Predicting Mobile Phone Prices

This presentation explores a dataset of smartphone models to predict their prices based on various features and specifications.

- **Student Name**
- **Nikhil Savaliya**
- **Institute Name**
- **Digicrome Academy**

Problem Statement

1

Understand Key Drivers

Identify the most important factors that influence smartphone prices.

2

Develop Prediction Model

Build a machine learning model to accurately forecast mobile phone prices.

3

Inform Business Decisions

Help companies make strategic strategic pricing and product product decisions.

Dataset Overview

Features

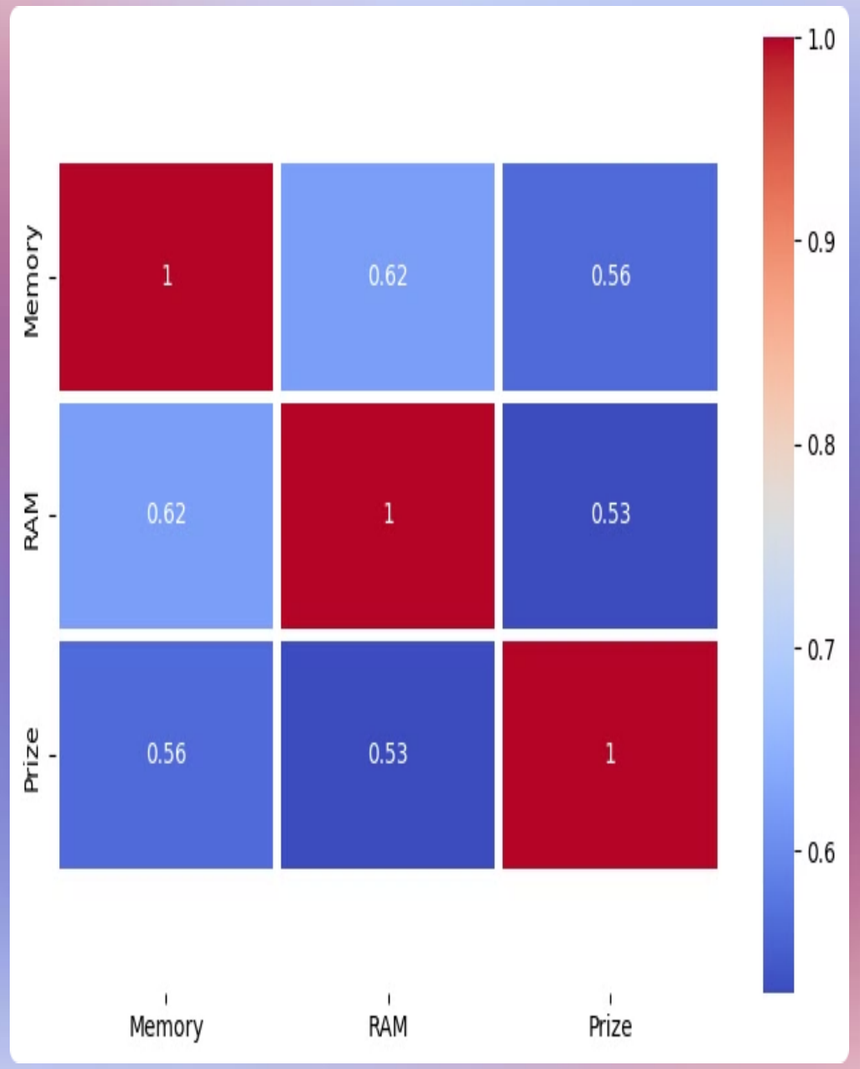
The dataset contains details on smartphone specs like display size, size, camera, processor, RAM, and and more.

Prices

The target variable is the actual market price for each smartphone model.

Diversity

The dataset covers a wide range of of smartphone brands, models, and and price points.



Feature Engineering

1

Feature Selection

The first step in feature engineering is to carefully select the most relevant variables from the dataset that are likely to impact smartphone pricing. This involves analyzing the data to identify the key technical specifications, hardware components, and other characteristics that are strong predictors of a phone's market value. We need to deeply understand the drivers of pricing in this industry to ensure our model focuses on the most important factors.

2

The image is a heatmap showing the correlation between three variables: Memory, RAM, and Prize.

Memory, RAM, and Prize. The correlation values are color-coded, with red with red indicating higher correlation and blue indicating lower. The The diagonal values are all 1, indicating perfect correlation with themselves. The off-diagonal values show the correlation between between different variables.

Memory and RAM: 0.62
RAM and Prize: 0.53
Memory and Prize: 0.56

These values suggest there is a strong positive correlation between these variables.

Data Transformation

```
data_encoded.head()
```

Unnamed: 0	Model	Colour	Memory	RAM	Battery_	Rear Camera	Front Camera	AI Lens	Mobile Height	Processor_	Prize
0	23	159	64.0	4	6000	3	10	1	16.76	113	7299.0
1	23	20	64.0	4	6000	3	10	1	16.76	113	7299.0
2	37	149	128.0	8	5000	10	4	0	16.64	75	11999.0
3	69	201	32.0	2	5000	13	10	0	16.56	56	5649.0
4	12	130	128.0	8	5000	10	10	1	16.76	14	8999.0

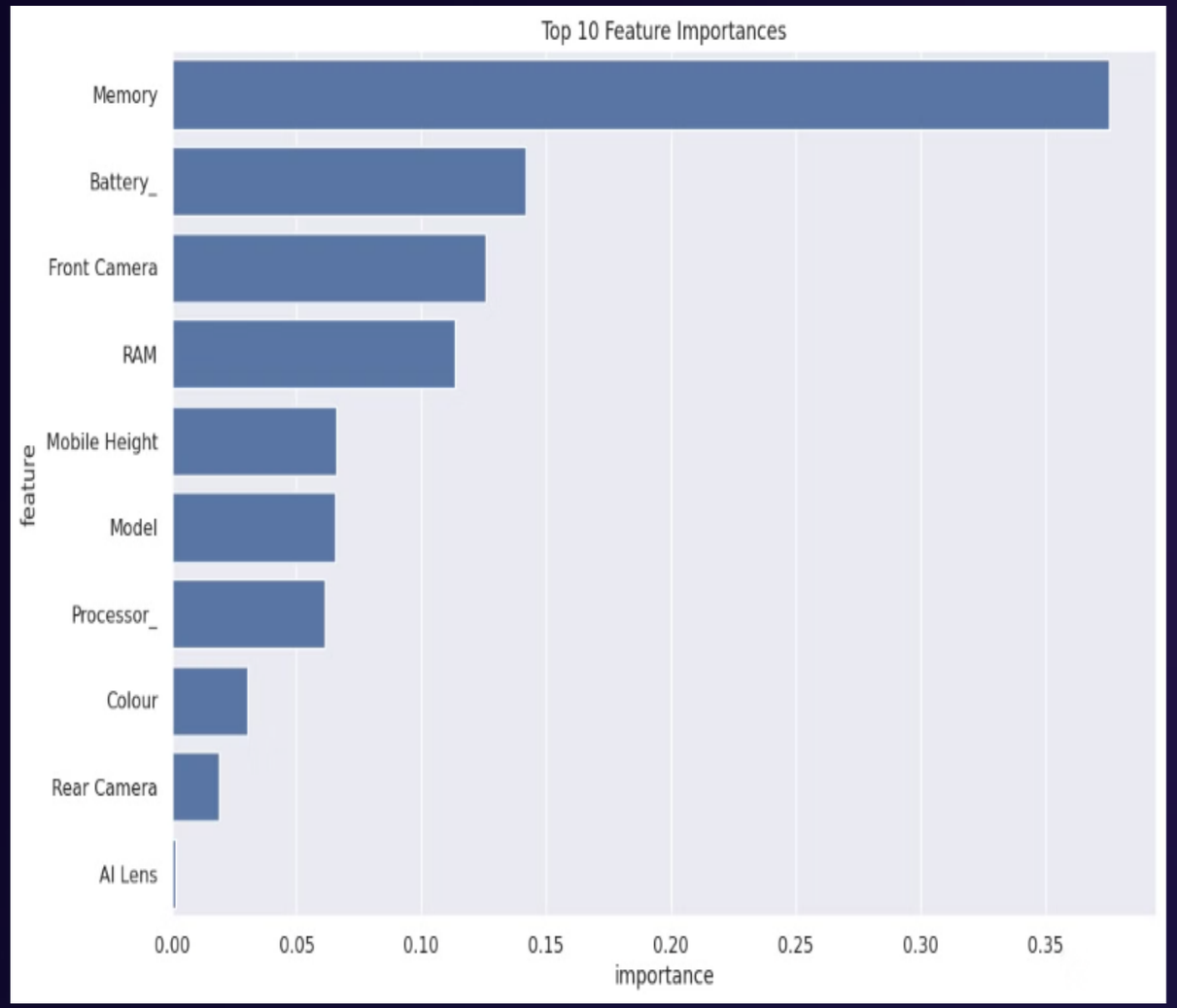
```
data_encoded.isnull().sum()
```

```
Model           0
Colour          0
Memory          0
RAM             0
Battery_        0
Rear Camera     0
Front Camera    0
AI Lens         0
Mobile Height   0
Processor_      0
Prize           0
dtype: int64
```

```
data_encoded.shape
```

Once we have identified the key features, the next step is to clean, format, and prepare the data for modeling. This may involve handling missing values, converting categorical variables to numerical, scaling features, and engineering new variables that capture important relationships in the data. The goal is to transform the raw data into a format that can be effectively utilized by our machine learning algorithms.

Feature Selection



The first step in feature engineering is to carefully select the most relevant variables from the dataset that are likely to impact smartphone pricing. This involves analyzing the data to identify the key technical specifications, hardware components, and other characteristics that are strong predictors of a phone's market value. We need to deeply understand the drivers of pricing in this industry to ensure our model focuses on the most important factors.

```python
# Import necessary library check accuracy in linear model.

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```python
# Train the Linear Regression model
lr = LinearRegression()
lr.fit(X_train, y_train)
```

```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```python
# Predict on the test set

y_pred_lr = lr.predict(X_test)
```

```python
# Evaluate the model
r2_score_lr = lr.score(X_test, y_test)
mae_lr = mean_absolute_error(y_test, y_pred_lr)
rmse_lr = mean_squared_error(y_test, y_pred_lr, squared=False)
```

```python
print(f"Linear Regression r2_score: {r2_score_lr}")
print(f"Linear Regression MAE: {mae_lr}")
print(f"Linear Regression RMSE: {rmse_lr}")
```

```
Linear Regression r2_score: 0.40313295434825924
Linear Regression MAE: 3928.883306222554
Linear Regression RMSE: 5747.195750307849
```

Model Selection

Linear Regression

A simple yet powerful model to predict continuous target variables.

Conclusion

- **Summary**:
    - The linear regression model provides a bad accuracy in predicting mobile phone prices, with an R2 score is only 40 %, MAE of 3928.88 and an RMSE of 5747.20. While it captures some relationships between features and prices, there is still significant variance unexplained, indicating room for improvement.

```
# Predict on the test set
y_pred_dt = dt.predict(X_test)
```

```
# Evaluate the model
r2_score_dt = dt.score(X_test, y_test)
mae_dt = mean_absolute_error(y_test, y_pred_dt)
rmse_dt = mean_squared_error(y_test, y_pred_dt, squared=False)
```

```
print(f"Decision Tree r2_score: {r2_score_dt}")
print(f"Decision Tree MAE: {mae_dt}")
print(f"Decision Tree RMSE: {rmse_dt}")
```

```
Decision Tree r2_score: 0.8994300545050623
Decision Tree MAE: 987.5514018691589
Decision Tree RMSE: 2359.1268653018187
```

Decision Trees

Captures non-linear relationships and feature interactions.

Conclusion

- **Summary**:
  - The Decision Tree model exhibits strong performance with an R^2 score of 89.94%, explaining a significant amount of variance in mobile phone prices. The lower error metrics (MAE: 987.55 and RMSE: 2359.13) highlight the model's accuracy and its capability to make precise predictions.
- **Comparison to Linear Regression**:
  - Compared to the linear regression model, the Decision Tree model shows superior performance, with a higher R^2 score and significantly lower MAE and RMSE values.

```
# Get the best parameters
best_params = grid_search.best_params_
best_model = grid_search.best_estimator_

# Print the best parameters and model
print("Best parameters found: ", best_params)
print("Best model found: ", best_model)
```

```
Best parameters found:  {'criterion': 'poisson', 'max_depth': None, 'max_featu
_split': 2}
Best model found:  DecisionTreeRegressor(criterion='poisson', random_state=4)
```

```
# Evaluate the best model on the test set
y_pred = best_model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mse)
```

```
print(f"Test MSE: {mse}")
print(f"Test R² Score: {r2}")
print(f"Test MAE: {mae}")
print(f"Test RMSE: {rmse}")
```

```
Test MSE: 6225901.033878504
Test R² Score: 0.887496033335037
Test MAE: 1055.6355140186915
Test RMSE: 2495.1755517154506
```

Model Evaluation: Hyperparameter-Tuned Tuned Decision Tree

Conclusion

- **Summary**:
  - The hyperparameter-tuned Decision Tree model exhibits strong performance with a Test R^2 score of of 88.75%, explaining a significant amount of variance in variance in mobile phone prices. The error metrics (Test metrics (Test MAE: 1055.64 and Test RMSE: 2495.18) 2495.18) highlight the model's continued accuracy and accuracy and its capability to make precise predictions, predictions, albeit with slightly higher errors compared compared to the untuned model.
- **Comparison to Untuned Decision Tree**:
  - The hyperparameter-tuned model shows a marginal decrease in R^2 score and slight increases in MAE and RMSE values. However, the model still maintains strong performance and accuracy.

```
# Cross-Validation
cv_scores = cross_val_score(rf, X_train, y_train, cv=5)
print(f'Cross-validation scores: {cv_scores}')
print(f'Mean cross-validation score: {np.mean(cv_scores)}')
```

```
Cross-validation scores: [0.85708329 0.87105515 0.89886844 0.90966448 0.9044339 ]
Mean cross-validation score: 0.8882210548631415
```

```
# Predict on the test set
y_pred_rf = rf.predict(X_test)
```

```
# Evaluate the model
r2_score_rf = rf.score(X_test, y_test)
mae_rf = mean_absolute_error(y_test, y_pred_rf)
rmse_rf = mean_squared_error(y_test, y_pred_rf, squared=False)
```

```
print(f"Random Forest r2_score: {r2_score_rf}")
print(f"Random Forest MAE: {mae_rf}")
print(f"Random Forest RMSE: {rmse_rf}")
```

```
Random Forest r2_score: 0.9152404667507072
Random Forest MAE: 1372.2701269470401
Random Forest RMSE: 2165.7656790937194
```

Random Forest Regressor

An ensemble method that improves accuracy and reduces overfitting.

Conclusion

- **Summary**:
    - The Random Forest model exhibits excellent performance with a high R^2 score of 91.52%, explaining a substantial amount of variance in mobile phone prices. The error metrics (MAE: 1372.27 and RMSE: 2165.77) highlight the model's accuracy and its capability to make precise predictions.
- **Comparison to Other Models**:
    - Compared to the Decision Tree and Linear Regression models, the Random Forest model shows superior performance, with a higher R^2 score and competitive error values. The model's robustness is further demonstrated by the consistent cross-validation scores.

```
best_rf = RandomForestRegressor(
    bootstrap=False,
    max_depth=20,
    min_samples_leaf=1,
    min_samples_split=2,
    n_estimators=200,
    random_state=42
)

# Train the model on the training data
best_rf.fit(X_train, y_train)
```

```
RandomForestRegressor(bootstrap=False, max_depth=20, n_estimators=200,
                      random_state=42)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
# Predict on the test set
y_pred_best_rf = rf.predict(X_test)
```

```
r2_score_best_rf = best_rf.score(X_test, y_test)
mae_best_rf = mean_absolute_error(y_test, y_pred_best_rf)
rmse_best_rf = mean_squared_error(y_test, y_pred_best_rf, squared=False)
```

```
print(f"Random Forest Hyperparameter tunning r2_score: {r2_score_best_rf}")
print(f"Random Forest Hyperparameter tunning MAE: {mae_best_rf}")
print(f"Random Forest Hyperparameter tunning RMSE: {rmse_best_rf}")
```

```
Random Forest Hyperparameter tunning r2_score: 0.9043526741083743
Random Forest Hyperparameter tunning MAE: 1372.2701269470401
Random Forest Hyperparameter tunning RMSE: 2165.7656790937194
```

Model Evaluation: Hyperparameter-Tuned Tuned Random Forest

Conclusion

- **Summary**:
    - The hyperparameter-tuned Random Forest model model exhibits excellent performance with a high R^2 high R^2 score of 0.9044, explaining a substantial substantial amount of variance in mobile phone phone prices. The error metrics (MAE: 1372.27 and 1372.27 and RMSE: 2165.77) highlight the model's model's accuracy and its capability to make precise precise predictions.
- **Comparison to Untuned Random Forest**:
    - The hyperparameter-tuned model maintains a high level of performance, with an R^2 score slightly lower than the untuned model (0.9152), but still demonstrating strong predictive power. The MAE and RMSE values remain consistent, indicating stable error magnitudes.

Visualizing the Results Actual Vs Predicted

Random Forest Data Data visualization

Decision tree data visualization

Linear Regression data visualization

Analysis:

Among the three models, the Random Forest model has the highest R2 score (91.52%), indicating that it explains the variance in the data the best. Both the Decision Tree and Random Forest models have significantly higher R2 scores compared to the Linear Regression model, suggesting they fit the data better in terms of explaining its variability.

In terms of the mean absolute error (MAE) and root mean squared error (RMSE), lower values indicate better performance. Here, the Decision Tree model has the lowest MAE (987.55) and RMSE (2359.13), which means it has the least average error and deviation from the actual values among the three models.

Conclusion

Best Fit Model: Random Forest

The higher R^2 score and lower RMSE indicate that the Random Forest model generally generally performs better in terms of explaining variance and overall prediction accuracy, accuracy, even though its MAE is higher.


thank you