COMPSCI 591/691NR
Neural Networks and Neurodynamics

1/22/2020

Chris Bishop, PRML

PATTERN RECOGNITION
AND MACHINE LEARNING

CH 1&2: INTRO TO PROBABILITY DENSITY ESTIMATION
Example

Handwritten Digit Recognition

0  /  1  /  2  /  3  /  4

5  /  6  /  7  /  8  /  9
Character Recognition

Handwritten characters: a and b
Examples in the data set (e.g. 50 of each)
Classification task (2 classes)

Humans: easy by *observing*

Computers:
  Require encoding into numerical strings
  Develop a classification algorithm (parametric or non-parametric)
  Convert high-dimensional data space into low-dim feature space
  (see next.)
Parametric Classification by machines - example

Calculate the aspect ratio

\[ R = \frac{\text{height}}{\text{width}} \]

Modeling **assumption:** \( R(b) > R(a) \)

It’s true in most of the cases (not always!)

Perform classification based on the model

Needs a decision criterion

Calculate error rate of classification

\[ C = \frac{\# \text{ misclassified}}{\# \text{ total samples}} \]
Classification (cont’d)

Develop a good classifier

With low error rate C

E.g., calculate class centers and use decision surface at the middle line between them

We will see this approach is optimal in some sense

Expecting good results for new examples

--> good GENERALIZATION !!

Not easy
Higher dimensional (dim>1) feature spaces

Decision along not just a threshold but a surface: a subspace of co-dimension 1 in the feature space

Can be linear discriminant

In this case the decision surface is a (high-dimensional) plane

More complicated: nonlinear discriminant

(statistical model, or model-free NN)
Classification vs prediction

The two main problems in NN studies
Now we concentrate on classification
Later discuss regression and prediction
Not independent problem
Preprocessing

Feature extraction

Eg normalization

Data compression (decrease dimensionality)

Get rid off drift and other unwanted effects

Be aware!! Don’t destroy the data

You may think something is not important but in fact it is!!

Preprocessing often a matter of ‘art’ and it is difficult to give general rules

Must think!! Proper preprocessing is often a key to successful classification...
Statistical Pattern Recognition versus NNs

**Statistical classifier (feature extraction)**

- Input (high dim)
- Preprocessing (normalize, …)

**Neural network classifier**

- Compression
- NN

**Output (class)**
- Yes/No / a or b etc.

```
0111010101000010101000100001111100101

111101010100001010100010011000111000

01010000011110001010

0 / 1

Stat. Discr. Bayes
```
Example: Curve fitting

Given: x and t vectors
   Eg, x=1:100
   t function is known at given (time) instances

Goal: fit this curve and get y at any value
   (not only the given 100 points)

Expect good results in apices x(i): y(x(i)) \approx t(i)
   Error function: sum of squared errors!!
   Also desired good approximation at points in between interpolation and also extrapolation
Polynomial Curve Fitting

\[ y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j \]
Error Function

Sum of squared errors
- It is a metric for the goodness of the model
- Distance in the Euclidean space (N dim)

We want to adjust the parameters of the model
- (eg, coeff’s of the polynomial) that minimize this error

NB: there are more complicated error functions
- but SSE is optimal in some sense (see later)
Sum-of-Squares Error Function

\[ E(w) = \frac{1}{2} \sum_{n=1}^{N} \left[ y(x_n, w) - t_n \right]^2 \]
Generalization

Optimum model complexity!!!

By following too closely the training data
you won’t be able to perform well in new data
--> meaning bad generalization

You don’t want to learn the noise
noise is inherently present in life
Eg, handwriting: slipping of pen, tired, etc...
$0^{th}$ Order Polynomial
$1^{\text{st}}$ Order Polynomial

\[ M = 1 \]
3rd Order Polynomial

\[ M = 3 \]
9th Order Polynomial

\[ M = 9 \]
Over-fitting

Root-Mean-Square (RMS) Error: \( E_{RMS} = \sqrt{2E(w^*)/N} \)
### Polynomial Coefficients

