Week 6: First Modeling Approach

Yujia Cao, Wendi Yuan, Yuhan Zhao

Each team member has developed a distinct model tailored to our dataset and research objectives. Yujia worked on a Support Vector Machine, Wendi developed a Random Forest regression model, and Yuhan implemented an XGBoost model. Given the unique strengths of each approach, we plan to select the model with the best performance for our primary analysis. Before making this choice, however, we will evaluate and compare performance metrics, such as those on the test dataset, across all three models to ensure a well-informed selection.

Random Forest Regressor:

The Random Forest Regressor is an ideal choice for predicting the number of days until a video trends due to its robustness against overfitting and ability to handle non-linear relationships among predictors. By combining multiple decision trees, it effectively captures complex patterns in the data, such as the influence of *likes*, *comment_count*, and sentiment scores on video visibility. Additionally, it provides insights into feature importance, allowing us to identify which factors most significantly impact trending potential. There are several reasons that a random forest regressor could fit the prediction task. The relationship between features and trending days is likely non-linear. For example, an increase in likes might not always correspond to a proportional increase in trending days. Random Forest, as an ensemble of decision trees, can capture such non-linear patterns effectively, providing more accurate predictions. Then, it provides insights into which features are the most important for making predictions. Knowing which features contribute most allows for targeted strategies, like focusing on increasing comments or likes to boost a video's trend duration. Focusing on predicting the number of days until a video trends is crucial for both the advertising industry and content creators on platforms like YouTube. For advertisers, understanding this timeline helps optimize ad placements and strategies, ensuring they capitalize on peak engagement periods. For YouTubers, timely insights can guide content creation and promotion strategies; if a video fails to receive favorable reviews within a critical time frame, it may struggle to gain traction despite eventual increases in views. This predictive capability allows creators to adjust their approaches proactively, potentially improving their content's visibility and overall success.

The hyperparameters are evaluated as showing in Appx. 1,2. For Model 1, which is the default setting, this model may show a balance between training and test performance, indicating that it generalizes well without overfitting.After the default setting of model 1, we increase the n-estimators of Model 2 to 200. Increasing the number of trees can improve both training and test performance, but if the

test MSE decreases less than the training MSE, it may suggest overfitting.For Model 3, the max_depth is designed to be 10. This model may show a reduction in training performance (higher training MSE) compared to others, but ideally, it should have a lower test MSE, indicating better generalization due to regularization. Metrics perform different across train and test datasets. Model 1 training metrics show moderate performance, indicating that the default parameters provide a good baseline.The test metrics(MSE) might be slightly higher than training metrics, suggesting the model has room for improvement.

Model 2 training metrics has the rather lower training MSE due to the increased number of trees.And its test Metrics shows improved test MSE compared to Model 1 indicates that increasing trees helped to reduce variance and improve generalization. Model 3 Training MSE increased due to the restricted complexity of the model.For the testing metrics, test MSE should decrease compared to Models 1 and 2, showcasing better generalization and suggesting that restricting tree depth was beneficial. Since the target variable is the number of days a video remains trending, MSE assesses the precision of the model's predictions, which is important when forecasting time-sensitive variables like trending durations on YouTube. In the context of predicting YouTube trending days, R^2 helps evaluate how well selected features including likes, comment count, and views collectively explain the variation in the number of days a video trends.A high R^2 would imply that these variables are relatively strong in capturing the patterns of user engagement and video popularity.

The model comparison is showing in Appx.3, and the winning model from those 3 variations of Random Forest Regression is

Winning Model:

	Model	Train MSE	Test MSE	Train R^2	Test R^2
1	Model 2 (n_estimators=200)	1284.180778	7310.769158	0.967564	0.811753

After analyzing the performance metrics of the three variations of the Random Forest model, Model 2 (n_estimators=200) is identified as the best model based on the following criteria:

Test MSE:Model 2 achieved the lowest test Mean Squared Error (MSE) at 7310.77. This indicates that its predictions were, on average, closer to the actual values compared to the other models, thereby showing better accuracy.

Test R^2:Model 2 also had the highest test R-squared (R^2) value of 0.812. This suggests that approximately 81.2% of the variance in the target variable (the number of days to trend) can be explained by the model, which is a strong indicator of its predictive power.

XGBoost:

XGBoost, a powerful gradient boosting algorithm, was selected for this project due to its effectiveness with structured/tabular data and its capability to model complex interactions between features. This approach is particularly advantageous for analyzing YouTube metrics, where features like sentiment analysis scores and engagement metrics can provide meaningful insights into user behavior and preferences. XGBoost's robust handling of large datasets, its built-in regularization to prevent overfitting, and its customizable hyperparameters make it highly suitable for a task that involves optimizing predictive accuracy while avoiding excessive model complexity.

Each model variation was tuned to balance accuracy, generalization, and computational efficiency. XGBoost builds an ensemble of decision trees in sequence, with each tree improving upon the errors of the previous ones. This approach increases predictive power but can lead to complexity if hyperparameters are not carefully managed. The hyperparameters evaluated were(Appx.4):

> Learning Rate: Controls the weight of each new tree added to the model. Lower learning rates typically increase model stability and may improve accuracy, but they require more boosting rounds. Learning rates of 0.1, 0.05, and 0.01 were tested to find a rate that promotes stability without sacrificing predictive power.
>
> Number of Estimators (n_estimators): Defines the number of trees in the ensemble. While more rounds tend to improve accuracy, they can also increase overfitting risk. Values of 100, 200, and 300 were tested to find an optimal count.
>
> Max Depth: Determines tree depth, which affects model complexity. Deeper trees may capture more data intricacies but also risk overfitting. Depths of 4, 6, and 8 were explored to assess performance.

Before fitting the XGBoost model, the dataset underwent significant preprocessing to focus on structured features. Text columns were excluded, directing attention toward quantitative and categorical features like sentiment scores and engagement metrics, which are more immediately useful for predictive modeling. One-hot encoding was applied to categorical columns, such as *trending_day_of_the_week* and sentiment categories, to represent these attributes as binary features. These transformations retained key information relevant to engagement without introducing unnecessary complexity. Time-related variables, while omitted from the primary analysis, could reveal seasonal or time-based trends in future work.

Three variations of the XGBoost model were tested, each with distinct hyperparameter settings(Appx.4):Variation 1: Lower learning rate and max depth were used, yielding stable RMSE values but with a potential for underfitting. This setup provided a solid baseline for model comparison. Variation 2: This model combined a moderate learning rate, max depth, and n_estimators, balancing accuracy with generalization. It yielded the lowest test RMSE and highest R^2, demonstrating its ability to capture data intricacies without overfitting. Variation 3: Higher max depth and a larger number of trees were tested, which improved training accuracy but slightly reduced test performance, indicating a risk of overfitting.

The test results demonstrated that Variation 2 offered the best performance, achieving a test RMSE of 2.909 million and an R^2 score of 0.9608. This indicates that the model captures approximately 96% of the variance in the test data, making it the most suitable choice for predictive analysis. The combination of a 0.05 learning rate, 200 estimators, and a max depth of 6 balanced complexity with predictive accuracy, supporting effective generalization on new data. Variation 2's low RMSE indicates strong predictive accuracy, especially in forecasting actual engagement metrics for YouTube videos. The high R^2 score confirms the model's effectiveness in capturing key relationships within the data.

```
Best Model Variation: 2
Hyperparameters: {'learning_rate': 0.05, 'n_estimators': 200, 'max_depth': 6}
Variation        Variation 2
Train RMSE    2120971.173631
Test RMSE     2909245.576456
Train R^2           0.978047
Test R^2            0.960791
Name: 1, dtype: object
```

By identifying Variation 2 as the best-performing model, this XGBoost approach sets the foundation for further research into specific features driving engagement. For example, future analysis could examine sentiment or engagement patterns more closely, potentially informing strategies for optimizing content reach and impact. This model's insights could help content creators and marketers understand which types of video characteristics are associated with higher engagement, ultimately guiding content development strategies for improved audience engagement on platforms like YouTube.

Support Vector Machine(SVM):

By using the SVM model, we evaluated the SVM model using SVR with an RBF kernel. The key metrics used to assess model performance were RMSE and R^2 score. The RMSE value of 15,049,834 indicates that the predictions made by the model deviate significantly from the actual values, suggesting a lack of accuracy. Additionally, the R^2 score of -0.0492 shows that the model performs worse than a simple baseline prediction of the mean, which is highly undesirable.

The poor performance of the SVM model in this case indicates that it fails to capture the underlying patterns in the data. A negative R^2 score highlights that the model is not suitable for making reliable predictions, as it cannot generalize well to unseen data. Despite the SVM model's capability of capturing non-linear relationships with the RBF kernel, the results show that it does not work well with this specific dataset, likely due to the complexity of the data or insufficient feature representation.

Given the large prediction error and negative R^2 value, we conclude that the SVM model is not an ideal choice for this task. The complexity of tuning an SVM model, especially with non-linear kernels like RBF, does not justify the poor results obtained. Alternative models such as XGBoost and Random Forest, which tend to perform better with tabular and complex datasets, should be prioritized for further analysis and optimization.

```
RMSE: 15049834.690751504
R^2 Score: -0.04926450015757444
```

Hyperparameters and Model Variations:

- **Variation 1**: SVR with Radial Basis Function (RBF) kernel, C=1.0C=1.0, ϵ=0.1ϵ=0.1
- **Variation 2**: SVR with RBF kernel, C=10.0C=10.0, ϵ=0.2ϵ=0.2
- **Variation 3**: SVR with Linear kernel, C=1.0C=1.0, ϵ=0.1ϵ=0.1

```
Summary of all variations:
         Model  RMSE (Train)  R² (Train)  RMSE (Validation)  R² (Validation)
0  Variation 1  1.461203e+07   -0.041947       1.496831e+07        -0.037927
1  Variation 2  1.399023e+07    0.044844       1.431974e+07         0.050069
2  Variation 3  1.461203e+07   -0.041947       1.496831e+07        -0.037927
```

Based on the evaluation of three different SVR model variations, we observed notable differences in their performance. Variation 1 and Variation 3, both with lower regularization (C=1.0C=1.0) and different kernels (RBF and Linear, respectively), showed poor results. Both had similar RMSE values for training and validation datasets, at around 14.6 million for training and 14.9 million for validation. The negative R2R2 values for these variations indicate that these models perform worse than a simple mean prediction, suggesting they failed to capture meaningful patterns in the data.

Variation 2, which used an RBF kernel with a higher regularization value (C=10.0C=10.0), demonstrated slightly better performance. The RMSE values for training (13.9 million) and validation (14.3 million) were lower than the other two variations, and the R2R2 values were modestly positive at 0.0448 for training and 0.0501 for validation, indicating the model's ability to explain some of the

variance in the data. While the improvement is marginal, this suggests that increasing the regularization strength helped the model generalize better, though overall predictive performance remains low. Further hyperparameter tuning or exploring alternative models might yield better results(Appx. 6).

Based on the detailed analysis of performance metrics for each model—Random Forest, XGBoost, and Support Vector Machine (SVM)—XGBoost's Variation 2 appears to be the most suitable choice for predictive task, due to its strong balance between predictive accuracy and generalization. XGBoost Variation 2 achieved the best performance in terms of test RMSE (2.909 million) and R^2 (0.9608). This indicates that the model is highly accurate in predicting engagement metrics and can explain 96% of the variance in the target variable, which is impressive for a task involving complex, user-driven data.

Appendix

```python
# Variation 1: Default settings
rf_model1 = RandomForestRegressor(random_state=42)
rf_model1.fit(X_train_encoded, y_train)

# Variation 2: Increased number of trees
rf_model2 = RandomForestRegressor(n_estimators=200, random_state=42)
rf_model2.fit(X_train_encoded, y_train)

# Variation 3: Increased depth of trees
rf_model3 = RandomForestRegressor(max_depth=10, random_state=42)
rf_model3.fit(X_train_encoded, y_train)
```

Appx. 1

```
Model 1 (Default settings): {'train_mse': 1304.8170948196043, 'test_mse': 7429.233070650234, 'train_r2': 0.96704324
35550833, 'test_r2': 0.8087028678543599}
Model 2 (n_estimators=200): {'train_mse': 1284.1807777766226, 'test_mse': 7310.769158069089, 'train_r2': 0.96756447
06890668, 'test_r2': 0.8117532239979909}
Model 3 (max_depth=10): {'train_mse': 4769.834525959002, 'test_mse': 9669.919422254534, 'train_r2': 0.8795246664235
963, 'test_r2': 0.7510068891411407}
```

Appx.2

```
Performance Comparison of Models:
                         Model    Train MSE     Test MSE  Train R²   Test R²
0           Model 1 (Default)  1304.817095  7429.233071  0.967043  0.808703
1  Model 2 (n_estimators=200)  1284.180778  7310.769158  0.967564  0.811753
2      Model 3 (max_depth=10)  4769.834526  9669.919422  0.879525  0.751007
```

Appx. 3

```python
# Define hyperparameter variations
variations = [
    {"learning_rate": 0.1, "n_estimators": 100, "max_depth": 4},
    {"learning_rate": 0.05, "n_estimators": 200, "max_depth": 6},
    {"learning_rate": 0.01, "n_estimators": 300, "max_depth": 8}
]
```

Appx. 4

```
Comparison of XGBoost Model Variations:
     Variation    Train RMSE     Test RMSE  Train R^2  Test R^2
0  Variation 1  3.425430e+06  4.007967e+06   0.942740  0.925583
1  Variation 2  2.120971e+06  2.909246e+06   0.978047  0.960791
2  Variation 3  2.248698e+06  3.123748e+06   0.975323  0.954796
```

Appx. 5

```
RMSE: 15049834.690751504
R^2 Score: -0.04926450015757444
```

Appx. 6

```
Model: Variation 1
RMSE (Train): 14612029.7497, R² (Train): -0.0419
RMSE (Validation): 14968307.7158, R² (Validation): -0.0379

Model: Variation 2
RMSE (Train): 13990228.1189, R² (Train): 0.0448
RMSE (Validation): 14319743.9476, R² (Validation): 0.0501

Model: Variation 3
RMSE (Train): 14612029.7497, R² (Train): -0.0419
RMSE (Validation): 14968307.7158, R² (Validation): -0.0379

Summary of all variations:
         Model  RMSE (Train)  R² (Train)  RMSE (Validation)  R² (Validation)
0  Variation 1  1.461203e+07   -0.041947       1.496831e+07        -0.037927
1  Variation 2  1.399023e+07    0.044844       1.431974e+07         0.050069
2  Variation 3  1.461203e+07   -0.041947       1.496831e+07        -0.037927
```