گزارش‌کار پروژه پایتون

درس اقتصاد مهندسی

فاطمه ایرجی / ۹۹۱۰۳۶۷۷

محمد مهدی مهری / ۹۹۱۰۹۹۹۳

متین خرازی / ۹۹۱۰۳۸۵۸

حنانه ملکوتی / ۹۹۱۰۰۹۰۳

علیرضا آزادی / ۴۰۰۱۰۳۱۱۹

سارا یادگاری / ۴۰۱۱۰۴۵۶۵

بهار ۱۴۰۳

فهرست

توابع اقتصاد مهندسی

در این قسمت با توجه به اموزش‌های داده شده در ویدئو کلاس حل تمرین، چهار تابع ساخته شده‌است که این توابع هر کدام از موارد F، P، A، را به یک دیگر با توجه به فاکتورهای مربوطه تبدیل می‌کند. این توابع در قسمت‌های بعدی پروژه به‌صورت ماژولار استفاده خواهند شد. این کد در فایل factors.py ذخیره شده‌است.

```
#Converts a future value (f) to a present value (p) using a
discount rate (i) and the number of periods (n).
def f_to_p (i, n, f):
factor = 1 / (1 + i) ** n
return f * factor

#Converts an annuity value (a) to a present value (p) using
a discount rate (i), growth rate (j), and number of periods
(n).
def a_to_p (i, j, n, a):
if i == j:
factor = n / (1 + i)
else:
factor = (1 - (((1 + j) ** n) * ((1 + i) ** -n))) / (i - j)

return a * factor

#Converts a present value (p) to an annuity value (a) using
a discount rate (i) and the number of periods (n).
def p_to_a (i, n, p):
factor = (i * ((1 + i) ** n)) / (((1 + i) ** n) - 1)
return p * factor

#Converts a present value (p) to a future value (f) using a
discount rate (i) and the number of periods (n).
def p_to_f (i, n, p):
factor = (1 + i) ** n
return p * factor
```

شرح تمرین

در این بخش هر یازده بخش خواسته‌شده، توضیح داده خواهد شد و خروجی‌ها را در قالب شکل و عدد نشان می‌دهیم.

حذف داده‌های null با میانگین

در این قسمت خواسته‌شده است که داده‌های null مربوط به ستون salary را در جدول اکسل پر کنیم. این کار با از کتابخانه pandas و با میانگین گیری از سایر داده‌ها انجام می‌دهیم. داده‌ها برای سه سال گمشده به‌صورت زیر است.

years	2003	2010	2013
salary	76003	76003	76003

1.dealing with missing values

```
In [1]: import pandas as pd
        import numpy as np
        from factors import *
        from sklearn.linear_model import LinearRegression

In [2]: file_path = 'data.xlsx'

In [3]: data_frame = pd.read_excel(file_path)

In [4]: # fills missing values in the 'Salary' column with the mean salary

        mean_Salary = data_frame['Salary'].mean()

        data_frame['Salary'].fillna(mean_Salary, inplace=True)

In [5]: data_frame.head(10)
```

Out[5]:

	Years	Salary	material 1	material 2	demand	price
0	1990	39343.0	200.000000	89.600000	103	1200.0
1	1991	46205.0	212.000000	92.662500	104	1375.0
2	1992	37731.0	224.720000	95.807250	101	1451.0
3	1993	43525.0	238.203200	99.034819	99	1593.0
4	1994	39891.0	252.495392	102.345626	95	1718.5
5	1995	56642.0	267.645116	105.739927	96	1844.0
6	1996	60150.0	283.703822	109.217795	95	1969.5
7	1997	54445.0	300.726052	112.779099	92	2095.0
8	1998	64445.0	318.769615	116.423489	92	2220.5
9	1999	57189.0	337.895792	120.150370	87	2346.0

```
In [6]: data_frame.to_excel('New_data.xlsx')
```

داده‌های جدید را در یک فایل اکسل با نام New_data ذخیره می‌کنیم و باقی سؤال‌ها را با این داده انجام می‌دهیم.

پیش‌بینی حقوق کارکنان

در این قسمت با توجه به اموزش‌های داده شده در ویدئو کلاس حل تمرین، یک مدل رگرسیون خطی را با توجه سال‌ها و دستمزدها که به‌صورت x، y، تعریف‌شده‌اند فیت می‌کنیم و سپس دستمزدها برای سال‌های 2023 تا 2050 پیش‌بینی می‌کنیم.

