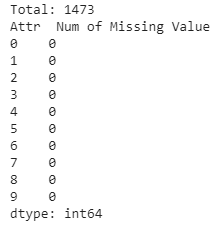
.

**DATA SCIENCE CLASSIFICATION**

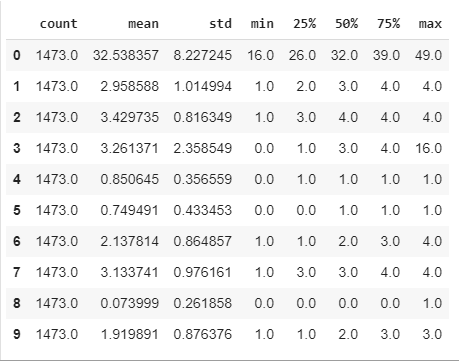
*Group 4*

* *Dataset*
* *Data source: Our group select the Contraceptive Method Choice Data Set (*<https://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice>).

* *Data quality and preprocessing:*
* *Check for missing value: Our result showed that there is no missing value in the data set. Therefore, we can move to next step.*



* *Print and Descriptive Statistics Table: Our purpose is to have an overview of our dataset*



* *Checking Correlation: Based on the descriptive information form the statistic table above, we can see the difference in range of values in each feature, especially for the first feature. Hence, we might need the normalization for this data. Next, we need to check the correlation of features by plotting the correlation matrix. We obtained a correlation heat map as below:*

*From the correlation matrix above, it shows clearly that there is not any pair feature which are highly correlated together. So, we can keep all features.*

*Next, we would like to see how the feature's distribution in the followed figures.*

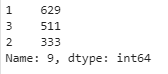
* *Histogram Plots for Features: The below figures show that at feature number 8, there is two values only: [0,1]. However, there is a high imbalance between the counting of value "0" compared to value "1". The question is raised that this feature is a noise feature or not? Can we drop this feature? <Obviously, almost the samples in this feature are 0, no matter what class they are.>*

*The features: number 4 and number 5 also get the highly imbalance in distribution of samples. They should be checked for the removing or not.*

* *Outliers Removal: We need to go to check the outlier of data before use. Because the noise of data can affect deeply to the final performance of model. We plot the outlier of each feature in the figures as follows*.

*Observe from the outlier figures, we firstly should drop the feature number 8.*

* *Handling the normalization:* 
  + *Check the balance of the data set: we will check the balance of this dataset. If data set is not balance and training produces a low accuracy, we will go back to dataset and perform over/under sampling technique. Then processing the training again to improve the accuracy. We found out that this data set is unbalance.*



* + *We need to normalize by fit the train data with min-max scaler, then transform the corresponding test data with that scale. As a result, our data will be in a range between 0 and 1.*

## Support Vector Machine Approach

From the above preprocessing data progress, we found that data may be imbalanced. Hence, we will split into two cases: data set is balance, and the data set is imbalanced. We used SMOTE to handle imbalanced data set.

Support Vector Machine applied Cross Validation will be mentioned as SVM\_CV in this report.

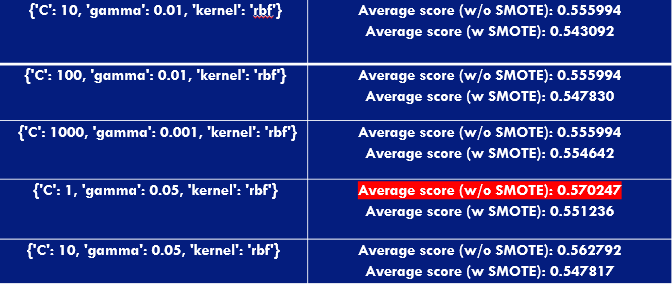
Support Vector Machine applied Cross Validation and applied SMOTE will be mentioned as SVM\_CV\_SMOTE.

We applied cross validation by randomly splitting our cleaned dataset into 10 consecutive folds using K-Fold cross validation. This method provides train/test indices to split data in train/test sets.

Firstly, we run the function SVC(support vector classifier) with the default parameter values for both SMV\_CV and SVM\_CV\_SMOTE. The outcome results are as follow. The result shows that applying SMOTE give us a lower accuracy scores in classifying our data. Therefore, applying SMOTE won’t be helpful.

In order to improve the accuracy of our SVC method, we apply parameter tuning. We used GridSearchCV(grid search cross validation method) in scikit-learn library to estimate the best parameter sets that may be able to help our SVC produces a better result. After running cross-validation procedures, search the best parameters for SVM algorithm and recording a list of the best parameter sets, we went back and applied these parameters to your SVC method.

* {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}
* {'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}
* {'C': 1000, 'gamma': 0.001, 'kernel': 'rbf'}
* {'C': 10, 'gamma': 0.05, 'kernel': 'rbf'}
* {'C': 1, 'gamma': 0.05, 'kernel': 'rbf'}



By using parameter tuning, we found out the best fit parameter for our SVC. As the result, we improve the accuracy for our SVM\_CV from 55.60% to 57.02% which is about 1.5%.