Evaluation Metrics for Classification 1

Logistic Regression I

Learning Objectives

1. Interpret confusion matrix.
2. Understand metrics used to evaluate classification models: accuracy, sensitivity, specificity, precision and F1 score.

Making Predictions

- Logistic Regression model estimates the probability of default, given a predictor X given by $p(X) = Pr(Y|X)$

```
probs <- fitted(model2)
pp <- train %>% mutate(y_prob = probs)
head(pp)
```

```
    balance   income student default p_default     y_prob
1 1521.4949 18018.90     Yes      No         0 0.41734312
2 1191.4803 30040.57      No      No         0 0.14333612
3 2110.5569 19345.10     Yes     Yes         1 0.97261392
4 1360.8656 22310.93     Yes      No         0 0.19810811
5 1112.3490 33468.82      No      No         0 0.09011374
6  773.5937 29661.46      No      No         0 0.01038311
```

Making Predictions

- Select a cut-off threshold of 0.5 on predicted probability

```
pp <- pp %>% mutate(y_pred = ifelse(y_prob > 0.5, "Yes", "No"))
head(pp)
```

```
    balance   income student default p_default     y_prob y_pred
1 1521.4949 18018.90     Yes      No         0 0.41734312     No
2 1191.4803 30040.57      No      No         0 0.14333612     No
3 2110.5569 19345.10     Yes     Yes         1 0.97261392    Yes
4 1360.8656 22310.93     Yes      No         0 0.19810811     No
5 1112.3490 33468.82      No      No         0 0.09011374     No
6  773.5937 29661.46      No      No         0 0.01038311     No
```

Evaluating Logistic Regression Models

How does a classification model fail?

- Logistic regression model to predict `default` makes errors when it assigns an individual to the wrong category:
  - Individual does not default, but model assigns them to `default = Yes` category (Type 1 error)
  - Individual defaults, but model assigns them to `default = No` category (Type 2 error)
- Which error should we reduce? Ideally both, but in reality, its a trade-off and it depends on the application.

Confusion Matrix

Counting Errors

- Confusion matrix is a summary of prediction results for a classification problem.
  - Positive: event of interest (e.g. $default = Yes$).
  - Negative: negative (or usual) event (e.g. $default = No$).

		Predicted Value	
		Positive	Negative
Actual Value	Positive	True Positive, TP	False Negative, FN (Type 2 Error)
	Negative	False Positive, FP (Type 1 Error)	True Negative, TN

Confusion Matrix

Code

```r
pp$default <- factor(pp$default, levels = c("Yes", "No"))
pp$y_pred <- factor(pp$y_pred, levels = c("Yes", "No"))

conf_mat <- table(pp$default, pp$y_pred, deparse.level = 0)

colnames(conf_mat) <- c("y_pred=Yes", "y_pred=No")
rownames(conf_mat) <- c("default=Yes", "default=No")

conf_mat
```

```
            y_pred=Yes y_pred=No
default=Yes       2397       567
default=No         512      7188
```

Confusion Matrix

Credit default data

		Predicted Value: Positive	Predicted Value: Negative	
Actual Value	Positive	TP = 2397	FN = 567	# Actual Positives = 2964
	Negative	FP = 512	TN = 7188	# Actual Negatives = 7700
		# Predicted Positives = 2909	# Predicted Negatives = 7755	Total # = 10664

Accuracy

Overall, how often is the model correct?

Predicted Value

Actual Value	Positive	Negative	
Positive	TP = 2397	FN = 567	# Actual Positives = 2964
Negative	FP = 512	TN = 7188	# Actual Negatives = 7700
	# Predicted Positives = 2909	# Predicted Negatives = 7755	Total # = 10664

$$\text{Accuracy} = \frac{\text{No. of correct predictions}}{\text{Total no. of predictions}} = \frac{\text{TP+TN}}{\text{TP+FN+FP+TN}} = \frac{2397 + 7188}{10664} = \frac{9585}{10664} = 0.899 \tag{1}$$

Sensitivity

When the actual value is yes, how often does the model predict it as yes?

		Predicted Value: Positive	Predicted Value: Negative	
Actual Value	Positive	TP = 2397	FN = 567	# Actual Positives = 2964
	Negative	FP = 512	TN = 7188	# Actual Negatives = 7700
		# Predicted Positives = 2909	# Predicted Negatives = 7755	Total # = 10664

$$\text{Sensitivity} = \frac{\text{No. of true positives}}{\text{No. of actual positives}} = \frac{\text{TP}}{\text{TP+FN}} = \frac{2397}{2964} = 0.809 \tag{2}$$

- Sensitivity is also called recall or true positive rate (TPR) in literature

Specificity

When the actual value is no, how often does the model predict it as no?

Actual Value \ Predicted Value	Positive	Negative	
Positive	TP = 2397	FN = 567	# Actual Positives = 2964
Negative	FP = 512	TN = 7188	# Actual Negatives = 7700
	# Predicted Positives = 2909	# Predicted Negatives = 7755	Total # = 10664

$$\text{Specificity} = \frac{\text{No. of true negatives}}{\text{No. of actual negatives}} = \frac{\text{TN}}{\text{TN+FP}} = \frac{7188}{7700} = 0.934 \tag{3}$$

- An associated metric is the false positive rate, $\text{FPR} = 1 - \text{Specificity}$. Here, $\text{FPR} = 0.067$
- Good classification model maximises both sensitivity and specificity

Precision

When the model predicts a yes, how often is it correct?

		Predicted Value		
		Positive	Negative	
Actual Value	Positive	TP = 2397	FN = 567	# Actual Positives = 2964
	Negative	FP = 512	TN = 7188	# Actual Negatives = 7700
		# Predicted Positives = 2909	# Predicted Negatives = 7755	Total # = 10664

$$\text{Precision} = \frac{\text{No. of true positives}}{\text{No. of predicted positives}} = \frac{\text{TP}}{\text{TP+FP}} = \frac{2397}{2909} = 0.824 \tag{4}$$

- Precision is also known as positive predictive value (PPV) in literature.
- Higher precision is better.

F1-Score

Combined Precision and Recall

		Predicted Value		
		Positive	Negative	
Actual Value	Positive	TP = 2397	FN = 567	# Actual Positives = 2964
	Negative	FP = 512	TN = 7188	# Actual Negatives = 7700
		# Predicted Positives = 2909	# Predicted Negatives = 7755	Total # = 10664

$$\mathrm{F1} = 2.\frac{\text{precision.recall}}{\text{precision} + \text{recall}} = \frac{\mathrm{TP}}{\mathrm{TP} + \frac{1}{2}(\mathrm{FP} + \mathrm{FN})} = \frac{2397}{2397 + \frac{1}{2}.(512 + 567)} = 0.816 \tag{5}$$

- Higher F1 score is better.

Evaluation Metrics

Code

```
TP <- conf_mat[1,1]
TN <- conf_mat[2,2]
FP <- conf_mat[2,1]
FN <- conf_mat[1,2]
Total <- nrow(pp)

acc <- (TP+TN)/nrow(pp)
sens <- TP/(TP+FN)
spec <- TN/(TN+FP)
prec <- TP/(TP+FP)
f1s <- TP/(TP+0.5*(FP+FN))

evalnames <- c("accuracy", "sensitivity", "specificity", "precision",
    "F1_score")
evaldata <- c(acc, sens, spec, prec, f1s)
evalmetrics <- data.frame(Metric = evalnames, Value = evaldata)
evalmetrics
```

Evaluation Metrics

```
       Metric     Value
1    accuracy 0.8988185
2 sensitivity 0.8087045
3 specificity 0.9335065
4   precision 0.8239945
5    F1_score 0.8162779
```

References I

Bruce, P., Bruce, A. G., and Gedeck, P. (2020).
Practical statistics for data scientists: 50 essential concepts using R and Python.
OReilly.