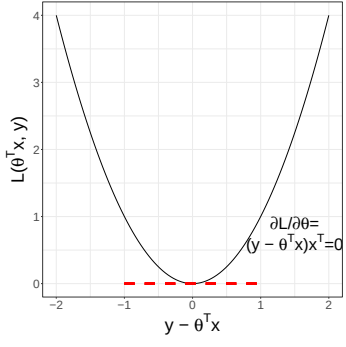
Regression & Classification

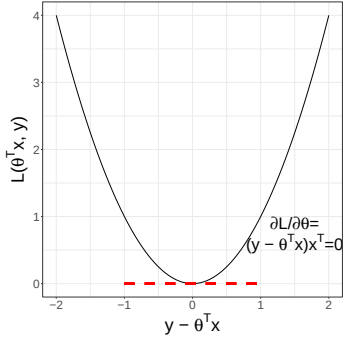


Learning goals

- Understand basic concept of regressors and classifiers
- Understand difference between L1 and L2 Loss
- Know basic idea of OLS estimator
- Know concepts of probabilistic and scoring classifier
- Know distinction between discriminant and generative approach
- Understand ideas of logistic regression and Naive Bayes

Introduction to Machine Learning

Supervised Regression: In a Nutshell



Learning goals

- Understand basic concept of regressors
- Understand difference between L1 and L2 Loss
- Know basic idea of OLS estimator

LINEAR REGRESSION TASKS

Learn linear combination of features for predicting the target variable
Find best parameters of the model by training w.r.t. a loss function
$CreditBalance = \theta_0 + \theta_1 Rating + \theta_2 Income + \theta_3 CreditLimit$

Training

Input: Labeled data

Rating	Income	Credit Limit	Credit Card Balance
283	14.891	3606	333
483	106.025	6645	903
514	104.593	7075	580

Learning

Output

Regressor

Prediction

Input: Unlabeled data

Rating	Income	Credit Limit	Credit Card Balance
107	32.318	4351	?
471	88.180	5042	?
512	121.218	8101	?

Regressor

Prediction

Credit Card Balance
482
720
987

EMPIRICAL RISK MINIMIZATION / 2

Minimizing this surface is called **empirical risk minimization** (ERM).

$$\hat{\theta} = \arg\min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}).$$

Usually we do this by numerical optimization.

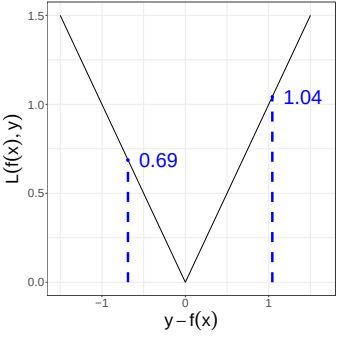
$$\mathcal{R} : \mathbb{R}^d \to \mathbb{R}.$$

Model	$\boldsymbol{\theta}_{intercept}$	$\boldsymbol{\theta}_{slope}$	$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$
f_1	2	3	194.62
f_2	3	2	127.12
f_3	6	-1	95.81
f_4	1	1.5	57.96
f_5	1.25	0.90	23.40

In a certain sense, we have now reduced the problem of learning to **numerical parameter optimization**.

LINEAR MODELS: L1 VS L2 LOSS

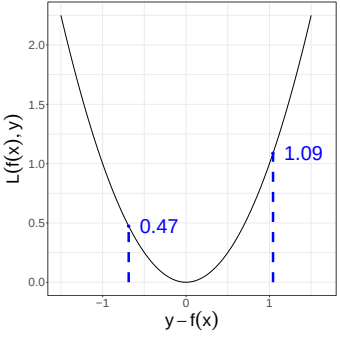
Loss can be characterized as a function of residuals $r = y - f(\mathbf{x})$



L1 penalizes the **absolute** value of residuals

$L(r) = |r|$

Robust to outliers



L2 penalizes the **quadratic** value of residuals

$L(r) = r^2$

Easier to optimize

LINEAR MODELS: L1 VS L2 LOSS

L1 Loss is not differentiable in $r = 0$

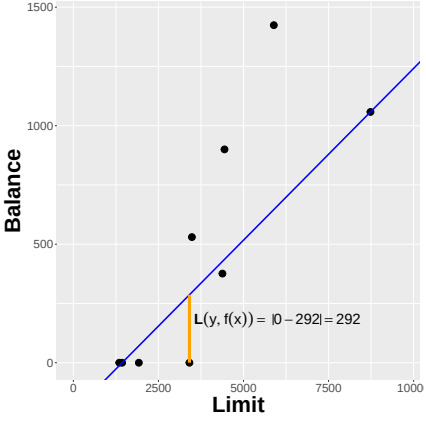
Optimal parameters are computed numerically

L2 is a smooth function hence it is differentiable everywhere

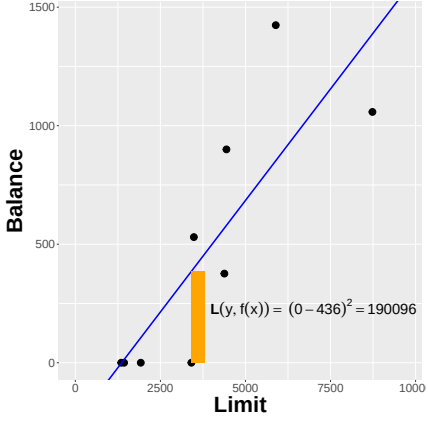
Optimal parameters can be computed analytically or numerically

LINEAR MODELS: L1 VS L2 LOSS

The parameter values of the best model depend on the loss type



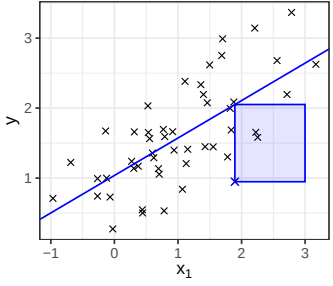
$\hat{\theta}_{L_1} = 0.14 \rightarrow$ if the Credit Limit increases by 1$ the Credit Balance increases by 14 Cents



$\hat{\theta}_{L_2} = 0.19 \rightarrow$ if the Credit Limit increases by 1$ the Credit Balance increases by 19 Cents

Introduction to Machine Learning

Supervised Regression: Linear Models with *L2* Loss



Learning goals

Grasp the overall concept of linear regression

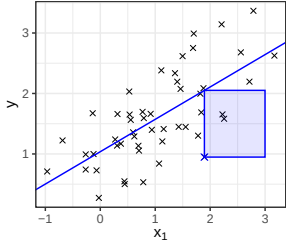
Understand how *L2* loss optimization results in SSE-minimal model

MODEL FIT

How to determine LM fit? $\rightsquigarrow$ define risk & optimize

Popular: *L2* **loss** / **quadratic loss** / **squared error**

$$L(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2 \text{ or } L(y, f(\mathbf{x})) = 0.5 \cdot (y - f(\mathbf{x}))^2$$

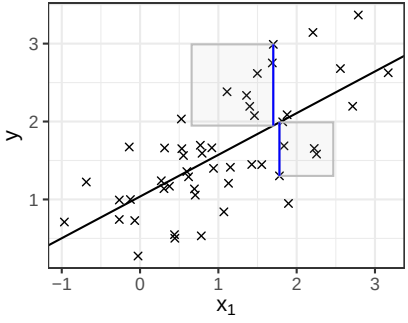


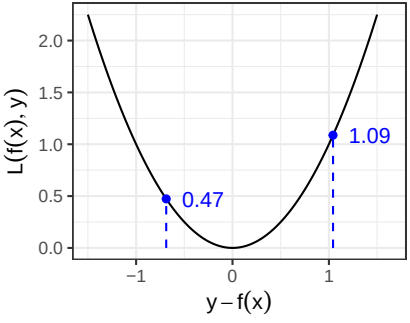
Why penalize **residuals** $r = y - f(\mathbf{x})$ quadratically?

- Easy to optimize (convex, differentiable)
- Theoretically appealing

LOSS PLOTS

We will often visualize loss effects like this:





Data as $y \sim x_1$

Prediction hypersurface
$\rightsquigarrow$ here: line

Residuals $r = y - f(\mathbf{x})$
$\rightsquigarrow$ squares to illustrate loss

Loss as function of residuals
$\rightsquigarrow$ strength of penalty?
$\rightsquigarrow$ symmetric?

Highlighted: loss for residuals shown on LHS

OPTIMIZATION

Resulting risk equivalent to **sum of squared errors (SSE)**:

$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\theta}^{\top} \mathbf{x}^{(i)} \right)^2$$

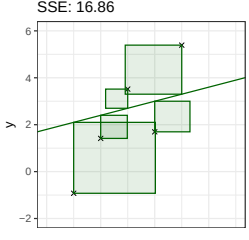
Consider example with $n = 5 \rightsquigarrow$ different models with varying SSE

OPTIMIZATION

Resulting risk equivalent to **sum of squared errors (SSE)**:

$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} \right)^2$$

Consider example with $n = 5 \rightsquigarrow$ different models with varying SSE

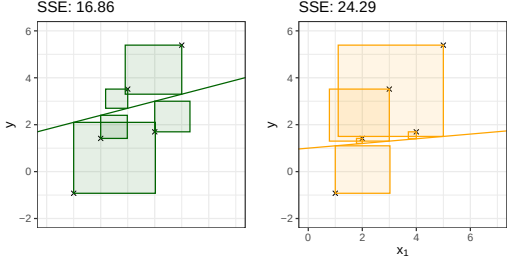


OPTIMIZATION

Resulting risk equivalent to **sum of squared errors (SSE)**:

$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} \right)^2$$

Consider example with $n = 5 \rightsquigarrow$ different models with varying SSE

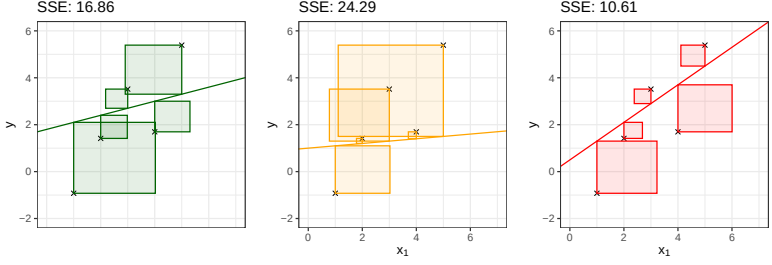


OPTIMIZATION

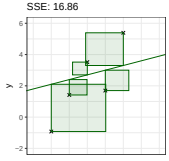
Resulting risk equivalent to **sum of squared errors (SSE)**:

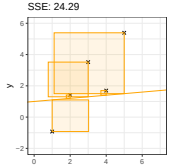
$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} \right)^2$$

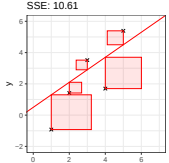
Consider example with $n = 5 \rightsquigarrow$ different models with varying SSE

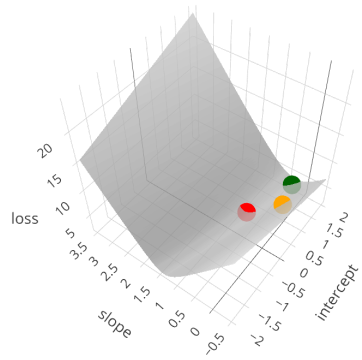


OPTIMIZATION



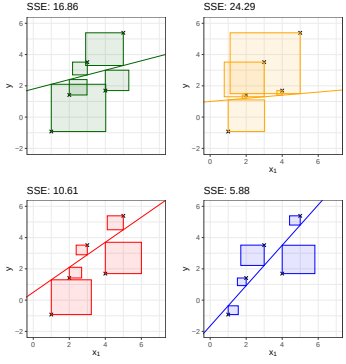


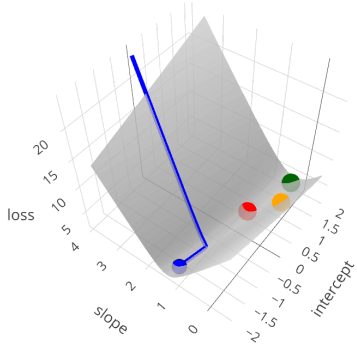




Intercept θ_0	Slope θ_1	SSE
1.80	0.30	16.86
1.00	0.10	24.29
0.50	0.80	10.61

OPTIMIZATION





Intercept θ_0	Slope θ_1	SSE
1.80	0.30	16.86
1.00	0.10	24.29
0.50	0.80	10.61
-1.65	1.29	5.88

Instead of guessing, of course, use **optimization**!

ANALYTICAL OPTIMIZATION

Special property of LM with *L2* loss: **analytical solution** available

$$\begin{aligned}\hat{\boldsymbol{\theta}} \in \arg\min_{\boldsymbol{\theta}} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) &= \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)}\right)^2 \\ &= \arg\min_{\boldsymbol{\theta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2\end{aligned}$$

