UNIVERSIDAD DE CHILE

Minería de Datos

Welcome to the Machine Learning class

Valentin Barriere

Universidad de Chile – DCC

CC5205, Fall 2025

Supervised Learning – Intro

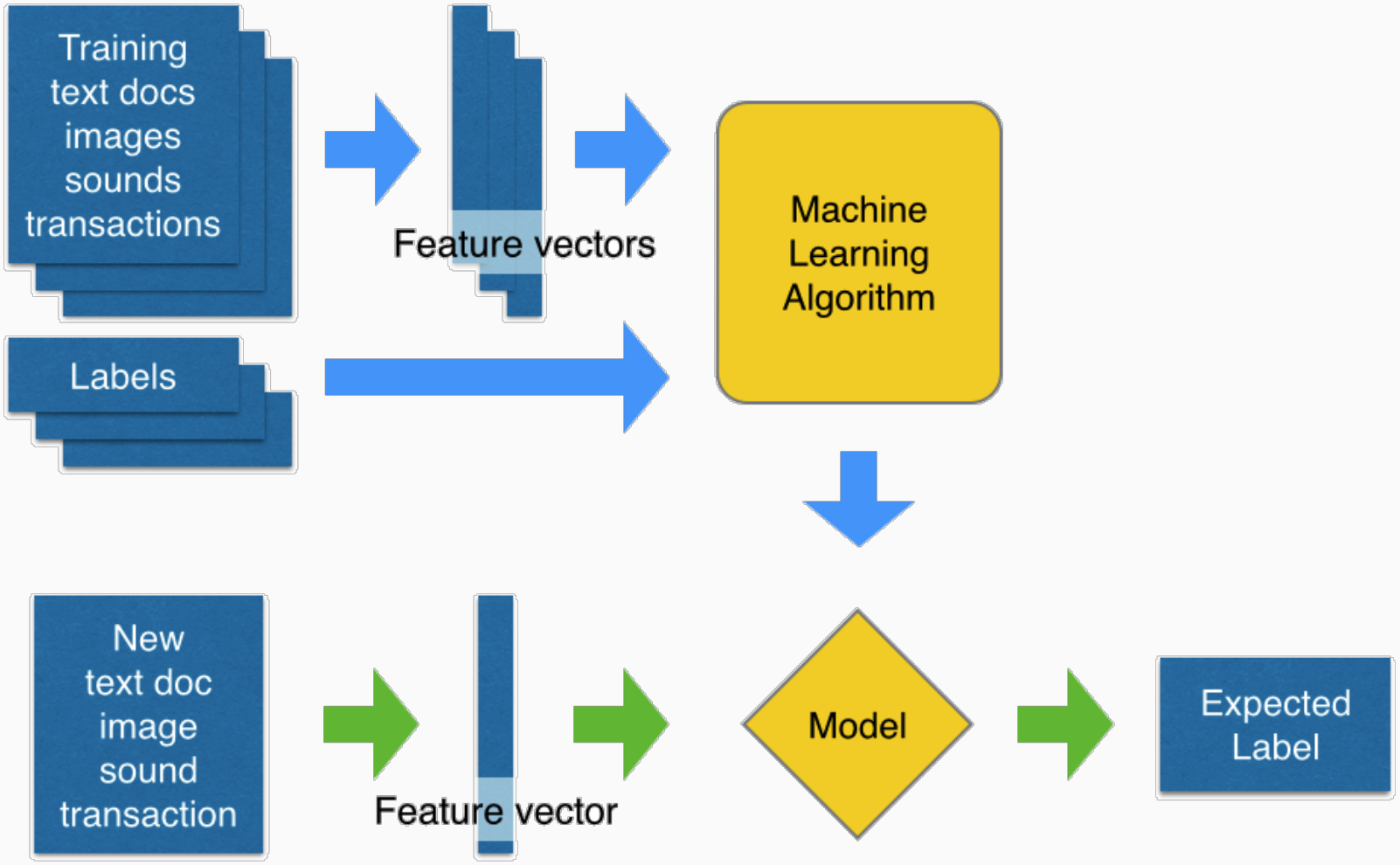
Outline: Supervised Learning

Global Framework

Outline : Global Framework

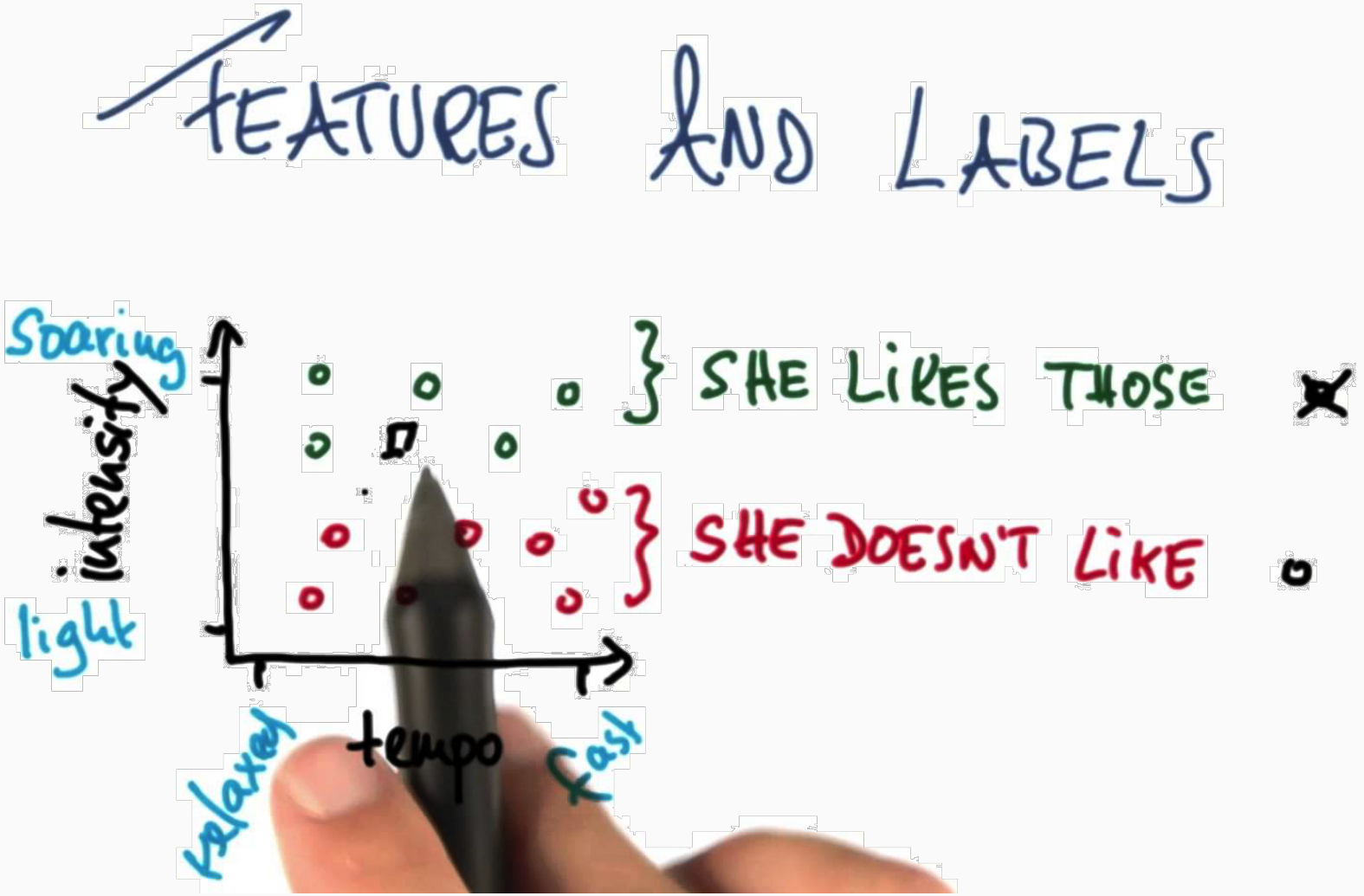
Outline : Generalities

Supervised Learning



Predictive Modeling Data Flow

Descriptors and Labels



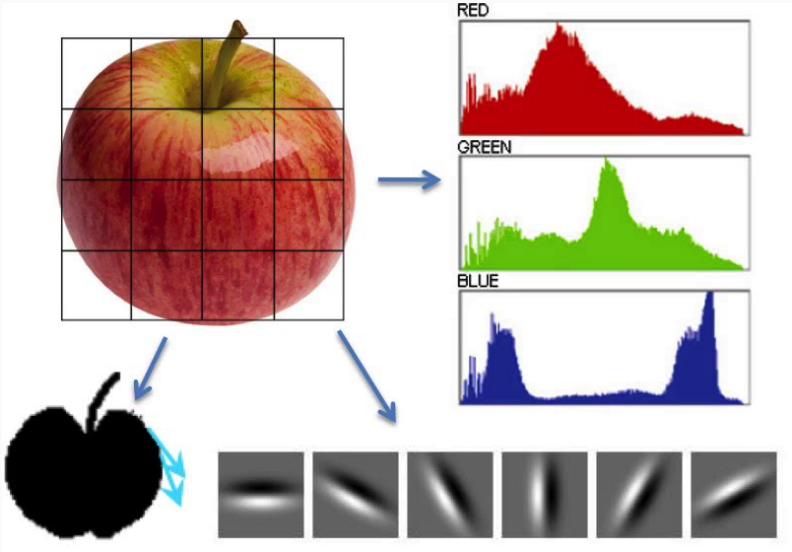
Attributes

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

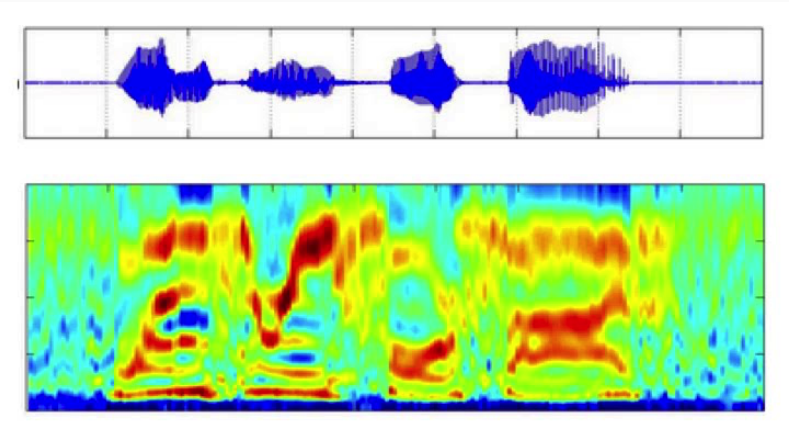
Figure 1: A structured database

Attributes

- Images: Color Histogram



- Sounds: Time-Frequency Representation



In Summary

I **Having Labeled Data**

II **Extracting the Descriptors**: transforming documents into vectors

III **Creating a Mathematical Model f_θ**

IV **Implementing a Cost (Error) Function ℓ to Minimize**

V **Finding the Parameters θ^* Such That $\ell(f_{\theta^*}(\mathbf{X}_i), Y_i)$ Is Small**

VI **Testing f_{θ^*} on New Data Using an Appropriate Evaluation Metric**

In Summary

I **Having Labeled Data**
- Dataset of size n, $\mathcal{D}_n = \{(\text{Doc}_i, Y_i), i = 1 \dots n\}$
- Doc is a sample (for example: a person)
- Y are the labels (for example: the granted loan amount)

II **Extracting the Descriptors**: transforming documents into vectors

III **Creating a Mathematical Model f_θ**

IV **Implementing a Cost (Error) Function ℓ to Minimize**

V **Finding the Parameters θ^* Such That $\ell(f_{\theta^*}(\mathbf{X}_i), Y_i)$ Is Small**

VI **Testing f_{θ^*} on New Data Using an Appropriate Evaluation Metric**

In Summary

I **Having Labeled Data**
- Dataset of size n, $\mathcal{D}_n = \{(\text{Doc}_i, Y_i), i = 1 \dots n\}$
- Doc is a sample (for example: a person)
- Y are the labels (for example: the granted loan amount)

II **Extracting the Descriptors**: transforming documents into vectors
- **X** is a vector of features (for example: age, gender, salary)
- Y are the labels (for example: the granted loan amount)

III **Creating a Mathematical Model f_θ**

IV **Implementing a Cost (Error) Function ℓ to Minimize**

V **Finding the Parameters θ^* Such That $\ell(f_{\theta^*}(\mathbf{X}_i), Y_i)$ Is Small**

VI **Testing f_{θ^*} on New Data Using an Appropriate Evaluation Metric**

In Summary

I **Having Labeled Data**
- Dataset of size n, $\mathcal{D}_n = \{(\text{Doc}_i, Y_i), i = 1 \dots n\}$
- Doc is a sample (for example: a person)
- Y are the labels (for example: the granted loan amount)

II **Extracting the Descriptors**: transforming documents into vectors
- $\mathbf{X}$ is a vector of features (for example: age, gender, salary)
- Y are the labels (for example: the granted loan amount)

III **Creating a Mathematical Model f_θ**
- A model f_θ such that $f_\theta(\mathbf{X})$ is close to Y (for regression)
- θ is the set of parameters of the mathematical model

IV **Implementing a Cost (Error) Function ℓ to Minimize**

V **Finding the Parameters θ^* Such That $\ell(f_{\theta^*}(\mathbf{X}_i), Y_i)$ Is Small**

VI **Testing f_{θ^*} on New Data Using an Appropriate Evaluation Metric**

In Summary

I **Having Labeled Data**
- Dataset of size n, $\mathcal{D}_n = \{(\text{Doc}_i, Y_i), i = 1 \dots n\}$
- Doc is a sample (for example: a person)
- Y are the labels (for example: the granted loan amount)

II **Extracting the Descriptors**: transforming documents into vectors
- $\mathbf{X}$ is a vector of features (for example: age, gender, salary)
- Y are the labels (for example: the granted loan amount)

III **Creating a Mathematical Model f_θ**
- A model f_θ such that $f_\theta(\mathbf{X})$ is close to Y (for regression)
- θ is the set of parameters of the mathematical model

IV **Implementing a Cost (Error) Function ℓ to Minimize**
- The more the model errs, the higher the cost
- In general, a small cost is desired

V **Finding the Parameters θ^* Such That $\ell(f_{\theta^*}(\mathbf{X}_i), Y_i)$ Is Small**

VI **Testing f_{θ^*} on New Data Using an Appropriate Evaluation Metric**

In Summary

I **Having Labeled Data**
- Dataset of size n, $\mathcal{D}_n = \{(\text{Doc}_i, Y_i), i = 1 \dots n\}$
- Doc is a sample (for example: a person)
- Y are the labels (for example: the granted loan amount)

II **Extracting the Descriptors**: transforming documents into vectors
- $\mathbf{X}$ is a vector of features (for example: age, gender, salary)
- Y are the labels (for example: the granted loan amount)

III **Creating a Mathematical Model f_θ**
- A model f_θ such that $f_\theta(\mathbf{X})$ is close to Y (for regression)
- θ is the set of parameters of the mathematical model

IV **Implementing a Cost (Error) Function ℓ to Minimize**
- The more the model errs, the higher the cost
- In general, a small cost is desired

V **Finding the Parameters θ^* Such That $\ell(f_{\theta^*}(\mathbf{X}_i), Y_i)$ Is Small**
- $\theta^* = \arg\min_\theta \sum_i \ell(f_\theta(\mathbf{X}_i), Y_i)$

VI **Testing f_{θ^*} on New Data Using an Appropriate Evaluation Metric**

In Summary

I **Having Labeled Data**
- Dataset of size n, $\mathcal{D}_n = \{(\text{Doc}_i, Y_i), i = 1 \dots n\}$
- Doc is a sample (for example: a person)
- Y are the labels (for example: the granted loan amount)

II **Extracting the Descriptors**: transforming documents into vectors
- $\mathbf{X}$ is a vector of features (for example: age, gender, salary)
- Y are the labels (for example: the granted loan amount)

III **Creating a Mathematical Model f_θ**
- A model f_θ such that $f_\theta(\mathbf{X})$ is close to Y (for regression)
- θ is the set of parameters of the mathematical model

IV **Implementing a Cost (Error) Function ℓ to Minimize**
- The more the model errs, the higher the cost
- In general, a small cost is desired

V **Finding the Parameters θ^* Such That $\ell(f_{\theta^*}(\mathbf{X}_i), Y_i)$ Is Small**
- $\theta^* = \arg\min_\theta \sum_i \ell(f_\theta(\mathbf{X}_i), Y_i)$

VI **Testing f_{θ^*} on New Data Using an Appropriate Evaluation Metric**

Outline : Theoretical Context

Supervised Learning I

Mathematical Framework

- Input measurement: $\mathbf{X} = (X^{(1)}, X^{(2)}, \ldots, X^{(d)}) \in \mathcal{X}$
- Output measurement: $Y \in \mathcal{Y}$
- $(X, Y) \sim \mathbf{P}$ where $\mathbf{P}$ is unknown
- Training set: $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$
- Often:
  - $\mathbf{X} \in \mathbb{R}^d$ and $Y \in \{1, \ldots, C\}$ for a classification task
  - $\mathbf{X} \in \mathbb{R}^d$ and $Y \in \mathbb{R}$ for a regression task

A classifier is a function in $\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y}\}$ (measurable).

