Logistic regression
1. Linear classifiers
Last two lectures, we studied linear regression; the output/label space $\mathcal{Y}$ was $\mathbb{R}$. 
Today, the goal is a linear **linear classifier**; the output/label space $\mathcal{Y}$ is discrete.
For now, let's consider binary classification: \( \mathcal{Y} = \{-1, +1\} \).

A linear predictor \( w \in \mathbb{R}^d \) classifies according to \( \text{sign}(w^T x) \in \{-1, +1\} \).

Given \( ((x_i, y_i))_{i=1}^n \), a predictor \( w \in \mathbb{R}^d \), we want \( \text{sign}(w^T x_i) \) and \( y_i \) to agree.
A hyperplane in $\mathbb{R}^d$ is a linear subspace of dimension $d-1$.

- A $\mathbb{R}^2$-hyperplane is a line.
- A $\mathbb{R}^3$-hyperplane is a plane.
- As a linear subspace, a hyperplane always contains the origin.

A hyperplane $H$ can be specified by a (non-zero) normal vector $\mathbf{w} \in \mathbb{R}^d$.

The hyperplane with normal vector $\mathbf{w}$ is the set of points orthogonal to $\mathbf{w}$:

$$H = \left\{ \mathbf{x} \in \mathbb{R}^d : \mathbf{x}^T \mathbf{w} = 0 \right\}.$$

Given $\mathbf{w}$ and its corresponding $H$: $H$ splits the sets labeled positive $\{\mathbf{x} : \mathbf{w}^T \mathbf{x} > 0\}$ and those labeled negative $\{\mathbf{x} : \mathbf{w}^T \mathbf{w} < 0\}$.
Classification with a hyperplane

Projection of \( x \) onto \( \text{span}\{w\} \) (a line) has coordinate

\[
\|x\|^2 \cdot \cos(\theta)
\]

where

\[
\cos(\theta) = \frac{x^T w}{\|w\|^2 \|x\|^2}
\]

(Distance to hyperplane is \( \|x\|^2 \cdot |\cos(\theta)| \).)

Decision boundary is hyperplane (oriented by \( w \)):

\[
x^T w > 0 \iff \|x\|^2 \cdot \cos(\theta) > 0 \iff x \text{ on same side of } H \text{ as } w
\]

What should we do if we want hyperplane decision boundary that doesn’t (necessarily) go through origin?
Projection of $x$ onto $\text{span}\{w\}$ (a line) has coordinate

$$\|x\|_2 \cdot \cos(\theta)$$

where

$$\cos(\theta) = \frac{x^\top w}{\|w\|_2 \|x\|_2}.$$

(Distance to hyperplane is $\|x\|_2 \cdot |\cos(\theta)|$.)
Classification with a hyperplane

Projection of \( \mathbf{x} \) onto \( \text{span}\{\mathbf{w}\} \) (a line) has coordinate

\[
\|\mathbf{x}\|_2 \cdot \cos(\theta)
\]

where

\[
\cos(\theta) = \frac{\mathbf{x}^\top \mathbf{w}}{\|\mathbf{w}\|_2 \|\mathbf{x}\|_2}.
\]

(Distance to hyperplane is \( \|\mathbf{x}\|_2 \cdot |\cos(\theta)| \).)

**Decision boundary** is hyperplane (oriented by \( \mathbf{w} \)):

\[
\mathbf{x}^\top \mathbf{w} > 0 \iff \|\mathbf{x}\|_2 \cdot \cos(\theta) > 0 \iff \mathbf{x} \text{ on same side of } H \text{ as } \mathbf{w}
\]
Classification with a hyperplane

Projection of $x$ onto $\text{span}\{w\}$ (a line) has coordinate

$$\|x\|_2 \cdot \cos(\theta)$$

where

$$\cos(\theta) = \frac{x^T w}{\|w\|_2 \|x\|_2}.$$ 

(Distance to hyperplane is $\|x\|_2 \cdot |\cos(\theta)|$.)

**Decision boundary** is hyperplane (oriented by $w$):

$$x^T w > 0 \iff \|x\|_2 \cdot \cos(\theta) > 0 \iff x \text{ on same side of } H \text{ as } w$$

What should we do if we want hyperplane decision boundary that doesn’t (necessarily) go through origin?
Is it always possible to find $\mathbf{w}$ with $\text{sgn}(\mathbf{w}^\top \mathbf{x}_i) = y_i$?
Is it always possible to find a hyperplane separating the data?
(Appending 1 means it need not go through the origin.)

Linearly separable.

