Kickstarting Your Kickstarter: Creating a Linear Regression Model to Predict the Success of Crowdfunding Campaigns

Connor Gorry, Faraz Shaikh, Kenny Cheng, & Sydney Kessler
COGS 109: Data Modeling and Analysis

Contents

Introduction

Among the recent trend of recent crowdfunding websites, Kickstarter is by far the most popular. Per their website, "[Kickstarter's] mission is to bring creative projects to life." Throughout the company's history, nearly 150,000 projects have been successfully funded, bringing countless products, projects, and pieces of art to life that may have never seen the light of day without access to funding. But for every project that is successful on Kickstarter, nearly two projects fail. For a donor, backing a Kickstarter project does not guarantee the delivery of a final product.

In this analysis, we will be using linear regression to create a model based on past Kickstarter projects, to see whether we can predict the potential success of a campaign, even before it begins.

Data-set Information

Our dataset comes in the form of a compilation of 378,662 individual Kickstarter campaigns, compiled and published by Kaggle user Mickaël Mouillé.

Each sample has 15 features associated with it, which will be detailed in the following section. These features provide background on each project – its country of origin, its original asking goal, the amount of money it raised, the duration of the campaign, and whether or not it reached its goal. In order to improve the precision of our model, we have created some additional features as well.

Before analyzing our dataset, we did some initial processing and clean up. First, we made the decision to focus our data analysis exclusively on the US, where the vast majority of Kickstarter campaigns are based. Although this shrunk our sample size, it simplified our process by eliminating a need to convert between multiple international currencies, as well as cutting down on outside factors that might affect our data, such as the added price of international shipping skewing product sales. Likewise, we removed any campaigns that were canceled or suspended, as they were likely to skew our data.

Ultimately, we analyzed a modified dataset of 286,608 samples, with a total of 22 features. To create our test and training sets, we randomized our modified array based on campaign ID number, which is assigned randomly by Kickstarter. We used 80% of our total data as our training set, saving the remaining 20% of our data to be used as our test set.

Attribute Information

We based our analysis upon the features listed below.

Utilized features

Attribute	*Explanation*	*Variable Used*
Index	The index of each campaign, from 1 to 289,837	betterIndex
Goal	The USD goal of each campaign. When a campaign reaches the goal that its creators have set in dollars pledged, the campaign is considered to be a success.	goal
Percent Funded* (We will be using this as our predicted value)	The percentage value to which the campaign was funded. This value was calculated by dividing Pledged by Goal. Any campaign with a value of over 1.0 is considered successful.	percentFunded
Backers	The number of donors.	backers
Rate of Backers*	The rate at which backers were donating to campaigns. This value was calculated by dividing Backers by Campaign Duration.	rateOfBackers
Campaign Duration*	The duration of the campaign. This value was calculated by subtracting the launch date from the deadline, using MATLAB's date-time formatting.	duration
Category	A set of 15 attribute columns for each of Kickstarter's main categories. A binary value represents whether a campaign falls within that category.	categoriesLogical
Interaction of Duration and Backers	Interaction shows the influence between two variables that is not necessarily additive. Here, we look for the interaction of Duration and Backers taken in tandem.	Duration*Backers
Interaction of Goal and Backers	The interaction of Goal and Backers, taken in tandem.	Goal*Backers
Interaction of Goal and Duration	The interaction of Goal and Duration, taken in tandem.	Goal*Duration

*features modified from or calculated from the original dataset. A similar table illustrating all of the features originally present in the dataset can be found in the appendix, along with the code used to create our new features.

Research Question

Can we predict if a Kickstarter campaign will be successful (based on percentage funded)? What features contribute to a kickstarter campaigns success?

Methodology

Below are the linear regression models that we utilized for our analysis. We built these models by adding features to the previous models to see which features had the largest effect on SSE.

M1: $\text{percentFunded} = w_0 + w_1(\text{Backers}) + w_2(\text{Duration}) + w_3(\text{rateOfBackers})$

M2: $\text{percentFunded} = w_0 + w_1(\text{Backers}) + w_2(\text{Duration}) + w_{3\text{-}18}(\text{Categories})$

M3: $\text{percentFunded} = w_0 + w_1(\text{Backers}) + w_2(\text{Goal}) + w_3(\text{Duration}) + w_{4\text{-}19}(\text{Categories})$

M4: $\text{percentFunded} = w_0 + w_1(\text{Backers}) + w_2(\text{Goal}) + w_3(\text{Goal})^2 + w_4(\text{Duration}) + w_5(\text{Duration}) + w_{6\text{-}21}(\text{Categories})$

M5: $\text{percentFunded} = w_0 + w_1(\text{Backers}) + w_2(\text{Goal}) + w_3(\text{Goal})^2 + w_4(\text{Duration}) + w_5(\text{Duration*Backers}) + w_6(\text{Goal*Backers}) + w_7(\text{Goal*Duration}) + w_{8\text{-}23}(\text{Categories})$

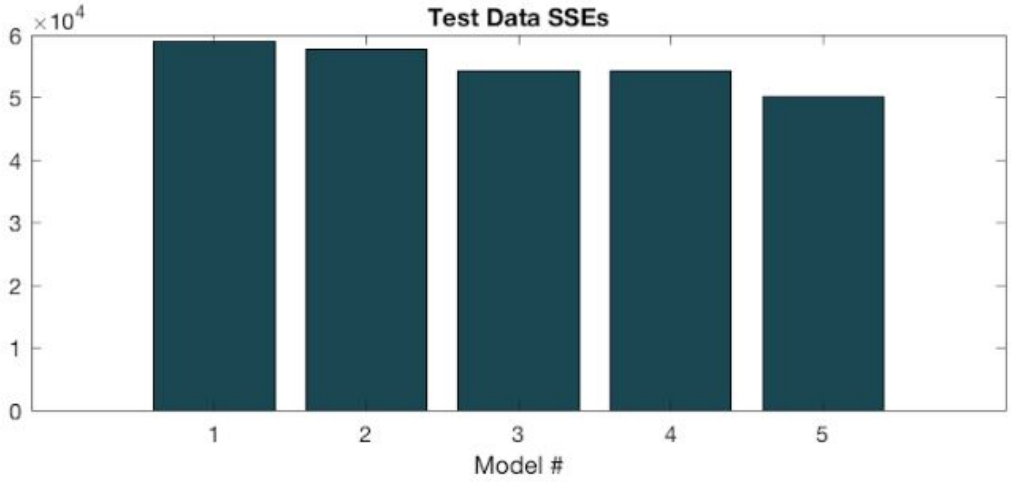
We separated our data into a training set and test set. The training set was created using 80% of the total dataset, while the test set consisted of the remaining 20%. We developed the weights of the models by fitting them to the training set, and tested them against the training set. We calculated the sum squared error (SSE) values of each model and compared them in order to assess the strength of each model.

During our analysis, we encountered a few problems. For Model 2, although the input and methodology were correct, we got a “Rank deficiency” error. This was due to the high volume of zero values in the Categories vectors - some Kickstarter categories, such as Dance and Journalism, have a very small amount of projects. In order to eliminate the Rank deficiency error, we added Gaussian noise to the Categories section. We continued to do this for the following models (which all included Category features).

Additionally, during our first run of linear regression we found that our SSEs were inordinately large. This was primarily caused by a small number of outliers that exceeded their goal by far over a thousand percent; because of their size, these outliers had a huge effect on the SSEs. To account for this, we eliminated any campaigns that exceeded one thousand percent of their goal - approximately 0.1% of the data. This enabled us to be able to perform a regression analysis that more appropriately reflects the rest of the dataset.

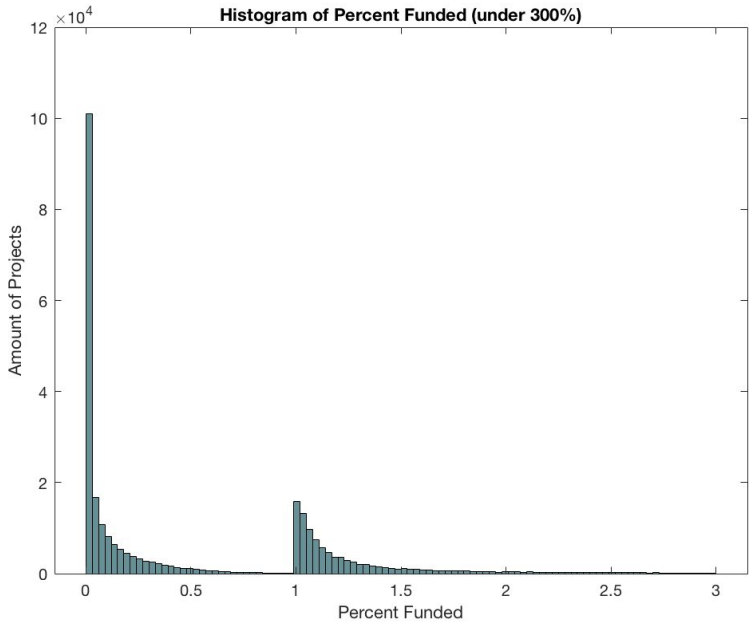
Error/Results

We were able to develop five different models using different variations of the features in the dataset.



Initially, we had predicted the categories feature would have some effect on SSE and while it did, the SSEs were only marginally different (see Model 1 vs. Model 2). Goal amount was introduced in Model 3; we expected this to be a stronger predictor, and it had more of a significant effect than the categories (Models 1&2 vs. Model 3). Due to the better results of Model 3, in Model 4, we added a feature for squared Goal amount, but this barely had any affect (Model 3 vs. Model 4).

Model 5 - which accounted for the number of backers, goal amount, and duration of the projects as well as the interactions between these features - had the smallest SSE. Model 5 was unique because we included interactions between features; evidently, the interactions are one of the stronger predictors of success.



Another noteworthy finding from our analysis came as a result of plotting a histogram of our outcome variable percentage funded. This graph uses only the samples where the percent funded was less than 300% for visualization purposes (as opposed to the models, which used all samples less than 1000%). The bimodal distribution apparent in the graph reveals that there were more projects that were fully funded (greater than 1) than were almost funded. Perhaps this trend is a psychological effect on the part of backers: backers might be more inclined to contribute to a campaign that is close to being funded, rather than one that does not have any contributions at all.

Conclusion

We were able to create a model to predict the success of a Kickstarter Campaign with some degree of accuracy. Comparing our five models, the model with the lowest SSE had incorporated the most interactions between our variables. This implies that our features independently do not explain the success of a campaign as well as the same features taken in tandem.

Using this model, Kickstarter creators have a basis to evaluate the strength of their campaign by looking at features such as, backers, goal amount, category, and more. If we were to expand on these models, we could include countries and how different categories fare in different countries as an feature, as well as more interactions of features.

References

Mouillé, Mickaël. Kickstarter Projects [Dataset]. Version 7. Accessed Monday, July 23, 2018. <kaggle datasets download -d kemical/kickstarter-projects>

"Kickstarter Stats." *Kickstarter*, 28 July 2018, 7:12 PM, www.kickstarter.com/help/stats?ref=about_subnav.

Appendix

Below you will find our resource appendix, which includes the original features of our dataset, as well as our full commented code.

Original features

Attribute	*Explanation*	*Variable Used*
Index	The index of each campaign.	
Name	The name of the campaign.	
Subcategory	The more specific subcategory of the campaign (eg Narrative Short Film, or Mobile App Design)	
Main Category	The umbrella category of the campaign (eg Tech, Film, or Journalism).	
Currency	The currency in which the campaign is being conducted.	
Deadline	The end date and time of the campaign.	
Goal	The USD goal of each campaign. When a campaign reaches the goal that its creators have set in dollars pledged, the campaign is considered to be a success.	goal
Launched	The start date and time of the campaign.	
Pledged	The amount of money, in USD, that donors have pledged towards the campaign.	pledged
State	Whether the campaign was 'Successful', 'Failed', 'Suspended', or 'Canceled'.	
Backers	The number of donors.	backers
Country	The country from which the campaign was launched.	

Code Appendix

```
%import data, clean up variables

    format long g %gets rid of scientific notation hell yeah!!!!
    load('allUSD_clean.mat');
    allUSD = sortrows(allUSD,'ID','descend'); %using ID to sort in order to
randomize data

    %VARIABLES VECTORS
      betterIndex = [1:size(allUSD,1)]';
      backers = table2array(allUSD(:,11));
      pledged = table2array(allUSD(:,14));
      goal = table2array(allUSD(:,15));
      loggoal = log(goal);
      loggoalsquared = loggoal.^2;

%Calculate duration

      launched = allUSD(:,8);
      launched = table2array(launched);
      deadline = table2array(allUSD(:,6));
      launched = datetime(launched,'InputFormat', 'MM/dd/yyyy HH:mm');
      deadline = datetime(deadline,'InputFormat','MM/dd/yyyy');
      duration = deadline-launched;
      duration = days(duration);

      rateOfBackers = backers./duration;
      amountPerBacker = pledged./backers;
      percentFunded = pledged./goal;

      %Interactions
      duraback = duration.*backers;
      goaback = goal.*backers;
      goadura = goal.*duration;

      %main categories:
            art = allUSD.main_category == 'Art';
            comics = allUSD.main_category == 'Comics';
            crafts = allUSD.main_category == 'Crafts';
            dance = allUSD.main_category == 'Dance';
            design = allUSD.main_category == 'Design';
            fashion = allUSD.main_category == 'Fashion';
            film = allUSD.main_category == 'Film & Video';
            food = allUSD.main_category == 'Food';
            games = allUSD.main_category == 'Games';
            journalism = allUSD.main_category == 'Journalism';
```

```
            music = allUSD.main_category == 'Music';
            photography = allUSD.main_category == 'Photography';
            publishing = allUSD.main_category == 'Publishing';
            tech = allUSD.main_category == 'Technology';
            theater = allUSD.main_category == 'Theater';
            categoriesLogical = [art comics crafts dance design fashion film
food games journalism music photography publishing tech theater];

    %recreate matrix with variables of interest as arrays (instead of
    %tables
      BIGGERBOI = [percentFunded backers pledged loggoal duration rateOfBackers
amountPerBacker categoriesLogical loggoalsquared duraback goaback goadura];
      BIGGERBOI(isnan(BIGGERBOI))=0;%remove outliers with PF>100.00
      BIGBOI = BIGGERBOI(BIGGERBOI(:,1)<10,:);
      BIGBOI(isnan(BIGBOI))=0;
      numSamples = size(BIGBOI,1);
      xVals = ["Percent Funded", "Backers", "Pledged", "Goal", "Duration",
"Rate of Backers", "Amount Per Backer", "Art", "Comics", "Crafts", "Dance",
"Design", "Fashion", "Film & Video", "Food", "Games", "Journalism", "Music",
"Photography", "Publishing", "Technology", "Theater", "Goal Squared",
"DurationXBackers", "GoalXBackers", "GoalXDurations"];

%%

%sort into training set and test set
    allTraining = BIGBOI(1:round(.8*numSamples),:);
      training = allTraining(:,2:26);
      pftrain = allTraining(:,1); %percent funded for training
    allTest = BIGBOI(size(training,1)+1:numSamples,:);
      test = allTest(:,2:26);
      pftest = allTest(:,1); %percent funded for test

%creating a ones vector
    onesVector = ones(size(training,1),1);

%HERE THAR BE MODELS

    %M1: Backers, duration, rateofbackers
      A1 = [onesVector training(:,1) training(:,4) training(:,5)];
      w1 = A1\pftrain;
    %M2: backers, duration, categories (DOESN'T WORK)
      A2noise = [zeros(size(training,1),3)
normrnd(0,0.01,size(training,1),15)];
      A2 = [onesVector training(:,1) training(:,4) training(:,7:21)];
      A2noise = A2noise + A2;
      w2 = A2noise\pftrain;
      %%
    %M3: backers, goal, duration, categories
```

```
    %(DOESN'T WORK)
      A3noise = [zeros(size(training,1),4)
normrnd(0,0.01,size(training,1),15)];
      A3 = [onesVector training(:,1) training(:,3) training(:,4)
training(:,7:21)];
      A3noise = A3noise + A3;
      w3 = A3noise\pftrain;
      %%
    %M4: backers, goal, goalsquared, duration, categories
      A4noise = [zeros(size(training,1),5)
normrnd(0,0.01,size(training,1),15)];
      A4 = [onesVector training(:,1) training(:,3) training(:, 22),
training(:,4) training(:,7:21)];
      A4noise = A4noise + A4;
      w4 = A4noise\pftrain;

      %%
    %M5: backers, goal, goalsquared, goaback,  duration, duraback, goaback,
goadura, categories,
      A5noise = [zeros(size(training,1),8)
normrnd(0,0.02,size(training,1),15)];
      A5 = [onesVector training(:,1) training(:,3) training(:,22) training(:,4)
training(:,23:25) training(:,7:21) ];
      A5noise = A5noise + A5;
      w5 = A5noise\pftrain;

%calculate SSE for training data
    yhat1 = A1*w1;
    yhat2 = A2*w2;
    yhat3 = A3*w3;
    yhat4 = A4*w4;
    yhat5 = A5*w5;
    sse1 = sum((yhat1-pftrain).^2);
    sse2 = sum((yhat2-pftrain).^2);
    sse3 = sum((yhat3-pftrain).^2);
    sse4 = sum((yhat4-pftrain).^2);
    sse5 = sum((yhat5-pftrain).^2);

%%
%run test data through Models
    testOnesVector = ones(size(test,1),1);
  %M1
    A1Test = [testOnesVector test(:,1) test(:,4) test(:,5)];
  %M2
    A2Testnoise = [zeros(size(test,1),3) normrnd(0,0.01,size(test,1),15)];
    A2Test = [testOnesVector test(:,1) test(:,4) test(:,7:21)];
    A2Testnoise = A2Testnoise + A2Test;
  %M3
```

```
    A3Testnoise = [zeros(size(test,1),4) normrnd(0,0.01,size(test,1),15)];
    A3Test = [testOnesVector test(:,1) test(:,3) test(:,4) test(:,7:21)];
    A3Testnoise = A3Testnoise + A3Test;
  %M4
    A4Testnoise = [zeros(size(test,1),5) normrnd(0,0.01,size(test,1),15)];
    A4Test = [testOnesVector test(:,1) test(:,3) test(:, 22), test(:,4)
test(:,7:21)];
    A4Testnoise = A4Testnoise + A4Test;
  %M5
    A5Testnoise = [zeros(size(test,1),8) normrnd(0,0.02,size(test,1),15)];
    A5Test = [testOnesVector test(:,1) test(:,3) test(:,22) test(:,4)
test(:,23:25) test(:,7:21) ];
    A5Testnoise = A5Testnoise + A5Test;

%calculate SSE for test data
    yhat1Test = A1Test*w1;
    yhat2Test = A2Testnoise*w2;
    yhat3Test = A3Testnoise*w3;
    yhat4Test = A4Testnoise*w4;
    yhat5Test = A5Testnoise*w5;
    sse1test = sum((yhat1Test-pftest).^2);
    sse2test = sum((yhat2Test-pftest).^2);
    sse3test = sum((yhat3Test-pftest).^2);
    sse4test = sum((yhat4Test-pftest).^2);
    sse5test = sum((yhat5Test-pftest).^2);

%FUN. WITH. GRAPHS.
% SORTING THE DATA BASED ON CATAGORIES

%%
%Sort data for ART
target = 1;
art_temp = find(BIGBOI(:,8) == target);
art_sort = (BIGBOI(art_temp,1:7));
art_success = size(art_sort(art_sort(:,1)>=1,:),1);
art_fail = size(art_sort(art_sort(:,1)<1,:),1);
clear art_temp

%%
%sort data for comics

comics_temp = find(BIGBOI(:,9) == target);
comics_sort = (BIGBOI(comics_temp,1:7));
comics_success = size(comics_sort(comics_sort(:,1)>=1,:),1);
comics_fail = size(comics_sort(comics_sort(:,1)<1,:),1);
clear comics_temp
%%
%sort Craft data
```

```
craft_temp = find(BIGBOI(:,10) == target);
craft_sort = (BIGBOI(craft_temp,1:7));
craft_success = size(craft_sort(craft_sort(:,1)>=1,:),1);
craft_fail = size(craft_sort(craft_sort(:,1)<1,:),1);
clear craft_temp
%%
%sort dance data
dance_temp = find(BIGBOI(:,11) == target);
dance_sort = (BIGBOI(dance_temp,1:7));
dance_success = size(dance_sort(dance_sort(:,1)>=1,:),1);
dance_fail = size(dance_sort(dance_sort(:,1)<1,:),1);
clear dance_temp

%%
%Sort for design
design_temp = find(BIGBOI(:,12) == target);
design_sort = (BIGBOI(design_temp,1:7));
design_success = size(design_sort(design_sort(:,1)>=1,:),1);
design_fail = size(design_sort(design_sort(:,1)<1,:),1);
clear design_temp

%%
%Sort for fashion
fashion_temp = find(BIGBOI(:,13) == target);
fashion_sort = (BIGBOI(fashion_temp,1:7));
fashion_success = size(fashion_sort(fashion_sort(:,1)>=1,:),1);
fashion_fail = size(fashion_sort(fashion_sort(:,1)<1,:),1);
clear craft_temp

%%
%film
film_temp = find(BIGBOI(:,14) == target);
film_sort = (BIGBOI(film_temp,1:7));
film_success = size(film_sort(film_sort(:,1)>=1,:),1);
film_fail = size(film_sort(film_sort(:,1)<1,:),1);
clear film_temp

%%
% food
food_temp = find(BIGBOI(:,15) == target);
food_sort = (BIGBOI(food_temp,1:7));
food_success = size(food_sort(food_sort(:,1)>=1,:),1);
food_fail = size(food_sort(food_sort(:,1)<1,:),1);
clear food_temp

%%
%games
```

```
games_temp = find(BIGBOI(:,16) == target);
games_sort = (BIGBOI(games_temp,1:7));
games_success = size(games_sort(games_sort(:,1)>=1,:),1);
games_fail = size(games_sort(games_sort(:,1)<1,:),1);
clear games_temp

%%
%journalism

journalism_temp = find(BIGBOI(:,17) == target);
journalism_sort = (BIGBOI(journalism_temp,1:7));
journalism_success = size(journalism_sort(journalism_sort(:,1)>=1,:),1);