Find via **normal equations**

$$\frac{\partial \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0$$

Solution: **ordinary-least-squares (OLS)** estimator

$$\hat{\boldsymbol{\theta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

ANALYTICAL OPTIMIZATION – PROOF

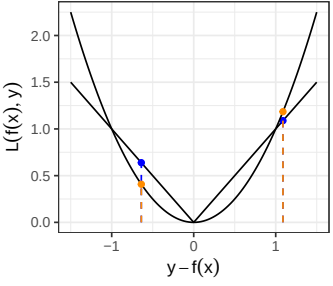
$$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \underbrace{\left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)}\right)}_{=:\epsilon_i}{}^2 = \| \underbrace{\mathbf{y} - \mathbf{X}\boldsymbol{\theta}}_{=:\boldsymbol{\epsilon}} \|_2^2; \quad \boldsymbol{\theta} \in \mathbb{R}^{\tilde{p}} \text{ with } \tilde{p} := p + 1$$

$$\begin{aligned}
0 &= \frac{\partial \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \quad \text{(sum notation)} \\
0 &= \frac{\partial}{\partial \boldsymbol{\theta}} \sum_{i=1}^{n} \epsilon_i^2 \quad \Big| \text{ sum \& chain rule} \\
0 &= \sum_{i=1}^{n} \frac{\partial \epsilon_i^2}{\partial \epsilon_i} \frac{\partial \epsilon_i}{\partial \boldsymbol{\theta}} \\
0 &= \sum_{i=1}^{n} 2\epsilon_i (-1) (\mathbf{x}^{(i)})^\top \\
0 &= \sum_{i=1}^{n} (y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)}) (\mathbf{x}^{(i)})^\top \\
\sum_{i=1}^{n} \boldsymbol{\theta}^\top \mathbf{x}^{(i)} (\mathbf{x}^{(i)})^\top &= \sum_{i=1}^{n} y^{(i)} (\mathbf{x}^{(i)})^\top \quad \Big| \text{ transpose} \\
\boldsymbol{\theta} \sum_{i=1}^{n} \underbrace{(\mathbf{x}^{(i)})^\top \mathbf{x}^{(i)}}_{1 \times 1} &= \sum_{i=1}^{n} \mathbf{x}^{(i)} y^{(i)} \\
\boldsymbol{\theta} &= \underbrace{\sum_{i=1}^{n} \underbrace{\left((\mathbf{x}^{(i)})^\top \mathbf{x}^{(i)}\right)^{-1}}_{1 \times 1} \underbrace{\mathbf{x}^{(i)}}_{\tilde{p} \times 1} \underbrace{y^{(i)}}_{1 \times 1}}_{\tilde{p} \times 1}
\end{aligned}$$

$$\begin{aligned}
0 &= \frac{\partial \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \quad \text{(matrix notation)} \\
0 &= \frac{\partial \|\boldsymbol{\epsilon}\|_2^2}{\partial \boldsymbol{\theta}} \\
0 &= \frac{\partial \boldsymbol{\epsilon}^\top \boldsymbol{\epsilon}}{\partial \boldsymbol{\theta}} \quad \Big| \text{ chain rule} \\
0 &= \frac{\partial \boldsymbol{\epsilon}^\top \boldsymbol{\epsilon}}{\partial \boldsymbol{\epsilon}} \cdot \frac{\partial \boldsymbol{\epsilon}}{\partial \boldsymbol{\theta}} \\
0 &= 2\boldsymbol{\epsilon}^\top \cdot (-1 \cdot \mathbf{X}) \\
0 &= (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^\top \mathbf{X} \\
0 &= \mathbf{y}^\top \mathbf{X} - \boldsymbol{\theta}^\top \mathbf{X}^\top \mathbf{X} \\
\boldsymbol{\theta}^\top \mathbf{X}^\top \mathbf{X} &= \mathbf{y}^\top \mathbf{X} \quad \Big| \text{ transpose} \\
\mathbf{X}^\top \mathbf{X} \boldsymbol{\theta} &= \mathbf{X}^\top \mathbf{y} \\
\boldsymbol{\theta} &= \underbrace{\underbrace{(\mathbf{X}^\top \mathbf{X})^{-1}}_{\tilde{p} \times \tilde{p}} \underbrace{\mathbf{X}^\top}_{\tilde{p} \times n} \underbrace{\mathbf{y}}_{n \times 1}}_{\tilde{p} \times 1}
\end{aligned}$$

Introduction to Machine Learning

Supervised Regression: Linear Models with $L1$ Loss



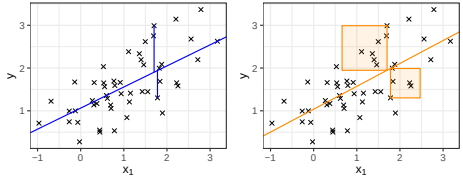
Learning goals

Understand difference between $L1$ and $L2$ regression

See how choice of loss affects optimization & robustness

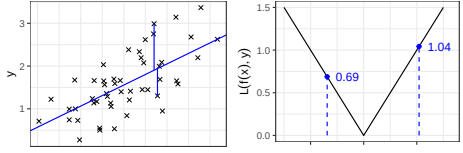
ABSOLUTE LOSS

*L*2 regression minimizes quadratic residuals – wouldn't **absolute** residuals seem more natural?

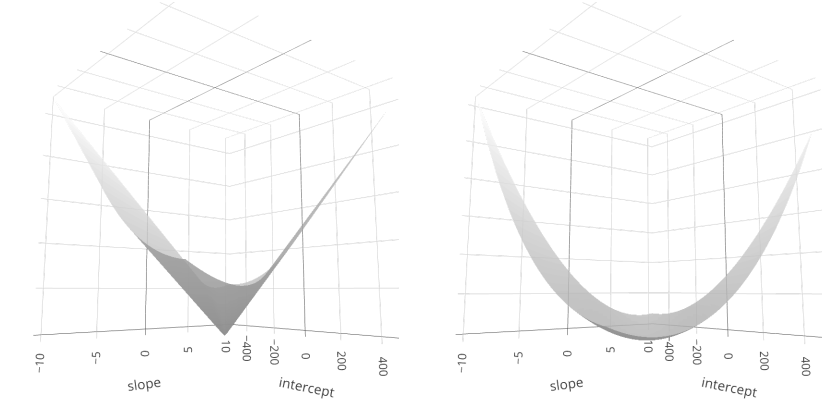


*L*1 **loss / absolute error / least absolute deviation (LAD)**

$$L(y, f(\mathbf{x})) = |y - f(\mathbf{x})|$$



*L*1 **VS** *L*2 **– LOSS SURFACE**



*L*1 loss (left) harder to optimize than *L*2 loss (right)

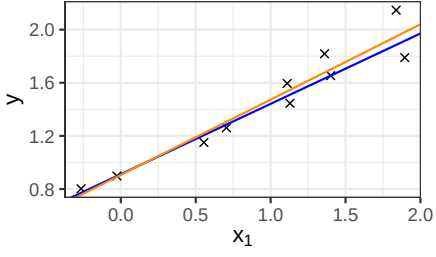
- Convex but **not differentiable** in $y - f(\mathbf{x}) = 0$
- No analytical solution

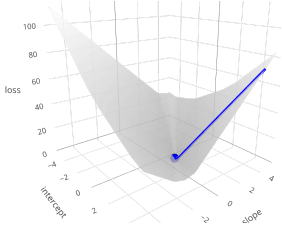
*L*1 **VS** *L*2 **– ESTIMATED PARAMETERS**

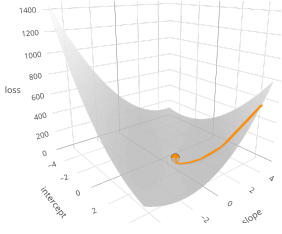
Results of *L*1 and *L*2 regression often not that different

Simulated data: $y^{(i)} = 1 + 0.5x_1^{(i)}$

	intercept	slope
*L*1	0.91	0.53
*L*2	0.91	0.57





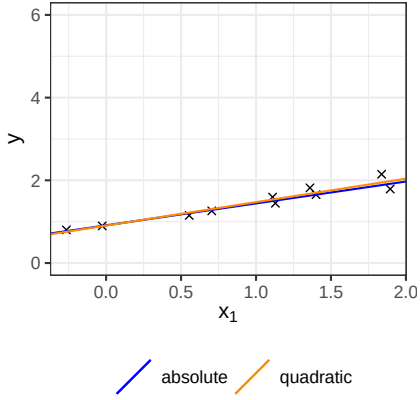


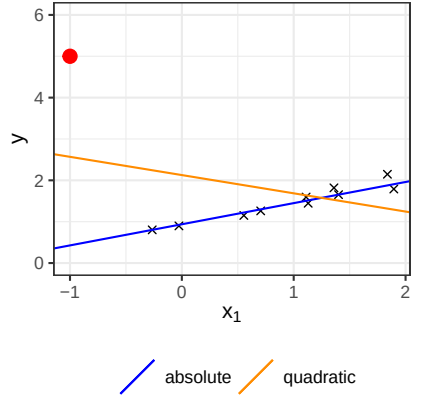
*L*1 **VS** *L*2 **– ROBUSTNESS**

*L*2 quadratic in residuals $\rightsquigarrow$ outlying points carry lots of weight

E.g., $3\times$ residual $\Rightarrow$ $9\times$ loss contribution

*L*1 more **robust** in presence of outliers (example ctd.):



