Applied Machine Learning
Class 3

Master 2 Physique Fondamentale et Applications

Daniel Förster
daniel.forster@univ-orleans.fr
UFR Sciences et Techniques Université d'Orléans

Chapter 3: Supervised Learning: Classification
B: Logistic Regression

Logistic Regression

Predict the probability of a binary outcome (e.g., yes/no, success/failure) based on one or more independent variables.

- The output is a probability value between 0 and 1.
- Used for classification tasks where the outcome is categorical with two possible values.

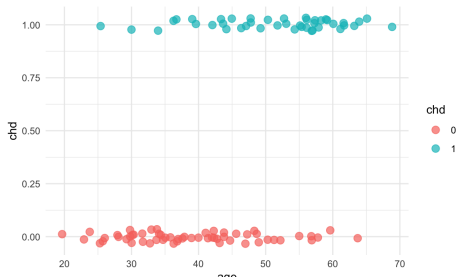
Examples of Use Cases:

- Predicting whether a patient has a disease.
- Determining if an email is spam or not.
- Forecasting customer behavior (e.g., will they buy?).

Example Problem: Probability of Coronary Heart Disease (CHD) with Age

Problem Setup:

- Data: 100 patients with information on medical variables.
- Goal: Predict the presence (`chd = 1`) or absence (`chd = 0`) of coronary heart disease.
- Predictor: age (in years).



Scatterplot of age vs. chd with jitter added

Image source [1]

Calculating Conditional Means

Why not linear regression?

- Linear regression cannot really predict probabilities/binary outcomes.
- A sign-transformation helps, but it's not ideal.
- An alternative: conditional means.

Conditional expectation:

$$E(y \mid x).$$

The sigmoid pattern can be approximated using the logistic function:

$$P(y = 1 \mid x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}.$$

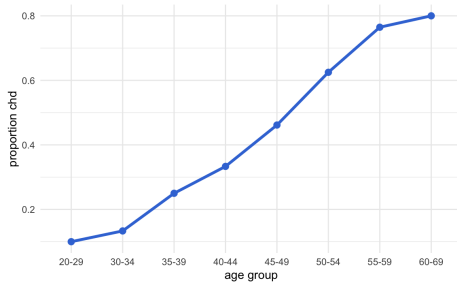


Image source [2]

Fit: Measuring Goodness of Fit in Logistic Regression

Objective: Find the "best fit" for the model.

- In logistic regression, goodness of fit is measured using the **logistic loss** (or **log loss**).
- The log loss is derived from the negative **log-likelihood** of the data.

Log Loss for a Single Data Point:

$$\ell_k = \begin{cases} -\ln p_k & \text{if } y_k = 1, \\ -\ln(1 - p_k) & \text{if } y_k = 0. \end{cases}$$

- p_k: Predicted probability for $y_k = 1$.
- $1 - p_k$: Predicted probability for $y_k = 0$.

Interpretation of Log Loss

- Measures the "surprisal" or Shannon information of the actual outcome relative to the prediction.
- Log loss:
  - Equals 0 for a perfect prediction.
  - Approaches infinity for highly incorrect predictions.
  - Always > 0 and $< \infty$ (logistic outputs are strictly between 0 and 1).

Log Loss for the Dataset:
Summing across all data points gives the **total loss**, the negative log likelyhood $-\ell$:

$$-\ell = \sum_{k=1}^{n} \Big(- y_k \ln p_k - (1 - y_k) \ln(1 - p_k) \Big).$$

This represents the **cross-entropy** between the predicted and actual distributions.

Optimizing the Model

- Minimize the negative log-likelihood ($-\ell$), or equivalently:
- Maximize the positive log-likelihood, or:

$$\ell = \sum_{k=1}^{n} \Big(y_k \ln p_k + (1 - y_k) \ln(1 - p_k) \Big).$$

- Maximize the likelihood Function:

$$L = \prod_{k:y_k=1} p_k \cdot \prod_{k:y_k=0} (1 - p_k),$$

where $p_k = \sigma(\beta_0 + \beta_1 x_k) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_k)}}$

$\rightarrow$ use gradient descent to find β_0 and β_1.

Confusion Matrix: Evaluating Classification Performance

Confusion Matrix: Table where one axis represents the **actual labels**, and the other axis the **predicted labels**.

