ML and Opt  Lecture 6

- Cross-entropy loss for fitting multiclass logistic regression
- Train/validation/test splits for model fitting, selection, and generalization reporting
- Linear algebra primer
MCE for Categorical Model

\[ y | x \sim \text{Categorical}\left( p_1(x), \ldots, p_K(x) \right) \]

where

\[ p_i(x) = \frac{e^{\Theta^T_i x}}{Z_\Theta(x)} \quad \text{where} \quad Z_\Theta(x) = \sum_{i=1}^{K} e^{\Theta^T_i x} \]

recall

\[ p_\Theta(x) = \left[ \begin{array}{c} p_1(x) \\ \vdots \\ p_K(x) \end{array} \right] = \frac{e^{\Theta x}}{1^T e^{\Theta x}} = \text{softmax}(\Theta x) \]

where \( \Theta = \left[ \begin{array}{c} \Theta_1^T \\ \vdots \\ \Theta_K^T \end{array} \right] \)
Given \( \{ (y_i, x_i) \}_{i=1}^n \), where \( y_i \in \mathbb{R}^k \) one-hot encodes the class of the \( i \)th example (e.g. class 1 encodes as \( [1, 0, \ldots, 0] \) and class \( k \) as \( [0, \ldots, 0, 1] \)), our goal is to learn the model parameters \( \Theta \in \mathbb{R}^{k \times d} \) using HLE:

\[
L(\Theta) = \frac{-1}{n} \sum_{i=1}^{n} \log p_\Theta(y_i | x_i)
\]

Recall

\[
p_\Theta(y_i = e_j | x_i) = e_j^T p_\Theta(x_i) = y_i^T p_\Theta(x_i) = y_i^T \text{softmax}(\Theta x_i)
\]
So

\[ L(\theta) = \frac{1}{n} \sum_{i=1}^{n} - \log P(y_i | x_i) \]

\[ = \frac{1}{n} \sum_{i=1}^{n} - \log [y_i^T \text{softmax}(\Theta x_i)] \]

\[ = \frac{1}{n} \sum_{i=1}^{n} - y_i^T \log \left[ \text{softmax}(\Theta x_i) \right] \]

apply log entrywise to its vector argument

\[ \Rightarrow \quad \frac{1}{n} \sum_{i=1}^{n} \ell(\text{softmax}(\Theta x_i), y_i) \]

\[ \text{what is the correct choice of } \ell \text{ to make the NLL have the form of an empirical risk} \]
Guess: $l$ is the cross-entropy loss

What is cross-entropy? Easiest to first define $\text{KL}$-divergence

$$D_{\text{KL}}(p \| q) = \sum p_i \ln \frac{p_i}{q_i} \quad \text{if } p \text{ and } q \text{ are two probability vectors}$$

$$= \sum p_i \ln p_i - \sum p_i \ln q_i$$

so minimizing $H(p, q)$ w.r.t. $q$ minimizes $D_{\text{KL}}(p \| q)$ w.r.t. $q$
Recall the negative log-likelihood w.r.t. the $i$th sample

$$-y_i^T \log \left( \text{softmax} \left( \Theta x_i \right) \right)$$

$$= \sum_k (y_i)_k \log \left( p_\Theta(x_i)_k \right)$$

$$= H(y_i, p_\Theta(x_i))$$

Note that if the $i$th example is from class $j$, so $y_i = e_j$, then

$$-y_i^T \log \left( \text{softmax} \left( \Theta x_i \right) \right) = -e_j^T \log (p_\Theta(x_i)_j)$$

$$= -\log (p_\Theta(x_i)_j)$$

is the log of the probability assigned to the true class.
Upshot is that

$$L(\Theta) = \frac{1}{n} \sum_{i=1}^{n} H(y_i, \text{softmax}(\Theta x_i))$$

$$= \frac{1}{n} \sum_{i=1}^{n} -\ln[p_{e(x_i)}y_i]$$

So minimizing the NLL learns model parameters $\hat{\Theta}$ that maximize the probabilities assigned by the model to the true classes of the observation, and the loss function for this ERM is

$$l(u, v) = H(u, v)$$
The optimization problem is

\[ \theta^* = \arg\min_{\theta} \mathcal{L}(\theta) \]

\[ = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} -y_i^T \ln \left( \text{softmax}(\theta x_i) \right) \]

\[ = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} -y_i^T \ln \left( \frac{e^{\theta x_i}}{Z_\theta(x_i)} \right) \]

\[ = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left[ \ln(e^{\theta x_i}) - (\ln Z_\theta(x_i))^1 \right] \]

\[ = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \left[ -y_i^T \theta x_i + y_i^T 1 \ln Z_\theta(x_i) \right] \]
\[
\begin{align*}
= \arg\min_{\Theta} & \frac{1}{n} \sum_{i=1}^{n} \left[-y_{i}^{T} \Theta x_{i} + \ln Z_{\Theta}(x_{i})\right] \\
= \arg\min_{\Theta} & \frac{1}{n} \sum_{i=1}^{n} -y_{i}^{T} \Theta x_{i} + \frac{1}{n} \sum_{i=1}^{n} \ln Z_{\Theta}(x_{i}) \\
= \arg\min_{\Theta} & \frac{1}{n} \sum_{i=1}^{n} -y_{i}^{T} \Theta x_{i} + \frac{1}{n} \sum_{i=1}^{n} \ln \left( \sum_{j=1}^{K} e^{\Theta_{j}^{T} x_{i}} \right) 
\end{align*}
\]

This is the optimization problem we solve in practice to learn $\Theta$ using MLE for multiclass classification.
Practical Considerations

Train vs Test vs Validation Splits

\[ \hat{f} = \arg \min_{f \in \mathcal{F}} \sum_{i=1}^{n_{\text{train}}} l(f(x_i), y_i) \]

\[ R_n(f) \]

\[ R(f) = \mathbb{E}_D l(f(x), y) \]

\[ \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} l(f(x_i), y_i) \]

\[ \frac{1}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} l(f(x_i), y_i) \]

We use the test set to estimate the "generalization error"
What if we want to choose between different model classes, e.g. $F_1$ and $F_2$?

- Use training set to find best (empirically) models $\hat{f}_1 \in F_1$ and $\hat{f}_2 \in F_2$

- Use validation dataset to select best of $\hat{f}_1$ and $\hat{f}_2$ on a fresh independent set of samples

- Estimate the generalization gap of this selected model on a fresh independent test set
General flow for supervised ML

- Fix our problem domain > risk (loss) measure
- Collect appropriate data \( \{ (x_i, y_i) \}^n_{i=1} \)
- Split our data into train/validation/test (60/20/20)
- Determine ahead of time a set of models and hyperparameters to try
- Fit each of these using the same training data
- Measure their performance using the validation data and select the best \( f_{opt} \)
- Report an estimate of the gap gap/pop risk of \( f_{opt} \) using the test data
Lin Alg

vectors in $\mathbb{R}^d$, $\mathbb{R}^n$ — column vectors

Linear (in)dependence

$\mathbb{R}^2$: $v$ and $u$ are linearly dependent, because $v = \alpha u$

$\mathbb{R}^3$: $v$ and $z$ are linearly independent, because $v \neq \alpha z$ for any $\alpha$

In general, vectors $v_1, \ldots, v_k \in \mathbb{R}^n$ are linearly dependent if there exist $\alpha_i$ (not all zero) so that

$$\alpha_1 v_1 + \cdots + \alpha_k v_k = 0$$

If on the other hand

$$\alpha_1 v_1 + \cdots + \alpha_k v_k = 0$$

iff $\alpha_i = 0$ for all $i$, then the vectors are linearly independent.
Write \( V = \begin{bmatrix} v_1 & \ldots & v_k \end{bmatrix} \in \mathbb{R}^{n \times k} \)

the vectors are independent iff

\[ V\alpha = 0 \iff \alpha = 0 \]

i.e. \( V \) has full column rank (rank \( k \))

\[ \text{span} \left\{ v_1, \ldots, v_k \right\} = \mathbb{R}^n : \quad x = \alpha_1 v_1 + \ldots + \alpha_k v_k \text{ for some } \alpha_1, \ldots, \alpha_k \in \mathbb{R} \]

\[ = \mathbb{R}^n : \quad x = V\alpha \text{ for } \alpha \in \mathbb{R}^k \]

this is a linear subspace, with dimension given by column rank of \( V \)
bases
Given a subspace \( \Pi \subseteq \mathbb{R}^d \), the vectors \( \frac{\mathbf{v}}{2}, \frac{\mathbf{v}_2}{2}, \ldots, \frac{\mathbf{v}_k}{2} \) is a basis for \( \Pi \) if:
1) \( \Pi = \text{span} \left\{ \frac{\mathbf{v}}{2}, \frac{\mathbf{v}_2}{2}, \ldots, \frac{\mathbf{v}_k}{2} \right\} \)
2) \( \frac{\mathbf{v}}{2}, \frac{\mathbf{v}_2}{2}, \ldots, \frac{\mathbf{v}_k}{2} \) are linearly independent

dot product / inner product
\( \mathbf{u}, \mathbf{v} \in \mathbb{R}^d \):
\[
\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u}^T \mathbf{v} = \sum_{i=1}^{d} u_i v_i
\]
Euclidean / \( L_2 / \| \cdot \|_2 \) norm
\[
\| \mathbf{u} \|_2^2 = \langle \mathbf{u}, \mathbf{u} \rangle = \sum_{i=1}^{d} u_i^2
\]
**Cauchy-Schwarz Inequality**

