

ARCHITECTURE DESIGN

Insurance Premium Prediction

Done By : Md Rian Ur Rahaman

Contents

Abstract	3
1. Introduction	3
1.1 Why this Architecture Design Document?	3
2. Architecture	4
3. Architecture Design	4
3.1 Data Collection	5
3.2 Data Description	6
3.3 Loading dataset	6
3.4 Data Preprocessing	6
3.5 Modeling Process	7
3.6 UI Integration	7
3.7 Data from User	7
3.8 Data Validation	8
3.9 Rendering the Results	8
3.10 Deployment	8

ABSTARCT

This data science project focuses on developing a comprehensive solution to predict and mitigate customer churn in a subscription-based business model. The project follows a systematic end-to-end approach, encompassing data collection, preprocessing, exploratory data analysis (EDA), feature engineering, model selection, training, and deployment. The primary objective is to empower businesses with actionable insights to retain valuable customers and optimize revenue streams.

The project begins with data collection from diverse sources, including customer interactions, transaction history, and demographic information. Data preprocessing involves cleaning, handling missing values, and transforming raw data into a format suitable for analysis. The EDA phase employs statistical and visual techniques to uncover patterns, correlations, and potential factors influencing customer churn.

Feature engineering is a crucial step, extracting meaningful insights from the data to enhance predictive model performance. Various machine learning algorithms, such as logistic regression, decision trees, and ensemble methods, are explored and evaluated for their ability to predict customer churn accurately. Model hyperparameter tuning and cross-validation techniques are applied to optimize the model's performance.

The project emphasizes the importance of interpretability, providing stakeholders with clear insights into the factors contributing to customer churn. Model explanations and visualizations aid in understanding the decision-making process and guide strategic decision-making.

1. Introduction

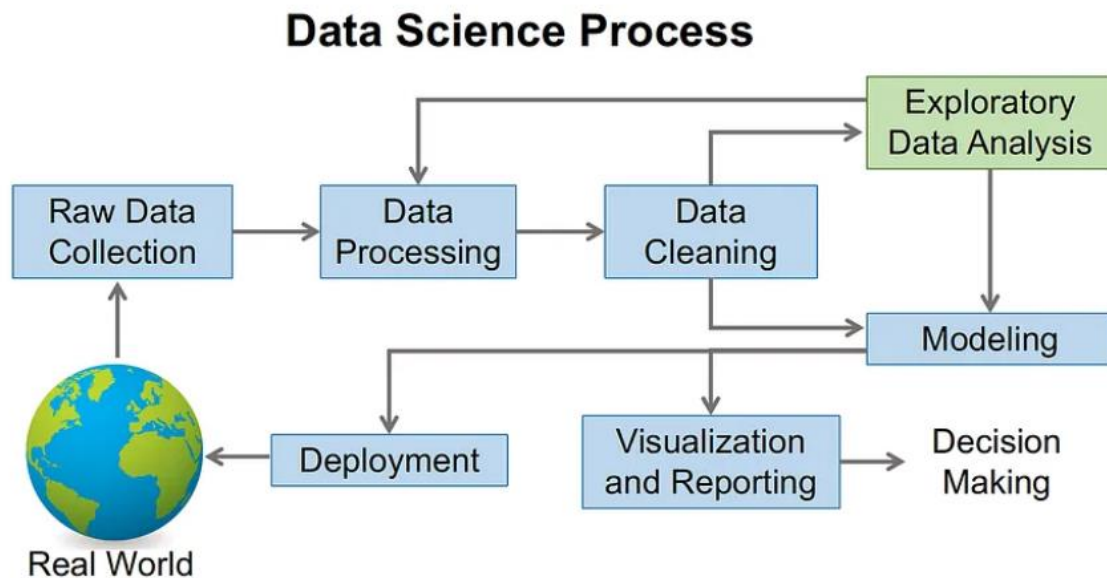
1.1 Why this Architecture Design Document ?

The main objective of the Architecture design documentation is to provide the internal logic understanding of the Rental Bike share demand prediction code.

The Architecture design documentation is designed in such a way that the programmer can directly code after reading each module description in the documentation.

