CHAPTER 8

Model Evaluation

In this chapter, the most commonly used methods for testing the quality of a data science model will be formally introduced. Throughout this book, various *validation* techniques have been used to split the available data into a training set and a testing set. In the implementation sections, different types of performance operators in conjunction with validation have been used without an in detail explanation of how these operators really function. Several ways in which predictive data science models are evaluated for their performance will now be discussed.

There are a few main tools that are available to test a classification model's quality: *confusion matrices (or truth tables), lift charts, ROC (receiver operator characteristic) curves, area under the curve (AUC)*. How these tools are constructed will be defined in detail and how to implement performance evaluations will be described. To evaluate a numeric prediction from a regression model, there are many conventional statistical tests that may be applied (Black, 2008) a few of which were discussed in Chapter 5, Regression Methods.

DIRECT MARKETING

Direct marketing (DM) companies, which send out postal mail (or in the days before do-not-call lists, they called prospects) were one of the early pioneers in applying data science techniques (Berry, 1999). A key performance indicator for their marketing activities is of course the improvement in their bottom line as a result of their utilization of predictive models.

Assume that a typical average response rate for a direct mail campaign is 10%. Further assume that: cost per mail sent = $1 and potential revenue per response = $20. If they have 10,000 people to send out their mailers to, then they can expect to receive potential revenues of $10{,}000 \times 10\% \times \$20 = \$20{,}000$, which would yield a net return of $10,000. Typically, the mailers are sent out in batches to spread costs over a period of time. Further assume that these are sent out in batches of 1000. The first question someone would ask is how to divide the list of names into these batches. If the average expectation of return is 10%, then would it not make a lot of sense to just send one batch of mails to those prospects that make up this 10% and be done with the campaign?

(*Continued*)

Data Science. DOI: https://doi.org/10.1016/B978-0-12-814761-0.00008-3

(Continued)

Clearly this would save a lot of time and money and the net return would jump to $19,000!

Can all of these 10 percenters be identified? While this is clearly unrealistic, classification techniques can be used to rank or score prospects by the likelihood that they would respond to the mailers. Predictive analytics is after all about converting future uncertainties into usable probabilities (Taylor, 2011). Then a predictive method can be used to order these probabilities and send out the mailers to only those who score above a particular threshold (say 85% chance of response).

Finally, some techniques may be better suited to this problem than others. How can the different available methods be compared based on their performance? Will logistic regression capture these top 10 percenters better than support vector machines? What are the different metrics that can be used to select the best performing methods? These are some of the things that will be discussed in this chapter in more detail.

8.1 CONFUSION MATRIX

Classification performance is best described by an aptly named tool called the *confusion matrix* or truth table. Understanding the confusion matrix requires becoming familiar with several definitions. But before introducing the definitions, a basic confusion matrix for a binary or binomial classification must first be looked at where there can be two classes (say, Y or N). The accuracy of classification of a specific example can be viewed in one of four possible ways:

- The predicted class is Y, and the actual class is also Y → this is a True Positive or TP
- The predicted class is Y, and the actual class is N → this is a False Positive or FP
- The predicted class is N, and the actual class is Y → this is a False Negative or FN
- The predicted class is N, and the actual class is also N → this is a True Negative or TN

A basic confusion matrix is traditionally arranged as a 2×2 matrix as shown in Table 8.1. The predicted classes are arranged horizontally in rows and the

Table 8.1 Confusion Matrix

		Actual Class (Observation)	
		Y	**N**
Predicted class (expectation)	Y	*TP* correct result	*FP* unexpected result
	N	*FN* missing result	*TN* correct absence of result

TP, *true positive;* FP, *false positive;* FN, *false negative;* TN, *true negative.*

actual classes are arranged vertically in columns, although sometimes this order is reversed (Kohavi & Provost, 1998). A quick way to examine this matrix or a truth table as it is also called is to scan the diagonal from top left to bottom right. An ideal classification performance would only have entries along this main diagonal and the off-diagonal elements would be zero.

These four cases will now be used to introduce several commonly used terms for understanding and explaining classification performance. As mentioned earlier, a perfect classifier will have no entries for FP and FN (i.e., the number of FP = number of FN = 0).

