Рубежный контроль №2

Садыков Марк, ИУ5-63Б

Тема: Технологии использования и оценки моделей машинного обучения.

In [4]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
from sklearn.naive_bayes import MultinomialNB, ComplementNB, BernoulliNB,Categor
icalNB
from sklearn.metrics import accuracy_score
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
from IPython.display import set_matplotlib_formats
set_matplotlib_formats("retina")
pd.set_option("display.width", 70)
```

In [5]:

```
data_train = fetch_20newsgroups(subset='train', remove=('headers', 'footers'))
data_test = fetch_20newsgroups(subset='test', remove=('headers', 'footers'))
```

In [7]:

```
data_train.target.shape
```

Out[7]:

```
(11314,)
```

In [9]:

```
data.dtypes
```

Out[9]:

```
Employee_Name                object
EmpID                       float64
MarriedID                   float64
MaritalStatusID             float64
GenderID                    float64
EmpStatusID                 float64
DeptID                      float64
PerfScoreID                 float64
FromDiversityJobFairID      float64
PayRate                     float64
Termd                       float64
PositionID                  float64
Position                     object
State                        object
Zip                         float64
DOB                          object
Sex                          object
MaritalDesc                  object
CitizenDesc                  object
HispanicLatino               object
RaceDesc                     object
DateofHire                   object
DateofTermination            object
TermReason                   object
EmploymentStatus             object
Department                   object
ManagerName                  object
ManagerID                   float64
RecruitmentSource            object
PerformanceScore             object
EngagementSurvey            float64
EmpSatisfaction             float64
SpecialProjectsCount        float64
LastPerformanceReview_Date   object
DaysLateLast30              float64
dtype: object
```

In [8]:

```
vectorizer = TfidfVectorizer()
vectorizer.fit(data_train.data + data_test.data)
```

Out[8]:

```
TfidfVectorizer(analyzer='word', binary=False, decode_error='stric
t',
                dtype=<class 'numpy.float64'>, encoding='utf-8',
                input='content', lowercase=True, max_df=1.0, max_fea
tures=None,
                min_df=1, ngram_range=(1, 1), norm='l2', preprocesso
r=None,
                smooth_idf=True, stop_words=None, strip_accents=Non
e,
                sublinear_tf=False, token_pattern='(?u)\\b\\w\\w+
\\b',
                tokenizer=None, use_idf=True, vocabulary=None)
```

In [9]:

```
X_train = vectorizer.transform(data_train.data)
X_test = vectorizer.transform(data_test.data)

y_train = data_train.target
y_test = data_test.target
```

In [11]:

```
def test(model):
    print(model)
    model.fit(X_train, y_train)
    ac = accuracy_score(y_test, model.predict(X_test))
    print("accuracy:", ac)
    return ac
```

In [12]:

```
test(LogisticRegression(solver='lbfgs', multi_class='auto'))
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_interce
pt=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, ve
rbose=0,
                   warm_start=False)
accuracy: 0.774429102496017
```

Out[12]:

```
0.774429102496017
```

In [13]:

```
test(LinearSVC())
```

```
LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True,
          intercept_scaling=1, loss='squared_hinge', max_iter=1000,
          multi_class='ovr', penalty='l2', random_state=None, tol=0.
0001,
          verbose=0)
accuracy: 0.8048327137546468
```

Out[13]:

```
0.8048327137546468
```

In [14]:

```
test(MultinomialNB())
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
accuracy: 0.72623473181094
```

Out[14]:

```
0.72623473181094
```

In [15]:

```
test(ComplementNB(alpha=0.3))
```

```
ComplementNB(alpha=0.3, class_prior=None, fit_prior=True, norm=Fals
e)
accuracy: 0.812931492299522
```

Out[15]:

```
0.812931492299522
```

In [16]:

```
test(BernoulliNB())
```

```
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=Tru
e)
accuracy: 0.5371747211895911
```

Out[16]:

```
0.5371747211895911
```

In [17]:

```
balpa=0
bac=0
i=1
while i>0:
    res=test(ComplementNB(alpha=i))
#     print(res)
    if res>bac:
        balpa=i
        bac=res
    i-=0.1
print(balpa," ",bac,"\n")
```

```
ComplementNB(alpha=1, class_prior=None, fit_prior=True, norm=False)
accuracy: 0.8089484864577802
ComplementNB(alpha=0.9, class_prior=None, fit_prior=True, norm=Fals
e)
accuracy: 0.8102761550716941
ComplementNB(alpha=0.8, class_prior=None, fit_prior=True, norm=Fals
e)
accuracy: 0.8101433882103027
ComplementNB(alpha=0.7000000000000001, class_prior=None, fit_prior=T
rue,
             norm=False)
accuracy: 0.8117365905469994
ComplementNB(alpha=0.6000000000000001, class_prior=None, fit_prior=T
rue,
             norm=False)
accuracy: 0.8120021242697822
ComplementNB(alpha=0.5000000000000001, class_prior=None, fit_prior=T
rue,
             norm=False)
accuracy: 0.8117365905469994
ComplementNB(alpha=0.40000000000000013, class_prior=None, fit_prior=
True,
             norm=False)
accuracy: 0.8127987254381307
ComplementNB(alpha=0.30000000000000016, class_prior=None, fit_prior=
True,
             norm=False)
accuracy: 0.812931492299522
ComplementNB(alpha=0.20000000000000015, class_prior=None, fit_prior=
True,
             norm=False)
accuracy: 0.811603823685608
ComplementNB(alpha=0.10000000000000014, class_prior=None, fit_prior=
True,
             norm=False)
accuracy: 0.8101433882103027
ComplementNB(alpha=1.3877787807814457e-16, class_prior=None, fit_pri
or=True,
             norm=False)
accuracy: 0.6955655868295274
0.30000000000000016   0.812931492299522


