Video Game Data Analysis

Reem Abdelaziz 1949677
Benjamin Truong 1965196
Miguel Garcia 1887771
Guillermo Martinez Somoza 1769385
Devin Stockton 2068216

Introduction (Guillermo Martinez Somoza)

Background: This chosen dataset for our topic is a video game dataset. The motivation behind selecting this specific dataset is deeply rooted in our shared passion for video games. Given the widespread interest in video games and the accompanying cultural phenomena, delving into this dataset offers an exciting opportunity to explore a subject that captivates a broad audience.

Research Question: The primary objective of our project is to determine the whether a game is successful through the variable global_sales. Based on the predictors: publisher, genre, critic_score, user_score, and rating, we can find how impactful attributes of a game are to the success of the game.

Dataset Description: Our current project revolves around a rich dataset concerning the sales of video games, sourced from Kaggle.com in the following link: VideoGameDataset. In terms of specifics, the dataset boasts an impressive magnitude, encompassing a total of around 6,900 individual cases. These cases are meticulously characterized by 17 unique variables, each offering a distinct perspective into the world of successful video games. These variables encompass fundamental details such as the game's title, the platform it is designed for, the pivotal year of its public release, and its designated genre. Furthermore, the dataset provides insights into the commercial success of these games by incorporating sales data from major regions: North America, the European Union, and Japan, culminating in a global sales metric. On the critical reception side, the dataset includes both aggregated critic scores and counts, as well as user-generated scores and corresponding user review counts, providing a comprehensive view of how these games were perceived, ultimately offering insight on what drives higher global sales. Because global sales was a continuous variable, we opted to represent it as the categorical variable GS_Category. By splitting the original global sales by quantile into 3 roughly equal parts, we generated the new variable with 3 levels: low, medium, high.

- **Name**: The name or title of the video game.
- **Platform**: The gaming platform or console on which the game is played
- **Year_of_Release**: The year in which the game was released to the public.
- **Genre**: The category or type of the game based on its gameplay characteristics.
- **Genre_factor**: A factor related to the game's genre.
- **Publisher**: The company or entity responsible for publishing and distributing the game.
- **NA_Sales**: Number of sales in North America.
- **EU_Sales**: Number of sales in the European Union.
- **JP_Sales**: Number of sales in Japan.
- **Other_Sales**: Number of sales in other regions.
- **Global_Sales**: Number of sales of the game worldwide.
- **Critic_Score**: The aggregated score given to the game by critics or reviewers.
- **Critic_Count**: Number of critic reviews contributing to the critic score.
- **User_Score**: The user or player assigned score or rating for the game.

- **User_Count**: The number of user reviews contributing to the user score.
- **Developer**: The company or individuals responsible for developing the game.
- **Rating**: The age or content rating assigned to the game.
- **GS_Category**: A classification or category associated with the game.

Methodology (Reem Abelaziz)

Support Vector Machines

Overview: Support Vector Machines (SVM) are powerful supervised learning models used for classification and regression. They aim to find an optimal hyperplane that separates data into different classes while maximizing the margin between them. SVM works by mapping data into higher-dimensional spaces to facilitate separation and then identifying the hyperplane that best divides the classes. It's effective in high-dimensional spaces and offers versatility through different kernel functions for complex data. However, SVM might struggle with large datasets due to computational complexity, and kernel choice impacts performance. Regularization helps control overfitting in SVM.

Model Formula:
Linear Kernel: The linear kernel is applied when the data is capable of being separated by a single line, indicating linear separability.

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{p} x_{ik} x_{jk} \text{ - linear kernel}$$

Polynomial Kernel: The 'degree' parameter specifies a value 'd'. When fitting an SVM with a polynomial kernel (using kernel = "polynomial"), it generates highly flexible boundary shapes.

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \langle \mathbf{x}_i, \mathbf{x}_j \rangle)^d = (1 + \sum_{k=1}^{p} x_{ik} x_{jk})^d \text{ - polynomial kernel}$$

Radial Kernel: The radial kernel showcases local behavior by relying on nearby training data to determine the class label for a test observation. Gamma, denoted as γ, defines the spread of the kernel, influencing the decision region.

$$K(\mathbf{x}_i, \mathbf{x}_j) = exp(-\gamma \sum_{k=1}^{p} (x_{ik} - x_{jk})^2), \gamma > 0 \text{ - radial kernel.}$$

Advantages: Support Vector Machines (SVM) excel in finding optimal hyperplanes for data separation, even in high-dimensional spaces. Their ability to handle complex datasets through various kernel functions is advantageous. SVMs are effective for both linear and nonlinear classification tasks, offering robustness against overfitting when regularization is applied.

Disadvantages: Support Vector Machines (SVM) face challenges with larger datasets due to higher computational demands. Choosing the appropriate kernel and regularization parameters significantly impacts their performance. SVMs might also struggle with noisy datasets and require proper scaling of features for optimal results.

K-Nearest Neighbors

Overview: K-Nearest Neighbors (KNN) is a versatile machine learning method used for classification and regression tasks. It makes predictions based on the majority class or average value of its closest data points. The 'K' represents the number of neighbors considered for decision-making. KNN calculates distances between points, assigns the class based on the most common neighbors, and is simple to implement. However, it can be sensitive to outliers and requires careful selection of the optimal K value for accurate predictions.

Model Formula: To employ k-Nearest Neighbors, we initially identify the K closest points to x0 within the training data (referred to as N0). We estimate the conditional probability for class j, denoted as $Pr(Y = j \mid X = x0)$, by computing the proportion of points in N0 where the class equals j. In mathematical notation: $Pr(Y = j \mid X = x0) = 1/K * \Sigma(i \in N0) I(yi = j)$. Subsequently, we assign the observation to the class with the highest estimated probability, determined by $Cˆknn(x0) = j$ if $pˆj(x0) = max\{pˆ1(x0), ..., pˆJ (x0)\}$. The flexibility of this method depends on the chosen k value: lower k values offer greater flexibility, while larger k values tend to reduce flexibility.

Advantages: K-Nearest Neighbors (KNN) stands out for its simplicity, versatility across data types, and effectiveness with smaller datasets. It adapts well to varied classification or regression scenarios without presuming data distributions, making it suitable for irregular decision boundaries. Plus, it requires minimal training and operates for both classification and regression tasks without complex parameter tuning.

Disadvantages: K-Nearest Neighbors (KNN) struggles with larger datasets due to high computational requirements. It's sensitive to outliers and can be influenced by irrelevant features. Moreover, selecting the optimal K value significantly impacts its performance, requiring careful tuning.

Data Analysis (Benjamin Truong)

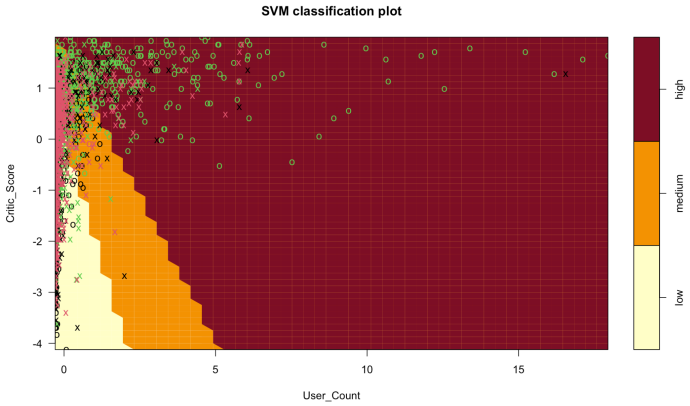
Support Vector Machines

Thought Process: Prior to applying SVM on the data, we initiated the model fitting process. Firstly, we eliminated all null values and converted character-based entries into numerical values

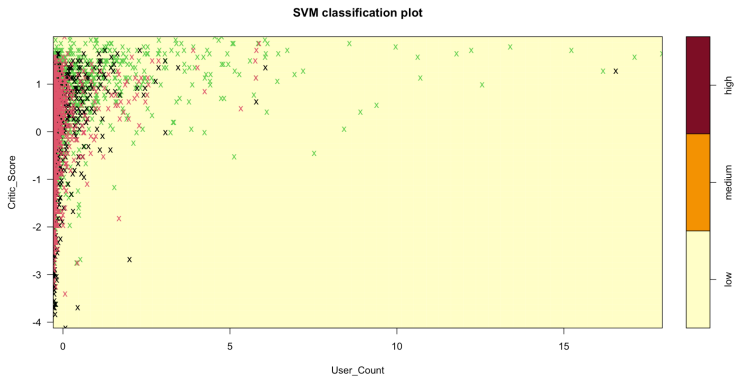
to accommodate SVM's requirement for numerical data. Subsequently, we pruned columns deemed irrelevant for this model, such as an extraneous number row, Name, Genre, NA_Sales, EU_Sales, JP_Sales, and Other_Sales. Ensuring standardized values, we then scaled the dataset for consistency. Noteworthy predictors retained for consideration encompass Platform, Year_of_release_Genre_factor, Publisher, Global_Sales, Critic_Score, Critic_Count, User_Score, User_Count, Developer, Rating, and GS_Category. We will then proceed with the SVM process.

Model Evaluation: We split the complete dataset, allocating 80% for training and 20% for testing, using a fixed seed value of 1. We evaluated the performance of each SVM model using both training and testing datasets, emphasizing accuracy, confusion matrices, and training error.

