Machine Learning

Training & Testing

Conclusion

Review

• Training Set / Testing Set (60/40)
  – In sample
  – Out of sample

• Validation Metrics (training and testing & different models)
  – Root Mean Square Error (RMSE)
  – Mean Absolute Error (MAE) (fixes large residuals)
  – Cross Validation
  – Correlation

• Over-fitting, Under-fitting
  – KNN (k small : over-fitting, k large : under-fitting)
  – Parametric Models (Linear → Polynomials of degree)
  – Compare to In-Sample & Out-of-Sample and over/under fitting

Review: Learning Models

• Parametric Models (Linear to Polynomials)
• KNN
• Decision Tree
  – Node Splitting
    • Which feature to split on?
      – Classification maximize: information gain (concepts random, entropy, information gain, Gini impurity)
      – Regression: Variance reduction, correlation, mean square error (rmse), mean absolute error (mae)
    • Which value to split on? Median, average, random
  – Depth of tree
  – Aggregating Leaves (to control complexity) return average prediction.

• Ensemble Learners
  – Same Class of Learners
  – Different Class of Learners.

• Bootstrap Aggregating
  – Manipulate training sample (generate it randomly)
  – Ada Boost

• Random Forest (Breiman, Cutler)
  – Ensemble of random decision trees grown using bagging.
  – Idea: A single tree in the forest is weaker than a full decision tree, but by putting them all together, we get better overall performance thanks to diversity.
  – Motivation: overcome the over-fitting problem of single full decision trees.

• Randomness comes in two (sometimes three).
  – Bagging creating training data.
  – Feature selection at node:
    • Best features among a random (smaller) subset of random features
  – Value selection/threshold
    • Randomly select value to split on

• Project a variation (simplified) of Random Forest.
  – Example. Subset of features is just 1.
• Thinking: Which model to use for what type of data?
  – What does the data look like?
• Adding Time Series how do the assumption change?

Project 2. Highlights

Overview (see “NEW”):
  New: All of the decision tree learners you create should be implemented using a matrix data representation (numpy ndarray).

• DT Learner
  – BuildTree()
  – addEvidence() \(\rightarrow\) calls BuildTree()
  – SelectSplitFeature()
    • Best feature to split on
    • Absolute value correlation with Y.

• RTLearner
  – BuildTree()
  – addEvidence()
  – SelectSplitFeature() \(\rightarrow\) only one different from DT
    • Randomly selects a feature


• BagLearner()
• InsaneLearner()
  – Bag of “BagLearners” (LinearRegression)
  – 20 BagLearner instances where:
    • each instance is composed of 20 LinRegLearner instances.