Section 2: Midterm review

STATS 202: Statistical Learning and Data Science

Linh Tran

tranlm@stanford.edu

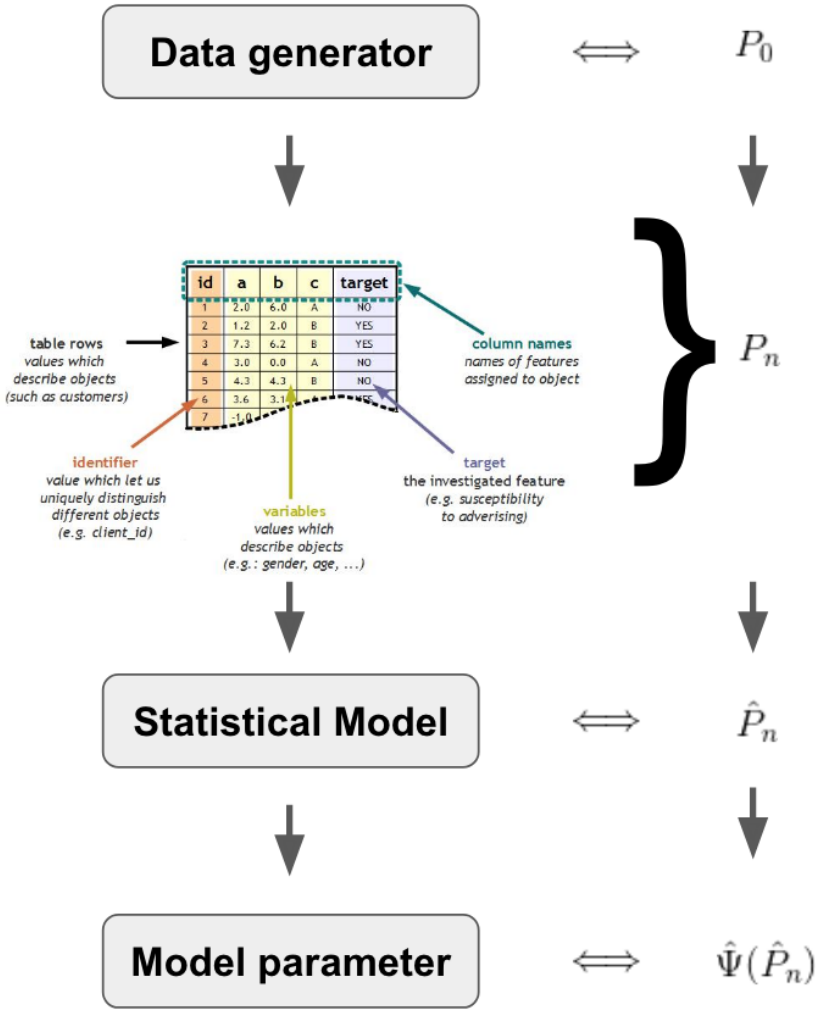
Department of Statistics
Stanford University

July 11, 2025

Announcements

- Homework 2 due next Monday.
- Homework 1 grading complete.
- You should have confirmation if you require any accommodations for the midterm.

Empirical vs true distributions



Ideally, we want $\Psi(P_0)$.

Supervised learning

Motivation: Why learn f_0?

Prediction

- Useful when we can readily get $X_1, X_2, ..., X_p$, but not Y.
- Allows us to predict what Y likely is.
- Example: Predict stock prices next month using data from last year.

