k-NN Example-4

Linear Regression နှင့် k-NN Regression model တို့၏ တိကျမှု(accuracy)ကို နှိုင်းယှဉ်ခြင်း

ယခု ဥပမာတွင် Linear Regression model နှင့် k-NN Regression model တို့၏ တိကျမှု(accuracy)ကို နှိုင်းယှဉ်ရန် သရုပ်ပြထားသည့် ဥပမာဖြစ်သည်။

k အရေအတွက် ပြောင်းလဲခြင်းဖြင့် k-NN Regression model တိကျမှု(accuracy) ပြောင်းလဲသည်။ k အရေအတွက် (၄) ဖြင့် ရရှိသည့် တိကျမှု(accuracy) သည် Linear Regression model ၏ တိကျမှု(accuracy) ထက် ပိုကောင်းပုံကို သရုပ်ပြထားသည်။ ထို့ကြောင့် k-NN တွင် တိကျမှု(accuracy)ကောင်းရန်အတွက် k အရေတွက်သည် အလွန်အရေးကြီးသည့် အချက်ဖြစ်သည်။

- sklearn.datasets မှ boston ဒေတာများကို import လုပ်သည်။ sklearn မှ datasets ထဲတွင် ဒတောမြား ရှိသည်။ ထို့ကြောင့်ပြင်ပမှ ဒေတာများကို လိုက်ရှာထည့်နေရန် မလိုအပ်ပါ။

- sklearn.model_selection မှ train_test_split ကို import လုပ်သည်။ train_test_split သည် train data set နှင့် test data set များ ခွဲခြား(split)ရန် အတွက် အသုံးပြုသည်။

- sklearn.linear_model မှ LinearRegression ကို import လုပ်သည်။ LinearRegression သည် Linear Regression machine learning model တည်ဆောက်ရန်အတွက် ဖြစ်သည်။

- from sklearn.neighbors မှ KNeighborsRegressor ကို import လုပ်သည်။ KNeighborsRegressor သည် k-NN Regressor machine learning model တည်ဆောက်ရန်အတွက် ဖြစ်သည်။

In [1]:
```
from sklearn.datasets import load_boston
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
```

ဒေတာဖတ်ခြင်း

In [2]:
```
boston = load_boston()
X = boston.data
y = boston.target
print('X.shape:', X.shape)
```

```
X.shape: (506, 13)
```

ဒေတာ ခွဲခြင်း

In [3]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2
5, random_state=42)
```

Linear Regression machine learning model တည်ဆောက်ခြင်း

In [4]:
```
linreg = LinearRegression()
linreg
```

Out[4]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

Linear Regression model ထဲသို့ ဒေတာ ထည့်ခြင်း

In [5]:
```
linreg.fit(X_train, y_train)
```

Out[5]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

Linear Regression Model Train score နှင့် Test score တွက်ခြင်း

In [6]:
```
print('Linear Regression Train/Test: %.3f/%.3f' %
      (linreg.score(X_train, y_train),
       linreg.score(X_test, y_test)))
```

```
Linear Regression Train/Test: 0.748/0.684
```

Linear Regression၏ Train score မှာ 0.748 ဖြစ်ပြီး Test score မှာ 0.684 ဖြစ်သည်။

k-NN Regression machine learning model တည်ဆောက်ခြင်း(k=1)

In [7]:
```
knnreg = KNeighborsRegressor(n_neighbors=1)
knnreg
```

Out[7]:
```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski
',
          metric_params=None, n_jobs=None, n_neighbors=1, p=2,
          weights='uniform')
```

k-NN Regression model ထဲသို့ ဒေတာ ထည့်ခြင်း

In [8]:
```
knnreg.fit(X_train, y_train)
```

Out[8]:
```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski
',
          metric_params=None, n_jobs=None, n_neighbors=1, p=2,
          weights='uniform')
```

k-NN Regression ၏ Train score နှင့် Test score တွက်ခြင်း(k=1)

In [9]:
```
print('KNeighborsRegressor Train/Test: %.3f/%.3f' %
      (knnreg.score(X_train, y_train),
       knnreg.score(X_test, y_test)))
```

```
KNeighborsRegressor Train/Test: 1.000/0.481
```

k-NN Regression၏ Train score မှာ 0.699 ဖြစ်ပြီး Test score မှာ 0.653 ဖြစ်သည်။

k-NN Regression ၏ Train score နှင့် Test score တွက်ခြင်း(k=2)

In [10]:
```
knnreg = KNeighborsRegressor(n_neighbors=2)
knnreg.fit(X_train, y_train)
print('KNeighborsRegressor Train/Test: %.3f/%.3f' %
      (knnreg.score(X_train, y_train),
       knnreg.score(X_test, y_test)))
```

