13	2	Fisher Court ()	ZBT	08:30:05	352	08:24:29
2015-01-13~T~3~T-B1.1	T~3~T-B1.1	T~4~T-B1.1	T-B1.1	2015-01-13	3	Fisher Court ()	ZBT	08:57:53	252	08:53:57
2015-01-13~T~4~T-B1.1	T~4~T-B1.1	T~5~T-B1.1	T-B1.1	2015-01-13	4	Fisher Court ()	ZBT	09:35:40	623	09:25:32
2015-01-13~T~5~T-B1.1	T~5~T-B1.1	T~6~T-B1.1	T-B1.1	2015-01-13	5	Fisher Court ()	ZBT	10:05:40	338	10:00:20
2015-01-13~T~6~T-B1.1	T~6~T-B1.1	T~7~T-B1.1	T-B1.1	2015-01-13	6	Fisher Court ()	ZBT	10:45:54	1072	10:28:18
2015-01-13~T~7~T-B1.1	T~7~T-B1.1	T~8~T-B1.1	T-B1.1	2015-01-13	7	Fisher Court ()	ZBT	11:23:49	97	11:22:27
2015-01-13~T~8~T-B1.1	T~8~T-B1.1	T~9~T-B1.1	T-B1.1	2015-01-13	8	Fisher Court ()	ZBT	12:05:04	1164	11:45:54

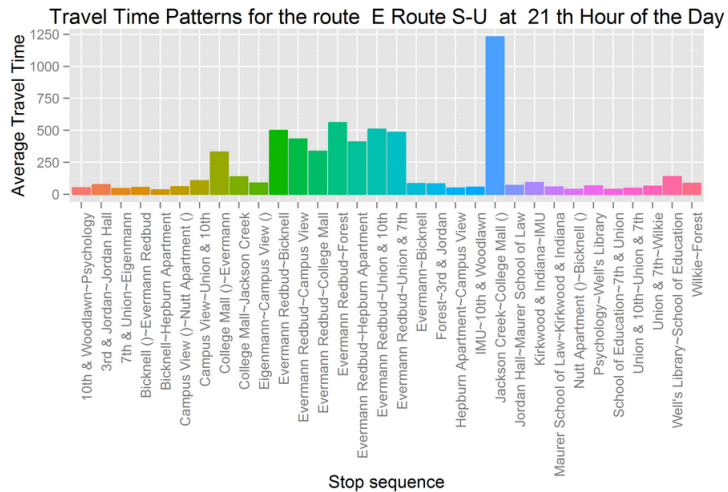
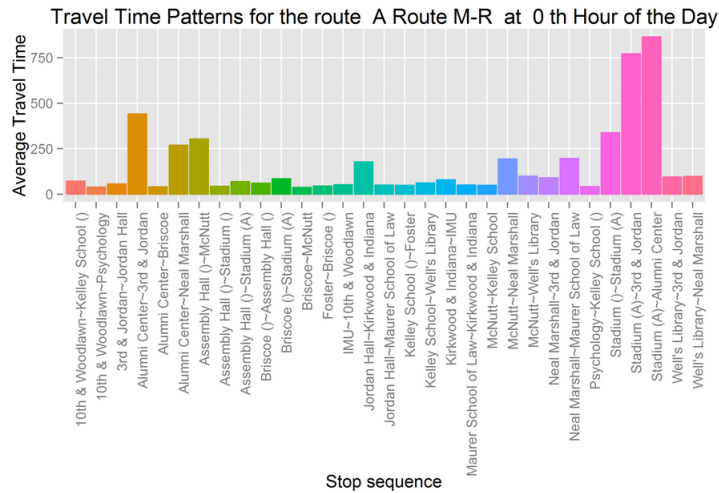
However, even with this, we would not be able to match the schedule and run. It might happen that for a given trip, the bus may not have stopped in stop 1, hence the whole trip would be lost. Hence we calculate an reconciliation key, $RECON_K EY$. It can be seen that these two keys need not be same and the new key can be used to join with the schedule table to calculate variance.

