MARKETFORECASTING

ARTIFICIALINTELIGENCE

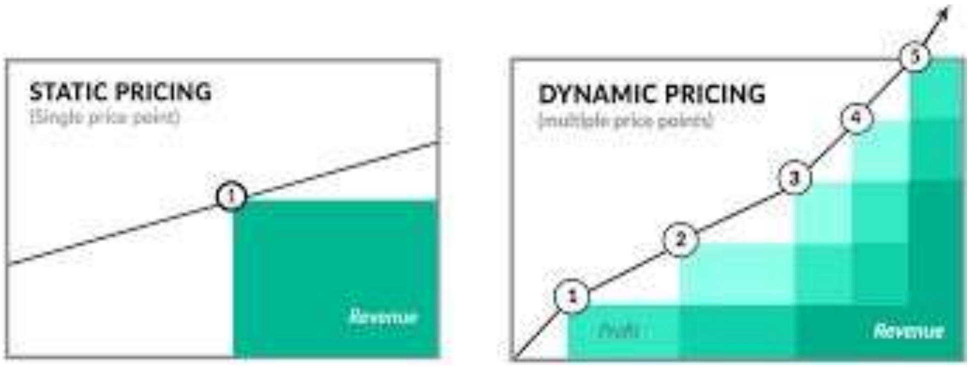
PROJECTREPORT

Submitted by

Yeswanth Kumar Muttha 801398789
Hemanth Borra 801428928
Adarsh Kodumuru 801365902

INTRODUCTION

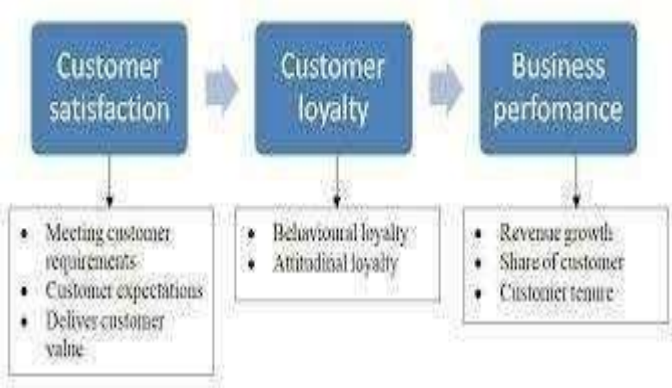
Traditional costing methods are accounting approaches designed to calculate the production cost of a product with the goal of achieving profitability. This system distributes indirect manufacturing expenses, known as overhead, across various products. In contrast, price optimization is a revenue-maximization strategy where businesses set prices based on how much customers are willing to pay. This strategy involves assessing consumer behavior to identify the most profitable price point. Key elements in this approach include analyzing competitor prices, evaluating customer demand, and monitoring market trends. By customizing prices for different customer groups, companies can increase revenue while enhancing customer satisfaction.



Static and Dynamic Pricing

GOAL

The objective is to find the most effective price point for mobile phones—one that maximizes profits, minimizes customer dissatisfaction, and improves overall customer satisfaction. This optimal pricing should be determined by conducting thorough market research, collecting customer feedback, and evaluating competitors' pricing strategies.



Company Objectives

PROBLEM STATEMENT

Problem: With the launch of new mobile phone technology, what is the best approach to set an optimal price that will maximize profitability? Which critical factors should be evaluated in determining this price?

Goal: To find the ideal sales price for a mobile phone by examining the current demand curve and identifying key factors that drive demand, with the ultimate goal of maximizing profits.

Fig 1.4: Mobile phone sales for the company.

Project Objectives Overview

This project is designed with the following primary objectives:

1. **Create an Advanced Pricing Model**: Apply machine learning algorithms to forecast and dynamically adjust mobile phone prices in real time, factoring in demand, competitor prices, and other crucial variables.

2. **Overcome the Constraints of Traditional Pricing Models**: Enhance traditional cost-based pricing approaches by incorporating real-time market dynamics into the pricing model.

3. **Increase Profitability**: Determine optimal pricing for mobile devices that will boost sales, maximize profit margins, and maintain competitive positioning in the market.

4. **Assess Model Efficiency**: Measure the performance of the machine learning pricing model against traditional methods to showcase its flexibility and effectiveness in responding to market changes.

These objectives shape the project, enabling accurate predictions of ideal sales prices.

Approach

The project approach includes the following critical steps:

System Requirements: Specify the necessary hardware and software tools, including programming languages like Python, development environments such as Google Colab, and operating systems like Windows 11.

Data Preparation: Organize and clean the dataset for analysis by filling in missing values, encoding categorical data, and standardizing data values.

```python
for i in data.columns:
    print(i,len(pd.unique(data[i])))
```

```
brand 5
model 136
base_color 41
processor 10
screen_size 5
ROM 9
RAM 7
display_size 20
num_rear_camera 4
num_front_camera 3
battery_capacity 34
ratings 11
num_of_ratings 196
sales_price 158
discount_percent 36
sales 248
sales_increment 43
Market price 218
manufacturing cost 229
avg_profit 218
avg_units_sold 77
```

```python
data.isnull().sum()
```

```
brand                 0
model                 0
base_color            0
processor             0
screen_size           0
ROM                   0
RAM                   0
display_size          0
num_rear_camera       0
num_front_camera      0
battery_capacity      0
ratings               0
num_of_ratings        0
sales_price           0
discount_percent      0
sales                 0
sales_increment       0
Market price          0
manufacturing cost    0
avg_profit            0
avg_units_sold        0
dtype: int64
```

```python
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 42)
```

```python
from sklearn.preprocessing import MinMaxScaler
norm = MinMaxScaler()
norm_fit = norm.fit(x_train)
new_xtrain = norm_fit.transform(x_train)
new_xtest = norm_fit.transform(x_test)
print(new_xtrain)
```

Model Implementation

This phase involves deploying various regression models—including Linear Regression, Ridge Regression, Lasso Regression, and Random Forest Regression—to forecast ideal pricing. The process encompasses training each model on the dataset, fine-tuning hyperparameters, and evaluating overall model accuracy and performance.

```python
d={'selling_price':unit_price,'optimal_selling_price':optimal_prices }
result=pd.DataFrame(data=d)
print(result.head())
p=list(optimal_prices)
```

```
   selling_price  optimal_selling_price
0          39757                  35781
1          59530                  71436
2          39757                  35781
3          47776                  48280
4          70560                  84671
```

Optimal selling price without employing a machine learning algorithm.

```
d={'selling_price':data2["Market price"],'optimal_selling_price':optimal_prices }
result=pd.DataFrame(data=d)
print(result.head())
q=list(optimal_prices)
```

	selling_price	optimal_selling_price
0	39757	32999
1	59530	72967
2	39757	33330
3	47776	48280
4	70560	84025

Optimal mobile selling price after applying a machine learning algorithm.

```
rnd = RandomForestRegressor(n_estimators = 50, random_state = 42)
rnd.fit(x_train, y_train)
x_predict = list(rnd.predict(x_test))

predicted_df = {'predicted_values': x_predict, 'original_values': y_test}

pd.DataFrame(predicted_df).head(10)
```

	predicted_values	original_values
0	14134.54	15973
1	46753.80	65788
2	52746.00	52746
3	36511.38	39757
4	19474.68	18946
5	70151.84	70560
6	36781.84	31913
7	33805.18	31706
8	60806.60	59530
9	21966.34	15956

Random Forest Regression estimates the selling price by analyzing historical data through multiple decision trees.

```
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor(random_state = 10)
regressor.fit(x_train, y_train)
x_predict = list(regressor.predict(x_test))
#x_pred = regressor.predict([[3750]])
predicted_df = {'predicted_values': x_predict, 'original_values': y_test}

pd.DataFrame(predicted_df).head(10)
```

	predicted_values	original_values
0	13986.0	15973
1	26803.0	65788
2	52746.0	52746
3	39757.0	39757
4	18946.0	18946
5	70560.0	70560
6	31913.0	31913
7	31706.0	31706
8	59530.0	59530
9	25760.0	15956

Estimating the selling price using Decision Tree Regression involves leveraging the model's ability to predict prices based on historical data and input features.

Model testing and evaluation focus on determining the accuracy and reliability of each machine learning approach in predicting the ideal price. This process also examines how well the models influence both profitability and the product's position in the market, ensuring they contribute to competitive pricing strategies.

```
[ ]  a=regressor.score(x_test, y_test)
     print('score of model is : ',a)
     p.append(a)
     q.append("DecisionTreeRegressor")

     score of model is :  0.8204159046298444

[ ]  lr = LinearRegression()
     %time lr.fit(x_train, y_train)
     a = lr.score(x_test, y_test)
     print(a)
     p.append(a)
     q.append("LinearRegression")

     CPU times: user 6.35 ms, sys: 3.07 ms, total: 9.42 ms
     Wall time: 22.5 ms
     0.8837308059069138

[ ]  ridge = RidgeCV()
     %time ridge.fit(x_train, y_train)
     a=ridge.score(x_test, y_test)
     print(a)
     p.append(a)
     q.append("RidgeCV")

     CPU times: user 5.59 ms, sys: 9.46 ms, total: 15 ms
     Wall time: 13.2 ms
     0.8830906608750886
```

