Finance & Banking-ATM cash demand forecasting

A PROJECT REPORT

Submitted by

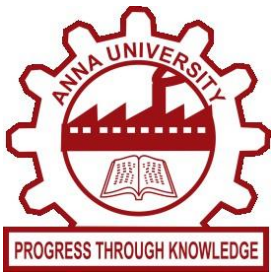
RAMESH M	2116231801134
SARATH KUMAR P	2116231801156
SRICHARAN N	2116231801170

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE



RAJALAKSHMI ENGINEERING COLLEGE
(AUTONOMOUS), CHENNAI – 602 105

OCTOBER 2025

BONAFIDE CERTIFICATE

Certified that this Report titled "Finance & Banking-ATM cash demand forecasting" is the Bonafide work of "RAMESH M **(**2116231801134**)** SARATH KUMAR P **(**2116231801156**)** SRICHARAN N **(**2116231801170**)"** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

Dr. Suresh Kumar S M.E., Ph.D.,

Professor,

Department of Artificial Intelligence & Data Science,

Rajalakshmi Engineering College

Thandalam – 602 105.

Submitted to Project Viva-Voce Examination held on _____

Internal Examiner External Examiner

ACKNOWLEDGEMENT

Initially I thank the Almighty for being with us through every walk of my life and showering his blessings through the endeavor to put forth this report.

My sincere thanks to our Chairman **Mr. S. MEGANATHAN, M.E., F.I.E.,** and our Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN, M.E., Ph.D.,** for providing me with the requisite infrastructure and sincere endeavoring educating me in their premier institution.

My sincere thanks to **Dr. S.N. MURUGESAN M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time.

I express my sincere thanks to **Dr. J M Gnanasekar M.E., Ph.D.,** Head of the Department of Artificial Intelligence and Data Science for his guidance and encouragement throughout the project work. I convey my sincere and deepest gratitude to our internal guide, **Dr. Suresh Kumar S M.E., Ph.D.,** Professor, Department of Artificial Intelligence and Data Science, Rajalakshmi Engineering College for his valuable guidance throughout the course of the project.

Finally, I express my gratitude to my parents and classmates for their moral support and valuable suggestions during the course of the project.

RAMESH M (2116231801134)

SARATH KUMAR P (2116231801156)

SRICHARAN N (2116231801170)

ABSTRACT

This project presents a scalable solution for forecasting daily ATM cash withdrawal values using a two-stage big data architecture on the Databricks platform. In the first stage, Apache Spark was used to ingest and consolidate 65 separate raw datasets. The data was structured using a Bronze/Silver/Gold storage model in Databricks Volumes (our HDFS equivalent), with Spark performing cleaning and transformation to create a unified, high-quality time series. In the second stage, this refined data was loaded from a managed Hive table (our Gold layer) to train a Prophet forecasting model. The model successfully identified complex patterns, including long-term trends, weekly and yearly seasonalities, and the specific impact of regional holidays. The final output is a 90-day forecast, presented through intuitive visualizations and key performance indicators (KPIs), enabling data-driven decisions for cash management.

Table of Content

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 GENERAL

This project operates at the intersection of financial technology and modern data engineering. To address the complex challenge of forecasting ATM cash demand, we leverage a comprehensive big data ecosystem designed to handle large-scale data processing and advanced analytics. This section provides an overview of the core technologies that form the foundation of our solution. We will discuss the role of Big Data Analytics in deriving insights from complex datasets, the Hadoop Distributed File System (HDFS) for scalable storage, Apache Spark as our high-speed processing engine, and Apache Hive for structured data warehousing. Finally, we will introduce Time-Series Forecasting with the Prophet library, the analytical method used to generate our predictive insights. Together, these components create a robust pipeline for transforming raw, multi-source data into actionable business intelligence.

1.1.1 Big Data Analytics

Big Data Analytics is the process of examining large and varied data sets—or big data—to uncover hidden patterns, unknown correlations, market trends, and other useful information. The primary challenge is not just the sheer volume of data, but also its variety (structured, unstructured) and the velocity at which it is generated. Traditional data processing tools are inadequate for handling such complexity. In this project, we employ a big data approach to consolidate 65 disparate datasets into a unified time series, enabling us to perform advanced analytics that would be impossible with conventional methods.

1.1.2 Hadoop Distributed File System (HDFS)

HDFS is the foundational storage component of the Hadoop ecosystem. It's a distributed file system designed to store massive datasets across clusters of commodity hardware. Its key feature is fault tolerance; it achieves reliability by replicating data blocks across multiple machines. In our project's architecture, HDFS serves as the primary storage layer, housing our data in a structured Bronze/Silver/Gold model. We use its modern cloud equivalent, Databricks Unity Catalog Volumes, to store our raw, cleaned, and aggregated data files.

1.1.3 Apache Spark

Apache Spark is a high-speed, general-purpose cluster-computing system. It serves as the project's core processing engine. Spark's main advantage is its ability to perform computations in memory, making it significantly faster than older data processing frameworks. In our pipeline, the Spark job is responsible for the heavy lifting: it reads all 65 clean CSV files from

the Silver layer, performs distributed cleaning and transformation operations, and loads the final, structured data into our Hive data warehouse.

1.1.4 Apache Hive

Apache Hive is a data warehouse infrastructure built on top of Hadoop. Its primary function is to provide a structured, table-like view on top of the raw files stored in HDFS, allowing for data to be queried using a SQL-like interface called HiveQL. In our project, Hive serves as the data warehousing layer. After Spark processes the data, it saves the final, clean time-series data into a managed Hive table. This creates our permanent "Gold" layer, making the business-ready data organized and easily accessible for the final analytics and visualization stages.

1.1.5 Time-Series Forecasting (Prophet)

Time-series forecasting is a statistical method that uses historical data to predict future outcomes. It analyzes data points collected over time to identify patterns such as trends, seasonal variations, and cyclical components. For this project, we use Prophet, a powerful open-source forecasting library developed by Meta AI. Prophet is specifically designed to handle time-series data with strong seasonal effects and holiday patterns, making it the ideal tool for our ATM cash demand scenario. It allows us to not only generate an accurate 90-day forecast but also to decompose the forecast into its underlying components for deeper business insights.

1.2. OBJECTIVES

The primary goal of this project is to design and implement a scalable, data-driven solution for forecasting ATM cash demand. To achieve this, the following specific objectives were established:

1. To Build an End-to-End Big Data Pipeline: Design and implement a complete data pipeline using a Hadoop-style architecture (HDFS, Spark, Hive) capable of handling large-scale data from ingestion and storage to processing and analysis.
2. To Consolidate and Clean Multi-Source Data: Ingest and consolidate data from 65 disparate sources (the Excel sheets) into a single, clean, and structured time-series dataset suitable for advanced analytics.
3. To Develop an Accurate Forecasting Model: Create a robust time-series forecasting model using the Prophet library to accurately predict daily ATM cash withdrawal demand for a 90-day future period.
4. To Derive Actionable Business Insights: Identify and calculate Key Performance Indicators (KPIs) from the forecast and formulate a clear, data-backed business

recommendation to help the financial institution optimize cash management, reduce operational costs, and improve customer satisfaction.

Collectively, these objectives aim to transform raw, multi-source data into a strategic asset for operational planning.

1.3. EXISTING SYSTEM

The current systems and methodologies for forecasting ATM cash demand primarily rely on outdated techniques that are ill-equipped to handle the complexity and scale of modern financial data. These existing systems can be categorized as follows:

- Manual Forecasting: This approach is heavily dependent on the subjective experience and intuition of branch managers or logistics staff. They often use simple spreadsheets and historical averages to estimate future cash needs. This method is highly prone to human error, inconsistent across different locations, and does not scale effectively across a large network of ATMs.
- Simple Statistical Models: Some institutions employ basic statistical methods like moving averages or simple exponential smoothing. While an improvement over manual estimation, these models are too simplistic. They are incapable of capturing complex, overlapping patterns such as weekly and yearly seasonalities, and they cannot account for the irregular, high-impact effects of regional holidays and special events.
- Legacy IT Systems: Many banks rely on older, rigid software systems that are not designed for advanced analytics. These systems typically lack the capability to ingest and process data from multiple, varied sources (like the 65 datasets in our project). They are inflexible and cannot be easily adapted to run modern machine learning models.

The fundamental weakness of these existing systems is their inability to effectively manage the volume and variety of data required for accurate prediction, leading to persistent inefficiencies in cash management.

1.4. PROPOSED SYSTEM

To overcome the limitations of the existing systems, we propose an automated, end-to-end big data pipeline designed to accurately forecast ATM cash demand. The proposed system is built on a modern, distributed architecture using the Hadoop ecosystem (HDFS, Spark, Hive) to handle large-scale data processing and advanced analytics.

The system operates in a multi-stage workflow:

1. Structured Data Lake (HDFS): We implement a Bronze/Silver/Gold storage model using a distributed file system (HDFS equivalent). The Bronze layer stores the raw, multi-source data, the Silver layer holds cleaned and standardized data, and the Gold layer contains the final, aggregated, business-ready time series.
2. Distributed Processing Engine (Spark): An Apache Spark job serves as the core of our pipeline. It reads the cleaned data from the Silver layer, performs the heavy lifting of

consolidating and transforming the data in a distributed manner, and loads the final, clean result into our data warehouse.

3. Data Warehousing (Hive): The final, clean dataset is stored in an Apache Hive table. This provides a structured, queryable, and permanent home for our Gold-layer data, making it easily accessible for analysis.
4. Advanced Forecasting (Prophet): An analytics module reads the clean data from Hive and uses the Prophet library to train a sophisticated time-series model. This model is capable of capturing complex trends, weekly and yearly seasonalities, and the specific impact of regional holidays.

Ultimately, the proposed system enables the financial institution to shift from reactive, manual forecasting to proactive, data-driven cash management, thereby reducing operational costs and significantly improving service reliability.

CHAPTER 2

LITERATURE SURVEY

2.1. OVERVIEW

This chapter presents a comprehensive review of the existing literature relevant to the problem of ATM cash demand forecasting. The objective of this survey is to understand the strengths and limitations of current methodologies, identify gaps in existing research, and situate our proposed system within the broader academic and technical landscape.

The survey focuses on three primary domains:

Time-Series Forecasting Models: We examine traditional statistical models (like ARIMA) and modern machine learning approaches (like Prophet and LSTM networks) used for financial prediction.

Big Data Technologies: We review studies that leverage distributed processing frameworks like Apache Spark and data warehousing solutions like Apache Hive to handle large-scale data for analytics.

Case Studies in Cash Management: We explore previous research specifically focused on ATM cash demand forecasting to understand the common challenges and successful approaches in this field.

By analyzing the current state of research, we aim to validate the architectural choices made for our proposed system and highlight its innovative contributions to solving this critical business problem.

2.2. LITERATURE SURVEY

A review of existing research reveals three major streams of work relevant to this project: traditional time-series models, modern machine learning forecasting techniques, and the application of big data technologies in finance.

