Working with Data & Regression

CS57300 – Data Mining
Purdue University

Instructor: Bruno Ribeiro
Goals:

- Understand How to Work with (Real) Data
- Review Linear Regression
  - Regression as a Naïve Predictor
Working with Data
Data representation is key to success
(and a source of headaches if not correctly done)
E.g.: Representing Fractions

- Consider creating a model that predicts if a soccer striker will score a goal in a game
- Data includes *no. shots on goals* and *no. goals* during the player's career

- Problems with absolute values:
  - number of goals
    (older players have larger values than young players)
  - no. shots on goals
    (does not reflect rate of shot $\rightarrow$ goal conversion)

- Feature: % shots on goal resulting in goal
  - Alice (Novice): 1 out of 1 $\rightarrow$ 100%
  - Bob (Senior): 300 out of 1000 $\rightarrow$ 30%

- Solution? (Bayesian as we will see)
Finding (Real) Patterns in Data

- **Data shows:** the top high school classrooms with highest average SAT scores are all small (few students)

- Why rural counties have the highest rates of cancer?
  - A: Small sample variance
  - Solution? (Also Bayesian)
Data Representation Issues

- How to represent the data?
- E.g.: Are product reviews (stars) really integers between 5 and 1?

![Customer Reviews](image)

© Bruno Ribeiro
Geometric Distribution

- Reviews geometrically distributed?

\[ P[X = k | p] = (1 - p)^{k-1} p \]
Normal Distribution

- Reviews normally distributed?

\[ P[X = x | \mu, \sigma] = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \]
Binomial distribution

- Reviews binomially distributed?

\[ P[X = k|p] = \binom{n}{k} p^k (1 - p)^{n-k} \]
Poisson distribution

- Reviews Poisson distributed?

\[ P[X = k | \lambda] = \frac{\lambda^k e^{-\lambda}}{k!} \]
Beta distribution

- Reviews beta distributed?

\[ P[X = x | \alpha, \beta] = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha) \Gamma(\beta)} x^{\alpha-1} (1 - x)^{\beta-1} \]
Feature Space

(Representing Data)
Datasets with same fixed set of numeric attributes can be thought of as points in a multi-dimensional space
  ◦ Each dimension represents a distinct attribute

Variety of techniques can use similarity/dissimilarity measures to characterize relationships between the instances
Many data mining techniques utilize similarity/dissimilarity measures to characterize relationships between instances

- Nearest-neighbor classification
- Cluster analysis

**Proximity**: general term to indicate similarity and dissimilarity

**Distance**: dissimilarity only
Metric properties

- A **metric** $d(x,y)$ is a dissimilarity measure that satisfies the following properties:
  
  - $d(x,y) \geq 0$ for all $x,y$ and $d(x,y) = 0$ iff $x = y$ **Positive**
  - $d(x,y) = d(y,x)$ for all $x,y$ **Symmetric**
  - $d(x,y) \leq d(x,k) + d(k,y)$ for all $x,y,k$ **Triangle inequality**
Distance metrics

- **Manhattan distance** (L1)

  \[ d_M(x, y) = \sum_{i=1}^{k} |x_i - y_i| \]

- **Euclidean distance** (L2)

  \[ d_E(x, y) = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \]
  - Most common metric
  - Assumes variables are commensurate

- **Weighted** Euclidean distance

  \[ d_E(x, y) = \sqrt{\sum_{i=1}^{k} w_i (x_i - y_i)^2} \]
  - Can weight variables by relative importance
Standardization (dealing with features of different scales)

- Normalization
  - Removes effect of scale
  - Divide each variable by its standard deviation
  - Weights all variables equally

\[
x'_i = \frac{x_i - \bar{x}_i}{\hat{\sigma}_i}
\]

subtract mean
divide by stdev

\[
d_E(x, y) = \sqrt{\sum_{i=1}^{k} (x'_i - y'_i)^2}
\]

Features:
- Gender: M=0, F=1
- Income: 10k – 10M

Data:
- Employee1 = (0, 30k)
- Employee2 = (1,60k)

Without normalization income dominates Euclidean distance
Covariance and correlation
(examples of distance measures between random variables)
Covariance

- Measures how variables $X_j$ and $X_k$ vary together

$$COV(X_j, X_k) = \frac{1}{n} \sum_{i=1}^{n} (x_{ij} - \bar{X}_j)(x_{ik} - \bar{X}_k)$$

- Positive if large values of $X_j$ are associated with large values of $X_k$
- Negative if large values of $X_j$ are associated with small values of $X_k$

Covariance matrix ($\Sigma$)

- Symmetric matrix of covariances for $p$ variables

Measures linear relationship
Correlation coefficient

- Covariance depends on ranges of $X_j$ and $X_k$
- Correlation standardizes covariance by dividing through standard deviations

$$\rho(X_j, X_k) = \frac{1}{n} \sum_{i=1}^{n} (x_{ij} - \bar{X}_j)(x_{ik} - \bar{X}_k)$$

- Correlation matrix
  - Symmetric matrix of correlations for $p$ variables
  - What values are on the diagonal?
Issues with multidimensional distance metrics?

