Not So Standard Deviations

Kickstarter Analysis

Meredel Ababa

Nora Vasquez

Background:

Kickstarter was founded in 2009 and is based in Brooklyn, New York. It is a crowdfunding platform to “help bring creative projects to life.” It is essentially a website that offers “creators” and “backers” an innovative way to connect. So far over 14 million people have become backers and over 138 thousand projects have been funded. Kickstarter offers a platform for anyone to post a creative project within various categories. Projects are posted to one of eight categories. The categories are:

- Design & Tech
- Comics & Illustration
- Games
- Food & Craft
- Music
- Publishing
- Film
- Arts

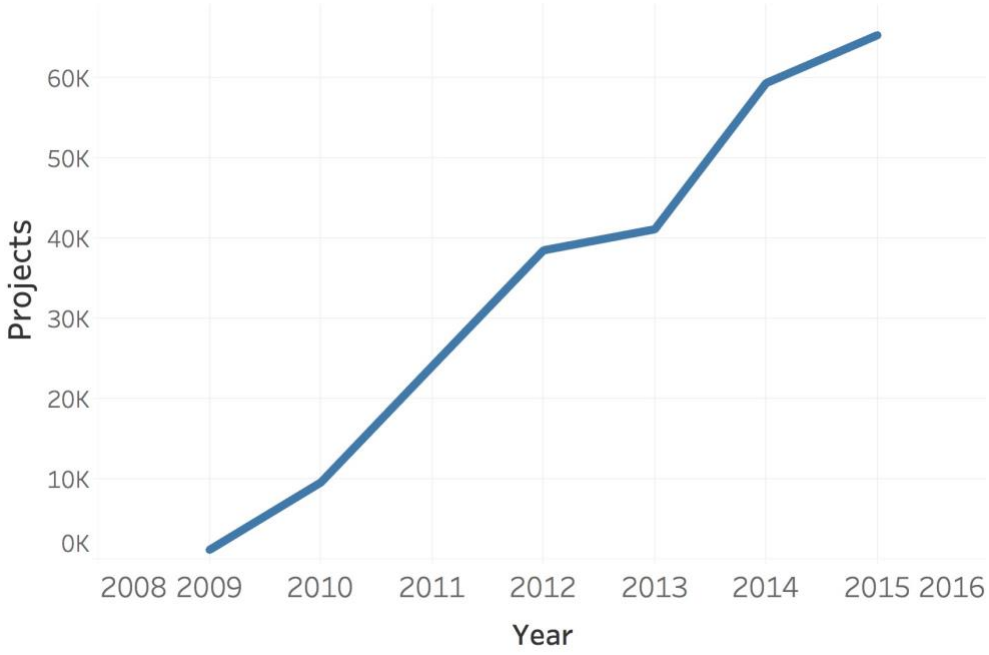
Creators post projects and choose a deadline, which can be up to 60 days. People who donate money to a project are considered “backers”. Additionally, the creator of the project chooses levels of rewards that the backers will receive in exchange for their donation to the project. They “pledge” a certain amount of money. It is considered a “pledge” because the money is not collected until the deadline of the goal. If the goal is reached, then the amount pledged is awarded to the creator of the project.

A few things make Kickstarter unique in comparison to other platforms. Backers are incentivized by being the first to “invest” in a project, the rewards are usually an exceptional value for the products. And backers get a first or special behind-the-scenes look into the process and development of the project. For example, when Cards Against Humanity launched, one of the rewards for pledging $15 or more was to receive: “the boxed game with professionally and laser-cut cards and also the digital version as soon as you pledge.” This game now retails for about $25. These types of rewards are enticing for both backers and creators. Having early insight into the beginnings of a company or concept is a great selling point. The large number of backers is a fertile environment for people looking for a place to find funding for their ideas. With

Kickstarter only taking 5% of pledges, this is attractive for artists/entrepreneurs. As well as for people who want to support said projects, knowing that most of their money will go towards the project and not Kickstarter. Creators of projects will post updates of the timeline on the project page to keep their backers involved in the project. Given that projects are not limited to only new creators, it is a great way for creators to establish rapport for subsequent projects.

Motivation:

The site draws thousands of people who both start and fund creative projects every year. Some are successful, and most are not. Both funders and project creators might be interested in this analysis to drive their decisions in funding or starting projects on the platform. Backers play an important role in getting projects funded. Therefore, creators will find it insightful of what features of the project can assist with getting pledges.