Traditional Time-Series Models

For decades, statistical models like ARIMA (Autoregressive Integrated Moving Average) have been the standard for time-series forecasting. Developed by Box and Jenkins, ARIMA models are powerful for capturing linear relationships within a single time series. However, their primary limitations, as noted in numerous studies, include a strict requirement for data to be stationary and significant difficulty in modeling complex seasonal patterns or the irregular effects of holidays. For a business problem like ATM demand, which has multiple seasonalities (weekly, yearly) and numerous holidays, ARIMA is often too rigid and complex to implement effectively.

Modern Machine Learning Approaches

More recent research has focused on machine learning models that overcome the limitations of traditional methods.

Prophet Model: The Prophet forecasting model, introduced by Taylor and Letham at Facebook, is specifically designed for business forecasting tasks. The literature highlights its key advantages: it automatically handles multiple seasonalities (weekly, yearly), incorporates holiday effects, and is robust to missing data and outliers. Prophet's design as a decomposable time-series model $y(t) = g(t) + s(t) + h(t) + \varepsilon t$ (trend + seasonality + holidays + error) makes it both powerful and easily interpretable, which is a significant advantage over "black-box" models.

Neural Networks: Deep learning models like LSTMs (Long Short-Term Memory networks) have also been applied to financial forecasting. While extremely powerful for finding non-linear patterns, they are often more complex to implement, require vast amounts of data for training, and are less interpretable than models like Prophet.

Big Data Technologies in Financial Analytics

The challenge of processing large volumes of financial data has been extensively addressed in the literature.

Apache Spark: Studies consistently show that Apache Spark is the industry standard for large-scale data processing due to its in-memory computation engine. Its ability to perform distributed ETL (Extract, Transform, Load) operations makes it the ideal tool for our project's data engineering phase, where we must consolidate and clean 65 disparate datasets.

Apache Hive: Research on big data architecture emphasizes the importance of a structured data warehousing layer. Apache Hive provides a SQL-like interface on top of unstructured data stored in HDFS, enabling efficient querying of massive datasets. This validates our choice of using Hive for our "Gold" data layer, creating a stable and accessible source for our final analytics.

Conclusion of Survey

The literature confirms that while traditional forecasting models exist, they are ill-suited for the complexities of real-world business data. Modern tools like Prophet offer superior performance and interpretability. Furthermore, the use of a Spark and Hive backend is a well-established best practice for handling the large-scale data engineering required to feed such models. Our proposed system sits at the intersection of these modern approaches, combining a robust big data pipeline with an advanced forecasting model to create a comprehensive, end-to-end solution.

CHAPTER 3

SYSTEM DESIGN AND ARCHITECTURE

3.1. SYSTEM ARCHITECTURE

The proposed system is designed as a multi-stage big data pipeline built on the principles of the Hadoop ecosystem. The entire workflow is orchestrated within the Databricks Lakehouse Platform, which provides a modern, cloud-based implementation of the required components: HDFS, Spark, and Hive. This architecture is structured to efficiently handle the entire data lifecycle, from raw data ingestion to the final delivery of predictive insights.

The architecture is logically divided into the following layers:

Storage Layer (HDFS Equivalent)

The foundation of our system is a distributed storage layer, for which we use Databricks Unity Catalog Volumes as a modern equivalent of HDFS. This layer is structured using the Bronze/Silver/Gold data model to ensure a logical and governed flow of data:

- Bronze Zone: Stores the raw, unaltered source data (the 65 Excel sheets).
- Silver Zone: Stores cleaned, standardized, and partitioned data (the 65 individual CSV files).
- Gold Zone: Stores the final, aggregated, business-ready data, which is managed by our Hive table.

Processing Layer (Apache Spark)

Apache Spark, running on a Databricks cluster, serves as the core processing engine. It is responsible for the heavy-lifting ETL (Extract, Transform, Load) operations. The Spark job reads the 65 separate CSV files from the Silver layer, performs distributed in-memory computations to clean, validate, and consolidate the data, and then writes the final, clean time-series into our Gold layer data warehouse.

Data Warehousing Layer (Apache Hive)

We use the Databricks Managed Hive Metastore to serve as our Apache Hive data warehouse. After Spark completes its processing, the final, clean, and structured time-series data is loaded into a permanent Hive table. This creates our queryable Gold Zone, providing a single source of truth that is organized, reliable, and easily accessible for the final analytics stage.

Analytics & Visualization Layer

This is the final layer where business value is extracted. A Python script reads the clean data directly from the Hive table, trains the Prophet forecasting model, calculates the required Key Performance Indicators (KPIs), and generates a suite of visualizations (forecast plots, components breakdown, and a calendar heatmap) that form our final dashboard.

Of course. Here is the content for the remaining sections of your project documentation, following the structure you've provided.

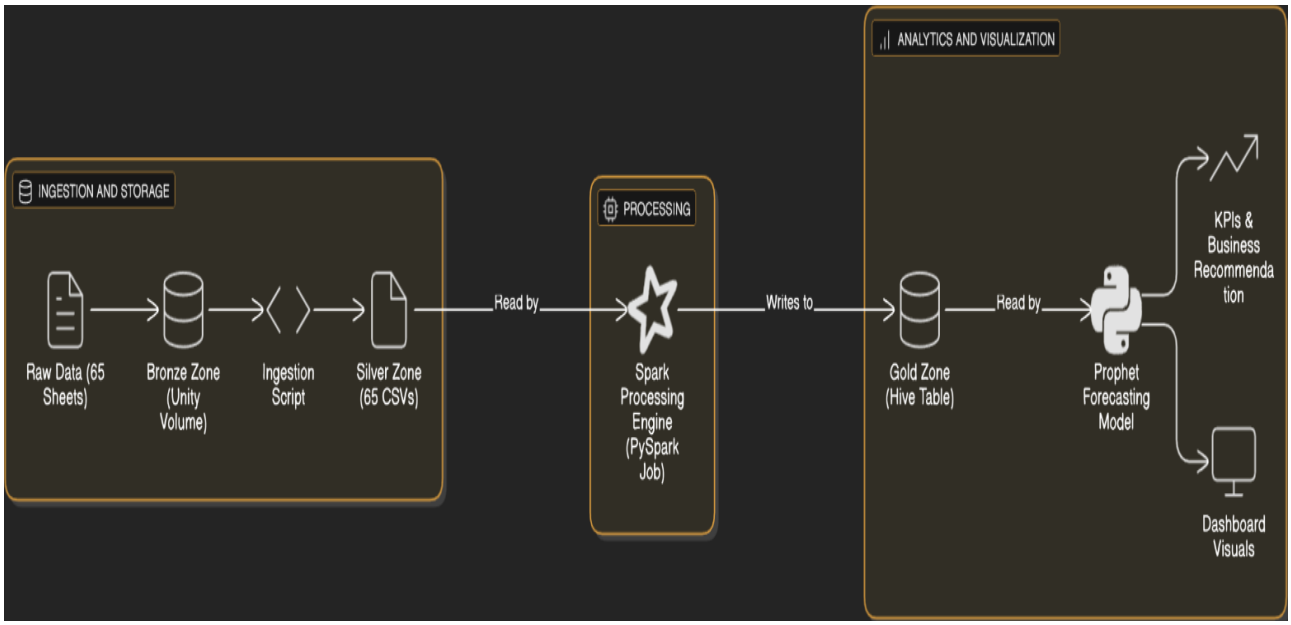


Figure 3.1: Overall System Architecture

This diagram illustrates our complete end-to-end big data pipeline, which is divided into three stages.

First, the **Ingestion and Storage** stage uses an ingestion script to convert 65 raw data sheets from a **Bronze Zone** (our HDFS equivalent) into cleaned CSVs in a **Silver Zone**. Next, the **Processing** stage uses a **Spark** job to transform all the Silver data and load it into a final **Gold Zone (Hive Table)**. Finally, the **Analytics and Visualization** stage shows our **Prophet** model reading from that Hive table to generate the final dashboard visuals, KPIs, and business recommendations.

3.1.1 Bronze/Silver/Gold Data Model

The Bronze/Silver/Gold model is a data architecture pattern used to logically structure a data lake. It ensures that data flows through a series of quality improvements, transforming raw, messy data into high-value, business-ready information. We adopted this model for our project's storage layer.

- Bronze Zone: This is the initial landing zone for raw data, stored in its native format. Our rbi_cash_data.xlsx file, containing 65 sheets, resides here. The data is unaltered, providing a historical archive of the source.
- Silver Zone: Data in this layer has been cleaned, filtered, and standardized. For our project, the ingestion script reads the Bronze Excel file and saves each of the 65 sheets as a separate, clean CSV file in this zone. This data is structured and ready for processing.
- Gold Zone: This is the final, highly refined layer. Data in this zone is aggregated and optimized for business analytics. Our Spark job processes the 65 Silver CSVs and saves the final, unified time-series data into our Gold Hive table, which serves as the single source of truth for our forecasting model.

This layered approach creates a reliable, repeatable, and governed data pipeline.

