National University of Singapore

Comparison of Matthews Correlation Coefficient (MCC) to Accuracy, F1, Precision and Recall for Imbalanced Data

DSA4199 Final Year Project

Submitted by:

Michelle Yong Pei Feng
A0221827H

Supervisor:

Dr Chia Hui Teng

Date:

6th March 2025

Table of Content

1. Introduction

Evaluating machine learning models is a critical task, particularly in domains where imbalanced datasets are common. While accuracy, precision, and F1-score are widely used, these measures may offer partial or even misleading views of the model performance in scenarios with class imbalance (Chicco & Jurman, 2020). This creates a gap in the industry, where models may appear highly accurate yet perform poorly on minority classes.

The Matthews Correlation Coefficient (MCC) has emerged as a potential standard metric that effectively summarises a classifier's performance into a single value such as accuracy, F1-score, etc. However, unlike these other standard metrics, MCC accounts for all components of the confusion matrix, making it particularly valuable for imbalanced datasets (Chicco & Jurman, 2020). This paper explores the origins, significance, and applications of MCC in the context of class imbalance and machine learning.

Most machine learning algorithm analysis focuses on the model's performance rather than that of the evaluation metrics. This paper shifts the focus from model performance to the effectiveness of evaluation metrics in assessing models, assuming that the model itself is not the issue. It explores MCC as a robust metric that can enhance the standard model evaluation process by addressing gaps left by existing metrics like accuracy, F1-score, etc.

To achieve this, the study introduces two quantitative imbalance metrics: the Imbalance ratio (IR) and Shannon entropy. IR quantifies the degree of skewness in class distributions, while Shannon Entropy measures the uncertainty in class labels, providing a complementary perspective on dataset imbalance. Through a systematic examination of classification metrics in relation to these imbalance measures, this research highlights the strengths and limitations of MCC and its potential role in improving classification performance evaluation in imbalanced datasets.

Both synthetic and real-world datasets are used. The first phase of the study involves generating artificial datasets with controlled levels of class imbalance to analyse how different evaluation metrics respond to varying imbalance ratios. The second phase applies the same analysis to a real-world dataset, specifically the Credit Card Fraud Detection dataset, which exhibits extreme class imbalance. By comparing these two approaches, this research provides insights into the generalizability of MCC and its reliability as an evaluation metric across different data distributions.

By shifting the focus from model performance to metric effectiveness, this study aims to bridge the gap in machine learning evaluation practices. The findings contribute to a better understanding of how MCC compares to conventional metrics, reinforcing the need for more comprehensive evaluation methods when dealing with highly imbalanced classification problems.

2. What is MCC?

2.1 Phi coefficient and MCC

In statistics, the Phi coefficient is a special case of Pearson Product Moment Correlation that gives a linear association between variables that are naturally dichotomous with the underlying discrete distribution (Islam & Rizwan, 2022). It is derived from a frequency distribution similar to that in Table 1.

Table 1: A frequency table with two variables, X and Y, with levels 0 and 1.

		Variable Y		
		Y = 0	Y = 1	
Variable X	X = 0	a	b	(a+b)
	X = 1	c	d	(c+d)
		(a+c)	(b+d)	

The formula for the Phi coefficient is:

$$\phi = \frac{bc - ad}{\sqrt{(a+b)(c+d)(a+c)(b+d)}}$$

The frequency distribution table shares similar characteristics with the confusion matrix in a binary class classification, where true labels are tabulated against predicted labels.

Table 2: An example of a confusion matrix.

		Predicted Class		
		Positive	Negative	
Actual Class	Positive	TP	FN	(TP + FN)
	Negative	FP	TN	(TN + FP)
		(TP + FP)	(TN + FN)	

The formula for MCC is:

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Like Phi, MCC accounts for all elements of the confusion matrix, ensuring that it reflects positive and negative class performance. By embedding the Phi coefficient into a confusion matrix framework, MCC bridges statistical theory and practical machine learning evaluation. Its value ranges from 1 for perfect positive correlation to -1 for perfect negative correlation, with 0 indicating none (Chicco & Jurman, 2021).

2.2 Applications of MCC

MCC was developed by British biochemist Brian W. Matthews in 1975 and found initial applications in bioinformatics. It helps researchers predict protein structures and gene occurrences where positive cases are often rare (Matthews, 1975).

MCC was adopted and incorporated across various fields and programs due to its ability to handle class imbalances effectively. The reviewed papers are summarised in Table 3, and the following highlights a few notable domains and applications:

1. <u>Bioinformatics and Genomics</u>

   MCC is widely used to evaluate the performance of binary classifiers in tasks such as gene expression classification, protein structure prediction, and disease susceptibility prediction. An example is classifying healthy vs. disease samples in genomic datasets with imbalanced distributions (Grandini et al., 2020).

2. <u>Medical Diagnostics</u>

   MCC is crucial in assessing diagnostic models where false negatives can have severe implications, such as cancer detection, rare disease identification, and medical imaging analysis. One example is evaluating models for identifying tumours in highly imbalanced datasets (Chicco & Jurman, 2020).

3. <u>Fraud Detection</u>

   Used extensively in the finance and e-commerce sectors to detect fraudulent transactions, which are often rare compared to legitimate transactions. A common example is credit card fraud detection systems (Machine Learning Group - ULB, 2016).

Table 3: A summary of studies on Matthews Correlation Coefficient

#	Authors	Origin	Title	Summary	Major Themes
1	Boughorbel, S., Jarray, F., & El-Anbari, M.	USA	Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric	Proposes an optimal Bayes classifier for MCC metric, addressing classification challenges in imbalanced datasets.	MCC Optimisation, Imbalanced Data, Bayesian Classifier
2	Cai, Z., Fan, L., Ma, J., Wu, W.	China	Matthews Correlation Coefficient Loss for Deep Convolutional Neural Networks	Proposes a novel loss function based on MCC for training deep convolutional neural networks, aiming to enhance performance in scenarios with class imbalance.	MCC Loss Function, Deep Learning, CNNs, Class Imbalance
3	Chicco, D., & Jurman, G.	Italy	The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation	Highlights the superiority of MCC over F1-score and accuracy in binary classification evaluation.	Evaluation Metrics, MCC, Classification
4	Chicco, D., Warrens, M. J., & Jurman, G.	Canada, Netherlands, Italy	The Matthews Correlation Coefficient (MCC) is More Informative Than Cohen's Kappa and Brier Score in Binary Classification Assessment	This study compares MCC with Cohen's Kappa and Brier Score, demonstrating that MCC provides a more truthful and informative evaluation of binary classifications, especially in imbalanced datasets.	Binary Classification, MCC, Cohen's Kappa, Brier Score, Imbalanced Datasets
5	Foody, G. M.	UK	Challenges in the real-world use of classification accuracy metrics: From recall and precision to the Matthews correlation coefficient	Discusses the challenges associated with various classification accuracy metrics, including recall, precision, and MCC, highlighting how factors like prevalence and imperfect reference standards can influence their reliability.	Classification Accuracy, MCC, Recall, Precision, Prevalence, Reference Standards
6	Grandini, M., Bagli, E., & Visani, G.	Italy	Metrics for Multi-Class Classification: an Overview	Analyses classification metrics for multi-class problems, including MCC's effectiveness in imbalanced settings.	Multi-Class Classification, MCC, Imbalanced Data
7	Islam, T. U., & Rizwan, M.	Pakistan	Comparison of correlation measures for nominal data	Explore the best measure of association for nominal data. Monte Carlos simulations reveal that Phi and Pearson correlation perform equally well for naturally dichotomous variables.	Correlation Coefficient, Pearson and Phi Correlation

