**Architecture Design**

**TRAVEL PACKAGE PURCHASE PREDICTION**

|  |  |
| --- | --- |
| Written by | Aadil Hussain Teeli |
| Version | 1.0 |
| Date | 13-02-2024 |

**Index**

|  |  |
| --- | --- |
| **Content** | **Page No** |
| Abstract | 3 |
| 1. Introduction | 3 |
| 1.1 What is Architecture Design? | 3 |
| 1.2 Scope | 3 |
| 1.3 Constraints | 3 |
| 2. Technical Specification | 4 |
| 2.1 Dataset | 4 |
| 2.2 Logging | 7 |
| 2.3 New Feature Generation | 7 |
| 2.4 Deployment | 7 |
| 3. Technology Stack | 8 |
| 4. Proposed Solution | 8 |
| 5. Architecture Description | 8 |
| 6. User Input/Output Workflow | 11 |

**Abstract**

The "Travel Purchase Package Prediction" project addresses a classification problem at the intersection of travel and data science. Focused on predicting travel package purchases, the system utilizes historical data encompassing customer interactions and purchase behaviors. The primary goal is to classify customers into distinct categories based on their likelihood of purchasing specific travel packages.

Drawing insights from past purchase records and various customer attributes, the system employs a diverse range of machine learning algorithms. These algorithms discern intricate patterns and relationships within the data, contributing to the development of a robust classification model. The model's efficacy lies in its ability to accurately categorize potential buyers, providing businesses with valuable insights to tailor marketing strategies and optimize sales efforts.

In essence, this project revolutionizes customer engagement and revenue generation in the travel industry by empowering businesses to proactively address the diverse needs and preferences of their customer base. The outcome is a sophisticated classification model that not only enhances the precision of predictions but also serves as a strategic tool for businesses seeking to elevate their marketing initiatives and maximize sales in the dynamic landscape of the travel sector.

**1. Introduction**

**1.1 What is Architecture Design?**

The goal of Architecture Design (AD) or a low-level design document is to give the internal design of the actual program code for the `Travel Package Purchase Prediction System`. AD describes the class diagrams with the methods and relation between classes and program specification. It describes the modules so that the programmer can directly code the program from the document.

**1.2 Scope**

Architecture Design(AD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software, architecture, source code, and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work. And the complete workflow.

**1.3 Constraints**

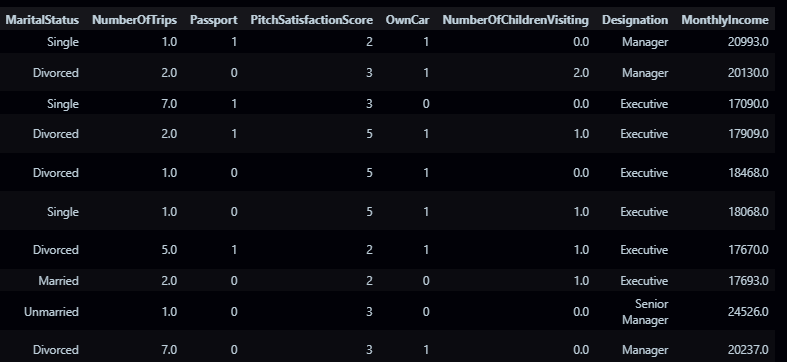
We only predict the expected casual and registered customers based on the weather condition and date information.

**2. Technical Specification**

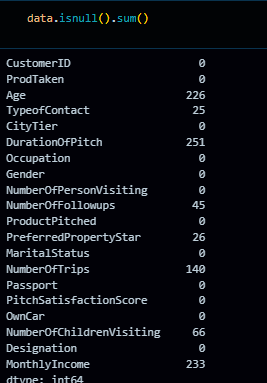
**2.1 Dataset**

The Travel Package Purchase Prediction dataset consists of 4888 observations and includes 20 features. Each observation represents a potential customer, and the features provide details on diverse aspects relevant to travel package purchases. The columns in the dataset are outlined as follows:



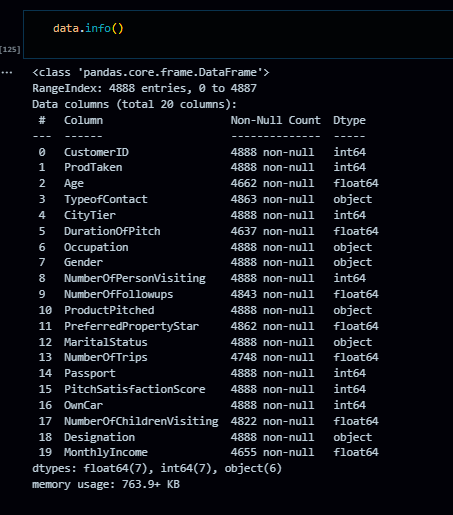


The data Set has various missing values:

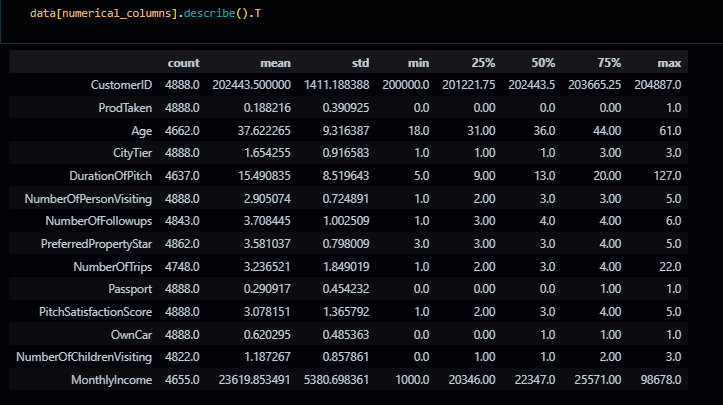


The data set consists of various data types from integer to floating to object as shown in Fig.In the raw data, there can be various types of underlying patterns which also gives an in-depth knowledge about the subject of interest and provides insights into the problem. But caution should be observed

with respect to data as it may contain null values, or redundant values, or various types of ambiguity, which also demands pre-processing of data. The dataset should therefore be explored as much as possible.



Various factors important by statistical means like mean, standard deviation, median, count of values and maximum value, etc. are shown below for numerical attributes.



The preprocessing steps for this dataset involve an in-depth analysis of the independent variables. This includes identifying and addressing null values in each column, replacing them with suitable data types to ensure a seamless flow in analysis and model fitting without compromising accuracy. The visual representations provided above utilize Pandas tools to showcase variable counts for numerical columns and mode values for categorical columns.

Examining numerical columns involves assessing maximum and minimum values, as well as percentile values for the median. This information is pivotal in determining the priority for selecting values for subsequent exploration tasks and analysis. Additionally, understanding the data types of various columns becomes crucial for label processing and implementing a one-hot encoding scheme during the model-building phase.

**2.2 Logging**

We should be able to log every activity done by the user

* The system identifies at which step logging require.
* The system should be able to log each and every system flow.
* Developers can choose logging methods. Also can choose database logging.
* The system should be not be hung even after using so much logging. Logging just because we can easily debug issuing so logging is mandatory to do.

**2.3 Deployment**

For the hosting of the project, we will use AWS ECR



**3. Technology Stack**

|  |  |
| --- | --- |
| Front End | HTML |
| Backend | Flask |
| Deployment | AWS |

**4. Proposed Solution**

Our proposed solution initiates with leveraging exploratory data analysis (EDA) to uncover significant relationships between various attributes and the likelihood of customers purchasing travel packages. Using advanced machine learning algorithms, including fine-tuning through hyperparameter adjustment, we aim to construct a predictive model capable of anticipating future sales demand. Clients will engage with a user-friendly web application, inputting relevant features, which will then undergo validation, preprocessing, and backend prediction generation. The deployment of this integrated system is designed to furnish clients with actionable insights, enabling them to refine marketing strategies, allocate resources efficiently, and maximize revenue generation amidst the dynamic landscape of the travel industry.

**5** **Architecture detail:**

Start

Data gathering

Data Cleaning

Handling Missing Data

Parameter tuning

Model building

Model saving

End

Feature Generation

Deployment

Export into csv

Push to GitHub

Flask setup

Encoding Categorical Data

New feature creation

**5.1Data Gathering**

Data source: <https://question.transtutors.com/6129343_1_tourism-data.xlsx>

Train and Test data are stored in .csv format.

