Kickstarter Project Capstone

Rob Shipley

2025-12-8

Introduction and Overview

The goal for this project is to ingest the Kickstarter Project dataset and evaluate to determine if it provides adequate data to establish a predictive model for determining if a project will succeed or fail. The initial analysis will evaluate the dataframe to assess the columns and categories. This initial insight will be used to perform additional normalization and regularization. The final intent it to apply two models to the dataframe using a train-test split to see if the results are worthy of assisting future projects to assess if a final Kickstarter fund raising activity is likely to be successful. When the dataset is converted into a dataframe, we will need to remove the Goal labeled column. If it is left in the dataframe, the model can essentially cheat to achieve a near 100% accuracy by determining if the goal is met by comparing the pledge column.

Initial research indicates that enough data is available to perform a 80/20 train-test split. It contains over 300K rows, so that justifies the split since enough data is available for the train set. The challenge is to determine if the data frame contains enough relevant observations to assess if a Kickstarter project is likely to succeed or fail. If we are close to 50%, then we know the feature columns lack enough context to proceed with using the data to indicate success or failure outcomes.

A key challenge will be the project Names column of the kickstarter projects only provides the title and lacks any significant context. Because of this, sentiment analysis will not be used. Instead, we need to determine the best predictive model with only the categories, timeframe, pledge amount and number of backers as variables.

Project Objectives

- Analyze the kickstarter dataframe to determine context
- Perform data wrangling to normalize and clean the dataframe
- Generate graphs to provide insight into the column features and relevance to the outcome
- Create a test-train split that can be used for modeling
- Apply ranger random forest and logistic regression models and determine success/fail prediction accuracy
- Demonstrate understanding of the outcome and provide a conclusion

Methodology and Analysis

Key Steps

The following key steps will be used during this project:

1. Load the libraries needed for analysis
2. Download the dataset, assigning datatypes to columns
3. Run a basic analysis of the kickstarter dataset to identify columns and summary of content
4. Apply graphs to the dataset for initial analysis
5. Clean the memory for optimal performance

6. Apply data factorization to the kickstarter dataframe
7. Split the kickstarter dataset into train and test sets
8. Store the train and test sets in memory
9. Apply ranger random forest
10. Apply normalization and regularization for logistic regression
11. Apply logistic regression models to the train set
12. Predict the outcome for test sets
13. Calculate the accuracy for test sets
14. Compare the two models
15. Provide a conclusion

R Code Analysis of Kickstarter Dataset

Load the Libraries

```
#----------------------------------------------------
# Load needed libraries
#----------------------------------------------------

if(!require(dplyr)) install.packages("dplyr", repos = "https://cloud.r-project.org")
```

```
## Loading required package: dplyr
```

```
##
## 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(dplyr)
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(caret)
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ lubridate 1.9.4     ✓ tibble    3.2.1
## ✓ purrr     1.0.4     ✓ tidyr     1.3.1
## ✓ readr     2.1.5
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ✖ purrr::lift()   masks caret::lift()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(tidyverse)
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
library(ggplot2)
if(!require(ranger)) install.packages("ranger", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: ranger
```

```
## Warning: package 'ranger' was built under R version 4.5.2
```

```
library(ranger)
```

Data Loading and Preparation

```
#----------------------------------------------------------
# Read Kickstarter dataset
#----------------------------------------------------------
kickstarter_ds <- read_csv("kickstarter_projects.csv")
```

```
## Rows: 374853 Columns: 11
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (5): Name, Category, Subcategory, Country, State
## dbl  (4): ID, Goal, Pledged, Backers
## dttm (1): Launched
## date (1): Deadline
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#-----------------------------------------------------
# Create a column with time between Launch and Deadline
# This is useful to determine how much time is allowed to raise funds
#-----------------------------------------------------
kickstarter_ds$DeadlineTime <- difftime(kickstarter_ds$Deadline, kickstarter_ds$Launched, units
= "hours")
head(kickstarter_ds)
```

ID	Name	Category
<dbl>	<chr>	<chr>
1860890148	Grace Jones Does Not Give A F$#% T-Shirt (limited Edition)	Fashion
709707365	CRYSTAL ANTLERS UNTITLED MOVIE	Film & Video
1703704063	drawing for dollars	Art
727286	Offline Wikipedia iPhone app	Technology
1622952265	Pantshirts	Fashion
2089078683	New York Makes a Book!!	Journalism

6 rows | 1-4 of 12 columns

```
#-----------------------------------------------------
# Show unique States of all Kickstarter projects
# Remove all State except Successful and Failed.
# The other states of projects are not useful for determining if Successful or Failed
#-----------------------------------------------------
kickstarter_unique_state <- unique(kickstarter_ds$State)
print(kickstarter_unique_state)
```

```
## [1] "Failed"     "Successful" "Canceled"   "Suspended"  "Live"
```

```
kickstarter_ds <- filter(kickstarter_ds, State != "Canceled")
kickstarter_ds <- filter(kickstarter_ds, State != "Live")
kickstarter_ds <- filter(kickstarter_ds, State != "Suspended")
kickstarter_state <- unique(kickstarter_ds$State)
print(kickstarter_state)
```

```
## [1] "Failed"     "Successful"
```

```
#-----------------------------------------------------
# Run a summary and view the column Headers
# View the columns and determine context
#-----------------------------------------------------
summary(kickstarter_ds)
```

```
##        ID                Name              Category         Subcategory
##  Min.   :5.971e+03   Length:331462      Length:331462      Length:331462
##  1st Qu.:5.372e+08   Class :character   Class :character   Class :character
##  Median :1.075e+09   Mode  :character   Mode  :character   Mode  :character
##  Mean   :1.074e+09
##  3rd Qu.:1.610e+09
##  Max.   :2.147e+09
##    Country             Launched                      Deadline
##  Length:331462      Min.   :2009-04-21 21:02:48   Min.   :2009-05-03
##  Class :character   1st Qu.:2013-03-29 18:12:33   1st Qu.:2013-05-01
##  Mode  :character   Median :2014-11-05 20:01:05   Median :2014-12-08
##                     Mean   :2014-09-02 06:35:12   Mean   :2014-10-05
##                     3rd Qu.:2016-03-05 18:32:53   3rd Qu.:2016-04-08
##                     Max.   :2017-12-29 03:22:32   Max.   :2018-01-02
##       Goal              Pledged            Backers             State
##  Min.   :        0   Min.   :       0   Min.   :     0.0   Length:331462
##  1st Qu.:     2000   1st Qu.:      50   1st Qu.:     2.0   Class :character
##  Median :     5000   Median :     788   Median :    15.0   Mode  :character
##  Mean   :    41523   Mean   :    9940   Mean   :   116.5
##  3rd Qu.:    15000   3rd Qu.:    4609   3rd Qu.:    63.0
##  Max.   :166361391   Max.   :20338986   Max.   :219382.0
##   DeadlineTime
##  Min.   :   0.1214 hours
##  1st Qu.: 698.1340 hours
##  Median : 711.7678 hours
##  Mean   : 801.3699 hours
##  3rd Qu.: 857.0214 hours
##  Max.   :2207.1036 hours
```

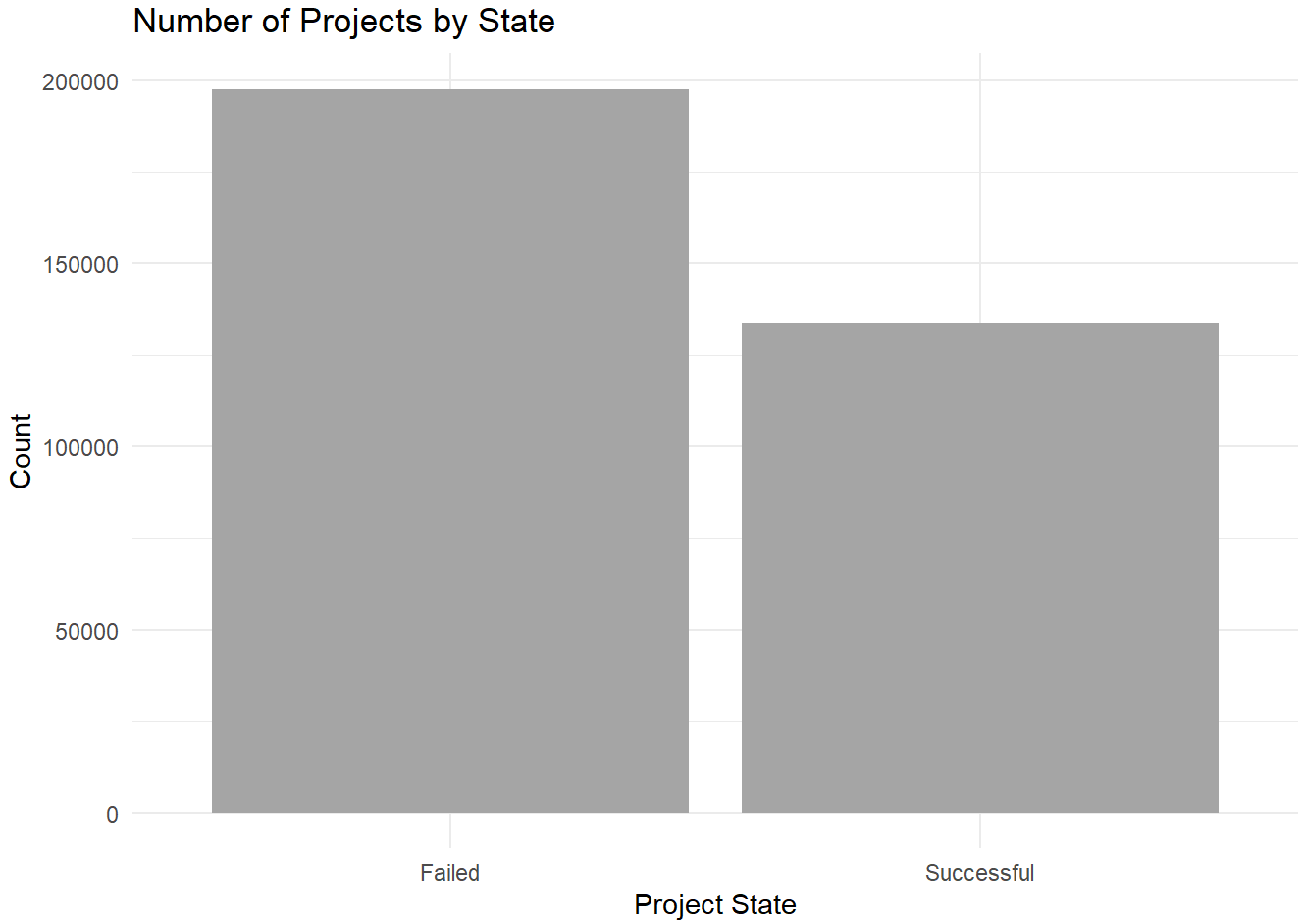
```
head(kickstarter_ds)
```

ID	Name	Category
<dbl>	<chr>	<chr>
1860890148	Grace Jones Does Not Give A F$#% T-Shirt (limited Edition)	Fashion
709707365	CRYSTAL ANTLERS UNTITLED MOVIE	Film & Video
1703704063	drawing for dollars	Art
727286	Offline Wikipedia iPhone app	Technology
1622952265	Pantshirts	Fashion
2089078683	New York Makes a Book!!	Journalism

6 rows | 1-4 of 12 columns

Basic Analysis of the Kickstarter Dataset

```
#-------------------------------------------------------
# Plot a bargraph of the count of Successful projects per Category
# Compares the number of successful vs. failed projects
#-------------------------------------------------------
ggplot(kickstarter_ds, aes(x = State)) +
  geom_bar(fill = "darkgray") +
  labs(title = "Number of Projects by State", x = "Project State", y = "Count") +
  theme_minimal()
```




```
#-------------------------------------------------------
# Evaluate unique values for each column, Country, Category and Subcategory
# Show what is unique for each of the categorical columns
#-------------------------------------------------------

kickstarter_unique_country <- unique(kickstarter_ds$Country)
print(kickstarter_unique_country)
```

```
##  [1] "United States"  "United Kingdom" "Canada"         "Australia"
##  [5] "New Zealand"    "Netherlands"    "Sweden"         "Denmark"
##  [9] "Norway"         "Ireland"        "Germany"        "France"
## [13] "Spain"          "Belgium"        "Italy"          "Switzerland"
## [17] "Austria"        "Luxembourg"     "Singapore"      "Hong Kong"
## [21] "Mexico"         "Japan"
```

```
kickstarter_unique_category <- unique(kickstarter_ds$Category)
print(kickstarter_unique_category)
```

```
##  [1] "Fashion"      "Film & Video" "Art"          "Technology"   "Journalism"
##  [6] "Publishing"   "Music"        "Photography"  "Games"        "Design"
## [11] "Theater"      "Crafts"       "Comics"       "Food"         "Dance"
```

```
kickstarter_unique_subcategory <- unique(kickstarter_ds$Subcategory)
print(kickstarter_unique_subcategory)
```

```
##   [1] "Fashion"           "Shorts"            "Illustration"
##   [4] "Software"          "Journalism"        "Fiction"
##   [7] "Rock"              "Photography"       "Puzzles"
##  [10] "Graphic Design"    "Film & Video"      "Publishing"
##  [13] "Documentary"       "Theater"           "Sculpture"
##  [16] "Electronic Music"  "Nonfiction"        "Painting"
##  [19] "Indie Rock"        "Public Art"        "Art"
##  [22] "Crafts"            "Jazz"              "Music"
##  [25] "Comics"            "Children's Books"  "Narrative Film"
##  [28] "Tabletop Games"    "Video Games"       "Digital Art"
##  [31] "Food"              "Animation"         "Conceptual Art"
##  [34] "Pop"               "Hip-Hop"           "Country & Folk"
##  [37] "Periodicals"       "Webseries"         "Product Design"
##  [40] "Performance Art"   "Art Books"         "World Music"
##  [43] "Knitting"          "Technology"        "Classical Music"
##  [46] "Graphic Novels"    "Poetry"            "Radio & Podcasts"
##  [49] "Design"            "Hardware"          "Webcomics"
##  [52] "Dance"             "Translations"      "Crochet"
##  [55] "Games"             "Photo"             "Mixed Media"
##  [58] "Space Exploration" "Photobooks"        "Musical"
##  [61] "Audio"             "Community Gardens" "R&B"
##  [64] "Fabrication Tools" "Textiles"          "Architecture"
##  [67] "Immersive"         "Literary Journals" "Spaces"
##  [70] "Video"             "Apps"              "DIY Electronics"
##  [73] "Academic"          "Experimental"      "Anthologies"
##  [76] "Plays"             "Video Art"         "Comic Books"
##  [79] "Letterpress"       "Couture"           "Robots"
##  [82] "Festivals"         "Installations"     "Sound"
##  [85] "Typography"        "Stationery"        "Camera Equipment"
##  [88] "Horror"            "Flight"            "Residencies"
##  [91] "Workshops"         "Chiptune"          "Civic Design"
##  [94] "Weaving"           "Young Adult"       "Web"
##  [97] "Makerspaces"       "Glass"             "Quilts"
## [100] "Pottery"           "Romance"           "Ceramics"
## [103] "Embroidery"        "Candles"           "DIY"
## [106] "Printing"          "Gadgets"           "Zines"
## [109] "Kids"              "Footwear"          "Pet Fashion"
## [112] "Events"            "Thrillers"         "Woodworking"
## [115] "Animals"           "Vegan"             "Ready-to-wear"
## [118] "Gaming Hardware"   "Movie Theaters"    "Accessories"
## [121] "Punk"              "Metal"             "Bacon"
## [124] "Family"            "Food Trucks"       "Fantasy"
## [127] "Places"            "Live Games"        "Science Fiction"
## [130] "Drama"             "Latin"             "Calendars"
## [133] "Television"        "Music Videos"      "Apparel"
## [136] "Comedy"            "Faith"             "Restaurants"
## [139] "Nature"            "Cookbooks"         "Drinks"
## [142] "3D Printing"       "People"            "Childrenswear"
## [145] "Farmer's Markets"  "Jewelry"           "Interactive Design"
## [148] "Fine Art"          "Blues"             "Wearables"
## [151] "Small Batch"       "Farms"             "Action"
```

```
## [154] "Performances"       "Playing Cards"      "Mobile Games"
## [157] "Taxidermy"          "Print"              "Literary Spaces"
```

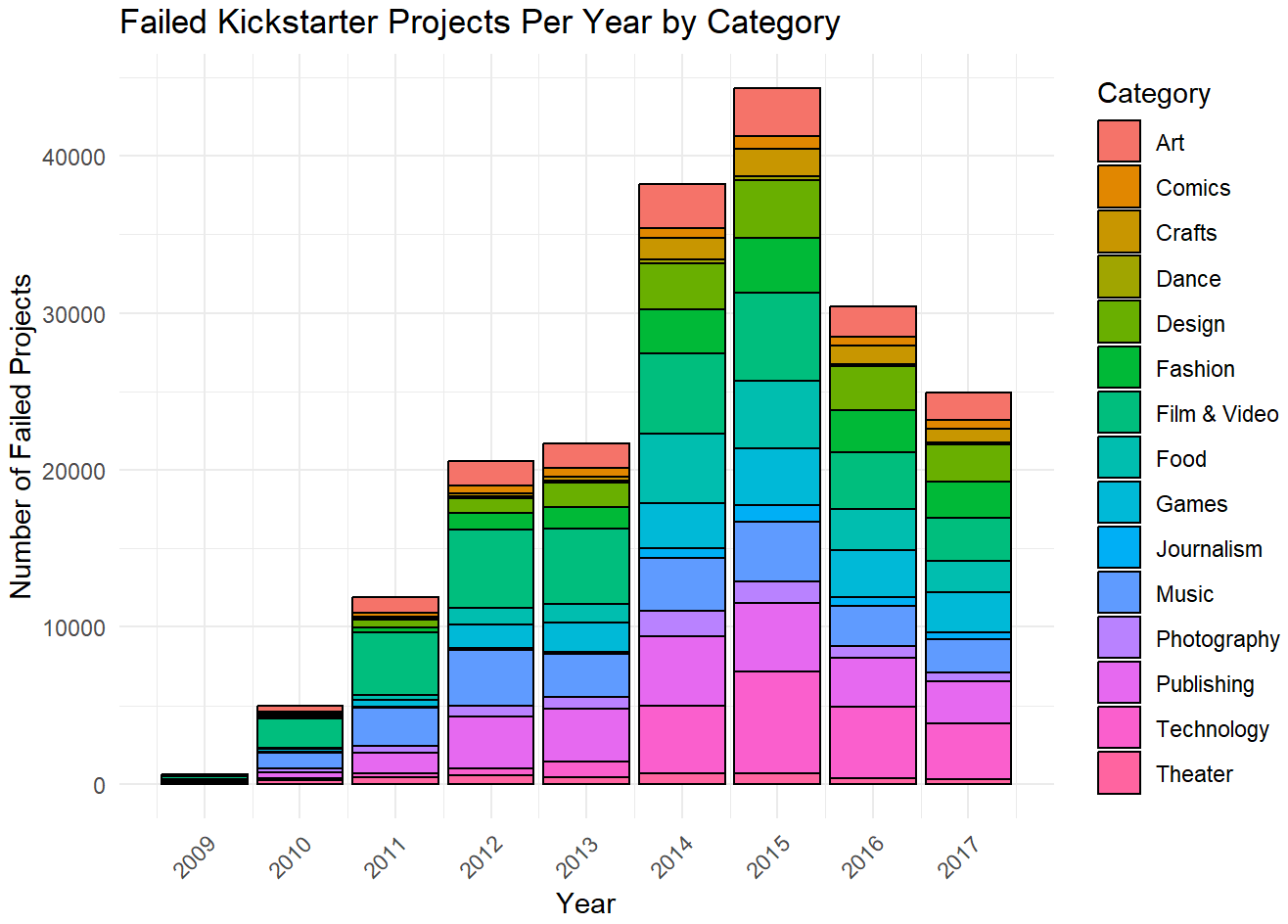
```
#-------------------------------------------------------
# Take a look at Failed projects
#-------------------------------------------------------
failed_projects <- kickstarter_ds %>% filter(State == "Failed")
failed_projects$Year <- year(failed_projects$Launched)
# Count failed projects by Year and Category
failed_by_year_category <- failed_projects %>%
  group_by(Year, Category) %>%
  summarise(Count = n())
```

```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

```
print(failed_by_year_category)
```

```
## # A tibble: 135 × 3
## # Groups:   Year [9]
##     Year Category     Count
##    <dbl> <chr>        <int>
##  1  2009 Art             70
##  2  2009 Comics          10
##  3  2009 Crafts           7
##  4  2009 Dance            4
##  5  2009 Design          18
##  6  2009 Fashion         20
##  7  2009 Film & Video   173
##  8  2009 Food             9
##  9  2009 Games           18
## 10  2009 Journalism      18
## # ℹ 125 more rows
```

```
#-------------------------------------------------------
# Plot a stack bar of count of Failed projects per Year
#
# Plot themes assisted by Claude 4
# (Sonnet 4.5) by Anthropic (2024)
# https://claude.ai
#-------------------------------------------------------

ggplot(failed_by_year_category, aes(x = Year, y = Count, fill = Category)) +
  geom_bar(stat = "identity" , color = "black") + # Border for each Category
  labs(
    title = "Failed Kickstarter Projects Per Year by Category",
    x = "Year",
    y = "Number of Failed Projects",
    fill = "Category"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) + # Year lables at 45 degrees
  scale_x_continuous(breaks = seq(min(failed_by_year_category$Year),
                                  max(failed_by_year_category$Year),
                                  by = 1))
```



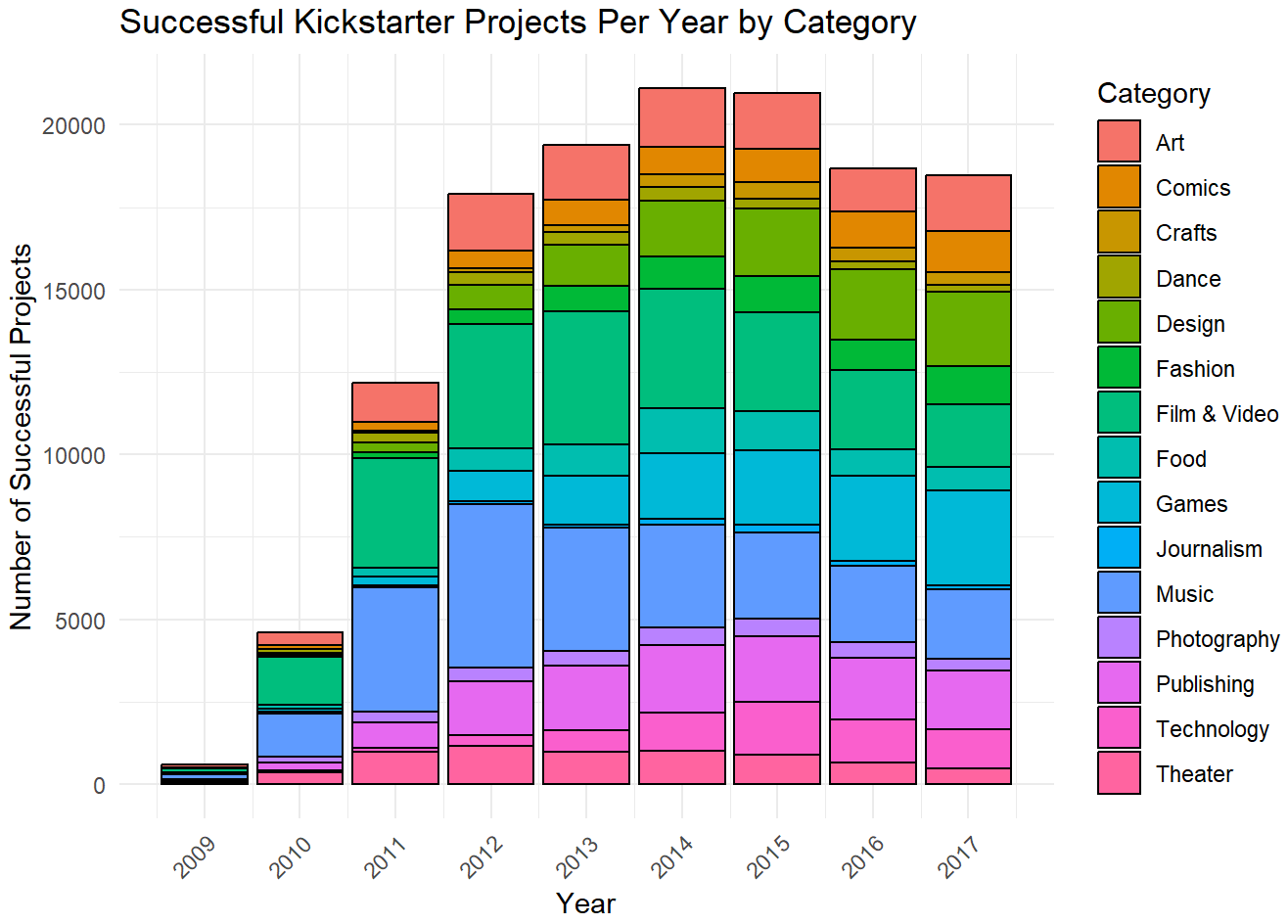
```
#-------------------------------------------------------
# Take a look at Successful projects
#-------------------------------------------------------

successful_projects <- kickstarter_ds %>% filter(State == "Successful")
successful_projects$Year <- year(successful_projects$Launched)

successful_by_year_category <- successful_projects %>%
  group_by(Year, Category) %>%
  summarise(Count = n())
```

```
## `summarise()` has grouped output by 'Year'. You can override using the
## `.groups` argument.
```

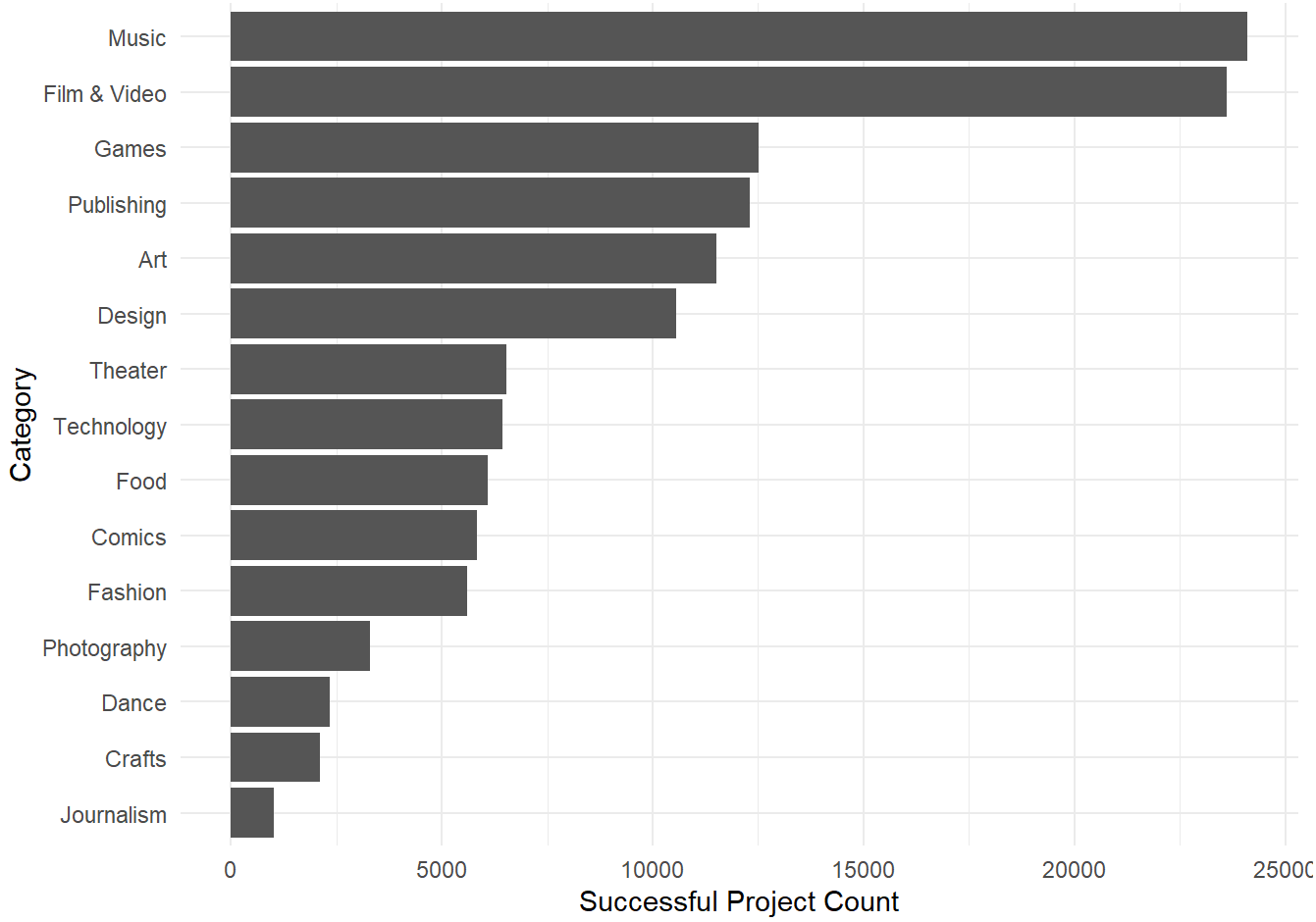
```
print(successful_by_year_category)
```

