ARCHITECTURE DESIGN

BANK CREDIT RISK PREDICTION

DOCUMENT VERSION CONTROL

DATE ISSUED	VERSION	DESCRIPTION	AUTHOR
	V1.0	Architecture Design-V1.0	ADHIRAJ SINGH SHEKHAWAT

Abstract

This is an architecture design for a machine learning project aimed at predicting bank credit risk using south German credit data. The objective of the project is to develop an accurate and scalable machine learning model that can predict which customers are likely to default a loan, so that the bank can take proactive measures to retain them.

The proposed architecture comprises three main components: data preprocessing, model training and evaluation, and model deployment. The data preprocessing component involves data cleaning, feature selection and engineering, and data transformation, and is implemented using python packages. The model training and evaluation component involves selecting and tuning a suitable machine learning algorithm, and evaluating the performance of the model using cross-validation and other metrics. The model deployment component involves deploying the trained model to a production environment using a Flask API.

The key features and benefits of the proposed architecture include its scalability, modularity, and flexibility, as well as its ability to handle large volumes of data and support a variety of machine learning algorithms. The expected performance of the system is a high accuracy rate and fast prediction times, and the system is designed to be easily maintained and updated as new data becomes available. Overall, the proposed architecture is expected to provide a reliable and efficient solution for predicting customer churn and supporting customer retention efforts.

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 'Prediction Bank Credit Risk Using South German Credit Data '. 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.

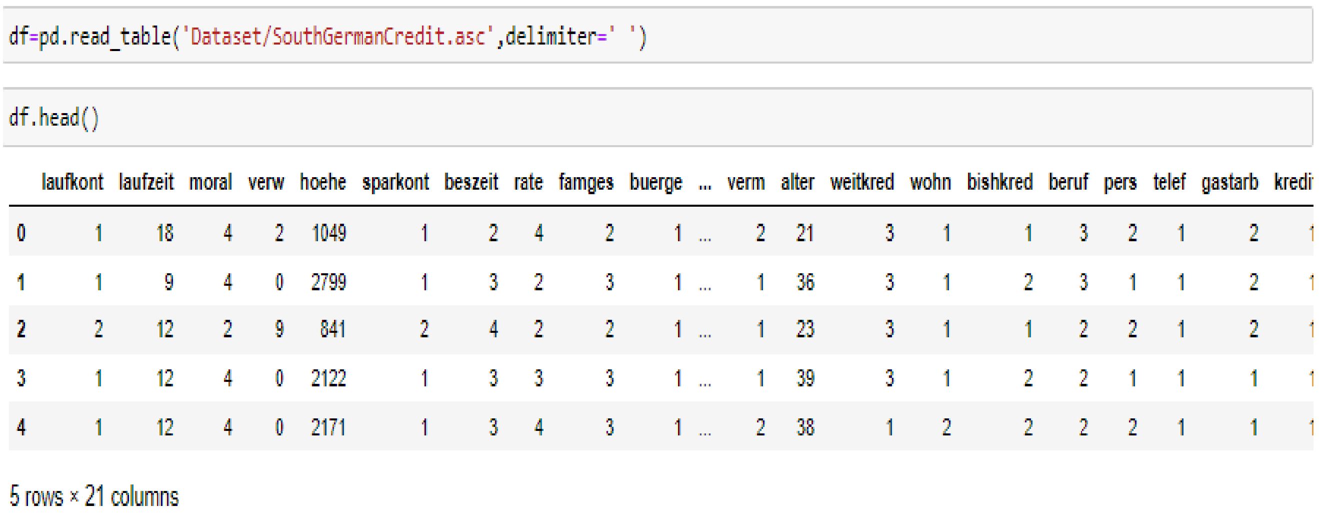
1.3 Constraints

We predict the expected estimating cost of expenses customers based on some personal health information.

2. Technical Specification

2.1 Dataset

The dataset containing verified historical data, consisting of the aforementioned information which classifies people described by a set of attributes as good or bad credit risks. There are 1000 entries in the dataset, which have been sampled as train and test data. This is an adaptation of the UCI South German Credit Dataset. The objective is to find a way to classify the good and bad credit risk identified as 1 and 0 respectively using values in the other feature columns like their status, duration, credit history, purpose, amount, savings, employment duration, installment rate, personal status sex, other debtors, present residence, property, age, other installment plans, housing, number credits, job, people liable, telephone, foreign worker, credit risk. The dataset looks like as follow:



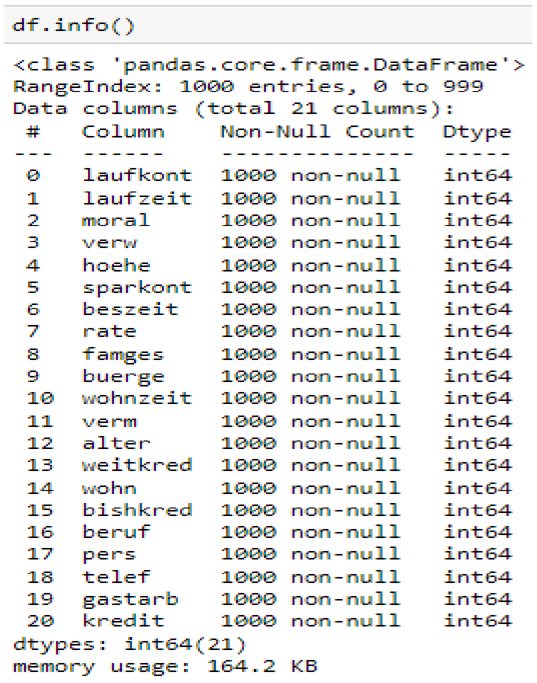
```
df=pd.read_table('Dataset/SouthGermanCredit.asc',delimiter=' ')
```

```
df.head()
```

	laufkont	laufzeit	moral	verw	hoehe	sparkont	beszeit	rate	famges	buerge	...	verm	alter	weitkred	wohn	bishkred	beruf	pers	telef	gastarb	kredi
0	1	18	4	2	1049	1	2	4	2	1	...	2	21	3	1	1	3	2	1	2	1
1	1	9	4	0	2799	1	3	2	3	1	...	1	36	3	1	2	3	1	1	2	1
2	2	12	2	9	841	2	4	2	2	1	...	1	23	3	1	1	2	2	1	2	1
3	1	12	4	0	2122	1	3	3	3	1	...	1	39	3	1	2	2	1	1	1	1
4	1	12	4	0	2171	1	3	4	3	1	...	2	38	1	2	2	2	2	1	1	1

5 rows × 21 columns

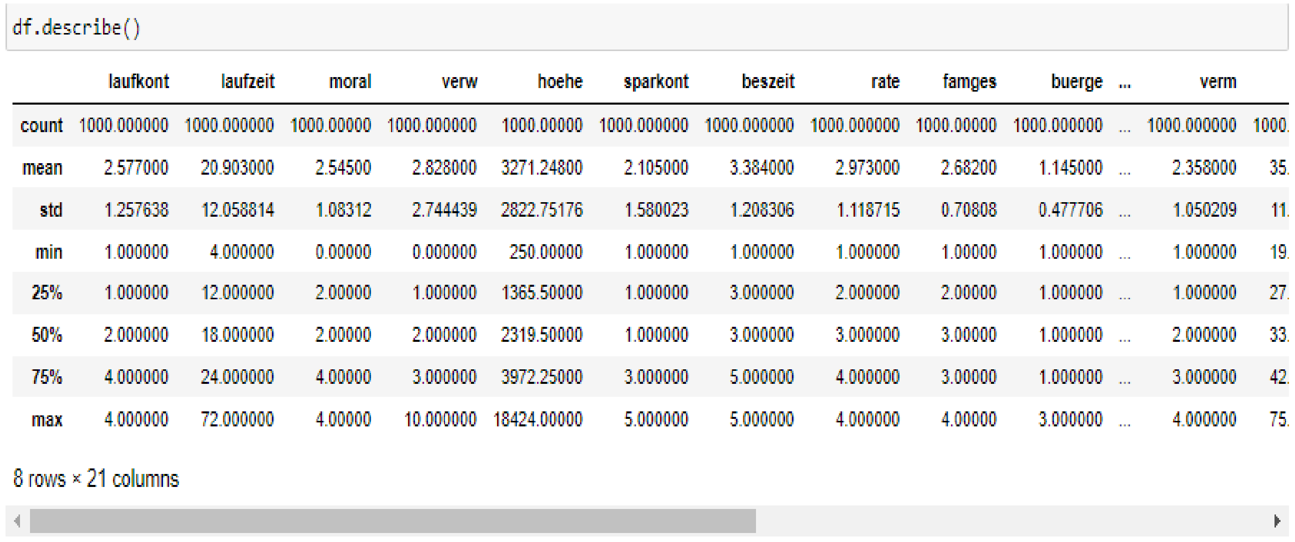
The data set consists of various data types from integer to floating to object as shown in Fig



