# MyWay: an Interactive Visualization System for Anormal Driving Style Detection

KETIAN XU DEARSUMMER@ZJU.EDU.CN
3120102066

## 1. Introduction

The inspiration and motivation of MyWay comes from a competition[1] on Kaggle, which is held by AXA and asks competitors to use telematics data to identify a driver signature. For this competition, Kaggle participants must come up with a "telematics fingerprint" capable of distinguishing when a trip was driven by a given driver.

We could simply see it as an outlier detection problem. However, instead of using traditional unsupervised machine learning or statistical learning algorithms to get the result automatically, I decide to build an interactive visualization system that can help *human* to easily identify trips which are not from the driver of interest *manually*.

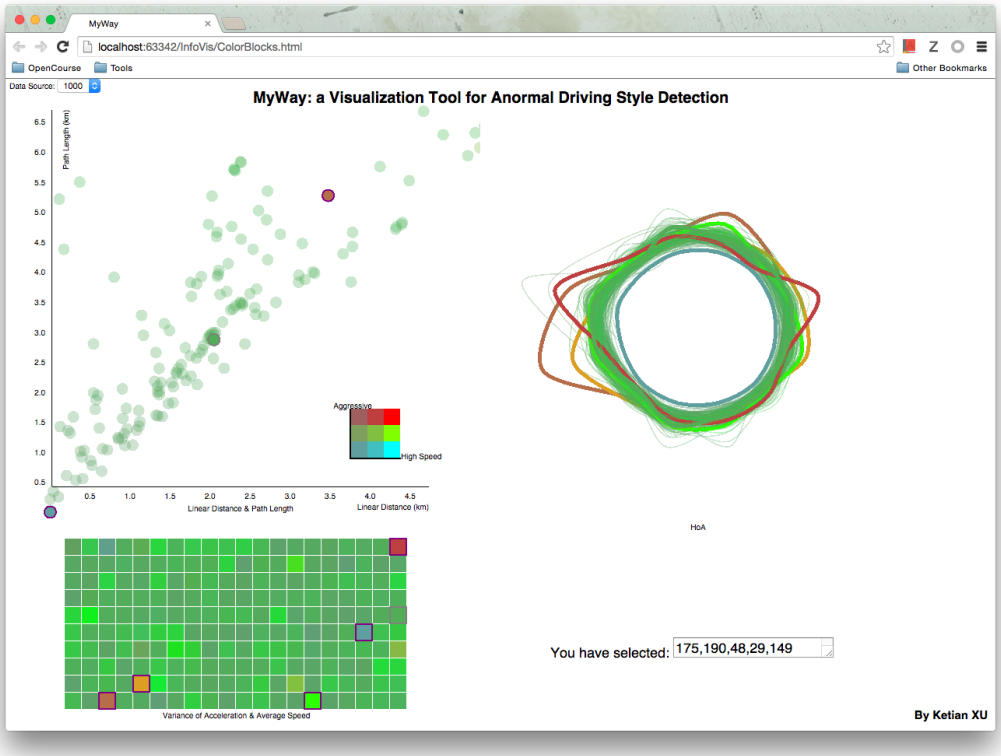


Figure 1

The rest of this report was arranged as following: Section 2 describes the steps I took to design and implement MyWay. More technical details about these steps were stated among Section 3 to Section 5, which are followed by a conclusion section.

[1] http://www.kaggle.com/c/axa-driver-telematics-analysis

# 2. Process Model

We know that experts have developed several different visualization process models including (Haber, 1990), (Card, 1999), and so on. Here is the process model I applied to understand the task and to design and implement MyWay:

**STEP 1:** Understanding the certain social phenomenon, that is, a driver may have his or her own driving type, which can be both quantitative and qualitative analysis through some factors including speed, accelerate, etc.

**STEP 2:** Understanding the certain task my visualization system will be applied to. This step is extremely important since it can help us to decide which features I may need and how should I visualize those features properly. Here, our goal is to help user easily identify trips (unknown number) which are not from the driver of interest among 200 trips.

**STEP 3:** Extracting features I think might be helpful based on STEP 1 and STEP 2. The kinds of features and the reasons for me to choose them were presented in Section 3 in detail.

**STEP 4:** After STEP 3, each trip was described by a high dimension vector. What I should do is to group features and find a proper way to visualize the data for each group. Also, I tried to ensure the visualization can really help users identify outliers. Like STEP 3, more details were introduced in Section 4.

**STEP 5:** In this step, design concepts were going to be implemented using D3.JS[2], a practical data visualization framework based on JavaScript. Section 5 includes more implementation details. Section 5 also mentions details of implementation of feature extraction.

# 3. Feature Extraction

The competition provides raw data in a directory containing a number of folders, which present different drivers. Within each folder are 200 .csv files representing driving trips, one for each. The car's position (in meters) every second is recorded in the .csv file and look like the following:

```
x,y
0.0,0.0
18.6,-11.1
36.1,-21.9
...
```

To describe a trip, raw data is just far less powerful than our expectation. Based on real-life experience, some candidate features were come up with, and are grouped as following:

- Direction (total, segmented, …)
- Distance (total, directed, …)

[2] http://d3js.org

- Speed (instantaneous, average, maximum, variance, during turns…)
- Acceleration (instantaneous, average, minimum, maximum, variance…)

Not all features are useful, and also, too many kinds of features also increase the difficulty both for me to visualize and for users to analysis. However, unsupervised dimension reduction algorithms like PCA someway lead inference and reasoning impossible. No doubt it is needed to do feature selection. With human intelligence (that is by myself instead of by computer), I selected features based on two simple criterions:

a) A feature is selected if it is significantly relative to driving type, and I myself can explain why;
b) A feature is not selected if either it is not relative to driving type, or it shares similar reason for being chosen with another feature but I myself can explain why it is not better than the counterpart.

Notice that, in order to protect privacy of the drivers' location, the trips were centered to start at the origin (0, 0), randomly rotated, and short lengths of trip data were removed from the start/end of the trip. That also limited my choice of features in some way, for example, direction between the origin and destination is no longer meaningful because of random rotation.

Finally, I chose these features (let each sampling point in raw data be $\boldsymbol{p_0}, \boldsymbol{p_1}, \boldsymbol{p_2}, \dots, \boldsymbol{p_n}$):

a) Linear distance between origin and destination: $l = \|\boldsymbol{p_n} - \boldsymbol{p_0}\| = \|\boldsymbol{p_n}\|$
b) Total distance the car has moved in a trip: $s = \sum_{i=1}^{n} \|\boldsymbol{p_i} - \boldsymbol{p_{i-1}}\|$
c) Average speed during a trip: $\bar{v} = \sum_{i=0}^{n-1} \|\boldsymbol{v_i}\| / n$
   where $\boldsymbol{v_i} \sim (\boldsymbol{p_{i+1}} - \boldsymbol{p_i})/1$ according to Lagrange's Theorem. Here $\boldsymbol{v_i}$ equals the instantaneous speed at the $i + 0.5$ second approximately.
d) Variance of accelerations per second: $VAR(a) = \sum_{i=1}^{n-1} (a_i - \bar{a})^2 / (n-1)$
   where $a_i \sim (\|\boldsymbol{v_i}\| - \|\boldsymbol{v_{i-1}}\|)/1 \sim (\|\boldsymbol{p_{i+1}} - \boldsymbol{p_i}\| - \|\boldsymbol{p_i} - \boldsymbol{p_{i-1}}\|)/2$ according to Lagrange's Theorem. Here $a_i$ is a scalar and equals the acceleration at the $i$ second approximately.
e) Histogram of norms of normalized directed accelerations: it is like HoG in computer vision. Each directed acceleration votes for its corresponding interval to get the histogram. And the histogram should be normalized so that to dismiss the effect caused by different *trip time* and different *travel direction*. Here, we let $\boldsymbol{a_i} \sim (\boldsymbol{v_i} - \boldsymbol{v_{i-1}})/1 \sim (\boldsymbol{p_{i+1}} - 2\boldsymbol{p_i} + \boldsymbol{p_{i-1}})/2$ where $\boldsymbol{a_i}$ is a 2-D vector.

It is worthy to notice that if I do not consider the travel direction when doing normalization in e), the new result can tell, instead of the driver's behavior when making turns, some geographic information of this trip. However, since the trip has been randomly rotated, the geographic information is meaningless and helpless.

