kd_tree

December 20, 2021

```
[29]: import seaborn as sns
      import numpy as np
      from matplotlib import pyplot as plt
      %matplotlib inline
```

OBSERVATIONS

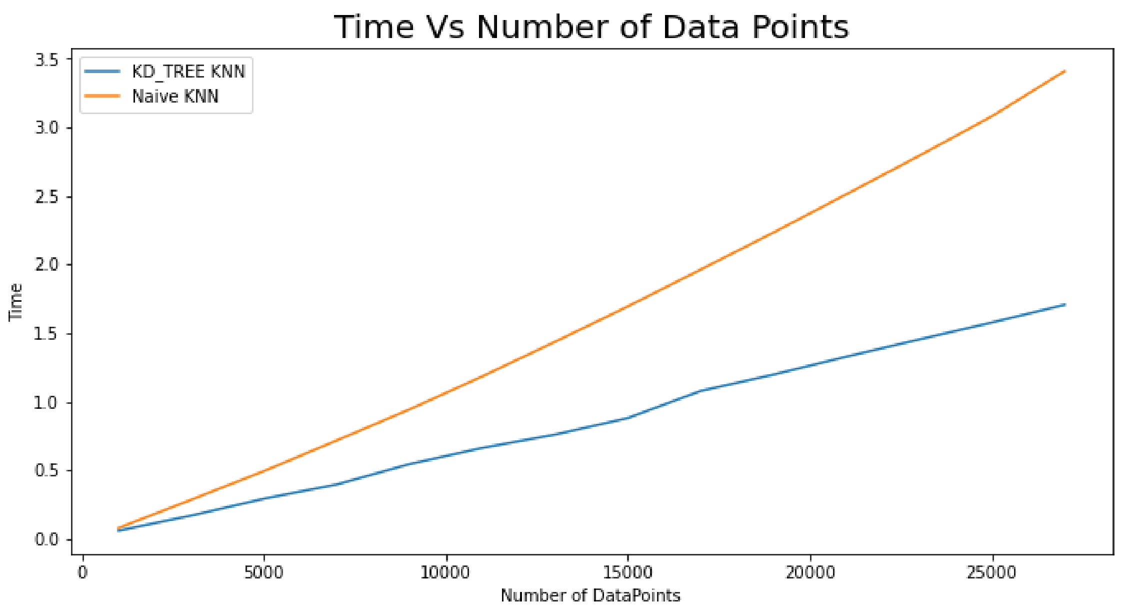
kd_KNN = [[0.059830427169799805, 1000], [0.16922879219055176, 3000], [0.29261302947998047, 5000], [0.3960397243499756, 7000], [0.5449154376983643, 9000], [0.6625206470489502, 11000], [0.7599623203277588, 13000], [0.8802840709686279, 15000], [1.0778067111968994, 17000], [1.1967675685882568, 19000], [1.3272411823272705, 21000], [1.452544927597046, 23000], [1.5764403343200684, 25000], [1.7053594589233398, 27000]]

naive_KNN = [[0.07917547225952148, 1000], [0.28507041931152344, 3000], [0.49408745765686035, 5000], [0.7174642086029053, 7000], [0.9435546398162842, 9000], [1.184443712234497, 11000], [1.436415195465088, 13000], [1.694065809249878, 15000], [1.9626212120056152, 17000], [2.2320971488952637, 19000], [2.5108184814453125, 21000], [2.791146755218506, 23000], [3.079211473464966, 25000], [3.4082326889038086, 27000]]

```
[4]: kd_x = [x[0] for x in kd_KNN]
     kd_y = [x[1] for x in kd_KNN]
     naive_x = [x[0] for x in naive_KNN]
     naive_y = [x[1] for x in naive_KNN]
```

```
[11]: fig, ax1 = plt.subplots(1, 1,)
      fig.tight_layout(w_pad = 4)
      fig.set_size_inches(10, 5)
      ax1.plot(kd_y,kd_x , label='KD_TREE KNN')
      ax1.plot(naive_y,naive_x, label='Naive KNN')
      ax1.set_title("Time Vs Number of Data Points", fontsize=20)
      ax1.set_ylabel("Time")
      ax1.set_xlabel("Number of DataPoints")
      ax1.legend()
```

```
[11]: <matplotlib.legend.Legend at 0x7f178007b340>
```



with the increase in the number of data-points to be searched , The time required by the Brute force KNN increases rapidly but KNN using KD tree is not effected much in comparison to brute force KNN

