

# US VISA APPROVAL

## Machine Learning Project



# Overview

**This project presents a ML based solution using ML Algorithm to predict which visa applications will be approved and thus recommend a suitable profile for applicants whose visa have a high chance of approval.**

**Github-link**



## **Objective:**

- **Develop a predictive model to determine the likelihood of a US visa application's approval.**
- **Provide actionable insights to visa applicants and consultants based on historical data.**

## **Problem Statement:**

- **Visa approval processes can be complex and time-consuming.**  
**Understanding the factors influencing approval can help streamline preparation.**

## **Key Technologies & Tools:**

- **Machine Learning: Python, Scikit-learn, XGBoost**
- **Data Management: MongoDB**
- **Cloud & DevOps: AWS (EC2, ECR), Docker, GitHub Actions**
- **Web Framework: Flask (for the web interface)**

# Data Collection & Preprocessing

## Data Sources:

- **Publicly available datasets from government and immigration websites.**
  - **Historical visa application data, including applicant demographics, education, job information, and visa type.**

# Data Collection Methods:

- **Web scraping and API integrations.**
  - **Data ingestion pipeline using Python scripts and MongoDB for storage.**

# Data Validation & Cleaning:

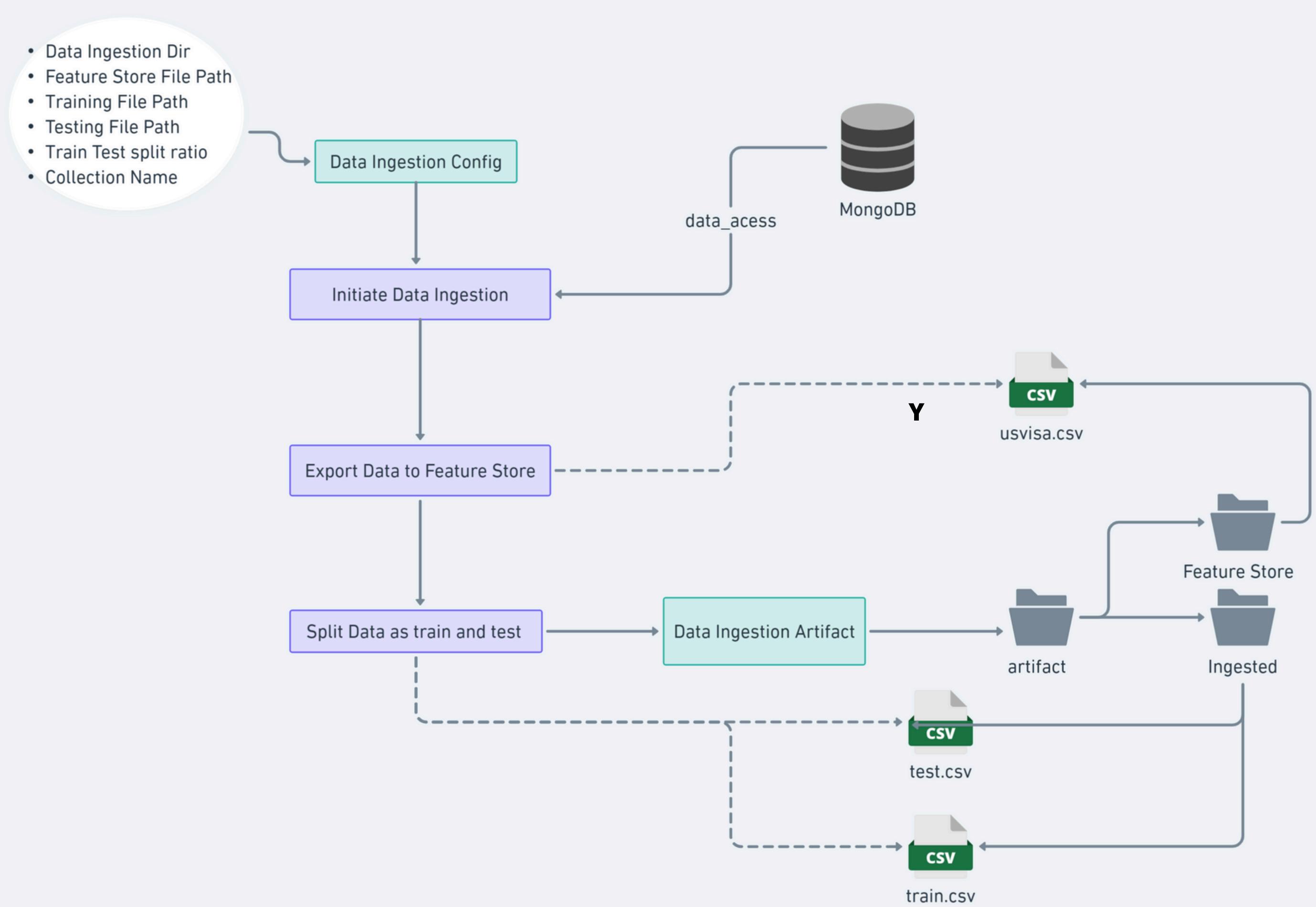
- Removal of duplicates and null values.
  - Standardization of data formats (e.g., dates, categorical variables).
  - Handling of missing values through imputation or removal.