Objective

Construct a satisfactory classifier $\hat{f}$ using the training data.

Note: $\hat{f}$ depends on $\mathcal{D}_n$.

Supervised Learning II

Learning, Task, Performance Measure

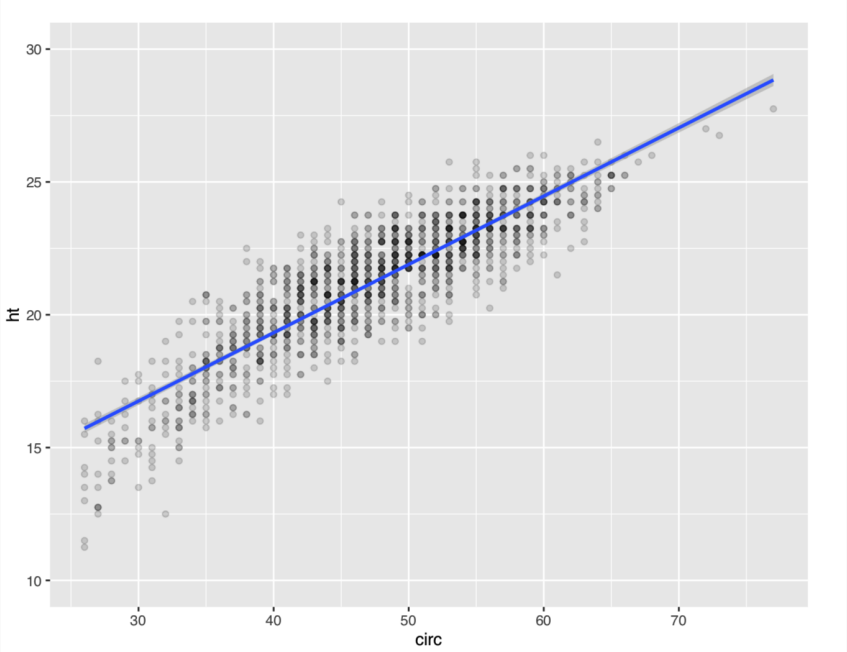
- Training set: $\mathcal{D}_n = \{(\mathbf{X}_1, Y_1), \ldots, (\mathbf{X}_n, Y_n)\}$
- Classifier: $f : \mathcal{X} \to \mathcal{Y}$ (measurable).
- Cost/Loss Function: $\ell(Y, f(\mathbf{X}))$ measures the quality of f's *prediction* relative to Y
- Risk:

$$\mathcal{R}(f) = \mathbb{E}[\ell(Y, f(\mathbf{X}))]$$

Objective

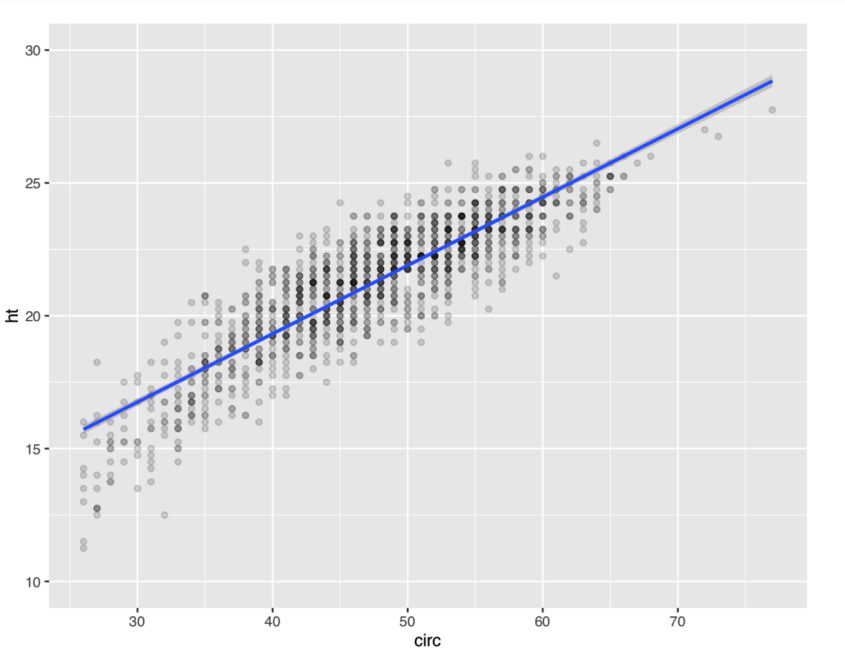
Learn to construct a classifier $f \in \mathcal{F}$ from the training data $\mathcal{D}_n$ such that the risk $\mathcal{R}(f)$ is small on average with respect to $\mathcal{D}_n$.

Example: Eucalyptus



- A simple and classic database
- Predict the size based on the circumference
- $\mathbf{X} = \mathbf{circ}$
- $Y = ht$

Example: Eucalyptus



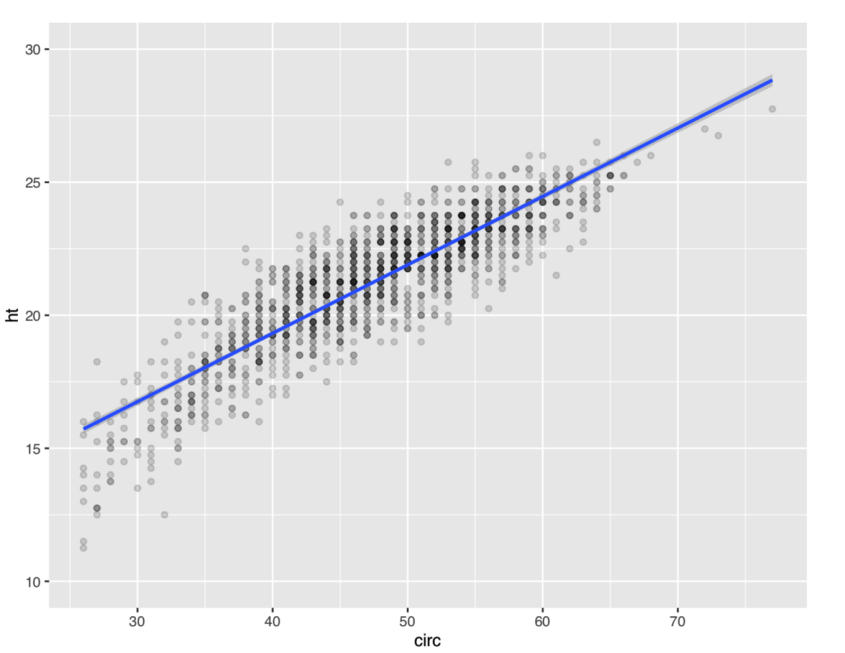
Linear Model

- Parametric model:

$$f_\beta(\textbf{circ}) = \beta_1 + \beta_2 \textbf{circ} = \hat{ht}$$

- How to choose $\beta = (\beta_1, \beta_2)$?

Example: Least Squares



Methodology

- Naturally:

$$\sum_{i=1}^{n} |Y_i - f_\beta(\mathbf{X}_i)|^2 = \sum_{i=1}^{n} |ht_i - f_\beta(\mathbf{circ}_i)|^2 = \sum_{i=1}^{n} |ht_i - (\beta_1 + \beta_2 \mathbf{circ}_i)|^2$$

- Choose β that minimizes this criterion:

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^2}{\arg\min} \sum_{i=1}^{n} |ht_i - (\beta_1 + \beta_2 \mathbf{circ}_i)|^2$$

Heuristics

Linear Regression

- **Statistical Model**: $(\mathbf{circ}_i, ht_i)$ i.i.d. with the same distribution as a generic $(\mathbf{circ}, ht)$
- **Performance Criterion**: Find an f with a low average error

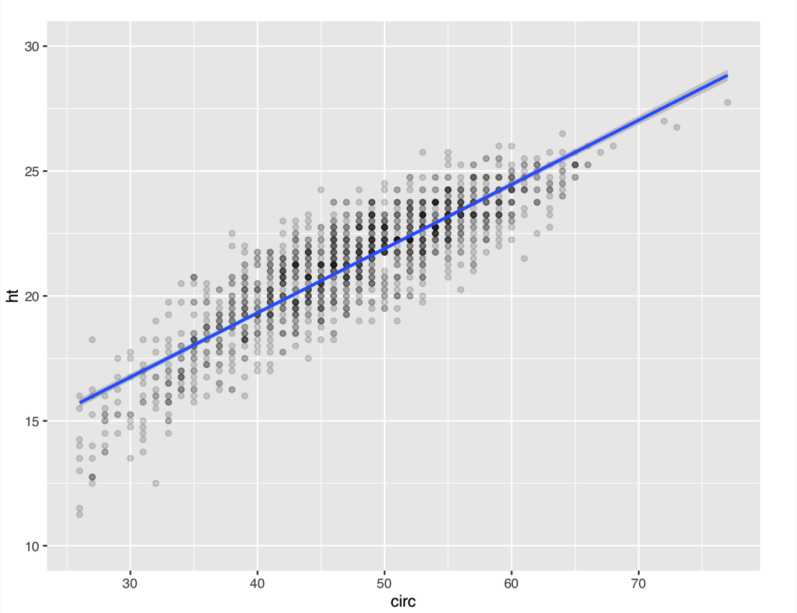
$$\mathbb{E}[|ht - f(\mathbf{circ})|^2]$$

- **Empirical Criterion**: Replace the unknown distribution with its empirical counterpart

$$\frac{1}{n}\sum_{i=1}^{n} |ht_i - f(\mathbf{circ}_i)|^2$$

- **Choosing a Classifier**: Avoid an overly complex model; stick to the simplest model that yields good results (e.g., a large neural network/too many features for very few examples)
- **Learning the Model**: Optimize over the data

Example: Which Degree to Choose?



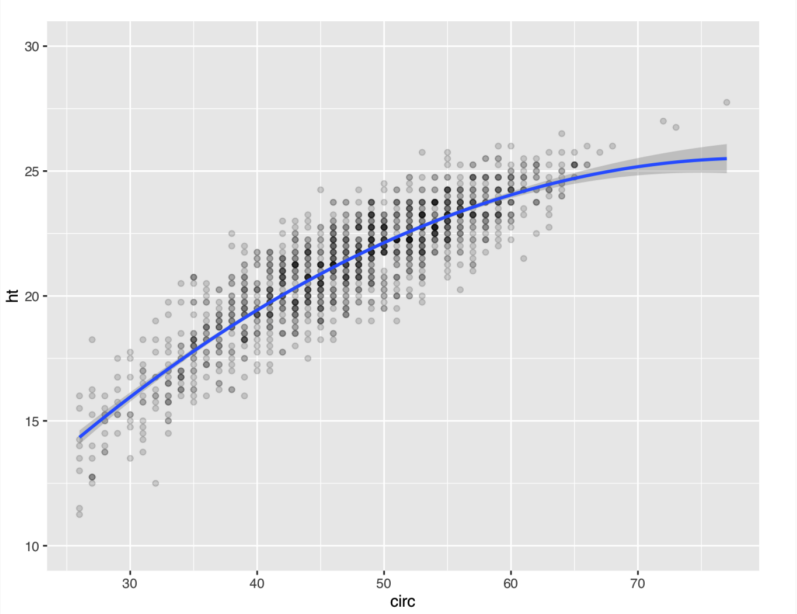
Models

- Increasing the polynomial degree d corresponds to an increasing model complexity :

$$\beta \in \mathbb{R}^d = \text{more parameters}$$

- This model no longer fits the data it is shown.
- Overfitting: it fits $\mathcal{D}_n$ too closely and fails to generalize.

Example: Which Degree to Choose?



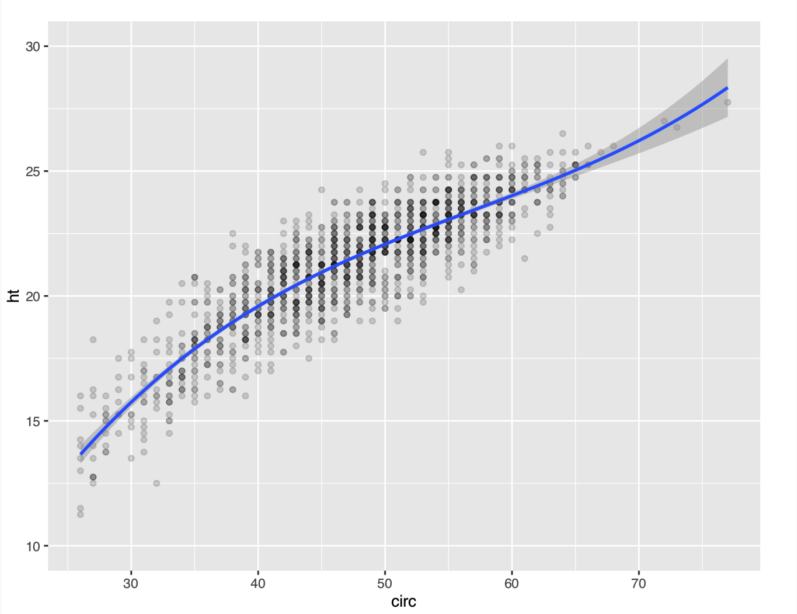
Models

- Increasing the polynomial degree d corresponds to an increasing model complexity :

$$\beta \in \mathbb{R}^d = \text{more parameters}$$

- This model no longer fits the data it is shown.
- Overfitting: it fits $\mathcal{D}_n$ too closely and fails to generalize.

Example: Which Degree to Choose?



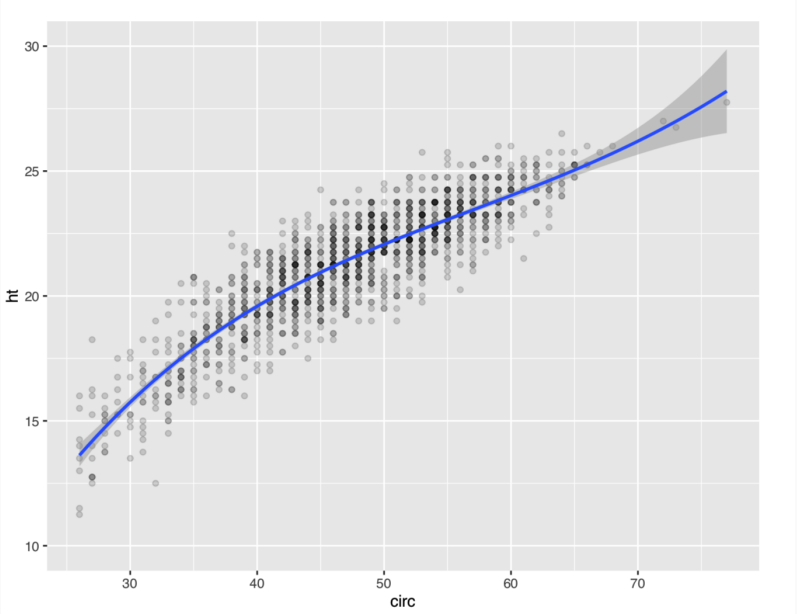
Models

- Increasing the polynomial degree d corresponds to an increasing model complexity :

$$\beta \in \mathbb{R}^d = \text{more parameters}$$

- This model no longer fits the data it is shown.
- Overfitting: it fits $\mathcal{D}_n$ too closely and fails to generalize.

Example: Which Degree to Choose?



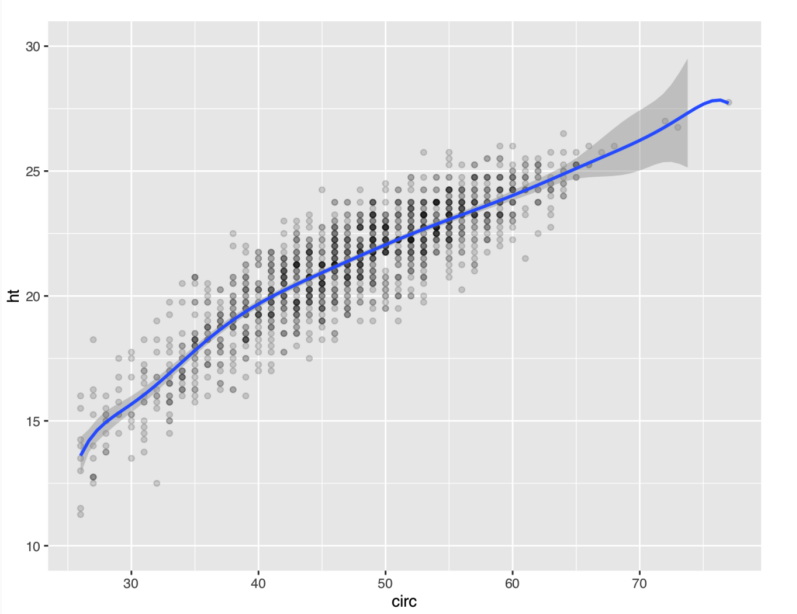
Models

- Increasing the polynomial degree d corresponds to an increasing model complexity :

$$\beta \in \mathbb{R}^d = \text{more parameters}$$

- This model no longer fits the data it is shown.
- Overfitting: it fits $\mathcal{D}_n$ too closely and fails to generalize.

Example: Which Degree to Choose?



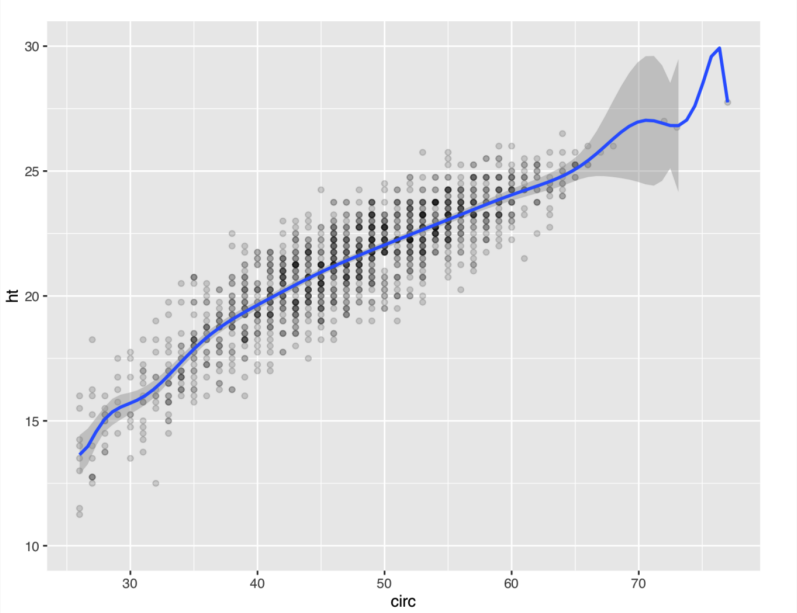
Models

- Increasing the polynomial degree d corresponds to an increasing model complexity :

$$\beta \in \mathbb{R}^d = \text{more parameters}$$

- This model no longer fits the data it is shown.
- Overfitting: it fits $\mathcal{D}_n$ too closely and fails to generalize.

Example: Which Degree to Choose?



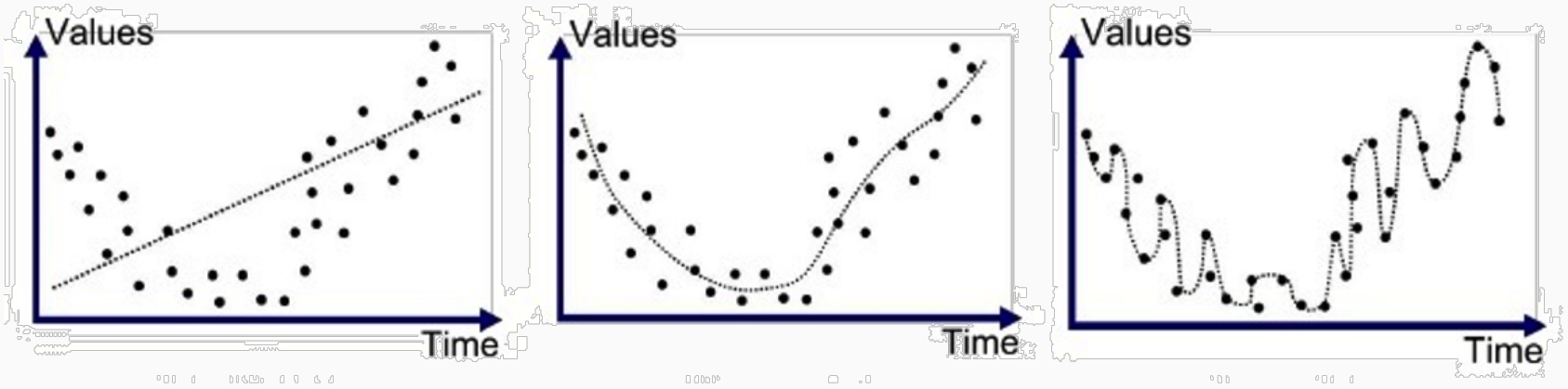
Models

- Increasing the polynomial degree d corresponds to an increasing model complexity :

$$\beta \in \mathbb{R}^d = \text{more parameters}$$

- This model no longer fits the data it is shown.
- Overfitting: it fits $\mathcal{D}_n$ too closely and fails to generalize.

Example: Which Degree to Choose?



Underfitted | Good Fit/Robust | Overfitted

Complexity

- If the model is too simple, it does not follow the data.
- If the model is too complex, it learns all the irregularities of the dataset $\mathcal{D}_n$.
- Example: if the model consists of the central curve plus a noise component not accounted for in the features, the model on the right learns that noise.

Outline : Learning

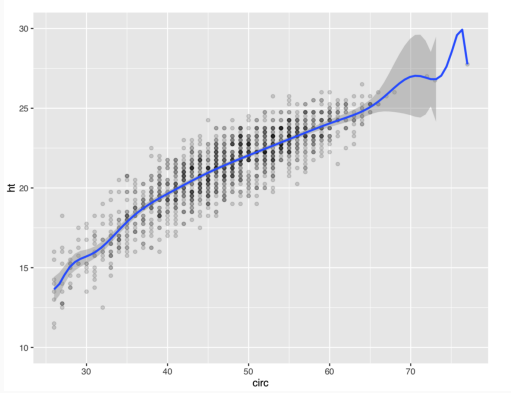
Supervised Learning

Different Concepts

- Labeled Data: Regression or Classification
- Feature Extraction: Tone, Intensity, Tempo or Age, Salary, Gender, etc.
- Model f_θ: SVM, Logistic Regression, Random Forest, CNN
- Cost Function to optimize: Hinge Loss, Cross-Entropy Loss, Logistic Loss, Squared Loss, etc.
- Optimization Algorithm: Adam, SGD, BFGS, etc.
- Evaluation Metric: Recall, Precision, Mean Squared Error, etc.

In the previous case:

- Features and labels:
- Model:
- Cost Function:
- Optimizer:
- Evaluation Metric:



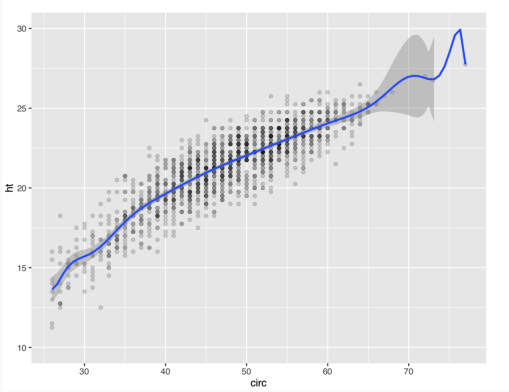
Supervised Learning

Different Concepts

- Labeled Data: Regression or Classification
- Feature Extraction: Tone, Intensity, Tempo or Age, Salary, Gender, etc.
- Model f_θ: SVM, Logistic Regression, Random Forest, CNN
- Cost Function to optimize: Hinge Loss, Cross-Entropy Loss, Logistic Loss, Squared Loss, etc.
- Optimization Algorithm: Adam, SGD, BFGS, etc.
- Evaluation Metric: Recall, Precision, Mean Squared Error, etc.

In the previous case:

- Features and labels: Circumference and height
- Model: Degree-6 Polynomial
- Cost Function: Squared Loss
- Optimizer: Stochastic Gradient Descent (SGD)
- Evaluation Metric: Mean Squared Error



Training: Cost Function ℓ

Principle

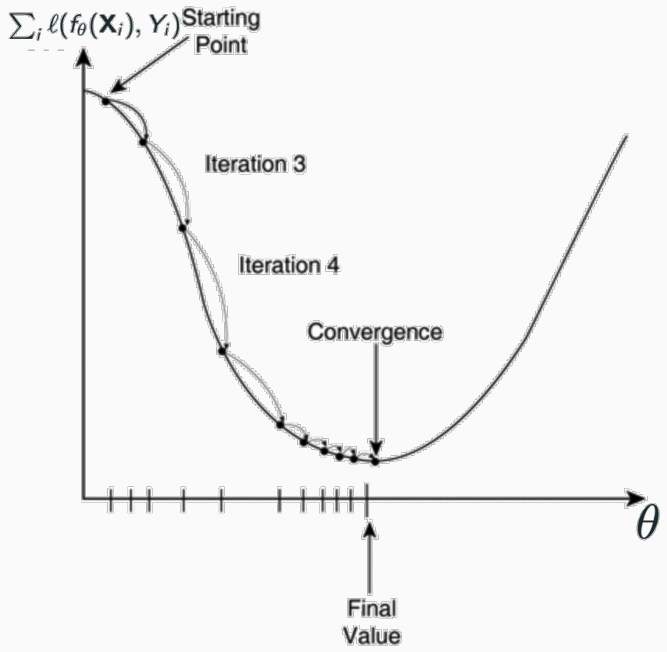
- A function that penalizes the model when it makes errors.
- Minimize this function over the training set (empirical risk) to find satisfactory model parameters:

$$f_{\hat{\theta}} = \arg\min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(\mathbf{X}_i))$$

- Convexity: typically the 0/1 cost function $\ell^{0/1}$ is replaced by a convex function ℓ' that is easier to optimize.

- It expresses the error from a numerical perspective.
- It conveys to the learning algorithm what is important for the task.
- It must be a function that can be efficiently optimized (convex).
  The function $\ell^{0/1} = \mathbb{1}_{f(\mathbf{X})=Y}$ is not usable (not even continuous).

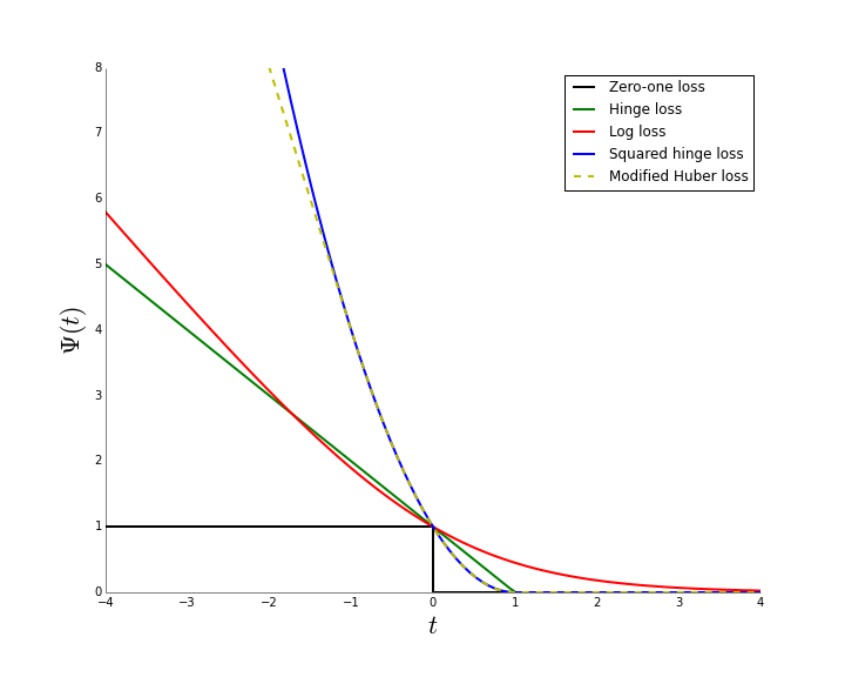
Cost Function and Convexity



Examples of Classic Cost Functions

- Logistic (Softmax): $\ell'(Y, f(\mathbf{X})) = \log(1 + \exp^{-Yf(\mathbf{X})})$
- Hinge: $\ell'(Y, f(\mathbf{X})) = (1 - Yf(\mathbf{X}))_+$
- Exponential: $\ell'(Y, f(\mathbf{X})) = \exp^{-Yf(\mathbf{X})}$
- Cross-Entropy: $\ell'(Y, f(\mathbf{X})) = -(Y \ln(f(\mathbf{X})) + (1 - Y) \ln(1 - f(\mathbf{X})))$

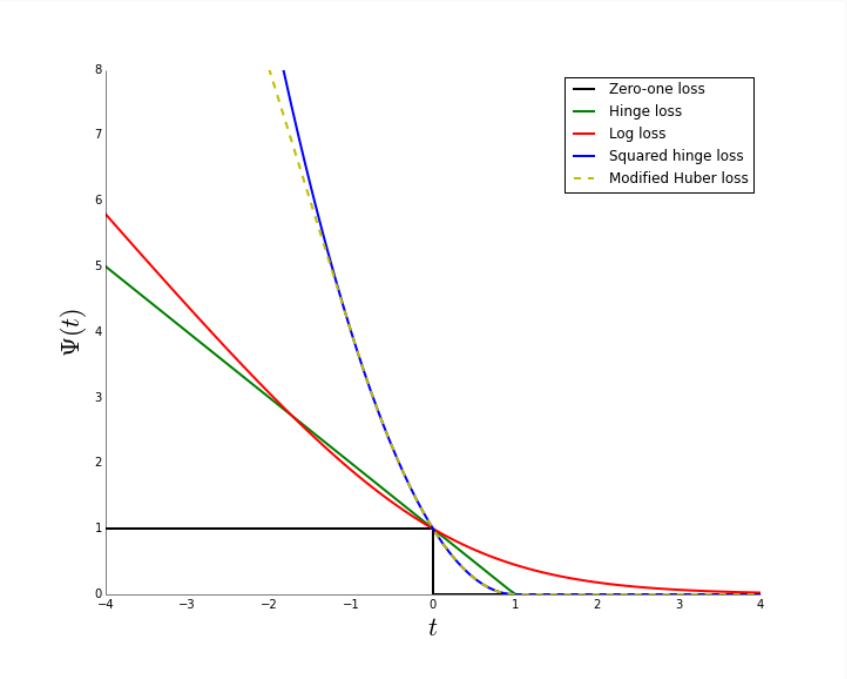
Cost Function and Convexity



Examples of Classic Cost Functions

- Logistic (Softmax): $\ell'(Y, f(\mathbf{X})) = \log(1 + \exp^{-Yf(\mathbf{X})})$
- Exponential: $\ell'(Y, f(\mathbf{X})) = \exp^{-Yf(\mathbf{X})}$
- With $Y = \pm 1$, we want $f(\mathbf{X})$ to be very positive/negative for $Y = +1/-1$.
- If $f(\mathbf{X}) = \text{sign}(Y)$, then $\exp^{-Yf(\mathbf{X})}$ is small.

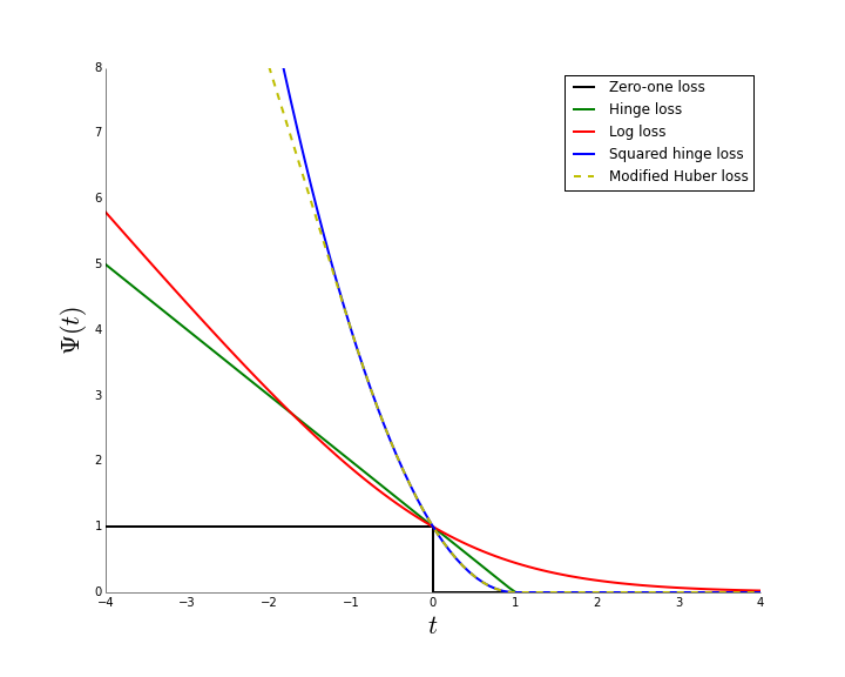
Cost Function and Convexity



Examples of Classic Cost Functions

- Squared: $\ell'(Y, f(\mathbf{X})) = (1 - f(\mathbf{X})Y)^2$
- With $Y = \pm 1$, we require exactly $Y = f(\mathbf{X})$ (both sign and magnitude).

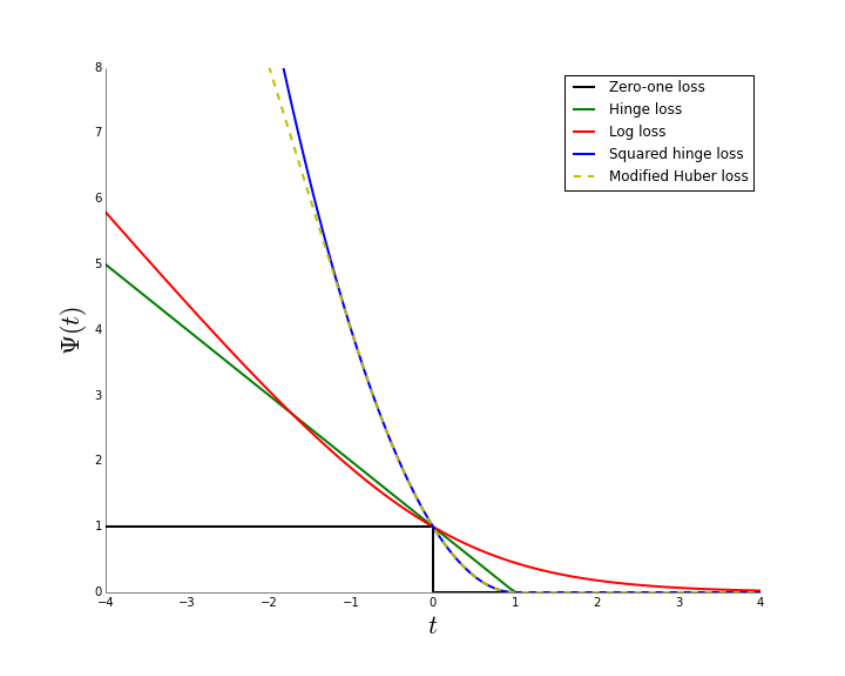
Cost Function and Convexity



Examples of Classic Cost Functions

- Hinge: $\ell'(Y, f(\mathbf{X})) = (1 - Yf(\mathbf{X}))_+$
- With $Y = \pm 1$.
- If $f(\mathbf{X}) = \text{sign}(Y)$ and $|f(\mathbf{X})| > 1$, then $Yf(\mathbf{X}) > 1$.

Cost Function and Convexity

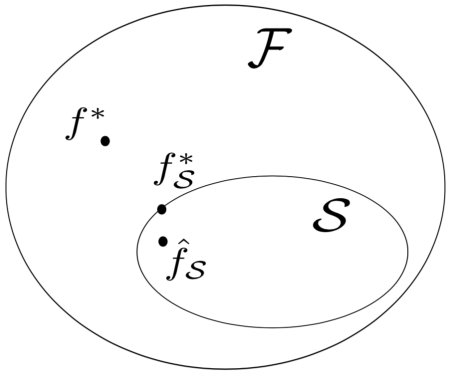


Examples of Classic Cost Functions

- Cross-Entropy: $\ell'(Y, f(\mathbf{X})) = -(Y \ln(f(\mathbf{X})) + (1 - Y) \ln(1 - f(\mathbf{X})))$
- With $Y \in \{0, 1\}$, we desire $f(\mathbf{X}) = Y$.
- $$\ell'(Y, f(\mathbf{X})) = \begin{cases} -\ln(1 - f(\mathbf{X})) & \text{if } Y = 0 \\ -\ln(f(\mathbf{X})) & \text{if } Y = 1 \end{cases}$$

Complexity and Models

- $\mathcal{F} = \{f$:measurable functions $\mathcal{X} \to \mathcal{Y}\}$
- Optimal solution $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f)$
- Class of functions $\mathcal{S} \subset \mathcal{F}$ used as models
- Ideal target in $\mathcal{S}$: $f_{\mathcal{S}}^* = \arg\min_{f \in \mathcal{S}} \mathcal{R}(f)$
- Estimate obtained in $\mathcal{S}$: obtained $f_{\mathcal{S}}$ after training



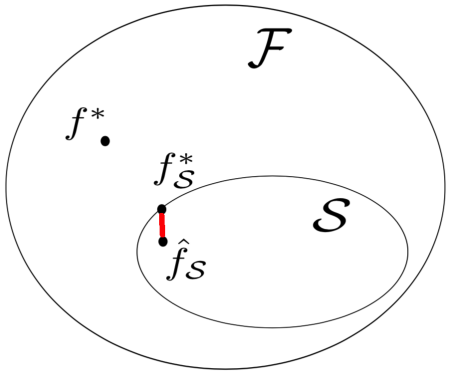
Approximation Error and General Error

$$\mathcal{R}(\hat{f}_{\mathcal{S}}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f_{\mathcal{S}}^*) - \mathcal{R}(f^*)}_{\text{approximation error}} + \underbrace{\mathcal{R}(\hat{f}_{\mathcal{S}}) - \mathcal{R}(f_{\mathcal{S}}^*)}_{\text{estimation error}}$$

- Approximation error can be large if the model $\mathcal{S}$ is not adapted
- Estimation error can be large if the model is complex

Complexity and Models

- $\mathcal{F} = \{f$:measurable functions $\mathcal{X} \to \mathcal{Y}\}$
- Optimal solution $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f)$
- Class of functions $\mathcal{S} \subset \mathcal{F}$ used as models
- Ideal target in $\mathcal{S}$: $f_{\mathcal{S}}^* = \arg\min_{f \in \mathcal{S}} \mathcal{R}(f)$
- Estimate obtained in $\mathcal{S}$: obtained $f_{\mathcal{S}}$ after training



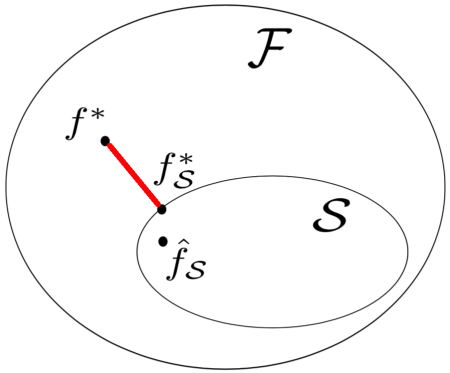
Approximation Error and General Error

$$\mathcal{R}(\hat{f}_{\mathcal{S}}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f_{\mathcal{S}}^*) - \mathcal{R}(f^*)}_{\text{approximation error}} + \underbrace{\mathcal{R}(\hat{f}_{\mathcal{S}}) - \mathcal{R}(f_{\mathcal{S}}^*)}_{\text{estimation error}}$$

- Approximation error can be large if the model $\mathcal{S}$ is not adapted
- Estimation error can be large if the model is complex

Complexity and Models

- $\mathcal{F} = \{f$:measurable functions $\mathcal{X} \to \mathcal{Y}\}$
- Optimal solution $f^* = \arg\min_{f \in \mathcal{F}} \mathcal{R}(f)$
- Class of functions $\mathcal{S} \subset \mathcal{F}$ used as models
- Ideal target in $\mathcal{S}$: $f_{\mathcal{S}}^* = \arg\min_{f \in \mathcal{S}} \mathcal{R}(f)$
- Estimate obtained in $\mathcal{S}$: obtained $f_{\mathcal{S}}$ after training

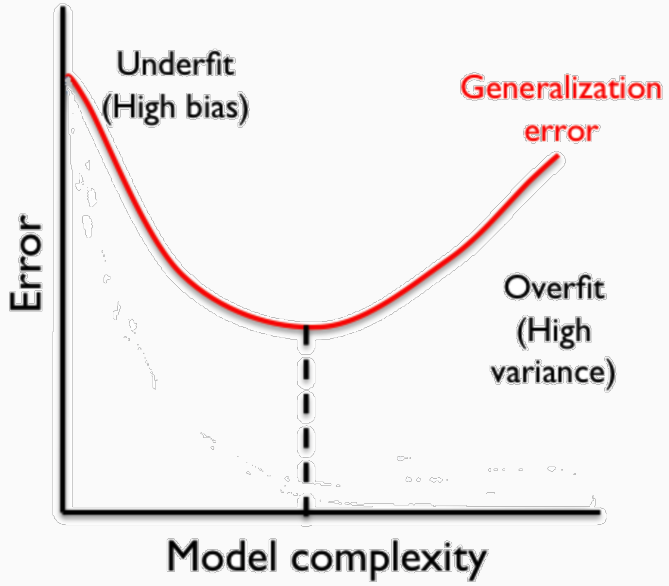


Approximation Error and General Error

$$\mathcal{R}(\hat{f}_{\mathcal{S}}) - \mathcal{R}(f^*) = \underbrace{\mathcal{R}(f_{\mathcal{S}}^*) - \mathcal{R}(f^*)}_{\text{approximation error}} + \underbrace{\mathcal{R}(\hat{f}_{\mathcal{S}}) - \mathcal{R}(f_{\mathcal{S}}^*)}_{\text{estimation error}}$$

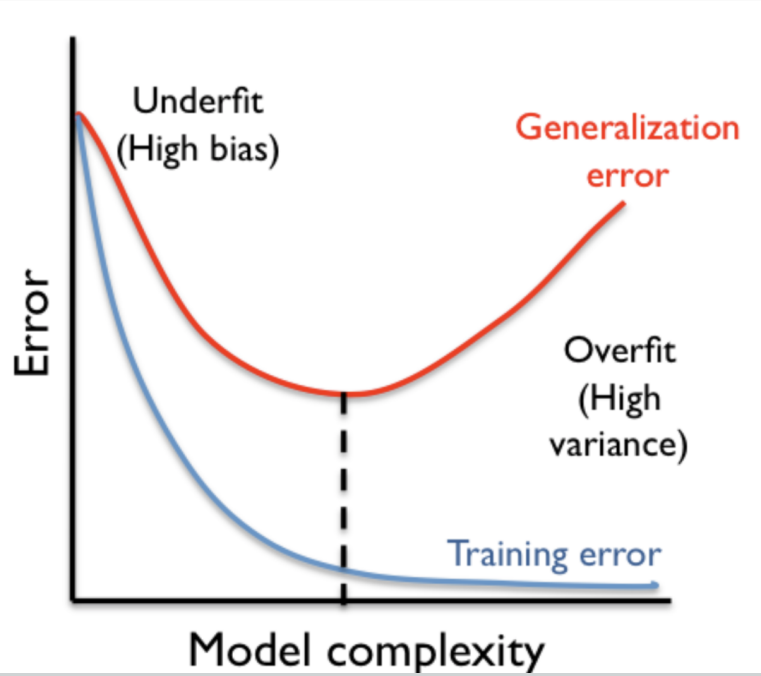
- Approximation error can be large if the model $\mathcal{S}$ is not adapted
- Estimation error can be large if the model is complex

Overfitting and Underfitting



- Depending on the complexity of the model (for instance, training time), different behaviors are observed
- Less complex models are learned easily, but the approximation error may be high (underfitting)
- Highly complex models may have the correct target, but a large estimation error (overfitting)

Overfitting: Problem

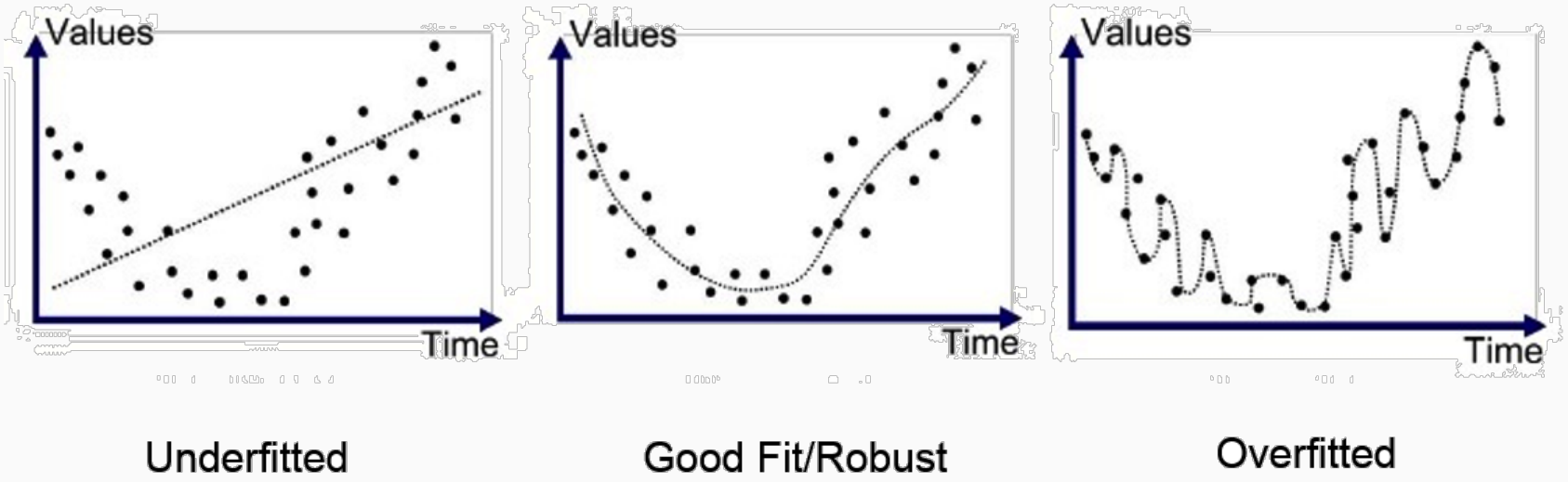


Errors and Risks

- The empirical risk (error on the training set) decreases with increased model complexity
- The true risk (error on new observations) is very different. We have a generalization problem!

- Overfitting: the learned parameters are too specific to the training set
- A criterion other than the training set error must be used

Overfitting: Complexity



Complexity

- If the model is too simple, it no longer fits the data
- If the model is too complex, the model learns all irregularities of dataset $\mathcal{D}_n$
- Example: if the model is the curve in the center plus a noise component not considered in the variables, the model on the right learns this noise

Overfitting: Regularization

A solution to combat this generalization problem: regularization

Principle

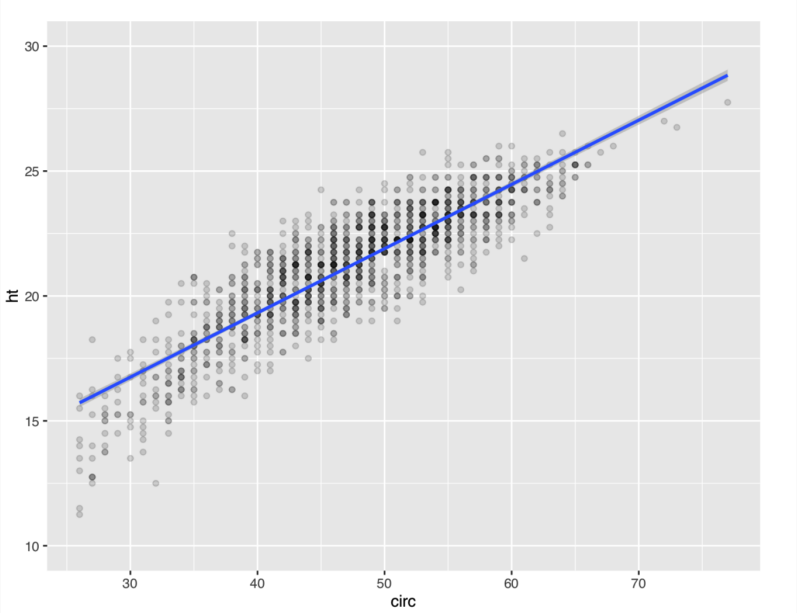
- The empirical risk of an estimator selected from a given family of functions based on data is biased
- By adding a penalty relative to the complexity of the model f_θ, we can reduce it and decrease overfitting:

$$\mathcal{R}_n(f_\theta) \to \mathcal{R}_n(f_\theta) + pen(\theta)$$

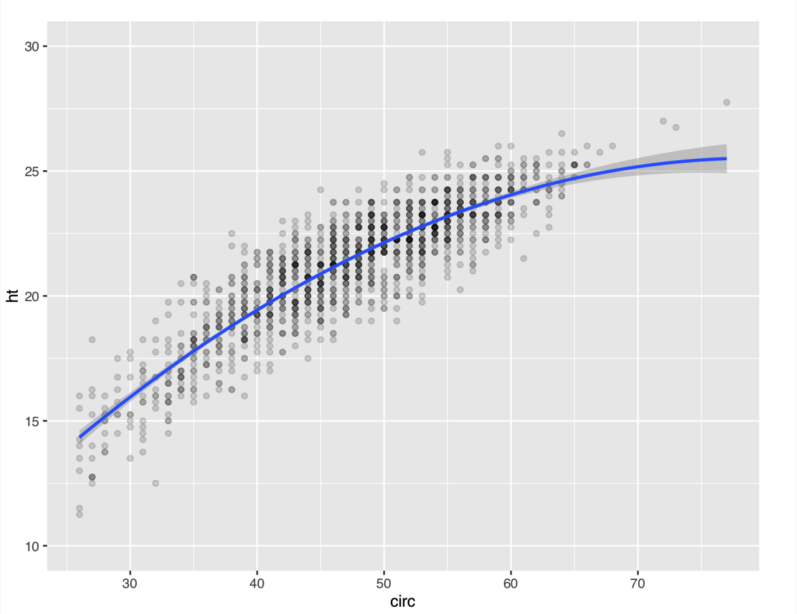
- Then, the risk becomes:

$$\arg \min_{f_\theta, \theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_\theta(\mathbf{X}_i)) + pen(\theta)$$

