STK-IN4300 Compendium

Gabriel Sigurd Cabrera

(Dated: Friday 29th November, 2019)

I. OVERVIEW OF TOPICS

A. Lecture 1

1. Basics

Typical Scenario:

An *outcome* Y (*dependent variable*, *response*) can be *categorical* or *qualitative*.

We want to predict this outcome based on a set of *features* $X_1, X_2, ..., X_p$ (*independent variables*, *predictors*).

In practice, we have a *training set* that is used to create a *learner* (or model/rule $f(X_i) \approx Y_i$.)

A *supervised learning problem* is when the outcome is measured in the training data, and can be used to construct a learner Y.

B. Least Squares Estimate

Given a training set $\{(x_{i1}, x_{i2}, ..., x_{ip}, y_i\}$ a *least-squares* model is given by:

$$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \epsilon_i$$

With a least-square estimate:

$$\hat{\beta} = (X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}y$$

Where $X^{\mathrm{T}}X$ is known as the *Gramian*.

C. Invertability

If $X^{\mathrm{T}}X$ is *not invertible*, then we can use *dimension reduction* or *shrinkage methods*.

Some dimension reduction methods are to:

- Remove variables with *low correlation* (forward selection/back substitution)
- More formal subset selection
- Selecting optimal linear combinations of variables (*principal component analysis*.)

Some shrinkage methods are:

- *Ridge regression*
- *LASSO*
- *Elastic net*

D. Conventions

Quantitative response: *Regression*

Qualitative response: *Classification*

E. Least-Squares

For *ordinary least-squares* (OLS), we estimate β by minimizing the *residual sum of squares* (RSS):

$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N}(y_i - x_i^{\mathrm{T}}\beta)^2 = (y - X\beta)^{\mathrm{T}}(y - X\beta)$$

Where $X \in \mathbb{R}^{N\times p}$, $X \in \mathbb{R}^N$.

F. K-Nearest-Neighbors

The *k-nearest-neighbors* (KNN) of x is the mean:

$$\hat{Y}(x) = \frac{1}{k}\sum_{i:x_i\in N_k(x)} y_i$$

G. Other Methods

OLS and KNN are the basis of most modern techniques; some of these are:

- *Kernel methods* that weigh data according to distance
- In higher dimensions, weighing variables based on correlation
- Local regression models
- Linear models of functions of X
- *Projection pursuit* and *neural network*

H. Statistical Decision Theory

Statistical decision theory gives a *mathematical framework* for finding the optimal learner.

Given $X \in \mathbb{R}^p$, $Y \in \mathbb{R}$ and a *joint distribution* $p(X, Y)$, our goal is to find a function $f(X)$ for predicting Y given X.

This requires a *loss function* $L(Y, f(X))$ for penalizing errors in $f(X)$ when the truth is Y.

An example is *squared error loss*:

$$L(Y, f(X)) = (Y - f(X))^2$$

The *expected prediction error* of $f(X)$ is given by:

$$\text{EPE}(f) = E_{X,Y}[L(Y, f(X)] = \int_{x,y} L(y, f(x))p(x, y)\ \mathrm{d}x\ \mathrm{d}y$$

Next, we must find the f that minimizes EPE(f).

For the *squared error loss* $L(Y, f(X) = (Y - f(X))^2$, we have:

$$\text{EPE}(f) = E_{X,Y}[(Y - f(X))^2] = E_X E_{Y|X}[(Y - f(X))^2|X]$$

It is sufficient to minimize $E_{Y|X}[(Y - f(X))^2|X]$:

$$f(x) = \text{argmin}_c E_{Y|X}[(Y - c)^2|X = x] = E[Y|X = x]$$

This is known as the *conditional expectation*, or the *regression function*. This implies that the best prediction of Y at any point $X = x$ is the *conditional mean*.

I. Error Decomposition

$$E[(Y-\hat{f}(X))^2] = \underbrace{\sigma^2}_{\text{irreducible error}} + \underbrace{\underbrace{\text{Var}(\hat{f}(X))}_{\text{variance}} + \underbrace{E[\hat{f}(X) - f(X)]^2}_{\text{bias}^2}}_{MSE}$$

J. Assumptions for OLS

- A function is linear in its arguments; $f(x) \approx x^{\mathrm{T}}\beta$.
- $\text{argmin}_\beta E[(Y - X^{\mathrm{T}}\beta)^2|X = x] \rightarrow \beta = E[XX^{\mathrm{T}}]^{-1}E[XY]$.
- Replacing the expectations by averages over the training data leads to $\hat{\beta}$.

