**Capstone Project I – Milestone Report**

**Data Mining and Data Wrangling and Inferential Statistics:**

Data is obtained from United States Cancer Statistics, 1999-2013 provided at the Centers for Disease Control and Prevention.

<https://wonder.cdc.gov/CancerMort-v2013.html>

Data was downloaded from the above website into 60 text files in which 30 files were downloaded for females and 30 files for males, each having same feature variables like cancer sites, states, year, age, death, crude rate, population and race. These 60 files were concatenated after adding new variable name ‘sex’ and assigning female and male values to the corresponding files.

Subsequently, data wrangling was performed to deal with redundant columns likes State code, notes, etc. These columns were deleted. Also rows with Nans/Null values were deleted. Some rows had ‘Not Applicable’ or ‘Suppressed’ values for Death variable these rows were also deleted. Numeric features like Deaths, Crude Rate, Population were checked for datatypes and were converted into numeric datatype for further calculations.

Further, data wrangling was performed to remove year feature from the data. Data was grouped using all other features categorical features and numeric features were summed over all the years and year column was dropped.

Frequency distribution of Deaths was visualized to see that most of the deaths were below 1000 deaths. Further % deaths per crude rate were calculated and it was seen that for some rows crude rate which is number of cancer cases reported to the hospital were more than deaths due to cancer, which means there is discrepancy in the recording of crude rate, therefore % deaths per population was calculated. After carefully exploring the frequency distribution of % deaths per population, two risk cutoff were selected,

1. 0.05%: which is Low/Intermediate risk cutoff i.e. anything below 0.05% is considered to be Low risk and anything above this is Intermediate risk.
2. 0.165%: which is Intermediate/High risk cutoff i.e. anything below 0.165% is considered to be Intermediate risk and anything above this is High risk.

According to this cutoff scheme a new column of Risk with values High, Intermediate, Low was added according to the value of % Death per population and % Death per population was dropped.

**Preprocessing for Machine Learning (ML):**

Now data had 5 independent categorical predictor variables that are State, Cancer site, Age group, Sex and Race and one dependent categorical variable Risk.

In order to perform ML algorithms Risk variable was encoded into numeric variable using LabelEncoder to form new column called Risk\_code in which High was encoded 0, Intermediate was encoded 1 and Low was encoded 2. Also, Risk column was dropped. After labeling 46.5% were LOW, 29.4% were Intermediate and 24.1% were High. This column was used as Y variable in ML.

Rest of the independent variables were also encoded into either 0 or 1 using get\_dummies function of pandas in which each variable was converted into as many variables as the unique values in that variable containing either 0 or 1 indicating the absence/presence of that unique value in that particular row. These columns were used as X variables in ML.

Further both X and Y variables were converted in to array that could be used for ML algorithms.

**Split the data into Train and Test datasets:**

Next we used train\_test\_split to make train and test datasets for both X and Y variables. Train datasets contains 75% of the samples and test sample contains 25% of the samples. Next we check the shape of these 4 data sets. There are certain features of X and Y test and train datasets that hold true. First of all there are same number of rows. The total number of rows in both X datasets is equal to the total number of observations in the data, number of columns in any X dataset is equal to number of features in the data.

**Logistic Regression:**

Train dataset for X and Y were used to fit the Logistic regression model with default parameters which uses L2 penalty and regularization parameter of C=1. Predictions were made on both test data and train data and subsequently accuracy was checked for both Test data and train data. It was found that

***Accuracy of Logistic Regression on Test data: 0.893382352941***

***Accuracy of Logistic Regression on Training data 0.903711484594***

Here accuracy of Training data is 0.903711484594 is pretty good which also means that there is low bias in the model. Also, accuracy of Test data is 0.893382352941 is very close to accuracy of training data hence it means it has low variance too. Both low bias and low variance are good for the model.

Next we tried to improve the accuracy for both training and test data by optimizing the regularization parameter C for the model. By default scikit learn uses L2 regularization model. Here we have optimized L2 regularization (Ridge) which forces the parameters to be relatively small, therefore the bigger the penalization, the smaller (and the more robust) the coefficients are. We used GridSearchCV to check multiple values and obtain the best value for hyperparameter C. After obtaining the best value for C we found that

***Accuracy of Tuned L2 Logistic Regression on Test data: 0.920693277311***

***Accuracy of Tuned L2 Logistic Regression on Training data 0.9375***

Optimizing the regularization parameter C improved the accuracy of both training and test data by.

Next we tried using L1 regularization model to see if that works better. Using L1 penalty without tuning the hyperparameter, the accuracy of the model was found to be

***Accuracy of L1 Logistic Regression on Test data: 0.918067226891***

***Accuracy of L1 Logistic Regression on Training data 0.926645658263***

The model already performs better with Test accuracy of 0.918067226891 and training accuracy of 0.926820728291 as compared to untuned L2 regularized model which had Test data accuracy of 0.893382352941 and Training accuracy of 0.903711484594. So we further optimized the value of regularization parameter C for this model. We used GridSearchCV to check multiple values and obtain the best value for hyperparameter C. After obtaining the best value for C we found that

***Accuracy of Tuned L1 Logistic Regression on Test data: 0.921218487395***

***Accuracy of Tuned L1 Logistic Regression on Training data 0.9375***

After tuning the hyperparameter for L1 regularized model we found that accuracy of training data was exactly same as accuracy for training data with tuned L2 regularised model which is 0.9375. However, the model performs slightly better on Test data with tuned L1 Regularized model with accuracy of 0.921218487395 as compared to tuned L2 regularized model with test accuracy of 0.920693277311.

Since this is a classification model with classes which are slightly unbalanced we also checked the classification report to look at Precision (which tells you how many of selected values are relevant), recall (which tell you how many relevant items were selected), and f1-score and support (number of variables of each type). This shows that there are enough presentation of every class in both training and test model and there is no effect of unbalanced data on the accuracy of the model.

**Training Classification Report:**

precision recall f1-score support

0 0.93 0.96 0.95 1362

1 0.93 0.85 0.89 1687

2 0.94 0.98 0.96 2663

avg / total 0.94 0.94 0.94 5712

**Test Classification Report:**

precision recall f1-score support

0 0.96 0.91 0.94 501

1 0.82 0.90 0.86 503

2 0.96 0.94 0.95 900

avg / total 0.92 0.92 0.92 1904

**Conclusion and further analysis:**

Using logistic regression to fit and predict the data shows pretty high accuracy for both training and test datasets. After using regularizations the accuracy can be increased further. Also further analyzing the model the effect of unbalanced data could not be seen. To further enhance the accuracy of the model other ML algorithms which can perform better on unbalanced data like Decision Trees or Random Forest classifier can be applied to see if those would increase the accuracy of the model.