NeuroOPTIMA

A PROJECT REPORT

BY

Arsh Srivastava (E23CSEU0316)

Krrish Chhabra (E23CSEU0320)

Hardik Aggarwal (E23CSEU0370)

A blue and red text on a black background

Description automatically generated

SUBMITTED TO

Dr. Yajnaseni Dash

SCHOOL OF COMPUTER SCIENCE ENGINEERING AND TECHNOLOGY, BENNETT UNIVERSITY

GREATER NOIDA, 201310, UTTAR PRADESH, INDIA

April 2025

# DECLARATION

I/We hereby declare that the work which is being presented in the report entitled “NeuroOptima”, is an authentic record of my/our own work carried out during the period from JAN, 2025 to April, 2025 at School of Computer Science and Engineering and Technology, Bennett University Greater Noida.

The matters and the results presented in this report has not been submitted by me/us for the award of any other degree elsewhere.

Arsh Srivastava

(Enroll. No. E23CSEU0316)

Krrish Chhabra

(Enroll. No. E23CSEU0320)

Hardik Aggwaral

(Enroll. No. E23CSEU0370)

# ACKNOWLEDGEMENT

I/We would like to take this opportunity to express my/our deepest gratitude to my/our mentor, **Dr. Yajnaseni Dash**  for guiding, supporting, and helping me/us in every possible way. I/we was/were extremely fortunate to have him as my/our mentor as he provided insightful solutions to problems faced by me/us thus contributing immensely towards the completion of this capstone project. I/We would also like to express my/our deepest gratitude to VC, DEAN, HOD, faculty members and friends who helped me/us in successful completion of this project.

Arsh Srivastava

(Enroll. No. E23CSEU0316)

Krrish Chhabra

(Enroll. No. E23CSEU0320)

Hardik Aggwarwal

(Enroll. No. E23CSEU0370)

**TABLE OF CONTENTS**

[ABSTRACT ix](#_Toc85881410)

[1. INTRODUCTION 1](#_Toc85881411)

[1.1. Problem Statement 1](#_Toc85881412)

[2. Background Research 2](#_Toc85881413)

[2.1. Proposed System 2](#_Toc85881414)

[2.2. Goals and Objectives 2](#_Toc85881415)

[3. Project Planning 3](#_Toc85881416)

[3.1. Project Lifecycle 3](#_Toc85881417)

[4. Project Tracking 5](#_Toc85881422)

[4.1. Deliverables 7](#_Toc85881425)

[5. SYSTEM ANALYSIS AND DESIGN 8](#_Toc85881426)

[5.1. Overall Description 8](#_Toc85881427)

[5.3. Design diagrams/ UML diagrams/ Flow Charts/ E-R diagrams 9](#_Toc85881429)

[5.3.1. Use Case Diagrams 9](#_Toc85881430)

[5.3.2. Sequence Diagram 12](#_Toc85881433)

[6. User Interface 14](#_Toc85881435)

[6.1. UI Description 14](#_Toc85881436)

[6.2. UI Mockup 14](#_Toc85881437)

[7. Algorithms/Pseudo Code 15](#_Toc85881438)

[8. Project Closure 16](#_Toc85881439)

[8.1. Goals / Vision 16](#_Toc85881440)

ABSTRACT

In the rapidly evolving landscape of artificial intelligence and machine learning, selecting the right model for a given problem remains a complex and time-consuming task—especially for those without deep technical expertise. **NeuroOptima** addresses this challenge by serving as an intelligent, conversational recommendation system that bridges the gap between users and machine learning model selection.

At its core, NeuroOptima utilizes a **Gemini-powered chatbot interface** to engage with users in natural language. By analyzing the problem statement provided, it accurately identifies the nature of the task—such as classification or regression—and recommends the most suitable **supervised machine learning model**. The recommendation process is designed to prioritize **model accuracy**, ensuring users receive suggestions that are not only relevant but also empirically strong in performance.

NeuroOptima’s strength lies in its ability to simplify a traditionally technical decision-making process. It is designed to support a wide range of use cases, including educational platforms for ML learners, enterprise-level AI prototyping, research benchmarking, and AutoML system integration. By transforming model selection into an intuitive, guided experience, NeuroOptima enhances both productivity and accessibility in the field of machine learning.

