Advanced Tutorial on Implementing Frequent-Pattern Growth Algorithm

---

In this tutorial, we explain how the Frequent Pattern-growth (FP-growth) algorithm can be implemented by varying the minimum support values

Step 1: Import the FP-growth algorithm and pandas data frame

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

Step 2: Specify the following input parameters

In [2]:
```
inputFile = 'transactional_T10I4D100K.csv'
seperator='¥t'
minimumSupportCountList = [100, 150, 200, 250, 300]
#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 FP-growth algorithm
```

Step 3: Execute the FP-growth algorithm using a for loop

In [3]:
```
algorithm = 'FPGrowth'  #specify the algorithm name
for minSupCount in minimumSupportCountList:
    obj = alg.FPGrowth('transactional_T10I4D100K.csv', minSup=minSupCount, sep=seper
    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 frequentPatternGrowth algorithm
Frequent patterns were generated successfully using frequentPatternGrowth algorithm
Frequent patterns were generated successfully using frequentPatternGrowth algorithm
Frequent patterns were generated successfully using frequentPatternGrowth algorithm
Frequent patterns were generated successfully using frequentPatternGrowth algorithm
```

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

```
  Algorithm  minSup  patterns    runtime     memory
0  FPGrowth     100     27532  11.474215  517918720
1  FPGrowth     150     19126   9.795341  510083072
2  FPGrowth     200     13255  10.240599  503455744
3  FPGrowth     250      7703   9.773888  500404224
4  FPGrowth     300      4552  11.242900  497012736
```

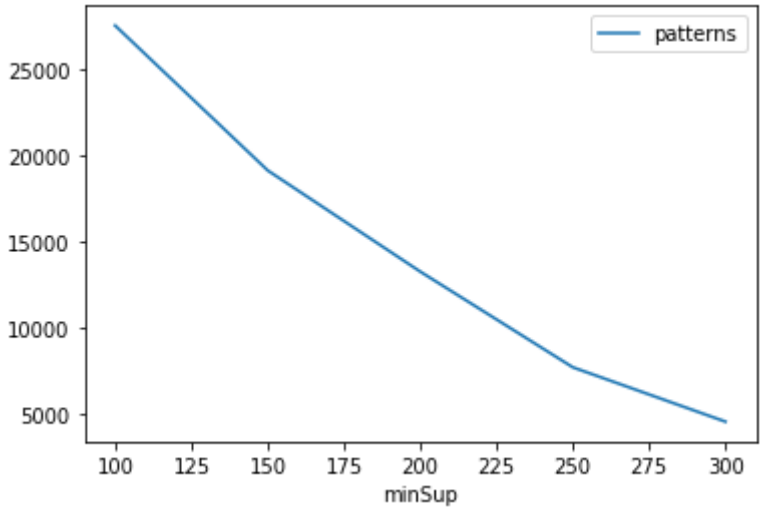
Step 5: Visualizing the results

Step 5.1 Importing the plot library

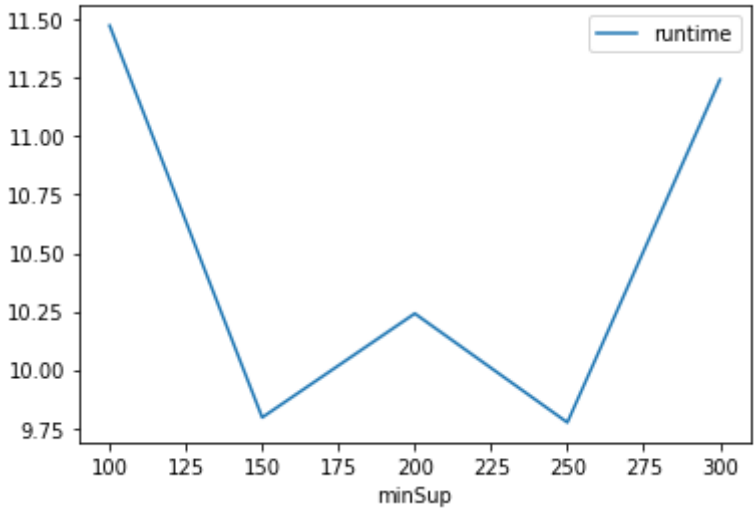
In [5]:
```
from PAMI.extras.graph import plotLineGraphsFromDataFrame as plt
```

Step 5.2. Plotting the number of patterns

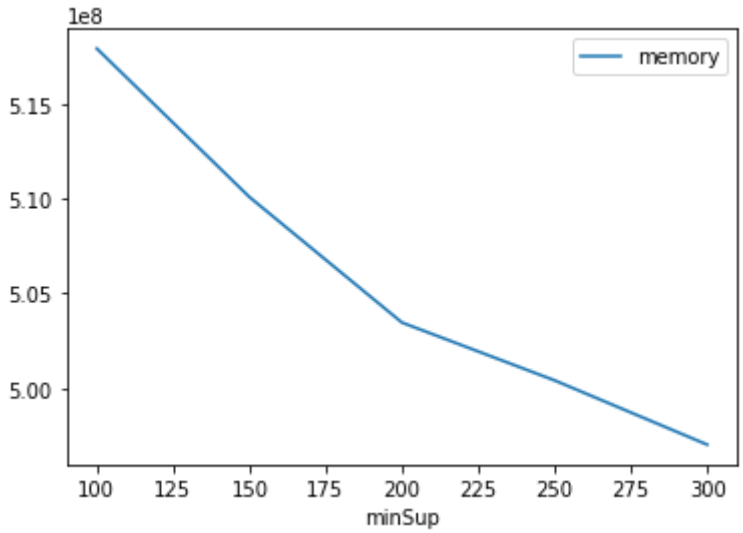
In [6]:
```
ab = plt.plotGraphsFromDataFrame(result)
ab.plotGraphsFromDataFrame() #drawPlots()
```



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)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [7], in <cell line: 2>()
      1 from PAMI.extras.graph import generateLatexFileFromDataFrame as gdf
----> 2 gdf.generateLatexCode(result)

File ~/.local/lib/python3.10/site-packages/PAMI/extras/graph/generateLatexFileFromDa
taFrame.py:8, in generateLatexCode(result)
      6 titles = result.columns.tolist()
      7 titles.remove("minSup")
----> 8 titles.remove("algorithm")
      9 for i in range(0, len(titles)):
     10     legendary = pd.unique(result[['algorithm']].values.ravel())

ValueError: list.remove(x): x not in list
```

In [ ]: