IE 598 Group Project

Team Members:

Bingjie Han	NetID: bingjie5
Wenning Li	NetID: wenning5
Qi Guo	NetID: qiguo3
Zhanjie Shen	NetID: zhanjie2
Jingyi Wang	NetID: jingyiw9
Siyang Zhang	NetID: siyangz2
Zefang Zhao	NetID: zefangz2

Part I

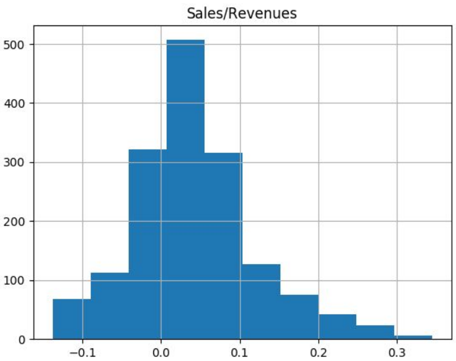
Chapter 1 Introduction and Exploratory Data Analysis

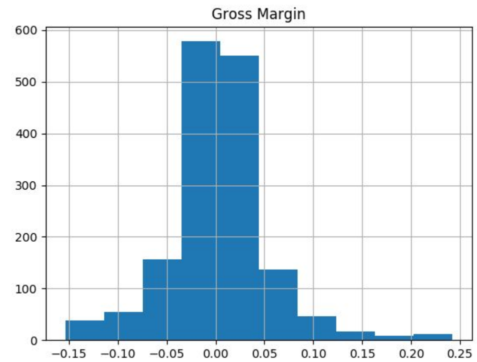
This part of the project aims at classifying the credit ratings of a set of firms based on 26 financial and accounting metrics. Our first mission is to predict whether a firm is considered investment grade. And the second task is to classify a firm's credit rating into one of the 16 categories.

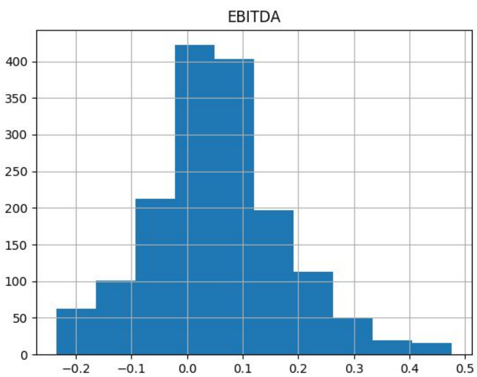
Our team start with exploratory data analysis. A quick examination of the data showed us that many features have extreme outliers. For example, the maximum value of the Total Liquidity feature is 280.14, while the mean and standard deviation are -0.856 and 22.93, respectively. This means that standardization is crucial for our classification accuracy.

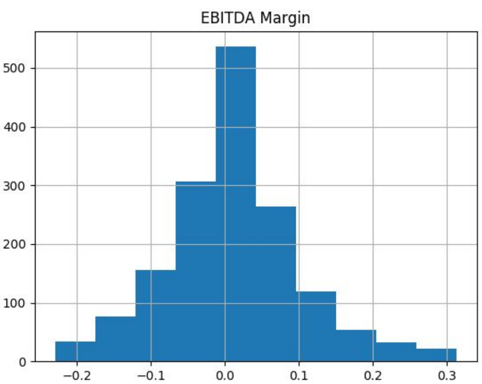
The statistical properties of each feature are summarized in the table below. We also plotted histograms after outliers were removed.

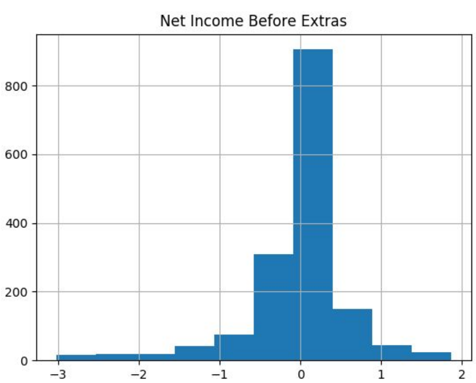
Features	Mean	Median	Standard Deviation	Skewness	Kurtosis
Sales/Revenues	0.050377682272353	0.0339996745	0.161909637899435	4.43199511546541	49.0482522366757
Gross Margin	0.026006580678235	0.0034025015	0.273768267387604	7.81607892843089	77.3217427017191
EBITDA	0.068718292291176	0.0494817075	0.237364826650835	4.8577699299205	53.4988777068635
EBITDA Margin	0.021074299612941	0.011133941	0.189025280142482	7.59020134311942	142.320134317037
Net Income Before Extras	0.123025518688823	0.056626844	14.4756891304199	16.9652919589302	798.86550064084
Total Debt	0.822405347696471	0.0058861025	13.3170751355259	19.8852096422223	400.551439885077
Net Debt	-0.41980965456	-0.003060007	28.3857021843411	10.4938182344155	599.441705063444
LT Debt	1.25516837635706	-0.002078429	16.2244529813005	15.4867537958203	244.900273564108
ST Debt	3.14279679850235	0.043091966	51.9865500925135	36.0230826253271	1389.41615926553
Cash	0.466620196342941	0.0758196255	1.85949371703699	8.36310663551683	113.565094118542
Free Cash Flow	-0.312325155065883	-0.0584751805	8.89513647584915	-11.7575954970176	364.374449171013
Total Debt/EBITDA	0.731196943574118	-0.0123021215	12.2804933329858	19.7989193867086	397.167989854868
Net Debt/EBITDA	-0.819863349865295	-0.034452293	22.0025504266437	-12.823795414834	341.324068084868
Total MV	0.092042815710588	0.066836331	0.385111132124002	2.5162305477545	16.7948071753651
Total Debt/MV	1.27020152859823	-0.0184644745	22.7970537683406	23.0241754291156	577.714229790535
Net Debt/MV	-0.398624264799412	-0.032054877	41.2358763574098	15.5366972024554	767.172336412805
CFO/Debt	-0.165088053856471	0.0128473415	6.27760623541455	-21.2386240339156	516.511111894265
CFO	-0.189317003970588	0.0469825595	5.66866913157595	-22.2527917983175	558.014474896672
Interest Coverage	0.298785322784706	0.0432159695	5.26529062323527	28.66500329108	908.881562227916
Total Liquidity	-0.855714432617647	-0.2290984075	22.9268622695874	-9.02886624889876	237.652511064316
Current Liquidity	0.436001988609412	0.0404459715	1.90428244154403	7.94587417060794	96.0469204542072
Current Liabilities	0.07280241129	0.0417846355	0.266471036327726	3.31682151314657	38.0841910797782
EPS Before Extras	0.032195954517647	0.0660274935	6.15199404778729	14.0883693967764	556.484606798893
PE	0.497705036749412	-0.040405385	12.1025024105001	23.1493847126894	645.503742283589
ROA	0.019393932610588	-0.009403133	14.5941929459526	15.2939521891822	776.890493277924

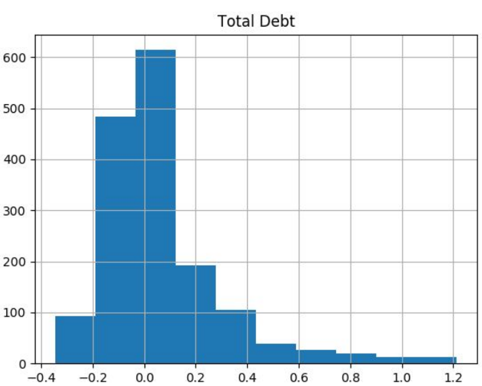

Sales/Revenues
0
100
200
300
400
500
−0.1
0.0
0.1
0.2
0.3

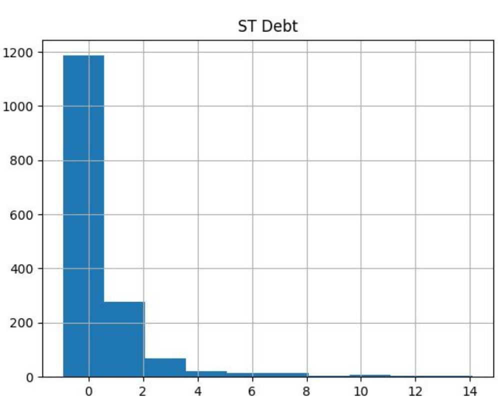

Gross Margin
0
100
200
300
400
500
600
−0.15
−0.10
−0.05
0.00
0.05
0.10
0.15
0.20
0.25

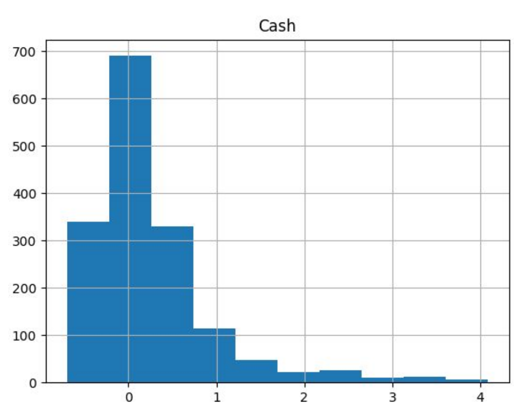

EBITDA
0
50
100
150
200
250
300
350
400
−0.2
−0.1
0.0
0.1
0.2
0.3
0.4
0.5

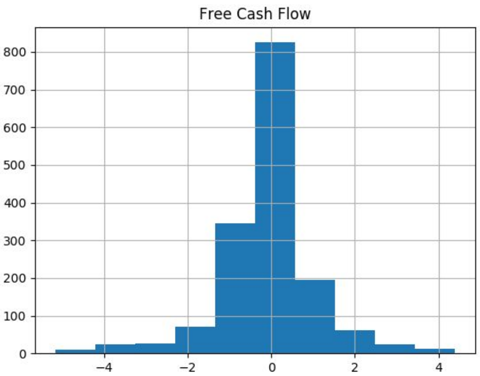

EBITDA Margin
0
100
200
300
400
500
−0.2
−0.1
0.0
0.1
0.2
0.3

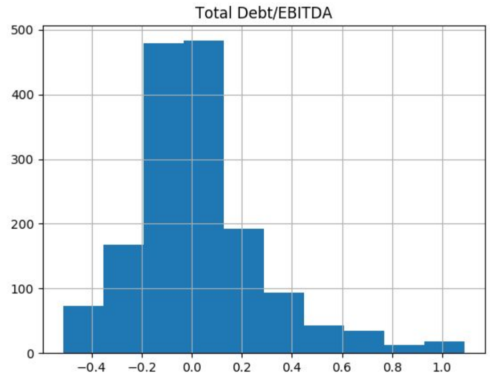

Net Income Before Extras
0
200
400
600
800
−3
−2
−1
0
1
2

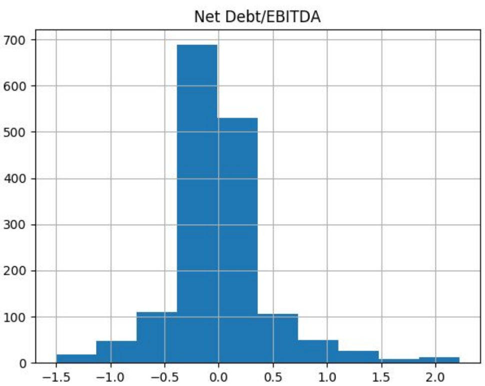

Total Debt
0
100
200
300
400
500
600
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2

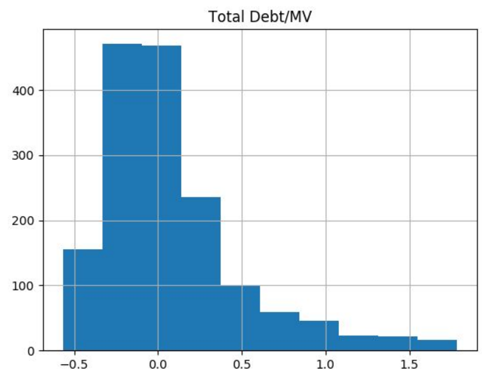

ST Debt
0
200
400
600
800
1000
1200
0
2
4
6
8
10
12
14

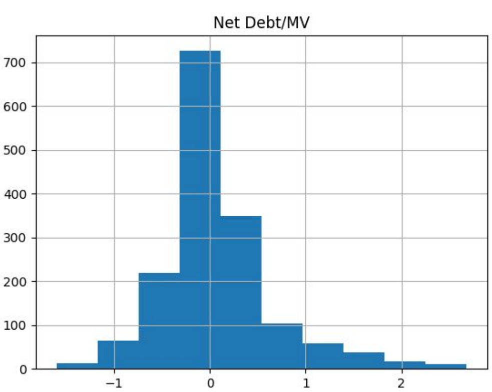

Cash
0
100
200
300
400
500
600
700
0
1
2
3
4

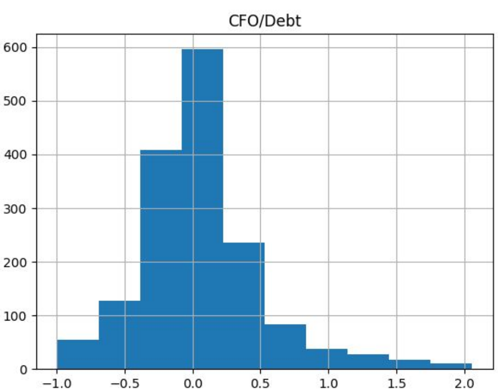

Free Cash Flow
0
100
200
300
400
500
600
700
800
−4
−2
0
2
4

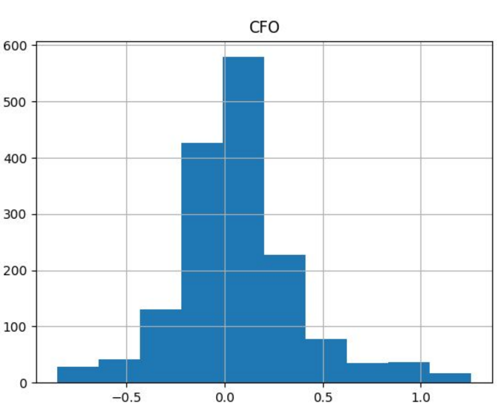

Total Debt/EBITDA
0
100
200
300
400
500
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
1.0

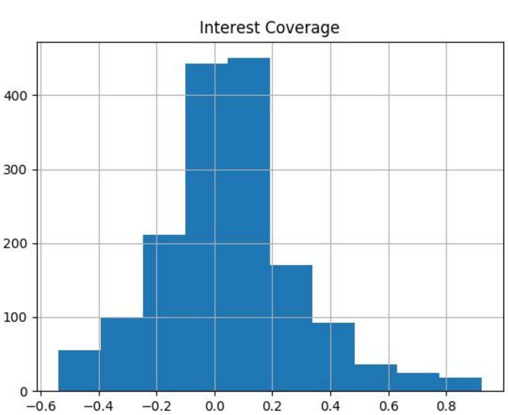

Net Debt/EBITDA
0
100
200
300
400
500
600
700
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

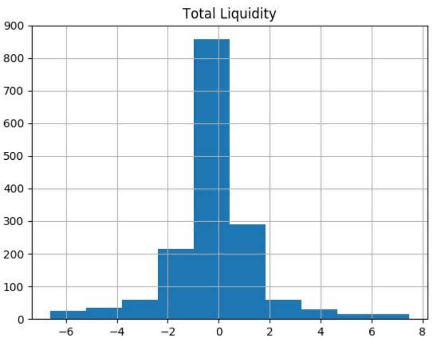

Total Debt/MV
0
100
200
300
400
−0.5
0.0
0.5
1.0
1.5

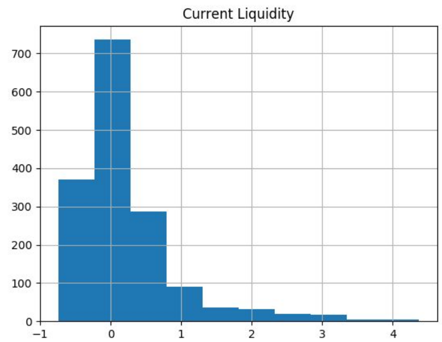

Net Debt/MV
0
100
200
300
400
500
600
700
−1
0
1
2

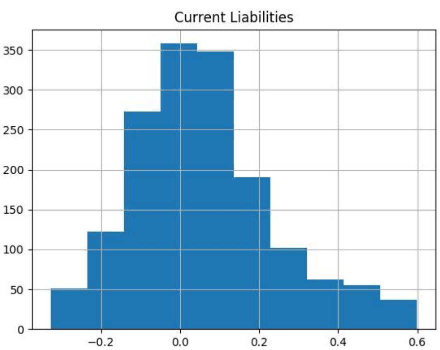

CFO/Debt
0
100
200
300
400
500
600
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

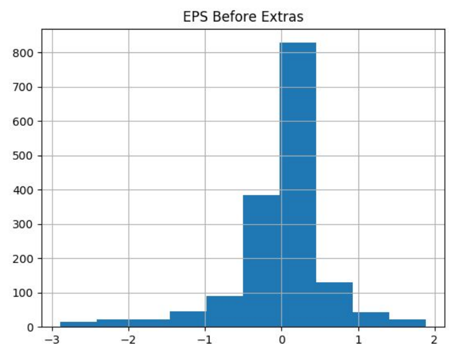

CFO
0
100
200
300
400
500
600
−0.5
0.0
0.5
1.0

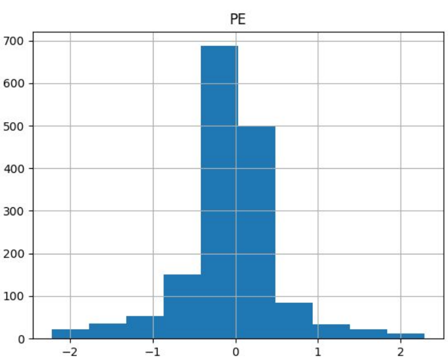

Interest Coverage
0
100
200
300
400
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8

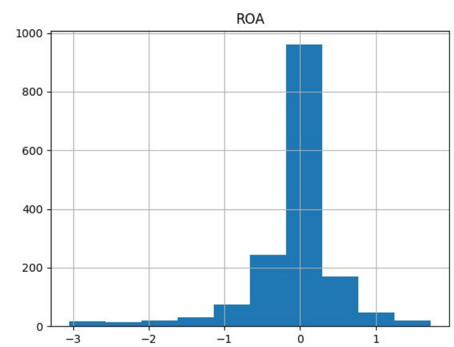

Total Liquidity
900
800
700
600
500
400
300
200
100
0
−6
−4
−2
0
2
4
6
8

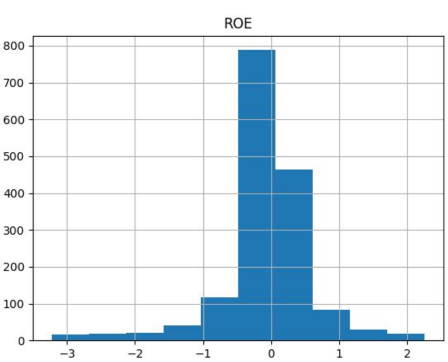

Current Liquidity
700
600
500
400
300
200
100
0
−1
0
1
2
3
4


Current Liabilities
350
300
250
200
150
100
50
0
−0.2
0.0
0.2
0.4
0.6

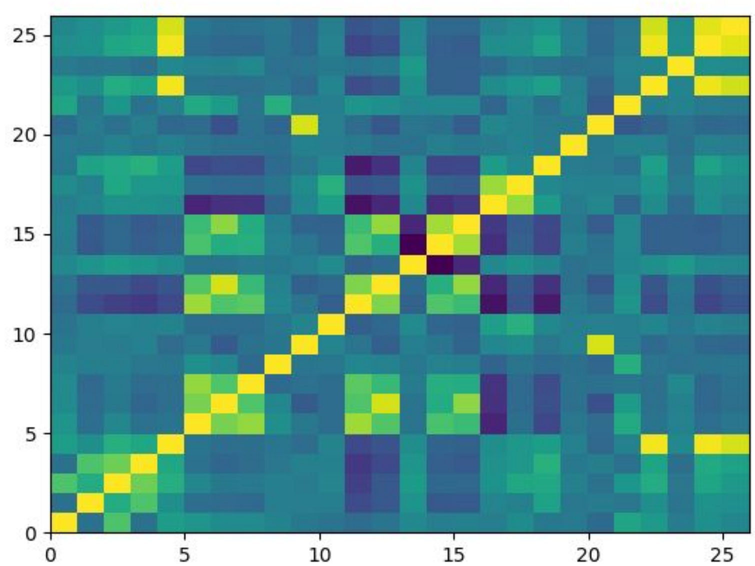

EPS Before Extras
800
700
600
500
400
300
200
100
0
−3
−2
−1
0
1
2

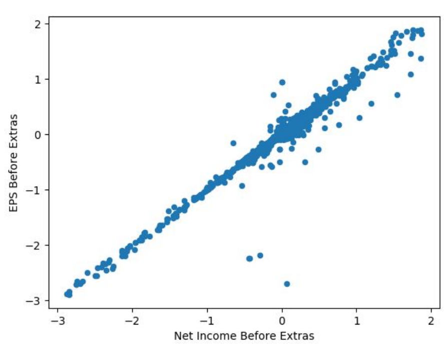

PE
700
600
500
400
300
200
100
0
−2
−1
0
1
2

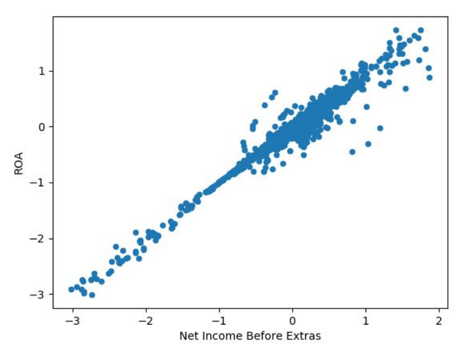

ROA
1000
800
600
400
200
0
−3
−2
−1
0
1

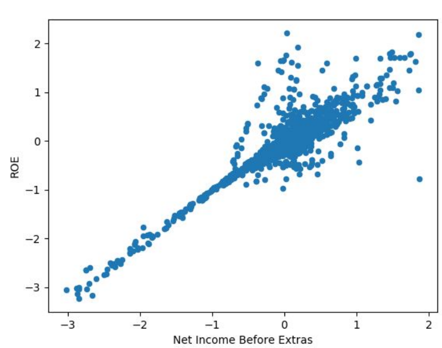

ROE
800
700
600
500
400
300
200
100
0
−3
−2
−1
0
1
2

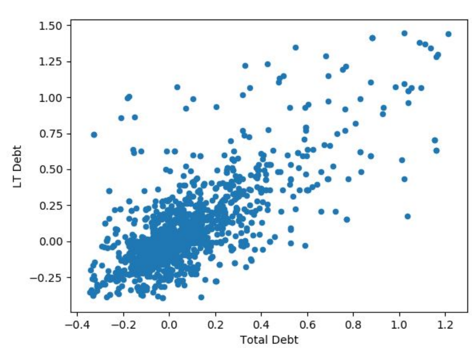
To investigate the relationship among different features, our group plotted the heatmap of the covariance matrix. And the scatter plots of those highly correlated features are drawn below. Some of the features are highly correlated. We will use feature selection techniques such as PCA, kernel PCA and LDA to avoid the curse of dimensionality.

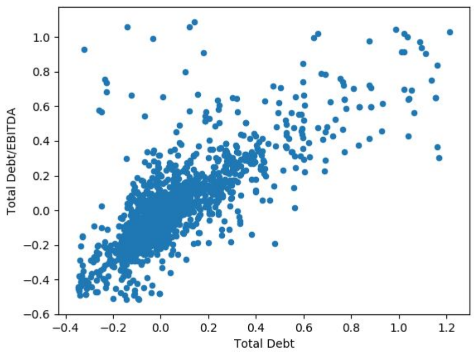


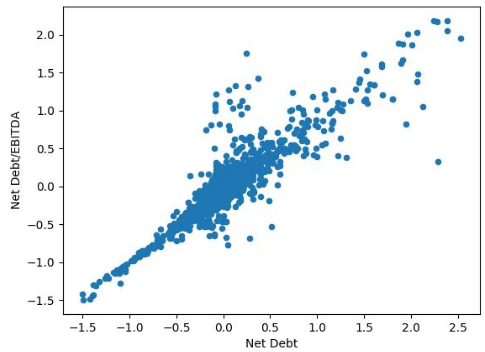


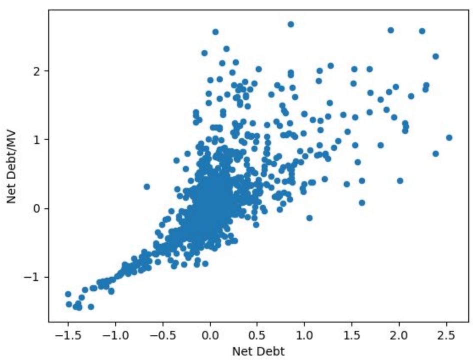


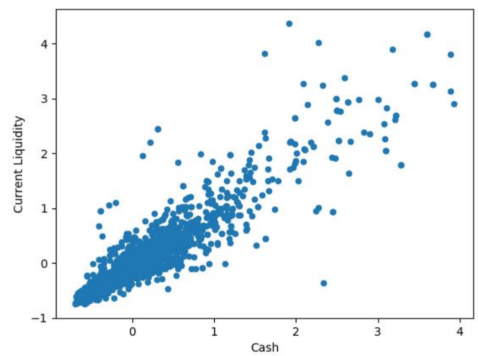


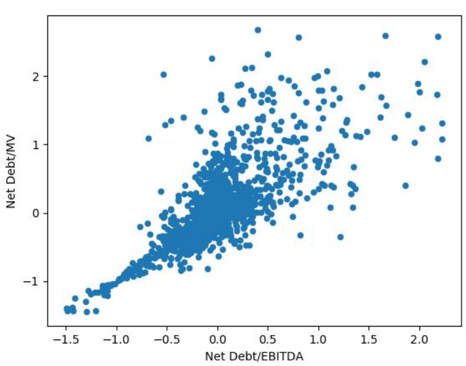


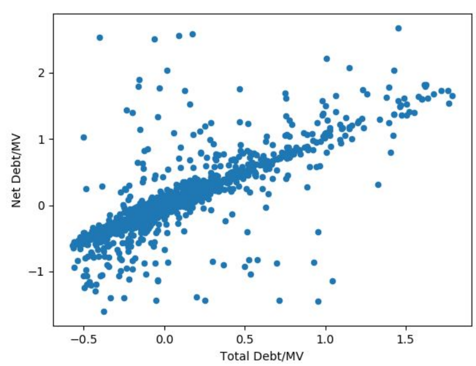

Total Debt/EBITDA
Total Debt

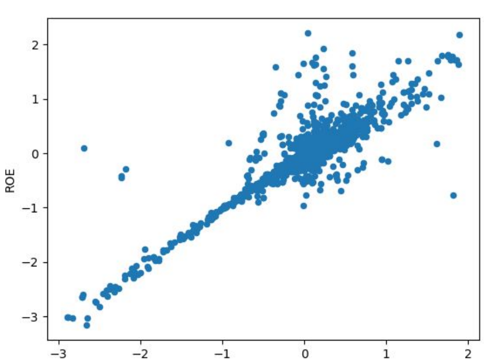

Net Debt/EBITDA
Net Debt

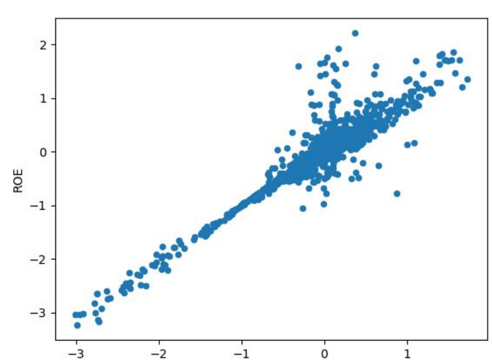

Net Debt/MV
Net Debt


Current Liquidity
Cash


Net Debt/MV
Net Debt/EBITDA


Net Debt/MV
Total Debt/MV


ROE
EPS Before Extras


ROE
ROA

Chapter 2 Preprocessing, Feature Extraction and Selection

First, check if the dataset has any NA

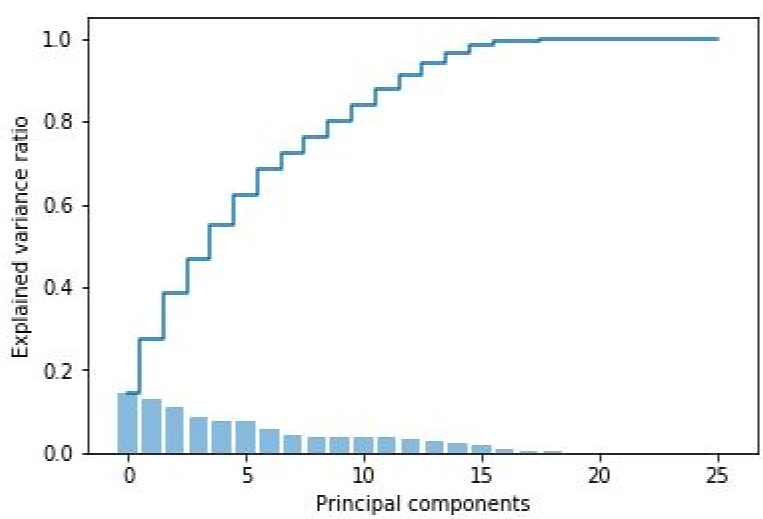
Summary the data frame

Index	Sales/Revenues	Gross Margin	EBITDA	EBITDA Margin	Income Before Ex	Total Debt	Net Debt	LT Debt	ST Debt	Cash	Free Cash Flow	Total Debt/EBITDA	Net Debt/EBITDA
count	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700
mean	0.0503777	0.0260066	0.0687183	0.0210743	0.123026	0.822405	-0.41981	1.25517	3.1428	0.46662	-0.312325	0.731197	-0.819863
std	0.16191	0.273768	0.237365	0.189025	14.4757	13.3171	28.3857	16.2245	51.9866	1.85949	8.89514	12.2805	22.0026
min	-0.661715	-0.794722	-0.782254	-0.805153	-289	-0.903014	-493.306	-0.921515	-0.997692	-0.990982	-238.75	-0.910486	-495.356
25%	-0.00569281	-0.0200279	-0.02264	-0.0427714	-0.158478	-0.0763158	-0.120725	-0.094767	-0.337959	-0.195117	-0.527219	-0.134477	-0.181621
50%	0.0339997	0.0034025	0.0494817	0.0111339	0.0566268	0.0058861	-0.00306001	-0.00207843	0.043092	0.0758196	-0.0584752	-0.0123021	-0.0344523
75%	0.0830045	0.0255948	0.124533	0.060566	0.222219	0.136449	0.160251	0.174735	0.649475	0.483113	0.396581	0.141443	0.163697
max	2.27723	3.20271	3.54242	4.14118	478.28	281.604	865.195	289.388	2038	36.98	125.786	256.05	360.926

Index	Total MV	Total Debt/MV	Net Debt/MV	CFO/Debt	CFO	Interest Coverage	Total Liquidity	Current Liquidity	Current Liabilities	EPS Before Extras	PE	ROA	ROE
count	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700	1700
mean	0.0920428	1.2702	-0.398624	-0.165088	-0.189317	0.298785	-0.855714	0.436002	0.0728024	0.032196	0.497705	0.0193939	-0.217604
std	0.385111	22.7971	41.2359	6.27761	5.66867	5.26529	22.9269	1.90428	0.266471	6.15199	12.1025	14.5942	15.389
min	-0.871567	-0.93919	-781.502	-172.654	-161.609	-0.991976	-502	-0.994141	-0.684678	-96.25	-59.7951	-305.462	-373.837
25%	-0.113241	-0.206442	-0.267345	-0.211115	-0.115159	-0.0969955	-0.857013	-0.227327	-0.0727337	-0.152894	-0.293521	-0.208483	-0.233955
50%	0.0668363	-0.0184645	-0.0320549	0.0128473	0.0469826	0.043216	-0.229098	0.040446	0.0417846	0.0660275	-0.0404054	-0.00940313	-0.0203915
75%	0.236566	0.242868	0.27471	0.251992	0.216432	0.17734	0.512778	0.416067	0.161215	0.236046	0.168897	0.156136	0.201596
max	3.96112	676.443	1352.09	15.8217	13.0058	182.132	280.139	34.3725	4.19438	187	381.243	474.847	343.145

Since the EDA shows that there are many extreme values in several features, it is important to standardize the data before feature extraction. After standardization, 3 dimensionality reduction is applied for each classification.

For the binary classification, we plan to apply Principal component analysis, linear discriminant analysis and nonlinear dimensionality reduction to the dataset. The plot of explained variance ratios are displayed below:



From the explained variance ratio plot, 15 is the reasonable number of principal component.

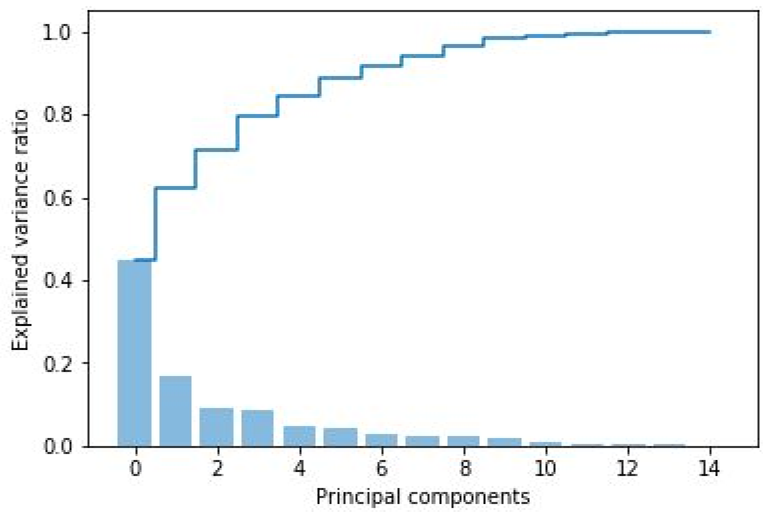
The plot indicates that the first principal component alone accounts for approximately 15 % of the variance. Also, we can see that the first 5 principal components combined explain almost 60 % of the variance in the dataset. Since it is a high-dimensional dataset, L1 regularization can be used as a feature selection technique

Then use LDA as a technique for feature extraction to increase the computational efficiency and reduce the degree of overfitting due to the curse of dimensionality in non-regularized models. Since it is a nonlinear problem, considering linear transformation techniques for dimensionality reduction, such as PCA and LDA, may not be the best choice. Try kPCA with sigmoid as kernel. Considering logistic regression is simpler implementation and easily updated feature, use it as comparison with PCA, LDA and KPCA

	Experiment 1 (binary)			
	Logistic		After tuning	
Baseline	Train Acc	0.7717	Train Acc	0.7733
	Test Acc	0.7529	Test Acc	0.76
PCA transform	Train Acc	0.77019		
	Test Acc	0.75294		
LDA transform	Train Acc	0.77176		
	Test Acc	0.75764		
kPCA transform	Train Acc	0.77411		
	Test Acc	0.75764		

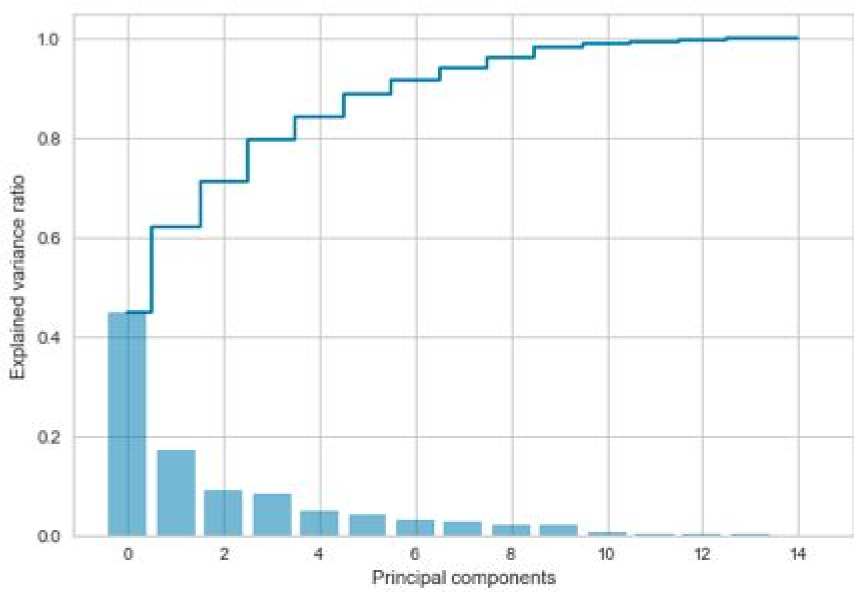
It shows that due to the dataset's feature, logistic regression is not a good choice since it does not show much linearity, before tuning the logistic model, consider the overall performance, the kPCA transform provides the best estimation. While the difference between three methods is not significant.

For Moody's credit rating , Since it is multiclass classification, logistic regression does not provide good results. We have tried fitting the model to the dataset, only 20 percent accuracy rate is achieved. Then the K Nearest Neighbor model is used to estimate the feature extraction.
PCA:



From the explained variance ratio plot, 15 is the reasonable number of principal component.

LDA:



From the explained variance ratio plot, number of principal component is 9 is reasonable.

	Experiment 2 (multiclass)			
	KNN		After tuning	
Baseline	Train Acc	1.0	Train Acc	1.0
	Test Acc	0.41411	Test Acc	0.46352
PCA transform	Train Acc	1.0		
	Test Acc	0.39529		
LDA transform	Train Acc	1.0		
	Test Acc	0.40470		
kPCA transform	Train Acc	1.0		
	Test Acc	0.41411		

It shows that due to the dataset's feature, logistic regression is not a good choice since it does not show much linearity. Even with multiclass regression, only 20 percent accuracy is reached, which is basically the same level as random guessing. It turns out that KNN is a good estimation than linear classifier. Before tuning the KNN model, considering the overall performance, the kPCA/baseline KNN transform provides better estimation, while the difference between these two methods is not significant.

Chapter 3 Model Fitting and Evaluation

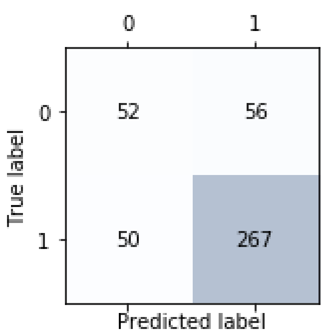
Decision Tree Classifier - Investment Grade (binary classification).

The first model that will be fitted to the dataset is the Decision Tree Classifier. Since the PCA shows the highest accuracy score in Decision Tree model in the feature extraction section, the data used in this model are all transformed through PCA. To establish a baseline, the parameters are kept default; the criterion is gini with a random state of 1. The accuracy score is then calculated for the in sample and out of sample test. The result is listed below:

Train accuracy: 1.000

Test accuracy: 0.751

Also a confusion matrix:



The results suggests that the base model is overfitting the data; the out of sample test score is much higher than the in sample test (which is 1). The model needs to be tuned to reduce the bias.

The confusion matrix shows that the model is doing a fairly good job in predicting that the asset is of investment grade, whereas it misclassified almost half of the assets that are not of investment grade. As a result the base tree appears to be overly optimistic about the credit ratings of the assets.

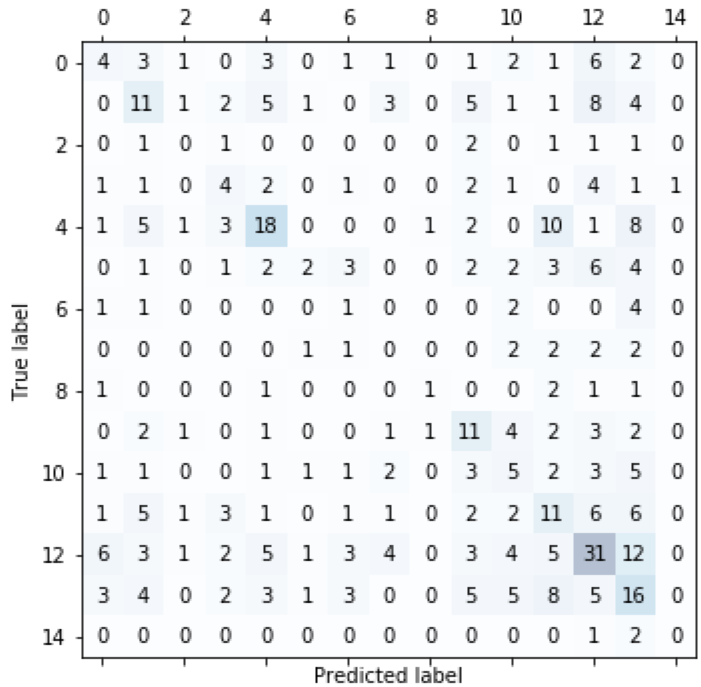
Decision Tree Classifier - Moody's score (multiclass classification)

In the case of multiclass classification, the baseline decision tree classifier yields the following results:

Train accuracy: 1.000

Test accuracy: 0.271

Also the confusion matrix:



The difference between the in sample and out of sample accuracy is surprisingly large. The out of sample test score is much lower, which indicates very high model bias. The confusion matrix confirms the idea. It shows that some Moody's ratings are largely correctly classified, especially the top and bottom ratings. The assets that has mediocre rating scores are largely misclassified. It makes sense because according to Moody's rating, bonds with the best and worst quality have only one type of rating whereas the bonds in between each have three types of ratings. So it will be very hard for the model to distinguish among those kinds of bond.

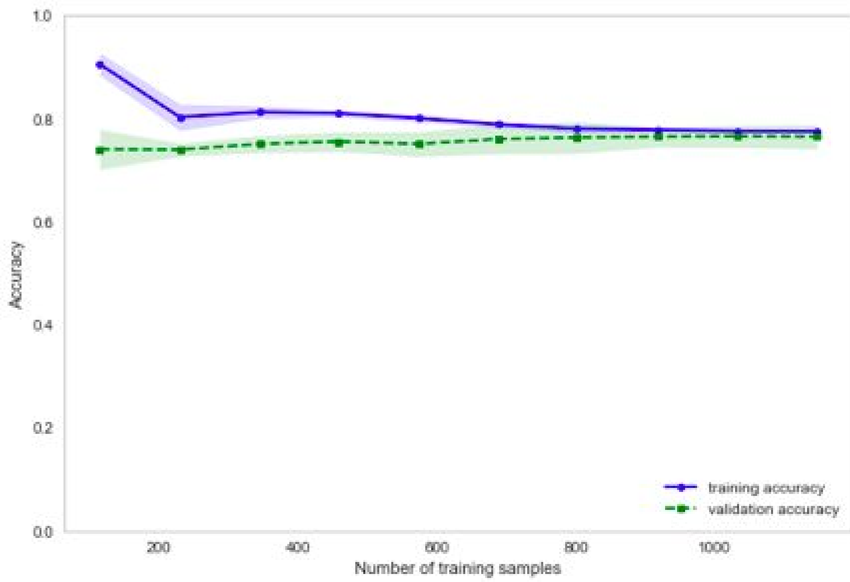
Logistic Regression Classifier - Investment Grade (binary classification).

The second model that will be fitted to the dataset is the Logistic Regression Classifier . To establish a baseline, the parameters are kept default;. The accuracy score is then calculated for the in sample and out of sample test. The result is listed below:

Train accuracy: 0.7717647058823529
Test accuracy: 0.7529411764705882

The results suggests that since the data is not linear, the logistic regression model's ability to classify is limited.

The learning curve is used to visualize of the performance:



The two curves are quite close to each other around the desired accuracy and quite stable, which shows that there is a pretty good bias-variance trade off within the tree.

	precision	recall	f1-score	support
0	0.71	0.11	0.19	108
1	0.76	0.98	0.86	317
avg / total	0.75	0.76	0.69	425

It shows that logistic regression is bad at estimating the 0 investment grade samples, high precision and low recall means it returns very few results, but most of its predicted labels are correct when compared to the training labels.

Confusion Matrix:

[[12 96]
 [5 312]]

The confusion matrix shows that the model is good at predicting the asset of investment grade and really bad at those of not investment grade.

k-fold cross-validation scorer:
CV accuracy scores:
[0.765625   0.75 0.75 0.78125 0.7421875 0.77165354
0.76377953 0.75590551 0.77165354 0.76377953]
Test accuracy: 0.7615834

KNN Classifier - Moody's score (multiclass classification)

In the case of multiclass classification, the baseline KNN Classifier yields the following results:

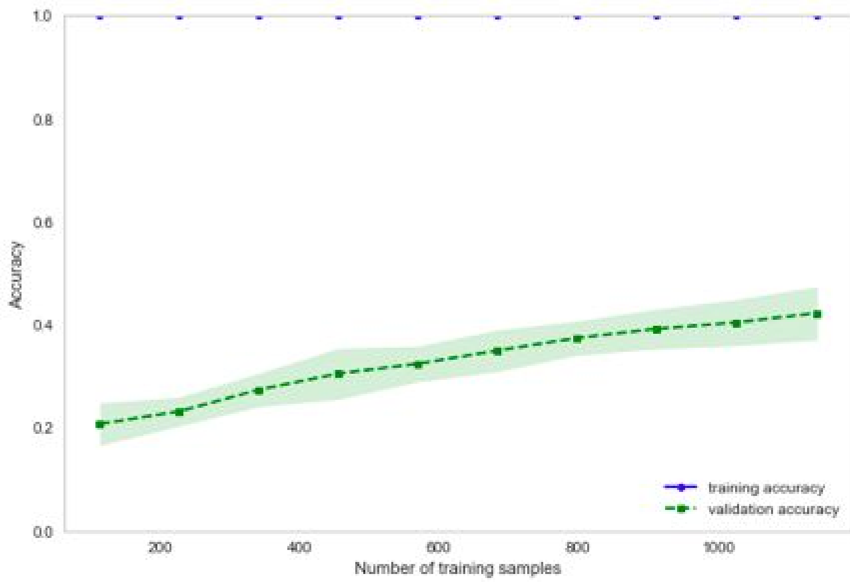
Train accuracy: 1.000
Test accuracy: 0.414