I divided those features into 3 groups according to their relations between each other:

a) $l, s$: other than their own physical meanings, the relation between these two can somewhat reflect the kind of trip, for example, whether it is an one-way trip or round-trip ? Or, whether the trip is zigzag or direct?
b) $\bar{v}, VAR(a)$: though the relation between these two features, I can generally assume the trip or the driver to be:

| | small $\bar{v}$ | large $\bar{v}$ |
|---|---|---|
| small $VAR(a)$ | timid (may be a cautious novice) or normal road in city | normal highway |
| large $VAR(a)$ | reckless | racing |

c) HoA: it somewhat shows the driver's behavior when he/she makes turns after normalization according to total trip time and current travel direction. Different distributions present different behavior modes.

Each group of features can form a feature space, in which every trip will have its own position.

# 4. Visualization Design

After determining what kinds of features to use, now it's time to consider the way to visualize them. As selecting features, I set two simple criterions at the beginning:

a) The difference between the outlier and normal trips should be easy to see in the same feature space;

b) In order to help user make reasonable decision, the correlation of different feature spaces should be easy to see when user interacting with my tool.

According to these two criterions, three interactive images are created, each of which can make effect on other two when it is focused or clicked. And they are:

a) $l - s$ Scatterplot: each trip is simply presented by a dot and scattered on a scatterplot according to the tuple $(l, s)$. The color of every dot is the same as the color of every block in the heat map bellow. To make it more friendly and convenient, the scatterplot can be dragged or zoomed in and out using roller or simply double clicking. The axes are also dynamic, which means that they will change accordingly when the plot is dragged or zoomed.

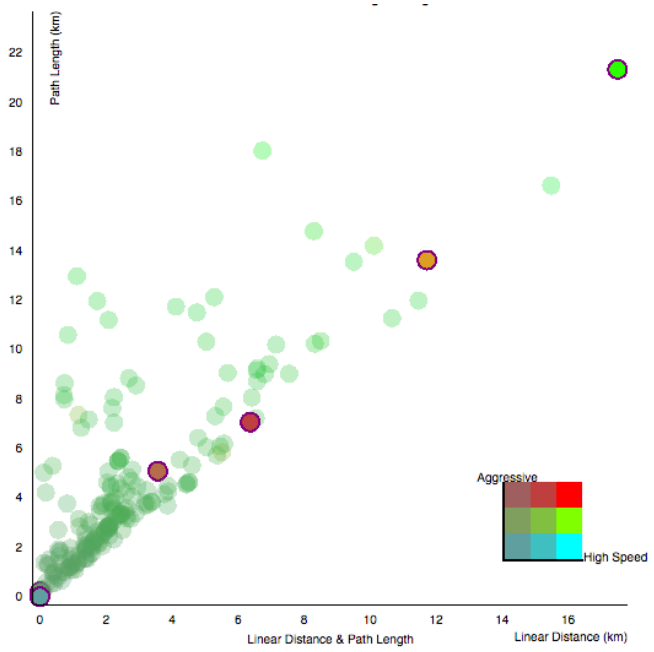


Figure 2: there are two dots selected in left corner, can you distinguish them?

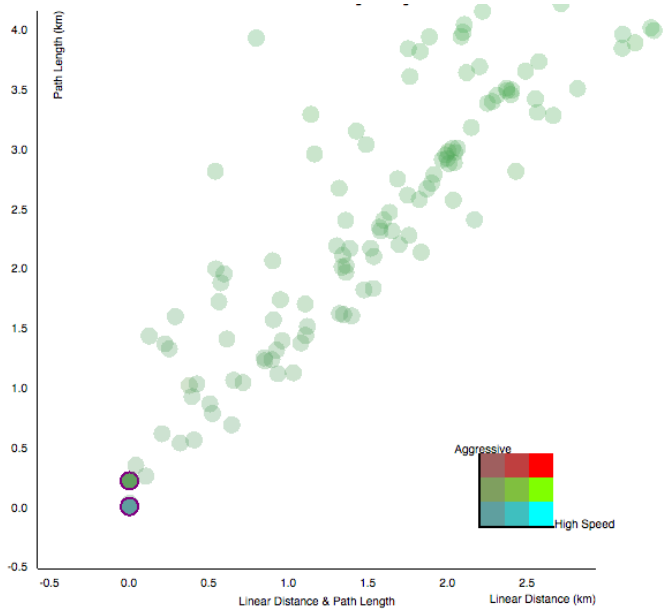


Figure 3: zoomed in and dragged

b) $\bar{v} - VAR(a)$ Heat Map: to visualize $\bar{v}$ as well as $VAR(a)$, I use heat map in HSL color space. Value $\bar{v}$ is encoded to saturation and $VAR(a)$ is encoded to hue. Light is set to 50% all the time.

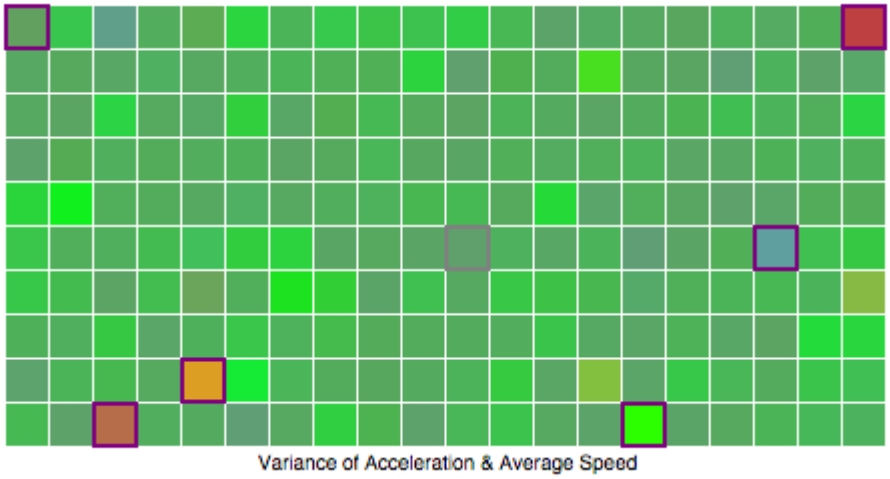


Figure 4: notice the border: purple denotes selected, gray denotes focused

c) HoA Rose Diagram: although the most common way to visualize it is drawing the histogram directly. Considering the x axis stands for the direction of vectors, I finally picked rose diagram. The number of intervals was set to 12, that is, each interval ranges 30 degree.

To make it nicer to see, lines representing unselected trips are translucent, that is, opacity = 0.3 which will turn to 1 when the trip is selected. It is the same with the dots in scatterplot.

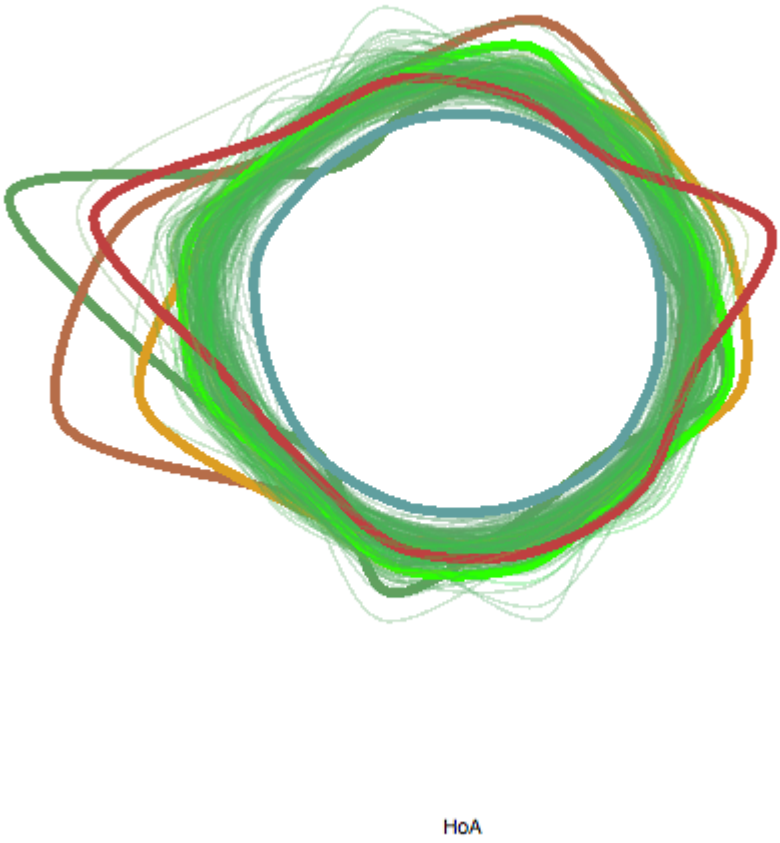


Figure 5: wider line means selected

# 5. Implementation

To implement MyWay, I chose D3.js. Lacking knowledge of web building and D3.js itself, I mainly referred examples and tutorials on its official website[3]. And also, I referred w3schools[4] for elementary information about HTML, CSS, and JavaScript.

