МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЁТ

по лабораторной работе №3

по дисциплине «Машинное обучение»

Тема: Регрессия

Вариант 2Б

Студент гр. 1384 __________________ Шушков Е.В.

Преподаватель __________________ Жангиров Т.Р.

Санкт-Петербург

2024

Цель работы.

В данной лабораторной работе будет изучаться регрессия: линейная и полиномиальная. На практике будет разобрано, как работают разные регрессоры на конкретном наборе данных.

Задание.

1. **Линейная регрессия**

1.1. Загрузите набор данных соответствующей цифре вашего варианта. Убедитесь, что загрузка прошла корректно.

1.2. Используя train_test_split разбейте выборку на обучающую и тестовую. Проверьте, что тестовая выборка соответствует обучающей. *Можно использовать диаграммы рассеяния/нормированные гистограммы/boxplot/violin plot*.

1.3. Проведите линейную регрессию используя LinearRegression. Получите коэффициенты регрессии и объясните полученные результаты.

1.4. Для обучающей и тестовой выборки рассчитайте коэффициент детерминации, MAPE, MAE. Объясните полученные значений метрик. Сравните метрики для обучающей и тестовой выборки, сделайте выводы о качестве обобщения полученной модели.

1.5. Для модели, которая дала лучшие результаты, постройте диаграмму рассеяния между предикторами и откликом. На диаграмме изобразите какое значение должно быть, и какое предсказывается. Визуально оцените качестве построенного регрессора.

2. **Нелинейная регрессия.**

2.1. Загрузите набор данных соответствующей букве вашего варианта. Убедитесь, что загрузка прошла корректно.

2.2. Используя train_test_split разбейте выборку на обучающую и тестовую. Проверьте, что тестовая выборка соответствует обучающей.

2.3. Проверьте работу стандартной линейной регрессии на загруженных данных. Постройте диаграмму рассеяния данных с выделенной полученной линией регрессии. Объясните полученный результат.

2.4. Конструируя полиномиальный признаки для разных степеней полинома найдите степень полинома наилучшим образом аппроксимирующая данные. Постройте график зависимости коэффициента детерминации от степени полинома(на одном графике изобразите линии для обучающей и тестовой выборки отдельно). Сделайте вывод о том, при какой степени полинома модель начинает переобучаться.

2.5. Для выбранной степени полинома, рассчитайте и проанализируйте полученные коэффициенты. Рассчитай значение метрик коэффициент детерминации, MAPE, MAE.

2.6. Для выбранной степени полинома, постройте диаграмму рассеяния данных с линией соответствующей полученному полиному. Сделайте выводы о качестве аппроксимации.

3. Оценка модели регрессии

3.1. Загрузите набор данных Student_Performance.csv . Данный набор данных содержит информацию о характеристиках студента, а также качестве его обучения.

3.2. Проведите предобработку набора данных - замена текстовых данных, удаление null значений, удаление дубликатов. Разделите на обучающую и тестовую выборку.

3.3. Постройте модель, которая будет предсказывать значение признака Performance Index на основе остальных признаков. *Модель выберите самостоятельно.*

3.4. Проанализируйте полученную модель. Сделайте выводы о значимости/информативности признаков. Опишите какие проблемы могут возникнуть при применении модели.

Выполнение работы.

Задание 1. Линейная регрессия.

1.1. Загрузим файл варианта в формате DataFrame из библиотеки Pandas. Проверим корректность загрузки с помощью метода head() (см. Листинг 1.1 и Листинг 1.2).

Листинг 1.1 – Чтение и проверка датасета из файла lab3_lin2.csv

```
dFrameLin = pd.read_csv("E:\LETI2024_EVS\Машинное
обучение\LB3\src\lab3_lin2.csv")
print(dFrameLin.head())
```

Листинг 1.2 – Вывод результата работы метода head()

```
        x1      x2      x3        y
0 -0.7187  0.5925  0.7909  84.4459
1  2.1996  0.8186 -0.2378  -2.7299
2  2.1307  0.4212  0.1537  23.2095
3 -0.1811 -0.2669 -0.3704 -37.1237
4  0.0208 -1.0145  0.0584 -15.5751
```

1.2. Разделим данные на тестовую и тренировочную выборку в соотношении по умолчанию: 25\75 (см. Листинг 1.3). Чтобы удостовериться в том, что выборки образовались корректно, визуализируем их с помощью диаграммы рассеивания (см. Рисунок 1.1). На диаграмме представлена зависимость признака *y* от *x1, x2, x3* по отдельности. Можно сделать предположение, что линейная регрессия будет больше всего зависеть от признака *x3*, т.к. он линейный.

