МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)

Кафедра прикладной математики

ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 7

по дисциплине «Технологии и инструментарий анализа больших данных»

Выполнил студент группы ИКБО-20-21 Фомичев Р.А.

Проверил ассистент кафедры ПМ ИИТ Тетерин Н.Н.

Москва 2024

Задание 1

Найти данные для задачи классификации или для задачи регрессии (данные не должны повторятся в группе). Код представлен на рисунке 1.

```
data = pd.read_excel('Obesity_Dataset.xlsx')

x = data.drop('Class', axis=1)
y = data['Class']

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.8)
```

Рисунок 1 – Код программы

Задание 2

Реализовать баггинг. Код программы с выводом представлены на рисунках 2, 3.

```
# Случайный лес (баггинг)
model = RandomForestClassifier()

params = {
    "max_depth": [12, 16],
    "min_samples_leaf": [2, 6],
    "min_samples_split": [4, 8],
}

grid_search_forest = GridSearchCV(estimator=model, param_grid=params, scoring="f1_macro", cv=4)

start_random_forest = time.time()
grid_search_forest.fit(x_train, y_train)
end_random_forest = time.time()

best_model = grid_search_forest.best_estimator_
best_model

y_pred = best_model.predict(x_test)

print(f"F-score: {f1_score(y_test, y_pred, average='macro')}")
print(f"Время обучения модели: {(end_random_forest - start_random_forest):.4f} секунд")
```

```
F-score: 0.5551306719840727
Время обучения модели: 9.7634 секунд
```

Рисунок 2 – Код программы с выводом

```python
# DecisionTreeClassifier
models = []
fit_time = 0.0

for i in range(10):
    model_tree = tree.DecisionTreeClassifier(max_depth=8, random_state=1)
    x1, y1 = resample(x, y)

    start_my_forest = time.time()
    model_tree.fit(x1, y1)
    end_my_forest = time.time()

    fit_time += end_my_forest - start_my_forest
    models.append(model_tree)

y_preds = np.array([model.predict(x_test) for model in models])
y_pred_majority = mode(y_preds, axis=0)[0]

print(f"F-score: {f1_score(y_test, y_pred_majority, average='macro')}")
print(f"Время обучения модели: {fit_time:.4f} секунд")
```

```
F-score: 0.8933921633048284
Время обучения модели: 0.0827 секунд
```

Рисунок 3 – Код программы с выводом

Задание 3

Реализовать бустинг на тех же данных, что использовались для баггинга. Код программы с выводом представлены на рисунках 4, 5.

```python
# бустинг
# GradientBoost
model2 = GradientBoostingClassifier()

params2 = {
    "n_estimators": [10, 30, 60, 100]
}

grid_search_boost = GridSearchCV(estimator=model2, param_grid=params2, scoring="f1_macro", cv=4)

start_gradient_boost = time.time()
grid_search_boost.fit(x_train, y_train)
end_gradient_boost = time.time()

best_model = grid_search_boost.best_estimator_

y_pred2 = best_model.predict(x_test)

print(f"F-score: {f1_score(y_test, y_pred2, average='macro')}")
print(f"Время обучения модели: {(end_gradient_boost - start_gradient_boost):.4f} секунд")
```

```
F-score: 0.652721626141348
Время обучения модели: 6.1899 секунд
```

Рисунок 4 – Код программы с выводом

```python
# CatBoost
catmodel = cb.CatBoostClassifier(iterations=3000)

start_cat_boost = time.time()
catmodel.fit(x_train, y_train)
end_cat_boost = time.time()

y_pred_cat = catmodel.predict(x_test)

print(f"F-score: {f1_score(y_test, y_pred_cat, average='macro')}")
print(f"Время обучения модели: {(end_cat_boost - start_cat_boost):.4f} секунд")
```

```
2972:   learn: 0.0250530        total: 4.65s    remaining: 42.2ms
2973:   learn: 0.0250389        total: 4.65s    remaining: 40.7ms
2974:   learn: 0.0250306        total: 4.65s    remaining: 39.1ms
2975:   learn: 0.0250198        total: 4.66s    remaining: 37.6ms
2976:   learn: 0.0250137        total: 4.66s    remaining: 36ms
2977:   learn: 0.0250040        total: 4.66s    remaining: 34.5ms
2978:   learn: 0.0249952        total: 4.67s    remaining: 32.9ms
2979:   learn: 0.0249846        total: 4.67s    remaining: 31.3ms
2980:   learn: 0.0249703        total: 4.67s    remaining: 29.8ms
2981:   learn: 0.0249588        total: 4.67s    remaining: 28.2ms
2982:   learn: 0.0249436        total: 4.68s    remaining: 26.7ms
2983:   learn: 0.0249338        total: 4.68s    remaining: 25.1ms
2984:   learn: 0.0249273        total: 4.68s    remaining: 23.5ms
2985:   learn: 0.0249206        total: 4.69s    remaining: 22ms
2986:   learn: 0.0249136        total: 4.69s    remaining: 20.4ms
2987:   learn: 0.0249014        total: 4.69s    remaining: 18.8ms
2988:   learn: 0.0248923        total: 4.7s     remaining: 17.3ms
2989:   learn: 0.0248856        total: 4.7s     remaining: 15.7ms
2990:   learn: 0.0248736        total: 4.7s     remaining: 14.1ms
2991:   learn: 0.0248650        total: 4.7s     remaining: 12.6ms
2992:   learn: 0.0248540        total: 4.7s     remaining: 11ms
2993:   learn: 0.0248445        total: 4.7s     remaining: 9.42ms
2994:   learn: 0.0248334        total: 4.7s     remaining: 7.85ms
2995:   learn: 0.0248219        total: 4.7s     remaining: 6.28ms
2996:   learn: 0.0248132        total: 4.7s     remaining: 4.71ms
2997:   learn: 0.0247990        total: 4.71s    remaining: 3.14ms
2998:   learn: 0.0247886        total: 4.71s    remaining: 1.57ms
2999:   learn: 0.0247783        total: 4.71s    remaining: 0us
F-score: 0.6558146590363831
Время обучения модели: 5.6121 секунд
```

Рисунок 5 – Код программы с выводом

Задание 4

Сравнить результаты работы алгоритмов (время работы и качество моделей). Сделать выводы.

Самым точным оказался методом обучения на данных по классификации ожирения оказался баггинг, реализованный на основе DecisionTreeClassifier.

Случайный лес (RandomForestClassifier) оказался наименее точным согласно метрике F1. Также время его работы оказалось наибольшим.

Бустинг находится посередине между представленными выше методами по времени и по метрике F1.