```
## # A tibble: 135 × 3
## # Groups:   Year [9]
##     Year Category     Count
##    <dbl> <chr>        <int>
##  1  2009 Art             77
##  2  2009 Comics          12
##  3  2009 Crafts           4
##  4  2009 Dance            3
##  5  2009 Design           8
##  6  2009 Fashion          4
##  7  2009 Film & Video   124
##  8  2009 Food            17
##  9  2009 Games           23
## 10  2009 Journalism      20
## # ℹ 125 more rows
```

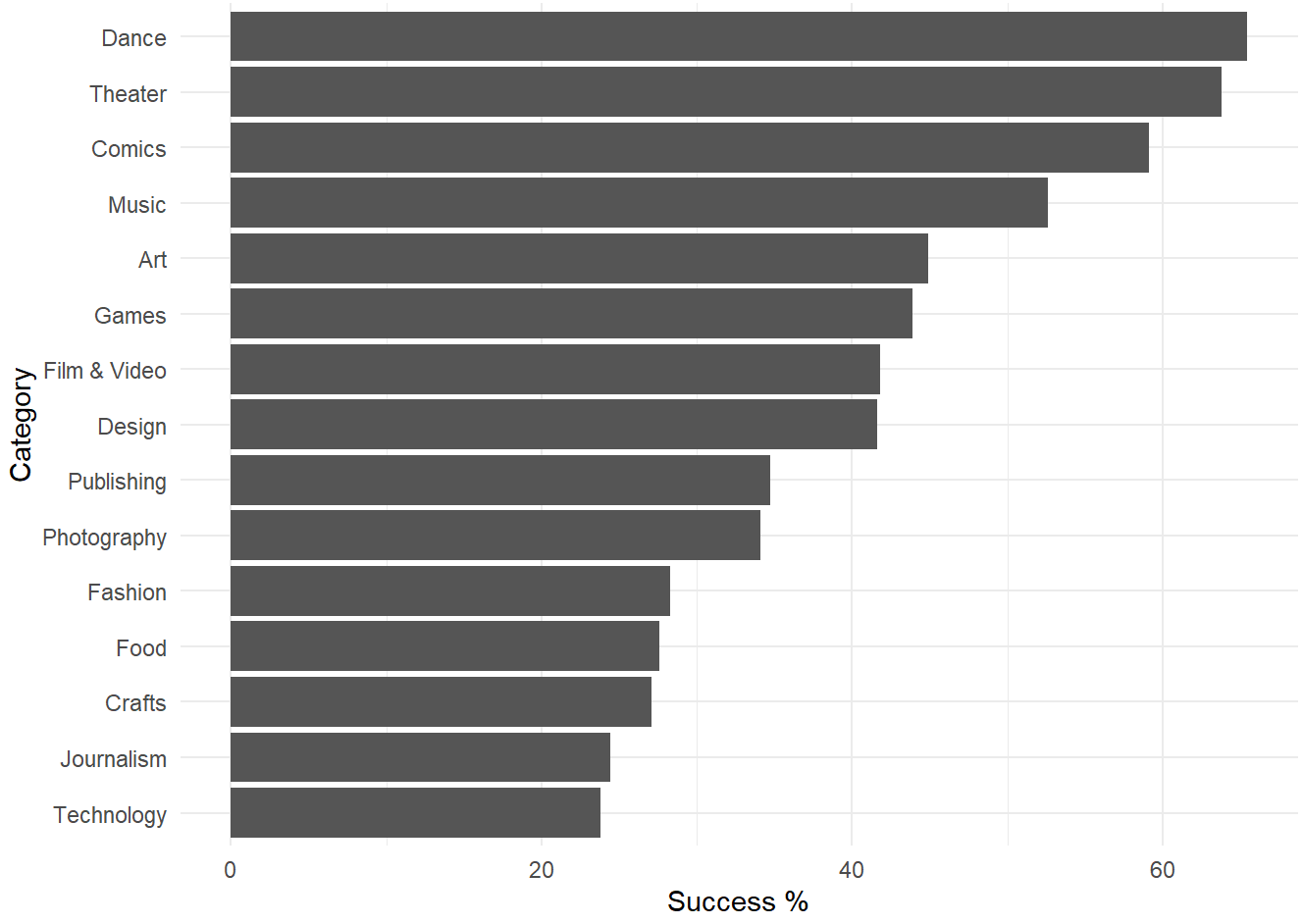
```
#-------------------------------------------------------
# Plot a stack bar of count of Successful projects per Year
#
# Plot themes assisted by Claude 4
# (Sonnet 4.5) by Anthropic (2024)
# https://claude.ai
#-------------------------------------------------------

