Smartphone Price Patterns Prediction Using Machine Learning Algorithms

1st Shaik Rafi
Dept of CSE,
Narasaraopeta Engineering College,
Narasaraopet-522601,Palnadu,
Andhra Pradesh,India
shaikrafinrt@gmail.com

2nd Yalagala Rambabu
Dept of CSE,
Narasaraopeta Engineering College,
Narasaraopet-522601,Palnadu,
Andhra Pradesh,India
ramsy55550000@gmail.com

3rd Poluri Rajasekhar
Dept of CSE,
Narasaraopeta Engineering College,
Narasaraopet-522601,Palnadu,
Andhra Pradesh,India
p.rajasekhar567@gmail.com

4th Bolnedi Suhas
Dept of CSE,
Narasaraopeta Engineering College,
Narasaraopet-522601,Palnadu,
Andhra Pradesh,India
Suhasbolnedi@gmail.com

5th M.Sathyam Reddy
Dept of CSE,
Narasaraopeta Engineering College,
Narasaraopet-522601,Palnadu,
Andhra Pradesh,India
sathyamreddym@gmail.com

6th Sireesha Moturi
Dept of CSE,
Narasaraopeta Engineering College,
Narasaraopet-522601, Palnadu,
Andhra Pradesh, India
sireeshamoturi@gmail.com

7th Venkat Reddy Doda
Dept of CSE,
Narasaraopeta Engineering College,
Narasaraopet-522601, Palnadu,
Andhra Pradesh, India
doddavenkatareddy@gmail.com

***Abstract*—Selecting the best smartphone can be challenging due to the wide range of models available on the market. This study shows how the machine learning models can predict mobile phone prices based on their features We evaluated several machine learning techniques, including Logistic Regression, Decision Trees, Random Forest, SVC, K-Neighbors Classifier, Gaussian Naive Bayes (GaussianNB), AdaBoost, Gradient Boosting, Extra Trees, Bagging Classifiers, and XGBoost. The primary objective was to identify the most effective model for price forecasting and to investigate the factors influencing phone prices. Our research offers insights to both consumers and manufacturers, helping them make more informed decisions about phone features and pricing. We emphasize the importance of using diverse datasets that accurately represent various smartphone models and pricing points. Key factors affecting phone costs were identified, and model performance was assessed using metrics such as accuracy, F1-score, and classification reports. Model performance was further enhanced through hyperparameter tuning with GridSearchCV, achieving 97% accuracy with the Decision Tree, K-Neighbors Classifier, SVC, AdaBoost, and Random Forest models. Among these, the Decision Tree and SVC was selected as the optimal models, offering a good trade-off between accuracy, flexibility, and time complexity. This study aims to provide valuable data to guide consumers in making informed choices about mobile phone features and price ranges.**

***Index Terms*—mobile device cost, cell phone choices, pricing determinants, device features, manufacturer choices.**

I. INTRODUCTION

With the advancement of technology, the demand for smartphones is at an all-time high. There are many phone models in the market, and quite often, it creates confusion among customers to decide on the best phone model. Machine learning can solve such issues. Recently, a mobile phone price prediction was done using attributes of the phones based on machine learning algorithms. In this present paper, the prediction of mobile phone prices will be done by the following machine learning methods: Logistic Regression, Decision Tree, Random Forest, SVM, K-Neighbors Classifier, GaussianNB, AdaBoost Classifier, Gradient Boosting Classifier, Extra Trees Classifier, Bagging Classifier, and XGBoost [1]. The key objectives of this study will revolve around indicating the best machinery learning model to accomplish this task, while also determining what factors influence the prices of mobile phones. These findings could help manufacturers and consumers make quite realistic decisions on the specifications of phones and prices of phones [2]. The high demand for smartphones in the fast-moving technology environment makes selection quite difficult for the customer. However, a number of advances in machine learning offer a possible solution. For instance, it is possible to predict mobile phone prices based on their unique features through the use of a machine learning algorithm. This paper follows the use of machine learning for predicting mobile phone prices as done in [1].

II. Comparative Analysis of Existing Work

During the past few years, machine learning-based price prediction, especially for mobile phones, became a trendy area of research. Most of these studies compare different algorithms to improve their forecasting accuracy, thus offering unique insights related to the effectiveness of these methods. Khan and Asim applied SVM and logistic regression to predict mobile phone prices based on features related to the quality of cameras, screen size, and storage capacity. The authors were able to show that SVM achieves about 92% accuracy in price range classification. This model is highly valued as this model separates classes quite aptly for price prediction tasks [3]. Zhang et al. have tried ensemble methods such as random forests and decision trees. They observed the superiority of the former over the latter in predicting the prices of mobile phones on an accuracy of around 95%. In this domain, random forests are a better choice because of their robustness as well as the ability to manage large datasets with multiple features [4]. Fofanah (2021) proposed a deep learning model in the form of Convolutional Neural Networks (CNNs) to predict mobile phone prices, focusing on visual attributes. It is achievable to attain the accuracy level of up to 90%. Thus, it is easy to add image features into the predictive model. Such an approach can be applied to those cases where the aesthetic value of the product enormously contributes to its market value [5]. Recent breakthroughs in ensemble learning have brought prediction accuracy to a higher level. For example, stacking and AdaBoost methods are effective in mobile phone price prediction tasks. Using ensemble methods, known as stacking, which combined models, one managed to achieve 96% accuracy; using AdaBoost, which emphasizes improvement of weak learners, obtains 94%. These improve the accuracy of the prediction model, based on the merits of different algorithms [6]. Sharma et al. has demonstrated that the existing market trends, which include seasonal demand shifts and new product releases, can be included in their model to allow for more precise predictions of prices. Adding time-series features to the model allows it to reach an accuracy as high as 95%, which would indicate how the recommendation needs to be contextual for the fast-moving technology marketplace [7]. Overall, this work educates us about what machine learning can do: predict the price of mobile phones reasonably enough. And any increase in accuracy, especially by ensemble methods and deep learning approaches, would itself be a useful reservoir of information for more complex and reliable pricing models. Interestingly, it turns out that such development is fundamental, both for online marketplaces and for manufacturers, as a source of competitive advantage concerning the rapidly changing technology environment.

III. Methodology

This flow diagram shows how the entire process works to predicts the best results for the mobile phone price ranges.

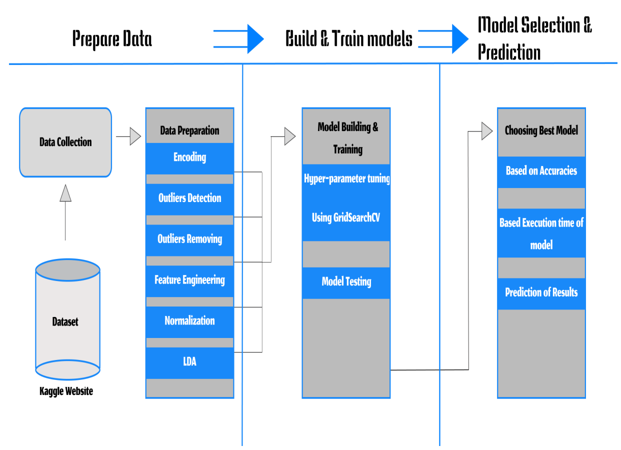

Fig. 1. Work flow

A. *Data Collection*

The dataset used in the present study has been taken from Kaggle. It is a very popular tool among data scientists and machine learning practitioners, providing practical tools and materials for professionals and scholars. This dataset has been further utilized containing 2000 observations and 21 features.

*1) **Data Overview**:* Features of the data-set are Battery-power, Blue, Clock-speed, Dual-sim, fc, Four-g, int-memory, M-dep, Mobile-wt, N-cores, pc, Px-height, Px-width, ram, Sc-h, Sc-w, talk-time, Three-g, touch-screen, WiFi, and price-range. The dataset link for further studies: Click to get dataset

B. *Data Preprocessing*

*1) **Data visualization**:* Data visualization is an integral step in the analysis of data as it makes complex data more accessible and allows for the communication of results to stakeholders intuitively and interactively.

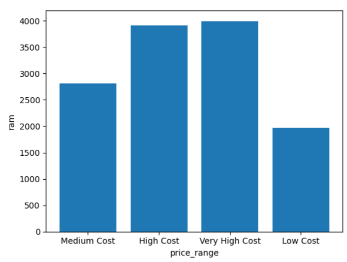

Fig. 2. price range vs ram

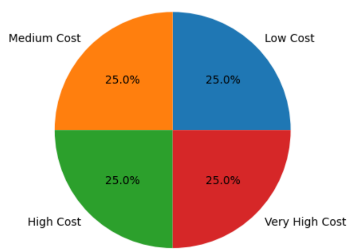

Fig. 3. Price range Distribution

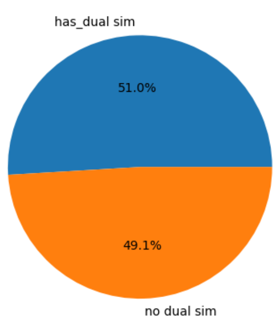

Fig. 4. Dual sim Availability

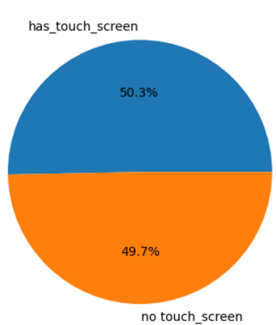

Fig. 5. Touch Screen Availability

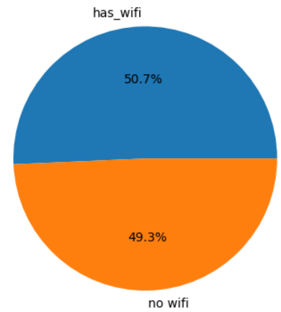

Fig. 6. Wi-Fi Availability

The process of conversion of raw data into useful graphs and charts, etc [8].These plots are a good summary of the distribution of the key features of the mobile phone dataset and their relation to price [8].

*2) **Data Cleaning** :* In this study, there is no “missing values” in the dataset. ”data.isnull ().sum()” by using this statement we get ”0” for all columns (features). The process for identifying and removing duplicate records from the dataset is called ”removing duplicates”. But in this study we get ”0” duplicates in the dataset.

*3) **Data Transformation**:* Following this, all categorical variables in the data had to be changed into numerical values: performing label encoding on the feature of price-range, encoding ’Medium Cost’ to the value 1, ’High Cost’ to 2, ’Low Cost’ to 0, and ’Very High Cost’ to 3. remaining categorical features are blue, dual-sim, four-g, three-g, touch-screen, wifi, encoded into binary values 0’s and 1’s [9].

*4) **Outliers detection and Outliers removing**:* Outliers detection and Outliers removing in this study are made using the IQR Method. Interquartile Range (IQR) technique will be used to identify outliers in the dataset.

*5) **No Outliers Found**:* Outliers were detected in battery-power, blue, clock-speed, dual-sim, four-g, int-memory, m-dep, mobile-wt, n-cores, pc, px-width, ram, sc-h, sc-w, talk-time and touch-screen, wifi.

*6) **Outliers Identified**:*

- “(fc)front-camera-mpixels”: outliers detected in 17 entries, the values much higher - for instance, 17 or 18 MP.
- “px-height”: Two entries have abnormally high pixel height values of 1949 and 1960.
- “Three-g”: 477 entries as outliers, probably because of binary values or values that don’t occur very often.

*7) **Removing Outliers**:* By removing the rows where values in the current column fall outside the calculated bounds-that is, effectively removes outliers from the dataset using the IQR method.

C. *Feature Engineering*

In this case study, Chi-square, chi2 testing was used for categorical features. This statistical test counts the independence of categorical variables and hence assists in the selection of those features that are highly related to the target variable. For numerical features, the study employed Lasso regression. Lasso regression uses the regularization method to shrink the coefficients towards zero. This is the process whereby this effectively zeroes out features less important and hence a subset of relevant features is selected [10].

*1) **Chi-2 Method**:* chi-2(square) test = chi-2(x, y): for every attribute in x calculate the chi-squared statistic with respect to the target variable y. In the below figure, chi-squared test results show that the feature ’has-4g’, ’touch-screen’, ’bluetooth’, ’dual-sim’ and ’wifi’ are very significant while ’has-3g’ feature is showing least importance. The important features are: ’has-4g’, ’touch-screen’,’ bluetooth’, ’dual-sim’ and ’wifi’.The feature that is insignificant here is ’has-3g’. In general, features with higher chi-squared values will have a higher degree of association with the target variable and will, therefore, be more important in making predictions. our machine learning model will thus allow its performance to be enhanced by prioritizing more relevant information [11].

