**Mahmut Semih Güner 190101033**

**Meme Popularity Decision Tree Classification Report**

**1. Introduction**

Memes are cultural elements, typically in the form of images, videos, or text, that spread rapidly and often humorously on the internet. These digital artifacts convey a particular idea, behavior, or style that is easily imitated and shared among users. The primary goal of this project is to analyze social media content, specifically memes, using a decision tree model. The project focuses on understanding the factors that contribute to the popularity of memes.

In this report, I present the implementation and results of a decision tree classifier that I collected data for Meme popularity based on the . The decision tree was trained using the training data in ml3\_train.xlsx, and the classification results were obtained for the test data is obtained using k-fold.  
I used 4 folds (26 entry per fold) in order to get the output. The target variable for prediction is whether an entry is popular or not.

**2. Data Preprocessing**

The dataset is preprocessed through label encoding for categorical features. The following labels are defined and mapped to corresponding indices for different features:

* Humor Type: Absurdism, Dark, Sarcasm/Irony, Rage-Bait, Wholesome, Satirical, Dank, Relatable
* Format: Photo, Video, Text
* Featuring Song: Yes , No
* Visual Features: Multiple\_Features, Reaction, Absurd, Animals, Pop\_Culture, Surreal, Deep\_Fried
* OP Platform: Twitter(X), Instagram, TikTok, Unknown, Youtube, FaceBook, Reddit
* Is Popular (target variable): Yes, No

**3. Model Training**

A Decision Tree Classifier with entropy as the criterion is utilized for the popularity prediction. The model is trained using K-fold cross-validation with k=4 folds. For each fold, the training and testing sets are extracted, the model is trained, and predictions are made on the test set. The results, including descriptions, predictions, confusion matrix, and classification report, are printed for each fold. Libraries used include;  
  
pandas: To get the excel file   
sklearn.tree:To train the dataset and create the tree model.  
sklearn.model\_selection: To make cross validation , Value Prediction and K-Fold  
sklearn.metrics: To calculate confusion matrix.  
matplotlib: To visualize the tree.

**4. Results and Analysis**

* Descriptions and Predictions: The code prints the descriptions and predictions for each entry in the test set for every fold. Classification Report: The classification report, including precision, recall, and F1-score, is printed for each fold.
* Confusion Matrix: The confusion matrix is calculated and printed for each fold, providing insight into the model's performance.
* Predictions: The model's predictions for each fold are printed.

Confusion Matrix - Fold 2:

[[9 5]

[8 4]]

Classification Report - Fold 2:

precision recall f1-score support

0 0.53 0.64 0.58 14

1 0.44 0.33 0.38 12

accuracy 0.50 26

macro avg 0.49 0.49 0.48 26

weighted avg 0.49 0.50 0.49 26

Confusion Matrix - Fold 1:

[[8 4]

[7 8]]

Classification Report - Fold 1:

precision recall f1-score support

0 0.53 0.67 0.59 12

1 0.67 0.53 0.59 15

accuracy 0.59 27

macro avg 0.60 0.60 0.59 27

weighted avg 0.61 0.59 0.59 27

*Average Accuracy Across Folds: 0.6001*

Confusion Matrix - Fold 4:

[[11 6]

[ 4 5]]

Classification Report - Fold 4:

precision recall f1-score support

0 0.73 0.65 0.69 17

1 0.45 0.56 0.50 9

accuracy 0.62 26

macro avg 0.59 0.60 0.59 26

weighted avg 0.64 0.62 0.62 26

Confusion Matrix - Fold 3:

[[10 3]

[ 5 8]]

Classification Report - Fold 3:

precision recall f1-score support

0 0.67 0.77 0.71 13

1 0.73 0.62 0.67 13

accuracy 0.69 26

macro avg 0.70 0.69 0.69 26

weighted avg 0.70 0.69 0.69 26

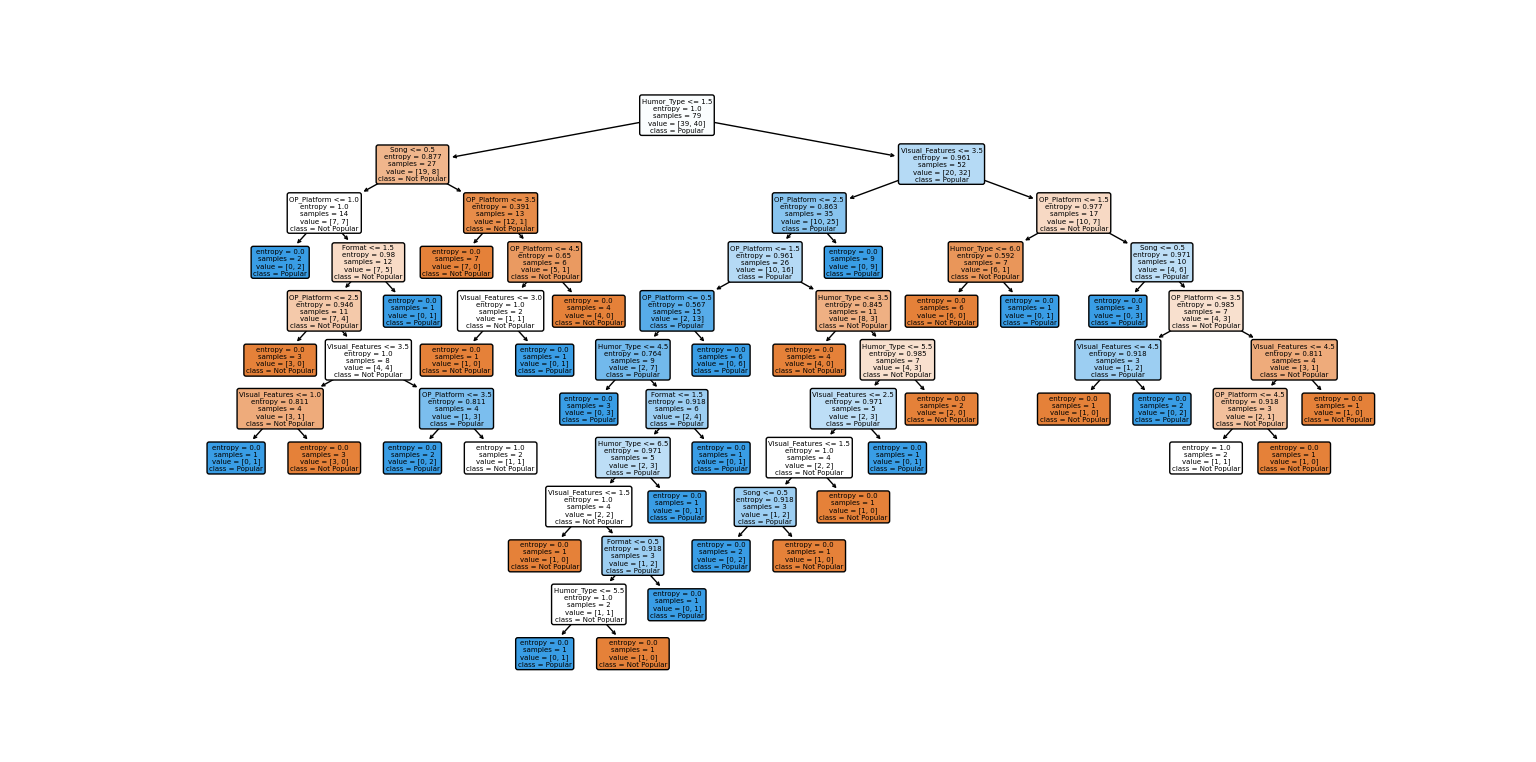
\*0’s and 1’s in the first parts stand for the individual values of the target variables.

* **Precision**: The ratio of correctly predicted positive observations to the total predicted positives. Precision for class 0 is 0.53, and for class 1 is 0.44.
* **Recall (Sensitivity)**: The ratio of correctly predicted positive observations to all the observations in the actual class. Recall for class 0 is 0.64, and for class 1 is 0.33.
* **F1-Score**: The weighted average of precision and recall. It's a balance between precision and recall. F1-score for class 0 is 0.58, and for class 1 is 0.38.
* **Support**: The number of actual occurrences of the class in the specified dataset.
* **Accuracy**: The ratio of correctly predicted instances to the total instances. Accuracy is 0.50.
* **Macro Avg**: The average of precision, recall, and F1-score across both classes.
* **Weighted Avg**: The weighted average of precision, recall, and F1-score, where each class's score is weighted by its support.

**5. Predictions on Entire Dataset**

After the K-fold cross-validation, the model is used to predict the entire dataset. Predictions for the entire dataset are printed, along with corresponding entries.

**6. Decision Tree Visualization**

The decision tree is visualized using the plot\_tree function from scikit-learn, providing an illustrative representation of the model.

**7. Conclusion**

In conclusion, this project delved into the intriguing realm of memes, seeking to decipher the elements that contribute to their popularity on social media platforms. We embarked on a journey through the evolution of memes, explored the dataset's features, and trained a decision tree model to predict their popularity.