JULY. 10, 2025

JULY. 10, 2025

JULY. 10, 2025

INTERNSHIP PROJECT SHOWCASE – MACHINE LEARNING APPLICATIONS

Unified Mentor-

Sachin Bhat UMID15042530602

UNIFIED MENTOR

PRESENTATION HIGHLIGHTS

A collection of four data-driven machine learning projects focused on solving real-world problems using classification, regression, and deep learning techniques.

Each project is supported by model optimization and evaluation metrics.

Projects Included:

1. Animal Image Classification
2. Mobile Phone Price Range Prediction
3. Vehicle Price Regression
4. American Sign Language Detection

PROJECT 1: ANIMAL IMAGE CLASSIFICATION

Objective:

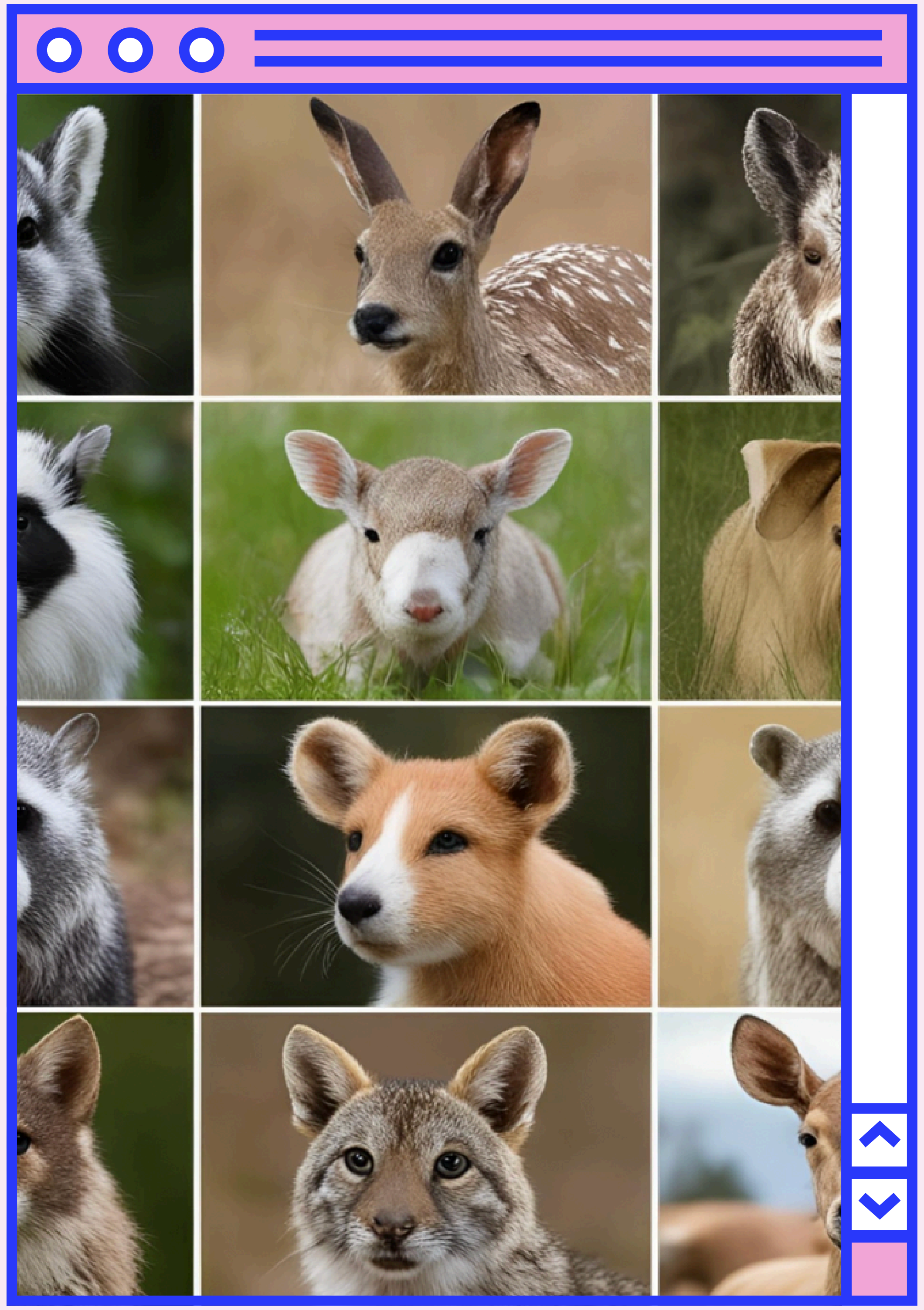
Develop a deep learning-based system to automatically classify animal species from images, eliminating the need for manual labeling and enabling faster, large-scale recognition.

Context:

Manual identification of animals from images is time-consuming, inconsistent, and requires domain expertise. This limits its practicality in wildlife monitoring, ecological research, and educational platforms that rely on large image datasets.

Relevance:

An accurate and scalable classification system can assist conservationists, researchers, and educators by automating image analysis, improving species documentation, and enabling real-time animal detection in various applications.



PROBLEM STATEMENT

Experts face difficulties in classifying large sets of animal images due to:

- High time and labor cost
- Difficulty in distinguishing visually similar species
- Lack of reliable, scalable image-based solutions

Goal: Develop a deep learning model to accurately and automatically classify animals from images.

