MAP 534
Introduction to machine learning

Linear models for classification I

Alain Durmus

A quick clarification regarding homework and Labs

- No librairies can be used for the homeworks.
- Extension for the first one to Monday.
- Students which would have submitted their homework in due date will have a bonus for the next one +2 points.
- Correction for some exercises/labs will be provided.

Outline

Introduction to classification: setting

- We consider a supervised setting.
- Decide how to encode inputs and outputs: this defines the input space X, and the output space Y.
- Here we consider specifically the classification problem: Y is a finite set,

$$\mathsf{Y} = \{1, \dots K\}\ ,\ \text{in most of this lecture even } \mathsf{Y} = \{0, 1\} \text{ (or \{-1,1\})}\ . \quad (1)$$

- In this lecture, we apply the three machine learning paradigms to address:

$$y_{\mathsf{pred}} = \mathscr{C}(x_{\mathsf{new}})\ , \quad (2)$$

  and aim to quantify if possible the uncertainties of our predictions.
- Here $\mathscr{C}$ is called a classifier.
- Recall that the three paradigms are:
  - deterministic discriminative learning;
  - probabilistic discriminative learning;
  - probabilistic generative learning;
- We will use this three paradigms to learn particular classifiers.
- A first question we deal with is the existence of an optimal classifier, also called Bayes classifier.

Outline

- Consider $\mathsf{Y} = \{0, 1\}$ and $\mathsf{X} = \mathbb{R}^d$.
- What would be an optimal classifier $\mathscr{C} : \mathsf{X} \to \{0, 1\}$.
- To this end, we take a decision theoretical approach.
- Consider a probabilistic model for the data (X, Y) and denote by $(x, y) \mapsto p(x, y)$ the corresponding density.
- Namely,
$$\mathbb{P}(X \in \mathsf{A}, Y = i) = \int_{\mathsf{A}} p(x, i)\mathrm{d}x \;, \tag{3}$$
where p_0, p_1 are probability densities on $\mathbb{R}^d$.
- We define the risk of a classifier $\mathscr{C} : \mathsf{X} \to \{0, 1\}$ as
$$\mathrm{Risk}(\mathscr{C}) = \mathbb{E}\left[\mathbb{1}\left\{\mathscr{C}(X) \neq Y\right\}\right] \;. \tag{4}$$
- We can show that a classifier which minimizes the risk is given by $x \mapsto \mathscr{C}^\star(x)$ where $\mathscr{C}^\star(x) = \mathbb{1}\left\{q^\star(x) \geq 1/2\right\}$ with
$$q^\star(x) = \mathbb{E}\left[Y|X = x\right] = \mathbb{P}(Y = 1|X = x) = p(x, 1)/(p(x, 0) + p(x, 1)) \;. \tag{5}$$
- Proof: see in small classes.

- Take $\mathsf{Y} = \{0, 1\}$. Recall $\mathscr{C}^\star(x) = \mathbb{1}\{q^\star(x) \geq 1/2\}$.
- Here we aim to model somehow $q^\star(x) = \mathbb{P}(Y = 0|X = x)$ ($\mathbb{P}(Y = 1)$ is then completely determined...)
- To this end, we approximate $x \mapsto q^\star(x)$ by functions

$$h_w : x \mapsto \sigma \circ f_w(x) \quad \sigma : \mathbb{R} \to [0, 1] \, , \tag{6}$$

  - $f_w \in \mathcal{F}$ where $\mathcal{F}$ is a hypothesis class of functions from X to $\mathbb{R}$ parametrized by w.
  - For example,

$$\mathcal{F}_\phi = \left\{x \mapsto \phi(x)^{\mathrm{T}} w + w_0 \; : \; w \in \mathbb{R}^{d+1}\right\} \, , \quad \phi(x) = (\phi_1(x), \ldots, \phi_d(x))^{\mathrm{T}} \, . \tag{7}$$

- Take $\mathsf{Y} = \{0, 1\}$. Recall $\mathscr{C}^\star(x) = \mathbb{1}\left\{q^\star(x) \geq 1/2\right\}$.
- Here we aim to model somehow $q^\star(x) = \mathbb{P}(Y = 1|X = x)$ ($\mathbb{P}(Y = 0|X = x)$ is then completely determined...)
- To this end, we approximate $x \mapsto q^\star(x) - 1/2$ by functions

$$h_w : x \mapsto \sigma \circ f_w(x) \quad \sigma : \mathbb{R} \to [0, 1] \,, \tag{8}$$

with

  - $f_w \in \mathcal{F}$ where $\mathcal{F}$ is a hypothesis class of functions from X to $\mathbb{R}$ parametrized by w.
  - Here σ is called an activation function.
  - Ideally, in discriminative deterministic learning, we would like to choose $\sigma_{\mathbf{1}} : t \mapsto \mathbb{1}\left\{t - 1/2 \geq 0\right\}$ but non-smooth. Hence we use a regularization of this function (in general).
  - In many cases, σ will come from distribution function from probabilistic models that we will use.

- Take $\mathsf{Y} = \{0, 1\}$. Recall $\mathscr{C}^\star(x) = \mathbb{1}\{q^\star(x) \geq 1/2\}$.
- Here we aim to model somehow $q^\star(x) = \mathbb{P}(Y = 1|X = x)$ ($\mathbb{P}(Y = 0|X = x)$ is then completely determined...)
- To this end, we approximate $x \mapsto q^\star(x) - 1/2$ by functions

$$h_w : x \mapsto \sigma \circ f_w(x) \quad \sigma : \mathbb{R} \to [0, 1] \,, \tag{9}$$

  with $f_w \in \mathcal{F}$ where $\mathcal{F}$ is a hypothesis class of functions from X to $\mathbb{R}$ parametrized by w.
- This defines a hypothesis class of functions from X to $[0, 1]$,

$$\mathcal{H} = \{\sigma \circ f \,:\, f \in \mathcal{F}\} = \{\sigma \circ f_w \,:\, w \in \Theta\} \,. \tag{10}$$

