Define the problem.

It is important to have a lucid perception of the problem that we are trying to tackle, so the foremost step to form a solution is to draw a clear appreciation of the problem and decide upon an objective measure of success.

Data Collection

A machine learning solution is as good as the reliability of data. If possible special care is needed to ensure collection of accurate and consistent data. An insight into the data collection process is also useful while treated outliers and missing data.

Exploration & Visualization

- The number of records in the dataset
- The number of attributes (or features)
- For each attribute, what are the data types (nominal, ordinal, or continuous)
- For nominal attributes, what are the categories?
- For continuous attribute, the statistics and distribution
- For any attributes, the number of missing values
- For any attribute, the number of distinct values
- For labeled data, check that the classes are approximately balanced.
- For a given attribute, check that values are within the expected range.

Use various plots to discover relationships between labels and features. Most useful are:

- Correlation Plots.
- Box plots.
- Scatter Plots.

Outliers

In statistics, an outlier is defined as an observation which stands far away from the most of other observations. Outliers can bias the models disrupting its accuracy. Some models are more sensitive to outliers than others. Z-scores normalization depends on mean which can be strongly deviated by outliers. Therefore, one of the most important task in data analysis is to identify and (if is necessary) to remove the outliers.

Often an outlier is present due to the measurements error, or casual errors introduced while recording the observations like appending zeros. These extreme values may be valid so they must be treated according to the nature of their production. It is tempting to remove outliers. Don't do this without a very good reason. Models that ignore exceptional (and interesting) cases often perform poorly.

Outlier Detection:

- The box plot of two numeric variables to identify the outliers.
- Scatter plot of two numeric variables.
- Categories with extremely few instances may be treated as outliers.

Possible Outlier Treatment:

- Remove rows.

- Clip Values
- Log Transformation: This is a great way to include valid extreme values in your model.

Missing Values

Classification: The proportion of label class can be used to see if it has a particular bias towards any label class.

- Separate the dataset on basis of the attribute, missing vs non-missing.
- Observe the proportion of label class in each division of data using visualization capability in Azure ML.
- Use two population z-test to measure if the difference is significant, use online calculators.
  Sample vs Population Z test Calculator.
  Two samples z test calculator.

Regression: The mean of label value can be used to see if it has a particular bias towards any label distribution.

- Separate the dataset on basis of the attribute, missing vs non-missing.
- Observe the mean of label attribute in each division of data using visualization capability in Azure ML.
- Use two population z-test to measure if the difference is significant.
  T test for two means

If the difference is statistically significant, the missing value convey some information about the label class. So create an indicator variable, which will be used as a predictor by the ML algorithm. Alternately Pearson Co-relation and Chi Squared Test can be performed on missing indicator variable to decide if the indicator variable is useful.

If the difference is not significant and rows can be regarded as randomly missing with respect to label and any of the following treatments can be applied.

Missing Value Treatment:

Remove rows, Mean, Median, Mode, MICE.

- MICE can bias the data by overestimating covariance and correlation estimates in the data (if label class used in MICE).
- Apply all and check which improves Cross Validation accuracy.
- Use indicator columns if not randomly missing.
- Treat missing values after transformations which make the attribute distribution more normal.

Randomly Missing with respect to the value of attribute:

- Create an indicator column and check perform Mutual Information, Chi Squared and correlation tests to analyze this columns dependence on any other variables.

- If any significant dependence is observed than an appropriate substitution value can be devised rather than using the mean or median.
  E.g If body weight of a population are sampled and it turns out that most of the missing values belong to female gender. It will be appropriate to replace the missing values with mean weight of females rather than mean of whole sample.
- Be careful with these techniques, they may lead to bias.

Feature Selection

Filter-Based Feature Selection Module

The **Filter-Based Feature Selection** provides a variety of metrics for assessing the information value in each column. Two most commonly used are

- **Pearson Correlation (for numeric data)**

  Pearson's correlation statistics or Pearson's correlation coefficient is also known in statistical models as the **r** value. For any two variables, it returns a value that indicates the strength of the correlation.

  Pearson's correlation coefficient is computed by taking the covariance of two variables and dividing by the product of their standard deviations. The coefficient is not affected by changes of scale in the two variables.

- **Chi Squared (for categorial data)**

  The two-way chi-squared test is a statistical method that measures how close expected values are to actual results. The method assumes that variables are random and drawn from an adequate sample of independent variables. The resulting chi-squared statistic indicates how far results are from the expected (random) result.

