**Architecture**

**Stores Sales Prediction**

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

**Table of Contents**

1. **Architecture**
2. **Architecture Description**  
   2.1. **Data Description**  
   2.2. **Data Collection**  
   2.3. **Data Transformation**  
   2.4. **Data Pre-processing**  
   2.5. **Model Building**  
   2.6. **Data from User**  
   2.7. **Data Validation**  
   2.8. **Model Call for Prediction**  
   2.9. **Deployment**

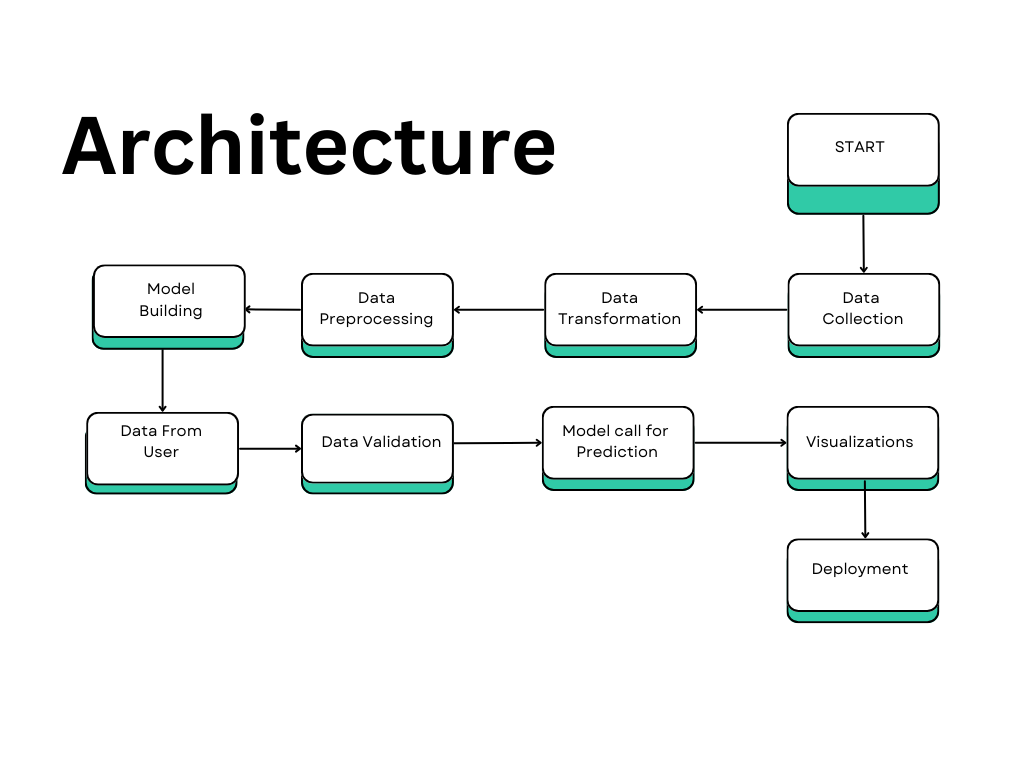
**1. Architecture**

The architecture of the store sales prediction system is designed to handle the process from data collection to deployment efficiently. It incorporates machine learning models (Random Forest and Gradient Boosting) to predict sales, and leverages various techniques such as data transformation, validation, and deployment via Streamlit. The goal is to predict sales based on input features like product attributes, store size, and promotions.

**2. Architecture Description**

The system architecture consists of several components working together:

* **Data Input Layer**: The user provides input, such as store ID, product ID, date, and promotions.
* **Data Preprocessing Layer**: This includes validation and transformations to ensure data quality.
* **Modeling Layer**: This includes the use of trained machine learning models (Random Forest and Gradient Boosting) to make sales predictions.
* **Output Layer**: The predictions are shown to the user along with visualizations for further analysis.
* **Deployment Layer**: The system is deployed on a cloud platform via Streamlit to allow users to interact with it.