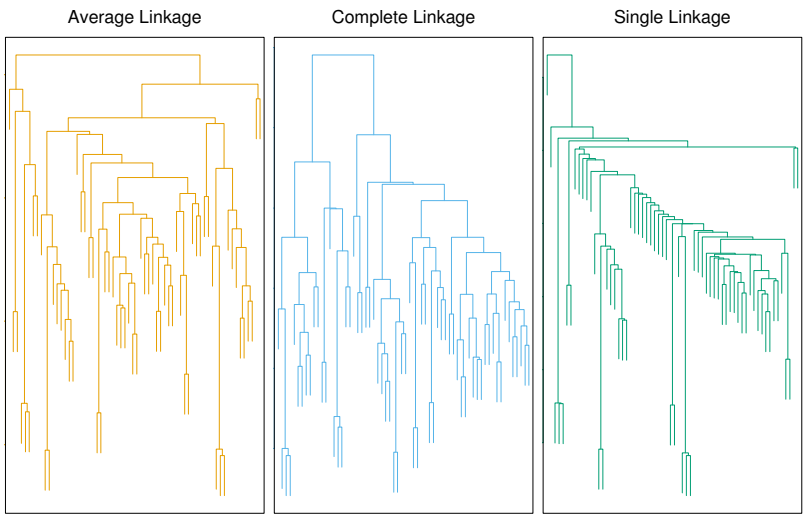
Inference

- Allows us to understand how differences in $X_1, X_2, ..., X_p$ might affect Y.
- Example: What is the influence of genetic variations on the incidence of heart disease.

Unsupervised learning

- In unsupervised learning, all the variables are on equal standing, no such thing as an input and response.
- Clustering is typically applied
  - Hierarchical clustering (single, complete, or average linkage).
  - K-means clustering.
  - Expectation maximization (using Gaussian mixtures).

Hierarchical clustering



- Agglomerative algorithm produces a *dendrogram*.
- At each step we join the two clusters that are "closest":
  - **Complete:** distance between clusters is maximal distance between any pair of points.
  - **Single:** distance between clusters is minimal distance.
  - **Average:** distance between clusters is the average distance.
- Height of a branching point = distance between clusters joined.

K-means clustering

- The number of clusters is fixed at K.
- Goal is to minimize the average distance of a point to the average of its cluster.
- The algorithm starts from some assignment, and is guaranteed to decrease this average distance.
- This find a local minimum, not necessarily a global minimum, so we typically repeat the algorithm from many different random starting points.

Supervised learning

We're interested in a response variable Y associated to each vector of predictors $\mathbf{X}$.

Regression: $f_0 = \mathbb{E}_0[Y|X_1, X_2, ..., X_p]$

- A scalar value, i.e. $f_0 \in \mathbb{R}$
- $\hat{f}_n$ therefore gives us estimates of y

Classification: $f_0 = \mathbb{P}_0[Y = y|X_1, X_2, ..., X_p]$

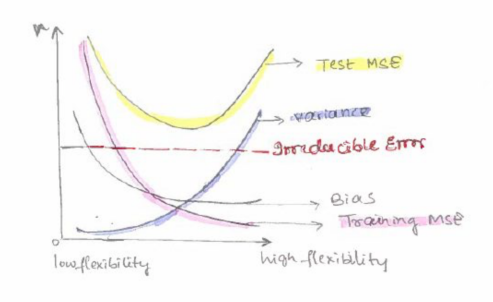
- A vectored value, i.e. $f_0 = [p_1, p_2, ..., p_K] : p_j \in [0, 1], \sum_K p_j = 1$
- n.b. In a binary setting this simplies to a scalar, i.e. $f_0 = p_1 : p_1 = \mathbb{P}_0[Y = 1|X_1, X_2, ..., X_p] \in [0, 1]$
- $\hat{f}_n$ therefore gives us predictions of each class
- Can take the arg max, giving us Bayes Classifier

Bias variance decomposition

Let x_0 be a fixed point, $y_0 = f_0(x_0) + \epsilon$, and $\hat{f}_n$ be an estimate of f_0 from $(x_i, y_i) : i = 1, 2, ..., n$.

The MSE at x_0 can be decomposed as

$$\begin{aligned} MSE(x_0) &= \mathbb{E}_0[y_0 - \hat{f}_n(x_0)]^2 && (1) \\ &= \text{Var}(\hat{f}_n(x_0)) + \text{Bias}(\hat{f}_n(x_0))^2 + \text{Var}(\epsilon_0) && (2) \end{aligned}$$



Regression methods

- Multiple linear regression
- Stepwise selection methods (e.g. forward, backward, etc.)
- Nearest neighbors regression

