Linear Models for Classification

Logistic Regression

Prof. Dr. Christoph Lippert

Digital Health & Machine Learning

Diagnosing breast cancer biopsies using Logistic Regression

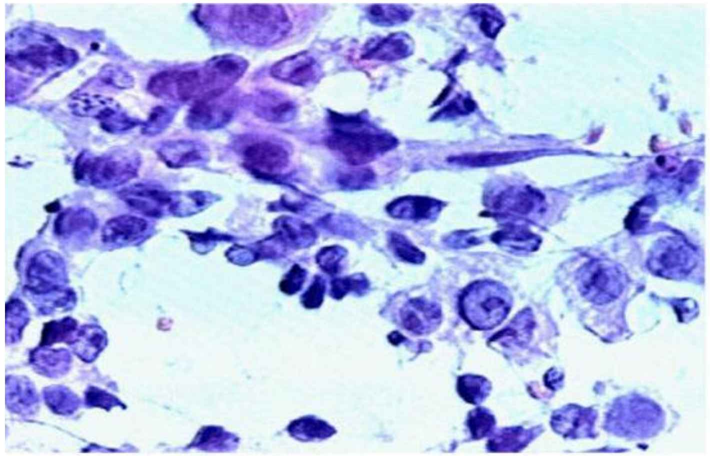
Given:

- **Training Data** with known diagnosis
  - 249 benign
  - 149 malignant
- **2 features** from pre-processed images

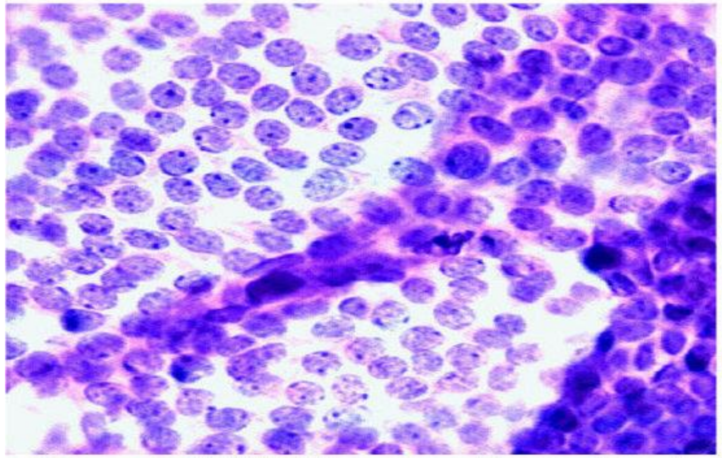
Task:

- **classify new** biopsies from features
  - c_1: Malignant (**M**)
  - c_2: Benign (**B**)

Fine needle aspartate biopsy images



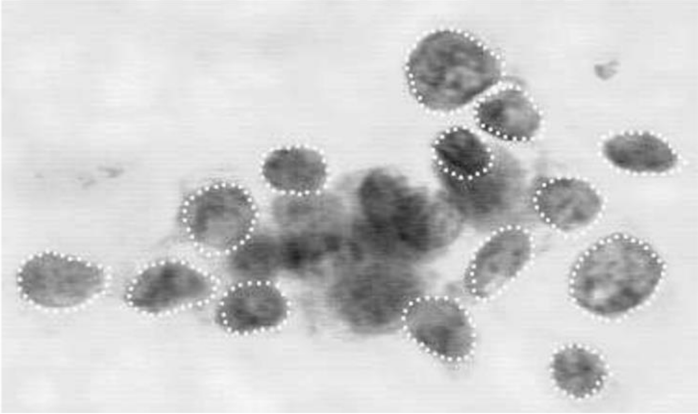
c_1: Malignant (**M**)



c_2: Benign (**B**)

Segmentation of nuclei

Pre-processing (given)



Features:

x_1 (concavity_mean):

- Fraction of chords outside nucleus

x_2 (texture_mean):

- Variance in gray-scale intensities

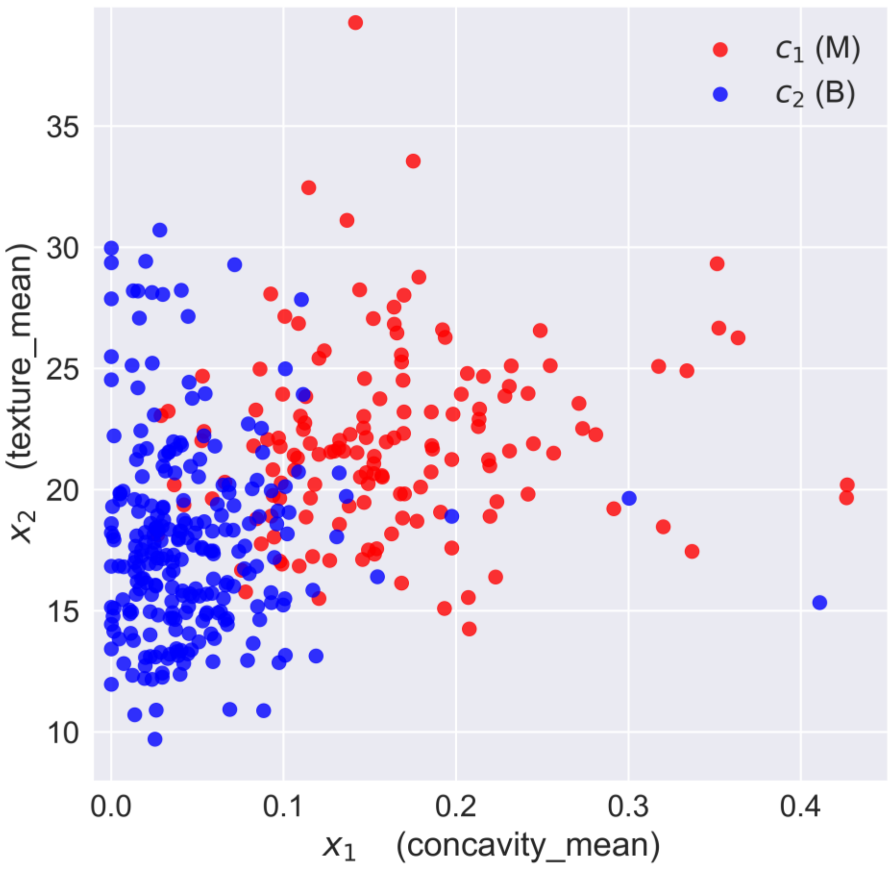
Wisconsin Diagnostic Breast Cancer (WDBC, 1993)

Linear Classification

Use the **training data** to find a **linear decision function**

$$x_1 w_1 + x_2 w_2 + b = 0$$

to **separate** the **two classes** $c_1 = \mathrm{M}$ and $c_2 = \mathrm{B}$.



Linear Classification

Use the **training data** to find a **linear decision function**

$$x_1 w_1 + x_2 w_2 + b = 0$$

to **separate** the **two classes** $c_1 = \mathrm{M}$ and $c_2 = \mathrm{B}$.

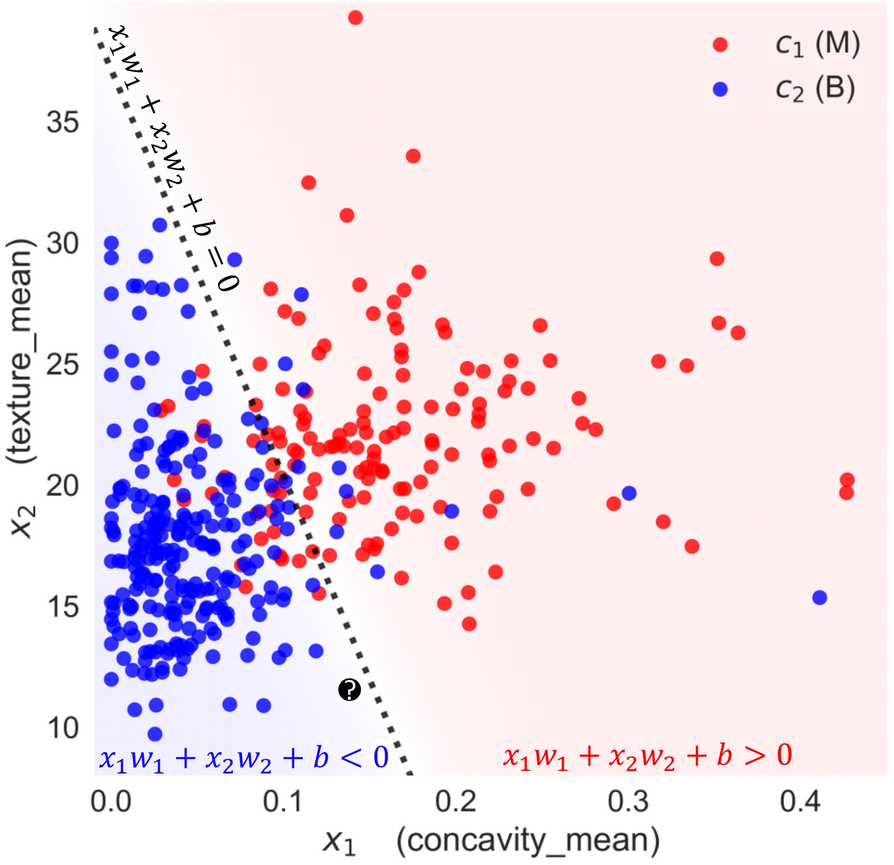
or equiv. $\mathbf{xw} = 0$ where $\mathbf{x} = [x_1 \quad x_2 \quad 1]$ **feature vector** (given)

and $\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix}$ **weight vector** (unknown)

Questions we will answer:

Q1. How to deal with **samples at the boundary**?

A1: Predict **probabilities** $0 \le p(y = c_1 | \, \mathbf{x}) \le 1$



Linear Classification

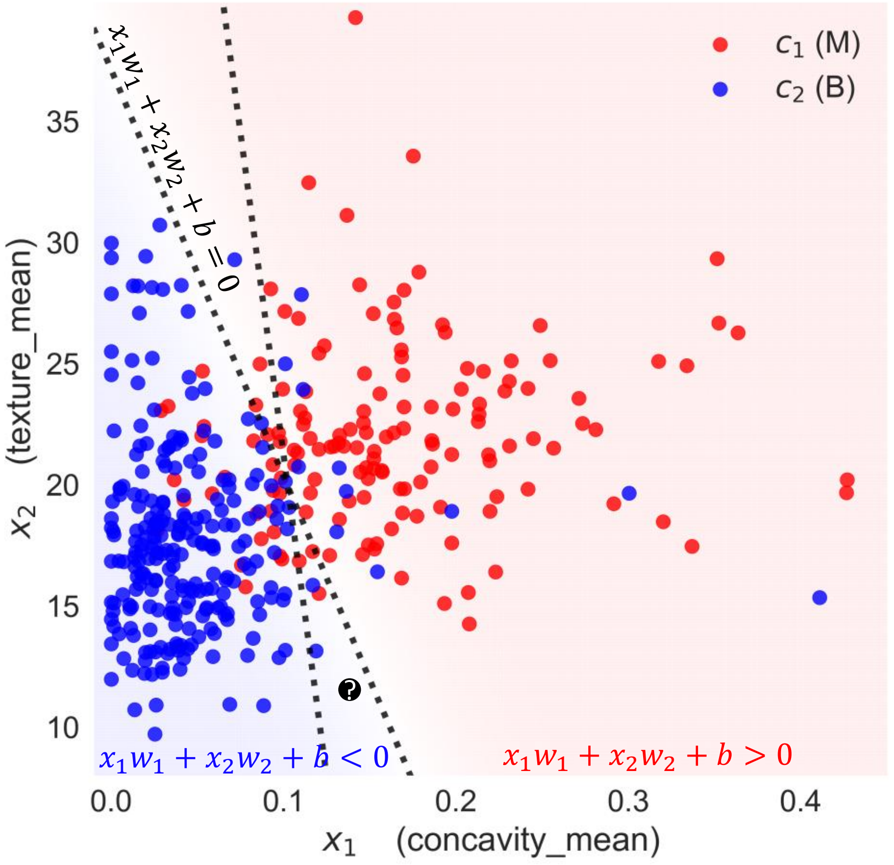
Use the **training data** to find a **linear decision function**

$$x_1 w_1 + x_2 w_2 + b = 0$$

to **separate** the **two classes** $c_1 = \mathrm{M}$ and $c_2 = \mathrm{B}$.

or equiv. $\quad \mathbf{x}\mathbf{w} = 0 \quad$ where $\mathbf{x} = [x_1 \quad x_2 \quad 1]$ **feature vector** (given)

and $\quad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix}$ **weight vector** (unknown)



Questions we will answer:

Q1. How to deal with **samples at the boundary**?

A1: Predict **probabilities** $\quad 0 \leq p(y = c_1 | \, \mathbf{x}) \leq 1$

Q2. How to **compare different** functions?

A2: log-loss

Q3. How to **determine** the **best** function?

A3: Use **optimization**

Q4. How to assess the **classifier performance**?

A4: Quality metrics

Q1. How to deal with **uncertain predictions**?