Data Sources:

We sourced our data from Kaggle. It was scraped by a Kickstarter enthusiast. The data collected ranges from April 2009 to January 2018. It consists of 378,661 rows/observations and 13 columns/features. The features are as follows:

- “Name” is the project name, this is chosen by the creator of the project.
- “Category” is a subset and a more specific section.
- “Main_category” is a more broader feature than Category
- “Currency” contains the type of currency the project chooses to be funded in.
- “Deadline” is when the project ends, this date is chosen by the creator of the project and is formatted in year-month-day and hour:minute:second.
- “Goal” contains the funding that the creator is asking the backers to pledge for the project. This is essentially the minimum amount of funding that the project needs to be classified as a success in our state column/feature.
- “Launched” is the date and time when the project was first live and is formatted in year-month-day and hour:minute:second as well as “Deadline”
- “Backers” is how many people pledged money to the project
- “Pledged” is the monetary amount for a project depending on the currency
- “Usd pledged” contains the information of the conversion of ‘pledged’ to U.S. dollars.
- “State” of the projects are classified as
    - ‘Failed’ - did not meet their minimum goal and no money is collected
    - ‘Successful’ - met their minimum goal at their deadline.
    - ‘Live’ - project is still in the process of getting backers and pledges, and has not reached the deadline
    - ‘Canceled’(sic) -(misspelled by the scraper of the data) creator has terminated the project before the deadline
    - ‘Suspended’ - Kickstarter reported or disqualified their project most likely for breach of the Terms of Service.
    - ‘Undefined’ - an error in input.

Data Inspection and Preprocessing:

We assessed for missing value and outliers, using summary statistics on all of our data. Our target variable “state” was transformed. First any non-desired state responses such as: “undefined”, “suspended”, “canceled” and “live” were converted to NULL and removed. This left us with only “failed” and successful.” In order to run the types of analysis we wanted, we converted to 0-failed and 1-successful.

The launch and deadline proved to be some of our most valuable data from which we derived several more features. These were useful once we applied a lubridate function to them which allows for arithmetic calculations on date. We extracted and created variables ‘launchseason’ and ‘deadlineseason.’ These variables include fall, winter, spring, summer. Numeric conversions were necessary for analysis so ‘launchseasonnum’ and ‘deadlineseasonnum’ were created as well. Lubridate was also used to create our ‘dayz’ variable which was the amount of days the project was live.

We removed rows that are still ‘live’ as we do not have the information if the project was successful or failed. In addition to removing ‘live’ from our state variable, we also removed observations that were not labeled as ‘successful’ and ‘failed’ which brought our overall number of observation from 378,661 down to 331,675. We assessed for outliers especially in the ‘usd pledge’ and ‘backers,’ but decided to leave them and choose a more flexible analytical approach if needed that may not rely on assumptions of normality across the variables.

Goals and Objectives of Analysis:

Hypothesis: Success of Kickstarter projects

Null: There is no statistically significant effect of the number of days the project is live on the success of a project.

Null: There is no statistically significant effect of the category the project is placed in on the success of a project.

Milestones:

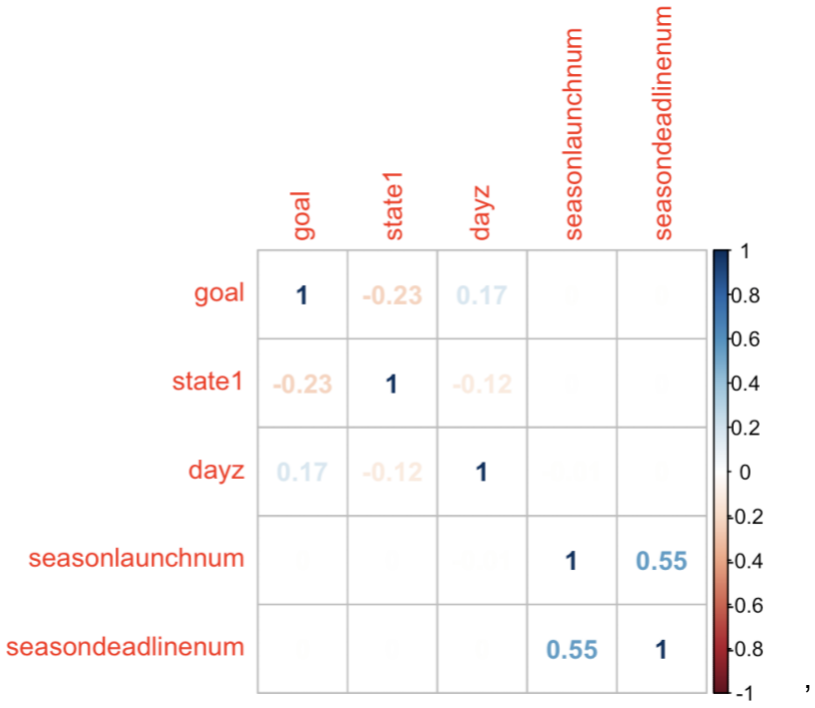
What are the best measures of success for a Kickstarter project?