<table>
<thead>
<tr>
<th></th>
<th>$M = 0$</th>
<th>$M = 1$</th>
<th>$M = 3$</th>
<th>$M = 9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_0^*$</td>
<td>0.19</td>
<td>0.82</td>
<td>0.31</td>
<td>0.35</td>
</tr>
<tr>
<td>$w_1^*$</td>
<td></td>
<td>-1.27</td>
<td>7.99</td>
<td>232.37</td>
</tr>
<tr>
<td>$w_2^*$</td>
<td></td>
<td></td>
<td>-25.43</td>
<td>-5321.83</td>
</tr>
<tr>
<td>$w_3^*$</td>
<td></td>
<td></td>
<td>17.37</td>
<td>48568.31</td>
</tr>
<tr>
<td>$w_4^*$</td>
<td></td>
<td></td>
<td></td>
<td>-231639.30</td>
</tr>
<tr>
<td>$w_5^*$</td>
<td></td>
<td></td>
<td></td>
<td>640042.26</td>
</tr>
<tr>
<td>$w_6^*$</td>
<td></td>
<td></td>
<td></td>
<td>-1061800.52</td>
</tr>
<tr>
<td>$w_7^*$</td>
<td></td>
<td></td>
<td></td>
<td>1042400.18</td>
</tr>
<tr>
<td>$w_8^*$</td>
<td></td>
<td></td>
<td></td>
<td>-557682.99</td>
</tr>
<tr>
<td>$w_9^*$</td>
<td></td>
<td></td>
<td></td>
<td>125201.43</td>
</tr>
</tbody>
</table>
Data Set Size: $N = 15$

9$^\text{th}$ Order Polynomial
Data Set Size: $N = 100$

9$^\text{th}$ Order Polynomial
Regularization

Penalize large coefficient values

\[ \widetilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \| \mathbf{w} \|^2 \]
Regularization: $\ln \lambda = -18$
Regularization: \( \ln \lambda = 0 \)
Regularization: $E_{RMS}$ vs. $\ln \lambda$
## Polynomial Coefficients

<table>
<thead>
<tr>
<th>( w_i^* )</th>
<th>( \ln \lambda = -\infty )</th>
<th>( \ln \lambda = -18 )</th>
<th>( \ln \lambda = 0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( w_0^* )</td>
<td>0.35</td>
<td>0.35</td>
<td>0.13</td>
</tr>
<tr>
<td>( w_1^* )</td>
<td>232.37</td>
<td>4.74</td>
<td>-0.05</td>
</tr>
<tr>
<td>( w_2^* )</td>
<td>-5321.83</td>
<td>-0.77</td>
<td>-0.06</td>
</tr>
<tr>
<td>( w_3^* )</td>
<td>48568.31</td>
<td>-31.97</td>
<td>-0.05</td>
</tr>
<tr>
<td>( w_4^* )</td>
<td>-231639.30</td>
<td>-3.89</td>
<td>-0.03</td>
</tr>
<tr>
<td>( w_5^* )</td>
<td>640042.26</td>
<td>55.28</td>
<td>-0.02</td>
</tr>
<tr>
<td>( w_6^* )</td>
<td>-1061800.52</td>
<td>41.32</td>
<td>-0.01</td>
</tr>
<tr>
<td>( w_7^* )</td>
<td>1042400.18</td>
<td>-45.95</td>
<td>-0.00</td>
</tr>
<tr>
<td>( w_8^* )</td>
<td>-557682.99</td>
<td>-91.53</td>
<td>0.00</td>
</tr>
<tr>
<td>( w_9^* )</td>
<td>125201.43</td>
<td>72.68</td>
<td>0.01</td>
</tr>
</tbody>
</table>
The Rules of Probability

Sum Rule

\[ p(X) = \sum_Y p(X, Y) \]

Product Rule

\[ p(X, Y) = p(Y|X)p(X) \]
Bayes Theorem

**Probability:**

\[ P(A) \] - the probability of occurrence of A

- Head/tail: 1/2, dice: 1/6, etc.

**Conditional probability:**

\[ P(A|B) \] - probability of A, assuming B

- \[ P(3|\text{uneven})=1/3 \]

**Bayes formula:**

\[ P(A|B)P(B) = P(B|A)P(A) \]

Based on joint probabilities
Interpretation of Bayes Theorem

X - sample example
C_i - denotes the i-th class (i=1, 2)

\[ P(C_i | X) = \frac{P(X | C_i)P(C_i)}{P(X)} \]

- \( P(C_i | X) \) - posterior probability of X in class i
- \( P(X | C_i) \) - class-conditional probability
- \( P(C_i) \) - prior probability of class C_i
- \( P(X) \) - probability of observing X in general

**Classification task**: determine \( P(C_i | X) \) !!
Classification: using Bayes

Determine or estimate RHS probabilities
Class-conditional, prior, probability of the event
Use Bayes to evaluate posterior probability
Decision criteria/ discriminant function:

Assign class \( k \) if \( P(C_k | X) > P(C_j | X) \) valid for all \( j \neq k \)

Maximum likelihood discriminant function
Bayes’s decision criterion/rule

Q: what is the best decision surface in a general nonlinear classification problem?

Two steps:
1. **Inference**: determine the ‘posterior probabilities’ $P(C_k|X)$; $k=1,2$ (2 classes)
2. **Decision**: assign the actual input $X$ to one of the classes

Under certain plausible statistical conditions choosing the max posterior prob is the best as it minimizes the classification error $SSE$
Implementing the Bayes optimal decision rule in NNs

Assume NN is a black box
Input X
Output: approximates posterior probability!!

Classification:
Choose the output class node with the max value and that is the best choice!

This is the beauty of NNs!
BUT: be aware: there is no gain without pain...
Probability Density Estimation Methods

1. Parametric methods
   - Normal distribution
   - Determine parameters: using maximum likelihood
   - Inference in Bayesian approach

2. Non-parametric methods
   - Histograms
   - Kernel methods
   - K nearest neighbors

3. Semi-parametric
   - Mixture models (also NNs)
Advantages of Normal Distribution

1. Simple analytical form
2. It is common in practice due to CLT
3. For any nonsingular linear transformation the distance remains positive definite and of quadratic form => remains normal distribution
4. Marginal densities (integrated over some variables] are normal
5. Conditional densities (fixing some variables] are normal
6. Exists a LINEAR transformation that diagonalizes covariance matrix , and in this new system the variables are independent (factorized to components].
7. Normal density maximizes entropy for some $\mu$ and $\Sigma$. 
The Gaussian Distribution

\[ \mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{ -\frac{1}{2\sigma^2}(x - \mu)^2 \right\} \]

\[ \mathcal{N}(x|\mu, \sigma^2) > 0 \]

\[ \int_{-\infty}^{\infty} \mathcal{N}(x|\mu, \sigma^2) \, dx = 1 \]
Gaussian Mean and Variance

\[ \mathbb{E}[x] = \int_{-\infty}^{\infty} \mathcal{N}(x|\mu, \sigma^2) x \, dx = \mu \]

\[ \mathbb{E}[x^2] = \int_{-\infty}^{\infty} \mathcal{N}(x|\mu, \sigma^2) x^2 \, dx = \mu^2 + \sigma^2 \]

\[ \text{var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \sigma^2 \]
The Multivariate Gaussian

\[
\mathcal{N}(x | \mu, \Sigma) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}
\]
Gaussian Parameter Estimation

\[
p(x) = \prod_{n=1}^{N} \mathcal{N}(x_n | \mu, \sigma^2)
\]
Discriminant Function for Gaussian

Our discriminant function is
\[ y_k = \log p(x|C_k) + \log P(C_k) \]
substitute the normal function
\[ y_k = (x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) - \frac{1}{2} \log |\Sigma_k| + \log P(C_k) \]

Assume:

Covariances are independent of k classes
Then 2\text{nd} term is class-independent and also quadratic
term in x is class-independent
Covariance and inverse is symmetric:
\[ y_k = W_k^T x + w_{k0} \quad -- \text{this is simple linear form}!!! \quad ...\text{derive!} \]
Voronoi tessallation
Here \( W_k^T = \mu_k^T \Sigma_k^{-1} \) and \( w_{k0} = -1/2 \mu_k^T \Sigma_k^{-1} \mu_k + \log P(C_k) \)
Prototypes/ Template Matching

If the following conditions are satisfied:

1. k-class problem with identical $\Sigma$ covariances
2. Independent variables (i.e., diagonal $\Sigma$)

We can drop the constant term of discriminant

$$y_k = -\|x-\mu_k\|^2/2\sigma^2 + \log P(C_k)$$

**Discriminant function**

Nearest class center will be chosen!

As measured by Euclidean distance
Model optimization:
Maximum Likelihood vs Bayesian Inference

Maximum likelihood:
Maximize the value of the likelihood function based on training data
E.g., $y_k$ as derived earlier

Bayesian Inference
Estimate the parameters of the probability density function
  Initial set prior distribution
  Use Bayes Theorem to convert to posterior distribution
  Final probability density: integrate over all possible values of parameters weighted by the posterior probability
Nonparametric Methods

Histogram

Calculate by dividing into n-bins

This is a simple estimation of probability distribution

   Can be normal distribution
   Or multi-modal normal, etc.