ggplot(successful_by_year_category, aes(x = Year, y = Count, fill = Category)) +
  geom_bar(stat = "identity" , color = "black") + # stack with black borders per category
  labs(
    title = "Successful Kickstarter Projects Per Year by Category",
    x = "Year",
    y = "Number of Successful Projects",
    fill = "Category" # Legend
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +# Rotate 45 degree labels for x-axis
  scale_x_continuous(breaks = seq(min(successful_by_year_category$Year),
    max(successful_by_year_category$Year),
    by = 1))
```



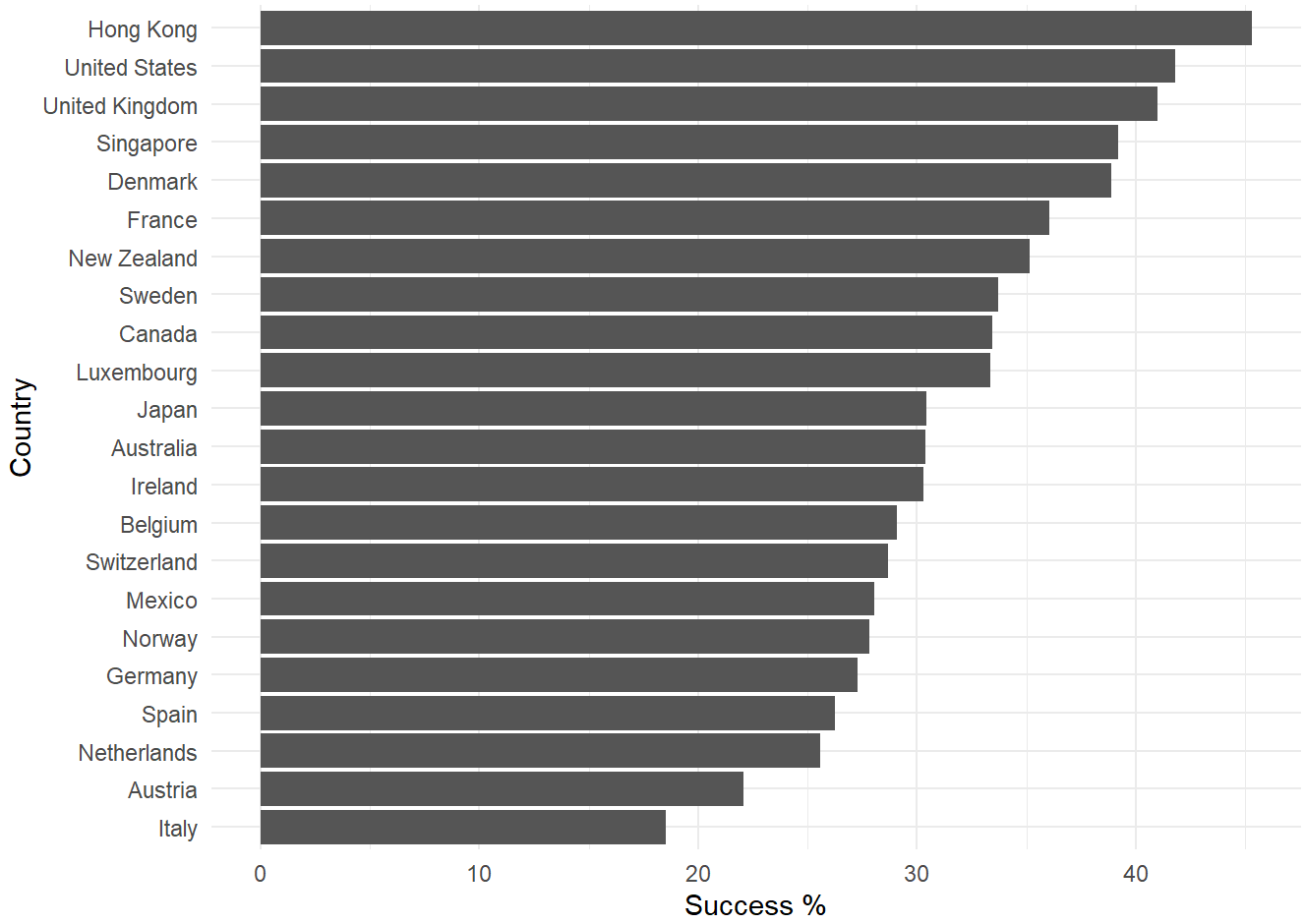
```
#---------------------------------------------------------
# Plot a bar graph of the count of Successful projects per Category
#---------------------------------------------------------

successful_by_category <- kickstarter_ds %>%
  filter(tolower(State) == "successful") %>%
  group_by(Category) %>%
  summarise(Count = n()) %>%
  arrange(desc(Count))

ggplot(successful_by_category, aes(x = reorder(Category, Count), y = Count)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(x = "Category", y = "Successful Project Count") +
  theme_minimal()
```



```
#-------------------------------------------------------
# Evaluate the Success percentage per Category
#-------------------------------------------------------

success_rate_per_category <- kickstarter_ds %>%
  group_by(Category) %>%
  summarise(
    Total = n(),
    Successful = sum(tolower(State) == "successful"),
    Success_Percentage = (Successful / Total) * 100
  ) %>%
  arrange(desc(Success_Percentage))

ggplot(success_rate_per_category, aes(x = Success_Percentage, y = reorder(Category, Success_Perc
entage))) +
  geom_bar(stat = "identity") +
  labs(x = "Success %", y = "Category") +
  theme_minimal()
```



```
#-------------------------------------------------------
# Now evaluate the Success percentage per Country
#-------------------------------------------------------
success_rate_by_country <- kickstarter_ds %>%
  group_by(Country) %>%
  summarise(
    Total = n(),
    Successful = sum(tolower(State) == "successful"),
    Success_Percentage = (Successful / Total) * 100
  ) %>%
  arrange(desc(Success_Percentage))

#-------------------------------------------------------
# Plot a bar graph of the percent Successful projects per Country
#-------------------------------------------------------
ggplot(success_rate_by_country, aes(x = Success_Percentage, y = reorder(Country, Success_Percent
age))) +
  geom_bar(stat = "identity") +
  labs(x = "Success %", y = "Country") +
  theme_minimal()
```



Clean up Memory

```
#-------------------------------------------------------
# Clean up some memory from successful and failed projects datasets
#-------------------------------------------------------
rm(failed_projects,successful_projects)
```

Data Model Preparation and Factorization

```
#-------------------------------------------------------
# Modify the Kickstarter dataset to a dataframe for preparing for applying models
# Drop columns that have no relevance to results.
#-------------------------------------------------------
kickstarter_df <- select(kickstarter_ds, Category, Country, Pledged, Backers,DeadlineTime, Stat
e)
head(kickstarter_df)
```

Category <chr>	Country <chr>	Pledged <dbl>	Backers <dbl>	DeadlineTime <drtn>	State <chr>
Fashion	United States	625	30	938.9533 hours	Failed
Film & Video	United States	22	3	2111.8686 hours	Failed
Art	United States	35	3	194.1325 hours	Successful
Technology	United States	145	25	1902.3942 hours	Successful
Fashion	United States	387	10	681.8225 hours	Failed
Journalism	United States	3329	110	418.0719 hours	Successful

6 rows

```
#-------------------------------------------------------
# To apply the data model the State needs to be changed to a binary 0 or 1 factor
# If you look at the columns it is changed to fct type
#-------------------------------------------------------

kickstarter_df$State <- factor(
  ifelse(tolower(kickstarter_df$State) == "successful", 1, 0),
  levels = c(0, 1),
  labels = c("Failed", "Successful")
)

#-------------------------------------------------------
# To apply the data models we need to change the character columns to factors
#-------------------------------------------------------

kickstarter_df$Category <- as.factor(kickstarter_df$Category)
kickstarter_df$Country <- as.factor(kickstarter_df$Country)

#-------------------------------------------------------
# Get rid of data that is NA within the Kickstarter dataframe
#-------------------------------------------------------

kickstarter_df <- na.omit(kickstarter_df)
summary(kickstarter_df)
```

```
##          Category                Country          Pledged
##  Film & Video: 56503   United States :261358   Min.   :       0
##  Music       : 45801   United Kingdom: 29453   1st Qu.:      50
##  Publishing  : 35413   Canada        : 12370   Median :     788
##  Games       : 28520   Australia     :  6616   Mean   :    9940
##  Technology  : 27046   Germany       :  3436   3rd Qu.:    4609
##  Art         : 25640   France        :  2520   Max.   :20338986
##  (Other)     :112539   (Other)       : 15709
##     Backers          DeadlineTime                    State
##  Min.   :     0.0   Min.   :   0.1214 hours   Failed    :197611
##  1st Qu.:     2.0   1st Qu.: 698.1340 hours   Successful:133851
##  Median :    15.0   Median : 711.7678 hours
##  Mean   :   116.5   Mean   : 801.3699 hours
##  3rd Qu.:    63.0   3rd Qu.: 857.0214 hours
##  Max.   :219382.0   Max.   :2207.1036 hours
##
```

Data Split into Test and Train

```
#---------------------------------------------------------
# It is going to be an 80-20 split for train test sets since we have enough data
#---------------------------------------------------------

set.seed(3)  # Set this so split is repeatable.
train_index <- createDataPartition(kickstarter_df$State, p = 0.8, list = FALSE)
train_data <- kickstarter_df[train_index, ]
test_data <- kickstarter_df[-train_index, ]

#---------------------------------------------------------
# Evaluate another summary and column heads for the train set
#---------------------------------------------------------

summary(train_data)
```

```
##          Category               Country          Pledged
##  Film & Video:45196   United States :209095   Min.   :       0
##  Music       :36590   United Kingdom: 23542   1st Qu.:      50
##  Publishing  :28439   Canada        :  9922   Median :     788
##  Games       :22910   Australia     :  5294   Mean   :    9907
##  Technology  :21562   Germany       :  2793   3rd Qu.:    4617
##  Art         :20540   France        :  2010   Max.   :20338986
##  (Other)     :89933   (Other)       : 12514
##     Backers          DeadlineTime                   State
##  Min.   :     0.0   Min.   :   0.1214 hours   Failed    :158089
##  1st Qu.:     2.0   1st Qu.: 698.1567 hours   Successful:107081
##  Median :    15.0   Median : 711.9483 hours
##  Mean   :   115.9   Mean   : 802.1750 hours
##  3rd Qu.:    63.0   3rd Qu.: 860.4274 hours
##  Max.   :219382.0   Max.   :2207.1036 hours
##
```

```
head(train_data)
```

Category	Country	Pledged	Backers	DeadlineTime	State
<fct>	<fct>	<dbl>	<dbl>	<drtn>	<fct>
Film & Video	United States	22	3	2111.8686 hours	Failed
Art	United States	35	3	194.1325 hours	Successful
Technology	United States	145	25	1902.3942 hours	Successful
Fashion	United States	387	10	681.8225 hours	Failed
Journalism	United States	3329	110	418.0719 hours	Successful
Film & Video	United States	41	3	717.9275 hours	Failed

6 rows

Apply the Random Forest Ranger Model to Train and Determine Accuracy

```
#---------------------------------------------------------
# Model 1: Random Forest Ranger Model
#---------------------------------------------------------
# For random forest ranger model to work, we need to define the controls and tuning
# Going to use cross-validation method, 10 fold, keep the probability estimates
# For tuning, mtry is generally sqrt of feature count, so 3 is good, split rule gini is default,
1 is minumum size
#---------------------------------------------------------

train_control <- trainControl(
  method = "cv",       # Using Cross-validation
  number = 10,         # Use 10 fold to increase accuracy
  classProbs = TRUE    # Get probability estimates
)

tune_grid <- expand.grid(
  mtry = 3,            # Number of column features to use at a time
  splitrule = "gini",  # Default index for split
  min.node.size = 1    # This is the minimum columns possible for classification
)

#---------------------------------------------------------
# Run the model on train set using control and tuning parameters
#---------------------------------------------------------

ranger_model <- train(
  State ~ Category + Country + Pledged + Backers + DeadlineTime,
  data = train_data,
  method = "ranger",          # Using ranger random forest algorithm
  trControl = train_control,
  tuneGrid = tune_grid,
  num.trees = 100,            # 100 trees to start, can increase to improve accuracy but takes l
onger
  importance = "impurity"     # Determine the importance of each variable to decision tree
)


#---------------------------------------------------------
# Show the results for the model parameters
#---------------------------------------------------------
print(ranger_model)
```

```
## Random Forest
##
## 265170 samples
##      5 predictor
##      2 classes: 'Failed', 'Successful'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 238653, 238653, 238653, 238652, 238653, 238653, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8641966  0.7214533
##
## Tuning parameter 'mtry' was held constant at a value of 3
## Tuning
##  parameter 'splitrule' was held constant at a value of gini
## Tuning
##  parameter 'min.node.size' was held constant at a value of 1
```

```
#-------------------------------------------------------
# Show the accuracy for train set
#-------------------------------------------------------
cat("Accuracy:", ranger_model$results$Accuracy)
```

```
## Accuracy: 0.8641966
```

Apply Ranger Model to Test Set and Determine Accuracy

```
#-------------------------------------------------------
# ranger_predict will make a prediction on success failed projects in test set
#-------------------------------------------------------

ranger_predict <- predict(ranger_model, newdata = test_data)
summary(ranger_predict)
```

```
##     Failed Successful
##      37505      28787
```

```
#-------------------------------------------------------
# Now apply the tuned ranger model to the test set
#-------------------------------------------------------

ranger_predict <- factor(ranger_predict, levels = levels(test_data$State))

#-------------------------------------------------------
# Run the confusion Matrix to determine if prediction is correct
#-------------------------------------------------------

ranger_confusionmatrix <- confusionMatrix(ranger_predict, test_data$State)
print(ranger_confusionmatrix)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Failed Successful
##   Failed      34066       3439
##   Successful   5456      23331
##
##                Accuracy : 0.8658
##                  95% CI : (0.8632, 0.8684)
##     No Information Rate : 0.5962
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7247
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8620
##             Specificity : 0.8715
##          Pos Pred Value : 0.9083
##          Neg Pred Value : 0.8105
##              Prevalence : 0.5962
##          Detection Rate : 0.5139
##    Detection Prevalence : 0.5658
##       Balanced Accuracy : 0.8667
##
##        'Positive' Class : Failed
##
```

```
#-------------------------------------------------------
# Show the accuracy result for the ranger model predictions
#-------------------------------------------------------

cat("Accuracy:", ranger_confusionmatrix$overall['Accuracy'])
```

```
## Accuracy: 0.8658209
```

Apply the Logistic Regression Model to the Train Set

```
#---------------------------------------------------------
# Model 2: Logistic Regression Model
#---------------------------------------------------------

#---------------------------------------------------------
# Run the logistic model on the train set
# We are going to suppress the warnings because we know the model will be highly successful with
some Pledge amounts
#---------------------------------------------------------

logistic_model <- suppressWarnings(glm(State ~ ., data = train_data, family = binomial))

#---------------------------------------------------------
# Apply the logistic model to the test set
# The State is a 0 and 1s factor, predict based on near value
#
# factor assisted by Claude 4
# (Sonnet 4.5) by Anthropic (2024)
# https://claude.ai
#---------------------------------------------------------
logistic_predict <- predict(logistic_model, test_data, type = "response")
logistic_predict <- factor(ifelse(logistic_predict > 0.5, "Successful", "Failed"),
                           levels = c("Failed", "Successful"))

summary(logistic_predict)
```

```
##     Failed Successful
##      45657      20635
```

Apply Logistic Regression Model to Test Set and Determine Accuracy

