Internet of Things Report - Air Pollution-Based Traffic Management System

May 10, 2016

Main Report

1 Scenario Outline

At the Paris climate COP21 conference in December 2015, 195 countries adopted a universal climate deal which sets out to limit global warming to well below 2°C until 2100. The agreement is due to enter into force in 2020 and will force authorities around the globe to rethink their national climate policies.

Air pollution has long been directly linked to traffic and congestion levels in cities. It is no surprise today that cities with high congestion levels exhibit high levels of particulate matter (PM) and are a health hazard to residents. The World Health Organization measured London's PM10 to be 23 ug/m3, which is comparable to Paris and New York (24 and 23 respectively) [1]. However, TomTom rates London as the 16th most congested city in the world [2].

Automatically optimising traffic and air quality are conflicting objectives but can be tackled with the power of computing. Traffair aims to mitigate traffic congestion while meeting air quality standards, which are policies driven by the COP21 agreement, by providing real-time traffic management system based on air pollution readings at various locations in London.

The project is based on an existing research project called iTRAQ [3] which gathers air quality information from high rises and from space to provide a traffic management system. We aimed to recreate this project by using sensor readings from the ground instead.

2 Design Methodology and Implementation

2.1 Architectural Overview

The Traffair Traffic Management System predicts the level of traffic in various locations in London based on the level of NO2 at these locations and provides a recalculated route based on the predictions. In order to achieve this, the Traffair system consists of a Computational Intelligence module (CI) which is a statistical model that has been fed with real-time traffic and gas readings over a period of 2 weeks (See Appendix A [5.4]). The system also features a Traffic Prediction module (TP) which uses the CI model to compare real time data coming from the sensors to the statistical model and offer a traffic level prediction for each of the locations. Finally, the system features a Routing Algorithm (RA) which uses the predicted traffic level coming from the TP module to output a redirection route. All this information is displayed in a user friendly dashboard (See Appendix B [5.4]) and a real-time database/API to extend the system's usage (See https://traffair.firebaseio.com/data.json).

2.2 Design

2.2.1 Gathering Research Information

Before building Traffair, we set out on a long period of time of research to discover ideas that fit the IoT & Big data concepts. This meant that we had to find a project that fit the following criteria:

- Heavily depends on sensor use.

- Can be deployed and tested or simulated easily.
- Gathering a lot of data will provide added value to the service.
- Data gathered provides rich and valuable information.
- The data gathered can be fed back to the module/sensor to improve their readings/processing (adaptive readings) or enhance the system (for example, an adaptive recommendation search engine based on previous search history).
- Does not only "alert" or provide "smart" features.

Keywords guidance for the brainstorming sessions included: management, efficiency, control, occupancy, metering, and patterns recognition.

We finally stumbled upon an existing research project from De Montfort University, Leicester called iTRAQ which is an integrated traffic management and air quality control system that used space services [4]. Though the research already had extensive resources and progress, we decided to deploy a similar concept based on ground rather than space readings.

2.2.2 Preparation

While preparing for this project we developed the following action points:

1. **Finding data to build the statistical/Machine Learning model:** This consisted of figuring out which database or APIs already existed that could provide us with gas and traffic readings around London. We investigated the London Data Store [5] which only provided offline non-realtime data, the iTRAQ project who was not too keen on sharing data, the London Air API and the various Traffic APIs from HERE Maps, Google Maps, Waze, and Miscrosoft.
2. **Investigating which sensors and edge computers are most appropriate:** We quickly settled on the Raspberry Pi 3 as the edge computer because it provided an integrated Wifi chip, making it perfect for IoT applications. The Raspberry Pi also comes with a dedicated connector for a camera which we could use for traffic monitoring based on image recognition (more on its lack of use in the Engineering Tradeoff section [3.4]). Additionally, the hardware's development language was based in Python which matched the groups' skills better than its C-language counterpart featured in Arduino devices. We originally set out to use temperature, humidity, Carbon Monoxide, and Nitrogen Dioxide sensors and finally integrated only the Carbon Monoxide sensor (More on this in the Engineering Tradeoff section 3.3).
3. **Investigating which Cloud Platform is most appropriate:** Google Cloud had recently launched which featured many of Google's renowned prediction services. However, with the support provided internally for IBM Bluemix and the appeal of Watson services, we decided to use IBM Bluemix as the Cloud platform that would power Traffair.
4. **Investigating which Data Analytics Platform is best:** This decision came further along in the process when we realized that IBM offered Watson Predictive Analysis API for realtime usage. For offline analytics, we used a mix of Matlab and Excel which matched the skill level of our group.
5. **Establishing Timeline for Development & Meetings:** Once the research and preparation period was successfully concluded, we developed a timeline for meetings and deliverables (See Appendix C for Gantt Chart [5.4])

2.3 Implementation

2.3.1 Building the Computational Intelligence Module

The CI module was built by gathering NO2 readings at 3 nearby South Kensington locations (See Appendix B [5.4]) from the London Air API [7]and matching them with their current traffic flow readings from the HERE Traffic API [6] into a database. From this database, we created a classifier with location as a nominal input class along with time of the day and NO2 level as continuous inputs which left the traffic level (or Jam Factor) as the targeted output of the statistical model.

2.3.2 Building the Traffic Prediction System & Routing Algorithm

This model was then fed into the IBM Bluemix Cloud platform as the TP module, powered by IBM's Watson Predictive analysis & SPSS Modeler. The TP module outputs the predicted traffic level for a given NO2 and timestamp (time of the day) reading. We then built a rudimentary Routing Algorithm module within the NodeRED platform which redirects traffic from highly congested areas to lowly congested areas.

