Confusion Matrices

A **confusion matrix** compares predictions with outcomes for classification problems. It is a square matrix with as many rows and columns as there are classes.

Usually, the K rows and columns are augmented with one additional row and one additional column to show the totals.

Ideally, the confusion matrix would be diagonal. In that case, the method predicts perfectly.

Consider some kind of medical test, where the outcomes are P (for positive) and N (for negative).

Outcomes are AP (actual positive) and AN (actual negative).

Predictions are PP (predicted positive) and PN (predicted negative).

By addition, PP = FP + TP, PN = FN + TN, AP = TP + FN, and AN = TN + FP, where "F" means false and "T" means true.

Table 1: **A Confusion Matrix for Two Outcomes**

Prediction/Outcome	Positive	Negative	Total
Positive	TP	FP	PP
Negative	FN	TN	PN
Total	AP	AN	TOT

For example, we might have 1000 observations divided as follows.

Table 2: **Example of a Confusion Matrix**

Prediction/Outcome	Positive	Negative	Total
Positive	323	13	336
Negative	19	645	664
Total	342	658	1000

In this case, the test works pretty well.

We can easily compute error rates from the confusion matrix.

- The fraction of positive outcomes incorrectly identified as negative is $19/342 = 0.0556$.
- The fraction of negative outcomes incorrectly identified as positive is $13/658 = 0.0198$.
- The fraction of positive outcomes correctly identified as positive is TP/(TP + FN). This is the test's **sensitivity**. In this case, it is $323/342 = 0.944$.
- The fraction of negative outcomes correctly identified as negative is TN/(TN + FP). This is the test's **specificity**. It is equal to $1 - \text{FP}/(\text{TN} + \text{FP})$. In this case, it is $645/658 = 0.980$.
- The **test error rate** is $(\text{FP} + \text{FN})/\text{TOT} = 1 - (\text{TP} + \text{TN})/\text{TOT}$.

If we are comparing alternative classification methods, it is entirely possible that one may perform better for one outcome and another for a different outcome.

For the example in Table 2, specificity is higher than sensitivity. We do a better job at identifying negative outcomes than positive ones.

Perhaps another learning method has the following confusion matrix:

Table 3: **A Confusion Matrix for a Different Method**

Prediction/Outcome	Positive	Negative	Total
Positive	336	22	358
Negative	6	636	652
Total	342	658	1000

Now sensitivity, TP/(TP + FN), is $336/342 = 0.982$, but specificity, TN/(TN + FP), is $636/658 = 0.967$.

So the second method has better sensitivity—it is more likely to predict a positive correctly, since $0.982 > 0.944$. But it has worse specificity—it is less likely to predict a negative correctly, since $0.967 < 0.980$.

It is easy to test whether a classification procedure is useful at all by using a standard χ^2 test for no association in a contingency table.

We calculate the "expected" quantities using the row and column totals. Thus

$$\mathrm{TP_E} = \frac{\mathrm{PP} \cdot \mathrm{AP}}{\mathrm{TOT}} \tag{1}$$

and

$$\mathrm{TN_E} = \frac{\mathrm{PN} \cdot \mathrm{AN}}{\mathrm{TOT}}. \tag{2}$$

The test statistic is

$$\frac{(\mathrm{TP} - \mathrm{TP_E})^2}{\mathrm{TP_E}} + \frac{(\mathrm{TN} - \mathrm{TN_E})^2}{\mathrm{TN_E}} + \frac{(\mathrm{FP} - \mathrm{FP_E})^2}{\mathrm{FP_E}} + \frac{(\mathrm{FN} - \mathrm{FN_E})^2}{\mathrm{FN_E}}. \tag{3}$$

This is asymptotically distributed as $\chi^2(1)$.

For three outcomes, the confusion matrix is 3×3, and the analogous test statistic is asymptotically distributed as $\chi^2(4)$.

ROC Curves

The performance of a binary classification method can be evaluated by plotting its **ROC curve**.

This can be plotted for training data, for test data, or for training data using cross-validation. An ROC curve based on training data will tend to be too optimistic.

"ROC" stands for "receiver operating characteristics."

In another context, it is called a **size-power tradeoff curve**.

It involves plotting the false positive rate (FPR) on the horizontal axis against the true positive rate (TPR) on the vertical axis.

The false positive rate is

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}, \tag{4}$$

which equals 1 minus the specificity.

The true positive rate is

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{5}$$

which equals the sensitivity.

Because FPR and TPR have different denominators, one may be estimated more accurately than the other, leading to an ROC curve that is rougher in one dimension than in the other.