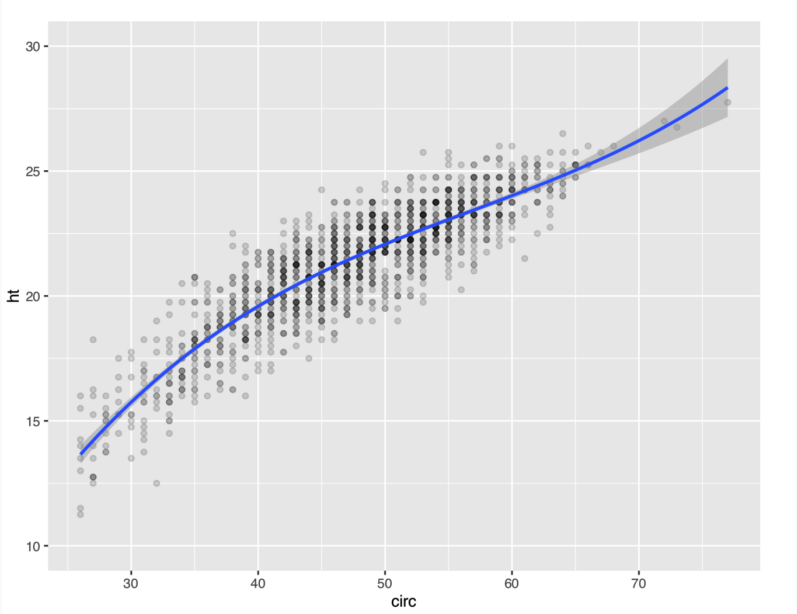
Overfitting: Regularization



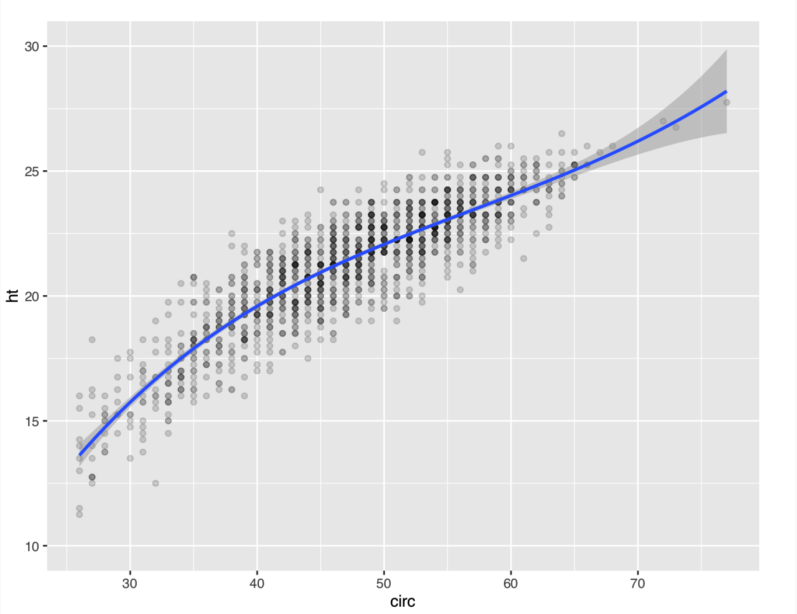
Overfitting: Regularization



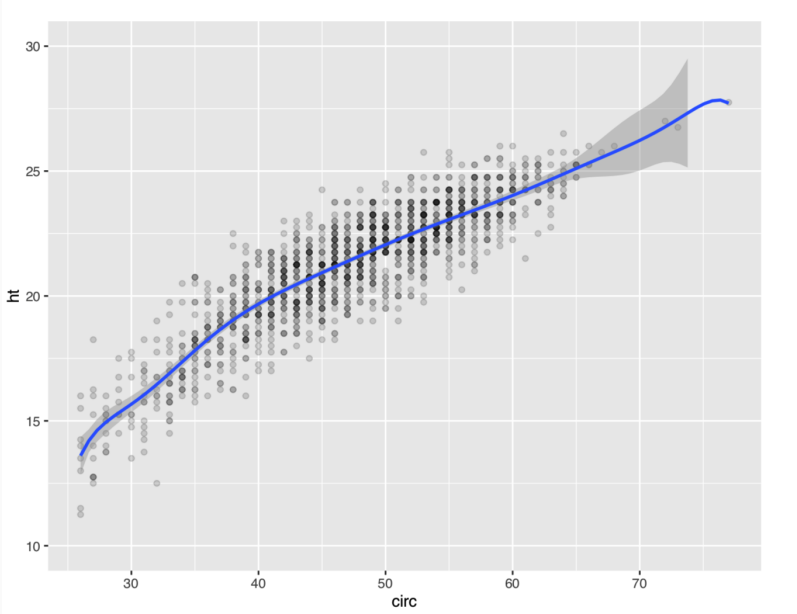
Overfitting: Regularization



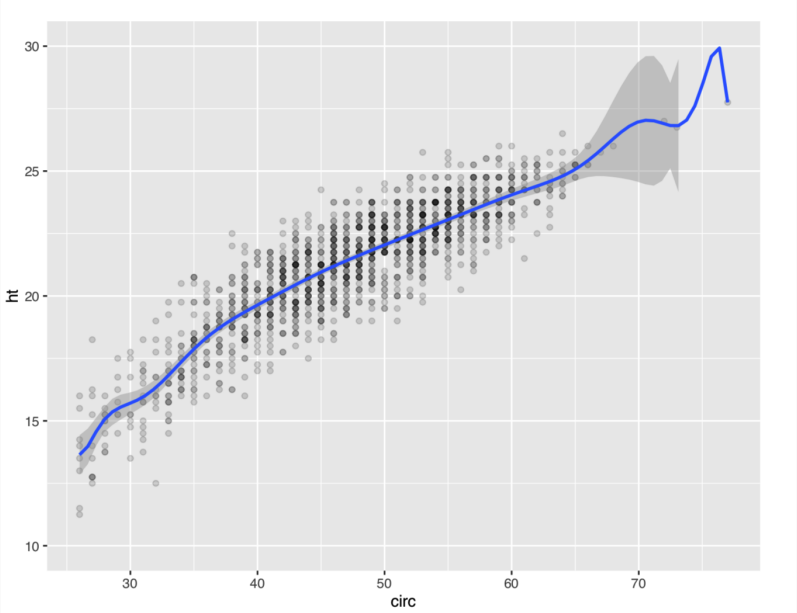
Overfitting: Regularization



Overfitting: Regularization



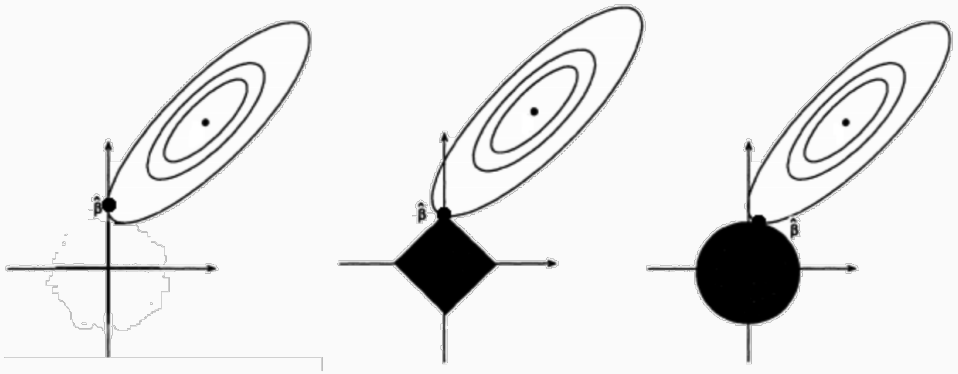
Overfitting: Regularization



Regularization

- Parsimony principle (Occam's razor): the simpler and effective the model, the better.
- Avoid considering too many data particularities.
- Intuition: decrease model norm, number of coefficients, or number of branches (pruning)

Overfitting: Regularization



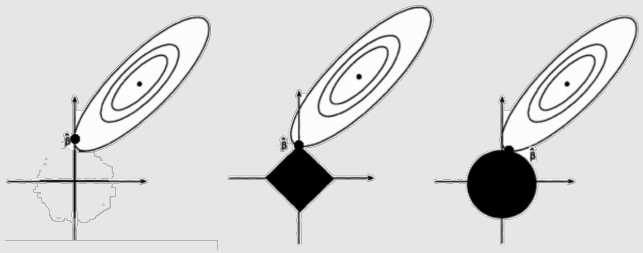
Classical Regularizations

- AIC: $pen(\theta) = \lambda||\theta||_0$ (non-convex, sparse, seldom used)
- Ridge: $pen(\theta) = \lambda||\theta||_2$ (convex, non-sparse)
- Lasso: $pen(\theta) = \lambda||\theta||_1$ (convex, sparse)
- Elastic Net: $pen(\theta) = \lambda_1||\theta||_1 + \lambda_2||\theta||_2$ (convex, sparse)

- Optimization simple if the cost (regularization) is convex
- **Need to specify the** λ, which become new hyperparameters

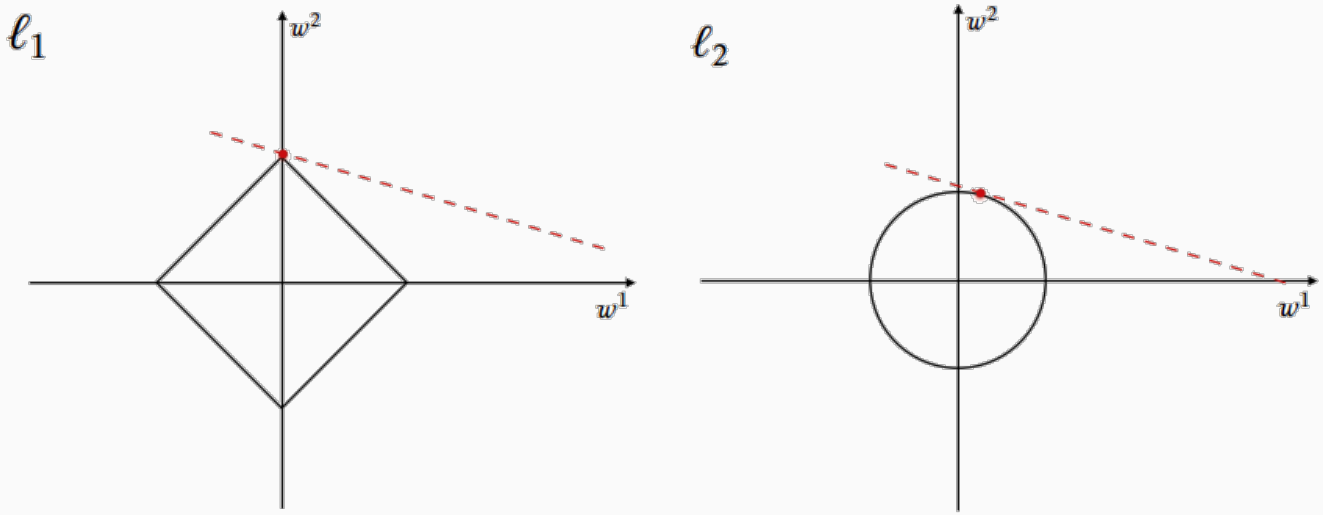
Intuition of Sparsity

Lasso Induces Sparsity

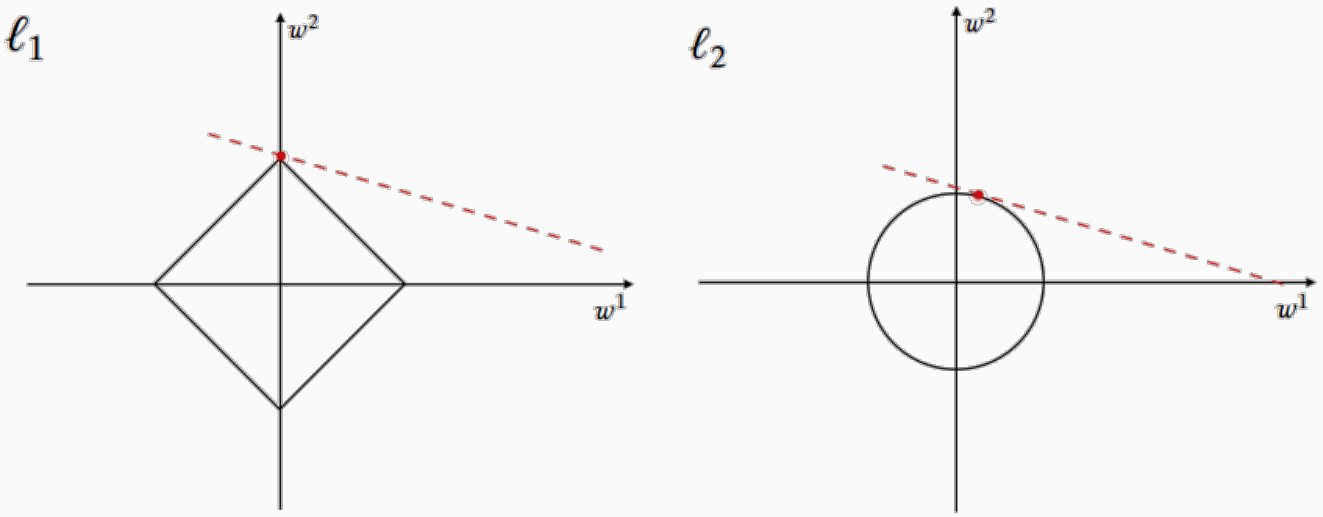


In black, $\mathcal{B}^n = \{x \mid x \in \mathbb{R}^d \text{ and } ||x||_n < 1\}$ for $n = \{0, 1, 2\}$ in $\mathbb{R}^2$

- In high dimensions, most of $\mathcal{B}^1$ concentrates along the axes. This is equivalent to having zero values for the other coordinates.

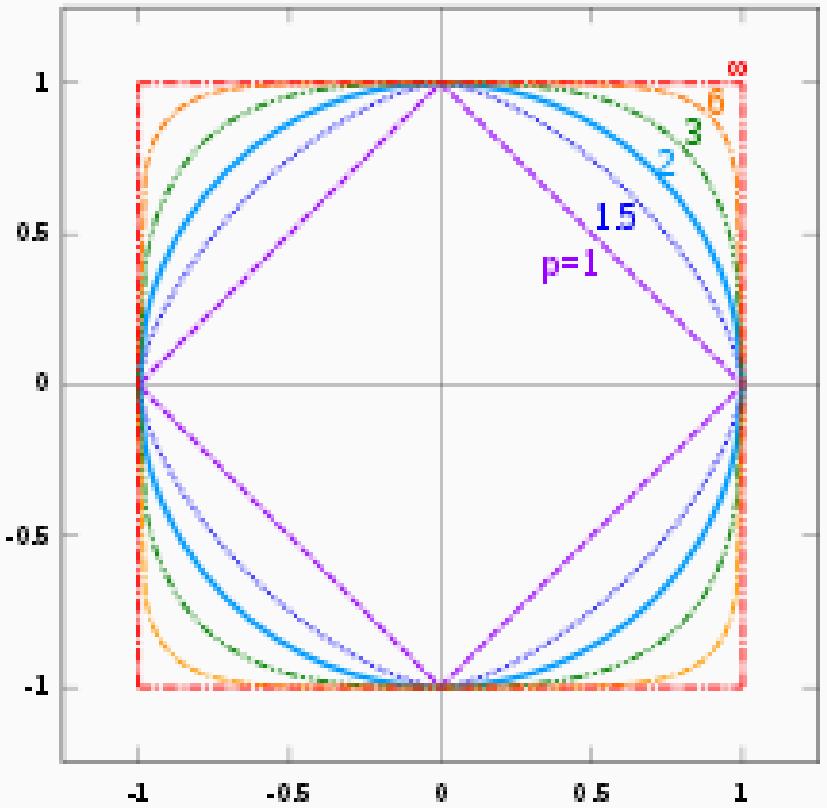


Intuition of Sparsity: Norms

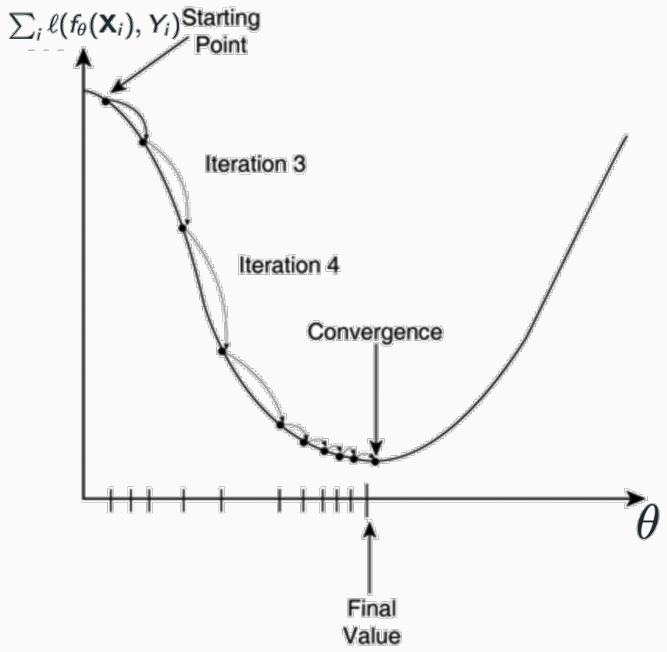


Exercise: We have seen ball 1 and ball 2. What is the shape of the infinity-norm ball?

