Scikit-learn

COMSW4705 Fall 2017
Elsbeth Turcan and Fei-Tzin Lee
Pipeline

- Data gathering/preprocessing
- Vectorization
- Training
- Prediction

```python
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import TfidfVectorizer
import sklearn.metrics
import sklearn.neighbors

print("Loading 20 newsgroups dataset for categories:")
data_train = fetch_20newsgroups(subset='train', shuffle=True, random_state=42)
data_test = fetch_20newsgroups(subset='test', shuffle=True, random_state=42)
print('data loaded')

"""Create tf-idf vectors for the input"
vectorizer = TfidfVectorizer(sublinear_tf=True, max_df=0.9,
                             stop_words='english')
X_train = vectorizer.fit_transform(data_train.data)
X_test = vectorizer.transform(data_test.data)
y_train = data_train.target
y_test = data_test.target

"""Train a K-Neighbors Classifier on the data"
n_neighbors = 2
weights = 'uniform'
clf = sklearn.neighbors.KNeighborsClassifier(n_neighbors, weights=weights)
clf.fit(X_train, y_train)

"""Make predictions on the test data using the trained classifier"
y_predicted = clf.predict(X_test)
print ('Classification report:')
print sklearn.metrics.classification_report(y_test, y_predicted,
target_names=data_test.target_names)
```
Feature Selection

- Selects a subset of features
- Can be the best-performing features; can eliminate redundant features
Tuning

- Models have various parameters and certain parameter settings are more appropriate for your problem

- Use the performance on the development set to determine the optimal parameter settings

Cross-validation

- N-fold (often tenfold) cross-validation splits the training data into N sections, or “folds”, and iterates over them, treating each fold as a miniature test set in one iteration and training on all other data.

- Useful for analyzing the robustness of your model.


Saving models

- Scikit-learn saves models to file using the built-in library **pickle**

  ```python
  pickle.dump(model, open('model.pkl', 'w+'))
  ```

- Models can be loaded in new files without knowing what they originally were

  ```python
  model = pickle.load(open('model.pkl', 'r'))
  model.predict(...)
  ```
Demo: numpy + pickle

- Numpy arrays vs. regular lists of lists
- Converting from double lists to np arrays and back
- Indexing into np matrices
- Pickling and unpickling arrays