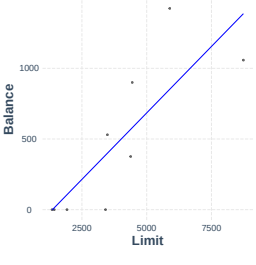


POLYNOMIAL REGRESSION

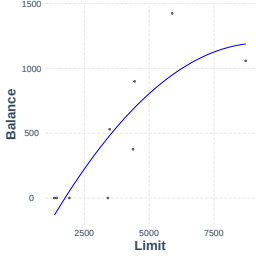
Adding polynomial terms to the linear combination leads to more flexible regression functions

Too high degrees can lead to overfitting (Details later)

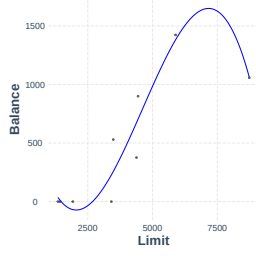
$Balance = \theta_0 + \theta_1 Limit$



$Balance = \theta_0 + \theta_1 Limit + \theta_2 Limit^2$

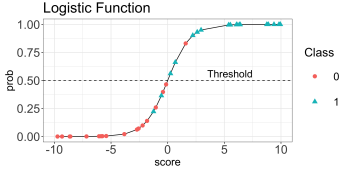


$Balance = \theta_0 + \theta_1 Limit + \theta_2 Limit^2 + \theta_3 Limit^3$



Introduction to Machine Learning

Supervised Classification: In a Nutshell



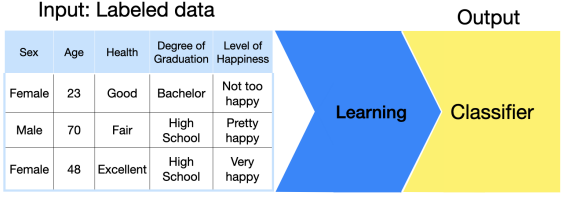
Learning goals

- Understand basic concept of classifiers
- Know concepts of probabilistic and scoring classifier
- Know distinction between discriminant and generative approach
- Understand ideas of logistic regression and Naive Bayes

CLASSIFICATION TASKS

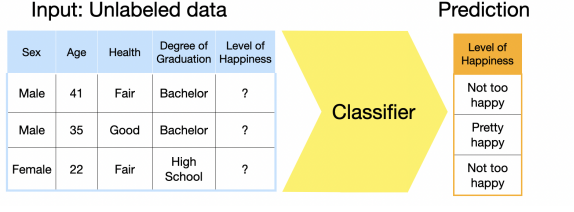
- Learn function that assigns categorical class labels to observations
- Each observation belongs to exactly one class
- The task can contain two (binary) or multiple (multi-class) classes

Training



Sex	Age	Health	Degree of Graduation	Level of Happiness
Female	23	Good	Bachelor	Not too happy
Male	70	Fair	High School	Pretty happy
Female	48	Excellent	High School	Very happy

Prediction

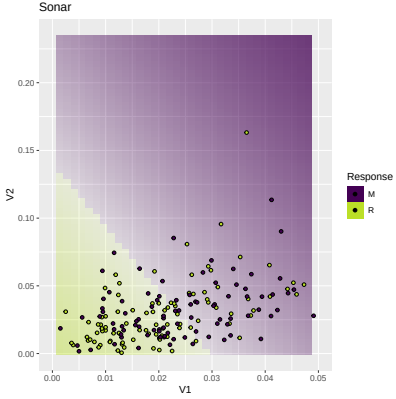


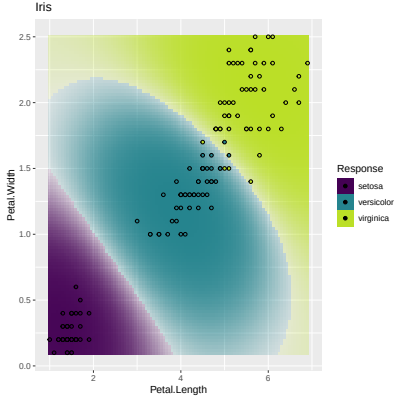
Sex	Age	Health	Degree of Graduation	Level of Happiness
Male	41	Fair	Bachelor	?
Male	35	Good	Bachelor	?
Female	22	Fair	High School	?

Level of Happiness
Not too happy
Pretty happy
Not too happy

BINARY AND MULTICLASS TASKS

The task can contain 2 classes (binary) or multiple (multiclass).





CLASSIFICATION TASKS

In classification, we aim at predicting a discrete output

$$y \in \mathcal{Y} = \{C_1, ..., C_g\}$$

with $2 \leq g < \infty$, given data $\mathcal{D}$.

