Modeling Recap

Justin Post

Modeling Ideas

What is a (statistical) model?

- A mathematical representation of some phenomenon on which you've observed data
- Form of the model can vary greatly!

Statistical learning - Inference, prediction/classification, and pattern finding

- Supervised learning - a variable (or variables) represents an **output** or **response** of interest
    - May model response and
        - Make **inference** on the model parameters
        - **predict** a value or **classify** an observation

Our Goal: Understand what it means to be a good predictive model (not make inference)

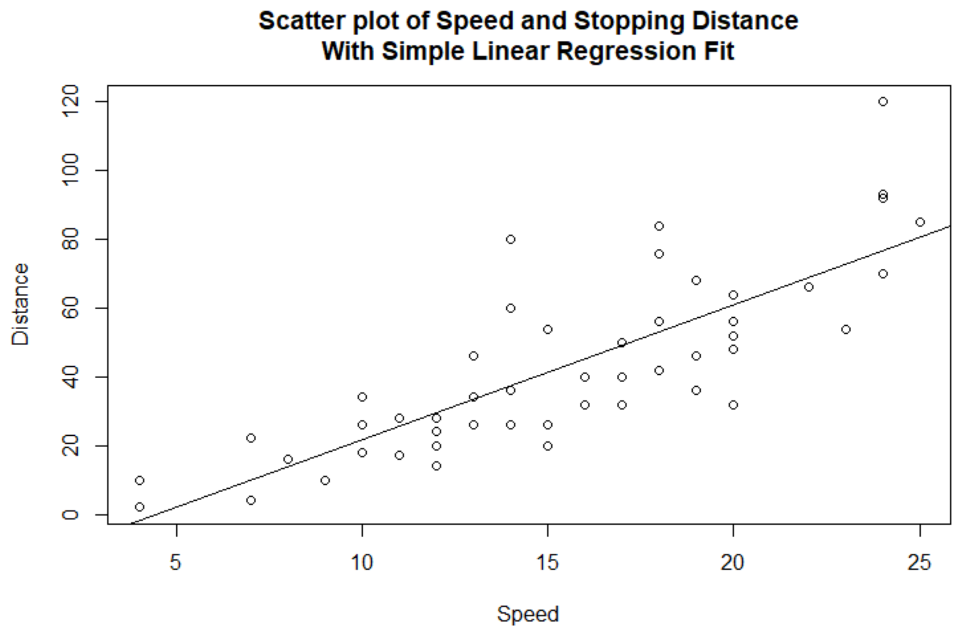
What is a Statistical Model?

- A mathematical representation of some phenomenon on which you've observed data
- Predictive model used to:
  - *Predict* a **numeric response**
  - *Classify* an observation into a **category**
- Common Supervised Learning Models
  - Least Squares Regression
  - Penalized regression
  - Generalized linear models
  - Regression/classification trees
  - Random forests, boosting, bagging

... and many more - tons of models!

Fitting a Model

Given a model, we **fit** or **train** it using the data



- Models can be used to yield predicted responses for each observation, call these $\hat{y}_i$

Quantifying How Well the Model Predicts

Need a way to quantify how well our prediction is doing (a model metric)

- For a numeric response, we commonly use squared error loss to evaluate a prediction

$$L(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$$

- Use Root Mean Square Error as a metric across all observations

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{y}_i)} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Quantifying How Well the Model Predicts

Need a way to quantify how well our prediction is doing (a model metric)

- For classification (binary response here), accuracy and log-loss are common:
  - Accuracy

$$\frac{\text{Sum of correct predictions}}{\text{Total number of predictions}}$$

  - Log-loss

$$\sum_{i=1}^{n}(y_i log(\hat{p}_i) + (1 - y_i) log(1 - \hat{p}_i))$$

where $\hat{p}_i$ is the model's estimate of the probability of success ($y = 1$) for that observation

Training vs Test Sets

Ideally we want our model to predict well for observations **it has yet to see**

- Predictions over the observations used to fit or train the model are called the **training (set) error**

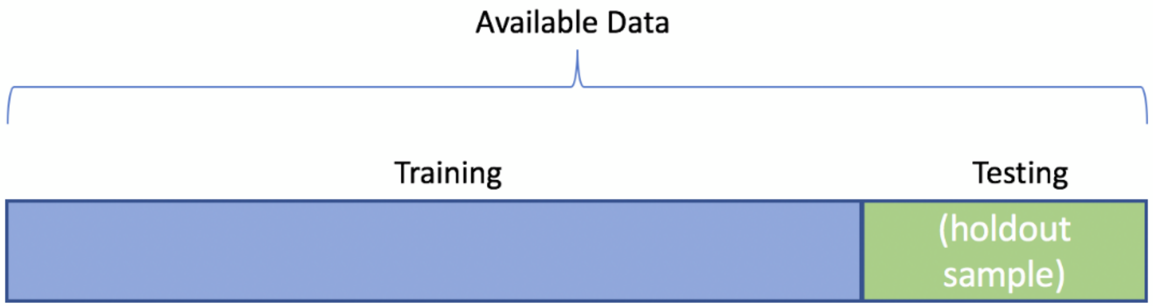
$$\text{Training RMSE} = \sqrt{\frac{1}{\#\text{ of obs used to fit model}} \sum_{\text{obs used to fit model}} (y - \hat{y})^2}$$

- If we only consider this, we'll have no idea how the model will fare on data it hasn't seen!

Training vs Test Sets

One method is to split the data into a **training set** and **test set**

- On the training set we can fit (or train) our models
- We can then predict for the test set observations and judge effectiveness with RMSE



Issues with Training vs Test Sets

Why may we not want to just do a basic training/test set?

- If we don't have much data, we aren't using it all when fitting the models
- Data is randomly split into training/test
- Instead, we could consider splitting the data multiple ways and averaging the test error over the results!

Cross-Validation Idea

k fold Cross-Validation (CV)

- Split data into k folds
- Train model on first k-1 folds, find test error on kth fold
- Train model on first k-2 folds and kth fold, find test error on (k-1)st fold
- …

Find CV error by combining test errors appropriately

- Key = no predictions used in the RMSE were done on data used to train that model!
- Once a best model is chosen, model is refit on entire data set

May Use Both Training/Test & CV

- Recall: LASSO model is similar to an MLR model but shrinks coefficients and may set some to 0
    - Tuning parameter must be chosen (usually by CV)
- Training/Test split gives us a way to validate our model's performance
    - CV can be used on the training set to select **tuning parameters**
    - Helps determine the 'best' model for a class of models
- With many competing model types, compare best models on test set via our metric

Plan

- Learn a few more supervised learning methods
- Implement in `tidymodels`!