Московский государственный технический

университет им. Н.Э. Баумана

Факультет Радиотехнический

Кафедра РТ5

Курс «Технологии машинного обучения»

Отчет по лабораторной работе №6

«Ансамбли машинного обучения. Часть 2.»

Выполнил:

студент группы РТ5-61Б:

Бабасанова Н. С.

Руководитель:

преподаватель каф. ИУ5

Гапанюк Ю.Е.

Москва, 2025г.

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 следующие ансамблевые модели:
   - одну из моделей группы стекинга.
   - модель многослойного персептрона. По желанию, вместо библиотеки scikit-learn возможно использование библиотек TensorFlow, PyTorch или других аналогичных библиотек.
   - двумя методами на выбор из семейства МГУА (один из линейных методов COMBI / MULTI + один из нелинейных методов MIA / RIA) с использованием библиотеки gmdh.
   - **В настоящее время библиотека МГУА не позволяет решать задачу классификации !!!**
5. Оцените качество моделей с помощью одной из подходящих для задачи метрик. Сравните качество полученных моделей.

Текст программы

```
import gmdh
import pandas as pd
import numpy as np
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

```
data = pd.read_csv('/content/sample_data/housing.csv')
data
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY
...	...	...	...	...	...	...	...	...	...	...
20635	-121.09	39.48	25.0	1665.0	374.0	845.0	330.0	1.5603	78100.0	INLAND
20636	-121.21	39.49	18.0	697.0	150.0	356.0	114.0	2.5568	77100.0	INLAND
20637	-121.22	39.43	17.0	2254.0	485.0	1007.0	433.0	1.7000	92300.0	INLAND
20638	-121.32	39.43	18.0	1860.0	409.0	741.0	349.0	1.8672	84700.0	INLAND
20639	-121.24	39.37	16.0	2785.0	616.0	1387.0	530.0	2.3886	89400.0	INLAND

20640 rows × 10 columns

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

	0
longitude	0
latitude	0
housing_median_age	0
total_rooms	0
total_bedrooms	207
population	0
households	0
median_income	0
median_house_value	0
ocean_proximity	0

dtype: int64

```
data = data.dropna(axis=0, how='any')
data
data['ocean_proximity'].unique()
```

```
ohe = OneHotEncoder()
encoded_data = ohe.fit_transform(data[['ocean_proximity']])
```

```
encoded_data = pd.get_dummies(data, columns=['ocean_proximity'],
prefix='ocean').astype(int)
encoded_data
```

```
X = encoded_data.drop(['median_house_value'], axis=1)
y = encoded_data['median_house_value']
```

```
scaler = StandardScaler()
scaler.fit(X)
encoded_data
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
random_state=42)
```

```
from sklearn.ensemble import StackingRegressor
from sklearn.linear_model import Ridge
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR

# Базовые модели
base_models = [
    ('ridge', Ridge()),
    ('tree', DecisionTreeRegressor()),
    ('svr', SVR())
]

# Мета-модель (финальный предсказатель)
meta_model = Ridge()

# Стекинг
stacking_model = StackingRegressor(estimators=base_models,
final_estimator=meta_model)
stacking_model.fit(X_train, y_train)

# Оценка
score = stacking_model.score(X_test, y_test)
```

```
from sklearn.neural_network import MLPRegressor

mlp = MLPRegressor(hidden_layer_sizes=(100, 50), activation='relu',
solver='adam', max_iter=500)
mlp.fit(X_train, y_train)
```

```
from gmdh import Combi, split_data
import numpy as np

X = X.values
y = y.values.ravel()


X_train_combi, X_test_combi, y_train_combi, y_test_combi =
train_test_split(X, y, test_size=0.33, random_state=42)

combi = Combi()
combi.fit(X_train_combi, y_train_combi)
y_pred_com = combi.predict(X_test)
```

```python
from gmdh import Mia
import numpy as np

mia = Mia()
mia.fit(X_train_combi, y_train_combi)
y_pred_mia = mia.predict(X_test_combi)
```

```python
from sklearn.metrics import r2_score

print(f"Score (COMBI): {r2_score(y_test, y_pred_com):.3f}")
print(f"Score (MIA): {r2_score(y_test, y_pred_mia):.3f}")
```

```
Score (COMBI): 0.632
Score (MIA): 0.602
```

```python
print(f"Score (Stacking): {score:.3f}")
print(f"Score (MLP): {mlp.score(X_test, y_test):.3f}")
```

```
Score (MLP): 0.708
Score (Stacking): 0.683
```