Video Games Sales Prediction

By: Ahmad Khaled Jaber

Summary

The goal of this project is to predict the sales of video games by using machine learning. We have a data set that contains the features of different video games and their global sales. This problem falls under supervised learning and it's a regression problem as we want to predict a numerical value. We analyzed the dataset, preprocessed it, and prepared it for training. We have different categorical features which we encoded using different encoding methods. We splitted the data into training and testing sets, and we then trained regression machine learning models on the different types of the encoded dataset (one hot encoding, and ordinal encoding), and we also trained the models on ordinal encoded and normalized data. After training we evaluated the regression models on the testing set using RMSE. We found that our best results are from training a Polynomial Regression model on the ordinal encoded and normalized data.

Introduction

In this project, we want to predict the sales of video games, to do that we want to train a machine learning model and use the output model to predict the sales of video games. Machine Learning is a subfield in Computer Science which focuses on giving computers the ability to learn and make decisions. In order to give the systems the ability to learn, they are trained on the training set, which is a set that contains the examples that the system will learn from.

Machine learning can be classified based on the amount of supervision during training into supervised, unsupervised, semi supervised, and reinforcement learning. Supervised learning requires the training set to include the actual output (label). Where we feed the training set, features and labels into the model to learn from. Supervised learning problems can be classified into regression and classifications. In regression, we use the set of features to predict a numerical value. Where in classification, the system predicts a discrete output. In order to train a machine learning model, we start by choosing the dataset, then we analyze the dataset. After analyzing we clean the data, preprocess it, and prepare it for training. And after training, we evaluate the model using evaluation methods.

In this project, we first chose a dataset that we will use for training the model and later evaluation. The dataset contains multiple features such as the game platform, publisher, genre, and the year released. Also for each game we have the global sales.

Since we have the input features and the output data in the dataset, this problem falls under supervised learning. As we will use the input columns and the output column to train the machine learning model. Our output (sales) is a numerical value, this means that our problem is a regression problem, where we have to predict a value. It's also a multiple regression problem, since we have multiple features to predict the output.

After choosing the dataset, we analyzed the features, cleaned the data, preprocessed and prepared it for training. We then chose different regression models to use for training. We split the dataset into training and testing. And used the training set for machine learning training. We selected different machine learning regression models to train the data, and we evaluated the output of each model on the testing set.

The Data set

Our dataset contains the sales of video games, it contains 16598 rows and includes these columns: Rank, Name, Platform, Year, Genre, Publisher, NA_Sales, JP_Sales, Other_Sales, and Global_Sales as shown below. The source of the dataset (Video Game Sales) is Kaggle.

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37
6	Tetris	GB	1989.0	Puzzle	Nintendo	23.20	2.26	4.22	0.58	30.26
7	New Super Mario Bros.	DS	2006.0	Platform	Nintendo	11.38	9.23	6.50	2.90	30.01
8	Wii Play	Wii	2006.0	Misc	Nintendo	14.03	9.20	2.93	2.85	29.02
9	New Super Mario Bros. Wii	Wii	2009.0	Platform	Nintendo	14.59	7.06	4.70	2.26	28.62
10	Duck Hunt	NES	1984.0	Shooter	Nintendo	26.93	0.63	0.28	0.47	28.31

The Rank is the ranking of overall sales, The Name is the name of the video game, the Publisher is the company that published the game, the Platform is the system that is used for playing the game (such as PC, PS4), the Year is the year where the game was published. The Genre of the category of the game (such as Action, Sport).

The sales here reflect the sales in North America, Europe, Japan, other, and global sales. The global sales is the total world wide sales. The sales units are in millions.

The Rank and the Name of the video game does not provide any meaningful data so we will not use it for prediction. We will also remove NA_Sales, JP_Sales, Other_Sales as they are highly correlated with the Global sales.

The data that we will explore as our features are Platform, Publisher, Genre, and Year. And the predicted output is the Global sales. Platform, Publisher and Genre are all Categorical variables. We can see that this problem is a regression problem, where we have a set of independent features and a scalar output value.

Methodology

We started by cleaning the data, and then analyzing the data. We then prepared the data before applying different machine learning models, and recorded the results. We evaluated the results and chose our best model.