What changes along the ROC curve (but is not explicitly plotted) is the **threshold** used to classify an observation as positive (i.e. 1) or negative (i.e. 0).

The ROC curve shows the TPR/FPR tradeoff for all possible thresholds.

For binary data, a model like logistic regression or KNN gives us $\Pr(y_i = 1)$ for every observation in the training, or test, dataset.

But we do not have to classify observation i as 1 if $\Pr(y_i = 1) \geq 0.5$. Instead of 0.5, we could use any threshold, say τ, between 0 and 1. In fact, we do not even need a probability model.

- For example, the logit model has $\Pr(y_i = 1) = \Lambda(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}})$.
- We could use a threshold for $\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}$ instead of a threshold for $\Lambda(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}})$ and get the same ROC curve.
- Instead of τ between 0 and 1, we would use τ between $\min(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}})$ and $\max(\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}})$ when classification is based directly on $\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}$.

When τ is very large, we never classify an observation as 1. When it is very small, we never classify an observation as 0.

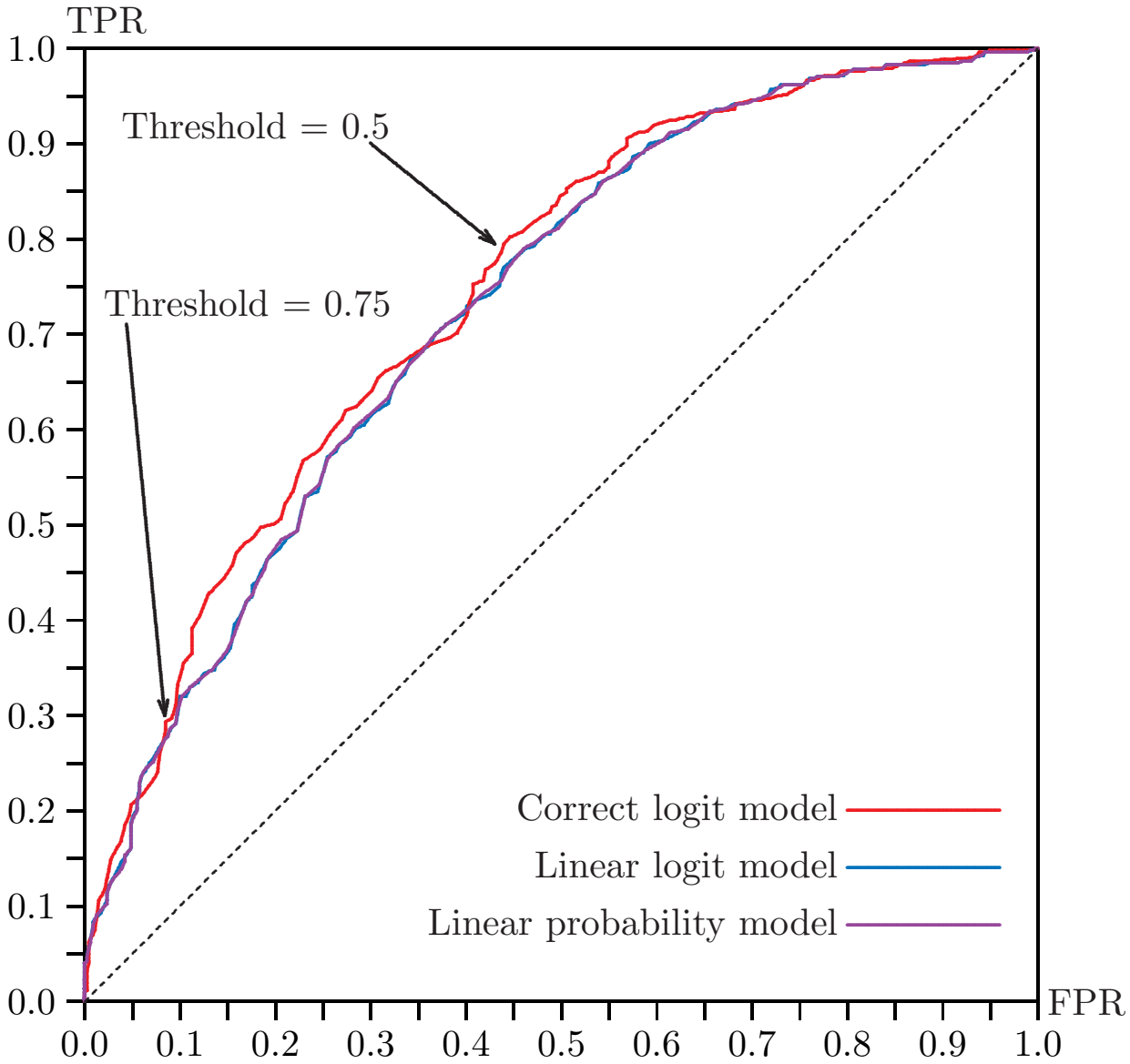
As τ decreases, both the false positive rate and the true positive rate increase. If the model is good, TPR should initially increase much faster than FPR. For a useless model, the ROC curve is the 45° line.

Ideally, the ROC curve will look like a Γ. As τ decreases initially, FPR will remain at 0, but TPR will shoot up to 1.

All we need to plot an ROC curve is an ordered list of predictions and outcomes. Consider the logit model with $\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}} = 0.6 + 0.8 x_i$.

- We can evaluate this for any x_i and see whether $y_i = 0$ or $y_i = 1$.
- If the smallest value of x_i in the test dataset is -3, then the smallest value of the index function is $0.6 - 0.8 \times 3 = -1.8$.
- For any τ below -1.8, both TPR and FPR will be 1.
- If the largest value of x_i in the test dataset is 2.5, then the largest value of the index function is $0.6 + 0.8 \times 2.5 = 2.6$.
- For any τ above 2.6, both TPR and FPR will be 0.
- For values of τ between -1.8 and 2.6, at least one of TPR and FPR will be greater than 0 and less than 1.
- We can look at all the y_i and how they are classified for a large number of values of τ between -1.8 and 2.6.
- If we start at $\tau = 2.6$ and decrease it, both TPR and FPR will increase. TPR should initially increase much faster than FPR.

Figure 7.1 (Three ROC Curves)



I generated a simulated dataset with two $N(0,1)$ explanatory variables. The DGP is logit, but there is an interaction term, so that it is somewhat nonlinear.

Initially, I used three estimators: logit (nonlinear, as in the DGP), logit (linear in both predictors), and LPM (linear in both predictors).

Only one of these is the "correct" model.

- The training sample has 2000 observations, and the test sample has 1000 observations.
- For the logit models, the threshold ran from 0 to 1. I evaluated FPR and TPR at 201 points, from 0.000, 0.005, 0.010, and so on, to 1.000. [Maybe 201 is too big in this case.]
- For the LPM, the threshold ran from just below the lowest to just above the largest fitted value. These were -0.0835 and 1.259.

ROC curves were plotted using the test data. See Figure 7.1.

Surprisingly, LPM and linear logit are all but identical!

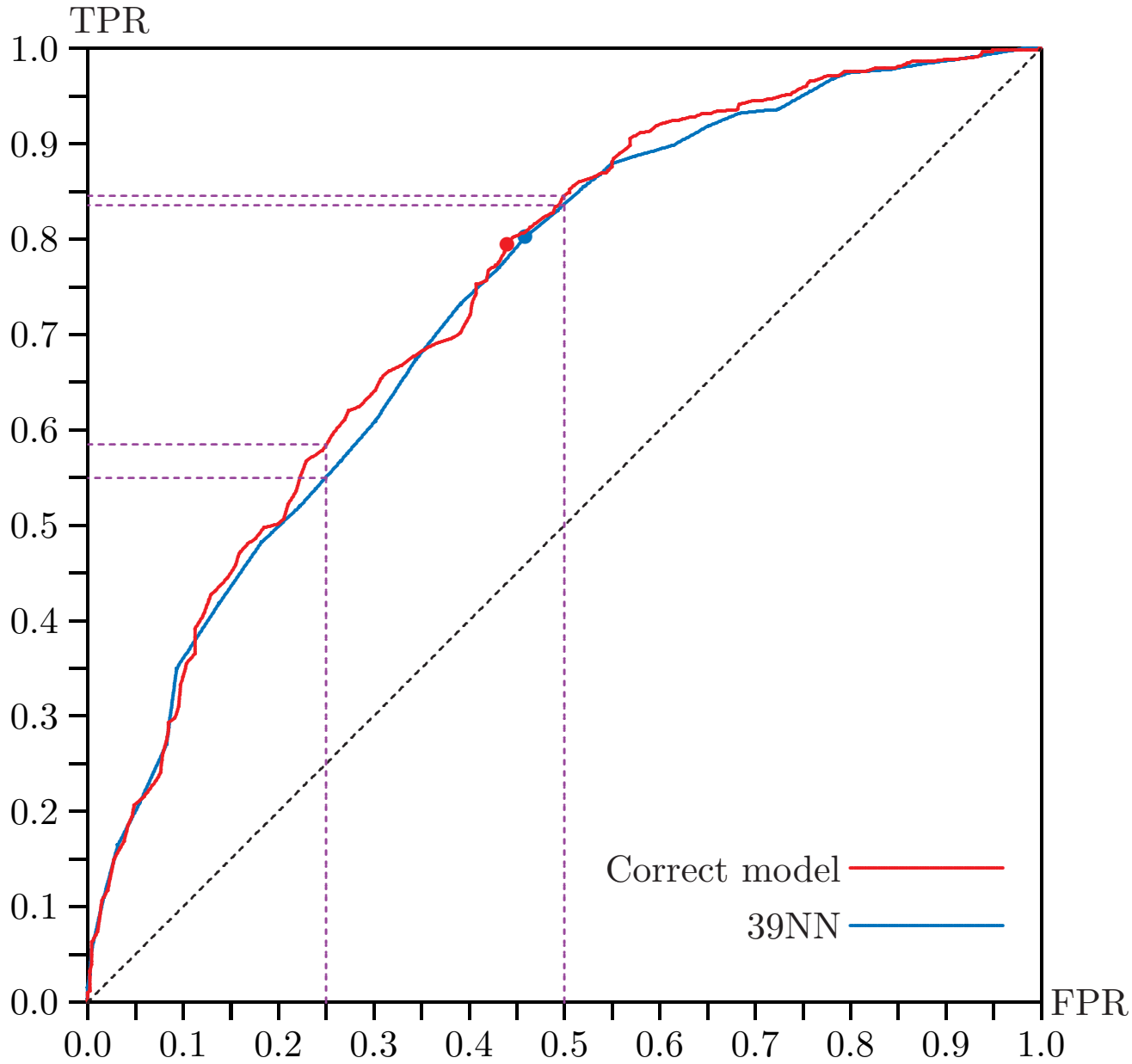
I also tried KNN with this artificial dataset. See Figure 7.2.

- Based on the test data, the optimal value of K is 39. Of course, this is cheating a bit.
- The ROC curve for 39NN is close to the one for the correct model, and much smoother.
- That is because there are only 40 values of $\widehat{\Pr}(1 \mid x_0)$, two of which are 0 and 1.
- Thus the values of the false positive rate and the true positive rate can only change 39 times.
- For the correct logit model, they can change 200 times the way I plotted the figure.

It appears that 39NN works a bit better than either logit or the LPM based on the incorrect assumption of linearity.

The bullets show FPR/TPR values for a threshold of 0.5.

Figure 7.2 (ROC curve for 39NN)



There are numerous packages in R that can draw ROC curves.

Many of them can also compute the **area under the curve** or **AUC**, and perhaps a confidence interval for the AUC.

Ideally, the AUC would be 1. For a useless classifier, it is 0.5.

I suspect that the algorithm for computing AUC is a bit tricky.

The **Gini coefficient** is $2\text{AUC} - 1$.

People sometimes rank classifiers by their AUCs. However, this treats all mistakes as equally harmful.

Two ROC curves could cross but have the same AUC.

We may prefer one of them to the other. If the cost of missing a 1 is very high (e.g. "1" means that a plane crashes), then we may be happy to accept a lot of false positives (FPR high).

Conversely, if the cost of missing a 1 is low and the cost of a false positive is high, we may be willing to accept a lot of false negatives (TPR low).

The shape of an ROC curve depends on how the data are actually generated and on how well our estimates approximate the DGP.

When the test sample is small, the curve may be quite rough.

