Machine Learning & Predictive Analytics

Class 3

Arnab Bose, Ph.D.

MSc Analytics

University of Chicago

Historical Data Challenges – Selection Bias

Selection Bias

- Select a sample – however the sample selects certain characteristics
- Example – confirmation bias – select data samples that confirm pre-existing hypothesis

- Detect
  - Distribution is close to population distribution – especially important for unbalanced population
- Correct
  - Random sample – for example in batch stochastic gradient descent, it is imperative that mini-batches are random so that subsequent gradient estimates to be independent

Historical Data Challenges – Missing Data

Missing Data

- To Impute or Not to Impute – depends on the dataset size and missing %
- Imputations
    - Missing Completely At Random (MCAR) – no pattern in missing data
    - Missing At Random (MAR) – pattern between missing variable and another variable
    - Missing Not At Random (MNAR) – pattern in probability of missing based on data value

https://pypi.org/project/autoimpute/

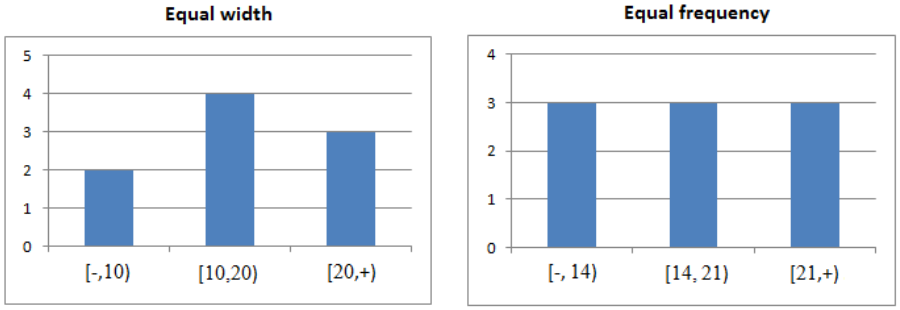
Feature Engineering

1. Presumably redundant variables can help with classification
2. **Perfectly** correlated variables are redundant
3. Imperfectly correlated variables can be useful and complementary
4. Variable that is not useful for modeling by itself can be useful in combination with another variable
5. Categorical variables (non-ordinal) – one-hot encoding
6. Categorical variables (ordinal) – label encoding

http://www.jmlr.org/papers/volume3/guyon03a/guyon03a.pdf

Binning

1. Convert a continuous feature into a categorical feature.
2. Define a series of ranges called *bins.*
3. Good way to handle outliers – very high or very low values simply end up in the highest or lowest bin.
4. Low # of bins – lose a lot of info but have a large # of instances in each bin.
5. High # of bins – closer to original data distribution but some bins may have very few instances.
6. Equal-width binning – simple and intuitive but as the distribution of values in the continuous feature moves away from a uniform distribution, some bins may have few instances.
7. Equal-frequency binning – accurately models the heavily range of the values of the continuous feature but resulting bins may appear non-intuitive.



Resampling and Data Validation

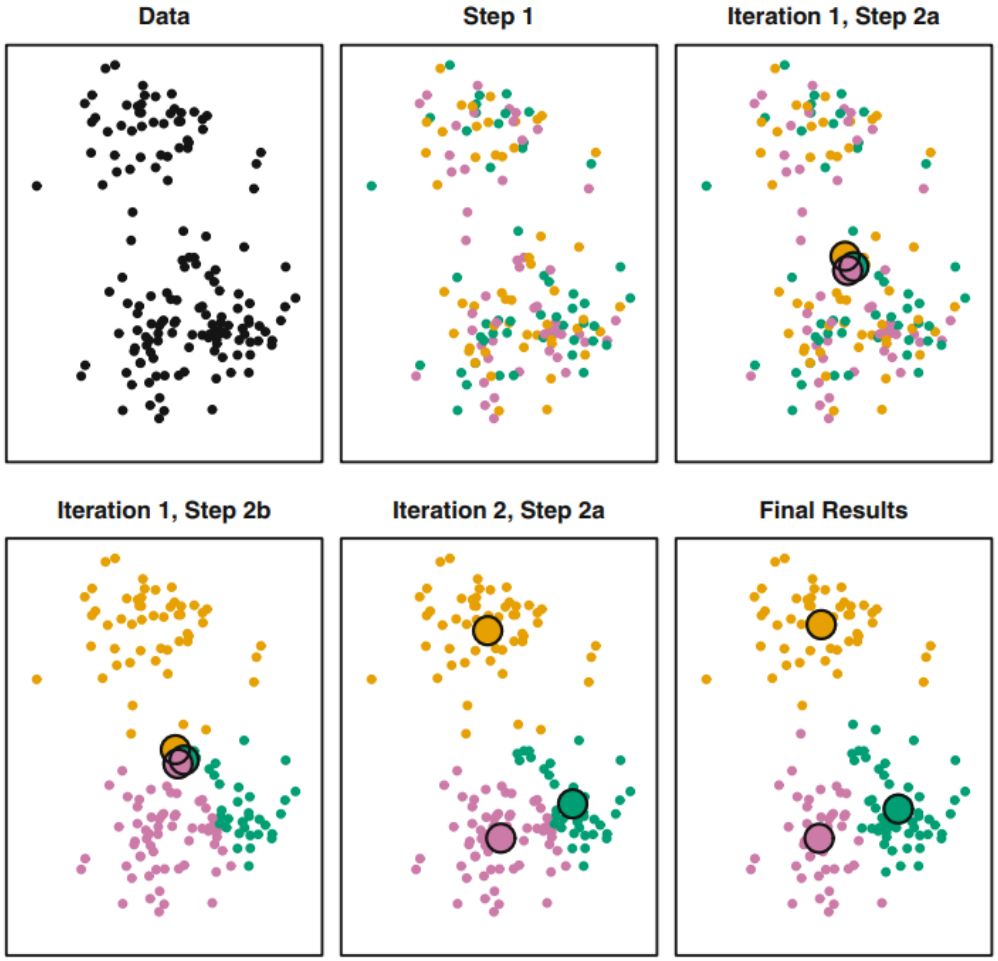
https://lagunita.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/cv_boot.pdf

(also available in your reading material)

Similarity Based Learning

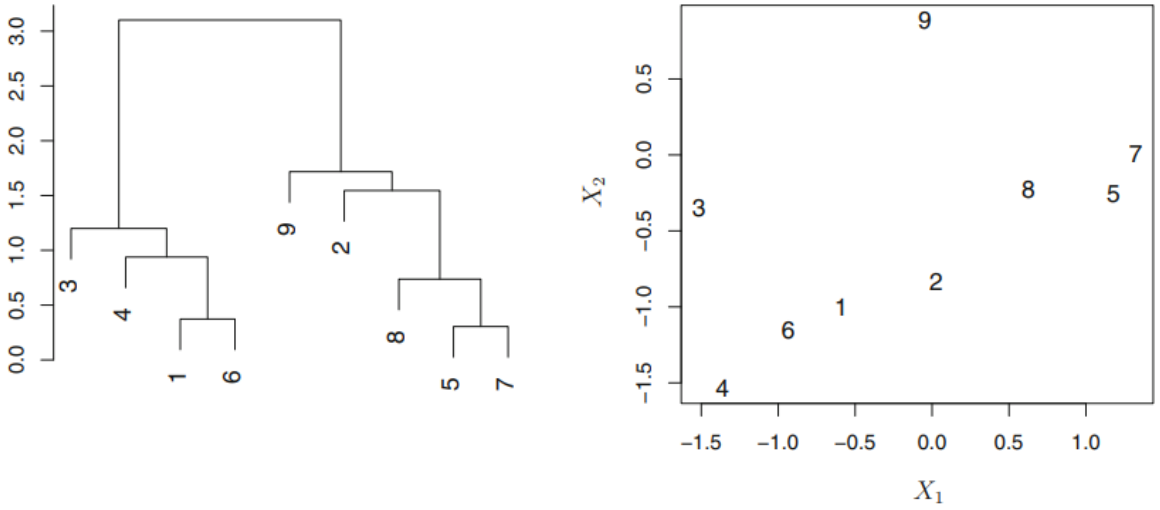
- Find *similar* groups of data
- Requirements:
  - A **feature space representation** of the data
  - A measure of **similarity** using a **metric** that conform the 4 criteria for 2 instances *a, b*:
    1. Non-negativity: $metric(a, b) \geq 0$
    2. Identity: $metric(a, b) = 0 \Leftrightarrow a = b$
    3. Symmetry: $metric(a, b) = metric(b, a)$
    4. Triangular inequality: $metric(a, b) \leq metric(a, c) + metric(b, c)$

Unsupervised Learning with *K-means* Clustering



G.James, D.Witten, T.Hastie, R.Tibshirani, *An Introduction to Statistical Learning*, Springer, 2013.

Hierarchical Clustering



G.James, D.Witten, T.Hastie, R.Tibshirani, *An Introduction to Statistical Learning*, Springer, 2013.

Supervised Learning with Nearest Neighbor

1. Creates a set of local models each defined using a subset of the training dataset.
2. Creates a global prediction model on the full dataset.
3. Decision boundary – a boundary between regions of the feature space in which different target levels will be predicted.
4. Advantage – very simple model and easy to update for new training data.
5. Disadvantage – very sensitive to noise. Does not work well with high dimension data – *curse of high dimensionality*.

K - Nearest Neighbor

1. Use a majority target within the set of *k (number of data points)* nearest neighbors.
2. *k* too low – algorithm sensitive to noise in data and overfit.
3. *k* too high – lose data patterns and underfit.
4. *k* too high problem is acute for imbalanced datasets – the majority target begins to dominate the feature space.
5. one way to address *k* value is to use a weighted *k*-NN – the contribution of each neighbor to the prediction is a function of the inverse distance between the neighbor and query.

https://lagunita.stanford.edu/c4x/HumanitiesScience/StatLearning/asset/unsupervised.pdf

Decision Trees

1. Information-based learning.
2. Utilize a sequence of tests to split the dataset into pure groups with respect to target values.
3. Information gain is a formal metric to calculate the measure of information that got organized due to a dataset split.
4. The learning algo can be considered nonparametric if it is allowed to learn a tree of arbitrary size.
5. Usually learning algos are constrained with tree size to avoid overfitting.
6. Target variable is continuous – regression trees
7. Target variable is discrete – classification trees

Regression Tree

1. Split the predictor space into non-overlapping regions.

2. For every observation that falls into the region R_j the mean of the region becomes the prediction.

3. Goal is to find regions $R_1, \ldots\ldots, R_j$ that minimize the Residual Sum of Squares (RSS)

$$RSS = \sum_{j=1}^{J} \sum_{i \epsilon R_j} \left(y_i - \widehat{y_{R_j}} \right)^2$$

where $\widehat{y_{R_j}}$ is the mean response for the training data in region R_j

Regression Tree

1. Computationally infeasible to consider every possible partition.
2. Top-down greedy approach known as recursive binary splitting.
3. Tree pruning – short sighted to restrict depth of tree.
4. Instead select a subtree that leads to the lowest test error rate.
5. Cost complexity pruning – consider a sequence of trees indexed by a nonnegative tuning parameter α.
6. Goal is to find a subtree$T \subset T_0$ for a given value of α that minimize

$$\sum_{m=1}^{|T|} \sum_{x_i \epsilon R_m} \left(y_i - \widehat{y_{R_m}}\right)^2 + \alpha |T|$$

where |T| indicates the number of terminal nodes of the tree T

R_m is the subset of predictor space corresponding to the *m*th terminal node

$\widehat{y_{R_m}}$ is the mean response for training data in R_m

Classification Tree

1. Predict that each observation belongs to the most commonly occurring class of training observations.

2. Use classification error rate to make the binary splits.

3. Gini index:

$$G = \sum_{k=1}^{K} \widehat{p_{mk}}(1 - \widehat{p_{mk}})$$

Where $\widehat{p_{mk}}$ is the proportion of training observations in the *m*th region that are from the *k*th class

4. Gini index is a measure of total variance across *K* classes and takes a small value if all $\widehat{p_{mk}}$ are close to 0 (very few observations are from the *k*th class) or 1 (almost all observations from same class).

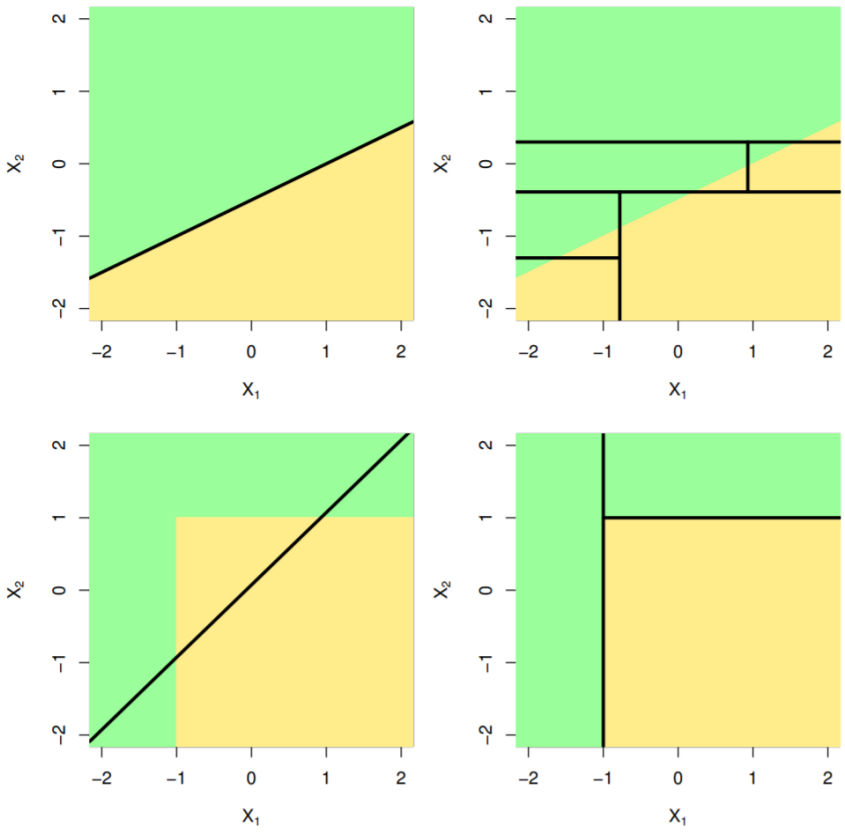
5. Alternatively, entropy:

$$D = -\sum_{k=1}^{K} \widehat{p_{mk}}\ log\ \widehat{p_{mk}}$$

6. Both Gini index and entropy take a small value if the *m*th node is pure.

Tree vs Linear Models

- Which to use – depends on the problem.
- If linear approximation works satisfactorily, then linear model.
- If non-linear/complex relation between features and response then trees.



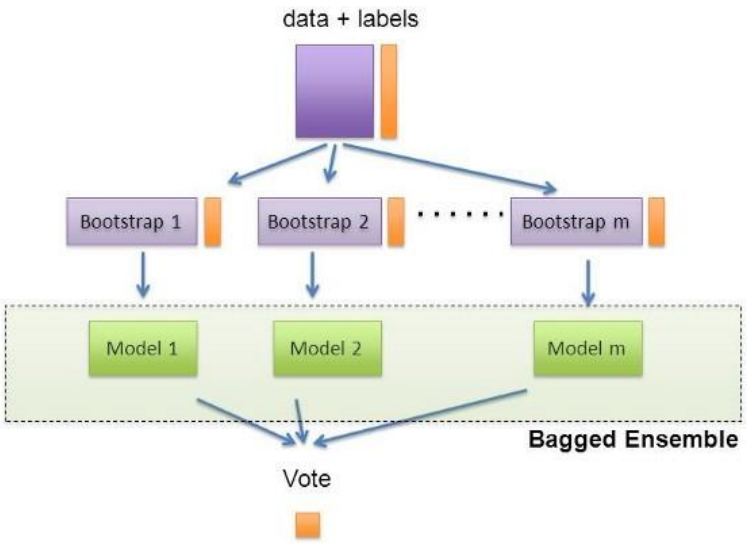
Tree Advantages / Disadvantages

1. ▲ Trees are very easy to explain to people. In fact, they are even easier to explain than linear regression.
2. ▲ Some people believe that decision trees more closely mirror human decision-making than do the regression and classification approaches seen in previous chapters.
3. ▲ Trees can be displayed graphically, and are easily interpreted even by a non-expert (especially if they are small).
4. ▲ Trees can easily handle qualitative predictors without the need to create dummy variables.
5. ▼ Trees suffer from high variance.
6. ▼ Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches.

Bagging

1. Bootstrap aggregation – general purpose procedure for reducing the variance of a model.
2. Fit multiple models to bootstrapped samples from training data set.
3. Applied to trees –
    1. build **B** regression trees using **B** boostrapped training sets and average the resulting predictions – the trees are deep for low bias and high variance.
    2. For classification trees use majority voting for class predicted by the trees.

"Bagging" : **B**ootstrap **AGG**regat**ING**



https://medium.com/@rrfd/boosting-bagging-and-stacking-ensemble-methods-with-sklearn-and-mlens-a455c0c982de

Out-of-Bag (OOB) Data

1. Out-of-bag (OOB) data – on average, each bagged tree uses 2/3rd of the data, leaving the remaining 1/3rd not used for fitting that can be used for prediction.
2. Test error estimation – do not need CV. For a data use the predictions from the trees that do not have the data in their bootstrapped sample. For **B** trees this corresponds to **B**/3 predictions.
3. OOB error – for sufficiently large **B** OOB error is virtually equivalent to leave-one-out CV error.
4. Bagging improves prediction accuracy at the expense of interpretability. But there is some interpretability of effect of a predictor –
    1. Regression – total amount in RSS decrease
    2. Classification – total amount in Gini index decrease

Random Forests

1. Provide an improvement over bagging by decorrelating trees to reduce variance.
2. Each tree uses a random subset of features to split.
3. Using a small value of features to build a random forest is helpful when there are a large number of correlated predictors.
4. Extremely random tree is using random thresholds for each feature instead of searching for the best possible thresholds.
5. Depth of tree controls the number of interaction terms.
6. Categorical variables with different levels bias feature importance [refer to paper below].

https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-8-25

Textbook Chapters and Assignment

- Materials covered available in book:
  - DL: Chapter 5
  - HML: Chapter 2, 6 – 7, 9
  - ISL: Chapters 5, 8
  - DLP: Chapter 4

- Code: https://github.com/ageron/handson-ml2