**Design Document**

**For the**

**Analysis of**

**Prudential Life Insurance**

Contact Persons:

Reema Dmello

Zhaofeng Li

Himaja Vadaga

**Version Control:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Changed by** | **Reviewed by** |
| 10/25/2016 | 1.0 | First Draft | Himaja Vadaga | Reema Dmello  Zhaofeng Li |
| 10/25/2016 | 1.1 | Final | Reema Dmello | Himaja Vadaga,  Zhaofeng Li |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# **Preface**

This document is created to describe the design, development and deployment procedures taken to perform the analysis of the Prudential Life Insurance. It will illustrate the design of the machine learning algorithm, preprocessing and Data modelling in the design document.

# **INTRODUCTION**

This dataset was taken from Kaggle. It has 1 csv file. The csv file named as train.csv had maximum data needed for the analysis.

# **About the Dataset**

* For this project we had a dataset with single csv files so our major task was to understand the dataset in an appropriate manner where we could do some useful analysis and build models accordingly.
* The table has details as follows:

1. Id A unique identifier associated with an application.
2. Product\_Info\_1-7 A set of normalized variables relating to the product applied
3. Ins\_Age Normalized age of applicant
4. Ht Normalized height of applicant
5. Wt Normalized weight of applicant
6. BMI Normalized weight of applicant
7. Employment\_Info\_1-6 A set of normalized variables relating to the employment history of the applicant
8. InsuredInfo\_1-6 A set of normalized variables providing information about the applicant
9. Insurance\_History\_1-9 A set of normalized variables relating to the insurance history of the applicant
10. Family\_Hist\_1-5 A set of normalized variables relating to the family history of the applicant
11. Medical\_History\_1-41 A set of normalized variables relating to the medical history of the applicant
12. Medical\_Keyword\_1-48 A set of dummy variables relating to the presence of/absence of a medical keyword being associated with the application
13. Response This is the target variable, an ordinal variable relating to the final decision associated with an application

* The table had missing values and the data type were given wrong for many columns
* There were about 58381 rows and 128 columns in the match table. Among this there was about 39000 values were missing data.
* In order to build necessary models it was necessary to pre-process the dataset for maximum Accuracy.
* Challenges faced were: Fixing NAs, Reducing the dimensions i.e.: feature selection, converting the categorical columns to numerical data

**Data Preprocessing**

We checked the demographics of the data using Weka and R and found out a pattern. There were some columns which had more than 60% nulls, we decided to eliminate those columns from our dataset.

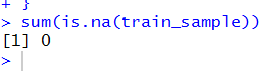
* Count the number of NAs **no\_of\_na**



1. Percentage of missing values with the rest of the data using **lapply** function
2. Excluding the data with 50% or more missing

* Convert character columns to IDs
* Fix the NA Values using a variable called **feature.names** which will calculate the median value for substituting it.
* Check the columns having only 2 values - convert these into binary 1-0 in the same column instead of splitting it into 2 different columns after selecting unique values using **feat.bin** variable.
* Convert the remaining categorical columns into 1-N transformations - dummy variables **catVars --** we have eliminated the columns which would split into more than 3 variables
* For every unique value in the string column, create a new 1/0 column. This is what Factors do "under-the-hood" automatically when passed to function requiring numeric data

After preprocessing:



**After Preprocessing, we have split the data into train – 90% and test – 10%**

**Feature Selection:**

1. We used the trainControl method to select the features that are important for the prediction of the Response variable
2. We then plot these features and we get 20 most important features
3. We then run a LR model on the entire dataset to find the most significant features.
4. We have then combined the both the list of variables together and reduced the number of important features to around 50 variables.

We went back and forth with the preprocessing and selection of the important features.

* R squared value was very low initially – 0.34 – before splitting the categorical columns
* After transforming the categorical values – 0.39
* We initially did not include Product\_Info\_2 because it has more than 6 values, but while running LR I found that it had some significance on the dependent variable and therefore included it in.

