Project Overview

OBJECTIVE

1. Design complete solution to demonstrate end-to-end pipeline development and manage an AIML project
2. Develop understanding of all stages of a machine learning project lifecycle
3. Demonstrate understanding of challenges encountered during the project development and provide ways to tackle them
4. Showcase understanding of software engineering best practices while developing the project

REFERENCES

https://www.sciencedirect.com/science/article/pii/S0306457316306471#sec0007
https://link.springer.com/chapter/10.1007/978-3-642-36973-5_47
https://ieeexplore.ieee.org/document/8929515

PROBLEM STATEMENT

Develop an application which takes a webpage as an input and predicts its genre and credibility score.
With more than a billion websites present over the web constituting to trillion web pages used by 4+ Billion internet users, and thousands more getting added on a daily basis, Moreover, this information can be misleading and incorrect, which can have serious consequences for people who increasingly rely on web sources for information such as security, health, academia etc.
Develop a tool for automated genre-aware credibility assessment of web pages to extract web page features instances and identify genre a web page belongs to while assessing it's Genre Credibility Score. Develop a data pipeline which includes the all the following stages of Machine Learning Project Life Cycle –

1. Data Ingestion
2. Data Preparation
3. Model Training
4. Model Deployment
5. Model Prediction

SETUP DETAILS

Stages are elaborated as follows –

1. DATA COLLECTION AND INGESTION
   Project Name - data-ingestion-service
   The objective of this project is to source new data to re-train the model.
   Ingestion could be done either as follows –
   a) From the given genre labeled url dataset, scrape the text of the webpage and generate relevant features for classification of the genre.

b) Use web scraping libraries (For e.g. by using BeautifulSoup to scrape urls data)
c) Use the given features for predicting the credibility score

Data Ref:-
https://data.world/credibilitycoalition/basic-november2018
https://sourceforge.net/projects/reconcile2011/
https://rawgit.com/s8811/reconcile-tags/master/describe_data.nb.html

- **Data Cleaning** - Since different web pages could have text in different formats, transform the API response for all the web pages to the same raw format.
- **Scheduling** - Schedule storing data of new urls using a process scheduler (every few mins/ hours), or with custom thread management code.
- **Data Storage** – Use a Publisher – Subscriber model to collect data and push it to the store. Create a kafka topic and use it to queue cleaned responses from all sources. The sink for the stream should be MySQL/ MongoDB.

2. DATA PREPARATION (PROPROCESING)

Project Name – data preparation-service

1. The objective of this project is to prepare the data for building a classification model which will classify the genre of the web page.
    a. Perform data cleaning and preprocessing
    b. Perform feature engineering techniques and remove unwanted features.
2. Prepare the data for building a regression model which will predict the genre credibility score.
    c. Perform data cleaning and preprocessing
    d. Perform feature engineering techniques and remove unwanted features.
    e. Normalize the feature values

3. Model Training

Project Name – model-training-service

a. Train a classification model for predicting the genre of the given webpage url.
b. Train a regression model for predicting the genre credibility of the given webpage url

4. Model Prediction

Project Name – model-prediction-service

A separate classifier project and prediction project picks up the trained models either from a location or from the model registry, and exposes it for prediction in following mode –

a. Real Time REST (flask) API – allow user to enter the url of a webpage and returns the classified genre category and the genre credibility score of the webpage.

User Interface - A simple Streamlit app or a basic HTML page containing a form can be exposed in this project.

1. It takes the url of the webpage, calls the flask API for prediction, and shows the predicted classification.
2. Once the genre is classified, calls the flask API for prediction, and shows the predicted genre credibility score.

INFRASTRUCTURE/ DEPLOYMENT/ ADDITIONAL GUIDELINES

1. Docker images may be created for all the 3 projects. These images can then be used for deployment as containers.
2. Deployment can be orchestrated by using docker-compose
3. Flask APIs (wherever required) should use Gunicorn/ Bjoern/ CherryPy as a WSGI server.

ARCHITECTURE/ DESIGN EXPLAINABILITY

An important aspect of the ML software development pipeline is explainability. Use UML (Unified Modelling Language) to explain the project architecture and design. Following diagrams are expected –

1. Use Case Diagram – identify the actors in the system, and associated actions
2. Activity Diagram – prepare logical workflows which illustrate the flow of control in the system.
3. Sequence Diagram – prepare sequence of interactions with the system.
4. Component/ Deployment Diagram – diagram to show how components have been organized

ML TOOLING

Following tools to be used for project setup –

1. PyCharm or Jupyter as Python IDE
2. PlantUML for UML diagrams documentation – install it as a plugin in PyCharm
3. Virtual Environment – use venv or virtualenvwrapper to setup separate environments for all projects described
4. MySQL/ MongoDB as datastore
5. PySpark for stream processing
6. POSTMAN for testing Flask APIs
7. Apache Zookeeper + Kafka for message queue/ streams
8. MLFlow for model versioning + hyper-parameters versioning

MILESTONES

There would be 4 milestones –

1) Week 1 –
   a) Participants are expected to complete documentation and architectural design by creating the UML diagrams.
   b) Prepare the local environment with the right tools and installations

    c) Ingest data source and prepare data-ingestion-service which populates “raw_data”

2) Week 2 –
    a) Setup the model-training-service project
    b) Setup appropriate connectors to load data from MongoDB/ MySQL into PySpark RDD.
    c) Understand cleaning + preprocessing steps necessary to transform the raw data and complete data preparation step.
3) Week 3 –
    a) Segregate the preprocessed data into training and test
    b) complete model training and deployment (and model registry)
    c) Convert the project to On-Demand service, to be able to retrain the model as needed. This could be a flask API which could trigger the entire pipeline in model-training-service (loading data into PySpark, preprocessing, model training)
    d) Serialize the re-trained model.
    e) Push this model to model-registry (MLFlow) along with hyperparameters
4) Week 4 –
    a) Expose model via model-prediction-service in the form of flask API.
    b) Dockerize all the projects by adding appropriate Dockerfile
    c) Prepare a simple HTML page containing a form to take an article as input and print the predicted category.

TEAM

Project manager: Rahul Gupta

1. Data collection- Nischay Agarwal, DontiReddy Sai Rakesh Reddy
2. Data preprocessing - Agamjot Singh, Rohit Singh Parihar, Sathyapriya Srinivasan
3. Front end - website - Sidhanth Sanil, Prakash Rankawat, Y Vijay Kumar Reddy
4. Backend - Surya Teja K, Sumeet Mohanty, Praveen Samavedam
5. AIML module - Babloo Bandi, S N Sai Teja, Vineet Sajwan, Harsh S Rahamatkar, Mayur Mahesh Pujari
6. Interpretability - Radha Atul Kumbhare, Sandeep KM
7. Security - Sagar, Dolly Sulochana Rani Talasila
8. Testing - Archana Pradhan, Gunnu Sirohi, Irfan khan MA
9. Integration & Deployment - Garima Badhan, Padala Naresh, Sindhu Tirth Sahoo, Partha Pratim Samadder, Arukala Himateja
10. Documentation- Shoumik Pravin Kulkarni, Sanju Gawade, Vadapalli Venkata Sree Harsha