CS573 Data Visualizations Final Project Process Book

-Yifei Gao, Jingjun Zhang, Hongmei Zong

Overview and Motivation:

Everybody want a dream car. However, buying a car is a big issue for most people. It is very hard for customer to choose the car most suitable for them. There are three basic reasons why it is so difficult to choose a car. Firstly, there are hundreds of models of car sold in the market, so customers have so many choices. Secondly, a model of car has so many specs such as price, MPG, Horsepower, drive type etc., thus it is difficult to compare. Finally, a lot of customers are not professional about the cars, so they just want a type of a car similar to their friends' or they see in a movie. While it is not easy to classify the cars which are similar to the car they love.

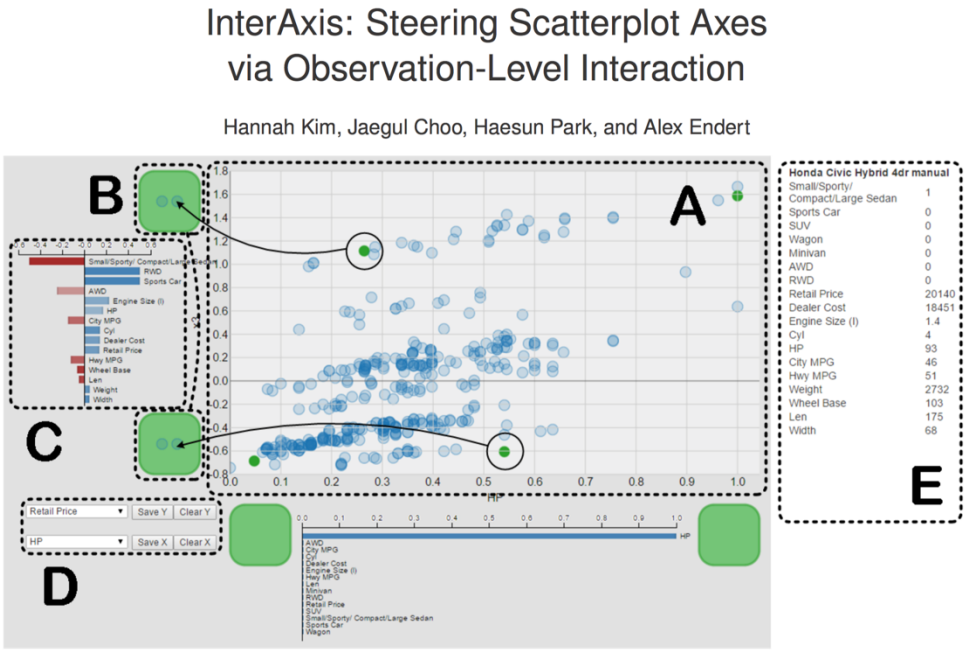
In this project, we are going to build a data visualization for different car models and help users to find the car they want the most. Through the data visualization, we could solve those three problems mentioned above. We are supposed to represent all cars models by scatter plot. Scatter plotting the car data set is much more clear than checking hundreds of data entry of car models. Also, we are going to use InterAxis technique to implement the dimension reduction. In this way we could use a one-dimension axis to represent multiple dimension specs of the car.

Thus, we would implement an inter axis plot as the main visualization for this project. User could select the car they are interested in, adjust the weight of features of a car and finally find several cars they might be also interested in.

In addition, we are supposed to apply some other visualization to assist the inter axis chart. First, we need a geo map to assist the user to filter the cars by country. It would narrow down the search range and makes the searching easier. Second, a radar chart would be shown when the user selects some specific points, which corresponding to specific models, in the InterAxis chart. A radar chart would represent more detail of one model, especially when compared two different models. Third, we add a bipartite chart into the project to help user figure out which feature is concerned with which brand. This chart

would give users an overview of the relationship between specs and manufacture, which would help user configuring the weight of specs in InterAxis chart.

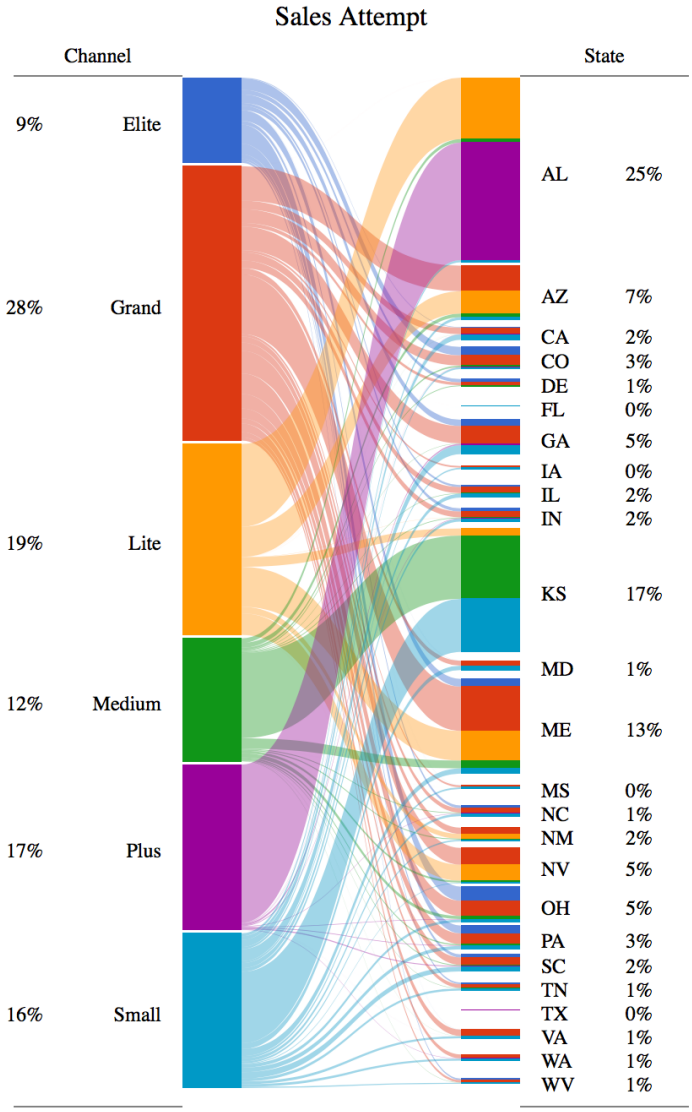
Related Work:



A paper named ‘InterAxis: Steering Scatterplot Axes via Observation-Level Interaction’, which recommended by professor for the research lab, really inspired us. We found InterAxis scatter plot is very suitable for our project for the following reasons. First, it could handle multiple dimension data, which means it could represent many features of car model in only a simple scatter plot. Second, it could classify and cluster the similar data, which is what we proposed to do to help user find their interested car. Last, it is a new technique and has not been applied too much. We are willing to challenge ourselves to implement this new technique.

From the data set, there are many records, but it is very hard to find out the trend between different car bands and the features they provided. If we can visualize different car brands with some user concerned car features, that will be very helpful for users to narrow down to a couple of car brands. We browsed the d3 examples and found out an interesting d3 visualization called double vertical bipartite chart. This visualization can serve for this purpose.

We believe it can be adopted to our design to illustrate the overall ratio information of different features vs. the car models from different manufactures. Here is the link to the visualization. http://bl.ocks.org/NPashaP/cd80ab54c52f80c4d84cad0ba9da72c2. The graph is also shown in the following.



Questions trying to answer:

The overall goal is to help users to find out the right car to buy based on the technical point of view. From the visualization, we also allow users to explore some features that some particular manufactures provide. So they will get some general ideas about key features among different car models and car

brands. Here listed four questions that our visualization will help users to answer.

1. *Which cars are similar to my interested car and which cars are not.*
   This question can be answered from the scatter plot implement inter axis functions. It clustered the car models to similar cluster of your preferred.
2. *Which manufacture has some specific feature I care most (for example, AWD).*
   Through double vertical bipartite chart, users can select a feature that they care and visualization will give users the percentage information of different types of this feature vs. different manufactures.
3. *What are advantages and disadvantages of one model over the other models.*
   Radar chart visualization will clearly show the values of many key features of selected car models. So it is very easy for users to see the advantage and disadvantage of a car model vs. other car models.
4. How could I classify different cluster of models by my preference.
   Bars located on the left of InterAxis chart are best indicator of user preference on cars. To separate cars into different clusters, what we should do is simply maximize the bar of corresponding bar and minimize others. You will the clusters after the chart is refreshed.

Data source:

There are plenty datasets regarding cars in the internet and it is not so difficult to find one. However, most of them are not adequate for this project because the data are neither up to date nor complete. Finally, we decide to use the data fetched from a API provided by www.carqueryapi.com. Car Query API provides specs data for nearly every model in recent years. The API is JSON based and it is easy to retrieving data by writing a JavaScript.

In JavaScript, we mainly use the statement below to execute the API accessing. $.getJSON("http://www.carqueryapi.com/api/0.3/"+"?callback=?", {cmd:"getModel", model: modelID}, function(data) {};

By this statement, we could get the data of each model by the car model ID. However, the data entries are too many to retrieve. There are 65000 entries in total. Another bad news is that API have a time interval limitation for retrieving data. If we access API shorter than the limitation time interval, an access denied error would occurs. Thus we have to set a time interval in JavaScript. In this way, three of our team members collected the whole data set in almost one week using three computers.

Data processing and clean up:

The dataset gathered by CarQuery API is not very complete or consistent. Here are some major problems in this raw dataset, and we have different solutions for them:

a. Massive missing of values for some features. There are usually two simple methods to deal with missing data, calculating new average value if possible, or discard the data if you have a very large dataset. But this time, we are lucky enough to find that features that have massive missing data are not very interesting data for common customers, like engine compression. So we decided to remove such kind of features, then we will remove data points with missing features. In this way we can reserved as many useful data as possible.

b. Unit of fuel economy. This dataset use liter per 100 kilometers as unit of fuel economy. Instead of modifying it in raw data, we decide to convert it to MPG when we are reading data. MPG is widely used in America.

c. Inconsistency of nominal values for same category. For example, for 4 wheel drive cars, some are labeled as “4wd”, some are “four wheel drive”. To deal with this situation, we decided to match keywords of categories to those value then re-label them with new unified value.

d. Lacking of some features we care about. We did not find original country of car models in this data set, we have to use a script to automatically generate this feature. Users may need this information when they choose to buy a car.

e. Outdated data points. This dataset has data of cars since 1930's, some of which are no longer interest to common customers. Also, those car models are discontinued. So we decide to remove all the cars that had been made for more than 5 years.

f. Duplicated data points. There are repeated data points found. Some of them are only the difference of capital letters or lower case letters of car brands or models. We just simply removed them.

g. Over-classification of some features. Trim of car has been categorized into about twenty different sets, but it is not necessary and will introduce extra workload of data processing. We simplified such kind of features into at most 5-6 categories.

Design and generation of data structure for visualization:

After processing of raw data, we have about 1000 data entries which are reprinted as array of objects. However, this cannot satisfy the requirement of our major visualizations. So we customized a data structure for interAxis plot which reserved most features of data in a list as an attribute for visualization and calculation, plus coordinate of data point on X axis and Y axis, which will be recalculated as long as weight of any feature is changed. The original id is reserved as unique identity for a data point while evolution of visualization.

```
var dataUnit = function(id, name, country) {
  this.id = id;
  this.name = name;
  this.country = country;
  this.x = 0;
  this.y = 0;
  this.data = [];
};
```

Raw data will be read entry by entry and for each entry, we will save numbers directly into a new dataUnit structure created for this entry. But for categorical

data, we have to create more features for one original feature, for example, we split model body into six features with value range of only 1 for yes and 0 for no, and they are SUV, sedan, truck, wagon, minivan and sports. Other features have similar procedures.

Once the raw data processing is completed, they are stored into a list of dataUnit at once. And this will be the major data source of our visualizations. None of them will be changed except x and y attribute of all data points.

For double vertical bipartite visualization, the data format is “A”, “B”, #. The number is a summarized number. For example, “Ford”, “AWD”,35. This row data means there are 35 ford car models counted in the processed data set with AWD feature. We use python to pre-process the processed data set and generate new data sets to feed the double vertical bipartite visualization. From the car csv file, it groups by brand name and the other feature, such as “drive type” then count the number of entries. Finally, we got the data for “drive type”, “body type”, “fuel type” and “year” of all car brands and the total count for each type. All these data sets are saved as variables in JS as input of double vertical bipartite visualization function.

Exploratory Data Analysis:

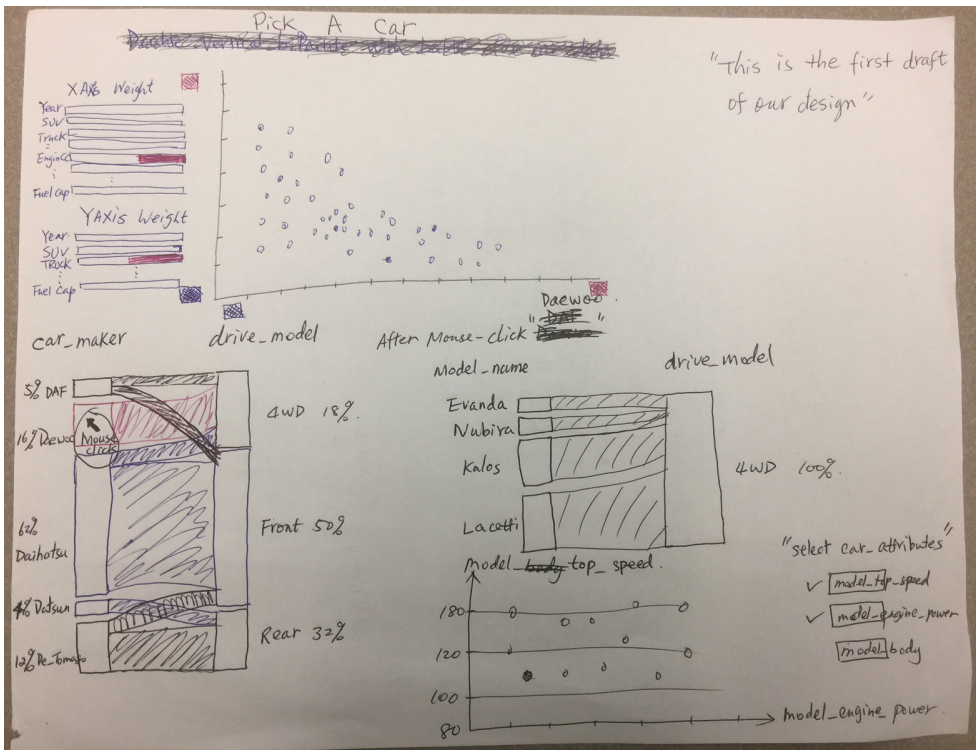
After our data is cleaned up, there are approximately 1000 entries and each entry includes 22 fields. We would like to show not only these instances but also some key features interested by users. Since the whole data set has too many instances to be shown in one chart, we need to let user to filter the car models they are interested and compare with each other. Based on these analysis, several visualizations came out to serve those objectives.

Since we did an assignment for plotting a small car dataset, it is very common to use scatter plot to visualize the car data set. There are around 1000 records in car data set, it may be too many dots to see clearly. We have to find a way to filter the data and narrow down selected car models. So geo map came to our mind. It can display all these car models based on countries. When users interest American car brands, they can click USA in the map and all car models are filtered as American cars. When users focus on several car models, we need to think a way to show more features of the car. Then radar

chart is a solution to do the work. Base on the data set, it is nice to summarize some patterns or trends for different car brands. For example, which brands make more AWD cars, or which car maker make electrical cars, etc. All the summarized information will help user to choose the car models they like. There are many insights we gained from the project. Multiple-views coordinate with each other, allowing users to interactive with visualizations, etc.

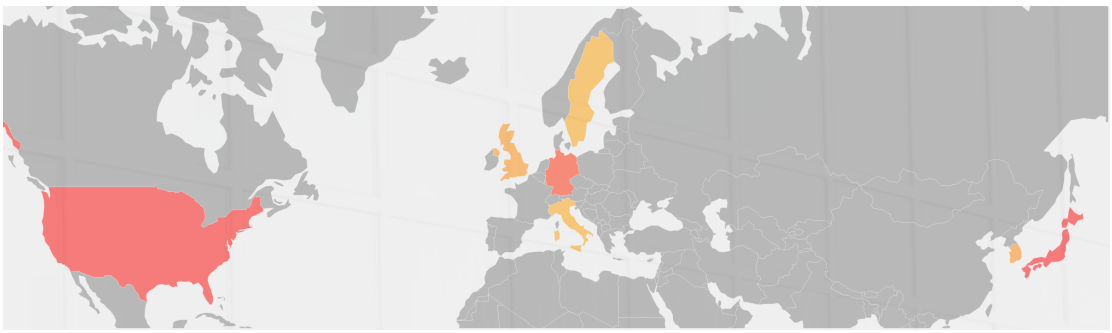
Design Evolution:

Several different visualizations have been considered in this project. Geomap, scatterplot, double vertical bipartite chart and radar chart, image buttons, and so on. The following screen shot shows first version of our visualization design. There is one scatter plot with inter Axis implemented and one bP chart implemented mouse over functions. In the first version, we would like to let user to select some attributes to do another scatter plot. Then later on, we use radar chart to do the comparison of features of different car models. Since radar chart can show more dimension of a car model.

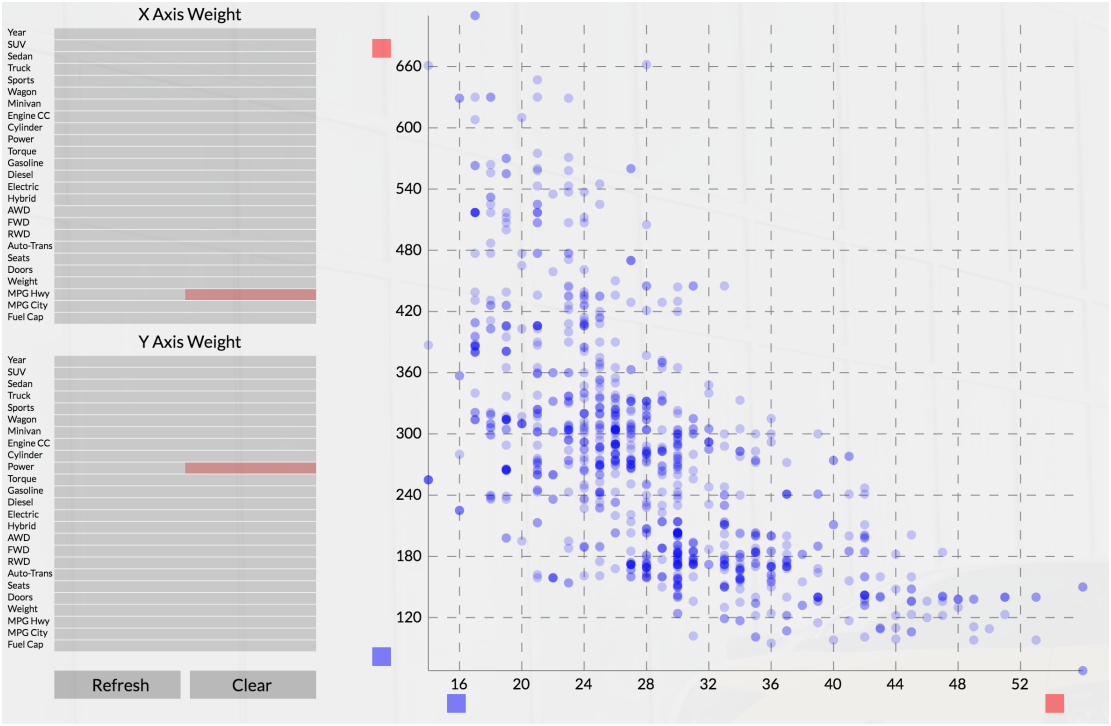


Geo map displays the map of the world. It is a good way to display the data set with geography information in it, since the car brands are from different countries. All the car makers from the same country can be shown in the

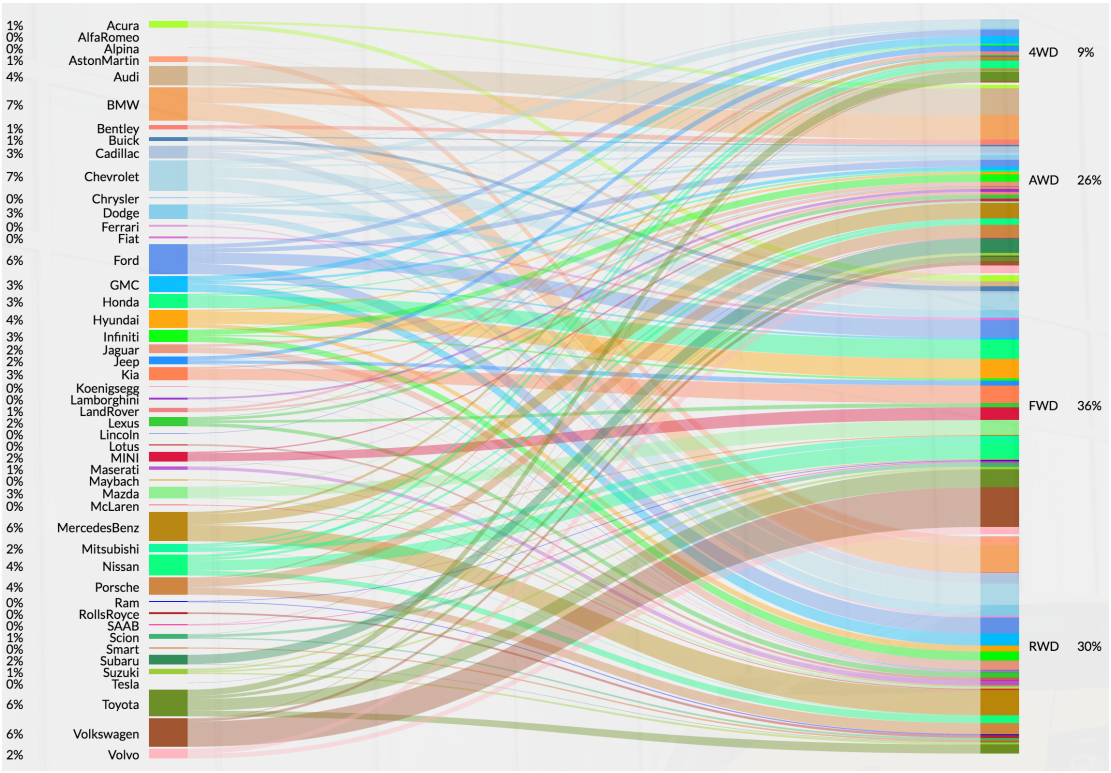
same country in the geo map. The background color of map is grey. The countries that make cars will display colors as yellow colors and red colors. The darker the color, the more car models made by that country. In this point we use color palette from light to dark which indicates car models volume. When mouse over the country, it displays the car brands produced by that country. It allows user to interactive with the dataset. Here is our prototype version of our geo map.



