Towards Understanding the AUC

The Biased Coins Problem

Mehdi Hakim-Hashemi

2023-11-15

Contents

1 Simple ROC

A Classification Problem

A physician assigns the following probabilities to 10 patients

$$U = \{0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89\}$$

where each one is the probability (propensity) that a certain patient will get the flu in the fall season. The following table shows who got and who did not get the flu after the season was over.

Propensity	Got Flu
0.13	No
0.14	No
0.21	No
0.34	No
0.42	Yes
0.55	No
0.63	No
0.68	Yes
0.74	Yes
0.89	Yes

What does a Classifier Do

A classifier uses a threshold to classify the patients as **Will Get the Flu**, and **Will not Get the Flu** before the flu season. After the season is over we inspect the classifier's performance using the following terminology.

- **True Positive TP**: These are the patients who are classified as **Will Not Get the Flu** and did not get it.
- **True Negative TN**: These are the patients who are classified as **Will Get the Flu** and got it.
- **False Positive FP**: These are the patients who are classified as **Will Not Get the Flu** and got it.
- **False Negative FN**: These are the patients who are classified as **Will Get the Flu** and did not get it.

Note: These definitions might be counter-intuitive but there is a reason we define them like this that will become apparent when we define TPR and FPR as CDF's of certain random variables.

1.1 The Confusion Table

The Confusion Table

We examine several thresholds and gether the data in a table known as **The Confusion Table**. The ralated calculations used in ROC plot are

$$\text{True Positive Rate TPR aka Sensitivity } = \frac{TP}{TP + FN}$$

$$\text{False Positive Rate FPR} = \frac{FP}{FP + TN}$$

$$\text{True Negative Rate aka Specificity} = \frac{TN}{FP + TN} = 1 - \text{FPR}$$

		Actual Outcome	
		Positive	Negative
Predicted	Positive	TP True Positive	FP False Positive
	Negative	FN False Negative	TN True Negative

The Confusion Table

1.2 Sensitivity and Specificity as Functions of the Threshold

Sensitivity and Specificity as functions of the Threshold

Both **Sensitivity (TPR) and Specifity (1 - FPR)** are functions of the threshold t. Given a (discrete) dataset both TPR and FPR are piecewise linear functions of t. Let us denote them by $F_0(t)$ and $F_1(t)$, where we assume that true positives are members of class 0 that are classified correctly. The following two functions calulate $F_0(t)$ and $F_1(t)$ using the following trick. The actual labels and the predicted ones are either 0 or 1. Placing the labels and the predictions side by side we have the following four situations.

Actual	Predicted	As a Binary Number	Base 10	Verdict	
0	0	00	0	True Positive	TP
0	1	01	1	False Negative	FN
1	0	10	2	False Positive	FP
1	1	11	3	True Negative	TN

Let AL be the **Actual Labels** and PL be the **Predicted Labels**, both column matrices of the same size. The $|AL| \times 1$ column matrix

$$\text{DM} = 2\text{AL} + \text{PL}$$

consists of number from 0 to 3. Call it the **Decision Matrix = DM**. Then

$$\text{TPR} = \frac{TP}{TP + FN} = \frac{\#\text{of 0's in the DM}}{\#\text{of 0's in the DM} \ + \ \#\text{of 1's in the DM}}$$

and

$$\text{FPR} = \frac{FP}{FP + TN} = \frac{\#\text{of 2's in the DM}}{\#\text{of 2's in the DM} \ + \ \#\text{of 3's in the DM}}$$

```
#
# This function calculates the SENSITIVITY
# that is TPR for a given data
# Input the labels (0's and 1's)
# The first term in tpr is 1 which is for t = 0
# The rest are tpr for values of t between
# two consecutive terms in the original data
# Note: We don't need the original data - JUST the labels
#
tpr <- function(data_labels){
  n <- length(data_labels)
  tpr <- rep(0, n)
  a <- matrix(1, n, n)
  # Prediction Matrix
  a[lower.tri(a)] <- 0
  labels_matrix <- matrix(data_labels, n, n, byrow = TRUE)
  decision_matrix <- 2*labels_matrix + a
  for (i in 1 : n){
    row_i <- decision_matrix[i,]
    tpr[i] <-
      length(which(row_i==0))/(length(which(row_i==0)) + length(which(row_i==1)))
  }
  #tpr <- c(tpr, 0)
  tpr
}
```

```
#
# This function calculates the 1 - SPECIFICITY
# that is FPR for a given data
# Input the labels (0's and 1's)
# The last term in fpr is 1 which is for t = 1
# The rest are fpr for values of t between
# two consecutive terms in the original data
# Note: We don't need the original dataset - JUST the labels
#
fpr <- function(data_labels){
  n <- length(data_labels)
  fpr <- rep(0, n)
  a <- matrix(1, n, n)
  # Prediction Matrix
  a[lower.tri(a)] <- 0
  labels_matrix <- matrix(data_labels, n, n, byrow = TRUE)
  decision_matrix <- 2*labels_matrix + a
  for (i in 1 : n){
    row_i <- decision_matrix[i,]
    fpr[i] <-
      length(which(row_i==2))/(length(which(row_i==2)) + length(which(row_i==3)))
  }
  fpr
}
```

1.3 Example of ROC Curve and AUC

Example

Example 1 *Consider the data set*

$$U = \{0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89\}$$

with labels

$$Labels = \{0, 0, 0, 0, 1, 0, 0, 1, 1, 1\}$$

The TPR and FPR are

$$F_0(t) = \begin{cases} 0 & t \in [0, 0.13) \\ 0.167 & t \in [0.13, 0.14) \\ 0.333 & t \in [0.14, 0.21) \\ 0.500 & t \in [0.21, 0.34) \\ 0.667 & t \in [0.34, 0.55) \\ 0.833 & t \in [0.55, 0.63) \\ 1 & t \in [0.63, 1] \end{cases} \qquad \textit{and} \qquad F_1(t) = \begin{cases} 0 & t \in [0, 0.42) \\ 0.25 & t \in [0.42, 0.68) \\ 0.50 & t \in [0.68, 0.74) \\ 0.75 & t \in [0.74, 0.89) \\ 1 & t \in [0.89, 1] \end{cases}$$

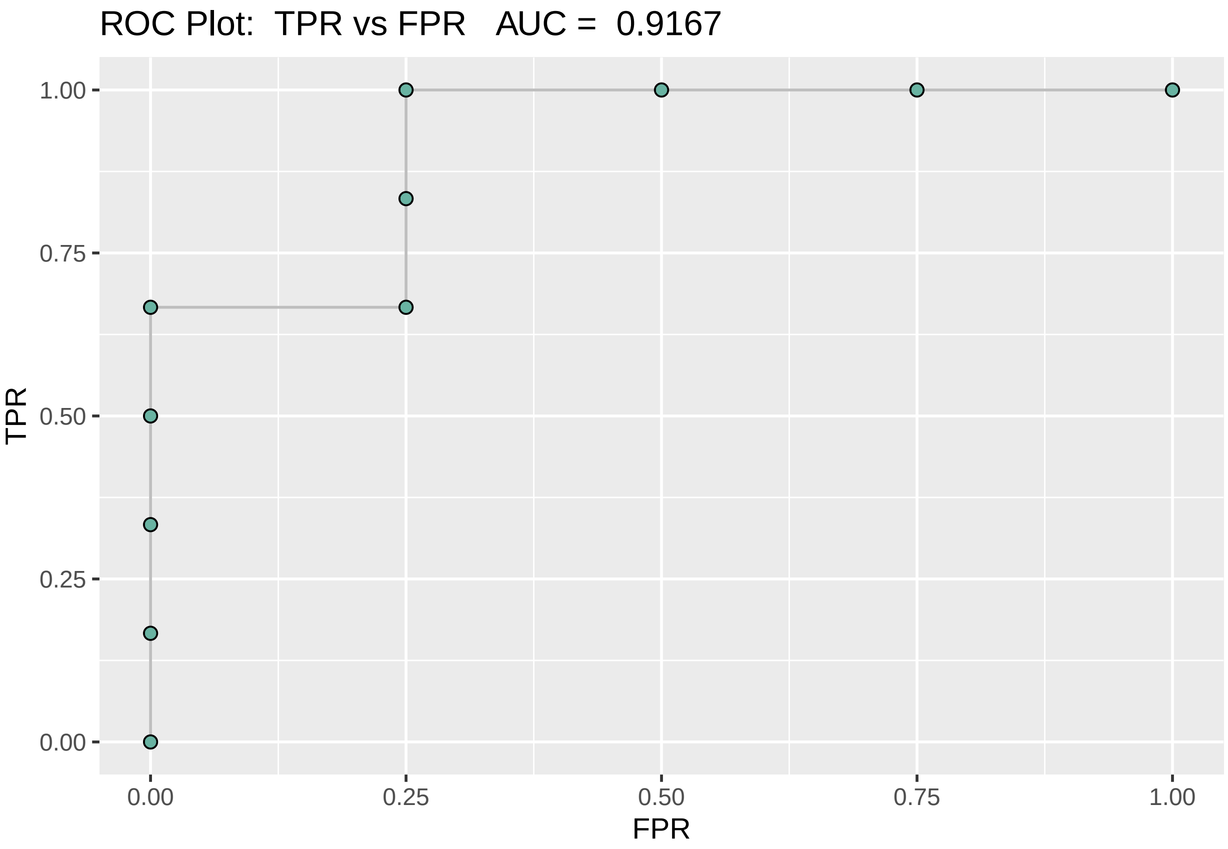
```r
# Calculating TPR and FPR for the above example
data <- c(0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89)
#
# Reasonable AUC
#
data_labels <- c(0, 0, 0, 0, 1, 0, 0, 1, 1, 1)
# data_labels <- sample(0:1, 50, replace=TRUE)
#
# Low AUC
#
# data_labels <- c(0,0,0,0,0,0,1,0,0,0)
#
# TPR
#
data_tpr <- tpr(data_labels)
data_and_tpr_array <- c(data, data_tpr)
tpr_and_data <- matrix(data_and_tpr_array, 2, byrow =TRUE)
# print(tpr_and_data)
#
# FPR
#
data_fpr <- fpr(data_labels)
data_and_fpr_array <- c(data, data_fpr)
fpr_and_data <- matrix(data_and_fpr_array, 2, byrow =TRUE)
# print(fpr_and_data)
```

```r
#
# ROC Graph and AUC
#
data_fpr <- c(data_fpr, 1)
data_tpr <- c(data_tpr, 1)
sensitivity <- data_tpr
```

```
specificity <- 1 - data_fpr
#
nn <- length(data_tpr) - 1
#
# Calculating AUC
# We use the Trapezoidal Rule to find the integral
# AUC = int_0^1 F_0(t) F_1'(t) dt given below
#
AUC <- 0
for (i in 1 : nn){
  AUC <- AUC + (data_tpr[i+1] + data_tpr[i])*(data_fpr[i+1] - data_fpr[i])/2
}
print(c("AUC = ", AUC))
```

```
## [1] "AUC = "            "0.916666666666667"
```

```
ROC_Data <- data.frame("FPR"=data_fpr, "TPR"=data_tpr)
# print(ROC_Data)
roc_title <- paste("ROC Plot:  TPR vs FPR   ")
auc_in_title <- paste("AUC = ", round(AUC, digits = 4))
ggplot(ROC_Data, aes(x=FPR, y=TPR)) +
    geom_line( color="grey") +
    geom_point(shape=21, color="black", fill="#69b3a2", size=2) +
  ggtitle(paste0(roc_title, auc_in_title))
```



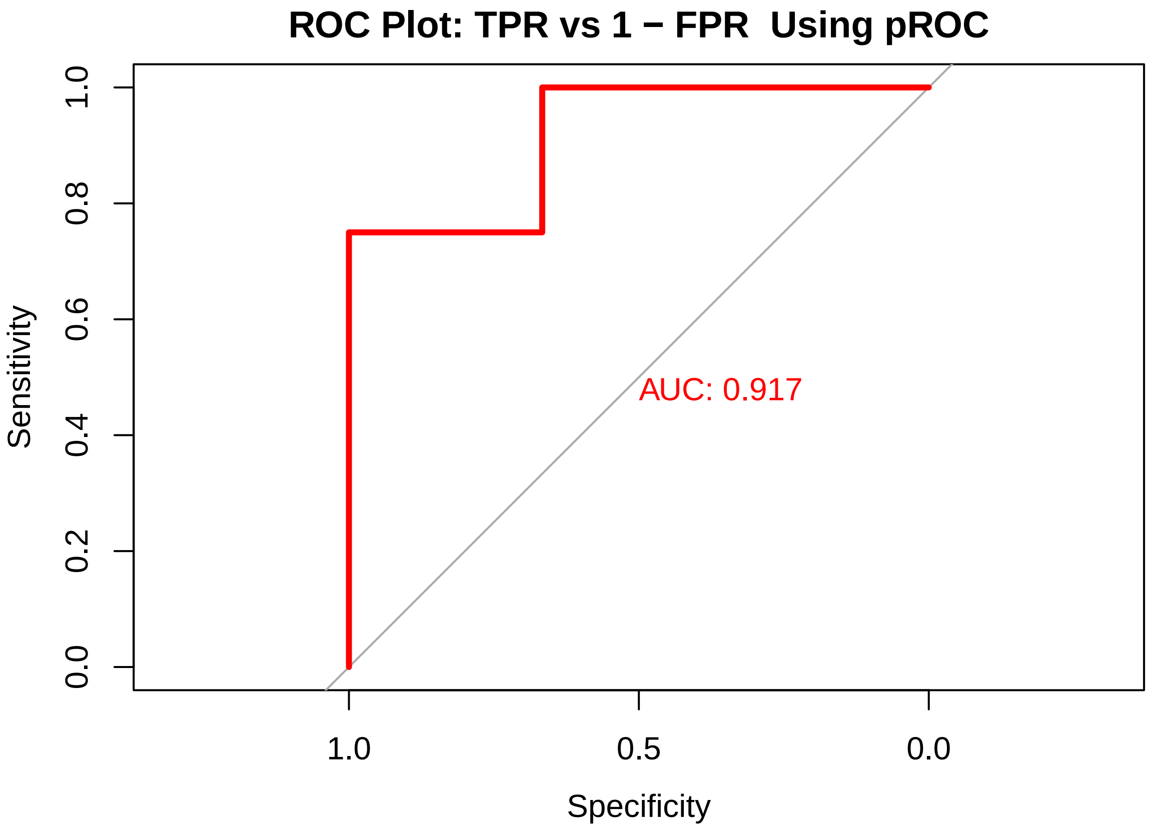
2 Using the pROC Package for the Above Example

pROC Package

pROC package in an R package that plots the ROC curce and calculate the AUC. The inputs are the **labels** and the **(ordered) data**. It should be noted that the data does not matter. As long as we have have an ordered dataset with the same number of data points we will get the same result. As we observe pROC results in the same ROC curve and the same AUC. This is illustrated in the following R codes which plots $1 - \text{FPR}$ vs TPR. One must note that 0 and 1 classes in pROC are different from 0 and 1 classes in our approach to ROC.

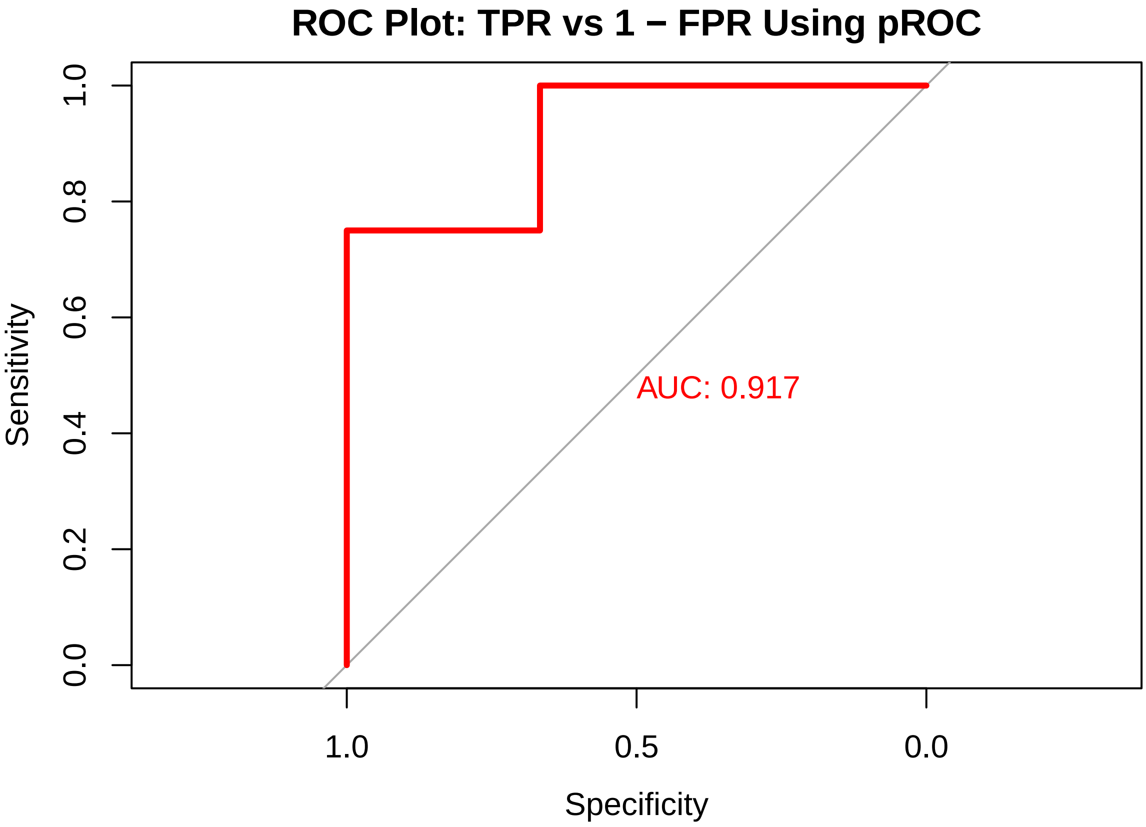
```r
#
# Using the pROC Package with the original data
#
labels <- c(0,0,0,0,1,0,0,1,1,1)
data <- c(0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89)
plot(roc(labels, data, direction="<"),
     col="red", lwd=3, main="ROC Plot: TPR vs 1 - FPR  Using pROC",print.auc=TRUE)
```

```
## Setting levels: control = 0, case = 1
```



```r
#
# Using the pROC Package using a different
# ordered dataset with the same number of data points
#
labels <- c(0,0,0,0,1,0,0,1,1,1)
data <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
plot(roc(labels, data, direction="<"),
    col="red", lwd=3, main="ROC Plot: TPR vs 1 - FPR Using pROC", print.auc=TRUE)
```

```
## Setting levels: control = 0, case = 1
```



3 Explicit Formula for AUC

AUC: Definition 1

Definition 1 *$F_0(t)$ and $F_1(t)$, introduced in the previous sections, can be veiwed as CDF's if we extend t to all of $\mathbb{R}$ by letting $F_i(t) = 0$ for $t < 0$ and $F_i(t) = 1$ for $t > 1$. AUC is now the foillowing integral*

$$AUC = \int_{[0,1]} F_0(t)\ dF_1(t)$$

where CDF's $F_i(t)$ are treated as mesures on $\mathbb{R}$. In other words AUC is the area under the TPR-FPR graph. If $F_1(t)$ has a PDF (well defined derivative) $f_1(t)$ then

$$AUC = \int_0^1 F_0(t)\ dF_1(t) = \int_0^1 F_0(t)F_1'(t)\ dt = \int_0^1 F_0(t)f_1(t)\ dt$$

In our case, where $F_0(t)$ and $F_1(t)$ are piecewise linear, the derivatives are simply the jumps at every $x_i \in S$, the dataset. See example 1.

