# AI/ML INTERNSHIP

## COMPLETION

UNIFIED MENTOR
Nov 2025 - Feb 2026

4 Projects • 85-92% Accuracy • Swipe to see results →

# Offer Letter & Letter of Recommendation

UNIFIED MENTOR
YOUR SKILL. SUCCESS & JOURNEY

Address
DLF Forum, Cybercity, Phase III,
Gurugram, Haryana 122002

Unified Mentor Pvt. Ltd.
Tel:+91 6283 800330
www.unifiedmentor.com

**Date:** 05/11/2025
**UNID:** UMID03112569157

**Dear** Muthukuda Arachchige Kevin Senuja Dias

I'm pleased to offer you temporary employment as a Machine Learning Intern for a period of 3 months on behalf of Unified Mentor Pvt. Ltd. Starting from 05/11/2025 to 05/02/2026 . If you agree to this proposal, your internship with the company will start right away. You'll have "temporary employment" status while you're an intern. All of the perks that permanent employees of the company receive are not available to you as a temporary employee.

By accepting this offer, you acknowledge that you understand participation in this program is not an offer of employment and successful completion of the program does not entitle you to an employment offer from Unified Mentor.

This letter supersedes all past conversations and agreements about your internship and is the final agreement between you and the Company. Only a written amendment that is endorsed by both of us may change the terms of this letter. We look forward to having you begin your career at Unified Mentor and wish you a successful internship.

**Regards,**

Drishti Madaan

**Drishti Madaan**
HR Manager

**Phone**
+91 6283 800330

**Email**
info@unifiedmentor.com

Unified Mentor Pvt. Ltd.
Authorized Signatory

**Issued Date:** 05/02/2026

**CIN No- U85500HR2023PTC115118**

# CERTIFICATE
OF INTERNSHIP

UNIFIED MENTOR
YOUR SKILL. SUCCESS & JOURNEY

*Muthukuda Arachchige Kevin Senuja Dias*

***For successfully completing** three months **internship as** Machine Learning Intern **at Unified Mentor Pvt Ltd. Dated from** 05/11/2025 **to** 05/02/2026*
***During the internship we found him/her consistent & hard-working. We wish them all the best for their future endeavors.***

Verify at:

Paras Grover
**Paras Grover**
Director

ISO 9001:2015 CERTIFIED COMPANY

Sanket Patil
**Sanket Patil**
Awarded By

AN ISO **9001:2015** Certified Company

# 4 PROJECTS DELIVERED

### Mobile Price Classifier
88% Accuracy • Random Forest

### Animal Image Classifier
72% Accuracy • CNN + VGG16

### Heart Disease Detection
85% Accuracy • Classification

### Vehicle Price Predictor
• Linear Regression


# Mobile Price Classifier

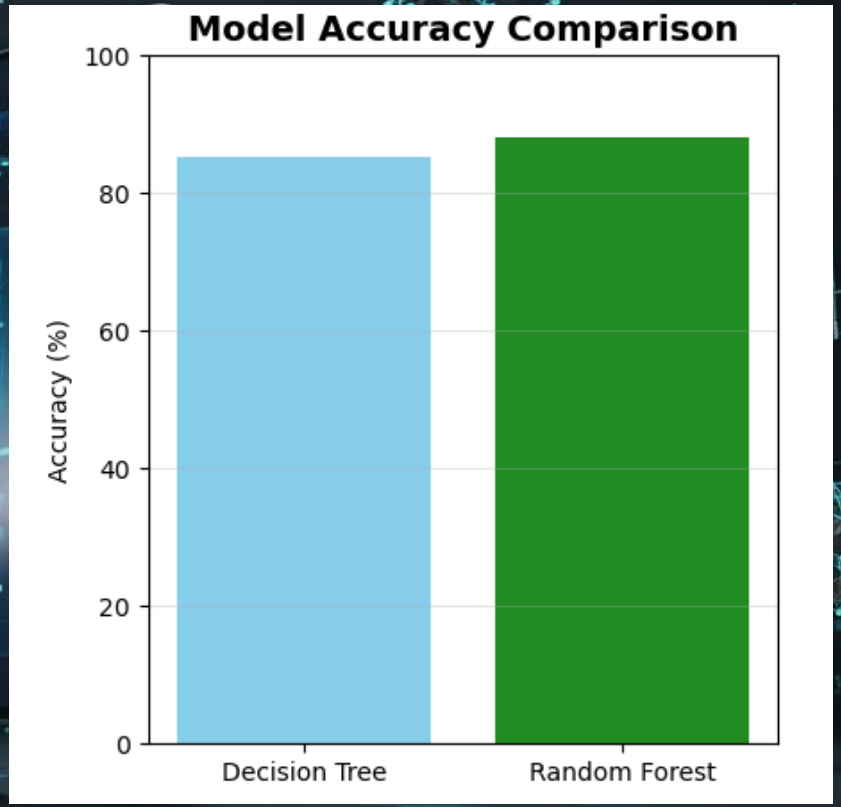
# Mobile Price - Model Comparison

```
# Detailed report
print("\n" + "="*60)
print("CLASSIFICATION REPORT - DECISION TREE")
print("="*60)
class_names = ['Low', 'Medium', 'High', 'Very High']
print(classification_report(y_test, y_pred_dt, target_names=class_names))
```

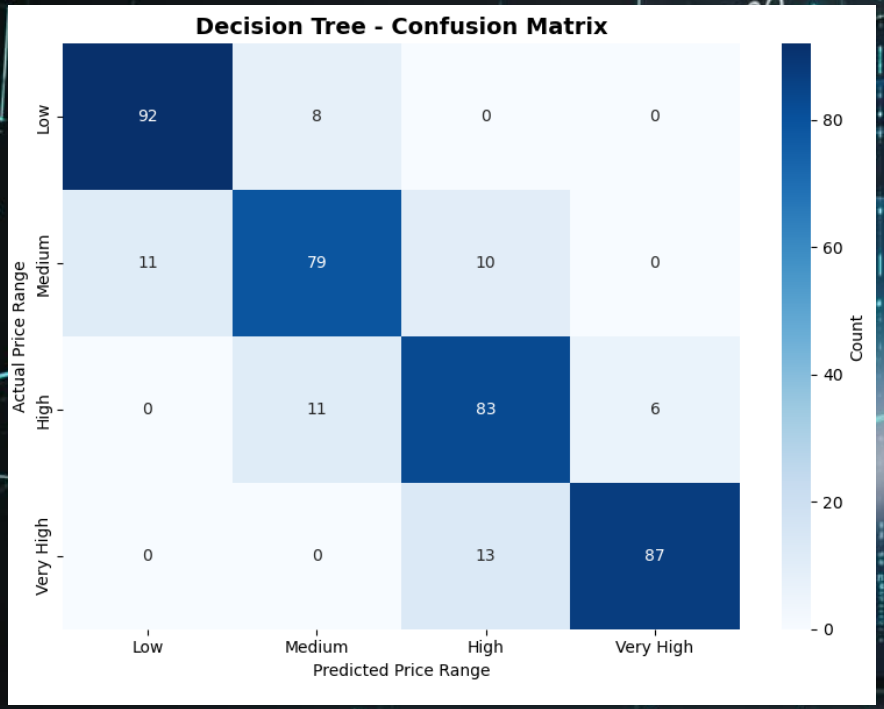
```
============================================================
CLASSIFICATION REPORT - DECISION TREE
============================================================
              precision    recall  f1-score   support

         Low       0.89      0.92      0.91       100
      Medium       0.81      0.79      0.80       100
        High       0.78      0.83      0.81       100
   Very High       0.94      0.87      0.90       100

    accuracy                           0.85       400
   macro avg       0.85      0.85      0.85       400
weighted avg       0.85      0.85      0.85       400
```