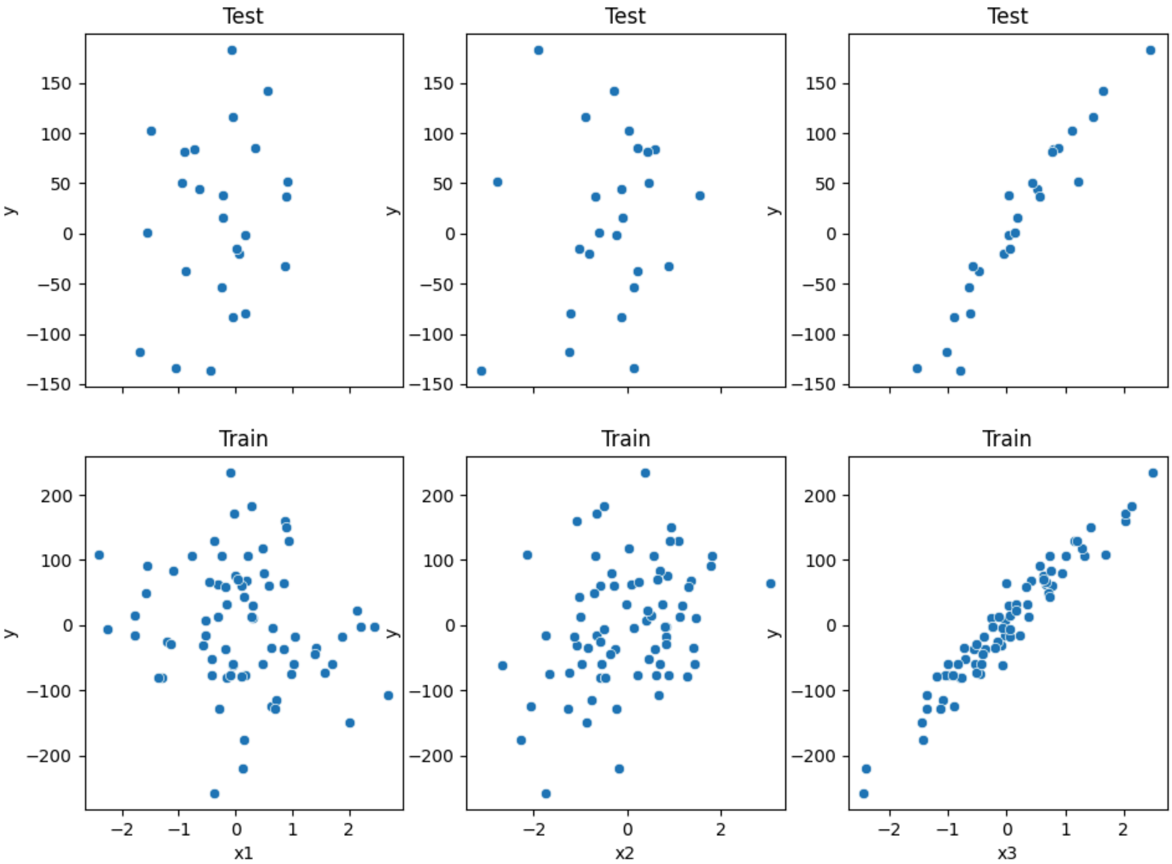

Рисунок 1.1 – Диаграмма рассеивания для тренировочной и тестовой выборки

Листинг 1.3 – Диаграмма рассеивания для выборок

```
fig, axs = plt.subplots(2, 3, sharex='col')

for axIdx in range (3):
    sns.scatterplot(dFrameLinTest, ax = axs[0][axIdx], x = "x" +
str(axIdx + 1), y = "y").set(title="Test")
    sns.scatterplot(dFrameLinTrain, ax = axs[1][axIdx], x = "x" +
str(axIdx + 1), y = "y").set(title="Train")

plt.show()
```

1.3. Перед проведением линейной регрессии разберём, что это такое. Линейная регрессия - вид регрессии, в котором производится поиск линейной зависимости между предикторами и откликом.

В нашем случае (3 признака) регрессионная модель будет иметь следующий вид:

$$f(x,b) = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \varepsilon \tag{1.1}$$

Задача линейной регрессии – минимизировать ошибку, подобрав наиболее подходящие коэффициенты *b*. Обычно для этого используется метод наименьших квадратов.

Проведём линейную регрессию и получим коэффициенты регрессии (см. Листинг 1.4 и Таблицу 1.1). Как видно из таблицы, наибольший коэффициент регрессии у признака *x3*. Это подтверждает наше предположение о том, что этот признак будет больше всего влиять на модель. При этом признак *x1* почти не влияет на регрессию, а признак *x2* влияет, но слабее *x3*.

Таблица 1.1 – Коэффициенты регрессии для признаков

x1	x2	x3
3.07429094e-03	2.13546695e+01	9.04636912e+01

Листинг 1.4 – Линейная регрессия

```
trainX = dFrameLinTrain.iloc[:,0:3]
trainY = dFrameLinTrain.iloc[:,3:4]

testX = dFrameLinTest.iloc[:,0:3]
testY = dFrameLinTest.iloc[:,3:4]

linReg = LinearRegression()

linReg.fit(trainX, trainY)
predictTestY = linReg.predict(testX)

dFrameLinPredict = testX.reset_index(drop=
True).assign(y=pd.DataFrame(predictTestY))

print("Коэффициенты регрессии: ", linReg.coef_)
```

1.4. Дадим определение коэффициентам, которые нас просят найти:

- Коэффициент детерминации R_2:

$$R^2 = 1 - \frac{SSE}{SST}, SST = \sum_{i=1}^{N}(y_i - \overline{y})^2 \quad , \qquad (1.2, 1.3)$$

где:

$$SSE = \sum_{i=1}^{N}(y_i - f(x_i, b))^2 = \sum_{i=1}^{N} err_i \qquad (1.4)$$

SSE – ошибка всей модели, SST – остаточная сумма отклонений.

R2 изменяется от 0 до 1, и равен 1 если модель хорошо приближает

- Средняя абсолютная ошибка:

$$MAE = 1/N \cdot \sum |y - \widehat{y}| \qquad (1.5)$$

Измеряется в тех же величинах, что и отклик.

- Средняя абсолютная ошибка:

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i-\widehat{y}|}{|y_i|} \tag{1.6}$$

Найдём эти коэффициенты для нашей линейной регрессии (см. Листинг 1.5 и Таблицу 1.2). Исходя из таблицы можно сказать, что линейная регрессия прошла хорошо. Т. к. R_2 почти равен единице, можно сказать, что почти идеально. Метрики MAE и MAPE малы, что также подтверждает это.

Это верно для обеих выборок (тестовой и тренировочной). Если сравнивать их показатели, то они отличаются незначительно. Можно утвердить, что переобучения не произошло.

Таблица 1.2 – Вывод метрик для линейной регрессии

	R2	MAE	MAPE
Train	0.999947757313	0.512250637582	0.025580427009
Test	0.999950589543	0.485401980669	0.046121932036

Листинг 1.5 – Метрики для линейной регрессии

```
TrainR2 = r2_score(trainY, predictTrainY)
TestR2 = r2_score(testY, predictTestY)
print("R2 (train - test): ", TrainR2, TestR2)

TrainMAE = mean_absolute_error(trainY, predictTrainY)
TestMAE = mean_absolute_error(testY, predictTestY)

print("MAE (train - test): ", TrainMAE, TestMAE)

TrainMAPE = mean_absolute_percentage_error(trainY, predictTrainY)
TestMAPE = mean_absolute_percentage_error(testY, predictTestY)

print("MAPE (train - test): ", TrainMAPE, TestMAPE)
```

1.5. Построим диаграмму рассеивания для предиката и отклика линейной регрессии (см. Листинг 1.6 и Рисунок 1.2). Визуально линейная регрессия прошла успешно. У некоторых точек имеется отклонение, но оно незначительное.

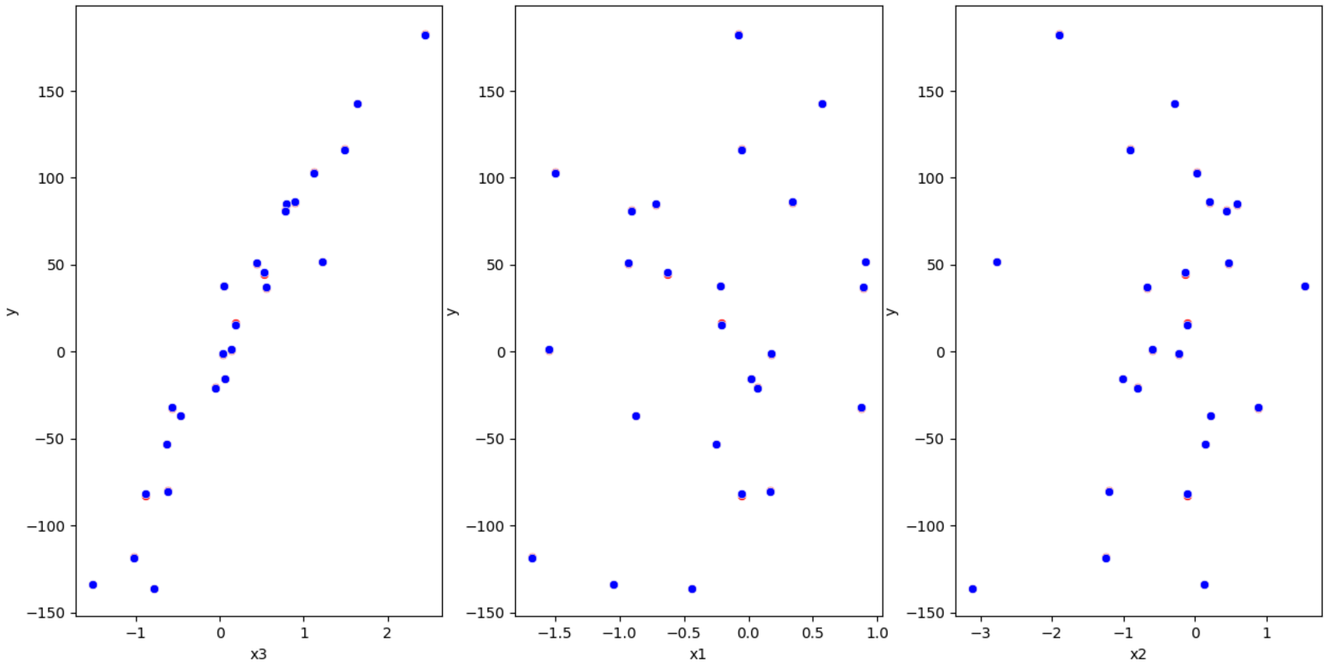

Рисунок 1.2 – Диаграммы рассеивания для линейной регрессии (синий – отклик, красный – предикат)

Листинг 1.6 – Построение диаграмм рассеивания

```
fig, axs = plt.subplots(1, 3, sharex='col')

for axIdx in range (3):
    sns.scatterplot(dFrameLinTest, ax = axs[axIdx-2], x = "x" +
str(axIdx + 1), y = "y", color="red")
    sns.scatterplot(dFrameLinPredict, ax = axs[axIdx-2], x = "x" +
str(axIdx + 1), y = "y", color="blue")

plt.show()
```

Задание 2. Нелинейная регрессия.

2.1. Загрузим файл варианта в формате DataFrame из библиотеки Pandas. Проверим корректность загрузки с помощью метода head() (см. Листинг 2.1 и Листинг 2.2). В отличие от прошлого датасета, здесь мы будем искать зависимость y только от одного признака x.

Листинг 2.1 – Чтение и проверка датасета из файла lab3_poly2.csv

```
dFramePoly = pd.read_csv("E:\LETI2024_EVS\Машинное
обучение\LB3\src\lab3_poly2.csv")
print(dFramePoly.head())
```

Листинг 2.2 – Вывод результата работы метода head()

```
        x       y
0  1.2429  0.2452
1 -0.6314 -1.0334
2  0.9256  1.9695
3  0.6894  0.3605
4 -0.1864 -0.1788
```

2.1. Аналогичным п. 1.2 способом разобьём выборку на тестовую и тренировочную. Проверим получившееся разбиение с помощью графика рассеяния (см. Рисунок 2.1). Визуально разделение выборки прошло корректно: обе имеют одинаковую нелинейную форму.

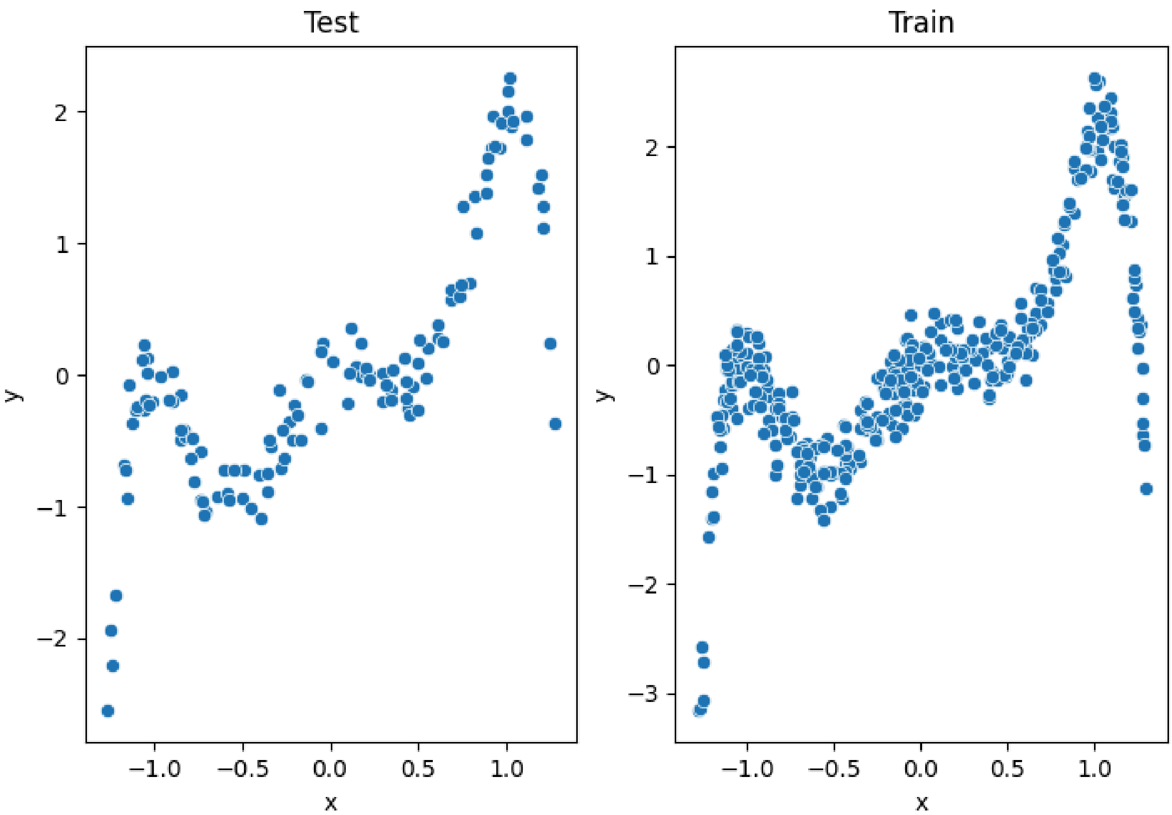

Рисунок 2.1 – Диаграммы рассеивания для тренировочной и тестовой выборки

2.3. Проверим стандартную линейную регрессию, используемую в п. 1.3 на этом наборе данных. Т. к. датасет не линейный, то регрессия должна получиться совершенно неверной. Результат представлен на Рисунке 2.2. Предположение подтверждается: мы получили линейное приближение, которое плохо описывает форму нашей изначальной выборки.