id	integration_key	RECON_KEY	Route	time_date
2015-01-13~T~2~T-B1.1	T~2~T-B1.1	T~3~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~3~T-B1.1	T~3~T-B1.1	T~4~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~4~T-B1.1	T~4~T-B1.1	T~5~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~5~T-B1.1	T~5~T-B1.1	T~6~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~6~T-B1.1	T~6~T-B1.1	T~7~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~7~T-B1.1	T~7~T-B1.1	T~8~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~8~T-B1.1	T~8~T-B1.1	T~9~T-B1.1	T-B1.1	2015-01-13

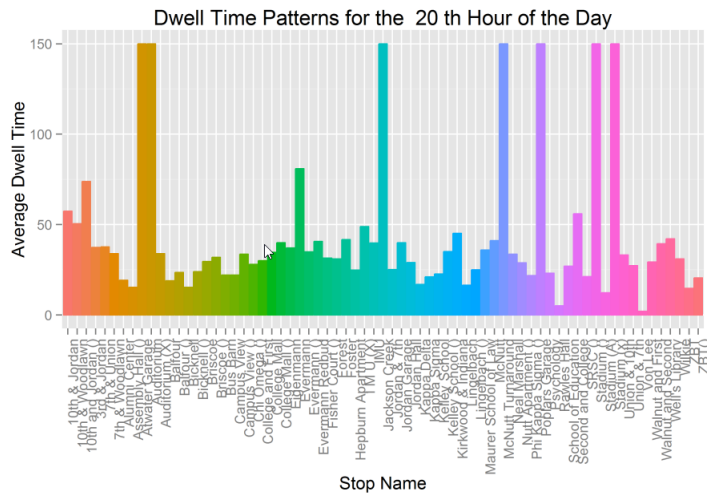
The variance was just then the select away and is calculated directly.

4 Exploratory Analysis

Any Start of Data Analytics involves understanding of the data. An initial Exploratory data analysis was made using R and ggplot2 library. For example the below graph indicates the average time between the pairs of stops for the 12 AM for Route A. It can be seen that the travelling time is very less during morning 12AM.



However, a change in trend happens during the "peak times". During evening 7, the average time traveling time increases drastically, for Eg, 10th and Wood Lawn to psychology takes less than a minute during morning, however during evenings, it double to about 2 minutes. The same trend can be seen at below plots.



All these graphs and more visualizations are available in the Github Repository, <https://github.com/sarvothaman/nefarious-octo-rutabaga/tree/master/EDA>

5 Reporting

The Main objective of this case study is to report how much the busses run in accordance to the schedule. ie, the time deviation $t_{schedule} - t_{actual}$ is an important metric. Tableau reporting solution was used to report these findings.

Tableau provides an excellent platform for developing these reports. The variance reports are reports which gives the flexibility to select the month, data, route and trip and check the variance in each stops. This is available on cloud and can be accessed by anyone for free. Below is the screenshot of the application hosted.

Stadium						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	Stadium (A)	11:13:00	11:14:08	-68
Wells Library						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	Wells Library	11:20:00	11:21:52	-112
3rd and Jordan						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	3rd & Jordan	11:22:00	11:26:10	-250
IMU						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	IMU	11:32:00	11:31:42	18
Stadium Return						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	Stadium (I)	11:40:00	11:39:14	46

Month
☐ (All)
☒ January
☐ February
☐ March
☐ April

Date
1/13/2015
4/30/2015

Route
☐ (All)
☐ M-A1.1
☐ M-A3.1
☒ M-A4.1
☐ R-A1.1
☐ R-A3.1
☐ R-A4.1
☐ R-A5.1
☐ T-A1.1
☐ T-A3.1

Trip #
☐ (All)
☐ 0
☐ 1
☐ 2
☒ 3
☐ 4
☐ 5
☐ 6
☐ 7
☐ 8

For example, the above shown trip is the 3 trip of A1 on January 26th 2015. It can be seen that the departure from stadium A is at 11:13, however the bus departed at 11:14 bringing in an one minute delay. It is interesting that in 3rd and Jordan, the was about 3 minutes late, however it reached IMU one minute early. This suggests that there is still space for optimizing the schedule. Multiple trips can also be selected as shown below to access trip information.

