Introduction to Machine Learning

Lecture 3: Regression

Alexis Zubiolo
alexis.zubiolo@gmail.com

Data Science Team Lead @ Adcash

November 3, 2016

Before we start

Would you be interested in **a more advanced course**? I can propose

- Machine learning **from scratch** (how to implement an ML algorithm with no library)
- A **more advanced version** of this course (with more theoretical technical details)
- **Large-scale** machine learning (distributed computing)

Regression in Machine Learning

This lecture is about **regression** in Machine learning.

Reminder: In regression, the output y is **continous**.

Example:

- **Price estimation**: $y =$ price (*e.g.* 50000 BGN for a house)
- **Predicting the future** (*e.g.* weather forecast): $y =$ temperature or amount of rain

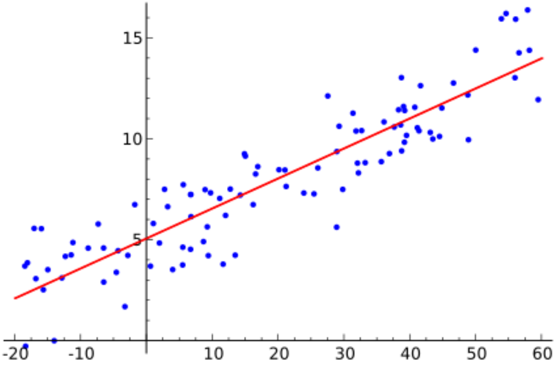
Regression in Machine Learning: Applications

Domains of application:

- Price estimation/prediction
- Weather forecast
- Production quantity estimation
- Stock option price prediction
- Fit statistical model to data
- Physics & chemistry
- ... and others

Linear and polynomial regression

Purpose of regression: **approximate solutions** of **overdetermined systems**.



In this course, we will see

- Linear regression
- Polynomial regression

Linear regression

Linear regression

Principal components:

- Old problem (least-squares method usually credited to Carl Friedrich Gauss in 1795)

Linear regression

Principal components:

- Old problem (least-squares method usually credited to Carl Friedrich Gauss in 1795)
- Several ways to approximate the data
  - Linear model
  - Polynomial model (remember kernels from SVMs)
  - Fit a distribution
  - ...

Linear regression

Principal components:

- Old problem (least-squares method usually credited to Carl Friedrich Gauss in 1795)
- Several ways to approximate the data
  - Linear model
  - Polynomial model (remember kernels from SVMs)
  - Fit a distribution
  - ...
- Several ways to formulate the problem
  - Least Squares
  - Support Vector regression
  - ...

Linear regression

Principal components:

- Old problem (least-squares method usually credited to Carl Friedrich Gauss in 1795)
- Several ways to approximate the data
  - Linear model
  - Polynomial model (remember kernels from SVMs)
  - Fit a distribution
  - ...
- Several ways to formulate the problem
  - Least Squares
  - Support Vector regression
  - ...
- Several ways to solve the problem
  - Closed-form expression (exact formula)
  - Optimization

Linear regression: Toy example

living area (m^2)	**# bedrooms**	price (1000's euros)
50	**1**	30
76	**2**	48
26	**1**	12
102	**3**	90

Linear regression: Toy example

living area (m^2)	**# bedrooms**	price (1000's euros)
50	**1**	30
76	**2**	48
26	**1**	12
102	**3**	90
61	**2**	?

Linear regression: Toy example

living area (m^2)	**# bedrooms**	price (1000's euros)
50	**1**	30
76	**2**	48
26	**1**	12
102	**3**	90
61	**2**	?

Linear model: price $=$ **a** $\times$ area $+$ **b** $\times$ $\#$ bedrooms $+$ **c**

Linear regression: Toy example

living area (m^2)	**# bedrooms**	price (1000's euros)
50	**1**	30
76	**2**	48
26	**1**	12
102	**3**	90
61	**2**	?

Linear model: price $=$ **a** $\times$ area $+$ **b** $\times$ $\#$ bedrooms $+$ **c**

Problem: optimal values for **a**, **b** and **c**?

Linear regression: Toy example

living area (m^2)	**# bedrooms**	price (1000's euros)
50	**1**	30
76	**2**	48
26	**1**	12
102	**3**	90
61	**2**	?

