ST3248 Finals Cheatsheet

by Devansh Shah

Regression Problems

Key problem: estimate

$$E(Y|X=x) = E(f(x)+\varepsilon|X=x) = f(x)$$

Mean Squared Error
= reducible + irreducible error
= $bias^2 + var$ + irreducible error

$$E[(y_0 - \hat{f}(x_0))^2] = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\varepsilon)$$

where $Y = f(x) + \varepsilon$

Assumptions of linear regression:

- $\varepsilon \perp X$ (constant variance aka homoscedasticity)
- $E[\varepsilon] = 0$

For simple linear regression,

$$\hat{\beta_1} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\hat{\beta_0} = \bar{y} - \hat{\beta_1}\bar{x}$$

Mean Squared Error (aka Loss Function)

$$L(e) = \frac{1}{n}\sum_{i=1}^n (y_i - \hat{f}(x_i))^2$$

Standard Errors

Here, $\sigma^2 = Var(\varepsilon)$ and we assume that the errors ε_i are uncorrelated with common variance. In practice, we can estimate σ by the residual standard error (RSE).

$$SE(\hat{\beta}_0)^2 = \sigma^2\left[\frac{1}{n} + \frac{\bar{x}^2}{\sum_{i=1}^n (x_i - \bar{x})^2}\right]$$

$$SE(\hat{\beta}_1)^2 = \frac{\sigma^2}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

t-statistic for hypothesis testing: $\frac{\hat{\beta}_1}{\hat{SE}(\hat{\beta}_1}$). Under H_0, t has t-distribution with $n-2$ degrees of freedom.

For a linear regression model with p predictors, the F-statistic:

$$F = \frac{(TSS - RSS)/p}{RSS/(n-p-1)} \sim \chi^2_{p,n-p-1}$$

Residual standard error, R^2, Adjusted R^2

$$RSE = \sqrt{\frac{1}{n-p-1}RSS}$$

$$RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

$$TSS = \sum_{i=1}^n (y_i - \bar{y})^2$$

$$AdjR^2 = 1 - \frac{RSS/(n-p-1)}{TSS/(n-1)}$$

- R^2 can be interpreted as the "proportion of variance in y that can be explained by the model".
- For simple linear regression, $R^2 = r^2$ where r is the sample correlation, given by:

$$r = \hat{Cor}(X,Y) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2}\sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

- Higher adjusted $R^2 \implies$ better model.

Confidence and Prediction Intervals

Confidence Interval - Used when we're estimating how close $\hat{Y}$ will be to $f(X)$. Since we're estimating the average value of the response, we can ignore the random error ε since it's average is zero.

Prediction Interval - Used when we're making a prediction for an individual Y given X. This needs to account for the random error ε

So, prediction interval is always wider than a confidence interval (for the same degree of confidence).

Remarks on Regression

- We interpret β_j as the average effect on Y of one unit increase in X_j, *holding all other predictors fixed.*
- p-value for each individual predictor provides information about whether an individual predictor is related to the response, *after adjusting for the other predictors.*
- Even if one of the p-values is small, it does NOT mean that the overall F-statistic must have a small p-value.

Classification

Key problem: estimate

$$p_j(x) = P(Y = j|X = x_0)$$

Irreducible error

Bayes error rate

$$1 - E(\max_j P(Y = j|X))$$

lowest possible test error rate (irreducible error)

Bayes classifier

choose $\hat{y}_0 = j$ from data s.t.

$$\max_j P(Y = j|X = x_0)$$

Bayes decision boundary: $P(Y = 1|X) = P(Y = 2|X) = 0.5$
Bayes classifier achieves Bayes error rate. However, we need to estimate $P(Y = j|X = x_0)$ before using Bayes classifier (which is not possible in practice).

Error rate

$$L(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^n I(y_i \neq \hat{y}_i)$$

Logistic Regression

The logistic function ensures that the output is in the range $[0, 1]$ for *all* values of X:

$$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

For the "simple" logistic regression (only 1 predictor), the decision boundary is linear. But we can add more predictors to increase the flexibility of the decision boundary.