Not linearly separable.
Decision boundary with quadratic feature expansion

elliptical decision boundary

hyperbolic decision boundary
Decision boundary with quadratic feature expansion

Same *feature expansions* we saw for linear regression models can also be used here to “upgrade” linear classifiers.
Why not feed our goal into an optimization package, in the form

$$\arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} 1[\text{sign}(\mathbf{w}^T \mathbf{x}_i) \neq y_i]$$
Why not feed our goal into an optimization package, in the form

$$\arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[\text{sign}(\mathbf{w}^T x_i) \neq y_i]$$

- Discrete/combinatorial search; often NP-hard.
Relaxing the ERM problem

Let’s remove one source of discreteness:

\[
\frac{1}{n} \sum_{i=1}^{n} 1[\text{sign}(w^T x_i) \neq y_i] \quad \rightarrow \quad \frac{1}{n} \sum_{i=1}^{n} 1[y_i (w^T x_i) \leq 0].
\]

Did we lose something in this process? Should it be “>” or “≥”? 
Relaxing the ERM problem

Let’s remove one source of discreteness:

\[
\frac{1}{n} \sum_{i=1}^{n} 1[\text{sign}(w^T x_i) \neq y_i] \quad \rightarrow \quad \frac{1}{n} \sum_{i=1}^{n} 1[y_i(w^T x_i) \leq 0].
\]

Did we lose something in this process? Should it be “>” or “≥”?

\(y_i(w^T x_i)\) is the (unnormalized) margin of \(w\) on example \(i\);
we have written this problem with a margin loss:

\[
\hat{R}_{zo}(w) = \frac{1}{n} \sum_{i=1}^{n} \ell_{zo}(y_i w^T x_i) \quad \text{where} \quad \ell_{zo}(z) = 1[z \leq 0].
\]

(remainder of lecture will use single-parameter margin losses.)
2. Logistic loss and risk
Logistic loss

Let’s state our classification goal with a generic margin loss $\ell$:

$$\hat{R}(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i w^T x_i);$$

the key properties we want:

- $\ell$ is continuous;
- $\ell(z) \geq c \mathbb{1}[z \geq 0] = c \ell_{zo}(z)$ for some $c > 0$ and any $z \in \mathbb{R}$, which implies $\hat{R}_\ell(w) \geq c \hat{R}_{zo}(w)$.
- $\ell'(0) < 0$ (pushes stuff from wrong to right).
Logistic loss

Let’s state our classification goal with a generic margin loss $\ell$:

$$\hat{\mathcal{R}}(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(y_i w^T x_i);$$

the key properties we want:

- $\ell$ is continuous;
- $\ell(z) \geq c \mathbb{1}[z \geq 0] = c \ell_{zo}(z)$ for some $c > 0$ and any $z \in \mathbb{R}$, which implies $\hat{\mathcal{R}}_{\ell}(w) \geq c \hat{\mathcal{R}}_{zo}(w)$.
- $\ell'(0) < 0$ (pushes stuff from wrong to right).

Examples.

- **Squared loss**, written in margin form: $\ell_{ls}(z) := (1 - z)^2$; note $\ell_{ls}(y\hat{y}) = (1 - y\hat{y})^2 = y^2(1 - y\hat{y})^2 = (y - \hat{y})^2$.
- **Logistic loss**: $\ell_{log}(z) = \ln(1 + \exp(-z))$. 
Squared and logistic losses on linearly separable data

\[ \sum \begin{bmatrix} x \end{bmatrix} = 0 \]

Logistic loss.  
Squared loss.
Squared and logistic losses on linearly separable data II

Logistic loss.

Squared loss.
Logistic risk and separation

If there exists a perfect linear separator, empirical logistic risk minimization should find it.

**Theorem.**
If there exists a perfect linear separator, empirical logistic risk minimization should find it.

**Theorem.** If there exists $\bar{w}$ with $y_i \bar{w}^T x_i > 0$ for all $i$, then every $w$ with $\hat{R}_{\text{log}}(w) < \frac{\ln(2)}{2n} + \inf_v \hat{R}_{\text{log}}(v)$, also satisfies $y_i w^T x_i > 0$.
Logistic risk and separation

If there exists a perfect linear separator, empirical logistic risk minimization should find it.

**Theorem.** If there exists \( \bar{w} \) with \( y_i \bar{w}^T x_i > 0 \) for all \( i \), then every \( w \) with \( \hat{\mathcal{R}}_{\log}(w) < \ln(2)/2n + \inf_v \hat{\mathcal{R}}_{\log}(v) \), also satisfies \( y_i w^T x_i > 0 \).