K. Assumptions for KNN

- Uses $f(x) = E[Y|X = x]$ directly.
- $\hat{f}(x_i) = \text{mean}(y_i)$ for observed x_i.
- Normally, there is at most one observation for each point x_i.
- Uses points in the neighborhood:

$$\hat{f}(x) = \text{mean}(y_i|x_i \in N_k(x))$$

- There are two approximations:

  - *Expectation* is approximated by averaging over sample data.

  - *Conditioning* on a point is related to conditioning on a neighborhood.

- $f(x)$ can be approximated by a *locally constant function.*
- For $N \rightarrow \infty$, all $x_i \in N_k(x) \approx x$.
- For $k \rightarrow \infty$, $\hat{f}(x)$ is getting more stable.
- Under mild regularity conditions on $p(X, Y)$:

  $\hat{f}(x) \rightarrow E[Y|X = x]$ for $N, k \rightarrow \infty$ s.t. $k/N \rightarrow 0$

- It is unnecessary to implement the *squared loss error* function (L_2 loss function.)
- A valid alternative is the L_1 loss function, whose solution is the conditional median:

$$\hat{f}(x) = \text{median}(Y|X = x)$$

- More robust estimates than those obtained with conditional mean.
- The L_1 loss function has discontinuities in its derivatives which leads to numerical difficulties.

L. Conclusion

OLS: stable but biased

KNN: less biased and less stable

For higher dimensions, KNN suffers from the *curse of dimensionality*

II. LECTURE 2

A. Gauss-Markov Theorem

The least square estimator $\hat{\theta} = a^{\mathrm{T}}(X^{\mathrm{T}}X)^{-1}X^{\mathrm{T}}y$ is the:

B	est	smallest error (MSE)
L	inear	$\hat{\theta} = a^{\mathrm{T}}\beta$
U	nbiased	$E[\hat{\theta}] = \theta$
E	stimator	

Given the *error decomposition*, then any estimator $\tilde{\theta} = c^{\mathrm{T}}Y$ s.t. $E[c^{\mathrm{T}}Y] = a^{\mathrm{T}}\hat{\beta}$ has $\mathrm{Var}(c^{\mathrm{T}}Y) \geq \mathrm{Var}(a^{\mathrm{T}}\hat{\beta})$.

B. Hypothesis Testing/F-Score

To test $H_0 : \beta_j = 0$ we use the *Z-score statistic*:

$$z_j = \frac{\hat{\beta}_j - 0}{sd(\beta_j)} = \frac{\hat{\beta}_j}{\hat{\sigma}\sqrt{(X^{\mathrm{T}}X)^{-1}_{[j,j]}}}$$

When σ^2 is unknown, under H_0:

$$z_j \sim t_{\mathrm{N}-p-1}$$

where t_k is a Student t distribution with k degrees of freedom

When σ^2 is known, under H_0:

$$z_j \sim N(0;1)$$

To test $H_0 : \beta_j, \beta_k = 0$:

$$F = \frac{(RSS_0 - RSS_1)/(p1 - p0)}{RSS_1/(N - p - 1)}$$

Where 1 refers to a larger model, and 0 to a smaller one.

C. Variable Selection

Sparser models (with fewer variables) have a smaller variance, are easier to intuitively grasp, and are more portable, or easier to use in practice.

Some approaches are:

- *Forward Selection*
- *Backward Elimination*
- *Stepwise* and *Stepback* elimination
- *Best subset*
- *Stagewise selection*

1. Forward Selection

Start with null model $Y = \beta_0 + \epsilon$

Among a set of possible variables, add that which reduces the unexplained variability the most.

Repeat until a stopping criterion (like a particular *p-value*) is met.

2. Backward Elimination

Start with full model $Y = \beta_0 + \beta_1 X + \cdots \beta_p X_p + \epsilon$

Remove the variable that contributes the least in explaining the outcome variability

Repeat until stopping criterion is reached.

3. Stepwise/Stepback Selection

Mixture of forward selection/backward elimination

Allows both adding and removing variables at each step.

Starting from the null model is *stepwise selection*, which starting from the full model is *stepback selection*.

4. Best Subset

Compute all the 2^p possible models (each variable in/out)

Choose the model which minimizes a loss function (e.g. AIC)

5. Stagewise Selection

Similar to forward selection

At each step, the specific regression coefficient is updates only using the information related to the corresponding variable.

