ARCHITECTURE

Store Sales Prediction

Abstract

This document contains the architecture details of the Store Sales Prediction flask application

Manu Vats

Manuvats1990@gmail.com

Contents

Introduction 2

What is Architecture design document? 2

Folder Architecture 3

Model Creation Stages 5

Data Insertion in Cassandra 5

Retrieve data from database 5

One-Hot-Encoding 5

Data Splitting 5

Machine Learning model Creation 6

Data fromUser 6

Prediction of Sales Figures 6

Deployment 6

Introduction

What is Architecture design document?

Any software needs the architectural design to represents the design of software. IEEE defines architectural design as “the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system.” The software that is built for computer-based systems can exhibit one of these many architectures.

Each style will describe a system category that consists of:

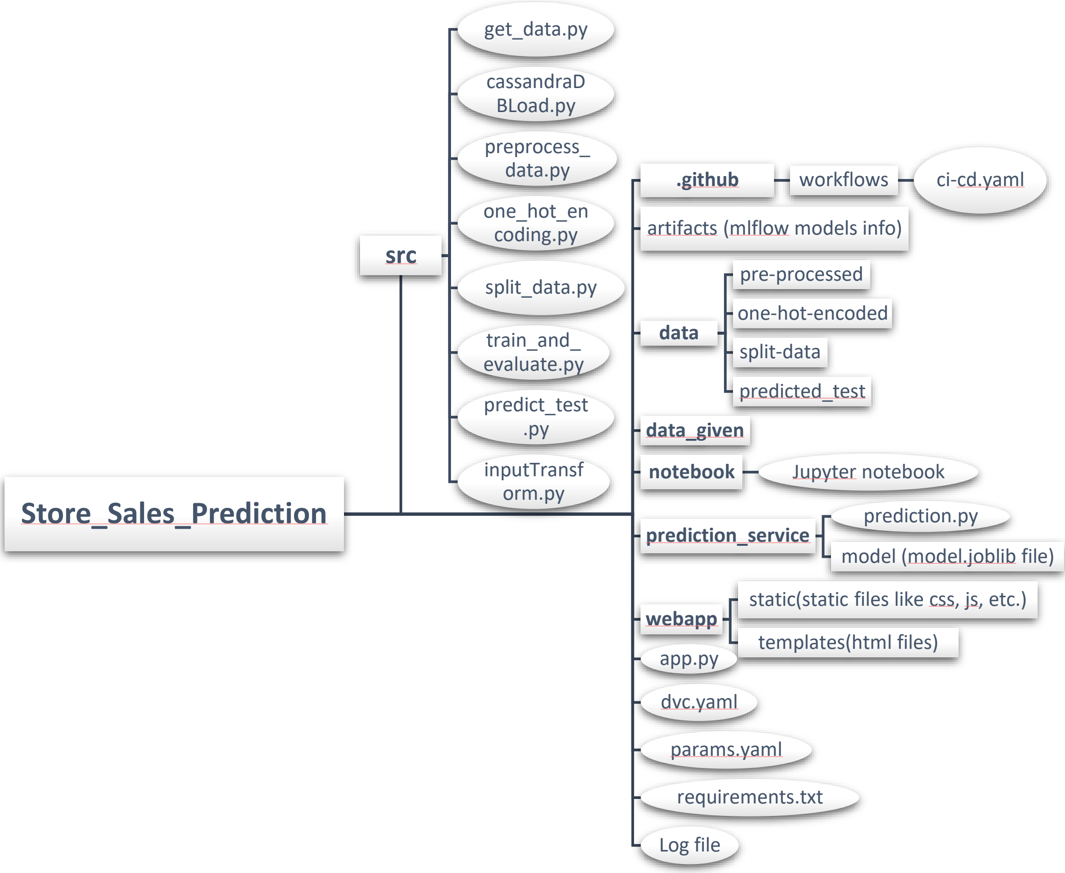
A set of components (e.g.: a database, computational modules) that will perform a function required by the system.
The set of connectors will help in coordination, communication, and cooperation between the components.
Conditions that how components can be integrated to form the system.

Semantic models that help the designer to understand the overall properties of the system.

Scope

Architecture Design Document (ADD) is an architecture design process that follows a step-by-step refinement process. The process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the design principles may be defined during requirement analysis and then refined during architectural design work.