```
#---------------------------------------------------------
# Run the confusion matrix to determine if prediction is correct with the test set
#---------------------------------------------------------
logistic_confusionmatrix <- confusionMatrix(logistic_predict, test_data$State)
print(logistic_confusionmatrix)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Failed Successful
##   Failed      36993       8664
##   Successful   2529      18106
##
##                Accuracy : 0.8312
##                  95% CI : (0.8283, 0.834)
##     No Information Rate : 0.5962
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6359
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9360
##             Specificity : 0.6764
##          Pos Pred Value : 0.8102
##          Neg Pred Value : 0.8774
##              Prevalence : 0.5962
##          Detection Rate : 0.5580
##    Detection Prevalence : 0.6887
##       Balanced Accuracy : 0.8062
##
##        'Positive' Class : Failed
##
```

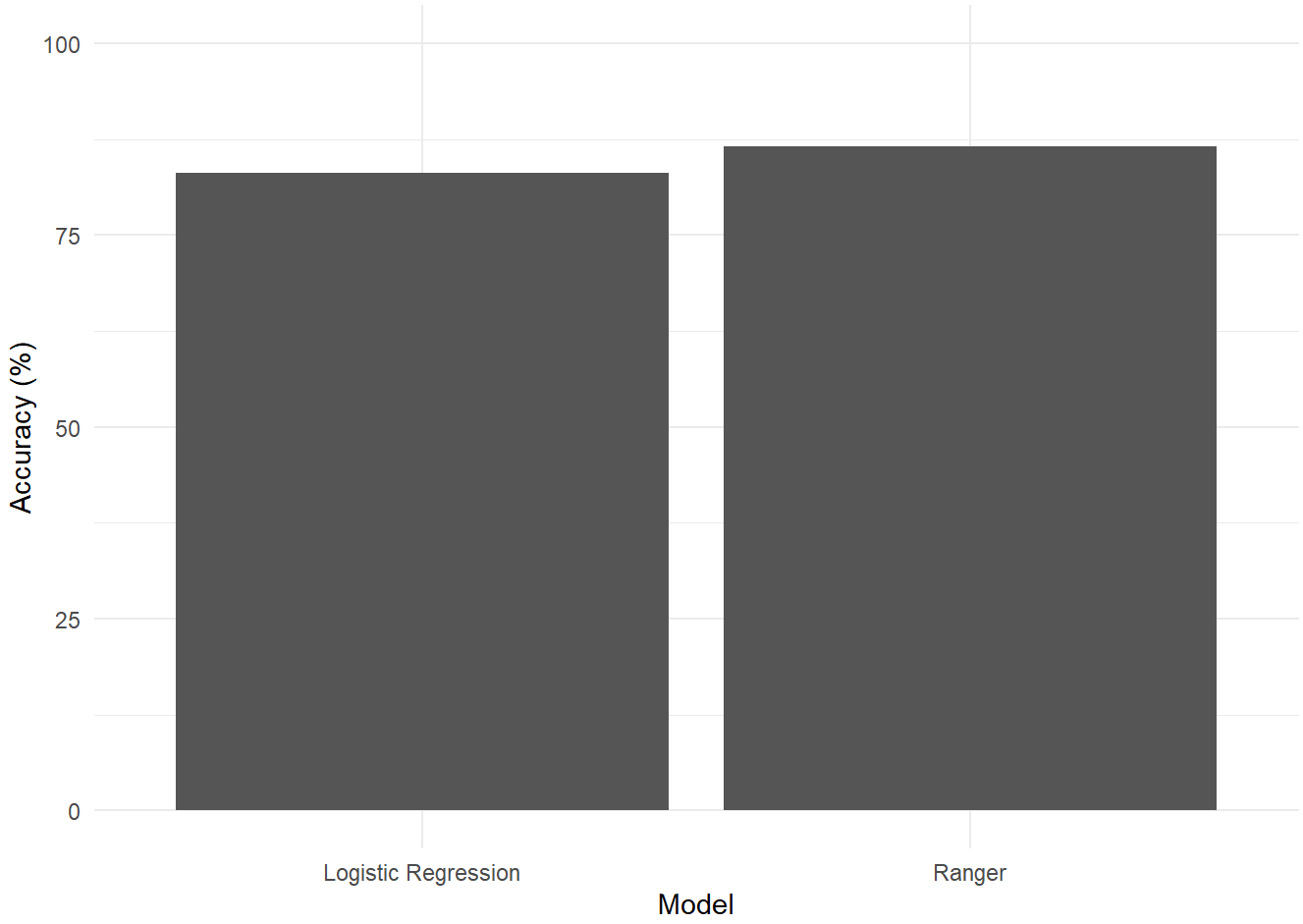
```
#---------------------------------------------------------
# Show the accuracy result for the logistic regression model
#---------------------------------------------------------

cat("Accuracy:", logistic_confusionmatrix$overall['Accuracy'])
```

```
## Accuracy: 0.8311561
```

Compare Accuracy for Ranger and Logistic

Regresssion Models

```
#-------------------------------------------------------
# Graph to compare models
#
# model_compare assisted by Claude 4
# (Sonnet 4.5) by Anthropic (2024)
# https://claude.ai
#-------------------------------------------------------

model_compare <- data.frame(
  Model = c("Ranger", "Logistic Regression"),
  Accuracy = c(
    ranger_confusionmatrix$overall['Accuracy'] * 100,    # Ranger accuracy as percentage
    logistic_confusionmatrix$overall['Accuracy'] * 100   # Logistic regression accuracy as perce
ntage

  )
)

ggplot(model_compare, aes(x = Model, y = Accuracy)) +
  geom_bar(stat = "identity") +
  ylim(0, 100) +
  labs(x = "Model", y = "Accuracy (%)") +
  theme_minimal()
```



Results

This project successfully determined that the Kickstarter dataset has enough column features to apply models to determine if a project has a likelihood of success. While the accuracy for the models was below 90%, there is significant variance in the categories to create a model that could assist kickstarter project predictions to increase success. It is likely that increasing folding or using another random forest algorithm is possible to increase the accuracy of predictions. The ranger and logistic regression models performed well, however they were scaled back to reduce the amount of time for convergence.

Conclusion and Summary

The random forest ranger algorithm performed well on the kickstarter dataset. The logistic regression algorithm can be used but appears it might have lower accuracy results. There was no expectation that the column features would be adequate to predict project success or failure. However, it was fascinating results considering we removed the Goal column feature to prevent the algorithms from having the ability to cheat and approach 100% accuracy.

Limitations

The capstone was limited to only using two data models. There is no dynamic tuning, dummy coding or one hot coding applied. The results are likely to significantly improve with some level of normalization, regularization, and adjusting the tuning and control parameters. The column for the Name of the projects was removed too. If there was additional context about the nature of the project, it is likely the prediction of the possible project success could drastically improve by introducing additional context and using sentiment analysis.

Future Work

Applying the insights from this capstone project can provide further analysis of similar data sets to predict success. This analysis of the kickstarter dataset can enhance the ability to consult project owners about how to adjust there probability of success by adjusting the project category type, amount to set for pledge per backer, country of origin for raising money and adjusting the velocity of the project.

References

Pedersen, U. T. (2018). Kickstarter Projects Dataset. https://www.kaggle.com/datasets/ulrikthygepedersen/kickstarter-projects (https://www.kaggle.com/datasets/ulrikthygepedersen/kickstarter-projects)

Anthropic. (2024). Claude 4 (Sonnet 4.5) [Large language model]. https://claude.ai (https://claude.ai)