Introduction to classification: hypothesis class

- Take $\mathsf{Y} = \{0, 1\}$. Recall $\mathscr{C}^\star(x) = \mathbb{1}\{q^\star(x) \geq 1/2\}$.
- Here we aim to model somehow $q^\star(x) = \mathbb{P}(Y = 0|X = x)$ ($\mathbb{P}(Y = 1)$ is then completely determined...)
- To this end, we approximate $x \mapsto q^\star(x)$ by functions

$$h_w : x \mapsto \sigma \circ f_w(x) \quad \sigma : \mathbb{R} \to [0, 1] \, , \tag{11}$$

  $f_w \in \mathcal{F}$ where $\mathcal{F}$ is a hypothesis class of functions from X to $\mathbb{R}$ parametrized by w.
- This defines a hypothesis class of functions from X to $[0, 1]$,

$$\mathcal{H} = \{\sigma \circ f \, : \, f \in \mathcal{F}\} = \{\sigma \circ f_w \, : \, w \in \Theta\} \, . \tag{12}$$

- Given a learned $h_{\hat{w}}$, the predictor/classifier is

$$\mathscr{C}_{\hat{w}} : x \mapsto \mathbb{1}\{h_{\hat{w}}(x) \geq \eta\} \, , \text{ where } \eta \text{ is a threshold, most of the times } 1/2 \text{ or } 0 \, . \tag{13}$$

- The set

$$\{x \in \mathsf{X} \, : \, h_{\hat{w}}(x) = \eta\} \, . \tag{14}$$

  is called the decision boundary associated with $h_{\hat{w}}$.

Introduction to classification: hypothesis class

- Take $\mathsf{Y} = \{0, 1\}$. If we change the risk with an asymmetric risk

$$\mathrm{Risk}_\eta(\mathscr{C}) = \eta\mathbb{P}(Y = 1, \mathscr{C}(X) = 0) + (1-\eta)\mathbb{P}(Y = 0, \mathscr{C}(X) = 1) \,, \quad (15)$$

  the optimal classifier is no more $\mathscr{C}^\star(x) = \mathbb{1}\left\{q^\star(x) \geq 1/2\right\}$ but $\mathscr{C}^\star(x) = \mathbb{1}\left\{q^\star(x) \geq \eta\right\}$.
- Here recall $q^\star(x) = \mathbb{P}(Y = 0|X = x)$.
- The extreme cases are easy... ✎
- Given a learned $h_{\hat{w}}$ approximating $q^\star$, the predictor/classifier is then in that case

$$\mathscr{C}_{\hat{w}} : x \mapsto \mathbb{1}\left\{h_{\hat{w}}(x) \geq \eta\right\} \,, \text{ where } \eta \text{ is a threshold, most of the times } 1/2 \text{ or } 0 \,. \quad (16)$$

- The set

$$\{x \in \mathsf{X} \,:\, h_{\hat{w}}(x) = \eta\} \,. \quad (17)$$

  is called the decision boundary associated with $h_{\hat{w}}$.

- Take $\mathsf{Y} = \{0, 1\}$.
- We consider first deterministic discriminative learning;
- and the activation function $\sigma(t) = \mathbb{1}\{t \geq 1/2\}$.
- In this simple case, the previous discussion simplifies and the classifiers have the form

$$\mathscr{C}_w : x \mapsto \mathbb{1}\{f_w(x) \geq 0\} \ , \quad f_w \in \mathcal{F} \ , \tag{18}$$

  with $\mathcal{F}$ hypothesis class of functions from $\mathsf{X} \to \mathbb{R}$.
- In the following, we consider only linear classifiers:

$$\mathscr{C}_w(x) = \mathbb{1}\{f_w(x) > 0\} \ , \quad f_w(x) = w^{\mathrm{T}}\Phi(x) \ , \quad \Phi : \mathsf{X} \to \mathbb{R}^d \ . \tag{19}$$

Some reminders

- Choose a class of hypotheses/representations $\mathcal{C} = \{\mathscr{C}_w : \mathsf{X} \to \mathsf{Y} \ : \ w \in \Theta\}$.
- Choose a loss function $\ell : \mathsf{Y} \times \mathsf{X} \times \Theta$.
- Define the error function

$$E_\ell(w) = \sum_{i=1}^{n} \ell(y_i, x_i, w) \ , \quad \text{equivalently} \quad E_\ell(w) = n^{-1} \sum_{i=1}^{n} \ell(y_i, x_i, w) \ . \tag{20}$$

- Choose an algorithm to solve

$$\text{minimize } E_\ell \ . \tag{21}$$

- Here, we start by specifying $\mathcal{C}$ and the form of ℓ.

- First, choose the class $\mathcal{C} = \{\mathscr{C}_w : \mathsf{X} \to \mathsf{Y} \ : \ w \in \mathbb{R}^d\}$.
- Consider $\mathsf{Y} = \{0, 1\}$ and $\mathsf{X} = \mathbb{R}^d$.
- The most simple predictors are linear predictor

$$\mathscr{C}_w : x \mapsto \mathbb{1}\{f_w(x) \geq 0\} \ , \quad f_w(x) = w^{\mathrm{T}} x + w_0 \ . \tag{22}$$

- Here the decision boundary is a the affine hyperplan:

$$\mathsf{H} = \{(w_0, w) \in \mathbb{R}^{d+1} \ : \ w^{\mathrm{T}} x + w_0 = 0\} \ . \tag{23}$$

- w is called a weight vector;
- w_0 a bias (not in statistical sense) or sometimes a negative threshold (why?).

- First, choose the class $\mathcal{C} = \{\mathscr{C}_w : \mathsf{X} \to \mathsf{Y} \ : \ w \in \mathbb{R}^d\}$.
- Consider $\mathsf{Y} = \{0, 1\}$ and $\mathsf{X} = \mathbb{R}^d$.
- The most simple predictors are linear predictors

$$\mathscr{C}_w : x \mapsto \mathbb{1}\{f_w(x) \geq 0\} \ , \quad f_w(x) = w^{\mathrm{T}} x + w_0 \ . \tag{24}$$

- Adding a component equal to 1 to x (dummy variable), we can simply consider

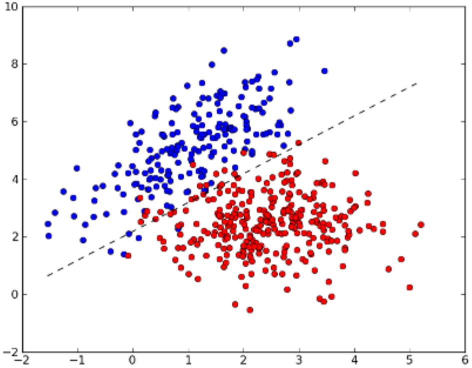
$$\mathscr{C}_w : x \mapsto \mathbb{1}\{f_w(x) \geq 0\} \ , \quad f_w(x) = w^{\mathrm{T}} x \ . \tag{25}$$

- In what follows, if not specify, I always consider that a dummy variable is added to the data...
- Define a loss ℓ and the error function

$$\boxed{E_\ell(w) = n^{-1} \textstyle\sum_{i=1}^n \ell(y_i, f_w(x_i)) \ .} \tag{26}$$

- Choose an algorithm to solve minimize E_ℓ.

Geometrically



- Learn a boundary to separate two "groups" of points.

- Here we consider $\mathsf{Y} = \{-1, 1\}$ which allow a more simple presentation of the perceptron algorithm.
- If data $\in \{0, 1\}$, apply the simple transformation

$$y \mapsto 2y - 1\;, \quad \text{with inverse } y \mapsto (y+1)/2\;. \tag{27}$$

- We consider here a slight generalization of the previous class function using basis functions $\{\phi_j : \mathsf{X} \to \mathbb{R}\}$.
- Denote by $\phi(x) = (\phi_1(x), \ldots, \phi_d(x))^{\mathrm{T}}$ and consider the class of linear discriminant functions associated with:

$$\mathcal{C} = \left\{ x \mapsto \mathscr{C}(x) = 2 \times \mathbb{1}\left\{ \phi(x)^{\mathrm{T}} w > 0 \right\} - 1 \;:\; w \in \mathbb{R}^d \right\} \tag{28}$$

- Now which loss and error functions to estimate the weight w?

- Here we consider $\mathsf{Y} = \{-1, 1\}$ which allow a more simple presentation of the perceptron algorithm.
- Denote by $\phi(x) = (\phi_1(x), \ldots, \phi_d(x))^{\mathrm{T}}$ and consider the class of linear discriminant functions associated with:

$$\mathcal{C} = \left\{ x \mapsto \mathscr{C}_w(x) = 2 \times \mathbb{1}\left\{ f_w(x) > 0 \right\} - 1 \,:\, w \in \mathbb{R}^d \right\} , \quad f_w(x) = \phi(x)^{\mathrm{T}} w \,. \tag{29}$$

- Here we consider the error function:

$$E_{\mathrm{perc}}(w) = -\sum_{i=1}^{N} y_i f_w(x_i) \mathbb{1}\left\{ y_i f_w(x_i) < 0 \right\} \,. \tag{30}$$

- Justification: we aim to fit the data:
  - if the data i well-classified, no contribution since $y_i f_w(x_i) \geq 0$;
  - if the data i not well-classified, linear contribution $-y_i \phi_x(x_i)^{\mathrm{T}} w \geq 0$.
- Exercise: what is the corresponding loss?

The perceptron algorithm: introduction to stochastic gradient descent

- We aim to minimize $E(w) = \sum_{i=1}^{N} E_i(w)$, $E_i = y_i f_w(x_i) \mathbb{1}\{y_i f_w(x_i) < 0\}$.
- $\nabla E_i(w) = ??$ ✎
- First option gradient descent (GD):

$$w_{k+1} = w_k - \eta \nabla E(w) \; . \tag{31}$$

- The parameter $\eta > 0$ is called the learning rate in ML or stepsize in optimization.
- Problem of GD: ∇E not always accessible
  - $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ not always entirely available;
  - data may arrive sequentially/ in a continuous stream;
  - yet you want to make some predictions and adjust them with new data...
  - N is too large...

- We aim to minimize $E(w) = \sum_{i=1}^{N} E_i(w)$, $E_i = y_i f_w(x_i) \mathbb{1}\{y_i f_w(x_i) < 0\}$.

Algorithm 1: Online SGD

- Initialize w_0
- Given a stream of data $\{(x_i, y_i)\}_{i=1}^{N}$, for $k = 0 \ldots$
  - $w_{k+1} = w_k - \eta_k \nabla E_k(w_k) = w_k - \eta_k y_k \phi(x_k) \mathbb{1}\{y_k \phi(x_k)^{\mathrm{T}} w_k < 0\}$

- $(\eta_k)_{k \in \mathbb{N}}$ is a sequence of stepsize which either $\to 0$ and is constant $\eta_k \equiv \eta$.
- The data is used online.
- Once the k-th pair (x_k, y_k) has been used, it can be forgotten.
- Number of iterations: number of data which can be collected.

