Data Science Project in Python on BigMart Sales Prediction Project Overview

Overview

Sales forecasting enables businesses to allocate resources for future growth while managing cash flow properly. Sales forecasting also assists firms in precisely estimating their expenditures and revenue, allowing them to predict their short- and long-term success. Retail Sales Forecasting also assists retailers in meeting customer expectations by better understanding consumer purchasing trends. This results in more efficient use of shelf and display space within the retail establishment and optimal use of inventory space.
The Bigmart sales forecast project can help you comprehend project creation in a professional atmosphere. This project entails extracting and processing data in the Amazon Redshift database before further processing and building various machine-learning models for sales prediction.
We will study several data processing techniques, exploratory data analysis, and categorical correlation with Chi-squared, Cramer's v tests, and ANOVA. In addition to basic statistical models like Linear Regression, we will learn how to design cutting-edge machine-learning models like Gradient Boosting and Generalized Additive Models. We will investigate splines and multivariate adaptive regression splines (MARS), as well as ensemble techniques like model stacking and model blending, and evaluate these models for the best results.

Aim

This data science project aims to build and evaluate different predictive models and determine the sales of each product at a particular store. This analysis will help BigMart understand the properties of products and stores, which are crucial in increasing sales and developing better business strategies.

Data Description

The BigMart sales prediction dataset contains 2013's annual sales records for 1559 products across ten stores in different cities. Such vast data can reveal insights about apparent customer preferences as a specific product and store attributes have been defined in the dataset.

- item_identifier: unique identification number for particular items
- item_weight: weight of the items
- item_fat_content: fat content in the item such as low fat and regular fat

- item_visibility: visibility of the product in the outlet
- item_type: category of the product such as Dairy, Soft Drink, Household, etcs
- item_mrp: Maximum retail price of the product
- outlet_identifier: unique identification number for particular outlets
- outlet_establishment_year: the year in which the outlet was established
- outlet_size: the size of the outlet, such as small, medium, and high
- outlet_location_type: location type in which the outlet is located, such as Tier 1, 2 and 3
- outlet_type: type of the outlet such as grocery store or supermarket
- item_outlet_sales: overall sales of the product in the outlet

Tech Stack

- ➔ Language: Python
- ➔ Libraries: Pandas, NumPy, matplotlib, sklearn, redshift connector, Pyearth, Pygam

Approach

- Data Exploration with Amazon Redshift
- Data Cleaning and Imputation
- Exploratory Data Analysis
    - Categorical Data
    - Continuous Data
    - Correlation
        - Pearson’s Correlation
        - Chi-squared Test and Contingency Tables
        - Cramer’s V Test
        - One way ANOVA
- Feature Engineering
    - Outlet Age
    - Label Encoding for Categorical Variables
- Data Split
- Model Building and Evaluation
    - Linear Regressor
    - Elastic Net Regressor
    - Random Forest Regressor
    - Extra Trees Regressor
    - Gradient Boosting Regressor
    - MLP Regressor

- Multivariate Adaptive Regression Splines (MARS)
- Spline Regressor
- Generalized Additive Models - LinearGAM, PoissonGAM, GammaGAM
- Voting Regressor
- Stacking Regressor
- Model Blending

Modular code overview:

```
data

lib
|_bigmart_sales_prediction.ipynb

ml_pipeline
|_processing.py
|_train.py
|_test.py
|_utils.py

engine.py

requirements.txt

readme.md
```

Once you unzip the modular_code.zip file, you can find the following folders.

1. data
2. lib
3. ml_pipeline
4. engine.py
5. requirements.txt
6. readme.md

1. The lib folder is a reference folder and contains the original ipython notebook as in the lectures.

2. The ml_pipeline folder contains all the functions put into different python files, which are appropriately named. The engine.py script then calls these python functions to run the steps in one go to train the model and print the results.

3. The requirements.txt file has all the required libraries with respective versions. Kindly install the file using the command **pip install -r requirements.txt**

4. **All the instructions for running the code are present in readme.md file**

Project Takeaways

1. Understanding the sales prediction problem statement
2. Performing data exploration with Amazon Redshift
3. Understanding SQL queries for data preprocessing
4. Data Cleaning and Imputation with SQL
5. Exploratory Data Analysis on Categorical and Continuous Data
6. Understand Correlation Analysis
7. Categorical Correlation with Chi-squared and Cramer's V Tests
8. Correlation between Categorical and Target Variables with ANOVA
9. Label Encoding for Categorical Variables
10. Linear Regression Implementation
11. Elastic Net Implementation
12. Random Forest Implementation
13. Extra Trees Implementation
14. Gradient Boosting Implementation
15. Multi-Layer Perceptron Implementation
16. Splines and Multivariate Adaptive Regression Splines (MARS) Implementation
17. Implement Generalized Additive Models - LinearGAM, PoissonGAM, GammaGAM
18. Understand and Implement Voting Regressor
19. Understand Stacking and Blending Models
20. Implement Stacking Regressor
21. Implement Model Blending from Scratch
22. Evaluate Models with Regression Metric - R-squared