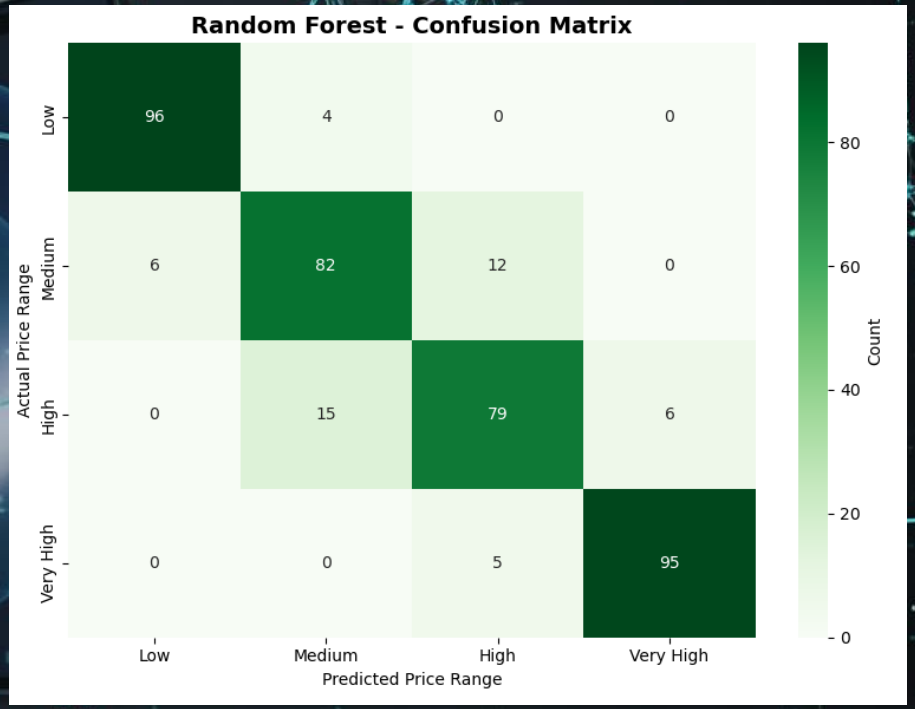
```
# Detailed report
print("\n" + "="*60)
print("CLASSIFICATION REPORT - RANDOM FOREST")
print("="*60)
print(classification_report(y_test, y_pred_rf, target_names=class_names))
```

```
============================================================
CLASSIFICATION REPORT - RANDOM FOREST
============================================================
              precision    recall  f1-score   support

         Low       0.94      0.96      0.95       100
      Medium       0.81      0.82      0.82       100
        High       0.82      0.79      0.81       100
   Very High       0.94      0.95      0.95       100

    accuracy                           0.88       400
   macro avg       0.88      0.88      0.88       400
weighted avg       0.88      0.88      0.88       400
```



# Mobile Price – Confusion Matrix



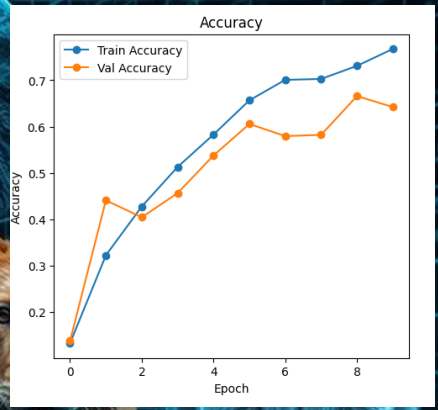
# 🐾 Animal Image Classifier

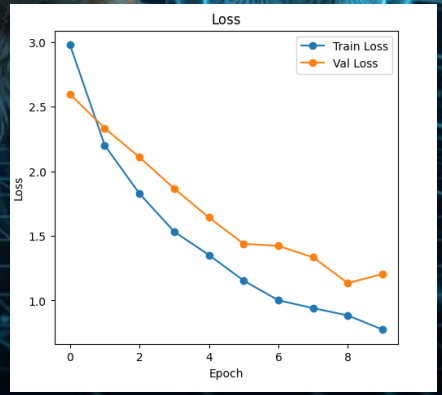
# Animal Classifier - Training History

[98]:
```
model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.BatchNormalization(),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.5),
    layers.BatchNormalization(),
    layers.Dense(256, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(NUM_CLASSES, activation='softmax')
])
```

