PREDICTION

MODEL BUILDING FOR PREDICTION

Tamer Çetin

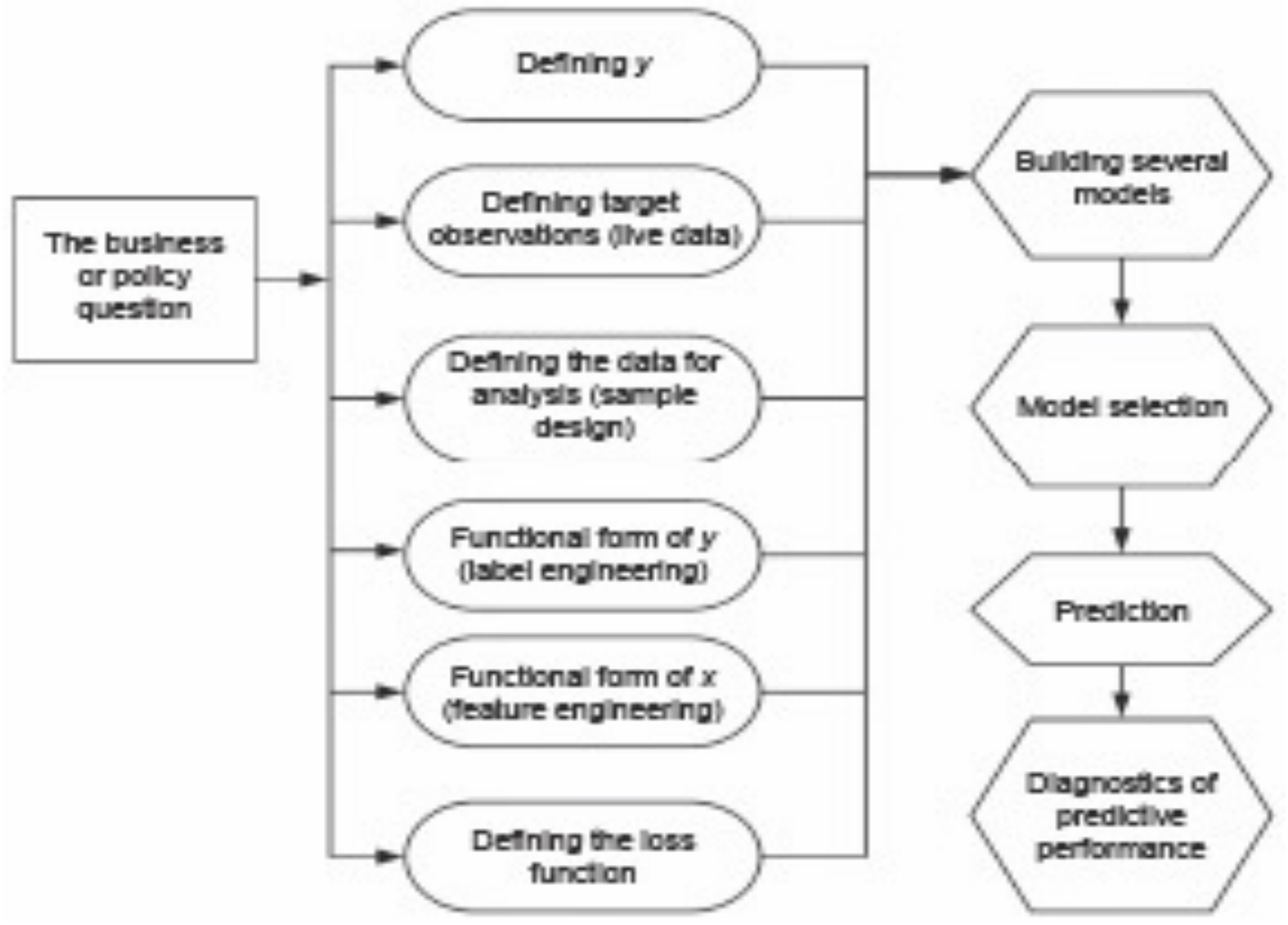
Model Building for Prediction

- We have learned to use BIC and CV to select the best regression model for prediction and how to think about its external validity.
- But, how should we specify regression models for prediction?
- In particular, when we have too many predictors, maybe more than observations, and how should we decide on their functional forms?
- In this lecture, we will explore;
  - how to build/specify regression models for prediction,
  - how to evaluate the predictions they produce,
  - how to select variables to use in prediction models
  - when to take logs of y,
  - and the functional forms of predictions.