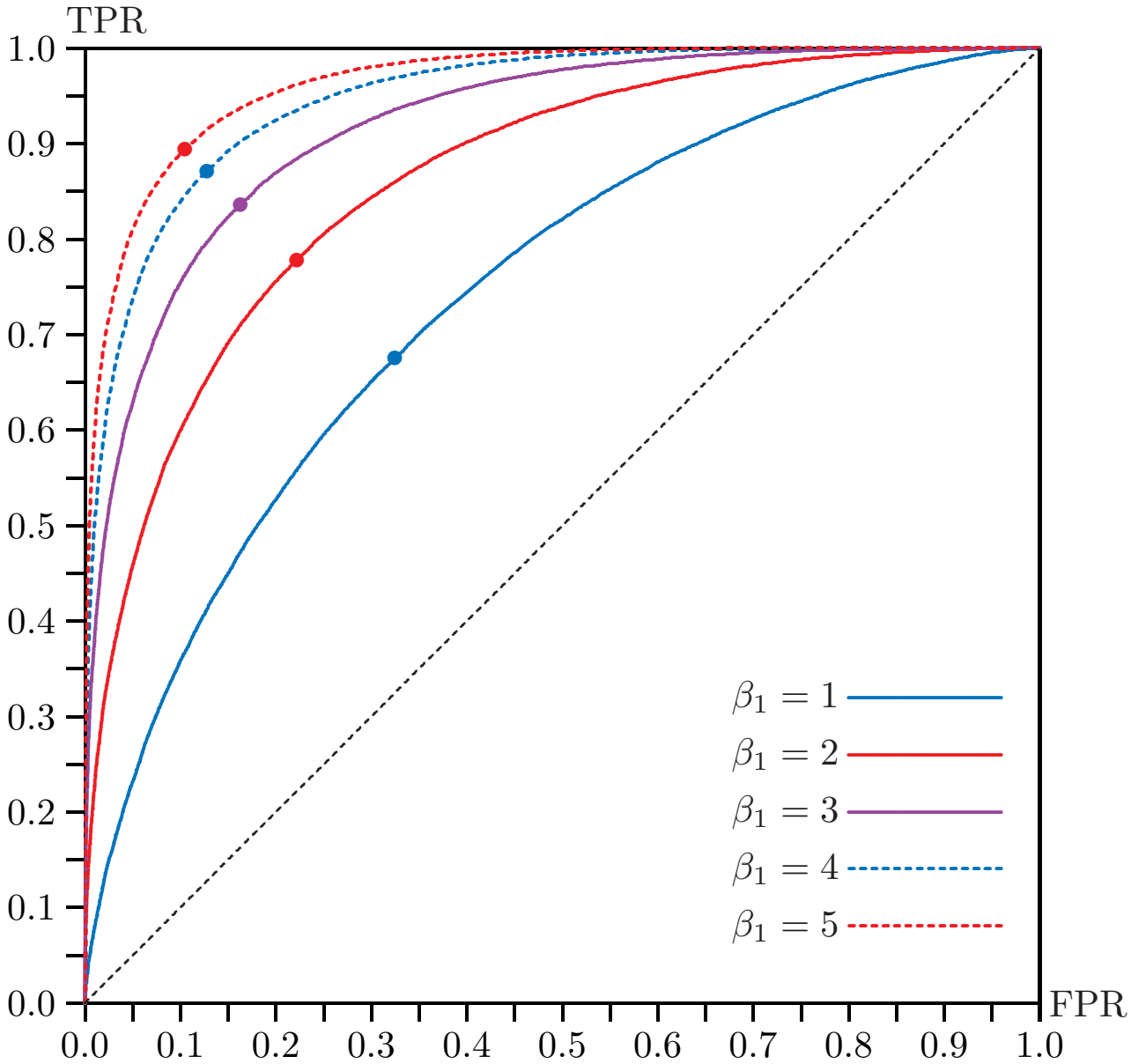
When either the training or test samples are small, the shape of the curve may vary a lot across samples.

In the next few figures, I assume that we know the DGP and that the test sample is infinitely large (actually, 100,000).

Thus the ROC curves show the tradeoff between the false positive rate (FPR) and the true positive rate (TPR) in the best possible situation.

- The model is a logit model with a single predictor, x, which is $\mathrm{N}(0,1)$. The log of the odds is equal to $\beta_0 + \beta_1 x$.
- Initially, $\beta_0 = 0$, so that (on average) half the observations equal 1 and half equal 0.
- As β_1 increases, our ability to classify improves.
- The bullets show the Bayesian classifier (threshold = 0.5).

