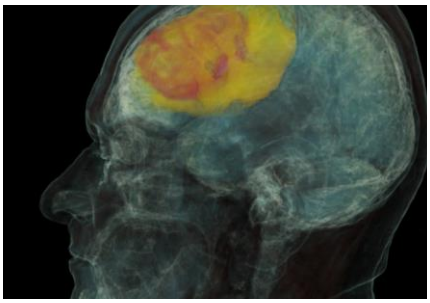
TÉCNICO LISBOA

Practical Introduction to ML Workshop

Guillherme Ilunga
guilherme.ilunga@tecnico.ulisboa.pt

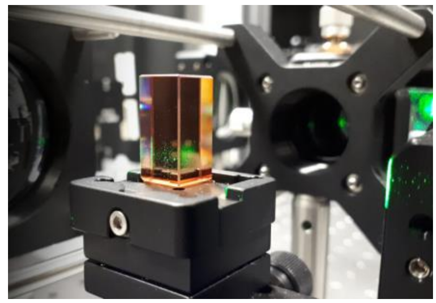
Short Bio

InnerEye



https://www.microsoft.com/en-us/research/project/medical-image-analysis/

Holographic Storage Devices



https://www.microsoft.com/en-us/research/project/hsd/

Workshop Outline

1. Introduction to Supervised Learning
    a) Introduction to Regression with Linear Regression
    b) Introduction to Classification with Logistic Regression

2. Practical ML Example – Titanic Survival Prediction

3. Resources

TÉCNICO LISBOA

Introduction to Supervised Learning

What is Machine Learning?

- *"Set of methods that can automatically* ***detect patterns in data****, and then use the uncovered patterns to* ***predict future data****, or to perform other kinds of* ***decision making*** *under uncertainty"*

  Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective.*

- *"(…) AI systems need the ability to acquire their own knowledge, by* ***extracting patterns from raw data****. This capability is known as machine learning"*

  Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*.

- Machine Learning: A Probabilistic Perspective: https://www.cs.ubc.ca/~murphyk/MLbook/
- Deep Learning Book: https://www.deeplearningbook.org/

Supervised vs Unsupervised Learning

Supervised Learning

- Predictive approach
- Requires labelled data
- Most widely used in practice
- Examples:
  - Predict house price (regression)
  - Classify images (classification)

Unsupervised Learning

- Descriptive approach
- Does not require labels
- Harder problem
- Examples:
  - Discover groups (clustering)
  - Reduce dimensions (e.g., PCA)

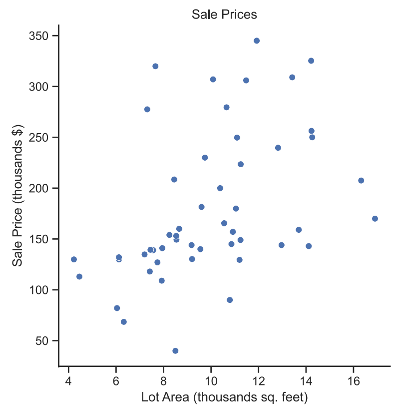
- Predictive approach: learn a mapping from x to y, i.e., how to predict y from x
- Descriptive approach: find interesting patterns in x
- Unsupervised is harder since there is no obvious error metric or well-defined goal

Supervised Learning
Basic Idea

- Goal is to learn **hypothesis** h which maps from input to target
- **Learning Algorithm** takes in dataset and returns h
- Regression: predict a **continuous** value
- Classification: predict a **discrete** value

Linear Regression

Example: Predict Sale Prices from Lot Area



Data: https://www.kaggle.com/c/house-prices-advanced-regression-techniques/overview

Linear Regression
Notation and Terminology

- Scalar: a
- Vector: $\boldsymbol{a}$
- Matrix: $\boldsymbol{A}$

Linear Regression

Notation and Terminology

- Input for example k: $\boldsymbol{x}^k = \begin{bmatrix} x_1^k \\ \vdots \\ x_n^k \end{bmatrix}$, where x_1^k is the first feature

$$\boldsymbol{x}^k = house^k = \begin{bmatrix} Lot\ Area \\ \vdots \\ \#Bedrooms \end{bmatrix}$$

- Note, sometimes the superscript k is omitted

Linear Regression

Notation and Terminology

- Input for example k: $\boldsymbol{x}^k = \begin{bmatrix} x_1^k \\ \vdots \\ x_n^k \end{bmatrix}$, where x_1^k is the first feature

- Target for example k: y^k

$$y^k = price\ of\ house^k$$

- Note, sometimes the superscript k is omitted

Linear Regression
Notation and Terminology

- Input for example k: $\boldsymbol{x}^k = \begin{bmatrix} x_1^k \\ \vdots \\ x_n^k \end{bmatrix}$, where x_1^k is the first feature
- Target for example k: y^k
- Training example: $(\boldsymbol{x}^k, y^k)$
- Dataset: $\{(\boldsymbol{x}^k, y^k); \mathrm{k} = 1, \ldots, \mathrm{m}\}$

- Note, sometimes the superscript k is omitted

Linear Regression
Notation and Terminology

- Input data for all examples:

$$\boldsymbol{X} = \begin{bmatrix} x_1^1 & \cdots & x_n^1 \\ \vdots & \ddots & \vdots \\ x_1^m & \cdots & x_n^m \end{bmatrix}$$

House ID	Lot Area (sq. feet)	# Bedrooms
1	8450	2
2	9600	3

- Note, sometimes the superscript k is omitted

Linear Regression

Notation and Terminology

- Target data for all examples:

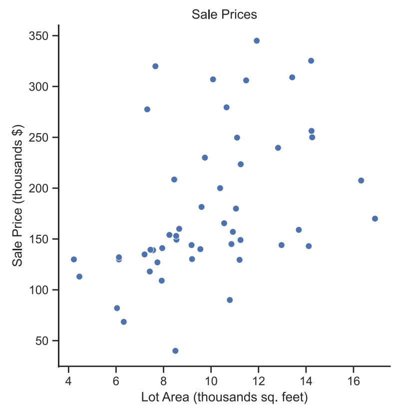
$$\boldsymbol{y} = \begin{bmatrix} y^1 \\ \vdots \\ y^m \end{bmatrix}$$

House ID	Lot Area (sq. feet)	# Bedrooms	Price
1	8450	2	208500
2	9600	3	181500

- Note, sometimes the superscript k is omitted

Linear Regression

Example: Predict Sale Prices from Lot Area



28 October 2020 IST PhD Open Days 15

Data: https://www.kaggle.com/c/house-prices-advanced-regression-techniques/overview

Linear Regression

Split Data

- Data needs to represent the **real world**
- Split into **training** and **test** sets
- Learn **hypothesis** using training set
- Evaluate using the test set
- This is one of the **most important steps**!

More information:

- https://en.wikipedia.org/wiki/Independent_and_identically_distributed_random_variables
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Linear Regression
Hypothesis Definition

$$h_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Linear Regression

Hypothesis Definition

$$h_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

parameters or **weights**

Linear Regression
Hypothesis Definition

$$h_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

parameters or **weights**
input or **features**

Linear Regression
Hypothesis Definition

$$h_\theta(\boldsymbol{x}) = \underline{\theta_0} + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

parameters or **weights**
input or **features**
bias or **intercept**

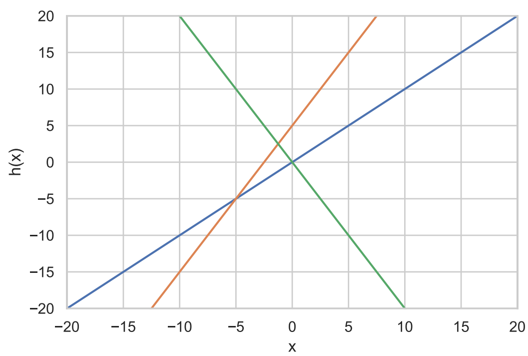
Linear Regression
Hypothesis Definition

$$h_\theta(x) = \underline{\theta_0 x_0} + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

parameters or **weights**
input or **features**
bias or **intercept**
Extra feature (always 1)

Linear Regression
Hypothesis Definition

$$h_\theta(\boldsymbol{x}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$



- $\theta_0 = 0, \theta_1 = 1$
- $\theta_0 = 5, \theta_1 = 2$
- $\theta_0 = 0, \theta_1 = \text{-}2$

Linear Regression
Hypothesis Definition

$$h_\theta(\boldsymbol{x}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

$$\Longleftrightarrow$$

$$h_\theta(\boldsymbol{x}) = \sum_{i=0}^{n} \theta_i x_i = [\theta_0 \quad \cdots \quad \theta_n] \begin{bmatrix} x_0 \\ \vdots \\ x_n \end{bmatrix} = \boldsymbol{\theta}^T \boldsymbol{x}$$

- Core idea: use **parameters** to map linearly from **features** to target

Linear Regression

Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$

$$J_{\theta}(\boldsymbol{X}, \boldsymbol{y}) = \frac{1}{2}\sum_{k=1}^{m}\left(h_{\theta}\left(\boldsymbol{x}^{k}\right) - y^{k}\right)^{2}$$

- Note: we use x^k and y^k which is the same as $\boldsymbol{X}_{k,:}$ and $\boldsymbol{y}_k$ if you index the matrix/vector

Linear Regression
Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$

$$J_\theta(\boldsymbol{X}, \boldsymbol{y}) = \frac{1}{2}\sum_{k=1}^{m}\left(h_\theta\left(\boldsymbol{x}^k\right) - y^k\right)^2$$

Cost or Loss function

Linear Regression
Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$

$$J_\theta(\boldsymbol{X}, \boldsymbol{y}) = \frac{1}{2}\sum_{k=1}^{m}\left(h_\theta\left(\boldsymbol{x}^k\right) - y^k\right)^2$$

Cost or Loss function
Hypothesis for x^k

Linear Regression
Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$

$$J_\theta(\boldsymbol{X}, \boldsymbol{y}) = \frac{1}{2}\sum_{k=1}^{m}\left(h_\theta\left(\boldsymbol{x}^k\right) - y^k\right)^2$$

Cost or Loss function

Hypothesis for x^k

Real result for x^k (target)

Linear Regression
Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$

$$J_\theta(\boldsymbol{X}, \boldsymbol{y}) = \frac{1}{2}\sum_{k=1}^{m}\underline{\left(h_\theta\left(\boldsymbol{x}^k\right) - y^k\right)^2}$$

Cost or Loss function

Hypothesis for x^k

Real result for x^k (target)

Squared error

Linear Regression

Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$

$$J_{\theta}(\boldsymbol{X}, \boldsymbol{y}) = \frac{1}{2}\sum_{k=1}^{m}\underline{\left(h_{\theta}\left(\boldsymbol{x}^{k}\right) - y^{k}\right)^{2}}$$

Cost or Loss function

Hypothesis for x^k

Real result for x^k (target)

Squared error

- Core idea: **cost** depends on **error** of the **hypothesis** given the **target**

Linear Regression
Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$
  2. Minimize cost on training data with gradient descent

Note: Instead of gradient descent, you can directly solve for the gradient being 0. For more information check pages 9 and 10 of these lecture notes: http://cs229.stanford.edu/notes2020fall/notes2020fall/cs229-notes1.pdf

Linear Regression
Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$
  2. Minimize cost on training data with gradient descent

$$\theta_i \coloneqq \theta_i - \alpha \frac{\partial}{\partial \theta_i} J_\theta(\boldsymbol{X}, \boldsymbol{y})$$

Linear Regression
Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$
  2. Minimize cost on training data with gradient descent

$$\theta_i \coloneqq \theta_i - \alpha \frac{\partial}{\partial \theta_i} J_\theta(\boldsymbol{X}, \boldsymbol{y})$$

Linear Regression

Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$
  2. Minimize cost on training data with gradient descent

$$\theta_i \coloneqq \theta_i - \alpha \frac{\partial}{\partial \theta_i} J_\theta(\boldsymbol{X}, \boldsymbol{y})$$

$$\Leftrightarrow$$

$$\theta_i \coloneqq \theta_i - \alpha \sum_{k=1}^{m} \left(h_\theta\left(x^k\right) - y^k\right) x_i$$

Linear Regression

Learning the Parameters

- How to Learn?
  1. Measure the quality of the hypothesis for $\boldsymbol{\theta}$
  2. Minimize cost on training data with gradient descent

$$\theta_i \coloneqq \theta_i - \alpha \frac{\partial}{\partial \theta_i} J_\theta(\boldsymbol{X}, \boldsymbol{y})$$

$$\Leftrightarrow$$

$$\theta_i \coloneqq \theta_i - \alpha \sum_{k=1}^{m} \left(h_\theta\left(x^k\right) - y^k\right) x_i$$