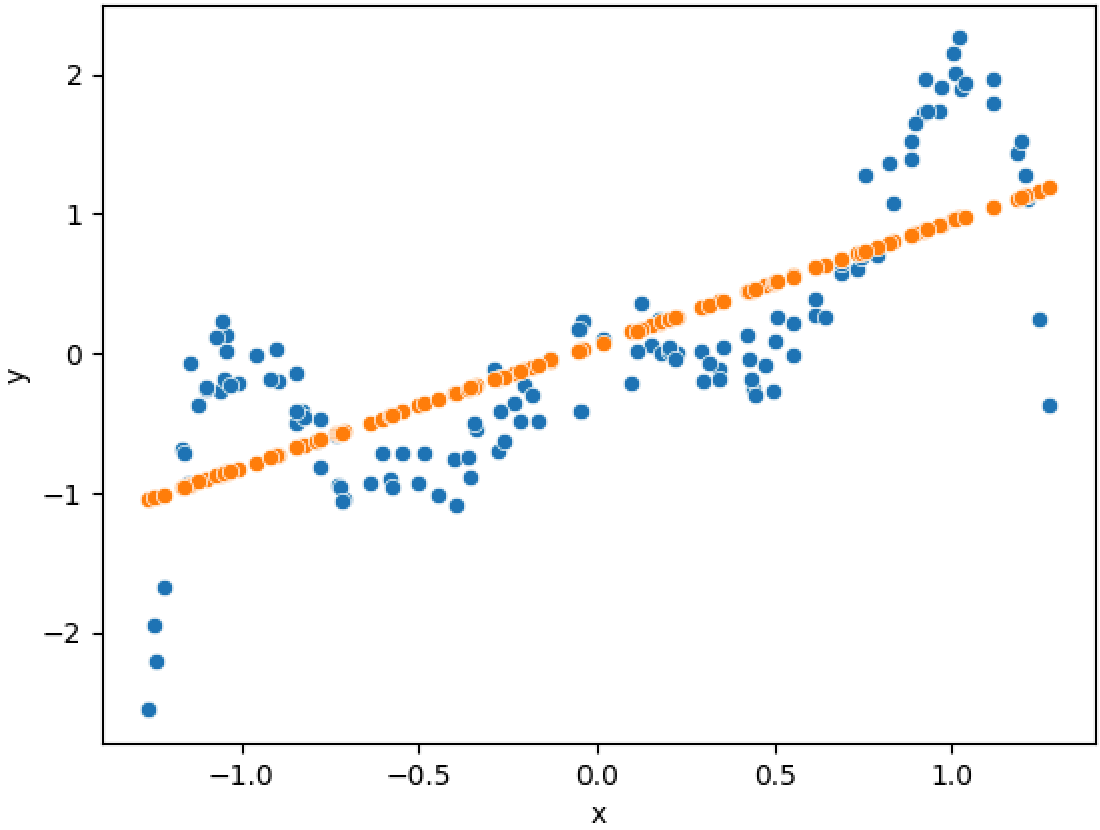

Рисунок 2.2 – Диаграмма рассеивания для линейной регрессии тренировочной выборки и самой выборки

2.4. Чтобы правильно аппроксимировать нашу выборку, сконструируем полиномиальные признаки. Построение этих признаков работает следующим образом:

Линейная регрессия имеет вид согласно Формуле 1.1, т. е. является линейной комбинацией параметров. Таким образом, мы можем переписать её в следующем виде:

$$y = f(b,x) = b_0 + b_1c_1(x) + b_2c_2(x) + \dots \quad (2.1)$$

То есть коэффициент возле параметра можно представить как функцию, зависящую от предиктора

Если предположить, что $c_1(x) = x$, $c_2(x) = x^2$ и так далее ($c_n(x) =$

x^n), то уравнение линейной регрессии от одного предиктора можно представить как:

$$y = f(b,x) = b_0 + b_1x + b_2x^2 + \ldots + b_nx^n$$

Таким образом, полиномиальная регрессия может быть решена как линейная.

Чтобы правильно аппроксимировать данные, нужно подобрать такую степень полинома, при которой нет переобучения, но при этом регрессия имеет высокую точность. Воспользуемся коэффициентом детерминации R_2, чтобы определить эту точность (см. Рисунок 2.3 и Листинг 2.3).

По графику видно, что самая точная степень полинома, которая нам подходит – это 6. После неё уже идёт переобучение.

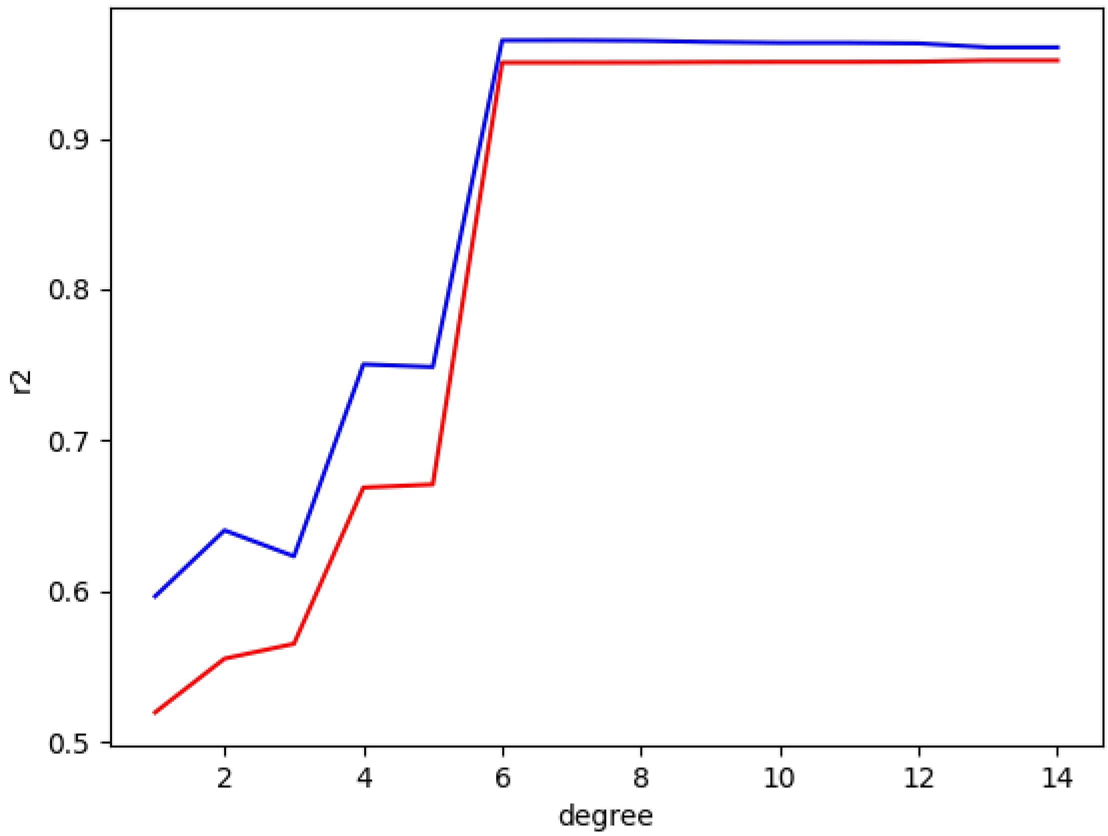

Рисунок 2.2 – График зависимости R_2 от степени полинома

Листинг 2.3 – Построение графика зависимости R_2 от степени полинома

```
def PolyFeatures(k):
    poly = PolynomialFeatures(degree=k)
    polyTrainX = poly.fit_transform(trainX)
    polyTestX = poly.transform(testX)

    linReg = LinearRegression()
    linReg.fit(polyTrainX, trainY)

    predictTestY = linReg.predict(polyTestX)
    predictTrainY = linReg.predict(polyTrainX)

    TrainR2 = r2_score(trainY, predictTrainY)
    TestR2 = r2_score(testY, predictTestY)
```

Листинг 2.3 – Построение графика зависимости R_2 от степени полинома

```
    return [TrainR2, TestR2]

K = [k for k in range(1, 15)]

sns.lineplot(pd.DataFrame([PolyFeatures(k)[0] for k in K],
columns=["r2"]).assign(degree=K), x = "degree", y = "r2",
color="red")
sns.lineplot(pd.DataFrame([PolyFeatures(k)[1] for k in K],
columns=["r2"]).assign(degree=K), x = "degree", y = "r2",
color="blue")

plt.show()
```

2.5. Используя эту степень полинома построим полиномиальные признаки для тестовой и тренировочной выборки, а также вычислим необходимые метрики (см. Таблицу 2.1).

Как мы можем понять из таблицы, регрессия прошла хорошо, значения R_2 для тестовой и тренировочной выборки близки к единице. Метрики MAE и MAPE также дают хорошие результаты, которые не сильно больше, чем у линейной регрессии из п. 1. Это говорит о том, что приближение полиномом достаточно точно аппроксимировало нашу выборку.

Таблица 2.1 – Вывод метрик для линейной регрессии с полиномиальными признаками

	R2	MAE	MAPE
Train	0.950347624439	0.162855413455	0.979151008700
Test	0.965033548513	0.144585977955	1.205445727765

2.6. Построим диаграмму рассеяния для тестовой выборки, а также линию полинома, который у нас получился (см. Листинг 2.4 и Рисунок 2.3). Визуально линия хорошо аппроксимирует изначальную тестовую выборку. Можно утверждать, что регрессия прошла успешно.

