# **Tamil Nadu Marginal Workers**

# **Development 1.0**

**Phase :03 Project Number : 03**

**Introduction:**

In today's data-driven world, the ability to extract valuable insights from large and complex datasets is paramount. This data analysis project endeavors to explore and derive actionable conclusions from [briefly describe the dataset or data source, including its origin and nature]. Through the application of various data analysis techniques and methodologies, we aim to uncover patterns, trends, and hidden knowledge within this dataset.

**Project Objectives:**

* Python for Data Science
* Using the In-Built libraries (pandas, matplotlib)
* Using TN-Marginal Dataset to visualize

**Data Set:**

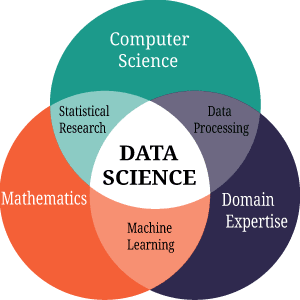
Here we are using the data set from the govt organization official site(tn.data.gov.in).

This dataset from: <https://tn.data.gov.in/resource/marginal-workers-classified-age-industrial-category-and-sex-scheduled-caste-2011-tamil>

This dataset contains 71 columns and 595 rows.

**Python for Data Science:**

Python is an incredibly popular and versatile programming language for data science. It offers a wide range of libraries and tools that make it a powerful choice for tasks like data analysis, data visualization, machine learning, and more. Here's a brief overview of some of the key libraries and tools used in Python for data science.



**Using the Built-In Libraries (Pandas):**

Pandas is a powerful Python library for data manipulation and analysis. It provides data structures and functions for working with structured data, making it an essential tool for data scientists and analysts. Here are some of the key features and concepts related to Pandas:

* Data Structures:

DataFrame: A two-dimensional, tabular data structure with labeled axes (rows and columns). It's similar to a spreadsheet or SQL table and is the primary data structure used in Pandas.

Series: A one-dimensional labeled array capable of holding data of various types. Series can be thought of as a column in a DataFrame.

* Data Import and Export:

Pandas can read and write data from/to various file formats, including CSV, Excel, SQL databases, and more.

Common methods for reading data include pd.read\_csv(), pd.read\_excel(), and pd.read\_sql().

Data can be exported using methods like to\_csv(), to\_excel(), and to\_sql().

* Data Cleaning and Preprocessing:

Pandas provides functions for handling missing data, duplicate data, and data type conversion.

Methods like drop(), fillna(), replace(), and astype() are used for data cleaning.

* Indexing and Selection:

You can use labels or integer-based indexing to select and filter data in Pandas DataFrames.

Common selection methods include loc[] for label-based indexing and iloc[] for integer-based indexing.

* Data Manipulation:

Pandas supports a wide range of operations, such as filtering, grouping, aggregating, sorting, and merging datasets.

Functions like groupby(), agg(), sort\_values(), and merge() are commonly used for these operations.

* Data Visualization:

While Pandas itself is not primarily a visualization library, it can be easily integrated with libraries like Matplotlib and Seaborn for data visualization.

Pandas DataFrames can be used to create basic plots and charts.

* Time Series Data:

Pandas has robust support for time series data, with specialized data structures and functions for handling date and time-related operations.

The Timestamp and DatetimeIndex are used for working with dates and times.

* Reshaping Data:

Pandas provides methods for reshaping data, such as pivot, melt, and stack/unstack, which are useful for transforming data from wide to long format and vice versa.

* Handling Categorical Data:

Categorical data can be handled efficiently using Pandas' Categorical data type, which can reduce memory usage and improve performance.

* Statistical Analysis:

Pandas offers a wide range of statistical functions for basic data analysis and summary statistics.

Functions like mean(), median(), sum(), and describe() can provide insights into the data.

* Combining Data:

You can merge, join, and concatenate DataFrames to combine data from multiple sources using methods like concat(), merge(), and join().

**Analytics of Data:**

Here we use the TN-Marginal workers dataset, further analyzing the data we show some examples of visualization following.

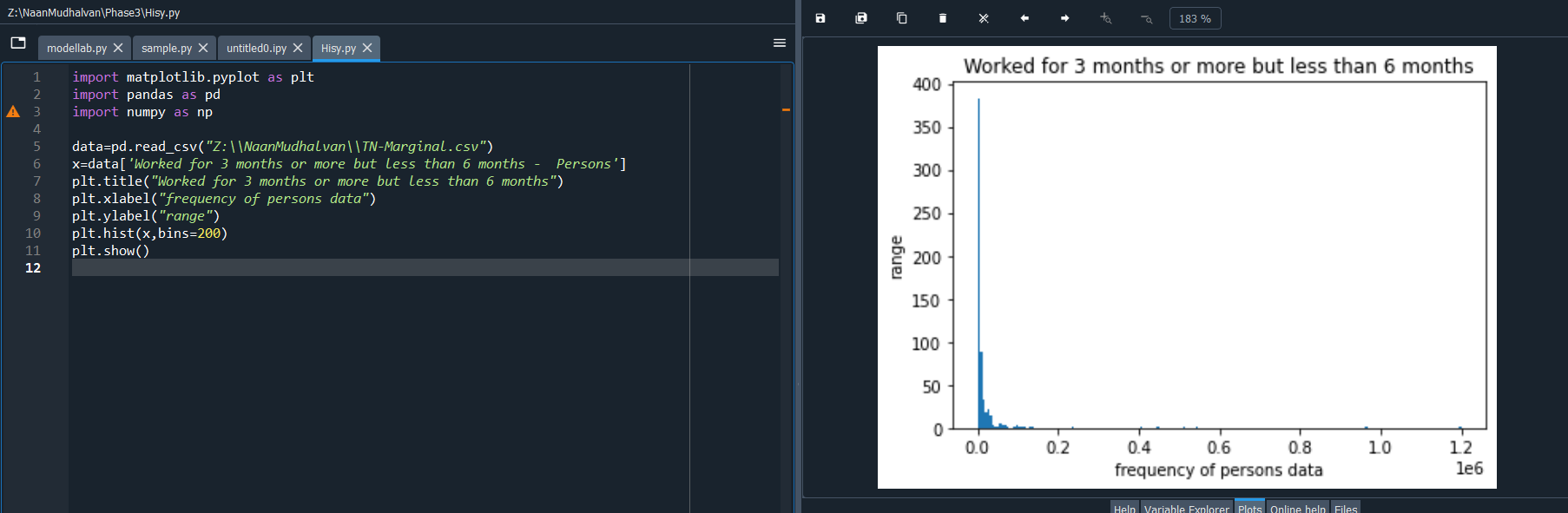
Worked Range Analysis:

That category satisfying more visualization tool, but here we choose the histogram plot,

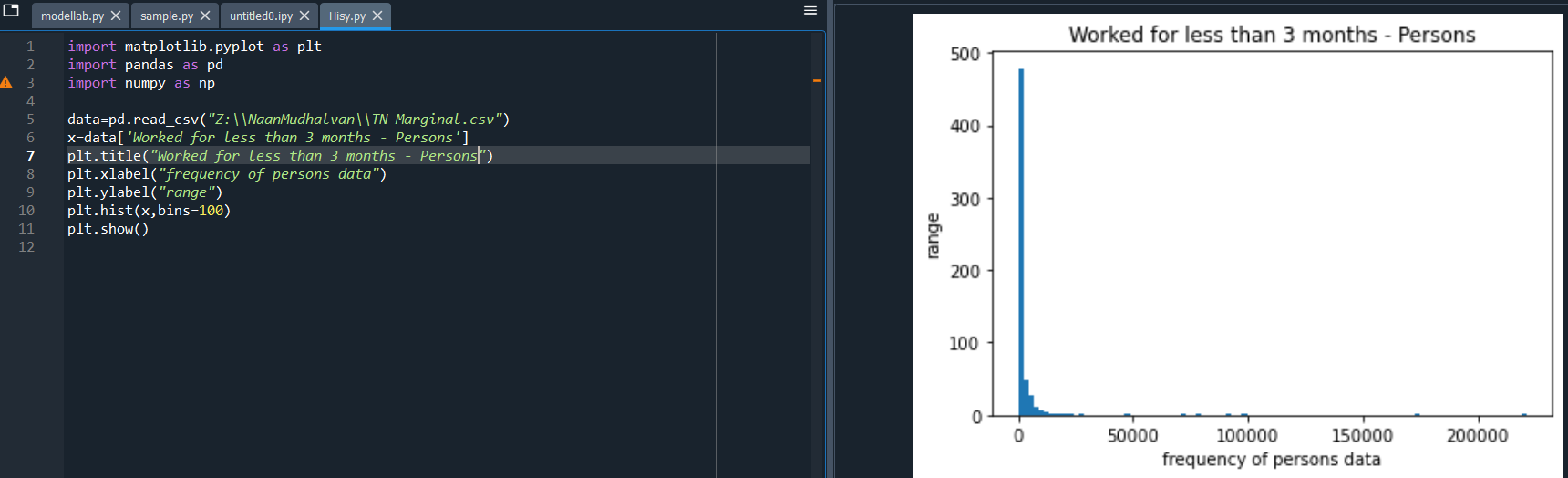
The hist is a function of matplotlib library, it require two mandatory data

1. X -data
2. Bins-range/interval

* Worked for 3 months or more but less than 6 months



* Worked for less than 3 months – Persons

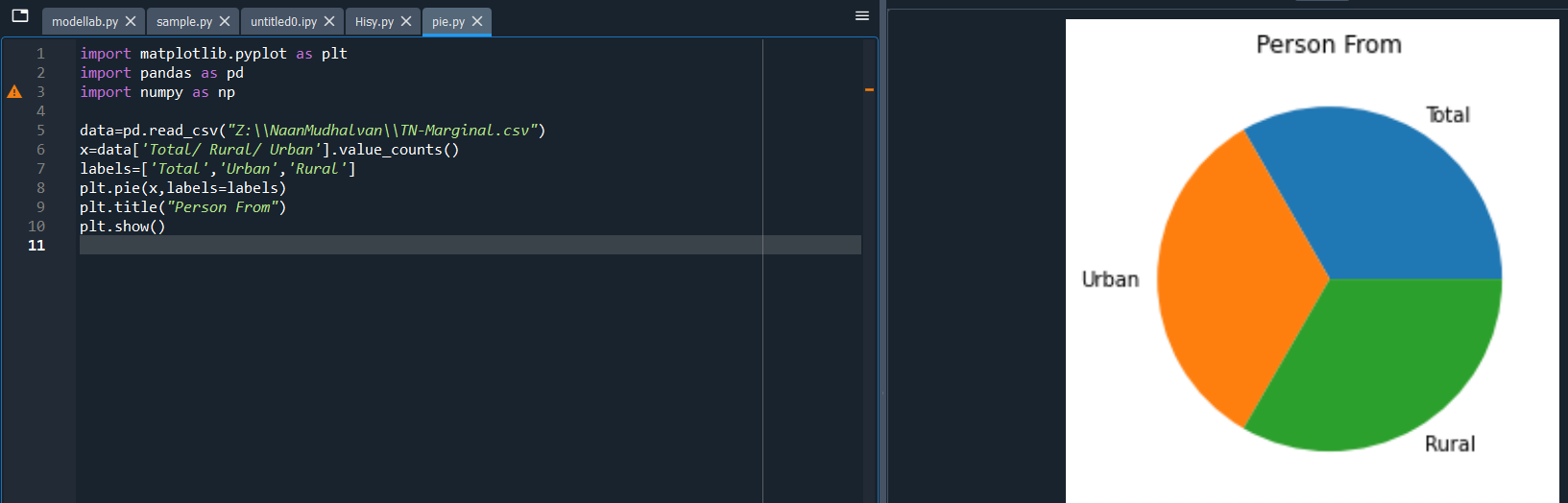


Person From:

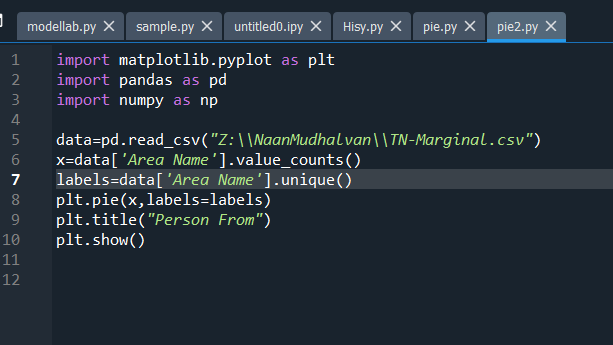
That category satisfying more visualization tool, but here we choose the Pie Chart,

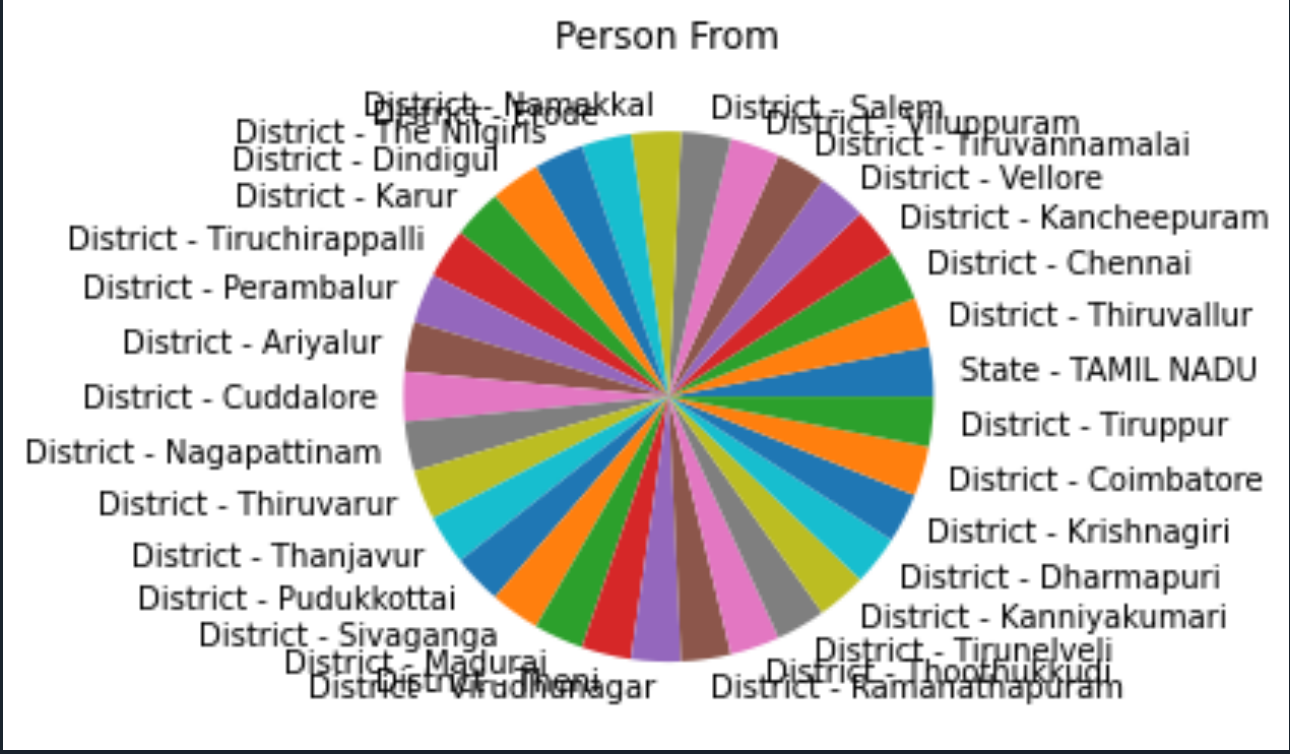
The pie chart requires one mandatory parameter data.