2.Predict Salary

In [7]:
```
data_frame2 = pd.read_excel('New_data.xlsx')
```

In [8]:
```
X = data_frame2['Years'].values.reshape(-1,1)
Y = data_frame2['Salary'].values.reshape(-1,1)
```

In [9]:
```
model = LinearRegression()
```

In [10]:
```
model.fit(X,Y)
```

Out[10]:
```
LinearRegression()
```

In [11]:
```
intercept = model.intercept_[0]
coefficient = model.coef_[0][0]
```

In [12]:
```
#predicting salary for years 2023 to 2050

predict_salary = {}

for year in range(2023, 2051) :
    predict_salary[year] = (intercept + coefficient * year)

#predict_salary
```

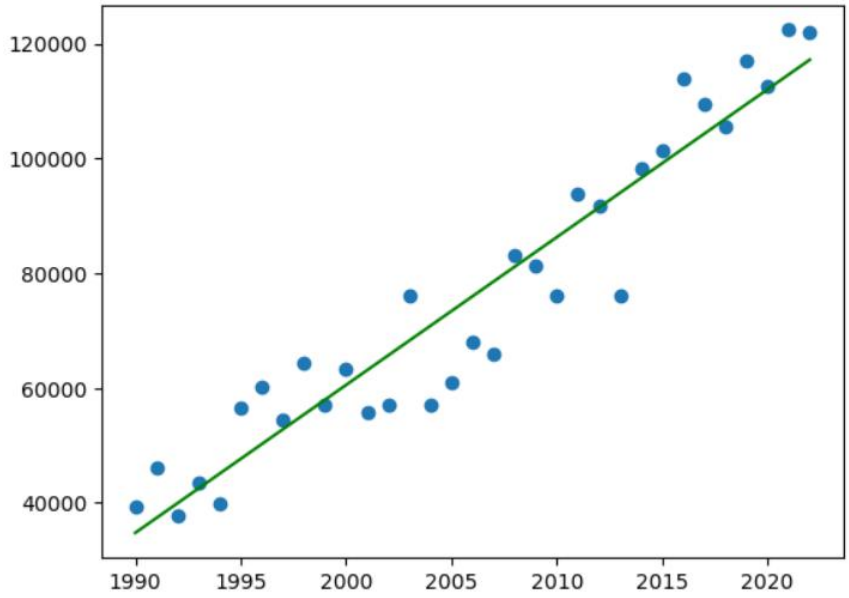
In [13]:
```
predict_salary_df = pd.DataFrame.from_dict(predict_salary, orient='index', columns=['Salary']).reset_index()
```

خروجی این مدل به‌صورت زیر می‌شود:

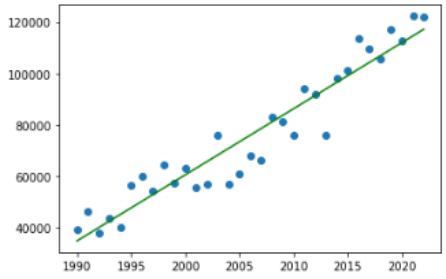
	index	Salary
0	2023	119794.784091
1	2024	122370.771390
2	2025	124946.758690
3	2026	127522.745989
4	2027	130098.733289
5	2028	132674.720588
6	2029	135250.707888
7	2030	137826.695187
8	2031	140402.682487
9	2032	142978.669786
10	2033	145554.657086
11	2034	148130.644385
12	2035	150706.631684
13	2036	153282.618984
14	2037	155858.606283
15	2038	158434.593583
16	2039	161010.580882
17	2040	163586.568182
18	2041	166162.555481
19	2042	168738.542781
20	2043	171314.530080
21	2044	173890.517380
22	2045	176466.504679
23	2046	179042.491979
24	2047	181618.479278
25	2048	184194.466578
26	2049	186770.453877
27	2050	189346.441176

نمودار داده حقوق کارکنان و مدل پیش‌بینی‌شده

در این قسمت با استفاده از کتابخانه matplotlib، نمودار شامل داده‌های حقوق کارکنان و خط رگرسیون به‌دست امده، به‌صورت زیر است. همچنین دقت مدل به‌دست امده، تقریباً برابر با 0.91094 است.




```
In [15]: #model.predict(X)

In [16]: import matplotlib.pylab as plt

In [17]: plt.scatter( data_frame2['Years'] , data_frame2['Salary'] )

         plt.plot( data_frame2['Years'] , model.predict(X) , c = 'green')
         plt.show()
```




```
In [18]: score = model.score(X,Y)
         #score

In [19]: score
Out[19]: 0.9109459138386182
```

پیش‌بینی قیمت هر واحد ماده اولیه ۱

در این قسمت، قیمت هر واحد ماده‌ی اولیه ۱ برای سال‌های آینده را می‌خواهیم به‌دست آوریم. همان طور که گفته شده‌است، این قیمت در سال بعد برابر است با قیمت سال قبل ضرب‌در جمع نرخ بهره و تورم یک درصد. بنابراین با استفاده از حلقه for این قیمت را برای سال‌های آینده به‌دست می‌اوریم.

4.Predict Price Material 1

In [20]:
```
material1_price_predict = {}

m1 = data_frame2['material 1'].values
m1_price = m1[-1]
```

In [21]:
```
#predicting the price of material 1 for years 2023 to 2050
for year in range(2023, 2051) :

    m1_price = m1_price * 1.06
    material1_price_predict[year] = m1_price
```

In [22]:
```
material1_price_predict_df = pd.DataFrame.from_dict(material1_price_predict , orient='index', columns=['material 1']).reset_index
```

خروجی این پیش‌بینی به‌صورت زیر می‌شود:

	index	material 1
0	2023	1368.117977
1	2024	1450.205055
2	2025	1537.217358
3	2026	1629.450400
4	2027	1727.217424
5	2028	1830.850469
6	2029	1940.701498
7	2030	2057.143587
8	2031	2180.572203
9	2032	2311.406535
10	2033	2450.090927
11	2034	2597.096383
12	2035	2752.922165
13	2036	2918.097495
14	2037	3093.183345
15	2038	3278.774346
16	2039	3475.500807
17	2040	3684.030855
18	2041	3905.072706
19	2042	4139.377069
20	2043	4387.739693
21	2044	4651.004074
22	2045	4930.064319
23	2046	5225.868178
24	2047	5539.420269
25	2048	5871.785485
26	2049	6224.092614
27	2050	6597.538171

پیش‌بینی هر واحد ماده اولیه ۲

در این قسمت، قیمت هر واحد ماده‌ی اولیه ۲ برای سال‌های آینده را می‌خواهیم به‌دست آوریم. برای این کار، ارزش قیمت‌های داده شده را با استفاده از تابع f_to_p، که ان را در فایل توابع اقتصاد مهندسی داشتیم، به سال ۱۹۹۰، تبدیل می‌کنیم. مشاهده می‌شود، طبق گفته سؤال این ارزش‌ها به‌صورت نزولی‌اند. سپس مدل رگرسیون خطی را بر روی ان فیت می‌کنیم و بعد ارزش‌ها را برای سال‌های آینده پیش‌بینی می‌کنیم. صورت س‌ؤال قیمت هر واحد ماده‌ی اولیه ۲ را برای سال‌های آینده خواسته‌است. بنابراین با استفاده از تابع p_to_f این ارزش‌ها را به‌صورت قیمت در می‌اوریم.

5.Predict Price Material 2

In [24]:
```
m2 = data_frame2['material 2'].values
```

In [25]:
```
from factors import f_to_p
```

In [26]:
```
#calculating the present value of the material 2

material2_price_predict_first = []

for n,f in enumerate(m2) :

    m2_price = f_to_p(.05, n, f)
    material2_price_predict_first.append(m2_price)

material2_price_predict_first = np.array(material2_price_predict_first)
```

In [27]:
```
X1 = data_frame2['Years'].values.reshape(-1,1)
Y1 = material2_price_predict_first.reshape(-1,1)
```

In [28]:
```
model1 = LinearRegression()
```

In [29]:
```
model1.fit(X1,Y1)
```

Out[29]:
```
▾ LinearRegression
LinearRegression()
```

In [30]:
```
intercept1 = model1.intercept_[0]
coefficient1 = model1.coef_[0][0]
```

In [31]:
```
#predicting the price of material 2 for years 2023 to 2050

predict_price_material2 = []

for year in range(2023, 2051) :

    number = (intercept1 + coefficient1 * year)
    predict_price_material2.append(number)
```

In [32]:
```
#calculating the future value of the material 2
material2_price_predict_last = {}

for n,p in enumerate(predict_price_material2) :

    material2_price_predict_last[n+2023] = p_to_f(.05, n+33, p)

#material2_price_predict_last
```

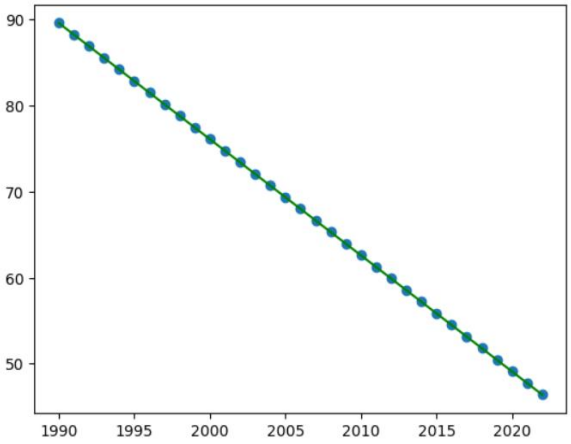
In [33]:
```
material2_price_predict_df = pd.DataFrame.from_dict(material2_price_predict_last , orient='index', columns=['material 2']).reset
```

پیش‌بینی این مدل به‌صورت زیر می‌شود:

	index	material 2
0	2023	225.393644
1	2024	229.571306
2	2025	233.603251
3	2026	237.464462
4	2027	241.127785
5	2028	244.563780
6	2029	247.740555
7	2030	250.623598
8	2031	253.175594
9	2032	255.356231
10	2033	257.121992
11	2034	258.425938
12	2035	259.217475
13	2036	259.442100
14	2037	259.041144
15	2038	257.951487
16	2039	256.105262
17	2040	253.429535
18	2041	249.845973
19	2042	245.270480
20	2043	239.612824
21	2044	232.776225
22	2045	224.656935
23	2046	215.143774
24	2047	204.117656
25	2048	191.451066
26	2049	177.007523
27	2050	160.640999

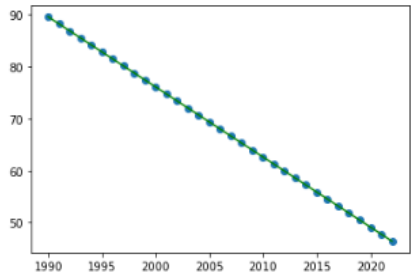
نمودار قیمت هر واحد ماده اولیه ۲ و مدل پیش‌بینی‌شده

در این قسمت با استفاده از کتابخانه matplotlib، نمودار شامل داده‌های قیمت هر واحد ماده‌ی اولیه ۲ و خط رگرسیون به‌دست امده، به‌صورت زیر است. همچنین دقت مدل به‌دست امده، برابر با ۱ است.



6.Visualization Predicted Price Material 2

```
In [35]: #model1.predict(X1)

In [36]: plt.scatter( data_frame2['Years'] , material2_price_predict_first )

         plt.plot( data_frame2['Years'] , model1.predict(X1) , c = 'green')
         plt.show()
```




```
In [37]: score1 = model1.score(X1,Y1)
         #score1

In [38]: score1
Out[38]: 1.0
```

تابع‌نمایی

اوریم. برای این کار در این قسمت خروجی تابع داده شده، که تقاضا ضریب از ان است را به درست می درست کرده، و با استفاده از کتابخانه exponential_factorیک ستون جدید در فایل اکسل به نام اوریم.دست می این ضریب را بهexp و تابع numpy

exponential_factor
5.19E-18
5.09E-18
4.99E-18
4.89E-18
4.79E-18
4.70E-18
4.60E-18
4.51E-18
4.42E-18
4.33E-18
4.25E-18
4.16E-18
4.08E-18
4.00E-18
3.92E-18
3.84E-18
3.77E-18
3.69E-18
3.62E-18
3.55E-18
3.48E-18
3.41E-18
3.34E-18
3.28E-18
3.21E-18
3.15E-18
3.08E-18
3.02E-18
2.96E-18
2.91E-18

2.85E-18
2.79E-18
2.74E-18

7.demand_factor column

```
In [39]: data_frame3 = pd.read_excel('New_data.xlsx')

In [40]: #Adding an exponential_factor column to the DataFrame using an exponential decay formula
         data_frame3['exponential_factor'] = np.exp(-(data_frame3['Years'] - 1990)  / 50)

In [41]: data_frame3.to_excel('New_data.xlsx')
```

داده جدید را در New_data ذخیره می‌کنیم.

پیش‌بینی تقاضا

حال مدل رگرسیون خطی را بر روی این ضریب به‌دست امده و تقاضاها فیت می‌کنیم و مقدار پیش‌بینی شده تقاضا را در سال ۲۰۲۳ تا ۲۰۵۰ به‌دست می‌اوریم. این تقاضا ضریب از تابع داده شده در قسمت قبل است.

8.Predict demand

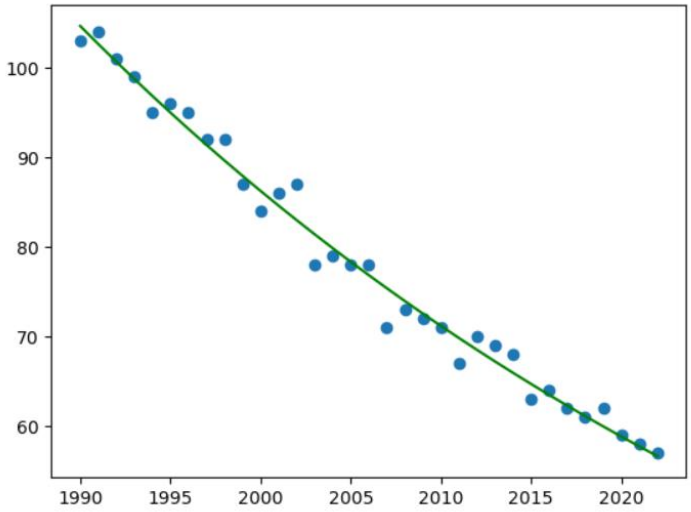
```
In [42]: X3 = data_frame3['exponential_factor'].values.reshape(-1,1)
         Y3 = data_frame3['demand'].values.reshape(-1,1)

In [43]: model3 = LinearRegression()

In [44]: model3.fit(X3,Y3)

Out[44]: ▾ LinearRegression
         LinearRegression()

In [45]: intercept3 = model3.intercept_[0]
         coefficient3 = model3.coef_[0][0]

In [46]: #predicting the demand of material 2 for years 2023 to 2050
         predict_demand = {}

         for year in range(2023, 2051) :
             predict_demand[year] = (intercept3 + coefficient3 * np.exp(-(year - 1990)/ 50))

         #predict_demand

In [47]: predict_demand_df = pd.DataFrame.from_dict(predict_demand , orient='index', columns=['demand']).reset_
```

خروجی این مدل به‌صورت زیر می‌شود:

	index	demand
0	2023	55.655541
1	2024	54.618177
2	2025	53.601354
3	2026	52.604666
4	2027	51.627714
5	2028	50.670106
6	2029	49.731461
7	2030	48.811401
8	2031	47.909561
9	2032	47.025577
10	2033	46.159098
11	2034	45.309777
12	2035	44.477273
13	2036	43.661253
14	2037	42.861392
15	2038	42.077370
16	2039	41.308872
17	2040	40.555591
18	2041	39.817226
19	2042	39.093482
20	2043	38.384069
21	2044	37.688703
22	2045	37.007107
23	2046	36.339007
24	2047	35.684136
25	2048	35.042232
26	2049	34.413039
27	2050	33.796305

نمودار تقاضا و مدل پیش‌بینی‌شده

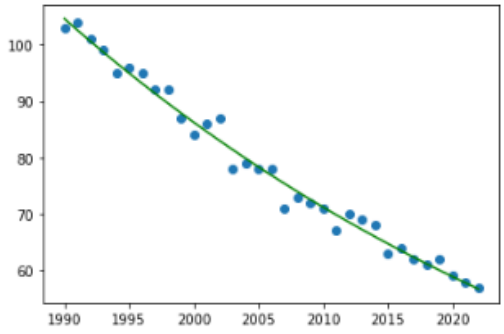
نمودار شامل داده‌های تقاضا و خط رگرسیون به‌دست امده، به‌صورت زیر است. همچنین دقت مدل به‌دست امده، تقریباً برابر با 0.984957 است.



9.Visualization Predicted demand

In [49]:
```
#model3.predict(X3)
```

In [50]:
```
plt.scatter( data_frame2['Years'] , data_frame2['demand'] )

plt.plot( data_frame2['Years'] , model3.predict(X3) , c = 'green')
plt.show()
```



In [51]:
```
score3 = model3.score(X3,Y3)
```

In [52]:
```
score3
```

Out[52]: 0.9849573838479752

پیش‌بینی قیمت هر واحد محصول نهایی

در این قسمت هم مثل قسمت‌های قبل بر روی داده قیمت مدل رگرسیون خطی فیت می‌کنیم و سپس برای سال‌های ۲۰۲۳ تا ۲۰۵۰ قیمت را پیش‌بینی می‌کنیم.

10.Predict Price

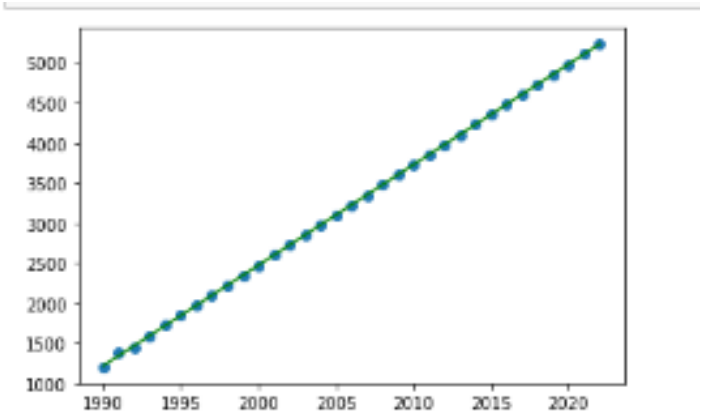
```
In [53]: def predict(x,intercept,coefficient):
             return intercept + coefficient * x

In [54]: X4 = data_frame["Years"].values.reshape(-1,1)
         Y4 = data_frame["price"].values.reshape(-1,1)

In [55]: model4 = LinearRegression()

In [56]: model4.fit(X4,Y4)

Out[56]: ▾ LinearRegression
         LinearRegression()

In [57]: intercept4 = model4.intercept_[0]
         coefficient4 = model4.coef_[0][0]

In [58]: #predicting the price for years 2023 to 2050

         predicted_price = pd.DataFrame({'Years' : [i for i in range(2023, 2051)],
                                         'price': [predict(x,intercept4,coefficient4) for x in range(2023,2051)]})
```

خروجی این مدل به‌صورت زیر می‌شود:

	Years	price
0	2023	5358.0
1	2024	5483.5
2	2025	5609.0
3	2026	5734.5
4	2027	5860.0
5	2028	5985.5
6	2029	6111.0
7	2030	6236.5
8	2031	6362.0
9	2032	6487.5
10	2033	6613.0
11	2034	6738.5
12	2035	6864.0
13	2036	6989.5
14	2037	7115.0
15	2038	7240.5
16	2039	7366.0
17	2040	7491.5
18	2041	7617.0
19	2042	7742.5
20	2043	7868.0
21	2044	7993.5
22	2045	8119.0
23	2046	8244.5
24	2047	8370.0
25	2048	8495.5
26	2049	8621.0
27	2050	8746.5

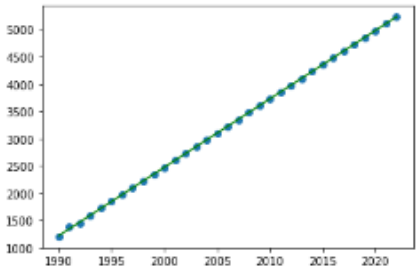
نمودار قیمت هر واحد محصول نهایی و مدل پیش‌بینی

نمودار شامل داده‌های قیمت محصول نهایی و خط رگرسیون به‌دست امده، به‌صورت زیر است. همچنین دقت مدل به‌دست امده، تقریباً برابر با 0.999965 است.



