Independent Study – Final Report

Joyce Tian
December 2018

1. Introduction

Machine learning can be used to classify and cluster data sets and identify complex trends and relationships which are not immediately noticeable through other routes of analysis. This is particularly true for data sets with many variables as the relationship may be multivariate and non-linear. In this case, machine learning may provide insights by allowing for holistic analysis. For instance, the field of computer vision which attempts to identify objects within data, Although most major applications of machine learning focus mainly on analysis of purely quantitative data (Bishop, 2006), one can both classify and cluster qualitative data as well based on conceptual understandings (Fisher, 2018).

However, the results of machine learning algorithms must be statistically analyzed to test the validity of the classification/clustering. The most fundamental way of testing the algorithm is to understand the general accuracy – how well an algorithm's prediction formed by learning from a set of known data can be generalized to unknown or unexposed data (Sokolova & Lapalme, 2009). As such, it is clear that statistics and machine learning should be used in tandem to ensure that the insights machine learning gives are not flawed.

Economists have also been harnessing the power of machine learning and statistics to re-evaluate hypotheses which were previously simply assumed to be true. Within this intersection of economics and computer science, I hope to introduce a rigorous analytical approach to the search fund process, wherein an individual entrepreneur, or search fund principal, seeks to acquire and grow a company, with the potential of selling it at a significant profit after 5-10 years of leadership. The search fund model is continually being refined and renewed while the idea only started in the mid-1990s. In 2016, Stanford's Center for Entrepreneurial Studies published a general search fund survey finding that, in comparison to 2009 search funds, recently successful search funds spent increasingly more time

searching for possible companies in various industries and less time personally contacting acquisition targets and performing due diligence research. Furthermore, 27% of all search funds historically fold without making an acquisition, despite reviewing numerous prospects (Kelly et al., 2016). This suggests that if a process is available for quick screening and labeling of potential targets, search fund investors could spend more time making inquisitions or planning acquisition and consequently have a heightened chance to come out the search with a meaningful outcome. Currently, most AI or machine learning models in economical fields were built upon massive quantitative data sets and relatively transparent, reliable information, leveraging regression algorithms that rank companies for investment (Athey, 2018). However, investment decisions should also consider qualitative profile data, such as potential growth, brand image, customer relationship (Stanford, 2017). It is clear that a holistic economical model should evaluate features both quantitative and qualitative, instead of purely quantitative.

One of the key challenges that I faced during the initial phase of the project is information collection and analysis. Because search fund investors primarily target privately-held companies, it is difficult to obtain companies' core information which can factor heavily into how accurately the evaluation can be made. For instance, the search fund program I work with looks for a business with a stable recurring revenue and an EBITDA between $1-3 million. In reality, revenue and capital structure information are often unavailable, nor are they necessarily accurate. Unlike traditional private equity, there is a distinct lack of information regarding privately-held companies, where information is typically gathered from online sources with uncertified quality. Thus, the current screening strategy involves a generalized intuition towards whether a business is enduringly profitable, within the target range of margin, and possesses room for growth. While some characteristics distinctly raise the likelihood of being chosen, others hold ambiguity and need to be evaluated in the context of individual cases. For example, when considering regionality, manufacturing companies tend to stay regional because cost is high for expansion. In contrast, it is much easier for IT service companies to go national or global. Because all software companies can do that, the market is also competitive. Thus, the value of regionality

alone is not sufficient to drive decision making but has to be evaluated in the context of other characteristics.

In this independent study, I set out to explore whether modern machine learning techniques can be implemented in search fund screening process running on data sets containing company's profile information. Input data are cleaned and categorized before ingesting into various machine learning algorithms. The accuracy of predication from these algorithms are compared and analyzed.

2. Data Collection and Formulation

To gain intuition regarding acquisition standards used by search funds, I worked with a search fund principal and main analyst to gather the most important criteria that they apply in company profiling. These observations are summarized in Table 1.

Table 1. Industry and Company-Specific Criteria for Potential Search Fund Acquisitions

Industry-Level Criteria	
Criteria	**Importance**
Recurring Revenue Model	The general revenue model of an industry or industry niche gives an inexperienced CEO greater visibility into the expected revenue pipeline, which facilitates budgeting, long-term planning and valuation. This also allows an inexperienced CEO to take the time to truly understand the business following acquisition rather than having to refresh revenue pipeline immediately.
Healthy and Sustainable Margins	High EBITDA margins (15%+), like growing industries, give inexperienced CEOs cushioning in the event of economic down-turns or other bumps in the road. Also, high margins provide a cushion for increased investment in the business (e.g., hiring additional sales people, upgrading IT systems) if the seller had under-invested.
Stable Cash Flows	Stable cash flows will ensure that the company will be able to service its debt. In part, stability is achieved through a recurring revenue model and predictable cost structure. In part, positive cash flows are achieved through low capex.
Growing Industry	A growing industry provides a tail-wind. In theory, a rising tide raises all boats, a company in that industry will experience growth without the need to make drastic changes to the business, increase market share, develop new service lines,

	etc. This is a more favorable competitive environment, especially for an inexperienced CEO. By contrast, declining industries often deteriorate into a zero-sum game, increasing competitive pressure as companies compete primarily on price. Kessler and Ellis in "Search Funds- Death and the Afterlife" list low/negative growth as "Theme One" among unsuccessful search fund acquisitions.
Fragmented Industry	Fragmented industries (1) are less likely to have a dominant player that distorts the competitive environment (e.g., through lowering prices in order to increase market share), which makes for a favorable competitive environment, (2) provide more targets for acquisitions, (3) valuations will be more reasonable because there will be less strategic buyers, and (4) provide more avenues for growth through product/service or geographic expansion.
Straightforward Operations	An industry with straightforward operations will allow an inexperienced CEO to focus on increasing value immediately, rather than having to spend time researching company and industry dynamics. Kessler and Ellis in "Search Funds - Death and the Afterlife" list complex operations as "Theme Two" among unsuccessful search fund acquisitions.
Low Exogenous Risk	Exogenous Risk (e.g., technology change, regulatory, litigation, environmental, commodity exposure, and seasonality/cyclicality) could adversely impact the entire industry and majorly disrupt the traditional business model. This may result in a growth industry turning negative; barriers to entry could be eliminated; margins could disappear; a major substitute could be introduced. An example would be how peer-to-peer ride sharing has disrupted the taxi industry by leveraging common drivers, a large resource, to overcome specialized taxi driver knowledge.
High Barriers to Entry	If threat of new entrants is high, there will be pressure on (1) market share, (2) price, (3) costs and (4) rate of investments.
Low Intensity of Rivalry	Highly competitive industries tend to lead to price competition which erodes margins and makes sustainable growth more difficult.
Low Customer Power	Powerful customers will capture more value by (1) forcing prices down, (2) demanding better quality/service, (3) playing companies off one another.
Low Supplier Power	Powerful suppliers can capture more value by (1)

	charging higher prices, (2) limiting quality/service or (3) shifting costs away. Ex: Microsoft eroded the margins of computer makers by increasing the price of Windows operating system.
High Switching Costs	High switching costs lock customers in and enable you to incrementally raise prices every year without worrying that the customers will find better alternatives with similar characteristics or at similar price points.
Few/No Substitutes	Too many substitutes can result in a price ceiling (so as to avoid consumers pursuing substitutes) and customers will thus have high bargaining power.
Company-Level Criteria	
Criteria	**Importance**
Appropriate Size	If too large, the entry multiple will be higher and operations are more likely to be complex, among other things. If too small, there is less money to invest in growth, smaller room for error, and it is less likely that the company will have a track record of growth, among other things.
Track Record of Growth and Profitability	Turn-arounds are complicated and are often unsuccessful. A track record of growth and profitability provides a stable base for an inexperienced CEO to grow incrementally.
Healthy and Sustainable Margins	See industry criteria.
Recurring Revenue Model	See industry criteria.
Stable Cash Flows	See industry criteria.
Low/No Customer Concentration	Losing a large customer can have an out-sized impact on a business if concentration is high. This risk is especially acute if the seller has a strong relationship with the key customer. Additionally, a key customer can exert a lot of power, thus taking value for itself.
Realistic Exit Options	Search fund CEOs aim to significantly grow the business, but also will likely aim to sell or transfer company power in 5-10 years of time. Exit options are realistic if there are strategic buyers in the industry or if you experience enough growth to interest PE shops, among other things.
Strong Management Team	A strong management team will make the transition easier for an inexperienced CEO as well as increase likelihood of maintaining important client relationships and institutional knowledge.

In this particular study, a major part of information was extracted from online sources, for example, company's website, LinkedIn, company profile databases such as Owler, Buzzfile and

Crunchbase. In many cases, company's exact trend of financial performance and grow this unknown but to be estimated. Moreover, many company's revenue model is not obvious, partially because smaller companies often utilize a mixed model of offering services and products, making labeling difficult as products (with dependency on orders) tend to be less recurring and service (with dependency on contracts) more recurring. Even within products, there is a difference between making one-time retail purchases versus regular subscription-type purchases (subscription can be viewed as a contract). While theoretically there is a spectrum between totally non-recurring (e.g. purchase goods once but unlike to buy again) and totally recurring revenue (e.g. purchasing utilities), it is difficult to gather quantitative information to support these scenarios relying on publically available data.

Rather, it would be more reasonable to evaluate companies using bucketed values such as "high", "medium" and "low" in qualitative categories. While somewhat arbitrary, this allows for easier ranking within and between categories. Based on these thoughts, I narrowed down seventeen features to be analyzed for screening companies as acquisition targets: recurring revenue, margin level, stable cash flow, industry growth, industry fragmentation, straightforwardness of operations, exogenous risk, barriers to entry, stickiness of offerings, level of supplier power, level of customer power, service to product ratio, level of long-term client relationships, level of customer concentration, industry niche, year founded and employee estimate. The first 11 features are directly taken from the expert opinion surveying, wherein firms are expected to be better when ranked higher in these values. The next three features (service to product ratio, level of long-term client relationships, and level of customer concentration) aim to measure how likely it would be that a firm has recurring revenue. Such insights are generally emphasized in company newsletters and promotions, and thus are more defined given public information. The industry niche is a written description of the general function of the company. The year founded indicates when the company originally started its business. Finally, the employee estimate, per the search fund principal, is a decent way of estimating revenue amount, with most companies within the ideal range usually ranging from 10-200 employees. The employee estimate is bucketed per LinkedIn estimates from 2-10, 11-50, 51-200, and 201-1000. These would all then be mapped to two possible

target classes indicating whether the company was accepted or rejected for further pursuit. Table 2 and 3 show a list of these features, data types and sample values.

Table 2. Input Features for Machine Learning Model

	Feature Name	**Data Type**	**Sample Value**
1	Industry	Categorical	cleaning_services
2	Year_Founded	Numerical	1995
3	Recurring_Revenue (amount in millions)	Numerical	7.4
4	Margins	Categorical	high or medium or low
5	Stable_Cash_Flow	Categorical	high or medium or low
6	Industry_Growth	Categorical	high or medium or low
7	Fragmented_Industry	Categorical	high or medium or low
8	Straightforward_Operations	Categorical	yes or no
9	Exogenous_Risk	Categorical	high or medium or low
10	Barriers_to_Entry	Categorical	high or medium or low
11	Employee_Estimate	Categorical	2-10 or 11-50 or 51-200 or 201-1000
12	Stickiness	Categorical	high or medium or low
13	Supplier_Power	Categorical	high or medium or low
14	Customer_Power	Categorical	high or medium or low
15	Service_vs_Product	Categorical	service or product
16	Longterm_Relationships	Categorical	yes or no
17	Customer_Concentration	Categorical	high or medium or low

Table 3. Target Classes for Machine Learning Model

	Feature Name	**Type**	**Sample Value**
1	Score	Categorical	Accept or Reject

The data set also contains an “URLtrim” column which is the URL link of company website. This column is used in any machine learning models but is reserved in the data set nevertheless as the identifier for individual companies.

3. Data Processing and Machine Learning Model Creation

I used Google Colaboratory as the work environment for this project. Google Colaboratory, or “Google Colab” for short, is a free Jupyter notebook environment that runs in cloud and requires no local setup. All the code in this project was written in Python version 3.

Because Google Colab is fully integrated with GitHub, I used a folder under my GitHub account to store data file as well as the Jupyter notebook. Data file was read directly from the GitHub path into the Python code in Colab. The input data contain nineteen columns: 'Score', 'Industry', 'Year Founded', 'URLtrim', 'Recurring Revenue (million)', 'Margins', 'Stable Cash Flow', 'Industry Growth', 'Fragmented Industry', 'Straightforward Operations', 'Exogenous Risk', 'Barriers to Entry', 'Employee Estimate', 'Stickiness', 'Supplier Power', 'Customer Power', 'Service vs Project/Product', 'Longterm Relationships', and 'Customer Concentration'. The ‘Score’ column is the target classification column with two possible values: Accept and Reject. The 'URLtrim' contains URLs of company websites and is not used for machine learning. The other seventeen columns are attributes. Out of the seventeen attribute columns, fifteen contain categorical data and two contain numerical data. Although the numerical data fields contain data in very different scale, because these columns got dropped during recursive feature elimination step during data pre-processing, I found it is optional to perform scaling on input data, meaning that model accuracy does not seem to be affected by whether or not autoscaling was performed. Figure 1 is a screenshot taken from the Jupyter notebook showing unique values for individual attribute columns.

Figure 1. Unique Values in Individual Columns

```
In [4]: cols = ['Industry','Year Founded','Recurring Revenue', 'Margins', 'Stable Cash Flow',
                'Industry Growth', 'Fragmented Industry', 'Straightforward Operations',
                'Exogenous Risk', 'Barriers to Entry', 'Employee Estimate',
                'Stickiness', 'Supplier Power', 'Customer Power',
                'Service vs Project/Product', 'Longterm Relationships',
                'Customer Concentration']
        for col in dataframe[cols].columns.values:
            print(col, dataframe[col].unique())
```

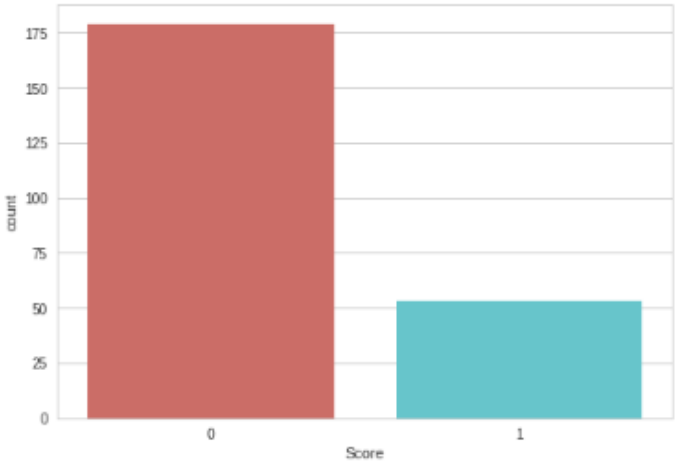
```
Industry ['cleaning_service' 'septic_service' 'distributor' 'manufacturer'
 'IT_service' 'medical_equipment' 'medical_service' 'HR_service'
 'investment' 'news_service' 'marketing_service' 'education'
 'construction' 'inspection_testing_services' 'repairement_service'
 'scan_surveying_services']
Year Founded [1984 1993 1979 2000 2015 2010 1971 1998 1994 1991 1940 2014 1999 2016
 2013 2001 2009 2011 1982 1983 1962 1981 1986 2008 1995 1980 1992 1974
 1939 1946 1990 2002 1959 1960 1881 1949 1950 1970 1987 1975 1873 1968
 1972 2006 1977 1969 2005 1966 1954 1928 1997 1965 2003 1956 1988 2004
 1996 1985 1989 1953 1936 1963 1958 1948 1976 1918]
Recurring Revenue [7.3e+00 8.4e+00 5.6e+00 1.6e-01 5.7e+00 6.0e-01 1.3e+00 2.0e+00 1.9e+00
 6.0e+00 3.8e+00 2.6e-01 9.0e-01 1.0e+00 5.0e-02 4.0e-02 3.0e-01 2.2e-01
 5.0e-01 7.4e+00 4.3e+00 4.4e+00 1.2e+01 3.7e+01 2.2e+01 7.0e+00 1.4e+01
 7.9e+00 2.1e+01 5.0e+00 8.0e+00 1.3e+01 1.6e+01 4.2e+01 1.7e+01 3.0e+01
 2.8e+00 1.0e+01 5.1e+01 2.6e+01 4.0e-01 1.5e+00 4.8e+01 3.9e+00 2.0e-01
 6.4e+00 7.2e+00 4.5e+00 1.5e+01 1.0e-01 9.8e+00 4.6e+00 4.1e+00 7.0e-02
 4.0e+01 1.2e+00 8.3e+00 7.2e+01 4.2e+00 2.6e+00 3.2e+00 2.9e+01 3.0e+00
 2.9e+00 3.8e+01 4.0e+00 2.5e+01 5.0e+01 6.5e+00 1.8e+01 2.8e+01 1.1e+01
 2.0e+01 4.4e+01 2.3e+01 7.0e-01 3.5e+01 2.7e+01 9.0e+00 2.4e+01 5.2e+01
 5.3e+01 1.9e+01]
Margins ['low' 'medium' 'high']
Stable Cash Flow ['high' 'medium' 'low']
Industry Growth ['high' 'medium' 'low']
Fragmented Industry ['high' 'medium' 'low']
Straightforward Operations ['yes' 'no']
Exogenous Risk ['low' 'medium' 'high']
Barriers to Entry ['low' 'medium' 'high']
Employee Estimate ['51-200' '2-10' '11-50' '201-1000']
Stickiness ['low' 'medium' 'high']
Supplier Power ['low' 'medium' 'high']
Customer Power ['high' 'medium']
Service vs Project/Product ['service' 'product']
Longterm Relationships ['low' 'medium' 'yes' 'no']
Customer Concentration ['low' 'medium' 'high']
```

Like data sets collected in real life scenarios, during data exploration, I observed a number of issues with the data set I use for this project. First, the size of the data set is small. I have less than 232 records to work with. This is because reviewing information and populating attribute columns proved very time-consuming and I ran out of time to get more data populated for this study. This limitation can be mitigated in the future if other students would like to continue and leverage this work. With more data points collected and ingested into the study, machine learning models can gain insight from a bigger pool of information and make more accurate inferences. Second, the data set is imbalanced between the two target classes. For example, out of the 232 hundred records, 179 records belong to the Rejected class while only 53 in the Accept class. The ratio between reject and accept class is 77:22.

Figure 2. Target Class Ratio Shows Imbalanced Data Set (0 =Reject, 1 = Accept)

```
In [7]: sns.countplot(x='Score',data=dataframe, palette='hls')
        plt.show()
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:1428: FutureWarning: remove_na is deprecated
and is a private function. Do not use.
  stat_data = remove_na(group_data)
```




```
In [8]: count_reject = len(dataframe[dataframe['Score']==0])
        count_accept = len(dataframe[dataframe['Score']==1])
        pct_of_reject = count_reject/(count_reject + count_accept)
        print("percentage of reject is", pct_of_reject*100)
        pct_of_accept = count_accept/(count_reject+count_accept)
        print("percentage of accept", pct_of_accept*100)
```

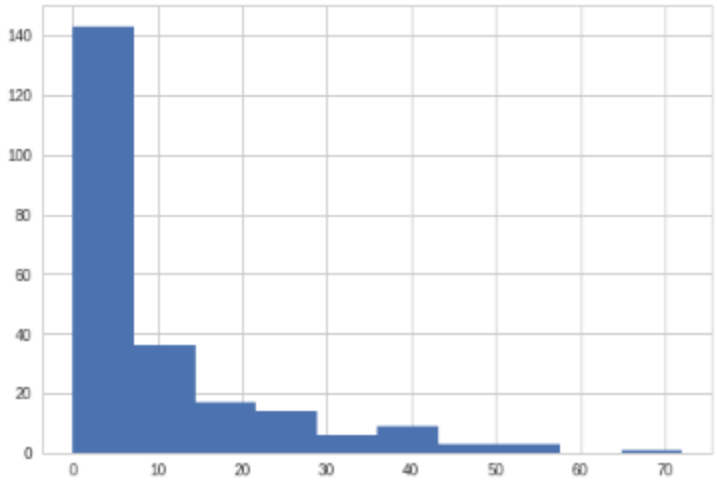
```
percentage of reject is 77.15517241379311
percentage of accept 22.844827586206897
```

Furthermore, as shown in the histogram in Figure 3, numeric attributes such as the Recurring Revenue is heavily skewed to one side. Most companies in the data set have small revenues between 0.5 to 15 million, but there are some outliers with relatively higher revenue. However, since we did consider and label all the companies during search and evaluation process, I chose not to remove the outliers. In future, it may make sense to bucketize the revenue using arbitrary categories instead of using real values as did in this study.

