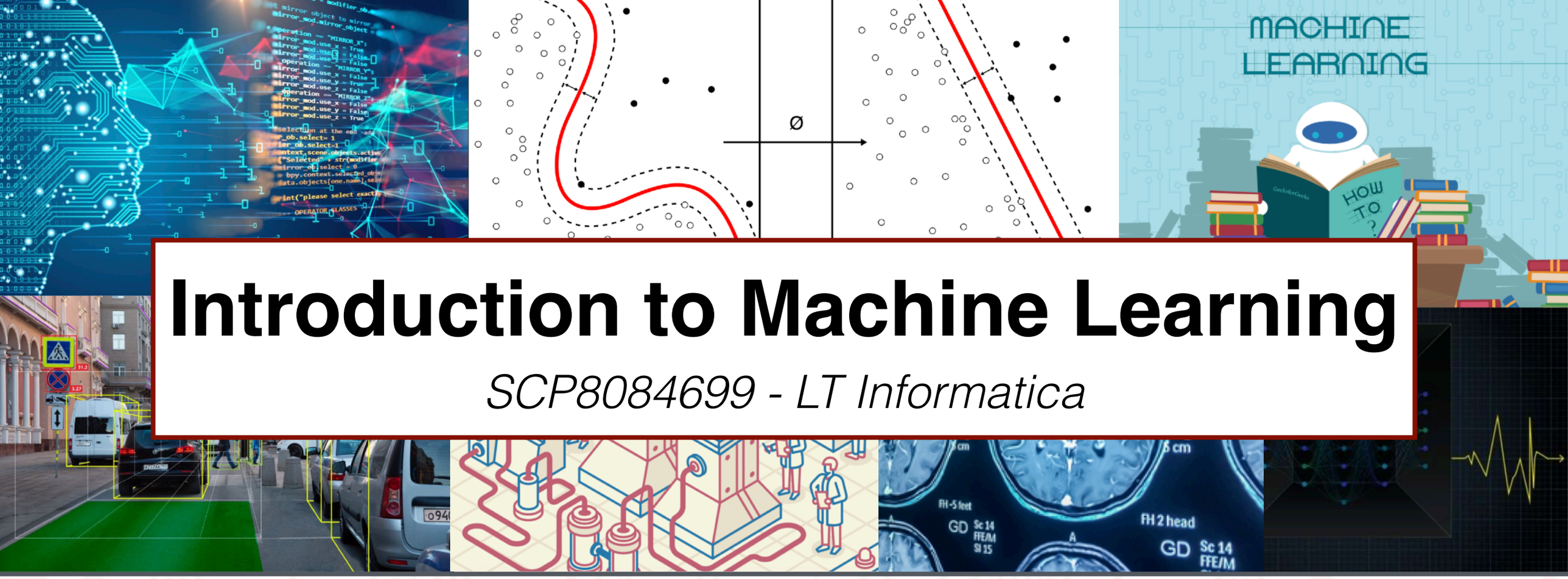


Introduction to Machine Learning

SCP8084699 - LT Informatica

Evaluation, Learning Curves & Babysitting

Prof. Lamberto Ballan

University of Padova, A.Y. 2022/23 April 4, 2023

DIPARTIMENTO MATEMATICA
Department of Mathematics “Tullio Levi-Civita”

What we will learn today?

- A bit more on model selection and evaluation…
  - … and a few advice for applying machine learning
- Everything you need to know about metrics
- Babysitting machine learning models

Model Selection and Evaluation

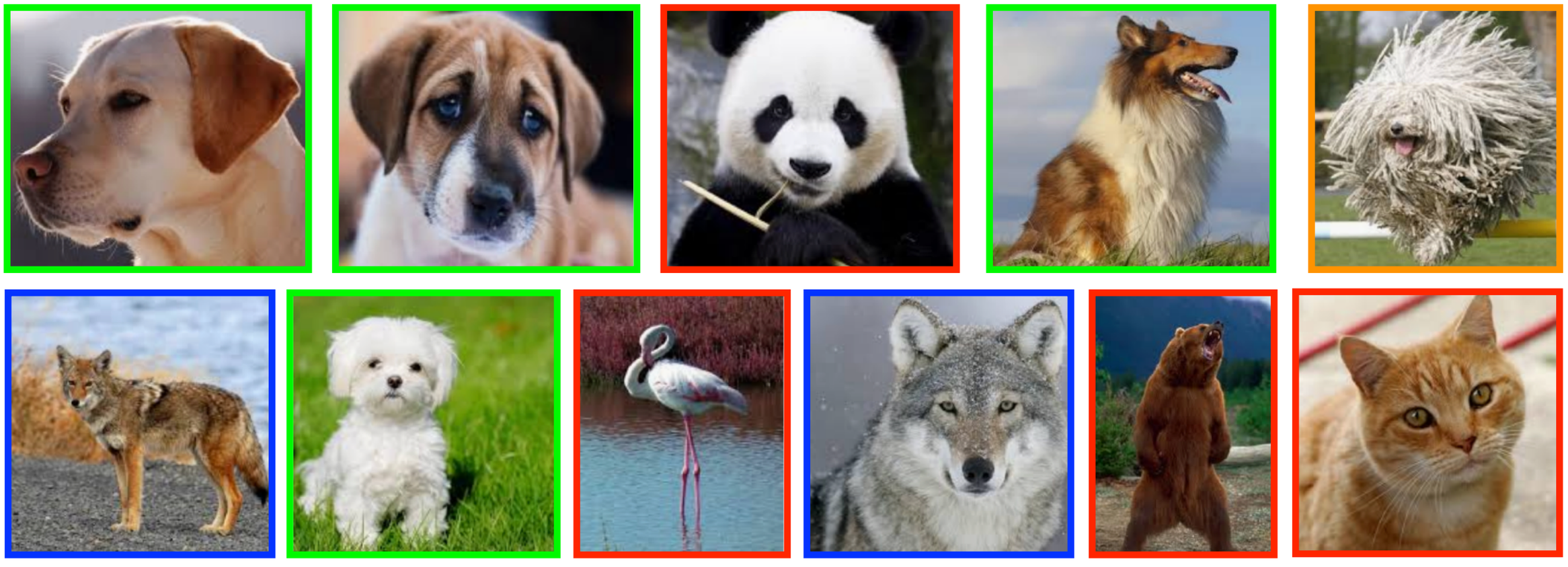
- **Hold-out**: we keep a subset of v samples from the training set (the validation set) to evaluate our model
  - A classifier/regressor is trained on m-v samples
  - Parameters are optimized on the *training*-*validation* sets: then you should evaluate performances on the *test* set
    - Size (cardinality) of training+validation sets should be greater than test set, e.g. 70%,15%,15%
- ***k*-fold cross validation**: iterate on *k* disjoint subsets
- Given a task, pick the "right" evaluation metric

Model Selection and Evaluation

- ***k*-fold cross validation**: an example (*5*-fold)

Metrics

- How to evaluate performances?
  - We will look at different metrics for classification tasks
  - Let's start with a simple example: dog vs no-dog



TP: True Positive **FP:** False Positive **TN:** True Negative **FN:** False Negative

Metrics

- How to evaluate performances?
  - **Accuracy** $= \frac{TP + TN}{P + N} = \frac{\textit{all correct}}{\textit{all instances}} = \frac{4 + 4}{11} = 0.73$

TP: True Positive **FP:** False Positive **TN:** True Negative **FN:** False Negative

Metrics

- How to evaluate performances?
  - **Confusion Matrix**:

		Actual Class (groundtruth)	
		dog	no-dog
Predicted	dog	**TP:** True Positive	**FP:** False Positive *Type I error*
	no-dog	**FN:** False Negative *Type II error*	**TN:** True Negative
		P	**N**

Metrics

- How to evaluate performances?
  - **Precision** vs **Recall**
  - **Precision**: the fraction of retrieved instances that are relevant

$$\text{Prec} = \frac{\text{TP}}{\text{TP + FP}} = \frac{\text{TP}}{\textit{all predicted}}$$

  - **Recall**: the fraction of relevant instances that are retrieved

$$\text{Rec} = \frac{\text{TP}}{\text{TP + FN}} = \frac{\text{TP}}{\textit{all groundtruth instances}}$$

  - **F1-score**: harmonic mean of Prec and Rec

$$\text{F1} = 2 \cdot \frac{\text{Prec} \cdot \text{Rec}}{\text{Prec + Rec}}$$

Metrics

- How to evaluate performances?
  - **Precision** vs **Recall**
  - **Precision**: the fraction of retrieved instances that are relevant

$$\text{Prec} = \frac{\text{TP}}{\text{TP + FP}} = \frac{\text{TP}}{\textit{all predicted}}$$

  - **Recall**: the fraction of relevant instances that are retrieved

$$\text{sensitivity} = \text{TPR} = \text{Rec} = \frac{\text{TP}}{\text{TP + FN}} = \frac{\text{TP}}{\textit{all groundtruth instances}}$$

  - Others:

$$1 - \text{specificity} = \text{FPR} = \frac{\text{FP}}{\text{FP + TN}} = 1 - \frac{\text{TN}}{\text{FP + TN}}$$

Metrics

- Why do we need different metrics?

$$\text{Prec} = \frac{\text{TP}}{\text{TP} + \text{FP}} = 4/6 = 0.67 \qquad \text{Rec} = \frac{\text{TP}}{\text{TP} + \text{FN}} = 4/5 = 0.8$$

TP: True Positive **FP:** False Positive **TN:** True Negative **FN:** False Negative

Metrics

- Let's take a look at the results of a different model:

$$\text{Prec} = \frac{\text{TP}}{\text{TP} + \text{FP}} = 2/2 = 1 \qquad \text{Rec} = \frac{\text{TP}}{\text{TP} + \text{FN}} = 2/5 = 0.4$$

TP: True Positive **FP:** False Positive **TN:** True Negative **FN:** False Negative

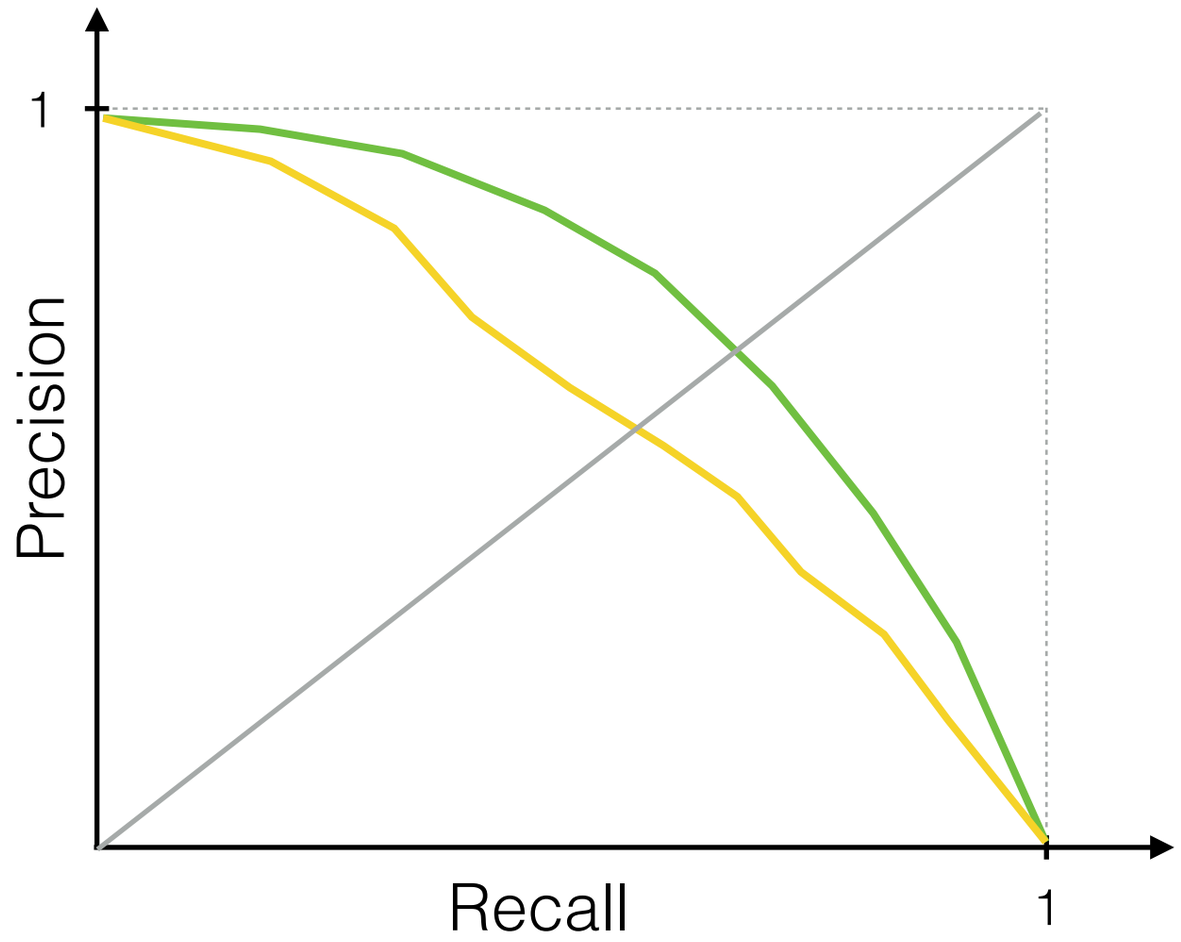
Metrics

- You need a “right” tradeoff between Prec and Recall
  - This depends on the application/task you are addressing
    - E.g. in a video-surveillance application you might prefer high recall (to not miss any anomalous event)
    - … but you need balance! If you get too many alarms the human operator will not keep attention to the alarms