To extracting features from raw data, I wrote a Matlab® script that can store features into a JSON file after reading all original record files in one folder and finishing features extraction.

All source code can be found in attachment, including 10 JSON files which are extracted from a part of competition data. Complete dataset can be find on Kaggle with a total size about 1.5GiB and contains hundreds of thousands records.

## 5.1. Feature Extraction

Using Matlab®, it is easy to computing features I need from raw data according to the equations mentioned. Normalizations and equalizations are also done in the

[3] http://d3js.org/
[4] http://www.w3schools.com/

script. Features from 200 trips of each driver are stored in one structure variable. And stored into a JSON file using the library Jsonlab provided by Qianqian Fang in Harvard.

As mentioned before, most feature extraction algorithm is apparent and easy to implement. One thing might be confusing is normalizing the acceleration vectors according to travel direction. That is applied to dismiss the effect caused by current travel direction. See the following figure:

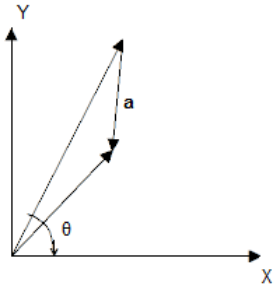


Figure 6

The vector $\boldsymbol{a}$ is the difference of two velocity vectors. Normalization is to rotate it $\theta$ degree, that is, $\boldsymbol{a}_{\boldsymbol{normalized}} = \boldsymbol{a} \times \begin{bmatrix} cos\,\theta & sin\,\theta \\ -sin\,\theta & cos\,\theta \end{bmatrix}$ where $\theta$ is easy to get.

And in practice, I use standard deviation of accelerations instead of its variance to decrease the disparity of data, otherwise, the visualization will become too extreme since the distribution of variance tends to be like this:

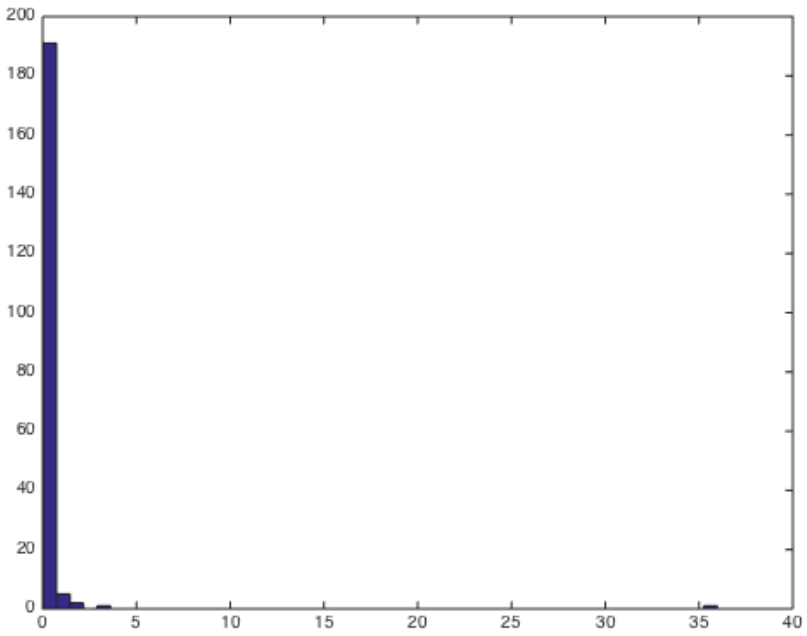


Figure 7: data from 1001, notice the bar near 35

The distribution is extremely skew with extreme abnormal maximum (minimum). It makes equalization perform very poor.