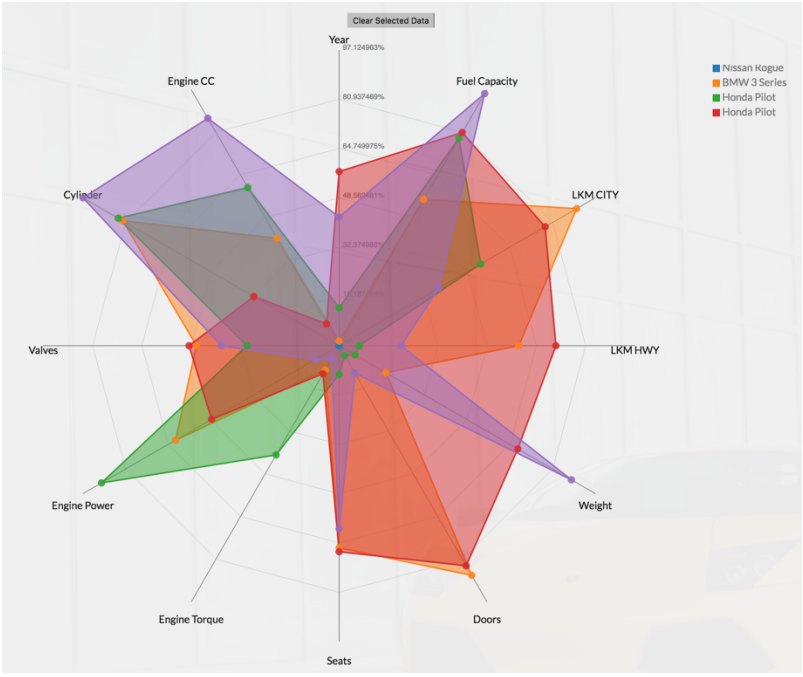
Scatter plot is good for display a small to medium size data entries. The scatter plot shows the whole data set or filtered data set and it coordinates with geo map, the idea is from our assignment. Our scatter plot visualization implements the interAxis functions. Users can define their preference by click the instance of car model in the plot and choose high end by clicking the red square, low end by clicking the blue square. Then the scatter plot can be refreshed and clustered. But the interAxis weight bars are not adjustable at that prototype version. The screen shot below is the first version of implementation of InterAxis scatter plot.



The double vertical bipartite chart shows the relationship between A group and B group, in our case, it is brand name and feature types. The below screen shot is a prototype version of bP chart. We choose different colors for brand name (category data) and use the same color group for brands of each country. Since there are more car brands in Japan and USA. We use blue colors for USA's car brands and green colors for Japan's car brands. Other countries choose red, brown, orange and pink colors. It has the mouse over functions implemented. When mouse over, visualization will change. Here is a screen shot of prototype bP chart.

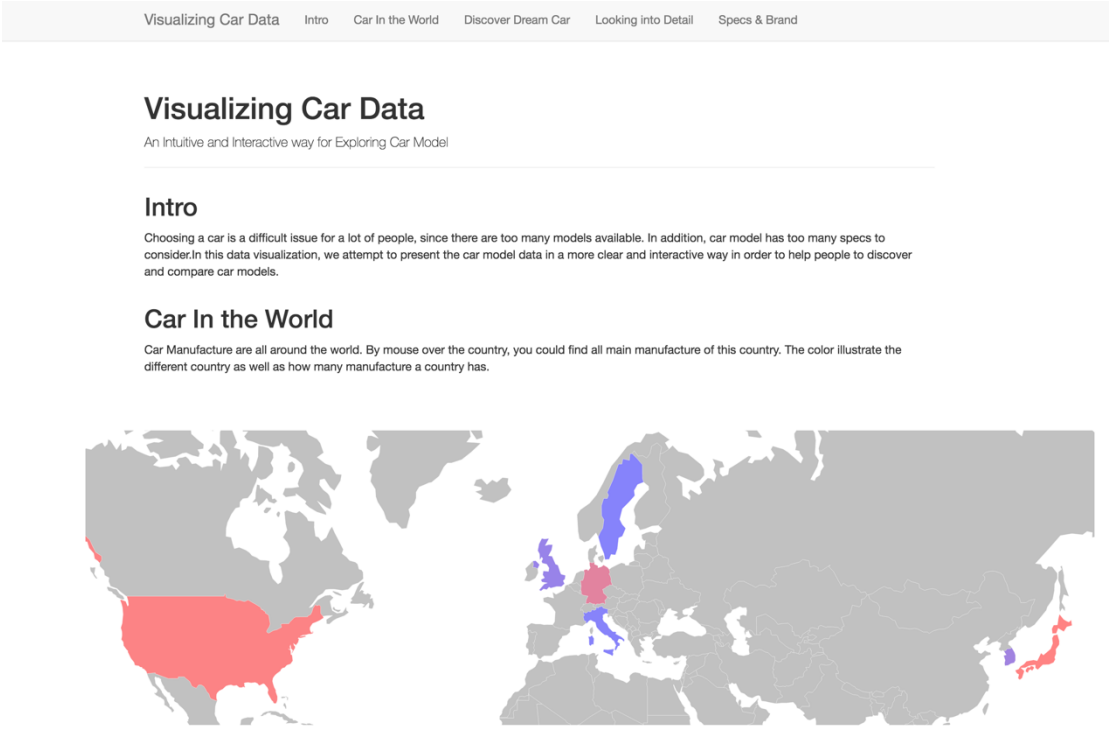


After filtered by geo map and InterAxis scatter plot, when clicking the points in the scatter plot, the radar chart will update for these instances. Here is the prototype version of radar chart. Since all these attributes are not normalized, the range is very large, so it need to be improved in our final implementation.



Implementation details:

The page in the following is our first page of our project website, it provides five buttons on the top for user to go directly to different visualizations. In this page, the geo map shows the countries that make cars with bright colors. We adjust the color palette from prototype to the final implementation. Red colors represent there are more car models made in this country, whereas blue colors mean there are less car models in that country. In the interAxis scatter plot, the colors keep the same. The car brands name is displayed when mouse over a country. By clicking the country, the scatter plot will display all the instances of car models for the selection.



InterAxis Scatter Plot

The InterAxis scatter plot is our main data visualization. Here is the summarization of functionalities of InterAxis Plot:

a. Interaction with geomap. Event of mouse over or mouse click can trigger the display of data points related to specific counties.

b. Auto adjustment of X and Y axis. After recalculation of axes, there could be both positive number and negative number on the axes. We introduced

automatic adjustment so the axes will be placed on where 0 appears unless 0 is not on the plot.

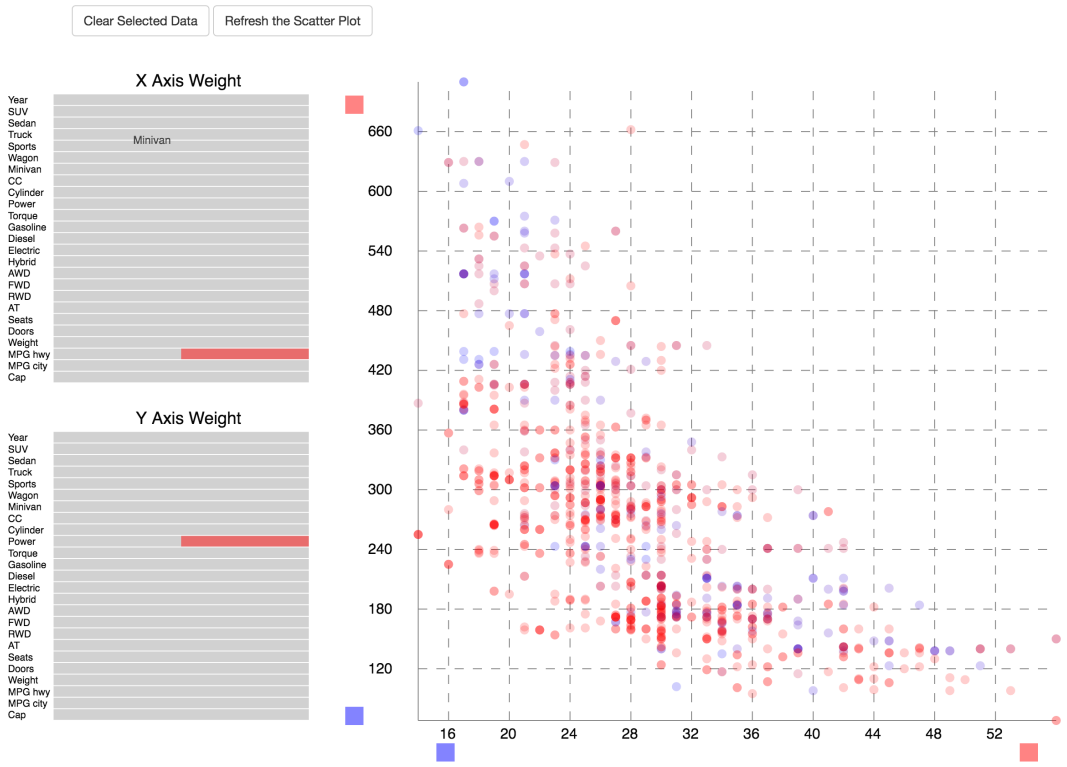
c. Auto adjustment of scale of axes. After recalculation of axes, the magnitude of data could change drastically and we will automatically adjust start value and interval of ticks to facilitate better understanding of data.

d. Colored data points according original country. We use different color that is consistent with color design in geomap to represent data point on the plot.

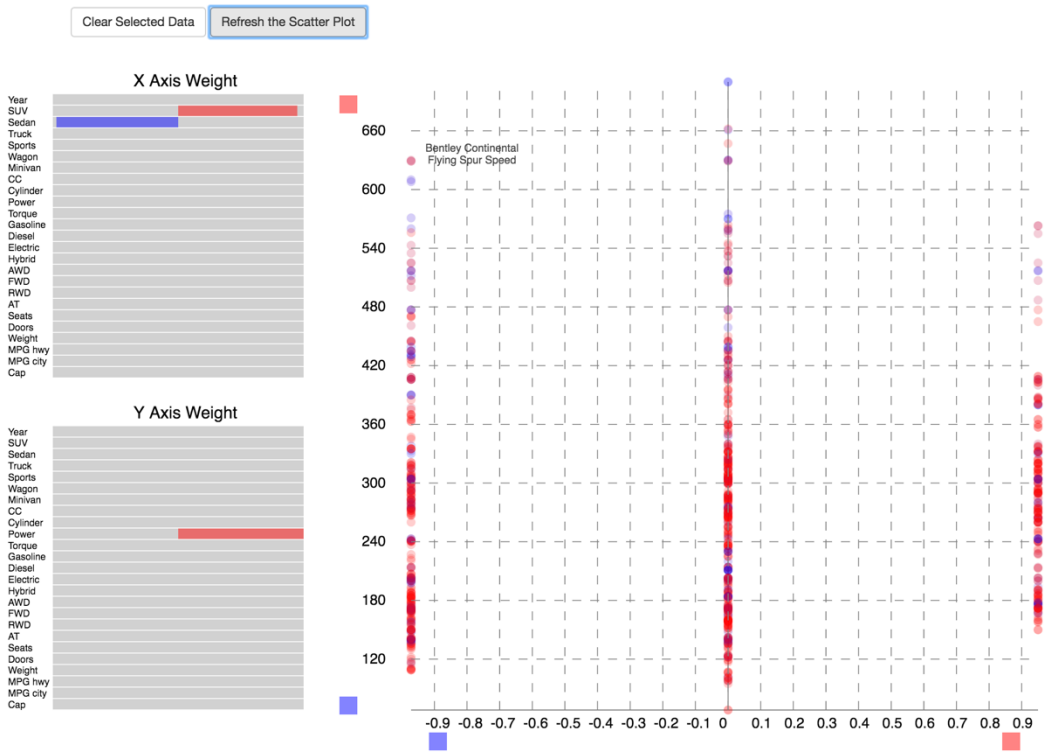
e. Select data points into any ends to recalculate axes. Any data points in the plot can be put into an end of axes and used to change the axes.

f. Adjust bar size to change the weight of a specific attribute. InterAxis also allow change of bar length to control the weight of X or Y axis. This feature can be used as selector to cluster data points according user preference.

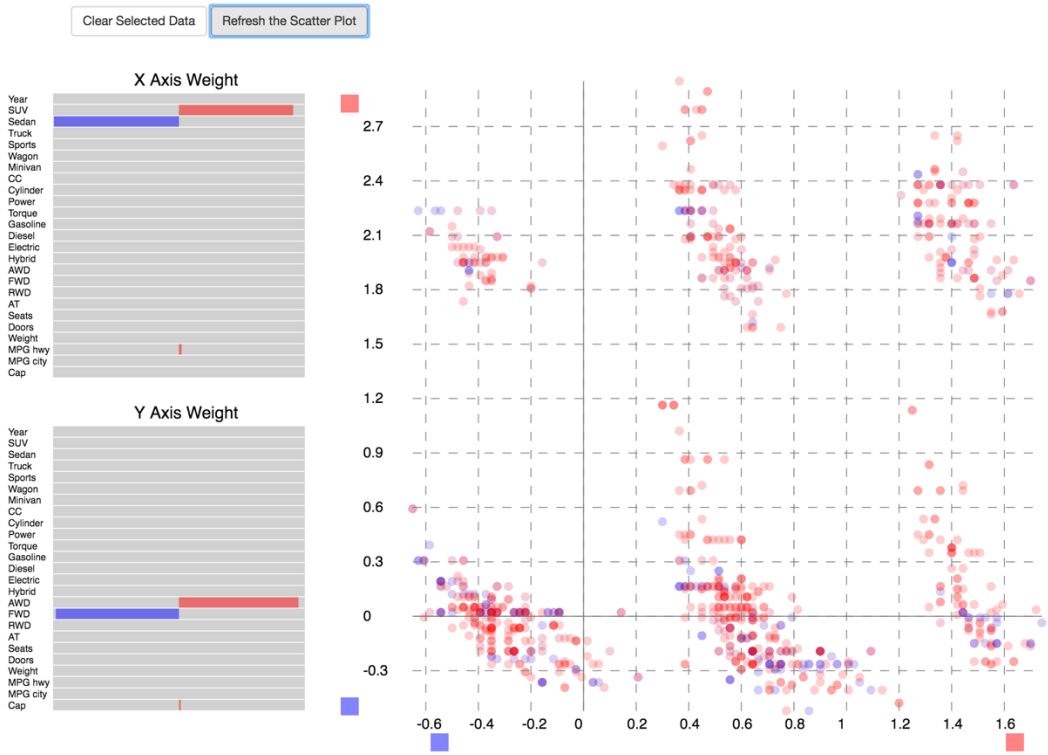
Feature ‘a’ to ‘d’ are all new features that the original interAxis does not have. The only thing that the original interAxis design have and we don’t have is the drag and drop operation, if we have enough time, we could implement it.



The initial state of InterAxis reflects relation between MGP(highway) and engine power features, the car models are from USA.



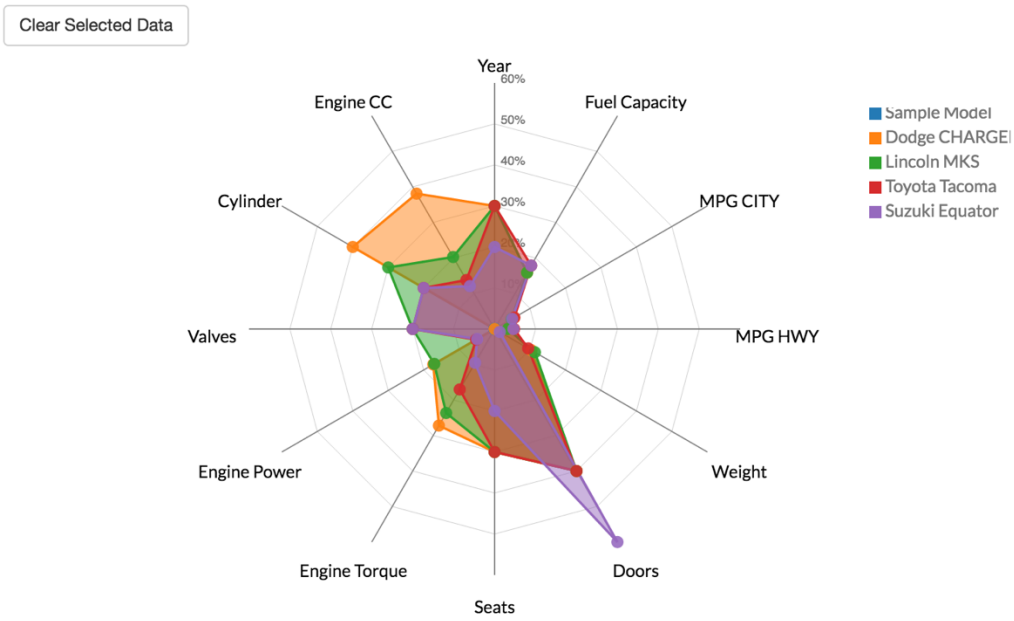
After we select a positive weight for SUVs and a negative weight for sedan on X axis, then select engine power on Y axis, we can see all data points are separated into three lines representing three categories, SUV, sedan and others, ordered by engine power.



We can also select a positive weight for SUVs and negative weight for sedan on X axis, then select a positive weight for AWD cars, negative weight for FWD cars, we can see all data points are clustered into six groups. There is one fact we should notice, that is position of data points on InterAxis is based on weighted average value of all features, which means some feature like weight could have dominate influence on the position, so we need to make sure we set the weight of most of these features to zero. On the other hand, we sometimes need features with wider value domain to scatter the points, otherwise points could overlap each other on six positions.

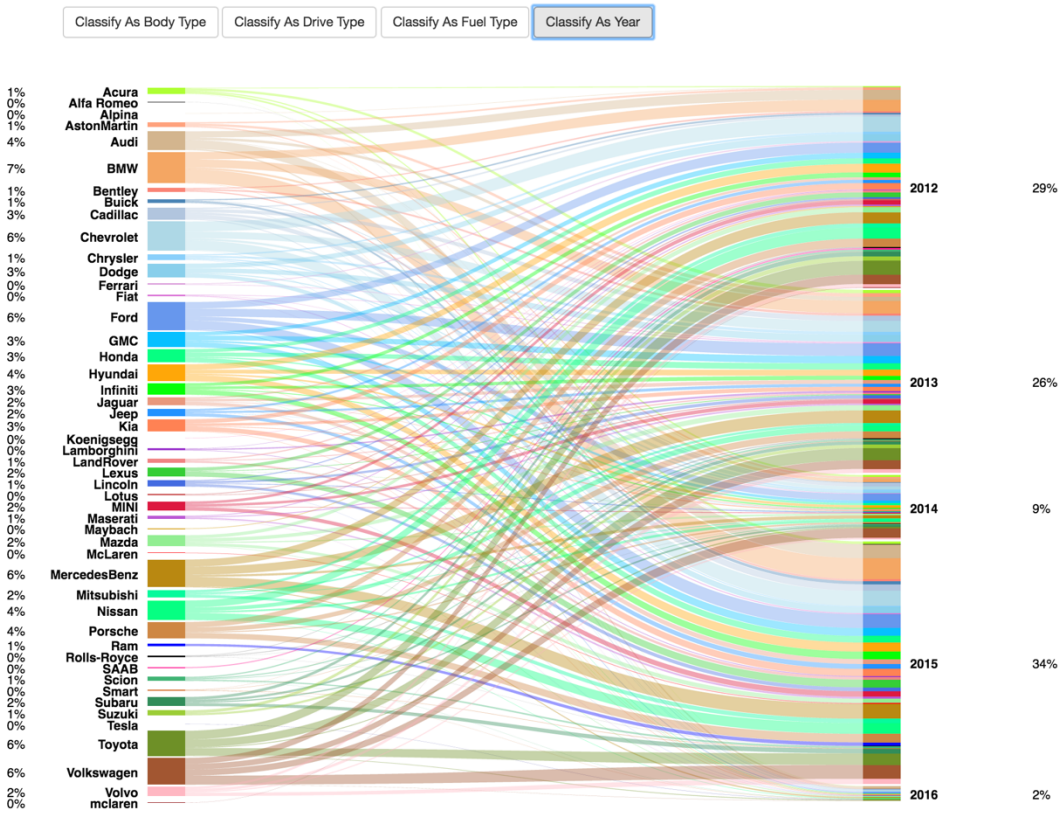
Radar Plot:

When selection made from the scatter plot, the radar chart shows the instances of the car models. And the data values have been normalized, so the visualization is much more smooth that the prototype radar plot. The colors are chosen from the website for category data set. The screen shot of radar plot is shown below.

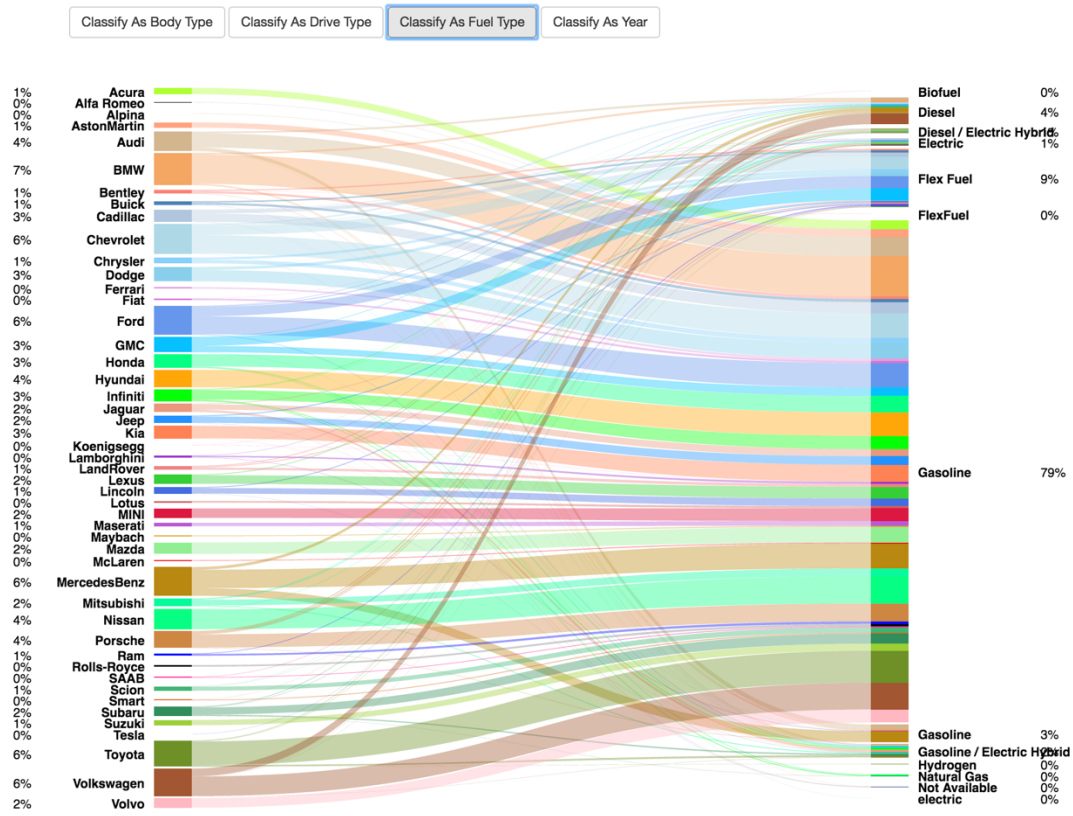


Double vertical bP plot:

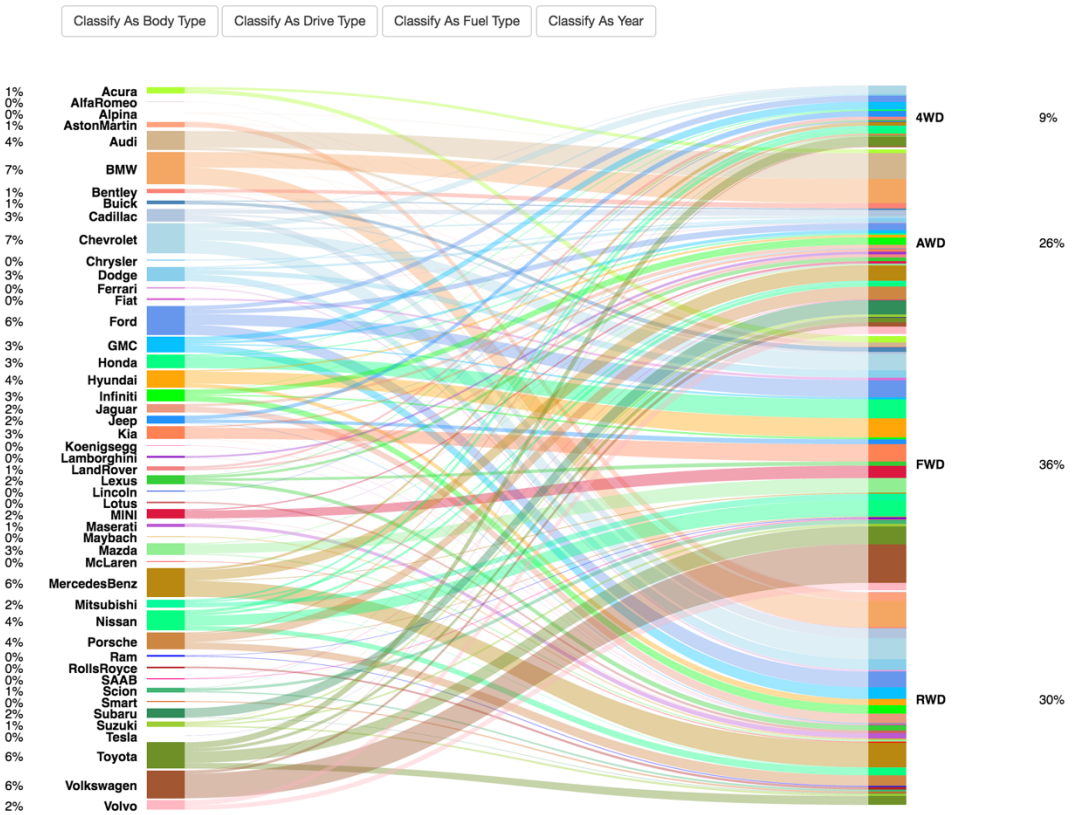
There are four types of features user can choose from by clicking the buttons. When click “classify as year”. It plots the year information vs. different car models for our pre-processed dataset. We use the same color design schema as prototype. Here is the car brands with year bP plot.



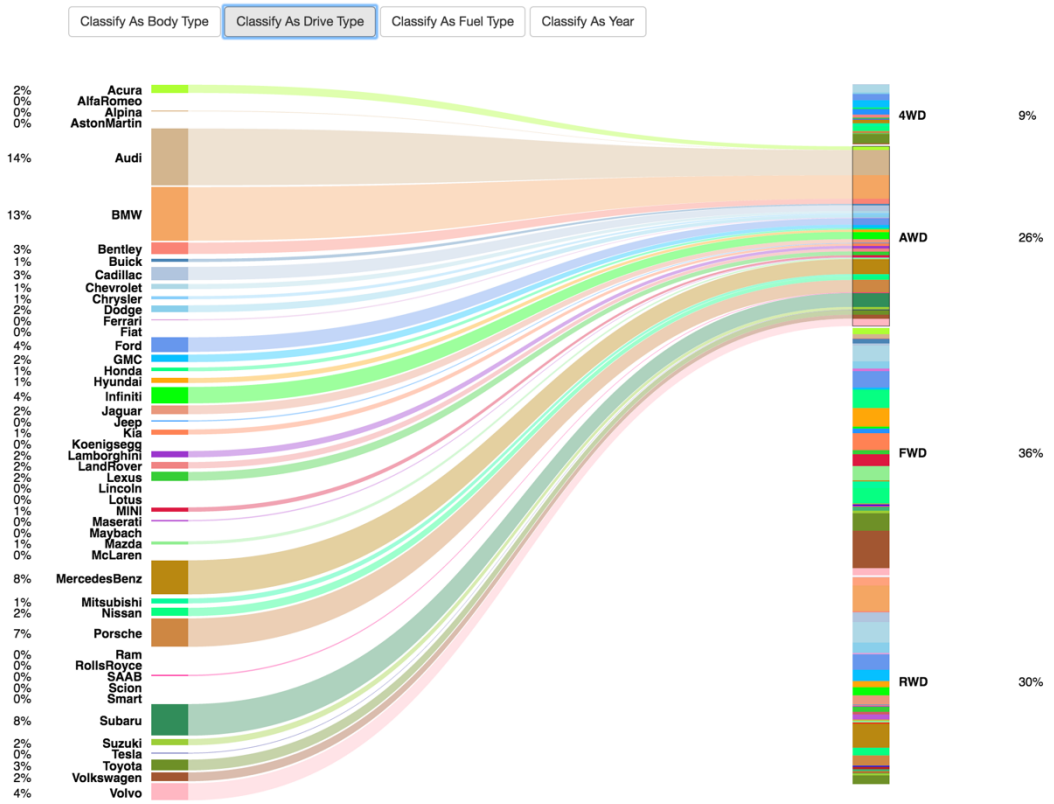
The below is the car brands with fuel type bP plot.



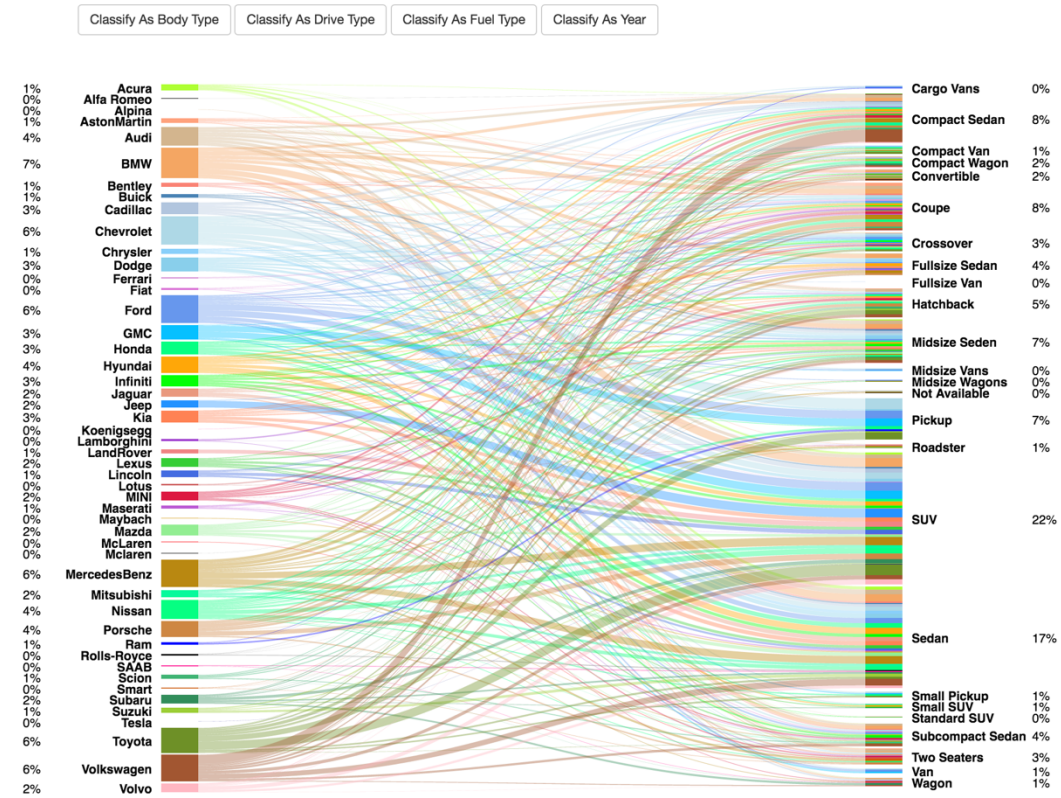
Next screen shot is the drive type vs. car brands.



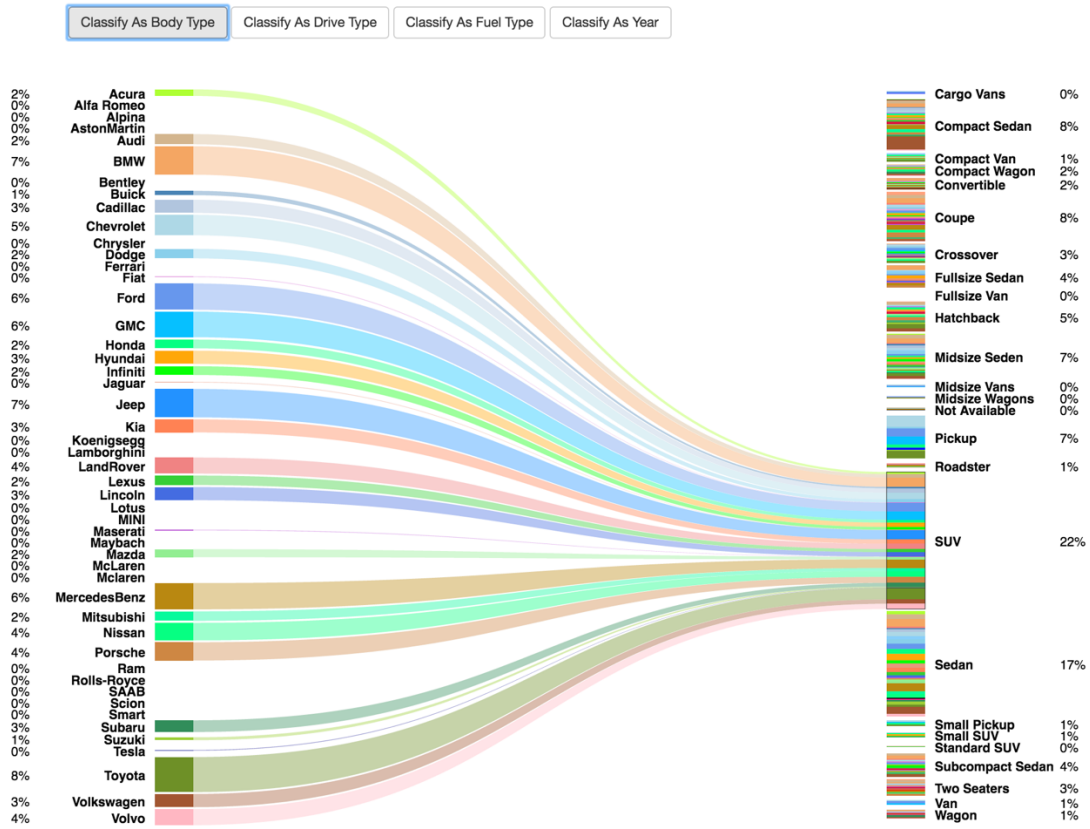
When “AWD” is selected, it shows the distribution of car brands for AWD feature.



This is the car brands vs. body type bP plot.



Next bP Vis shows the SUV vs. different car brands.



Evaluation:

Out implementation of visualization reflects the underlaying user preference that usually is buried in fancy advertisements and it helps users to compare all cars they are interested in.

In the near future, we plan to modify the way to calculation coordinate by standardizing value of feature. This will be a modification to the original algorithm specified by InterAxis paper, and it will allow us to balance the influence of features with higher value. We will provide an option to user to use original method or standardization method to calculate coordinates.