Supervised Classification

Usually we dont directly predict hard labels h( ). Scores / Probabilities are more informative and its much easier to work with continuos values in optimization

Setup — Categorial Label Prediction

Classification tasks aim at predicting a discrete output (finite number of classes) $y \in \mathcal{Y} = \{C_1, \dots, C_g\}$ with $2 \le g < \infty$ given Data D.

We encode $\mathcal{Y}$ as:
- binary case $g=2$: $\mathcal{Y} = \{0,1\}$ (better for probabil. models) or $\mathcal{Y} = \{-1,1\}$ (easier to interpret decision boundary).
- multiclass case $g>2$: $\mathcal{Y} = \{1, \dots, g\}$ (usually unordered classes; score/prob. for each class)

! Our model $f: \mathcal{X} \to \mathbb{R}^g$ outputs scores/probabilities and not classes.
- ↳ Continuous function is easier to optimize than discrete-valued functions.
- ↳ Scores/Probabilities contain more information and can be transformed into classes. But this transformation is non-injective

Scoring classifier

Construct g **discriminant/scoring functions** $f_1, \dots, f_g : \mathcal{X} \to \mathbb{R}$.

We choose the class with the maximum score (for each observation x(!)), i.e. $h(x) := \arg\max_{k \in \{1,\dots,g\}} f_k(x)$

Special case $g=2$: (always 1 DF)

One discriminant/scoring function is sufficient. $f(x) = f_1(x) - f_{-1}(x)$ (≙ "preference" score for positive class) (Note: $\mathcal{Y} = \{-1,1\}$ — here this encoding is very natural for positive vs. negative class)

$h(x) = \text{sgn}(f(x))$ and $|f(x)|$ is called **confidence**.

≥ Probabilistic classifier

Construct g **probability functions** $\pi_1, \dots, \pi_g : \mathcal{X} \to [0,1]$ with $\sum_{k=1}^{g} \pi_k = 1$.

We choose the class with the maximum score /Prob., i.e. $h(x) := \arg\max_{k \in \{1,\dots,g\}} \pi_k(x)$

Special case $g=2$:

One discriminant/scoring function is sufficient. $\pi(x) = 1 - \pi_0(x)$ (Note: $\mathcal{Y} = \{0,1\}$)

$h(x) = \mathbb{1}[\pi(x) \ge c]$ with c being a specified threshold.

Linear Classifier — Affine linear in features x

(if we allow transf. of feature space (e.g. polynom) we can handle non-linear classif. in original space (can learn non-linear boundary))

If the discriminant/scoring functions $f_1, \dots, f_g$ can be specified as linear functions, i.e. $g(f_k(x)) = w_k^T x + b_k$ with g being (optional) a rank-preserving, monotone transformation, then we call $f_1, \dots, f_g$ **linear classifiers**. (no transf. ⇒ w_k, b_k refer to params θ_k, but in general these are just some const. values / vectors so that g( ) is affine-linear)

The decision boundaries are hyperplanes.

$\forall x$ with score tie: $f_i(x) = f_j(x) \overset{g()}{\Leftrightarrow} w_i^T x + b_i = w_j^T x + b_j \Leftrightarrow (w_i - w_j)^T x + (b_i - b_j) = 0$ (or $x^T w_i = x^T w_j$)

Decision Region

A **decision region for class k** is defined as $\mathcal{X}_k := \{x \in \mathcal{X} \mid h(x) = k\}$

The **decision boundaries** are defined as hypersurfaces in $\mathcal{X}$ with tied maximal score:

$\{x \in \mathcal{X} \mid \exists i \ne j \in \{1,\dots,g\} : f_i(x) = f_j(x) \wedge \forall k \ne i,j : f_i(x) \ge f_k(x) \wedge f_j(x) \ge f_k(x)\}$ (e.g. $f_1 = f_2 > f_3$ or $f_1 = f_2 = f_3$)

Full indifference? Sol i) preference hierarchy with "≥" (easiest for g=2); Sol ii) random allocation.

Special case $g=2$ with threshold c the decision boundary is $\{x \in \mathcal{X} \mid f(x) = c\}$

Binary case: Convert scores or prob. to class outputs by thresholding $h(x) := [f(x) \ge c]$ ($\pi(x)$); standard $c=0$ for scores, $c=0.5$ for prob.

Approaches to construct classifiers

Generative Approach

Goal: Model joint data-generating process to obtain class prob. directly [of being in class k given feature vector x]. Model posterior probability $\pi_k(x)$ using Bayes

Idea: Assume that the data generating process for x depends on y.

Then we want to answer "Which y tends to have x like that?"

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

(denominator same for all classes), with π_j being a prior estimated from data, (estimate) by using e.g. relative frequencies.

⇒ Model posterior distributions, then choose best decision boundary based on that

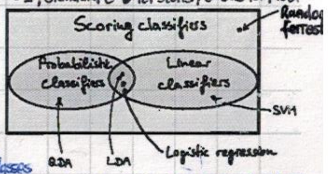
Discriminant Approach

Goal: Directly learn Decision boundary with ERM

Idea: Use empirical risk minimization with a suitable Loss-function

We want to answer "What is the best prediction for y given x?".

$$\hat{f} = \arg\min_{f \in \mathcal{H}} \mathcal{R}_{emp}(f) = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}))$$



Connections and differences of LDA, QDA and (Gaussian) NB

Assuming numerical features, cause LDA, QDA require that!