journalism_fail = size(journalism_sort(journalism_sort(:,1)<1,:),1);
clear journalism_temp

%%
% music

music_temp = find(BIGBOI(:,18) == target);
music_sort = (BIGBOI(music_temp,1:7));
music_success = size(music_sort(music_sort(:,1)>=1,:),1);
music_fail = size(music_sort(music_sort(:,1)<1,:),1);
clear music_temp

%%
%photography

photo_temp = find(BIGBOI(:,19) == target);
photo_sort = (BIGBOI(photo_temp,1:7));
photo_success = size(photo_sort(photo_sort(:,1)>=1,:),1);
photo_fail = size(photo_sort(photo_sort(:,1)<1,:),1);
clear photo_temp

%%
%publishing
publishing_temp = find(BIGBOI(:,20) == target);
publishing_sort = (BIGBOI(publishing_temp,1:7));
publishing_success = size(publishing_sort(publishing_sort(:,1)>=1,:),1);
publishing_fail = size(publishing_sort(publishing_sort(:,1)<1,:),1);
clear publishing_temp

%%
%tech

tech_temp = find(BIGBOI(:,21) == target);
tech_sort = (BIGBOI(tech_temp,1:7));
tech_success = size(tech_sort(tech_sort(:,1)>=1,:),1);
tech_fail = size(tech_sort(tech_sort(:,1)<1,:),1);
```

```
clear tech_temp

%%
%theater
theater_temp = find(BIGBOI(:,22) == target);
theater_sort = (BIGBOI(theater_temp,1:7));
theater_success = size(theater_sort(theater_sort(:,1)>=1,:),1);
theater_fail = size(theater_sort(theater_sort(:,1)<1,:),1);
clear theater_temp
%%

categoryNames = {'Art'; 'Comics'; 'Crafts'; 'Dance'; 'Design'; 'Fashion'; 'Film
& Video'; 'Food'; 'Games'; 'Journalism'; 'Music'; 'Photography'; 'Publishing';
'Technology'; 'Theater'};

KSblue = round([03 71 82]./256,2);

%bargraph of sses
figure

    %for training data
    subplot(2,1,1)
    ssearray = [sse1 sse2 sse3 sse4 sse5];
    b1 = bar([1 2 3 4 5], ssearray);
    b1.FaceColor = KSblue;
    title("Training Data SSEs")
    xlabel("Model #")

    %for test data
    subplot(2,1,2)
    sseTestarray = [sse1test sse2test sse3test sse4test sse5test];
    b2 = bar([1 2 3 4 5], sseTestarray,'FaceColor',KSblue)
    b2.FaceColor = KSblue;
    title("Test Data SSEs")
    xlabel("Model #")
%%
%bar chart of categories for success vs fail
    successes = [art_success, comics_success, craft_success, dance_success,
design_success, fashion_success, film_success, food_success, games_success,
journalism_success, music_success, photo_success,
publishing_success,tech_success, theater_success];
    failures = [art_fail, comics_fail, craft_fail, dance_fail, design_fail,
fashion_fail, film_fail, food_fail, games_fail, journalism_fail, music_fail,
photo_fail, publishing_fail,tech_fail, theater_fail];
    figure
    successCompare = [successes; failures]';
    b = bar(successCompare)
    names = gca;
```

```
    names.XTickLabel = categoryNames
    b(1).FaceColor = 'g'
    b(2).FaceColor = 'r'
    title("Successes Vs. Failures by Category")
    ylabel("Frequency")
    legend("Success", "Failure")
 %%
%histogram of normalized goal
    figure
    h1 = histogram(loggoal);
    title("Histogram of Normalized Goal Amounts")
    xlabel("log(Goal)")
    h1.FaceColor = KSblue;
  %%
%histogram of percent funded (under 1000%)
    figure
    PFmini = BIGBOI(BIGBOI(:,1)<3);
    h2 = histogram(PFmini);
    title("Histogram of Percent Funded (under 1000%)")
    xlabel("Percent Funded")
    h2.FaceColor = KSblue;
```