KEY RESULTS

Following metrics are of prime importance when evaluating the effectiveness of the classifier.

- Accuracy will measure the overall correctness of the model by calculating the ratio of correctly predicted instances to the total number of instances. This metric provides a broad view of the model's performance but may not fully capture its effectiveness, particularly in cases where class imbalances exist.
- Precision will evaluate the quality of the model's positive predictions by calculating the ratio of true positive predictions to the total number of positive predictions (both true and false positives). High precision indicates that the model has a low false positive rate, meaning it is adept at predicting instances of a class without mistakenly labeling irrelevant instances as part of that class.
- Recall (or sensitivity) will measure the model's ability to identify all relevant instances within a specific class by calculating the ratio of true positive predictions to the total number of actual positive instances (both true positives and false negatives). High recall indicates that the model successfully captures most of the relevant instances, though it may come at the expense of increasing false positives.
- F1-Score provides a harmonic mean of precision and recall, balancing the two metrics to give a single measure of performance that accounts for both false positives and false negatives. The F1-score is particularly useful when the data has class imbalances, as it provides a more nuanced view of the model's effectiveness by considering both precision and recall simultaneously.

By employing these metrics, we can comprehensively evaluate the performance of our models. This multifaceted evaluation will ensure that our model not only performs well overall but also excels in distinguishing AI-generated text from human-written content across various contexts.

Table 1 and 2 provide values of the metrics for SVM, Multinomial Naive Bayesian, Linear Regression, Random forest, XGBoost and Neural Network classifiers in two different preprocessing settings with CountVectorization and TF-IDF respectively.

Classifier	Accuracy	Precision	F1-Score	Recall
SVM	54.4%	40.2%	47.7%	58.5%
MultinomialNB	63.7%	49.3%	60.4%	78.2%
Linear Regression	98.3%	97.3%	97.7%	98.0%
Random Forest	98.3%	98.6%	96.6%	97.6%
XGBoost	98.9%	99.0%	99.0%	99.0%
NN	99.1%	99.0%	99.0%	99.0%

Table 1: Results of Classifiers with CountVectorization

Classifier	Accuracy	Precision	F1-Score	Recall
SVM	55.3%	41.5%	50.2%	63.6%
MultinomialNB	66.7%	52.2%	61.1%	73.6%
Linear Regression	98.5%	97.6%	97.9%	98.1%

Random Forest	98.2%	98.1%	96.8%	97.5%
XGBoost	83.3%	84.0%	83.0%	87.0%
NN	98.9%	99.0%	99.0%	99.0%

Table 2: Results of Classifiers with TF-IDF

After preprocessing with either TF-IDF or Count vectorization, with a maximum of 1000 features, the majority of the classifiers show high accuracy rates. The SVM stands out, nonetheless, because of its substantially poorer accuracy. The structure of the TF-IDF representation and the sensitivity of an SVM to the selected hyperparameters may have contributed to this disparity, as they may not have been ideal for the distribution of the data in feature space. Even with this SVM and TF-IDF anomaly, the overall performance trends are in line with expectations, demonstrating the effectiveness of the used techniques in text classification tasks.

It should come as no surprise that the neural network model performed better in our text categorization tests than other conventional classifiers. The higher performance of neural networks can be ascribed to multiple principal benefits.

- Because they can discover intricate patterns and connections in the data, neural networks are particularly good at processing high-dimensional data, such as those generated by text vectorization techniques
- Neural networks are superior to most conventional algorithms in modeling non-linear relationships because of their layered design and non-linear activation functions.

EVALUATION

Model Performance Analysis

We implemented several classification algorithms to differentiate between AI-generated and human-written text, employing CountVectorizer and TF-IDF for preprocessing. The performance metrics used were accuracy, precision, F1-score, and recall.

Support Vector Machine (SVM): The SVM with an RBF kernel and C parameter set to 0.15 showed suboptimal performance, particularly with TF-IDF, resulting in accuracy rates of 54.4% (CountVectorizer) and 55.3% (TF-IDF). The precision and F1-score were also low, indicating a high rate of false positives.

Multinomial Naive Bayes (NB): This model performed better than SVM, with accuracies of 63.7% (CountVectorizer) and 66.7% (TF-IDF). The high recall indicates it captured most of the relevant instances but at the cost of precision.

Logistic Regression: Demonstrated strong performance with accuracies of 98.3% (CountVectorizer) and 98.5% (TF-IDF). High precision and recall values indicate robust and reliable predictions.

Random Forest: Achieved high accuracy rates of 98.3% (CountVectorizer) and 98.2% (TF-IDF), with excellent precision and recall, demonstrating its effectiveness in text classification tasks.

XGBoost: Showed superior performance, especially with TF-IDF (83.3% accuracy), and achieved near-perfect precision, recall, and F1-scores with CountVectorizer (98.9% accuracy).

Neural Network: The neural network model outperformed all other classifiers, with accuracy rates of 99.1% (CountVectorizer) and 98.9% (TF-IDF). Its superior ability to recognize complex patterns is evident from its high precision, recall, and F1-score.

Comparison of Preprocessing Techniques

CountVectorizer: Generally resulted in slightly higher accuracy and performance metrics for most models compared to TF-IDF. This suggests that a simpler bag-of-words representation may be sufficient for this classification task.

TF-IDF: Although it typically aims to reduce the impact of common but less informative words, TF-IDF led to comparable, though slightly lower, performance metrics. The SVM's significantly lower performance with TF-IDF suggests that it may not be as well-suited for this preprocessing method.

Strengths and Weaknesses

Support Vector Machine (SVM): Strength lies in its effectiveness in high-dimensional spaces. However, it struggled with the specific feature distributions of TF-IDF, indicating sensitivity to hyperparameters and preprocessing methods.

Multinomial Naive Bayes (NB): Simple and efficient, performing well with CountVectorizer. Its probabilistic nature allows for straightforward interpretation but sacrifices precision for higher recall.

Logistic Regression: Balances simplicity and performance, making it a reliable choice for text classification with strong overall metrics.

Random Forest: Robust and handles high-dimensional data effectively. Its ensemble nature leads to high accuracy but requires careful tuning to avoid overfitting.

XGBoost: Excels in handling large-scale tasks with high efficiency and accuracy. However, its complexity and requirement for extensive hyperparameter tuning can be challenging.

Neural Network: Outperforms traditional methods due to its ability to model non-linear relationships and complex patterns. The downside is the higher computational cost and risk of overfitting without proper regularization techniques.

Unexpected Results or Anomalies

The SVM's poor performance with TF-IDF preprocessing was unexpected, highlighting its sensitivity to the choice of hyperparameters and preprocessing methods. This anomaly underscores the importance of thorough hyperparameter tuning and model selection tailored to the data distribution.