Stadium						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	Stadium (A)	08:20:00	08:20:47	-47
3/10/2015	T-A1.1	3	Stadium (A)	08:55:00	08:37:42	1,038
3/24/2015	T-A1.1	3	Stadium (A)	08:20:00	08:05:30	870
Wells Library						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	Wells Library	08:27:00	08:25:25	95
3/10/2015	T-A1.1	3	Wells Library	09:02:00	09:03:48	-108
3/24/2015	T-A1.1	3	Wells Library	08:27:00	08:25:02	118
3rd and Jordan						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	3rd & Jordan	08:29:00	08:28:39	21
3/10/2015	T-A1.1	3	3rd & Jordan	09:04:00	09:07:18	-198
3/24/2015	T-A1.1	3	3rd & Jordan	08:29:00	08:28:46	14
IMU						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	IMU	08:39:00	08:34:05	295
3/10/2015	T-A1.1	3	IMU	09:14:00	09:12:58	62
3/24/2015	T-A1.1	3	IMU	08:39:00	08:33:30	330
3/31/2015	T-A1.1	3	IMU	09:14:00	09:12:22	98
Stadium Return						
Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	Stadium (I)	08:47:00	08:39:47	433
3/10/2015	T-A1.1	3	Stadium (I)	09:22:00	09:20:10	110
3/24/2015	T-A1.1	3	Stadium (I)	08:47:00	08:39:52	428
3/31/2015	T-A1.1	3	Stadium (I)	09:22:00	09:20:30	80

Month
☐ (All)
☐ January
☐ February
☒ March
☐ April

Date
1/13/2015
4/30/2015

Route
☐ (All)
☐ M-A1.1
☐ M-A2.1
☐ M-A3.1
☐ M-A4.1
☐ M-A5.1
☐ R-A1.1
☐ R-A2.1
☐ R-A3.1
☐ R-A4.1

Trip #
☐ (All)
☐ 2
☒ 3
☐ 4
☐ 5
☐ 6
☐ 7
☐ 8
☐ 9

Status Reports

Sometimes we are interested in seeing that only whether the bus has arrived

on-time or early to the main stops. The below report is called the logistic report and is available on cloud. This gives the user the option to drill down a specific date, month, route then trip and displays the status.