# Feature Engineering:

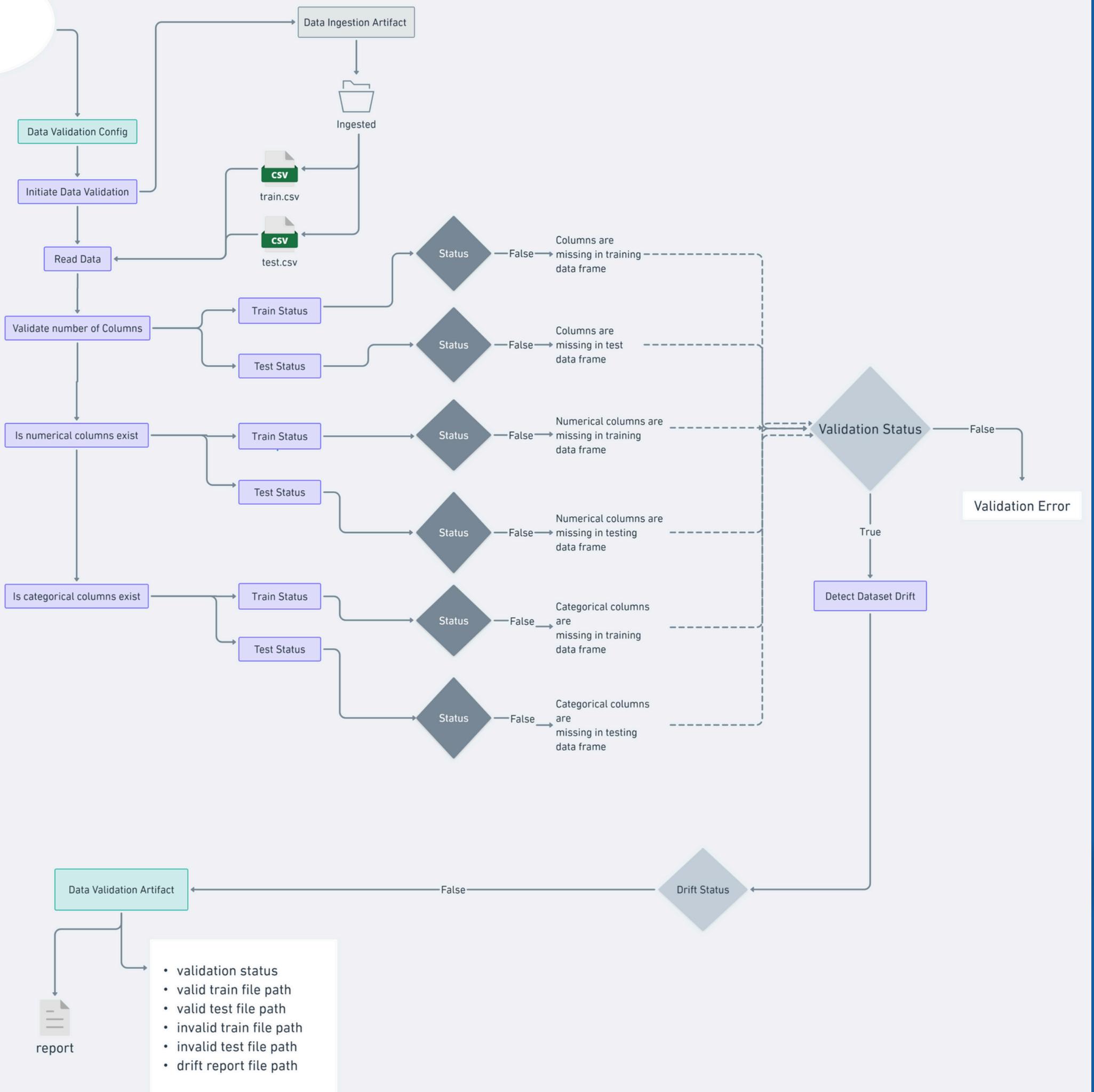
- **Creation of new features such as "years of experience" or "visa category."**
  - **Encoding of categorical features (one-hot encoding, label encoding).**
  - **Normalization and scaling of numerical features.**



- Data is ingested by exporting visa application data from MongoDB into a DataFrame, saving it in a feature store, and splitting it into training and testing datasets for model development.