DATASET SUMMARY:

Source: Custom folder-based dataset with 15 animal classes, each in a separate directory.

Image Input Size: All images resized to 224x224x3 pixels (RGB).

Preprocessing:

Applied data augmentation (rotation, zoom, flip)

Scaled pixel values to [0, 1]

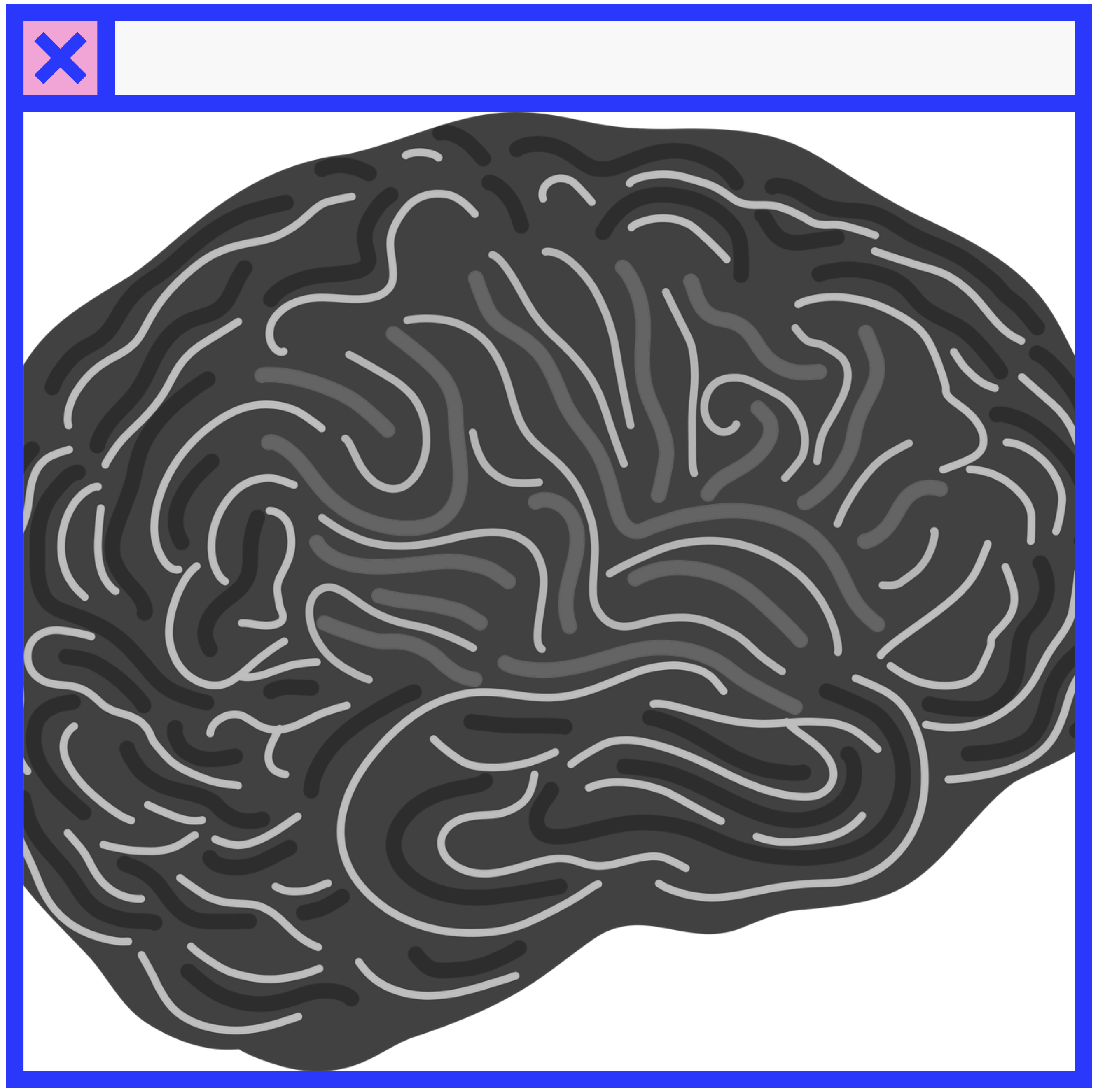
Split: 80% training, 20% validation

Challenge: Multi-class classification with significant visual overlap among classes.

While many deep learning workflows employ a three-way data split (train/validation/test), my current setup using a train/test split has proven sufficient for the scope and goals of this academic project.

MODEL ARCHITECTURE:

Approach: Transfer Learning using MobileNetV2



Base Model:

MobileNetV2 - A lightweight convolutional neural network pre-trained on the ImageNet dataset.

Custom Layers Added:

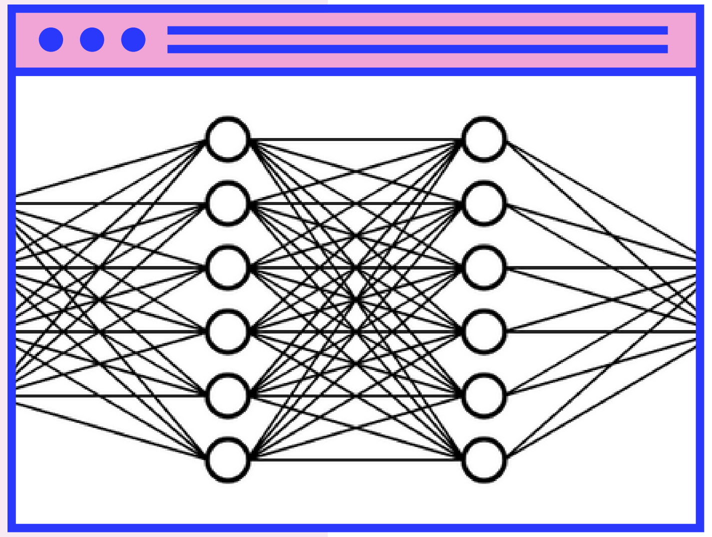
- Global Average Pooling: Reduces the output of the base model to a 1D vector by averaging feature maps, making it suitable for classification.
- Dropout (rate = 0.5): Randomly disables 50% of neurons during training to prevent overfitting and improve generalization.
- Dense Layer + Softmax: Final layer with 15 neurons (one per class), using softmax activation to output a probability distribution over all classes.

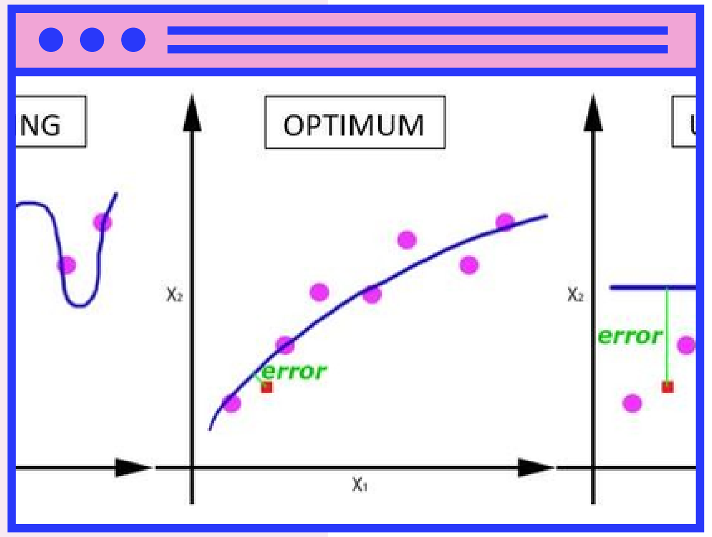
Freezing Base Model:

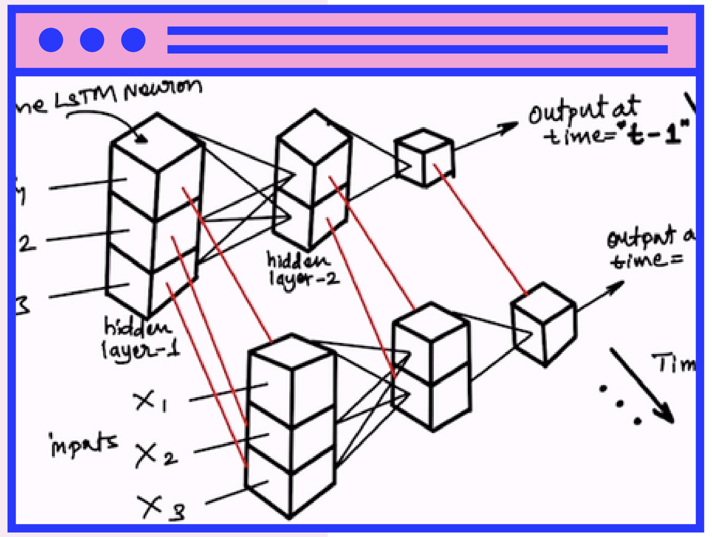
All layers of MobileNetV2 are initially frozen (not trainable), so the pre-learned visual features remain unchanged during training. Only the custom top layers are trained on the animal dataset.

MODEL METRICS/ CONFIGURATION

UNIFIED MENTOR • JULY 10, 2025







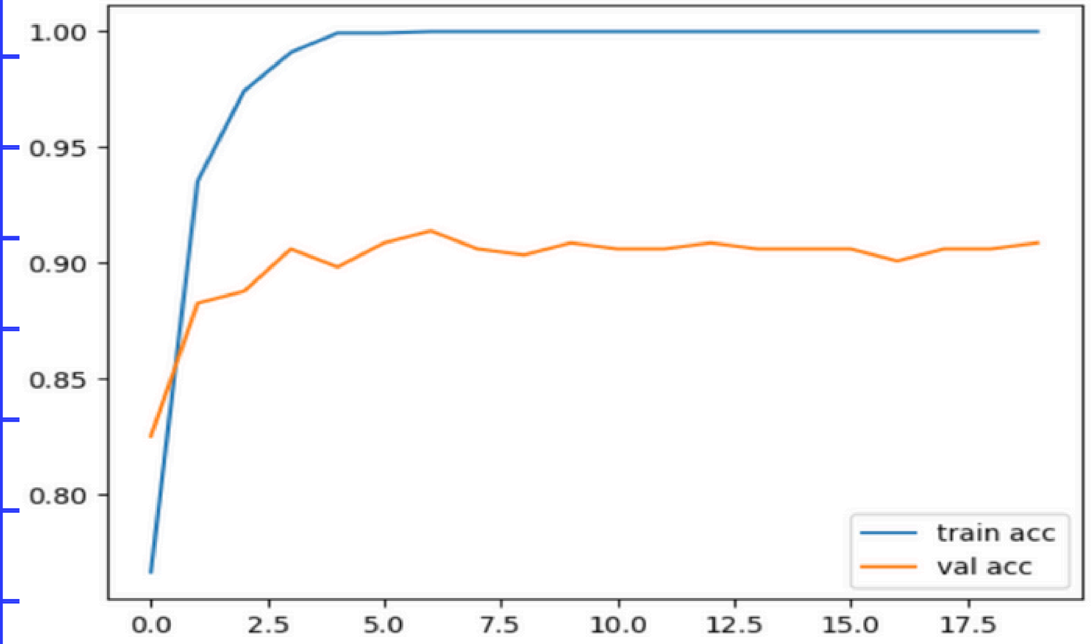
VALIDATION ACCURACY:

ACHIEVED A VALIDATION ACCURACY OF APPROXIMATELY 90% AFTER 20 EPOCHS OF TRAINING.WITH MINIMAL OVERFITTING DUE TO REGULARIZATION (DROPOUT) AND EARLY STOPPING.

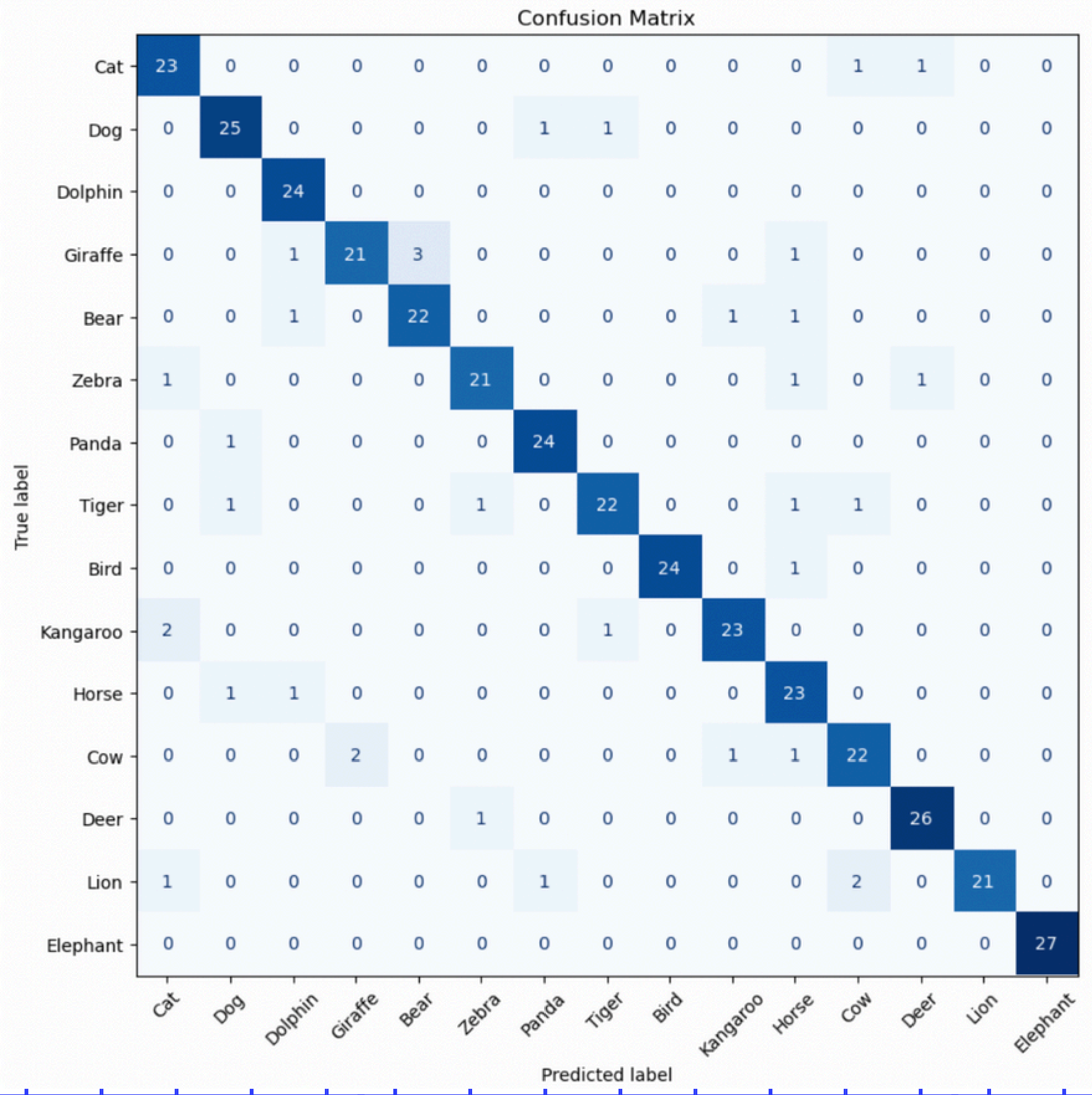
- LOSS FUNCTION USED: CATEGORICAL CROSSENTROPY
- TRAINING STRATEGY: EARLYSTOPPING WITH PATIENCE = 10
- BATCH SIZEX: 32 | LEARNING RATE: 0.0001

- THE MODEL CONVERGED EFFICIENTLY WITHIN THE 20-EPOCH WINDOW.
- FINAL ACCURACY WAS BALANCED ACROSS MOST OF THE 15 CLASSES.

MODEL EVALUATION



Accuracy Charts per epoch



Confusion Matrix of all Test samples

Classification Report

	precision	recall	f1-score	support
Cat	0.85	0.92	0.88	25
Dog	0.89	0.93	0.91	27
Dolphin	0.89	1.00	0.94	24
Giraffe	0.91	0.81	0.86	26
Bear	0.88	0.88	0.88	25
Zebra	0.91	0.88	0.89	24
Panda	0.92	0.96	0.94	25
Tiger	0.92	0.85	0.88	26
Bird	1.00	0.96	0.98	25
Kangaroo	0.92	0.88	0.90	26
Horse	0.79	0.92	0.85	25
Cow	0.85	0.85	0.85	26
Deer	0.93	0.96	0.95	27
Lion	1.00	0.84	0.91	25
Elephant	1.00	1.00	1.00	27
accuracy			0.91	383
macro avg	0.91	0.91	0.91	383
weighted avg	0.91	0.91	0.91	383

Classiciation Report of Model with necessary testing metrics

PROJECT 2: MOBILE PRICE RANGE CLASSIFICATION

Goal: Predict the price category of a mobile phone (Low to Very High) based solely on its technical specifications.

Problem Context: Manual categorization is inconsistent and inefficient for product positioning in retail, manufacturing, and online catalogs.

Solution: Automate the price tier classification using structured data and supervised learning.

PROBLEM STATEMENT

Consumers and businesses struggle to determine phone price categories due to:

Ambiguity in how features influence price.

Inconsistent human judgment.

Lack of scalable, data-driven solutions.

The project aims to develop a model that:

Takes structured specifications (RAM, battery, 4G, etc.)

Outputs a category:

0 = Low, 1 = Medium, 2 = High, 3 = Very High

DATASET SUMMARY

Dataset Size: 2,000 samples

Features: 20 technical specifications (e.g. ram, battery_power, px_height, px_width, four_g, dual_sim)

Target Variable: price_range

→ Categories: 0 = Low, 1 = Medium, 2 = High, 3 = Very High (evenly distributed)

Preprocessing and EDA Highlights

EDA Goals:

Analyze feature distributions and identify correlation.

Key Findings:

ram showed strong correlation with price (+0.91)

Other features had weak to moderate correlation (< 0.2)

No missing or duplicate values

Binary features (e.g., four_g, wifi) were evenly distributed

No major outliers in key numeric columns

Preprocessing Steps:

Standardized numeric features using StandardScaler

Stratified 70/15/15 split for training, validation, and test

Applied RFE to select the top 5 most impactful features

Exploratory Data Analysis

MODEL ARCHITECTURE

BASELINE MODELS:

- Logistic Regression (achieved 96% accuracy)
- Decision Tree Classifier

ADVANCED MODELS:

- Random Forest
- XGBoost with:
- GridSearchCV for hyperparameter tuning
- RFE (Top 5 features used for final model)

TRAINING STRATEGY

Hyperparameter Tuning:

RFE reduced dimensionality, improving performance and reducing complexity

Final Model Training:

Trained on combined train + validation set

Evaluated only once on untouched test set (to avoid data leakage)

SELECTED FINAL MODEL:

XGBoost Classifier

Tuned for n_estimators, max_depth, learning_rate, subsample

EVALUATION METRICS

Accuracy: 97%

Precision: 96.9%

Recall: 96.8%

F1-Score: 96.8%

- Minor misclassifications occur mostly between adjacent price categories
- Model generalizes well, especially in edge cases

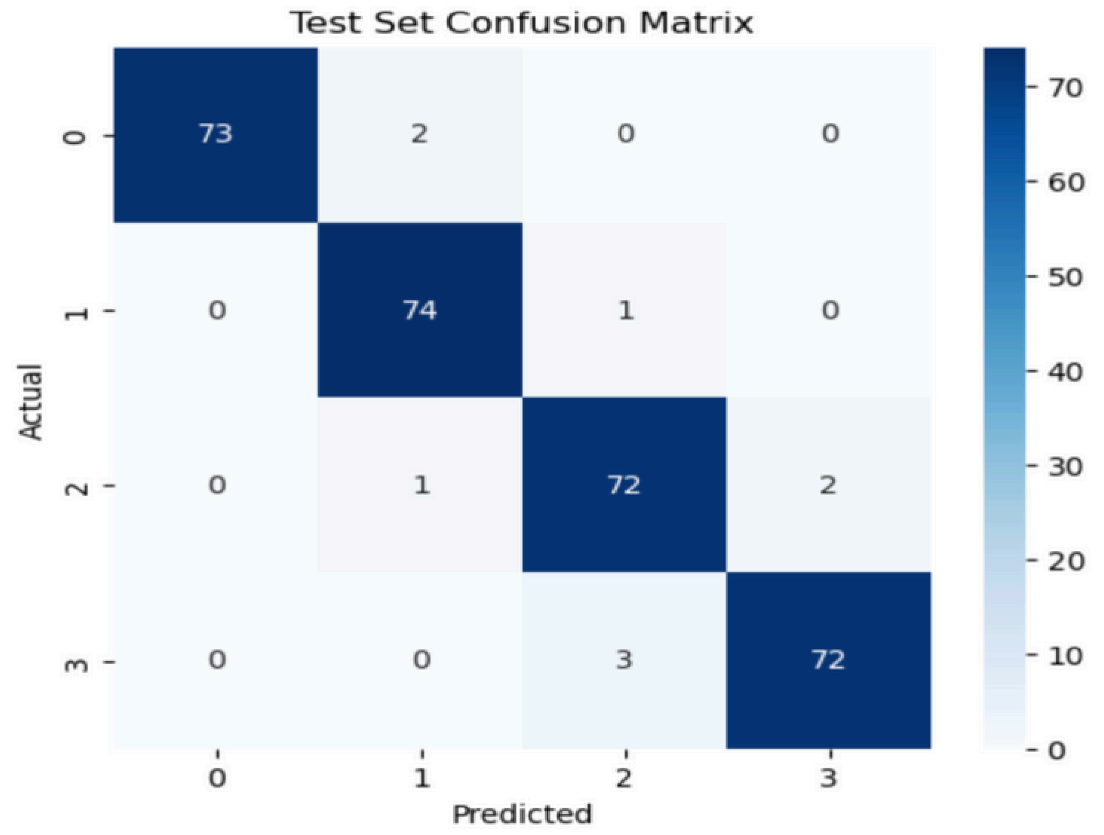
UNIFIED MENTOR • JULY 10, 2025
UNIFIED MENTOR • JULY 10, 2025

