High Level Design (HLD)

Insurance Premium Prediction

Revision Number: 1.0

Last date of revision: 03/11/2022

Vishal Choudhary

Contents

Document Version Control

Date Issued	Version	Description	Author
03-11-2022	1.0	First Version of Complete HLD	Vishal Choudhary

Abstract

Health insurance is a necessity nowadays, and almost every individual is linked with a government or private health insurance company. Factors determining the amount of insurance vary from company to company. Also people in rural areas are unaware of the fact that the government of India provide free health insurance to those below poverty line. It is very complex method and some rural people either buy some private health insurance or do not invest money in health insurance at all. Apart from this people can be fooled easily about the amount of the insurance and may unnecessarily buy some expensive health insurance.

Our project does not give the exact amount required for any health insurance company but gives enough idea about the amount associated with an individual for his/her own health insurance.

1 Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) document is to add the necessary detail to the project description to represent a suitable model and coding for application. This document is also intended to help detect contradictions before coding and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

- Present all of the design aspects and define them in detail
- Describe the user interface is implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project
- List and describe the non-functional attributes like:
    - Security
    - Reliability
    - Maintainability
    - Portability
    - Reusability
    - Application compatibility
    - Resource utilization
    - Serviceability

1.2 Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology stack. The HLD uses non-technical to mildly technical terms which should be understandable to the administrators of the system.

2 General Description

2.1 Problem Statement & Product Perspective

The dataset contains Age, Sex, BMI (Body Mass index), Children, Smoker, Region and Expenses where I have to predict Insurance Premium Expenses with

- To detect BMI value affects the premium.
- To detect smoking affects the premium of the insurance.
- To create API interface to predict the premium

2.2 Proposed Solution

The solution proposed here is an estimating premium of insurance basedon people health data and this can be implemented to perform above mention use cases. In first case, analyzing how BMI value affect the people health as well as premium of the insurance. In the second case, ifmodel detects the smoking affecting the premium, we will inform that to people. And in the last use case, we will be making an interface to predict the premium.

2.3 Technical Requirements

The solution can be a cloud-based or application hosted on an internal server or even be hosted on a local machine. For accessing this applicationbelow are the minimum requirements:

- Good internet connection.
- Web Browser.

For training model, the system requirements are as follows:

1. +4 GB RAM preferred

- Operation System: Windows, Linux, Mac
- Visual Studio Code / Jupyter notebook

2.4 Data Requirements

Data requirements completely depends on out problem statement.

- Comma separated values (CSV) file.
- Input file feature/field names and its sequence should be followedas per decided.

2.5 Tools used

- Python programming language.
- Libraries such as Pandas, Numpy, Scikit-Learn, Matplotlib
- Database: MongoDB
- Framework: Flask
- IDE: Jupyter Notebook,VSCode.
- Cloud service: Heroku
- CI/CD pipeline : Github Actions

- Git and Github.
- VSCode and Jupyter notebook is used as IDE.
- For Visualization of the plots Matplotlib is used.
- Heroku is used for deployment of the model.
- Front-end development is done using HTML/CSS.
- Python Flask is used for backend development.
- Github Actions is used for CI/CD pipeline.
- GitHub is used for version control system

2.6 Constraints

MLOPs on the cloud must be fully automated in consideration of continuous integration, continuous deployment with retraining approach of model, and archiving the data over time.

The application should be user friendly as automated as possible. Users can easily use the application and not needed to know any of the workings.

2.7 Assumptions

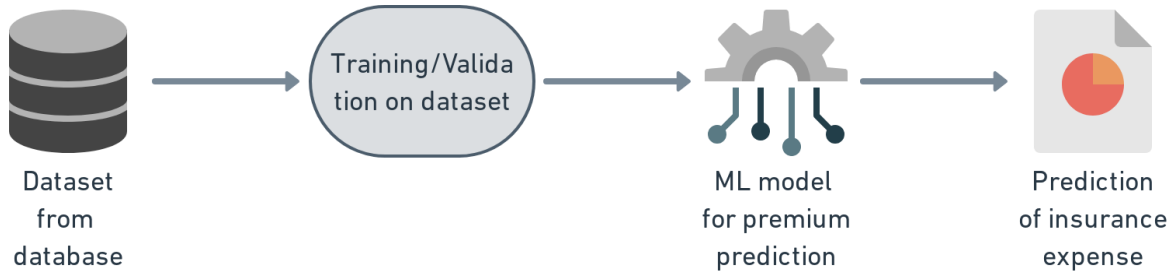
The main objective of the project is to develop an API to predict the premium for people on the basis of their health information. Machine learning based regression model is used for predicting above mentioned cases on the input data.

