



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет имени  
Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

---

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

## **Лабораторная работа № 8 по дисциплине «Методы машинного обучения»**

Тема Анализ социологического исследования, часть 2

Студент Сапожков А.М.

Группа ИУ7-23М

Преподаватель Солодовников В.И.

Москва, 2025

# Содержание

<b>ВВЕДЕНИЕ</b>	<b>4</b>
<b>1 Аналитическая часть</b>	<b>5</b>
1.1 Классификация	5
1.2 Деревья решений	5
1.3 Ансамбли	5
<b>2 Технологическая часть</b>	<b>7</b>
2.1 Средства реализации	7
2.2 Реализация алгоритмов	7
<b>3 Исследовательская часть</b>	<b>11</b>
<b>ЗАКЛЮЧЕНИЕ</b>	<b>15</b>

# ВВЕДЕНИЕ

Классификация данных представляет собой одну из ключевых задач машинного обучения, для решения которой разработано множество эффективных алгоритмов. Среди них особое место занимают деревья решений и ансамблевые методы, демонстрирующие высокую интерпретируемость результатов и устойчивость к переобучению.

Целью данной лабораторной работы является изучение деревьев решений и ансамблевых классификаторов на примере анализа социологического исследования.

Задачи данной лабораторной работы:

- 1) определить, какие из признаков состояния наиболее сильно связаны с интегральной оценкой счастья (благополучия) респондента;
- 2) пользуясь найденными закономерностями спрогнозировать попадание респондентов, у которых интегральная характеристика отмечена как "Неизвестно" в укрупнённые группы шкалы Кантрила;
- 3) построить следующие классификаторы: многоклассовую логистическую регрессию, дерево решений, ансамблевый классификатор;
- 4) сравнить матрицы ошибок и метрики качества классификации.

# 1 Аналитическая часть

## 1.1 Классификация

Классификация (classification) — это задача присвоения меток класса (class label) наблюдениям (Observation) объектам из предметной области. Множество допустимых меток класса конечно. В свою очередь класс — это множество всех объектов с данным значением метки. Требуется построить алгоритм, способный классифицировать (присвоить метку) произвольный объект из исходного множества. Классификация, как правило, на этапе настройки использует обучение с учителем.

## 1.2 Деревья решений

Деревья решений относятся к методам поиска логических закономерностей в данных, а также являются основным подходом, применимым в теории принятия решений.

Они позволяют осуществлять решение целого класса задач классификации и регрессии в виде многошагового процесса принятия решений и используют особенности древовидных классификаторов, связанных с учётом локальных свойств классифицируемых объектов на каждом уровне и в каждом узле дерева, что позволяет реализовать как прямую, так и обратную цепочку рассуждений.

Основными достоинствами деревьев решений является

- простота и наглядность описания процесса поиска решения;
- с точки зрения математики, обучение нейронных сетей — это многопараметрическая задача нелинейной оптимизации.
- представление правил в виде продукций «если. . . то. . . ».

## 1.3 Ансамбли

Ансамбли — это контролируемые алгоритмы обучения, которые комбинируют прогнозы из двух и более алгоритмов машинного обучения для построения более точных результатов. Результаты можно комбинировать с помощью голосования или усреднения. Первое зачастую применяется в классификации, а второе — в регрессии.

Существует 3 основных типа ансамблевых алгоритмов.

Обучение перцептрона:

1. **Бэггинг**. Алгоритмы обучаются и работают параллельно на разных тренировочных наборах одного размера. Затем все они тестируются на одном наборе данных, а конечный результат определяется с помощью голосования.
2. **Бустинг**. В этом типе алгоритмы обучаются последовательно, а конечный результат отбирается с помощью голосования с весами.
3. **Стекинг (наложение)**. Исходя из названия, этот подход состоит из двух уровней,

расположенных друг на друге. Базовый представляет собой комбинацию алгоритмов, а верхний — мета-алгоритмы, основанные на базовом уровне.

## 2 Технологическая часть

### 2.1 Средства реализации

В качестве языка программирования для реализации алгоритмов был выбран язык программирования Python ввиду наличия библиотек для обучения регрессионных моделей, таких как sklearn и numpy.

### 2.2 Реализация алгоритмов

На листинге 2.1 представлена реализация алгоритма обучения классификаторов респондентов, принимавших участие в социологическом исследовании.

Листинг 2.1 — Классификация с использованием дерева решений логистической регрессии и ансамблевого классификатора

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import re
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report,
    matthews_corrcoef
from sklearn.tree import DecisionTreeClassifier, plot_tree,
    DecisionTreeRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.linear_model import LogisticRegression
from operator import itemgetter
from scipy.special import expit

pd.options.mode.copy_on_write = True

from google.colab import drive
drive.mount('/content/drive')

dataset = pd.read_excel('/content/drive/MyDrive/Colab Notebooks/
    ml_lab_08/ММО_ЛР8_Исходные_данные.xlsx')
na = set(dataset.columns).difference(dataset.dropna(axis=1).columns)
class_names = list(set(dataset['Ощущаемое.счастье']))
dataset = dataset.drop(['Респондент'], axis=1)
dataset['Сообщество'] = dataset['Сообщество'].apply(lambda it: re.
    findall(r'\b\d+\b', it)[0])
```

```

dataset_unknowns = dataset[dataset['Ощущаемое.счастье'] == 'Неизвестно']
dataset_knowns = dataset[dataset['Ощущаемое.счастье'] != 'Неизвестно']
dataset_knowns['Ощущаемое.счастье'] = dataset_knowns['Ощущаемое.счастье'].
    apply(lambda it: class_names.index(it))

plt.figure(figsize=(30,24))
sns.heatmap(dataset_knowns.corr().round(decimals=2), annot=True,
    linewidths=1, cmap='Reds')

dataset_knowns['Ощущаемое.счастье'] = dataset_knowns['Ощущаемое.счастье'].
    apply(lambda it: class_names[it])
X_train, X_test, y_train, y_test = train_test_split(dataset_knowns.
    drop(['Ощущаемое.счастье'], axis=1), dataset_knowns['
    Ощущаемое.счастье'])

decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_train, y_train)

plt.figure(figsize=(200, 30))
plot_tree(decision_tree, max_depth=5, feature_names=list(dataset),
    class_names=dataset_knowns['Ощущаемое.счастье'].unique(), filled=True
    , impurity=True, rounded=True, fontsize=6)
plt.savefig("output.pdf", format="pdf")

group = {
    'Prospering': 'Thriving',
    'Thriving': 'Thriving',
    'Blooming': 'Thriving',
    'Doing well': 'Thriving',
    'Just ok': 'Struggling',
    'Coping': 'Struggling',
    'Struggling': 'Struggling',
    'Suffering': 'Suffering',
    'Depressed': 'Suffering',
    'Hopeless': 'Suffering',
}
y_test_grouped = itemgetter(*y_test)(group)

def plot_classification(y_test, y_pred, y_test_grouped,
    y_pred_grouped):
    print(f'MCC: {matthews_corrcoef(y_test, y_pred)}')
```

```

fig, axes = plt.subplots(2, 2, figsize=(17, 10))
sns.heatmap(pd.DataFrame(classification_report(y_test, y_pred,
        output_dict=True)).iloc[: -1, :].T, ax=axes[0, 0], annot=True,
        cmap='Reds')
sns.heatmap(pd.DataFrame(classification_report(y_test_grouped,
        y_pred_grouped, output_dict=True)).iloc[: -1, :].T, ax=axes[0,
        1], annot=True, cmap='Reds')
sns.heatmap(confusion_matrix(y_test, y_pred), ax=axes[1, 0], annot=
        True, fmt='d', cmap='Reds')
sns.heatmap(confusion_matrix(y_test_grouped, y_pred_grouped), ax=
        axes[1, 1], annot=True, fmt='d', cmap='Reds')
axes[0, 0].set_title('Decision tree classifier (by class)')
axes[0, 1].set_title('Decision tree classifier (by group)')
plt.show()

y_pred = decision_tree.predict(X_test)
print(f'MCC: {matthews_corrcoef(y_test, y_pred)}')
y_pred_grouped = itemgetter(*y_pred)(group)
plot_classification(y_test, y_pred, y_test_grouped, y_pred_grouped)

lr = LogisticRegression()
lr.fit(X_train, y_train)
y_pred = lr.predict(X_test)
y_pred_grouped = itemgetter(*y_pred)(group)
plot_classification(y_test, y_pred, y_test_grouped, y_pred_grouped)

valid_class_names = list(set(dataset_knowns['Ощущаемое.счастье']))
y = np.fromiter((valid_class_names.index(elem) for elem in y_train),
        np.int32) # y_train.apply(lambda it: class_names.index(it))
gbdt = DecisionTreeRegressor()
gbdt.fit(X_train, y)

map_int = {np.int64(k): v for k, v in enumerate(valid_class_names)}
y_pred = gbdt.predict(X_test).astype(np.int64)
y_pred = itemgetter(*y_pred)(map_int)
y_pred_grouped = itemgetter(*y_pred)(group)
plot_classification(y_test, y_pred, y_test_grouped, y_pred_grouped)

plt.figure(figsize=(200, 30))
plot_tree(gbdt, max_depth=5, feature_names=list(dataset), class_names
        =dataset_knowns['Ощущаемое.счастье'].unique(), filled=True, impurity=

```



```
True, rounded=True, fontsize=6)
plt.savefig("output2.pdf", format="pdf")

