Week 4 Assignment

ALY 6015 Intermediate Analytics

Submitted to:

<u>Joseph Manseau</u>

Date :03/18/2020

Submitted by:

Deepak Natarajan (001088182)

Ashlesha Kshirsagar (001082234)

Introduction:

Logistic regression is used when a linear regression fails on a dependent variable with binary data such as 0,1 or yes, no. logistic regression adapts the linear regression formula but works as a classifier. ("The Basics: Logistic Regression and Regularization", 2020). Decision tree is a type of supervised learning algorithm that can be used in both regression and classification problems. It works for both categorical and continuous input and output variables. Decision trees are an extremely effective method for predicting outcomes within a dataset. They consist of nodes representing certain variables that are split into branches indicating the possible values of that variable. By traversing the tree from the root to the terminal nodes (or leaves), we can predict the value of the dependent variable and see the impact that other variables have towards its outcome. Some of the benefits to using decision trees are that they are computationally cheap to build, easy to understand, and can handle issues such as missing values and irrelevant data (Bati, 2015). Some of the strengths of Decision tree model is Fast to Evaluate, Easily Interpretable and its supports both numeric and categorical variables. Weakness of Decision tree model includes Overfitting, Accuracy is not always high and splitting methods might not be optimal. In this assignment we have used housing data from source to predict the house price data.

Data Exploration

The Housing data has 10 columns and 20640 rows. Obtained from the source

https://www.kaggle.com/harrywang/housing#housing.csv

Data Attributes

- longitude
- latitude

- housing_median_age
- total_rooms
- total_bedrooms
- population
- households
- median_income
- median_house_value
- ocean_proximity

Analysis of the data is meaningful only if the data quality is good. This step is performed to clean and manipulate the data in order to extract the valuable information that can be used further for analysis and predictions.

Here, we perform

- Loading of the data, installation of the required packages and libraries.
- Cleaning of the data by determining missing values, filtering out the groups based on requirements and combining the data frames.
- Identifying the valuable records, correlation and extract them to interpret the results and predictions.

<u>Code</u>

```
library(rpart)
library(corrplot)
library(rpart.plot)
library(glmnet)
library(caret)

data <- read.csv('housing.csv')
nrow(data)
```

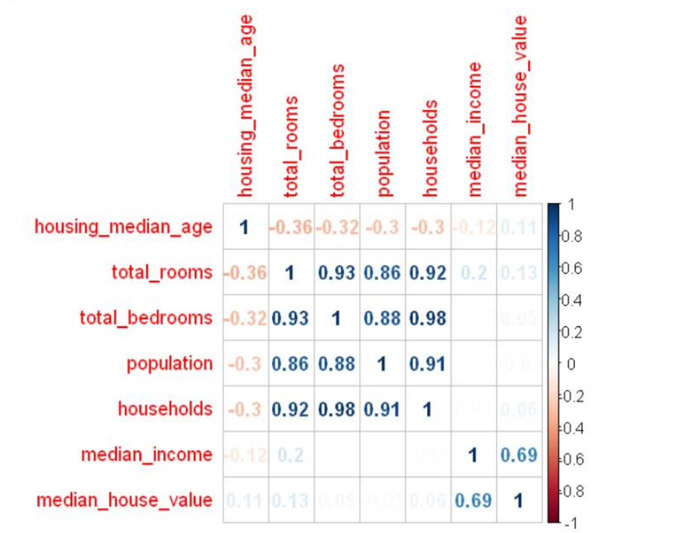
```
colSums(is.na(data)) #checking the null values
data <- na.omit(data) # removing null values
rownames(data) <- 1:nrow(data)
nrow(data)
head(data)
str(data) # structure of data

# removing the columns that are not useful for prediction
data$longitude <- NULL
data$latitude <- NULL
data$ocean_proximity <- NULL
str(data)

summary(data)
scaled_data <- as.data.frame(scale(data))
```