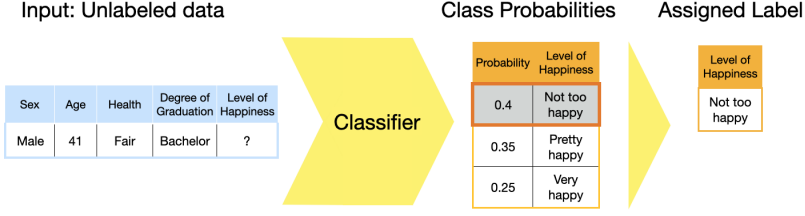
In this course, we assume the classes to be encoded as

$\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, +1\}$ (in the binary case $g = 2$)

$\mathcal{Y} = \{1, \ldots, g\}$ (in the multiclass case $g \geq 3$)

CLASSIFICATION MODELS

- For every observation a model outputs the probability (probabilistic classifier) or score (scoring classifier) of each class
- In the multi-class case, the class label is usually assigned by choosing the class with the maximum score or probability
- In the binary case, a class label is assigned by choosing the class whose probability or score exceeds a threshold value c



SCORING CLASSIFIERS

- Construct g **discriminant / scoring functions** $f_1, ..., f_g : \mathcal{X} \to \mathbb{R}$
- Scores $f_1(\mathbf{x}), \ldots, f_g(\mathbf{x})$ are transformed into classes by choosing the class with the maximum score

$$h(\mathbf{x}) = \underset{k \in \{1,...,g\}}{\arg\max}\ f_k(\mathbf{x}).$$

- For g=2, If the score exceeds a fixed threshold c, the sample is classified as class 1 (positive class), otherwise it is classified as class 0 (negative class).

PROBABILISTIC CLASSIFIERS

Construct g **probability functions**
$\pi_1, ..., \pi_g : \mathcal{X} \to [0, 1],\ \sum_i \pi_i = 1$
Probabilities $\pi_1(\mathbf{x}), \ldots, \pi_g(\mathbf{x})$ are transformed into labels by predicting the class with the maximum probability

$$h(\mathbf{x}) = \underset{k \in \{1,...,g\}}{\arg\max}\ \pi_k(\mathbf{x})$$

For $g = 2$ one $\pi(\mathbf{x})$ is constructed (note that it would be natural here to label the classes with $\{0, 1\}$)

PROBABILISTIC CLASSIFIERS / 2

Both scoring and probabilistic classifiers can output classes by thresholding (binary case) / selecting the class with the maximum score (multiclass)

Thresholding: $h(\mathbf{x}) := [\pi(\mathbf{x}) \geq c]$ or $h(\mathbf{x}) = [f(\mathbf{x}) \geq c]$ for some threshold c.

Usually $c = 0.5$ for probabilistic, $c = 0$ for scoring classifiers.

There are also versions of thresholding for the multiclass case

CLASSIFICATION APPROACHES

Two fundamental approaches exist to construct classifiers:
The **generative approach** and the **discriminant approach**.

They tackle the classification problem from different angles:

- **Generative** classification approaches assume a data-generating process in which the distribution of the features **x** is different for the various classes of the output y, and try to learn these conditional distributions:
  "Which y tends to have **x** like these?"

- **Discriminant** approaches use **empirical risk minimization** based on a suitable loss function:
  "What is the best prediction for y given these **x**?"

GENERATIVE APPROACH

The **generative approach** models $p(\mathbf{x}|y = k)$, usually by making some assumptions about the structure of these distributions, and employs the Bayes theorem:

$$\pi_k(\mathbf{x}) = \mathbb{P}(y = k \mid \mathbf{x}) = \frac{\mathbb{P}(\mathbf{x}|y = k)\mathbb{P}(y = k)}{\mathbb{P}(\mathbf{x})} = \frac{p(\mathbf{x}|y = k)\pi_k}{\sum_{j=1}^{g} p(\mathbf{x}|y = j)\pi_j}$$

Prior class probabilities π_k are easy to estimate from the training data.

Examples:

- Naive Bayes classifier

DISCRIMINANT APPROACH

The **discriminant approach** tries to optimize the discriminant functions directly, usually via empirical risk minimization.