3 Design Details

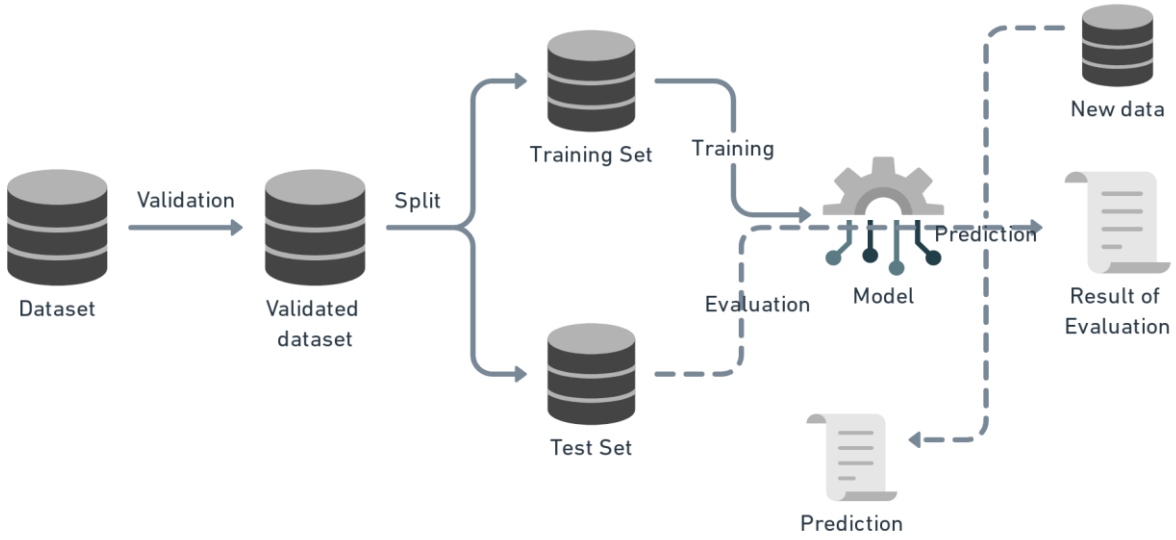
3.1 Process flow

For finding the insurance premium we will be using machine learning model. Below is the process flow diagram.

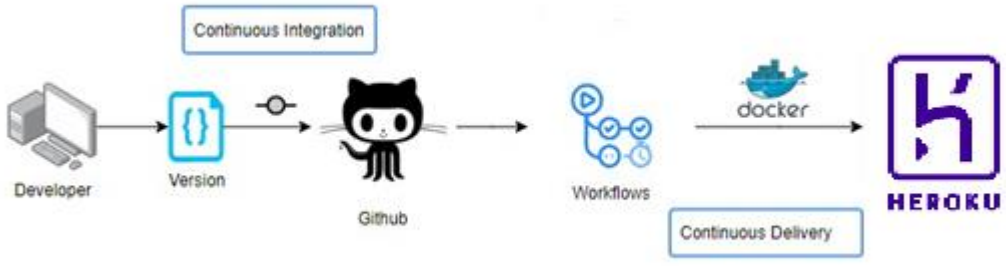
Proposed methodology



3.1.1 Model Training and Evaluation



3.1.2 Deployment Process



3.2 Event log

The system should log every event so that the track of every detail will be known and what process is running currently could be seen.

Initial Step-By-Step Description:

1. The System identifies at what step logging is required.
2. The System should be able to log each and every system flow.
3. Developer can choose logging method. You can choose database logging/ File logging as well.
4. System should not hang even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.

3.3 Error Handling

The system should identify the errors encountered, an explanation will be displayed as to what went wrong? An error will be defined as anything that falls outside the normal and intended usage.

3.4 Optimization

Data strategy derives performance:

1. Filling missing values.
2. Replacing outliers.
3. Creating new features from cut expenses feature.
4. Hyper parameter tuning
5. Validating score again

3.5 Reusability

The entire solution will be done in modular fashion and will be API oriented. So, in the case of the scaling the application, the components are completely reusable.

3.6 Application compatibility

The different components for this project will be using Python as an interface between them. Each component will have its task to perform, and it is the job of Python to ensure the proper transfer of information.

3.7 Deployment

4 Conclusion

In this projects, The system shows us that the different techniques that are used in order to estimate the how much amount of premium required on the basis of individual health situation. After analyzing it shows how a smoker and non-smokers affecting the amount of estimate. Also, significant difference between male and female expenses. Accuracy, which plays a key role in prediction-based system. From the results we could see that Gradient Boosting turned out to be best working model for this problemin terms of the accuracy. Our predictions help user to know how much amount premium they need on the basis of their current health situation.