BI2025 Experiment Report - Group 16

Muhammad Ibrar*
TU Wien
Austria

Ahmad Ibrahim†
TU Wien
Austria

Abstract

This report documents a machine learning experiment conducted by Group 16 following the CRISP-DM methodology. The goal is to predict the price range of mobile phones based on their technical specifications using supervised learning techniques. The experiment focuses on data understanding, preparation, modeling, evaluation, and deployment considerations, while emphasizing transparency and reproducibility.

CCS Concepts

• **Computing methodologies → Machine learning**.

Keywords

CRISP-DM, Machine Learning, Random Forest, Mobile Price Prediction

ACM Reference Format:
Muhammad Ibrar and Ahmad Ibrahim. 2025. BI2025 Experiment Report - Group 16. In *Proceedings of Business Intelligence (BI 2025)*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.5281/zenodo.18291709

1 Business Understanding

1.1 Data Source and Scenario

Data Source: The dataset is the Kaggle "Mobile Price Classification" dataset containing 2,000 mobile phones. Each phone is described by 20 technical features (battery power, RAM, internal memory, camera specs, connectivity options, screen dimensions, etc.) and one target variable price_range with four classes (0 = low, 3 = very high).

Scenario: A new mobile company wants to price upcoming phone models competitively against major brands. Instead of relying only on expert judgment, the company wants to analyze historical specifications of phones and their corresponding price ranges to support pricing decisions.

1.2 Business Objectives

The primary business objectives of this project are as follows:

- Support pricing decisions by predicting the most suitable price range for new phone models based on technical specifications.
- Reduce manual effort and time required for estimating price categories during early product planning stages.
- Improve product positioning across budget, mid-range, high-end, and flagship market segments.
- Increase transparency and understanding of how technical features influence pricing decisions.

1.3 Business Success Criteria

The success of the project from a business perspective is measured using the following criteria:

- The machine learning system is regularly used by product management and pricing teams.
- A reduction of at least 30% in the time required for initial price-range estimation.
- More than 80% of newly launched smartphone models remain within a ±10% deviation from their initially selected price band after market entry.
- Pricing decisions become more consistent and data-driven across all phone segments.

1.4 Data Mining Goals

The data mining goals translate the business objectives into concrete analytical tasks:

- Build a multi-class classification model that predicts the `price_range` (0–3) from 20 phone features.
- Achieve robust predictive performance on unseen data and generalize well to new device configurations.
- Identify and analyze the most influential features (e.g., RAM, pixel resolution, battery power) affecting price range.
- Provide probabilistic outputs to support uncertainty-aware pricing decisions.

1.5 Data Mining Success Criteria

The technical success of the data mining task is evaluated using the following criteria:

- Achieve at least 90% classification accuracy on the validation or test dataset.
- Reach a macro F1-score of at least 0.88, with no individual class having recall below 0.80.
- Demonstrate stable model performance across different random train-test splits.
- Produce reasonably calibrated probability estimates suitable for business decision-making.

*Student A, Matr.Nr.: 12350094
†Student B, Matr.Nr.: 11826186

BI 2025, -

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.5281/zenodo.18291709

1.6 AI Risk Aspects

Several potential risks related to the use of machine learning in pricing decisions were identified:

- Misclassification may lead to incorrect pricing decisions, negatively affecting revenue or sales volume.
- The dataset may not represent future device trends, introducing the risk of model drift over time.
- Over-reliance on automated predictions could result in poor decisions if expert judgment is excluded.
- Although no personal data is involved, systematic bias across device categories may still occur.
- Exposure of the pricing logic could pose a business security risk if the model is deployed externally.

2 Data Understanding

Dataset Description: The dataset consists of 2,000 instances with balanced class distribution across four price ranges. All features are numerical, including binary indicators and continuous measurements. Feature names, data types and a brief description is included in Table 1.

Table 1: Raw Data Features

Feature Name	Data Type	Description
battery_power	integer	Battery capacity (mAh).
blue	integer	Bluetooth support (0/1).
clock_speed	double	Processor clock speed (GHz).
dual_sim	integer	Dual SIM support (0/1).
fc	integer	Front camera (MP).
four_g	integer	4G support (0/1).
int_memory	integer	Internal memory (GB).
m_dep	double	Phone thickness (cm).
mobile_wt	integer	Weight (grams).
n_cores	integer	Number of processor cores.
pc	integer	Primary camera (MP).
price_range	integer	Target class (0–3).
px_height	integer	Screen pixel height.
px_width	integer	Screen pixel width.
ram	integer	RAM (MB).
sc_h	integer	Screen height (cm).
sc_w	integer	Screen width (cm).
talk_time	integer	Battery talk time (hours).
three_g	integer	3G support (0/1).
touch_screen	integer	Touchscreen support (0/1).
wifi	integer	WiFi support (0/1).

2.1 Categorical Feature Distributions

Categorical variables, primarily binary indicators (e.g., `blue`, `dual_sim`, `four_g`, `three_g`, `wifi`, `touch_screen`), were analyzed using pie-chart plots (Figure 1).

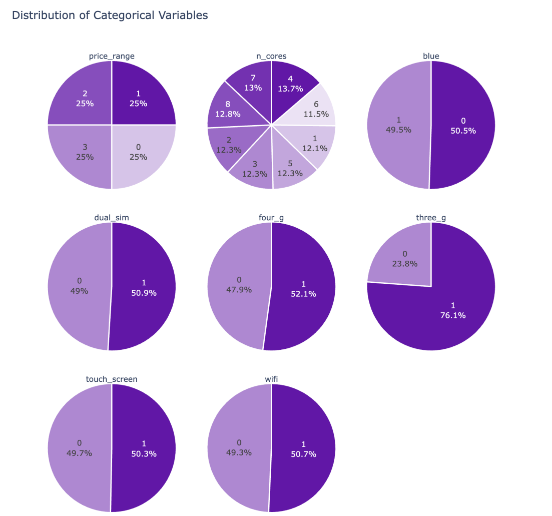

Figure 1: Distribution of categorical features and target variable `price_range`.

The target variable `price_range` is perfectly balanced, with exactly 500 samples per class, which is advantageous for multi-class classification. Most binary features show near-even splits between 0 and 1. An exception is `three_g`, where approximately 76% of devices support 3G. No unexpected or degenerate category distributions were observed.

Histograms with kernel density estimates were generated for all numerical attributes (Figure 2). Most numerical features exhibit unimodal distributions with realistic value ranges. However, some variables such as `px_height` and `sc_w` show a notable concentration of values near zero, suggesting potential noise or atypical records.

Skewness was computed for all numerical variables to assess distribution asymmetry. While several features exhibit mild skewness, no extreme distortions were observed that would immediately invalidate modeling. These findings inform later decisions regarding scaling or transformation during data preparation.

2.2 Outlier Detection

Outlier detection was performed using a z-score threshold of 3.0 across all numerical attributes. The analysis revealed that nearly all features contain no statistically significant outliers. The only exception is the `fc` (front camera megapixels) attribute, for which 12 observations were flagged as potential outliers.

Overall, the dataset appears largely clean, with only a small number of extreme values that may require attention in the data preparation phase.

2.3 Correlation Analysis

A correlation heatmap was computed for all numerical variables, including the target `price_range` and is shown in Figure 3. Most feature pairs show weak linear correlations, indicating low multicollinearity and suggesting that features provide largely complementary information.

Figure 2: Distributions of numerical features with histograms and kernel density estimates.

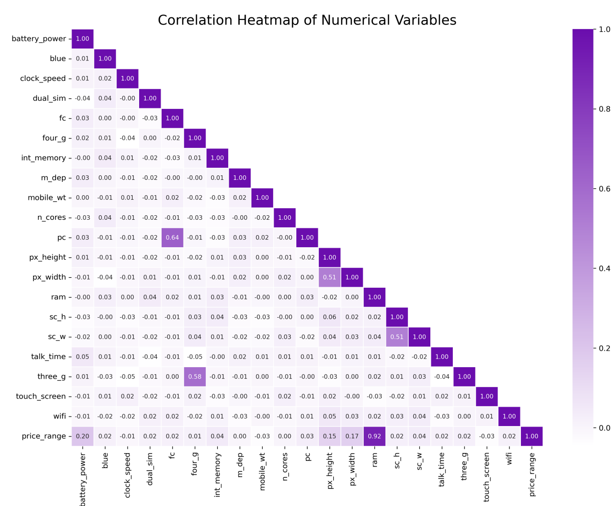

Figure 3: Correlation heatmap of numerical features, including the target variable `price_range`.

Moderate positive correlations were observed between related attributes, such as screen height and width, pixel height and width, and front versus primary camera resolution. Notably, a strong positive correlation of 0.92 was observed between `ram` and the target variable `price_range`, indicating that memory capacity is a key determinant of price category. Correlations between most other individual features and the target variable are moderate, supporting the suitability of multivariate modeling approaches.

2.4 Data Quality Assessment

A systematic data quality assessment showed that the dataset contains no missing values in any of the 21 columns and no duplicate rows. All data types are consistent with the intended semantics of the attributes.

A potential issue is that px_height contains 2 zero values and sc_w contains 180 zero values. Zero screen height or width is implausible for real mobile devices and may indicate noisy, special, or incorrectly recorded cases. These findings should be considered in the data preparation phase (e.g., deciding whether to filter, impute, or keep these records).

2.5 Ethical and Bias Aspects

The dataset contains no personal or sensitive attributes; all features describe technical phone specifications only (e.g., battery capacity, memory, screen resolution, connectivity options). The target variable `price_range` is perfectly balanced, with 500 instances per class, which reduces the risk of systematic bias in terms of underrepresented target categories.

Some device types and feature combinations are less frequent (for example, rare combinations of extreme screen or camera configurations). However, these imbalances primarily affect model performance and generalization rather than human fairness, because they do not correspond to demographic or protected population groups. Based on the available information, no ethical issues related to discrimination or sensitive attributes are apparent, and no specific demographic bias risks are present in this dataset.

2.6 Risks and Expert Questions

Despite the absence of personal data, several data- and model-related risks were identified. First, the representativeness of the dataset with respect to the real smartphone market is uncertain. Some feature combinations—such as very high camera megapixel values or screen dimensions close to zero—may not reflect realistic devices and could introduce noise or distort model behaviour. Second, the dataset may not include newer technologies or recent device trends, which increases the risk of model drift if the model is applied to future products.

These uncertainties motivate consultation with a domain expert. In particular, the following questions should be clarified:

- Are zero values for `px_height` or `sc_w` technically valid (e.g., encoding a special case), or are they more likely to be measurement or recording artifacts?
- Are unusually high values in `fc` (front camera megapixels) realistic device specifications, or should they be treated as outliers?

- Does the dataset represent a realistic mix of budget, mid-range, and high-end devices, or are certain market segments over- or underrepresented?
- Are any important device characteristics missing that strongly influence real-world pricing (e.g., brand, release year, or build quality)?

Answers to these questions would help to better assess potential data quality issues, refine preprocessing decisions, and estimate the robustness of the model in practical deployment scenarios.

2.7 Actions Required for Data Preparation

Based on the data understanding analysis, several concrete actions are recommended for the data preparation phase:

- Investigate and handle zero values in `px_height` (2 cases) and `sc_w` (180 cases), as such values are unlikely for real devices. Depending on domain expert feedback, these records may be filtered out or adjusted via imputation.
- Re-assess the 12 outliers detected in `fc` (front camera megapixels). Depending on whether they are confirmed as valid device specifications or artifacts, decide whether to retain, cap, or remove them.
- Standardize or scale numerical features (e.g., `ram`, `battery_power`, pixel resolution attributes) to account for differing value ranges and to support algorithms that are sensitive to feature scaling.
- Ensure that categorical binary features (e.g., `blue`, `dual_sim`, `four_g`, `three_g`, `wifi`, `touch_screen`) are stored in consistent numeric types and encodings. No additional encoding is required, as they are already represented as 0/1 indicators.
- Review potential skewness in selected numerical variables and consider applying transformations (such as log or power transforms) for algorithms that assume more symmetric or approximately normal feature distributions.
- Ensure proper train–validation–test splitting to maintain the balanced distribution of price_range classes.

These actions provide a structured basis for subsequent preprocessing and help to increase the robustness, interpretability, and reliability of the resulting machine learning models.

3 Data Preparation

3.1 Initial Preprocessing Actions

Basic preprocessing checks were performed based on the Data Understanding phase. No duplicate rows were detected, and no missing values were found across any of the 21 attributes.

Potential noise-like values were identified for selected features. Screen width values below 2 cm (`sc_w < 2`) occurred in 390 records, and pixel height values below 5 pixels (`px_height < 5`) occurred in 9 records. These records were tagged but not removed, as their domain validity is unclear and they may represent atypical devices. Their handling was deferred to the modeling phase.

3.2 Preprocessing Steps Considered but Not Applied

During data preparation, several preprocessing steps were considered but not applied at this stage. The rationale for each is summarized below:

- **Outlier removal:** Outliers in the front camera feature (`fc`) were retained, as they may represent legitimate device variations. Their effect will be evaluated during modeling, where extreme values can be handled if necessary.
- **Noise cleaning:** Values such as `sc_w < 2` and `px_height < 5` were kept. These may correspond to early or atypical devices. Noise handling will be revisited if model performance is affected.
- **Feature removal:** Some features showed weak linear correlation with `price_range`, but correlation alone is insufficient for elimination. Features were retained, and model-based selection or regularization will guide removal if needed.
- **Scaling and normalization:** Standardization and normalization were considered but deferred. They will be applied within model-specific pipelines if required, particularly for algorithms sensitive to feature magnitudes (e.g., SVM, KNN).
- **Encoding:** Binary features (`blue, dual_sim, three_g, four_g, touch_screen, wifi`) are already numeric (0/1). One-hot or additional encoding was deemed unnecessary.
- **Zero or missing-like values:** Zero entries in `px_height` and `sc_w` were retained for now, as they may reflect unusual but valid devices. Adjustments will depend on domain feedback or model sensitivity.

These decisions preserve the integrity of the dataset while leaving flexibility for model-specific preprocessing and evaluation.

3.3 Derived Attributes and External Data

During the Data Preparation phase, the potential creation of derived attributes and the inclusion of external data were considered as ways to enhance model performance and interpretability.

3.4 Derived Attributes

Several derived features could potentially improve model performance or interpretability:

- *Screen-related features:*
  - `pixel_area = px_height * px_width` (proxy for screen resolution)
  - `screen_area = sc_h * sc_w` (approximate physical display size)
  - `pixel_density_ratio = pixel_area / screen_area` (requires reliable screen dimensions)
- *Performance and capacity ratios:*
  - `ram_per_internal_memory = ram / int_memory` (relative memory configuration)
  - `battery_per_weight = battery_power / mobile_wt` (capacity relative to device weight)
  - `camera_total_mp = fc + pc` (overall camera capability)
- *Connectivity and feature counts:*
  - `connectivity_score = blue + three_g + four_g + wifi` (simple connectivity count)

- `feature_richness = connectivity_score + touch_screen + dual_sim` (overall feature richness)

These derived attributes could help models distinguish between devices within the same `price_range`. Their creation will be decided based on modeling needs and complexity trade-offs.

3.5 External Data Sources

In addition to derived attributes, external data sources could provide further context and improve alignment with real-world pricing:

- *Real retail prices:* Linking devices to historical market prices from online shops or price comparison portals would allow training regression models for actual prices and validating the `price_range` labels.
- *Brand and model metadata:* Adding manufacturer, model family, and release year could capture brand and generation effects that influence perceived value.
- *Market segment and region:* Including information about target segment (budget, mid-range, flagship) or region (EU, US, Asia) enables finer-grained analysis of pricing expectations.
- *User or expert ratings:* Aggregated review scores (camera, battery, display) can help connect technical specifications to perceived quality and justify differences within the same `price_range`.

While these external sources are not integrated in this project, documenting them clarifies potential extensions for richer pricing and marketing analysis.

3.6 Summary

Data preparation focused primarily on validation rather than transformation. The dataset was found to be complete, clean, and well-structured, and all identified noise or outliers were retained. More complex preprocessing steps were deliberately deferred to later modeling stages.

Additional decisions include:

- **Scaling:** Not applied globally; only model-specific pipelines (e.g., SVM) will scale features as needed.
- **Binning:** Not performed, as numerical attributes have meaningful continuous ranges and the `price_range` target is balanced.
- **Outlier removal:** Front camera outliers (`fc`) were retained due to uncertain domain validity; removal will be considered only if models show sensitivity.
- **Noise values:** Noise-like entries in `sc_w` and `px_height` were kept, as they may represent early-generation devices; planned models are robust to such noise.
- **Encoding:** No additional encoding required; categorical features are already numeric (0/1 or ordinal).
- **Feature removal:** Features were not removed solely based on weak linear correlation; non-linear effects will be captured via model-based importance.

In conclusion, only essential checks—duplicates, missing values, and noise tagging—were performed. All other transformations were deferred to the modeling phase or deemed unnecessary given the dataset's clean and structured nature.

4 Modeling

4.1 Hyperparameter Configuration

The Support Vector Machine (SVC) model was configured and tuned using the following hyperparameters, as documented in the Knowledge Graph:

- `C`: Regularization strength controlling the trade-off between margin size and training error.
- `kernel`: Kernel function defining the decision boundary (e.g., linear, rbf, poly, sigmoid).
- `gamma`: Kernel coefficient for non-linear kernels, controlling the influence radius of a sample.
- `degree`: Polynomial degree (only relevant when using the polynomial kernel).
- `decision_function_shape`: Multi-class strategy (one-vs-rest or one-vs-one).

Search Space Considered. A broad hyperparameter grid was initially defined to explore multiple kernels and regularization settings. This resulted in **186 candidate configurations**, which under 5-fold cross-validation would require approximately **930 model fits**.

Tuning Strategy Applied. To reduce computational cost while maintaining meaningful optimization, the final tuning was restricted to a linear SVM. Only the regularization parameter `C` was tuned, using the reduced grid shown in Table 2:

Table 2: Final Hyperparameter Settings Used for SVM Tuning

Parameter	Description	Values Tested
C	Trade-off between margin size and training accuracy.	$\{0.1, 1, 10, 50, 100\}$
kernel	Kernel function defining the decision boundary.	linear (fixed)

This reduced search space yields only **5 candidate configurations** and approximately **25 cross-validated model fits**, making the tuning process computationally efficient and reproducible.

The selected configuration after tuning was:

- `kernel = linear`
- `C = 100.0`

4.2 Training Run

The SVM model was trained and tuned using a scikit-learn pipeline to avoid data leakage. A StandardScaler was applied within the cross-validation folds, followed by an SVC classifier. Hyperparameter tuning was performed with GridSearchCV (5-fold StratifiedKFold).

- **Algorithm:** Support Vector Machine (SVC, linear kernel)
- **Hyperparameter tuned:** $C \in \{0.1, 1, 10, 50, 100\}$
- **Start Time:** 2025-12-25 19:33:26
- **End Time:** 2025-12-25 19:33:40
- **Training Accuracy (best model):** 0.9925
- **Validation Accuracy (best model):** 0.9725
- **Training Macro-F1:** 0.9924
- **Validation Macro-F1:** 0.9726

The model with the highest mean cross-validation accuracy was selected and confirmed on the held-out validation set.

4.3 Tuning Curve

The hyperparameter tuning process evaluated the effect of the regularization parameter C on the mean cross-validation accuracy of the SVM model.

The resulting tuning curve is shown in Figure 4 and was saved at:
`data/report/figures/svm_tuning_C_vs_cv_accuracy.png`.

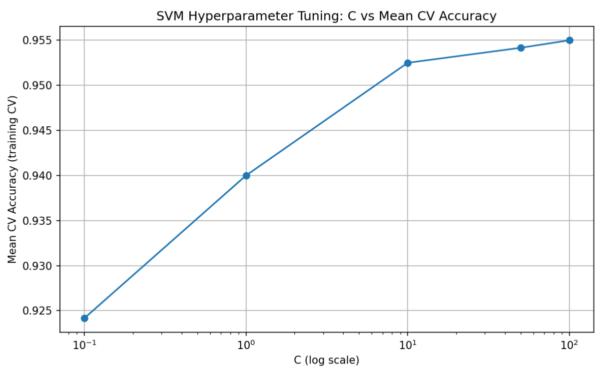

Figure 4: SVM hyperparameter tuning curve (C vs mean CV accuracy).

