Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Технологии машинного обучения»

Отчет по лабораторной работе №6

Выполнила:		Проверил:
студентка группы ИУ5-61		преподаватель каф. ИУ5
Абросимова Надежда		Гапанюк Ю.Е.
Подпись и дата:		Подпись и дата:

г. Москва, 2019 г.

Задание

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.
5. Произведите для каждой модели подбор значений одного гиперпараметра. В зависимости от используемой библиотеки можно применять функцию GridSearchCV, использовать перебор параметров в цикле, или использовать другие методы.
6. Повторите пункт 4 для найденных оптимальных значений гиперпараметров. Сравните качество полученных моделей с качеством моделей, полученных в пункте 4.

Текст программы

```
import numpy as np
import pandas as pd
from typing import Dict, Tuple
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.metrics import accuracy_score, balanced_accuracy_score
from sklearn.metrics import precision_score, recall_score, f1_score,
classification report
from sklearn.ensemble import RandomForestClassifier, RandomForestRegressor
sklearn.ensemble import GradientBoostingClassifier,
GradientBoostingRegressor
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
data=pd.read_csv('heart.csv', sep=",")
data.head()
```

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

```
X_train, X_test, y_train, y_test = train_test_split(
    data, data['target'], test_size= 0.2, random_state= 1)
```

Обучение двух ансамблевых моделей

```
#случайный лес (n_estimators - число деревьев)

randomforest = RandomForestClassifier(n_estimators=5, max_depth=1, random_
state=0).fit(X_train, y_train)
target_randomforest = randomforest.predict(X_test)
```

```
accuracy_score(y_test, target_randomforest), \
precision_score(y_test, target_randomforest), \
recall_score(y_test, target_randomforest)
```
(0.8852459016393442, 0.8157894736842105, 1.0)
```
#градиентный бустинг
gradient_boosting = GradientBoostingClassifier(n_estimators=5, max_depth=1
, learning_rate=0.01).fit(X_train, y_train)
target_gradient_boosting = gradient_boosting.predict(X_test)
accuracy_score(y_test, target_gradient_boosting), \
precision_score(y_test, target_gradient_boosting), \
recall_score(y_test, target_gradient_boosting)
```
(0.5081967213114754, 0.5081967213114754, 1.0)

Подбор гиперпараметра

```
parameters_random_forest = {'n_estimators':[1, 3, 5, 7, 10],
                            'max_depth':[1, 3, 5, 7, 10],
                            'random_state':[0, 2, 4, 6, 8, 10]}
best_random_forest = GridSearchCV(RandomForestClassifier(), parameters_ran
dom_forest, cv=3, scoring='accuracy')
best_random_forest.fit(X_train, y_train)
parameters_gradient_boosting = {'n_estimators':[1, 3, 5, 7, 10],
                            'max_depth':[1, 3, 5, 7, 10],
                            'learning_rate':[0.001, 0.0025, 0.005, 0.0075,
0.01, 0.025]}
best_gradient_boosting = GridSearchCV(GradientBoostingClassifier(), parame
ters_gradient_boosting, cv=3, scoring='accuracy')
best_gradient_boosting.fit(X_train, y_train)
```
In [13]:
```
best_random_forest.best_params_
```
Out[13]:
```
{'max_depth': 1, 'n_estimators': 1, 'random_state': 6}
```
In [14]:
```
best_gradient_boosting.best_params_
```
Out[14]:
```
{'learning_rate': 0.025, 'max_depth': 1, 'n_estimators': 5}
```
Для оптимальных значений:
```
#случайный лес (n_estimators - число деревьев)

randomforest2 = RandomForestClassifier(n_estimators=1, max_depth=1, random
_state=6).fit(X_train, y_train)

target_randomforest2 = randomforest2.predict(X_test)
accuracy_score(y_test, target_randomforest2), \
precision_score(y_test, target_randomforest2),  \
recall_score(y_test, target_randomforest2)
```
(1.0, 1.0, 1.0)

```
#градиентный бустинг gradient_boosting2 =
GradientBoostingClassifier(n_estimators=5, max_depth=1,
learning_rate=0.025).fit(X_train, y_train)
target_gradient_boosting2 = gradient_boosting2.predict(X_test)
accuracy_score(y_test, target_gradient_boosting2), \
precision_score(y_test, target_gradient_boosting2),  \
recall_score(y_test, target_gradient_boosting2)
```
(1.0, 1.0, 1.0)