**DOKUZ EYLUL UNIVERSITY**

**ENGINEERING DEPARTMENT**

**DEPARTMENT OF COMPUTER ENGINEERING**

**CME4403 – INTRODUCTION TO MACHINE LEARNING**

**TERM PROJECT**

**KICKSTARTER PROJECTS**

**2016510036 – Türkü Su KAPLAN**

**2016510082 – Mustafa Çağatay ÖNAL**

**January 2020**

**IZMIR**

**TABLE OF CONTENTS**

**LIST OF TABLES……..…………………………………………………………..………...iii**

**LIST OF FIGURES………………………………………………………………………….iii**

**1. INTRODUCTION1**

1.1. Dataset Description2

1.2. Summarize & Analyze Dataset2

1.3. Visualizing Data7

1.3.1. Plot Distribution of Features7

1.3.2. Correlation Matrix7

**2. DATA PREPRARATION1**

**3. MODEL TRAINING AND EVALUATION7**

3.1. Support Vector Machine9

2.1.2. 10-Fold Cross Validation8

2.1.2. Train %70 – Test %108

3.2. Decision Tree Classifier9

2.2.2. Train %90 – Test %109

2.2.2. 10-Fold Cross Validation10

3.3. K-Nearest Neighbour11

2.3.2. Train %90 – Test %1011

2.3.2. 10-Fold Cross Validation12

**4. CONCLUSION20**

**APPENDIX A. CODE20**

**REFERENCES30**

**LIST OF FIGURES**

[Figure 1: Target Feature Distribution Chart 7](#_Toc61906259)

[Figure 2: Main Category Distribution 7](#_Toc61906260)

[Figure 3: Projects by Country Distribution 8](#_Toc61906261)

[Figure 4: Projects Launched by Year Distribution 8](#_Toc61906262)

[Figure 5: Plot Distributions of the Features 9](#_Toc61906263)

[Figure 6: Correlation Matrix 10](#_Toc61906264)

[Figure 7: Categories and Rates of Success 12](#_Toc61906265)

[Figure 8: Cross Validation-Accuracy Plot 13](#_Toc61906266)

[Figure 9: Feature Importance Plot 13](#_Toc61906267)

[Figure 10: Removing Outliers - Before and After Boxplots 14](#_Toc61906268)

[Figure 11: Accuracy Rate of Raw Dataset (SVM) 15](#_Toc61906269)

[Figure 12: Accuracy Rate of Preprocessed Data (SVM) 16](#_Toc61906270)

[Figure 13: Decision Tree Cross Validation Accuracy Boxplot 17](#_Toc61906271)

[Figure 14: Decision Tree Cross Validation Accuracy Plot 18](#_Toc61906272)

[Figure 15: Decision Tree 19](#_Toc61906273)

[Figure 16: KNN Accuracy Box Plot 20](#_Toc61906274)

[Figure 17: Model Comparison Plot 21](file:///C:\Users\HP\Desktop\MachineLearning\Term%20Project\termproject.docx#_Toc61906275)

**LIST OF TABLES**

[Table 1: Target Feature Category Counts 6](#_Toc61906298)

[Table 2: Feature Distributions and Their Common Attributes 7](#_Toc61906299)

[Table 3: Confusion Matrix of SVM – Raw Dataset 16](#_Toc61906300)

[Table 4: Evaluation Metrics - SVM 17](#_Toc61906301)

[Table 5: Confusion Matrix of Decision Tree Model 20](#_Toc61906302)

**CHAPTER 1**

**INTRODUCTION**

* 1. **Dataset Description**

Kickstarter projects dataset is a dataset collected from the website of the corporation known as Kickstarter where original and creative projects that need a starter funding are published. It is a global crowdfunding platform for gathering the necessary starting money for a project to come to life. For funds to be collected, the project creators must meet the project deadline. Backers pledge to the projects that they want to come to life and project creators set a minimum goal to publish the project.

In this project it is aimed to determine which features make the Kickstarter Projects successful or not. For this purpose, data preprocessing and cleaning operations will be made to have a smoother dataset and different machine learning algorithms will be trained in order to find the best performing model.

* 1. **Summarize & Analyze Data**

The dataset contains 378661 projects between 1 January 1970 to 2 January 2018. There are 14 independent variables containing seven categorical and seven numeric variables.

The target feature is the “state” column which determines the project condition. There are six different conditions for any project. These are successful, failed, canceled, live and undefined. As per the dataset, around 2799 projects were still live (active) at the end of 2018.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Failed | Successful | Canceled | Undefined | Live | Suspended |
| 197719 | 133956 | 38779 | 3562 | 2799 | 1846 |

Table 1: Target Feature Category Counts

The Kickstarter Project has a numeric identifier called “ID” which is unique for each project and categorical value called "name".

Projects have 15 different “main\_category” which are limited by Kickstarter, these are art, comics, crafts, dance etc. and have 159 different “category” under “main\_category”.

The “currency” variable depends on the “country” variable, currency is a system of money in general use in a particular country. On the other hand, “country” is where the project is started.

“Deadline” and “launched” define the project start date and intended due date. If people like the idea of the project, they can pledge money to carry out the project. So, “pledged” variable defines that.

“Goal” is the funding goal of the project which funding came from pledged money. Funding on Kickstarter is all-or-nothing. “backers” are the number of people who supported and funding pledges to the project.

The pledged amount "usd\_pledged" has an equivalent value converted to USD, called "usd\_pledged\_real" and the “usd\_goal\_real” is converted goal value.

There are 3797 missing values in the dataset, all of them exist in the usd\_pledged feature.

There are 23 different countries and 14 different currencies in this dataset. The most common country is the US and the most common currency is USD. And the most common category is Product Design and the most common main\_catagory is Film & Video.

There are no duplicated data in the dataset.

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Count | Common | Common Probability |
| Category | 159 | Product Design | 5.89 |
| Country | 23 | US | 77.3 |
| Currency | 14 | USD | 78.0 |
| Deadline | 3164 | 2014-08-08 | 0.186 |
| Launched | 378089 | 1970-01-01 | 0.00185 |
| Main\_Category | 15 | Film & Video | 16.8 |
| Name | 375765 | New EP/Music Dev. | 0.0108 |
| State | 6 | Failed | 52.2 |

Table 2: Feature Distributions and Their Common Attributes

* 1. **Visualizing Data**

We will start looking at the state column distribution that might be our key to understand this dataset. Failed and successful states consist of 87.6% of the dataset. It seems more projects failed than succeeded.

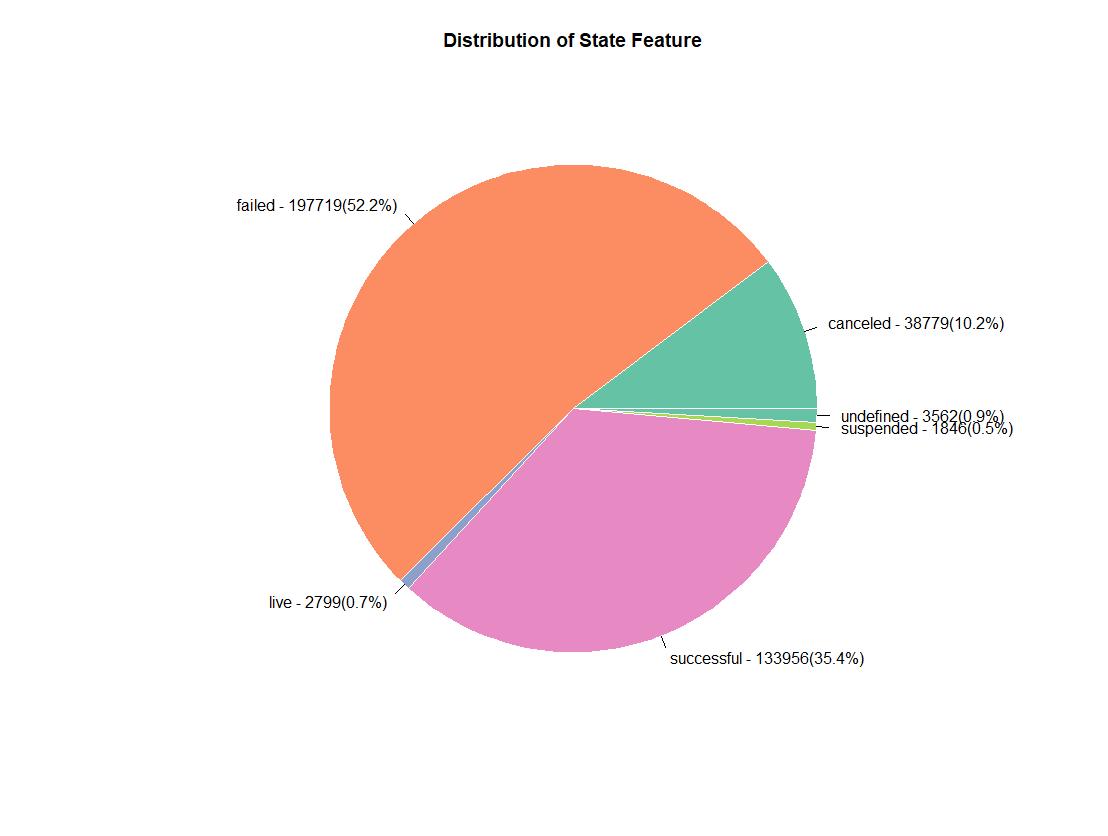


Figure 1: Target Feature Distribution Chart

Now, let’s look at the main\_category distribution. As we can see Film & Video is the most popular main\_category and Dance is the least one.

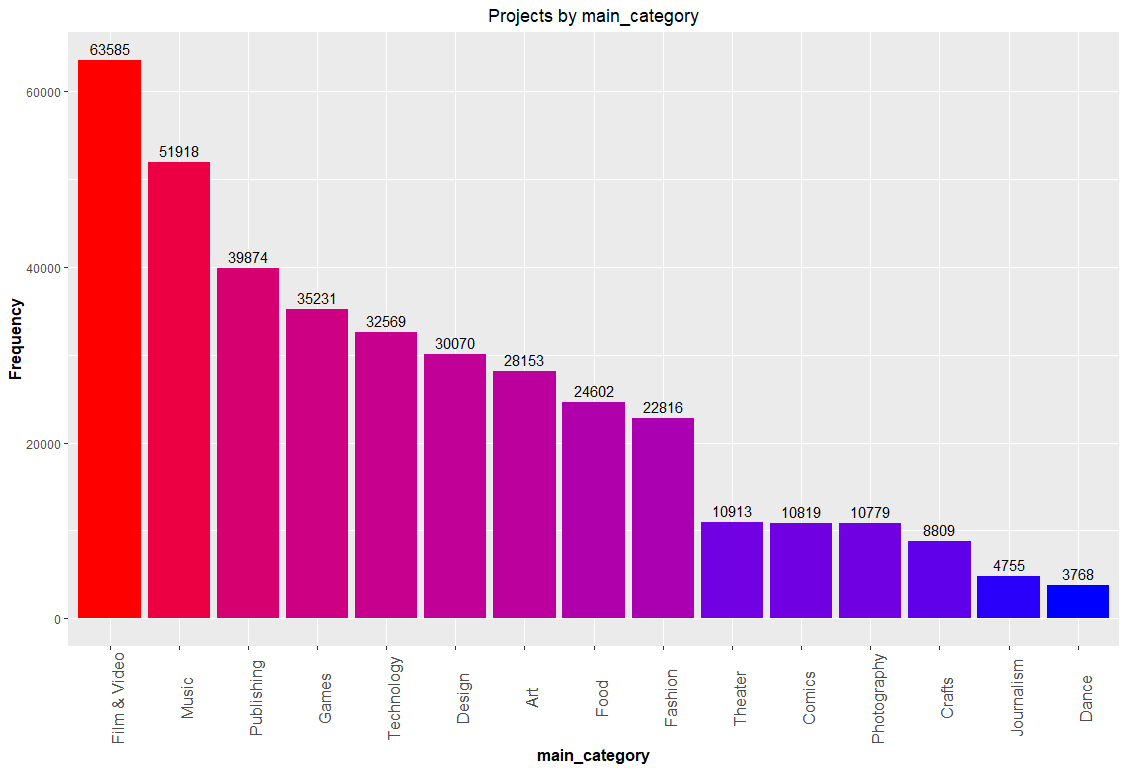


Figure 2: Main Category Distribution

Projects are most commonly developed in the US by far. So, the US leads with a number of kickstarters followed by Great Britain, Canada, and Australia.

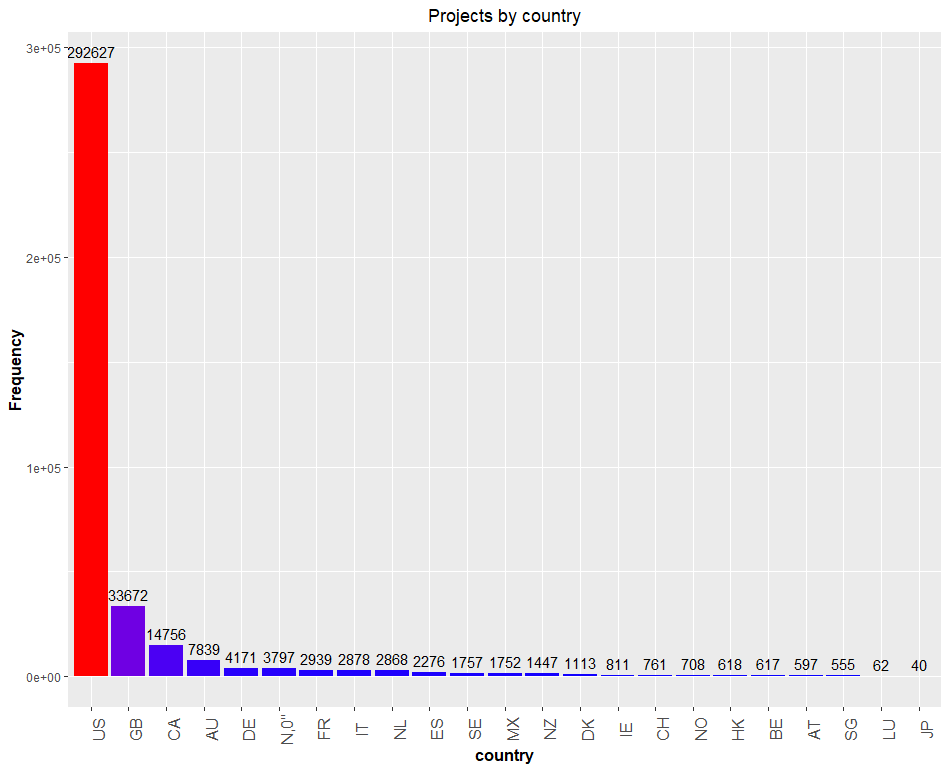


Figure 3: Projects by Country Distribution

Projects were launched between 1970 and 2018 and as we can see there is a huge gap between 1970 to 2009. Kickstarter was launched in 2009 and there are only 7 projects in the dataset before this date.

