SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Established in collaboration with MIT

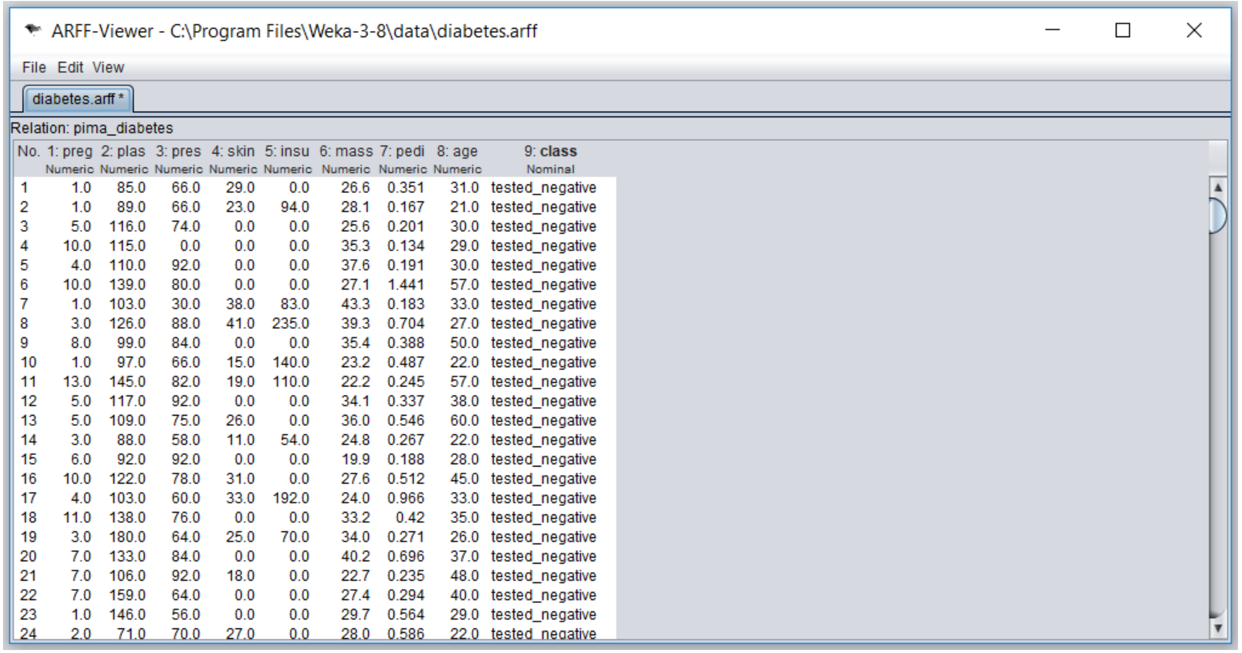
Classification using Weka & Scikit-learn

PROF. K. H. LIM

50.038 Computational Data Science

Load in datasets

- Weka already has some pre-installed datasets
  - Go to Tools → ArffViewer → File → Open
  - Then select dataset ./Weka-3-8/data/diabetes.arff



Relation: pima_diabetes

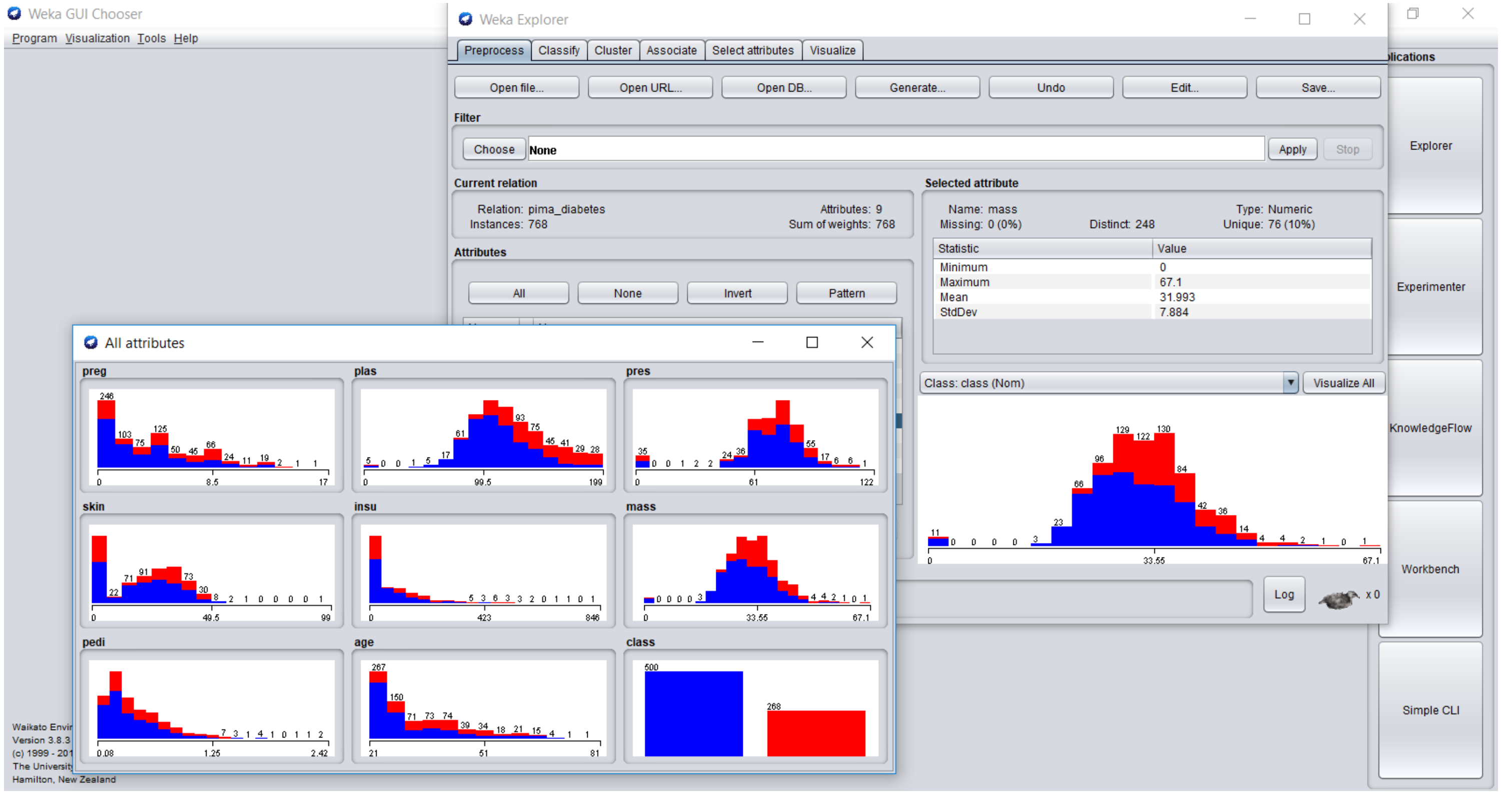
No.	1: preg Numeric	2: plas Numeric	3: pres Numeric	4: skin Numeric	5: insu Numeric	6: mass Numeric	7: pedi Numeric	8: age Numeric	9: **class** Nominal
1	1.0	85.0	66.0	29.0	0.0	26.6	0.351	31.0	tested_negative
2	1.0	89.0	66.0	23.0	94.0	28.1	0.167	21.0	tested_negative
3	5.0	116.0	74.0	0.0	0.0	25.6	0.201	30.0	tested_negative
4	10.0	115.0	0.0	0.0	0.0	35.3	0.134	29.0	tested_negative
5	4.0	110.0	92.0	0.0	0.0	37.6	0.191	30.0	tested_negative
6	10.0	139.0	80.0	0.0	0.0	27.1	1.441	57.0	tested_negative
7	1.0	103.0	30.0	38.0	83.0	43.3	0.183	33.0	tested_negative
8	3.0	126.0	88.0	41.0	235.0	39.3	0.704	27.0	tested_negative
9	8.0	99.0	84.0	0.0	0.0	35.4	0.388	50.0	tested_negative
10	1.0	97.0	66.0	15.0	140.0	23.2	0.487	22.0	tested_negative
11	13.0	145.0	82.0	19.0	110.0	22.2	0.245	57.0	tested_negative
12	5.0	117.0	92.0	0.0	0.0	34.1	0.337	38.0	tested_negative
13	5.0	109.0	75.0	26.0	0.0	36.0	0.546	60.0	tested_negative
14	3.0	88.0	58.0	11.0	54.0	24.8	0.267	22.0	tested_negative
15	6.0	92.0	92.0	0.0	0.0	19.9	0.188	28.0	tested_negative
16	10.0	122.0	78.0	31.0	0.0	27.6	0.512	45.0	tested_negative
17	4.0	103.0	60.0	33.0	192.0	24.0	0.966	33.0	tested_negative
18	11.0	138.0	76.0	0.0	0.0	33.2	0.42	35.0	tested_negative
19	3.0	180.0	64.0	25.0	70.0	34.0	0.271	26.0	tested_negative
20	7.0	133.0	84.0	0.0	0.0	40.2	0.696	37.0	tested_negative
21	7.0	106.0	92.0	18.0	0.0	22.7	0.235	48.0	tested_negative
22	7.0	159.0	64.0	0.0	0.0	27.4	0.294	40.0	tested_negative
23	1.0	146.0	56.0	0.0	0.0	29.7	0.564	29.0	tested_negative
24	2.0	71.0	70.0	27.0	0.0	28.0	0.586	22.0	tested_negative