- Dimensions (features) may be correlated
- E.g.:
  - Gender and income are correlated
  - Age and income are correlated
  - Equity and income are correlated
Correlation inflates distance

- Normalization helps little if many features are correlated

\[ x'_i = \frac{x_i - \bar{x}_i}{\hat{\sigma}_i} \]  
  subtract mean 
  divide by stdev

- Solution?
Mahalanobis distance

\[ d_{MH}(x, y) = \sqrt{(x - y)^T \Sigma^{-1} (x - y)} \]

- Automatically accounts for scaling
- Corrects for correlation between attributes
- Tradeoff:
  - Covariance matrix can be hard to estimate accurately
  - Memory and time complexity is quadratic rather than linear
Distance measures for binary data

- $d(x, y)$ when items $x$ and $y$ are $p$-dimensional binary vectors

- Let $n_{11}$ be the number of attributes where both items have value 1, etc.

$$n_{11} = \sum_i^p \mathbb{I}(x_i + y_i = 2)$$

- Matching coefficient
  - Hamming distance normalized by number of bits

$$d_{MC}(x, y) = \frac{n_{11} + n_{00}}{n_{11} + n_{00} + n_{10} + n_{01}}$$

- Jaccard coefficient
  - If we don’t care about matches on zeros

$$d_{JC}(x, y) = \frac{n_{11}}{n_{11} + n_{00} + n_{10} + n_{01}}$$
More data issues:

Sometimes data is missing
Missing values

- Reasons for missing values
  - Information is not collected (e.g., people decline to give their age)
  - Attributes may not be applicable to all cases (e.g., annual income is not applicable to children)

- Ways to handle missing values
  - Eliminate entities with missing values
  - Estimate attributes with missing values
  - Ignore the missing values during analysis
  - Replace with all possible values (weighted by their probabilities)
  - Impute missing values
Duplicate Data

- Data set may include data entities that are duplicates, or almost duplicates of one another
  - Major issue when merging data from heterogeneous sources
  - Example: same person with multiple email addresses

- Sometimes “duplication” happens (different users, same features).
  - Issues with some models.

- Data cleaning
  - Finding and dealing with duplicate entities
  - Finding and correcting measurement error
  - Dealing with missing values
Predictions From Data
Naïve Prediction: Linear Regression
Linear Regression (use A)

- Interpolation
  (something is missing)
- \((x_1, \ldots, x_t)\)
- \((y_1, \ldots, y_t)\)
Auto-regression: Predicting Next Value After t Steps
Linear Regression (use B)

Google Finance

\[ x_{t+1} = \sum_{i=1}^{t} a_i x_i + \epsilon_{\text{noise}} \]

Similar problem to linear regression: express unknowns as a linear function of knowns
Predictions from High-Dimensional Historical Data

- \( X_{[t \times w]} \cdot a_{[w \times 1]} = y_{[t \times 1]} \)

- Over-constrained problem
  - \( a \) is the vector of the regression coefficients
  - \( X \) has the \( t \) values of the \( w \) indep. variables
  - \( y \) has the \( t \) values of the dependent variable

Ack: C. Faloutosos
Looking Into Multiplication

\[ X_{[t \times w]} \cdot a_{[w \times 1]} = y_{[t \times 1]} \]

Predicting corn prices over time…

\[
\begin{bmatrix}
X_{11}, X_{12}, \ldots, X_{1w} \\
X_{21}, X_{22}, \ldots, X_{2w} \\
\vdots \\
X_{t1}, X_{t2}, \ldots, X_{tw}
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\vdots \\
a_w
\end{bmatrix}
= 
\begin{bmatrix}
y_1 \\
y_2 \\
\vdots \\
y_t
\end{bmatrix}
\]

may want to add social media variables

© Bruno Ribeiro
How to Estimate $a$?

- $a = (X^T \cdot X)^{-1} \cdot (X^T \cdot y)$

$X^+ = (X^T \cdot X)^{-1} \cdot X^T$ is the Moore–Penrose pseudoinverse

Or: $a = X^+ y$

$a$ is the vector that minimizes the Root Mean Squared Error (RMSE) of $(y - X \cdot a^T)$
Details: Least Squares Optimization

- Least squares cost function:

\[
C = \frac{1}{2} \sum_{i=1}^{t} (y_i - x_i^T a)^2 = \frac{1}{2} (y - Xa)^T (y - Xa)
\]

- Find \( a \) that minimizes cost \( C \)

\[
\frac{\partial C}{\partial a} = \frac{1}{2} \frac{\partial}{\partial a} (y - Xa)^T (y - Xa) = -(y - Xa)^T X
\]

- Optimal value at:

\[
\frac{\partial C}{\partial a} = 0 \implies X^T y = X^T X a \implies a = (X^T X)^{-1} X^T y
\]
How to Estimate \(a\)?

\[
a = (X^T \cdot X)^{-1} \cdot (X^T \cdot y)
\]

\(X^+ = (X^T \cdot X)^{-1} \cdot X^T\) is the Moore–Penrose pseudoinverse

Or: \(a = X^+ y\)

\(a\) is the vector that minimizes the Root Mean Squared Error (RMSE) of \((y - X \cdot a^T)\)

Problems:
Matrix \(X\) grows over time & needs matrix inversion

\[
\text{O}(t \cdot w^2)\text{ computation}
\]
\[
\text{O}(t \cdot w)\text{ storage}
\]
Recursive Least Squares

At time $t$ we know $\mathbf{X}_t = (x_1, \ldots, x_t), \quad \mathbf{y}_t = (y_1, \ldots, y_t)$

Least squares is solving

$$\arg\max_{\mathbf{a}^*} \|\mathbf{a}^T \mathbf{X}_t - \mathbf{y}_t\|^2$$

which gives

$$\mathbf{a}^* = \mathbf{X}^+ \mathbf{y}$$

where $\mathbf{X}^+ = (\mathbf{X}^T \cdot \mathbf{X})^{-1} \cdot \mathbf{X}^T$

Let

$$\Phi_t = \mathbf{X}_t^T \mathbf{X}_t \quad \mathbf{\theta}_t = \mathbf{X}_t^T \mathbf{y}_t$$

Then $\Phi_{t+1}^{-1}$ is

$$\Phi_{t+1}^{-1} = (\Phi_t + \mathbf{x}_{t+1} \mathbf{x}_{t+1}^T)^{-1} = \Phi_t^{-1} - \frac{\Phi_t^{-1} \mathbf{x}_{t+1} \mathbf{x}_{t+1}^T \Phi_t^{-1}}{1 + \mathbf{x}_{t+1}^T \Phi_t^{-1} \mathbf{x}_{t+1}}$$
Recursive Least Squares Algorithm

\[ \Phi_{t+1}^{-1} = \Phi_t^{-1} - \frac{\Phi_t^{-1} x_{t+1}^T x_{t+1} \Phi_t^{-1}}{1 + x_{t+1}^T \Phi_t^{-1} x(t + 1)} \]

\[ \theta_{t+1} = \theta_t + x_{t+1}^T y_{t+1} \]

\[ a_{t+1} = \Phi_{t+1}^{-1} \theta_{t+1} \]
Matrix Inversion Formula

If $A$ and $B$ are $m \times m$ positive definite matrices, $D$ is a $n \times n$ matrix, and $C$ is a $m \times n$ matrix such that

$$A = B^{-1}CD^{-1}C^T,$$

then

$$A^{-1} = B - BC(D + C^TBC)^{-1}C^T B.$$
Exponentially Weighted Recursive Least Squares Algorithm

for \( \lambda > 1 \)

\[
\Phi_{t+1}^{-1} = \frac{1}{\lambda} \Phi_t^{-1} - \frac{1}{\lambda^2} \frac{\Phi_t^{-1} x_{t+1}^T x_{t+1} x_{t+1}^T \Phi_t^{-1}}{1 + x_{t+1}^T \Phi_t^{-1} x(t + 1)}
\]

\[
\theta_{t+1} = \lambda \theta_t + x_{t+1}^T y_{t+1}
\]

\[
a_{t+1} = \Phi_{t+1}^{-1} \theta_{t+1}
\]
Comparison

Original Least Squares
- Needs large matrix (growing in size) $O(t \times w)$
- Costly matrix operation $O(t \times w^2)$

Recursive LS
- Need much smaller, fixed size matrix $O(w \times w)$
- Fast, incremental computation $O(1 \times w^2)$
- No matrix inversion
Other data preprocessing methods

- Sampling
- Dimensionality reduction
- Attribute transformation (e.g., discretization, distance calculations)
- Feature construction and selection
- We will discuss these in more detail throughout the course