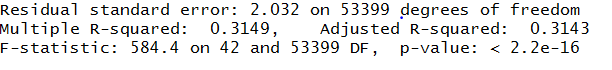
# **Machine Learning Algorithms:**

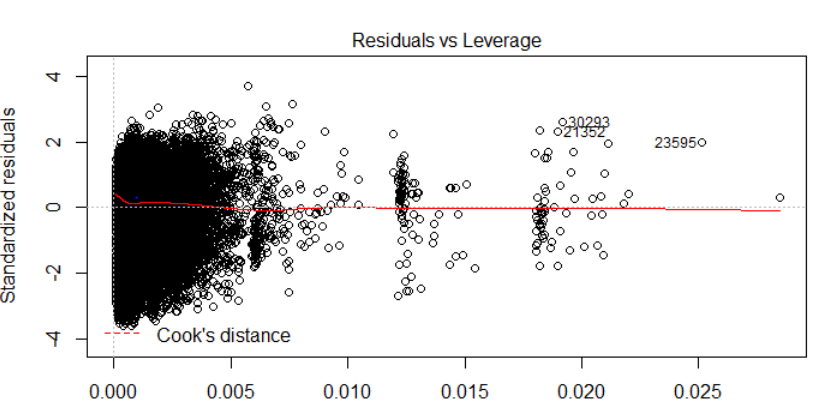
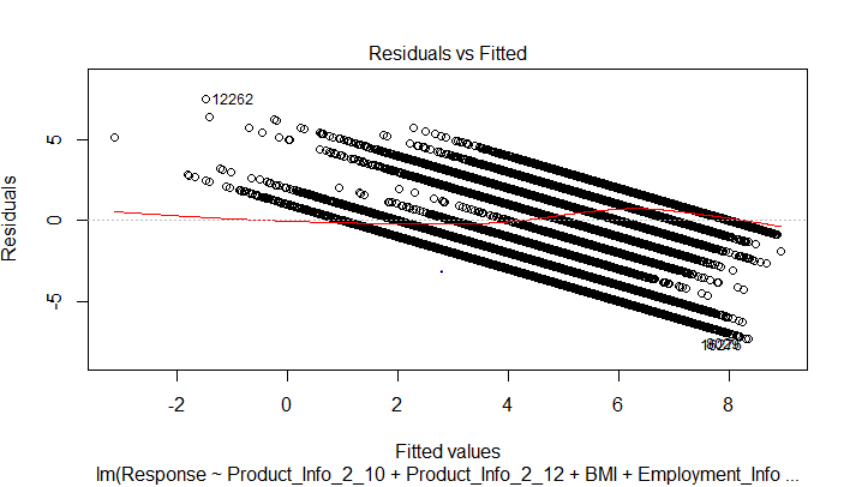
**Linear Regression**

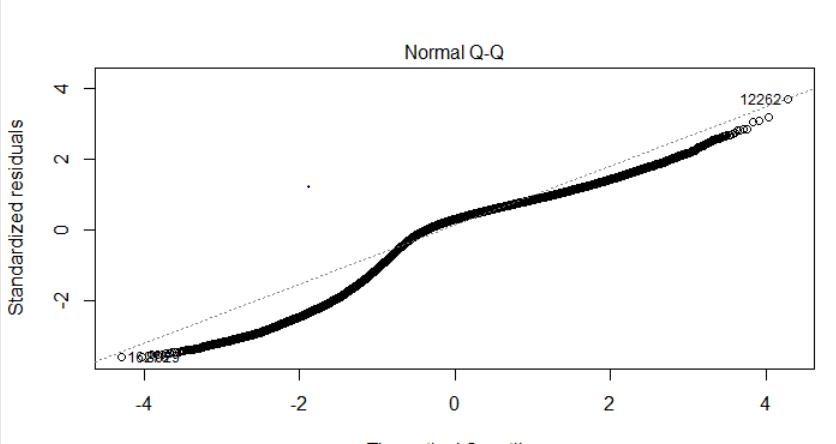
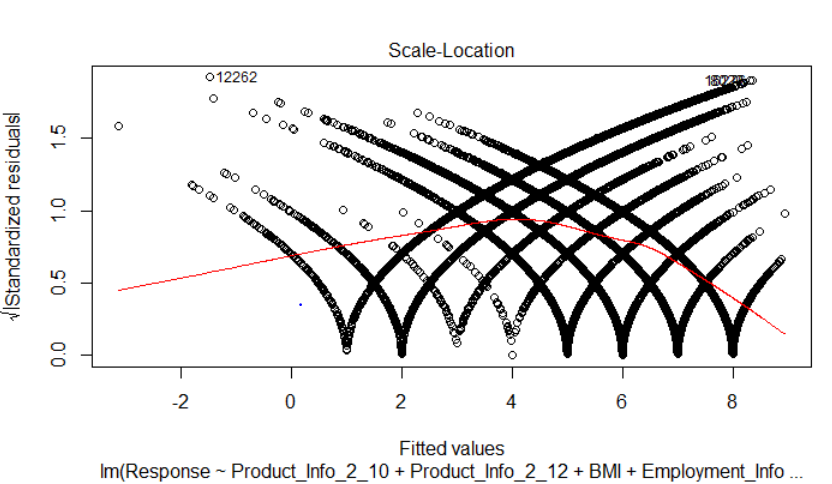
Linear regression is the most basic and commonly used predictive analysis. Regression estimates are used to describe data and to explain the relationship between one dependent variable and one or more independent variables.