1. ***Sensitivity*** is the ability of a classifier to select all the cases that *need* to be selected. A perfect classifier will select all the actual Y's and will not miss any actual Y's. In other words it will have no FNs. In reality, any classifier will miss some true Y's, and thus, have some FNs. Sensitivity is expressed as a ratio (or percentage) calculated as follows: TP/(TP + FN). However, sensitivity alone is not sufficient to evaluate a classifier. In situations such as credit card fraud, where rates are typically around 0.1%, an ordinary classifier may be able to show sensitivity of 99.9% by picking nearly all the cases as legitimate transactions or TP. The ability to detect illegitimate or fraudulent transactions, the TNs, is also needed. This is where the next measure, specificity, which ignores TPs, comes in.
2. ***Specificity*** is the ability of a classifier to reject all the cases that *need* to be rejected. A perfect classifier will reject all the actual N's and will not deliver any unexpected results. In other words, it will have no FPs. In reality, any classifier will select some cases that need to be rejected, and thus, have some FPs. Specificity is expressed as a ratio (or percentage) calculated as: TN/(TN + FP).
3. ***Relevance*** is a term that is easy to understand in a document search and retrieval scenario. Suppose a search is run for a specific term and that search returns 100 documents. Of these, let us say only 70 were useful because they were *relevant* to the search. Furthermore, the search actually missed out on an additional 40 documents that could actually have been useful. With this context, additional terms can be defined.
4. ***Precision*** is defined as the proportion of cases found that were actually relevant. From the example, this number was 70, and thus, the precision is 70/100 or 70%. The 70 documents were TP, whereas the remaining 30 were FP. Therefore, precision is TP/(TP + FP).
5. ***Recall*** is defined as the proportion of the relevant cases that were actually found among all the relevant cases. Again, with the example, only 70 of the total 110 (70 found + 40 missed) relevant cases were

Table 8.2 Evaluation Measures

Term	Definition	Calculation
Sensitivity	Ability to select what needs to be selected	TP/(TP + FN)
Specificity	Ability to reject what needs to be rejected	TN/(TN + FP)
Precision	Proportion of cases found that were relevant	TP/(TP + FP)
Recall	Proportion of all relevant cases that were found	TP/(TP + FN)
Accuracy	Aggregate measure of classifier performance	(TP + TN)/ (TP + TN + FP + FN)

TP, *true positive;* FP, *false positive;* FN, *false negative;* TN, *true negative.*

actually found, thus, giving a recall of 70/110 = 63.63%. It is evident that recall is the same as sensitivity, because recall is also given by TP/(TP + FN).

6. ***Accuracy*** is defined as the ability of the classifier to select all cases that need to be selected and reject all cases that need to be rejected. For a classifier with 100% accuracy, this would imply that FN = FP = 0. Note that in the document search example, the TN has not been indicated, as this could be really large. Accuracy is given by (TP + TN)/ (TP + FP + TN + FN). Finally, *error* is simply the complement of accuracy, measured by (1 − accuracy).

Table 8.2 summarizes all the major definitions. Fortunately, the analyst does not need to memorize these equations because their calculations are always automated in any tool of choice. However, it is important to have a good fundamental understanding of these terms.

8.2 ROC AND AUC

Measures like accuracy or precision are essentially aggregates by nature, in the sense that they provide the average performance of the classifier on the dataset. A classifier can have a high accuracy on a dataset but have poor class recall and precision. Clearly, a model to detect fraud is no good if its ability to detect TP for the fraud = yes class (and thereby its class recall) is low. It is, therefore, quite useful to look at measures that compare different metrics to see if there is a situation for a trade-off: for example, can a little overall accuracy be sacrificed to gain a lot more improvement in class recall? One can examine a model's rate of detecting TPs and contrast it with its ability to detect FPs. The receiver operator characteristic (ROC) curves meet this need and were originally developed

in the field of signal detection (Green, 1966). A ROC curve is created by plotting the fraction of TPs (TP rate) versus the fraction of FPs (FP rate). When a table of such values is generated, the FP rate can be plotted on the horizontal axis and the TP rate (same as sensitivity or recall) on the vertical axis. The FP can also be expressed as (1 − specificity) or TN rate.

Consider a classifier that could predict if a website visitor is likely to click on a banner ad: the model would be most likely built using historic click-through rates based on pages visited, time spent on certain pages, and other characteristics of site visitors. In order to evaluate the performance of this model on test data, a table such as the one shown in Table 8.3 can be generated.

The first column "Actual Class" consists of the actual class for a particular example (in this case a website visitor, who has clicked on the banner ad). The next column, "Predicted Class" is the model prediction and the third column, "Confidence of response" is the confidence of this prediction. In order to create a ROC chart, the predicted data will need to be sorted in decreasing order of confidence level, which has been done in this case. By comparing columns Actual class and Predicted class, the type of prediction can be identified: for instance, spreadsheet rows 2 through 5 are all TPs and row 6 is the first instance of a FP. As observed in columns "Number of TP" and "Number of FP," one can keep a running count of the TPs and FPs and also calculate the fraction of TPs and FPs, which are shown in columns "Fraction of TP" and "Fraction of FP."

Observing the "Number of TP" and "Number of FP" columns, it is evident that the model has discovered a total of 6 TPs and 4 FPs (the remaining 10 examples are all TNs). It can also be seen that the model has identified nearly 67% of all the TPs before it fails and hits its first FP (row 6 above). Finally, all TPs have been identified (when Fraction of TP = 1) before the next FP was run into. If Fraction of FP (FP rate) versus Fraction of TP (TP rate) were now to be plotted, then a ROC chart similar to the one shown in Fig. 8.1 would be seen. Clearly an ideal classifier would have an accuracy of 100% (and thus, would have identified 100% of all TPs). Thus, the ROC for an ideal classifier would look like the dashed line shown in Fig. 8.1. Finally, an ordinary or random classifier (which has only a 50% accuracy) would possibly be able to find one FP for every TP, and thus, look like the 45-degree line shown.

As the number of test examples becomes larger, the ROC curve will become smoother: the random classifier will simply look like a straight line drawn between the points (0,0) and (1,1)—the stair steps become extremely small. The area under this random classifier's ROC curve is basically the area of a

Table 8.3 Classifier Performance Data Needed for Building a ROC Curve

Actual Class	Predicted Class	Confidence of "Response"	Type?	Number of TP	Number of FP	Fraction of FP	Fraction of TP
Response	Response	0.902	TP	1	0	0	0.167
Response	Response	0.896	TP	2	0	0	0.333
Response	Response	0.834	TP	3	0	0	0.500
Response	Response	0.741	TP	4	0	0	0.667
No response	Response	0.686	FP	4	1	0.25	0.667
Response	Response	0.616	TP	5	1	0.25	0.833
Response	Response	0.609	TP	6	1	0.25	1
No response	Response	0.576	FP	6	2	0.5	1
No response	Response	0.542	FP	6	3	0.75	1
No response	Response	0.530	FP	6	4	1	1
No response	No response	0.440	TN	6	4	1	1
No response	No response	0.428	TN	6	4	1	1
No response	No response	0.393	TN	6	4	1	1
No response	No response	0.313	TN	6	4	1	1
No response	No response	0.298	TN	6	4	1	1
No response	No response	0.260	TN	6	4	1	1
No response	No response	0.248	TN	6	4	1	1
No response	No response	0.247	TN	6	4	1	1
No response	No response	0.241	TN	6	4	1	1
No response	No response	0.116	TN	6	4	1	1

ROC, *receiver operator characteristic;* TP, *true positive;* FP, *false positive;* TN, *true negative.*

right triangle (with side 1 and height 1), which is 0.5. This quantity is termed Area Under the Curve or AUC. AUC for the ideal classifier is 1.0. Thus, the performance of a classifier can also be quantified by its AUC: obviously any AUC higher than 0.5 is better than random and the closer it is to 1.0, the better the performance. A common rule of thumb is to select those classifiers that not only have a ROC curve that is closest to ideal, but also an AUC higher than 0.8. Typical uses for AUC and ROC curves are to compare the performance of different classification algorithms for the same dataset.

8.3 LIFT CURVES

Lift curves or lift charts were first deployed in direct marketing where the problem was to identify if a particular prospect was worth calling or sending an advertisement by mail. It was mentioned in the use case at the beginning

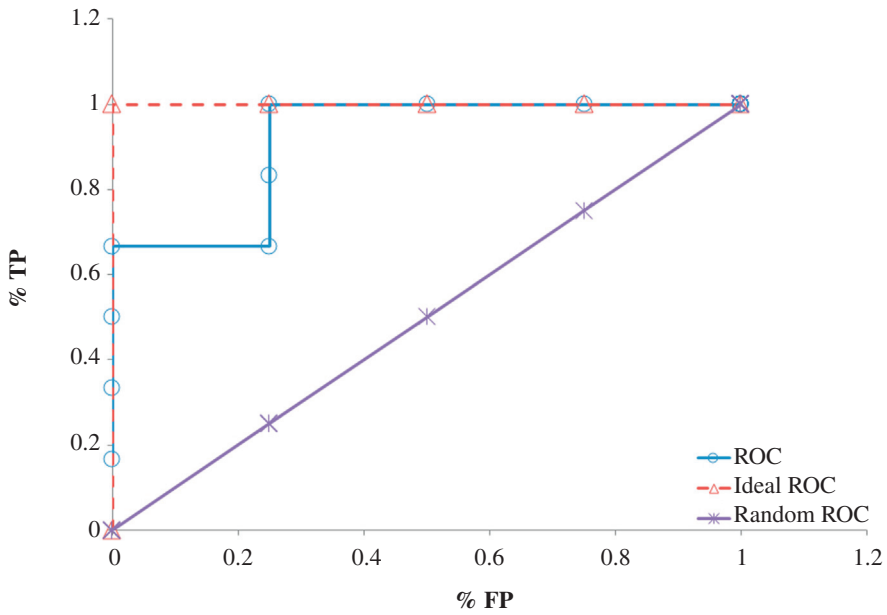

FIGURE 8.1
Comparing ROC curve for the example shown in Table 8.3 to random and ideal classifiers. *ROC*, receiver operator characteristic.

of this chapter that with a predictive model, one can *score* a list of prospects by their propensity to respond to an ad campaign. When the prospects are sorted by this score (by the decreasing order of their propensity to respond), one now ends up with a mechanism to systematically select the most valuable prospects right at the beginning, and thus, maximize their return. Thus, rather than mailing out the ads to a random group of prospects, the ads can now be sent to the first batch of "most likely responders," followed by the next batch and so on.

Without classification, the "most likely responders" are distributed randomly throughout the dataset. Suppose there is a dataset of 200 prospects and it contains a total of 40 responders or TPs. If the dataset is broken up into, say, 10 equal sized batches (called deciles), the likelihood of finding TPs in each batch is also 20%, that is, four samples in each decile will be TPs. However, when a predictive model is used to classify the prospects, a good model will tend to pull these "most likely responders" into the top few deciles. Thus, in this simple example, it might be found that the first two deciles will have all 40 TPs and the remaining eight deciles have none.

Lift charts were developed to demonstrate this in a graphical way (Rud, 2000). The focus is again on the TPs and, thus, it can be argued that they indicate the sensitivity of the model unlike ROC curves, which can show the relation between sensitivity and specificity.

The motivation for building lift charts was to depict how much better the classifier performs compared to *randomly selecting x% of the data (for prospects to call) which would yield x% targets (to call or not)*. Lift is the improvement over this random selection that a predictive model can potentially yield because of its scoring or ranking ability. For example, in the data from Table 8.3, there are a total of 6 TPs out of 20 test cases. If one were to take the unscored data and randomly select 25% of the examples, it would be expected that 25% of them were TPs (or 25% of 6 = 1.5). However, scoring and reordering the dataset by confidence will improve this. As can be seen in Table 8.4, the first 25% or quartile of scored (reordered) data now contains four TPs. This translates to a lift of 4/1.5 = 2.67. Similarly, the second quartile of the unscored data can be expected to contain 50% (or three) of the TPs. As seen in Table 8.4, the scored 50% data contains all six TPs, giving a lift of 6/3 = 2.00.

The steps to build lift charts are:

1. Generate scores for all the data points (prospects) in the test set using the trained model.
2. Rank the prospects by decreasing score or confidence of response.
3. Count the TPs in the first 25% (quartile) of the dataset, and then the first 50% (add the next quartile) and so on; see columns Cumulative TP and Quartile in Table 8.4.
4. *Gain* at a given quartile level is the ratio of the cumulative number of TPs in that quartile to the total number of TPs in the entire dataset (six in the example). The 1st quartile gain is, therefore, 4/6 or 67%, the 2nd quartile gain is 6/6 or 100%, and so on.
5. *Lift* is the ratio of gain to the random expectation at a given quartile level. Remember that random expectation at the *x*th quartile is *x*%. In the example, the random expectation is to find 25% of 6 = 1.5 TPs in the 1st quartile, 50% or 3 TPs in the 2nd quartile, and so on. The corresponding 1st quartile lift is, therefore, 4/1.5 = 2.667, the 2nd quartile lift is 6/3 = 2.00, and so on.

The corresponding curves for the simple example are shown in Fig. 8.2. Typically lift charts are created on deciles not quartiles. Quartiles were chosen here because they helped to illustrate the concept using the small 20-sample test dataset. However, the logic remains the same for deciles or any other groupings as well.

Table 8.4 Scoring Predictions and Sorting by Confidences Is the Basis for Generating Lift Curves

Actual Class	Predicted Class	Confidence of "Response"	Type?	Cumulative TP	Cumulative FP	Quartile	Gain	Lift
Response	Response	0.902	TP	1	0	1st	67%	2.666667
Response	Response	0.896	TP	2	0	1st		
Response	Response	0.834	TP	3	0	1st		
Response	Response	0.741	TP	4	0	1st		
No response	Response	0.686	FP	4	1	1st		
Response	Response	0.616	TP	5	1	2nd	100%	2
Response	Response	0.609	TP	6	1	2nd		
No response	Response	0.576	FP	6	2	2nd		
No response	Response	0.542	FP	6	3	2nd		
No response	Response	0.530	FP	6	4	2nd		
No response	No response	0.440	TN	6	4	3rd	100%	1.333333
No response	No response	0.428	TN	6	4	3rd		
No response	No response	0.393	TN	6	4	3rd		
No response	No response	0.313	TN	6	4	3rd		
No response	No response	0.298	TN	6	4	3rd		
No response	No response	0.260	TN	6	4	4th	100%	1
No response	No response	0.248	TN	6	4	4th		
No response	No response	0.247	TN	6	4	4th		
No response	No response	0.241	TN	6	4	4th		
No response	No response	0.116	TN	6	4	4th		

TP, *true positive;* FP, *false positive;* TN, *true negative.*

8.4 HOW TO IMPLEMENT

A built-in dataset in RapidMiner will be used to demonstrate how all the three classification performances (confusion matrix, ROC/AUC, and lift/gain charts) are evaluated. The process shown in Fig. 8.3 uses the *Generate Direct Mailing Data* operator to create a 10,000 record dataset. The objective of the modeling (Naïve Bayes used here) is to predict whether a person is likely to respond to a direct mailing campaign or not based on demographic attributes (age, lifestyle, earnings, type of car, family status, and sports affinity).

Step 1: Data Preparation

Create a dataset with 10,000 examples using the *Generate Direct Mailing Data* operator by setting a local random seed (default = 1992) to ensure

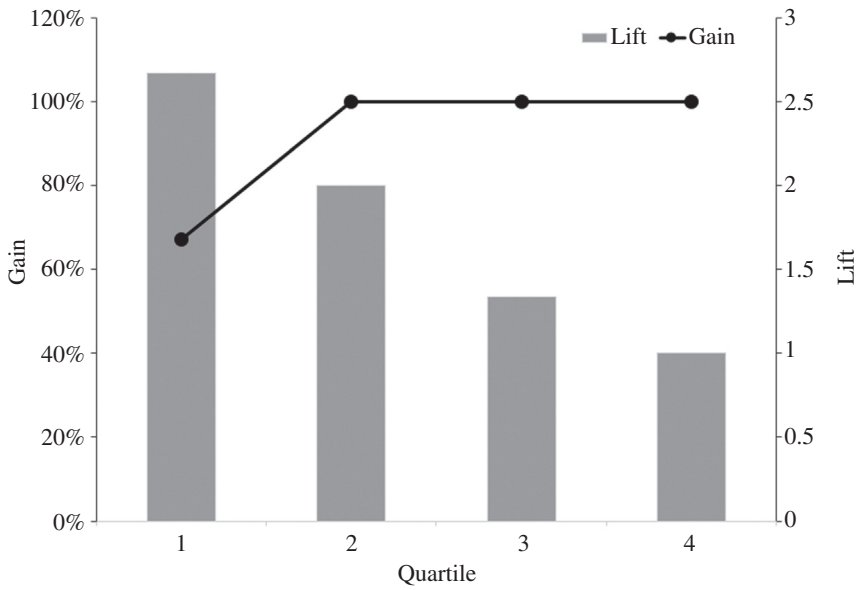

FIGURE 8.2
Lift and gain curves.

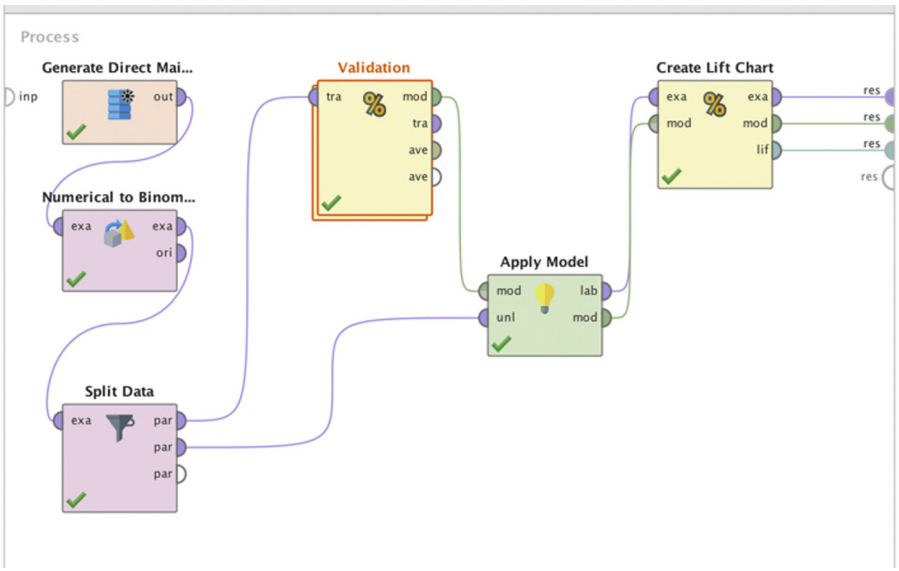

FIGURE 8.3
Process setup to demonstrate typical classification performance metrics.

repeatability. Convert the label attribute from polynomial (nominal) to binominal using the appropriate operator as shown. This enables one to select specific binominal classification performance measures.

Split data into two partitions: an 80% partition (8000 examples) for model building and validation and a 20% partition for testing. An important point to note is that data partitioning is not an exact science and this ratio can change depending on the data.

Connect the 80% output (upper output port) from the *Split Data* operator to the *Split Validation* operator. Select a relative split with a ratio of 0.7 (70% for training) and shuffled sampling.

Step 2: Modeling Operator and Parameters

Insert the *naïve Bayes* operator in the Training panel of the *Split Validation* operator and the usual *Apply Model* operator in the Testing panel. Add a *Performance (Binomial Classification)* operator. Select the following options in the performance operator: accuracy, FP, FN, TP, TN, sensitivity, specificity, and AUC.

Step 3: Evaluation

Add another *Apply Model* operator outside the *Split Validation* operator and deliver the model to its mod input port while connecting the 2000 example data partition from Step 3 to the unl port. Add a *Create Lift Chart* operator with these options selected: target class = response, binning type = frequency, and number of bins = 10. Note the port connections as shown in Fig. 8.3.