Metrics

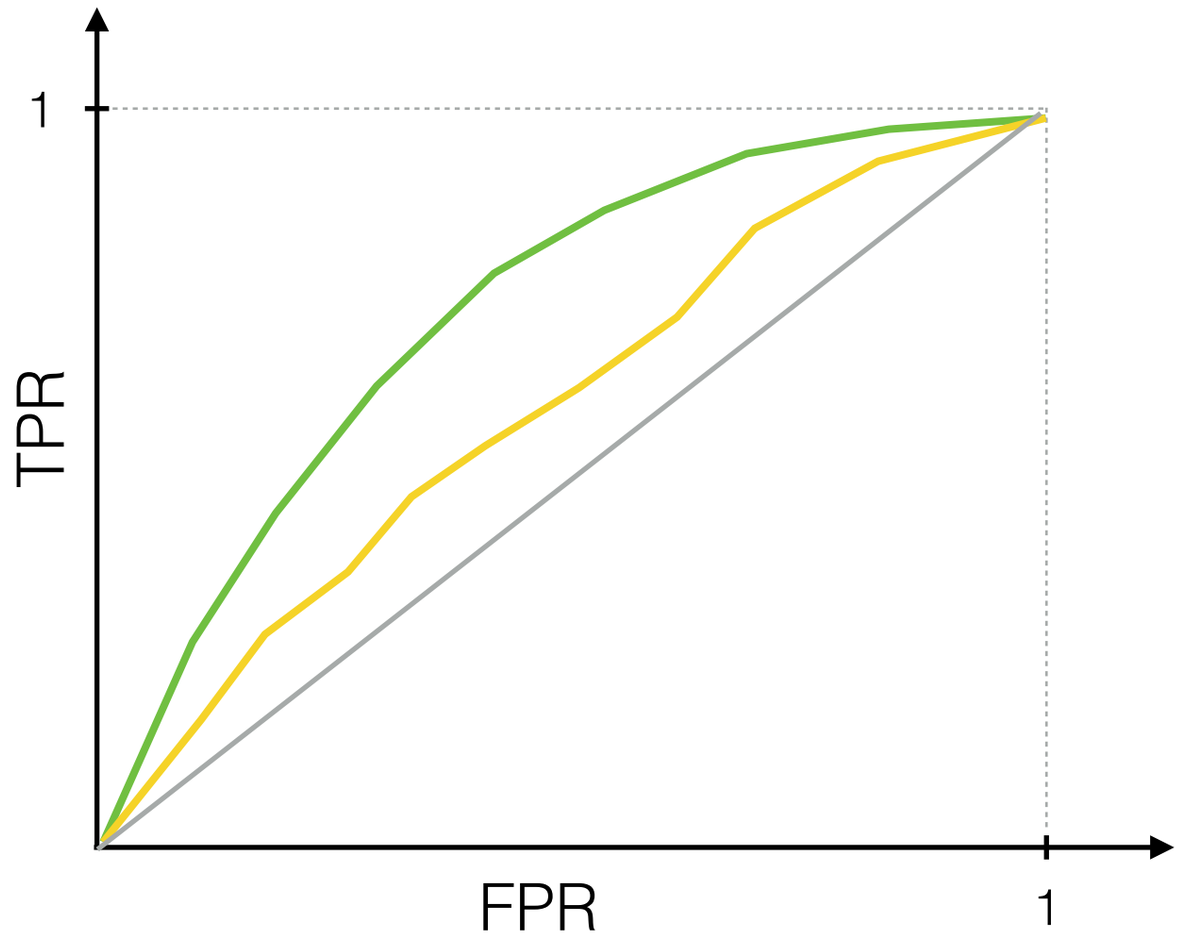
- How to evaluate performances?
  - **Precision-Recall curves**



PR curves are obtained by computing precision and recall figures as a function of hyperparameters (such as a threshold)

Metrics

- How to evaluate performances?
  - **ROC curves** (similar to PR curves but w.r.t. FPR and TPR)

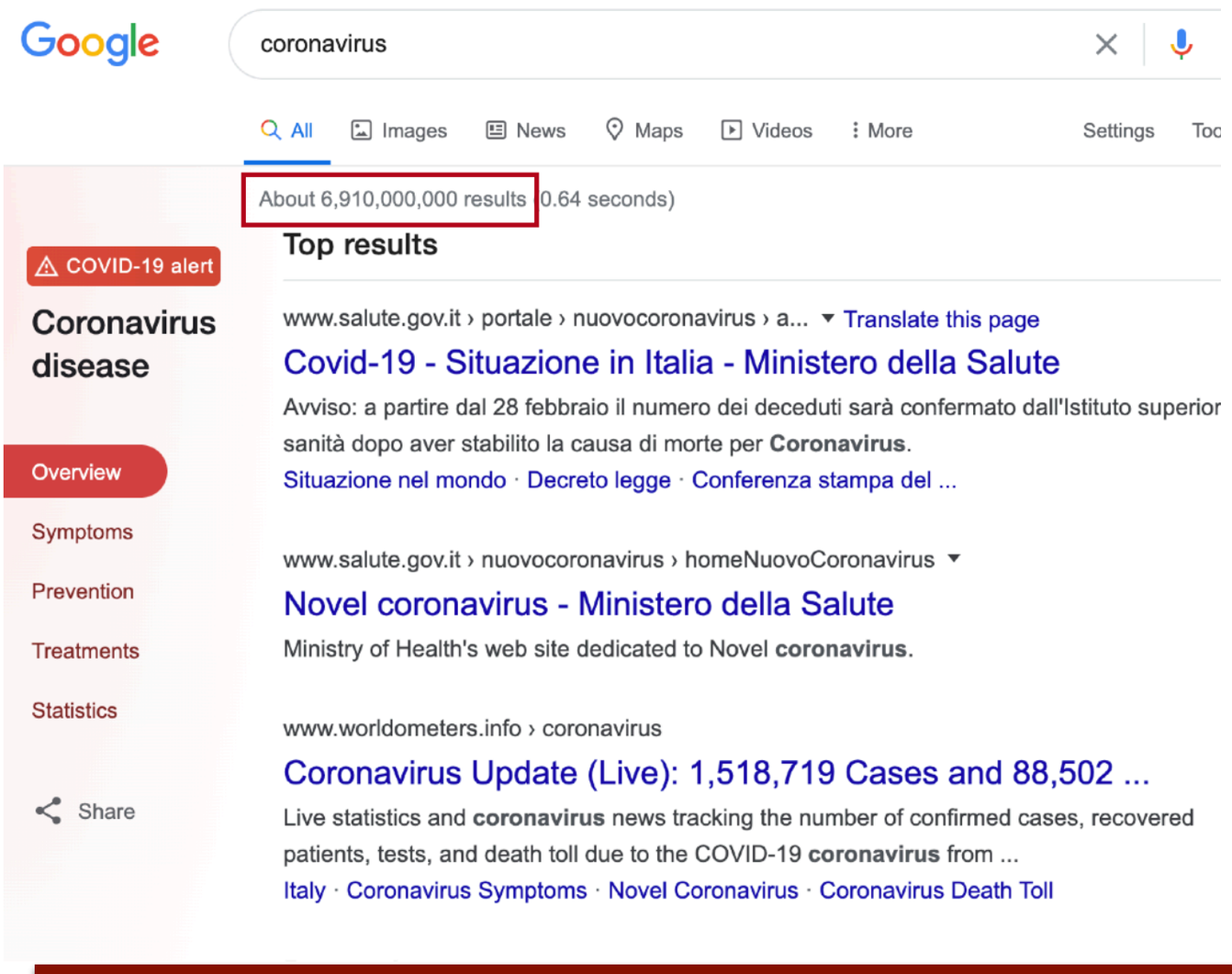


ROC curves are obtained by computing precision and recall figures as a function of hyperparameters (such as a threshold)

  - **Area Under Curve (AUC)**: a single score computed from ROC

Metrics

- How to evaluate performances?
  - **Precision-Recall** are the common metrics for information retrieval tasks



- The relevant set (all groundtruth instances) might be huge
- Usually, as an output, you get a ranked list
- Often we compute **Prec@K** and **Rec@K** on the top-K results

Metrics

- How to evaluate performances?
  - **Average Precision (AP)**: is a measure that combines recall and precision for ranked retrieval results
    - AP takes into account also the position in the ranking
    - Usually we report mean AP (**mAP**) for a set of queries

$$AP = \frac{\sum_{k=1}^{n} P(k) \cdot rel(k)}{\textit{all groundtruth instances}}$$

$P(k)$ is precision at cut-off k in the list

$rel(k)$ is an indicator function equaling to 1 if the item at rank k is relevant, 0 otherwise

$$mAP = \frac{1}{Q} \cdot \sum_{q=1}^{Q} AP(q)$$

where Q is the number of queries

Metrics

- How to evaluate performances?
  - **Average Precision (AP)**: is a measure that combines recall and precision for ranked retrieval results

Rank

Relevant	Precision	Recall
1	1	0.2
1	1	0.4
0	0.67	0.4
1	0.75	0.6
0	0.6	0.6
1	0.67	0.8
0	0.57	0.8
0	0.5	0.8
0	0.44	0.8
1	0.5	1

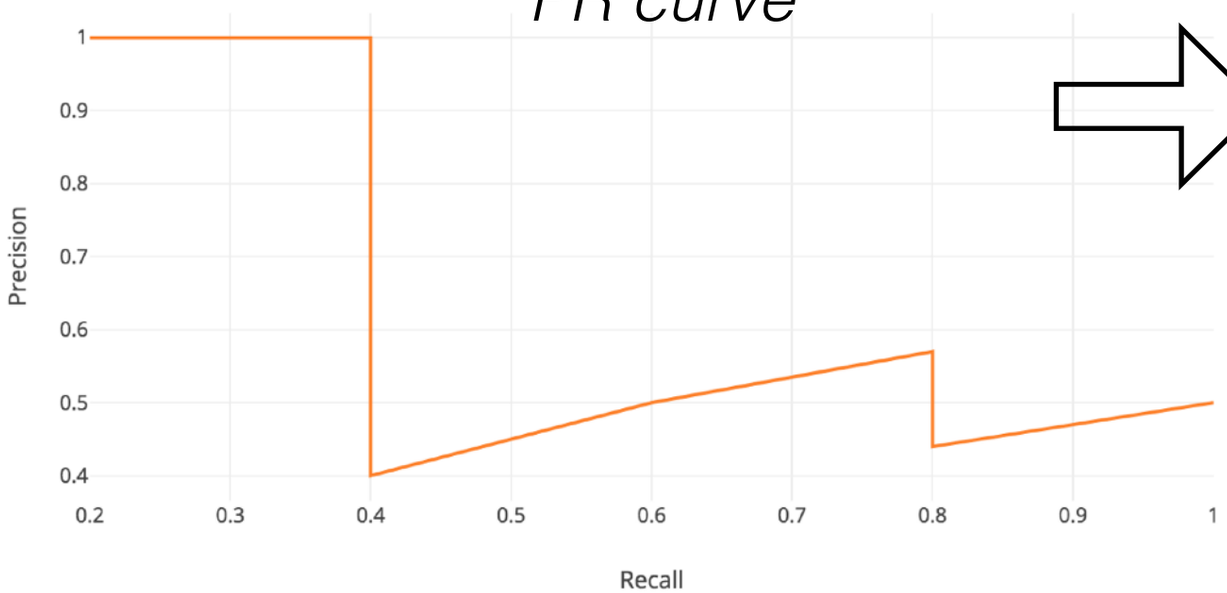
Query: “dog”

