

DOCUMENT VERSION CONTROL

| Date Issue | Version | Description | Author |
| --- | --- | --- | --- |
| 07/03/2023 | 1 | Initial Architecture – V 1.0 | Abdul Jaweed  Lovely Patra |
| 17/03/2023 | 2 | Final Architecture – V 2.0 | Abdul Jaweed  Lovely Patra |

CONTENT

* DOCUMENT VERSION
* ABSTRACT
* INTRODUCTION
* WHY THIS ARCHITECTURE DESIGN DOCUMENT
* ARCHITECTURE DESCRIPTION
* DATA DESCRIPTION
* DATA STORAGE
* DATA VALIDATION
* DATA TRANSFORMATION
* DATABASE OPERATION
* DATA PREPROCESSING
* MODEL TRAINING
* LOAD PRODUCTION MODEL
* CLOUD SETUP
* PUSH APP TO CLOUD
* DATA FROM CLIENT SIDE FOR PREDICTION PURPOSE
* DATA PROCESSING AND PREDICTION
* EXPORT PREDICTION TO CSV

ABSTRACT

Financial threats are displaying a trend about the credit risk of commercial banks as the incredible improvement in the financial industry has arisen. In this way, one of the biggest threats faced by commercial banks is the risk prediction of credit clients. The goal is to predict the probability of credit default based on credit card owner's characteristics and payment history.

For this, I have analyzed the “Default of Credit Card Clients” dataset released under the public license of Creative Commons to predict the default condition of credit card holders. Five classification models which are Logistic Regression, K-Nearest Neighbor, Naive Bayes Classifier, Support Vector Machines (LinearSVC), Decision Trees, Random Forest Classifier, Ensemble have been used to compare and contrast the performance of these algorithms.

The dataset was a training dataset which is used for model training and that trained model helped to come up with some predictions. Then the predicted observations were compared with actual data to test and verify the accuracy of the model.

Later the accuracies and matrices of all these models were compared. After this comparison it was found that Random Forest and Gaussian Naïve Bayes algorithms performed better than the remaining models.

Lastly, Gaussian Naïve Bayes is best suited in this case because it gives best

evaluation score compared to other models.

INTRODUCTION

WHY THIS ARCHITECTURE DESIGN DOCUMENT?

The main objective of the Architecture design documentation is to provide the internal logic understanding of the credit cards default 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.

ARCHITECTURE DESCRIPTION

DATA DESCRIPTION :

This dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005.

The credit card default prediction dataset which is available in UCI Machine Learning Repository consists of 30000 rows and 25 columns.

DATA STORAGE :

After the data is split into multiple csv files, we have uploaded the data to S3 buckets. From there we shall be picking the data for validation and preprocessing and model training.

DATA VALIDATION :

The data stored in S3 buckets are validated based on the filename, column names, missing values in the column. Once the validation is done the data is transferred to the good data bucket and other data to the bad data bucket.

DATA TRANSFORMATION :

Data validation is done on the data extracted from the sources, in which data types of the column, encoding of the categorical columns, and feature engineering are performed.

DATABASE OPERATION :

Now that we have processed and validated the data, and generated good data which is stored in the S3 bucket, we need to store them in a database which is MongoDB. MongoDB Atlas cluster is used as configuration. Once the good data is stored in MongoDB we need to export it as a csv file for training which will be stored in S3 buckets.

DATA PREPROCESSING :

We will be exploring our data set here and do EDA if required and perform data preprocessing depending on the data set. We first explore our data set in Jupyter Notebook and decide what pre-processing and Validation we have to do such as imputation of null values etc. and then we have to write separate modules according to our analysis, so that we can implement that for training as well as prediction data.

MODEL TRAINING :

Platform provides the ability for the end user to train models using our library and pipeline on the transformed and validated data based on request. The model training will add the best trained model to the system.

Here the best trained model is returned after completion of the complete machine learning pipeline.

LOAD PRODUCTION MODEL :

Once the models are trained and logged in MLFlow, now we need to push the best models for the particular cluster to production and staging or staging and archived based on condition.

CLOUD SETUP:

Here We will do cloud setup for model deployment. Here we also create our fast api app and user interface and integrate our model with fast api

PUSH APP TO CLOUD :

After doing cloud setup and checking the app locally, we will push our app to the cloud to start the application.

DATA FROM CLIENT SIDE FOR PREDICTION PURPOSE :

Now our application on cloud is ready for doing prediction. The prediction data which we receive from the client side.

DATA PROCESSING AND PREDICTION :

Client data will also go along the same process Data pre-processing and according to that we will predict those data.

EXPORT PREDICTION TO CSV :

Finally when we get all the predictions for client data, then our final task is to export the prediction to a csv file and hand it over to the client.