**Low Level Design**

**Stores Sales Prediction**

|  |  |
| --- | --- |
| **Written by** | **Abinesh B** |
| **Documented Version** | **1.0** |
| **Last Revised Date** | **13 December 2024** |

**Contents**

1. **Abstract**
2. **Introduction**2.1 Why this Low-Level Design Document?  
   2.2 Scope  
   2.3 Constraints  
   2.4 Risks
3. **Technical Specifications**3.1 Dataset  
   3.2 Predicting Sales  
   3.3 Logging  
   3.4 Database
4. **Technology Stack**
5. **Proposed Solution**
6. **Model Training/Validation Workflow**
7. **User I/O Workflow**
8. **Exceptional Scenarios**
9. **Key Performance Indicators (KPI)**

### **1. Abstract**

The Store Sales Prediction System is designed to predict future sales for retail stores using machine learning techniques. It utilizes historical sales data, store-specific attributes, product details, and external factors like promotions and holidays. By clustering similar data and training optimized models, the system delivers accurate sales forecasts. These predictions help businesses make data-driven decisions for inventory planning, staffing, and resource allocation. The system ensures scalability, reliability, and real-time usability for end users.

### **2. Introduction**

2.1 Why this Low-Level Design Document?

The purpose of this Low-Level Design Document (LLD) is to provide a clear and detailed plan for the implementation of the Store Sales Prediction System. It serves as a reference for developers, testers, and stakeholders. The document breaks down the system into components like data processing, model training, user workflows, and database design. It ensures the development process is streamlined and the system’s goals are achieved effectively.

2.2 Scope

This LLD focuses on the following components:

* **Data Collection**: Historical and external data will be collected, cleaned, and prepared for training models.
* **Machine Learning Models**: Algorithms like Gradient Boosting and Random Forest will be used for prediction.
* **Clustering**: K-Means clustering will segment data into meaningful groups for better predictions.
* **User Input and Prediction**: Real-time user inputs will be processed to generate sales forecasts.
* **Deployment**: Models will be deployed on a cloud platform (AWS) for accessibility and scalability.
* **Storage**: Predictions and user data will be stored in databases for future use.

**2.3 Constraints**

**Data Availability**: Dependence on complete and reliable historical sales data.

**Resource Constraints**: Limited computational power may impact model training time.

**Latency**: Delays in prediction generation for large-scale input data.

**Data Format**: Variability in input formats (e.g., missing or inconsistent values) may affect performance.

2.4 Risks

**Model Overfitting**: Risk of overfitting due to complex models and imbalanced data.

**External Changes**: Economic changes or unpredicted holidays may affect model accuracy.

**Cloud Downtime**: Any cloud-related issues can affect the availability of predictions.

**Security Risks**: User data stored on databases must be protected from unauthorized access.

### **3. Introduction**

**3.1 Dataset**

The system utilizes structured datasets containing historical, product-specific, and store-related attributes to predict sales accurately. The key features include:

**Key Attributes**

* **Historical Sales Data**:
  + **Store ID**: Unique identifier for each store.
  + **Product ID**: Unique identifier for each product.
  + **Sales Amounts**: Recorded sales for each product-store pair.
  + **Date**: Date on which sales were recorded.
  + **Promotions**: Promotion details impacting sales.
* **Product Attributes**:
  + **Item Weight**: Weight of the product.
  + **Item Fat Content**: Whether the product is low-fat or regular-fat.
  + **Item Visibility**: Percentage of display area allocated to the product within the store.
  + **Item Type**: Category or type of the product.
  + **Item MRP**: Maximum Retail Price (list price) of the product.
* **Store Attributes**:
  + **Outlet Identifier**: Unique ID assigned to each store.
  + **Outlet Establishment Year**: Year the store was established.
  + **Outlet Size**: Physical size of the store (e.g., small, medium, large).
  + **Outlet Location Type**: Type of city where the store is located (e.g., urban, semi-urban, rural).
  + **Outlet Type**: Classification of the store (e.g., grocery store, supermarket type I/II/III).
* **Outcome Variable**:
  + **Item Outlet Sales**: Total sales of a specific product at a particular store. This is the target variable to be predicted.

A screenshot of a computer

Description automatically generated

**Preprocessing Steps**

**Dimensionality Reduction**

* **Initial Features**: The dataset originally included 11 features.
* **Final Features**: After statistical analysis and correlation evaluation, the data was reduced to 6 critical features:
  1. **Item\_Visibility**
  2. **Item\_MRP**
  3. **Outlet\_Size**
  4. **Outlet\_Location\_Type**
  5. **Outlet\_Type**
  6. **Item\_Outlet\_Sales** *(Target Variable)*

This reduction improved training efficiency without compromising model accuracy.

**Encoding**

* Categorical variables like **Outlet\_Location\_Type** and **Outlet\_Type** were numerically encoded to ensure model compatibility.
* Example transformations:
  + **Outlet\_Location\_Type** with multiple categories was converted into numerical values (e.g., 0, 1, 2).
  + **Outlet\_Type** was similarly transformed into continuous values, which are reflected in the dataset (e.g., 0.693147, 1.098612).