```
KNeighborsRegressor Train/Test: 0.840/0.554
```

k-NN Regression၏ Train score မှာ 0.840 ဖြစ်ပြီး Test score မှာ 0.554 ဖြစ်သည်။

k-NN Regression ၏ Train score နှင့် Test score တွက်ခြင်း(k=3)

In [11]:
```
knnreg = KNeighborsRegressor(n_neighbors=3)
knnreg.fit(X_train, y_train)
print('KNeighborsRegressor Train/Test: %.3f/%.3f' %
      (knnreg.score(X_train, y_train),
       knnreg.score(X_test, y_test)))
```

```
KNeighborsRegressor Train/Test: 0.752/0.695
```

k-NN Regression၏ Train score မှာ 0.752 ဖြစ်ပြီး Test score မှာ 0.695 ဖြစ်သည်။

k-NN Regression ၏ Train score နှင့် Test score တွက်ခြင်း(k=4)

In [12]:
```
knnreg = KNeighborsRegressor(n_neighbors=4)
knnreg.fit(X_train, y_train)
print('KNeighborsRegressor Train/Test: %.3f/%.3f' %
      (knnreg.score(X_train, y_train),
       knnreg.score(X_test, y_test)))
```

```
KNeighborsRegressor Train/Test: 0.699/0.653
```

k-NN Regression ၏ Train score မှာ 0.699 ဖြစ်ပြီး Test score မှာ 0.653 ဖြစ်သည်။

တွေ့ရှိချက်

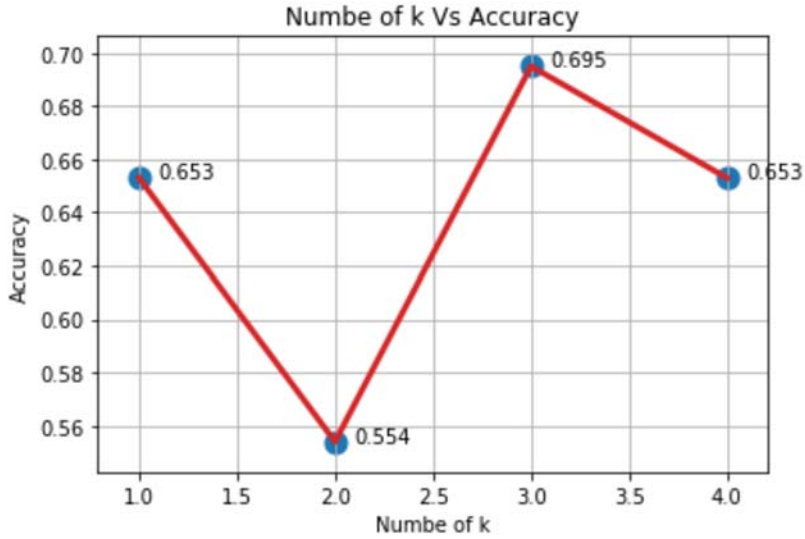
k-NN Regression model တွင် k အရေအတွက်သည် တိကျမှု(accuracy)ကို ဆုံးဖြတ်ပေးနိုင်သည့် အလွန်အရေးကြီးသည့် အချက်ဖြစ်သည်။ ယခု ဥပမာတြင် k အရေအတွက်ကို ပြောင်းလဲပေးရုံဖြင့် တိကျမှု(accuracy) ပြောင်းလဲသွားပုံကို သရုပ်ပြပြီးဖြစ်သည်။ k အရေအတွက် (၄) ဖြင့် ရရှိသည့် တိကျမှု(accuracy) သည် Linear Regression model ၏ တိကျမှု(accuracy) ထက် ပိုကောင်းပုံကို သရုပ်ပြထားသည်။ ထို့ကြောင့် k-NN တွင် တိကျမှု(accuracy)ကောင်းရန်အတွက် k အရေတွက်သည် အလွန်အရေးကြီးသည့် အချက်ဖြစ်သည်။

Linear Regression model တွင် k အရေအတွက် ကဲ့သို့ တိကျမှု(accuracy)ကို ဆုံးဖြတ်ပေးနိုင်သည့် parameter မပါ ရှိပါ။

အောက်တွင် k အရေအတွက် နှင့် တိကျမှု(accuracy)ကို ဂရပ်ပုံဖြင့်ဖော်ပြထားသည်။

In [13]:
```
no_of_k = [1, 2, 3, 4]
knn_test_accuracy =  [0.653, 0.554, 0.695, 0.653]
```

In [15]:
```
import matplotlib.pyplot as plt
plt.plot(no_of_k  , knn_test_accuracy , 'C3', lw=3)
plt.scatter(no_of_k  , knn_test_accuracy , s=120)
plt.title('Numbe of k Vs Accuracy')
plt.xlabel('Numbe of k')
plt.ylabel('Accuracy')
plt.grid()
for i in range(0, 4):
    plt.text(no_of_k[i]+0.1, knn_test_accuracy[i], str(knn_test_accurac
y[i]))
```



In [ ]:
```

```