Logistic Report							Hr Day	
Trip Id	Route	Stop 1 Status	Stop 2 Status	Stop 3 Status	Stop 4 Status		7	15
February 2, 2015, M, 1	M-A3.1	Delayed	On Time	Early	Delayed	9	<input type="checkbox"/> (All) <input type="checkbox"/> January <input checked="" type="checkbox"/> February <input checked="" type="checkbox"/> March <input checked="" type="checkbox"/> April	
	M-A4.1	Early	Early	Early	Early	9		
	M-A5.1	Early	On Time	Early	Delayed	9		
February 2, 2015, M, 2	M-A1.1	Delayed	Delayed	On Time	Delayed	8	<input checked="" type="checkbox"/> M-A1.1 <input checked="" type="checkbox"/> M-A2.1 <input checked="" type="checkbox"/> M-A3.1 <input checked="" type="checkbox"/> M-A4.1 <input checked="" type="checkbox"/> M-A5.1 <input checked="" type="checkbox"/> R-A1.1 <input checked="" type="checkbox"/> R-A2.1 <input checked="" type="checkbox"/> R-A3.1 <input checked="" type="checkbox"/> R-A4.1 <input checked="" type="checkbox"/> R-A5.1 <input checked="" type="checkbox"/> T-A1.1 <input checked="" type="checkbox"/> T-A2.1 <input checked="" type="checkbox"/> T-A3.1 <input checked="" type="checkbox"/> T-A4.1 <input checked="" type="checkbox"/> T-A5.1 <input checked="" type="checkbox"/> W-A1.1 <input checked="" type="checkbox"/> W-A2.1 <input checked="" type="checkbox"/> W-A3.1 <input checked="" type="checkbox"/> W-A4.1 <input checked="" type="checkbox"/> W-A5.1	
	M-A3.1	Delayed	Delayed	On Time	Delayed	10		
	M-A4.1	Delayed	On Time	Early	Delayed	9		
	M-A5.1	On Time	On Time	Early	Delayed	10		
February 2, 2015, M, 3	M-A1.1	On Time	On Time	Early	Delayed	8		
	M-A3.1	Delayed	Early	Early	Early	10		
	M-A4.1	Delayed	Early	Early	Early	10		
	M-A5.1	Delayed	Early	Early	Early	11		
February 2, 2015, M, 4	M-A1.1	Delayed	Early	Early	Early	9		
	M-A3.1	Delayed	Delayed	Delayed	Delayed	11		
	M-A4.1	Delayed	On Time	On Time	Delayed	11		
	M-A5.1	Delayed	Early	Early	Early	11		
February 2, 2015, M, 5	M-A1.1	Delayed	Delayed	Delayed	Delayed	10		
	M-A3.1	Delayed	Delayed	On Time	Delayed	12		
	M-A4.1	Early	Early	Early	Early	12		
	M-A5.1	Delayed	Early	Early	On Time	12		
February 2, 2015, M, 6	M-A1.1	Delayed	On Time	On Time	Delayed	11		
	M-A3.1	Delayed	On Time	Early	Early	12		
	M-A4.1	Delayed	Early	Early	On Time	12		
	M-A5.1	Delayed	On Time	On Time	Delayed	13		
February 2, 2015, M, 7	M-A1.1	Data Missing	On Time	Early	Early	11		
	M-A3.1	Delayed	Delayed	Delayed	Delayed	13		
	M-A4.1	Delayed	Delayed	On Time	Delayed	13		
	M-A5.1	Delayed	Early	Early	Early	13		

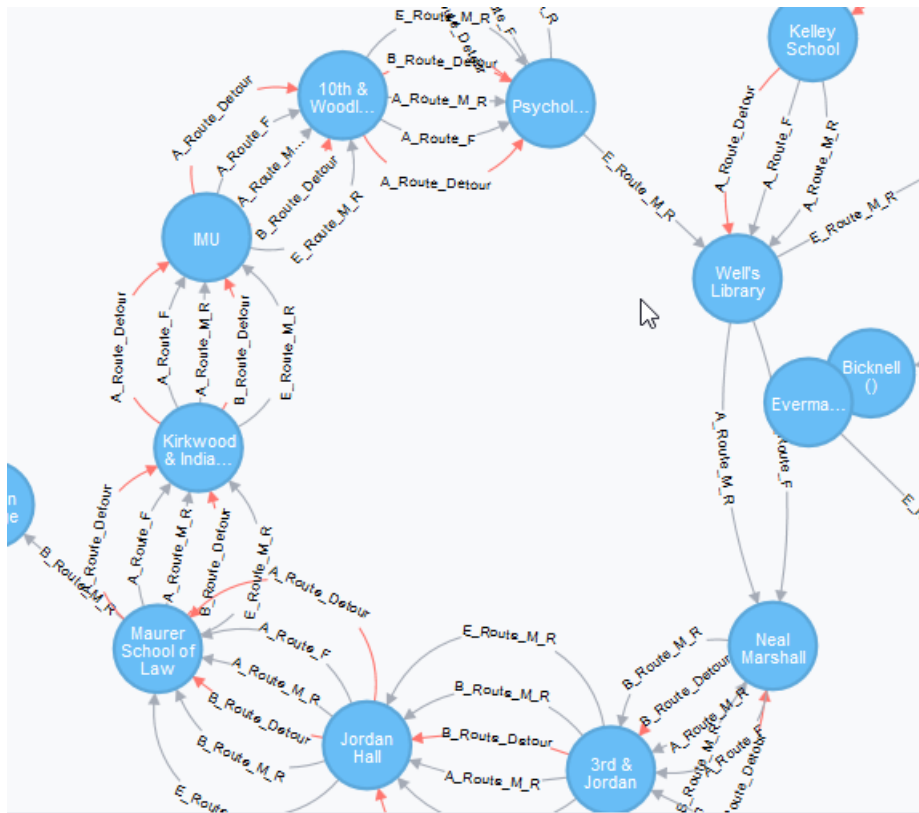
For example the February 2nd, A4 route, 1st trip reached all its stop ahead time.

