**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:**

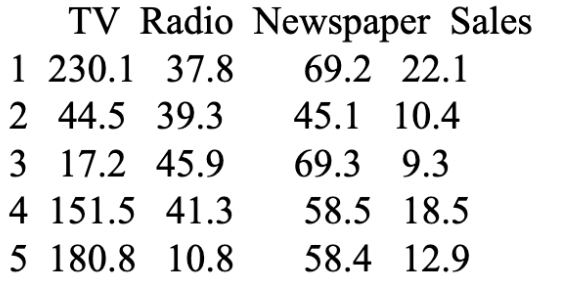
**Хотин П.Ю.**

**ИУ5-24М**

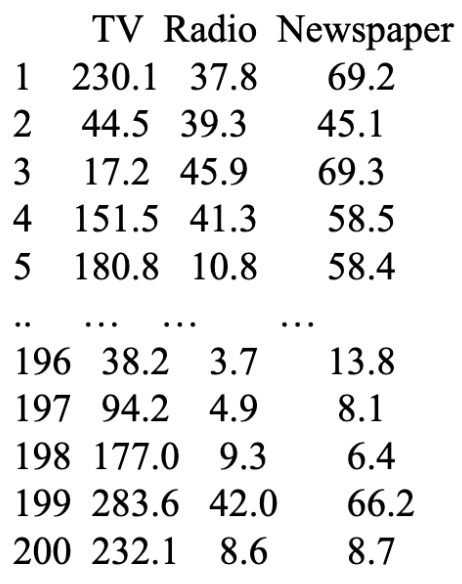
Москва, 2020 год

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()

data\_X = data[["TV", "Radio", "Newspaper"]]

data\_X

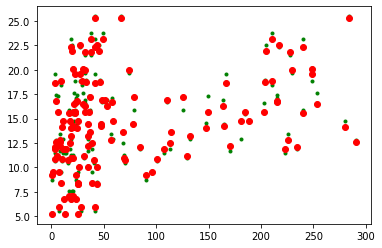
200 rows × 3 columns

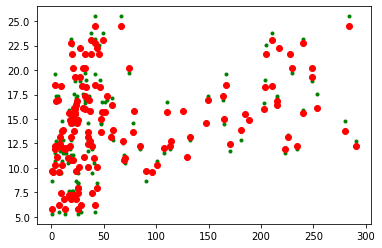
data\_Y = data[["Sales"]]

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=0.48997309191670874  
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)