**Handling large datasets on Python: Pandas, Dask or PySpark**

**Memory management in Python**

**By Dinesh Pandey**

**I have completed this project with three different possible methods using,**

1. **Pandas**

Relative\_Sales\_Pandas.ipynb

1. **Dask**

Relative\_Sales\_Dask.ipynb

1. **PySpark**

Relative\_Sales\_PySpark.ipynb

**All the methods outputs same results. The project was done using Jupyter Notebook. While executing the project, please change the path to the directory where the data is located.**

**Summary of the Project**

Choosing the right methodology really depends upon the size of datasets. If the dataset is less than 1 GB, I would go with pandas and will have no concern with the performance. By default, Pandas executes its functions as a single process using a single CPU core.

However, if the dataset is big, I will go for a multi-threading approach.

1. I can still use pandas using the chunk size to load the file into pandas dataframe.
2. I can import the data into Dask dataframe. Dask can enable efficient parallel computations on single machines by leveraging their multi-core CPUs and streaming data efficiently from disk.

Pandas exports the dataframe as a single CSV. Hence, Dask takes more time compared to Pandas. I would say for the datasets between 1 to 100 GB, overall performance of the Dask to handle is quite impressive. Also, Dask has a better Dashboard to monitor the memory usage.

1. If the dataset is really big, I will recommend using PySpark dataframe. PySpark can handle petabytes of data efficiently because of its distribution mechanism. The SQL like operations are intuitive to data scientists which can be run after creating a temporary view on top of Spark DataFrame. Spark SQL also allows users to tune the performance of workloads by either caching data in memory or configuring some experimental options.

For the small to medium datasets, PySpark run slower than other libraries. This is because, by default when we run spark in SQL Context or Hive Context it will use 200 partitions. We can change the setting of partitions based on our requirement

There are several approaches to handle different datasets based on their size and processing required. For this project, I would go with DASK for the parallel computation. In the future, if the dataset grows and becomes difficult to handle, I will switch from Dask to PySpark.