Model Building for Prediction: Learning Outcomes

- After walking through this lecture, you should be able to:
  - understand the process of specifying regression models for prediction,
  - carry out predictions of y using models with $\ln y$,
  - use LASSO to specify a regression model for prediction and understand how to use this method for variable selection.
  - define and use a holdout sample to evaluate a prediction using various diagnostic tools.

Steps of Prediction


The business or policy question
Defining *y*
Defining target observations (live data)
Defining the data for analysis (sample design)
Functional form of *y* (label engineering)
Functional form of *x* (feature engineering)
Defining the loss function
Building several models
Model selection
Prediction
Diagnostics of predictive performance

Sample Design

- Sample design means selecting observations from the raw data (dataset we collected) to create the data we will use for predictive analysis.
- This (sample) data we will use for prediction analysis is called original data.
- Original data is assumed to represent the future live data but derived from the raw data by using the selected observations from the raw data.
- Note that the main task of a prediction is to use patterns of associations between y and x in original data to predict y in live data.

Sample Design

- Because we do not have live data from the future for the current prediction analysis, we cannot observe y the in the future or live data.
- Second issue with sample design is the existence of missing values in data.
- Such observations should be dropped from data.
  - We cannot use them for prediction y.
- Another issue is extreme values or outliers.
  - We should remove outliers from data to avoid their potential impact overfitting.

Future Engineering: Dealing with Missing Values

- Label engineering is about the functional form of y.
  - Numerical?
  - Categorical/binary?
  - Level vs log?
  - Scaling: Standardization/Normalization?

Label Engineering and Predicting Log y

- Whether we should transform y into a log value for prediction is an important modelling choice but not the one that is straightforward.
- Mostly, we try level and log and select one of them giving the better prediction, for instance, by CV.
- Moreover, we want to predict the original y variable not its log.
- Thus, when we have $\ln y$ in the prediction model, we want to transform its predicted value back to y.
- Note that the reason behind the log correction is a little complicated.
  - First, the exponential form is not linear
  - Second, regression and most other predictions are models of the expected value.

Label Engineering and Predicting Log y

- When we use ln y, regression predicts the expected value of (mean) of ln y, while we need the expected value (mean) of y.
- The problem is that the mean of ln $e^{ln\, y}$ is not the same as the mean of y since the log function is not linear.
- We use a correction term as approximation to get $\hat{y}$ from $\widehat{lny}$.
- This correction/adjustment is a function of the standard deviation of the residuals of the regression model.
- So, this is correct only when is lognormally distributed conditional on the conditioning x variables.
  - Because the closer the distribution y is to lognormal, the better the approximation, when log values of y are representative to its level values we can use ln y.

Feature Engineering: Dealing with Missing Values

- Feature engineering is the part of prediction analysis when we take the variables in the raw data and create the x variables/features that we will include in the predictive model.
- The first part of feature engineering is data cleaning.
  - We just want to pick the variables we want to use!
- This includes exploring and handling missing values.
  - Missing value is an observation we may not be used for prediction!
  - In some cases, we want to imput data as a set of observations with all predictors x and target variable y that need to be non-missing!
- We should start by describing missing value problem, variable by variable, listing the percent of missing values for each feature.

Feature Engineering: Dealing with Missing Values

- As we said above, the first option is to keep observations with missing values and replace missing values with something else.
    - For qualitative features, we may simply create another category as a corresponding binary variable.
    - For quantitative features, we can replace missing value with imputed value.
- Note that binary (flag) variable's value does not matter for prediction if it is randomly missing.
- But we need to be careful if it is a missing variable as the result of some selection.
- Also, we need to check the source of why values are missing:
    - In some cases, missing means zero so imputed value should be zero.