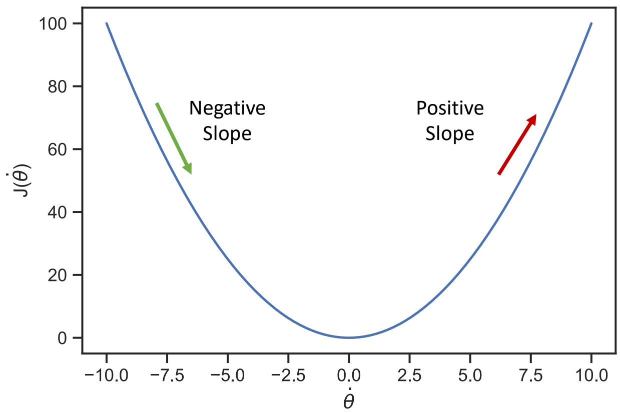
- Core idea: move **parameters** towards the direction of lower **cost**

28 October 2020 IST PhD Open Days 34

- Note: we run gradient descent for multiple epochs

Linear Regression

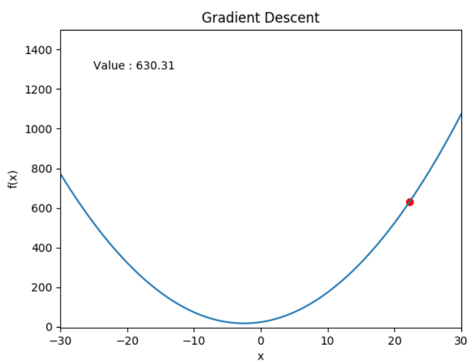
Gradient Descent



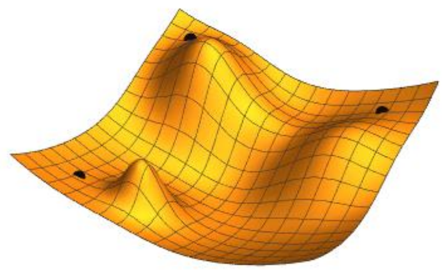
Slope subtraction: https://medium.com/@aerinykim/why-do-we-subtract-the-slope-a-in-gradient-descent-73c7368644fa

Linear Regression

Gradient Descent Examples



https://github.com/Shathra/gradient-descent-demonstration



https://commons.wikimedia.org/wiki/File:Gradient_descent.gif

Linear Regression
Checklist

1. Split data
2. Define hypothesis
3. Define cost function
4. Define learning algorithm
5. Train for multiple epochs

- At this stage, we move to the Linear Regression Jupyter Notebook

Side note: Polynomial Regression

The dangers of overfitting and the importance of validation

- Instead of Linear Regression, we can do Polynomial Regression!

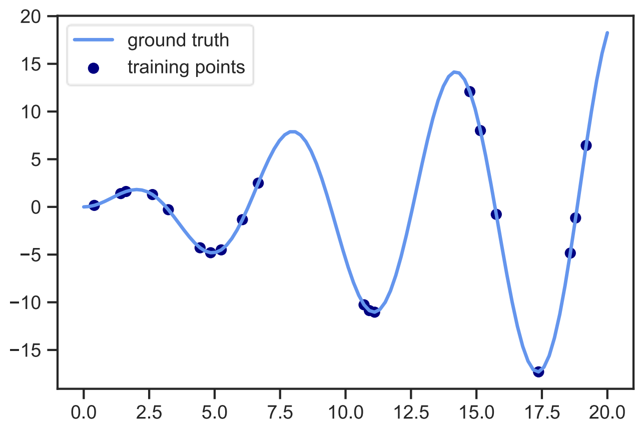
- Basic idea:
  - Compute polynomial combinations of features up to degree n
  - Apply Linear Regression using those features

- When should we stop?

- Polynomial combinations: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html

Side note: Polynomial Regression

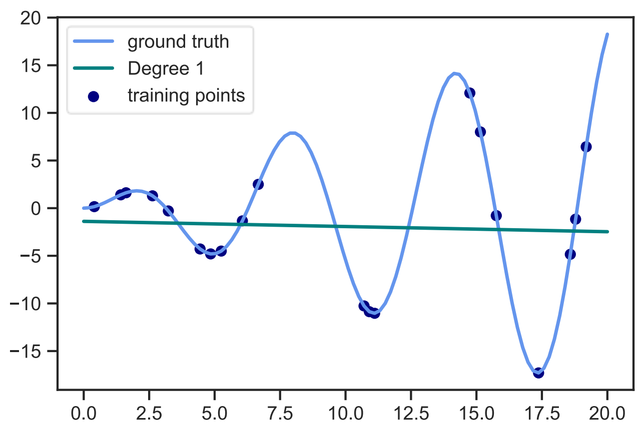
The dangers of overfitting and the importance of validation



