Московский государственный технический университет им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

Лабораторная работа №6
по курсу «Методы машинного обучения»
на тему
«Ансамбли моделей машинного обучения.»

Выполнил:

Хотин П.Ю.

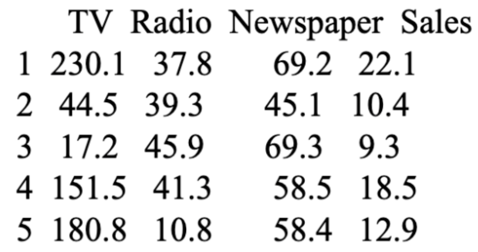
ИУ5-24М

Москва, 2020 год

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor,
GradientBoostingRegressor
from sklearn.metrics import mean_absolute_error,accuracy_score,
r2_score
```

```
data = pd.read_csv("advertising.csv")
```

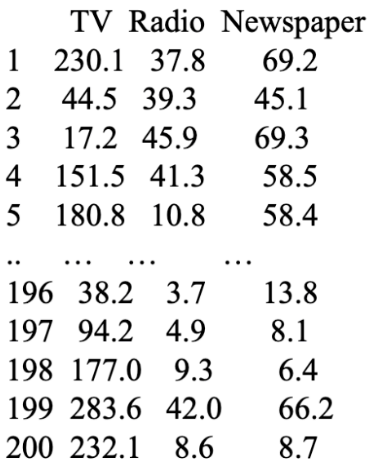
```
data.head()
```



```
     TV  Radio  Newspaper  Sales
1  230.1   37.8      69.2   22.1
2   44.5   39.3      45.1   10.4
3   17.2   45.9      69.3    9.3
4  151.5   41.3      58.5   18.5
5  180.8   10.8      58.4   12.9
```

```
data_X = data[["TV", "Radio", "Newspaper"]]
```

```
data_X
```



```
       TV  Radio  Newspaper
1    230.1   37.8      69.2
2     44.5   39.3      45.1
3     17.2   45.9      69.3
4    151.5   41.3      58.5
5    180.8   10.8      58.4
..     ...    ...       ...
196   38.2    3.7      13.8
197   94.2    4.9       8.1
198  177.0    9.3       6.4
199  283.6   42.0      66.2
200  232.1    8.6       8.7
```

200 rows × 3 columns

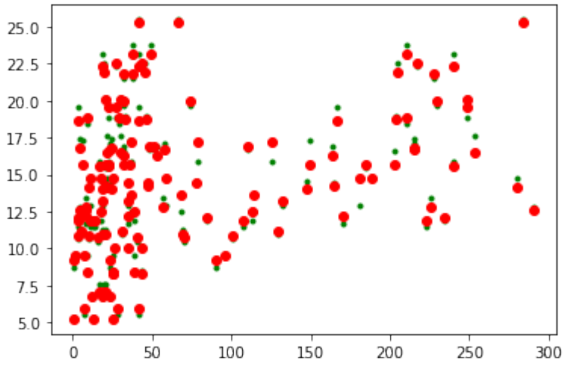
```
data_Y = data[["Sales"]]
```

```
X_train, X_test, y_train, y_test = train_test_split(
    data_X, data_Y, test_size=0.25, random_state=1)
```

```
# Качество отдельных моделей
def val_mae(model):
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    plt.plot(X_test, y_test, 'g.')
    plt.plot(X_test, y_pred, 'ro')
    plt.show()
    result = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    print(model)
    print('MAE={}'.format(result))
    print('R2={}'.format(r2))
```

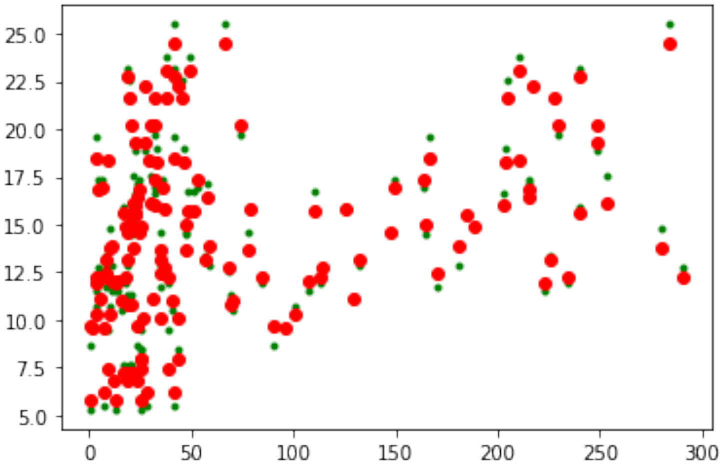
```
for model in [
    GradientBoostingRegressor(),
    RandomForestRegressor(n_estimators=50)
]:
    val_mae(model)
    print('==========================\n\n')
```

```
/usr/local/lib/python3.7/site-packages/sklearn/ensemble/_gb.py:1454:
DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples, ), for
example using ravel().
  y = column_or_1d(y, warn=True)
```



```
/usr/local/lib/python3.7/site-packages/ipykernel_launcher.py:3:
DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples,), for
example using ravel().
  This is separate from the ipykernel package so we can avoid doing
imports until
```

```
GradientBoostingRegressor(alpha=0.9, ccp_alpha=0.0,
criterion='friedman_mse',
                          init=None, learning_rate=0.1, loss='ls',
max_depth=3,
                          max_features=None, max_leaf_nodes=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=100,
                          n_iter_no_change=None, presort='deprecated',
                          random_state=None, subsample=1.0,
tol=0.0001,
                          validation_fraction=0.1, verbose=0,
warm_start=False)
MAE=0.489973091916708 74
R2=0.9831579266623767
==========================
```



```
RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      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=False,
                      random_state=None, verbose=0, warm_start=False)
MAE=0.5131199999999992
R2=0.9815468551914713
==========================
```

Модель градиентного бустинга показала лучший результат на тестовой выборке

```
from sklearn.model_selection import RandomizedSearchCV

n_estimators = [int(x) for x in np.linspace(start = 200, stop = 2000,
num = 10)]

max_features = ['auto', 'sqrt']

max_depth = [int(x) for x in np.linspace(10, 110, num = 11)]
max_depth.append(None)

min_samples_split = [2, 5, 10]

min_samples_leaf = [1, 2, 4]

bootstrap = [True, False]

random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               'bootstrap': bootstrap}
random_grid
```

```
{'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800,
2000],
 'max_features': ['auto', 'sqrt'],
 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, None],
 'min_samples_split': [2, 5, 10],
 'min_samples_leaf': [1, 2, 4],
 'bootstrap': [True, False]}
```

```
rf = RandomForestRegressor()

rf_random = RandomizedSearchCV(estimator = rf, param_distributions =
random_grid, n_iter = 100, cv = 3, verbose=2, random_state=42, n_jobs
= -1)

rf_random.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 16 concurrent
workers.
[Parallel(n_jobs=-1)]: Done   9 tasks      | elapsed:    3.3s
[Parallel(n_jobs=-1)]: Done 130 tasks      | elapsed:   20.3s
[Parallel(n_jobs=-1)]: Done 300 out of 300 | elapsed:   47.8s finished
/usr/local/lib/python3.7/site-packages/sklearn/model_selection/
_search.py:739: DataConversionWarning: A column-vector y was passed
when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
  self.best_estimator_.fit(X, y, **fit_params)
```

```
RandomizedSearchCV(cv=3, error_score=nan,
                   estimator=RandomForestRegressor(bootstrap=True,
                                                   ccp_alpha=0.0,
                                                   criterion='mse',
                                                   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=100,
                                                   n_jobs=None,
oob_score=Fals...
                   param_distributions={'bootstrap': [True, False],
                                        'max_depth': [10, 20, 30, 40,
50, 60,
                                                      70, 80, 90, 100,
110,
                                                      None],
                                        'max_features': ['auto',
'sqrt'],
                                        'min_samples_leaf': [1, 2, 4],
                                        'min_samples_split': [2, 5,
10],
                                        'n_estimators': [200, 400,
600, 800,
                                                         1000, 1200,
1400, 1600,
                                                         1800, 2000]},
                   pre_dispatch='2*n_jobs', random_state=42,
refit=True,
                   return_train_score=False, scoring=None, verbose=2)

rf_random.best_params_
```

```
{'n_estimators': 800,
 'min_samples_split': 2,
 'min_samples_leaf': 1,
 'max_features': 'auto',
 'max_depth': 100,
 'bootstrap': True}
```

```
def evaluate(model, test_features, test_labels):
    predictions = model.predict(test_features)
    error = mean_absolute_error(y_test, predictions)
    r2 = r2_score(y_test, predictions)
    print('Model Performance')
    print('MAE: {:0.4f}'.format(error))
    print('R2 score: {:0.4f}'.format(r2))
    print('======================\n\n')

base_model = RandomForestRegressor(n_estimators = 10, random_state =
42)
base_model.fit(X_train, y_train)
evaluate(base_model, X_test, y_test)
```

```
Model Performance
MAE: 0.5994
R2 score: 0.9713
======================
```

```
/usr/local/lib/python3.7/site-packages/ipykernel_launcher.py:11:
DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples,), for
example using ravel().
  # This is added back by InteractiveShellApp.init_path()
```

```
best_random = rf_random.best_estimator_
evaluate(best_random, X_test, y_test)
```

```
Model Performance
MAE: 0.5178
```

```
R2 score: 0.9820
======================
```

Видно, что подбор гиперпараметров улучшил нашу модель, уменьшив ошибку на 0.08

```
n_estimators = [int(x) for x in np.linspace(start = 200, stop = 2000,
num = 10)]

max_features = ['auto', 'sqrt']

max_depth = [int(x) for x in np.linspace(10, 110, num = 11)]
max_depth.append(None)

min_samples_split = [2, 5, 10]

min_samples_leaf = [1, 2, 4]

bootstrap = [True, False]

random_grid_Booster = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               }

gb = GradientBoostingRegressor()

gb_random = RandomizedSearchCV(estimator = gb, param_distributions =
random_grid_Booster, n_iter = 100, cv = 3, verbose=2, random_state=42,
n_jobs = -1)

gb_random.fit(X_train, y_train)
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 16 concurrent
workers.


Fitting 3 folds for each of 100 candidates, totalling 300 fits
```

```
[Parallel(n_jobs=-1)]: Done   9 tasks      | elapsed:    0.4s
[Parallel(n_jobs=-1)]: Done 221 tasks      | elapsed:    7.7s
[Parallel(n_jobs=-1)]: Done 300 out of 300 | elapsed:   10.2s finished
/usr/local/lib/python3.7/site-packages/sklearn/ensemble/_gb.py:1454:
DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples, ), for
example using ravel().
  y = column_or_1d(y, warn=True)
```

```
RandomizedSearchCV(cv=3, error_score=nan,
                   estimator=GradientBoostingRegressor(alpha=0.9,
ccp_alpha=0.0,

criterion='friedman_mse',
                                                       init=None,

learning_rate=0.1,
                                                       loss='ls',
max_depth=3,

max_features=None,

max_leaf_nodes=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=100,
                                                       n_...
                   iid='deprecated', n_iter=100, n_jobs=-1,
                   param_distributions={'max_depth': [10, 20, 30, 40,
50, 60,
                                                      70, 80, 90, 100,
110,
                                                      None],
                                        'max_features': ['auto',
'sqrt'],
                                        'min_samples_leaf': [1, 2, 4],
                                        'min_samples_split': [2, 5,
10],
                                        'n_estimators': [200, 400,
600, 800,
                                                         1000, 1200,
1400, 1600,
                                                         1800, 2000]},
                   pre_dispatch='2*n_jobs', random_state=42,
refit=True,
                   return_train_score=False, scoring=None, verbose=2)
```

```
gb_random.best_params_
```

```
{'n_estimators': 1400,
 'min_samples_split': 10,
 'min_samples_leaf': 2,
 'max_features': 'auto',
 'max_depth': 40}
```

```
def evaluate(model, test_features, test_labels):
    predictions = model.predict(test_features)
    error = mean_absolute_error(y_test, predictions)
    r2 = r2_score(y_test, predictions)
    print('Model Performance')
    print('MAE: {:0.4f}'.format(error))
    print('R2 score: {:0.4f}'.format(r2))
    print('=====================\n\n')

base_model = GradientBoostingRegressor()
```

```
base_model.fit(X_train, y_train)
evaluate(base_model, X_test, y_test)
```

```
Model Performance
MAE: 0.4890
R2 score: 0.9832
======================
```

```
/usr/local/lib/python3.7/site-packages/sklearn/ensemble/_gb.py:1454:
DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples, ), for
example using ravel().
  y = column_or_1d(y, warn=True)
```

```
best_random = gb_random.best_estimator_
evaluate(best_random, X_test, y_test)
```

```
Model Performance
MAE: 0.5078
R2 score: 0.9839
======================
```

Подбор параметров в градиентном бустинге не дал прироста качества (оно и так в целом было достаточно высокое)