Racap - Decision Rules

- **Supervised learning**
  - Baysian based - Maximum Posterior Probability (MPP): For a given $x$, if $P(w_1|x) > P(w_2|x)$, then $x$ belongs to class 1, otherwise 2.
    - Parametric Learning
      - Case 1: Minimum Euclidean Distance (Linear Machine), $\Sigma_i = \sigma^2 I$
      - Case 2: Minimum Mahalanobis Distance (Linear Machine), $\Sigma_i = \Sigma$
      - Case 3: Quadratic classifier, $\Sigma_i = \text{arbitrary}$
      - Estimate Gaussian parameters using MLE
    - Nonparametric Learning
      - K-Nearest Neighbor
  - Neural network
    - Perceptron
    - BPNN
  - Kernel-based approaches
    - Support Vector Machine
  - Least-square based

- **Unsupervised learning**
  - Kmeans
  - Winner-takes-all

- **Supporting preprocessing techniques**
  - Normalization
  - Dimensionality Reduction (FLD, PCA)
  - Performance Evaluation (metrics, confusion matrices, ROC, cross validation)
# COSC 522 - Machine Learning (Fall 2020) Syllabus

<table>
<thead>
<tr>
<th>Lecture</th>
<th>Date</th>
<th>Content</th>
<th>Tests</th>
<th>Assignment</th>
<th>Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8/20</td>
<td>Introduction</td>
<td>HW0</td>
<td></td>
<td>8/25</td>
</tr>
<tr>
<td>2</td>
<td>8/25</td>
<td>Bayesian Decision Theory - MPP</td>
<td>HW1</td>
<td></td>
<td>9/1</td>
</tr>
<tr>
<td>3</td>
<td>8/27</td>
<td>Discriminant Function - MD</td>
<td>Proj1 - Supervised Learning</td>
<td></td>
<td>9/10</td>
</tr>
<tr>
<td>4</td>
<td>9/1</td>
<td>Parametric Learning - MLE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>9/3</td>
<td>Non-parametric Learning - kNN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>9/8</td>
<td>Unsupervised Learning - kmeans</td>
<td>HW2</td>
<td></td>
<td>9/15</td>
</tr>
<tr>
<td>7</td>
<td>9/10</td>
<td>Dimensionality Reduction - FLD</td>
<td>Proj2 - Unsupervised Learning and DR</td>
<td></td>
<td>9/24</td>
</tr>
<tr>
<td>8</td>
<td>9/15</td>
<td>Dimensionality Reduction - PCA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9/17</td>
<td>Linear Regression</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9/22</td>
<td>Performance Evaluation</td>
<td>HW3</td>
<td></td>
<td>9/29</td>
</tr>
<tr>
<td>11</td>
<td>9/24</td>
<td>Fusion</td>
<td>Proj3 - Regression</td>
<td></td>
<td>10/8</td>
</tr>
<tr>
<td>12</td>
<td>9/29</td>
<td>Midterm Exam</td>
<td>Final Project - Milestone 1: Forming Team</td>
<td></td>
<td>9/29</td>
</tr>
<tr>
<td>13</td>
<td>10/1</td>
<td>Gradient Descent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>10/6</td>
<td>Neural Network - Perceptron</td>
<td>HW4</td>
<td></td>
<td>10/13</td>
</tr>
<tr>
<td>15</td>
<td>10/8</td>
<td>Neural Network - BPNN</td>
<td>Proj4 - BPNN</td>
<td></td>
<td>10/22</td>
</tr>
<tr>
<td>16</td>
<td>10/13</td>
<td>Neural Network - Practices</td>
<td>Final Project - Milestone 2: Choosing Topic</td>
<td></td>
<td>10/13</td>
</tr>
<tr>
<td>17</td>
<td>10/15</td>
<td>Kernel Methods - SVM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>10/20</td>
<td>Kernel Methods - SVM</td>
<td>HW5</td>
<td></td>
<td>10/27</td>
</tr>
<tr>
<td>19</td>
<td>10/22</td>
<td>Kernel Methods - SVM</td>
<td>Proj5 - SVM &amp; DT</td>
<td></td>
<td>11/5</td>
</tr>
<tr>
<td>20</td>
<td>10/27</td>
<td>Decision Tree</td>
<td>Final Project - Milestone 3: Literature Survey</td>
<td></td>
<td>10/27</td>
</tr>
<tr>
<td>21</td>
<td>10/29</td>
<td>Random Forest</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>11/3</td>
<td></td>
<td>HW6</td>
<td></td>
<td>11/10</td>
</tr>
<tr>
<td>23</td>
<td>11/5</td>
<td>From PCA to t-SNE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>11/10</td>
<td>From Gaussian to Mixture and EM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>11/12</td>
<td>From Supervised/Unsupervised to RL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>11/17</td>
<td>From Classification/Regression to Generation</td>
<td>Final Project - Milestone 4: Prototype</td>
<td></td>
<td>11/17</td>
</tr>
<tr>
<td>27</td>
<td>11/19</td>
<td>From NN to CNN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>11/24</td>
<td></td>
<td></td>
<td>Final Exam</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>12/3</td>
<td>8:00-10:15</td>
<td>Final Presentation Final Project - Report</td>
<td></td>
<td>12/4</td>
</tr>
</tbody>
</table>
References

• Christopher J.C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, 2, 121-167, 1998


Questions

- What does generalization and capacity mean?
- What is VC dimension?
- What is the principled method?
- What is the VC dimension for perceptron?
- What are support vectors?
- What is the cost function for SVM?
- What is the optimization method used?
- How to handle non-separable cases using SVM?
- What is kernel trick?
A bit about Vapnik

- Started SVM study in late 70s
- Fully developed in late 90s
- While at AT&T lab

http://en.wikipedia.org/wiki/Vladimir_Vapnik
Generalization and capacity

• For a given learning task, with a given finite amount of training data, the best generalization performance will be achieved if the right balance is struck between the accuracy attained on that particular training set, and the “capacity” of the machine.

• Capacity – the ability of the machine to learn any training set without error
  – Too much capacity - overfitting
Bounds on the balance

- Under what circumstances, and how quickly, the mean of some empirical quantity converges uniformly, as the number of data point increases, to the true mean.
- True mean error (or actual risk)
  \[ R(\alpha) = \int \frac{1}{2} |y - f(x, \alpha)| p(x, y) dx dy \]
- One of the bounds
  \[ R(\alpha) \leq R_{\text{emp}}(\alpha) + \sqrt{\left( \frac{h(\log(2/l) + 1) - \log(\eta/4)}{l} \right)} \]
  \[ R_{\text{emp}}(\alpha) = \frac{1}{2l} \sum_{i=1}^{l} |y_i - f(x_i, \alpha)| \]

\( f(x, \alpha) \): a machine that defines a set of mappings, \( x \rightarrow f(x, \alpha) \)
\( \alpha \): parameter or model learned
\( h \): VC dimension that measures the capacity. non-negative integer
\( R_{\text{emp}} \): empirical risk
\( \eta \): 1-\( \eta \) is confidence about the loss, \( \eta \) is between [0, 1]
\( l \): number of observations, \( y_i \): label, \{+1, -1\}, \( x_i \) is n-D vector

Principled method: choose a learning machine that minimizes the RHS with a sufficiently small \( \eta \)
\[ R(T_i) \leq R_{\text{emp}}(T_i) + \frac{\ln N - \ln n}{\lambda} \left(1 + \sqrt{1 + \frac{2R_{\text{emp}}(T_i)\lambda}{\ln N - \ln n}}\right) \]

ALL YOUR BAYES ARE BELONG TO US
VC dimension

• For a given set of \( l \) points, there can be \( 2^l \) ways to label them. For each labeling, if a member of the set \( \{f(\alpha)\} \) can be found that correctly classifies them, we say that set of points is shattered by that set of functions.

• VC dimension of that set of functions \( \{f(\alpha)\} \) is defined as the maximum number of training points that can be shattered by \( \{f(\alpha)\} \)

• We should minimize \( h \) in order to minimize the bound
Example \((f(\alpha) \text{ is perceptron})\)

Let's now consider hyperplanes in \(\mathbb{R}^n\). The following theorem will prove useful (the proof is in the Appendix):

**Theorem 1**

Consider some set of \(m\) points in \(\mathbb{R}^n\). Choose any one of the points as origin. Then the \(m\) points can be shattered by oriented hyperplanes if and only if the position vectors of the remaining points are linearly independent.

**Corollary**: The VC dimension of the set of oriented hyperplanes in \(\mathbb{R}^n\) is \(n+1\), since we can always choose \(n+1\) points, and then choose one of the points as origin, such that the position vectors of the remaining \(n\) points are linearly independent, but can never choose \(n+2\) such points (since no \(n+1\) vectors in \(\mathbb{R}^n\) can be linearly independent).

An alternative proof of the corollary can be found in (Anthony and Biggs, 1995), and references therein.

2.3. The VC Dimension and the Number of Parameters

The VC dimension thus gives concreteness to the notion of the capacity of a given set of functions. Intuitively, one might be led to expect that learning machines with many parameters would have high VC dimension, while learning machines with few parameters would have low VC dimension. There is a striking counterexample to this, due to E. Levin and J.S. Denker (Vapnik, 1995): A learning machine with just one parameter, but with infinite VC dimension (a family of classifiers is said to have infinite VC dimension if it can shatter \(l\) points, no matter how large \(l\)).

Define the step function \(\theta(x), x \in \mathbb{R}\):

\[
\begin{align*}
\theta(x) &= 1 \quad \forall \ x > 0; \\
\theta(x) &= -1 \quad \forall \ x \leq 0.
\end{align*}
\]

Consider the one-parameter family of functions, defined by

\[
f(x, \alpha) \equiv \theta(\sin(\alpha x)), \ x, \alpha \in \mathbb{R}.
\]

(4)

You choose some number \(l\), and present me with the task of finding \(l\) points that can be shattered. I choose them to be:

*Figure 1.* Three points in \(\mathbb{R}^2\), shattered by oriented lines.
Questions

• What does generalization and capacity mean?
• What is VC dimension?
• What is the principled method?
• What is the VC dimension for perceptron?
• What are support vectors?
• What is the cost function for SVM?
• What is the optimization method used?
• How to handle non-separable cases using SVM?
• What is kernel trick?
Linear SVM – The separable case

Figure 5. Linear separating hyperplanes for the separable case. The support vectors are circled.

These can be combined into one set of inequalities:

\[ y_i (x_i \cdot w + b) - 1 \geq 0 \quad \forall i \]  

Now consider the points for which the equality in Eq. (10) holds (requiring that there exists such a point is equivalent to choosing a scale for \( w \) and \( b \)). These points lie on the hyperplane \( H_1: x_i \cdot w + b = 1 \) with normal \( w \) and perpendicular distance from the origin \( \frac{|1 - b|}{\|w\|} \). Similarly, the points for which the equality in Eq. (11) holds lie on the hyperplane \( H_2: x_i \cdot w + b = -1 \), with normal again \( w \), and perpendicular distance from the origin \( \frac{|-1 - b|}{\|w\|} \). Hence \( d^+ = d^- = \frac{1}{\|w\|} \) and the margin is simply \( \frac{2}{\|w\|} \).

Note that \( H_1 \) and \( H_2 \) are parallel (they have the same normal) and that no training points fall between them. Thus we can find the pair of hyperplanes which gives the maximum margin by minimizing \( \|w\|^2 \), subject to constraints (12).

Thus we expect the solution for a typical two dimensional case to have the form shown in Figure 5. Those training points for which the equality in Eq. (12) holds (i.e. those which wind up lying on one of the hyperplanes \( H_1, H_2 \)), and whose removal would change the solution found, are called support vectors; they are indicated in Figure 5 by the extra circles.

We will now switch to a Lagrangian formulation of the problem. There are two reasons for doing this. The first is that the constraints (12) will be replaced by constraints on the Lagrange multipliers themselves, which will be much easier to handle. The second is that in this reformulation of the problem, the training data will only appear (in the actual training and test algorithms) in the form of dot products between vectors. This is a crucial property which will allow us to generalize the procedure to the nonlinear case (Section 4).
\[
\begin{align*}
\begin{cases}
    x_i \cdot w + b \geq 1 \quad \text{for} \quad y_i = +1 \\
    x_i \cdot w + b \leq -1 \quad \text{for} \quad y_i = -1
\end{cases}
\end{align*}
\]

Minimizing \( \|w\|^2 \)

s.j. \( y_i (x_i \cdot w + b) - 1 \geq 0 \)

Minimize \( L_p = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{l} \alpha_i y_i (x_i \cdot w + b) + \sum_{i=1}^{l} \alpha_i \)

\[
\frac{\partial L_p}{\partial w} = 0 \Rightarrow w = \sum_i \alpha_i y_i x_i, \quad \frac{\partial L_p}{\partial b} = 0 \Rightarrow \sum_i \alpha_i y_i = 0
\]

Maximize \( L_D = -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j + \sum \alpha_i \)
Questions

• What does generalization and capacity mean?
• What is VC dimension?
• What is the principled method?
• What is the VC dimension for perceptron?
• What are support vectors?
• What is the cost function for SVM?
• What is the optimization method used?
• How to handle non-separable cases using SVM?
• What is kernel trick?
Non-separable cases

- SVM with soft margin
- Kernel trick
Non-separable case – Soft margin

\[
\begin{align*}
\text{Minimizing } & \|w\|^2 \\
\text{s.j. } & y_i(x_i \cdot w + b) - 1 + \xi_i \geq 0
\end{align*}
\]

Minimize \( L_p = \frac{1}{2} \|w\|^2 - C \left( \sum_i \xi_i \right)^k \)

Maximize \( L_D = -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j + \sum_i \alpha_i \)

s.j. \( 0 \leq \alpha_i \leq C, \sum_i \alpha_i y_i = 0 \)
Non-separable cases – The kernel trick

- If there were a “kernel function”, K, s.t.

\[ K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j) = e^{-\frac{\|x_i-x_j\|^2}{2\sigma^2}} \]

Gaussian Radial Basis Function (RBF)
Comparison - XOR
Limitation

- Need to choose parameters
A toy example