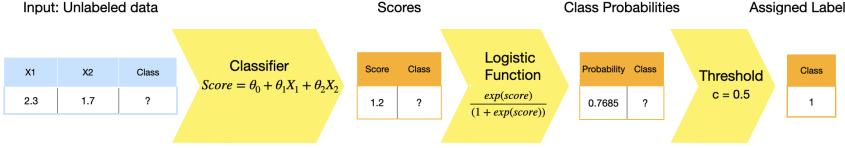
$$\hat{f} = \underset{f \in \mathcal{H}}{\arg\min}\, \mathcal{R}_{\text{emp}}(f) = \underset{f \in \mathcal{H}}{\arg\min} \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right).$$

Examples:

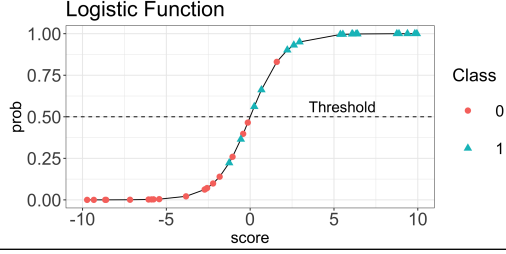
- Logistic regression (discriminant, linear)
- Neural networks
- Support vector machines

LOGISTIC REGRESSION

- Logistic regression is a **discriminant approach** for binary classification. It turns scores into probabilities with the logistic function.
- We just need to compute the probability for **one** class (usually class 1).
- If the probability exceeds a threshold value **c** $\Rightarrow$ class 1 is predicted.

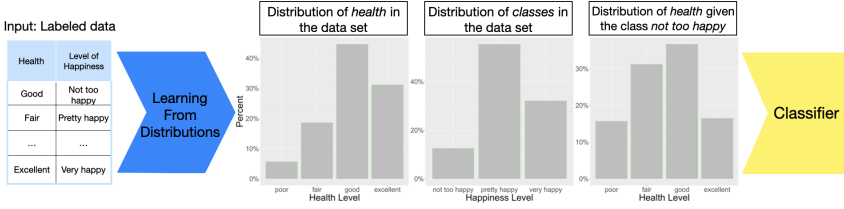


The logistic function puts all scores in order along an s-shaped line.



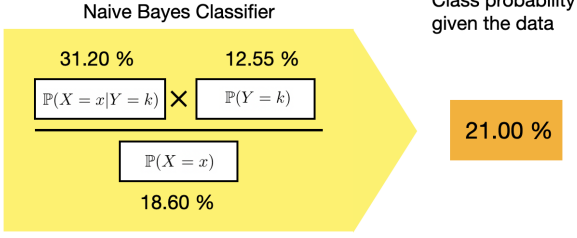
NAIVE BAYES

- Naive Bayes is a **generative multi-class approach**. It computes the class probability for each class based on the training data.
- It considers the data distribution on three different levels:
  - Marginal distributions $\mathbb{P}(X)$ of each feature (in the entire data set)
  - Marginal distribution $\mathbb{P}(Y)$ of classes (in the entire data set)
  - Conditional distributions $\mathbb{P}(X|Y)$ of each feature in each class



NAIVE BAYES / 2

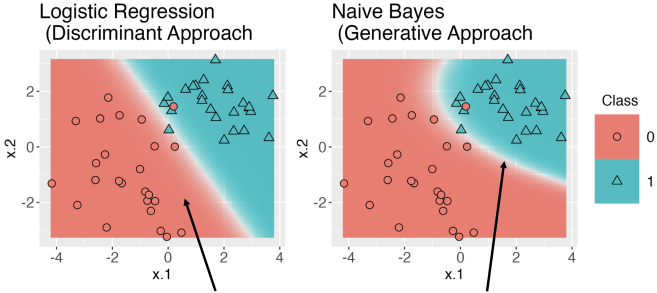
- Example: Class probability of “not too happy” given health = “fair”:



Decision Boundary

A decision boundary is a hypersurface or a boundary that separates the instances of different classes in a classification problem.

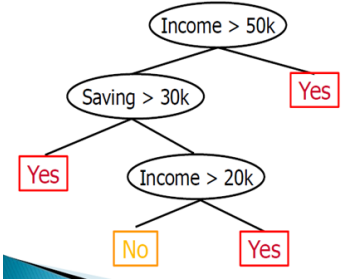
- **Linear boundary: A straight line (in 2D) or a hyperplane (in higher dimensions)**
- **Nonlinear boundary**

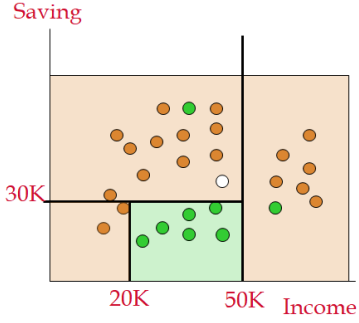


Decision Boundary

Decision tree

A series of splits along the feature axes, dividing the feature space into rectangular regions.





Decision Boundary

Neural networks

Highly non-linear and complex, depending on the architecture, activation functions, and weights of the network.