

**WESTERN SYDNEY
UNIVERSITY**



Computing, Engineering & Mathematics

ASSIGNMENT / REPORT COVER SHEET

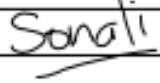
This sheet must be attached to all material being submitted for marking.

Student name: Student number:	Sonali Marasini 19913230	Student name: Student number:	
Sections completed individually		Sections completed individually	
Unit name & number	Data Science: 301044		
Tutorial day and time:	Friday 11 am to 1 pm		
Title of Assignment:	World Happiness Report		
Student Submitting the Assignment:	Sonali Marasini		
Date submitted:	16/10/2020		

Student Declaration (must be signed)

Declaration:

- ☐ I hold a copy of this assignment if the original is lost or damaged.
- ☐ I hereby certify that no part of this assignment or product has been copied from any other student's work or from any other source except where due acknowledgement is made in the assignment.
- ☐ No part of the assignment/product has been written / produced for me by any other person except where collaboration has been authorised by the subject lecturer/tutor concerned
- ☐ I am aware that this work may be reproduced and submitted to plagiarism detection software programs for the purpose of detecting possible plagiarism (**which may retain a copy on its database for future plagiarism checking**)
- ☐ I hereby certify that no part of this assignment or product has been submitted by me in another (previous or current) assessment, except where appropriately referenced, and with prior permission from the Lecturer/Tutor/ Unit Co-ordinator for this unit.

Student signature and date:		16/10/2020
Student signature and date		

Note: An examiner or lecturer/tutor has the right to not mark this assignment if the above declaration has not been signed.

World Happiness Report for Data Science Assignment

Sonali Marasini

*School of Computer, Data and Mathematical Sciences
Western Sydney University
19913230@student.westernsydney.edu.au*

Abstract—This paper reviews the correlation between the happiness score of several countries and socioeconomic factors using several data modelling and analysing techniques. These models have been tested and validated. In the end, it has been found that there is a direct relationship between several socioeconomic factors of a country and the happiness of the people of that country. However, since happiness is a tricky aspect to measure, a further detailed study needs to be conducted to discover what other factors could affect the happiness of an individual.

1. Introduction

A healthy mindset requires inner happiness. Despite mental health being as important as physical health, it is often overlooked. This happiness is a feeling and can be a difficult thing to measure.

A ground-breaking study of the state of global happiness, the World Happiness Report, rates 156 countries by how happy their people consider themselves to be. [1], [2] have reported this study in detail. This study mainly centres on the innovations, societal norms, disputes, and policies of the government that have been used in ranking these countries.

There are six factors used to determine the happiness score of each country and rank them in the happiness ladder. The six factors are levels of Gross Domestic Product (GDP), life expectancy, generosity, social support, freedom, and corruption.

This study has combined data from the years 2017 through 2019 to make the sample size large enough for accurate study. The typical annual sample they have collected 1000 people, meaning through 3 years they have collected 3000 samples per country.

2. Data Exploration and Pre-processing

2.1. Data Cleaning

There are 156 observations in the dataset. There are six independent variables namely, GDP per capita, Social support, Healthy life expectancy, Freedom to make life choices, Generosity, and Perceptions of corruption. The predictor

variable is the happiness score which is calculated using all these independent variables. Based on the happiness score, all the 156 countries are then ranked with 1 being the country with the highest happiness score and 156 being the country with the lowest happiness score.

A new column with the name Score Level has been added to the data set. This column has two values 'H' and 'L' depending on the value of the happiness score. If the happiness score is less than or equal to 5.99 then, the Score Level is 'L' indicating a low happiness score whereas, if the happiness score is more than 5.99 then the Score Level for that country is 'H' indicating a high happiness score.

A few values in the data set are found to be 0. The zeroes in the data set have been replaced with the mean value of the entire column values. This has been done so that the modelling gives a more accurate result.

2.2. Data Relationship

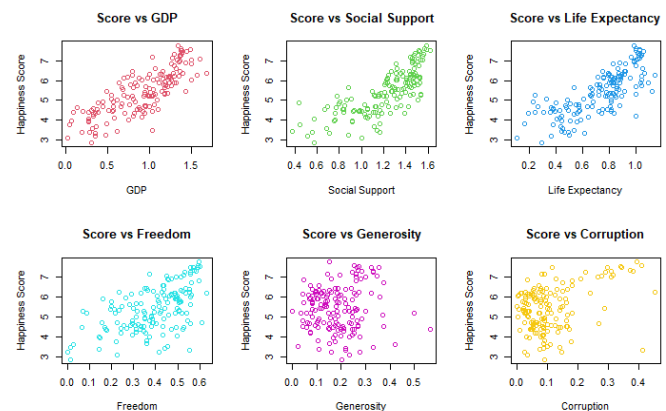


Figure 1. Data Relationship

The relationship between the independent variables and the outcome variable has been plotted in figure 1. There seems to be a strong linear relationship between the happiness score against GDP, social support, and life expectancy of a country. There does not seem to be a linear pattern between happiness score and generosity and happiness score

and corruption. Lastly, the relationship between happiness score and freedom seems to be linear but a weak one.

2.3. Correlation

[3] has defined correlation coefficient as a kind of numerical measure to determine a statistical relationship between any two variables. To find out if the independent variables have a positive or a negative relationship with the outcome variable, the correlation values were calculated. All the variables have a positive correlation with the outcome variable. As expected, from the plots, GDP, social support, and life expectancy have a strong positive relationship with the happiness score with correlation coefficients of 0.7964624, 0.7727645, and 0.7709061 respectively. Generosity has the weakest relationship with a correlation coefficient of 0.07582369. The correlation coefficients of freedom and corruption are 0.5667418 and 0.3856131 respectively.

3. Aims and Objectives

The objective of this data science study is to analyse the happiness score data and determine if there is any pattern. The aim is to predict the happiness score of any country given the independent variable values. The goal is to see if out of the several socio-economic factors, which factors have the most impact on the happiness score.

4. Multiple Linear Regression

4.1. Fitting a Model

Multiple Linear Regression is a statistical technique that predicts the outcome of an outcome variable using multiple explanatory variables. [4] A multiple linear regression model has been constructed and the least square estimates of the model parameters have been calculated. The fitted model is as follows,

$$E(\text{Happiness Score}) = 1.6589 + 0.8570 \text{ GDP} + 1.2354 \text{ SocialSupport} + 0.9781 \text{ LifeExpectancy} + 1.3963 \text{ Freedom} + 0.5910 \text{ Generosity} + 0.8384 \text{ Corruption}$$

4.2. Testing the Significance

TABLE 1. P VALUES

Variables	P Values
GDP	0.000123
Social Support	1.19e-06
Life Expectancy	0.005350
Freedom	0.000306
Generosity	0.241562
Corruption	0.132554

Hypothesis testing has been conducted to test the significance of the model parameters, $H_0 = \beta_i$ vs $H_1 \neq \beta_i$. The

p values of the coefficient parameters for different variables is given in table 1.

It can be observed that the p values for Generosity and Corruption are greater than 0.05. Thus, there is not enough evidence to reject null hypothesis for Generosity and Corruption. Thus, Generosity and Corruption do not have any significant effect on the happiness score. Hence, the resulting model can be obtained by removing these variables from the predictors. The model thus obtained is,

$$E(\text{Happiness Score}) = 1.7773 + 0.8924 \text{ GDP} + 1.1315 \text{ SocialSupport} + 1.0197 \text{ LifeExpectancy} + 1.7710 \text{ Freedom}$$

In this model, the R squared value is calculated to be 0.7686. This implies that about 76.86% of the variation in the dataset is explained by the model.

4.3. Assessing the Accuracy

The variance table has been analysed to assess the accuracy of the model. From the Anova table, F statistics = 125.4 on 4 and 151 degree of freedom which is greater than 2.431562. Hence, the model is adequate and GDP, Social Support, Life Expectancy, and Freedom variables have significant linear relationship with Happiness Score.

4.4. Residuals

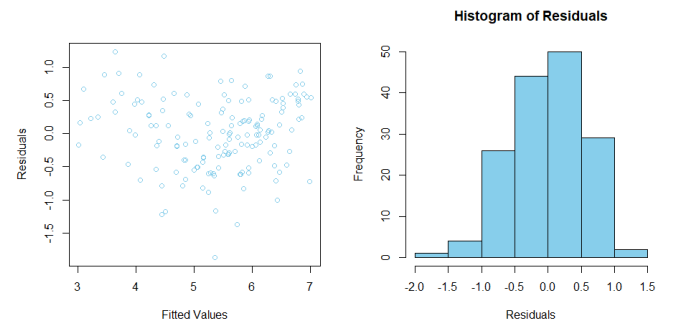


Figure 2. Residuals

In figure 2, the residuals from the model have been plotted against the predicted values and the histogram of the residuals has been created. It can be depicted that the mode is closer to zero and the residuals follow a normal distribution which is slightly positively skewed.

The residuals have been plotted in figure 3. We would expect the residuals versus predicted values plot to be random. The first graph shows a nonlinear pattern. Hence, the linearity assumption is violated. There is some pattern in the residual implying that the pattern in the dataset is not completely captured by the model.

5. Cross Validation

Cross-validation is a technique of resampling used on a small data set to validate models. Six different models with

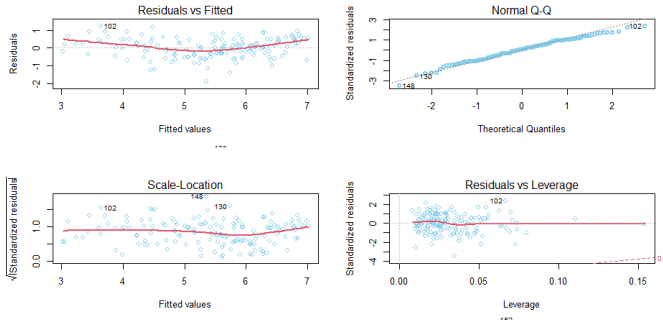


Figure 3. Residual Plots

different variables and combination of these variables have been created. A validation set approach for each of these models have been used and the goal is to select the model with minimum validation error.

The data set has been divided into training and validation in the ratio of 60:40. The data is then fitted in the six different models. Out of these six models, the first model has one variable, second model has two variables, third model has three variables and so on. Using these model, the target variable is predicted for training data and validation data. Similarly, Leave One Out Cross Validation (LOOCV) and K-Fold Validation is done in these six models. The value of K is considered as 10. The errors calculated from these four different cross validation approaches has been shown in table 2.

TABLE 2. ERRORS

	Model1	Model2	Model3	Model4	Model5	Model6
Training	0.3849502	0.3294415	0.3146589	0.2715977	0.2674578	0.2516869
Validation	0.5488410	0.4050460	0.3814438	0.3090415	0.3051749	0.3287890
10-Fold	0.4593798	0.3793884	0.3583951	0.3085208	0.3098113	0.3150800
LOOCV	0.4610816	0.3671266	0.3543573	0.3047021	0.3044499	0.3065170

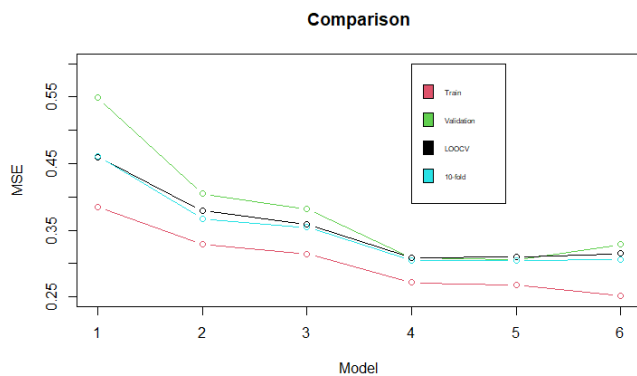


Figure 4. Validation Set Approach

The mean squared values for training, validation, LOOCV and 10-Fold for these six models are plotted in

figure 4. Upon observing the plot, it can be deduced that the best selection would be model 4. Model 4 has been created using variables GDP, social support, life expectancy, freedom and generosity.

6. Decision Trees

6.1. Classification Tree

There is a categorical variable with the name of Score Level created in the data set in the preprocessing stage. This variable has been converted into a factor variable. The data set has been divided into training and testing data set. A classification decision tree has been fitted on the training data set. The decision tree is shown in figure 5.

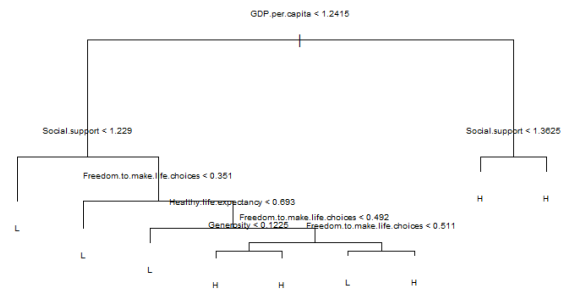


Figure 5. Decision Tree

In this decision tree, there are 9 terminal nodes. The variables that are actually included in the tree construction are GDP, social support, freedom, life expectancy and generosity. Corruption has not been considered while making this tree. We can deduce that this variable is not needed to make the model more significant.

6.2. Pruning

The cross-validation plot is shown in figure 6. According to the cross-validation plot, a rapid decrease can be observed from order 1 to order 2. From this point, there is a constant decrease till order 7 and the plot is somewhat constant after. Therefore, it can be deduced that the best size of the tree model is 7.

The pruned classification tree as shown in figure 7 has 7 terminal nodes. The variables that have been included in this pruned tree are GDP, social support, life expectancy and freedom. From the classification decision tree, it can be concluded that higher the GDP the higher the happiness score. Similarly, life expectancy and freedom also have a positive impact on happiness score.

6.3. Testing the Accuracy

The accuracy of the model on the training data is 0.05645 or 5.6%. It is obvious that the model would perform well in the training data set.

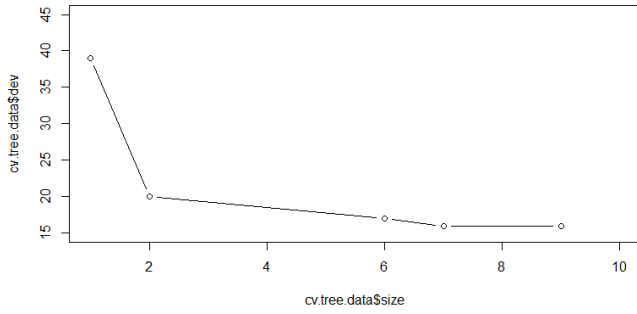


Figure 6. Cross Validation Plot

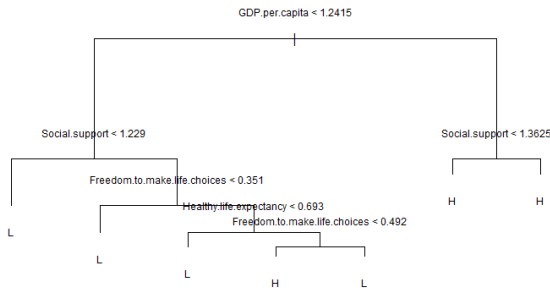


Figure 7. Pruned Tree

To determine if the model has good accuracy, it needs to perform well on the test data set. The target variable for the test data set have been predicted. The predicted values have been compared against the actual values. The confusion matrix for the test data set is shown in table 3.

TABLE 3. CONFUSION MATRIX

		Predicted	
		H	L
Actual	H	12	4
	L	1	15

Therefore, it the misclassification rate for the testing data is calculated to be 0.15625 or 15.625%. This training misclassification rate is significantly higher than the testing misclassification rate. Therefore, it could be said that the model is overfitting in the training data set.

7. Principal Component Analysis

Principle Component Analysis (PCA) is an unsupervised learning technique which has been performed to increase the interpretability of the data set without the loss of informa-

tion. [5] First, the mean and the variance of the independent variables have been observed as seen on table 4.

TABLE 4. MEAN AND VARIANCE

	GDP	SocSupp	LifExp	Freedom	Generos	Corrup
Mean	0.9051474	1.2088141	0.7252436	0.3925705	0.1848462	0.1106026
Variance	0.9051474	1.2088141	0.7252436	0.3925705	0.1848462	0.1106026

Scaling has been done to avoid the variable with the highest mean dominate the first principle component. The coefficients required to calculate the six different principle component is given in table 5.