2.3.3 Integrating Sensors & Edge Computer

Having the TP system in place, we got rid of the London Air API KC5 location, which was used as one of the NO2 inputs, and replaced it with our own sensor which simulated the South Kensington KC5 location. The hardware is built using the Raspberry Pi 3 as the edge computer along with Arduino for ADC capabilities and actually uses Figaro TGS 2442 Carbon Monoxide Gas Sensor (more about this decision in the Engineering Tradeoff Section [3.3]).

2.3.4 Monitoring Dashboard System & Real-time Database

When the Cloud implementation was finished, the data gathered in real time included the gas readings, the predicted traffic level (or Jam Factor) along with the confidence index, the date and timestamp of the readings, the suggested redirect route along with the coordinates of the redirect to/from location. We uploaded this data to a real-time database (See https://traffair.firebaseio.com/data.json). that can be used as an API to extend the use of the Traffair System (eg. Twitter integration). Additionally we developed a monitoring dashboard which displays these values in real-time (See Appendix A [5.4]).

2.3.5 Offline Analytics

The data pushed to the dashboard and API was also downloaded for offline data analytics. The original classifier based on the London Air API & HERE Traffic API readings was also downloaded for offline data analytics. We developed comparative models in Excel and regression analysis in Matlab.

3 Engineering Trade-Offs

Traffair is a proof of concept and most probably lacks the accuracy that the iTRAQ project is able to provide. Due to the limited number in development team members and the time constraint, the project had to undergo a few engineering tradeoffs. The inaccuracies of the project were the results of the following engineering tradeoffs.

3.1 Using APIs instead of In-House Sensors

In order to fit the project within the time constraint, we decided to use external data to build our classifier and statistical model instead of deploying our own sensors in the wild for two weeks. This way, we were able to save at least 2 weeks in development time. This decision was also backed by the fact that we wouldn't be able to leave 3 sensor-equipped micro-computers in the wild for 2 full weeks. Unfortunately, the London Air API did not provide a constant stream of data while the Cloud system was polling both APIs every 15 minutes regardless. This created mismatch in data between the gas readings and their respective traffic readings. For example, traffic level measured at 8 AM on Sunday would see the same NO2 level as if it were taken on Monday 3PM. This created high error rates in the statistical model.

3.2 Choosing a "Universal" Timestamp

Because of the London Air API inaccuracies, our data ended up with 3 different timestamps. The first one being the time at which the gas readings were last recorded (from the London Air API). The second being the time at which the traffic was last recorded at that location (from the HERE Traffic API). The last, and having inherently the same value as the HERE API timestamp, being the time at which this data was polled (provided by the IBM Bluemix platform). We chose to use the IBM Bluemix timestamp as the universal timestamp across the application to provide a sense of real-time even though it incurred discrepancies in the statistical model. Had we used the London Air API timestamp instead, we would have had much less data points (since the data didn't update rapidly enough) over these two weeks and hence unable to move forward with the project. In hindsight, we could have selected that option instead and increase the period of data gathering to a full month instead.

3.3 Selecting Sensors

The iTRAQ project uses various meteorological data for their forecasting algorithm such as temperature, air pressure, or wind speed (See Appendix D [5.4]). We initially set out to measure temperature, humidity, carbon monoxide, and nitrogen dioxide. However, due to added complexity of integrating yet another API (Weather) and the build up of inaccuracies in the statistical model, we took the decision to only measure nitrogen dioxide. At that point, we had already ordered the sensors and carbon monoxide was the only one available. Since we had already undergone the development of the system based on nitrogen dioxide at the South Kensington locations, we decided to keep the CO sensor and translate it to NO2 readings. We used readings from Columbia University [8] about the composition of air and developed a ratio between traces of CO and NO2 to convert our readings to NO2 readings. Obviously, this would not be implemented in a commercialized solution.

3.4 Image Processing for Traffic Prediction

When initially investigating the scope of the project, we set out to build our own classifier based on readings from our own sensors (see Using APIs instead of In-House Sensors [3.1]). This also meant that we would have to estimate traffic level at the sensor and is the reason why we initially ordered Raspberry Pi Cameras. While developing the vehicle detector using OpenCV, we realized that the accuracy of the model was very low and instead we used the HERE Traffic API to estimate the level of traffic at the sensor location. This solution still provided some discrepancy but was probably lower than that of the vehicle detection classifier.

3.5 Number of Traffic Nodes & Routing Algorithm

We decided to only use 3 locations instead of a larger network. This was mainly due to the fact that implementing this solution for a very large network would have been quite challenging given the time constraint. 3 nodes gave the opportunity to show the potential of Traffair while keeping the development of a routing algorithm feasible. The algorithm itself is mainly hard coded, meaning that jam factor scores are grouped together in many small groups of cases (routing if-statements). Moving forward, a more comprehensive and dynamic algorithm would have to be developed.

4 Commercial / Market Considerations

4.1 Target Market

With the recent COP21 agreement legally binding 195 countries to limit global warming to 2°C until 2100, governmental authorities in highly congested cities will be looking of solutions that reduce their carbon footprint. Additionally, these local authorities will be interested to match the sustained concern to provide high air quality levels for their residents. Cities ranking highest in congestion and pollution level include Rio de Janeiro, Tianjin, Beijing, or Hangzhou - all scoring above 40% congestion level [2] and 40 ug/m3 of PM10 [1].

