Evaluating Prediction performance for classification problems

Yang Wang

College of Charleston

Classification problem: estimation of class membership

- Consider a two-class case with classes C_1 and C_2 (eg. defaulter/non-defaulter)
- Suppose C_1 is the class of importance
- Most classification algorithms produce a vector of estimated probabilities (propensities) for belonging to the class of interest
- If the estimated probability for a record is greater than a pre-specified *cutoff value*, the record is assigned to the class of interest
- The default cutoff value in a two-class case is 0.5
- However, it is possible to use a cutoff value different from 0.5, if the problem requires such treatment (Why?)
  - Sometimes the cost of misclassification for C_1 is different from that of C_2
  - If it is okay to allow some misclassification for C_2 at the cost of classifying C_1 correctly, a cutoff < 0.5 will be used

Prediction accuracy measures for classification problem: confusion matrix

- Consider a two-class case with classes C_1 and C_2 (eg. defaulter/non-defaulter)
- Estimated misclassification rate: $err = (FP + FN)/(TP + FP + FN + TN)$
- Overall accuracy: $accuracy = 1 - err$

		Actual Class C_1	Actual Class C_2
Predicted class	C_1	TP = number of C_1 records classified correctly (True Positive)	FP = number of C_2 records classified incorrectly as C_1 (False Positive)
	C_2	FN = number of C_1 records classified incorrectly as C_2 (False Negative)	TN = number of C_2 records classified correctly (True Negative)

Prediction accuracy measures for classification problem: confusion matrix

- Balanced data

		Actual Class C_1	Actual Class C_2
Predicted class	C_1	45	5
	C_2	5	45

- Unbalanced data

		Actual Class C_1	Actual Class C_2
Predicted class	C_1	5	5
	C_2	5	85

- Suppose C_1 is the class of importance
  - $\mathit{Sensitivity} = TP/(TP + FN)$ (ability to detect the important class)
  - $\mathit{Specificity} = TN/(FP + TN)$ (ability to rule out the unimportant class)

Prediction accuracy measures for classification problem: confusion matrix

- Consider a two-class case with classes C_1 and C_2 (C_1 is the class of importance)
- $Error = (FP + FN)/(TP + FP + FN + TN)$
- $accuracy = 1 - error$
- $Sensitivity = TP/(TP + FN)$ (ability to detect the important class)
- $Specificity = TN/(FP + TN)$ (ability to rule out the unimportant class)

		Actual Class C_1	Actual Class C_2
Predicted class	C_1	TP = number of C_1 records classified correctly (True Positive)	FP = number of C_2 records classified incorrectly as C_1 (False Positive)
	C_2	FN = number of C_1 records classified incorrectly as C_2 (False Negative)	TN = number of C_2 records classified correctly (True Negative)
		Sensitivity = $\frac{TP}{TP+FN}$	Specificity = $\frac{TN}{FP+TN}$

Prediction accuracy measures for classification problem: Cohen's kappa coefficient (κ)

- Cohen's kappa measures the agreement between the actual and predicted class
- $\kappa = \frac{p_o - p_e}{1 - p_e}$, where p_o is the observed accuracy (*i.e.* agreement) and p_e is the expected accuracy
- Observed accuracy: $p_o = (TP + TN)/(TP + FP + FN + TN)$
- Expected accuracy: $p_e = p_{C_1} + p_{C_2}$ (sum of the probabilities of the agreement for classes C_1 and C_2)
  - p_{C_1} = Prob(actual class is C_1) $\times$ Prob(predicted class is C_1) = $\frac{TP+FN}{TP+FP+FN+TN} \times \frac{TP+FP}{TP+FP+FN+TN}$
  - p_{C_2} = Prob(actual class is C_2) $\times$ Prob(predicted class is C_2) = $\frac{FP+TN}{TP+FP+FN+TN} \times \frac{FN+TN}{TP+FP+FN+TN}$

Prediction accuracy measures for classification problem: Cohen's kappa coefficient (κ)

κ value	interpretation
0.00-0.20	slight agreement
0.21-0.40	fair agreement
0.41-0.60	moderate agreement
0.61-0.80	substantial agreement
0.81-1.00	almost perfect agreement
negative value	agreement worse than expected

Prediction accuracy measures for classification problem: Cohen's kappa coefficient (κ)

		Actual C_1	Actual C_2
Predicted	C_1	45	5
	C_2	5	45

		Actual C_1	Actual C_2
Predicted	C_1	5	5
	C_2	5	85

Table: (a) Balanced data, (b) Unbalanced data

(a) Balanced data

- Misclassification rate $= \frac{FP+FN}{TP+FP+FN+TN} = \frac{5+5}{100} = 0.1$
- Accuracy $= \frac{TP+TN}{TP+FP+FN+TN} = \frac{45+45}{100} = 0.9$
- Sensitivity $= \frac{TP}{TP+FN} = \frac{45}{45+5} = 0.9$
- Specificity $= \frac{TN}{FP+TN} = \frac{45}{45+5} = 0.9$
- p_{C_1} = Prob(actual class is C_1) $\times$ Prob(predicted class is C_1) = $\frac{50}{100} \times \frac{50}{100} = 0.25$
- p_{C_2} = Prob(actual class is C_2) $\times$ Prob(predicted class is C_2) = $\frac{50}{100} \times \frac{50}{100} = 0.25$
- Expected Accuracy $= p_{C_1} + p_{C_2} = 0.25 + 0.25 = 0.5$
- Kappa $= \kappa = \frac{p_o - p_e}{1 - p_e} = \frac{0.9-0.5}{1-0.5} = 0.8$

Prediction accuracy measures for classification problem: Cohen's kappa coefficient (κ)

		Actual C_1	Actual C_2
Predicted	C_1	45	5
	C_2	5	45

		Actual C_1	Actual C_2
Predicted	C_1	5	5
	C_2	5	85

Table: (a) Balanced data, (b) Unbalanced data

(b) Unbalanced data

- Misclassification rate $= \frac{FP+FN}{TP+FP+FN+TN} = \frac{5+5}{100} = 0.1$
- Accuracy $= \frac{TP+TN}{TP+FP+FN+TN} = \frac{45+45}{100} = 0.9$
- Sensitivity $= \frac{TP}{TP+FN} = \frac{5}{5+5} = 0.5$
- Specificity $= \frac{TN}{FP+TN} = \frac{85}{85+5} = 0.94$
- p_{C_1} = Prob(actual class is C_1) $\times$ Prob(predicted class is C_1) = $\frac{10}{100} \times \frac{10}{100} = 0.01$
- p_{C_2} = Prob(actual class is C_2) $\times$ Prob(predicted class is C_2) = $\frac{90}{100} \times \frac{90}{100} = 0.81$
- Expected Accuracy $= p_{C_1} + p_{C_2} = 0.01 + 0.81 = 0.82$
- Kappa $= \kappa = \frac{p_o - p_e}{1 - p_e} = \frac{0.9-0.82}{1-0.82} = 0.44$

Prediction accuracy measures for classification problem: confusion matrix

eBayAuctions data

```
logistic.model <- glm(Competitive ~ ., family=binomial(link='logit'),data=training.data)
install.packages('caret', dependencies = TRUE)
library(caret)
# Confusion matrix for training data
pred.prob.train <- logistic.model$fitted.values
pred.y.train <- ifelse(pred.prob.train > 0.5, 1, 0) # using cutoff = 0.5
> confusionMatrix(as.factor(pred.y.train), as.factor(training.data$Competitive),
                positive = "1")
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0 544 210
         1  76 550
               Accuracy : 0.7928
                 95% CI : (0.7704, 0.8139)
    No Information Rate : 0.5507
    P-Value [Acc > NIR] : < 2.2e-16
                  Kappa : 0.5894
 Mcnemar's Test P-Value : 3.707e-15
            Sensitivity : 0.7237
            Specificity : 0.8774
         Pos Pred Value : 0.8786
         Neg Pred Value : 0.7215
             Prevalence : 0.5507
         Detection Rate : 0.3986
   Detection Prevalence : 0.4536
      Balanced Accuracy : 0.8006
       'Positive' Class : 1
```

Prediction accuracy measures for classification problem: confusion matrix

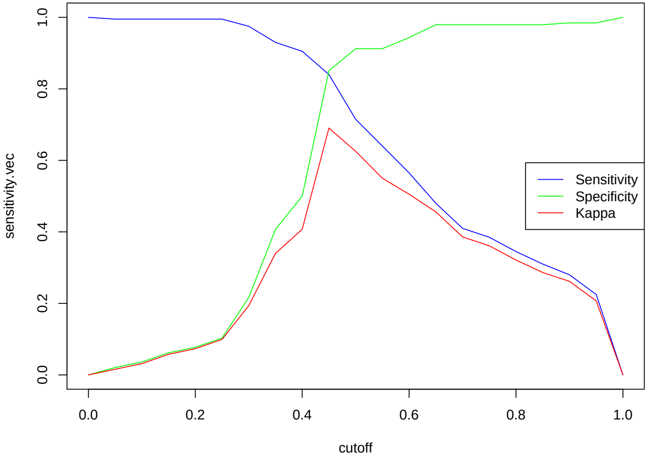
eBayAuctions data

```
# Confusion matrix for test data
pred.prob.test <- predict(logistic.model, newdata = test.data,type = "response")
pred.y.test <- ifelse(pred.prob.test > 0.5, 1, 0) # using cutoff = 0.5
> confusionMatrix(as.factor(pred.y.test), as.factor(test.data$Competitive),
                positive = "1")
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 259  96
         1  27 210
               Accuracy : 0.7922
                 95% CI : (0.7573, 0.8242)
    No Information Rate : 0.5169
    P-Value [Acc > NIR] : < 2.2e-16
                  Kappa : 0.5872

 Mcnemar's Test P-Value : 8.713e-10

            Sensitivity : 0.6863
            Specificity : 0.9056
         Pos Pred Value : 0.8861
         Neg Pred Value : 0.7296
             Prevalence : 0.5169
         Detection Rate : 0.3547
   Detection Prevalence : 0.4003
      Balanced Accuracy : 0.7959
       'Positive' Class : 1
```

Prediction accuracy measures for classification problem: application of different cut-off values

- partition data into training, validation and test set.
- fit model on training data.
- use trained model to make prediction on validation data.
- use validation data for choosing the cut-off value.
- get the final model by fitting the model with the appropriate cut-off value on the combined training and validation data.
- use test data for evaluating model performance with the chosen cut-off value.

Prediction accuracy measures for classification problem: application of different cut-off values



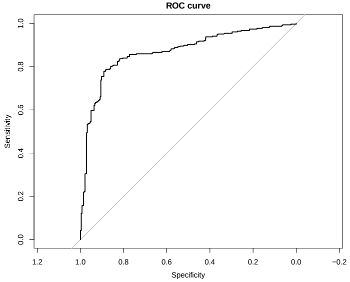
Prediction accuracy measures for classification problem: application of different cut-off values

```
# Evaluate performance on test data
cpred.prob.test <- predict(clogistic.model.final, newdata = ctest.data,type = "response")
cpred.y.test <- ifelse(cpred.prob.test > 0.45, 1, 0) # using cutoff = 0.5
> confusionMatrix(as.factor(cpred.y.test), as.factor(ctest.data$Competitive),
                      positive = "1")
Confusion Matrix and Statistics
          Reference
Prediction   0   1
         0 164  45
         1  27 159
               Accuracy : 0.8177
                 95% CI : (0.776, 0.8546)
    No Information Rate : 0.5165
    P-Value [Acc > NIR] : < 2e-16
                  Kappa : 0.6361
 Mcnemar's Test P-Value : 0.04513
            Sensitivity : 0.7794
            Specificity : 0.8586
         Pos Pred Value : 0.8548
         Neg Pred Value : 0.7847
             Prevalence : 0.5165
         Detection Rate : 0.4025
   Detection Prevalence : 0.4709
      Balanced Accuracy : 0.8190
       'Positive' Class : 1
```

Prediction accuracy measures for classification problem: ROC curve

- Plot specificity vs. sensitivity as the cutoff value descends from 1 to 0
- Better performance is reflected by curves that are closer to the top-left corner
- Note: in the absense of any predictor information, it will be intuitive to assign a new observation to the majority class (similar to estimating the response for a new observation with $\bar{y}$ in the absense of a predictive model). This is called *naive rule*.
- The diagonal (comparison curve) reflects the performance of naive rule for varying cutoff values
- Classifier performance may be measured by Area Under Curve (AUC) which ranges from 1 (perfect discrimination between classes) to 0.5 (naive rule).

```
# Create ROC
library(pROC)
# use this for getting ROC curve on training data
r <- roc(training.data$Competitive, logistic.model$fitted
# use this for getting ROC curve on test data
r <- roc(test.data$Competitive, pred.prob.test)
plot.roc(r, main = "ROC curve")

#compute Area under curve
> auc(r)
Area under the curve: 0.8702
```



Prediction accuracy measures for classification problem: Lift chart

- Objective of the predictive model to produce a score such that when rank ordered by the score the top deciles of the population will have as much members of the target class as possible
- Better performance is reflected by curves that are closer to the top-left corner
- The diagonal (comparison curve) reflects the performance of naive rule
- In the following example, top 30% cases (177 out of 592) captures 53.9% of total competitive.

```
# Create Lift chart
library(gains)
# use this for getting Lift chart on training data
gain <- gains(training.data$Competitive, logistic.model$f
# use this for getting Lift chart on test data
gain <- gains(test.data$Competitive, pred.prob.test)
# Plot Lift chart: Percent cumulative response
x <- c(0, gain$depth)
pred.y <- c(0, gain$cume.pct.of.total)
avg.y <- c(0, gain$depth/100)
plot(x, pred.y, main = "Cumulative Lift Chart", xlab = "d
     ylab = "Percent cumulative response", type = "l", co
lines(x, avg.y, type = "l")
```