/home/mark/.local/lib/python3.7/site-packages/sklearn/naive_bayes.p
y:507: UserWarning: alpha too small will result in numeric errors, s
etting alpha = 1.0e-10
  'setting alpha = %.1e' % _ALPHA_MIN)
```

In [18]:

```
balpa=0
bac=0
i=0.2
while i>0:
    res=test(MultinomialNB(alpha=i))
#     print(res)
    if res>bac:
        balpa=i
        bac=res
    i-=0.02
print(balpa," ",bac,"\n")
```

```
MultinomialNB(alpha=0.2, class_prior=None, fit_prior=True)
accuracy: 0.7738980350504514
MultinomialNB(alpha=0.18000000000000002, class_prior=None, fit_prior
=True)
accuracy: 0.7769516728624535
MultinomialNB(alpha=0.16000000000000003, class_prior=None, fit_prior
=True)
accuracy: 0.7804036112586299
MultinomialNB(alpha=0.14000000000000004, class_prior=None, fit_prior
=True)
accuracy: 0.782793414763675
MultinomialNB(alpha=0.12000000000000004, class_prior=None, fit_prior
=True)
accuracy: 0.7851832182687202
MultinomialNB(alpha=0.10000000000000003, class_prior=None, fit_prior
=True)
accuracy: 0.7886351566648965
MultinomialNB(alpha=0.08000000000000003, class_prior=None, fit_prior
=True)
accuracy: 0.7908921933085502
MultinomialNB(alpha=0.060000000000000026, class_prior=None, fit_prio
r=True)
accuracy: 0.7943441317047265
MultinomialNB(alpha=0.04000000000000002, class_prior=None, fit_prior
=True)
accuracy: 0.7987254381306426
MultinomialNB(alpha=0.02000000000000002, class_prior=None, fit_prior
=True)
accuracy: 0.8023101433882103
MultinomialNB(alpha=2.0816681711721685e-17, class_prior=None, fit_pr
ior=True)
accuracy: 0.7458842272968667
0.02000000000000002   0.8023101433882103


/home/mark/.local/lib/python3.7/site-packages/sklearn/naive_bayes.p
y:507: UserWarning: alpha too small will result in numeric errors, s
etting alpha = 1.0e-10
  'setting alpha = %.1e' % _ALPHA_MIN)
```

In [19]:

```
balpa=0
bac=0
i=0.1
while i>0:
    res=test(BernoulliNB(alpha=i))
#     print(res)
    if res>bac:
        balpa=i
        bac=res
    i-=0.01
print(balpa," ",bac,"\n")
```

```
BernoulliNB(alpha=0.1, binarize=0.0, class_prior=None, fit_prior=Tru
e)
accuracy: 0.6435209771640998
BernoulliNB(alpha=0.09000000000000001, binarize=0.0, class_prior=Non
e,
            fit_prior=True)
accuracy: 0.6456452469463622
BernoulliNB(alpha=0.08000000000000002, binarize=0.0, class_prior=Non
e,
            fit_prior=True)
accuracy: 0.6473712161444504
BernoulliNB(alpha=0.07000000000000002, binarize=0.0, class_prior=Non
e,
            fit_prior=True)
accuracy: 0.650292087095061
BernoulliNB(alpha=0.06000000000000002, binarize=0.0, class_prior=Non
e,
            fit_prior=True)
accuracy: 0.6524163568773235
BernoulliNB(alpha=0.05000000000000002, binarize=0.0, class_prior=Non
e,
            fit_prior=True)
accuracy: 0.6548061603823686
BernoulliNB(alpha=0.040000000000000015, binarize=0.0, class_prior=No
ne,
            fit_prior=True)
accuracy: 0.6589219330855018
BernoulliNB(alpha=0.030000000000000013, binarize=0.0, class_prior=No
ne,
            fit_prior=True)
accuracy: 0.6631704726500266
BernoulliNB(alpha=0.02000000000000001, binarize=0.0, class_prior=Non
e,
            fit_prior=True)
accuracy: 0.6684811471056824
BernoulliNB(alpha=0.01000000000000001, binarize=0.0, class_prior=Non
e,
            fit_prior=True)
accuracy: 0.6743228890069038
BernoulliNB(alpha=1.0408340855860843e-17, binarize=0.0, class_prior=
None,
            fit_prior=True)
accuracy: 0.7132235793945831
1.0408340855860843e-17   0.7132235793945831


/home/mark/.local/lib/python3.7/site-packages/sklearn/naive_bayes.p
y:507: UserWarning: alpha too small will result in numeric errors, s
etting alpha = 1.0e-10
  'setting alpha = %.1e' % _ALPHA_MIN)
```

In [ ]: