# Bank Marketing Prediction System - Final Report

## 1. Introduction

This report presents the development and deployment of a machine learning-based prediction system using the Bank Marketing Dataset. The main goal is to predict whether a client will subscribe to a term deposit based on various features such as job, education, housing status, and more. The solution involves data preprocessing, model training, cloud deployment, and CI/CD pipeline automation.

---

## 2. Dataset Overview

The dataset was obtained from the UCI Machine Learning Repository and contains 4521 records and 17 attributes, including the target variable `y`. Key features include:

- Demographic: age, job, marital, education
- Financial: default, balance, housing loan, personal loan
- Contact: contact type, last contact date/duration
- Social/economic indicators: employment variation rate, consumer price/confidence indices, euribor rate, number of employees

The target is binary: `y = yes` (subscribed) or `no` (not subscribed).

---

## 3. Data Preprocessing

- **Missing Values:** No true NaNs, but several 'unknown' values in categorical columns like `job`, `education`, etc. were filtered.
- **Encoding:** Used `pd.get_dummies()` for one-hot encoding of categorical features.
- **Scaling:** StandardScaler was applied to numerical features to normalize values.

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

---

## 4. Model Training & Evaluation

### Models Used:

- Logistic Regression
- Support Vector Machine (SVM)

### Results:

Logistic Regression slightly outperformed SVM in terms of F1-score and recall for predicting positive class (`y = yes`).

| Metric | SVM | Logistic Regression |
|---|---|---|
| Accuracy | 89% | 90% |
| Recall (True) | 22% | 34% |
| F1-Score (True) | 0.32 | 0.43 |

---

## 5. Flask Application & Cloud Deployment

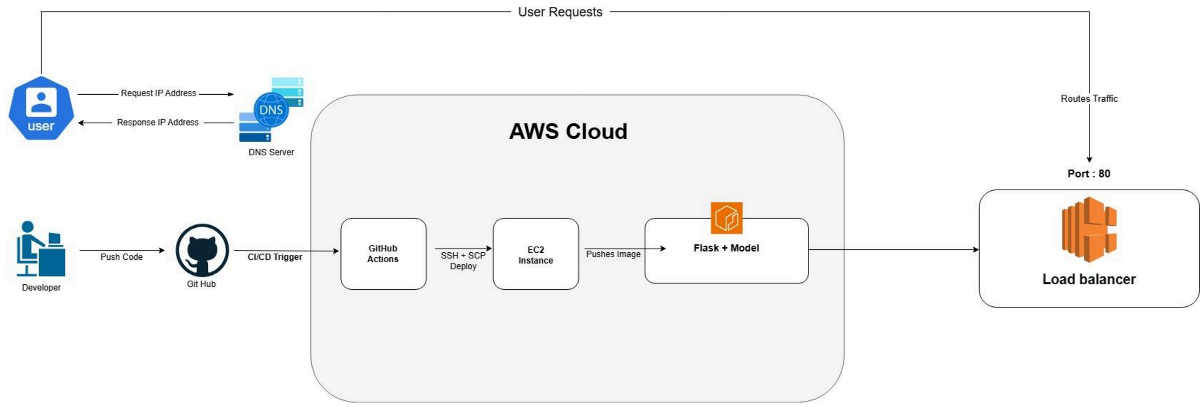
A Flask web application was created that receives user input (20 features), applies scaling, runs predictions using the trained logistic regression model, and returns whether the client is likely to subscribe.

### Hosting

- Platform: **AWS EC2 (Ubuntu 22.04)**
- Flask app listens on port 5000
- Endpoint: `/predict` (POST)

### Architecture Diagram:

Deployment Architecture Diagram



## 6. CI/CD Pipeline

The deployment pipeline uses GitHub Actions to automate delivery:

### Workflow:

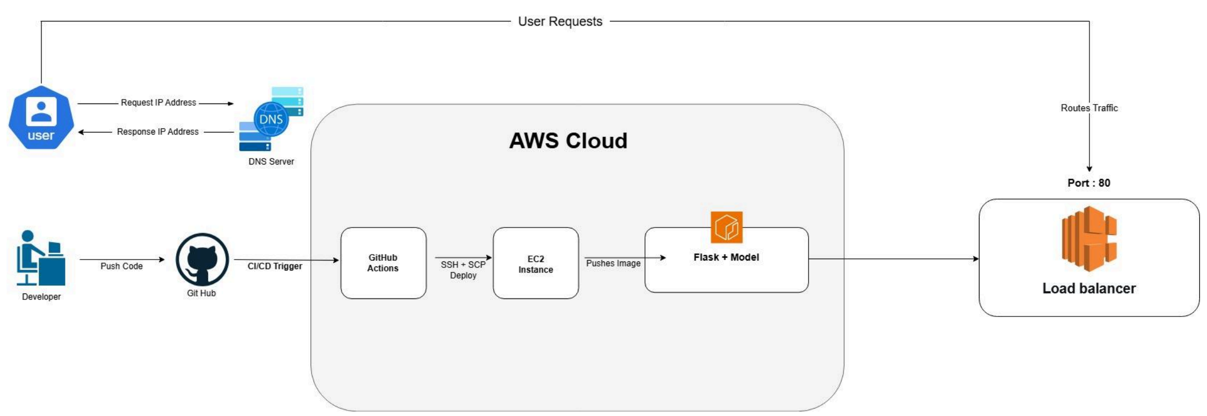
1. Developer pushes code to GitHub `main` branch
2. GitHub Actions is triggered
3. SSH + SCP used to transfer files to EC2
4. Flask server is restarted with new changes using `nohup`

### CI/CD Diagram:

## 7. Final Deployment Architecture

A complete deployment architecture includes DNS resolution, GitHub CI/CD triggers, and routing via a load balancer.

**Deployment Architecture Diagram**



GITHUB REPO - https://github.com/theek23/banking-ml-coursework

---

# 8. Conclusion

This project demonstrates a full machine learning pipeline from data preprocessing to cloud deployment and automation. While Logistic Regression gave promising results, performance could be further improved with model tuning and class balancing techniques. The deployment is fully automated and scalable.

---

# 9. References

- UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/bank+marketing
- Flask Documentation: https://flask.palletsprojects.com/
- GitHub Actions: https://docs.github.com/en/actions
- AWS EC2: https://aws.amazon.com/ec2/

---

*Prepared by: Theekshana De Silva*