

### **3 CORPORATE REPORTING DATA AND ANALYSES**

If we have observations of the speed performances of sports cars based on pre-determined road conditions, we can try to predict the speed performances of the next batch of sports cars based on similar road conditions. However, our predictions will reasonably be improved if we also know how the cars were made, their engine structures, gear transmission dynamics, wheel types, parts, and so on. Similarly, we could improve prediction of a company's performance – its earnings, stock returns (for a listed stock), and so on – not by just knowing the past historical data of these performances, but also by knowing the structure and business operations of the company. The latter are captured largely in accounting statements that are mandatory reporting statements to be made by the company, at least for listed companies that tap on funding from the public.

A company can produce its accounting statements as frequently as monthly for its internal use, but official submissions to the regulators as public documents for public access, and hence availing such data for data analyses, are typically on standard time frames in U.S. The U.S. Securities and Exchange Commission (SEC) requires publicly listed firms to submit annual reports called 10-K forms/filings that include audited financial statements. The firms are also required to disclose corporate performance information on an ongoing quarterly basis in 10-Q forms.

In this chapter, we show how accounting variables can be used to predict future (following year's) earnings or profit margins.

#### **3.1 Financial Statements**

There are 3 key financial reporting statements in the corporate reports – the Balance Sheet, the Income Statement, and the Cashflow Statement. The Balance Sheet is a snapshot of a company's assets and liabilities at a specific time point. The Income Statement is a record of revenues, expenses, and net profits incurred by a company over the specified reporting period, e.g., a year or a quarter. The Cashflow Statement is a report of the aggregated sources and uses of cashflows of the company, including operating cashflows, investing cashflows, and financing cashflows.

The basic ingredients of a Balance Sheet are as follows in a fictitious firm Z. The units are in \$ millions. Year is YY.

Z Inc.  
Balance Sheet  
(As of) December 31, YY, \$ millions

<u>Assets</u>		<u>Liabilities and Equity</u>	
Current Assets		Current Liabilities	
Cash	10	Accounts Payable	10
Accounts Receivable	30	Short-term Debt <sup>+</sup>	10
Inventories	20	Taxes Payable	<u>5</u>
Prepaid Expense*	<u>10</u>		
Total Current Assets	<b><u>70</u></b>	Total Current Liabilities	<b><u>25</u></b>
Investments**	<b><u>10</u></b>	Long-Term Debt	<b><u>45</u></b>
		Total Liabilities	<b><u>70</u></b>
Property, Plant, & Equipment		Stockholder's Equity	
Land & Buildings	50	Paid-in Capital	70
Equipment	20	Retained Earnings	<u>20</u>
Less accumulated dep'n <sup>++</sup>	<u>(10)</u>	Total Owner's Equity	<b><u>90</u></b>
Net PP&E	<b><u>60</u></b>		
Intangible Assets			
Goodwill	10		
Tradenames	<u>10</u>		
Total Intangible Assets	<b><u>20</u></b>		
Total Assets	<b><u>160</u></b>	Total Liabilities and Equity	<b><u>160</u></b>

\*Insurance coverage, leased office equipments, etc., where expenses have been paid but advantages will accrue over time in incomes. These are recognized as assets.

\*\* Long-term over 1 year, e.g., long bonds.

<sup>+</sup> Example is bank overdrafts payable in less than a year whereas long-term debts are due in more than a year.

<sup>++</sup> dep'n stands for "depreciation" – this amount is not expensed yet, i.e., not yet a cash outflow, but is typically kept to pay for replacement of the capital in due time.

The net working capital (NWC), that is the company's total current assets less total current liabilities, can be computed from the Balance Sheet. Positive NWC enables the company to fund its current operations on an ongoing daily

basis as the current assets can be easily liquidated for cash or enable more short-term loans.

The Balance Sheet also indicates the Invested Capital (total long-term debt and equity) = NWC + Investments + PP&E + Intangibles. The company's net profits should be measured versus the invested capital to see the return rate of capital amongst other measures. The amount of long-term debt versus equity also provides a measure of the leverage, whether the company is excessively using debt to fund its business operations, and whether this may become detrimental when interest costs of debts rise.

The basic ingredients of an Income Statement are as follows in the fictitious firm Z. The units are in \$ millions. Report is for Year ended December 31, YY.

Z Inc.  
Income Statement  
For the year ended December 31, YY, \$ millions

Sales	110
Cost of Goods Sold (COGS)	<u>80</u>
Gross Profit (Total Income)	<u><b>30</b></u>
 Selling, General & Admin Expenses (SG&A)	 12
Depreciation & Amortization* (D&A)	<u>5</u>
Total Expenses	<u><b>17</b></u>
 Operating Profit/Income (EBIT**)	 <u><b>13</b></u>
 Net Non-operating Revenue <sup>+</sup>	 3
Less Interest Expense	<u>(4)</u>
Net Income before tax	<u><b>12</b></u>
Less Income Tax	<u><b>(2)</b></u>
 Net Income (EAIT <sup>++</sup> )	 <u><b>10</b></u>

\*Depreciation is expensing a fixed asset to reflect its deterioration and to save the cash and accumulate over time to pay for replacement of the capital in due time. Amortization is the spreading of cost of an intangible asset over time.

\*\*Earnings before interest and tax

<sup>+</sup>Non-operating revenue includes interest received from investments, and gains (losses) in investments. It may also include extraordinary item such as a one-time sale of some plants or buildings.

<sup>++</sup> Earnings after interest and tax. Note here income tax rate is 2/12 or about 16 2/3%.

The Income (or Profit & Loss) Statement reflects if the company is profitable in that period. The time series of income statements also show the trend in sales and various costs, and thus allow a gauge of the sustainability of the business operations. Ratios of Income Statement items and also ratios of Income Statement items versus Balance Sheet items such as net income over total equity (return on equity, ROE), net income over sales, operating margin (operating profit/gross profit), show how profitable the business is.

These can be compared across different companies to see how well they perform relative to one another. Other financial ratios such as sales in Income Statement over total assets in the Balance Sheet (asset turnover ratio) indicate the company's efficiency or effectiveness in producing sales from assets. One key ratio in valuation of a company's share for mergers and acquisition, takeover, or leveraged buyout of a company's shares is its Price earnings ratio (P/E) where P is the share price and E is the earnings per share (EPS). EPS is net earnings (EAIT, excluding extraordinary items) divided by total shares outstanding. For companies not facing credit risk, a low P/E often signal a good buy.

#### Z Inc.

#### Cash Flow Statement

For the year ended December 31, YY, \$ millions

Add Net Income	10
Add Non-cash charges (D&A)	5
Less Increase in Current Assets	(20)
Add Increase in Current Liabilities	<u>15</u>
Operating Cashflow	<u><b>10</b></u>
Less Capital Expenditures	(20)
Add Sale of PP&E	<u>0</u>
Investing Cashflow (net CAPEX)	<u><b>(20)</b></u>
Add additional (seasoned) Equity issue	15
Add new Long-term Debt issue	15
Less Dividends to Equity holders	<u>(5)</u>
Financing Cashflow	<u><b>25</b></u>
Total Cashflow	<u><b>15</b></u>

The financial analyses of a company is not complete without the Cash Flow Statement. A cash flow statement informs how much cash is entering and exiting the company business during a given period. It indicates if there is enough cash or liquidity to fund the company's operating expenses and pay its debts. If not, the Operating Cashflow would be negative. It complements the other two Statements. Cashflow reporting is harder to manipulate than that of net income – the revenues and costs in the Income Statement are subject to accrual basis in accounting that allows discretion by the company to report revenues (accrued revenues) before the money is received or delay reporting of cost, and thus may provide a better optics of positive net income when the cashflow could be negative.

The total Cashflow of the company is the sum of Operating Cashflow, Investing Cashflow, and Financing Cashflow. Together they reflect the cash situation of the company and if there are enough Cash flows to continue smooth operations of the business. Low or negative cashflows may adversely affect the company's stock prices. However, there are peculiar situations where a company may appear to have negative cash flows due to its decision to expand and incur large capital expenses with the view of huge payoffs in the near future – this may on the contrary augur great stock returns in future. In the example of Z Inc., there is a capital expenditure of \$20 million. This amount would appear in the Balance Sheet as an addition to PP&E and an increase in Long-Term Debt. It does not appear in the Income Statement. The changes in current assets and in current liabilities are not reflected in the Income Statement but are reflected in the Balance Sheet as adjustments in Current Asset and in Current Liability from the previous reporting year.

Often, free cashflow (FCF) is also calculated. This is Operating Cashflow less Investing Cashflow (net CapEx) with after tax interest payment added -- FCF is the cash flow available for the company to repay creditors or pay dividends and interest to investors. FCF per share can augment use of the Earnings per share (EPS) as it can better reflect availability of cash, though the number may be lumpy and uneven as CapEx is lumpy over time. In the above example,  $FCF = 10 - 20 + 4(1 - 2/12) = -6 \frac{2}{3}$ , which is generally not a positive signal. The positive total cashflow in the example is due to new issues of additional equity and long-term debt of \$15 million each.

The 3 key accounting statements are inter-related. Income statement's net income and non-cash charges are key components of the operating cash flows

in the Cash Flow statement. Net income in the Income Statement has direct impact on stock prices, hence equity value. This equity value is reflected in mark-to-market valuation of stocks in the Balance Sheet. Changes in asset and liabilities in the Balance Sheet are also reflected in the Cashflows. Total Cashflow (including paid interest) is approximately reflected as change in the Cash item in the Balance Sheet. In the rest of this chapter, we show how to employ accounting variables to predict earnings change using logistic regression. Performance measures of any prediction are discussed next.

### **3.2 Binary Classification Performance Metrics**

In statistical hypothesis testing, when a null hypothesis that is true is rejected (not accepted), it is called a Type 1 error. When a null hypothesis that is false is accepted (not rejected), it is called a Type 2 error.

In a binary classification problem, there are two classes or types of the targets or labels in the sample. Suppose we define Type A to be the positive class and Type B to be the negative class. The definition of whether a type is a positive or a negative case is subjective. For example, a company that defaults (within a specific future horizon) may be Type A and one that does not is Type B. A true positive (TP) is an outcome whereby the model correctly predicts that the class/type belongs to the positive case, i.e., correctly predicting a firm will default. A true negative (TN) is an outcome whereby the model correctly predicts that the class/type belongs to the negative case, i.e., correctly predicting a firm will not default.

A false negative (FN) is an outcome where the model incorrectly predicts that the class/type belongs to the negative class, i.e., incorrectly predicting a firm will not default when it does default as in a positive case. This may be called a prediction Type 1 error if we associate the positive class with the classical idea of null hypothesis. A false positive (FP) is an outcome where the model incorrectly predicts that the class/type belongs to the positive class, i.e., incorrectly predicting a firm will default, when it does not. This may be called a prediction Type 2 error. However, it may be redundant to invoke use of Type 1, 2 errors in the prediction context.

The confusion matrix refers to the matrix containing the numbers of cases (values) in each of the 4 situations after prediction, viz.,

		Predicted Values	
		Positive	Negative
Actual Values	Positive	<b>TP</b> 10	<b>FN</b> 10
	Negative	<b>FP</b> 80	<b>TN</b> 900

Figure 3.1

Suppose there are a total of 1000 cases/firms in the cross-sectional sample during a specific time-period. Number of TP case is 10. This is simply written as TP = 10. FP = 80. FN = 10, and TN = 900. Note that the number of positive cases and number of negative cases are imbalanced, i.e., 20 versus 980.

For imbalanced cases, the accuracy measure, depicting the percent of correct predictions of positive or negative cases, may not be very informative. In this confusion matrix,

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{FN} + \text{FP} + \text{TN}) = 910/1000 = 91\%.$$

When knowing 980 out of 1000 cases are negative cases, one could use a naïve approach of predicting all cases as negative. This leads to TP = 0 and TN = 980, so accuracy is 98%. Even if someone attempts to predict the 20 positive cases (but getting them wrong), then FN = 20 (all actual positive cases are predicted as negative cases) and TP = 0. With 20 predictions on positive cases, then FP = 20. And TN = 960. Accuracy would still be a high 96%. Such high accuracies are due to the imbalance.

Naïve prediction of negative cases or trusting high accuracy in imbalanced data prediction could be bad if the context is to be able to predict the default (positive) cases and where inability to predict such cases that would default is

very costly. Imbalanced data as such can be helped by randomly deleting some negative cases or resampling on positive cases.

Besides the misleading high accuracy measure in such a situation, other predictive performance measures as follows could possibly provide more information on the predictive ability of the algorithm.

Another prediction performance measure is precision, which is

$$\text{Precision} = TP / (TP + FP)$$

that shows what percent of the predicted positive cases turns out to be correctly positive. Here it is  $10 / (10 + 80) = 1/9$  or about 11.11%. This provides an indication of the performance of the model if predicting positive cases is important. A naïve approach could not be of use when precision is measured.

Another prediction performance measure is recall, which is

$$\text{Recall} = TP / (TP + FN)$$

that shows the percentage of the actual number of positive cases that were predicted correctly. Here it is  $10 / (10 + 10) = 50\%$ . Recall is sometimes also called the true positive rate (TPR) or sensitivity.

Whether precision or recall is a more informative measure depends on whether FP or FN is more of a problem, respectively. Suppose a bank is making a prediction if the firms it lends money to will default or not in the short run, i.e., predicting if the firms are positive cases (default cases). If a prediction is correct, i.e., TP, the bank can either recall the loan before the firm defaults or else hedge the default by buying credit protection from a secondary market such as credit default swaps. If the prediction is incorrect, i.e., FN, which is predicting no default when the firm defaults, then it is costly to the bank due to the default losses. In this situation, a high FN is bad. Thus, a large recall (ratio)  $TP / (TP + FN)$  with small FN is an important prediction performance metric for this bank. Recall (or TPR) marks the percentage of the actual number of default firms that were predicted correctly. In this situation, an FP (predicting a firm to default when it does not) may only cost the bank insurance payment fees in insuring the credit risk, which is a smaller sum. Thus, high FP or lower precision in  $TP / (TP + FP)$  is not as harmful. In other words, in this situation, higher recall is more important and more informatively significant to the bank than higher precision.

Suppose a company selling a line of sports products besides advertisements on social media decides to use signboard advertisements in selected towns to



reach out to its potential customers. Each town is either one with potential customers receptive to signboard advertisements (positive case) or not (negative case). The company makes predictions on which town is positive and which is negative.

When a positive case is predicted, the company puts in signboard advertising expenses for that town. Otherwise, it does not. If it is correctly predicted, i.e., TP, then the company will make profits. However, if the prediction is incorrect, i.e., FP, then the company would have sunk in a lot of advertising costs but got little in sales and profit. This situation of low precision or low  $TP / (TP + FP)$  due to high FP produces losses and is bad for the company. On the other hand, if the company predicts a negative case, it does not put up any signboard advertisements. However, if it is a FN, there will be missed opportunities, but the cost of this would be less than if it were a FP. Thus, low recall  $TP / (TP + FN)$  with a large FN is not as harmful or bad for the company. In other words, in this situation, higher precision is more important and more informatively significant to the company than higher recall. For the company, a higher percentage of the correctly predicted positive cases is more critical. The prediction of the signboard response could be done based on a historical database of potential customers and their features, and survey of how they responded to such advertisements.

Consumer business such as Spotify or Netflix and others would do well to predict consumer preferences correctly. For example, requiring high precision is important when high FP is a problem such as recommending music (film) of type A (positive class) wrongly to new platform subscribers who dislike music (film) of type A – this may cause business loss due to customer dissatisfaction and churning.<sup>1</sup> In other situations, requiring high recall is important when high FN is a problem such as not being able to correctly screen patients for allergies (positive class) to some treatment drugs. Such FN may result in fatal treatments using those medications/drugs.

Sometimes a mixture measure is used such as maximizing the F1-Score which is the harmonic mean of precision and recall, i.e.,

---

<sup>1</sup> Based on history of past subscribers, supervised learning on them may lead to predictive analytics that recommends music (film) types to new subscribers with the associated profiles (features). For existing subscribers with log-in history, unsupervised learning may make further recommendations such as highly correlated music (film) types to each subscriber based on the listening history.

$$F1\text{-Score} = 2/(\text{recall}^{-1} + \text{precision}^{-1}).$$

F1-Score increases with increases in recall and precision. This is useful when both recall and precision need to be high.

The above prediction performance metrics of precision, recall, and F1-score are measured with respect to the positive (P class), or type A. If we define type B as the P case, then the precision, recall, and F1-score may be different numbers with respect to type B. Suppose we define TP with respect to type A (P class) as  $TP_A$  and TP with respect to type B (P class) as  $TP_B$ . Similarly, we define FP with respect to type A (P class) as  $FP_A$  and FP with respect to type B (P class) as  $FP_B$ .

If precision, recall, and F1-score with respect to type A as P are  $\text{prec}_A$ ,  $\text{rec}_A$ , and  $\text{f1}_A$ , and precision, recall, and F1-score with respect to type B as P are  $\text{prec}_B$ ,  $\text{rec}_B$ , and  $\text{f1}_B$ , and suppose there are  $n_A$  number of cases that are of type A and  $n_B$  number of cases that are of type B, then

$$\text{Macro average of precision} = (\text{prec}_A + \text{prec}_B)/2$$

$$\text{Macro average of recall} = (\text{rec}_A + \text{rec}_B)/2$$

$$\text{Macro average of F1-score} = (\text{f1}_A + \text{f1}_B)/2$$

that are based on simple arithmetic averages. The weighted averages of these measures are respectively  $n_A/(n_A+n_B) \times \text{prec}_A + n_B/(n_A+n_B) \times \text{prec}_B$ ,  $n_A/(n_A+n_B) \times \text{rec}_A + n_B/(n_A+n_B) \times \text{rec}_B$ , and  $n_A/(n_A+n_B) \times \text{f1}_A + n_B/(n_A+n_B) \times \text{f1}_B$ .

In the example of Figure 3.1, precision with respect to A is 10/90 but precision with respect to B is 900/910. Hence the macro average of precision is 55%. The recall with respect to A is 10/20 but recall with respect to B is 900/980. The macro average of recall is 70.9%. Hence averaging generally reduces the extreme effect of imbalanced data. The weighted average precision is  $(20/1000) \times (10/90) + (980/1000) \times (900/910) = 0.97$ , a very high precision number. This reflects the big problem when using imbalanced data.

Another type of averaged prediction performance is

$$\text{Micro average precision} = (TP_A + TP_B)/(TP_A + FP_A + TP_B + FP_B)$$

$$\text{and Micro average recall} = (TP_A + TP_B)/(TP_A + FN_A + TP_B + FN_B).$$

The above two measures are equal and the same as accuracy. Such “averaged” measures, with respect to alternative definitions of positive and negative cases, are useful when there is no discrimination between the cases. They are used

more to compare with macro average measures. In the case in Figure 3.1, when the data is imbalanced, micro averages tend to produce higher scores, e.g., 0.91 here compared to the macro averages for precision and for recall of 0.55 and 0.71 respectively.

Another measure is specificity or true negative rate (TNR)

$$\text{Specificity} = \text{TN} / (\text{TN} + \text{FP})$$

This is the percentage of the actual number of negative cases that were predicted correctly. The false positive rate

$$\text{FPR} = 1 - \text{Specificity} = \text{FP} / (\text{TN} + \text{FP}).$$

FPR is the percentage of the actual number of negative cases that were predicted as FP, i.e., wrongly predicted. Note that TPR and FPR both lie between 0 and 1, possibly inclusive.

In a binary classification, the model typically will output a predicted probability of positive class (where the positive class is appropriately defined). It is typically the case to use a hyperparameter threshold of 0.5 whereby if the predicted probability is  $> (<=)$  0.5, then the case is assigned as a predicted positive (negative) case. The threshold can however be adjusted to improve performance if the types are highly imbalanced. For example, in the 1000 firms' study, given the predicted probabilities by the algorithm, precision can be raised if the threshold for predicting positive is raised so that FP, which is a relatively large number to TP, is decreased as less cases are then predicted as positive due to the increased threshold. Less of the negative cases are incorrectly predicted as positive – hence smaller FP. Thus, changing threshold is an interesting way to find more information about the predictive ability of the algorithm.

One of the most important more comprehensive performance metrics in a classification model's performance is the AUC (Area Under The Curve) of the ROC (Receiver Operating Characteristics) curve. It may also be called the AUROC (Area Under the Receiver Operating Characteristics).

The ROC is a curve that plots the TPR against FPR at various threshold values. This is shown below in Figure 3.2.

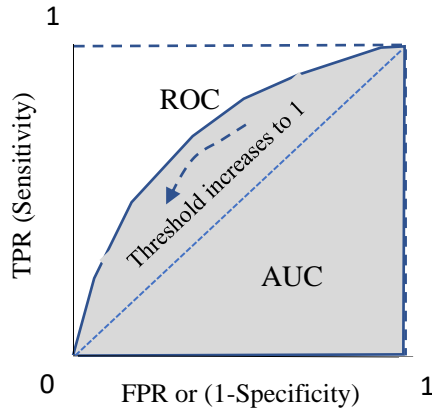


Figure 3.2

Given the predicted probabilities, as the threshold is increased toward 1, less cases are predicted as P, hence TP decreases toward zero, and FN increases (since  $TP + FN$  add up to a fixed number of actual positive cases). Thus, TPR or sensitivity  $TP / (TP + FN)$  goes toward 0. Similarly, as the threshold is increased toward 1, more cases are predicted as N, hence TN increases, and FP decreases (since  $TN + FP$  add up to a fixed number of actual negative cases). Thus, FPR or  $FP / (TN + FP)$  also goes toward 0. In other words, as threshold increases, sensitivity (recall) drops while specificity increases, i.e., detection of positive cases becomes less frequent due to more conservatism while detection of negative cases becomes more frequent.

Conversely, as the threshold approaches zero, more cases are predicted as P, hence TP increases, and FN decreases (since  $TP + FN$  add up to a fixed number of actual positive cases). Thus, TPR or sensitivity  $TP / (TP + FN)$  increases toward 1. Similarly, as the threshold is decreased toward 0, less cases are predicted as N, hence TN decreases, and FP increases (since  $TN + FP$  add up to a fixed number of actual negative cases). Thus, FPR,  $FP / (TN + FP)$ , also goes toward 1. In other words, as threshold decreases, sensitivity (recall) increases while specificity decreases, i.e., detection of positive cases becomes more frequent due to less conservatism while detection of negative cases becomes less frequent.

In the diagram, any point on the diagonal dotted line represents a situation when  $TPR = FPR$ . Points of ROC curve above this diagonal line represent the situations  $TPR > FPR$ , where the proportion of correctly predicted positive cases of all positive cases is larger than the proportion of incorrectly predicted positive cases of all negative class.

When  $TPR > FPR$ , solving  $TP/(TP+FN) > FP/(TN+FP)$  gives  $TP \times TN > FP \times FN$ . Here there is a high chance that the model can distinguish the positive cases from the negative cases because the model can detect more numbers of TPs and TNs (in product term) than FPs and FNs. On the diagonal line when  $TP \times TN = FP \times FN$ , the model is not able to distinguish between positive and negative cases, i.e., akin to predicting the classes randomly.

Across all thresholds, higher AUC (where  $AUC > 0.5$ ), or higher averages of  $TP \times TN$  than averages of  $FP \times FN$ , the better is the algorithm's ability to distinguish between positive cases as positive and negative classes as negative. The ideal situation is for the ROC curve to be sharply rising near  $FPR = 0$ , where  $TPR$  is close to 1 on the left region. There are more detailed analyses of the ROC (see Nahm (2022)).  $AUC = 1$  means perfect classification.

The AUC is based on the ROC using  $TPR$  and  $FPR$  measures based on a specific class as positive. It turns out that if we compute  $TPR$  and  $FPR$  based on the other binary class as positive instead, we would still obtain the same AUC measure. Hence it does not make any difference which case we treat as positive or which negative for computation of the area under the ROC.

For competing predicting models, the one with a higher AUC is the better model that can predict. AUC measures predictive performance in a more comprehensive way by considering all thresholds and not just the usual threshold of 0.5. When AUC is below 0.5 or close to 0.5, then the algorithm is not effective in prediction. It is possible for ROC curve to be partly below and partly above the diagonal. If this flip occurs around the threshold of typical 0.5, then the algorithm is also not effective. AUC is a single performance metric. Whether sensitivity or specificity is more important in the prediction problem does not enter into the AUC metric. This may be done separately in cost-benefit extension of the confusion matrix.

In a multi-class classification, the computations will be based on TP, FN, FP, and TN computed differently for each of the category/class. Take for example a prediction of one of 3 classes, viz., whether a business class airline passenger prefers Eastern (E), Western (W), or Mediterranean (M) food on

board, in order to make optimal food meals stock-up before flight. The numbers of (actual preference j, predicted preference k) cases are indicated in the (j,k) box below.

		Predicted Preference		
		E	W	M
Actual Preference	E	$X_{EE}$	$X_{EW}$	$X_{EM}$
	W	$X_{WE}$	$X_{WW}$	$X_{WM}$
	M	$X_{ME}$	$X_{MW}$	$X_{MM}$

The TP, FN, FP, and TN for the E class (those who prefer E – the positive cases, those who do not prefer E are the negative cases) are  $X_{EE}$ ,  $(X_{EW} + X_{EM})$ ,  $(X_{WE} + X_{ME})$ , and  $(X_{WW} + X_{WM} + X_{MW} + X_{MM})$ . The TP, FN, FP, and TN for the W class (those who prefer W – the positive cases, those who do not prefer W are the negative cases) are:  $X_{WW}$ ,  $(X_{WE} + X_{WM})$ ,  $(X_{EW} + X_{MW})$ , and  $(X_{EE} + X_{EM} + X_{ME} + X_{MM})$ . The TP, FN, FP, and TN for the M class (those who prefer M – the positive cases, those who do not prefer M are the negative cases) are:  $X_{MM}$ ,  $(X_{ME} + X_{MW})$ ,  $(X_{EM} + X_{WM})$ , and  $(X_{EE} + X_{EW} + X_{WE} + X_{WW})$ . Thus, we can compute an AUC for each class, and a clearly better performing model would produce higher AUC in as many classes.

Besides the confusion matrix performance matrix, there are also a couple of other common prediction performance metrics. The Kappa value is a metric that compares an observed accuracy with respect to that of an expected accuracy based on random chance. For example, a machine learning method could produce an observed accuracy  $A = (TP + TN) / (TP + FN + FP + TN)$ , and the sample observations could have P number of positives and N number of negatives in the sample size of  $P + N = T$ . Based on this apriori sample distribution, a random guess with a pick from the bin would suggest probability  $P/T$  designation of P and probability  $N/T$  designation of N. An optimal decision to maximize probability of correct guess would choose for example N for every draw if  $N/T > P/T$ . Then the random but rational guess would have  $TP = 0$ ,  $TN = N$ , and an expected accuracy of  $A^* = (0 + N) / T$ . In this case, the Kappa value is  $(\text{observed accuracy } A - A^*) / (1 - A^*)$  where  $A > A^*$ . This measure indicates the improvement in the accuracy via the machine learning method as a fraction of the inaccurate

segment. There is generally no definitive benchmark for what constitutes good performance for a Kappa value. Across different machine learning methods, obviously the higher the Kappa value of a particular method, then that method would be considered to have performed better.

### 3.3 Logistic Regression

Logistic regression or logit model can be used to estimate the probabilities of events, given attributes or features that can possibly determine such probabilities. If an event is a binary classification of either type A or type B (positive or negative; yes or no; default or no default; contagious or not contagious; criminal or not criminal; exit or no exit, pass or fail, and so on), we can define the  $i^{\text{th}}$  case as type A with label as  $Y_i = 1$ . Without loss of generality, if the  $i^{\text{th}}$  case is not type A, then it is type B with label  $Y_i = 0$ .

Then the logistic/logit regression model can be applied, viz.

$$E(Y_i|X_i) = \text{Prob}(Y_i = 1|X_i) = \exp(\beta^T X_i) / [1 + \exp(\beta^T X_i)] = 1 / [1 + \exp(-\beta^T X_i)]$$

where  $X_i$  is a vector of feature variables associated with case  $i$ .  $\beta$  is vector of coefficients on  $X_i$ .

For example, in studying many firms during a fiscal year, firm  $i$  could see an earnings increase ( $Y_i = 1$ ) while another firm  $j$  could see an earnings decrease ( $Y_j = 0$ ). The latter includes case of unchanged earnings. Now  $\exp(\beta^T X_i) / [1 + \exp(\beta^T X_i)]$  is the logistic (cumulative) distribution function that lies between 0 and 1. This function is also the probability of a Bernoulli distribution with binary outcome 1, and with probability  $[1 - \exp(\beta^T X_i) / [1 + \exp(\beta^T X_i)]]$  for outcome 0. Then the likelihood function of an observed training sample of  $N$  cases ( $N_1$  cases of outcome 1 and  $N - N_1$  cases of outcome 0) is

$$L = \prod_{i=1}^N [F(\beta^T X_i)]^{Y_i} [1 - F(\beta^T X_i)]^{1-Y_i}$$

where  $F(\beta^T X_i) = \text{Prob}(Y_i = 1 | X_i)$ , and  $\sum_i Y_i = N_1$ .  $X_i$  and  $Y_i$  are from the training set. The loss function (same as negative of log likelihood function in classical statistics) is

$$loss = - \sum_{i=1}^N (Y_i \log F + (1 - Y_i) \log(1 - F))$$

which is to be minimized. This is done via a numerical iterative procedure to find the optimal parameters  $\hat{\beta}$  that is then used to evaluate the predicted probability of  $\hat{Y}_k$  based on  $X_k$  from the test set cases  $k$ . In sklearn LogisticRegression app, the options can be set to (loss function) of multi\_class= 'multinomial'. This includes the binary, hence binomial situation. Solver can be by the 'newton-cg' method. Apps metrics, classification\_report, accuracy score, confusion\_matrix, etc., from sklearn can then be applied to find the various prediction performance results. Also, it should be noted that by default sklearn LogisticRegression applies L2 regularization in the estimation. For sklearn LogisticRegression, when the estimated probability of

$$F(\hat{\beta}^T X_i) = \text{Prob}(Y_i = 1 | X_i)$$

exceeds 0.5, the predicted class is considered as  $Y_i = 1$ . Otherwise, it is  $Y_i = 0$ .

However, just as in linear regression, logistic regression faces some problems when there is large multicollinearity amongst the features – even failure to converge to the optimal solution. Large multi-collinearity amongst features (or regressors in classical terminology) can be detected using 'variance\_inflation\_factor' app in statsmodels. Each feature is regressed, via linear regression, on all the other features, and the variance-inflation-factor (VIF) =  $1/(1-R^2)$  can be computed where  $R^2$  is the coefficient of determination of the linear regression. If the  $R^2 > 0.9$  (very high collinearity), then  $VIF > 10$ .

With multi-collinearity, one problem is that prediction (or forecasting in classical statistics) can become unstable and can produce bad results as the coefficients or weight vector on features for prediction can be unstable. Secondly, since these weight vectors  $\beta$  can change in unstable ways, it is also not accurate to try to find out which features  $X_i$  are more important, i.e., have bigger weights in the prediction.

To more conveniently assess the logistic regression estimation results and check on the statistical significance of the various feature impact (eliminating those that are multi-collinear and also which may have negligible impact), logit model regression in statsmodels is used to complement the prediction exercise, besides using sklearn LogisticRegression.



### 3.4 Worked Example – Data

U.S. listed company annual (end December) accounting ratios from 1971 till 2015 are used in this demonstration.<sup>2</sup> Due to the proprietary nature of the data, they are not provided herewith. Only the python notebook Chapter3-1.ipynb is provided.

Some companies appeared in the data set after 1971 while some showed data for only a few years. Some companies whose accounting ratios were not complete were not included.