Thus care has been taken during this case study that due importance is given to usability of the output artifacts and can be used by the business rather than the esoteric data miners.

6 Visualization

Another aspect of the data is the ability to visualize the data to understand the route. Neo4j is used as the background data store for saving node data. pyMysql module was used to connect to mysql and py2neo was used to connect to the cloud instance.

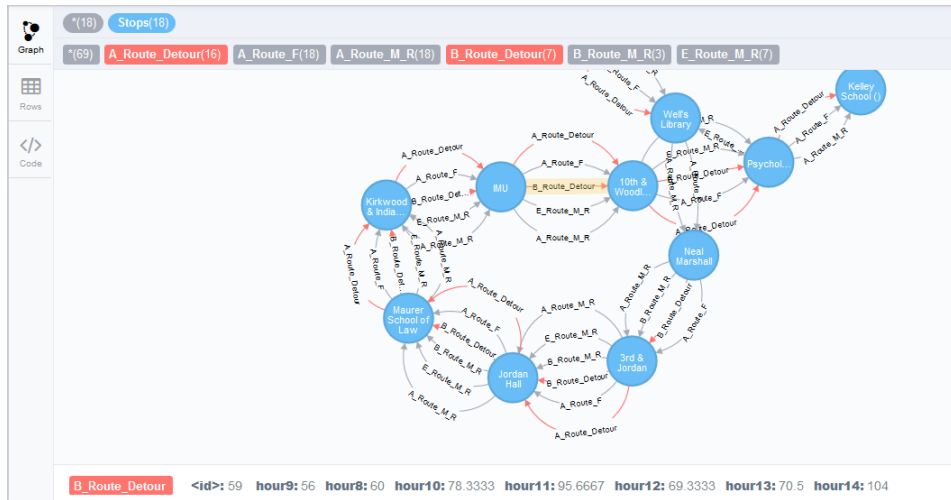
A python program was written to aggregate the data from MySql and load it to cloud Neo4j instance. Once the entire data is loaded, the graph looks as follows.



Neo4j uses the language called Cypher instead of the SQL query. An example cypher query is as follows

```
MATCH p =(a:Stops { name: "Neal Marshall" })
-[:A_Route_M_R *]-
(b:Stops { name: "IMU" })
RETURN p, length(p)
```

