Pramod Krishnachari

Washington, DC | +1 (571) 457-1364 | itskpramod@gmail.com | linkedin.com/in/pramodkrishnachari | github.com/ChariPramod

SUMMARY

AI/ML engineer with an M.S. in Data Science who builds and evaluates production LLM agents, RAG pipelines, and retrieval systems end-to-end. Experienced in Python, PyTorch, FastAPI, vector search, PySpark, and cloud deployment on GCP and AWS, with a track record of shipping evaluation plans, grounding checks, and analytics pipelines that measurably improve model and process quality. Eligible for OPT / STEM OPT.

EDUCATION

George Washington University **Aug 2024 – May 2026**
Master of Science in Data Science *Washington, DC*

Presidency University **Aug 2020 – May 2024**
B.Tech, Computer Science (Artificial Intelligence & Machine Learning) *Bengaluru, India*

TECHNICAL SKILLS

Languages: Python, SQL, R, Bash

ML & Deep Learning: PyTorch, scikit-learn, Hugging Face Transformers, SBERT, SciBERT, BERTopic, BM25, FAISS, Qdrant, knowledge distillation, ResNet-50, EfficientNet, ViT, PCA, t-SNE, GridSearchCV, cross-validation

LLM & Agents: LangGraph, LangChain, LlamaIndex, RAG, multi-agent orchestration, tool use, prompt design, grounding and hallucination analysis, guardrails, evaluation rubrics, regression testing, scenario-based QA, human-in-the-loop, OpenAI / Anthropic / Gemini APIs, Vertex AI, real-time voice agents

Data Engineering: PySpark, Databricks, Delta tables, BigQuery, PostgreSQL, Redis, Azure Data Lake Storage Gen2, Azure Data Factory, ETL/ELT, schema validation, data quality checks

Cloud & Infra: GCP (Vertex AI, BigQuery, GCS, GCE), AWS (EC2, S3), Docker, Git, GitHub Actions CI/CD, FastAPI, Flask, REST/JSON, WebSockets

Voice & Telephony: Twilio (voice, WhatsApp, Telegram, media streams), Deepgram (STT), Cartesia (TTS), streaming audio, barge-in and interruption handling

BI & Visualization: Power BI, Tableau, Plotly, Streamlit, Matplotlib, Seaborn, Excel

Methods & Testing: A/B testing, hypothesis testing, time series forecasting (ARIMA), clustering (K-Means), anomaly detection, control charts, feature engineering, pytest, evaluation harnesses

Optimization: Gurobi, mixed-integer programming, Monte Carlo simulation

AI Coding Tools: Claude Code, Cursor, Windsurf, Gemini CLI, GitHub Copilot

PROFESSIONAL EXPERIENCE

SuperCX **May 2026 - Present**
AI Engineer *Remote*

- Built and maintained a production real-time voice AI pipeline on Google **Vertex AI**, chaining **Deepgram speech-to-text**, **Gemini** real-time reasoning, and **Cartesia text-to-speech**, and owning the latency and quality trade-offs across every stage of the conversational loop.
- Deployed **Gemini** real-time models through **Vertex AI** for streaming audio processing and in-turn reasoning, handling partial transcripts, **barge-in**, and interruption so the agent responds mid-utterance instead of waiting for end of speech.
- Integrated **Twilio telephony** to reach client customers over standard phone calls, **WhatsApp**, and **Telegram** from a single conversation backend, mapping each channel to shared session state and channel-specific media handling.
- Streamed bidirectional audio over **WebSockets** between **Twilio media streams** and the **Vertex AI** inference path, managing buffering, sample-rate conversion, and reconnection so live calls degraded gracefully rather than dropping.
- Benchmarked competing **speech-to-text**, **LLM**, and **text-to-speech** providers on transcription accuracy, response quality, and **end-to-end latency**, documenting results so provider selection was driven by measured evidence rather than vendor claims.
- Designed structured evaluation plans for conversational agents covering intent handling, fallback behavior, tool-call correctness, and failure modes, turning ad hoc spot checks into a repeatable release gate.
- Iterated system **prompts** and tool definitions using production call analytics, isolating recurring failure patterns and validating each change against a fixed scenario set before rollout.
- Instrumented the pipeline for observability across turn-level timings, channel, and provider responses, making regressions traceable to a specific stage rather than to the conversation as a whole.

The George Washington University School of Business **Aug 2025 - May 2026**
Student Technical Support Assistant II (Part-time) *Washington, DC*

- Supported technology-enabled classrooms through proactive system monitoring and troubleshooting, resolving audiovisual, network, and presentation-system faults before they interrupted scheduled instruction.
- Collaborated with campus IT teams to diagnose and resolve recurring technical issues, escalating with reproducible detail so root causes were addressed rather than repeatedly patched.
- Applied data-driven problem-solving to recurring support tickets, identifying patterns across rooms and equipment types to prioritize the fixes that removed the most downtime.

- Documented setup procedures and known failure modes for classroom technology, reducing time to resolution for other assistants handling the same equipment.
- Contributed to a reliable academic environment across School of Business facilities by maintaining readiness checks ahead of high-traffic teaching blocks.

Suprajit Engineering Limited – Electronics Division **Dec 2023 - Aug 2024**
Data Science & Analytics Intern *Bengaluru, India*

- Built a trustworthy first-pass-yield dataset for digital instrument clusters by joining production orders, serial numbers, test attempts, rework records, firmware versions, fixture IDs, and failure codes in **SQL** and **pandas**, replacing a headline pass rate that concealed unstable process behavior.
- Reconstructed the full test journey for each serial number and standardized reason codes across first test, retest, rework, and final disposition, separating true product defects from communication timeouts, incomplete tests, and equipment-driven repeats.
- Delivered **Power BI** views of yield, retest frequency, and rework by product, line, shift, fixture, and firmware version, giving quality, test, and production teams one shared definition of yield instead of three conflicting ones.
- Engineered features from raw sensor and end-of-line test curves in **Python**, computing smoothness, slope consistency, endpoint error, signal noise, and hysteresis, then applied **control charts** and **anomaly flags** to surface units drifting while still inside specification limits.
- Built fixture health indicators by joining test results with fixture IDs, calibration history, golden-sample measurements, and maintenance events, calculating rolling deviations that distinguished gradual measurement drift from sudden fixture failure.
- Processed larger event and traceability tables with **PySpark** on **Azure Databricks** over **Delta tables**, standardizing join keys, timestamp alignment, and duplicate handling so product genealogy could be reconstructed reliably across **ERP**, **SMT**, flashing, and test systems.
- Negotiated metric definitions directly with production supervisors, quality engineers, and test engineers, submitting every definition for senior review before implementation so the resulting numbers were accepted rather than disputed.

Tequed Labs **Dec 2022 – Apr 2023**
Data Science Intern *Bengaluru, India*

- Built a scalable **PySpark ETL** pipeline processing 50,000+ daily transactions with automated **schema validation**, **deduplication**, and **structured logging** for ongoing data quality monitoring.
- Audited fragmented client datasets spanning 5 sources and standardized schemas, then developed **Python** and **SQL** EDA pipelines with **pandas**, **Matplotlib**, and **Seaborn** that surfaced previously unknown demand segments.
- Cut manual data-cleaning time by roughly 40% by replacing ad hoc spreadsheet steps with reproducible, version-controlled transformation code.
- Built 20+ interactive **Power BI** and **Tableau** dashboards driven by **SQL**-based KPIs, enabling real-time operational tracking across the client portfolio.
- Partnered with non-technical client stakeholders to translate operational pain points into analytics requirements, presenting weekly performance insights and short written update notes summarizing changes.

Maruti Flex Traders LLP **Jun 2022 – Nov 2022**
Data Analyst Intern *Bengaluru, India*

- Built **ARIMA demand forecasting** models on 5,000+ SKUs in **Python** to address chronic inventory inefficiency, reducing excess inventory by 15% across two trading workflows.
- Implemented **K-Means** clustering in **scikit-learn** to segment behavioral profiles across 20,000+ customer records, enabling targeted outreach and improving customer retention by 18%.
- Developed a modular **Python** reporting framework that automated weekly inventory summaries previously assembled by hand, reducing manual reporting effort by 35% and freeing analyst time for deeper analysis.
- Created automated **Tableau** dashboards with KPI tracking across 30+ product lines, presenting inventory forecasts and segment insights to operations managers in weekly reviews.
- Documented pipeline assumptions, edge cases, and refresh steps in a shared internal wiki so the next analyst could reproduce every report without re-deriving the logic.

EXPERIENTIAL LEARNING PROJECTS

The Build Fellowship (Open Avenues Foundation) Build Projects are eight-week experiential learning engagements supervised by an industry project leader, not employment. Role title: Build Student Consultant.

