Build a Lead Scoring Model using Machine Learning in Python
Project Overview

Overview
A company specializing in B2C sales, specifically offering Data Science and Data Engineering courses, is currently facing challenges in efficiently identifying and prioritizing high-quality leads. The current approach involves sales representatives manually qualifying and scoring leads based on limited information, resulting in wasted time and resources on low-quality leads.

To address this issue, the company aims to implement a more effective lead scoring system. Lead scoring is a methodology used by businesses to rank and prioritize leads based on their level of interest and potential for conversion into customers. By adopting a robust lead scoring model, the company can focus on the most promising leads, allowing their sales and marketing teams to allocate their time and resources more efficiently.

The lead scoring process involves assigning a numerical value or score to each lead based on specific criteria and behaviors. These criteria typically include demographic information such as job title and company size, as well as behavioral data like website visits, email engagement, and social media interactions. The weights assigned to each criterion may vary depending on the business and its target audience.

The objective of this project is to develop a machine learning-based lead scoring model that accurately predicts the likelihood of a lead converting into a paying customer. By leveraging advanced algorithms, this model will empower the sales team to prioritize high-quality leads, thereby enhancing the overall efficiency of the sales process.

Aim
The aim of this project is to develop a machine learning-based lead scoring model that accurately predicts the conversion likelihood of leads, enabling efficient prioritization of high-quality leads for the sales team.

Data Description
The dataset contains the following columns:

- Lead Id: A unique identifier assigned to each lead in the dataset.

- Lead Owner: The internal salesperson associated with the lead.
- Interest Level: Indicates the level of interest expressed by the lead. This information is entered manually.
- The lead created: The date when the lead was created.
- Lead Location(Auto): The location of the lead, is automatically detected.
- Creation Source: The source from which the lead was generated.
- Next activity: The date scheduled for the next activity with the lead.
- What do you do currently?: Describes the current profile or occupation of the lead.
- What are you looking for in a Product?: Specifies the specific requirements or expectations of the lead regarding the product.
- Website Source: The source from which the lead visited the website.
- Lead Last Update time: The timestamp of the last update made to the lead's information.
- Marketing Source: The marketing source through which the lead was acquired.
- Lead Location(Manual): The manually entered location of the lead.
- Demo Date: The date scheduled for a product demonstration with the lead.
- Demo Status: Indicates the status of the demo booked with the lead.
- Closure date: The date when the lead was successfully closed or converted into a customer.

Tech Stack

➔ Language: Python

➔ Libraries: pandas, numpy, matplotlib, scikit-learn, xgboost, lightgbm

Approach

- Exploratory Data Analysis (EDA):
    - Understand the features and their relationships with target variables
    - Check for missing or invalid values and their imputation

- Data Preprocessing:
    - Encode the variables using label encoding
    - Split the dataset into training and testing sets

- Model Building and Testing:
    - Logistic Regression

- Naive Bayes
- Support Vector Machines
- Random Forest
- Light Gradient Boosting
- Extreme Gradient Boosting
- Neural Network

Modular code overview:

```
├── data
│   └── raw            <- The original, immutable data dump
│
├── models             <- Trained and serialized models
│
├── notebooks          <- Comprehensive jupyter notebooks
│
├── references         <- Explanatory materials
│
├── reports            <- Generated analysis as HTML, PDF etc.
│
├── requirements.txt   <- Requirements file for conda environment with proper versions for each library
│
├── src                <- Source code for use in this project.
│   │
│   ├── ML_Pipeline    <- Modularized code (.py files) -- all the functions must be paramatrized
│   │   │
│   │   │
│   │   ├── processing.py                  <- Source code to process data
│   │   │
│   │   ├── modeling.py                    <- Source code to train and score models
│   │   │
│   │   │
│   │
│   └── Engine.py      <- Main function where all the files from ML_Pipeline are called
```

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

1. data
2. models
3. src
4. reports
5. references
6. notebooks
7. requirements.txt
8. readme.md

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

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

Project Takeaways

1. Gain a comprehensive understanding of the dataset's features and their relationships with the target variables through exploratory data analysis (EDA).
2. Identify any missing or invalid values in the dataset and apply appropriate imputation techniques.
3. Encode categorical variables using label encoding or other suitable methods for better model performance.
4. Implement and evaluate the performance of various machine learning algorithms including Logistic Regression, Naive Bayes, Support Vector Machines, Random Forest, Light Gradient Boosting, Extreme Gradient Boosting, and Neural Network.
5. Compare the results of different algorithms to determine the most effective approach for lead scoring.
6. Assess the accuracy of each model and identify their strengths and weaknesses.
7. Gain insights into the factors that significantly influence lead conversion and customer acquisition.