**Philips Hackabout 2018**

Challenge - 1: Mobile price category classification challenge

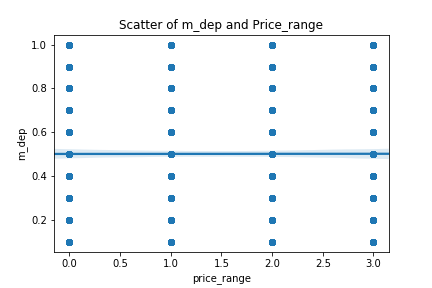
**About the challenge:**

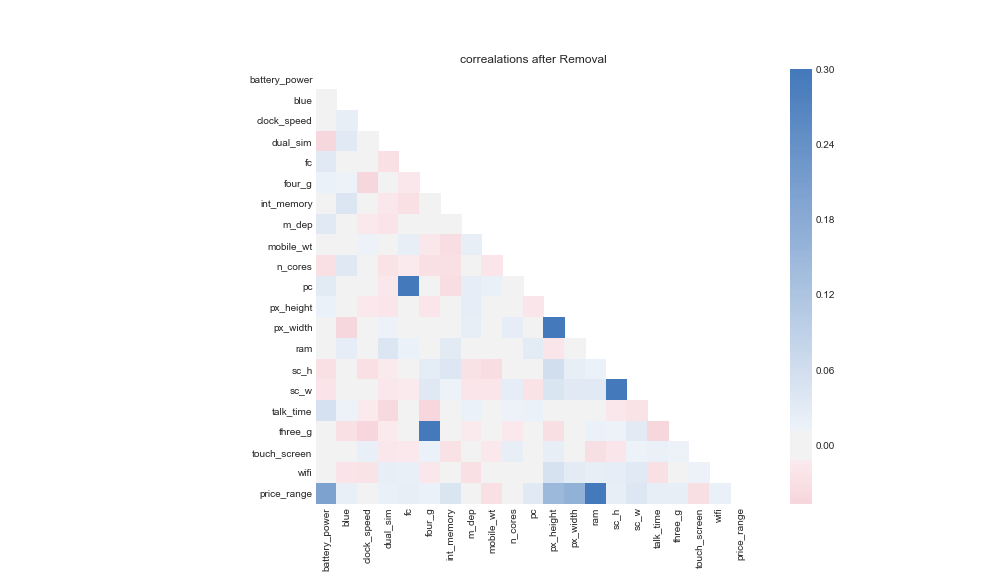
We were asked to make a model to predict one out of the three the categories of the price a mobile lies in.

The variables consisted of details of the mobile including its internal memory, ram, physical details like dimensions, weight and features like 3G, 4G, wifi etc.

The final output gave us a result of **96.3%** accuracy using the **SVM classifier** model using a **linear kernel.**

**Exploratory Analysis Steps:**

* In this step we did same basic analytics to get to know the dataset.
  + It showed us that in the 20 explanatory variables and another response variable, none of the values had any missing values in both the datasets.
  + We identified the categorical, numerical and boolean variables in the dataset.
* Then we proceed to univariate analysis where we thoroughly went through each data column.
  + We constructed box plots to identify the outliers
  + Looked at their mean and standard deviation values etc.
* In the next step we did a detailed bivariate analysis for each of the variables with the price\_range and among themselves.
  + Pairplot helped us identify 3 columns whose existence did not affect the price range at all.
  + We confirmed this visually by plotting graphs.
  + Correlations without the uncorrelated columns.



The faint correlations with the battery\_power, px\_height and width and ram become more clearer now. Also, the negative correlations with mobile\_wt is more prominent.

Another parallel data set with principal component analysis selecting components from 10-20 were also created.

**The models trained for predictions:**

1. SVM classifier with PCA gave accuracy 91% to 95% for number of components from 10 to 20.

2. Applying SVM directly gave an accuracy of 96.3% using the linear kernel and c value of 1.(highest)

3. Random forest algorithm gave us an accuracy of 88%

4. Neural network with 2 hidden layers with 2000 neurons each with a dropout layer to reduce overfitting gave us an accuracy of 95.8%