8	Matthews, B. W.	USA	Comparison of the predicted and observed secondary structure of T4 phage lysozyme	Discusses the accuracy of secondary structure predictions for T4 phage lysozyme and introduces the Matthews correlation coefficient (MCC).	Protein Structure Prediction, MCC
9	Zhu, Q.	China	On the performance of Matthews correlation coefficient (MCC) for imbalanced dataset	Examines the performance of MCC in imbalanced datasets, providing insights into its reliability and potential limitations when class distributions are skewed.	MCC Performance, Imbalanced Datasets, Classification Metrics

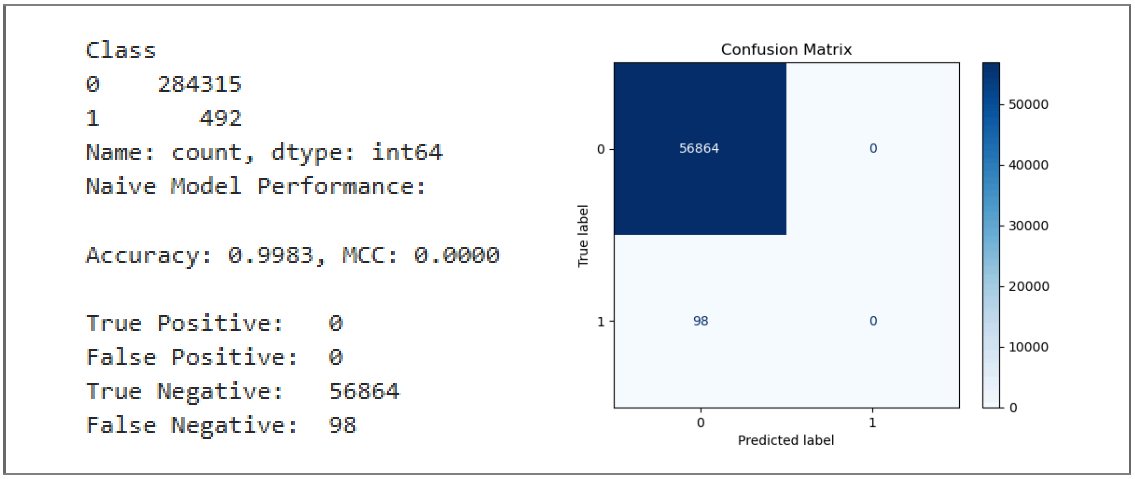
2.3 Advantages of MCC

One of MCC's greatest strengths is that it considers all four components of the confusion matrix: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Unlike metrics such as accuracy, precision, recall, or F1-score, which only focus on selected elements, MCC evaluates the classifier's performance holistically. This ensures a more balanced assessment, especially in cases where the dataset is skewed (Chicco & Jurman, 2020).

For instance, in a binary classification scenario where one class dominates the dataset, accuracy may suggest that the classifier performs well when predicting the majority class. MCC avoids this issue by considering positive and negative predictions.

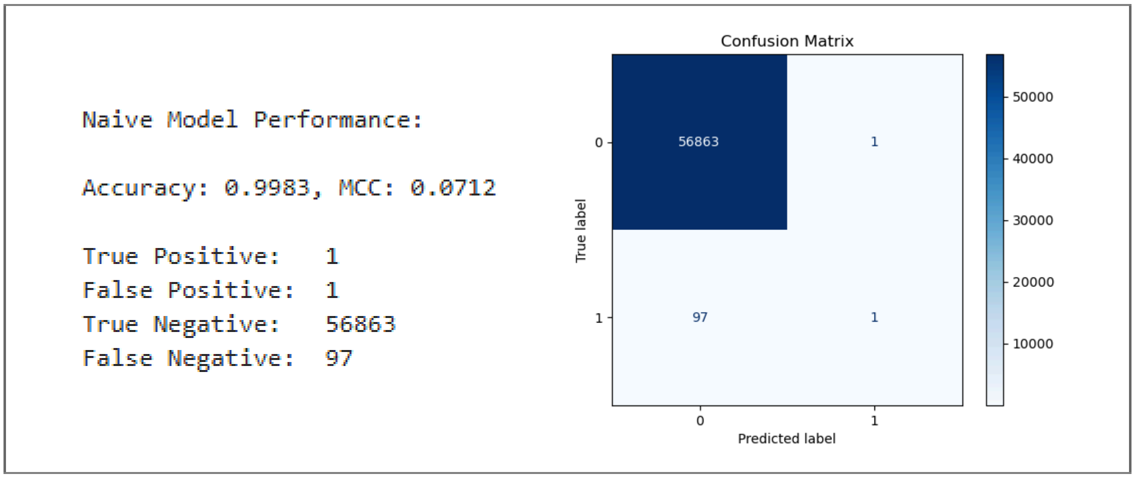
In a fraud detection dataset, if 99% of transactions are non-fraudulent and a classifier predicts all transactions as non-fraudulent, the accuracy would still be 99%, even though the model has no predictive value. MCC, however, would correctly assign this classifier an MCC score close to 0, reflecting its poor performance. This makes MCC one of the most reliable and fair metrics for imbalanced classification problems.

Image 1: Accuracy and MCC scores of an imbalanced credit card dataset (zero entries).



```
Class
0    284315
1       492
Name: count, dtype: int64
Naive Model Performance:

Accuracy: 0.9983, MCC: 0.0000

True Positive:   0
False Positive:  0
True Negative:   56864
False Negative:  98
```

Image 2: Accuracy and MCC scores of an imbalanced credit card dataset (non-zero entries).



```
Naive Model Performance:

Accuracy: 0.9983, MCC: 0.0712

True Positive:    1
False Positive:   1
True Negative:    56863
False Negative:   97
```

Image 1 shows the accuracy and MCC scores of an imbalanced credit card dataset with 284315 instances of non-fraudulent instances (99.8%) and 492 instances of fraudulent cases (0.2%). A naïve model is used to test the scores of both metrics. Since the dataset is highly imbalanced, the naïve model takes the most frequent class (non-fraud) and always predicts it. The naïve model fails to capture the minority class, resulting in an MCC score of 0, which correctly reflects that the no model has no predictive value. However, accuracy is misleadingly high, with a score of 99.8%. This simple example illustrates that accuracy is misleading when a dataset is heavily skewed, whereas MCC detects class imbalance and provides a more indicative score.

With reference to section 2.4, the TP and FP values for the naïve model in image 2 are hardcoded to be 1 instead of 0. By removing the zero column entries, the MCC score increased slightly, indicating that the model considers the minority class to a very small extent. However, MCC continues to show that the naïve model fails at predicting the minority class. Accuracy, on the other hand, remains deceivingly high.

Another key advantage of MCC is its symmetry. Many classification metrics, such as precision and recall, can be biased toward one class. MCC, on the other hand, treats both the positive and negative classes equally, ensuring that neither class dominates the evaluation. This is particularly useful in medical and biological applications, where false negatives (missed diagnoses) and false positives (misdiagnoses) carry different consequences (Boughorbel et al., 2017). Unlike F1-score, which ignores true negatives, MCC provides a balanced view of classification performance regardless of the dataset's distribution.

Traditional classification metrics such as accuracy and F1-score have significant limitations in imbalanced datasets. Accuracy can be misleading if one class dominates the dataset, and the F1-score does not account for true negatives, making it an incomplete measure.

In contrast, MCC captures all aspects of classification performance. Studies have demonstrated that MCC outperforms both F1-score and accuracy, especially when dealing with imbalanced classification tasks (Chicco & Jurman, 2020). This makes MCC a preferred metric in research and real-world applications.

2.4 Limitations of MCC

MCC is most commonly used for binary classification problems. It can be extended to multi-class problems using a generalised MCC formula, but this is computationally more complex and less intuitive than traditional metrics like accuracy or F1-score (Grandini et al., 2020). For multi-class classification, MCC requires computing a correlation matrix across all confusion matrix elements, making its interpretation less straightforward compared to precision, recall, or accuracy. The matrix computation also involves multiple summations and cross-products, making it more difficult to compute efficiently in large datasets. For multi-class classification, alternatives like the Macro-Averaged F1-score or the Balanced Accuracy metric may be easier to compute (Ferrer, 2022).

Another well-known limitation of the MCC is that it becomes undefined when a row or column of the confusion matrix contains only zeros. This situation arises when a classifier predicts only one class, meaning:

- It never predicts the positive class (TP = FP = 0) → Entire first row of the matrix is zero.
- It never predicts the negative class (TN = FN = 0) → Entire second row of the matrix is zero.

This commonly happens with trivial majority classifiers, which always predict the majority class in an imbalanced dataset. When this occurs, the denominator in the MCC formula contains a zero product, making the calculation mathematically undefined (Chicco & Jurman, 2020).

Table 4: Scenarios where MCC becomes undefined.

Scenario	TP	FP	TN	FN	MCC
Predicts only ‘Non-fraud’ cases	0	0	990	10	Undefined
Predicts only ‘Fraud’ cases	10	990	0	0	Undefined
Balanced predictions	50	50	50	50	Defined

In the first two cases, the classifier only predicts one class, leading to a confusion matrix row or column filled with zeros. As a result, MCC is undefined. However, in the last case, where both classes are predicted, MCC remains calculable and meaningful.

Addressing this limitation is crucial when working with imbalanced datasets, as trivial classifiers can give misleading performance results. Researchers should consider alternative performance metrics or apply perturbation techniques when necessary. To handle this, a workaround involves approximating MCC for these edge cases using a small perturbation value ε. They show that when a nearly trivial classifier is considered, MCC approaches 0, aligning with the expected value for a random coin-tossing classifier (Chicco & Jurman, 2020).

By substituting the zero entries with an arbitrarily small value ε, we ensure all positions of the MCC are defined. The formula with this small perturbation value ε is:

$$MCC = \frac{(TP+\varepsilon)(TN+\varepsilon)-(FP+\varepsilon)(FN+\varepsilon)}{\sqrt{(TP+FP+\varepsilon)(TP+FN+\varepsilon)(TN+FP+\varepsilon)(TN+FN+\varepsilon)}}$$

2.5 Relation to Other Metrics

MCC is often compared with other binary classification metrics, such as:

- Precision and Recall: Focus on the positive class. Precision forgoes TN and FN, while recall forgoes TN and FP. Both metrics do not consider TN in their formulae, which may be ineffective if a model is classifying a positively imbalanced dataset with minority negative cases.

$$Precision = \frac{TP}{TP+FP} \qquad Recall = \frac{TP}{TP+FN}$$

- Accuracy: Despite taking into account all 4 components of the confusion matrix, it can be misleading in the case of imbalanced datasets, as shown in images 1 and 2. The model can simply achieve high accuracy by predicting the majority class most of the time.

$$Accuracy = \frac{TP+TN}{TP+FN+TN+FP}$$

- F1-Score: Balances precision and recall but does not account for TN just like precision and recall. F1-Score would be ineffective in evaluating a model classifying a positively imbalanced dataset with minority negative cases.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

MCC combines the strengths of these metrics while mitigating their weaknesses, making it a preferred choice in many scenarios (Chicco & Jurman, 2020).

Table 5: Comparison of performance metrics based on confusion matrix elements.

Metrics	TP	TN	FP	FN
Accuracy	✔	✔	✔	✔
F1-score	✔		✔	✔
Precision	✔		✔	
Recall	✔			✔
MCC	✔	✔	✔	✔

To illustrate the effectiveness of MCC, consider a clinical example of a positively imbalanced dataset with 9 healthy patients and 91 sick patients (Chicco & Jurman, 2020). Suppose we have the following confusion matrix:

Table 6: Example inspired by Chicco & Jurman (2020) case A1.

Class	Sick	Healthy
Sick	TP = 90	TN = 0
Healthy	FP = 9	FN = 1

Accuracy = $\frac{90 + 0}{90 + 0 + 9 + 1}$ = 0.900

F1 = $\frac{2 \times 90}{2 \times 90 + 9 + 1}$ = 0.947

Precision = $\frac{90}{90 + 9}$ = 0.909

Recall = $\frac{90}{90 + 1}$ = 0.989

MCC = $\frac{(90 \times 0) - (9 \times 1)}{(90 + 9)(90 + 1)(0 + 9)(0 + 1)}$ = -0.000111

While accuracy (0.900) and F1-score (0.947) are misleadingly high, MCC reveals a more balanced evaluation of the classifier's performance (≈0). In fact, as MCC gives a negative score, it indicates that the model fails to classify the cases correctly. Since the model's accuracy is lower than the NIR[1] (0.91), this indicates that the classifier performs worse than a naïve model that always predicts "Sick". This further supports the conclusion that the classifier is overfitting to the majority class while failing to detect healthy individuals. MCC remains a more informative metric for assessing classifier performance in imbalanced datasets, as it accounts for both correct and incorrect classifications in all categories.

[1] The No Information Rate (NIR) represents the accuracy a naïve classifier would achieve by always predicting the majority class.

3. What is Class Imbalance?

3.1 Definition

Class imbalance occurs when the distribution of classes in a dataset is significantly skewed. For instance, in fraud detection datasets, fraudulent transactions may account for less than 1% of all transactions, leading to an extreme imbalance between the positive and negative classes. Similarly, in medical diagnostics, cases of rare diseases may represent only a small fraction of the total dataset.

3.2 Why is Class Imbalance Important When Discussing MCC?

Class imbalance plays a crucial role in evaluating the effectiveness of MCC because many traditional classification metrics fail in imbalanced datasets (Chicco & Jurman, 2020). When one class is significantly overrepresented compared to another, common performance measures such as accuracy, precision, recall, and F1-score often provide misleading evaluations. MCC, on the other hand, is specifically designed to handle imbalanced class distributions, making it an essential metric for evaluating models trained on such datasets.

Additionally, many classification metrics are inherently biased toward the majority class, especially in highly imbalanced datasets. MCC is symmetric, meaning that it treats both classes equally, regardless of their prevalence. It ensures that a classifier must correctly predict both classes to receive a high score, making it a much fairer evaluation metric in imbalanced datasets. When comparing multiple models trained on an imbalanced dataset, MCC provides a fair and consistent way to rank their performances. Unlike F1-score or accuracy, which may favour models biased toward the majority class, MCC gives a more objective evaluation of classification performance.

A practical example of MCC's effectiveness in imbalanced datasets can be seen in Table 6 involving patient classification. This example reinforces the importance of using MCC when evaluating models trained on highly imbalanced datasets.

Table 7: A summary of studies on class imbalance.

#	Authors	Origin	Title	Summary	Major Themes
1	Fernandez, A., & Garcia, S.	Spain	Foundations on Imbalanced Classification	Provides an overview of imbalanced classification challenges, techniques, and evaluation strategies.	Imbalanced Classification, Machine Learning, Evaluation Techniques
2	Luque, A., Carrasco, A., Martín, A., & de las Heras, A.	Spain	The impact of class imbalance in classification performance metrics based on the binary confusion matrix	Explores how class imbalance affects classification performance metrics, highlighting MCC's robustness.	Class Imbalance, Confusion Matrix, MCC, Performance Metrics
3	Saito, T., & Rehmsmeier, M.	Germany	The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets	Demonstrates that precision-recall curves offer better insights than ROC curves for imbalanced datasets.	Evaluation Metrics, Imbalanced Classification, Precision-Recall
4	Shannon, C. E.	USA	A Mathematical Theory of Communication	Introduces information theory, entropy, and channel capacity, forming the foundation of modern communication systems.	Information Theory, Entropy, Channel Coding
5	Weiss, G. M.	USA	Imbalanced Learning: Foundations, Algorithms, and Applications	Explores the foundations of imbalanced learning, and provide a clear mapping of these issues to the methods that can be used to address them.	Class Imbalance, Machine Learning, Sampling methods

4. Methodology

This section outlines the methodological approach used to explore the effectiveness of the MCC in evaluating classification performance under different levels of class imbalance. The methodology is grounded in insights derived from research papers on classification metrics and class imbalance, guiding the experimental design and analytical framework.

The research process follows the structured timeline in the Gantt chart, detailing key milestones from literature review to experimental evaluation. The methodology involves implementing code to assess class imbalance and analyse how MCC compares against conventional metrics such as accuracy, precision, recall, and F1-score. By systematically varying the degree of imbalance in synthetic and real-world datasets, this study aims to highlight MCC's robustness across diverse classification scenarios.

4.1 Hypothesis

This study hypothesises that MCC offers a reliable and comprehensive evaluation of classification performance in imbalanced datasets just like conventional metrics such as accuracy, precision, recall, and F1-score. MCC provides informative assessment in scenarios with skewed class distributions, ensuring that both minority and majority classes contribute equally to model evaluation. MCC provides a more balanced view of classification performance and should be included as a standard evaluation metric alongside accuracy, F1-score, and precision-recall curves. MCC's mathematical formula incorporates all four components of the confusion matrix, making it particularly valuable in domains such as medical diagnosis, fraud detection, and bioinformatics, where imbalanced datasets are prevalent.

To test this hypothesis, the research adopts a methodology that develops a quantitative imbalance metric to effectively measure class distribution disparities while capturing key statistical properties of skewed datasets. Classification scenarios are simulated with varying sample sizes and imbalance levels to analyse how MCC performs along with other accuracy metrics under diverse conditions. Through this approach, the study aims to empirically validate MCC's robustness and effectiveness as a classification metric in imbalanced data settings.

To ensure a consistent and controlled comparison of evaluation metrics, this research employs the Random Forest classifier throughout the analysis. Random Forest is chosen due to its robustness, versatility, and widespread adoption in machine learning applications. As an ensemble learning method, it mitigates overfitting, handles both balanced and imbalanced datasets effectively, and provides stable predictions across various scenarios (Mohammed Zakariah, 2014). By using a single, well-established model, this study isolates the impact of different evaluation metrics without introducing variability caused by model selection, thereby ensuring that the findings remain focused on metric effectiveness rather than model performance.

4.2 Class Imbalance Metrics

1. Imbalance Ratio based on class proportions

The imbalance ratio (IR) is a simple way to quantify the degree of skewness in class distributions by calculating the ratio of the majority class count to the total sample.

$$IR = \frac{N_{majority}}{N_{minority}}$$

Based on prior research, there is no agreement or standard concerning the exact degree of class imbalance required for a data set to be considered truly "imbalanced." But most practitioners would certainly agree that a data set where the most common class is less than twice as common as the rarest class would only be marginally unbalanced, that data sets with an IR of about 10:1 would be modestly imbalanced, and data sets with IRs above 1000:1 would be extremely unbalanced. But ultimately what we care about is how the imbalance impacts learning and, in particular, the ability to learn the rare classes (Weiss, G. M., 2013).

As such, for this study, the IR threshold used to classify datasets into different levels of imbalance follows:

- Slight Imbalance: $1.5 \leq IR < 3$
- Moderate Imbalance: $3 \leq IR < 10$
- Extreme Imbalance: $IR \geq 10$

Understanding IR is crucial in evaluating model performance across datasets with varying imbalance levels. Metrics like MCC, F1-score, and precision-recall curves behave differently as IR increases, and incorporating IR-based analysis can provide deeper insights into classifier effectiveness.

2. Shannon Entropy

Shannon entropy, commonly used in information theory, is also an effective tool for quantifying class imbalances in datasets. Shannon entropy is related to class imbalance because it quantifies the uncertainty or diversity in a class distribution. It provides a numerical measure of how balanced or imbalanced a dataset's class labels are. Shannon entropy is defined as:

$$H(x) = -\sum_{i=1}^{k} p_i log_2(p_i)$$

where p_i is the proportion of the i^{th} class, k is the total number of classes.

Entropy H reaches its maximum value $log_2(N)$ when all classes are equally represented, i.e. $p_i = \frac{1}{N} \forall i$. A normalized entropy $H_{norm} = \frac{H}{log_2(N)}$ scales to the range (0, 1]. When one class dominates, $p_k \approx 1$ and the entropy H approaches 0. This reflects minimal diversity in the dataset (Shannon, 1948).

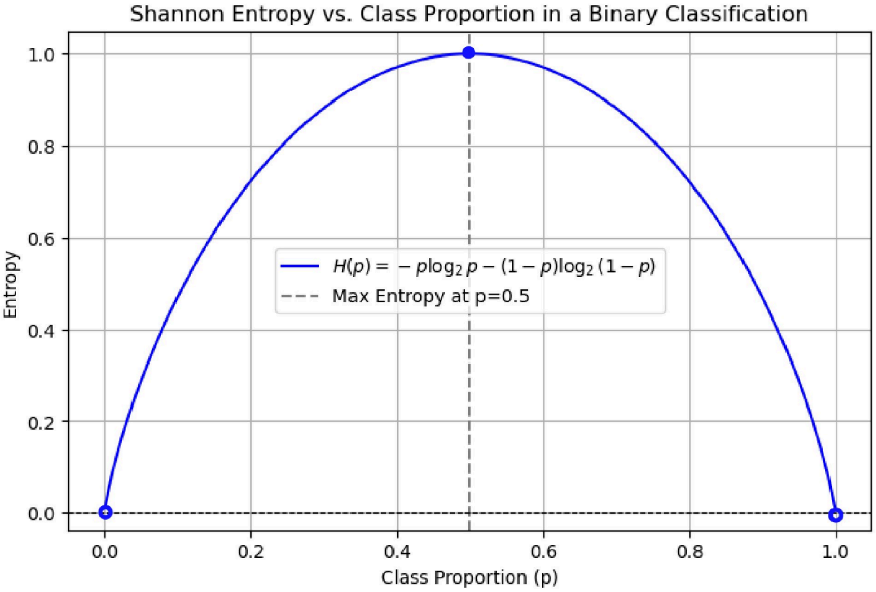
A high normalized entropy (closer to 1) indicates a balanced dataset where all classes are nearly equally represented. Such datasets are less likely to suffer from class imbalance issues. A low normalized entropy (closer to 0) indicates significant imbalance, where one or more classes dominate the dataset. This can lead to biased model training and reduced generalisability.

This approach allows entropy to serve as a generalised measure of imbalance, particularly when comparing datasets with different class distributions. Previous studies have used entropy to assess class dominance in machine learning models (Shannon, 1948). In the context of class imbalance, it measures the diversity of class distributions. A perfectly balanced dataset, where all classes have equal representation, achieves maximum entropy. Conversely, as the dataset becomes more imbalanced, the entropy decreases, indicating less unpredictability in class membership.

Consider a simple example of a two-class distribution dataset with two different scenarios.

1. Balanced Dataset (50-50 class proportion)
   $p_1 = 0.5,\ p_2 = 0.5$
   $H = -(0.5log_2 0.5 + 0.5log_2 0.5) = 1$ (maximum entropy)
2. Imbalanced Dataset (90-10 class proportion)
   $p_1 = 0.9,\ p_2 = 0.1$
   $H = -(0.9log_2 0.9 + 0.1log_2 0.1) \approx 0.469$ (lower entropy)

Image 3: Visualization of Shannon entropy against class proportion



At p = 0, entropy is zero because all data belongs to a single class, resulting in no uncertainty or impurity but maximum class imbalance. A classifier can trivially predict the majority class but lacks generalisation ability. At p = 0.5, entropy reaches its peak as both classes are equally represented, creating the highest uncertainty and impurity while minimising class imbalance. This balanced distribution makes classification most challenging. At p = 1, entropy again drops to zero, identical to p = 0, as the dataset is entirely homogeneous with no class diversity.

Image 4: Visualization of Shannon entropy against class proportion

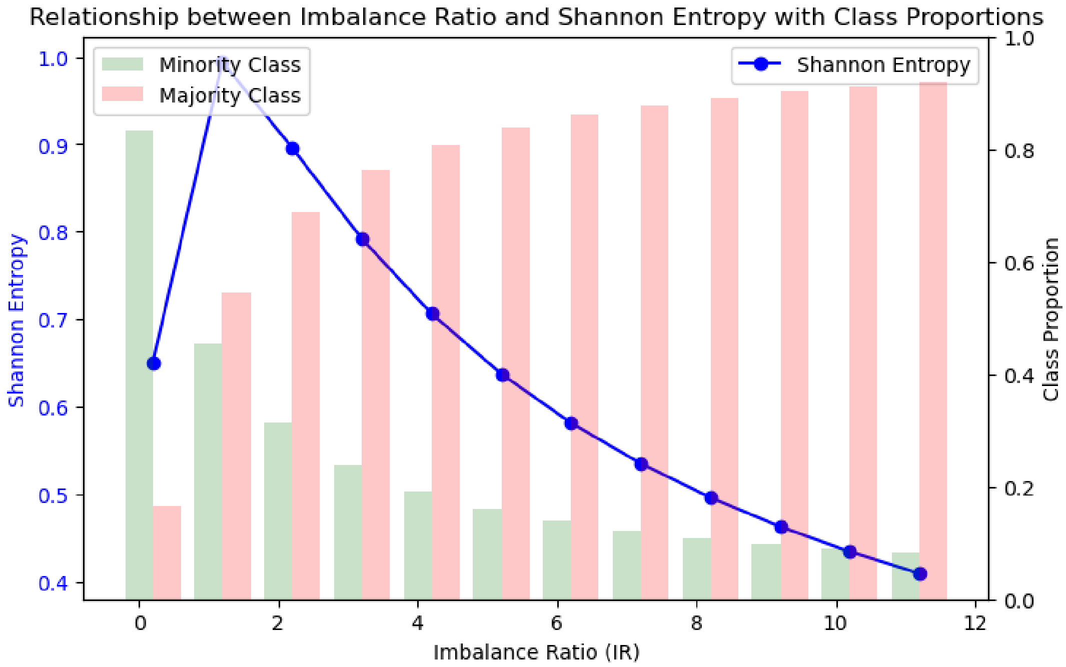


Image 4 illustrates the relationship between IR and Shannon Entropy, highlighting how class distribution affects dataset uncertainty. At low IR values (near 1), entropy is at its highest, indicating a balanced dataset with maximum uncertainty and impurity. As IR increases, meaning one class becomes increasingly dominant, entropy steadily declines, reflecting reduced uncertainty due to the growing imbalance. When IR is extreme, entropy approaches zero, signifying a near-pure dataset where classification becomes trivial but lacks meaningful learning.

Table 7: Summary table for IR and Shannon entropy.