Step 4: Execution and Interpretation

When the above process is run, the confusion matrix and ROC curve for the validation sample should be generated (30% of the original 80% = 2400 examples), whereas a lift curve should be generated for the test sample (2000 examples). There is no reason why one cannot add another *Performance (Binomial Classification)* operator for the test sample or create a lift chart for the validation examples. (The reader should try this as an exercise—how will the output from the *Create Lift Chart* operator be delivered when it is inserted inside the *Split Validation* operator?)

The confusion matrix shown in Fig. 8.4 is used to calculate several common metrics using the definitions from Table 8.1. Compare them with the RapidMiner outputs to verify understanding.

	true no response	true response	class precision
pred. no response	1231	146	89.40%
pred. response	394	629	61.49%
class recall	75.75%	81.16%	

FIGURE 8.4
Confusion matrix for validation set of direct marketing dataset.

$$TP = 629,\ \ TN = 1231,\ \ FP = 394,\ \ FN = 146$$

Term	Definition	Calculation
Sensitivity	TP/(TP + FN)	629/(629 + 146) = 81.16%
Specificity	TN/(TN + FP)	1231/(1231 + 394) = 75.75%
Precision	TP/(TP + FP)	629/(629 + 394) = 61.5%
Recall	TP/(TP + FN)	629/(629 + 146) = 81.16%
Accuracy	(TP + TN)/ (TP + TN + FP + FN)	(629 + 1231)/(629 + 1231 + 394 + 146) = 77.5%

Note that RapidMiner makes a distinction between the two classes while calculating precision and recall. For example, in order to calculate a class recall for "no response," the *positive* class becomes "no response" and the corresponding TP is 1231 and the corresponding FN is 394. Therefore, a class recall for "no response" is 1231/(1231 + 394) = 75.75%, whereas the calculation above assumed that "response" was the positive class. Class recall is an important metric to keep in mind when dealing with highly unbalanced data. Data are considered unbalanced if the proportion of the two classes is skewed. When models are trained on unbalanced data, the resulting class recalls also tend to be skewed. For example, in a dataset where there are only 2% responses, the resulting model can have a high recall for "no responses" but a very low class recall for "responses." This skew is not seen in the overall model accuracy and using this model on unseen data may result in severe misclassifications.

The solution to this problem is to either balance the training data so that one ends up with a more or less equal proportion of classes or to insert penalties or costs on misclassifications using the *Metacost* operator as discussed in Chapter 5, Regression Methods. Data balancing is explained in more detail in Chapter 13, Anomaly Detection.

The AUC is shown along with the ROC curve in Fig. 8.5. As mentioned earlier, AUC values close to 1 are indicative of a good model. The ROC captures the sorted confidences of a prediction. As long as the prediction is correct for the examples the curve takes one step up (increased TP). If the prediction is wrong the curve takes one step to the right (increased FP). RapidMiner can

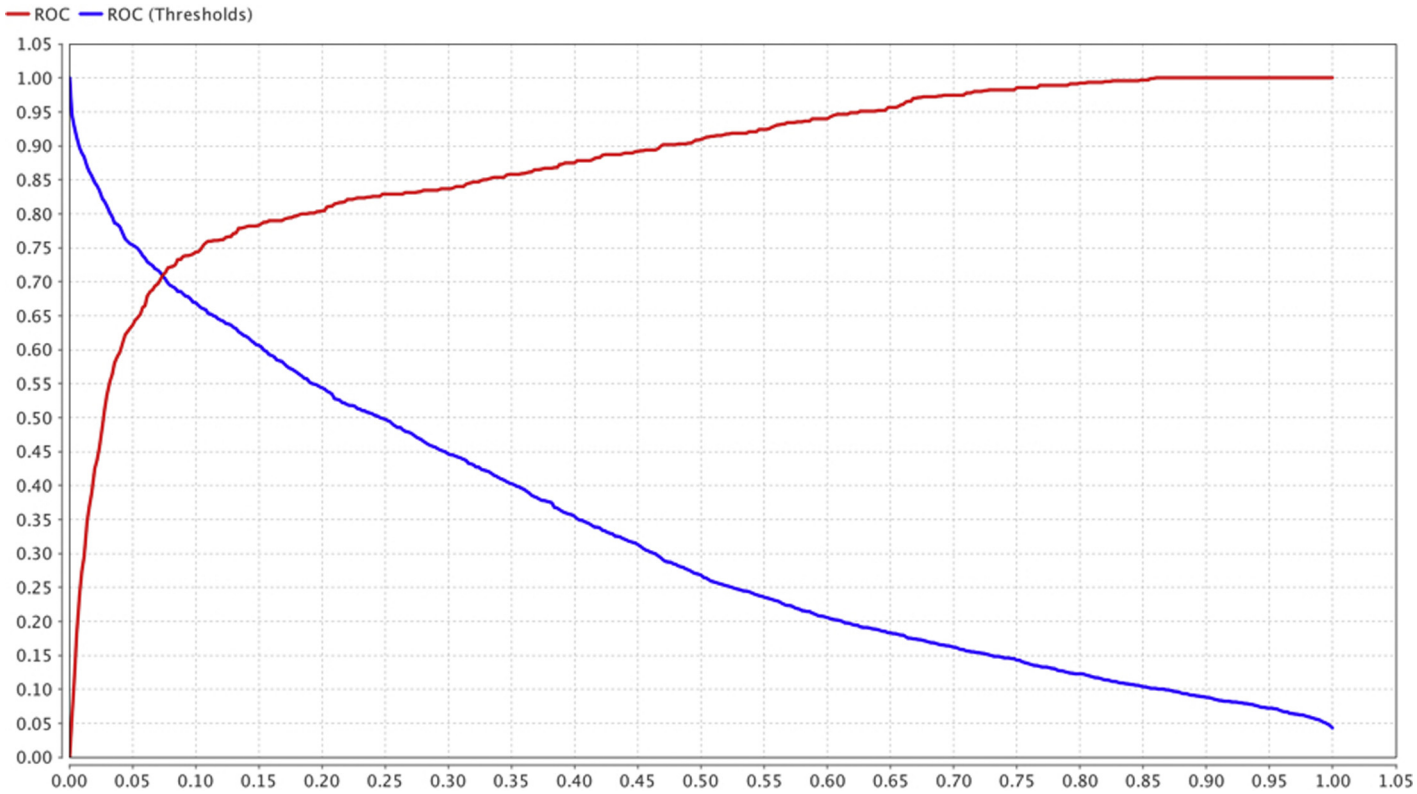

FIGURE 8.5

ROC curve and AUC. *ROC*, receiver operator characteristic; *AUC*, area under the curve.

show two additional AUCs called optimistic and pessimistic. The differences between the optimistic and pessimistic curves occur when there are examples with the same confidence, but the predictions are sometimes false and sometimes true. The optimistic curve shows the possibility that the correct predictions are chosen first so the curve goes steeper upwards. The pessimistic curve shows the possibility that the wrong predictions are chosen first so the curve increases more gradually.

Finally, the lift chart outputs do not directly indicate the lift values as has been demonstrated with the simple example earlier. In Step 5 of the process, 10 bins were selected for the chart and, thus, each bin will have 200 examples (a decile). Recall that to create a lift chart all the predictions will need to be sorted by the confidence of the positive class (response), which is shown in Fig. 8.6.

The first bar in the lift chart shown in Fig. 8.7 corresponds to the first bin of 200 examples after the sorting. The bar reveals that there are 181 TPs in this bin (as can be seen from the table in Fig. 8.6 that the very second example, Row No. 1973, is an FP). From the confusion matrix earlier, 629 TPs can be seen in this example set. A random classifier would have identified 10% of these or 62.9 TPs in the first 200 examples. Therefore, the lift for the first decile is $181/62.9 = 2.87$. Similarly the lift for the first two deciles is $(181 + 167)/(2 \times 62.9) = 2.76$ and so on. Also, the first decile contains $181/629 = 28.8\%$ of the TPs, the first two deciles contain $(181 + 167)/629 = 55.3\%$ of the TPs, and so on. This is shown in the cumulative (percent) gains curve on the right hand y-axis of the lift chart output.

As described earlier, a good classifier will accumulate all the TPs in the first few deciles and will have extremely few FPs at the top of the heap. This will result in a gain curve that quickly rises to the 100% level within the first few deciles.

8.5 CONCLUSION

This chapter covered the basic performance evaluation tools that are typically used in classification methods. Firstly the basic elements of a confusion matrix were described and then the concepts that are important to understanding it, such as sensitivity, specificity, and accuracy were explored in detail. The ROC curve was then described, which has its origins in signal detection theory and has now been adopted for data science, along with the equally useful aggregate metric of AUC. Finally, two useful tools were described that have their origins in direct marketing applications: lift and gain charts. How to build these curves in general and how they can be

Row No.	label	prediction(l...	confidence(no response)	confidence(response)	name	age	lifestyle	zip code
1	no response	no response	0.881	0.119	iHHbMWkc	27	cozily	70096
2	response	response	0.090	0.910	ZWkb86b8	69	active	96274
3	no response	response	0.389	0.611	C1JDpJQO	49	active	37767
4	no response	no response	0.681	0.319	FjYd8PWL	42	healthy	59235
5	no response	no response	0.911	0.089	y7rkBg7t	22	active	57999
6	response	response	0.125	0.875	sPJLeRoJ	68	cozily	66375
7	response	response	0.066	0.934	etE6NiMS	67	cozily	95286
8	no response	no response	0.515	0.485	goA20OUe	45	healthy	60108
9	no response	no response	0.861	0.139	WQkOyMLF	32	healthy	78263
10	response	response	0.248	0.752	A1KrhHiP	51	healthy	98457

FIGURE 8.6
Table of scored responses used to build the lift chart.

