Introduction

In this individual project report, we will discuss how I build two models based on the Kickstarter dataset. Three major sections which Data Pre-Processing, Regression Model Building, and Classification Model Building will be discussed later.

The aim of this individual project is to predict the value of pledged amount (USD) of a project and whether a project will be successful or failure in the future.

Data Pre-Processing

For the project requirement and nature of the raw data, a data pre-processing needs to be addressed before two models will be built. Following steps shows how I modify the Kickstarter dataset:

1. Variable “goal” represents the target amount request for the project, and it shows the amount of money based on local currency. To make it consistent, I changed the all values in this column to USD dollars by conversion rate provided in the column “usd_pledged.”
2. Variable “state” contains five statuses which are Canceled, Failed, Live, Successful, and Suspended. In this project, we only consider the status Failed and Successful; therefore, I dropped observations with the other status.
3. Variable “category” includes 1471 observations in which these observations do not have assigned category labels. These count for around 8% of the total observation. Instead of dropping all these observations, I choose to replace all those missing values to “N/A.”
4. I assume removing outliers is out of scope in terms of this project.
5. Features that are removed are listed and explained in the table below

	Feature Name	Explanation
Removed for Both Models	**project_id**	Project ID and name are irrelavent to both predictors (usd_pledged and state) of models
	name	
	pledged	usd_pledged is calculated by pledge * static_usd_rate, and the amount of goal in my dataset is also converted to US dollars by static_usd_rate. Therefore, all these features are dependent, and we need to remove these two features to avoid collinearity problem.
	static_usd_rate	
	disable_communication	Projects with project's status of failed and successful all have communication. Thus, this column cannot be a predictor because of its unary nature (not a constant variable)
	country	Each country or region has its own currency; therefore, to prevent collinearity, I remained currency rather than country.
	deadline	These variables are redundant because they have been explained in later variables such as deadline_month, created_at_month, etc.
	state_changed_at	
	created_at	
	launched_at	
	staff_pick	We can only know staff pick, backers, and spotlight information after the prediction takes place; therefore, we cannot include variables happens after prediction.
	backers_count	
	spotlight	
	name_len_clean	Cleaned name length and name length are the same thing, so I remained name length rather than cleaned name length
	blurb_len_clean	Cleaned blurb length and blurb length are the same thing, so I remained blurb length rather than cleaned blurb length
	state_changed_at_weekday	We only know information with state after the prediction
	state_changed_at_month	
	state_changed_at_day	
	state_changed_at_yr	
	state_changed_at_hr	
	launch_to_state_change_days	
Removed for Regression Model	**state**	These two predictors are mutually exclusive
Removed for Classification Model	**usd_pledged**	

6. Categorical features that are dummified are state, currency, category, deadline_weekday, created_at_weekday, launched_at_weekday, deadline_month, deadline_yr, created_at_month, created_at_yr, launched_at_month, and launched_at_yr.

Regression Model Building

The first step for building a sound regression model is feature selection. In this case, I used two feature selection methods which are LASSO and Random Forest. All feature selection methods rely on a dataset which has already been cleaned by dropping irrelevant features and dummyfing categorical variables. Following table lists details of each method:

Method	Explanation	Total Number
LASSO	LASSO with alpha = 0.01, positive = True, and random_state = 0	119
Random Forest	RnadomForestRegressor with random_state = 0; SelectFromModel with threshold = 0.001	89
Combination	Selects the overlapped feature coming from LASSO and Random Forest	81

The second step is to use different features to feed different classification models for regression and calculate the mean squared error (MSE) for each combination. I chose three classification models for regression which are Random Forest Regressor (RFR), Support Vector Regressor (SVR), and KNN. I input different features into the base model (model without any hyperparameter; in other words, each model uses its default value) of RFR, SVR, and KNN, then calculated the MSEs at the different random state of splitting dataset. For example, when a random state of splitting dataset was 26, I used predictors selected from LASSO and then calculated the MSE for RFR model. After that, I used same predictors but calculated the MSE for SVR model and KNN model. Following tables show the average MSE of each combination:

RFR

Split dataset @ random state =	LASSO + RF	RF + RF	Comb + RF	Average MSE
26	18954537104	17376784197	18897758140	18409693147
9	21658296775	22130823225	20566949596	21452023199
25	14848698386	14998640579	13776229593	14541189519
12	25235677784	25281168091	25372093969	25296313282
18	17391894686	18986297715	18967587414	18448593272
17	17288331922	15708466256	16839294348	16612030842

2	17134483992	15415644773	16390307978	16313478914
31	13769973540	15196956339	14219466459	14395465446
28	17698587032	19231777902	18638536882	18522967272
22	23362201798	23733712182	23261126991	23452346990
Average MSE	18734268302	18806027126	**18692935137**	**18744410188**

SVR

Split dataset @ random state =	LASSO + SVM	RF + SVM	Comb + SVM	Average MSE
26	18990279924	18989271899	18996380150	18991977324
9	22745802164	22749154015	22755362437	22750106205
25	14512243441	14516755040	14522548161	14517182214
12	25824946383	25823564358	25831809883	25826773542
18	19720094019	19718648060	19726046804	19721596295
17	16581791879	16582042995	16588656523	16584163799
2	16434950990	16434677644	16440402179	16436676938
31	12901595570	12900127604	12907768959	12903164044
28	19656883250	19656933103	19664621792	19659479382
22	24581202133	24579602645	24587086438	24582630405
Average MSE	19194978975	19195077736	19202068333	**19197375015**

KNN

Split dataset @ random state =	LASSO + KNN	RF + KNN	Comb + KNN	Average MSE
26	20725831412	20330331089	20442396997	20499519833
9	24906331287	24162609251	24596948555	24555296364
25	16758202840	16513579380	16501232976	16591005065
12	26901394976	26794183974	26790375160	26828651370
18	20557223334	20622727857	20553996453	20577982548
17	17917143752	17799852290	17712024447	17809673496
2	18440472420	17296458083	17376415194	17704448566
31	15352129219	15427486141	15375404266	15385006542
28	21186568227	20765230852	21029130285	20993643121
22	26199026597	25381485803	26062714043	25881075481
Average MSE	20894432406	20509394472	20644063838	**20682630239**

By comparing MSEs, we can see that RFR model gives us the lowest average MSE which is 18744410188 (highlight in yellow), meanwhile, in RFR method, Combination feature selection method gives us a better MSE which 18692935137 (highlight in light green) Therefore, I choose to use RFR model with predictors generated from Combination method.

Lastly, I played with some hyperparameters (holding n_estimators constant) in RFR model to find which parameter will give me the best result:

Hyperparameter	Range	Best Value	MSE
random_state	1 – 30	random_state = 6	10686515143.375744
max_features	2 - 31	max_features = 28	10668192823.609177
max_depth	2 – 22	max_depth = 10	10571809791.76272
min_samples_split	2 – 11	min_samples_split = 2	10686515143.375744
together			10767363326.146101
Best RFR Model	***RFR (random_state = 6, min_samples_split = 2, n_estimators = 100)***		**10571809791.76272**

Classification Model Building

Classification model building uses the same logic as it of regression model building. To be noticed, I only used Random Forest feature selection method because LASSO only gave me 22 variables which is not enough to a prediction (Random Forest method gives 87 variables at threshold = 0.0035). Regarding the classification model, I chose to run base models of Random Forest Classifier (RFC) and Logistic Regression (LR). RFC gives me a better accuracy score (0.7076073098172546) than LR does (0.6572460688482787). Thus, I chose to play with some hyperparameters (holding n_estimators constant) in RFC model to find the best model:

Hyperparameter	Range	Best Value	Average cv score
random_state	1 – 31	random_state = 17	0.7415371195133315
max_features	2 - 88	max_features = 9	0.7415371195133315
max_depth	2 – 62	max_depth = 44	0.7417284053837523
min_samples_split	2 – 62	min_samples_split = 17	0.7424303626170301
together			0.7424303626170301
Best RFR Model	***RFC (random_state = 17, max_features = 9, max_depth = 44, min_samples_split = 17, n_estimators = 100)***		**Accuracy Score: 0.7396940076498087**