**Introduction**

In this project assignment, a system was developed to analyze texts written by different authors and predict the author of an unknown text. For this purpose, various feature extraction methods were applied, including TF-IDF-based word features, word-based 2-grams and 3-grams, and character-based 2-grams and 3-grams. Additionally, text representation was obtained using the deep learning-based BERT model. Using these representations, classification was performed with various machine learning algorithms such as Random Forest, Support Vector Machine (SVM), XGBoost, Naïve Bayes, Multi-Layer Perceptron (MLP), and Decision Tree.

**Materials and Methods**

**Development Environment**

This project aims to develop a text classification system capable of predicting the author of a given text based on samples from known authors. The project includes steps such as data analysis, feature extraction, model development, and evaluation of results. The development process was carried out in the Google Colab environment, a cloud-based Jupyter Notebook platform that enables interactive work in data science and machine learning projects. The Python programming language was used throughout the project. For text representation, TF-IDF, word and character-based n-grams (2-grams and 3-grams), and the BERT model were applied. For classification, various models were trained and compared using algorithms such as Random Forest, Support Vector Machine (SVM), XGBoost, Naïve Bayes, Multi-Layer Perceptron (MLP), and Decision Tree.

**Datasheets**

The dataset used in the scope of the project was accessed through the integration of Google Drive with the Google Colab environment. The dataset consists of *x* authors, each having a varying number of texts. These texts were analyzed in a way that reflects the linguistic characteristics of the authors and were used to train and test machine learning algorithms.

**Text Representation Methods:**

In this project, various text representation methods were used to make the texts suitable for processing by machine learning algorithms. Four different methods used are detailed below:

**3.1. TF-IDF (Term Frequency-Inverse Document Frequency)**

The TF-IDF method is a classical text mining technique used to determine the importance of a word across documents. In this project, each word was considered as a term, and a weighted feature vector was created by taking into account the frequency of the words in each document as well as their prevalence across all documents. This approach reduces the influence of frequently used but non-distinctive words and highlights more meaningful and distinctive terms.

**3.2. Word-Based N-Grams (2-gram and 3-gram)**

In addition to the TF-IDF method, word-based 2-grams and 3-grams were also used as separate feature sets. In this approach, words were considered not only individually but also in consecutive groups (e.g., “big city”, “living in big cities”). This method allowed for better modeling of authors' expression styles, word combinations, and linguistic patterns.

**3.3. Character-Based N-Grams (2-gram and 3-gram)**

Character-level 2-gram and 3-gram representations were also used to capture linguistic features at a more granular scale beyond the word level. This method has been useful in analyzing differences based on writing style, the use of affixes, and character sequences. Character-level n-grams, in particular, have provided significant contributions in analyzing short texts or datasets with fewer words.

**3.4. BERT (Bidirectional Encoder Representations from Transformers)**

Finally, a deep learning-based text representation method, the BERT model, was used. BERT processes texts bidirectionally, taking context into account, and evaluates the meaning of each word in relation to the surrounding words. This allows for a deeper, more semantic, and contextual representation. By using the pre-trained version of the model, each text was transformed into a high-dimensional vector, and these vectors were used as input during the classification phase.

**Machine Learning Algorithms**

The project was trained using Decision Tree, Support Vector Machine, Random Forest, Naive Bayes, Multi-Layer Perceptron, and XGBoost algorithms.

**Decision Tree**

A decision tree is a flowchart-like diagram that maps out all potential solutions to a specific problem. It is commonly used by organizations to help determine the most appropriate course of action by comparing all possible outcomes of a series of decisions.

**Support Vector Machine**

Support Vector Machine aims to find the hyperplane that best separates the data, ensuring the maximum margin between classes. Its effectiveness on high-dimensional datasets and its ability to be extended to non-linear classifications through kernel methods have made SVM a widely preferred method in various fields.

**Random Forest**

Random Forest is an ensemble learning method based on decision trees and is effectively used for both classification and regression problems. This algorithm works by training a large number of decision trees on randomly selected subsets of the data and combines the results of each tree to produce the final prediction. It is frequently preferred in data mining and machine learning applications due to its ability to reduce overfitting risk, provide high accuracy, and measure variable importance.

**Naive Bayes**

Despite its simple structure, this method can produce fast and effective results on large datasets and is widely used in areas such as text classification, spam filtering, and sentiment analysis. The model is referred to as "naive" because it is based on the assumption that all features are independent of each other.

**Multi-Layer Perceptron**

Multi-Layer Perceptron is a feedforward structure belonging to the family of artificial neural networks, consisting of an input layer, one or more hidden layers, and an output layer. Each neuron receives inputs from the neurons in the previous layer, combines them with assigned weights, and processes them through an activation function. Thanks to its ability to model non-linear relationships, MLP is successfully used in various areas such as classification, regression, and pattern recognition.

