Architecture Design

Insurance Premium Prediction

Document Version Control

Version	Date	Author	Comments
1.1	31/01/2024	Yash Dahekar	

Contests

Abstract

Machine Learning is a category of algorithms that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. The basic premise of machine learning is to build models and employ algorithms that can receive input data and use statistical analysis to predict an output while updating outputs as new data becomes available. These models can be applied in different areas and trained to match management expectations so that proper steps can be taken to achieve the organization's target. In this project, we will estimate the insurance premium based on personal health information. Taking various aspects of a dataset collected from people and the methodology followed for building a predictive model.

1 Introduction

1.1 What is Architecture Design?

The goal of Architecture Design (AD) is to give the internal design of the actual program code for the `Insurance Premium Prediction`. 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 estimating cost of expenses to customers based on some personal health information.

2 Technical Specifications

2.1 Dataset

The dataset contains verified historical data, consisting of the aforementioned information and the actual medical expenses incurred by over 1300 customers. The objective is to find a way to estimate the value in the "expenses" column using the values in the other columns like their age, sex, BMI, no. of children, smoking habits and region. Using all the observations it is inferred what role certain properties of user and how they affect their expenses. The dataset looks like as follow:

```
# Displaying dataset

dataset
```

	age	sex	bmi	children	smoker	region	expenses
0	19	female	27.9	0	yes	southwest	16884.92
1	18	male	33.8	1	no	southeast	1725.55
2	28	male	33.0	3	no	southeast	4449.46
3	33	male	22.7	0	no	northwest	21984.47
4	32	male	28.9	0	no	northwest	3866.86
...	...	...	...	...	...	...	...
1333	50	male	31.0	3	no	northwest	10600.55
1334	18	female	31.9	0	no	northeast	2205.98
1335	18	female	36.9	0	no	southeast	1629.83
1336	21	female	25.8	0	no	southwest	2007.95
1337	61	female	29.1	0	yes	northwest	29141.36

1338 rows × 7 columns

```
--> This dataset consist of 1338 rows and 7 columns.
```

The data set consists of various data types from integer to floating to object as shown in Fig.

```
# Datatype of each feature.
dataset.dtypes
```

[10] ✓ 0.1s

```
age            int64
sex           object
bmi          float64
children       int64
smoker        object
region        object
expenses     float64
dtype: object
```

In the dataset, 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. Looks like 'age', 'children', 'BMI' (body mass index) and 'expenses' are numbers, whereas 'sex', 'smoker', and 'region' are strings (possibly categories). Various factors important by statistical means like mean, standard deviation, median, count of values and maximum value, etc. are shown below for numerical attributes.

```
dataset.describe()
```

[12] ✓ 0.2s

	age	bmi	children	expenses
count	1337.000000	1337.000000	1337.000000	1337.000000
mean	39.222139	30.665520	1.095737	13279.121638
std	14.044333	6.100664	1.205571	12110.359657
min	18.000000	16.000000	0.000000	1121.870000
25%	27.000000	26.300000	0.000000	4746.340000
50%	39.000000	30.400000	1.000000	9386.160000
75%	51.000000	34.700000	2.000000	16657.720000
max	64.000000	53.100000	5.000000	63770.430000

Preprocessing of this dataset includes doing analysis on the independent variables like checking for null values in each column and then replacing or filling them with supported appropriate data types so that analysis and model fitting is not hindered from their way to accuracy. Shown above are some of the representations obtained by using Pandas tools which tell about variable count for numerical columns and model values for categorical columns. Maximum and minimum values in numerical columns, along with their percentile values for median, play an important factor in deciding which value to be chosen as a priority for further exploration tasks and analysis. Data types of different columns are used further in label processing and a one-hot encoding scheme during the model building.

2.2 Logging

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

- The system identifies which step logging require.
- The system should be able to log each and every system flow.
- The system should 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.