AUC: Definition 2

Definition 2 *Consider the population*

$$C = \{c_i\}_{i=1}^{n}$$

and probabilities

$$S = \{x_i\}_{i=1}^{n} \qquad \text{where} \qquad \begin{cases} P(c_i = H) = x_i \\ P(c_i = T) = 1 - x_i \end{cases}$$

and S is sorted (ascending). Suppose a labeling system labels the population and puts them into two categories, D_0 and D_1.

$$D_0 = \{d_1, \cdots, d_{n_0}\} \qquad \text{and} \qquad D_1 = \{d_{n_0+1}, \cdots, d_n\} \qquad \text{with} \qquad D_0 \cup D_1 = C$$

with corresponding probabilities

$$S_0 = \{\hat{x}_j\}_{j=1}^{n_0} \qquad \text{and} \qquad S_1 = \{\hat{x}_k\}_{k=n_0+1}^{n} \qquad \text{and} \qquad S_0 \cup S_1 = S$$

Then the AUC is defined in the following way

$$AUC = \frac{\sum_{i=1}^{n_0} \sum_{j=n_0+1}^{n} 1_{(\hat{x}_i < \hat{x_j})}}{n_0 \times (n - n_0)}$$

where

$$1_E = \begin{cases} 1 & \text{if } E \text{ is TRUE} \\ 0 & \text{if } E \text{ is FALSE} \end{cases}$$

Example Using Definition 2 of AUC

Example 2 *In example 1 we calculated $AUC = 0.9167$. In this example we use the summation definition given above to calculate AUC. labels and probabilities are*

$$Labels = \{0, 0, 0, 0, 1, 0, 0, 1, 1, 1\}$$

$$S = \{0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89\}$$

Which gives the two categories

$$S_0 = \{0.13, 0.14, 0.21, 0.34, 0.55, 0.63\} \qquad \text{and} \qquad S_1 = \{0.42, 0.68, 0.74, 0.89\}$$

Then

$$AUC = \frac{\sum_{i=1}^{n_0} \sum_{j=n_0+1}^{n} 1_{(\hat{x}_i < \hat{x_j})}}{n_0 \times (n - n_0)} = \frac{22}{(6 \times 4)} = 0.9167$$

Which is the same as the AUC calculated in example 1.

4 Probabilistic Interpretation of AUC

Probabilistic Interpretation of AUC

Both $F_0(t)$ and $F_1(t)$ can be viewed as CDF's. In fact we can extend their definition to include values below zero and above 1. This way both satisfy conditions for being a CDF. If X_0 and X_1 are the random variebles they represent then both take values in $I = [0,1]$. To define them explicitly we need the following setup: Let $S = S_0 \cup S_1$ be as defined in definition 3 and let $J_k = [0, \hat{x}_k) \subseteq I, \hat{x}_k \in S$. Then for $t \in [\hat{x}_{k-1}, \hat{x}_k)$

$$t \in [\hat{x}_{k-1}, \hat{x}_k) \implies \begin{cases} F_0(t) = P(X_0 \le t) = \frac{|J_k \cap S_0|}{|S_0|} \\ \\ F_1(t) = P(X_1 \le t) = \frac{|J_k \cap S_1|}{|S_1|} \end{cases} \qquad \text{and} \qquad F_0(1) = F_1(1) = 1$$

Both F_0 and F_1 are piecewise continuous functions. Let $f_i(t) = F_i'(t), i = 1,2$ be piecewise derivatives. Then AUC is

$$\text{AUC} = \int_0^1 F_0(t) F_1'(t)\ dt = \int_0^1 F_0(t) f_1(t)\ dt = \int_0^1 f_1(t) \left[\int_0^t f_0(t')\ dt' \right]\ dt$$

Where we used the Fundamental Theorem of Calculus to convert $F_0(t)$ to the inside integral. The bounds for the inside integral implies that $t' \le t$. Hence the double integral becomes

$$\int_0^1 \int_0^1 1_{(t' \le t)} f_0(t') f_1(t)\ dt\ dt'$$

This can be written as probability of certain event, i.e., $\{\omega \mid X_0(\omega) \le X_1(\omega)\}$.

$$\text{AUC} = \int_0^1 \int_0^1 1_{(t' \le t)} f_0(t') f_1(t)\ dt\ dt' = P(X_0 \le X_1)$$

In other words AUC is the probability that a classifier ranks an instance of class 0 lower than an instance of class 1. I.e., a **measure** of the classifier's performance.

5 Cost Function

Total Probability

Given classes S_0 and S_1 (as in example 3) let π_0 and π_1 be the probabilities that an element of $S = S_0 \cup S_1$ belongs to S_0 or S_1 respectively. Let

$$F(t) = \pi_0 F_0(t) + \pi_1 F_1(t) \qquad t \in \mathbb{R}$$

$F(t)$ can be viewed as a CDF too. Let X be the random variable attached to it then

$$P(X \le t) = \pi_0 F(X_0 \le t) + \pi_1 P(X_1 \le t) \qquad t \in \mathbb{R}$$

The corresponding PDF is

$$f(t) = \pi_0 f_0(t) + \pi_1 f_1(t) \qquad t \in [0,1]$$

5.1 An Important Theorem about $f(t)$

An Impoprtant Theorem about $f(t)$

Theorem 1 *For a discrete dataset $S = \{x_1, x_2, \cdots, x_N\}$ with N elements $f(t)$ will be a uniform distribution on S. I.e.,*

$$f(t) = \frac{1}{N} \qquad \forall t \in S \qquad \textit{and zero otherwise}$$

Proof: $f(t)$ is discrete with values concentrayed at x_i's. Let $x_i \in S$ with $i \geq 2$. Then

$$f_0(x_i) = F_0(x_i) - F_0(x_{i-1}) \quad \text{and} \quad f_1(x_i) = F_1(x_i) - F_1(x_{i-1})$$

Let

$$\begin{cases} F_0(x_{i-1}) = \dfrac{a}{a+b} & a = \text{TP} \quad \text{and} \quad b = \text{FN} \\ \\ F_1(x_{i-1}) = \dfrac{c}{c+d} & c = \text{FP} \quad \text{and} \quad d = \text{TN} \end{cases}$$

With these notations we also have

$$\pi_0 = \frac{a+b}{N} \quad \text{and} \quad \pi_1 = \frac{c+d}{N}$$

When $t = x_i$ the predicted label is 0 and the actual label can be 0 or 1. We check both cases

- **Actual Value is 0:** This situation doesn't change FPR, i.e, $F_1(t)$. But $F_0(t)$ changes

$$\begin{cases} t = x_i \\ \text{actual label} = 0 \end{cases} \implies \begin{cases} F_0(x_i) = \dfrac{a+1}{a+1+b-1} = \dfrac{a+1}{a+b} \\ \\ F_1(x_i) = \dfrac{c}{c+d} \end{cases}$$

  Therefore

$$f(x_i) = \pi_0 f_0(x_i) + \pi_1 f_1(x_i) = \left(\frac{a+b}{N}\right)\left[\frac{a+1}{a+b} - \frac{a}{a+b}\right] + 0 = \left(\frac{a+b}{N}\right)\left(\frac{1}{a+b}\right) = \frac{1}{N}$$

- **Actual Value is 1:** This situation doesn't change TPR, i.e., $F_0(t)$. But $F_1(t)$ changes

$$\begin{cases} t = x_i \\ \text{actual label} = 1 \end{cases} \implies \begin{cases} F_0(x_i) = \dfrac{a}{a+b} \\ \\ F_1(x_i) = \dfrac{c+1}{c+1+d-1} = \dfrac{c+1}{c+d} \end{cases}$$

  Therefore

$$f(x_i) = \pi_0 f_0(x_i) + \pi_1 f_1(x_i) = 0 + \left(\frac{c+d}{N}\right)\left[\frac{c+1}{c+d} - \frac{c}{c+d}\right] = \left(\frac{c+d}{N}\right)\left(\frac{1}{c+d}\right) = \frac{1}{N}$$

This completes the proof.

