Architecture

Architecture

Insurance Premium Prediction

Revision Number – 1.0

Last Date of Revision – 29-06-2023

**ILMA ARSHAD**

1

Architecture



# Liabaries Requirment

Flask

pandas

numpy

scikit-learn

seaborn

ARCHITECTURE

**Contents**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**Abstract**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.gjdgxs) | | | | | | | | | | | | | | | [4](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.gjdgxs) | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| [**Introduction**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.30j0zll) | | |  | | | | | | | | | | | | [4](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.30j0zll) | | | |
|  |  | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | [**Why this Architecture Design documentation?**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.1fob9te) | | | | | | | | | | | | | |  | | [4](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.1fob9te) |  |
| [**1 Architecture**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.3znysh7) | | | |  | | | | | | | | | | | [4](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.3znysh7) | | | |
|  | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| [**2 Architecture design**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.3dy6vkm) | | | | | | | |  | | | | | | | [5](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.3dy6vkm) | | |  |
|  |  | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | [**2.1 Data gathering from main source**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.1t3h5sf) | | | | | | | | | | | | | | [5](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.1t3h5sf) | | |  |
|  |  | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | [**2.2 Data description**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.4d34og8) | | | | | | | | | | | | |  | [5](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.4d34og8) | | |  |
|  |  | | | |  |  |  |  |  |  |  |  |  | |  |  |  |  |
|  | [**2.3 Upload data into Cassandra**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.2s8eyo1) | | | | | | | | | | | |  | | [5](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.2s8eyo1) | | | |
|  |  | | | |  |  |  |  | |  |  |  |  | |  |  |  |  |
|  | [**2.4 Export data from database**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.17dp8vu) | | | | | | | | | | |  |  | | [5](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.17dp8vu) | | |  |
|  |  | | | |  |  |  |  | |  |  |  | | |  |  |  |  |
|  | [**2.5 Data pre-processing**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.3rdcrjn) | | | | | | | | | | |  | | |  | [5](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.3rdcrjn) | |  |
|  |  | | | |  |  |  |  | |  |  | | | |  | |  |  |
|  | [**2.6 Modelling**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.26in1rg) | | | |  | | | | |  | | | | |  | | [6](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.26in1rg) | |
|  |  | | | |  |  |  |  | | |  | | | |  | |  |  |
|  | [**2.7 UI integration**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.lnxbz9) | | | | |  | | | | | | | | | [6](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.lnxbz9) | | |  |
|  |  | | | | |  |  |  | | |  | | | |  | |  |  |
|  | [**2.8 Data from user**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.35nkun2) | | | | | | | | | | | | | | [6](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.35nkun2) | | |  |
|  |  | | | | |  |  |  | | |  | | | |  | |  |  |
|  | [**2.9 Data validation**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.1ksv4uv) | | | | | | |  | | | | | | | [6](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.1ksv4uv) | | |  |
|  |  | | | | |  | |  | | |  | | | |  | |  |  |
|  | [**2.10 Rendering the results**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.44sinio) | | | | | | | | | |  | | | | [6](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.44sinio) | | |  |
|  | [**2.11 Deployment**](https://docs.google.com/document/d/14Wck6wDTC8bscpCxNjrSRu01-UC_tJ5Y/edit#heading=h.2jxsxqh) | | | | | | | | | |  | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

3

ARCHITECTURE

Abstract

Insurance premium prediction plays a crucial role in the insurance industry, enabling accurate risk assessment and pricing strategies. This study presents a comprehensive analysis of insurance premium prediction using advanced machine learning techniques. The research leverages a diverse dataset comprising historical insurance claims, policyholder information, and external factors impacting risk, such as demographic data and socio-economic indicators.

The proposed predictive model employs state-of-the-art algorithms, including ensemble methods, deep learning architectures, and regression techniques, to capture complex relationships and patterns within the data. Feature engineering techniques are employed to extract relevant information and optimize predictive performance. The model is trained and validated on a large-scale dataset, ensuring robustness and generalizability.

The study evaluates the performance of the predictive model by employing various evaluation metrics, including mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE). Additionally, interpretability techniques are applied to gain insights into the factors driving premium calculations, providing transparency and aiding decision-making processes.

Experimental results demonstrate the effectiveness of the proposed predictive model in accurately estimating insurance premiums. The model outperforms traditional approaches and achieves high prediction accuracy, thereby assisting insurance companies in setting appropriate premium rates based on individual risk profiles. The insights gained from the interpretability analysis further enhance understanding of premium determinants, enabling insurers to offer personalized products and improve customer satisfaction.

Overall, this research contributes to the field of insurance premium prediction by developing a robust and accurate model that integrates diverse data sources and advanced machine learning techniques. The findings have the potential to revolutionize the insurance industry, leading to improved risk assessment, fairer pricing, and enhanced profitability for insurers, while ensuring policyholders receive appropriate coverage at competitive rates.

