Comprehensive Summary Report for Mobile Games Sales Prediction Project

1 Overview

This report summarizes the machine learning pipeline implemented in the `Project-1.ipynb` Jupyter notebook to predict total sales (`sales_total`) for mobile games using a dataset (`mobile games data.csv`) with 16,598 entries. The dataset includes features like `title_name`, `device_type`, `launch_year`, `game_genre`, `publisher_name`, and regional sales (`sales_usa`, `sales_europe`, `sales_asia`, `sales_misc`). The pipeline encompasses data loading, exploration, preprocessing, feature engineering, model training, evaluation, and deployment, with visualization used to interpret results.

2 Data Loading and Initial Exploration

- **Purpose**: Load and understand the dataset's structure and characteristics.
- **Actions**:
  - Loaded the dataset using `pandas.read_csv` from `d:/gradious/mobile games data.csv`.
  - Inspected the first (`df.head()`) and last (`df.tail()`) five rows to preview data.
  - Used `df.info()` to check data types and missing values: 16,598 entries, 10 columns, with 271 missing values in `launch_year` and 58 in `publisher_name`.
  - Summarized numerical features with `df.describe()`: Revealed skewed sales data (e.g., `sales_total` mean: 0.5374, max: 82.74).
  - Analyzed categorical features with `df.describe(include='object')`: Identified high cardinality in `title_name` (11,493 unique) and dominant categories (e.g., `game_genre`: Action, `publisher_name`: Electronic Arts).
- **Key Insights**:
  - *Numerical*: Sales columns are skewed, with outliers (e.g., `sales_usa` max: 41.49, 75th percentile: 0.24).

- *Categorical*: device_type (31 unique), game_genre (12 unique), and publisher_name (578 unique) suggest encoding needs.

- **Visualization Need**:
  - *Histograms*: Plot distributions of numerical features (launch_year, sales columns) to visualize skewness and outliers.
  - *Bar Plots*: Show frequency of top categories for device_type, game_genre, and publisher_name.

3 Data Preprocessing

- **Purpose**: Clean and prepare data for modeling.
- **Actions**:
  - *Imputation*:
    * *Numerical*: Missing launch_year values (271) imputed using SimpleImputer with a mean or median strategy.
    * *Categorical*: Missing publisher_name values (58) imputed with the most frequent category.
  - *Scaling*: Applied RobustScaler to numerical features to handle outliers.
  - *Encoding*: Used OneHotEncoder to convert categorical features (device_type, game_genre, publisher_name) into binary columns.
  - *Feature Selection*:
    * VarianceThreshold: Removed low-variance features to eliminate noise.
    * SelectKBest with f_regression: Selected top features correlated with sales_total.
- **Key Insights**:
  - Robust handling of missing values and outliers ensures model stability.
  - Encoding high-cardinality categoricals increases feature count, requiring careful selection.
- **Visualization Need**:
  - *Box Plots*: Display pre- and post-scaling distributions of numerical features.
  - *Correlation Heatmap*: Visualize relationships between numerical features and sales_total.

4 Model Creation

- **Purpose**: Build and train regression models to predict `sales_total`.
- **Actions**:
  - *Pipeline*: Used `Pipeline` and `ColumnTransformer` to combine preprocessing (imputation, scaling, encoding) and modeling.
  - *Models*:
    - *RandomForestRegressor*:
      - `n_estimators`: 200
      - `max_depth`: None
      - `min_samples_split`: 5
      - `max_features`: 0.8
    - *GradientBoostingRegressor*:
      - `n_estimators`: 300
      - `learning_rate`: 0.1
      - `max_depth`: 5
      - `subsample`: 1.0
    - *XGBRegressor*:
      - `n_estimators`: 200
      - `learning_rate`: 0.1
      - `max_depth`: 5
      - `subsample`: 1.0
      - `colsample_bytree`: 0.8
    - *LGBMRegressor*:
      - `n_estimators`: 300
      - `learning_rate`: 0.05
      - `max_depth`: 7
      - `num_leaves`: 31
      - `subsample`: 0.8
      - `colsample_bytree`: 0.8
    - *VotingRegressor*: Ensemble of LightGBM, GradientBoosting, and XGBoost with above parameters.
  - *Tuning*: Used `RandomizedSearchCV` to optimize hyperparameters.

  - *Train-Test Split*: Split data via `train_test_split` (e.g., 80:20) for training and evaluation.
- **Key Insights**:
  - Ensemble models handle non-linear relationships and mixed data types well.
  - Tuning balances bias, variance, and computational efficiency.
- **Visualization Need**:
  - *Learning Curves*: Plot training and validation errors vs. training size to assess model fit.

