Въведение в Машинното обучение

Лабораторно упражнение №5, 6

Регресия. Продължение.

1. Цел на упражнението.

Целта на лабораторното упражнение е студентите да усвоят темата регресия, като решават задачи с множествена, полиномна и логистична регресия

2. Общ процес на изграждане на регресия в машинното обучение

1. Събиране на данни: Необходимо е да съберете данни, които ще използвате за обучение на модела. Тези данни трябва да съдържат информация за входните признаци и резултата, който искате да предскажете.
2. Подготовка на данните: Предварителна обработка на данните може да включва почистване, нормализация, трансформация и други операции, които имат за цел да подготвят данните за обучение на модела.
3. Разделяне на данните: Данните се разделят на обучаващо множество и тестващо множество. Обучаващото множество се използва за обучение на модела, а тестващото множество се използва за проверка на точността на модела.
4. Избор на модел: Избира се подходящ модел за линейна регресия, който е способен да моделира зависимостта между входните признаци и целевият изход.
5. Обучение на модела: Моделът се обучава върху обучаващото множество, като се използва методът на най-малките квадрати за определяне на коефициентите на линейната регресия.
6. Оценка на модела: Моделът се тества върху тестващото множество и се измерва неговата точност. Ако точността е недостатъчна, може да бъде необходимо да се подобри модела чрез подбор на различни параметри или избор на по-подходящ модел.
7. Приложение на модела: След успешното обучение и оценка, моделът може да бъде използван за предсказване на нови данни, които не са били използвани по време на обучението на модела.

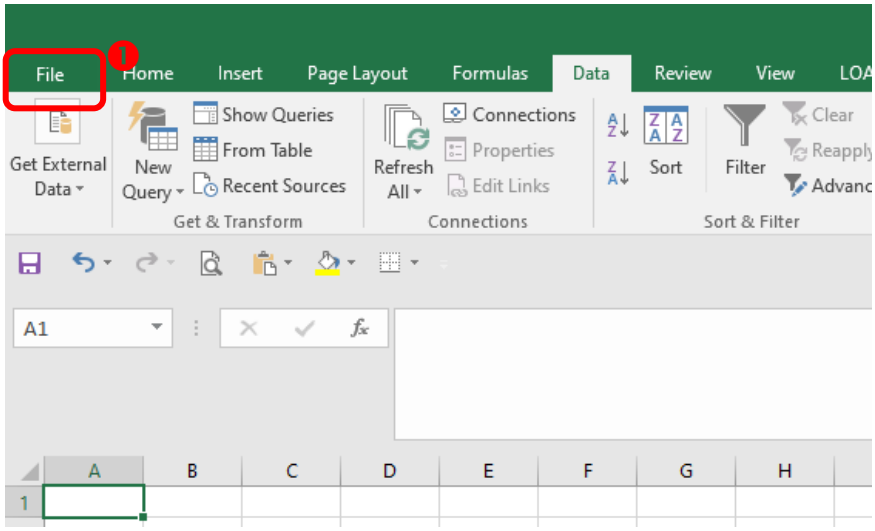
Тези стъпки са общи за много модели за линейна регресия, но могат да има допълнителни стъпки или вариации в зависимост от конкретната задача или метода на обучение.

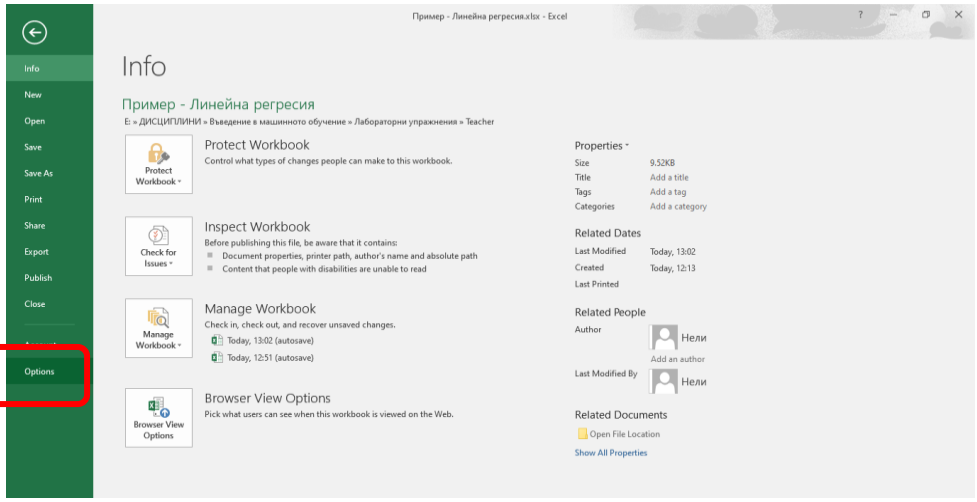
3. Регресионен анализ в Microsoft Excel

Във функционалността на Microsoft Excel съществуват инструменти, предназначени за извършване на регресионен анализ.

а) Добавяне на пакета за анализ. Стъпки:

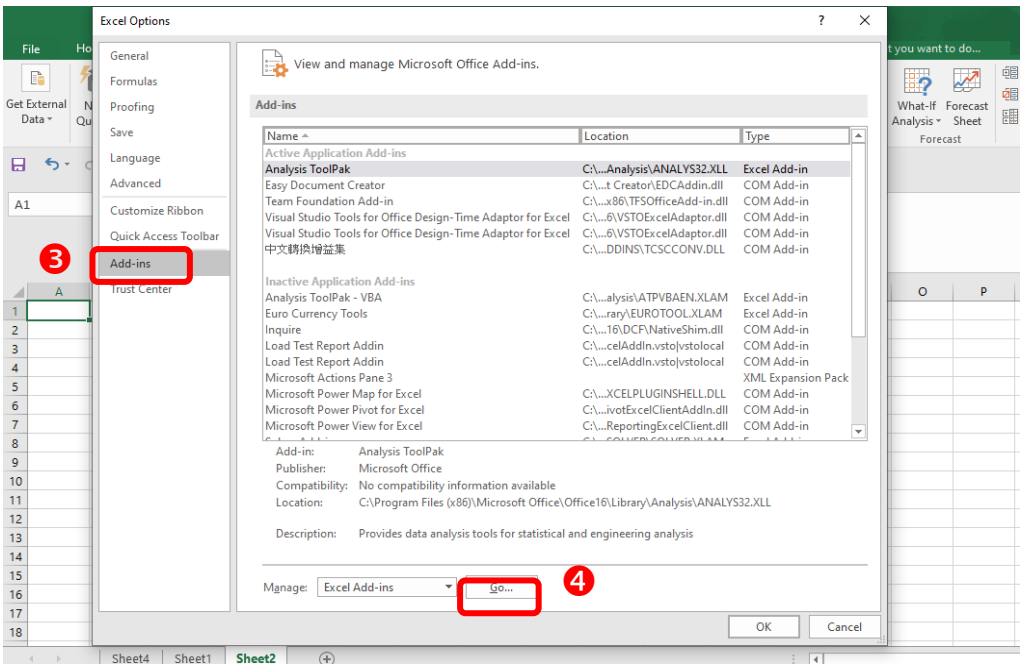
- От меню File ❶ изберете Options ❷



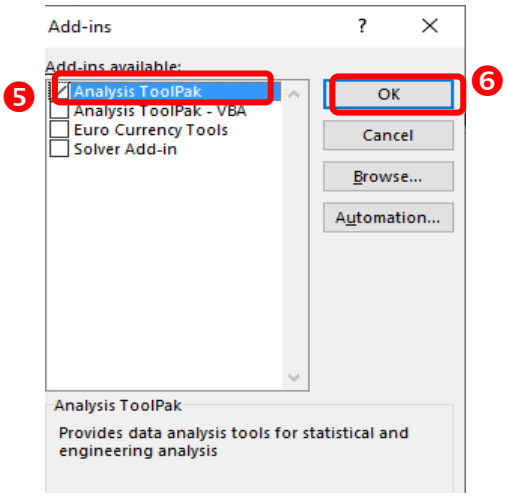


❷

- От Add-ins ❸ изберете GO (от Manage:) ❹

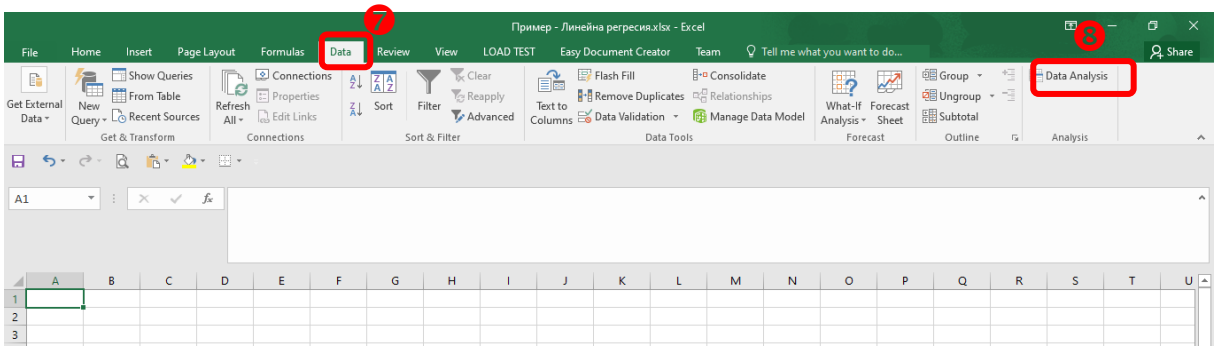


изберете Analysis ToolPak ❺ OK ❻



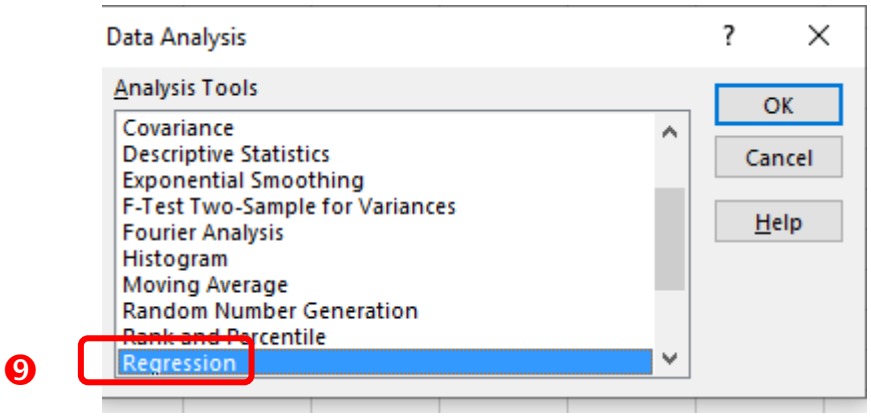
b) Активиране на пакета за анализ

- От меню Data ❼ изберете Data Analysis ❽

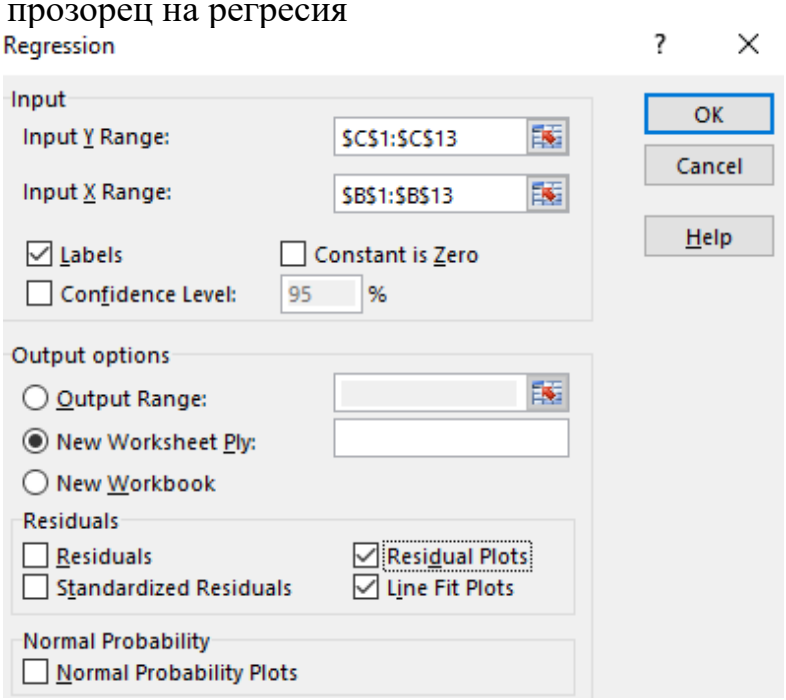


c) Активиране на регресия

- Data Analysis изберете Regression ❾ и OK



d) Диалогов прозорец на регресия



· *Input Y Range* – Въвежда се диапазона от стойности за зависимата променлива. Данните трябва да са в една колона.

· *Input X Range* - Въвежда се диапазона от стойности за независимите променливи. Microsoft Excel подрежда независимите променливи в нарастващ ред.

· *Confidence Level* – При избор се извежда допълнителен резултат в изходната таблица относно доверителните интервали. В съответното поле се въвежда доверителното ниво. В таблицата с изходните резултати са включени стойностите на границите на 95% доверителен интервал, без да е необходимо да се активира тази опция.

· *Constant is Zero* – При избор регресионната линия минава през началото на координатната система, т. е. в линейната зависимост да липсва свободен член.
· *Residuals* – При избор се включат остатъците (разликите между прогнозните и наблюдавани стойности на зависимата променлива) в изходна таблица.
· *Standardized Residuals* – При избор се включват стандартизираните остатъците в изходната таблица.
· *Residual Plots* - При избор се генерира графика на зависимостта между стойностите на независимите променливи и остатъците.
· *Line Fit Plots* - При избор се генерира графика на линейната зависимост, като по различен начин са означени прогнозираните и наблюдавани стойности на зависимата променлива.
· *Normal Probability Plots* - При избор се генерира графика на нормалното разпределение.

е) Изходните резултати от регресионния анализ

Изходните резултати от регресионния анализ се дават с няколко таблици, чиито брой зависи от зададените опции в началния диалогов прозорец **Regression**.
Първата изходна таблица **(Summary Output)** съдържа следните основни елементи:
Multiple R – коефициента на множествена корелация (между зависимата и независимите променливи). По определение той е равен на корен квадратен от коефициента на детерминация. Има стойност от 0 до 1. При линейна регресия с една независима променлива Multiple R е равен на коефициента на корелация между Y и X.