The data consist of 35,074 rows of annual company data and 47 columns of accounting ratios. Some rows are different annual data of the same company. The last column 'NPM\_FWD\_CHG' (net profit margin forward change) is not an accounting ratio. It is 1 if the net profit margin (typically net profit over sales) increased in the following year. It is 0 otherwise. In the reported sklearn metrics, label of 1 is the negative case and label of 0 is the positive case (because sklearn lines up the confusion matrix starting with first row as 0 or the positive case). Note that this forward variable is of course not observed at the time-stamp of the features, but it is used for training and for testing the predictive abilities of the model using the features.

The purpose of the exercise is to harness the companies' accounting ratios each year to see if they can help predict if the following year's net profit margin would increase or not.

Code line [4] shows that in the data set there are 17,689 annual cases of NPM increase and 17,385 cases no NPM increase. Thus, the data set is balanced in the two categories.

```
[4]: ### exploring the data/ .value_counts - shows counts of unique values
      data['NPM_FWD_CHG'].value_counts()

[4]: NPM_FWD_CHG
      1      17689
      0      17385
      Name: count, dtype: int64
```

The columns with objects such as 'datadate', 'fyear', 'tic', and 'Company Name' are dropped, leaving 47 accounting ratios to form the feature space.

---

<sup>2</sup> The data set is `co_fin_ratios.csv` from the author's earlier research file and were obtained from subscribed WRDS.

Logit regression using statsmodel is first performed on the entire data set with ‘NPM\_FWD\_CHG’ as the target variable or label. This is to have an initial idea of how the various features impact or affect the label. Using statsmodel here (and sklearn logistic regression later) more conveniently provides regression estimates and their p-values so that exceedingly insignificant features may be discarded due to their redundancy. Note that in linear regression, multi-collinear features often have estimated coefficients that are not statistically significantly different from zero due to the large standard errors of the coefficient estimators. Hence some multi-collinear features may also be discarded. See code line [11]. Only a part of the output table is shown below to economize on space.

```
[11]: ### now perform logit regression using statsmodel package
      ### using statsmodel here (and sklearn logistic regression later)
      ### more conveniently provides regression estimates and p-values
      ### so that exceedingly insignificant features can be discarded
      ### due to their redundancy or multicollinearity with other features
      import statsmodels.api as sm
      result=sm.Logit(y,X).fit()
      print(result.summary())
```

Optimization terminated successfully.  
Current function value: 0.689962  
Iterations 9

Logit Regression Results						
=====						
Dep. Variable:	NPM_FWD_CHG	No. Observations:	35074			
Model:	Logit	Df Residuals:	35026			
Method:	MLE	Df Model:	47			
Date:	Sun, 22 Sep 2024	Pseudo R-squ.:	0.004541			
Time:	14:15:15	Log-Likelihood:	-24200.			
converged:	True	LL-Null:	-24310.			
Covariance Type:	nonrobust	LLR p-value:	2.449e-24			
=====						
	coef	std err	z	P> z	[0.025	0.975]
-----						
const	-0.0412	0.018	-2.315	0.021	-0.076	-0.006
CASH_DEBT	-0.0153	0.003	-4.431	0.000	-0.022	-0.009
CURRENT	-0.0038	0.006	-0.608	0.543	-0.016	0.008
PCHG_CURRENT	0.0014	0.001	1.026	0.305	-0.001	0.004
QUICK	-0.0083	0.007	-1.226	0.220	-0.022	0.005
PCHG_QUICK	-0.0022	0.002	-1.364	0.172	-0.005	0.001
DAYS_SALEAR	0.0001	3.72e-05	3.811	0.000	6.89e-05	0.000
PCHG_DAYS_SALEAR	-0.0027	0.001	-3.229	0.001	-0.004	-0.001
DEBT_EQ	-7.146e-05	0.000	-0.504	0.614	-0.000	0.000
PCHG_DEBT_EQ	0.0003	0.001	0.355	0.722	-0.001	0.002
DEPR_PPE	0.0026	0.005	0.525	0.600	-0.007	0.012

PCHG_DEPR_PPE	-0.0005	0.001	-0.889	0.374	-0.002	0.001
DIV_CASH	0.0041	0.006	0.640	0.522	-0.008	0.017
EQ_FA	1.23e-05	4.2e-05	0.293	0.770	-7e-05	9.46e-05
PCHG_EQ_FA	4.624e-06	4.26e-05	0.109	0.914	-7.89e-05	8.81e-05
INV_TURN	0.0002	0.000	1.541	0.123	-4.47e-05	0.000
PCHG_INV_TURN	-0.0004	0.000	-0.907	0.364	-0.001	0.000
INVT_AT	0.3920	0.093	4.205	0.000	0.209	0.575
PCHG_INV_AT	0.0214	0.010	2.178	0.029	0.002	0.041
LTDEBT_EQ	6.145e-06	0.000	0.014	0.989	-0.001	0.001
NI_CF	0.0010	0.001	1.317	0.188	-0.000	0.002
PCHG_OPMBD	0.0012	0.000	2.748	0.006	0.000	0.002
PCHG_PTPM	-0.0003	0.000	-1.097	0.273	-0.001	0.000
REPAY_DEBT	0.0001	0.000	1.020	0.308	-9.96e-05	0.000
ROA	0.0004	0.000	0.874	0.382	-0.000	0.001
SALE_AT	0.0351	0.011	3.097	0.002	0.013	0.057
PCHG_SALE_AT	0.0007	0.002	0.409	0.682	-0.003	0.004
SALE_CASH	8.135e-06	7.6e-06	1.070	0.284	-6.76e-06	2.3e-05
SALE_FA	-0.0013	0.001	-2.177	0.030	-0.002	-0.000
SALE_INV	-0.0001	7.97e-05	-1.356	0.175	-0.000	4.81e-05
PCHG_SALE_INV	0.0043	0.003	1.405	0.160	-0.002	0.010
SALE_NWC	7.757e-05	3.42e-05	2.265	0.024	1.04e-05	0.000
PCHG_SALE_NWC	-0.0003	0.000	-1.051	0.293	-0.001	0.000
SALE_RECT	-4.43e-05	5.6e-05	-0.791	0.429	-0.000	6.55e-05
TIE	-2.543e-05	9.83e-06	-2.588	0.010	-4.47e-05	-6.17e-06
PCHG_TIE	0.0001	0.000	0.867	0.386	-0.000	0.000
WKCAP_AT	0.0004	0.000	1.185	0.236	-0.000	0.001
PCHG_WKCAP_AT	-0.0003	0.000	-0.991	0.322	-0.001	0.000
PCHG_AT	3.149e-05	0.000	0.283	0.777	-0.000	0.000
PCHG_COGS	-9.369e-05	0.000	-0.308	0.758	-0.001	0.001
PCHG_DEPR	0.0007	0.001	0.973	0.331	-0.001	0.002
PCHG_DLTT	2.494e-05	3.38e-05	0.738	0.460	-4.13e-05	9.11e-05
PCHG_INV	-0.0075	0.004	-1.980	0.048	-0.015	-7.74e-05
PCHG_SALE	0.0009	0.001	0.849	0.396	-0.001	0.003
PCHG_WORKINGCAP	-3.247e-05	0.000	-0.171	0.864	-0.000	0.000
CHG_DIVIDEND	0.0009	0.002	0.395	0.693	-0.003	0.005
PE_RATIO	0.0002	0.000	1.792	0.073	-1.7e-05	0.000
PCHG_PE_RATIO	6.914e-05	0.000	0.418	0.676	-0.000	0.000

The logistic regression shows that estimated probability  $\text{Prob}(Y_i = 1 | X_i)$  is found via maximization of likelihood function or minimization of loss function in estimating  $\hat{\beta}$ . The estimated probability is also

$$F(\hat{\beta}^T X_i) = 1/[1 + \exp(-\hat{\beta}^T X_i)]$$

where  $X_i$  is the vector of 47 features for a particular company in a particular year, and  $\hat{\beta}$  is the vector of 47 estimated coefficients as shown in the Logit Regression Results Table. In other words,

$$\hat{\beta}^T X_i = \hat{\beta}_0 + \hat{\beta}_1 X_{i1} + \hat{\beta}_2 X_{i2} + \cdots + \hat{\beta}_k X_{ik} + \cdots + \hat{\beta}_{47} X_{i47}$$

where the  $k^{\text{th}}$  element of  $\hat{\beta}$  is scalar  $\hat{\beta}_k$ , and its associated feature of the  $i^{\text{th}}$  case in the logistic regression is  $X_{ik}$ . If  $\hat{\beta}_k > 0$ , then when  $X_{ik}$  increases, *ceteris paribus*,  $F(\hat{\beta}^T X_i)$  or estimated Prob ( $Y_i = 1|X_i$ ) increases. The latter implies that there is more likelihood that earnings would increase. On the contrary, if  $\hat{\beta}_k \leq 0$ , then when  $X_{ik}$  increases,  $F(\hat{\beta}^T X_i)$  or estimated Prob ( $Y_i = 1|X_i$ ) decreases. The latter implies that there is more likelihood that earnings would stagnate or decrease.

The results in the Table indicate that the coefficients on DAYS\_SALEAR, INVT\_AT, PCHG\_INVT\_AT, PCHG\_OPMBD, SALE\_AT, SALE\_NWC are significantly positive at p-value < 5%, and increase in their associated feature values would increase the likelihood of positive NPM change the following year. The coefficients on CASH\_DEBT, PCHG\_DAYS\_SALEAR, SALE\_FA, TIE, PCHG\_INVT are significantly negative at p-value < 5%, and increase in their associated feature values would reduce the likelihood of positive NPM change the following year.

Some of these results are easier to explain. For example, increase in PCHG\_OPMBD or percentage change in operating profit margin before depreciation often augurs improvement in NPM. Increase in sales to asset (SALE\_AT) and increase in sales to net working capital (SALE\_NWC) signals asset and funding efficiency and future improved NPM. Increase in percentage of days' sales in accounts receivable (PCHG\_DAYS\_SALEAR) which is the number of days in a year divided by the accounts receivable turnover ratio or the average time to collect cash on receivables, implies weaker collection policy and hence lower future NPM expectation. Increase in sale of fixed asset SALE\_FA is often linked with financial troubles (other than strategic business realignment) at the firm, and hence would imply poor NPM the following year. Percentage increase in investments (PCHG\_INVT) would typically mean short-run drain on NPM but perhaps with long run increases when the investments become successful. Some of the other accounting ratio such as times interest earned (TIE) impacts are harder to explain with the diverse set of firms.

Next the data is randomly split into 70% training set (24,551 rows), and 30% test set (10,523 rows).

```
[13]: from sklearn.linear_model import LogisticRegression
      from sklearn import metrics
      X_train, X_test, y_train, y_test = \
          train_test_split(X, y, test_size=0.3, random_state=0)
      logreg = LogisticRegression()
      lresult=logreg.fit(X_train, y_train)
```

sklearn LogisticRegression is then applied to the training set fitting and then the prediction of the test set. One difference between the outputs from sklearn Logistic Regression and statsmodel logit regression is that the former adds L2 regularization in the estimation, so their results may differ by a little. We shall assume that if there are hyperparameters, they were fine-tuned in a validation phase, and what we do here is to employ a re-grouped training data (earlier training data with validation data) for re-fitting (using fine-tuned hyperparameters) via ‘logreg’ in [13], and then testing via ‘logreg.predict’ and ‘logreg.score’ in [14].

The accuracy of the test set prediction is shown to be about 53%. See code line [14].

```
[14]: y_pred = logreg.predict(X_test)
      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: 0.53

The confusion matrix and the various prediction performance metrics are shown in [17], [18]. Note for confusion matrix in [17] here in sklearn, the number of true positives (‘NPM\_FWD\_CHG’ = 0 or predicted label ‘0’) is 2384. The number of true negatives (‘NPM\_FWD\_CHG’ = 1 or predicted label ‘1’) is 3090. Hence accuracy is  $(2,524+3,090)/10,523 = 53.34\%$ .

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

```
[[2384 2837]
 [2077 3225]]
```

In other words, the first line of the confusion matrix denotes the case ‘NPM\_FWD\_CHG’ = 0 as true positive, or the case that next net profit margin does not increase. The second line denotes the case ‘NPM\_FWD\_CHG’ = 1 as true negative, or the case that next net profit margin increases.

```
[18]: from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred, digits=4))
### classification_report(y_test, y_pred, digits=8) gives more decimal places
```

	precision	recall	f1-score	support
0	0.5344	0.4566	0.4925	5221
1	0.5320	0.6083	0.5676	5302
accuracy			0.5330	10523
macro avg	0.5332	0.5324	0.5300	10523
weighted avg	0.5332	0.5330	0.5303	10523

The tables show the following for precision. For the 5221 positive cases of no increase in NPM, the percentage of the predicted positive cases that turns out to be correctly positive is  $2384/(2384+2077) = 53.44\%$ . For the 5302 negative cases (or increase in NPM), the percentage of the predicted negative cases that turns out to be correctly negative is  $3225/(3225+2837) = 53.20\%$ . These are the precision numbers corresponding to positive case (row ‘0’) and negative case (row ‘1’).

In terms of recall, the tables show that for the 5221 positive cases of no increase in NPM, recall is  $2384/5221 = 45.66\%$ , which indicates a very low percentage that is correctly predicted as being positive. But for the 5302 negative cases of increase in NPM, recall is  $3225/5302 = 60.83\%$ , indicating 60.83% is correctly predicted as being negative.

It may be argued that ability to predict negative cases (increase in NPM) may be more important. However, it is much better to be able to predict both negative and positive cases.

```
[19]: import sklearn.metrics as metrics
### calculate the fpr and tpr for all thresholds of the classification
preds_logreg = logreg.predict_proba(X_test)[: ,1]
### preds_logreg or logreg.predict_prob computes 10,523 prob numbers
```

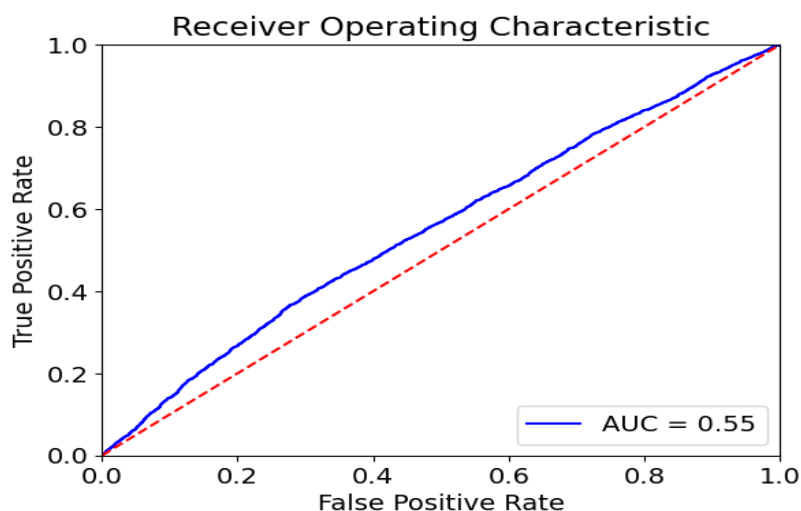
```
fpr, tpr, thresholds = metrics.roc_curve(y_test, preds_logreg)
### matches y_test of 1's and 0's versus pred prob of 1's for each of
### the 10,523 test cases
### sklearn.metrics.roc_curve(y_true, y_score,...) requires y_true as
### 0,1 input and y_score as prob inputs
### this metrics.roc_curve returns fpr, tpr, thresholds (Decreasing
### thresholds used to compute fpr and tpr)

roc_auc_logreg = metrics.auc(fpr, tpr)
### sklearn.metrics.auc(fpr,tpr) returns AUC using trapezoidal rule
### Compute Area Under the Curve (AUC) using the trapezoidal rule.
### This is a general function, given points on a curve. For computing
### the area under the ROC-curve, see roc_auc_score.
roc_auc_logreg
```

[19]: 0.5527816854878569

Code line [19] output shows how the AUC of the ROC is computed as 55.28%. The plotting of the ROC curve is shown in the output of code line [20].

```
[20]: import matplotlib.pyplot as plt
plt.rc("font", size=14)
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc_logreg)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1], 'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



### 3.5 Dimension Reduction and Principal Component Analysis

If we were able to use all 47 features in the logistic regression, it may be overfitting. One way to reduce the dimensionality and overfitting is to prune the number of features. Chen et. Al. (2022) and other showed how careful selection of the features (sometimes using stepwise approach by removing insignificant features that do not affect  $R^2$ ) could be important.

A common dimension reduction technique is principal component analysis (PCA) whereby the 47 features/variables could be collapsed (reduced) to a smaller number of variables. Each of the resulting variable is a principal component (PC) that is a linear combination of the 47 variables. One nice feature of PCA is that each PC is orthogonal to another (zero correlation) and with enough number of PCs, their total variance could match close to if not equal to the total variance of all the individual features. The orthogonality would alleviate the multicollinearity issue in regression problems, including the logistic regression.

A brief theory of PCA can be explained as follows. Suppose the  $N$  features are random variables  $x_1, x_2, x_3, \dots, x_N$ . Now let  $X_{N \times 1} = (x_1, x_2, x_3, \dots, x_N)^T$  be a random vector. Let the mean of  $X_{N \times 1}$  be  $\mu_{N \times 1}$ . Let the covariance matrix of  $X$  be  $\Sigma_{N \times N}$ . Let  $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_N$  be the  $N$  eigenvalues and  $v_1, v_2, v_3, \dots, v_N$  be the  $N$  corresponding eigenvectors (dimension  $N \times 1$ ) of  $\Sigma$  such that  $\Sigma v_j = \lambda_j v_j$  for any  $j=1,2,\dots,N$ . It can be shown that the eigenvalues of  $\Sigma$ , a real symmetric positive-definite matrix, are all (real) positive, though some may not be unique.

The solutions to the eigenvalues and eigenvectors are constrained by normalized orthogonal eigenvectors so that  $v_j^T v_j = 1$  for all  $j$ , and  $v_j^T v_k = 0$  for  $j \neq k$ . Let  $\Lambda_{N \times N} = \text{diag}(\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_N)$ , a diagonal matrix. Without loss of generality, let the diagonal elements be arranged in descending order so that the left-top element  $\lambda_1$  is the largest eigenvalue and the right-bottom element  $\lambda_N$  is the smallest eigenvalue.

Let  $\Gamma_{N \times N} = (v_1, v_2, v_3, \dots, v_N)$  be such that  $\Gamma^T \Gamma = I_{N \times N}$ , the  $N$ -dimension identity matrix. Now let random variable  $Y_j = v_j^T (X - \mu)$  which has a  $1 \times 1$  dimension. This linear combination of the  $N$  detrended random features is called the  $j^{\text{th}}$  scalar principal component (PC) of  $X$ . There are  $N$  such PCs since there are  $N$  number of eigenvectors  $v_j$  for  $j=1,2,\dots,N$ . Let  $Y_{N \times 1} = (Y_1, Y_2, Y_3, \dots, Y_N)^T$  be vector of the  $Y_j$ 's.



Properties of the PC vector  $Y$  can be seen as follows. Given  $\Gamma$ , its expectation or mean,  $E(Y) = \Gamma^T E(X - \mu) = 0$ . Variance of vector  $Y$  is  $\text{var}(Y)$  or  $[\text{cov}(Y_i, Y_j)]_{N \times N} = E(YY^T) = E(\Gamma^T(X - \mu)(X - \mu)^T\Gamma) = \Gamma^T\Sigma\Gamma$  given  $\Gamma$ . But by the definition of eigenvalues and eigenvectors,  $\Sigma\Gamma = \Gamma\Lambda$ . Hence,

$$\text{var}(Y) = E(YY^T) = \Gamma^T\Sigma\Gamma = \Gamma^T\Gamma\Lambda = \Lambda.$$

Clearly, the variance of the  $j^{\text{th}}$  principal component  $Y_j$  is the  $j^{\text{th}}$  diagonal element of  $\Lambda_{N \times N}$  or  $\lambda_j$ . Different principal components are also uncorrelated with each other. If the PCs are normally distributed, then they are independent.

From the way we construct  $\Lambda$ , the first PC,  $Y_1$ , has the largest variance followed by the variance of the second PC  $Y_2$ , and so on.

Since  $\Gamma^T\Gamma = \Gamma\Gamma^T = I_{N \times N}$ , trace of  $\Lambda$  or  $\text{tr}(\Lambda) = \text{tr}(\Gamma^T\Sigma\Gamma) = \text{tr}(\Sigma\Gamma\Gamma^T) = \text{tr}(\Sigma) = \sum_{j=1}^N \text{var}(x_j)$  or the total variance of the features. Note that the trace operation is invariant under circular shifts shown above. Therefore, the sum of all the variances of the PCs is equal to  $\text{tr}(\Lambda)$  which is also the total variance of the features. The contribution of each PC variance is also called the explained variance (information explained using the eigenvectors) and is often expressed as a ratio to the total variance, i.e.,  $\text{var}(Y_j) / \sum_{j=1}^N \text{var}(x_j)$ . The sum of these ratios across all eigenvalues equal to one.

Since we do not know the exact  $\Sigma$  nor mean  $\mu$  of the vector  $X$ , they must be estimated using the time series of the features, assuming the features are stationary. Estimate of  $\mu$  is  $\hat{\mu} = (\hat{\mu}_1, \hat{\mu}_2, \dots, \hat{\mu}_N)$  where  $\hat{\mu}_j = \frac{1}{T} \sum_{t=1}^T x_{jt}$ . The  $j^{\text{th}}$  random feature takes realized time series values  $x_{j1}, x_{j2}, \dots, x_{jT}$ .  $T$  is the sample size. Estimate of  $\Sigma$  is  $\hat{\Sigma} = [\widehat{\text{cov}}(x_i, x_j)]$  where  $\widehat{\text{cov}}(x_i, x_j) = \frac{1}{T} \sum_{t=1}^T (x_{it} - \hat{\mu}_i)(x_{jt} - \hat{\mu}_j)$ . Any computed eigenvalue  $\hat{\lambda}_j$  and eigenvector  $\hat{v}_j$  would be based on sample estimates  $\hat{\Sigma}$  and  $\hat{\mu}$ . Solution of  $\hat{\lambda}_j$  is via a polynomial equation involving determinant  $\det(\hat{\Sigma} - \hat{\lambda}_j I) = 0$ . Eigenvector  $\hat{v}_j$  is then solved in a second step. Once the eigenvectors are estimated, the  $j^{\text{th}}$  PC based on realized features data is  $Y_j = \hat{v}_j^T (X^* - \hat{\mu})$  where  $X^*$  is the realized values of  $X$ .

We will perform a principal component analysis and possible dimension reduction of the feature space. Dimension reduction occurs if we choose only a subset of the PCs incorporating PCs with the largest variances. First, we standardize the features. See code line [23].

```
[23]: from sklearn.preprocessing import StandardScaler
      scaler = StandardScaler()
      #`StandardScaler` in `sklearn.preprocessing` module is used to
      # standardize features by removing the mean and scaling to unit variance.
      X1=scaler.fit_transform(X)
```

Then we check the degree of multicollinearity that may affect the logistic regression. Code lines [26], [27] show that there are 53 correlations of pairs of features in X that have correlation exceeding 0.9.

```
[26]: # check correlation of features after standardization
      cor=X1.corr()
```

```
[27]: count = np.count_nonzero(cor > 0.9)
      print(count)
```

53

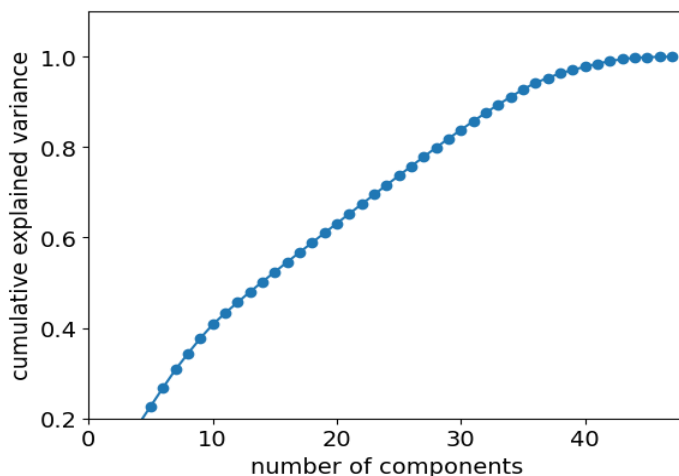
In code line [29], all 47 PCs are extracted after the features are normalized. The normalization or standardization is important as it then assigns “equal weights” to all features, so they are on the same scale. This is particularly important in feature space where some features with large measures may swamp those with smaller measures and produce distorted combined features.

```
[29]: from sklearn.decomposition import PCA
      pca = PCA(n_components = 47)
      pc = pca.fit_transform(X1)
      print('explained variance ratio: %s' %pca.explained_variance_ratio_)
      # % is used for string formatting placed in front of following numbers

explained variance ratio: [0.05183334 0.04664873 0.04402974 0.04330785 0.04119352 0.04107028
 0.03985298 0.03542389 0.033818 0.03049519 0.02525618 0.02363967
 0.02324064 0.02235872 0.02181575 0.02179855 0.02147621 0.02141851
 0.02131531 0.02129143 0.0212768 0.02123959 0.02113885 0.021079
 0.02092839 0.02087299 0.02062019 0.02056914 0.01983438 0.01934671
 0.01905125 0.01866613 0.01784745 0.01716822 0.01650159 0.01479251
 0.01069929 0.01028556 0.00737799 0.00710649 0.00668856 0.00589261
 0.0042024 0.00306136 0.00134249 0.00068641 0.00043914]
```

The ‘pca.explained\_variance\_ratio’ is the ratio of each eigenvalue to the sum of all eigenvalues which represents the total sample variances of all the feature variables. The individual ratios add up to one. The cumulative contribution of the explained variances ratios is shown in [31].

```
[31]: plt.plot(range(1,48),np.cumsum(pca.explained_variance_ratio_))
plt.scatter(range(1,48),np.cumsum(pca.explained_variance_ratio_))
plt.xlim(0,48)
plt.ylim(0.2,1.1)
plt.xlabel("number of components")
plt.ylabel("cumulative explained variance")
```



The PCs are used to form the new 47 features in pc\_X. See code line [32].

```
[32]: pc_X = pd.DataFrame(pc,columns = \
    ['pc_1','pc_2','pc_3','pc_4','pc_5','pc_6','pc_7','pc_8',\
    'pc_9','pc_10','pc_11','pc_12','pc_13','pc_14','pc_15',\
    'pc_16','pc_17','pc_18','pc_19','pc_20','pc_21','pc_22',\
    'pc_23','pc_24','pc_25','pc_26','pc_27','pc_28','pc_29',\
    'pc_30','pc_31','pc_32','pc_33','pc_34','pc_35','pc_36',\
    'pc_37','pc_38','pc_39','pc_40','pc_41','pc_42','pc_43',\
    'pc_44','pc_45','pc_46','pc_47'])
```

We can experiment with a smaller number of PCs and check their performances in the predictions. Here we utilize all PCs to perform the same logistic regression and predictions. One problem with regression using the principal components is that it is now difficult to interpret the impact of each PC which is a linear combination of many accounting ratios.

Then accuracy measurement of the test cases now using features pc\_X is performed in code lines [35], [36].

```
[35]: from sklearn import metrics
X_train, X_test, y_train, y_test = train_test_split(pc_X, y, \
                                                    test_size=0.3, random_state=0)
logreg = LogisticRegression()
lresult=logreg.fit(X_train, y_train)
```

```
[36]: y_pred1 = logreg.predict(X_test)
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:      0.54
```

There is an improvement of about 1% relative to the 53% in the feature space before PCA transformations.

The confusion matrix and classification reports are shown via code lines [37], [38].

```
[37]: from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test, y_pred1)
print(confusion_matrix)
```

```
[[2647 2574]
 [2268 3034]]
```

```
[38]: from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred1, digits=4))
```

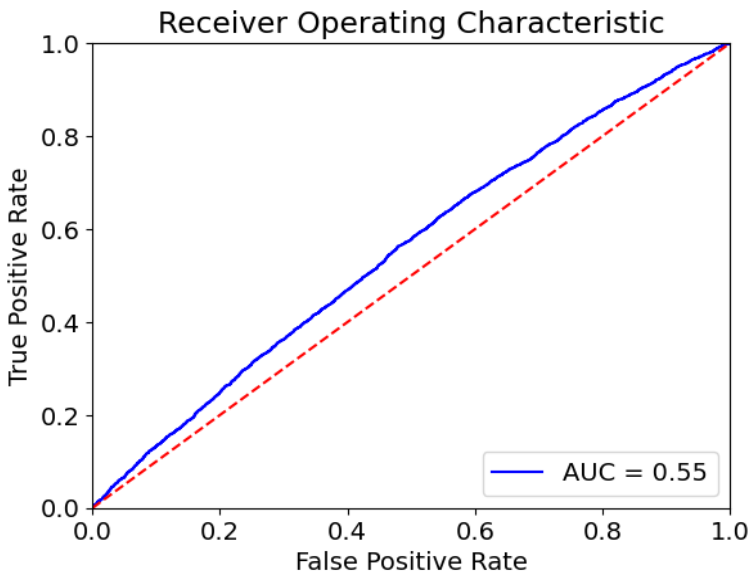
	precision	recall	f1-score	support
0	0.5386	0.5070	0.5223	5221
1	0.5410	0.5722	0.5562	5302
accuracy			0.5399	10523
macro avg	0.5398	0.5396	0.5392	10523
weighted avg	0.5398	0.5399	0.5394	10523

The area under ROC is now 55.34%, showing a marginal improvement from ROC in the original feature space. See code lines [40], [41].

```
[40]: ### Instead of above first two commands, we can use
from sklearn.metrics import roc_auc_score
logit_roc_auc = roc_auc_score(y_test, logreg.predict_proba(X_test)[:,:1])
print(logit_roc_auc)
### But we still require the 3rd command metrics.auc(fpr, tpr) to draw the ROC

0.5534208432402845
```

```
[41]: import matplotlib.pyplot as plt
plt.rc("font", size=14)
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc_logreg)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1], 'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



It is seen that PCA transformed features in this case (eliminating multi-collinearity) provide marginally better prediction results. There is no guarantee that PCA transformation will produce significantly better results in prediction even when multi-collinearity is reduced or eliminated. However, it is correct to say that reducing multi-collinearity (with or without PCA) generally should improve statistical tests and interpretations of significance of features.

### 3.6 Concluding Thoughts

In this chapter we see that machine learning method can be deployed to predict if a company's next year (period) earning will rise or not. The recall is above 60% while the precision is about 53%. These results can certainly be improved

with more thorough selection and pre-processing of the company accounting ratios as the features.

We also demonstrate how to use standardization on the features and then construct principal component variables that are not correlated. The latter may be important if the features are highly correlated (high multi-collinearity) and dampen the predictive accuracies.

The logistic regression is employed and is seen as an efficient tool to predict binary classifications. The logistic regression (together with regularization) could be used as a basic benchmark when comparing more sophisticated machine learning methods in the later chapters.

Perhaps more important than just statistical performance, the outputs such as in the confusion matrix could be used for analysing economic consequences. Conditional probabilities based on numbers in the confusion matrix can be used as a basic measure of expected outcomes. For example, 2697 predicted cases of  $\text{NPM\_FWD\_CHG}' > 0$  (label “1” or negative cases) when in fact label is “0” or no NPM increase means that conditional probability of wrong prediction based on label being “0” is  $2697/(2524+2697) = 51.65\%$ . This high conditional probability may imply trading losses if stock is purchased based on predicted next year  $\text{NPM} > 0$ . It is also seen that sometimes recall is an economically more significant measure in a bank loan context, while at other times, precision is more important in sales targeting.

### 3.7 Other References

[https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LogisticRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)

<https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8>

Chen Xi, Yang Ha Cho, Yiwei Dou, and Baruch Lev (2022), “Predicting Future Earnings Changes Using Machine Learning and Detailed Financial Data,” *Journal of Accounting Research* Vol 60, No 2.

Francis Sahngun Nahm (2022), “Receiver operating characteristic curve: overview and practical use for clinicians”, *Korean J Anesthesiol*, 75(1), pp. 25-36.

Kian Guan Lim, “Financial Valuation and Econometrics”, 2<sup>nd</sup> edition, chapter 24, World Scientific 2015.