k-NN Example 5 (k-NN by python function)

ယခု ဥပမာတွင် machine learn package များ မသုံးဘဲ python code များဖြင့်သာ kNN machine learn algorithm ဖြင့် ခန့်မှန်းချက်များ ပြုလုပ်ပါမည်။

ဥပမာတွင် ဒေတာ point (၆)ခုကို ထည့်သည်။ အနီ point (၃)ခု ကို အနီအစု(red class) ဟုသတ်မှတ်ပြီး အပြာ point (၃)ခု ကို အပြာအစု(blue class) ဟုသတ်မှတ်မည်။ ထို့နောက် စမ်းသပ်မည့် ဒေတာ point (test point)တစ်ခုကို ထည့်၍ မည်သည့် အစုတွင် ပါဝင်မည်ဟု ခန့်မှန်းမည်။

numpy နှင့် matplotlib.pyplot ကို import လုပ်သည်။

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

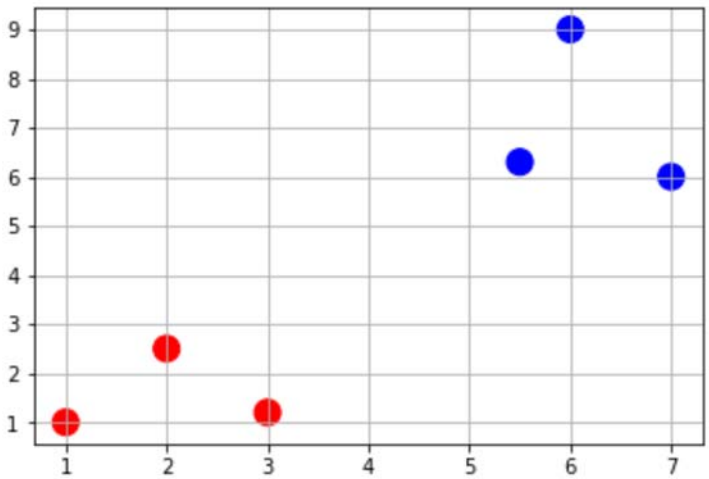
ပြင်ပမှ ဒေတာ များကို မသုံးဘဲ အလွယ်တကူ ဒေတာများရအောင် ပြင်ဆင်သည်။ `[1,1], [2,2.5], [3,1.2], [5.5, 6.3], [6,9],[7,6]` စသည့် ဒေတာ point (၆)ခုကို ထည့်သွင်းသည်။ ပထမဆုံး ဒေတာ (၃)ကို အနီရောင်များဖြင့် ဖော်ပြမည်။ ကျန်ဒေတာ (၃)ခုကို အပြာရောင်ဖြင့် ဖော်ပြမည်။

Point များ၏ coordinate တန်ဖိုးများ(x,y)ကို feature သို့မဟုတ် X_train ဒေတာများအဖြစ် သတ်မှတ်မည်။ X_train သည် two dimension (2D) array တစ်ခု ဖြစ်သည်။ အရောင်များကို label သို့မဟုတ် Y_train ဒေတာများအဖြစ် သတ်မှတ်မည်။

In [2]:
```
X_train = np.array([[1,1], [2,2.5], [3,1.2], [5.5, 6.3], [6,9],[7,6]])
Y_train = ['red', 'red', 'red', 'blue', 'blue', 'blue']
```

ထို point (၆)ခုကို ဂရပ်ပေါ်တွင် ဆွဲ၍ data visualization လုပ်ကြည့်မည်။

In [3]:
```
plt.scatter(X_train[:,0], X_train[:,1], s=170, color = Y_train[:])
plt.grid()
```