2. Architecture

Key Steps in the Data Science Process



Key Steps in the Data Science Process

3. Architecture Design

3.1 Data Collection

The dataset was taken from Kaggle dataset ,available in this link :



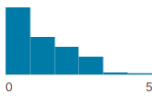
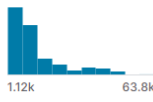
<https://www.kaggle.com/datasets/noordeen/insurance-premium-prediction>

3.2 Data Description

About Dataset

Context

The insurance.csv dataset contains 1338 observations (rows) and 7 features (columns). The dataset contains 4 numerical features (age, bmi, children and expenses) and 3 nominal features (sex, smoker and region) that were converted into factors with numerical value designated for each level.

# age	Δ sex	# bmi	# children	✓ smoker	Δ region	# expenses
	male 51% female 49%			true 0 0% false 0 0%	southeast 27% southwest 24% Other (649) 49%	
19	female	27.9	0	yes	southwest	16884.92
18	male	33.8	1	no	southeast	1725.55
28	male	33.0	3	no	southeast	4449.46
33	male	22.7	0	no	northwest	21984.47
32	male	28.9	0	no	northwest	3866.86
31	female	25.7	0	no	southeast	3756.62
46	female	33.4	1	no	southeast	8240.59
37	female	27.7	3	no	northwest	7281.51
37	male	29.8	2	no	northeast	6406.41
60	female	25.8	0	no	northwest	28923.14
25	male	26.2	0	no	northeast	2721.32
62	female	26.3	0	yes	southeast	27808.73
23	male	34.4	0	no	southwest	1826.84
56	female	39.8	0	no	southeast	11090.72
27	male	42.1	0	yes	southeast	39611.76
19	male	24.6	1	no	southwest	1837.24

3.3 Loading Dataset

Local file system was used for loading the dataset using read_csv function of Pandas Library.

3.4 Data Preprocessings

To preprocess this dataset, which contains information about individuals' age, sex, BMI, number of children, smoking status, region, and medical expenses, several steps are necessary to prepare it for analysis or modeling. First, we'll handle any missing values to ensure the dataset's completeness, as missing data can impact model performance. Next, we'll convert categorical variables, such as 'sex', 'smoker', and 'region', into numerical formats using techniques like one-hot encoding. This step is crucial for algorithms that require numerical input. For instance, 'sex' will be transformed into binary variables, while 'region' and 'smoker' can be encoded into separate columns.

We will also normalize or standardize the numerical features, such as 'age', 'BMI', and 'expenses', to ensure they are on a similar scale, which can improve the performance of many machine learning algorithms. Additionally, the 'children' column, which contains count data, will be left as is since it is already numerical. After these transformations, we will split the data into training and testing sets to evaluate the model's performance effectively. If necessary, feature engineering techniques may be applied to create new features that could enhance the model's predictive power. Finally, we will ensure that the data is ready for input into various machine learning models, aiming to predict the 'expenses' based on the other attributes. This comprehensive preprocessing will help build a robust model and derive meaningful insights from the data.

3.5 Model Training :

The provided code snippet demonstrates the process of training and selecting the best model for predicting insurance premiums. The code begins by importing essential libraries, including various regression models from sklearn, custom exception handling, logging functionalities, and utility functions for saving objects and evaluating models. The `ModelTrainerConfig` dataclass is used to specify the file path for saving the trained model. The `ModelTrainer` class is then defined, initializing the configuration and providing the `initiate_model_training` method.

In the `initiate_model_training` method, the training and testing datasets are split into dependent (y) and independent (X) variables. A dictionary of regression models is created, including `LinearRegression`, `DecisionTreeRegressor`, `RandomForestRegressor`, `AdaBoostRegressor`, `SGDRegressor`, `Lasso`, `ElasticNet`, `SVR`, and `Ridge`. The `evaluate_model` function is then called to evaluate each model's performance on the training and testing data, and the results are logged and printed.

The method identifies the best-performing model based on the highest R2 score from the evaluation report. The name and score of the best model are logged and printed, highlighting the best model found. Finally, the best model is saved to the specified file path using the `save_object` utility function. Exception handling ensures that any errors during model training are logged, and a custom exception is raised. This structured approach ensures robust model training and selection, facilitating effective prediction of insurance premiums.

3.6 UI Integration

As stated in the problem statement we have to create a user interface page where users can input their data and our machine learning model will give them the predicted result.

3.7 Data Taken from the user

Data from the user is retrieved using the HTML & CSS web app, and output is calculated using the `app.py` as our Flask application which is using our machine learning model to give the predicted result.

3.8 Data Validation

The data which is entered by the user is validated by the `app.py` file which is built using the python flask and then this data is transferred to our model.

3.9 Rendering the result

The result for our model is rendered at the HTML page. As we created an API for our predicted result page.

