Kushal Adhyaru

+1-408-748-6342 | kushalatulbhai.adhyaru@sjsu.edu | linkedin.com/in/kushal-adhyaru | github.com/kushal511
San Jose, California - 95113, USA

Education

San Jose State University — California, USA
Master of Science in Data Science with a specialization in Data Engineering; GPA: 3.5/4.0 — *August 2024 - May 2026*

Indus Institute of Technology & Engineering — Gujarat, India
Bachelor of Technology in Computer Science; GPA: 4.0/4.0 — *June 2019 - May 2023*

Skills

Programming: Python, SQL, Java, C++, HTML, CSS, JavaScript

Data Analysis & Visualization: Pandas, NumPy, Matplotlib, Seaborn, Power BI, Tableau, MS Office Suite (Excel, PowerPoint, Word); Statistical Analysis, Hypothesis Testing, Regression Analysis

Machine Learning, Deep Learning & NLP/LLMs: scikit-learn (modeling), MLflow (experiment tracking), TensorFlow/PyTorch (deep learning), NLTK/Hugging Face Transformers (NLP), LangChain (RAG), Flask (web development).

Big Data, Databases & Cloud: Hadoop/Spark (distributed computing), ETL (Prefect, Dagster), MySQL (Relational Database), MongoDB (NoSQL Database), AWS (cloud deployment)

Experience

Software Development Engineer Intern — May 2025 – August 2025
Amazon — *Seattle, USA*

- Implemented an end-to-end **search query classification** system to fetch and normalize **50,000 search queries**, applied **LLM prompt templates** via a Python script to categorize search queries into distinct categories, and stored search queries along with their classifications in **Amazon S3**.
- Fetched search queries and their classifications from Amazon S3, cached them in memory, and executed category-specific queries for precise, classification-aligned results.
- Built a prefix-based **search-query autocomplete system**: generated autocomplete suggestions from search-query prefixes, filtered noisy/misspelled terms with **LLM** prompts, and persisted the filtered search queries to Amazon S3.
- Loaded catalog titles and filtered search queries into a **Trie** and ranked autocomplete suggestions by a **weighted relevance score** combining popularity and search-query frequency, serving higher-scoring suggestions first.

Data Science Intern — January 2024 – April 2024
Innomatics Research Labs — *Hyderabad, India*

- Engineered a system to analyze and classify over **8,500 product reviews**, leveraging **Prefect** for ETL pipeline automation and scheduling.
- Trained sentiment analysis models using **BoW, TF-IDF, Word2Vec, and BERT**, achieving an **F1-Score of 0.92**, while utilizing **MLflow** for model management and experiment tracking.
- Developed and deployed a sentiment analysis web application on **AWS**, enabling real-time customer feedback insights.
- Constructed a **Generative AI Code Review application** using the **OpenAI API**, reducing code review time by **40%** with accurate bug detection and optimization suggestions.

Projects

SkinShade AI — January 2025 – March 2025

- Built a skin-tone classification and color recommendation system using **MTCNN** for face detection, **HSV**-based skin segmentation, and **K-Means** to extract dominant skin color; categorized tones into **Light/Medium/Dark** and utilized the **CelebA** dataset during development.
- Deployed with **TensorFlow Lite** and a **Gradio** UI; combined **OpenCV** image processing, **scikit-learn K-Means** to find the dominant skin color, and **PyTorch MTCNN** for face detection to generate personalized **5-color** palettes.

Semantic Search Engine — March 2024 – April 2024

- Spearheaded the development of a semantic search engine using **BERT-based SentenceTransformers** to vectorize movie subtitles and user queries, improving retrieval accuracy for 30,000 subtitles.
- Applied comprehensive data preprocessing and document chunking; used **cosine similarity** for more accurate matches; managed **ChromaDB** embeddings to speed up data retrieval.

RAG System on "Leave No Context Behind" Research Paper — January 2024 – February 2024

- Designed a Retrieval-Augmented Generation (RAG) system for the 'Leave No Context Behind' paper using **LangChain** and **Google Gemini 1.5 Pro**. Processed documents with **PyPDFLoader**, chunked and embedded them, and stored in **ChromaDB** for efficient retrieval.
- Enhanced **Large Language Model** performance by integrating external data to improve context-awareness, optimizing the generation of accurate, contextually relevant responses.