This project represents a significant step toward democratizing AI development, empowering users at all levels—from students to professionals—to make smarter, data-driven decisions with confidence and speed.

1. INTRODUCTION

Choosing the right machine learning model is a foundational yet often challenging task in any AI-driven project. With a growing number of algorithms available—each suited to different data types, problem structures, and goals—the model selection process can become overwhelming, particularly for beginners or non-experts. Even experienced practitioners may waste valuable time experimenting with multiple models before finding one that delivers optimal results.

**NeuroOptima** is designed to solve this problem by offering a smart, conversational system that recommends the most accurate **supervised machine learning model** based on a user’s problem description. Built on top of **Google’s Gemini LLM**, NeuroOptima interacts naturally with users through a chatbot interface, interprets their problem statement, and recommends the best-fit model for maximum performance—saving time, effort, and computational resources.

 Understands natural language problem statements using LLM capabilities.

 Identifies the type of ML task (e.g., classification, regression).

 Recommends the most accurate supervised learning model based on problem type and dataset context.

 Provides a user-friendly chatbot interface for accessibility and ease of use.

 Accelerates model selection and prototyping, reducing trial-and-error.

 Can be integrated into educational tools, enterprise systems, and AutoML platforms.

 Makes machine learning more accessible to students, startups, and non-experts.

* 1. Problem Statement

In the field of machine learning, selecting the most appropriate algorithm for a given problem is a critical step that significantly influences the success and accuracy of a model. However, this selection process is often complex, requiring a deep understanding of various supervised learning techniques, data characteristics, and performance trade-offs. This creates a barrier for beginners, data analysts, and even experienced practitioners who may spend considerable time experimenting with different models to find the best fit.

There is currently a lack of intelligent systems that can accurately interpret a user's problem statement and recommend the most suitable machine learning model in a fast, reliable, and user-friendly manner. Most existing solutions either rely on static decision trees or require technical expertise to operate effectively.

**NeuroOptima** aims to address this gap by providing an AI-driven recommendation system that uses a large language model to understand the user's problem and suggest the most accurate supervised learning model for it. By simplifying and accelerating the model selection process, NeuroOptima enhances productivity, reduces time to deployment, and makes machine learning more accessible to a wider audience.

1. Background Research
   1. Proposed System

The proposed system, NeuroOptima, is designed as an intelligent model recommendation platform that utilizes the capabilities of a large language model (LLM) to streamline the process of supervised machine learning model selection. At its core, NeuroOptima features a conversational interface powered by Google’s Gemini chatbot, which enables users to input their problem statements in natural language. The system then processes the input to identify the type of machine learning task—such as classification, regression, or forecasting—and analyzes contextual cues to determine the characteristics of the dataset or problem.

Once the problem type is classified, NeuroOptima evaluates and recommends the supervised learning model that is expected to yield the highest accuracy for the given scenario. This recommendation is based on model performance benchmarks, historical data, and known strengths of various algorithms in specific contexts. The system not only simplifies the decision-making process for users but also significantly reduces the time and effort traditionally required to test and compare multiple models manually.

In addition, the modular architecture of NeuroOptima allows for easy integration with existing AutoML pipelines, educational tools, and enterprise AI solutions. Its user-friendly, conversational interface makes it accessible to users of varying technical expertise, from students and researchers to business analysts and developers. By leveraging LLM capabilities and focusing on performance-driven recommendations, NeuroOptima offers a powerful, scalable solution to the model selection problem in supervised learning.

* 1. Goals and Objectives

|  |  |
| --- | --- |
| **#** | **Goal or Objective** |
| 1 | **Develop an intuitive conversational interface** powered by **Google’s Gemini LLM** to understand natural language problem statements and classify the type of machine learning task (e.g., classification, regression). |
| 2 | **Recommend the most accurate supervised machine learning model** based on the problem description, dataset characteristics, and empirical performance benchmarks. |
| 3 | **Simplify the model selection process** by providing clear and understandable recommendations, reducing the time and effort required to experiment with different models. |
| 4 | **Ensure scalability and integration** with existing AutoML tools, enterprise systems, and educational platforms to enhance the accessibility and usability of machine learning for users of varying expertise. |
| 5 | **Continuously improve recommendation accuracy** by incorporating user feedback, keeping the system updated with the latest advancements in machine learning algorithms and best practices. |

1. Project Planning

This section covers the details of the project planning. Selecting the lifecycle of the development, project stakeholders, resources required, assumptions made (if any) are detailed in the sections below.

* 1. Project Lifecycle

**1. Requirements Gathering & Analysis**

* **Objective:** Understand the problem space and user needs.
* **Key Activities:**
  + Conduct user interviews and surveys to understand the types of problems users face in model selection.
  + Analyze existing tools and solutions for model recommendation.
  + Define system requirements, including supported problem types and performance metrics.
* **Output:** Detailed project requirements document.

**2. System Design & Architecture**

* **Objective:** Design the system's overall architecture and define its core components.
* **Key Activities:**
  + Choose the appropriate machine learning models and performance benchmarks.
  + Design the conversational interface (Gemini chatbot).
  + Plan integration with external systems like AutoML platforms or enterprise tools.
  + Define data flow, database schema (for storing models and performance data), and backend APIs.
* **Output:** System design and architecture blueprint.

**3. Development & Implementation**

* **Objective:** Build and implement the core components of NeuroOptima.
* **Key Activities:**
  + Develop the **LLM-powered conversational interface** using Google’s Gemini.
  + Implement model recommendation algorithms based on problem description analysis.
  + Integrate the recommendation system with benchmark datasets for model performance validation.
  + Implement user feedback mechanisms to continuously improve recommendations.
* **Output:** Working prototype of NeuroOptima with core functionality.

**4. Testing & Validation**

* **Objective:** Ensure the system performs as expected and meets requirements.
* **Key Activities:**
  + Conduct **unit tests** to check the functionality of individual components (e.g., chatbot interactions, model recommendations).
  + Perform **integration testing** to ensure seamless interaction between the system components.
  + Validate **accuracy and reliability** of model recommendations by comparing the suggested models to real-world use cases.
  + Gather feedback from initial users to identify potential improvements.
* **Output:** A fully-tested system ready for deployment.

**5. Deployment & Integration**

* **Objective:** Deploy the system for use in real-world environments and integrate it with other tools.
* **Key Activities:**
  + Deploy NeuroOptima on cloud platforms or enterprise systems.
  + Integrate with external **AutoML platforms**, **educational tools**, or **AI workflows**.
  + Set up **user authentication** and access control for enterprise use.
  + Monitor system performance and track usage patterns for continuous improvement.
* **Output:** Fully deployed and operational system.

**6. Monitoring & Continuous Improvement**

* **Objective:** Continuously improve the system based on user feedback and evolving machine learning practices.
* **Key Activities:**
  + Monitor system performance and user interaction to identify bottlenecks or usability issues.
  + Collect feedback from users to refine model recommendations and improve the chatbot interface.
  + Update the system with new algorithms, datasets, and model performance benchmarks.
  + Regularly evaluate the system's effectiveness and adapt it to new trends in machine learning.
* **Output:** Regularly updated system with enhanced features and improved accuracy.

1. Project Tracking
   1. Deliverables

|  |  |
| --- | --- |
| **#** | **Deliverable** |
| 1 | Code |
| 2 | Dataset Files |
| 3 | Deployment |
| 4 | Data Preprocessing |
| 5 | Models Trained |
| 6 | Powerpoint Presentation |
| 7 | Project Report |
| 8 | Working Demo |

1. SYSTEM ANALYSIS AND DESIGN

This section describes in detail about the design part of the system.

* 1. Overall Description

\*\*NeuroOptima\*\* is an intelligent, LLM-powered recommendation system designed to assist users in selecting the most accurate supervised machine learning model based on a given problem statement. Using a \*\*Gemini chatbot interface\*\*, NeuroOptima interacts with users through natural language, analyzing the input to determine the nature of the task—whether it's classification, regression, or another supervised learning problem. The system then recommends the most suitable model, prioritizing accuracy and efficiency, thus reducing the time and effort typically required for trial-and-error model selection.

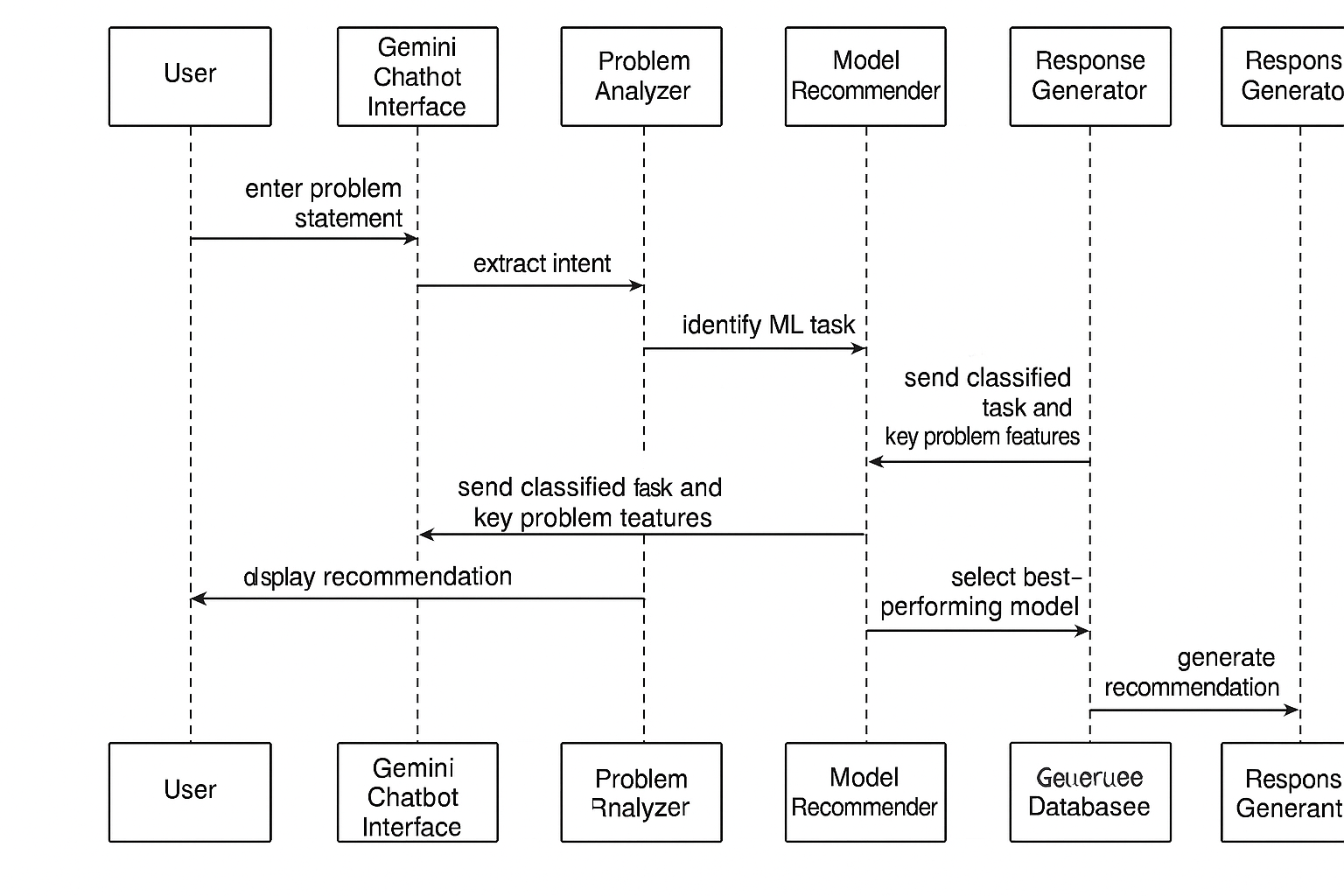
This project aims to democratize machine learning by making model selection more accessible, particularly for individuals without deep technical expertise. NeuroOptima integrates seamlessly with existing AutoML platforms, enterprise AI systems, and educational tools, offering a scalable solution for both beginners and professionals. By automating the process of choosing high-performing models, NeuroOptima enhances productivity, reduces computational costs, and accelerates the development of AI-driven applications.

* + 1. Architecture Diagram

A diagram of a model

AI-generated content may be incorrect.

* + 1. Sequence Diagram



1. User Interface
   1. UI Description

The NeuroOptima user interface is designed with a sleek, modern aesthetic centered around simplicity, usability, and conversational interaction. Upon landing on the main screen, users are greeted by a clean, dark-themed background with a subtle grid and starfield pattern, creating a futuristic and research-oriented visual tone. The bold, welcoming headline “Your Research AI Assistant” (customized here for NeuroOptima) draws immediate attention, with supportive subtext that clearly communicates the platform’s core functionality—helping users identify the most accurate supervised machine learning model based on their problem statement. Two prominent buttons—“Start Chatting” and “See Examples”—invite users to engage with the system intuitively, encouraging both exploration and immediate use.

Once the user begins interacting, a chatbot window titled “Research Assistant” (customized for NeuroOptima) expands to the center of the screen. This chat interface maintains the same clean design language, featuring a minimalistic layout with a centered chatbot icon and a friendly prompt: “How can I help with your research?” The chat input bar at the bottom allows users to enter their problem statement in natural language, which the system then processes to classify the task (e.g., classification or regression) and recommend the best-fit machine learning model based on accuracy. The overall UI emphasizes clarity, responsiveness, and ease of access, making advanced model selection approachable even for users with limited ML expertise. By combining conversational AI with a user-friendly design, the NeuroOptima interface ensures that powerful ML insights are just a message away.

* 1. UI Mockup

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Algorithms/Pseudo Code OF CORE FUNCTIONALITY

<<# 📦 Imports

import pandas as pd

import numpy as np

import torch

import torch.nn as nn

from torch.utils.data import Dataset, DataLoader

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

# 📂 Load CSVs from ai\_pro dataset

genral\_df = pd.read\_csv('/kaggle/input/ai-pro/Genral\_Model\_Performance.csv')

human\_df = pd.read\_csv('/kaggle/input/ai-pro/HumanEval.csv')

reasoning\_df = pd.read\_csv('/kaggle/input/ai-pro/Reasoning\_Dataset.csv')

# 🧹 Clean column names and data

def clean\_columns(df):

df.columns = df.columns.str.strip().str.replace('"', '').str.replace("'", "'").str.replace('"', '').str.replace('"', '').str.replace('\u200b', '').str.replace('\t', '')

# Also clean tab characters from data values if they're strings

for col in df.select\_dtypes(include=['object']).columns:

df[col] = df[col].astype(str).str.replace('\t', '')

return df

genral\_df = clean\_columns(genral\_df)

human\_df = clean\_columns(human\_df)

reasoning\_df = clean\_columns(reasoning\_df)

# 🧱 Rename 'Model' column

def fix\_model\_col(df):

for col in df.columns:

if 'model' in col.lower():

df.rename(columns={col: 'Model'}, inplace=True)

break

return df

genral\_df = fix\_model\_col(genral\_df)

human\_df = fix\_model\_col(human\_df)

reasoning\_df = fix\_model\_col(reasoning\_df)

# 🔗 Merge on 'Model'

df = genral\_df.merge(human\_df, on='Model', how='outer').merge(reasoning\_df, on='Model', how='outer')

df = df.drop\_duplicates(subset=['Model'])

df = df.set\_index('Model')

# ❌ Remove unwanted columns

if 'HumanEval (0 shot)' in df.columns:

df = df.drop(columns=['HumanEval (0 shot)'])

# 🧹 Keep only numeric columns and drop NaNs

df = df.select\_dtypes(include=[np.number]).dropna()

# 🔍 Show columns to help select target

print("\n🧠 Available columns for prediction:\n")

for col in df.columns:

print(f"- {col}")

# After cleaning and merging the dataframes

print("\n🔍 All available columns in the dataset:")

for col in df.columns:

print(f"- {col}")

# Then choose a target column that actually exists in your data

# For example (you'll need to pick one from the printed list):

target\_col = 'MMLU' # Replace this with a column name that exists in your dataset

# Continue with your check

if target\_col not in df.columns:

from difflib import get\_close\_matches

suggestion = get\_close\_matches(target\_col, df.columns, n=1)

raise ValueError(f"❌ Column '{target\_col}' not found. Did you mean: {suggestion}?")

# 🎯 Split features and target

X = df.drop(columns=[target\_col])

y = df[target\_col]

# 📏 Normalize features

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# 📦 PyTorch Dataset

class ModelDataset(Dataset):

def \_init\_(self, X, y):

self.X = torch.tensor(X, dtype=torch.float32)

self.y = torch.tensor(y.values, dtype=torch.float32).view(-1, 1)

def \_len\_(self):

return len(self.X)

def \_getitem\_(self, idx):

return self.X[idx], self.y[idx]

# 🔁 Train/test split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

train\_ds = ModelDataset(X\_train, y\_train)

test\_ds = ModelDataset(X\_test, y\_test)

train\_loader = DataLoader(train\_ds, batch\_size=8, shuffle=True)

test\_loader = DataLoader(test\_ds, batch\_size=8)

# 🧠 Simple Regressor

class Regressor(nn.Module):

def \_init\_(self, input\_dim):

super().\_init\_()

self.net = nn.Sequential(

nn.Linear(input\_dim, 64),

nn.ReLU(),

nn.Linear(64, 32),

nn.ReLU(),

nn.Linear(32, 1)

)

def forward(self, x):

return self.net(x)

model = Regressor(X.shape[1])

loss\_fn = nn.MSELoss()

optimizer = torch.optim.Adam(model.parameters(), lr=0.001)

# 🚂 Training loop

for epoch in range(100):

model.train()

for xb, yb in train\_loader:

preds = model(xb)

loss = loss\_fn(preds, yb)

optimizer.zero\_grad()

loss.backward()

optimizer.step()

if epoch % 10 == 0:

print(f"📉 Epoch {epoch+1}, Loss: {loss.item():.4f}")

# 🧪 Evaluation

model.eval()

total\_loss = 0

with torch.no\_grad():

for xb, yb in test\_loader:

preds = model(xb)

loss = loss\_fn(preds, yb)

total\_loss += loss.item()

print(f"\n✅ Final Test MSE: {total\_loss / len(test\_loader):.4f}")

# 🔮 Feature importance analysis

def get\_feature\_importance(model, X\_names):

weights = model.net[0].weight.data.abs().mean(dim=0).numpy()

importance = pd.DataFrame({'Feature': X\_names, 'Importance': weights})

return importance.sort\_values('Importance', ascending=False)

# Print feature importance

importance\_df = get\_feature\_importance(model, X.columns)

print("\n🔍 Feature Importance:")

print(importance\_df.head(10))

# 📊 Make predictions on test set

model.eval()

with torch.no\_grad():

X\_test\_tensor = torch.tensor(X\_test, dtype=torch.float32)

predictions = model(X\_test\_tensor).numpy().flatten()

# Create comparison DataFrame

results = pd.DataFrame({

'Actual': y\_test.values,

'Predicted': predictions

})

# Calculate errors

results['Error'] = results['Actual'] - results['Predicted']

results['Abs\_Error'] = results['Error'].abs()

# Print summary statistics

print("\n📊 Prediction Results:")

print(f"Mean Absolute Error: {results['Abs\_Error'].mean():.4f}")

print(f"Max Error: {results['Abs\_Error'].max():.4f}")

print(f"Min Error: {results['Abs\_Error'].min():.4f}")

# Top 5 worst predictions

print("\n❌ Top 5 Worst Predictions:")

worst\_predictions = results.sort\_values('Abs\_Error', ascending=False).head(5)

for idx, row in worst\_predictions.iterrows():

print(f"Model at index {idx}: Actual {row['Actual']:.4f}, Predicted {row['Predicted']:.4f}, Error {row['Error']:.4f}")

1. Project Closure
   1. Goals / Vision

The vision of NeuroOptima is to democratize access to machine learning by making the model selection process intelligent, intuitive, and accessible to users of all backgrounds. It aims to eliminate the complexity and trial-and-error traditionally associated with choosing the right machine learning model by leveraging the power of large language models. By doing so, NeuroOptima aspires to become a go-to solution for students, researchers, developers, and enterprises looking to accelerate their AI development with confidence and ease.

The primary goals of NeuroOptima are to empower users to select the most suitable supervised learning models through natural language interaction, and to automate the model recommendation process in a way that prioritizes performance and accuracy. It seeks to bridge the gap between machine learning expertise and practical application by offering a smart, conversational interface powered by Google's Gemini. Additionally, the system is designed to integrate seamlessly with AutoML pipelines, educational platforms, and enterprise systems, ensuring scalability and broad adoption across different domains. Through these goals, NeuroOptima positions itself as a valuable tool in the modern AI ecosystem.