y = np.fromiter((valid_class_names.index(elem) for elem in y_train),
               np.int32)
abdt = AdaBoostRegressor(DecisionTreeRegressor(), n_estimators=250)
abdt.fit(X_train, y)
y_pred = abdt.predict(X_test).astype(np.int64)
y_pred = itemgetter(*y_pred)(map_int)
y_pred_grouped = itemgetter(*y_pred)(group)
plot_classification(y_test, y_pred, y_test_grouped, y_pred_grouped)
```

### **3 Исследовательская часть**

Для тестирования разработанного алгоритма применялась облачная платформа Google Colab, не требующая установки ПО на локальный компьютер.

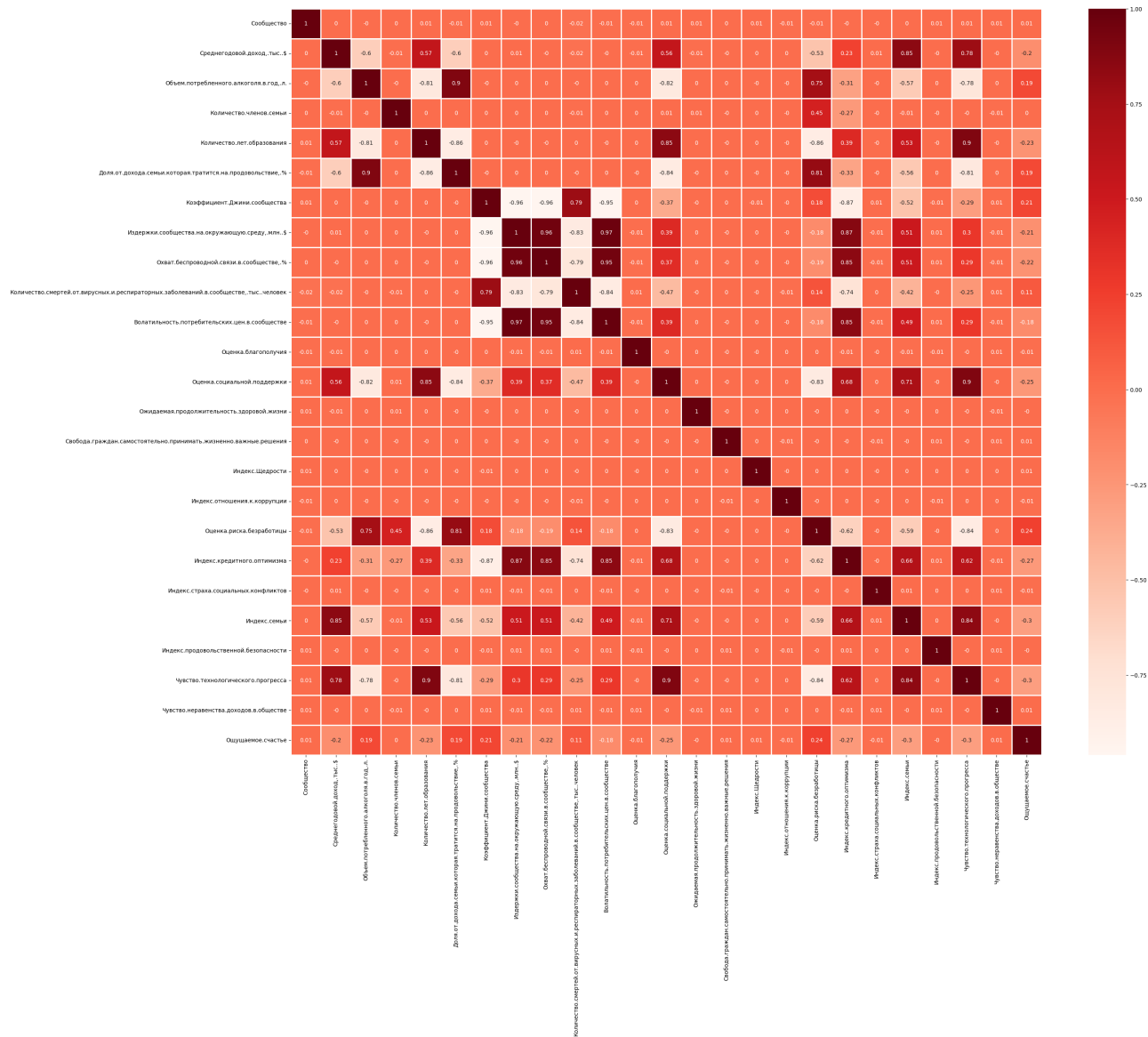


Рисунок 3.1 — Корреляция признаков состояния с интегральной оценкой счастья

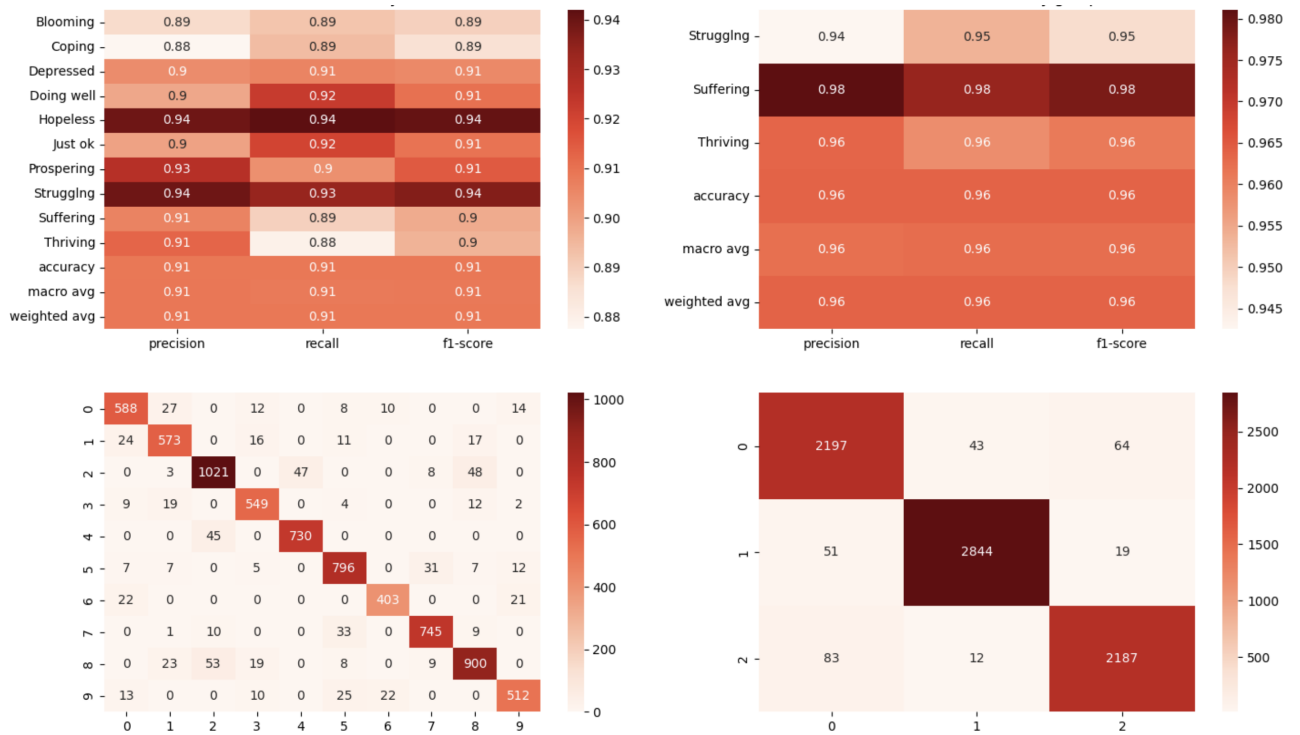


Рисунок 3.2 — Результат классификации с помощью дерева решений (MCC: 0.898)

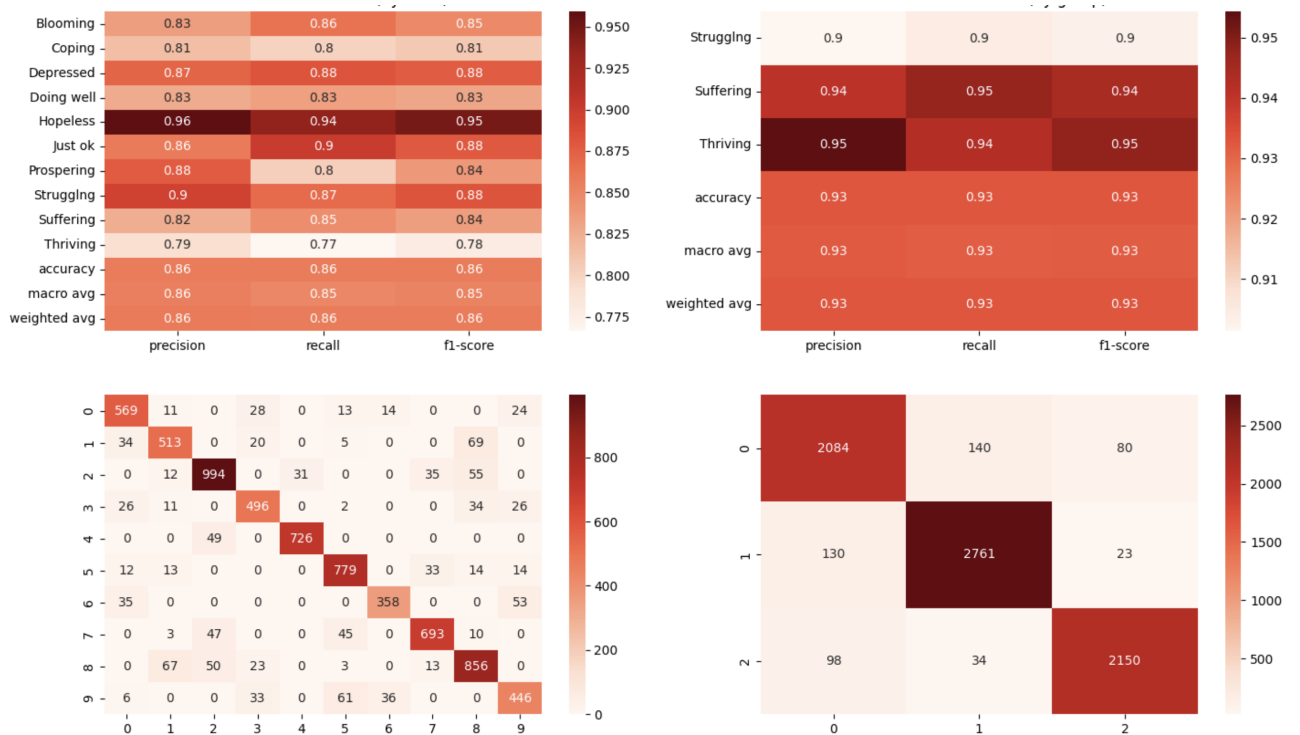


Рисунок 3.3 — Результат классификации с помощью логистической регрессии (MCC: 0.840)

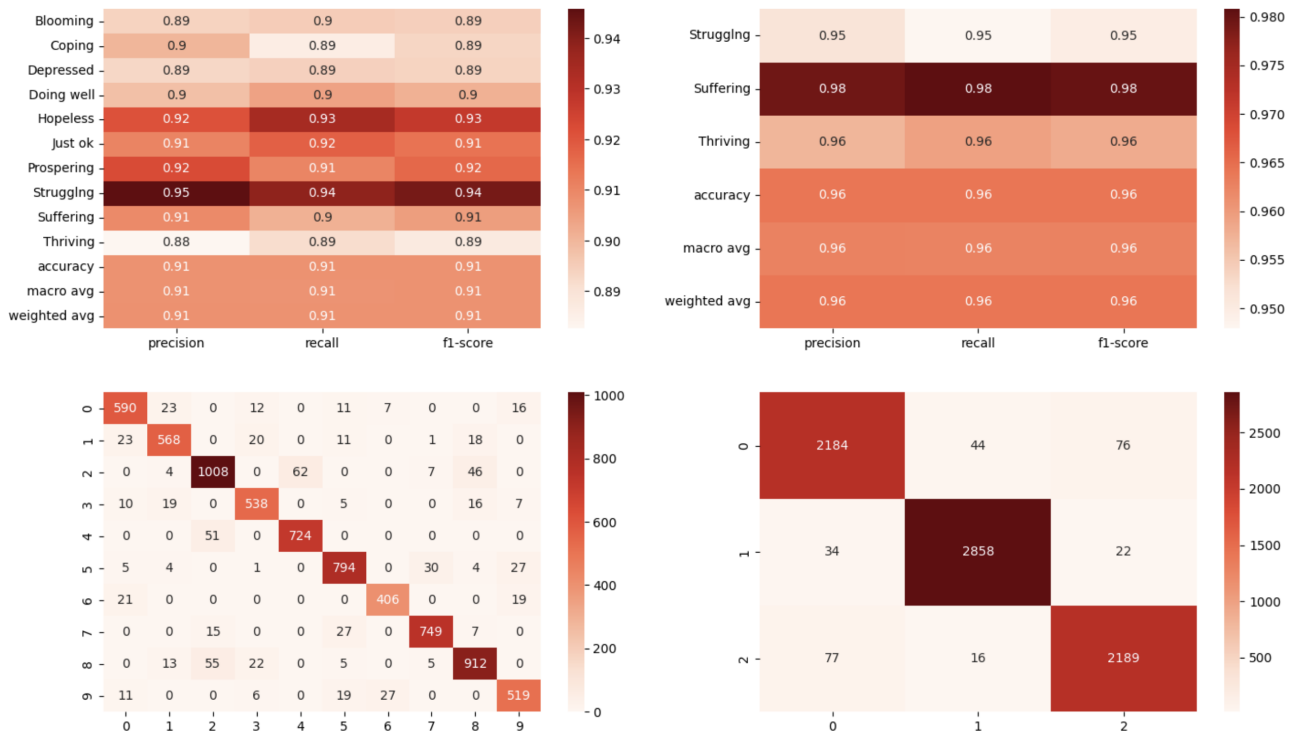


