Analyzing Kickstarter Data

datasource: https://www.kaggle.com/kemical/kickstarter-projects (https://www.kaggle.com/kemical/kickstarter-projects)

378661 data points, 15 predictors, classification problem with logistic regression

Analyzing the different predictors of successful and failed campaigns. Can we build a model we can use to determine if a kickstarter campaign will be successful?

partners: Luyanda, Kalyani, Ian

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy as sci
import math

% matplotlib inline
```

In [95]:
```
df = pd.read_csv("ks-projects-201801.csv")
```

In [96]:
```
#head of big dataframe
df.head()
```

Out[96]:

	ID	name	category	main_category	currency	deadline	goal	launched	pledged	
0	1000002330	The Songs of Adelaide & Abullah	Poetry	Publishing	GBP	2015-10-09	1000.0	2015-08-11 12:12:28	0.0	
1	1000003930	Greeting From Earth: ZGAC Arts Capsule For ET	Narrative Film	Film & Video	USD	2017-11-01	30000.0	2017-09-02 04:43:57	2421.0	
2	1000004038	Where is Hank?	Narrative Film	Film & Video	USD	2013-02-26	45000.0	2013-01-12 00:20:50	220.0	
3	1000007540	ToshiCapital Rekordz Needs Help to Complete Album	Music	Music	USD	2012-04-16	5000.0	2012-03-17 03:24:11	1.0	
4	1000011046	Community Film Project: The Art of Neighborhoo...	Film & Video	Film & Video	USD	2015-08-29	19500.0	2015-07-04 08:35:03	1283.0	ca

In [97]:
```
len(df)
```

Out[97]: 378661

Create a dataframe with only USA data points and create a datetime object to use later

In [100]:
```
US = df[df["country"] == "US"]
USA = df[df["country"] == "USA"]
us_df = pd.concat([US, USA])
```

In [6]:
```
len(us_df)
```

Out[6]:
```
292627
```

In [7]:
```
us_df.columns
```

Out[7]:
```
Index(['ID', 'name', 'category', 'main_category', 'currency', 'deadline',
       'goal', 'launched', 'pledged', 'state', 'backers', 'country',
       'usd pledged', 'usd_pledged_real', 'usd_goal_real'],
      dtype='object')
```

In [101]:
```
type(us_df["launched"].iloc[0])
```

Out[101]:
```
str
```

In [102]:
```
us_df["launched"] = us_df["launched"].apply(lambda x: x[:-9])
```

In [103]:
```
us_df.head()
```

Out[103]:

	ID	name	category	main_category	currency	deadline	goal	launched	pledged
1	1000003930	Greeting From Earth: ZGAC Arts Capsule For ET	Narrative Film	Film & Video	USD	2017-11-01	30000.0	2017-09-02	2421.0
2	1000004038	Where is Hank?	Narrative Film	Film & Video	USD	2013-02-26	45000.0	2013-01-12	220.0
3	1000007540	ToshiCapital Rekordz Needs Help to Complete Album	Music	Music	USD	2012-04-16	5000.0	2012-03-17	1.0
4	1000011046	Community Film Project: The Art of Neighborhoo...	Film & Video	Film & Video	USD	2015-08-29	19500.0	2015-07-04	1283.0
5	1000014025	Monarch Espresso Bar	Restaurants	Food	USD	2016-04-01	50000.0	2016-02-26	52375.0

In [104]:
```
from datetime import date, time, datetime
us_df["launched"] = list(map(lambda x: datetime.strptime(x,'%Y-%m-%d'), \
                             us_df["launched"]))
us_df["deadline"] = list(map(lambda x: datetime.strptime(x,'%Y-%m-%d'), \
                             us_df["deadline"]))
us_df["time length"] = us_df["deadline"] - us_df["launched"]
```

In [105]:
```
us_df_sort = us_df.drop_duplicates()
```

Add day of the week and months to the data frame as possible predictors

In [106]:
```
us_df["Day of the week"] = us_df["launched"].apply(lambda x: \
                                                   datetime.strftime(x, '%A'))
us_df["launched month"] = list(map(lambda x: x.month, us_df["launched"]))
```

In [107]:
```
us_df.head()
```

Out[107]:

	ID	name	category	main_category	currency	deadline	goal	launched	pledged
1	1000003930	Greeting From Earth: ZGAC Arts Capsule For ET	Narrative Film	Film & Video	USD	2017-11-01	30000.0	2017-09-02	2421.0
2	1000004038	Where is Hank?	Narrative Film	Film & Video	USD	2013-02-26	45000.0	2013-01-12	220.0
3	1000007540	ToshiCapital Rekordz Needs Help to Complete Album	Music	Music	USD	2012-04-16	5000.0	2012-03-17	1.0
4	1000011046	Community Film Project: The Art of Neighborhoo...	Film & Video	Film & Video	USD	2015-08-29	19500.0	2015-07-04	1283.0
5	1000014025	Monarch Espresso Bar	Restaurants	Food	USD	2016-04-01	50000.0	2016-02-26	52375.0

In [18]:
```
us_df["category"].unique()
```

Out[18]:
```
array(['Narrative Film', 'Music', 'Film & Video', 'Restaurants', 'Food',
       'Drinks', 'Product Design', 'Documentary', 'Indie Rock', 'Crafts',
       'Games', 'Design', 'Comic Books', 'Art Books', 'Fashion',
       'Theater', 'Comics', 'Animation', 'Public Art', 'Webseries',
       'Illustration', 'Photography', 'Tabletop Games', 'Pop', 'People',
       'Art', 'Family', 'Food Trucks', 'Fiction', 'Rock', 'Gadgets',
       'Web', 'Jazz', 'Ready-to-wear', 'Festivals', 'Video Games',
       'Shorts', 'Electronic Music', 'Radio & Podcasts', 'Cookbooks',
       'Apparel', 'Metal', 'Comedy', 'Hip-Hop', 'Periodicals', 'Dance',
       'Technology', 'Painting', 'World Music', 'Publishing',
       'Photobooks', 'Hardware', 'Flight', 'Playing Cards', 'Punk',
       'Anthologies', 'Thrillers', "Children's Books", 'Ceramics',
       'Vegan', 'Fabrication Tools', 'Performances', 'Sculpture',
       'Mobile Games', 'Accessories', 'Sound', 'Nonfiction', 'Print',
       'Poetry', 'Classical Music', 'Apps', 'Country & Folk',
       'Mixed Media', 'Journalism', 'Animals', 'Digital Art',
       'Performance Art', 'Software', 'Knitting', 'Graphic Design',
       'Small Batch', 'Installations', 'Young Adult', 'DIY',
       'DIY Electronics', 'Wearables', 'Camera Equipment', 'Jewelry',
       'Farms', 'Fantasy', 'Webcomics', 'Horror', 'Experimental',
       'Science Fiction', 'Puzzles', 'R&B', 'Music Videos',
       'Architecture', 'Drama', 'Spaces', 'Plays', 'Bacon',
       'Community Gardens', 'Faith', 'Fine Art', 'Live Games',
       'Woodworking', 'Places', 'Graphic Novels', '3D Printing',
       'Academic', 'Zines', 'Musical', 'Movie Theaters', 'Workshops',
       'Conceptual Art', 'Footwear', 'Events', 'Video', 'Immersive',
       'Television', 'Audio', 'Action', 'Space Exploration', 'Couture',
       'Makerspaces', "Farmer's Markets", 'Nature', 'Typography', 'Latin',
       'Robots', 'Crochet', 'Letterpress', 'Translations', 'Calendars',
       'Photo', 'Textiles', 'Childrenswear', 'Weaving', 'Candles',
       'Video Art', 'Quilts', 'Glass', 'Pet Fashion', 'Printing',
       'Gaming Hardware', 'Interactive Design', 'Romance', 'Kids',
       'Literary Journals', 'Civic Design', 'Embroidery', 'Blues',
       'Pottery', 'Stationery', 'Taxidermy', 'Chiptune',
       'Literary Spaces', 'Residencies'], dtype=object)
```

Narrowing down the predictors to relevant ones

In [108]:
```
us_reduced = us_df.drop(["ID", "name", "category", "currency", \
                         "deadline", "launched", "usd pledged"], axis = 1)
```

In [109]:
```
us_reduced.head()
```

Out[109]:

	main_category	goal	pledged	state	backers	country	usd_pledged_real	usd_goal_real	time length
1	Film & Video	30000.0	2421.0	failed	15	US	2421.0	30000.0	60 days
2	Film & Video	45000.0	220.0	failed	3	US	220.0	45000.0	45 days
3	Music	5000.0	1.0	failed	1	US	1.0	5000.0	30 days
4	Film & Video	19500.0	1283.0	canceled	14	US	1283.0	19500.0	56 days
5	Food	50000.0	52375.0	successful	224	US	52375.0	50000.0	35 days

We are just testing on "successful" and "failed" projects, not live or else. Thus, only fails nad successes are kept (reduce n by ~30k).

