Review

Exam 2, cross-validation, feature engineering, Bayesian Estimation, Time Series, etc.

DS 6030 | Fall 2021

review.pdf

Contents

---

1 Exam 2 questions?

2 Cross-Validation

Cross-Validation useful for doing two things:

1. Model Assessment: Assessing out of sample performance
2. Model Selection: Choosing best model and/or tuning parameter(s)

2.1 Overview of resampling based model assessment

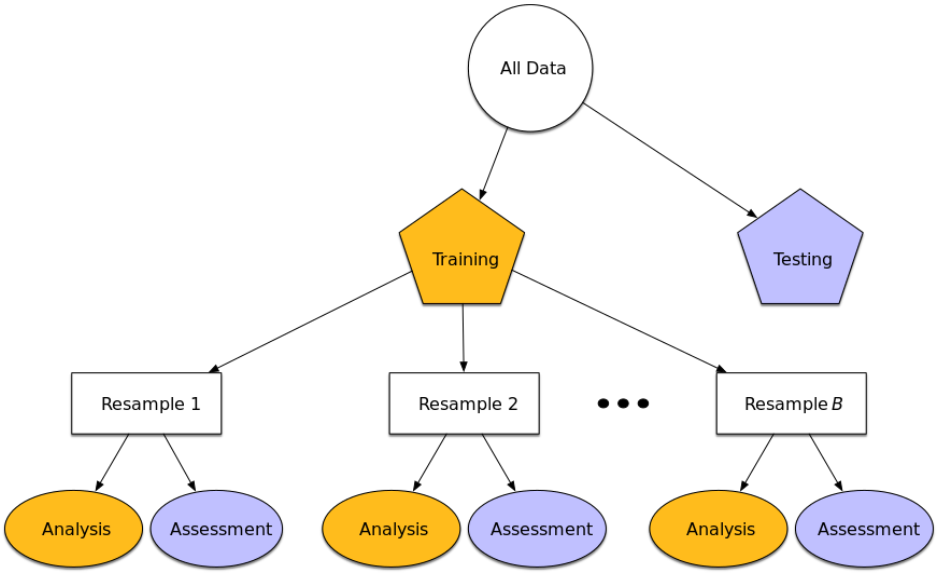
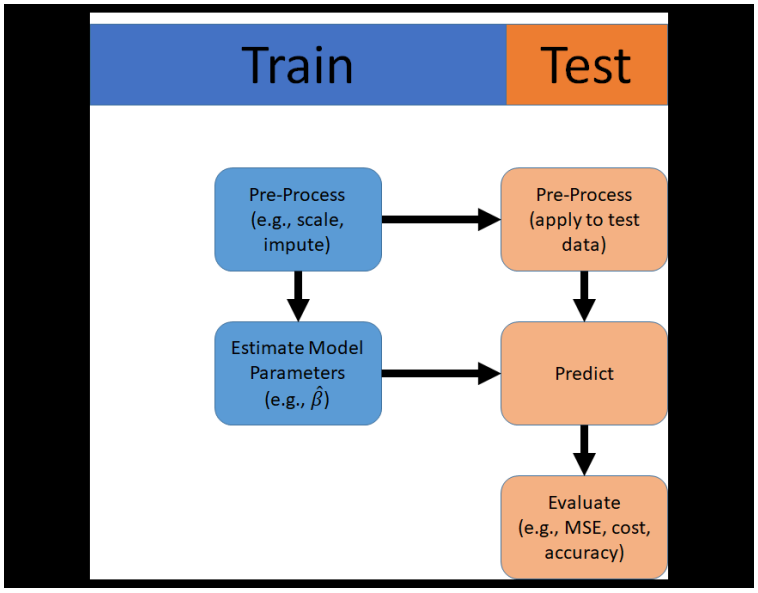
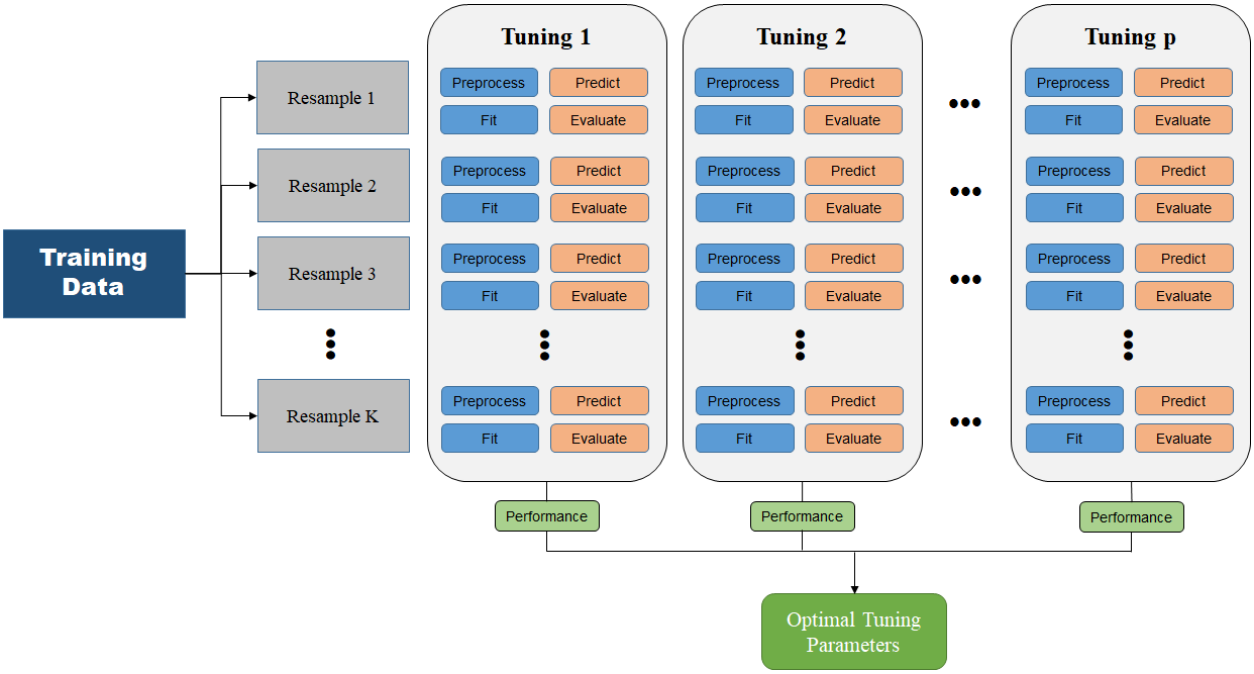