6 The Continuous Case

The Continuous Case

Suppose the probability scores (propensities) are given as a distribution. That is: Let $f(t)$ be the PDF of the scores S with $I = [0,1]$ as its support. Let $C_0 \subseteq I$ be a subset of I with Lebesgue measure π_0. Let $C_1 = I - C_0$ and define a function $\mathbf{e}$ in the following way

$$\mathbf{e}(x) = \begin{cases} 0 & x \in C_0 \\ 1 & x \in C_1 \end{cases}$$

Then $\mathbf{e}$ can be viewed as a **Bernoulli** random variable with parameter $p = \pi_0$. Let $\pi_1 = 1 - \pi_0$ then,

$$\begin{cases} P(\mathbf{e} = 0) = \pi_0 \\ P(\mathbf{e} = 1) = \pi_1 \end{cases}$$

Let $T \in [0,1]$ and define

$$\begin{cases} \text{TP}(T) = P(S < T \ \wedge \ \mathbf{e} = 0) = \displaystyle\int_{[0,T]\cap C_0} f(t)\ dt \\ \\ \text{TN}(T) = P(S \geq T \ \wedge \ \mathbf{e} = 1) = \displaystyle\int_{[T,1]\cap C_1} f(t)\ dt \\ \\ \text{FN}(T) = P(S \geq T \ \wedge \ \mathbf{e} = 0) = \displaystyle\int_{[T,1]\cap C_0} f(t)\ dt \\ \\ \text{FP}(T) = P(S < T \ \wedge \ \mathbf{e} = 1) = \displaystyle\int_{[0,T]\cap C_1} f(t)\ dt \end{cases}$$

Also define

$$\begin{cases} F_0(T) = \dfrac{\text{TP}(T)}{\text{TP}(T) + \text{FN}(T)} = \dfrac{\text{TP}(T)}{\int_{C_0} f(t)\ dt} \\ \\ F_1(T) = \dfrac{\text{TN}(T)}{\text{TN}(T) + \text{FP}(T)} = \dfrac{\text{TN}(T)}{\int_{C_1} f(t)\ dt} \end{cases}$$

Example: Using Uniform$[0,1]$ for the Data

Let us use the same data, i.e,

$$\text{Labels} = \{0,0,0,0,1,0,0,1,1,1\}$$

$$S = \{0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89\}$$

The discrete distribution is

$$f(t) = \frac{1}{10} \qquad \forall t \in S$$

which turns into the Uniform$[0,1]$ with $f(t) = 1$ for $t \in [0,1]$ and zero otherwise. C_0 and C_1 **Can Be**

$$C_0 = [0, 0.34) \cup [0.55, 0.63) \quad \text{and} \quad C_1 = [0.34, 0.55) \cup [0.63, 1]$$

With

$$|C_0| = 0.42 \quad \text{and} \quad |C_1| = 0.58$$

The two functions, F_0 and F_1 are

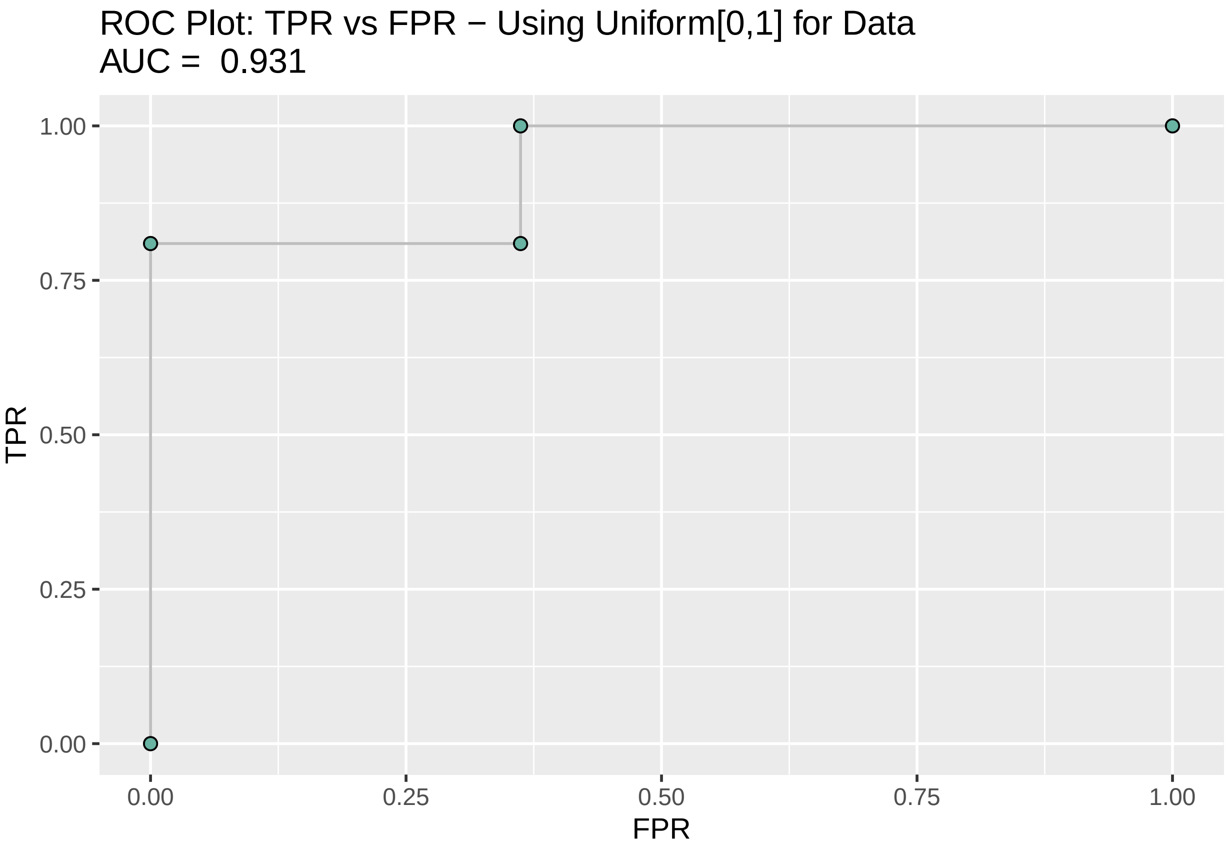
$$F_0(T) = \begin{cases} \frac{T}{0.42} & 0 \le T \le 0.34 \\ \frac{0.34}{0.42} & 0.34 < T \le 0.55 \\ \frac{T-0.21}{0.42} & 0.55 < T \le 0.63 \\ 1 & 0.63 < T \le 1 \end{cases} \quad \text{and} \quad F_1(T) = \begin{cases} 0 & 0 \le T \le 0.34 \\ \frac{T-0.34}{0.58} & 0.34 < T \le 0.55 \\ \frac{0.21}{0.58} & 0.55 < T \le 0.63 \\ \frac{T-0.42}{0.58} & 0.63 < T \le 1 \end{cases}$$

The ROC graph is plotted below and the AUC is

$$\text{AUC} = \left(\frac{0.21}{0.58}\right)\left(\frac{0.34}{0.42}\right) + \left(1 - \frac{0.21}{0.58}\right)(1) = 0.9310345$$

I emphasized on "Can Be" because there are infinitely many different options that give the same labels. For example $[0.34, 0.55] \subseteq C_1$ only includes one point, i.e, $x = 0.42$. Another option can be $[0.4, 0.43]$ which also includes 0.42. The question is: Is there an arrangement of C_0 and C_1 that gives a maximum value for AUC? Or better: Does AUC even have a maximum value?

```r
#
# The ROC curve for the Continuouys Example
#
data_tpr <- c(0,0.34/0.42, 0.34/0.42, 1, 1)
data_fpr <- c(0, 0, 0.21/0.58, 0.21/0.58, 1)
#
ROC_Data <- data.frame("FPR"=data_fpr, "TPR"=data_tpr)
# print(ROC_Data)
AUC <- (0.21/0.58)*(0.34/0.42) + (1 - 0.21/0.58)*(1)
roc_title <- paste("ROC Plot: TPR vs FPR - Using Uniform[0,1] for Data ")
auc_in_title <- paste("AUC = ", round(AUC, digits = 4))
ggplot(ROC_Data, aes(x=FPR, y=TPR)) +
  geom_line( color="grey") +
  geom_point(shape=21, color="black", fill="#69b3a2", size=2) +
 ggtitle(paste0(roc_title, '\n', auc_in_title))
```



Example: Using Beta Distribution for the Data

We can fit a Beta distribution to the data using mean and variance of the data. I.e., if $S \sim \text{Beta}(\alpha, \beta)$ then

$$E[S] = \mu = \frac{\alpha}{\alpha + \beta} \quad \text{and} \quad \text{Var}[S] = \sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

Solving for α and β we get

$$\alpha = \frac{\mu^2(1-\mu)}{\sigma^2} - \mu \quad \text{and} \quad \beta = (1-\mu)\left[\frac{\mu(1-\mu)}{\sigma^2} - 1\right]$$

For the following data set we have

$$S = \{0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89\} \implies \begin{cases} \alpha = 1.933 \\ \beta = 1.330 \end{cases}$$

The graph of $\text{Beta}(1.933, 1.330)$ is given below.