Confusion matrix:

```
[[10  1  0  0  6  2  0  0  0  0  0  2  3  1  0]
 [ 8 11  1  0 10  1  0  0  0  3  0  2  3  3  0]
 [ 0  0  1  0  0  0  0  0  0  1  0  3  1  1  0]
 [ 0  2  1  6  2  0  0  0  0  0  0  3  0  4  0]
 [ 5  2  0  0 28  0  0  0  0  1  0  5  3  6  0]
 [ 1  2  0  1  1  8  1  0  0  1  3  1  4  3  0]
 [ 0  2  0  0  0  0  1  0  1  0  0  0  5  0  0]
 [ 0  0  0  0  0  1  0  0  1  1  1  1  5  0  0]
 [ 2  0  0  0  0  0  0  0  3  0  0  0  1  1  0]
 [ 1  2  1  0  0  2  0  0  1 16  0  1  3  1  0]
 [ 1  2  1  1  1  0  1  1  0  3 10  1  1  2  0]
 [ 1  3  0  1  2  0  0  0  1  3  1 21  4  3  0]
 [ 6  2  0  2  5  1  1  1  0  6  2  6 36 12  0]
 [ 6  1  1  2  6  0  1  0  1  4  2  2  5 24  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  1  1  0  1]]
```

The difference between the training and test accuracy is surprisingly large which indicates high model bias. The confusion matrix confirms the idea. The model is good at classifying bottom ratings.. The assets that has mediocre rating scores are largely misclassified. It makes sense because according to Moody’s rating, bonds with the best and worst quality have only one type of rating whereas the bonds in between each have three types of ratings. So it will be very hard for the model to distinguish among those kinds of bond.

Cross validation score:
CV accuracy scores:
[0.48507463 0.44029851 0.46616541 0.35877863 0.3828125
0.4140625 0.38095238 0.45454545 0.49586777 0.34453782]
CV accuracy: 0.422 +/- 0.051



The training and validation curves shows there the model has very high variance. Further improvement of the model will include ensemble using bagging in the next section. Meanwhile, it seems that the KNN is a good model fit the Moody's rating classification. The accuracy increases robustly with the increase of samples, if with more samples, the model will provide better estimation.

Support Vector Machine

Support vector machine is a supervised classification algorithm. It is designed to be a linear classifier but can be generalized to nonlinear case using the so-called kernel tricks. In this project, we applied support vector machine to classify both Investment Grade and Credit Rating. For investment grade, we were able to achieve an out-of-sample accuracy of 82.94%. Since there are 1700 firms and 1287 of them are considered investment grade, a random guess will give us an accuracy of 75.71%. SVM shows a clear sign of improvement. For credit rating, the out of sample accuracy decreases to 43.23%. Since the most common credit rating appeared 326 times, a random guess would be accurate 19.2% of the time. Therefore, although the test accuracy of SVM seems unsatisfactory, there is still an indication of improved performance. The confusion matrices of both classifications are listed below.

Investment Grade

45	39
19	237

Credit Rating

7	2	0	0	5	0	1	0	0	0	0	2	4	3	0
7	9	0	0	2	0	1	0	0	4	1	1	4	4	0
0	1	1	0	0	0	2	0	0	0	0	1	0	0	0
1	1	0	5	0	0	0	0	0	1	0	4	2	1	0
2	5	0	1	23	1	0	0	0	0	2	3	2	3	0
0	1	0	0	1	5	0	0	0	2	1	1	4	4	0
0	0	0	0	0	1	2	0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	3	3	0
1	1	0	0	1	0	0	0	1	2	0	0	0	0	0
1	0	0	0	0	0	0	0	0	16	0	1	5	0	0
1	1	1	0	1	1	1	1	0	3	8	1	4	0	0
2	2	1	1	3	0	0	0	0	1	1	13	4	3	0
2	3	0	2	1	1	2	0	0	3	1	7	38	2	1
3	3	0	0	4	1	1	0	0	3	5	2	2	18	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Chapter 4 Hyperparameter Tuning

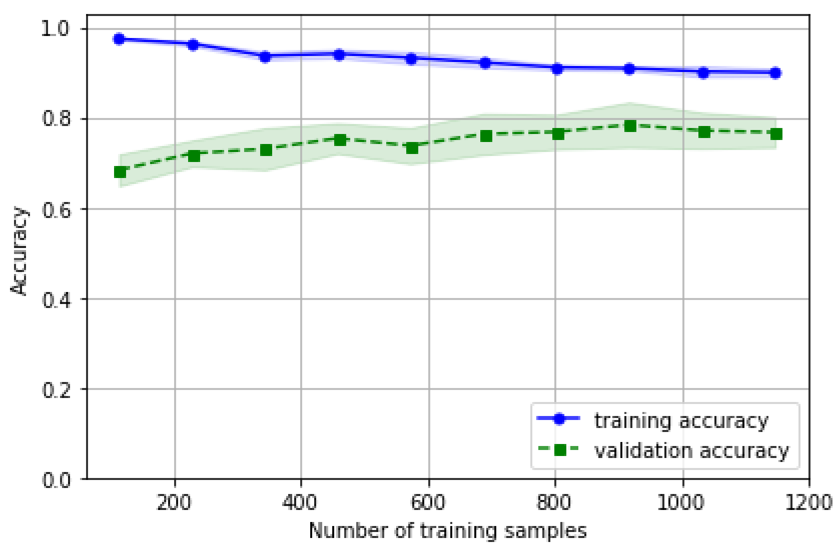
Decision Tree Classifier - Investment Grade (binary classification).

By using Grid Search, the optimum max depth is found to be 7, which results in the following accuracy score:

Train accuracy: 0.896

Test accuracy: 0.765 best score: 0.775

The out of sample accuracy score increases from 0.75 to 0.77. It does not seem like a large improvement, but still the problem of overfitting is slightly reduced. The learning curve provides a better visualization of the performance:



.

The two curves are quite close to each other around the desired accuracy, which shows that there is a pretty good bias-variance trade off within the tree.

Decision Tree Classifier - Moody's score (multiclass classification)

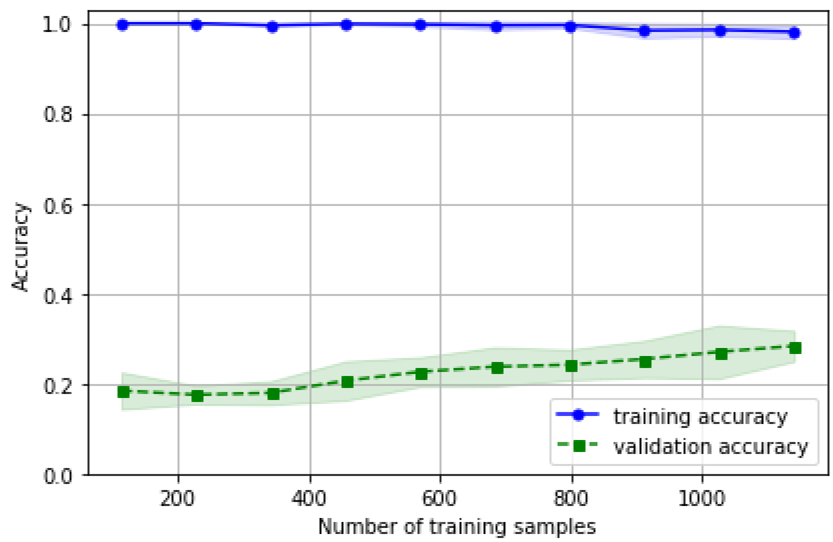
In hope of improving the overfitting, the grid search for tuning the tree model yields a max depth of 19, which means the tree branch is much deeper than the binary classification. It makes sense since the multiclass is more sophisticated than the binary. It also explains the problem of overfitting with a very branched tree. The tuned model has the following accuracy score:

Train accuracy: 0.947

Test accuracy: 0.282 best score: 0.285

The score has improved by a slight amount. However, the improvement of tuning does not seem to be significant.

The learning curving is shown below:



the training and validation curves are very far away from each other, which means there the model has very high variance. There might also be bias in the tuned model, but it is not clear enough in the graph.