$$AP = \frac{1}{5} \cdot (1 + 1 + 0.75 + 0.67 + 0.5) = 0.78$$

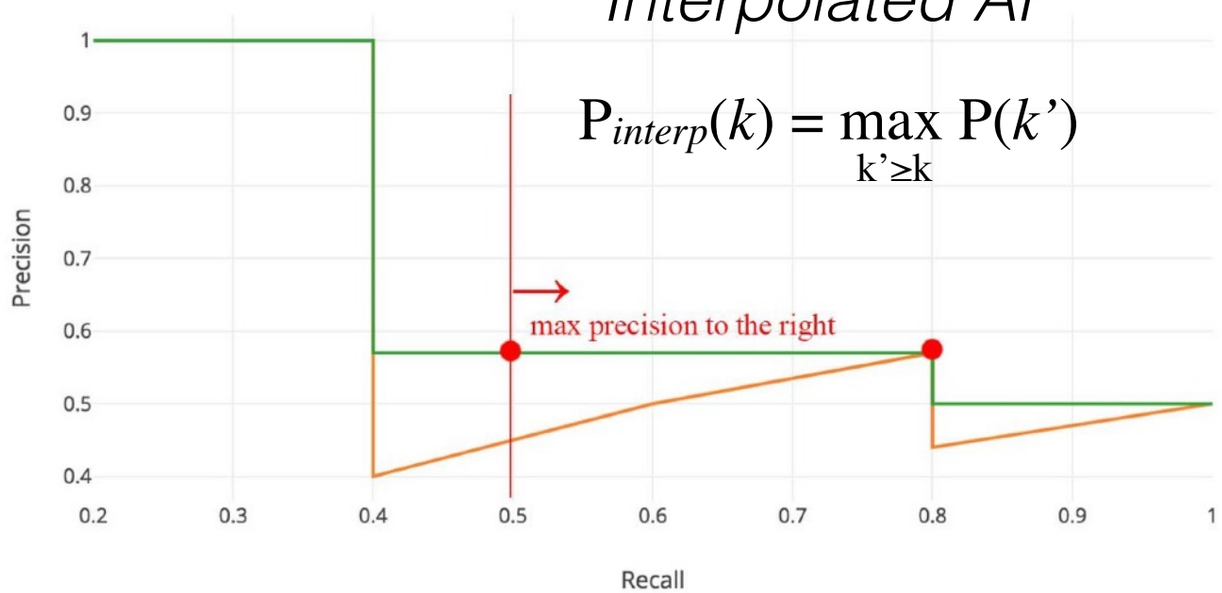
Metrics

- How to evaluate performances?
  - The general definition for the **Average Precision (AP)** is finding the area under the precision-recall curve above
    - Recall increases as we go down the prediction ranking
    - Precision has a “zigzag” pattern (it goes down with false positives and goes up again with true positives)

PR curve



Interpolated AP

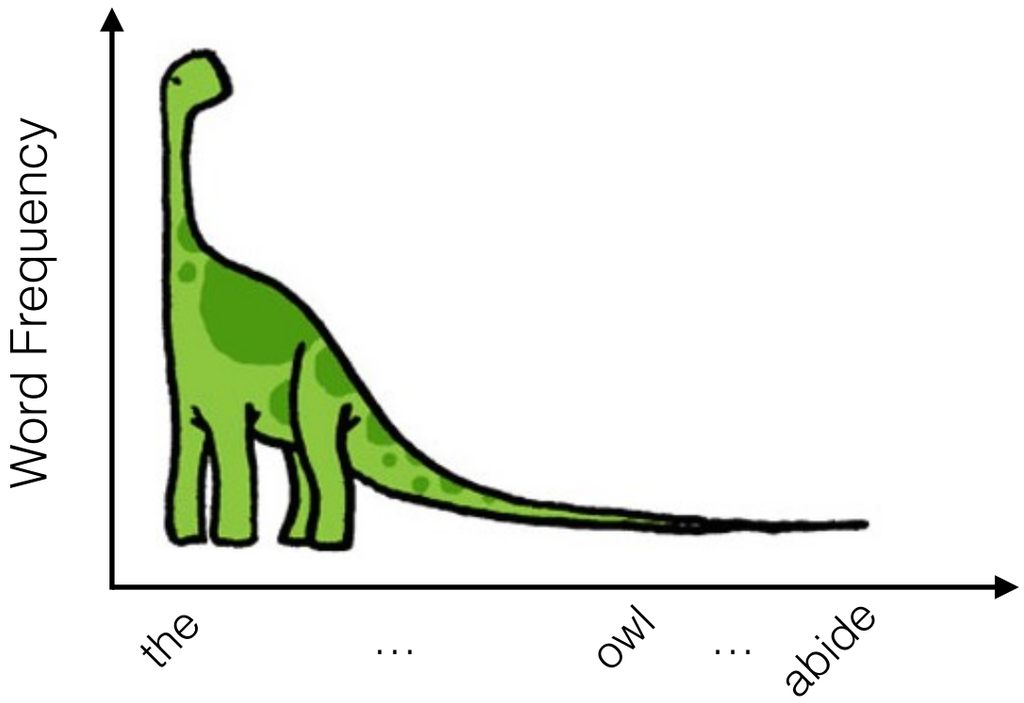


Metrics vs Loss (cost) Functions

- Metrics are selected on a dataset/benchmark and they measure what we care about (performance)
  - We typically cannot directly optimize for the metrics!
- The loss (cost) function should reflect the problem we are solving
  - We hope it will yield models that will do well on our dataset

Unbalanced Data

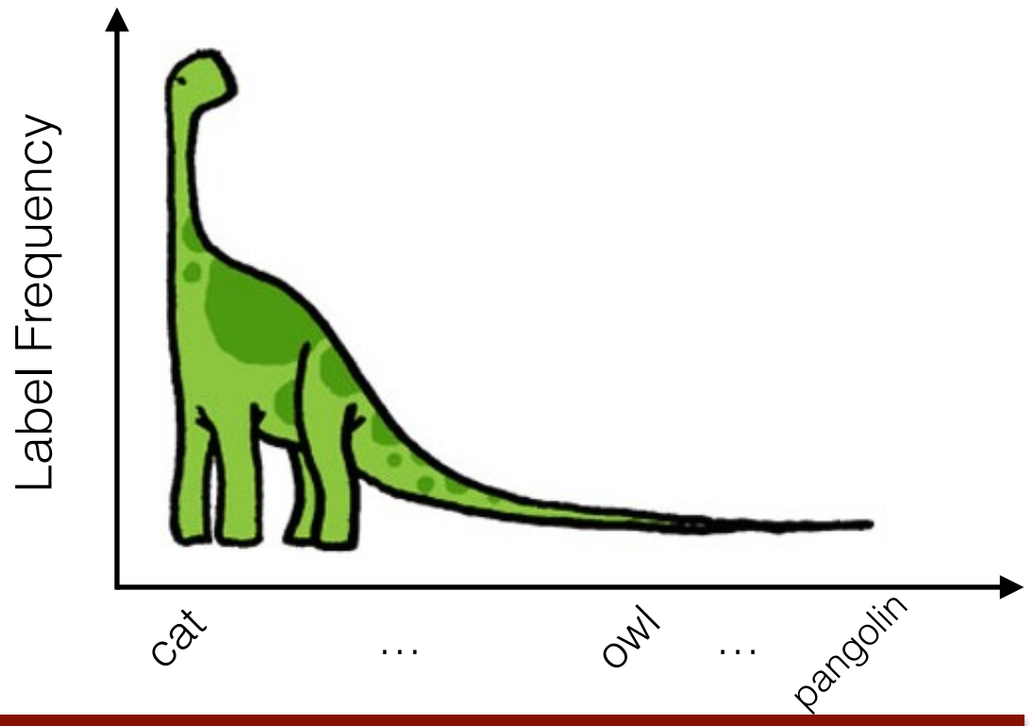
- Here we refer to classification problems where we have *unequal instances* for different classes
  - Having unbalanced data is pretty common (actually we can say this is the standard in real-world)
  - Long tail distributions: in almost any scenario the real distribution of data (labels) is long tailed



This could be the case of word frequencies in a document

Unbalanced Data

- Here we refer to classification problems where we have *unequal instances* for different classes
  - Having unbalanced data is pretty common (actually we can say this is the standard in real-world)
  - Long tail distributions: in almost any scenario the real distribution of data (labels) is long tailed



This could be the case of label (class) frequencies in a dataset such as ImageNet or, more in general, on the web

(Un)balanced Data

- Accuracy is very bad if you have unbalanced data
  - Let's take another look at our 1st example: dog vs no-dog

Accuracy = (4+4)/11 = 0.73 Chance = 0.5

TP: True Positive **FP:** False Positive **TN:** True Negative **FN:** False Negative

Unbalanced Data

8 3

- Accuracy is very bad if you have unbalanced data
  - Let's try with a different (unbalanced) dataset

Accuracy = (5+2)/11 = 0.64 Chance = 0.5

TP: True Positive **FP:** False Positive **TN:** True Negative **FN:** False Negative

Unbalanced Data

- Accuracy is very bad if you have unbalanced data
  - Let's try with a different (unbalanced) dataset

$\text{Accuracy}_{alldogs} = (8+0)/11 = 0.73$ Chance = 0.5

TP: True Positive **FP:** False Positive **TN:** True Negative **FN:** False Negative

Unbalanced Data

- I would recommend to use accuracy only if the classes are balanced
  - Precision-Recall and mAP are usually more robust
  - However, you can compute *weighted accuracy* by taking into account the number of instances in each class

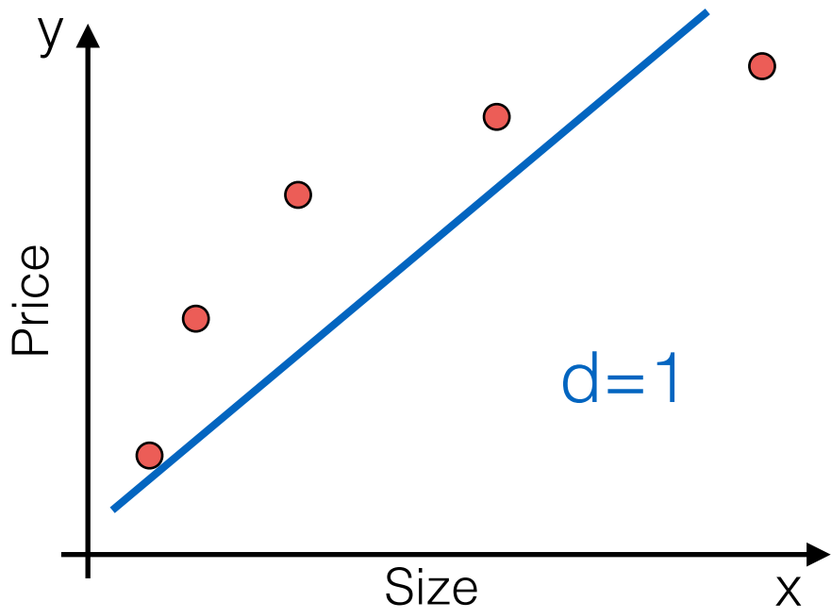
- More in general, in case of multiple classes you can compute both *micro* an *macro* averages
  - i.e. you can compute averages w.r.t. to each class or single instance assignment

Diagnosing bias *vs* variance

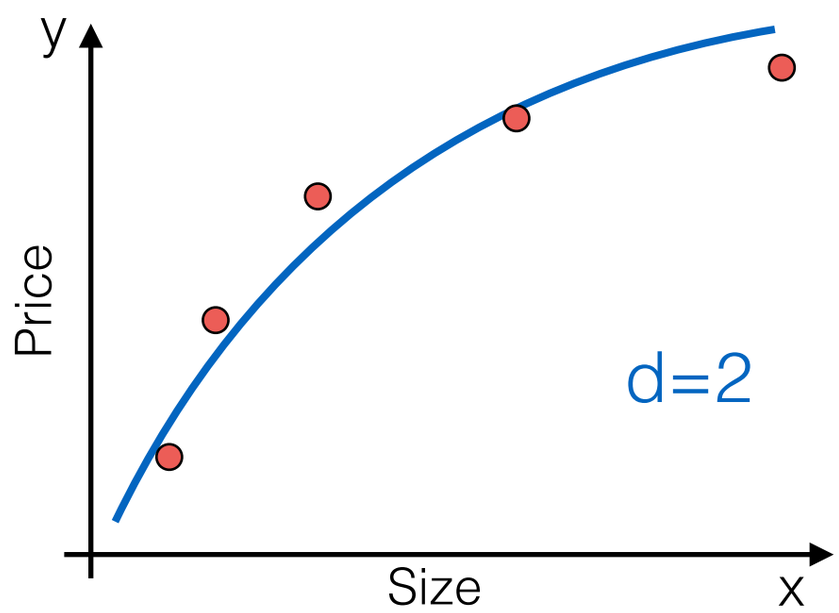
- If your learning model doesn't work as expected, almost all the time it will be because you have either a *high bias* problem or a *high variance* problem
  - How to figure out what's happening (in practice)?
  - What can we do to fix/alleviate the problem?

Diagnosing bias vs variance

Underfitting (high bias)

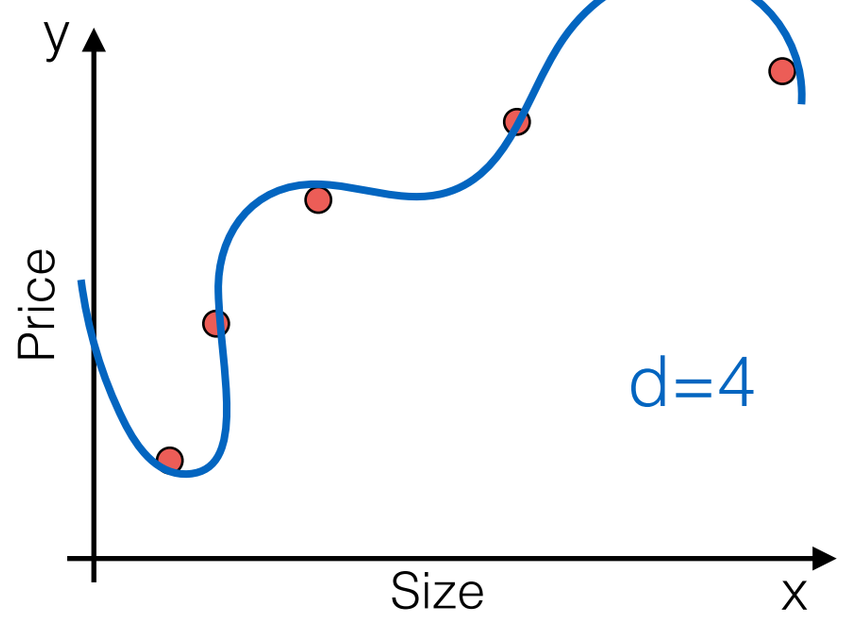


$h_\theta(x) = \theta_0 + \theta_1 x$



$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$

Overfitting (high variance)

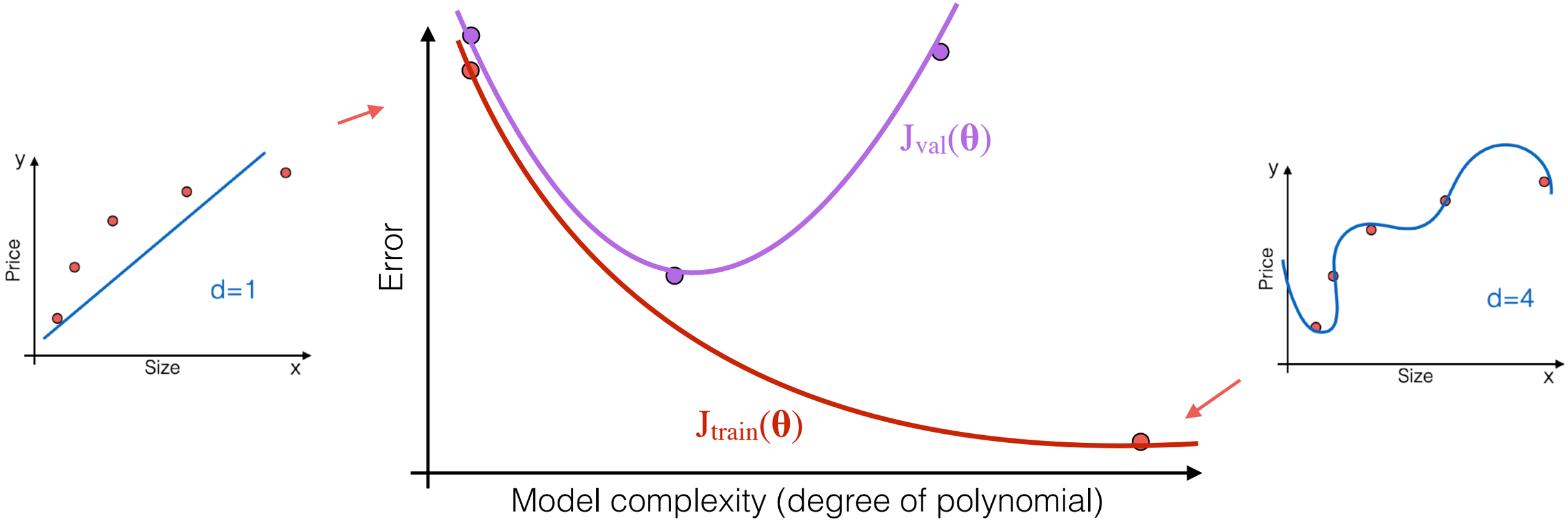


$\theta_0+\theta_1 x+\theta_2 x^2+\theta_3 x^3+\theta_4 x^4$

- We can now look again at this example taking into account hold-out and bias-variance tradeoff

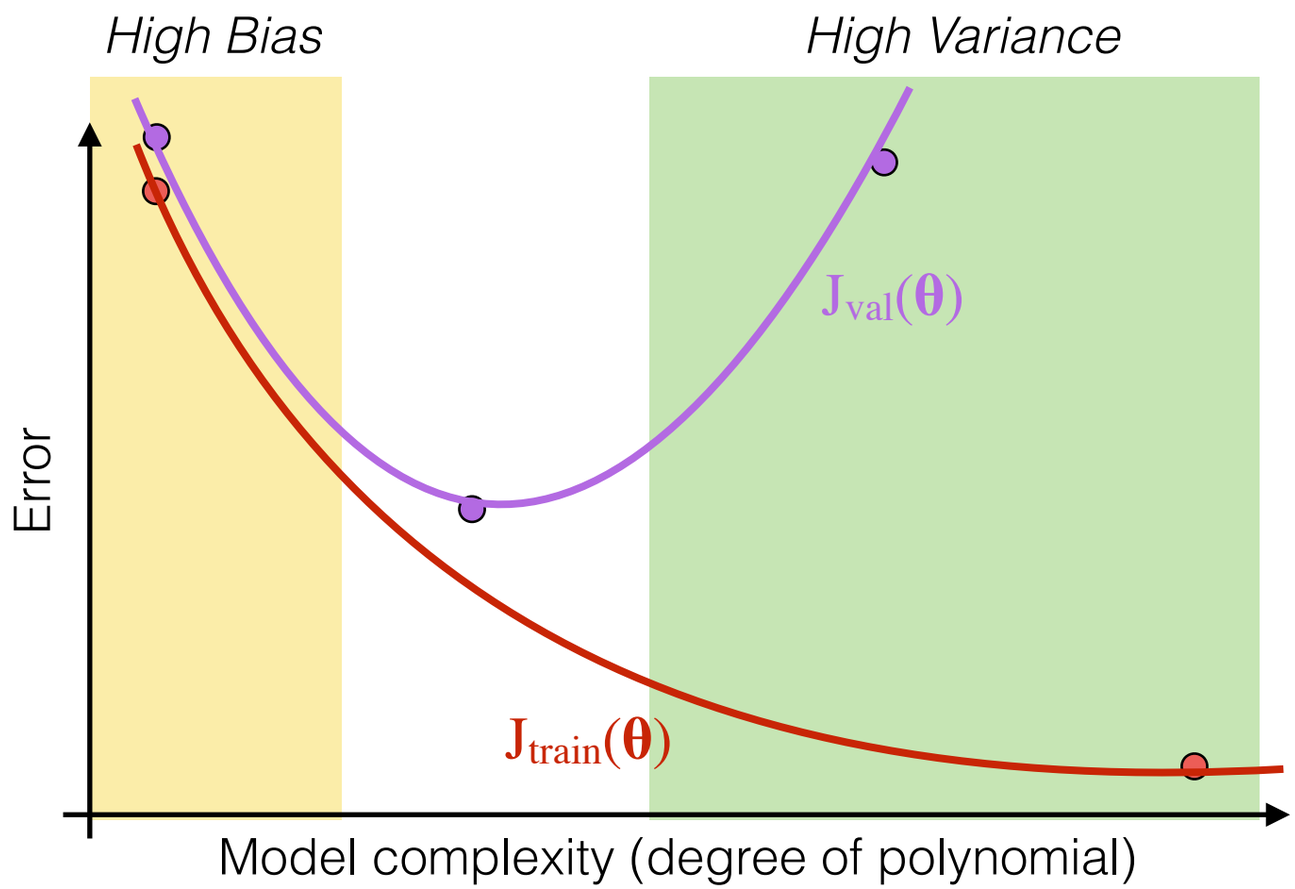
Diagnosing bias vs variance

- "Measuring" bias vs variance:
  - Training Error: $\mathbf{J_{train}(\theta)} = \frac{1}{2m_t} \sum_{i=1}^{m_t} (h_\theta(x^{(i)}) - y^{(i)})^2$
  - Validation Error: $\mathbf{J_{val}(\theta)} = \frac{1}{2m_v} \sum_{i=1}^{m_v} (h_\theta(x^{(i)}) - y^{(i)})^2$



Diagnosing bias vs variance

- Our learning model doesn't work as expected; is it a bias problem or a variance problem?



High bias (underfit):

$J_{train}(\theta)$ will be high

$J_{val}(\theta) \approx J_{train}(\theta)$

High variance (overfit):

$J_{train}(\theta)$ will be low

$J_{val}(\theta) \gg J_{train}(\theta)$

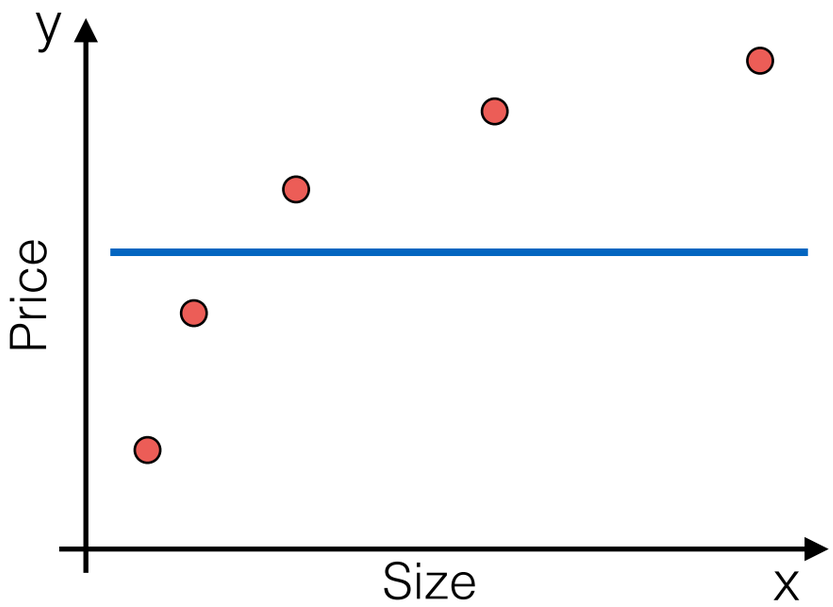
Diagnosing bias vs variance

- What's the contribution of regularization?

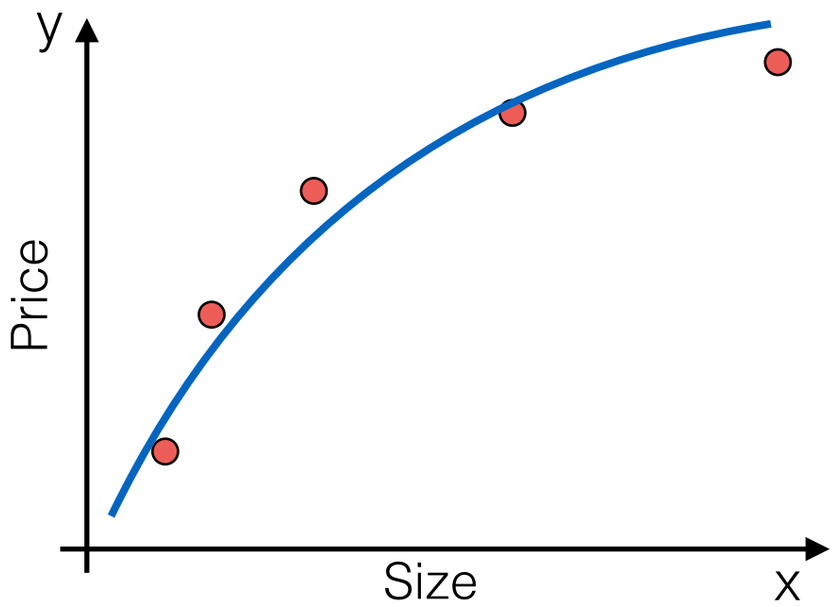
$$\min_{\boldsymbol{\theta}} \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

(high bias)

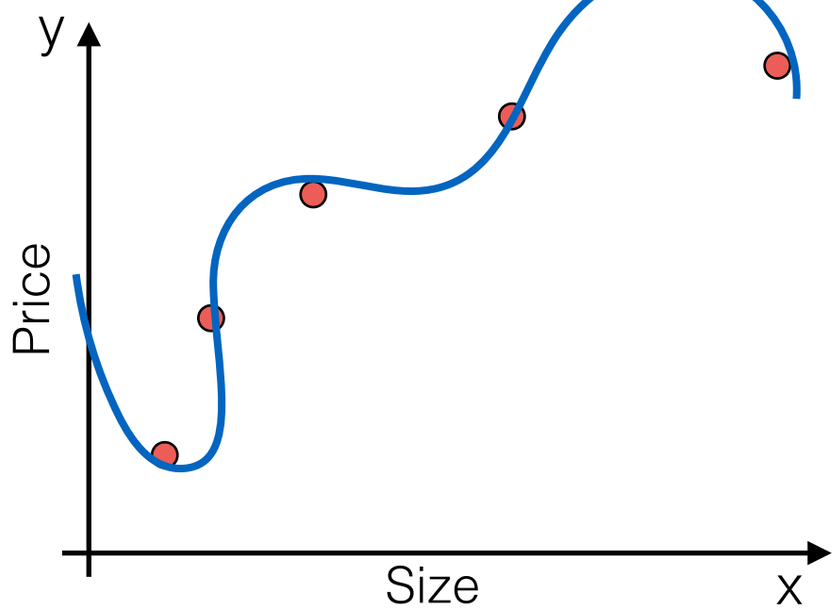


Large λ $h_\theta(x) \approx \theta_0$



Intermediate λ

(high variance)



Small λ (i.e. $\lambda \approx 0$)

Diagnosing bias vs variance

- Choosing the regularization parameter λ:

$$\min_{\boldsymbol{\theta}} \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

- Note: our definition of J_{train}, J_{val}, J_{test} don't change
  - Training Error: $J_{train}(\boldsymbol{\theta}) = \frac{1}{2m_t}\sum_{i=1}^{m_t}(h_\theta(x^{(i)}) - y^{(i)})^2$
  - Validation Error: $J_{val}(\boldsymbol{\theta}) = \frac{1}{2m_v}\sum_{i=1}^{m_v}(h_\theta(x^{(i)}) - y^{(i)})^2$
  - Test Error: $J_{test}(\boldsymbol{\theta}) = \frac{1}{2m_e}\sum_{i=1}^{m_e}(h_\theta(x^{(i)}) - y^{(i)})^2$

Diagnosing bias vs variance

- Choosing the regularization parameter λ:

$$\min_{\boldsymbol{\theta}} \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$

Model: $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Model Selection

1: try λ=0 → $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \boldsymbol{\theta}^{(1)}$ → $J_{val}(\boldsymbol{\theta}^{(1)})$

2: try λ=0.01 → $\boldsymbol{\theta}^{(2)}$ → $J_{val}(\boldsymbol{\theta}^{(2)})$

3: try λ=0.02 → $\boldsymbol{\theta}^{(3)}$ → $J_{val}(\boldsymbol{\theta}^{(3)})$ *(lowest)*

4: try λ=0.04 → $\boldsymbol{\theta}^{(4)}$ → $J_{val}(\boldsymbol{\theta}^{(4)})$

5: try λ=0.08 → $\boldsymbol{\theta}^{(5)}$ → $J_{val}(\boldsymbol{\theta}^{(5)})$

⋮

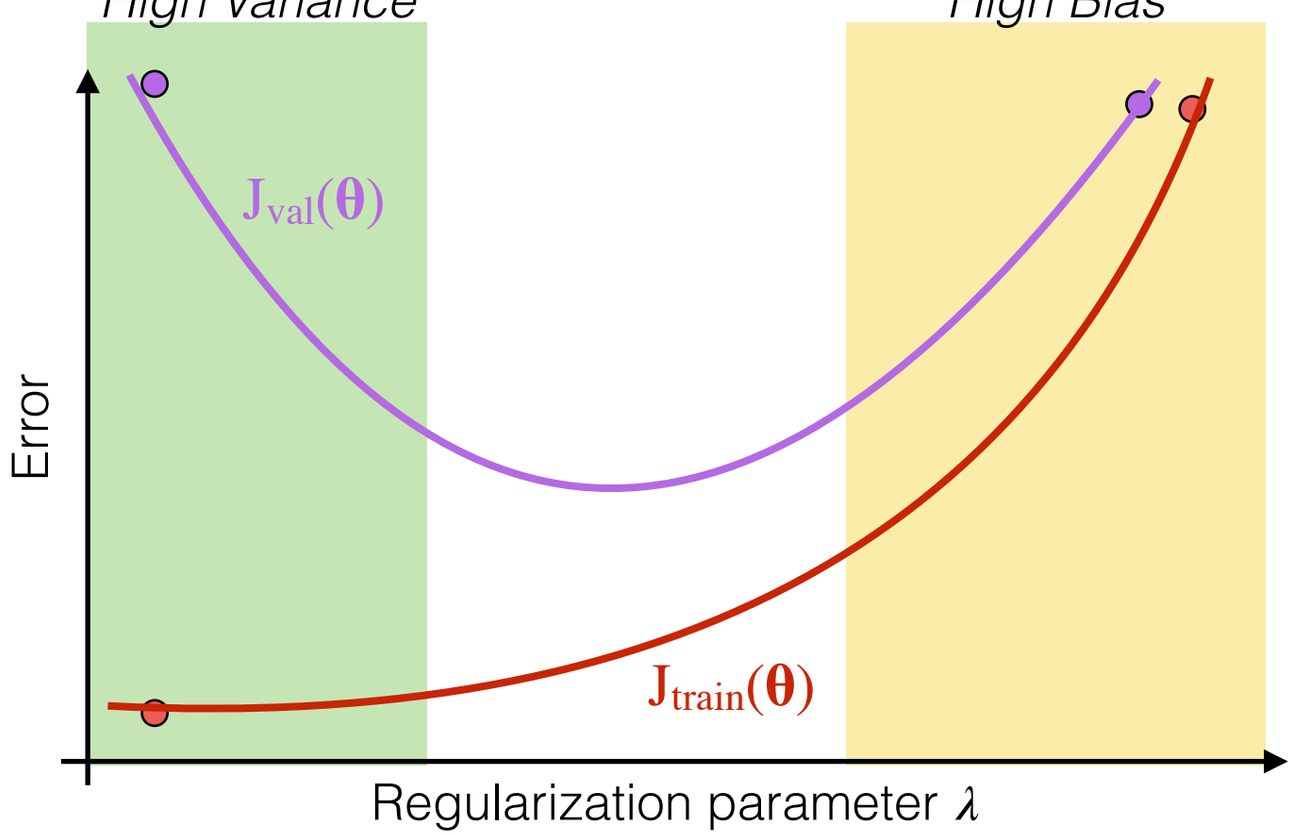
12: try $\lambda \approx 10$ → $\boldsymbol{\theta}^{(12)}$ → $J_{val}(\boldsymbol{\theta}^{(12)})$

Adapted from slide by A. Ng

Diagnosing bias vs variance

- Bias/Variance as a function of the parameter λ:

$$\min_{\theta} \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$



$$J_{train}(\theta) = \frac{1}{2m_t}\sum_{i=1}^{m_t}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$J_{val}(\theta) = \frac{1}{2m_v}\sum_{i=1}^{m_v}(h_\theta(x^{(i)}) - y^{(i)})^2$$