The perceptron algorithm: introduction to stochastic gradient descent

- We aim to minimize $E(w) = \sum_{i=1}^{N} E_i(w)$, $E_i = y_i f_w(x_i) \mathbb{1}\left\{y_i f_w(x_i) < 0\right\}$.
- When N is too large

Algorithm 2: Batch SGD

- Initialize w_0, size of a batch b;
- Given a dataset $\{(x_i, y_i)\}_{i=1}^{N}$, for $k = 0...$
  - Take a random batch $\mathsf{B}_k \subset \{1, \dots, N\}$ with $\mathrm{Card}(\mathsf{B}_k) = b$;
  - $w_{k+1} = w_k - \eta_k \sum_{i \in \mathsf{B}_k} \nabla E_i(w_k) =$
    $w_k - \eta_k \sum_{i \in \mathsf{B}_k} y_i \phi(x_i) \mathbb{1}\left\{y_i \phi(x_i)^{\mathrm{T}} w_k < 0\right\}$

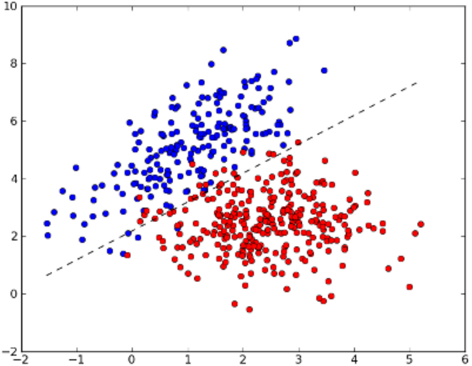
- The data is used by batches.
- Number of iterations: potentially infinite.
- Under appropriate condition $(w_k)_{k \in \mathbb{N}}$ converges to a minimizer of E.
- An alternative to $(w_k)_{k \in \mathbb{N}}$ is the corresponding Polyak-Ruppert averaging.

Cons of the perceptron algorithm

- Cons
  - Perceptron does not generalize well for $K > 2$;
  - does not provide probabilistic outputs;
- In the sequel, we present probabilistic models to circumvent the last issue.

Classification

Geometrically



- Learn a boundary to separate two “groups” of points.
- ? : what about Y with $K = \mathrm{Card}(Y) > 2$?

Case $K = \mathrm{Card}(\mathsf{Y}) > 2$

- Several strategies for $\mathsf{Y} = \{1, \dots, K\}$.
- One-versus-the rest: consider the $K-1$ classifiers $\mathscr{C}_k$ associated to the $\{0,1\}$-prediction problems for $k \in \{1, \dots, K-1\}$:

$$y = k\ , \quad y \neq k\ . \tag{32}$$

- The one-versus-the rest classifier:

$$\mathscr{C}^{\mathrm{OR}} = \sum_{k=1}^{K-1} (k+1)\mathscr{C}_k\ . \tag{33}$$

- This makes sense for $K = 2$?
- However, this leads to ambiguous classification, i.e., classifiers give non-consistent results
- Example: $\mathscr{C}_{K-1}(x) = \mathscr{C}_{K-2}(x) = 1$ or $\mathscr{C}^{\mathrm{OR}}(x) \notin \mathsf{Y} = \{1, \dots, K\}$.

Case $K = \mathrm{Card}(\mathsf{Y}) > 2$

- Several strategies for $\mathsf{Y} = \{1, \ldots, K\}$.
- One-versus-one: consider the $K(K-1)/2$ classifiers $\mathscr{C}_{k,i}$ associated to the $\{0,1\}$-prediction problems for $k, i \in \{1, \ldots, K\}$, $i \neq k$:

$$y = k\ , \quad y = i\ . \tag{34}$$

- The one-versus-the rest classifier:

$$\mathscr{C}^{\mathrm{OR}} = \sum_{\substack{k,i=1 \\ k \neq i}}^{K} (k+1)\mathscr{C}_{k,i}\ . \tag{35}$$

- However, this leads to ambiguous classification again.

- Several strategies for $\mathsf{Y} = \{1, \dots, K\}$.
- A consistant solution is to consider K discriminant functions $\{f_k : \mathsf{X} \to \mathbb{R}\}_{k=1}^{K}$ and set

$$\mathscr{C}^{K,\mathrm{lin}}(x) = \underset{k \in \mathsf{Y}}{\mathrm{argmax}}\, f_k(x)\ . \tag{36}$$

- It is in line with the formal classification problem.
- Defining a classifier is in fact equivalent to define a partition of $\mathsf{X} = \sqcup_{i=1}^{K} \mathsf{X}_i$.
- Indeed, if we have a classifier $\mathscr{C}$, we can define $\mathsf{X}_i = \{x\ \mathscr{C}(x) = i\} = \mathscr{C}^{-1}(\{i\})$.
- If we have a partition, we can define $\mathscr{C}(x) = \sum_{i=1}^{K} i \mathbb{1}_{\mathsf{X}_i}(x)$.

Case $K = \mathrm{Card}(\mathsf{Y}) > 2$

- Several strategies for $\mathsf{Y} = \{1, \ldots, K\}$.
- A consistant solution is to consider K discriminant functions $\{f_k : \mathsf{X} \to \mathbb{R}\}_{k=1}^{K}$ and set

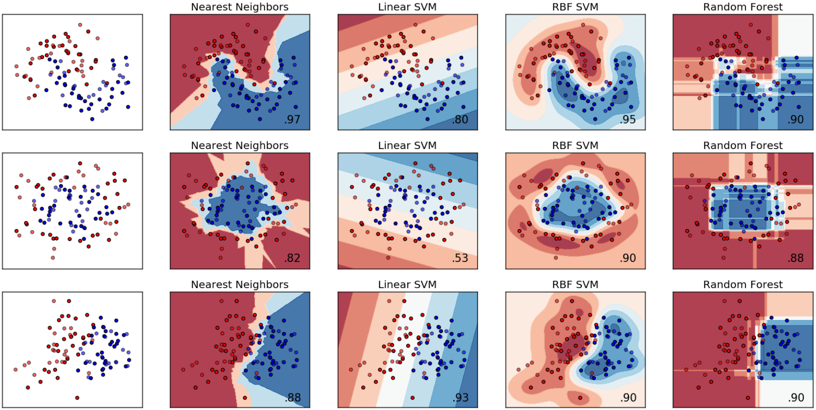
$$\mathscr{C}^{K,\mathrm{lin}}(x) = \min \operatorname*{argmax}_{k \in \mathsf{Y}} f_k(x) \; . \tag{37}$$

- It is in line with the formal classification problem.
- The min in the definition (39) is to make a choice in the case $\mathrm{argmax}_{k \in \mathsf{Y}} f_k(x)$ admits strictly more than one element.
- For $\mathsf{A} \subset \mathsf{Y}$, $\mathsf{A} = \{k_1, \ldots\}$,

$$\mathsf{D}_{\mathsf{A}} = \{x \; : \; \operatorname*{argmax}_{k} f_k(x) = \mathsf{A}\} \; , \tag{38}$$

is called the decision boundary associated with the labels in A.