- Fit model conditional distr. of features $p(x \mid y=k)$ (for given class) as multivariate Gaussian
- QDA: $x \mid y=k \sim \mathcal{N}(\mu_k, \Sigma_k)$ LDA: $x \mid y=k \sim \mathcal{N}(\mu_k, \Sigma) \mid \Sigma_k = \Sigma$ (same cov. ∀ classes)
- ↳ LDA and (gaussian) NB are special cases! (G)NB: $x \mid y=k \sim \mathcal{N}(\mu_k, D_k) \mid \Sigma_k = D_k = \text{diag}(\sigma_k^2) = \begin{pmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_p^2 \end{pmatrix}$ In each class k it holds $\text{cov}(x_i, x_j) = 0 \Rightarrow \text{corr}() =$

We only have limited data n, if p gets "too large" for that then we may prefer simpler LDA approach

Reminder: LDA might be preferable over QDA in higher dimensions of features p↑↑ (much less params to est. than QDA)

Why? NB requires the least amount of parameter estimators for smallish k and large(r) p

Often NB is the best model of the three although the cond. independence assumption

Examples

Logistic (sigmoid) function $P(y=1 \mid x, \theta) = \pi(x \mid \theta)$ — for binary classfic: s( ) estimates prob. of class "1"

Logistic Regression (Discriminant)

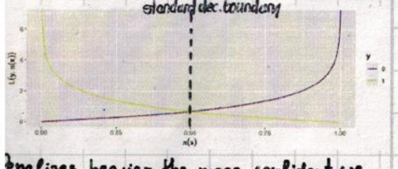
$\mathcal{H} = \{\pi : \mathcal{X} \to [0,1] \mid \pi(x) = s(\theta^T x)\}$ ($\theta^T x$: linear score)

with $s(\theta^T x) = \frac{\exp(\theta^T x)}{1 + \exp(\theta^T x)} = \frac{1}{1 + \exp(-\theta^T x)}$ (sigmoid function $\in [0,1]$)

Logistic/Bernoulli Loss function:

$L(y, \pi(x)) = -y \log(\pi(x)) + (1-y)\log(1 - \pi(x))$

Optimization is done numerically (GD). Log Regr. is linear classifier



Penalizes heavier the more confident we are in a wrong prediction! (loss → ∞)

not to confuse with Discriminant Approach!

Discriminant Analysis (Generative)

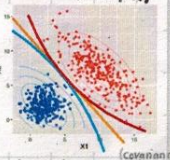
Only sensible for numerical features. Multivariate gaussian distribution in each class:

$p(x \mid y=k) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1} (x-\mu_k)\right)$

Linear discriminant analysis assumes that each class density is modeled as $x \mid y=k \sim \mathcal{N}(\mu_k, \Sigma)$ where the covariance Σ is equal for all classes.

Note: $\pi_k(x) \propto p(x \mid y=k) \cdot \pi_k \propto \exp(\theta_{0k} + x^T \theta_k)$

$\Rightarrow \log(\pi_k(x)) = \theta_{0k} + x^T \theta_k \Rightarrow$ LDA gives a linear classifier (linear decision boundary)



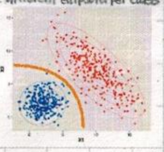
LDA: classes share same ellipsoid (covariance) exactly same gaussian shape, just different position μ_k

QDA: different ellipsoid (covariance) per class

Quadratic discriminant analysis assumes that each class density is modeled as $x \mid y=k \sim \mathcal{N}(\mu_k, \Sigma_k)$ where the covariances Σ_k do not need to be equal.

Note: $\log(\pi_k(x))$ defines a quadratic function in x, i.e. not linear.

⇒ quadratic decision boundary. But if quadratic terms cancel out, we can also get linear boundary!



Naive Bayes (Generative)

Features can be numerical / categorial or mixture!

Idea: $\pi_k(x) \approx \mathbb{P}(y=k \mid x) = \frac{p(x \mid y=k) \cdot p(y=k)}{p(x)} = \frac{p(x \mid y=k)\, \pi_k}{\sum_{j=1}^{g} p(x \mid y=j)\, \pi_j}$ (Priors easily estimated as rel. frequencies from training data), where $p(x \mid y=k) = \prod_{j=1}^{p} p(x_j \mid y=k)$ by NB assumption

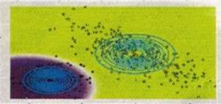
In Naive Bayes we assume that given class y the features are conditionally independent, i.e. $p(x \mid y=k) = \prod_{j=1}^{p} p(x_j \mid y=k)$ (joint cond. distr. (prob. for discrete x) = product, factorizes across feat.; $P((x_1, x_2, \dots, x_p) \mid y=k)$)

- For numerical features: We further assume that $x_j \mid y=k \sim \mathcal{N}(\mu_{jk}, \sigma_{jk}^2)$ (univariate gaussian distr.) **Gaussian Naive Bayes (GNB)** (We could choose other distribut. tha[n] …) (only need to estimate these univariate distr.)

This results in $x \mid y=k \sim \mathcal{N}(\mu_k, D_k)$ with D_k being a diagonal covariance matrix ($|D_k| = \prod_j \sigma_{jk}^2$). ⇒ GNB is a special case of QDA.

$p(x \mid y=k) = \frac{1}{(2\pi)^{p/2} |D_k|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T D_k^{-1} (x-\mu_k)\right) = \prod_{j=1}^{p} \frac{1}{\sqrt{2\pi\sigma_{jk}^2}} \exp\left(-\frac{1}{2}\frac{(x_j - \mu_{jk})^2}{\sigma_{jk}^2}\right)$ (Product of univariate gau[ssians])

$= \frac{1}{(2\pi)^{p/2} |D_k|^{1/2}} \exp\left(-\frac{1}{2}\sum_{j=1}^{p} \frac{(x_j - \mu_{jk})^2}{\sigma_{jk}^2}\right)$



GNB: different (cov.) concentrics, features uncorrelated

Categorial NB: $p(x_j \mid y=k) = \prod_m \rho_{kjm}^{[x_j = m]}$ (Always 1 active categ.: Indicator 1 for $x_j = m$, 0 else [no real prod]) bzw. $p(x \mid y=k) = \prod_{j=1}^{p} p(x_j \mid y=k)$ folgt daraus?

$\hat{\rho}_{kjm} = \frac{n_{kjm}}{n_k}$ (count of $x_j = m$ in class k / total count in class k)

- For categorical features: We further assume that $x_j \mid y=k$ is multinomial distributed, i.e. $p(x \mid y=k) \propto \prod_m \rho_{km}^{n_m}$ (not the same; n_m: in feature vector x)

with (n_m) / $\hat{\rho}_{kjm}$ being the (absolute)/relative frequency of category m in feature j restricted to class k. **Multinomial Naive Bayes** (Mult. coeff. $\frac{N!}{\prod_m n_m!}$ left out)

Note: The multinomial Naive Bayes classifier is a linear classifier.

↳ the posterior prob. $\pi_k(x_j = m, x_1 = \dots, x_2 = \dots, \dots) = 0$ just due to one factor in $\prod p(x_i \mid y=k)$ is 0! (Wipes out information about all other features)

! If $\exists m : \rho_{kjm} = 0$, then we replace the 0 with a small value (Laplace Smoothing)

Exercise Classification: Generative Approaches

1. Proof that **LDA** is a linear classifier — model class distribution (feature) as $x|y=k \sim \mathcal{N}(\mu_k, \Sigma)$

NB assumes $P(x|y=k) = \prod_{j=1}^{p} P(x_j|y=k)$ feature independence

Starting point is modelling posterior probability [of being in class k given feature vector x]:

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

proportional, insert $x|y=k \sim \mathcal{N}(\mu_k, \Sigma)$ without normalise constant — only consider this — same for all classes

$$\propto \pi_k \cdot p(x|y=k) \propto \pi_k \exp\left(-\tfrac{1}{2}(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)\right) = \pi_k \exp\left(\underbrace{-\tfrac{1}{2}x^T\Sigma^{-1}x}_{\text{quadratic form in } x \text{ resulting in scalar (dep. of } x)} - \underbrace{\tfrac{1}{2}\mu_k^T\Sigma^{-1}\mu_k}_{\text{some const. scalar (indep. of } x)} + \underbrace{x^T\Sigma^{-1}\mu_k}_{\text{scalar} \Rightarrow x^T\Sigma^{-1}\mu_k = (x^T\Sigma^{-1}\mu_k)^T = \mu_k^T(\Sigma^{-1})^T x}\right)$$

$\Sigma^{(-1)}$ symmetr. Kovarianzmatrix $\Sigma^{(-1)T} = \Sigma^{(-1)}$

$\downarrow \tfrac{1}{2}x^T\Sigma^{-1}\mu_k + \tfrac{1}{2}\mu_k^T\Sigma^{-1}x = x^T\Sigma^{-1}\mu_k$ or ...

$$= \exp\left(\log\pi_k - \tfrac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \underbrace{x^T\Sigma^{-1}\mu_k}_{\text{linear in } x}\right)\exp\left(-\tfrac{1}{2}x^T\Sigma^{-1}x\right)$$ but independent of class k since Σ is by assumption identical for all classes!

$$= \exp(\underbrace{w_{0k}}_{b_k} + x^T w_k)\exp\left(-\tfrac{1}{2}x^T\Sigma^{-1}x\right) \propto \exp(w_{0k} + x^T w_k)$$

Now with $\log(\cdot)$ transformation we get $\log(\pi_k(x)) \propto w_{0k} + x^T w_k$

by defining $w_{0k} := \log\pi_k - \tfrac{1}{2}\mu_k^T\Sigma^{-1}\mu_k$ and $w_k := \Sigma^{-1}\mu_k$

would still work with scalar $\exp(-\frac{1}{2}x^T\Sigma^{-1}x) > 0$.

transformed score-fct. is linear

here $w_{0k}, w_k \triangleq$ params θ_{0k}, θ_k ignoring some multipl. constants

Linear decision boundary $\log(\pi_a(x)) = \log(\pi_b(x)) \Leftrightarrow x^T\underbrace{(w_a - w_b)}_{\Sigma^{-1}(\mu_a-\mu_b)} + (b_a - b_b) = 0$

Contrast this with **QDA** Model class (feature) distribution as $x|y=k \sim \mathcal{N}(\mu_k, \Sigma_k)$

ignoral! $|\Sigma|^{-1/2}$ for LDA as a constant, but here it depends on class k $\Rightarrow$ therefore no further simplifications possible

$$\pi_k(x) \propto \ldots \pi_k |\Sigma_k|^{-1/2}\exp\left(-\tfrac{1}{2}x^T\Sigma_k^{-1}x - \tfrac{1}{2}\mu_k^T\Sigma_k^{-1}\mu_k + x^T\Sigma_k^{-1}\mu_k\right)$$

(Two-class Problem)

quadratic decision boundary: Assume indifference betw. classes a and b

$$\log(\pi_a(x)) = \log(\pi_b(x)) \Leftrightarrow -\tfrac{1}{2}x^T(\Sigma_a^{-1} - \Sigma_b^{-1})x + x^T(\Sigma_a^{-1}\mu_a - \Sigma_b^{-1}\mu_b) + b = 0$$

scalar with cond. rest

If $\Sigma_a^{-1} = \Sigma_b^{-1}$ this simplifies to $x^T(\Sigma^{-1}(\mu_a - \mu_b)) + b = 0$ → LDA

$\Rightarrow$ for same Σ across classes quadrat. form disappears & we get linear boundary!

$\log(\cdot) = \log\pi_k - \frac{1}{2}\log|\Sigma_k| - \underbrace{\frac{1}{2}\mu_k^T\Sigma_k^{-1}\mu_k}_{\text{const. scalar (depending on k)}} + \underbrace{x^T\Sigma_k^{-1}\mu_k}_{\text{"linear" in } x} - \underbrace{\frac{1}{2}x^T\Sigma_k^{-1}x}_{\text{Quadratic form in } x}$

Same for Gaussian naive Bayes with numerical features. Just replace Σ_k with $\text{diag}(\sigma_k^2)$

Parameter Estimators in LDA and QDA

LDA: $\hat{\pi}_k = \frac{n_k}{n}$ where n_k is number of class k-observations

QDA: $\hat{\pi}_k = \frac{n_k}{n}$ prior probabilities $\triangleq$ class frequencies

$$\hat{\mu}_k = \frac{1}{n_k}\sum_{i:y^{(i)}=k}^{n_k} x^{(i)}$$ for average feature vector of class

$$\hat{\mu}_k = \frac{1}{n_k}\sum_{i:y^{(i)}=k} x_k^{(i)}$$

$$\hat{\Sigma} = \underbrace{\frac{1}{n-g}\sum_{k=1}^{g}\sum_{i:y^{(i)}=k}^{n_k}(x^{(i)}-\hat{\mu}_k)(x^{(i)}-\hat{\mu}_k)^T}_{\text{average everything across all classes to get same cov. matrix for all classes}} = \underbrace{\frac{1}{\sum_{k=1}^{g}(n_k-1)}}_{=1/(n-g)}\sum_{k=1}^{g}\sum_{i:y^{(i)}=k}^{n_k}(x^{(i)}-\hat{\mu}_k)(x^{(i)}-\hat{\mu}_k)^T$$

$\Rightarrow$ estimate "pooled cov. matrix"

$$\hat{\Sigma}_k = \frac{1}{n_k-1}\sum_{i:y^{(i)}=k}^{n_k}(x^{(i)}-\hat{\mu}_k)(x^{(i)}-\hat{\mu}_k)^T$$

Overall: $(k-1) + k\cdot p + \underbrace{k}_{\text{different cov. in each class}}\cdot\frac{p(p+1)}{2}$ parameters to estimate (!)

Gaussian Naive Bayes

GNB: $(k-1) + k\cdot p + k\cdot p = (k-1) + 2kp$

Cov. is Diagonal Matrix

Overall: $\underbrace{(k-1)}_{\hat{\pi}_k\text{s}} + \underbrace{k\cdot p}_{\hat{\mu}_k\text{s}} + \underbrace{\frac{p(p+1)}{2}}_{\hat{\Sigma}} = \frac{p^2-p}{2} + p$ parameters to estimate

- $\hat{\pi}_k$s: one for each class
- $\hat{\mu}_k$s: p-times on top as μ_k entails p-averages (one for each feature)
- $\frac{p^2-p}{2}$: cov. matrix symmetric
- $+p$: inner feature variances on diagonal

p: features

k: classes, may also use g for that

Note] For QDA we have to estimate way more parameters because of estimation of separate cov. matrix per class

$\Rightarrow$ Weird side effect: LDA might perform better than QDA in higher dimensions even if data-generating process matches QDA assumptions (fully or better)!!

class error ↗ cross-validation — in terms of feature space, e.g. 100+ features ($p \geq 100$) — in every dimension

The Reason is that we have a limited amount of data (n) and therefore we can't estimate QDA params perfectly. If n too small and/or p too large LDA maybe preferred

→ Least amount of parameter estimators in almost all scenarios

Then it should not surprise that Gaussian NB often works best eventhough assumption of cond. feature independence is usually unrealistic!

Exercise Logistic Regression (Discriminant Approach)

hard label prediction $\hat{y}=1 \Leftrightarrow \pi(x) \geq \alpha$, $\alpha \in (0,1)$

Proof that decision boundary is a linear hyperplane — Remember: Logistic regression estimates prob. $p(y=1|x,\theta) = \pi(x|\theta) = \frac{1}{1+\exp(-\theta^T x)}$

$$\pi(x) = \frac{1}{1+\exp(-\theta^T x)} = \alpha \Leftrightarrow 1+\exp(-\theta^T x) = \frac{1}{\alpha} \Leftrightarrow \exp(-\theta^T x) = \frac{1}{\alpha} - 1 \Leftrightarrow \underbrace{\theta^T x}_{\text{linear comb. of features}} = \underbrace{-\log\left(\frac{1}{\alpha}-1\right)}_{\text{scalar } b \text{ depending on } \alpha}$$ $\hat{y}=1$ for all points on or above this plane

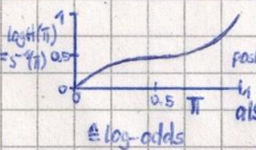
For $\alpha = 0.5$: $\theta^T x = -\log(2-1) = -\log(1) = 0$, so $\theta^T x \geq 0 \overset{\text{if}}{\Leftrightarrow} \hat{y}=1$ "1" halfspace

$\theta^T x < 0 \Leftrightarrow \hat{y}=0$ "0" halfspace

Logistic Regression

Regression mit binärer Zielgrösse (codiert als 1/0) $\Rightarrow$ Ein Parameter $\pi^{(i)} = \mathbb{P}(y^{(i)}=1) = \mathbb{E}(y^{(i)})$ bzw. $\mathbb{P}(y_i=1|x_i) = \mathbb{E}(y_i|x_i)$

Wir interessieren uns dafür, wie Einflussgrößen X mit Erwartungswert bzw. Wahrscheinlichkeit für das "1" Ereignis zusammenhängen



positive logits indicate probab. > 0.5 and vice versa

Bernoulli-Verteilung: $y^{(i)}|x^{(i)} \sim B(\pi^{(i)})$ $i=1,\dots,n$ unabhängig

also $s^{-1}(\pi(x))$ in ML, features linear on logits $= B^T x$ bzw. $\theta^T x$

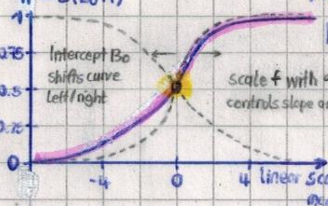
[GLM] Logistic Regression is simply Bernoulli-Regression with its natural Link Function, the **Logit function** $\log\left(\frac{\pi^{(i)}}{1-\pi^{(i)}}\right) = x^{(i)T}B = \eta_i$ (linear predictor / score), $r^{-1}(\pi^{(i)})$

$\hookrightarrow$ natural response $r(x^{(i)T}B) = \pi_i = \frac{\exp(x^{(i)T}B)}{1+\exp(x^{(i)T}B)} = \frac{1}{1+\exp(-x^{(i)T}B)}$

We could use other sigmoid types, and CDF would fit requirement, but logistic function is the most natural

Logistic function (a sigmoid) squashes linear predictor / estimated linear scores to $[0,1]$ probability interval — rank preserving "Logistic function", also $s(B^T x)$ in ML, $s(f(x))$ with score $f(x) = \theta^T x$

$\pi = s(B_0 + f)$



linear score $f = x^T B$ (B-changes captured by x-axis, $\hookrightarrow$ different $B_1 \dots B_p$ same plot shape) — may plot without intercept B_0!

We always get sigmoid shape with turning point at (score = 0, $\pi = 0.5$), here maximal slope in region of highest dec. uncertainty

$s(0) = \frac{e^0}{1+e^0} = 1/2$ $\;\hat{=}$ log-odds $= 0$, $\log\left(\frac{0.5}{1-0.5}\right) = \log(1) = 0$

natural decision boundary at $\pi = 0.5$ s.t. scores > 0 ($\to$ log-odds > 0) lead to class "1" and vice versa

however there are usecases where we don't base decision boundary on "natural" $\pi = 0.5$. Identical at boundary $\pi = 0.5$: $L(y=0|y=1) = L(y=1|y=0)$ equal treatment of misclassified observations (classes)

$f = x^T B$ as x-Axis captures all linear comb. of features into one dimension

In 1D we could also plot feature on x-Axis, then different B_1 change slope!

$\hookrightarrow$ Still sigmoidal shape with turning point at ($\pi = 0.5$, but possibly $x_1 >/< 0$)

$\Rightarrow$ have to group them to analyze, or $\varepsilon_i = 0$ or 1 for hard $\hat{y}_i$; very small/large residuals at edges, medium in the middle; always $\varepsilon_i = \hat{\pi}_i$ or $1-\hat{\pi}_i$ and depend in size on curve location

GLM vs. classical LM: no residuals by model design & useless to analyze (follow no distribution)

Es besteht ein linearer Zusammenhang mit den Log-Odds! Interpretation ist schwieriger als bei klassischem linearen Modell für marginale c.p. Effekte von Δx

$\mathbb{P}(y=1|x) = \mathbb{E}(y|x) = \frac{\exp(x^T B)}{1+\exp(x^T B)}$ $\;\overset{q = \frac{p}{1-p}}{\Rightarrow}\;$ $q(y=1|x) = \dots \exp(x^T B)$ estimated odds of class "1" for given feature vector x

If we define $x^{+k} = x + (0,0,\dots,0,\underset{\text{index }k}{1},0,\dots,0,0)$ then we get $q(y=1|x^{+k}) = \exp(B_k)\, q(y=1|x)$ — multiplicative odds change when x_k increases by one unit

$\log(q(y=1|x^{+k})) = \log(q(y=1|x)) + B_k$ — additive log odds change (also odds ratio change by factor $\exp(B_k)$)

$\boxed{\log(q(y=1|x)) = x^T B}$ Logistic Regression assumes linear structure in X for the log-odds of positive class

don't misinterpret $\exp(B^k)$ as relative risk!

Only $\approx$ risk if "1" is very rare $\Rightarrow q(\,) \approx P(\,)$ ($1-P(\,) \approx 1$)

MLE estimator in logistic function

$\hat{B}_{MLE} = \arg\max \ell(B)$ (log-Likelihood) with $L(B) = \prod_{i=1}^{n} r(x^{(i)T}B)^{y_i}(1-r(x^{(i)T}B))^{1-y_i}$ and $\ell(B) = \log L(B) = \sum y_i \log(r(x^{(i)T}B)) + (1-y_i)\log(1-r(x^{(i)T}B))$

score-fct. $s(B) = \frac{\partial \ell}{\partial B} = \sum_{i=1}^{n}\left(y_i \frac{r(x^{(i)T}B)(1-r(x^{(i)T}B))x^{(i)}}{r(x^{(i)T}B)} + (1-y^{(i)})\frac{-r(x^{(i)T}B)(1-r(x^{(i)T}B))x^{(i)}}{1-r(x^{(i)T}B)}\right) = \dots \sum_{i=1}^{n}\left(y^{(i)}x^{(i)} - r(x^{(i)T}B)x^{(i)}\right) = \sum\left(y^{(i)} - r(x^{(i)T}B)\right)x^{(i)}$

$r' = r(1-r)$ with knowledge of $r(\,) = \frac{\exp(x^T B)}{1+\exp(x^T B)} = \frac{1}{1+\exp(-x^T B)} = (1+\exp(-x^T B))^{-1}$

without inner function chain rule (just taking the (-1), not x for now): $r' = \frac{-(\exp(-x^T B)\cdot(-1))}{(1+\exp(-x^T B))^2} = \frac{(1+\exp(\,)-1)\cdot 1}{(1+\exp(\,))^2} = r(1-r) \overset{\text{here}}{=} \pi(1-\pi)$

(-1) weil nachdiff $(-\exp(\,))$!

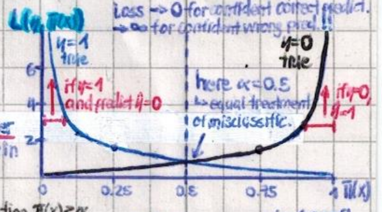
$\Rightarrow$ For MLE parameter estimator: $s(\hat{B}_{MLE}) = \sum_{i=1}^{n}(y^{(i)} - r(x^{(i)T}\hat{B}_{MLE}))x^{(i)} = 0$

$\delta_{kk} \hat{=}$ k-th diag. elem. of inverse Fisher matrix at point $\hat{B}_{MLE}$ $\mathcal{I}^{-1}(\hat{B})$

Wald confidence interval $\hat{B}_k \pm \hat{\sigma}_{\hat{B}_k} z_{1-\alpha/2}$ with $\hat{\sigma}_{\hat{B}_k} = \sqrt{\delta_{kk}}$

For odds-ratios change-factor we get transformed CI $\exp[\hat{B}_k \pm \hat{\sigma}_{\hat{B}_k} z_{1-\alpha}]$

With standard threshold $\pi = 0.5$



Log-Loss penalizes heavier the more confident we are in a wrong prediction

Prob. estimate $P(y=1|x,\theta)$ $\;\hat{y} \lessgtr^{1}_{0} \Rightarrow$ hard label choice: $h(x^{(i)}|\hat{\theta},\alpha) = \mathbb{I}_{[\alpha,1]}\left(\frac{1}{1+\exp(-\hat{\theta}^T x^{(i)})}\right)$ with $\alpha \in (0,1)$, typically $\alpha = 0.5$ — threshold for class "1" prediction $\pi(x) \geq \alpha$

Hypothesis Space: $\mathcal{H} = \{\pi : \mathcal{X} \to [0,1] \mid \pi(x) = s(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)},\ \theta \in \mathbb{R}^{p+1}\}$ $\;\left(\hat{\pi} = \arg\min_{\pi \in [0,1]^n} L(y,\pi)\right)$

Predicted score (linear) with $f(x|\theta) = \theta^T x$ and $L(y,f) = -yf + \log(1+\exp(f))$ — convex in θ: f linear and log-sum-exp $\log(e^0 + e^f)$

Logistic Regression in Machine Learning ERM requires suitable loss function to find optimal parameters $\hat{\theta} = \arg\min_{\theta \in \Theta} R_{emp}(\theta) = \arg\min_{\theta} \sum_{i=1}^{n} L(y^{(i)}, f(x^{(i)}|\theta))$

$\mathbb{P}(y^{(1)},\dots,y^{(n)}) = \prod_{i=1}^{n} \mathbb{P}(y^{(i)} = y^{(i)})$ $\searrow$ with $y^{(i)} \in \{0,1\}$ for negative/positive class ($\{0,1\}$ encoding)

Start with Likelihood $L(\theta) = \prod_{i:y^{(i)}=1} \pi(x^{(i)}|\theta) \prod_{i:y^{(i)}=0}(1-\pi(x^{(i)}|\theta)) = \prod_{i=1}^{n} \pi(x^{(i)}|\theta)^{y^{(i)}}(1-\pi(x^{(i)}|\theta))^{1-y^{(i)}}$

cleverly combining the two cases by using exponents. Only one of the two factors is not 1 ("active") depending if $y^{(i)} = 1$ or $= 0$

Now log to turn nasty product into nice sum $\ell(\theta) = \log(L(\theta)) = \sum_{i=1}^{n}\left(y^{(i)}\log(\pi(x^{(i)}|\theta)) + (1-y^{(i)})\log(1-\pi(x^{(i)}|\theta))\right)$ maximize this to find $\hat{\theta}$