Figure 3. Histogram of Recurring Revenue with Skewed Distribution

```
plt.hist(dataframe['Recurring Revenue'])
```

```
Out[10]: (array([143.,  36.,  17.,  14.,   6.,   9.,   3.,   3.,   0.,   1.]),
          array([4.0000e-02, 7.2360e+00, 1.4432e+01, 2.1628e+01, 2.8824e+01,
                 3.6020e+01, 4.3216e+01, 5.0412e+01, 5.7608e+01, 6.4804e+01,
                 7.2000e+01]),
          <a list of 10 Patch objects>)
```



On the other hand, the imbalance between target classes (Reject vs Accept) is something that should be addressed. To correct the problem, I chose an imbalance enhancing technique called "SMOTE" (Synthetic Minority Over-sampling Technique). Currently, data imbalance problems are mostly handled by using one or two of the three methods:

1. Over-sample the minority class
2. Under-sample the majority class
3. Synthesize new minority class

SMOTE belongs to the third method in the above list, as it is an over-sampling approach in which the minority class is over-sampled by creating "synthetic" examples rather than by over-sampling with replacement (Chawla, 2002). This approach has been shown successful in achieving better classifier performance (in ROC space) than only under-sampling the majority class. While in most cases SMOTE seems beneficial with low-dimensional data, it has been shown in a high dimensional setting, SMOTE strongly biases the classification towards the minority class and should be used with variable selection to mitigate the risk especially when using with K-NN classifier (Blagus, 2013). The data set in this study has only 17 variable columns which is rather small and therefore, a suitable use case for SMOTE. After

applying the SMOTE technique, proportion for reject and accept class in oversampled data pool is 0.5 each. Both Reject and Accept classes now contain 128 samples. The target class is evenly balanced by creating synthetic samples for the Accept class.

Figure 4. Correction of Class Imbalance Problem using SMOTE

```
In [0]: from imblearn.over_sampling import SMOTE

os = SMOTE(random_state=0)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
columns = X_train.columns

os_data_X,os_data_y=os.fit_sample(X_train, y_train)
os_data_X = pd.DataFrame(data=os_data_X,columns=columns )
os_data_y= pd.DataFrame(data=os_data_y,columns=['Score'])
# we can Check the numbers of our data
print("length of oversampled data is ",len(os_data_X))
print("Number of reject in oversampled data",len(os_data_y[os_data_y['Score']==0]))
print("Number of accept data",len(os_data_y[os_data_y['Score']==1]))
print("Proportion of reject data in oversampled data is ",len(os_data_y[os_data_y['Score']==0])/len(os_dat
a_X))
print("Proportion of accept data in oversampled data is ",len(os_data_y[os_data_y['Score']==1])/len(os_dat
a_X))
```

```
length of oversampled data is  256
Number of reject in oversampled data 128
Number of accept data 128
Proportion of reject data in oversampled data is  0.5
Proportion of accept data in oversampled data is  0.5
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/utils/validation.py:761: DataConversionWarning: A column-vec
tor y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example u
sing ravel().
  y = column_or_1d(y, warn=True)
```

Machine learning models require input data in numeric format. Before submitting the data set for ingestion, I converted categorical variables into dummy variables using *pandas.get_dummies* method. The method pivots categorical values into individual columns and populate the columns with 0 or 1 or other numbers as indicators of the category the data point belongs. After the conversion, the data set has sixty columns (a big increase from the original nineteen columns) since we have a lot of categorical data.

Figure 5. Converting Categorical Variables to Dummy Variables with Numeric Indicators

Convert categorical variables to numerical dummy variables using pandas.get_dummies function. {sklearn ref: https://pandas.pydata.org/pandas-docs/stable/generated/pandas.get_dummies.html}

In [0]:
```
cat_vars=['Industry','Margins', 'Stable Cash Flow',
       'Industry Growth', 'Fragmented Industry', 'Straightforward Operations',
       'Exogenous Risk', 'Barriers to Entry', 'Employee Estimate',
       'Stickiness', 'Supplier Power', 'Customer Power',
       'Service vs Project/Product', 'Longterm Relationships',
       'Customer Concentration']
for var in cat_vars:
    cat_list='var'+'_'+var
    cat_list = pd.get_dummies(dataframe[var], prefix=var)
    dataframe1=dataframe.join(cat_list)
    dataframe=dataframe1

cat_vars=['Industry','Margins', 'Stable Cash Flow',
       'Industry Growth', 'Fragmented Industry', 'Straightforward Operations',
       'Exogenous Risk', 'Barriers to Entry', 'Employee Estimate',
       'Stickiness', 'Supplier Power', 'Customer Power',
       'Service vs Project/Product', 'Longterm Relationships',
       'Customer Concentration', 'URLtrim']
dataframe_vars=dataframe.columns.values.tolist()
to_keep=[i for i in dataframe_vars if i not in cat_vars]
dataframe_final=dataframe[to_keep]
dataframe_final.columns.values
print(dataframe_final.shape)
```

```
(232, 60)
```

Before ingesting the data for model creation, feature ranking and elimination was performed using *sklearn.feature_selection.*RFE package. RFE stands for "Recursive Feature Elimination". It uses an external estimator algorithm, for example, Linear Regression or SVM, to assign weights to features (e.g. the coefficients of a linear model). The goal of the recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through a coef_ attribute or through a feature_importances_ attribute. Then, the least important features are pruned from current set of features. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached (scikit-learn documentation). RFE has been a popular technique widely adopted among researchers and engineers to improve model accuracy and efficiency (Mishra,2015, Yan, 2015). For this study, I used both "logistic regression" and "SVR" (Epsilon-Support Vector Regression) as estimator to select top forty most important features from the data set.

Figure 6. Use Recursive Feature Elimination (RFE) to Rank and Select Features Using Different Classification Estimators

```
In [0]: from sklearn.feature_selection import RFE

logreg = LogisticRegression()
rfe = RFE(logreg, 40) #reduce attributes to 40 dimensions
rfe = rfe.fit(os_data_X.astype(float), os_data_y.values.ravel().astype(int))
print(rfe.support_)
print(rfe.ranking_)

'''
from sklearn.svm import SVR
estimator = SVR(kernel="linear")
rfe = RFE(estimator, 50, step=1)
rfe = rfe.fit(os_data_X.astype(float), os_data_y.values.ravel().astype(int))
print(rfe.support_)
print(rfe.ranking_)
'''
```

```
[False False  True  True  True False  True False  True  True  True  True
  True False  True  True  True  True False  True  True  True  True  True
 False False False False False False  True  True False  True  True False
 False False  True  True  True  True False  True  True False  True  True
  True  True  True  True  True  True  True  True  True False  True]
[20 13  1  1  1  5  1  3  1  1  1  1  1  8  1  1  1  1  6  1  1  1  1  1
  4 15 14 10  7 19  1  1 17  1  1 18  9 16  1  1  1  1 12  1  1  2  1  1
  1  1  1  1  1  1  1  1  1 11  1]
```

The impact of feature elimination using different estimators were measured by coupling with various machine learning classifiers for prediction accuracy on test data after being trained using the same data. The result is shown in Table 4.

Table 4. Classification Accuracy Resulted from Features Selected by Different RFE Estimators

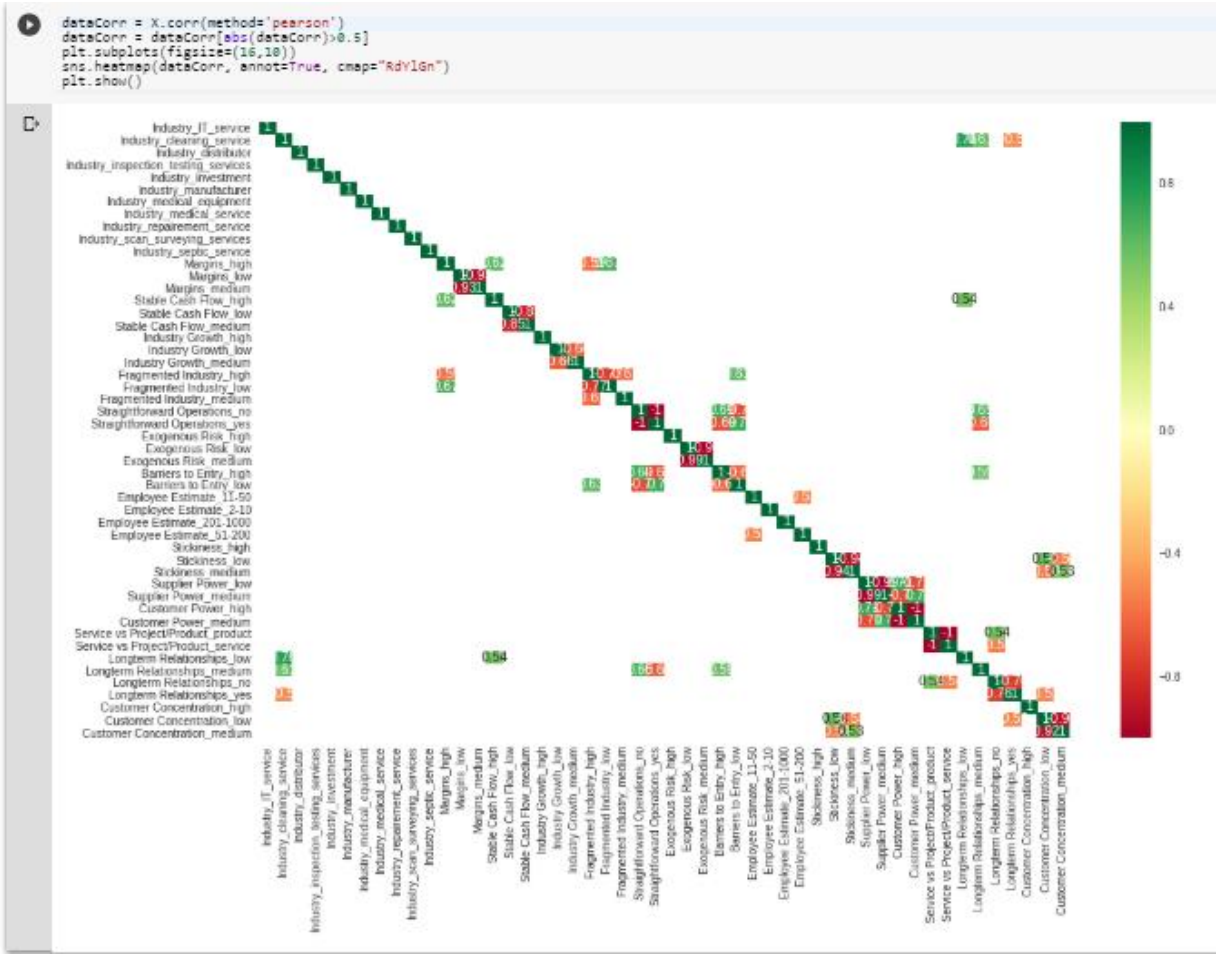
RFE Estimator	Downstream Classifier	Accuracy on Test Data
Logistic Regression	Logistic Regression	0.87
	Decision Tree	0.90
	K-NN	0.88 (k=5)
	Linear Discriminant Analysis	0.94
	Gaussian Naive Bayes	0.88
	Support Vector Machine	0.79
	Random Forest	0.88
	Gradient Boosting Classifier	0.91
	MLP Neural Net	0.91
SVR (support vector regression)	Logistic Regression	0.86
	Decision Tree	0.87
	K-NN	0.90 (k=10)
	Linear Discriminant Analysis	0.95
	Gaussian Naive Bayes	0.84
	Support Vector Machine	0.81
	Random Forest	0.87
	Gradient Boosting Classifier	0.88
	MLP Neural Net	0.88

The observation in Table 4 indicates that features selected by Logistic Regression and SVR as RFE estimators produced similar accuracy when Logistic Regression was used as classifier. The accuracy associated with Support Vector Machine (SVM) was higher when SVR estimator was used (0.81 vs. 0.79), so was K-NN (0.88 vs 0.9). Decision Tree, Random Forest, Gradient Boosting and MLP performed slightly better with Logistic Regression estimator.

As mentioned previously, one problem I had in this study is the small size of data set. The other problem I had after converting categorical variables to dummy variables is the strong correlations (1 or close to 1) among a handful of variables. Changing the initiating parameters of Logistic Regression instance by setting resolver to 'liblinear' and penalty to 'l2'(or ‘l1’) did seem to impact accuracy much. To mitigate the impact of these problems, I employed a dimension expansion technique that adds synthetic attributes into the data set (Dixit, 2014). Pairs of attributes with high correlation score were identified and used to create the new synthetic features. The new data set obtained by the above method can then subjected to Principal Component Analysis (PCA) for feature extraction. The combination of dimension expansion and PCA methodology generates a new data set with extracted features and reduces dimensionality to a desired number of features.

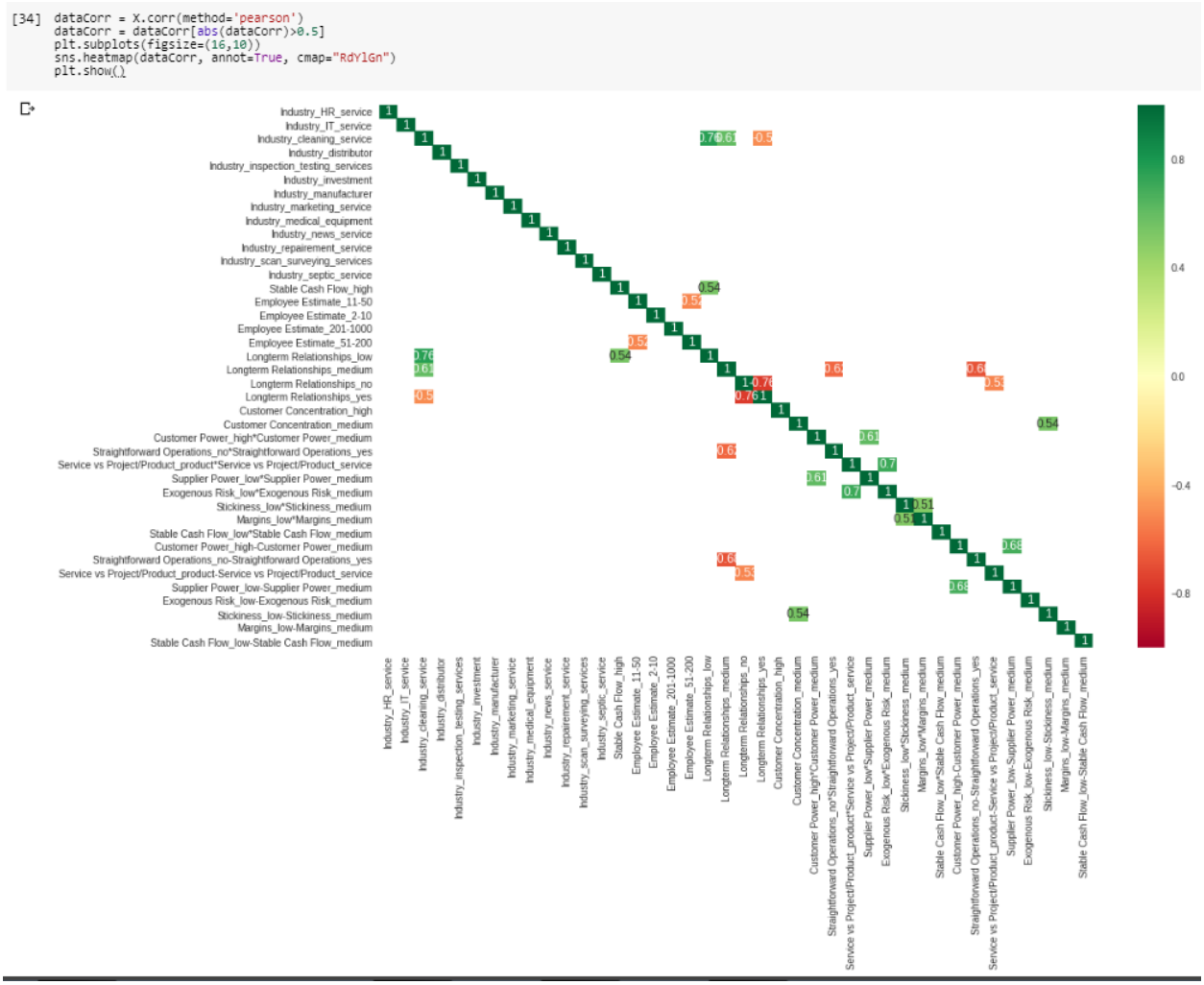
In this study, correlations were calculated on the 40 variables selected in the RFE process. Figure 7 shows a heatmap visualization of the pairwise correlation for variable pairs with correlation score higher than 0.5.

Figure 7. Heatmap Visualization of Pair-wise Correlation for Pairs with Scores > 0.5



A user-defined function was created to rank attribute pairs based on correlation scores and identify the pairs having top scores. Synthetic features were created for these pairs using both multiplication and minus operators. The original variables were dropped from the data set while the synthetic features were added. Figure 8 shows that after adding the synthetic features and dropping out the original ones, there are fewer pairs of variables with correlation score greater than 0.5, thus demonstrating how synthetic features can be used to make models less redundant and more robust.

Figure 8. Heatmap Visualization of Pair-wise Correlation for Pairs with Scores > 0.5



A machine learning classifier, in this case, a Logistic Regression model (a.k.a. logit model), was created using the new data set. Figure 9 and 10 show the comparison of accuracy, confusion matrix, classification report and ROC between Logistic Regression models before and after dimension expansion. The accuracy of prediction increased from 0.81 to 0.87 after adding the synthetic attributes. The later model also made more correct predictions 67 vs 62, and area under ROC increased from 0.8 to 0.88. Interestingly, subjecting the dimensionally expanded data set to Principal Component Analysis (PCA) decreased accuracy to 0.83 and slight drop in area of ROC as well. The observation actually makes sense since my data set was small and only contains 17 features. Feature extraction procedures such as PCA can

cause further reduce of information from the small data set and thus caused the drop of prediction accuracy. For studies that use big data set with a large number of feature dimensions, the application PCA can be helpful in extracting important information and also make model execution more efficient. Nevertheless, my observation in this study is consistent with other reports that dimension expansion can be used to augment model accuracy in small data set classification and in combination with PCA for dimensionality reduction if there is a need for it (Ruparel, 2013).

Figure 9. Accuracy, Confusion Matrix, Classification Report, and ROC Before Dimension Expansion

```
[ ]  y_pred = logreg.predict(X_test.astype(float))
     print('Accuracy of logistic regression classifier on test set: {:.2f}'.format(logreg.score(X_test, y_test)))

     Accuracy of logistic regression classifier on test set: 0.81
```

```
[ ]  from sklearn.metrics import confusion_matrix
     confusion_matrix = confusion_matrix(y_test, y_pred)
     print(confusion_matrix)

     [[28  9]
      [ 6 34]]
```

```
[ ]  from sklearn.metrics import classification_report
     print(classification_report(y_test, y_pred))

                   precision    recall  f1-score   support

                0       0.82      0.76      0.79        37
                1       0.79      0.85      0.82        40

        micro avg       0.81      0.81      0.81        77
        macro avg       0.81      0.80      0.80        77
     weighted avg       0.81      0.81      0.80        77
```

```
[ ]  from sklearn.metrics import roc_auc_score
     from sklearn.metrics import roc_curve
     logit_roc_auc = roc_auc_score(y_test, logreg.predict(X_test))
     fpr, tpr, thresholds = roc_curve(y_test, logreg.predict_proba(X_test)[:,1])
     plt.figure()
     plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
     plt.plot([0, 1], [0, 1],'r--')
     plt.xlim([0.0, 1.0])
     plt.ylim([0.0, 1.05])
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('Receiver operating characteristic')
     plt.legend(loc="lower right")
     plt.savefig('Log_ROC')
     plt.show()
```

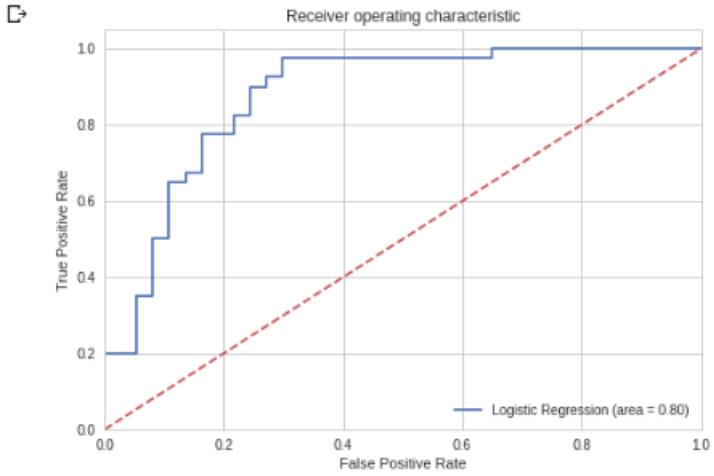


Figure 10. Accuracy, Confusion Matrix, Classification Report and ROC After Dimension Expansion

```
Accuracy of logistic regression classifier on test set: 0.87
[[37  0]
 [10 30]]
              precision    recall  f1-score   support

           0       0.79      1.00      0.88        37
           1       1.00      0.75      0.86        40

   micro avg       0.87      0.87      0.87        77
   macro avg       0.89      0.88      0.87        77
weighted avg       0.90      0.87      0.87        77
```

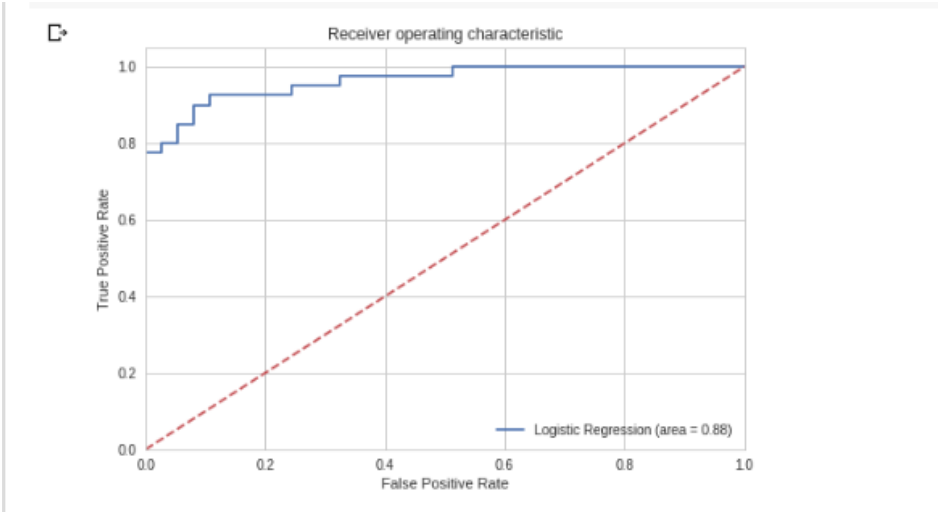
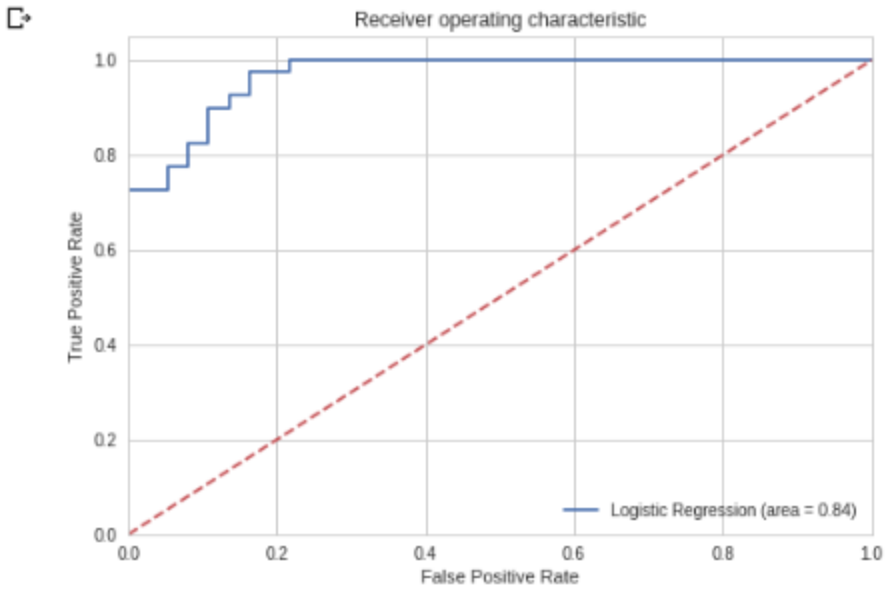


Figure 11. Accuracy, Confusion Matrix, Classification Report and ROC After Dimension Expansion and Feature Extraction using PCA

```
original shape:    (256, 40)
transformed shape: (256, 30)
Accuracy of logistic regression classifier on test set: 0.83
```



4. Discussion and Conclusion

There are two primary challenges I found during this study. The first challenge was with regards to the small data size. Although as a team, my colleagues and I collectively labeled close to 2000 companies for accept or reject, I was only able to populate a small portion of those labeled companies with features I need as input for machine learning. Although those features reflect the thought process we used during evaluation, it is not required for people to pen down those values and therefore I had to repeat the process in order to populate the data set. When the number of representative training samples is relatively small, especially when the situation is coupled with large dimensional count of features and thus of classifier parameters to be estimated, the well-known problem of the curse of dimensionality (i.e., the Hughes phenomenon) occurs (Hughes, 1968; Chi, 2008). This results in the risk of overfitting of the training data and can lead to poor generalization capabilities of the classifier. Since the goal is to label companies with target class of either accept or reject, supervised classification is the suitable algorithm. One approach to overcome the small data set issue is adding synthetic data to the system. Li and his colleagues proposed a mega-trend-diffusion (MTD) technique to estimate the domain range of a small data set, create randomly produced artificial samples within the domain, and ingest the artificial samples

as part of the training data to improve classification accuracy (Li, 2007). The approach is particularly suitable for algorithms sensitive to noisy data or outliers, for example, the Support Vector Machine (SVM), as those algorithms assume all data points bearing the same importance. Fuzzy Membership Computation Method can be used to mitigate the impact from noisy data by assigning a fuzzy membership value as a weight to each training data point and uses this weight to control the importance of those data points (Le, 2010). In addition to data expansion techniques like MTD, small data set problem can also be handled using dimension expansion techniques which generate new attributes into the data set. There are two main methods included in the attribute expansion approach: one is building up the class-possibility as new attributes; the other is constructing other new attributes (i.e. synthetic attributes) using the original ones which have high correlation (Li, 1999). The second method of the dimension expansion technique was applied in this study, and synthetic features were added into the data set while original features with high correlation were dropped out. In addition, a Synthetic Minority Over-sampling Technique (SMOTE) was applied to correct the class imbalance issue of the data set. Similar to MTD, SMOTE also generates synthetic data and adding them to training data set. The difference is MTD (and other data expansion techniques) generates new training data by operating in "data space" by doing operations such as rotation and skew on real data points. SMOTE, on the other hand, operates in "feature space" of the minority class. The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any of the k minority class nearest neighbors (Chawla, 2002). Both dimension expansion and SMOTE techniques increased data set size, and resulted in enhanced accuracy of prediction. Nine classification algorithms were tested side-by-side in this study using the same split of training and test data: Logistic Regression, Decision Tree, K-NN, Linear Discriminant Analysis, Gaussian Naive Bayes, Support Vector Machine, Random Forest, Gradient Boosting Classifier, and MLP Neural Net. Among these algorithms, K-NN, Random Forest, and Gradient Boosting produced the best performance in overall accuracy. Support Vector Machine (SVM) didn't seem to work particularly well with this data set probably due to the noisy

data points (outliers) that I included in the data set. On average, even with the limited data size, the accuracy of prediction was in the 75% - 90% range indicating the model worked reasonably well.

The second challenge I experienced regarded data collection. Because search fund targets privately-held companies, information was collected from various online sources which contained only incomplete information. Besides, there is no telling how updated the information was or the accuracy of the information. I also had to make a lot of estimations for revenue, margin, long customer relationship, and other attributes when populating the data set. To make a machine learning model truly useful, a comprehensive and reliable source of information is critical. Commercial subscription-based data sources such as Mergent, Hoover, Dun & Bradstreet can be used to mitigate this problem. These commercial sources were unavailable in this project, but should be used if real implementation is considered. Despite the source data and information issue, the idea of creating a predictive model to guide search fund process is valid.

Due to time constraints, I did not get to test the data set using more sophisticated and intuitive algorithms such the deep learning neural networks. Deep learning usually require a much larger data size and the small data set used in this study would likely not be a suitable use case. But as to make a note for the future, if more data is prepared for this study, it would be interesting to subject the data set to more multi-level analysis and more rigorous frameworks so long as the feature-to-sample ratio is smaller.

Current Bibliography

Athey, S. (2017). The impact of machine learning on economics. Economics of Artificial Intelligence. University of Chicago Press.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. New York: Springer.

Blagus, R. and Lara Lusa, L. (2013) SMOTE for high-dimensional class-imbalanced data. BMC Bioinformatics, 14-106.

Chawla, Nitesh & Bowyer, Kevin & O. Hall, Lawrence & Philip Kegelmeyer, W. (2002). SMOTE: Synthetic Minority Over-sampling Technique. J. Artif. Intell. Res. (JAIR). 16. 321-357.

Chen, S., Goo, Y.-J. J., & Shen, Z.-D. (2014). A Hybrid Approach of Stepwise Regression, Logistic Regression, Support Vector Machine, and Decision Tree for Forecasting Fraudulent Financial Statements. The Scientific World Journal, 968712.

Chi, M., Feng, R., Bruzzone, L. (2008). Classification of hyperspectral remote-sensing data with primal SVM for small-sized training data set problem. Advances in Space Research, 41(11), 1793-1799.

Creswell, J. W., & Creswell, J. D. (2017). Research design: Qualitative, quantitative, and mixed methods approaches. Sage publications.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. Neural computation, 10(7), 1895-1923.

Dixit, S. Gwal, N. (2014) An Implementation of Data Pre-Processing for Small Data set. International Journal of Computer Applications, 103(6).

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering.

Hu, Y., Guo, D., Fan, Z., Dong, C., Huang, Q., Xie, S., & Xie, Q. (2015). An improved algorithm for imbalanced data and small sample size classification. Journal of Data Analysis and Information Processing, 3(03), 27.

King, G., Honaker, J., Joseph, A., & Scheve, K. (2001). Analyzing incomplete political science data: An alternative algorithm for multiple imputation. American political science review, 95(1), 49-69.

Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering, 160, 3-24.

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. The Journal of Machine Learning Research, 18(1), 559-563.

Le, T., Tran, D., Ma, WL, Sharma, D (2010). A new fuzzy membership computation method for fuzzy support vector machines, International Conference on Communications and Electronics 2010, 153-157.

Li, D. C., Wu, C. S. Tung-I Tsai, T. I., Lin, Y. S. (2008). Using Mega-Trend-Diffusion and Artificial Samples in Small Data Set Learning. Computers & Operations Research, 34(4), 966-982

Li, D., Liu, C. (2010). Extending Attribute Information for Small Data Set Classification. IEEE Transactions on Knowledge & Data Engineering, 24(3), 452-464.

Ruparel, N. H., Shahane, N. M., & Bhamare, D. P. (2013, May). Learning from small data set to build classification model: A survey. In Proc. IJCA Int. Conf. Recent Trends Eng. Technol.(ICRTET), 23-26)).

Mishra, S. Mishra, D. (2015) SVM-BT-RFE: An improved gene selection framework using Bayesian T-test embedded in support vector machine (recursive feature elimination) algorithm. Karbala International Journal of Modern Science, 1(2), 86-96.

Raudys, S. J., & Jain, A. K. (1991). Small sample size effects in statistical pattern recognition: Recommendations for practitioners. IEEE Transactions on Pattern Analysis & Machine Intelligence, (3), 252-264.

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information Processing & Management, 45(4), 427-437.

Stanford Graduate School of Business (2017). A Primer on Search Funds.

Yan, K., Zhang, D. (2015) Feature selection and analysis on correlated gas sensor data with recursive feature elimination. Sensors and Actuators B: Chemical, 212, 353-363.