The Build Fellowship (Open Avenues Foundation) **2025 – 2026**
Build Student Consultant *Remote*

AI-Powered Image Retrieval With Vector Databases | project leader: Kamalesh Kalirathinam *Sep 2025 – Nov 2025*

- Built an image retrieval system using **PyTorch** CNN **embeddings** with **FAISS** and **Qdrant vector databases**, applying **PCA** and **t-SNE** for **dimensionality reduction** and embedding quality analysis across 10K+ image **embeddings**.
- Benchmarked three embedding architectures (**ResNet-50**, **EfficientNet**, **ViT**) on **precision@k** and **recall**, tuning hyperparameters via **GridSearchCV** with 5-fold **cross-validation** in **scikit-learn** and improving top-5 accuracy by 25%.
- Ran ablation studies across vector index configurations (IVF versus HNSW) to quantify the search quality against latency trade-off and select the final index.
- Deployed the end-to-end pipeline on **AWS** with **Docker** containerization and **GitHub Actions CI/CD**, achieving 350 ms **p99** retrieval latency.

- Built an interactive **Streamlit** dashboard for visual search demonstration, giving non-technical reviewers a way to inspect retrieval quality directly.

Drone Flight Planner System: Flight Path for Efficient Data Capture | project leader: Ayush Baid *Mar 2026 – Apr 2026*

- Designed an autonomous drone flight path planner that computes efficient routes between waypoints using A* and **Dijkstra graph search** with live **obstacle avoidance**, targeting a 25% reduction in flight time versus naive routing.
- Modeled the flight environment as a weighted graph with dynamic cost updates, allowing the planner to re-route around newly detected obstacles without recomputing the full search space.
- Architected **RESTful APIs** exposing planning, validation, and telemetry endpoints so the planner could be driven programmatically by downstream flight control software.
- Built a **CI/CD** validation pipeline that automatically tests planned routes against safety constraints before deployment to physical hardware, reducing manual QA overhead.
- Wrote **regression tests** covering degenerate cases such as unreachable waypoints, dense obstacle fields, and boundary violations, so safety failures surfaced in CI rather than in flight.

Multi-Objective Investment Portfolio Optimization | project leader: Ce Luo *Jun 2025 – Aug 2025*

- Constructed a portfolio optimizer in **Python** and **Gurobi** selecting the best mix of 15 assets under real-world trading constraints including position limits, trade sizes, and maximum trade counts, delivering a tuned portfolio in 4 trades with an **Information Ratio** of 0.824.
- Formulated the problem as a **mixed-integer program** balancing tracking error, transaction cost, and trade frequency, making the trade-offs explicit rather than hidden inside a single objective.
- Ran **Monte Carlo simulations** across multiple risk-tolerance settings to identify the optimal risk-return balance and stress-test allocations against varied market paths.
- Automated constraint pipelines with reusable functions, cutting model calibration time by 40% and improving the interpretability of the resulting allocations for reviewers.
- Documented model assumptions, data sources, and limitations so results could be reproduced and challenged by other student consultants on the team.

PROJECTS

DockWise AI | *Maritime LLM Agent & RAG Pipeline* | LangGraph, BigQuery, FastAPI, GCP, Plotly **2025 – 2026**

- Built a **LangGraph multi-agent system** over 54.7M **AIS** vessel position records in **BigQuery** with **tool-use orchestration**, letting analysts query multi-year maritime traffic in natural language instead of hand-written **SQL**.
- Implemented **grounding** checks and fallback handling that validate vessel, port, and timestamp references against the underlying tables before an answer is returned, reducing **hallucinated** entities in agent output.
- Wrote evaluation **prompts** and a scoring **rubric** covering factual **grounding**, response tone, and workflow alignment, used to compare agent versions before each release.
- Engineered a multi-year **AIS** ingestion pipeline (2019–2024) for the Port of Los Angeles, loading daily compressed .csv.zst files into **BigQuery** via **GCP** compute and storage with automated cleaning and **deduplication**.
- Computed vessel dwell time distributions, congestion indices, and seasonal arrival patterns, and applied **Kruskal-Wallis hypothesis testing** to identify statistically significant seasonal congestion shifts ($p < 0.01$).
- Served the system through a **FastAPI** backend deployed on **GCP** with interactive **Plotly** dashboards for exploratory review of agent findings.

Scholarly Topic Navigator | *Hybrid Retrieval & Classification* | Python, NLTK, spaCy, FAISS, BM25, BERTopic, Streamlit **2025**

- Processed 22,522 research papers from ArXiv and Semantic Scholar, building an **NLP** preprocessing pipeline with **NLTK** and **spaCy** covering tokenization, stopword removal, stemming, and language detection across 2.7M tokens.
- Engineered a hybrid retrieval system combining **BM25** lexical search with **FAISS** dense search over **Word2Vec**, **SBERT**, and **SciBERT embeddings**, achieving an **MRR** of 0.83 on a held-out query set.
- Benchmarked **LDA** against **BERTopic** for **topic modeling** and selected on measured coherence (0.448 versus 0.420) rather than on qualitative inspection alone.
- Applied **knowledge distillation** from a **BART-MNLI zero-shot** teacher into a lightweight student classifier, matching teacher accuracy at 67% across 12 categories while running approximately 50x faster at inference.
- Added **LIME** explanations and **structured logging** so every classification could be inspected, traced, and reused as a **regression test** case.
- Shipped the full pipeline as a **Streamlit** dashboard, exposing retrieval, topic assignment, and explanation in a single interface.

Provenance Guard | *Content Provenance Scoring Service, CodePath AI201* | Python, Flask, REST, pytest **2026**

- Built a **Flask REST** service that scores submitted text for AI provenance by combining a model signal and a style signal under an explicit weighted rule, returning a per-signal breakdown instead of a single opaque verdict.
- Documented what each signal measures and, critically, what it cannot see, then used those blind spots to construct an **adversarial** attack set that deliberately evaded the scorer.
- Hardened the service against the attack set and added request **rate limiting**, re-running the full suite before and after to quantify which evasions the fix actually closed.
- Designed an appeal workflow allowing flagged submissions to be re-reviewed with recorded justification, keeping human judgment in the loop for contested decisions.

- Wrote a 32-test **pytest regression suite** covering scoring edge cases, malformed payloads, and appeal state transitions, so **rubric** changes could not silently alter prior outcomes.
- Structured scoring as configurable weights so evaluation criteria could be re-tuned without rewriting the service.

FitFindr | *Multi-Tool Planning-Loop Agent, CodePath AI201* | Python, LLM APIs, pytest **2026**

- Built a planning-loop agent over three tools for listing search, outfit suggestion, and result card generation, decomposing user requests into ordered tool calls and re-planning when a call returned nothing usable.
- Specified a tool inventory with typed inputs, explicit return contents, and a defined empty-result contract for every tool, so the agent branched on absent data instead of crashing mid-loop.
- Ran a structured before-and-after evaluation across a fixed query set, assigning pass or fail verdicts, diagnosing each failure to a specific loop step, and shipping one targeted fix measured against the same set.
- Logged full loop traces per run, making it possible to diagnose why a specific plan was chosen rather than inspecting only the final answer.
- Wrote **pytest** coverage targeting agent failure modes including tool selection errors, unbounded planning loops, and empty tool responses.

TakeMeter | *Text Classifier Training & Evaluation, CodePath AI201* | Python, PyTorch, scikit-learn, Jupyter **2026**

- Designed a label taxonomy for community post classification with written definitions, worked examples, and an explicit decision rule for the hardest boundary between adjacent labels.
- Built and hand-labeled the training dataset, then trained a classifier locally and benchmarked it against a **zero-shot** baseline rather than reporting accuracy in isolation.
- Evaluated with a per-class **confusion matrix** and an **inter-annotator agreement** report, using disagreement between labelers to expose where the taxonomy itself was ambiguous.
- Diagnosed misclassifications to specific taxonomy and data issues, applied one targeted improvement, and re-measured on the same held-out set to confirm the gain was real.

The Unofficial Guide | *RAG Pipeline with Evaluation Loop, CodePath AI201* | Python, ChromaDB, sentence-transformers **2026**

- Built a **retrieval-augmented generation** pipeline over a custom document corpus, selecting **chunk size** and overlap from observed document structure rather than defaults and documenting the reasoning behind each value.
- Compared fixed-size, semantic, and recursive **chunking** strategies on retrieval quality for multi-section queries, tracing every retrieved chunk back to its source file and producing function.
- Ran a before-and-after evaluation assigning verdicts to generated answers, diagnosing failures as retrieval versus generation problems, and validating one targeted fix against the same query set.
- Documented remaining failure modes and their causes instead of reporting only the improved runs.

CERTIFICATIONS

CodePath – Applications of AI Engineering (AI201), Summer 2026

The Build Fellowship (Open Avenues Foundation) – AI-Powered Image Retrieval With Vector Databases, Nov 2025; Drone Flight Planner System, Apr 2026; Multi-Objective Investment Portfolio Optimization, Aug 2025

ADDITIONAL

Work Authorization: F-1 OPT, eligible for STEM OPT extension.

Relevant Coursework: Machine Learning, Natural Language Processing, Data Visualization, Introduction to Data Science (A/B testing and hypothesis testing), Optimization, Cloud Computing, Database Systems.