https://cnvrg.io/data-science-platform/dataset-management/

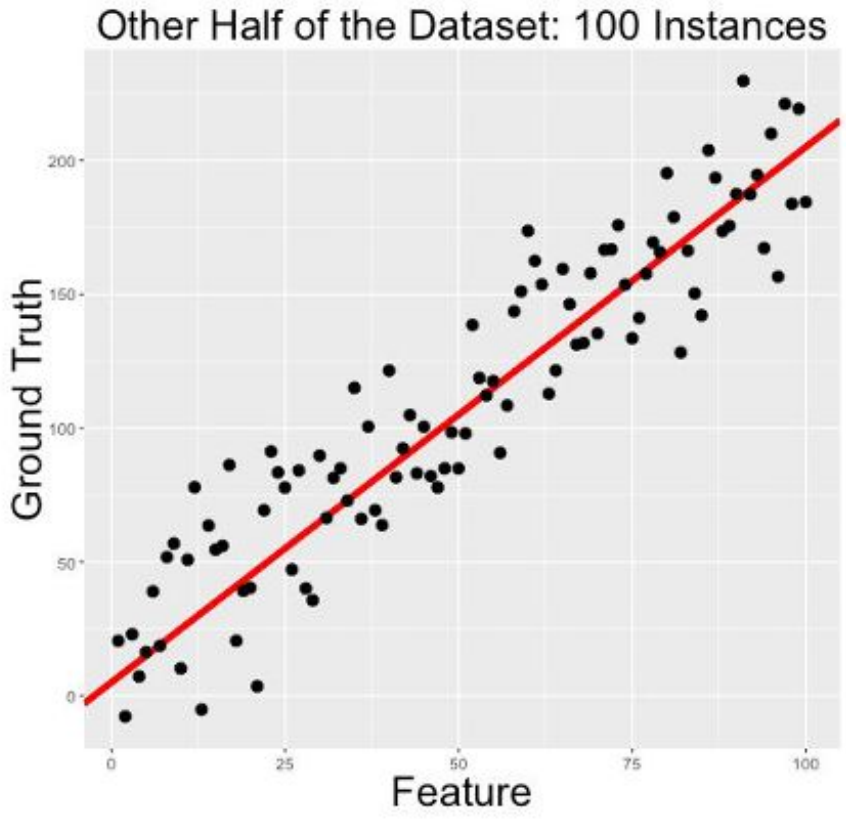
Training, Validation, and Test Data Sets

PhD. Msc. David C. Baldears S.
PhD(s). Msc. Diego Lopez Bernal

TC3007C

Generalization for the Model

- Divide the dataset
    - Look at our performance using the best linear model.
        - Training RMSE = 22.24
        - Test data RMSE = 21.98
        - Pretty similar.

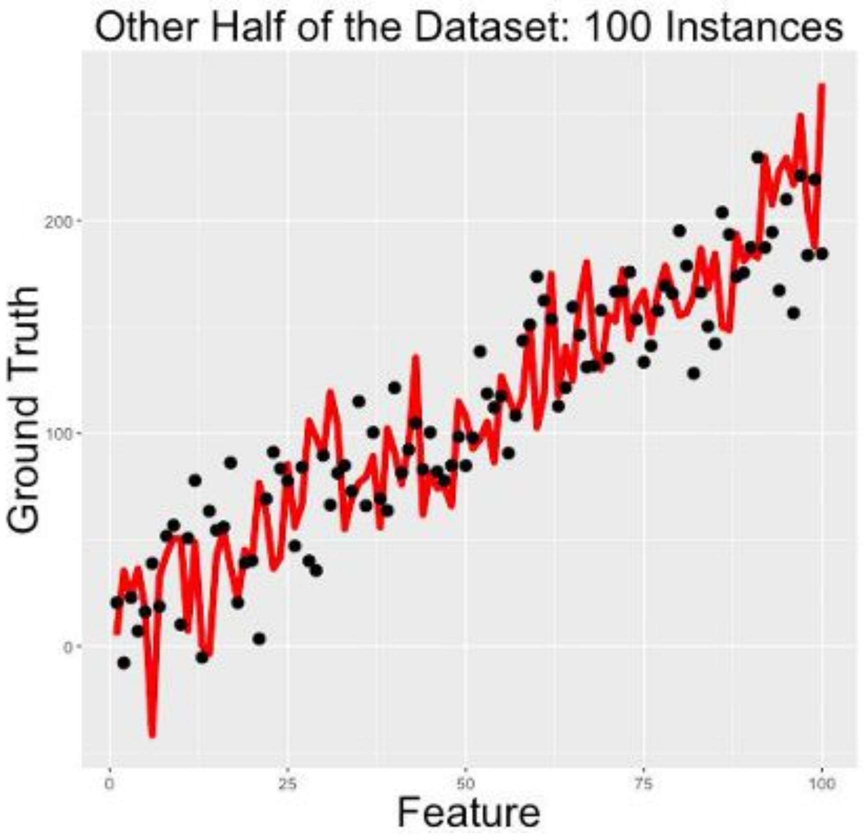


Generalization for the Model

- Divide the dataset
  - Look at the performance using the non-linear model.
    - Old RMSE = 21.44
    - New RMSE = 22.74
    - Still pretty similar.

Generalization for the Model

- Divide the dataset
  - “Perfect” Model
    - Training data RMSE = 0
    - Test data RMSE = 32
    - This did not generalize to new data!
    - This is called overfitting.



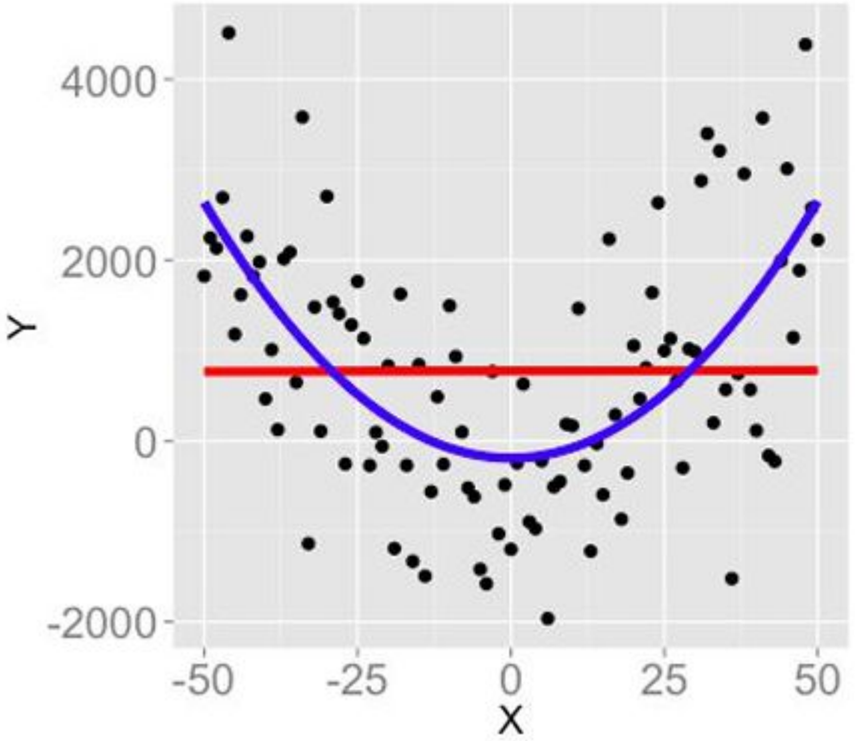
Model Complexity

Our goal is to determine what model complexity is most appropriate.

It must Generalize to predict correctly unseen data

What to do?