TABLE 5. PCA

	PC1	PC2	PC3	PC4	PC5	PC6
GDP	-0.51459462	0.2278181	-0.02380878	0.2404018	-0.183763758	-0.76887077
Social Support	-0.49064918	0.2202837	0.28141961	-0.0633134	0.771259058	0.18081020
Life Expectancy	-0.51056655	0.1922719	0.02808632	0.2748057	-0.509355608	0.60547761
Freedom	-0.38095770	-0.3521218	0.11855036	-0.8104252	-0.240066817	-0.04905325
Generosity	-0.05948407	-0.6935067	0.58081716	0.4189146	-0.005250993	-0.05142470
Corruption	-0.29173692	-0.5076063	-0.75368730	0.1743609	0.232996913	0.06701907

Using these coefficients, the principle components can be represented as below,

$$PC1 = -0.51583025 * GDP - 0.48232080 * Social Support - 0.50821063 * Life Expectancy - 0.38551214 * Freedom - 0.05806569 * Generosity - 0.30168944 * Corruption$$

$$PC2 = 0.2082045 * GDP + 0.2223429 * Social Support + 0.2141943 * Life Expectancy - 0.3435152 * Freedom - 0.7039110 * Generosity - 0.4978373 * Corruption$$

$$PC3 = -0.07765971 * GDP + 0.35572318 * Social Support - 0.05577799 * Life Expectancy + 0.24935585 * Freedom + 0.48379134 * Generosity - 0.75371507 * Corruption$$

$$PC4 = 0.22026792 * GDP + 0.02339788 * Social Support + 0.26912517 * Life Expectancy - 0.78471459 * Freedom + 0.51123624 * Generosity + 0.03696948 * Corruption$$

$$PC5 = -0.27711289 * GDP + 0.76435099 * Social Support - 0.43511439 * Life Expectancy - 0.23541908 * Freedom - 0.06950865 * Generosity + 0.29899478 * Corruption$$

$$PC6 = -0.7478164713 * GDP + 0.0813105789 * Social Support + 0.6564896736 * Life Expectancy + 0.0004850166 * Freedom - 0.0298279513 * Generosity + 0.0478589759 * Corruption$$

The plot of the resultant principal components is given in figure 8. It can be observed that for PC1 the variables GDP, Social Security and Life Expectancy have contributed the

most whereas the highest contribution to PC2 is Generosity followed by Corruption and Freedom.

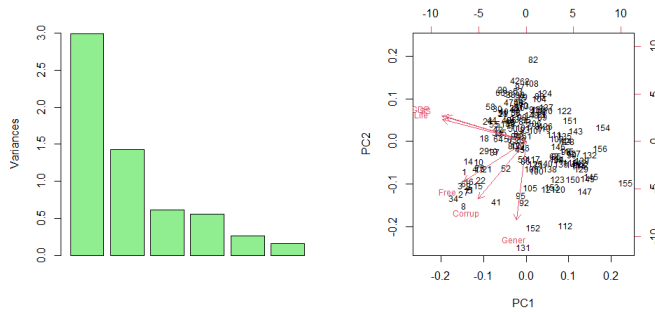


Figure 8. PCA

The proportional variances explained by each principle component have been computed. These values have been plotted in 9. It can be observed that the first principle component explains about 49.8% of the variation in the data, the second principle component covers about 23.7% of the variation, the third principle component covers about 10.1% of the variation and 9% of the variation is covered by the fourth principle component. Therefore, more than 90% of the variation of the data set is covered by the first four principle components.

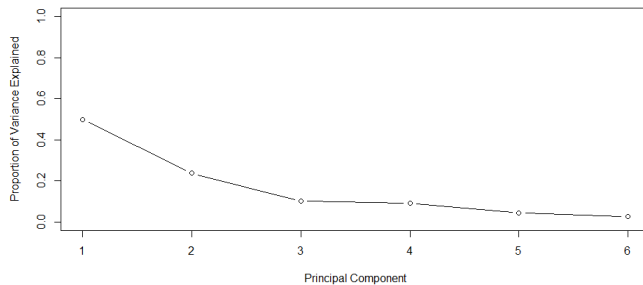


Figure 9. PVE

8. Models Comparison

TABLE 6. MULTIPLE LINEAR REGRESSION COMPARISON

	R squared	F statistic	P Value
Model 1	0.7673	86.16 on 6 and 149 DF	< 2.2e-16
Model 2	0.7625	125.4 on 4 and 151 DF	< 2.2e-16

The models which have been created in this study have given similar result. Upon analysing the p values in the multiple linear regression model, as shown in table 1, it has been deduced that generosity and corruption do not have

significant impact on the happiness score. This deduction has been further solidified by the cross validation approach. Looking at the error comparison as seen in figure 4, it can be concluded that the model created with only GDP, social security, life expectancy and freedom is the best model. The statistical values of the two linear regression models which were created earlier can be seen in table 6.

TABLE 7. DECISION TREE COMPARISON

	Terminal Nodes	Residual Mean Deviance	Error Rate
Model 1	9	0.2226	0.05645
Model 2	7	0.2699	0.06452

In the decision tree model, the misclassification error rate has not been changed by pruning. It has been deemed redundant to apply the pruning on the decision tree as there has been no improvement in the accuracy. The statistical comparison between the decision tree model before and after pruning is given in 7.

Lastly, from Principle Component Analysis, it has been inferred that more than 90% of the variation in the data is explained by the first four principle components.

9. Results and Recommendation

Upon the completion of this data analysis study, it can be concluded that factors such as GDP, social security, life expectancy and freedom have a huge impact on the perception of happiness of people. Objectively, it can be said that if these factors are positive in a country, the people of the country are deemed happier. It has also been proved that corruption and generosity have no significant impact on the happiness score.

It is recommended that factors like unemployment rate, literacy rate and crime rate be taken into account while calculating the happiness score. It would be interesting to see the relationship between these factors and the impact they might have on the people.

This research study has been an extremely fruitful and informative learning experience.

Acknowledgments

The author would like to thank Dr. Liwan Liyanage for the opportunity to conduct this research study. The knowledge provided by Dr. Liyanage along with Lakmini Weiss in Data Science class has been instrumental in the successful completion of this report. The author would also like to acknowledge Western Sydney University for providing the requisite resources for this research study.

References

- [1] J. F. Helliwell, R. Layard, J. D. Sachs, and J.-E. De Neve, "Happinessreport," 2019. [Online]. Available: <https://worldhappiness.report/>

- [2] J. D. Sachs, R. Layard, J. F. Helliwell *et al.*, “World happiness report 2018,” Various institutions, Tech. Rep., 2018.
- [3] J. Benesty, J. Chen, Y. Huang, and I. Cohen, “Pearson correlation coefficient,” in *Noise reduction in speech processing*. Springer, 2009, pp. 1–4.
- [4] R. H. Myers and R. H. Myers, *Classical and modern regression with applications*. Duxbury press Belmont, CA, 1990, vol. 2.
- [5] S. Wold, K. Esbensen, and P. Geladi, “Principal component analysis,” *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

Appendix

Sonali Marasini

29/09/2020

```
#packages
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.2
```

```
##
```

```
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
##      filter, lag
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##      intersect, setdiff, setequal, union
```

```
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.0.2
```

```
library(boot)
```

```
#reading data
```

```
data<-read.csv('C:/Users/sonal/Desktop/WSU/2020 Spring/Data Science/Assignment/2019.csv')
```

```
head(data)
```

```
## Overall.rank Country.or.region Score GDP.per.capita Social.support
## 1          1          Finland 7.769          1.340          1.587
## 2          2          Denmark 7.600          1.383          1.573
## 3          3          Norway 7.554          1.488          1.582
## 4          4          Iceland 7.494          1.380          1.624
## 5          5      Netherlands 7.488          1.396          1.522
## 6          6      Switzerland 7.480          1.452          1.526
## Healthy.life.expectancy Freedom.to.make.life.choices Generosity
## 1          0.986          0.596          0.153
## 2          0.996          0.592          0.252
## 3          1.028          0.603          0.271
## 4          1.026          0.591          0.354
```



```
## 5          0.999          0.557      0.322
## 6          1.052          0.572      0.263
## Perceptions.of.corruption
## 1          0.393
## 2          0.410
## 3          0.341
## 4          0.118
## 5          0.298
## 6          0.343
```

```
#checking if all the independent variables are numeric or not
sapply(data,class)
```

```
##          Overall.rank          Country.or.region
##          "integer"          "character"
##          Score          GDP.per.capita
##          "numeric"          "numeric"
##          Social.support          Healthy.life.expectancy
##          "numeric"          "numeric"
## Freedom.to.make.life.choices          Generosity
##          "numeric"          "numeric"
## Perceptions.of.corruption
##          "numeric"
```

```
#data cleaning
```

```
#replacing 0s with NA
data[data == 0] <- NA
```

```
View(data)
```

```
#replacing NAs with the mean of the column
```

```
data$GDP.per.capita[is.na(data$GDP.per.capita)] <- round(mean(data$GDP.per.capita, na.rm = TRUE))
data$Social.support[is.na(data$Social.support)] <- round(mean(data$Social.support, na.rm = TRUE))
data$Healthy.life.expectancy[is.na(data$Healthy.life.expectancy)] <- round(mean(data$Healthy.life.expectancy, na.rm = TRUE))
data$Freedom.to.make.life.choices[is.na(data$Freedom.to.make.life.choices)] <- round(mean(data$Freedom.to.make.life.choices, na.rm = TRUE))
data$Generosity[is.na(data$Generosity)] <- round(mean(data$Generosity, na.rm = TRUE))
data$Perceptions.of.corruption[is.na(data$Perceptions.of.corruption)] <- round(mean(data$Perceptions.of.corruption, na.rm = TRUE))
```

```
View(data)
```

```
#adding new column
```

```
data_new <- data %>%
  mutate(Score.Level = if_else(data$Score > 5.99, 'H', 'L'))
```

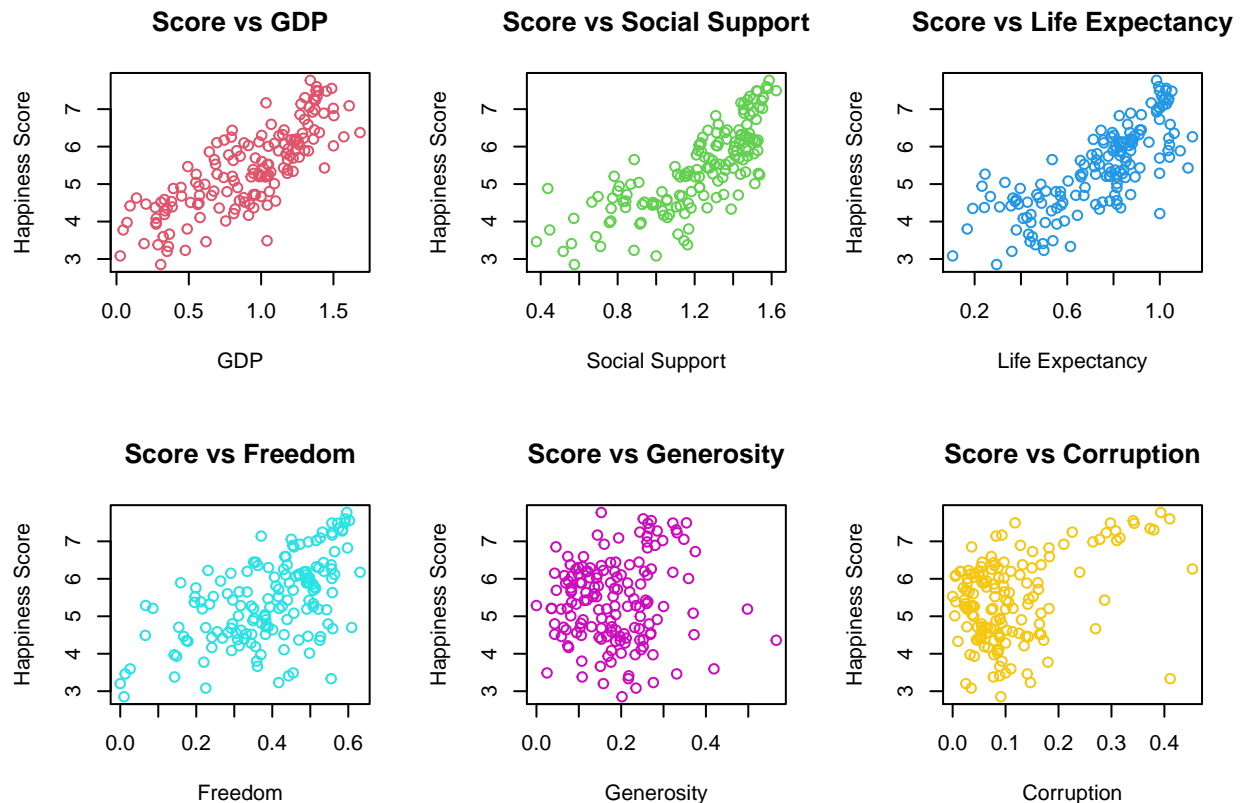
```
View(data_new)
```

```
summary(data_new)
```

```
## Overall.rank Country.or.region Score GDP.per.capita
## Min. : 1.00 Length:156 Min. :2.853 Min. :0.0260
## 1st Qu.: 39.75 Class :character 1st Qu.:4.545 1st Qu.:0.6170
## Median : 78.50 Mode :character Median :5.380 Median :0.9715
```

```
## Mean : 78.50 Mean :5.407 Mean :0.9116
## 3rd Qu.:117.25 3rd Qu.:6.184 3rd Qu.:1.2325
## Max. :156.00 Max. :7.769 Max. :1.6840
## Social.support Healthy.life.expectancy Freedom.to.make.life.choices
## Min. :0.378 Min. :0.1050 Min. :0.0000
## 1st Qu.:1.056 1st Qu.:0.5540 1st Qu.:0.3080
## Median :1.272 Median :0.7920 Median :0.4170
## Mean :1.215 Mean :0.7317 Mean :0.3926
## 3rd Qu.:1.452 3rd Qu.:0.8858 3rd Qu.:0.5072
## Max. :1.624 Max. :1.1410 Max. :0.6310
## Generosity Perceptions.of.corruption Score.Level
## Min. :0.0000 Min. :0.0000 Length:156
## 1st Qu.:0.1087 1st Qu.:0.0470 Class :character
## Median :0.1775 Median :0.0855 Mode :character
## Mean :0.1848 Mean :0.1106
## 3rd Qu.:0.2482 3rd Qu.:0.1412
## Max. :0.5660 Max. :0.4530
```

```
#Relationship
par(mfrow=c(2,3))
plot(data_new$Score~data_new$GDP.per.capita,main='Score vs GDP',xlab='GDP',ylab='Happiness Score',col=2)
plot(data_new$Score~data_new$Social.support,main='Score vs Social Support',xlab='Social Support',ylab='Happiness Score',col=3)
plot(data_new$Score~data_new$Healthy.life.expectancy,main='Score vs Life Expectancy',xlab='Life Expectancy',ylab='Happiness Score',col=4)
plot(data_new$Score~data_new$Freedom.to.make.life.choices,main='Score vs Freedom',xlab='Freedom',ylab='Happiness Score',col=5)
plot(data_new$Score~data_new$Generosity,main='Score vs Generosity',xlab='Generosity',ylab='Happiness Score',col=6)
plot(data_new$Score~data_new$Perceptions.of.corruption,main='Score vs Corruption',xlab='Corruption',ylab='Happiness Score',col=7)
```



```
#Correlation
```

```
cor(data_new$Score,data_new$GDP.per.capita) #0.7964624
```

```
## [1] 0.7964624
```

```
cor(data_new$Score,data_new$Social.support) #0.7727645
```

```
## [1] 0.7727645
```

```
cor(data_new$Score,data_new$Healthy.life.expectancy) #0.7709061
```

```
## [1] 0.7709061
```

```
cor(data_new$Score,data_new$Freedom.to.make.life.choices) #0.5667418
```

```
## [1] 0.5667418
```

```
cor(data_new$Score,data_new$Generosity) #0.07582369
```

```
## [1] 0.07582369
```

```
cor(data_new$Score,data_new$Perceptions.of.corruption) #0.3856131
```

```
## [1] 0.3856131
```

```
#Multiple Linear Regression
```

```
lin.reg=lm(data_new$Score~data_new$GDP.per.capita+data_new$Social.support+data_new$Healthy.life.expectancy+data_new$Freedom.to.make.life.choices+data_new$Generosity+data_new$Perceptions.of.corruption)  
summary(lin.reg)
```

```
##
```

```
## Call:
```

```
## lm(formula = data_new$Score ~ data_new$GDP.per.capita + data_new$Social.support +
```

```
##     data_new$Healthy.life.expectancy + data_new$Freedom.to.make.life.choices +
```

```
##     data_new$Generosity + data_new$Perceptions.of.corruption)
```

```
##
```

```
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
```

```
## -1.73766 -0.36135  0.05866  0.37605  1.29967
```

```
##
```

```
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)      1.6589     0.2214   7.494 5.50e-12 ***
```

```
## data_new$GDP.per.capita      0.8570     0.2173   3.943 0.000123 ***
```

```
## data_new$Social.support      1.2354     0.2439   5.065 1.19e-06 ***
```

```
## data_new$Healthy.life.expectancy      0.9781     0.3460   2.827 0.005350 **
```

```
## data_new$Freedom.to.make.life.choices      1.3963     0.3777   3.697 0.000306 ***
```

```
## data_new$Generosity      0.5910     0.5027   1.176 0.241562
```

```
## data_new$Perceptions.of.corruption      0.8384     0.5543   1.512 0.132554
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.537 on 149 degrees of freedom
## Multiple R-squared:  0.7763, Adjusted R-squared:  0.7673
## F-statistic: 86.16 on 6 and 149 DF,  p-value: < 2.2e-16

#Remove Generosity and Corruptions
lin.reg2=lm(data_new$Score~data_new$GDP.per.capita+data_new$Social.support+data_new$Healthy.life.expectancy)
summary(lin.reg2)

##
## Call:
## lm(formula = data_new$Score ~ data_new$GDP.per.capita + data_new$Social.support +
##     data_new$Healthy.life.expectancy + data_new$Freedom.to.make.life.choices)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.86825 -0.36054  0.02308  0.43481  1.23763
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.7773     0.2092   8.495 1.74e-14 ***
## data_new$GDP.per.capita      0.8924     0.2153   4.146 5.63e-05 ***
## data_new$Social.support      1.1315     0.2400   4.714 5.46e-06 ***
## data_new$Healthy.life.expectancy      1.0197     0.3476   2.934 0.00387 **
## data_new$Freedom.to.make.life.choices      1.7710     0.3428   5.167 7.43e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5425 on 151 degrees of freedom
## Multiple R-squared:  0.7686, Adjusted R-squared:  0.7625
## F-statistic: 125.4 on 4 and 151 DF,  p-value: < 2.2e-16

#overall accuracy
anova(lin.reg2)

## Analysis of Variance Table
##
## Response: data_new$Score
##              Df Sum Sq Mean Sq F value    Pr(>F)
## data_new$GDP.per.capita      1 121.828  121.828 413.9358 < 2.2e-16
## data_new$Social.support      1  15.106   15.106  51.3271 3.216e-11
## data_new$Healthy.life.expectancy      1   2.818    2.818   9.5759 0.00235
## data_new$Freedom.to.make.life.choices      1   7.857    7.857  26.6942 7.430e-07
## Residuals            151  44.442    0.294
##
## data_new$GDP.per.capita      ***
## data_new$Social.support      ***
## data_new$Healthy.life.expectancy      **
## data_new$Freedom.to.make.life.choices ***
## Residuals
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
qf(0.95,4,151) #2.431562
```

```
## [1] 2.431562
```

```
#Residuals  
predict(lin.reg2)
```

```
##      1      2      3      4      5      6      7      8  
## 6.829720 6.855365 7.011361 6.939214 6.750334 6.885441 6.743003 6.784061  
##      9     10     11     12     13     14     15     16  
## 6.792036 6.652365 6.796063 6.300695 6.268623 6.850117 6.519664 6.804719  
##     17     18     19     20     21     22     23     24  
## 6.530830 6.532254 6.399938 6.339746 6.501132 6.674797 5.873927 6.462122  
##     25     26     27     28     29     30     31     32  
## 6.169757 5.930966 5.635256 6.152898 6.636804 6.328071 6.276031 5.809973  
##     33     34     35     36     37     38     39     40  
## 6.272769 6.983658 5.457244 6.084286 6.378034 6.080860 6.140123 6.237419  
##     41     42     43     44     45     46     47     48  
## 6.060576 5.945778 5.941647 6.593807 5.517677 5.597339 6.104585 5.867712  
##     49     50     51     52     53     54     55     56  
## 6.069751 5.842732 6.305352 6.139341 5.789756 5.655588 6.377974 5.906383  
##     57     58     59     60     61     62     63     64  
## 6.058812 6.464613 5.490941 5.999848 5.462659 5.641217 5.911848 6.122046  
##     65     66     67     68     69     70     71     72  
## 5.752143 6.404418 4.483820 5.807755 5.618643 5.608365 5.078669 5.606294  
##     73     74     75     76     77     78     79     80  
## 5.492216 4.880687 5.696704 6.428809 5.942834 5.229080 5.541102 5.935866  
##     81     82     83     84     85     86     87     88  
## 5.607857 5.276558 5.597163 5.584177 4.659786 5.533004 5.824062 4.936786  
##     89     90     91     92     93     94     95     96  
## 4.914607 5.411659 5.256633 5.511869 5.801775 5.790962 5.422770 4.309166  
##     97     98     99    100    101    102    103    104  
## 5.845584 4.475017 4.059296 5.030321 5.419657 3.645372 4.458766 5.163822  
##    105    106    107    108    109    110    111    112  
## 5.155119 5.329819 5.151401 5.342238 5.287151 4.858293 4.561375 4.722755  
##    113    114    115    116    117    118    119    120  
## 5.243946 3.712618 4.106642 5.061804 5.052265 4.024784 4.701107 4.227587  
##    121    122    123    124    125    126    127    128  
## 4.702208 4.221952 4.347560 4.857499 4.999399 4.829058 3.974558 4.264371  
##    129    130    131    132    133    134    135    136  
## 3.762067 5.735255 5.242904 3.460317 5.150642 4.397683 5.375127 4.365350  
##    137    138    139    140    141    142    143    144  
## 4.851180 4.687924 3.606903 4.798186 3.992679 3.644790 3.886308 4.343518  
##    145    146    147    148    149    150    151    152  
## 3.101233 4.445268 3.347910 5.356253 3.229094 3.870680 4.074730 4.511149  
##    153    154    155    156  
## 4.450789 3.042742 3.437561 3.019517
```

```
resid(lin.reg2)
```

```
##      1      2      3      4      5      6  
## 0.939279684 0.744634810 0.542639206 0.554786484 0.737665677 0.594558880
```

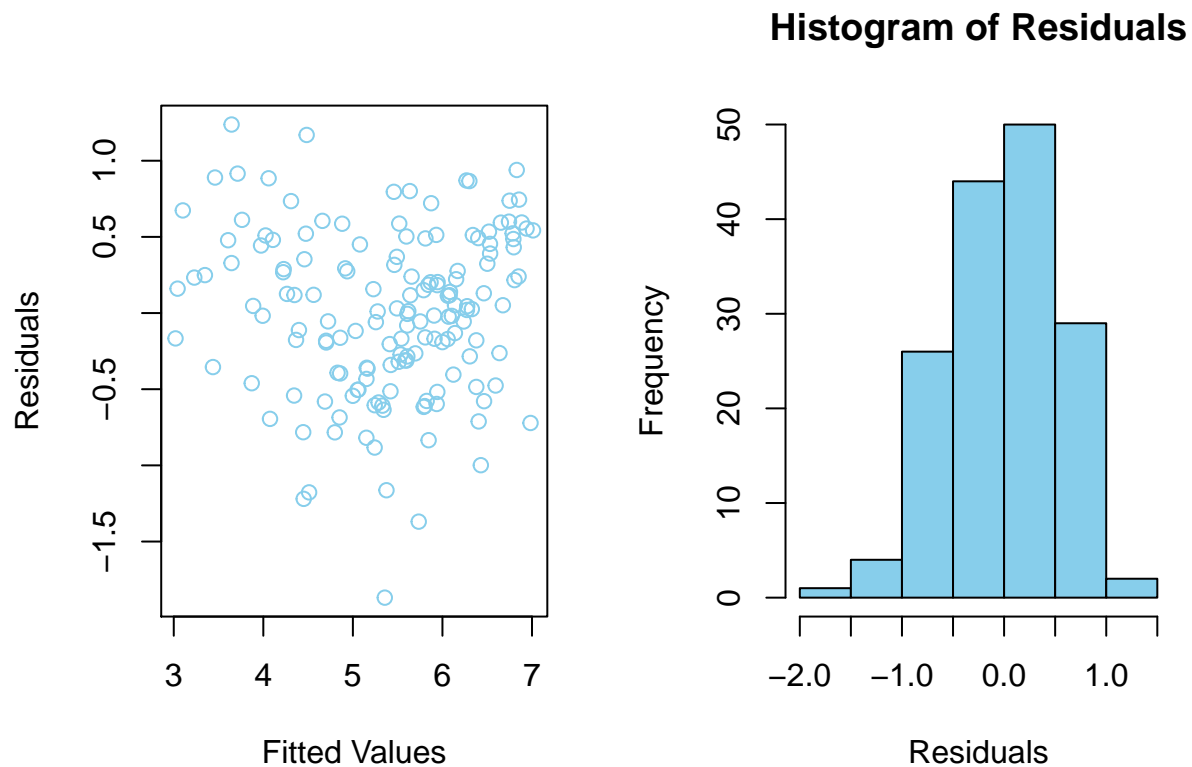
##	7	8	9	10	11	12
##	0.599997086	0.522938619	0.485963548	0.593635004	0.431937056	0.866304851
##	13	14	15	16	17	18
##	0.870377005	0.239883156	0.534336183	0.216281148	0.454169832	0.390746310
##	19	20	21	22	23	24
##	0.492061511	0.512253707	0.323867733	0.051202714	0.721072556	0.129878200
##	25	26	27	28	29	30
##	0.276242855	0.513033738	0.800743795	0.222101632	-0.262804073	0.025929470
##	31	32	33	34	35	36
##	0.044969224	0.490027429	0.020231165	-0.721657903	0.795756266	0.138713762
##	37	38	39	40	41	42
##	-0.179034332	0.117139537	0.051876733	-0.055418732	0.113423542	0.203222054
##	43	44	45	46	47	48
##	0.183352587	-0.475806885	0.587322617	0.502661472	-0.018584917	0.202287893
##	49	50	51	52	53	54
##	-0.023751068	0.185268211	-0.284352303	-0.131341477	0.150244368	0.239412086
##	55	56	57	58	59	60
##	-0.484974313	-0.016382736	-0.170812235	-0.578612759	0.369059431	-0.190848067
##	61	62	63	64	65	66
##	0.316341488	0.116782628	-0.168848147	-0.404045560	-0.055142849	-0.711418431
##	67	68	69	70	71	72
##	1.169180279	-0.159754598	0.012357257	-0.005364686	0.450331274	-0.081293817
##	73	74	75	76	77	78
##	0.030784239	0.586312555	-0.264703838	-0.998808563	-0.517833909	0.156920357
##	79	80	81	82	83	84
##	-0.168101699	-0.596865648	-0.284856872	0.010442425	-0.312162830	-0.310176789
##	85	86	87	88	89	90
##	0.605213622	-0.272004324	-0.577061959	0.274213574	0.293393080	-0.203659090
##	91	92	93	94	95	96
##	-0.059633119	-0.319868738	-0.610774505	-0.615961705	-0.340769567	0.734834370
##	97	98	99	100	101	102
##	-0.834584392	0.520982809	0.884703936	-0.117320673	-0.513656609	1.237627716
##	103	104	105	106	107	108
##	0.353234335	-0.364821685	-0.359118628	-0.607819223	-0.432401401	-0.635237741
##	109	110	111	112	113	114
##	-0.587151008	-0.162293036	0.119624760	-0.054755420	-0.604945987	0.915381520
##	115	116	117	118	119	120
##	0.480357997	-0.502803840	-0.504265339	0.509216101	-0.182106541	0.288412980
##	121	122	123	124	125	126
##	-0.193208274	0.268047898	0.118439744	-0.396498549	-0.543399401	-0.392057934
##	127	128	129	130	131	132
##	0.443442046	0.125628915	0.611933260	-1.369255453	-0.882903902	0.889683341
##	133	134	135	136	137	138
##	-0.818641672	-0.111682922	-1.163126590	-0.176350411	-0.685179968	-0.580924309
##	139	140	141	142	143	144
##	0.478096744	-0.783186306	-0.017679366	0.328210125	0.046691717	-0.541517819
##	145	146	147	148	149	150
##	0.673766708	-0.782268458	0.249089897	-1.868253214	0.232905640	-0.460679586
##	151	152	153	154	155	156
##	-0.694729657	-1.177149034	-1.219788767	0.160257709	-0.354560879	-0.166516830

```
par(mfrow=c(1,2))
```

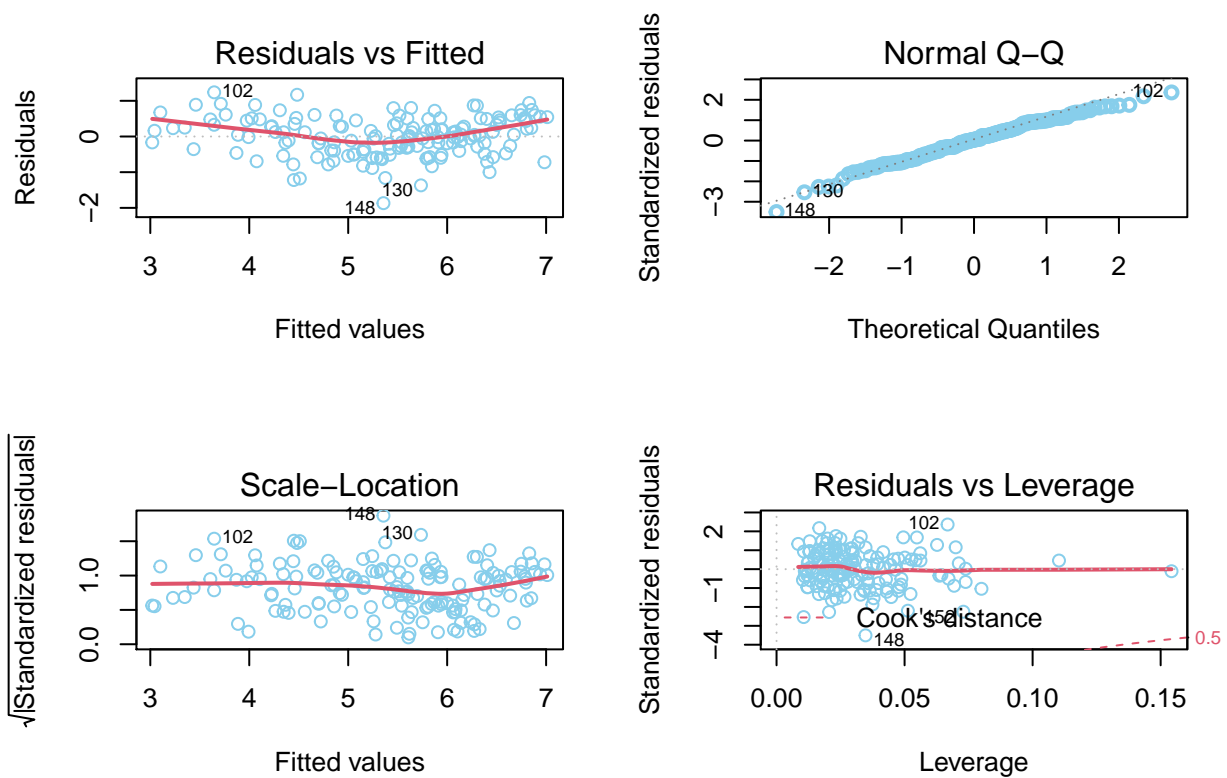
```
#plotting the residuals against the predicted values
```

```
plot(predict(lin.reg2),resid(lin.reg2), xlab = "Fitted Values", ylab = "Residuals",col='skyblue')

#histogram of the residuals
hist(resid(lin.reg2), main = paste("Histogram of Residuals"), xlab = "Residuals",col='skyblue')
```



```
#residual plots
par(mfrow=c(2,2))
plot(lin.reg2,lwd=2,col='skyblue',pch=1)
```

```
#####
```

```
#Cross Validation
```

```
data_cv=data_new[,3:9]
```

```
m1=glm(Score~GDP.per.capita,data=data_cv)
```

```
m2=glm(Score~GDP.per.capita+Social.support,data=data_cv)
```

```
m3=glm(Score~GDP.per.capita+Social.support+Healthy.life.expectancy,data=data_cv)
```

```
m4=glm(Score~GDP.per.capita+Social.support+Healthy.life.expectancy+Freedom.to.make.life.choices,data=data_cv)
```

```
m5=glm(Score~GDP.per.capita+Social.support+Healthy.life.expectancy+Freedom.to.make.life.choices+Generosity,data=data_cv)
```

```
m6=glm(Score~GDP.per.capita+Social.support+Healthy.life.expectancy+Freedom.to.make.life.choices+Generosity,data=data_cv)
```

```
#training, testing, cv dataset
```

```
#Validation-set approach 60/40
```

```
#set.seed(2)
```

```
tr.id <- sample(1:nrow(data_cv),nrow(data_cv)*.6)
```

```
#train dataset
```

```
train_cv <- data_cv[tr.id , ]
```

```
#validation dataset
```

```
validation_cv <- data_cv[-tr.id , ]
```

```
dim(train_cv) # 93 7
```

```
## [1] 93 7
```

```

dim(validation_cv) #63 7

## [1] 63 7

#error estimates

#create initial error vector with values 0's
train_error <- rep(0,6)
#create initial error vector with values 0's
validation_error <- rep(0,6)

m1t = glm(m1, subset = tr.id) #fit model1 for training data
score_hat1 <- predict (m1t ,data_cv) #predict happiness score
train_error[1] <- mean((data_cv$Score-score_hat1)[tr.id]^2) #calculate training error
validation_error[1] <- mean((data_cv$Score-score_hat1)[-tr.id]^2) #calculate validation error

m2t = glm(m2, subset = tr.id) #fit model1 for training data
score_hat2 <- predict (m2t ,data_cv) #predict happiness score
train_error[2] <- mean((data_cv$Score-score_hat2)[tr.id]^2) #calculate training error
validation_error[2] <- mean((data_cv$Score-score_hat2)[-tr.id]^2) #calculate validation error

m3t = glm(m3, subset = tr.id) #fit model3 for training data
score_hat3 <- predict (m3t ,data_cv) #predict happiness score
train_error[3] <- mean((data_cv$Score-score_hat3)[tr.id]^2) #calculate training error
validation_error[3] <- mean((data_cv$Score-score_hat3)[-tr.id]^2) #calculate validation error

m4t = glm(m4, subset = tr.id) #fit model4 for training data
score_hat4 <- predict (m4t ,data_cv) #predict happiness score
train_error[4] <- mean((data_cv$Score-score_hat4)[tr.id]^2) #calculate training error
validation_error[4] <- mean((data_cv$Score-score_hat4)[-tr.id]^2) #calculate validation error

m5t = glm(m5, subset = tr.id) #fit model5 for training data
score_hat5 <- predict (m5t ,data_cv) #predict happiness score
train_error[5] <- mean((data_cv$Score-score_hat5)[tr.id]^2) #calculate training error
validation_error[5] <- mean((data_cv$Score-score_hat5)[-tr.id]^2) #calculate validation error

m6t = glm(m6, subset = tr.id) #fit model6 for training data
score_hat6 <- predict (m6t ,data_cv) #predict happiness score
train_error[6] <- mean((data_cv$Score-score_hat6)[tr.id]^2) #calculate training error
validation_error[6] <- mean((data_cv$Score-score_hat6)[-tr.id]^2) #calculate validation error

#loocv
loocv_error=rep(0,6)
loocv_error[1]=cv.glm(data_cv,m1)$delta[1]
loocv_error[2]=cv.glm(data_cv,m2)$delta[1]
loocv_error[3]=cv.glm(data_cv,m3)$delta[1]
loocv_error[4]=cv.glm(data_cv,m4)$delta[1]
loocv_error[5]=cv.glm(data_cv,m5)$delta[1]
loocv_error[6]=cv.glm(data_cv,m6)$delta[1]

#kfold
cv_errorKF= rep (0,6)

```

```

#set.seed(2) #seed can be any value
cv_errorKF[1] <- cv.glm(data_cv,m1, K=10)$delta[1]
cv_errorKF[2] <- cv.glm(data_cv,m2, K=10)$delta[1]
cv_errorKF[3] <- cv.glm(data_cv,m3, K=10)$delta[1]
cv_errorKF[4] <- cv.glm(data_cv,m4, K=10)$delta[1]
cv_errorKF[5] <- cv.glm(data_cv,m5, K=10)$delta[1]
cv_errorKF[6] <- cv.glm(data_cv,m6, K=10)$delta[1]

#mse values
train_error

## [1] 0.4039207 0.3323384 0.3153765 0.2671776 0.2629036 0.2466433

validation_error

## [1] 0.5358354 0.3901411 0.3740266 0.3146881 0.3081892 0.3325908

cv_errorKF

## [1] 0.4584381 0.3653040 0.3537523 0.3062573 0.3099824 0.3022022

loocv_error

## [1] 0.4610816 0.3671266 0.3543573 0.3047021 0.3044499 0.3065170

#plotting
model <- c(1:6)
plot(model, train_error, type = "b", xlab = "Model", ylab = "MSE", col = 2, main = "Comparison",ylim=c(
{ lines(validation_error, type = "b", col = 3)
  lines(cv_errorKF, type = "b", col = 9)
  lines(loocv_error, type = "b", col = 5)
}
cols=c(2,3,9,5)
legend(x = 4, y = 0.6,cex=0.5, c("Train", "Validation", "LOOCV", "10-fold"), fill=cols)

#####

#Decision Tree

#keeping only the decision variables and score level
data_tree=data_new[,c(-1:-3)]
View(data_tree)

#converting character variable to factor variable
data_tree$Score.Level = as.factor(data_tree$Score.Level)
sapply(data_tree,class)

##                GDP.per.capita                Social.support

```

```
##           "numeric"           "numeric"
##   Healthy.life.expectancy Freedom.to.make.life.choices
##           "numeric"           "numeric"
##           Generosity    Perceptions.of.corruption
##           "numeric"           "numeric"
##           Score.Level
##           "factor"
```

```
#fitting a tree on training data
```

```
set.seed(2)
train.dt = sample(1:nrow(data_tree), nrow(data_tree)/1.25)
test.dt=data_tree[-train.dt,]
length(train.dt) #124
```

```
## [1] 124
```

```
length(test.dt) #7
```

```
## [1] 7
```

```
tree.data=tree(Score.Level~.,data_tree,subset=train.dt)
summary(tree.data)
```

```
##
## Classification tree:
## tree(formula = Score.Level ~ ., data = data_tree, subset = train.dt)
## Variables actually used in tree construction:
## [1] "GDP.per.capita"          "Social.support"
## [3] "Freedom.to.make.life.choices" "Healthy.life.expectancy"
## [5] "Generosity"
## Number of terminal nodes: 9
## Residual mean deviance: 0.2226 = 25.6 / 115
## Misclassification error rate: 0.05645 = 7 / 124
```

```
#0.05645 = 7 / 124 misclassified
0.05645*100
```

```
## [1] 5.645
```

```
#plotting
```

```
plot(tree.data)
text(tree.data, pretty=0, cex=0.6)
```

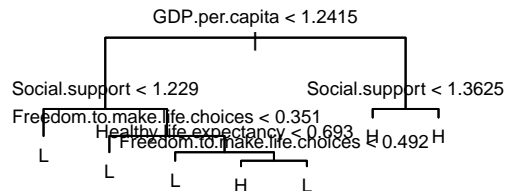
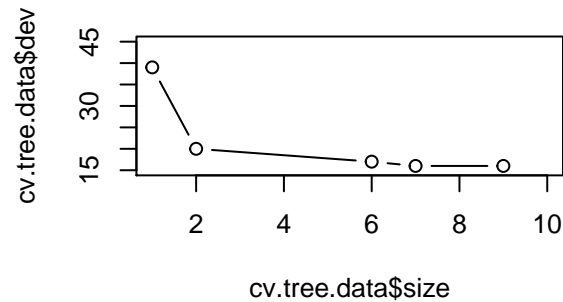
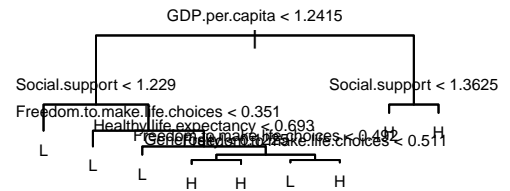
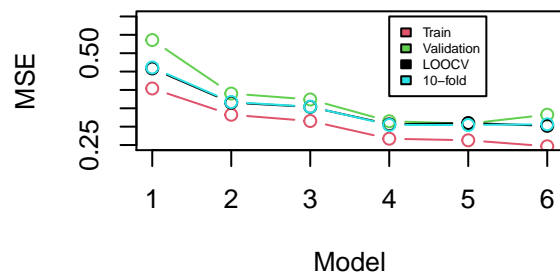
```
#cross validation
```

```
cv.tree.data=cv.tree(tree.data,FUN = prune.misclass)
plot(cv.tree.data$size, cv.tree.data$dev,type="b",xlim=c(1,10),ylim=c(15,45))
#6
```

```
#pruning
```

```
prune.tree.data=prune.tree(tree.data,best=7)
plot(prune.tree.data)
text(prune.tree.data, pretty=0, cex=0.75)
```

