Overview

- Statistical model for regression problems
- Linear regression models
- MLE and ERM

Real-valued predictions I

Figure 1: Galton board

Real-valued predictions II

- Physical model: hard
- Statistical model: final position of ball is random
  - *Normal (Gaussian) distribution* with mean $\mu$ and variance $\sigma^2$
  - Written $N(\mu, \sigma^2)$
- Goal: predict final position accurately, measure *squared loss* (also called *squared error*)
  $$(\text{prediction} - \text{outcome})^2$$
- Note: outcome is random, so look at *expected squared loss* (also called *mean squared error*)
Optimal prediction for mean squared error

- Predict $\hat{y} \in \mathbb{R}$; true final position is $Y$ (random variable) with mean $E(Y) = \mu$ and variance $\text{var}(Y) = E[(Y - E(Y))^2] = \sigma^2$.
- Squared error is $(\hat{y} - Y)^2$.
- Bias-variance decomposition:

So optimal prediction is $\hat{y} =$

- When parameters are unknown, can estimate from related data, ...

Example: Old Faithful I

- Example: When will “Old Faithful” geyser erupt?
- Predict “time between eruptions”
- Old Faithful Geyser Data

- Mean on past 136 observations: $\hat{\mu} = 70.7941$ minutes
  - So predict $\hat{y} = \hat{\mu} = 70.7941$

- Mean squared error on next 136 observations: 187.1894
  - Square root: 13.6817 minutes

Looking at the data

- Henry Woodward observed that “time between eruptions” seems related to “duration of latest eruption”

- Use “duration of latest eruption” as feature $x$
- Can use $x$ to predict time until next eruption, $y$
Statistical model for regression

- Setting is same as for classification except:
  - Label is real number, rather than \(\{0, 1\}\) or \(\{1, 2, \ldots, K\}\)
  - Care about squared error, rather than whether prediction is correct
  - Risk of \(f\):
    \[
    \mathcal{R}(f) := \mathbb{E}[(f(X) - Y)^2],
    \]
    the expected squared loss of \(f\) on random example
- Note: “error rate” is also “risk”, but with different loss function, called zero-one loss \(\mathbb{I}\{f(x) \neq y\}\)

Optimal prediction function for regression

- If \((X, Y)\) is random test example, then optimal prediction function is
  \[
  f^*(x) = \mathbb{E}[Y \mid X = x]
  \]
- Also called the regression function
- Prediction function with smallest risk
- Depends on conditional distribution of \(Y\) given \(X\)

Linear regression models

- Suppose \(x\) is given by \(d\) real-valued features, so \(x \in \mathbb{R}^d\)
- Linear regression model for \((X, Y)\):
  - \(Y \mid X = x \sim N(x^T w, \sigma^2)\) (or really, any distribution with mean \(x^T w\) and variance \(\sigma^2\))
  - \(w \in \mathbb{R}^d\) is parameter vector of interest
  - \(\sigma^2 > 0\) is another parameter (not important for prediction)
  - \(w\) and \(\sigma^2\) not involved in marginal distribution of \(X\) (which we don’t care much about)

Upgrading linear regression

- Make linear regression more powerful by being creative about features
- Instead of using \(x\) directly, use \(\varphi(x)\) for some transformation \(\varphi\) (possibly vector-valued)
- Examples:
  - Non-linear scalar transformations, e.g., \(\varphi(x) = \ln(1 + x)\)
  - Logical formula, e.g., \(\varphi(x) = (x_1 \land x_5 \land \neg x_{10}) \lor (\neg x_2 \land x_7)\)
  - Trigonometric expansion, e.g., \(\varphi(x) = (1, \sin(x), \cos(x), \sin(2x), \cos(2x), \ldots)\)
  - Polynomial expansion, e.g., \(\varphi(x) = (1, x_1, \ldots, x_d, x_1^2, \ldots, x_d^2, x_1 x_2, \ldots, x_{d-1} x_d)\)
  - Headless neural network \(\varphi(x) = N(x) \in \mathbb{R}^k\), where \(N: \mathbb{R}^d \to \mathbb{R}^k\) is a map computed by a intermediate layer of a neural network
Example: Taking advantage of linearity

- Example: $y$ is health outcome, $x$ is body temperature
  - Physician suggests relevant feature is (square) deviation from normal body temperature $(x - 98.6)^2$
  - What if you didn’t know the magic constant 98.6?

