TUGAS KELOMPOK KELAS

JUPYTER XXI

Nama Kelompok :

- MIKO ARYADI
- ROSSY PRIMA NADA UTAMI
- IRANA PUTRI JULIANI
- AHMAD FAISAL SIREGAR
- DIANA EKA RIYANI
- NONI KURNIA DEWI
- WAFA SANDWI MUSTHOFA
- IKHRAM PRATAMA RAMADHAN
- EVIDA OKTAVIANA
- FITRI MARHAMAH SUPRIYADI
- MUHAMMAD JAFAR SHODIQ
- MOHAMMAD LUTHFAN FAOHAN
- NYAYU CHIKA MARSELIN

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
import statsmodels.api as sm
from scipy import stats
import seaborn as sns
%matplotlib inline
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.metrics import mean_squared_error, mean_absolute_error
from sklearn.model_selection import train_test_split
```

```
! wget -O Crime_R.csv https://www.sheffield.ac.uk/polopoly_fs/1.937192!/file/Crime_R.csv
```

```
--2022-03-15 11:19:51--  https://www.sheffield.ac.uk/polopoly_fs/1.937192!/file/Crime_R.csv
Resolving www.sheffield.ac.uk (www.sheffield.ac.uk)... 143.167.2.102
Connecting to www.sheffield.ac.uk (www.sheffield.ac.uk)|143.167.2.102|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4881 (4.8K) [text/csv]
Saving to: 'Crime_R.csv'

Crime_R.csv         100%[===================>]   4.77K  --.-KB/s    in 0s

2022-03-15 11:19:52 (934 MB/s) - 'Crime_R.csv' saved [4881/4881]
```

```
[ ] reglin = pd.read_csv('Crime_R.csv')
```

```
[ ] reglin.columns
```

```
Index(['CrimeRate', 'Youth', 'Southern', 'Education', 'ExpenditureYear0',
       'LabourForce', 'Males', 'MoreMales', 'StateSize', 'YouthUnemployment',
       'MatureUnemployment', 'HighYouthUnemploy', 'Wage', 'BelowWage',
       'CrimeRate10', 'Youth10', 'Education10', 'ExpenditureYear10',
       'LabourForce10', 'Males10', 'MoreMales10', 'StateSize10',
       'YouthUnemploy10', 'MatureUnemploy10', 'HighYouthUnemploy10', 'Wage10',
       'BelowWage10'],
      dtype='object')
```

