Fair sequential prediction

- We wish to predict (generate) labels \( y_1, \ldots, y_n \) associated with input examples \( x_1, \ldots, x_n \).
- In a fair sequential prediction we predict each label based on the corresponding input and preceding labels and examples:
  
  \[
  \begin{align*}
  y_1 & \text{ is predicted based on } x_1 \text{ alone} \\
  y_2 & \text{ is predicted based on } x_2 \text{ and } D_1 = \{(x_1, y_1)\} \\
  y_3 & \text{ is predicted based on } x_3 \text{ and } D_2 = \{(x_1, y_1), (x_2, y_2)\} \\
  & \text{ …} 
  \end{align*}
  \]

Our fair sequential prediction method defines a valid probability distribution over the training labels given the examples:

\[
P(y_1|x_1)P(y_2|x_2, D_1)P(y_3|x_3, D_2) \cdots P(y_n|x_n, D_{n-1})
\]
**Fair sequential prediction**

- Our fair sequential prediction method defines a valid probability distribution over the training labels given the examples:
  \[ P(y_1|x_1)P(y_2|x_2, D_1)P(y_3|x_3, D_2) \cdots P(y_n|x_n, D_{n-1}) \]
  How well this distribution predicts the training labels depends on the "complexity" of the model \( P(y|x, \theta), \theta \in \Theta \) and how appropriate the prior \( P(\theta) \) is.

- if the model is too flexible: the posterior \( P(\theta|D_{n-1}) \) requires many training examples before it focuses on useful parameter values
- if the model is too simple: the posterior concentrates quickly but the predictions remain poor

**Description length and model selection**

- We need
  \[ -\log_2 \int P(y_1|x_1, \theta) \cdots P(y_n|x_n, \theta) P(\theta) d\theta \]
  bits to communicate labels \( y_1, \ldots, y_n \) given examples \( x_1, \ldots, x_n \) with a model \( P(y|x, \theta), \theta \in \Theta \) and prior \( P(\theta) \).
- Minimum description length (MDL) principle:
  We select the model and prior combination that requires the fewest number of bits (maximizes the Bayesian marginal likelihood)

**Asymptotic approximation**

- For large \( n \) we can use the following asymptotic expansion:
  \[ \approx \text{DL of data} \sum_{i=1}^{n} (-\log_2 P(y_i|x_i, \bar{\theta})) + \frac{d}{2} \log_2(n) \]
  where \( \bar{\theta} \) is the maximum likelihood setting of the parameters and \( d \) is the effective number of parameters in the model.
- The negative of this is also known as the Bayesian information criterion or BIC for short.

**Description length and probabilities**

- If we can predict the training labels with high probability, then we can communicate them effectively (with few bits)
- It takes \(-\log_2 P(y_1, \ldots, y_n)\) bits to communicate \( y_1, \ldots, y_n \) according to distribution \( P \).

Example: suppose each configuration \( (y_1, y_2, y_3) \) is equally likely according to \( P \):

- 1st bit
- 2nd bit
- 3rd bit

We need \(-\log_2 P(y_1, y_2, y_3) = -\log_2(1/8) = 3\) bits to describe each \( y \).
Description length: example
- Example: polynomial logistic regression, $n = 100$

\[
\sum_{i=1}^{n} \left( -\log_2 P(y_i | x_i, \hat{\theta}) \right) + \frac{d}{2} \log_2(n)
\]

Description length: example
- Example: polynomial logistic regression, $n = 100$

<table>
<thead>
<tr>
<th>degree</th>
<th># param</th>
<th>DL(data)</th>
<th>DL(model)</th>
<th>MDL score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>5.6 bits</td>
<td>9.9 bits</td>
<td>15.5 bits</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>2.4 bits</td>
<td>19.9 bits</td>
<td>22.3 bits</td>
</tr>
</tbody>
</table>

Topics
- Sequential prediction and description length
  - minimum description length principle (MDL), asymptotic expansion
- Probability models and structure
  - mixing, mixtures, and the EM-algorithm

What are we missing?
- So far we have solved simple binary classification problems, predicting $y$ given $x$, by estimating
  - discriminant functions (e.g., SVMs and boosting)
  - conditional probabilities (e.g., logistic regression)
- What about problems where
  - we have to predict multiple inter-connected labels for each input example (e.g., a set of topics for a document)
  - we have to switch between classifiers in the course of making predictions (e.g., changes in market conditions)
  - the inputs are incomplete in the sense that some of the components are missing (e.g., patient records)
  - the input examples come in different potentially unobserved types (e.g., mixed populations)

Structure and mixtures
- If we wish to take into account the fact that there are different underlying types of examples, we have to first identify them
- We can hypothesize that
  1. there are $m$ underlying types $y = 1, \ldots, m$
  2. each type $y$ occurs with frequency $P(y)$
  3. examples of type $y$ are governed by distribution $p(x|y)$
- According to this model each observed example $x$ can be assumed to have come from a “mixture distribution”:

\[
p(x) = \sum_{j=1}^{m} P(y = j)p(x|y = j)
\]

- We need to parameterize and estimate such models from samples $x_1, \ldots, x_n$

Mixture densities
- A mixture of Gaussians model

\[
p(x|\theta) = \sum_{i=1}^{m} p_j p(x|\mu_j, \Sigma_j)
\]

where $\theta = \{p_1, \ldots, p_m, \mu_1, \ldots, \mu_m, \Sigma_1, \ldots, \Sigma_m\}$ contains all the parameters of the mixture model. \(\{p_j\}\) are known as mixing proportions or coefficients.
Mixture density estimation
• We get soft labels or posterior probabilities of which Gaussian generated each observation example.

\[ p(x|\theta) = \sum_{j=1,2} p_j \cdot p(x|\mu_j, \Sigma_j) \]  \hspace{1cm} \text{(mixture of Gaussians)}

• Any data point \( x \) could have been generated in two ways; the component responsible for generating \( x \) needs to be inferred.

Mixture density estimation: credit assignment
• Suppose we want to estimate a two component mixture of Gaussians model.

\[ p(x|\theta) = p_1 p(x|\mu_1, \Sigma_1) + p_2 p(x|\mu_2, \Sigma_2) \]

• If each example \( x_i \) in the training set were labeled \( y_i = 1, 2 \) according to which mixture component (1 or 2) had generated it, then the estimation would be easy.

Labeled examples \( \Rightarrow \) no credit assignment problem

Mixture density estimation: credit assignment
• We get soft labels or posterior probabilities of which Gaussian generated each example:

\[ \hat{p}(j|i) \leftarrow P(y_i = j|x_i, \theta) \]

where \( \sum_{j=1,2} \hat{p}(j|i) = 1 \) for all \( i = 1, \ldots, n \).
The EM algorithm: iteration $k$

**E-step:** softly assign examples to mixture components

$$\hat{p}(j|i) \leftarrow P(y_i = j|x_i, \theta^{(k)}), \text{ for all } j = 1, 2 \text{ and } i = 1, \ldots, n$$

**M-step:** estimate new mixture parameters $\theta^{(k+1)}$ based on the soft assignments (can be done separately for the two Gaussians)

$$\hat{\eta}_j \leftarrow \frac{\hat{n}_j}{\hat{n}}, \text{ where } \hat{n}_j = \sum_{i=1}^{n} \hat{p}(j|i)$$

$$\hat{\mu}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \hat{p}(j|i) x_i$$

$$\hat{\Sigma}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \hat{p}(j|i) (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^T$$

The EM-algorithm

- Each iteration of the EM-algorithm *monotonically* increases the (log-)likelihood of the $n$ training examples $x_1, \ldots, x_n$:

$$\log p(\text{data} | \theta^{(k)}) = \sum_{i=1}^{n} \log \left( \hat{p}_1 p(x_i | \mu_1, \Sigma_1) + \hat{p}_2 p(x_i | \mu_2, \Sigma_2) \right)$$

where $\theta^{(k)} = \{p_1, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$ specifies the parameters of the mixture model at the $k^{th}$ iteration.