Folder Architecture



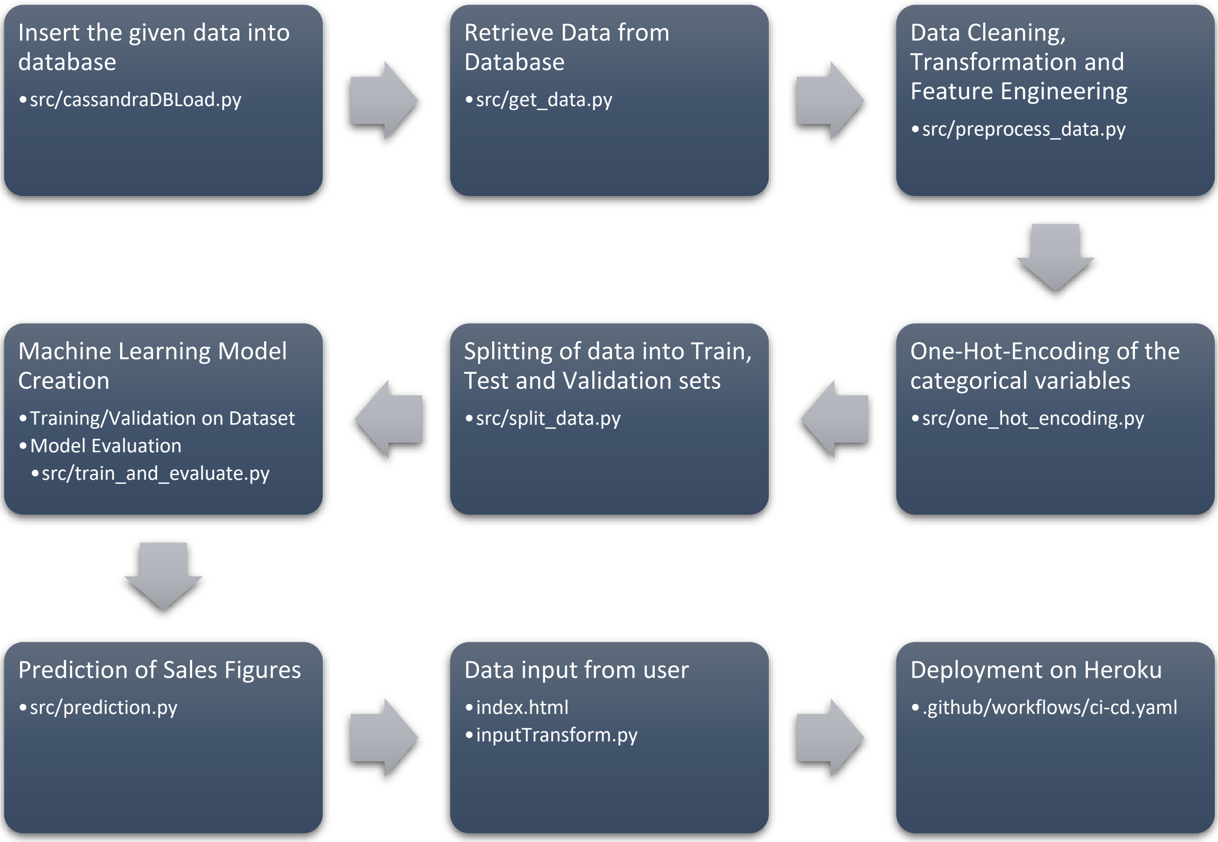
The above figure is the folder structure of the entire code. This structure can be explained in the following points-:

- **src folder** – This folder contains all the python code which is necessary to perform the following steps-:
    1. *cassandraDBLoad.py* - Insertion of raw data in Cassandra
    2. *preprocess_data.py* - Fetching of data from Cassandra and performing data cleaning, transformation and data engineering on the data.
    3. *one_hot_encoding.py* - Transforming the categorical variables into dummy variables. This is done as the model doesn't process on categorical columns. We have to convert them into numerical columns.
    4. *split_data.py* - Splitting the train dataset into Train and Validation data set.
    5. *predict_test.py* – Prediction of the outcome for test dataset
    6. *inputTransform.py* – Transformation of the input data entered by the user according to the Train dataset so that the input data takes the structure of the train data and the trained model is able to perform predictions on it.
- **data_given –** This folder contains the raw dataset

- **data** – This folder contains the data obtained from each stage of the machine learning process namely *data pre-processing, one hot encoding, splitting data* and *training and evaluation*.
- **notebook** – This folder contains the Jupyter Notebook in which all the EDA, data pre-processing and testing of different models was done.
- **prediction_service** – This folder contains following files which are necessary to make the predictions on the test data and the input data.
    1. *prediction.py* – Code to make prediction
    2. *model.joblib* – The model which was trained on the train data
- **.github/workflows** – This folder has the *ci-cd.yaml* file which is responsible for continuous integration and continuous deployment of the application using Github actions.
- **artifacts** – This folder stores the model information generated by the **mlflow** library.
- **app.py** – This is the main file which is used to run the flask application
- **params.yaml** – This file contains all the configuration details of the application. The details include the model parameters, dataset storage locations, etc.
- **dvc.yaml** – This file is the configuration file for DVC (Data Version Control) functionality and it contains the details of various stages of ML model training and deployment. DVC helps us to track the different dataset versions which are used in model creation and deployment.
- **requirements.txt** – This file contains the list of all the libraries/packages which are needed for this application
- **Log file** – This file records and contains the log information of the application.
- **webapp** – This directory stores the css templates and html files needed for the frontend.

Model Creation Stages

The architecture of this application can be depicted in the following way. Each stage shows the major code files which are imperative to carry out that stage-:



Insert the given data into database
- src/cassandraDBLoad.py

Retrieve Data from Database
- src/get_data.py

Data Cleaning, Transformation and Feature Engineering
- src/preprocess_data.py

One-Hot-Encoding of the categorical variables
- src/one_hot_encoding.py

Splitting of data into Train, Test and Validation sets
- src/split_data.py

Machine Learning Model Creation
- Training/Validation on Dataset
- Model Evaluation
  - src/train_and_evaluate.py

Prediction of Sales Figures
- src/prediction.py

Data input from user
- index.html
- inputTransform.py

Deployment on Heroku
- .github/workflows/ci-cd.yaml

Data Insertion in Cassandra

The raw dataset is inserted in Cassandra database. The database used is a free cloud Database provided by DataStax.

Retrieve data from database

The data is fetched from Cassandra to perform data cleaning, transformation and feature engineering.

Data Pre-processing

Feature

One-Hot-Encoding

In this step, all the categorical variables are one-hot-encoded so that our Machine Learning Model can treat them as numerical variables to predict the outcome.

Data Splitting

After pre-processing and one-hot-encoding, data is split into train and test sets to train the model.

Machine Learning model Creation

Different models have been tested in the Jupyter notebook and the best model is used for training the model and prediction.

Data from User

Here we will collect item details data from user such as item weight, type, visibility, MRP and also outlet details.

Prediction of Sales Figures

After Model building, it is fed on the Test dataset and input data to predict the sales figures.

Deployment

We will be deploying the model to Heroku Cloud Platform.