Test Accuracy: 0.97

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.97	0.99	75
1	0.96	0.99	0.97	75
2	0.95	0.96	0.95	75
3	0.97	0.96	0.97	75
accuracy			0.97	300
macro avg	0.97	0.97	0.97	300
weighted avg	0.97	0.97	0.97	300

C;assification Report of Final XGBoost model



Confusion Matrix of final model

PROJECT 3: VEHICLE PRICE PREDICTION

CONTEXT: MANUAL PRICING METHODS ARE INEFFICIENT, SUBJECTIVE, AND DON'T SCALE FOR HIGH-VOLUME LISTINGS.

Objective: Predict resale prices of used vehicles using structured features like mileage, age, fuel type, engine size, and brand.

Solution: Build a regression model that automates pricing based on learned market patterns to assist dealerships, buyers, and sellers.

PROBLEM STATEMENT

Pricing used vehicles accurately is difficult due to:

- Interdependent factors like brand, age, mileage, transmission, fuel type, etc.
- Regional differences and volatile demand
- Inconsistent human judgments or outdated pricing tables

UNIFIED MENTOR • JULY 10, 2025

Challenges addressed:

- Subjective and error-prone manual pricing
- Poor scalability for large dealer inventories
- High variance between actual and listed prices

Source: Aggregated vehicle listings

Features: Engine size, fuel type, mileage, transmission, brand, etc.

Target: Actual vehicle resale price (numeric)

EDA AND PREPROCESSING HIGHLIGHTS:

- MISSING VALUES IN MILEAGE, FUEL_TYPE, ETC. WERE IMPUTED (MEDIAN/MODE)
- PRICE DISTRIBUTION SHOWED RIGHT SKEW → LOG-TRANSFORMATION CONSIDERED
- OUTLIERS HANDLED USING IQR FILTERING (E.G., CAPPING EXTREME MILEAGE)
- CORRELATIONS:
  - STRONG: VEHICLE_AGE, MILEAGE, ENGINE_SIZE
  - CATEGORICAL: ONE-HOT ENCODED (FUEL_TYPE, TRANSMISSION)
- FEATURE ENGINEERING: DERIVED VEHICLE_AGE FROM YEAR

DATASET SUMMARY & EDA

Summary Comparison Table:

Model	R^2 Score	MAE	RMSE	Final Decision
Linear Regression	-1.6e+17	$98,851,327,653.92	$240,802,708,915.32	Rejected (baseline)
Random Forest	~0.84	$4,566.18	$7,379.94	Shortlisted
Gradient Boosting	0.9003	$3,943.40	$5,999.29	Shortlisted
HistGradientBoosting	0.67	$7,146.84	$12,428.47	Rejected (underperformed)
XGBoost	0.9158	$4,024.61	$5,514.05	Top-performing single model
Stacking Regressor	**0.8324** (test)	**$4,898.86**	**$9,256.80**	**Selected Final**

*Note: All models were initially evaluated on the validation set. The stacked model was chosen based on the best average validation performance and then evaluated on the test set, hence the slightly lower final test metrics.

Final Model Chosen: Stacked Regressor - An ensemble of multiple base learners (Random Forest, Gradient Boosting, XGBoost) with a final Ridge Regressor meta-model to combine their outputs.

MODEL ARCHITECTURE

TRAINING STRATEGY

- Train/Validation/Test split (70/15/15) using stratified sampling
- Outliers and scaling issues carefully handled to prevent inflated errors
- Important Fix: Removed inappropriate scaling that distorted predictions
- Feature selection: Top 5 predictors used for final model training

Final Model Used:

Stacked Regressor trained on combined train + validation set for best generalization

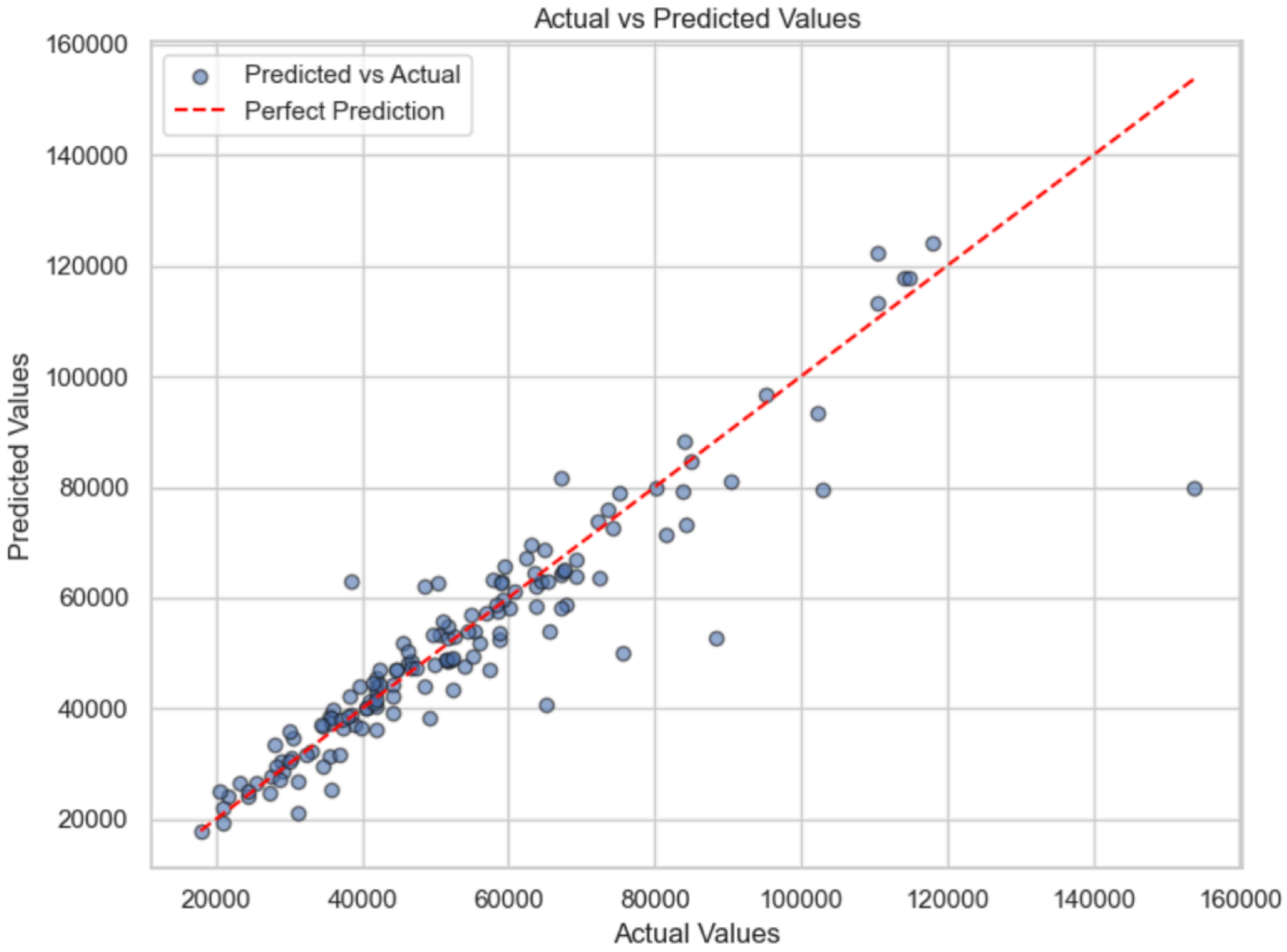
EVALUATION METRICS & RESULTS

Once Validated, the model was evaluatedon the test set on the same metrics:

Mean Absolute Error (MAE): $4,898.86

Root Mean Squared Error (RMSE): $9,256.80

R^2 Score: 0.8324



UNIFIED MENTOR • JULY 10, 2025

PROJECT 4: ASL DETECTION AND PREDICTION

OBJECTIVES AND GOALS:

Goal: Build a model that can classify static ASL hand gestures (A-Z, space, delete, nothing) using deep learning.

Context: Manual ASL interpretation is time-consuming and inaccessible in many digital environments.

Solution: A GPU-accelerated, image-based CNN model trained on over 87,000 labeled samples.

PROBLEM STATEMENT

ASL recognition systems are typically:

Hardware-dependent (e.g., sensors, gloves)

Not scalable or deployable on consumer devices

Limited in speed and generalization

This project aims to:

- Enable real-time ASL classification using static RGB images
- Build a generalizable CNN model trained end-to-end
- Optimize for speed using CUDA-enabled GPU training

DATASET SUMMARY & PREPROCESSING

Dataset: Kaggle ASL Alphabet – 87,000+ images across 29 classes (A–Z, space, del, nothing)

Input Shape: 64 X 64 X 3 RGB images

Preprocessing:

Resize → Normalize to [-1, 1] → Tensor conversion

Used torchvision.ImageFolder + DataLoader

Split: 70% training, 15% validation, 15% test

Batch Size: 64 (optimized for GPU memory)

MODEL ARCHITECTURE

CUSTOM CNN DESIGN (PYTORCH)

Conv2D → ReLU → MaxPool:

Extracts visual features, adds non-linearity, and reduces image size for faster learning.

AdaptiveAvgPool2d:

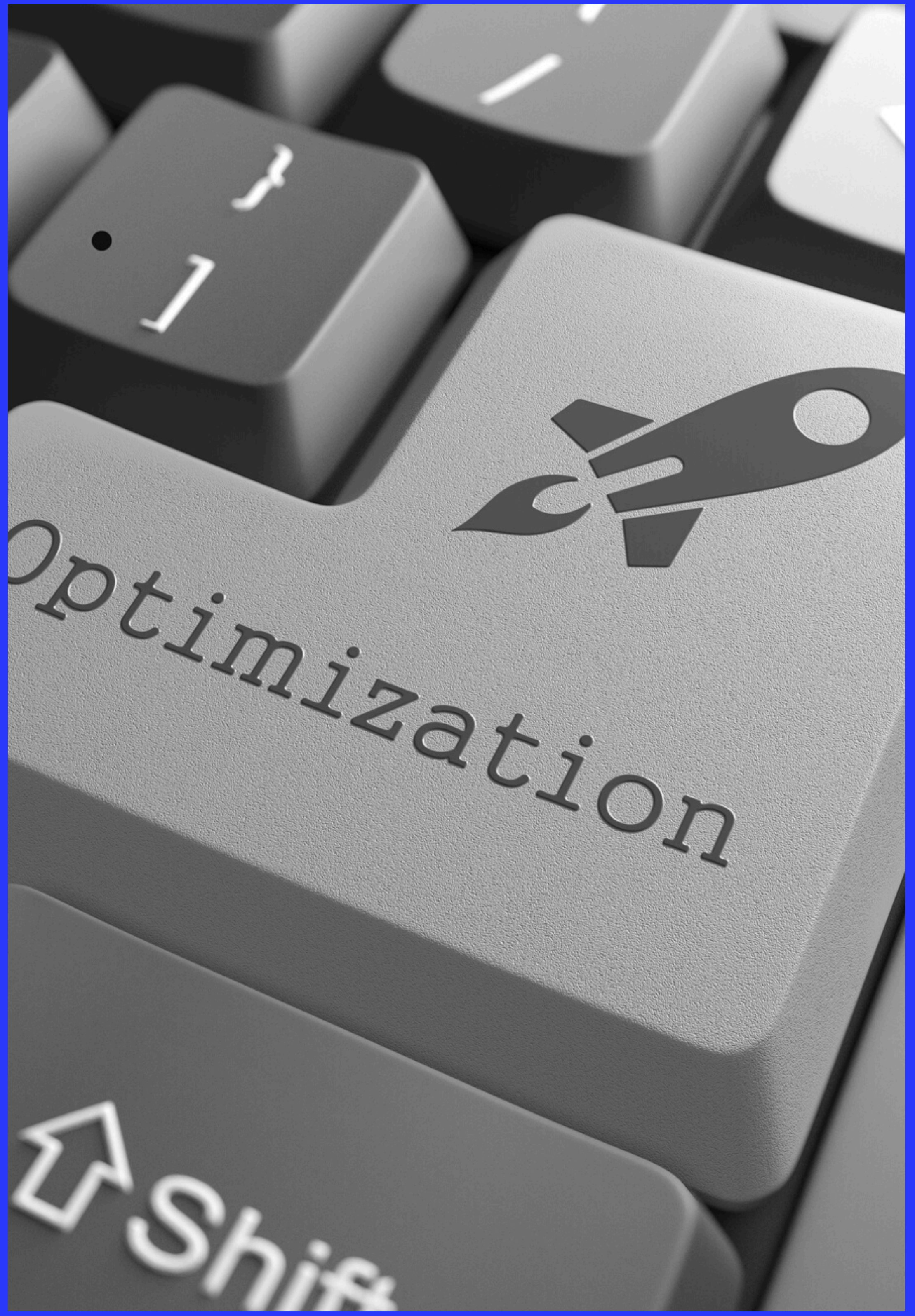
Standardizes output shape to prevent flattening errors, making the model resolution-flexible.

Fully Connected (512 neurons + Dropout):

Learns class-specific patterns; dropout reduces overfitting.

Output (29 logits → Softmax):

Final layer outputs class probabilities for all 29 ASL labels.



TRAINING AND OPTIMIZATION

Loss Function: CrossEntropyLoss

Optimizer: Adam (lr = 0.001)

Training Duration: 20 epochs with early stopping after 3 stagnant val losses

Training Speed:

CPU (Mac): ~42 min/epoch

GPU (RTX 2060 Super via WSL2): ~1.5 min/epoch

Performance Monitoring: tqdm progress bar + real-time validation tracking

UNIFIED MENTOR • JULY 10, 2025
UNIFIED MENTOR • JULY 10, 2025

FINAL EVALUATION

Test Accuracy: 99.74%

Training Loss: 0.025

Validation Loss: 0.0063

Evaluation Tools:

sklearn.metrics.accuracy_score

Best model checkpoint saved during training

Observation:

Strong generalization, minimal overfitting

Achieved near-perfect classification with a simple, efficient architecture

FINAL SUMMARY & REFLECTIONS

- 4 diverse machine learning projects completed across image classification, price prediction, and deep learning.
- Applied advanced modeling techniques including CNNs, XGBoost, Random Forest, RFE, and transfer learning.
- Emphasized real-world usability through data cleaning, tuning, and Streamlit app deployments.
- Tackled key challenges like overfitting, outliers, scaling issues, and model interpretability.
- Learned to manage large datasets, GPU-based training, and model deployment workflow

Key Takeaway:

Each project demonstrates a complete ML pipeline - from EDA and modeling to evaluation and user interaction - making the solutions scalable, accurate, and production-ready.

THANK YOU!