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 21 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   laufkont  1000 non-null   int64
 1   laufzeit  1000 non-null   int64
 2   moral     1000 non-null   int64
 3   verw      1000 non-null   int64
 4   hoehe     1000 non-null   int64
 5   sparkont  1000 non-null   int64
 6   beszeit   1000 non-null   int64
 7   rate      1000 non-null   int64
 8   famges    1000 non-null   int64
 9   buerge    1000 non-null   int64
 10  wohnzeit  1000 non-null   int64
 11  verm      1000 non-null   int64
 12  alter     1000 non-null   int64
 13  weitkred  1000 non-null   int64
 14  wohn      1000 non-null   int64
 15  bishkred  1000 non-null   int64
 16  beruf     1000 non-null   int64
 17  pers      1000 non-null   int64
 18  telef     1000 non-null   int64
 19  gastarb   1000 non-null   int64
 20  kredit    1000 non-null   int64
dtypes: int64(21)
memory usage: 164.2 KB
```

Various factors important by statistical means like mean, standard deviation, median, count of values and maximum value, etc. are shown below for numerical attributes



```
df.describe()
```

	laufkont	laufzeit	moral	verw	hoehe	sparkont	beszeit	rate	famges	buerge	...	verm	
count	1000.000000	1000.000000	1000.00000	1000.000000	1000.00000	1000.000000	1000.000000	1000.000000	1000.00000	1000.000000	...	1000.000000	1000.
mean	2.577000	20.903000	2.54500	2.828000	3271.24800	2.105000	3.384000	2.973000	2.68200	1.145000	...	2.358000	35.
std	1.257638	12.058814	1.08312	2.744439	2822.75176	1.580023	1.208306	1.118715	0.70808	0.477706	...	1.050209	11.
min	1.000000	4.000000	0.00000	0.000000	250.00000	1.000000	1.000000	1.000000	1.00000	1.000000	...	1.000000	19.
25%	1.000000	12.000000	2.00000	1.000000	1365.50000	1.000000	3.000000	2.000000	2.00000	1.000000	...	1.000000	27.
50%	2.000000	18.000000	2.00000	2.000000	2319.50000	1.000000	3.000000	3.000000	3.00000	1.000000	...	2.000000	33.
75%	4.000000	24.000000	4.00000	3.000000	3972.25000	3.000000	5.000000	4.000000	3.00000	1.000000	...	3.000000	42.
max	4.000000	72.000000	4.00000	10.000000	18424.00000	5.000000	5.000000	4.000000	4.00000	3.000000	...	4.000000	75.

8 rows × 21 columns

Pre-processing of this dataset includes doing analysis on the independent variables like checking for null values in each column and then replacing 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 tells 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, plays an important factor in deciding which value to be chosen at 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 at which step logging require.
- The system should be able to log each and every system flow.
- The system should be 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 deployment of the project,

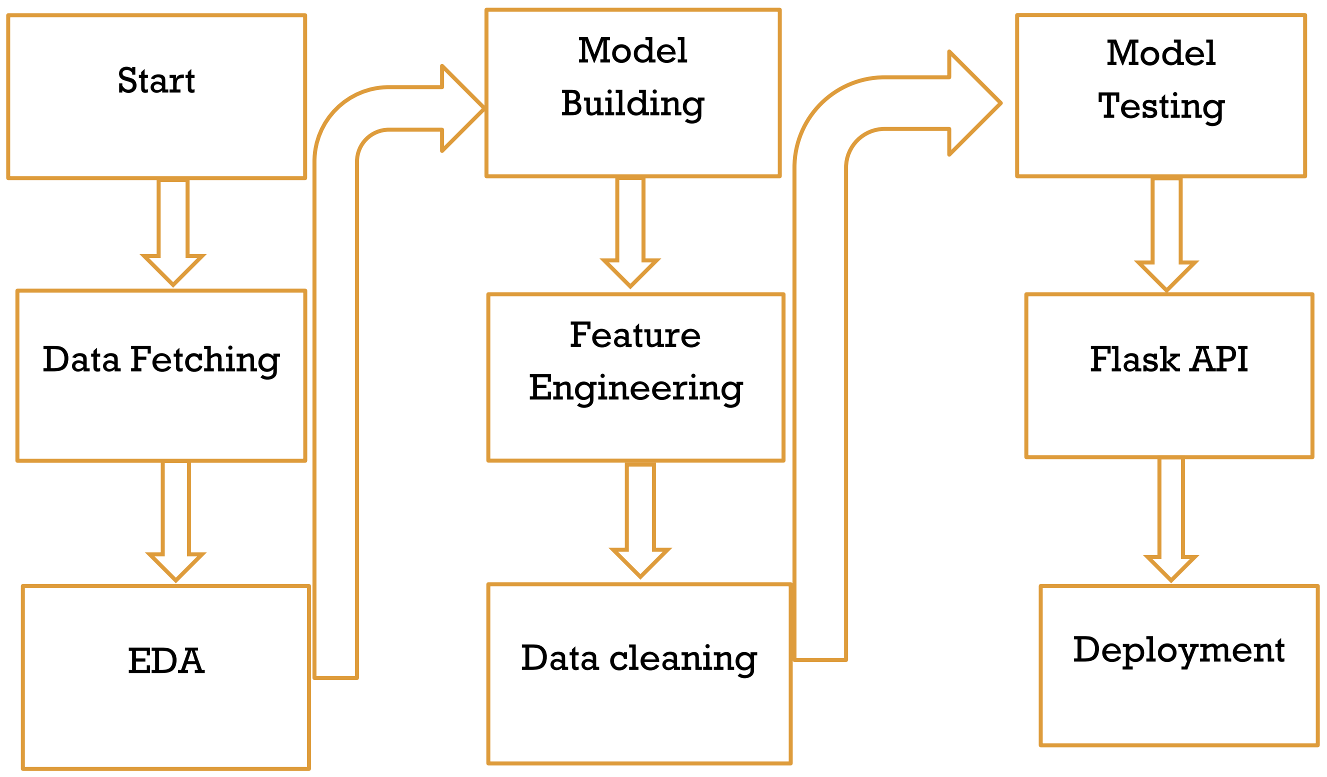
3. Technology Stack

Front End	Html
Backend	Python/Flask
Deployment	

4. Proposed Solution

The solution of the this problem statement if perform EDA on the dataset to generate meaningful insights from the data and use this data to hyper tune with appropriate machine learning algorithms which will have the maximum accuracy in predicting the credit risk. Thus creating a user interface where a user can put in the various features of the data which will in return give the credit risk is present or not.

5. Architecture



5.1 Data Gathering

Data source: https://archive.ics.uci.edu/ml/machine-learning-databases/00573/SouthGermanCredit.zip Dataset is stored in asc 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 the Good and Bad credit risk i.e. 1 and 0 respectively.

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, relevant features are selected and engineered to improve the performance of the machine learning model. This can involve techniques such as feature selection, dimensionality reduction, and feature scaling. Some of the features were showing skewness and even after removing the skewness using log transformation 4 columns were dropped. The outliers were also removed by assigning upper limit values to outliers above the limit and similarly lower limit values to outliers below the lower limit.

5.5 Model Building

After doing all kinds of pre-processing operations mention above and performing scaling and encoding, the data set is passed through a pipeline to all the mode Bagging, Random Forest, Gradient boost, Xgboost, Adaboost Classifier algorithms . It was found that

5.6 Model Saving

Model is saved using pickle library in pickle format.

5.7 Flask API 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 bank credit risk good or bad i.e. 1 or 0 respectively, this is performed in this stage.

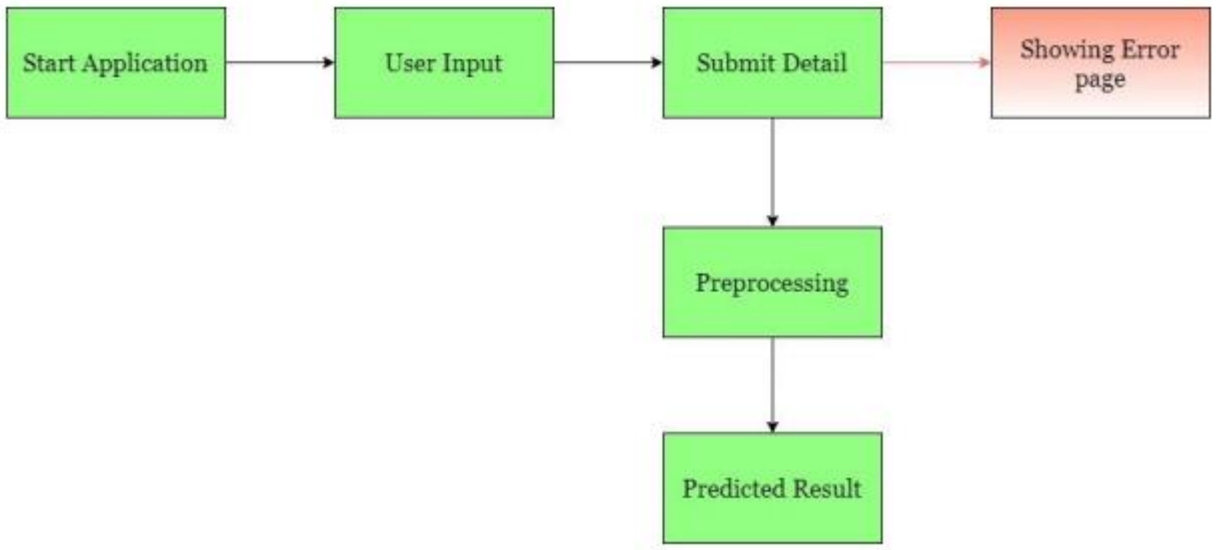
5.8 GitHub

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

5.9 Deployment

The project was deployed from GitHub into .

6. User Input / Output Workflow


Start Application
User Input
Submit Detail
Showing Error page
Preprocessing
Predicted Result