Metric	Imbalance Ratio	Shannon Entropy
Definition	Measures proportion ratio between largest and smallest class	Measures probabilistic uncertainty in class distribution
Range	(0, ∞)	(0, 1]
Best for	Binary datasets, quick imbalance check	Multi-class datasets, general imbalance analysis
Sensitive to	Only the most and least frequent classes	All class distributions

Algorithm 1 *measure_class_imbalance*

Input: Data set y
Output: IR, Shannon entropy

```
1: # Compute class distribution
   class_counts ← COUNT_INSTANCES(y)
   n_samples ← LENGTH(y)
   n_classes ← NUMBER_OF_UNIQUE_CLASSES(y)
2: # Compute IR
   majority ← MAXIMUM_VALUE(class_counts)
   minority ← MINIMUM_VALUE(class_counts)
   imbalance_ratio ← majority / minority if minority > 0 else ∞
3: # Compute Shannon Entropy
   class_probabilities ← class_counts / n_samples
   Shannon_entropy ← -SUM(class_probabilities * LOG2(class_probabilities + OFFSET))
```

Both IR and Shannon entropy are necessary to provide complementary insights into class imbalance. IR is a simple, intuitive measure that quickly identifies the ratio of the largest to smallest class. Making IR ideal for binary classification. However, it ignores the distribution of intermediate classes in multi-class settings. Shannon entropy, on the other hand, captures the overall diversity of class distributions, where multiple classes may be imbalanced at different levels. Using both ensures a more complete understanding: IR for quick imbalance detection and Shannon entropy for deeper multi-class analysis.

4.3 Simulation

To empirically evaluate the performance of classification metrics under varying levels of class imbalance, a simulation framework is designed. This framework generates synthetic datasets with controlled IRs, applies classification models, and measures the effectiveness of different evaluation metrics, including MCC, accuracy, precision, recall, and F1-score.

Varying class proportions ranging from extreme imbalance (i.e. 1:99) to perfect balance (i.e. 50:50) are simulated to mimic real-world skewed distributions commonly observed in domains such as fraud detection and medical diagnostics. For each synthetic dataset, classification models are trained and evaluated using MCC, accuracy, precision, recall, and F1-score. The performance of these metrics is then analysed under different imbalance conditions to determine their reliability and sensitivity to skewed class distributions.

Additionally, to assess the robustness of MCC and other metrics across various data scales, the simulation of various class proportions is repeated with different dataset sizes. This ensures that the findings are not biased by specific sample sizes and that MCC remains stable across different scales of classification tasks.

By integrating synthetic data generation, entropy-based imbalance measurement, and comparative metric evaluation, this simulation framework provides a comprehensive empirical foundation for validating MCC's effectiveness in handling class imbalance.

4.4 Choice of Dataset

Besides synthetically generated data, the Credit Card Fraud Detection Dataset from Kaggle has been selected for this study due to its highly imbalanced nature, making it an ideal candidate for evaluating the performance of classification metrics under different class distribution scenarios. This dataset consists of 284,807 transactions, with only 492 fraudulent cases, representing a fraud occurrence rate of 0.172%. The extreme imbalance in this dataset mirrors real-world applications where rare events, such as fraud detection, medical diagnosis, and cybersecurity threats, pose significant challenges to traditional classification metrics.

Algorithm 3 *Simulation with Synthetic Dataset*

```
1: DEFINE sample_sizes = [100, 500, 1000, 5000, 10000]
   DEFINE minority_proportions = [1, 2, 3,..., 49, 50]
   results = []
2: FOR each sample size:
         FOR each minority proportion:
             GENERATE datasets with different class imbalance levels
             COMPUTE IRs and Shannon entropy for each dataset
             TRAIN random forest classifier
             PREDICT y_pred on test data
             COMPUTE accuracy, precision, recall, f1, MCC
3: PLOT metrics’ performance against IR / Shannon entropy
```

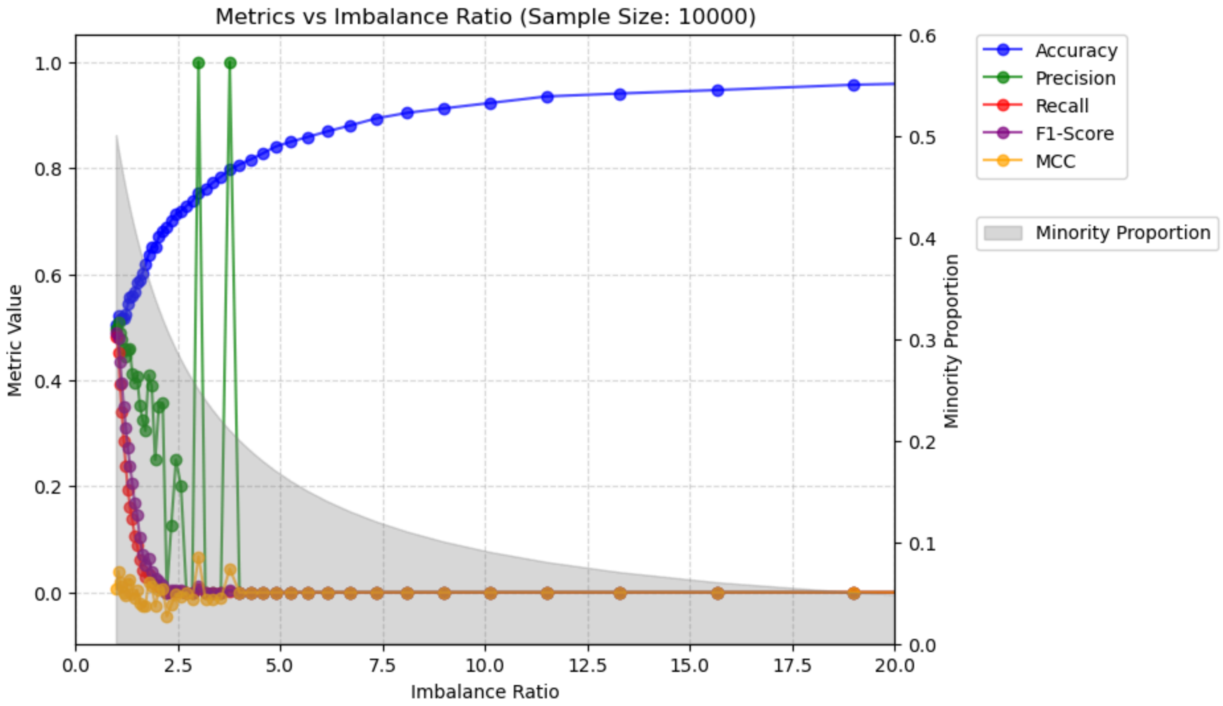
Algorithm 4 *Simulation with Credit Card Fraud Dataset*

```
1: LOAD dataset 'creditcard.csv'
   EXTRACT features X and target y ('Class')
   DEFINE minority_proportions = [1, 2, 3,..., 49, 50]
2: FOR each proportion in minority_proportions
            DO APPLY SMOTE to balance classes at given proportion
            SPLIT resampled data into 70% training and 30% testing
            TRAIN random forest classifier
            PREDICT y_pred on test data
            COMPUTE accuracy, precision, recall, f1, MCC
3: PLOT metrics vs. minority proportion
```

5. Results and Interpretations

5.1 Synthetically Generated Dataset

Image 5: Classification Metrics vs. IR (n = 10000)



The plots illustrate the behavior of these metrics against both Shannon Entropy and IR across a large dataset (n = 10,000), providing insights into how model performance is affected by varying degrees of class distribution skewness.

