DATA 8

Fall 2016

Lecture 34, November 16

Implementing a Classifier

Slides created by Ani Adhikari and John DeNero

Announcements

- Project 3 will be released today. Checkpoint Tuesday 11/22, final deadline Tuesday 11/29
- Homework will be assigned on Friday:
  - Early submission: Wed 11/23 (usual schedule)
  - “Regular” submission: Monday 11/28 after the break
- GSI/Tutor office hours locations from now on:
  - Mondays Etcheverry 3106
  - Wednesdays Etcheverry 3108
  - Other days: no change

Classification

- Response variable is categorical; values are **classes**
- **Binary response**: Only two classes, **0 and 1**

- Try to **classify** the response into one of the categories, based on:
  - Values of predictor variables, called **attributes**
  - **Training set** of data in which the classes of the individuals are known

k-Nearest Neighbor Classifier

- New individual, unknown class
- Find the *k* closest individuals in the training set
  - They are the new individual's "*k* nearest neighbors"
- Assign the new individual the same class as the majority of the *k* nearest neighbors (*k* is usually taken to be an odd number)

The Test Set

- Split original training set at random into two sets
- Use one of the sets for training:
  - Explore as much as you want
  - Develop classifier
- Use the other set (**test set**) to compare the classifier's predictions and the true classes

Rows of Tables

- Each row contains all the data for one individual
- `tbl.row(i)` evaluates to `i`th row of `tbl`
- `tbl.row(i).item(j)` is item indexed `j` of row `i`
- Type: “row object”; not all elements are of same type
- If all elements are of the same type (e.g. all numerical), then `np.array(my_row)` converts `my_row` to an array
- `tbl.apply(function_name)` applies the function to each row of `tbl`; each entire row is passed to `function_name`

(Demo)

Distance Between Two Points

- Two attributes *x* and *y*:

$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}.$$

- Three attributes *x*, *y*, and *z*:

$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2}$$

  - and so on ...

Finding the *k* Nearest Neighbors

To find the *k* nearest neighbors of a point:

- Find the distance between the point and each point in the training set
- Augment the training data table with a column containing all the distances
- Sort the augmented table in increasing order of the distances
- Take the top *k* rows of the sorted table

(Demo)

The Classifier

To classify a point:

- Find its *k* nearest neighbors
- Take a majority vote of the *k* nearest neighbors to see which of the two classes appears more often
- Assign the point the class that wins the majority vote

Assessing Accuracy

- Separate the data at random into a training set and a test set
- Use the training set to classify each point in the test set
- Find the fraction of points for which the classification is correct