When is the optimal time to post a project for success?

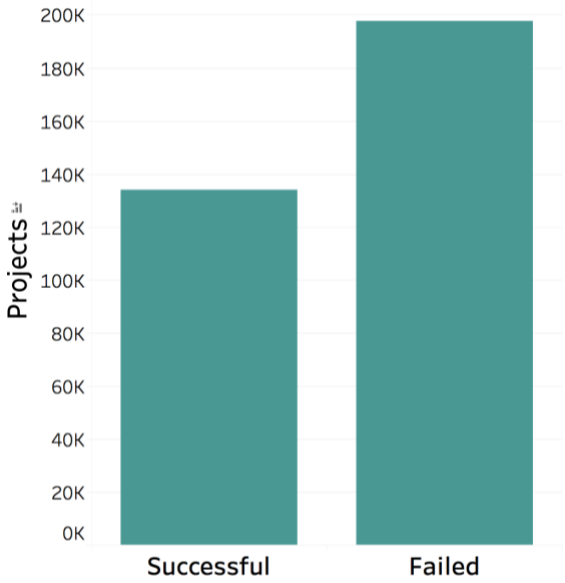
Does the length of time have an effect on the success of a project?

Exploratory/Preliminary Analysis Results:

Our correlation plot below shows that the independent variables are not correlated with each other so this allowed us to use these variables in our final KNN model.

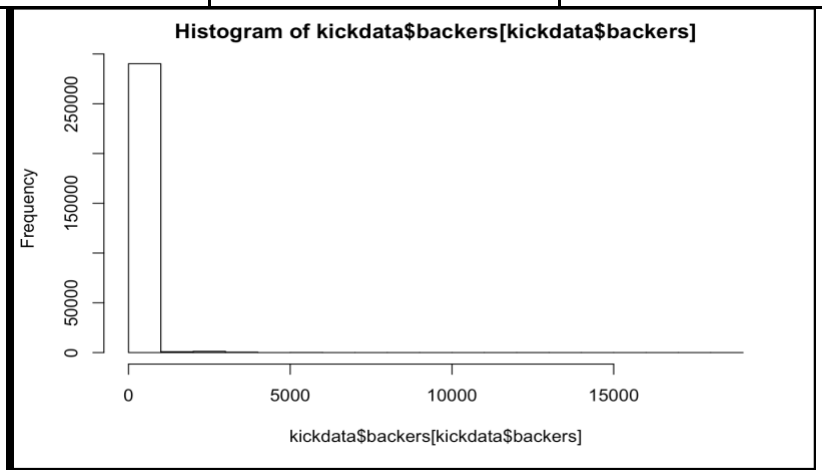


A bar graph including the total projects that were Failed and Successful was also created to assess the quantity of both responses. Our dependent variable, state, was not severely imbalanced and no advanced statistical techniques for balancing the dataset was utilized as shown by the figure below.