R-square – коефициент на множествена детерминация, който измерва каква част от общата вариация на зависимия признак се обяснява чрез влиянието на множеството независими променливи, включени в модела или с други думи, показва процента от общата дисперсия на зависимата променлива, който се "обяснява " чрез линейната регресия.
(*R-square= Regression SS/Total SS).*
Adjusted R-square= 1-(Residual SS/df)/(Total SS/df).
Standard Error of the estimate – Тази статистика измерва дисперсията (отклонението) на оригиналните (наблюдавани стойности) относно регресионната линия.
Observations – брой на двойките (x,y)

Втората изходна таблица от регресионния анализ е таблицата на дисперсионния анализ **(ANOVA)**, която съдържа следните основни термини:
df *– degrees of freedom* – степени на свобода.
Regression df = броят на независимите променливи;
Residual df = *(броят на наблюденията-1) – броят на независимите променливи*
Total df= Regression df + Residual df
Regression SS – регресионна сума от квадратите

Residual SS – остатъчна сума от квадратите
Total SS – обща сума от квадратите.
Regression MS = Regression SS/df ;
Residual MS = Residual SS/ df – средни квадрати
F = *Regression MS/ Residual MS*, F-отношението
Significance F е *P- стойността* за проверката на статистическата значимост на регресията.
Нулевата хипотезата Но: Всички коефициенти в регресионното уравнение пред независимите променливи са 0;
H1: поне един от тези коефициент е различен от 0
Третата изходна таблица включва стойностите на регресионните коефициенти, техните стандартни грешки, съответните *t-статистики* и *p-стойности,* чрез които се определя статистическата значимост на регресионните коефициенти, горната и долна граница на доверителните интервали за регресионните коефициенти при избраното ниво на доверие.

Тази таблица съдържа следните термини:

Coefficients- стойността на съответния коефициент в регресионното уравнение.
Intercept е наименованието на свободния член в регресионното уравнение.
Standard Error - стандартна грешка на коефициента.
t Stat – стойността на *t*-статистиката за тестване на статистическата значимост на съответния регресионен коефициент.
P-value – p-стойността, от която правим извод дали съответният регресионен коефициент е статистически значим, т.е. дали е различен от 0.
Lower 95%, Upper 95% - долната и горна граница на 95% доверителен интервал свободния член и регресионните коефициенти.
Ако в диалоговия прозорец **Regression** е активирана кутийката за избор *Line Fit Plots*, програмата генерира още една изходна таблица (RESIDUAL OUTPUT). В колоната “Predicted ” са дадени прогнозните стойности на зависимия признак за съответните стойности на независимия (от началните данни), а колоната "Residuals" съдържа разликите (отклоненията) на прогнозираните стойности на у от експерименталните. Освен това се генерира графика на линейната зависимост между зависимия и независимия признак, като по различен начин са означени точките, съответстващи на прогнозираните и наблюдавани стойности на зависимата променлива.

Задача 1. В таблицата са дадени данни за средната температура и количество валежи за всеки месец от годината. Да се определи как температурата влияе на валежите? Да се построи регресионен модел с **Regression** в Excel**.** (1 точка)

	A	B	C
1	Месец	Средна температура, C	Количество валежи, мм
2	Януари	-2.5	42.2
3	Февруари	-3	25.4
4	Март	5.5	52.2
5	Април	10.7	58.2
6	Май	19.5	65.2
7	Юни	24	68.3
8	Юли	28.8	70.2
9	Август	26.5	65.7
10	Септември	18.2	60.8
11	Октомври	9.8	62.2
12	Ноември	2.3	45.9
13	Декември	1.2	46.4

От менюто **Data**, се избира **Data Analysis** и от списъка със статистически процедури – **Regression.** На екрана се появява диалоговия прозорец ***Regression.***
В полето *Input Y Range: се* въвежда диапазона от стойности за зависимата променлива (количество валежи) - C1:C13.
В полето *Input X Range: се* въвежда диапазона от стойности за независимата променлива (средна температура) - B1:B13
Активира се кутийката за избор Labels, тъй като в маркираните области са включени и имената на признаците.
Активира се кутийката за избор *Line Fit Plots*, за да се генерира графика на линейната зависимост, а също и допълнителна таблица **RESIDUAL OUTPUT**, съдържаща прогнозните стойности на зависимия признак за съответните стойности на независимия, разликите (отклоненията) на експерименталните и прогнозирани стойности на y.

РЕЗУЛТАТИ

За разглеждания пример първата таблица с резултатите има следния вид:

SUMMARY OUTPUT	
Regression Statistics	
Multiple R	0.900617706
R Square	0.811112252
Adjusted R Square	0.792223477
Standard Error	6.031851079
Observations	12

Коефициентът на множествена корелация, който в случая е коефициентът на линейна корелация между зависимия (количество валежи) и независимия (средна температура) признак е 0,9006 (Multiple R). Коефициентът на детерминация има

стойност 0,81, което показва, че приблизително 81% от варирането на зависимия признак (количество валежи) се "обяснява" чрез регресията.

Втората изходна таблица от регресионния анализ е таблицата на дисперсионния анализ (ANOVA), която е свързана с различните видове дисперсии:

ANOVA					
	df	*SS*	*MS*	*F*	*Significance F*
Regression	1	1562.350226	1562.35	42.9415	6.44864E-05
Residual	10	363.8322744	36.38323		
Total	11	1926.1825			

В разглеждания пример, F-отношението има стойност 42,94, която е значително по-голяма от съответната критична стойност в таблицата с **F-разпределението**. Тази критична стойност не е включена в таблицата с резултатите, но стойността на **Significance F,** която е от порядъка на 6,44.10-5, т.е. число по-малко от 0,001, дава основание да се направи извод, че частта от дисперсията на зависимия признак (количество валежи), която се "обяснява" чрез регресията е значително по-голяма от остатъчната дисперсия (Residual), която се дължи на ненаблюдавани фактори (признаци).

Третата таблица включва стойностите на регресионните коефициенти, техните стандартни грешки, съответните t статистики и p-стойности, чрез които се определя статистическата значимост на регресионните коефициенти, горната и долна граница на доверителните интервали за регресионните коефициенти при избраното ниво на доверие. В първата колона на таблицата са наименованията на свободния член (Intercept) и регресионните коефициенти пред съответните независими променливи. Разглежданият пример е на линейна регресия с един независим признак и X Variable1 (средна температура) е названието на реда на съответния регресионен коефициент.

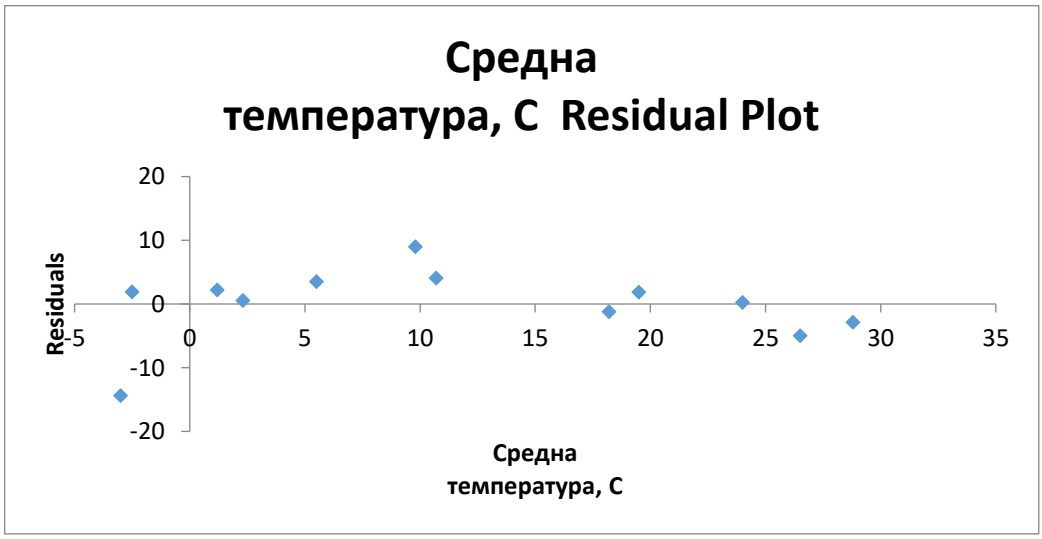
	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*
Intercept	42.92257629	2.560564347	16.76294	1.2E-08	37.21728338	48.627869
Средна температура, C	1.047014784	0.159776987	6.552976	6.45E-05	0.691009471	1.4030201