Intuition of Sparsity: Norms



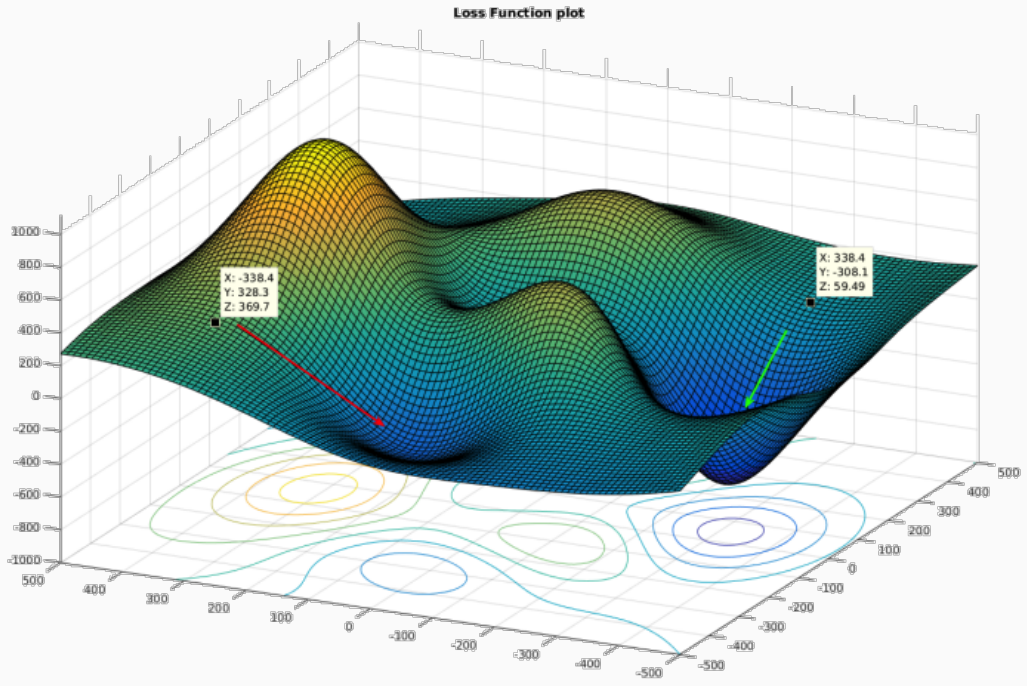
Optimization



Optimization of the Cost Function

- It is used to converge to the minimum value of the cost function on the training dataset
- Best case: fast and accurate
- Often approximations are made to speed up the process

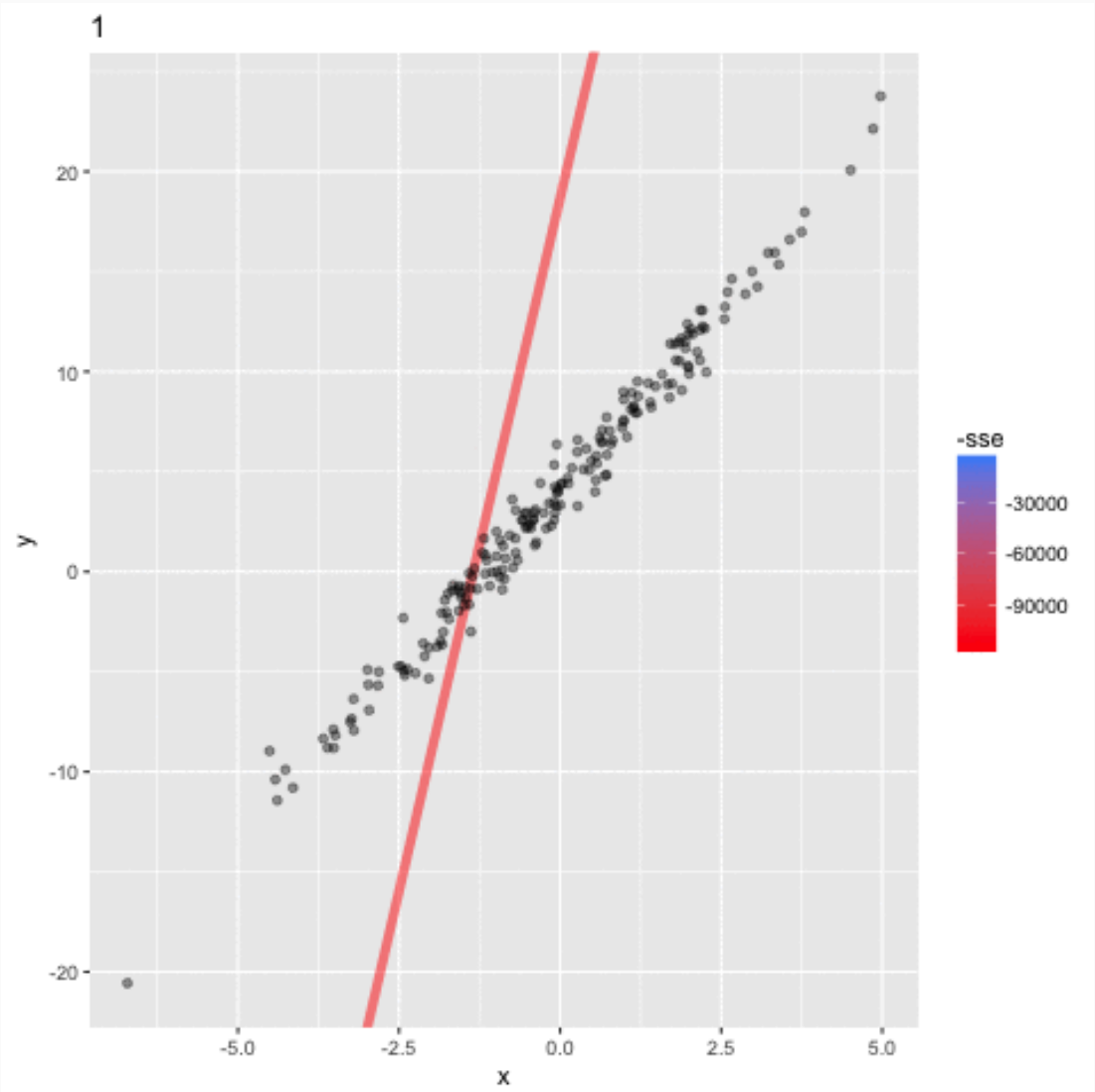
Visualization of the Cost Function



The value of the cost function can be visualized as a surface:

- The parameter values θ vary along the plane, while the value of the function $\ell(\mathcal{D}_n; \theta)$ varies in height.
- Convergence occurs when the parameters are located in a dip in this surface (either a local or global minimum, depending on the model)

Optimization: Visualization



Optimization: Stochastic Gradient Descent

Gradient Descent

After each evaluation of the cost function $\ell(Y_i, f_\theta(\mathbf{X}_i); \theta)$, the gradient of this function is computed to update the parameters θ:

$$\theta \leftarrow \theta - \alpha * \nabla_\theta \ell(Y_i, f_\theta(\mathbf{X}_i); \theta)$$

Example

- Let $f_\theta(\mathbf{X}) = \theta^T \mathbf{X} = \sum_k^d \theta_k X^{(k)}$ and $\ell(Y, f_\theta(\mathbf{X})) = \frac{1}{2}(Y - f_\theta(\mathbf{X}))^2$

$$\nabla_\theta \ell(Y_i, f_\theta(\mathbf{X}_i); \theta) = \begin{pmatrix} \frac{\partial}{\partial \theta_1} \\ \vdots \\ \frac{\partial}{\partial \theta_d} \end{pmatrix} \ell(Y_i, f_\theta(\mathbf{X}_i); \theta) = \begin{pmatrix} X_i^{(1)}(Y - f_\theta(\mathbf{X}_i)) \\ \vdots \\ X_i^{(d)}(Y_i - f_\theta(\mathbf{X}_i)) \end{pmatrix}$$

Optimization: Stochastic Gradient Descent

Gradient Descent

After each evaluation of the cost function $\ell(Y_i, f_\theta(\mathbf{X}_i); \theta)$, the gradient of this function is computed to update the parameters θ:

$$\theta \leftarrow \theta - \alpha * \nabla_\theta \ell(Y_i, f_\theta(\mathbf{X}_i); \theta)$$

Example

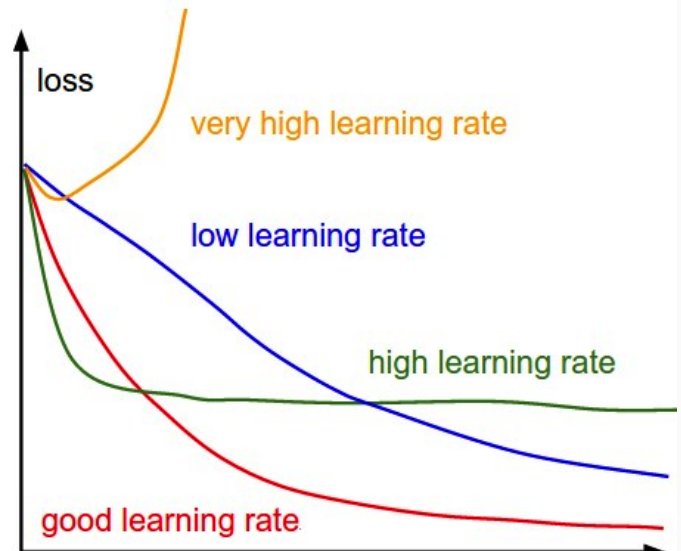
- Let $f_\theta(\mathbf{X}) = \theta^T \mathbf{X} = \sum_k^d \theta_k X^{(k)}$ and $\ell(Y, f_\theta(\mathbf{X})) = \frac{1}{2}(Y - f_\theta(\mathbf{X}))^2$

$$\begin{pmatrix} \theta_1 \\ \vdots \\ \theta_d \end{pmatrix} \leftarrow \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_d \end{pmatrix} - \alpha * \begin{pmatrix} X_i^{(1)}(Y - f_\theta(\mathbf{X}_i)) \\ \vdots \\ X_i^{(d)}(Y_i - f_\theta(\mathbf{X}_i)) \end{pmatrix}$$

Optimization: Importance of the Learning Rate

Learning Rate

- A learning rate (α) that is too small does not advance the learning process
- A learning rate that is too small prolongs the training time
- A learning rate that is too large prevents reaching the minimum (the updates circle around the dip in the surface)
- A learning rate that is too large prevents any progress
- One solution: gradually decrease α over time



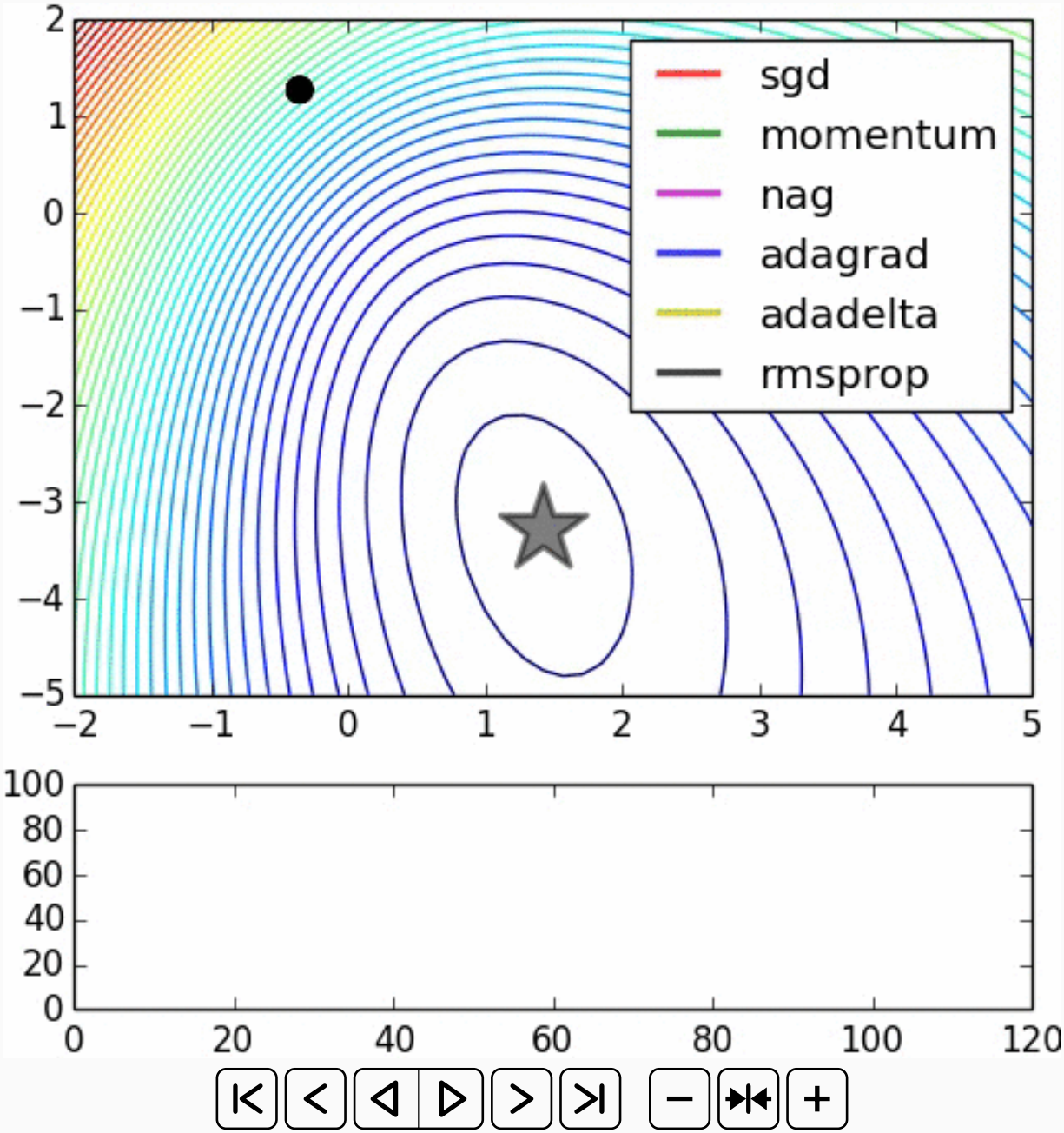
Gradient Descent Algorithms

There are other algorithms based on stochastic gradient descent (SGD) with specific modifications to improve their efficiency:

- Stochastic gradient descent with momentum
- Nesterov Accelerated Gradient (NAG)
- Adaptive Gradient (AdaGrad)
- Adam
- RMSprop

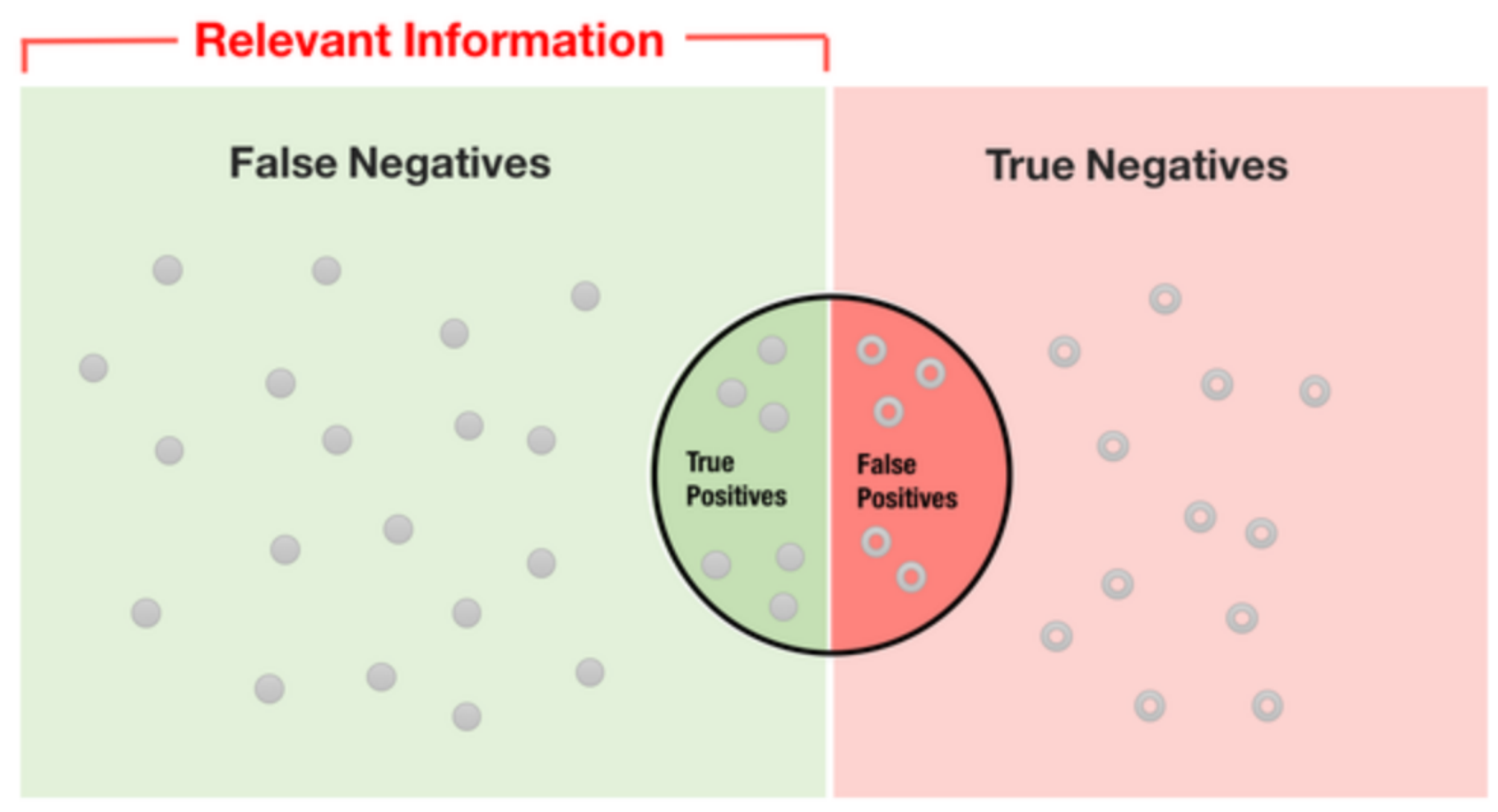
Finally, there are also other more classical descent methods: BFGS, L-BFGS, Quasi-Newton, ...