Further improvement of the model will include ensemble where the models discussion later will be combined into a Majority Voting Classifier etc. in the next section. Meanwhile, it appears that the decision tree is a fairly good model for the investment grade classification, which is binary. On the other hand, it does not seem to fit the Moody's rating classification. Hopefully, other models, which will be mentioned below, could have a better performance.

Logistic Regression Classifier - Investment Grade (binary classification).

Use RandomizedSearchCV to tune the logistic regression classifier

Best parameters:
C=55.992073973395804,
multi_class='ovr',
penalty='l2',
solver='lbfgs'
penalty='l2'(since lbfgs solver only support l2 penalties)

Improved test accuracy: 0.762
The test accuracy improves from 0.752 to 0.762, resulting a slightly better estimation.

KNN - Moody's score (multiclass classification)

Use GridSearchCV to hypertune the parameters of KNN to improve the overfitting.

best score is: 0.46352941176470586
best params are: {'algorithm': 'auto', 'leaf_size': 1, 'n_neighbors': 1, 'weights': 'uniform'}
Origin test predict accuracy score: 0.41411764705882353

It shows that the test accuracy improves to some degree after use gridsearchCV to tune the parameters of KNN.

For support vector machine, we used Gridsearch to look for the best parameter combinations. For investment grade classification, it turns out feature selection techniques such as PCA, Kernel PCA, and LDA are not very helpful. The best parameters are kernel = rbf, C = 100, gamma = 0.1. The out of sample accuracy is 82.94%. For credit rating classification, Kernel PCA slightly outperforms other feature selection techniques. The best parameters are kernel = rbf, C = 100, gamma = 1. The out of sample accuracy is 43.24%.

Chapter 5 Ensembling

Majority Voting - Investment Grade (binary classification).

The first method of Ensembling is the Majority Voting Classifier. The Majority Voting also includes three models: Logistic Regression, Decision Tree and Kernel SVM. The accuracies are displayed below:

Accuracy: 0.76 (+/- 0.01) [Logistic Regression]

Accuracy: 0.77 (+/- 0.04) [Decision Tree]

Accuracy: 0.82 (+/- 0.04) [Kernel SVM]

Accuracy: 0.80 (+/- 0.03) [Ensemble]

The Majority Voting achieves an accuracy of 0.80 with a relatively small variance of 0.03, which is higher than the accuracy of the logistic regression and decision tree. Kernel SVM seem to be of better performance but with a higher variance. In conclusion, the majority voting classifier provides a better bias and variance trade off. The resulting model is performing well on the sample

Majority Voting - Moody's score (multiclass classification)

Accuracy: 0.41 (+/- 0.04) [KNN]

Accuracy: 0.24 (+/- 0.02) [Decision Tree]

Accuracy: 0.39 (+/- 0.03) [Kernel SVM]

Accuracy: 0.40 (+/- 0.03) [Ensemble]

The Majority Voting achieves a score of 0.40 which is much higher than Decision Tree. It is safe to say that a single decision tree is not enough for the classifying the multiclass dataset. It is more susceptible to overfitting due to the high dimensionality of the dataset. The ensembled estimator achieves a higher performance with an acceptable variance. One would conclude that there is a good bias and variance trade off.

Bagging - Investment Grade (binary classification).

The second method of Ensembling is the Bagging.
Consider dataset's high dimensionality, which can easily lead to overfitting in single decision trees, the bagging algorithm is suitable for this situation.

Using bag and compared with decision tree classifier, there is significant improvement of the testing accuracy:

Decision tree train/test accuracies 1.000/0.772

Bagging train/test accuracies 1.000/0.871

Bagging provides significant improvement in estimation accuracy. The test accuracy increase from 77.2% to 87.1%.

Bagging - Moody's score (multiclass classification)

Use Bagging :

Decision tree train/test accuracies 1.000/0.367

Bagging train/test accuracies 1.000/0.624

Bagging provides significant improvement in estimation accuracy. The test accuracy increase from 36.7% to 62.4%.

We also tried AdaBoosting with support vector machine. The out-of-sample accuracy of investment grade classification decreases from 82.94% to 75.58%. For credit rating, the out-of-sample accuracy decreases from 43.23% to 19.70%. The reason of reduced accuracy might be that SVM is not an ideal weak learner. Hence aggregating the results of several SVM classifiers may not improve the performance by much.

Chapter 6 Conclusion

In this project, we worked on the MLF_GP1_CreditScore data set to build an optimum model with a good bias and variance tradeoff with the goal to classify the 1700 observations of 26 financial and accounting metrics for a set of firms in several different industries. The first objective is a multiclass classification to classify the Moody's score, which contains 16 categories, of the assets. The second objective is to classify whether the assets are of investment grade or not.

The process starts with an exploratory data analysis. It is shown that many features have extreme outliers. Also there a high correlation between several features. This indicates that standardization and feature extraction would be important before fitting models. Then in the feature extraction section, we examined 3 techniques: PCA, LDA and KPCA. We performed each transformation on the models that we will fit the data in the next stage to get a general idea, the values are summarized as below

	LR			KNN	
	Train accuracy	Test accuracy		Train accuracy	Test accuracy
PCA	0.77019	0.75294		1	0.39529
LDA	0.77176	0.75764		1	0.404705
KPCA	0.77411	0.75764		1	0.41411
	SVM			SVM	
	Train	Test		Train	Test
PCA	0.908	0.8		0.9823	0.4264
LDA	0.7705	0.7705		0.9808	0.397
KPCA	0.8779	0.8235		0.9992	0.4323
	Tree			Tree	
	Train	Test		Train	Test
PCA	1	0.751		1	0.271
LDA	1	0.647		1	0.256
KPCA	1	0.739		1	0.249

According to the accuracy scores, it seems appropriate to use PCA before the decision tree model and KPCA for the other models.

In the next step of fitting models, our group chose KNN, to start with something simple and then gradually add complexity, Decision Tree, Logistic Regression and SVM. The results are summarized below

Investment Grade (binary classification)				Moody's score (multiclass classification)	
	In sample accuracy	Out Sample accuracy		In sample accuracy	Out Sample accuracy
Decision Tree	1	0.751		1	0.271
Tuned	0.896	0.765		0.947	0.282
Logistic Regressio	0.771764706	0.752941176			
Tuned	0.77333	0.76			
KNN				1	0.414117647
Tuned				1	0.46352
SVM	0.8066	0.7647		0.2448	0.2382
Tuned	0.9235	0.8294		0.9992	0.4323
Ensembel					
Majority Voting	0.80 (+/- 0.03)			0.40 (+/- 0.03)	
Bagging	1	0.871		1	0.624
AdaBootsting	0.8015	0.7558		0.2257	0.197

It is clear that the ensembles in both classification (binary and multiclass) achieves the highest accuracy score. In the sections of individual models, several evaluation metrics and graphs are used to help interpret the accuracy. In conclusion, for the Investment Grade classification, the dataset appears to be straight forward, where all baseline models achieved 70% accuracy or higher. After further parameter tuning and ensembling, the model managed to achieve even higher accuracy with a better bias and variance trade-off. The best model for the binary classification appears to be the Bagged Random Forest model with an accuracy of 87.1%. In case of the Moody's score classification, the multiclass classification is much harder than the previous case. The high correlation among the features makes overfitting a serious drawback on the accuracy of the models. The base models all have accuracy of 20-30%. With hyper parameter tuning and ensemble methods, we managed to increase the accuracy score to around 40%. The best performing model is also the bagged random forest model with an accuracy of 62.4%, which is a significant increase than the baseline.

Overall, we have found the optimum model of classification to be the Random Forest Model with tuned parameter for both the binary and multiclass classification. The intercorrelation within the data and the fact that the credit rating seems a little vague in the middle section presented obstacles for the multiclass classification. Fortunately, ensembling provides us with a solution. Although ensembling exerts a significant increase in computational complexity, the curse of dimensionality is not serious enough, hence it still presents us with the best model.