In [104]:

*#Import package pandas untuk import data*

**import**

**pandas**

**as**

**pd**

*#Import package numpy untuk membuat object menjadi n-multidimensi array*

**import**

**numpy**

**as**

**np**

*#Melakukan import data*

data1

=

pd

.

read\_csv

(

'testh2.csv'

)

*#Mengisi Nilai Attribut yang kosong dengan nilai 0*

data

=

data1

.

fillna

(

0

)

*#print(data.head())*

*#print(data.tail())*

print

(

data

)

*#Membuat array 2D dari data yang telah di import*

ArrayData

=

np

.

array

(

data

)

*#print(ArrayData)*

*#print(ArrayData.shape)*

*#Memilih X sebagai Fitur dari Array yang telah dibuat*

X

=

ArrayData

[:

,

2

:

5

]

*#Memilih Y sebagai Label dari Array yang telah dibuat*

Y

=

ArrayData

[:

,

5

:

6

]

*#print(X.shape)*

*#print(Y.shape)*

*#Mengubah Y menjadi array 1D sehingga bisa dijadikan target pada proses training dan testing*

*data*

Y\_1D

=

Y

.

ravel

()

print

(

X

)

print

(

Y

)

*#print(Y\_1D)*

b

In [105]:

**from** **sklearn.model\_selection** **import** train\_test\_split

*#Memisah data X (fitur) dan Y (label) secara random menjadi data testing dan data train ing, dimana data testing sebanyak 33% dan data training sebanyak 66% dari data asli* x\_train, x\_true, y\_train, y\_true=train\_test\_split(X,Y\_1D,test\_size=0.3) print(x\_train) print(x\_true) print(y\_train) print(y\_true)

In [106]:

**from** **sklearn.neighbors** **import** KNeighborsRegressor

*#Menggunakan KNN sebagai metode regresi*

neigh = KNeighborsRegressor(n\_neighbors=8,metric='minkowski')

*#Mencocokan model data menggunakan x\_train sebagai data latih dan y\_train sebagai nilai target*

neigh.fit(x\_train, y\_train) *#Memprediksi dari data x\_true* y\_predict = neigh.predict(x\_true) print(y\_predict)

[107]:

**from**

**sklearn.metrics**

**import**

r2\_score

print

(

"Koefisien Determinasinya Adalah"

)

*#Menghitung koefisien determinan dari data testing Y dan data prediksi Y*

print

(

r2\_score

(

y\_true

,

y\_predict

))

Koefisien Determinasinya Adalah

0.8720099599553754

In [108]:

**from**

**sklearn**

**import**

neighbors

**from**

**sklearn.metrics**

**import**

mean\_squared\_error

**from**

**math**

**import**

sqrt

**import**

**matplotlib.pyplot**

**as**

**plt**

%

**matplotlib**

inline

In [109]:

rmse\_val = [] *#Membuat Variabel dalam bentuk array untuk menyimpan nilai rmse dari k ya ng berbeda* **for** K **in** range(350):

K = K+1

*#Menggunakan KNN sebagai metode regresi*

model = neighbors.KNeighborsRegressor(n\_neighbors = K)

*#Mencocokan model data menggunakan x\_train sebagai data latih dan y\_train sebagai n ilai target*

model.fit(x\_train, y\_train) *#Memprediksi dari data x\_true* pred=model.predict(x\_true) *#Menghitung nilai RMSE setiap K*

error = sqrt(mean\_squared\_error(y\_true,pred)) *#Menyimpan nilai RMSE ke dalam array untuk setiap K* rmse\_val.append(error)

print('RMSE value for k= ' , K , 'is:', error)

In [110]:

*#Mengubah*

curve

=

pd

.

DataFrame

(

rmse\_val

)

*#Membuat grafik dari nilai RMSE*

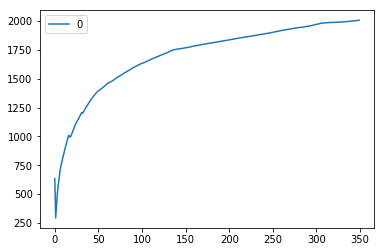
curve

.

plot

()

Out[110]: <matplotlib.axes.\_subplots.AxesSubplot at 0x1cf959c7240>



In [111]:

**from**

**sklearn.model\_selection**

**import**

GridSearchCV

params

=

{

'n\_neighbors'

:[

2

,

3

,

4

,

5

,

6

,

7

,

8

,

9

]}

knn

=

neighbors

.

KNeighborsRegressor

()

model

=

GridSearchCV

(

knn

,

params

,

cv

=

5

)

model

.

fit

(

x\_train

,

y\_train

)

model

.

best\_params\_

Out[111]:

{'n\_neighbors': 2}