Assuming \( p(x) \) is approx constant

\[ p(x) = \frac{K}{NV} \]

N data points, K is within a region R, V is volume of R
Curve Fitting Re-visited

\[ y(x_0, w) \]

\[ p(t \mid x_0, w, \beta) = \mathcal{N}(t \mid y(x_0, w), \beta^{-1}) \]
Maximum Likelihood

\[ p(t|x, w, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n | y(x_n, w), \beta^{-1}) \]

\[ \ln p(t|x, w, \beta) = -\frac{\beta}{2} \sum_{n=1}^{N} (y(x_n, w) - t_n)^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E(w) \]

Determine \( w_{ML} \) by minimizing sum-of-squares error, \( E(w) \).

\[ \frac{1}{\beta_{ML}} = \frac{1}{N} \sum_{n=1}^{N} (y(x_n, w_{ML}) - t_n)^2 \]
Model Selection

Cross-Validation

run 1
run 2
run 3
run 4
Curse of Dimensionality
Decision Theory

Inference step

Determine either \( p(t|x) \) or \( p(x, t) \).

Decision step

For given \( x \), determine optimal \( t \).
Minimum Misclassification Rate

\[
p(\text{mistake}) = p(x \in \mathcal{R}_1, \mathcal{C}_2) + p(x \in \mathcal{R}_2, \mathcal{C}_1)
\]

\[
= \int_{\mathcal{R}_1} p(x, \mathcal{C}_2) \, dx + \int_{\mathcal{R}_2} p(x, \mathcal{C}_1) \, dx.
\]
Minimum Expected Loss

\[ \mathbb{E}[L] = \sum_k \sum_j \int_{\mathcal{R}_j} L_{kj} p(x, C_k) \, dx \]

Regions \( \mathcal{R}_j \) are chosen to minimize

\[ \mathbb{E}[L] = \sum_k L_{kj} p(C_k | x) \]
Generative vs Discriminative

Generative approach:

Model \( p(t, x) = p(x|t)p(t) \)

Use Bayes’ theorem \( p(t|x) = \frac{p(x|t)p(t)}{p(x)} \)

Discriminative approach:

Model \( p(t|x) \) directly
Entropy

\[ H[x] = - \sum_x p(x) \log_2 p(x) \]

Important quantity in
- coding theory
- statistical physics
- machine learning
Entropy

Coding theory: $x$ discrete with 8 possible states; how many bits to transmit the state of $x$?

All states equally likely

$$H[x] = -8 \times \frac{1}{8} \log_2 \frac{1}{8} = 3 \text{ bits.}$$
Entropy

<table>
<thead>
<tr>
<th>$x$</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(x)$</td>
<td>$\frac{1}{2}$</td>
<td>$\frac{1}{4}$</td>
<td>$\frac{1}{8}$</td>
<td>$\frac{1}{16}$</td>
<td>$\frac{1}{64}$</td>
<td>$\frac{1}{64}$</td>
<td>$\frac{1}{64}$</td>
<td>$\frac{1}{64}$</td>
</tr>
<tr>
<td>code</td>
<td>0</td>
<td>10</td>
<td>110</td>
<td>1110</td>
<td>111100</td>
<td>111101</td>
<td>111110</td>
<td>111111</td>
</tr>
</tbody>
</table>

\[
H[x] = -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{4} \log_2 \frac{1}{4} - \frac{1}{8} \log_2 \frac{1}{8} - \frac{1}{16} \log_2 \frac{1}{16} - \frac{4}{64} \log_2 \frac{1}{64} \\
= 2 \text{ bits}
\]

average code length = \[\frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{16} \times 4 + 4 \times \frac{1}{64} \times 6\]

= 2 bits
In how many ways can $N$ identical objects be allocated $M$ bins?

$$W = \frac{N!}{\prod_{i} n_i!}$$

$$H = \frac{1}{N} \ln W \sim - \lim_{N \to \infty} \sum_{i} \left( \frac{n_i}{N} \right) \ln \left( \frac{n_i}{N} \right) = - \sum_{i} p_i \ln p_i$$

Entropy maximized when $\forall i : p_i = \frac{1}{M}$
Entropy

\[ H = 1.77 \]

\[ H = 3.09 \]
The Kullback-Leibler Divergence

\[
\text{KL}(p||q) = - \int p(x) \ln q(x) \, dx - \left( - \int p(x) \ln p(x) \, dx \right) \\
= - \int p(x) \ln \left\{ \frac{q(x)}{p(x)} \right\} \, dx
\]

\[
\text{KL}(p||q) \simeq \frac{1}{N} \sum_{n=1}^{N} \{ - \ln q(x_n|\theta) + \ln p(x_n) \}
\]

\[
\text{KL}(p||q) \geq 0 \quad \text{KL}(p||q) \neq \text{KL}(q||p)
\]
Mutual Information

\[ I[x, y] \equiv KL(p(x, y) \| p(x)p(y)) \]
\[ = - \int \int p(x, y) \ln \left( \frac{p(x)p(y)}{p(x, y)} \right) \, dx \, dy \]