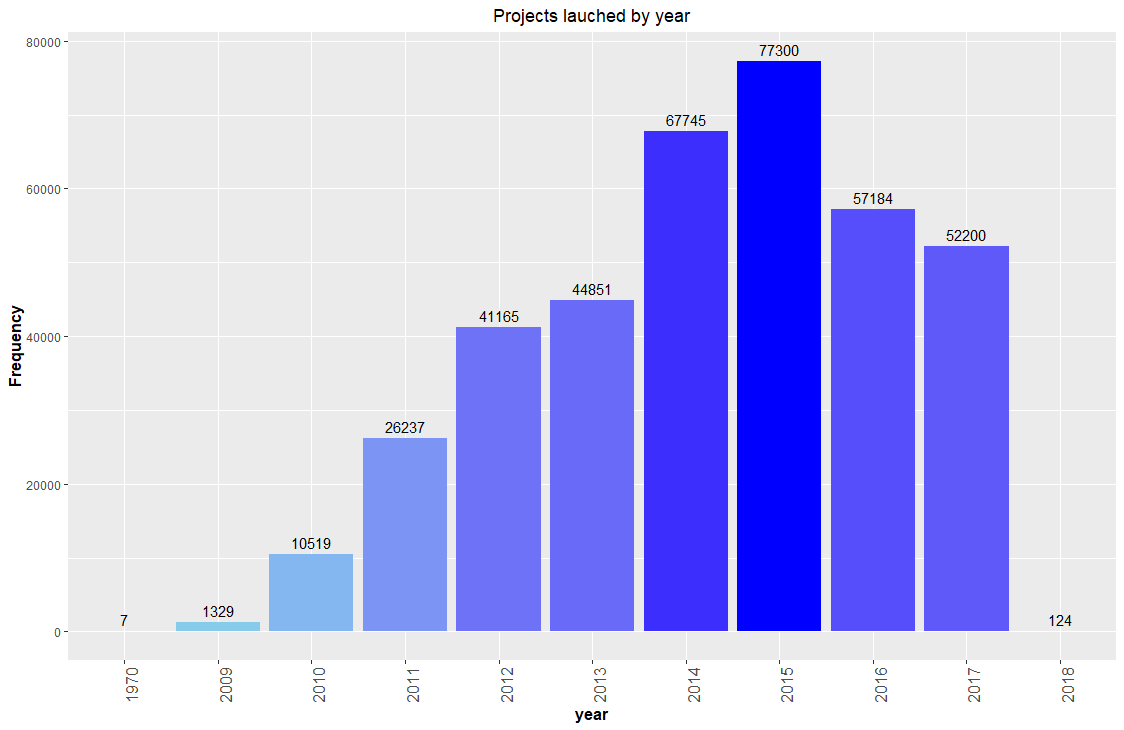


Figure 4: Projects Launched by Year Distribution

* + 1. ***Plot Distributions of the Features***

As we can see from below graphs, there are too many projects with zero goal, backers and pledged amounts. ID and name almost have unique value in each row. Currency and country have one variable that have high frequency which is USD and US.

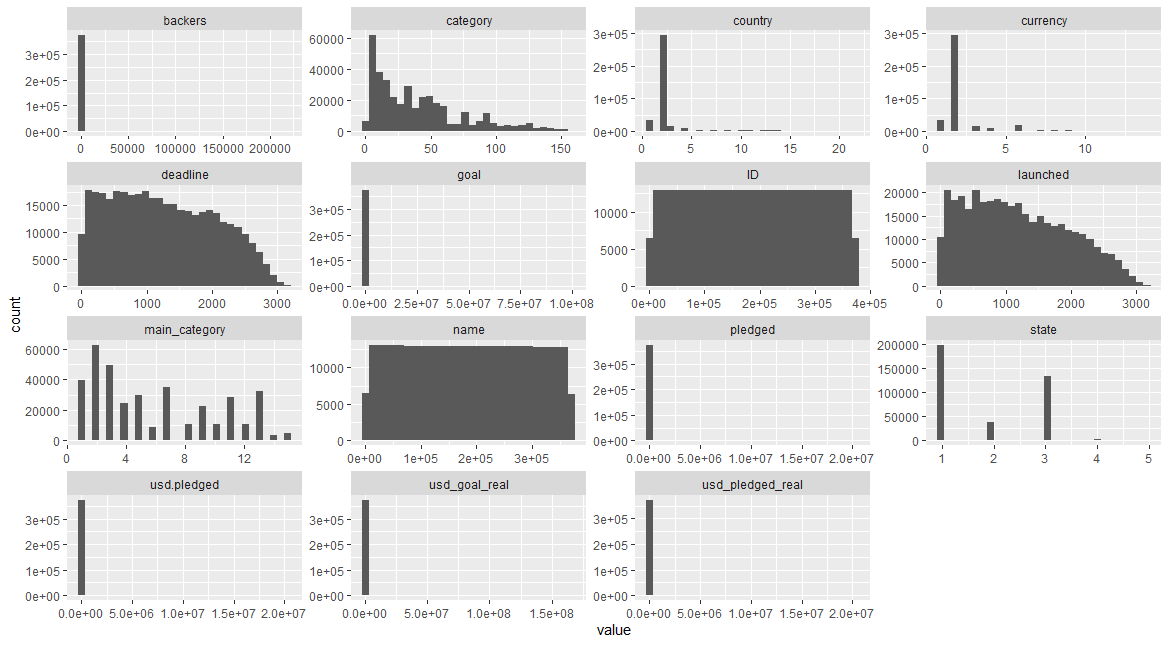


Figure 5: Plot Distributions of the Features

* + 1. ***Correlation Matrix***

Let’s see the correlation between variables in our dataset. Correlation matrix analysis is very useful to study dependences or associations between variables. There are different methods for correlation analysis : Pearson correlation test, Spearman and Kendall rank-based correlation analysis. For this purpose cor() function used to calculate p-values and correlation values. The Pearson correlation is computed by default with the cor() function, so we used a Pearson test in this project. First of all, correlation ranges from -1 to 1. Correlation close to zero indicates that two variables are independent and correlation close to -1 or +1 indicates there is negative or positive correlation between two variables.

Correlation between ID and name is 0.99 which means there is a positive correlation and p-values are 0 which means lower than significance value(0.005). There is a positive correlation between currency and country (0.94) which makes total sense. There is a positive correlation between goal and usd\_goal\_real(0.94). There is a positive correlation between pledged and backers(0.72), usd\_pledged(0.86), usd\_pledged\_real(0.95). If the backers person number increased to the project, the pledged amount will be increased as well. There is again positive correlation between backers and usd\_pledged(0.7) and usd\_pledged\_real(0.75). And finally there is a positive correlation between usd\_pledged and usd\_pledged\_real(0.91). So, we have to consider this results for further processing and deal with it properly.

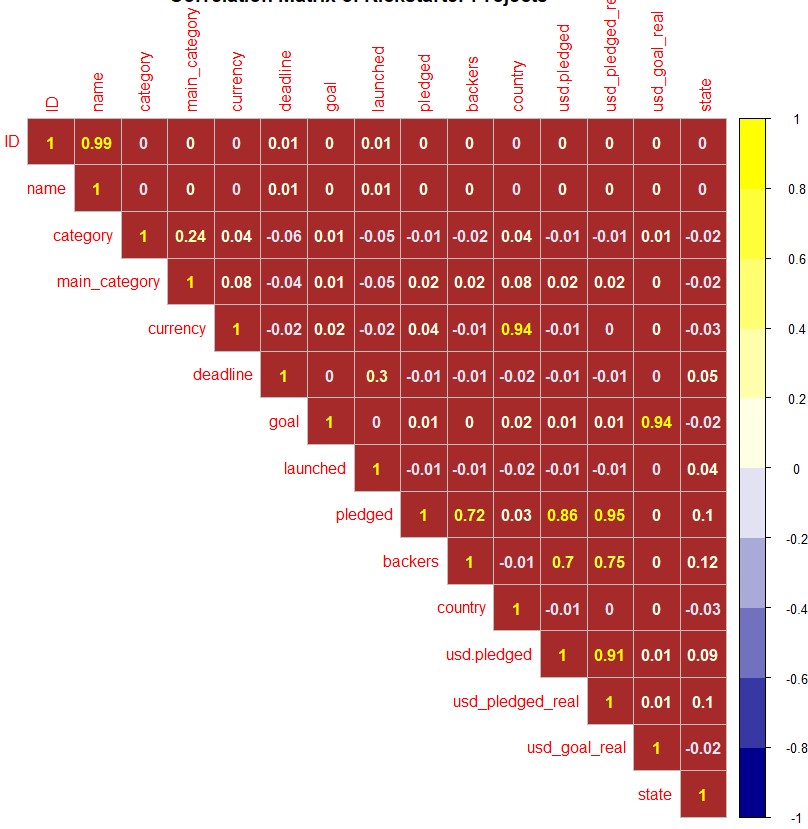


Figure 6: Correlation Matrix

**CHAPTER 2**

**DATA PREPARATION**

**2.1. Dimensionality Reduction**

Kickstarter dataset has 15 features with multiple data types. The ‘ID’ and the ‘name’ features are unique to each element of the dataset and do not contribute when predicting the possibility of a project’s success. Due to this, both columns are removed from the dataset.

Category and Main Category features are related to each other. Category feature has 159 unique values, and it is the sub feature of the Main Category column. Instead of using both features, Category column is removed.

As it was stated in Chapter 1, Currency column is depended to the Country column and both have biased distribution with Currency = ‘USD’ as %78 and Country = ‘US’ as %77.3. Due to the lack of information about other countries’ Kickstarter projects, both features are removed. Dataset’s feature dimension is reduced to twelve features.

**2.2. Handling Missing Values**

Dataset has total of 3797 missing values, all from the usd\_pledged feature. The corresponding target feature of these missing values are all labeled as ‘undefined’. The ‘undefined’ label can not be predicted using a machine learning model because it can be a reason of different factors. So, the way of handling the missing values is omitting them from the dataset.

**2.3. Removing Unimportant Target Labels**

The target feature of the dataset has 6 unique values, however except the ‘failed’ and the ‘successful’ label, all other labels are useless for prediction. ‘Suspended’ and ‘canceled’ labels are correlated to the business problems. This project aims to predict whether a published Kickstarter project will fail or not. Every row with ‘live’, ‘canceled’, ‘suspended’ and ‘undefined’ labels are removed from the dataset.

**2.4. Data Preprocessing**

In order to have a better machine learning model, categorical features of the dataset must be encoded. Except the ‘Main Category’ and the target ‘State’ features, all other features are either float data type or numeric data type. Float data type is converted into numerical. Categorical features are encoded using the R’s factor() function each having labels from 1 to N. Target feature is now translated into failed = 0 and successful = 1.

Chart, bar chart

Description automatically generated

Figure 7: Categories and Rates of Success

***2.4.1. Feature Selection***

Feature selection can improve the efficiency of the machine learning model and help with the efficiency. Raw dataset has 12 features. Using two different feature selection techniques, features can be reduced. First technique that will be applied on the data is using cross-validation with random forest classifier to find the minimum number of features with highest accuracy result. N-fold cross validation with n = 10 is used and the accuracies in Figure 8 were computed.

The top 4 variables: usd\_goal\_real, goal, usd\_pledged\_real, pledged.

Figure 9 shows the most important features for the classification model. Here, top four features are selected as ‘usd\_goal\_real’, ‘goal’, ‘usd\_pledged\_real’ and ‘pledged’. It can be deducted from both methods that these four features can correctly train the model and give high accuracy results. Hence, the dataset features are reduced to the selected features.

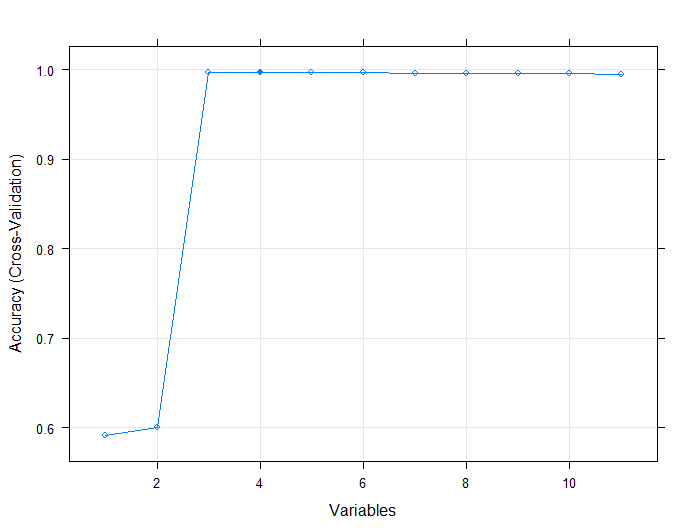


Figure 8: Cross Validation-Accuracy Plot

The second method for feature selection is the feature importance score method. It is calculated by using n-fold cross validation over the dataset and estimating the variable importance. The n value is selected as 10 and the classification method is selected as multinominal logistic regression model.

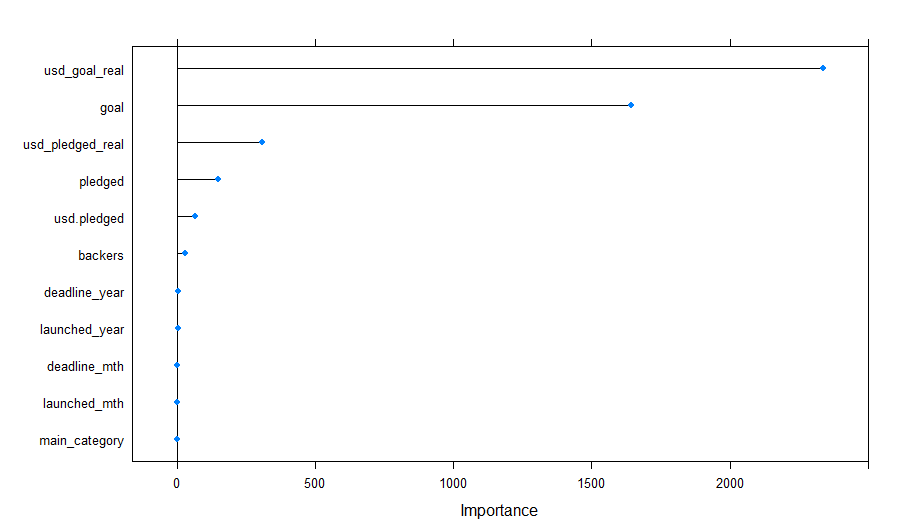


Figure 9: Feature Importance Plot

***2.4.4. Removing Outliers***

Outliers can be misleading when building a machine learning model. After the feature selection, dataset now has 4 independent features and one target feature. Each feature is checked for outliers using the distribution-based outlier detection. Using the five-number summary (minimum, first quartile, median, third quartile, maximum) of a feature’s distribution, outliers can be detected and removed.

Interquantile range(IQR) = Q3 - Q1

Upper Range = Q3 + 1.5 \* IQR

Lower Range = Q1 – 1.5 \* IQR

Rows below the lower range and above the upper range are considered as an outlier and are removed. Figure 10 shows each feature’s range before and after the outliers are removed.

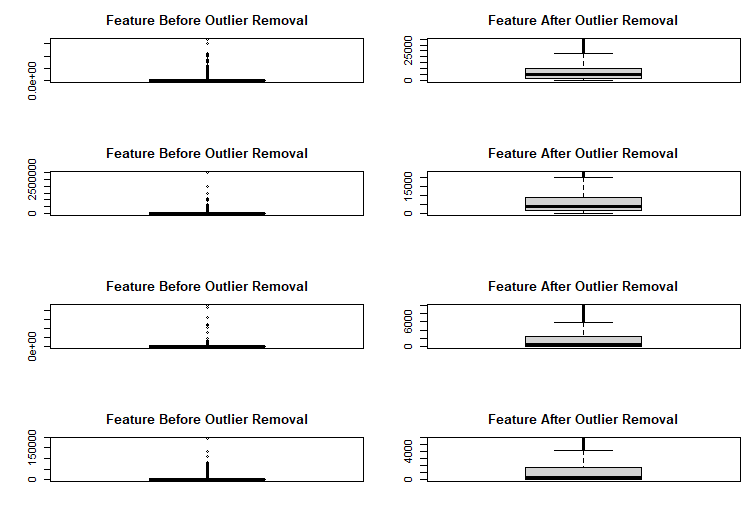


Figure 10: Removing Outliers - Before and After Boxplots

***2.4.3. Undersampling & Random Sampling***

The dataset, after dimensionality reduction and removing outliers, has 170.849 rows left. This value is quite large and could result in an overfitted machine learning model. Also, the ‘failed’ label ratio over the whole dataset is %73 and the ‘successful’ label is %27. This creates a biased dataset which will affect the prediction results badly. To overcome this problem, undersampling method is applied to the dataset. Random selection of ‘failed’ labels with same count as the ‘successful’ rows gives a balanced dataset with 145.810 elements. However, this number is still quite large, so random sampling is applied to the dataset. The final dataset now has 30.000 rows.

**CHAPTER 3**

**MODEL TRAINING AND EVALUATION**

**3.1. SUPPORT VECTOR MACHINE**

**3.1.1. 10-Fold Cross Validation using Raw Dataset**

The original dataset without using the outlier removing techniques and feature selection methods is used in this section for comparing the results with the preprocessed dataset. 10-fold cross validation technique with Suppor Vector Machine classifier (kernel = ‘linear’) is used for training. Following results were calculated:

Mean Accuracy: %86,35

Mean Recall: %72,79

Mean Precision: %91,66

Mean F-Score: %81,13

|  |  |  |
| --- | --- | --- |
|  | Predicted ‘Failed’ | Predicted ‘Successful’ |
| Actual ‘Failed’ | 1731 | 63 |
| Actual ‘Successful’ | 335 | 871 |

Table 3: Confusion Matrix of SVM – Raw Dataset

Table 3 shows an example confusion matrix from one of the fold’s.

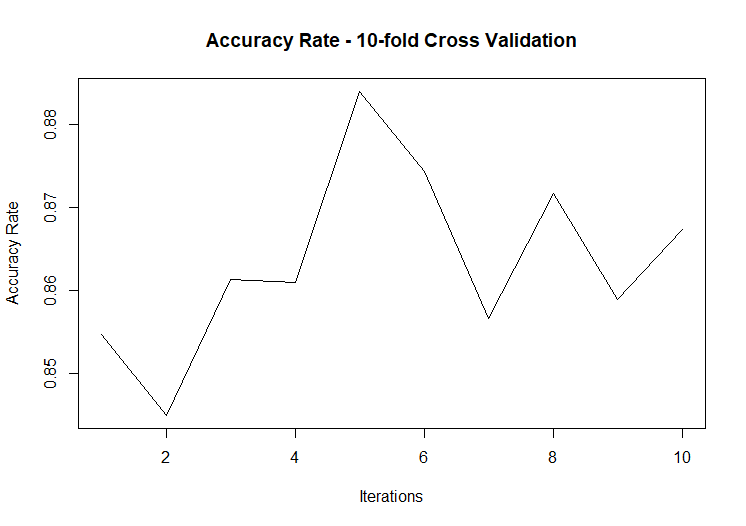


Figure 11: Accuracy Rate of Raw Dataset (SVM)

**3.1.2. 10-Fold Cross Validation using Preprocessed Dataset**

The preprocessed dataset is acquired using the methods mentioned in Chapter 2. The results compared to the raw dataset with the same machine learning model was more efficient at predicting the results.

Mean Accuracy = %97.03

Mean Recall = %98.77

Mean Precision = %95.45

Mean F-Score = %97.08

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Folds | Accuracy | Recall | Precision | F-Score |
| 1 | 99.23 | 100 | 98.29 | 99.13 |
| 2 | 99.40 | 100 | 98.44 | 99.21 |
| 3 | 99.26 | 100 | 98.45 | 99.21 |
| 4 | 99.53 | 100 | 98.38 | 99.18 |
| 5 | 98.93 | 100 | 97.83 | 98.90 |
| 6 | 99.30 | 100 | 98.51 | 99.25 |
| 7 | 99.33 | 100 | 98.40 | 99.19 |
| 8 | 99.16 | 100 | 98.69 | 99.34 |
| 9 | 99.16 | 100 | 98.59 | 99.29 |
| 10 | 99.16 | 100 | 98.50 | 99.24 |

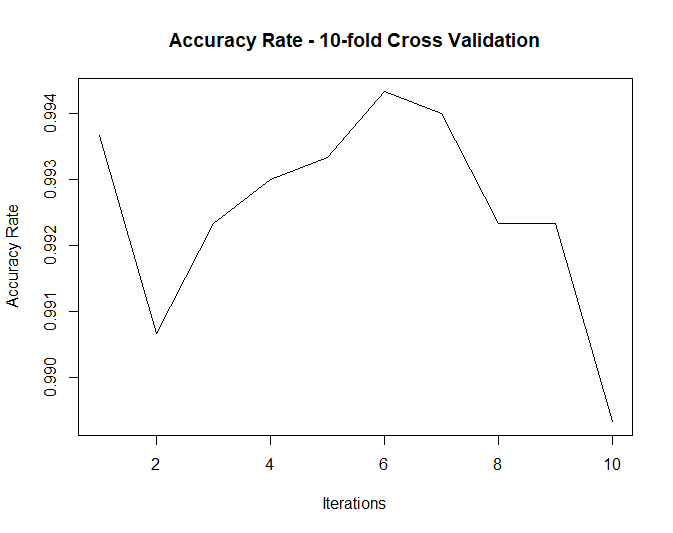
Table 4: Evaluation Metrics - SVM

Figure 12: Accuracy Rate of Preprocessed Data (SVM)

**3.2. DECISION TREE CLASSIFIER**

***3.2.1. 10-Fold Cross Validation***

Cross validation is a resampling approach which makes it possible to obtain a more honest error rate estimation of the model computed on the whole dataset. It is almost available on all the data mining software. Advantages of cross validation is fast computation speed and a very effective method to estimate the prediction error and the accuracy of a model. Basically, the cross validation consists to randomly split the data in K folds. We reiterate the following process, by turning the sub-samples: learning the model on (K-1) folds, computing the error rate on the fold number K. The error rate in cross-validation is the mean of these error rates collected. It is a better estimator of the classifier performance than the resubstitution error rate. From this description, we transcribe the operations in R. We randomly create a column indicating the individuals belonging to the folds.

N <- nrow(dataset\_feature)

decision\_tree\_errors <- numeric()

K <- 10

size <- n%/%K

vol <- runif(n)

rank <- rank(vol)

blok <- (rank -1)%/%size + 1

blok <- as.factor(blok)

We observe that we have the same number of examples in each fold, 3000 rows.

Figure 13 shows the boxplot of the accuracies acquired from cross validation. It can be observed from this plot that the mean of the accuracies are quite high, near to 0.98.

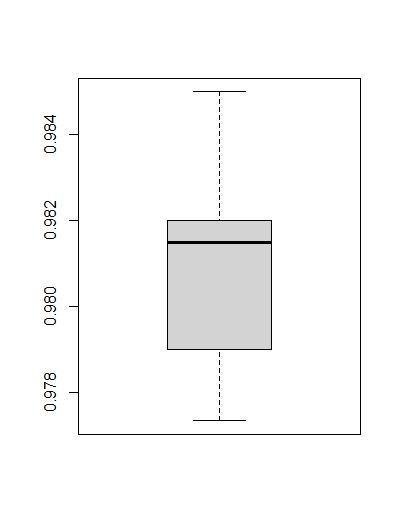


Figure 13: Decision Tree Cross Validation Accuracy Boxplot

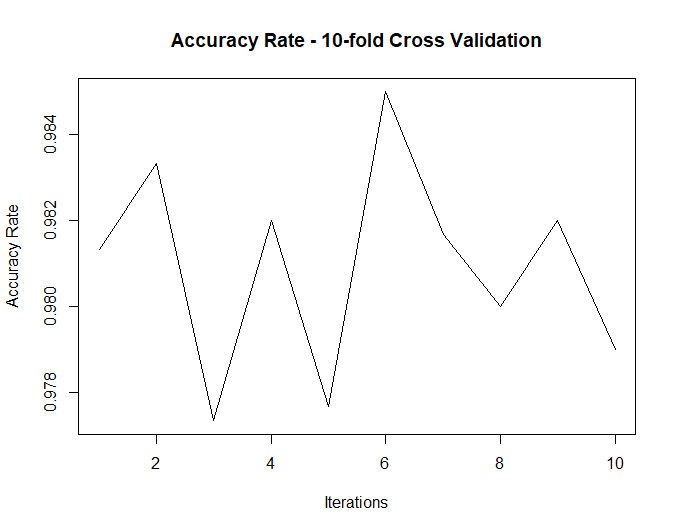


Figure 14: Decision Tree Cross Validation Accuracy Plot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Folds | Accuracy | Recall | Precision | F-Score |
| 1 | 97.06 | 98.57 | 95.59 | 97.06 |
| 2 | 96.80 | 98.63 | 95.27 | 96.92 |
| 3 | 97.76 | 98.46 | 97.09 | 97.77 |
| 4 | 96.53 | 98.70 | 94.79 | 96.71 |
| 5 | 97.33 | 99.12 | 95.66 | 97.36 |
| 6 | 97.20 | 98.82 | 95.50 | 97.13 |
| 7 | 96.93 | 98.78 | 95.16 | 96.94 |
| 8 | 97.26 | 99.08 | 94.57 | 97.35 |
| 9 | 96.66 | 98.91 | 95.13 | 96.69 |
| 10 | 96.80 | 98.59 | 95.69 | 96.85 |

Mean Accuracy = %97.03

Mean Recall = %98.77

Mean Precision = %95.45

Mean F-Score = %97.08

***3.1.2. %70 Training Set - %30 Test Set***

The dataset is split into %70 training set and %30 test set. Decision tree classifier using the R’s rpart class. The accuracy is %96.84. Comparing the result with the cross validation results, it can be observed that the accuracy score’s are almost equal and the model gives good predictions of the test results.

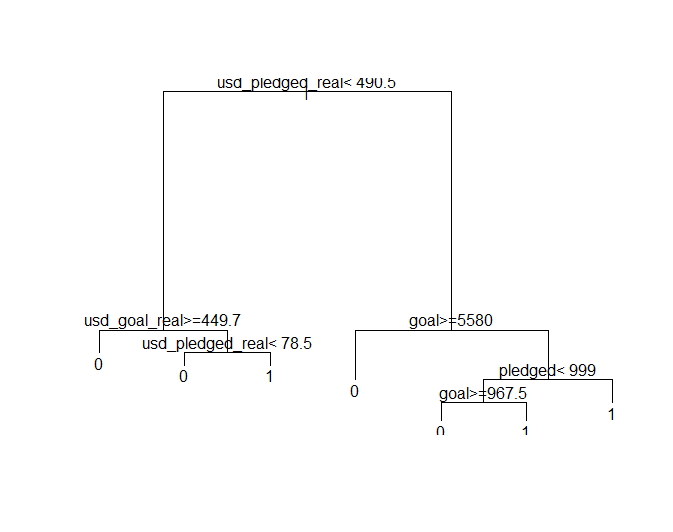


Figure 15: Decision Tree

Figure 15 shows the Decision Tree of the dataset.

|  |  |  |
| --- | --- | --- |
|  | Predicted ‘Failed’ | Predicted ‘Successful’ |
| Actual ‘Failed’ | 4289 | 221 |
| Actual ‘Successful’ | 63 | 4427 |

Table 5: Confusion Matrix of Decision Tree Model

**3.3. K-NEAREST NEIGHBOUR**

This chapter uses the R’s caret library for cross validation. Library has the trainControl and train functions to evaluate the cross validation scheme on the dataset. The K value has the range from 1 to 10. For each K value, an accuracy score is evaluated.

|  |  |  |
| --- | --- | --- |
| K | Accuracy | Kappa |
| 1 | 0.9992000 | 0.9984000 |
| 2 | 0.9989333 | 0.9978666 |
| 3 | 0.9989667 | 0.9979333 |
| 4 | 0.9988333 | 0.9976666 |
| 5 | 0.9986667 | 0.9973333 |
| 6 | 0.9986667 | 0.9973333 |
| 7 | 0.9985000 | 0.9970000 |
| 8 | 0.9984000 | 0.9968000 |
| 9 | 0.9982000 | 0.9964000 |
| 10 | 0.9982333 | 0.9964666 |

The model selected the optimal K value as 1 however each accuracy is quite high and similar to each other.

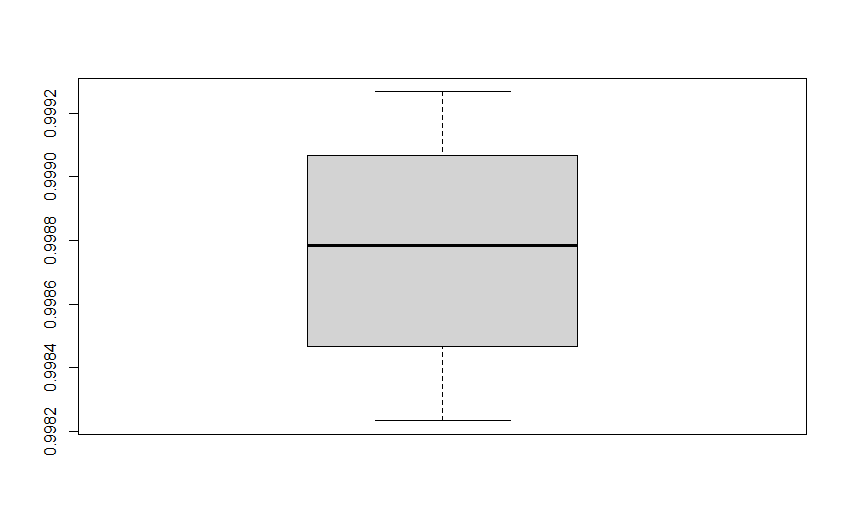


Figure 16: KNN Accuracy Box Plot

**CHAPTER 4**

**CONCLUSION**

The K-Nearest Neighbour model has the best accuracy results compared to the other machine learning models used in this project for the Kickstarter dataset. The main reason the accuracies were close to 1.0 is, using data preprocessing techniques have made our dataset easy to predict. The features used can directly affect the project’s success rate due to the fact that, if a project does not meet its ‘goal’ feature, it usually fails.

