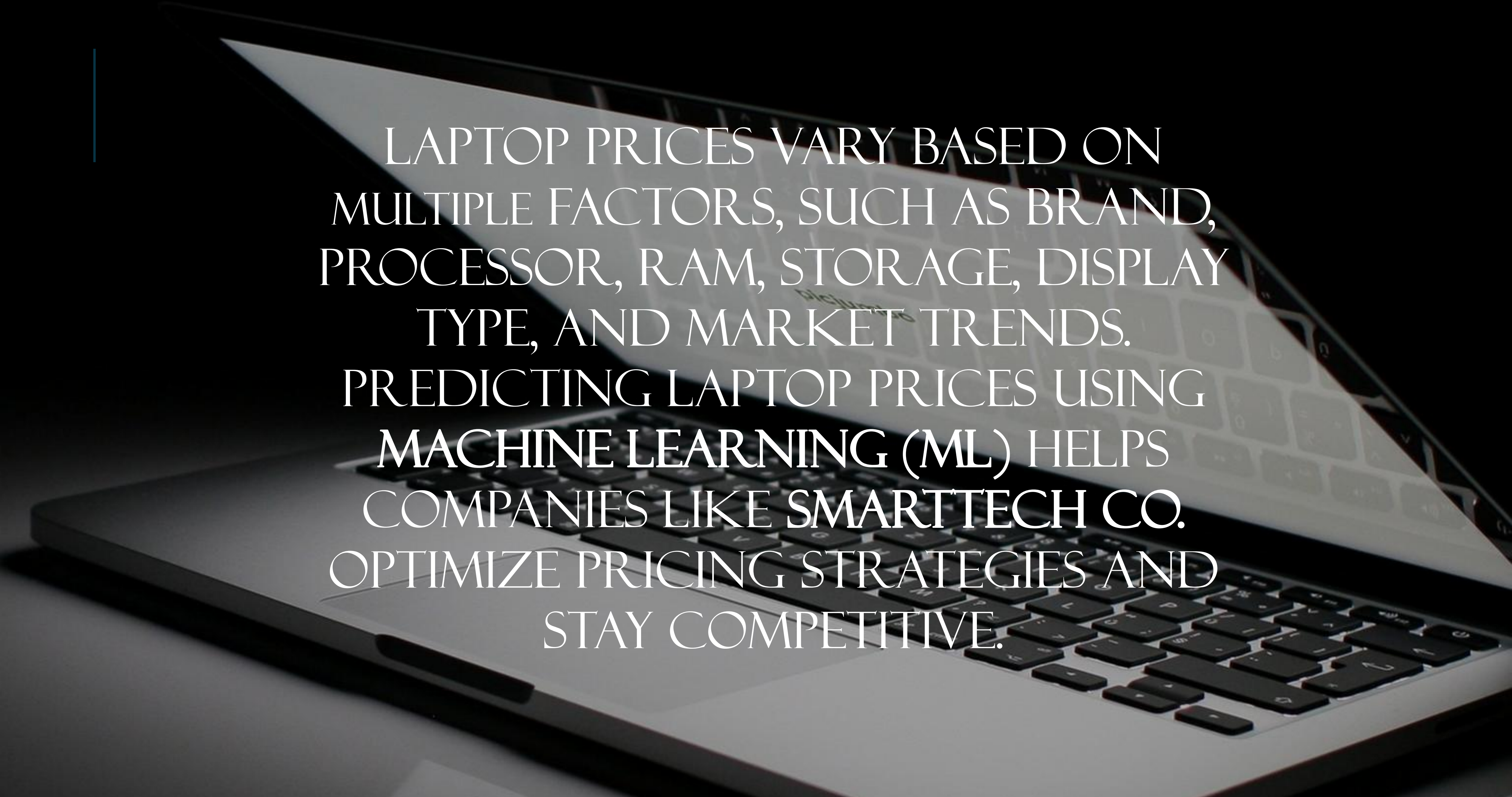
A MACHINE LEARNING APPROACH

A SMART TECH CO. STORY

PRESENTED BY: AJAY KUMAR

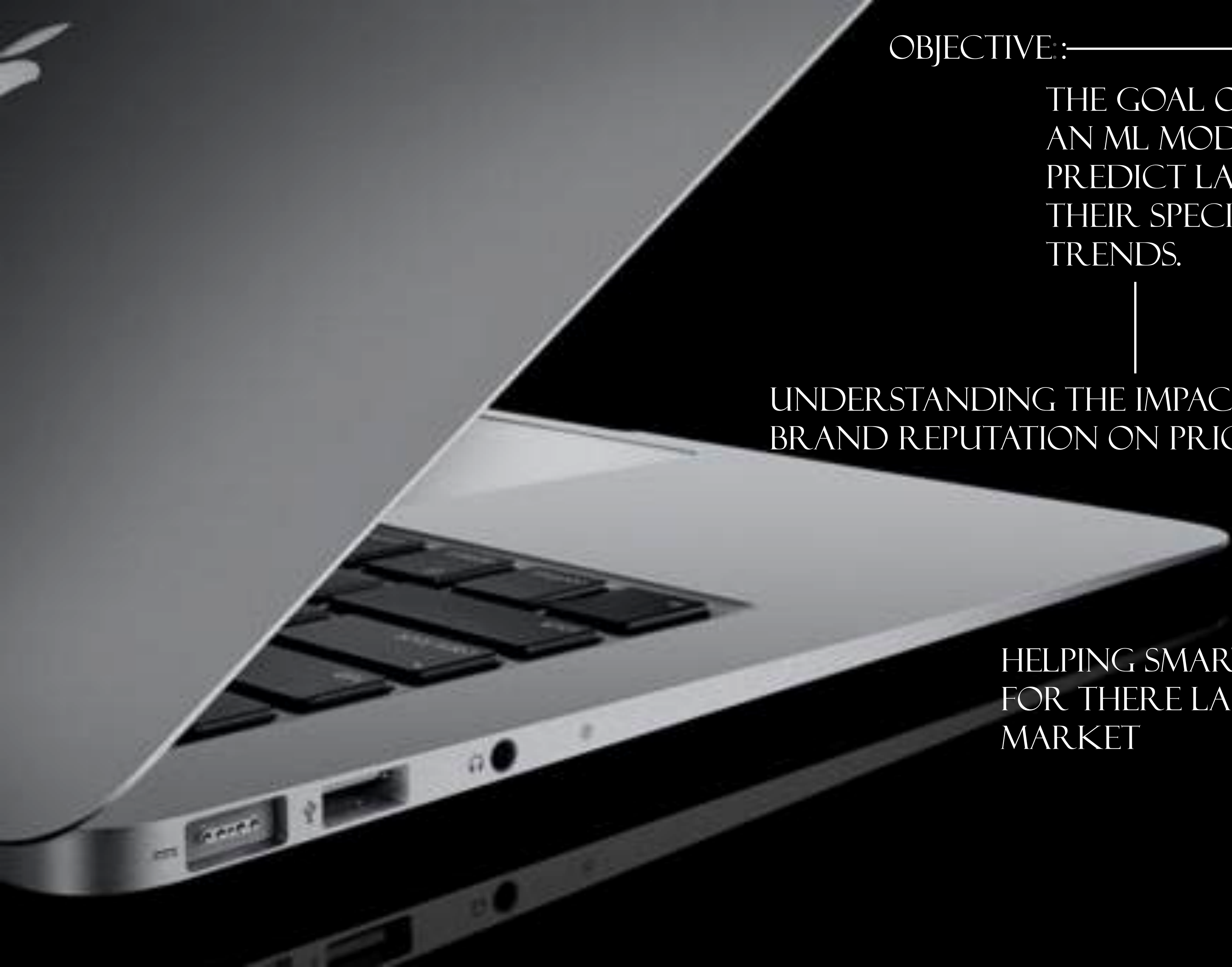


Laptop prices vary based on multiple factors, such as brand, processor, RAM, storage, display type, and market trends. Predicting laptop prices using **Machine Learning (ML)** helps companies like **SmartTech Co.** optimize pricing strategies and stay competitive.

Dataset & Features:

The dataset includes key laptop features such as:

- **Brand** (Dell, HP, Lenovo, Apple, etc.)
- **Processor** (Intel i3/i5/i7, AMD Ryzen)
- **RAM** (4GB, 8GB, 16GB, etc.)
- **Storage Type & Capacity** (HDD, SSD, 256GB, 512GB, etc.)
- **Graphics Card** (Integrated, NVIDIA, AMD)
- **Screen Size & Resolution**
- **Operating System** (Windows, macOS, Linux)
- **Price (Target Variable)**



OBJECTIVE :

THE GOAL OF THIS PROJECT IS TO BUILD AN ML MODEL THAT CAN ACCURATELY PREDICT LAPTOP PRICES BASED ON THEIR SPECIFICATIONS AND MARKET TRENDS.