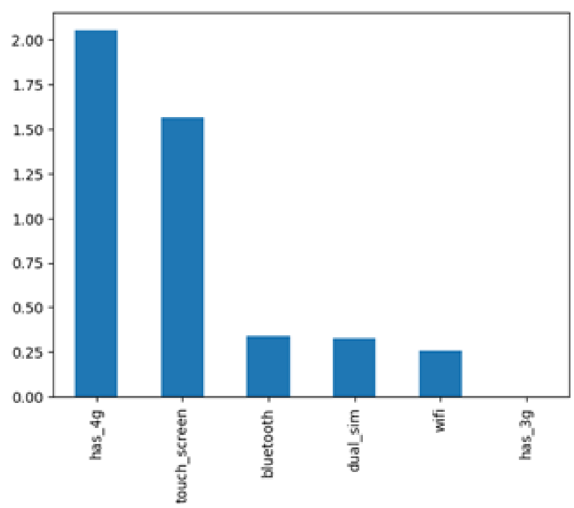

Fig. 7. chi-2(square) test

*2) **Lasso Method**:* Useful for features selection in continuous valued data. these are the important features after doing lasso method: ’ram’, ’mobile-wt’, ’Battery-power’, ’Int-memory’, ’Px-height’ and ’Px-width’. The following features have coefficients near zero and are less important, so these are unimportant features: ’talk-time’, ’Sc-w’, ’Sc-h’, ’N-cores’, ’M-dep’, ’fc’(front-camera-mpixels), ’pc’(Primary Camera mega pixels), Clock-speed. removing unimportant feature from data set. The features are ’has-3g’, ’n-cores’, ’sc-h’, ’sc-w’, ’talk-time’, ’m-dep’, ’pc’, ’front-camera-mpixels’ and ’clock-speed’.

Following the removal of the non-key features, the dataset is 1506 rows × 11 columns without price-range. The eleven columns are ram, touch-screen, px-width, px-height, wifi, mobile-wt, has-4g, dual-sim and bluetooth [11].

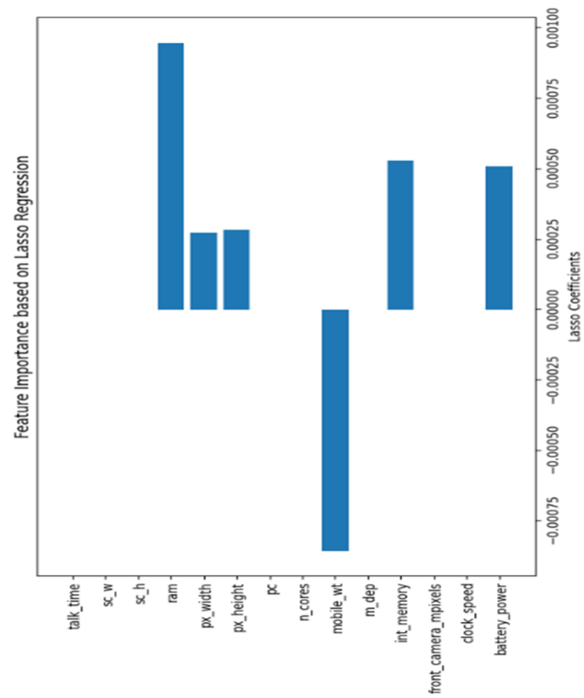

Fig. 8. Lasso method

D. *Normalization*

In this study, the data set is splits into two portions. setting the test size to 20 percent, so the training data makes up 80 percent of the total and used a random state of 42 for reproducibility. Applying normalization technique to the dataset of all features without target variable ‘pricerange’. the data set values lie in between 0 and 1 [8]. After applying standard scaler technique, the machine can easily understand the dataset information [12].

E. *Model Training*

The ”LDA-Linear Discriminant Analysis” was used for the analysis of this work. It maximizes class separability and reduces the dimensions, giving the maximum classification performance. Hyperparameter tuning of different machine learning models by using GridSearchCV: The parameter grids for different algorithms:

*1) **RandomForest Classifier** :* In the case of Random Forest, the hyperparameter grid includes a range of ’n-estimators', the number of trees in the forest, set to 100, 200, and 300. The 'max-depth' of each tree is either left unlimited (None) or capped at 5 or 10 to control overfitting. Additionally, the 'min-samples-split', which governs the minimum number of samples needed to split an internal node, is varied between 2, 5, and 10.

*2) **DecisionTree Classifier**:* In Decision Tree models, the 'max-depth' of the tree is explored with values of None (no limit), 5, 10, and 15, allowing for flexibility in tree complexity. The 'min-samples-split' is tested with 2, 5, and 10, and the 'min-samples-leaf', the minimum number of samples at a leaf node, is set at 1, 5, and 10. For the splitting criterion, both Gini impurity and entropy are considered as 'criterion' values.

*3) **LogisticRegression**:* For Logistic Regression, the inverse regularization strength, 'C', is tested with a variety of values including 0.001, 0.01, 0.1, 1, and 10. The 'penalty' terms include both 'l1' (lasso) and 'l2' (ridge) regularization methods. The solvers 'liblinear' and 'saga' are used depending on the penalty type, and the 'max-iter' is adjusted to 100, 200, and 300 to ensure convergence.

*4) **K-Nearest Neighbors (KNN)**:* In K-Neighbors Classifier, the number of neighbors, 'n-neighbors', is explored through 12 values generated between 2 and 20. The 'weights' are either 'uniform' or 'distance', where the former gives equal weight to all points, while the latter gives more influence to closer neighbors. The 'metric' used for distance calculation includes 'euclidean', 'manhattan', and 'minkowski'. Additionally, 'leaf-size' is varied with values of 1, 3, 5, and 12 [13].

*5) **Support Vector Classifier (SVC)**:* For Support Vector Classifier (SVC), the 'kernel' is set to 'rbf', focusing on the radial basis function. The 'gamma' parameter, controlling the influence of a single training example, is tested with 0.001, 0.01, and 0.1. The 'C' value, which defines the trade-off between correct classification and margin maximization, takes values of 0.1, 1, 10, 50, and 100.

*6) **Gradient Boosting Classifier (GBC)**:* This section tests Gradient Boosting using the 'learning-rate' hyperparameter of values 0.05, 0.1, and 0.2 to control the contribution of each tree. The 'min-samples-split' is set at 2, 3, and 10, and the 'min-samples-leaf' at 1, 3, and 10, helping to avoid overfitting by ensuring sufficient samples in each leaf.

*7) **AdaBoost Classifier** :* In AdaBoost, 'n-estimators' is explored with values of 50, 100, and 200, and the 'learning-rate' is varied to 0.01, 0.1, and 1.0. The 'algorithm' can either be 'SAMME' or 'SAMME.R'; most of the time, 'SAMME.R' outperforms because of its real-valued predictions.

*8) **ExtraTrees Classifier**:* 'n-estimators' for Extra Trees is set to 50, 100, and 200, while the 'max-depth' is tested against values of 1, 3, and 5. 'min-samples-split' and 'min-samples-leaf' are set with values of 2, 5, and 10 to control node splitting. Features considered as 'criterion' for the split are both 'gini' and 'entropy'. The 'bootstrap' parameter, which controls whether sampling should include replacement, is toggled between True and False; 'max-features' used on each split includes None, 'sqrt', and 'log2'.

*9) **Bagging Classifier** :* For the Bagging Classifier, 'n-estimators' is explored with values of 50, 100, and 200. The 'max-samples' and 'max-features' are varied at 0.5, 0.8, and 1.0, determining the proportion of samples and features used for each model. 'bootstrap' and 'bootstrap-features' are also toggled between True and False, controlling whether both samples and features are drawn with replacement.

*10) **XGBoost Classifier** :* For XGBoost, the 'max-depth' of trees is explored with values of 3, 5, 7, and 9, while the 'learning-rate' is set at 0.01, 0.1, 0.5, and 1.0. The 'n-estimators' is similarly varied at 10, 50, 100, and 200. Other hyperparameters include 'gamma', which reduces overfitting by controlling the minimum loss reduction needed for further partitioning, set at 0, 0.1, 0.5, and 1.0. The 'subsample' fraction, determining the portion of data used for each tree, is tested with 0.5, 0.7, 0.9, and 1.0.

*11) **GaussianNB** :* Gaussian Naive Bayes uses the 'var-smoothing' parameter to stabilize variances, tested with values ranging from 1e-9 to 1e-5. For 'priors', multiple distribution configurations are considered, including None or fixed probability distributions such as 0.3, 0.4, and 0.2. The above mentioned, algorithms are trained by using hyperparameter tuning of GridSearchCV.

IV. Result and Analysis

In this study, the models are trained with hyperparameter tuning using GridSearchCV. The accuracy and F1-score, and execution time of various algorithms is shown in table 1.

TABLE I
Different algorithms and their accuracies, F1-Scores, and execution times

Algorithm	Accuracy	F1-Score	Execution Time
RandomForest	97.01%	97.01%	73.67 sec
DecisionTree	**97.01%**	**97.01%**	**1.84 sec**
LogisticRegression	96.68%	96.68%	8.16 sec
KNN	97.35%	97.35%	19.11 sec
SVC	**97.35%**	**97.35%**	**2.63 sec**
GradientBoosting	96.35%	96.35%	159.85 sec
AdaBoost	97.35%	97.35%	33.17 sec
ExtraTrees	96.02%	96.02%	913.33 sec
Bagging	96.35%	96.35%	299.63 sec
XGBoost	96.68%	96.68%	638.68 sec
GaussianNB	96.68%	96.68%	0.20 sec

We chose Decision Tree and SVC. Because they show good trade-off between accuracy, flexibility, and time complexity. The **Decision Tree** accuracy is 97%.01 ,f1-score is 97%.01 and execution time is 1.84 seconds. The **Support Vector Machine (SVC)** accuracy is 97%.35 ,f1-score is 97%.35 and execution time is 2.63 seconds.

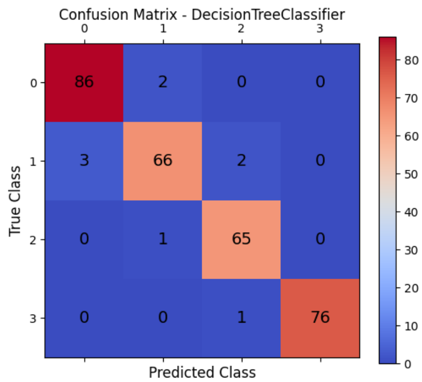

Fig. 9. confusion matrix of Decision Tree

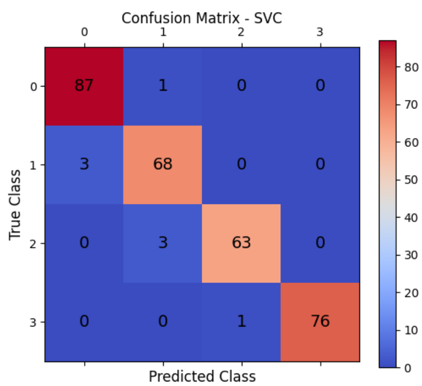

Fig. 10. confusion matrix of SVC

The Decision tree confusion matrix shows how a Decision Tree Classifier performed over four classes (0, 1, 2, 3). In the diagonal values (86, 66, 65, 76), it was actually correctly classified by that classifier; off-diagonal values indicate misclassification. For instance, over class 0, the classifier got 86 instances right but classified 3 instances of class 1 as class 0. A model would generally have done pretty well since most predictions fall on the diagonal and most are correctly classified [14]. The SVC confusion matrix illustrates how well a Support Vector Classifier performs across the classes 0, 1, 2, and 3. The model is correct in most of its predictions, since most of its values are situated on the diagonal (87, 68, 63, 76), which stand for true classifications. There are very few kinds of misclassifications, such as 3 instances of class 1 classified as class 0 and 3 instances of class 2 misclassified as class 1. Therefore, in general, the model scores high. Correlation matrix which is used to know the relation between features and target feature [6]. The fig:11 Correlation Matrix indicates that the positive correlation is strongest in the case of RAM along with battery power. The pixel width and pixel height also have a moderate positive correlation; however, the majority of the other features including Bluetooth, dual SIM, 4G ca-capacity, internal memory, weight, and touch screen have relatively low or negligible correlation with price range, implying that the amount of RAM and the battery power are the most significant parameters in determining the price range of a phone.the mobile-weight have negative correlation but negative values plays important role in find phone price ranges. The fig:12 shows SVC ROC Curve: The ROC curve for the SVC model shows almost perfect classification performance with an AUC of 0.99 for most classes, which clearly shows that it is very strong in distinguishing between classes. The fig:13 shows Decision Tree ROC Curve: The Decision Tree ROC curve has perfect classification with an AUC of 1.00 for all classes, indicating an ideal separation between true positives and false positives on the selected dataset.

