Survival Analysis of Cancer Patients

Sushmita Sivaprasad

March 2017

1 Abstract

The given training data set consists of details of 15387 patients and their state of cancer at the time of diagnosis and the progression of the disease. We also know by the end of the 7th year if the patient is still alive or not. The data-set shows 6651 patients survived whereas 8734 died may or may not be of cancer. Our goal here is to predict from the given test data set given the same features if we can correctly predict a patient survives or not after 7 years from the year of diagnosis. This is a binary classification based problem , where we need to classfy the patients as alive or dead after 7 years based on the given features.

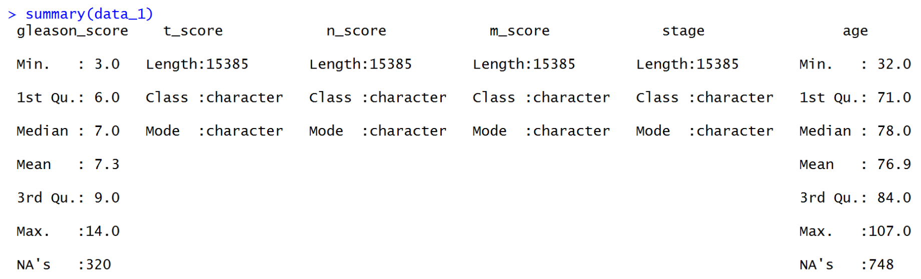
The model build is in R code and contains description of the data preprocessing , mining and prediction.

2 Exploring the data

1. **Data Dimension** : The dimension of the data can be obtained from r code

   ```
   > dim(data_1)
   > [1] 15385 31
   ```

2. **Summary of the Dataset** : The figure shows the summary of a few

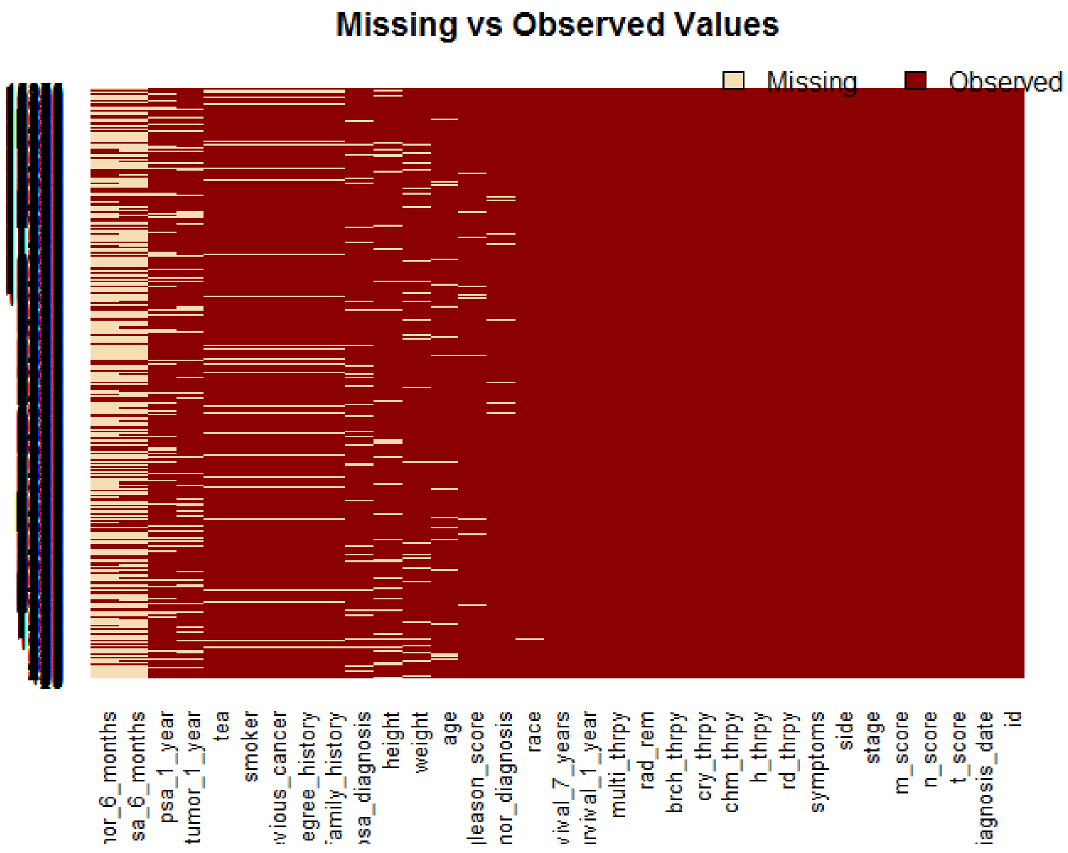


```
> summary(data_1)
 gleason_score    t_score            n_score            m_score            stage                age
 Min.   : 3.0   Length:15385       Length:15385       Length:15385       Length:15385       Min.   : 32.0
 1st Qu.: 6.0   Class :character   Class :character   Class :character   Class :character   1st Qu.: 71.0
 Median : 7.0   Mode  :character   Mode  :character   Mode  :character   Mode  :character   Median : 78.0
 Mean   : 7.3                                                                               Mean   : 76.9
 3rd Qu.: 9.0                                                                               3rd Qu.: 84.0
 Max.   :14.0                                                                               Max.   :107.0
 NA's   :320                                                                                NA's   :748
```

of the variables in the dataset , R gives us the nature of the variables provided and the statistical data for numeric values. It even shows us the missing values under each variable , given here as NA. From this we can we that 16 of the 32 variables have missing values within them.

3. **Missing Values** I used the package **Amelia** in order to visually show the missing values in the dataset. It generates an image as follows , showing the Missing Vs Observed Value in the dataset.

```
>library("Amelia")
>missmap(data_1, main= ("Missing vs Observed Values"))
```



4. **Finding Outliers using Box Plot** : The data consists of 3 continuous variables , age , height and weight. When a box plot is taken on these 3 variables , we can observe certain outliers. Which may or maynot indicate error's within the dataset

- **Age** : We can see that a few of the patients have age above 100 , which is certainly an outlier, this could be a data entry error.

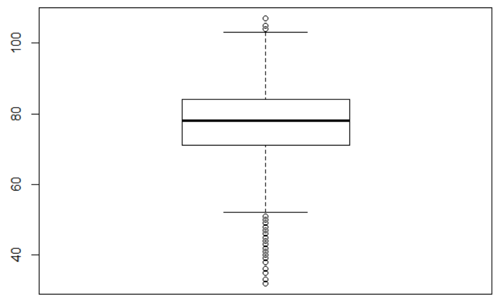


Figure 1: Box Plot of Age

- **Weight** : We can see here that a few of the patients have weights more than 300 which is definitely over weight, but these value may or may not be correct here.

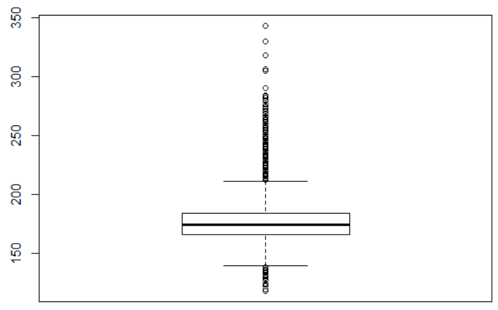


Figure 2: Box Plot of Weight

- **Height** : This plot seems quite normal and we dont observe any outliers in this variable.

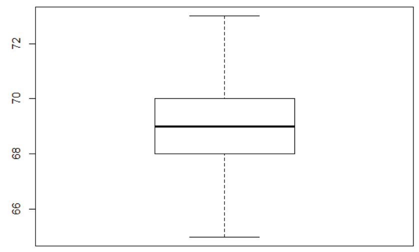


Figure 3: Box Plot of Height

3 Data-preprocessing

The training data set consists of 32 features , of which we have both numeric and categorical variables. The following steps were done in order to clean the data to bring it to the required form to perform different prediction models

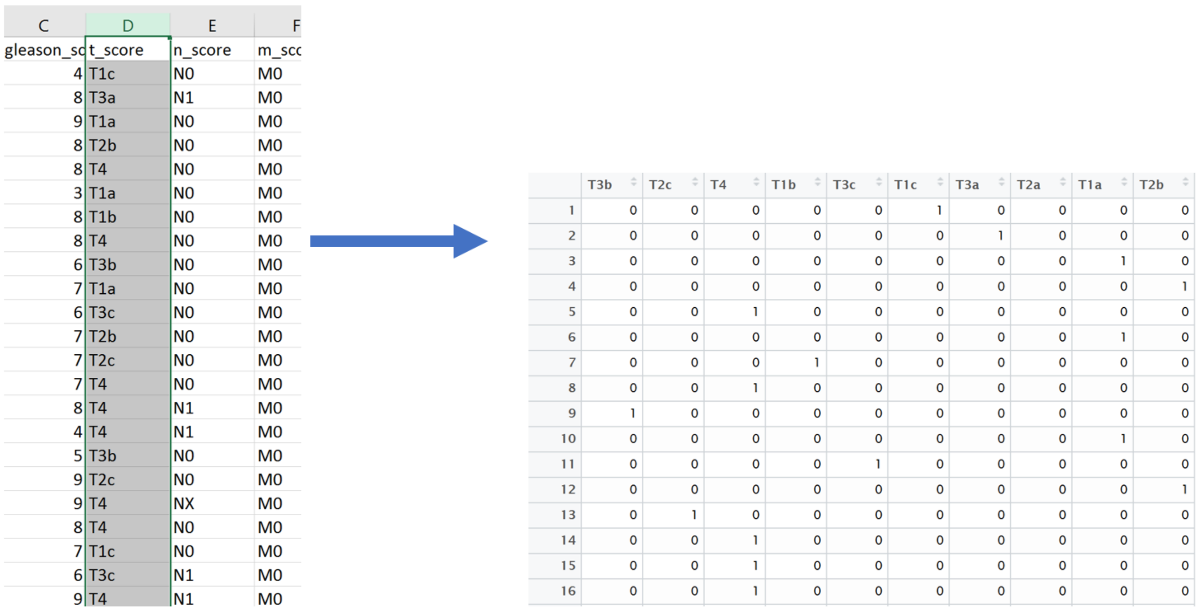
- **Removing ordinal data** : The column serial number and the date was removed as it was unnecessary. The dataset already provides us a 7 year data of the patients and date being an ordinal data type wasn't considered as an important feature here.
- **Converting datatypes** : Since we have a few character type data , we would be requiring to convert them into numerical variables for which we first convert them into character types from integer type using **as.character** in R.

  ```
  > typeof(data_$symptoms)
  "integer"
  ```

- **Splitting the symptoms variable**: The "symptoms" variable consisted of a number of different symptoms comma separated in the same row of the variable column , this categorical variable had to be converted to binary numerical format , for which each of the comma separated values were split using **cSplit** , the character type was converted to an integer type and unique symptoms were extracted as shown in the below code

  ```
  >symp_split <- cSplit(symp, splitCols = 'symptoms', sep=",",
  direction = "long", stripWhite = TRUE, makeEqual = NULL,
  type.convert=FALSE)
  >symptom_list <- as.vector(t(as.matrix(unique(symp_split))))
  ```

- **Handling categorical variables** : The categorical variables were converted into dummy variables which is more suitable for training the model on and doing analysis. The categorical variables such as **t_score,n_score, m_score, stage, side** were converted to numerical format by identifying the unique values in them and converting them into columns and proving a binary value.



C	D	E	F
gleason_sc	t_score	n_score	m_scc
4	T1c	N0	M0
8	T3a	N1	M0
9	T1a	N0	M0
8	T2b	N0	M0
8	T4	N0	M0
3	T1a	N0	M0
8	T1b	N0	M0
8	T4	N0	M0
6	T3b	N0	M0
7	T1a	N0	M0
6	T3c	N0	M0
7	T2b	N0	M0
7	T2c	N0	M0
7	T4	N0	M0
8	T4	N1	M0
4	T4	N1	M0
5	T3b	N0	M0
9	T2c	N0	M0
9	T4	NX	M0
8	T4	N0	M0
7	T1c	N0	M0
6	T3c	N1	M0
9	T4	N1	M0

	T3b	T2c	T4	T1b	T3c	T1c	T3a	T2a	T1a	T2b
1	0	0	0	0	0	1	0	0	0	0
2	0	0	0	0	0	0	1	0	0	0
3	0	0	0	0	0	0	0	0	1	0
4	0	0	0	0	0	0	0	0	0	1
5	0	0	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	1	0
7	0	0	0	1	0	0	0	0	0	0
8	0	0	1	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	1	0
11	0	0	0	0	1	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	1
13	0	1	0	0	0	0	0	0	0	0
14	0	0	1	0	0	0	0	0	0	0
15	0	0	1	0	0	0	0	0	0	0
16	0	0	1	0	0	0	0	0	0	0

Figure 4: Converting Categorical(t_score) to Binary Numeric(T3b,T2c,..) Variable

- **Handling Missing Values**: We can see that 16 of these variables have missing values in them. To impute the missing values based on all the other values the mice package was used in R (multivariate imputation by chained equation) to fill all the missing values. Under MICE the predictor mean method was used. The missing values here are filled in a random manner from a non-missing observed value. The PMM selects the missing value by imputing the values in the regressor equation.

4 Feature Selection using Regression

After converting all the categorical to numerical values and removing the redundant variables, we are left with a dataset containing 66 variables. We would

need to perform a feature selection on this set of variables to find which of the variables are least contributing to predict the survival of a cancer patient for the 7th year.
A correlation matrix is built on the dataset and the highlycorrelated variables are found. Eliminating the variables which are more correlated with each other helps in building a better model as we are reducing the dimensions of the dataset. I used the **Caret Package** in R where the `findCorrelation` function was used and a cut off of 0.3 was given to extract more number of variables which can be eliminated.

```
>highlyCorrelated < findCorrelation(correlationMatrix, cutoff=0.3)
>print(highlyCorrelated)
>[1] 61 55 52 53 60 44 62 59 15 12 11 37 14 58 41 56 17 18
64 2 6 5 65
```

It shows us the particular columns numbers that can be eliminated from the dataset with a correlation value higher that 0.3

5 Building a Model

2 different models were built on the dataset , Binomial Logistic Regression and Decision Trees as the dataset is large and we have a lot of variables. Both or these algorithms are faster to process in a short span of time and computationally less expensive.

5.1 Binomial Logistic Regression

Assuming normality and linearity of the dataset ,I have used logistic regression to predict a binary value for the `survival_7_years variable`. The function used here is **glm()**.The coefficient of all the predictors and the intercepts are calculated here. The predicted values are converted into a binary form as our goal here is to find if the patient is still alive during the 7th year , which is represented as 0 if the person is dead and 1 if the patient is still alive.

The model is built using the survival_7_years as the class variable

```
>model <-glm(survival_7_years ., family= binomial(logit), data=feature_sel)
>summary(model)
```

The survival_7_years is predicted for the test data

```
>fitted.results <- predict(model,newdata=subset(test_imputed_data_final,
select=c(-highlyCorrelated)),type='response')
```

The values are converted into a binary form where 0 represents dead and 1 represents alive

```
>test_imputed_data_final$survival_7_years <- ifelse(fitted.results
> 0.5,1,0)
```

5.2 Decision Trees

: It performs a multiple variable analysis and is used to classify the data to predict the predictor variable. It attempts to find a strong link between the input variables and the target predictor variables. I used the library **rpart** in order to run a decision tree algorithm
rpart is used to build the decision tree model using class as a method

```
> decision_tree <- rpart(survival_7_years ., data= feature_sel,
method="class")
> head(decision_tree)
```

The predictor variables in the test dataset `dec_test` is replaced with the binary values after performing the decision tree model.

```
> dec_test$survival_7_years<-predict(decision_tree,newdata=subset
(test_imputed_data_final,select=c(-highlyCorrelated)),type='class')
```

The figure below shows us the variables used and how the tree gets classified at each level.I used the **rattle package** on R to get this visualization

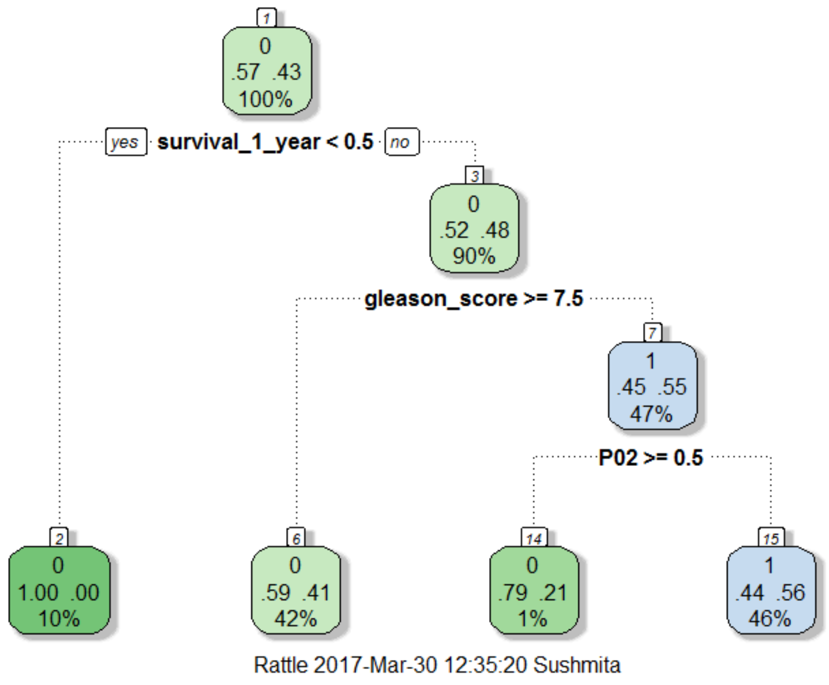


Figure 5: Decision Tree Visualization

The figure above shows the decision tree for the variables used in the model, we have survival_1_year , gleason_score , P02. We take the root node survival_1_year where the box is green as the majority of the values are 0 (represents blue if

majority value is 1) , we can see the split here as 57% of the data is 0 and 43% of the data has 1. The value of the split classfies the data into value above and below the split value. The % values within the box gives us an idea of how the data gets split under the parent nodes into the leaf nodes. We can even build rules within decision trees , Here leaf node 15 shows the values when is has satisfied all the conditions where
survival_1_year > 0.5 and gleason_score < 7.5 and P02 < 0.5 will give a value of 1 for the 56% of the 46% of the entire data.

6 Validation on the Training Dataset

In order to validate the data and to calculate the accuracy of the 2 models that were built , I decided to test the model on the already given training dataset by splitting into a train and test dataset with a 75:25 split.

6.1 Using Logistic Regression

The dataset after data validation and feature selection was used , on which the logistic regression model was built and tested on the test dataset.

```
   Data Validation
> correlationMatrix_train <- cor(train[,1:66])
> print(correlationMatrix_train)
> highlyCorrelated_train <- findCorrelation(correlationMatrix_train,
cutoff=0.3)
> print(highlyCorrelated_train)
Feature Selection
> feature_sel_train <- train[-highlyCorrelated_train]
> write.csv(feature_sel_train,"feature_sel_train.csv")
Building the model > model_test <-glm(survival_7_years ., family=
binomial(logit), data=feature_sel_train)
> summary(model_test)
> fitted.results.train <- predict(model_test,newdata=test,type='response')
> test$survival_7_years_pred <- ifelse(fitted.results.train
> 0.5,1,0)
Evaluating the model
> accuracy <- sum(test$survival_7_years == test$survival_7_years_pred)/nrow(test)
> accuracy
```

The accuracy of this model on the training dataset was calculated as 63.89%

6.2 Using Decision Trees

The same train and test dataset was used to build the model upon and to predict the predictor values for the test dataset.Here the train dataset is labelled as `feature_sel_train` and the test dataset is `test`

```
Evaluating the model using classfication by decision Trees
on training dataset
> decision_tree_train <- rpart(survival_7_years  ., data=
feature_sel_train,method="class")
> test$survival_7_years_pred_tree<- predict(decision_tree_train,
newdata=test,type='class')
> accuracy_tree <- sum(test$survival_7_years ==
test$survival_7_years_pred_tree)/nrow(test)
> accuracy_tree
```

It gives us a simplistic decision tree which results in a similar result when compared with the logistic regression model as the accuracy of this model was 62.95%

7 Future Work

- **Ensemble Methods** : Ensembles methods such as bagging (bootstrap aggregation) and boosting could be implemented. Bagging can help in reducing the variance of the data and help predict the predictor variable , where we can use random forest model. In Boosting we can use Gradient Boosting , which is a tree based method. We use boosting to reduce the bias.
- **SVM** :It is a non-linear classification model thaat performs regression and classification and detects any outliers in the dataset.The dataset is huge and given the time, the model takes a lot of time to process because of the complex algorithm involved.
- **Time Series Survival Analysis**: Taking survival in year 7 as an event here, we can predict the event by converting the dataset into a time series.

8 Conclusion

From the models performed on the dataset , logistic regression shows us that about 57.17% of the patients have died due to cancer or any other form of disease or cause of death, which is calculated from the dataset "sushmita_score_logistic.csv" . 52.79% deaths have been observed in the "sushmita_score_decisiontree.csv" dataset , where the result was the output of the decision tree classification model.