Machine Learning Project: Mobile Price Classification

MARIO IZQUIERDO ÁLVAREZ, ANTONIO GIMÉNEZ LÓPEZ, FERNANDO NÚÑEZ SÁNCHEZ, and MARTÍN ROMERO ROMERO

This study addresses the challenge of mobile phone price prediction in a market inundated with diverse options, aiming to assist consumers in making informed purchasing decisions. It emphasizes the significance of selecting a phone that offers value for money, considering the desired features and budget constraints. Utilizing a dataset commonly adopted in previous research, this project applies machine learning techniques to estimate mobile phone prices based on their various features. Five different algorithms—Artificial Neural Networks (ANN), k-Nearest Neighbors (kNN), Support Vector Machines (SVM), Decision Trees (DT), and Stacking—are implemented and tested. The study concludes that the ANN model, without dimensionality reduction, outperforms other models with a remarkable accuracy rate. This model's effectiveness is demonstrated through rigorous cross-validation and testing on unseen data, showcasing its potential to guide consumers in the ever-evolving mobile phone market.

Additional Key Words and Phrases: Mobile Phone Price Prediction, ANN, KNN, SVM, Decision Tree, Stacking, Ensemble, PCA, Machine Learning, Cross-Validation, Feature Selection, Dimensionality Reduction.

ACM Reference Format:
Mario Izquierdo Álvarez, Antonio Giménez López, Fernando Núñez Sánchez, and Martín Romero Romero. 2023. Machine Learning Project: Mobile Price Classification. 1, 1 (December 2023), 18 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

1.1 Explanation On Problem To Be Solved

Mobile phones play a significantly important role in today's society, acting as fundamental tools that go beyond their function of communication. These devices are now considered personal extensions, facilitating a wide range of everyday activities. From managing work tasks to instant connectivity with friends and family, mobile phones have become an essential component of our daily lives. In view of this, the main concern of the users is that the phone should be worth the price it cost in terms of their desired features.

In the rapidly evolving market of mobile technology, consumers are faced with too many options when it comes to select a device that aligns with their preferences and needs. The abundance of mobile phones, with diverse features and intervals of prices, has made the decision-making process increasingly complex. This wide range of options has given rise to the need for an effective and efficient mechanism to aid consumers in making informed decisions based on their budget constraints and desired specifications. The importance of this project is not only to predict the price of the mobiles based on their different properties, but also to enable the customers to have a certain knowledge of the standard price of phones, which will enhance people to make rational decisions when buying a cell phone in the future.

For this work, machine learning techniques will be applied for estimating mobile phone prices based on different features of them. It will test four different types of models: Artificial neural networks (**ANN**), k-Nearest Neighbors (**kNN**), Support Vector Machines (**SVM**) and Decision Trees (**DT**).

Authors' address: Mario Izquierdo Álvarez; Antonio Giménez López; Fernando Núñez Sánchez; Martín Romero Romero.

2023. XXXX-XXXX/2023/12-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1.2 State Of The Art

In a review of the literature, it can be found that several investigations have been conducted about mobile phone price prediction (many of them with this dataset) with artificial intelligence and machine learning algorithms.

For this purpose, in Ercan et al. [1] compared Logistic Regression (LR), Support Vector Machine, Decision Tree and k-Nearest Neighbors without using dimensionality reduction techniques. Their results were that Support Vector Machines algorithms were the one that achieve the highest accuracy on their predictions about mobile price intervals. Others studies, like the conducted by the University College London [2], or by the Istanbul Technical University [3], also support that the best models are those based on SVMs. These two also demonstrated by statistical test that the feature selection method is the most appropriated for this dataset. In their investigations, they used a dataset that contained the four most correlated features with the price range. The best models that have been achieved in all these studies have achieved an accuracy of approximately 95%.

In Kalaivani et al. [4], it was used 3 machine learning models as Support Vector Machine, Random Forest and Logistic Regression selecting the first ten characteristics, getting a SVM model with 95% of accuracy, in this dataset. Then, the applied feature selection (top 10 correlated features with price range) and finally the obtained a SVM model with higher accuracy (97%).

Moreover, in Güvenç et al. [5] compared k-nearest neighbors and Deep Neural Networks (DNN) models and they found that DNN achieved a better performance for mobile price range predictions, with an accuracy of the 94%.

Table 1. Studies methodology and results

Author	Year	Preprocessing	Algorithms	Accuracy Rate
Çetin	2021	Standarization Feature Selection	SVM - DT - KNN	96% - 87% - 91%
Kalaivani	2021	Normalization Feature Selection	SVM	95%
Güvenç	2021	Normalization No Dimensionality Reduction	KNN - DNN	93% - 94%
Hu	2022	Normalization Feature Selection	SVM - DT - KNN	96% - 87% - 94%
Aksoy Ercan	2023	Normalization No Dimensionality Reduction	SVM - DT - KNN	96% - 82% - 41%

Many machine learning models have been tested in various studies, some with a low number of features and some including external evaluations.In view of this situation, this study aims to enhance the literature on predicting phone prices by employing 5 distinct machine learning algorithms and different approaches to pre-processing the database.

Note that the studies discussed here were conducted using the same dataset that we will utilise.

1.3 Theoretical Framework Of Machine Learning Algorithms

In this section, it will be described the different machine learning algorithms that will be implemented and tested in this project:

(1) **Decision Tree**:
Decision tree is a supervised machine learning method employed in classification tasks.Their primary objective is to build a predictive model that infers the values of target variables by learning direct decision rules derived from data features. There are decision and leaf nodes according to the goal and independent variables in the decision tree algorithm. Because it is a categorization technique that produces a tree-like structure, it is known as a decision tree. Conceptually, a tree can be perceived as a piece-wise invariable estimation. Notably, due to the simple principles guiding this algorithm, as well as its efficacy and visual interpretability, decision trees are used in diverse fields such as medicine and data mining analysis.

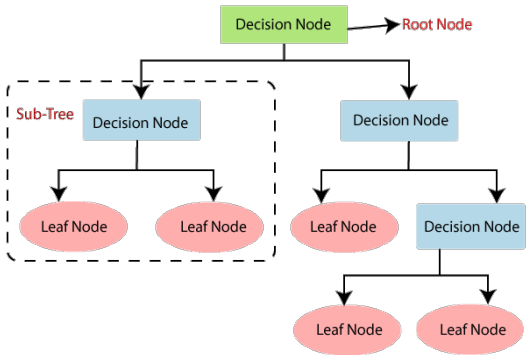

Fig. 1. Decision Tree Architecture

(2) **K-Nearest Neighbors**:
Neighbors based classification is a non-generalizing learning method based in the real data, retaining instances from the training data without attempting to construct a conventional internal model. KNN stands out as the most frequently utilized technique within this category, relying on the proximity of neighbors to each query point. The steps in the KNN classifier algorithm are straightforward: begin by assigning a value to k (determining the number of nearest neighbors to consider), then calculate the distance between the testing objects and every object in the set of training objects. Subsequently, identify the closest training object relative to the test object. The subsequent step involves selecting the class with the highest count of matched objects, and iteratively repeat the process until the same class is consistently obtained. Note that as a lazy learning method, it operates at a slow pace in handling large amounts of data.

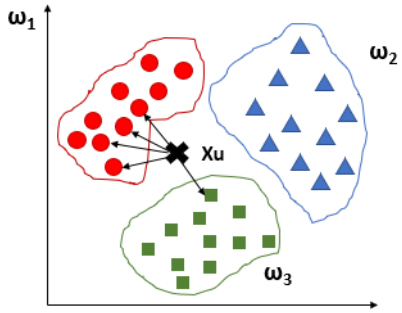

Fig. 2. KNN Architecture

(3) **Support Vector Machine**:
SVM models demonstrate efficacy when the training data is linearly separable, trying to identify the partition hyperplane that maximizes the hard margin. This preference arises from the recognition that such a hyperplane yields the most robust classification outcomes. In scenarios where the data is linearly separable, this classifier is a linearly separable support vector machine. Additionally, the classifier can adapt through soft margin maximization when the training data is only roughly linearly separable, giving rise to a linear support vector machine.

However, when the data defies linear separation, SVM employs kernel methods and soft interval maximization to establish non-linear support vector machines. These techniques broaden the applicability of SVM to handle complex, non-linear relationships within the data, enhancing its versatility across various problem domains.

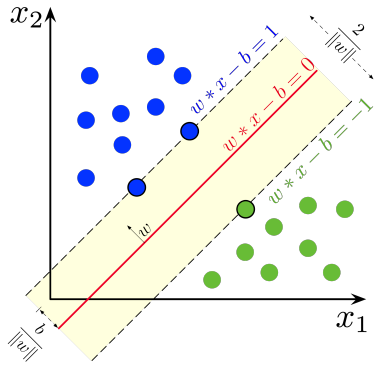

Fig. 3. SVM Architecture

(4) **Artificial Neural Network**:
Artificial Neural Networks (ANNs) are a machine learning model inspired by the functioning of the human brain. They are excellent at complex tasks, which is why they're used in a wide range of applications. ANNs do not require explicit programming as opposed to traditional algorithms, but they learn from data and deduce intricate patterns and relationships.

ANNs consist of interconnected nodes, like artificial neurons, organised into layers. The neural network comprises an input layer, one or more hidden layers, and an output layer. Its internal settings are modified through backpropagation as the network gains knowledge. This process involves adjusting neuron connections' strengths to minimize the differences between predicted and actual outcomes.

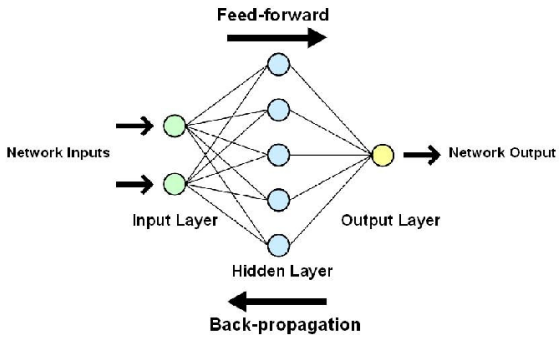

Fig. 4. ANN Architecture

(5) **Stacking**:

Stacking, a potent ensemble learning method, combines predictions from several base models to enhance overall predictive performance. Stacking leverages the strengths of diverse models, mitigates individual drawbacks and improves overall accuracy unlike traditional algorithms that rely on a single model.

The process consists of two stages. First, a range of diverse base models is trained on the input data. These models can incorporate diverse algorithms, capturing varied aspects and nuances within the dataset. Subsequently, the forecasts of these fundamental models are fed as input to an advanced model, commonly known as the meta-model or blender.

The meta-model is developed to study the combined predictions of the base models, identifying patterns and relationships that may be missed by individual models. The integration of diverse base models leads to a more robust and comprehensive predictive model in the second stage of stacking.

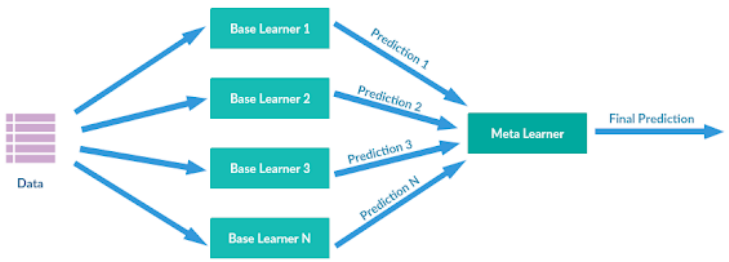

Fig. 5. Stacking Architecture

2 MATERIALS AND METHODS

2.1 Description of the database

The dataset used to build and evaluate the methods is called "Mobile Price Classification" and was taken from Kaggle.com. This dataset contains two data files called "**train**" and "**test**" with 2000 and 1000 samples of mobile phone, respectively. The first one presents 2000 samples of mobile phones with 20 features of each one and one last column with their price range (**target**). On the other hand, the "test" data file contains 1000 more samples with their 20 features, but it does not have the targets, so it cannot be used to test the effectiveness of the models.

In this situation, we have used the **"train" data file** as the **complete data**, from which the train and test data set are obtained using the hold-out method.

The database includes the phone specifications listed below, of which 7 out of 20 features are binary:

- "**battery power**" which is measured in mAh.
- "**blue**", which represent if the phone has Bluetooth or not.
- "**dual-sim**", also representing if the phone has or not dual sim.
- "**fc**" indicating the front camera's megapixels.
- "**m_dep**" and "**mobile_wt**" denoting the phone's depth and weight; "**n_cores**" representing the number of processor cores.
- "**pc**" representing the primary camera's megapixels.
- "**px_height**" and "**width**" referring to pixel resolution's height and width.
- "**ram**" is random access memory in megabytes.
- "**sc_h**" and "**sc_w**" is screen height and width of mobile in centimeters
- "**talk time**" represents the longest duration that a single battery charge will last.
- "**price range**", which is the target variable and represent the range price of the sample phone. It is divided into 4 groups, which are 0, 1, 2 and 3, corresponding to low, median, high and very high price levels, respectively.

The descriptive statistics of the features of the dataset are shown below, in Table 2.

Table 2. Descriptive statistics of the features of the samples

	count	mean	std	min	25%	50%	75%	max
battery_power	2000	1238.519	439.4182	501	851.75	1226	1615.25	1998
blue	2000	0.495	0.5001	0	0	0	1	1
clock_speed	2000	1.52225	0.816004	0.5	0.7	1.5	2.2	3
dual_sim	2000	0.5095	0.500035	0	0	1	1	1
fc	2000	4.3095	4.341444	0	1	3	7	19
four_g	2000	0.5215	0.499662	0	0	1	1	1
int_rmemory	2000	32.0465	18.14571	2	16	32	48	64
m_dep	2000	0.50175	0.288416	0.1	0.2	0.5	0.8	1
mobile_wt	2000	140.249	35.39965	80	109	141	170	200
n_cores	2000	4.5205	2.287837	1	3	4	7	8
pc	2000	9.9165	6.064315	0	5	10	15	20
px_height	2000	645.108	443.7808	0	282.75	564	947.25	1960
px_width	2000	1251.516	432.1994	500	874.75	1247	1633	1998
ram	2000	2124.213	1084.732	256	1207.5	2146.5	3064.5	3998
sc_h	2000	12.3065	4.213245	5	9	12	16	19
sc_w	2000	5.767	4.356398	0	2	5	9	18
talk_time	2000	11.011	5.463955	2	6	11	16	20
three_g	2000	0.7615	0.426273	0	1	1	1	1
touch_screen	2000	0.503	0.500116	0	0	1	1	1
wifi	2000	0.507	0.500076	0	0	1	1	1
price_range	2000	1.5	1.118314	0	0.75	1.5	2.25	3

Analyzing the Figure 6, it can be seen that characteristics have a distribution that is quite uniform. Moreover, there are not missing values in any characteristic in the data set. Regarding the **price range**, each class has 500 instances, so we have that all classes are **well-balanced**. This will be useful in order to avoid problems related to the shortage of instances of some classes that may lead to models not detecting some of these classes.

2.2 Generating the data-sets

The dataset has 20 features that allow to know the range in which the phone is located. For the training of the models, three datasets will be used, which will be created from this initial dataset. For the first dataset, the PCA dimensionality reduction technique will be used. For the second one only four components of the dataset will be used (the features most related to the output). And the last dataset will be the complete dataset. Furthermore, for the measurement of the metrics, the dataset has been split in two parts: one for training (80% of the dataset) and one for test (remaining 20%).

2.3 Data Preprocessing

For the data processing, a normalization process have been carried out, which allows the models to converge faster, improving the generalization capacity of the model and making it more robust. For this dataset, its different characteristics have been analyzed and we have decided to apply a MinMax normalization. This type of normalization has been chosen because it is preferable to apply MinMax normalization on data that do not follow a Gaussian distribution. In order to get to know the nature of the data and to see whether or not they follow a Gaussian distribution, histograms have been computed for each feature of the dataset, as is shown in Figure 6.

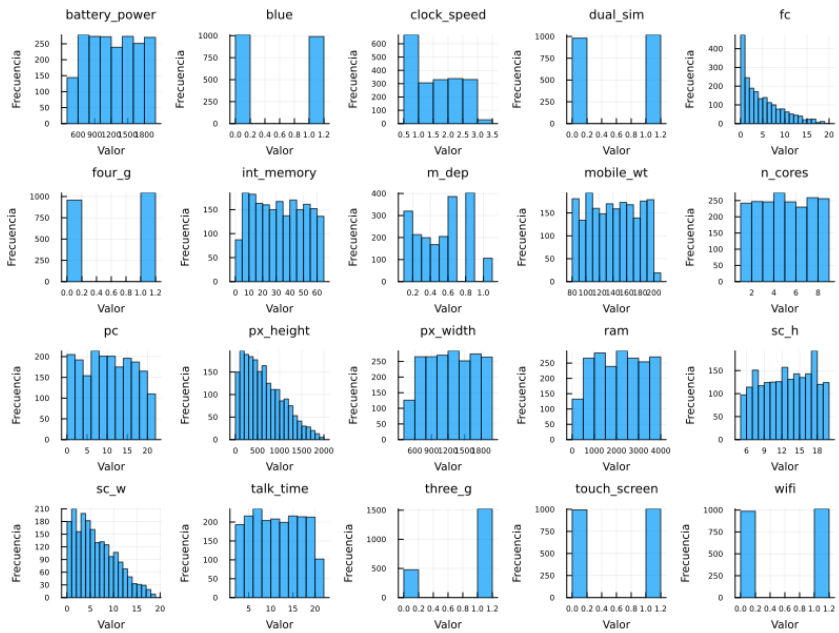

Fig. 6. Features histograms

2.4 Dimensionality Reduction

For the subsequent training of the models, the data will be preprocessed by applying the PCA dimensionality reduction technique, and the results of the models will be compared with the models that have been trained without using any dimensionality reduction technique. Another dimensionality reduction method we used has been feature selection. Specifically, four features that were most closely related to the outputs were selected in order to train the model. In order to know which features are more relevant, a heat map (Figure 7) has been calculated which allowed us to know the relationship between the features and the output.

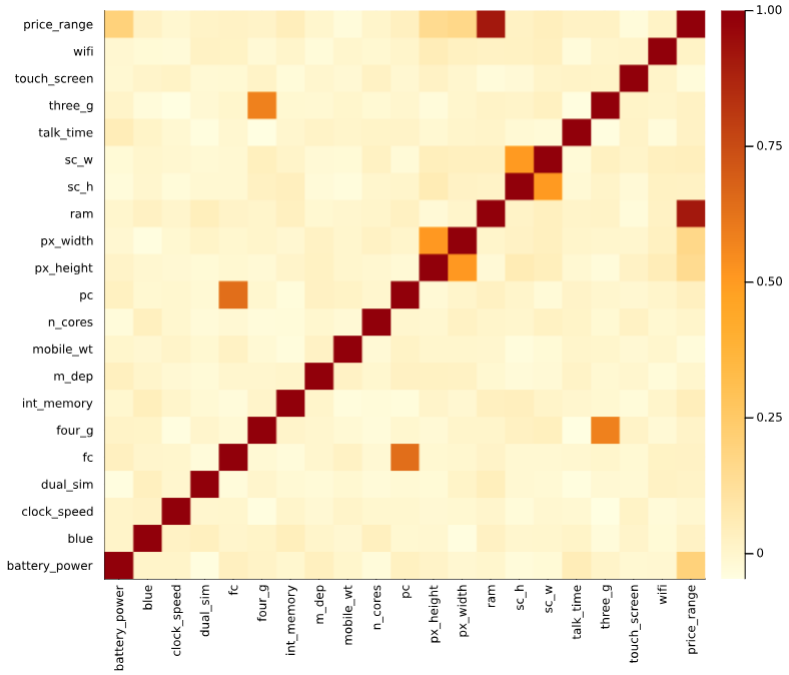

Fig. 7. Correlation Matrix

2.5 Experimental Methodology

For the development of the experimental part, KNN, ANN, SVM and DT models have been created for the three data sets (NDR, PCA and FSelection).

To train these models we have used the cross-validation method, in which the data will be divided into 10 folders (number of folders widely used in cross-validation). This method allows us to obtain very reliable results of how our model will behave, and then train the final model without this technique using all possible training data. For the training process of each model, first some hyperparameters are initialized and we see which are the results obtained. After analyzing these results, we redefine other hyperparameters based on the models with better results and changing (from these better models) some parameter to try to improve the results. This process is repeated several times until we try to achieve the best possible results.

For the process of creating the stacking models, the best models trained in the previous process have been used as the models that will be part of the ensemble. In addition, a *"leave one out"* technique will be used in which for each stacking model one of the individual models is left out. This allows us to see which models perform better and worse in a quick way and without having to create too many ensembles. Despite using this *"leave one out"* technique, we have also trained an ensemble with all the individual models.

2.6 Justification of the metric or metrics to be used

As previously discussed, it is being used a well-balanced dataset of mobile price ranges. Therefore, we do not need to be concern about facing problems of shortage of instance of a single class that can lead to difficulties to predict that mentioned class. For the context, prioritizing the reduction of the number of false positives or false negatives is not a significant factor (as it can be in a medical environment).

In the light of this, the performance of each of the models will be obtained and compared by the accuracy rate, which has also been used in several of the above-mentioned studies [3] [5] [4]. This metric, represents the percentage of well-classified samples and it is expressed as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

Furthermore, sometimes several models may have similar accuracy, so the f-score will also be obtained to get more information on the performance of the model.

This f-score is calculated by the harmonic mean of recall and precision, and it is used when it is critical to reduce the number of false positives as well as false negatives. It is used especially when there is an imbalance between classes. It follows this expression:

$$F = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2}$$

where

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Note that in the notebooks all the following metrics will be displayed, but only these two will be used for the comparison between models: Accuracy, Sensitivity, Specificity, PPV, NPV, F-score and Error Rate.

Finally, for the final evaluation of the top-performing model achieved, aside from measuring accuracy and f-score, we will exhibit the normalized confusion matrix. This allows us to examine the overall performance of the model and identify any shortcomings in the prediction of any specific class.

2.7 Explanation of the code structure and how it has been organised.

In this section it will be explained the structure of the repository of the project. This resource aims to facilitate comprehension of the development of the dataset analysis and pre-processing methods, as well as the training and comparison of models and subsequent re-training of the final model for performance evaluation on new data.

Here is the list of the key files and folders:

- `data_expl_and_preproc.ipynb`: This is the main executable file for the exploration of the database and the preprocessing of the data. In this file is performed the oneHotEncoding used to train and test the final model, it is also performed the split of the dataset into train and test inputs and outputs. and then the normalization of the input train and test datasets. Finally, three different approaches of preprocessing are conducted, and the final datasets are stored.

- `final_evaluation.ipynb`: This is the main executable file for the final evaluation of the best obtained model during the experimental phase. In this file, the HoldOut at the beginning of the study is used to train the model with the whole train dataset, while the test data is used for the first time to obtain all metrics and the normalized confusion matrix to analyze the final performance of the model on new unseen data.

- `modules/`: This folder contains all the functions necessary for the realization of the project, divided and grouped by modules in .jl files.

  - `modules/ANN_Modeling.jl`: Contains function definitions necessary to create, train and evaluate ANN based models with `Flux`.

  - `modules/Evaluation.jl`: This module provides different utilities to evaluate the models with various metrics.

  - `modules/ModelSelection.jl`: This module provides functions for splitting the data with different standard techniques, such as HoldOut and cross-validation.

  - `modules/Plotting.jl`: This module provides functionalities for printing and plotting different metrics of evaluation.

  - `modules/Preprocessing.jl`: This module provides functions for data preprocessing: Encodings, normalization, etc.

  - `modules/Sk_modeling.jl`: This module provides functionalities for modeling with sklearn utilities. It covers full training and evaluation functions of different base models and also for the ensemble `StackingClassifier`.

Then, we have **three folder**s which follow the same structure. These folders contain the data preprocessed with one of the three approaches proposed in this project and the experiments performed to analyze the performance of each of the models on that data.

These three folders are **No Dimensionality Reduction (NDR/)**, **Principal Components Analysis (PCA/)** and **Features Selection (FSelection/)**. Inside each is a `data.h5` file with the

data preprocessed according to its approach, and also we have five jupyter notebooks files with all the process of training and evaluating different architectures of models created from Artificial Neural Networks, k-Nearest Neighbors, Decision Trees, Support Vector Machines and Stacking Ensembles techniques, respectively.

3 DISCUSSION

3.1 Results Of The Experimental Part

In this section, the results obtained during the testing phase of different architectures of each of the models will be discussed.

3.1.1 ***Non dimensionality reduction.***

Table 3. No dimensionality reduction model comparison

Metrics	Decision Tree	KNN	SVM	ANN	Stacking
Accuracy	0.849 ± 0.026	0.694 ± 0.035	0.957 ± 0.01	0.991 ± 0.008 *	0.966 ± 0.015
F-Score	0.85 ± 0.025	0.701 ± 0.032	0.957 ± 0.01	0.991 ± 0.008 *	0.966 ± 0.015

For data for which no dimensionality reduction techniques have been applied, analizing Table 3, it can be determined that:

The **ANN** model demonstrated the **best performance**, achieving the highest scores in both accuracy and F-score. This suggests that the **ANN** model was able to correctly classify a high proportion of instances in our dataset. The high F-score indicates that the **ANN** model achieved a good balance between precision (the proportion of true positive instances among all instances predicted as positive) and recall (the proportion of true positive instances among all actual positive instances).

The **Stacking** and **SVM** model also showed excellent performance. While they did not match the results of **ANN**, they still achieved respectable scores in terms of accuracy and F-score. The moderate performance of **SVM** models suggests that they are still a viable choice for the problem, perticularly if the computational cost or interpretability are crucial considerations.

Finally, the **kNN** and **DT** models demonstrated the lowest performance in terms of both metrics. However, it is worth noting that **DT** models can still be usefull in certain scenarios, as they are highly interpretable.

All the findings are consistent with the existing literature, which indicates that **SVM** based models perform best with an accuracy of **95%** [1]. However, no **ANN** or **ensembler** models have been tested. This discovery suggests that ANN and Stacking models may be more effective for this type of problem.

3.1.2 ***Principal component analysis.***

Table 4. Principal component analysis model comparison

Metrics	Decision Tree	KNN	SVM	ANN	Stacking
Accuracy	0.614 ± 0.035	0.706 ± 0.035	0.939 ± 0.014	0.977 ± 0.01 *	0.936 ± 0.02
F-Score	0.615 ± 0.037	0.713 ± 0.032	0.939 ± 0.014	0.977 ± 0.01 *	0.936 ± 0.02

For data to which PCA has been applied, analyzing Table 4, it has been discovered that:

The **ANN** model maintained its **superior performance**, obtaining the highest accuracy score. This indicates the ability of the ANN model to correctly classify a significant number of instances in our dataset, even after the application of PCA for dimensionality reduction.

As in the previous section, **SVM** and the **Stacking** model showed excellent performance, with accuracy scores very close to the **ANN** model. This indicates that these models were also able to handle the dimensionality reduction well and classify most instances correctly.

The **KNN** model showed improved performance compared to the previous analysis without PCA, but it still did not match the performance of the **ANN**, **SVM**, and **Stacking** models.

The **DT** model showed the lowest accuracy score, lower than the value obtained without the dimensional reduction. this suggest that this model might not be the best choice when using PCA.

As with no dimensional reduction performed, all the findings made applying PCA are consistent with the existing literature, which indicates that **SVM**-based models perform best, achieving accuracies rates **95%** and **96%** [1] [2] [3] [4]. However, no **ANN** or **ensembler** based models have been tested neither when applying PCA, and this study suggest that these models may be more effective for this problem.

3.1.3 ***Feature selection.***

Table 5. Feature selection model comparison

Metrics	Decision Tree	KNN	SVM	ANN	Stacking
Accuracy	0.881 ± 0.022	0.912 ± 0.022	0.958 ± 0.016	0.964 ± 0.015 *	0.958 ± 0.016
F-Score	0.881 ± 0.022	0.913 ± 0.022	0.958 ± 0.016	0.964 ± 0.014 *	0.958 ± 0.016

For data to which feature selection has been applied, studying Table 5, it is revealed that:

As expected, the **ANN** model achieved the **greatest results**, getting the highest score in both metrics. This indicates that **ANN** models are the most suitable for the data and generate the most accurate predictions, irrespective of the reduction technique employed.

As in previous sections, the **SVM** and **Stacking** models showed excellent performance, with accuracy scores very close to the **ANN** model. This indicates that these models were also able to handle the feature selection well and classify most instances correctly.

The **KNN** model showed improved performance compared to the previous analysis without feature selection, but it still did not match the performance of the **ANN**, **SVM**, and **Stacking** models.

The **DT** model showed significantly improved accuracy score compared to the previous analyses. This suggests that this model have benefited from the feature selection process.

As in previous studies, revised in the literature, it can be seen that excluding **ANN** and **Stacking** models, applying the feature selection technique improve the performance of all the models tests. Again, the best **SVM** model have a similar accuracy rate than in the literature, but **ANN** and **Stacking** models outperforms it.

3.2 Comparison Of The Dimensional Reduction Techniques

Our analysis revealed significant differences in the performance of the models for the different dimensionality reduction techniques. Here are the detailed results:

The three charts above show the results that are consistent with those obtained in previous studies. As discussed in the literature review, the graphs show that models based on **SVM**, **KNN** or **DT** perform **worse** when the **feature selection** is **not applied**. All the others techniques tested has given worse results. The excellent performance of **SVM** may indicate that the data are highly likely to be **linearly separable**. Furthermore, for these models, applying **PCA** is significantly **worse** than not applying any **dimensionality reduction technique** in terms of accuracy and f-score.

On the flip side, both **ANN** and **Stacking** exhibit **exceptional performance** regardless of the type of dimensionality reduction technique employed. However, these algorithms demonstrate superior performance when **no reduction technique** is utilised, yielding significantly **better results** than those reported in the **literature**. This shows clearly that the **ANN** models, in this problem, receive significant **benefits** from having a **larger quantity** of features within the dataset for training **parameter optimization**. This improvement is more pronounced as the number of features used increases. In the case of **Stacking** models, the use of **principal components** as data-set leads to a **worse optimisation** of the parameters of the **base models** and therefore to worse results than if the original characteristics were used.

3.3 Final Performance Of The Optimal Models

Once all the machine learning models have been rigorously evaluated using cross-validation across the different dimensionality reduction techniques, it can be seen that the models which yields the best performances were the **ANN** and **Stacking**, **without applying** any dimensionality reduction technique, and **SVM** with **Feature Selection**. These models have exhibited a great performance, with an accuracy of **(0.991 ± 0.008)**, **(0.966 ± 0.015)** and **(0.958 ± 0.016)** respectively in the evaluation with cross-validation, as can be seen in **Table 3** and **Table 5**.

To further validate the robustness of the chosen models, the whole train data has been used to train the ANN, Stacking and SVM. Afterward, the test dataset which was held out at the beginning of this study was used. In this evaluation process, we have used the set of hyperparameters which have demonstrated better performance in the experimental phase:

ANN:

- The topology consists of **two hidden layers** of **128** and **64** neurons.
- The **Sigmoid** was used as the transfer function.
- The learning rate was fixed to **0.01**
- We used a **20%** of training data to validate.
- We fixed the maximum number of epochs without improvement in validation loss to **80 epochs**.
- **2000 epochs** as maximum.
- Given the stochastic nature of ANNs, the experiment was replicated **ten times**, obtaining a more reliable evaluation.
- The minimum loss variation to consider as improvement was of **0.0**

Stacking:

- The topology consists of 3 models: **DT**, **KNN**, and **ANN**.
  - **DT model:**
    - The max depth is **6**.
    - The criterion used is **entropy**.
  - **KNN model:**
    - The number of neighbors are **360**.
    - The metric used is **mahalanobis** distance.
  - **ANN model:**
    - The topology conist of **two hidden layers** of **64** and **32** neurons.
    - The activation used is **logistic**.
    - The learning rate is **0.01**.
    - The validation fraction is **0.2**.
    - The number of no iterations change are **80**.
    - The maximum of iterations are **2000**.

SVM:

- The kernel of the model is **linear**.
- The C value is **8**.

The results of these experiments yielded an overall accuracy of **(0.9612 ± 0.0064)** for ANN, **(0.945)** for Stancking **(0.96)** for SVM. This slight decrease from the cross-validation accuracy observed during the training phase can be attributed to the model being exposed to new, previously unseen data.

To provide further insight into the models' predictive capabilities across the different classes, the normalized confusions matrices from Figures 8, 9 and 10 are used.

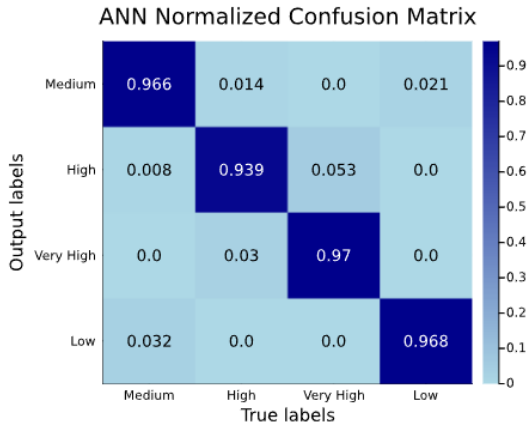

Fig. 8. Normalized Confusion Matrix for the Optimal ANN Model.

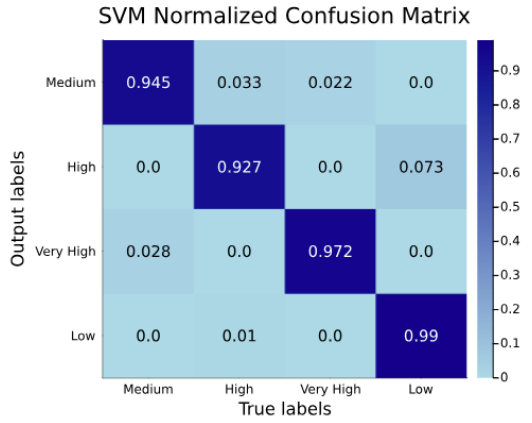

Fig. 9. Normalized Confusion Matrix for the Optimal SVM Model.

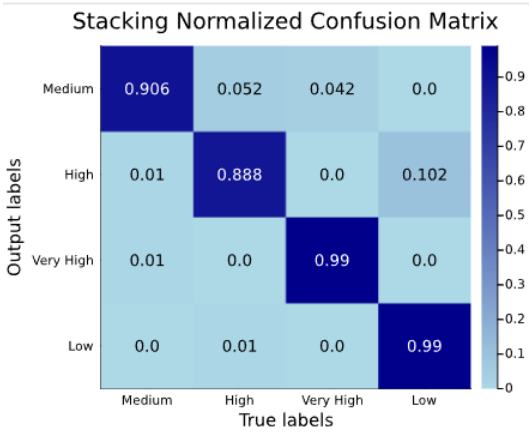

Fig. 10. Normalized Confusion Matrix for the Optimal Stacking Model.

Table 6. Final models comparison

Metrics	ANN	Stacking	SVM
Accuracy	0.9612 ± 0.0064	0.945	0.96
F-Score	0.9625 ± 0.005	0.945	0.9601

Comparing these three models, we see that the best results are offered by ANN, followed by SVM and finally Stacking. Even so, the values obtained are quite similar and the choice of one of these models could be due to non-behavioral reasons, such as computational capacity.

If we look at the ANN, which is the one that has given the best result, it is particularly noteworthy that the **Very High** class exhibited the most favorable prediction outcomes, while the **High** class was associated with the least performance. The matrix also elucidates that misclassifications predominantly occurred between classes that are adjacent in terms of the attribute being predicted (i.e., **Very High** ↔ **High**, **Medium** ↔ **High**, and **Low** ↔ **Medium**). This does not occur in Stacking and SVM, where an error in these models will make the final classification farther from reality than an erroneous ANN classification would, (i.e., **High** ↔ **Low** or **Medium** ↔ **Very High**).

In contrast to the studies presented in the **Section 1.2**, our ANN model without dimensionality reduction has attained an accuracy of **(0.9612 ± 0.0064)**, which substantiates and, in some instances, **slightly surpasses** the performance of models reported in the literature, such as those by Çetin et al. [3], Kalaivani et al. [4], while the accuracy rates reported by Ercan et al.[1] and Hu et al. [2] **align closely** with our findings. In the case of SVM, with an accuracy of **0.96**, we obtained results similar to those obtained by Hu [2] and Ercan et al. [1], and even **slightly surpassing** those of Kalaivani et al [4]. To conclude, the final stacking model continues to outperform, albeit narrowly, the other models.

4 CONCLUSION

This comprehensive study of **cell phone price prediction** using various machine learning algorithms provides valuable insights into the relative performance of these models. The results unanimously support the **effectiveness** of models based on **support vector machines** and **artificial neural networks**, with an accuracy of approximately 95%, highlighting their ability to deal with the complexity of cell phone pricing data.

The application of dimensionality reduction techniques, such as **Principal Component Analysis** and **feature selection**, reveals different effects on model performance. It was observed that while **SVM** maintained its **robust performance** through these techniques, the **ANN** model showed **superior performance** when **no dimensionality reduction** was applied, suggesting a possible positive correlation with the number of available features.

In addition, it is important to emphasize that although **k-nearest neighbor** and **decision tree** models perform **poorly** compared to **SVM** and **ANN**, they should not be discarded completely. KNN and DT models, despite their lower accuracy, can be valuable in situations where model interpretability and simplicity are a priority.

As for the **stacking** approach, it is observed that this technique has shown promise by combining the strength of several individual models. The effectiveness of the stacking model is evidenced by its **competitive** performance, approaching the accuracies achieved by **SVM** and **ANN**, despite the simplicity of its base models.

The methodological structure of this study comprehensively addresses the issues of **dataset description**, **preprocessing**, and **model evaluation**, providing a solid foundation for replication and extension of this work in future research. In addition, the importance of careful consideration of **dimensionality reduction** techniques based on the specific **nature** of the **problem** and model architecture is emphasized.

In summary, this study provides a comprehensive and balanced view of the performance of various models in cell phone price prediction, emphasizing the importance of careful selection of models and preprocessing techniques based on the specific objectives of the problem and the intrinsic characteristics of the data.

5 BIBLIOGRAPHY

[1] Seda İpek AKSOY ERCAN and Murat ŞİMŞEK. 2023. Mobile phone price classification using machine learning. *International Journal of Advanced Natural Sciences and Engineering Researches*, 7, 4, 458–462.

[2] Ningyuan Hu. 2022. Classification of mobile phone price dataset using machine learning algorithms. In *2022 3rd International Conference on Pattern Recognition and Machine Learning (PRML)*. IEEE, 438–443.

[3] Mustafa Çetın and Yunus Koç. 2021. Mobile phone price class prediction using different classification algorithms with feature selection and parameter optimization. In *2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*. IEEE, 483–487.

[4] KS Kalaivani, N Priyadharshini, S Nivedhashri, and R Nandhini. 2021. Predicting the price range of mobile phones using machine learning techniques. In *AIP Conference Proceedings* number 1. Vol. 2387. AIP Publishing.

[5] Ercüment Güvenç, Gürcan Çetin, and Hilal Koçak. 2021. Comparison of knn and dnn classifiers performance in predicting mobile phone price ranges. *Advances in Artificial Intelligence Research*, 1, 1, 19–28.