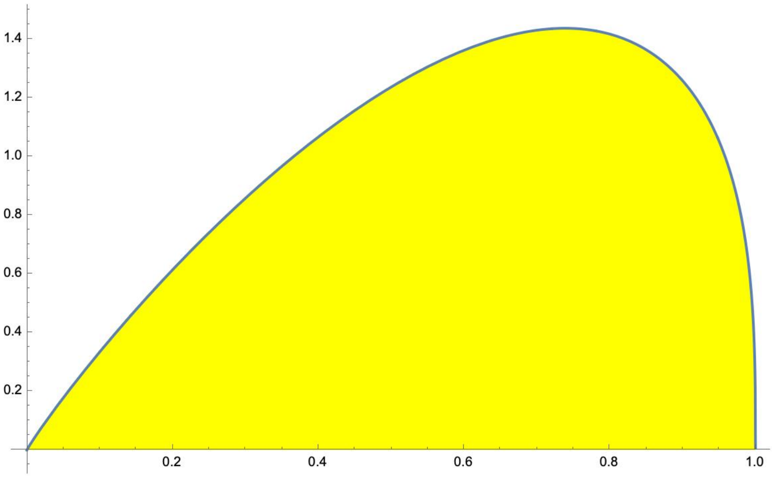


Figure 1: Beta$(1.933, 1.330)$

C_0, C_1, F_0, and F_1

We have

$$C_0 = [0, 0.34] \cup [0.55, 0.63] \quad \text{and} \quad C_1 = [0.34, 0.55] \cup [0.63, 1]$$

Both F_0 and F_1 are calculated using a Boxcar function, which is

$$\text{Box}(t; a, b) = \begin{cases} 1 & t \in [a, b] \\ 0 & \text{otherwise} \end{cases} \implies \text{Box}(t; a, b) = \text{Heaviside}(x, a) - \text{Heaviside}(x, b)$$

with the value of Heaviside function at $x = a$ being zero (not 0.5). Let $f(t)$ be the PDF of Beta(α, β) then

$$F_0(T) = \frac{\int_0^T f(t)[\text{Box}(t, 0, 0.34) + \text{Box}(t, 0.55, 0.63)]\, dt}{\int_0^1 f(t)[\text{Box}(t, 0, 0.34) + \text{Box}(t, 0.55, 0.63)]\, dt}$$

and

$$F_1(T) = \frac{\int_0^T f(t)[\text{Box}(t, 0.34, 0.55) + \text{Box}(t, 0.63, 1)]\, dt}{\int_0^1 f(t)[\text{Box}(t, 0.34, 0.55) + \text{Box}(t, 0.63, 1)]\, dt}$$

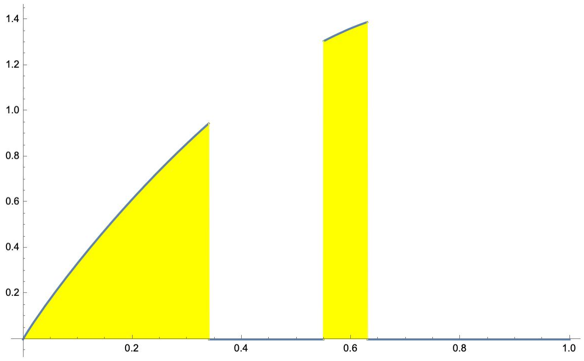


Figure 2: Beta$(1.933, 1.330)[\text{Box}(t, 0, 0.34) + \text{Box}(t, 0.55, 0.63)]$. Used for F_0

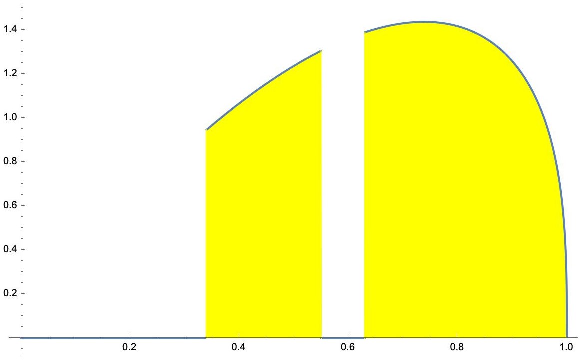


Figure 3: Beta(1.933, 1.330)[Box(t, 0.34, 0.55) + Box(t, 0.63, 1)]. Used for F_1

```
#
# Data
#
S <- c(0.13, 0.14, 0.21, 0.34, 0.42, 0.55, 0.63, 0.68, 0.74, 0.89)
#
mu <- mean(S)
sigma_2 <- var(S)
#
# parameter for the Beta distribution
#
alpha <- mu^2*(1 - mu)/sigma_2 - mu
beta <- (1 - mu)*(mu*(1 - mu)/sigma_2 - 1)
print(paste0("alpha = ", alpha))
```

```
## [1] "alpha = 1.19334733594008"
```

```
print(paste0("beta = ", beta))
```

```
## [1] "beta = 1.32958572101569"
```

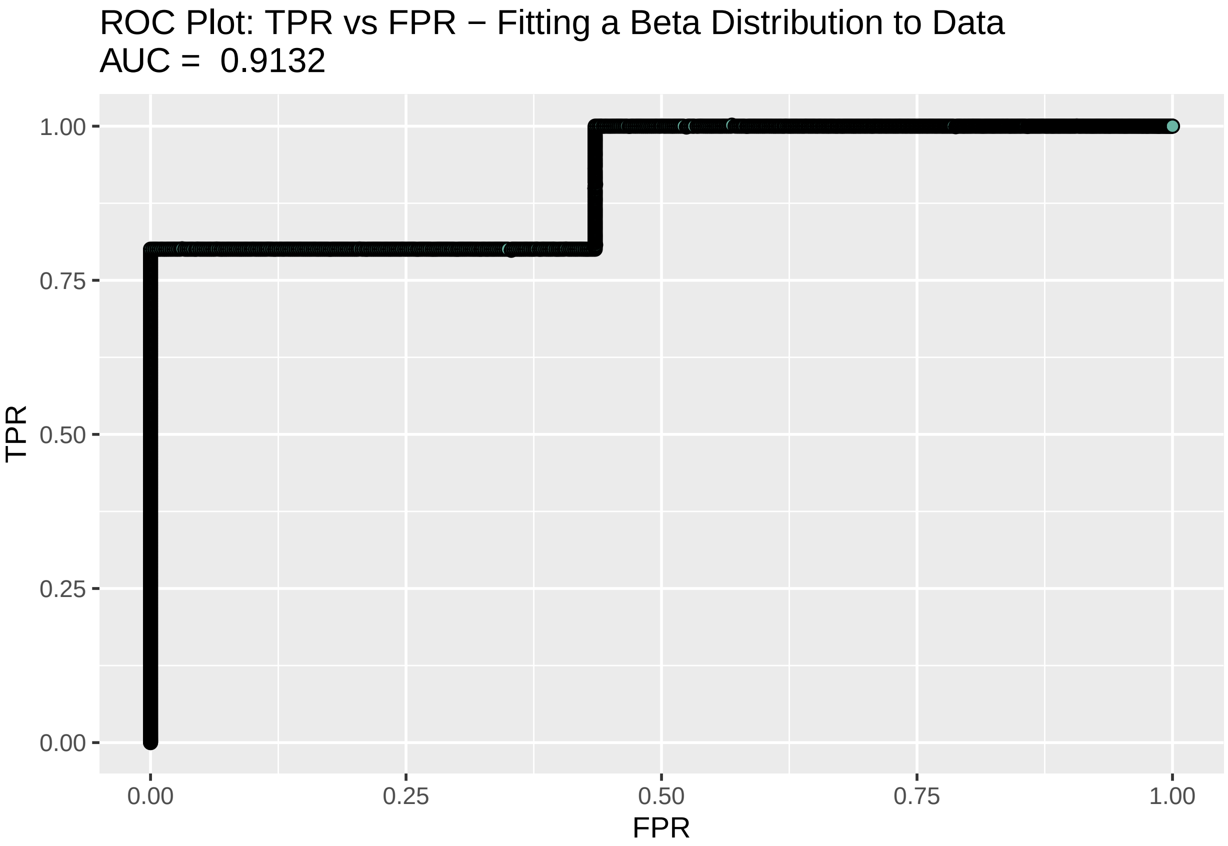
```
#
# Beta function
#
beta_d <- function(x, alpha, beta){
  beta_d <- dbeta(x, alpha, beta)
  beta_d
}
#
# My Heaviside
#
my_heaviside <- function(x, a){
  my_heaviside <- heaviside(x, a)
  my_heaviside[which(my_heaviside == 0.5)] <- 0
  my_heaviside
}
#
# Boxcar Function on [a, b]
# boxcar(x) = 1 for x in [a, b]
# zero otherwise
#
```

```
boxcar <- function(x, Ci){
  boxcar <- my_heaviside(x, Ci[1]) - my_heaviside(x, Ci[2])
  boxcar
}
```

```
#
# Inline Beta Function
#
inline_beta_d <- function(x) beta_d(x, alpha, beta)
#
# Truncated Beta Function For ROC Calculation
#
boxcar_beta <- function(x, C){
  #
  # C a the matrix of intervals
  # over which Beta PDF is itself
  # Otherwise zero
  #
  n_row_C <- nrow(C)
  sum <- 0
  for (i in 1:n_row_C){
      sum <- sum + boxcar(x, C[i,])
  }
  boxcar_beta <- inline_beta_d(x)*sum
}
```

```
#
# Calculating F_0 and F_1
#
# Generate Points
#
dT <- 0.001
x <- seq(0, 1, by = dT)
n <- length(x)
F_0 <- numeric(n)
F_1 <- numeric(n)
C_0 <- matrix(c(0, 0.34, 0.55, 0.63), nrow = 2, byrow = TRUE)
C_1 <- matrix(c(0.34, 0.55, 0.63, 1), nrow = 2, byrow = TRUE)
#
# Define the inline functions to be integrated
#
boxcar_beta_C_0 <- function(x) boxcar_beta(x, C_0)
boxcar_beta_C_1 <- function(x) boxcar_beta(x, C_1)
C_0_denominator <- integrate(boxcar_beta_C_0, 0, 1)$value
C_1_denominator <- integrate(boxcar_beta_C_1, 0, 1)$value
for (i in 1:n){
  F_0[i] <- integrate(boxcar_beta_C_0, 0, x[i])$value/C_0_denominator
  F_1[i] <- integrate(boxcar_beta_C_1, 0, x[i])$value/C_1_denominator
}
ROC_Data <- data.frame("FPR"= F_1, "TPR"= F_0)
#
# AUC
#
nn <- n - 1
```

```r
AUC <- 0
for (i in 1 : nn){
  AUC <- AUC + (F_0[i+1] + F_0[i])*(F_1[i+1] - F_1[i])/2
}
#
# ROC Plot
#
roc_title <- paste("ROC Plot: TPR vs FPR - Fitting a Beta Distribution to Data ")
auc_in_title <- paste("AUC = ", round(AUC, digits = 4))
ggplot(ROC_Data, aes(x=FPR, y=TPR)) +
    geom_line( color="grey") +
    geom_point(shape=21, color="black", fill="#69b3a2", size=2) +
  ggtitle(paste0(roc_title, '\n', auc_in_title))
```