...many ways to separate points!



Case $K = \mathrm{Card}(\mathsf{Y}) > 2$

- Several strategies for $\mathsf{Y} = \{1, \dots, K\}$.
- A consistant solution is to consider K discriminant functions $\{f_k : \mathsf{X} \to \mathbb{R}\}_{k=1}^{K}$ and set

$$\mathscr{C}^{K,\mathrm{lin}}(x) = \min \operatorname*{argmax}_{k \in \mathsf{Y}} f_k(x) \; . \tag{39}$$

- Example: for $\mathsf{X} = \mathbb{R}^d$, $f_k(x) = w_k^{\mathrm{T}} x$
- For the moment $\{w_k\}_{k=1}^{K}$ are arbitrary and define the hypothesis class of our classifiers.
- They have to be learned to fit the data; up coming!
- Then $\mathsf{D}_{\{1,2\}} = \{x \, : \, (w_1 - w_2)^{\mathrm{T}} x\}$.
- For any $\mathsf{A} \subset \mathsf{Y}$, D_{A} is in fact convex (polygon):

$$x_1, x_2 \in \mathsf{D}_{\mathsf{A}} \implies t x_1 + (1-t) x_2 \in \mathsf{D}_{\mathsf{A}} \text{ for any } t \in [0,1] \; . \tag{40}$$

- ? : how to find $\{f_k\}_{k=1}^{K}$ or equivalently here $\{w_k\}_{k=1}^{K}$?

Least square for classification

- We consider a 1 of K binary coding scheme

$$y = k \mapsto y = \boldsymbol{e}_k = \begin{pmatrix} 0 \\ \vdots \\ 1 \text{ position } k \\ \vdots \\ 0 \end{pmatrix} . \tag{41}$$

- And then K linear discriminant (bias included...):

$$f_{w^{(k)}}(x) = [w^{(k)}]^{\mathrm{T}} x . \tag{42}$$

- This gives the model

$$F_{\boldsymbol{W}}(x) = \boldsymbol{W} x , \quad \text{where } \boldsymbol{W} = [w^{(1)} \cdots w^{(K)}] \in \mathbb{R}^{d \times K} . \tag{43}$$

- We expect that if $y_i = k$, $(\boldsymbol{W} x_i)_k > (\boldsymbol{W} x_i)_{k'}$, $k' \neq k$.
- This motivates the introduction of the error function:

$$E(\boldsymbol{W}) = \frac{1}{2} \sum_{i=1}^{N} \left\| y_i - \boldsymbol{W}^{\mathrm{T}} w_i \right\|^2 . \tag{44}$$

- E can be written of the of the form

$$E(\boldsymbol{W}) = \frac{1}{2}\mathrm{trace}\left((\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{W})^{\mathrm{T}}(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{W})\right) , \tag{45}$$

with

$$\boldsymbol{Y} = [y_1 \cdots y_N]^{\mathrm{T}} \in \mathbb{R}^{N\times K} , \quad \boldsymbol{X} = [x_1 \cdots x_N]^{\mathrm{T}} \in \mathbb{R}^{N\times d} . \tag{46}$$

- E is quadratic ! and admits then a unique minimizer:

$$\hat{\boldsymbol{W}} = \boldsymbol{X}^{\dagger}\boldsymbol{Y} = (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y} , \tag{47}$$

if $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ is invertible.

Comments on LS for classification

- Drawbacks
  - lack of robustness to outliers;
  - does not corresponds to any data: Solution to E corresponds to the maximum likelihood estimator for Gaussian conditional likelihood.
- In the sequel, we explore more models to addresse these issues.

Fisher's linear discriminant

- We consider the two classes problem $\mathsf{Y} = \{0, 1\}$.
- Projection can lead to significant loss of information in the data.
- However, we can seek a vector w such that the projection onto $\mathrm{Span}(w)$ which maximizes the separation between the two classes.
- The simplest measure of separation is the separation between the projected means:

$$\text{maximize } w \mapsto w^{\mathrm{T}}\{m_1 - m_0\}\ , \text{ subject to } \|w\| = 1\ , \tag{48}$$

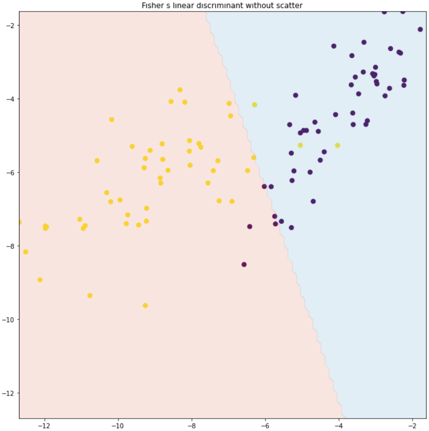
$$m_i = \sum_{j=1}^{N} x_j \mathbb{1}\{y_j = i\} \Big/ \sum_{j=1}^{N} \mathbb{1}\{y_j = i\}\ . \tag{49}$$

- Solution

$$\hat{w} = (m_1 - m_0) / \|m_1 - m_0\|\ . \tag{50}$$

- However this criteria is too sensible to correlated data: does not catch second order information.

