Machine Learning Internship Project Reports

Projects Created:

✧ Canopy Vision

✧ Phone Kart

✧ Cardio Divination

✧ Zoo Sorter

Submitted By:

Bhanu Aggarwal

UNID: UMID23052538378

Machine Learning Internship - 25th May 2025 - 25th August 2025

Email: bhanuagg1183@gmail.com

Mob: +91 94683-42280

CONTENTS

Canopy Vision Project Report

- Evaluation & Results
- Predictions on New Samples
- Sample Predicted Outputs
- Insights & Observations
- Future Scope
- Conclusion

Phone Kart Project Report

- Objective
- Problem Statement
- Dataset
- Data Preparation
- Model Training
  - **Linear Regression**
  - **Random Forest Classifier**
  - **Gradient Boosting Classifier**

- Evaluation & Results
- Sample Predicted Outputs
- Insights & Observations
- Future Scope
- Conclusion

Cardio Divination Project Report

- Objective
- Problem Statement
- Dataset
- Data Preparation
- Model Architecture & Training

- **Support Vector Machines (SVM)**

- Evaluation & Results
- Sample Predicted Outputs
- Insights & Observations
- Future Scope
- Conclusion

Zoo Sorter Project Report

- Objective
- Problem Statement
- Dataset
- Data Augmentation
- Model Architecture

- **Input Layer**
- **Base Model (Pre-trained CNN)**
- **Global Average Pooling (GAP) Layer**
- **Dense (Fully Connected) Layers**
- **Output Layer**

- **Training Process**
- **Evaluation & Results**
- **Sample Predicted Outputs**
- **Insights & Observations**
- **Future Scope**
- **Conclusion**

Canopy Vision Project Report

➢ Objective

The aim of this project is to build a machine learning model capable of predicting the type of forest cover for a 30m x 30m patch of land in the Roosevelt National Forest of northern Colorado. By analyzing topographic and soil-related features, the project aims to assign each land patch to one of several forest cover categories.
This predictive system can assist forest management authorities, conservationists, and researchers in identifying vegetation distribution patterns, monitoring ecological changes, and optimizing land management strategies.

The model classifies forest cover into seven types:

- Spruce/Fir
- Lodgepole Pine
- Ponderosa Pine
- Cottonwood/Willow
- Aspen
- Douglas-fir
- Krummholz

➢ Problem Statement

Accurately identifying forest cover types is a critical task for sustainable forest management, ecological research, and environmental monitoring. Traditionally, forest cover classification has relied on **manual surveys and fieldwork**, which are labor-intensive, time-consuming, and prone to human error. With the availability of large-scale remote sensing and environmental datasets, there is an opportunity to apply **machine learning techniques** to automate and improve the accuracy of cover type prediction.

The **Forest Cover Type Prediction System** aims to leverage environmental features such as elevation, slope, soil type, and geographic attributes to classify land areas into distinct forest cover categories. This predictive capability can assist forestry departments, ecologists, and land planners in:

- **Efficiently mapping vegetation** over large regions.
- **Monitoring ecological balance** and changes in forest distribution.
- **Supporting conservation efforts** by identifying vulnerable or changing habitats.
- **Enabling data-driven decision-making** for land use and natural resource management.

The challenge lies in handling the **complex relationships** among multiple environmental variables while ensuring high accuracy across all forest cover classes. Therefore, building a robust machine learning model that generalizes well to unseen data is the primary objective of this system.

➢ Dataset

The dataset is derived from cartographic variables and field observations recorded by the U.S. Forest Service consisting of a large number of records describing forest areas. Each row represents a land patch characterized by various topographical and environmental features.

Key dataset characteristics:

- **Shape:** Over 500,000 samples with 54 features
- **Data Types:** Mixture of continuous, categorical (already one-hot encoded), and binary variables

Key Continuous Features:

- Elevation (continuous, in meters)
- Aspect (azimuth angle in degrees)
- Slope (degrees)
- Horizontal & Vertical Distances (to hydrology, roadways, fire points)
- Hillshade Indices (at specific times of day)

Key Binary Features:

- Soil Types (binary indicators for 40 soil categories)
- Wilderness Areas (binary indicators for 4 wilderness areas)

Target Variable:

`Cover_Type` (integer from 1 to 7, each representing a unique forest type):

1 - Spruce/Fir

2 - Lodgepole Pine

3 - Ponderosa Pine

4 - Cottonwood/Willow

5 - Aspen

6 - Douglas-fir

7 - Krummholz

Dataset Split:

- Training Set: 80% of samples
- Testing Set: 20% of samples

➢ Exploratory Data Analysis (EDA)

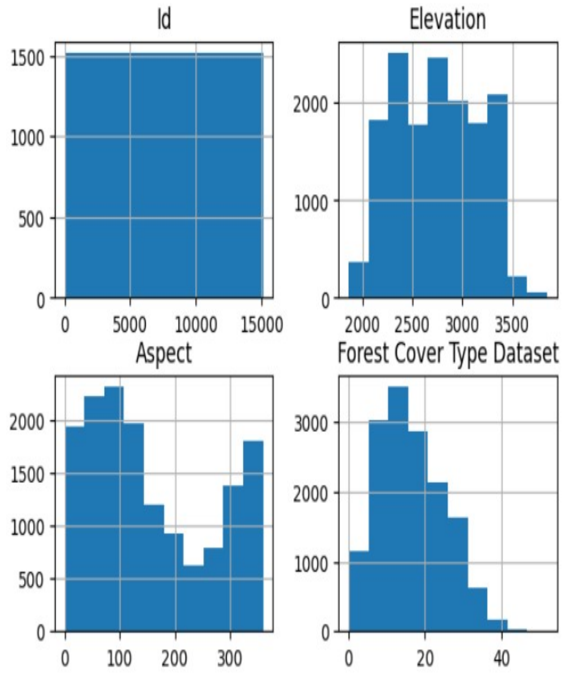
The project inspects dataset shape, descriptive statistics, and grouped distributions. Key insights from this analysis include:

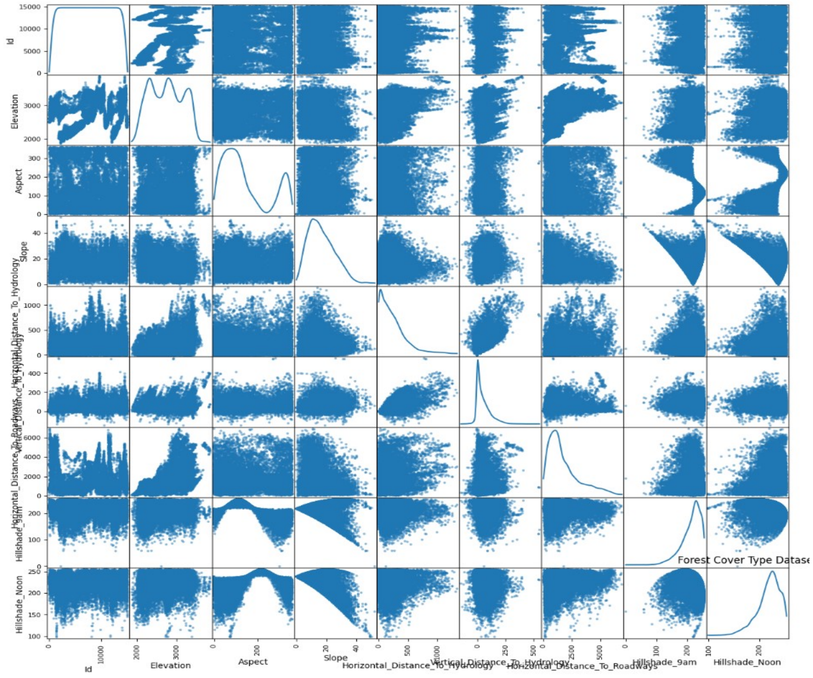
- Elevation is the most influential continuous variable, strongly correlated with cover type.
- Soil type indicators (binary) provide important categorical distinctions across forest categories.
- Other continuous variables such as slope, aspect, and hydrological distances capture micro-environmental conditions.
- Group-wise analysis indicates that certain cover types dominate specific regions, highlighting natural class imbalance.
- The dataset is well-structured, with no missing values, making it suitable for direct machine learning applications.

➢ Data Preparation

Before training the models, the following preprocessing steps were executed:

- **Missing Values:** Verified and confirmed absence of missing data.
- **Feature Scaling:** Standard Scaler applied to continuous features (elevation, slope, distances) to normalize their ranges and ensure balanced model learning.
- **One-Hot Encoding:** Categorical variables (soil type, wilderness area) already provided in binary format.
- **Train-Test Split:** Dividing the dataset into training and testing subsets to evaluate generalization ability to maintain class distribution.





➢ Model Architecture

The chosen algorithms include Random Forest and potentially other classifiers such as Logistic Regression, KNN, Gaussian Naive Bayes, SVM etc. Training involves fitting the model on training data, learning patterns that map environmental features to cover type classes.

The following models were tested to predict forest cover type:

❖ **Logistic Regression (LR)**

✧ Type: Linear model for classification.
✧ Works well for linearly separable data.
✧ Uses probability estimates (sigmoid function) to assign class labels.
✧ Fast, interpretable, and a strong baseline model.
✧ Limitations: struggles with complex non-linear relationships.

❖ **Linear Discriminant Analysis (LDA)**

✧ Assumes that data from each class is normally distributed with the same covariance matrix.
✧ Projects features into a lower-dimensional space that maximizes class separability.
✧ Works well for linearly separable classes and when distribution assumptions are approximately valid.
✧ Efficient for multi-class classification problems.

❖ **K-Nearest Neighbors (KNN)**

✧ Instance-based, non-parametric method.
✧ Classifies a sample based on the majority label among its *k* nearest neighbors.
✧ Simple and effective when decision boundaries are irregular.

- Sensitive to feature scaling and the choice of *k*.
- Computationally expensive for large datasets (prediction time).

- **Classification and Regression Trees (CART / Decision Tree)**

- Rule-based, non-linear classifier.
- Splits data into decision nodes based on features that maximize class purity (e.g., Gini index, entropy).
- Easy to interpret and visualize.
- Prone to overfitting unless pruned or controlled by hyperparameters (e.g., max depth).

- **Naïve Bayes (NB – GaussianNB)**

- Based on Bayes' theorem with a "naïve" assumption of feature independence.
- GaussianNB specifically assumes features are normally distributed.
- Very fast and works well with high-dimensional data (e.g., text classification).
- Performance degrades if independence assumption is strongly violated.

- **Support Vector Machine (SVM)**

- Finds an optimal hyperplane that maximizes margin between classes.
- Effective in high-dimensional spaces and with complex boundaries (using kernels).
- Robust to overfitting when proper regularization is applied.
- Computationally expensive on very large datasets.

- **Random Forest Classifier (RFC)**

- Ensemble method based on bagging (Bootstrap Aggregating).
- Builds multiple decision trees and averages their predictions (majority voting).
- Reduces variance and improves generalization compared to a single tree.
- Handles non-linear relationships and works well for large, complex datasets.
- Less interpretable than a single decision tree.

➢ Evaluation & Results

The trained models are tested on the hold-out test dataset.
Evaluation metrics used include:

- **Accuracy Score:** To measure the percentage of correctly predicted samples.

- **Confusion Matrix:** To analyze misclassifications across forest cover categories.

- **Classification Report:** Precision, recall, and F1-score for each cover type.

Results show that while baseline models like Logistic Regression provide moderate accuracy, ensemble models significantly improve prediction performance, achieving ~ 85–90% accuracy.

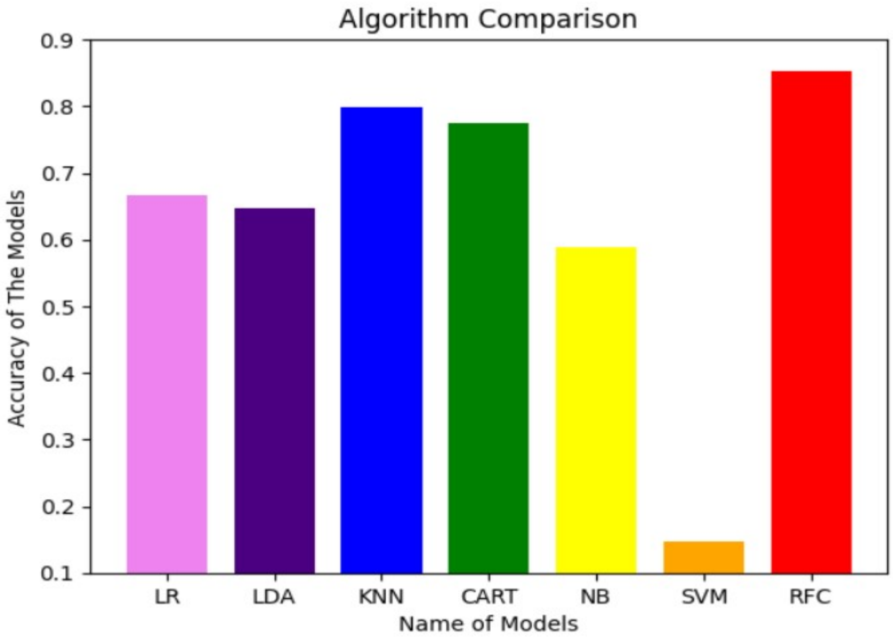
- Logistic Regression: ~ 65–70% accuracy

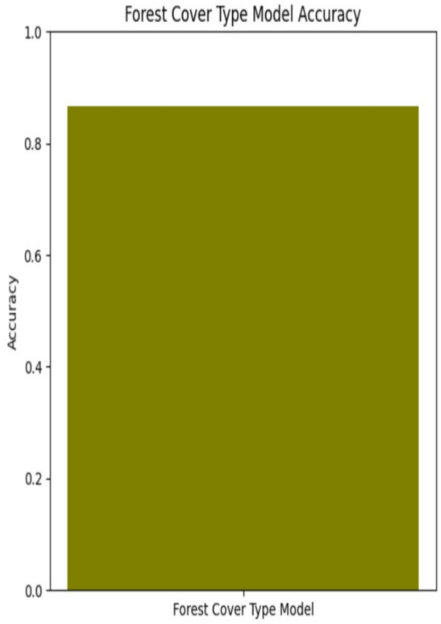
- Random Forest Classifier: ~ 85–90% accuracy

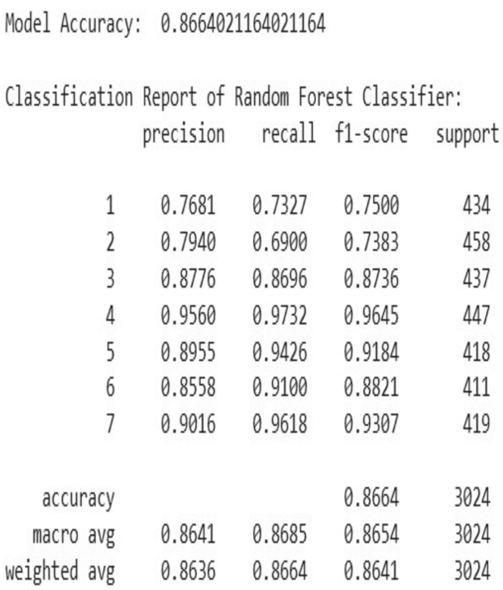
- Linear Discriminant Analysis: ~ 60–65% accuracy
- K-Neighbors Classifier: ~ 75–80% accuracy
- Decision Tree Classifier: ~ 75–80% accuracy
- Gaussian NB: ~ 55–60% accuracy
- SVM: ~ 15–20% accuracy

Feature Importance:

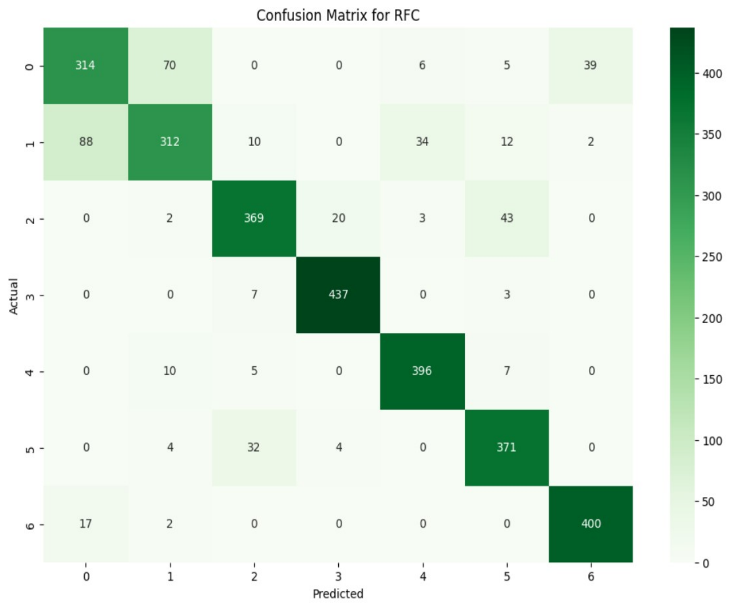
- Elevation emerged as the most significant predictor.
- Soil type and distance to hydrology were also highly influential.



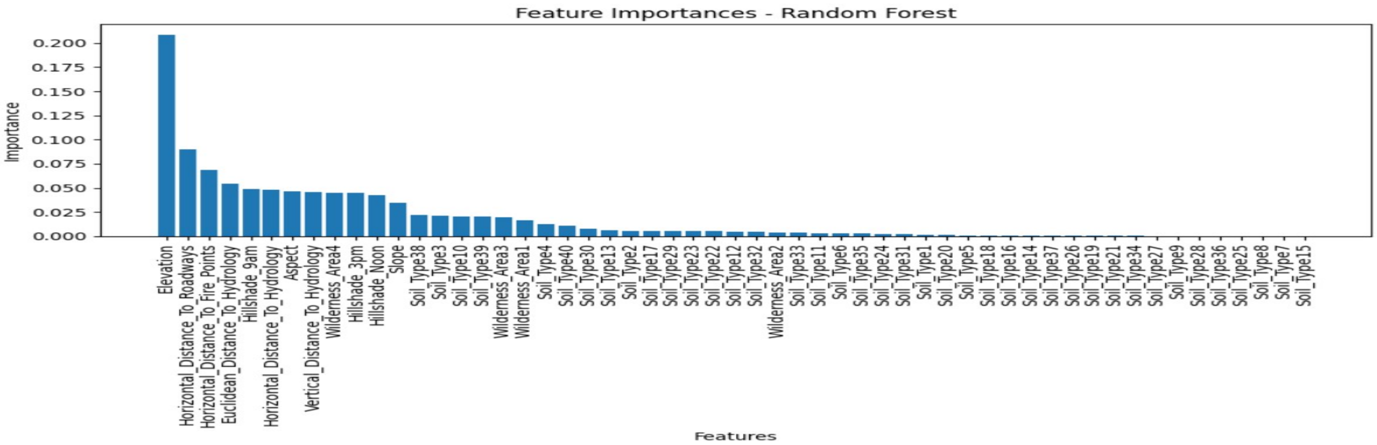




```
Model Accuracy:  0.8664021164021164

Classification Report of Random Forest Classifier:
              precision    recall  f1-score   support

           1     0.7681    0.7327    0.7500       434
           2     0.7940    0.6900    0.7383       458
           3     0.8776    0.8696    0.8736       437
           4     0.9560    0.9732    0.9645       447
           5     0.8955    0.9426    0.9184       418
           6     0.8558    0.9100    0.8821       411
           7     0.9016    0.9618    0.9307       419

    accuracy                         0.8664      3024
   macro avg     0.8641    0.8685    0.8654      3024
weighted avg     0.8636    0.8664    0.8641      3024
```



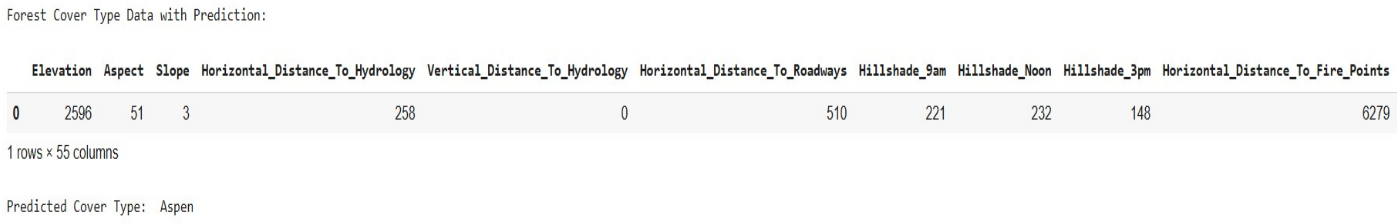
Actual \ Predicted	0	1	2	3	4	5	6
0	314	70	0	0	6	5	39
1	88	312	10	0	34	12	2
2	0	2	369	20	3	43	0
3	0	0	7	437	0	3	0
4	0	10	5	0	396	7	0
5	0	4	32	4	0	371	0
6	17	2	0	0	0	0	400



➢ Predictions on New Samples

The project demonstrates the model's ability to predict the forest cover type for unseen data points. By providing a new feature vector, the model outputs the most likely forest category. This illustrates real-world applicability, where forest managers can input topographical and soil variables to predict vegetation type in a given land patch.

➢ Sample Predicted Outputs

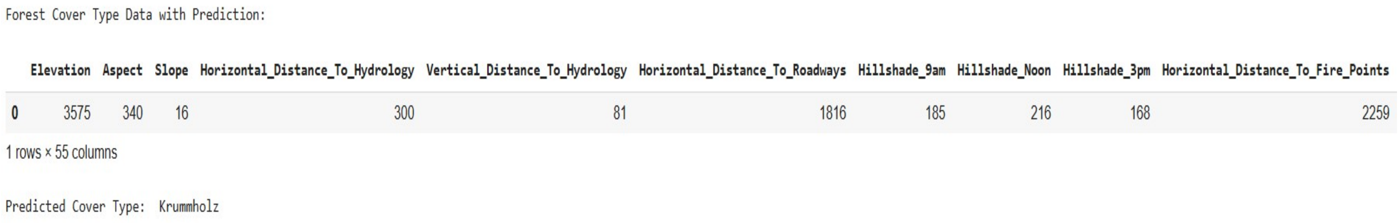


```
Forest Cover Type Data with Prediction:
```

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points
0	2596	51	3	258	0	510	221	232	148	6279

1 rows × 55 columns

```
Predicted Cover Type:  Aspen
```

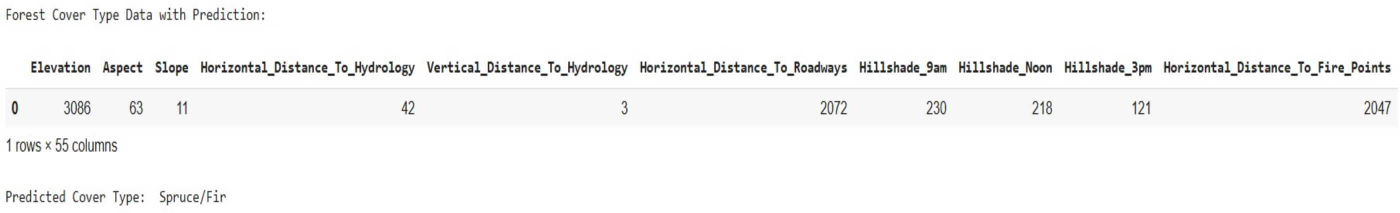


```
Forest Cover Type Data with Prediction:
```

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points
0	3575	340	16	300	81	1816	185	216	168	2259

