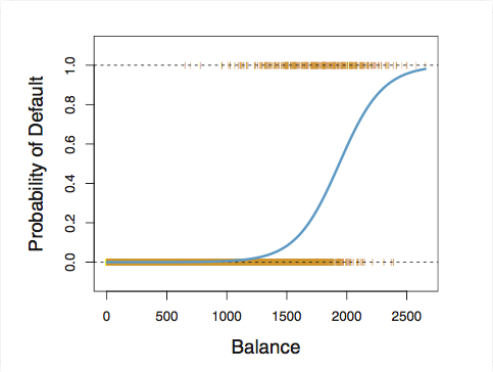
Logistic Regression

Predicting Credit Card Defaulters

When to use logistic regression?

- With linear regression in the previous module, we assumed that the response variable (median house value) was quantitative. However, sometimes we are interested in qualitative response variables
- examples: default/no default, cancer/no cancer, buy/not buy
- with qualitative response variables, we can use logistic regression
- let's consider credit card defaults as an example. Our response variable is default/no default, and our independent variable is bank balance.

Logistic Regression



source: Introduction to Statistical Learning

- response variables take a value between 0 (no default) and 1(default). These can be interpreted as a probability
- P(default = Yes | balance)
- we choose an appropriate classification cut-off point depending on our use case. To be more conservative, we can choose a lower threshold. However, a lower threshold also means we might classify a non-defaulter as a defaulter.

Evaluating our Model

- Confusion Matrix

		Predicted class		
		− or Null	+ or Non-null	Total
True class	− or Null	True Neg. (TN)	False Pos. (FP)	N
	+ or Non-null	False Neg. (FN)	True Pos. (TP)	P
	Total	N*	P*	

TABLE 4.6. *Possible results when applying a classifier or diagnostic test to a population.*

source: Introduction to Statistical Learning

- Confusion Matrices are confusing, so how can we make things easier?

ROC Curve

- Visualise accuracy using an ROC curve
- ROC Curves plot true positive rate against false positive rate.
- Ideally should hug the top left corner

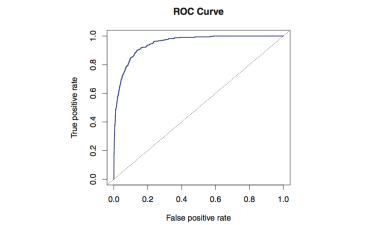
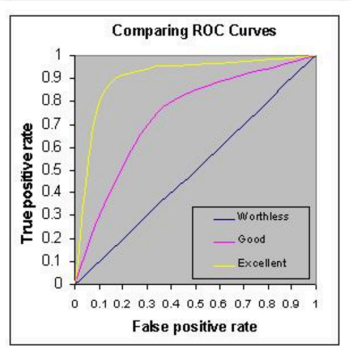


FIGURE 4.8. *A ROC curve for the LDA classifier on the* `Default` *data. It traces out two types of error as we vary the threshold value for the posterior probability of default. The actual thresholds are not shown. The true positive rate is the sensitivity: the fraction of defaulters that are correctly identified, using a given threshold value. The false positive rate is 1-specificity: the fraction of non-defaulters that we classify incorrectly as defaulters, using that same threshold value. The ideal ROC curve hugs the top left corner, indicating a high true positive rate and a low false positive rate. The dotted line represents the "no information" classifier; this is what we would expect if student status and credit card balance are not associated with probability of default.*

ROC Curves continued



- As the curve moves to the top-left, accuracy (area under the curve) increases. Therefore we want a large area under the curve.

figure source: **http://gim.unmc.edu/dxtests/roc3.htm**

Let's get started with a dataset

Credit Card Defaulters

A simulated data set containing information on ten thousand customers. The aim here is to predict which customers will default on their credit card debt.

default: A factor with levels No and Yes indicating whether the customer defaulted on their debt

student: A factor with levels No and Yes indicating whether the customer is a student

balance: The average balance that the customer has remaining on their credit card after making their monthly payment

income: Income of customer

```
library(MASS)
library(ISLR)
library(caret)
data(Default)
```

```
#look at first 10 columns
str(Default)
```

```
'data.frame':   10000 obs. of  4 variables:
 $ default: Factor w/ 2 levels "No","Yes": 1
1 1 1 1 1 1 1 1 1 ...
 $ student: Factor w/ 2 levels "No","Yes": 1
2 1 1 1 2 1 2 1 1 ...
 $ balance: num  730 817 1074 529 786 ...
 $ income : num  44362 12106 31767 35704
38463 ...
```

Perform train/test split

```
set.seed(88)
train_Index <-
createDataPartition(Default$default, p=0.8,
list=FALSE)
train_credit <- Default[train_Index,]
test_credit <- Default[-train_Index,]
head(train_credit)
```

```
  default student   balance   income
2      No     Yes  817.1804 12106.13
3      No      No 1073.5492 31767.14
4      No      No  529.2506 35704.49
5      No      No  785.6559 38463.50
7      No      No  825.5133 24905.23
9      No      No 1161.0579 37468.53
```

Model 1: Single Variable Logistic Regression

```
model1 <- train(default ~ balance, data =
train_credit, method = "glm", family =
"binomial")
```

Interpreting the Coefficients

```
coef(model1$finalModel)
```

```
  (Intercept)       balance
-10.489595472   0.005398078
```

our estimates will be on a log-odds scale. For example, for every one unit increase in balance, the log-odds of default increases by about 0.005. To make the coefficients more understandable, we can use the exponential function to calculate the odds ratio for our predictors

```
exp(coef(model1$finalModel))
```

```
 (Intercept)      balance
2.782445e-05 1.005413e+00
```

Therefore, for every one unit increase in balance, our odds of having good credit increases by 1.01

Using our Model for Prediction

```
pred <- predict(model1, newdata=test_credit)
head(pred)
```

```
[1] No No No No No No
Levels: No Yes
```

Alternatively we can output class probabilities

```
pred2 <- predict(model1, test_credit, type =
"prob")
head(pred2)
```

```
           No          Yes
1   0.9985741 1.425858e-03
6   0.9960323 3.967732e-03
8   0.9978159 2.184143e-03
12  0.9801750 1.982496e-02
13  0.9999000 1.000233e-04
17  0.9999722 2.782367e-05
```

Evaluating our Model: Confusion Matrix

```
confusionMatrix(data=pred,
test_credit$default)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1928   42
       Yes    5   24

               Accuracy : 0.9765
                 95% CI : (0.9689, 0.9827)
    No Information Rate : 0.967
    P-Value [Acc > NIR] : 0.007885

                  Kappa : 0.4951
 Mcnemar's Test P-Value : 1.512e-07

            Sensitivity : 0.9974
            Specificity : 0.3636
         Pos Pred Value : 0.9787
         Neg Pred Value : 0.8276
             Prevalence : 0.9670
         Detection Rate : 0.9645
   Detection Prevalence : 0.9855
      Balanced Accuracy : 0.6805

       'Positive' Class : No
```

```
# Kappa is an alternative to accuracy that
takes into account dataset imbalance
```

Area under ROC Curve

```
library(pROC)
rpartROC <- roc(test_credit$default,
pred2[,"No"])
rpartROC
```

```
Call:
roc.default(response = test_credit$default,
predictor = pred2[,     "No"])

Data: pred2[, "No"] in 1933 controls
(test_credit$default No) > 66 cases
(test_credit$default Yes).
Area under the curve: 0.963
```

Model2: Three Variable Logistic Regression

```
model2 <- train(default~ ., data =
train_credit, family='binomial')
```

```
note: only 2 unique complexity parameters in
default grid. Truncating the grid to 2 .
```

Do-it-yourself

```
## using the code above as a template, apply
model2 to test data. Then tabulate a
Confusion Matrix ##
## your code goes here ##
```

EXTENSION: Feature Engineering

Finding the optimal combination of independent variables is known as feature engineering.

A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. The data were collected by the US National Institute of Diabetes and Digestive and Kidney Diseases.

Load Data

The dataset has already been split for us, for we load the train and test set individually

```
train_PIMA <- Pima.tr
test_PIMA <- Pima.te
str(train_PIMA)
```

```
'data.frame':   200 obs. of  8 variables:
 $ npreg: int  5 7 5 0 0 5 3 1 3 2 ...
 $ glu  : int  86 195 77 165 107 97 83 193
142 128 ...
 $ bp   : int  68 70 82 76 60 76 58 50 80 78
...
 $ skin : int  28 33 41 43 25 27 31 16 15 37
...
 $ bmi  : num  30.2 25.1 35.8 47.9 26.4 35.6
34.3 25.9 32.4 43.3 ...
 $ ped  : num  0.364 0.163 0.156 0.259 0.133
...
 $ age  : int  24 55 35 26 23 52 25 24 63 31
...
 $ type : Factor w/ 2 levels "No","Yes": 1 2
1 1 1 2 1 1 1 2 ...
```

Our variables

- npreg: number of pregnancies.
- glu : plasma glucose concentration in an oral glucose tolerance test.
- bp : diastolic blood pressure (mm Hg).
- skin : triceps skin fold thickness (mm).
- bmi : body mass index (weight in kg/(height in m)^2).
- ped : diabetes pedigree function.
- age : age in years.
- type : Yes or No, for diabetic according to WHO criteria.

Try finding a good combination of features for predicting the onset of diabetes

```
## your code goes here ##
```