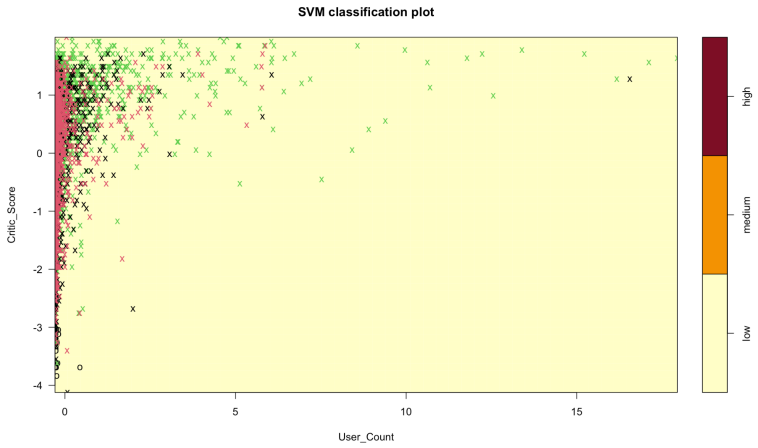
1. **Linear Kernel**:
    a. Best Cost Parameter: 0.5
    b. Training Error: 0.261472
    c. Testing Error: 0.3798561
    d. Accuracy: 0.6201439
    e.



2. **Radial Basis Function Kernel**:
    a. Best Cost Parameter: 10
    b. Best Gamma: 2
    c. Training Error: 0
    d. Testing Error: 0.5985612
    e. Accuracy: 0.4014388



3. **Polynomial Kernel**:
    a. Best Cost Parameter: 0.01
    b. Best Degree: 3
    c. Training Error: 0.6609681
    d. Testing Error: 0.6482014
    e. Accuracy: 0.3517986



After splitting the dataset, we trained the model with 80% of the data and used the remaining 20% to test our model using the different kernels. The linear kernel, optimized with a cost parameter of 0.5, exhibits a training error of 0.261472 and a testing error of 0.3798561. Resulting with an accuracy of 0.6201. This kernel performs relatively well with moderate generalization to unseen data, showcasing lower testing error compared to other kernels evaluated in this context.

K-Nearest Neighbors

Thought Process: Prior to applying KNN on the data, we initiated the model fitting process. Firstly, we eliminated all null values and converted character-based entries into numerical values to accommodate KNN's requirement for numerical data. Subsequently, we pruned columns deemed irrelevant for this model, such as an additional number row, Name, Genre, and

Sales_Success. Ensuring standardized values, we then scaled the dataset for consistency, because knn uses distance calculations, it is crucial to scale these values. Noteworthy predictors retained for consideration encompass Platform, Year_of_release_Genre_factor, Publisher, NA_Sales, EU_Sales, JP_Sales, Global_Sales, Critic_Score, Critic_Count, User_Score, User_Count, Developer, Rating, and GS_Category. We will then proceed with the KNN process.

Model Evaluation: We split the complete dataset, allocating 80% for training and 20% for testing, using a fixed seed value of 1.

After splitting the dataset, we trained the model with 80% of the data and used the remaining 20% to test our model using various K values. The optimal K outcome found was 1, yielding a testing error of 0.4697842.

```
K:  1
error:  0.4697842
K:  3
error:  0.5014388
K:  5
error:  0.5201439
K:  8
error:  0.5302158
K:  10
error:  0.5482014
K:  15
error:  0.5388489
K:  20
error:  0.5489209
```

After evaluating the model with the most optimal K of 1, the KNN's accuracy stands at 0.5302, with a 95% confidence interval between 0.5036 and 0.5567. The 'No Information Rate' (NIR) is 0.3518, and the p-value (acc > NIR) is less than 2e-16, indicating statistical significance. The Kappa statistic, measuring agreement beyond chance, is 0.2958. Additionally, the Mcemar's Test P-value is 0.02738, suggesting a significant difference between the observed and expected classification frequencies. Overall, the model performs moderately better than chance, as indicated by the Kappa value and accuracy above the No Information Rate, but there's still room for improvement.

```
Overall Statistics

               Accuracy : 0.5302
                 95% CI : (0.5036, 0.5567)
    No Information Rate : 0.3518
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.2958

 Mcnemar's Test P-Value : 0.02738
```

Interpretations

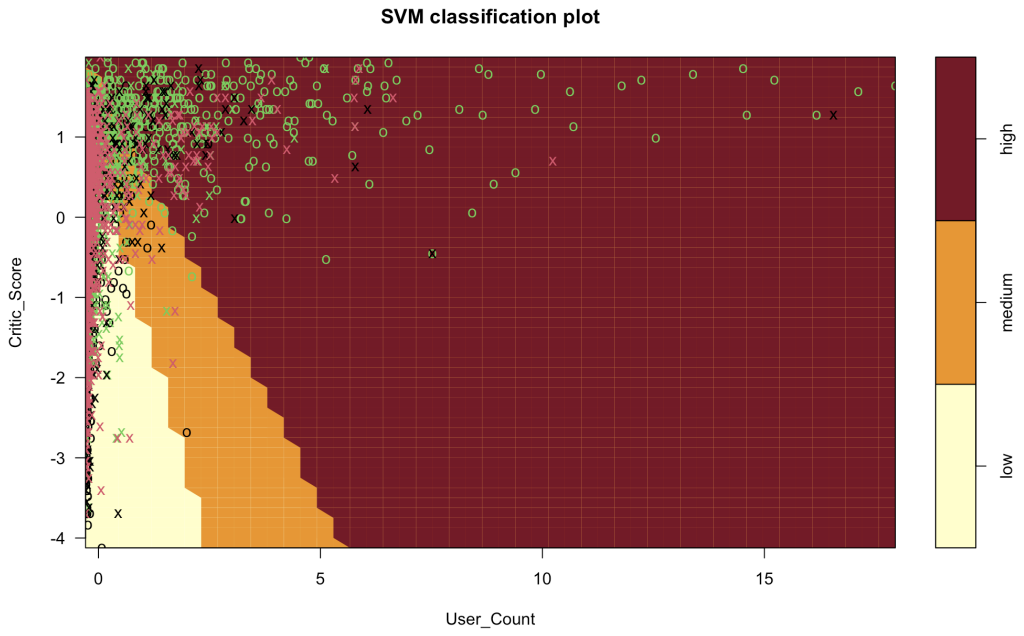
Model Comparison:

1. SVM: Using the linear kernel with a cost parameter of 0.5, SVM achieves an accuracy of 0.6201, with a training error of 0.261472 and a testing error of 0.3798561. This kernel demonstrates moderate generalization to unseen data, displaying lower testing error compared to other evaluated kernels.
2. KNN: KNN attains an accuracy of 0.5302, surpassing the No Information Rate (NIR) of 0.3518 significantly. It exhibits statistical significance (p-value < 2e-16) and a Kappa statistic of 0.2958, implying moderate performance above chance, yet with potential for improvement.

SVM with the linear kernel presents higher accuracy compared to KNN. KNN shows statistical significance and a moderate Kappa statistic, indicating performance above chance, but slightly lower accuracy compared to SVM. Therefore, considering accuracy as the main criterion and based on the provided data, the SVM with the linear kernel may be the better model for our dataset.

Testing Best Model: Having identified the SVM with the linear kernel as the optimal model, we applied this model to the entire dataset without performing a data split.

1. Best Cost Parameter: 0.5
2. Testing error: 0.2684612
3. Accuracy: 0.7315388



	train.pred		
	low	medium	high
low	1846	397	130
medium	401	1432	444
high	144	349	1804

Interpretations: The results indicate that the SVM with the linear kernel, identified as the most optimal model, achieved an overall performance accuracy of 0.7315388. This accuracy suggests a reasonably respectable predictive performance for the model. The model's ability to predict outcomes aligns with the original goal or question, demonstrating a satisfactory level of accuracy in its predictions.

Conclusion (Miguel Garcia)

Our project's main goal is to predict a video game's success based on the variable global_sales. The SVM with a linear kernel emerged as the best model, achieving an accuracy of 0.7315388. This level of accuracy indicates a respectable predictive capability, aligning well with our project's objective.

However, we encountered challenges due to hardware limitations, particularly during the computationally intensive task of tuning hyperparameters for Support Vector Machines.

To enhance data analysis, we can explore more sophisticated models to capture intricate relationships, optimize performance through fine-tuning via grid or randomized search, address dataset imbalances, and manage missing data using strategies like imputation or models handling missing values effectively.

References (Devin Stockton)

DevDocs — R documentation. (n.d.). Devdocs.io. Retrieved December 5, 2023, from https://devdocs.io/r/

Le, James. "Support Vector Machines in R Tutorial." *DataCamp*, DataCamp, www.datacamp.com/tutorial/support-vector-machines-r.

Ray, S. (2023a, April 27). *Learn how to use support vector machines (SVM) for Data Science*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

Shafi, A. (2023, February). *KNN Classification Tutorial using Sklearn Python*. Www.datacamp.com. https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn

SID_TWR. (2019, May 10). *Video games sales dataset*. Kaggle. https://www.kaggle.com/datasets/sidtwr/videogames-sales-dataset?select=Video_Games_Sales_as_at_22_Dec_2016.csv

Srivastava, T. (2019, March 7). *Introduction to KNN, K-Nearest Neighbors : Simplified*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

Appendix (Devin Stockton)

R Code (Reem Abdelaziz)

https://github.com/bngei/Video_Game_Data_Analysis/blob/main/Video_Game.R