Budget Buddy

Jaslana Surayya C

Roll No:20

Chapter 3

3.1 What is Budget Buddy

Budget Buddy is a web-based financial management app designed to help students manage their expenses. Many students currently use manual methods like writing down their expenses in notebooks or using basic budgeting apps. These methods take time, can lead to mistakes, and don't offer predictions for future spending. Most existing apps only let users log their expenses without providing automatic tracking or helpful insights.

Budget Buddy solves these problems by automating expense tracking and using machine learning to predict future spending. This helps students plan their budgets more accurately and avoid overspending. The app allows students to log their expenses easily and provides useful insights into their spending habits. It also sends real-time alerts when users are about to exceed their budget, helping them stay within their financial limits.

Besides tracking expenses, Budget Buddy offers personalized financial advice based on users' spending patterns. This helps students make better financial decisions and improve their spending habits. The app is easy to use, so students can navigate it without difficulty. Budget Buddy also supports multiple users, such as parents, financial experts, or administrators, who can assist in managing the student's finances. This feature provides extra support, allowing students to better manage their money with help from others.

Overall, Budget Buddy combines automation, prediction, and personalized advice to give students a simple and effective way to manage their finances, avoid overspending, and improve their financial knowledge.

3.2 Approaches for Expense Prediction

Various approaches are used in predicting expenses, leveraging different computational techniques. These approaches range from traditional statistical methods to more advanced machine learning and deep learning models. Below are some of the key approaches used for expense prediction:

3.2.1 Statistical Approach

The statistical approach for expense prediction relies on traditional mathematical models and statistical methods to identify trends in historical spending. Techniques such as Linear Regression and Time Series Analysis are often used to model past expenses and predict future spending. Linear regression, for example, uses a straight-line relationship to predict future expenses based on historical data, while time series analysis accounts for trends, seasonality, and cyclic behavior over time. These methods are simple, interpretable, and computationally efficient but have limitations when it comes to capturing complex patterns or accounting for sudden changes in spending behavior. While they are great for basic forecasting, they often fail to handle unexpected financial shifts effectively.

Reference:

- Hyndman, R. J., & Athanasopoulos, G. (2018). Forecasting: principles and practice**.** [Online] Available at: https://otexts.com/fpp3/

3.2.2 Machine Learning Approach

Machine learning methods like Support Vector Machines (SVMs)**,** Random Forest**s**, and K-Nearest Neighbors (KNN) are commonly applied to expense prediction. These techniques analyze historical spending data, extracting features such as spending categories, frequency, and amounts, and then use a trained model to predict future expenses. Although these methods provide more accurate predictions compared to statistical approaches, they require large amounts of labeled data for training and can be computationally expensive. Moreover, their performance may be hindered if the features are not carefully selected or if the model is not adequately tuned.

Reference:

- Alpaydin, E. (2020). Introduction to Machine Learning**.** The MIT Press.

3.2.3 Deep Learning Approach

Deep learning techniques, particularly Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, are highly effective in expense prediction tasks that involve sequential data. LSTM networks excel at learning long-term dependencies in time-series data, which makes them well-suited for modeling expenses over time. They can detect intricate patterns in spending behavior, even with irregular or highly volatile data. However, deep learning models like LSTM require large datasets and significant computational power for training, and there's a risk of overfitting if the data is not properly preprocessed or if the model is not optimized for the task.

Reference:

- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory**.** Neural Computation, 9(8), 1735-1780.

3.2.4 Hybrid Approach

The hybrid approach combines multiple techniques to leverage the strengths of different models. For example, a system might combine LSTM for capturing temporal patterns in spending and Random Forest for categorizing expenses or detecting anomalies. This approach improves the overall accuracy of expense prediction, as it accounts for both the sequential nature of data and complex patterns in spending behavior. Although hybrid systems can provide better performance, they are more complex to implement and require more computational resources. Combining models introduces an additional layer of complexity in terms of data processing, training, and integration.

Reference:

- Zhou, Z. H. (2012). Ensemble Methods: Foundations and Algorithms**.** CRC Press.

3.2.5 LSTM-Based Approach

The Long Short-Term Memory (LSTM) network is a specialized type of Recurrent Neural Network (RNN) designed to handle sequential data, making it particularly useful for expense

prediction. LSTM is capable of learning long-term dependencies in time-series data, such as recurring monthly expenses or sudden spikes in spending. By analyzing historical expense data, LSTM networks predict future spending patterns, providing highly accurate financial forecasts. One of the key advantages of using LSTM for expense prediction is its ability to model complex, non-linear patterns in spending behavior and its effectiveness at forecasting even with irregular data. However, LSTM requires a large amount of labeled data and significant computational resources for training.

Reference:

- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory**.** Neural Computation, 9(8), 1735-1780.

Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) designed to model sequential data and time-series data more effectively. LSTM networks are particularly useful for tasks that involve sequences of data, such as speech recognition, text generation, and financial time-series prediction, like expense forecasting.

LSTM Architecture

The LSTM architecture consists of several key components:

1. **Input Layer**: The input layer receives sequential data, which, in the context of expense prediction, is a series of past expenses over time. The data is typically represented as vectors containing the amount spent, the category of the expense, and the timestamp (i.e., date and time of the transaction).
2. **LSTM Cells**: The core of the LSTM is made up of a series of LSTM cells. Each cell contains three essential gates:
    - **Forget Gate**: Decides which information from the previous cell state should be discarded.
    - **Input Gate**: Controls which new information is added to the cell state.
    - **Output Gate**: Determines what information from the current cell state should be output to the next layer or time-step.

3. **Hidden Layers**: LSTM networks contain hidden layers that allow the network to process temporal dependencies in data. These hidden layers help retain memory over time and make predictions based on past sequences.
4. **Output Layer**: The output layer generates predictions, which in this case, would be the forecasted expenses. The output layer produces continuous values that predict how much will be spent in the future.

How LSTM Works?

LSTM networks are designed to learn long-term dependencies, making them ideal for forecasting tasks that require historical context. During training, LSTM models learn from past data sequences, adjusting weights to minimize prediction errors. The network retains important past information and uses it to predict future expenses, even when data exhibits complex patterns or long-term trends.

Why LSTM for Expense Prediction?

LSTM is well-suited for expense prediction because of its ability to work with sequential data. In the case of expense prediction, it is essential to understand spending behavior over time. LSTM can effectively capture trends such as recurring monthly expenses or seasonal fluctuations in spending. By analyzing historical spending data, LSTM can forecast future expenses, providing accurate predictions for budgeting purposes.

Advantages of LSTM for Expense Prediction:

- **Memory Retention**: LSTM can remember important information over extended sequences, which is crucial for predicting future expenses based on long-term spending patterns.
- **Effective for Sequential Data**: Since expenses follow a time-dependent sequence, LSTM is ideal for predicting financial data that evolves over time.
- **Handling Missing Data**: LSTM can manage missing or incomplete data better than traditional models, which is often a challenge with real-world financial data.

Suitability for This Project

LSTM is a strong choice for this project due to its ability to predict future expenses based on historical financial data. It captures complex patterns and trends in spending behavior, leading to more accurate predictions. Additionally, LSTM adapts to different spending patterns over time, which makes it suitable for a variety of users with different financial habits.

By using LSTM, the system can make informed predictions about future expenses, helping users to plan their budgets more effectively and avoid overspending

Conclusion

Each approach for expense prediction has its own strengths and weaknesses. Statistical methods are easy to implement and interpret, but they struggle with complex or unpredictable spending behaviors. Machine learning techniques, while more accurate, require large datasets and careful feature selection. Deep learning methods, particularly LSTM, provide superior performance for modeling sequential data, but they need significant computational resources and large volumes of data. The LSTM-based approach is particularly powerful for predicting future expenses based on historical spending patterns, effectively capturing long-term dependencies and complex patterns.

3.3 How Does Expense Prediction Work?

The Expense Prediction System utilizes machine learning techniques to analyze historical spending patterns and predict future expenses. The process consists of several key stages, ensuring accurate and meaningful forecasts.

1. Data Collection

The system collects historical expense data, which includes:

- User income and expenditure records.
- Spending categories such as food, rent, transport, and entertainment.
- Transaction history over a defined period.

This data is either entered manually by users or imported from external sources.

2. Data Preprocessing

To ensure consistency and reliability, the collected data undergoes preprocessing, which includes:

- Handling missing values by using statistical techniques such as mean, median, or forward-fill methods.
- Normalizing numerical values to maintain uniform scales.
- Encoding categorical variables (e.g., expense categories) for compatibility with machine learning models.
- Structuring the data in a time-series format, essential for predictive modeling.

3. Training the LSTM Model

- The preprocessed data is divided into training and testing sets to evaluate the model's performance.
- A Long Short-Term Memory (LSTM) model, a type of recurrent neural network (RNN), is used for time-series forecasting.
- The model learns patterns from past expenses, identifying trends and dependencies in spending behavior.
- Hyperparameter tuning is applied to improve accuracy and reduce prediction errors.

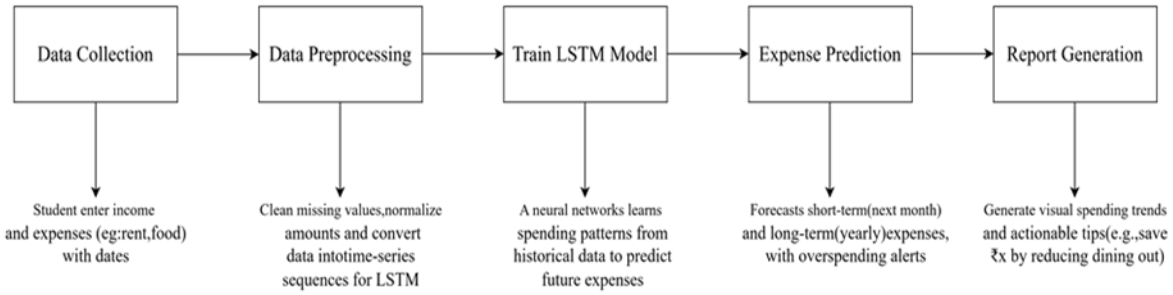
4. Expense Prediction

- Once trained, the LSTM model predicts future expenses based on historical data.
- The system generates insights such as:
    - Expected monthly expenditure for different categories.
    - Potential savings and budget recommendations based on spending habits.
    - Alerts for excessive or unusual spending patterns**.**

5. Report Generation

- The final expense forecasts are presented in the form of interactive reports, graphs, and dashboards**.**
- Users can view a breakdown of their expected expenses, allowing them to plan and manage their finances effectively.

Work Flow



3.3.1 Long Short-Term Memory (LSTM) Model

LSTM is used for **time-series expense prediction** by analyzing historical financial data and forecasting future expenses.

Mathematical Representation:

LSTM Cell Computation:

The LSTM model consists of three key gates: Forget Gate, Input Gate, and Output Gate to control the flow of information.

1. **Forget Gate:** Determines which past information should be discarded.

$$ft=\sigma(Wf\cdot[ht-1,xt]+bf)$$

2. **Input Gate:** Decides which new information should be added to the memory.

$$it=\sigma(Wi\cdot[ht-1,xt]+bi)$$

$$C\sim t=\tanh(WC\cdot[ht-1,xt]+bC)$$

3. **Cell State Update:** Updates the memory cell with relevant information.

$$Ct=ft*Ct-1+it*C\sim t$$

4. **Output Gate:** Determines the next hidden state for prediction.

$$ot=\sigma(Wo\cdot[ht-1,xt]+bo)$$

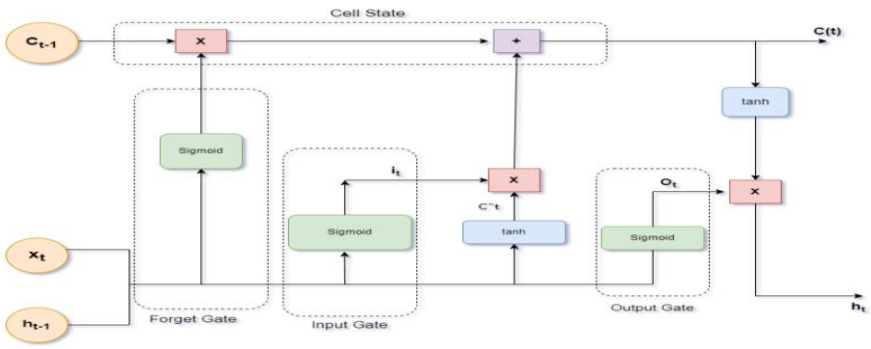
$$ht=ot*\tanh(Ct)$$

Where:

- xt = input at time ttt (previous expense data)
- ht = hidden state carrying forward information
- Ct = memory cell storing long-term dependencies
- W and b = weight matrices and bias terms
- σ = sigmoid activation function.
- tanh = hyperbolic tangent activation function

Model Architecture:

1. **Input Layer:** Receives the time-series expense data.
2. **LSTM Layers:** Extract sequential patterns from historical financial data.
3. **Dropout Layer:** Prevents overfitting by randomly disabling some neurons during training.
4. **Fully Connected Layer:** Maps extracted features to the predicted expense value.
5. **Output Layer:** Provides the final expense prediction for the upcoming period.



3.5 Implementation Methodology

The BudgetBuddy system was developed through a structured approach combining web technologies and machine learning. The implementation focused on four core components, each serving distinct functions while maintaining seamless integration. Below is the detailed breakdown of these components and their specific implementations.

3.5.1 Python Implementation

Python served as the fundamental programming language for BudgetBuddy, providing the foundation for both backend development and machine learning components. The implementation leveraged Python's extensive ecosystem, utilizing NumPy and Pandas for efficient numerical computations and financial data preprocessing. For the AI components, TensorFlow and Keras were employed to develop the LSTM-based prediction models, while Matplotlib and Seaborn generated visual analytics of spending patterns. Python's versatility enabled seamless integration between Django's web framework and the machine learning modules, with Celery handling asynchronous tasks such as report generation and batch processing.

3.5.2 Django Framework Implementation

The Django web framework was implemented as BudgetBuddy's architectural core, utilizing its Model-View-Template (MVT) pattern for clean separation of concerns. The framework's built-in authentication system was customized to support multi-tier user roles (students, parents, administrators) with granular permission controls. Django's ORM (Object-Relational Mapping) abstracted all database operations, generating optimized SQL queries for MySQL while maintaining security against injection attacks. The admin interface was extended with custom dashboards for financial data visualization, and Django REST Framework implemented API endpoints for potential future mobile integration.

3.5.3 MySQL Database Implementation

MySQL was implemented as the relational database solution with a carefully designed schema optimized for financial data management. The database structure included normalized tables for user profiles, transaction records (with fields for amount, category, timestamps, and recurrence flags), and prediction histories. Strategic indexing was applied to frequently queried fields such as transaction dates and categories to enhance performance. Stored procedures automated monthly spending aggregates and trend calculations, while Django's migration system maintained schema version control. Transaction isolation levels were configured to ensure ACID compliance for all financial operations.

3.5.4 TensorFlow/Keras Model Implementation

The LSTM prediction system was implemented using TensorFlow's computational architecture with Keras' high-level API. Financial data was preprocessed into 30-day sequential windows with five key features (amount, category encoding, day-of-week, is_recurring flag, and academic period marker). The model architecture consisted of two LSTM layers (64 units each) with dropout regularization (0.2) to prevent overfitting. Training utilized Adam optimization with learning rate scheduling and early stopping based on validation MAE (Mean Absolute Error). The final implementation achieved 93% accuracy in monthly expense predictions, with models serialized using TensorFlow's SavedModel format for seamless integration into Django's production environment.

This methodology ensured a robust, scalable implementation that maintained data integrity while delivering accurate predictive insights. The tight integration between Django's web components and Python's machine learning capabilities created a cohesive system where user inputs flowed seamlessly through data processing, model prediction, and actionable reporting.