Figure taken from: **Feature Engineering and Selection: A Practical Approach for Predictive Models** by Max Kuhn and Kjell Johnson (2019-06-21)




Training Data
Resample 1
Resample 2
Resample 3
Resample K
Tuning 1
Tuning 2
Tuning p
Preprocess
Predict
Fit
Evaluate
Performance
Optimal Tuning Parameters

3 Penalized Regression (elastic net)

4 Feature Engineering

Feature Engineering and Selection: A Practical Approach for Predictive Models by Max Kuhn and Kjell Johnson

> ... we are sometimes frustrated to find that the best models have less-than-anticipated, less-than-useful useful predictive performance. This lack of performance may be due to a simple to explain, but difficult to pinpoint, cause: **relevant predictors that were collected are represented in a way that models have trouble achieving good performance**.
>
> Key relationships that are not directly available as predictors may be between the response and:
>
> - a transformation of a predictor,
> - an interaction of two or more predictors such as a product or ratio,
> - a functional relationship among predictors, or
> - an equivalent re-representation of a predictor.
>
> Adjusting and reworking the predictors to enable models to better uncover predictor-response relationships has been termed *feature engineering*.

4.1 Feature Selection and Dimension Reduction

4.1.1 Difference between Feature Selection and Dimension Reduction

- *Feature Selection*: only use a subset of available/collected predictors
  - E.g., best subsets
- *Dimension Reduction*: reduce the number of predictors/parameters used by a model
  - E.g., principal component regression (PCR)

4.1.2 Feature Selection

See Feature Engineering for more details and ideas.