OBSERVATIONS

kd_K_alpha_100 = [[13.162370443344116, 5], [13.279274225234985, 20], [13.22063684463501, 50], [13.227225065231323, 100]] naive_K_100= [[27.640732049942017, 5], [27.486680269241333, 20], [27.53957772254944, 50], [27.526809453964233, 100]]

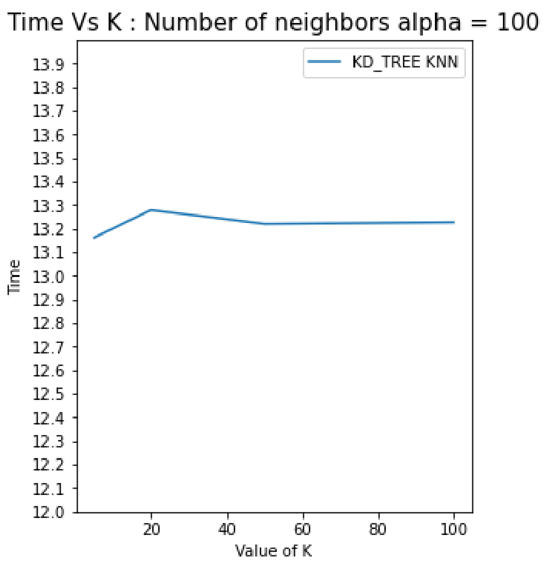
kd_K_alpha_30 = [[14.048462629318237, 5], [14.151240110397339, 20], [14.075403213500977, 50], [14.10598611831665, 100]] naive_K_30= [[27.738129377365112, 5], [27.612406015396118, 20], [27.572325706481934, 50], [27.58188557624817, 100]]

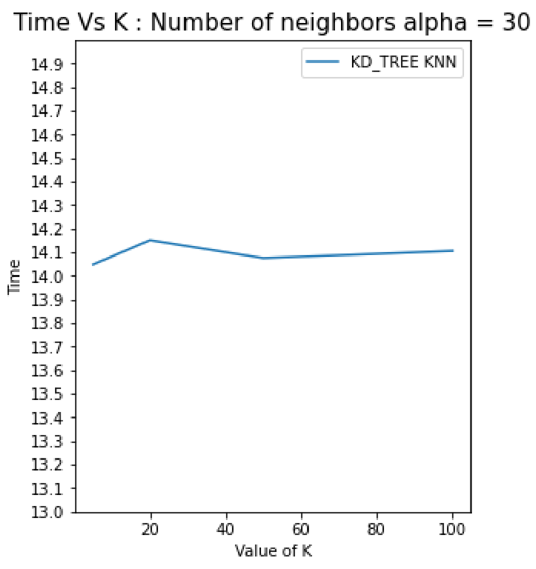
[17]:
```
y_alpha_100_k = [x[0] for x in kd_K_alpha_100]
x_alha_100_k = [x[1] for x in kd_K_alpha_100]
y_naive_100 = [x[0] for x in naive_K_100]
x_naive_100 = [x[1] for x in naive_K_100]

y_alpha_30_k = [x[0] for x in kd_K_alpha_30]
x_alha_30_k = [x[1] for x in kd_K_alpha_30]
y_naive_30 = [x[0] for x in naive_K_30]
x_naive_30 = [x[1] for x in naive_K_30]
```

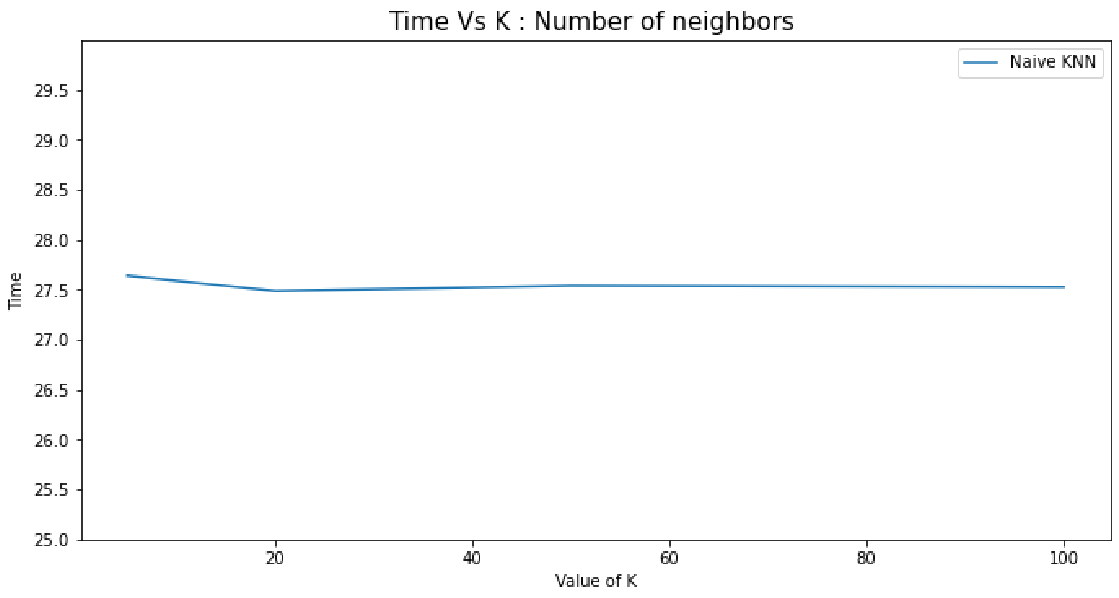
[44]:
```
fig, (ax1, ax2) = plt.subplots(1, 2,)
fig.tight_layout(w_pad = 4)
fig.set_size_inches(10, 5)
ax1.plot(x_alha_100_k ,y_alpha_100_k, label='KD_TREE KNN')
# ax1.plot(x_naive_100,y_naive_100, label='Naive KNN')
```

```
ax1.set_title("Time Vs K : Number of neighbors alpha = 100", fontsize=15)
ax1.set_ylabel("Time")
ax1.set_xlabel("Value of K")
ax1.yaxis.set_ticks(np.arange(12, 14, 0.1))
ax1.set_ylim([12,14])
ax1.legend()

ax2.plot(x_alha_30_k ,y_alpha_30_k, label='KD_TREE KNN')
# ax2.plot(x_naive_30,y_naive_30, label='Naive KNN')
ax2.set_title("Time Vs K : Number of neighbors alpha = 30", fontsize=15)
ax2.set_ylabel("Time")
ax2.set_xlabel("Value of K")
ax2.yaxis.set_ticks(np.arange(13, 15, 0.1))
ax2.set_ylim([13,15])
ax2.legend()


fig, ax1 = plt.subplots(1, 1,)
fig.tight_layout(w_pad = 4)
fig.set_size_inches(10, 5)
# ax1.plot(x_alha_100_k ,y_alpha_100_k, label='KD_TREE KNN')
ax1.plot(x_naive_100,y_naive_100, label='Naive KNN')
ax1.set_title("Time Vs K : Number of neighbors ", fontsize=15)
ax1.set_ylabel("Time")
ax1.set_xlabel("Value of K")
ax1.yaxis.set_ticks(np.arange(25, 30, 0.5))
ax1.set_ylim([25,30])
ax1.legend()
```

```
[44]: <matplotlib.legend.Legend at 0x7f177d4975e0>
```







Here We can see from the figure that

1.KD_Tree KNN is taking around 13.2 seconds irresective of the value of K when alpha is 100

2.When alpha is 30, Depth of tree is increased and takes time approximately 14.1 seconds for same Query irrespective of K

3.Brute force or Naive KNN is performing worst taking around 27.5 seconds irrespective of value of k.