UNDERSTANDING THE IMPACT OF BRAND REPUTATION ON PRICING

HELPING SMART TECH CO. TO POSITION FOR THERE LAPTOPS STRATEGICALLY IN MARKET

APPROACH :

Data collections

Data cleaning

Data Preprocessing

Exploratory Data Analysis

Model Development

Model Evaluation

DATA PREPROCESSING :

STEP 01 :

```python
from sklearn.preprocessing import LabelEncoder

print(type(df))
```

```
<class 'method'>
```

```python
#preprocessing

cat_col=['Company','TypeName','OpSys','Cpu','Gpu']
for col in cat_col:
    data[col] = LabelEncoder().fit_transform(data[col])
```

DATA PREPROCESSING :

STEP 02 :

```python
#converting weight col to numeric column

data['Weight']=data['Weight'].astype(str)

# replacing ? with float(0)
data['Weight']=data['Weight'].replace('?', float('0'))

#removig any leading and trailing spces  and kg convert to float
data['Weight']=data['Weight'].str.strip().str.replace('kg', '').astype(float)

#hand;ing NAN values with mean
data['Weight']=data['Weight'].fillna(data['Weight'].mean())

print(data['Weight'].head())
```

[66]

```
0    1.37
1    1.34
2    1.86
3    1.83
4    1.37
Name: Weight, dtype: float64
```

DATA PREPROCESSING :

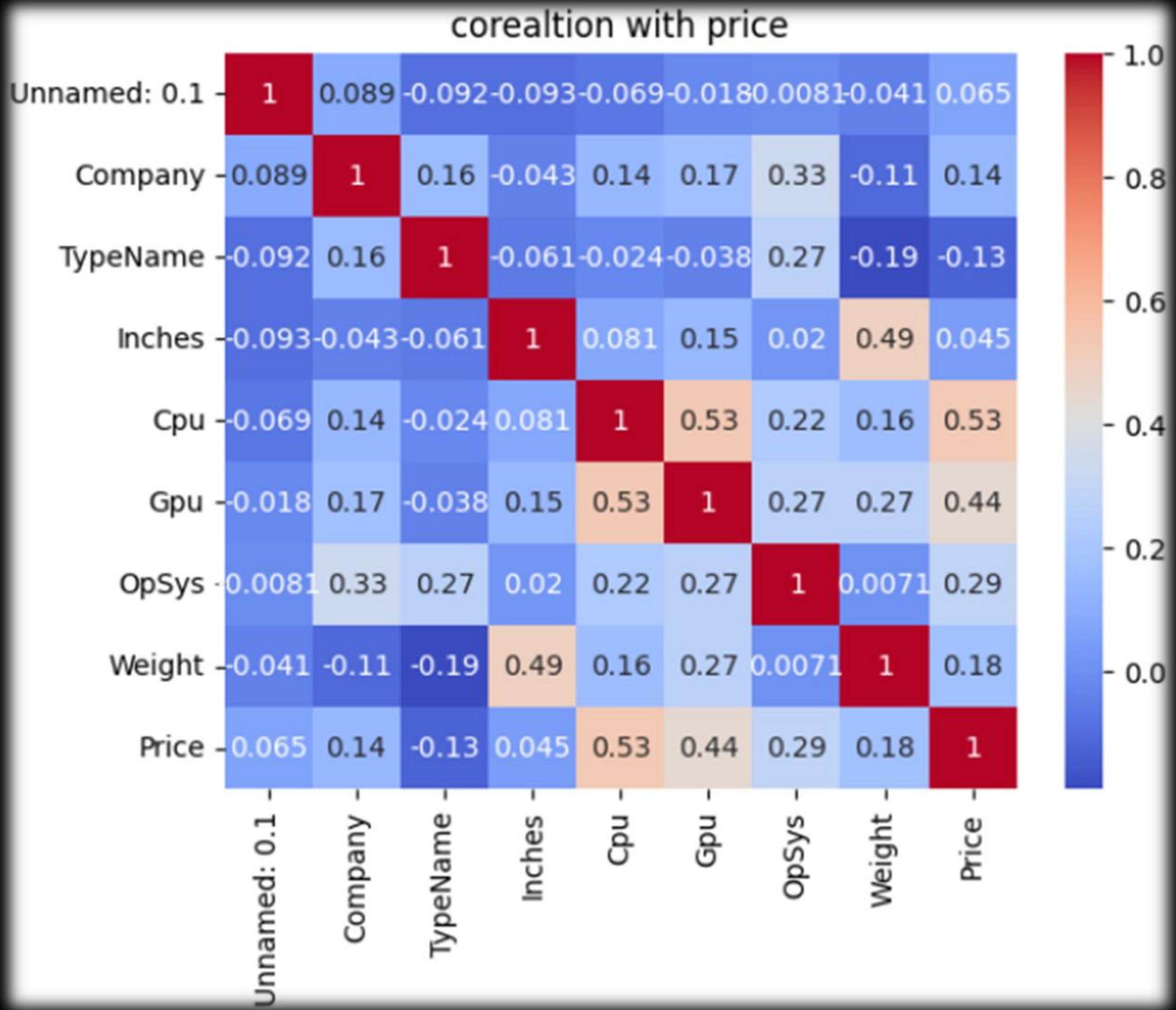
STEP 03 :

```
# Standardize numerical columns by importing class ' StandardScalar'
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()

data['Inches']=data['Inches'].astype(str)
data['Inches']=data['Inches'].replace('?', float('0'))
data['Inches']=data['Inches'].str.strip().astype(float)

num_cols = ['Inches', 'Weight']

# Apply standardization

data[num_cols] = scaler.fit_transform(data[num_cols])

# Step 5: Print first few rows to check the result

print(data[num_cols].head())
```

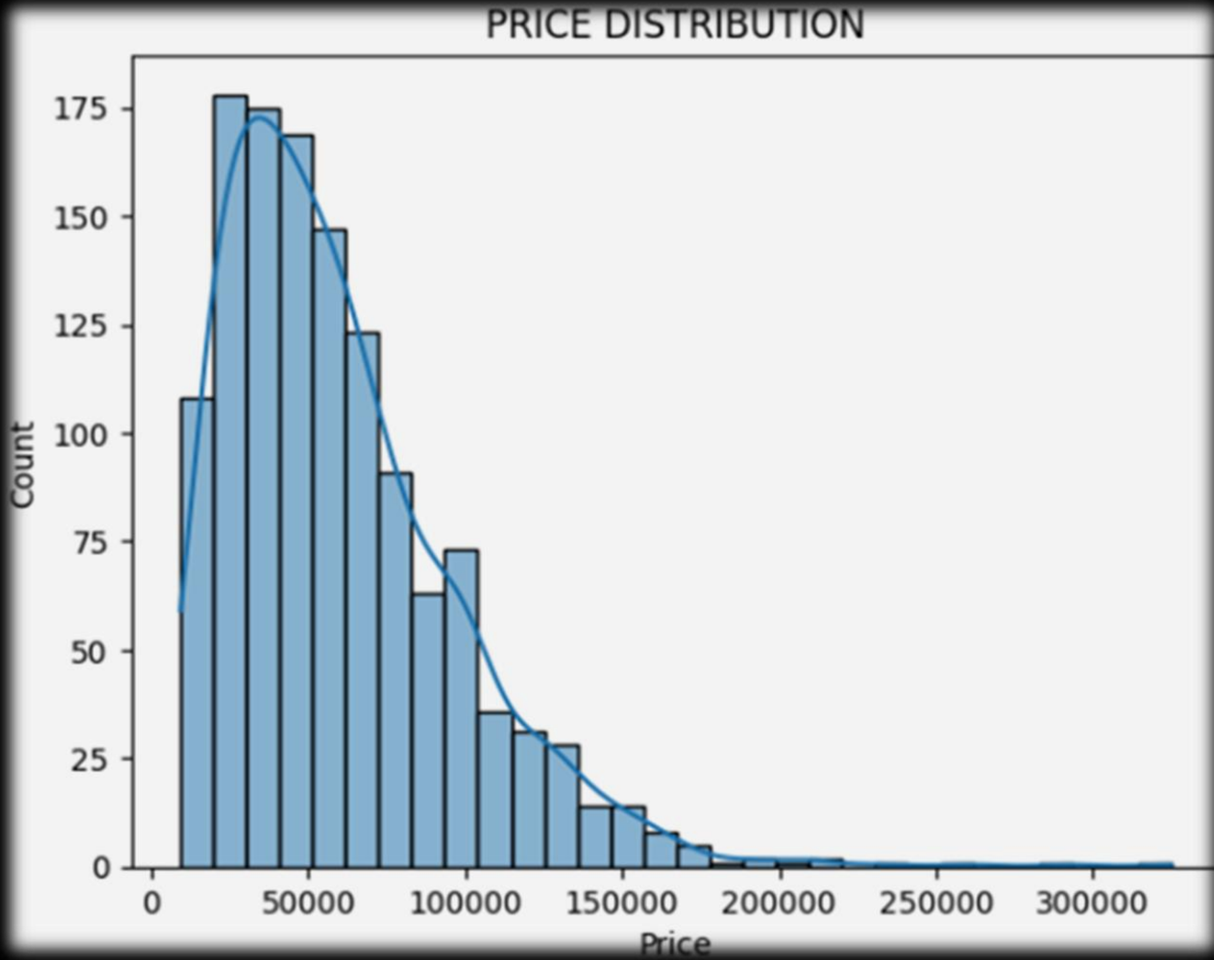
[67]

```
     Inches    Weight
0 -0.937118 -0.886932
1 -0.937118 -0.924534
2  0.240155 -0.272764
3  0.137783 -0.310366
4 -0.937118 -0.886932
```

EXPLORATORY DATA ANALYSIS :



EXPLORATORY DATA ANALYSIS :



MODEL DEVELOPMENT :

LINEAR REGRESSION MODEL :

```python
import numpy as np
# Linear Regression Model

from sklearn.impute import SimpleImputer


# Handle missing values by imputation (using the mean strategy)

imputer = SimpleImputer(strategy='mean')
X_train = imputer.fit_transform(X_train)
X_test = imputer.transform(X_test)
y_train = imputer.fit_transform(np.array(y_train).reshape(-1, 1)).ravel()


# Initialize the Linear Regression model
lr = LinearRegression()


# Fit the model to the training data
lr.fit(X_train, y_train)


# Make predictions on the test set
lr_preds = lr.predict(X_test)
```

MODEL DEVELOPMENT :

RANDOM FOREST REGRESSION MODEL :

```
# Initialize the Random Forest Regressor model
rf = RandomForestRegressor()

# FIt the model to the training data
rf.fit(X_train, y_train)

# Make predictions on the test set
rf_preds = rf.predict(X_test)
```

EVALUATION OF PERFORMANCES :

```python
from sklearn.impute import SimpleImputer

imputer_y = SimpleImputer(strategy="mean")
y_test = imputer_y.fit_transform(np.array(y_test).reshape(-1, 1)).ravel()

print('Linear Regression:')

print('RMSE:', np.sqrt(mean_squared_error(y_test, lr_preds)))

print('R2 Score:', r2_score(y_test, lr_preds))
```

```
Linear Regression:
RMSE: 20720.700454428752
R2 Score: 0.6984025318007299
```

EVALUATION OF PERFORMANCES :

```python
print('Random Forest Regressor:')

print('RMSE:', np.sqrt(mean_squared_error(y_test, rf_preds)))

print('R2 Score:', r2_score(y_test, rf_preds))
```

```
Random Forest Regressor:
RMSE: 16823.697699336222
R2 Score: 0.8011792358695081
```

COMPARISON OF MODELS :

```python
model_comparison = pd.DataFrame({
    'Model': ['Linear Regression', 'Random Forest Regressor'],

    'RMSE': [np.sqrt(mean_squared_error(y_test, lr_preds)),
             np.sqrt(mean_squared_error(y_test, rf_preds))],

    'R2 Score': [
        r2_score(y_test, lr_preds),
        r2_score(y_test, rf_preds)]
})

print(model_comparison)
```

	Model	RMSE	R2 Score
0	Linear Regression	20720.700454	0.698403
1	Random Forest Regressor	16823.697699	0.801179

KEY INSIGHTS

FEATURE IMPORTANCE

. brand
. processor & ram
. screen size

. Higher- end specification correlate with higher prices
. Lesser known brands exhibit competitive price

PRICING VARIABILITY

CATEGORICAL FACTOR

. Operating systems (E.g., MacOs , windows) contribute to price variation

CONCLUSION

Empowering SmartTech co. With Insights

The **Laptop Price Prediction** project uses **machine learning** to estimate laptop prices based on key specifications like **brand, processor, RAM, and storage**. This helps businesses make **data-driven pricing decisions** and stay competitive. Improving the model with **real-time data** can further enhance accuracy and insights.

- Apple
- HP
- Acer
- Asus
- Dell
- Lenovo
- nan
- Chuwi
- MSI
- Microsoft
- Toshiba
- Huawei
- Xiaomi
- Vero
- Razer
- Mediacom
- Samsung
- Google
- Fujitsu
- LG

Thank you for Exploring The World of Laptop Pricing

ajaymudiraj8790@gmail.com