Loss Function

A predictor-response data consists of tuples of the form $(\mathbf{x}_n, y_n)$ where $\mathbf{x}_n$ is a relization of a random vector $\mathbf{X}$ and $y_n \in \{0,1\}$ is a realization of a random variable Y with a Bernoulli distribution with certain parameter. In practice Y is a decision the predictor-response machinery has made about $\mathbf{x}_n$. I.e., $Y = y_n = 0$ means that $\mathbf{x}_n$ belongs to category zero and $Y = y_n = 1$ means that $\mathbf{x}_n$ belongs to category one. Let $q(\mathbf{x})$ be the PDF of a random variable that models the probability

$$\eta(\mathbf{x}) = P(Y = 1 \mid \mathbf{X} = \mathbf{x})$$

The definition implies that the support of $q(\mathbf{x})$ is $[0,1]$. Consider the loss function

$$L(y_n \mid q_n)$$

This is a loss due to our prediction $q_n = a(\mathbf{x}_n)$. I.e., y_n is the real world classification of $\mathbf{x}_n$ and q_n is the probability that $y_n = 1$ given $\mathbf{X} = \mathbf{x}_n$. For example $q_n = 0.25$ means that with probability 0.25 the predictor-response machinery has predicted that x_n belongs to category 1. If we have a sample size of N then the loss will be

$$\mathcal{L}(q) = \frac{1}{N}\sum_{n=1}^{N} L(y_n \mid q_n)$$

Total Cost Due to Miscalculation

This section. and the subsequent ones are from Hand [4]. Total cost due to miscalculation is

$$L(b, c; t) = b\pi_0(1 - F_0(t)) + c\pi_1 F_1(t) \qquad b, c, t \in [0, 1]$$

We have two cases:

- **Case 1:** We assume that all CDF's (F_0 and F_1) and PDF's (f_0 and f_1) are continuous. To calculate the expected value $E[L(b, c; t)]_{f(t)}$ we need the following integrals

$$\int_{-\infty}^{\infty} F_0(t) f_0(t)\ dt \qquad \int_{-\infty}^{\infty} F_0(t) f_1(t)\ dt \qquad \int_{-\infty}^{\infty} F_1(t) f_0(t)\ dt \qquad \int_{-\infty}^{\infty} F_1(t) f_1(t)\ dt$$

  The ROC graph starts at $(0, 0)$ and ends at $(1, 1)$. Consider the loop $(0, 0)-(1, 1)-(1, 0)-(0, 0)$ with ROC graph as one side. Using Green's Theorem for this closed curve we have

$$\int_{-\infty}^{\infty} F_0(t) f_1(t)\ dt + \int_{-\infty}^{\infty} F_1(t) f_0(t)\ dt = 0$$

  which implies

$$\text{AUC} = \int_{-\infty}^{\infty} F_0(t) f_1(t)\ dt = -\int_{-\infty}^{\infty} F_1(t) f_0(t)\ dt$$

  The other two integrals become

$$\int_{-\infty}^{\infty} F_0(t) f_0(t)\ dt = \int_{-\infty}^{\infty} F_0(t) d(F_0(t)) = \frac{1}{2} F_0(t)^2 \Big|_{-\infty}^{\infty} = \frac{1}{2}$$

  and

$$\int_{-\infty}^{\infty} F_1(t) f_1(t)\ dt = \int_{-\infty}^{\infty} F_1(t) d(F_1(t)) = \frac{1}{2} F_1(t)^2 \Big|_{-\infty}^{\infty} = \frac{1}{2}$$

  Therefore

$$\int_{-\infty}^{\infty} F_0(t) f(t)\ dt = \int_{-\infty}^{\infty} F_0(t)[\pi_0 f_0(t) + \pi_1 f_1(t)]\ dt = \frac{1}{2}\pi_0 + \pi_1 \text{AUC}$$

  and

$$\int_{-\infty}^{\infty} F_1(t) f(t)\ dt = \int_{-\infty}^{\infty} F_1(t)[\pi_0 f_0(t) + \pi_1 f_1(t)]\ dt = -\pi_0 \text{AUC} + \frac{1}{2}\pi_1$$

  Hence

$$E[L(b, c; t)]_{f(t)} = b\pi_0 - \frac{1}{2}(b\pi_0^2 + c\pi_1^2) - \pi_0\pi_1(b + c)\text{AUC}$$

- **Case 2:** When the dataset has N elements we have a discrete PDF $f(t)$ given by $f(t) = \frac{1}{N}$ for values in the dataset and zero otherwise. Hence

$$E[L(b, c; t)]_{f(t)} = b\pi_0 + \frac{1}{N}\left[-b\pi_0 \sum_{i=1}^{N} F_0(x_i) + c\pi_1 \sum_{i=1}^{N} F_1(x_i)\right]$$

Minimizing Cost and Minimum Expected Cost

Let the value of $t \in [0,1]$ that minimizes cost be denoted by $T(c_0, c_1)$. Then

$$T(c_0, c_1) = \arg\min_t [c_0\pi_0(1 - F_0(t) + c_1\pi_1 F_1(t)]$$

This cost value is the same for any multiple of the pair(c_0, c_1). Let $k = \frac{c_1}{c_0}$ and transform the pair (c_0, c_1) to $\left(\frac{1}{1+k}, \frac{k}{1+k}\right)$. If we let $c = \frac{1}{1+k}$ then $\frac{k}{1+k} = 1 - c$. This reduces the argument of the above expression to a function of one cost variable $c \in [0,1]$. I.e.,

$$T(c) = \arg\min_t [c\pi_0 F_0(t) + (1-c)\pi_1 F_1(t)]$$

We also assume that the above relationship is one-to-one. I.e., for any c there is only one $T(c)$ that minimizes the argument. Denote the cost for any t by

$$Q(t, c) = c\pi_0 F_0(t) + (1-c)\pi_1 F_1(t)$$

and assume that the cost C has a distribution with PDF $w(c)$. Then the expected minimum cost will be

$$E[Q(T(c), c)]_{w(c)} = \int_0^1 [c\pi_0(1 - F_0(T(c))) + (1-c)\pi_1 F_1(T(c))]w(c)\ dc$$

With the one-to-one assumption between $T(c)$ and c we can change the variables in the above integral to obtain

$$L = E[Q(c)]_{w(c)} = \int_0^1 [c\pi_0(1 - F_0(c)) + (1-c)\pi_1 F_1(c)]w(c)\ dc$$

Hand's H Measure

Hand in [4] defines a measure, denoted by H, by setting

$$H = 1 - \frac{L}{L_{\text{ref}}}$$

Where L_{ref} is a loss that is achieved by setting $F_0 \equiv F_1 = F$ where

$$F(t) = \begin{cases} 0 & t < \pi_1 \\ 1 & t \geq \pi_1 \end{cases}$$

This assumption results in

$$\int_{\pi_1}^1 c\pi_0(1 - F(c))w(c)\ dc = 0 \quad \text{and} \quad \int_0^{\pi_1} (1-c)\pi_1 F(c)w(c)\ dc = 0$$

and

$$\int_0^{\pi_1} c\pi_0(1 - F(c))w(c)\ dc = \int_0^{\pi_1} c\pi_0 w(c)\ dc \quad \text{and} \quad \int_{\pi_1}^1 (1-c)\pi_1 F(c)w(c)\ dc = \int_{\pi_1}^1 (1-c)\pi_1 w(c)\ dc$$

therefore

$$L_{\text{ref}} = \int_0^{\pi_1} c\pi_0 w(c)\ dc + \int_{\pi_1}^1 (1-c)\pi_1 w(c)\ dc$$

7 Gini Impurity Index G_1

Gini Impurity Index G_1 Definition

Definition 3 *Given a classification with TPR and FPR fiunctions $F_0(t)$ and $F_1(t)$ the Gini Impurity Index is defined as*

$$G_1 = 2\int_0^1 F_0(t)F_1'(t)\ dt - 1 = 2AUC - 1$$

8 A Classification Example

Example

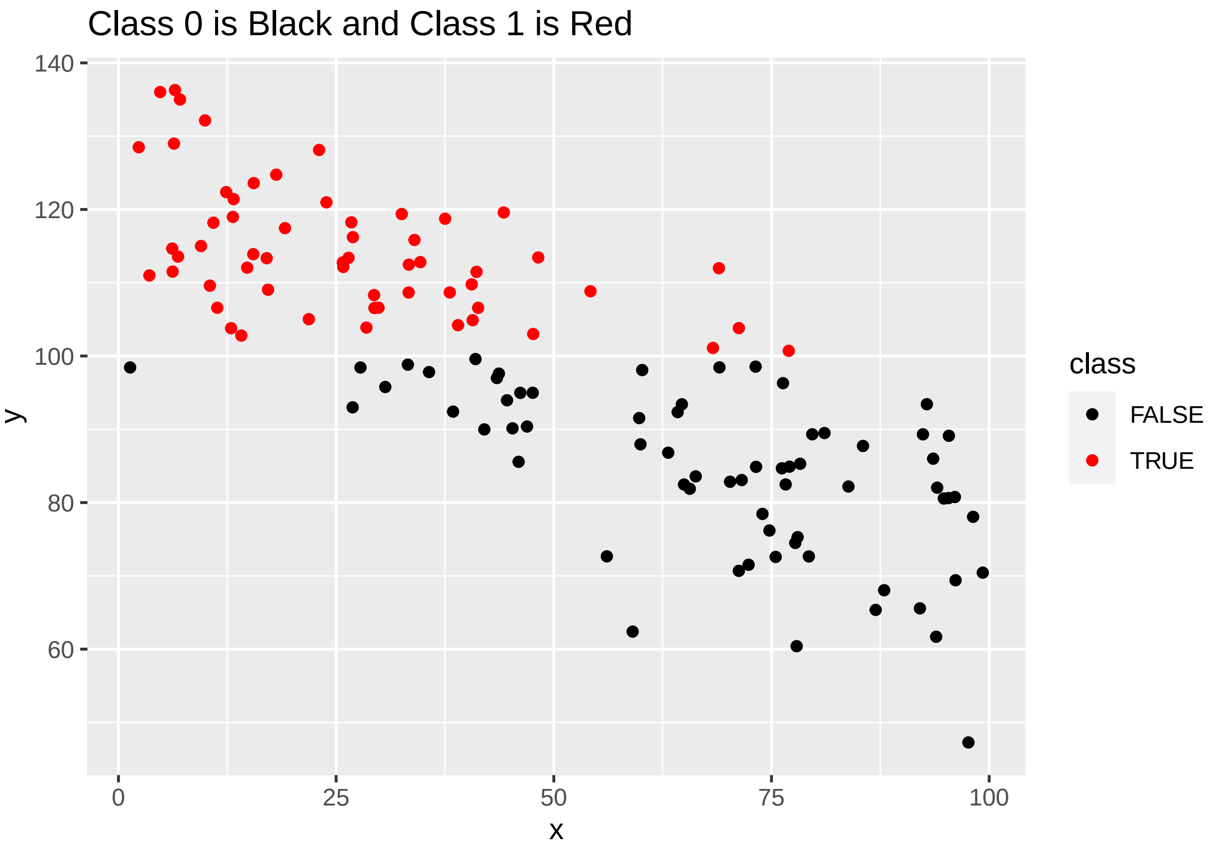
In this example we generate a randonm vector then divide it into two subsets, training set, and test set. We will use the training set to generate a Generalized Linear Model and use it to classify the test set. In the end we will graph the ROC curve and will calculate the AUC.

8.1 Using our Own Code

```
#
# This function generates a random sample.
# It generates a random sample of size N from Uniform(0,1)
# Then adds a noise which is Normal(mu, sd)
# Where mu < 600 and sd <100
#
simulation_data <- function(mu, sd, threshold){
  x <- runif(mu, min = 0, max = 100)
  y <- 122 -x/2 + rnorm(mu, sd = sd)
  class <- factor(y > threshold)
  data.frame(x, y, class)
}
```

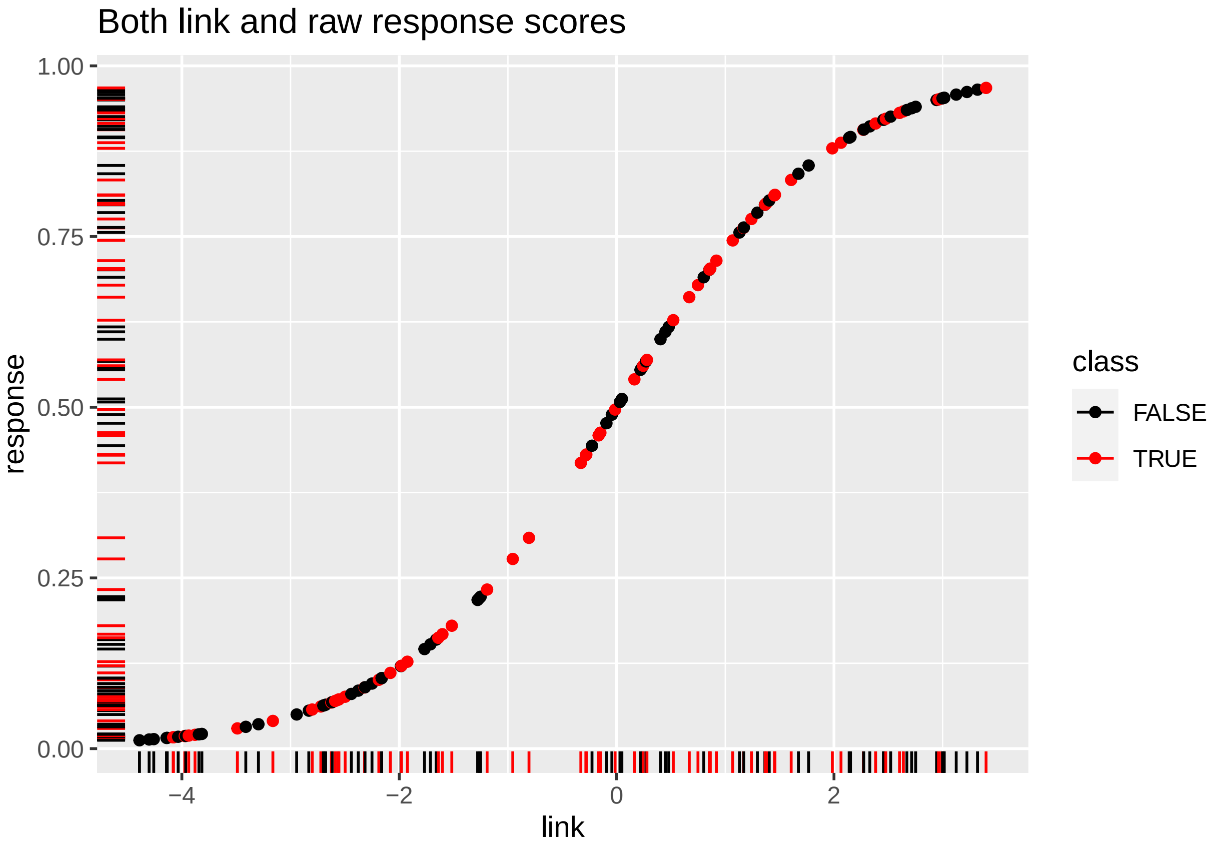
```
#
# Generate data
#
data <- simulation_data(500, 10, 100)
#
# Labesl for the test set
#
labels <- sample(1:nrow(data), size = floor(nrow(data)/4))
#
# Test set consists of 1/4 of the data
#
test_set <- data[labels,]
#
# Training Set is the rest
#
training_set <- data[-labels,]
#
# plot the test set
#
```

```r
ggplot(test_set, aes(x = x, y = y, col = class)) +
  scale_color_manual(values=c("black", "red")) +
  geom_point() +
  ggtitle("Class 0 is Black and Class 1 is Red")
```



```r
#
# Fitting a Generalized Linear Model (glm)
# We use the logit link function which is
# used with a Binomial glm
# That is
# logit(y) = alpha_0 + alpha_1(x_1)
#
# Fit the model to the training set
#
glm_model <- glm(class ~ x, training_set, family = binomial(link = "logit"))
#
# Apply the model to the test set
#
test_set_class_link <- predict(glm_model, test_set, type = "link")
#
# Any Response must be sorted
#
test_set_class_link <- sort(test_set_class_link)
#
# To observe the raw response vs the logit of the response
# we graph them against each other
#
```

```
test_set_class_Raw_Response <- predict(glm_model, test_set, type = "response")
#
# Any Response must be ordered
#
test_set_class_Raw_Response <- sort(test_set_class_Raw_Response)
#
classification_data <- data.frame(link = test_set_class_link,
                                  response = test_set_class_Raw_Response,
                                  class = test_set$class,
                                  stringsAsFactors=FALSE)
ggplot(classification_data, aes(x = link, y = response, col = class)) +
  scale_color_manual(values=c("black", "red")) +
  geom_point() +
  geom_rug() +
  ggtitle("Both link and raw response scores")
```

