STAT 535 10/5/23

Lecture 3

HW1 TB posted

Tutorial 10/10 at 2 pm

Decision boundaries

K-NN predictors

hyperplanes, linear, quadratic classifiers

Lecture Notes I – Examples of Predictors

Marina Meilă
mmp@stat.washington.edu

Department of Statistics
University of Washington

September 25, 2023

Prediction problems by the type of output ✓

The "learning" paradigm and vocabulary ✓

Some concepts in Classification - real valued output f ✓
- decision regions ←

The Nearest-Neighbor and kernel predictors ←

Linear predictors
- Least squares regression
- Linear Discriminant Analysis (LDA)
- QDA (Quadratic Discriminant Analysis)
- Logistic Regression
- The PERCEPTRON algorithm

Classification and regression tree(s) (CART)

The Naive Bayes classifier

Reading HTF Ch.: 2.3.1 Linear regression, 2.3.2 Nearest neighbor, 4.1–4 Linear classification, 6.1–3. Kernel regression, 6.6.2 kernel classifiers, 6.6.3 Naive Bayes, 9.2 CART, 11.3 Neural networks, Murphy Ch.: 1.4.2 nearest neighbors, 1.4.4 linear regression, 1.4.5 logistic regression, 3.5 and 10.2.1 Naive Bayes,4.2.1–3 linear and quadratic discriminant, 14.7.3– kernel regression, locally weighted regression, 16.2.1–4 CART, (16.5 neural nets), Bach Ch.:

Classifiers with real-valued output

Binary classification

- Since $y \in \{\pm 1\}$, naturally $f : \mathsf{X} \to \{\pm 1\}$
- But sometimes we prefer a classifier $f : \mathsf{X} \to \mathbb{R}$ (from a predictor class $\mathcal{F}$ of real-valued functions)
- In this case, the prediction $\hat{y}$ is usually

$$\hat{y} \;=\; \operatorname{sgn}(f(x)) \tag{1}$$

This is sometimes known as the sign trick.

Examples of real-valued classifiers

- Logistic Regression
- Naive Bayes
  in both of the above, $f(x) = P[Y = 1|X = x] \in [0, 1]$. Hence

$$\hat{y} \;=\; \operatorname{sgn}\left(f(x) - \frac{1}{2}\right) \tag{2}$$

- Support Vector Machines
- Kernel classifiers
- Neural Networks

Sign trick

The *sign* function $\operatorname{sgn}(y) = y/\lvert y \rvert$ if $y \neq 0$ and 0 iff $y = 0$ turns a real valued variable Y into a discrete-valued one.

Why real valued f?

- for statistical models $f(x) = P[Y = 1 \mid X = x]$ Example: Logistic regression
- for non-statistical models, $|f(x)|$ measures **confidence** in prediction $\hat{y}$, with $|f(x)| \approx 0$ meaning low confidence. Example: SVM
- if f is differentiable[1], the gradient ∇f is used in **learning algorithms** Examples: Logistic Regression, neural networks, some forms of linear regression such as Lasso

The margin (assuming $y \in \{\pm 1\}$)

- The **margin** of a classifier f at point $x \in \mathrm{X}$ is defined as

$$z = yf(x). \tag{3}$$

- Note that $\mathrm{sgn}(z) = y\hat{y}$.

- If $z > 0$, $\hat{y} = y$ and $f(x)$ is correct
- If $z \gg 0$, then $f(x)$ is correct, and classifier has high confidence
- If $z < 0$, then $f(x)$ is incorrect, and $|z|$ measures "how wrong" is f on this x
- Note also that $z \approx 0$ means that the classification $\hat{y}$ is not robust, whether correct or not

[1] and ∇f not 0 almost everywhere

Real valued multi-way classifiers

- We train m classifier $f_{1:m} : \mathsf{X} \rightarrow \mathbb{R}$. Then (typically)

$$\hat{y} \;=\; \operatorname*{argmax}_{c=1:m} f_{1:m}(x). \tag{4}$$

- $\hat{y} = y$ means the classifier is correct
- the training can be done
  - independently for each f_c, $c = 1 : m$ (e.g. generative classifiers – in Lecture II)
  - or at the same time (e.g. neural networks, SVM)

- The **margin** is defined as

$$z(x) \;=\; f_y - \max_{c \neq y} f_c(x) \tag{5}$$

In other words

- if $\hat{y} = y$ (correct), then $z = f_{\mathrm{true}} - f_{\mathrm{nextbest}} > 0$
- if $\hat{y} \neq y$ (mistake), then $z = f_{\mathrm{true}} - f_{\hat{y}} < 0$ (since $f_{\hat{y}}(x)$ is the max of $f_c(x)$)

Decision regions, decision boundary of a classifier

Let $f(x)$ be a classifier (not necessarily binary)

- $\hat{y}(x)$ takes a finite set of values
- The **decision region** associated with class y = the region in X space where f takes value y, i.e. $D_y = \{x \in \mathbb{R}^d,\ f(x) = y\} = f^{-1}(y)$.
- The boundaries separating the decision regions are called **decision boundaries**.

For $y \in \{1, .. m\}$ or $y \in \{\pm 1\}$

$$D_y = f^{-1}(y) = \{x \in \mathbb{R}^d \mid f(x) = y\}$$

$$f^{-1}(A) = \{x \mid f(x) \in A\}$$

for any $f : \mathbb{R}^d \to \mathbb{R}$

inverse image of A

Decision regions, decision boundary of a classifier

Let $f(x)$ be a classifier (not necessarily binary)

- $\hat{y}(x)$ takes a finite set of values
- The **decision region** associated with class y = the region in X space where f takes value y, i.e. $D_y = \{x \in \mathbb{R}^d,\ f(x) = y\} = f^{-1}(y)$.
- The boundaries separating the decision regions are called **decision boundaries**.