Good for higher dimensions, bad for lower ones!

D. Model Assessment and Selection

Model Assessment:

Evaluate the performance (in terms of a prediction) of a selected model

Model Selection:

Select the best model for the task

Generalization:

A prediction model must be valid in broad generality, not specific for a specific dataset

Define Y as target variable, X as input matrix, and $\hat{f}(X)$ as prediction rule, trained on a training set $\mathcal{T}$.

The error is measured through a *loss function* $L(Y, \hat{f}(X))$ which penalizes the differences between Y and $\hat{f}(X)$.

1. Continuous Outcomes

Typical choices are the *quadratic loss*:

$$L(Y, \hat{f}(X)) = (Y - \hat{f}(X))^2$$

And the *absolute loss*

$$L(Y, \hat{f}(X)) = |Y - \hat{f}(X)|$$

2. Categorical Variables

Let G be the target variable, which takes K values in $\mathcal{G}$

Typical choices are the *0-1 loss*:

$$L(Y, \hat{f}(X)) = \mathbb{1}(G \neq \hat{G}(X))$$

And the *deviance*:

$$L(Y, \hat{f}(X)) = -2\log(\hat{p}_{\mathrm{G}})(X)$$

E. Test Error

The *test error* or *generalization error* is the prediction error over an independent test sample

$$\mathrm{Err}_{\mathcal{T}} = E[L(Y, \hat{f}(X))|\mathcal{T}]$$

Where both X and Y are drawn randomly from their joint distribution.

The *specific training set* $\mathcal{T}$ used to derive the prediction rule is fixes – the test error refers to the error for this specific $\mathcal{T}$.

Generally, we want to minimize the *expected prediction error*

$$\mathrm{Err} = E[L(Y, \hat{f}(X))] = E[\mathrm{Err}_{\mathcal{T}}]$$

We would like to calculate Err, but we only have information on the single training set, so our goal is to estimate $\mathrm{Err}_{\mathcal{T}}$

The training error

$$\bar{\mathrm{Err}} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(X))$$

is not a good estimator of $\mathrm{Err}_{\mathcal{T}}$.

We do not want to minimize the training error becaue of overfitting issues (i.e. the model is very specifically constructed for the training data)

F. Data Splitting

Ideally, the best option is to split the data randomly into three independent sets:

Set	Purpose	Suggestion
Training	Fit the model(s)	50%
Validation	Identify best model	25 %
Test	Assess best model performance	25 %

G. Bias-Variance

1. KNN

For *k-nearest-neighbors*, the number of neighbors is inversely related to the complexity

A smaller k implies a smaller bias, larger variance, and vice-versa

H. OLS

The complexity is directly related to p

I. Optimism of Training Error Rate

For many loss functions, the optimism is:

$$\omega = \frac{2}{N}\sum_{i=1}^{N} \mathrm{Cov}(\hat{y}_i, y_i)$$

Optimism depends on how much y_i affects its own prediction

The harder we fit the data, the larger the optimism

J. Model Selection

What to choose between AIC:

$$-2E[\log(p_{\hat{\theta}})(Y)]$$

and BIC:

$$BIC = -2\ell(\hat{\theta}) + d\log(N)$$

There is no clear winner, but BIC leads to a sparser model, and AIC tends to be best for prediction. BIC is consistent, but for finite sample sizes, BIC tends to select models that are too sparse.

III. LECTURE 3

A. Cross-Validation

,

Aims to estimate the *estimated test error* $\text{Err} = E[L(Y, \hat{f}(X))]$.

With enough data, can be split into training/test sets

Since this is often impossible, the data is split into k-folds $\mathcal{F}_1, ..., \mathcal{F}_k$ of approximately equal size.

Then, use $k-1$ folds to train the model sequentially with the last fold used to test the data.

No clear solution on choosing k, but:

- Smaller k reduces variance, increases bias
- Larger k increases variance, reduces bias
- When $k = N$, we call this *leave-one-out* cross validation (LOOCV).

LOOCV estimates the expected test error approximately unbiased

LOOCV has a large variance

Often, $k = 5$ or $k = 10$ is selected.

The *generalized cross-validation* or GCV:

$$GCV(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{y_i - \hat{f}(x_i)}{1 - \text{trace}(S)/N}\right]^2$$

is a convenient approximation of LOOCV for linear fitting under square loss, and is computationally advantageous.