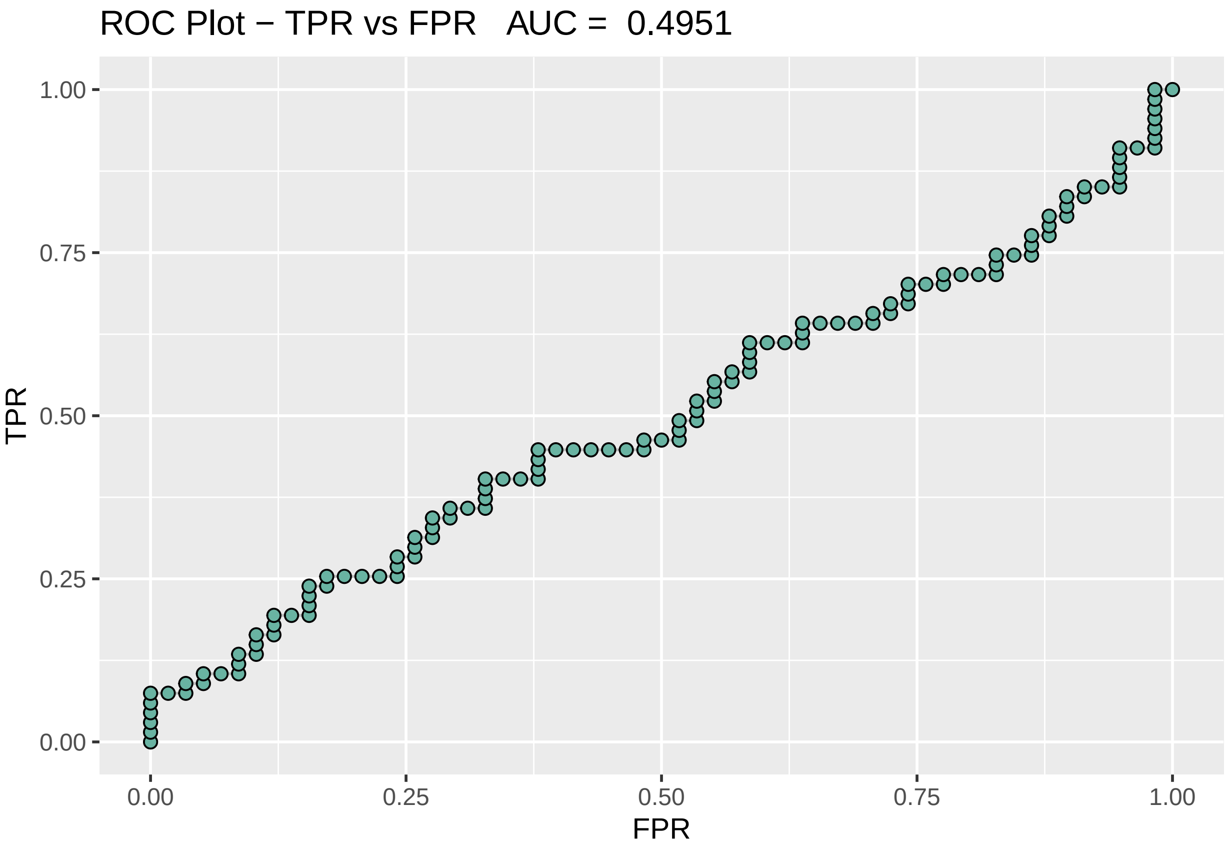


```
data_labels <- test_set$class
data_labels <- as.numeric(data_labels) - 1
#
# TPR
#
data_tpr <- tpr(data_labels)
data_and_tpr_array <- c(data, data_tpr)
tpr_and_data <- matrix(data_and_tpr_array, 2, byrow =TRUE)
# print(tpr_and_data)
#
# FPR
```

```
#
data_fpr <- fpr(data_labels)
data_and_fpr_array <- c(data, data_fpr)
fpr_and_data <- matrix(data_and_fpr_array, 2, byrow =TRUE)
# print(fpr_and_data)
#
# ROC Graph and AUC
#
data_fpr <- c(data_fpr, 1)
data_tpr <- c(data_tpr, 1)
sensitivity <- data_tpr
specificity <- 1 - data_fpr
#
nn <- length(data_tpr) - 1
#
# Calculating AUC
# We use the Trapezoidal Rule to find the integral
# AUC = int_0^1 F_0(t) F_1'(t) dt given below
#
AUC <- 0
for (i in 1 : nn){
  AUC <- AUC + (data_tpr[i+1] + data_tpr[i])*(data_fpr[i+1] - data_fpr[i])/2
}
print(c("AUC = ", AUC))
```

```
## [1] "AUC = "          "0.49511065362841"
```

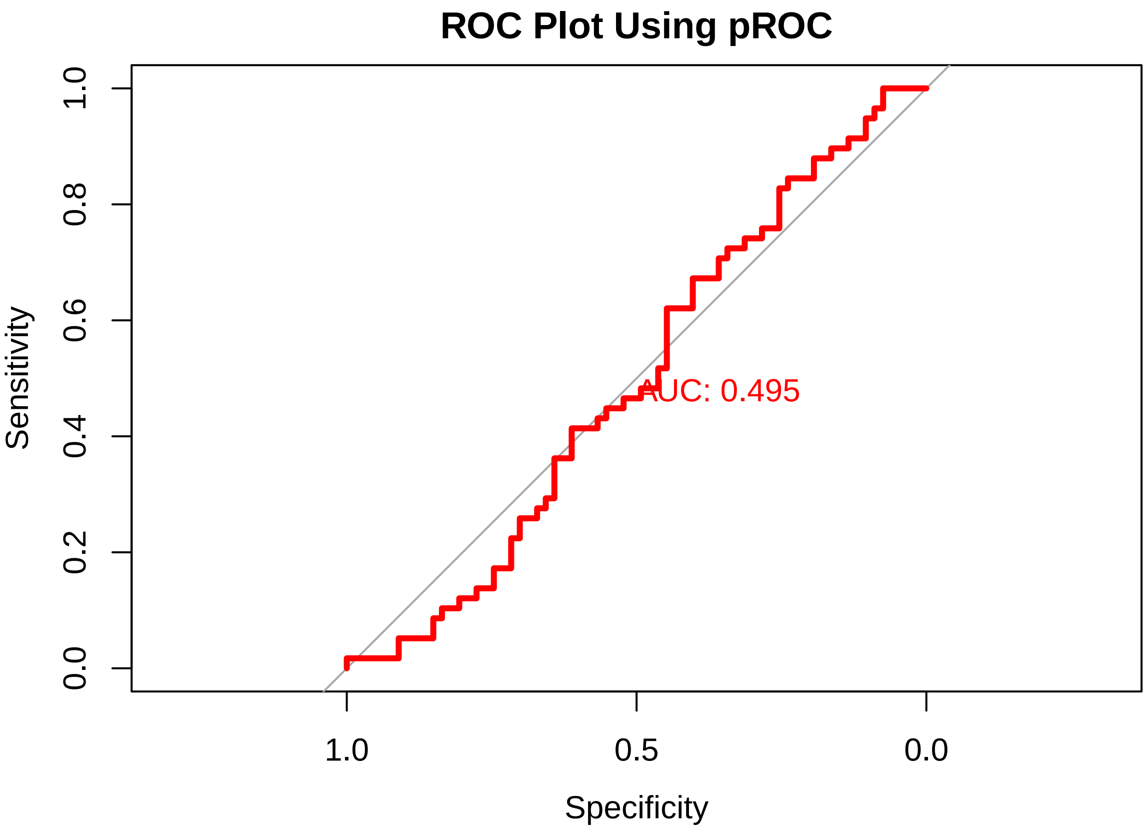
```
ROC_Data <- data.frame("FPR"=data_fpr, "TPR"=data_tpr)
# print(ROC_Data)
roc_title <- paste("ROC Plot - TPR vs FPR   ")
auc_in_title <- paste("AUC = ", round(AUC, digits = 4))
ggplot(ROC_Data, aes(x=FPR, y=TPR)) +
    geom_line( color="grey") +
    geom_point(shape=21, color="black", fill="#69b3a2", size=2) +
  ggtitle(paste0(roc_title, auc_in_title))
```



8.2 Using pROC

```
#
# Don't forget: Any Response Must be ordered before
# Feeding to pROC
# Important Note: Unordered Response will result in
# a wrong plot and AUC.
#
plot(roc(test_set$class, test_set_class_Raw_Response, direction="<"),
     col="red", lwd=3, main="ROC Plot Using pROC", print.auc=TRUE)
```

```
## Setting levels: control = FALSE, case = TRUE
```



References

[1] P. L. Batlett, P. L., Jordan, M. I., and McAuliffe, J. D., *Convexity, classification, and risk bounds.* Journal of the American Statistical Association, (2003).

[2] Berger James O., *Statistical Decision Theory and Bayesian Analysis.* Springer (1985).

[3] Buja, A., Stuetzle, W., and Shan, Y., *Loss functions for binary class probability estimation and classification: structure and applications.* Technical Report, The Wharton School. http://www-stat.wharton.upenn.edu/ buja/PAPERS/paper-proper-scoring.pdf

[4] Hand David, J., *Mesuring classifier performance: a coherent alternative to the area under the ROC curve.* Machine Learning, (2009) 77: 103-123.