Double-click (or enter) to edit

```
import numpy as np
```

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
df = pd.read_csv(r"https://github.com/YBI-Foundation/Dataset/raw/main/Boston.csv")
```

```
df.head()
```

	CRIM	ZN	INDUS	CHAS	NX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	L
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 506 entries, 0 to 505
Data columns (total 14 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   CRIM     506 non-null    float64
 1   ZN       506 non-null    float64
 2   INDUS    506 non-null    float64
 3   CHAS     506 non-null    int64  
 4   NX       506 non-null    float64
 5   RM       506 non-null    float64
 6   AGE      506 non-null    float64
 7   DIS      506 non-null    float64
 8   RAD      506 non-null    int64  
 9   TAX      506 non-null    float64
 10  PTRATIO  506 non-null    float64
 11  B        506 non-null    float64
 12  LSTAT    506 non-null    float64
 13  MEDV     506 non-null    float64
dtypes: float64(12), int64(2)
memory usage: 55.5 KB
```

```
df.describe()
```

	CRIM	ZN	INDUS	CHAS	NX	RM	AC
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.0000(
mean	3.613524	11.363636	11.136779	0.069170	0.554695	6.284634	68.5749(
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.1488(
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.9000(
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.0250(
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.5000(
75%	3.677083	12.500000	18.100000	0.000000	0.624000	6.623500	94.0750(
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.0000(

```
df.columns
```

```
Index(['CRIM', 'ZN', 'INDUS', 'CHAS', 'NX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX',
       'PTRATIO', 'B', 'LSTAT', 'MEDV'],
      dtype='object')
```

```
y = df["MEDV"]
```

```
X = df[['CRIM', 'ZN', 'INDUS', 'CHAS', 'NX', 'RM', 'AGE', 'DIS', 'RAD', 'TAX',
       'PTRATIO', 'B', 'LSTAT']]
```

```
from sklearn.model_selection import train_test_split
```

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.3,random_state =
```

```
X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

```
((354, 13), (152, 13), (354,), (152,))
```

```
from sklearn.preprocessing import StandardScaler
```

```
sc = StandardScaler()
```

```
X_train_sc = sc.fit_transform(X_train,y)
```

```
X_test_sc = sc.fit_transform(X_test,y)
```

```
X_train_sc
```

```
array([[-0.14113619, -0.48175769, -0.19860022, ...,  0.00438903,
        -0.05084503, -0.01555641],
       [-0.42121529,  3.02166196, -1.33410259, ..., -1.68641979,
         0.42969249, -1.33650784],
       [-0.41266839, -0.48175769,  0.22414717, ...,  0.14148164,
         0.19739169, -0.10842497],
       ...,
       [-0.38944304, -0.48175769, -0.19860022, ...,  0.00438903,
         0.37963873,  0.77313338],
       [-0.41404001,  0.41002186, -0.81324318, ..., -0.72677154,
         0.43161763,  0.09671754],
       [-0.41578561,  2.06618387, -1.3831586 , ..., -0.04130851,
         0.39707198, -0.68781395]])
```

```
X_test_sc
```

```
array([[-0.36714008, -0.50235603, -0.6925381 , ..., -0.57641511,
         0.2366856 , -1.24860568],
       [-0.40880876, -0.50235603, -0.58591169, ..., -0.33768188,
         0.43031542, -0.31886558],
       [-0.41291768, -0.50235603, -0.12035979, ..., -0.38542852,
         0.36717526,  0.17122998],
       ...,
       [-0.34428827, -0.50235603,  1.66375525, ...,  1.23795746,
         0.30005961, -0.18769294],
       [-0.05769974, -0.50235603,  1.31684399, ..., -1.86557456,
        -0.3514533 , -0.15886379],
       [-0.42293258,  1.25907688, -0.66100071, ..., -0.48092181,
         0.43031542, -0.75418575]])
```

```
from sklearn.neighbors import KNeighborsRegressor
```

```
knnreg = KNeighborsRegressor(n_neighbors=3)
```

```
knnreg.fit(X_train_sc,y_train)
```

```
KNeighborsRegressor(n_neighbors=3)
```

```
y_pred = knnreg.predict(X_test_sc)
```

```
from sklearn.metrics import mean_absolute_error,mean_absolute_percentage_error
```

```
mean_absolute_error(y_test,y_pred)
```

```
2.5309210526315793
```

```
 mean_absolute_percentage_error(y_test,y_pred)
```

```
0.12289250095895413
```

