Predicting Instagram Post Impressions

Data Bootcamp Final Project

Hariharan Janardhanan hj2342

Introduction

Instagram is a widely used social media platform, where billions of users share posts daily, making it a hub for marketers, advertisers, and influencers to connect with their audiences. This project aims to build a predictive model to estimate the Impressions (visibility) of Instagram posts based on various engagement metrics, helping users optimize their social media strategies and maximize reach.

Problem Statement

The project develops a predictive model to estimate Instagram post Impressions using features such as Likes, Saves, Profile Visits, Follows, and Hashtag Count. By leveraging machine learning models—including Multiple Linear Regression, KNN, Decision Trees, Random Forest, XGBoost, and Artificial Neural Networks—the analysis aims to identify the most influential metrics contributing to Impressions. These insights will help social media influencers, marketers, and advertisers tailor their strategies for improved visibility.

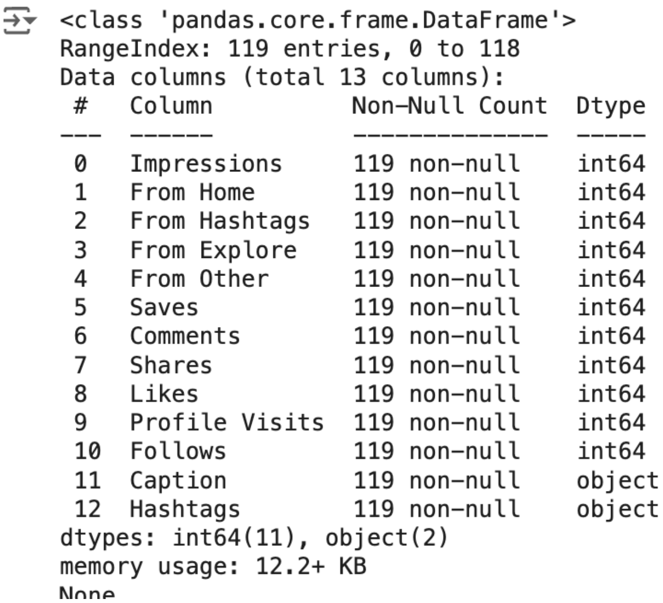
Dataset Description

Dataset Description The dataset for this project was sourced from Kaggle, specifically the "Instagram Data" dataset by Amir Motefaker. It contains engagement metrics and metadata for Instagram posts, providing valuable information for analyzing and predicting the impressions of a post based on its features. The dataset consists of 119 rows and 13 columns, all of which are clean and non-null, requiring minimal preprocessing.

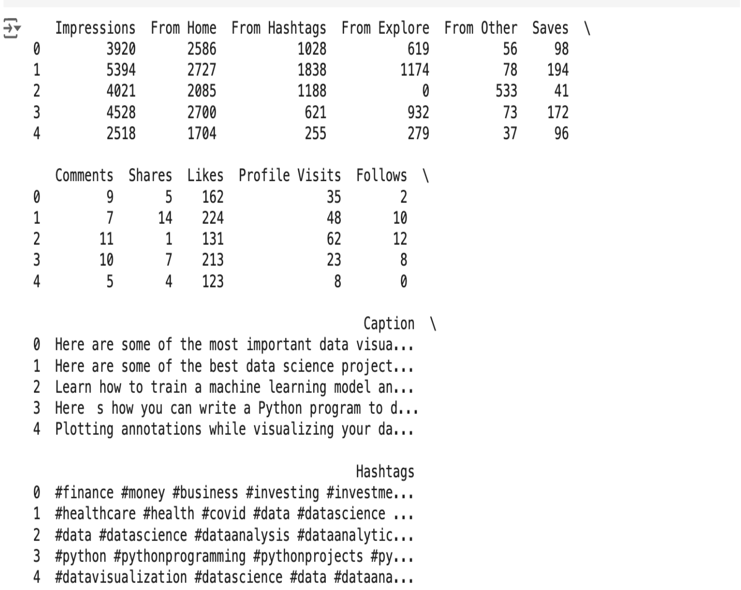
Key Features:

1. Impressions (Target Variable): Total number of times a post was viewed.
2. Likes, Saves, Comments, Shares: Engagement metrics representing user interactions with posts.
3. Profile Visits, Follows: Indicators of deeper user engagement and post influence.
4. From Home, From Hashtags, From Explore, From Other: Breakdown of where impressions originated.
5. Caption, Hashtags: Text-based features describing the content of the posts.

The dataset is relatively small, with only 119 records, but it provides a diverse set of features to explore the relationships between different engagement metrics and impressions. Due to the limited size of the dataset, model performance could be impacted, particularly for complex algorithms, which might benefit from additional data for training and validation.



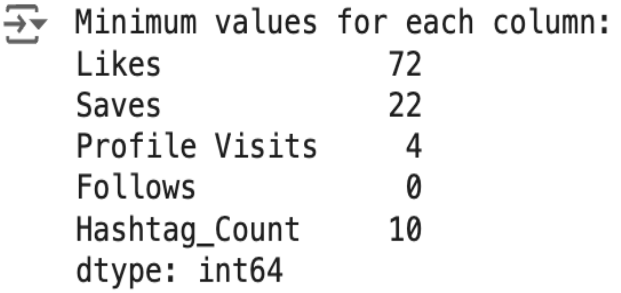
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119 entries, 0 to 118
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Impressions     119 non-null    int64
 1   From Home       119 non-null    int64
 2   From Hashtags   119 non-null    int64
 3   From Explore    119 non-null    int64
 4   From Other      119 non-null    int64
 5   Saves           119 non-null    int64
 6   Comments        119 non-null    int64
 7   Shares          119 non-null    int64
 8   Likes           119 non-null    int64
 9   Profile Visits  119 non-null    int64
 10  Follows         119 non-null    int64
 11  Caption         119 non-null    object
 12  Hashtags        119 non-null    object
dtypes: int64(11), object(2)
memory usage: 12.2+ KB
None
```



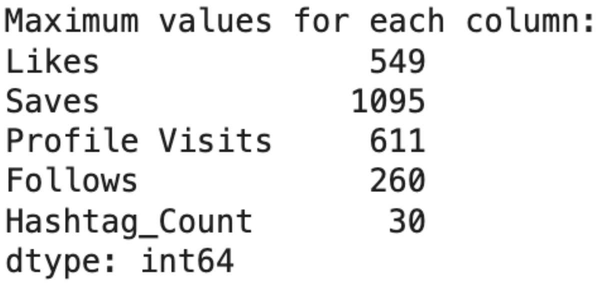
```
   Impressions  From Home  From Hashtags  From Explore  From Other  Saves  \
0         3920       2586           1028           619          56     98
1         5394       2727           1838          1174          78    194
2         4021       2085           1188             0         533     41
3         4528       2700            621           932          73    172
4         2518       1704            255           279          37     96

   Comments  Shares  Likes  Profile Visits  Follows  \
0         9       5    162              35        2
1         7      14    224              48       10
2        11       1    131              62       12
3        10       7    213              23        8
4         5       4    123               8        0

                                             Caption  \
0  Here are some of the most important data visua...
1  Here are some of the best data science project...
2  Learn how to train a machine learning model an...
3  Here  s how you can write a Python program to d...
4  Plotting annotations while visualizing your da...