In [110]:
```
us_reduced["state"].unique()
```

Out[110]:
```
array(['failed', 'canceled', 'successful', 'live', 'suspended'],
      dtype=object)
```

In [49]:
```
len(us_reduced)
```

Out[49]:
```
292627
```

In [111]:
```
failed = us_reduced.loc[us_reduced["state"] == "failed"]
success = us_reduced.loc[us_reduced["state"] == "successful"]
us_sf = failed.append(success)
```

In [112]:
```
us_sf
```

Out[112]:

	main_category	goal	pledged	state	backers	country	usd_pledged_real	usd_goal_r
1	Film & Video	30000.0	2421.00	failed	15	US	2421.00	3000(
2	Film & Video	45000.0	220.00	failed	3	US	220.00	4500(
3	Music	5000.0	1.00	failed	1	US	1.00	500(
7	Food	25000.0	453.00	failed	40	US	453.00	2500(
12	Crafts	5000.0	0.00	failed	0	US	0.00	500(
13	Games	200000.0	0.00	failed	0	US	0.00	20000(

In [113]:
```
us_sf = us_sf.drop(["pledged", "usd_pledged_real", "usd_goal_real"], axis = 1)
```

In [114]:
```
us_sf = us_sf.drop(["country"], axis = 1)
```

In [115]:
```
final_dataset = us_sf
```

Changed the "successes" and "fails" to 1 or 0 respectively to use as outputs/results.

In [116]:
```
final_dataset["state"] = final_dataset["state"].apply(lambda x: int(x == "successful"))
```

In [117]:
```
final_dataset.head()
```

Out[117]:

	main_category	goal	state	backers	time length	Day of the week	launched month
1	Film & Video	30000.0	0	15	60 days	Saturday	9
2	Film & Video	45000.0	0	3	45 days	Saturday	1
3	Music	5000.0	0	1	30 days	Saturday	3
7	Food	25000.0	0	40	45 days	Monday	2
12	Crafts	5000.0	0	0	30 days	Tuesday	9

Used Binarizer to one hot encode the different categorical variables important to classification of the kickstarter. (categories, day of the week, month)

In [118]:
```
from sklearn.preprocessing import LabelBinarizer #with the help of Kalyani and Luyanda
```

In [119]:
```
lb_style = LabelBinarizer()
lb_results = lb_style.fit_transform(final_dataset["main_category"])
dummy = pd.DataFrame(lb_results, columns = lb_style.classes_)
dummy.head()
```

Out[119]:

	Art	Comics	Crafts	Dance	Design	Fashion	Film & Video	Food	Games	Journalism	Music	Photography	F
0	0	0	0	0	0	0	1	0	0	0	0	0	
1	0	0	0	0	0	0	1	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	1	0	
3	0	0	0	0	0	0	0	1	0	0	0	0	
4	0	0	1	0	0	0	0	0	0	0	0	0	

In [131]:
```
lb_style = LabelBinarizer()
lb_results = lb_style.fit_transform(final_dataset["Day of the week"])
dummy_date = pd.DataFrame(lb_results, columns = lb_style.classes_)
dummy_date.head()
```

Out[131]:

	Friday	Monday	Saturday	Sunday	Thursday	Tuesday	Wednesday
0	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0
2	0	0	1	0	0	0	0
3	0	1	0	0	0	0	0
4	0	0	0	0	0	1	0

In [132]:
```
lb_style = LabelBinarizer()
lb_results = lb_style.fit_transform(final_dataset["launched month"])
dummy_month = pd.DataFrame(lb_results, columns = lb_style.classes_)
dummy_month.head()
```

Out[132]:

	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	0	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	1	0	0	0

Transfer of the different dummy variables to one big data frame with the other predictors as well, to the final df called dummy

In [123]:
```
dummy["goal"] = list(final_dataset["goal"] )
dummy["state"] = list(final_dataset["state"])
dummy["backers"] = list(final_dataset["backers"])
dummy["time length"] = list(final_dataset["time length"])
dummy["Monday"] = list(dummy_date["Monday"])
dummy["Tuesday"] = list(dummy_date["Tuesday"])
dummy["Wednesday"] = list(dummy_date["Wednesday"])
dummy["Thursday"] = list(dummy_date["Thursday"])
dummy["Friday"] = list(dummy_date["Friday"])
dummy["Saturday"] = list(dummy_date["Saturday"])
dummy["Sunday"] = list(dummy_date["Sunday"])
```

In [136]:
```
dummy["Jan"] = list(dummy_month[1])
dummy["Feb"] = list(dummy_month[2])
dummy["Mar"] = list(dummy_month[3])
dummy["Apr"] = list(dummy_month[4])
dummy["May"] = list(dummy_month[5])
dummy["June"] = list(dummy_month[6])
dummy["July"] = list(dummy_month[7])
dummy["Aug"] = list(dummy_month[8])
dummy["Sept"] = list(dummy_month[9])
dummy["Oct"] = list(dummy_month[10])
dummy["Nov"] = list(dummy_month[11])
dummy["Dec"] = list(dummy_month[12])
```

In [137]:
```
dummy.head()
```

Out[137]:

	Art	Comics	Crafts	Dance	Design	Fashion	Film & Video	Food	Games	Journalism	...	Mar	Apr	May	Ju
0	0	0	0	0	0	0	1	0	0	0	...	0	0	0	
1	0	0	0	0	0	0	1	0	0	0	...	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	1	0	0	
3	0	0	0	0	0	0	0	1	0	0	...	0	0	0	
4	0	0	1	0	0	0	0	0	0	0	...	0	0	0	

5 rows × 38 columns

In [141]:
```
dummy.corr()
```

Out[141]:

	Art	Comics	Crafts	Dance	Design	Fashion	Film & Video	Food	Game
Art	1.000000	-0.052299	-0.044518	-0.031821	-0.080376	-0.070703	-0.136304	-0.079313	-0.08438
Comics	-0.052299	1.000000	-0.027429	-0.019606	-0.049523	-0.043563	-0.083982	-0.048868	-0.05199
Crafts	-0.044518	-0.027429	1.000000	-0.016689	-0.042154	-0.037081	-0.071486	-0.041597	-0.04425
Dance	-0.031821	-0.019606	-0.016689	1.000000	-0.030131	-0.026505	-0.051098	-0.029733	-0.03163
Design	-0.080376	-0.049523	-0.042154	-0.030131	1.000000	-0.066949	-0.129068	-0.075103	-0.07990
Fashion	-0.070703	-0.043563	-0.037081	-0.026505	-0.066949	1.000000	-0.113535	-0.066064	-0.07028
Film & Video	-0.136304	-0.083982	-0.071486	-0.051098	-0.129068	-0.113535	1.000000	-0.127362	-0.13550
Food	-0.079313	-0.048868	-0.041597	-0.029733	-0.075103	-0.066064	-0.127362	1.000000	-0.07885
Games	-0.084386	-0.051993	-0.044257	-0.031635	-0.079906	-0.070289	-0.135506	-0.078850	1.00000
Journalism	-0.031967	-0.019696	-0.016765	-0.011984	-0.030270	-0.026627	-0.051332	-0.029870	-0.03178
Music	-0.124259	-0.076561	-0.065169	-0.046582	-0.117662	-0.103502	-0.199535	-0.116107	-0.12353
Photography	-0.049138	-0.030276	-0.025771	-0.018421	-0.046530	-0.040930	-0.078906	-0.045914	-0.04885
Publishing	-0.102280	-0.063019	-0.053643	-0.038343	-0.096851	-0.085195	-0.164242	-0.095570	-0.10168
Technology	-0.079372	-0.048904	-0.041628	-0.029755	-0.075159	-0.066114	-0.127456	-0.074165	-0.07890
Theater	-0.052415	-0.032295	-0.027490	-0.019649	-0.049632	-0.043659	-0.084168	-0.048976	-0.05210
goal	-0.001991	-0.002834	-0.004265	-0.003288	-0.000044	-0.003926	0.014875	-0.001314	0.00103
state	0.027277	0.062800	-0.044349	0.057461	-0.001492	-0.066299	0.003131	-0.067064	0.02601
backers	-0.021473	0.005103	-0.013284	-0.007907	0.048233	-0.011124	-0.020746	-0.015321	0.08903
time length	-0.036449	-0.003300	-0.035357	-0.011787	0.003675	-0.027105	0.046811	-0.005796	-0.03948
Monday	-0.002524	0.009621	0.006265	-0.004125	0.007968	0.004906	-0.009002	0.003365	-0.00081
Tuesday	-0.008415	-0.001202	-0.000691	-0.004551	0.031653	0.001559	-0.010850	-0.006855	0.01750
Wednesday	-0.000775	-0.003879	-0.000352	0.002144	0.005424	0.003252	-0.000852	0.000220	-0.00537
Thursday	0.005723	-0.010777	-0.004261	0.000481	-0.004305	-0.002339	0.012172	0.003810	-0.01217
Friday	0.003886	-0.002669	-0.000270	0.004093	-0.014980	0.003543	-0.000104	0.007014	-0.00221
Saturday	0.002976	0.004299	-0.001149	0.002567	-0.019975	-0.003231	0.005436	-0.006337	0.00543
Sunday	0.001143	0.009609	0.000475	0.000596	-0.022608	-0.014180	0.009025	-0.003172	-0.00428
Jan	-0.000421	-0.002158	0.004201	-0.001738	-0.002739	0.003140	-0.000753	-0.001810	-0.00192
Feb	0.001555	0.000299	-0.001325	0.001837	-0.007579	-0.002180	0.002626	0.000081	-0.00136
Mar	0.003391	0.003292	-0.000900	0.001272	-0.001909	-0.004510	0.005793	-0.005109	-0.00041
Apr	-0.002610	-0.000880	-0.005780	0.005751	-0.003832	-0.007934	0.011218	-0.001494	-0.00073
May	0.001802	0.001906	-0.010736	0.001666	-0.005156	-0.003518	0.001954	-0.009855	0.00137
June	0.002404	0.001454	-0.007621	-0.001347	0.003097	0.002444	0.009453	-0.013662	-0.00420
July	0.006793	-0.003352	0.008175	-0.002722	-0.007474	-0.000787	-0.007001	0.033538	-0.00673
Aug	0.004699	0.001874	0.009523	-0.004296	-0.002867	0.001704	-0.007309	0.007337	0.00123
Sept	-0.001731	0.000637	0.001707	-0.001423	0.004508	-0.001323	-0.006838	0.002037	0.00671
Oct	-0.006758	0.009273	0.002605	0.001280	0.004190	0.004527	-0.004631	-0.006838	0.01278
Nov	-0.006735	-0.000163	0.002877	-0.002561	0.019354	0.008852	-0.007717	-0.005004	0.00097
Dec	-0.003799	-0.015097	-0.003339	0.002751	0.001386	0.000159	0.003753	-0.001325	-0.00902

38 rows × 38 columns

In [139]:
```
len(list(dummy))
```

Out[139]: 38

We want to change the time length to an int value instead of a "delta days" time object

In [125]:
```
from datetime import *
```

In [126]:
```
dummy["time length" ] = dummy["time length"].dt.days
```

Export the cleaned dataset to a csv for logistic regression and cross validation.

In [140]:
```
dummy.to_csv("final_kickstarter_data.csv")
```