Goals:

1. Cost and time savings: Collecting predictors can be expensive
2. Reduce model variance: removing predictors lowers model variance
3. Interpretation: easier to understand model with fewer relevant predictors

Three main approaches:

1. Embedded

Embedded feature selection methods are build into the model/algorithm. Examples include trees and lasso.

2. Wrappers

Wrapper methods for feature selection attempt to find the best subset of features for a particular model. They can *sometimes* perform better than embedded methods (e.g., Boruta), but they will involve extra computation.

Examples include fully enumerated *best subsets* and evolutionary optimization algorithms like *genetic algorithms*. Can consider this a binary integer programming optimization.

An example of *False Selection Rate (FSR)* feature selection designed for random forest is called **Boruta**.

- Boruta for trees
  - Introduce additional shuffled features
  - Calculate importance scores for all features
  - Record the "hits": all original features with importance scores greater than *max importance from all shuffled features* (these features are deemed important)
  - Repeat the process M time (100 by default; or sequential)
  - Determine which predictors have significantly more "hits" than expected under null of not-important.

3. Filters

Filter methods are the quickest, but not usually the best. Basically, filter methods do feature selection first before any modeling is done. For example, run p simple linear regression models (one for each predictor) and keep the features with significant coefficients for further modeling.

Although they are sometimes claimed to be model-free, most (all?) filter methods do (implicitly) have a model they are using to decide on the relevant predictors. There is no guarantee (or even assurity) that features selected by the filter are appropriate for the final model.

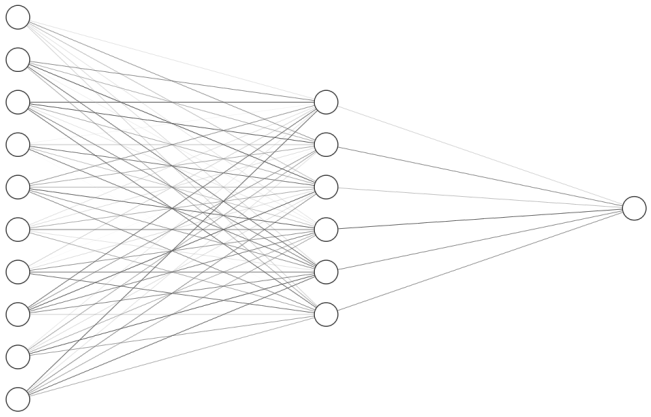
4.1.3 Dimension Reduction

See the class notes on Dimension Reduction with Derived Features and ISLR 6.3. for more details.

Basically, dimension reduction methods are based on transforming the raw data (e.g., PCA) and then using a subset of the transformed predictions.

Principal Component Regression (PCR) and Partial Least Squares (PLS) are classic examples of dimension reduction.

- These don't actually remove predictors since using linear combination. So you can gain on reduced model variance, but don't get easier interpretation and still need all input predictors.



5 Time Series

- Forecasting: Principles and Practice (2nd ed). Rob J Hyndman and George Athanasopoulos

Key Issues:

- Sequential Data (observed at regular times)
- Non-independent observations
  - What happens tomorrow depends on what happens today (and yesterday, etc)
- Seasonality
  - Repeating patterns
  - What happens today will be similar to what happened at this time last year (or week or month, etc)
- Desire predictions h units in future.

5.1 Basic models:

1. AR (regression)

Use past observations as the predictors.

$$\hat{y}_{t+h} = \hat{\beta}_0 + \sum_{i=1}^{p} \hat{\beta}_0 y_{t-i}$$

2. Exponential Smoothing

Example: Holt-Winters Additive Model

$$\begin{aligned}
\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t+h-m(k+1)} \\
\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m},
\end{aligned}$$

3. Several deep learning models
   - RNN (e.g., (LSTM) Long short-term memory network)
   - CNN (e.g., (TCM) Temporal convolutional network)

5.2 Fitting/Tuning

- Loss functions include MSE, MAE, and percentage error (MAPE).
- Can't use cross-validation due to the sequential ordering of the data. Instead using rolling-windows (back-testing).

6 Bayesian Estimation

Two purposes:

- full (posterior) distribution incorporating *prior beliefs*
- regularize or penalize for unfeasible solutions
  - I.e., prior beliefs suggest model coefficients should be close to zero.
- The approaches we've covered in class can somewhat mimic these
  - use bootstrap resampling to get approximate distribution
  - use penalties to remove implausible solutions (e.g., betas close to 0)
  - See Bootstrap Notes for example

6.1 Bayesian Estimation

In Bayesian analysis, the parameter(s) are *random variables*.

- In MLE, the parameters are assumed fixed, but unknown.

Prior knowledge, any information known about the parameter(s) *before the data are seen*, is captured in the *prior distribution*.

- Let $g(\theta)$ be the (possibly multivariate) prior pmf/pdf

Bayes theory gives us the *posterior distribution*,

$$f(\theta|D) = \frac{P(D|\theta)g(\theta)}{\int_{\theta\in\Theta} P(D|\theta)g(\theta)\,\mathrm{d}\theta}$$

- $P(D|\theta) = P(X_1, X_2, \ldots, X_n) =$ *likelihood*
- $\int_{\theta\in\Theta} P(D|\theta)g(\theta)\,\mathrm{d}\theta = P(D)$ is the *normalizing constant* (not function of θ).
- $P(\theta|D)$ is the *posterior distribution*, which contains the updated knowledge about the parameter(s).

6.1.1 Bayesian Point Estimation

1. Posterior Mean

$$\hat{\theta}_{\mathrm{PM}} = E[\theta|D] = \int_{\theta\in\Theta} \theta f(\theta|D)\,\mathrm{d}\theta$$

2. MAP (Maximum a posteriori)

$$\begin{aligned}\hat{\theta}_{\mathrm{MAP}} &= \arg\max_{\theta\in\Theta} f(\theta|D) \\ &= \arg\max_{\theta\in\Theta} P(D|\theta)g(\theta) \\ &= \arg\max_{\theta\in\Theta} \log P(D|\theta) + \log g(\theta)\end{aligned}$$

7 Deep NN vs. classical stat learning

- See ISLR Chapter 10 for good introduction to Deep learning from a statistical learning perspective

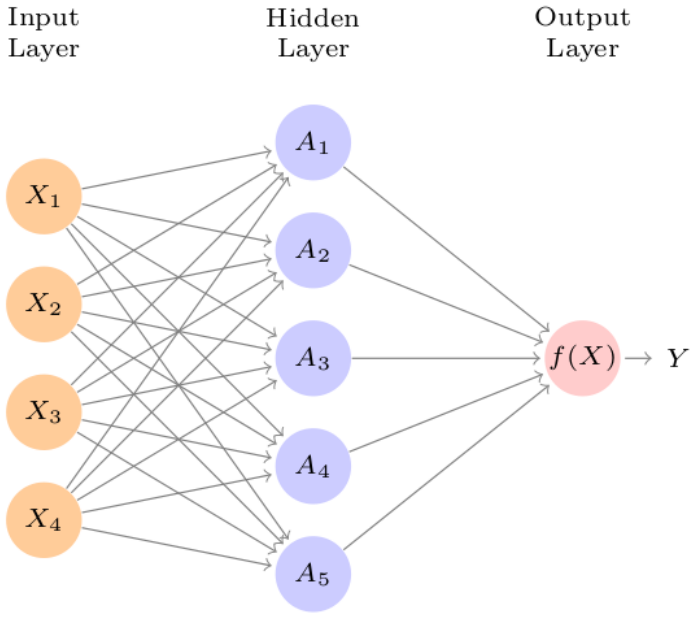


FIGURE 10.1. *Neural network with a single hidden layer. The hidden layer computes activations $A_k = h_k(X)$ that are nonlinear transformations of linear combinations of the inputs $X_1, X_2, \ldots, X_p$. Hence these A_k are not directly observed. The functions $h_k(\cdot)$ are not fixed in advance, but are learned during the training of the network. The output layer is a linear model that uses these activations A_k as inputs, resulting in a function $f(X)$.*