



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» (ИУ)  
КАФЕДРА «Информационная безопасность» (ИУ8)

## ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3 (3.3) МАШИННОЕ ОБУЧЕНИЕ

Студент

ИУ8-92  
(Группа)

Лагов С. П.  
(И. О. Фамилия)

Преподаватель:

Коннова Н.С.  
(И.О. Фамилия)

Цель работы:

Познакомиться с основными приемами работы с моделями регрессии в scikit-learn.

Ход работы:

1. Загрузите встроенный датасет о ценах на недвижимость в Калифорнии.

### 3.3 регрессия

```
california = fetch_california_housing()
type(california)
```

[126]

```
... sklearn.utils._bunch.Bunch
```

```
california.keys()
```

[127]

```
... dict_keys(['data', 'target', 'frame', 'target_names', 'feature_names', 'DESCR'])
```

```
print(type(california.data), california.data.shape)
print(type(california.target), california.target.shape)
```

[128]

```
... <class 'numpy.ndarray'> (20640, 8)
<class 'numpy.ndarray'> (20640,)
```

```
data = pd.DataFrame(california.data, columns = california.feature_names)
data['Price'] = california.target
data.head()
```

[129]

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	Price
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422

▷ ▾

```
data.describe().round(2)
```

[130]

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	Price
count	20640.00	20640.00	20640.00	20640.00	20640.00	20640.00	20640.00	20640.00	20640.00
mean	3.87	28.64	5.43	1.10	1425.48	3.07	35.63	-119.57	2.07
std	1.90	12.59	2.47	0.47	1132.46	10.39	2.14	2.00	1.15
min	0.50	1.00	0.85	0.33	3.00	0.69	32.54	-124.35	0.15
25%	2.56	18.00	4.44	1.01	787.00	2.43	33.93	-121.80	1.20
50%	3.53	29.00	5.23	1.05	1166.00	2.82	34.26	-118.49	1.80
75%	4.74	37.00	6.05	1.10	1725.00	3.28	37.71	-118.01	2.65
max	15.00	52.00	141.91	34.07	35682.00	1243.33	41.95	-114.31	5.00

2. Постройте модель регрессии для предсказания цены конкретного объекта. Оцените качество построенной модели с помощью визуализации и коэффициента детерминации.

Доп. задание - Напишите функцию, которая автоматически обучает все перечисленные модели и для каждой выдает оценку точности.

```
accuracies = {}

def predict(model, x_train, y_train, x_test, y_test, method, show_coefs=False):
    model.fit(x_train, y_train)
    if show_coefs:
        print("Coefs: \n", model.coef_)
        _ = [print(k, v) for k, v in zip(x_train.columns, model.coef_)]
        print("Intercept: \n", model.intercept_)

    score = model.score(x_test, y_test)
    print(f"Score: {score}")

    y_pred = model.predict(x_test)
    plt.scatter(y_pred, y_test)
    plt.plot(y_test, y_test, c='r')
    plt.title(method)
    plt.grid()
    plt.xlabel('y (real)')
    plt.ylabel('y (predicted)')

    global accuracies
    accuracies[method] = score
```

```
table = PrettyTable()
table.field_names = ["Method", "Precision"]
for method in accuracies.keys():
    table.add_row([method, round(accuracies[method], 5)])
print(table)
```

```
predict(LinearRegression(), x_train, y_train, x_test, y_test, 'Linear Regression', show_coeffs=True)
```

Coeffs:

```
[ 4.40204508e-01  9.56823673e-03 -1.14222566e-01  6.82418276e-01  
 -3.32392057e-06 -3.61425838e-03 -4.20361154e-01 -4.33341342e-01]
```

