1 Introduction

*Machine learning* uses data examples to predict a label or value for a **example**.

**Supervised vs. Unsupervised Learning**

- In *supervised learning*, the dataset is a collection of labeled examples \( \{(x_i, y_i)\}_{i=1}^N \), where \( x_i = [x_1^{(i)}, \ldots, x_D^{(i)}] \) is a \( D \)-dimensional *feature vector*.

  e.g. Here are data with \( N = \_\_\_\_1 \) and \( D = \_\_\_\_2 \) from three kids of \( \{(x_i = [\text{height, weight}], y_i = \text{age})\} \):

<table>
<thead>
<tr>
<th></th>
<th>height</th>
<th>weight</th>
<th>age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>44</td>
<td>70</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>45</td>
<td>75</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>38</td>
<td>40</td>
<td>4</td>
</tr>
</tbody>
</table>

We create a model to map new examples to suitable labels, e.g.:

- A *support vector machine* (SVM) is a **classifier** that uses a line to separate points in a plane into two groups (or it separates \( D \)-dimensional points with a \((D-1)\)-dimensional hyperplane). e.g.\(^2\)

- *Linear regression* predicts a **label** given an unlabeled example as \( y \leftarrow f_{w,b}(x) = wx + b \) for scalar \( y \), vector \( x \), and parameter vector \( w \). e.g.

---

\(^1\)These notes are based on Andriy Burkov’s “The Hundred-Page Machine Learning Book” [http://themlbook.com](http://themlbook.com).

Logistic regression models a probability in $[0, 1]$. e.g.

A decision tree is a directed acyclic graph that we use like a classification tool to make a decision. At each node, if the value of some feature $j$ is less than a threshold, the left branch is followed; otherwise the right branch is followed. e.g.

from https://scikit-learn.org/stable/_images/sphx_glr_plot_logistic_001.png
from https://scikit-learn.org/stable/_images/iris.svg
- **k-nearest neighbors** (k-NN) classification assigns a new x the ____________ label among its __ nearest neighbors. k-NN regression assigns x the ____________ value among its k nearest neighbors. e.g.

- In **unsupervised learning**, the dataset is a collection of ____________ examples \( \{x_i\}_{i=1}^N \) and we infer a function on x to solve a problem or find hidden structure in \( \{x_i\} \). e.g.:

  - **Density estimation** models the probability density function of the (__________) distribution from which data were drawn. e.g.

```
```
- **Clustering** maps each unlabeled example \( x \) to a ___________. e.g.\(^6\)

- **Dimensionality reduction** maps \( x \) into a vector with ____________ to remove ____________ features, reduce ____________ data (since we can only see up to 3D), and facilitate simple interpretable models.

- **Outlier detection** quantifies how far \( x \) is from ____________ examples. e.g.\(^7\)

\(^6\)from [https://scikit-learn.org/stable/_images/sphx_glr_plot_dbscan_001.png](https://scikit-learn.org/stable/_images/sphx_glr_plot_dbscan_001.png) and [https://scikit-learn.org/stable/_images/sphx_glr_plot_digits_classification_001.png](https://scikit-learn.org/stable/_images/sphx_glr_plot_digits_classification_001.png) and [https://scikit-learn.org/stable/_images/sphx_glr_plot_kmeans_digits_001.png](https://scikit-learn.org/stable/_images/sphx_glr_plot_kmeans_digits_001.png)

\(^7\)from [https://scikit-learn.org/stable/_images/sphx_glr_plot_oneclass_001.png](https://scikit-learn.org/stable/_images/sphx_glr_plot_oneclass_001.png)
Support Vector Machine (SVM): The Linear Model

- A hyperplane in a $D$-dimensional space is a $(D - 1)$-dimensional space. e.g. A hyperplane is a __________ in 1D, a __________ in 2D, and a __________ in 3D.

- SVM using a linear model finds a hyperplane decision boundary specified by $wx + b = 0$ that separates label +1 examples from label $-1$ examples. (Note: $wx = w^{(1)}x^{(1)} + \ldots + w^{(D)}x^{(D)}$.)

- Training learns optimal values $w^*$ and $b^*$.

- The SVM labels a new $x$ with $y = f(x) = \text{sign}(w^*x + b^*) \in \{-1, 1\}$.

- In the easiest hard margin SVM case where the two labeled subsets are linearly separable, training consists of minimizing Euclidean norm $||w|| = \sqrt{\sum_{i=1}^{D} (w^{(i)})^2}$ subject to constraints

$$\begin{cases}
wx_i + b \geq 1 & \text{if } y_i = +1 \\
wx_i + b \leq -1 & \text{if } y_i = -1,
\end{cases}$$

or equivalently subject to $y_i(wx_i + b) \geq 1$, for $i = 1, \ldots, N$.

(We omit the details of this constrained optimization problem.)

- The parallel hyperplanes $wx + b = 1$ and $wx + b = -1$ have normal vector __________. Let $x_1$ be any point in the first hyperplane. The normal line through $x_1$ is $x_1 + wt$ for $t \in \mathbb{R}$. It intersects the second hyperplane when

$$w(x_1 + wt) + b = -1 \implies t = \frac{-(wx_1 + b) - 1}{ww} = -2 \frac{2w}{||w||^2}$$

The intersection point is $x_2 = x_1 + w \left( \frac{-2}{||w||^2} \right) = x_1 - \frac{2w}{||w||^2}$

The distance from $x_1$ to $x_2$ is $\left| x_1 - \left( x_1 - \frac{2w}{||w||^2} \right) \right| = \frac{2}{||w||}.$

- $||w||$ is in the denominator of the distance, so minimizing $||w||$ __________ the margin between +1 and $-1$ support vectors.

- A sample on either of the constraint/margin boundaries is called a __________ vector.

Coming in §3:

- An SVM can have a hyperparameter (parameter controlling learning; not trained) to penalize misclassification of outliers (positives on the negative side of the boundary or negatives on the positive side).

- An SVM can include a kernel that allows a __________ decision boundary.

---

8Burkov uses $wx - b = 0$. I use $wx + b = 0$ to match scikit-learn.

9We return to SVMs in §3 to address some harder cases.
Python

- `from sklearn import svm` loads the `svm` module
- `clf = svm.SVC(kernel='linear', C=1)` gives a SVM support vector classification model. (A large C, like C=1000, gives $\approx$ the hard-margin version above; we will explore C more in §3.)
- `clf.fit(X, y)` fits the model to $X_{N \times D}$ and $y_{N \times 1}$
- `clf.coef_` gives $w^*$ and `clf.intercept_` gives $b^*$
- `clf.predict(X)` does classification on examples in $X$
- `clf.score(X, y)` gives the average accuracy on $X$ with respect to $y$

To learn more:


\[^{10}\text{In the code, } X \text{ is 2D while } y \text{ is 1D.}\]