Odds

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0 + \beta_1 X}$$

log-odds/logit

$$log(\frac{p(X)}{1 - p(X)}) = \beta_0 + \beta_1 X$$

Likelihood function

$$L(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'}))$$

Confusion matrix

True class (horizontal)

Predicted class		Positive	Negative
	Positive	TP	FP
	Negative	FN	TN

FP rate := % negative examples that are classified as positive
FN rate := % positive examples that are classified as negative
Sensitivity := True positive rate $(TP/(TP + FP))$
Specificity := True negative rate $(TN/(TN + FP))$

Linear Discriminant Analysis

- LDA is more stable than logistic regression (especially when the classes are well-separated)
- LDA is more popular when we have more than 2 response classes.

We define $\pi_k = P(Y = k)$ and $f_k(X) = Pr(X = x|Y = k)$, then

$$p_k(x) = P(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{i=1}^K \pi_i f_i(x)}$$

If we assume the conditional distributions $f_k(x)$ are normal with their own means μ_k but common variance σ^2, we end up with a linear boundary.
Then, the formula for LDA is equivalent to picking k with the largest value of:

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

Since $\delta_k(x$ is linear in x, the decision boundary is also linear.

In practice, we estimate the quantities using information from our sample:

$$\hat{\pi}_k = \frac{n_k}{n}$$

$$\hat{\mu}_k = \frac{1}{n_k}\sum_{i:y_i=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-K}\sum_{k=1}^K \sum_{i:y_i=k} (x_i - \hat{\mu}_k)^2$$

In case of multiple predictors, we assume that the $X = (X_1, X_2, \ldots, X_p)$ has a (conditional) multi-variate normal distribution with class-specific mean vector and common covariance matrix.

Quadratic Discriminant Analysis

QDA relaxes the common variance assumption: so the distribution of $X$for each class of Y is normally distributed and has its own mean *and covariance matrix.*
The discriminant function $\delta_k(x)$ has a term quadratic in x, hence the name. This results in a quadratic decision boundary.
QDA is more flexible than LDA. So, QDA has lower bias, higher variance (because more parameters to estimate).https://www.overleaf.com/project/655c632aa9a81470b6fd3b6f

AUC, ROC

ROC: receiver operating characteristic curve
AUC: area under the curve

- Compares FP rate (x-axis) and TP rate (y-axis)
- Higher AUC, larger area = better
- Useful to compare different probability threshold
- When threshold = 1, FP=0, TP=0 (lower left)
- When threshold = 0, FP=1, TP=1 (upper right)

Note: $\hat{P}(Y|X) \geq$ threshold will be classified as positive

Cross-validation (CV)

Key problem: estimate test MSE using given data set. Here, the "bias" and "variance" is in estimating the test MSE, *not the model parameters.*

K-fold Cross-validation

1. Randomly split n samples into K blocks, each block has $n_K = n/K$ obs
2. For model $j = 1, \cdots, M$:
   [2.1] For block $k = 1, \cdots, K$
   [2.1.1] Fit model j on all blocks except block k
   [2.1.2] Evaluate model j based on block k
   [2.2.2] Calculate $CV_{(K)}^{(j)} = \sum_{k=1}^K (n_k/n)MSE_k$
   Note $(n_k/n) \approx 1/K$

Estimated test error

$$CV_{(K)} = \frac{1}{K}\sum_{i=1}^K MSE_i$$

Variance

$$\hat{Var}(CV_{(K)}) = \sum_{k=1}^K \frac{(MSE_k - \bar{MSE}_k)^2}{K-1}$$

$$= \frac{1}{K}\hat{Var}(CV_k) + 2\sum_{i<j}^K Cov(CV_i, CV_j)$$

Limitation

- Bias still exist (min bias when $K = n$)
- High variance due to overlapping blocks used for training model in each iteration
- CV should be performed before feature selection step

Leave-one out CV (LOOCV)

Using Cross-validation with $K = n$
Special result for least-squares linear, polynomial regression

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^n \left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

$h_i :=$ leverage, $n :=$ number of samples (i.e. LOOCV)
Limitation

- High variance since estimates from each fold are highly correlated (nearly identical training data)
- Computational intensive (in general case)

Bootstrap

Key idea: resampling with replacement
Suppose we're estimating the standard error and bias of the statistic $\hat{\alpha}$ (obtained using original data),

1. For bootstrap sample $r = 1, \cdots, B$:
   Randomly select n observations with replacement from the training set.
   Compute $\hat{\alpha}^{*r}$ for this sample.

Then, define $\hat{\alpha}' = \frac{1}{B}\sum_{r=1}^{B}\hat{\alpha}^{*r}$ (mean of all statistics across all bootstrap samples)

$$\hat{SE}(\alpha) = \sqrt{\frac{1}{B-1}\sum_{r=1}^{B}(\hat{\alpha}^{*r} - \hat{\alpha}')^2}$$

$$\hat{Bias}(\hat{\alpha}) = \frac{1}{B}\sum_{r=1}^{B}\hat{\alpha}^{*r} - \hat{\alpha}$$

Linear Model Selection and Regularization

When $p \geq n$, there is no unique least squares estimate.
3 Categories:

1. **Subset Selection**: Identify a subset of the p predictors and then use least squares on the reduced set of variables.
   (a) Best Subset Selection: brute force all 2^p models.
   (b) Forward Stepwise Selection: start from ϕ model and greedily select predictor which results in smallest RSS in the resulting model. Useful when $p \geq n$ because we can stop at the model with $k = n - 1$ predictors. Explores a total of $1 + p(p-+1)/2$ models.
   (c) Backward Stepwise Selection: same as forward but in reverse - remove predictor which results in lowest increase in RSS.
   (d) Mixed Selection: after each forward step, remove variables that are no longer useful.
2. **Shrinkage**: We use all p predictors but the estimated coefficients are shrunken towards zero.
3. **Dimension Reduction**: We project p predictors into a M-dimensional subspace, where $M < p$, and then use these projections as the predictors to fit with least squares.

Approaches to select the best model with respect to test error:

1. Indirectly estimate test error by making an analytical adjustment to the training error, to account for overfitting. All of them add a penalty term for the predictors.
   (a) **Mallows' C_p**: for a model with d predictors,
   $$C_p = \frac{1}{n}(RSS + 2d\hat{\sigma}^2)$$
   where $\hat{\sigma}^2$ is the estimated variance of error ε.
   If $\hat{\sigma}^2$ is unbiased for σ^2, then C_p is unbiased for the test error.
   (b) **AIC (Akaike Information Criterion)**: proportional to C_p
   $$AIC = \frac{1}{n\hat{\sigma}^2}(RSS + 2d\hat{\sigma}^2)$$
   (c) **BIC (Bayesian Information Criterion)**: heavier penalty, lower number of predictors
   $$BIC = \frac{1}{n\hat{\sigma}^2}(RSS + \log(n)d\hat{\sigma}^2)$$
   (d) **Adjusted R^2**:
   $$\text{Adj. } R^2 = 1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$$
   where RSS $= \sum_{i=1}^{n}(y_i - \hat{y})^2$ and TSS $= \sum_{i=1}^{n}(y_i - \bar{y})^2$
2. Directly estimate the test error, using validation set or cross-validation approach:
   - makes fewer assumptions about the true underlying model (as it is an empirical method).
   - useful when it is hard to estimate the error variance σ^2
   - useful when the asymptotic condition $n \gg p$ is not met

Shrinkage Methods

Introduces some bias by imposing penalty for larger coefficients, but can significantly reduce their variance. Works well when p is almost as large as n
Ridge Regression (aka L2 normalization) aims to minimize:

$$RSS + \lambda\sum_{j=1}^{p}\beta_j^2$$

where $\lambda \geq 0$ is a tuning parameter.