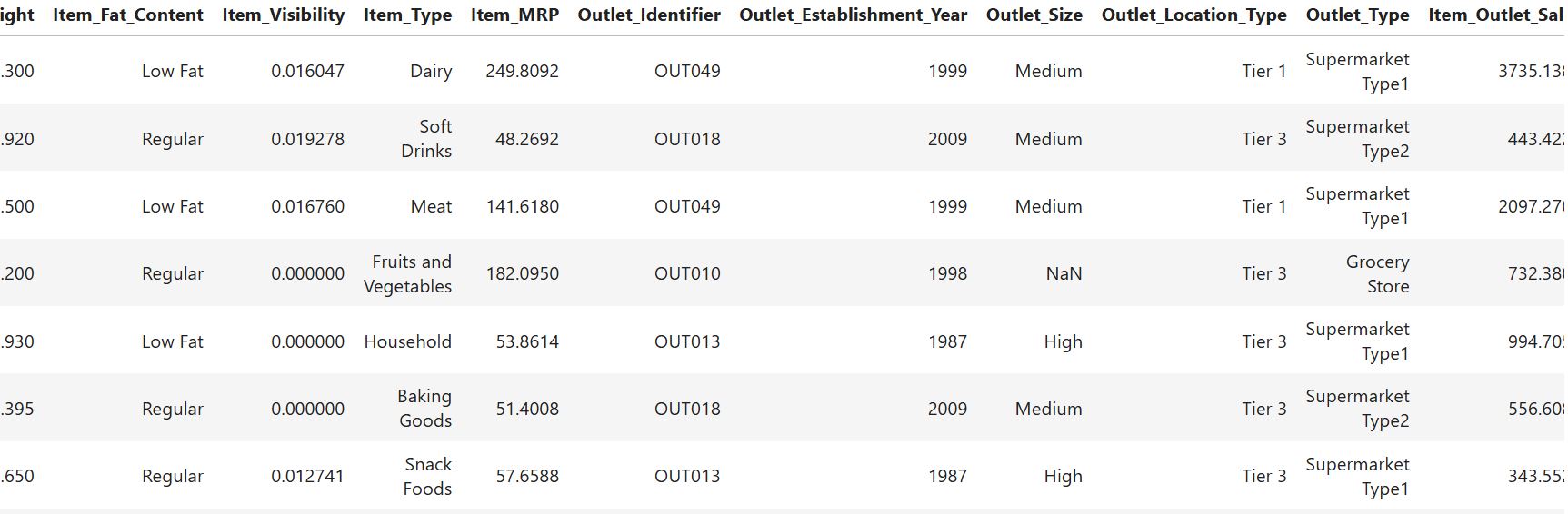
**2.1 Data Description**

The dataset used for training the model includes historical sales data and store-specific information. Key features in the dataset include:

* **Item Weight**: The weight of the product.
* **Item MRP (Maximum Retail Price)**: The list price of the product.
* **Outlet Size**: The size of the store.
* **Outlet Type**: The type of store (e.g., grocery store, supermarket).
* **Item Outlet Sales**: The dependent variable representing the sales of the product at the given store.

**2.2 Data Collection**

The data is collected from multiple sources, including:

* **Historical Sales Data**: This includes past sales data for various stores and products, collected in CSV or other suitable formats.
* **External Factors**: External data, such as promotions, holidays, or weather conditions, that could affect sales are also collected.
* **Store Attributes**: Information about the store, such as its size, type, and location, is gathered.
* 

