ML Projects Milestone 1

Online Articles Popularity Prediction

Team ID: SC_7

فرح صفوت عز الرجال علي 2021170392

سهير خالد محمد السيد 2021170245

محمد خالد توفيق محمد سعد 2021170463

نور سامي محمود ياغي 2021170592

مارك ماجد موريس عزيز 2021170425

مازن محمد عبد الحميد ابراهيم 2021170661

Introduction:

In our Online Articles Popularity Dataset,

we looked closely at the dataset to understand what it's all about. We wanted to see how different things in the data relate to each other.

Our main goal was to build models that could predict certain outcomes based on the information we had.

But before we could do that, we had to tidy up the data. We fixed any missing or weird values and made sure everything was in a format that the computer could understand.

Then, we tried out different ways of building these prediction models, each with its own way of looking at the data.

Finally, we compared how well each model worked to see which one was the most helpful in making predictions.

- **Overview on the Data:**

  The dataset contains 38,643 rows and 46 columns.

  The data consists of various features or attributes that describe different aspects of something we're interested in. These features could be numerical values like 'timedelta' and 'n_tokens_content' columns, or they could be categorical values like 'weekday' column.

Preprocessing Techniques:

1. **Handling Missing Values & Duplicates**: we first checked for any missing data using method **isnull()** to find affected columns. Then, we took two main steps: removing rows with NaN values and filling missing values in columns using **dropna()** to avoid affecting our analysis and we used **duplicated()** to check for any duplicates, but there was no duplicates found.

For missing values, we replaced '**[]**' in the 'channel type' column with the mode value '**data_channel_is_world**'.

2. **Feature Engineering:** we engineered another feature by computing the length of title length stored it in a new column called **'title length'.**

   This feature captures the textual characteristic of the titles, which could influence the level of reader engagement and consequently, the number of shares an article receives.

   These feature engineering steps enrich our dataset, empowering our machine learning model with additional insights to better understand the relationship between article attributes and their popularity.

3. **Handling Outliers:** we have dealt with the outliers with replacing them with the Upper bounds.

4. **Encoding:** firstly, in 'channel type' column, we removed 'data_channel_is' from its values, as it's common between all values and then applied one-hot encoding to the column.

   Secondly, we instantiated a **LabelEncoder** object.

   Then, we used the **fit_transform()** method to encode the 'weekday' column. This process assigned a unique numerical label to each category within this column, allowing machine learning algorithms to interpret it as numerical feature.

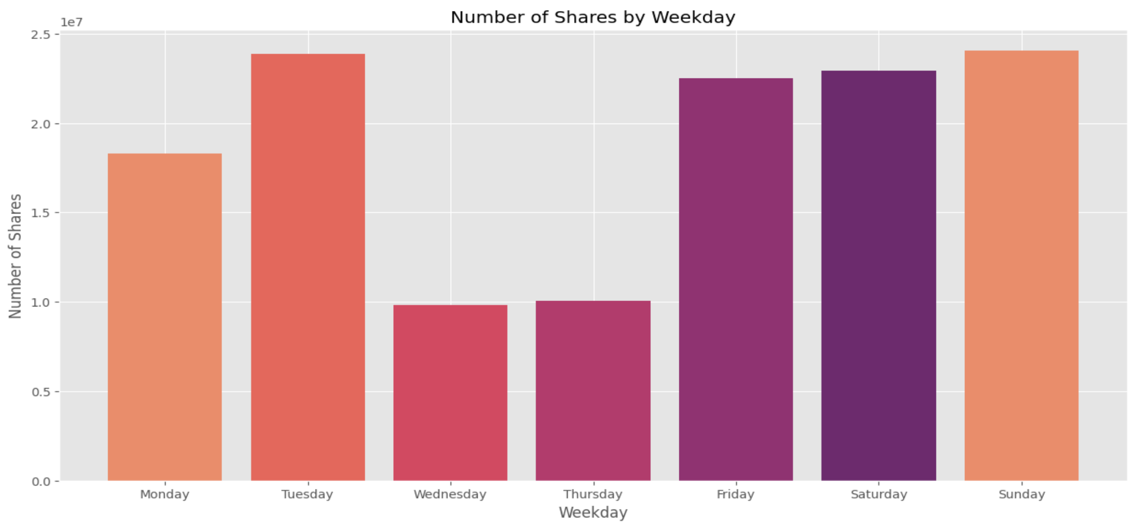
   Additionally, we replaced the binary 'isWeekEnd' column values ('Yes' and 'No') with numeric equivalents (1 and 0) using the **replace()** function, simplifying whether it is a weekend or not.