- We do not penalize the intercept term β_0 because it is simply a measure of the average sample response.
- Ridge regression is not scale equivariant - the model *is* affected by the scale of the predictors. Hence, we need to standardise (to ensure every predictor is penalized fairly).
- It cannot perform variable selection - even for large λ, the coefficient estimates may be small, but non-zero.
- Dual optimisation problem:
$$\arg\min_{\beta}(RSS) \text{ subject to } \sum_{j=1}^{p}(\beta_j)^2 \leq s$$
- Use `alpha=0` in `glmnet` function to indicate "ridge"

Lasso Regression (aka L1 normalization) aims to minimize:

$$RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

- The l_1 penalty term has the effect of forcing some of the estimates to be exactly zero, for large enough λ - thereby performing variable selection.
- Use `alpha=1` in `glmnet` function to indicate "lasso"

`glmnet` is a numerical fitting method - lower the error threshold using the argument `thresh`.

Dimension Reduction Methods

We generate $Z_1, Z_2, \cdots, Z_M$ linear combinations of our original p predictors, i.e., $Z_m = \sum_{j=1}^{p}\phi_{jm}X_j$ and then fit the linear regression model $y_i = \theta_0 + \sum_{m=1}^{M}\theta_m z_{im} + \varepsilon_i$ using least squares.

- The transformed model is a special case of the original linear model where each estimated β_j is constrained to take the form of a linear combination of ϕ_{jm}'s. This constraint can incur bias but it also reduces variance.

Principal Component Regression uses PCA as a pre-processing step.

- First PC direction is that along which the observations vary the most (does not consider the response).
- We require $\sum_j^p \phi_{jm}^2 = 1$ so that PC is a "unit" vector and cannot blow up the variance arbitrarily.
- Every PC must be uncorrelated with all preceding PCs, and aim to maximize variance along it. So, successive PCs will contain lesser variability by design.
- PCR makes the assumption that the PC directions are also the directions that are associated with the response.
- PCR is not scale equivariant since it favors predictors on a larger scale (since it is looking for highest variability). So, we should standardize the predictors.
- PCR and ridge regression are closely related - ridge regression can be seen as a smoothed version of PCR.
- `scale=TRUE` in `pcr` to scale predictors. CV score in output is for $\sqrt{MSE}$.

Partial Least Squares is a supervised method (uses response) that assigns weights to the predictors based on their marginal correlation with the response. To compute the first PLS direction:

1. Standardize the p predictors.
2. Set ϕ_{j1} equal to the coefficient from the simple linear regression of Y onto X_j (this is proportional to $r_{X_j,Y}$
3. $Z_1 = \sum_{j=1}^{p}\phi_{j1}X_j$ places the highest weight on variables strongly correlated with Y.

This picks the direction that contain most of the variation in the response and predictors.
To find the second PLS direction, we first *adjust* each of the predictors for Z_1, by regressing each predictor on Z_1 and taking the residuals to be the new predictor. Z_2 will also be a linear combination of the original predictors.

High-Dimensional Settings

When $p \geq n$,

- We can no longer use least squares.
- C_p, AIC and BIC are not appropriate.
- Estimating $\hat{\sigma}^2$ is not possible.
- Adj. R^2 will just yield 1 when there is a perfect fit.
- The multicollinearity problem is extreme: any variable can be written as a linear combination of the other variables. So, we can never know exactly which variables (if any) are truly predictive of the outcome.

Hence, we should use less flexible methods such as forward stepwise selection, lasso, PCR. We should always use results from an independent test set, or CV errors, *not training data*, as evidence of good model fit.