4.2 Product/Service Offering

The true power of Traffair is through its real-time API capabilities rather than its stand-alone monitoring dashboard. Like many other API providers, the Traffair pricing strategy will be based on a monthly subscriptions with pricing tiers depending on number of locations polled, API calls per month (polling frequency), and unlocking other premium features and endpoints including "redirection routes" or possibly "traffic light control".

4.3 Benefits to Traffair & Competitive Advantage

Traffair brings the cost of developing and deploying traffic management solutions down as sensors become less complex and data streamed is much smaller than video streaming (in comparison to traffic cameras). Deploying Traffair in cities can result in the improvement of the air quality of affected regions and hence drop mortality rates tied to air pollution. Because of its modular nature, Traffair can be integrated with any equipment that can output JSON data, meaning customers can integrate Traffair with their own equipment and bring the deployment cost further down. Traffair is opened to integration through its public API that allows developers to build customized applications.

4.4 Costs & Market launch

In order to launch Traffair, sensors will have to be initially deployed throughout the targeted cities which will consist of the main cost of operations. This can be covered by setting up governmental auction bids to deploy this equipment similar to telecommunication companies' bids for laying out optical fibre. The Traffair service will have to be able to monitor, deploy, and repair these sensors while providing test results on the quality of forecast, optimization, and overall gain for governmental authorities.

4.5 Differences in Implementation and Design for Commercial Use

As it is today, Traffair is a rudimentary version of the overall potential solution. Before commercialization, Traffair will have to undergo an overhaul of its routing algorithm and use more data points (such as weather and atmospheric data) in its prediction algorithm. Additionally, energy management and harvesting will have to be investigated. The most viable option at the moment seems to be a seamless integration with traffic lights in terms of control and power supply.

4.6 New Verticals

In the future, Traffair's data driven approach could be exploited to explore new verticals such as telematics for freight traffic management and tracking or other smart cities applications.

5 Experimental Results

5.1 Traffic Flow Prediction Results

The figures in Appendix E [5.4] show screenshots of a 24 hour time period of the predicted traffic level compared to the measured level at each of the South Kensington locations. Though this data was gathered mostly on a Sunday, which usually exhibits a strange behavior in terms of traffic compared to the other days of the week, there is a clear trend across all three locations. Because of the accumulated discrepancies in this research project, the graphs do not exhibit exceptionally high accuracy, especially during rush hour periods. Location KC3 shows much more discrepancy which is probably due to the fact that the statistical model in IBM SPSS did not have as many diverse data points for KC3 as the other locations because of a slower update (See Using APIs instead of In-House Sensors [3.1].

5.2 Confidence level & Predictor Importance

Because of the various engineering tradeoffs discussed above, the confidence level for the predicted traffic level (Jam) is recurrently low, especially when NO2 levels are not skyrocketing and the time does not indicate "rush hour". While analysing the data, it has come to our attention that the confidence level was often much higher around 9AM and 6PM which corresponds to London's rush hour times. SPSS's model analysis actually displayed that time was viewed as a more important predictor than NO2 levels (see Appendix F [5.4]). This is probably due to the fact that timestamp data is much more constant and variable than the NO2 data gathered from the London API (See Using APIs instead of In-House Sensors in Engineering Tradeoffs 3.1).

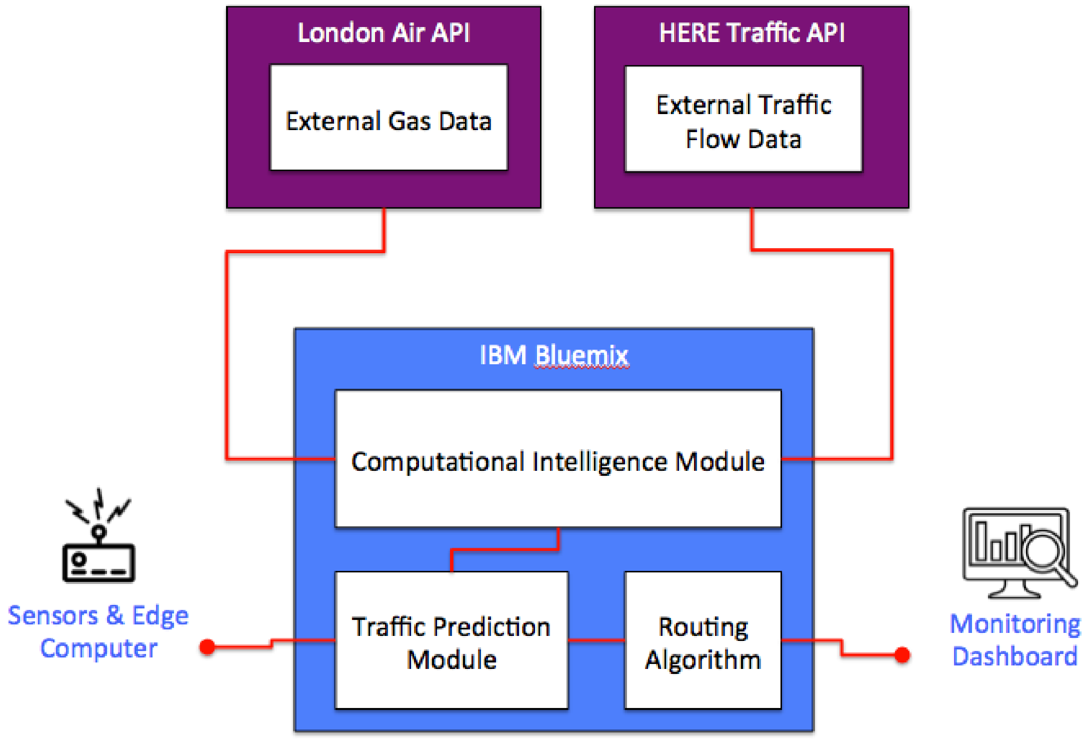
5.3 Offline Matlab Analysis

In this research's offline analysis, we applied an Artificial Neural Network Model based on our training set to classify jam factor according to the known inputs: location, air quality and time. Appendix G [5.4] illustrates the error rate of classification of the first 150 iterations. After each iteration, the neural network updates weight of each parameter, in order to explore the lowest error rate. It is normal that the error rate is fluctuant during the training process, though in this case we can clearly see the model has difficulty in establishing a constant error rate (See Using APIs instead of In-House Sensors in Engineering Tradeoffs 3.1).

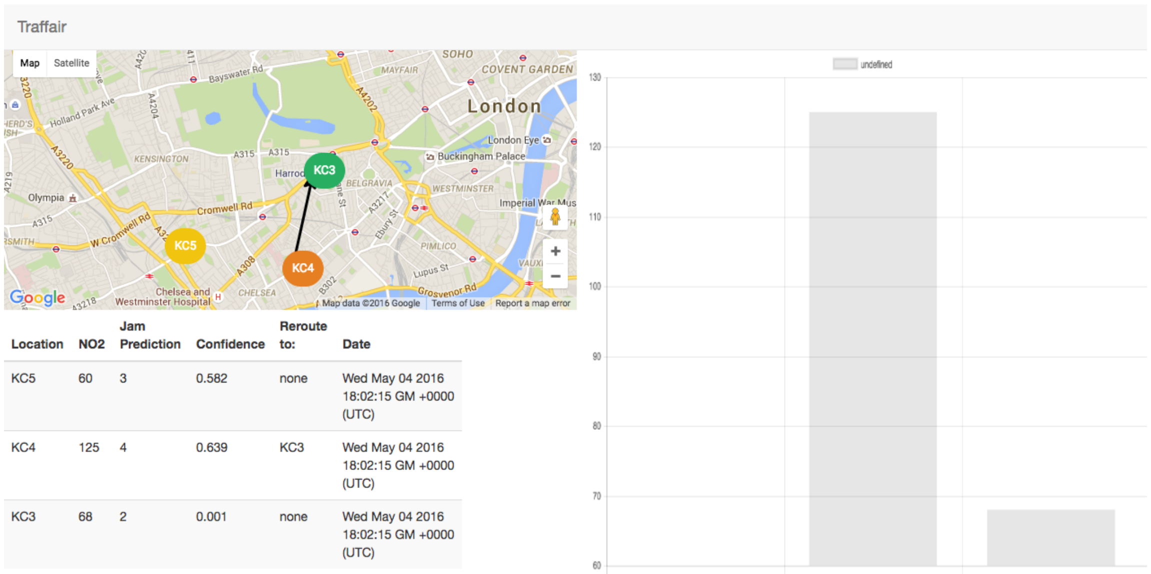
5.4 Regression

We've developed a regression analysis to estimate the relationship between the variables we observed. At first, we set the output (jam factor) as dependent variable, and the other three inputs (NO2, timestamp, location) as independent variables and analysed them into Matlab. Appendix H [5.4] shows a regression analysis over 50 data points collected. The difference between the actual and the predicted Jam Factor is mapped as the residual value while the lines describe the confidence interval of these data points. The results show that the residual values mostly fall around 0 and 2 degrees of variation compared to the actual Jam Factor data. Though a predicted Jam Factor of 3 can still somewhat represent a measured Jam Factor of 5, it is still fairly inaccurate. The confidence intervals show that the confidence of the residual values is actually fairly low and contains a wide range of possible values.

Appendix A - Traffair Architectural Model



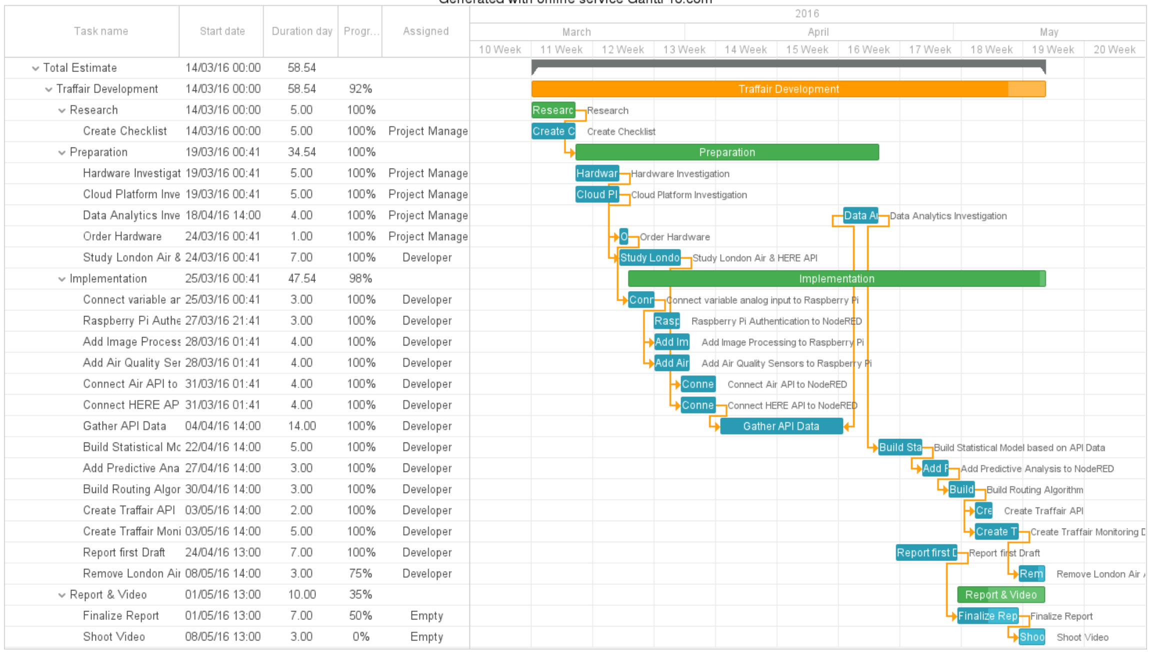
Appendix B - Traffair Monitoring Dashboard