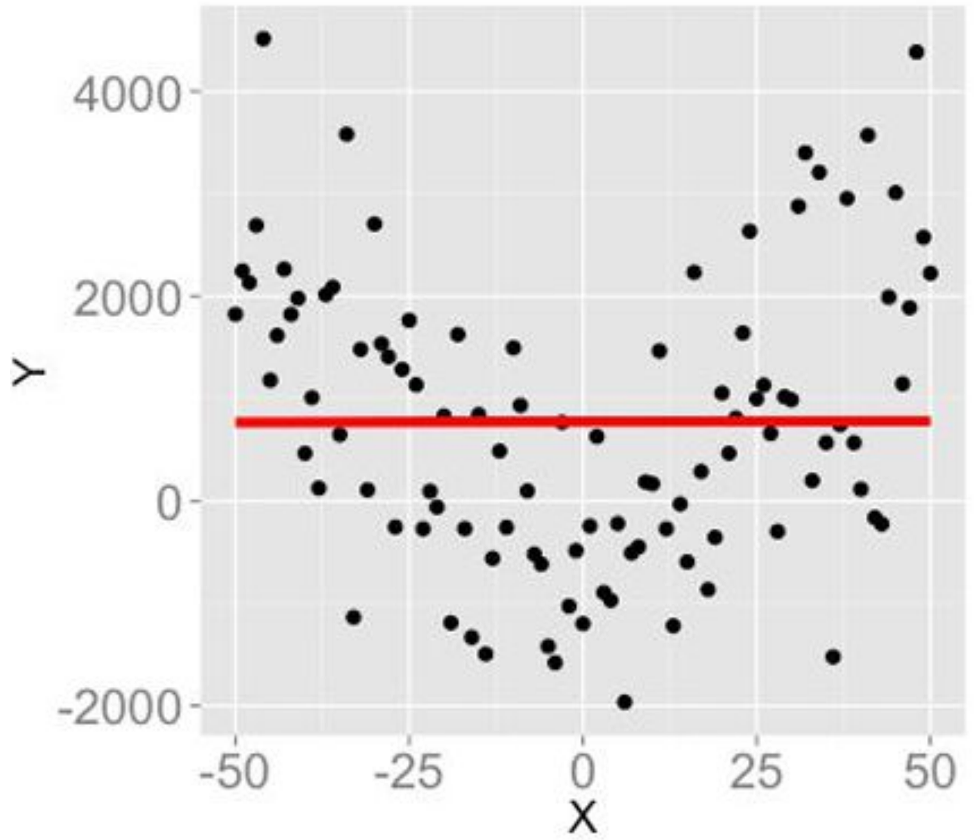
- If we make a very complex model then can perfectly (or near perfectly) fit the training data we just memorize versus the goal of generalizing.
- Remember our goal is to build a system to deal with new data!



An Underfit Model

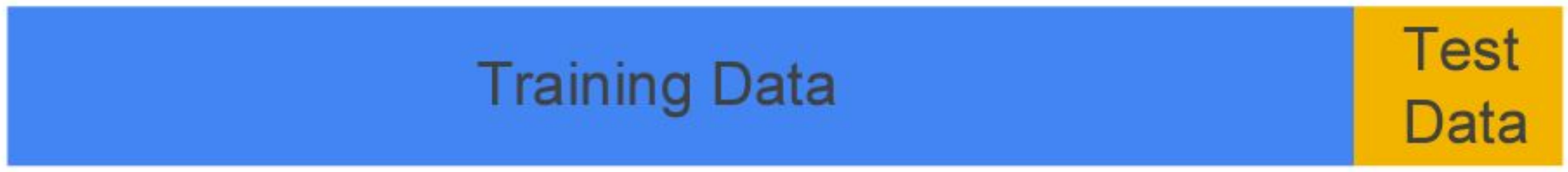
Underfitting happens when you try to use a model that is too simple.

- How can we define how complex a learning model is?
- How can we measure how well our model generalizes?

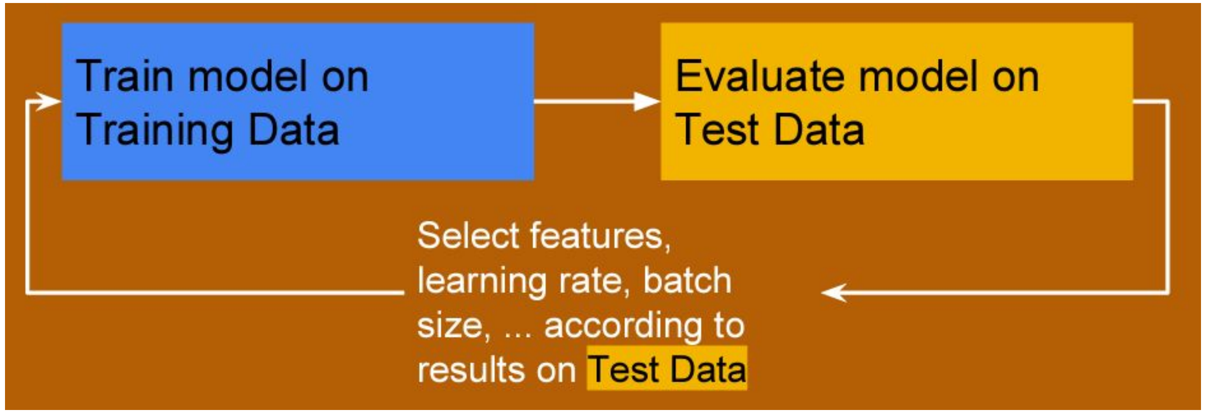


Partitioning Data Sets

- Set aside some of the data as test data
  - Often just do at random
  - Sometimes use most recent data as test data