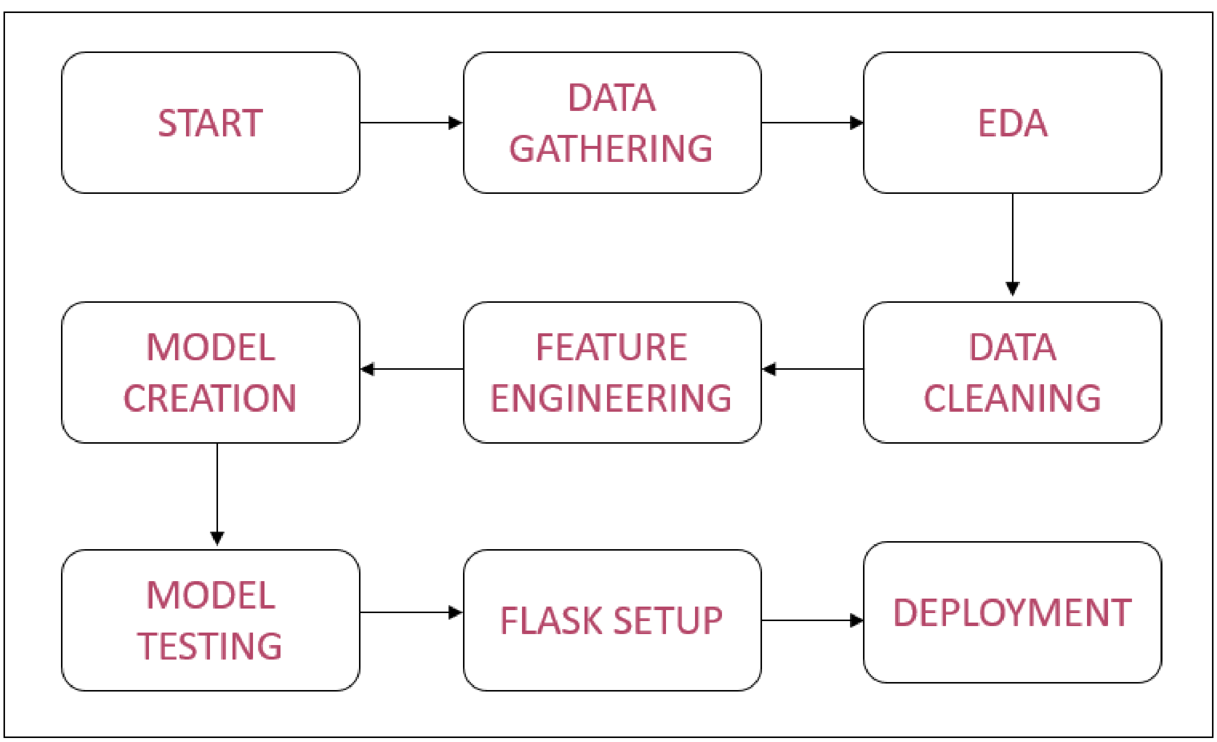
3 Technology Stack

Front End	HTML/CSS
Backend	Python/Flask
Deployment	AWS

4 Proposed Solution

We will use performed EDA to find the important relation between different attributes and will use a machine-learning algorithm to estimate the cost of expenses. The client will be filled the required feature as input and will get results through the web application. The system will get features and it will be passed into the backend where the features will be validated and preprocessed and then it will be passed to a hyper parameter tuned machine learning model to predict the final outcome.

5 Architecture



5.1 Data Gathering

Data source: https://www.kaggle.com/noordeen/insurance-premium-prediction
Dataset is stored in .csv format.

5.2 Raw Data Validation

After data is loaded, various types of validation are required before we proceed further with any operation. Validations like checking for zero standard deviation for all the columns, checking for complete missing values in any columns, etc. These are required because the attributes which contain these are of no use. It will not play role in contributing to the estimating cost of the premium.

5.3 Exploratory Data Analysis

Visualized the relationship between the dependent and independent features. Also checked relationship between independent features to get more insights about the data.

5.4 Feature Engineering

After pre-processing standard scalar is performed to scale down all the numeric features. Even one hot encoding is also performed to convert the categorical features into numerical features. For this process, pipeline is created to scale numerical features and encoding the categorical features.

5.5 Model Building

After doing all kinds of pre-processing operations mentioned above and performing scaling and encoding, the data set is passed through a pipeline to all the models, Linear Regression, Decision tree, Random Forest and Gradient booster. It was found that Gradient boosting performs best with the on-test data after that we perform Gradient Search CV on gradient booster to find out best parameters, So after 'Gradient Search CV' on gradient booster our model performance increases.

5.6 Model Saving

The model is saved using the dill library in. pkl format.

5.7 Flask Setup for Web Application

After saving the model, the API building process started using Flask. Web application creation was created in Flask for testing purpose. Whatever user will enter the data and then that data will be extracted by the model to estimate the premium of insurance, this is performed in this stage.

5.8 GitHub

The whole project directory will be pushed into the GitHub repository.

5.9 Deployment

The project was deployed from GitHub into the AWS platform.

6 User I/O Workflow