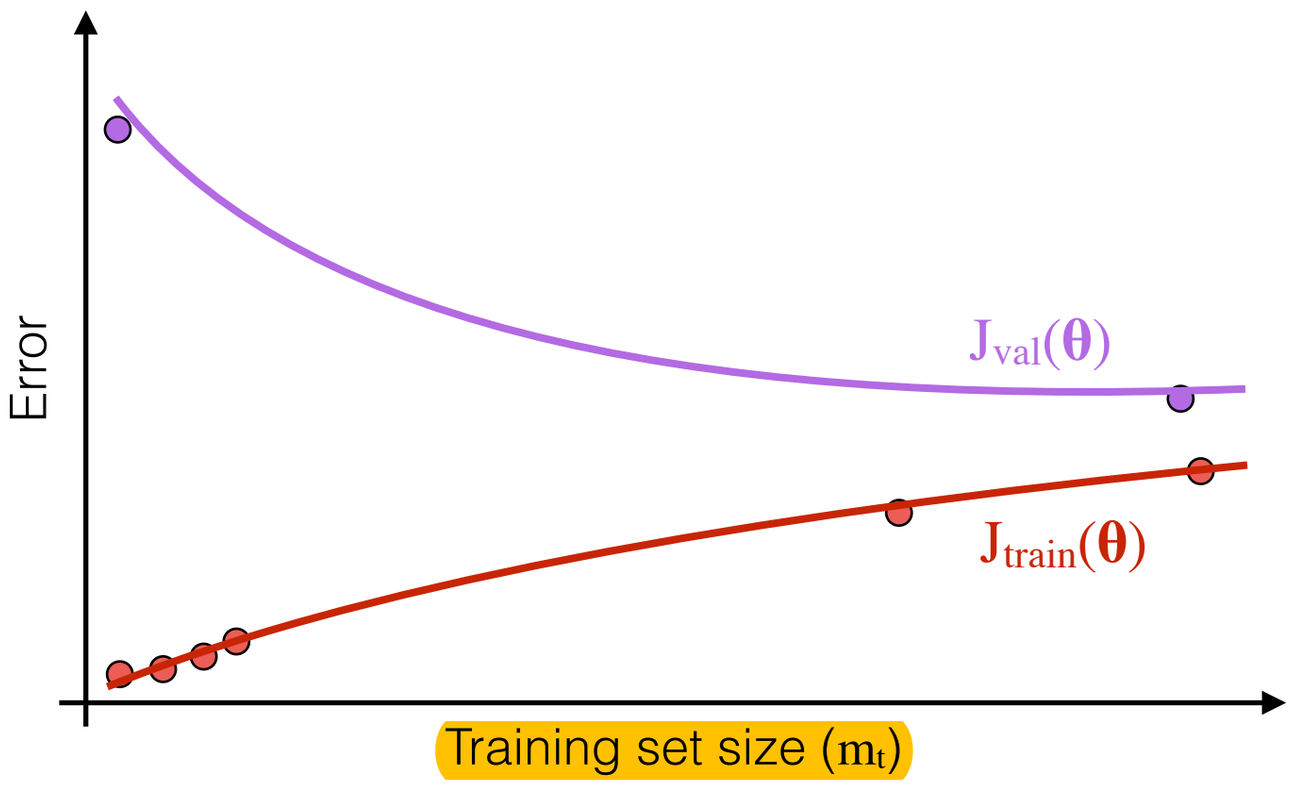
Diagnosing bias vs variance

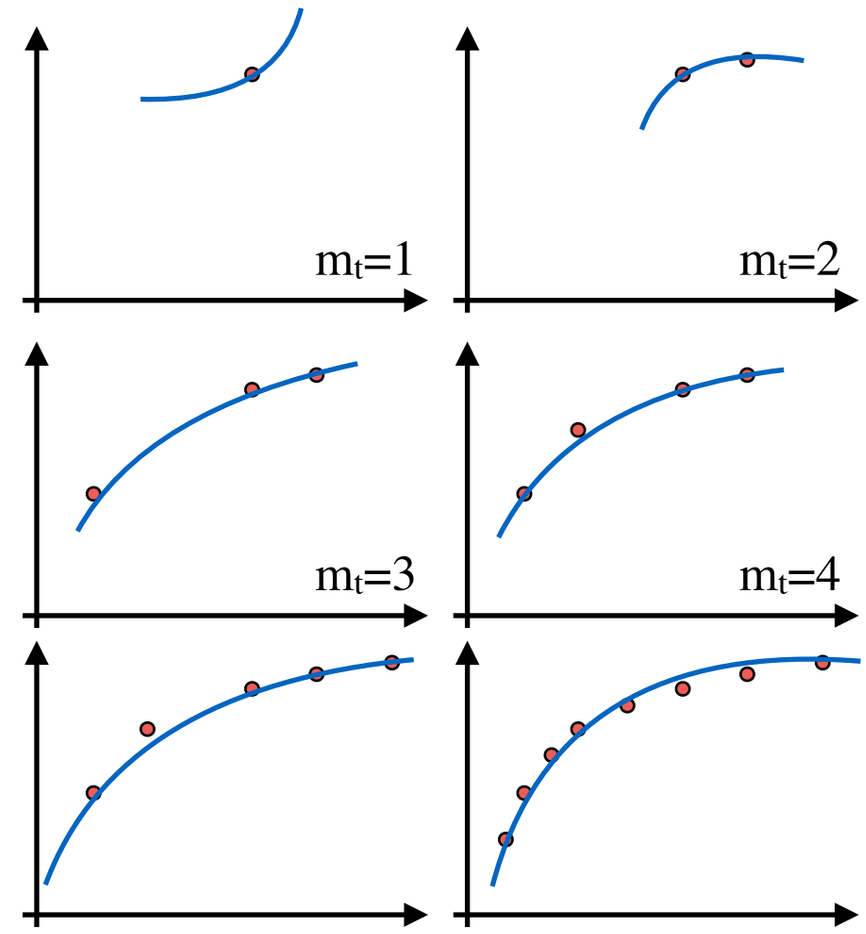
- By now you have seen bias and variance from a lot of different perspectives
- Let's now take all the insights we have gone through in order to build a "diagnostic tool" for ML systems

Learning Curves

- Learning curves can be used to diagnose if a model may be suffering from bias, variance or a bit of both

$$J_{train}(\boldsymbol{\theta}) = \frac{1}{2m_t} \sum_{i=1}^{m_t} (h_\theta(x^{(i)}) - y^{(i)})^2 \qquad J_{val}(\boldsymbol{\theta}) = \frac{1}{2m_v} \sum_{i=1}^{m_v} (h_\theta(x^{(i)}) - y^{(i)})^2$$



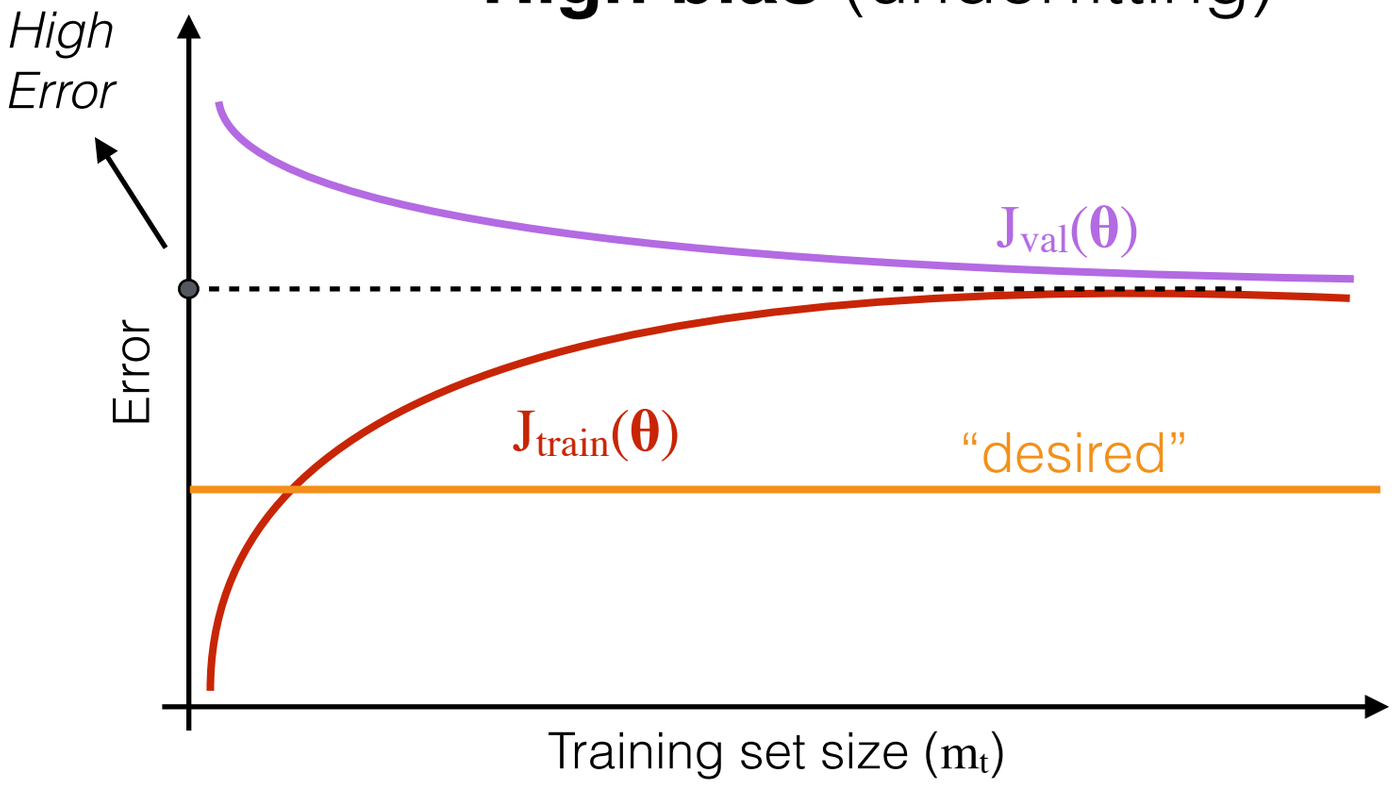


$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$

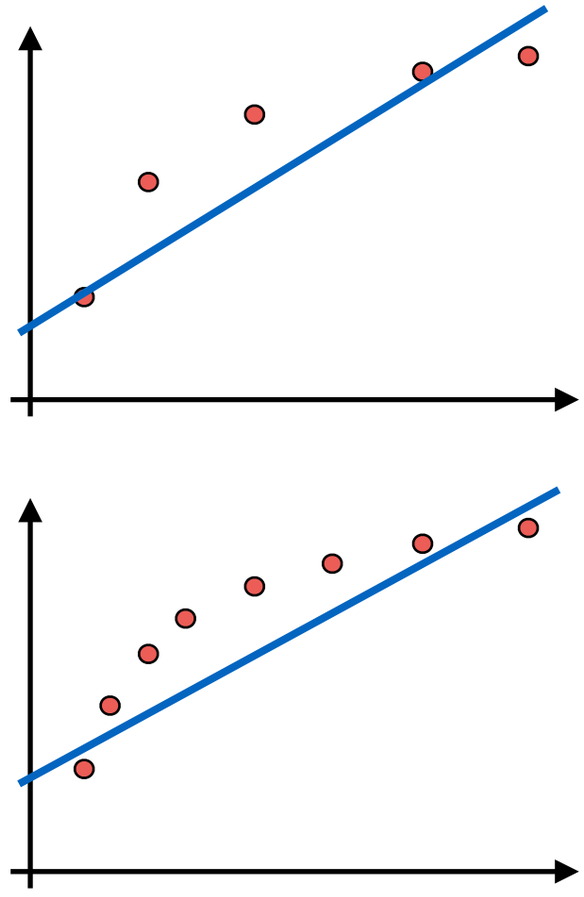
Learning Curves

- That's the general intuition… but what's about bias and variance problems?

High bias (underfitting)



$h_\theta(x) = \theta_0 + \theta_1 x$

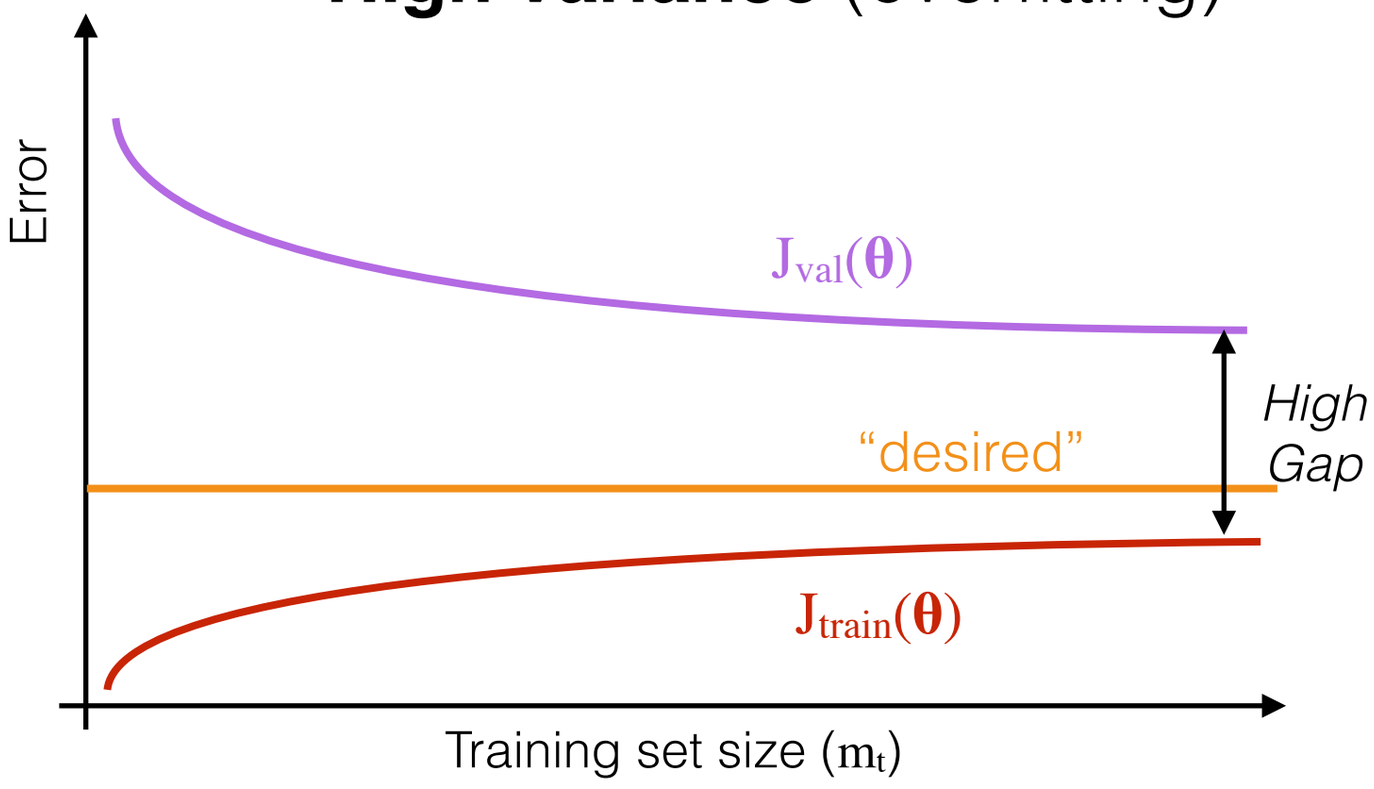


Note: in case of high bias, getting more training data will not help much

Learning Curves

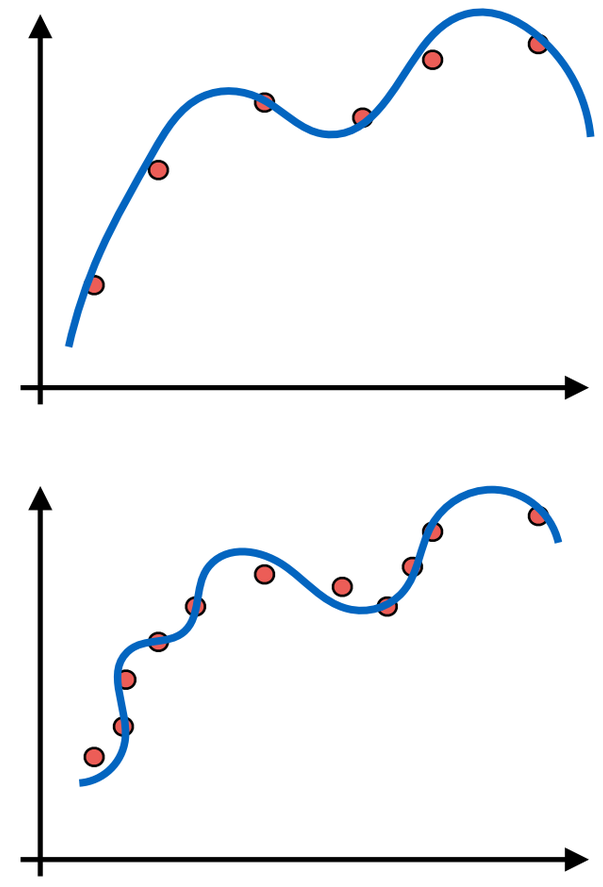
- That's the general intuition… but what's about bias and variance problems?

High variance (overfitting)



$h_\theta(x) = \theta_0 + \theta_1 x + \ldots + \theta_{50} x^{50}$

(*and small* λ)

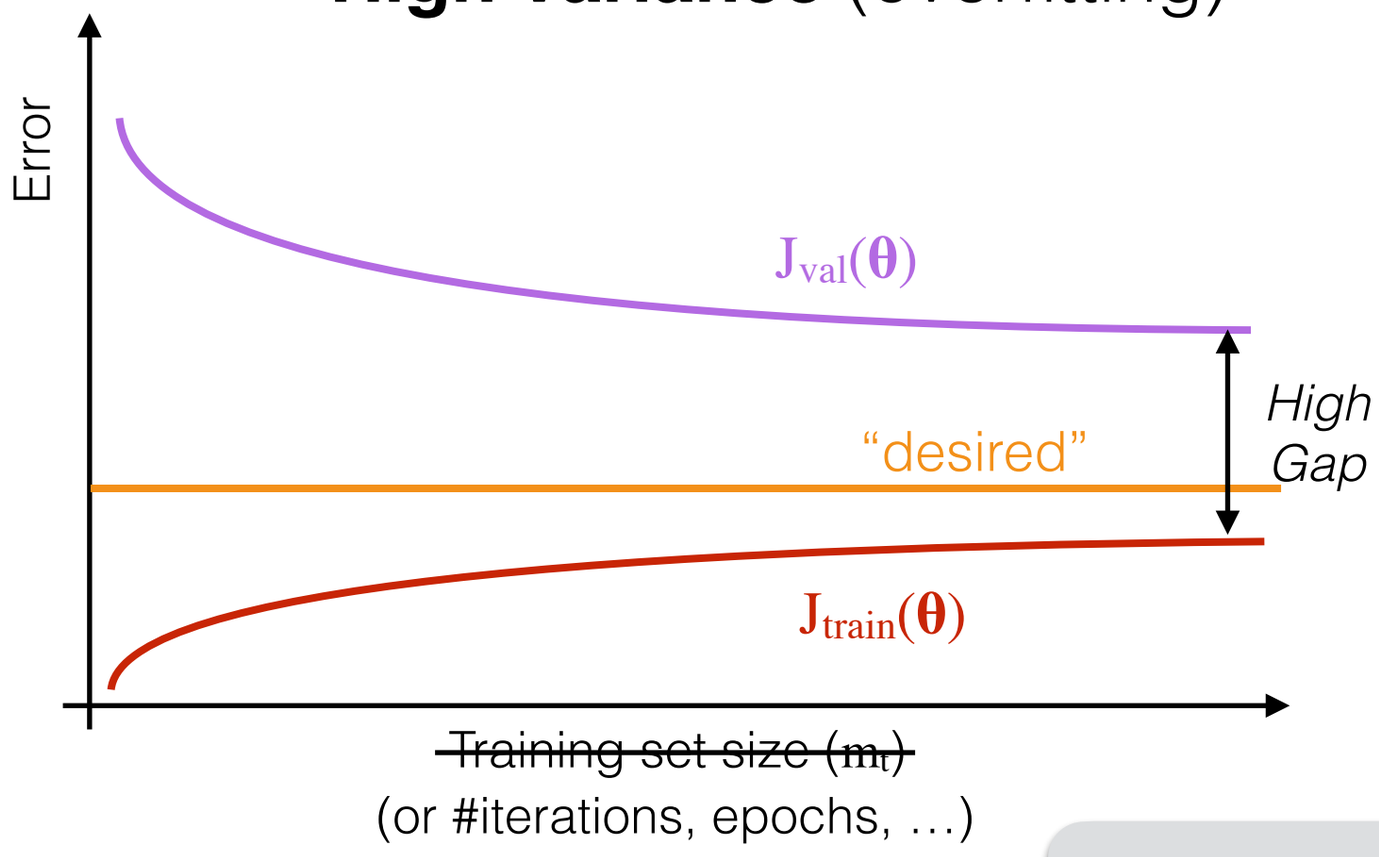


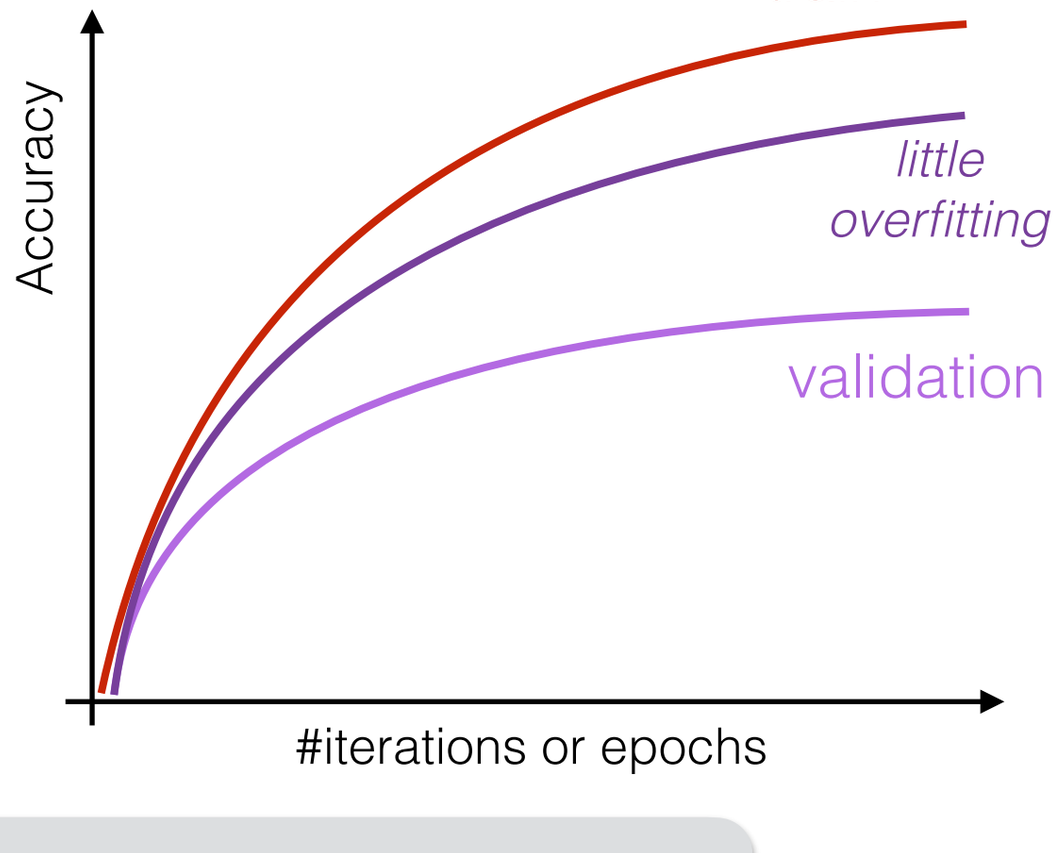
Note: *in case of high variance, getting more training data is likely to help*

Learning Curves

- You can compute learning curves w.r.t. different "dimensions" (e.g. evaluation measures, no. samples)

High variance (overfitting)



