**Predictive Analysis of Smartphone Selling Prices using Linear Regression**

**Introduction**

This report provides an analysis prediction based off the factors that influence price of smartphones, such as brand, color, memory, storage, and rating. The purpose of this report is to illustrate those factors that influence the prices of smartphones while also providing future price estimates.

Kaggle was the source of this data set. It is a structured non binary dataset consisting of data about sales of different brands of smart phones over a given period of time. Below is the description for the columns of the dataset:

1. Brands – Describes the various brands of smartphones in the dataset e.g., Samsung, Apple, HTC, Infinix, Google pixel.
2. Colors – Representing the smart phone color e.g., black, gold.
3. Memory – The smartphone storage capacity, which is measured either in gigabytes (GB) or megabytes (MB).
4. Storage – Indicates the internal storage space of a smartphone, which is expressed in Gigabytes (GB) or Megabytes (MB).
5. Rating – User rating scores assigned to the smart phones, reflecting user satisfaction or performance.
6. Selling price – Price at which smart phones are sold to the customers.
7. Original price – Original or list price of the smartphone before any discounts or promotions.
8. Mobile - Shows whether the device is a mobile phone.
9. Discount – Relates to how much the discount is applied to the original price to obtain the selling price.
10. Discount percentage: Percentage discount taken from the original price to find the selling price.

The dataset allows us to do various analysis like customer segmentation, popular phone brands classification and clustering smart phones common brands or brands of phones being used. We will use this dataset to predict how several factors influence future prices of smartphones. We will use linear regression to analyze the dataset. Linear regression was chosen for its ability to show the independent variables that affect the continuous dependent variable. This method is used to analyze the relationship between two variables by computing how the independent variables (brands, colors, memory, and rating) impact the value of the dependent variable (selling price). Linear regression will enable us to examine the following about the dataset:

1. **Relationships among the variables** - In the case of linear regression, we'll find out if the independent variables (brands, colors, RAM, storage) and the dependent variable (selling price) are in some relations. Having a look at the regression equation coefficients, we know if there is a positive or negative impact on the dependent variable and the magnitude of that influence from the independent variables. For instance, if the coefficient of a given feature e.g., rating is positive, it shows that there is a positive relationship between the increase of that feature and the target variable (Mali, 2024). On the contrary, a negative coefficient indicates that a rise in the feature leads to a drop in the target variable.
2. **Measuring the strength of association** - The R-squared indicator assists us in determining linear regression ‘s strength of association among the independent and dependent variables. R-squared indicates the amount of dependent variable variation that is attributed to the independent variables (Marcelino, 2024). A higher R-squared value means a better correlation between the variables, which implies that the explanatory variables explain the value of the dependent variable quite well. On the other hand, a lower R-squared value demonstrates a weak relationship, suggesting that the independent variables might not be good predictors of the dependent variable.
3. **Identification of Outliers and meaningful observations** - Correlation and linear regression analysis can be used in the search of outliers and influential observations in the dataset. Outliers are data points that are significantly separated from rest data and may have a large influence on the descriptive power of the regression model. Through analyzing the differences between the actual & predicted values we can uncover the outliers & observations that have a greater influence hence need further investigation (Mali, 2024). Properly handling outliers and influential observations not only helps to increase the precision but also makes the model healthier.
4. **Prediction** - The most important role of linear regression is prediction. Through the coefficients acquired from regression, one could set up a predictive model for the dependent variable in terms of the independent variables (Marcelino, 2022). These provide an opportunity to project results for future, or to measure the variable of interest given observed outputs. The accuracy of these predictions can be evaluated using metrics like mean square error (MSE) that specifies the average squared difference between the expected and actual values. A lower MSE suggests better predictive performance and hence it means the model is doing a good job to predict the dependent variable based on independent variables.