Dataset Cleaning

We imported the dataset and used the Rank column as the index column. We started data cleaning by removing unnecessary columns: Name, NA_Sales, JP_Sales, Other_Sales.
Then we checked the count of missing values in the dataset.

```
data.isnull().sum()
```

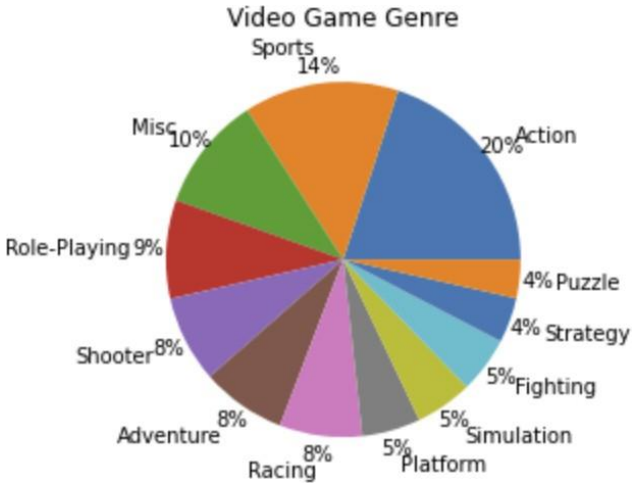
```
Platform          0
Year            271
Genre             0
Publisher        58
Global_Sales      0
```

As we can see from the figure above that 271 rows are missing the Year value, and 58 rows are missing the publisher value. To handle the missing values, we removed the rows with empty publisher value, and we replaced the missing year values with the most common year value using the mode function. Then we converted the Year values from Float into Integer.