Weights assigned by Regression Trained Model.

Linear regression model assigns weights to each feature which can be visualized from a trained model. Here the magnitude of the assigned weight is the measure of the how much a label is swayed by a unit change in that feature's value. Thus this can be used as measure of that features importance. Pruning out the features that have low magnitudes of weights will result in simpler models that avoid over-fitting and have greater predictive capabilities.

Permutation Feature Importance

You can use the Permutation Feature Importance module to compute a set of feature importance scores for your dataset, based on how much the performance of a model based on the dataset changes when the feature values are randomly shuffled.

Pruning Features

It is a good idea to remove the features that do score well in above tests. Remove those features and if this removal does not significantly lower cross validation results, permanently remove these features so that you have a simpler model.

Feature Engineering:

- Transformation which makes the variable closer to normal improves regression because regression assumes normal distribution.
- Transformation like log will make the right skewed distribution normal and also include valid outliers.
- If the attribute contains 0 values ln +1 transformation must be used.
- Apply math operation module can be used to apply such transformations.
- Interaction terms. Terms where two attributes are multiplied or divided can be introduced and will incorporate polynomial relationships in the linear regression models.
- Create new features based on domain knowledge.

Apply transformations which:

- Increase correlation with the label.
- Improve attribute distribution.
- Improve the Model evaluation scores.

Model Selection

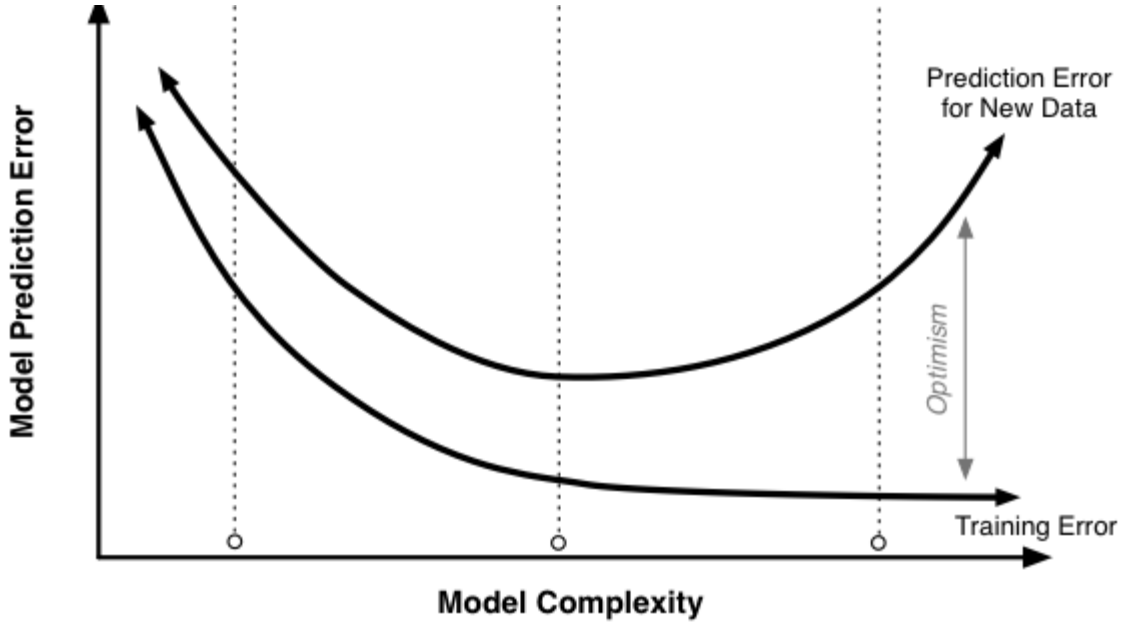
There are no fixed rules for model selection. When in confusion it is a good idea to train all candidate models and use the model evaluation module in Azure ML to select the model which performs better. When multiple models have similar performance, opt for the simpler model.

- Ensemble Models generally perform better than individual models but they take longer to train.
- Linear Regression is an elegant model which offers actionable insights about relationship between feature and labels, so it should be used if the performance is not significantly improved by alternatives.
- Decision Tree based Models are better at handling datasets which have complex non-linear relationships.

Tuning Model Parameters

Tune model hyper parameters module was used to get a rough estimate of parameter values and then manual selection was made by monitoring model performance.

While finding the optimal model parameters we must be very careful and strike the balance between model complexity and predictive power. Increasing the number of decision trees and other features that enhance the complexity of model may result in overfitting the test data and producing models that are biased by minor patterns in test data and perform poorly when exposed to new data.



As shown in the above figure, error in training data will continue to decrease with training data but after a point it will increase with model complexity. We need to find the optimum value that corresponds to the dip in curve of Prediction error. For regression Models regularization weights can be varied to find this optimum point.

Validation

Azure ML provides a module for model evaluation. Data must be divided into Test and Training sets, where the model is trained on the training set and predicts the label for the test set which it has never scene. This is important because a model may perform very well on data it is trained on but show poor performance with the new data. This happens because of overfitting where the model also captures low level noise along with salient trends. To avoid over fitting and build models with enhanced predictive capabilities we evaluate the model on Test data.
A more robust approach for evaluation is N-fold cross validation. This avoids the bias that may be introduced by statistical difference that may arise between test and train data due to your sampling technique.

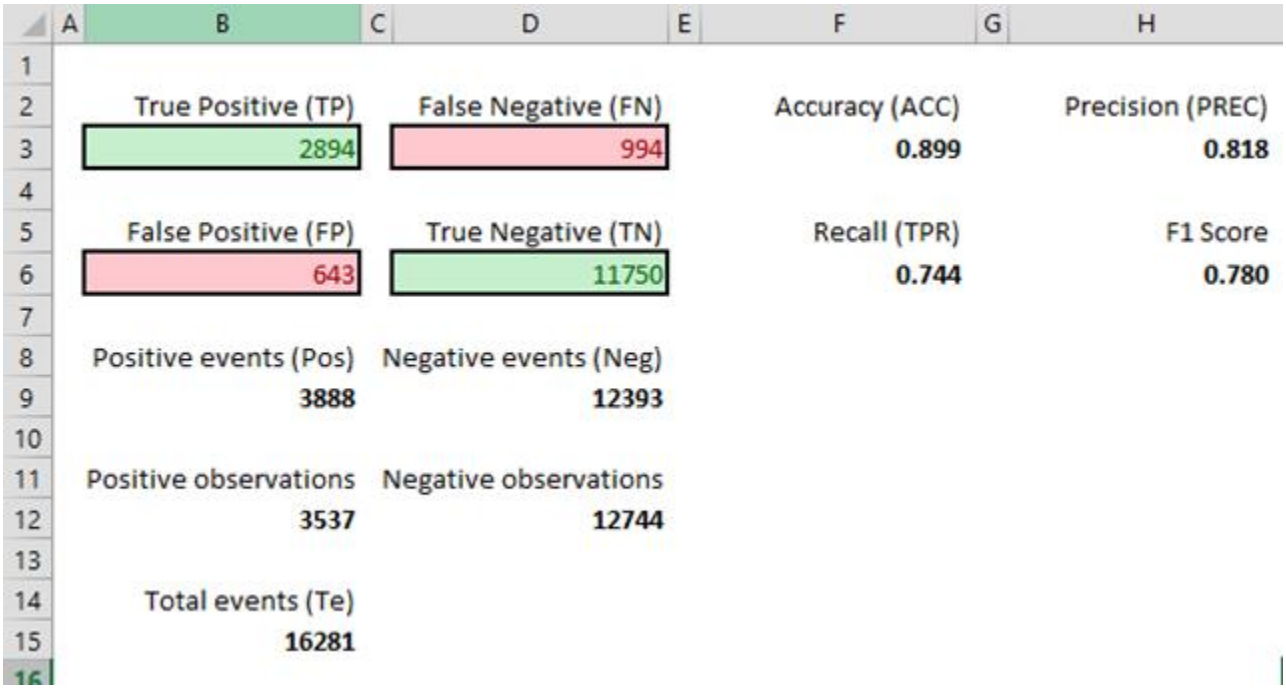
Accessing performance metrics in Azure ML for Classification

To view the performance metrics, right-click on the output port of the Evaluate Model item and select Visualize. After you click the Visualize button, you'll initially see the ROC plot. Scroll down to see the performance metrics:

True Positive	False Negative	Accuracy	Precision	Threshold	AUC
2894	994	0.899	0.818	0.5	0.954