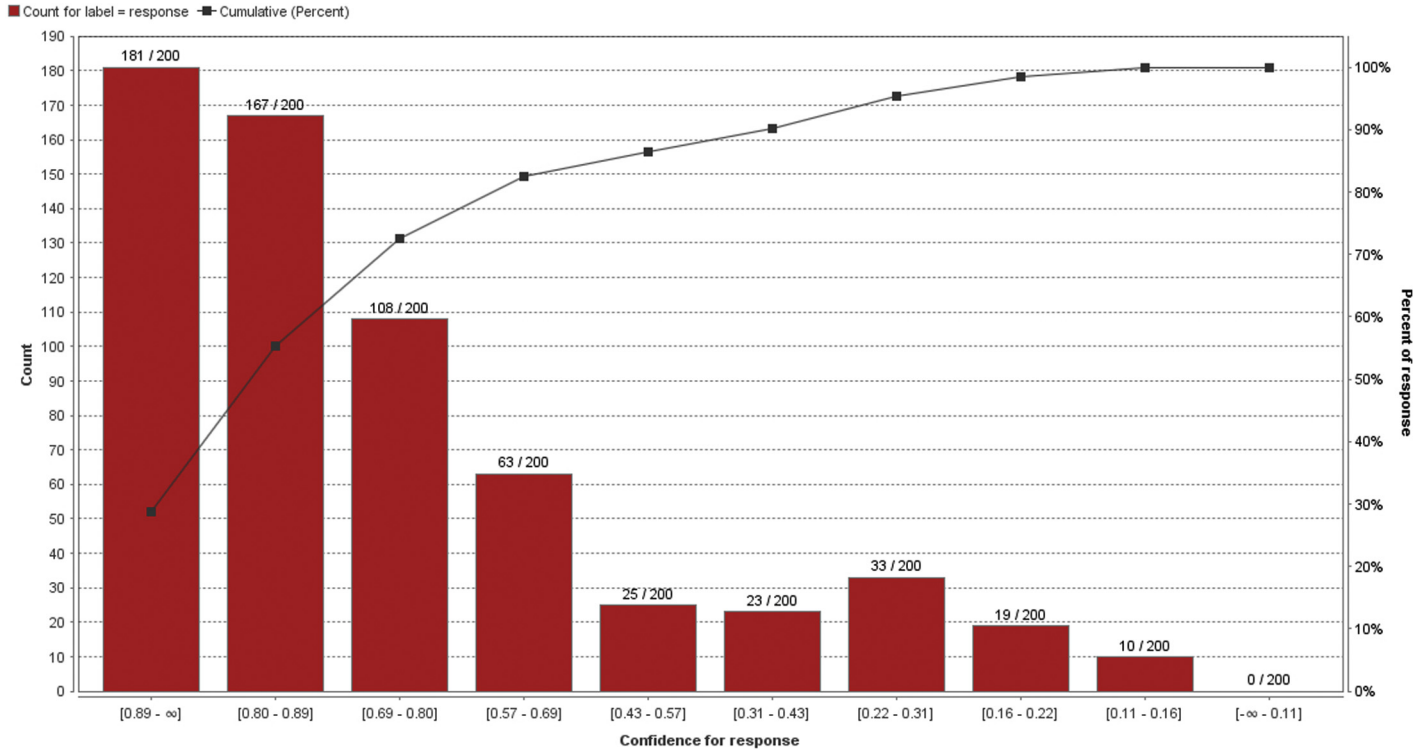

FIGURE 8.7
Lift chart generated.

constructed using RapidMiner was discussed. In summary, these tools are some of the most commonly used metrics for evaluating predictive models and developing skill and confidence in using these is a prerequisite to developing data science expertise.

One key to developing good predictive models is to know when to use which measures. As discussed earlier, relying on a single measure like accuracy can be misleading. For highly unbalanced datasets, rely on several measures such as class recall and precision in addition to accuracy. ROC curves are frequently used to compare several algorithms side by side. Additionally, just as there are an infinite number of triangular shapes that have the same area, AUC should not be used alone to judge a model—AUC and ROCs should be used in conjunction to rate a model's performance. Finally, lift and gain charts are most commonly used for scoring applications where the examples in a dataset need to be rank-ordered according to their propensity to belong to a particular category.

References

Berry, M. A. (1999). *Mastering data mining: The art and science of customer relationship management*. New York: John Wiley and Sons.

Black, K. (2008). *Business statistics for contemporary decision making*. New York: John Wiley and Sons.

Green, D. S. (1966). *Signal detection theory and psychophysics*. New York: John Wiley and Sons.

Kohavi, R., & Provost, F. (1998). Glossary of terms. *Machine Learning, 30*, 271–274.

Rud, O. (2000). *Data mining cookbook: Modeling data for marketing, risk and customer relationship management*. New York: John Wiley and Sons.

Taylor, J. (2011). *Decision management systems: A practical guide to using business rules and predictive analytics*. Boston, Massachusetts: IBM Press.