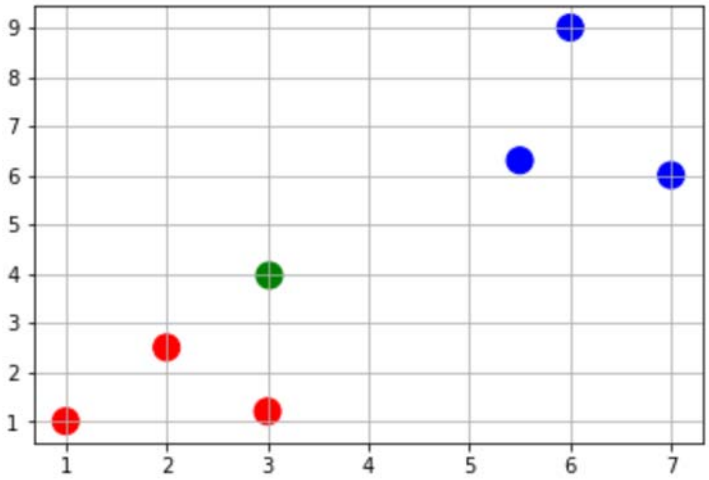


စမ်းသပ်မည့် ဒေတာ point (test point)တစ်ခုကို ထည့်သည်။

In [4]:
```
X_test = np.array([3,4])
```

train ဒေတာ point (၆)ခုနှင့် test ဒတာ point (၁)ခု ပေါင်း(၇)ကို ဂရပ်ပေါ်တင်ဆွဲသည်။

In [5]:
```
plt.scatter(X_train[:,0], X_train[:,1], s=170, color = Y_train[:])
plt.scatter(X_test[0], X_test[1], s = 170, color='green')
plt.grid()
```



Nearest Neighbour Classifier အတွက် distance function တစ်ခုကို define လုပ်သည်။

In [6]:
```
def dist(x,y):
    return np.sqrt(np.sum((x-y)**2))
```

Define လုပ်ပြီးသည့် distance function ၏ အလုပ်လုပ်ပုံ မှန် မမှန် စစ်ဆေးမည်။

In [7]:
```
a = np.array([1,1])
b = np.array([2,2])
c= dist(a,b)
c
```

Out[7]:
```
1.4142135623730951
```

point a (1,1) နှင့် point b (2,2) နှစ်ခုအကြား၏ အကွာအဝေးမှာ 1.414 ဖြစ်သည်။(Root 2 = 1.414)။ Define လုပ်ပြီးသည့် distance function ၏ အလုပ်လုပ်ပုံ မှန်သည်။

အကွာအဝေး(distance)(၆)ခု ကို တွက်မည်

အခုဆက်လက်ပြီး X_train မှ point (၆)ခု နှင့် X_test point (၁)ခု အကြားရှိ အကွာအဝေး(distance)(၆)ခု ကို တွက်မည်။ num နှင့် distance တို့ကို assign လုပ်သည်။

In [8]:
```
num = len(X_train) #Number of points in X_train
distance =  np.zeros(num) # Numpy arrays of zeros
```

for loop ဖြင့် distance function ကို သုံး၍ အကွာအဝေး(distance)(၆)ခု ကို တွက်မည်။

In [21]:
```
for i in range(num):
    distance[i] = dist(X_train[i], X_test)
    print(distance[i])
```

```
3.605551275463989
1.8027756377319946
2.8
3.3970575502926055
5.830951894845301
4.47213595499958
```

အကယ်၍ k အရေအတွက် (၁)ခုတည်းဖြင့် ဆုံးဖြတ်လျှင် (k = 2) ဒုတိယ point ၏ အကွာအဝေး(distance)သည် 1.80 ဖြစ်သောကြောင့် X_test point နှင့် အနီးဆုံးဖြစ်သည်။ ထို့ကြောင့် X_test point သည် အနီအုပ်စုဝင်ဖြစ်ရမည်ဟု ခန့်မှန်းသည်။

In [23]:
```
min_index = np.argmin(distance) # Index with smallest distance
print(Y_train[min_index])
```

```
red
```

အကယ်၍ k အရေအတွက် (၂)ခုဖြင့် (k = 2) ဆုံးဖြတ်လျှင် ဒုတိယ point နှင့် တတိယ point ၏ အကွာအဝေး(distance)သည် 1.80 နှင့် 2.8 ဖြစ်သောကြောင့် X_test point နှင့် အနီးဆုံးဖြစ်သည်။ ထို့ကြောင့် X_test point သည် အနီအုပ်စုဝင်ဖြစ်ရမည်ဟု ခန့်မှန်းသည်။

k = 3 ဆိုလျှင်လည်း ထို့အတူပင်ဖြစ်သည်။

https://www.youtube.com/watch?v=dgt6IfEXgDk (https://www.youtube.com/watch?v=dgt6IfEXgDk)

In [ ]:
```
```