Correlation Plot

Code

```
#correlation plot
numeric_var <- which(sapply(scaled_data, is.numeric))
num_data <- scaled_data[, numeric_var]
correlation <- cor(num_data)
options(repr.plot.width = 5, repr.plot.height = 5)
corrplot(correlation, method = 'number')
```

```
hist(data$median_house_value)
```



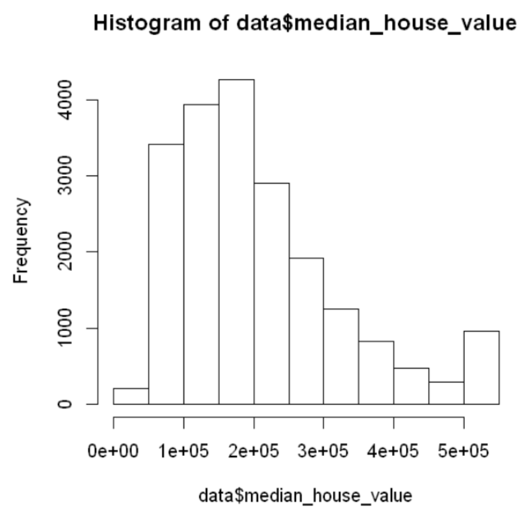


Figure 1.1 Correlation pattern between different value independent variables

Figure 1.2 Histogram of Median house

Interpretation

We have used correlation matrix to check the co relation amongst the variable. From figure 1.1 there exists a multicollinearity amongst the independent variables. There exists the correlation more than 90% for many variables such as Household, total number of rooms, number of bedrooms available, population. Figure shows the histogram of median house value. The data for median house value is left skewed.

Data splitting

Code

```
set.seed(42)
data_split <- sample(3, nrow(scaled_data), replace = T, prob = c(0.8,0.1,0.1))
train_data <- scaled_data[data_split == 1,]
val_data <- scaled_data[data_split == 2,]
test_data <- scaled_data[data_split == 3,]
nrow(train_data)
nrow(val_data)
nrow(test_data)
custom_parameter <- trainControl(method = 'repeatedcv',
                                 number = 10,
                                 repeats = 5,
                                 verboseIter = T)
```

Interpretation

In order to begin the process of creating decision tree classification model, we split the data into three parts 80% Training dataset, 10% validation data and 10 % of Testing data. This is necessary because it prevents overfitting, which occurs when we include too many branches in the tree including outliers and branches with unnecessary information. Overfitting is a problem because it impacts the accuracy of our decision tree model (Han, Kamber, & Pei, 2011). But by splitting the data, we reduce the amount of data included in the tree therefore lowering the chance of overfitting

Feature Selection (Lasso regression)

Lasso Regression method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces.

Code:

```
set.seed(42)
lasso_reg <- train(median_house_value ~ ., train_data, method = 'glmnet',
                   tuneGrid = expand.grid(alpha = 1, lambda = seq(0,0.001,length = 10)),
                   trControl = custom_parameter)

options(repr.plot.width = 5, repr.plot.height = 3)
plot(lasso_reg)

options(repr.plot.width = 7, repr.plot.height = 4)
plot(varImp(lasso_reg))
```