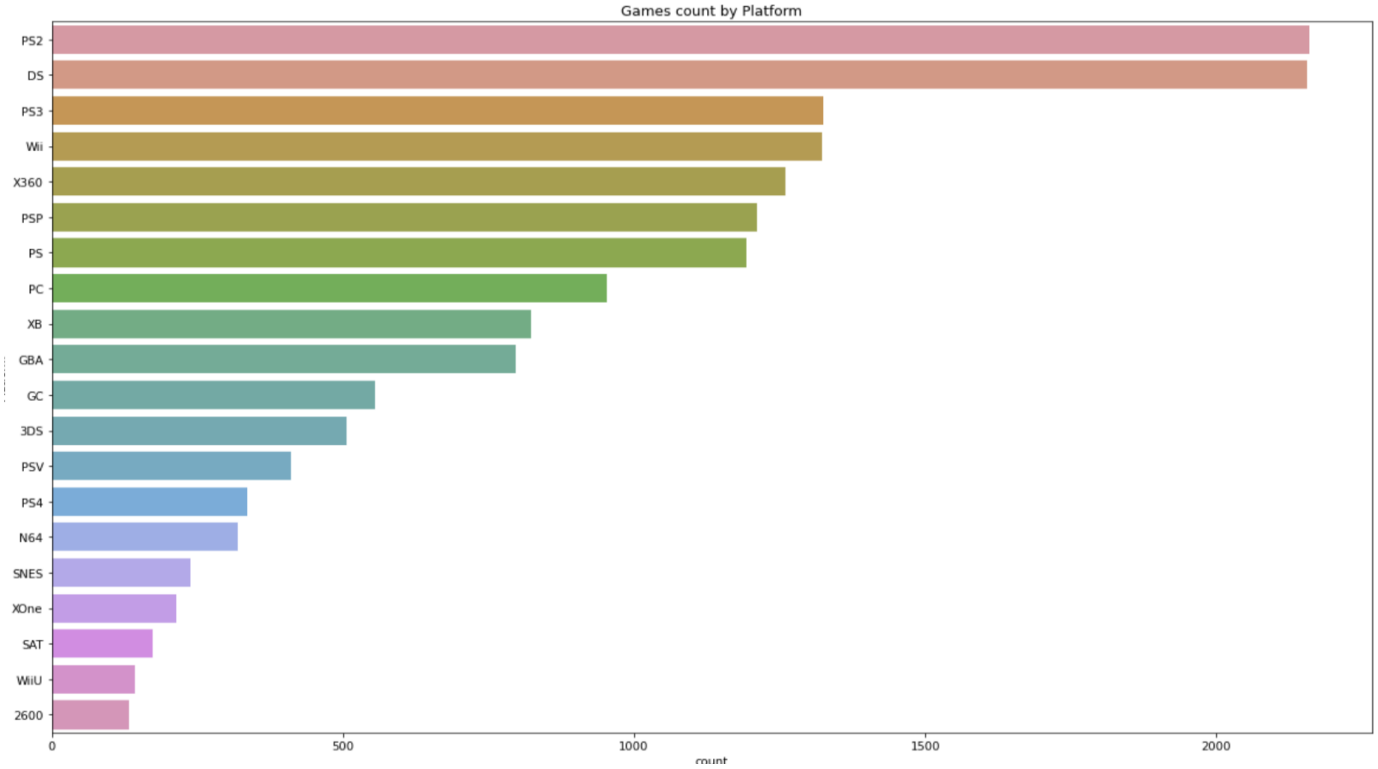
Dataset Analysis and visualization

We analyzed the data in our columns. First we analyzed the Genre column, the Genre column has 10 unique values, and we can see from the figures below that Action and Sport are the most used Genres in our dataset.

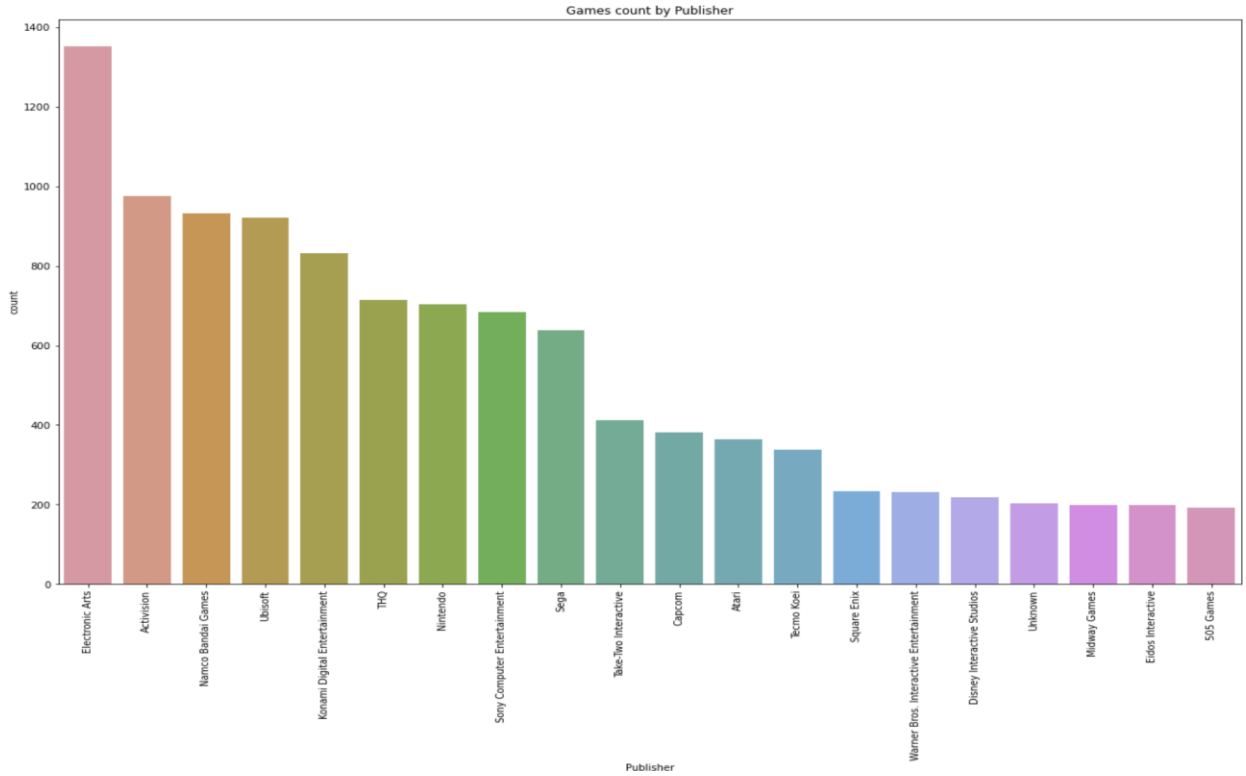
Action	3316
Sports	2346
Misc	1739
Role-Playing	1488
Shooter	1310
Adventure	1286
Racing	1249
Platform	886
Simulation	867
Fighting	848
Strategy	681
Puzzle	582



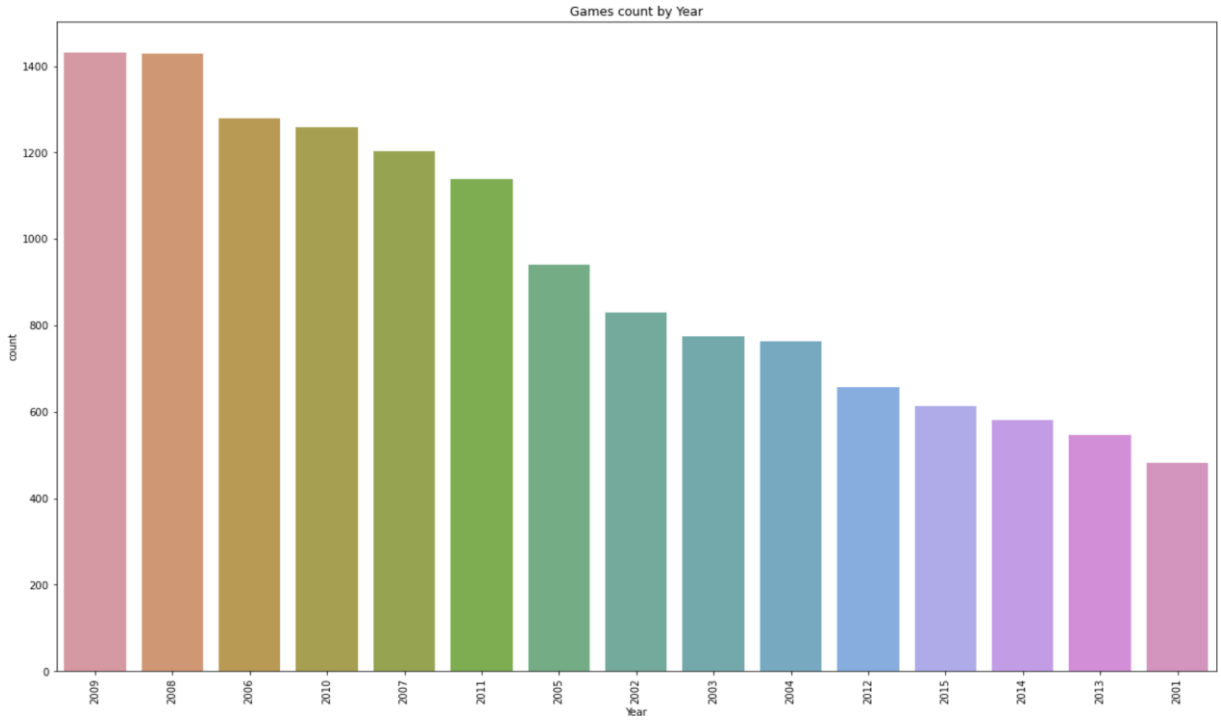
Then we analyzed the Platform data, The Platform column has 31 unique values. The figure below shows the top 20 platforms used for the games in our dataset. We can see that DS and PS2 are the most common platforms with over 2000 games each.



Then, we analyzed the Publisher data. The Publisher column has 579 unique publisher values. The figure below shows the top 20 publishers. We can see that Electronic Arts is the top publisher in the dataset with over 1300 games.



Then we analyzed the Year data, the year values start from 1980 up to 2020. The figure below contains the top 15 common years in the dataset. We can see that 2008 and 2009 are the top common years with over 1400 games each, followed by 2006, 2010, 2007 with around 1200 games each, and around 1100 games in 2011.



Data preprocessing and preparation

Before training machine learning models on the dataset, we first pre processed and prepared the data. First we handled the large number of publishers' unique values.

```
data['Publisher'].describe()
```

```
count               16540
unique                578
top       Electronic Arts
freq                 1351
Name: Publisher, dtype: object
```

The publisher column contains 579 unique values, and around 200 publishers occurred only once in the dataset as we can see from the below figure.

```
(data['Publisher'].value_counts() == 1).value_counts()
```

```
False    384
True     194
Name: Publisher, dtype: int64
```

Also, around 530 publishers occurred less than 50 times, which were part of about 3000 rows out of the whole dataset. Which means around 530 publishers were rare and not repeated enough. We replaced the value of these rare publishers into “Other”.

```
(data['Publisher'].value_counts() < 50).value_counts()
```

```
True     532
False     46
Name: Publisher, dtype: int64
```

We also removed rows with outliers for global sales values. Then we shifted and rescaled the Year column values. The years data represents the relation of years passing on the sales. So we represented the start year as 0 and the year values in the data will represent how many years passed from the start year.

Then we handled the categorical variables: Publisher, Platform, and Genre. We have to encode these variables and convert them into numerical data before using them as input for machine learning models. There are two ways of encoding categorical variables: One hot encoding and Ordinal encoding.

We experiment with both encoding types. One hot encoding adds dummy variables for each value of the categorical variable, and each of these dummy variables holds values of 0 and 1. We encoded Publisher, Platform, and Genre using One Hot encoding as shown in the figure below. We can see that as a result of one hot encoding, we have now 92 columns in the dataset.

Rank	Year	Global_Sales	2600	3DO	3DS	DC	DS	GB	GBA	GC	...	SquareSoft	THQ	Take-Two Interactive	Tecmo Koei	Ubisoft	Unknown	Virgin Interactive	Vivendi Games	E
333	19	3.72	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
334	20	3.71	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
335	24	3.70	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	
336	18	3.70	0	0	0	0	0	1	0	0	...	0	0	0	0	0	0	0	0	
337	33	3.69	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	

5 rows × 92 columns

Ordinal encoding assigns a number for each value of the categorical variable as shown in the below figure. We can see that the values for Publisher, Platform, and Genre are replaced by numerical values.

Rank	Platform	Year	Genre	Publisher	Global_Sales
333	1	19	1	1	3.72
334	1	20	2	2	3.71
335	2	24	3	3	3.70
336	3	18	4	4	3.70
337	4	33	5	3	3.69

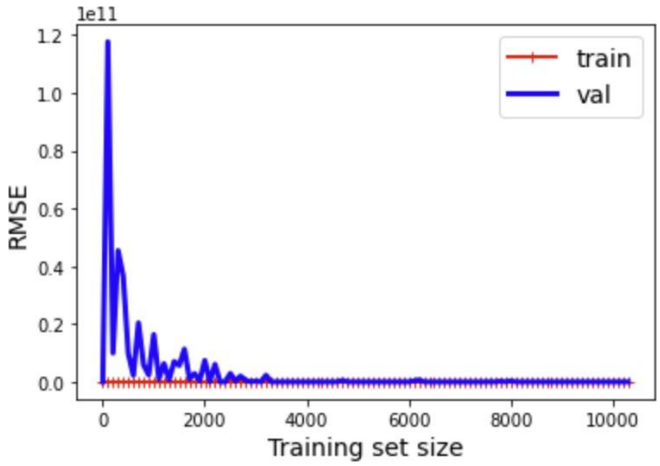
We also did experiments with normalization, and normalized the ordinal encoded data.

Machine Learning Regression Models

We experimented with different regression models. And applied them on different versions of the dataset, we first did experiments with the one hot encoded data, then we experimented with the ordinal encoded data, lastly we applied the models on the ordinal encoded and normalized data. The regression models we experimented with were: Linear Regression, Polynomial Regression, and SVM.

For each version of the dataset: one hot encoded, ordinal encoded, and ordinal encoded and normalized version, we split the data into training and test sets with 80:20 ratio, meaning 80% of the data is used for training and 20% is used for testing.

First we will explain the experiments on the one hot encoded dataset. We first used the Linear Regression model. We trained the model on the training set, the figure below shows the model learning curve, and then we used the trained model to predict output of the testing set. Then we used the predicted output and original output to evaluate the model.



After that, we experiment with Polynomial Regression. First we used PolynomialFeatures to fit and transform the input data with the degree specified, we then used the output of PolynomialFeatures to train the Linear Regression as the new input features. We experiment with 2 degrees as input for the PolynomialFeatures. And used the final trained model to predict the output of both the training and testing set to evaluate the model and to check if it's overfitting.

The last experiments with the one hot encoded data were with SVM Regression models. We first trained the model using LinearSVR and evaluated the output model on both the training and testing set. Later we used an RBF kernel and used randomized search and cross validation to find the appropriate hyper parameters gamma and c, and our best SVR estimator is SVR(C=4.745401188473625, gamma=0.07969454818643928). Then we used our best estimator model to see how it performs on both the training and testing sets and evaluated our model.

After that, we did similar experiments with the ordinal encoded data. We first trained a linear regression model on the training dataset, then we used the model to predict the output of the testing set to evaluate the model.

Then we trained the model with 3 degrees Polynomial Regression, and evaluated the model on both training and testing sets to evaluate the model and check if it overfits.

The last experiment on ordinal encoded data was with SVM, we trained the LinearSVR model on the training set and predicted the output of the testing set to evaluate the model.

After that, we experimented with the final version of our dataset, the ordinal encoded and normalized data. We first trained the model using Linear Regression and evaluated the mode on the test data set. Then, we used polynomial regression, and experimented with different polynomial degrees. We evaluated the polynomial model on both training and test. We chose 4 as the best degree for the model. Finally we trained the LinearSVR model on the dataset, and evaluated the model on the testing set.

Models Evaluation

To evaluate our models, we used the trained model to predict the output on the testing set, and then computed Root Mean Squared Error (RMSE) using the predicted and actual output. The model with smaller RMSE on the testing set is the best model for the dataset. We also evaluated the model on the training set and computed the training RMSE and compared it with the testing RMSE to check if the model overfits the data.

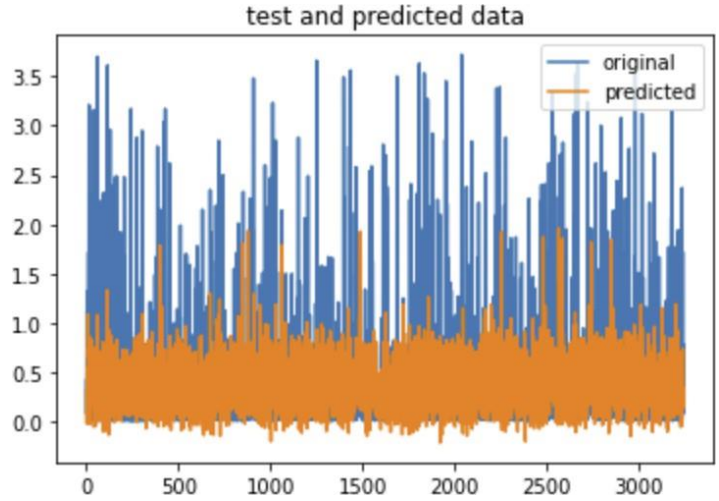
Results

We compared the results from the regression modules for each version of the dataset: one hot encoded, ordinal encoded, and finally ordinal encoded and normalized.

First, we start by the model's results when we trained the models on the One Hot Encoded dataset.

1. Linear Regression:

Figure below compares the actual output to the predicted output we got from the Linear Regression model. The Root mean squared error is 0.5219
We can see that the model does not give great results as the RMSE is quite large.

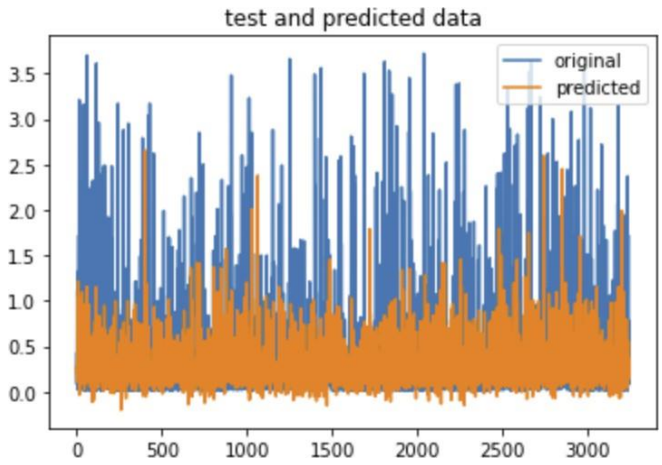


2. Polynomial Regression

The polynomial regression was not suitable for the dataset with one hot encoding. And the model was overfitting. The RMSE error we got from the training dataset was 0.46, and the testing RMSE was 29443199.12

3. SVM Regression

We trained the LinearSVM model on the training dataset, and our testing RMSE was 0.52. We then used Randomized search and cross validation with RBF kernel to find the best gamma and c, and we evaluated our best estimator model on the testing dataset, and the RMSE we got was 0.52, the figure below compares the actual data and the predicted output we got from the model.

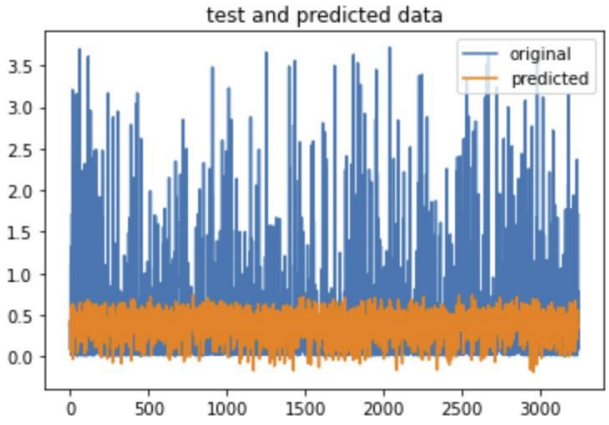


With one hot encoded we got similar results with both Linear Regression and SVM, however, Polynomial Regression was not suitable for the data and the model was overfitting, as it performed well in the training data but didn't perform well on the test data and the testing error was very large.

Next, we will show the results of the regression models when applied on the Ordinal encoded data:

1. Linear Regression

Figure below shows the comparison between the Actual and Predicted data we got when we evaluated the linear Regression model on the testing dataset. The Root Mean Squared Error of the model was 0.564

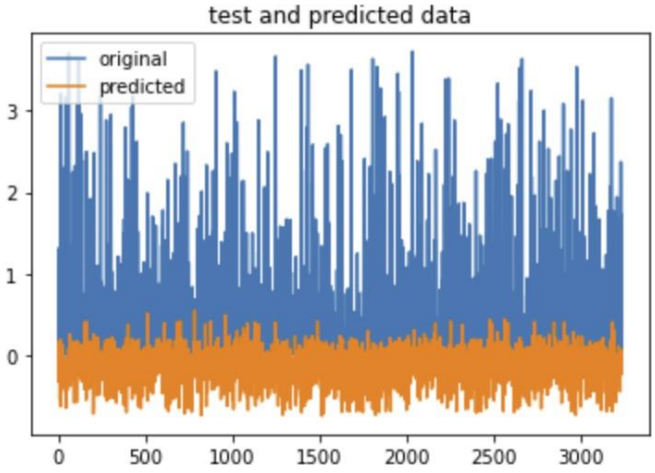


2. Polynomial Regression

The Polynomial Regression (3 degrees) performed well on both the training and testing data, with close RMSE errors. The Train RMSE was 0.534, and the Testing RMSE error was 0.553. The Polynomial Regression results were very similar to the Linear Regression.

3. SVM Regression

SVM Regression performed worse with Ordinal encoding data, The Test RMSE we got was 0.77. The figure below shows the comparison between the Actual data and the Predicted output from the model.

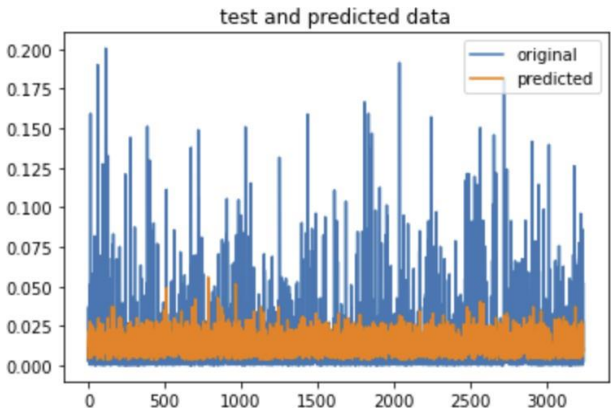


With ordinal encoded data, both Linear Regression and Polynomial regression performed well on the dataset and we got very close RMSE results from both models. SVM Regression performed worse on the ordinal encoded data in comparison to the one hot encoded dataset.

Next we will show the results of our regression models when trained on ordinal encoded and normalized data.

1. Linear Regression

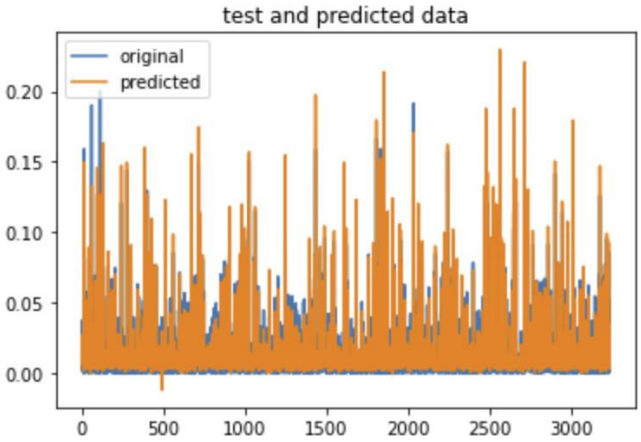
Linear Regression performed well on the data, The Figure below shows the comparison between the Actual output and the Predicted output from the model. The Testing RMSE of the model was 0.021.



2. Polynomial Regression

Polynomial degrees performed the best on the data, The Actual output and Predicted output from the model were close as shown on the figure below, The training RMSE was 0.0061, and the testing RMSE = 0.0064

We experimented with different polynomial degrees, and the best result we got was with 4 degrees polynomial.



3. SVM Regression

SVM regression performed similar to Linear Regression and we got very close RMSE testing result: 0.02

Conclusion

In this problem, we wanted to predict the sales of video games, we used the video games sales dataset, which contained different features of the games (Platform, Genre, Action, Year) and

their global sales. The sales prediction falls under regression problems where the output we want to predict is a value.

We started by analyzing and visualizing the data, and then we cleaned the data. After that, we prepared the data before applying machine learning models. We have multiple categorical features in the dataset, and we encoded the data using Ordinal encoding and One Hot encoding. We also experimented with applying normalization on the ordinal encoded data.

Then we applied different machine learning regression models on the different versions of the dataset: One Hot encoded data, Ordinal encoded data, and finally normalized and ordinal encoded data. We split the data into 80% training and 20% testing and we trained Linear Regression, Polynomial Regression, and SVM Regression on the training data. We used the trained model to precinct the output of the test test and we calculated RMSE to evaluate and compare our models.

With the one hot encoded data, both Linear Regression and SVM regression performed well and gave us similar results, however, Polynomial regression was overfitting and didn't perform well on the data. The results from Linear Regression and SVM models were not great and the RMSE error was quite large, with one hot encoding we got a lot of dummy features, even though we reduced the number of categories for the Publisher categorical variable. And it could have affected the models as we have a huge number of features in comparison to the dataset size.

With ordinal encoded data, both Linear Regression and Polynomial regression performed well. SVM regression performed worse than One Hot encoded data. However, the output from the best model on ordinal encoded data was still similar to the model trained on one hot encoded data, and we still got large RMSE values. However, after we normalized the ordinal encoded data, we got better results with Polynomial regression. The RMSE value was small, and the Actual output and the Predicted output from the polynomial model were very close.