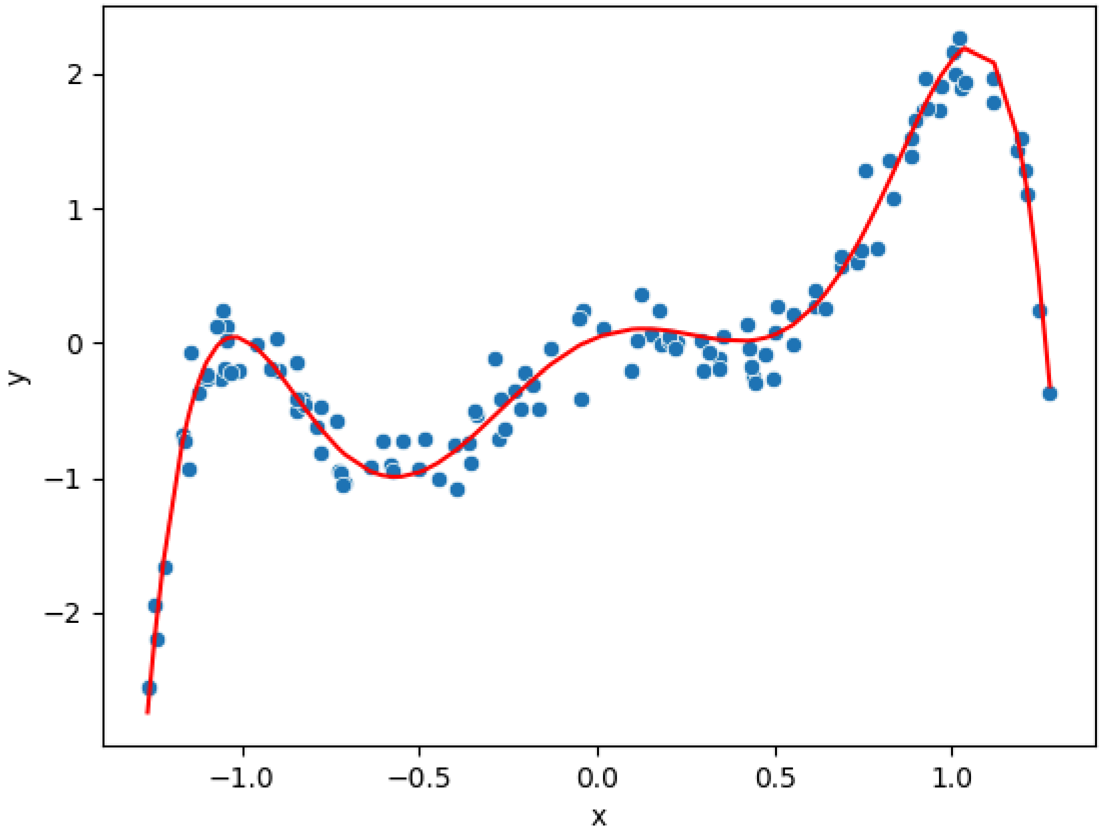

Рисунок 2.2 – График рассеяния для тестовой выборки и линия полученного полинома

Листинг 2.4 – Построение графика рассеяния и полинома

```
dFramePolyPredict = testX.reset_index(drop=
True).assign(y=pd.DataFrame(predictTestY))
sns.lineplot(dFramePolyPredict, x = "x", y = "y", color="red")
sns.scatterplot(dFramePolyTest, x = "x", y = "y")

plt.show()
```

Задание 3. Оценка модели регрессии.

3.1. Загрузим файл варианта в формате DataFrame из библиотеки Pandas. Проверим корректность загрузки с помощью метода head() (Листинг 3.1). В данном датасете 6 признаков, нам надо будет найти зависимость “Performance Index” (имеет бинарные значения) от остальных пяти.

Листинг 1.2 – Вывод результата работы метода head()

```
Hours Studied  Previous Scores Extracurricular Activities  Sleep Hours
7               99                         Yes                   9
4               82                         No                    4
8               51                         Yes                   7
5               52                         Yes                   5
7               75                         No                    8
\\
Sample Question Papers Practiced   Performance Index
                 1                      91.0
                 2                      65.0
                 2                      45.0
                 2                      36.0
                 5                      66.0
```

3.2. Предобработаем данные перед регрессией. Для этого сначала заменим текстовые значения в столбце “Extracurricular Activities” на числовые с помощью LabelEncoder (см. Листинг 3.3, стр. 1–5, Листинг 3.4), отбросим строки-дубликаты (см. Листинг 3.3, стр. 7, Листинг 3.5), а также проверим, что в выборке нет Null значений с помощью метода info() (см. Листинг 3.3, стр. 9, Листинг 3.6).

Разделим выборку на тестову и тренировочную (см. Листинг 3.4, стр. 11–12).

Листинг 3.3 – Предобработка данных

```
le = LabelEncoder()
le.fit(dFrameStud["Extracurricular Activities"])

dFrameStud["Extracurricular Activities"] =
le.transform(dFrameStud["Extracurricular Activities"])

dFrameStud = dFrameStud.drop_duplicates()

print(dFrameStud.info())

dFrameStudTrain, dFrameStudTest = train_test_split(dFrameStud,
random_state=42)
```

Листинг 3.4 – head() для столбца “Extracurricular Activities” после использования LabelEncoder()

```
    Extracurricular Activities
0                1
1                0
2                1
3                1
4                0
```

Листинг 3.5 – len(dFrameStud) до и после удаления дубликатов

```
Количество записей до удаления дубликатов:  10000
Количество записей после удаления дубликатов:  9873
```

Листинг 3.6 – info() для выборки

```
<class 'pandas.core.frame.DataFrame'>
Index: 9873 entries, 0 to 9999
Data columns (total 6 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   Hours Studied                     9873 non-null   int64
 1   Previous Scores                   9873 non-null   int64
 2   Extracurricular Activities        9873 non-null   int32
 3   Sleep Hours                       9873 non-null   int64
 4   Sample Question Papers Practiced  9873 non-null   int64
 5   Performance Index                 9873 non-null   float64
dtypes: float64(1), int32(1), int64(4)
memory usage: 501.4 KB
None
```

3.3. Воспользуемся обычной линейной регрессией, используемой в п.1. Коэффициенты линейной регрессии для этого набора данных представлены в Таблице 3.1. По таблице можно сказать, что на оценку больше всего повлияло время, которое ученик потратил на обучение, что с точки зрения реальности звучит логично. Также сильнее по отношению к другим признакам повлияло на итоговую оценку – оценка на предыдущем тесте, что также выглядит в реальной жизни как прямая корреляция, пусть и не такая явная.

Таблица 3.1 – Коэффициенты линейной регрессии

Hours Stud.	Prev. Scores	Extra Act.	Sleep Hours	SQPP
2.85364318	1.01801988	0.57797717	0.4715297	0.19003358

3.3. Чтобы оценить регрессию вычислим метрики, которые мы также вычисляли в п.1 и п.2 (см. Таблицу 3.2). Несмотря на сложность модели из-за большого количества параметров, мы всё равно получили хорошую регрессию. Коэффициент детерминации даже больше, чем в п. 2, а MAE и MAPE показывают незначительное отклонение, что лишь подтверждает, что приближение прошло успешно.