$y \in \{\pm 1\}$ Binary

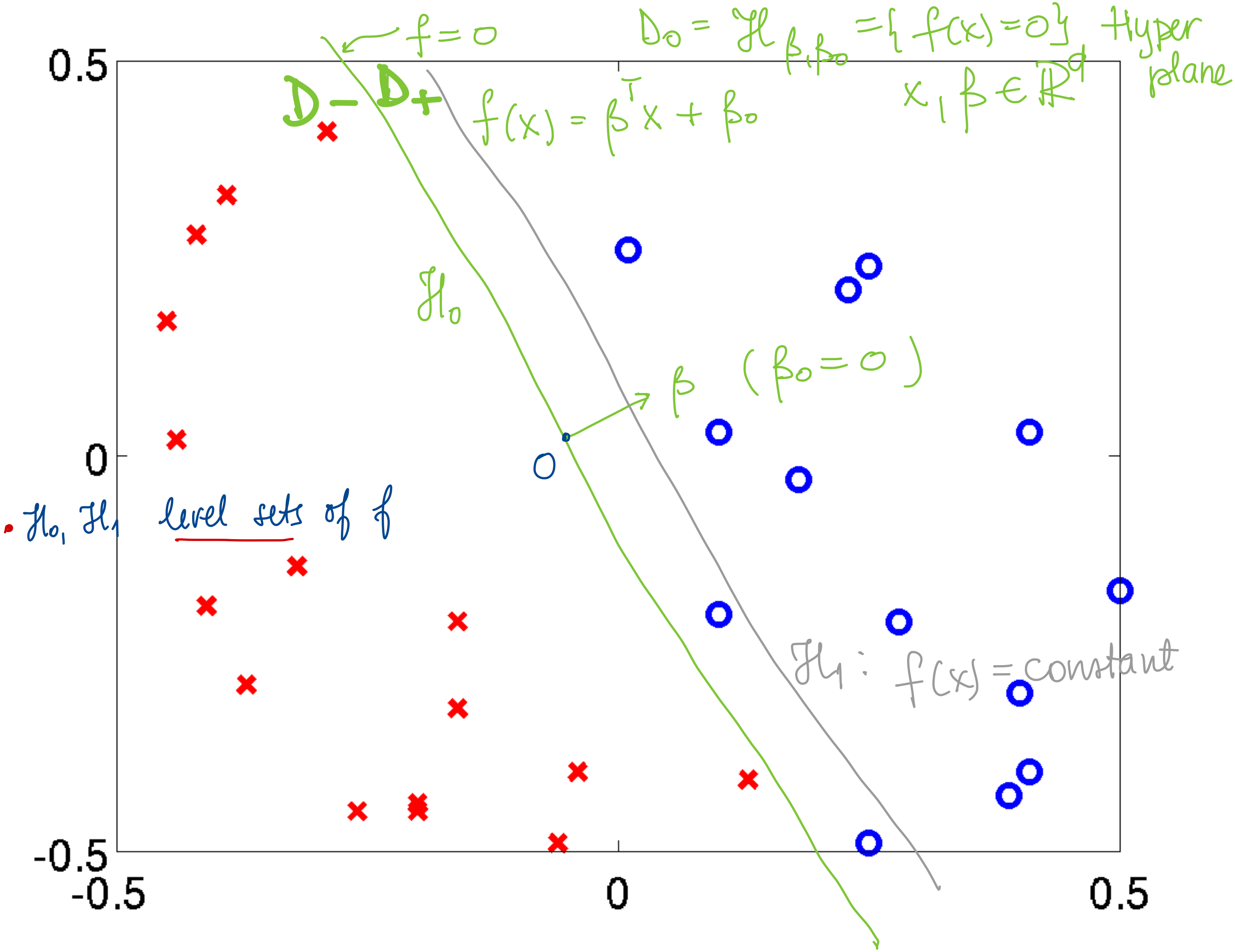
$D_+ = \{x \mid f(x) > 0\}$

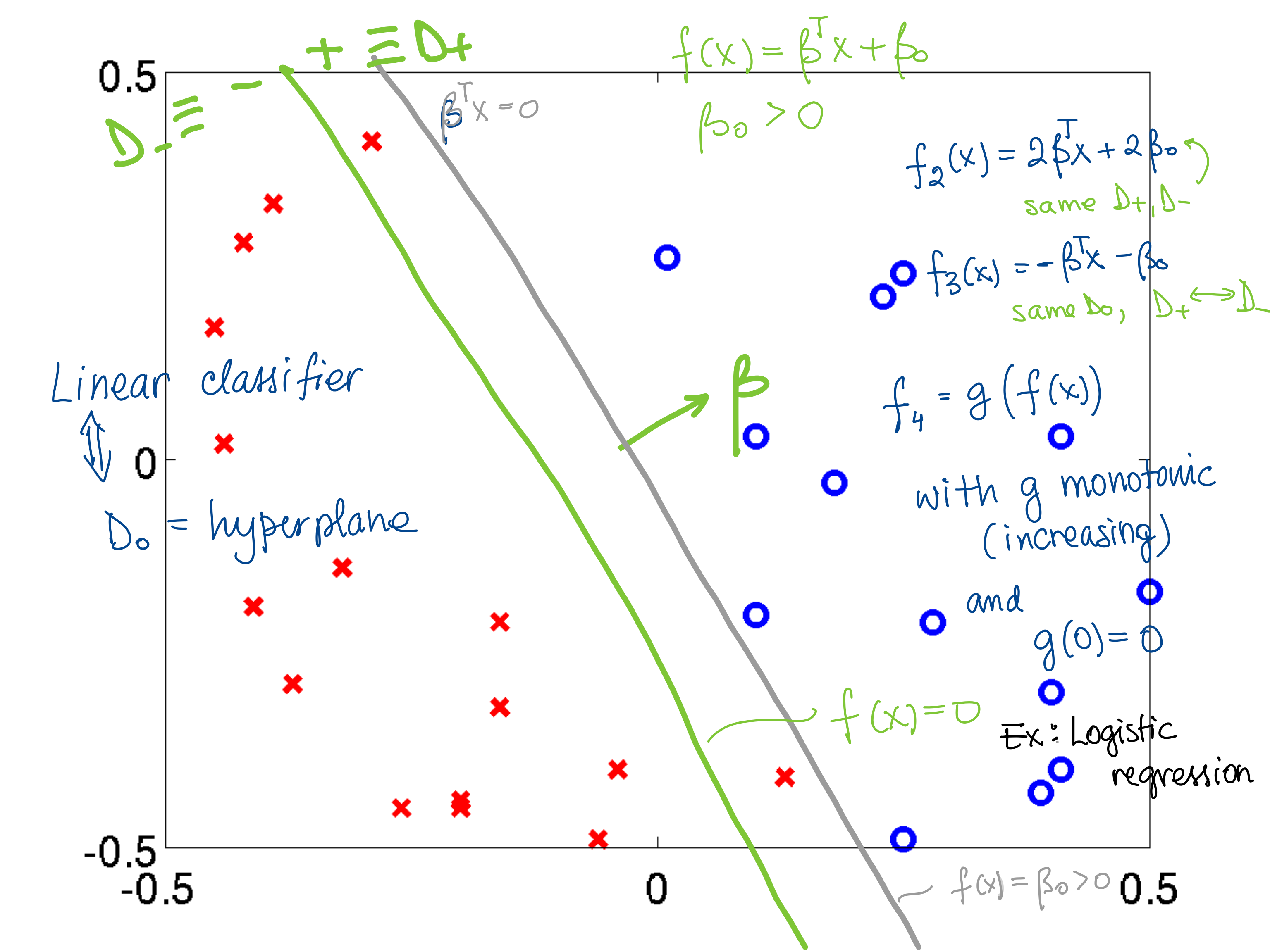
$D_- = \{x \mid f(x) < 0\}$

decision regions

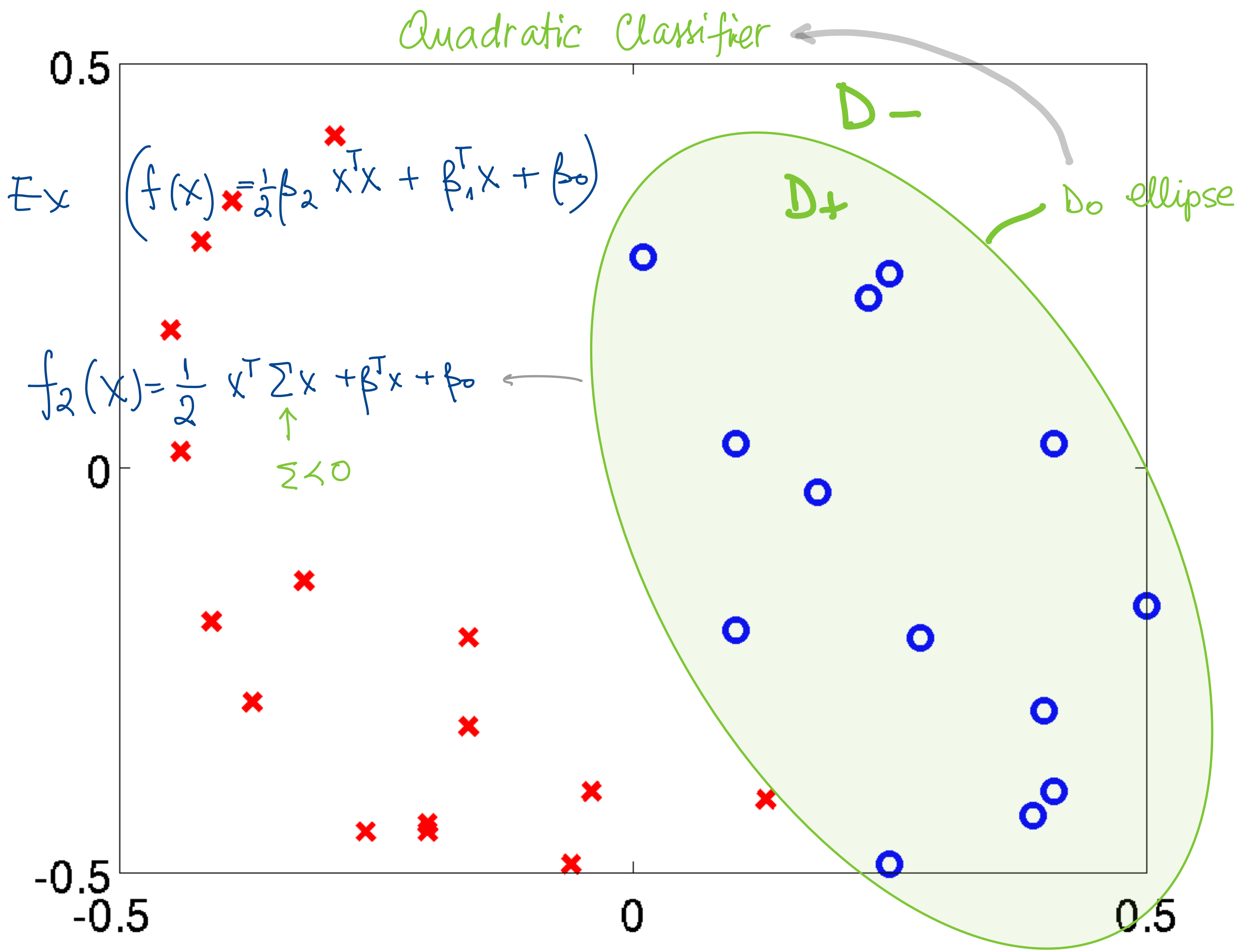
$D_0 = \{x \mid f(x) = 0\}$ decision boundary

↳ boundary between D_+, D_- if f continuous

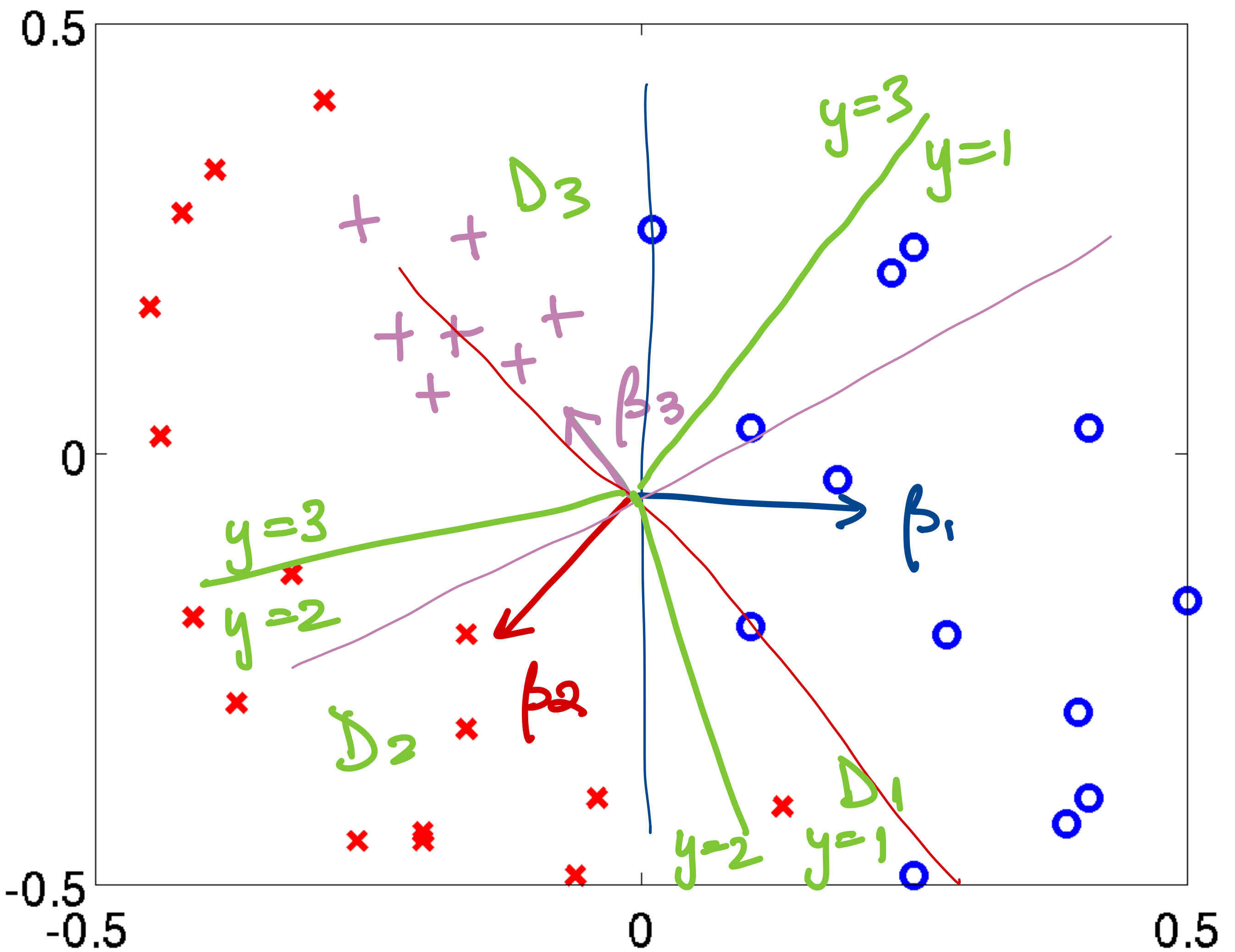

f=0
D_0 = H_{β,β_0} = {f(x)=0} Hyper plane
x, β ∈ R^d
D− D+
f(x) = β^T x + β_0
H_0
β (β_0 = 0)
O
• H_0, H_1 level sets of f
H_1 : f(x) = constant
0.5
0
-0.5
-0.5
0
0.5



0.5
0
-0.5
-0.5
0
0.5
D₋ ≡ − + ≡ D+
f(x) = βᵀx + β₀
β₀ > 0
βᵀx = 0
f₂(x) = 2βᵀx + 2β₀
same D+, D-
f₃(x) = -βᵀx - β₀
same D₀, D+ ⟷ D-
β
Linear classifier
D₀ = hyperplane
f₄ = g(f(x))
with g monotonic
(increasing)
and
g(0) = 0
f(x) = 0
Ex: Logistic regression
f(x) = β₀ > 0



Quadratic Classifier
0.5
0
-0.5
-0.5
0
0.5
D−
D+
D₀ ellipse
Ex $\left(f(x) = \frac{1}{2}\beta_2 x^T x + \beta_1^T x + \beta_0\right)$
$f_2(x) = \frac{1}{2} x^T \Sigma x + \beta^T x + \beta_0$
$\Sigma < 0$