Example adapted from: https://scikit-learn.org/stable/auto_examples/linear_model/plot_polynomial_interpolation.html#sphx-glr-auto-examples-linear-model-plot-polynomial-interpolation-py

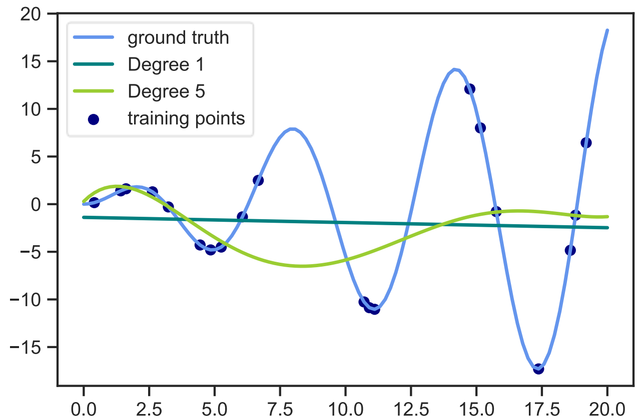
Side note: Polynomial Regression

The dangers of overfitting and the importance of validation



Side note: Polynomial Regression

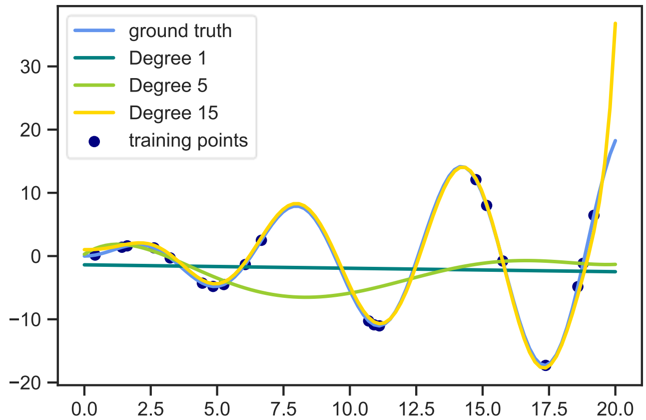
The dangers of overfitting and the importance of validation



- Note: these models are underfitting

Side note: Polynomial Regression

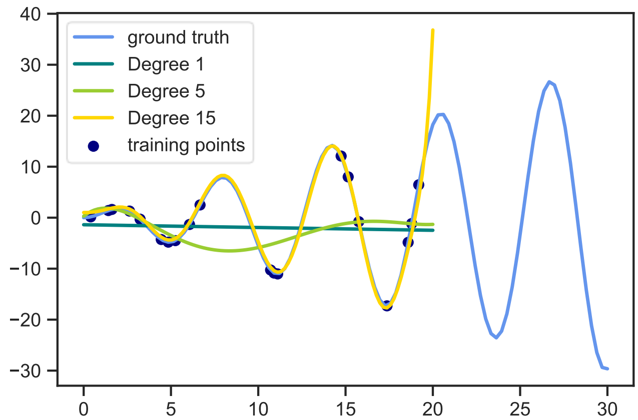
The dangers of overfitting and the importance of validation



- This is not good! Our new model has memorized the training data

Side note: Polynomial Regression

The dangers of overfitting and the importance of validation



- If we start going outside the training range, the result stops being consistent!