Comparison



```
summary(prune.tree.data)
```

```
##
## Classification tree:
## snip.tree(tree = tree.data, nodes = 47:46)
## Variables actually used in tree construction:
## [1] "GDP.per.capita" "Social.support"
## [3] "Freedom.to.make.life.choices" "Healthy.life.expectancy"
## Number of terminal nodes: 7
## Residual mean deviance: 0.2699 = 31.58 / 117
## Misclassification error rate: 0.06452 = 8 / 124
```

```
#0.1129 = 14 / 124
```

```
#testing model accuracy
```

```
tree_predicted<-predict(tree.data, test.dt, type="class")
```

```
#confusion matrix
```

```
table(tree_predicted,test.dt$Score.Level)
```

```
##
## tree_predicted H L
##               H 12 4
##               L 1 15
```

```
#calculating misclassification rate
matrix<-table(tree_predicted,test.dt$Score.Level)
misrate<-((matrix[1,2]+matrix[2,1])/sum(matrix))
paste("Misclassification error rate is ",misrate)
```

```
## [1] "Misclassification error rate is 0.15625"
```

```
# 0.15625
```

```
#####
```

```
#Principle Component Analysis
```

```
#keeping only the independent variables
```

```
pca.data=data_new[,4:10]
head(pca.data)
```

```
## GDP.per.capita Social.support Healthy.life.expectancy
## 1 1.340 1.587 0.986
## 2 1.383 1.573 0.996
## 3 1.488 1.582 1.028
## 4 1.380 1.624 1.026
## 5 1.396 1.522 0.999
## 6 1.452 1.526 1.052
## Freedom.to.make.life.choices Generosity Perceptions.of.corruption Score.Level
## 1 0.596 0.153 0.393 H
## 2 0.592 0.252 0.410 H
## 3 0.603 0.271 0.341 H
## 4 0.591 0.354 0.118 H
## 5 0.557 0.322 0.298 H
## 6 0.572 0.263 0.343 H
```

```
#renaming for simplification purpose
```

```
colnames(pca.data)[c(1:6)]=c("GDP","SS","Life","Free","Gener","Corrup")
```

```
#mean and variance
```

```
sapply(pca.data[,1:6],mean)
```

```
## GDP SS Life Free Gener Corrup
## 0.9115577 1.2152244 0.7316538 0.3925705 0.1848462 0.1106026
```

```
sapply(pca.data[,1:6],var)
```

```
## GDP SS Life Free Gener Corrup
## 0.153445100 0.080328149 0.055676305 0.020531872 0.009073408 0.008937402
```

```
#PCA
```

```
obj = prcomp(pca.data[,1:6], scale. = TRUE)
names(obj)
```

```
## [1] "sdev" "rotation" "center" "scale" "x"
```

```
obj$rotation
```

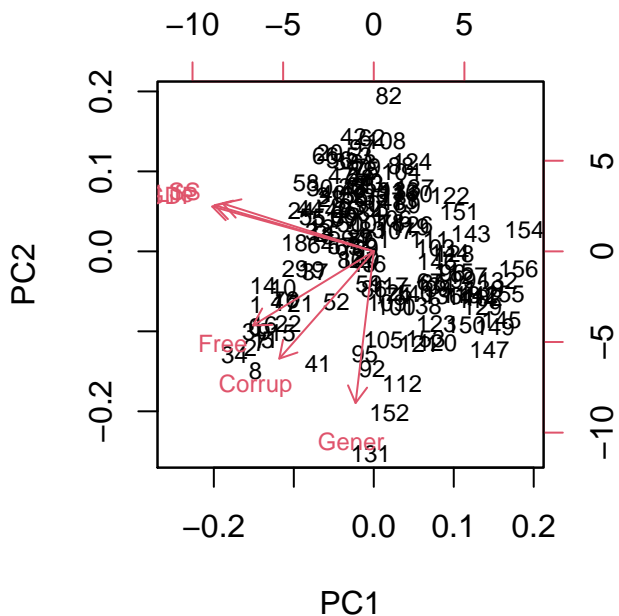
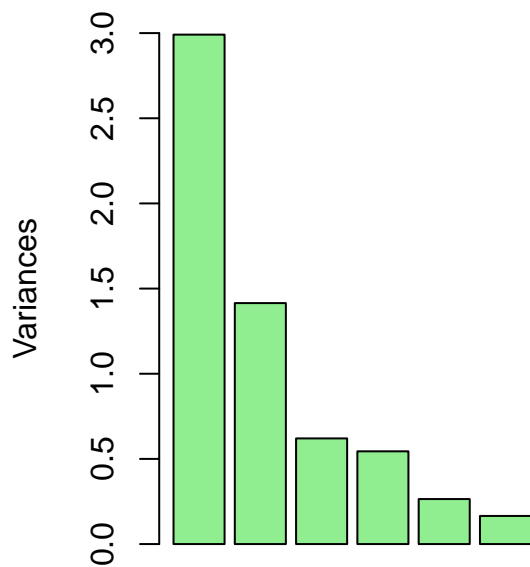
	PC1	PC2	PC3	PC4	PC5	PC6
## GDP	-0.51583025	0.2082045	-0.07765971	0.22026792	-0.27711289	-0.7478164713
## SS	-0.48232080	0.2223429	0.35572318	0.02339788	0.76435099	0.0813105789
## Life	-0.50821063	0.2141943	-0.05577799	0.26912517	-0.43511439	0.6564896736
## Free	-0.38551214	-0.3435152	0.24935585	-0.78471459	-0.23541908	0.0004850166
## Gener	-0.05806569	-0.7039110	0.48379134	0.51123624	-0.06950865	-0.0298279513
## Corrup	-0.30168944	-0.4978373	-0.75371507	0.03696948	0.29899478	0.0478589759

```
obj
```

```
## Standard deviations (1, ..., p=6):  
## [1] 1.7294757 1.1894226 0.7876006 0.7379261 0.5139982 0.4063791  
##  
## Rotation (n x k) = (6 x 6):  
##
```

	PC1	PC2	PC3	PC4	PC5	PC6
## GDP	-0.51583025	0.2082045	-0.07765971	0.22026792	-0.27711289	-0.7478164713
## SS	-0.48232080	0.2223429	0.35572318	0.02339788	0.76435099	0.0813105789
## Life	-0.50821063	0.2141943	-0.05577799	0.26912517	-0.43511439	0.6564896736
## Free	-0.38551214	-0.3435152	0.24935585	-0.78471459	-0.23541908	0.0004850166
## Gener	-0.05806569	-0.7039110	0.48379134	0.51123624	-0.06950865	-0.0298279513
## Corrup	-0.30168944	-0.4978373	-0.75371507	0.03696948	0.29899478	0.0478589759

```
par(mfrow=c(1,2))  
  
#screeplot  
screeplot(obj,col='lightgreen',main='',ylim=c(0,3))  
  
#biplot  
biplot(obj,cex=0.75)
```

```
#proportion of variance
```

```
#sd of each pc
```

```
obj$sdev
```

```
## [1] 1.7294757 1.1894226 0.7876006 0.7379261 0.5139982 0.4063791
```

```
#calculate var of each pc
```

```
obj.var = obj$sdev^2
```

```
#proportional variance explained by each pc
```

```
pve = obj.var/sum(obj.var)
```

```
#plot pve
```

```
plot(pve,xlab="Principal Component",ylab="Proportion of Variance Explained",
     ylim = c(0,1), type = 'b')
```