Attributes in diabetes.arff

- **Preg**: Number of times pregnant
- **Plas**: Plasma glucose concentration
- **Pres**: Diastolic blood pressure (mm Hg)
- **Skin**: Triceps skin fold thickness (mm)
- **Insu**: 2-Hour serum insulin (mu U/ml)
- **Mass**: Body mass index (weight in kg/(height in m)^2)
- **Pedi**: Diabetes pedigree function
- **Age**: Age (years)
- **Class**: Test results for diabetes

Weka Data Explorer

- Study the diabetes.arff dataset using Weka Explorer



Filter: None

Relation: pima_diabetes
Instances: 768
Attributes: 9
Sum of weights: 768

Name: mass
Missing: 0 (0%)
Distinct: 248
Type: Numeric
Unique: 76 (10%)

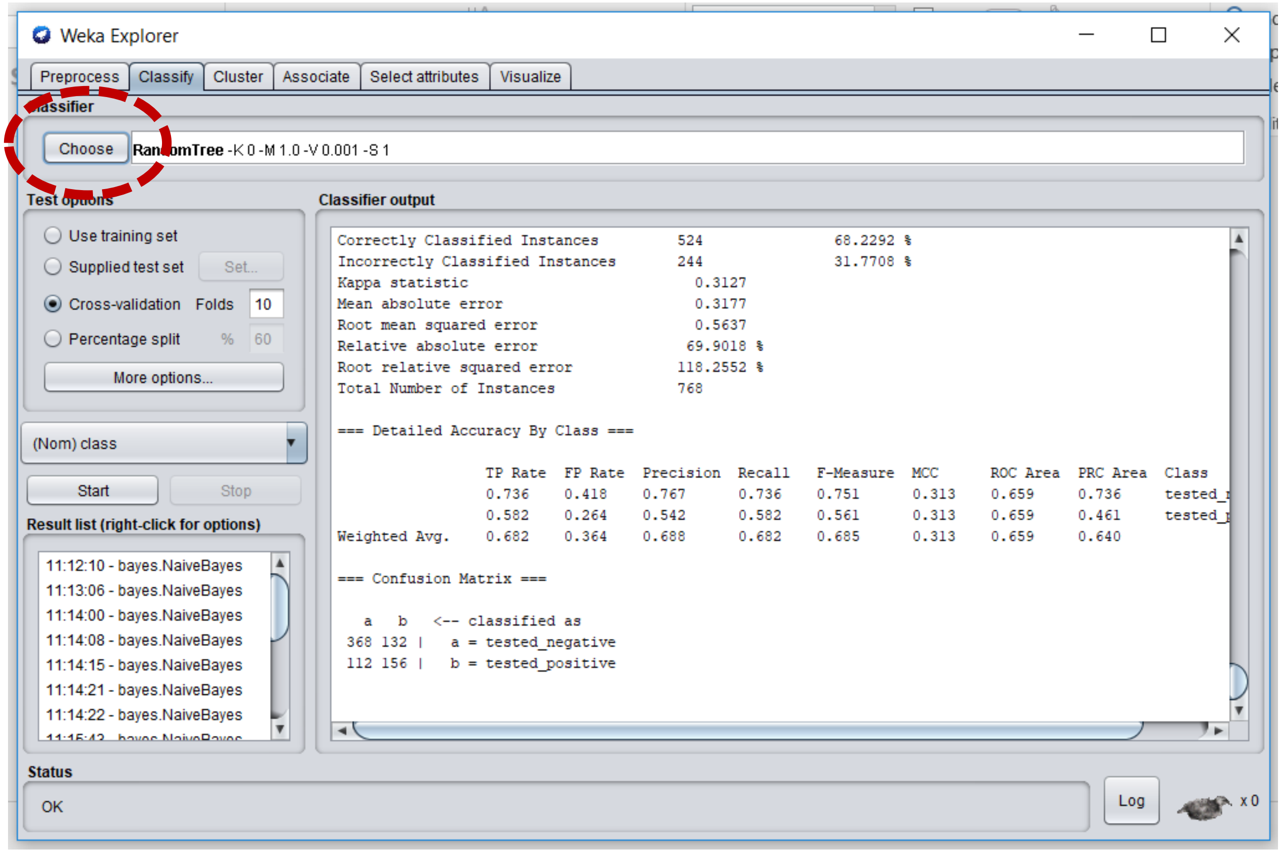
Statistic	Value
Minimum	0
Maximum	67.1
Mean	31.993
StdDev	7.884

Class: class (Nom)

Waikato Envir
Version 3.8.3
(c) 1999 - 201
The Universit
Hamilton, New Zealand

Training and Testing Classifiers

- Select from multiple classifiers (e.g., Naïve Bayes, Decision Trees, Random Forests, etc)



Choose: RandomTree -K 0 -M 1.0 -V 0.001 -S 1

Test options:
- ○ Use training set
- ○ Supplied test set
- ● Cross-validation Folds 10
- ○ Percentage split % 60

(Nom) class

Result list (right-click for options):

```
11:12:10 - bayes.NaiveBayes
11:13:06 - bayes.NaiveBayes
11:14:00 - bayes.NaiveBayes
11:14:08 - bayes.NaiveBayes
11:14:15 - bayes.NaiveBayes
11:14:21 - bayes.NaiveBayes
11:14:22 - bayes.NaiveBayes
11:15:43 - bayes.NaiveBayes
```

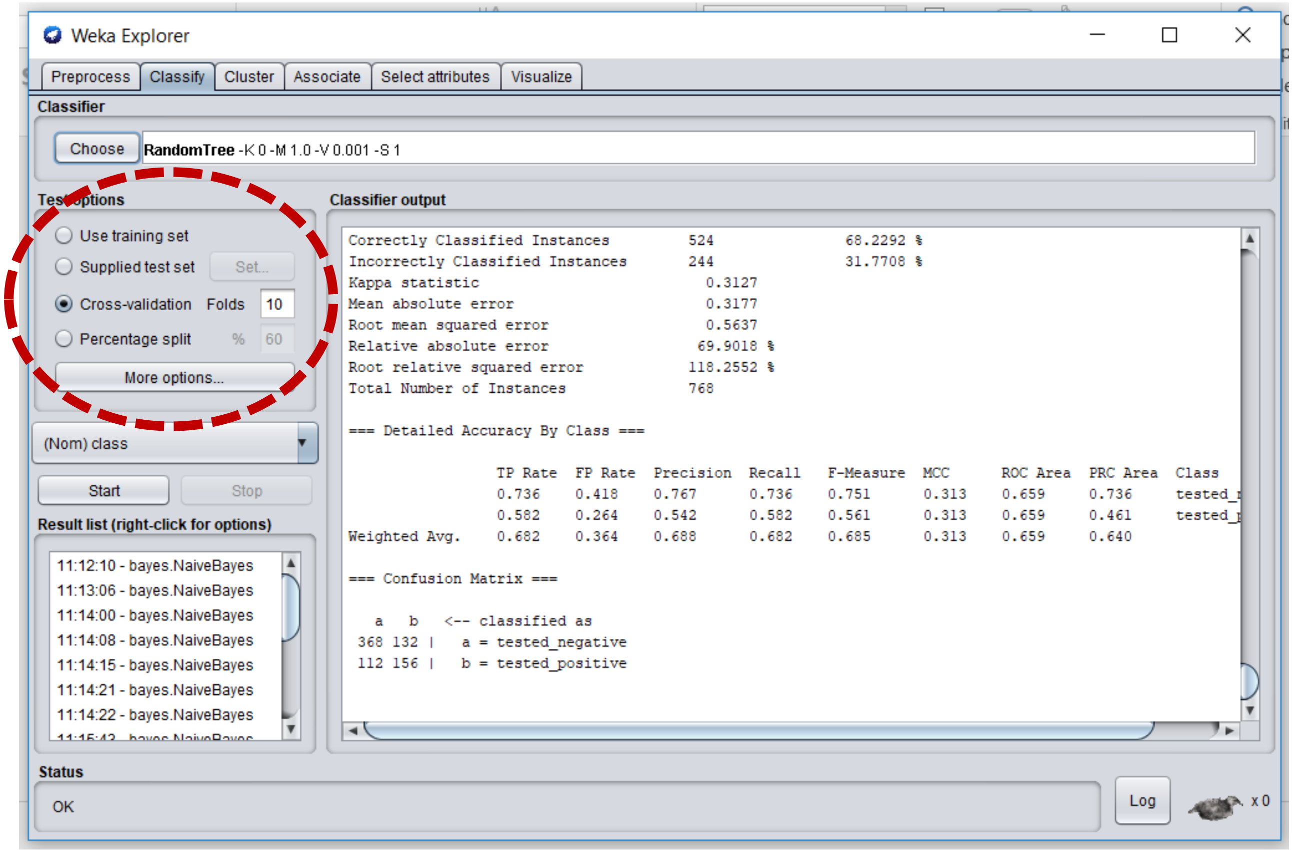
Classifier output:

```
Correctly Classified Instances         524               68.2292 %
Incorrectly Classified Instances       244               31.7708 %
Kappa statistic                          0.3127
Mean absolute error                      0.3177
Root mean squared error                  0.5637
Relative absolute error                 69.9018 %
Root relative squared error            118.2552 %
Total Number of Instances              768

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.736    0.418    0.767      0.736    0.751      0.313    0.659     0.736     tested_
                 0.582    0.264    0.542      0.582    0.561      0.313    0.659     0.461     tested_
Weighted Avg.    0.682    0.364    0.688      0.682    0.685      0.313    0.659     0.640

=== Confusion Matrix ===

   a   b   <-- classified as
 368 132 |   a = tested_negative
 112 156 |   b = tested_positive
```

Status: OK

Training and Testing Classifiers

- Various evaluation approach (e.g., using pre-fixed train/test sets, k-fold cross validation, etc)



Classifier

RandomTree -K 0 -M 1.0 -V 0.001 -S 1

Test options

○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 60

(Nom) class

Result list (right-click for options)

```
11:12:10 - bayes.NaiveBayes
11:13:06 - bayes.NaiveBayes
11:14:00 - bayes.NaiveBayes
11:14:08 - bayes.NaiveBayes
11:14:15 - bayes.NaiveBayes
11:14:21 - bayes.NaiveBayes
11:14:22 - bayes.NaiveBayes
11:15:43 - bayes.NaiveBayes
```

Classifier output

```
Correctly Classified Instances         524               68.2292 %
Incorrectly Classified Instances       244               31.7708 %
Kappa statistic                          0.3127
Mean absolute error                      0.3177
Root mean squared error                  0.5637
Relative absolute error                 69.9018 %
Root relative squared error            118.2552 %
Total Number of Instances              768

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.736    0.418    0.767      0.736    0.751      0.313    0.659     0.736     tested_
                 0.582    0.264    0.542      0.582    0.561      0.313    0.659     0.461     tested_
Weighted Avg.    0.682    0.364    0.688      0.682    0.685      0.313    0.659     0.640

=== Confusion Matrix ===

   a   b   <-- classified as
 368 132 |   a = tested_negative
 112 156 |   b = tested_positive
```

Status

OK

Exercise

1. Load in the diabetes.arff dataset

2. How many features does this dataset have? What will you use as labels?

   8

3. Use Weka explorer, find out: (i) what is the average "mass"; and (ii) what are the minimum and maximum "age"

   31.993

   21 81

4. Run the "RandomTree" and "RandomForest" classifiers using a 10-fold cross validation.
   - What do you observe in terms of their performance and running time?

Python Scikit-learn

- Machine learning library based on Python
- Contains functionalities for data pre-processing, classification, clustering, etc

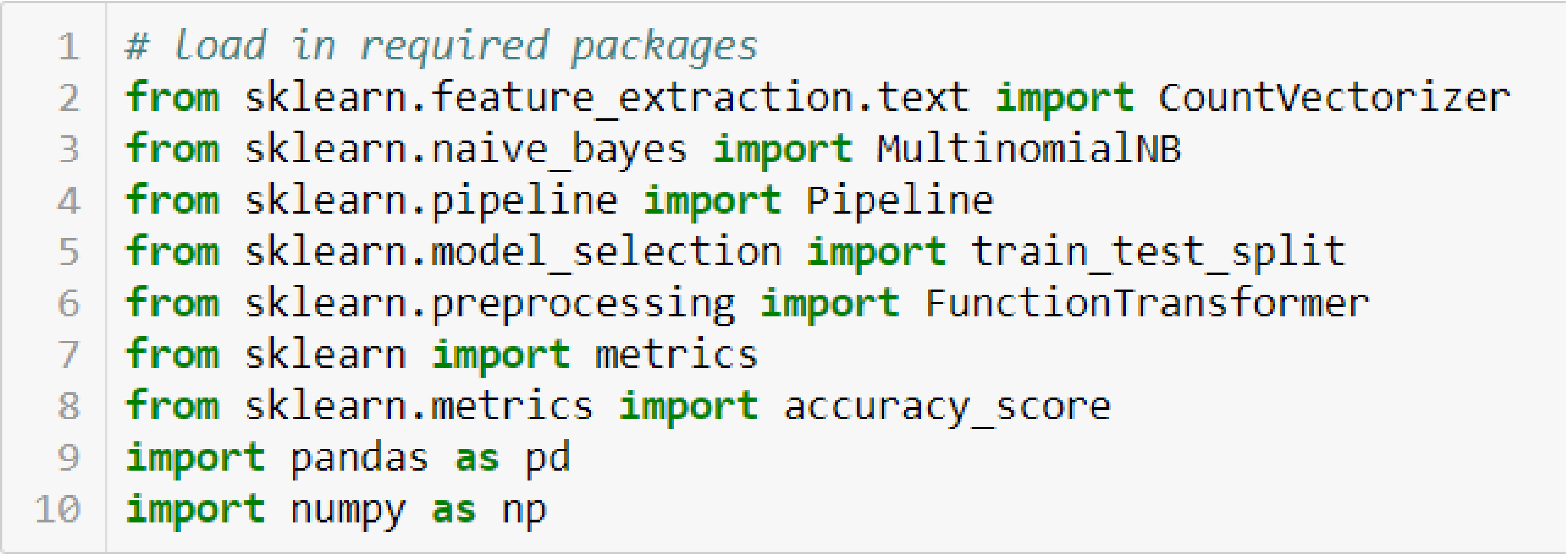
Twitter Dataset

- Using a Twitter sentiment dataset
  - http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip
  - https://docs.google.com/file/d/0B04GJPshIjmPRnZManQwWEdTZjg/edit

- This dataset comprises 1.6M tweets with various columns, we will make use of the first and last column (sentiment label and tweet text)
  - For sentiment, a value of 4 = positive and 0 = negative

Load Packages

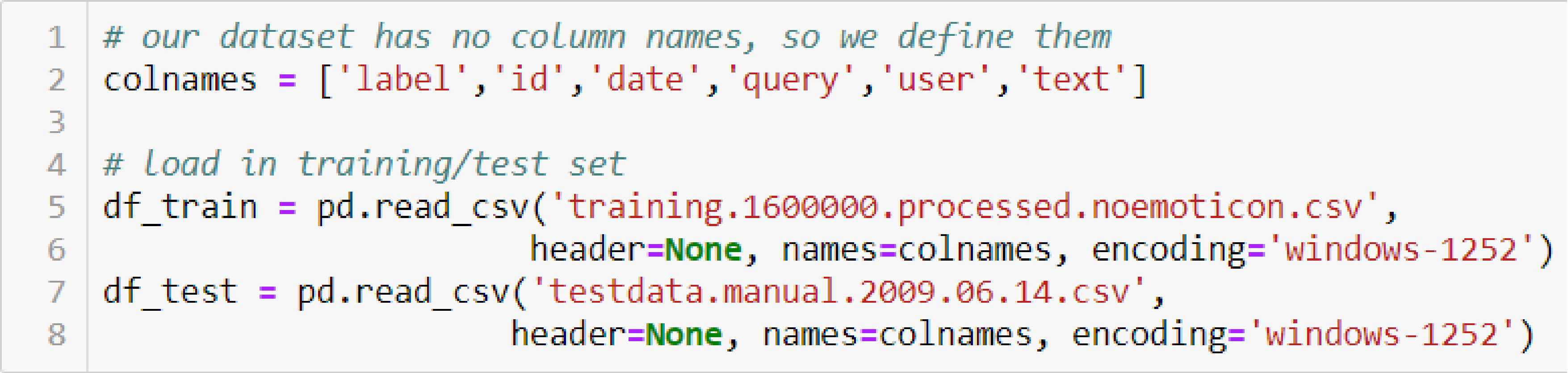
- Import relevant packages



```python
# load in required packages
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import FunctionTransformer
from sklearn import metrics
from sklearn.metrics import accuracy_score
import pandas as pd
import numpy as np
```

Load Dataset

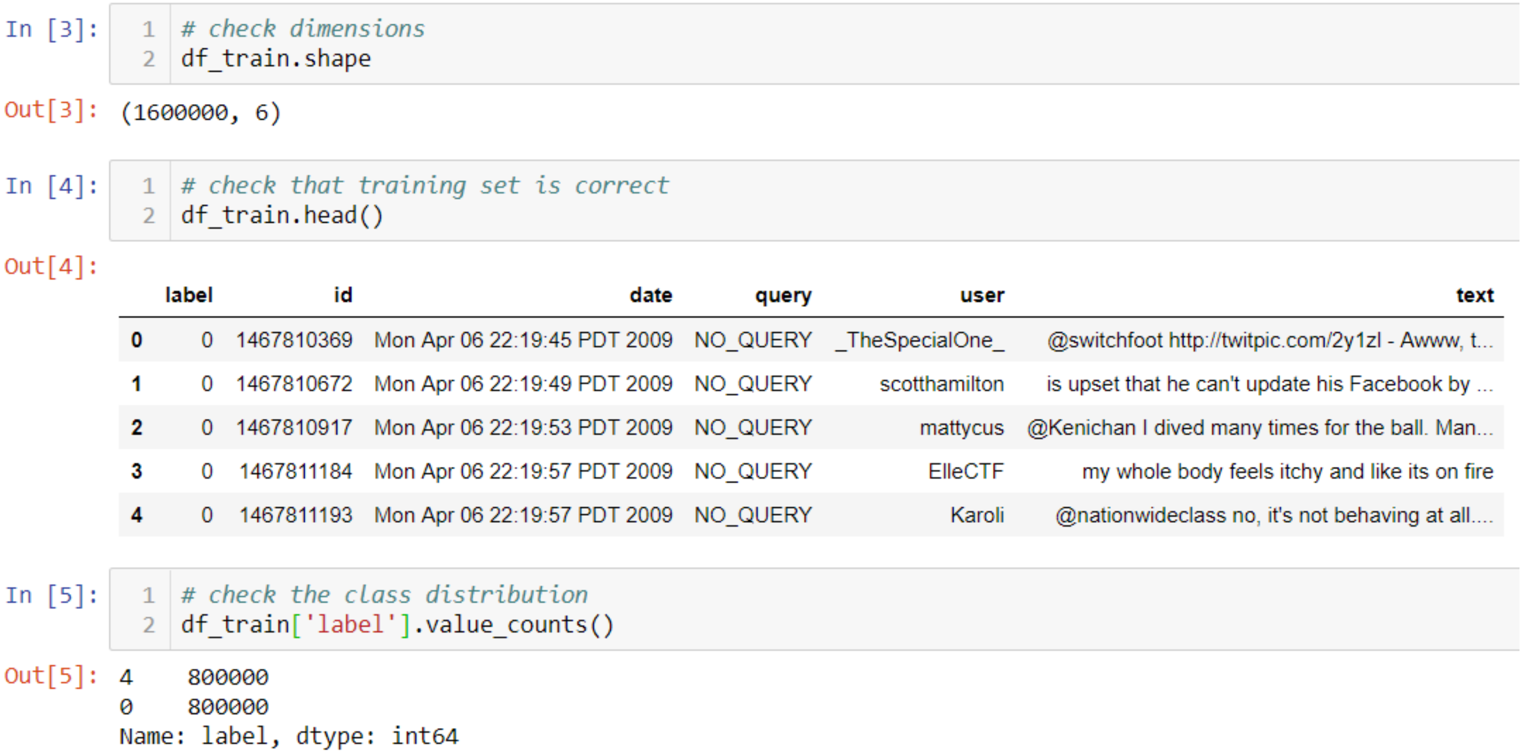
- Does dataset contain headers?
- What type of encoding for text?



```python
# our dataset has no column names, so we define them
colnames = ['label','id','date','query','user','text']

# load in training/test set
df_train = pd.read_csv('training.1600000.processed.noemoticon.csv',
                       header=None, names=colnames, encoding='windows-1252')
df_test = pd.read_csv('testdata.manual.2009.06.14.csv',
                      header=None, names=colnames, encoding='windows-1252')
```

Sanity Check

- Is the dataset how you expected it to be?



```
In [3]:  # check dimensions
         df_train.shape

Out[3]:  (1600000, 6)
```

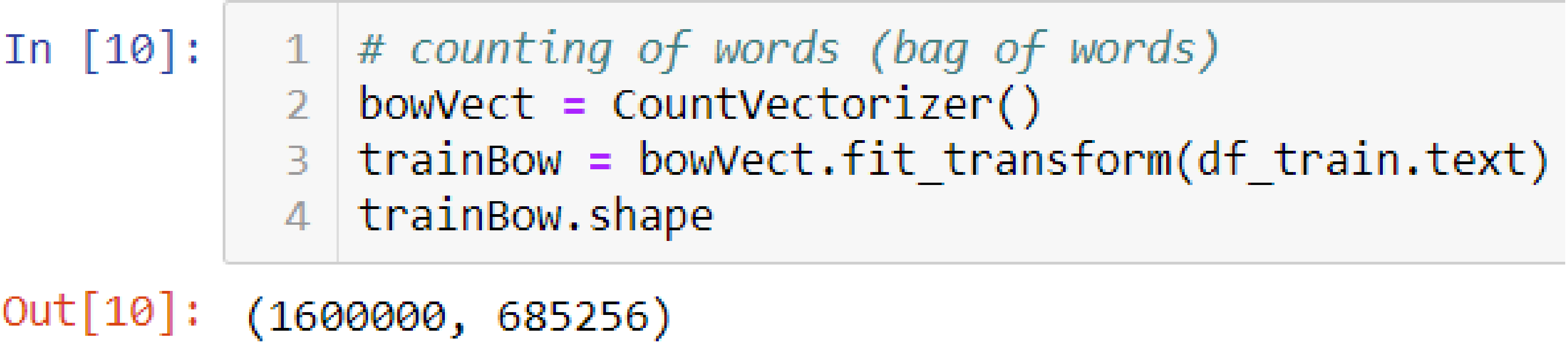
```
In [4]:  # check that training set is correct
         df_train.head()

Out[4]:
```

	label	id	date	query	user	text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

```
In [5]:  # check the class distribution
         df_train['label'].value_counts()

Out[5]:  4    800000
         0    800000
         Name: label, dtype: int64
```

Bag of Words (BoW)

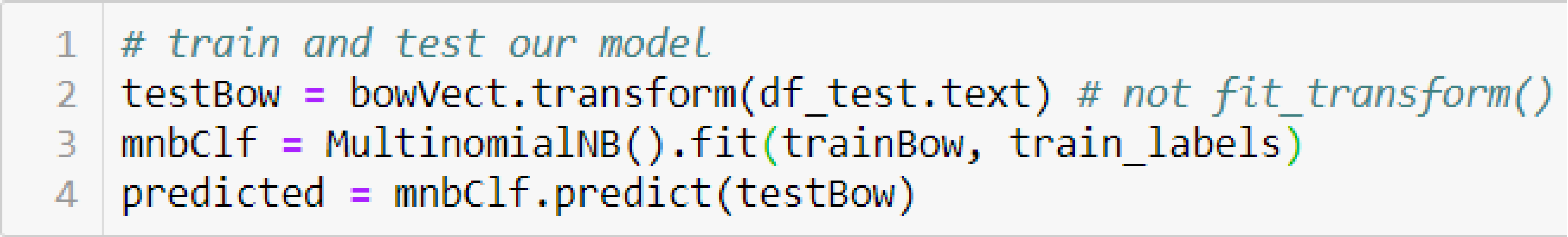
- Need to count the frequency of each word
- Use the CountVectorizer() function



```
In [10]:  # counting of words (bag of words)
           bowVect = CountVectorizer()
           trainBow = bowVect.fit_transform(df_train.text)
           trainBow.shape

Out[10]: (1600000, 685256)
```

Training and Testing

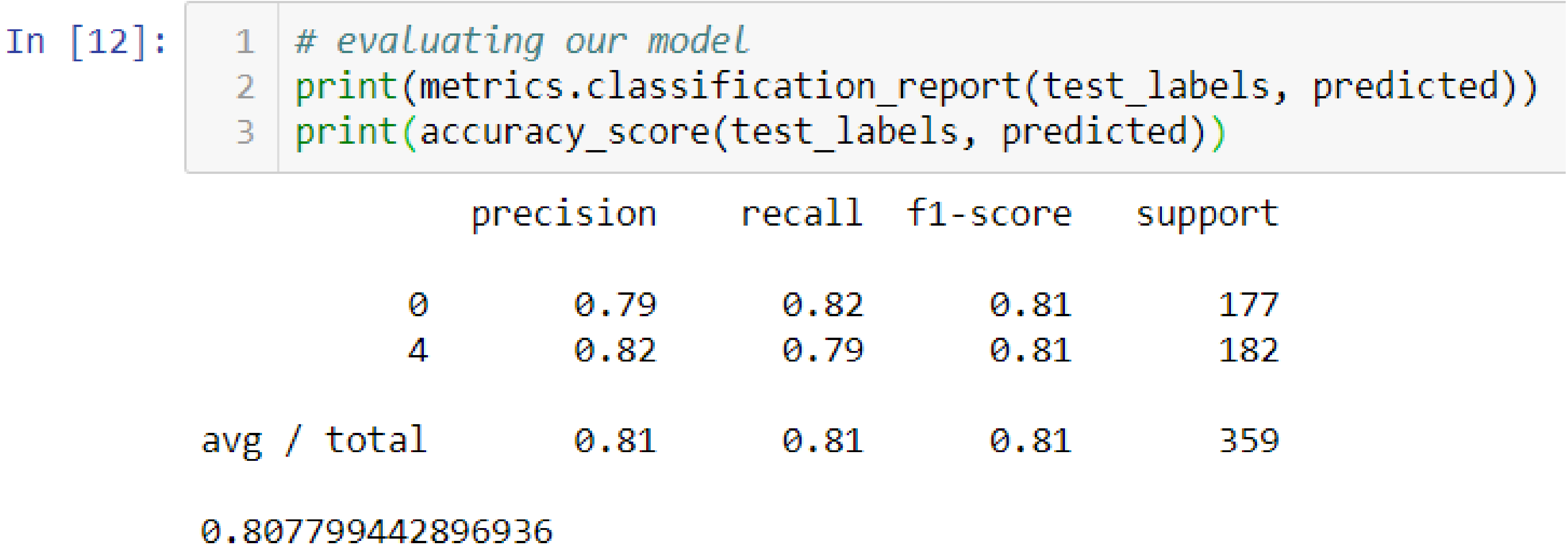
- Convert test set to BoW representation
- Train our model by calling fit() on our training set and labels
- Test our model by calling predict()



```
# train and test our model
testBow = bowVect.transform(df_test.text) # not fit_transform()
mnbClf = MultinomialNB().fit(trainBow, train_labels)
predicted = mnbClf.predict(testBow)
```

Evaluation Scores

- Look at Accuracy, Precision, Recall and F-measure



```
In [12]:
# evaluating our model
print(metrics.classification_report(test_labels, predicted))
print(accuracy_score(test_labels, predicted))
```

```
             precision    recall  f1-score   support

          0       0.79      0.82      0.81       177
          4       0.82      0.79      0.81       182

avg / total       0.81      0.81      0.81       359

0.807799442896936
```

Different Features and Models

- What happens if we need to evaluate different features or classifiers?
  - Instead of words (uni-grams), use bi-grams and tri-grams?
  - Instead of Naïve Bayes, use Decision Tree, Random Forest, SVM, etc?

- Can use Scikit-learn’s Pipeline
  - It takes an input, does a series of pre-processing step before feeding it to a final classifier
  - More information at http://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html

Exercise

5. Load in Twitter dataset (both files) and count the frequency of each class
6. Implement a simple pipeline with Naïve Bayes classifier with words (unigrams) as features
   - Using “training.1600000.processed.noemoticon.csv”, evaluate your model with 80% training and 20% test
   - hint: look at the train_test_split() function
7. Enhance the Naïve Bayes model by removing stop words, and experimenting with bi-grams and tri-grams
   - hint: look at the various parameters for CountVectorizer()