**Methodology**

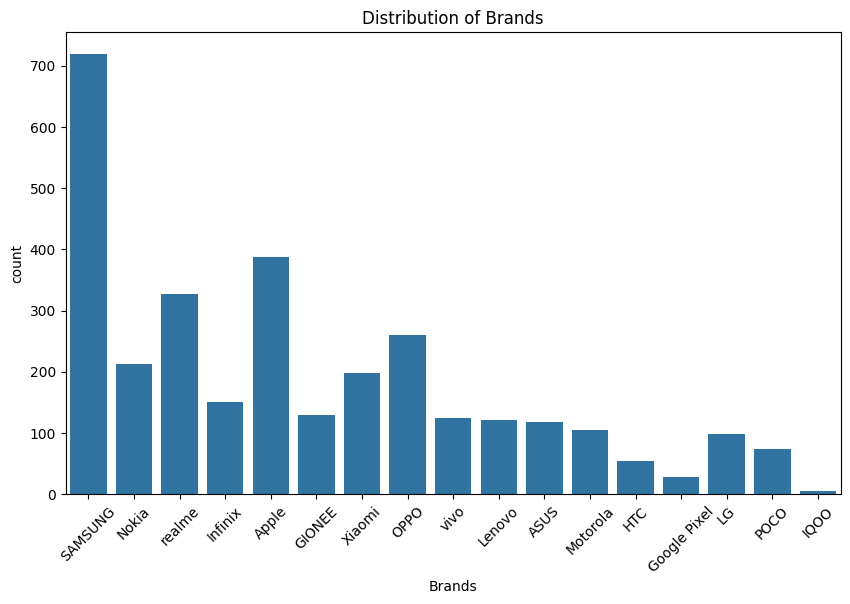
1. Data collection and Preprocessing

The dataset used featured various smartphone brands, colors and specifications such as memory, original prices, discounts, rating scores and selling prices.

The following data was analyzed:

1. Count of Smart phones brands

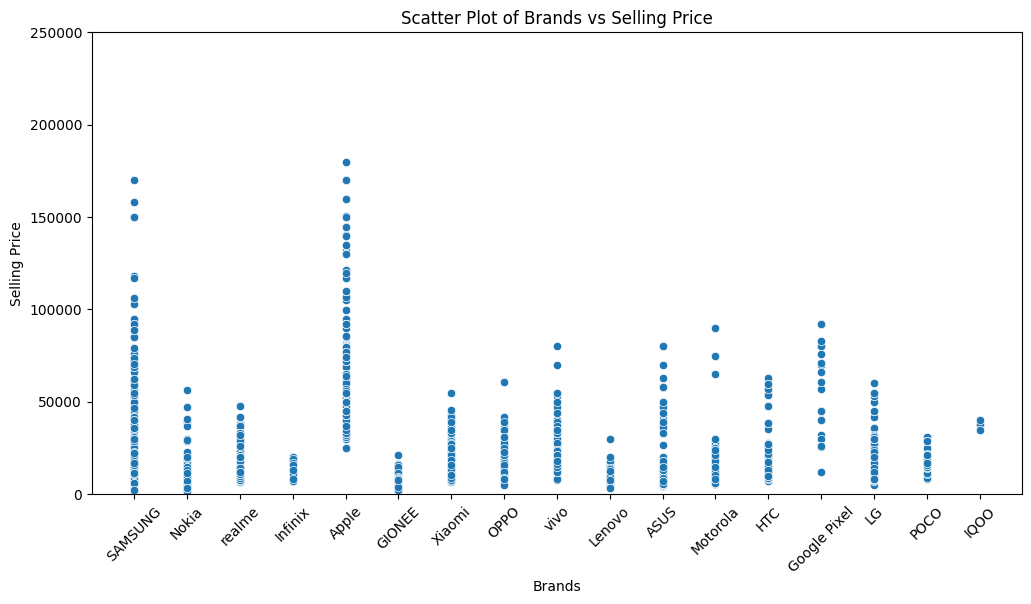
Samsung brand smart phones had the highest count, with over 700 phones. They were followed by Apple smart phones, RealMe, OPPO, Nokia and Xiaomi. Google pixel and IQOO had the lowest count of smart phones.



