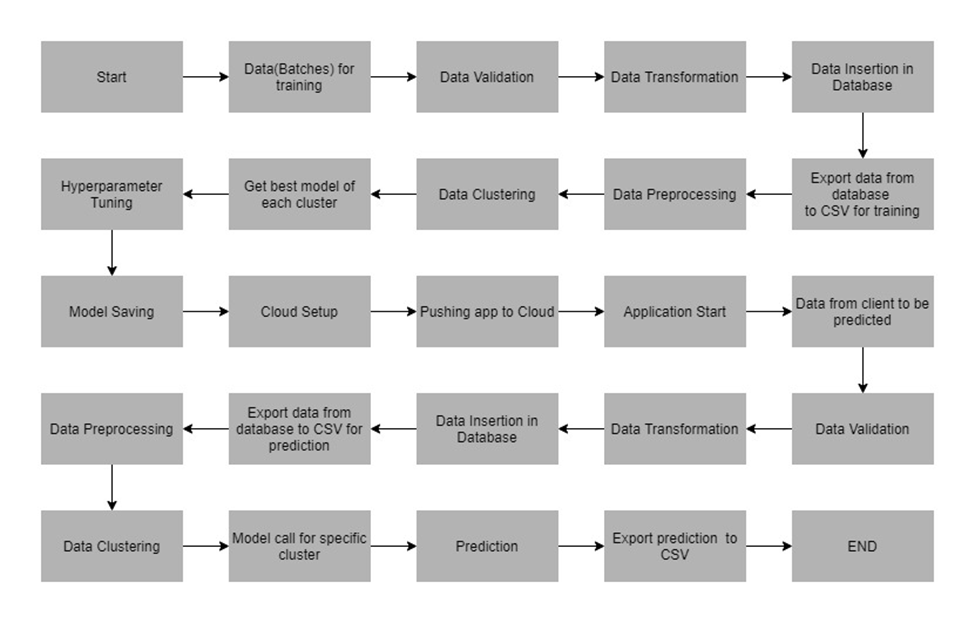
**Problem Statement**

build a classification methodology to determine whether Loan default prediction status non deliquent, deliquent.

**Architecture**



**Data Description**

This dataset, I got from analyticsvidhya.com and data store in batch file in this project. The data has been extracted from the census bureau.

The data contains the following attributes:

**Features:**

1. **loan\_id**: Unique loan ID
2. **source**: Loan origination channel
3. **financial\_institution** : Nameof the bank
4. **interest\_rate**: Loan interest rate
5. **unpaid\_principal\_bal**: Loan unpaid principal balance
6. **loan\_term**: Loan term (in days)
7. **origination\_date** : Loan origination date (YYYY-MM-DD)
8. **first\_payment\_date**: First installment payment date
9. **loan\_to\_value**: Loan to value ratio
10. **number\_of\_borrowers** : Number of borrowers
11. **debt\_to\_income\_ratio**: Debt-to-income ratio
12. **borrower\_credit\_score**: Borrower credit score
13. **loan\_purpose**: Loan purpose
14. **insurance\_percent** : Loan Amount percent covered by insurance
15. **co-borrower\_credit\_score**: Co-borrower credit score
16. **insurance\_type** : 0 - Premium paid by borrower, 1 - Premium paid by Lender
17. **m1 to m12** : Month-wise loan performance (delinquency in months)

**Target Label:**

Whether the claim is loan deliquency status or not

1. **m13** : target, loan delinquency status (0 = non deli delinquent, 1 = delinquent)

Apart from training files, we also require a "schema" file from the client, which contains all the relevant information about the training files such as: Name of the files, Length of Date value in FileName, Length of Time value in FileName, Number of Columns, Name of the Columns, and their datatype.

**Data Validation**

In this step, we perform different sets of validation on the given set of training files.

1. Name Validation- We validate the name of the files based on the given name in the schema file. We have created a regex pattern as per the name given in the schema file to use for validation. After validating the pattern in the name, we check for the length of date in the file name as well as the length of time in the file name. If all the values are as per requirement, we move such files to "Good\_Data\_Folder" else we move such files to "Bad\_Data\_Folder."

2. Number of Columns - We validate the number of columns present in the files, and if it doesn't match with the value given in the schema file, then the file is moved to "Bad\_Data\_Folder."

3. Name of Columns - The name of the columns is validated and should be the same as given in the schema file. If not, then the file is moved to "Bad\_Data\_Folder".

4. The datatype of columns - The datatype of columns is given in the schema file. It is validated when we insert the files into Database. If the datatype is wrong, then the file is moved to "Bad\_Data\_Folder".

5. Null values in columns - If any of the columns in a file have all the values as NULL or missing, we discard such a file and move it to "Bad\_Data\_Folder".

**Data Insertion in Database**

1. Database Creation and connection - Create a database with the given name passed. If the database has already been created, open a connection to the database.
2. Collection creation in the database - Collection with name - "Good\_Data", is created in the database for inserting the files in the "Good\_Data\_Folder" based on given column names and datatype in the schema file. If the collection is already present, then the new collection is not created, and new files are inserted in the already present table as we want training to be done on new as well as old training files.
3. Insertion of files in the collection - All the files in the "Good\_Data\_Folder" are inserted in the above-created collection. If any file has invalid data type in any of the columns, the file is not loaded in the table and is moved to "Bad\_Data\_Folder".

**Model Training**

1. **Data Export from Db -** The data in a stored database is exported as a CSV file to be used for model training.
2. **Data Preprocessing**
3. Drop the columns not required for prediction.
4. Check for null values in the columns ,hear null value not present
5. Replace and encode the categorical values with numeric values.
6. Scale the numeric values using the standard scaler.
7. **Clustering –** KMeans algorithm is used to create clusters in the preprocessed data. The optimum number of clusters is selected by plotting the elbow plot, and for the dynamic selection of the number of clusters, we are using "KneeLocator" function. The idea behind clustering is to implement different al model The Kmeans is trained over preprocessed data, and the model is saved for further use in prediction gorithms
8. **Model Selection -** After the clusters have been created, we find the best model for each cluster. We are using two algorithms, “SVM” and "XGBoost". For each cluster, both the algorithms are passed with the best parameters derived from GridSearch. We calculate the AUC scores for both models and select the model with the best score. Similarly, the model is selected for each cluster. All the models for every cluster are saved for use in prediction.

**Prediction Data Description**

Prediction dataset, I got from analyticsvidhya.com and data store predication file details of all loans for which the participants are to submit the delinquency status - 0/1 (not probability) .

Apart from prediction files, we also require a "schema" file from the client, which contains all the relevant information about the training files such as:

Name of the files, Length of Date value in FileName, Length of Time value in FileName, Number of Columns, Name of the Columns and their datatype.

**Data Validation**

In this step, we perform different sets of validation on the given set of training files.

1. **Name Validation -** We validate the name of the files based on given Name in the schema file. We have created a regex pattern as per the name given in the schema file, to use for validation. After validating the pattern in the name, we check for the length of date in the file name as well as the length of the timestamp in the file name. If all the values are as per requirement, we move such files to "Good\_Data\_Folder" else we move such files to "Bad\_Data\_Folder".
2. **Number of Columns -** We validate the number of columns present in the files, and if it doesn't match with the value given in the schema file, then the file is moved to "Bad\_Data\_Folder".
3. **Name of Columns -** The name of the columns is validated and should be same as given in the schema file. If not, then the file is moved to "Bad\_Data\_Folder".
4. **Datatype of columns –** The datatype of columns is given in the schema file. This is validated when we insert the files into Database. If the datatype is incorrect, then the file is moved to "Bad\_Data\_Folder".
5. **Null values in columns –** If any of the columns in a file has all the values as NULL or missing, we discard such file and move it to "Bad\_Data\_Folder".

**Data Insertion in Database**

1. **Database Creation and connection -** Create a database with the given name passed. If the database is already created, open the connection to the database
2. **Collection creation in the database –** Collection with name - "Good\_Data", is created in the database for inserting the files in the "Good\_Data\_Folder" based on given column names and datatype in the schema file. If the collection is already present, then a new table is not created, and new files are inserted into the already present collection as we want training to be done on new as well old training files.
3. **Insertion of files in the collection -** All the files in the "Good\_Data\_Folder" are inserted in the above-created collection. If any file has invalid data type in any of the columns, the file is not loaded in the collection and is moved to "Bad\_Data\_Folder".

**Prediction**

1. **Data Export from Db -** The data in the stored database is exported as a CSV file to be used for prediction.
2. **Data** **Preprocessing** :
3. Drop the columns not required for prediction.
4. Check for null values in the columns. If present, impute the null values using the categorical imputer.
5. Replace and encode the categorical values with numeric values.
6. Scale the numeric values using the standard scaler.
7. **Clustering** - KMeans model created during training is loaded, and clusters for the preprocessed prediction data is predicted.
8. **Prediction** - Based on the cluster number, the respective model is loaded and is used to predict the data for that cluster.
9. Once the prediction is made for all the clusters, the predictions along with the Wafer names are saved in a CSV file at a given location, and the location is returned to the client.

**Now let’s look at the project folder structure:**