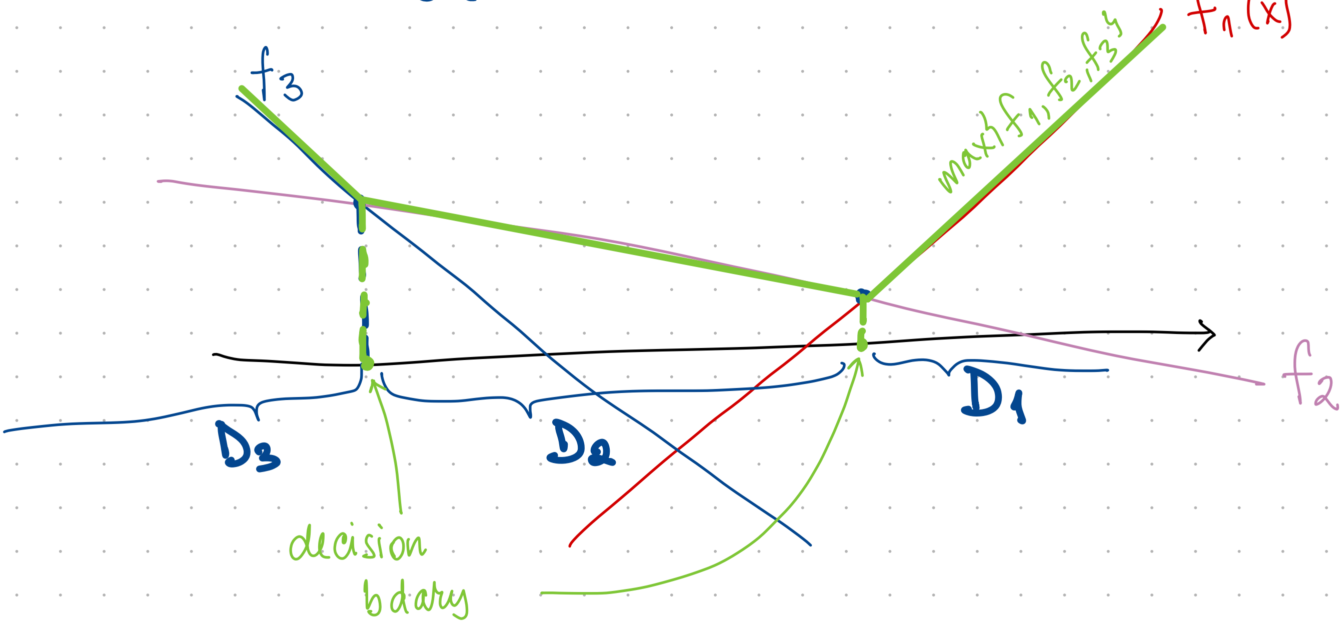

0.5
0
-0.5
-0.5
0
0.5
y=3
y=1
D3
β3
β1
y=3
y=2
β2
D2
D1
y=2
y=1

Linear multiway classifier

Exe: $D_0 \subseteq$ union of hyperplanes

f_c , $c \in \{1, \dots m\}$ $\quad f_c = \beta_c^T x + \beta_{0,c}$

$$\hat{y}(x) = \underset{c=1:m}{\operatorname{argmax}} f_c(x)$$



Decision regions, decision boundary of a classifier

Let $f(x)$ be a classifier (not necessarily binary)

- $\hat{y}(x)$ takes a finite set of values
- The **decision region** associated with class y = the region in X space where f takes value y, i.e. $D_y = \{x \in \mathbb{R}^d,\ f(x) = y\} = f^{-1}(y)$.
- The boundaries separating the decision regions are called **decision boundaries**.

- For a binary classifier, we have two decision regions, D_+ and D_-. By convention $f(x) = 0$ on the decision boundary.
- For binary classifier with real valued $f(x)$ (i.e $\hat{y} = \mathrm{sgn} f(x)$) we define $D_+ = \{x \in \mathbb{R}^d,\ f(x) > 0\}$, $D_- = \{x \in \mathbb{R}^d,\ f(x) < 0\}$ and the decision boundary $\{x \in \mathbb{R}^d,\ f(x) = 0\}$

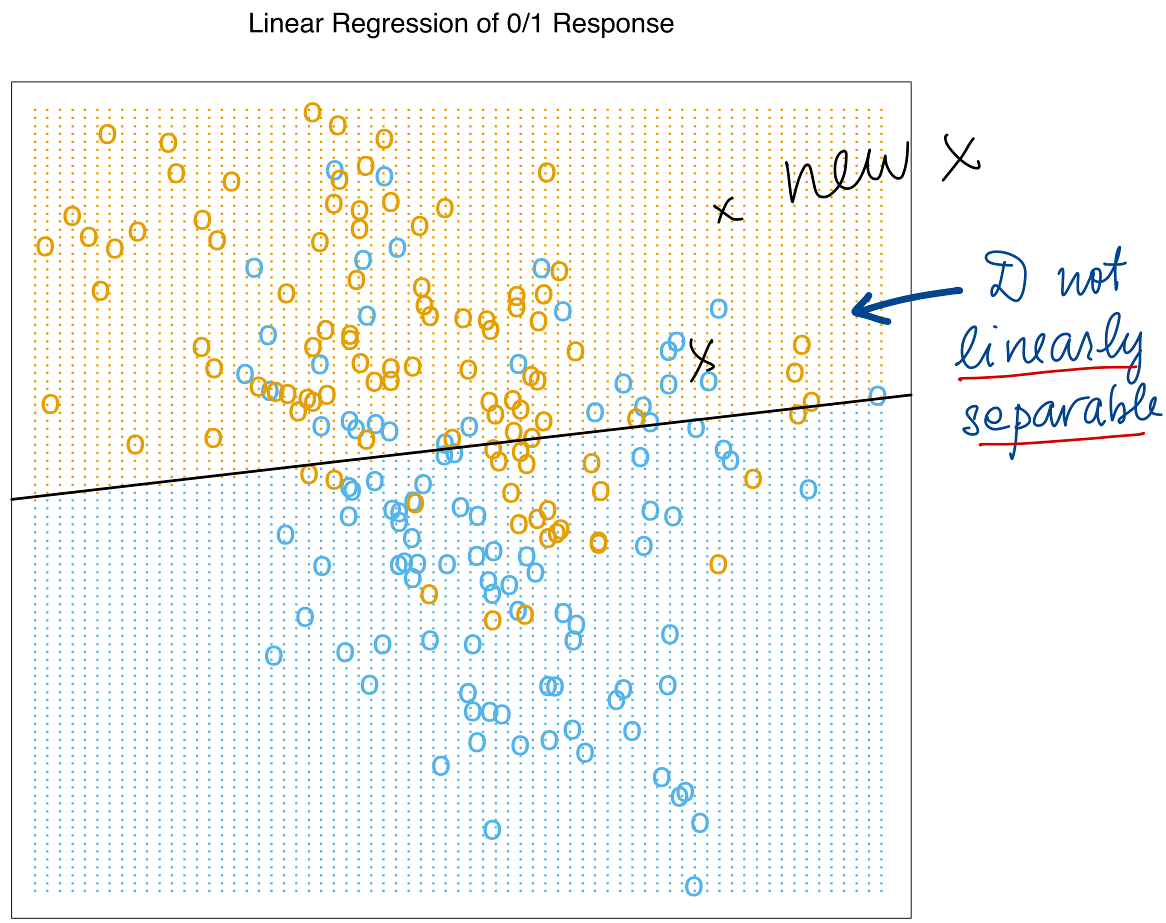


FIGURE 2.1. *A classification example in two dimensions. The classes are coded as a binary variable (*BLUE *= 0,* ORANGE *= 1), and then fit by linear regression. The line is the decision boundary defined by* $x^T\hat{\beta} = 0.5$. *The orange shaded region denotes that part of input space classified as* ORANGE, *while the blue region is classified as* BLUE.

linearly separable $\iff \exists\, f(x) = \beta^T x + \beta_0$ so that $\hat{y}(x^i) = y^i$ for all $i = 1:n$

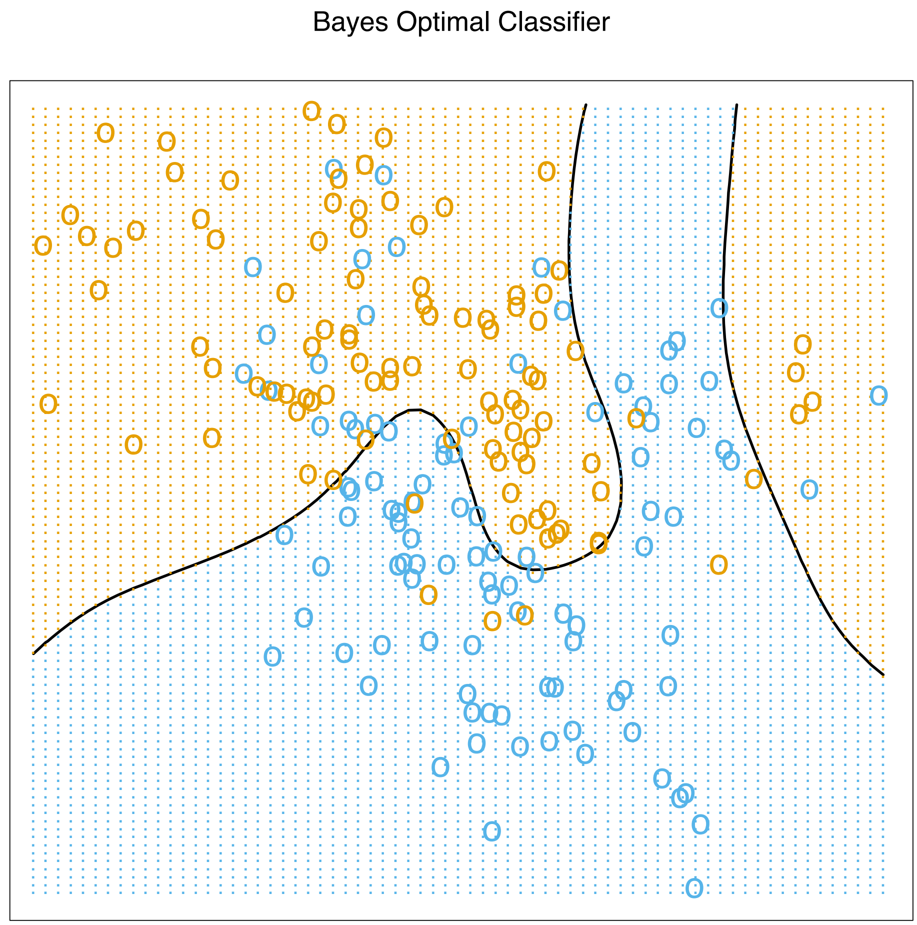


FIGURE 2.5. *The optimal Bayes decision boundary for the simulation example of Figures 2.1, 2.2 and 2.3. Since the generating density is known for each class, this boundary can be calculated exactly (Exercise 2.2).*

15 - NN

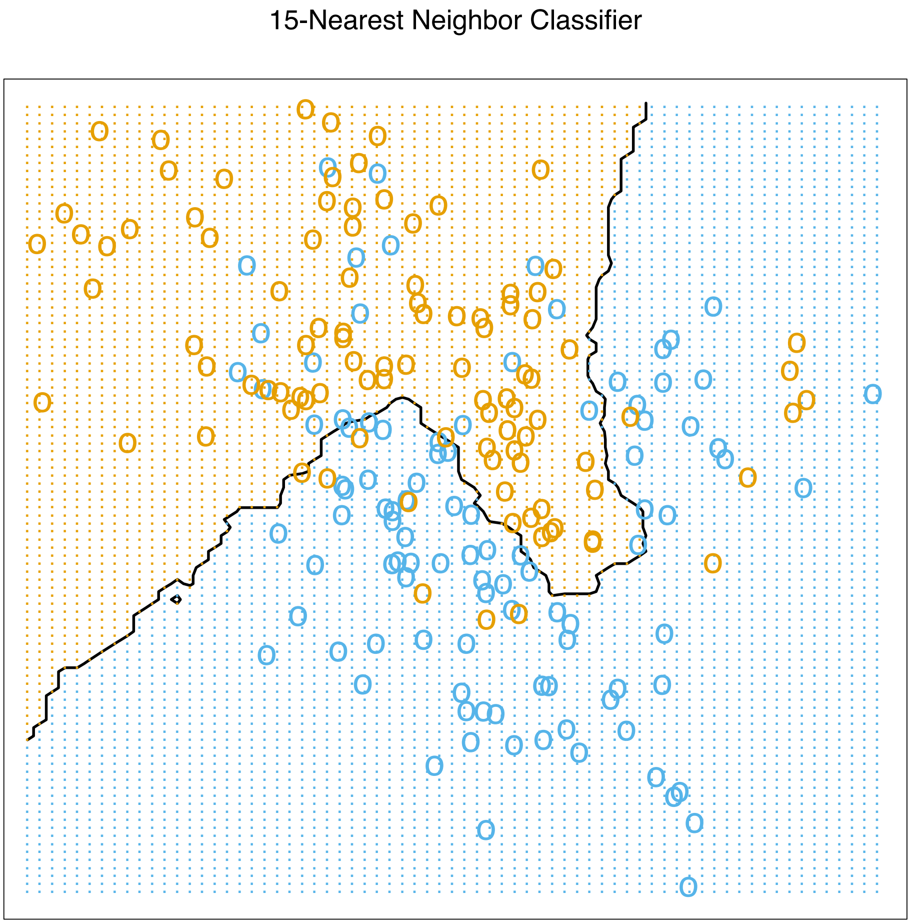


FIGURE 2.2. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$) *and then fit by* 15*-nearest-neighbor averaging as in (2.8). The predicted class is hence chosen by majority vote amongst the* 15*-nearest neighbors.*

The Nearest-Neighbor predictor

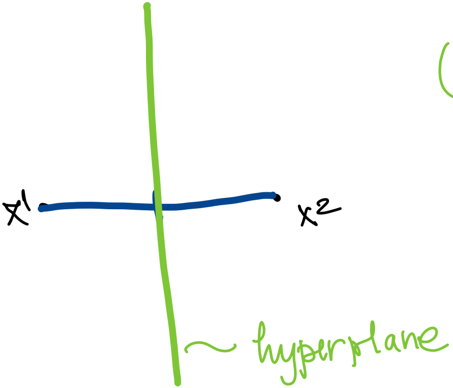
- **1-Nearest Neighbor** The label of a point x is assigned as follows:
  1. find the example x^i that is nearest to x in $\mathcal{D}$ (in Euclidean distance)
  2. assign x the label y^i, i.e.

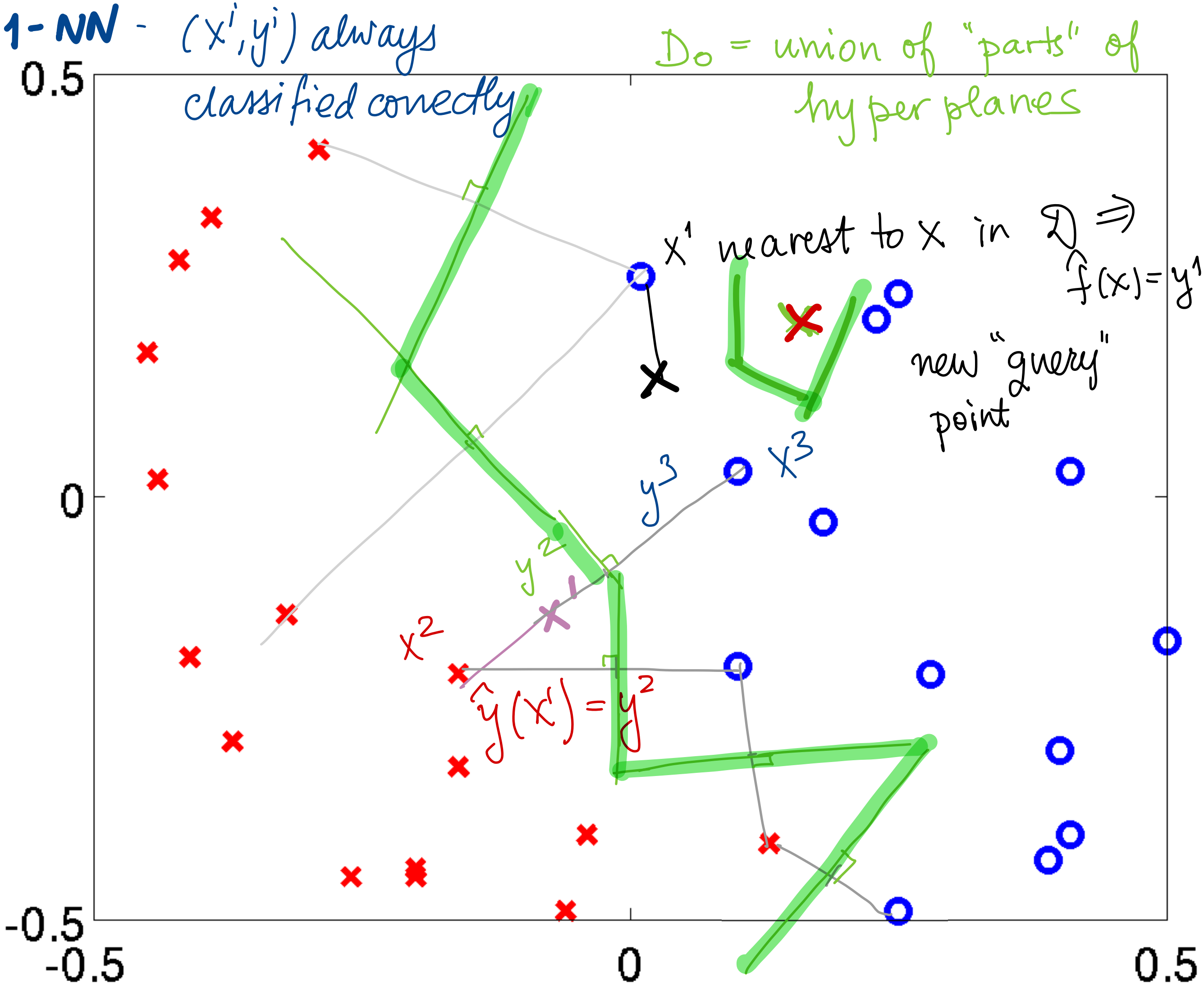
$$\hat{y}(x) = y^i$$

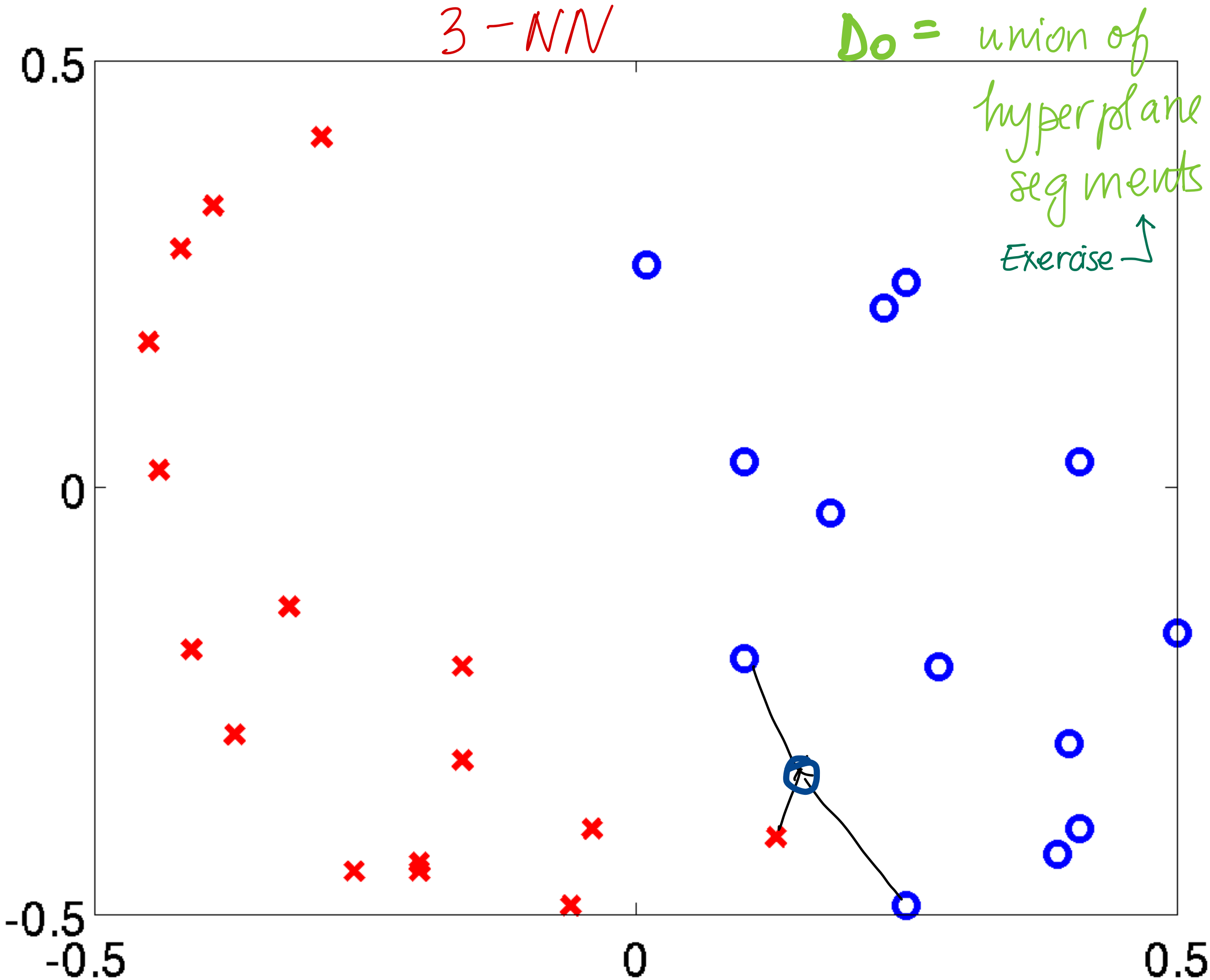
mediatrix $= \{ x \in \mathbb{R}^d \mid \|x - x^1\| = \|x - x^2\| \}$

distance(x, x^1)

(L^2, Euclidean)



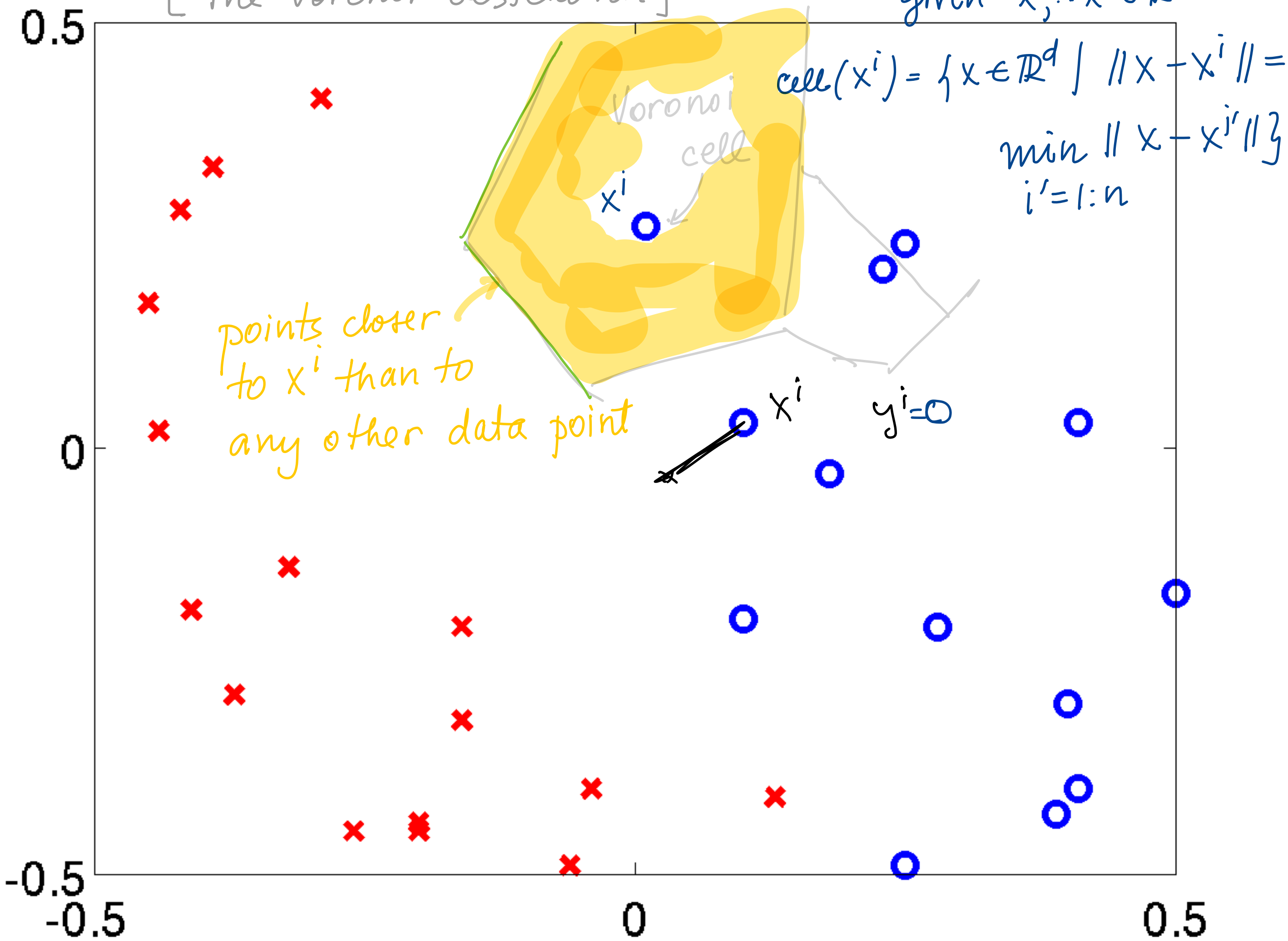

1-NN - (x^i, y^i) always classified correctly
D_0 = union of "parts" of hyperplanes
x^1 nearest to x in $\mathcal{D}$ ⇒
$\hat{f}(x) = y^1$
new "query" point
x^3
y^3
y^2
x^2
$\hat{y}(x^1) = y^2$
0.5
0
-0.5
-0.5
0
0.5


3-NN
D_0 = union of hyperplane segments
Exercise
0.5
0
-0.5
-0.5
0
0.5

[The Voronoi tesselation]

given $x^1, \dots x^n \in \mathbb{R}^d$

$$\text{cell}(x^i) = \{ x \in \mathbb{R}^d \mid \| x - x^i \| = \min_{i'=1:n} \| x - x^{i'} \| \}$$



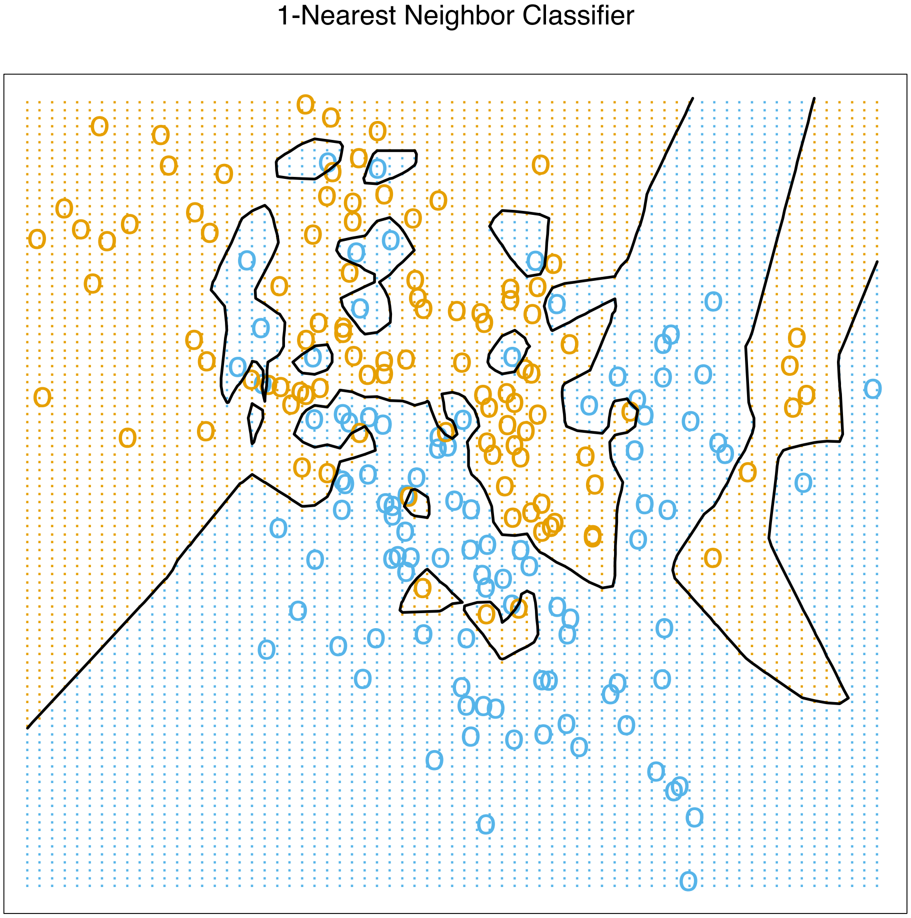


FIGURE 2.3. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$), *and then predicted by* 1*-nearest-neighbor classification.*

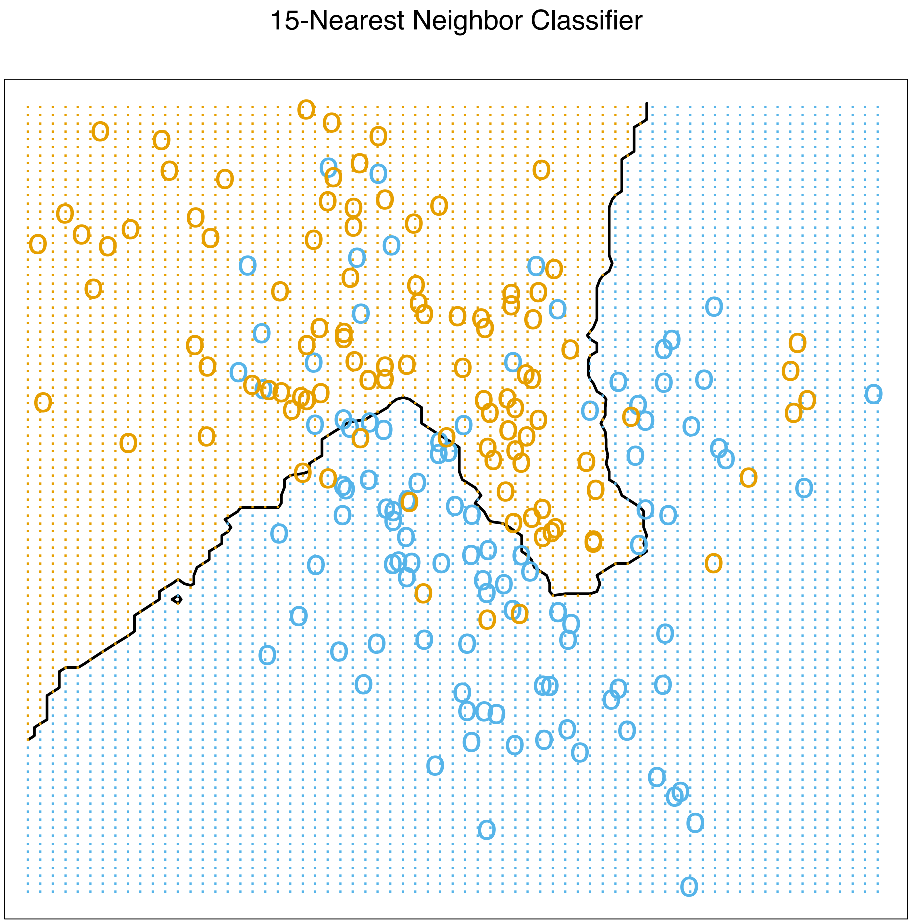
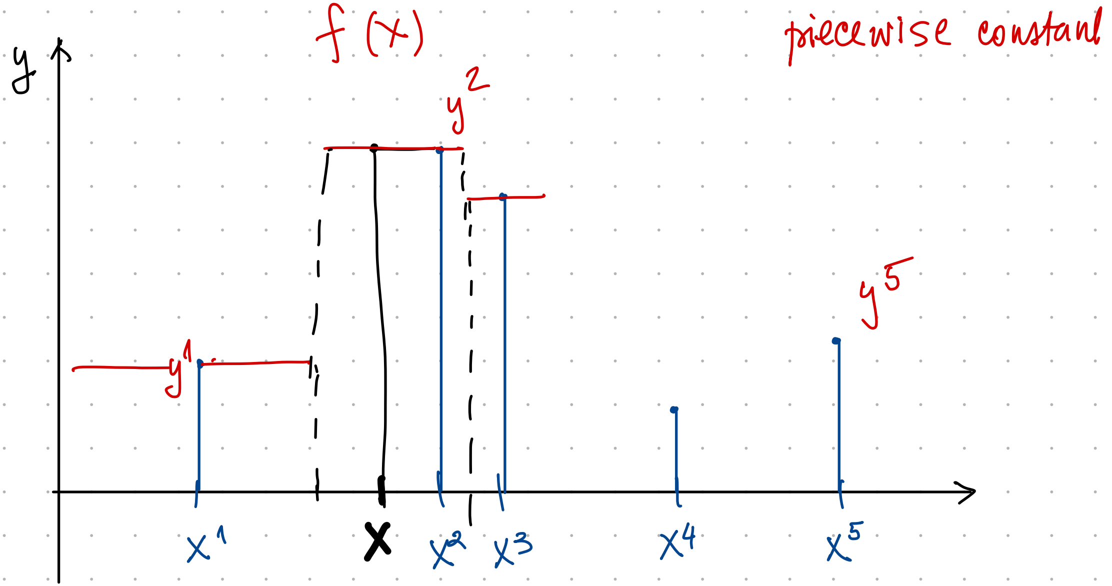


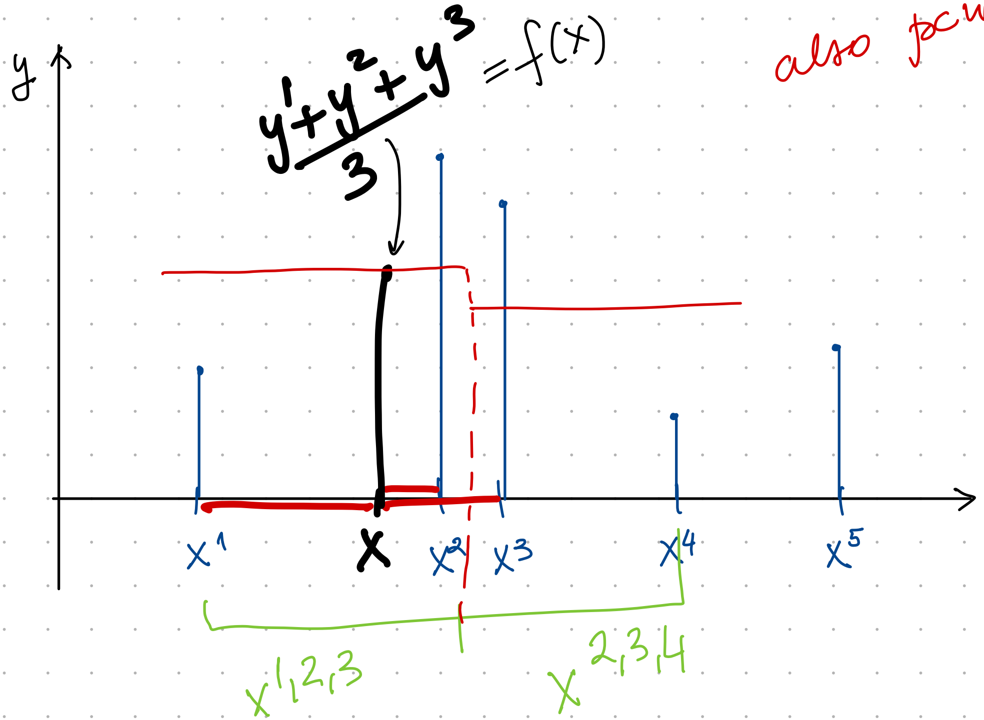
FIGURE 2.2. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$) *and then fit by* 15*-nearest-neighbor averaging as in (2.8). The predicted class is hence chosen by majority vote amongst the* 15*-nearest neighbors.*

1-NN regression



3-NN regression

also pcwise constant



The Nearest-Neighbor predictor

- **1-Nearest Neighbor** The label of a point x is assigned as follows:
  1. find the example x^i that is nearest to x in $\mathcal{D}$ (in Euclidean distance)
  2. assign x the label y^i, i.e.

$$\hat{y}(x) = y^i$$

- **K-Nearest Neighbor** (with $K = 3, 5$ or larger)
  1. find the K nearest neighbors of x in $\mathcal{D}$: $x^{i_1, \ldots i_K}$
  2. - for classification $f(x) =$ the most frequent label among the K neighbors (well suited for multiclass)
     - for regression $f(x) = \frac{1}{K} \sum_{i \text{ neighbor of } x} y^i$ = mean of neighbors' labels

multiway is natural

there could be ties

The Nearest-Neighbor predictor

- **1-Nearest Neighbor** The label of a point x is assigned as follows:
  1. find the example x^i that is nearest to x in $\mathcal{D}$ (in Euclidean distance)
  2. assign x the label y^i, i.e.

$$\hat{y}(x) = y^i$$

- **K-Nearest Neighbor** (with $K = 3, 5$ or larger)
  1. find the K nearest neighbors of x in $\mathcal{D}$: $x^{i_1, \ldots i_K}$
  2. - for classification $f(x) =$ the most frequent label among the K neighbors (well suited for multiclass)
     - for regression $f(x) = \frac{1}{K}\sum_{i \text{ neighbor of } x} y^i =$ mean of neighbors' labels

(hyperameter) smoothing parameter

- K must be chosen ← model selection problem

No training!

- No parameters to estimate!
- No training!
- But all data must be stored (also called memory-based learning)

Better with BIG data

- need to search $\mathcal{D}$ for each new x

n×d time/query

approximate NN search