                                            Hashtags
0  #finance #money #business #investing #investme...
1  #healthcare #health #covid #data #datascience ...
2  #data #datascience #dataanalysis #dataanalytic...
3  #python #pythonprogramming #pythonprojects #py...
4  #datavisualization #datascience #data #dataana...
```

Data Preprocessing

- Handling Missing Values: Imputed missing values using appropriate strategies (e.g., mean/mode). Our dataset appears to have no missing values, which is excellent for our analysis.
- Feature Scaling: I noticed that the dataset does not include a dedicated column for the number of hashtags per post. Instead, the "hashtags" column contains all the hashtags used in each post as a single string. To address this, we can create a new column that counts the number of hashtags in each post.
- Outlier Handling: Identified outliers to improve model performance.



```
Minimum values for each column:
Likes              72
Saves              22
Profile Visits      4
Follows             0
Hashtag_Count      10
dtype: int64
```



```
Maximum values for each column:
Likes              549
Saves             1095
Profile Visits     611
Follows            260
Hashtag_Count       30
dtype: int64
```

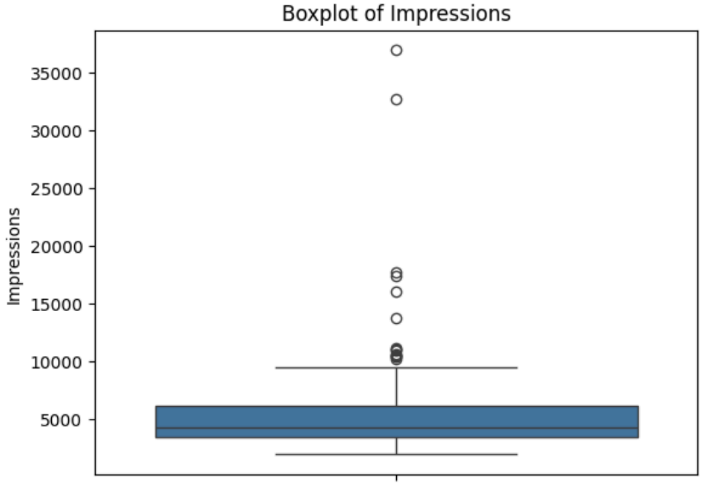
The dataset I am working with contains data from 119 Instagram posts. The engagement metrics show a wide range, with the number of likes varying from 72 to 549, saves ranging from 22 to 1095, and profile visits spanning from 4 to 611. The number of follows for each post ranges from 0 to 260, while the hashtag count varies between 10 and 30. This diverse range of values provides a solid basis for exploring the relationship between post features and impressions, although the relatively small dataset size might limit the robustness of some machine learning models.

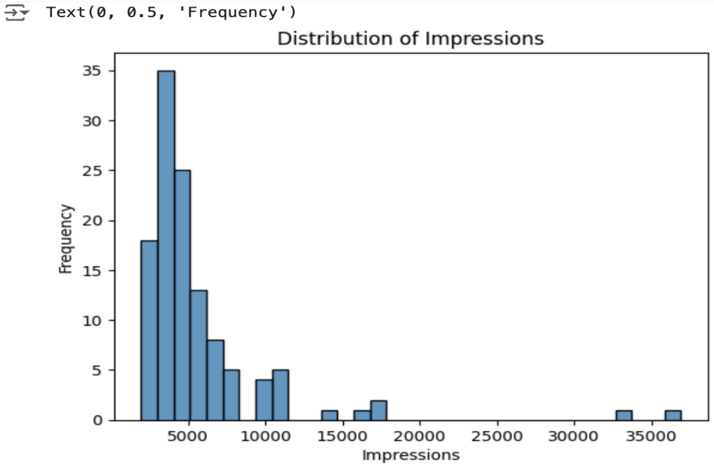
Exploratory Data Analysis (EDA):

○ **Descriptive Statistics**

1. Min Impressions: 1941
2. Max Impressions: 36919
3. Mean Impressions: 5703.991596638655
4. Variance of Impressions: 23462205.703318622
5. Standard Deviation of Impressions: 4843.780104765143

A high standard deviation suggests the possibility of outliers.

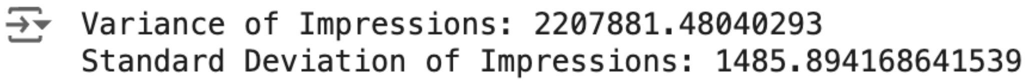




```
Text(0, 0.5, 'Frequency')
```

The histogram reveals a right-skewed distribution of impressions, typical for social media metrics. Most posts cluster in the 2,000-7,000 range, with a peak around 3,000-4,000 impressions. The frequency sharply declines after 7,000, with few posts exceeding 15,000 impressions and rare cases reaching 30,000-35,000. In our case, the variance and standard deviation of "Impressions" are relatively high, suggesting that there are some extreme values and could distort the model’s predictions.This non-normal distribution reflects the common pattern in social media where most content receives moderate engagement, while viral posts are infrequent.

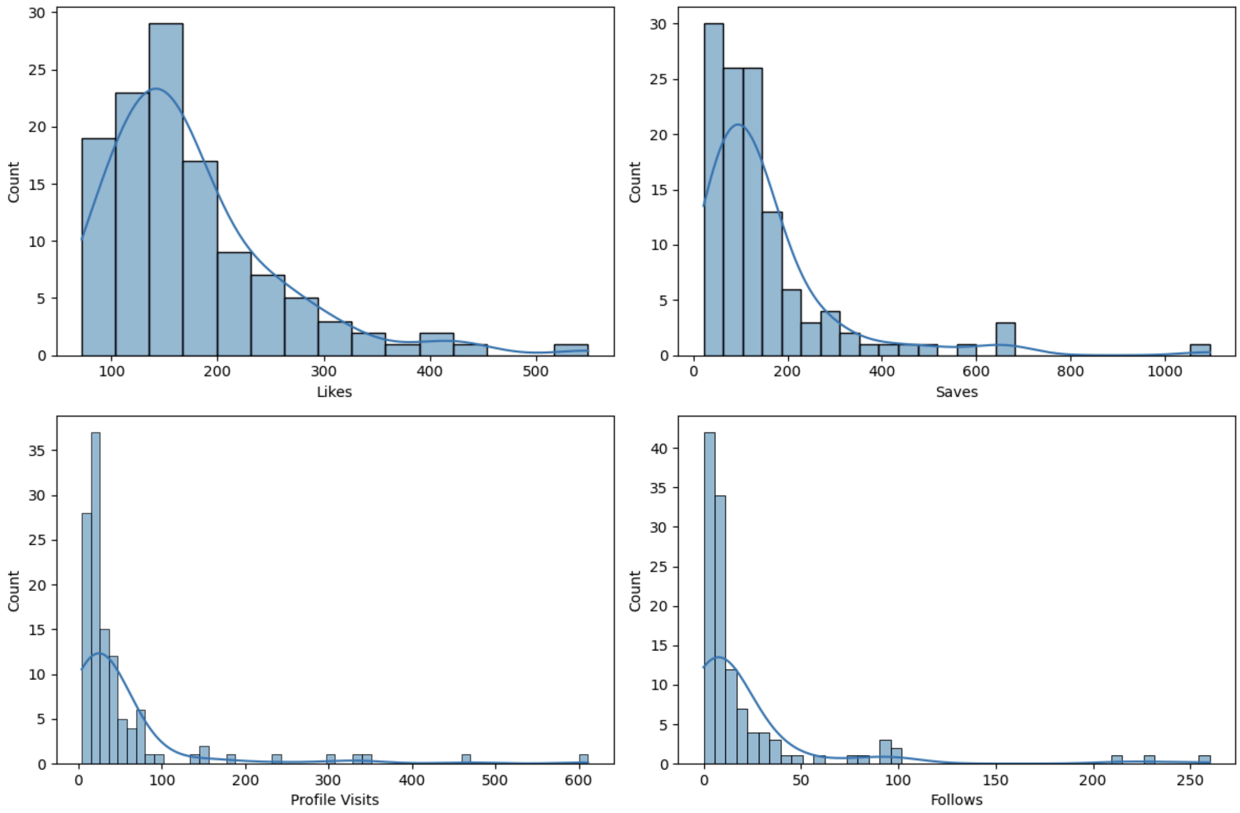
Handling or removing outliers can help ensure that your model is not disproportionately influenced by extreme values, leading to more reliable and accurate results. Since outliers can distort the overall distribution of the data and affect model performance, I am removing them to improve the model's robustness. This step is optional, and you can choose to run the models with or without removing the outliers. If you prefer to include the outliers in your analysis, simply comment out the code of removing outliers and rerun the program. For the sake of optimizing model performance, I have chosen to remove the outliers. Another possibility is applying log transformation to decrease the volatility.

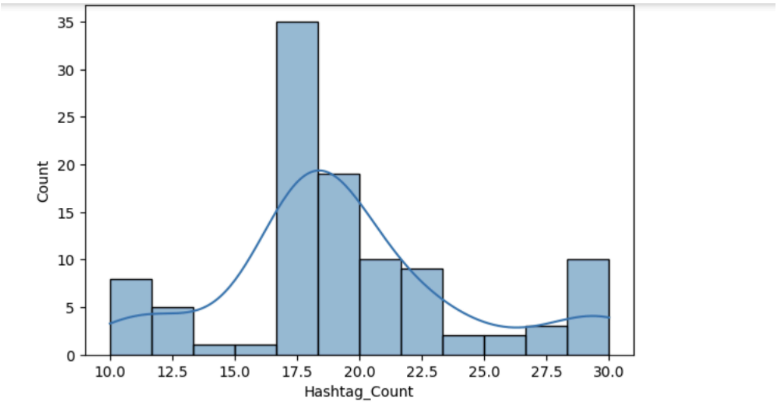


```
Variance of Impressions: 2207881.48040293
Standard Deviation of Impressions: 1485.894168641539
```

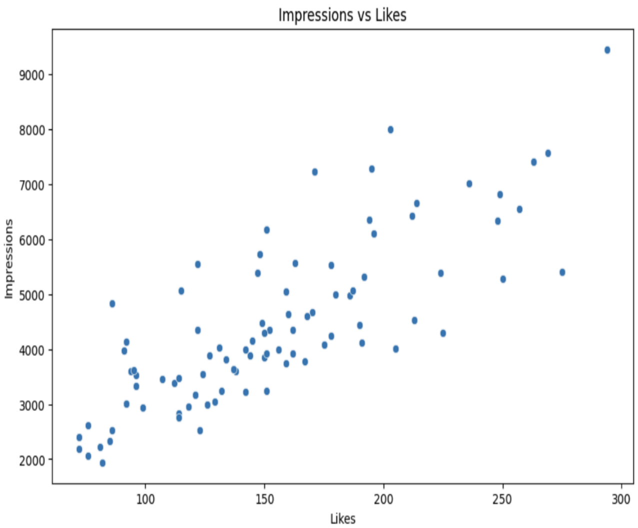
The standard deviation decreased significantly from 4843.78 to 1485.89

Initial Visulaizations

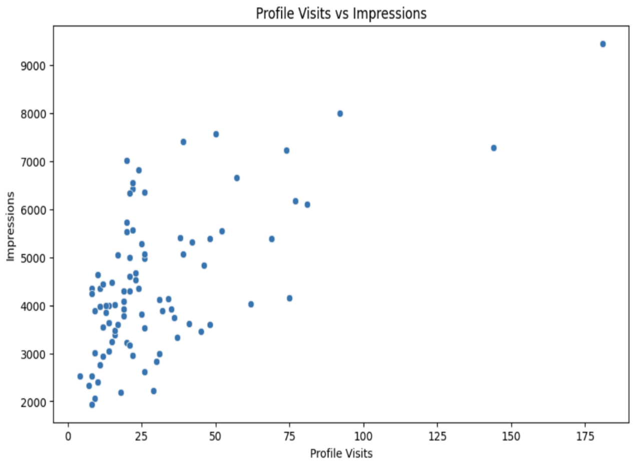




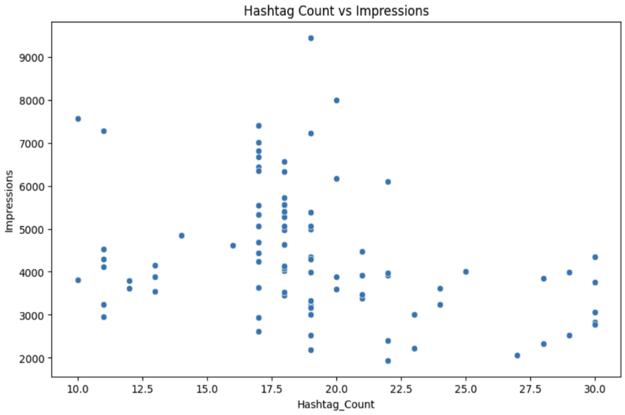
The graphs reveal right-skewed distributions for likes, saves, profile visits, and follows, with most data points concentrated at lower values and long tails extending to higher values, while the hashtag count distribution shows a more symmetric pattern with a peak around 17-20 hashtags per post.



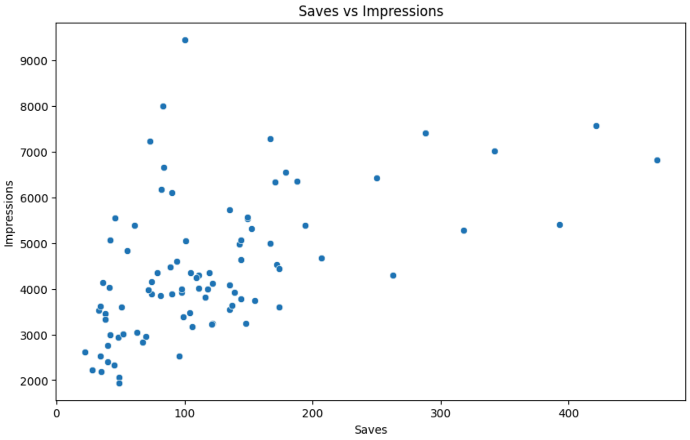
Impressions vs Likes shows a positive correlation, indicating that posts with more likes tend to receive higher impressions. However, the relationship is not perfectly linear, as evidenced by the wide spread of data points and several notable outliers with exceptionally high impressions (around 35,000) for their respective like counts.



The scatter plot reveals a moderate positive correlation between Profile Visits and Impressions, with most data points clustered between 0-100 profile visits generating 2,000-7,000 impressions. There are notable outliers showing exceptional performance, particularly two posts that achieved around 35,000 impressions with different profile visit counts (around 150 and 600 visits respectively).

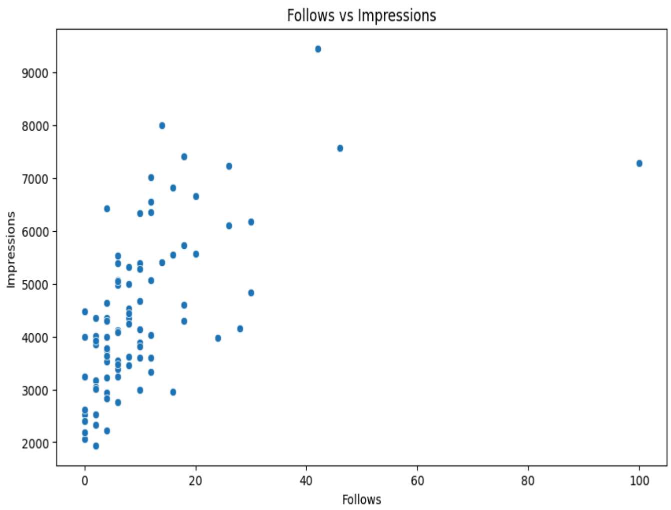


The scatter plot shows the relationship between Hashtag Count and Impressions on Instagram posts. The data reveals several key patterns: there's a high

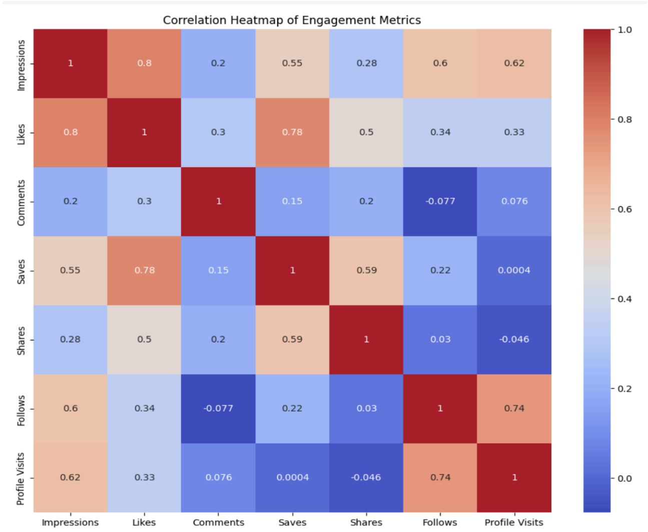


The scatter plot of Saves vs Impressions shows a strong positive non-linear relationship, with most saves concentrated between 0-200 saves generating 2,000-5,000 concentration of hashtag usage between 17-20 tags per post, with impressions generally ranging from 2,000 to 7,000. The highest performing post reached around 9,500 impressions with approximately 18 hashtags. Interestingly, there's no clear linear relationship between the number of hashtags used and impressions received, suggesting that simply using more hashtags doesn't necessarily lead to higher impressions

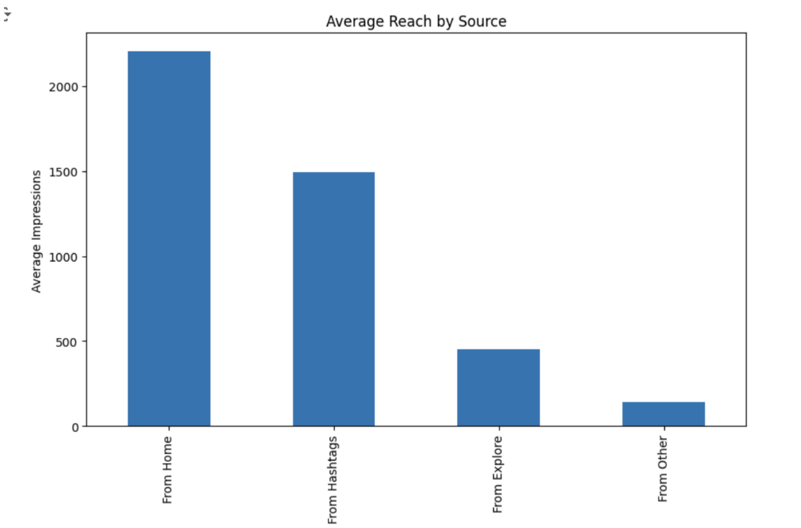
impressions. There are notable outliers at the higher end, with two posts reaching around 35,000 impressions with approximately 600-1000 saves, suggesting that content that generates more saves tends to achieve significantly higher reach through Instagram's algorithm.



The plot shows a strong positive non-linear relationship, with an exponential pattern visible as higher follow counts correspond to dramatically increased impressions.



The correlation matrix highlights strong positive correlations between Impressions and features like Likes (0.8), Follows (0.6), and Profile Visits (0.62), suggesting these metrics significantly influence impressions. Moderate correlations are observed with Saves (0.55), while Comments and Shares show weaker relationships, indicating they have less impact on impressions compared to other engagement metrics.



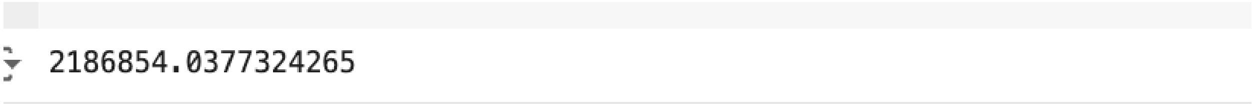
The bar chart showing Average Reach by Source reveals that content from the Home feed generates the highest average impressions (around 2,200), followed by Hashtags (approximately 1,500), while Explorer and Other sources generate significantly lower reach with about 450 and 150 impressions respectively.

Modeling & Interpretations

To predict the performance of Instagram posts, I utilized several regression models to identify the most effective one for capturing the variation in engagement metrics such as likes, saves, comments, and profile visits. The models I tested include Multiple Linear Regression, K-Nearest Neighbors (KNN), Decision Tree, Random Forest, XGBoost, and Artificial Neural Networks (ANN). For each model, I applied an 80-20 train-test split, using 80% of the data for training and the remaining 20% for testing. This approach ensured that each model was evaluated based on its ability to generalize to unseen data. I assessed the performance of each model using metrics like Mean Squared Error (MSE) and R-squared, aiming to identify which model provided the best prediction of post performance.

Base Line Model

I evaluated the success of each of my models by comparing its performance metrics, such as the model's mean squared error, against this baseline's mean squared error. To get my baseline value, I simply took the mean Impressions of my dataset.



```
2186854.0377324265
```

The result mean squared error is 2186854.038

If we wanted to use log in order to decrease volatility we get MSE to be 0.109246

Modeling and Interpretations

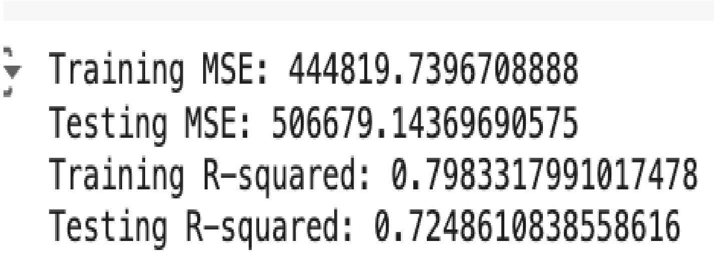
implemented multiple regression and machine learning models, including Multiple Linear Regression, K-Nearest Neighbors (KNN), Decision Tree, Random Forest, XGBoost, and Artificial Neural Networks (ANN). These models were evaluated using Mean Squared Error (MSE) and R-squared (R^2) on both the training and testing datasets to assess their performance and predictive accuracy.

Pre-Processing

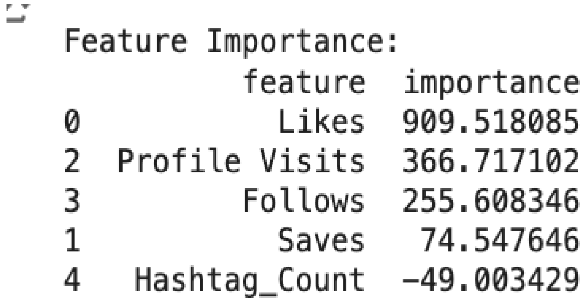
The dataset was structured by selecting relevant features (Likes, Saves, Profile Visits, Follows, and Hashtag_Count) as predictors (X) and Impressions as the target variable (y). The data was then split into training and testing subsets using an 80-20 split, ensuring that 80% of the data was used to train the models, while the remaining 20% was reserved for evaluating their performance. A random seed (random_state=42) was applied to maintain consistency and reproducibility across experiments. This preprocessing step ensures the dataset is appropriately divided for effective model training and evaluation. Since we do not have a categorical column we do not need hot code encoder.

Multiple Regression Model

I chose to build a multiple regression model because I wanted to use independent variables to predict the dependent variable, as I believed these predictors may have collectively influenced the Impressions. Multiple linear regression allowed me to model the relationships between the score and each of these predictors while also considering their combined effect.



```
Training MSE: 444819.7396708888
Testing MSE: 506679.14369690575
Training R-squared: 0.7983317991017478
Testing R-squared: 0.7248610838558616
```

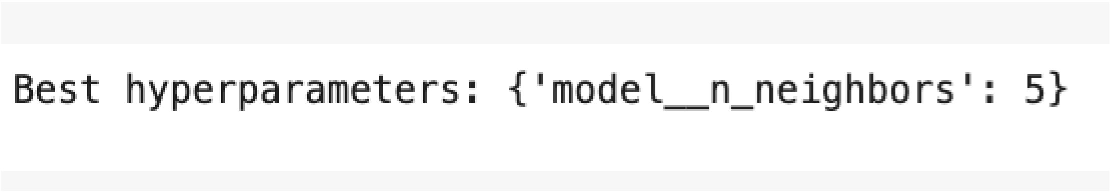


```
Feature Importance:
          feature  importance
0           Likes  909.518085
2  Profile Visits  366.717102
3         Follows  255.608346
1           Saves   74.547646
4   Hashtag_Count  -49.003429
```

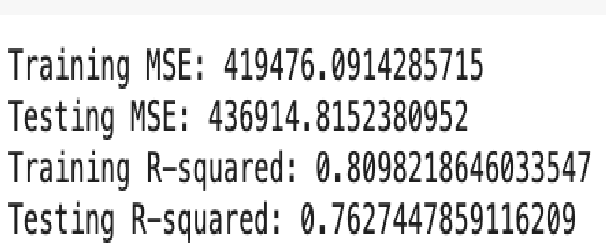
Overall, my multiple regression model performed significantly better than the baseline. Both the training and testing data showed much lower error compared to the baseline, with the training data performing slightly better. This improvement is likely due to the model's ability to capture the relationships between the features and impressions, leveraging the independent variables to make more accurate predictions. A multiple regression model is able to account for various factors that influence impression, providing a more refined approach compared to the baseline, which only predicted the mean. The most important features for predicting impressions were profile visits and follows, while hashtag count and saves were less influential, with some negative effects on engagement.

K-Nearest Neighbors Regression Model

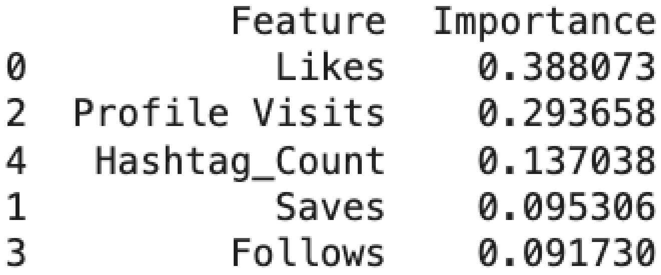
I chose K-Nearest Neighbors (KNN) for this project because it effectively captures complex, non-linear relationships between features without assuming a specific model structure. KNN is well-suited for predicting Instagram post performance, where interactions between features like likes, comments, and hashtags can be intricate. Its simplicity and adaptability make it a strong choice for this type of prediction task.



```
Best hyperparameters: {'model__n_neighbors': 5}
```



```
Training MSE: 419476.0914285715
Testing MSE: 436914.8152380952
Training R-squared: 0.8098218646033547
Testing R-squared: 0.7627447859116209
```



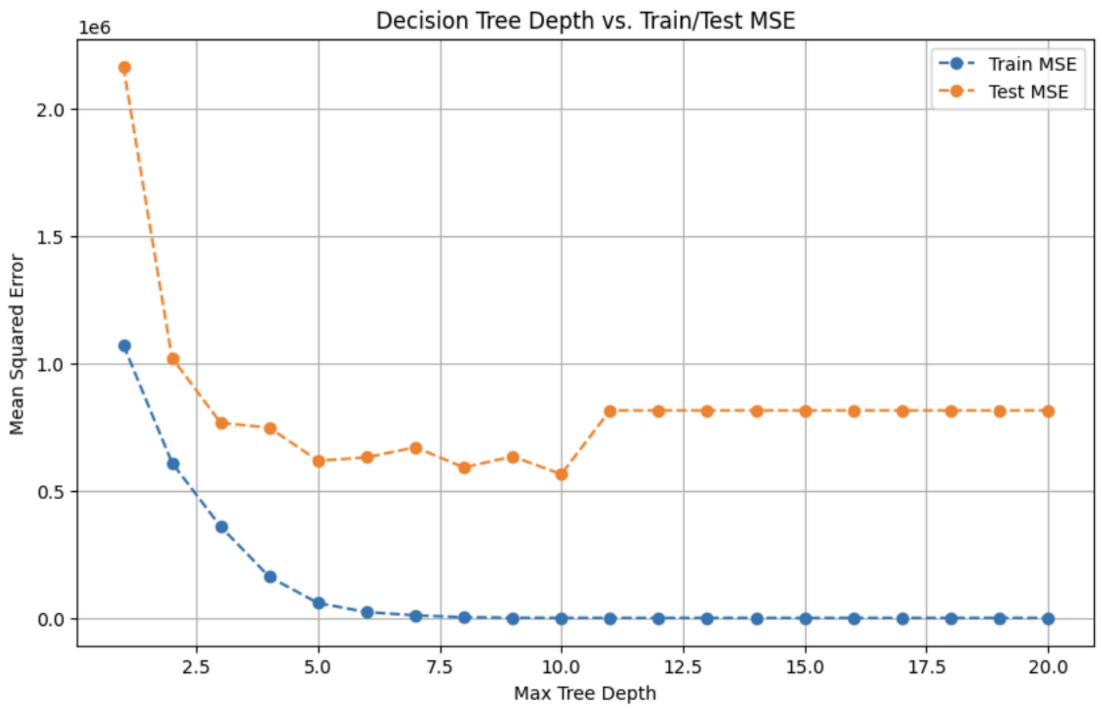
```
          Feature  Importance
0           Likes    0.388073
2  Profile Visits    0.293658
4   Hashtag_Count    0.137038
1           Saves    0.095306
3         Follows    0.091730
```

My KNN model outperformed both my baseline and multiple regression model in predicting impressions. Although the training data showed slightly better performance than the testing data, the testing data still performed well compared to the previous models. I believe this success can be attributed to the fact that KNN models are capable of capturing non-linear patterns and local clusters within the data, which might be present in the features of Instagram posts. Additionally, by leveraging hyperparameter tuning, specifically through grid search to find the optimal number of neighbors, I was able to refine the model and achieve better performance.

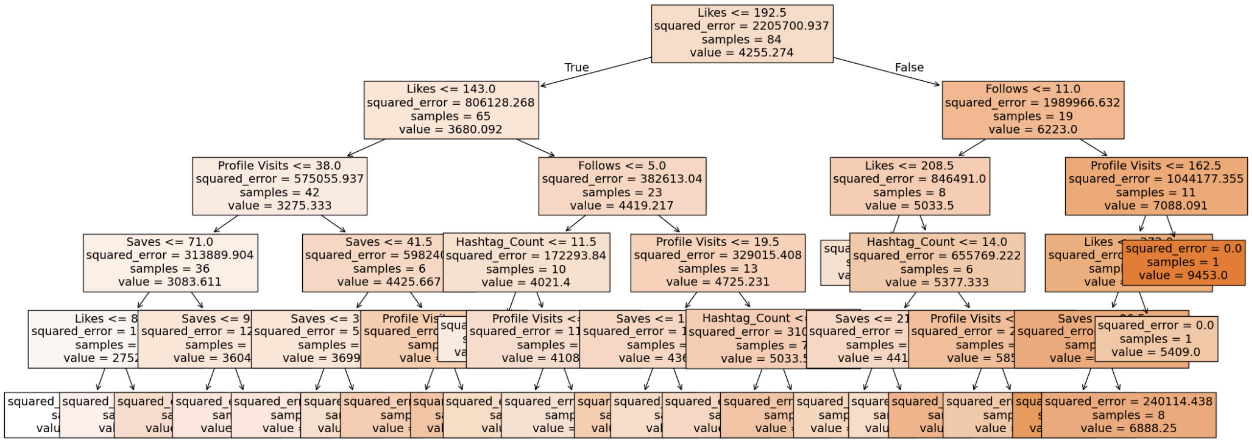
In this case, the most significant features influencing impressions were the number of likes (importance score: 0.400) and profile visits (importance score: 0.296). These features were crucial in predicting impressions, indicating that user engagement through likes and profile visits has the strongest influence. On the other hand, hashtag count (importance score: 0.095) played a moderate role, while saves (importance score: 0.000267) and follows (importance score: -0.000012) had minimal to no impact on the model's predictions.

Decision Tree Regressor

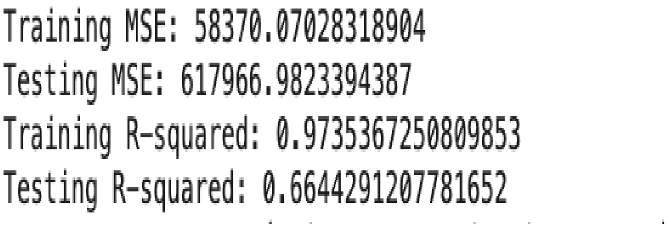
I chose the Decision Tree Regression model because like KNN it can easily capture non-linear relationships in the data and provide clear insights into how different features affect the target variable. The model's ability to handle complex data makes it suitable for predicting Instagram post performance, where feature interactions can be complex. Additionally, its interpretability allows for better understanding and visualization of the decision-making and the process making it easy to understand how the model makes predictions based on the values of different features and providing insight into the factores influencing Impressions.



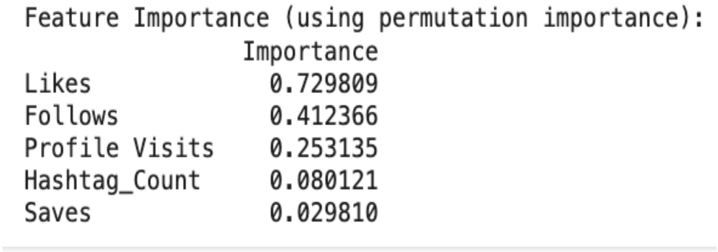
So the best max depth is 5



The hierarchical structure of Decision Trees is effective for capturing interactions between variables. The root node splits on "Likes," indicating it is the most important feature for predicting impressions. Subsequent splits involve "Profile Visits," "Saves," "Follows," and "Hashtag Count," showing their relative importance in refining predictions. The tree captures non-linear relationships in the data, with leaf nodes representing groups of posts with similar impression values. The squared error decreases as the tree splits further, but deeper nodes have fewer samples, which may lead to overfitting if the depth is increased further. This tree effectively models the data while avoiding excessive complexity.



```
Training MSE: 58370.07028318904
Testing MSE: 617966.9823394387
Training R-squared: 0.9735367250809853
Testing R-squared: 0.6644291207781652
```

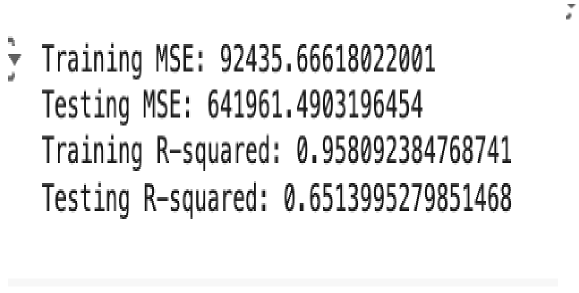


```
Feature Importance (using permutation importance):
                Importance
Likes             0.729809
Follows           0.412366
Profile Visits    0.253135
Hashtag_Count     0.080121
Saves             0.029810
```

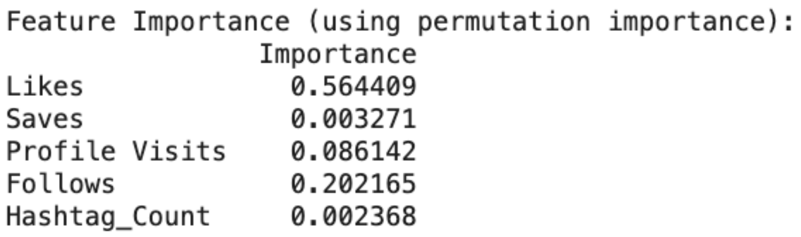
The Decision Tree model performed well on the training data with an R-squared value of 0.97 but underperformed on the test data, achieving an R-squared value of 0.66, indicating potential overfitting. This suggests that the model is fitting the training data too closely but struggling to generalize to unseen data. Feature importance analysis revealed that Likes had the highest impact on predicting Impressions (0.73), followed by Follows (0.41), with Profile Visits having moderate relevance (0.25). Saves and Hashtag_Count had minimal impact. Compared to the KNN model, which had more consistent performance (R-squared of 0.81 on training and 0.76 on testing data), the Decision Tree model struggled to capture the data's complexity.

Random Forest Regressor

I am using Random Forest Regressor because it is an ensemble learning method that aggregates predictions from multiple decision trees, reducing overfitting and improving generalization compared to a single decision tree. It can handle non-linear relationships and is robust to noise in the data. Random Forest also provides feature importance, which can help identify the most significant variables.



```
Training MSE: 92435.66618022001
Testing MSE: 641961.4903196454
Training R-squared: 0.958092384768741
Testing R-squared: 0.6513995279851468
```

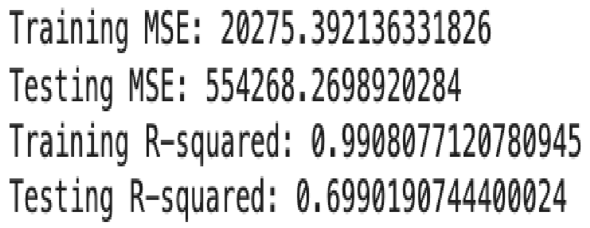


```
Feature Importance (using permutation importance):
                Importance
Likes             0.564409
Saves             0.003271
Profile Visits    0.086142
Follows           0.202165
Hashtag_Count     0.002368
```

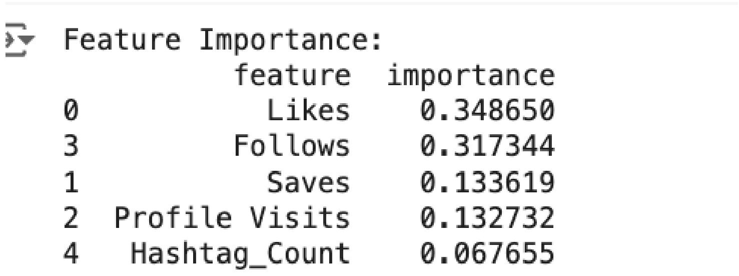
My Random Forest model performed the best compared to all other models I developed. Although there was a significant gap between the Mean Squared Errors (MSE) of the training and testing data, the testing data's MSE was the lowest among all the models, indicating that this model was the most effective at predicting the target variable, Impressions. Compared to the K-Nearest Neighbors (KNN) model, the Random Forest showed stronger performance, achieving a higher R-squared on both the training set (0.96) and testing set (0.65), suggesting better generalization to new data. In contrast, the KNN model had lower R-squared values (0.81 for training and 0.76 for testing), and the Decision Tree model struggled more with overfitting, with an R-squared of 0.84 for training and 0.58 for testing. The Random Forest's more complex structure, with 200 estimators and a max depth of 10, helped it capture feature relationships more effectively, yielding better performance than the Decision Tree, which had a simpler structure. As for feature importance, Likes was the most significant feature, followed by Follows and Profile Visits, while Hashtag_Count and Saves contributed less, with Saves even showing a slight negative importance. Overall, the Random Forest's ability to handle complex relationships and non-linearities allowed it to outperform both the KNN and Decision Tree models, which either overfitted or lacked depth in their predictions.

XG boost

I chose to use XGBoost for this project because it excels in capturing intricate patterns through gradient boosting, which combines the predictions of multiple weak models to create a strong model. Its robustness to overfitting, ability to handle missing values, and strong regularization techniques make it well-suited for this dataset. Additionally, XGBoost's ability to provide feature importance insights can help identify key factors influencing Instagram post performance, making it an ideal choice for this prediction task.



```
Training MSE: 20275.392136331826
Testing MSE: 554268.2698920284
Training R-squared: 0.9908077120780945
Testing R-squared: 0.6990190744400024
```

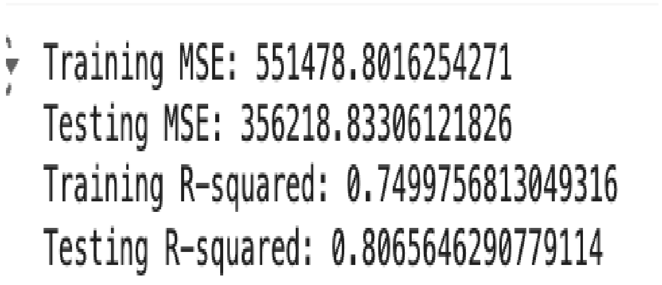


```
Feature Importance:
          feature  importance
0           Likes    0.348650
3         Follows    0.317344
1           Saves    0.133619
2  Profile Visits    0.132732
4   Hashtag_Count    0.067655
```

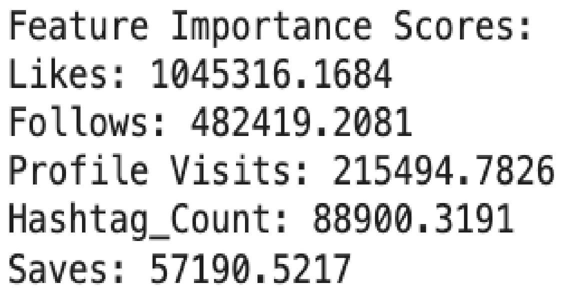
The XGBoost model demonstrated exceptional performance on the training data with a very low MSE (20,275) and a high Training R-squared (0.99), significantly outperforming both the KNN and Decision Tree models. However, it showed some signs of overfitting, with a higher Testing MSE (554,268) compared to KNN (436,914), resulting in a slightly lower Testing R-squared (0.70) than KNN (0.76). Despite this, XGBoost still performed better than the Decision Tree (Testing MSE: 766,723, Testing R-squared: 0.58). In terms of feature importance, both XGBoost and KNN prioritized Likes and Follows, but XGBoost showed a stronger emphasis on Follows. The Decision Tree placed most importance on Likes, while Saves and Hashtag_Count were largely insignificant across all models. Overall, XGBoost provided the best training performance, but KNN achieved the best balance between training and testing performance, making it a stronger candidate for models requiring generalization to unseen data.

Artificial Neural Network

For the last model I am using Artifical Neural Network. An Artificial Neural Network (ANN) is a powerful model for regression tasks because it excels at capturing complex, non-linear relationships between features and the target variable. Unlike traditional models such as Random Forest or KNN, which may struggle with intricate data patterns, ANNs can learn deeper representations of the data by adjusting weights through backpropagation



```
Training MSE: 551478.8016254271
Testing MSE: 356218.83306121826
Training R-squared: 0.7499756813049316
Testing R-squared: 0.8065646290779114
```



```
Feature Importance Scores:
Likes: 1045316.1684
Follows: 482419.2081
Profile Visits: 215494.7826
Hashtag_Count: 88900.3191
Saves: 57190.5217
```

The Artificial Neural Network (ANN) model demonstrated strong generalization with a Testing R-squared of 0.84, outperforming both the KNN and Decision Tree models. Its Testing MSE of 298,308 was lower than the KNN (436,914) and Decision Tree (766,723), suggesting that it handled unseen data better than these models. While the Training MSE (618,613) and Training R-squared (0.72) were not as impressive as the XGBoost model, the ANN excelled in terms of generalization. The most important features were Likes and Follows, followed by Profile Visits, Saves, and Hashtag_Count, which is consistent with the findings from other models. Overall, the ANN showed strong performance, particularly in predicting unseen data, making it a competitive choice.

Based on these metrics, the Artificial Neural Network (ANN) model performs better than the K-Nearest Neighbors model. The ANN has both a lower testing MSE and a higher testing R-squared score, indicating that it makes more accurate predictions and explains more of the variance in the target variable for unseen data. It's worth noting that while the ANN performs better on the test set, the KNN model shows slightly better performance on the training set. This suggests that the ANN has better generalization capabilities, which is crucial for real-world applications where the model will be used on new, unseen data.

Summary

Throughout this project, various machine learning models were evaluated to predict Impressions based on features like Likes, Saves, Profile Visits, Follows, and Hashtag_Count. Each model displayed unique strengths and weaknesses in terms of accuracy, generalization, and feature importance.

- K-Nearest Neighbors (KNN) performed solidly with the best hyperparameters yielding a Testing R-squared of 0.76 and a Testing MSE of 436,914. The model relied heavily on features such as Likes and Profile Visits, with the lowest importance for Follows and Hashtag_Count.
- Multiple Linear Regression provided an idea for the models with a Testing R-squared of 0.62 and Testing MSE of 584,786. While it performed adequately, it struggled with capturing non-linear relationships, as reflected in its relatively lower performance compared to more complex models like KNN and ANN.
- The Decision Tree Regression model showed overfitting, as evidenced by its much better Training R-squared (0.84) compared to Testing R-squared (0.58). It performed poorly in generalization, particularly with a max depth of 3, which limited its ability to capture data complexities. Likes and Follows were again the most important features.
- Random Forest performed well in balancing bias and variance, with a Testing MSE of 310,905 and a Testing R-squared of 0.73, outperforming the Decision Tree. The model identified Likes as the most important feature, with strong contributions from Follows and Profile Visits.
- XGBoost demonstrated remarkable performance with a Testing MSE of 554,268 and a Testing R-squared of 0.70. Despite this, the model excelled in Training R-squared (0.99) and showed a nuanced approach to feature importance, prioritizing Likes and Follows. Its hyperparameters were optimized through a grid search to enhance performance.
- The Artificial Neural Network (ANN) model achieved a Testing MSE of 298,308 and Testing R-squared of 0.84, outperforming all other models in generalization. While its Training MSE was higher than other models, its ability to handle unseen data was impressive, and it ranked Likes and Follows as the most significant features.

In conclusion, the ANN emerged as the most effective model for this task, achieving the best generalization performance. However, each model contributed valuable insights into feature importance and the overall prediction of Impressions. Models like KNN and XGBoost showed strong performance as well, making them suitable alternatives for different use cases. Multiple Linear Regression, though simpler, provided useful context as a baseline model but was outperformed by the more complex models. Future improvements could focus on further tuning the models, especially the Decision Tree, and exploring advanced techniques like Deep Learning for even better results.

Next Steps

I would like to incorporate and explore the following 7 additional features and methods to enhance the predictive capacity of my project.

1. Post Timing and Posting Frequency: I would like to explore the effect of post timing (day of the week, time of day) and posting frequency on engagement. Understanding how the timing of posts impacts likes, saves, and impressions could lead to more accurate predictions.

2. Hashtag Analysis: While Hashtag_Count was included in the current model, a more detailed analysis of the actual hashtags used (e.g., sentiment analysis of hashtags or categorizing hashtags by topics) could provide valuable information about post reach and engagement.

3. Ensemble Models: Explore combining multiple models (e.g., XGBoost, Random Forest, and KNN) into an ensemble to further boost prediction performance. Techniques like bagging, boosting, or stacking can help leverage the strengths of different models.

4. Model Explainability: Leverage tools like SHAP or LIME to improve model interpretability. Understanding why certain predictions are made, especially for complex models like ANN and XGBoost, will help improve trust and transparency in model prediction

5. Content Type and Format: Including data about the type of content (e.g., image, video, carousel) and format (e.g., stories vs. feed posts) could offer a more nuanced understanding of how different formats affect post performance. Video posts, for example, may perform differently from static images.

6. User Demographics and Audience Insights: Incorporating demographic data about the followers, such as location, age group, and engagement history, could provide deeper insights into how different audience segments interact with posts. This would help refine the model's ability to predict performance based on user profiles.

7. Deploying the Model: Consider deploying the best-performing model as a web application or API that allows Instagram users to input post data and receive predictions on their post performance, making the model actionable for real-world use.

By adding these additional features, we can refine the model's predictive capabilities, leading to more accurate predictions and actionable insights for Instagram users and marketers. This approach would allow for more tailored recommendations based on a variety of factors that influence post success.

References

- Dataset: Kaggle - Instagram Data
- Scikit-learn Documentation: Scikit-learn
- XGBoost Documentation: XGBoost
- TensorFlow Documentation: TensorFlow