BSIT-S24-009

03-135211-014 MUHAMMAD NASR

03-135211-008 EASHA FAHEEM

Used Mobile Phones Price Prediction for E-commerce Applications

In partial fulfilment of the requirements for the degree of

Bachelor of Science in Information Technology

Supervisor: Muhammad Faraz Manzoor

Department of Computer Sciences

Bahria University, Lahore Campus

January 2025

C e r t i f i c a t e

We accept the work contained in the report titled

"Used Mobile Phones Price Prediction for E-commerce Applications"

written by

MUHAMMAD NASR

EASHA FAHEEM

as a confirmation to the required standard for the partial fulfilment of the degree of Bachelor of Science in Information Technology.

Approved by:

Supervisor: Muhammad Faraz Manzoor

_____ (Signature)

January 05, 2025

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Enrolment	Name	Signature
03-135211-014	MUHAMMAD NASR	
03-135211-008	EASHA FAHEEM	

Date : January 05, 2025

Specially dedicated to
my beloved grandmother, mother and father
(MUHAMMAD NASR)
my beloved grandmother, mother and father
(EASHA FAHEEM)

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to our research supervisor, Mr Muhammad Faraz Manzoor for his invaluable advice, guidance and his enormous patience throughout the development of the research.

In addition, We would also like to express our gratitude to our loving parent and friends who had helped and given us encouragement.

MUHAMMAD NASR
EASHA FAHEEM

Used Mobile Phones Price Prediction for E-commerce Applications

ABSTRACT

The second-hand mobile phone market is riddled with challenges such as inconsistent pricing and inaccurate condition evaluations, which impact buyer trust and seller profitability. This project aims to address these challenges by developing a comprehensive system for predicting the condition and price of used mobile phones, tailored to the Pakistani e-commerce market. We used a dual approach with the deep learning and the machine learning methodologies. During the condition prediction, the VGG16 model was selected as the best model with high accuracy across front on, front off and back view image classifications into Good, Average and Bad categories. An ensemble of machine learning models including XGBoost was used for price prediction; namely, XGBoost outperformed others by the R^2 score and error metrics, showing its strength in dealing with complex pricing features. Next, these models were integrated into a user friendly Flutter based mobile application, through which users could upload images and provide specifications for the real time predictions. It is an application that bridges usability and technology, and gives a scalable solution to market inefficiencies. Future work includes expanding the dataset to include additional phone models and market data, increasing the number of image views for condition assessment, and adapting the system for international markets. This project lays a strong foundation for creating transparency and consistency in the used mobile phone marketplace.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

RAM	random access memory
AI	artificial intelligence
ML	machine learning
DT	decision tree
RF	random forest
SVM	support vector machine
KNN	k nearest neaighbours
XGB	xg boost
CNN	convolutional neural networks
ANN	artificial neural networks
DNN	deep neural networks
LSTM	long short term memory
BNN	bayesian neural networks
AE	autoencoders
BP	back propogation
LB-MCM	like block monte carlo method
iOS	iphone operating system
PTA	pakistan telecommunication authority
CSV	comma separated values
VGG	visual geometry group
Pkl	pickle, a type of file for saving non keras models

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

E-commerce platforms are the main source for selling and purchasing of new as well as old products nowadays. There are currently 3.8 billion [1] e-commerce users worldwide and the figure is expected to reach 4.5 billion by 2028 [2]. The main reason for the success of an e-commerce website is considered the accurate price bracket it provides for its products [3], same is the case for mobile phone industry. If the price bracket for a product is not set accordingly, i.e. the price for a high value product is set low it will not be profitable for the sellers while if the price of a low value product is set too high it might repel the buyers [4].

The price of a used mobile phone is estimated using different set of features such as the external condition of the phone, the launch year of the phone, series, company and many others [5]. The prices of the same phones with same exact features varies over different platforms creating inconsistency in the market [6].

The aim of this study is to predict accurate price of the used mobile phones based on the described features such as RAM, internal storage, model, external condition etc. employing different machine learning and deep learning algorithms. One of the main contribution of this study is the benchmark dataset developed for the prediction of the prices of used mobile phones based on the Pakistan market.

1.2 Problem Statements

The used mobile phone market in e-commerce faces challenges due to inconsistent and inaccurate pricing. Reliable price brackets are essential for both buyers and sellers, but issues like incorrect pricing affecting profitability and price variations across platforms create inefficiencies, requiring an effective solution.

Setting inappropriate price brackets for used mobile phones disrupts the balance between buyer affordability and seller profitability. If Overpriced, potential buyers get deterred which ultimately leads to fewer sales, and if underpriced, the seller makes few financial losses. High rates of these inaccuracies make e-commerce platforms relatively inefficient and a robust system to predict price brackets becomes even more crucial.

Inconsistent pricing for the same mobile phone model across different platforms creates confusion for buyers and affects seller competitiveness. These discrepancies, driven by varying valuation methods and market conditions, highlight the need for a standardized approach to ensure consistent and fair pricing across platforms.

1.3 Aims and Objectives

The objectives of the thesis include:

i) To develop a comprehensive benchmark dataset comprising real features that influence the prices of used mobile phones in the Pakistani market.

ii) To propose a robust framework leveraging various machine learning and deep learning models to predict the prices of used mobile phones.

iii) To enhance the authenticity of price brackets on E-commerce applications, particularly in the context of the Pakistani market.

1.4 Scope of Project

This project focuses on the development of a price prediction model for used mobile phones. The deliverables include:

- A comprehensive dataset sourced from leading Pakistani e-commerce platforms.
- A mobile application developed using Flutter to provide users with an interactive price prediction interface.
- Integration of various machine learning algorithms to deliver accurate price predictions, considering factors like device specifications, condition, and market trends.

CHAPTER 2

LITERATURE REVIEW

Over the past several years, machine learning has grown to efficiently improve the prediction of prices across multiple industries, including the rapidly developing eCommerce market. In today's volatile markets, accurate price predictions are critical because businesses want to reduce risks and maximize profits and use its resources properly. For businesses in the competitive eCommerce market, precise price prediction not only helps businesses improve their pricing strategies, but also facilitates their success by predicting market trends to make necessary adjustments to their strategies. In a similar way, machine learning models are used to predict prices for used mobile phones, and other products. While these models are of great help to the users on deciding the value of products based on features, they usually neglect the external condition of used products that greatly matters in determining their actual market price.

2.1 Used Mobile Phones Price Prediction

Helping consumers to make sound decisions when purchasing second hand devices with their features, used mobile phone price prediction is key to helping sellers sell their devices at the right prices. Precise price prediction models are necessary for businesses to be able to estimate the market trends and adjust products accordingly to make profits in a market with too many options. Finally, price prediction of used mobile phone enables efficient market and promotes the satisfaction of consumers in a highly dynamic industry.

2.1.1 Machine Learning

Due to its consistency and simplicity, mobile phone price prediction has been studied by several works using machine learning algorithms. Mobile phone price prediction has been achieved by linear regression and logistic regression. Maesya et al. [7] used linear regression to predict mobile phone prices and showed that it exhibits highly accurate results but just slightly outperforms with random forest as it is more capable to handle complex mobile price prediction. This suggests that supervised learning algorithms could accurately predict what prices mobile phones will go for. Like Subhiksha et al. [8], we also classified phones into price categories using logistic regression. Finally, their study emphasized the need for choosing proper features and the perfect accuracy was achieved by logistic regression and SVM. The study shows the effectiveness of regression models in predicting price, though more advance algorithms will provide greater marginal improvements. Feature reduction is an important factor, in decision for predictive models high prediction accuracy, and this study also revealed the crucial task of feature selection in maximizing the prediction accuracy of the predictive models.

Mobile phone price prediction has consistently employed Support Vector Machines (SVM) as a top-performing algorithm. In their work Subhiksha et al. [8] state that SVM, along with logistic regression, had the highest accuracy on classifying phones as phones at different prices. It also highlights the importance of the feature selection in obtaining the high prediction accuracy for the predictive models. This was supported by other author as in Hu [9] who found that SVM outperformed other classifiers. They then studied feature selection and its effect on the predictive performance of SVM, observing that features such as RAM, battery power and pixel resolution were important to the accuracy of the price prediction. On the basis of this, Jose et al. [10] joined the evidence and reported that SVM was the most efficient algorithm among numerous others including KNN and Decision Trees. The results of these studies across are extremely consistent suggesting that the use of SVM for predicting mobile phone prices is highly accurate and reliable.

Mobile phone price prediction has been extensively applied in the context of Decision Trees and Random Forest, both tree-based algorithms. For example, Maesya

et al. [7] found that Random Forest provides slightly better accuracy compared to liner regression in predicting mobile phone prices, while Çetin et al. [11] used several classification algorithms including Decision Trees and found that although SVM achieves highest accuracy, the tree based models like Decision Trees were effective with feature selection techniques.

It has been widely acknowledged that K-Nearest Neighbors (KNN) is a simple and powerful tool in a variety of tasks of price prediction in different domains. KNN was included in a comprehensive comparative study that also included Support Vector Machines (SVM) and other algorithms by Hu [9]. Hu found that SVM outperformed KNN, especially when it comes to using feature selection, but KNN still performed pretty well in terms of accuracy. K-Nearest Neighbors (KNN), along with Support Vector Machines (SVM), Decision Trees, and Random Forests, were also studied in depth by Jose et al. [10] in relation to price prediction. Specifically, they were interested in how well these algorithms performed under different feature sets and different data conditions. In addition, in a study to predict housing prices, Chen [12] compared KNN with linear regression. Chen found that KNN had a slight edge over linear regression from the results of accuracy, especially when there are non linear relationships in data.

2.1.2 Deep Learning and Neural Networks

Despite the considerable variety in structure and dimensions, deep learning models, especially in the area of using neural networks, applied to complex classification tasks, show greater performance. The research done by Zehtab-Salmasi et al. [13] was done by using a multimodal approach with CNN and InceptionV3. Similarly, Nasser et al. [14] developed an Artificial Neural Network (ANN) model to predict mobile phone range. Güvenç et al. [15] also contribute to this field by comparing Deep Neural Networks (DNN) to K-Nearest Neighbors (KNN).

Zaher et al. [16] also compare time series forecasting models using Long Short Term Memory (LSTM) neural networks and Support Vector Regression (SVR) to forecast phone prices in European markets. In addition, the study demonstrated that the LSTM models significantly bettered SVR models, in particular when the

dimensions were higher. Kalejahi et al. [17] compare the effectiveness of the Hedonic Regression and Artificial Neural Networks (ANN) in pricing mobile phone in the Iranian market. The results show that both methods have their advantages and the ANN model outperforms Hedonic Regression in terms of accuracy in capturing the non linear relationships between features that affect mobile phone pricing.

2.2 Car Price Prediction

As more and more people turn to the second hand market for cheaper prices, it has become necessary to be able to predict the prices of used cars and mobile phones more accurately. This market has expanded rapidly and it has become necessary to develop reliable pricing models that help buyers and sellers to make decisions. In this context, machine learning algorithms are key to the analysis of the factors that effect the value of these products. Car price prediction models then use these algorithms in order to ensure that the prices are fair and decision making is well informed, as is the case with mobile phone price prediction models. This predictive model of the transaction process, through the use of advanced data driven methodologies, provides a strong framework to increase transparency and trust within the transaction process as well as both parties within the marketplace.

2.2.1 Machine Learning

In price prediction task, KNearest Neighbors (KNN) is an easy to use but powerful algorithm to predict prices. Research conducted by Budiono et al. [18] essentially demonstrated that the KNN regression model performed adequately with small datasets, and made satisfactory predictions, with very good correlation between observed and predicted prices. This is simple and simple can be good in the price prediction tasks, when we focus on interpretation and usability.

As these are simple and successful, we can use linear regression, Lasso regression and Ridge regression to predict the used car. Chuyang Jin [19] particularly considered several regression techniques and concluded that random forest regression, the more advanced version, was most accurate. This shows regression models' ability to learn the fine details of used car pricing. Refining this, Ganesh et al. [20] assessed

Lasso Regression, Multiple Regression, and Regression Tree models, concluding that Multiple Regression was the model with the lowest error rate. Like Jin's study, theirs also highlight the robustness of regression models in the pricing prediction tasks. The results of Bharambe et al. [21] further demonstrate further the efficacy of regression techniques by reporting that Lasso Regression outperformed all of the other regression techniques they tested in terms of accuracy, and that accuracy is a strong function of the regression technique used in the car market.

Decision Trees and Random Forests are very popular and well regarded tree based algorithms because of their ability to work well with complex datasets and complex relationships between variables. In Jin [19] study, he found that random forest regression was the best performing model. It further illustrates the success of random forest in dealing with the complex datasets and making reliable prices predictions from used cars. Likewise, Ganesh et al. [20] used regression trees as well as other regression models for prediction of used car prices and were able to show, while multiple regression had the least amount of error, regression trees can also work, and work well especially for feature selection. Taken together, these studies strongly validate the robustness of tree based models such as Random Forest for modeling systemically complex interactions as powerful tools for accurate price predictions in the used car market.

2.2.2 Deep Learning

Accurate prediction of used car prices are of vital importance to the market, and deep learning algorithms, particularly artificial neural networks (ANN) are highly effective in dealing with complex and multi dimensional data. In Pillai [22], for predicting used car prices an artificial neural networks (ANN) based deep learning framework is proposed. One important finding of the study was that ANNs were very good at dealing with complex and multi dimensional data, far exceeding the performance of the more traditional models like linear regression, decision trees, random forest and gradient boosting. Furthermore Pudaruth et al. [23] build on the strengths of deep learning to look at the effectiveness of using Artificial Neural Networks (ANN) for price prediction of second hand cars in Mauritius. In their study, they compare the results produced by ANNs with those of both Support Vector Regression and Linear

Regression and k-Nearest Neighbour (kNN) and showed that although the results were good all the methods yield reasonable results, Support Vector Regression gives a slightly better one. The ANN model however was still capable of making quite a good prediction in terms of price forecast but for mid range priced cars. These results are consistent with the Indian used car market as Sareen et al. [24] further emphasize that the ANN model gives better R-squared value compared to Linear Regression and a Random Forest. This study mainly concentrates on the performance comparison and assessment of the traditional ML models like Random Forest, Linear Regression with a deep learning model i.e. Artificial Neural Network (ANN).

As Cui et al. [25] also used other models for car price prediction, they introduced an iterative framework, which combines these two models with a deep residual network. Cui et al's iterative framework not only outperformed traditional models but also demonstrated potential of combining multiple machine learning models to further improve prediction accuracy. As Huang et al. [26] and colleagues do in a similar way, it addresses the difficulty of accurately estimating used car prices with a model which combines Bayesian neural networks (BNNs) with latent factor extraction by autoencoders (AE). With this approach, data noise and overfitting, common problems in traditional models, are mitigated by taking account of uncertainty in the parameters of BNNs and by filtering out irrelevant data using latent factors. Results show superior prediction accuracy compared to standard methods with data processing by cloud computing and real time update. Shi et al. [27] also proposes a price evaluation model of second-hand cars based on an optimized Backpropagation (BP) neural network. This study then seeks to enhance accuracy of price prediction by choosing the best number of hidden neurons using the Like Block Monte Carlo Method (LB-MCM) with its higher prediction accuracy and robustness than traditional models.

CHAPTER 3

DESIGN AND METHODOLOGY

The methodology details the series of steps involved in the development of such models for predictions for condition and price of mobile phones. It offers a step by step roadmap from the collection of data, into pre-processing it to train and implement the model. Through breaking down the whole methodology into multiple processes, it guarantees that everything is structured properly, and all the processes contributes towards accurate predictions. Data preparation, model evaluation and predictive model deployment are all critical within each section of the methodology.

The methodology is divided into two key parts: the first focuses on condition prediction, the second on price. Data goes through pre processing, training and evaluation stage for each part, to make sure both the pictorial and specification dataset are properly utilized to build up robust models. Then, the models are evaluated in terms of performance before use for a real world application..

3.1 Data Collection

This is the most basic step in any machine learning project, because it is all about collecting initial (raw) information required to train and test our models. In this workflow, two types of data are essential for condition and price prediction: A pictorial data, which provides visual insights into the physical state of the mobile phones, and a specs dataset, that contains technical information about the devices. Later, these two data sources will be merged so that more accurate, comprehensive predictions can be made.

3.1.1 Data Collection Criteria

Extensive data was collected from multiple sources, including online marketplaces and physical markets, to ensure that our data set is diverse and we have a model that generalizes well on different conditions and use cases. Finally, we collected user generated data using Google Forms so that the model can learn to be more accurate with predicting prices by considering the often less reported conditions.

By combining feature data including brand, model and RAM with pictorial data representing the physical condition of the mobile phone from multiple sides, the model predicts better and more accurate used mobile phone market value than relying on feature or pictorial data alone.

Table 3.1 Data Collection Criteria

Criteria#	Criteria Name	Description
1.	Global Reach	The selected websites Amazon.com, Target.com have a global presence and widely used across different regions. These websites provide detailed information on used mobile phones, including technical specifications (e.g., RAM, storage, processor) and physical condition.
2.	Consumer Diversity	OLX.com, Bikjaa.pk caters to a wide spectrum of consumer segments, from budget-conscious buyers to those seeking premium products. This diversity helps capture varying consumer behaviors and price sensitivities, ensuring the model can generalize well across different market segments.
3.	User-Generated Data	Google Forms, OLX.com are the Platforms that allow for user-generated content. Such data captures nuanced details that may not be available on traditional retail websites. Images

		of the products is essential, Visual data enables the model to assess the physical condition of the devices.

3.1.2 Pictorial Data Collection

Separately, we collected a dataset, pictorial data of different views of the mobile phones. This dataset consists of images depicting three major sides of the phones which were collected across different platforms and provide key visual points of the mobile phones for the later condition prediction analysis. It was collected pictorial dataset to give a visual context for the technical specifications to holistically look at the mobile phones.

3.1.2.1 Sides Selection for Pictorial Data

Finally, after gathering the dataset we chose some sides, i.e. picked some parts, or subset of the collected data, that are the most important for the research objectives. In this step we selected the particular sides of the phone, front on, front off, and back which are a complete view of what makes a used mobile phone depends on its resale value. Combining these views, we have the holistic point of view, that ensures that enough factors are considered to affect the phone's condition, and as a result its resale value.

Table 3.2: Sides Selection Criteria

Criteria#	Criteria Name	Description
1.	Front On Condition	Image with screen on, checking for dead pixels, discoloration, or burn-in. Ensures touchscreen responsiveness and display quality without glare.
2.	Front Off Condition	Image with screen off, assessing scratches, cracks, or dents on glass and frame. Taken

		at an angle to highlight damage and minimize reflections.
3.	Back Condition	Image of the back, focusing on material condition, scratches, dents, and camera lens wear. Clear and well-lit to reveal all issues.

3.1.2.2 Classification of Pictorial Data

After selecting the relevant sides of the mobile phones, the pictorial dataset was classified into three categories: Good, Average, and Bad. The images were visually inspected to classify them into their condition based on the overall condition of each phone. Good phones had minimal to no visible wear and screens and body in good condition. Worn but still serviceable and somewhat visually acceptable average phones. Damage on bad phones included deep scratches, damage that showed the whole phone's innards, or anything that would really destroy the phone's aesthetic appeal or really make it unusable. For follow up analysis, this classification process was critical because it offered a more precise and nuanced view into how the physical condition of the phones affects their market value.

3.1.2.3 Validation of Collected Data

To validate the pictorial dataset, this process went through a thorough evaluation to make sure that the images corresponded to the mobile models and their conditions. The images were initially validated by two experts who independently validated the quality and condition of the phones that were shown. The evaluations involved checking any inconsistencies or discrepancy between the images and the reported conditions. To eliminate any bias or disagreement, a third expert was involved. The third person was an arbitrator, in the sense that he resolved any conflicts between the first two validators. For instance, suppose one expert rated a phone as clean, and the second rates the phone as fair; the third expert looked at the evidence, and gave a final, definitive rating so that we have a consistent dataset.

Cohen's Kappa statistic was used to quantify the agreement between the two primary validators. The inter rater reliability of this method was assessed, with the team examining the level of agreement between the two validators, beyond chance alone. The formula used was:

$$k = (p\ o - pe) / (1 - pe)$$

where:

po: Observed relative agreement among validators

pe: Chance agreement hypothetical probability

But the fact that we could build a statistical approach to find and correct any bias or inconsistency in the dataset made this so important to us. The team applied Cohen's Kappa to ensure that the validation was robust, the dataset is reliable and accurate. The validation of this process proved to be very meticulous and its importance for forming a reliable model for mobile phone condition assessment.

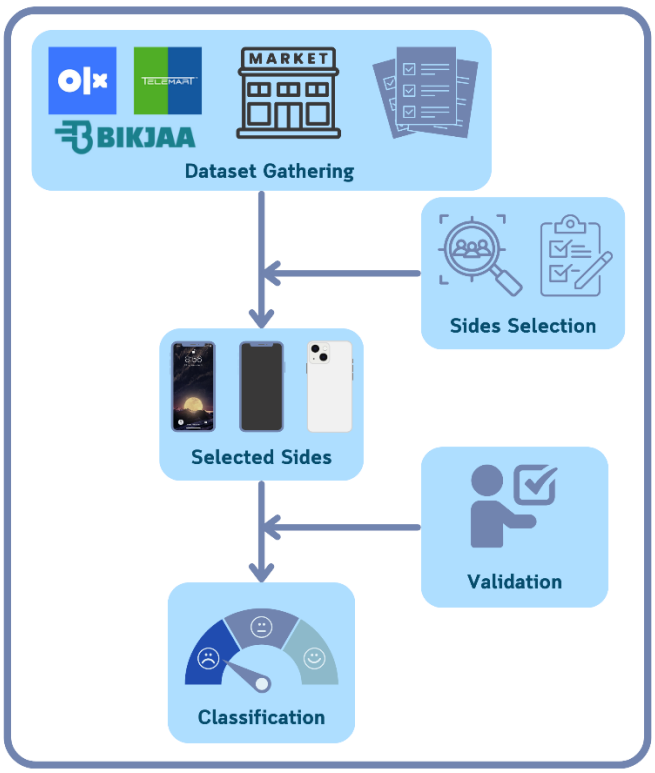

Figure 3.1: Pictorial Data Collection Methodology Diagram

3.1.3 Features Data Collection

In order to have an exhaustive dataset, we began with collecting detailed specifications of leading mobile phone manufacturers – Xiaomi, Infinix, Samsung, and Apple, etc. In this step, we determined and wrote down all possible features which exist on the models of these brands. The technical specifications for the data collected included battery capacity, processor type, operating system, etc. The building of this robust foundation was crucial for the dataset through this comprehensive gathering phase.

3.1.3.1 Selecting Relevant Features

However, not all attributes were needed for the analysis that was intended, so only the most important and meaningful features have been identified. The key attributes were selected to participate in this selection process. The features chosen include:

Table 3.3: Selected Features Description

Feature	**Description**
Brand	The manufacturer of the phone (e.g., Apple, Samsung, Huawei).
Model	Specific model of the phone (e.g., iPhone 11, Samsung Galaxy S20).
RAM	The amount of Random Access Memory in the device (e.g., 4GB, 8GB).
Storage	Internal storage capacity of the phone (e.g., 64GB, 128GB).
Processor	Type and speed of the processor (e.g., Snapdragon 865, Apple A13 Bionic).
Battery Capacity	Measured in milliampere-hours (mAh), indicating the battery's power capacity.
Operating System	The version of the operating system installed on the phone (e.g., iOS 14, Android 11).

External Condition	Condition of the phone's body, including any visible damage like scratches, dents, or cracks on the front and back of the phone.
5G Support	Indicates whether the phone supports 5G connectivity.
PTA Approved	Whether the phone is approved by the Pakistan Telecommunication Authority (PTA).
Year of Release	The year the phone was originally released to the market.
Original Price	The price of the phone when it was first released.
Current Price	The current market price of the phone.

3.1.3.2 Collecting the Data

Data was collected from online platforms such as OLX, Telemart, Bikjaa and from physical markets as well as from Google forms. Diverse and practical data on mobile phones were drawn from these sources, which reflect real world market trends and user preference. The dataset was comprehensive and representative by this design.

3.1.3.3 Refining the Data with Selected Features

The dataset is collected and refined in the step of Refining the Dataset with Selected Features after the dataset collected. The data was then filtered using the identified key features, removing any data that wasn't relevant. This made a concise focused dataset without any unnecessary or redundant data.

3.1.3.4 Compiling the Data

The merged validation and classification datasets were finally merged into an integrated, standardized dataset for analysis. In particular, the data was organized as a CSV (Comma Separated Value) file, which can be used by several statistical tools and software used in data analysis. The feature and pictorial data could be seamlessly

integrated into the CSV format and the relationships between mobile phone features, visual conditions and their market performance explored.

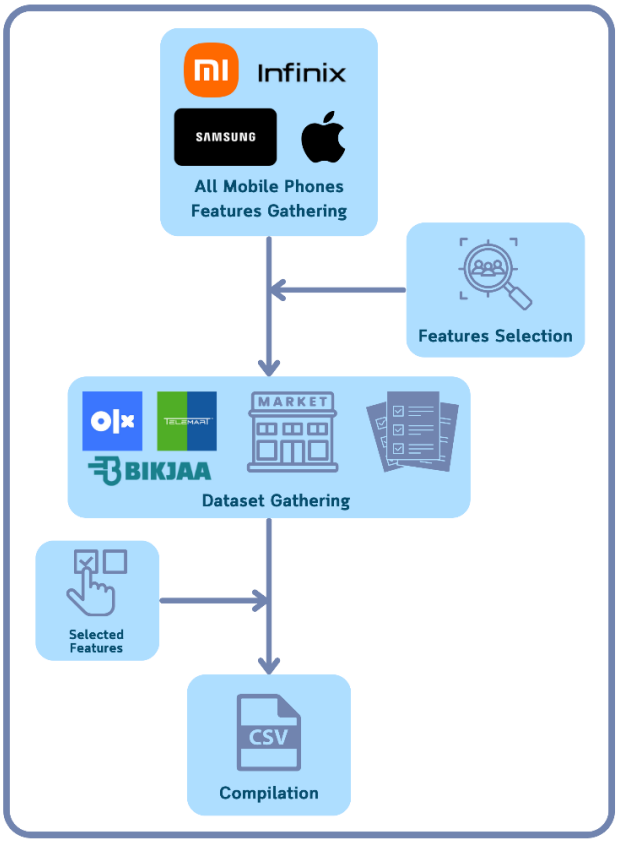

Figure 3.2: Features Data Collection Methodology Diagram

3.2 Preprocessing Pictorial Data

The collected data must go through pre processing (to be clean, consistent and ready for analysis) before we start diving into model training. Extract features to highlight some things of interest about the images. Then the dataset is split to training and testing subset for model's development. These sub processes are all important for preparing the data for prediction of the condition.

3.2.1 Feature Extraction

It extracts features of pictorial data. In this case, techniques such as CNN (Convolutional Neural Networks), and other image based methods were used so that meaningful data can be gathered for prediction.

3.2.2 Splitting Data

A prediction model is built by splitting the dataset in training and testing sets. The training set consists of 326 pictures for each side, similarly the test set consists of 80 pictures.

3.3 Model Training & Testing (Pictorial Data)

Once the data is preprocessed, we move onto training and testing a series of machine learning models specifically chose to predict the condition of mobile phones. First, the model is trained on a part of the data in order to learn what patterns to expect in the features. The model is trained once (or several times), where it is first tested on unseen data to see how good of a job it did generalizing the properties it learned to new cases. Finally, the model is assessed in terms of evaluation metrics to evaluate its performance.

3.3.1 Training the Model

Extracted features of the pictorial data are provided to deep learning and transfer learning models which are trained to predict the condition of the phone (Good, Average, Bad).

3.3.2 Testing the Model

We test the trained models on testing data, and evaluate their performance.

3.3.3 Evaluation

The performances of the condition prediction models are evaluated by metrics [28] including accuracy, precision, recall, F1 score and loss.

3.4 Condition Prediction

The condition of the mobile phone is predicted based on the best performing model trained. The results provided by the prediction process are in predefined condition categories Good, Average, Bad.

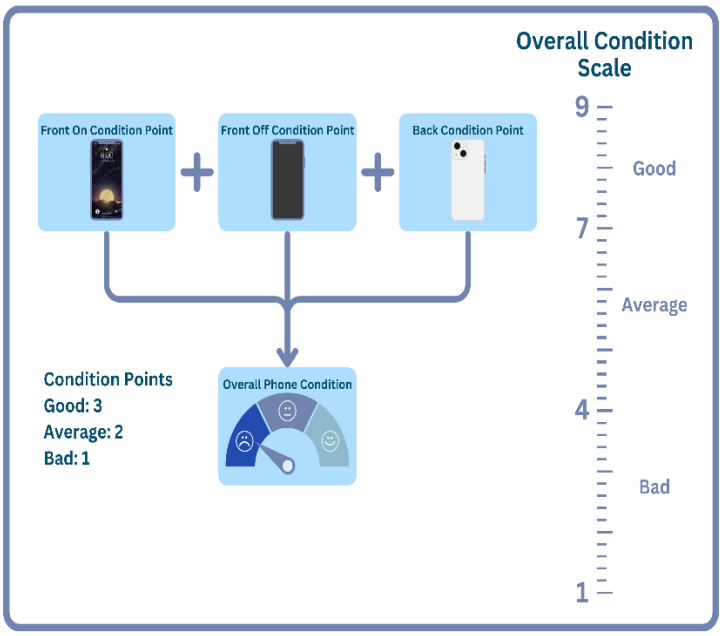

Figure 3.3: Condition Prediction Process Diagram

3.5 Preprocessing Features Data

Pre processing is an imperative stage before predicting price of an mobile phone just like the condition prediction phase. Here, we prepare the merged dataset (combining predicted condition and mobile phone specifications) by encoding, feature extraction, and data splitting. This means they are all ready to be used by the machine learning

algorithm itself. Good pre processing not only enhances performance of the model, but also limits the chances of any error during the prediction.

3.6 K-Fold Cross Validation

K FOLD Cross Validation is a way in model development to ensure your model is robust and generalizable. Splitting the dataset into several subsets (folds) and repeatedly training and testing the model on multiple different combinations of these subsets is what it's about. Using this technique we can reduce the risk of overfitting and make sure the model does well with unseen data. We validate the model across multiple folds to ensure that the price prediction model is reliable enough to handle such real world data variability.

3.7 Model Training & Testing (Features Data)

In the next phase, we will pre process and validate the data and then train the machine learning model to predict the Mobile Phone price depending on each specification and condition.

3.7.1 Training the Model

To train the model, we fed it input data so it can learn what relationship there are with the features and the target variable, price in this case.

3.7.2 Testing the Model

As with the condition prediction phase, the model is tested on unseen data and evaluation metrics will be used to measure how well the model predicts actual prices.

3.7.3 Evaluation

Just as with the condition prediction the price prediction model is evaluated in terms of metrics [29] like Mean Absolute Error (MAE), Mean Squared Error (MSE), or R squared, to measure the model's accuracy.

3.8 Price Prediction

Finally, the test data is used to predict the price of the mobile phone in the best performing model.

3.9 Implementation

Finally, the price prediction model is implemented. This model can be deployed and is able to take in new mobile phone data (images and specifications) as input and provide a predicted condition and price as output. All this is made possible by machine learning and data preprocessing to make condition and price prediction for mobile phones based on pictorial and specification data.

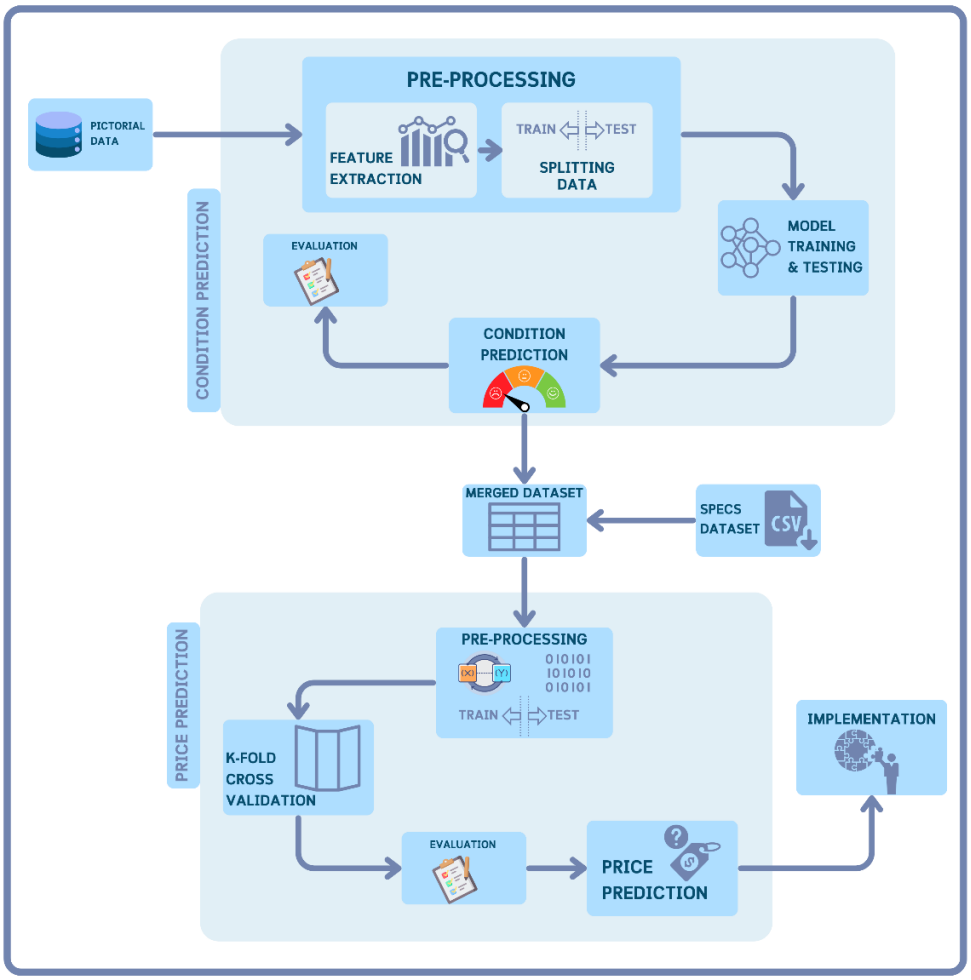

Figure 3.4: Complete Methodology Diagram

3.10 Mobile Application Design

We developed a mobile application using Flutter for this demonstration on the final implementation. Where we prompt the user to take pictures of their phone, then predict its condition. The user of the condition prediction can also predict the price of the phone he has filled a form regarding the specs of the phone and on the basis of their phone specs and condition our model predict the phone's price. The design consists of following four screens:

- Splash Screen
- Home Screen
- Condition Prediction Screen
- Price Prediction Screen

3.10.1 Splash Screen

In the Splash screen the company name “PhoneLo” is shown to grab the user’s attention and make the application more interactive.

Figure 3.5: Splash Screen

3.10.2 Home Screen

In the Home screen on the left corner the company name is displayed. Below that there is a carousal with three slides are designed so the main page can be interactive and the user know what type of features the mobile application provides. In the center there are two buttons named as “Conditon” and “Price” by clicking each one it takes the user to that specific page from where either the user can go to the condition prediction page or to the price prediction page. After that the design also shows the accuracy of our model. Furthermore we have developed a Bottom Navigation bar which also takes the user on the same pages as the above buttons to predict condition or price of their mobile phone.

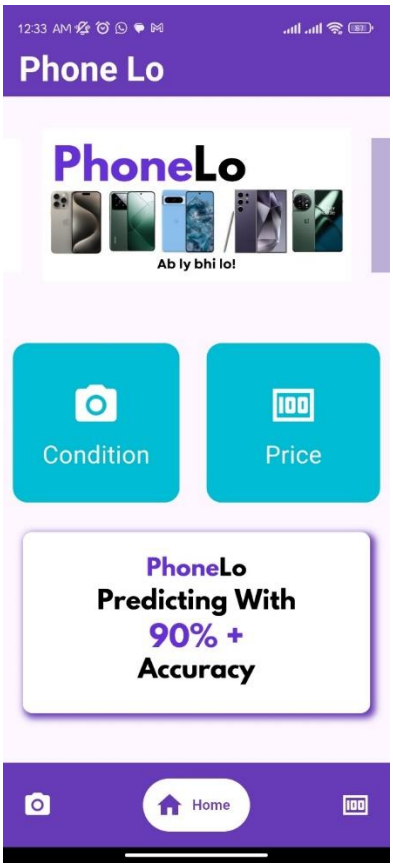

Figure 3.6: Home Screen

3.10.3 Condition Prediction Screen

In Condition Prediction screen firstly there is a warning mentioned that to get the best results of the condition of the mobile phone do remove the cover and the protector of the phone. After that there are two options on how to get the pictures either the user can have it from the gallery or can take the picture at that moment. The page also provides a button for the price prediction page only when all three pictures are uploaded.

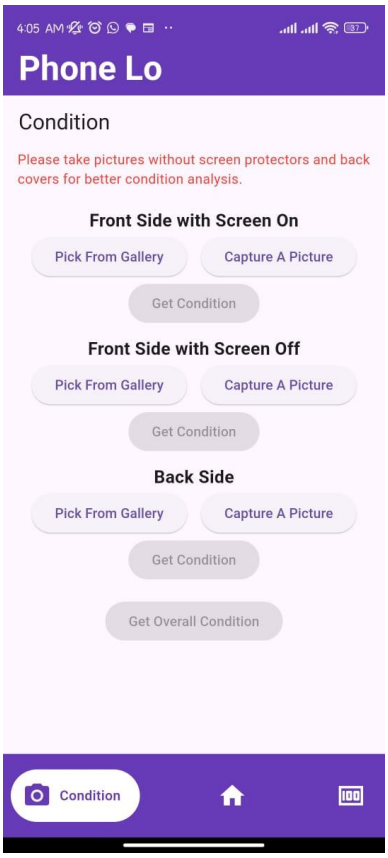

Figure 3.7: Condition Prediction Screen

3.10.4 Price Prediction Screen

In the price prediction screen we have developed a form in which we gather the specifications of the model of the mobile. The specifications includes Phone Model, Phone Company, RAM, Internal Storage, Operating System, Chipset, 5G, PTA Approved, Battery Capacity, Launched Year and the Original Price of the phone after all the information there is a button to predict the price of phone.

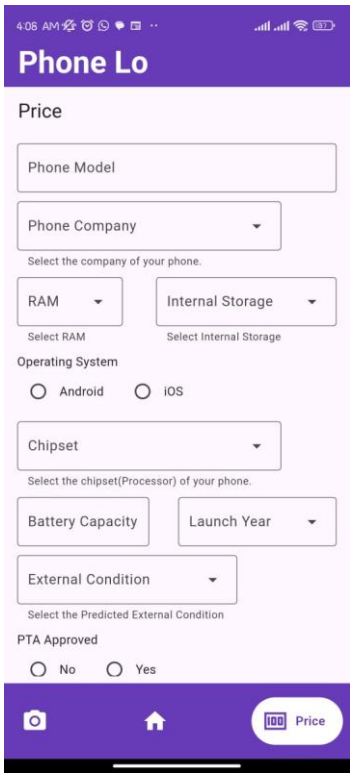

Figure 3.8: Price Prediction Screen

CHAPTER 4

DATA AND EXPERIMENTS and IMPLMENTATION

The implementation phase includes three phases Condition prediction Model Implementation, Price prediction Model Implementation, and the Flutter application development. Each component plays a vital role in delivering a seamless and accurate solution for evaluating second-hand mobile phones.

The deep learning techniques are used by the condition prediction model to classify phones based on the physical state of the phone using image data. The model classifies carefully using visual features via Convolutional Neural Networks (CNN), VGG and Inception V3 architectures.

Price prediction model uses a range of machine learning algorithms including decision trees, random forest, K nearest neighbors, XGBoost and AdaBoost to predict resale value of phones based on the attributes of the brand, model and condition.

Finally, these trained models are integrated into a user friendly mobile application using the Flutter implementation. The app provides a practical interface for end users with real time predictions of price and condition. Through this multi phase implementation, a robust and comprehensive solution that marries the power of machine learning with an intuitive mobile experience was possible.

4.1 Condition Prediction Phase

In this phase, we predicted the condition of second hand mobile phones using image based classification. In this phase of the work, the phone images were processed and

analyzed to accurately classify phones into different condition categories, thereby yielding reliable and consistent assessments. The model was trained using advanced deep learning techniques and image data augmentation to increase accuracy and robustness to a wide variety of inputs. Through this automated approach, the condition evaluation process was simplified to achieve objective results for the first-hand mobile phone market, and also second-hand mobile phone market (buyers and sellers).

4.1.1 Dataset

Image data stored in a structured format form the dataset. Inputs for training and testing the deep learning models are these images. The dataset consists of three image classes (Back side, Front-On side, and Front-Off side) pertaining to the condition prediction task and therefore needs high quality labeled images to deliver accurate model predictions..

4.1.2 Image Data Augmentation

To improve model generalization and prevent overfitting, image data augmentation techniques were applied which included:

- Rotation: Rotating images to different angles.
- Flipping: Horizontally or vertically flipping images.
- Zooming: Zooming into image areas to highlight details.
- Shifting: Translating images across horizontal and vertical planes.

Image augmentation increased the diversity of the training dataset, allowing the model to handle variations in real-world scenarios.

4.1.3 Models Applied

CNN (Convolutional Neural Network) [30] a fundamental model known for its efficiency in extracting features from image. CNNs process images through convolutional layers, pooling layers, and fully connected layers to generate predictions.

VGG 16 (Visual Geometry Group Network) [31] a deeper CNN architecture that uses small convolutional filters to capture fine-grained image features, enhancing accuracy in complex visual recognition tasks.

Inception V3 [32] is a sophisticated CNN architecture designed to optimize both accuracy and efficiency. It employs inception modules that use multiple convolutional filters of different sizes within the same layer, enabling the network to capture features at various scales.

4.1.4 Evaluation

Model evaluation was conducted to determine the performance of each applied model using key metrics:

- Accuracy: Measures the proportion of correctly predicted images.
- Loss: Computes the error between the predicted and actual outcomes, indicating how well the model is learning.
- Precision: Measures the proportion of correctly predicted positive cases among all cases predicted as positive.
- Recall: Measures the proportion of correctly predicted positive cases among all actual positive cases.
- F1-Score: Represents the harmonic mean of precision and recall, providing a balanced measure when there is an uneven class distribution.

These metrics helped in comparing the models' performance and selecting the most reliable one for deployment.

4.1.5 Best Model Selection

The best performing model was identified based on the evaluation results based on the accuracy and the loss. It was found that this model was most appropriate for prediction of real world condition. Further factors such as Precision, Recall and F1 Score were also considered.

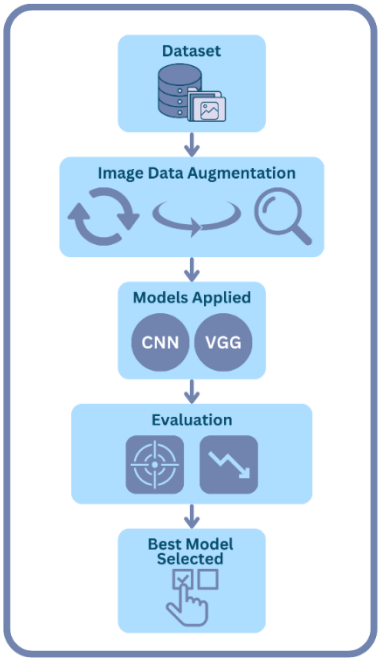

Figure 4.1: Condition Prediction Implementation

4.2 Price Prediction

The implementation was aimed at creating a strong price prediction system based on machine learning techniques. First, an acquisition of a broad CSV dataset was made, and then this data was thoroughly preprocessed to train this data. To tackle the complexities of the prediction task, a number of algorithms were used and their performance was hyperparameter tuned. Key Performance Metrics were used to evaluate the models and the best performing solution was chosen for deployment as its accuracy and reliability in real world scenarios.

4.2.1 Dataset

Our project started by getting a CSV dataset with the needed features and target variables to predict the price. The machine learning pipeline was fed by this dataset as the foundational input.

Following are some graphical and numerical representations of features involved in the dataset.

Table 4.1: Dataset Collection Sources

Sr.	Source	Instances
1	OLX.com	430
2	Bikjaa.pk	220
3	Google Forms	350

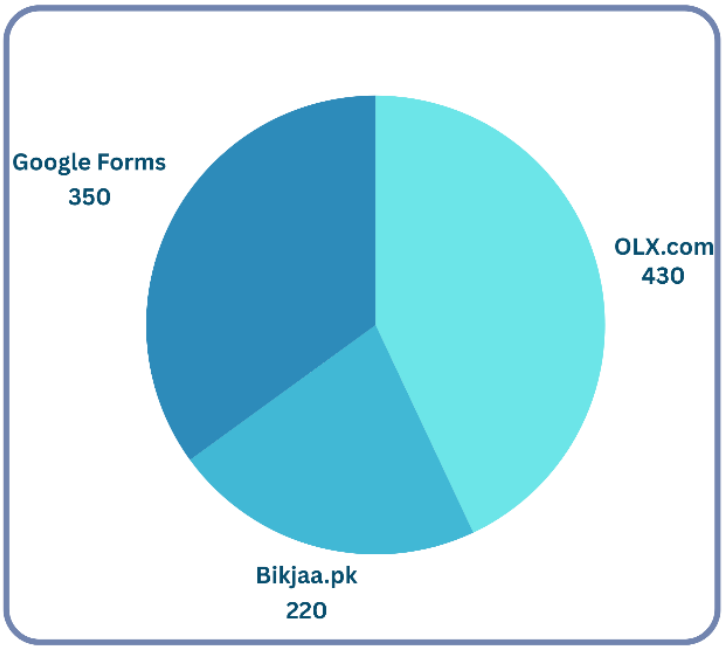

Figure 4.2: Graphical Representation of Dataset Collection Sources

Table 4.2: Number of Instances of Phone Companies

Sr.	Source	Instances
1	Samsung	224
2	Apple	184
3	Vivo	138
4	Oneplus	98
5	Google	83
6	Xiaomi	76
7	Infinix	45
8	Oppo	32
9	Tecno	31
10	LG	25

11	Huawei	21
12	Poco	20
13	Realme	15

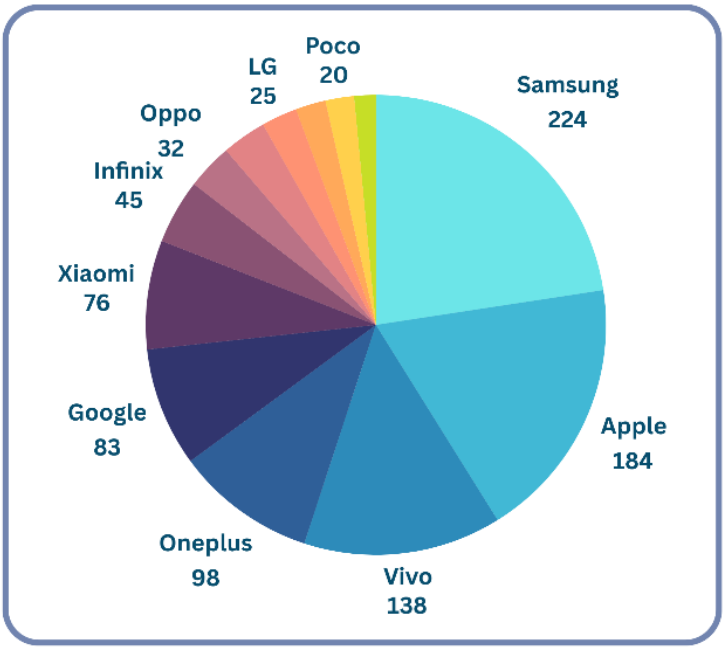

Figure 4.3: Graphical Representation of Number of Instances per Phone Company

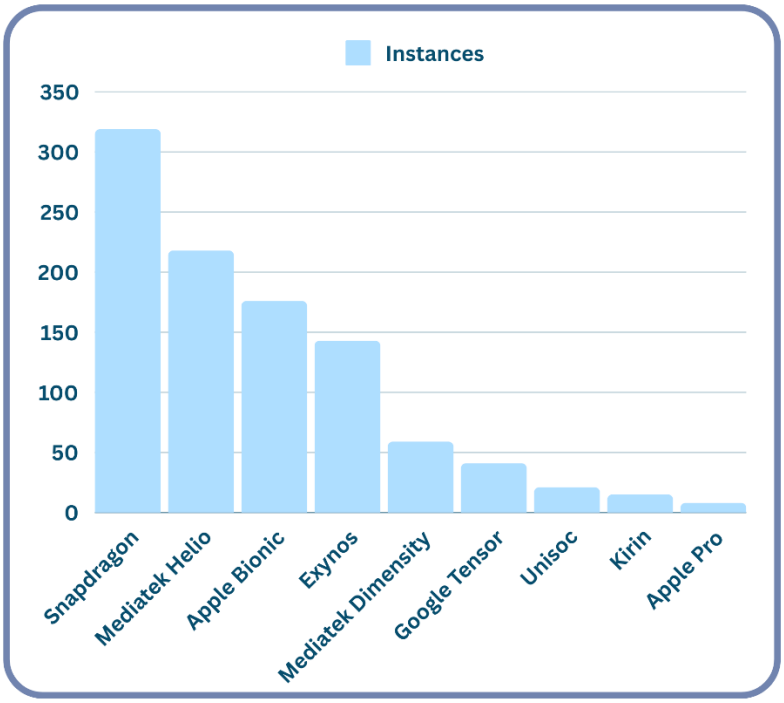

Figure 4.4: Graphical Representation of Number of Instances per Processor

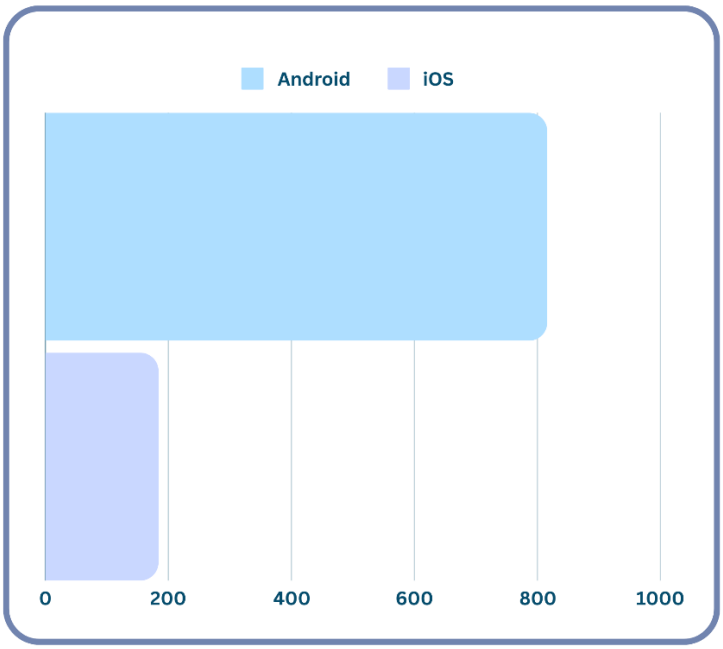

Figure 4.5: Graphical Representation of Number of Instances per Operating System

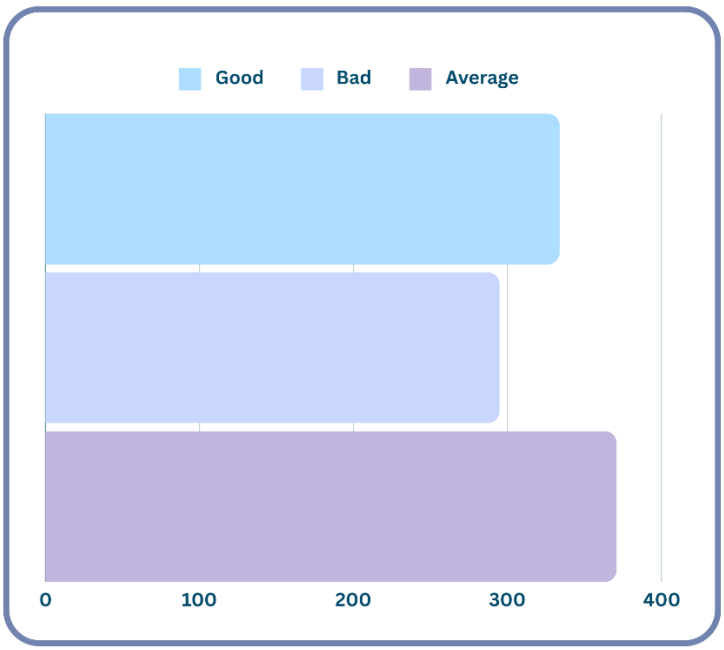

Figure 4.6: Graphical Representation of Number of Instances per External Condition

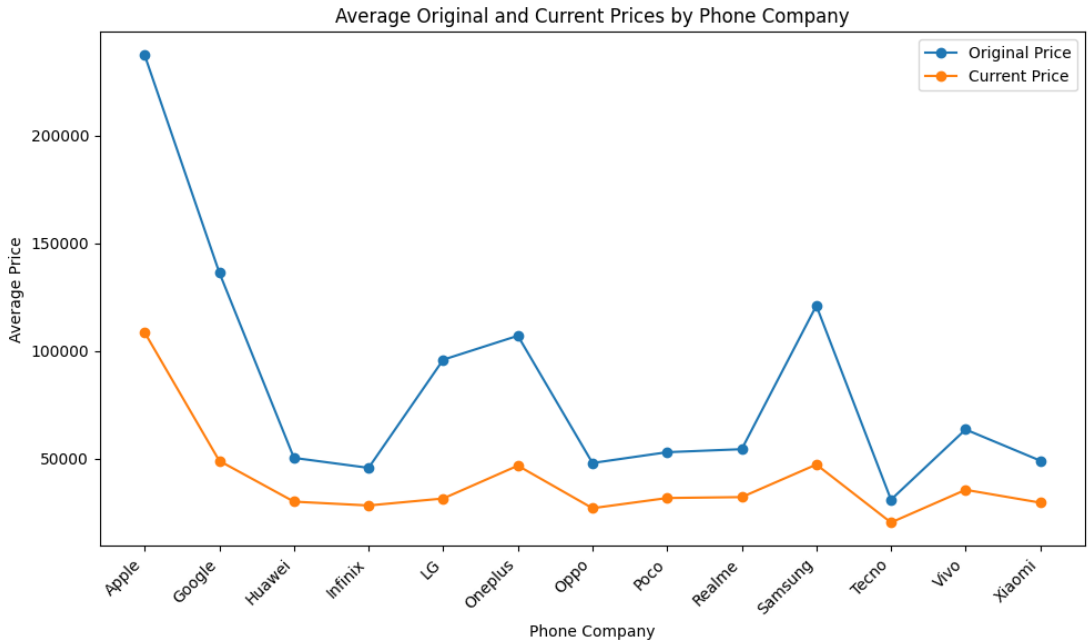

Figure 4.7: Graphical Representation of Average Current and Original Prices by Phone Company

4.2.2 Data Preprocessing

Several preprocessing steps were applied to ensure the dataset's suitability for model training:

- Data Cleaning: Missing values were addressed by imputing or removing incomplete records.
- Variable Identification: Independent and dependent variables were identified to clearly define input features and the target variable.
- Feature Encoding: Categorical variables were converted into numerical representations using encoding techniques.
- Data Splitting: The dataset was divided into training and testing sets to evaluate model performance on unseen data.

These steps ensured the dataset was properly prepared for training while maintaining data integrity and clarity.

4.2.3 Hyperparameter Tuning

To optimize model performance, hyperparameter tuning was conducted using Grid Search and Random Search. These techniques systematically adjusted key parameters, identifying the best configuration for each algorithm, enhancing accuracy and efficiency.

4.2.4 Machine Learning Models Applied

Various machine learning models were employed to predict prices:

- Decision Tree (DT): Valued for its simplicity and interpretability [33] .
- Random Forest (RF): An ensemble model that reduced overfitting and improved prediction accuracy [34] .
- K-Nearest Neighbors (KNN): A distance-based model relying on neighboring data points for predictions [35] .
- XGBoost (XGB): A powerful gradient boosting model known for its speed and accuracy [36] .
- AdaBoost (ADA): A boosting algorithm combining weak learners into a strong predictive model [37] .

4.2.5 Model Evaluation

Each model was evaluated using the following metrics [29] :

- R^2 (Coefficient of Determination): Measured how well the model explained variance in the target variable.
- MSE (Mean Squared Error): Calculated the average squared difference between predicted and actual values.
- MAE (Mean Absolute Error): Assessed the average absolute difference between predictions and actual results.

4.2.6 Best Model Selection

We chose the best performing model to based on evaluation metrics like R2 score, Mean Absolute Error and Mean Squared Error. It was selected because it had the highest accuracy and lowest error, and it was the most suitable for real world price prediction scenarios.

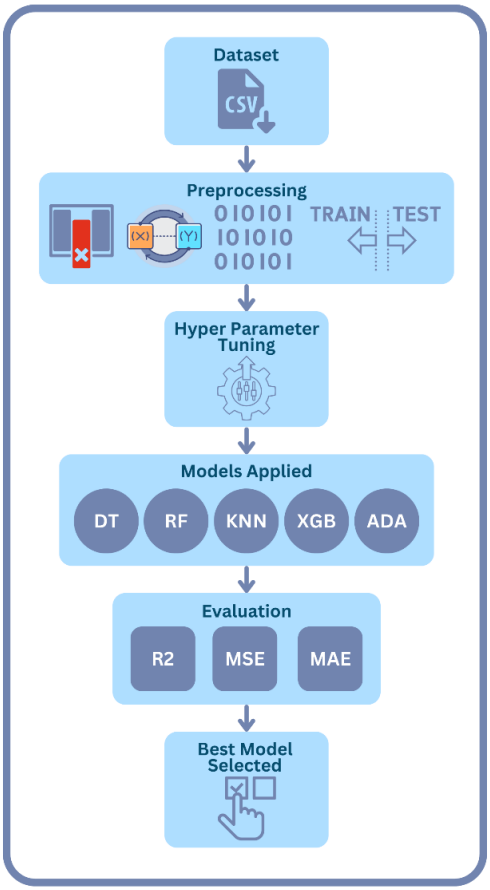

Figure 4.8: Price Prediction Implementation

4.3 Mobile Application Implementation

Flutter is used to build the mobile app so that users can have a finger tale solution to predict the condition and price of their phones. The app uses image recognition and machine learning so that users can take pictures of their phones to get a condition assessment. Besides, the users can input their phone's specs to get a complete price estimate. The app has an intuitive design with four key screens — Splash Screen, Home Screen, Condition Prediction Screen and Price Prediction Screen — and

advanced technology combined with practical functionality. This solution makes for simple phone evaluation and allows it to be accessible and efficient for your users.

4.3.1 Home Screen

The Home Screen serves as the central hub of the application, featuring two prominently displayed buttons that guide users to key functionalities: phone condition prediction and price estimation. The users are given a page to tap the 'Predict Condition' button that takes them to a page where they can either upload or capture the image of their phone for the condition analysis. This 'Predict Price' button takes the user to a form where the user can enter his phone's specifications (eg: Brand, Model, RAM and Storage) and estimate a price. This simple navigation makes the core features easy to find for users, giving them a nice, simple and efficient experience.

4.3.2 Condition Screen

Users can upload an image or capture the image of their phone on the Condition Prediction page to analyze its condition. Users can submit front side with screen on photo, front side with screen off photo, and back side photo to get a thorough condition assessment. In addition to having a clear page for user instructions for screen protectors and back covers to pull off for the best accuracy. From the app you can select or take images by pressing the "Pick From Gallery" or "Capture A Picture" buttons. Users first need to upload all images, and then tap the "Get Condition" button, to get an overall condition o their phone. The streamlined process followed ensures interaction in condition evaluation.

4.3.3 Price Screen

Users can input detailed information of their phone on the Price Prediction page to estimate its market value. Users have to enter Phone Model, Phone Company, RAM, Internal Storage, Operating System (Android, iOS) Chipset, Battery Capacity, and Launch Year. Features such as PTA Approval and 5G Support can be given as "Yes" or "No" in addition to the base version. It provides a structured form to collect all critical specifications which makes it possible to accurately estimate price. The easy

to navigate layout teamed with clean input fields make for easy use of the page, and a smooth flow of user provided information to create a reasonable price prediction.

CHAPTER 5

RESULTS AND DISCUSSIONS

After the successful implementation of the condition prediction and price prediction models, the best performing models for both of the modules were evaluated and then selected for the implementation in the flutter application.

5.1 Condition Prediction

The condition prediction module included training 3 different models for each of the side and then analysing the result. The models included, CNN, VGG16, InceptionV3.

5.1.1 Front On

Following are the results compiled after successful implementation of the three models on the Front On dataset.

Table 5.1: Front On Models Results

Sr.	Model	Accuracy	Precision	Recall	F1
1.	CNN	0.39	0.47	0.38	0.42
2.	VGG16	0.58	0.55	0.63	0.59
3.	InceptionV3	0.49	0.52	0.61	0.56

As VGG16 clearly outperforms the other two models, therefore VGG16 model file is converted into keras file and used for mobile application implementation.

5.1.2 Front Off

Following are the results compiled after successful implementation of the three models on the Front Off dataset.

Table 5.2: Front Off Models Results

Sr.	Model	Accuracy	Precision	Recall	F1
1.	CNN	0.57	0.32	0.57	0.41
2.	VGG16	0.63	0.53	0.68	0.61
3.	InceptionV3	0.57	0.45	0.54	0.50

As VGG16 clearly outperforms the other two models, therefore VGG16 model file is converted into keras file and used for mobile application implementation.

5.1.3 Back

Following are the results compiled after successful implementation of the three models on the Back dataset.

Table 5.3: Back Models Results

Sr.	Model	Accuracy	Precision	Recall	F1
1.	CNN	0.52	0.39	0.45	0.42
2.	VGG16	0.60	0.55	0.60	0.57
3.	InceptionV3	0.58	0.40	0.36	0.38

As VGG16 clearly outperforms the other two models, therefore VGG16 model file is converted into keras file and used for mobile application implementation.

5.2 Price Prediction

The model for price prediction module was selected after applying and evaluating different machine learning and ensemble learning models. These models included:

- Decision Tree Regressor
- Random Forest Regressor
- KNN Regressor
- XG Boost Regressor
- Ada Bosst Regressor

After successful implementation of these regressor and utilizing the RandomSearch hyperparameter tuning technique, following results were compiled.

Table 5.4: Price Prediction Models Results

Sr.	Regressor	R2 Score	MSE	MAE
1.	Decision Tree	0.70	793462446	14616
2.	Random Forest	0.86	332979312	11145
3.	KNN	0.76	558807487	14691
4.	XG Boost	0.88	277441365	9774
5.	Ada Boost	0.77	527124255	15926

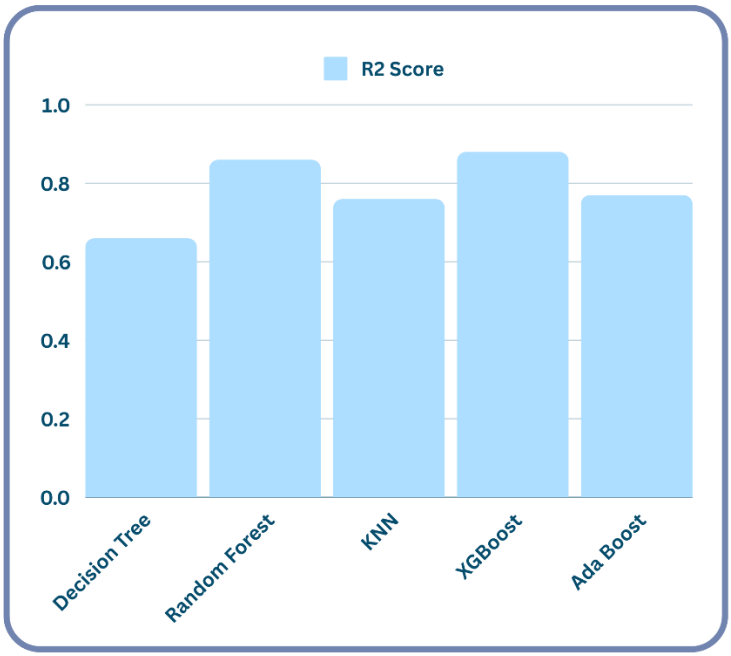

Figure 5.1: Graphical Representation of R2 Score for each Regressor

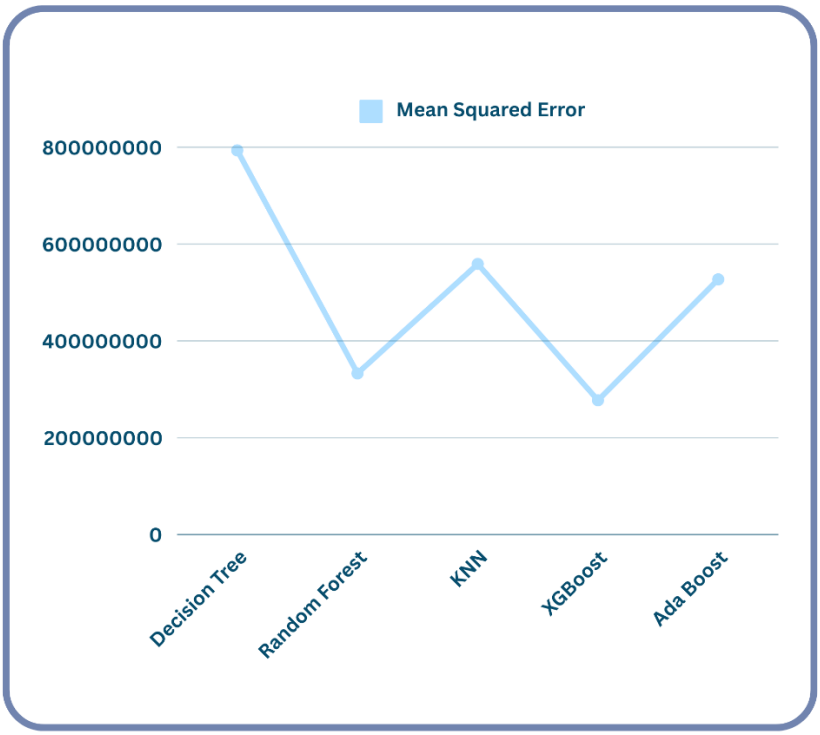

Figure 5.2: Graphical Representation of MSE for each Regressor

Figure 5.3: Graphical Representation of MAE for each Regressor

As XGBoost clearly outperforms the other regressors, so XGBoost regressor is converted into pkl file and then used for price prediction module in the flutter implementation.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

This study successfully developed a robust system for predicting the condition and price of used mobile phones. The system leveraged deep learning for condition classification and machine learning for price prediction, with an integration into a Flutter-based mobile application.

6.1 Conclusion

The VGG16 model consistently outperformed other models, achieving high accuracy for front-on, front-off, and back-side condition assessments. This highlights the effectiveness of deep learning in extracting and analyzing features from visual data.

The XGBoost model demonstrated superior performance, with the highest R^2 score and the lowest error metrics among the tested regressors. This underscores the reliability of ensemble learning methods for complex prediction tasks.

The Flutter application effectively combined these models into a user-friendly platform, enabling seamless condition and price predictions for end-users.

6.2 Recommendations and Future Work

Building on the success of our solution, we recommend the following steps for further improvement.

- Enhance the image preprocessing pipeline to handle variations in lighting, resolution, and angles, ensuring accurate condition predictions for all users.
- Explore incorporating additional views (e.g. top, bottom, and sides) to provide a more holistic evaluation of the phone's physical state.
- Increase the size and diversity of the dataset by collecting data from more sources, including international markets, to improve model robustness.
- Utilize user-generated content, such as images and descriptions, to capture a broader range of phone conditions and price influences.
- Customize the system for other regions by accounting for local market trends, consumer preferences, and economic factors, ensuring global applicability.
- Add real-time guidance within the app to help users capture high-quality images, improving prediction accuracy.
- Provide detailed insights and breakdowns of the price predictions, fostering user trust in the system.

REFERENCES

[1] B. Vuleta, "99firms." [Online]. Available: https://99firms.com/blog/ecommerce-statistics/#gref

[2] Statista, "Statista." [Online]. Available: https://www.statista.com/outlook/emo/ecommerce/worldwide

[3] E. Colla and P. Lapoule, "E-commerce: Exploring the critical success factors," *Int. J. Retail Distrib. Manag.*, vol. 40, no. 11, pp. 842–864, 2012, doi: 10.1108/09590551211267601.

[4] K. Kalyanam and T. S. Shively, "Estimating Irregular Pricing Effects: A Stochastic Spline Regression Approach," *J. Mark. Res.*, vol. 35, no. 1, pp. 16–29, 1998, doi: 10.1177/002224379803500104.

[5] F. Dye, "Easy Tech Junkie." [Online]. Available: https://www.easytechjunkie.com/what-factors-affect-a-smartphone-price.htm

[6] A. Hannak, G. Soeller, D. Lazer, A. Mislove, and C. Wilson, "Measuring price discrimination and steering on E-commerce web sites," *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*, pp. 305–318, 2014, doi: 10.1145/2663716.2663744.

[7] A. Maesya, Y. Yanfi, and Lukas, "Mobile Phone Price Prediction Based on Supervised Learning Algorithms," *Int. J. Appl. Eng. Technol.*, vol. 5, no. 1, pp. 41–44, 2023.

[8] S. Subhiksha, S. Thota, and J. Sangeetha, "Prediction of Phone Prices Using Machine Learning Techniques," *Adv. Intell. Syst. Comput.*, vol. 1079, pp. 781–789, 2020, doi: 10.1007/978-981-15-1097-7_65.

[9] N. Hu, "Classification of Mobile Phone Price Dataset Using Machine Learning Algorithms," *2022 3rd Int. Conf. Pattern Recognit. Mach. Learn. PRML 2022*, pp. 438–443, 2022, doi: 10.1109/PRML56267.2022.9882236.

[10] J. Jose, V. Raj, S. V. Seaban, and D. V. Jose, "Machine Learning Algorithms for Prediction of Mobile Phone Prices," *Lect. Notes Networks Syst.*, vol. 537 LNNS, no. February, pp. 81–89, 2023, doi: 10.1007/978-981-99-3010-4_7.

[11] M. Cetin and Y. Koc, "Mobile Phone Price Class Prediction Using Different Classification Algorithms with Feature Selection and Parameter Optimization,"

ISMSIT 2021 - 5th Int. Symp. Multidiscip. Stud. Innov. Technol. Proc., pp. 483–487, 2021, doi: 10.1109/ISMSIT52890.2021.9604550.

[12] Y. Chen, "Prediction of Different Types of Mobile Phone Prices based on Machine Learning Models," *Highlights Sci. Eng. Technol.*, vol. 92, pp. 275–279, 2024, doi: 10.54097/shgcew53.

[13] A. Zehtab-Salmasi, A. R. Feizi-Derakhshi, N. Nikzad-Khasmakhi, M. Asgari-Chenaghlu, and S. Nabipour, "Multimodal Price Prediction," *Ann. Data Sci.*, vol. 10, no. 3, pp. 619–635, 2023, doi: 10.1007/s40745-021-00326-z.

[14] I. M. Nasser and M. Al-Shawwa, "Developing Artificial Neural Network for Predicting Mobile Phone Price Range," *Int. J. Acad. Inf. Syst. Res.*, vol. 3, no. 2, pp. 1–6, 2019.

[15] E. Güvenç, G. Çetin, and H. Koçak, "Comparison of KNN and DNN Classifiers Performance in Predicting Mobile Phone Price Ranges," *Adv. Artif. Intell. Res.*, vol. 1, no. 1, pp. 19–28, 2021, [Online]. Available: www.dergipark.com/aair/

[16] H. Bakir, G. Chniti, and H. Zaher, "E-Commerce price forecasting using LSTM neural networks," *Int. J. Mach. Learn. Comput.*, vol. 8, no. 2, pp. 169–174, 2018, doi: 10.18178/ijmlc.2018.8.2.682.

[17] S. V. T. Kalejahi and D. M. Nazari, "Comparing mobile phone price estimators in the Iran Market: Hedonic Regression and Artificial Neural Network," *Bus. Manag. Strateg.*, vol. 3, no. 1, pp. 46–60, 2011, doi: 10.5296/bms.v3i1.1083.

[18] D. A. Budiono, K. S. Utomo, R. Kenny, J. Wibowo, and M. J. Wiradinata, "Used Car Price Prediction Model: A Machine Learning Approach," *Int. J. Comput. Inf. Syst. Peer Rev. J.*, vol. 05, no. 01, pp. 2745–9659, 2024, [Online]. Available: https://ijcis.net/index.php/ijcis/index

[19] C. Jin, "Price Prediction of Used Cars Using Machine Learning," *Proc. 2021 IEEE Int. Conf. Emerg. Sci. Inf. Technol. ICESIT 2021*, pp. 223–230, 2021, doi: 10.1109/ICESIT53460.2021.9696839.

[20] P. Venkatasubbu and M. Ganesh, "Used Cars Price Prediction using Supervised Learning Techniques," *Int. J. Eng. Adv. Technol.*, vol. 9, no. 1s3, pp. 216–223, 2019, doi: 10.35940/ijeat.a1042.1291s319.

[21] H. Dahiya, C. Aggarwal, S. Goyal, and M. Agarwal, "Used Car Price Prediction Using Machine Learning," *Int. J. Multidiscip. Adv. Sci. Res. Innov.*, vol. 1, no. 10, pp. 246–251, 2021, doi: 10.53633/ijmasri.2021.1.10.002.

[22] A. S. Pillai, "A Deep Learning Approach for Used Car Price Prediction," *J. Sci.*

Technol., vol. 3, no. 3, pp. 31–50, 2022.

[23] N. Walker, *Proceedings of the Second International Conference on Data Mining, Internet Computing, and Big Data, Reduit, Mauritius 2015*, no. June 2015. 2017.

[24] A. P. Singh and B. Sareen, "Comparison Between Machine Learning Models and Deep Learning Model for Prediction of Car Prices," *2023 7th Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2023*, pp. 1–6, 2023, doi: 10.1109/ICCUBEA58933.2023.10392182.

[25] B. Cui, Z. Ye, H. Zhao, Z. Renqing, L. Meng, and Y. Yang, "Used Car Price Prediction Based on the Iterative Framework of XGBoost+LightGBM," *Electron.*, vol. 11, no. 18, 2022, doi: 10.3390/electronics11182932.

[26] J. Huang *et al.*, "A Latent Factor-Based Bayesian Neural Networks Model in Cloud Platform for Used Car Price Prediction," *IEEE Trans. Eng. Manag.*, vol. 71, pp. 12487–12497, 2024, doi: 10.1109/TEM.2023.3270301.

[27] N. Sun, H. Bai, Y. Geng, and H. Shi, "Price evaluation model in second-hand car system based on BP neural network theory," *Proc. - 18th IEEE/ACIS Int. Conf. Softw. Eng. Artif. Intell. Netw. Parallel/Distributed Comput. SNPD 2017*, pp. 431–436, 2017, doi: 10.1109/SNPD.2017.8022758.

[28] Geeks for Geeks, "Evaluation Metrics For Classification Model in Python." [Online]. Available: https://www.geeksforgeeks.org/evaluation-metrics-for-classification-model-in-python/

[29] Geeks for Geeks, "Regression Metrics." [Online]. Available: https://www.geeksforgeeks.org/regression-metrics/

[30] Mohd Sanad Zaki Rizvi, "Image Classification Using CNN." [Online]. Available: https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/

[31] Geeks for Geeks, "VGG-16 | CNN model." [Online]. Available: https://www.geeksforgeeks.org/vgg-16-cnn-model/

[32] Keras, "InceptionV3." [Online]. Available: https://keras.io/api/applications/inceptionv3/

[33] Scikit-learn, "Decision Trees." [Online]. Available: https://scikit-learn.org/1.5/modules/tree.html

[34] Scikit-learn, "RandomForestRegressor," 2024, [Online]. Available: https://scikit-

learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[35] Scikit-learn, “KNeighborsRegressor,” 2024, [Online]. Available: https://scikit-learn.org/1.5/modules/generated/sklearn.neighbors.KNeighborsRegressor.html

[36] XGBoost, “Get Started with XGBoost.” [Online]. Available: https://xgboost.readthedocs.io/en/latest/get_started.html

[37] Scikit-learn, “AdaBoostRegressor.” [Online]. Available: https://scikit-learn.org/1.6/modules/generated/sklearn.ensemble.AdaBoostRegressor.html

APPENDICES

APPENDIX A: Codes

Flutter

Main.dart

```dart
import 'package:flutter/material.dart';
import 'package:phone_lo/splashScreen.dart';

void main() {
  runApp(const MaterialApp(
      debugShowCheckedModeBanner: false, home: SplashScreen()));
}
```

Homepage.dart

```dart
import 'package:flutter/material.dart';
import 'package:google_nav_bar/google_nav_bar.dart';
import 'package:phone_lo/condition.dart';
import 'package:phone_lo/home.dart';
import 'package:phone_lo/price.dart';

class HomeScreen extends StatefulWidget {
  const HomeScreen({super.key});
  @override
  State<HomeScreen> createState() => _HomeScreenState();
}
```

```dart
class _HomeScreenState extends State<HomeScreen> {
  int _selectedIndex = 1;
  void _navBottomBar(int index) {
    setState(() {
      _selectedIndex = index;
    });
  }

  final List<Widget> _pages = [
    const Condition(),
    UserHome(onNavigate: (index) => {}),
    const Price()
  ];
  @override
  Widget build(BuildContext context) {
    _pages[1] = UserHome(onNavigate: _navBottomBar);
    return Scaffold(
      appBar: AppBar(
        backgroundColor: Colors.deepPurple,
        title: const Text(
          'Phone Lo',
          textAlign: TextAlign.center,
          style: TextStyle(
              fontSize: 35, color: Colors.white, fontWeight: FontWeight.bold),
        ),
      ),
      body: _pages[_selectedIndex],
      bottomNavigationBar: Container(
        color: Colors.deepPurple,
        child: Padding(
          padding: const EdgeInsets.symmetric(horizontal: 8, vertical: 16),
          child: GNav(
              backgroundColor: Colors.deepPurple,
              color: Colors.white,
              gap: 8,
              padding: const EdgeInsets.all(12),
              activeColor: Colors.deepPurple,
              tabBackgroundColor: Colors.white,
              selectedIndex: _selectedIndex,
              onTabChange: _navBottomBar,
              tabs: const [
                GButton(
                  icon: Icons.camera_alt,
                  text: "Condition",
                  iconSize: 30,
                ),
                GButton(
                  icon: Icons.home,
                  text: "Home",
                  iconSize: 30,
                ),
                GButton(
                  icon: Icons.money,
                  text: "Price",
                  iconSize: 30,
                ),
              ]),
        ),
      ),
    );
  }
}

```

Home.dart

```dart
import 'package:carousel_slider/carousel_slider.dart';
import 'package:flutter/material.dart';

class UserHome extends StatelessWidget {
  final Function(int) onNavigate;
  final carouselItems = [
    Image.asset('assets/images/1.png'),
    Image.asset('assets/images/2.png'),
    Image.asset('assets/images/3.png'),
  ];
  UserHome({super.key, required this.onNavigate});
  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        CarouselSlider(
          options: CarouselOptions(
            autoPlay: true,
            viewportFraction: 0.8,
            enlargeCenterPage: true,
            autoPlayInterval: const Duration(seconds: 3),
          ),
          items: carouselItems,
        ),
        Row(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          children: [
            Container(
              decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(15),
                color: Colors.cyan,
              ),
              padding: const EdgeInsets.all(30),
              margin: const EdgeInsets.symmetric(horizontal: 10, vertical: 30),
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  IconButton(
                      iconSize: 50,
                      color: Colors.white,
                      onPressed: () => {onNavigate(0)},
                      icon: const Icon(Icons.camera_alt)),
                  const Text("Condition",
                      style: TextStyle(fontSize: 25, color: Colors.white)),
                ],
              ),
            ),

```

```
Container(
              decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(15),
                color: Colors.cyan,
              ),
              padding: const EdgeInsets.symmetric(horizontal: 55, vertical: 30),
              margin: const EdgeInsets.symmetric(horizontal: 10, vertical: 30),
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  IconButton(
                      iconSize: 50,
                      color: Colors.white,
                      onPressed: () => {onNavigate(2)},
                      icon: const Icon(Icons.money)),
                  const Text("Price",
                      style: TextStyle(fontSize: 25, color: Colors.white)),
                ],
              ),
            )
          ],
        ),
        Container(
          decoration: BoxDecoration(
            border: Border.all(
              width: 8.0,
              color: Colors.white,
            ),
            boxShadow: const [
              BoxShadow(
                color: Colors.deepPurple,
                offset: Offset(3.0, 3.0),
                spreadRadius: 2.0,
                blurRadius: 5.0,
              ),
            ],
            borderRadius: const BorderRadius.all(Radius.circular(10.0)),
          ),
          margin: const EdgeInsets.symmetric(horizontal: 20.0),
          child: Image.asset(
            "assets/images/accuracy.png",
          ),
        ),
      ],
    );
  }
}
```

Condition.dart

Back Prediction (Mobile App)

```dart
Future<void> _pickImageFromGalleryBack() async {
    final image = await _picker.pickImage(source: ImageSource.gallery);

    setState(() {
      if (image != null) {
        _backImage = File(image.path);
      }
    });
  }

  Future<void> _pickImageFromCameraBack() async {
    final image = await _picker.pickImage(source: ImageSource.camera);
    setState(() {
      if (image != null) {
        _backImage = File(image.path);
      }
    });
  }

  Future<void> _uploadImageBack() async {
    if (_backImage == null) return;

    setState(() {
      _loadingBack = true;
    });

    final uri = Uri.parse('http://192.168.107.85:5000/predictBack');
    final request = http.MultipartRequest('POST', uri);
    request.files
        .add(await http.MultipartFile.fromPath('file', _backImage!.path));

    final response = await request.send();
    if (response.statusCode == 200) {
      final responseData = await response.stream.bytesToString();
      final result = json.decode(responseData);
      String temp;
      setState(() {
        temp = result['predicted_class'].toString();
        switch (temp) {
          case '0':
            _predictionBack = 'Average';
            backPoints = 2;
            break;
          case '1':
            _predictionBack = 'Bad';
            backPoints = 1;
            break;
          case '2':
            _predictionBack = 'Good';
            backPoints = 3;
            break;
        }
      });
    } else {
      setState(() {
        _predictionBack = 'Error: ${response.reasonPhrase}';
      });
    }

    setState(() {
      _loadingBack = false;
    });
  }

```

Front On Prediction (Mobile App)

```dart
Future<void> _pickImageFromGalleryFrontOn() async {
    final image = await _picker.pickImage(source: ImageSource.gallery);

    setState(() {
      if (image != null) {
        _frontOnImage = File(image.path);
      }
    });
  }

  Future<void> _pickImageFromCameraFrontOn() async {
    final image = await _picker.pickImage(source: ImageSource.camera);
    setState(() {
      if (image != null) {
        _frontOnImage = File(image.path);
      }
    });
  }

  Future<void> _uploadImageFrontOn() async {
    if (_frontOnImage == null) return;

    setState(() {
      _loadingFrontOn = true;
    });

    final uri = Uri.parse('http://192.168.107.85:5000/predictFrontOn');
    final request = http.MultipartRequest('POST', uri);
    request.files
        .add(await http.MultipartFile.fromPath('file', _frontOnImage!.path));

    final response = await request.send();
    if (response.statusCode == 200) {
      final responseData = await response.stream.bytesToString();
      final result = json.decode(responseData);
      String temp;
      setState(() {
        temp = result['predicted_class'].toString();
        switch (temp) {
          case '0':
            _predictionFrontOn = 'Average';
            frontOnPoints = 2;
            break;
          case '1':
            _predictionFrontOn = 'Bad';
            frontOnPoints = 1;
            break;
          case '2':
            _predictionFrontOn = 'Good';
            frontOnPoints = 3;
            break;
        }
      });
    } else {
      setState(() {
        _predictionFrontOn = 'Error: ${response.reasonPhrase}';
      });
    }

    setState(() {
      _loadingFrontOn = false;
    });
  }
```

Front Off Predicition (Mobile App)

```dart
Future<void> _pickImageFromGalleryFrontOff() async {
    final image = await _picker.pickImage(source: ImageSource.gallery);

    setState(() {
      if (image != null) {
        _frontOffImage = File(image.path);
      }
    });
  }

  Future<void> _pickImageFromCameraFrontOff() async {
    final image = await _picker.pickImage(source: ImageSource.camera);
    setState(() {
      if (image != null) {
        _frontOffImage = File(image.path);
      }
    });
  }

  Future<void> _uploadImageFrontOff() async {
    if (_frontOffImage == null) return;

    setState(() {
      _loadingFrontOff = true;
    });

    final uri = Uri.parse('http://192.168.107.85:5000/predictFrontOff');
    final request = http.MultipartRequest('POST', uri);
    request.files
        .add(await http.MultipartFile.fromPath('file', _frontOffImage!.path));

    final response = await request.send();
    if (response.statusCode == 200) {
      final responseData = await response.stream.bytesToString();
      final result = json.decode(responseData);
      String temp;
      setState(() {
        temp = result['predicted_class'].toString();
        switch (temp) {
          case '0':
            _predictionFrontOff = 'Average';
            frontOffPoints = 2;
            break;
          case '1':
            _predictionFrontOff = 'Bad';
            frontOffPoints = 1;
            break;
          case '2':
            _predictionFrontOff = 'Good';
            frontOffPoints = 3;
            break;
        }
      });
    } else {
      setState(() {
        _predictionFrontOff = 'Error: ${response.reasonPhrase}';
      });
    }

    setState(() {
      _loadingFrontOff = false;
    });
  }
```

Overall Prediction (Mobile App)

```
Future<void> _getOverallCondition() async {
    setState(() {
      if (_predictionBack!.isNotEmpty &&
          _predictionFrontOff!.isNotEmpty &&
          _predictionFrontOn!.isNotEmpty) {
        overallPoints = backPoints + frontOnPoints + frontOffPoints;
        if (overallPoints >= 8) {
          _overallCondition = 'Good';
        } else if (overallPoints >= 5) {
          _overallCondition = 'Average';
        } else if (overallPoints < 5) {
          _overallCondition = 'Bad';
        }
      }
    });
  }

```

Price.dart

```
import 'dart:convert';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

class Price extends StatefulWidget {
  const Price({super.key});

  @override
  State<Price> createState() => _PriceState();
}
```

Price Prediction (Mobile App)

```dart
class _PriceState extends State<Price> {
  TextEditingController _controllerModel = TextEditingController();
  TextEditingController _controllerBattery = TextEditingController();
  TextEditingController _controllerOriginal = TextEditingController();
  String _selectedModel = "";
  int? _selectedCompany;
  int? _selectedRAM;
  int? _selectedStorage;
  int? _selectedOS;
  int? _selectedChipset;
  int? _selectedBattery;
  int? _selectedYear;
  int? _selectedCondition;
  int? _selectedPTA;
  int? _selected5G;
  int? _selectedOriginal;
  String? _price;
  bool _loading = false;

  Future<void> _predictPrice() async {
    setState(() {
      _loading = true;
    });

    final uri = Uri.parse('http://192.168.107.85:5000/predictPrice');
    final response = await http.post(
      uri,
      headers: {'Content-Type': 'application/json'},
      body: json.encode({
        "Phone Company": _selectedCompany,
        "RAM": _selectedRAM,
        "Storage": _selectedStorage,
        "Processor": _selectedChipset,
        "Battery Capacity": _selectedBattery,
        "Operating System": _selectedOS,
        "External Condition": _selectedCondition,
        "5g Support": _selected5G,
        "PTA Approved": _selectedPTA,
        "Year of Release": _selectedYear,
        "Original Price": _selectedOriginal
      }),
    );
    if (response.statusCode == 200) {
      final result = json.decode(response.body);
      setState(() {
        _price = result['predicted_price'].toString();
      });
    } else {
      setState(() {
        _price = 'Error: ${response.reasonPhrase}';
      });
    }

    setState(() {
      _loading = false;
    });
  }

```

API

Front On Prediction (Backend)

```python
@app.route('/predictFrontOn', methods=['POST'])
def predictFrontOn():
    try:
        if 'file' not in request.files:
            return jsonify({'error': 'No file provided'}), 400

        file = request.files['file']
        file_path = os.path.join(app.config['upload_folder'], file.filename)
        file.save(file_path)

        image = tf.keras.preprocessing.image.load_img(file_path, target_size=(256, 256))
        image_array = tf.keras.preprocessing.image.img_to_array(image) / 255.0
        image_array = np.expand_dims(image_array, axis=0)

        prediction = modelFrontOn.predict(image_array)
        predicted_class = np.argmax(prediction, axis=-1)[0]
        confidence = np.max(prediction)

        os.remove(file_path)

        return jsonify({
            'predicted_class': int(predicted_class),
            'confidence': float(confidence)
        })

    except Exception as e:
        return jsonify({'error': str(e)}), 500
```

Front Off Prediction (Backend)

```python
@app.route('/predictFrontOff', methods=['POST'])
def predictFrontOff():
    try:
        if 'file' not in request.files:
            return jsonify({'error': 'No file provided'}), 400

        file = request.files['file']
        file_path = os.path.join(app.config['upload_folder'], file.filename)
        file.save(file_path)

        image = tf.keras.preprocessing.image.load_img(file_path, target_size=(256, 256))
        image_array = tf.keras.preprocessing.image.img_to_array(image) / 255.0
        image_array = np.expand_dims(image_array, axis=0)

        prediction = modelFrontOff.predict(image_array)
        predicted_class = np.argmax(prediction, axis=-1)[0]
        confidence = np.max(prediction)

        os.remove(file_path)

        return jsonify({
            'predicted_class': int(predicted_class),
            'confidence': float(confidence)
        })

    except Exception as e:
        return jsonify({'error': str(e)}), 500
```

Back Prediction (Backend)

```python
@app.route('/predictBack', methods=['POST'])
def predictBack():
    try:
        # Check if the file is in the request
        if 'file' not in request.files:
            return jsonify({'error': 'No file provided'}), 400

        # Get the file from the request
        file = request.files['file']
        file_path = os.path.join(app.config['upload_folder'], file.filename)
        file.save(file_path)

        # Preprocess the image
        image = tf.keras.preprocessing.image.load_img(file_path, target_size=(256, 256))
        image_array = tf.keras.preprocessing.image.img_to_array(image) / 255.0
        image_array = np.expand_dims(image_array, axis=0)

        # Predicting using the model
        prediction = modelBack.predict(image_array)
        predicted_class = np.argmax(prediction, axis=-1)[0]
        confidence = np.max(prediction)

        os.remove(file_path)

        # Returning the prediction result as JSON
        return jsonify({
            'predicted_class': int(predicted_class),
            'confidence': float(confidence)
        })

    except Exception as e:
        # Handling errors
        return jsonify({'error': str(e)}), 500
```

Price Prediction (Backend)

```python
@app.route('/predictPrice', methods=['POST'])
def predictPrice():
    try:
        if not request.is_json:
            return jsonify({'error': 'Request must be in JSON format'}), 400

        data = request.get_json()

        expected_features = [
            'Phone Company', 'RAM', 'Storage', 'Processor', 'Battery Capacity',
            'Operating System', 'External Condition', '5g Support', 'PTA Approved',
            'Year of Release', 'Original Price'
        ]

        if not all(feature in data for feature in expected_features):
            return jsonify({'error': f'Missing one or more features: {expected_features}'}), 400

        input_data = [data[feature] for feature in expected_features]
        input_array = np.array(input_data).reshape(1, -1)

        predicted_price = modelPrice.predict(input_array)[0]

        predicted_price = float(predicted_price)

        return jsonify({'predicted_price': round(predicted_price, 2)})

    except Exception as e:
        print(f"Error occurred: {e}")
        return jsonify({'error': str(e)}), 500
```

Plagiarism Report

ORIGINALITY REPORT

14%	9%	7%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	3%
2	Submitted to SUNY, Empire State College Student Paper	1%
3	V. Sharmila, S. Kannadhasan, A. Rajiv Kannan, P. Sivakumar, V. Vennila. "Challenges in Information, Communication and Computing Technology", CRC Press, 2024 Publication	1%
4	www.mdpi.com Internet Source	1%
5	H.L. Gururaj, Francesco Flammini, S. Srividhya, M.L. Chayadevi, Sheba Selvam. "Computer Science Engineering", CRC Press, 2024 Publication	1%
6	Pinheiro, Daniel Silva. "A Double Crystal Spectrometer for High Precision X-Ray Spectroscopy of Fundamental Physics and Applications", Universidade NOVA de Lisboa	<1%