95-865: Support Vector Machines, Decision Trees and Forests

George Chen
Support Vector Machines
What should the label of this new point be?
Decision boundary
Which decision boundary is best?
Which decision boundary is best?
SVM solution: maximize “margin” between red and blue points
(make decision boundary line thicker until it hits a data point—this thickness is the size of the margin)
Which decision boundary is best?

SVM solution: maximize “margin” between red and blue points

(make decision boundary line thicker until it hits a data point—this thickness is the size of the margin)

The points that the margin hits are called support vectors
The points that the margin hits are called support vectors.

The decision boundary that the SVM outputs only depends on the support vectors.
What if the points cannot actually be separated by a line?

Hyperparameter $C$ is a penalty for a point being on the wrong side of the decision boundary.
What if the points cannot actually be separated by a line?

Penalty incurred for highlighted blue point: $C \times$ length of purple line

Hyperparameter $C$ is a penalty for a point being on the wrong side of the decision boundary

Larger $C \rightarrow$ work harder to fit all points
C-Support Vector Classification

- Basic version measures distance using Euclidean distance
  - Turns out to correspond to measuring similarity between two points by taking their dot product

- Can instead use a different similarity function ("kernel" function) instead (popular choice: Gaussian kernel, also called "radial basis function" kernel)
C-Support Vector Classification

Demo
Decision Trees
Example Made-Up Data

<table>
<thead>
<tr>
<th>Age (years)</th>
<th>Weight (lb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>30</td>
</tr>
<tr>
<td>50</td>
<td>20</td>
</tr>
</tbody>
</table>

Red: diabetic
Blue: not diabetic
Example Decision Tree

- **Age > 40?**
  - **no**
    - **Age > 30?**
      - **no**
        - **not diabetic**
      - **yes**
        - **diabetic**
  - **yes**
    - **Weight > 200?**
      - **yes**
        - **diabetic**
      - **no**
        - **not diabetic**
Learning a Decision Tree

- Many ways: general approach actually looks a lot like divisive clustering *but accounts for label information*

- I’ll show one way (that nobody actually uses in practice) but it’s easy to explain
Learning a Decision Tree

1. Pick a random feature (either age or weight)

2. Find threshold for which red and blue are as “separate as possible” (on one side, mostly red; on other side, mostly blue)
Learning a Decision Tree

Within each side, recurse until a termination criterion is reached!

Example termination criteria: \( \geq 90\% \) points within region has same label, number of points within region is <5

Note: label within each region is majority vote

Red: diabetic
Blue: not diabetic
For a new person with feature vector (age, weight), easy to predict!
Decision Forest for Classification

- Typically, a decision tree is learned with randomness (e.g., we randomly chose which feature to threshold)
  ➔ by re-running the same learning procedure, we can get different decision trees that make different predictions!
- For a more stable prediction, use many decision trees

Learn each tree separately using same training data

New test data point

Tree 1 ➔ diabetic

Tree 2 ➔ not diabetic ➔ diabetic

Tree 3 ➔ diabetic

Tree $T$ ➔ diabetic

Final prediction: majority vote of the different trees’ predictions
Randomly sample \( n \) points

New test data point

\( n \) training data points

Randomizing training data this way is called **bagging** (bootstrap aggregating)

Tree 1 → diabetic

Tree 2 → not diabetic

Tree 3 → diabetic

\( \ldots \) → diabetic

\( T \) → diabetic

**Question:** What happens if all the trees are the same?

*Adding randomness can make trees more different!*

- **Random Forest:** in addition to randomly choosing features to threshold, also randomize training data used for each tree

- **Extremely randomized trees:** further randomize thresholds rather than trying to pick clever thresholds