11.Visualization Predicted price

```
In [60]: plt.scatter(data_frame3['Years'] , data_frame3['price'] )

         plt.plot( data_frame3['Years'] , model4.predict(X4) , c = 'green')
         plt.show()
```




```
In [61]: score4 = model4.score(X4,Y4)

In [62]: score4

Out[62]: 0.9999653378861689
```

خواسته‌ها

خواسته ۱

در این قسمت می‌خواهیم ببینیم که این روند تا چه سالی سودده خواهد بود.

ابتدا تمام داده‌هایی که در قسمت‌های قبل پیش‌بینی کردیم را با هم ترکیب می‌کنیم تا در غالب یک دیتا فریم آن را داشته باشیم و ستون‌های مورد نیاز را فقط نگه می‌داریم.

Question1

```
In [63]: dicted data into a single DataFrame.
         d.concat([predict_salary_df, material1_price_predict_df , material2_price_predict_df ,predict_demand_df,predicted_price], axis=1)
```

```
In [95]: predicted_data.head()
```

Out[95]:

	Years	Salary	price	demand	material 1	material 2
0	2023	119794.784091	5358.0	55.655541	1368.117977	225.393644
1	2024	122370.771390	5483.5	54.618177	1450.205055	229.571306
2	2025	124946.758690	5609.0	53.601354	1537.217358	233.603251
3	2026	127522.745989	5734.5	52.604666	1629.450400	237.464462
4	2027	130098.733289	5860.0	51.627714	1727.217424	241.127785

```
In [65]: predicted_data = predicted_data[["Years","Salary","price","demand","material 1","material 2"]]
```

```
In [96]: predicted_data.head()
```

Out[96]:

	Years	Salary	price	demand	material 1	material 2
0	2023	119794.784091	5358.0	55.655541	1368.117977	225.393644
1	2024	122370.771390	5483.5	54.618177	1450.205055	229.571306
2	2025	124946.758690	5609.0	53.601354	1537.217358	233.603251
3	2026	127522.745989	5734.5	52.604666	1629.450400	237.464462
4	2027	130098.733289	5860.0	51.627714	1727.217424	241.127785

حال این جدول را با جدول داده‌های اولیه قبل از پیش‌بینی ترکیب می‌کنیم.

```
In [67]: #combining the actuall data and predicted one
         final = pd.concat([data_frame, predicted_data] , axis = 0).reset_index()
```

```
In [68]: final
```

Out[68]:

	index	Years	Salary	material 1	material 2	demand	price
0	0	1990	39343.000000	200.000000	89.600000	103.000000	1200.0
1	1	1991	46205.000000	212.000000	92.662500	104.000000	1375.0
2	2	1992	37731.000000	224.720000	95.807250	101.000000	1451.0
3	3	1993	43525.000000	238.203200	99.034819	99.000000	1593.0
4	4	1994	39891.000000	252.495392	102.345626	95.000000	1718.5
...	...	...	...	...	...	...	...
56	23	2046	179042.491979	5225.868178	215.143774	36.339007	8244.5
57	24	2047	181618.479278	5539.420269	204.117656	35.684136	8370.0
58	25	2048	184194.466578	5871.785485	191.451066	35.042232	8495.5
59	26	2049	186770.453877	6224.092614	177.007523	34.413039	8621.0
60	27	2050	189346.441176	6597.538171	160.640999	33.796305	8746.5

61 rows × 7 columns

حال باید سود هر سال را به‌دست آوریم. می‌دانیم که هزینه‌ی تمام شده به‌ازای خرید ماد هی اولیه 2 بستگی به تعداد تقاضا دارد به‌طوری که اگر میزان تقاضا بیشتر از 60 باشد 18 درصد تخفیف، اگر بین 40 تا خود عدد 60 باشد 10 درصد تخفیف، اگر میزان تقاضا بین 20 تا خود عدد 40 باشد 5 درصد تخفیف و اگر تقاضا کمتر یا مساوی 20 باشد تخفیف نمی‌گیرد.

پس یک تابع تعریف می‌کنیم که با توجه به میزان تقاضا درصد تخفیف را خروجی دهد.

```
In [69]: #Defines a discount function based on demand
         def discount(demand):
             if demand > 60:
                 discount_rate = 0.18
             elif demand > 40 and demand <= 60:
                 discount_rate = 0.1
             elif demand > 20 and demand <= 40:
                 discount_rate = 0.05
             else:
                 discount_rate = 0
             return discount_rate
```

حال یک ستون جدید به اسم New Price material 2 تعریف می‌کنیم و بر روی هر سطر قیمت جدید ماده اولیه ۲ را به‌دست می‌آوریم و در این ستون جدید ذخیره می‌کنیم.

```
In [70]: final["New Price material 2"] = final.apply(lambda row : row["material 2"] * (1 -discount(row["demand"])), axis=1)

In [71]: final
```

