Sarmitha S *Machine Learning Engineer*

Profile

I'm Sarmitha, an Electronics and Instrumentation Engineering graduate from Coimbatore, with a strong interest in Artificial Intelligence and Machine Learning. My journey blends creativity and logic—rooted in childhood passions like drawing and dancing, and evolving into building smart systems with code. I thrive on curiosity, continuous improvement, and meaningful problem-solving. Driven by both art and analytics, I believe in creating with purpose and learning with passion.

Education

- I completed my schooling at Vimal Jyothi Convent Higher Secondary School, Coimbatore, with academic scores of 448/500 in 10th grade, 465/600 in 11th grade, and 548/600 in 12th grade.
- I have completed College in Sri Ramakrishna Engineering College with 9.15 cgpa, with 2021-2023 highest CGPA in my department and received Sri.P.Ramasamy Naidu Memorial Prize.
- I would like to join the role of machine learning engineer, Artificial Intelligence Engineer or Data Engineering

Projects

Chest X-ray Pneumonia Detection using Ensemble Deep Learning & Grad-CAM
Objective
The goal of this project was to develop a robust AI system capable of classifying chest X-ray images into **NORMAL** or **PNEUMONIA** using deep learning. Inspired by the increasing need for **AI-assisted diagnosis in radiology**, the objective was not only to achieve high accuracy but also ensure **model interpretability** and real-world deployment via a user-friendly web application.
Techniques Explored & Used
1. Transfer Learning
We leveraged **pretrained models** (ImageNet weights) to save training time and improve generalization, given the medical image dataset was relatively small.

- **EfficientNetB0**
- **MobileNetV2**
- **Custom CNN**

2. Grad-CAM (Gradient-weighted Class Activation Mapping)
Used to highlight **important regions in X-ray images** that influenced the model's decision, improving **trust and explainability** of predictions.
3. Ensemble Learning
To reduce individual model biases and stabilize predictions, we implemented:

- **Confidence-based Weighted Averaging Ensemble**
  Each model's prediction was weighted based on its confidence level for the given image, rather than simple averaging.

4. Techniques Tried but Not Used / Failed

- **Unfreezing the entire pretrained model** led to overfitting quickly due to the small dataset.
- **Equal weighted ensemble** performed slightly worse than confidence-weighted due to misalignment in confidence strength.
- **Ensemble with voting** was tried but discarded since it lacked probability awareness per sample.

5. Challenges Faced

- **Overfitting during fine-tuning** - Solved by freezing early layers and using dropout + data augmentation.
- **Mismatched predictions across models** - Tackled with a confidence-based ensemble to make more balanced predictions.
- **Gradient vanishing in custom CNN** - Addressed by reducing depth and using ReLU activation with batch normalization.
- **Explaining black-box decisions to users** - Solved with Grad-CAM visualizations using top_conv layer in EfficientNet.
- **Streamlit UI minimalism vs. hackathon standards** - Enhanced UI with color-coded sections, clear model separation, Grad-CAM side-by-side display, and ensemble confidence bar.

6. Evaluation Metrics (Best Ensemble Model)
- **Accuracy:** 94.71%
- **Precision:**
- **Recall:**
- **F1-Score:**
- **AUC Score:** 0.9874

The final ensemble outperformed each individual model by leveraging their strengths and reducing misclassifications through weighted averaging.

7. Deployment
- **Frontend:** Streamlit web interface
- **Backend:** Pretrained .keras models (CNN, MobileNetV2, EfficientNetB0)
- **Explainability:** Grad-CAM heatmaps dynamically generated for uploaded image
- **Deployment Platform:** Streamlit Cloud (Free-tier hosting for real-time demo)

8. Conclusion & Final Remarks
This project successfully fused **deep learning**, **model interpretability**, and **deployment engineering** into one production-ready system. The **ensemble approach** significantly improved reliability, while **Grad-CAM** boosted trustworthiness — crucial in medical AI systems. The project was not just about building accurate models, but also about building an **AI system that users can trust and interact with intuitively.**

Telecom Customer Churn Prediction
Objective
The goal of this project was to develop a robust, interpretable, and deployable machine learning system to **predict customer churn** in a telecom setting. We systematically explored multiple families of models—**Linear**, **Tree-based**, **Boosting**, and **Stacking Ensembles**—to identify the best solution for balanced accuracy, recall (especially for churners), and deployment efficiency.
Phase 1: Linear Models
We began with baseline linear classifiers due to their **speed, simplicity, and interpretability**. Feature preprocessing involved scaling and one-hot encoding. We trained:
- **Logistic Regression**
- **Support Vector Machine (SVM)**
- **K-Nearest Neighbors (KNN)**
- **Naive Bayes**

Observations:
- **Logistic Regression** with L2 regularization performed best, showing high AUC (~0.83) and good recall (~80%) on churners.
- **SVM** had decent performance but was computationally expensive.
- **KNN** and **Naive Bayes** struggled with class imbalance and nonlinear boundaries, even with SMOTE applied.

Conclusion: We retained **Logistic Regression** due to its stable performance and explainability via coefficients.
Phase 2: Tree-Based Models
Next, we experimented with tree models capable of **handling nonlinearities** and **interactions between variables**:
- **Decision Tree**
- **Random Forest**
- **Extra Trees**

Observations:
- **Random Forest** performed the best among tree models with AUC ~0.84.
- Trees offered better interpretability via feature importance but were prone to overfitting without proper tuning.
- **Extra Trees** were fast but less stable across validation folds.

We shortlisted **Random Forest** as a solid tree-based learner.
Phase 3: Boosting Algorithms
Boosting models were explored for **fine-grained decision boundaries**, better **recall**, and overall **predictive strength**:
- **XGBoost**
- **CatBoost**
- **LightGBM**

Evaluation Summary:
XGBoost-~0.83(AUC)Strong- slower to train
CatBoost-~0.82(AUC)-Great with raw categoricals (not used here)

LightGBM-~**0.85**-Fastest, best performance, and SHAP-compatible
LightGBM was selected for its **speed**, **accuracy**, and seamless integration with **SHAP for explainability**.
Phase 4: Stacked Ensemble
To combine **global interpretability** and **local accuracy**, we created a **stacked ensemble**:
Architecture:
Base learners:
- Logistic Regression (L2 penalty)
- LightGBM (boosted trees)

Meta Learner:
Logistic Regression to blend base outputs
Final Evaluation:
- **Accuracy**: 79%
- **AUC Score**: **0.834**
- **Precision (Churn)**: 64%
- **Recall (Churn)**: 49%
- **Macro F1 Score**: 0.71

The stacked model effectively **balanced bias and variance**, outperforming individual models, especially in **recall stability**.
Deployment
The final model was deployed using **Streamlit**, with:
- **Manual input** and **CSV upload** options
- Adjustable **churn threshold slider**
- **SHAP explanations** for uploaded or individual samples
- CSV **download** with predictions

Tech Stack
Python · Scikit-learn · LightGBM · SHAP · Pandas · Matplotlib · Streamlit · SMOTE (imbalanced-learn)
Summary
"I began with interpretable linear models, moved to trees for nonlinearity, adopted LightGBM for optimized boosting, and finally combined it with Logistic Regression using a stacked ensemble. The result was a balanced and explainable churn prediction model, deployed with a user-friendly interface."

Linear Algebra Visual Toolkit
Objective
To build an interactive educational toolkit that **visualizes core linear algebra concepts** like solving linear systems, matrix transformations, and Principal Component Analysis (PCA) using NumPy. The project is aimed at helping students and beginners intuitively understand the geometric and algebraic foundations behind linear algebra operations through visual, real-time examples.
Initial Process and Techniques Used
1. **Core Modules Designed**
Gaussian Elimination Solver:
- Accepts user-defined systems of linear equations
- Builds augmented matrices and performs forward + back substitution
- Step-wise solution display with matrix formatting

Matrix Transformation Visualizer:
- 2D and 3D vector visualization
- User inputs custom vectors and matrices
- Applies linear transformations and shows before/after on coordinate grids

PCA Visualizer (from scratch):
- Covariance matrix computation
- Eigen decomposition using NumPy
- Displays original data with projected principal components
- Compresses data to user-selected top k PCs

Tools & Technologies
- NumPy, Matplotlib, Seaborn for computation & visualization
- Streamlit for interactive UI and sliders
- Responsive layout using streamlit.columns, dynamic dropdowns, and file upload

Techniques That Failed or Evolved

- **Matplotlib 3D Plotting** was clumsy and laggy in Streamlit → Replaced with optimized 2D transformations and fallback visualization for clarity
- Initially used PCA() from scikit-learn → Switched to a fully **manual implementation of PCA** using covariance and eigens for educational depth
- Tried automatic equation parsing with sympy → Rejected in favor of matrix-based manual input for simplicity and focus on core learning

Challenges Faced

- **Visualizing 3D transformations** in a compact interface → Solved by giving users the option to switch between 2D and 3D and simplifying 3D space
- Keeping matrix math human-readable while formatting equations → Built LaTeX-style matrix formatting using Markdown in Streamlit
- Making PCA visual intuitive → Used centered scatter plots and principal axes with arrows for clarity

Final Solution & How It Was Completed

We modularized the app into three Streamlit pages:

- **Solve Linear Systems** using Gaussian Elimination (forward + back pass)
- **Matrix Transformation**: Visual before/after of applying matrices on user vectors
- **PCA Explorer**: Upload dataset → visualize variance → project top k components → download reduced CSV

Features:

- Matrix builder UI with sliders for dimension control
- Latex-style matrix preview and step-by-step walkthrough
- Interactive vector inputs and transformation playback
- PCA reduction with explained eigen decomposition
- CSV download for PCA-compressed data

Deployment

- Built entirely in **Streamlit**
- Hosted using **Streamlit Cloud** with public access
- Includes file upload, matrix entry, download of PCA results

Skills

•

Programming Language: Python (Intermediate)
ML Libraries & Tools: NumPy, Pandas, Scikit-learn,
Keras, SHAP, SMOTE, Grad-CAM
Deployment & Frameworks: Flask, Streamlit, Render, Git,
Google Colab

Certificates

•

1. Linear Algebra for Machine Learning and Data
Science
Coursera
2. Artificial Intelligence Primer Certification
Infosys Springboard | Score : 82.5
3. Industrial IoT & Industry 4.0
NPTEL (Silver Certificate, Merit Holder)
4. Python Basics
HackerRank Skill certification Test
5. SQL Basics
HackerRank Skill Certification Test
6. BEC Preliminary English Exam
Cambridge (Score: 152/170)
7. Finished machine learning Masterclass from Novitech
private limited.

Publications

ICAISS-2023, Care College of Engineering, Trichy
"Monitoring of Prosthetic Leg During Rehabilitation Using IoT" (Scopus Indexed) Real-time movement tracking of prosthetic and normal legs using IoT sensors via ThingSpeak.

Hackathons

• Top 50 Finalist – Thryve Digital National Healthcare Hackathon
• Participated in Annual Innovation Expo – MVJ College of Engineering, Bangalore
• Built predictive model in Humidity Prediction Challenge – MachineHack

Languages

- Tamil-proficient
- English-Fluent

Hobby/ Interests

- My artistic journey began at a young age with a simple apple sketch that sparked admiration and encouragement from my family—this early support ignited a deep passion for art that continues to grow.
- Over the years, sketching has become more than a hobby—it's a personal space where I reflect, express emotions, and constantly evolve as an artist.
- Dancing entered my life through a beautiful exchange of talents with a childhood friend—she taught me to dance, and I taught her to draw. That friendship planted the seed for a lifelong love for movement.
- From school performances to being part of the Fine Arts Club in college, dance has always been my way of expressing energy, joy, and storytelling through rhythm.

Professional Experience

Open Source Engineering Cooperation Bengaluru, India. Explored the concepts of C fundamentals and the working of sensors and Microcontrollers