False Positive	True Negative	Recall	F1 Score
643	11750	0.744	0.780

Above, all measures except AUC can be calculated based on the left-most four numbers. I created a spreadsheet which allows you to experiment with these numbers. You can also use the spreadsheet to review the formulas for each of the metrics. Here's what the spreadsheet looks like:



	B	D	F	H
2	True Positive (TP)	False Negative (FN)	Accuracy (ACC)	Precision (PREC)
3	2894	994	**0.899**	**0.818**
5	False Positive (FP)	True Negative (TN)	Recall (TPR)	F1 Score
6	643	11750	**0.744**	**0.780**
8	Positive events (Pos)	Negative events (Neg)		
9	**3888**	**12393**		
11	Positive observations	Negative observations		
12	**3537**	**12744**		
14	Total events (Te)			
15	**16281**			

You can download the spreadsheet here: [5141.Binary Model Performance Metric Calculator.xlsx]

Understanding false positives and false negatives

The performance of a trained model is determined by how good the observations (predictions) reflect the actual events. When you train a (supervised) model, you need a "labeled" data set which includes the actual values which you're trying to predict. You set aside a subset of the labeled data so that, after you trained the model, you can evaluate the model against labeled data which was not used to train the model. By comparing the predicted values against the labeled ones, you end up with four bins:

	Event = positive	Event = negative
Prediction = positive	**True Positive**	**False Positive**
Prediction = negative	**False Negative**	**True Negative**

True positives and true negatives are the observations which were correctly predicted, and therefore shown in green.

The terms "false positive" and "false negative" can be confusing. False negatives are observations where the actual event was positive. The way to think about it is that the terms refer to the observations and

not the actual events. So if the term starts with “false”, the actual value is the opposite of the word that follows it.

Accuracy

Accuracy is perhaps the most intuitive performance measure. It is simply the ratio of correctly predicted observations. Using accuracy is only good for symmetric data sets where the class distribution is 50/50 and the cost of false positives and false negatives are roughly the same.
Example: you are building a model which predicts whether a device is defective. The class distribution is such that 6 in 1000 devices is truly defective (positive). A model which simply returns “negative” – i.e. not defective – all the time gets it right 99.4% of the time and therefore has an accuracy of 0.994, when in fact it never correctly identifies a defective device!

Precision

Precision looks at the ratio of correct positive observations.

The formula is True Positives / (True Positives + False Positives).

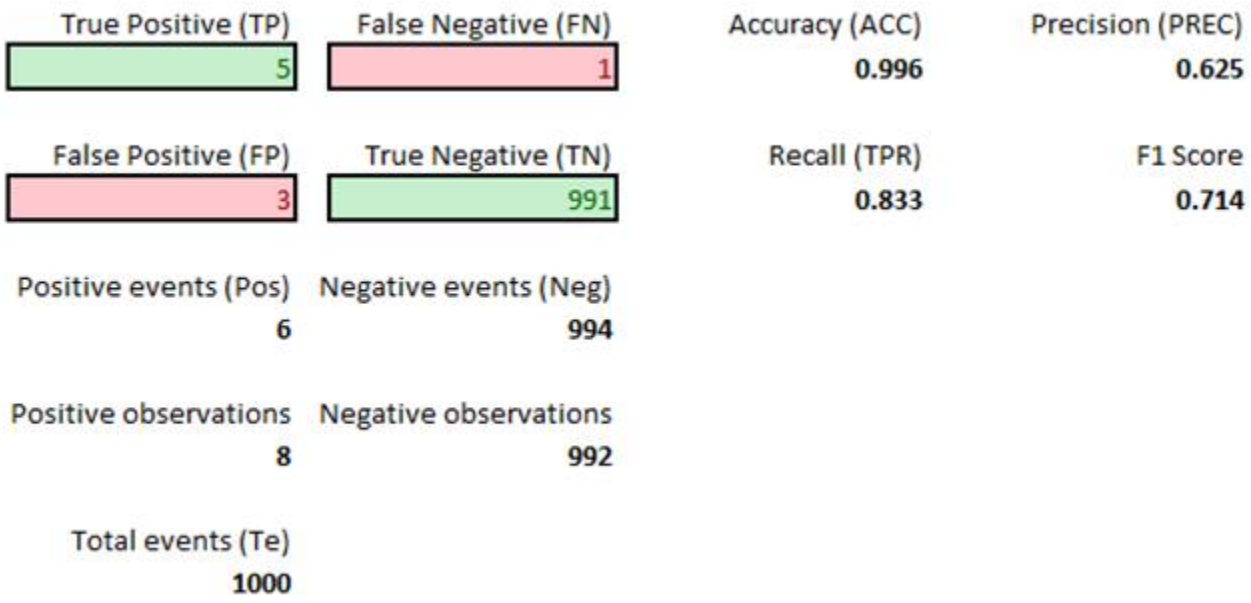
Recall

Recall is also known as **sensitivity** or **true positive rate**. It’s the ratio of correctly predicted positive events.