Out[71]:

	index	Years	Salary	material 1	material 2	demand	price	New Price material 2
0	0	1990	39343.000000	200.000000	89.600000	103.000000	1200.0	73.472000
1	1	1991	46205.000000	212.000000	92.662500	104.000000	1375.0	75.983250
2	2	1992	37731.000000	224.720000	95.807250	101.000000	1451.0	78.561945
3	3	1993	43525.000000	238.203200	99.034819	99.000000	1593.0	81.208551
4	4	1994	39891.000000	252.495392	102.345626	95.000000	1718.5	83.923414
...	...	...	...	...	...	...	...	...
56	23	2046	179042.491979	5225.868178	215.143774	36.339007	8244.5	204.386586
57	24	2047	181618.479278	5539.420269	204.117656	35.684136	8370.0	193.911773
58	25	2048	184194.466578	5871.785485	191.451066	35.042232	8495.5	181.878513
59	26	2049	186770.453877	6224.092614	177.007523	34.413039	8621.0	168.157147
60	27	2050	189346.441176	6597.538171	160.640999	33.796305	8746.5	152.608949

61 rows × 8 columns

حال که همه هزینه‌ها و قیمت‌ها را داریم سود را محاسبه می‌کنیم.

سود به‌صورت زیر محاسبه می‌شود:

$$\text{سود} = \text{تقاضا} \times \text{قیمت هر واحد محصول نهایی} - \text{تقاضا} \times \text{قیمت هر واحد ماده اولیه } 1 - \text{حقوق کارکنان} - \text{تقاضا} \times \text{قیمت هر واحد ماده اولیه } 2 -$$

61 rows x 8 columns

```
In [72]: #calculating profit
final["profit"] = (final["demand"] * (final["price"] -  final["material 1"] - final["New Price material 2"] ) )- (final["Salary"]
```

```
In [73]: final
```

Out[73]:

	index	Years	Salary	material 1	material 2	demand	price	New Price material 2	profit
0	0	1990	39343.000000	200.000000	89.600000	103.000000	1200.0	73.472000	56089.384000
1	1	1991	46205.000000	212.000000	92.662500	104.000000	1375.0	75.983250	66844.742000
2	2	1992	37731.000000	224.720000	95.807250	101.000000	1451.0	78.561945	78188.523555
3	3	1993	43525.000000	238.203200	99.034819	99.000000	1593.0	81.208551	82560.236614
4	4	1994	39891.000000	252.495392	102.345626	95.000000	1718.5	83.923414	91406.713475
...	...	...	...	...	...	...	...	...	...
56	23	2046	179042.491979	5225.868178	215.143774	36.339007	8244.5	204.386586	-76775.616023
57	24	2047	181618.479278	5539.420269	204.117656	35.684136	8370.0	193.911773	-87531.262071
58	25	2048	184194.466578	5871.785485	191.451066	35.042232	8495.5	181.878513	-98627.082397
59	26	2049	186770.453877	6224.092614	177.007523	34.413039	8621.0	168.157147	-110072.384452
60	27	2050	189346.441176	6597.538171	160.640999	33.796305	8746.5	152.608949	-121877.090065

61 rows × 9 columns

برای این‌که ببینیم تا کجا سودده بوده است باید ببینیم تا چه سالی سود مثبت بوده است و بعد از آن صفر یا منفی شده‌است.

```
In [74]: #Identifies years where the profit is negative
final[final["profit"] <= 0]
```

Out[74]:

	index	Years	Salary	material 1	material 2	demand	price	New Price material 2	profit
48	15	2038	158434.593583	3278.774346	257.951487	42.077370	7240.5	232.156339	-1504.126589
49	16	2039	161010.580882	3475.500807	256.105262	41.308872	7366.0	230.494736	-9819.926295
50	17	2040	163586.568182	3684.030855	253.429535	40.555591	7491.5	228.086582	-18422.593016
51	18	2041	166162.555481	3905.072706	249.845973	39.817226	7617.0	237.353674	-27814.671743
52	19	2042	168738.542781	4139.377069	245.270480	39.093482	7742.5	233.006956	-36988.974506
53	20	2043	171314.530080	4387.739693	239.612824	38.384069	7868.0	227.632182	-46465.427902
54	21	2044	173890.517380	4651.004074	232.776225	37.688703	7993.5	221.137414	-56250.562906
55	22	2045	176466.504679	4930.064319	224.656935	37.007107	8119.0	213.424088	-66351.429998
56	23	2046	179042.491979	5225.868178	215.143774	36.339007	8244.5	204.386586	-76775.616023
57	24	2047	181618.479278	5539.420269	204.117656	35.684136	8370.0	193.911773	-87531.262071
58	25	2048	184194.466578	5871.785485	191.451066	35.042232	8495.5	181.878513	-98627.082397
59	26	2049	186770.453877	6224.092614	177.007523	34.413039	8621.0	168.157147	-110072.384452
60	27	2050	189346.441176	6597.538171	160.640999	33.796305	8746.5	152.608949	-121877.090065

تا سال 2037 سودده بوده است چون بعد از این سال سود منفی میشود.

با توجه به این جدول تا سال ۲۰۳۷ سود مثبت بوده است و بعد از این تاریخ سود منفی بوده است پس تا سال ۲۰۲۳۷ سودده بوده است.

خواسته ۲

این ذی‌نفع می‌خواهد ۲۰ درصد از ۲۵ درصد سود خود را از سال ۲۰۲۲ تا ۲۰۳۲ پس‌انداز کند. نرخ بهره نیز ۵ درصد است.

پس ابتدا مقدار سود هر ساله‌اش را که می‌خواهد پس‌انداز کند را به‌دست می‌آوریم و در ستون saved ذخیره می‌کنیم و مقدار آن را در سال صفر به‌دست می‌آوریم که از تابع f_to_p در فایل factors استفاده می‌کنیم.

Question2

In [99]:
```
table = final[final["Years"] <= 2022]
```

In [100]:
```
table.head()
```

Out[100]:

	index	Years	Salary	material 1	material 2	demand	price	New Price material 2	profit
0	0	1990	39343.0	200.000000	89.600000	103.0	1200.0	73.472000	56089.384000
1	1	1991	46205.0	212.000000	92.662500	104.0	1375.0	75.983250	66844.742000
2	2	1992	37731.0	224.720000	95.807250	101.0	1451.0	78.561945	78188.523555
3	3	1993	43525.0	238.203200	99.034819	99.0	1593.0	81.208551	82560.236614
4	4	1994	39891.0	252.495392	102.345626	95.0	1718.5	83.923414	91406.713475

In [101]:
```
#Calculating the savings as 25% of the profit, further reduced by 20%
table["saved"] = table["profit"] * 0.25 * 0.2
```

In [103]:
```
table.head()
```

Out[103]:

	index	Years	Salary	material 1	material 2	demand	price	New Price material 2	profit	saved
0	0	1990	39343.0	200.000000	89.600000	103.0	1200.0	73.472000	56089.384000	2804.469200
1	1	1991	46205.0	212.000000	92.662500	104.0	1375.0	75.983250	66844.742000	3342.237100
2	2	1992	37731.0	224.720000	95.807250	101.0	1451.0	78.561945	78188.523555	3909.426178
3	3	1993	43525.0	238.203200	99.034819	99.0	1593.0	81.208551	82560.236614	4128.011831
4	4	1994	39891.0	252.495392	102.345626	95.0	1718.5	83.923414	91406.713475	4570.335674

In [79]:
```
#This loop calculates the present value (PV) of the profits using a discount rate (i) of 5%
i = 0.05
PV = 0
for n,f in enumerate(table["profit"]):
    PV += f_to_p(i, n, f)
```

In [80]:
```
PV
```

Out[80]:
```
1725861.5667714828
```

مقدار ارزش فعلی مقدار سودی که این شخص می‌خواهد ذخیره کند 1725861.56677 می‌باشد. حال باید مقدار ثابت را در سال‌های ۲۰۲۲ تا ۲۰۳۲ به‌دست آوریم. پس از تابع p_to_a در فایل توابع استفاده می‌کنیم.

In [81]:
```
n = 2032 - 2022 + 1
saving = p_to_a(i, n, PV)
```

In [82]:
```
#Calculating the annuity value (saving) for the period from 2022 to 2032 using the present value (PV)
saving
```

Out[82]:
```
207774.56088996632
```

این شخص باید از سال 2022 تا 2032 مقدار ثابت 20774.56 دلار پس انداز کند.

این شخص باید از سال ۲۰۲۲ تا ۲۰۳۲ مقدار ثابت ۲۰۷۷۴,۵۶ دلار پس‌انداز کند.

خواسته ۳

برای این‌که ببینیم اگر شخصی بخواهد در سال 2022 این شرکت را خریداری کند، با چه قیمتی شرکت را بخرد تا حداقل 250000 دلار(در سال پایه) به او سود دهد، باید ابتدا تمامی سودها را به سال پایه2022 با تابع p_to_f ببریم و با توجه به سود ۲۵۰۰۰۰ دلاری مقدار خرید شرکت را در سال پایه به‌دست آوریم. نرخ بهره نیز ۵ درصد است.

Question3

```
In [96]: #This loop calculates the future value (FV) of the profits using a discount rate (i) of 5%.
         i = 0.05
         FV = 0
         for n,p in enumerate(table["profit"]):
             FV += p_to_f(i, 32 - n , p)

In [97]: FV
Out[97]: 8223629.348578641

In [98]: profit = 250000

In [99]: #The expected profit is given as 250,000.
         #The final price is calculated by subtracting the expected profit from the future value (FV)
         price = FV - profit

In [100]: price
Out[100]: 7973629.348578641
```

خرید این شخص باید7973629.348دلار باشد تا 250000دلار در سال پایه سود کند.

خرید این شخص باید۷۹۷۳۶۲۹,۳۴۸دلار باشد تا ۲۵۰۰۰۰دلار در سال پایه سود کند.