# **Лабораторная работа 2. Нелинейная регрессия**

## **Цели**

Приобрести навыки построения моделей нелинейной регрессии

## **Задачи**

1. Построить модель для нелинейной регрессии
2. Оценить точность модели и получить её уравнение

## **Теоретические сведения**

Методические указания для решения поставленного задания

### **Регрессия**

Регрессия – предсказание поведения одной величины в зависимости от поведения другой. Силу зависимости одной величины от другой называют корреляцией. Предсказание строится на основе уравнения параметрами которого являются данные. Например, с помощью такого уравнения можно предсказать наиболее вероятный вес человека, зная его рост и возраст. В данной работе модель будет работать с синтетическими данными которые не имеют интерпретации

Из-за ошибок при сборе данных в набор могут попасть явно выделяющиеся значения – выбросы. Общепринятого метода автоматического удаления выбросов не существует – необходимо проверять каждое значение отдельно. Выбросы могут не только уменьшить значение корреляции, но и испортить уравнение регрессии.

Для того, чтобы рассчитать уравнение линейной регрессии нужно провести следующие расчеты, где – величина выборки, – на основе чего будет делаться предсказание, – эталонное значение, которые нужно предсказать. Уравнение регрессии имеет вид

Для составления уравнения нелинейной регрессии нужно преобразовать и – само уравнение не меняется.

|  |  |  |
| --- | --- | --- |
| Тип регрессии |  |  |
| Линейная |  |  |
| Логарифмическая |  |  |
| Полиномиальная, второй степени |  |  |
| Степенная |  |  |
| Экспоненциальная |  |  |
| Гиперболическая |  |  |

### **Построение модели с помощью scikit-learn**

scikit-learn – популярная библиотека для машинного обучения, включена в дистрибутив Anaconda по умолчанию

Пример построения модели будет приведён ниже, демонстрация будет проводиться на первых 10 000 данных из набора данных [non-linear regression](https://www.kaggle.com/datasets/toubali/nonlinear-regression) (рус. Нелинейная регрессия)

#### **Подключение библиотек**

Подключение библиотек – matplotlib для графиков, numpy для работы с тензорами, seaborn – для красивых графиков, sklearn – другое название scikit-learn, библиотека не импортирует автоматически все свои подмодули, поэтому используются конструкции from sklearn import …

**import** matplotlib.pyplot **as** plt  
**import** numpy **as** np  
**import** pandas **as** pd  
**import** seaborn **as** sns  
  
**from** sklearn.linear\_model **import** LinearRegression  
**from** sklearn.preprocessing **import** PolynomialFeatures  
**from** sklearn.pipeline **import** make\_pipeline

#### **Работа с набором данных**

Подключение набора данных, удаление всех строк за исключением первых десяти тысяч

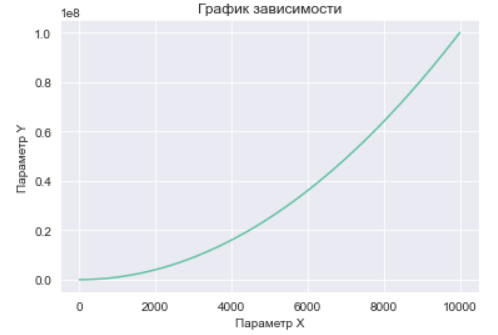
PATH = "regression.csv"  
DATASET\_SIZE = 10000  
  
dataset = pd.read\_csv(PATH)  
dataset = dataset.head(DATASET\_SIZE)

Данные из набора хранятся в строковом виде, что мешает проводить расчеты. Переводу их в целочисленный тип мешает символ \xa0 используемый вместо пробела. Код ниже решает эти проблемы

**for** column **in** dataset.columns:  
 dataset[column] = dataset[column].str.split().str.join("")  
 dataset[column] = pd.to\_numeric(dataset[column])

Код для построения графика по имеющимся данным

sns.set\_style('darkgrid')  
sns.set\_palette('Set2')  
  
sns.lineplot(  
 x=dataset["x"],  
 y=dataset["y"],  
)  
  
plt.title('График зависимости')  
plt.xlabel('Параметр X')  
plt.ylabel('Параметр Y')  
plt.show()



#### **Построение модели**

Модель, предоставляемая библиотекой sklearn принимает на вход только тензоры размерностью (длина набора данных, 1), как в качестве входных данных, так и выходных. Код ниже меняет размерность исходных данных

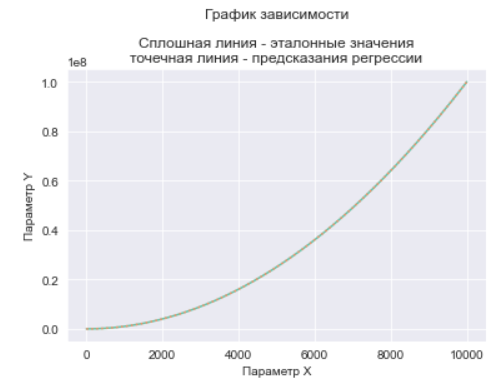
x = np.array(dataset["x"]).reshape(-1, 1)  
y = np.array(dataset["y"]).reshape(-1, 1)

Создание модели, где переменная DEGREES обозначает степень полинома. Тут же рассчитывается среднеквадратическая ошибка предсказания модели.

DEGREES = 2  
  
regression = make\_pipeline(PolynomialFeatures(DEGREES), LinearRegression())  
regression.fit(x, y)  
predictions = regression.predict(x)  
mean\_squared\_error = np.mean((predictions - np.array(y)) \*\* 2)  
print(f'Среднеквадратическая ошибка = {mean\_squared\_error}')

Среднеквадратическая ошибка составила – 4,323661516743e-15. Далее приведен код для составления графика, сравнивающего эталонные значения и предсказания модели

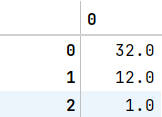
sns.lineplot(  
 x=dataset["x"],  
 y=dataset["y"],  
 linestyle="solid"  
)  
  
sns.lineplot(  
 x=dataset["x"],  
 y=predictions.reshape(-1),  
 linestyle="dotted"  
)  
  
plt.title(  
 'График зависимости\n\n'  
 'Сплошная линия - эталонные значения\n'  
 'точечная линия - предсказания регрессии'  
)  
plt.xlabel('Параметр X')  
plt.ylabel('Параметр Y')  
plt.show()



Графики полностью совпали, что было ожидаемо из полученной среднеквадратической ошибки.

Код для извлечения из модели коэффициентов уравнения

x\_parameters = np.append(  
 regression['linearregression'].intercept\_[0],  
 regression['linearregression'].coef\_[0][1:]  
)  
  
x\_parameters



Из чего следует уравнение

## **Задание**

Выбрать с сайта [kaggle.com](https://www.kaggle.com/) набор данных в формате .csv, пригодный для построения регрессии, загрузить и подготовить его к дальнейшей обработке. Наборы данных не должны повторяться внутри группы. Задание индивидуальное. Требования:

1. Построить модель для нелинейной регрессии
2. Оценить точность модели
3. Сравнить предсказания модели с эталонами, с помощью графиков
4. Построить уравнение регрессии
5. Указать какие знания можно получить из набора
6. Сохранить IPython Notebook

### **Продвинутое задание**

Построить вторую модель, без использования средств scikit-learn