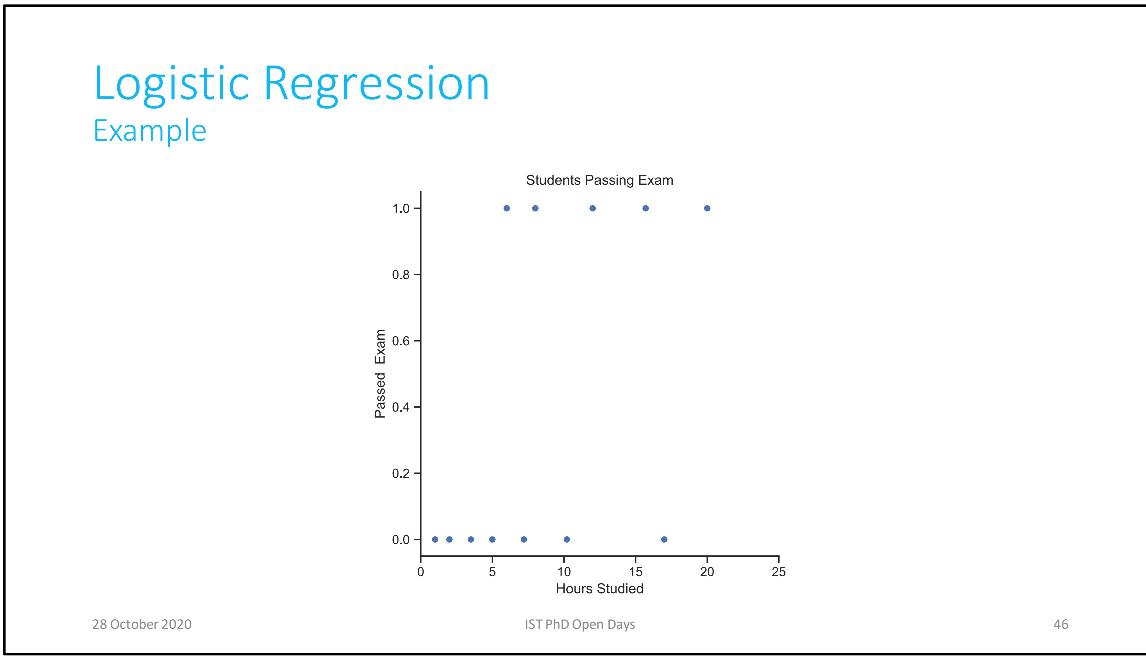
Side note: Polynomial Regression

The dangers of overfitting and the importance of validation

- A model performing well on training data can have poor results during test
- This indicates overfitting – it does not generalize to unseen data
- Start with simple baselines
- Always train and evaluate on a validation set
- Evaluate the final model using the real test set

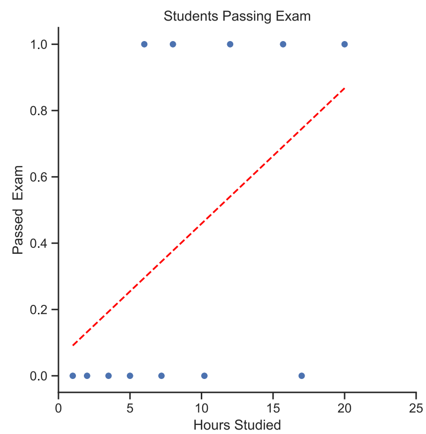
- Note: Regularization terms can be added to the cost function to prevent overfitting (check Lasso and Ridge regressions)
- Note: Evaluation and making decisions on a validation set avoids the problem of “training” on a test set by evaluating multiple different hypotheses on it

Questions?



- 0.0 = failed, 1.0 = passed
- We are still using a regression algorithm

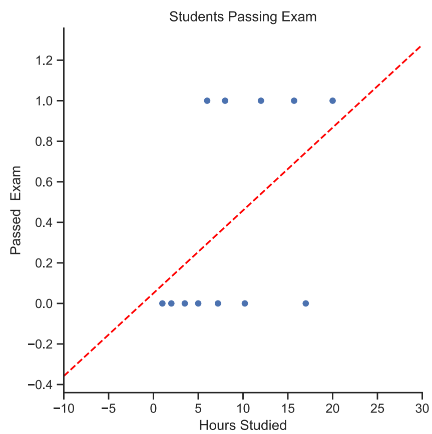
Logistic Regression
Example



28 October 2020 IST PhD Open Days 47

- 0.0 = failed, 1.0 = passed
- We are still using a regression algorithm!

Logistic Regression
Example



- Not restricted to {0, 1}!

