Bayesian model selection for linear regression

Universität Augsburg
Institut für Mathematik

Philipp Düren

Augsburg, 22nd January 2015

Miguel de Benito

Python

```
tt = generator.generate(xx)
pl.clf()
pl.plot(xx, tt, 'ro')
fig = pl.gcf()
ps_out(fig)
```

Busy...

- What fits best?

- What fits best?
- Polynomial of order ...? Sines and cosines? Wavelets?

Questions

- What fits best?
- Polynomial of order ...? Sines and cosines? Wavelets?
- How to choose among these?

(you may sleep now)

- Some examples.

- Some examples.
- We observe **data** x and **targets** t in a training set.

- Some examples.
- We observe **data** x and **targets** t in a training set.
- Assume

$$t = y(x, w) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

- Some examples.
- We observe **data** x and **targets** t in a training set.
- Assume

$$t = y(x, w) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

- Where

$$y(x, w) = \sum_{j=0}^{M_k - 1} w_j \phi_j(x),$$

and

$$\phi_j = \phi_j^{(k)} \in \mathcal{H}_k = \{\phi_0^{(k)}, \ldots, \phi_{M_k - 1}^{(k)}\}, M_k \in \mathbb{N}$$

- We start with:

- We start with:
- A uniform prior on the models:

$$p(\mathcal{H}_k) = \frac{1}{K}.$$

- We start with:
- A uniform prior on the models:

$$p(\mathcal{H}_k) = \frac{1}{K}.$$

- A Gaussian prior on the parameters (for every model):

$$p(w|\mathcal{H}_k) \sim \mathcal{N}(0, \sigma_w^2 \,\mathrm{Id}).$$

- We start with:
- A uniform prior on the models:

$$p(\mathcal{H}_k) = \frac{1}{K}.$$

- A Gaussian prior on the parameters (for every model):

$$p(w|\mathcal{H}_k) \sim \mathcal{N}(0, \sigma_w^2 \,\mathrm{Id}).$$

- We want to compute:

$$p(\mathcal{H}_k|\boldsymbol{t}) = \frac{p(\boldsymbol{t}|\mathcal{H}_k)\, p(\mathcal{H}_k)}{p(\boldsymbol{t})} = \frac{\int_W p(\boldsymbol{t}|w, \mathcal{H}_k)\, p(w|\mathcal{H}_k)\, \mathrm{d}w}{p(\boldsymbol{t})}\, p(\mathcal{H}_k).$$

- To compute the **evidence** for $\mathcal{H}_k$:

$$p(\boldsymbol{t}|\mathcal{H}_k) = \int_W p(\boldsymbol{t}|w, \mathcal{H}_k)\, p(w|\mathcal{H}_k)\, \mathrm{d}w,$$

- To compute the **evidence** for $\mathcal{H}_k$:

$$p(\boldsymbol{t}|\mathcal{H}_k) = \int_W p(\boldsymbol{t}|w, \mathcal{H}_k)\, p(w|\mathcal{H}_k)\, \mathrm{d}w,$$

- we use Laplace's approximation:

$$\int_W p(\boldsymbol{t}|w, \mathcal{H}_k)\, p(w|\mathcal{H}_k)\, \mathrm{d}w \approx \frac{p(\boldsymbol{t}|w_{\mathrm{MAP}}, \mathcal{H}_k)\, p(w_{\mathrm{MAP}}|\mathcal{H}_k)}{\sqrt{\det(A/2\,\pi)}},$$

- To compute the **evidence** for $\mathcal{H}_k$:

$$p(\boldsymbol{t}|\mathcal{H}_k) = \int_W p(\boldsymbol{t}|w, \mathcal{H}_k)\, p(w|\mathcal{H}_k)\, \mathrm{d}w,$$

- we use Laplace's approximation:

$$\int_W p(\boldsymbol{t}|w, \mathcal{H}_k)\, p(w|\mathcal{H}_k)\, \mathrm{d}w \approx \frac{p(\boldsymbol{t}|w_{\mathrm{MAP}}, \mathcal{H}_k)\, p(w_{\mathrm{MAP}}|\mathcal{H}_k)}{\sqrt{\det(A/2\,\pi)}},$$

- which in the case of Gaussians is **exact**.

```
>>> from Hypotheses import *
    from ModelSelection import LinearRegression
    from Test import generate_noise_and_fit
    ###### Random data:
    sigma = 5  # observation noise sigma
    #generator = TrigonometricHypothesis(halfM=2, variance=4,
    #                                    noiseVariance=sigma**2)
    generator = PolynomialHypothesis(M=3, variance=3, noiseVariance=sigma**2)
    ###### Our hypotheses:
    hc = HypothesisCollection()
    hc.append(PolynomialHypothesis(M=1, variance=3, noiseVariance=sigma**2))
    hc.append(PolynomialHypothesis(M=2, variance=3, noiseVariance=sigma**2))
    hc.append(PolynomialHypothesis(M=3, variance=3, noiseVariance=sigma**2))
    hc.append(TrigonometricHypothesis(halfM=4, variance=2,
              noiseVariance=sigma**2))
    hc.append(TrigonometricHypothesis(halfM=2, variance=2,
              noiseVariance=sigma**2))
    ###### Now choose the best one:
    lr = LinearRegression(hc, sigma)
    generate_noise_and_fit(lr, generator, xmin=-1.0, xmax=6, num=30)
```

```
>>> from Hypotheses import *
    from ModelSelection import LinearRegression
    from Test import generate_noise_and_fit
    ###### Random data:
    sigma = 5  # observation noise sigma
    #generator = TrigonometricHypothesis(halfM=2, variance=4,
    #                                     noiseVariance=sigma**2)
    generator = PolynomialHypothesis(M=3, variance=3, noiseVariance=sigma**2)
    ###### Our hypotheses:
    hc = HypothesisCollection()
    hc.append(PolynomialHypothesis(M=1, variance=3, noiseVariance=sigma**2))
    hc.append(PolynomialHypothesis(M=2, variance=3, noiseVariance=sigma**2))
    hc.append(PolynomialHypothesis(M=3, variance=3, noiseVariance=sigma**2))
    hc.append(TrigonometricHypothesis(halfM=4, variance=2,
              noiseVariance=sigma**2))
    hc.append(TrigonometricHypothesis(halfM=2, variance=2,
              noiseVariance=sigma**2))
    ###### Now choose the best one:
    lr = LinearRegression(hc, sigma)
    generate_noise_and_fit(lr, generator, xmin=-1.0, xmax=6, num=30)
Update completed in 98 milliseconds.
```

- You can find it at github.com/mdbenito/ModelSelection.
- Things to do...

EOT

⟨*insert-picture-of-cute-baby-animal*⟩

```
>>> import sys
>>> sys.path.extend(['/Users/miguel/Devel/ML/ModelSelection/src'])
>>> from Hypotheses import *
    import matplotlib.pyplot as pl
>>> sigma = 5
    generator=PolynomialHypothesis(M=3, variance=5, noiseVariance=sigma**2)
    xx = np.array(np.arange(-1.0, 3.0, step=0.04))
>>>
```