Notice that even standard deviation is not good enough:

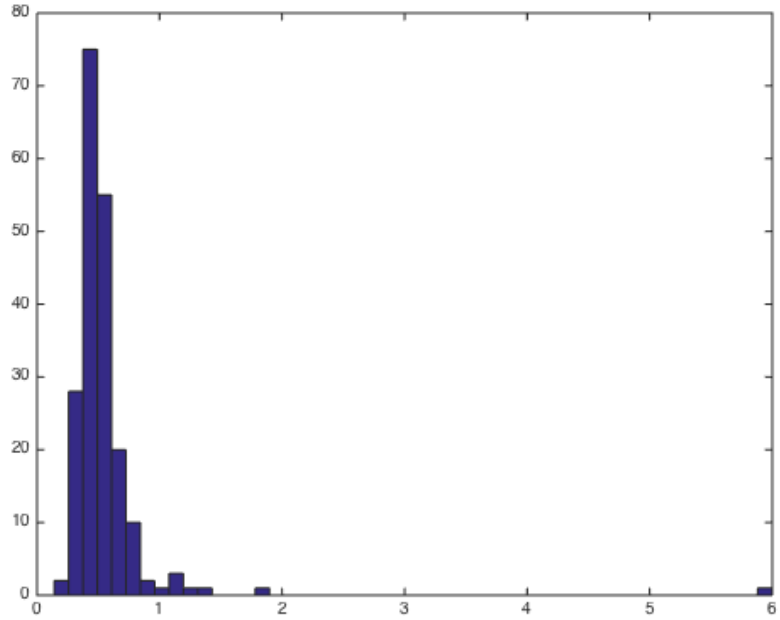


Figure 8: same data with Figure 7, notice the bar near 6

A trick is used during equalization: the largest two values and the smallest two values are equalized to 0, 20 and 160, 180 respectively while the left are equalized to 40~140 using normal method.

Considering the total number of trips of one driver is not very large, the script is not highly vectorized to gain more readability and to avoid unnecessary debugging cost.

## 5.2. Visualization Website

Building the visualization tool using D3.js is tough for me since I know little about HTML, not to mention CSS or JavaScript. Although my design concepts were implemented finally, the code may not be neat with high efficiency. Here is a very brief description:

a) To read JSON files, d3.json() is called, the data will be stored as arrays in JavaScript, which is like list in Python, so that manipulating it is not too hard.

b) The heat map is the first chart to be implemented because it is the easiest one. I did it by following steps: Add enough rectangles at first, with which data is bound. Then write a function that can receive a trip's information and return its corresponding color. After that, fill all rectangles by calling the color function respective. Finally, add title and other attachments.

c) The next chart to be completed is the scatterplot. Through drawing circle on proper position and adding axes, it is not hard to get a static scatterplot. Taking use of d3.behaviour.zoom and writing some coordinates transforming code with d3.scale, it became zoomable. Adding a transparent rectangle with the same size as the SVG of scatterplot could make zooming more easy to use.

d) Using some knowledge about the polar coordinate, we can compute the way to

draw rose diagram. We can bind these methods to a d3.svg.line() and set its interpolate "cardinal-close". Then it could help drawing lines on the path I appended.

e) Add more subsidiary functions like using a HTML5 label textarea to show the trips user has selected. Add titles and author's name. Finally adjust the layout.

# 6. Conclusion

A visualization tool for anormal driving style detection has been implemented. It is easy to use and somewhat friendly to users. And also, it can indeed make some outliers very easy to see and to be marked among extensive dataset.

For the lack of experience in web building, the implementation may not be neat and efficient. Bugs are still possible to remain. And the design concepts are not very complex. Shortcuts also include the color scheme, if more time was offered, I may try to refer website like colorbrew, and use a professional color scheme.

I have to admit that there is deficiency with my work. However, the more important thing is that, through working on it, I understood the complete process of solving a task of information visualization. I performed it by myself completely: finding problem, analyzing it, making proposal and plan, building models and thinking mathematically, choosing proper visualization charts to display proper features, designing interaction, patching required knowledge in a very short time, combining knowledge I had learned to implement it… And finally, I made it! It's really fascinating and amazing to see it truly work! I am really proud of myself moving so far.

# Reference

Bostock, M. (2015). Retrieved 2015, from D3: Data-Driven Documents: http://d3js.org/

Card, S. K. (1999). Readings in Informa4on Visualization: Using Vision to Think.

Haber, R. B. (1990). Visualization idioms: A conceptual model for scientific visualization systems.

Kaggle Inc. (2015). Retrieved 2015, from kaggle: http://www.kaggle.com/

Refsnes Data. (2015). Retrieved 2015, from w3schools: http://www.w3schools.com/