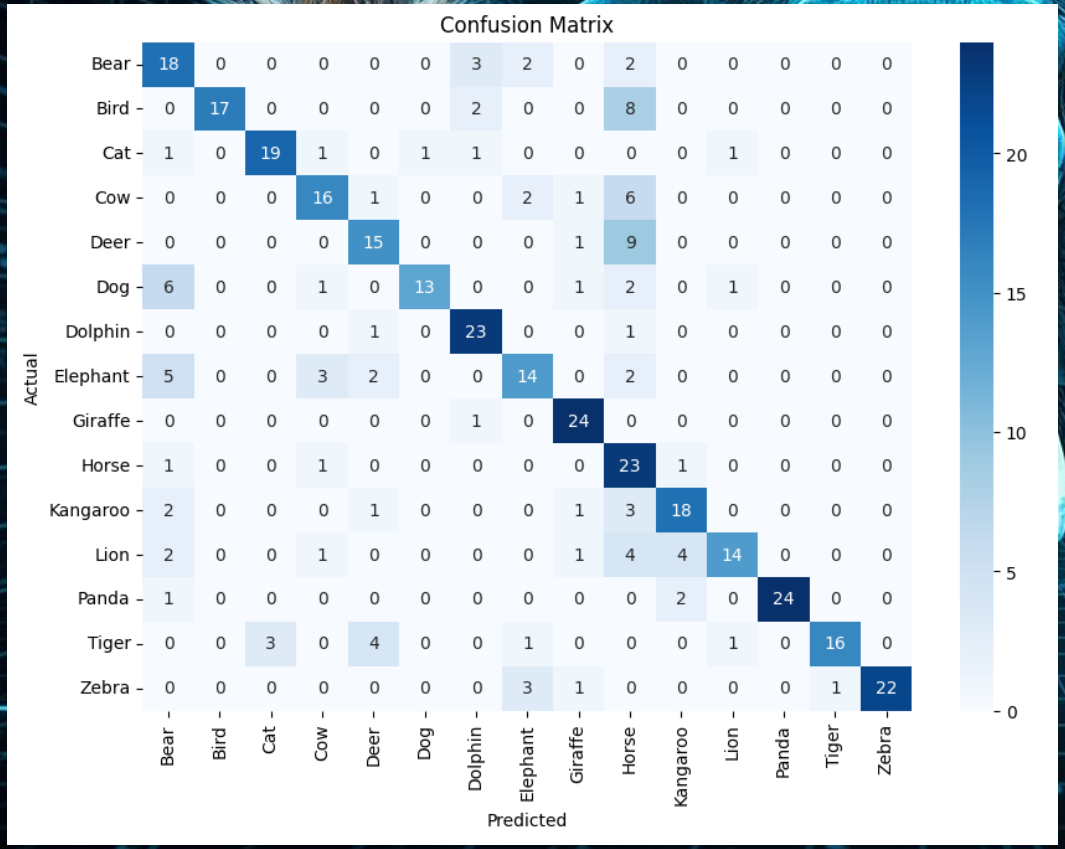
[105]:
```
history = model.fit(
    train_generator,
    validation_data=validation_generator,
    epochs=10
)
```

```
Epoch 1/10
49/49 [==============================] - 28s 563ms/step - loss: 0.7182 - accuracy: 0.7835 - val_loss: 1.0524 - val_accuracy: 0.6841
Epoch 2/10
49/49 [==============================] - 31s 628ms/step - loss: 0.6548 - accuracy: 0.7950 - val_loss: 1.0526 - val_accuracy: 0.6762
Epoch 3/10
49/49 [==============================] - 31s 637ms/step - loss: 0.6185 - accuracy: 0.8181 - val_loss: 1.0308 - val_accuracy: 0.6710
Epoch 4/10
49/49 [==============================] - 30s 606ms/step - loss: 0.5768 - accuracy: 0.8264 - val_loss: 1.1722 - val_accuracy: 0.6188
Epoch 5/10
49/49 [==============================] - 30s 604ms/step - loss: 0.5563 - accuracy: 0.8322 - val_loss: 1.1664 - val_accuracy: 0.6501
Epoch 6/10
49/49 [==============================] - 30s 606ms/step - loss: 0.5165 - accuracy: 0.8450 - val_loss: 1.0633 - val_accuracy: 0.6371
Epoch 7/10
49/49 [==============================] - 30s 606ms/step - loss: 0.4996 - accuracy: 0.8520 - val_loss: 0.9261 - val_accuracy: 0.7102
Epoch 8/10
49/49 [==============================] - 31s 624ms/step - loss: 0.5174 - accuracy: 0.8418 - val_loss: 0.9001 - val_accuracy: 0.7285
Epoch 9/10
49/49 [==============================] - 30s 620ms/step - loss: 0.4688 - accuracy: 0.8623 - val_loss: 0.9750 - val_accuracy: 0.6841
Epoch 10/10
49/49 [==============================] - 31s 624ms/step - loss: 0.4576 - accuracy: 0.8591 - val_loss: 0.9136 - val_accuracy: 0.7154
```



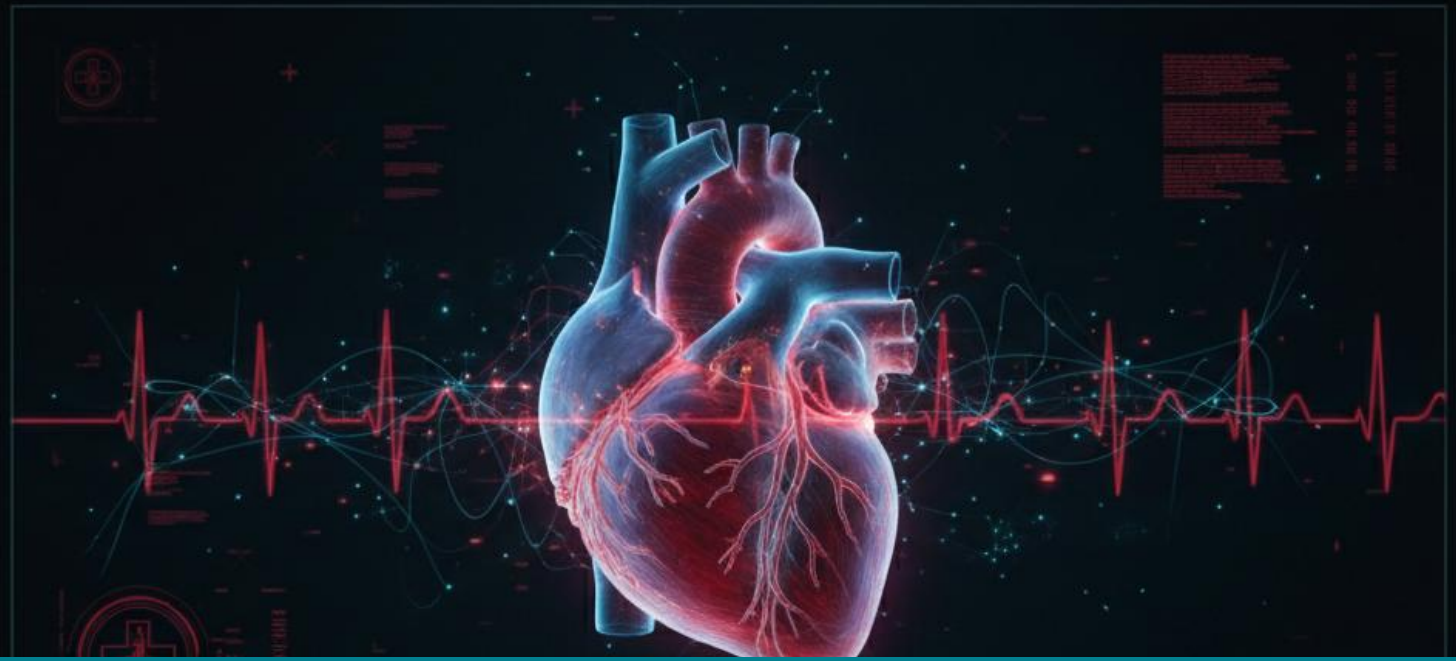



# Animal Classifier – Confusion Matrix



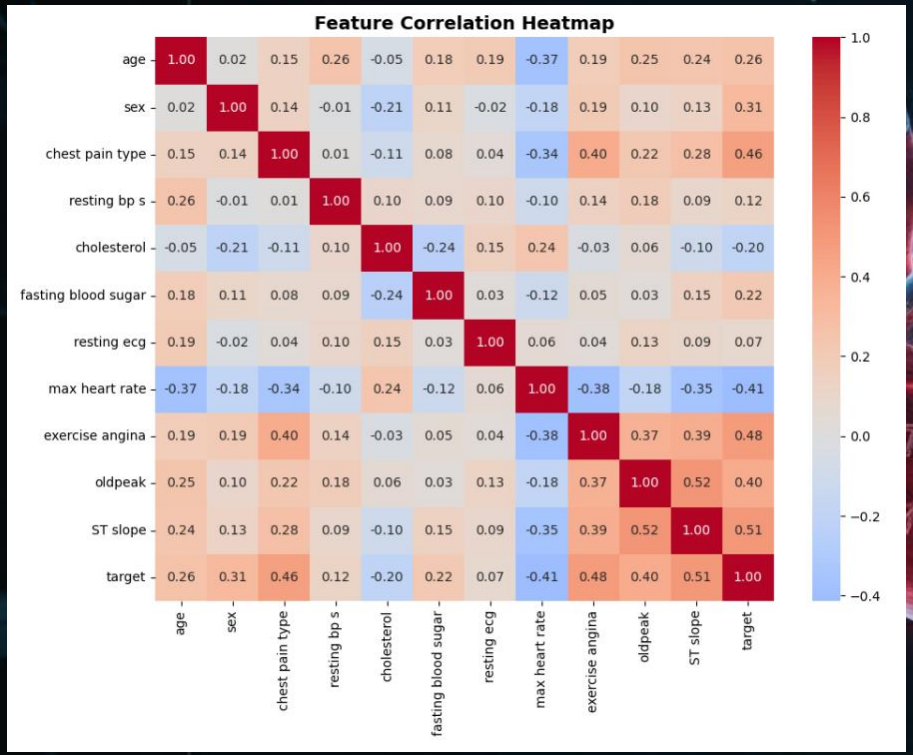

```
print(classification_report(y_true, y_pred, target_names=class_names))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Bear | 0.50 | 0.72 | 0.59 | 25 |
| Bird | 1.00 | 0.63 | 0.77 | 27 |
| Cat | 0.86 | 0.79 | 0.83 | 24 |
| Cow | 0.70 | 0.62 | 0.65 | 26 |
| Deer | 0.62 | 0.60 | 0.61 | 25 |
| Dog | 0.93 | 0.54 | 0.68 | 24 |
| Dolphin | 0.77 | 0.92 | 0.84 | 25 |
| Elephant | 0.64 | 0.54 | 0.58 | 26 |
| Giraffe | 0.80 | 0.96 | 0.87 | 25 |
| Horse | 0.38 | 0.88 | 0.53 | 26 |
| Kangaroo | 0.72 | 0.72 | 0.72 | 25 |
| Lion | 0.82 | 0.54 | 0.65 | 26 |
| Panda | 1.00 | 0.89 | 0.94 | 27 |
| Tiger | 0.94 | 0.64 | 0.76 | 25 |
| Zebra | 1.00 | 0.81 | 0.90 | 27 |
| accuracy | | | 0.72 | 383 |
| macro avg | 0.78 | 0.72 | 0.73 | 383 |
| weighted avg | 0.78 | 0.72 | 0.73 | 383 |



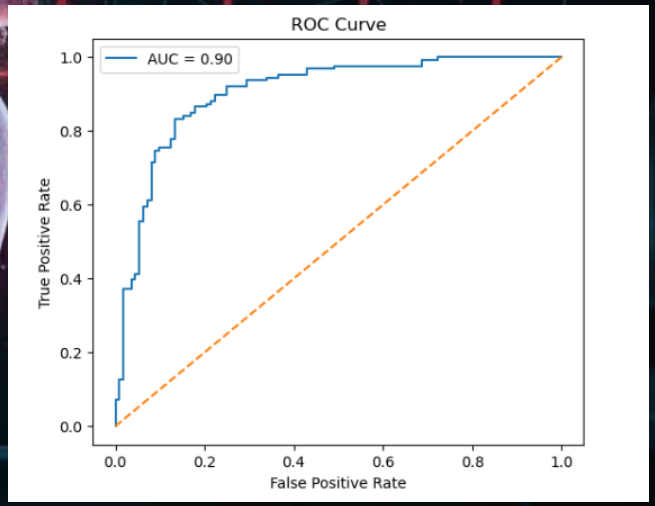
# Heart Disease

# Heart Disease – Correlation Heatmap & Predictions