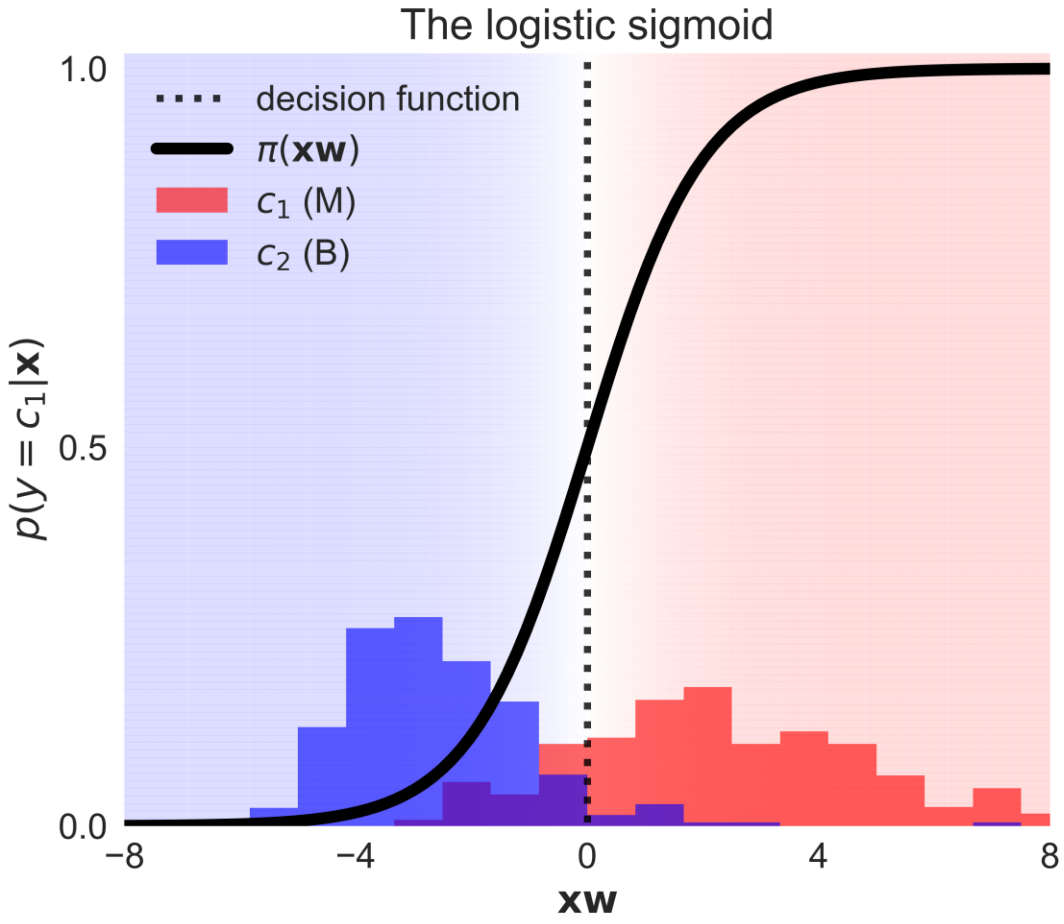
Predict **probabilities.**

The **logistic sigmoid:**

$$p(y = c_1 | \mathbf{x}) = \pi(\mathbf{xw})$$
$$= \frac{1}{1+\exp(-\mathbf{xw})}$$

By symmetry:

$$p(y = c_2 | \mathbf{x}) = 1 - \pi(\mathbf{xw})$$



Q2. How to **compare different** functions?

The **log-loss** function

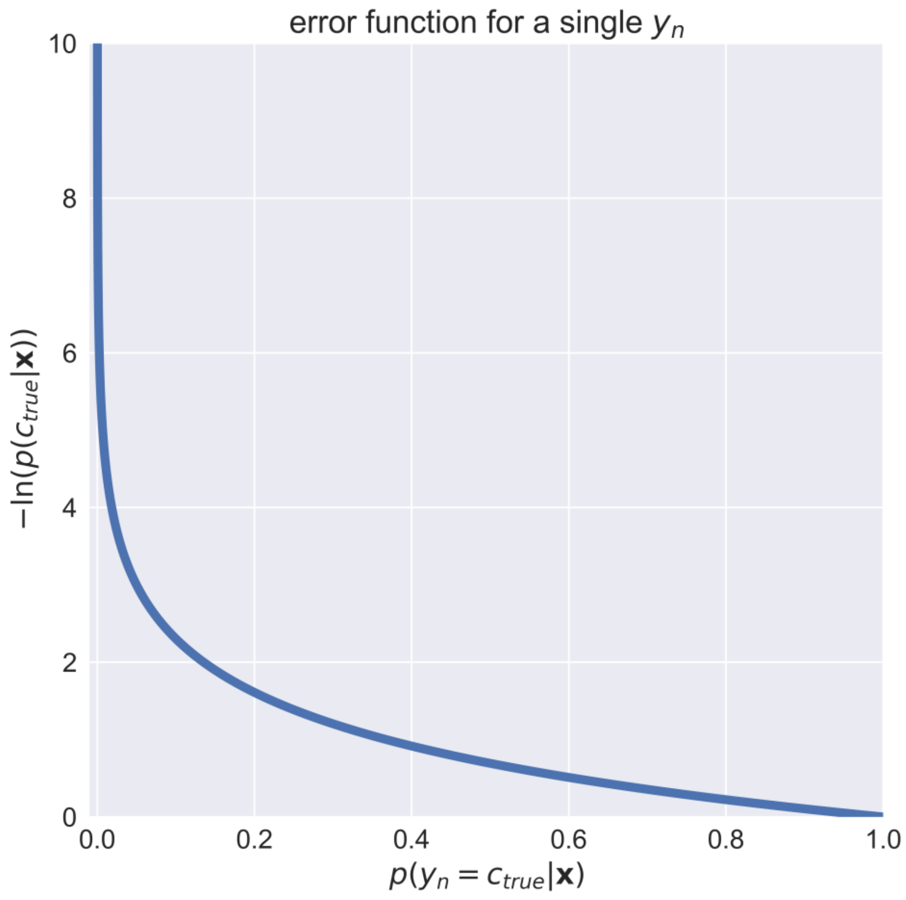
- **log-error** function for a **single sample**

$$-\ln p(y = c_{true}|\mathbf{x})$$

  - **Large**, when assigning **low probability**
  - **Small**, when assigning **high probability**

- **log-loss** function for **training data set**
  - Sum error functions for all data points

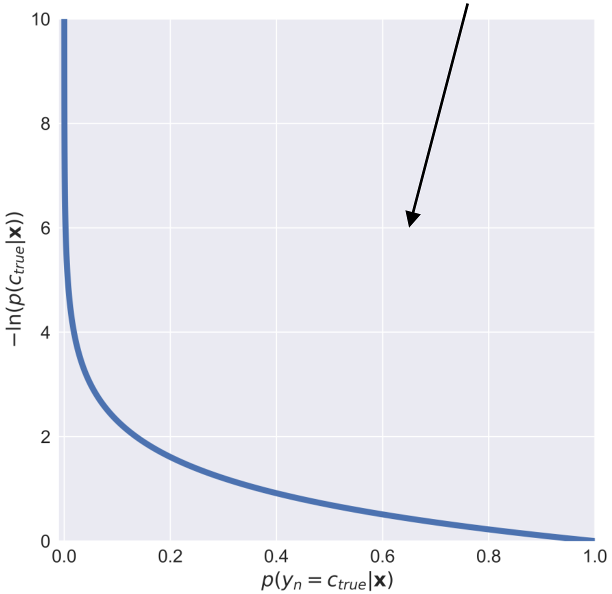
$$loss = -\sum_{n \in c_1} \ln(\pi(\mathbf{x}_n \mathbf{w})) - \sum_{n' \in c_2} \ln(1 - \pi(\mathbf{x}_{n'} \mathbf{w}))$$



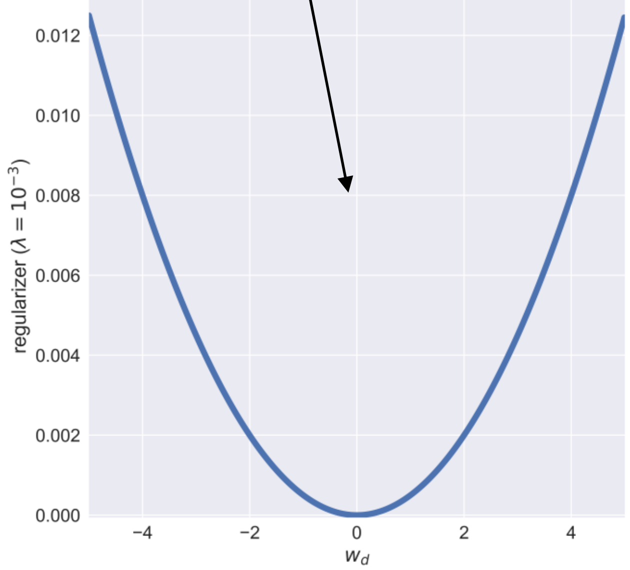
Q2. How to **compare different** functions?

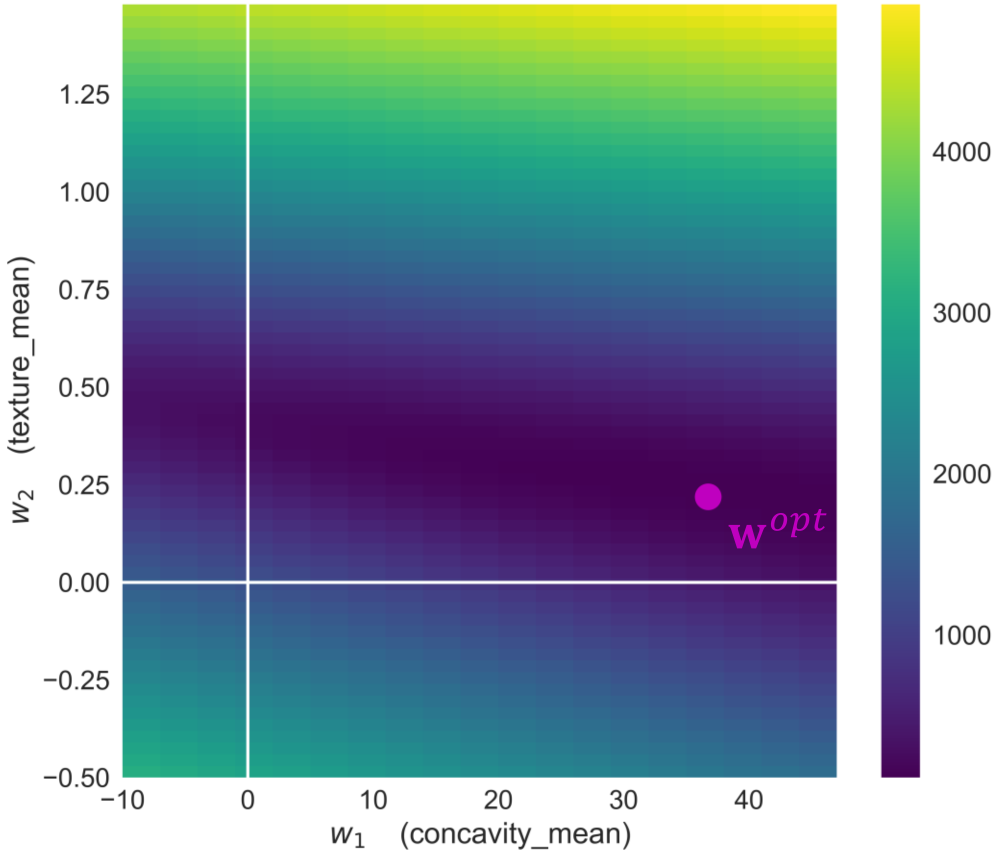
Objective function

$$L(\mathbf{w}) = \underbrace{-\sum_{n\in c_1} \ln(\pi(\mathbf{x}_n \mathbf{w})) - \sum_{n'\in c_2} \ln(1-\pi(\mathbf{x}_{n'} \mathbf{w}))}_{loss} + \underbrace{\lambda \cdot 0.5 \cdot \sum_{d=1}^{D} {w_d}^2}_{regularizer}$$



+





Q3. How to **determine** the **best** function?

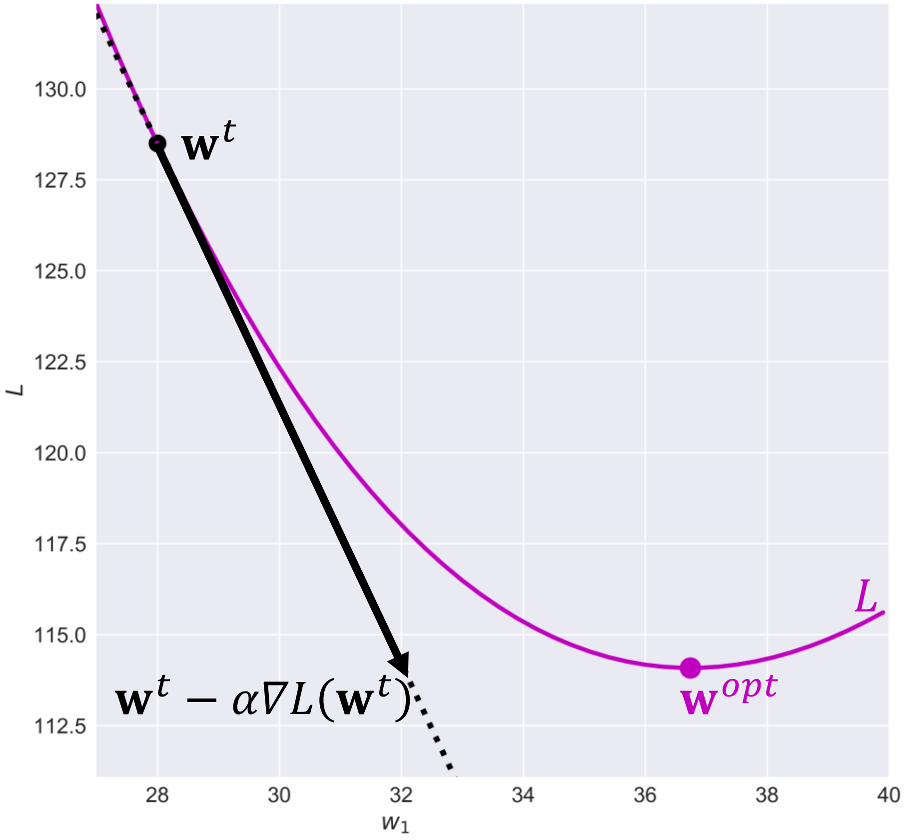
Steepest descent

$$L(\mathbf{w}) = \underbrace{-\sum_{n\in c_1} \ln(\pi(\mathbf{x}_n\mathbf{w})) - \sum_{n'\in c_2} \ln(1-\pi(\mathbf{x}_{n'}\mathbf{w}))}_{loss} + \underbrace{\lambda \cdot 0.5 \cdot \sum_{d=1}^{D} {w_d}^2}_{regularizer}$$