*Fig 1: Distribution of brands and their counts*

1. Distribution of Brands against their Selling price

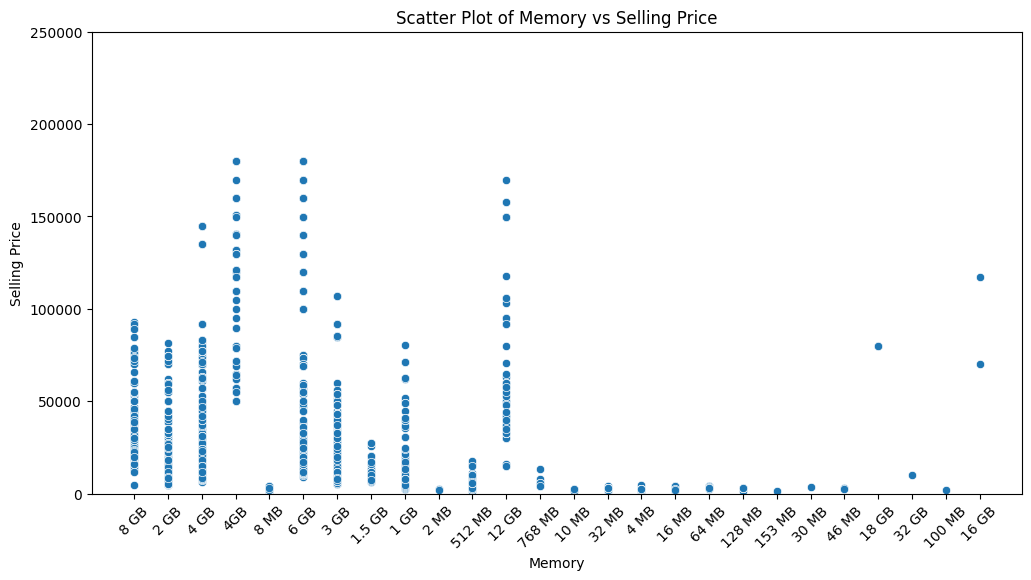
The smart phones with the highest selling prices were Apple, closely followed by Samsung. Infinix smart phones had the lowest selling prices.



*Fig 2: A scatter plot showing the brands of smartphones against their selling prices.*

1. Distribution of Memory against their selling prices

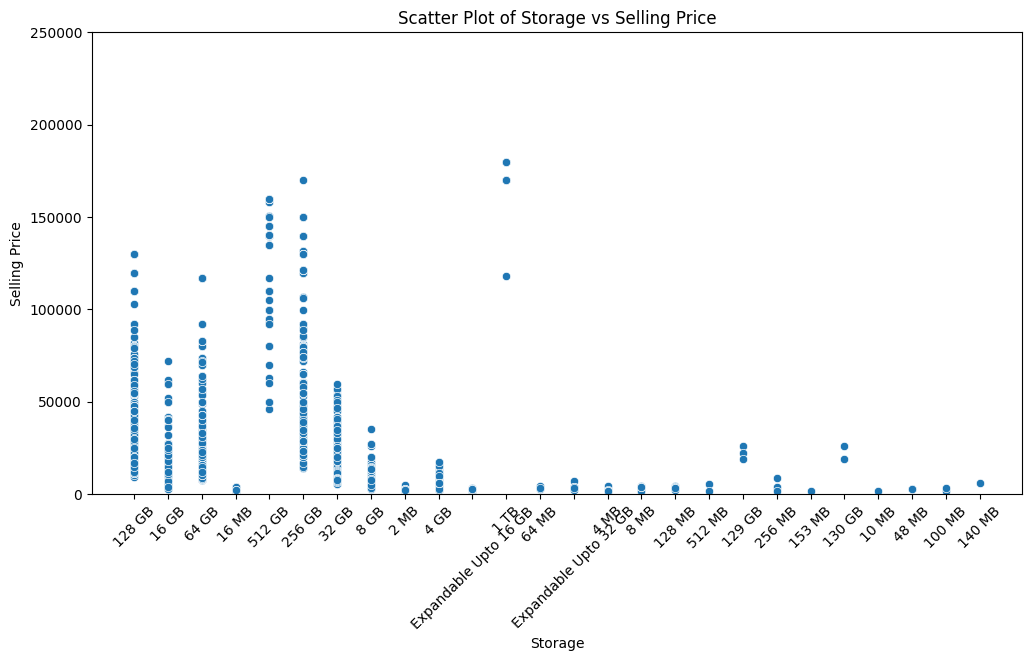
Smart phones with a memory of 4 GB and 6 GB had the highest selling price. They were closely followed by those with 12GB memory. Thise with the lowest selling price had memories of 10MB, 32MB, 4MB, 16MB, 64MB. This indicates that the selling price of the smart phones was directly related to memory space.