Чтобы визуально оценить приближение построим диаграмму рассеяния для всех признаков по отношению к Performance Index (см. Рисунок 3.1).

Как можно увидеть визуально имеются некоторые отклонения, но они не критичны. Это подтверждает, что обучение прошло успешно и может быть использовано в реальных условиях для вычисления примерных баллов на тесте.

Также график подтверждает, что вышеупомянутые признаки и правду влияют сильнее других, т. к. у них прослеживается линейная зависимость от оценки.

Таблица 3.2 – Вывод метрик для линейной регрессии

	R2	MAE	MAPE
Train	0.988797946763	1.614528352850	0.034602701572
Test	0.988314921247	1.648633480452	0.034872457385

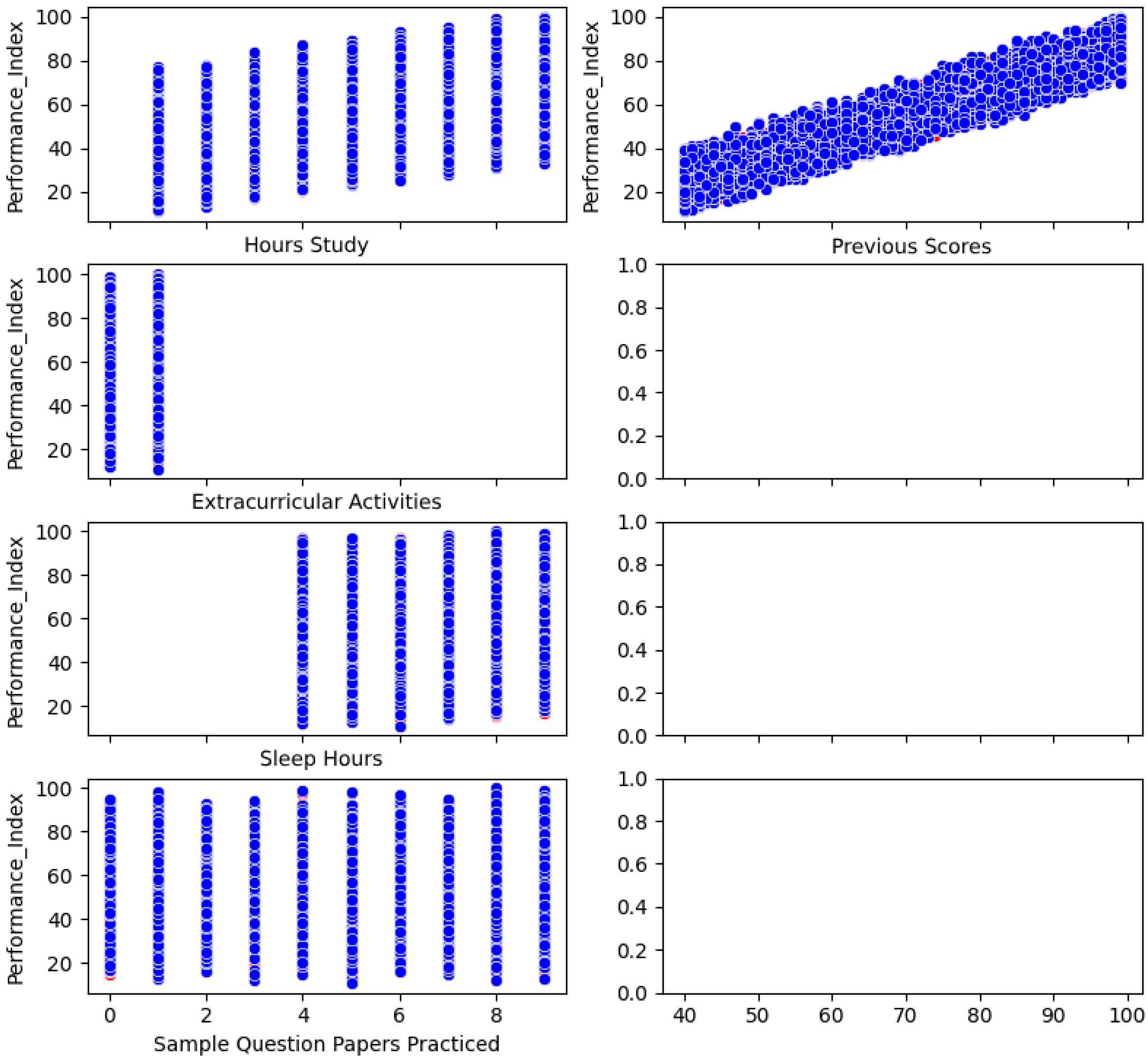

Рисунок 3.1 – График рассеяния для всех признаков тестовой выборки и получившейся регрессии (тестовая выборка – синяя, регрессия – красная)

Выводы.

В ходе данной лабораторной работы был изучен и применён на практике линейный регрессор, а также его модификация для нелинейных данных – полиномиальные признаки. По итогам лабораторной можно сделать следующие выводы:

- Линейная регрессия сильнее всего зависит от признаков, которые показывают линейную зависимость между признаками, которые мы пытаемся предугадать.
- Полиномиальные признаки полезны в ситуации, когда прослеживается явная полиномиальная связь между признаками. Добиться приближения можно использование разных степеней полинома.
- Разобран пример реального применения линейного регрессора для “предугадывания” оценки студента исходя из его подготовки. Пусть и с погрешностью, но можно понять, как ученики из тестовой выборки могут сдать экзамен.