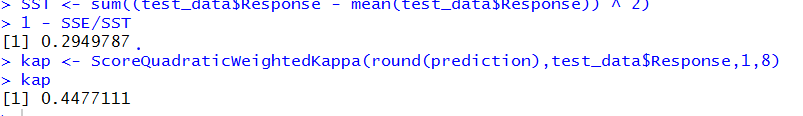
* Train the model based on the features and train data
* predicting the response on the test data from the model
* Calculating the Rsquare
* Calculating the kappa score

For the latest model:





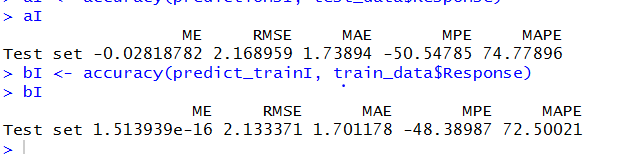
 

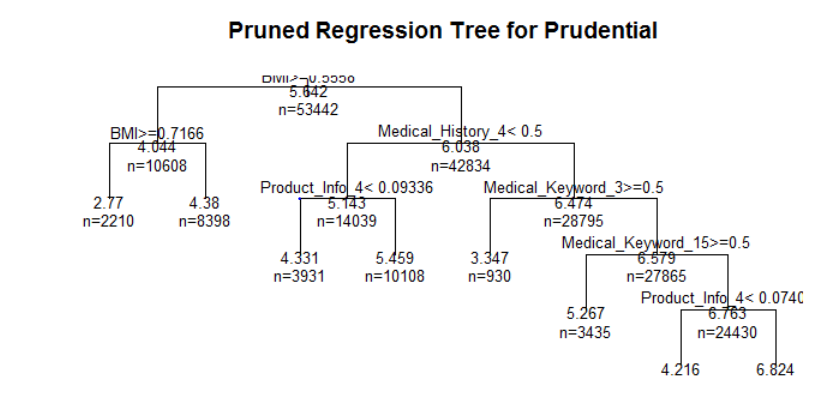


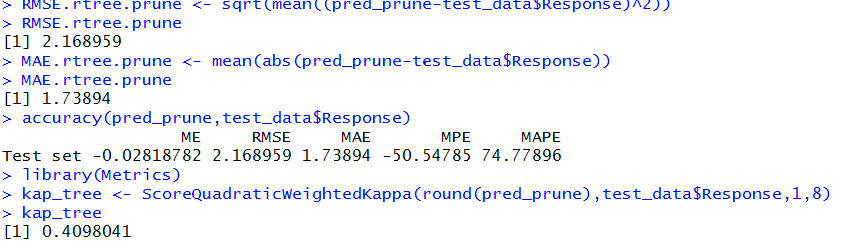
**Decision Tree Regressor**

Decision tree builds regression or classification models in the form of a tree structure. It brakes down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

* Train the model based on the features and train data
* predicting the response on the test data from the model
* Calculating the RMSE
* Calculating the kappa score
* Changing the parameters on min split, max-depth and check the RMSE
* Calculate the best CP value
* Prune the tree to avoid overfitting and use the bestcp value
* Re-calculating the RMSE
* Re-calculating the kappa score







**Support Vector Machine**

In machine learning, **support vector machines** are supervised learning models with associated learning algorithms that analyse data used for classification and regression analysis. An SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible.

* We have considered 10% of the entire data as it takes lot of time for large data set.
* Splitting the dataset, hence training it through the svm model.
* predicting the data from the svm model
* Calculating rmse Error on the predictions
* Performing grid search to tune the result taking two parametres (epsilon = seq(0,0.2,0.01), cost = 2^(2:9) and epsilon = seq(0,1,0.1), cost = 2^(2:9))
* Choosing the best model from the above two tuned results
* Calculating the RMSE error for he tuned Model.
* We performed an epsilon-regression, we did not set any value for epsilon (ϵ), but it took a default value of 0.1. There is also a cost parameter which we can change to avoid Overfitting
* This means we can try another grid search in a narrower range we will try with ϵϵ values between 0 and 0.2.

**CONCLUSION:**

This table gives the comparison of performance metrics of 4 different algorithms used in clustering

|  |  |  |  |
| --- | --- | --- | --- |
|  | Linear Regression | Decision Tree | Support Vector Machine |
| Kappa Score | 0.45 | 0.40 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 

# **References**

* <https://www.kaggle.com/>