Example: Fisher first attempt



- We consider the two classes problem $\mathsf{Y} = \{0, 1\}$.
- Projection can lead to significant loss of information in the data.
- However, we can seek a vector w such that the projection onto $\mathrm{Span}(w)$ which maximizes the separation between the two classes.
- To catch second order information, we aim to take into account the variance of the data once projected to w:

$$s^2 = w^{\mathrm{T}}\Sigma w \ , \quad \boldsymbol{S}_W = \sum_{j=1}^{N}(x_j - m_i)(x_j - m_i)^{\mathrm{T}}\mathbb{1}\{y_j = i\} \ . \tag{51}$$

- Remarks:
  - $N^{-1}\boldsymbol{S}_W$: empirical covariance for the $\{x_i\}_{i=1}^{N}$ within class.
  - It is implicitly assumed that the covariances of two classes are the same.
  - s is sometimes called scatter.

Fisher's linear discriminant

- We consider the two classes problem $\mathsf{Y} = \{0, 1\}$.
- Projection can lead to significant loss of information in the data.
- However, we can seek a vector w such that the projection onto $\mathrm{Span}(w)$ which maximizes the separation between the two classes.
- To catch second order information, we aim to take into account the variance of the data once projected to w:

$$s^2 = w^{\mathrm{T}} \Sigma w \ , \quad \boldsymbol{A} = \sum_{j=1}^{N} (x_j - m_i)(x_j - m_i)^{\mathrm{T}} \mathbb{1}\left\{y_j = i\right\} \ . \qquad (52)$$

- Based on these quantities, we can seek the projection which maximizes the separation between the projected means and with smallest scatter:

$$J(w) = (w^{\mathrm{T}}(m_1 - m_0))^2 / s^2 = \frac{w^{\mathrm{T}} \boldsymbol{S}_B w}{w^{\mathrm{T}} S_W w} \ , \qquad (53)$$

with $N^{-1} \boldsymbol{S}_B = (m_1 - m_0)(m_1 - m_0)^{\mathrm{T}} / N$ between-class covariance matrix.

- Based on these quantities, we can seek the projection which maximizes the separation between the projected means and with smallest scatter:

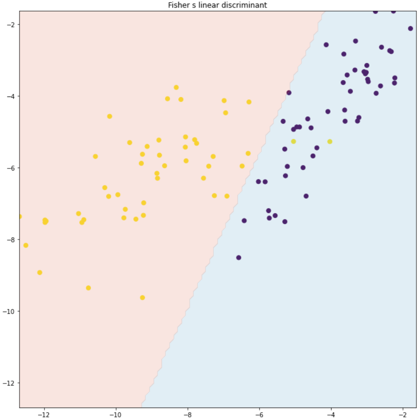
$$J(w) = (w^{\mathrm{T}}(m_1 - m_0))^2/s^2 = \frac{w^{\mathrm{T}}\boldsymbol{S}_B w}{w^{\mathrm{T}}S_W w} \;, \tag{54}$$

  with $N^{-1}\boldsymbol{S}_B = (m_1 - m_0)(m_1 - m_0)^{\mathrm{T}}/N$ between-class covariance matrix.
- $\nabla J(w) = 0$ implies that there exist $a_1, a_2, a_3 \in \mathbb{R}$ such that ✎

$$\text{Conclusion: } \hat{w} \propto \boldsymbol{S}_W^{-1}(m_1 - m_0) \;. \tag{55}$$

- Maximizer by Cauchy-Schwarz inequality!

Example: Fisher DA



- We consider now a probabilistic model for the data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$.
- We always assume if not specified that the pairs are i.i.d observations of a pair of random variable (X, Y)!
- In the generative probabilistic model setting:
  - we choose a prior on Y: $\mathbb{P}(Y = k) = q_k$;
  - a parametric model for $\{\mathbb{P}_w(X \in \cdot | Y = k) \; : \; w \in \mathbb{R}^d\}$;
  - we assume that $\mathbb{P}_w(X \in \cdot | Y = k)$ has a density $p_w(\cdot|k)$.
- This gives rise to the joint model and the likelihood

$$\mathrm{L}(w; \mathcal{D}) = \prod_{i=1}^{N} \{p_w(x_i|y_i) q_{y_i}\} \; . \tag{56}$$

- In what follows, w is then estimated by maximum likelihood estimation:

$$\hat{w} \in \operatorname*{argmax}_{w} \mathrm{L}(w; \mathcal{D}) \; . \tag{57}$$

- The prediction is then:

$$y_{\mathsf{pred}} = \operatorname*{argmax}_{k \in \mathsf{Y}} \mathbb{P}_{\hat{w}}(Y = k | X = x_{\mathsf{new}}) = \operatorname*{argmax}_{k \in \mathsf{Y}} [q_k p_{\hat{w}}(x_{\mathsf{new}}|k)] \; . \tag{58}$$

- We consider now a probabilistic model for the data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$.
- We always assume if not specified that the pairs are i.i.d observations of a pair of random variable (X, Y)!
- In the generative probabilistic model setting:
  - we estimate $\mathbb{P}(Y = k) = q_k$;
  - choose a parametric model for $\{\mathbb{P}_w(X \in \cdot | Y = k) : w \in \mathbb{R}^d\}$;
  - we assume that $\mathbb{P}_w(X \in \cdot | Y = k)$ has a density $p_w(\cdot | k)$.
- Estimation of $\mathbb{P}(Y = k) = q_k$? ✎

- We consider now a probabilistic model for the data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$.
- We always assume if not specified that the pairs are i.i.d observations of a pair of random variable (X, Y)!
- In the generative probabilistic model setting:
  - we choose a prior on Y: $\mathbb{P}(Y = k) = q_k$ or we can use the MLE...;
  - a parametric model for $\{\mathbb{P}_w(X \in \cdot | Y = k) \,:\, w \in \mathbb{R}^d\}$;
  - we assume that $\mathbb{P}_w(X \in \cdot | Y = k)$ has a density $p_w(\cdot | k)$.
- It remains to specify the model $\{\mathbb{P}_w(X \in \cdot | Y = k) \,:\, w \in \mathbb{R}^d\}$.