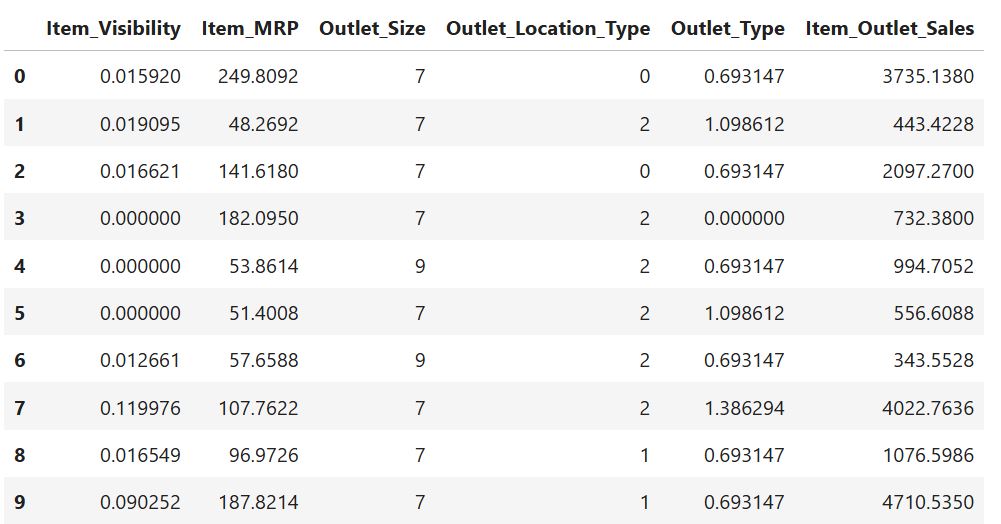
**2.3 Data Transformation**

Data transformation is performed to prepare the dataset for modeling:

* **Feature Encoding**: Categorical features such as store type and product category are one-hot encoded to allow them to be used by machine learning models.
* **Log Transformation**: Continuous features like Item MRP and Item Visibility are transformed using logarithms to handle skewness in the data.
* **Normalization**: Numerical features may be normalized or scaled to ensure uniformity and enhance model performance.

**2.4 Data Pre-processing**

Before training the model, pre-processing steps are conducted:

* **Missing Data Handling**: Missing values are imputed based on the type of variable (mean for continuous, mode for categorical).
* **Outlier Detection**: Outliers are identified and handled to prevent skewing the model’s predictions.
* **Data Splitting**: The dataset is split into training and testing sets to evaluate the model’s performance.
* 

**2.5 Model Building**

The prediction models are built using:

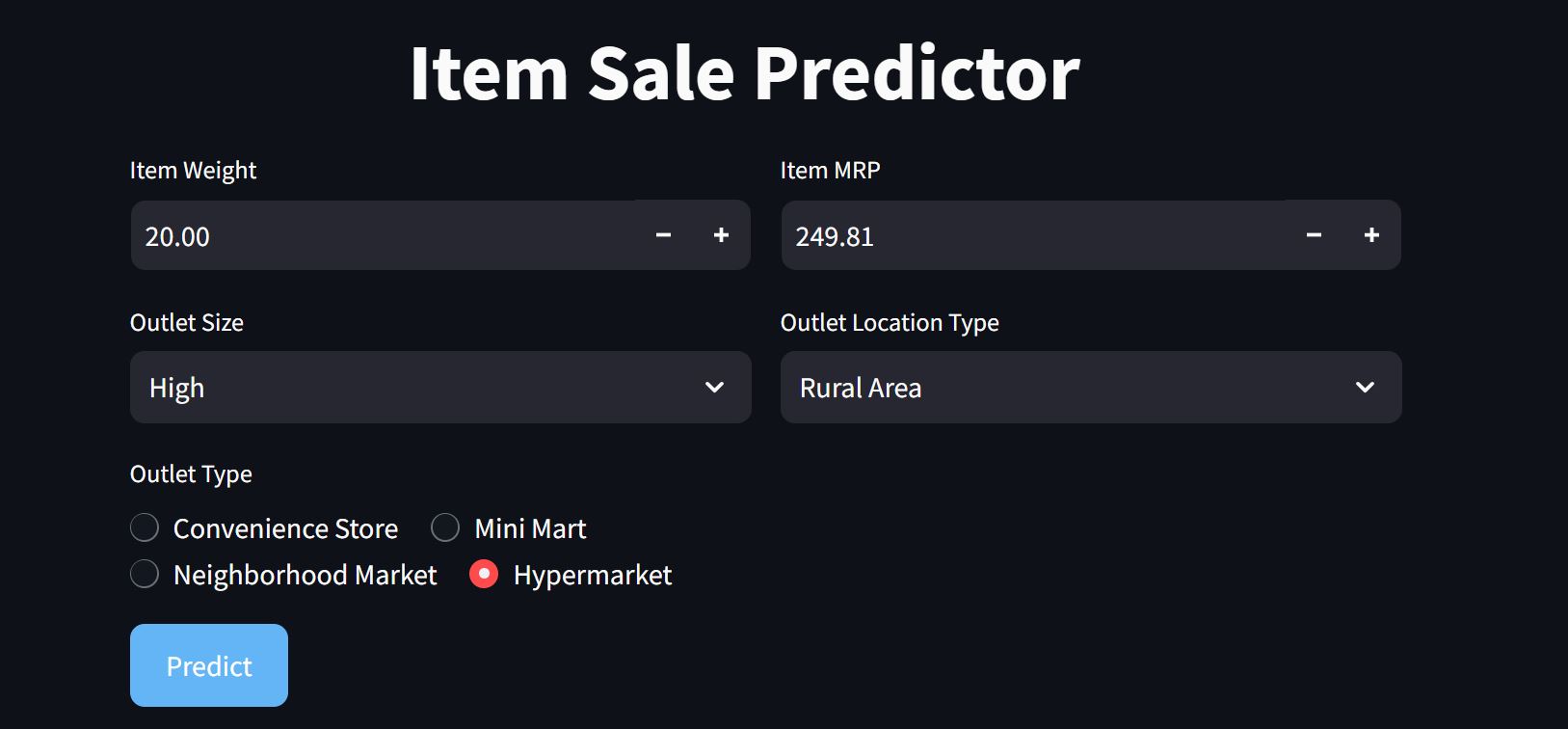
* **Random Forest**: An ensemble learning method that combines multiple decision trees to improve predictive accuracy. It is robust and handles large datasets well.
* **Gradient Boosting**: A boosting technique that builds trees sequentially, where each tree corrects the errors made by the previous one. These models are trained using the processed training data to predict Item Outlet Sales.