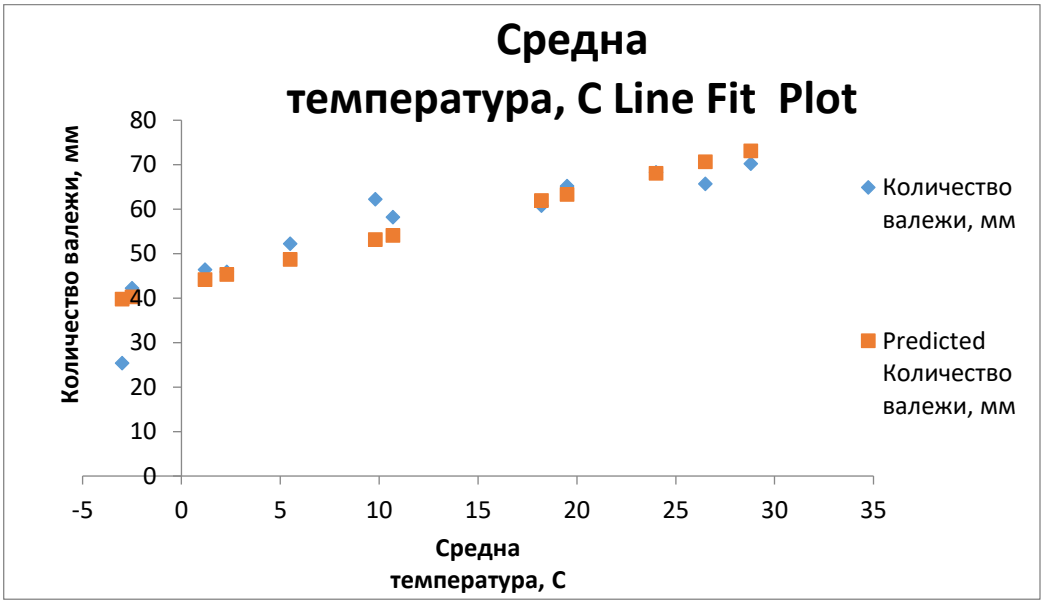
За разглеждания пример стойността на свободния член е 42,92, който е статистически значим (P-value = 1.2.10-8, т.е. P<0,05), а съответният 95% доверителен интервал е [37,21;48,63]. Регресионният коефициент пред независимия признак (средна температура) е 1,04. Той е статистически значим (P-value= 6,45.10-5 , т.е. P<0,001) , а съответният 95% доверителен интервал е [0,69; 1,40].

Регресионното уравнение има вида **y = 42,92+1,04.x**, където y е количество валежи, а x – средна температура.
Ако в диалоговия прозорец **Regression** е активирана кутийката за избор на Line Fit Plots, програмата генерира още една изходна таблица (RESIDUAL OUTPUT). В колоната “Predicted количество валежи” са дадени прогнозните стойности на зависимия признак за съответните стойности на независимия (от началните данни, номерирани в първата колона "Observation" (наблюдение), а колоната Residuals съдържа разликите (отклоненията) на експерименталните от прогнозираните стойности на y. Освен това се генерира графика на линейната зависимост между зависимия и независимия признак, като по различен начин са означени точките, съответстващи на прогнозираните и наблюдавани стойности на зависимата променлива.

RESIDUAL OUTPUT

Observation	*Predicted Количество валежи, мм*	*Residuals*
1	40.30503933	1.894960672
2	39.78153194	-14.38153194
3	48.6811576	3.5188424
4	54.12563448	4.074365523
5	63.33936458	1.860635424
6	68.0509311	0.249068896
7	73.07660207	-2.876602067
8	70.66846806	-4.968468064
9	61.97824536	-1.178245357
10	53.18332117	9.016678829
11	45.33071029	0.569289709
12	44.17899403	2.221005971





4. Множествена линейна регресия

Множествен линеен регресионен модел е статистическа модел, който се използва за оценка на връзката между една зависима променлива и множество независими променливи.

Ако в регресионния анализ е залегнало изучаването на едновременното влияние на повече от един фактор, зависимостта може да се моделира като множествена линейна регресия:

$$\hat{y} = a + b_1 x_1 + b_2 x_2 + \ldots + b_m x_m ,$$

където

$\hat{y}$ е теоретичната (оценъчна) стойност на резултативния признак;

x_i, $i = 1, \ldots, m$ са измерените стойности на фактор-признаците;

b_i, $i = 1, \ldots, m$ са коефициентите в уравнението на регресия;

a е свободният член.

Множествен линеен регресионен модел е полезен, когато има множество фактори, които могат да повлияят върху зависимата променлива, и когато се търси най-доброто уравнение, което може да описва връзката между тези променливи.

За построяването на множествен линеен регресионен модел е необходимо да се извършат стъпките по избор на подходящи независими променливи, събиране на данните, избор на модел и оценка на точността му.

След като се изгради множествен линеен регресионен модел, може да се използва за прогнозиране на стойностите на зависимата променлива при определени стойности на независимите променливи.

Задача 2. В таблица са дадени данни за общите разходи на туристите за почивка в конкретна дестинация и за съответните елементи на разходите: разходи за настаняване, разходи за храна и разходи за допълнителни услуги.

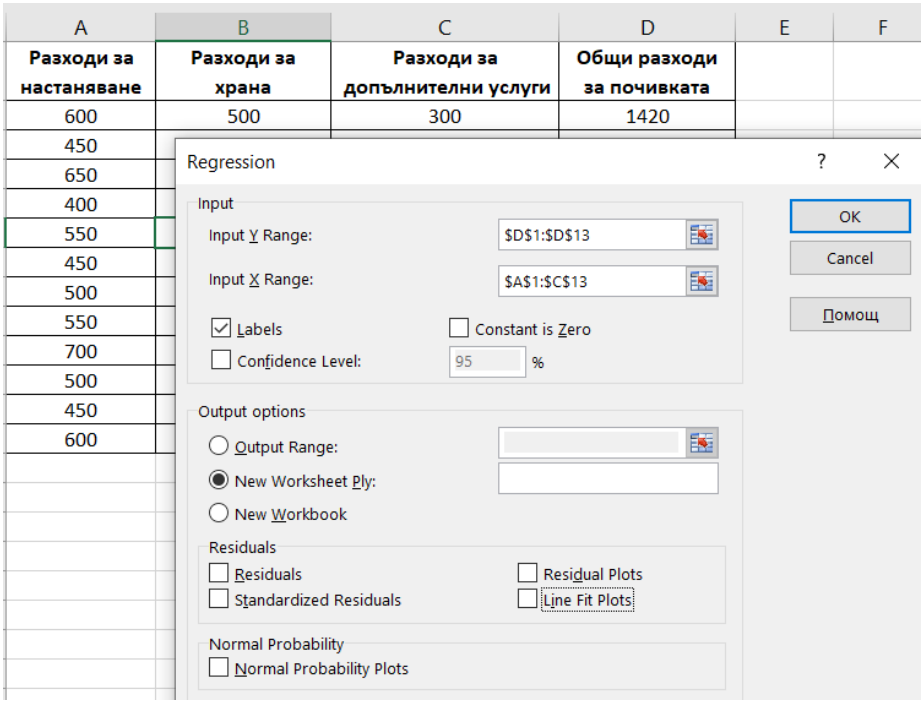
1. Да се построи множествен линеен регресионен модел, който определя зависимостта между общите разходи и съответните елементи на разходите.
2. Да се изведе регресионното уравнение.
3. Да се определят коефициента на множествена корелация и коефициента на множествена детерминация.
4. Да се генерира графика на зависимост между зависимия признак и независимите признаци. (2 точки)

Разходи за настаняване	Разходи за храна	Разходи за допълнителни услуги	Общи разходи за почивката
600	500	300	1420
450	550	470	1500
650	650	400	1720
400	450	300	1170
550	450	350	1380
450	600	420	1480
500	700	370	1600
550	550	450	1570
700	800	500	2000
500	780	440	1730
450	500	450	1420
600	450	540	1610

За решаването на този пример е подходящо да се използва инструментът Regression.