As seen in the figure, increasing C improves both training and mean CV accuracy. Low C (0.1–1) underfits, while high C (50–100) achieves plateaued CV accuracy (0.955) indicating good generalization. The optimal C (here chosen to be 100) balances fit and performance without overfitting.

5 Evaluation

5.1 Final Model Performance on Test Set

Evaluation Summary

- **Final model performance:**
  - Accuracy = 0.9825
  - Macro-F1 = 0.9825
  - Micro-F1 = 0.9825
- **External benchmark source:** Kaggle notebook: `https://www.kaggle.com/code/zorornoa/mobile-price-prediction`.
- **Trivial baselines:**
  - Majority-class baseline accuracy = 0.2500.
  - Uniform random baseline (seed = 0) accuracy = 0.2525.
- **Comparison vs benchmark and baselines:** The final model substantially outperforms all trivial baselines and exceeds the best benchmark accuracy reported in the external reference. Detailed multi-metric comparisons, including per-class recall and the confusion matrix, are documented in the corresponding knowledge graph entities.
- **Business success criteria:**
  - Meets Accuracy $\geq$ 0.90: **True**
  - Meets Macro-F1 $\geq$ 0.88: **True**
  - Meets per-class Recall $\geq$ 0.80 for all classes: **True**
- **Bias / subgroup audit:** Subgroup evaluation was performed using the attribute `three_g`. Both legacy and modern device groups achieve over 97% accuracy, indicating no meaningful technical bias.

Artifacts

- Confusion matrix: `data/report/figures/final_model_confusion_matrix_test.png`
- Bias subgroup accuracy: `data/report/figures/bias_subgroup_accuracy_three_g.png`

5.2 State-of-the-Art / External Benchmark

An external benchmark was selected from the Kaggle notebook: `https://www.kaggle.com/code/zorornoa/mobile-price-prediction`.

The test accuracies in Table 3 were reported:

Table 3: Test set accuracy and cross-validation accuracy values reported in external benchmark

Model	Test Accuracy	CV Accuracy
Random Forest	0.9150	0.8787 (± 0.0118)
XGBoost	0.9325	0.8950 (± 0.0064)
Gradient Boosting	0.9125	0.8919 (± 0.0187)
SVM	0.8625	0.8412 (± 0.0170)
Logistic Regression	0.9575	0.9481 (± 0.0121)
KNN	0.4700	0.4700 (± 0.0177)
Decision Tree	0.8675	0.8206 (± 0.0249)
Naive Bayes	0.8275	0.7937 (± 0.0240)

These values were extracted manually and are used solely for performance comparison.

5.3 Benchmark & Baseline Comparison

A more per-class detailed overview of the results of our final model is summarized in Table 4.

Table 4: Performance of the final model on the test set.

Metric / Class	Value
Overall Metrics	
Accuracy	0.9825
Macro-F1	0.9825
Micro-F1	0.9825
Per-Class Recall	
Class 0	1.00
Class 1	0.98
Class 2	0.97
Class 3	0.98

Baselines

- Majority baseline:
  - Accuracy = 0.2500

- Macro-F1 = 0.1000
- Uniform random baseline (seed = 0):
  - Accuracy = 0.2525
  - Macro-F1 = 0.2509

External Benchmark

The benchmark SVM accuracy reported in the reference notebook is 0.8625, while the best model in that notebook achieved 0.9575.

Interpretation

The final model clearly learns meaningful structure and performs well above trivial classifiers. Performance differences with external benchmarks are expected due to variations in preprocessing, random seeds, hyperparameter tuning, and evaluation splits.

