## Classification

Suppose we have a binary classification problem. For example, we want to predict whether a patient will respond to chemotherapy (Y=1) or not (Y=0).

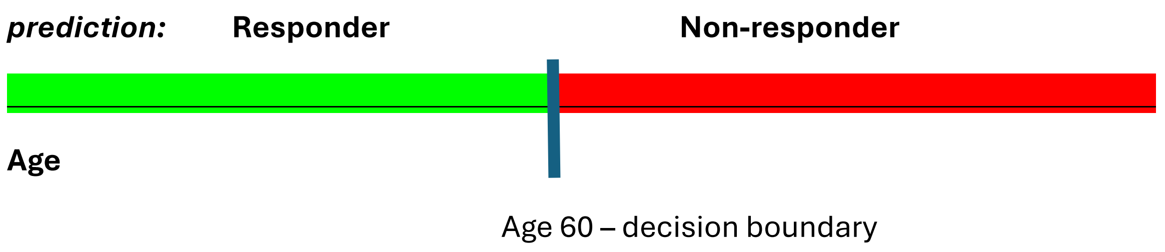
A simple classifier: We predict that if you are 60 years of age or older, you will not respond to chemotherapy with probability=0. And if you are younger than 60, you will respond with probability=1. This of course is not a very good model, but it is a helpful illustration.

**Decision function:**

**Y=1, if age < 60**

**Y=0, if age 60+**

This creates a decision boundary at age 60:



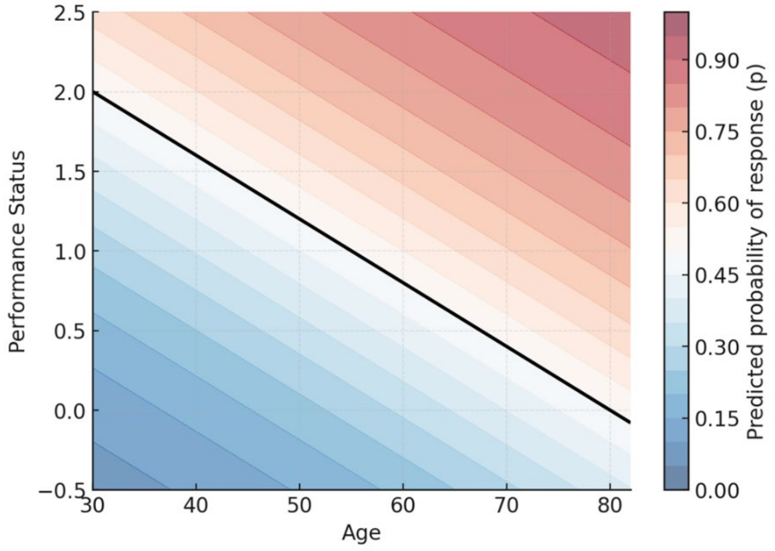
Patients that fall on one side of the decision boundary (ages < 60) would be assigned a probability of responding of 1, and patients on the other side (age 60+) would be assigned a probability of responding of 0. With this simple rule, we do not differentiate whether a patient is 20 or 59 years old – they would both get a predicted probability of 1. So not only do we use age as the only factor to predict chemotherapy response, but with our simple decision function we do not even assign lower probability of response to patients that are close in age to 60 (our decision boundary) vs. patients who are age 20. Furthermore, we do not know if age 60 is the optimal decision boundary or whether response probability may change in a more linear fashion according to age of the patient.

Let’s consider a slightly more flexible classification model.  For this we will use both patient age and performance status as predictors in the model.

**A 2-D example of a decision function using 2 variables in a logistic regression model – age and performance status.**

$$\log\left(\frac{p}{1-p}\right) = \widehat{\beta_0} + \widehat{\beta_{PS}} Performance\ Status + \widehat{\beta_{age}} Age$$

$$\text{where:}\ \ \widehat{\beta_{age}} = 0.04, \widehat{\beta_{PS}} = 1.0, \widehat{\beta_0} = -3.2$$



The red side represents the combinations of performance status and age that cause the model to have a predicted probability > 0.50 and predict “Responder” and the blue side represents all combinations that cause the model to predict “Non-responder”.  We can see that better performance status and younger ages are associated with “Responder” predictions.  As expected, a 30-year-old with good performance status is far away from the decision threshold with a high predicted probability of responding to chemotherapy, whereas a 60-year-old with moderate performance is sitting right at the boundary and would have a probability of responding close to average response rate.  An older patient with poor performance status deep in the blue area would have a very low predicted probability of responding.

**How was this decision function determined? And how did we get those model coefficients?**

**The answer lies in understanding cost functions.**

## Cost functions

A cost function (also called a loss function) is a way to quantify how good a classifier is at separating responders from non-responders based on the data at hand.

Suppose we look at the observed chemotherapy response for 5 patients along with the predicted probability of responding. For classification purposes, any patient with predicted probability ≥ 0.50 is predicted to be a responder, while any patient with predicted probability < 0.50 is predicted to be a non-responder.

| **Observation** | **Observed response** | **Prediction ($\hat{p}$)** | **Predicted class** |
|---|---|---|---|
| 1 | 1 | .71 | 1 |
| 2 | 1 | .52 | 1 |
| 3 | 0 | .05 | 0 |
| 4 | 1 | .25 | 0 |
| 5 | 0 | .90 | 1 |

**From this table, we see that the classifier we used was correct only 40% of the time.**

However, this does not take into account that the classifier was:

1. *confident*ly correct in its prediction for observation 3. This patient (who did not respond) had a predicted value of $\hat{p}$=0.05.
2. not confident for correctly classified observation 2 as the prediction ($\hat{p}$=0.52) was very close to the 0.50 decision boundary
3. *confidently* incorrect for observation 5 who did not respond but had a predicted probability of $\hat{p}$=0.90, far from the decision boundary

Given this, we want a better way to measure how well the model is doing.