Реализация

От менюто **Data**, се избира **Data Analysis** и от списъка със статистически процедури – **Regression.** На екрана се появява диалоговия прозорец ***Regression.***
В полето *Input Y Range: се* въвежда диапазона от стойности за зависимата променлива (общи разходи за почивка) - D1:D13
В полето *Input X Range: се* въвежда диапазона от стойности за независимите променливи - A1:C13



5. Методи на класа LinearRegression от библиотеката scikit-learn на Python

fit(X, y) е метод на класа LinearRegression от библиотеката scikit-learn, който се използва за обучение на линейни регресионни модели. Той приема два аргумента: X - матрица от признаци (features) и y - вектор от целевите стойности (target values). Методът използва X и y за да обучи модела на линейната регресия.

score(X, y) е метод на класа LinearRegression и се използва за измерване на качеството на модела на линейната регресия. Той приема два аргумента: X - матрица от признаци и y - вектор от целевите стойности. Методът връща коефициент на детерминацията (r^2), който представлява процента на вариацията във вектора от целеви стойности, който може да бъде обяснен от модела. Коефициентът на детерминация, също известен като R-squared (R^2), е статистическа мярка за оценка на качеството на модела на регресия. Той измерва колко добре линейната регресионна линия приближава реалните стойности на зависимата променлива. Коефициентът на детерминация е число между 0 и 1, като по-високата стойност означава по-добро приближение на модела към реалните стойности.

predict(X) е метод на класа LinearRegression, който се използва за предсказване на целевите стойности на нови наблюдения на базата на обучения модел. Той приема един аргумент - матрицата от признаци на новите наблюдения (X) и връща вектор от предсказани целеви стойности.

6. Атрибути на класа LinearRegression от библиотеката scikit-learn на Python

model.coef_ и **model.intercept_** са атрибути на класа LinearRegression в библиотеката scikit-learn. Когато използваме метода fit() на този клас, той се обучава върху дадения набор от данни и извлича параметрите на линейната регресия, които след това могат да се достъпят чрез тези атрибути.

coef е коефициентът на наклона на линията на регресия (slope), който отразява колко силно е свързана променливата x с променливата y. Например, ако коефициентът на наклона е положителен, това означава, че когато x нараства, y също ще нараства. Ако коефициентът на наклона е отрицателен, това означава, че когато x нараства, y ще намалява.

intercept е точката на пресичане на линията на регресия с y-оста, която отразява стойността на y, когато x е равно на нула.

Тези атрибути позволяват на потребителя да прогнозира стойността на зависимата променлива, като използва формулата на линейната регресия.

За повече информация:
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

7. Допълнителни пояснения към кода на Пример 1

а) за редове от 4 до 11

```
4.    df = pd.DataFrame({
5.    'квадратура': [140, 160, 170, 187, 110, 155, 235, 245, 142, 170],
6.    'брой спални': [3, 3, 3, 3, 2, 4, 3, 4, 3, 3],
7.    'брой бани': [1, 2, 2, 2, 1, 2, 4, 4, 2, 3],
8.    'басейн': [0, 0, 0, 0, 0, 0, 1, 1, 1, 0],
9.    'гараж': [1, 2, 2, 2, 1, 2, 2, 2, 1, 2],
10.   'цена': [242000, 277000, 329000, 369000, 190000, 250000, 529900, 599000, 255000, 242900]
11.   })
```

DataFrame е основна структура от данни в библиотеката pandas в Python. Той се използва за съхранение на многомерни данни в табличен формат, като същевременно осигурява функционалности за манипулация на данните.

б) за ред 32

```
32.   print(f'Прогнозирана цена на дома: {predicted_price[0]:.2f}')
```

Този код използва f-стрингове (f-strings), за да изведе на конзолата съобщение, което съдържа текст и стойност на променлива, форматирана с определен брой знака след десетичната запетая.

f-стринговете в Python 3.6 и по-нови версии се означават със символа "f" или "F" пред низа.

След като е изведена стойността на променливата, ":.2f" указва, че искаме да се изведат два знака след десетичната запетая. Това означава, че стойността ще бъде закръглена до два знака след запетаята.

Пример 1.

В таблица са дадени цените на вилни имоти на базата на следните данни: квадратура, брой спални, брой бани, басейн и гараж. Да се построи множествен линеен регресионен модел. Да се определят коефициентите на множествена регресия, коефициентът на детерминация и свободния член. Да се определи прогнозната цена на вилен имот при квадратура 180 кв. м., брой спални 3, брой бани 2, без басейн и 2 гаража.

квадратура	**брой спални**	**брой бани**	**басейн**	**гаража**	**цена**
140	3	1	0	1	242000
160	3	2	0	2	277000
170	3	2	0	2	329000
187	3	2	0	2	369000
110	2	1	0	1	190000
155	4	2	0	2	250000
235	3	4	1	2	529900
245	4	4	1	2	599000
142	3	2	1	1	255000
170	3	3	0	2	242900

```
1. import pandas as pd
2. from sklearn.linear_model import LinearRegression

3. # Входни данни за вилни имоти
4. df = pd.DataFrame({
5. 'квадратура': [140, 160, 170, 187, 110, 155, 235, 245, 142, 170],
6. 'брой спални': [3, 3, 3, 3, 2, 4, 3, 4, 3, 3],
7. 'брой бани': [1, 2, 2, 2, 1, 2, 4, 4, 2, 3],
8. 'басейн': [0, 0, 0, 0, 0, 0, 1, 1, 1, 0],
9. 'гараж': [1, 2, 2, 2, 1, 2, 2, 2, 1, 2],
10.'цена': [242000, 277000, 329000, 369000, 190000, 250000, 529900, 599000,
    255000, 242900]
11.})
```

```
# Подготовка на входните и изходните данни
X = df[['квадратура', 'брой спални', 'брой бани', 'басейн', 'гараж']]
y = df['цена']

# Изграждане на модела на множествена линейна регресия
model = LinearRegression().fit(X, y)

# Нови данни за прогноза
new_data = pd.DataFrame({
'квадратура': [180],
'брой спални': [3],
'брой бани': [2],
'басейн': [0],
'гараж': [2]
})
# Изготвяне на прогнозата
predicted_price = model.predict(new_data)

# Извеждане на резултата
print('Множествена линейна регресия на данните:')
print(f'Коефициент на детерминацията: {model.score(X, y):.2}')
print(f'Множествени коефициенти на регресия: {[f"{coef:.2f}" for coef in model.coef_]}')
print(f'Свободен член на регресията: {model.intercept_:.2f}')
print(f'Прогнозирана цена на дома: {predicted_price[0]:.2f}')
```

Резултати

Множествена линейна регресия на данните:
Коефициент на детерминация: 0.97
Множествени коефициенти на регресия: ['4138.40', '-14096.47', '-54079.38', '61382.07', '-12513.84']
Свободен член на регресията: -210001.79
Прогнозирана цена на дома: 359434.96

Задача 3.

В таблица са дадени цените на кутии на базата на следните данни: дължина, широчина и височина. Да се построи множествен линеен регресионен модел. Да се определят коефициентите на множествена регресия, коефициентът на детерминация и свободния член. Да се определи прогнозната цена на кутия с размери: дължина 7 см, широчина 6 см и височина 5 см. (2 точки)

дължина	широчина	височина	цена
7	5	5	6.5
10	4	5	6.55
8	5	5	6.31
9	5	5	5.51
10	6	5	4.44

8. Полиномна регресия

Полиномната регресия е метод за оценка на връзката между две променливи, която се базира на полиномна функция от степен n. При тази форма на регресия, променливата, която искаме да предскажем, се приближава чрез комбинация от степени на другата променлива. Например, ако имаме една независима променлива x и една зависима променлива y, можем да използваме полиномна регресия, за да определим връзката между x и y, като използваме уравнението:

$y = b_0 + b_1x + b_2x^2 + ... + b_nx^n$

където y е зависимата променлива, x е независимата, b_0, b_1, b_2 … b_n са коефициенти, които трябва да бъдат намерени чрез обучение на модела на полиномната регресия, а n е степента на полинома.

Както и при линейната регресия, целта на полиномната регресия е да се намерят коефициентите, които минимизират сумата на квадратите на разликата между предсказаните стойности и реалните стойности на зависимата променлива.

Полиномната регресия има предимството да може да моделира по-сложни връзки между променливите, което я прави подходяща за решаване на по-сложни проблеми в сравнение с линейната регресия.

9. Допълнителни пояснения към кода на Пример 2

а) за ред 9

9. plt.scatter(df['Време за обработка'], df['Цена на продукта'])

Функцията scatter() от библиотеката matplotlib.pyplot, изобразява графика на разсейване (scatter plot) на данните. Тази функция приема два аргумента - X и y, които са масиви (arrays) или списъци (lists) от еднакъв размер, представляващи

съответно стойностите на признака (feature) и целевата променлива (target variable) в линейната регресия.

б) За редове 14 и 15

14. X = df.iloc[:, 0:1].values
15. y = df.iloc[:, 1].values

df.iloc[:, 0:1] избира всички редове на DataFrame df, но само колоната с индекс 0 (първата колона в DataFrame). Това се постига чрез : в първата позиция, която указва, че трябва да бъдат избрани всички редове, и 0:1 във втората позиция, която указва, че трябва да бъде избрана само първата колона (тъй като срязът е от 0 до 1, като крайния индекс 1 не е включен).

Аналогично, **df.iloc[:, 1]** избира всички редове на DataFrame df, но само колоната с индекс 1 (втората колона в DataFrame). Това връща Pandas Series, който ще бъде записан в променливата y. Отново се използва .values методът, за да се извлече NumPy array от Series, който ще бъде записан в y.

Така X съдържа features на данните, а y - target стойностите, които могат да се използват за обучение на модел на машинно обучение

в) За ред 17

17. poly = PolynomialFeatures(degree=2)

PolynomialFeatures е клас от библиотеката scikit-learn, който се използва за генериране на полиномиални признаци (polynomial features) от данни. Това се прави като се вдигнат всички признаци до определена степен. Например, ако имаме признаците x1 и x2, и зададем степен 2, то се генерират следните признаци: x1, x2, $x1^2$, x1x2 и $x2^2$. Степента на полинома се задава чрез параметъра degree.

г) За ред 18

18. X_poly = poly.fit_transform(X)

poly.fit_transform(X) е метод на класа PolynomialFeatures от библиотеката scikit-learn, който се използва за генериране на полиномиални признаци (polynomial features) от данните в матрицата X.

Пример 2.

В таблица са дадени данните за времето за обработка на продукт и съответно неговата цена. Да се построи полиномен регресионен модел. Да се визуализират

данните в графика. Да се прогнозират цените на продукта въз основа на нови времена за обработка [11, 12, 13] и да се изведат прогнозните данни.

Време за обработка	**Цена на продукта**
1	1
3	9
5	25
7	49
9	81

Особености на примера: Зависимостта между данните е квадратична

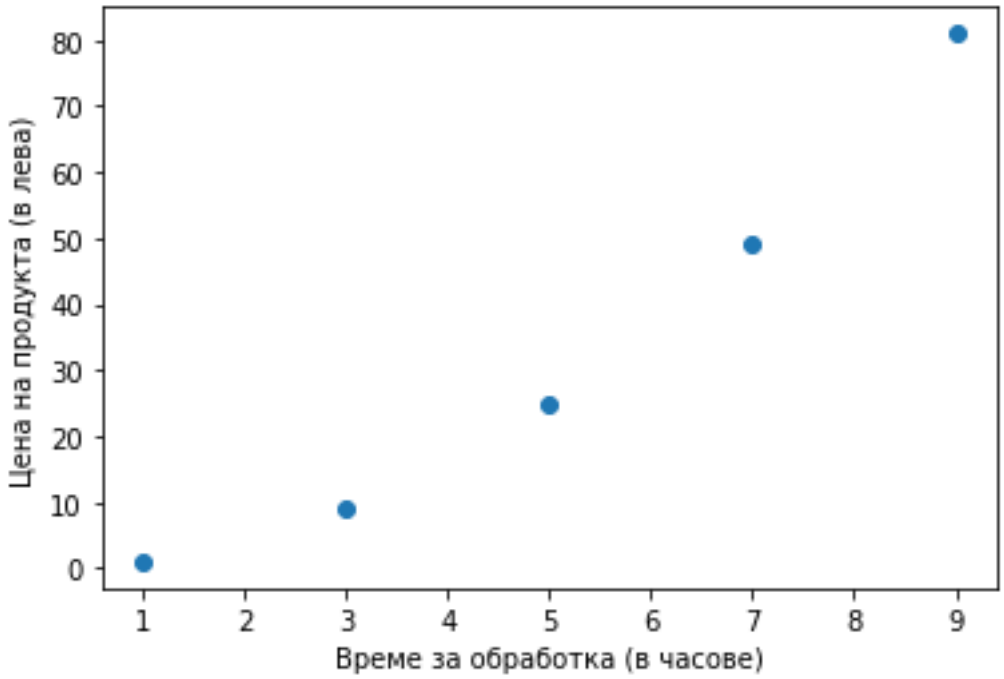
```
1. import numpy as np
1. import pandas as pd
2. from sklearn.linear_model import LinearRegression
3. from sklearn.preprocessing import PolynomialFeatures
4. import matplotlib.pyplot as plt


5. # Дефиниране на таблицата с данни
6. data = {'Време за обработка': [1, 3, 5, 7, 9], 'Цена на продукта': [1, 9, 25, 49, 81]}
7. df = pd.DataFrame(data)


8. # Визуализация на данните в графика
9. plt.scatter(df['Време за обработка'], df['Цена на продукта'])
10.plt.xlabel('Време за обработка (в часове)')
11.plt.ylabel('Цена на продукта (в лева)')
12.plt.show()


13.# Разделяне на данните на тренировъчен и тестов набор
14.X = df.iloc[:, 0:1].values
15.y = df.iloc[:, 1].values


16.# Създаване на полиномиални функции на данните
17.poly = PolynomialFeatures(degree=2)
18.X_poly = poly.fit_transform(X)
```