Рисунок 3.4 — Результат классификации с помощью дерева регрессии (MCC: 0.896)

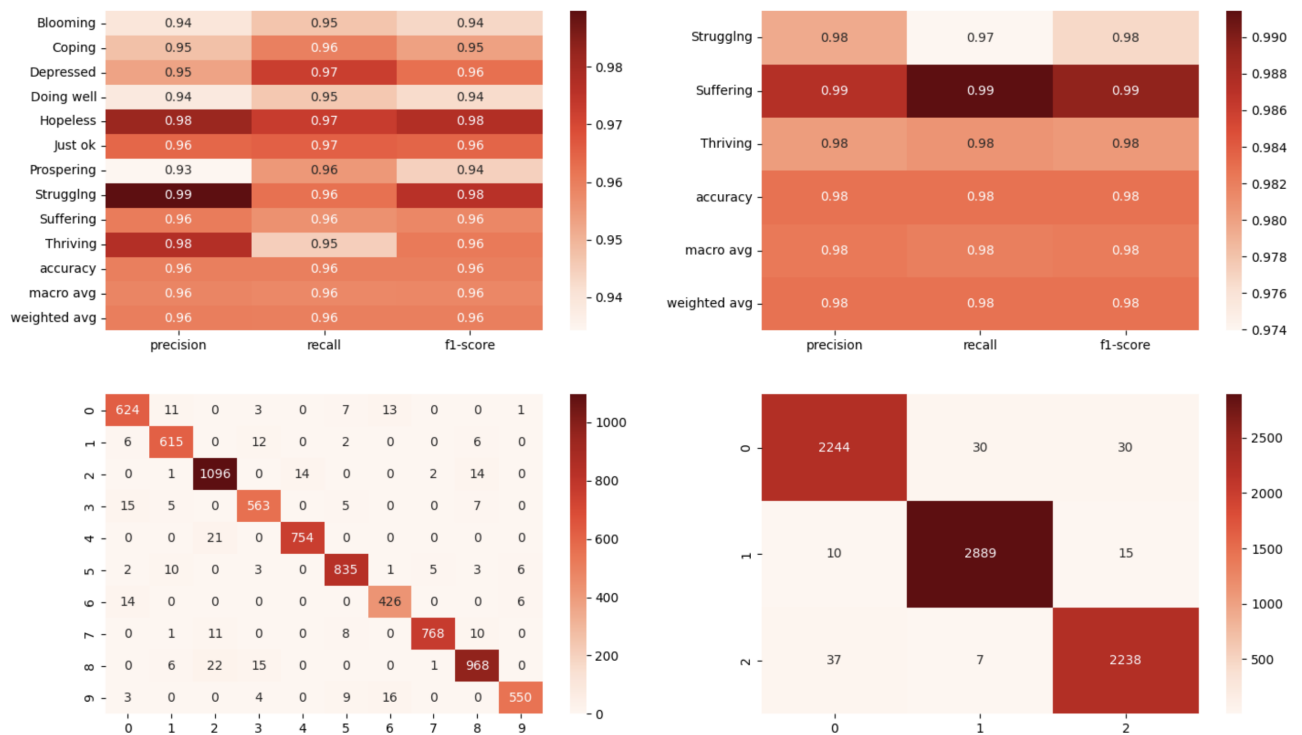


Рисунок 3.5 — Результат классификации с помощью AdaBoost над деревьями регрессии (MCC: 0.955)

# ЗАКЛЮЧЕНИЕ

В рамках лабораторной работы было проведено изучение деревьев решений и ансамблевых классификаторов на примере анализа социологического исследования.

1. Определено, какие из признаков состояния наиболее сильно связаны с интегральной оценкой счастья (благополучия) респондента;
2. Спрогнозировано попадание респондентов, у которых интегральная характеристика отмечена как «Неизвестно», в укрупнённые группы шкалы Кантрила;
3. Построены следующие классификаторы: многоклассовая логистическая регрессия, дерево решений, ансамблевый классификатор;
4. Проведено сравнение матриц ошибок и метрик качества классификации.

Наиболее точным классификатором оказалась ансамблевая модель AdaBoost над деревьями регрессии, для которой максимальное значение метрики MCC составила 0.955.