Model Performance Evaluation

Expected Results

We conducted a thorough comparison between the results obtained from our initial demand equation model and those generated by various machine learning algorithms applied to our dataset. The analysis included the following algorithms:

- Linear Regression
- Ridge Regression
- Lasso Regression
- Random Forest

This approach allowed us to assess the performance disparities between traditional statistical methods and contemporary machine learning techniques, focusing on their accuracy in predicting demand. By evaluating these models, we aimed to determine which method most effectively forecasts demand and how well they perform in comparison to one another.

Linear Regression:

```
lr = LinearRegression()
%time lr.fit(x_train, y_train)
a = lr.score(x_test, y_test)
print(a)
p.append(a)
q.append("LinearRegression")
```

```
CPU times: user 5.88 ms, sys: 4.15 ms, total: 10 ms
Wall time: 21.8 ms
0.8837308059069138
```

Accuracy of Linear regression

The model demonstrated an accuracy of 88%.

Ridge Regression:

```
ridge = RidgeCV()
%time ridge.fit(x_train, y_train)
a=ridge.score(x_test, y_test)
print(a)
p.append(a)
q.append("RidgeCV")
```

```
CPU times: user 9.48 ms, sys: 14.3 ms, total: 23.8 ms
Wall time: 16.7 ms
0.8830906608750886
```

Accuracy of Ridge regression

We then evaluated the Ridge Regression model's performance.

Lasso Regression:

Next, we examined the performance of the Lasso Regression model.

```
lasso = LassoCV()
%time lasso.fit(x_train, y_train)
a=lasso.score(x_test, y_test)

print(a)
p.append(a)
q.append("LassoCV")
```

```
CPU times: user 85.6 ms, sys: 141 ms, total: 226 ms
Wall time: 130 ms
0.45448499645912366
```

Accuracy of Lasso regression The

Lasso Regression model achieved an accuracy of 45%.

Comparison of regression models

	Accuracy
Linear Regression	0.883731
Ridge Regression	0.883091
Lasso Regression	0.454485

Random Forest regression:

```
a=rnd.score(x_test, y_test)
print('score of model is : ',a)
p.append(a)
q.append("RandomForestRegressor")
```

```
score of model is :  0.9396465030210591
```

The Random Forest Regression model delivered the best accuracy for this dataset, reaching an outstanding 93.9%.

Bibliography

[1] Q. Huang and F. Zhou, "Research on Retailer Data Clustering Algorithms Using Spark," *AIP Conference Proceedings*, vol. 1820, no. 1, pp. 080022, March 2017.

[2] A. Sayli, I. Ozturk, and M. Ustunel, "Brand Loyalty Analysis System Through the K-Means Algorithm," *Journal of Engineering Technology and Applied Sciences*, vol. 1, no. 3, 2016.

[3] M.N. Maingi, "An Overview of Clustering Algorithms in Sales Data Mining."

[4] S.H. Sastry, P. Babu, and M.S. Prasada, "Sales Data Analysis and Prediction in SAP ERP Systems Using Clustering Algorithms," 2013.

[5] V. Shrivastava and N. Arya, "A Comparative Analysis of Various Clustering Algorithms Applied to Retail Sales Data," *International Journal of Computer Communications and Networks*, vol. 1, no. 2, 2012.

[6] D. Rajagopal, "Using Data Mining Techniques for Customer Data Clustering," 2011.

[7] C.F. Tsai, H.C. Wu, and C.W. Tsai, "A Novel Method for Data Clustering in Large Database Mining," *Parallel Architectures, Algorithms, and Networks 2002. ISPAN'02. Proceedings of the International Symposium*, pp. 315-320, 2002.

[8] A.K. Mann and N. Kaur, "A Comprehensive Review of Clustering Techniques," *Global Journal of Computer Science and Technology*, 2013.

[9] N. Shah, M. Solanki, A. Tambe, and D. Dhangar, "Efficient Mining Techniques for Sales Prediction."

[10] M. Korolev and K. Ruegg, "Application of Gradient Boosted Trees for Store Sales Prediction," 2015.