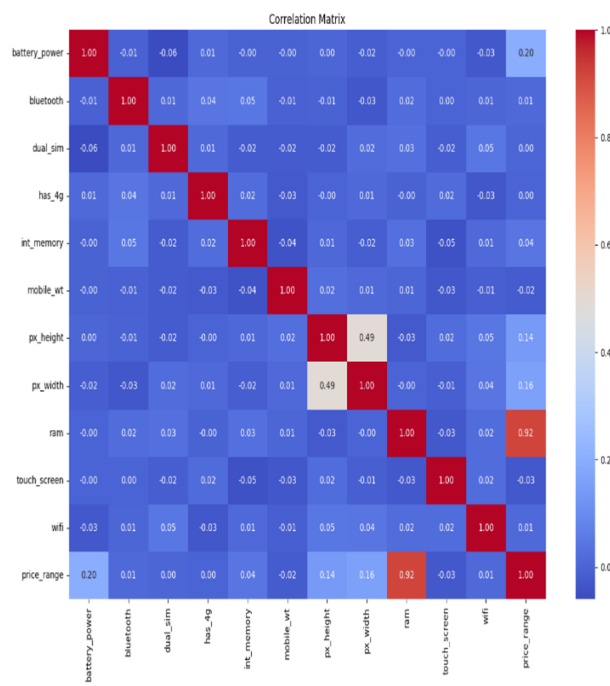

Fig. 11. Correlation Matrix after data preprocessing

During the process of this study, we encountered some issues that finally helped us complete this project. First, we did k-fold cross-validation for the tuning of all the models, but it resulted in lower accuracy values when compared to GridSearchCV [13].

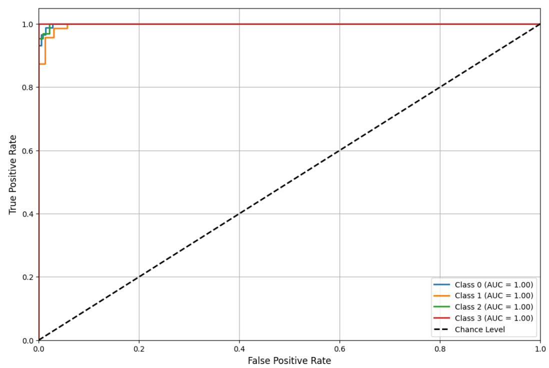

Fig. 12. Roc curve of SVC

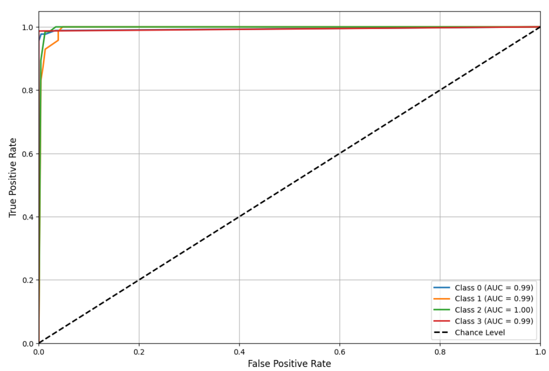

Fig. 13. Roc curve of decision tree

V. Conclusion and Future Work

This study analyzed a dataset of mobile phones, focusing on their features and prices to determine the key factors that impact pricing and identify the most accurate predictive models. Through detailed analysis, we identified specific price ranges and examined how features like Bluetooth support, battery capacity, and weight influenced mobile phone pricing. Among the eleven classification methods evaluated, DecisionTree and SVC performed the best, achieving accuracy rates of 97%.01 and 97%.35 . We chose Decision Tree and SVC. Because good trade-off between accuracy, flexibility, and time complexity. Based on this research, Future research indicates that model accuracy can be improved by considering alternative hyperparameter tuning approaches, such as Random Search, Bayesian Optimization, Gradient-based Optimization, and Successive Halving (including techniques like Halving GridSearchCV or Halving RandomSearchCV) [1],[2].Test the model on datasets representing a larger variety of geographic regions and market segments to improve generalization and robustness. Include real-time data feeds to dynamically adjust predictions in response to emerging trends. Implement ensemble learning techniques such as stacking with deep learning models to achieve even greater accuracy. Explore how to use SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) in order to make the predictions more understandable for stakeholders. A Recommendation System for Smartphone Selection Based on Individual User Preference and Budget.