Multiple linear regression

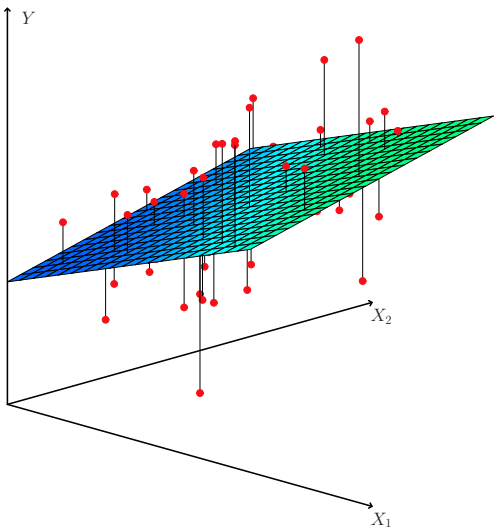
Extension of linear regression to handle multiple predictors

In multiple linear regression, we assume

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \epsilon$$

$$\epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2)$$

$$\mathbb{E}[Y|\mathbf{X}] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots \tag{3}$$



In matrix notation:

$$\mathbb{E}[Y|\mathbf{X}] = \mathbf{X}\boldsymbol{\beta} \tag{4}$$

where

$$\mathbf{X} = (1, X_1, X_2, ..., X_p) \tag{5}$$

$$\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_p)^\top \tag{6}$$

Which variables are important?

Some notes:

- The t-statistic associated to the j^{th} predictor is (equivalent to) the square root of the F-statistic for the null hypothesis which sets only $\beta_j = 0$.
- A low p-value for the j^{th} predictor indicates that the predictor is important.
- **Warning**: If there are many predictors, even under the null hypothesis, some of the t-tests will have low p-values. Ways of accounting for this include e.g.
  - controlling the family-wise error rate (FWER)
  - controlling the false discovery rate (FDR)

Also: Know how to compute confidence intervals.

K-nearest neighbors

Mathematically, we can represent KNN (in a classification setting) as

K-nearest neighbors

$$\mathbb{P}(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} \mathbb{I}(y_i = j) \tag{7}$$

We can apply Bayes rule to the resulting probabilities to get our classifier.

Comparing Linear Regression to K-nearest neighbors

Linear regression: prototypical parametric method
KNN regression: prototypical nonparametric method Long story short:

- KNN is only better when the function f_0 is not linear (and plenty of data)
  - **Question**: What if the true function f_0 IS linear?
- When n is not much larger than p, even if f_0 is nonlinear, linear regression can outperform KNN.
- KNN has smaller bias, but this comes at a price of (much) higher variance (c.f. overfitting)

Potential issues in linear regression

- Interactions between predictors
- Non-linear relationships
- Correlation of error terms
- Non-constant variance of error (heteroskedasticity)
- Outliers & studentized residuals
- High leverage points
- Collinearity
- Mis-specification

Naive bayes

$$\mathbb{P}_0(Y|X_1, X_2) \approx \frac{1}{Z}\mathbb{P}_0(Y)\prod_{i=1}^{2}\mathbb{P}_0(X_i|Y)$$

We can estimate $\mathbb{P}_0$ empirically

- e.g. *kernel density estimation*
- Could also use parametric models (e.g. Gaussian distribution)
- Question: What if the feature is categorical?

Remark: we don't need Z if we're just classifying

- Just take the class with the max value, e.g.

Example naive bayes classifier

$$\hat{y}_n = \mathcal{C}(X_1, X_2) = \underset{y\in\{\text{Orange},\text{Blue}\}}{\arg\max}\ \mathbb{P}_0(y)\prod_{i=1}^{2}\mathbb{P}_0(X_i|y) \quad (8)$$

Fitting a logistic regression

Solution:
Let's try to maximize the probability of our training data

$$\mathcal{L}(\boldsymbol{\theta}) = \prod_{i=1}^{n} \mathbb{P}(Y = y_i | \mathbf{X} = \mathbf{x}_i) \tag{9}$$

$$= \prod_{i=1}^{n} p_i^{y_i} \cdot (1 - p_i)^{1 - y_i} \tag{10}$$

where $p_i = g^{-1}(\beta_0 + \beta_1 x_{1,i} + \cdots + \beta_p x_{p,i})$

- We look for θ such that $\mathcal{L}(\boldsymbol{\theta})$ is maximized
- aka Maximum likelihood estimation (MLE)
- Has no closed form solution, so solved with numerical methods (e.g. Newton's method)

Quadratic Discriminant Analysis

Similar to LDA

- Assumes Gaussian $f_k(\mathbf{x})$
- Unlike LDA:
  - Assumes each class has its own covariance matrix ($\mathbf{\Sigma}_k$)

This results in a quadratic discriminant function:

$$\begin{aligned}\delta_k(x) &= -\frac{1}{2}(x-\mu_k)^\top \mathbf{\Sigma}_k^{-1}(x-\mu_k) - \frac{1}{2}\log|\mathbf{\Sigma}_k| + \log\pi_k \\ &= -\frac{1}{2}x^\top \mathbf{\Sigma}_k^{-1}x + x^\top \mathbf{\Sigma}_k^{-1}\mu_k - \frac{1}{2}\mu_k^\top \mathbf{\Sigma}_k^{-1}\mu_k - \frac{1}{2}\log|\mathbf{\Sigma}_k| + \log\pi_k \end{aligned} \tag{11}$$

This results in more parameters to fit:

- LDA: Kp parameters
- QDA: $Kp(p+1)/2$ parameters

Linear Discriminant Analysis vs Logistic regression

Assume a two-class setting with one predictor

Linear Discriminant Analysis:

$$\log\left[\frac{p_1(x)}{1-p_1(x)}\right] = c_0 + c_1 x \tag{12}$$

- c_0 and c_1 computed using $\hat{\mu}_0$, $\hat{\mu}_1$, and $\hat{\sigma}2$

Logistic regression:

$$\log\left[\frac{\mathbb{P}[Y=1|x]}{1-\mathbb{P}[Y=1|x]}\right] = \beta_0 + \beta_1 x \tag{13}$$

- β_0 and β_1 estimated using MLE

How do we estimate the test error?

- Our main technique is to split the data.
- Different approaches:
  1. **Validation set:** Split the data in two parts, train the model on one subset, and compute the test error on the other.
  2. k-**fold:** Split the data into k subsets. Average the test errors computed using each subset as a validation set.
  3. **LOOCV:** k-fold cross validation with $k = n$.
- No approach is superior to all others.
- What are the main differences? How do the bias and variance of the test error estimates compare? Which methods depend on the random seed?

Loss functions

Regression:

- MSE $((y_i - \hat{y}_i)^2)$

Classification:

- Cross-entropy $((y_i \log(\hat{p}_i))$
- 0-1 loss $(\mathbb{I}(y_i \neq \hat{y}_i))$
- Confusion matrix
- Receiver operating characteristic curve (& AUC)

Evaluating a classification method

		Predicted class		
		− or Null	+ or Non-null	Total
True class	− or Null	True Neg. (TN)	False Pos. (FP)	N
	+ or Non-null	False Neg. (FN)	True Pos. (TP)	P
	Total	N*	P*	

We can calculate a number of statistics from this table, e.g.

- True positive rate (aka Sensitivity, aka Recall)

$$\mathbb{P}[\mathit{Predicted}\ +\,|\,\mathit{True}\ +], \text{i.e. } \mathit{TP}/P \tag{14}$$

- True negative rate (aka Specificity)

$$\mathbb{P}[\mathit{Predicted}\ -\,|\,\mathit{True}\ -], \text{i.e. } \mathit{TN}/N \tag{15}$$

- Positive predicted value (aka Precision)

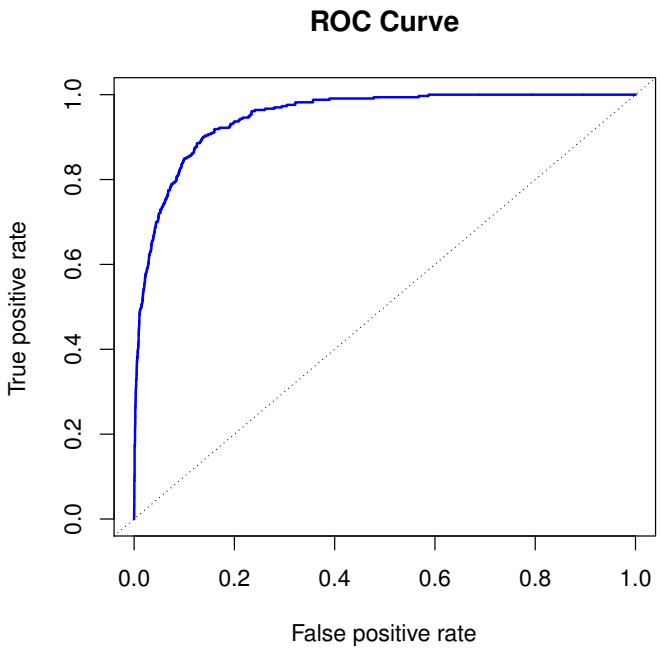
$$\mathbb{P}[\mathit{True}\ +\,|\mathit{Predicted}\ +], \text{i.e. } \mathit{TP}/P^* \tag{16}$$

- Negative predicted value

$$\mathbb{P}[\mathit{True}\ -\,|\mathit{Predicted}\ -], \text{i.e. } \mathit{TN}/N^* \tag{17}$$

Example: Predicting default

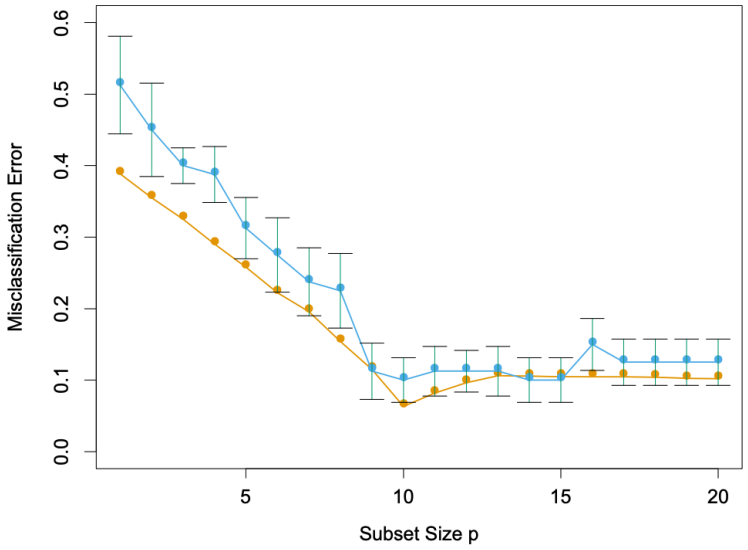
The *Receiver Operating Characteristic (ROC)* curve:



- Displays the performance for every threshold choice.
- The *Area Under the Curve (AUC)* summarizes the classifier performance, e.g.
  - The closer AUC is to 1, the better the performance.
  - $AUC = 0.5$ is equivalent to a random classifier.
    - The minimum value (otherwise, you'd just flip the class predictions).
  - *AUC* is equivalent to the Mann–Whitney U test.

How do we choose a model?

Forward stepwise selection



Blue: 10-fold cross validation
Yellow: True test error

- A number of models with $9 \leq p \leq 15$ have the same CV error.
- The vertical bars represent 1 standard error in the test error from the 10 folds.
- **Rule of thumb:** Choose the simplest model whose CV error is no more than one standard error above the model with the lowest CV error.

The Bootstrap

- **Main idea:** If we have enough data, the empirical distribution is similar to the actual distribution of the data.
- They can be used to:
  1. Approximate the standard error of a parameter (say, β in linear regression), which is just the standard deviation of the estimate when we repeat the procedure with many independent training sets.
  2. Compute CIs (e.g. normal-based, quantile, etc.).
  3. Estimate out of bag error, bias, etc.
  4. **Bagging**: By averaging the *predictions* $\hat{y}$ made with many independent data sets, we eliminate the variance of the predictor.

Each bootstrap iteration will only have about 2/3 of the original data, i.e.

$$\mathbb{P}(x_j \notin P_n^b) = (1 - 1/n)^n \tag{18}$$

n.b. Confirm that your data is iid when applying bootstrap.