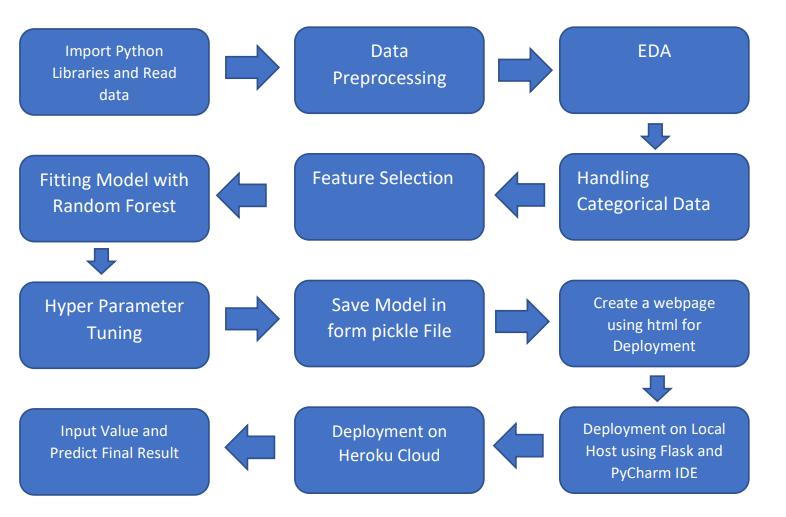
Top of Form

Introduction

Why this Architecture Design documentation?

The main objective of the Architecture design documentation is to provide the internal logic understanding of the insurance premium 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.

1 Architecture



4

ARCHITECTURE

2 Architecture design

Designing an architecture involves creating a blueprint or plan that outlines the structure, components, and interactions of a system. Here are some key considerations when designing an architecture:

1. Identify Requirements: Understand the functional and non-functional requirements of the system. This includes determining the goals, features, scalability, performance, security, and other aspects that the architecture should fulfill.
2. Choose Architectural Style: Select an appropriate architectural style or pattern that best suits the system requirements. Common styles include client-server, layered, microservices, event-driven, and service-oriented architectures.
3. Define Components: Identify the major components or modules that will make up the system. Components can be logical divisions such as user interface, business logic, data storage, external interfaces, and integrations.
4. Component Interaction: Determine how the components will interact with each other. Define the communication protocols, data flow, and dependencies between components. Consider whether synchronous or asynchronous communication is needed.
5. Scalability and Performance: Design the architecture to handle the expected workload and scale efficiently. Consider factors such as load balancing, caching, horizontal or vertical scaling, and optimizing performance bottlenecks.
6. Security: Incorporate security measures to protect the system and data. This includes authentication, authorization, encryption, input validation, and other security best practices relevant to the system.
7. Data Management: Plan how data will be stored, accessed, and managed. Consider the type of database (relational, NoSQL, etc.), data synchronization, data consistency, and backup strategies.
8. Deployment and Infrastructure: Determine the deployment architecture, including the hardware and software infrastructure required to host the system. Consider factors such as cloud platforms, server configurations, network architecture, and monitoring.
9. Error Handling and Resilience: Define strategies for handling errors and failures. This may involve implementing retries, redundancy, fault tolerance, and graceful degradation to ensure the system remains operational in the face of issues.
10. Maintainability and Extensibility: Design the architecture to be easily maintainable and extensible. Use modular and decoupled components, follow coding best practices, and consider future changes and integrations.
11. Documentation and Communication: Document the architecture design, including diagrams, descriptions, and any relevant documentation that helps communicate the design to stakeholders, developers, and other team members.

Remember that architecture design is an iterative process, and it may involve trade-offs and refinements based on feedback, constraints, and evolving requirements. Collaboration among stakeholders, architects, and development teams is crucial for creating a robust and effective architecture.

2.1 Data gathering from main source

The data for the current project is being gathered from Kaggle dataset, the link to the data is:

[LINK](https://www.kaggle.com/datasets/noordeen/insurance-premium-prediction)

2.2 Data description

Predicting insurance premium prediction is a complex task that involves analyzing various factors related to an individual's age, sex, bmi, and other relevant variables. While I can provide you with some general information, please note that predicting accurately for specific individuals can be challenging due to the wide range of factors involved and the inherent uncertainty of future outcomes.

2.3 Upload data into Cassandra

Created an api for the upload of the data into the Cassandra database, steps performed are:

* Connection is made with the database.
* Created a database with name insurance.
* Cqlsh command is written for creating the data table with required parameters.
* And finally, a cqlsh command is written for uploading the dataset into data table by bulk insertion.

2.4 Export data from database

In the above created api, the download url is also being created, which downloads the data into a csv file format.

2.5 Data pre-processing

Steps performed in pre-processing are:

* First the data types are being checked and found only the price column is of type integer.
* Checked for null values as there are few null values, those rows are dropped.
* Converted all the required column into the date time format.
* Performed one-hot encoding for the required columns.
* Scaling is performed for required data.
* And, the data is ready for passing to the machine learning algorithm.

2.6 Modelling

The pre-processed data is then visualized and all the required insights are being drawn. Although from the drawn insights, the data is randomly spread but still modelling is performed with different machine learning algorithms to make sure we cover all the possibilities. And finally, as expected random forest regression performed well and further hyperparameter tuning is done to increase the model’s accuracy.

2.7 UI integration

Both CSS and HTML files are being created and are being integrated with the created machine learning model. All the required files are then integrated to the app.py file and tested locally. Note I did not make the CSS and HTML File .

2.8 Data from user

The data from the user is retrieved from the created HTML web page.

2.9 Data validation

The data provided by the user is then being processed by app.py file and validated. The validated data is then sent for the prediction.

5

2.10 Rendering the results

The data sent for the prediction is then rendered to the web page.

2.11 Deployment

The tested model is then deployed to AWS. So, users can access the project from any internet devices.

Here is the link :

<http://insurance-premium-env.eba-mwe2erpv.us-east-1.elasticbeanstalk.com/>

6