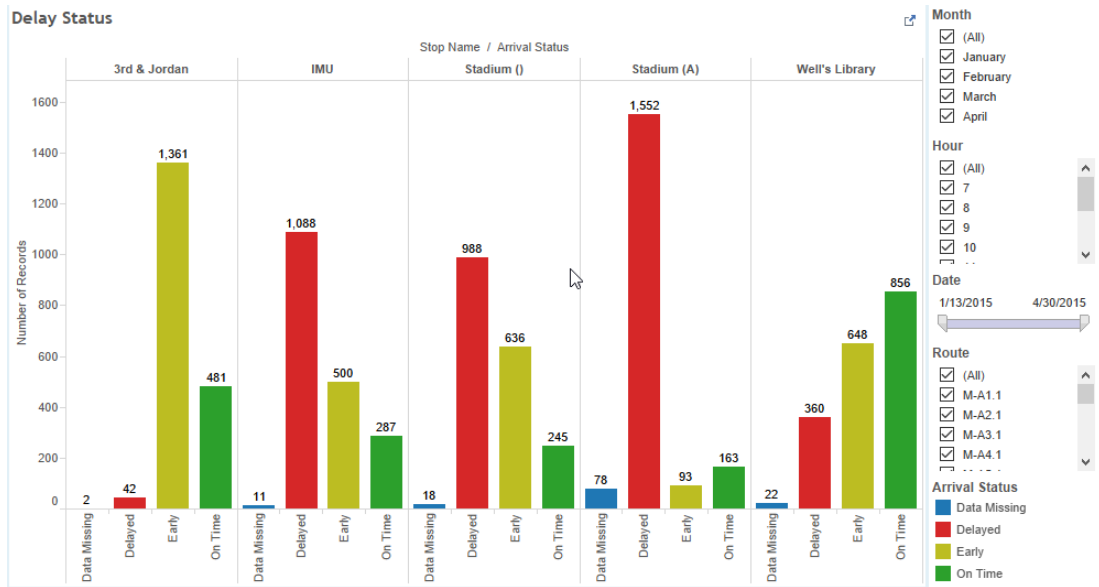
The above cypher selects a path from Neal Marshall to IMU and follows the route A in the Monday to Thursday schedule.



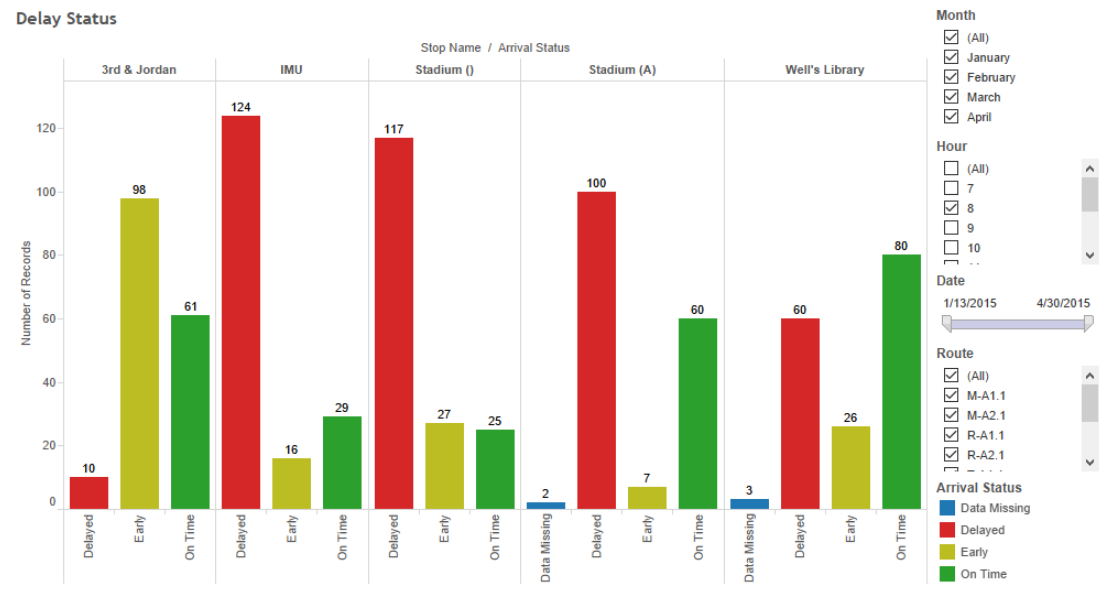
It can be seen below that there are parameters like hour9 indicates the travelling time at 9th hour of the day. This was used to predict the delay/arrival time and most of the time it predicted the time accurate upto Plus or minus three minutes.