The Jackknife

If $d > 1$:

$$\widehat{SE}_B(\hat{\alpha}_n) = \sqrt{\frac{n-d}{d\binom{n}{d}} \sum_z \left(\hat{\alpha}_n^{*,z} - \frac{1}{\binom{n}{d}} \sum_{z'} \hat{\alpha}_n^{*,z'} \right)^2} \tag{19}$$

When $d = 1$, this simplifies to:

$$\widehat{SE}_B(\hat{\alpha}_n) = \sqrt{\frac{n-1}{n} \sum_{i=1}^{n} \left(\hat{\alpha}_n^{*,i} - \frac{1}{n} \sum_{i'=1}^{n} \hat{\alpha}_n^{*,i'} \right)^2} \tag{20}$$

- Is a linear approximation to the bootstrap (though asymptotically equivalent)
- Can be less computationally expensive; esp for large data sets
- Doesn't work well for sample quantiles like the median

Stepwise regression

Two approaches:

1. Use a hold out set (e.g. validation or test set)
    - c.f. Cross-validation
2. Use *modified* metrics that account for the size of k, e.g.
    - Akaike Information Criterion (AIC)
    - Bayesian Information Criterion (BIC)
    - Adjusted R^2

Stepwise regression

Two approaches:

1. Use a hold out set (e.g. validation or test set)
    - c.f. Cross-validation
2. Use *modified* metrics that account for the size of k, e.g.
    - Akaike Information Criterion (AIC)
    - Bayesian Information Criterion (BIC)
    - Adjusted R^2

How the modified metrics compare to using hold out sets

- Can be (much) less expensive to compute
- Motivated by asymptotic arguments and rely on model assumptions (e.g. normality of the errors)
- Equivalent concepts for other models (e.g. logistic regression)

Issues with stepwise methods

Important things to keep in mind:

- The selected model is not guaranteed to be optimal
  - There are often several equally good models
- The procedure does not take into account a researcher's knowledge about the predictors
- Outliers can have a large impact on the procedure
- Some predictors should be considered together as a group (e.g. dummy indicators for seasons of the year)
- The coefficients, R^2, p-values, CI's, etc are all biased/invalid
- Should not over-interpret the order that the predictors are included
- Cannot conclude that all variables included are important, or all excluded variables are unimportant

Shrinkage methods

Allows us to use all p predictors, but will regularize (i.e. shrink) their coefficients in some way.

- Common to shrink them towards 0

Shrinkage methods

Allows us to use all p predictors, but will regularize (i.e. shrink) their coefficients in some way.

- Common to shrink them towards 0

Question: Why would shrunk coefficients be better?

- Will introduce bias, but can significantly reduce the variance
  - If the variance is noticeably larger, this decreases the test error
- There are Bayesian motivations to do this: the prior tends to shrink the parameters.

Shrinkage methods

Allows us to use all p predictors, but will regularize (i.e. shrink) their coefficients in some way.

- Common to shrink them towards 0

Question: Why would shrunk coefficients be better?

- Will introduce bias, but can significantly reduce the variance
  - If the variance is noticeably larger, this decreases the test error
- There are Bayesian motivations to do this: the prior tends to shrink the parameters.

Three common shrinkage methods:

1. Ridge regression
2. Lasso regression
3. Elastic net

Shrinkage summaries

$$\min_{\beta} \quad \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{i,j} \right)^2 + \lambda_2 \sum_{j=1}^{p} \beta_j^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| \quad (21)$$

Method	**Shrinkage parameters**
OLS	$\lambda_1 = \lambda_2 = 0$
Ridge	$\lambda_1 = 0, \lambda_2 > 0$
LASSO	$\lambda_1 > 0, \lambda_2 = 0$
Elastic net	$\lambda_1 > 0, \lambda_2 > 0$
$\hat{\beta}_n = 0$	$\lambda_1 = \infty$ or $\lambda_2 = \infty$

Self testing questions

For each of the regression and classification methods:

1. What are we trying to optimize?
2. What does the fitting algorithm consist of, roughly?
3. What are the tuning parameters, if any?
4. How is the method related to other methods, mathematically and in terms of bias, variance?
5. How does rescaling or transforming the variables affect the method?
6. In what situations does this method work well? What are its limitations?