Advanced Tutorial on Implementing Apriori Algorithm

In this tutorial, we explain how the Apriori algorithm can be implemented by varying the minimum support values

Step 1: Import the Apriori algorithm and pandas data frame

In [1]:
```
from PAMI.frequentPattern.basic import Apriori  as alg
import pandas as pd
```

Step 2: Specify the following input parameters

In [2]:
```
inputFile = 'transactional_T10I4D100K.csv'
seperator='¥t'
minimumSupportCountList = [1000, 1500, 2000, 2500, 3000]
#minimumSupport can also specified between 0 to 1. E.g., minSupList = [0.005, 0.006,

result = pd.DataFrame(columns=['algorithm', 'minSup', 'patterns', 'runtime', 'memory
#initialize a data frame to store the results of Apriori algorithm
```

Step 3: Execute the Apriori algorithm using a for loop

In [3]:
```
algorithm = 'Apriori'  #specify the algorithm name
for minSupCount in minimumSupportCountList:
    obj = alg.Apriori('transactional_T10I4D100K.csv', minSup=minSupCount, sep=seperа
    obj.startMine()
    #store the results in the data frame
    result.loc[result.shape[0]] = [algorithm, minSupCount, len(obj.getPatterns()), c
```

```
Frequent patterns were generated successfully using Apriori algorithm
Frequent patterns were generated successfully using Apriori algorithm
Frequent patterns were generated successfully using Apriori algorithm
Frequent patterns were generated successfully using Apriori algorithm
Frequent patterns were generated successfully using Apriori algorithm
```

In [4]:
```
print(result)
```

```
  algorithm  minSup  patterns     runtime     memory
0   Apriori    1000       385  484.487166  266063872
1   Apriori    1500       237  148.440647  265666560
2   Apriori    2000       155   63.079747  266498048
3   Apriori    2500       107   32.636862  265863168
4   Apriori    3000        60   14.269831  265187328
```

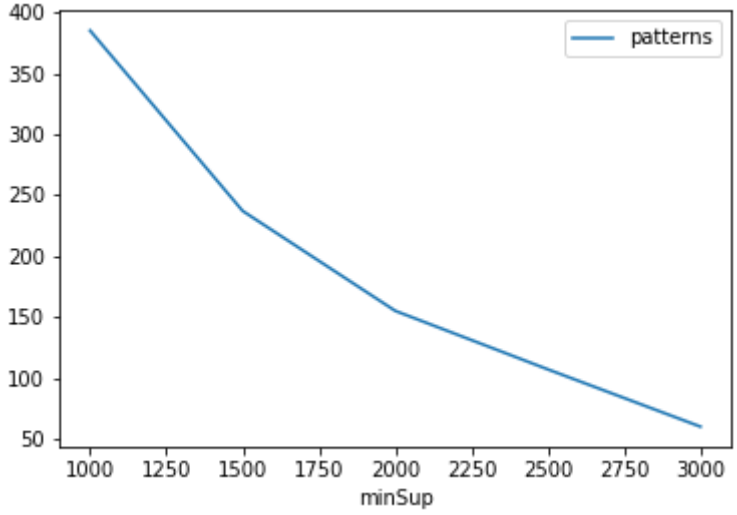
Step 5: Visualizing the results

Step 5.1 Importing the plot library

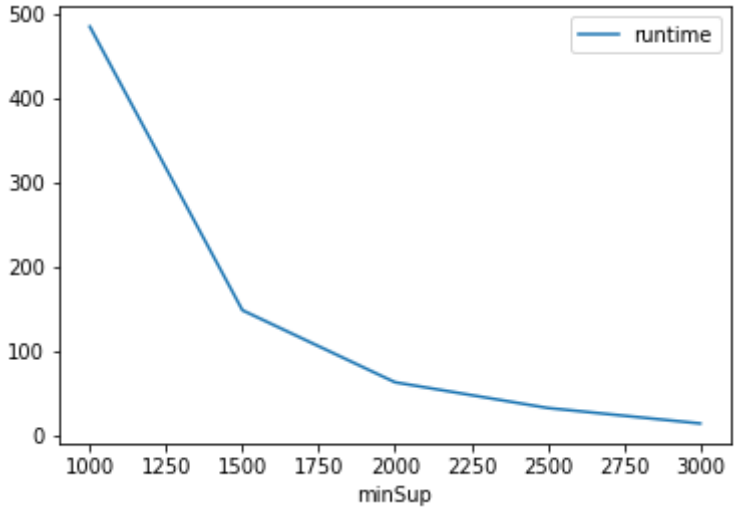
In [5]:
```
from PAMI.extras.graph import plotLineGraphsFromDataFrame as plt
```

Step 5.2. Plotting the number of patterns

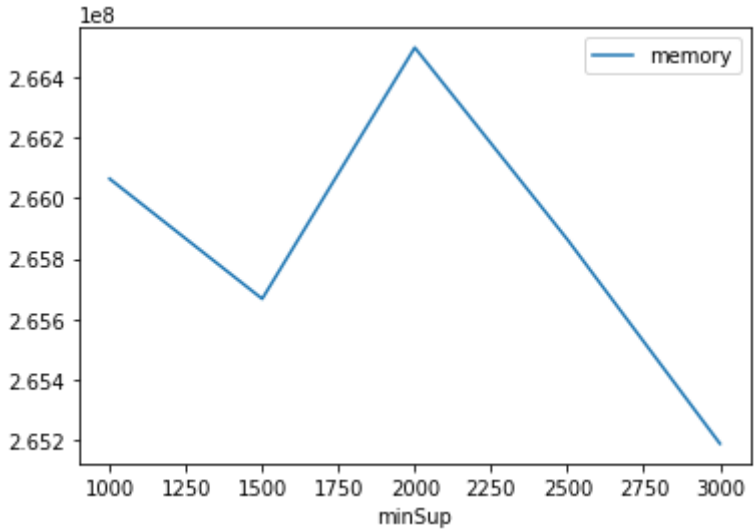
In [6]:
```
ab = plt.plotGraphsFromDataFrame(result)
ab.plotGraphsFromDataFrame() #drawPlots()
```



Graph for No Of Patterns is successfully generated!



Graph for Runtime taken is successfully generated!



Graph for memory consumption is successfully generated!

Step 6: Saving the results as latex files

In [7]:
```
from PAMI.extras.graph import generateLatexFileFromDataFrame as gdf
gdf.generateLatexCode(result)
```

Latex files generated successfully

In [ ]: