# Assignment Discussion

DAY-MM-YYYY

# Agenda

- Background Information & Problem Statement
- Data Source from Kaggle
- High Level Requirements & Assumptions
- Our Approach
- Architecture
- Development
  - ➢ Code walkthrough
  - ➢ Exploratory Data Analysis
  - ➢ Modelling
  - ➢ Experiment Results discussion
  - ➢ Tuning
- Deployment
  - ➢ Application Pipeline
  - ➢ Test data set
  - ➢ Web Application demo
- Next steps

## Background Information & Problem Statement

### Generic Information:

A telecom major company  name – “XXX” has a goal of increasing the recurring revenue from customers and reducing the customer attrition to enhance the bottom line.

Leadership team has identified 3 areas  to meet the above goal

- Identify the reason for customer churn
- Predict the customers who might get churn
- Send them targeted offers

Evaluation  Metrics

- 10% reduction of customer churn  in FY21-22

## Summary

“Build a customer churn prediction model to using machine learning"

Key Points:

- Provide the list of customers to marketing team
- Suggest the focus area

## Data Source

Open Source Dataset

- ➢ Downloaded from Kaggle

Key Points of dataset:

- ➢ The raw data contains 7043 rows (customers) and 21 columns (features).
- ➢ Target column is – Churn

## High Level Requirements & Assumptions

Requirements

- ➢ Suggest the most influencing feature for the churn
- ➢ Build a prediction model with an accuracy of over and above 75% on average, 85% - Good and 90% - efficient
- ➢ Provide customer list on a web page
- ➢ Solution needs to be deployed on cloud

Assumptions

- ➢ This is valid only for retail customers
- ➢ Only voluntary churn is in scope
- ➢ User of the model - Marketing team
- ➢ Model runs on demand

Deliverables

1. Experiment Results sheet
2. Solution pipeline deployed on cloud
3. Web page for user
4. Code

# Our Approach -Research Phase Development

Research Phase Development

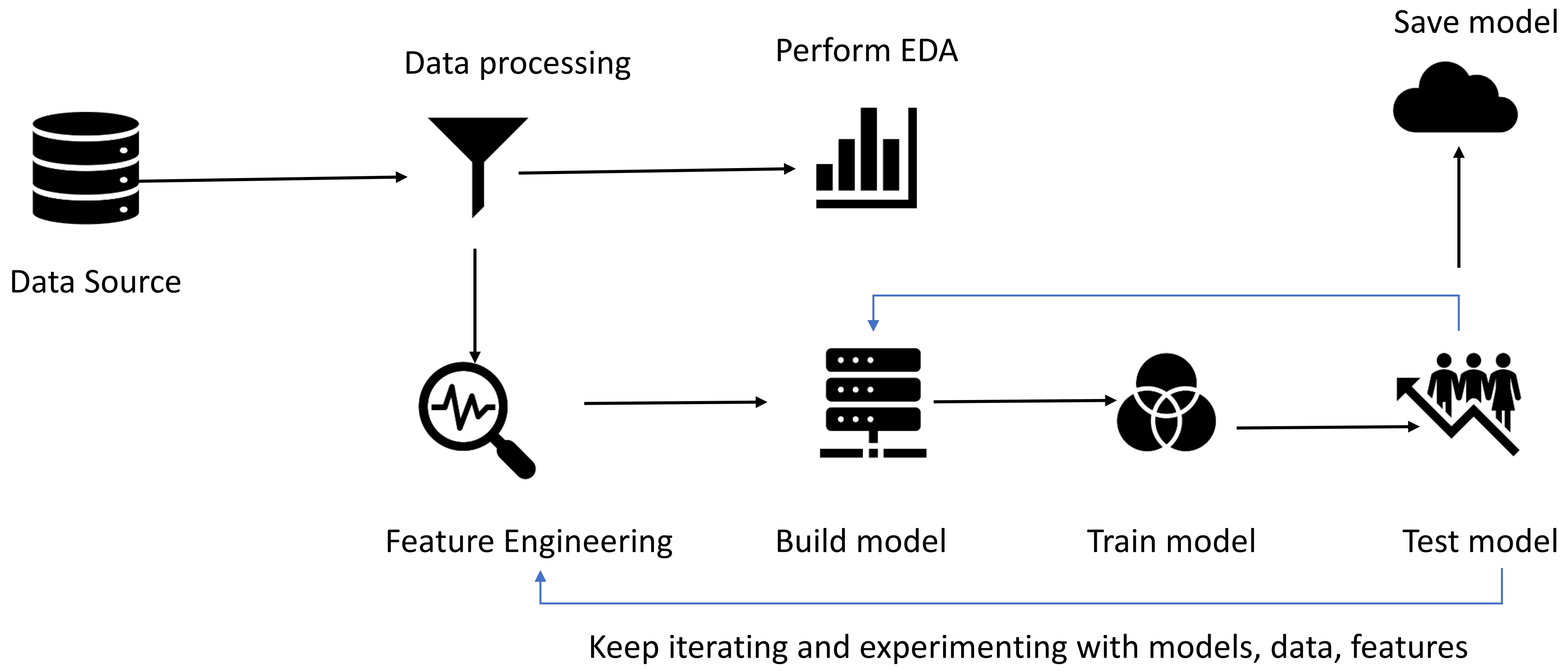
1. Understand dataset
2. Data Processing
3. Perform EDA(Exploratory Data Analysis)
4. Feature Engineering
5. Build a baseline model
6. Train and validate the model with 80:20 train test split
7. Validate the accuracy
8. Enhance the performance
9. Log the experiment results
10. Choose the best model
11. Save the model.

## Our Approach – Production Ready

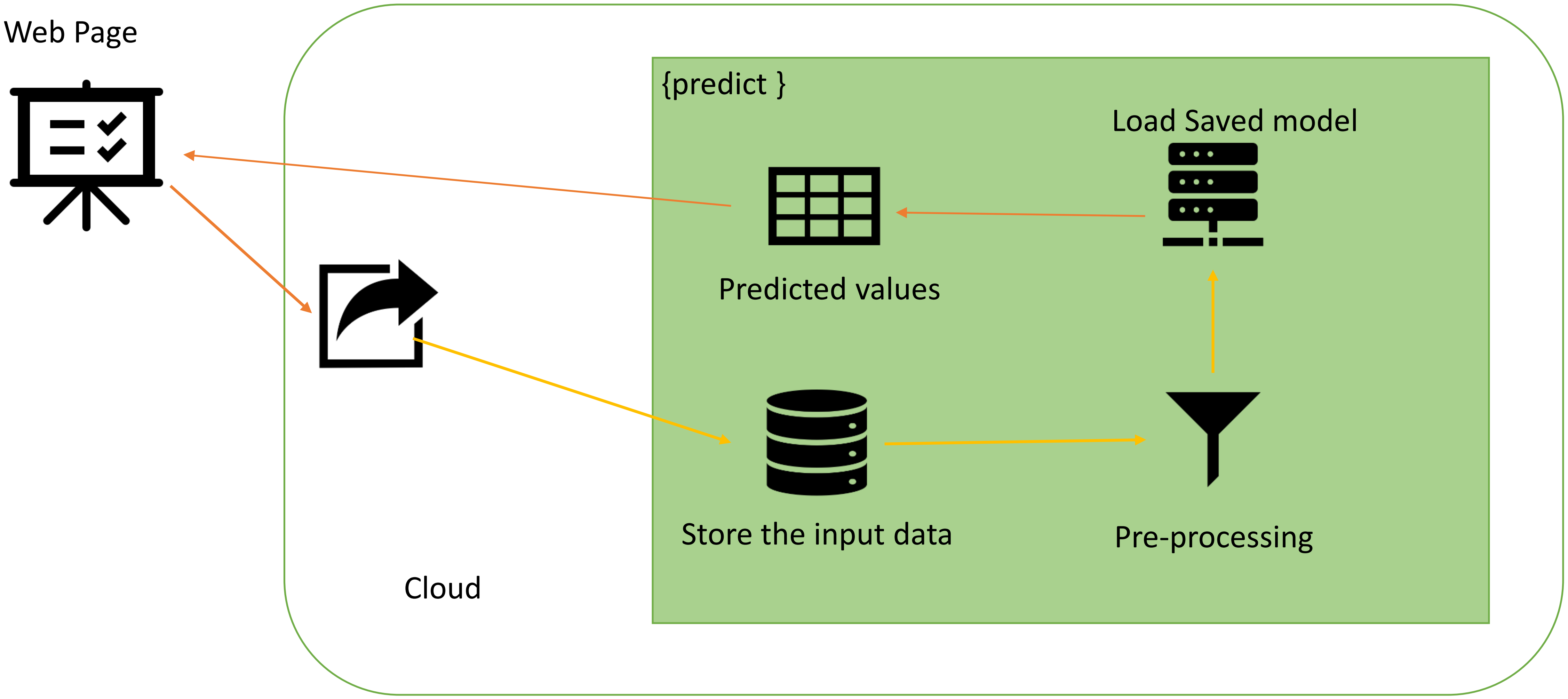
Readiness for deployment of model

1. Build a pipeline
2. Load the saved model
3. Build a Flask API
4. Expose the API to web page
5. Deploy the pipeline on Google cloud
6. Test the model with input
7. Log the results

# Model Development Architecture



# Production Architecture



## *Development-Notebook*

- Code walkthrough
- Exploratory Data Analysis
- Modelling
- Experiment Results discussion
- Tuning/model selection

## *Deployment*

- Application Pipeline
- Test data set
- Web Application demo

## Next Steps

- Automate the email notification with the predicted dataset
- Schedule the model training
- Implement solution version control for better development
  - GIT
  - Jenkins
- Automate the data ingestion
- Keep monitoring model performance
- Add interpretability of the models.
  - It will be good for random testing
  - Transparency
  - Algorithm - SHAPE or LIME can be tried