Gradient:

$$\nabla L\left(\mathbf{w}^t\right) = \begin{bmatrix} \frac{\partial L}{\partial w_1^t} \\ \vdots \\ \frac{\partial L}{\partial w_D^t} \end{bmatrix} = \underbrace{\mathbf{X}^T\left(\pi\left(\mathbf{X}\mathbf{w}^t\right) - I\left(\mathbf{y} == c_1\right)\right)}_{\nabla \mathrm{loss}\left(\mathbf{w}^t\right)} + \underbrace{\lambda \cdot \mathbf{w}^t}_{\nabla \mathrm{regularizer}\left(\mathbf{w}^t\right)}$$

direction of largest increase in $L(\mathbf{w}^t)$



Update rule:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \alpha\nabla L(\mathbf{w}^t)$$

for a small learning rate α (here, 10^{-4})

Q3. How to **determine** the **best** function?

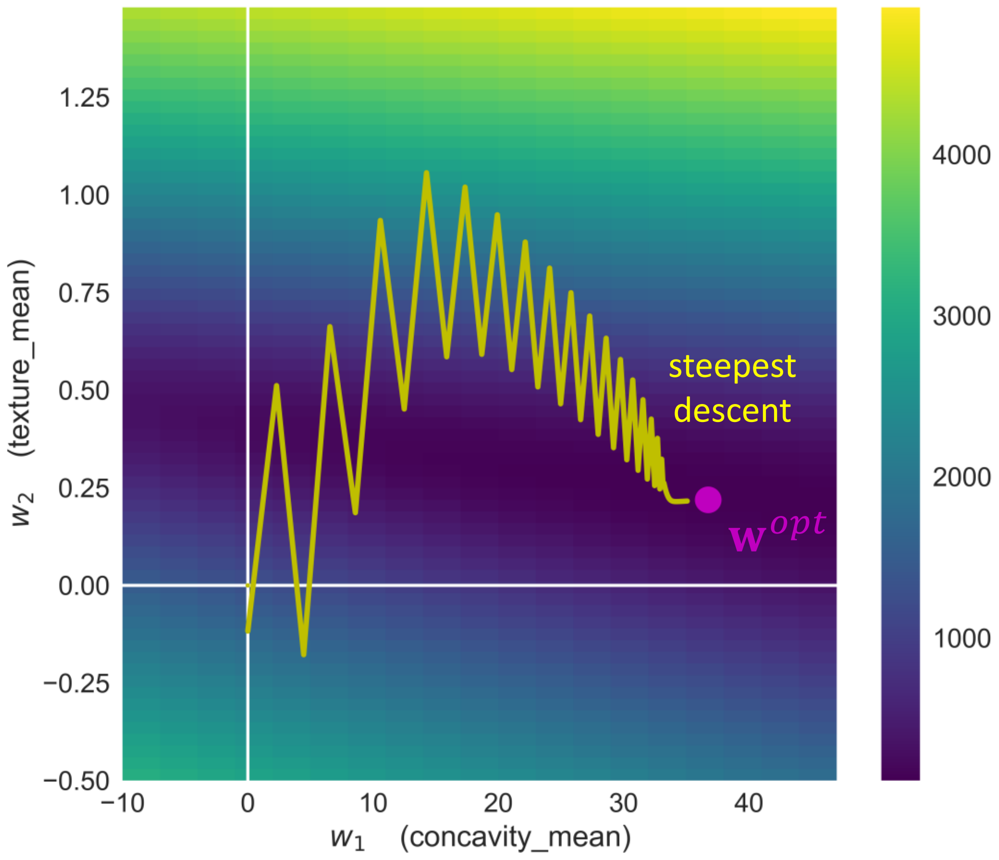
Steepest descent

$$L(\mathbf{w}) = \underbrace{-\sum_{n\in c_1} \ln(\pi(\mathbf{x}_n\mathbf{w})) - \sum_{n'\in c_2} \ln(1-\pi(\mathbf{x}_{n'}\mathbf{w}))}_{loss} + \underbrace{\lambda \cdot 0.5 \cdot \sum_{d=1}^{D} {w_d}^2}_{regularizer}$$

Gradient:

$$\nabla L\left(\mathbf{w}^t\right) = \begin{bmatrix} \frac{\partial L}{\partial w_1^t} \\ \vdots \\ \frac{\partial L}{\partial w_D^t} \end{bmatrix} = \underbrace{\mathbf{X}^T\left(\pi\left(\mathbf{X}\mathbf{w}^t\right) - I\left(\mathbf{y} == c_1\right)\right)}_{\nabla \text{loss}\left(\mathbf{w}^t\right)} + \underbrace{\lambda \cdot \mathbf{w}^t}_{\nabla \text{regularizer}\left(\mathbf{w}^t\right)}$$

direction of largest increase in $L(\mathbf{w}^t)$



Update rule:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \alpha\nabla L(\mathbf{w}^t)$$

for a small learning rate α (here, 10^{-4})

Q3. How to **determine** the **best** function?