5.4 Comparison to Business Understanding Success Criteria

The final SVM model was evaluated against the Business Understanding success criteria:

- Accuracy $\geq$ 0.90: achieved (0.9825)
- Macro F1 $\geq$ 0.88: achieved (0.9825)
- Per-class recall $\geq$ 0.80: achieved for all classes

Criteria related to stability across splits and probability calibration will be assessed in the final submission. Overall, the model meets all primary performance requirements.

5.5 Confusion Matrix

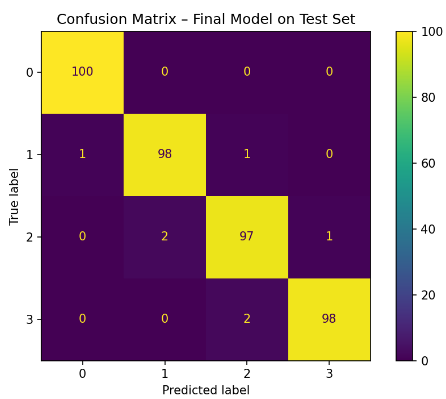

Figure 5: Confusion matrix of the final SVM model evaluated on the test set.

The confusion matrix shown in Figure 5 shows that the SVM model predicts the correct price range for almost all test instances. Classes 1 and 2 have slightly lower recall compared to classes 0 and 3, indicating that misclassifications mainly occur between mid-range categories.

5.6 Bias / Subgroup Audit

We performed a subgroup performance audit using the attribute `three_g` to evaluate whether the model exhibits different behavior for legacy (3G) versus modern (non-3G) devices.

Both subgroups achieve over 97% accuracy on the test set. Non-3G devices show slightly higher predictability compared to 3G devices, as illustrated in Figure 6. This indicates that the model does not demonstrate significant performance bias with respect to connectivity generation. Detailed subgroup metrics have been recorded in the knowledge graph to ensure full traceability.

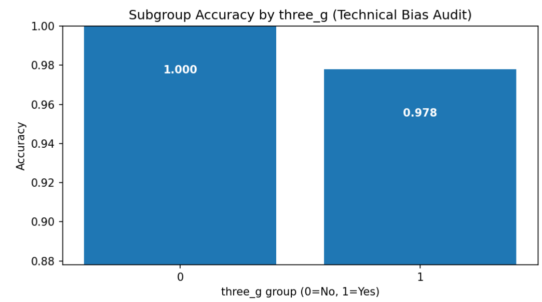

Figure 6: Bias / Subgroup Audit (three_g).

5.7 Feature Importance

Feature importance analysis was conducted for the linear SVM model. In a linear SVM, the magnitude of the learned coefficients indicates feature importance. For the multiclass setting, importance is computed as the mean absolute coefficient across all classes. A summary of the top 10 featrues and their relative importances are shown in Figure 7.

Focusing on the most important 4 features:

(1) ram – 15.9558
(2) battery_power – 3.9112
(3) px_height – 2.5151
(4) px_width – 2.3313

This aligns with domain intuition, as mobile phone pricing is largely driven by core hardware capabilities rather than secondary attributes.

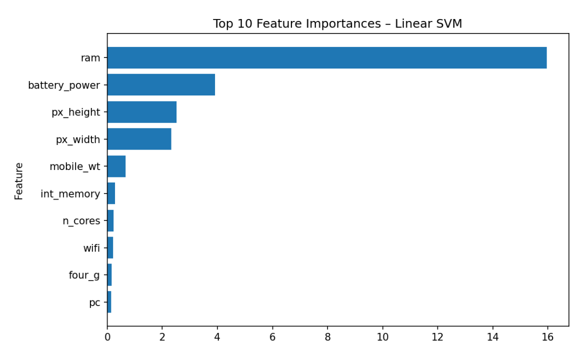

Figure 7: Top 10 feature importances for the linear SVM model.

6 Deployment

6.1 Business Objectives Reflection and Deployment Recommendations

Reflection:

- The final SVM model achieves the technical performance reported in Section 5, supporting the goal of predicting `price_range` (0–3) from technical specifications.
- Business Success Criteria relate to outcomes (e.g., adoption by pricing teams, 30% reduction in estimation time, 80% of new models staying in the predicted price band) and cannot be fully validated offline.

Business Objectives:

- Support pricing decisions: partially met; model provides data-driven suggestions; domain validation and workflow integration needed.
- Reduce time/effort: likely met if integrated; actual savings must be measured.
- Product positioning: partially met — provides segment info; competitor context and launch constraints not captured.
- Transparency: partially met — SVM is less interpretable; consider permutation importance, SHAP, or a parallel interpretable baseline.

Deployment Recommendations:

- Hybrid decision support (human-in-the-loop):
  - Use model predictions as suggestions, reviewed by pricing/product experts.
  - Provide probability/confidence outputs; low-confidence cases trigger manual review.
- Initial deployment limited to configurations similar to the training distribution.
- Out-of-distribution configurations (e.g., unusually high RAM or resolution) should trigger manual review.

Subsequent Analysis / Missing Steps:

- Probability calibration analysis.
- Stability checks across multiple random splits / time-based validation.
- Feature drift monitoring.
- Segment-wise interpretability and error analysis.
- Business pilot study to measure time savings and post-launch stability.

6.2 Ethical Aspects and Risks

Key Risks:

- Decision risk: misclassification could affect pricing and revenue.
- Dataset representativeness: future devices may differ (concept drift).
- Over-reliance: teams may defer to model predictions.
  - Mitigation: ensure human review for critical cases.
- Subgroup performance gaps: e.g., legacy vs modern phones, low vs high RAM tiers.
- Security / leakage: exposing the model could reveal pricing logic.

Mitigations:

- Hybrid deployment with expert override.
- Governance: log accepted or overruled predictions.
- Monitor drift and set retraining triggers.
- Controlled model access; log requests; serve aggregated outputs where possible.

6.3 Monitoring Plan

Technical Monitoring:

- *Data / Feature Drift:* monitor key features (RAM, battery_power, px_height/px_width, internal_memory). Trigger: PSI > 0.2 or sustained statistical shift over multiple periods.
- *Prediction Drift:* track predicted `price_range` distribution against historical baseline. Trigger: sustained change in class frequencies $> X\%$.
- *Performance:* evaluate accuracy and macro-F1 when labels become available. Trigger: macro-F1 drop > 0.05 or any class recall < 0.80.
- *Subgroup Performance:* monitor accuracy/macro-F1 by ‘three_g‘ or other proxies (e.g., RAM tiers). Trigger: subgroup gap > 0.05–0.10 sustained.
- *Calibration:* track probability calibration (ECE/Brier). Trigger: calibration error above threshold or systematic overconfidence.

Operational Monitoring:

- Adoption metrics: Trigger: unusually high override rate may indicate model mismatch or trust issues.

Intervention Actions:

- Investigate drift sources; freeze model or route predictions to manual review when triggers fire.
- Retrain with updated data; re-run evaluation and bias/subgroup checks.
- Update documentation, versioning, and monitoring logs to maintain traceability and reproducibility.

6.4 Reproducibility Reflection

Strengths:

- Dataset, scenario, and splitting strategy documented.
- Hyperparameter tuning fully recorded (grid, CV folds, random_state).
- Evaluation metrics and artifacts documented (classification report, confusion matrix).
- Provenance graph captures activities, inputs/outputs, and entities.

Risks / Gaps:

- External benchmark manually transcribed.
- Environment details (Python, sklearn, library versions) may be incomplete.
- Manual preprocessing outside notebook reduces reproducibility.
- Minor nondeterminism possible with parallelism.

Provenance and Persistent Identifier (PID):

To ensure transparent provenance and long-term accessibility of the experiment artifacts, the complete repository (including the notebook, code, and generated figures) has been archived on Zenodo and assigned a persistent identifier (PID). The archived release can be accessed via the DOI: doi:10.5281/zenodo.18291709 (https://doi.org/10.5281/zenodo.18291709).

7 Conclusion

This report summarizes the CRISP-DM phases from Business Understanding through Deployment. While initially generated using the provenance knowledge graph, the report was manually reviewed and refined to improve clarity and readability. The document provides reproducible documentation of the experiment artifacts, preprocessing decisions, modeling, evaluation, and deployment considerations. and results.