**2.6 Data from User**

The user interacts with the system by providing the following data:

* **Item Weight**: Weight of the product.
* **Item MRP**: The maximum retail price of the product.
* **Outlet Size**: Size of the store.
* **Outlet Type**: Type of the store (grocery store, supermarket, etc.).
* **Location Type**: Type of city or locality where the store is located. This input is entered into the web interface developed using Streamlit.

Here's a detailed explanation of each topic in the **Store Sales Prediction** architecture document:



**2.7 Data Validation**

Data validation ensures the integrity and consistency of the user input:

* **Format Checking**: Ensures all inputs are in the correct format (e.g., numerical inputs for weight and price, text inputs for store type).
* **Range Checking**: Validates that input values lie within expected ranges (e.g., the weight must be positive, price cannot be negative).
* **Missing Data**: Any missing or incomplete input is flagged, and the user is prompted to fill in all required fields.

**2.8 Model Call for Prediction**

Once the user input is validated, the pre-trained models (Random Forest or Gradient Boosting) are called to make predictions:

* The system loads the appropriate model based on the user’s store type or other input variables.
* The model processes the input and predicts the **Item Outlet Sales** based on the features provided.

2.9 Deployment

The system is deployed on a cloud platform using Streamlit. This makes the application accessible via a web interface, where users can interact with it in real-time. The application can be deployed on platforms such as AWS or Heroku to ensure scalability and availability.

**2.10 Workflow Description**

1. **User Input: The user interacts with the Streamlit app, providing the necessary inputs such as store ID, product ID, MRP (Maximum Retail Price), outlet size, and other required details via an input form.**
2. **Data Transfer to Pickle File: Once the user submits the input, the data is sent to a Pickle file. This file contains the pre-trained machine learning model, which was built using historical sales data and saved for deployment.**
3. **Model Loading and Prediction Process:**
   * **The Pickle file loads the trained model (such as Gradient Boosting or Random Forest).**
   * **The model processes the input values, applying necessary transformations (like log transformation) to handle skewness in the data.**
   * **Based on the input values, the model makes a sales prediction using the trained model.**
4. **Output Display:**
   * **After the model generates the prediction, the result (predicted sales value) is sent back to the Streamlit app.**
   * **The predicted value is displayed to the user on the Streamlit interface in a clear, user-friendly format.**
5. **Visualization:**
   * **Along with the prediction, the Streamlit app generates real-time visualizations, such as:**
     + **MRP vs Item Outlet Sales graph**
     + **Outlet Size vs Average Item Sales graph**
   * **These visualizations help users better understand the relationship between the features (e.g., MRP, outlet size) and the predicted sales, providing deeper insights into the data.**
6. **Final Display: The user can view both the prediction and the visualizations, helping them make data-driven decisions based on the output from the model.**

**Final View:**