## Cost function #1: Log Loss (cross entropy)

**Log loss for each observation: $L_L = -[Y \times LN(\hat{p}) + (1-Y) \times LN(1-\hat{p})]$**

*note: LN=natural log*

| Observation | Observed response | Prediction from the classifier ($\hat{p}$) | Log Loss ($L_L$) |
|---|---|---|---|
| 1 | 1 | .71 | 0.34 |
| 2 | 1 | .52 | 0.65 |
| 3 | 0 | .05 | 0.05 |
| 4 | 1 | .25 | 1.39 |
| 5 | 0 | .90 | 2.30 |

## Cost function #2: Hinge Loss

**Hinge loss for each observation: $L_H = 2(1-\hat{p})$ if Y=1 and $2\hat{p}$ if Y=0**

*Note: Hinge loss is typically defined based on labelling responses as -1 and +1 with a decision function f and decision boundary at f=0. Here, $L_H$= max(0,1 - Yf) and cost is 0 if Y=+1 and f ≥ 1 or Y=-1 and f ≤ -1 (correct and outside the margin).*

*For simplicity, I have re-expressed this in terms of probabilities, where $\hat{p} = (f+1)/2$. Under this transformation, the boundary f=0 corresponds to $\hat{p}$=0.50.*

| Observation | Observed response | Prediction from the classifier ($\hat{p}$) | Hinge Loss ($L_H$) |
|---|---|---|---|
| 1 | 1 | .71 | 0.58 |
| 2 | 1 | .52 | 0.96 |
| 3 | 0 | .05 | 0.10 |
| 4 | 1 | .25 | 1.50 |
| 5 | 0 | .90 | 1.80 |

**For both cost functions, we compute the cost for each observation and then sum up costs across all observations.**

## Differences between Log and Hinge loss

| Observation | Observed response | Prediction from the classifier ($\hat{p}$) | Log Loss ($L_L$) | Hinge Loss ($L_H$) |
|---|---|---|---|---|
| 1 | 1 | .71 | 0.34 | 0.58 |
| 2 | 1 | .52 | 0.65 | 0.96 |
| 3 | 0 | .05 | 0.05 | 0.10 |
| 4 | 1 | .25 | 1.39 | 1.50 |
| 5 | 0 | .90 | 2.30 | 1.80 |

The difference between the two types of loss is how errors are treated. For example, Observation 2 is on the correct side of the boundary ($y=1$, $\hat{p}=0.52$), but because the prediction is so close to 0.5, hinge loss penalizes it more (0.96 vs 0.65). This reflects hinge loss's focus on wanting observations far away from the margins of the decision boundary.

*We can see from this comparison that log loss penalizes very confident (predictions closer to 0 or 1) but incorrect decisions, while hinge loss penalizes less confident decisions closer to the decision boundary, even if they are on the correct side of it.*

## Cost function #3: Gini impurity

We use Gini impurity for tree-based classifiers, such as Random Forest, and it is a bit different than the other two cost functions. Decision trees don't optimize a loss for each individual observation and then sum over all observations, instead they optimize a measure of **node purity** (Gini) across all terminal nodes. For tree-based classifiers, a good classifier would have terminal nodes (i.e. determined by combinations of predictors such as age < 60 and performance status < 2) that consist mostly of observations with same response status (i.e. mostly all responders or mostly all non-responders).

**The Gini impurity for node k is computed as:**

$$\text{Gini}_k = 1 - p_R^2 - p_{NR}^2$$

Where $p_R$ = the proportion of responders to chemotherapy and $p_{NR}$ = the proportion of non-responders in the same node. If node k is "pure", consisting only of responders, then $p_R=1$

and Gini = 1 – 1- 0=0.  If 50% responders and 50% non-responders, then the impurity is at its maximum, and is equal to 1 - .25 - .25 = 0.50.

## Cost functions: finding the optimal decision boundary

Let's use our simple logistic regression for response to chemotherapy as an example.

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_{PS} Performance\ Status + \beta_{age} Age$$

For every combination of model coefficients ($\beta_{PS}$ and $\beta_{age}$), we can compute the log loss and sum it across all observations.  The optimal model would then be the one resulting in the lowest cost across all possible combinations of model coefficients.  Of course, computationally, the model fitting process does not do this via a huge grid search and instead uses an iterative process (e.g.: gradient descent) that can find an optimal model without having to compute the cost function for infinite combinations.

**All classifiers use a cost function to find an optimal decision boundary.**

**Table: Cost functions and standard output for each classifier**

| **Classifier** | **Standard Output** | **Typical Cost function used** |
|---|---|---|
| Logistic regression | Directly estimates Pr(Y=1) | Log loss |
| Random Forest | the proportion of trees in the ensemble that voted for Y=1 | Gini purity (to split data into nodes) |
| SVM | how far the observed value is from the decision boundary (i.e. the margin), rescaled to be between [0,1] | Hinge loss |
| XGB | a score that is rescaled to be between [0,1] | Log loss |
| Neural Networks | Directly estimates Pr(Y=1) | Log loss |