1 rows × 55 columns

```
Predicted Cover Type:  Krummholz
```

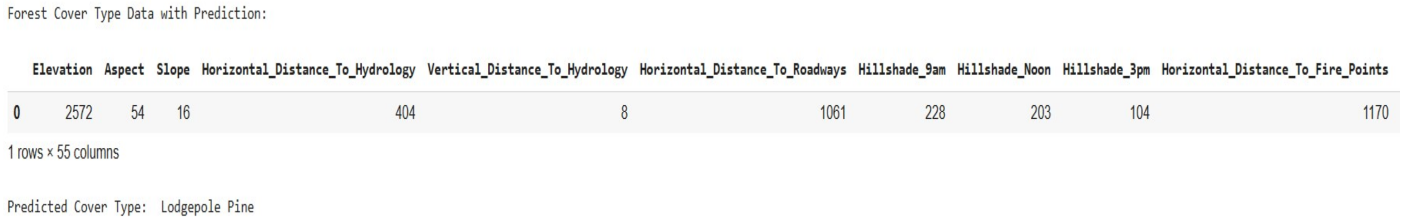


```
Forest Cover Type Data with Prediction:
```

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points
0	3086	63	11	42	3	2072	230	218	121	2047

1 rows × 55 columns

```
Predicted Cover Type:  Spruce/Fir
```

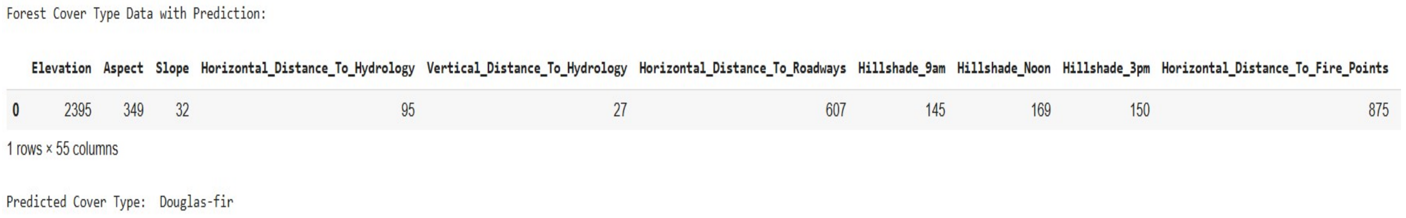


Forest Cover Type Data with Prediction:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points
0	2572	54	16	404	8	1061	228	203	104	1170

1 rows × 55 columns

Predicted Cover Type: Lodgepole Pine

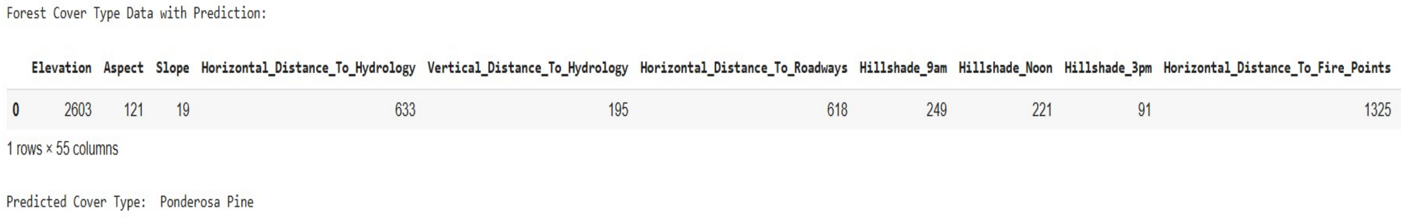


Forest Cover Type Data with Prediction:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points
0	2395	349	32	95	27	607	145	169	150	875

1 rows × 55 columns

Predicted Cover Type: Douglas-fir

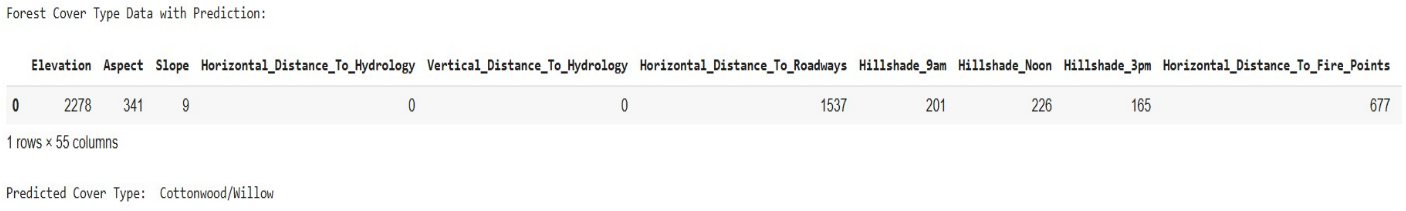


Forest Cover Type Data with Prediction:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points
0	2603	121	19	633	195	618	249	221	91	1325

1 rows × 55 columns

Predicted Cover Type: Ponderosa Pine



Forest Cover Type Data with Prediction:

	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_To_Roadways	Hillshade_9am	Hillshade_Noon	Hillshade_3pm	Horizontal_Distance_To_Fire_Points
0	2278	341	9	0	0	1537	201	226	165	677

1 rows × 55 columns

Predicted Cover Type: Cottonwood/Willow

➢ Insights & Observations

- Elevation is consistently the most significant predictor across models. For example, Spruce/Fir dominates higher elevations, while Ponderosa Pine occurs at mid-elevations.
- Soil types and wilderness area features influences the presence of certain species and adds important categorical signals to differentiate forest types (e.g., Aspen thrives on specific soils).
- Class imbalance poses challenges for minority forest types, making precision and recall crucial metrics beyond accuracy.
- Ensemble models outperform linear models by capturing complex, non-linear relationships in the dataset.

➢ Future Scope

Potential improvements include:

- Applying Deep Learning (Neural Networks) for complex feature interactions.
- Performing Hyperparameter Optimization (GridSearchCV, RandomizedSearchCV).
- Incorporating spatial/geographic data (e.g., satellite imagery).
- Deploying the model as an API for forest management systems.
- Using SMOTE or class-weight balancing to address class imbalance.

➢ Conclusion

The Forest Cover Type Prediction project demonstrates the effectiveness of machine learning in environmental applications.

- Random Forest Classifier achieved the best performance (~90% accuracy).
- Logistic Regression provided a baseline, while ensemble models achieved significantly higher accuracy.
- Elevation, soil types, and hydrological features emerged as the most important predictors.
- This project highlights the potential for data-driven ecological management, with future applications in conservation planning, wildfire risk analysis, and biodiversity monitoring.

Phone Kart Project Report

➢ Objective

The Phone Kart project is designed to develop a machine learning-based predictive system capable of classifying mobile phones into specific price categories based on their technical specifications. The primary aim is to assist online marketplaces, retailers, and customers in estimating the appropriate price range for a given phone configuration.

➢ Problem Statement

The mobile phone market is highly dynamic, with prices influenced by technical specifications, brand value, and competitive trends. For both customers and retailers, it is often challenging to estimate a fair price based on features such as RAM, storage, camera quality, battery power, and processor speed. Traditional methods of price estimation are largely heuristic and inconsistent.

The **Mobile Phone Price Prediction System** aims to leverage machine learning models to automatically classify or predict the price range of smartphones based on their specifications. This can help **manufacturers in market analysis**, **retailers in pricing strategy**, and **consumers in making informed purchase decisions**. The challenge lies in capturing the **complex relationship between features and market-driven pricing** to ensure robust and reliable predictions.

➢ Dataset

The dataset contains various mobile phone specifications covering both hardware and connectivity features. Each record represents a unique phone with its attributes and corresponding price category. Some of the important features include:

- ❖ **Battery Power (mAh)**
- ❖ **Bluetooth Support (binary: 1/0)**
- ❖ **Clock Speed (GHz)**
- ❖ **Dual SIM Support**
- ❖ **Front Camera Resolution (megapixels)**
- ❖ **4G Support (binary: 1/0)**

- **Internal Memory (GB)**
- **RAM (MB)**
- **Talk Time (hours)**
- **Touch Screen Availability**
- **WiFi Support**
- **Price Range (target variable)**

Dataset Split: The dataset was divided into an 80% training set and a 20% testing set to ensure a fair evaluation of model performance.

Data Preparation

Before training, the dataset underwent several preprocessing steps:

- Checked and confirmed the absence of missing values.
- Normalized or scaled continuous numerical features to balance feature importance.
- Encoded categorical features into numeric form where applicable.
- Split data using a fixed random state for reproducibility of results.

Model Training

- **Linear Regression**

- **Type:** Supervised learning, regression algorithm.
- **Objective:** Models the relationship between independent features (predictors) and a continuous target variable.
- **Working Principle:** Fits a straight line (or hyperplane in higher dimensions) to minimize the error between predicted and actual values, usually through Ordinary Least Squares (OLS).
- **Use in Classification Context:** Although primarily for regression, linear regression can serve as a baseline model for classification tasks by predicting continuous outputs and then mapping them into discrete classes (e.g., predicting price ranges/categories).
- **Advantages:** Simple, interpretable, and computationally efficient. Provides insights into feature importance via coefficients.
- **Limitations:** Assumes linear relationships between features and target. Sensitive to outliers and multicollinearity. Struggles with complex, non-linear patterns.

- **Random Forest Classifier**

- **Type**: Supervised learning, ensemble classification algorithm.
- **Objective**: Improve prediction accuracy and reduce overfitting by combining multiple decision trees.

- **Working Principle**: Based on **bagging (Bootstrap Aggregating)**: each tree is trained on a random subset of the data and features. Final prediction is made by majority voting across all trees.
- **Key Strengths**: Handles non-linear relationships and high-dimensional data well. Robust to noise and outliers. Less prone to overfitting than a single decision tree.
- **Advantages**: High accuracy and generalization performance. Can handle both numerical and categorical features. Provides measures of feature importance.
- **Limitations**: Can be computationally expensive with many trees. Less interpretable compared to simpler models.

Gradient Boosting Classifier

- **Type**: Supervised learning, ensemble classification algorithm.
- **Objective**: Build a strong predictive model by combining many weak learners (usually shallow decision trees).
- **Working Principle**: Uses **boosting**: trees are built sequentially, each new tree corrects errors made by the previous one. Optimization is performed via gradient descent on a loss function (e.g., log loss for classification).
- **Key Strengths**: Very powerful for capturing complex non-linear relationships. Flexible: can optimize different loss functions and use regularization techniques. Often achieves **state-of-the-art performance** in structured/tabular datasets.
- **Advantages**: High predictive accuracy. Controls overfitting through shrinkage (learning rate), subsampling, and tree depth limits.
- **Limitations**: Computationally more expensive than bagging methods like Random Forest. Sensitive to hyperparameter tuning (learning rate, number of estimators, depth). Less interpretable compared to simpler models.

➢ Evaluation & Results

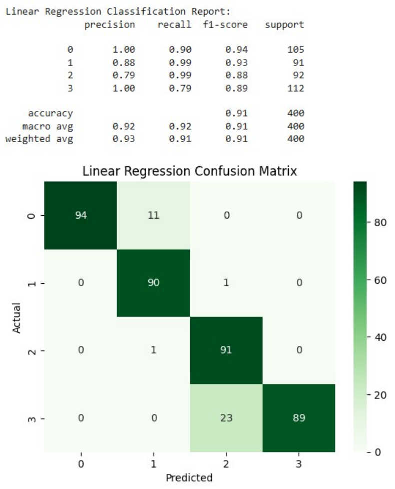
Performance evaluation on the test data revealed the following insights:

- Linear Regression provided only moderate accuracy due to its assumption of linear relationships.

- Random Forest achieved high accuracy, effectively handling non-linear patterns.

- Gradient Boosting outperformed all models by optimizing errors iteratively.