Location	NO2	Jam Prediction	Confidence	Reroute to:	Date
KC5	60	3	0.582	none	Wed May 04 2016 18:02:15 GM +0000 (UTC)
KC4	125	4	0.639	KC3	Wed May 04 2016 18:02:15 GM +0000 (UTC)
KC3	68	2	0.001	none	Wed May 04 2016 18:02:15 GM +0000 (UTC)

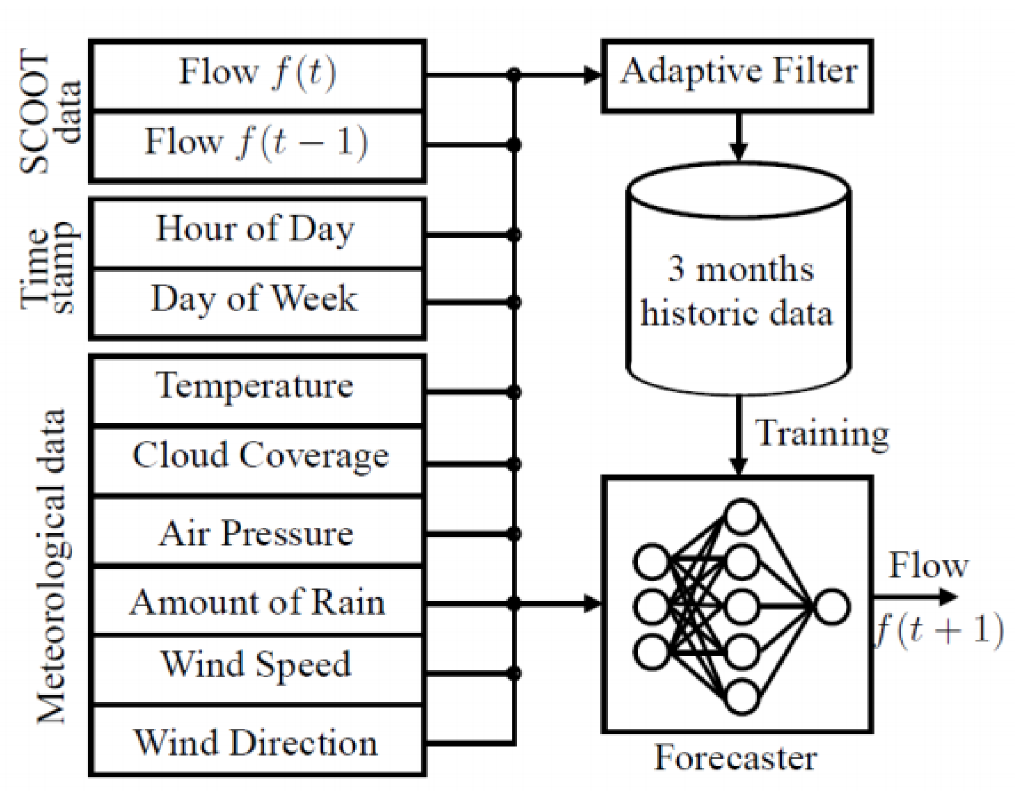
Appendix C - Project Gantt Chart

Generated with online service GanttPro.com



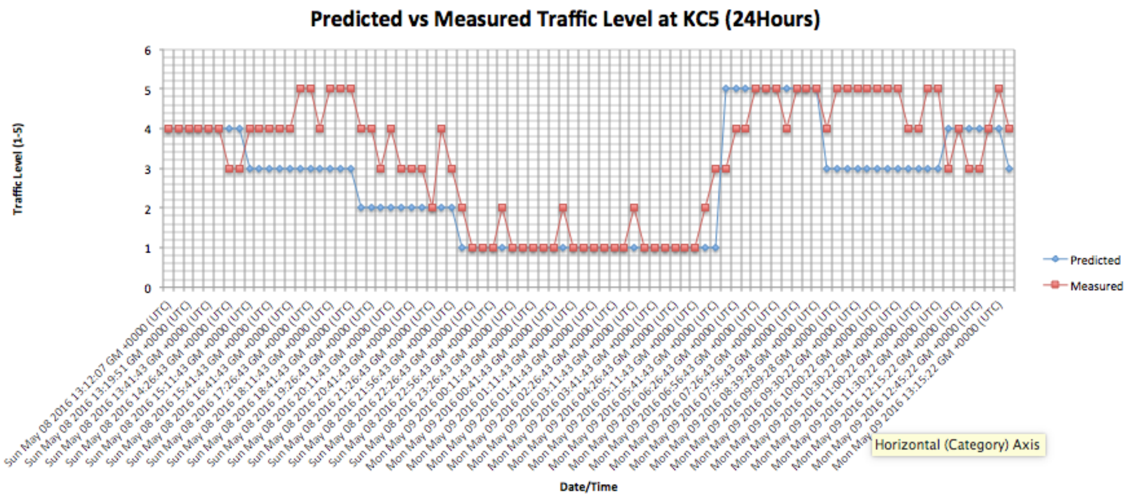
Task name	Start date	Duration day	Progr...	Assigned
Total Estimate	14/03/16 00:00	58.54		
Traffair Development	14/03/16 00:00	58.54	92%	
Research	14/03/16 00:00	5.00	100%	
Create Checklist	14/03/16 00:00	5.00	100%	Project Manage
Preparation	19/03/16 00:41	34.54	100%	
Hardware Investigat	19/03/16 00:41	5.00	100%	Project Manage
Cloud Platform Inve	19/03/16 00:41	5.00	100%	Project Manage
Data Analytics Inve	18/04/16 14:00	4.00	100%	Project Manage
Order Hardware	24/03/16 00:41	1.00	100%	Project Manage
Study London Air &	24/03/16 00:41	7.00	100%	Developer
Implementation	25/03/16 00:41	47.54	98%	