  - You need to be very careful here



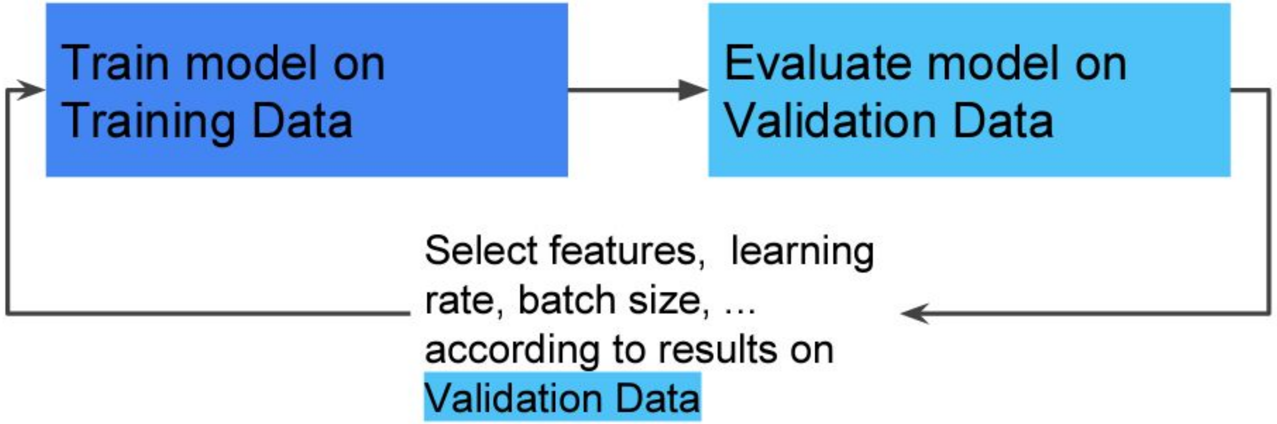
How do we know if our model is good?

- In practice to determine if our model do well on a new sample of data we use a new sample of data that we call the test set
- Good performance on the test set is a useful indicator of good performance on the new data as long as:
    - The test set is large enough
    - The test set is independent of the training set so it truly represents new data
    - Don’t cheat by using the test set more than once

Hence, the test set cannot be used as part of the training

A Solution to “Polluting” Test Data

- Divide the data provided for training our model into two datasets
  - Most of it will be in our Training Data
  - A portion of it (typically 5-10%) will be used as a Validation Data.
- The rest of the data is still used for testing



Pick model that does best on Validation Data
Check for generalization ability on Test Data

Small Validation Set

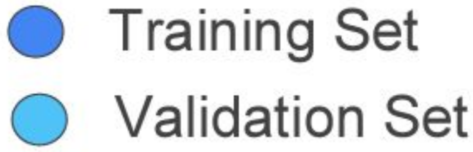
- If your training dataset was very small (say, 100 examples), and you use 5% of it for your validation set,
  - how big would your validation set be?
  - Can you think of any issues with using a validation set of this size?

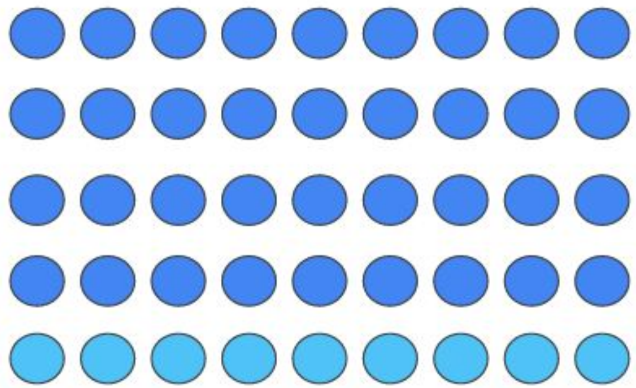
K-fold Cross Validation

- Test data must be set aside unless there is no concern about overfitting
- What if we don't have enough data to set aside enough for validation data?
- For these cases k-fold cross validation is often used
- Basic idea is to divide the data into k roughly even size pieces and in each of k training phases uses 1 piece as validation and the other k-1 as training data