**5.2 Data Validation**

Following the loading of data, it is essential to conduct various validations before proceeding with any further operations. These validations include checking for zero standard deviation across all columns and identifying columns with complete missing values. These checks are imperative because attributes exhibiting these characteristics are deemed useless and do not contribute to the sales of items from respective outlets.

For instance, if an attribute displays zero standard deviation, it implies that all values are the same, with a mean of zero. This suggests that regardless of whether the sales are increasing or decreasing, that attribute will remain constant. Similarly, if an attribute contains entirely missing values, including it in operations serves no purpose and unnecessarily increases the risk of the curse of dimensionality.

**5.3 Data Transformation**

Prior to sending the data into the database, it is necessary to undergo data transformation to convert it into a format suitable for easy insertion into the database. Notably, the 'Age,' 'Duration of Pitch,' and 'Monthly Income' attributes exhibit missing values. Therefore, in both the train set and the test set, these missing values are filled with appropriate data types to ensure completeness and compatibility with the database structure.

**5.4 Data Preprocessing**

Before constructing the model, extensive pre-processing was conducted on the customer data. The handling of missing values took into account data type and distribution; for instance, numerical features were imputed with the mean, categorical features were filled with the mode, and potentially removed if deemed significant. Correction or removal of invalid values depended on their severity and impact. Outliers were identified, and their influence on the analysis led to their addressing and removal. Additionally, feature scaling and normalization were implemented to guarantee uniform scales for all features, thereby enhancing the efficiency of the model building process.

**5.5 Feature Engineering**

Feature Engineering was conducted after the pre-processing stage, revealing that certain attributes were deemed unimportant for the specific outlet. Consequently, these irrelevant attributes were removed from the dataset. Additionally, one-hot encoding was implemented to transform categorical features into numerical features, enhancing their compatibility for subsequent analysis and modeling.

**5.6 Pipelining**

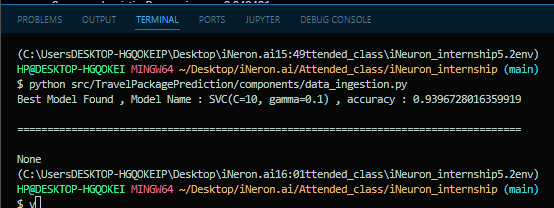
During the pre-processing phase of my project, distinct pipelines were set up for handling numerical and categorical features separately. The numerical pipeline is designed to address tasks such as imputation and scaling, while the categorical pipeline employs methods such as one-hot encoding. This customized approach ensures that each type of feature undergoes suitable processing, thereby optimizing both the performance and interpretability of the model. The implementation of these pipelines streamlines the pre-processing procedures, facilitating the efficient transformation of the dataset and ultimately improving the accuracy of our predictive models.

**5.7 Parameter Tuning**

The tuning of parameters was carried out using GridSearchCV, where various algorithms including Logistic Regression, Decision Tree SVM, Random Forest, Naive\_Bayes\_Classifier, KNeighborsClassifier and many more algorithms were employed for solving the problem. The parameters of these algorithms were fine-tuned and incorporated into the models. Remarkably, the SVC emerged as the most effective, yielding a training accuracy of 99% and a testing accuracy of 93.9%.

**5.8 Model Building**

After completing the comprehensive preprocessing operations outlined earlier, including scaling and hyperparameter tuning, the dataset underwent evaluation using different models. Among them, the Support Vector Classifier (SVC) exhibited the most favorable performance, achieving a remarkable training accuracy of 99% and a testing accuracy of 93.9%. These results indicate that the SVC model excelled in addressing the complexities of the given problem and demonstrated robust performance, particularly in accurately classifying instances in both the training and testing datasets.



**5.9 Model Saving**

Model is saved using pickle library in `.pkl` format.

**5.10 GitHub**

The whole project directory will be pushed into the GitHub repository.Then from the git hub it will be uploaded to the cloud platform for the deployment.

**5.11 Deployment**

The cloud environment was set up and the project was deployed from GitHub into the AWS cloud platform.

App link- https://vxh7bvmiaw.eu-west-3.awsapprunner.com/predictdata

**6. User Input / Output Workflow.**

****