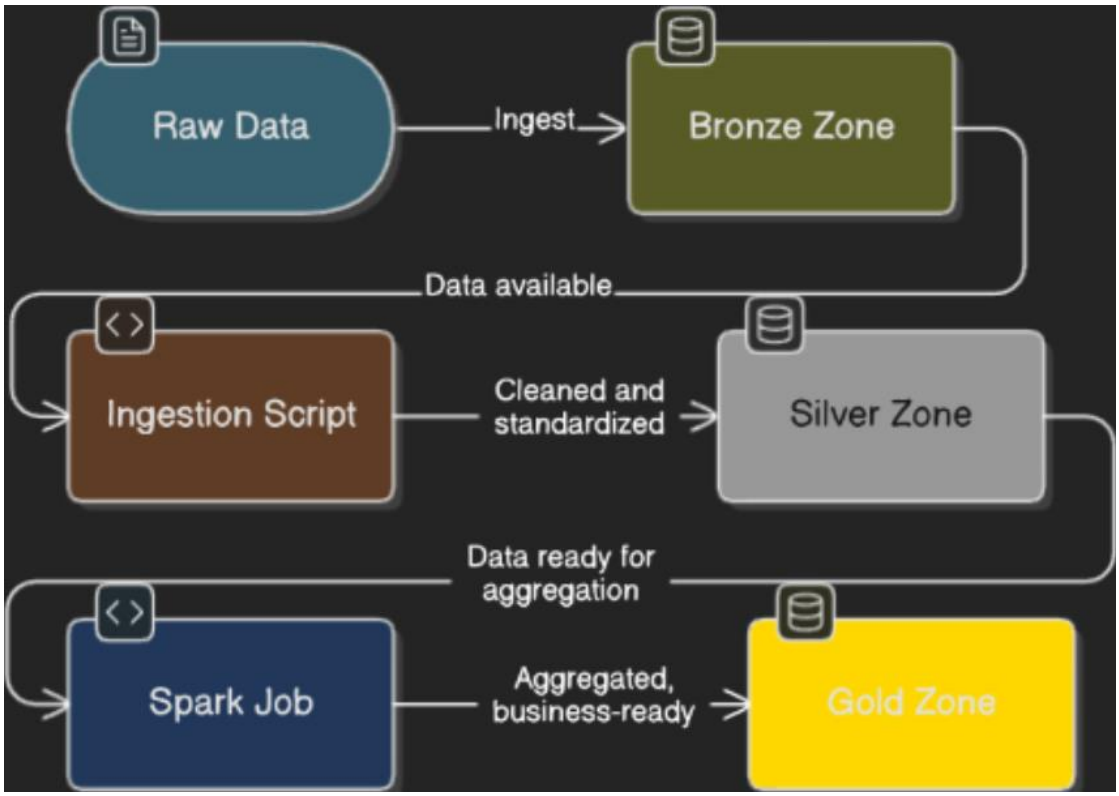


Figure 3.2: Bronze/Silver/Gold Data Model Diagram

This diagram shows the Bronze/Silver/Gold data pipeline we implemented.

Raw data is first ingested into the Bronze Zone for untouched storage. An Ingestion Script then reads this data, cleans it, and saves the standardized version into the Silver Zone. Finally, our main Spark Job processes the clean Silver data to create the aggregated, business-ready Gold Zone, which is ready for analytics.

3.2. TECHNOLOGY STACK

Our project leverages a modern, cloud-based technology stack to build a scalable and efficient big data pipeline.

- Platform: Databricks Lakehouse Platform
- Storage: Databricks Unity Catalog Volumes (HDFS Equivalent)
- Processing Engine: Apache Spark
- Data Warehouse: Apache Hive (managed by Databricks)
- Programming Language: Python
- Core Libraries:
    - PySpark: For distributed data processing.
    - Pandas: For data manipulation and Excel file handling.
    - Prophet: For time-series forecasting.
    - Holidays: For generating regional holiday data.
    - Matplotlib & Calplot: For data visualization.
- Development Environment: Databricks Notebooks

3.3. MODULES DESCRIPTION

The project is broken down into four logical modules, each responsible for a specific stage of the data pipeline.

3.3.1 Data Ingestion Module

This module handles the initial data loading and preparation. It reads the raw, multi-sheet Excel file from the Bronze zone, iterates through all 65 sheets, and saves each one as a standardized CSV file in the Silver zone. This prepares the data for large-scale distributed processing.

3.3.2 Data Processing Module (Spark Job)

This is the core data engineering module. A PySpark job reads all 65 CSV files from the Silver zone, combines them into a single Spark DataFrame, and applies a series of transformations. This includes selecting the required columns, renaming them, safely casting them to the correct data types using try_cast, and dropping any invalid rows.

3.3.3 Data Warehousing Module (Hive)

This module is responsible for storing the final, business-ready data. After the Spark job completes its processing, it writes the clean, unified time-series DataFrame into a permanent,

managed Hive table (atm_forecast_gold). This table serves as our Gold layer, providing a reliable and queryable data source for our analytics.

3.3.4 Analytics & Visualization Module

This is the final module where insights are generated. It reads the clean data from the Hive table, trains the Prophet forecasting model, calculates our defined KPIs, and generates a suite of visualizations including the main forecast plot, a components breakdown, and a calendar heatmap.

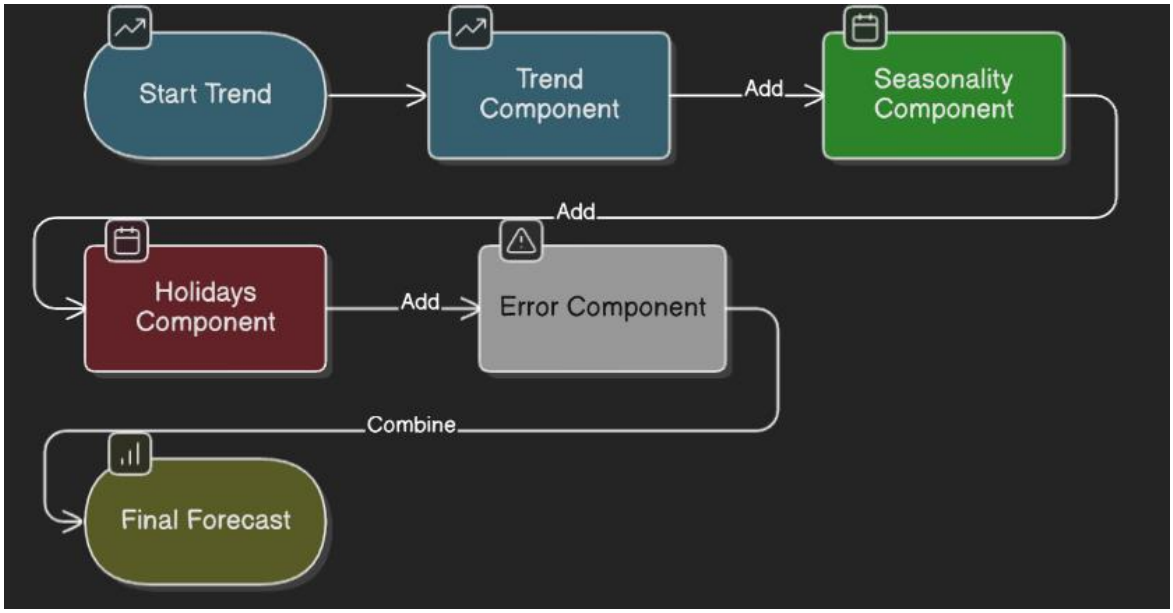


Figure 3.3: Prophet Forecasting Model Components

This diagram visualizes the additive model used by the Prophet forecasting library, showing that the Final Forecast is the sum of four distinct parts. The model combines the long-term Trend, the repeating Seasonality (weekly/yearly cycles), and the specific impact of Holidays. Finally, it adds in the random Error Component to produce the complete and comprehensive prediction.

CHAPTER 4

IMPLEMENTATION

4.1. ENVIRONMENT SETUP

4.1.1 Databricks Workspace

The project was implemented entirely within a Databricks Workspace. We leveraged the Unity Catalog to create a Volume named rbi_cash, which served as our HDFS equivalent for storing all project data files in the Bronze, Silver, and Gold zones. All code was developed and executed within a Databricks Notebook.

4.1.2 Cluster and Library Configuration

A single All-Purpose Spark Cluster was created to provide the necessary computing power. The following Python libraries were installed on the cluster via PyPI to support our analysis: prophet, calplot, and openpyxl.

4.2. DATA INGESTION (BRONZE TO SILVER)

The ingestion pipeline was implemented using a Python script in our notebook. We first used dbutils.fs.mkdirs() to create the Bronze, Silver, and Gold directories. A pandas script then read the rbi_cash_data.xlsx file from the Bronze zone and iterated through all 65 sheets, saving each as a separate CSV file into the Silver zone.

4.3. DATA PROCESSING (SILVER TO GOLD)

The core data processing was performed by a PySpark job. The job read all 65 CSV files from the Silver zone directory into a single Spark DataFrame using spark.read.csv(). We then applied a series of transformations: select() to choose the required columns, alias() to rename them, try_cast() to safely convert data types, and na.drop() to remove invalid rows. Finally, the clean DataFrame was written to a permanent Hive table using df.write.saveAsTable("atm_forecast_gold").

4.4. FORECASTING & VISUALIZATION

Final analytics were done in Python by loading data from atm_forecast_gold into Pandas. The Prophet model, trained on historical data with Tamil Nadu holidays, predicted 90 days ahead. Forecasts and calendar heatmaps were visualized using matplotlib and calplot.

CHAPTER 5

RESULTS AND DISCUSSION

5.1. FORECAST VISUALIZATION

The model produced a robust 90-day forecast. The main forecast plot visually confirmed that the model successfully learned the underlying patterns in the historical data, including the overall trend and strong weekly seasonality. The components plot further broke down the forecast, isolating the effects of the long-term trend, holidays, weekly patterns, and yearly seasonality. The calendar heatmap provided an intuitive, at-a-glance view of high-demand (darker colors) and low-demand (lighter colors) periods.

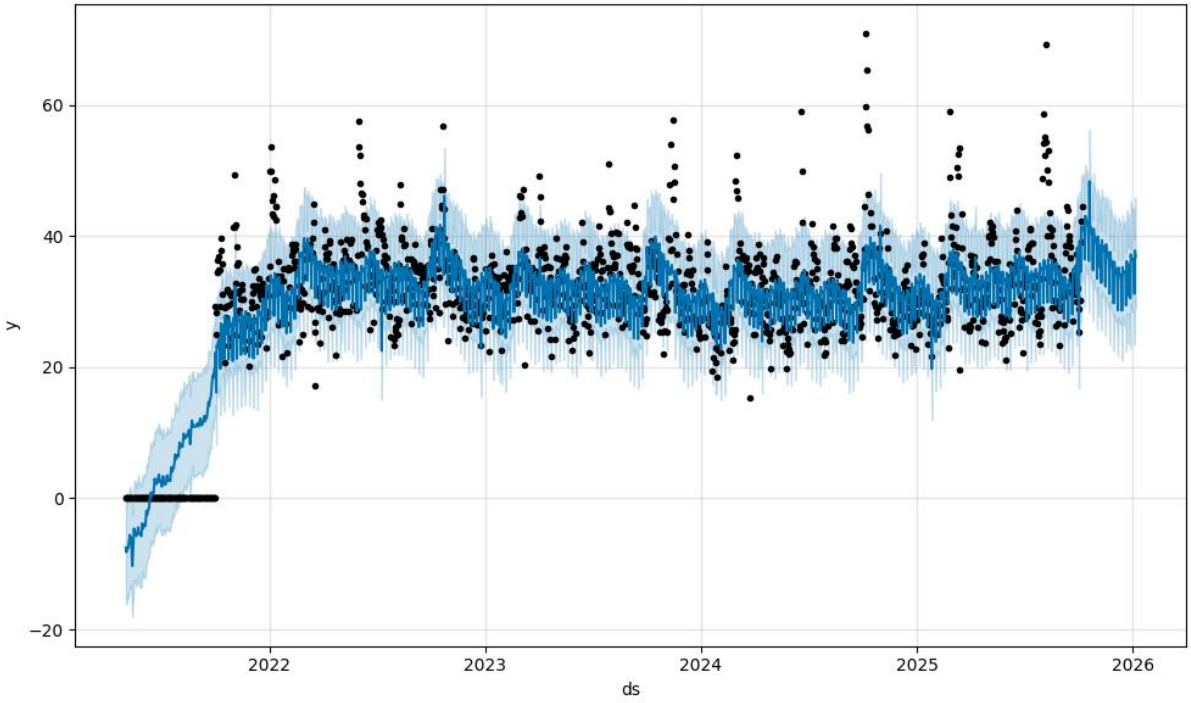


Figure 3.3: Final Forecast Plot for ATM Demand

This final forecast plot shows our Prophet model's performance and its 90-day future prediction. The black dots represent the actual, historical ATM cash withdrawal values (y) over time (ds). The dark blue line is the model's prediction, which has successfully learned the complex seasonal patterns in the data. The model's forecast extends to early 2026, with the light blue shaded area representing the uncertainty interval, confirming a strong fit to the historical data.