B. Bootstrap Methods

Like k-fold, but uses sampling with replacement from the original dataset; mimics new experiments.

C. Principal Component Analysis

Given an input matrix X, the SVD is given by:

$$X = UDV^{\text{T}}$$

The *eigendecomposition* of $X^{\text{T}}X$

$$X^{\text{T}}X = VD^2V^{\text{T}}$$

The eigenvectors v_j (columns of V) are used to define the principal components of X, $z_j = Xv_j$.

The first principal component z_1 has the largest sample variance.

Notes:

- PCR can be used in high dimensions for $M < n$
- If $M = N$, $\hat{\beta}_{\text{PCR}}(M) = \hat{\beta}_{\text{OLS}}$
- M is a *tuning parameter* and can be found through cross-validation
- *Shrinkage effect*
- Principal components are scale dependent, so remember to standardize X

D. Partial Least Squares

Similar to PCR:

- Construct a set of linear combinations of X
- PCR only uses X, ignoring y
- In PLS, we also consider y
- As for PCR, X must be standardized

E. Ridge Regression

When two predictors are strongly correlated – collinear

In the case of linear dependency – super-collinearity

In the case of super-collinearity, $X^{\text{T}}X$ is not invertible

So use $X^{\text{T}}X + \lambda I_p$ for $\lambda > 0$, which is invertible.

Ridge Estimator:

$$\hat{\beta}_{\text{ridge}}(\lambda) = (X^{\text{T}}X + \lambda I_p)^{-1}X^{\text{T}}y$$

This is minizing:

$$\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2$$

Subject to $\sum_{j=1}^{p}\beta_j^2 \leq t$, implying:

$$\hat{\beta}_{\text{ridge}}(\lambda) = \text{argmin}_\beta \left\{ \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p}\beta_j^2 \right\}$$

Notes:

- Ridge solution is not equivariant under scaling, so X must be standardized before applying the minimizer
- The intercept is not involved in the penalization

IV. LECTURE 4

A. LASSO

LASSO, or *least absolute shrinkage and selection operator* is similar to ridge regression, but uses an L_1 penalty instead of L_2:

$$\hat{\beta}_{\text{LASSO}}(\lambda) = \text{argmin}_\beta \left\{ \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p}|\beta_j| \right\}$$

X must be standardized, and β_0 is not considered in the penalty term.

Notes:

- Some estimates are forced to be zero (variable selection) due to the L_1 norm
- There exists no closed form for the estimator due to the L_1 norm
- $\lambda \to 0 \implies \hat{\beta}(\lambda) \to \hat{\beta}_{\text{OLS}}$
- $\lambda \to \infty \implies \hat{\beta}(\lambda) \to 0$

B. Generalized Linear Models

LASSO and ridge can be used with any linear regression model, e.g. logistic regression; the generalized model can be written as:

$$\tilde{\beta}(\lambda) = \text{argmin}_\beta \left\{ \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p}|\beta_j|^q \right\}$$

For $q \geq 0$. Notes

- $q = 0$ is *best subset selection*
- $q = 1$ is LASSO
- $q = 2$ is ridge
- $0 < q \leq 1$ is non-differentiable
- $1 < q < 2$ is a differentiable compromise between LASSO and ridge

C. Elastic Net

Another LASSO/ridge compromise:

$$\tilde{\beta}(\lambda) = \text{argmin}_\beta \left\{ \sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p}(\alpha|\beta_j| + (1-\alpha)\beta_j^2) \right\}$$

Notes:

- L_1 penalty takes care of variable selection
- L_2 penalty helps in correctly handling correlation
- α defines how much L_1 and L_2 should be used.
  - α is a tuning parameter that must be found independently of λ
  - A grid search is discouraged
  - Often close to zero or one in practice

D. Least Angle Regression

Least angle regression or LAR is a "democratic" version of *forward selection* which sequentially adds new predictors into the model.

Eventually, it reaches the least square estimator and is strongly connected with LASSO.

LASSO can be seen as a special case of LAR, and LAR is often used to fit LASSO models.

