Московский государственный технический

университет им. Н. Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Курс «Технологии машинного обучения»

Отчет по лабораторной работе №6

Ансамбли моделей машинного обучения

Группа: ИУ5-62Б

Студент: Селедкина А.С.

Преподаватель: Гапанюк Ю.Е.

Москва, 2020 г.

Цель лабораторной работы: изучение ансамблей моделей машинного обучения.

Описание задания

1. Выберите набор данных (датасет) для решения задачи классификации или регресии.
2. В случае необходимости проведите удаление или заполнение пропусков и кодирование категориальных признаков.
3. С использованием метода train_test_split разделите выборку на обучающую и тестовую.
4. Обучите две ансамблевые модели. Оцените качество моделей с помощью одной из подходящих для задачи метрик. Сравните качество полученных моделей.

Текст программы и примеры выполнения

Будем использовать датасет по определению наличия сердечного заболевания у пациента: https://www.kaggle.com/ronitf/heart-disease-uci.

```
data = pd.read_csv('data/heart.csv')
data.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
data.shape
```

```
(303, 14)
```

```
data.columns
```

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

```
data.dtypes
```

```
age           int64
sex           int64
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

```
data.isnull().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

Разделение выборки

```
x = data[['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal']]
y = data['target']
```

```
# С использованием метода train_test_split разделим выборку на обучающую и тестовую
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=1)
print("x_train:", x_train.shape)
print("x_test:", x_test.shape)
print("y_train:", y_train.shape)
print("y_test:", y_test.shape)
```

```
x_train: (227, 13)
x_test: (76, 13)
y_train: (227,)
y_test: (76,)
```

Случайный лес

```
random_forest = RandomForestClassifier(n_estimators=50, oob_score=True, random_state=1)
random_forest.fit(x_train, y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=50,
                       n_jobs=None, oob_score=True, random_state=1, verbose=0,
                       warm_start=False)
```

```
y_predicted_rf = random_forest.predict(x_test)
```

```
classification_report(y_predicted_rf, y_test, output_dict=True)
```

```
{'0': {'precision': 0.7142857142857143,
  'recall': 0.7142857142857143,
  'f1-score': 0.7142857142857143,
  'support': 35},
 '1': {'precision': 0.7560975609756098,
  'recall': 0.7560975609756098,
  'f1-score': 0.7560975609756099,
  'support': 41},
 'accuracy': 0.7368421052631579,
 'macro avg': {'precision': 0.735191637630662,
  'recall': 0.735191637630662,
  'f1-score': 0.7351916376306621,
  'support': 76},
 'weighted avg': {'precision': 0.7368421052631579,
  'recall': 0.7368421052631579,
  'f1-score': 0.7368421052631579,
  'support': 76}}
```

Бустинг

```
ada_boost = AdaBoostClassifier(n_estimators=50, algorithm='SAMME', random_state=1)
ada_boost.fit(x_train, y_train)
```

```
AdaBoostClassifier(algorithm='SAMME', base_estimator=None, learning_rate=1.0,
                   n_estimators=50, random_state=1)
```

```
y_predicted_ab = ada_boost.predict(x_test)
```

```
classification_report(y_predicted_ab, y_test, output_dict=True)
```

```
{'0': {'precision': 0.7428571428571429,
  'recall': 0.7647058823529411,
  'f1-score': 0.7536231884057971,
  'support': 34},
 '1': {'precision': 0.8048780487804879,
  'recall': 0.7857142857142857,
  'f1-score': 0.7951807228915663,
  'support': 42},
 'accuracy': 0.7763157894736842,
 'macro avg': {'precision': 0.7738675958188154,
  'recall': 0.7752100840336134,
  'f1-score': 0.7744019556486816,
  'support': 76},
 'weighted avg': {'precision': 0.7771318540253072,
  'recall': 0.7763157894736842,
  'f1-score': 0.7765891943058275,
  'support': 76}}
```

По всем метрикам качества моделей лучшие результаты показал бустинг.