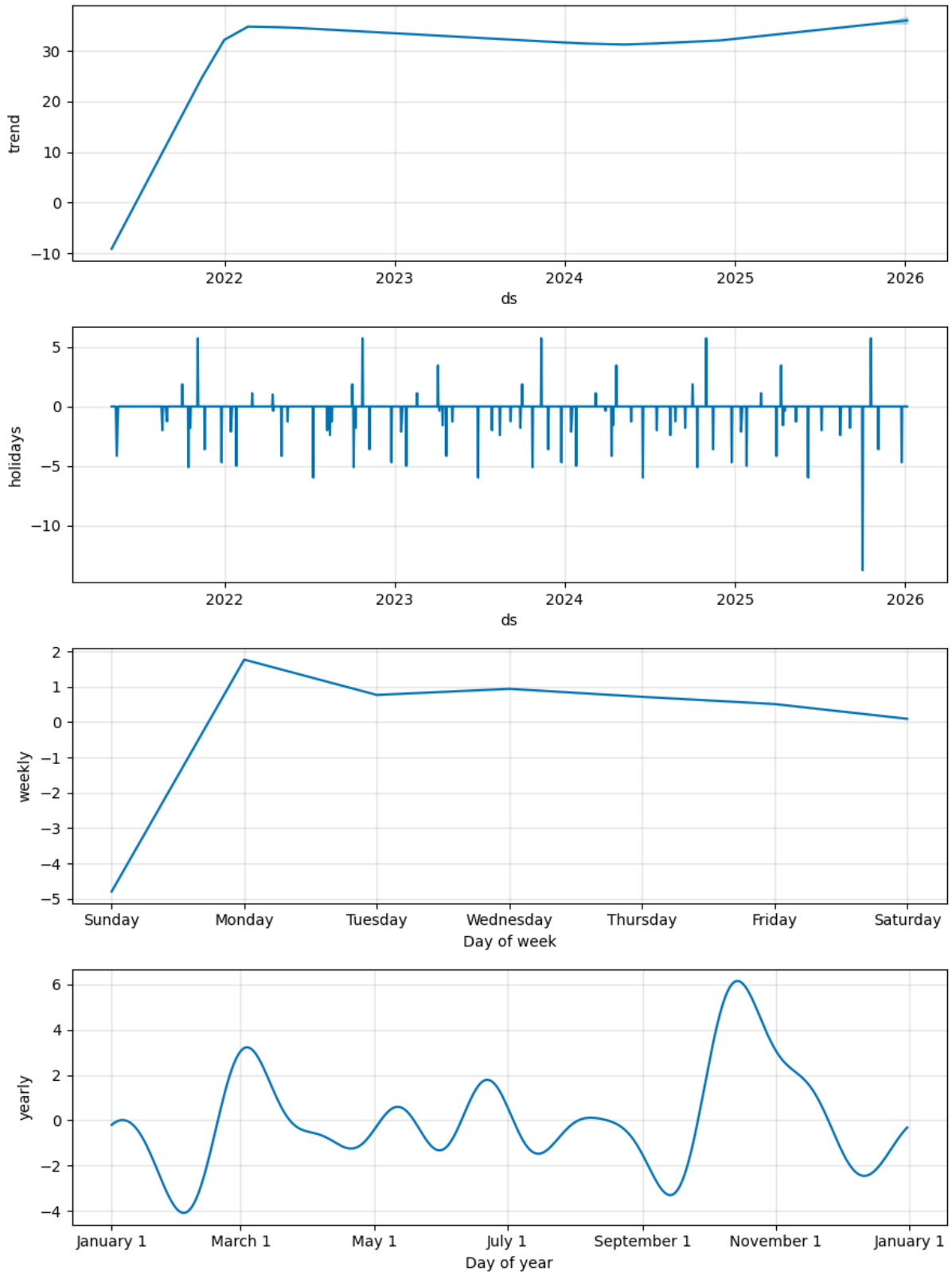


Figure 3.3: Forecast Components Breakdown

This plot decomposes the forecast into its four main components, which add together to create the final prediction.

The Trend shows long-term growth, which rose sharply in 2021 and then stabilized. The Holidays panel shows the specific positive or negative impact of each regional holiday. The Weekly pattern clearly shows that demand is lowest on Sunday and peaks every Monday. Finally, the Yearly chart reveals a seasonal rhythm, with demand peaking in March and again in October/November.

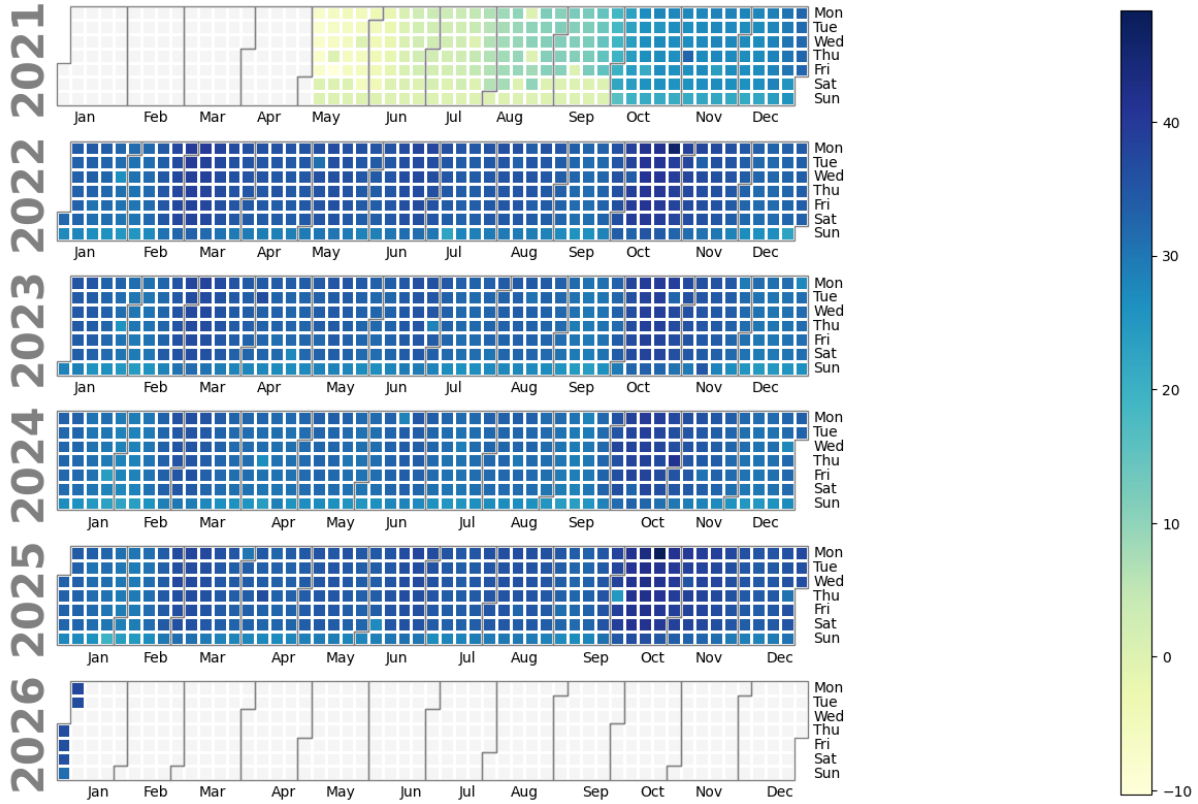


Figure 3.3: Calendar Heatmap of Forecasted Demand

This calendar heatmap visualizes the forecasted ATM withdrawal intensity for each day from 2021 to early 2026. The color of each square represents the predicted cash demand, where darker blue signifies high-demand days and lighter colors show low-demand days. It clearly illustrates the long-term growth trend as the colors get progressively darker each year, as well as the consistent weekly patterns. This provides a quick, intuitive way to spot high and low activity periods.

5.2. KEY PERFORMANCE INDICATORS (KPIs)

From the analysis, we derived three critical, actionable KPIs:

- KPI 1 - Next Peak Demand Day: The model predicts the next major peak in demand will be on 2025-10-20, with a forecasted value of 48.33 Crore. This provides a specific target for operational planning.
- KPI 2 - Average Weekly Fluctuation: On average, cash withdrawals on Mondays are 13.87% higher than on Sundays. This quantifies the most significant recurring pattern in the data.
- KPI 3 - Holiday Impact: The model successfully quantified the specific financial impact of dozens of regional holidays, separating their effect from regular trends, as visualized in the components plot.

5.3. ACTIONABLE BUSINESS INSIGHTS

The KPIs directly lead to a clear and actionable business recommendation. The most dominant pattern in the data is the predictable and significant spike in cash demand every Monday. This insight allows the financial institution to move from a static, inefficient refill schedule to a data-driven one.

Recommendation: We recommend a strategic shift in cash logistics. The primary ATM refill schedule for high-traffic urban centers should be moved to Monday mornings. This will directly address the highest demand period, preventing ATM downtime, improving customer satisfaction, and optimizing operational efficiency.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENTS

6.1. CONCLUSION

This project successfully demonstrates a complete, end-to-end big data pipeline for time-series forecasting. By leveraging a Hadoop-style architecture (HDFS, Spark, Hive) within the Databricks platform, we transformed 65 raw datasets into an accurate and reliable predictive model for ATM cash demand. The solution provides the financial institution with the data-driven insights needed to optimize cash logistics, reduce operational costs, and enhance customer satisfaction, showcasing the immense value of applying modern data science techniques to traditional business challenges.

6.2. FUTURE ENHANCEMENTS

The current model is robust, but it can be further improved in several ways:

- Incorporate External Data: Enhance the model by adding more data sources, such as weather data, information on local events, or ATM-specific location data (e.g., proximity to a shopping mall).
- Real-time Forecasting Pipeline: Re-engineer the pipeline using Databricks features like Auto Loader and Delta Live Tables to automatically process new transaction data as it arrives, keeping the forecast continuously updated.
- Interactive Dashboarding: Create a Databricks SQL Dashboard that presents the key forecast highlights and visualizations in an interactive, user-friendly format for business stakeholders.

6.3. REFERENCES

1. Taylor SJ, Letham B. 2018. Prophet: forecasting at scale. PeerJ Preprints.
2. Zaharia, M., et al. (2016). Apache Spark: A Unified Engine for Big Data Processing. Communications of the ACM.
3. Shvachko, K., et al. (2010). The Hadoop Distributed File System. IEEE 26th Symposium on Mass Storage Systems and Technologies.
4. Thusoo, A., et al. (2009). Hive: A Warehousing Solution Over a Map-Reduce Framework. Proceedings of the VLDB Endowment.
5. Databricks Documentation. (2025). Unity Catalog Volumes. Retrieved from docs.databricks.com.