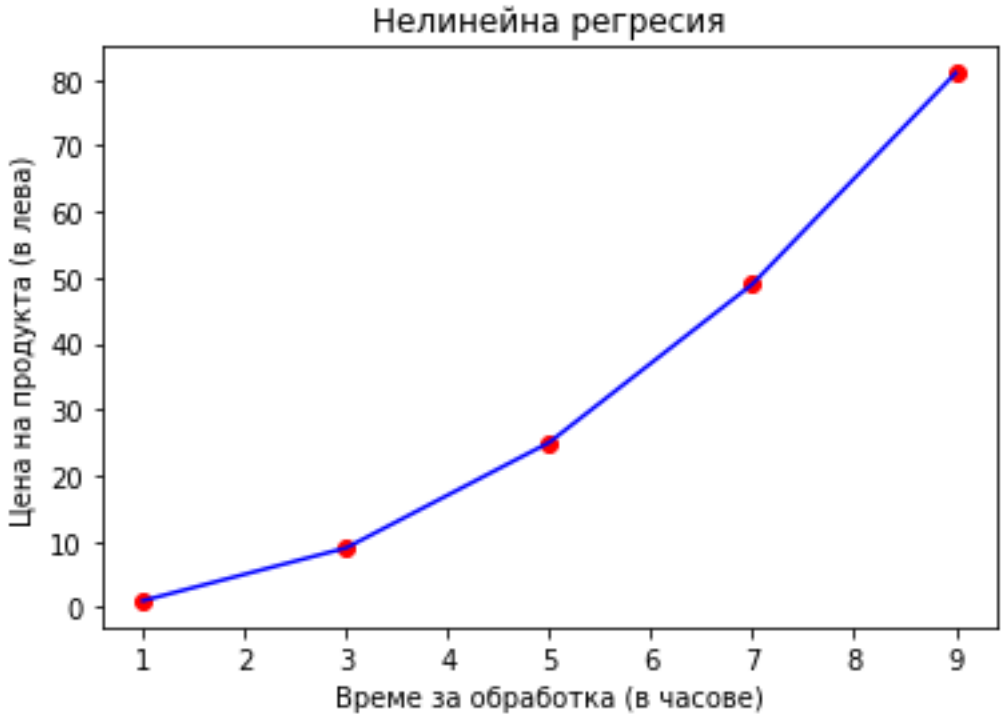
```
# Намиране на коефициентите на регресията
lin_reg = LinearRegression()
lin_reg.fit(X_poly, y)


# Визуализация на резултатите
plt.scatter(X, y, color='red')
plt.plot(X, lin_reg.predict(poly.fit_transform(X)), color='blue')
plt.title('Нелинейна регресия')
plt.xlabel('Време за обработка (в часове)')
plt.ylabel('Цена на продукта (в лева)')
plt.show()


# Прогноза за нови данни
X_test = np.array([11, 12, 13]).reshape((-1, 1))
X_test_poly = poly.transform(X_test)
y_pred = lin_reg.predict(X_test_poly)


# Извеждане на прогнозите на екрана
print("Прогнозираните цени на продукта са:")
print(y_pred)
```

Резултати

Прогнозираните цени на продукта са:

[121. 144. 169.]





Задача 4.

В таблица са дадени данните за разходите за реклама на продукт и съответно броя на реализираните продажби на този продукт. Да се построи полиномен регресионен модел. Да се визуализират данните в графика. Да се прогнозират продажбите на продукта (в бройки) въз основа на нови разходи за реклама на продукти [8, 9, 10] и да се изведат прогнозните данни. (2 точки)

Цена за реклама (в евро)	Реализирани обороти от продукта (в бр.)
2	10
3	29
4	66
5	127
6	218
7	345

Особености на задачата: Зависимостта между данните е кубична

10. Логистична регресия

Логистичната регресия е статистически метод, използван за моделиране на вероятността за определено събитие, като се анализира зависимостта между независими променливи и дискретна зависима променлива, която може да приема стойности от два класа. Това я прави особено полезна за прогнозиране на вероятността за появата на дадено събитие.

Моделът на логистичната регресия се състои от линейна комбинация на независимите променливи, които се умножават с техните съответни коефициенти и след това се подават през логистична функция. Логистичната функция трансформира резултата в интервала [0, 1], което може да се тълкува като вероятност за принадлежност към един от двата класа. Формално, математически модел на логистичната регресия може да се изрази по следния начин:

$$P(Y = 1|X) = \frac{1}{1 + e^{-z}}$$

където:

P(Y=1|X) е вероятността за появата на положителен клас (Y=1) при даден вектор от независими променливи X

z е линейната комбинация на X и техните съответни коефициенти:

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n * X_n$$

Коефициентите β_0, β_1, β_2, ..., β_n се определят чрез обучение на модела на логистичната регресия върху дадените данни. Техниките за обучение на

логистична регресия включват метода на максималната правдоподобност и метода на градиентното спускане.

11.Допълнителни пояснения към кода на Пример 3.

а) за редове 11 и 12

```
11.  def logistic_function(x, A, k, x0):
12.  return A / (1 + np.exp(-k*(x-x0)))
```

Функцията logistic_function има три параметъра: A, k и x0, където A е теоретичният максимум на функцията, когато x се приближава към безкрайност, k е скоростта на растеж на зависимата променлива и x0 е средната точка на зависимата променлива. Тези параметри са определени така, че да се приближат най-добре към действителните входни и изходни данни.

б) за ред 17

```
17.  popt, pcov = curve_fit(logistic_function, x, y)
```

popt е масив от най-добрите стойности на параметрите на модела, които бяха намерени от функцията curve_fit. Тези стойности са такива, че моделът най-добре описва дадените данни.

pcov е ковариационната матрица на параметрите на модела. Тя дава информация за това колко добре определени са параметрите на модела. Ако стойностите на pcov са малки, това означава, че параметрите на модела са добре определени, ако обаче стойностите са големи, това може да означава, че параметрите не са добре определени и апроксимацията не е толкова точна.

curve_fit е функция в библиотеката scipy.optimize, която се използва за намиране на параметрите на модела, които най-добре описват дадени данни. Тази функция използва метода на най-малките квадрати за оптимизиране на параметрите на модела.

logistic_function е функция, която се използва като модел за описване на данните. Тя е логистична функция, която се използва за моделиране на нарастващи или намаляващи тенденции, като например при определяне на растежа на населението, разпространението на инфекциозни болести и т.н.

в) за ред 22

```
22.  plt.plot(x, logistic_function(x, *popt), 'b-', label='Резултат')
```

При извикването на функцията **logistic_function** в кода, x е масивът със стойности на независимата променлива, който е подаден като втори аргумент, а останалите 3 аргумента са представени чрез **popt**, който е tuple от коефициентите, върнати от функцията curve_fit. Това става чрез звездичката (*), която разгръща елементите на tuple-а и ги подава като аргументи на функцията. Така функцията logistic_function получава стойности на всички аргументи.

Пример 3.

В таблица са дадени цените на един продукт, които в рамките на 10 месеца непрекъснато нарастват. Да се построи логистичен регресионен модел. Да се визуализират данните в графика. Да се прогнозира цената на продукта за месеците ноември (11) и декември (12) и да се изведат прогнозните данни.