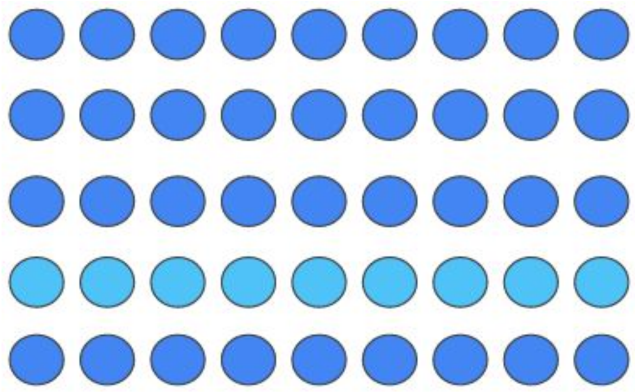
K-fold Cross Validation

- Demonstrate with k=5 where each row represents ⅕ of the data.

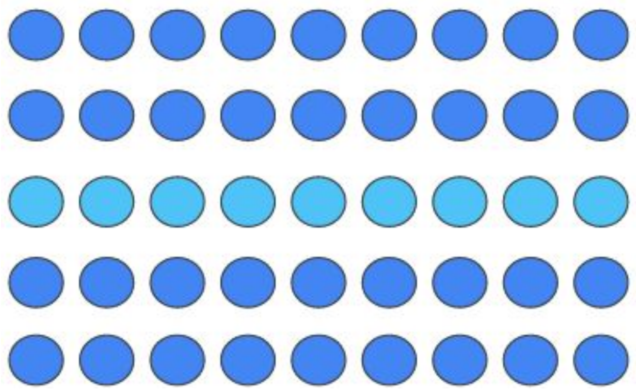




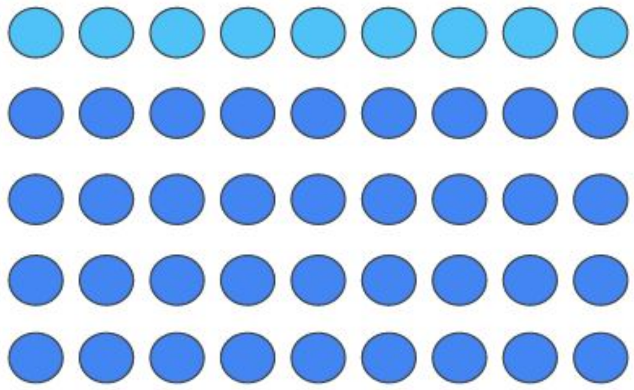
Performance measure m_1 computed just on the validation set for this fold



Performance measure m_2 computed just on the validation set for this fold



Performance measure m_3 computed just on the validation set for this fold



Performance measure m_5 computed just on the validation set for this fold

Cross-Validation: Compute Metric

- For each of the k training phases, compute the performance metric (over the validation set for that phase). This gives us $m_1, m_2, \ldots, m_k$
- Average $m_1, m_2, \ldots, m_k$ to get an aggregate performance metric.
- You can also check model stability by comparing the performance across the k runs and also compute standard statistical measures such as standard deviation and error bars over the k folds.

Cross-Validation: Train Final Model

- To train the final model, choose the hyperparameter setting that gives you best aggregated performance over the k runs.
- Now run the algorithm with the chosen hyperparameters using all examples (other than those set aside as test data throughout) as the training data to obtain the final model.
- Use the test data, which has not been used during cross validation to check for any issues with overfitting.

k-fold Cross-Validation: Pros and Cons

- The advantage of k-fold cross validation is that we can train on (k-1)/k of the data each phase and that all points are used for validation.
- The disadvantage is that k different models need to be trained for every set of hyperparameters being considered, which is slow.
- Only use k-fold cross-validation if you don't have enough labeled data to split into independent train, validate and test sets.