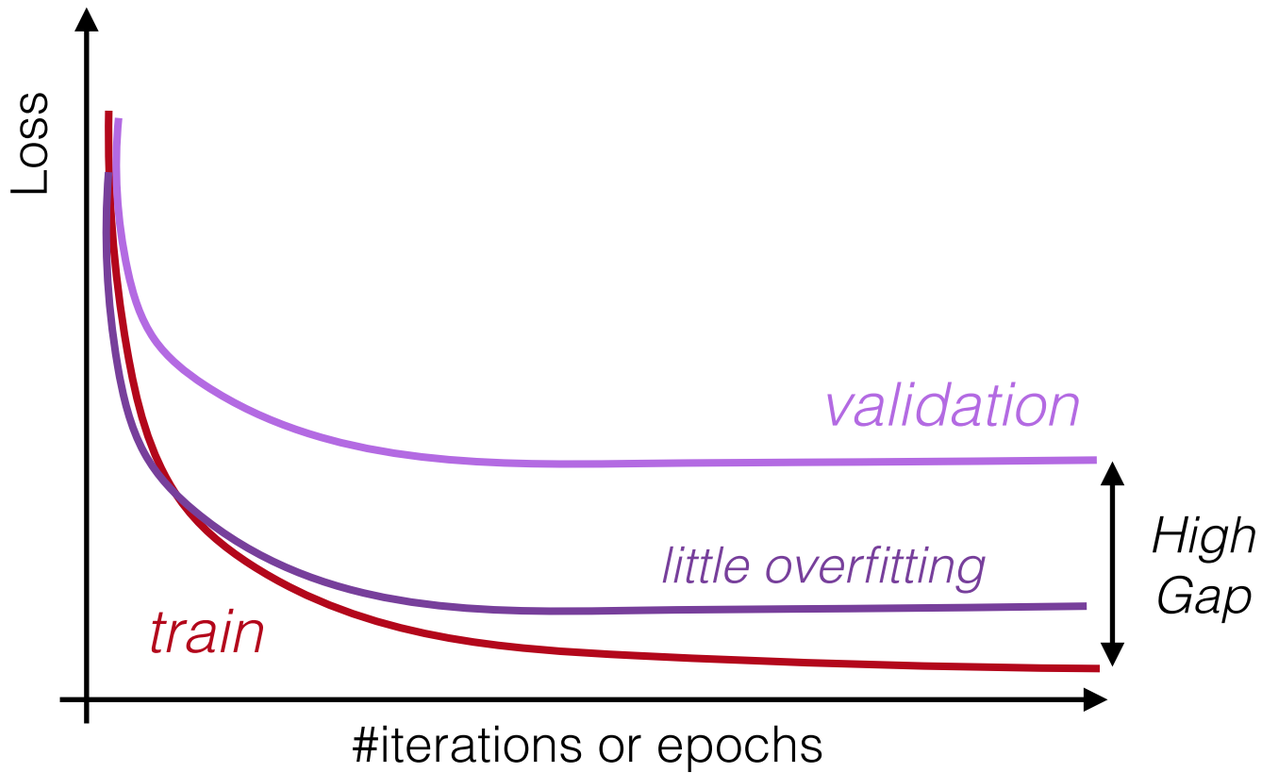


epoch = a pass of the entire set of examples

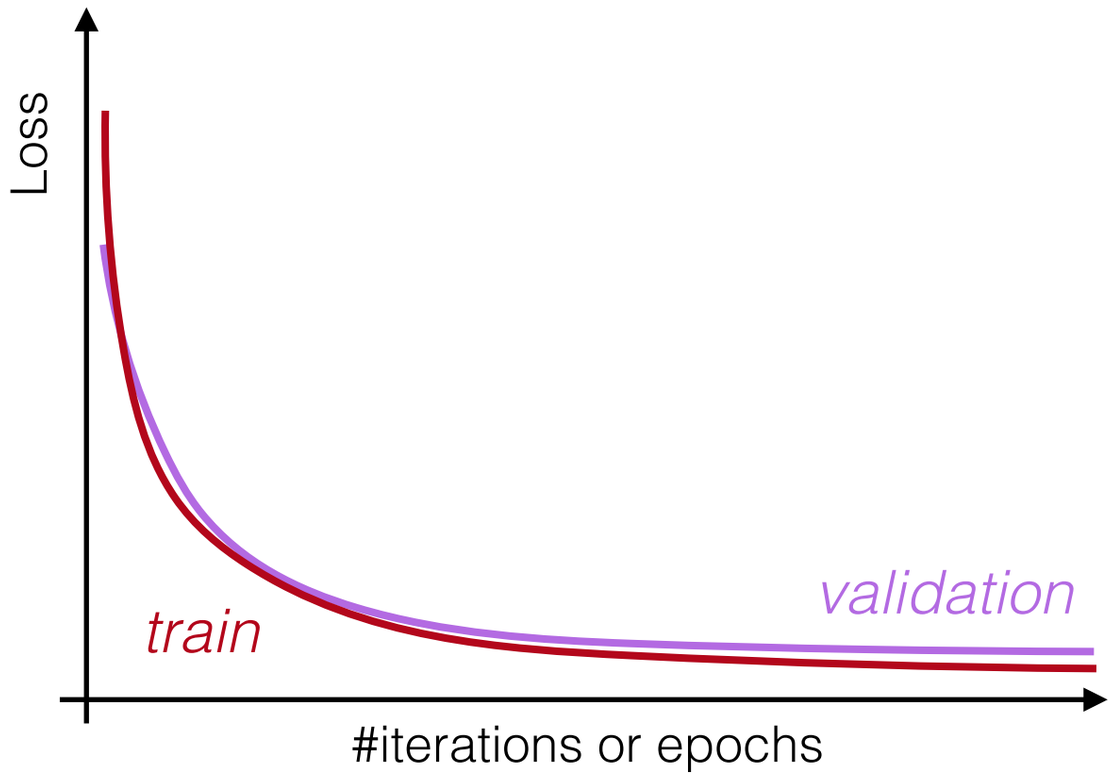
Learning Curves

- Often learning curves are plotted w.r.t. the loss

(overfitting)



(great fit)



What to do next

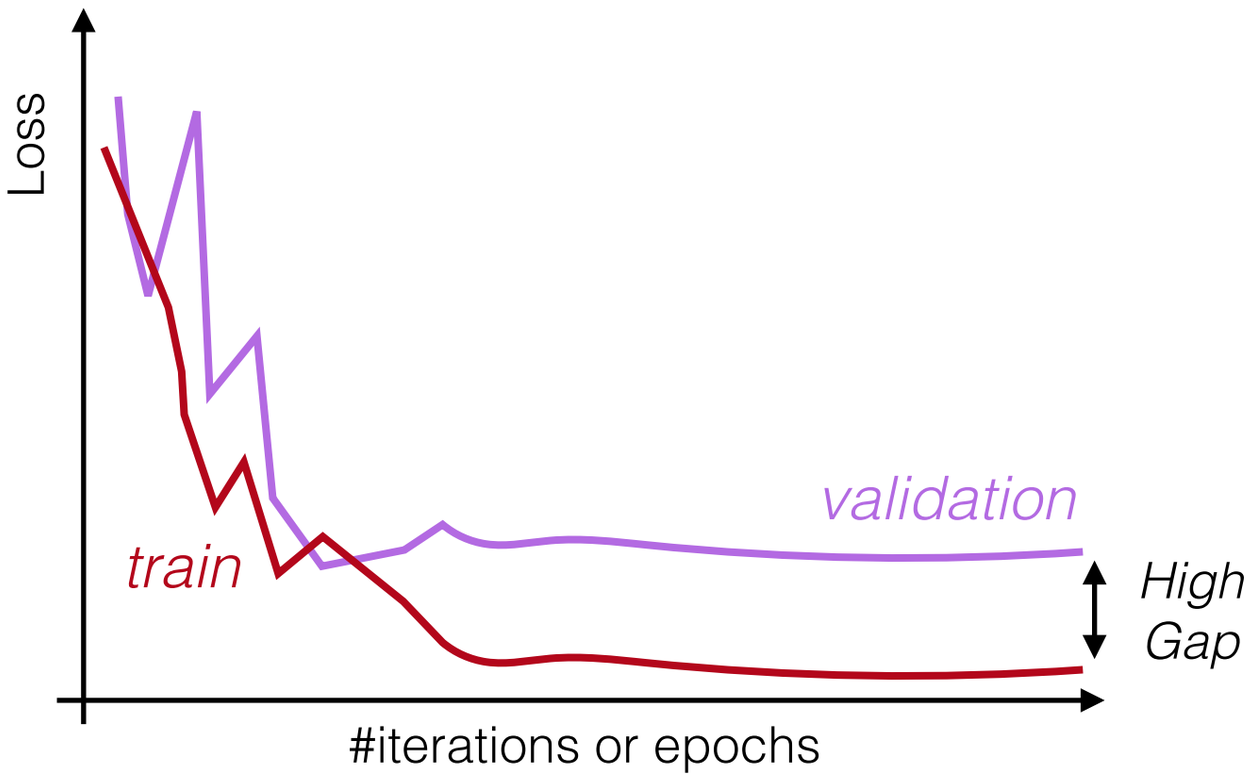
- Debugging (and babysitting) a learning algorithm:
  - Suppose you have implemented a regularized linear regression model for predicting housing prices
  - It doesn't work on new data; what should you do next?
    - You can get more training data → *Fixes high variance*
    - Try smaller set of features → *Fixes high variance*
    - Try getting more features → *Fixes high bias*
    - Try adding complexity to the model (e.g. polynomial features) → *Fixes high bias*
    - Try decreasing λ → *Fixes high bias*
    - Try increasing λ → *Fixes high variance*

Adapted from slide by A. Ng

Diagnosing our datasets

- Learning curves can be also used to diagnose the quality of our training/validation sets

Unrepresentative Training Set

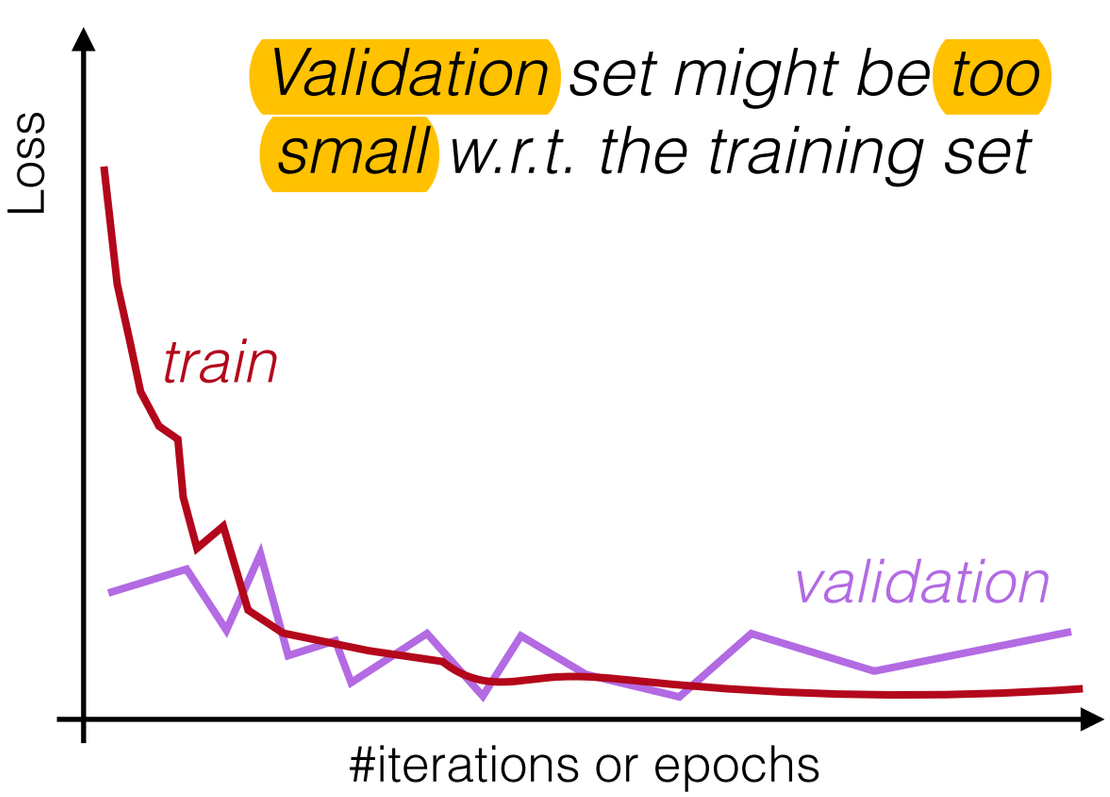


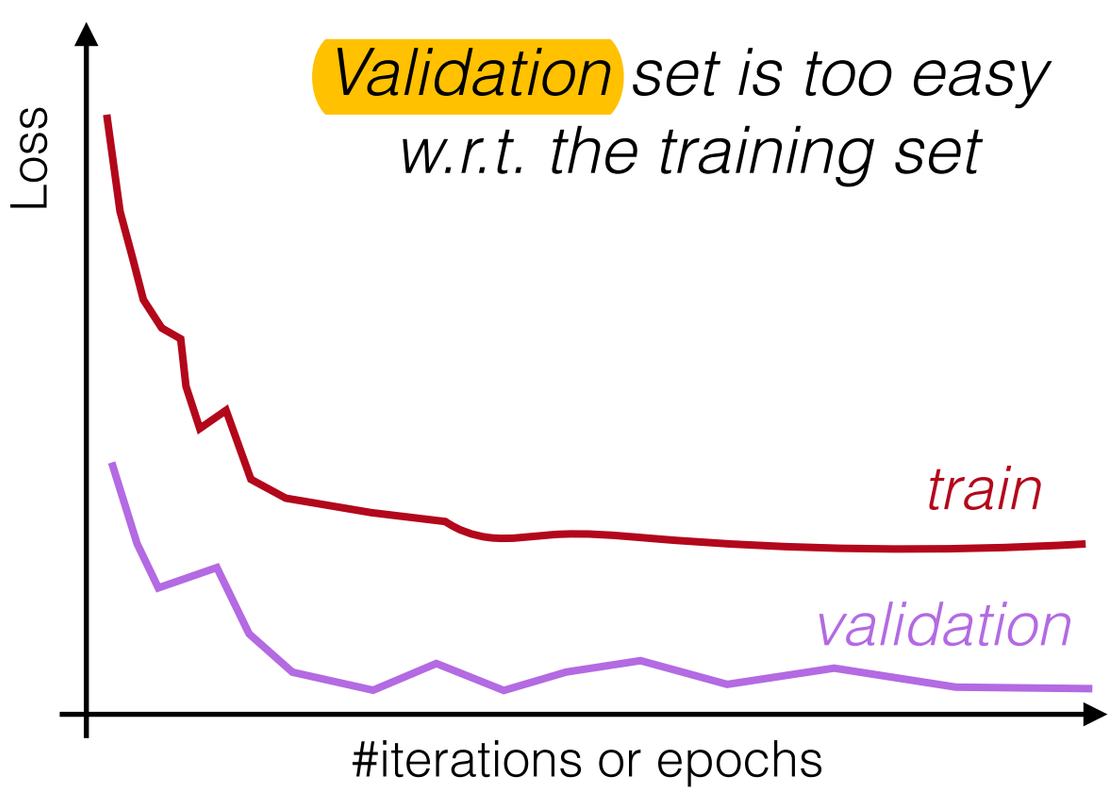
- The training set does not provide sufficient information to learn the problem
- It may occur if the training set has too few examples as compared to the validation set

Diagnosing our datasets

- Learning curves can be also used to diagnose the quality of our training/validation sets

Unrepresentative Validation Set





Contact

- **Office:** Torre Archimede, room 6CD3
- **Office hours** (ricevimento)**:** Friday 9:00-11:00

✉ lamberto.ballan@unipd.it

🏠 http://www.lambertoballan.net

🏠 http://vimp.math.unipd.it

@ twitter.com/lambertoballan