Account for the **curvature**!

$$L(\mathbf{w}) = \underbrace{-\sum_{n\in c_1} \ln(\pi(\mathbf{x}_n\mathbf{w})) - \sum_{n'\in c_2} \ln(1-\pi(\mathbf{x}_{n'}\mathbf{w}))}_{loss} + \underbrace{\lambda \cdot 0.5 \cdot \sum_{d=1}^{D} {w_d}^2}_{regularizer}$$

Hessian:

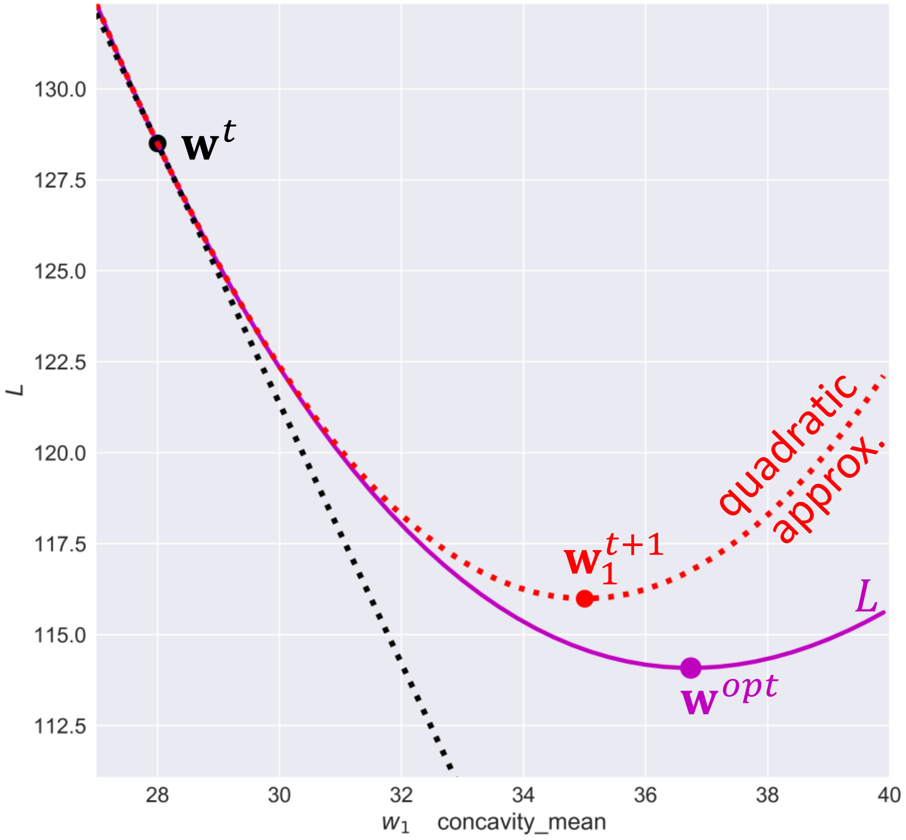
$$\mathbf{H}_{\mathbf{w}^t} = \begin{bmatrix} \partial^2 L/\partial^2 w_1 & \partial^2 L/\partial w_1 \partial w_2 & \dots & \partial^2 L/\partial w_1 \partial w_D \\ \vdots & & \ddots & \vdots \\ \partial^2 L/\partial w_D \partial w_1 & \partial^2 L/\partial w_D \partial w_2 & \dots & \partial^2 L/\partial^2 w_D \end{bmatrix}$$

$$= \underbrace{\left(\mathbf{X}\mathrm{diag}\left(\pi\left(\mathbf{X}\mathbf{w}^t\right) \cdot \left(\mathbf{1} - \pi(\mathbf{X}\mathbf{w}^t)\right)\right)\right)^T \mathbf{X}}_{\mathbf{H}_{\mathbf{w}^t}(\mathrm{loss})}$$

Update rule:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \mathbf{H}_{\mathbf{w}^t}^{-1} \nabla L\left(\mathbf{w}^t\right)$$

Newton-Raphson algorithm



Q3. How to **determine** the **best** function?

Account for the **curvature**!

$$L(\mathbf{w}) = \underbrace{-\sum_{n\in c_1}\ln(\pi(\mathbf{x}_n\mathbf{w})) - \sum_{n'\in c_2}\ln(1-\pi(\mathbf{x}_{n'}\mathbf{w}))}_{loss} + \underbrace{\lambda\cdot 0.5\cdot\sum_{d=1}^{D} {w_d}^2}_{regularizer}$$

Hessian:

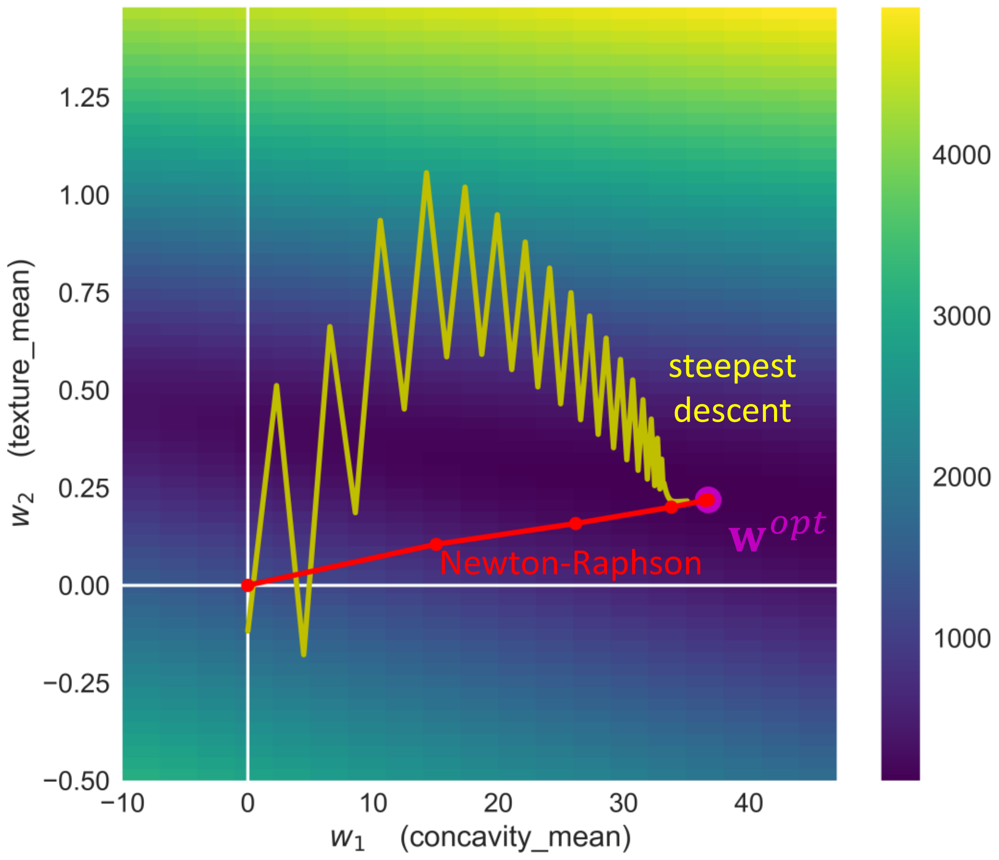
$$\mathbf{H}_{\mathbf{w}^t} = \begin{bmatrix} \partial^2 L/\partial^2 w_1 & \partial^2 L/\partial w_1\partial w_2 & \dots & \partial^2 L/\partial w_1\partial w_D \\ \vdots & & \ddots & \vdots \\ \partial^2 L/\partial w_D\partial w_1 & \partial^2 L/\partial w_D\partial w_2 & \dots & \partial^2 L/\partial^2 w_D \end{bmatrix}$$

$$= \underbrace{\left(\mathbf{X}\mathrm{diag}\left(\pi\left(\mathbf{X}\mathbf{w}^t\right)\cdot\left(1-\pi(\mathbf{X}\mathbf{w}^t)\right)\right)\right)^T\mathbf{X}}_{\mathbf{H}_{\mathbf{w}^t}(\mathrm{loss})} + \underbrace{\lambda\cdot\mathbf{I}_{D\times D}}_{\mathbf{H}_{\mathbf{w}^t}(\mathrm{regularizer})}$$

Update rule:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \mathbf{H}_{\mathbf{w}^t}^{-1}\nabla L\left(\mathbf{w}^t\right)$$

Newton-Raphson algorithm



Q4. How to **evaluate the model**?

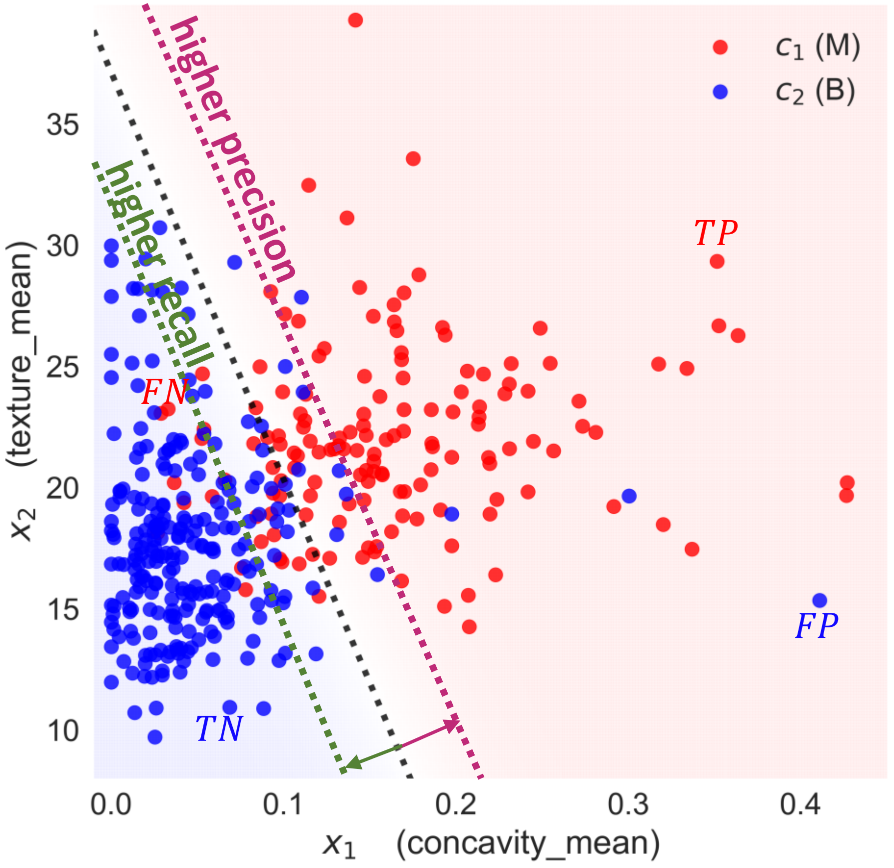
Quality measures

$$\textbf{accuracy} = \frac{1}{N}\ \#\ \text{correct}$$

$$\textbf{recall} = \frac{TP}{TP+FN}$$

$$\textbf{precision} = \frac{TP}{TP+FP}$$

	Predicted Positive	Predicted Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)



- Accept $p(y_1|\mathbf{x})$< 0.5 to increase sensitivity
- Require $p(y_1|\mathbf{x})$>0.5 to increase specificity

Q4. How to **evaluate the model**?

train vs. test

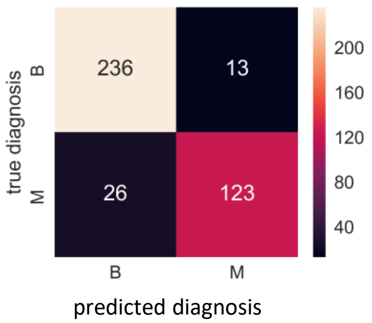
$$\textbf{accuracy} = \frac{1}{N}\ \#\ \text{correct}$$

$$\textbf{recall} = \frac{TP}{TP+FN}$$

$$\textbf{precision} = \frac{TP}{TP+FP}$$

	Predicted Positive	**Predicted Negative**
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

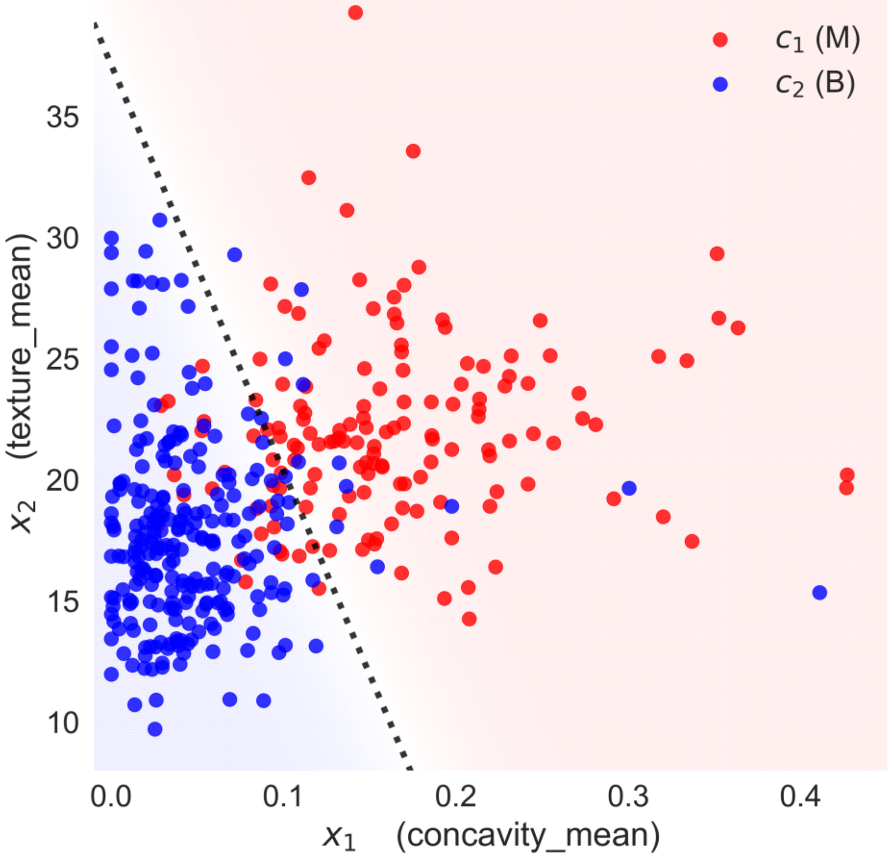
398 training samples



accuracy 90%

recall 83%

precision 90%



Q4. How to **evaluate the model**?

train vs. test

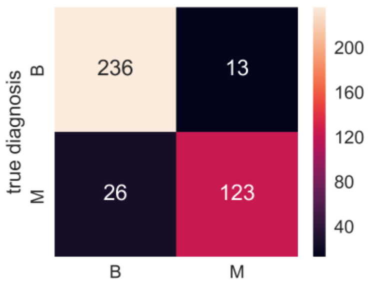
$$\textbf{accuracy} = \frac{1}{N}\ \#\ \text{correct}$$

$$\textbf{recall} = \frac{TP}{TP+FN}$$

$$\textbf{precision} = \frac{TP}{TP+FP}$$

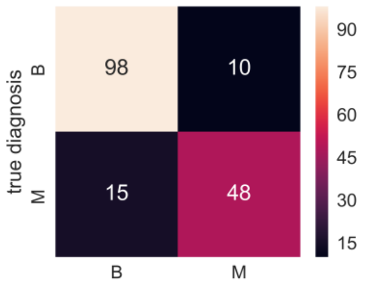
	Predicted Positive	Predicted Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

398 training samples

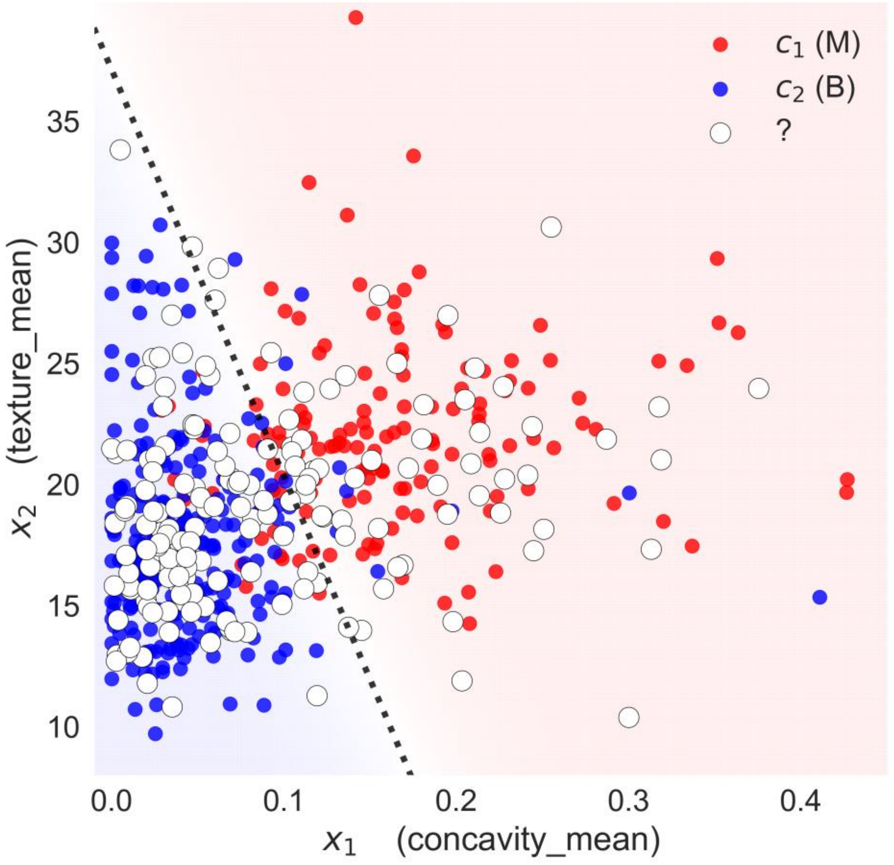


accuracy 90%
recall 83%
precision 90%

171 test samples



accuracy 85%
recall 76%
precision 83%

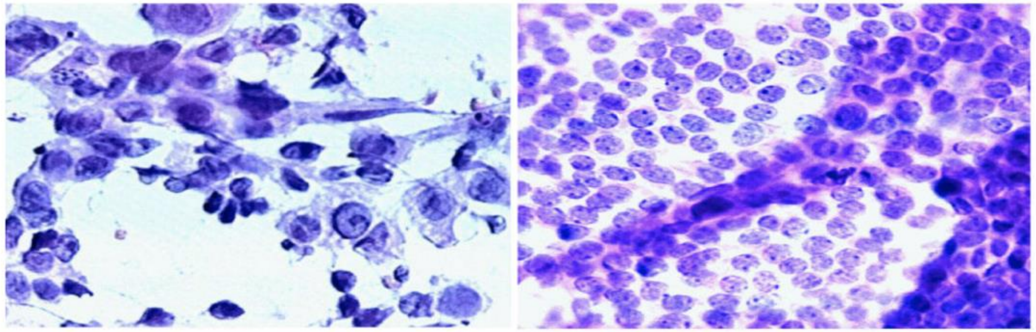


Observation:

- **Lower performance** on **test data**
- **Overfitting** to **training data**
- Test errors yield **unbiased** performance estimates

Recap:
Diagnosing breast cancer using Logistic Regression

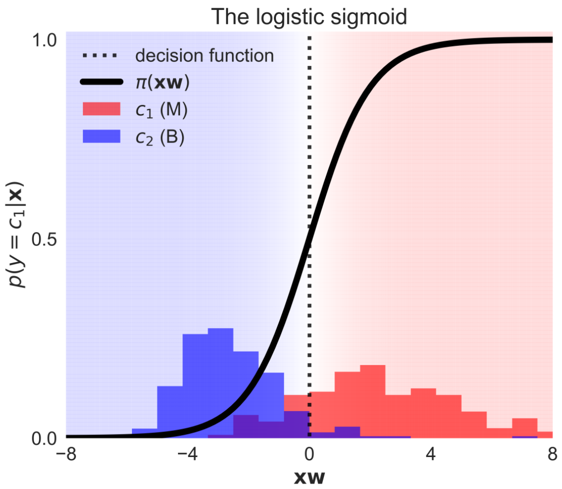
Fine needle aspartate biopsy images



c_1: Malignant (**M**) c_2: Benign (**B**)

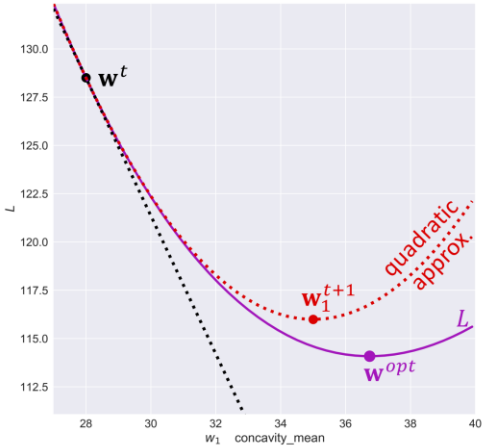
Logistic Regression model

- Linear classification
- Predicting probabilities



Model fitting

- log-loss
- Optimizing the log-loss



Model evaluation

- Precision vs. recall
- train vs. test

	Predicted Positive	**Predicted Negative**
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)