Feature Engineering: Dealing with Missing Values

- Second option is to remove the x variable.
    - When a large fraction of observations in a variable have missing values, removing the feature is the best option.
- This comes with some costs because we will lose some variation and comparison in prediction analysis.
- For that reason, if we think the same missing values are the same/similar to live data, we should comfortably remove this feature from data.
- Third option is to drop the observations with missing values.
    - This is recommended only if there are very few missing values for a few variables.
    - Otherwise, we can end up with losing too much variation in data.

Feature Engineering: What x Variables to Have and in What Functional Form

- The second part of feature engineering is feature selection.
  - What variables to have in prediction model and in what functional forms including potential interactions.
- The most common way to start is to extract data with all features and do EDA and feature engineering to see what variables are usable.
- But if you know the functional forms of prediction models you will use in modelling, you can start with selection features according to the complexity of functional forms of prediction models.
  - Less complex models would have fewer x variables in simpler functional forms.
  - More complex models would have more x variables in more complicated non-linear functional forms and maybe even with more interactions.

Feature Engineering: What x Variables to Have and in What Functional Form

- Specifying the functional form of the x variables is another difficult task of feature engineering.
- This includes capturing nonlinear relationships with quantitative x variables (such as quadratic and other polynomial functions), deciding on the number of qualitative variables and interactions.
- Note that the idea with all this feature engineering process is to get the best fit without overfitting data.
- But how should we answer all the above questions using feature engineering?
  - Domain knowledge once again!

Feature Engineering: What x Variables to Have and in What Functional Form

- In this sense, domain knowledge includes the experience with previous prediction analyses and/or theory about what tends to make y different.
- With such an experience, you can answer what x variables are likely to be more important versus less important, what interactions are likely important, and where should we be most worried about nonlinearity.
- The other factor is data itself.
- This is more important than domain knowledge in feature engineering:
  - First, prediction is a data-driven analysis.
  - Second, conditions in data vary over time.
-

Feature Engineering: What x Variables to Have and in What Functional Form

- EDA is a key part of feature engineering in predictive analysis.
- We do EDA:
  - To make sure we understand the content of each x variable,
  - To make sure they are as clean as possible,
  - To understand their distribution and
  - To check correlation among x variables.
- We do EDA to better understand data because we should never build models without understanding data and its variables.

We Cannot Try Out All Possible Models

- As we have seen, feature engineering is a long and tedious task.
- Some of it is unavoidable to know data.
- But other parts of feature engineering would be replaced by some algorithm or models, at least in principle.
- For instance, after doing missing and extreme values analysis, we can use all the remaining variables to run many different prediction algorithms, even clustering models.
    - The results will be deceptive even if we get some results!
    - Why?

Evaluating the Prediction Using a Holdout Set

- Remember that the basic idea with the training-test split of data is to use one part of data (training set) for model building, and another part (test set) for looking at how good the resulting model is at predicting y.
  - The reason for this split is to avoid overfitting data we use for model building!
  - The best model should fit the test set the best
- CV is an improved version of train-test split.
  - K-fold CV would be considered train-test split with k test sets.
- Note that when we picked the best model, we should go back and use all the data with no splitting for the final estimate and to make a prediction.

Evaluating the Prediction Using a Holdout Set

- That is all god but what part of data we should use to evaluate the final prediction on test data set?
- To answer that questions, we need data to evaluate the prediction on that is just like our original data, but which we have not used for estimation.
- Solution is similar to the training-test split.
- We do all of the work using one part of data:
  - Model building, selecting the best model by CV, and then making prediction by estimating coefficients of the linear regression.

Evaluating the Prediction Using a Holdout Set

- We start with the entire **original data**.
- First, we select the observations in the holdout set, by random sampling.
  - That's usually a small part of the original data, such as 20%.
- The remaining part of our data is our work set including training and test sets.
  - We further divide the work set k-times for training-test splits.
  - We build the models on the training sets, and pick the best model by how they fit the test sets – e.g., by picking the one with the smallest average MSE.
  - Then, we take that model and re-estimate it on the entire work set (the original data less the holdout set).
- Finally, we use that estimated model on the holdout set to see how it predicts y.

Evaluating the Prediction Using a Holdout Set

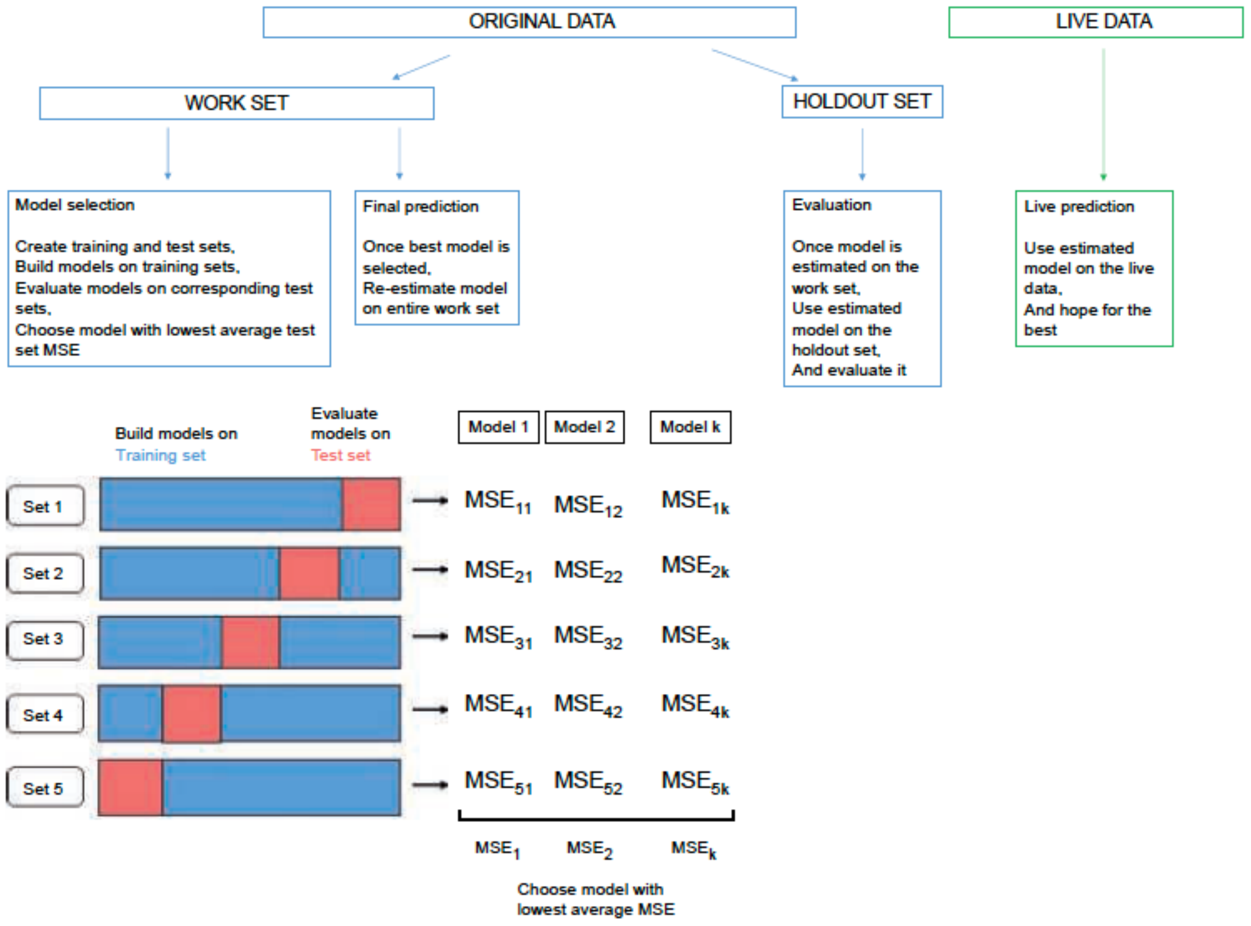

ORIGINAL DATA
LIVE DATA
WORK SET
HOLDOUT SET
Model selection
Create training and test sets, Build models on training sets, Evaluate models on corresponding test sets, Choose model with lowest average test set MSE
Final prediction
Once best model is selected, Re-estimate model on entire work set
Evaluation
Once model is estimated on the work set, Use estimated model on the holdout set, And evaluate it
Live prediction
Use estimated model on the live data, And hope for the best
Build models on Training set
Evaluate models on Test set
Model 1
Model 2
Model k
Set 1
Set 2
Set 3
Set 4
Set 5
MSE11 MSE12 MSE1k
MSE21 MSE22 MSE2k
MSE31 MSE32 MSE3k
MSE41 MSE42 MSE4k
MSE51 MSE52 MSE5k
MSE1 MSE2 MSEk
Choose model with lowest average MSE

Figure 14.6 Illustration of the uses of the original data and the live data

Selecting Variables and Regularization in Regressions by LASSO

- So far we have used OLS models to fit regressions on data.
- OLS is an intuitive method that tends to give a good approximation to average differences in y related to the difference in x.
- However, we can get better predictions from regression using some other methods.
- One approach is called shrinkage.
- Shrinkage models start with a regression with many right-hand-side features:
  - p of them, where p is a large number, sometimes larger than observations.

Selecting Variables and Regularization in Regressions by LASSO

- Even if the original x variables are not many, sometimes after feature engineering we can have additional so many variables.
- We need to handle this because it can create some problems in prediction such as the use of unnecessary variables stemming from curse of dimensionality and sparsity.
- The most widely used shrinkage method is LASSO (Least Absolute Shrinkage and Selection Operator).
- LASSO is an algorithm fitting a model by shrinking coefficients.
    - More importantly, shrinking some of them to zero!

Selecting Variables and Regularization in Regressions by LASSO

- Using LASSO, we can do two important things in prediction at the same time:
  - Selecting a subset of x variables fed into it to stay in regression while dropping the other variables.
  - Shrinking coefficients (compared to OLS) for some variables left in regression.
- The output of LASSO is a list of variables and their estimated coefficients!
- The way LASSO does this is by adding a penalty term to the usual OLS estimation algorithm.
- This term penalizes the sum of the absolute values of the coefficients!

Selecting Variables and Regularization in Regressions by LASSO

- When the estimated coefficients are closer to zero, the penalty term is smaller!
  - When coefficient is zero, there is no penalty term: the best prediction!
- Consider a linear regression as follows:

$$y^E = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = \beta_0 + \sum_{j=1}^{p} \beta_j - x_j$$

- where observations are n with each labeled with i and p are variables each labeled with j.

Selecting Variables and Regularization in Regressions by LASSO

- Recall that the usual way to estimate the coefficients of such as regression is by OLS:
  - minimizing the sum of squared residuals.

$$\underset{\beta}{minimize} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{p} (\beta_j x_{ij}) \right) \right)^2$$

- LASSO modifies this minimization by adding a penalty term.

$$\underset{\beta}{minimize} \sum_{i=1}^{n} \left(y_i - \left(\beta_0 + \sum_{j=1}^{p} (\beta_j x_{ij}) \right) \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

Selecting Variables and Regularization in Regressions by LASSO

- The penalty term allows LASSO to reduce the OLS coefficients' absolute values.
  - As a result, LASSO tends to reduce the absolute values of many coefficients to zero and thus drop them from the regression model!
- λ in the formula above is called the tuning parameter.
- It serves as a weight for the penalty term versus the OLS fit.
- Thus, it drives the strength of variable selection.
  - If λ is zero, prediction is the OLS model.
  - As λ increases, it leads to more aggressive feature selection and fewer features left in the regression with smaller coefficients.

Selecting Variables and Regularization in Regressions by LASSO

- But, what values should λ have?
- The solution is to have your computer try out many different values for λ and pick the one giving the best prediction in the end using k-fold CV.
  - First, LASSO algorithm is run on the training set with a specific value λ:
    - Finding its estimated coefficients,
    - Giving the prediction of corresponding test set,
    - Calculating MSE,
    - And, repeating the same process for the other train-test splits.

Selecting Variables and Regularization in Regressions by LASSO

  - When all the process is done, it saves the average MSE across the test sets.
  - Then, it proceeds the same analysis with a different value of λ.
- In fact, LASSO algorithm is embedded in another algorithm called Search Algorithm that searches for the best λ.
- Many statistical packages have their own embedded search algorithms for CV search for λ as part of their LASSSO packages.
- These statistical packages tend to have sensible default options for the algorithm so we can run them without more detailed knowledge of how the algorithm works.