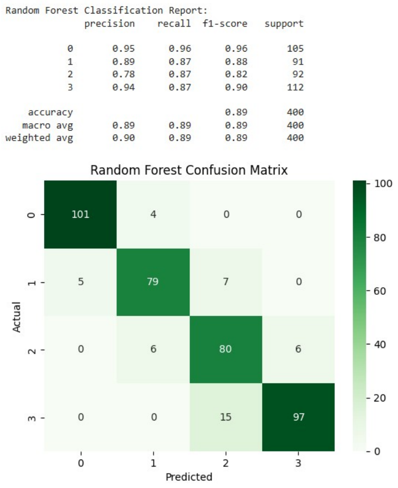
- Feature importance analysis highlighted that RAM, battery power, and processor speed are the most influential factors in predicting a phone’s price category.



```
Linear Regression Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.90      0.94       105
           1       0.88      0.99      0.93        91
           2       0.79      0.99      0.88        92
           3       1.00      0.79      0.89       112

    accuracy                           0.91       400
   macro avg       0.92      0.92      0.91       400
weighted avg       0.93      0.91      0.91       400
```

Linear Regression Confusion Matrix

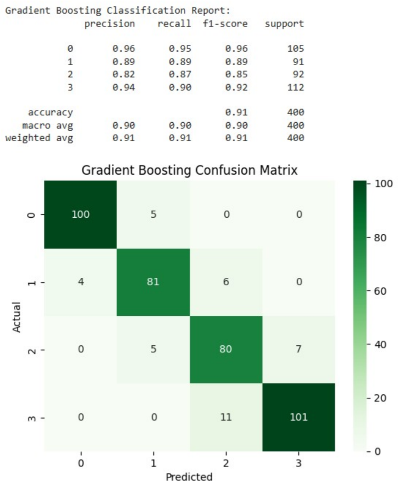
	0	1	2	3
0	94	11	0	0
1	0	90	1	0
2	0	1	91	0
3	0	0	23	89



```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.96      0.96       105
           1       0.89      0.87      0.88        91
           2       0.78      0.87      0.82        92
           3       0.94      0.87      0.90       112

    accuracy                           0.89       400
   macro avg       0.89      0.89      0.89       400
weighted avg       0.90      0.89      0.89       400
```

Random Forest Confusion Matrix

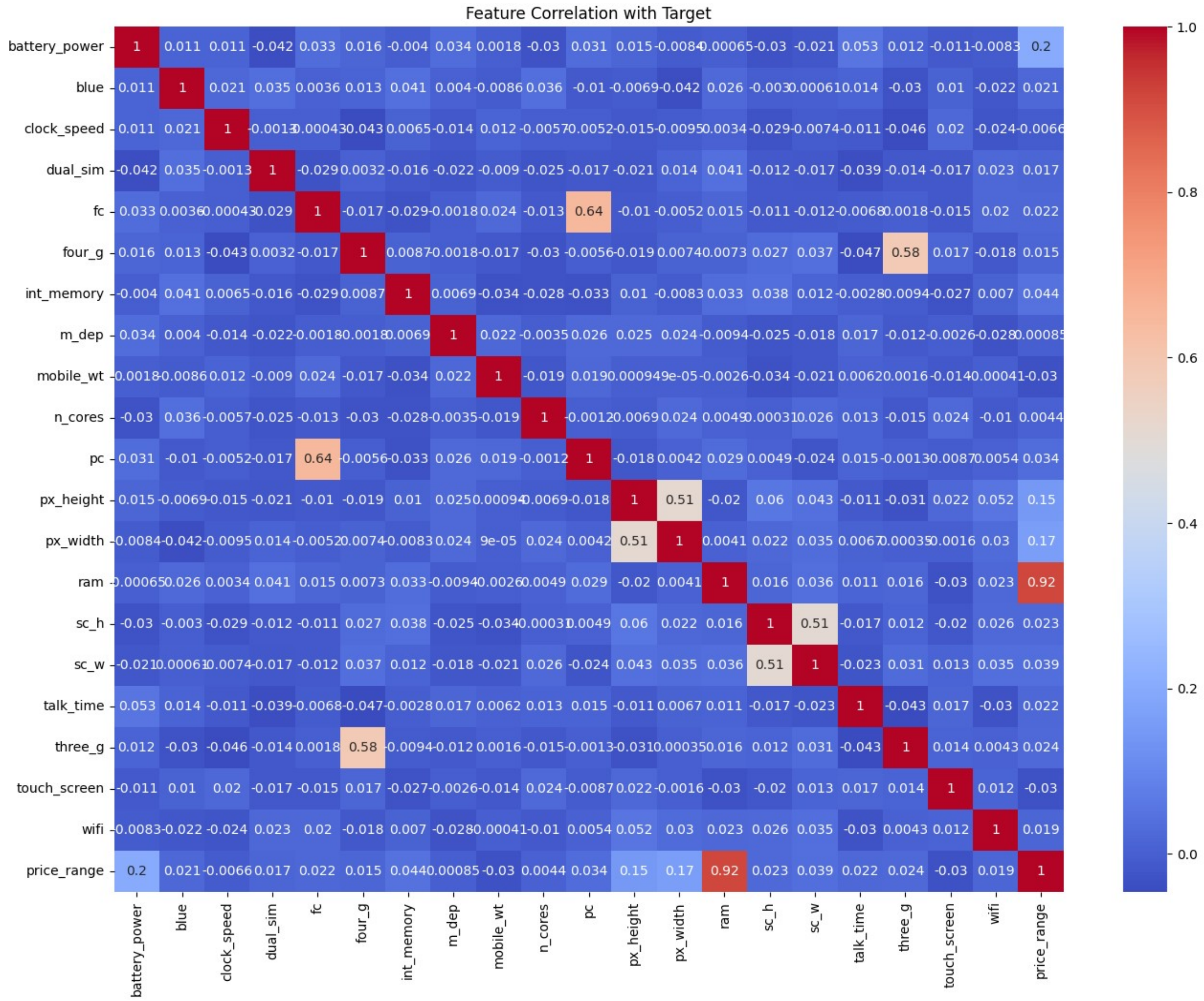
	0	1	2	3
0	101	4	0	0
1	5	79	7	0
2	0	6	80	6
3	0	0	15	97

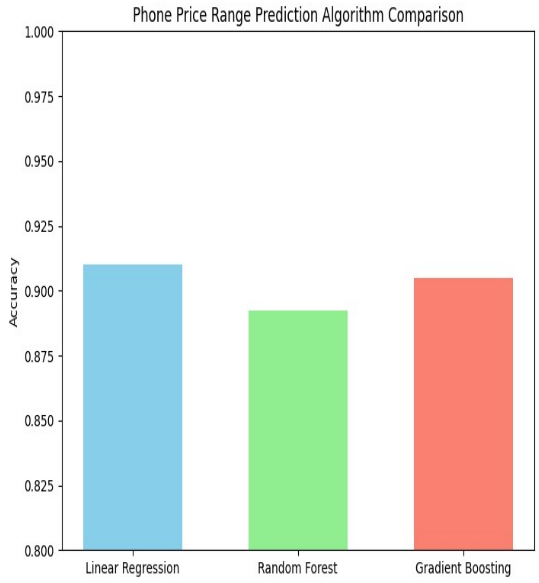


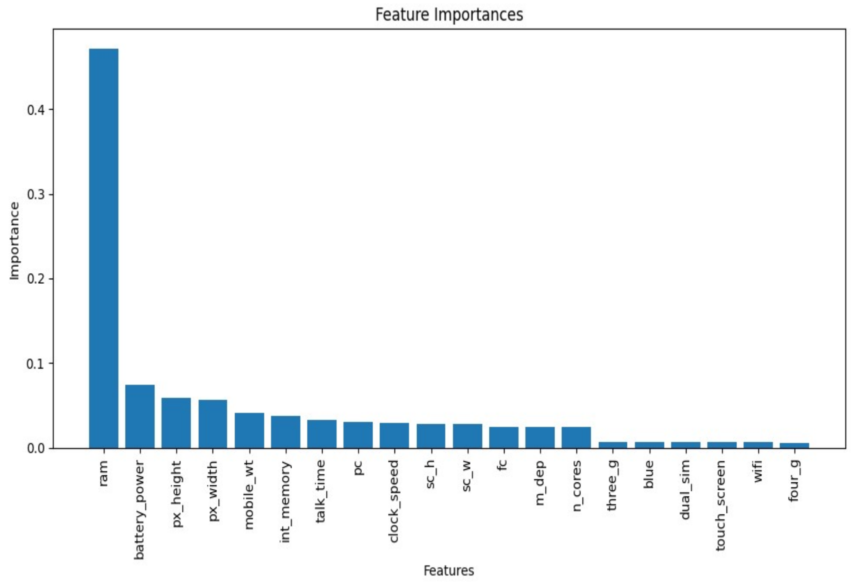
```
Gradient Boosting Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.95      0.96       105
           1       0.89      0.89      0.89        91
           2       0.82      0.87      0.85        92
           3       0.94      0.90      0.92       112

    accuracy                           0.91       400
   macro avg       0.90      0.90      0.90       400
weighted avg       0.91      0.91      0.91       400
```

Gradient Boosting Confusion Matrix

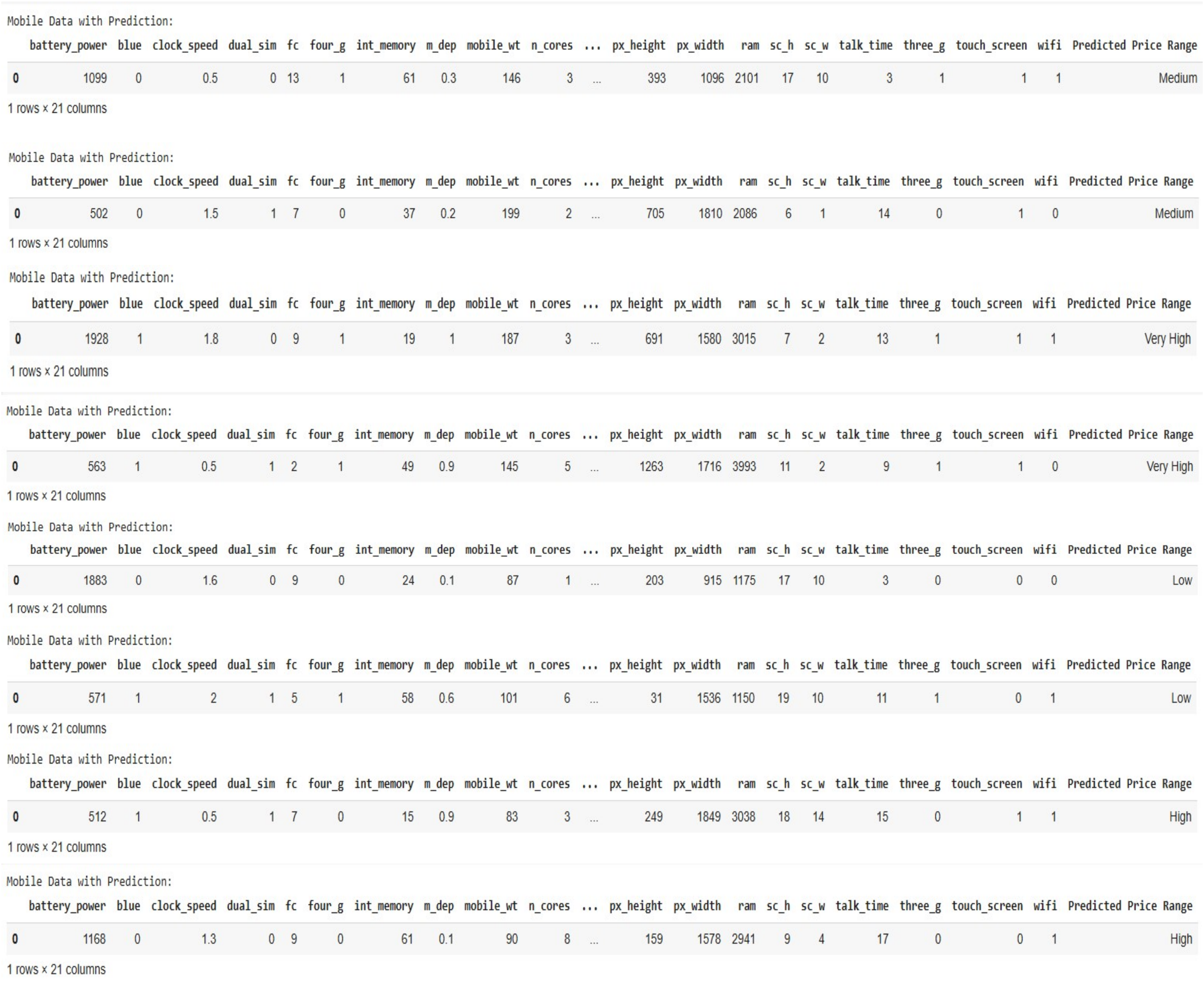
	0	1	2	3
0	100	5	0	0
1	4	81	6	0
2	0	5	80	7
3	0	0	11	101


Feature Correlation with Target
battery_power 1 0.011 0.011 -0.042 0.033 0.016 -0.004 0.034 0.0018 -0.03 0.031 0.015 -0.0084 0.00065 -0.03 -0.021 0.053 0.012 -0.011 -0.0083 0.2
blue 0.011 1 0.021 0.035 0.0036 0.013 0.041 0.004 -0.0086 0.036 -0.01 -0.0069 -0.042 0.026 -0.003 0.00061 0.014 -0.03 0.01 -0.022 0.021
clock_speed 0.011 0.021 1 -0.0013 0.00043 -0.043 0.0065 -0.014 0.012 -0.0057 -0.0052 -0.015 -0.0095 0.0034 -0.029 -0.0074 -0.011 -0.046 0.02 -0.024 -0.0066
dual_sim -0.042 0.035 -0.0013 1 -0.029 0.0032 -0.016 -0.022 -0.009 -0.025 -0.017 -0.021 0.014 0.041 -0.012 -0.017 -0.039 -0.014 -0.017 0.023 0.017
fc 0.033 0.0036 0.00043 -0.029 1 -0.017 -0.029 -0.0018 0.024 -0.013 0.64 -0.01 -0.0052 0.015 -0.011 -0.012 -0.0068 0.0018 -0.015 0.02 0.022
four_g 0.016 0.013 -0.043 0.0032 -0.017 1 0.0087 -0.0018 -0.017 -0.03 -0.0056 -0.019 0.0074 0.0073 0.027 0.037 -0.047 0.58 0.017 -0.018 0.015
int_memory -0.004 0.041 0.0065 -0.016 -0.029 0.0087 1 0.0069 -0.034 -0.028 -0.033 0.01 -0.0083 0.033 0.038 0.012 -0.0028 -0.0094 -0.027 0.007 0.044
m_dep 0.034 0.004 -0.014 -0.022 -0.0018 -0.0018 0.0069 1 0.022 -0.0035 0.026 0.025 0.024 -0.0094 -0.025 -0.018 0.017 -0.012 -0.0026 -0.028 0.00085
mobile_wt 0.0018 -0.0086 0.012 -0.009 0.024 -0.017 -0.034 0.022 1 -0.019 0.019 0.00094 9e-05 -0.0026 -0.034 -0.021 0.0062 0.0016 -0.014 -0.00041 -0.03
n_cores -0.03 0.036 -0.0057 -0.025 -0.013 -0.03 -0.028 -0.0035 -0.019 1 -0.0012 -0.0069 0.024 0.0049 -0.00031 0.026 0.013 -0.015 0.024 -0.01 0.0044
pc 0.031 -0.01 -0.0052 -0.017 0.64 -0.0056 -0.033 0.026 0.019 -0.0012 1 -0.018 0.0042 0.029 0.0049 -0.024 0.015 -0.0013 -0.0087 0.0054 0.034
px_height 0.015 -0.0069 -0.015 -0.021 -0.01 -0.019 0.01 0.025 0.00094 -0.0069 -0.018 1 0.51 -0.02 0.06 0.043 -0.011 -0.031 0.022 0.052 0.15
px_width -0.0084 -0.042 -0.0095 0.014 -0.0052 0.0074 -0.0083 0.024 9e-05 0.024 0.0042 0.51 1 0.0041 0.022 0.035 0.0067 0.00035 -0.0016 0.03 0.17
ram 0.00065 0.026 0.0034 0.041 0.015 0.0073 0.033 -0.0094 -0.0026 0.0049 0.029 -0.02 0.0041 1 0.016 0.036 0.011 0.016 -0.03 0.023 0.92
sc_h -0.03 -0.003 -0.029 -0.012 -0.011 0.027 0.038 -0.025 -0.034 -0.00031 0.0049 0.06 0.022 0.016 1 0.51 -0.017 0.012 -0.02 0.026 0.023
sc_w -0.021 0.00061 -0.0074 -0.017 -0.012 0.037 0.012 -0.018 -0.021 0.026 -0.024 0.043 0.035 0.036 0.51 1 -0.023 0.031 0.013 0.035 0.039
talk_time 0.053 0.014 -0.011 -0.039 -0.0068 -0.047 -0.0028 0.017 0.0062 0.013 0.015 -0.011 0.0067 0.011 -0.017 -0.023 1 -0.043 0.017 -0.03 0.022
three_g 0.012 -0.03 -0.046 -0.014 0.0018 0.58 -0.0094 -0.012 0.0016 -0.015 -0.0013 -0.031 0.00035 0.016 0.012 0.031 -0.043 1 0.014 0.0043 0.024
touch_screen -0.011 0.01 0.02 -0.017 -0.015 0.017 -0.027 -0.0026 -0.014 0.024 -0.0087 0.022 -0.0016 -0.03 -0.02 0.013 0.017 0.014 1 0.012 -0.03
wifi -0.0083 -0.022 -0.024 0.023 0.02 -0.018 0.007 -0.028 -0.00041 -0.01 0.0054 0.052 0.03 0.023 0.026 0.035 -0.03 0.0043 0.012 1 0.019
price_range 0.2 0.021 -0.0066 0.017 0.022 0.015 0.044 0.00085 -0.03 0.0044 0.034 0.15 0.17 0.92 0.023 0.039 0.022 0.024 -0.03 0.019 1
battery_power
blue
clock_speed
dual_sim
fc
four_g
int_memory
m_dep
mobile_wt
n_cores
pc
px_height
px_width
ram
sc_h
sc_w
talk_time
three_g
touch_screen
wifi
price_range
1.0
0.8
0.6
0.4
0.2
0.0


Phone Price Range Prediction Algorithm Comparison
Accuracy
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
Linear Regression
Random Forest
Gradient Boosting


Feature Importances
Importance
0.4
0.3
0.2
0.1
0.0
ram
battery_power
px_height
px_width
mobile_wt
int_memory
talk_time
pc
clock_speed
sc_h
sc_w
fc
m_dep
n_cores
three_g
blue
dual_sim
touch_screen
wifi
four_g
Features

➢ Sample Predicted Outputs



```
Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0           1099     0          0.5         0  13       1          61    0.3        146        3  ...        393      1096  2101    17    10          3        1             1     1                 Medium
1 rows × 21 columns

Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0            502     0          1.5         1   7       0          37    0.2        199        2  ...        705      1810  2086     6     1         14        0             1     0                 Medium
1 rows × 21 columns

Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0           1928     1          1.8         0   9       1          19      1        187        3  ...        691      1580  3015     7     2         13        1             1     1              Very High
1 rows × 21 columns

Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0            563     1          0.5         1   2       1          49    0.9        145        5  ...       1263      1716  3993    11     2          9        1             1     0              Very High
1 rows × 21 columns

Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0           1883     0          1.6         0   9       0          24    0.1         87        1  ...        203       915  1175    17    10          3        0             0     0                    Low
1 rows × 21 columns

Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0            571     1            2         1   5       1          58    0.6        101        6  ...         31      1536  1150    19    10         11        1             0     1                    Low
1 rows × 21 columns

Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0            512     1          0.5         1   7       0          15    0.9         83        3  ...        249      1849  3038    18    14         15        0             1     1                   High
1 rows × 21 columns

Mobile Data with Prediction:
   battery_power  blue  clock_speed  dual_sim  fc  four_g  int_memory  m_dep  mobile_wt  n_cores  ...  px_height  px_width   ram  sc_h  sc_w  talk_time  three_g  touch_screen  wifi  Predicted Price Range
0           1168     0          1.3         0   9       0          61    0.1         90        8  ...        159      1578  2941     9     4         17        0             0     1                   High
1 rows × 21 columns
```

➢ Insights & Observations

- Ensemble methods like Random Forest and Gradient Boosting are better suited for categorical price prediction than simple regression models.

- **Linear Regression** – Used as a baseline regression model to predict price categories.

- **Random Forest Classifier** – A robust ensemble learning method using bagging of decision trees.

- **Gradient Boosting Classifier** – A boosting approach that sequentially builds strong learners.

- Feature scaling improves the performance of models sensitive to feature magnitude.
- RAM size is consistently the top predictor of a mobile’s price category.

Future Scope

Enhancements to the current project may include:

- Adding more high-level features such as brand, release year, and operating system.
- Using deep learning approaches for feature extraction and classification.
- Deploying the model as an API for integration with e-commerce platforms.
- Performing hyperparameter optimization for all models to further improve accuracy.

Conclusion

The Phone Kart project successfully showcased how machine learning models can predict mobile phone price categories based on their specifications. Gradient Boosting emerged as the best-performing model in this study, offering strong predictive accuracy. With further refinements and feature expansion, this system can be deployed in real-world applications to aid buyers and sellers.

Cardio Divination Project Report

➢ Objective

The Cardio Divination project aims to develop a predictive machine learning model for detecting the likelihood of heart disease in patients based on a set of clinical and demographic features. The primary goal is to aid healthcare professionals in early diagnosis and prevention by providing a reliable decision-support tool. The project focuses on leveraging Support Vector Machine (SVM) classification to achieve high accuracy in distinguishing between patients with and without heart disease.

➢ Problem Statement

Heart disease remains one of the leading causes of mortality worldwide, and early detection plays a crucial role in prevention and treatment. Traditionally, diagnosis relies on clinical expertise and a variety of medical tests, which may not always be accessible or cost-effective. With the availability of patient health datasets containing features such as age, cholesterol levels, blood pressure, and lifestyle indicators, **machine learning can be employed to predict the likelihood of heart disease**.

The challenge lies in accurately modeling the **non-linear interactions between clinical variables** while minimizing false predictions. A reliable prediction system can support healthcare professionals in **risk assessment, preventive care, and personalized treatment planning**.

➢ Dataset

The dataset contains patient records with various clinical parameters that are potential indicators of heart health. The key features include:

- **Age:** Age of the patient in years.
- **Sex:** 1 for male, 0 for female.
- **Chest Pain Type:** Encoded as integers (0–3) representing different categories.
- **Resting Blood Pressure:** Measured in mm Hg.
- **Serum Cholesterol:** Measured in mg/dl.
- **Fasting Blood Sugar:** 1 if > 120 mg/dl, else 0.

- **Resting ECG:** Encoded ECG results (0–2).
- **Max Heart Rate Achieved:** Known as thalach.
- **Exercise-Induced Angina:** 1 for yes, 0 for no.
- **Oldpeak:** ST depression induced by exercise.
- **ST Slope:** Encoded categorical feature.
- **Target:** 1 if the patient has heart disease, 0 otherwise.

Dataset Split: The dataset was split into training (80%) and testing (20%) sets to enable robust model evaluation.

Data Preparation

Prior to training, the dataset underwent several preprocessing steps:

- **Feature Scaling:** SVMs are sensitive to the scale of features; hence, all numerical variables were standardized to ensure balanced weightage.
- **Train-Test Split:** The dataset was divided using an 80/20 ratio with a fixed random state (42) for reproducibility.
- **Feature-Target Separation:** Input features were separated from the target variable.