**Fact**

\[ |\langle u, v \rangle| \leq \|u\|_2 \|v\|_2 \quad \text{and equality occurs iff} \quad u = \alpha v \]

\[ \cos \theta = \frac{\langle u, v \rangle}{\|u\|_2 \|v\|_2} \in [-1, 1] \] by Cauchy-Schwarz, so we use this to define the angle between vectors in high-dimensional spaces.

\[ u \perp v \iff \langle u, v \rangle = 0 \iff \cos \theta = 0 \]
\[ \|u + v\|_a^2 = \langle u + v, u + v \rangle \]
\[ = \langle u, u + v \rangle + \langle v, u + v \rangle \]
\[ = \langle u, u \rangle + \langle u, v \rangle + \langle v, u \rangle + \langle v, v \rangle \]
\[ = \|u\|_a^2 + \|v\|_a^2 + 2 \langle u, v \rangle \]
\[ = \|u\|_a^2 + \|v\|_a^2 + 2 \cos \theta \|u\|_a \|v\|_a \]

high-dimensional law of cosines

If \( \langle u, v \rangle = 0 \), then
\[ \|u + v\|_a^2 = \|u\|_a^2 + \|v\|_a^2 \]  
Pythagorean theorem
**Outer Product**

For \( v \in \mathbb{R}^n \) and \( U \in \mathbb{R}^{n \times k} \) then outer product

\[
Uv^T = \sum_{i=1}^{k} u_i v_i^T \in \mathbb{R}^{n \times d}
\]

\[
(uv^T)_{ij} = u_i v_j
\]
**symmetric matrices**

A square matrix $M$ is symmetric if $M = M^T$  

$\iff M_{ij} = M_{ji}$ for all $i,j$

**eigenpairs**

$(\lambda, v)$ is an eigenpair ($\lambda \in \mathbb{R}$ is an eigenvalue of $M \in \mathbb{R}^{n \times n}$ and $v \in \mathbb{R}^d$ is an eigenvector) iff

$$Mv = \lambda v$$

Fact:

1) all eigenvectors of symmetric matrices are perpendicular to each other

2) all eigenvalues of symmetric matrices are real

3) we can write $M = \sum_{i=1}^{n} \lambda_i v_i v_i^T$ where $(\lambda_i, v_i)$ are eigenpairs
Note: if we write

\[ V = \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix} \in \mathbb{R}^{m \times n} \] collection of

\[ \Lambda = \begin{bmatrix} \lambda_1 & \cdots & \lambda_n \end{bmatrix} \in \mathbb{R}^{n \times n} \] (length 1) eigenvectors

\[ M = V \Sigma V^T \] of \( M \)

a diagonal matrix \( \omega \) with

corresponding eigenvalues
Geometric Interpretation

Associate each symmetric matrix $M$ with a quadratic form $M : x \mapsto x^T M x = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j M_{ij}$

$x^T \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} x = x_1^2 - x_2^2$

$v_0 = x^T \begin{bmatrix} 1 & 1 \end{bmatrix} x = x_1^2 + x_2^2$

Geometrically, $M = \sum_{i=1}^{n} \lambda_i v_i v_i^T$, the quadratic form curves upwards in the directions in the span of $v_i$ for which the $\lambda_i$ are positive, and downward in directions for which $\lambda_i$ are negative.
Positive Semidefiniteness

We say a symmetric matrix $A = \sum \lambda_i v_i v_i^T$ is positive semidefinite (PSD) or positive, written $A \succeq 0$, if $\lambda_i \geq 0$ for all $i$.

Positive definite, written $A \succ 0$ if $\lambda_i > 0$ for all $i$.

Facts

1) $A$ is pd iff the corresponding quadratic form $x^T A x \geq 0$ for all $x$.
2) If $A$ is pd then $C A C^T \succeq 0$.
3) For any matrix $A$, $AA^T \succeq 0$ and $A^T A \succeq 0$. 