**Proof.**

**Step 1:** low risk implies few mistakes. For any \( w \) with \( y_i w^T x_i \leq 0 \) for some \( j \),

\[
\hat{\mathcal{R}}_{\log}(w) \geq \frac{1}{n} \ln(1 + \exp(-y_j w^T x_j)) \geq \frac{\ln(2)}{n}.
\]

By contrapositive, any \( w \) with \( \hat{\mathcal{R}}_{\log}(w) < \ln(2)/n \) makes no mistakes.

**Step 2:** \( \inf_v \hat{\mathcal{R}}_{\log}(v) = 0 \). Note:

\[
0 \leq \inf_v \hat{\mathcal{R}}_{\log}(v) \leq \inf_{r > 0} \frac{1}{n} \sum_{i=1}^n \ln(1 + \exp(-ry_i \bar{w}^T x_i)) = 0.
\]

This completes the proof.
3. Minimizing the empirical logistic risk
Least squares and logistic ERM

**Least squares:**
- Take gradient of $\|Aw - b\|^2$, set to 0;
  - obtain normal equations $A^T Aw = A^T b$.
- One choice is *minimum norm solution* $A^+ b$. 

Remark: Is $A^+ b$ a "closed form expression"?
Least squares:

- Take gradient of $\|Aw - b\|^2$, set to 0; obtain normal equations $A^T Aw = A^T b$.
- One choice is minimum norm solution $A^+ b$.

Logistic loss:

- Take gradient of $\mathcal{R}_{log}(w) = \frac{1}{n} \sum_{i=1}^n \ln(1 + \exp(y_i w^T x_i))$ and set to 0.
Least squares:

- Take gradient of $\|Aw - b\|^2$, set to 0;
  
  obtain normal equations $A^T Aw = A^T b$.

- One choice is minimum norm solution $A^+ b$.

Logistic loss:

- Take gradient of $\hat{R}_{\text{log}}(w) = \frac{1}{n} \sum_{i=1}^{n} \ln(1 + \exp(y_i w^T x_i))$ and set to 0 ???

Remark. Is $A^+ b$ a “closed form expression”?
Decreasing $\hat{R}$

We need to move down the contours of $\hat{R}_{\text{log}}$: 

![Graph showing decreasing $\hat{R}$ contours](image-url)
Gradient descent

Given a function $F : \mathbb{R}^d \to \mathbb{R}$, gradient descent is the iteration

$$w_{i+1} := w_i - \eta_i \nabla_w F(w_i),$$

where $w_0$ is given, and $\eta_i$ is a learning rate / step size.
Given a function $F : \mathbb{R}^d \rightarrow \mathbb{R}$, gradient descent is the iteration

$$w_{i+1} := w_i - \eta_i \nabla w F(w_i),$$

where $w_0$ is given, and $\eta_i$ is a learning rate / step size.

Does this work for least squares?
Gradient descent

Given a function $F : \mathbb{R}^d \rightarrow \mathbb{R}$, gradient descent is the iteration

$$w_{i+1} := w_i - \eta_i \nabla_w F(w_i),$$

where $w_0$ is given, and $\eta_i$ is a learning rate / step size.

Does this work for least squares?
Later we’ll show it works for least squares and logistic regression due to convexity.
Gradient descent is the iteration: $\mathbf{w}_{i+1} := \mathbf{w}_i - \eta_i \nabla_{\mathbf{w}} \hat{R}_\log(\mathbf{w}_i)$.

- Note $\ell'_\log(z) = \frac{1}{1+\exp(z)}$, and use the chain rule (hw1!).
- Or use pytorch:

```python
def GD(X, y, loss, step = 0.1, n_iters = 10000):
    w = torch.zeros(X.shape[1], requires_grad = True)
    for i in range(n_iters):
        l = loss(X, y, w).mean()
        l.backward()
        with torch.no_grad():
            w -= step * w.grad
            w.grad.zero_()
    return w
```
The (negative) derivative $-\ell'_{\log}(z) = \frac{1}{1+e^z}$ is the logistic function.

We’ll explain its significance in subsequent lectures.
4. Summary
What data science methods are used at work?

Logistic regression is the most commonly reported data science method used at work for all industries except **Military and Security** where Neural Networks are used slightly more frequently.
Summary

- **Linearly separable** classification problems.
- **Logistic loss** $\ell_{\text{log}}$ and (empirical) risk $\hat{R}_{\text{log}}$.
- **Gradient descent**.