Регрессия

1. Постановка задачи

Реализовать линейную, полиномиальную и экспоненциальную регрессию.

2. Используемые данные

Набор искусственно созданных внутри программы двумерных данных с использованием функции random.

3. Решение

Импорт используемых модулей:

import numpy as np - для работы с массивами

import matplotlib.pyplot as plt - для визуализации результатов

from sklearn.linear_model import LinearRegression

from sklearn.preprocessing import PolynomialFeatures

from math import exp

from random import random - для рандомизации вводных данных

Получение искусственных данных:

```
X = np.array([i for i in range(1, 10000)]).reshape(-1, 1)
Y = np.array([i*(1+0.1*random())*exp(i**(1/3)) for i in range(1, 10000)]).reshape(-1, 1)
```

Реализация регрессии:

1. Линейной

```
reg_linear = LinearRegression()
reg_linear.fit(X, Y)
Y_linear = reg_linear.predict(X)
```

2. Полиномиальной

```
preprocess_polynomial = PolynomialFeatures(degree=4)
reg_polynomial = LinearRegression()
X_poly = preprocess_polynomial.fit_transform(X)
reg_polynomial.fit(X_poly, Y)
Y_polynomial = reg_polynomial.predict(X_poly)
```

3. Экспоненциальной

```
expo = np.polyfit(X[:, 0], np.log(Y[:, 0]), 1, w=np.sqrt(Y[:, 0]))

Y_exponential = np.array([])

for x1 in X:

    y = exp(expo[1] + expo[0] * x1[0])

    Y_exponential = np.append(Y_exponential, [y], axis=0)
```

Проверка качества реализованных типов регрессии с помощью метода наименьших квадратов:

```
diff_lin = 0

diff_pol = 0

diff_exp = 0

for i in range(len(Y)):

    diff_lin += (abs(Y[i][0]-Y_linear[i][0]))**2

    diff_pol += (abs(Y[i][0]-Y_polynomial[i][0]))**2

    diff_exp += (abs(Y[i][0]-Y_exponential[i]))**2


print('Linear difference be like', round(diff_lin))

print('Polynomial difference be like', round(diff_pol))

print('Exponential difference be like', round(diff_exp))

print('Biggest of them is', round(max(diff_lin, diff_pol, diff_exp)))

print('Smallest of them is', round(min(diff_lin, diff_pol, diff_exp)))
```

Визуализация полученных данных:

```
plt.figure(figsize=(10, 5))

plt.scatter(X, Y, s=1, c='blue')

plt.plot(X, Y_linear, c='red')

plt.plot(X, Y_polynomial, c='green')

plt.plot(X, Y_exponential c='pink')

plt.title('Regression types comparison')

plt.xlabel('X')

plt.ylabel('Y')
```

plt.show()

Рисунок 1: результаты работы программы (синий - начальные данные, красный - линейная, зелёный - полиномиальная, розовый - экспоненциальная) для данных:

```
X = np.array([i for i in range(1, 10000)]).reshape(-1, 1)
Y = np.array([i*(1+0.1*random())*exp(i**(1/3)) for i in range(1, 1000)]).reshape(-1, 1)
```

Наименьшее отклонение - у экспоненциальной регрессии.

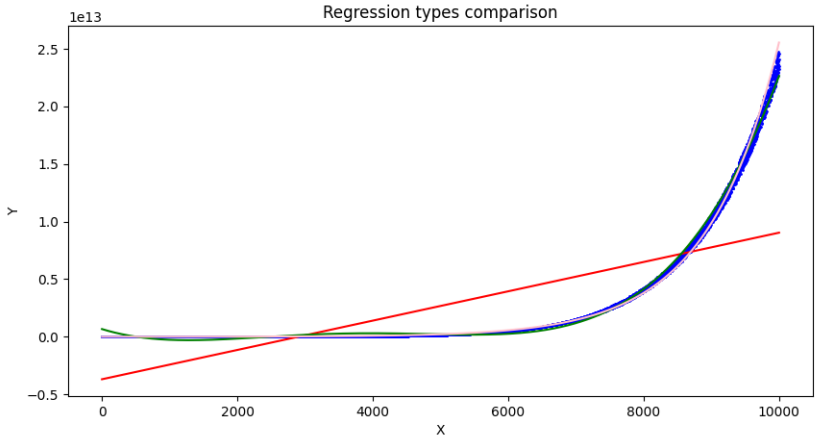


Рисунок 2: результаты работы программы для данных:

```
X = np.array([i for i in range(1, 1000)]).reshape(-1, 1)
Y = np.array([(1+0.2*random())*(i**3) for i in range(1, 1000)]).reshape(-1, 1)
```

Наименьшее отклонение - у полиномиальной регрессии.

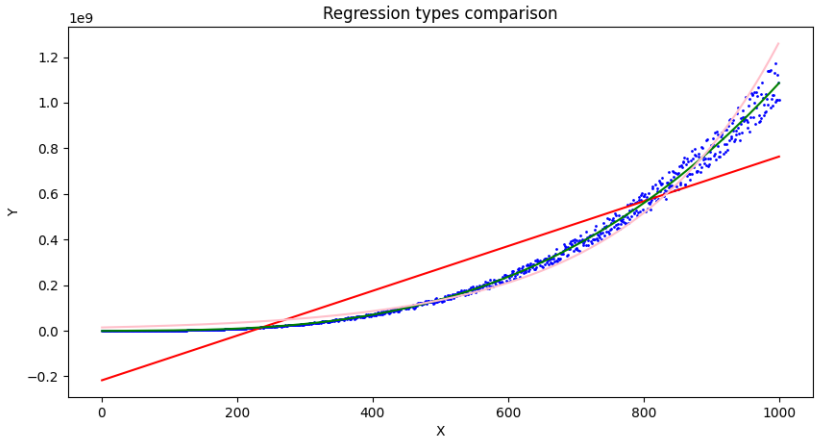
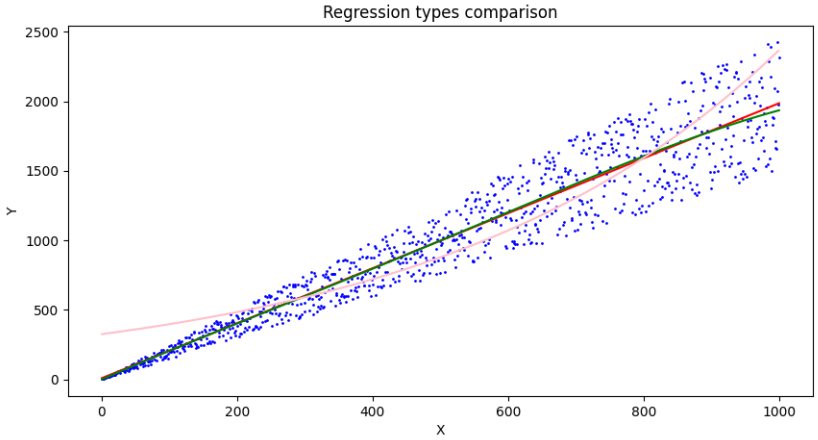


Рисунок 3: результаты работы программы для данных:

```
X = np.array([i for i in range(1, 1000)]).reshape(-1, 1)
Y = np.array([i+i*(1+(random()-0.5)) for i in range(1, 1000)]).reshape(-1, 1)
```

Наименьшее отклонение - у полиномиальной регрессии, но разница с результатом линейной регрессии крайне мала.



4. Выводы

Все три перечисленных типа регрессии реализованы с использованием библиотек.

Для оценки точности в каждом случае подсчитывается квадрат отклонения предполагаемого значения в точке от реального. Лучшие результаты демонстрируют попеременно полиномиальная и экспоненциальная регрессия (линейная - частный случай первой).