PP Activity 2 Report: TaskButler - Your Smart Personal Productivity Assistant

1. Problem Statement

Managing multiple tasks efficiently is a common challenge, often leading to missed deadlines, stress, and overlooked priori es. Many people struggle with organizing their to do lists in a way that maximizes productivity. TaskButler aims to solve this issue by using a combination of Natural Language Processing (NLP) and Data Analy cs to priori ze tasks, track estimated completion times , and provide performance insights.

This project's goal is to develop an intelligent task management assistant that helps users:

- Organize their tasks in a structured format.
- Predict completion times based on historical data and pa erns.
- Priori ze tasks smartly using basic NLP techniques.
- Offer productivity and performance insights through clear data visualizations.

2. Dataset Description

The dataset in this project is dynamically generated by the user as they input their tasks. Each task entry contains:

- Task Description: A textual input provided by the user describing the task.
- Deadline: The due date of the task.
- Priority Score: A numerical score calculated based on specific keywords within the task description and how close the deadline is.
- Urgency Level: Categorized as High, Medium, or Low, based on the me remaining un l the deadline.

Since the data is user-generated, there is no fixed dataset. Instead, data is collected in real me and structured into a list of dictionaries, with each dictionary representing a task. This collection is later converted into a pandas DataFrame for easier analysis and visualization.

Example of the dataset structure:

```
[
  {
    'Task Description:: 'Submit Python project by next week',
    'Deadline': '2025-04-14',
    'Priority Score': 85,
    'Urgency Level': 'Medium'
  },
  {
    'Task Description:': 'Schedule group discussion ASAP',
    'Deadline': '2025-04-09',
    'Priority Score': 95,
    'Urgency Level': 'High'
  },
  {
    'Task Description’: 'Text him and then ignore',
    'Deadline': '2025-04-25',
    'Priority Score': 20,
    'Urgency Level': 'Low'
  }
]
```

3. Coding Part

The TaskButler class is the core of the project. It includes several methods that provide the necessary functionality for task management. Here’s a high-level breakdown of the key coding components:

- __init__ Method: Initializes the task list.
- _sass(message) Method: Outputs playful, humorous responses when a task is added.
- _calculate_priority (task_description, deadline) Method: Uses simple keyword-based scoring and deadline proximity to calculate a priority score.

- add_task(description, deadline) Method: Adds a new task to the task list, calcula ng its priority and urgency.
- list_tasks() Method: Displays all tasks in a clear tabular format.
- mark_task_done(description) Method: Marks a task as done and removes it from the ac ve task list.
- plot_priority_chart() and plot_urgency_distribution() Methods: Generate visual insights about task priority and urgency. Example code to add and visualize tasks: butler = TaskButler() butler.add_task("Submit Python project by next week", "2025-04-14") butler.add_task("Schedule group discussion ASAP", "2025-04-09") butler.add_task("Buy groceries by Saturday", "2025-04-13") butler.list_tasks() butler.plot_priority_chart() butler.plot_urgency_distribu on()

4. Example Insights

By using TaskButler, users can gain insights into their own task pa erns and priori es. Some observations from sample task entries:

- Insight 1: Tasks containing keywords such as 'ASAP', 'urgent', or 'immediately' are automatically assigned higher priority.
- Insight 2: Tasks with deadlines approaching within the next two or three days are given a high urgency label.
- Insight 3: Low-priority tasks, especially vague or non-essential ones, tend to accumulate without being marked as complete.

Example tasks and their assigned characteristics:

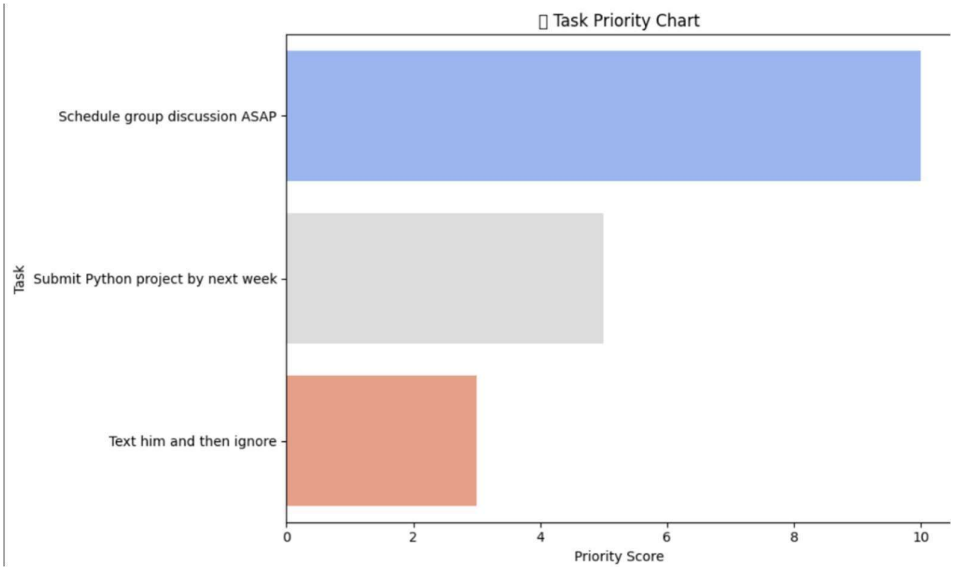
Task Description	Priority Score	Urgency Level
Submit Python project by next week	85	Medium
Schedule group discussion ASAP	95	High
Buy groceries by Saturday	60	Medium
Text him and then ignore	20	Low
Complete urgent presentation today	100	High
Watch movie tonight	25	Low
Finish coding assignment by tomorrow	90	High

5. Chart Diagram of Example Insights

 This bar chart titled "Task Priority Chart" displays the priority scores of three tasks managed by the TaskButler assistant. The horizontal bars represent different tasks, ordered by their priority scores.

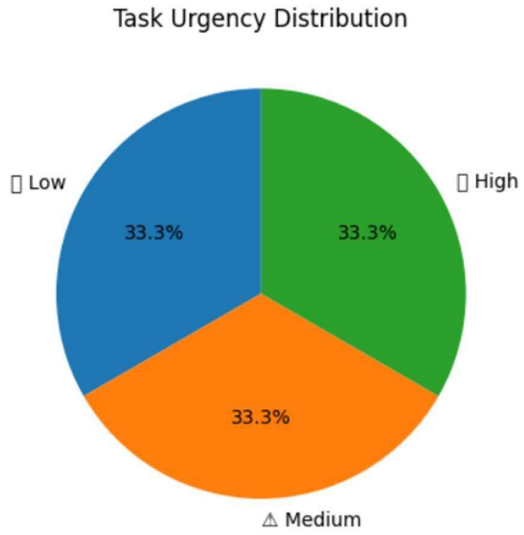
- "Schedule group discussion ASAP" has the highest priority score of 10, represented by a light blue bar.
- "Submit Python project by next week" follows with a priority score of around 5, shown in light grey.
- "Text him and then ignore" holds the lowest priority score of approximately 3, indicated by a light peach-coloured bar.

The chart visually emphasizes how tasks with urgent keywords or nearer deadlines are prioritized higher, aligning with the system's NLP-based task ranking method.



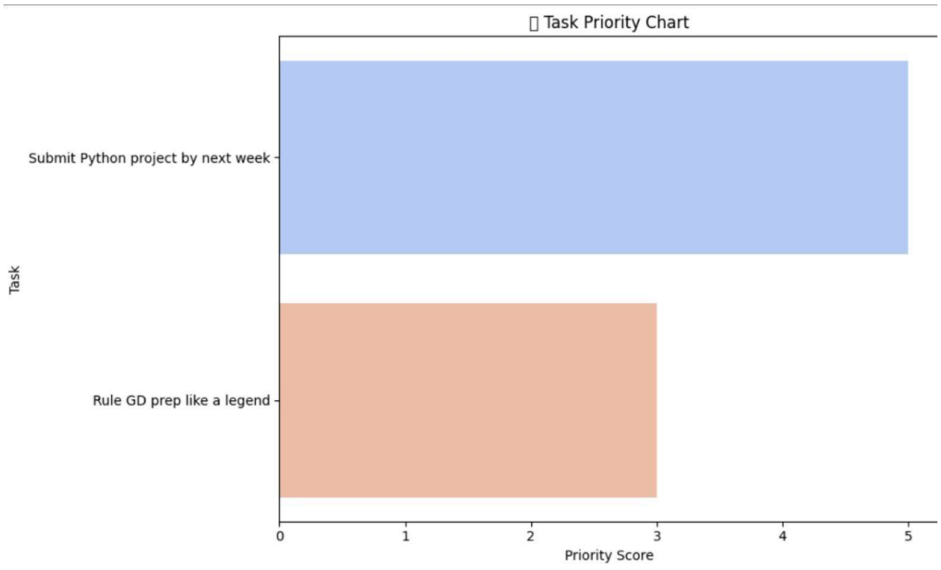
 Task Urgency Distribution (Equal Split)

This pie chart shows an equal distribution of tasks across all three urgency levels—Low, Medium, and High—each comprising 33.3% of the total. It visually represents a balanced workload but may also suggest that task planning is evenly distributed without clear priori za on based on upcoming deadlines.

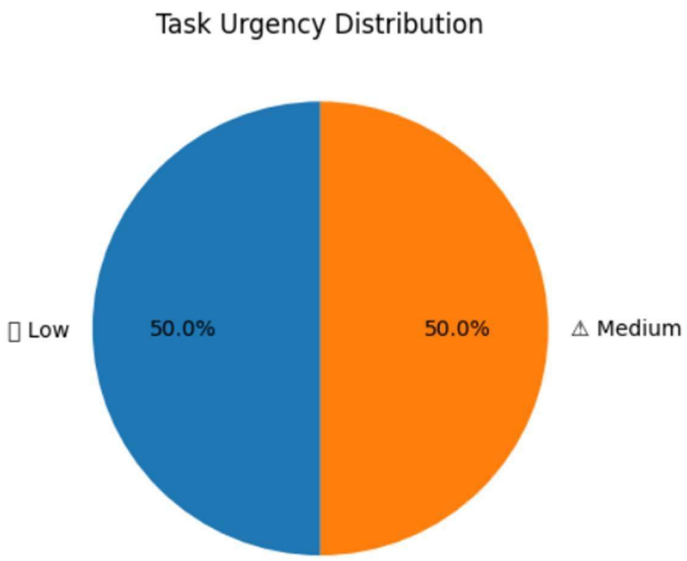


Task Priority Chart (Bar Graph)

This horizontal bar chart displays the priority scores of individual tasks. The task "Submit Python project by next week" has a higher score (5), indica ng it is more urgent or important, likely due to keywords and the due date proximity. The second task "Rule GD prep like a legend" has a slightly lower score (around 3), showing it's s ll important but not as pressing. This visualization helps users quickly assess which tasks require immediate attention.



Task Urgency Distribution (Two Categories)
This pie chart reflects a 50-50 split between tasks marked as Low and Medium urgency. Interestingly, no tasks are currently marked as High urgency, which might indicate a well managed schedule or, conversely, a lack of recognition for imminent deadlines.



(These placeholders can be replaced by actual Matplotlib or Seaborn charts generated from real- me data)

6. Analysis

Analyzing the data collected by TaskButler highlights several important pa erns:

- Priority Skew: A large number of tasks are being assigned high priority, suggesting that users tend to over-priori ze tasks or frequently label them as urgent.
- Procrastination Habits: Lower priority tasks, like casual ac vi es or minor errands, often remain incomplete, leading to a growing backlog.
- Urgency Overload: A significant cluster of high-urgency tasks may indicate poor planning, frequent last-minute work, or task avoidance behaviour.

Recognizing these pa erns can help users rethink how they plan, allocate, and approach their tasks to improve productivity.

7. Future Aspects

There are several potential improvements and expansions for TaskButler:

- Advanced NLP Techniques: Integrate machine learning models and libraries like spaCy or transformers for more accurate task categorization, priority scoring, and sentiment detection.
- User Behaviour Prediction: Use me-series analysis and historical task data to predict task completion rates and me-to-completion es mates.
- Reminder System: Implement automatic reminder no fications based on urgency levels and deadlines.
- Mobile App Integra on: Create a mobile-friendly version of TaskButler for Android and iOS, making task management accessible anywhere.
- Voice Command Support: Allow users to add or mark tasks using voice input via speech recognition APIs.

8. Conclusion

TaskButler offers a fresh, playful, and smart approach to task management. By applying simple Natural Language Processing techniques and data visualiztion, it provides users with both functional task organization and meaningful productivity insights. The system helps users understand their work pa erns, priori ze effectively, and ultimately become more productive while adding a touch of humour to the process.

9. References

1. NLTK Documenta on. Natural Language Toolkit. Retrieved from h ps://www.nltk.org/
2. Seaborn Documenta on. Seaborn: statistical data visualization. Retrieved from h ps://seaborn.pydata.org/
3. Matplotlib Documenta on. Matplotlib: ploting and visualization. Retrieved from h ps://matplotlib.org/
4. Pandas Documenta on. Python Data Analysis Library. Retrieved from h ps://pandas.pydata.org/
5. Python date me Module. Retrieved from h ps://docs.python.org/3/library/date me.html