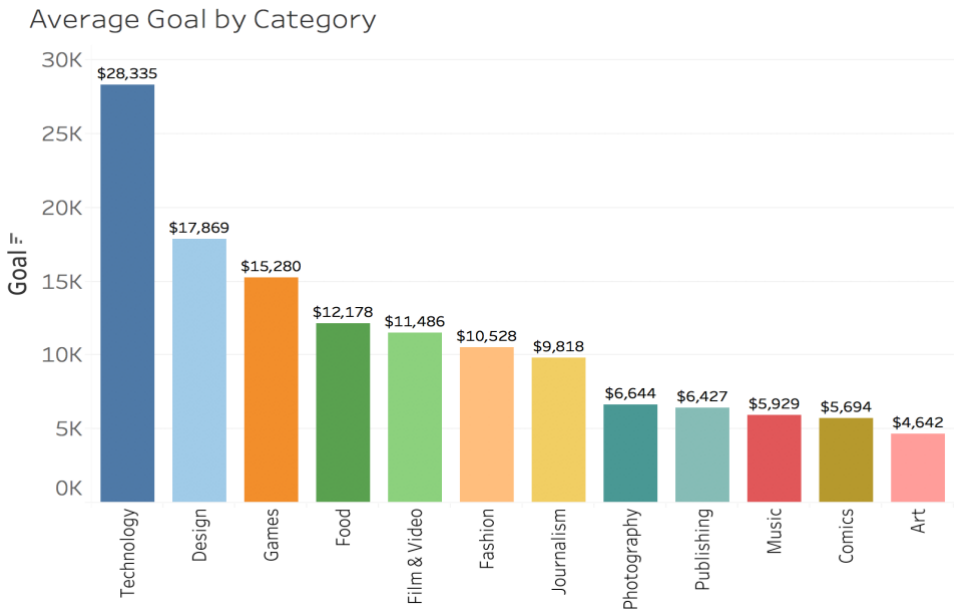


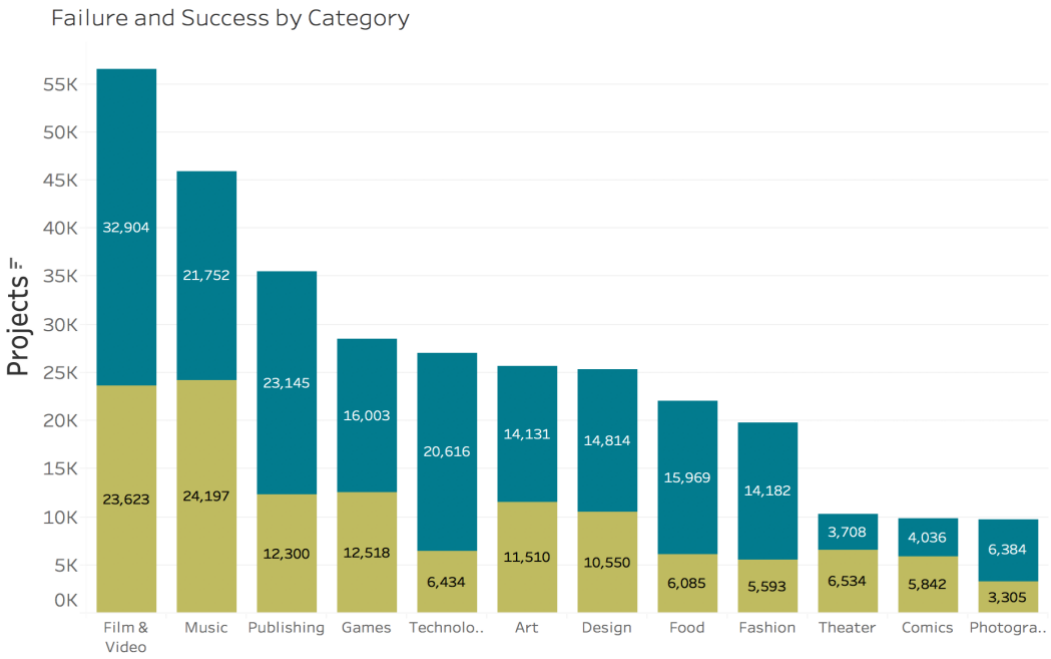
In the table below, our “backers” variable is highly right skewed. For our final model (KNN classification), “backers” was scaled. It was not ideal to use backers in other models such as logistic regression, as it violated the normality assumption. Also, trying to do log transformation was not practical given that there were a total of 38,847 zeros. Even after the removal of zeros, 1-10 backers is still where majority of our count was at 110,868 observations**.**

Backers		
Minimum	Median	Maximum
0	137.5	219,382



As we explored the features, we saw that goal could potentially play a major part in a project being successful. The first graph shows the average goal of a project with each category. The second graph shows the success and fail ratio of each category. It is interesting to point out that while Technology category has the largest average goal, unfortunately when compared to Failure and Success rate, Technology did not have a lot of successful projects.

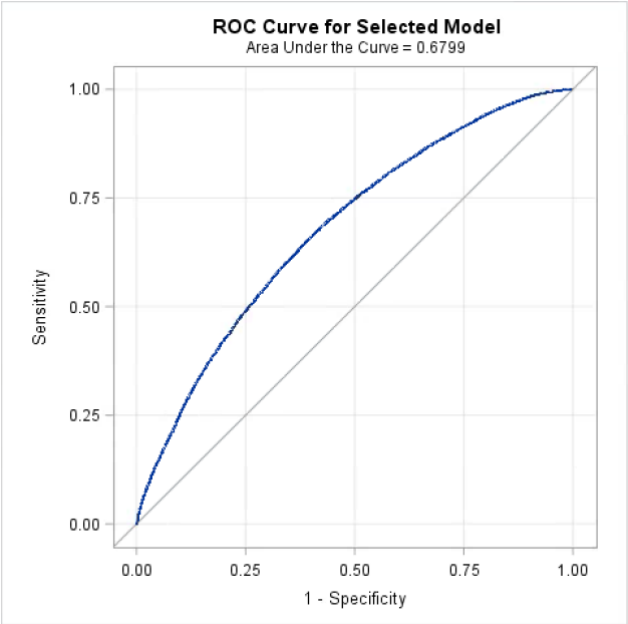




Analysis Results:

Logistic regression with forward selection was completed in SAS EG with these variables: ‘main_category’, ‘seasonlaunch’, ‘dayz’, ‘launched_month’, and ‘goal’ along with these interactions: dayz*launched_month, dayz*goal, and seasonlaunched*main_category. The cut off for the variable to be in the final model, was set to a significant level of $p < 0.05$. By computing the sensitivity and specificity the

model performed poorly at predicting success as evidenced by the AUC of 0.6799. Given that the model that our parameters found, performed poorly and was computationally expensive. We decided not to measure its performance on our test data set.



Using R, logistic regression was also attempted given the large data set. When 'backers' was removed from the model, logistic regression would give error probability of '1' or '0', the model did not converge. However, when we ran the model with 'backers', the results showed that it was significant with p value at <2e-16.

Another classification model considered for the dataset is Quadratic Discriminant Analysis. Assumption covariance between classes is not assumed with QDA. The best QDA misclassification error rate was 43.7% which included variables, 'category', 'season launch', 'season deadline', 'days', and 'deadline month'. However, we still wanted to assess the performance of the model by using cross-validation. The results of the misclassification error rate is 37% which means that this would be the best that our QDA model would perform with K=10.

Next, a non-parametric test, K Nearest Neighbors classification was performed. The state variable was first extracted from the dataset. Then data with the exception of 'state' was scaled for all numeric values. The variables that were scaled are 'backers', 'goal', 'dayz', 'seasonlaunch', and 'seasondeadline'. We set our seed to 123. The data was then split into training and test set, with our training at 70%, and our testing set at

30%. The KNN function was applied to the training set, and our K=1. With this model, misclassification error rate was 0.1133735.

A support vector machine was constructed which included all variables. It yielded great results on data that was not in it's best and most clean form. Unfortunately he SVM model ran overnight and it was too computationally expensive for us to run again in later phases of the project. We decided to focus our efforts on cleaning the data and trying attempting the other analyses.

The performance of our KNN model was assessed through 10 k-fold cross validation. It was set to find the error rate with k between 1:50. Our chosen seed remains at 123. According to the cross-validation results, the best k is at 33. And the minimum error rate 7.83%. Although a larger k value can create boundaries between the classes that are less distinct, it also reduces the effect of the noise on the classification. After consideration of model parsimony we decided on a k of 5.

Results, Interpretation and Data Products with Significance:

The goal was to find a model with the features which are best for success to aid a creator in creating a successful project. KNN does not specify which features contribute to model significance, therefore we are unable to distinguish which variables contributed to the low error rate of classification from our model. However given the information for all of these variables from a sample project, we can correctly classify it as successful or failed with high accuracy. Using forward selection in SAS EG we created a model that can be seen below. The probability modeled is 'state1' = 1.

Summary of Forward Selection

Step	Effect Entered	DF	Number In	Score Chi-Square	Pr > ChiSq
1	main_category	14	1	10716.1366	<.0001
2	dayz	1	2	3407.1561	<.0001
3	country	22	3	933.5816	<.0001
4	dayz*main_category	14	4	373.8520	<.0001
5	goal	1	5	109.0119	<.0001
6	goal*main_category	14	6	1287.9631	<.0001

Some methods that were tried included the removal of zeros from 'backers' and trying to categorize/group them. Given that our range was still 1-219,382, made it difficult to group them and use that new variable in a model. We did not want to remove

the large values since they are truly ‘successful’ and could provide insight into successful features. We also already had a disparity between successful and failed.

```
> a <- table(kickdata2$backers)
> a

    1     2     3     4     5     6     7     8     9    10    11    12    13    14
29736 20219 13947 10502  8470  7028  6148  5376  4871  4471  4022  3834  3696  3526
   15    16    17    18    19    20    21    22    23    24    25    26    27    28
 3323  3058  3091  2873  2852  2691  2565  2494  2418  2294  2321  2188  2107  2072
   29    30    31    32    33    34    35    36    37    38    39    40    41    42
 2006  1972  1890  1852  1784  1825  1705  1670  1664  1636  1565  1566  1471  1439
   43    44    45    46    47    48    49    50    51    52    53    54    55    56
 1425  1339  1306  1293  1258  1328  1124  1197  1137  1173  1144  1178  1106  1069
```

Although ‘backers’ was the only variable in our logistic regression we saw that it was significant, p value at <2e-16. Therefore, we can conclude that actionable recommendations would include be that creators need to focus their efforts largely on getting as many backers as possible to support their projects. Along with other featurizations that we were unable to access from our data set.

Future Analysis:

For future analysis, textual analysis might be of interest. On each project page there is a place for a description where the creator can go into details of the project, including, concept, milestones, and plans to execute the project plan. Considering that the description is where the project comes to life, there may be a correlation to a backer’s willingness to fund the project. Another source of information could be photos and videos. Analysis of these components could also help build a better model.

We would like to explore if there were any advertising for their projects, it would be interesting to see if they are correlated with success. Including market segmentation for the different types of projects to promote more successful projects. Also, as a business idea for Kickstarter, if we analyzed the information from market segmentation they could then use that to start an optional advertising service with a fee for the creators in hopes of creating more successful projects.

Reference and Work Cites:

- http://www.alexa.com/siteinfo/Kickstarter.com
- https://www.kaggle.com/kemical/kickstarter-projects/data
- https://thecreativeindependent.com
- https://en.wikipedia.org/wiki/Kickstarter

Appendix:

```{r}
kickdata = ks_projects_201801 # import data
kickdata$state1 <- kickdata$state # made the state1 column
kickdata$state1 <- gsub("canceled|undefined|live|suspended", "null", kickdata$state1) # replaced canceled, undefined, and live with null
kickdata <- kickdata[kickdata$state1 != "null", ]   # removed null
kickdata <- kickdata                # code that removes scientific notation.
options(scipen=999)  # removes scientific notation
Kickdata
kickdata$state1 = ifelse(kickdata$state =='failed', 0,1) # kickdata 0 is failed and 1 is successful
```

```{r}

# creating launched1 as lubridate object
kickdata$launched1 =          lubridate::as_date(kickdata$launched)
# creating deadline1 as lubridate object
kickdata$deadline1 = lubridate::as_date(kickdata$deadline)
# checking to see all variables
names(kickdata)
# creating new duration column
kickdata$dayz <- kickdata$deadline1 - kickdata$launched1 # called dayz
head(kickdata)
# seeing summary on new data set w/new columns
kickdata$l_time <- format(as.POSIXct(kickdata$launched) ,format = "%H:%M:%S")
#extracted time from launched column
kickdata$d_time <- format(as.POSIXct(kickdata$deadline) ,format = "%H:%M:%S")
#extracted time from deadline column
launched_month = month(as.POSIXlt(kickdata$launched1, format="%Y/%m/%d"))
#extract month from launched1 date
```

```
dead_month = month(as.POSIXlt(kickdata$deadline1, format="%Y/%m/%d")) #extract month from deadline1 date
kickdata=data.frame(kickdata, launched_month) #add launched month to df
kickdata=data.frame(kickdata, dead_month) #add deadline month to df
kickdata$launched_month=as.factor(kickdata$launched_month) # launched_month needs to be factor, so we convert factor
kickdata$dead_month=as.factor(kickdata$dead_month) #dead_month needs to be factor, so we convert to factor
```

```{r} # function that will take our date, transform to 2012 date, then find the season in which it occured
getSeason <- function(launched1) {
   WS <- as.Date("2012-12-15", format = "%Y-%m-%d") # Winter Solstice
   SE <- as.Date("2012-3-15",  format = "%Y-%m-%d") # Spring Equinox
   SS <- as.Date("2012-6-15",  format = "%Y-%m-%d") # Summer Solstice
   FE <- as.Date("2012-9-15",  format = "%Y-%m-%d") # Fall Equinox

   # Convert dates from any year to 2012 dates
   d <- as.Date(strftime(launched1, format="2012-%m-%d"))

   ifelse (d >= WS | d < SE, "Winter",
     ifelse (d >= SE & d < SS, "Spring",
       ifelse (d >= SS & d < FE, "Summer", "Fall")))
}

kickdata$launch2012 = as.Date(kickdata$launched1, format = "%Y-%m-%d") + 0:331674 #formatting to 2012
kickdata$seasonlaunch = as.factor(getSeason(kickdata$launch2012)) #use function to find season for launch
kickdata$launch2012 = NULL
```

```
kickdata$deadline2012 = as.Date(kickdata$deadline1, format = "%Y-%m-%d") +
0:331674 #formatting to 2012
kickdata$seasondeadline = as.factor(getSeason(kickdata$deadline2012))  # use
function to find season for deadline
kickdata$deadline2012 = NULL
```

```{r}
#histogram of backers
hist(kickdata$backers [kickdata$backers])
```
#assessing what type of variable it is.
```{r}
#seeing class of all our variables
class(kickdata$state1)
class(kickdata$backers)
class(kickdata$goal)
class(kickdata$l_time)
class(kickdata$d_time)
class(kickdata$launched_month)
class(kickdata$dead_month)
class(kickdata$name)
class(kickdata$country)
class(kickdata$state)
class(kickdata$usd.pledged)
class(kickdata$pledged)
class(kickdata$launched1)
class(kickdata$dayz)
```

```{r}
# dropping variables that won't be included in analysis
```

```
kickdata2 = kickdata
kickdata2$ID = NULL
kickdata2$name= NULL
kickdata2$currency= NULL
kickdata2$deadline= NULL
kickdata2$launched= NULL
kickdata2$launched1= NULL
kickdata2$deadline1 = NULL
kickdata2$state = NULL
kickdata2$pledged = NULL
kickdata2$usd.pledged=NULL #too correlated with goal
kickdata$launchyear = format(as.POSIXct(kickdata$launched), format = "%Y")
#extracting the correct format for launchyear
```

changing to factor

```{r}
# changing variables to factor
kickdata2$state1 <- factor(kickdata2$state1)
kickdata2$category <- factor(kickdata2$category)
kickdata2$main_category <- factor(kickdata2$main_category)
kickdata2$country <- factor(kickdata2$country)
kickdata2$l_time <- factor(kickdata2$l_time)
kickdata2$d_time <- factor(kickdata2$d_time)
kickdata2$launched_month <- factor(kickdata2$launched_month)
kickdata2$dead_month <- factor(kickdata2$dead_month)
kickdata2$dayz =as.numeric(paste(kickdata2$dayz))
kickdata2$backers =as.numeric(paste(kickdata2$backers))
```

split data to training and test set. Unfortunately, will not run.

```{r}
set.seed(123)
```

```
train <- sample(1:331675,331675*.70,rep=FALSE) # creating training set of data , 70%
test <- -train # test set, will be 30%, whatever is left from training set
kick.train <- kickdata2[train, ] #
kick.test <- kickdata2[-train, ]
#log.regression = glm(state1~ goal+launched_month, data = kick.train, family =
binomial(link="logit"))
#summary(log.regression)
```

Error rate is .437826

```{r}
qda_model = qda(state1~ launched_month+category+goal+seasonlaunch+dayz, data =
kick.train) #QDA model
qda_pred = predict(qda_model, kick.test)  # run the model on our test set
qda_pred_y = qda_pred$class
table(qda_pred_y, kick.test$state1)
mean(qda_pred_y != kick.test$state1)
```

#Better QDA model, but not by much. Error rate is .4370521

```{r}
library(MASS)
qda_model = qda(state1~ category+ seasondeadline+goal+seasonlaunch+dayz, data =
kick.train)
qda_pred = predict(qda_model, kick.test)
qda_pred_y = qda_pred$class
table(qda_pred_y, kick.test$state1)
mean(qda_pred_y != kick.test$state1)
```

#Performed even worse with error rate of .5602

```{r}
class(kickdata2$launched_month)
qda_model = qda(state1 ~ main_category + seasondeadline + goal +
seasonlaunch+dayz+dead_month+launched_month, data = kick.train)
```

```
qda_pred = predict(qda_model, kick.test)
qda_pred_y = qda_pred$class
table(qda_pred_y, kick.test$state1)
mean(qda_pred_y != kick.test$state1)
```

Cross validation on QDA : error rate is 34.215, with a K at 10

```{r}
CV.qda <- #create CV function
  function (data, model, dname, K=10, seed=150) {
    n <- nrow(data)
    set.seed(seed)
    dataY=data[,dname]

    f <- ceiling(n/K)
    s <- sample(rep(1:K, f), n)

    CV=NULL

    for (i in 1:K) {
      test.out <- seq_len(n)[(s == i)] #test data
      train.in <- seq_len(n)[(s != i)] #training data



      qda.fit=lda(model, data=data[train.in,])
      qda.y <- data[test.out, dname]
      qda.predy=predict(qda.fit, data[test.out,])$class

      error= mean(qda.y!=qda.predy)
      CV=c(CV,error)
    }
    list(call = model, K = K,
         qda_error_rate = paste(100*mean(CV), "%"), seed = seed)
```

```
  }
err.qda=CV.qda(data = kickdata2, state1~main_category + seasondeadline + goal + seasonlaunch+dayz+dead_month+launched_month, dname = "state1", K=10, seed = 123) #checking error
err.qda
```

#extracted year from launched so we can use that to compare projects by year

```{r}
kickdata$launchyear = format(as.POSIXct(kickdata$launched), format = "%Y")
```

```{r}
library(FNN)
kickdata3=kickdata2
kickdata3$category <- NULL
kickdata3$main_category <- NULL
kickdata3$country <- NULL
kickdata3$l_time <- NULL
kickdata3$d_time <- NULL
kickdata3$seasonlaunch <- NULL
kickdata3$seasondeadline <- NULL
kickdata3$X <- NULL
kickdata3$dayz =as.numeric(paste(kickdata3$dayz))
kickdata3$backers =as.numeric(paste(kickdata3$backers))
kickdata3$goal =as.numeric(paste(kickdata3$goal))
kickdata3$launched_month =as.numeric(paste(kickdata3$launched_month))
kickdata3$dead_month =as.numeric(paste(kickdata3$dead_month))
kickdata3$state1 =as.numeric(paste(kickdata3$state1))
```

```{r}
data = scale(kickdata3[,-c(3,4)]) # scale the variable; while also removing the predictor (state1)
set.seed(123) #setting seed
```

```
train <- sample(1:331675,331675*.7,rep=FALSE) # creating the training data set 70%
test <- -train # creating test data set
training_kick = data[train, ]
testing_kick = data[test,]
summary(data)
```

#includes state variable

```{r}
training_y = kickdata3$state1[train] # training set
testing_y = kickdata3$state1[test] # testing set
```

```
library(class) #to be able to run kin()
set.seed(123) #setting the seed
knn_pred_y = knn(training_kick, testing_kick, training_y, k = 1) # KNN
#dim(training_data)
#length(training_y)
table(knn_pred_y, testing_y) #Provides the confusion matrix
mean(knn_pred_y != testing_y) #Misclassification error rate at 0.1133735
```

Created a CV KNN Function

```{r}
knn_pred_y = NULL
error_rate = NULL
# creating function for KNN and getting error rate
for(i in 1:50){
set.seed(123)
knn_pred_y = knn(training_kick,testing_kick,training_y,k=i)
error_rate[i] = mean(testing_y != knn_pred_y)
}

### find the minimum error rate
min_error_rate = min(error_rate)
```

print(min_error_rate)

get the index of that error rate, which is the k
K = which(error_rate == min_error_rate)
print(K)
```

To visualize the misclassification error rate

```{r}
library(ggplot2)
qplot(1:300, error_rate, xlab = "K",
ylab = "Error Rate",
geom=c("point", "line"))

```

```
/*Logistic Forward Selection*/
/* ---------------------------------------------------------------------
   Code generated by SAS Task

   Generated on: Sunday, January 28, 2018 at 7:46:13 PM
   By task: Logistic Regression  (2)

   Input Data: Local:WORK.KICKDATA_TRAIN
   Server:  Local
   --------------------------------------------------------------------- */
ODS GRAPHICS ON;

%_eg_conditional_dropds(WORK.SORTTempTableSorted);
/* ---------------------------------------------------------------------
   Sort data set Local:WORK.KICKDATA_TRAIN
   --------------------------------------------------------------------- */

PROC SQL;
       CREATE VIEW WORK.SORTTempTableSorted AS
              SELECT T.state1, T.dead_month, T.launched_month, T.dayz, T.goal,
T.seasondeadline, T.main_category, T.country, T.seasonlaunch, T.category
       FROM WORK.KICKDATA_TRAIN as T
;
QUIT;
TITLE;
TITLE1 "Logistic Regression Results";
FOOTNOTE;
FOOTNOTE1 "Generated by the SAS System (&_SASSERVERNAME, &SYSSCPL) on
%TRIM(%QSYSFUNC(DATE(), NLDATE20.)) at %TRIM(%SYSFUNC(TIME(), TIMEAMPM12.))";
PROC LOGISTIC DATA=WORK.SORTTempTableSorted
              PLOTS(ONLY)=ALL
       ;
```
```

```
       CLASS seasondeadline (PARAM=EFFECT) main_category       (PARAM=EFFECT) country
       (PARAM=EFFECT) seasonlaunch (PARAM=EFFECT) category     (PARAM=EFFECT);
       MODEL state1 (Event = '1')=dead_month launched_month dayz goal seasondeadline
main_category country seasonlaunch goal*main_category dead_month*dayz
launched_month*seasondeadline main_category*seasonlaunch dayz*main_category
dayz*seasonlaunch dayz*goal seasonlaunch*category       /
              SELECTION=FORWARD
              SLE=0.05
              INCLUDE=0
              LINK=LOGIT
       ;
RUN;
QUIT;

/* ---------------------------------------------------------------------
   End of task code
   --------------------------------------------------------------------- */
RUN; QUIT;
%_eg_conditional_dropds(WORK.SORTTempTableSorted);
TITLE; FOOTNOTE;
ODS GRAPHICS OFF;
```