Scikit-learn

COMSW4705 Fall 2017
Elsbeth Turcan and Fei-Tzin Lee

Pipeline

- Data gathering/preprocessing
- Vectorization
- Training
- Prediction

```
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
- http://scikit-learn.org/stable/modules/feature_selection.html

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
- http://scikit-learn.org/stable/modules/grid_search.html

Parameters:

penalty : str, 'l1' or 'l2', default: 'l2'

Used to specify the norm used in the penalization. The 'newton-cg', 'sag' and 'lbfgs' solvers support only l2 penalties.

New in version 0.19: l1 penalty with SAGA solver (allowing 'multinomial' + L1)

dual : bool, default: False

Dual or primal formulation. Dual formulation is only implemented for l2 penalty with liblinear solver. Prefer dual=False when n_samples > n_features.

tol : float, default: 1e-4

Tolerance for stopping criteria.

C : float, default: 1.0

Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.

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
- http://scikit-learn.org/stable/modules/cross_validation.html
- http://scikit-learn.org/stable/tutorial/statistical_inference/model_selection.html

Saving models

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

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

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

```
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