   Lastly, we can't encode neither 'url' or 'title' as they have a huge amount of unique values so it was enough for us to extract the most important information from these columns and vectorizing this information in feature engineering before dropping them as they're not useful in our dataset now.

5. **Feature Scaling:** we normalized the numerical features in our dataset.

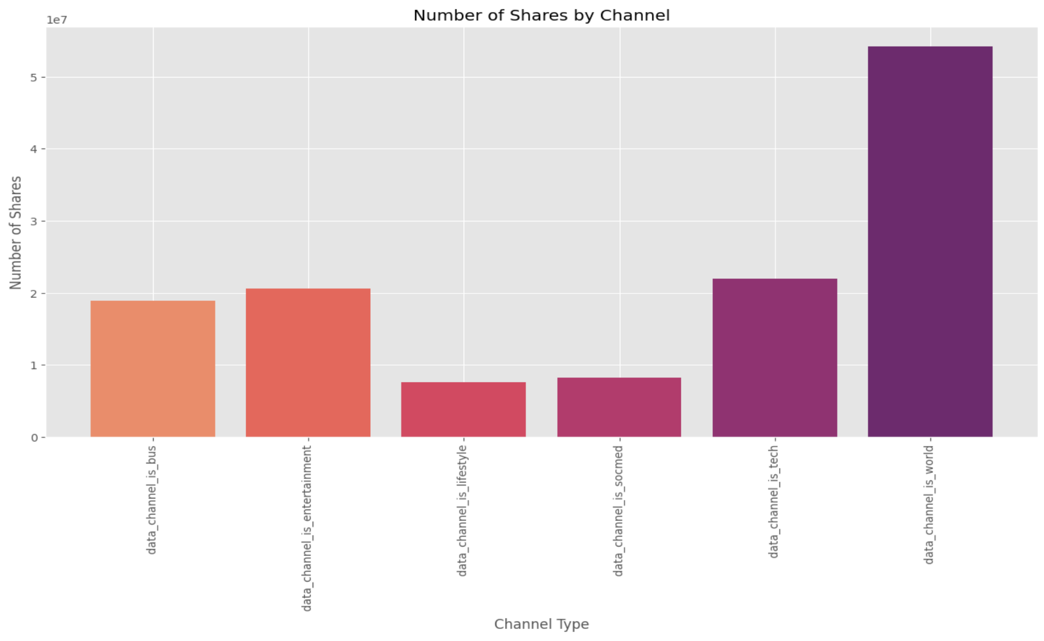
Analysis on the Dataset:

- Number of Shares by Weekday:



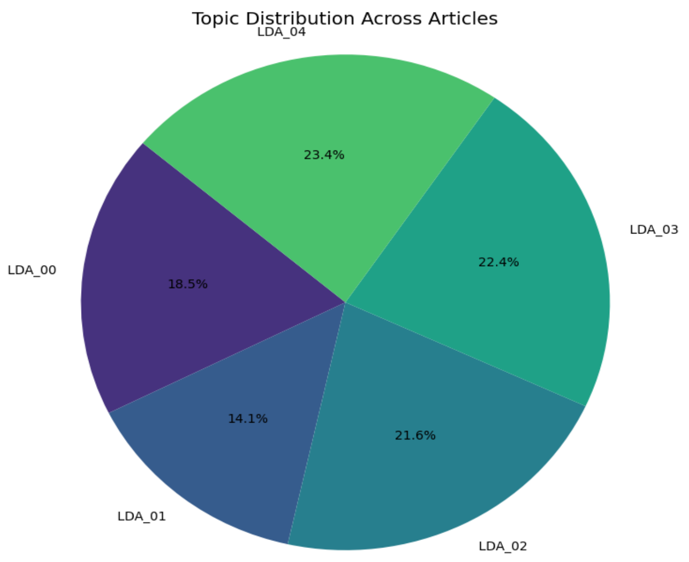
- It is obvious that Tuesday and Sunday Has the most Shares over all the Weekend days.

- Shares by Channel:



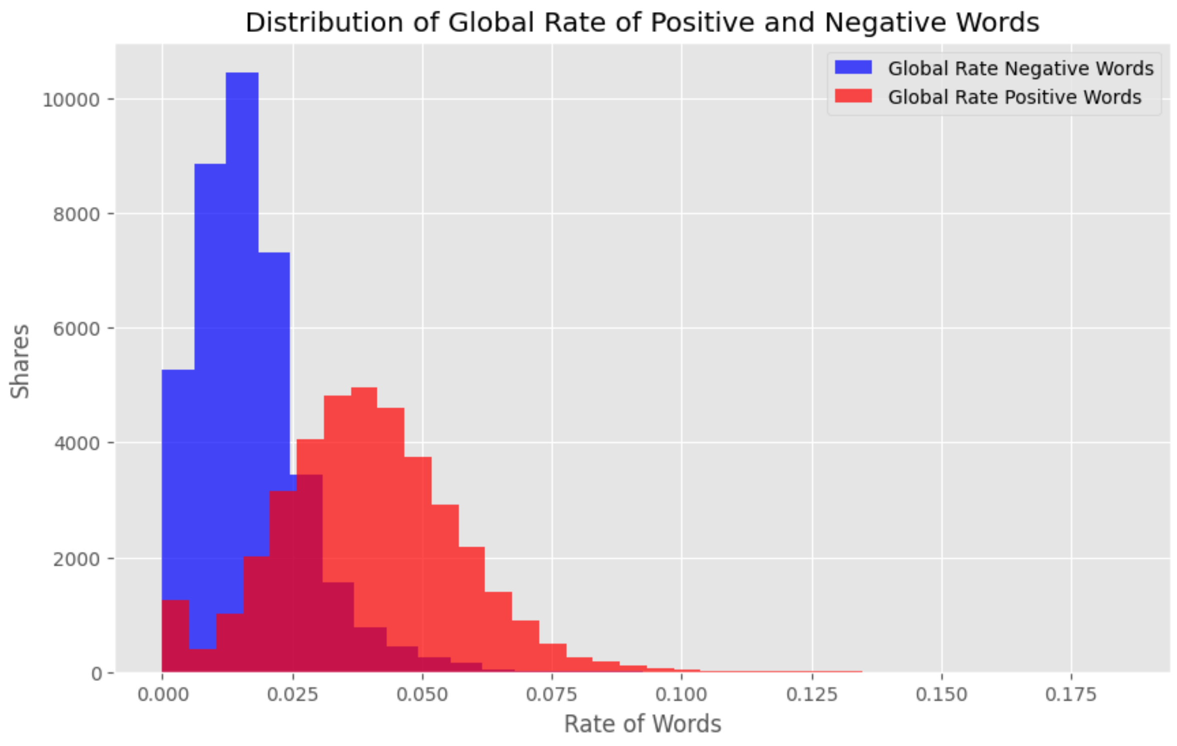
- it shows here that the 'data_channel_is_world' has the most shares over all the channels.

- Distribution Across Articles:



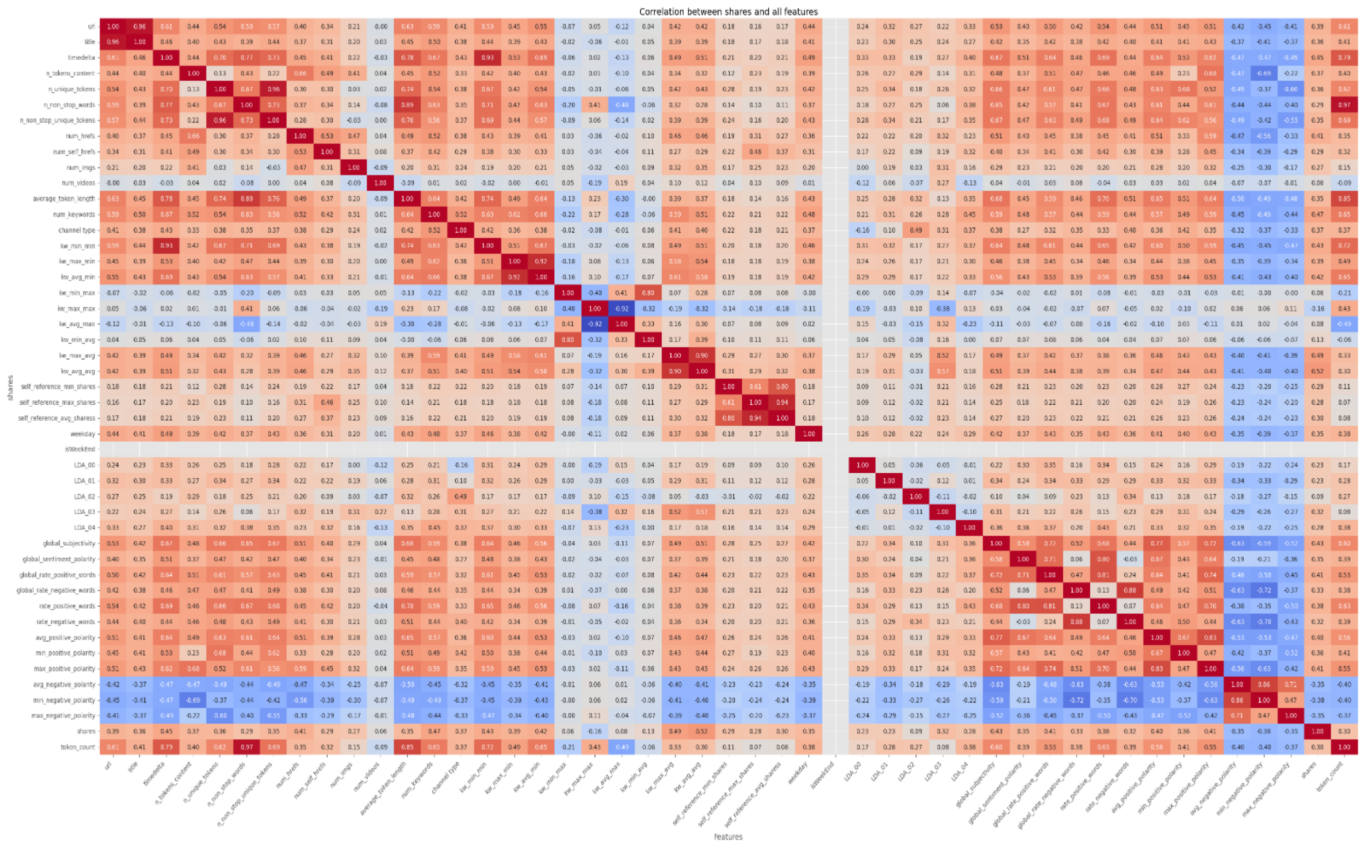
LDA_04 has the most Percentage of Distribution among all the other Articles

- Effect of Positive and Negative Words on Search:



Feature Selection:

1. Correlation Heat Map:



It is shown that:

kw_avg_avg	0.524116
kw_max_avg	0.487348
num_keywords	0.465296
timedelta	0.449236
global_subjectivity	0.430336

2. RFE:

We performed feature selection using Recursive Feature Elimination (RFE) with a Linear Regression estimator

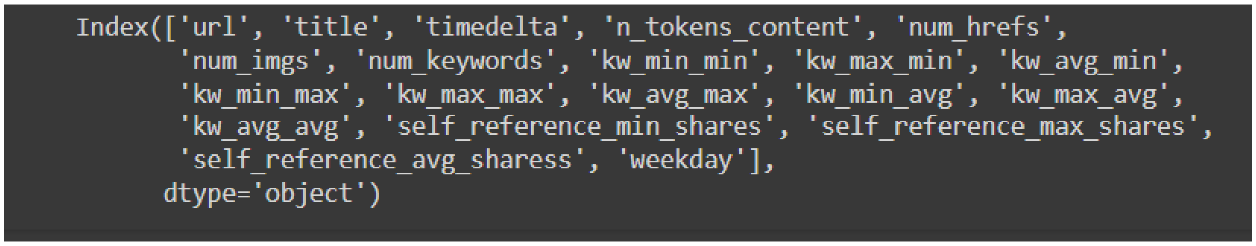
Then, an instance of Linear Regression was created as the estimator for RFE.

RFE is a technique that recursively removes features from the dataset while fitting the model and selects the subset of features that contributes most to predicting the target variable. Here, I specified to select the top 20 features.

After fitting the RFE object to the training data, the selected features were identified using the support attribute of RFE, which contains a boolean mask indicating the selected features.

These selected features were then assigned to the variable X for further analysis or modeling

Features Selected:



```
Index(['url', 'title', 'timedelta', 'n_tokens_content', 'num_hrefs',
       'num_imgs', 'num_keywords', 'kw_min_min', 'kw_max_min', 'kw_avg_min',
       'kw_min_max', 'kw_max_max', 'kw_avg_max', 'kw_min_avg', 'kw_max_avg',
       'kw_avg_avg', 'self_reference_min_shares', 'self_reference_max_shares',
       'self_reference_avg_sharess', 'weekday'],
      dtype='object')
```

And we discarded the rest of the features.

Sizes of Training and Testing Sets:

When splitting the data into training and testing sets, we had to take into consideration the amount of data we use in each to avoid overfitting and underfitting.

What we found works out best was 30% (0.30) for the testing set and 70% (0.70) for the training set.

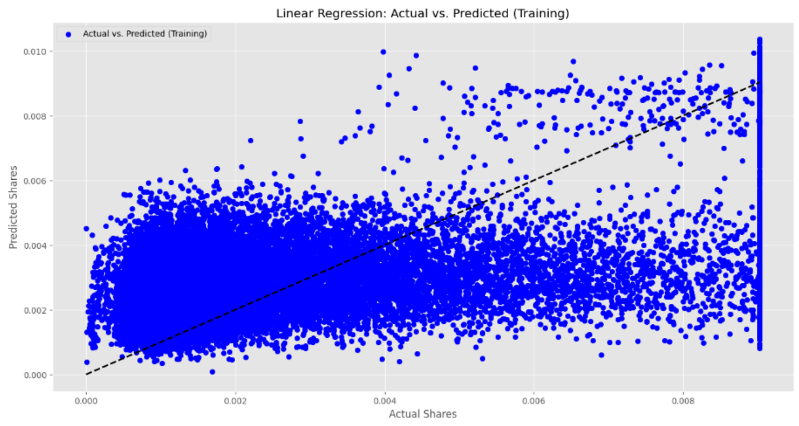
Regression Techniques Used:

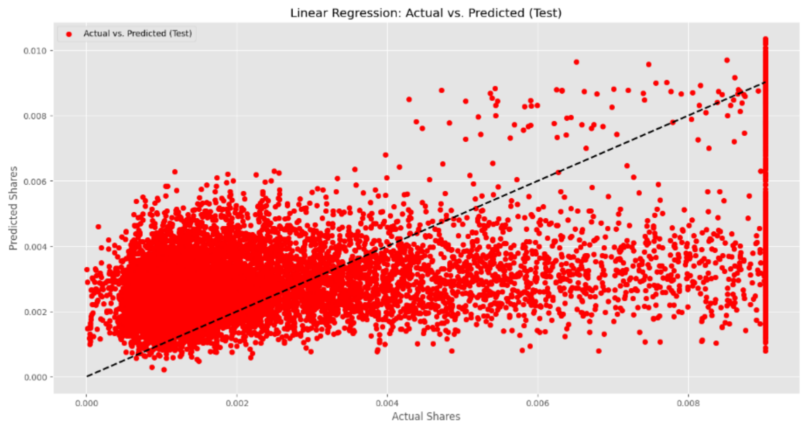
1. **Linear Regression:**

   - **Description:** Linear Regression is a fundamental regression technique that models the relationship between the independent variables (selected features) and the dependent variable (shares) as a linear equation.

   - **Implementation:** We used the Linear Regression model from the scikit-learn library to fit the data and predict the target variable.

   - **Evaluation:** Model's performance was evaluated using metrics: Mean Squared Error (MSE) and R-squared score on both the training and test sets.



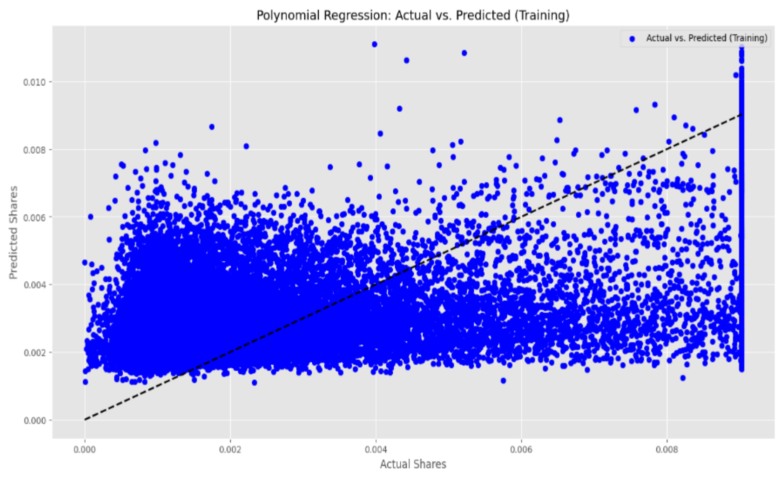


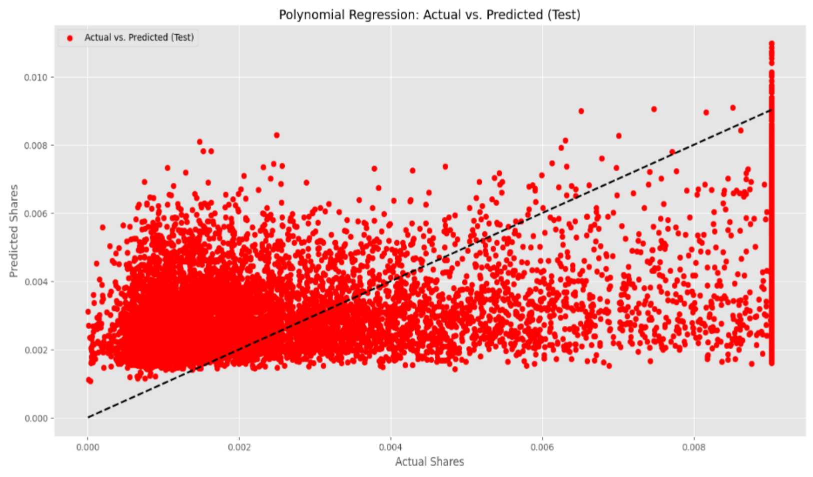
2. **Polynomial Regression:**

   - **Description:** Polynomial Regression extends the Linear Regression model by adding polynomial terms to capture non-linear relationships between the selected features and the target variable (shares).

   - **Implementation:** We employed Polynomial Regression with a degree of 3 based on the result of the grid search and Ridge regularization using the Ridge model from scikit-learn.

   - **Evaluation:** We evaluated the Polynomial Regression model's performance using MSE and R-squared score on both the training and test sets.





3. **XGBoost Regression:**

- **Description:** XGBoost (Extreme Gradient Boosting) is a highly efficient and widely used machine learning algorithm known for its performance in regression tasks. It belongs to the gradient boosting family and works by sequentially improving weak learners, such as decision trees, to minimize residual errors.

- **Implementation:** Implemented using Python's xgboost library, the XGBoost regression model was configured with key hyperparameters:

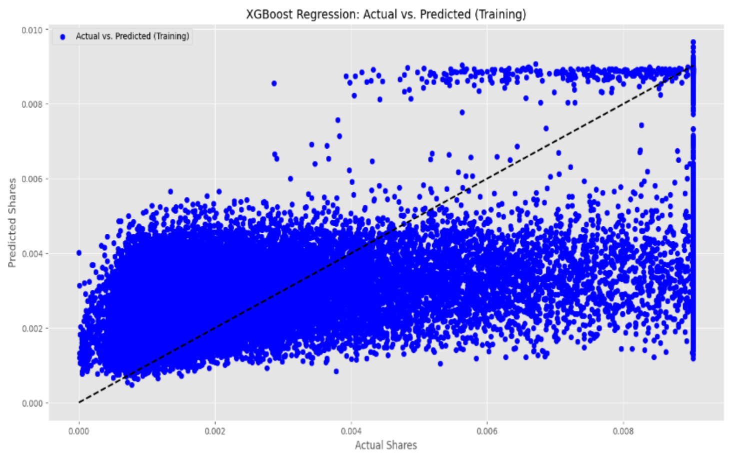
  - Maximum depth of the tree: 4

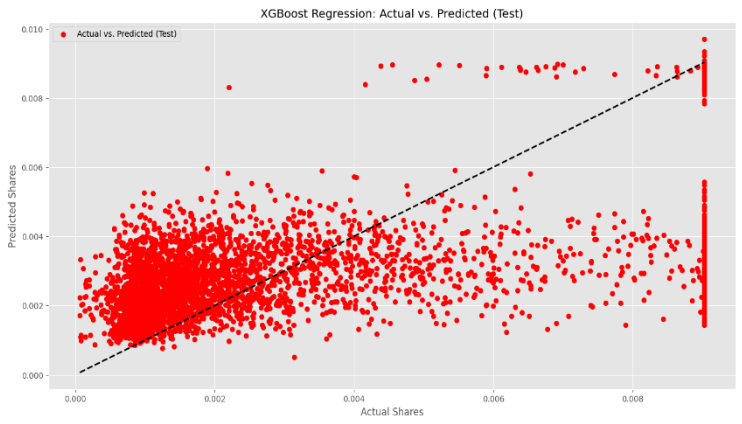
  - Learning Rate: 0.1

  - Number of Estimators: 200

  - Random state: 45

- **Evaluation:** The model's performance was assessed using mean squared error (MSE) and coefficient of determination (R^2). It demonstrated high accuracy and low error rates, indicating its effectiveness in capturing the underlying patterns in the data.





Model Evaluation:

Model	Mean Square Error	Accuracy
Linear Regression	5.29	37.012%
Polynomial Regression	5.717	32.048%
XGBoost Regression	4.671	44.274%

Difference between Test and Training Accuracy:

The difference between test and training accuracy, known as the "generalization gap," reflects a model's ability to perform on unseen data compared to its training set.

A smaller gap indicates better generalization, while a larger one suggests potential overfitting to the training data.

Further Techniques Used to Improve Results:

1. Recursive Feature Elimination (RFE): a feature selection technique, is applied to choose the most relevant features for the regression model. It starts by splitting the dataset into training and testing sets.

   Then, a LinearRegression estimator is instantiated. Recursive Feature Elimination (RFE) is utilized with the estimator to iteratively remove the least significant features until the desired number of features to select is reached.

   After fitting RFE to the training data, the selected features are identified based on their support.

   Finally, the dataset is updated to include only the selected features for further analysis.

2. Hyperparameter Tuning: Conducting a thorough search for optimal hyperparameters (degree for polynomial regression) to improve the model's performance by tuning the hyperparameter alpha.

3. Feature Engineering: I developed a function to extract length from Title using a regular expression pattern. This function was then applied to the 'url' column, resulting in the creation of a new column called 'title_word', we added 'title_length' column to our dataset.

Conclusion:

In summary, our milestone in the Online Articles Popularity Prediction project involved cleaning up our data, adding useful features, and testing different regression models to predict article popularity accurately.

We made sure our data was in good shape by handling missing values, outliers

Then, we added new features like title length to help our models make better predictions.

We tested several regression models: Linear Regression, Polynomial Regression, and XGBoost Regression. The best model was the XGBoost since it had the highest accuracy and the lowest error.

Our analysis also gave us insights into when articles are most popular and how different features relate to their popularity.

By using techniques like correlation heatmaps and feature selection, we narrowed down the most important factors for predicting article popularity.

Overall, our milestone marks progress in building a reliable model for predicting online article popularity.