**Методические указания к лабораторной работе №3**

В данной работе мы продолжаем работать с библиотекой scikit-learn (<http://scikit-learn.org>), и хотим выяснить ее возможности при работе с текстовыми документами.

Ниже приведены новые модули, которые будут использованы в данной работе.

GridSearchCV - <http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html> - полный перебор по сетке заданных значений параметров для классификации

Decision Tree - <http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier> – Метод деревьев решений

MultinominalNB - <http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html> - Полиномиальный (Мультиномиальный) Наивный Байесовский метод – разновидность Наивного Байесовского метода, которая хорошо работает с текстами, длины которых сильно варьируются[[1]](#footnote-1).

LogisticRegression - <http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html> - Логистическая регрессия

**Настройка параметров с использованием grid search**

С помощью конвейерной обработки (pipelines), рассмотренной в лабораторной работе №2, стало гораздо проще указывать параметры обучения модели, однако перебирать все возможные варианты вручную даже в нашем простом случае выйдет затратно. В случае, рассмотренном в лабораторной работе №2, имеется четыре настраиваемых параметра: max\_features, stop\_words, use\_idf и n\_neighbors:

|  |
| --- |
| text\_clf = Pipeline([('vect', CountVectorizer(max\_features= 1000, stop\_words = 'english')),  ('tfidf', TfidfTransformer(use\_idf = True)),  ('clf', KNeighborsClassifier (n\_neighbors=1)),]) |

Вместо поиска лучших параметров в конвейере вручную, можно запустить поиск (методом полного перебора) лучших параметров в сетке возможных значений. Сделать это можно с помощью объекта класса GridSearchCV.

|  |
| --- |
| parameters = {'vect\_\_max\_features': (100,500,1000,5000,10000),  'vect\_\_stop\_words': ('english', None),  'tfidf\_\_use\_idf': (True, False)  'clf\_\_n\_neighbors': (1,3,5,7)} |

Для того чтобы задать сетку параметров необходимо создать переменную-словарь, ключами которого являются конструкции вида: «НазваниеШагаКонвейера\_\_НазваниеПараметра», а значениями – кортеж из значений параметра:

Далее необходимо создать объект класса GridSearchCV, передав в него объект pipeline или классификатор, список параметров сетки, а также при необходимости, задав прочие параметры, такие так количество задействованых ядер процессора n\_jobs, количество фолдов кросс-валидации cv и другие

|  |
| --- |
| gs\_clf = GridSearchCV(text\_clf, parameters, n\_jobs=-1, cv=3) |

Теперь объект gs\_clf можно обучить по всем параметрам, как обычный классификатор, методом fit().

После того как прошло обучение, узнать лучшую совокупность параметров можно, обратившись к атрибуту best\_params\_:

|  |
| --- |
| gs\_clf.best\_params\_ |

Аккуратность классификации – обратившись к параметру best\_score\_:

|  |
| --- |
| gs\_clf.best\_score\_ |

А результаты по всей сетке параметров - cv\_results\_

|  |
| --- |
| gs\_clf.cv\_results\_ |

1. <https://logic.pdmi.ras.ru/~sergey/teaching/mlstc12/sem01-naivebayes.pdf> [↑](#footnote-ref-1)