Labels and explode are not necessary

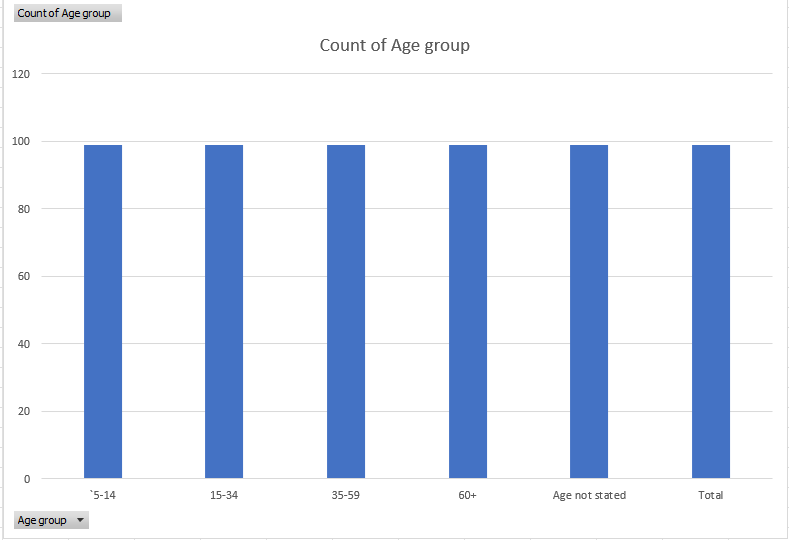
****

Person from (district vise):

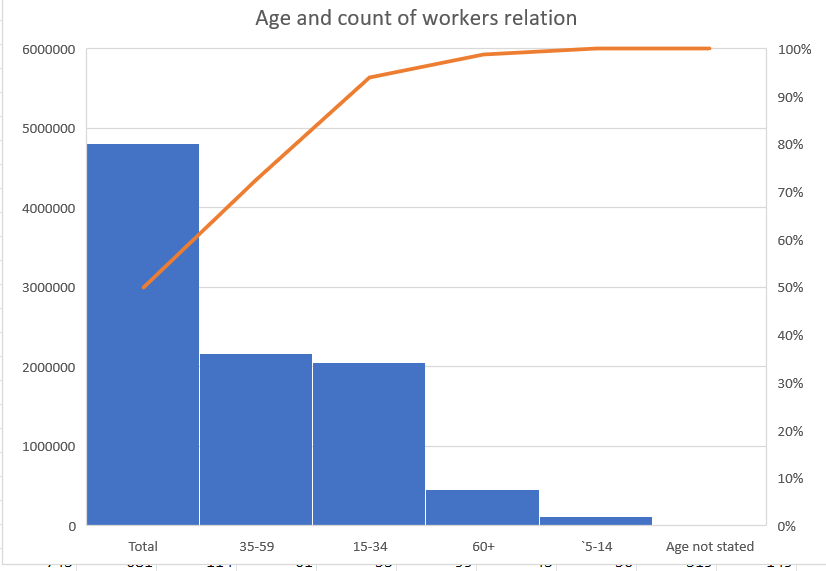


****

* frequency of age-group

****

* Age and frequent of person worked within 6 months

****

**Conclusion:**

The visualizations created for this assignment have played a crucial role in making these insights accessible and understandable. Visual aids such as bar charts, scatter plots, and heatmaps have not only made the data more approachable but have also allowed us to identify trends and patterns that might have otherwise gone unnoticed.

It is important to note that the choice of visualization techniques was instrumental in our ability to answer the assignment's questions effectively.