**Model Building and Scoring for Prediction**

What is an unbiased estimator

What is meant by best

Regularized regression Ridge, LASSO, Elastic Net

Compare and contrast Cross-validation

i. Model Building

Linear Regression Assumptions

* Linear regression is the **best linear unbiased estimator** (BLUE)
  + Estimator 🡪 slope you use to help estimate the predictor

BLUE

* Statistical/estimator biased // does not mean there is sampling bias
* Estimate the true s (can't see) - if on avg is aiming towards the true , it is unbiased
  + Take many samples and find avg
  + That avg would be the truth of what we are estimating
* The statistics have been proven as best linear estimator
* True s2 is a good estimator of σ2 🡪 shooting at the target in the center, not un-centered



* s are centered at the truth // aiming at the true slope that exists
* **What does it mean to be unbiased?**



* + notion that this beta hat this you have trying to estimate true beta , which you never see
  + If the distribution of statistic is centered over truth - its called unbiased
  + On avg, coefficients from all samples are centered around the true coefficients
* **What does it mean to be best?**
  + IF assumptions hold, is the minimum variance of all unbiased estimators
  + **If assumptions hold**, the spread of s (every sample gives a different sampling distribution spread) won’t be overly wide
  + Best - spread of guesses are as narrow as it get
  + Best & unbiased - how good is your guess
    - If it is aiming at the middle of the right target, the spread isn't wide
  + Might make prediction better at the cost of being unbiased
  + Not aiming at the true slope (getting biased): at the cost of interpretation
  + Can I find a way to make a different relationship between x and y more predictive at the cost of the true relationship --> regularized regression
* What if assumptions don’t hold?
  + What if biased estimators had smaller variance? (you lose interpretability for better predictions)

ii. Regularized Regression

Potential Problems

* As the number of variables increases, more problems tend to arise
  + Assumptions start to fail, multicollinearity concerns
  + Coefficients are changing and not trustworthy // multicollinearity can lead to overfitting
    - Choose prediction or interpretation
  + Regularized regression - can have multicollinearity
    - **Assumption**: at least **10 obs** per var
* Multicollinearity problems 🡪 coefficients vary widely
  + Variations lead to overfitting (only predicting the training data well, but not generalizing to the test dataset)
  + Higher variance than desired
* More vars than observations (genetic modeling)

Regularized Regression

* Regularized regression 🡪 (penalized / shrinkage regression) puts constraints on the estimated coefficients in our model and shrink these estimates to 0
  + Regularized 🡪 penalizing the model, changing the coefficients by shrinking them (making them closer to 0)
  + Moving s bc think might get a better prediction
  + Giving up interpretability of for the hope of being able to predict better
* Coefficients become biased, but potentially improve variance of the model
  + Changing coefficient and makes it biased (no longer aiming for true relationship)
  + Willing to give up interpretability to relax assumptions to make prediction better
  + A diagram of a diagram of a certain type of matter

    Description automatically generated
  + Aim for the center of the board: true β
  + Each sample = single dart thrown at board
  + Aiming for center, but probably won't fit the center
  + Unbiased estimator 🡪 shoot towards the middle, but there is a spread going on
    - Good: least amount of spread you can have when aiming for the center
  + Regularized regression: aiming slightly off target, but hoping of spread of possible errors is smaller // predictions would potentially be better
  + Guess is closer to the truth in an biased way = better prediction
    - If we give up some interpretability, we might be able to predict better

Penalties in Models

* Ordinary least squares (OLS) minimizes the sum of squared errors (SSE)
  + Take all vertical distances, square the, = best line // find minimum of the errors
* Regularized regression introduces a penalty term to the minimization:
  + Dart board: off center = biased // less spread = trade off
  + Still minimizing MSE, and penalize the model
* Penalty 🡪 slightly moving off the center of the target, but still get a better prediction overall // moves too much - aim is farther away from center
  + Penalty is moving the close knit darts farther from the middle
  + Want to make it so there is less spread, but that it is still close to the middle
  + Box-cox - OLS is the best Y to find X and Y relationship
    - Sticking to OLS as best shot, blend and change model to fix it
  + Regularized: don't changed the Y

iii. Ridge Regression

Penalties in Models

* Ridge regression introduces an “L2” penalty term to the minimization:
  + Taking all , sum the effects of those coefficients // make the all positive by squaring the betas (L2 penalty)
  + Maybe make SSE smaller if playing around with betas
* Penalty is controlled by tuning parameter, 𝜆
  + if , the its OLS (there is no penalty)
  + 𝜆 🡪 ∞, coefficients shrink to 0
    - 𝜆 = bias, the bigger the number the more biased the regression will be
    - If , there is no 2 half of the equation, there is just OLS
    - As 𝜆 gets bigger, no longer just OLS
    - Only thing that can change is the slope (X and Y are fixed)
    - Make smaller as 𝜆 gets bigger to counteract

iv. LASSO Regression

Penalties in Models

* Least absolute shrinkage and selection operator (LASSO) regression introduces an “L1” penalty term to the minimization:
* Penalty is controlled by tuning parameter, 𝜆
  + if , the its OLS (there is no penalty)
  + 𝜆 🡪 ∞, coefficients shrink to 0
    - Ridge: approaches 0 asymptotically, lasso can remove variables
* Take all vertical distances, square the, = best line // find minimum of the errors

A graph of a number and a line

Description automatically generated

* Estimates of MSE
* Red dot = avg error across folds
* Interval = variability of the error
* Minimum MSE, find beta that minimizes the error // mathematically best uses all vars
* One standard error above the minimum 🡺 if willing to account for variability pf the dots, you can get something close to the minimum error
  + Can get rid of vars to get close to minimum 0, one SE above to get the close error so model can have less var (trade off for min error)

v. Elastic Net Regression

Penalties in Models

* Both ridge and LASSO have advantages and disadvantages
  + LASSO does variable selection
  + Ridge keeps all variables (LASSO drops arbitrarily)
* Elastic net regression combines both penalty terms in the minimization:
* The glmnet function in R takes slightly different approach:
  + Any value of α between 0 and 1 gives a combination of both penalties (elastic net)

|  |  |
| --- | --- |
| A graph of different colored lines  Description automatically generated | * What happens to betas/coefficient when 𝜆 changes   + As 𝜆 increase, decreases   + All lines (all 36 vars/slopes) get closer to 0, can't interpret s from regularized regression   + Rigged regression helps with multicollinearity, forces it to look more reasonable before it reaches 0 |

* Ridge, shrink vars // Lasso can remove var // Elastic net - best between

vi. Optimizing Penalties

Fear of Overfitting

* Need to select 𝜆 for any of the regularized regression approaches
* Don’t want to minimize variance to the point of overfitting our model to the training data

Cross Validation

* Cross-validation (CV) 🡪 approach to prevent overfitting when tuning a parameter
  + Tune models - change some component of model and see how well it fits after
  + Almost like step-wise, ask enough question, figure out the answer on that dataset, not generalizable
  + Split data into many piece, protecting self from over-fitting on one slice of data
* Concept:

1. Split training data into multiple pieces
2. Build model on majority of pieces
3. Evaluate on remaining piece
4. Repeat process with switching out pieces for building and evaluation

k-fold Cross-Validation

A screenshot of a graph

Description automatically generated

* Hold out 1st 10% of data and build model on the 90%
* Put 10% back in and take out another 10% put and build it on the 90% until completed for all slices

vii. Model Comparison

Comparing Models

* The model results in a formula or rules.
* The data require modifications:
  + Derived inputs
  + Transformations
  + Missing value imputation
* To score/compare, you do not rerun the algorithm!
* Apply score code (equations) obtained from final model to the test data for comparing
  + Get to test data set, score the equation // plug in Xs to predict Y to get predictions
  + Plug in new x values for the equation you get to score // don’t rerun, can overfit
* Test dataset is for comparing final models and reporting final metrics
* DO NOT GO BACK AFTER TO REBUILD MODEL!
* DO NOT JUST BUILD 1000’s OF MODELS TO COMPARE IN THE TEST SET!
* We do not want to fit to the test dataset as it is our honest assessment of how good our models can do

Model Metrics

* Root MSE:
  + Note easily interpretable
* Mean absolute error (MAE):
  + Not scale invariant
* Mean absolute percentage error (MAPE):
  + Not symmetric