Unsupervised Learning

Principal Component Analysis

- Maximum number of principal components is $\min(n-1, p)$
- The loadings of the first PC give us the direction in the feature space along which the data vary the most, and the scores are the projections along these directions.
- Alternatively, PCs provide low-dimensional linear surfaces that are closest (in average squared euclidean distance) to the observations.
- When M is sufficiently large, $x_{ij} \approx \sum_{m=1}^{M} z_{im}\phi_{jm}$.
- Each PC loading vector is unique, up to a sign flip.
- It is necessary to standardize the variables before using PCA.
- Proportion of Variance Explained (PVE) by the mth PC:
$$PVE = \frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{jm}x_{ij}\right)^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2}$$
The plot of PVE vs. PC is called a *scree plot*.
- `pr.out$rotation` is a $p \times M$ rotation matrix where every column i is the loading vector for the ith PC. When $\mathbf{X}$ $(n \times p)$ is multiplied by it, we get the principal component scores for every observation.

Clustering

Goal is to find the cluster assignments that minimizes:

$$\sum_{k=1}^{K}\frac{1}{|C_k|}\sum_{i,i' \in C_k}\sum_{j=1}^{p}(x_{ij} - x_{i'j})^2$$

but the above is NP-Hard.

- We approximate a "good enough" solution using K-Means (local search) and Hierarchical (greedy) clustering.
- Again, there's a decision on whether or not to standardize the variables, depending on the context.
- Clustering results are not robust, i.e., they are sensitive to outliers (or even small perturbations). Use "soft" version of K-means or use probabilistic assignments as opposed to deterministic ones.

K-Means Clustering Based on this identity:

$$\frac{1}{|C_k|}\sum_{i,i' \in C_k}\sum_{j=1}^{p}(x_{ij} - x_{i'j})^2 = 2\sum_{i \in C_k}\sum_{j=1}^{p}(x_{ij} - \bar{x}_{kj})^2$$

where $\bar{x}_{kj} = \frac{1}{|C_k|}\sum_{i \in C_k} x_{ij}$ is the mean of feature j in cluster k.

- `tot.withinss`: $\sum_{k=1}^{K}\sum_{x_i \in C_k}(x_i - \mu_k)^2$
- `betweenss`: $\sum_{k=1}^{K}|C_k|(\mu_k - \bar{x})^2$
- `betweenss + tot.withinss = totss` $= \sum_{i=1}^{n}(x_i - \bar{x})^2$

Hierarchical Clustering: can be agglomerative or divisive. We consider agglomerative only.

- Dendogram: Tree-based representation of observations and their clustering (for any number of clusters, all at once).
- Generally dissimilarity is defined as euclidean distance.
- Clusters generated from a dendogram may not be optimal, and may not be the same as those resulted from K-means.
- Height of fusion in the dendogram tells us the dissimilarity between the 2 clusters at the point of fusion.
- Horizontal distance between observations does not matter since we can swap branches without affecting the meaning. Only the vertical distance, i.e., the height of fusions, matter.
- If observation x_i and x_j are not in the same cluster for $K = k$, they will not be in the same cluster for any $K = m > k$ - hierarchical relation between the clusters as K is varied.
- Specify `method` argument as "complete", "average", "single" to `hclust()` function
- `dist(x)` computes the inter-observation euclidean distance matrix.
- `cuttree` cuts the dendogram to generate a specified (2nd argument) number of clusters.

Measures of dissimilarity between clusters (aka linkage):

- Complete: Maximum intercluster dissimilarity, $\max(d(x,y) : x \in C_1, y \in C_2)$
- Single: Minimum intercluster dissimilarity, $\min(d(x,y) : x \in C_1, y \in C_2)$
- Average: Mean intercluster dissimilarity, $\frac{1}{|C_1| \times |C_2|}\sum_{x \in C_1}\sum_{y \in C_2} d(x,y)$
- Centroid: Dissimilarity between the centroids of the 2 clusters, $d(\mu_1, \mu_2)$

Note:

- Dendogram depends strongly on the linkage used.
- Centroid linkage can result in undesirable inversions (2 clusters are fused at a height below either of them).
- Single linkage can result in extended, trailing clusters - unbalanced dendogram.
- Obviously, the height of fusion of clusters will follow the order: complete $\geq$ average $\geq$ single

Alternative choice of dissimilarity measure: correlation-based distance (higher correlation, more similar).