Example: Affine expansion

- Another example: Woodward used affine expansion
  - $\varphi(x) = (1, x)$
  - Parameter vector $w = (a, b)$
  - $\varphi(x)^T w = a + bx$, so $a$ is intercept term
  - Generalizes to $d$ features: just prepend the constant 1 feature
    $\varphi(x) = (1, x) \in \mathbb{R}^{d+1}$

Text features

- How to get features for text?
  - Suppose input is a word (sequence of characters).
    - $x_{\text{starts with \_anti}} = 1$ (starts with “anti”)
    - $x_{\text{ends with \_ology}} = 1$ (ends with “ology”)
    - ... (same for all four- & five-letter prefixes & suffixes)
    - $x_{\text{length} \leq 3} = 1$ (length $\leq 3$)
    - $x_{\text{length} \leq 4} = 1$ (length $\leq 4$)
    - ... (same with all positive integers $\leq 20$)
  - Suppose input is a document (sequence of words).
    - $x_{\text{contains \_aardvark}} = 1$ (contains “aardvark”)
    - ... (same for all words in dictionary)
    - $x_{\text{contains \_each \_day}} = 1$ (contains “each day”)
    - ... (same for all “bigrams” of words in dictionary)
    - $x_{\text{count \_aardvark}} = \#$ appearances of “aardvark”
    - ... (same for all words, “bigrams”, ...)
  - End up with many features!

Sparse representations

- Sparse representation (e.g., via hash table)
  - E.g., “see spot run”
    - $x = \{ \text{"contains\_see":1, "contains\_spot":1, "contains\_run":1, "contains\_see\_spot":1, "contains\_spot\_run":1} \}$
  - C.f. dense representation, which stores a lot of zeros for all of the words / bigrams that don’t appear.
  - What is computational cost of computing $x^Tz$?
Fitting linear regression models to data

- Treat training examples as iid, same distribution as test example
  - \( Y \mid X = x \sim N(x^T w, \sigma^2) \)
- Log-likelihood of \((w, \sigma^2)\) given data \((x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^d \times \mathbb{R}\):
  \[
  \sum_{i=1}^{n} \left\{ -\frac{1}{2\sigma^2} (x_i^T w - y_i)^2 + \frac{1}{2} \ln \frac{1}{2\pi \sigma^2} \right\} + \{ \text{terms not involving } (w, \sigma^2) \}
  \]
- The \(w\) that maximizes log-likelihood is same \(w\) that minimizes
  \[
  \frac{1}{n} \sum_{i=1}^{n} (x_i^T w - y_i)^2.
  \]

MLE coincides with ERM

- Empirical distribution \(P_n\) on \((x_1, y_1), \ldots, (x_n, y_n)\): distribution that puts probability mass \(1/n\) on each training example.
- Execute the plug-in principle:
  - We want to find \(f: \mathbb{R}^n \rightarrow \mathbb{R}\) that minimizes risk
    \[
    \mathcal{R}(f) = \mathbb{E}[(f(X) - Y)^2],
    \]
    but we don’t know distribution \(P\) of \((X, Y)\) (or even conditional distribution of \(Y\) given \(X\)).
  - Replace \(P\) with \(P_n\) to get empirical risk
    \[
    \hat{\mathcal{R}}(f) := \frac{1}{n} \sum_{i=1}^{n} (f(x_i) - y_i)^2,
    \]
  - which is the risk of \(f\) pretending that the distribution of \((X, Y)\) is \(P_n\).
  - So find \(f\) to minimize empirical risk: **Empirical Risk Minimizer (ERM)**
    - For linear functions \(f(x) = x^T w\), same as MLE for \(w\) in linear regression model (!!)

Geometric picture of empirical risk

![Figure 6: Empirical risk of \(w\) is average of vertical squared distances from hyperplane to data points](image)

ERM in matrix notation

- Let \(A = \frac{1}{\sqrt{n}} \begin{bmatrix} x_1^T & \ldots & x_n^T \end{bmatrix} \in \mathbb{R}^{n \times d}\) and \(b = \frac{1}{\sqrt{n}} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n\)
- Empirical risk is
  \[
  \hat{\mathcal{R}}(w) = \frac{1}{n} \sum_{i=1}^{n} (x_i^T w - y_i)^2 = \|Aw - b\|_2^2.
  \]
Normal equations

- From calculus:
  - Necessary condition for \( w \) to be minimizer of \( \hat{R} \) is that gradient of \( \hat{R} \) at \( w \) should vanish: \( \nabla \hat{R}(w) = 0 \)
  - Equivalent to \( (A^T A)w = A^T b \)
  - System of linear equations in \( w \), called the normal equations
  - Every solution \( w \) to normal equations is a minimizer of \( \hat{R} \):

Algorithm for ERM

- Algorithm for finding ERM: Gaussian elimination to solve normal equations
  - Running time \( O(nd^2) \)
  - Can get good approximate solution in linear time \( O(nd) \)
  - Also called Ordinary Least Squares (OLS)

Linear algebraic interpretation of ERM

- Write \( A = \begin{bmatrix}
\uparrow & \cdots & \uparrow \\
\downarrow & & \downarrow \\
\end{bmatrix} \)
  - \( a_j \in \mathbb{R}^n \) is \( j \)-th column of \( A \)
  - Span of \( a_1, \ldots, a_d \) is \( \text{range}(A) \), a subspace of \( \mathbb{R}^n \)
  - Minimizing \( \| A w - b \|^2 \) over \( w \in \mathbb{R}^d \) is same as finding vector \( \hat{b} \) in \( \text{range}(A) \) closest to \( b \)
  - Solution \( \hat{b} \) is orthogonal projection of \( b \) onto \( \text{range}(A) \)

Performance of ERM

- How well does ERM solution \( \hat{w} \) work?
  - Study in context of IID model
  - Best linear predictor \( w^* \): minimizer of \( R(w) \).
  - Hope that \( R(\hat{w}) \approx R(w^*) \)

- **Theorem**: In IID model, ERM solution \( \hat{w} \) satisfies
  \[
  R(\hat{w}) \to R(w^*) + \frac{\text{tr}(\text{cov}(\varepsilon W))}{n}
  \]
  as \( n \to \infty \), where \( W = \mathbb{E}[XX^T]^{-1/2} X \) and \( \varepsilon = Y - X^T w^* \).

- If \( (X, Y) \) follows linear regression model \( Y \mid X = x \sim N(x^T w^*, \sigma^2) \),
  then theorem simplifies to
  \[
  R(\hat{w}) \to R(w^*) + \frac{\sigma^2 d}{n} = \left( 1 + \frac{d}{n} \right) \sigma^2.
  \]
Risk vs empirical risk

- Let \( \hat{w} \) be ERM solution.
- How do \( \hat{R}(\hat{w}) \) and \( R(\hat{w}) \) compare?
- **Theorem:** In IID model, \( E[\hat{R}(\hat{w})] \leq E[R(\hat{w})] \)

**Over-fitting:** when true risk is much higher than empirical risk.

- Note: Can estimate risk using test set, just as for classification problems.

Example of over-fitting

- \( \varphi(x) = (1, x, x^2, \ldots, x^k) \), degree-\( k \) polynomial expansion
- Dimension is \( d = k + 1 \)
- Any function of \( \leq k + 1 \) points can be interpolated by polynomial of degree \( \leq k \)
- So if \( n \leq k + 1 = d \), ERM solution \( \hat{w} \) will have \( \hat{R}(\hat{w}) = 0 \), even if true risk is \( \gg 0 \).

Outliers

- Common issue with using squared loss: sensitive to outliers
- Roughly: data points that don’t fit the same pattern as the rest
- Does removing the data point drastically change the fit?

Absolute loss

- One “fix”: change loss function
- Common choice: **absolute loss** \( |\hat{y} - y| \)
- Instead of solving linear system, now solve a linear program
- Less sensitive to abnormal \( y \)-values than squared loss
- However: changes what we are estimating ...
Heuristics for dealing with outliers

- **Heuristic I: random sample consensus (RANSAC)**
  - Pick a random subsample of data points — hopefully no outliers are picked! — and fit model to this subsample
  - If most of the remaining data are "well-fit", then halt
  - Else, try again

- **Heuristic II: iterative trimming**
  - Fit training data as usual
  - Throw out some of the least “well-fit” data points
  - Repeat until fit does not change too much

- Both heuristics are rather drastic!
  - What if outliers correspond to a subpopulation?
  - Should manually examine the putative outliers

Beyond empirical risk

- **Recall plug-in principle**
  - Want to minimize risk wrt ( unavailable) $P$; use $P_n$ instead

- **What if we can’t regard data as iid from $P$?**
  - Example: Suppose we know $P = 0.5M + 0.5F$ (mixture distribution)
  - We get size $n_1$ iid sample from $M$, and size $n_2$ iid sample from $F$, $n_2 \ll n_1$
  - How to implement plug-in principle?