Recall: $k$NN

- Classify a point as the most common label among the labels of the $k$ nearest training points.
- If we have a binary classification problem and $k$ is odd:
  \[
g(\tilde{x}) = \text{sign} \left( \sum_{i=1}^{k} y_{[i]}(\tilde{x}) \right)
\]
- $k$ controls the complexity of the hypothesis set $\Rightarrow k$ affects how well the learned hypothesis will generalize.
  - $k = 3$
  - $k = \lceil \sqrt{n} \rceil$
  - Cross-validation
**$k$NN Pros and Cons**

**Pros:**
- Intuitive / explainable
- No training / retraining
- Self-regularizes
- Provably near-optimal in terms of $E_{out}$

**Cons:**
- Computationally expensive
  - Always needs to store all data: $O(nD)$
  - Computing $g(\tilde{x})$ requires computing $d(\tilde{x}, \tilde{x}') \forall \tilde{x}' \in \mathcal{D}$ and finding the $k$ closest points: $O(nD + n \log(k))$
- Suffers from the “curse of dimensionality”
$k$NN

- $k$NN only considers some points and weights them equally
- What if we considered all points but weighted them unequally?
- Intuition: all points are useful but some points are more useful than others!
- Bonus: no need to choose $k$
• Suppose we have a binary classification problem
• Recall for $k$NN (when $k$ is odd):

\[
g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{k} y[i](\vec{x}) \right)
\]

• For RBF:

\[
g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{n} \phi(\vec{x}, \vec{x_i}) y_i \right)
\]
• Suppose we have a binary classification problem
• Recall for $k$NN (when $k$ is odd):

$$g(\tilde{x}) = \text{sign} \left( \sum_{i=1}^{k} y[i](\tilde{x}) \right)$$

• For RBF:

$$g(\tilde{x}) = \text{sign} \left( \sum_{i=1}^{n} \left( \frac{\phi(\tilde{x}, \tilde{x}_i)}{\sum_{i=1}^{n} \phi(\tilde{x}, \tilde{x}_i)} y_i \right) \right)$$
• Suppose we have a binary classification problem
• Recall for $k$NN (when $k$ is odd):

$$g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{k} y[i](\vec{x}) \right)$$

• For RBF:

$$g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{n} \left( \frac{\phi(\|\vec{x} - \vec{x}_i\|)}{\sum_{i=1}^{n} \phi(\|\vec{x} - \vec{x}_i\|)} y_i \right) \right)$$
Suppose we have a binary classification problem

Recall for $k$NN (when $k$ is odd):

$$g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{k} y[i] (\vec{x}) \right)$$

For RBF:

$$g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{n} \left( \frac{\phi \left( \frac{||\vec{x} - \vec{x}_i||}{r} \right)}{\sum_{i=1}^{n} \phi \left( \frac{||\vec{x} - \vec{x}_i||}{r} \right)} \right) y_i \right)$$
Setting $r$

- If $r$ is really small, then $\frac{||\hat{x} - \overline{x}_i||}{r}$ will always be large (unless $||\hat{x} - \overline{x}_i||$ is small)

- If $\frac{||\hat{x} - \overline{x}_i||}{r}$ is large, then $\phi \left( \frac{||\hat{x} - \overline{x}_i||}{r} \right)$ will always be small (unless $||\hat{x} - \overline{x}_i||$ is small)

- The smaller $r$ is, the closer $\hat{x}$ has to be to $\overline{x}_i$ in order for $\phi \left( \frac{||\hat{x} - \overline{x}_i||}{r} \right)$ to be non-zero

- As $r \to 0$, the RBF hypothesis approaches the nearest-neighbor hypothesis

- Set $r$ using cross-validation
Choosing $\phi$

- Intuitively, we want points close to $\tilde{x}$ to have large weights and points far from $\tilde{x}$ to have small weights
  - $\phi(z)$ is maximized when $z = 0$
  - $\phi(z) \to 0$ as $z \to \infty$

- Most common choice is the Gaussian kernel:

$$\phi(z) = e^{-\frac{z^2}{2}}$$
• Suppose we have a binary classification problem
• Recall for $k$NN (when $k$ is odd):

$$g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{k} y_{[i]}(\vec{x}) \right)$$

• For RBF:

$$g(\vec{x}) = \text{sign} \left( \sum_{i=1}^{n} \left( \frac{e^{-\frac{||\vec{x} - \vec{x}_i||^2}{2r^2}}}{\sum_{i=1}^{n} e^{-\frac{||\vec{x} - \vec{x}_i||^2}{2r^2}}} \right) y_i \right)$$
Recall:
Perceptron

• A linear model for classification:

\[ h(\hat{x}) = \text{sign} \left( \sum_{i=0}^{D} w_i x_i \right) = \text{sign}(\hat{w}^T \hat{x}) \]

• Perceptron Learning Algorithm (PLA) finds a linear separator in finite time, if the data is linearly separable
  • Input: \( D = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)\} \)
  • Initialize \( \vec{w} \) to all zeros or (small) random numbers
  • While \( \exists \) a misclassified training point
    • Pick a misclassified training point, \((\hat{x}, y)\)
    • Update \( \vec{w} : \vec{w} = \vec{w} + y\hat{x} \)
Linearly Separable Data

![Linearly separable data diagram](image-url)
Linearly Separable Data
Linearly Separable Data
Which linear separator is best?
Maximal Margin Linear Separators

- The margin of a separating hyperplane is the distance between the hyperplane and the nearest training point.

- Questions:
  - How can we efficiently find a maximal-margin linear separator?
  - Why are linear separators with larger margins better?
  - What can we do if the data is not linearly separable?
For linear models, hypotheses are $D$-dimensional hyperplanes defined by a weight vector, $[w_0, \vec{w}]$

$$[w_0, \vec{w}]^T [1, \vec{x}] = 0 \rightarrow \vec{w}^T \vec{x} + w_0 = 0$$

Problem: there are infinitely many weight vectors that describe the same hyperplane

- $x_1 + 2x_2 + 2 = 0$ is the same line as $2x_1 + 4x_2 + 4 = 0$, which is the same line as $1000000x_1 + 2000000x_2 + 2000000 = 0$

Solution: normalize weight vectors
Given a dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ where $y = \{-1, +1\}$, $[w_0, \mathbf{w}]$ is a linear separator if
\[ y_i (\mathbf{w}^T \mathbf{x}_i + w_0) > 0 \quad \forall \ (\mathbf{x}_i, y_i) \in \mathcal{D} \]

$h(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 = 0$ is a separating hyperplane if
\[ \min_{(\mathbf{x}_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) = 1 \]

If $[w_0, \mathbf{w}]$ is a linear separator, then $h(\mathbf{x}) = \frac{\mathbf{w}^T \mathbf{x}}{\rho} + \frac{w_0}{\rho} = 0$ is a separating hyperplane where
\[ \rho = \min_{(\mathbf{x}_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) \]
Computing the Margin

• Claim: the vector \( \vec{w} \) is orthogonal to the hyperplane
  \[ h(\vec{x}) = \vec{w}^T \vec{x} + w_0 = 0 \]

• Proof:
  • A vector is orthogonal to a hyperplane if it is orthogonal to every vector in that hyperplane
  • Vectors \( \vec{a} \) and \( \vec{b} \) are orthogonal if \( \vec{a}^T \vec{b} = 0 \)
  • Let \( \vec{x}' \) and \( \vec{x}'' \) be two arbitrary points on \( h(\vec{x}) = 0 \)
    • \( \vec{x}' - \vec{x}'' \) is a vector on \( h(\vec{x}) = 0 \)
    • \( \vec{w}^T \vec{x}' + w_0 = 0 \rightarrow \vec{w}^T \vec{x}' = -w_0 \)
    • \( \vec{w}^T (\vec{x}' - \vec{x}'') = \vec{w}^T \vec{x}' - \vec{w}^T \vec{x}'' = -w_0 + w_0 = 0 \)
Computing the Margin

- Claim: the vector $\vec{w}$ is orthogonal to the hyperplane

$$h(\vec{x}) = \vec{w}^T \vec{x} + w_0 = 0$$
Computing the Margin

- Let $\hat{x}'$ be an arbitrary point on the hyperplane $h(\hat{x}) = \overrightarrow{w}^T \hat{x} + w_0 = 0$ and let $\hat{x}''$ be an arbitrary point.
- The distance between $\hat{x}''$ and $h(\hat{x}) = 0$ is equal to the magnitude of the projection of $\hat{x}'' - \hat{x}'$ onto $\frac{\overrightarrow{w}}{||\overrightarrow{w}||}$, the unit vector orthogonal to $h(\hat{x}) = 0$. 
Let $\hat{x}'$ be an arbitrary point on the hyperplane $h(\hat{x}) = \vec{w}^T \hat{x} + w_0 = 0$ and let $\hat{x}''$ be an arbitrary point.