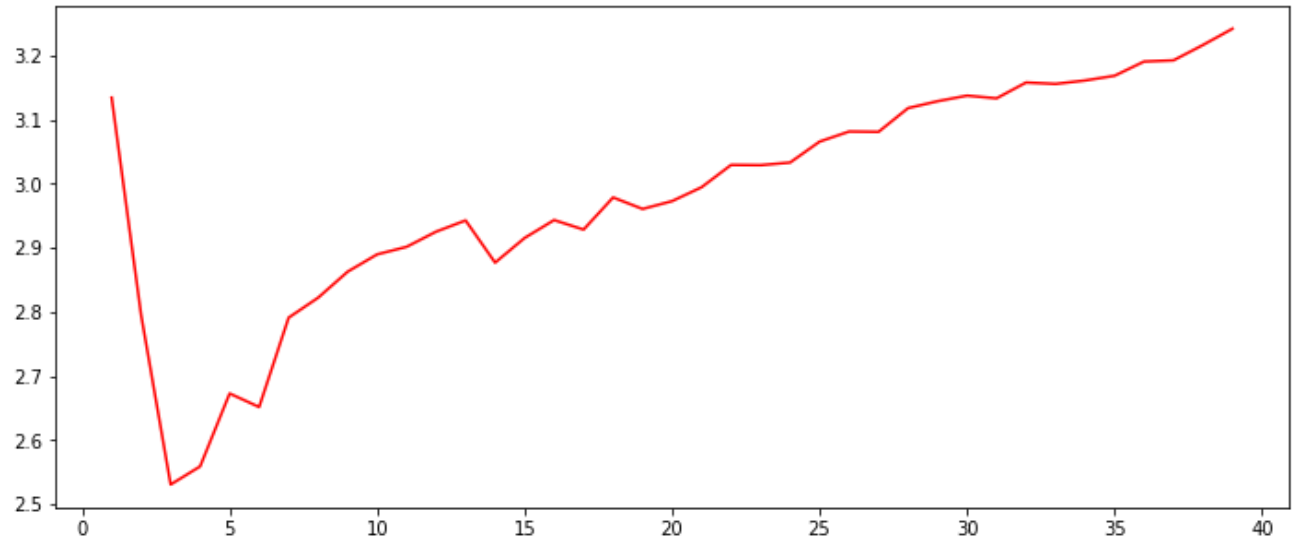
```
error_rate = []

for i in range(1,40):
  knn = KNeighborsRegressor(n_neighbors=i)
  knn.fit(X_train_sc,y_train)
  pred_i = knn.predict(X_test_sc)
  error_rate.append(mean_absolute_error(y_test,pred_i))


error_rate
```

```
[3.134210526315789,
 2.795065789473684,
 2.5309210526315793,
 2.5592105263157894,
 2.672894736842105,
 2.6517543859649124,
 2.7910714285714286,
 2.8222039473684206,
 2.8627192982456138,
 2.8897368421052634,
 2.9014952153110047,
 2.9251644736842106,
 2.9426619433198375,
 2.87687969924812,
 2.9153070175438596,
 2.943215460526315,
 2.928405572755418,
 2.9784722222222215,
 2.960560941828255,
 2.972763157894736,
 2.9944235588972425,
 3.029186602870813,
 3.02883295194508,
 3.0328673245614035,
 3.0655,
 3.0812500000000003,
 3.080750487329435,
 3.117763157894737,
 3.128493647912886,
 3.137214912280702,
 3.133000848896435,
 3.1576685855263156,
 3.1555223285486447,
 3.1607972136222906,
 3.1681766917293235,
 3.1903508771929823,
 3.192247510668563,
 3.2162049861495845,
 3.2415485829959514]
```

```
fig,ax= plt.subplots(figsize = (12,5))
ax.plot(range(1,40),error_rate,color = 'Red')
plt.show()
```




```
error
```

✓ 0s completed at 9:00 AM