DSC 10, Spring 2018 Lecture 26

Classification II

sites.google.com/eng.ucsd.edu/dsc-10-spring-2018

Credit: Anindita Adhikari and John DeNero

Announcements

- Lab 10 due Wednesday
- Project 10 due Saturday
  - Please re-click download link to get updated tests
- Please fill out CAPE
  - This is a new major and your feedback matters!

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

The Classifier (The Big Picture)

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 class that wins the majority vote

Finding the *k* Nearest Neighbors

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

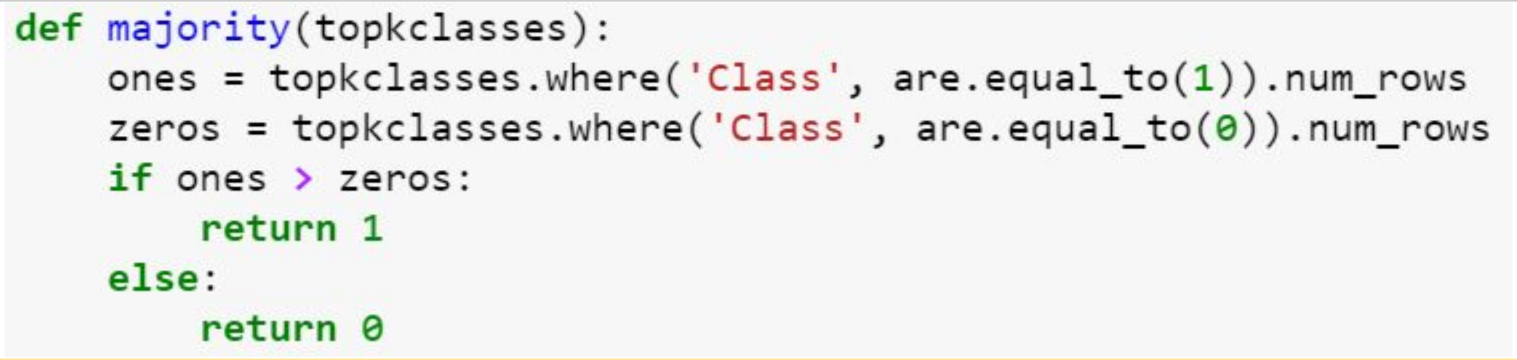
- Find the distance between the new point and each point in the training set
- Augment the training data table with a column containing all the distances
- Sort the augmented table in increasing order of the distances
- Take the top *k* rows of the sorted table

Taking a Majority Vote

To find the class to assign the new point:

- Find the majority of the top k rows’ classes
- Assign this class to the new point

Discussion Question



```
def majority(topkclasses):
    ones = topkclasses.where('Class', are.equal_to(1)).num_rows
    zeros = topkclasses.where('Class', are.equal_to(0)).num_rows
    if ones > zeros:
        return 1
    else:
        return 0
```

How could you implement the majority function in one line of code?

A. return topkclasses.group('Class', max).sort('Class', descending=True).row(0).item(1)

B. return topkclasses.group('Class', max).sort('Class', descending=False).row(0).item(0)

C. return topkclasses.group('Class').sort('count', descending=True).row(0).item(0)

D. return topkclasses.group('Class').sort('count', descending=False).row(0).item(0)

Measuring Accuracy

Accuracy of Classifier

- What fraction of individuals does it classify correctly?

- Need to compare:
  - Classifier’s predictions
  - True classes of individuals

- For this, need to know the true classes. But we only know those for the training set. So now what?

The Test Set

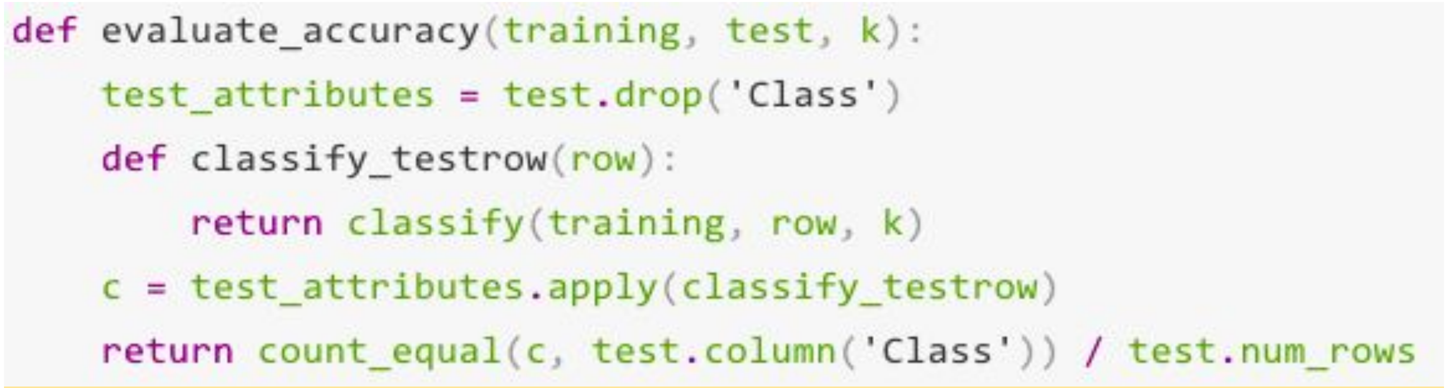
- Split original training set at random into two sets

- 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

(Demo)

Discussion Question

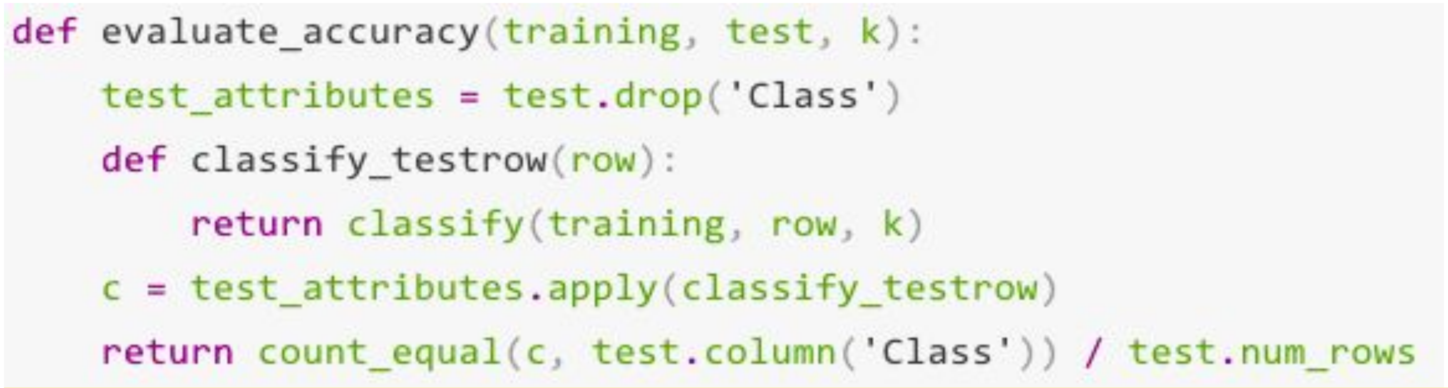


```
def evaluate_accuracy(training, test, k):
    test_attributes = test.drop('Class')
    def classify_testrow(row):
        return classify(training, row, k)
    c = test_attributes.apply(classify_testrow)
    return count_equal(c, test.column('Class')) / test.num_rows
```

What is the type of the `test_attribute` variable?

A. Number
B. Array
C. Table
D. Row
E. List

Discussion Question

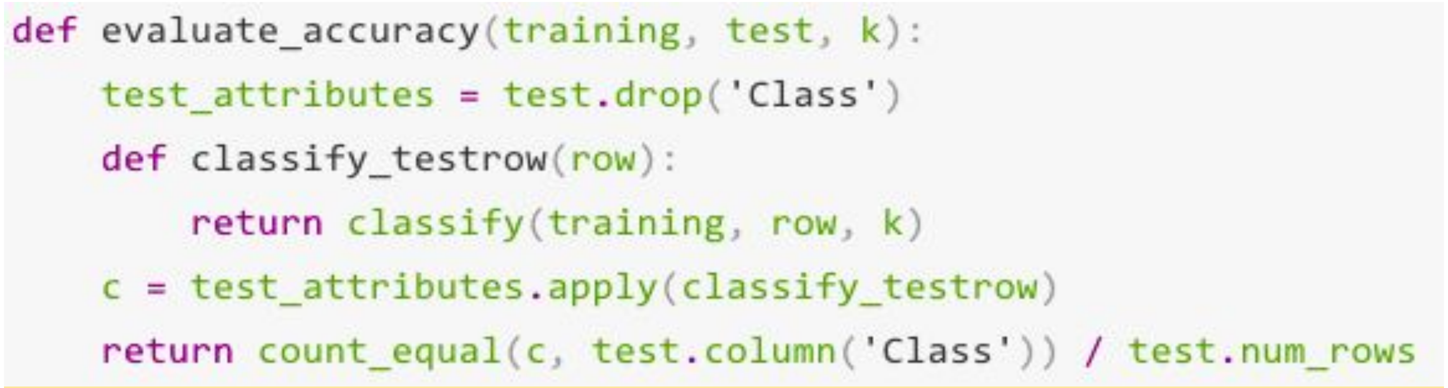


```python
def evaluate_accuracy(training, test, k):
    test_attributes = test.drop('Class')
    def classify_testrow(row):
        return classify(training, row, k)
    c = test_attributes.apply(classify_testrow)
    return count_equal(c, test.column('Class')) / test.num_rows
```

What is the purpose and return type of the `classify_testrow` function?

A. Predicts a class for one row, returns a number
B. Predicts a class for the table, returns an array
C. Predicts a class for one row, returns an array
D. Predicts a class for the table, returns a number
E. None of the above

Discussion Question

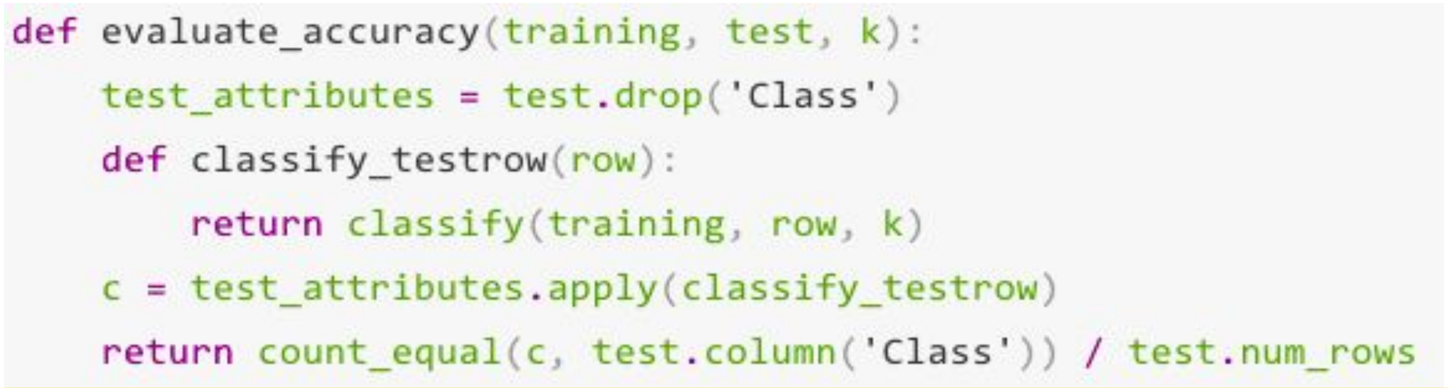


```
def evaluate_accuracy(training, test, k):
    test_attributes = test.drop('Class')
    def classify_testrow(row):
        return classify(training, row, k)
    c = test_attributes.apply(classify_testrow)
    return count_equal(c, test.column('Class')) / test.num_rows
```

What is the type of the variable named `c`?

A. Number
B. Array
C. Table
D. Row
E. None of the above

Discussion Question



```
def evaluate_accuracy(training, test, k):
    test_attributes = test.drop('Class')
    def classify_testrow(row):
        return classify(training, row, k)
    c = test_attributes.apply(classify_testrow)
    return count_equal(c, test.column('Class')) / test.num_rows
```

How does this function measure accuracy?

A. The number of 1’s in the column(‘Class’)
B. The number of 0’s in the column(‘Class’)
C. The number of rows where actual and predicted values are the same
D. The proportion of rows where actual and predicted values are the same
E. None of the above

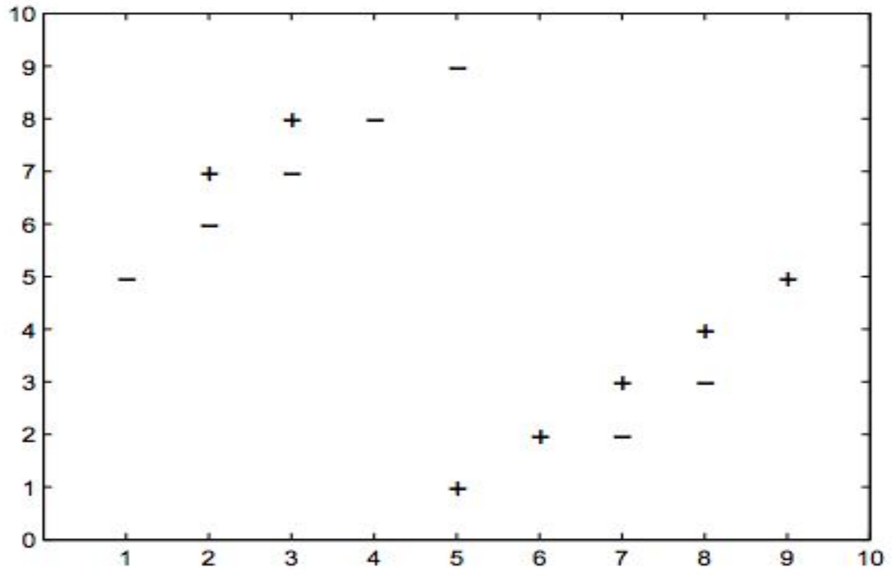
(Demo)

Discussion Question

Suppose you want to test your classifier using the training set. One point becomes a *test* point and everything else is *training*. Then you repeat until each point has been the unlabeled test point once.

What value of k will give us the largest error (number of misclassified labels)?

A. 0
B. 1
C. 5
D. 13



Discussion Question

When we run a computer program, we'd like it to run as fast as possible. k-NN algorithm has two stages: *training* and *testing*.

Which stage will take longer to run: training or testing?

A. Training
B. Testing
C. Same time for both
D. Depends on the problem