5 Model Evaluation

- **Purpose**: Assess model performance and select the best model.
- **Actions**:
  - *Metrics*:
    - `mean_squared_error`: Computed RMSE to measure prediction error.
    - `mean_absolute_error`: Assessed average error magnitude.
    - `r2_score`: Evaluated variance explained by the model.
  - *Cross-Validation*: Used `cross_val_score` for robust performance estimates.
  - *Results*:
    - *RandomForestRegressor*:
      - RMSE: 0.0599
      - MAE: 0.0230
      - R^2: 0.9671
    - *GradientBoostingRegressor*:
      - RMSE: 0.0555
      - MAE: 0.0228
      - R^2: 0.9717
    - *XGBRegressor*:

6 Feature Importance Analysis

- **Purpose**: Identify key predictors of `sales_total` for interpretability.
- **Actions**:
- Extracted feature importances from models supporting it (RandomForest, GradientBoosting, XGBoost, LightGBM).
- Combined numerical features (`launch_year`, sales columns) and one-hot encoded categorical features.
- Created a DataFrame of top 15 features by importance.
- Plotted bar charts for each model using `seaborn.barplot` in a 2×3 subplot grid (figsize: 15×12).
- Saved visualization as `feature_importances.png` (DPI: 300).
- **Key Insights**:
- Likely key features: Regional sales (e.g., `sales_usa`), `launch_year`, and certain `game_genre` or `device_type` categories.
- Helps understand drivers of sales for business insights.
- **Visualization Need**:
- *Bar Plots*: Already implemented—bar plots of top 15 features per model (saved as `feature_importances.png`).

7 Sample Predictions

The VotingRegressor's predictions on a test set sample (first 10 instances) are shown below, with actual values and percentage errors:

- Sample 1: Predicted: 0.116024 | Actual: 0.122218 | Error: 5.1%
- Sample 2: Predicted: 0.914119 | Actual: 0.924259 | Error: 1.1%
- Sample 3: Predicted: 0.133093 | Actual: 0.122218 | Error: 8.9%
- Sample 4: Predicted: 0.858542 | Actual: 0.854415 | Error: 0.5%
- Sample 5: Predicted: 0.160110 | Actual: 0.157004 | Error: 2.0%
- Sample 6: Predicted: 0.037774 | Actual: 0.019803 | Error: 90.8%

- Sample 7: Predicted: 0.799278 | Actual: 0.756122 | Error: 5.7%
- Sample 8: Predicted: 0.505743 | Actual: 0.500775 | Error: 1.0%
- Sample 9: Predicted: 0.162748 | Actual: 0.157004 | Error: 3.7%
- Sample 10: Predicted: 0.286073 | Actual: 0.285179 | Error: 0.3%

Prediction Statistics:

- Min: 0.037774
- Max: 0.914119
- Mean: 0.397350
- Std: 0.324810

Actual Values Statistics:

- Min: 0.019803
- Max: 0.924259
- Mean: 0.389900
- Std: 0.341151

8 Final Model Deployment

- **Purpose**: Finalize and deploy the best model for use.
- **Actions**:
- Selected VotingRegressor as the final model (RMSE: 0.0538).
- Stored in the `results` dictionary, accessed as `results['Ensemble']['mo`
- Printed confirmation: ”Deployed Model: Ensemble with RMSE: 0.0538”.
- Likely saved the model using `joblib` for future predictions.
- **Key Insights**:
- The VotingRegressor’s robustness and low error make it suitable for deployment.
- The model can predict sales for new games based on input features.

9 Conclusion

- **Summary**:
- The pipeline effectively processed a dataset of 16,598 mobile games, handling missing values, outliers, and categorical features.
- Five models were trained, with the VotingRegressor (ensemble of LightGBM, GradientBoosting, and XGBoost) achieving the best performance (RMSE: 0.0538, MAE: 0.0211, R^2: 0.9734).
- Feature importance highlighted key predictors, aiding interpretability.
- **Strengths**:
- Robust preprocessing and ensemble models handled skewed data and complex relationships.
- Cross-validation and tuning ensured reliable performance.
- **Limitations**:
- High cardinality in `title_name` may complicate modeling if included.
- Potential overfitting in cases with high error (e.g., Sample 6: 90.8% error).
- **Visualization Summary**:
- *Data Exploration*: Histograms (numerical distributions), bar plots (categorical frequencies).
- *Preprocessing*: Box plots (pre/post-scaling distributions), correlation heatmap (feature-target relationships).
- *Model Evaluation*: Bar chart (RMSE, MAE, R^2 comparison), scatter plot (prediction vs. actual).
- *Feature Importance*: Bar plots of top 15 features per model (saved as `feature_importances.png`).
- **Final Model**: VotingRegressor.