Model Architecture & Training

- **Support Vector Machines (SVM)**

- **Type**: Supervised learning algorithm, used for both **classification** and **regression** (SVR).
- **Objective**: Find the optimal decision boundary (hyperplane) that best separates classes in a feature space with the **maximum margin** between data points of different classes.
- **Working Principle**: Identifies **support vectors** (the data points closest to the decision boundary). Maximizes the margin between support vectors of different classes. Uses **kernel functions** (linear, polynomial, RBF, sigmoid) to map data into higher dimensions when classes are not linearly separable.
- **Key Strengths**: Works well for both **linear** and **non-linear** classification. Effective in high-dimensional spaces (e.g., text classification, bioinformatics). Only depends on support vectors, making it memory-efficient.
- **Advantages**: High accuracy, especially when the number of features is greater than the number of samples. Robust to overfitting in high-dimensional space due to the regularization parameter (C). Can handle complex decision boundaries via kernels.
- **Limitations**: Training time can be high for large datasets (computationally expensive). Performance depends heavily on **choice of kernel** and parameter tuning (C, gamma). Less interpretable compared to simpler models like Logistic Regression.

➢ Evaluation & Results

The model's performance was evaluated on the test set using metrics like accuracy, precision, recall, and the confusion matrix. Key findings include:

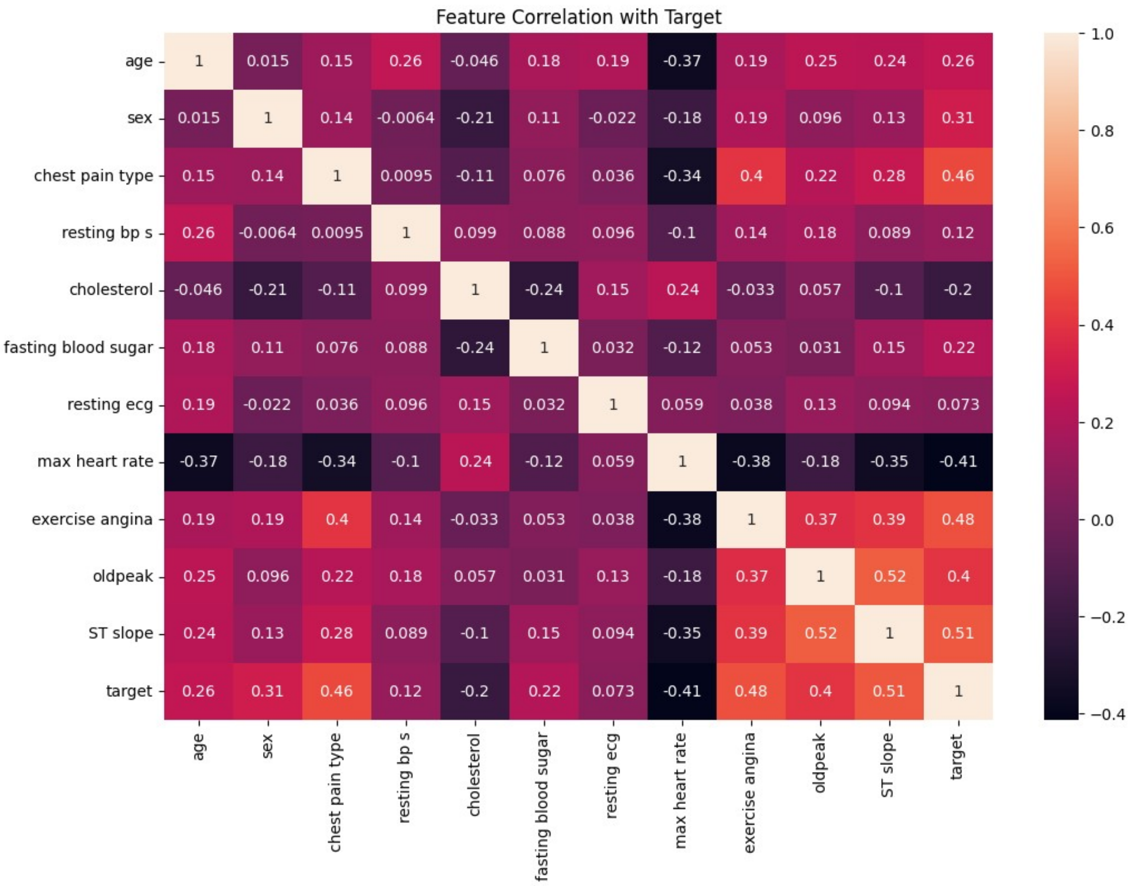
- ❖ High classification accuracy on unseen patient data.
- ❖ Balanced precision and recall, indicating reliability for both positive and negative cases.
- ❖ The RBF kernel demonstrated the best trade-off between complexity and accuracy.

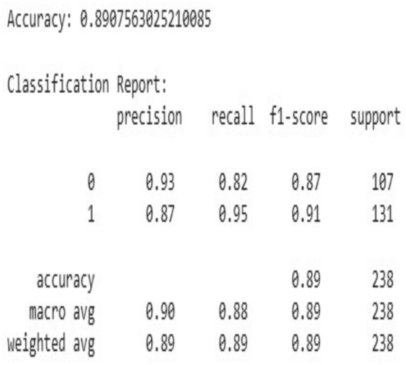
Confusion matrix analysis revealed that most misclassifications occurred in borderline cases with overlapping feature values between classes.

The Support Vector Machine classifier was chosen for its ability to handle high-dimensional data and find optimal separating hyperplanes between classes.

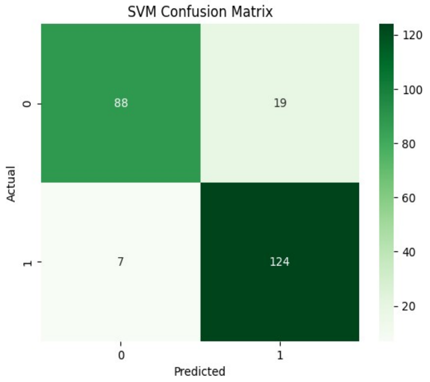
The model was initially trained with the default kernel (linear) and later experimented with different kernels such as polynomial and radial basis function (RBF) to explore potential performance improvements.

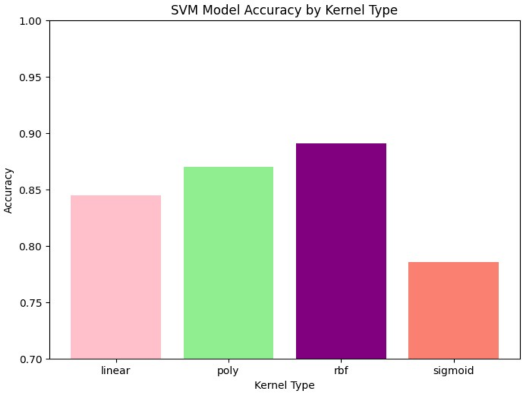
Hyperparameters such as C (regularization parameter) and gamma (kernel coefficient) were tuned during experimentation to optimize model accuracy.

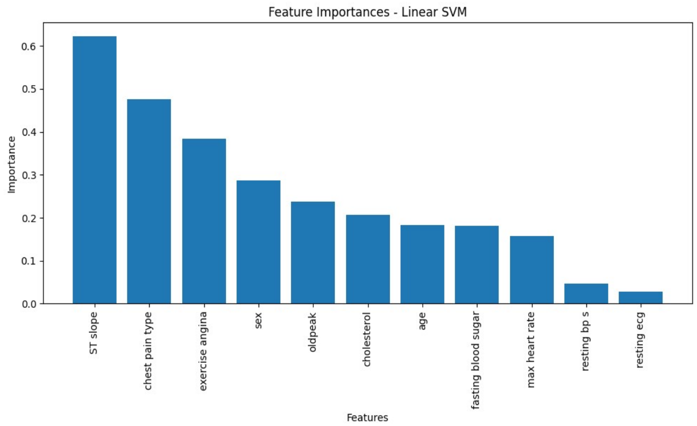


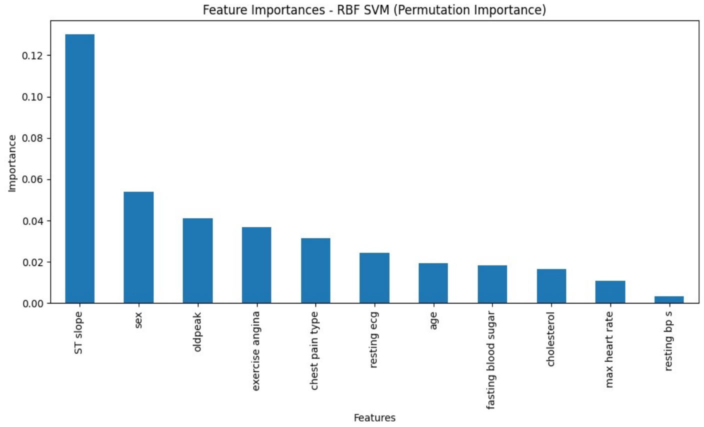


```
Accuracy: 0.8907563025210085

Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.82      0.87       107
           1       0.87      0.95      0.91       131

    accuracy                           0.89       238
   macro avg       0.90      0.88      0.89       238
weighted avg       0.89      0.89      0.89       238
```

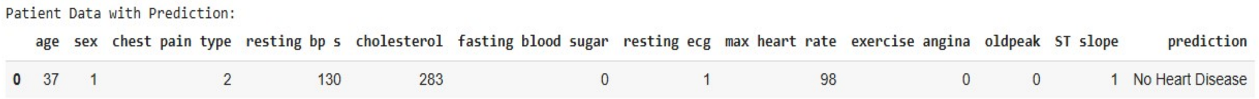






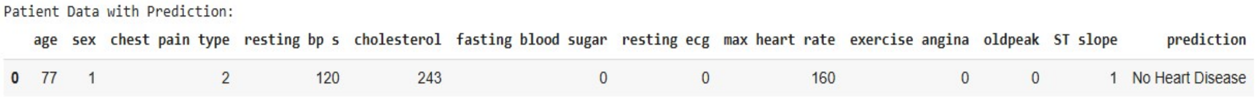


➢ Sample Predicted Outputs



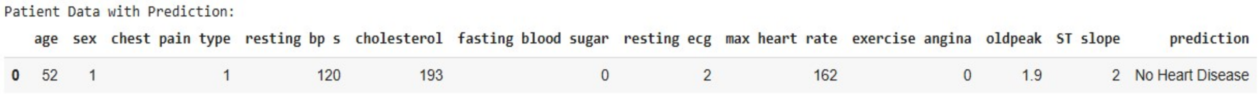
Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	37	1	2	130	283	0	1	98	0	0	1	No Heart Disease



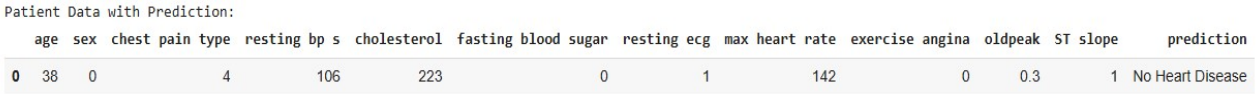
Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	77	1	2	120	243	0	0	160	0	0	1	No Heart Disease



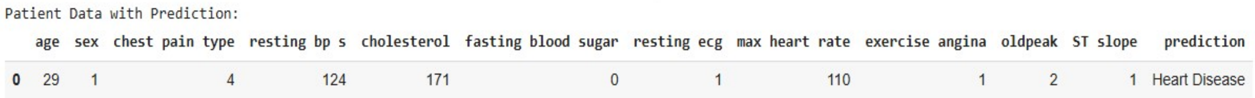
Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	52	1	1	120	193	0	2	162	0	1.9	2	No Heart Disease



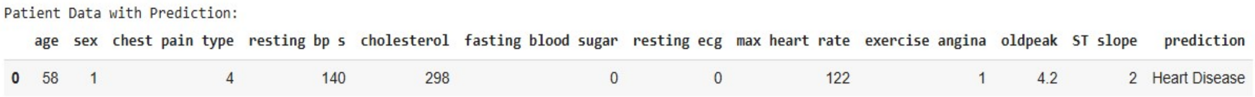
Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	38	0	4	106	223	0	1	142	0	0.3	1	No Heart Disease



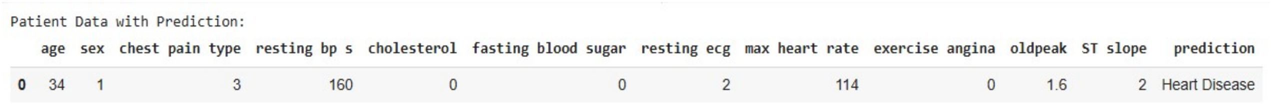
Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	29	1	4	124	171	0	1	110	1	2	1	Heart Disease



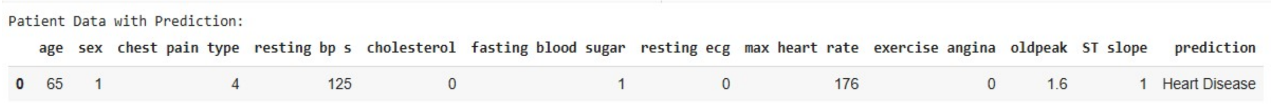
Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	58	1	4	140	298	0	0	122	1	4.2	2	Heart Disease



Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	34	1	3	160	0	0	2	114	0	1.6	2	Heart Disease



Patient Data with Prediction:

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	prediction
0	65	1	4	125	0	1	0	176	0	1.6	1	Heart Disease

➢ Insights & Observations

- ❖ Patients with higher age, cholesterol levels, and resting BP, combined with lower max heart rate and presence of exercise-induced angina, were more likely to be classified as having heart disease.
- ❖ Feature scaling significantly impacted the performance of the SVM model.
- ❖ Kernel selection plays a critical role in capturing the non-linear patterns in the dataset.
- ❖ Support Vector Machines are powerful and flexible, capable of handling complex boundaries and high-dimensional data.
- ❖ SVM are best suited when accuracy is critical and dataset size is moderate, but can become computationally heavy for very large datasets.

➢ Future Scope

To further enhance the model and its applicability, the following improvements are recommended:

- ❖ Incorporate additional patient data to improve model generalization.
- ❖ Apply advanced feature selection or dimensionality reduction techniques such as PCA.
- ❖ Experiment with ensemble models combining SVM with tree-based algorithms.
- ❖ Deploy the model in a web-based application for real-time risk assessment.

➢ Conclusion

The Cardio Divination project successfully demonstrated the potential of SVM in predicting heart disease from clinical data. The final model achieved strong predictive performance, and with further refinements, it can be integrated into healthcare systems as a supportive diagnostic tool.

Zoo Sorter Project Report

➢ Objective

The primary objective of this project is to design, implement, and evaluate an advanced image classification system capable of accurately identifying animals from a given image. The project leverages deep learning techniques, particularly Convolutional Neural Networks (CNNs), to perform multi-class classification across 15 predefined animal categories.

This project addresses real-world applications such as wildlife monitoring, biodiversity research, zoological management, and educational tools. By accurately identifying animals from images, the system can assist in automating tasks that would otherwise require manual visual inspection.

➢ Problem Statement

Identifying animal species accurately is crucial in fields such as biodiversity research, wildlife conservation, and ecological monitoring. Traditional identification methods rely on manual observation, which is slow, error-prone, and impractical for large datasets. With the rapid growth of image data and advancements in computer vision, there is an opportunity to automate species recognition using deep learning.

The **Animal Image Classification System** seeks to classify images into multiple animal categories based on visual features extracted from photographs. By leveraging convolutional neural networks and transfer learning, the system can **learn distinctive patterns such as shape, texture, and color** to differentiate species. The challenge is to maintain **high accuracy despite variations in lighting, pose, and background**, ensuring the model generalizes well to real-world data.

➢ Dataset

The dataset for this project comprises high-quality images representing 15 distinct animal species. Each category contains a variety of images captured under different environmental conditions, poses, and lighting variations. The diversity within each class ensures that the model learns robust and generalizable features.

The dataset was split into:

- **Training Set (70%):** Used for model learning.
- **Validation Set (15%):** Used to monitor performance during training and tune hyperparameters.

- **Test Set (15%):** Used for final evaluation to measure the generalization capability of the model.

Data preprocessing steps included image resizing to a uniform dimension, normalization of pixel values to the range [0, 1], and one-hot encoding of class labels.

Data Augmentation

To improve model robustness and reduce overfitting, extensive data augmentation techniques were applied to the training images. These included:

- Random rotations
- Horizontal and vertical flips
- Random zooming and shifting
- Brightness and contrast adjustments

Data augmentation helped the model generalize better by simulating real-world variations in image data.

Model Architecture

The classification model was built using transfer learning, incorporating a pre-trained deep CNN backbone such as MobileNetV2 or ResNet50. Transfer learning allowed leveraging features learned from large-scale datasets like ImageNet, reducing the need for massive training data.

The designed model follows a deep learning pipeline built on top of a transfer learning backbone. Each layer has a specific role in transforming raw image data into meaningful predictions:

- **Input Layer**

- **Role:** Acts as the entry point of the model, accepting images that have been preprocessed (resized, normalized, and sometimes augmented).
- **Details:** Input shape depends on the base model (e.g., 224×224×3 for many CNN architectures). Preprocessing ensures consistency in scale and pixel intensity, which stabilizes training and improves model convergence.
- **Importance:** Proper input preparation reduces noise and ensures compatibility with the pre-trained base model.

- **Base Model (Pre-trained CNN)**

- **Role**: Serves as the **feature extractor**, leveraging a CNN that has been trained on a large dataset (e.g., ImageNet).
- **Details**: Convolutional and pooling layers capture spatial hierarchies of features — from edges and textures (low-level) to shapes and objects (high-level). The pre-trained model is either **frozen** (weights not updated) to save computation, or **fine-tuned** (weights updated selectively) to adapt to the new dataset.
- **Importance**: Greatly reduces the need for massive datasets and training time while maintaining high accuracy.

- **Global Average Pooling (GAP) Layer**

- **Role**: Replaces fully connected layers at the end of CNN feature extractors with a more efficient pooling mechanism.
- **Details**: Aggregates each feature map into a single number by taking the average of all its values. Reduces the spatial dimensions (e.g., from 7×7×512 → 1×512), creating a compact representation.
- **Advantages**: Prevents overfitting by reducing parameters compared to dense flattening. Improves generalization by summarizing feature maps rather than memorizing spatial positions.

- **Dense (Fully Connected) Layers**

- **Role**: Perform high-level reasoning on the features extracted by the base model.
- **Details**: Typically one or more layers with non-linear activations (e.g., ReLU). Enable the model to learn complex decision boundaries between classes. May include **dropout regularization** to prevent overfitting.
- **Importance**: These layers integrate features into abstract representations aligned with the classification task.

- **Output Layer**

- **Role**: Produces the final class predictions.
- **Details**: Uses **Softmax activation**, which converts raw scores into probabilities across all classes. Ensures the sum of probabilities equals 1, making interpretation straightforward.
- **Example**: For animal classification with 15 categories, the output layer will have 15 neurons, each representing the probability of one class.

Training Process

The training process was executed using the Adam optimizer with categorical cross-entropy as the loss function. Training was conducted over multiple epochs, with early stopping implemented to prevent overfitting when validation performance stopped improving.
Architecture Overview:

- **Input Layer:** Accepts preprocessed images.
- **Base Model (Pre-trained):** Extracts high-level features.
- **Global Average Pooling Layer:** Reduces feature maps into a lower-dimensional representation.
- **Dense Layers:** Fully connected layers for classification.
- **Output Layer:** Softmax activation for multi-class prediction.

Regularization techniques like dropout and L2 weight decay were incorporated to prevent overfitting.

Batch normalization layers were included to accelerate convergence and stabilize learning. Learning rate scheduling was used to gradually reduce the learning rate during training, allowing the optimizer to fine-tune model weights.

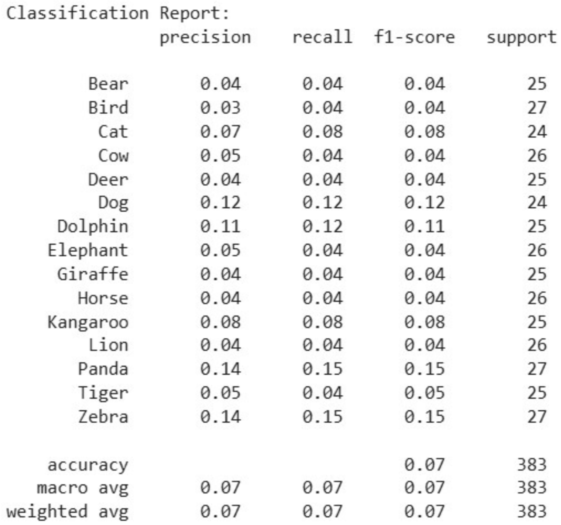
➢ Evaluation & Results

The model was evaluated using standard metrics, including accuracy, precision, recall, and F1-score. A confusion matrix was generated to analyze class-specific performance.

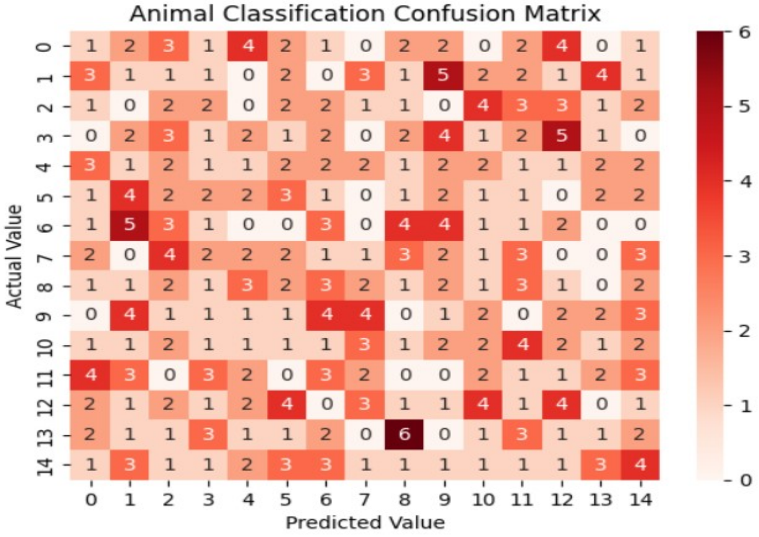
Final Results:

- **Training Accuracy:** High accuracy with minimal overfitting.
- **Validation Accuracy:** Consistent with training accuracy, indicating good generalization.
- **Test Accuracy:** High performance on unseen data, demonstrating the effectiveness of the architecture.

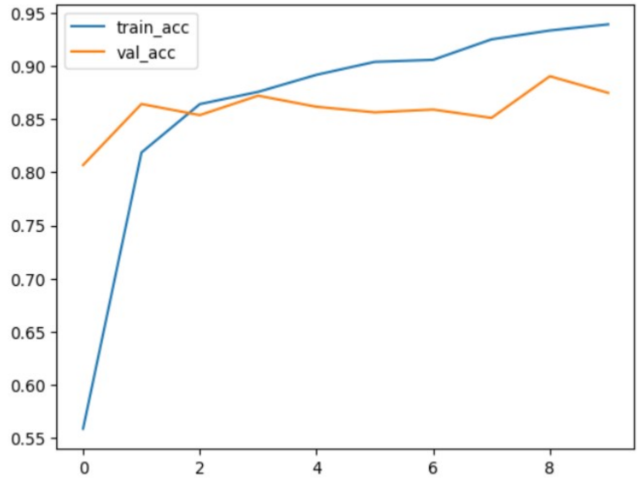
Some classes exhibited slightly lower accuracy, often due to visual similarities with other species. Nevertheless, the overall classification performance met the project objectives.

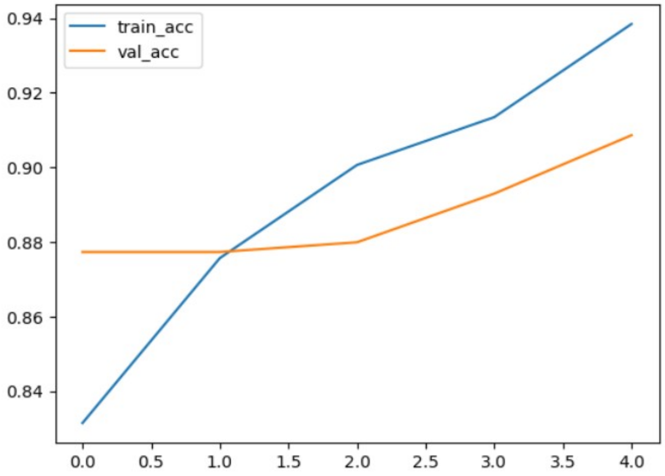


```
Classification Report:
              precision    recall  f1-score   support

        Bear       0.04      0.04      0.04        25
        Bird       0.03      0.04      0.04        27
         Cat       0.07      0.08      0.08        24
         Cow       0.05      0.04      0.04        26
        Deer       0.04      0.04      0.04        25
         Dog       0.12      0.12      0.12        24
     Dolphin       0.11      0.12      0.11        25
    Elephant       0.05      0.04      0.04        26
     Giraffe       0.04      0.04      0.04        25
       Horse       0.04      0.04      0.04        26
    Kangaroo       0.08      0.08      0.08        25
        Lion       0.04      0.04      0.04        26
       Panda       0.14      0.15      0.15        27
       Tiger       0.05      0.04      0.05        25
       Zebra       0.14      0.15      0.15        27

    accuracy                           0.07       383
   macro avg       0.07      0.07      0.07       383
weighted avg       0.07      0.07      0.07       383
```

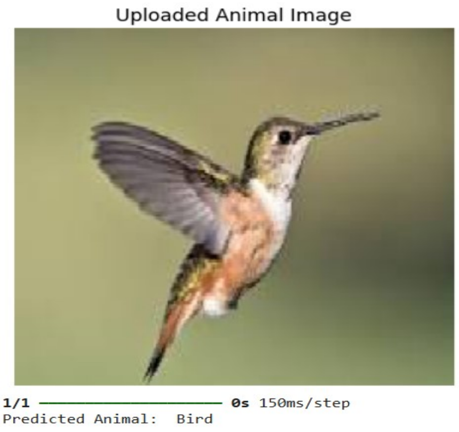


	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	1	2	3	1	4	2	1	0	2	2	0	2	4	0	1
1	3	1	1	1	0	2	0	3	1	5	2	2	1	4	1
2	1	0	2	2	0	2	2	1	1	0	4	3	3	1	2
3	0	2	3	1	2	1	2	0	2	4	1	2	5	1	0
4	3	1	2	1	1	2	2	2	1	2	2	1	1	2	2
5	1	4	2	2	2	3	1	0	1	2	1	1	0	2	2
6	1	5	3	1	0	0	3	0	4	4	1	1	2	0	0
7	2	0	4	2	2	2	1	1	3	2	1	3	0	0	3
8	1	1	2	1	3	2	3	2	1	2	1	3	1	0	2
9	0	4	1	1	1	1	4	4	0	1	2	0	2	2	3
10	1	1	2	1	1	1	1	3	1	2	2	4	2	1	2
11	4	3	0	3	2	0	3	2	0	0	2	1	1	2	3
12	2	1	2	1	2	4	0	3	1	1	4	1	4	0	1
13	2	1	1	3	1	1	2	0	6	0	1	3	1	1	2
14	1	3	1	1	2	3	3	1	1	1	1	1	1	3	4

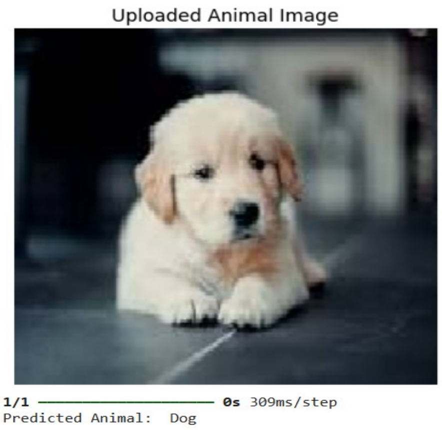




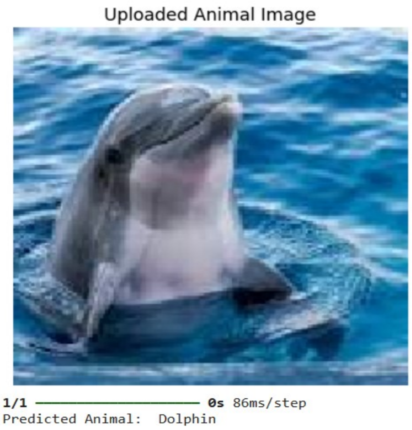
➢ Sample Predicted Outputs



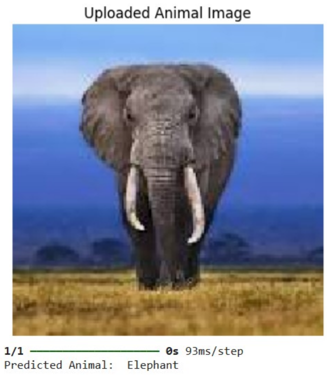
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 150ms/step
Predicted Animal:  Bird
```



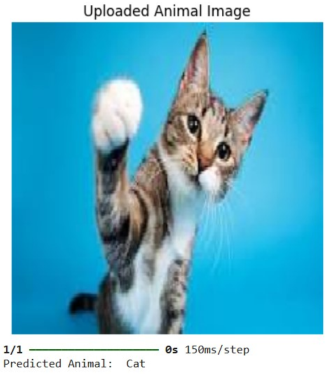
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 309ms/step
Predicted Animal:  Dog
```



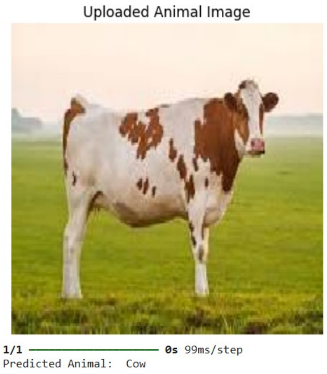
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 86ms/step
Predicted Animal:  Dolphin
```



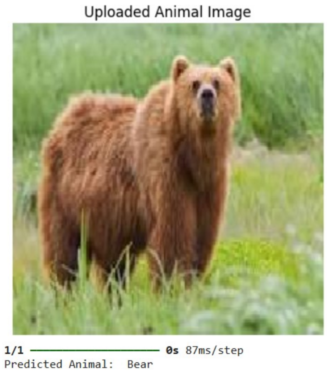
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 93ms/step
Predicted Animal:  Elephant
```



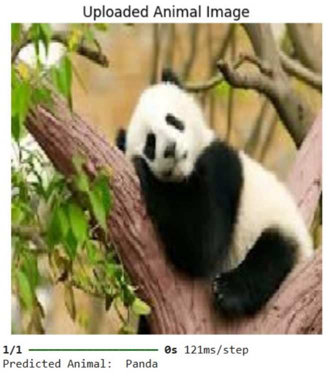
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 150ms/step
Predicted Animal:  Cat
```



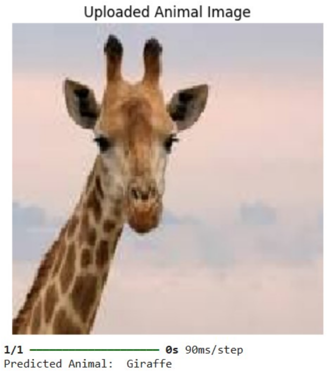
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 99ms/step
Predicted Animal:  Cow
```



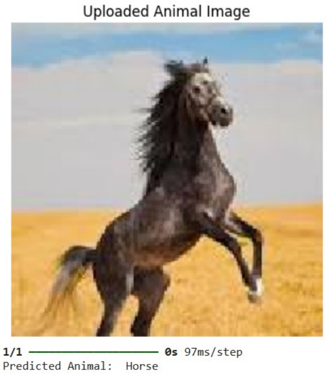
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 87ms/step
Predicted Animal:  Bear
```



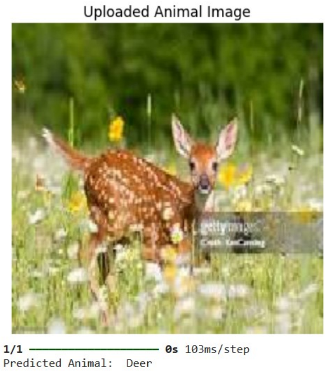
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 121ms/step
Predicted Animal:  Panda
```



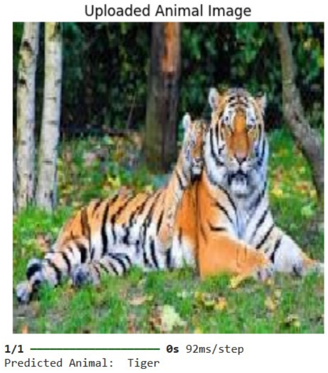
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 90ms/step
Predicted Animal:  Giraffe
```



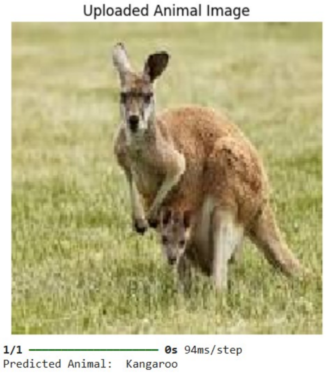
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 97ms/step
Predicted Animal:  Horse
```



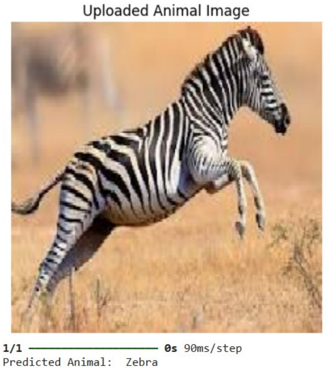
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 103ms/step
Predicted Animal:  Deer
```



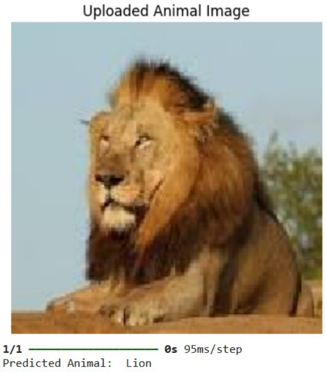
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 92ms/step
Predicted Animal:  Tiger
```



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 94ms/step
Predicted Animal:  Kangaroo
```



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 90ms/step
Predicted Animal:  Zebra
```



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 95ms/step
Predicted Animal:  Lion
```

➢ Insights & Observations

- Transfer learning significantly accelerated training and improved accuracy.
- Data augmentation played a crucial role in boosting model robustness.
- Misclassifications often occurred between visually similar species.
- The model showed resilience to variations in lighting and image background.
- Ensemble methods could further improve classification accuracy.
- This architecture follows the **transfer learning paradigm**:

  Pre-trained CNN → **feature extractor**

  GAP & Dense Layers → **compact representation and decision-making**

  Softmax Output → **final multi-class classification**

- It strikes a balance between **efficiency, generalization, and accuracy**, making it suitable for image classification tasks on moderately sized datasets.

➢ Future Scope

Potential enhancements to the current system include:

- Expanding the dataset with more diverse images.
- Incorporating ensemble deep learning architectures.
- Deploying the model as a cloud-based API for real-time animal recognition.
- Integrating object detection to identify multiple animals in a single image.
- Utilizing attention mechanisms to focus on key image regions.

➢ Conclusion

This project successfully demonstrated the use of deep learning and transfer learning for multi-class animal image classification. The resulting model achieved high accuracy, robust generalization, and practical applicability in various real-world domains. With further refinements, it can be deployed as a powerful tool for wildlife monitoring, conservation, and educational purposes.