*Fig 3: A scatter plot showing the selling prices of smartphones based on their memory sizes.*

1. Distribution of storage of smartphones

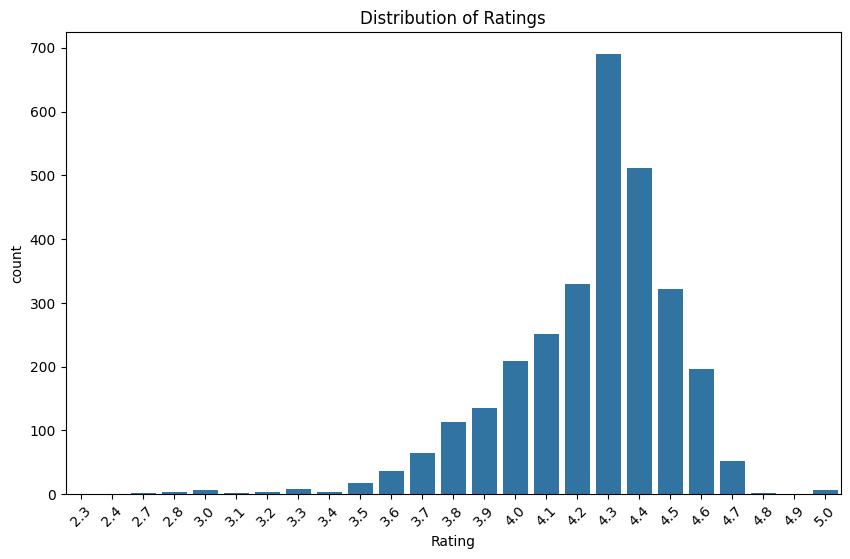
Smart phones with a storage of 1TB, 512GB and 256GB had the highest selling price. They were followed 128GB, 64GB, 32GB and 16GB. Smart phones whose storages were in Megabytes (MBs) had the lowest selling prices.



*Fig 4: A scatter plot illustrating the selling prices of smart phones based on their storage*

1. Distribution of ratings

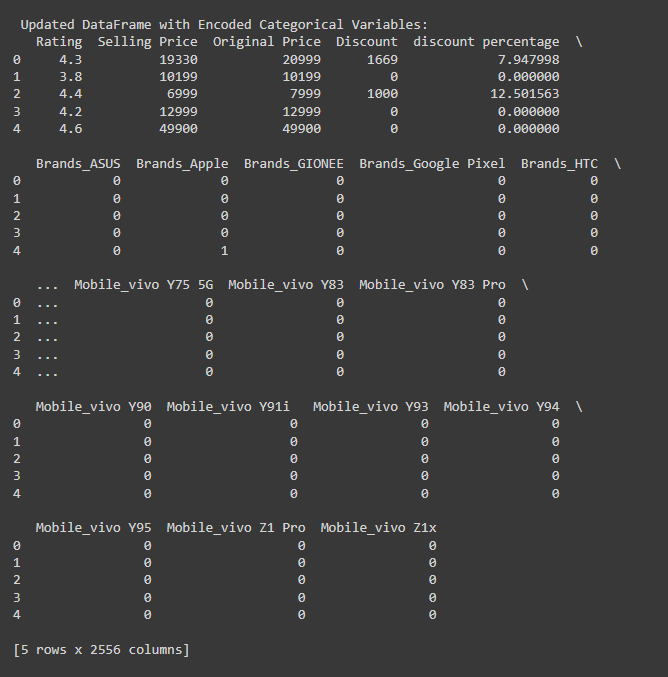
Most smartphones had a rating of 4.3, closely followed by 4.4, 4.2 and 4.5. Phones with a rating of 2.3 to 3.5 had the lowest count as shown in the distribution below.