Outline

- The Naive Bayes assumption assume that $X = (X^{(1)}, \ldots, X^{(d)})$ have independent component given Y:

$$\mathbb{P}_w\left(X|Y\right) = \prod_{i=1}^{d} \mathbb{P}_w\left(X^{(i)}\middle|Y\right) .$$

- Crude modeling for $\mathbb{P}_w\left(X|Y\right)$;
- Feature independence assumption;
- Simple featurewise model: binomial if binary, multinomial if finite and Gaussian if continuous.
- If all features are continuous, the law of X given Y is Gaussian with a **diagonal covariance matrix**!

- The Naive Bayes assumption assume that $X = (X^{(1)}, \dots, X^{(d)})$ have independent component given Y:

$$\mathbb{P}_w(X|Y) = \prod_{i=1}^{d} \mathbb{P}_w\left(X^{(i)}\middle|Y\right) .$$

- For $k \in \mathrm{Y} = \{1, \dots, K\}$, the conditional density of $X^{(i)}$ given $\{Y = k\}$ is

$$\boxed{p_w(x^{(i)}|k) = (2\pi\sigma_{i,k}^2)^{-1/2}\exp\left\{-(x^{(i)} - \mu_{i,k})^2/(2\sigma_{i,k}^2)\right\} .}$$

- The complete conditional distribution of X given $\{Y = k\}$ is then

$$\boxed{p_w(x|y = k) = (\det(2\pi\Sigma_k))^{-1/2}\exp\left\{-(x - \mu_k)^T\Sigma_k^{-1}(x - \mu_k)/2\right\} ,}$$

- The parameter w consists in $w = \{(\mu_k, \Sigma_k)\}_{k=1}^{K}$ with

$$\Sigma_k = \mathrm{diag}(\sigma_{1,k}^2, \dots, \sigma_{d,k}^2) \text{ and } \mu_k = (\mu_{1,k}, \dots, \mu_{d,k})^T . \tag{59}$$

- MLE? ✎

Gaussian Naive Bayes

- The Naive Bayes assumption assume that $X = (X^{(1)}, \dots, X^{(d)})$ have independent component given Y:

$$\mathbb{P}_w(X|Y) = \prod_{i=1}^{d} \mathbb{P}_w\left(X^{(i)}\middle|Y\right) .$$

- For $k \in \mathrm{Y} = \{1, \dots, K\}$, the conditional density of $X^{(i)}$ given $\{Y = k\}$ is

$$\boxed{p_w(x^{(i)}|k) = (2\pi\sigma_{i,k}^2)^{-1/2}\exp\left\{-(x^{(i)} - \mu_{i,k})^2/(2\sigma_{i,k}^2)\right\} .}$$

- The complete conditional distribution of X given $\{Y = k\}$ is then

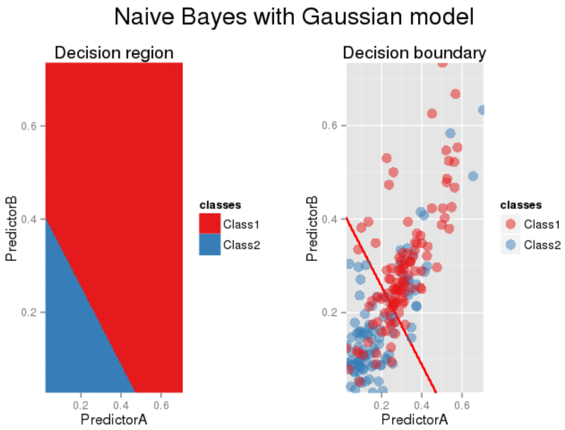
$$\boxed{p_w(x|y = k) = (\det(2\pi\Sigma_k))^{-1/2}\exp\left\{-(x - \mu_k)^T\Sigma_k^{-1}(x - \mu_k)/2\right\} ,}$$

- The parameter w consists in $w = \{(\mu_k, \Sigma_k)\}_{k=1}^{K}$ with

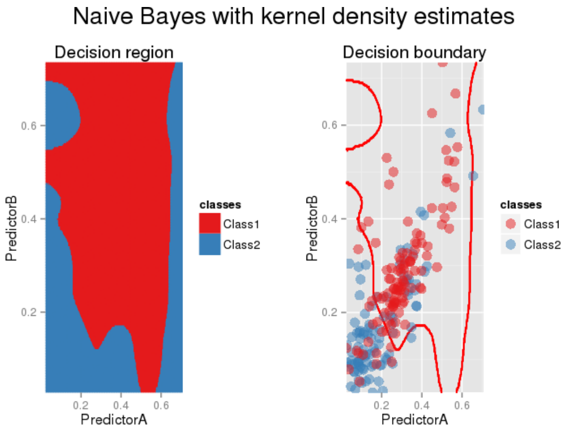
$$\Sigma_k = \mathrm{diag}(\sigma_{1,k}^2, \dots, \sigma_{d,k}^2) \text{ and } \mu_k = (\mu_{1,k}, \dots, \mu_{d,k})^T . \tag{60}$$

- MLE? ✎

Gaussian Naive Bayes



Kernel density estimate based Naive Bayes



- In this experiment, $\mathbb{P}(X^{(i)} \in \cdot | Y = k)$ or more exactly the densities $p(\cdot)$ for $k \in \mathrm{Y}$ are estimated by kernel density estimation and used to obtain a classifier:

$$y_{\text{pred}} = \underset{k \in \mathrm{Y}}{\operatorname{argmax}} [q_k \hat{p}(x_{\text{new}} | k)] \ . \tag{61}$$

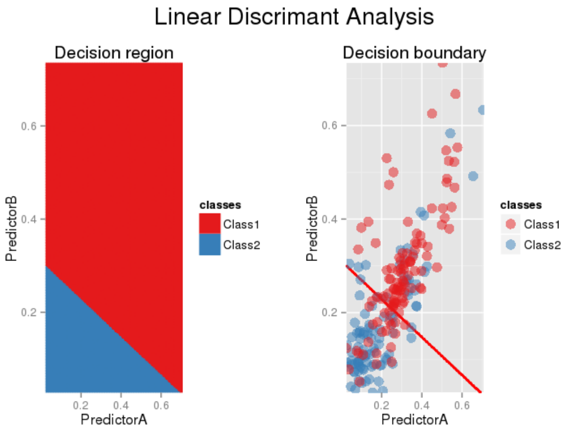
Discriminant Analysis (Gaussian model)

