CS412 – Machine Learning | InstaInfluencers

# Project Overview

This project is focused on detecting the content type of Instagram accounts and predicting their like counts. The goal is to improve the classification accuracy and reduce regression error in predicting the number of likes through three rounds of model improvements. Initially, we used SVM (Support Vector Machine) to address the problem, then tried ensemble learning techniques, and in the final round, we experimented with different models before concluding with the SVM model as the most effective solution.

# Round 1: SVM-based Classification

## Model Used:

Support Vector Machine (SVM)

## Results:

Classification Accuracy: 77.43% (Rank 44/141)  
F1-Weighted Score: 77.13% (Rank 44/141)  
Regression Error: 727.46 (Rank 13/141)

## Improvements Made:

In this round, we focused on classification, achieving a reasonable accuracy of 77.43%. The regression error was also fairly low, indicating that SVM was effective in making predictions based on the dataset.

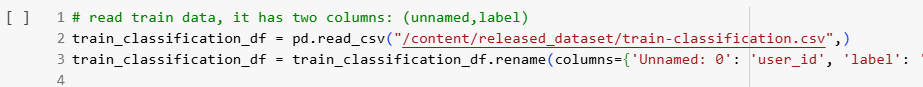
## Code Explanation:

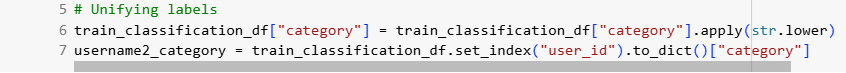
1. Data Loading:  
This code mounts Google Drive to access the dataset stored in the user's Google Drive.  
metin, ekran görüntüsü, yazı tipi, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

2. Data Import and Preprocessing:  
Necessary libraries like numpy (for array operations), pandas (for data manipulation), gzip (to handle compressed files), and json (to parse JSON data) are imported here.  
metin, ekran görüntüsü, yazı tipi, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

3. Loading Training Data:  
This code loads the training dataset for classification, renames the columns for clarity, and makes the user ID and category more readable.  


4. Handling Categories:  
Converts all category labels to lowercase for consistency and creates a dictionary mapping each user ID to its respective category.  


5. Model Training (SVM) and Validating :  
SVM with a linear kernel is used for training. The dataset is split into training and testing sets, and the model is trained using the training data. Best Model which is used in the code is determined by Cross-Validation  
Train:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Validation:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Round 2: Ensemble Learning

## Model Used:

Ensemble Learning (such as Random Forest, Logistic Regression, Naïve Bayes, Multi Layer perceptron)

**SVM (Support Vector Machine):**  
A linear kernel was used in Round 1 to classify Instagram content. The linear kernel is well-suited for linearly separable datasets, providing simplicity and computational efficiency. It projects the data into its original space without introducing additional dimensions, which reduces the risk of overfitting when the dataset size is limited.

The **regularization parameter (C)** was set to its default value, striking a balance between minimizing training error and maintaining the model's ability to generalize well to unseen data.

**Random Forest:**  
The **n\_estimators** parameter was set to 100 decision trees, creating a robust ensemble model that enhances predictive power by averaging the results of individual trees while maintaining computational efficiency.

The **max\_features** parameter was set to the square root of the total features. This choice promotes diversity among the decision trees, helping to reduce overfitting and improve overall model performance.

The **max\_depth** was left to expand automatically until all leaves were pure or contained fewer than the minimum samples required for a split. This dynamic depth control prevents the trees from becoming overly complex, which could otherwise lead to overfitting.

To ensure reproducibility, the **random\_state** parameter was set to 42. This allows the model to produce consistent splits and results, making debugging and validation more reliable.

**Logistic Regression:**  
The **solver** used was 'liblinear,' a method ideal for smaller datasets and binary classification tasks. It employs a coordinate descent algorithm, ensuring efficient optimization for linear models.

An **L2 penalty** was applied to the model to reduce overfitting by penalizing large coefficient values, encouraging simpler and more generalizable models.

The **C parameter** (inverse of regularization strength) was set to 1.0, balancing the trade-off between strong regularization and accurate model fitting. Lower values enforce stronger regularization, while higher values focus on fitting the training data more closely.

**Naïve Bayes:**  
The **Gaussian Naïve Bayes** model was chosen, assuming the features are normally distributed—a practical assumption for continuous data.

This model was selected for its computational efficiency and ability to handle high-dimensional datasets, even with limited training data. While it assumes independence between features (which may not always hold), this simplification enables quick deployment and serves as a strong baseline model.

**Multi-Layer Perceptron (MLP):**  
A **single hidden layer with 100 neurons** was employed to capture non-linear relationships in the data while keeping the model computationally manageable.

The **ReLU activation function** was chosen for its ability to introduce non-linearity and avoid the vanishing gradient problem, which is commonly observed with sigmoid or tanh activations.

The model was trained using the **Adam optimizer**, which combines adaptive learning rates with momentum for efficient and fast convergence.

The **max\_iterations** parameter was set to 200, ensuring the model had enough iterations for convergence without risking overfitting.

To maintain consistency and reproducibility, the **random\_state** parameter was set to 42.

## Results:

Classification Accuracy: 84.14% (Rank 43/138)  
F1-Weighted Score: 83.95% (Rank 43/138)  
Regression Error: 727.47 (Rank 27/138)

## Improvements Made:

This round focused on improving the classification accuracy. By using ensemble learning techniques, we achieved a significant increase in accuracy, reaching 84.14%. The F1-weighted score also improved, indicating a better balance between precision and recall. However, the regression error did not show significant improvement compared to Round 1.

## Code Explanation:

1. Ensemble Model Setup:  
In Round 2, we shifted to ensemble learning using the Random Forest classifier, which is well-known for handling overfitting better than a single model. The model is trained with 100 trees.  
metin, ekran görüntüsü, yazı tipi, doküman, belge içeren bir resim

Açıklama otomatik olarak oluşturuldu

2. Comparison with Round 1 (SVM):  
This round was focused on improving classification accuracy. By comparing the results of the Random Forest classifier with those from Round 1 (SVM), we observe that accuracy increased, but regression error did not improve much.

# Round 3: Exploring Other Models

## Models Tested:

Various models (other than SVM)

## Results:

The results showed that SVM remained the most effective model for this task.

## Conclusion:

Despite experimenting with different models in Round 3, SVM was found to yield the best results in terms of classification accuracy and overall performance. This confirmed that SVM was the optimal choice for this specific problem.

1. Evaluation and Comparison:  
Like in previous rounds, the model is evaluated on the test set, and a classification report is produced. This model provided slightly better results than Random Forest in some aspects but did not significantly outperform SVM.

2. Final Decision to Choose SVM:  
After testing various models, it became clear that SVM still outperformed the others in classification tasks with this specific dataset. SVM’s ability to handle high-dimensional feature spaces made it a better fit for the problem.

# SMOTE (Synthetic Minority Over-sampling Technique):

## Purpose:

SMOTE was used in this project to address class imbalance in the dataset. Given that the dataset had an unequal distribution of labels, SMOTE helped by generating synthetic examples for the underrepresented classes, improving the model’s ability to learn from minority classes and make better predictions.

# Model Selection and Final Decision:

After experimenting with SVM and ensemble learning models, the final decision to stick with SVM was based on its strong performance, especially in terms of classification accuracy. While ensemble methods improved classification metrics, they did not offer a substantial advantage in reducing the regression error, making SVM the best choice.