```
[ ] reglin.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 47 entries, 0 to 46
Data columns (total 27 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   CrimeRate           47 non-null     float64
 1   Youth               47 non-null     int64
 2   Southern            47 non-null     int64
 3   Education           47 non-null     float64
 4   ExpenditureYear0    47 non-null     int64
 5   LabourForce         47 non-null     int64
 6   Males               47 non-null     int64
 7   MoreMales           47 non-null     int64
 8   StateSize           47 non-null     int64
 9   YouthUnemployment   47 non-null     int64
 10  MatureUnemployment  47 non-null     int64
 11  HighYouthUnemploy   47 non-null     int64
 12  Wage                47 non-null     int64
 13  BelowWage           47 non-null     int64
 14  CrimeRate10         47 non-null     float64
 15  Youth10             47 non-null     int64
 16  Education10         47 non-null     float64
 17  ExpenditureYear10   47 non-null     int64
 18  LabourForce10       47 non-null     int64
 19  Males10             47 non-null     int64
 20  MoreMales10         47 non-null     int64
 21  StateSize10         47 non-null     int64
 22  YouthUnemploy10     47 non-null     int64
 23  MatureUnemploy10    47 non-null     int64
 24  HighYouthUnemploy10 47 non-null     int64
 25  Wage10              47 non-null     int64
 26  BelowWage10         47 non-null     int64
dtypes: float64(4), int64(23)
memory usage: 10.0 KB
```

1. Korelasi

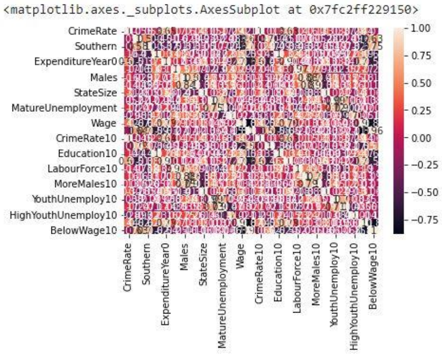
```
[ ] reglin.corr()
```

	CrimeRate	Youth	Southern	Education	ExpenditureYear0	LabourForce	Males	MoreMales	StateSize	YouthUnemployment	...	Expenditure\
CrimeRate	1.000000	-0.055002	-0.053465	0.157005	0.646211	0.169309	0.157113	0.141546	0.307945	-0.050613	...	0.6
Youth	-0.055002	1.000000	0.584355	-0.404477	-0.505737	-0.160949	-0.028680	-0.048581	-0.280638	-0.224381	...	-0.5
Southern	-0.053465	0.584355	1.000000	-0.496831	-0.372636	-0.505469	-0.314733	-0.349630	-0.049918	-0.172419	...	-0.3
Education	0.157005	-0.404477	-0.496831	1.000000	0.300018	0.427860	0.272360	0.057403	-0.001403	-0.026598	...	0.3
ExpenditureYear0	0.646211	-0.505737	-0.372636	0.300018	1.000000	0.121493	0.033760	0.036784	0.526284	-0.043698	...	0.9
LabourForce	0.169309	-0.160949	-0.505469	0.427860	0.121493	1.000000	0.513559	0.366911	-0.123672	-0.229400	...	0.1
Males	0.157113	-0.028680	-0.314733	0.272360	0.033760	0.513559	1.000000	0.836195	-0.410628	0.351892	...	0.0
MoreMales	0.141546	-0.048581	-0.349630	0.057403	0.036784	0.366911	0.836195	1.000000	-0.351102	0.429861	...	0.0
StateSize	0.307945	-0.280638	-0.049918	-0.001403	0.526284	-0.123672	-0.410628	-0.351102	1.000000	-0.038120	...	0.5
YouthUnemployment	-0.050613	-0.224381	-0.172419	-0.026598	-0.043698	-0.229400	0.351892	0.429861	-0.038120	1.000000	...	-0.0
MatureUnemployment	0.171835	-0.244843	0.071693	-0.222656	0.185093	-0.420762	-0.018692	0.059487	0.270422	0.745925	...	0.1
HighYouthUnemploy	-0.286033	-0.083029	-0.395545	0.310001	-0.239076	0.413431	0.386228	0.362814	-0.365080	0.076718	...	-0.2
Wage	0.424853	-0.670055	-0.636945	0.519187	0.787225	0.294632	0.179609	0.109640	0.308263	0.044857	...	0.7
BelowWage	-0.167318	0.639211	0.737181	-0.588214	-0.630500	-0.269886	-0.167089	-0.120562	-0.126294	-0.063832	...	-0.6
CrimeRate10	0.996596	-0.044827	-0.028568	0.142139	0.643465	0.141884	0.148907	0.141896	0.323885	-0.046139	...	0.6
Youth10	-0.015760	0.790785	0.471938	-0.360348	-0.452323	-0.130551	-0.047824	-0.069045	-0.180831	-0.080064	...	-0.4
Education10	0.142153	-0.397194	-0.489556	0.995153	0.278156	0.431150	0.262139	0.041105	-0.009439	-0.040403	...	0.2
ExpenditureYear10	0.629700	-0.513173	-0.376168	0.318456	0.993586	0.106350	0.022843	0.040843	0.513789	-0.051712	...	1.0
LabourForce10	0.138849	-0.073214	-0.478050	0.427725	0.047956	0.974818	0.535478	0.385071	-0.175913	-0.237177	...	0.0
Males10	0.163331	0.059896	-0.329930	0.173557	-0.003897	0.474179	0.882966	0.796917	-0.378248	0.321949	...	-0.0
MoreMales10	0.125157	0.005249	-0.263777	-0.056985	-0.012378	0.286188	0.745046	0.936117	-0.342549	0.449803	...	-0.0
StateSize10	0.303974	-0.282565	-0.054286	-0.000550	0.531186	-0.125571	-0.408964	-0.346462	0.999371	-0.040807	...	0.5
YouthUnemploy10	-0.038185	-0.201452	-0.186064	0.009220	-0.038003	-0.200756	0.373205	0.431853	-0.020832	0.991235	...	-0.0
MatureUnemploy10	0.165357	-0.244174	0.071269	-0.171931	0.135443	-0.347803	0.035302	0.082788	0.201788	0.726123	...	0.1
HighYouthUnemploy10	-0.281453	-0.097341	-0.317315	0.233205	-0.157784	0.198845	0.247763	0.260228	-0.143571	0.061028	...	-0.1
Wage10	0.436740	-0.615583	-0.615912	0.485811	0.787296	0.272300	0.177925	0.106842	0.308716	0.012540	...	0.7
BelowWage10	-0.076246	0.633203	0.751462	-0.623747	-0.538800	-0.254833	-0.161323	-0.105233	-0.060654	-0.087397	...	-0.5

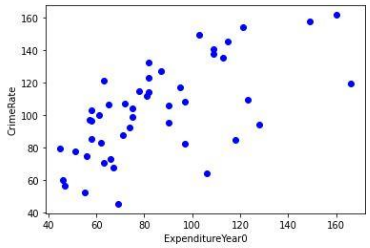
27 rows × 27 columns

```
sns.heatmap(reglin.corr(), annot=True)
```

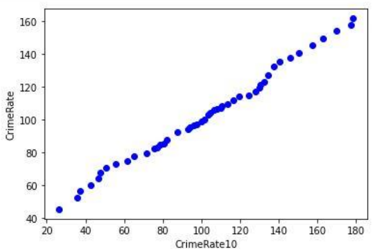
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc2ff229150>
```



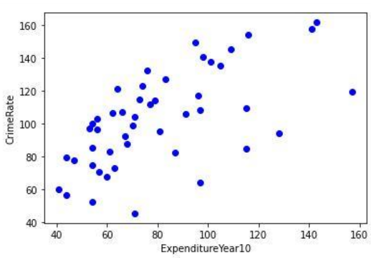
```
plt.scatter(reglin['ExpenditureYear0'], reglin['CrimeRate'], color='blue')
plt.xlabel("ExpenditureYear0")
plt.ylabel("CrimeRate")
plt.show()
```



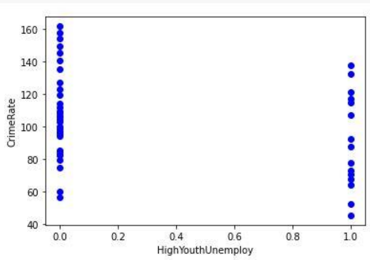

```
plt.scatter(reglin['CrimeRate10'], reglin['CrimeRate'], color='blue')
plt.xlabel("CrimeRate10")
plt.ylabel("CrimeRate")
plt.show()
```




```
plt.scatter(reglin['ExpenditureYear10'], reglin['CrimeRate'], color='blue')
plt.xlabel("ExpenditureYear10")
plt.ylabel("CrimeRate")
plt.show()
```




```
plt.scatter(reglin['HighYouthUnemploy'], reglin['CrimeRate'], color='blue')
plt.xlabel("HighYouthUnemploy")
plt.ylabel("CrimeRate")
plt.show()
```



Terlihat bahwa CrimeRate10, ExpenditureYear0, dan HighYouthUnemploy plotnya bervariasi. Untuk plot CrimeRate10 (0.996596) membentuk garis yang menunjukkan memiliki hubungan yang kuat dengan crimerate, sementara ExpenditureYear0 (0.646211) dan ExpenditureYear10 (0.629700) plotnya terlihat cenderung membentuk garis yang menunjukkan masih ada korelasi dengan crimerate, sedangkan untuk HighYouthUnemploy (-0.286033) tersebar kekiri dan ada kekanan tidak menentu yang artinya tidak ada korelasi terhadap crimerate. Jadi yang berkorelasi kuat itu apa bila memiliki nilai korelasi mendekati 1 sedangkan menjauhi 1 bahkan ke arah negatif maka korelasinya semakin rendah.

2. Model Regresi

```
#Splitting data
features = ['ExpenditureYear0', 'CrimeRate10','ExpenditureYear10']
X = reglin[features].values
Y= reglin.CrimeRate

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=23)
```

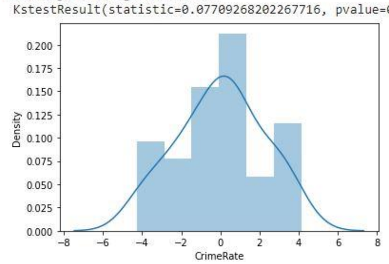
```
lin_reg = LinearRegression()
lin_reg.fit(X_train, Y_train)
```

```
LinearRegression()
```

```
y_predtrain = lin_reg.predict(X_train)
err = y_predtrain - Y_train
sns.distplot(err)

z_er = stats.zscore(err)
norm_er = stats.kstest(z_er, 'norm')
print('hasil uji Kolomogorov Smirnov\n', norm_er)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version.
  warnings.warn(msg, FutureWarning)
hasil uji Kolomogorov Smirnov
 KstestResult(statistic=0.07709268202267716, pvalue=0.9804350812762913)
```

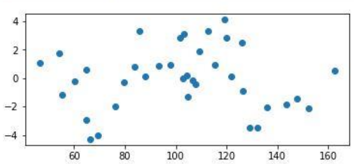


Ho : data berdistribusi normal

Ha : data tidak berdistribusi normal

Karena pvaiue > 0.05 (0.9804350812762913 > 0.05), maka Ho diterima. Gambar histogram yang membentuk lonceng dan puncaknya cenderung ke tengah pun mendukung, maka disimpulkan datanya sudah berdistribusi normal.

```
fig, ax = plt.subplots(figsize=(6,2.5))
_ = ax.scatter(y_predtrain, err)
```



Berdasarkan hasil output, terlihat plot errornya berada di sekitar angka yang sama meski nilai prediksinya bertambah, artinya nilai prediksi tidak terganggu oleh errornya atau dapat dikatakan tidak terjadi heteroskedastisitas

```
vif = [variance_inflation_factor(X_train, i) for i in range(len(X_train.T))]
pd.DataFrame({'VIF': vif[0:]}, index=features).T
```

	ExpenditureYear0	CrimeRate10	ExpenditureYear10
VIF	662.553116	13.868465	676.078565

Jika nilai VIF < 10 maka artinya tidak terjadi muktikolinieritas dalam model regresi.

Jika nilai VIF > 10 maka artinya terjadi muktikolinieritas dalam model regresi Jadi berdasarkan output diatas variabel expenditureyear0, expenditureyear10 dan CrimeRate10 terjadi multikolinearitas.

```
X_constant = sm.add_constant(X_train)          #Ingat lagi x1 = R&D, x2 = Marketing
linreg = sm.OLS(Y_train,X_constant).fit()
linreg.summary()
```

OLS Regression Results

Dep. Variable:	CrimeRate	**R-squared:**	0.995
Model:	OLS	**Adj. R-squared:**	0.994
Method:	Least Squares	**F-statistic:**	2021.
Date:	Tue, 15 Mar 2022	**Prob (F-statistic):**	1.87e-37
Time:	11:19:56	**Log-Likelihood:**	-81.057
No. Observations:	37	**AIC:**	170.1
Df Residuals:	33	**BIC:**	176.6
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	27.3347	1.309	20.886	0.000	24.672	29.997
x1	0.3077	0.116	2.663	0.012	0.073	0.543
x2	0.7188	0.013	55.152	0.000	0.692	0.745
x3	-0.2959	0.126	-2.342	0.025	-0.553	-0.039

Omnibus:	0.882	**Durbin-Watson:**	1.617
Prob(Omnibus):	0.643	**Jarque-Bera (JB):**	0.821
Skew:	0.108	**Prob(JB):**	0.663
Kurtosis:	2.303	**Cond. No.**	546.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Dikarenakan angka Durbin-Watson (DW) diantara -2 sampai +2, berarti tidak ada autokorelasi

3. Hitung MSE & R2

```
y_predtest = lin_reg.predict(X_test)    # Prediksi data testing

# MSE
MSE_train = mean_squared_error(Y_train, y_predtrain)
print('Nilai MSE data training = ', MSE_train)
MSE_test = mean_squared_error(Y_test, y_predtest)
print('Nilai MSE data testing = ', MSE_test)
# RMSE
RMSE_train = np.sqrt(MSE_train)
print('Nilai RMSE data training = ', RMSE_train)
RMSE_test = np.sqrt(MSE_train)
print('Nilai RMSE data testing = ', RMSE_test)

# MAE
MAE_train = mean_absolute_error(Y_train, y_predtrain)
print('Nilai MAE data training = ', MAE_train)
MAE_test = mean_absolute_error(Y_test, y_predtest)
print('Nilai MAE data testing = ', MAE_test)
```

```
Nilai MSE data training =  4.6813506945882155
Nilai MSE data testing =  5.911297440746997
Nilai RMSE data training =  2.1636429221542577
Nilai RMSE data testing =  2.1636429221542577
Nilai MAE data training =  1.7374743775979924
Nilai MAE data testing =  1.9901716759258221
```

Berdasarkan hasil diatas, maka model MSE data training yang paling baik adalah MAE data training dengan nilai 1.7374743775979924 dan model MSE data test yang paling baik adalah MAE data testing dengan nilai 1.9901716759258221. Semakin kecil nila MSE yang didapat, maka semakin baik

```
[ ]  # Model Lasso
     Lasso_reg = Lasso(alpha=0.1).fit(X_train, Y_train)
     y_predtrain_lasso = Lasso_reg.predict(X_train)
     y_predtest_lasso = Lasso_reg.predict(X_test)

     # Model Ridge
     Ridge_reg = Ridge(alpha=0.1).fit(X_train, Y_train)
     y_predtrain_ridge = Ridge_reg.predict(X_train)
     y_predtest_ridge = Ridge_reg.predict(X_test)

     # Support Vectore Regression
     Sup_reg = SVR().fit(X_train, Y_train)
     y_predtrain_svr = Sup_reg.predict(X_train)
     y_predtest_svr = Sup_reg.predict(X_test)

     # Decision Tree Regression
     Dt_reg = DecisionTreeRegressor().fit(X_train, Y_train)
     y_predtrain_dtr = Dt_reg.predict(X_train)
     y_predtest_dtr = Dt_reg.predict(X_test)
```

```
#MSE
print('Nilai MSE data training Regresi Linier = ', mean_squared_error(Y_train, y_predtrain))
print('Nilai MSE data testing Regresi Linier = ', mean_squared_error(Y_test, y_predtest), '\n')

print('Nilai MSE data training Regresi Lasso = ', mean_squared_error(Y_train, y_predtrain_lasso))
print('Nilai MSE data testing Regresi Lasso = ', mean_squared_error(Y_test, y_predtest_lasso), '\n')

print('Nilai MSE data training Regresi Ridge = ', mean_squared_error(Y_train, y_predtrain_ridge))
print('Nilai MSE data testing Regresi Ridge = ', mean_squared_error(Y_test, y_predtest_ridge), '\n')

print('Nilai MSE data training Regresi SVR = ', mean_squared_error(Y_train, y_predtrain_svr))
print('Nilai MSE data testing Regresi SVR = ', mean_squared_error(Y_test, y_predtest_svr), '\n')
print('Nilai MSE data training Regresi DTR = ', mean_squared_error(Y_train, y_predtrain_dtr))
print('Nilai MSE data testing Regresi DTR = ', mean_squared_error(Y_test, y_predtest_dtr))
```

```
Nilai MSE data training Regresi Linier =  4.6813506945882155
Nilai MSE data testing Regresi Linier =  5.911297440746997

Nilai MSE data training Regresi Lasso =  4.684034914742768
Nilai MSE data testing Regresi Lasso =  6.000206031032483

Nilai MSE data training Regresi Ridge =  4.681350974495067
Nilai MSE data testing Regresi Ridge =  5.912237658881045

Nilai MSE data training Regresi SVR =  564.6092871935691
Nilai MSE data testing Regresi SVR =  466.8101436301151

Nilai MSE data training Regresi DTR =  0.0
Nilai MSE data testing Regresi DTR =  9.375000000000025
```

```
[ ] #Nilai R2
print(f'R^2 score Regresi Linier: {lin_reg.score(X, Y)}')
print(f'R^2 score Regresi Lasso: {Lasso_reg.score(X, Y)}')
print(f'R^2 score Regresi Ridge: {Ridge_reg.score(X, Y)}')
print(f'R^2 score Regresi SVR: {Sup_reg.score(X, Y)}')
print(f'R^2 score Regresi DT: {Dt_reg.score(X, Y)}')
```

```
R^2 score Regresi Linier: 0.9939501999204646
R^2 score Regresi Lasso: 0.9939244614611806
R^2 score Regresi Ridge: 0.9939499548135027
R^2 score Regresi SVR: 0.33444073062805235
R^2 score Regresi DT: 0.9975587054262411
```

Kesimpulan

- Model regresi Support Vector Regression memiliki MSE yang terlalu tinggi dibandingkan model regresi lainnya dan R2 yang sangat rendah, sehingga kemungkinan pada model ini terjadi underfitting
- Model regresi Decision Tree Regression R2 paling bagus, mMSE data training 0 (sangat kecil), sementara MSE data testing jauh diatasnya, ini menunjukkan pada model ini terjadi overfitting
- Model regresi linier, regresi Lasso, dan regresi Ridge memiliki MSE yang tidak jauh antara testing dan trainingnya. Nilai R2nya pun sangat bagus, hampir mendekati 1. Ini menunjukkan model ini sudah good fit

4. Visualisasi Data Prediksi

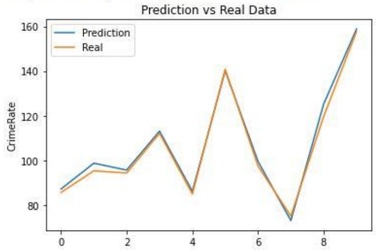
```
plt.plot(y_predtest)
plt.plot(Y_test.values)

plt.title('Prediction vs Real Data')

plt.ylabel('CrimeRate')

plt.legend(labels=['Prediction', "Real"], loc='upper left')
```

```
<matplotlib.legend.Legend at 0x7fc2fea97a50>
```



Berdasarkan chart diatas dapat disimpulkan bahwa antara prediksi dan real data nilai nya berdekatan dan determinan nilai nya mendekati 1. Artinya model regresi linier yang kita gunakan sudah termasuk baik.

KESIMPULAN

1. Uji Simultan = Terlihat nilai p-value uji-F (Prob (F-statistic)) adalah 1.87 x 10^-37 < 0.05, artinya secara bersama-sama 'ExpenditureYear0', 'CrimeRate10', 'ExpenditureYear10' berpengaruh signifikan terhadap CrimeRate
2. Uji Parsial = Terlihat nilai p-value uji-T (P>|t|) untuk ExpenditureYear0 adalah 0.012 < 0.05 ,untuk CrimeRate10 (0.000) < 0.05 , dan untuk ExpenditureYear10 (0.025) < 0.05 artinya secara sendiri-sendiri baik 'ExpenditureYear0', 'CrimeRate10', 'ExpenditureYear10' tidak memberi pengaruh yang signifikan terhadap CrimeRate, namun CrimeRate10 mungkin memiliki sedikit pengaruh karena nilai nya mendekati 0.05
3. Besar pengaruh feature = Perhatikan kolom "coef", pada x1 (ExpenditureYear0) nilainya 0.3077, artinya setiap kenaikan CrimeRate meningkatkan angka CrimeRate sebesar 0.3077. Koefisien x2 (CrimeRate10) sebesar 0.7188, setiap kenaikan CrimeRate meningkatkan angka CrimeRate sebesar 0.3077 Sedangkan pada koefisien x3 (ExpenditureYear10) sebesar -0.2959. Artinya selama ini pengaruh ExpenditureYear10 terhadap profit hanya -0.2959.