```python
# Cell: Show sample predictions
print("Sample Predictions:")
for i in range(10):
    print(f"Actual: {y_test.iloc[i]}, Predicted: {y_test_pred[i]}, Probability: {y_test_prob[i]:.2f}")
```

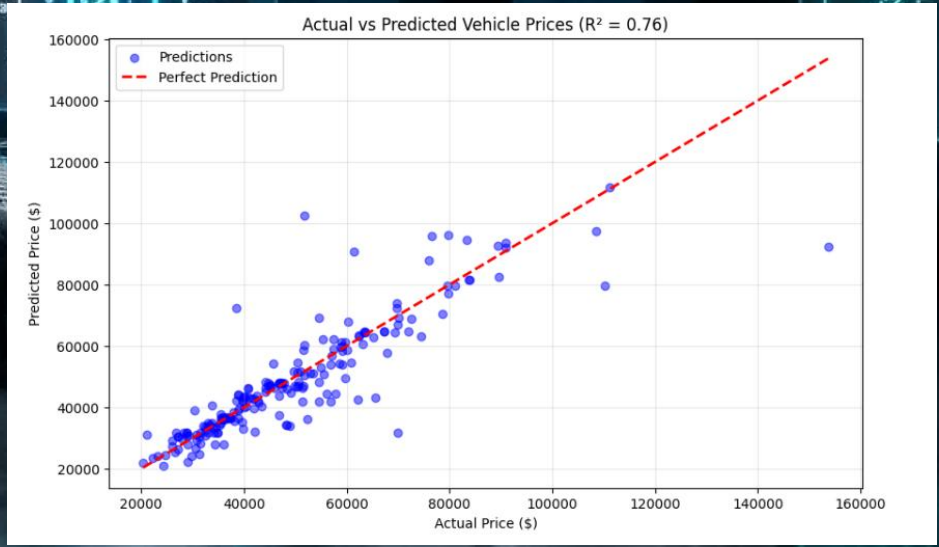
```
Sample Predictions:
Actual: 1, Predicted: 1, Probability: 0.96
Actual: 0, Predicted: 0, Probability: 0.45
Actual: 1, Predicted: 1, Probability: 0.94
Actual: 0, Predicted: 1, Probability: 0.53
Actual: 0, Predicted: 0, Probability: 0.09
Actual: 1, Predicted: 1, Probability: 0.89
Actual: 0, Predicted: 0, Probability: 0.02
Actual: 0, Predicted: 0, Probability: 0.22
Actual: 0, Predicted: 0, Probability: 0.01
Actual: 1, Predicted: 1, Probability: 0.90
```

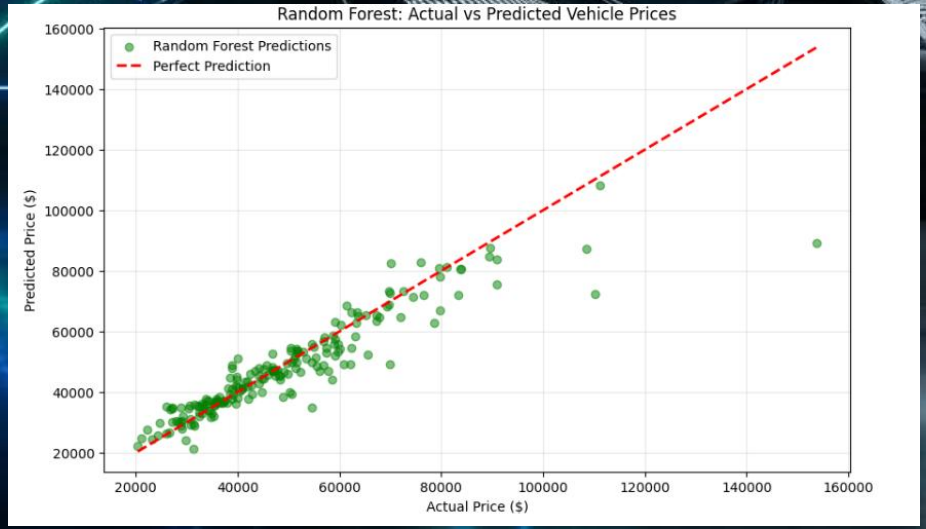



# Vehicle Price Prediction

# Vehicle Price - Best Fit Lines

## Linear Regression




## Random Forest classifier

# Vehicle Price - $R^2$ Score & Predictions

```
print(f"R²: {r2:.4f}  (higher is better, 0.6-0.8 is decent here)")
print(f"MAE:  ${mae:,.0f}   (average $ error)")
print(f"RMSE: ${rmse:,.0f}")
```

```
R²: 0.7598  (higher is better, 0.6-0.8 is decent here)
MAE:  $5,004   (average $ error)
RMSE: $9,317
```

Predictions

Linear Regression $R^2$: 0.7598

| | Actual | Predicted |
|---|---|---|
| 201 | 83940 | 81492 |
| 556 | 51803 | 60245 |
| 176 | 56105 | 44464 |
| 952 | 37335 | 36416 |
| 66 | 28860 | 31744 |
| 505 | 22260 | 23560 |
| 768 | 29111 | 27936 |
| 561 | 60080 | 58822 |
| 614 | 42150 | 31922 |
| 160 | 45038 | 47290 |

```
r2 = r2_score(y_actual, y_pred)
mae = mean_absolute_error(y_actual, y_pred)
rmse = np.sqrt(mean_squared_error(y_actual, y_pred))

print(f"R²: {r2:.4f}  (higher is better, 0.6-0.8 is decent here)")
print(f"MAE:  ${mae:,.0f}   (average $ error)")
print(f"RMSE: ${rmse:,.0f}")
```

```
R²: 0.8403  (higher is better, 0.6-0.8 is decent here)
MAE:  $4,147   (average $ error)
RMSE: $7,597
```

Predictions

Random Forest $R^2$: 0.8408

| | Actual | Predicted |
|---|---|---|
| 201 | 83940 | 80667 |
| 556 | 51803 | 53810 |
| 176 | 56105 | 47731 |
| 952 | 37335 | 37255 |
| 66 | 28860 | 28990 |
| 505 | 22260 | 27710 |
| 768 | 29111 | 30357 |
| 561 | 60080 | 54214 |
| 614 | 42150 | 37980 |
| 160 | 45038 | 47523 |

# TECH STACK

Python • TensorFlow • Keras • scikit-learn

Pandas • NumPy • Matplotlib • Seaborn

---

## Let's Connect!

#MachineLearning #AI #DataScience