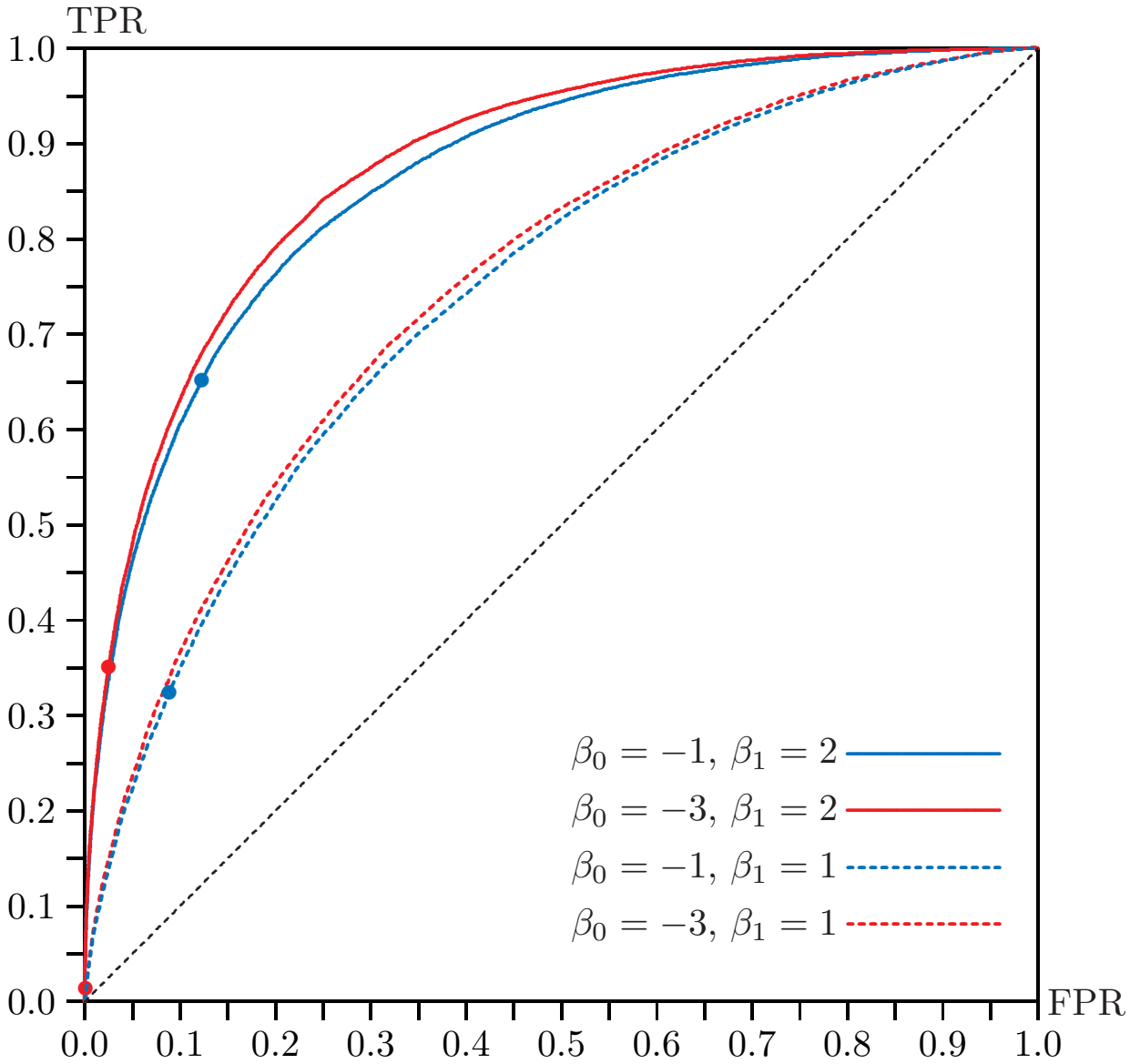
Figure 7.3 (Five ROC curves for true DGP)



In Figure 7.3, the two outcomes (1 and 0) are equally probable, because $\beta_0 = 0$ and the input has mean 0.

- As β_0 increases, outcome 1 becomes more likely.
- As β_0 decreases, outcome 0 becomes more likely.
- Figure 7.4 shows ROC curves where $\Pr(1) < \Pr(0)$.
- The shapes of the ROC curves do not change much, but the threshold needed to obtain any FPR/TPR pair changes radically.
- When $\Pr(1) << \Pr(0)$, we need a very low threshold if we want to predict more than a very few positives.
- When $\beta_0 = -1$ and $\beta_1 = 2$, $\Pr(1) = 0.354$.
- When $\beta_0 = -3$ and $\beta_1 = 2$, $\Pr(1) = 0.130$.
- When $\beta_0 = -1$ and $\beta_1 = 1$, $\Pr(1) = 0.303$.
- When $\beta_0 = -3$ and $\beta_1 = 1$, $\Pr(1) = 0.069$.
- In this case, using a threshold of 0.5 results in a false positive rate of 0.0005 and a true positive rate of 0.0119.

Figure 7.4 (More ROC curves for true DGP)



Comparing Several Methods

In Section 4.5 of ISLR/ISLP, six classification methods are compared for six scenarios using simulated data.

Comparisons are based on the **test error rate**, which is the number of off-diagonal observations in the confusion matrix divided by the total number of observations.

The confusion matrix is based on a test sample.

It would be better to compare predicted probabilities with outcomes, especially if some probabilities are small.

The methods are 1NN, KNN-CV, LDA, logit, naive Bayes, and QDA.

1NN minimizes bias but maximizes variance. It is stupid and is never the winner.

KNN-CV means KNN choosing K by cross-validation. Thus the actual value of K will differ across the simulated samples.

Figure 4.11 shows the distributions of the test error rates for Scenarios 1–3 using box plots.

- The box in the middle shows the 0.25 and 0.75 quantiles, with the line in the middle showing the median.
- The whiskers show extreme quantiles such as 5 and 95, 2.5 and 97.5, or 1 and 99. The book does not say which quantiles they are.
- In many cases, a few (or more than a few) extreme points are shown beyond the upper whisker.
- However, there are never any points below the lower whisker.

There is evidently quite a bit of variation across the simulated datasets.

This figure does not show the correlations across methods, which are probably high.

It also tells us nothing about how the results would change if the thresholds for classification were varied.

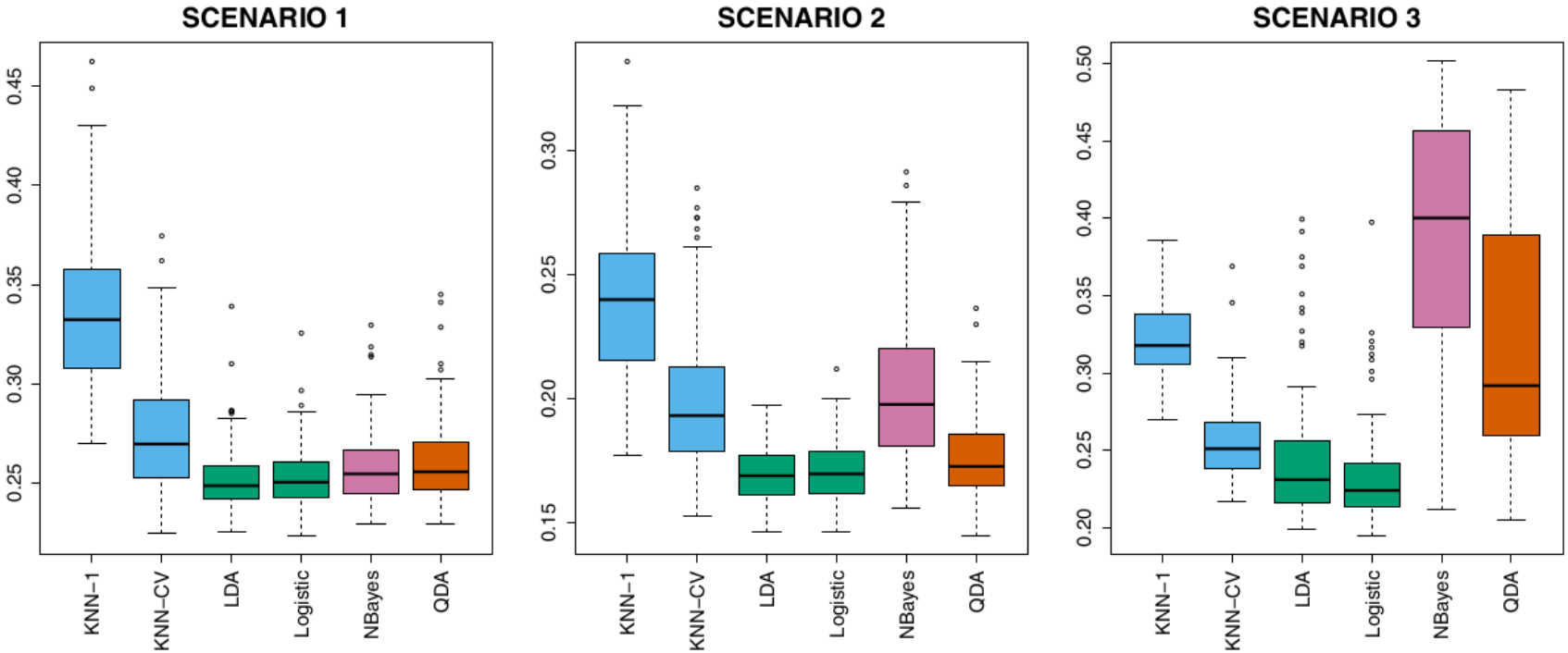


FIGURE 4.11. *Boxplots of the test error rates for each of the linear scenarios described in the main text.*

- Scenario 1 involves two uncorrelated, normally distributed linear predictors. So this is perfect for LDA.
- Scenario 2 is similar, but the predictors have correlation -0.5.

For both these scenarios, LDA is the winner, with logit a close second. QDA is worse, but considerably better than KNN-CV.

- In Scenario 3, the predictors came from a t distribution with unspecified degrees of freedom.