Logistic Regression

Hypothesis Definition

- We want to restrict the result to $\{0, 1\}$

$$h_\theta(\boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x})$$

- Our definition of the hypothesis is based on applying a function on the original linear regression hypothesis

Logistic Regression
Hypothesis Definition

- We want to restrict the result to $\{0, 1\}$

$$h_\theta(\boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x})$$

Logistic Regression
Hypothesis Definition

- We want to restrict the result to $\{0, 1\}$

$$h_\theta(\boldsymbol{x}) = g(\underline{\boldsymbol{\theta}^T \boldsymbol{x}})$$

Logistic Regression

Hypothesis Definition

- We want to restrict the result to $\{0, 1\}$

$$h_\theta(\boldsymbol{x}) = g(\underline{\boldsymbol{\theta}^T \boldsymbol{x}})$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

- This function makes the result be non-linear wrt to the input

Logistic Regression
Hypothesis Definition

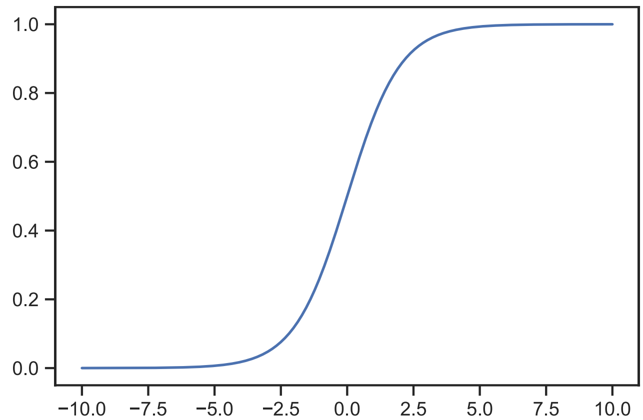
- We want to restrict the result to $\{0, 1\}$

$$h_{\theta}(\boldsymbol{x}) = g(\boldsymbol{\theta}^T \boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

Logistic Regression

Hypothesis Definition – the logistic function



- Logistic function squashes result to be between 0 and 1!
- But we want 0 or 1
- Note: this is also called the sigmoid function

Logistic Regression

Class Probabilities

- Hypothesis: $h_\theta(\boldsymbol{x}) = g(\boldsymbol{\theta}^T\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\theta}^T\boldsymbol{x}}}$
- Probability of passing: $P(y = 1|\boldsymbol{x};\boldsymbol{\theta}) = h_\theta(\boldsymbol{x})$
- Probability of failing: $P(y = 0|\boldsymbol{x};\boldsymbol{\theta}) = 1 - h_\theta(\boldsymbol{x})$
- Probability of y: $P(y|\boldsymbol{x};\boldsymbol{\theta}) = \left(h_\theta(\boldsymbol{x})\right)^y\left(1 - h_\theta(\boldsymbol{x})\right)^{1-y}$

- Regression over the class probabilities can be used for classification

Logistic Regression

Cost Function

- Probabilistic approach: Use the **likelihood**

$$L(\boldsymbol{\theta}) = p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\theta}) = \prod_{k=1}^{m} p\left(y^k \middle| \boldsymbol{x}^k;\boldsymbol{\theta}\right)$$

$$l(\boldsymbol{\theta}) = \log L(\boldsymbol{\theta}) = \sum_{k=1}^{m} y^k \log h_\theta\left(\boldsymbol{x}^k\right) + \left(1-y^k\right)\log\left(1-h_\theta\left(\boldsymbol{x}^k\right)\right)$$

- Likelihood is based on the independence of the training examples
- We compute likelihood of predicting the correct classes for the training dataset using the current parameters
- Note: the negative log likelihood is equal to the cross-entropy between the true distribution and our estimated distribution (h)

Logistic Regression
Gradient Descent

- We want to maximise the log likelihood
- Equal to minimising the negative log likelihood

$$\frac{\partial}{\partial \theta_i} \text{-} l(\boldsymbol{\theta}) = (h_\theta(\boldsymbol{x}^k) - y^k)x_i$$

$$\theta_i \coloneqq \theta_i - \alpha \sum_{k=1}^{m} (h_\theta(\boldsymbol{x}^k) - y^k)x_i$$

- Derivative of the negative log likelihood shown here is for a single training example
- The final update rule is the same as for Linear Regression but h is defined in a different way

Logistic Regression
Checklist

1. Split data
2. Define hypothesis
3. Define cost function
4. Define learning algorithm
5. Train for multiple epochs

- At this stage, we move to the Logistic Regression Jupyter Notebook
- For more, check: http://web.stanford.edu/~jurafsky/slp3/5.pdf

Side note: Softmax Regression

Generalize logistic regression to multiple classes

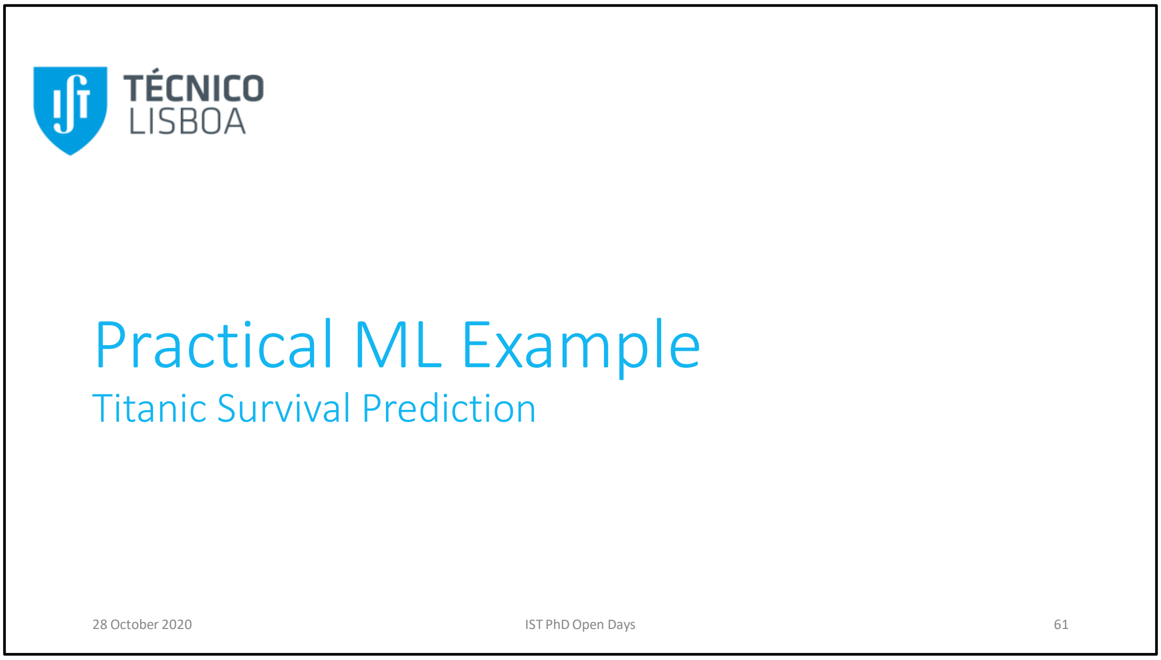
- Logistic regression can be extended to multiple classes

$$p(y = c|\boldsymbol{x}; \boldsymbol{\theta}_c) = \frac{e^{\boldsymbol{\theta}_c^T \boldsymbol{x}}}{\sum_{j=1}^{C} e^{\boldsymbol{\theta}_j^T \boldsymbol{x}}}$$

- Compute probability per class using class specific weights

- Softmax regression is also called multinomial logistic regression or the maxent classifier
- Need separate parameters per class (usually join them into a matrix)

Questions?



- At this time we move the Titanic Notebook

TÉCNICO LISBOA

Resources

28 October 2020 IST PhD Open Days 62

- Note that all resources shown here are free and available online

Workshop Topics

- Linear Regression, Logistic Regression, Generalized Linear Models:
  - http://cs229.stanford.edu/notes2020fall/notes2020fall/cs229-notes1.pdf

- Logistic Regression
  - http://web.stanford.edu/~jurafsky/slp3/5.pdf

Math Foundations

- Mathematics for Machine Learning Book:
  - https://mml-book.github.io/book/mml-book.pdf
- First part of the Deep Learning Book
  - https://www.deeplearningbook.org/

Introduction to Machine Learning

- CS229 Stanford Lectures:
  - http://cs229.stanford.edu/
  - https://www.youtube.com/playlist?list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU

- Coursera course (also by Andrew Ng):
  - https://www.coursera.org/learn/machine-learning

Deep Learning

- Deep Learning Book:
  - https://www.deeplearningbook.org/

- Deep Learning for Natural Language Processing:
  - http://web.stanford.edu/class/cs224n/
  - https://www.youtube.com/playlist?list=PLoROMvodv4rOhcuXMZkNm7j3fVwBBY42z

- Deep Learning for Computer Vision:
  - http://cs231n.stanford.edu/
  - https://www.youtube.com/playlist?list=PLC1qU-LWwrF64f4QKQT-Vg5Wr4qEE1Zxk