Месеци **X**	**Цена на продукта** **У**
1	2,1
2	3,5
3	4,2
4	5
5	6,3
6	7,1
7	8,2
8	9
9	10,1
10	11,2

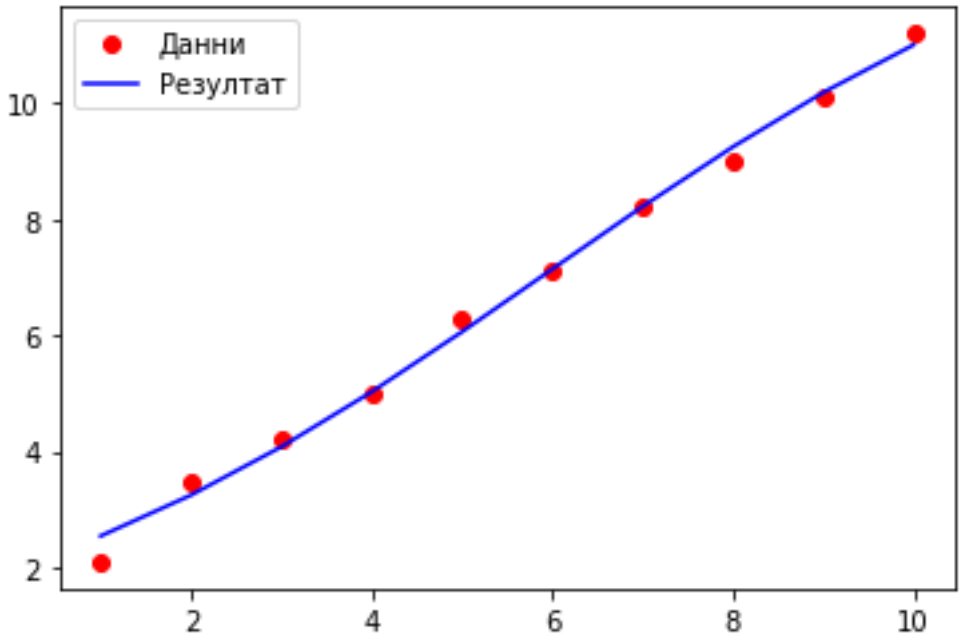
```
1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. from scipy.optimize import curve_fit


5. # зареждаме данните в DataFrame
6. data = pd.DataFrame({
7. 'X': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
8. 'Y': [2.1, 3.5, 4.2, 5.0, 6.3, 7.1, 8.2, 9.0, 10.1, 11.2]
9. })
```

```
10. # определяме функцията на Логистичната крива
11. def logistic_function(x, A, k, x0):
12. return A / (1 + np.exp(-k*(x-x0)))


13. # определяме x и y от данните
14. x = np.array(data['X'])
15. y = np.array(data['Y'])


16.# извършваме нелинейна регресия
17. popt, pcov = curve_fit(logistic_function, x, y)


18.# извеждаме коефициентите
19. print('Коефициенти: A = {:.4}, k = {:.2}, x0 = {:.3}'.format(*popt))


20.# създаваме графика
21.plt.plot(x, y, 'ro', label='Данни')
22.plt.plot(x, logistic_function(x, *popt), 'b-', label='Резултат')
23.plt.legend()
24.plt.show()


25.# правим прогноза
26.x_new = np.array([11, 12, 13, 14, 15])
27.y_new = logistic_function(x_new, *popt)
28.print(f'Прогнозирана цена= {y_new[0]:.2f}')
```

Резултати:

Коефициенти: A = 14.18, k = 0.31, x0 = 5.95

Прогнозирана цена= 11.70



Задача 5.

В таблица са дадени набор от данни представляващи нарастващ брой спам в рамките на една седмица в електронната поща на даден потребител. Да се построи логистичен регресионен модел. Да се визуализират данните в графика. Да се прогнозира колко спам ще се получи на десетия ден. Резултатът да се изведе като цяло число. (2 точки)

Ден	Спам (брой)
1	2
2	4
3	4
4	5
5	6
6	7
7	8

12.Допълнителни пояснения към кода на задача 6

а) за ред 14

14. train_data, test_data = train_test_split(data, test_size=0.3, random_state=42)

train_test_split е функция в библиотеката sklearn.model_selection, която се използва за разделяне на даден набор от данни на обучаващ и тестов набор, като позволява да се контролира големината на тестовия набор.

data е масив с данните, които искаме да разделим на обучаващ и тестов набор.
test_size е параметър, който определя какъв процент от данните да се използва за тестов набор. В случая, test_size=0.3 означава, че 30% от данните ще бъдат използвани за тестов набор, а останалите 70% ще бъдат използвани за обучаващ набор.
random_state е параметър, който се използва за контролиране на случайността при разделянето на данните. При задаване на определено число за този параметър, ще се получи винаги едно и също разделение на данните при повторното изпълнение на функцията със същото число. Това е полезно, когато искаме да можем да възпроизведем резултатите от даден експеримент.
train_data и test_data са двата резултата от функцията train_test_split. train_data е обучаващият набор от данни, който се използва за обучаване на модела, а test_data е тестовият набор от данни, който се използва за проверка на точността на модела.

б) за ред 35

35. accuracy = accuracy_score(y_test, y_pred)

accuracy_score е функция от библиотеката sklearn.metrics, която се използва за изчисляване на точността на класификационен модел. Точността (accuracy) е мярка за това колко точно моделът предвижда класовете на даден набор от тестови данни. Тя се дефинира като броят вярно класифицирани примери спрямо общия брой примери в тестовия набор.

Задача 6.

В таблица са дадени набор от данни съдържащи информация за електронната поща на даден потребител: дължина на писмо в брой символи, честота на срещанията на изпращача, размер на писмото в байтове и категория на писмото – спам или не-спам. Да се построи логистичен регресионен модел. Да се изведе категорията на писмото след въвеждан на нови данни за писмо: Length=150, Frequency=2 и Size=12. Да се изчисли и изведе точността на модела. В програмата на задачата липсват части от кода, които трябва да въведете. Това са редове 1, 8, 9, 10, 11, 27, 28 и ред 36. (1 точка)

Sender **Изпращач**	*Length* **Дължина на писмото (брой символи)**	*Frequency* **Честота на срещанията на изпращача**	*Size* **Размер на писмото (байтове)**	*Category* **Категория**
А	**248**	**2**	**22**	**спам**
Б	**180**	**3**	**16**	**спам**
В	**120**	**1**	**8**	**не-спам**
Г	**200**	**4**	**20**	**спам**
Д	**75**	**1**	**6**	**не-спам**
Е	**320**	**5**	**30**	**спам**
Ж	**150**	**2**	**12**	**не-спам**
З	**280**	**3**	**24**	**спам**
И	**90**	**1**	**5**	**не-спам**

```
#Импортиране на библиотеката pandas като pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score


# Зареждане на данните в DataFrame
data = pd.DataFrame({
'Sender': ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I'],
                                                    ,
                                                  ,
                                                  ,
                                                ]
})


# Разделяне на данните на обучаващо и тестващо множество
train_data, test_data = train_test_split(data, test_size=0.3, random_state=42)
```

```
# Изграждане на модел за логистична регресия
log_reg = LogisticRegression()
X_train = train_data[['Length', 'Frequency', 'Size']]
y_train = train_data['Category']
log_reg.fit(X_train, y_train)


# Тестване на модела
X_test = test_data[['Length', 'Frequency', 'Size']]
y_test = test_data['Category']
y_pred = log_reg.predict(X_test)


# Въвеждане на данните за писмото, което искаме да класифицираме
new_data = pd.DataFrame({
'Length': [150],
                ,
             ]
})


# Предсказване на категорията на писмото
y_new = log_reg.predict(new_data)


# Извеждане на предсказаната категория
print('Категория на писмото:', y_new[0])


# Оценка на точността на модела
accuracy = accuracy_score(y_test, y_pred)
                          #Извеждане на точността
```

Резултат:

Категория на писмото: non-spam

Accuracy: 1.0