MedInc 0.44020450799314953

HouseAge 0.009568236733306012

AveRooms -0.11422256559923852

AveBedrms 0.6824182755414572

Population -3.323920566899657e-06

AveOccup -0.0036142583762447168

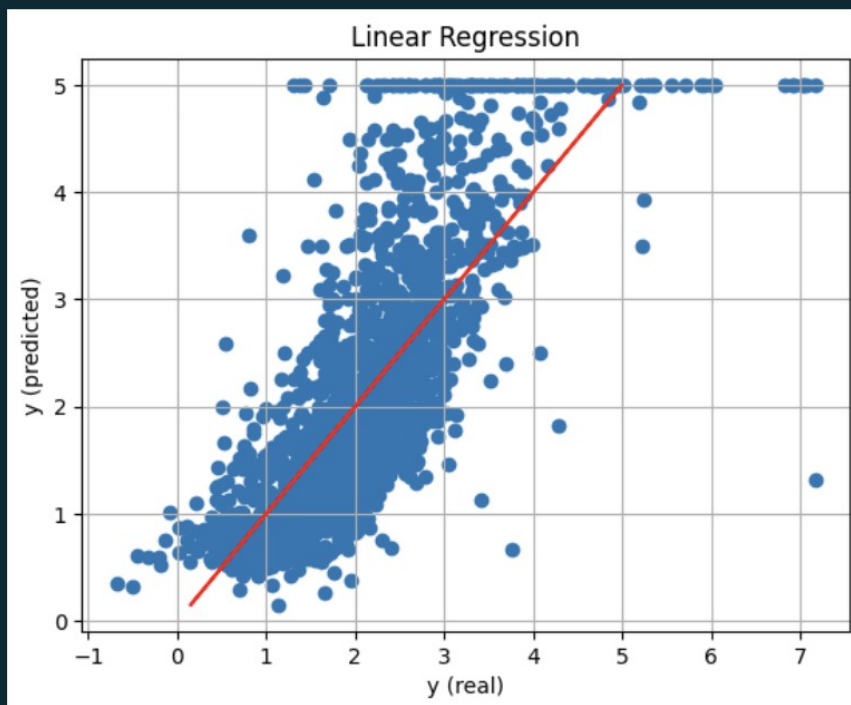
Latitude -0.42036115397719553

Longitude -0.43334134181312267

Intercept:

```
-36.85790997715772
```

Score: 0.5958833949634967

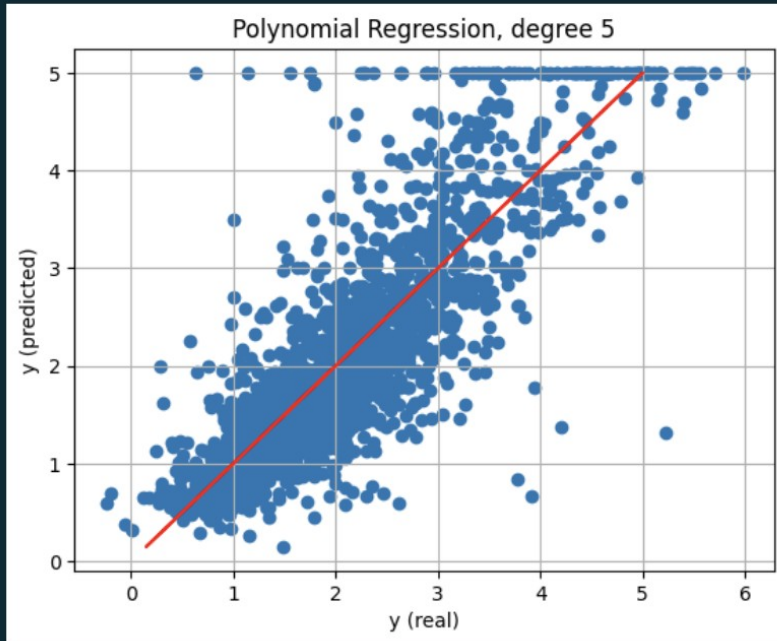


3. Постройте альтернативную полиномиальную модель, сравните ее с предыдущей.

```
degree = 5
poly_x_train = PolynomialFeatures(degree).fit_transform(x_train)
poly_x_test = PolynomialFeatures(degree).fit_transform(x_test)

predict(LinearRegression(), poly_x_train, y_train, poly_x_test, y_test, f'Polynomial Regression, degree {degree}')
```

Score: 0.7161634608924309



Вывод: полиномиальная модель более точно предсказывает значения, чем линейная, это видно по значению параметра score.

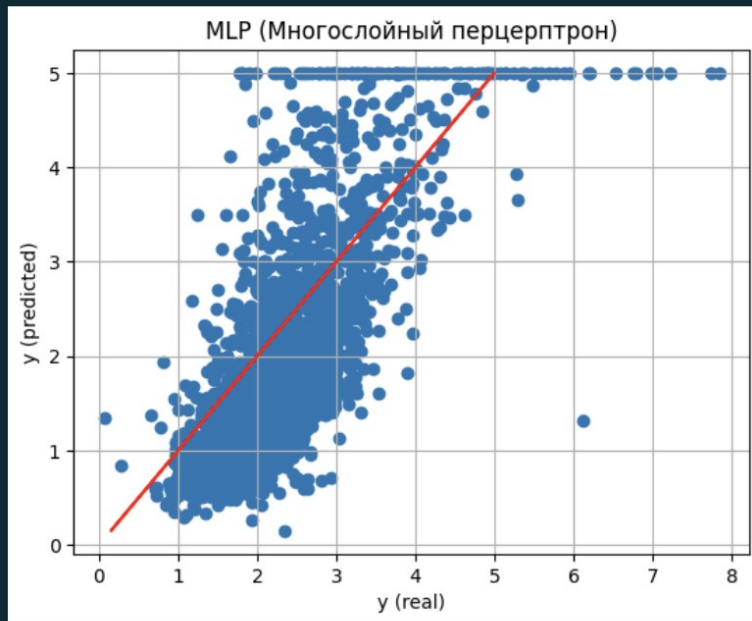
4. Попробуйте применить к той же задаче другие модели регрессии. Для каждой из них выведите матрицу классификации и оценку точности.

Использовались следующие модели:

- Многослойный перцептрон
- Случайный лес
- Пассивно-агрессивный регрессор Ridge

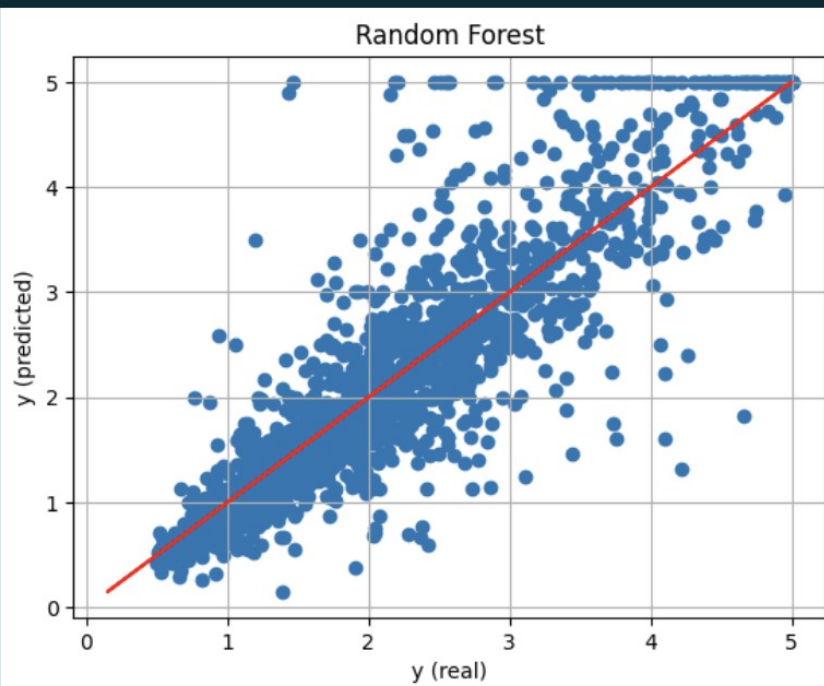
```
predict(MLPRegressor(), x_train.values, y_train.values, x_test.values, y_test.values, 'MLP (Многослойный перцептрон)')
```

Score: 0.4783123691302429



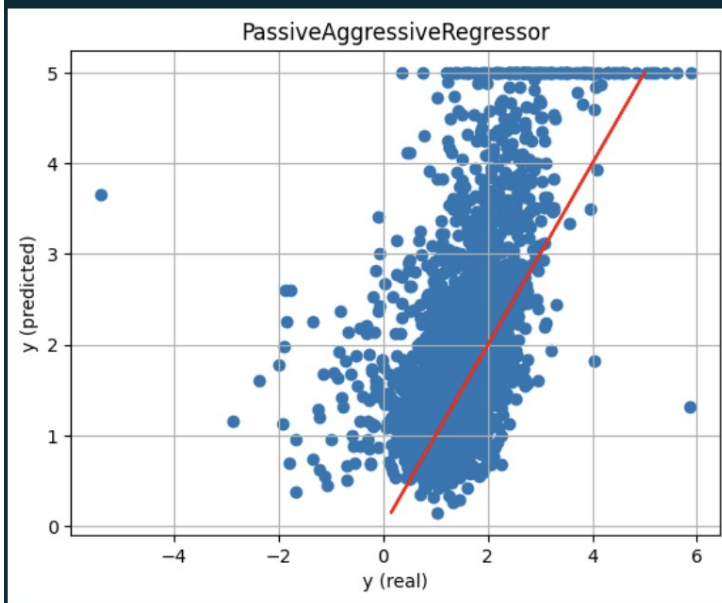
```
predict(RandomForestRegressor(), x_train.values, y_train.values, x_test.values, y_test.values, 'Random Forest')
```