Connect variable ar	25/03/16 00:41	3.00	100%	Developer
Raspberry Pi Authe	27/03/16 21:41	3.00	100%	Developer
Add Image Process	28/03/16 01:41	4.00	100%	Developer
Add Air Quality Ser	28/03/16 01:41	4.00	100%	Developer
Connect Air API to	31/03/16 01:41	4.00	100%	Developer
Connect HERE AP	31/03/16 01:41	4.00	100%	Developer
Gather API Data	04/04/16 14:00	14.00	100%	Developer
Build Statistical Mc	22/04/16 14:00	5.00	100%	Developer
Add Predictive Ana	27/04/16 14:00	3.00	100%	Developer
Build Routing Algor	30/04/16 14:00	3.00	100%	Developer
Create Traffair API	03/05/16 14:00	2.00	100%	Developer
Create Traffair Moni	03/05/16 14:00	5.00	100%	Developer
Report first Draft	24/04/16 13:00	7.00	100%	Developer
Remove London Air	08/05/16 14:00	3.00	75%	Developer
Report & Video	01/05/16 13:00	10.00	35%	
Finalize Report	01/05/16 13:00	7.00	50%	Empty
Shoot Video	08/05/16 13:00	3.00	0%	Empty

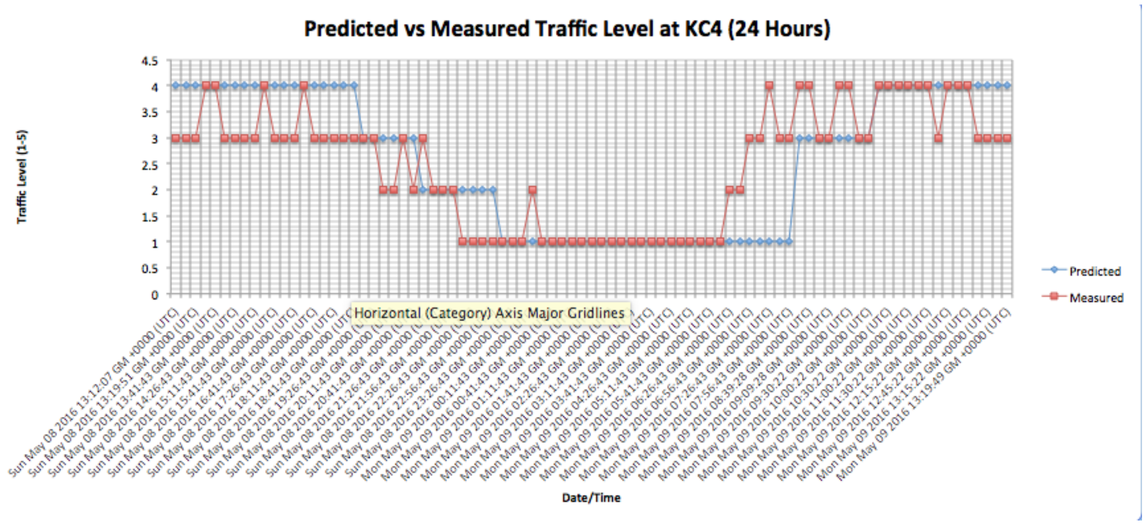
Appendix D - iTRAQ Computational Intelligence Factors

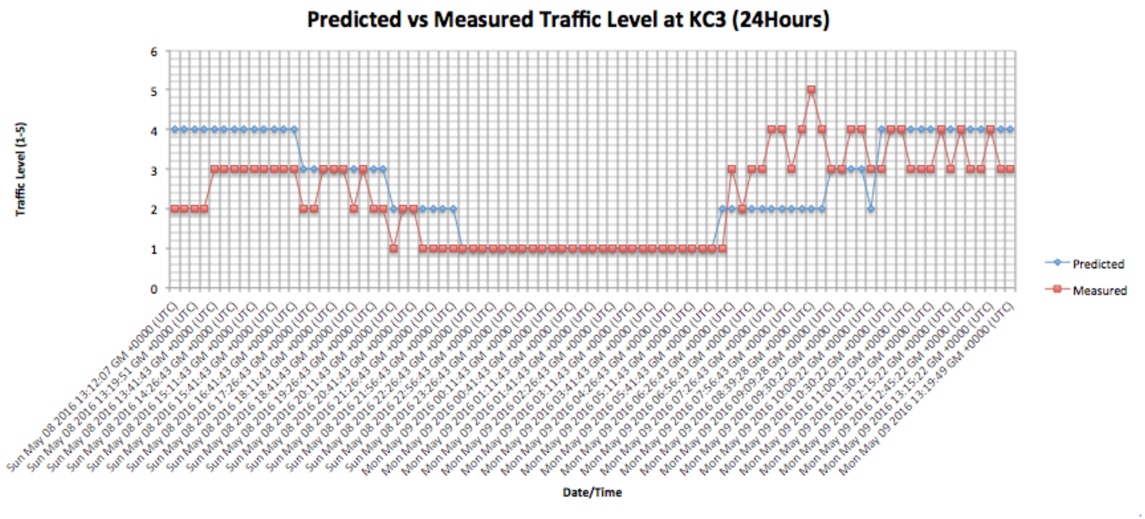


Source iTraq [4]

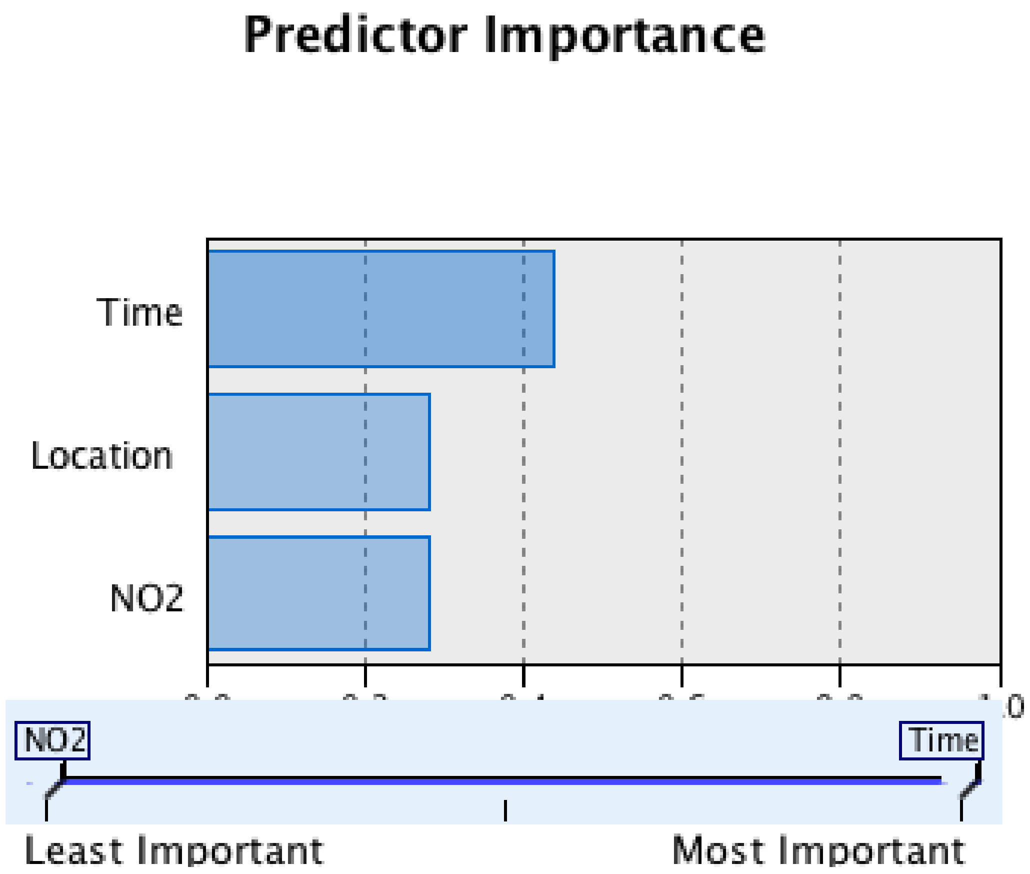
Appendix E - Comparative Results of Predicted and Measured Traffic Levels