**XGBoost**

XGBoost (Extreme Gradient Boosting) is a high-performance and optimized ensemble learning algorithm based on the gradient boosting method. Due to its speed and efficiency, ability to handle missing data, and use of regularization techniques to reduce overfitting, it is frequently preferred in large datasets and competitive environments.

**Evaluation Metrics**

Various metrics are used to evaluate the performance of machine learning models. These metrics measure whether the model makes correct classifications, as well as assess classification errors and the ratio of accurate predictions. In text classification tasks such as authorship attribution, these metrics are important for assessing the accuracy and reliability of the models. Especially when it comes to distinguishing subtle differences between authors, the ability of algorithms to generalize correctly becomes a critical factor. Therefore, evaluation using appropriate metrics is essential for measuring model success and identifying areas for improvement.

**Accuracy**

It represents the ratio of all correct predictions to the total number of predictions made by the model. Considering all classes, it indicates the proportion of correctly classified instances.

**Precision**

It indicates the ratio of true positive predictions to the total number of positive predictions made by the model. In other words, it measures how many of the instances predicted as positive by the model are actually positive.

**Recall**

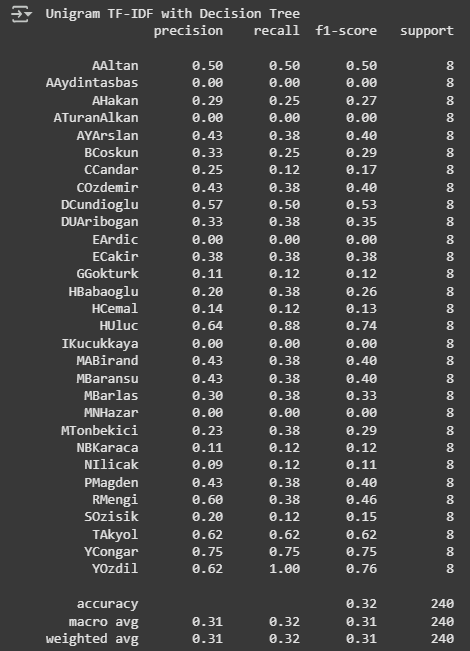
It represents the ratio of true positive predictions to the total number of actual positive instances. This metric shows how well the model identifies the positive class.

**F1 Score**

It is the harmonic mean of precision and recall and is a more meaningful performance metric, especially in imbalanced datasets. It evaluates the overall model performance by considering both false positive and false negative predictions.

**TF-IDF**

**TF-IDF and Decision Tree**

****

**TF-IDF + Multi-Layer Perceptron**

**metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**TF-IDF + Random Forest**

**metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**TF-IDF + Naïve Bayes**

**metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**Word Based 2-gram and 3-gram**

**Word Based 2-gram and 3-gram XGBoost**

**metin, ekran görüntüsü, menü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.** **metin, ekran görüntüsü, menü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**Word Based 2-gram and 3-gram Random Forest**

**metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.** **metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**Character Based 2-gram and 3 gram**

**Character-based 2-gram and 3-gram and Random Forest**

**metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.** **metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**Bert**

**BERT + Support Vector Machine**

**metin, menü, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.**

**Result**

Bu çalışmada, farklı metin temsil yöntemleri ve çeşitli makine öğrenmesi algoritmaları kullanılarak yazar sınıflandırması gerçekleştirilmiştir. Modeller, %80 eğitim - %20 test bölünmesiyle test edilmiş ve Accuracy, Precision, Recall ve F1-Score metrikleriyle değerlendirilmiştir.

**Genel sonuçlara göre:**

* **En başarılı kombinasyon**, TF-IDF + MLP ile elde edilmiştir (Accuracy: **0.84**, F1-score: **0.81**).
* TF-IDF + Naive Bayes ve TF-IDF + Random Forest modelleri de oldukça güçlü sonuçlar üretmiştir (F1-skorları sırasıyla **0.74** ve **0.72**).
* **BERT + SVM** kombinasyonu beklendiği gibi yüksek başarı sağlayamamış, Accuracy değeri sadece **0.52**'de kalmıştır. Bu durum, BERT temsillerinin daha derin ayarlama (fine-tuning) veya daha güçlü modellerle desteklenmesi gerektiğini göstermektedir.
* **XGBoost** ile yapılan Word Based 2-gram and 3-gram analizlerinde performans oldukça düşüktür (F1-score ~0.17–0.36), bu da XGBoost'un bu tür n-gram temsilinde yeterince etkili olmadığını göstermektedir.
* Karakter n-gram temsillerinde özellikle Character 3-gram + RF kombinasyonu diğerlerine göre daha iyi sonuç vermiştir (F1-score: **0.53**).

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

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

<https://github.com/nuhyalcin97/data_mining_proje.git>