- Other models for $\{\mathbb{P}_w(X \in \cdot | Y = k) \, : \, w \in \mathbb{R}^d\}$ in the case $\mathsf{X} = \mathbb{R}^d$.
- For any $k \in \mathsf{Y}$, linear discriminant analysis (QDA):

$$\mathbb{P}_w(X \in \cdot | Y = k) = \mathsf{N}(\mu_k, \Sigma) \, . \tag{62}$$

- Here the parameter $w = (\{\mu_k\}_{k=1}^K, \Sigma)$.
- MLE? ✎
- $K = 2$, then the **decision boundary is an affine hyperplane**.

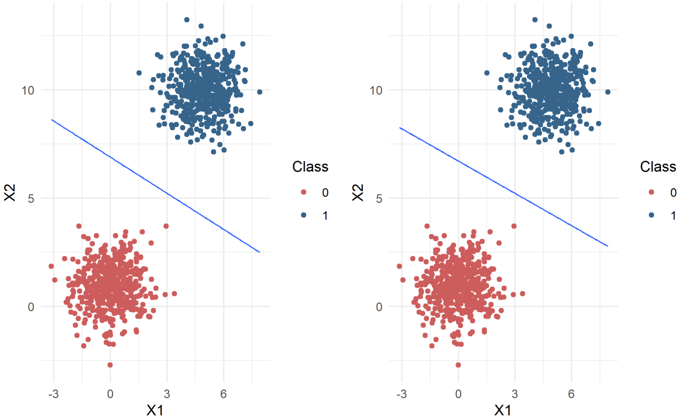
Example: LDA



Linear Discriminant Analysis
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

Example: LDA

```
boundary_true_parameters = function(x, mu0, mu1, Sigma, pi0){
  u = t(as.matrix(mu1-mu0)) %*% inv(Sigma)
  v = (u %*% (matrix(x - ((mu1+mu0)/2)) )) - log(pi0/(1-pi0))
  return(as.numeric(v))
}
```



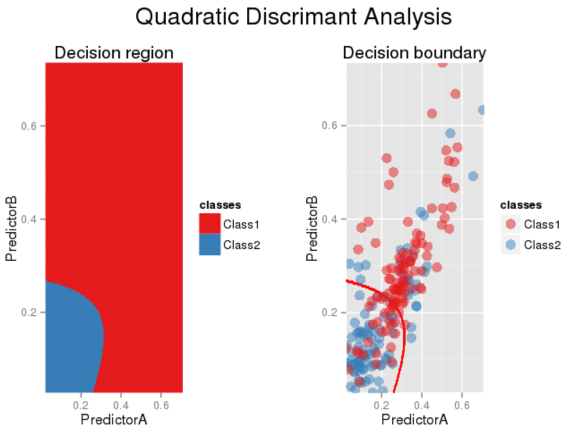
Discriminant Analysis (Gaussian model)

- Other models for $\{\mathbb{P}_w(X \in \cdot | Y = k) \,:\, w \in \mathbb{R}^d\}$ in the case $\mathsf{X} = \mathbb{R}^d$.
- For any $k \in \mathsf{Y}$, quadratic discriminant analysis (QDA):

$$\mathbb{P}_w(X \in \cdot | Y = k) = \mathsf{N}(\mu_k, \Sigma_k)\ . \tag{63}$$

- Here the parameter $w = \{(\mu_k, \Sigma_k)\}_{k=1}^{K}$
- MLE? ✎

Example: QDA

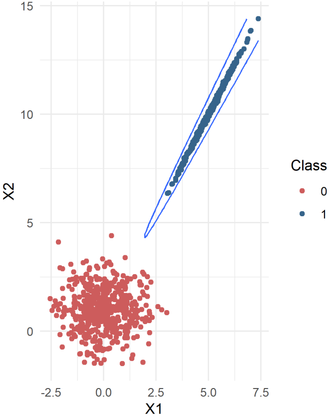

Quadratic Discriminant Analysis
Decision region
Decision boundary
PredictorB
PredictorA
0.2
0.4
0.6
classes
Class1
Class2

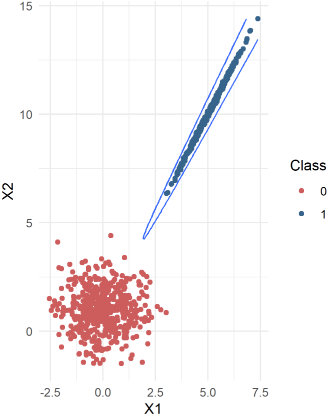
Example: QDA

```
boundary_true_parameters_quadratic = function(x, mu0, mu1, Sigma0, Sigma1, pi0){

  u1 =  -0.5*(t(as.matrix(x-mu1))) %*% inv(Sigma1) %*% as.matrix((x - mu1))
  u0 =  0.5*(t(as.matrix(x-mu0))) %*% inv(Sigma0) %*% as.matrix((x - mu0))
  cste = - log(pi0/(1-pi0))
  bonus = -0.5*log(abs(det(Sigma1))) + 0.5*log(abs(det(Sigma0)))

  return(as.numeric(u1+u0+cste+bonus))
}
```





Further reading

- Relation between Fisher's DA and least squares [Bis07, Section 4.1.5].
- Fisher's discriminant for multiple classes [Bis07, Section 4.1.6].
- Probabilistic generative models, discrete features/Exponential family [Bis07, Section 4.2.3, 4.2.4].

Bibliography i

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. 1st ed. Springer, 2007. ISBN: 0387310738.