Example: CHD Classification

	CHD Present (Actual)	**CHD Absent (Actual)**
CHD Predicted (1)	15 (True Positive, TP)	10 (False Positive, FP)
No CHD Predicted (0)	5 (False Negative, FN)	70 (True Negative, TN)

- **True Positives (TP):** Correctly predicted presence of CHD.
- **False Positives (FP):** Incorrectly predicted CHD when it's absent.
- **True Negatives (TN):** Correctly predicted absence of CHD.
- **False Negatives (FN):** Incorrectly predicted absence of CHD when it's present.

Accuracy: A Simple Performance Metric

Proportion of correctly classified examples out of all examples.

Formula:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Example: CHD Classification

- From the confusion matrix:

$$\text{Accuracy} = \frac{15 + 70}{15 + 70 + 10 + 5} = \frac{85}{100} = 0.85.$$

- Interpretation: The model correctly classified 85% of the patients.

Cost-Sensitive Accuracy: Adjusting for Class Importance

- Assign costs for false positives (C_{FP}) and false negatives (C_{FN}).
- Adjust the formula:

$$\text{Cost-Sensitive Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + C_{FP} \cdot \text{FP} + C_{FN} \cdot \text{FN}}.$$

- Example: For CHD diagnosis:
  - $C_{FP} = 1$ (less costly to predict CHD when it's absent).
  - $C_{FN} = 5$ (more costly to miss CHD when it's present).
  - Adjusted denominator:

$$\text{Cost-Sensitive Accuracy} = \frac{15 + 70}{15 + 70 + 1 \cdot 10 + 5 \cdot 5} = \frac{85}{120} = 0.708.$$

Precision and Recall

Precision:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

- Proportion of correct positive predictions out of all predicted positives.
- Example: For CHD classification: Precision $= 15/(15 + 10) = 0.6$.
- Interpretation: 60% of the model's positive CHD predictions were correct.

Recall:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- Proportion of correctly predicted positives out of all actual positives.
- Example: For CHD classification: Recall $= 15/(15 + 5) = 0.75$.
- Interpretation: The model identified 75% of all CHD cases.

Precision vs. Recall: Trade-offs and Use Cases

The Trade-off:

- High precision often leads to lower recall and vice versa.
- It's usually not possible to maximize both simultaneously.

Real-World Implications:

- For CHD diagnosis:
  - **High Precision:** Avoids false positives, reducing unnecessary tests or treatments.
  - **High Recall:** Minimizes false negatives, ensuring more CHD cases are detected.
- The choice depends on the clinical priority:
  - If early diagnosis is critical, prioritize high recall.
  - If avoiding unnecessary treatments is critical, prioritize high precision.

How to Adjust?

- Adjust the decision threshold for the model (e.g., require $P(\texttt{chd} = 1) > 0.9$ for a positive prediction to increase precision).

Receiver Operating Characteristic (ROC) Curve

Plots the **True Positive Rate (TPR)** against the **False Positive Rate (FPR)** for various thresholds.

Definitions:

$$\text{True Positive Rate (TPR)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad \text{(Same as Recall)}.$$

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

- Vary the threshold for positive predictions to compute TPR and FPR for each threshold.
- Example thresholds: $0, 0.1, 0.2, \ldots, 1$.
- Plot TPR vs. FPR to generate the ROC curve.

Interpreting the ROC Curve

- The ROC curve starts at (0, 0) and ends at (1, 1).
- If the threshold is 0:
  - All predictions are positive.
  - TPR = 1, FPR = 1 (upper right corner).
- If the threshold is 1:
  - All predictions are negative.
  - TPR = 0, FPR = 0 (lower left corner).
- A well-performing model has a curve closer to the upper left corner.

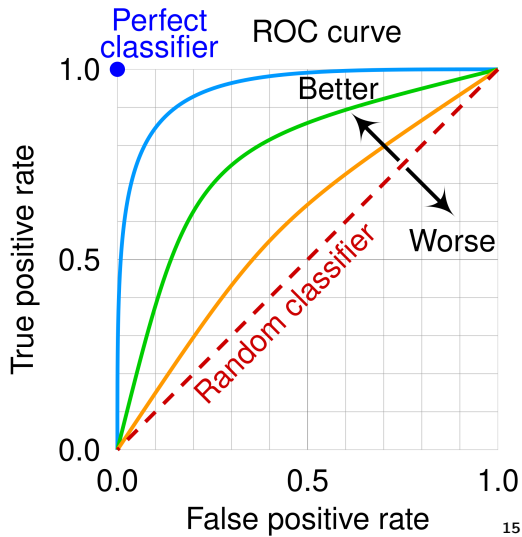


Image source [3]

Area Under the Curve (AUC)

- AUC measures the area under the ROC curve.
- Ranges from 0 to 1:
  - AUC $= 1$: Perfect classifier.
  - AUC $= 0.5$: Random guessing.
  - AUC < 0.5: Worse than random.
- Higher AUC indicates better model performance.

Chapter 3: Supervised Learning: Classification and Regression
C: Decision Trees

Introduction to Decision Trees

A decision tree is a predictive model used for classification and regression tasks.

- **Root Node:**
  - No incoming edge.
  - Zero or more outgoing edges.
- **Internal Node:**
  - One incoming edge.
  - Two (or more) outgoing edges.
- **Leaf Node:**
  - Assigned a class label if nodes are pure.
  - If not pure, the class label is determined by majority vote.

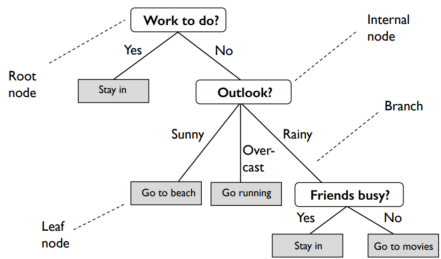


Image source [4]

Growing a Decision Tree

Algorithm Overview:

1. Select a feature that maximizes information gain when the parent node is split.
2. Stop if:
   - All data points in a node belong to the same class (pure node).
   - No further improvement in class purity is possible.
3. Recursively apply the process to each child node.

Design Considerations for Decision Trees

- **Feature Selection:**
  - How do we decide which feature to select for splitting a parent node into child nodes?
  - What criterion measures the goodness of the split?
- **Tree Stopping Criteria:**
  - When do we stop growing a tree?
  - Complete separation can easily lead to overfitting.
- **Prediction in Non-Pure Nodes:**
  - How do we make predictions if no attributes perfectly separate non-pure nodes further?
  - Majority vote.
- **Splitting Strategy:**
  - Multi-category splitting can be expressed as a series of binary splits.
  - Which approach is preferred?
- **Handling Continuous Inputs:**
  - Splitting categorical features is intuitive.
  - How can continuous inputs be handled effectively?

Shannon Entropy

$$H(p) = \sum_i p_i \log_2 \left(\frac{1}{p_i} \right)$$

Where p_i represents the probability of each possible outcome.

- Shannon entropy quantifies the amount of uncertainty or information in a probability distribution.
- A higher entropy value means the outcomes are more unpredictable.
- It is a fundamental concept in information theory, representing the minimum number of bits needed to encode a message.
- For example, a fair coin toss has high entropy ($H = 1$), while a biased coin has lower entropy.

Shannon Entropy Visualization in case of Binary Classification

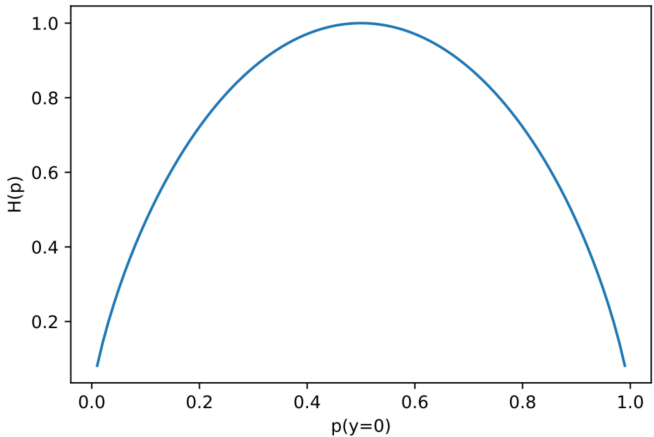


Image source [4]

Shannon Entropy and Connection to Statistical Physics

- Shannon entropy quantifies the uncertainty or information in a system.
- In statistical physics, entropy is defined as:

$$S = -k_{\mathrm{B}} \sum_i p_i \ln(p_i)$$

  where:
  - S: Entropy of the system.
  - k_B: Boltzmann constant.
  - W: Number of microstates consistent with the system's macrostate.
- The two concepts are analogous:
  - Shannon entropy measures uncertainty in terms of probability distribution.
  - Boltzmann entropy measures uncertainty in terms of the number of microscopic configurations.
- Both describe the "spread" or "disorder" of a system.

Maximizing Information Gain as Splitting Criterion

- Information gain measures the reduction in uncertainty or entropy after a split.
- The goal is to choose a feature x_j that maximizes the information gain.

Information Gain:

$$GAIN(D, x_j) = H(D) - \sum_{v \in Values(x_j)} \frac{|D_v|}{|D|} H(D_v)$$

- D: Training set at the parent node.
- D_v: Subset of D corresponding to value v of feature x_j.
- $H(D)$: Entropy of the dataset D.

Explanation:

- The first term $H(D)$ represents the entropy of the parent node.
- The second term represents the weighted sum (number of elements) of entropies of child nodes.

Gini Impurity as a Splitting Criterion

$$Gini(p) = 1 - \sum_i p_i^2$$

Where p_i is the probability of a data point belonging to class i.

- Measures the impurity or heterogeneity of a dataset.
- Gini impurity is 0 for a pure node (all data points belong to one class).
- Maximum impurity occurs when classes are equally distributed.

Comparison to Entropy:

- Gini impurity is computationally simpler than entropy.
- Both measures are often used interchangeably in decision tree algorithms.

Tree Pruning: Minimizing Overfitting

Reducing the size of a decision tree by removing parts that do not provide significant predictive power

Types of Pruning:

- **Pre-Pruning (Early Stopping):**
  - Stop growing the tree during the splitting process.
  - Criteria: minimum number of samples per node, maximum depth, or improvement threshold.
- **Post-Pruning:**
  - Grow the tree fully and then remove nodes that do not improve performance.
  - Typically uses a validation set to determine which nodes to prune.

Other Benefits of Pruning:

- Reduces model complexity.
- Improves interpretability of the decision tree.

Decision Trees for Regression

Regression trees predict continuous values instead of discrete labels.

Splitting Criterion:

- At each split, choose the feature and threshold that minimize the **variance** of the target variable within the child nodes.
- Variance reduction is calculated as:

$$\text{Reduction} = \text{Var}(D) - \left(\frac{|D_L|}{|D|} \text{Var}(D_L) + \frac{|D_R|}{|D|} \text{Var}(D_R) \right)$$

  Where:
  - D: Dataset at the parent node.
  - D_L, D_R: Datasets at the left and right child nodes.
  - $\text{Var}(D)$: Variance of the target variable in dataset D.

Leaf Node Predictions:

- The prediction at a leaf node is the **mean** of the target variable within that node.
- **Mean Squared Error (MSE)** can be used as a loss function

Pros and Cons of Decision Trees

Pros:

- Easy to interpret and communicate
- Independent of feature scaling

Cons:

- Easy to overfit
- Elaborate pruning required
- Expensive to just fit a "diagonal line"
- Output range is bounded (dep. on training examples) in regression trees

Random Forests

Multiple decision trees combined to reduce overfitting and increase accuracy using bootstrap aggregation (bagging) and random feature selection.

- Uses multiple random samples of the training set (with replacement).
- At each split, considers a random subset of features to reduce correlation between trees.
- Outputs the majority vote (classification) or average prediction (regression).

The Bootstrap Aggregation (Bagging) Algorithm

1. Create B random samples S_b ($b = 1, \ldots, B$) from the training set D using sampling with replacement.
2. Train a model f_b on each sample S_b.
3. Combine predictions from all B models:
   - **Regression:** Take the average:
   - **Classification:** Take the majority vote.

Random Forest Algorithm

1. Create B random samples with replacement from the training set.
2. Build a decision tree for each sample:
   - At each split, **consider only a random subset of features.**
3. Combine predictions as in bagging (average for regression, majority vote for classification).