Status Reporting

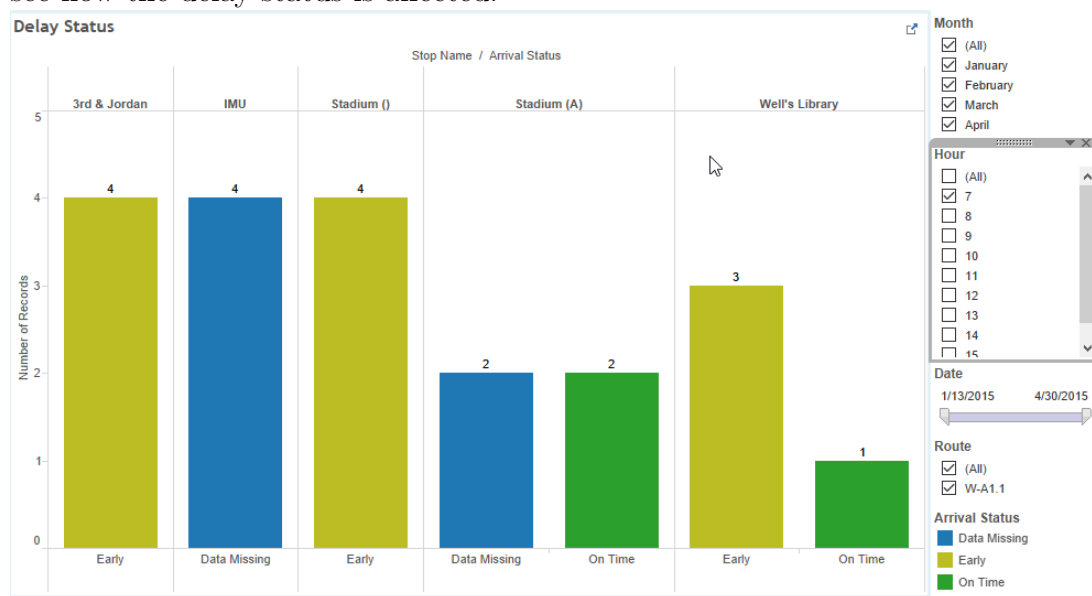
As seen in above reports, for effective decision making the counts of instances bus statuses over a period of time is important. The below plots help in such interpretation,



Eg, the above graph gives the count of the instances of the statuses. Generally it can be seen that stadium is always delayed and wells library is always on time.



This also lets the user to select the hour of the day, month, day, route and see how the delay status is affected.



In refreshing change, During the mornings, for most of the stops the bus reaches either early or on time.

7 Statistical Model

Logistic Regression Model

The data from the MySql database was made longitudinal again and was exported as an CSV file. This file as sourced to the R.

A logistic Regression model was built over this dataset to predict whether the thr A Route would be On-Time, or Delayed or Early.

As described earlier, if the actual time is a minute or earlier than the schedule time, it is marked Early. If the bus arrives a minute before or after the scheduled time, then it is termed as On-Time. If the bus is delayed by more than a minute, it is termed as late.

In essence,
 On time : one min before and after the scheduled time
 Early: Earlier than a minute to the scheduled time
 Late: Later than a minute to the scheduled time

Thus, the logistic model would have 3 levels, On-Time, Early and Delayed.