Recall is calculated as True Positives / (True Positives + False Negatives).

Comparing Precision and Recall

To better understand the difference between precision and recall, here is a screenshot with all the stats:



Both precision and recall work well if there's an uneven class distribution as is often case. They both focus on the performance of positives rather than negatives, which is why it's important to correctly assign the "positive" predicate to the value of most interest.

The precision measure shows what percentage of positive predictions where correct, whereas recall measures what percentage of positive events were correctly predicted. To put it in a different way: precision is a measure of how good predictions are with regard to false positives, whereas recall is measures how good the predictions are with regard to false negatives. Whichever type of error is more important – or costs more – is the one that should receive most

F1 Score

The F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. It works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

The formula for F1 Score is 2*(Recall * Precision)/(Recall + Precision)

Putting it together

The table below shows you which of the measures described in this article probably best to focus on, depending on the class distribution and the cost of false positives and false negatives:

		Class distribution	
		Even	Uneven
Cost	FN cost more	Recall	Recall
	Same cost	Accuracy or F1 score	F1 Score
	FP cost more	Precision	Precision

Threshold and scoring values for Classification

When you scroll down in the screen with the ROC plot, you'll see something like this:

The statistics you see here are for the blue curve which represents the left input of the Evaluate item in the experiment. Notice that when you drag the Threshold slider, all the numbers, except for AUC, change.

To understand why this is the case, you need to know that the learning model doesn't just predict positive or negative as an outcome, but it generates a score which is a real number between 0 and 1, and then uses the threshold setting to decide if the prediction is a "positive" or a "negative". By default, the threshold is set to 0.5 which means that all scoring values above that value are interpreted as positive, and everything below as negative. Therefore, the threshold impacts the predictions which the model makes and thus the related statistics.

While 0.5 is a good default to start with, you will eventually want to tweak the threshold to get the best results. This is the topic of the third article in this series.

The ROC plot

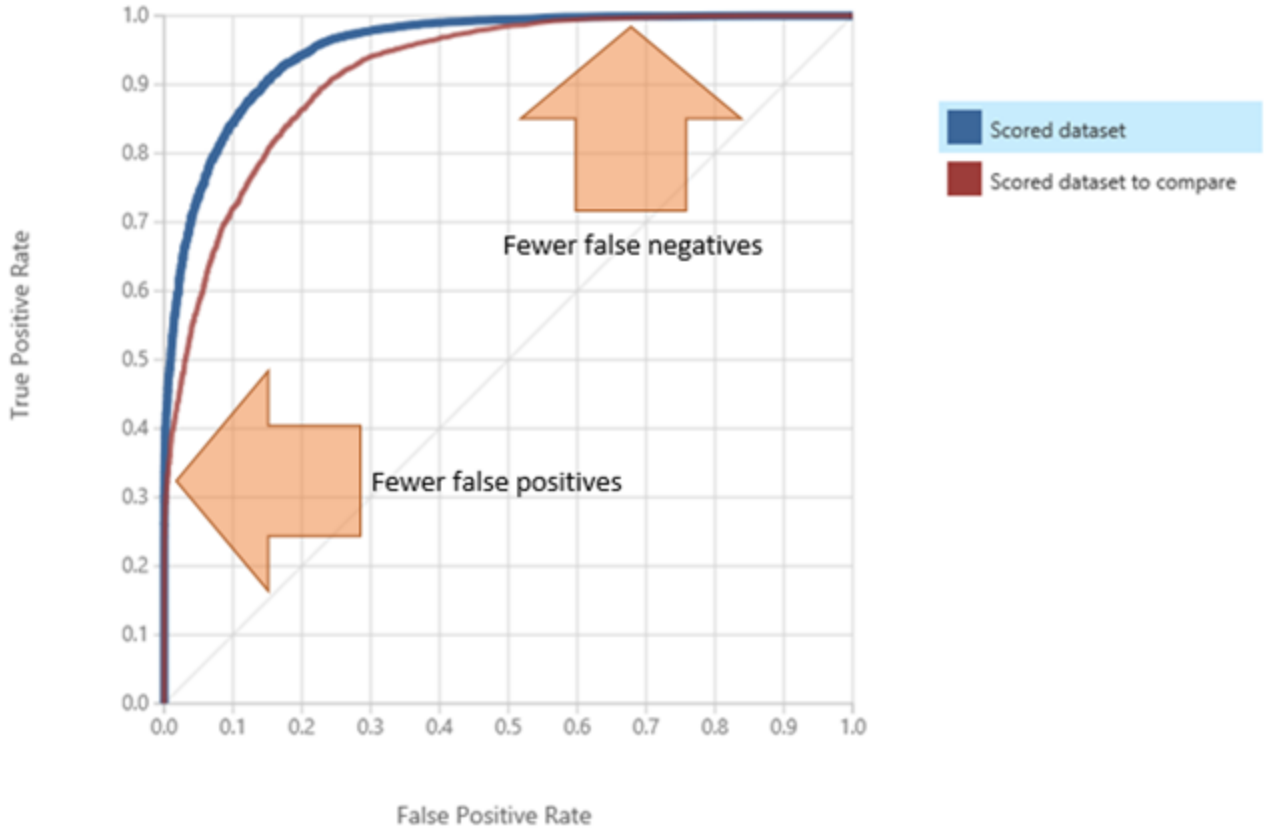
In the ROC plot which was shown before, two learning models are compared. The blue curve represents the model which feeds the left input of the Evaluate item, the red one represents the right input pin. (In this context performance does not relate to the speed but to how well the model predicts the desired outcome).