$\hat{\theta}$ maximizes Likelihood & minimizes R_{emp} of model

$-\dots$ to turn this into minimization problem: $-\ell(\theta) = \sum_{i=1}^{n} -y^{(i)}\log(\pi(x^{(i)}|\theta)) - (1-y^{(i)})\log(1-\pi(x^{(i)}|\theta))$

$\log(\pi) = \log((1+\exp(-f))^{-1}) = -\log(1+\exp(-f))$

$\log(1-\pi) = \log\left(\frac{\exp(-f)}{1+\exp(-f)}\right) = -f - \log(1+\exp(-f))$

also sometimes called cross-entropy loss, works beyond Log. Reg. for all types of binary classification probabilistic models with Discriminant ERM approach

$L(y,\pi) = \dots (1-y)f + \log(1+\exp(-f))$; $\log(1+\exp(f)) = -f + \log(1+\exp(f))$

$\Rightarrow$ **Bernoulli/Binomial/Log-Loss** $L(y,\pi) = -y\log(\pi) - (1-y)\log(1-\pi)$ with logit link $\pi(x) = \frac{1}{1+\exp(-f(x))} = s(f(x))$ $\Leftrightarrow$ score based $L(y,f) = -yf + \log(1+\exp(f))$

Convex ✓ Logistic Regression has unique solution under simple regularity conditions, so design matrix X must have full rank and data not perfectly linearly separable

[also score $f = \theta^T x$ linear] numerical optimization with 2nd order fct. e.g. Newton-Raphson — if linearly perfectly separable → unbounded, regularize to fix it

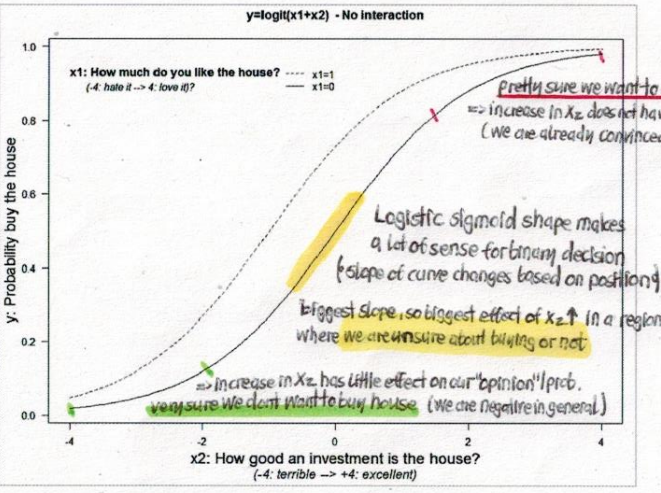
Interpretation von logistischer Regression

Die folgenden Aussagen gelten für eine Einflussgröße x_k bei Festhalten aller anderen Einflussgrößen (c.p.)

- Erhöht sich x_k um eine Einheit, so ändert sich die logarithmierte Chance von Y um β_k.
- Wenn x_k um eine Einheit steigt, so ändert sich die Chance von Y um den Faktor $\exp(\beta_k)$.
- Das Odds Ratio (Chancenverhältnis) zwischen Y bei x_k und Y bei $x_k + 1$ ist $\exp(\beta_k)$.

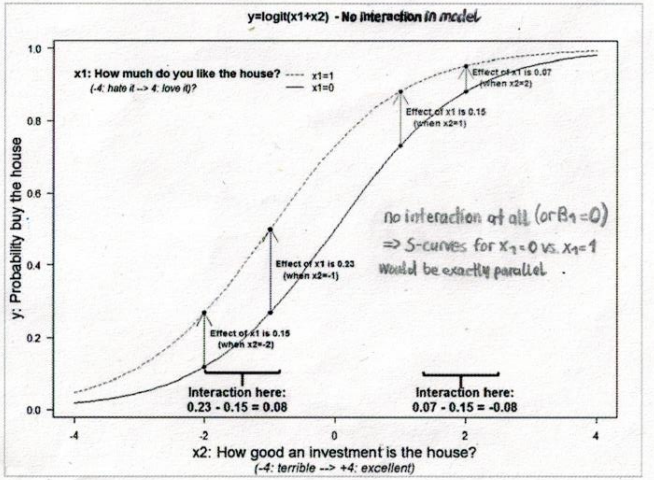
W'keit	0.01	0.05	0.1	0.3	0.4	0.5	0.6	0.7	0.9	0.95	0.99
Odds	1/99	1/19	1/9	3/7	2/3	1	1.5	7/3	9	19	99
Log odds	-4.6	-2.9	-2.2	-0.85	-0.41	0	0.41	0.85	2.2	2.9	4.6