- Data Validation Dir
- Valid Data Dir
- Invalid Data Dir
- Valid train file path
- Invalid train file path
- Invalid test file path
- Drift report file path



# Data is validated by checking for missing columns, ensuring schema compliance, and detecting data drift between training and testing datasets. This process ensures data integrity and consistency for model development.

# Model Training and Evaluation

## Model Selection:

- Initially tested multiple algorithms, including Logistic Regression, Decision Trees, Random Forest, and XGBoost.
- Chose the Random Forest model based on its interpretability, accuracy, and performance.

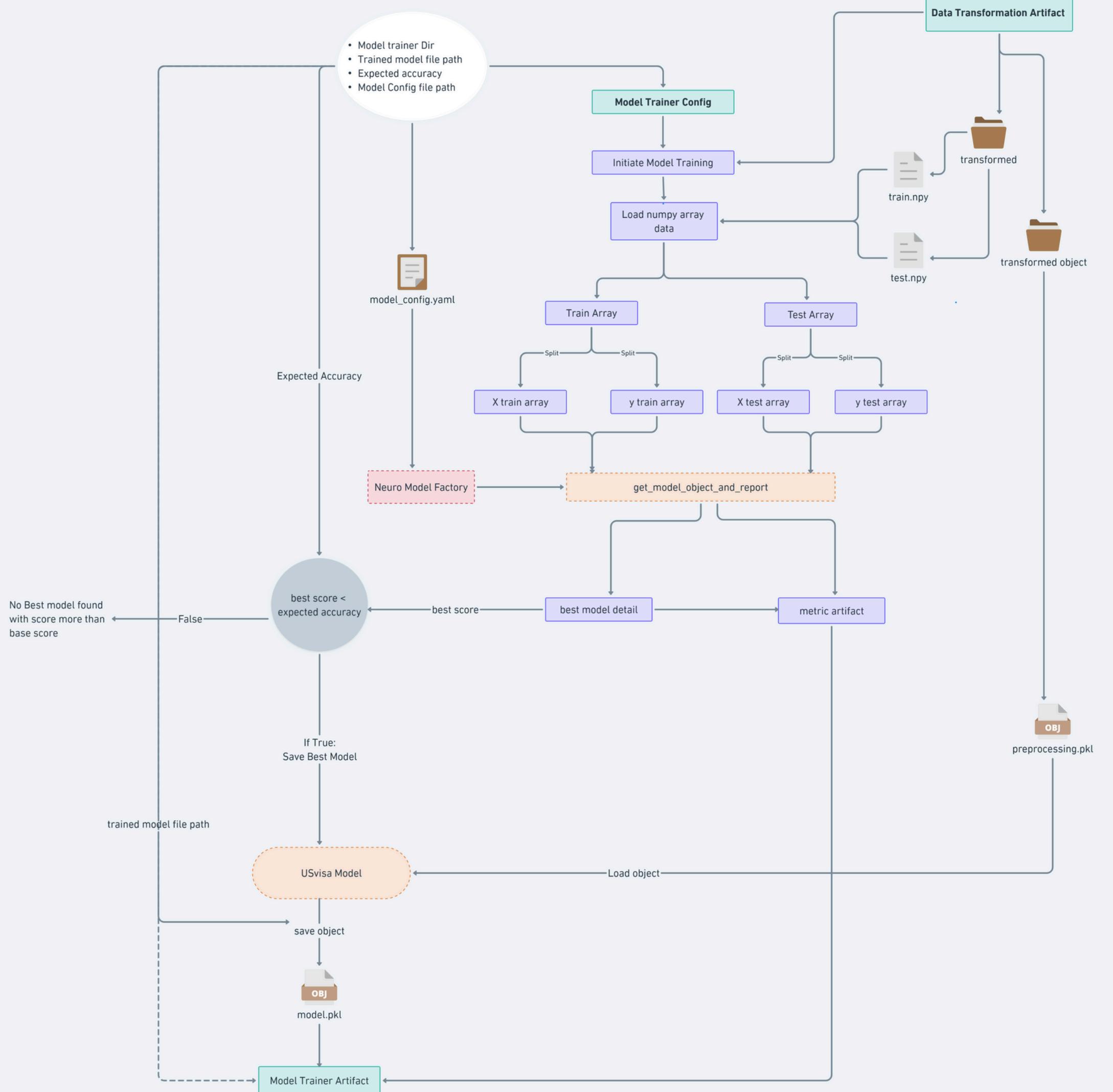
## Training and Hyperparameter Tuning:

- Split the data into training and testing sets (80/20 split) and performed hyperparameter tuning using GridSearchCV.
- Employed cross-validation to ensure the model's generalizability.

## Evaluation Metrics:

- Evaluated models using metrics like accuracy, precision, recall, and F1 score. The Random Forest model achieved the best results, with an accuracy of 85%.





**The model is trained by using neuro\_mf to select the best model based on expected accuracy. The selected model is evaluated on test data using metrics like accuracy, precision, recall, and F1 score.**

# Model Deployment

## Dockerization:

- **Created a Dockerfile to package the application and its dependencies into a Docker image, ensuring consistent environments across different stages (development, testing, production).**

## CI/CD Pipeline:

- **Set up a continuous integration and deployment (CI/CD) pipeline using GitHub Actions. This automated the testing and deployment process, ensuring that changes were quickly and reliably deployed to production.**

→ C [github.com/Tanujkumar24/US\\_VISA\\_APPROVAL\\_ML\\_PROJECT/actions/runs/9594994698](https://github.com/Tanujkumar24/US_VISA_APPROVAL_ML_PROJECT/actions/runs/9594994698)

← Deploy Application Docker Image to EC2 instance

**doc\_updated #6** [Re-run all jobs](#)

**Summary**

Triggered via push 8 minutes ago Status Success Total duration 5m 3s Artifacts

Tanujkumar24 pushed `-o 92424fd main`

Jobs

Continuous-Integration

Continuous-Deployment

Run details

Usage

Workflow file

aws.yaml

on: push

```
graph LR; A[Continuous-Integration 3m 21s] --> B[Continuous-Deployment 1m 23s]
```

# continuous integration and deployment (CI/CD)

## AWS Infrastructure Setup:

- **Elastic Container Registry (ECR): Used for secure storage of Docker images.**
- **EC2 Instances: Hosted the application on virtual machines.**
- **IAM Roles and Policies: Managed access and permissions for EC2 and ECR.**

The screenshot shows the AWS Elastic Container Registry (ECR) interface. On the left, there's a sidebar with navigation links for 'Private registry' (Repositories, Summary, Images, Permissions, Lifecycle Policy, Repository tags, Settings) and 'Public registry' (Repositories, Settings). The main content area is titled 'Amazon Elastic Container Registry' and shows a breadcrumb path: Amazon ECR > Private registry > Repositories > us\_visa. A blue banner at the top states: 'Image scan overview, status, and full vulnerabilities has moved to the Image detail page. To access, click an image tag.' Below this, the repository name 'us\_visa' is displayed with tabs for 'REPOSITORY' and 'Investment'. There are buttons for 'View push commands' and 'Edit'. Under the 'Images' section, there's a table with one entry:

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest
latest	Image	June 20, 2024, 14:59:28 (UTC+05:5)	1343.33	<input type="button" value="Copy URI"/>	sha256:3b91cdb0c3c460...

# Deployment Process:

1. Built the Docker image and pushed it to ECR.
2. Launched an EC2 instance and pulled the image from ECR.
3. Deployed the Docker container on the EC2 instance.

Final model

The screenshot shows the Amazon S3 console interface. On the left, a sidebar lists various AWS services: Buckets, Access Grants, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Block Public Access settings for this account, Storage Lens, Dashboards, Storage Lens groups, and AWS Organizations settings. The main area displays the contents of the 'usvisa-model-24' bucket. The 'Objects' tab is selected, showing one object named 'model.pkl'. The object details are as follows:

Name	Type	Last modified	Size	Storage class
model.pkl	pkl	June 20, 2024, 00:43:35 (UTC+05:30)	2.6 MB	Standard

# entered the values

The screenshot shows a web-based form for predicting visa status. The form consists of several input fields and a central processing area.

**Input Fields:**

- Requires Job Training: N
- Number of Employees: 14500
- Select Region: Northeast
- Prevailing Wage: 601
- Contract Tenure: Hour
- Full or Part Time: Y
- Age of the company: 15 (with up and down arrows)

**Buttons:**

- Predict Visa-Status (blue button)
- Activate Windows (gray text)

## Visa Prediction Status: Renderin

continent

Select Continent

Education of employee

Select Education

Has Job Experience

Has Experience

Requires Job Training

Required Training

Number of Employees

select between 15000 to 40000

Select Region

Region

Prevailing Wage

select between 700 to 7000

Contract Tenure

Contract Tenure

Full or Part Time

Full Time

Age of the company

select between 10 to 180

Predict Visa-Status

# Result

**Visa Prediction Status: Visa-approved**

A  
G