K Nearest Neighbor
Wrap Up
K- Means Clustering

Slides adapted from Prof. Carpuvat
**K Nearest Neighbor Classification**

Training Data

K: number of neighbors that classification is based on

Test instance with unknown class in \{−1; +1 \}

---

**Algorithm 3 KNN-Predict(D, K, \(\hat{x}\))**

1. \(S \leftarrow [\ ]\)
2. for \(n = 1\) to \(N\) do
3. \(S \leftarrow S \oplus \langle d(x_n, \hat{x}), n \rangle\) // store distance to training example \(n\)
4. end for
5. \(S \leftarrow \text{sort}(S)\) // put lowest-distance objects first
6. \(\hat{y} \leftarrow 0\)
7. for \(k = 1\) to \(K\) do
8. \(\langle \text{dist}, n \rangle \leftarrow S_k\) // \(n\) this is the \(k\)th closest data point
9. \(\hat{y} \leftarrow \hat{y} + y_n\) // vote according to the label for the \(n\)th training point
10. end for
11. return \(\text{sign}(\hat{y})\) // return +1 if \(\hat{y} > 0\) and −1 if \(\hat{y} < 0\)
Components of a k-NN Classifier

• Distance metric
  – How do we measure distance between instances?
  – Determines the layout of the example space

• The k hyperparameter
  – How large a neighborhood should we consider?
  – Determines the complexity of the hypothesis space
$K=1$ and Voronoi Diagrams

- Imagine we are given a bunch of training examples
- Find regions in the feature space which are closest to every training example
- Algorithm – if our test point is in the region corresponding to a given input point – return its label
Decision Boundaries for 1-NN
Decision Boundaries change with the distance function.
Decision Boundaries change with $K$
The k hyperparameter

• Tunes the complexity of the hypothesis space
  – If k = 1, every training example has its own neighborhood
  – If k = N, the entire feature space is one neighborhood!

• Higher k yields smoother decision boundaries

• How would you set k in practice?
What is the inductive bias of \( k \)-NN?

- Nearby instances should have the same label
- All features are equally important
- Complexity is tuned by the \( k \) parameter
Variations on k-NN: Weighted voting

• Weighted voting
  – Default: all neighbors have equal weight
  – Extension: weight neighbors by (inverse) distance
Variations on k-NN: Epsilon Ball Nearest Neighbors

• Same general principle as K-NN, but change the method for selecting which training examples vote

• Instead of using K nearest neighbors, use all examples $x$ such that

$$\text{distance}(\hat{x}, x) \leq \epsilon$$
Exercise: How would you modify KNN-Predict to perform Epsilon Ball NN?

**Algorithm 3 KNN-Predict(D, K, \( \hat{x} \))**

1. \( S \leftarrow [ ] \)
2. **for** \( n = 1 \) to \( N \) **do**
3. \( S \leftarrow S \oplus \langle d(x_n, \hat{x}), n \rangle \) \hspace{1cm} // store distance to training example \( n \)
4. **end for**
5. \( S \leftarrow \textbf{sort}(S) \) \hspace{1cm} // put lowest-distance objects first
6. \( \hat{y} \leftarrow 0 \)
7. **for** \( k = 1 \) to \( K \) **do**
8. \( \langle \text{dist}, n \rangle \leftarrow S_k \) \hspace{1cm} // \( n \) this is the \( k \)th closest data point
9. \( \hat{y} \leftarrow \hat{y} + y_n \) \hspace{1cm} // vote according to the label for the \( n \)th training point
10. **end for**
11. **return** \( \text{sign}(\hat{y}) \) \hspace{1cm} // return +1 if \( \hat{y} > 0 \) and \(-1\) if \( \hat{y} < 0 \)
Exercise: When are DT vs kNN appropriate?

<table>
<thead>
<tr>
<th>Properties of classification problem</th>
<th>Can Decision Trees handle them?</th>
<th>Can K-NN handle them?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary features</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Numeric features</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Categorical features</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robust to noisy training examples</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast classification is crucial</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Many irrelevant features</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Relevant features have very different scale</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Exercise: When are DT vs kNN appropriate?

<table>
<thead>
<tr>
<th>Properties of classification problem</th>
<th>Can Decision Trees handle them?</th>
<th>Can K-NN handle them?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary features</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Numeric features</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Categorical features</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Robust to noisy training examples</td>
<td>no (for default algorithm)</td>
<td>yes (when ( k &gt; 1 ))</td>
</tr>
<tr>
<td>Fast classification is crucial</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Many irrelevant features</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Relevant features have very different scale</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>
Recap

• Nearest Neighbors (NN) algorithms for classification
  – K-NN, Epsilon ball NN
  – Take a geometric view of learning

• Fundamental Machine Learning Concepts
  – Decision boundary
    • Visualizes predictions over entire feature space
    • Characterizes complexity of learned model
    • Indicates overfitting/underfitting
K-Means
an example of
unsupervised learning
When applying a learning algorithm, some things are properties of the problem you are trying to solve, and some things are up to you to choose as the ML programmer. Which of the following are properties of the problem?

- The data generating distribution
- The train/dev/test split
- The learning model
- The loss function
Today’s Topics

• A new algorithm
  – K-Means Clustering

• Fundamental Machine Learning Concepts
  – Unsupervised vs. supervised learning
  – Decision boundary
Clustering

• Goal: automatically partition examples into groups of similar examples

• Why? It is useful for
  – Automatically organizing data
  – Understanding hidden structure in data
  – Preprocessing for further analysis
What can we cluster in practice?

- news articles or web pages by topic
- protein sequences by function, or genes according to expression profile
- users of social networks by interest
- customers according to purchase history
- galaxies or nearby stars
- ...
Clustering

• Input
  – a set $S$ of $n$ points in feature space
  – a distance measure specifying distance $d(x_i,x_j)$ between pairs $(x_i,x_j)$

• Output
  – A partition $\{S_1,S_2, \ldots, S_k\}$ of $S$
Supervised Machine Learning as Function Approximation

Problem setting
- Set of possible instances $X$
- Unknown target function $f : X \rightarrow Y$
- Set of function hypotheses $H = \{h | h : X \rightarrow Y\}$

Input
- Training examples $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$ of unknown target function $f$

Output
- Hypothesis $h \in H$ that best approximates target function $f$
Supervised vs. unsupervised learning

- Clustering is an example of unsupervised learning
- We are not given examples of classes $y$
- Instead we have to discover classes in data
2 datasets with very different underlying structure!
The K-Means Algorithm

Algorithm 4 $K$-Means$(D, K)$

1: for $k = 1$ to $K$ do
2: \hspace{1em} $\mu_k \leftarrow$ some random location \hspace{1em} // randomly initialize mean for $k$th cluster
3: end for

4: repeat
5: \hspace{1em} for $n = 1$ to $N$ do
6: \hspace{2em} $z_n \leftarrow \text{argmin}_k \| \mu_k - x_n \| $ \hspace{1em} // assign example $n$ to closest center
7: \hspace{1em} end for
8: \hspace{1em} for $k = 1$ to $K$ do
9: \hspace{2em} $X_k \leftarrow \{ x_n : z_n = k \} $ \hspace{1em} // points assigned to cluster $k$
10: \hspace{2em} $\mu_k \leftarrow \text{MEAN}(X_k) $ \hspace{1em} // re-estimate mean of cluster $k$
11: \hspace{1em} end for
12: until $\mu$s stop changing
13: return $z$ \hspace{1em} // return cluster assignments
Example: using K-Means to discover 2 clusters in data
Example: using K-Means to discover 2 clusters in data
K-Means properties

- Time complexity: $O(KNL)$ where
  - $K$ is the number of clusters
  - $N$ is number of examples
  - $L$ is the number of iterations

- $K$ is a hyperparameter
  - Needs to be set in advance (or learned on dev set)

- Different initializations yield different results!
  - Doesn’t necessarily converge to best partition

- “Global” view of data: revisits all examples at every iteration
Impact of initialization
Impact of initialization