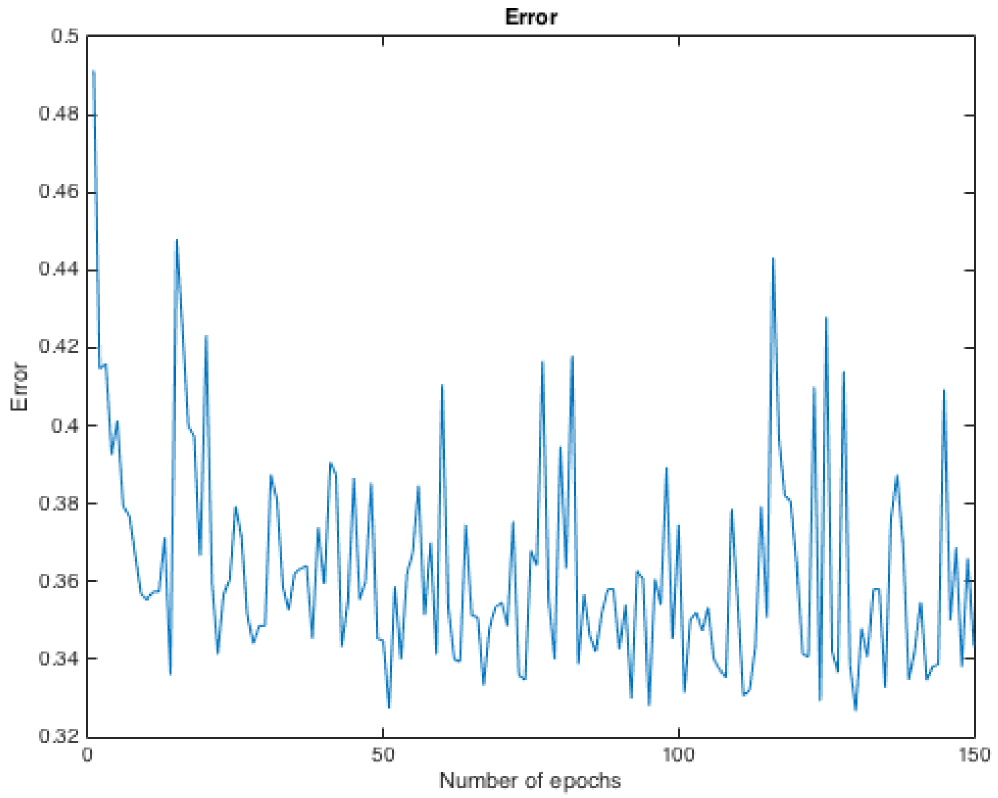

Predicted vs Measured Traffic Level at KC4 (24 Hours)
Traffic Level (1-5)
Date/Time
Predicted
Measured
Horizontal (Category) Axis Major Gridlines


Predicted vs Measured Traffic Level at KC3 (24Hours)
Traffic Level (1-5)
Date/Time
Predicted
Measured

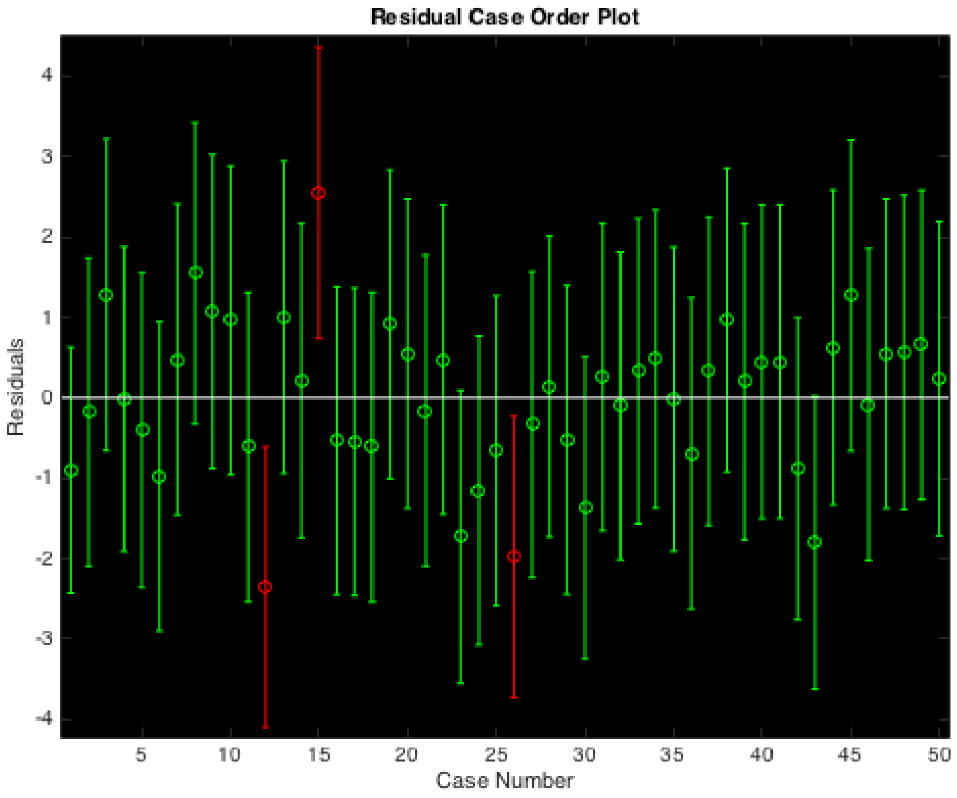
Appendix F - Predictor Importance



Appendix G - Accuracy of Jam Factor Classification



Appendix H - Residuals and Confidence Interval



References

[1] World Health Organization. Ambient Air Pollution PM Database. May 2014. Raw data. N.p. http://www.who.int/phe/health_topics/outdoorair/databases/cities/en/

[2] TomTom. "TOMTOM TRAFFIC INDEX MEASURING CONGESTION WORLDWIDE." Tomtom.com. N.p., n.d. Web. 8 May 2016. <https://www.tomtom.com/en_gb/trafficindex/list>.

[3] Elizondo, David. "ITRAQ." ITRAQ. N.p., n.d. Web. 08 May 2016. <http://www.dmu.ac.uk/research/research-faculties-and-institutes/technology/digits/partnerships-funding-and-projects/itraq.aspx>.

[4] Passow, Ben. "ITRAQ - Integrated Traffic Management and Air." (n.d.): n. pag. De Montfort University. Web. 8 May 2016. <http://ima.ac.uk/slides/BenPassow251113.pdf>.

[5] Greater London Authority. London Data Store. N.p., n.d. Web. 8 May 2016. <http://data.london.gov.uk/>.

[6] HERE Maps. "Flow." Traffic API. N.p., n.d. Web. 08 May 2016. <https://developer.here.com/rest-apis/documentation/traffic/topics_v6.1/resource-parameters-flow.html>.

[7] "London Air Quality Network || API." London Air Quality Network || API. King's College London, n.d. Web. 08 May 2016. <http://www.londonair.org.uk/LondonAir/API/>.

[8] "Gaseous Composition of Dry Air." Columbia University, 1995. Web. 8 May 2016. <http://eesc.columbia.edu/courses/ees/slides/climate/table_1.html>.