The KNN model has found the best accuracy result when the K is 1. Figure 17 shows the model accuracy comparisons. Chart, box and whisker chart

Description automatically generated

Figure 17: Model Comparison Plot

**APPENDIX A. CODE**

dataset <- read.csv("C:/Users/HP/Desktop/ks-projects-201801.csv")0

#extract launched and deadline year and month from launched and deadline date features

dataset$launched <- as.Date(dataset$launched, "%Y-%m-%d")

dataset$launched\_year <- substr(dataset$launched, 1,4)

dataset$launched\_mth <- substr(dataset$launched, 6,7)

dataset$deadline\_year <- substr(dataset$deadline, 1,4)

dataset$deadline\_mth <- substr(dataset$deadline, 6,7)

dataset$launched\_year <- as.integer(dataset$launched\_year)

dataset$launched\_mth <- as.integer(dataset$launched\_mth)

dataset$deadline\_year <- as.integer(dataset$deadline\_year)

dataset$deadline\_mth <- as.integer(dataset$deadline\_mth)0

#missing values in data set

colSums(is.na(dataset))

#Sum of missing values

sum(is.na(dataset))

#first 5 column in data set

head(dataset)

#show each feature frequencies in plot

library(inspectdf)

library(dplyr)

data\_cat <- dataset %>% inspect\_cat()

data\_cat %>% show\_plot(high\_cardinality = 1, col\_palette = 2)

#most populer categories

library(ggplot2)

library(devtools)

cat.freq <- dataset %>%

group\_by(main\_category) %>%

summarize(count=n()) %>%

arrange(desc(count)

cat.freq$main\_category <- factor(cat.freq$main\_category, levels=cat.freq$main\_category)

ggplot(cat.freq, aes(main\_category, count, fill=count)) + geom\_bar(stat="identity") +

ggtitle("Projects by main\_category") + xlab("main\_category") + ylab("Frequency") +

geom\_text(aes(label=count), vjust=-0.5) +

theme(plot.title=element\_text(hjust=0.5), axis.title=element\_text(size=12, face="bold"),

axis.text.x=element\_text(size=12, angle=90), legend.position="null") +

scale\_fill\_gradient(low="blue", high="red")

#most popular countries

countries <- dataset %>%

group\_by(country) %>%

summarize(count=n()) %>%

arrange(desc(count))

countries$country <- factor(countries$country, levels=countries$country)

ggplot(countries, aes(country, count, fill=count)) + geom\_bar(stat="identity") +

ggtitle("Projects by country") + xlab("country") + ylab("Frequency") +

geom\_text(aes(label=count), vjust=-0.5) +

theme(plot.title=element\_text(hjust=0.5), axis.title=element\_text(size=12, face="bold"),

axis.text.x=element\_text(size=12, angle=90), legend.position="null") +

scale\_fill\_gradient(low="blue", high="red")

#number of project lauch by year

lauch\_year <- dataset %>%

group\_by(launched\_year) %>%

summarize(count=n())

lauch\_year$launched\_year <- factor(lauch\_year$launched\_year, levels=lauch\_year$launched\_year)

ggplot(lauch\_year, aes(launched\_year, count, fill=count)) + geom\_bar(stat="identity") +

ggtitle("Projects lauched by year") + xlab("year") + ylab("Frequency") +

geom\_text(aes(label=count), vjust=-0.5) +

theme(plot.title=element\_text(hjust=0.5), axis.title=element\_text(size=12, face="bold"),

axis.text.x=element\_text(size=12, angle=90), legend.position="null") +

scale\_fill\_gradient(low="skyblue", high="blue")

#Target feature (state) info

stateMap <- sort(table(dataset$state), decreasing = TRUE)

print(stateMap)

print(length(stateMap))

#Pie chart of state

pie\_percent <- round(100 \* stateMap / sum(stateMap), 1)

pie(stateMap, labels = pie\_percent, main = "Distribution of State Feature",

col = rainbow(length(stateMap)), radius = 1)

legend("topright", rownames(stateMap), fill = rainbow(length(stateMap)))

#corralation matrix

encode\_ordinal <- function(x, order = unique(x)) {

x <- as.numeric(factor(x, levels = order, exclude = NULL))

}

corr\_matrix\_data <- dataset

corr\_matrix\_data$ID <- encode\_ordinal(dataset$ID)

corr\_matrix\_data$name <- encode\_ordinal(dataset$name)

corr\_matrix\_data$category <- encode\_ordinal(dataset$category)

corr\_matrix\_data$main\_category <- encode\_ordinal(dataset$main\_category)

corr\_matrix\_data$currency <- encode\_ordinal(dataset$currency)

corr\_matrix\_data$deadline <- encode\_ordinal(dataset$deadline)

corr\_matrix\_data <- select(corr\_matrix\_data, -8)

corr\_matrix\_data$country <- encode\_ordinal(dataset$country)

corr\_matrix\_data$state <- encode\_ordinal(dataset$state)

#p-values and corralation values based on pearson

library(Hmisc)

rcorr(as.matrix(corr\_matrix\_data))

library(RColorBrewer)

library(corrplot)

M<-cor(corr\_matrix\_data)

corrplot(M, type = "upper", method = "number", title = "Correlation Matrix of Kickstarter Projects",

col = colorRampPalette(c("darkblue", "white", "yellow"))(10), bg = "brown")

#Plot Distributions of the Features

library(purrr)

library(tidyr)

library(ggplot2)

corr\_matrix\_data %>%

keep(is.numeric) %>%

gather() %>%

ggplot(aes(value)) +

facet\_wrap(~ key, scales = "free") +

geom\_histogram()

#Rearranging and removing unnecessary columns

dataset <- dataset[c(4,7,9,11,13,14,15,16,17,18,19,10)]

#data set information

dim(dataset)

summary(dataset)

#Missing values are omitted from dataset

dataset <- na.omit(dataset)

#Removing unnecessary state labels

dataset <- subset(dataset, state != 'live')

dataset <- subset(dataset, state != 'canceled')

dataset <- subset(dataset, state != 'suspended')

dataset <- subset(dataset, state != 'undefined')

#Main\_category-State Relationship

barplot(table(dataset$state, dataset$main\_category),

main = "Categories and Their Rates of Succcess",

xlab = "Categories",

ylab = "Count",

col = c("red","green")

)

legend("topleft",

c("Failed","Successful"),

fill = c("red","green")

)

# Encoding categorical data - label encoding

dataset$main\_category = factor(dataset$main\_category,

levels = c('Film & Video', 'Music', 'Publishing', 'Games', 'Technology', 'Design', 'Art', 'Food', 'Fashion', 'Theater','Comics','Photography', 'Crafts',

'Journalism', 'Dance'),

labels = c(1:15))

dataset$state = factor(dataset$state, levels = c('failed','successful'), labels = c(0,1))

#Float-Integer Conversion

dataset$main\_category <- as.numeric(dataset$main\_category)

dataset$pledged <- as.integer(dataset$pledged)

dataset$main\_category <- as.integer(dataset$main\_category)

dataset$usd.pledged <- as.integer(dataset$usd.pledged)

dataset$usd\_pledged\_real <- as.integer(dataset$usd\_pledged\_real)

#################################################################################

newdt = dataset[sample(nrow(dataset), 30000),]

newdt[-12] = scale(newdt[-12])

#################################################################################

#FEATURE SELECTION USING CROSS VALIDATION ACCURACY RESULTS

# ensure the results are repeatable

# load the library

library(mlbench)

library(caret)

detach(package:purrr)

detach(package:dplyr)

detach(package:Hmisc)

set.seed(7)

# define the control using a random forest selection function

control <- rfeControl(functions=rfFuncs, method="cv", number=10)

# run the RFE algorithm

results <- rfe(newdt[,1:11], newdt[,12], sizes=c(1:11), rfeControl=control)

# summarize the results

print(results)

# list the chosen features

predictors(results)

# plot the results

plot(results, type=c("g", "o"))

#FEATURE SELECTION USING FEATURE IMPORTANCE SCORE

set.seed(7)

# load the library

library(caret)

# prepare training scheme

control <- trainControl(method="cv", number=10)

# train the model

model <- train(state~., data=newdt, method="multinom",trControl=control)

# estimate variable importance

importance <- varImp(model, scale=FALSE)

# summarize importance

print(importance)

# plot importance

plot(importance)

#USING SELECTED FEATURES

dataset\_feature <- dataset[, c('usd\_goal\_real', 'goal','usd\_pledged\_real', 'pledged', 'state')]

###############################################################################

#OUTLIER DETECTION AND REMOVAL

library(dplyr)

par(mfrow = c(4,2), mar=c(4,4,3,1))

for(i in 1:4){

boxplot(dataset\_feature[,i], main = "Feature Before Outlier Removal")

Q<-quantile(dataset\_feature[,i],probs=c(.25, .75), na.rm = FALSE)

iqr<-IQR(dataset\_feature[, i])

up <- Q[2]+1.5\*iqr # Upper Range

low<- Q[1]-1.5\*iqr # Lower Range

dataset\_feature<- subset(dataset\_feature, dataset\_feature[,i] > (Q[1] - 1.5\*iqr) & dataset\_feature[,i]< (Q[2]+1.5\*iqr))

boxplot(dataset\_feature[,i], main = "Feature After Outlier Removal")

}

failed\_state\_data <- dataset\_feature[dataset\_feature$state == "0", ]

failed\_state\_data <- failed\_state\_data[sample(nrow(failed\_state\_data), nrow(dataset\_feature[dataset\_feature$state == "1", ])), ]

successful\_state\_data <- dataset\_feature[dataset\_feature$state != "0", ]

dataset\_feature <- rbind(successful\_state\_data, failed\_state\_data)

dataset\_feature = dataset\_feature[sample(nrow(dataset\_feature), 30000),]

#################################################################################

library(caTools)

set.seed(123)

split = sample.split(dataset\_feature$state, SplitRatio = 0.70)

training\_set = subset(dataset\_feature, split == TRUE)

test\_set = subset(dataset\_feature, split == FALSE)

dataset\_feature$state = factor(dataset\_feature$state, levels = c(0,1), labels = c('failed','successful'))

##################################################################################

#decision tree with 10-fold cross validation

library(rpart)

library (ROCR)

n <- nrow(dataset\_feature)

decision\_tree\_accuracy <- numeric()

recall <- numeric()

precision <- numeric()

f\_measure <- numeric()

K <- 10

size <- n%/%K

vol <- runif(n)

rank <- rank(vol)

blok <- (rank -1)%/%size + 1

blok <- as.factor(blok)

for(k in 1:K) {

classifier\_4 <- rpart(state ~.,

data = dataset\_feature[blok!=k,],

method = "class")

pred <- predict(classifier\_4, newdata = dataset\_feature[blok==k,], type = "class")

cm\_2 <- table(dataset\_feature$state[blok==k], pred)

precision <- c(precision, cm\_2[2,2] / (cm\_2[2,2] + cm\_2[1,2]))

recall <- c(recall, cm\_2[2,2] / (cm\_2[2,2] + cm\_2[2,1]))

f\_measure <- c(f\_measure, (2\*precision[k]\*recall[k])/(precision[k] + recall[k]))

accuracy <- sum(diag(cm\_2)) / sum(cm\_2)

decision\_tree\_accuracy <- c(decision\_tree\_accuracy, sum(diag(cm\_2)) / sum(cm\_2))

print(accuracy)

}

print(decision\_tree\_accuracy)

mean(decision\_tree\_accuracy)

mean(f\_measure)

mean(recall)

mean(precision)

plot(decision\_tree\_accuracy, type="l", ylab="Accuracy Rate", xlab="Iterations", main="Accuracy Rate - 10-fold Cross Validation")

dt\_tr <- rpart(state ~.,

data = training\_set)

# Predicting the Test set results

y\_pred = predict(dt\_tr, newdata = test\_set[-5], type = 'class')

plot(dt\_tr)

# Making the Confusion Matrix

cm = table(test\_set[, 5], y\_pred)

accuracy\_dt = (sum(diag(cm) / sum(cm)))

# plot final DT

plot(dt\_tr, type = "uniform")

text(dt\_tr)

#########################

#SVM

recall <- numeric()

precision <- numeric()

f\_measure <- numeric()

svm\_accuracy <- numeric()

for(k in 1:K) {

library(e1071)

classifier = svm(formula = state ~ .,

data = dataset\_feature[blok!=k,],

kernel = 'linear')

pred <- predict(classifier, newdata = dataset\_feature[blok==k,], type = "class")

cm\_2 <- table(dataset\_feature$state[blok==k], pred)

precision <- c(precision, cm\_2[2,2] / (cm\_2[2,2] + cm\_2[1,2]))

recall <- c(recall, cm\_2[2,2] / (cm\_2[2,2] + cm\_2[2,1]))

f\_measure <- c(f\_measure, (2\*precision[k]\*recall[k])/(precision[k] + recall[k]))

accuracy <- sum(diag(cm\_2)) / sum(cm\_2)

svm\_accuracy <- c(svm\_accuracy, sum(diag(cm\_2)) / sum(cm\_2))

print(accuracy)

}

print(svm\_accuracy)

mean(svm\_accuracy)

mean(f\_measure)

mean(recall)

mean(precision)

plot(svm\_accuracy, type="l", ylab="Accuracy Rate", xlab="Iterations", main="Accuracy Rate - 10-fold Cross Validation")

#knn

# Encoding categorical data

dataset\_feature$state = factor(dataset\_feature$state,

levels = c(0,1),

labels = c('failed', 'successful'))

library(caret)

set.seed(123)

trCtrl.lr <- trainControl(method = "cv",

number = 10, #5-fold CV

classProbs = TRUE,

savePredictions = TRUE)

model.lr <- train(state ~ .,

data=dataset\_feature,

method="knn",

tuneGrid = expand.grid(k = 1:10),

metric = "Accuracy",

trControl = trCtrl.lr)

y\_pred <- predict(model.lr, test\_set[-5])

KNN\_accuracy <- model.lr$results['Accuracy']

##########################################

dataframe\_acc <- data.frame(KNN\_Accuracy = KNN\_accuracy, Decision\_Tree = decision\_tree\_accuracy, SVM = svm\_accuracy)

boxplot(dataframe\_acc)

**REFERENCES**

[1] <https://www.analyticsvidhya.com/blog/2018/05/improve-model-performance-cross-validation-in-python-r/>

[2] <http://www.sthda.com/english/articles/38-regression-model-validation/157-cross-validation-essentials-in-r/#:~:text=The%20k%2Dfold%20cross%2Dvalidation%20method%20evaluates%20the%20model%20performance,)%20(for%20example%205%20subsets)>

[3] <https://amunategui.github.io/binary-outcome-modeling/>

[4] <https://dataaspirant.com/decision-tree-classifier-implementation-in-r/>

[5] <https://www.r-bloggers.com/2020/02/a-guide-to-encoding-categorical-features-using-r/>

[6] <https://www.kaggle.com/kemical/kickstarter-projects?select=ks-projects-201801.csv>

[7] <http://www.sthda.com/english/articles/32-r-graphics-essentials/133-plot-one-variable-frequency-graph-density-distribution-and-more/>

[8] <https://drsimonj.svbtle.com/quick-plot-of-all-variables>