Selecting Variables and Regularization in Regressions by LASSO

- The only thing we need to decide for LASSO is the initial set of the x variables.
- For that reason, we should start with all possible features since LASSO will eliminate all the unnecessary variables.
- Finally, note that LASSO provides better OLS estimates, which are known to produce unbiased predictions.
- As a result, LASSO predictions become biased.
    - The larger λ, the larger the difference between LASSO and OLS
    - The larger λ, the bias of LASSO!

Selecting Variables and Regularization in Regressions by LASSO

- Recall that a biased prediction is not necessarily a worse prediction.
  - The total loss due to prediction error has two main parts:
    - Bias and variance!
  - So, it is possible to get a smaller total loss with a larger bias if the variance is smaller now.
  - And, this is exactly what LASSO achieves:
    - The search for the best λ finds the prediction with the smallest total loss, thus finding an optimal way to trade-off bias and variance!

Diagnostics

- The last step of prediction analysis process is to evaluate the actual prediction.
- At this point, we have a best model that we selected by CV.
- To make the final prediction, we estimate it using all of the work data.
- And, now we use the hold-out set of the observations from original data to evaluate the final prediction.
- Note that we have not done anything to this hold-out set, except for cleaning it.
- All EDA, model building, and model selection were done using the work set.

Diagnostics

- We retained the hold-out set for the evaluation of the final prediction.
- The process of evaluating the final prediction is called post-prediction diagnostics including the following three steps:
  - First, and most obviously, we want to evaluate the fit of the prediction using MSE or RMSE or another loss function to summarize the goodness of fit.
    - We can use a $\hat{y} - y$ graph to visualize it!
  - Second, we use the hold-out set to create the prediction interval around the prediction.
  - Third, and perhaps more importantly, we can zoom in to the kinds of observations we care about the most.
    - For instance, we can analyze the hold-out set to see whether and how the quality of our prediction varies across firms with different sizes, in different industries, or in different financial situations.

Diagnostics

- Finally, we should assess external validity and be explicit about that assessment when presenting the results of a prediction.
- Having a hold-out set makes sure that the prediction model is selected and estimated using a different part of the original data than the part used for diagnostics.
    - Meaning that the diagnostics show the fit and the uncertainty of prediction not in the data we have but more generally in the population, or general patterns, represented by original data.
- But we want to predict y in live data.
    - So using the predictions for subset of hold-out test for different observations at this point will help get some intuition about the prediction of y in live data in the future!

Prediction with Big Data

- We conclude the lecture with a note on Big Data and the additional opportunities and challenges it brings to predictive analytics.
- Big Data refers to many variables and data, which is continuously updated, but not only many observations.
- The most obvious advantage of using big data in prediction is that estimation error plays a little role in the prediction error.
- For that reason, big data analysis tends to produce better prediction than smaller data with the same variables.
- Many observations can help with picking the best model since they allow for smaller estimation error in training sets and a more accurate way to evaluate predictions in test sets.
  - Thus, model error tends to be smaller with big data!

Prediction with Big Data

- The relationship between big data and irreducible error depends on the number of variables in big data:
  - If we have more variables in big data, prediction tends to have smaller irreducible error.
    - Combined datasets!
  - If there is no change in the number of variables just because we have larger data, there will be no change in irreducible error.
- There are some challenges stemming from the use of big data.
  - Big data with more observations leads to spending too much time and computational resources for prediction from EDA to external validity.
    - The solution is to pick a modest random sample from big data to run the code for prediction analysis.
- Big data does not make positive contribution to external validity and domain knowledge.

Prediction with Big Data

- In sum, big data tends to decrease model error by minimizing estimation error.
  - So that we can have better prediction results!
- Using big data can be costly!
- Finally, some of the most important aspects of predictive analytics remain the same regardless of the number of observations, the uncertainty of prediction due to irreducible error, and potential issue with external validity.

Main Takeaways

- We can never evaluate all possible models to find the best model!
  - Model building matters in specifying models that are likely among the best.
  - LASSO is an algorithm that can help with model building by variable selection and regularization.
  - EDA and domain knowledge remain important alongside powerful algorithms for assessing and improving the external validity of prediction.