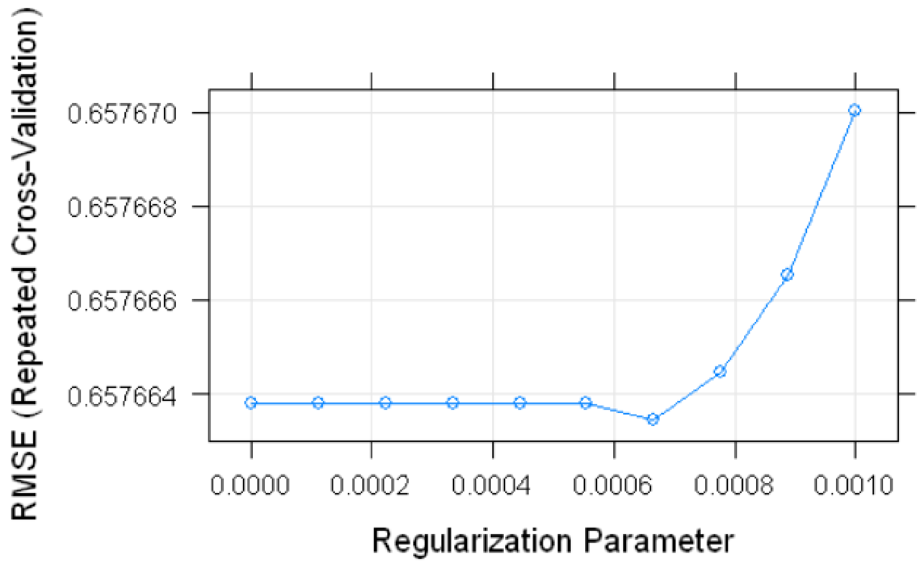


Figure 1.3 Root means square error (cross validation) Vs Regularization parameter

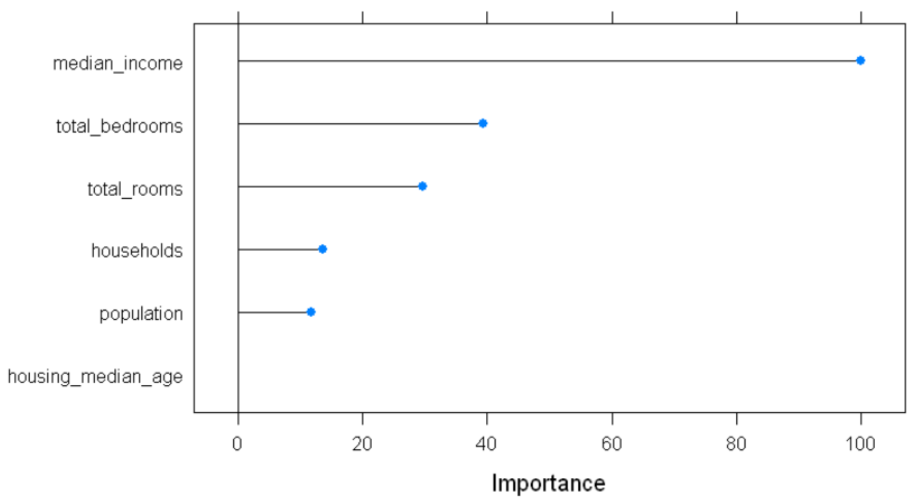


Figure 1.4 variable importance based on lasso regression

Interpretation

We can see that from figure 1. In the beginning, cutting coefficient reduces the overfitting and the generalization abilities of the model. Hence, the test error is slowly decreasing. However, as we are cutting more and more coefficient, the test error start increasing. After tuning different ranges for lambda, we got optimum range 0.0006 to 0.0008. Also, from fig We can see the summary of the model and understand which features are significant. In this GLM median income is significant.

Decision Tree

Code

```
tree <- rpart(median_house_value ~ median_income + total_bedrooms + total_rooms + households + population,
              train_data, control = rpart.control(minsplit = 200, maxdepth = 30, cp = 0.024))

options(repr.plot.width = 8, repr.plot.height = 5)
rpart.plot(tree,extra = 1)
```

```
summary(tree)
```

```
Call:
rpart(formula = median_house_value ~ median_income + total_bedrooms +
    total_rooms + households + population, data = train_data,
    control = rpart.control(minsplit = 500, maxdepth = 10))
  n= 10209

          CP nsplit rel error    xerror       xstd
1 0.30618690      0 1.0000000 1.0002491 0.01523436
2 0.08048765      1 0.6938131 0.6967742 0.01179180
3 0.05957659      2 0.6133254 0.6180093 0.01096074
4 0.01431244      3 0.5537489 0.5607733 0.01086676
5 0.01097065      4 0.5394364 0.5479633 0.01081077
6 0.01000000      5 0.5284658 0.5374907 0.01072570

Variable importance
median_income   total_rooms    households
           96             2             1
```