The distance between $\hat{x}''$ and $h(\hat{x}) = 0$ is equal to the magnitude of the projection of $\hat{x}'' - \hat{x}'$ onto $\frac{\vec{w}}{||\vec{w}||}$, the unit vector orthogonal to $h(\hat{x}) = 0$.

$$d(\hat{x}'', h) = \left| \frac{\vec{w}^T (\hat{x}'' - \hat{x}')}{||\vec{w}||} \right| = \frac{|\vec{w}^T \hat{x}'' - \vec{w}^T \hat{x}'|}{||\vec{w}||} = \frac{|\vec{w}^T \hat{x}'' + w_0|}{||\vec{w}||}$$
The margin of a separating hyperplane is the distance between the hyperplane and the nearest training point:

\[
\min_{(x_i, y_i) \in \mathcal{D}} d(x_i, h) = \min_{(x_i, y_i) \in \mathcal{D}} \frac{|\mathbf{w}^T x_i + w_0|}{\|\mathbf{w}\|}
\]

\[
= \frac{1}{\|\mathbf{w}\|} \min_{(x_i, y_i) \in \mathcal{D}} (\mathbf{w}^T x_i + w_0)
\]

\[
= \frac{1}{\|\mathbf{w}\|} \min_{(x_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T x_i + w_0)
\]

\[
= \frac{1}{\|\mathbf{w}\|} = \frac{1}{\sqrt{\mathbf{w}^T \mathbf{w}}}
\]
Maximizing the Margin

\[
\begin{align*}
\text{maximize} & \quad \frac{1}{\sqrt{\mathbf{w}^T \mathbf{w}}} \\
\text{subject to} & \quad \min_{(\mathbf{x}_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) = 1 \\
\downarrow & \\
\text{maximize} & \quad \frac{1}{\mathbf{w}^T \mathbf{w}} \\
\text{subject to} & \quad \min_{(\mathbf{x}_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) = 1 \\
\downarrow & \\
\text{minimize} & \quad \mathbf{w}^T \mathbf{w} \\
\text{subject to} & \quad \min_{(\mathbf{x}_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) = 1 \\
\downarrow & \\
\text{minimize} & \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} \\
\text{subject to} & \quad \min_{(\mathbf{x}_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) = 1
\end{align*}
\]
minimize $\frac{1}{2} \mathbf{w}^T \mathbf{w}$

subject to $\min_{(x_i, y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) = 1$

⇓

minimize $\frac{1}{2} \mathbf{w}^T \mathbf{w}$

subject to $y_i (\mathbf{w}^T \mathbf{x}_i + w_0) \geq 1 \ \forall \ (\mathbf{x}_i, y_i) \in \mathcal{D}$

• If $[w_0^*, \mathbf{w}^*]$ is the optimal solution, then $\exists$ at least one training point $(\mathbf{x}_i, y_i) \in \mathcal{D}$ s.t. $y_i (\mathbf{w}^* T \mathbf{x}_i + w_0^*) = 1$

• All points $(\mathbf{x}_i, y_i) \in \mathcal{D}$ where $y_i (\mathbf{w}^* T \mathbf{x}_i + w_0^*) = 1$ are known as support vectors
Maximizing the Margin

Converting the non-linear constraint (involving the min function) to $n$ linear constraints allows the optimization problem to be solved (approximately) using quadratic programming (QP) in $O(D^3)$ time.

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} \\
\text{subject to} & \quad \min_{(x_i,y_i) \in \mathcal{D}} y_i (\mathbf{w}^T \mathbf{x}_i + w_0) = 1 \\
\end{align*}
\]

\[\downarrow\]

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} \\
\text{subject to} & \quad y_i (\mathbf{w}^T \mathbf{x}_i + w_0) \geq 1 \forall (\mathbf{x}_i, y_i) \in \mathcal{D} \\
\end{align*}
\]
Consider three points in a **bounded** input space. 

- $\mathcal{H}$ = linear separators can shatter any such three points.
Why Maximal Margins?

- Consider three points in a **bounded** input space.
- $\mathcal{H}_\rho = \text{linear separators with minimum margin } \rho$ cannot always shatter all sets of three points.
VC-dimension of $\mathcal{H}_\rho$

- If the input space, $\mathcal{X}$, is a $D$-dimensional sphere of radius $R$, then:

$$d_{VC}(\mathcal{H}_\rho) \leq \min\left(D, \left\lceil \frac{R^2}{\rho^2} \right\rceil \right) + 1$$
minimize $E_{in}(\vec{w}) = \frac{1}{n} \sum_{i=1}^{n} (\vec{w}^T x_i - y_i)^2$

subject to $\sum_{j=0}^{d} w_j^2 \leq C$

<table>
<thead>
<tr>
<th>SVM</th>
<th>Regularization</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimize $\frac{1}{2} \vec{w}^T \vec{w}$</td>
<td>$E_{in}(\vec{w})$</td>
</tr>
<tr>
<td>subject to $E_{in}(\vec{w}) = 0$</td>
<td>$\vec{w}^T \vec{w} \leq C$</td>
</tr>
</tbody>
</table>