Logistische Regression: Logistische Link-Funktion



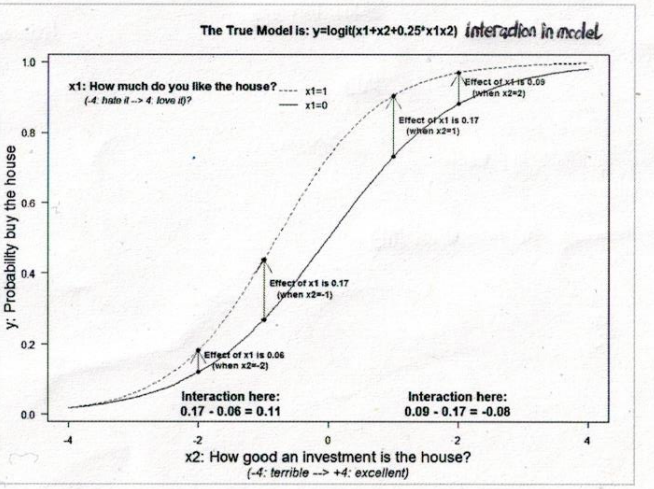
Source: `http://datacolada.org/57`

Logistische Regression: Interpretation von Interaktionen



Source: `http://datacolada.org/57`

Logistische Regression: Interpretation von Interaktionen

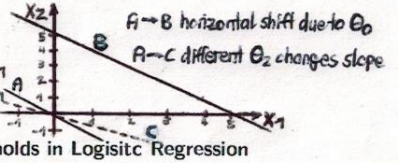


Source: `http://datacolada.org/57`

Linear classifiers like logistic regression learn a decision boundary that takes the form of a (linear) hyperplane.
Hyperplanes can be defined by equations $\theta^T x = b \Leftrightarrow \theta^T x - b = 0$ with coefficients θ and scalar $b \in \mathbb{R}$.

In order to see that such expressions actually describe hyperplanes, consider $\theta^\top \mathbf{x} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 = 0$. Sketch the hyperplanes given by the following coefficients and explain the difference between the parameterizations:

A • $\theta_0 = 0, \theta_1 = \theta_2 = 1$ $x_1 + x_2 = 0$, $x_2 = -x_1$
B • $\theta_0 = -5, \theta_1 = \theta_2 = 1$ $-5 + x_1 + x_2 = 0$, $x_2 = +5 - x_1$
C • $\theta_0 = 0, \theta_1 = 1, \theta_2 = 2$ $x_1 + 2x_2 = 0$, $x_2 = -\frac{1}{2} x_1$



Exercise 3: Decision Boundaries & Thresholds in Logisitc Regression

Learning goals

1) Understand that logistic regression finds a linear decision boundary
2) Get a feeling for how parameterization changes predicted probabilities

In logistic regression (binary case), we estimate the probability $p(y = 1|\mathbf{x}, \theta) = \pi(\mathbf{x}|\theta)$. In order to decide about the class of an observation, we set $\hat{y} = 1$ iff $\pi(\mathbf{x}|\theta) \geq \alpha$ for some $\alpha \in (0, 1)$.

Show that the decision boundary of the logistic classifier is a (linear) hyperplane.

Hint

Derive the value of $\theta^\top \mathbf{x}$ (depending on α) starting from which you predict $\hat{y} = 1$ rather than $\hat{y} = 0$.

Below you see the logistic function for a binary classification problem with two input features for different values $\theta^\top = (\theta_1, \theta_2)^\top$ (plots 1-3) as well as α (plot 4). What can you deduce for the values of θ_1, θ_2, and α? What are the implications for classification in the different scenarios?

Derive the equation for the decision boundary hyperplane if we choose $\alpha = 0.5$.

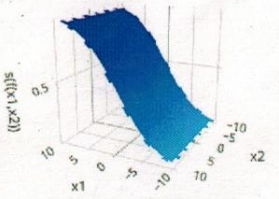
Explain when it might be sensible to set α to 0.5:

$\alpha = 0.5$: Losses of misclassified observations are treated equally ↑ Log-Loss $L(y,\pi) = -y\log(\pi) - (1-y)\log(1-\pi)$
i.e. $L(\hat{y}=0|y=1) = -\log(\pi)$ and $L(\hat{y}=1|y=0) = -\log(1-\pi)$ are identical at boundary $\pi(x|\theta) = 0.5$!

$\alpha >> 0.5$: Choose this if we need be careful with class "1" predictions
for example when decision triggers a costly/dangerous therapy, so we want to minimize false-positives
Vice versa: Go $\alpha << 0.5$ if we want to heavily emphasize penalty on False-Negatives, e.g. airport weapon security

Plot (1)

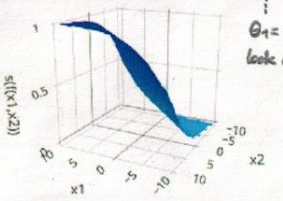
hypersurface is
- symmetric parallel across x_2-axis
  => $\theta_2 = 0$, x_2 has zero effect
- $\theta_1 \approx 0.5$ by visual cue



Plot (2)

- both feature dimensions affect logistic function, to equal degree. => $\theta_1 = \theta_2$; hypersurface not tilted in either direction
- We also see that $\theta_1, \theta_2 > 0$ as $s(\ ) \uparrow$ with $x_1 \uparrow$, $x_2 \uparrow$

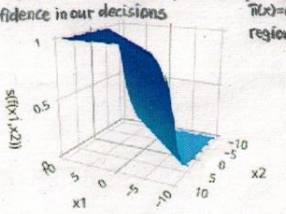
$\theta_1 = -\theta_2$ would look differently



Plot (3)

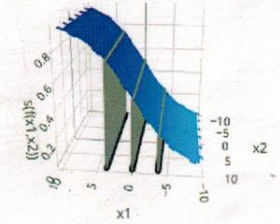
Very similar to (2) with minor differences:
- steeper logistic fct. => larger abs. θ_1, θ_2 values
- sharper separation between class predictions, less in the more confidence in our decisions

$\pi(x) = 0.5$ region



Plot (4)

This is Plot (1) with different α-thresholds visualised
- midline at $x_1 = 0$ corresponds to $\alpha = 0.5$
- Other hyperplanes ≙ higher (left) and lower (right) thresholds



misledding slices, just vertical
- it appears as if x_1 is const. at thresholds in general
- actually its just a geometric representation which looks like this here because of $\theta_2 = 0$