Gradient Descent Algorithms



Outline : Prediction Evaluation

Prediction Evaluation: True and False Positives

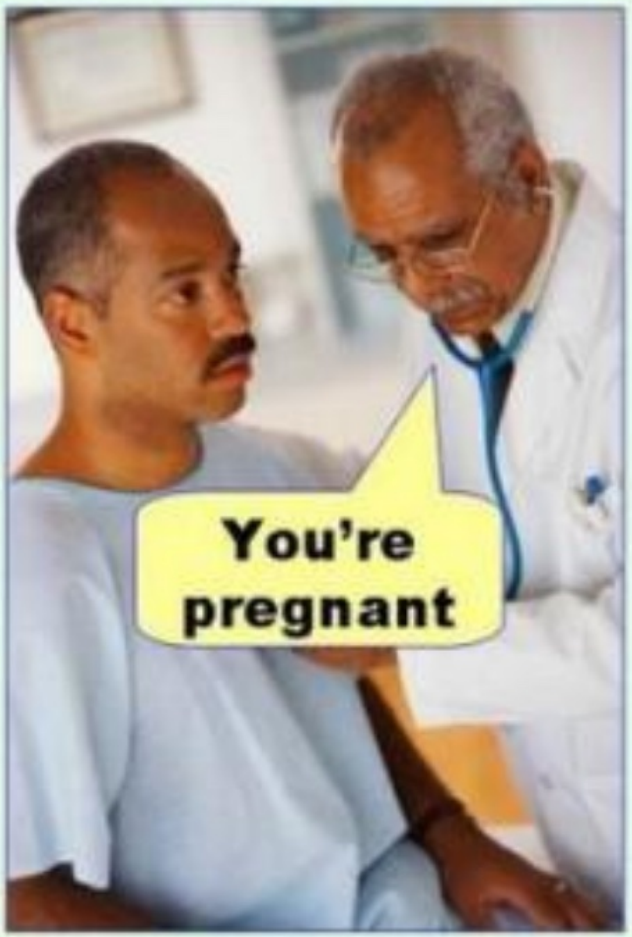


Pay attention

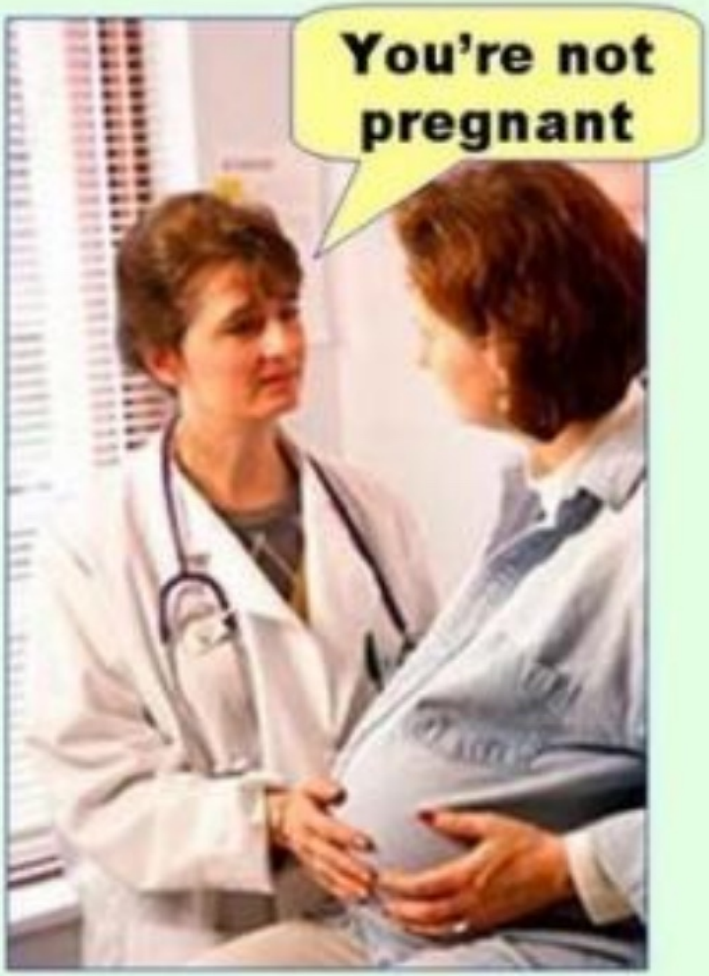
- If the dataset is imbalanced
- The relative importance assigned to different types of errors

Metrics: Types of Errors

Type I error
(false positive)



Type II error
(false negative)



Metrics: Confusion Matrix

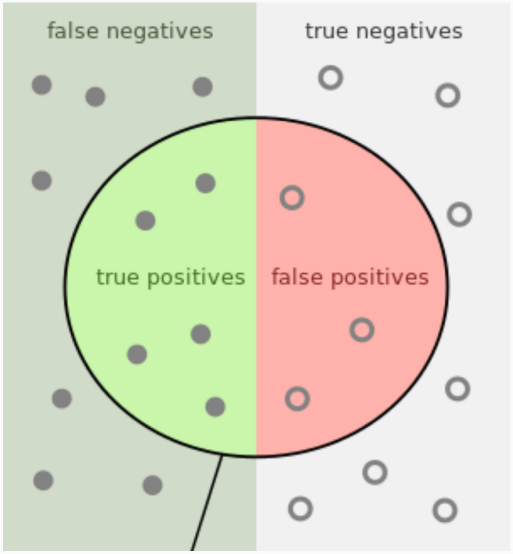
		Model prediction / Test result	
		Positive / Predicted positive	Negative / Predicted Negative
Actual / Ground truth / target / condition / Label	Positive (P)	True Positive (TP), hit	False Negative (FN), type II error, miss, Underestimation
	Negative (N)	False Positive (FP), type I error, false alarm, Overestimation	True negative (TN), correct rejection

Metrics: Recall

Recall: What proportion of the actual positives were classified as positive?

$$\text{Recall} = \frac{TP}{TP + FN}$$

- Have all positives been identified?
- In what case could this metric be the most important?

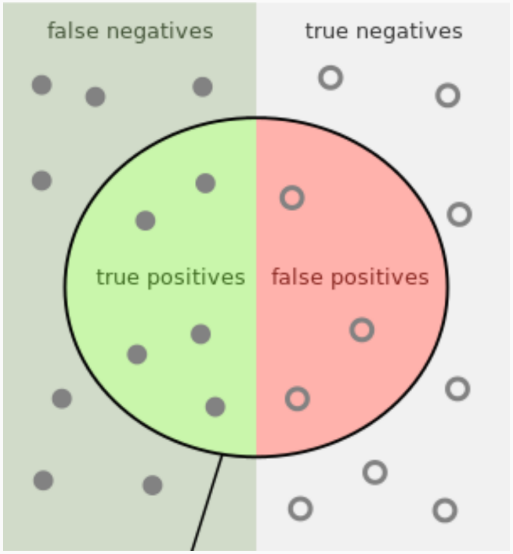


Metrics: Recall

Recall: What proportion of the actual positives were classified as positive?

$$\text{Recall} = \frac{TP}{TP + FN}$$

- Have all positives been identified?
- In what case could this metric be the most important?
- Mass screening for a contagious disease: we do not want to miss any infected individuals

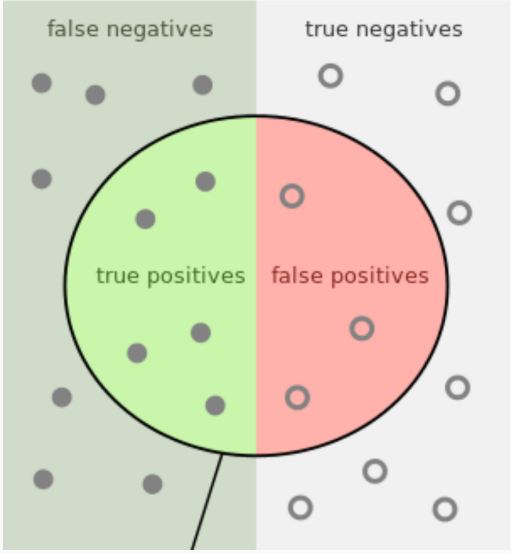


Metrics: Precision

Precision: What proportion of the positive predictions are correct?

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Have all instances been correctly classified?
- In what case could this metric be the most important?

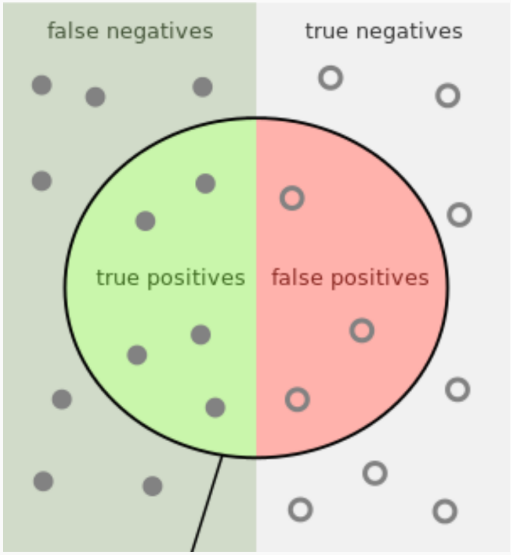


Metrics: Precision

Precision: What proportion of the positive predictions are correct?

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Have all instances been correctly classified?
- In what case could this metric be the most important?
- Precise detection of a deadly disease: we do not want to administer heavy treatment unnecessarily

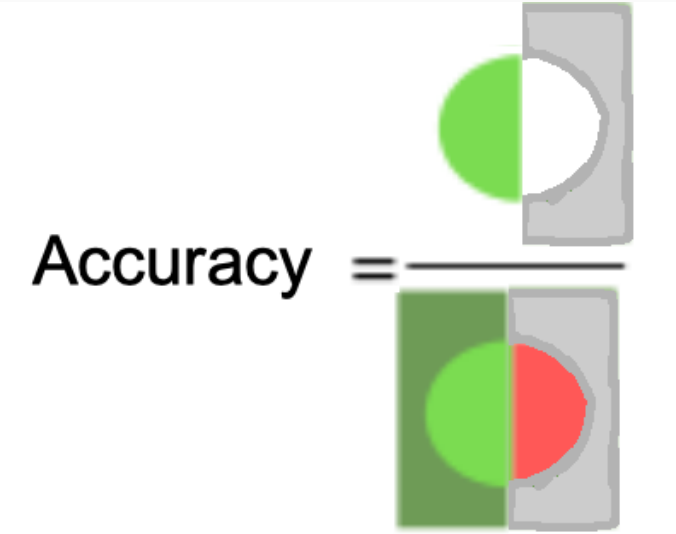


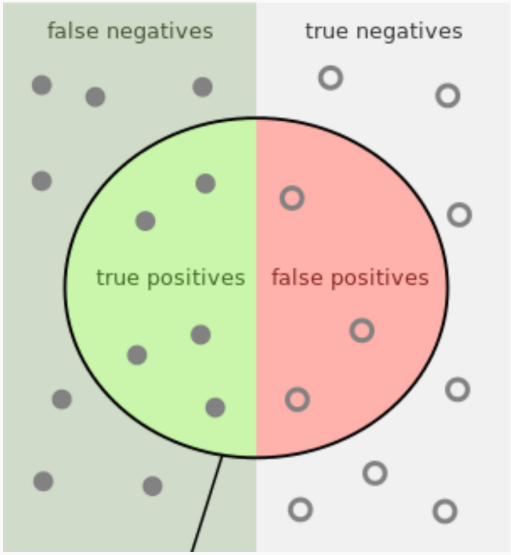
Metrics: Accuracy

Accuracy: What proportion of all predictions are correct?

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

- How well does the model perform overall on the dataset?
- In what case is this metric completely useless?



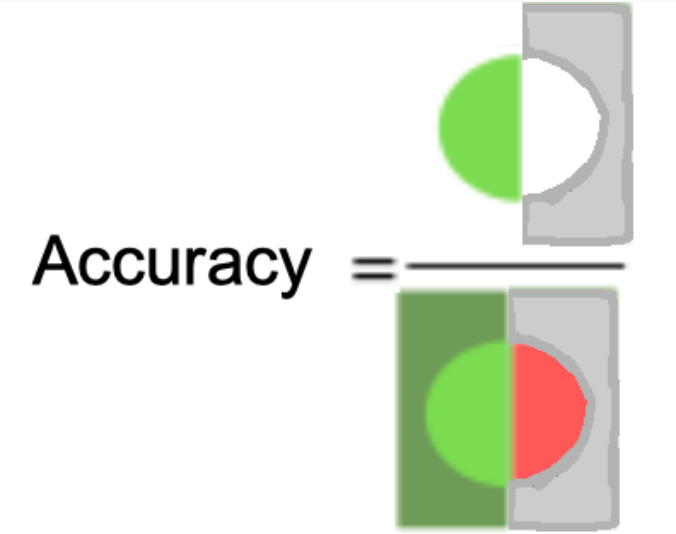


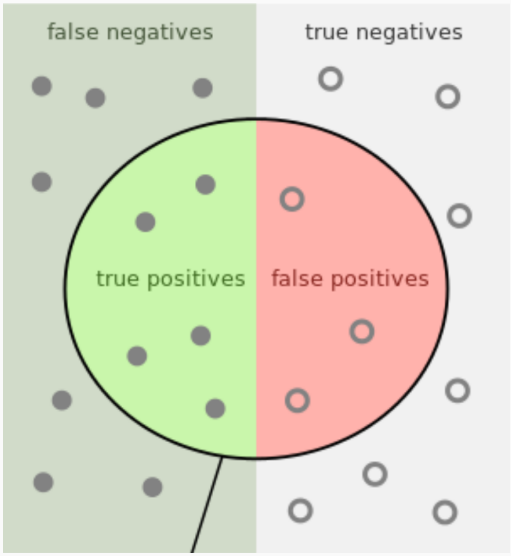
Metrics: Accuracy

Accuracy: What proportion of all predictions are correct?

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

- How well does the model perform overall on the dataset?
- In what case is this metric completely useless?
- Anomaly detection: with 1% positive cases, a system that always predicts negative would achieve 99% accuracy...





Metrics: Exercise I

Consider 286 women: 201 do not experience cancer recurrence after 5 years and 85 do.

Compare the models:

- M1: "all recur"
- M2: "none recur"

TODO

Build confusion matrices and compute accuracy, precision, recall, and F1 score.

Metrics: Exercise II

Considere 286 mujeres: 201 no tienen reincidencia de cáncer después de 5 años y 85 sí tienen. Compare los modelos:

M1: Todas reinciden

M1	+	-
+	85	0
-	201	0

M2: Ninguna reincide

M2	+	-
+	0	85
-	0	201

Accuracy: 85/286= 0.3
Precision: 85/286= 0.3
Recall: 1
F1: 2*0.3/(0.3+1)=0.46

Accuracy: 201/286 = 0.7
Precision: 0/0 = undef
Recall: 0/85=0
F1: undef

Metrics: Cost Matrix

The idea: Some errors are more severe than others

Assign a weight to each type of error

A veces yo se cuales errores son más costosos y cuales aciertos son más valiosos.

		Clase predicha	
Clase real	C(i|j)	clase = +	clase = -
	clase = +	C(+|+)	C(-|+)
	clase = -	C(+|-)	C(-|-)

C(i|j): Costo de clasificar un objeto como clase j dado que es clase i

Metrics: Cost Matrix

The idea: Some errors are more severe than others

Assign a weight to each type of error

A mayor costo peor el modelo.