****Model 1**** The model was built in such a way that the prediction in first stop would dependant on the tie of the hour and the trip of the day.
****Model 2**** The second model was built in such a way that this would consider the status of arrival in the first stop, hour and the day of travel.
****Iterative Models**** The consecutive models for stops iteratively built on this process, all constant factors and the previous stop status

```
setwd("C:\\Users\\Sarvo\\nefarious-octo-rutabaga\\R Models\\Data")
library(nnet)
routea<-read.csv("DATA.csv")
# Stop 1 Model
stop1.fit<-multinom(Stop.1.Status~Day+Hr.Day,data=routea)

## # weights:  24 (15 variable)
## initial  value 2614.551165
## iter   10 value 1136.297534
## iter   20 value 1087.220212
## final   value 1087.212606
## converged

# Predict Stop 1 Status
stop1.prob<-data.frame(Day=c("M"),Hr.Day=10)
predict(stop1.fit,newdata = stop1.prob,"probs")

## Data Missing      Delayed      Early      On Time
##  0.03082123    0.77910313    0.07317609    0.11689955

#Stop 2 status
stop2.fit<-multinom(Stop.2.Status~Day+Hr.Day+Stop.1.Status,data=routea)
```

```

## # weights: 36 (24 variable)
## initial value 2614.551165
## iter 10 value 2117.757741
## iter 20 value 2011.097163
## iter 30 value 1998.078206
## iter 40 value 1997.885031
## final value 1997.884320
## converged

#Prediction for Stop 2 status
stop2.prob<-data.frame(Day=c("M"),Hr.Day=10,Stop.1.Status=c("Delayed"))
predict(stop2.fit,newdata = stop2.prob,"probs")

## Data Missing      Delayed      Early      On Time
## 0.01664346 0.14224930 0.36711064 0.47399660

#Stop 3 Status
stop3.fit<-multinom(Stop.3.Status~Day+
                    Hr.Day+
                    Stop.1.Status+
                    Stop.2.Status,
                    data=routea)

## # weights: 48 (33 variable)
## initial value 2614.551165
## iter 10 value 861.842048
## iter 20 value 708.720592
## iter 30 value 685.807742
## iter 40 value 684.351992
## iter 50 value 684.315511
## final value 684.313809
## converged

#Prediction for Stop 3
stop3.prob<-data.frame(Day=c("M"),Hr.Day=10,
                       Stop.1.Status=c("Delayed"),
                       Stop.2.Status=c("On Time"))
predict(stop3.fit,newdata = stop3.prob,"probs")

```

```

## Data Missing      Delayed      Early      On Time
## 3.381334e-03 7.365777e-11 7.682267e-01 2.283920e-01

# Stop 4 Model
stop4.fit<-multinom(Stop.4.Status~Day+
                    Hr.Day+
                    Stop.1.Status+
                    Stop.2.Status+
                    Stop.3.Status,
                    data=routea)

## # weights:  60 (42 variable)
## initial  value 2614.551165
## iter   10 value 1261.966796
## iter   20 value 1136.788612
## iter   30 value 1099.663056
## iter   40 value 1096.691497
## iter   50 value 1096.544668
## iter   60 value 1096.531568
## final   value 1096.531328
## converged

#Prediction
stop4.prob<-data.frame(Day=c("M"),
                       Hr.Day=10,
                       Stop.1.Status=c("Delayed"),
                       Stop.2.Status=c("On Time"),
                       Stop.3.Status=c("On Time"))
predict(stop4.fit,newdata = stop4.prob,"probs")

## Data Missing      Delayed      Early      On Time
## 2.966553e-18 9.588480e-01 8.424239e-12 4.115202e-02

```

A keen observer can see that this prediction is as same as the results off Tableau Graphs. Even there, the pattern that was observed is that if the bus starts delayed, it comes on-time to Stop 2, and comes Delayed to Stop 3.

Thus this model can give a categorical answers to queries about the ex-

isting status of the bus.

8 Concluding remarks

Throughout our exploratory analysis and modeling we were able to make observations that are consistent with reality (verified by logic as well as with actual data) as well as make certain predictions (using neo4j model for example). While the results are preliminary and simple in nature, the framework and methods of analysis can be easily extended. We consider this an important step, since ability to quickly analyze the data is critical in achieving the overall objectives

9 Future Work

10 References

References

- [1] Indiana University. http://www.iubus.indiana.edu/campus_bus/index.html.
- [2] University of Virginia. <http://www.virginia.edu/parking/uts/>.
- [3] Boston University. <http://www.bu.edu/thebus/>.
- [4] Odacir A. Neves Marcos E. Cintra. A fuzzy decision tree for bus network management. Universidade Federal Rural do Semi-Arido, Brazil.
- [5] Indiana University. http://www.iubus.indiana.edu/campus_bus/bus_schedule.html.
- [6] NOAA. <http://www.ncdc.noaa.gov/cdo-web/search>.