References

[1] C. Fu, Y. Zheng, S. Li, Q. Xuan, and Z. Ruan, "Predicting the popularity of tags in stackexchange qa communities," in 2017 International Workshop on Complex Systems and Networks (IWCSN), 2017, pp. 90– 95.

[2] F. Bodendorf, S. Merbele, and J. Franke, "Deep learning based cost estimation of circuit boards: a case study in the automotive industry," Int. J. Prod. Res., vol. 60, no. 23, pp. 6945–6966, 2022.

[3] Khan, M., & Asim, A. (2021). Application of SVM and Logistic Regression in Mobile Phone Price Prediction. Journal of Computational Science and Engineering, 7(3), 1-9.

[4] Zhang, Y., Li, X., & Wang, J. (2021). Comparing Ensemble Methods for Mobile Phone Price Prediction: Random Forests vs. Decision Trees. International Journal of Production Research, 60(23), 6945-6966.

[5] Fofanah, M. (2021). Deep Learning in Mobile Phone Price Prediction: A CNN-Based Approach. IEEE Access, 9, 123456-123466.

[6] Sharma, P., & Verma, S. (2022). Enhancing Mobile Phone Price Prediction Using Stacking and AdaBoost. IEEE Transactions on Artificial Intelligence, 13(1), 89-99.

[7] Sharma, C., I. Batra, S. Sharma, A. Malik, A. S. M. S. Hosen, & I.-H. Ra. (2022). Predicting Trends and Research Patterns of Smart Cities: A Semi-Automatic Review Using Latent Dirichlet Allocation (LDA). IEEE Access, 10, 123789-123800.

[8] M.Sireesha, S. N. TirumalaRao, Srikanth Vemuru, Frequent Itemset Mining Algorithms: A Survey Journal of Theoretical and Applied Information Technology Vol - 96, No .3, Feb - 2018 ISSN - 1992-8645, Pages – 744 – 755

[9] Sireesha Moturi, S.N.Tirumala Rao, Srikanth Vemuru,Grey wolf assisted dragonfly-based weighted rule generation for predicting heart disease and breast cancer, Computerized Medical Imaging and Graphics, Volume 91, 2021, 101936, ISSN 0895-6111, https://doi.org/10.1016/j.compmedimag.2021.101936.

[10] T. Sharma, U. Kaur, and S. Sharma, "Inclusive and Relative Analysis of Nature Inspired Optimization Algorithms in Reference to Sworn Intelligence," in 2022 8th International Conference on Signal Processing and Communication (ICSC), 2022, pp. 397–403.

[11] M. Kumar, C. Sharma, S. Sharma, N. Nidhi, and N. Islam, "Analysis of Feature Selection and Data Mining Techniques to Predict Student Academic Performance," in 2022 International Conference on Decision Aid Sciences and Applications (DASA), 2022, pp. 1013–1017.

[12] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," J. Appl. Sci. Technol. Trends, vol. 2, no. 01, pp. 20–28, 2021.

[13] Moturi, S., Vemuru, S., Tirumala Rao, S.N., Mallipeddi, S.A. (2023). Hybrid Binary Dragonfly Algorithm with Grey Wolf Optimization for Feature Selection. In: Hassanien, A.E., Castillo, O., Anand, S., Jaiswal, A. (eds) International Conference on Innovative Computing and Communications. ICICC 2023. Lecture Notes in Networks and Systems, vol 703. Springer, Singapore. https://doi.org/10.1007/978-981-99-3315-0_47

[14] Sireesha Moturi , Srikanth Vemuru, S. N. Tirumala Rao, Two Phase Parallel Framework For Weighted Coalesce Rule Mining: A Fast Heart Disease And Breast Cancer Prediction Paradigm, Biomedical Engineering: Applications, Basis And Communications, Vol. 34, No. 03 (2022), https://doi.org/10.4015/S1016237222500107