Matrix Costo	Clase predicha		
	C(i|j)	+	-
Clase real	+	-1	100
	-	1	0

Modelo M1	Clase predicha		
		+	-
Clase real	+	150	40
	-	60	250

Accuracy(M1) = 0.8
C(M1) = -1*150+100*40+1*60+0*250 = 3910

Modelo M2	Clase predicha		
		+	-
Clase real	+	250	45
	-	5	200

Accuracy(M2) = 0.9
C(M2) = -1*250+100*45+1*5+0*200 = 4255

Metrics

Different metrics depending on the objective

- $$\text{Recall} = \frac{TP}{TP + FN}$$
- $$\text{Precision} = \frac{TP}{TP + FP}$$
- $$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$
- $$\text{F1} = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision} = \frac{2TP}{2TP + FP + FN}$$

  More generally: $\frac{1}{\text{F}_\beta} = \frac{1}{1+\beta^2}\left(\beta^2 \frac{1}{Recall} + \frac{1}{Precision}\right)$
- ROC curve, AUC, ...
- top-k accuracy, balanced accuracy

More info: `https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics`

Aggregation of Metrics I

If there is more than one class, the metrics must be aggregated per class:

- **Micro-averaging**: compute the metric for each class and then average
- **Macro-averaging**: create a binary confusion matrix for each class, combine the matrices, and then evaluate

Aggregation of Metrics II

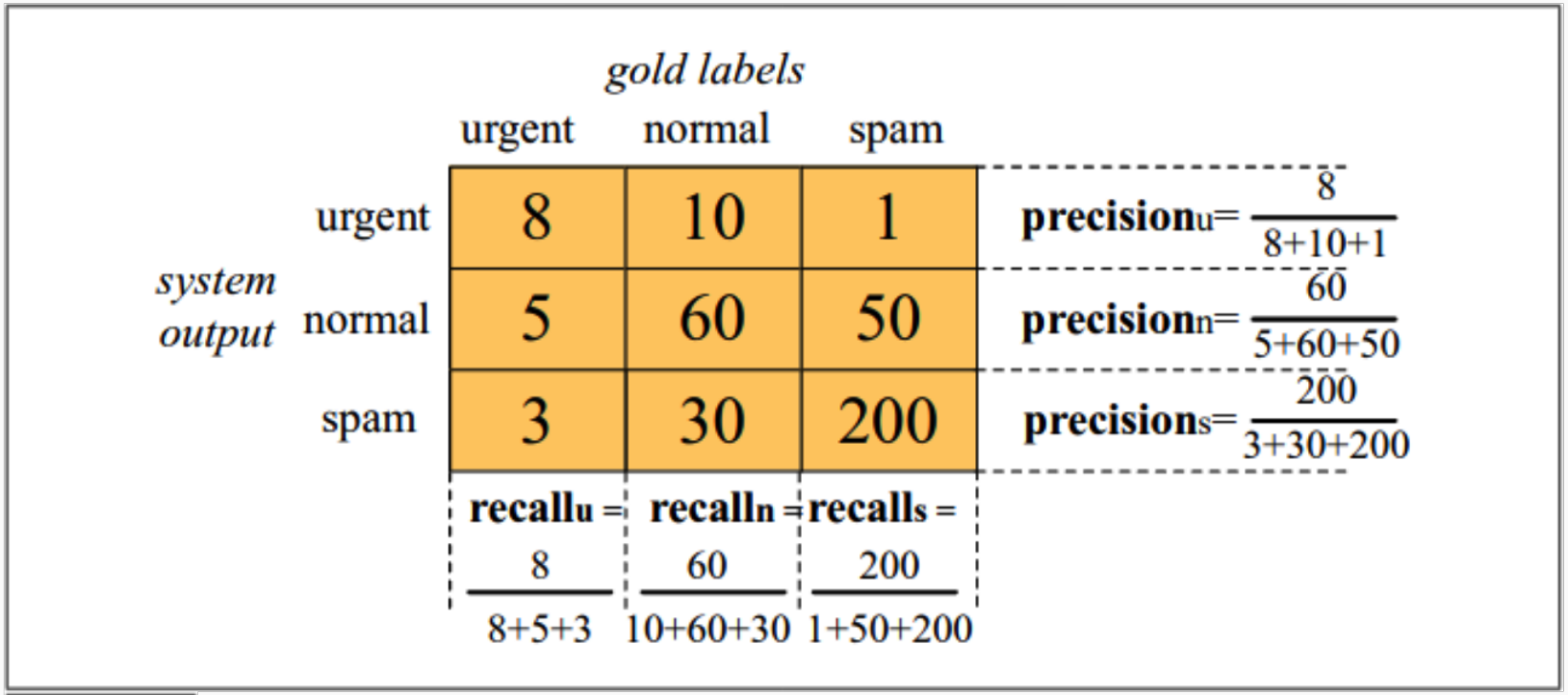
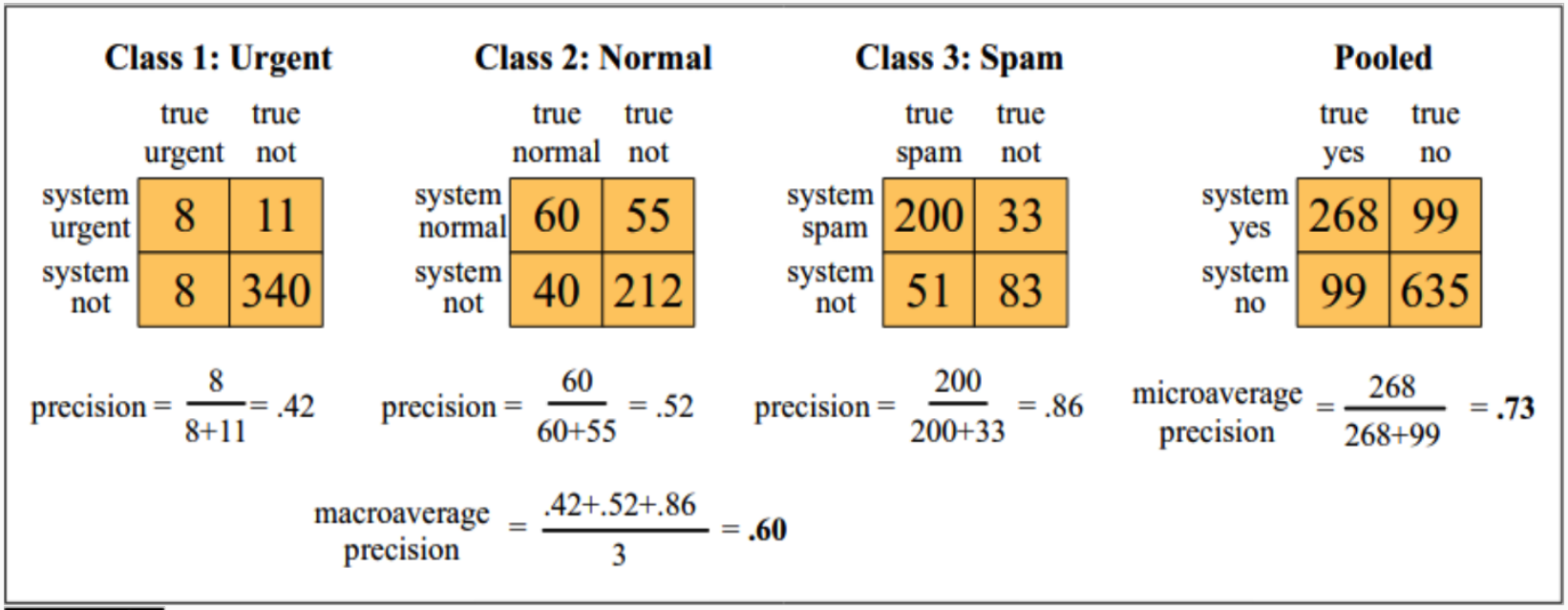


Figure 4.5 Confusion matrix for a three-class categorization task, showing for each pair of classes (c_1, c_2), how many documents from c_1 were (in)correctly assigned to c_2

Aggregation of Metrics II



Class 1: Urgent

	true urgent	true not
system urgent	8	11
system not	8	340

$$\text{precision} = \frac{8}{8+11} = .42$$

Class 2: Normal

	true normal	true not
system normal	60	55
system not	40	212

$$\text{precision} = \frac{60}{60+55} = .52$$

Class 3: Spam

	true spam	true not
system spam	200	33
system not	51	83

$$\text{precision} = \frac{200}{200+33} = .86$$

Pooled

	true yes	true no
system yes	268	99
system no	99	635

$$\text{microaverage precision} = \frac{268}{268+99} = \mathbf{.73}$$

$$\text{macroaverage precision} = \frac{.42+.52+.86}{3} = \mathbf{.60}$$

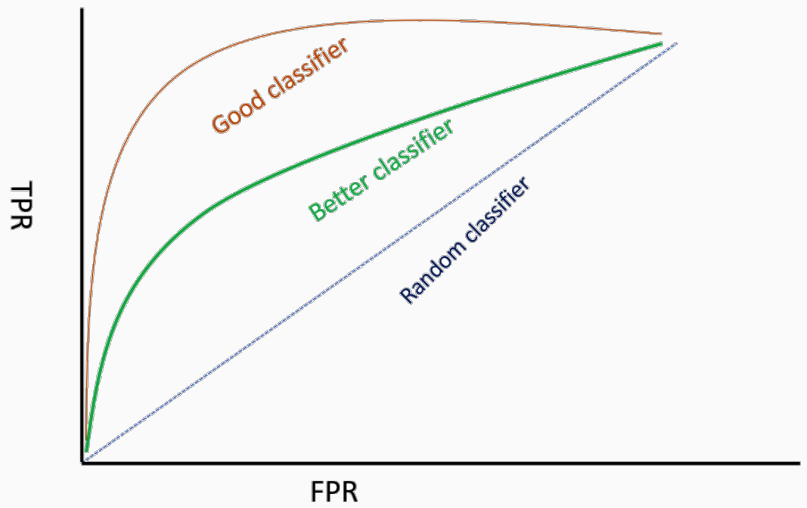
Figure 4.6 Separate contingency tables for the 3 classes from the previous figure, showing the pooled contingency table and the microaveraged and macroaveraged precision.

- Micro-averages are dominated by the most frequent classes.
- Macro-averages may over-represent minority classes.

Receiver Operating Characteristic (ROC)

Definition

It illustrates the performance of a binary classifier system as its discrimination threshold is varied. It is created by plotting the fraction of true positives among the positives (TPR = true positive rate) against the fraction of false positives among the negatives (FPR = false positive rate) for various threshold settings.



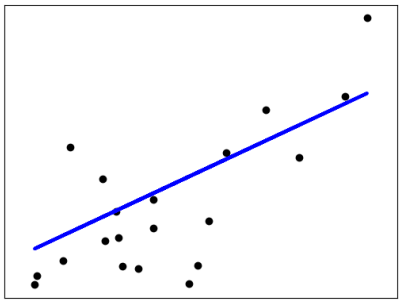
The AUC (area under the curve) is used because a larger area signifies better performance.

Evaluation of Prediction: Distance

Different Functions to Evaluate Regression Performance

- Mean Absolute Error: $\frac{1}{n}\sum_{i=1}^{n}|Y_i - f(\mathbf{X}_i)|$
- Mean Squared Error: $\frac{1}{n}\sum_{i=1}^{n}|Y_i - f(\mathbf{X}_i)|^2$
- Median Absolute Error: $\text{median}(|Y_1 - f(\mathbf{X}_1)|, ..., |Y_n - f(\mathbf{X}_n)|)$
- Coefficient of Determination R^2:

$$1 - \frac{\sum_{i=1}^{n}|Y_i - f(\mathbf{X}_i)|^2}{\sum_{i=1}^{n}|Y_i - \overline{Y}|^2} = \frac{\sum_{i=1}^{n}|f(\mathbf{X}_i) - \overline{Y}|^2}{\sum_{i=1}^{n}|Y_i - \overline{Y}|^2}$$



R^2 represents the proportion of the variance explained by the model

Cross-Validation

Principle

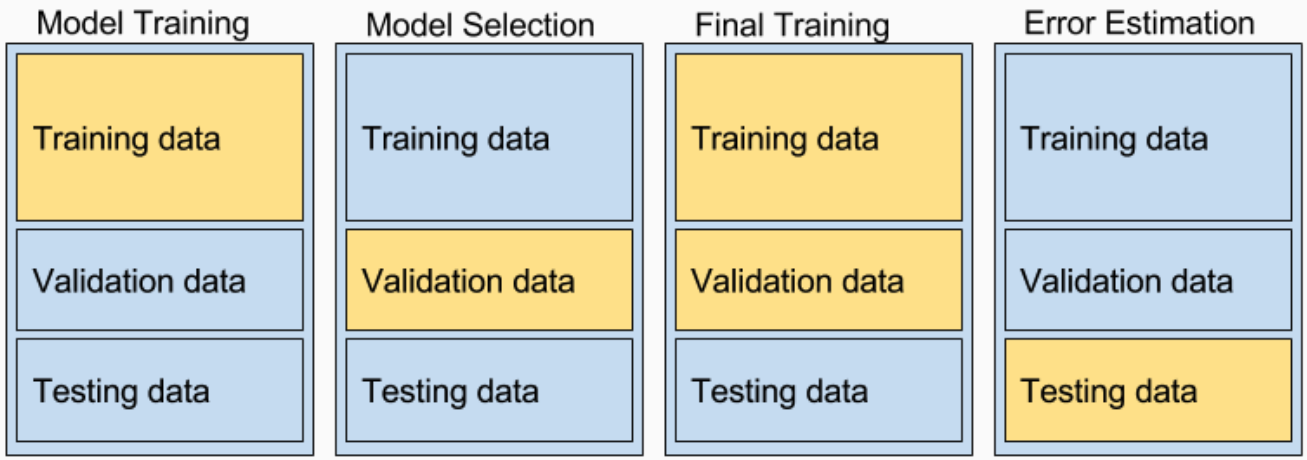
- Split the dataset $\mathcal{D}$ into V subsets $\mathcal{D}_v$ of similar sizes
- For each $v \in [\![1; V]\!]$:
  - Take $\mathcal{D}^{-v} = \mathcal{D} \setminus \mathcal{D}_v$
  - Train $\hat{f}^{-v}$ on $\mathcal{D}^{-v}$
  - Compute $\mathcal{R}^{-v}(\hat{f}^{-v}) = \frac{1}{n_v} \sum_{(\mathbf{x}_i, Y_i) \in \mathcal{D}_v} \ell(Y_i, \hat{f}^{-v}(\mathbf{X}_i))$
- Compute the general risk: $\mathcal{R}^{CV}(\hat{f}) = \frac{1}{V} \sum_{v=1}^{V} \mathcal{R}^{-v}(\hat{f}^{-v})$

Validation and Test Sets (Holdout)

- Cross-validation requires V training sessions. Is there a less costly method?

Splitting the dataset $\mathcal{D}$ into 3 subsets

- The training set is used to train the classifier
- The validation set provides an estimate of the trained model's generalization ability. It can be used to stop training
- The test set is an independent set used to evaluate the classifier's performance; it does not interact with the training process



70/10/20 or 70/15/15 are good proportions for train/val/test

Partition Size

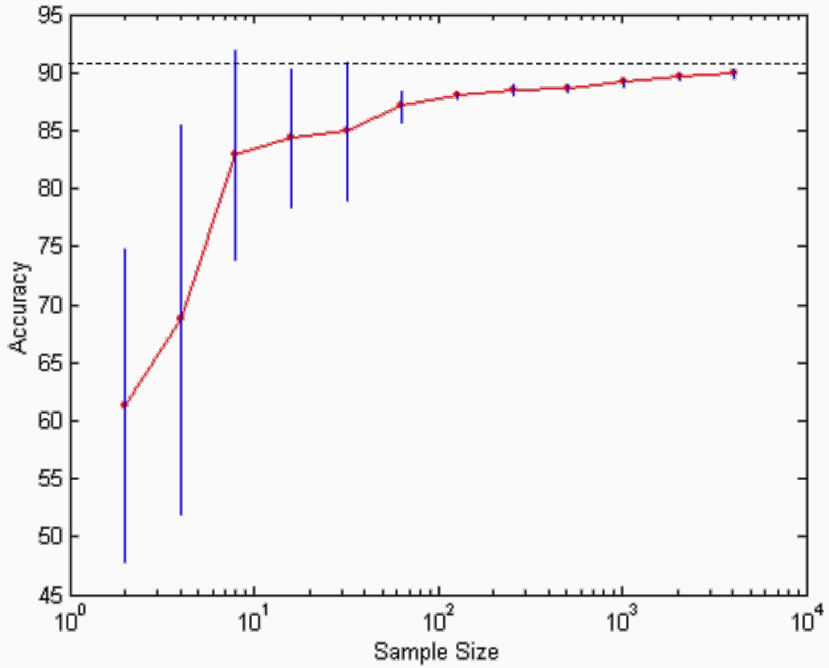


Figure 1: Performance of a model as a function of the amount of data seen

- The evaluation can vary greatly depending on the chosen splits.
- A very small training set $\Rightarrow$ biased model.
- A very small test set $\Rightarrow$ unreliable accuracy.

Questions?

References i