The blue and red lines in the plot connect the lower-left corner with the upper-right corner of the graph. Each position on the lines represents how the model performs in two different dimensions for a certain threshold setting. Unfortunately the ROC plot does not clearly show which point on the lines represents the current threshold value (perhaps the Azure ML Dev team can note this as a feature request:-), however the lower-left corner (0,0) represents a threshold of 1 whereas the upper-right corner (1,1) represents a threshold of 0. The two dimensions are the false positive rate on the horizontal axis and the true positive rate on the vertical axis.

By changing the threshold, you decrease the frequency of one type of error at the expense of increasing the other type. The position on the curve where the slope is 1 is where they are in balance such that, if you would move the threshold slider a little bit, the number of one type of errors would increase by the same percentage as the other would decrease.

Usually when you see two models compared in a graph as the one above, the lines only intersect in the corners. The line which comes closest to the upper-left corner – in this case the blue line – provides the best predictions. For every threshold value it scores better than the other model for both false positives and false negatives. The further a curve is to the middle, the worse is its performance. Theoretically the worst possible model is one which produces random predictions with the same distribution as the class distribution. Such a model has a ROC curve which approximates a straight diagonal line from (0,0) to (1,1).

You might wonder if curves can also occur *under* the diagonal line. The answer is that if the ROC plot for a learning model would consistently perform under the diagonal, you could simply use the opposite value of what the model predicts as your actual prediction, and the result would then perform better than a random prediction.



AUC

AUC stands for “area under curve”, and as it’s name implies, it refers to the amount of area under the ROC curve, which theoretically is a value between 0 and 1. As I explained, the worst possible curve in practice is a diagonal line, hence the AUC should never be lower than 0.5 (for large data sets) .

Using the AUC metric you can quickly compare multiple learning models. Remember that the ROC curves of two models usually don’t cross each other, hence when comparing two models, the one with a higher AUC will be the better one regardless of the threshold setting. Compared to the statistical measures of accuracy, precision, recall and F1 score, AUC’s independence of threshold makes it uniquely qualified for model selection.

On the other hand, unlike accuracy, precision, recall and F1 score, AUC does not tell us what performance to expect from the model for a given threshold setting, nor can it be used to determine the optimal value for threshold. In that regard it doesn’t take away the need for the other statistical measures.

Putting it together

The ROC plot and the AUC are very useful for comparing and selecting the best machine learning model for a given data set. A model with an AUC score near 1, and where the ROC curve comes close to the

upper left corner, has a very good performance. A model with a score near 0.5 will have a curve near to the diagonal and its performance is hardly better than a random predictor.

After selecting the best model, the next step will be to configure the threshold and the other model configuration settings. The Precision/Recall plot can be helpful for understanding the trade-off between false positives and false negatives.