SABARAGAMUWA UNIVERSITY OF SRI LANKA
FACULTY OF COMPUTING
DEPARTMENT OF SOFTWARE ENGINEERING
BSc HONOURS DEGREE PROGRAMME IN SOFTWARE ENGINEERING
SE 8101 Research Project in Software Engineering

Research Proposal

Student Name	Vijayakumar **Janarthan**	Student Index Number	19APSE4276
Research Topic	Time Series Forecasting of User Engagement in Python Software Applications Based on Usage Logs		
Supervisor	**Mrs. JDT Erandi** Lecturer (Probationary) Faculty of Computing - SUSL		
Co- Supervisor	**Mr. YMS Tharaka** Lecturer (Probationary) Faculty of Computing - SUSL		

Agreement

I …Vijayakumar Janarthan……, willing to submit the following Proposal format and I am confirming that this topic is novel research, and I am aware of the consequences occurs in the case of plagiarism found.

V Janarthan
…… ……30th May 2025……
Signature Date

Background (max of 2 pages)

There is a big question raise among development team and client that "how users interact with an application," which is quite active and common query, *User Engagement* is an important term to consider during the discussion of any Python application software. Through various metrics such as feature adoption, session duration, and daily active users (DAU), developers can comprehend user engagement and improve software experience and retention [1], [2], [3]. For applications developed in Python such as data science application and web frameworks, engagement behavior is usually stored in usage logs containing information like timestamps, feature usage, and error occurrences. Although these logs offer a valuable resource for examining user interaction with software, the challenge lies in making accurate predictions based on this data. While traditional analytics systems are helpful for looking back and evaluating engagement, they tend to lack the ability to predict the trends of user interaction, restricting proactive strategies [4], [5].

During the development and usage of the software application, predicting user interactions is not an easy task. Many reasons like feature improvements, irregular periodic usage, and even outside factors can cause a user's activity to change tremendously [6], [7]. Take this as an example, the Python library called "PDFplumber" used in academic learning software that help students to learn with pdfs efficiently. It could have a boom in usage activity during the semester exams and drop off afterward or experience major updates. Cohort analysis along with basic regression models, as many others, seem to ignore temporal dependencies within the usage of data, which leads to poor predictions [8], [9]. This specific gap in the existing frameworks displays the need for more sophisticated time-series techniques that incorporate trend and seasonal patterns, as well as irregular components that define the engagement metrics.

Time series forecasting is a powerful temporal data analysis solution. Models such as ARIMA (Autoregressive Integrated Moving Average) and LSTM (Long Short-Term Memory) networks are effective in recognizing patterns from the past and estimating future values [10], [11], [12]. ARIMA is adept at modeling linear trends and seasonal fluctuations which is useful for consistent engagement patterns [13], [14]. Meanwhile, LSTM networks, a subclass of recurrent neural networks, are superior to classical methods in the presence of complex user behavior because they manage nonlinear relationships and long-range dependencies [11], [15]. Other approaches, like integrating statistical models using machine learning, have also been effective in accurate improvement of the forecasted models [7].

Python has a comprehensive ecosystem that implements statistical models like ARIMA which can be done through the library stats models, automated forecasting can be done using Prophet, and LSTM networks can be made using TensorFlow [16], [15]. The application of Python permits smooth connections with data preprocessing workflow, which allows transforming raw usage logs into time series data sets. For instance, calculating the daily active users or calculating how often a session is run shows trends which will determine which models to use [17]. Also, Python's graphing tools like Matplotlib and Seaborn provide tools for examining data to determine whether there are seasonal automorphisms or anomalies in the engagement data [15], [5].

There are many ways to implement engagement forecasting. For instance, accurately predicting a drop-off of active users can help in designing retention plans, such as interactive prompts on features walkthroughs [6], [9]. As far as open-source projects implemented in Python are concerned, predicting contributor behavior or issue resolution can greatly improve project sustainability [18], [3]. Snapchat and other retail stores are already proving the benefit of temporal analysis on churn reduction and enhanced engagement and experience [8], [7]. Still, most Python applications operate with responsive metrics and do not employ proactive forecasting, which leads to misapplied strategies in addressing user demands and resource distribution [1], [5].

This research attempts to fill the gaps by developing a particular time series architecture for use within Python. Other studies focus on general engagement metrics across industries, but very few center on the unique challenges posed by Python, such as irregular usage patterns with developer tools and dependency on external libraries [8], [6]. Using logs of usage and analyzing ARIMA, LSTM, and hybrid models, this study seeks to offer pragmatic guidance to Python developers on forecasting engagement, feature prioritization, and user satisfaction. Such results could begin the use of temporal analytics for open source and proprietary Python systems and integrate temporal data collection with post-analysis strategic planning [11], [15], [7].

SABARAGAMUWA UNIVERSITY OF SRI LANKA
FACULTY OF COMPUTING
DEPARTMENT OF SOFTWARE ENGINEERING
BSc HONOURS DEGREE PROGRAMME IN SOFTWARE ENGINEERING
SE 8101 Research Project in Software Engineering

This research aims to accomplish (purpose) a time series forecasting model tailored towards predicting user activities through system log files of applications written in Python. This study attempts to propose a model using classic methods and modern techniques of advanced and machine learning along with hybrid approaches that guarantee high accuracy alongside practicality.

The results of this study will serve multiple groups: developers using Python will have an opportunity to deepen their understanding of user interaction and optimize feature implementation; managers and product teams will have foresight for strategic engagement planning; and other researchers will study the relevant gaps pertaining to software usage data forecasting. In general, this study is aimed at enhancing retention, planning, user satisfaction, and foresight in the ecosystem of Python applications.

Literature Review (max of 5 pages)

This detailed literature review explores new developments in user engagement prediction for Python applications, specifically on time series analysis techniques, machine learning models, and their applications. This review attempts to integrate findings from the most recent studies to present a coherent picture of the field, its methods, and emerging trends.

Importance of Predicting User Engagement

In the context of software engineering, particularly for mobile and web development with Python, *user engagement forecasting* has emerged as an important research area. The ability to predict user interaction with the software determines the level of strategic thinking, decision making, and resource distribution at hand, and which features need to be given priority. As Liu et al. noted, predictive analytics… has been on the rise for industries, as "financial institutions increasingly turning to Artificial Intelligence (AI) for enhanced risk assessment and management [19]." This also applies to software applications because the ability to anticipate future interacting behaviors of users can affect business and user satisfaction.

Research Gap and Problem Statement

Modern applications collect a considerable amount of usage data, but transforming it into precise, accurate, and practical predictions is still a difficulty. The lack of purposive forecasting models tailored towards user engagement data within a software context is the main gap of the research. In the same manner, other industries, like economics and meteorology, have well-established forecasting methodologies, but software applications offer peculiar hurdles such as sparse usage, engagement dictated by features, and other contingent activities like users escalated usage [8]. All these issues are especially prominent to applications in Python, which range from data science applications to web framework tools, with each domain having its own peculiar usage patterns.

Important Constructs and Terminology

To provide clarity, this review focuses on user engagement as active interaction with an application, captured by metrics like session length, feature engagement, and user retention. As discussed in many studies, *engagement* means "the attention and interactivity and controllability and impression to the public users" [20]. Making predictions based on the past is known as *forecasting* [21], while *time series analysis* consists of a set of methods designed explicitly for data indexed in time sequence [22]. The *Python ecosystems* describe the libraries and frameworks made accessible to developers for executing the forecasting techniques [23].

Overview and Organization

This review analyzes the literature from 2019 to 2025, concentrating on scholarly articles, conference papers, and technical reports that focus on forecasting approaches relevant to Python applications. The structure is organized thematically, starting with statistical methods towards sophisticated and practical applications of machine learning, hybrid models, and beyond. In this review, general studies in user experience will not be included as primary focus is placed on the quantitative forecast of engagement metrics due to limited available space. The strongest consideration in this selection is based on the reasoning from studies that provided claimed evidence, innovative methodology, or guidance from practical implementation.

Historical Development of Statistical Techniques

User engagement prediction is anchored in historical statistical methods, especially ARIMA models which have had a strong impact. “ARIMA models are a general class of models used for forecasting time series data . . . Usually denoted as ARIMA (p, d, q) where p is the order of autoregression, d is the degree of differencing, and q is the order of moving average [24].” Because these models effectively capture linear relationships and recurring seasonal trends, they are invaluable in forecasting consistent engagement levels.

Practical guidance to perform ARIMA out-of-sample predictions in Python is provided by Machine Learning Mastery, who also notes the need for seasonal differencing [25]. “The data has a strong seasonal component. We can neutralize this and make the data stationary by taking the seasonal difference [26].” This is useful in situations where there is yearly or quarterly engagement, such as in academic instructional application software or in tools designed for preparing tax returns. The article shows the steps developers need to perform stationarity to remove non-static properties, a prerequisite to accurate ARIMA modeling.

The baseline evaluation for predicting social media time series streams has been done by the researchers, and that work serves as a foundation for further methods [27]. Some of the more traditional models do seem to approximate forecasting engagement quite well, particularly those with a strong seasonal tendency and low non-linear complexity, as the researchers have demonstrated. However, with increasing complexity of applications and unpredictable user behaviors, it becomes exceedingly difficult to apply these traditional methods.

The traditional methods have strong support in the form of python libraries such as 'statsmodels' [28]. It provides ARIMA along with other statistical models via a scikit-learn friendly interface. This ease of access has allowed developers untrained in statistics to easily step into the world of time series analysis with python. Still, as several studies point out, traditional frameworks for statistics are unable to handle nonlinear relationships and intricate dependencies with modern user engagement data, which is purely boundless in nature, and this is why more complicated machine learning algorithms are being issued instead [29], [30].

Approaches Involved in Machine Learning for Complicated Engagement Patterns

Recently, user engagement projection has shifted towards a more machine learning oriented approach due to the limitations posed by traditional statistical techniques. Predictive analytics using deep learning models are now considered superior because of their ability to capture the conflict patterns and non-linear relationships associated with time series data. A survey has noted that "Deep learning based TSF [Time Series Forecasting] tasks stand out as one of the most valuable artificial intelligence scenarios for research and play an important role in explaining complex real-world phenomena" [31].

Along with the above claim, long short-term memory (LSTM) networks have become quite popular because they can accommodate long range dependencies in chronological data. These networks are especially useful for applications in user engagement due to their capability to 'model both linear and (nonlinear) patterns' [32]. The architecture of LSTM networks consists of specialized memory cells which enable the model to retain information over long sequences, thereby solving the vanishing gradient problem suffered by older recurrent neural networks.

Graph Neural Networks (GNNs) time series, especially in cases where interdependencies between different time series matter. A GNN survey for time series data communicates four dimensions where these models are useful which are, "forecasting, classification, anomaly detection, and imputation" [33]. These attributes simplify engagement analysis with GNNs as predictive analytics are often used alongside pattern anomaly detection and log data imputation.

GNNs also enable novel capabilities such as adaptive user behavior forecasting, real-time activity prediction and efficient resource utilization by processing vast amounts of streaming data in real-time. The transformer architecture has also been adapted for time series forecasting with promising results. It has been stated in more recent research that "Transformer models, which are adept at long-term dependencies, are increasingly becoming important design elements in time series forecasting" [34]. This approach enhances attention mechanisms as it enables models to concentrate on pertinent sections of the historical data sequence, which is important considering user engagement forecasting where the current behavior may be influenced by events from the distant past.

A multivariate time series analysis is reviewed and described as being from a "channel strategy perspective," shedding light on how models utilize correlations between user metrics and user engagements [35]. In another case, broader scope is covered deep learning time series forecasting where it is said that "recent research has shown that alternatives such as simple linear layers can outperform Transformers" in some instances [36]. It further illustrates the emphasis on choosing appropriate models based on data engagement features.

Even with their strengths, machine learning methods come with high data, resources, and complexity of the model to be computed, which raises issues of interpretability. For example, deep learning models still face challenges.

"They need to deal with the challenge of large-scale data in the information age, achieve longer forecasting ranges, and reduce excessively high computational complexity, etc." [31]. These restrictions have led to the investigation of purely dependent methods that compound the weaknesses of each forecasting strategy while enhancing individual strengths.

User Engagement Predictive Analytics Using Hybrid Models

The combination of statistical and machine learning techniques increases the accuracy of predicting user engagement. For example, Kong et al. (2021) proposed a hybrid model that combines Facebook's Prophet with Long Short-Term Memory (LSTM) networks. The methodology consists of first fitting time series data using Prophet to capture linear and seasonal trends. The residuals from this model are then used to train an LSTM network, which captures nonlinear relationships. This two-stage approach is better than using either model alone [37].

The same study also emphasized the effects of holidays as part of the predictors in the forecasting model. Inclusion of holidays, as an external factor which can greatly impact user activity, enhanced the hybrid model's performance which demonstrates the importance of adding specific contextual components from other domains to forecasting models [37].

Another method of user engagement forecasting is Dynamic Intent Guided Meta Network (DIGMN), which specifically characterizes and models user intent over a span of time. This model attempts to resolve the gap created by traditional forecasting approaches which tend to ignore the temporal changes in user intent. Through the lens of intent, engagement behaviors provide discriminatory accuracy that sharpens DIGMN's projections [38].

Stromer et al. (2024) highlights the usefulness of forecast tools with [39],

- Incremental support: Support that builds in both understanding and certainty.
- Model Explanations: The Explanations provided are logical from a human cognitive perspective.
- Domain Knowledge: External knowledge can be embedded into the system [39].

These tools are emphasized within Stromer's non-expert analytics framework while keeping case studies in mind [39]. These blended approaches make use of transparent statistical elements together with sophisticated machine learning parts. Such strategies answer the outlined needs and thus are applicable in business environments where stakeholder comprehension is paramount.

Contextual Factors in User Engagement Forecasting

In Liu et al. (2019), they used action graphs to represent each user's activities in Snapchat as temporally evolving action graphs. These graphs depict and inform user actions, and user actions inform user behavior models. The model proved engagement forecasts greatly benefited from the behavior patterns captured. It was shown that within-app action sequence transitions structurally represented offered more information than time series representations [8].

Tang et al. (2020) designed the FATE framework, which includes friendship relations as user actions and temporality associated as the three most important components affecting user engagement [40]. The FATE framework employs tensor-based GNNs, LSTMs, and attention-based mechanisms for tensor operation to provide explainable engagement forecasts. The FATE framework does not only provide higher accuracy, but also lower networked used, improved training and inferencing times [40].

Li et al. (2022) created the Dynamic Intent Guided Meta Network (DIGMN) model to track intent user changes of users on professional social networks like LinkedIn over time [41]. It explicitly shows that users selectively decide and engage with the platform for various purposes on job search, networking, etc. User intent shifts are mitigated with DIGMN because the model forecasts errors of engagement cyclically [41].

Kong and colleagues (2021) underlined the impact of holidays and similar external events on user engagement [20]. Adding holiday effects to the predictive models increased accuracy, which illustrates the integration of forecasting frameworks with calendars of relevant cyclic events to be integrated into forecasting pipelines as a calendar event for cyclic impact is needed [20].

There is ample literature showing that the engagements of older adults with mobile health (mHealth) applications are different from other age groups [42]. For instance, Pywell et al. (2020) described some of the barriers an older adult might have toward mental health interventions using mobile technology, such as lack of awareness or trust [43]. Similarly, Wang et al. (2019) performed a systematic review and reported older adults' willingness to engage with health information tracking, but noted adoption was stalled due to expense, privacy, and ease of use issues [44]. Such information underscores the need to customize some forecasting models to specific demographic considerations to improve accuracy in varied user populations.

Tools and Frameworks in Python Implementation

Meta's *prophet* is an open-source library for forecasting time series data. Prophet combines yearly, weekly, and daily seasonal components along with holiday effects to model non-linear trends. Prophet is especially useful for analysts and developers due to fantastic missing data handling and intuitive parameters. Its web traffic forecasting capabilities have been validated by several studies [45], [46].

Temporal-GCN-LSTM is a more sophisticated model that combines GCNs with LSTMs to capture complex user behaviors. This end-to-end multi-channel neural model predicts future user engagement by encoding temporal evolving action graphs, thus successfully capturing high-order information in action graphs [8].

The purpose of *TimeGym* is to test and debug forecasting pipelines built using Python. It helps automation in testing by providing generalized tests for forecasting pipelines so that developers can fix problems in their forecasting logic during 'forecaster sandbox' sessions before placing forecasting systems into production [47].

A framework has been developed in Python for prediction of event logs in a streaming mode which allows for generation of predictions while data is being streamed from a business process. This framework allows for the application of streaming algorithms such as n-gram language models and LSTMs that can update engagement forecasts in real-time [48].

New methods of *eco-visualization* which combine art and technology aim to reduce electricity consumption by enhancing user engagement through electricity usage feedback. Such behaviors are intended to encourage sustainable use of resources by users [49].

Sophisticated business intelligence tools like forecasting systems require an advanced technology stack. The optimal user experience on the other end makes use of *Docker* for containerization, APIs are constructed on *Fast API,* models are served through *BentoML*, and orchestration is done with *Kubernetes* and *Docker*. This approach allows reliable and scalable dissemination of data engagement forecasts to various applications and dashboards [50].

Summary of Key Findings

This review investigated user engagement prediction within Python frameworks and uncovered notable patterns. Statistical models are useful in providing the baseline when predicting modern user data, such as *ARIMA* models, however, they are far too simplistic, linear, and constrained when dealing with the complex, dynamic user behavior of today. Nevertheless, most machine learning models, e.g. *LSTMs* and *GNNs*, or even transformers, tend to do poorly with scant data and minimal computing resources. They also impose a greater requirement for organization and order.

All three of these factors, *accuracy*, *efficiency*, and *explainability* are critical to making hybrid models, which are far more effective as they integrate elements from both statistical and machine learning methods. Such models are particularly beneficial in pragmatic scenarios where system responsiveness is equally essential as system transparency. Moreover, forecasting context features such as intent, social networks, public holidays, and demographic data significantly enhance model accuracy. The Python ecosystem provides a range of tools, from user-centric right through to specialized, such as deep learning libraries (*PyTorch, TensorFlow*), *TimeGym* for model diagnostics, and even user-friendly options like *Prophet*.

Gaps in Existing Research – Detailed Study (max of 2 pages)

Even as interest grows in anticipating user engagement, the attention given to the area which concerns Python scripts, and their attributes remain shallow. While many forecasting techniques have been adopted for social networks, online shopping, and mobile applications, there is a lack of application of time-series forecasting models to educational software, development tools, and scripting tools for Python. These types of applications exhibit erratic bursty patterns of usage driven by exogenous work seasonality such as semester dates and project deadlines, which pour heavily controlled workstreams, posing significant difficulties for traditional models, particularly ARIMA, to depict properly.

There is an apparent deficiency in customized forecasting frameworks concerning Python software applications. Most engagement forecasting analyses are based on highly controlled settings with abundant usage data, such as Facebook or LinkedIn, while Python applications log domain-specific usage patterns with sparse logs and intricate usage subtleties.

Academic tools, such as PDF processing libraries, scientific computation environments, or personal automation scripts, for example, face periodic surges instead of daily consistent engagements. Known models fail to sufficiently address low session interval gaps, infrequent user interactions, interdependent library usage, and non-user intervention making them rendered useless when applied to social media contexts resulting in the absence of social media contextual usage on the systems.

Furthermore, although hybrid models like Prophet-LSTM and SARIMA-BiLSTM show heightened precision in social and environmental forecasting, there are no hybrid models focused explicitly on engagement forecasting in python software development contexts. There has been no research evaluating the predictive accuracy of these combinations on usage logs from Python-based systems. In addition, there is little attention paid to how hybrid residual-based modeling frameworks can be customized for domain-specific seasonal phenomena like version changes, user onboarding, or documentation shifts which are prevalent in tools tailored for developers.

Another related gap is the absence of attention to contextual tailoring within batch forecasting pipelines. Some holidays and major milestones are customarily integrated into the models, but recurring, domain-specific calendar schemas like Python package release cycles or annual developer conferences are often overlooked and not integrated into forecasting systems in a coherent manner. The FATE and DIGMN frameworks are some of the first efforts to encode user context, but these are geared toward real-time systems and large-scale social networks and not lightweight interpretable batch-process systems designed for Python applications. There is a demand for hybrid frameworks that lean on batch processing and incorporate context variables pertaining to domain features, such as model update logs and error spike annotations, to enhance the efficiency and clarity of explanatory models.

Moreover, evaluation frameworks from the literature are built around plentiful datasets, extensive labels, and rich metadata. Very little has been done to establish comprehensive benchmarking and testing frameworks (like TimeGym) for low-resource, sparse, or semi-structured engagement logs, the type produced by the Python scripts leveraged by small teams or solo developers. This fragmentation hampers system reproducibility and makes cross-system comparison difficult in lightweight context-agnostic forecasting frameworks.

Most of the literature does not look like a deployable design of forecasting systems within Python environments. Although FastAPI, BentoML, and Docker put interfaces to predictive services, the academic literature is shallow on streamlined batch-oriented engagement forecasting log cut-to-prediction output and dashboard rendering workflows. A complete, Python-optimized, end-to-end stack visualization decision support pipeline has yet to be authored by the ingested-data-focused researchers.

Gaps considered most crucial – Summary

- There is no custom forecasting models designed to accommodate the peculiar usage patterns associated with Python-based applications.
- There are no domain-specific compositional hybrid models of engagement analytics with software instruments (e.g. Prophet-LSTM, ARIMA-GRU).
- There is a limited consideration of domain-derived seasonality (e.g. library update, documentation release) feature integration into the forecasting pipelines.
- There is scarce attention given to contextual fusion within batch processing frameworks that provide explainable results.
- There is no evaluation criteria established for engagement logs with low-resource and sparse user interactions.
- There are no guidelines on deploying engagement forecasting models into development pipelines that use Python.

This study aims to develop a unique hybrid forecasting methodology designed for Python software applications by addressing these gaps. Such methodology would enhance planning features, retention of application usage, and improve stakeholder engagement by providing informed decision-making insights.

Problem Statement and Research Questions

Problem Statement

This fast-paced world of software development has made analyzing user interactions with Python applications crucial in maintaining engagement and enhancing the user experience. There's a prediction gap when it comes to guessing future trends based on the detailed logs provided. Using methods like ARIMA offers straightforward statistical forecasting but fails to capture intricate behavioral patterns. More contemporary approaches like LSTM and GNNs bring their own problems by requiring vast amounts of computational resources along with high-caliber datasets which are hardly available in application-focused domain-specific Python tools. These discrepancies concerning model adequacy and data interpretability create unfulfilled needs in proactive forecasting processes for applications tailored to developers.

This study is important because tools and research already available target attention detection algorithms only in the context of mobile or social media engagements, completely neglecting the standalone usage and unusual movements of standalone Python software like data analysis tools, educational platforms, and automation scripts. Patching the gap with the holiday breaks and academic calendars, these tools undergo version changes which triggers their domain-specific engagement cycles. Batch processing workflows lacks context-agnostic features temporal model reasoning. Thus, this study aimed to address the problem by developing a hybrid model of interpretable forecasting tailored to Python applications. Solving this problem enables software developers, product managers, and researchers to strategically manage resources and product value in ways that enhance retention.

Research Questions

RQ1: What is the accuracy of predictions of the classical statistical models (such as ARIMA, SARIMA, AR, MA) to predict activity parameters (such as daily active users) within Python software applications based on the remaining past usage logs?

RQ2: By what margin can complex deep learning models (e.g. LSTM, GRU) outperform simpler models when predicting engagement, particularly in cases of either a small amount of user activity or bursty / irregular activity?

RQ3: What is the success rate of hybrid models (Prophet-LSTM, ARIMA-GRU) in addressing not only seasonal trends but also nonlinear trends in the user engagement?

RQ4: How do the effects of the inclusion of a contextual feature (e.g., feature updates, holidays, library version releases) on the predictive performance of hybrid engagement forecasting models work?

RQ5: What are the effective ways of designing and assessing a Python-native, batch-based engagement forecasting tool, and how do they need to be interpretable, usable and fit the real-world developer workflows?

Research Objectives and Goals

Main Objective is through the usage logs; this study intends to build an accurate prediction model that forecasts user interaction with future Python-based software applications. This model will assist developers prepare for user interactions more strategically.

Specific Objectives

RO1: To explore, evaluate and experiment on some of the conventional time series forecasting models (ARIMA, SARIMA models) on data sets of usage logs in Python Applications.

RO2: At that point tried to compare how a more modern deep learning configuration (LSTM or GRU) could work on the same data, and that data is usually irregular and quite thin.

RO3: To start experimenting and streamlining hybrid adaptations of such models, Prophet-LSTM, ARIMA-GRU, combine statistical and machine-learning concepts to up accuracy levels.

RO4: Then, to incorporate some more low-hanging, specific to the topic features such as software version updates, academic calendars, or (public) holidays to understand to what degree these will be of any significance.

RO5: Lastly, composed a basic forecasting app that allowed developers to reuse their logs and receive not only predictions of relations but also certain visualizations. This was done entirely with FastAPI, TensorFlow, Prophet, and a few other Python packages.

These outcomes will provide the necessary resources and guidance needed from the research by Python developers facing challenges of engaging users in their software.

Research Objectives and Goals cont.....

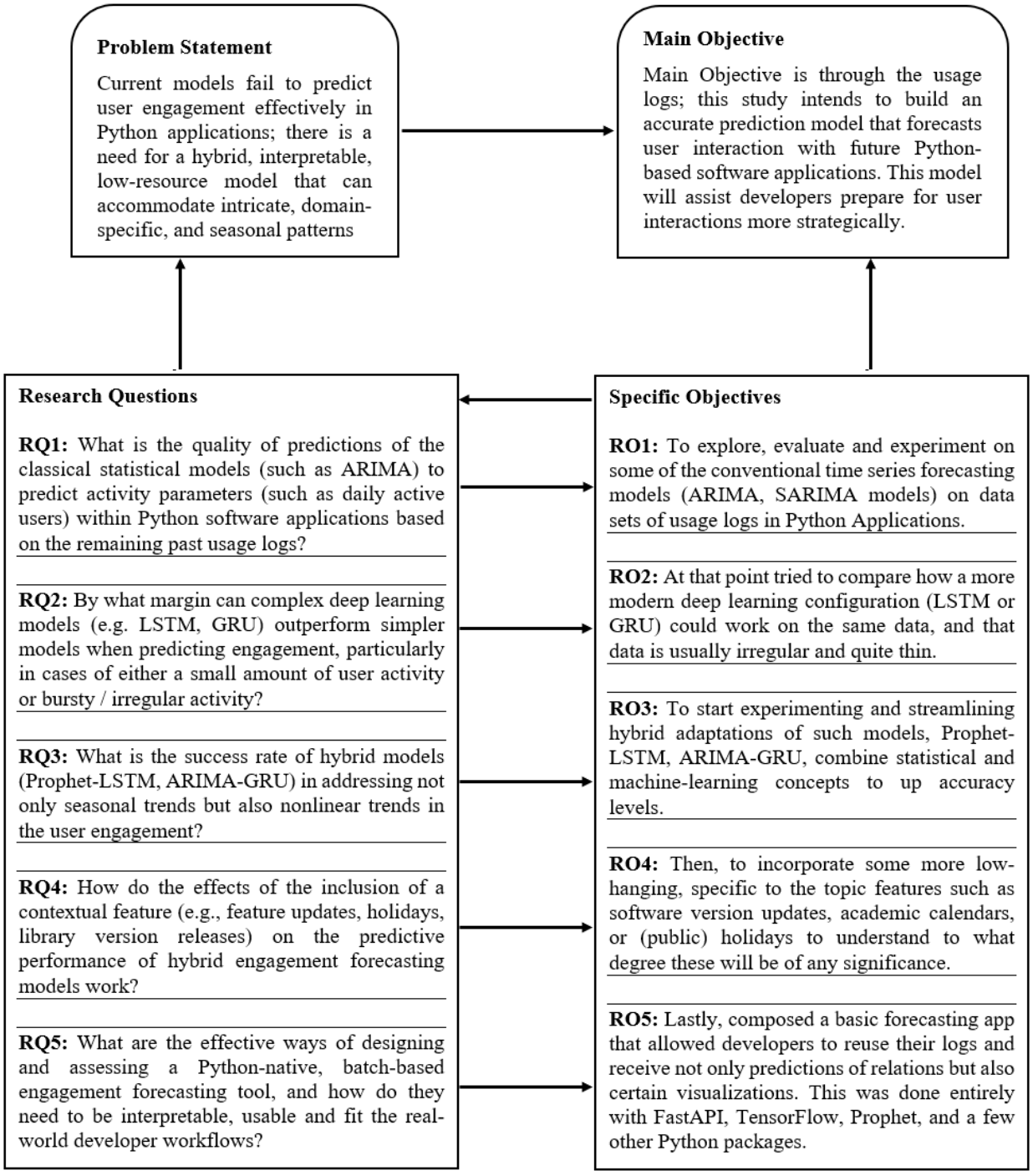

Figure 1 - Research Objectives and Goals

Proposed Research Method

This study will adopt a step-by-step approach with each iteration built around the requirements of predicting user interaction within software applications developed in Python. It focuses on the processing of domain-specific data as well as the creation of algorithms using the tools provided by the language Python which reflects as model building hybrid constructs, contextual modeling, and tool implementation. The entire process, or "workflow", as is often the case with software applications developed in Python, is characterized by fragmented, periodic, or burst-like patterns in user activity.

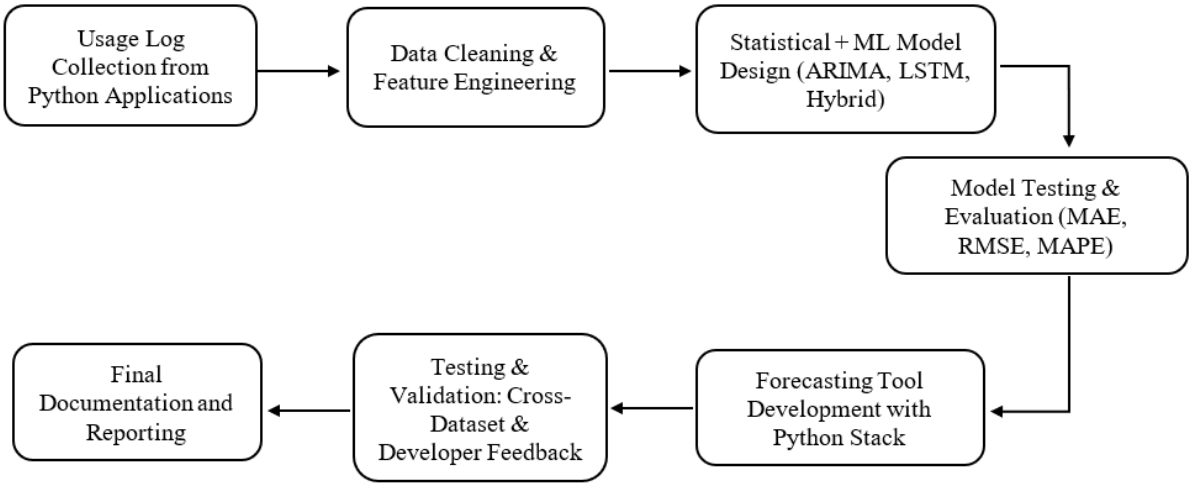

Figure 2 - Methodology diagram

1. Usage Log Collection from Python Applications

The initial phase pertains to the accumulation or simulation of user interaction data from educational Python-based software tools, open-source utilities, and productivity scripting tools. The logs will encapsulate timestamps, session duration, tracked user actions, and participation metrics at the feature level. In the absence of any authentic logs, datasets will be constructed using Python utilities (Faker, Locust) that portray mock scenarios like surge-based version engagements or academic period activity participation.

2. Data Cleaning & Feature Engineering

The logs will be cleaned and formatted into a time series that is more suitable for predictive analysis. This includes:

- The removal of duplicates, as well as outlier values.
- Arriving at a missing value through interpolation or other imputation methods.
- Aggregates being consolidated into relevant temporal intervals (for instance, daily active users).
- Contextual features such as holidays, release cycles, flag features, etc., are added.

With this technique, a dataset created for a particular domain, tailored to Python software will be assembled. This will specifically focus on capturing distinct trends and seasonality pertaining to the software.

Additional Section: Forecasting and Variable Design

This proposed research will help to forecast user interaction with Python software applications in the future based on organized usage log data. To fulfill this goal, it is necessary to accurately identify the dependent variable (forecasting), which in this case is a question of future user engagement, and the independent variables (supportive), which will be generated out of log data through feature engineering.

Dependent Variable (Forecasting Target)

- **Daily Active Users (DAU)**: Daily Active Users, abbreviated as DAU, is the main metric of forecasting. This indicator captures the sum of unique people using the application in one day, and, therefore, works as an actual engagement measure. It is computed by summing session or action-based logs based on the date of an interaction.

Independent (Supportive) Variables

- The variables specified below are important in determining the time frame as well as the explaining behavioral patterns and seasonal activators in any given forecasting model:

Category	Feature Name	Description
Temporal	day_of_week	A well-established weekday marker (0-6) that would allow one to identify the weekly patterns of the usage of an application.
	month, is_weekend	The information incorporated in this research is across seasons and patterns in behavior that influence users.
Usage Metrics	session_count	The progress of the analysis also follows the amount of sessions that are initiated daily.
	avg_session_duration	The average length of user activity during the day.
	feature_click_counts	The frequency of using specific features (e.g., PDF export) in the course of time is also mentioned in the dataset.
Contextual / External	holiday_flag	A binary flag indicates whether days are calendar or are public holidays.
	release_flag	Another variable is days after the introduction of new versions of software.
	academic_period_flag	The periods of semester or exams in the academic realms are also marked.
	error_count	Lastly, it is the count of user-affecting errors that are stipulated per day.

Note: The list of variables presented above will be viewed as the first baseline for model training. The feature set, however, is expected to change in the development phase depending on the availability of data, analysis of feature importance, or changing knowledge during experimentation. It is possible to add or modify the new features depending on how they affect the accuracy of the forecast.

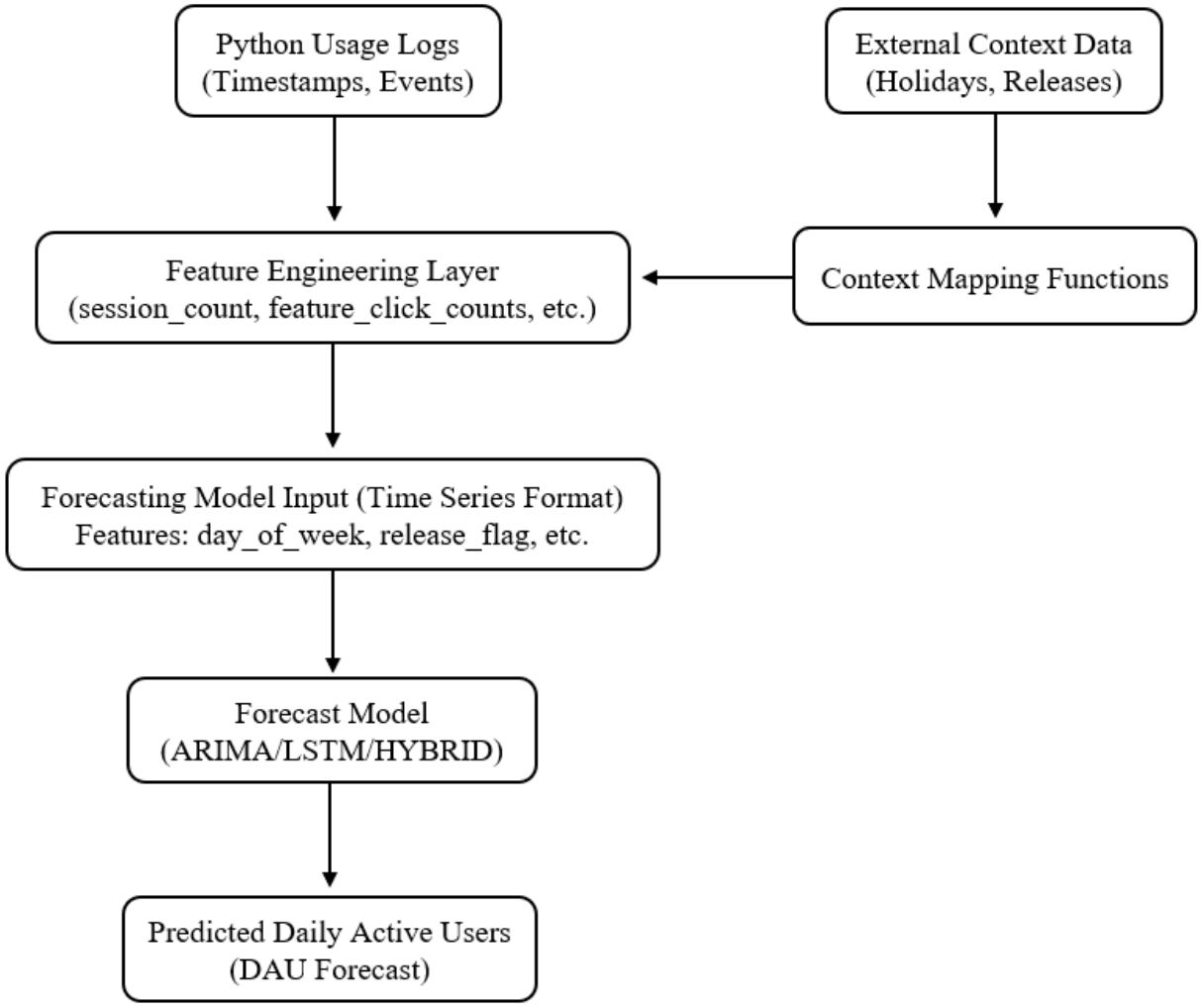

Figure 3 - Variable and Forecasting Structure

The most relevant statistical unit of choice is DAU and thus delivers a lucid operational definition to forecasting. The combination of time and contextual features improves universality and flexibility of the forecast, so that the output will better match actual user behaviors conducted on Python programs. The variables at once serve statistical and machine learning solutions and tolerate domain-specific peculiarities, such as semester-based spikes and surge in activity in the aftermath of update.

3. Statistical + ML Model Design (ARIMA, LSTM, Hybrid)

These are the three models we will develop and evaluate:

01. ARIMA, SASRIMA Type Models: The statistical baseline will include models that employ linear trends and seasonal periodic variations.

02. Deep Learning models: will perform with data characterized by long term dependencies and nonlinearity through LSTM/GRU networks.

03. Hybrid models: will include machine learning models for irregularities while using statistical models regarding underlying trends. Example: seasonality residuals. Example: Prophet-LSTM and ARIMA-GRU.

All models will optimally be set on the historical engagement data which show the usage dynamics around activity spikes post release, update feature releases, or other activity periods.

4. Model Testing & Evaluation (MAE, RMSE, MAPE)

The models will be tested and evaluated using forecast performance metrics such as:

- Mean Absolute Error (MAE).
- Root Mean Squared Error (RMSE).
- Mean Absolute Percentage Error (MAPE).

In addition, models will be tested across different use cases, such as sparse logs and bursty activity, to examine their robustness and practical suitability for Python application forecasting.

5. Forecasting Tool Development with Python Stack

A prototype forecasting tool will be developed using:

- FastAPI for a lightweight REST API interface.
- TensorFlow or PyTorch for the backend hybrid model deployment.
- Prophet for accounting trends and seasonal effects.
- Matplotlib and Seaborn for plotting.

This tool will permit developers to upload usage logs and process them in batch mode, thus generating forecasted engagement reports along with visual charts for decision support.

6. Testing & Validation: Cross-Dataset & Developer Feedback

The system will be validated through:

- Cross-validation on multiple time window applications.
- Exposing it to public or open-source usage datasets, like GitHub event logs.
- Qualitative evaluation from Python developers to analyze usability, ease of interpreting the forecast, and relevance to planning activities.

This ensures the model generalizes well and that the tool holds value in practical development scenarios.

7. Final Documentation and Reporting

We will furnish a detailed account of all the processes which include model configuration, data preparation, evaluation results, and tool usage. To aid exposition and reproducibility, visual summaries like flowcharts, model architecture diagrams, and examples of forecast outputs will also be included.

Expected Outcome

This study aims to develop a practical and efficient forecasting model that predicts user engagement for Python-based applications using logs from their previous engagements with the software. The primary results of the study are the following,

Hybrid Model of Forecasting

A well-documented model that integrates statistical methods (ARIMA or Prophet) along with machine learning techniques (LSTM or GRU) to optimize the user engagement metrics accuracy and efficiency.

Improved Accuracy on Predictions

Achieved better forecasting accuracy than other models in use and demonstrated adaptive finesse toward complex and sporadic usage patterns typical of Python tools and libraries.

A Tool Built with Python

A prototype or tool developed in the Python ecosystem (Prophet, TensorFlow, FastAPI) that processes usage logs, generates future engagement forecasts, and visualizes the results for effective decision-making.

SABARAGAMUWA UNIVERSITY OF SRI LANKA
FACULTY OF COMPUTING
DEPARTMENT OF SOFTWARE ENGINEERING
BSc HONOURS DEGREE PROGRAMME IN SOFTWARE ENGINEERING
SE 8101 Research Project in Software Engineering

Predictive Engagement Modeling

Contextual exogenous and endogenous features like holidays, release cycles and feature updates are utilized by an engagement prediction model which communicates their prospective worth towards model performance.

Research Contributions

Evaluating custom forecasting models provided in the form of Python software developed for benchmarking purposes alongside tailored software solutions, detailing the insights and foundational elements for prospective research within the scope.

Guidance for Developers

Comprehensive guides and tutorials aimed at assisting Python developers and software teams leverage time series forecasting to analyze and enhance user engagement metrics within their applications.

Research Plan and Timetable

Suggested Time Frame		Activity
Semester VII	1st Week -3rd Week	Finding a supervisor, preliminary discussions with the supervisor and literature survey; submitting the project outline; planning and starting the project
	4th Week	Submission of Proposal (project outline)
	4th Week	Submission of Proposal defense presentation slides and self-recorded video.
	5th Week	Proposal defending presentation at a seminar
Semester VIII	7th Week	Mid-term review
	10th Week	Completion of project work, preparation and submission of the draft thesis to the supervisor (The supervisors are expected to return the corrected project thesis within 1-2 week)
	14th Week	Preparation and submission of the final draft (consent form) of the project thesis and corrected Abstract.
	14th Week	Submission of Final defense presentation slides and self-recorded video.
	15th Week	Final Defense

Proposed Budget

SABARAGAMUWA UNIVERSITY OF SRI LANKA
FACULTY OF COMPUTING
DEPARTMENT OF SOFTWARE ENGINEERING
BSc HONOURS DEGREE PROGRAMME IN SOFTWARE ENGINEERING
SE 8101 Research Project in Software Engineering

References (Minimum 40 Related References to the Topic) (IEEE format)

[1] J. K.-G. Lead, "User Engagement Metrics - The Complete Guide," in *UXCam blog* vol. 2025, UXCam, Ed., ed: UXCam, 2025, pp. User Engagement Metrics - The Complete Guide.

[2] S. G.-C. P. Manager, "Top 12 SaaS User Engagement Metrics & How To Measure Them," in *Userpilot blog* vol. 2024, ed: Userpilot, 2024, p. SaaS User Engagement Metrics & How To Measure Them

[3] Contentsquare. *Top 8 user engagement metrics to track and measure*, Hotjar Ltd, Novakid, 2022. [Online]. Available: https://www.hotjar.com/user-engagement/metrics/.

[4] Umbrex. "Usage and Engagement Metrics." Umbrex - Designed by Filez. https://umbrex.com/resources/industry-analyses/how-to-analyze-a-saas-company/usage-and-engagement-metrics/ (accessed since 2015, 2015).

[5] A. Chia, "User Engagement Metrics To Know," in *User Engagement Metrics To Know* vol. 2024, S. Blogs, Ed., ed. Boston, Massachusetts: Splunk LLC, 2024, p. What are user engagement metrics.

[6] A. M. MARCUS, Andreas Drangel, "Predicting user churn using temporal information," *Early detection of churning users with machine learning using log-level data from a MedTech application,* December 25, 2022 2022. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:1834191/FULLTEXT01.pdf. Stockholm.

[7] E. Zhivaikina, "Time Series Forecasting: A Complete Guide," in *DATA VISUALIZATION 101* vol. 2025, I. Preset, Ed., ed. 548 Market St, PMB 51897
San Francisco, CA 94104-5401: Preset, Inc., 2023, pp. In the era of social media, digital shopping and IoT, time series are literally everywhere. Though it's not as flashy as other skills, time series analysis is a true data science black-belt super power because it can both open a window to the past, or help us see into the future!

[8] Y. Liu, X. Shi, L. Pierce, and X. Ren, "Characterizing and forecasting user engagement with in-app action graph: A case study of snapchat," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp. 2023-2031.

[9] I. E. Team, "11 Important User Engagement Metrics for You To Track," in *Career development* vol. 2025, I. E. Team, Ed., ed: Indeed 2025.

[10] O. w. bookdown. "Chapter 8 ARIMA models." OTextsTM with bookdown. https://otexts.com/fpp2/arima.html (accessed 2025, 2025).

[11] P. Ltd, "The Value of LSTM in Time Series Forecasting," in *PredictHQ* vol. 2025, ed: PredictHQ Ltd, 2025.

[12] A. Hayes, "Autoregressive Integrated Moving Average (ARIMA) Prediction Model," *ARIMA*, July 31, 2024. [Online]. Available: https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp

[13] J. Noble. "What are ARIMA models?" IBM. https://www.ibm.com/think/topics/arima-model (accessed 24 May 2024.

[14] R. Nau, *Lecture notes on forecasting - Introduction to ARIMA models*, F. S. o. Business, ed., Duke University: Duke University, 2014, p. 21. [Online]. Available: https://people.duke.edu/~rnau/Slides_on_ARIMA_models--Robert_Nau.pdf. Accessed on: 2014.

[15] M. B. S. K, "Python Time Series Forecasting: A Practical Approach," *Time series forecasting*, November 2. [Online]. Available: https://wandb.ai/madhana/Time_Series/reports/Python-Time-Series-Forecasting-A-Practical-Approach--VmlldzoyODk4NjUz

[16] J. Brownlee, "Time Series Forecasting With Prophet in Python," in *Time Series* vol. 2020, M. L. Mastery, Ed., ed: Guiding Tech Media, 2020, p. Time series forecasting can be challenging as there are many different methods you could use and many different hyperparameters for each method.

[17] K. Balboni, "45 best user engagement tools—and how to choose the right ones for your product," in *Growth*, Appcues, Ed., ed: Appcues, -.

[18] O. Kolisnykov, "Time Series Analysis and Forecasting: Examples, Approaches, and Tools," in *article - blog*, O. Kolisnykov, Ed., ed. AltexSoft AltexSoft 2022, pp. Oleksandr is a content strategist and editor. He leads (when possible) the team of independent-thinking writers and tech journalists at AltexSoft. With over 10 years of writing and editing tech-related pieces and scripts, he currently focuses on travel tech, data science, and AI. Outside of work, Oleksandr enjoys escapism in video games and game development.

[19] K. Moumane, I. E. Asri, I. Rharoubi, H. A. Abderrahman, and S. Faqihi, "Enhancing banking governance: A machine learning-based credit risk classification," *Journal of Autonomous Intelligence,* 2024.

[20] K. Y. Hern, L. K. Yin, and C. W. Yoke, "Forecasting facebook user engagement using hybrid prophet and long short-term memory model," in *International Conference on Digital Transformation and Applications (ICDXA)*, 2021, vol. 25, p. 26.

[21] S. Makridakis, S. Wheelwright, and R. Hyndman, "Forecasting: Methods and Applications," vol. 35, 1984.

[22] G. Tunnicliffe Wilson, "Time Series Analysis: Forecasting and Control,5th Edition, by George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel and Greta M. Ljung, 2015. Published by John Wiley and Sons Inc., Hoboken, New Jersey, pp. 712. ISBN: 978-1-118-67502-1," *Journal of Time Series Analysis,* vol. 37, pp. n/a-n/a, 03/01 2016, doi: 10.1111/jtsa.12194.

[23] W. McKinney, "Data Structures for Statistical Computing in Python," *Proceedings of the 9th Python in Science Conference,* 01/01 2010.

[24] N. Bora, "Understanding ARIMA Models for Machine Learning," in *Capital One* vol. 2021, N. Bora, Ed., ed: Capital One, 2021, p. A simple introduction to understanding autoregressive integrated moving averages.

[25] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

[26] J. Brownlee, "How to Make Out-of-Sample Forecasts with ARIMA in Python," *Time series forecasting*. [Online]. Available: https://machinelearningmastery.com/make-sample-forecasts-arima-python/

[27] K. W. Ng, F. Mubang, L. O. Hall, J. Skvoretz, and A. Iamnitchi, "Experimental evaluation of baselines for forecasting social media timeseries," *EPJ Data Science,* vol. 12, no. 1, p. 8, 2023.

[28] S. Seabold and J. Perktold, "Statsmodels: Econometric and Statistical Modeling with Python," *Proceedings of the 9th Python in Science Conference,* vol. 2010, 01/01 2010.

[29] A. Zemkoho, "A basic time series forecasting course with python," in *Operations Research Forum*, 2022, vol. 4, no. 1: Springer, p. 2.

[30] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing,* vol. 50, pp. 159-175, 2003/01/01/ 2003, doi: https://doi.org/10.1016/S0925-2312(01)00702-0.

[31] X. Liu and W. Wang, "Deep Time Series Forecasting Models: A Comprehensive Survey," *Mathematics,* 2024.

[32] Z. Chen, M. Ma, T. Li, H. Wang, and C. Li, "Long sequence time-series forecasting with deep learning: A survey," *Inf. Fusion,* vol. 97, p. 101819, 2023.

[33] M. Jin *et al.*, "A Survey on Graph Neural Networks for Time Series: Forecasting, Classification, Imputation, and Anomaly Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 46, no. 12, pp. 10466-10485, 2024, doi: 10.1109/TPAMI.2024.3443141.

[34] J. A. Miller *et al.*, "A Survey of Deep Learning and Foundation Models for Time Series Forecasting," p. arXiv:2401.13912doi: 10.48550/arXiv.2401.13912.

[35] J. Torres, D. Hadjout, A. Sebaa, F. Martínez-Álvarez, and A. Troncoso, "Deep Learning for Time Series Forecasting: A Survey," *Big Data,* vol. 9, 12/03 2020, doi: 10.1089/big.2020.0159.

[36] J. Kim, H. Kim, H. Kim, D. Lee, and S. Yoon, "A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges," *Artificial Intelligence Review,* vol. 58, no. 7, p. 216, 2025/04/23 2025, doi: 10.1007/s10462-025-11223-9.

[37] Y. H. Kong, K. Y. Lim, and W. Y. Chin, "Time Series Forecasting Using a Hybrid Prophet and Long Short-Term Memory Model," in *Soft Computing in Data Science*, Singapore, A. Mohamed, B. W. Yap, J. M. Zain, and M. W. Berry, Eds., 2021// 2021: Springer Singapore, pp. 183-196.

[38] S.-C. Necula, I. Hauer, D. Fotache, and L. Hurbean, "Advanced Hybrid Models for Air Pollution Forecasting: Combining SARIMA and BiLSTM Architectures," *Electronics,* vol. 14, no. 3, p. 549, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/14/3/549.

[39] R. Stromer, O. Triebe, C. Zanocco, and R. Rajagopal, "Designing forecasting software for forecast users: Empowering non-experts to create and understand their own forecasts," *ArXiv,* vol. abs/2404.14575, 2024.

[40] X. Tang, Y. Liu, N. Shah, X. Shi, P. Mitra, and S. Wang, "Knowing your fate: Friendship, action and temporal explanations for user engagement prediction on social apps," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 2269-2279.

[41] F. Li *et al.*, "DIGMN: Dynamic Intent Guided Meta Network for Differentiated User Engagement Forecasting in Online Professional Social Platforms," presented at the Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, Singapore, Singapore, 2023. [Online]. Available: https://doi.org/10.1145/3539597.3570420.

[42] J. van Acker, L. Maenhout, and S. Compernolle, "Older Adults' User Engagement With Mobile Health: A Systematic Review of Qualitative and Mixed-Methods Studies," (in eng), *Innov Aging,* vol. 7, no. 2, p. igad007, 2023, doi: 10.1093/geroni/igad007.

[43] J. Pywell, S. Vijaykumar, A. Dodd, and L. Coventry, "Barriers to older adults' uptake of mobile-based mental health interventions," *DIGITAL HEALTH,* vol. 6, p. 2055207620905422, 2020, doi: 10.1177/2055207620905422.

[44] J. Wang, "Mobile and Connected Health Technologies for Older Adults Aging in Place," *Journal of Gerontological Nursing,* vol. 44, no. 6, pp. 3-5, 2018, doi: doi:10.3928/00989134-20180509-01.

[45] H. Gnanasekaran, D. P, and U. Köse, "Time-series Forecasting of Web Traffic Using Prophet Machine Learning Model," vol. 1, pp. 161-177, 11/29 2023.

[46] F. O. Source. "Prophet Python API." Facebook. https://facebook.github.io/prophet/docs/quick_start.html (accessed 2025).

[47] D. Seca, "TimeGym: Debugging for Time Series Modeling in Python," *arXiv preprint arXiv:2105.01404,* 2021.

[48] B. Bollig, M. Függer, and T. Nowak, *A Framework for Streaming Event-Log Prediction in Business Processes*. 2024.

[49] P. Morreale, J. McAllister, S. Mishra, and T. Dowluri, "Turning leaf: Eco-visualization for mobile user engagement," *Procedia Computer Science,* vol. 52, pp. 690-694, 2015.

[50] A. Y. Doroshenko, Y. O. Haidukevych, V. Haidukevych, and O. S. Zhyrenkov, "User-centric technology stack for weather and air pollution forecasting," *PROBLEMS IN PROGRAMMING,* 2024.

Any Other Relevant Information