Logit now does noticeably better than LDA, and QDA and naive Bayes do much worse.

Perhaps the thicker tails make it hard to estimate the covariance matrix needed by LDA, the covariance matrices needed by QDA, and the densities needed by naive Bayes.

- In Scenario 4, the predictors are once again normal, but with correlations of $+0.5$ for class 1 and -0.5 for class 2.

This conforms to the assumptions of QDA, which works best. Interestingly, KNN-CV now beats both LDA and logit.

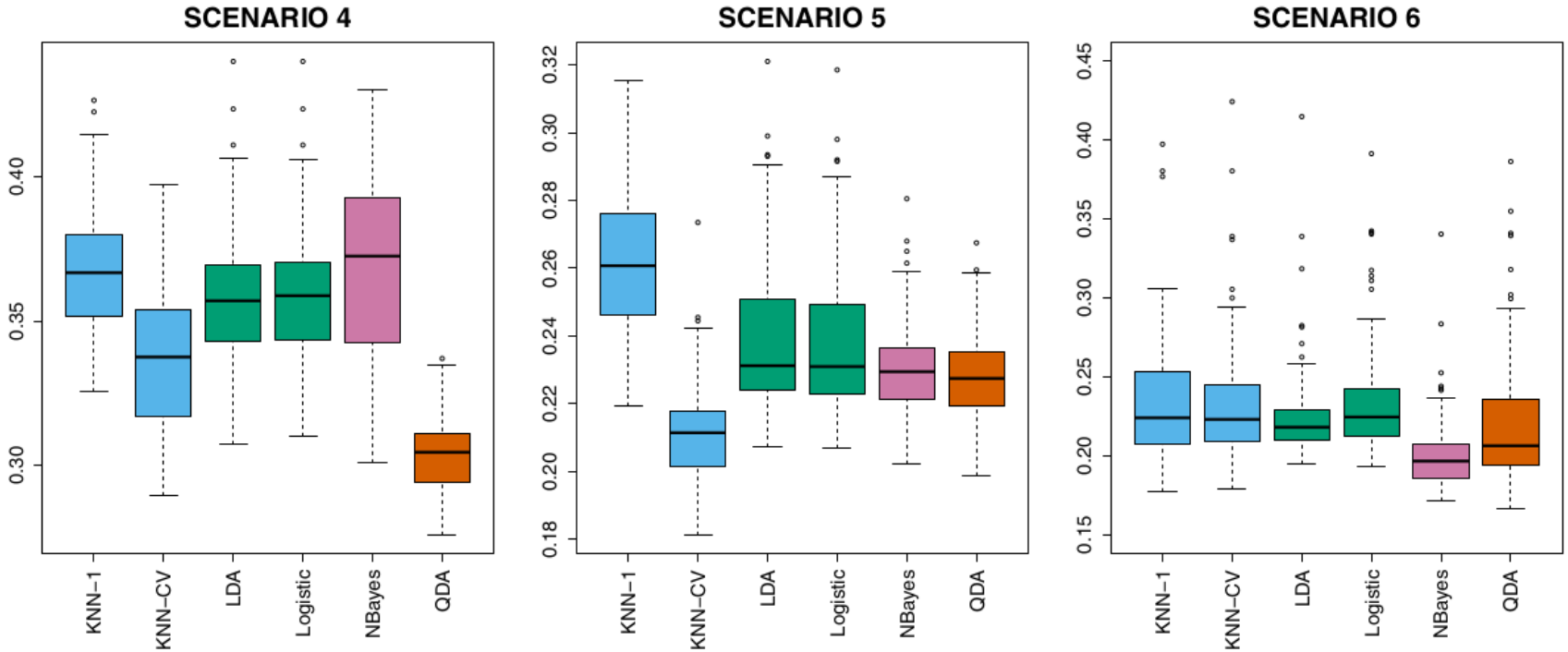


FIGURE 4.12. *Boxplots of the test error rates for each of the non-linear scenarios described in the main text.*

- In Scenario 5, the predictors are once again normal and uncorrelated, but the DGP is a logit model with "a complicated nonlinear function of the predictors."

KNN-CV is now the clear winner.

The linear logit and LDA models perform poorly.

Of course, if we modified the logit model to include squares and cross-products, it would probably work better.

- In Scenario 6, the predictors were generated from a normal distribution with a different diagonal covariance matrix for each class. However, the sample size was very small, with just 6 observations for each class.

Now naive Bayes works best, followed by QDA. The latter would have worked better if the sample size had been larger.

These experiments did not focus on the size of the training sample or the number of predictors.