Abhijit Darekar

Abhijit Darekar | LinkedIn
Abhijit Darekar | GitHub

Email: jeetdarekar@icloud.com
Phone: +91 9108221098

Professional Summary

Accomplished Machine Learning Practitioner with a 3-year track record of achieving results through advanced analytics and state-of-the-art machine learning techniques. Specialized in Machine Learning and Natural Language Processing. Adept at implementing machine learning algorithms and utilizing Deep learning frameworks like PyTorch, HuggingFace, and TensorFlow. Proficient in data preprocessing, feature engineering, and model optimization. Experienced in deploying solutions on AWS and Azure Cloud services, developing CI/CD pipelines such as AWS ECR, GitHub Actions and Sage maker. Seeking ways to contribute in open-source projects and an expert in Kaggle Notebooks.

Skills Summary

- **Languages:** Python, Bash-Scripting, PowerShell-Scripting, Java, Dart (Beginner).
- **Frameworks:** Scikit-learn, TensorFlow, PyTorch, Pandas, NumPy, MlFlow, LangChain, HuggingFace, OpenAI, Ollama, H2O, DagsHub. FastApi, NLTK, Flutter (Beginner), FastApi, Flask Api.
- **Cloud/Services:** AWS SageMaker, AWS CI/CD Components, GitHub Actions, VertexAi, Kubeflow, Docker, Azure CI/CD Components, AWS Lambda Functions.
- **OS/Platforms:** PyCharm, Anaconda, VSCode, Windows, Linux.
- **Version Control:** Git/GitHub, BeanStalk, DVC, MlFlow**.**
- **Software Skills:** Predictive Modelling, Deep Learning Models, Generative AI, Large Language Models (LLM), Deployment/Front End, Monitoring.
- **Soft Skills:** Rapport Building, Stakeholder Management, Excellent Communication, Team Collaboration.

Work Experience

Machine Learning Developer | Skycliff IT, *Bangalore* **Jan 2023 – *present***

DocumentGPT – RAG with OpenAI and LangChain

- Designed a sophisticated **DocumentGPT**, **RAG** Based Application using the LangChain and **OpenAI** Models.
- Loaded PDFs related to Patient Claims, extracted table(s) using **Camelot Framework**, and generated embeddings using embedding model form **HuggingFace** and stored in **Qdrant** vector database.
- Expanded the capabilities of **LangChain** by integrating models, enhancing DocumentGPT in conjunction with the vector database.
- significant efficiency gains, saving 70% of time and manpower for the Support/Billing/Transcription team in addressing queries related to claims, billing, and prescriptions.
- Designed Web-app using **Flask framework** for end users.
- Containerized web-app in **Docker** for Continuous Deployment in AWS ECR.

Message Suggestion – Statistical Modelling

- Motivation: The aim of this project/feature is to suggest text message to user while typing. Helping user(s) by cutting time on repetitive writing, reducing chances of spelling or grammatical errors.
- Data Collection and Preprocessing: Gathered data from movie dataset, cleaned data by removing sentences containing special characters, handled contractions using **regex**. Added necessary tokens, created **ngrams** for training for modelling.
- Design: Designed and evaluated multiple models (**Deep learning** and **Statistical Models**), Implemented **Kneser-Ney Smoothing** with **Back-off Mechanism** for Next Word Prediction for better latency when compared with **LSTM model**. Experimented with **Add-1 Smoothing Technique**.
- Deployment: Helped App team to convert code to Dart from Python, with Initial 10k rows for initial suggestion.
- Post Deployment: Guided app team to generate ngrams on user-specific textual data for personification.

Claim Amount Prediction – Predictive Modelling

- Objective: Develop a **Predictive Algorithm** to predict the insurance claims for patients undergoing specific medical procedures (e.g., surgery, pre-checkup, follow-up visits) within a hospital setting.
- Data Preprocessing: Gathered and processed data spanning the last three years, collaborating closely with domain experts to identify and validate crucial features.

- Model Development: Employed **MlFlow** for continuous model refinement, ensuring the registration and tracking of essential hyperparameters and metrics.
- Model Deployment: Implemented model architecture on Amazon Web Services for seamless inference, incorporating model registry, feature registry, hyperparameter tracking, and metrics using Amazon SageMaker.
- Retraining Workflow: Established a streamlined process for model retraining, capturing metrics and loss at regular intervals to enhance predictive accuracy**.**

Python Developer | TCS, *Bangalore* **Sep 2020 – Nov 2022**

ETL Job Run Dashboard (Time Series Analysis)

- Objective: Implemented a comprehensive solution to monitor, alert, analyse, and predict job run times daily. The primary aim was to proactively notify the operations team about potential failures and late runs.
- Data Processing: **Collaborated closely with the Ops** team to collect and analyse 90 days' worth of data. Methodically removed monthly and manually triggered ETL jobs for accurate analysis.
- Model Development: Engineered multiple **LSTM models** to forecast critical ETL jobs, tuning the hyperparameters for better **MAD** and **MAPE**.
- Integrated **MlFlow** for efficient **Model Tracking**, **Parameter Monitoring**, and **Metric Analysis** and **Model Registering**.
- Model Testing and Deployment: Designed **Flask-Web** app dashboard for local deployment, facilitating performance evaluation.
- Deployment and Retraining: Deployed models on a **Linux server** for real-time inference. Designed and tested a real-time retraining workflow, ensuring the model adapts to evolving trends in new data every month.

Projects

Student Marks Prediction – CI/CD on AWS

GitHub Link

- Goal of Project is to develop a Prediction Model using **Scikit-learn**, **Containerize**, and deploy the ML Model in **AWS ECR** using **GitHub Actions** and Trigger Configured on **AWS EC2 Instance**.
- Designed and Deployed an Rest Api using **FastApi Framework**.
- Added **MLFlow** for Parameters, Model tracking Locally.

MINST Hand Written Number identification – CI on Azure

- Designed a Deep learning model in **PyTorch** with necessary Data augmentation required for Image classification.
- Implemented project in Pythonic style. Used **GitHub Version Control**, Created **FastApi** for Real-time inference.
- Deployed on **Azure Web Apps** with connecting Web-App with GitHub Repo for **Continuous Integration**.

Fine-Tuning Flan-T5 – Hugging Face

- Fine Tuned **Flan-T5-base** on Multiple Prompts., used **HuggingFace Trainer** to fine-tune the pretrained T5 model, fine-tuned model and evaluated the inference on **CPU** and **GPU**.
- Implemented **Summarization** , **Equation to Latex**, English to Hindi,Kannada prompts.
- Designed API(s) for Realtime inference.

Quantization of Llama2 in Local

- Applied **QLoRA** using **HuggingFace PEFT** library for fine-tuning of Llama 2 locally.
- Trained Models with interchanging multiple **Hyperparameters** in **QLoRA** adapter.

Education

BE in Computer Science and Engineering @SKSVMACET, Laxmeshwar. 2016 - 2020
10+2 in Kendriya Vidyalaya No 1, Hubli 2014 - 2016

Hobbies

- Sketching
- Bike Riding, Exploring new places and Food.
- Reading Books
- Trekking