Procedure:

1. Standardize the predictors (mean zero, unit norm) and initialize:

   Residuals $r = y - \tilde{y}$

   Regression coefficent estimates $\beta_1 = \cdots = \beta_p = 0$

2. Find the predictor x_j that is most correlated with r

3. Move $\hat{\beta}_j$ towards its least-squares coefficient $\langle x_j, r\rangle$ until for $k \neq j$, $\text{corr}(x_k, r) = \text{corr}(x_j, r)$

4. Add x_k in the active list and update both $\hat{\beta}_j$ and $\hat{\beta}_k$ towards their joint least-squares coefficent until x_l has as much correlation with the current residual

5. Continue until all p predictors have been entered

E. Group LASSO

Suppose that p predictors are grouped into L groups, group LASSO minimizes the following:

$$\hat{\beta}(\lambda) = \text{argmin}_\beta \left\{ ||(y - \beta_0\vec{1} - \sum_{\ell=1}^{L} X_\ell \beta_\ell||_2^2 + \lambda \sum_{\ell=1}^{L} \sqrt{p_\ell}||\beta_j||_2 \right\}$$

Sparsity is encouraged at group level

F. Non-Negative Garrote

The origin of LASSO, we minimize the following:

$$\hat{\beta}_{\text{GARROTE}}(\lambda) = \text{argmin}_\beta \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} c_j \beta_j x_{ij})^2 \right\}$$

Subject to $c_j \geq 0$ and $\sum_j c_j \leq t$.

G. The Oracle Property

Let:

- $\mathcal{A} := \{j : \beta_j \neq 0\}$ be the set of true relevant coefficents.
- δ be a fitting procedure (e.g. LASSO, non-negative garrote, ...)
- $\hat{\beta}(\delta)$ be the coefficient estimator for the procedure δ

We would like that δ:

1. Identifies the right subset model $\{j : \hat{\beta}(\delta) \neq 0\} = \mathcal{A}$
2. Has the optimal estimation rate $\sqrt{n}(\hat{\beta}(\delta)_\mathcal{A} - \beta_\mathcal{A}) \xrightarrow{d} N(0, \Sigma)$ where Σ is the covariance matrix for the true subset model

If δ asymptotically satisfies these conditions, it is defined an *oracle procedure*.

H. Adaptive LASSO

$$\hat{\beta}_{\text{WEIGHT}}(\lambda) = \text{argmin}_\beta \left\{ \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} c_j \beta_j x_{ij})^2 + \sum_{j=1}^{p} w_j |\beta_j| \right\}$$

In which $\hat{w}_j = \frac{1}{|\hat{beta}_{\text{OLS}}|^\gamma}$

Notes:

- Enjoys the oracle properties
- When $\gamma = 1$ it is very closely related to the non-negative garrote
- Relies on $\hat{\beta}_j^{\text{OLS}} \rightarrow$ sometimes LASSO used in a first step
- 2-D tuning parameter

V. LECTURE 5

A. 1-D Kernel Smoothers

KNN has drawbacks; ugly discontinuities and the same weight to all points regardless of their distances to x.

We can remedy this by weighing the effect of each point based on its distance:

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{N} K(x_o, x_i) y_i}{\sum_{i=1}^{N} K(x_0, x_i)} \tag{1}$$

Where:

$$K_\lambda(x_0, x) = D\left(\frac{|x - x_0|}{\lambda}\right) \tag{2}$$

D is called the *kernel* and λ is the *bandwidth* or *smoothing parameter*.

D must be symmetric around x_0, and go off smoothly as a function of distance; here are some common choices:

Nucleus	$D(t)$	**Support**
Normal	$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}t^2\right)$	$\mathbb{R}$
Rectangular	$\frac{1}{2}$	$(-1,1)$
Epanechnikov	$\frac{3}{4}(1-t^2)$	$(-1,1)$
Biquadratic	$\frac{15}{16}(1-t^2)^2$	$(-1,1)$
Tricubic	$\frac{70}{81}(1-\lvert t\rvert^3)^3$	$(-1,1)$

1. *Choice of Bandwidth λ*

λ controls how large an interval is considered around x_0.

For *Epanechnikov*, *biquadratic*, and *tricubic* kernels, the radius of the support.

For *Gaussian* kernel, uses the standard deviation.

Larger values imply lower variance but higher bias

- λ small $\implies$ $\hat{f}(x_0)$ based on a few points $\to$ y_i's closer to y_0
- λ large $\implies$ more points $\implies$ stronger effect of averaging

Alternatively, we can adapt to the local density (fix k as in KNN), we can express it by substituting λ with $h_\lambda(x_0)$. Finally, we can keep the bias constant, while the variance is inverse proportional to the local density.