Linear model: price $= \mathbf{a} \times$ area $+ \mathbf{b} \times \#$ bedrooms $+ \mathbf{c}$

Problem: optimal values for **a**, **b** and **c**?

General formulation:

$$\hat{y} = w^T x$$

Linear regression with ordinary least-squares

Linear regression: Estimate y as a **linear** function of x:

$$\hat{y} = w^T x$$

Linear regression with ordinary least-squares

Linear regression: Estimate y as a **linear** function of x:

$$\hat{y} = w^T x$$

Least squares: Penalty (loss) is a **quadratic** function

$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

Regression formulation

2 main ways to solve the linear least-squares problem:

$$\min_w \sum_i \left(w^T x^{(i)} - y^{(i)} \right)^2 \tag{1}$$

Method 1: Closed-form expression

$$w = \left(X^T X \right)^{-1} X^T Y$$

It can be compuationally expensive (matrix multiplication, matrix inversion, matrix multiplication, matrix-vector multiplication)

Method 2: Numerical optimization For example gradient descent algorithms, ...

Polynomial regression

Polynomial regression ≈ Kernel trick

Remind the kernel trick from the SVM lecture.

Example:

$$x = (x_1, x_2)$$

can become

$$\phi(x) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x_2)$$

with a **second order kernel**.

Then we can find w solution of

$$\min_w \sum_i \left(w^T \phi\left(x^{(i)}\right) - y^{(i)}\right)^2 \tag{2}$$

Practical information

Variable standardization

Variables have various magnitudes. Example:

- Living area: Up to a few hundreds m^2
- Price: Up to a few 100 000s BGN (and even more)
- # bedrooms: usually much smaller than 10

This can be an issue when training a regression model.

Variable standardization

Variables have various magnitudes. Example:

- Living area: Up to a few hundreds m^2
- Price: Up to a few 100 000s BGN (and even more)
- # bedrooms: usually much smaller than 10

This can be an issue when training a regression model.
It is possible to calculate the **standard score** z of a variable x

$$z = \frac{x - \mu}{\sigma}$$

where

- μ is the mean of the variable
- σ is its standard deviation

Variable standardization

Variables have various magnitudes. Example:

- Living area: Up to a few hundreds m^2
- Price: Up to a few 100 000s BGN (and even more)
- # bedrooms: usually much smaller than 10

This can be an issue when training a regression model.
It is possible to calculate the **standard score** z of a variable x

$$z = \frac{x - \mu}{\sigma}$$

where

- μ is the mean of the variable
- σ is its standard deviation

Another option: Scale between 0 and 1

$$z = \frac{x - \min}{\max - \min}$$

Overfitting and underfitting

Illustration on a generated example: Try to fit the function

$$y = f(x) = \cos\left(\frac{3\pi}{2}x\right) + \text{noise}$$

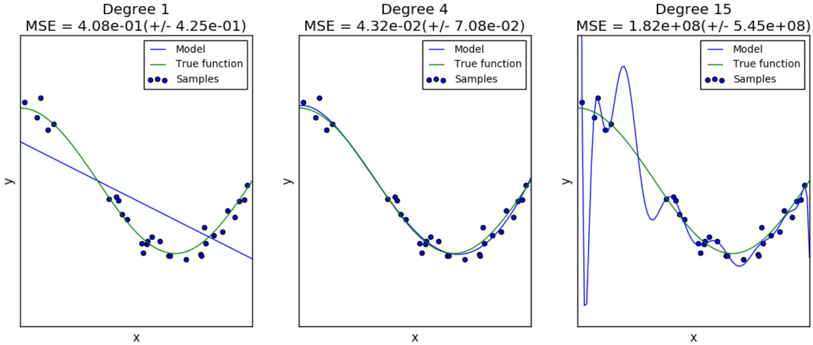
for $x \in [0, 1]$, with a polynomial regression

Overfitting and underfitting

Illustration on a generated example: Try to fit the function

$$y = f(x) = \cos\left(\frac{3\pi}{2}x\right) + \text{noise}$$

for $x \in [0, 1]$, with a polynomial regression



Parameter selection

As for classification models, parameter selection plays a key role in the regression performance:

- Degree of the polynomial
- Regularization parameter

Parameter selection

As for classification models, parameter selection plays a key role in the regression performance:

- Degree of the polynomial
- Regularization parameter

Optimal parameters can be chosen with cross-validation over a grid:

Parameter selection

As for classification models, parameter selection plays a key role in the regression performance:

- Degree of the polynomial
- Regularization parameter

Optimal parameters can be chosen with cross-validation over a grid:

- Split the data into train/test

Parameter selection

As for classification models, parameter selection plays a key role in the regression performance:

- Degree of the polynomial
- Regularization parameter

Optimal parameters can be chosen with cross-validation over a grid:

- Split the data into train/test
- Choose a degree $d \in \{1, \dots, 20\}$

Parameter selection

As for classification models, parameter selection plays a key role in the regression performance:

- Degree of the polynomial
- Regularization parameter

Optimal parameters can be chosen with cross-validation over a grid:

- Split the data into train/test
- Choose a degree $d \in \{1, \ldots, 20\}$
- Train on the train set with this degree

Parameter selection

As for classification models, parameter selection plays a key role in the regression performance:

- Degree of the polynomial
- Regularization parameter

Optimal parameters can be chosen with cross-validation over a grid:

- Split the data into train/test
- Choose a degree $d \in \{1, \ldots, 20\}$
- Train on the train set with this degree
- Test the model on the test set

Parameter selection

As for classification models, parameter selection plays a key role in the regression performance:

- Degree of the polynomial
- Regularization parameter

Optimal parameters can be chosen with cross-validation over a grid:

- Split the data into train/test
- Choose a degree $d \in \{1, \ldots, 20\}$
- Train on the train set with this degree
- Test the model on the test set

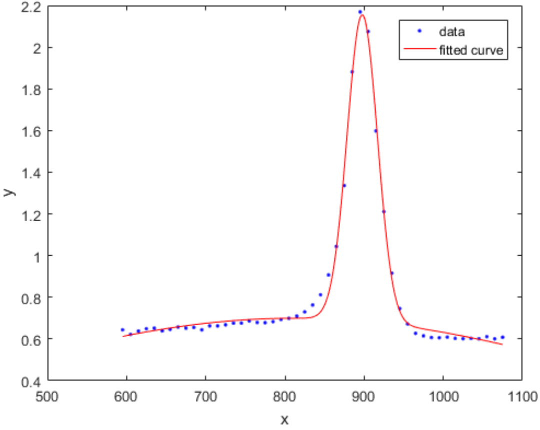
It can be done over several train/test splits.

Toy example: Fitting a distribution

Find A, x_0 and σ such that

$$\hat{y} = f(x) = Ae^{\dfrac{(x - x_0)^2}{2\sigma^2}}$$

best fits the data in terms of least-square error.



Regression with SVMs

Alternatives to least squares

It is possible to use a different loss function ℓ. Remember, we had

$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

Alternatives to least squares

It is possible to use a different loss function ℓ. Remember, we had

$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

We can use **support vector machines for regression** (SVR):

- If **within the margin** (*i.e.* $-\epsilon \leq \hat{y} - y \leq +\epsilon$) then **no penalty**
- linear or quadratic **penalty outside the margin**

(see flip-chart for illustration)

This loss function is called ϵ-insensitive.

Alternatives to least squares

It is possible to use a different loss function ℓ. Remember, we had

$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

We can use **support vector machines for regression** (SVR):

- If **within the margin** (*i.e.* $-\epsilon \leq \hat{y} - y \leq +\epsilon$) then **no penalty**
- linear or quadratic **penalty outside the margin**

(see flip-chart for illustration)

This loss function is called ϵ-insensitive.

Note: We can use kernels as for SVM

Conclusion

Regression: Output y is a **continuous variable**.

Conclusion

Regression: Output y is a **continuous variable**.

Several ways to **penalize errors**:

- Least squares
- Support vector regressions

Conclusion

Regression: Output y is a **continuous variable**.

Several ways to **penalize errors**:

- Least squares
- Support vector regressions

Several ways to **model the prediction**:

- Linear
- Quadratic
- Other kernel

Conclusion

Regression: Output y is a **continuous variable**.

Several ways to **penalize errors**:

- Least squares
- Support vector regressions

Several ways to **model the prediction**:

- Linear
- Quadratic
- Other kernel

Parameter selection is important

Thank you! Questions?