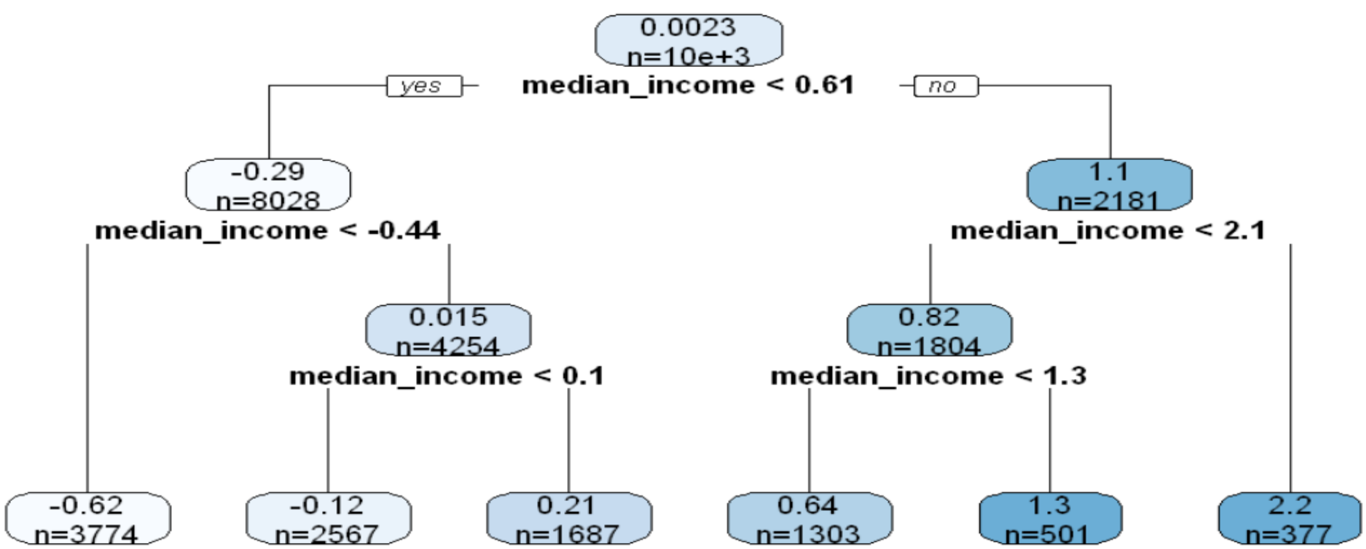


Figure 1.4 Decision tree on housing data

```
tree$variable.importance
```

median_income	4787.20651182616
total_rooms	112.367198457045
households	25.05523667505
total_bedrooms	19.4952337065122
population	16.9325875964869

Figure 1.5 Variable Importance

Observation

From the Figure 1.5 we can see that median income feature has highest variable importance value of **4787.2** whereas importance value of other variables is **very low**. This indicates median income is good enough in predicting the median house price. We created the Decision tree to predict median house price using only five We can see same in the Figure 1.4 that median house price is predicted only using **median income** of people living in that area.

Conclusion

Validation

```
val_pred = predict(tree, val_data)
tree.sse = sum((val_pred - val_data$median_house_value)^2)
rmse = sqrt(tree.sse)
rmse
```

34.0949363242896

Prediction (test data)

```
test_pred = predict(tree, test_data)
tree.sse1 = sum((test_pred - test_data$median_house_value)^2)
rmse1 = sqrt(tree.sse1)
rmse1
```

33.1564657985118

We first tested the model accuracy with the validation data which is 10% of original data and we got the Root Mean square error (RMSE) of **34.09**. After the tested the model with test data which is also 10% of original data and we got the RMSE of **33.16**.

Reference

1. The Basics: Logistic Regression and Regularization. (2020). Retrieved 2 February 2020, from https://towardsdatascience.com/the-basics-logistic-regression-and-regularization-828b0d2d206c
2. Bati, F. (2015, Fall). Classification using Decision Tree. Lecture presented at UMUC. Retrieved June 28, 2017.