Московский государственный технический университет им. Н.Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа
по дисциплине
«Методы машинного обучения»
на тему
«Технологии использования и оценки моделей машинного обучения.»

Выполнил:
студент группы ИУ5-64Б
Береговая Д.

_____

Москва — 2019 г.

1. Задание:

Необходимо решить задачу классификации текстов на основе любого выбранного Вами датасета (кроме примера, который рассматривался в лекции). Классификация может быть бинарной или многоклассовой. Целевой признак из выбранного Вами датасета может иметь любой физический смысл, примером является задача анализа тональности текста. Необходимо сформировать признаки на основе CountVectorizer или TfidfVectorizer. В качестве классификаторов необходимо использовать два классификатора, не относящихся к наивным Байесовским методам (например, LogisticRegression, LinearSVC), а также Multinomial Naive Bayes (MNB), Complement Naive Bayes (CNB), Bernoulli Naive Bayes. Для каждого метода необходимо оценить качество классификации с помощью хотя бы одной метрики качества классификации (например, Accuracy). Сделате выводы о том, какой классификатор осуществляет более качественную классификацию на Вашем наборе данных.

1.0.1. Выполнение:

Подключим библиотеки, загрузим набор данных:

[8]:
```
import os
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score,␣
 ↪classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import plot_confusion_matrix
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_absolute_error, mean_squared_error,␣
 ↪mean_squared_log_error, median_absolute_error, r2_score
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.svm import SVC, NuSVC, LinearSVC, OneClassSVM, SVR, NuSVR,␣
 ↪LinearSVR
from sklearn.tree import DecisionTreeClassifier, DecisionTreeRegressor,␣
 ↪export_graphviz
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
from sklearn.ensemble import ExtraTreesClassifier, ExtraTreesRegressor
from sklearn.ensemble import GradientBoostingClassifier,␣
 ↪GradientBoostingRegressor
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score, precision_score
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt
from sklearn.naive_bayes import MultinomialNB, ComplementNB, BernoulliNB
```

```
from sklearn.metrics import accuracy_score
from sklearn.svm import LinearSVC
from sklearn.datasets import fetch_20newsgroups_vectorized
from gmdhpy import gmdh
%matplotlib inline
sns.set(style="ticks")
```

[9]:
```
data_train = fetch_20newsgroups_vectorized(subset='train',␣
↪remove=('headers', 'footers'))
data_test = fetch_20newsgroups_vectorized(subset='test', remove=('headers',␣
↪'footers'))
```

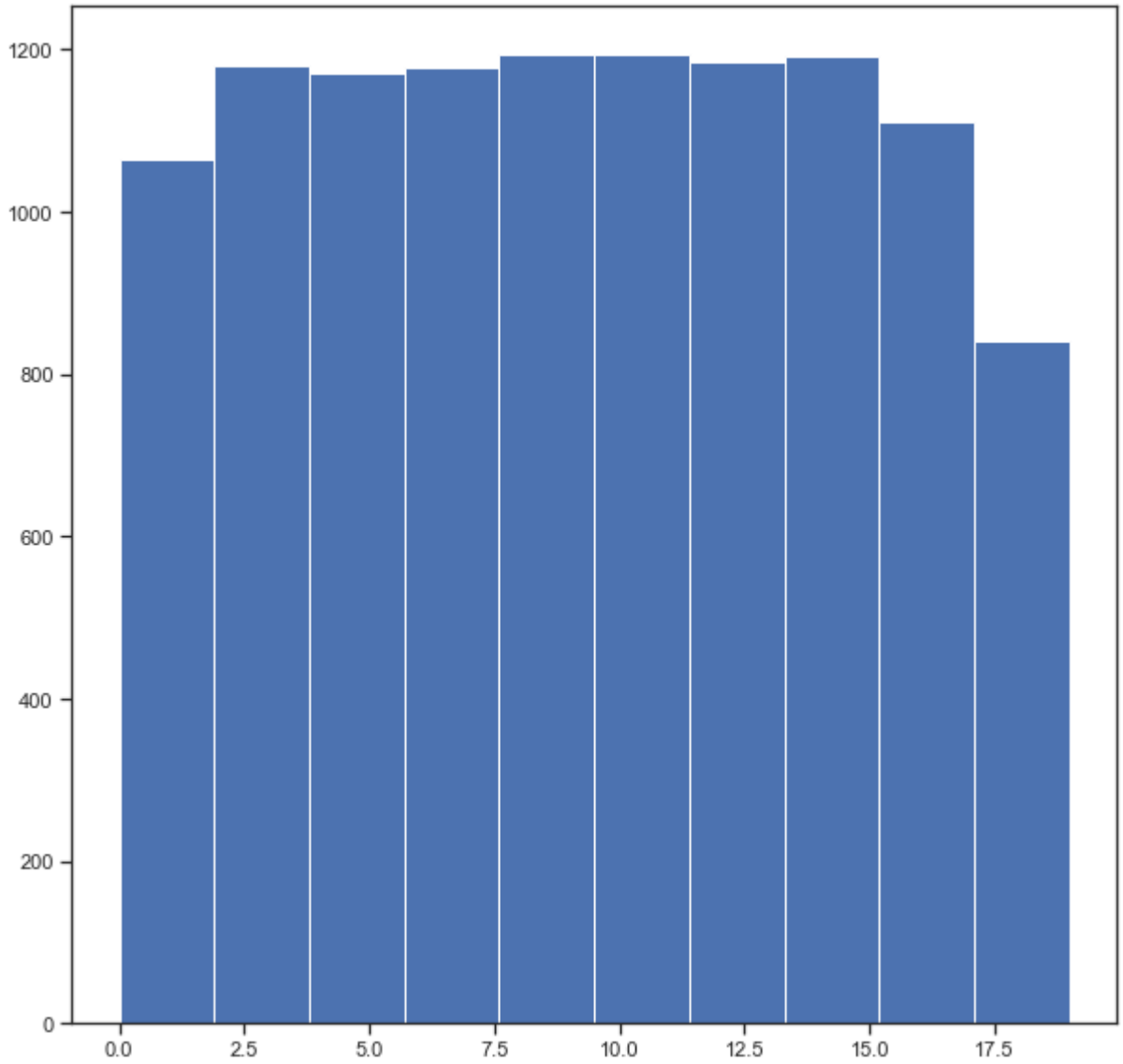
[10]:
```
data_train.target.shape
```

[10]:
```
(11314,)
```

[11]:
```
data_train.data[:3]
```

[11]:
```
<3x114751 sparse matrix of type '<class 'numpy.float64'>'
        with 592 stored elements in Compressed Sparse Row format>
```

[14]:
```
#
fig, ax = plt.subplots(figsize=(10,10))
plt.hist(data_train.target)
plt.show()
```



[15]:
```
def test(model):
    print(model)
    model.fit(data_train.data, data_train.target)
    print("accuracy:", accuracy_score(data_test.target, model.
 ↪predict(data_test.data)))
```

[16]:
```
test(LogisticRegression(solver='lbfgs', multi_class='auto',␣
 ↪max_iter=100000))
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100000,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
accuracy: 0.6558682952734998
```

[17]:
```
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: 0.7684545937334042
```

[18]:
```
test(MultinomialNB())
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
accuracy: 0.6289166224110462
```

[19]:
```
test(ComplementNB())
```

```
ComplementNB(alpha=1.0, class_prior=None, fit_prior=True, norm=False)
accuracy: 0.7943441317047265
```

[20]:
```
test(BernoulliNB())
```

```
BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)
accuracy: 0.544875199150292
```

2. Выводы

Метод Complement Naive Bayes задачу многоклассовой классификации в условиях дисбаланса классов решает лучше всего. Также хорошо себя показал метод LinearSVC

[ ]:
```

```