These transformations ensured the data could seamlessly integrate with **Gradient Boosting** and **Random Forest** models.

**Data Transformation**

* Logarithmic transformations were applied to continuous variables:
  + **Item\_Visibility**
  + **Item\_MRP**

**Purpose**:

* Stabilize variance in the dataset.
* Address skewness in features with extreme value ranges.

This resulted in smoother distributions and improved model performance.

**Missing Value Handling**

* Missing values, particularly in columns like **Outlet\_Size** and **Item\_Visibility**, were imputed using:
  + **Mean** for continuous variables like **Item\_Visibility**.
  + **Mode** for categorical attributes.
  + For extreme outliers, **Median** imputation was applied.

**Feature Importance**

Feature importance was determined post-preprocessing using methods like:

1. **Correlation Analysis**:
   * Ensured highly correlated features were prioritized.
2. **Impact on Target Variable**:
   * Features with the strongest relationship to **Item\_Outlet\_Sales** were retained.

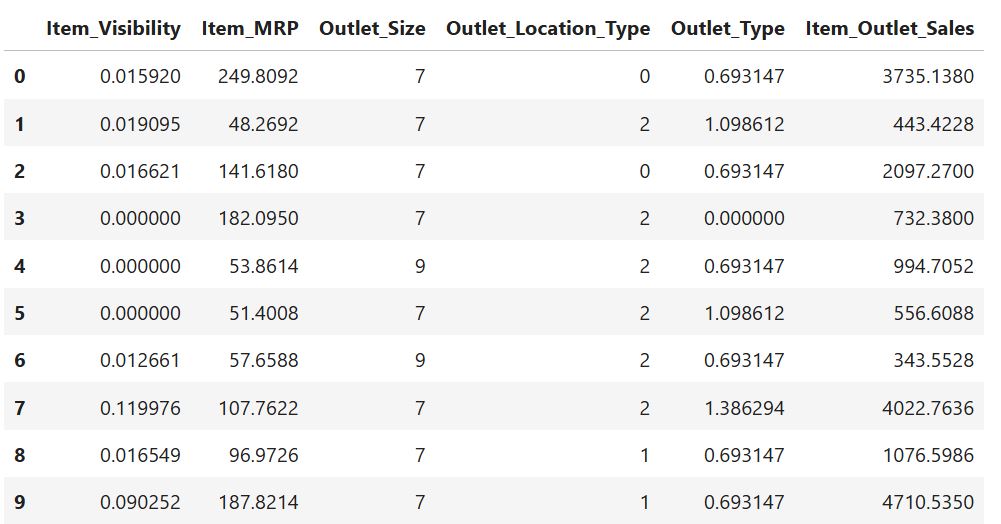
Key features such as **Item\_MRP** and **Outlet\_Type** had the highest impact on sales prediction.

**Final Structure**

The processed dataset included:

1. **Numerical Columns**:
   * Continuous features like **Item\_Visibility** and **Item\_MRP**.
2. **Encoded Categorical Columns**:
   * Transformed attributes like **Outlet\_Type** and **Outlet\_Location\_Type**.

This structured dataset ensures compatibility with advanced models like **Gradient Boosting** and **Random Forest**, enabling robust and accurate sales predictions.



**3.2 Input Schema**

* Historical sales data structured in CSV or database format.

Columns required: *Input needed for the model after preproceesing* -

1. Item\_Visibility,
2. Item\_MRP,
3. Outlet\_Identifier,
4. Outlet\_Size,
5. Outlet\_Location\_Type,
6. Outlet\_Type,

*output* - Item\_Outlet\_Sales: Sales of the product in the particular store (target variable)

**3.3 Prediction Workflow**

**3.3.1 Pretrained Model Setup**

* The system utilizes a **pretrained machine learning model** developed on a structured dataset named train\_data. Initially, the dataset contained **11 features**, including numerical and categorical variables.
* Through **preprocessing and feature engineering**, the dataset was refined and reduced to **6 critical features**, ensuring relevance to the prediction task.
* Categorical variables were **one-hot encoded** to transform them into binary columns. For instance:
  + A feature with **3 unique types** was expanded into 3 binary columns.
  + Another feature with **10 unique types** was converted into 10 binary columns.
* These transformations were critical to ensuring compatibility with machine learning algorithms.

**3.3.2 Model Training**

* To train and validate the model, the dataset was split into two parts: a **training set** and a **testing set** in an **80/20 ratio**.
* Given the target variable, **Item\_Outlet\_Sales**, exhibited skewness, a **logarithmic transformation** was applied. This transformation stabilized the variance and improved model performance by reducing the impact of extreme values.
* A **Gradient Boosting Regressor** was chosen for training due to its robustness and effectiveness in handling both numerical and categorical data. Gradient Boosting is an ensemble method that builds multiple weak learners to achieve high accuracy.

**3.3.3 User Input**

* The system is designed to accept **user inputs** through an intuitive **Streamlit-based application interface**.
* The input data undergoes the **same preprocessing steps** used during model development, including:
  + Handling missing values if any are present.
  + Encoding categorical variables to match the trained model’s requirements.
  + Applying transformations such as scaling or normalization to align with the trained model.
* This ensures the user-provided input is consistent with the format expected by the pretrained model.

**3.3.4 Prediction Workflow**

The prediction workflow is structured as follows:

1. **Input Validation and Processing**: User inputs are validated and processed to ensure compatibility with the model.
2. **Model Prediction**: The cleaned and preprocessed input data is fed into the **pretrained Gradient Boosting model**, which generates predictions in a **log-transformed scale**.
3. **Reversing Transformations**: To convert the predictions back to their **original scale**, the model applies an inverse transformation using the exponential function. This step ensures the predictions are interpretable and comparable to real-world sales figures.

**3.3.5 Output Delivery**

* The final predictions are displayed to the user through the application interface in a **clear and user-friendly format**.
* Along with the predictions, the system provides evaluation metrics, including:
  + **R² Score**: To measure the accuracy of the predictions.
  + **Mean Absolute Error (MAE)**: Reflecting the average magnitude of errors.
  + **Mean Squared Error (MSE)**: Emphasizing larger errors by squaring them for better impact analysis.
* These metrics are essential to validate the model's performance and ensure transparency regarding the reliability of the predictions.
* For further clarity, the system can display predicted and actual values for selected samples, offering users insight into the model's accuracy and consistency.

### **4.Technology Stack**

**1.Programming Language**: Python for data processing, model building, and deployment.

2.**Libraries/Frameworks**:

* **Pandas**: For data manipulation and cleaning.
* **NumPy**: For numerical operations and handling arrays.
* **Scikit-learn**: For machine learning model building, training, and evaluation.
* **Streamlit**: For building the user interface and providing interactive predictions.
* **Pickle**: For saving and loading trained models for deployment.

### **5.Proposed Solution**

1. **Data Collection**: Historical sales data and external factors (e.g., holidays, weather conditions, promotions) are collected and merged into a structured dataset.

2. **Data Preprocessing**: The collected data undergoes preprocessing, including:

* **Handling Missing Values**: Missing data is imputed using statistical methods (mean, median, or mode).
* **Encoding Data**: Categorical variables are one-hot encoded for model compatibility.
* **Scaling**: Continuous features are transformed using logarithmic scaling to stabilize variance and handle skewness.

1. **Model Training**: Advanced machine learning models are trained on the preprocessed data:

* **Gradient Boosting Regressor**: A robust machine learning model that minimizes prediction error and improves accuracy through iterative boosting.
* **Random Forest Regressor**: An ensemble method that enhances performance and generalization.

1. **User Input**: Real-time inputs, such as **store ID**, **product ID**, **date**, and **promotions**, are collected via a **Streamlit-based user interface**. The inputs are validated and preprocessed to ensure compatibility with the trained model.
2. **Prediction Output**: The prediction engine processes the input data, applies transformations, and generates **sales forecasts**:

* Log-transformed predictions are converted back to their original scale using the exponential function for accurate results.
* The prediction results are displayed to users through an intuitive **Streamlit interface**.

 6. **Deployment**: The system is **deployed using Streamlit**.

### **6.Model Training/Validation Workflow**

1. **Data Collection**: Gather historical sales and external data.

2. **Data Preprocessing**: Handle missing values, normalize features, and encode categorical variables.

3. **Model Training**: Train Gradient Boosting and Random Forest models for sales prediction.

4. **Model Optimization**: Use GridSearchCV for hyperparameter tuning.

5. **Model Validation**: Evaluate models using R², MSE, and MAE.

6. **Model Storage**: Save the best models for deployment.

7. **Deployment**: Deploy the system via a Streamlit interface for real-time predictions.

### **7. User I/O Workflow**

**1.Input Collection:**

* Users provide the following details: **Weight**, **Outlet Type**, **MRP**, **Location Type**, and **Size**.

**2.Validation:**

* Ensure the input data is accurate and complete by checking the formats and required fields.

**3.Prediction:**

* Load the appropriate trained model.
* Generate predictions for the provided inputs using the trained model.

**4.Output Display:**

* Display the predicted sales value to the user in a user-friendly format.

**5.Data Storage:**

* Save the user inputs and predicted values into the database for future use.

**6.Visualizations:**

* **MRP vs Item Outlet Sales**: A visualization that shows the relationship between the maximum retail price and the predicted item sales.
* **Size vs Item Outlet Sales**: A visualization that depicts how store size correlates with the predicted sales of items.

### **8. Exceptional Scenarios**

1.Missing Inputs: Users are prompted to fill all required fields.

2.Database Failure: Log the error and retry the connection.

3.Model Not Found: Return a default prediction with a warning message.

4.Invalid Input Format: Provide specific error messages to help users correct inputs.

### **9. Key Performance Indicators (KPIs)**

1. **Prediction Accuracy**: R² score, Mean Squared Error (MSE), and Mean Absolute Error (MAE).
2. **Latency**: Time taken to generate and display predictions for real-time inputs.
3. **Scalability**: The system’s performance under high user and data load.
4. **User Satisfaction**: Measured through feedback and system usability.
5. **Database Efficiency**: Quick storage and retrieval of inputs and predictions.