Score: 0.8092524122717131



```
predict(PassiveAggressiveRegressor(), x_train.values, y_train.values, x_test.values, y_test.values, 'PassiveAggressiveRegressor')
```

score: 0.09911216844332149



```
predict(Ridge(), x_train.values, y_train.values, x_test.values, y_test.values, 'Ridge (Гребневый классификатор)')
```

score: 0.5959044695435594

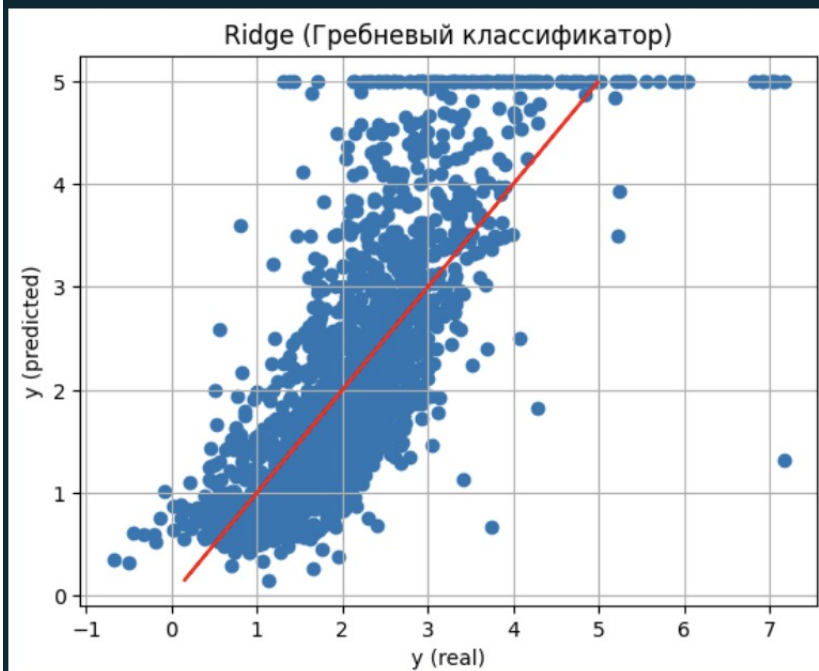




Таблица со значениями коэффициента детерминации:

Method	Precision
Linear Regression	0.59588
Polynomial Regression, degree 5	0.71616
MLP (Многослойный перцептрон)	0.47831
Random Forest	0.80925
Ridge (Гребневый классификатор)	0.5959
PassiveAggressiveRegressor	0.09911

Из всех моделей лучше всего себя показала модель Random Forest

### Ответы на контрольные вопросы

1. Чем отличается применение разных моделей регрессии в библиотеке *sklearn* от моделей классификации?

Главные отличия в целях задач:

- **Регрессия:** Модели регрессии предназначены для предсказания непрерывных значений. Например, это может быть прогнозирование цен на жилье, температуры, продаж и т.д. Основная задача состоит в том, чтобы оценить количественные зависимости между переменными.
  - **Классификация:** Модели классификации предназначены для предсказания категориальных значений. Например, это могут быть задачи бинарной классификации (да/нет) или многоклассовой классификации (определение типа объекта). Задача классификации предполагает, что необходимо разделить данные на различные категории или классы.
2. Что показывает коэффициент детерминации для модели регрессии?



Коэффициент детерминации, обозначаемый как  $R^2$ , показывает, какую долю вариации зависимой переменной объясняет модель. Он имеет следующее значение:

- Если  $R^2 = 1$ : модель идеально объясняет все вариации.
- Если  $R^2 = 0$ : модель не объясняет ни одной вариации (предсказания модели равны среднему значению).
- Если  $R^2 < 0$ : модель хуже, чем простое среднее значение; это указывает на очень плохую подгонку модели.

### 3. Какое значение имеют коэффициенты линейной регрессии?

Коэффициенты линейной регрессии представляют собой весовые коэффициенты для каждого из предикторов в модели. Они показывают, как изменяется зависимая переменная ( $Y$ ) при увеличении независимого переменного ( $X$ ) на одну единицу, при условии, что другие переменные остаются постоянными.

Положительный коэффициент указывает на положительную зависимость, а отрицательный — на отрицательную зависимость.

### 4. Какие атрибуты имеет объект линейной регрессии?

В объекте линейной регрессии (`LinearRegression`) из библиотеки `sklearn` есть несколько важных атрибутов, которые можно использовать после обучения модели:

- **`coef_`**: массив, содержащий коэффициенты (веса) для каждой из независимых переменных.
- **`intercept_`**: срез (свободный член) регрессионной модели, то есть значение  $Y$ , когда все предикторы равны 0.
- **`score(X, y)`**: метод, который возвращает значение  $R^2$  для данных ( $X, y$ ) на основе модели. Используется для оценки качества модели.

- **n\_iter\_:** (доступно только для некоторых моделей с регуляризацией) количество итераций, необходимых для сходимости алгоритма.