*Fig 5: A histogram showcasing the distribution of ratings of smartphones*

1. Feature Engineering

Dummy variables were created for categorical features and numerical features were scaled where necessary. These were created using one-hot encoding for color and brands columns. This was important to help us include categorical data in our regression model. We used scaling techniques such as normalization and standardization to ensure all numerical features were comparable. This helps to prevent features with higher magnitudes from dominating the model when training.

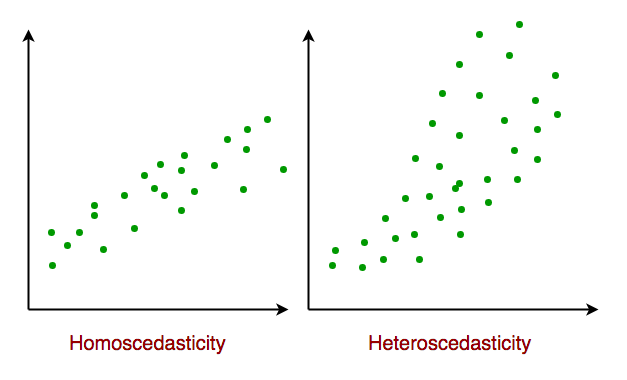


1. Model Selection

We selected and used a Linear regression algorithm for modelling. Linear regression is a statistical method that allows us to foretell a continuous dependent variable by the use of one or more independent variables. It used for predicting how one or more independent variables are related to the value of a dependent variable.  
In the case of our dataset selection, the dependent variable was the “Selling Price” of phones with which, the independent variables were the “Brand”, “Model”, “Color”, “Rating”, “Storage” and “Memory”.

**Assumptions made in Linear regression**

* **Linear relationship** – The relationship between the dependent variable and feature variables should be linear. This linearity can be tested using scatter plots.
* There is minor or no multi-collinearity – It is assumed that feature variables (independent variables) are not independent of each other (Geeks for Geeks, 2023).  
  It is also considered that there is little autocorrelation that takes place in situations where the residual errors are not independent.
* No outliers — We suppose that there are no outliers in the datasets Outliers are data points far away from the rest of the data and can affect the results of the analysis (Geeks for Geeks, 2023).
* **Homoscedasticity** – It describes a situation where the error term (disturbance in relationship between dependent and independent variables) is the same across all values of the independent variables (Geeks for Geeks, 2023).



*Fig 6: Homoscedasticity in Linear regression*

The formula for linear regression is as follows:

Y = **β0 + β1x1 + β2x2 + β3x3 … + ∈ (Mali, 2024)**

Where:

* **Y –** The dependent variable (selling price)
* **β0 –** The value of y when all other parameters are zero
* **β1x1 –** the first independent variable regression coefficient
* **β2x2 –** The second independent variable regression coefficient
* **∈ - The residual error**

1. **Procedure followed**

For this project, we worked with an ipynb notebook with python due to its speed and efficiency in executing fast results. We followed the following steps:

1. Importing the dataset – we uploaded the dataset to Google collab in CSV format and created the code for reading the CSV file and displaying basic information about the dataset.
2. Checking for missing values – We then checked and handled missing values and filled them with either mean or mode for numerical and categorical data respectively.
3. Next, we performed descriptive analysis for numerical columns in order to describe the numerical and categorical columns.
4. The next step involved exploration of distributions and relationships where we plotted a pair plot and a heatmap to showcase the relationships between the variables.
5. We then performed one-hot encoding for the colors and brands columns, which enabled them to be incorporated into the model.
6. Regression analysis was performed next on the data which involved splitting it into training and testing sets then performing the regression on the split data.
7. Finally, we calculated the R-squared value and the Mean squared error (MSE) to derive insights on the dataset.
8. Implementation

The following is the code generated to perform the regression analysis:

#Importing necessary libraries

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

from sklearn.preprocessing import StandardScaler

from scipy.stats import zscore

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

from sklearn.linear\_model import LinearRegression

from sklearn.preprocessing import MinMaxScaler

from sklearn.impute import SimpleImputer

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, median\_absolute\_error, r2\_score

# loading the dataset

df = pd.read\_csv("Sales.csv")

# Display basic information about the dataset

print(df.info())

# Display summary statistics for numerical variables

print(df.describe())

# Histogram for numerical variables

df.hist(figsize = (10, 8))

plt.tight\_layout()

plt.show()

# Set the figure size to make room for longer x-axis labels

plt.figure(figsize=(12, 6))

# Create the scatter plot

sns.scatterplot(x='Brands', y='Selling Price', data = df)

# Rotate the x-axis labels for better readability

plt.xticks(rotation=45)

# Set the y-axis limit to ensure all data points are visible

plt.ylim(0, 250000)

# Add title and labels

plt.title('Scatter Plot of Brands vs Selling Price')

plt.xlabel('Brands')

plt.ylabel('Selling Price')

# Show the plot

plt.show()

# Set the figure size to make room for longer x-axis labels

plt.figure(figsize=(12, 6))

# Create the scatter plot

sns.scatterplot(x='Memory', y='Selling Price', data = df)

# Rotate the x-axis labels for better readability

plt.xticks(rotation=45)

# Set the y-axis limit to ensure all data points are visible

plt.ylim(0, 250000)

# Add title and labels

plt.title('Scatter Plot of Memory vs Selling Price')

plt.xlabel('Memory')

plt.ylabel('Selling Price')

# Show the plot

plt.show()

# Set the figure size to make room for longer x-axis labels

plt.figure(figsize=(12, 6))

# Create the scatter plot

sns.scatterplot(x='Storage', y='Selling Price', data = df)

# Rotate the x-axis labels for better readability

plt.xticks(rotation=45)

# Set the y-axis limit to ensure all data points are visible

plt.ylim(0, 250000)

# Add title and labels

plt.title('Scatter Plot of Storage vs Selling Price')

plt.xlabel('Storage')

plt.ylabel('Selling Price')

# Show the plot

plt.show()

# Count plot for 'Brands'

plt.figure(figsize=(10, 6))

sns.countplot(x='Brands', data=df)

plt.xticks(rotation=45)

plt.title('Distribution of Brands')

plt.show()

# Count plot for 'Brands'

plt.figure(figsize=(10, 6))

sns.countplot(x='Rating', data=df)

plt.xticks(rotation=45)

plt.title('Distribution of Ratings')

plt.show()

# Step 2: Check for missing values

missing\_values = df.isnull().sum()

print("Missing Values:")

print(missing\_values)

# Handle missing values

# For numerical columns, fill missing values with mean

numerical\_cols = df.select\_dtypes(include=[np.number]).columns

df[numerical\_cols] = df[numerical\_cols].fillna(df[numerical\_cols].mean())

# For categorical columns, fill missing values with mode

categorical\_cols = df.select\_dtypes(include=[object]).columns

df[categorical\_cols] = df[categorical\_cols].fillna(df[categorical\_cols].mode().iloc[0])

# Step 3: Descriptive statistics for numerical columns

print("\n Descriptive Statistics for Numerical Columns:")

print(df.describe())

# Step 4: Explore distributions and relationships

# Pairplot for numerical columns

sns.pairplot(df[numerical\_cols])

plt.show()

# Correlation heatmap

correlation\_matrix = df.corr()

plt.figure(figsize=(10, 8))

sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt=".2f")

plt.title("Correlation Heatmap")

plt.show()

# Step 5: Encode categorical variables

# Use one-hot encoding for categorical variables

df = pd.get\_dummies(df, columns=categorical\_cols)

# Display updated dataframe

print("\n Updated DataFrame with Encoded Categorical Variables:")

print(df.head())

#Regression Analysis

# Step 1: Split data into features (X) and target variable (y)

X = df.drop(columns=['Selling Price'])

y = df['Selling Price']

# Step 2: Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=42)

# Step 3: Initialize and train the regression model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Step 4: Make predictions on the testing set

y\_pred = model.predict(X\_test)

# Step 5: Evaluate the model

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error:", mse)

print("R-squared:", r2)

# Interpret the coefficients

coefficients = pd.DataFrame({'Feature': X.columns, 'Coefficient': model.coef\_})

print("\n Coefficients:")

print(coefficients)

# Visualize model predictions vs. actual selling prices

plt.figure(figsize=(8, 6))

plt.scatter(y\_test, y\_pred)

plt.plot([min(y\_test), max(y\_test)], [min(y\_test), max(y\_test)], 'k--', lw=2)

plt.xlabel("Actual Selling Prices")

plt.ylabel("Predicted Selling Prices")

plt.title("Model Predictions vs. Actual Selling Prices (Linear Regression)")

plt.show()

**Results**

**Performance Analysis**

* Mean Squared Error: 9.391731717398328e-20
* R-squared: 1.0

The mean squared error (MSE) is extremely low, which indicates a very close fit between the actual and predicted selling prices. This suggests that the model performs well on the test data. The R-squared value of 1.0 indicates that the model explains 100% of the variance in the target variable (selling price). This is an excellent performance of the model.

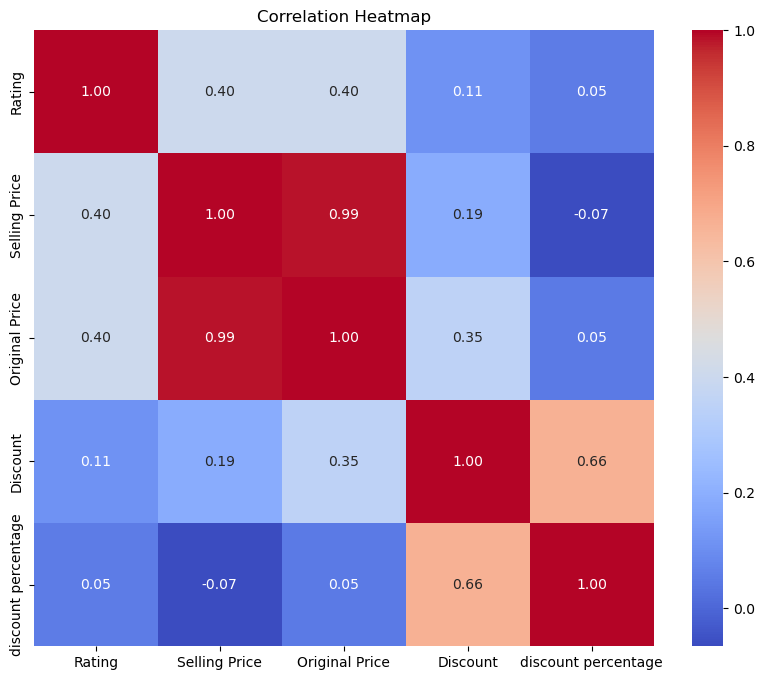
**Coefficients Analysis**

 Coefficients:

* **Rating:** 5.142820e-11
* **Original Price:** 1.000000e+00
* **Discount:** -1.000000e+00
* **Discount Percentage:** 1.403872e-12

Coefficients represent the impact of each feature on the selling price. The coefficient for the “original price” is 1.0, which suggests that for every unit increase in the original price, the predicted selling price increases by the same amount. This signifies a direct linear relationship between the original and selling price. The coefficient for discount is -1.0, which suggests that for every unit increase in the discount, the predicted selling price decreases by that amount. This signifies an inverse linear relationship between the discount and the selling price.

**Visualizing model predictions**

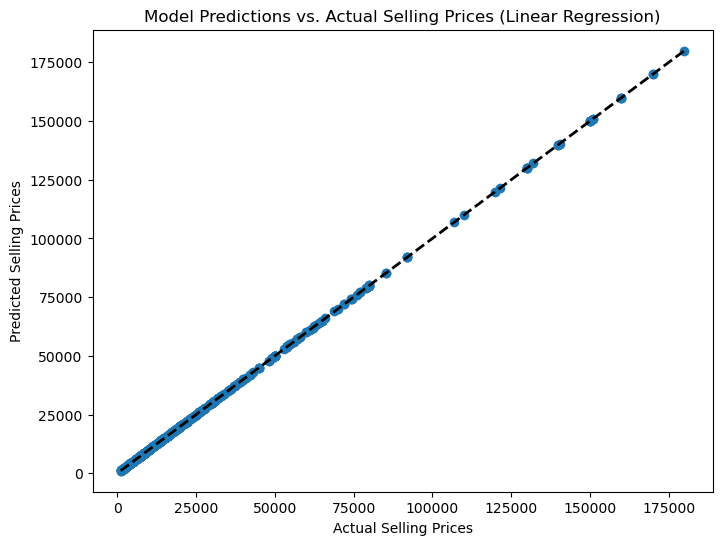
****

*Fig 5: A correlation matrix showing the relationships between model variables*

The correlation matrix highlights the correlation coefficients between various variables. Each column and row highlights a different variable, with the intersection carrying the correlation coefficient among the variables represented by that particular row and column (Bock, 2024). The strength and direction of a linear relationship between two variables are measured by correlation coefficients. There are three major reasons for computing a correlation matrix:

1. To summarize a large amount of data to detect patterns.
2. To use the output as input in other analyses.
3. To use as a diagnostic when checking other analyses. For instance, with linear regression, the high number of correlations suggests that linear regression estimates will be unreliable (Bock, 2024).

The correlation matrix shows the relationship between the various columns of the dataset. The selling price and original price are closely related with a value of 0.99. This indicates that most phones were sold at a price close to their original price. This can further be confirmed by the relationship between the selling price and discount, which is 0.19, and the discount percentage and the selling price, -0.07. They are low indicating a very small discount offered.

****

***Fig 6: Scatter plot showing the linear regression predicted of predicted selling price against actual selling price***

The scatter plot above depicts the predicted versus the actual selling prices. It provides a clear visualization of the model’s performance, showing its high accuracy.

**Conclusion**

In conclusion, the predictive analysis developed a linear regression model to estimate smartphone selling prices. It demonstrated high accuracy, indicating how factors such as original price and discount significantly influence smartphone prices. Insights gained from this analysis can be valuable in developing pricing strategies and market positioning for stakeholders in the smartphone industry.

**References**

Bock, T. (2024, January 2). *What is a Correlation Matrix? - Displayr*. Displayr. <https://www.displayr.com/what-is-a-correlation-matrix/>

GfG. (2023, December 7). *Linear Regression (Python Implementation)*. GeeksforGeeks. <https://www.geeksforgeeks.org/linear-regression-python-implementation/>

Mali, K. (2024, January 23). *Everything you need to know about Linear Regression!* Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/>

Marcelino. (2022, April 30). *Comprehensive data exploration with Python*. Kaggle. <https://www.kaggle.com/code/pmarcelino/comprehensive-data-exploration-with-python>

*Smartphone Sales Dataset*. (2024, March 3). Kaggle. <https://www.kaggle.com/datasets/yaminh/smartphone-sale-dataset>