Image 5, which maps classification metrics against the IR, confirms the limitations of conventional evaluation metrics in imbalanced datasets. Accuracy remains artificially high even in highly imbalanced cases, confirming its insensitivity to dataset skewness. While accuracy gradually declines as the dataset becomes more balanced, it remains consistently inflated compared to other evaluation metrics. Precision displays extreme fluctuations at low IRs, reflecting instability in the model's predictions for the minority class. Recall, on the other hand, shows a sharp decline as imbalance increases, reinforcing the model's bias toward predicting the majority class. F1-score follows the pattern of recall, fluctuating significantly at low IRs but stabilising as the dataset becomes increasingly skewed. MCC remains close to zero, confirming that the model struggles to achieve a balanced classification as the imbalance worsens. Unlike accuracy, MCC correctly reflects the degradation in classification performance as the IR increases, validating its role as a more reliable and informative metric for imbalanced datasets.

Image 6: Classification Metrics vs. Shannon Entropy (n = 10000)

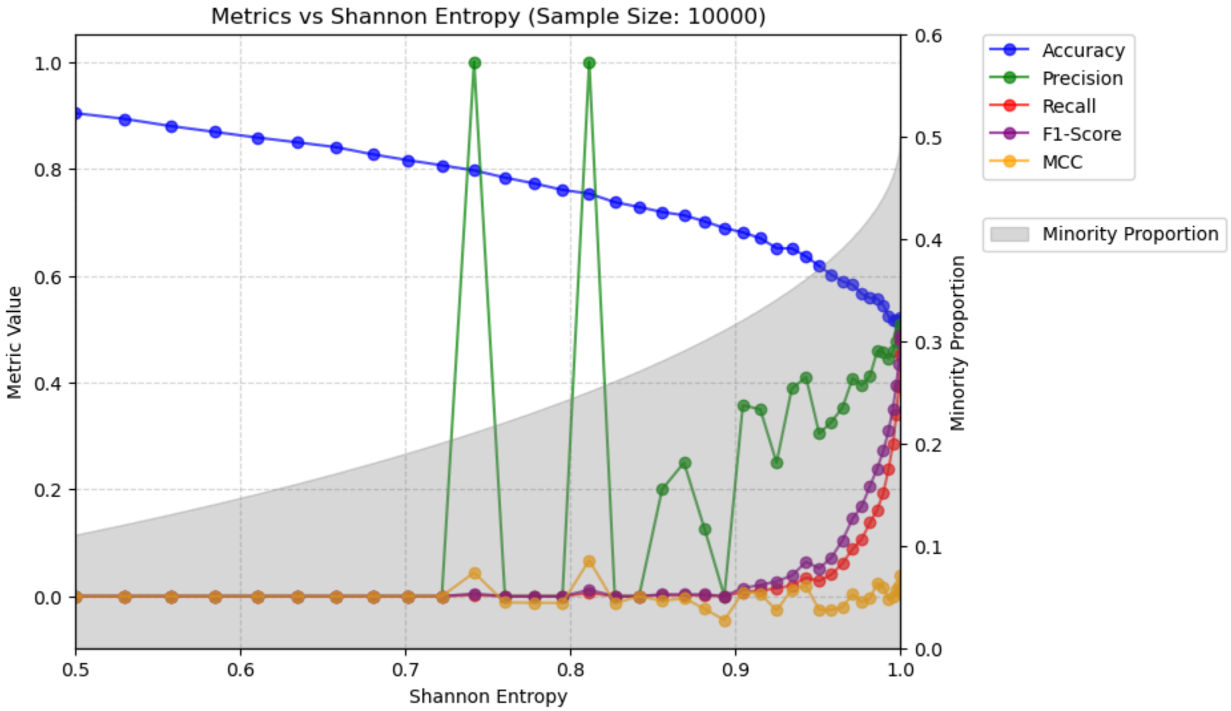


Image 6, depicting classification metrics against Shannon Entropy, reinforces the findings from Image 5, showing similar trends in classification performance under imbalance. Like in Image 5, accuracy remains high, gradually declining as entropy increases, confirming its insensitivity to class imbalance. Precision fluctuates significantly, particularly at higher entropy values, mirroring their erratic behavior in Image 5 at low imbalance ratios. F1-score follows recall and precision trends, further highlighting its instability in imbalanced settings.

MCC remains consistently low, just as in Image 5, effectively capturing classification performance deterioration across imbalance levels. Unlike accuracy and F1-score, MCC provides a stable, reliable measure, reinforcing its value as a standard metric for imbalanced datasets. The alignment between Image 5 and Image 6 demonstrates that both imbalance ratio and Shannon entropy yield consistent conclusions, strengthening the argument for MCC's effectiveness over conventional metrics for imbalanced datasets.

5.2 Credit Card Fraud Dataset

Image 7: Classification Metrics vs. Shannon Entropy (Credit Card Data)

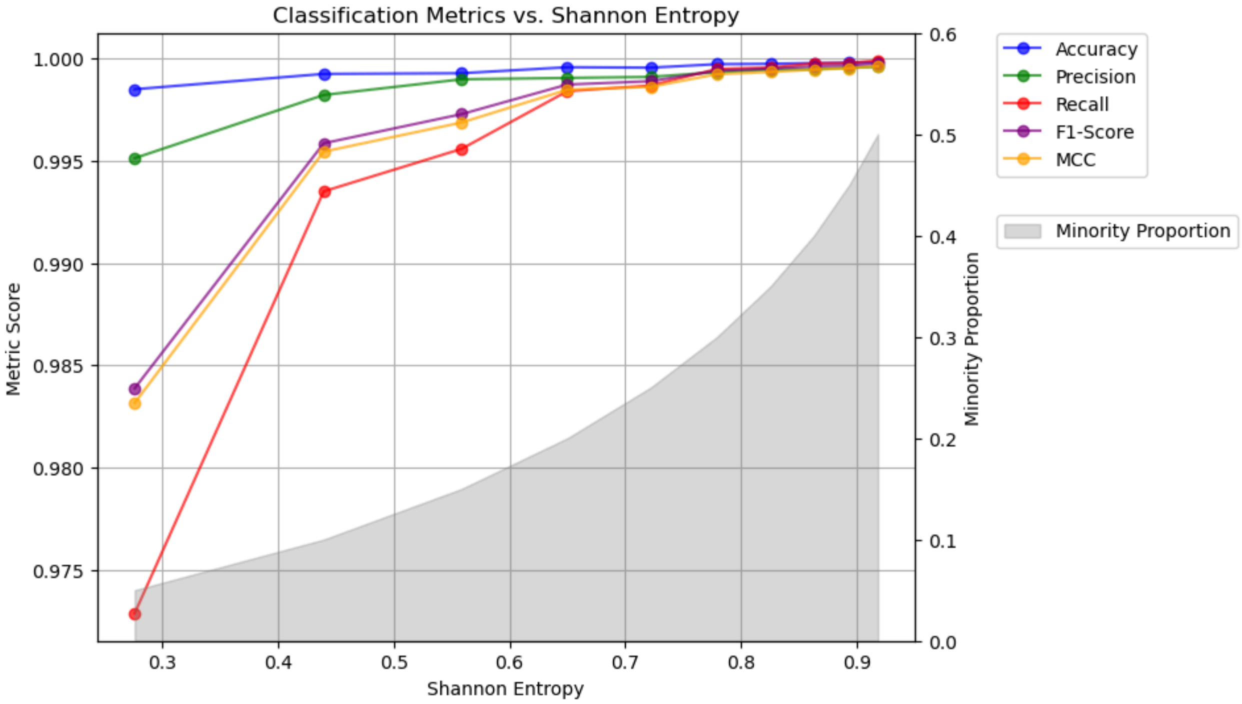


Image 7 illustrates the relationship between classification metrics and Shannon Entropy using the fraud detection credit card dataset. As Shannon Entropy increases, indicating a more balanced class distribution, all classification metrics converge toward their highest values. This suggests that the model performs optimally when the dataset is more balanced, reinforcing the notion that extreme class imbalance negatively impacts classification performance.

Accuracy remains consistently high across different entropy levels, with only a slight improvement as entropy increases. This highlights its insensitivity to class imbalance, as it does not adequately reflect changes in minority class predictions. Precision follows a similar trend, showing a steady rise before stabilizing, indicating that the classifier improves in identifying true positives as class balance improves. Recall, on the other hand, starts at a lower value and increases sharply with entropy, suggesting that models trained on imbalanced datasets struggle to detect minority class instances but significantly improve as the class distribution evens out.

F1-score follows recall closely, exhibiting a sharp increase at lower entropy levels before stabilizing at higher entropy values. This suggests that F1-score is heavily influenced by recall in imbalanced datasets, as misclassification of the minority class drastically lowers its value. MCC, a balanced evaluation metric, follows the same pattern as F1-score, indicating that it effectively captures classification performance across different levels of class imbalance. Unlike accuracy, MCC does not remain artificially high in imbalanced scenarios, reinforcing its reliability in evaluating classification performance under skewed distributions.

While both synthetic and real datasets exhibit similar trends—where classification metrics improve with increasing entropy—the real dataset demonstrates a more gradual and stable increase in metric values. In contrast, the synthetic data shows greater fluctuations in metric scores at high entropy levels, highlighting the unpredictability of minority class detection in artificially generated data. Despite these differences, the key takeaway remains consistent across both datasets: accuracy and other traditional classifier metrics alone are inadequate for imbalanced classification, while MCC consistently provides a stable and reliable assessment of model performance.

6. Concerns

6.1 Limitations

This study primarily focuses on the application of MCC in binary classification settings. While MCC can be extended to multi-class problems, its behaviour in such scenarios remains less explored in this research. Future studies should investigate how MCC performs in multi-class settings, considering techniques such as macro-averaging MCC or alternative MCC-based formulations to assess classification performance more comprehensively.

A key limitation of this study lies in its sensitivity to resampling techniques. The Synthetic Minority Oversampling Technique (SMOTE) was used to balance class proportions in the Credit Card Fraud dataset; however, SMOTE may introduce synthetic patterns that do not always accurately represent real-world distributions. This could lead to a biased understanding of MCC's performance under resampling conditions. Exploring alternative strategies, such as cost-sensitive learning, adaptive boosting, or hybrid ensemble methods, may provide a more realistic evaluation of classification performance in imbalanced settings.

The limited scope of datasets presents another constraint. While MCC was analysed using both synthetic data and the Credit Card Fraud dataset, these datasets may not fully generalise to other domains with vastly different class distributions, such as medical diagnosis, cybersecurity, or natural language processing (NLP). Expanding the dataset selection could help strengthen the study's applicability across multiple real-world scenarios and provide a more robust evaluation of MCC under various conditions.

Another limitation is computational constraints in large-scale simulations. The study evaluates MCC's performance across different sample sizes, ranging from 10 to 10,000 instances. However, conducting large-scale experiments with datasets containing millions of samples could provide better insights into MCC's scalability and computational efficiency. Due to hardware and processing limitations, this study was unable to fully explore the effects of MCC on massive datasets, but future research could incorporate distributed computing techniques to address this challenge.

6.2 Areas for Improvement

One of the most critical areas for future research is the extension of MCC analysis to multi-class classification. While MCC has been extensively studied in binary classification, its performance in multi-class settings remains an open question. Future studies could explore macro-averaged MCC, one-vs-all MCC formulations, or alternative approaches for generalising MCC to multi-class problems. Such research could provide a deeper understanding of how MCC behaves when evaluating classifiers in scenarios with more than two categories.

Another promising direction involves comparative studies with other class imbalance solutions. Investigating alternative imbalance-handling techniques such as class-weighted loss functions, cost-sensitive learning, and anomaly detection methods could help determine how MCC compares under different imbalance-handling strategies. A comprehensive benchmark study analyzing MCC alongside other metrics in various imbalance-handling approaches would provide valuable insights into when MCC is most effective.

By addressing these limitations and areas for improvement, future research can enhance MCC's applicability, extend its utility to multi-class problems, and refine classification performance evaluation in highly imbalanced datasets. Expanding the scope of MCC's evaluation will not only solidify its position as a robust classification metric but also provide practitioners with a more comprehensive toolset for handling imbalanced data challenges.

6.3 Future Development

Future research can build upon this study by extending the application of the MCC beyond binary classification. While MCC is widely regarded as a strong evaluation metric for imbalanced datasets, its adaptation to multi-class problems remains an area for further exploration. Future work could investigate different formulations of MCC for multi-class classification, such as macro-averaging MCC scores across different classes or introducing weighing mechanisms that account for varying levels of imbalance in multi-class settings. Comparative studies between MCC and other multi-class metrics like Cohen's Kappa and Balanced Accuracy could provide valuable insights into its reliability in more complex classification problems.

Another promising direction is the evaluation of MCC in combination with alternative class imbalance handling techniques. This study applied SMOTE for oversampling, but future research could analyse MCC's performance under different strategies such as cost-sensitive learning, hybrid sampling approaches, and anomaly detection methods.

Expanding the scope of evaluation across diverse machine learning models would further strengthen the applicability of MCC. While this research utilised logistic regression, future studies could analyse MCC's behaviour in deep learning architectures such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformers. Additionally, ensemble learning techniques like XGBoost, Random Forest, and LightGBM could be evaluated to determine how MCC interacts with models that rely on boosting or bagging strategies. Investigating MCC in interpretable machine learning models would also be a valuable extension, ensuring that performance gains align with real-world decision-making needs.

Overall, building on this research by extending MCC's applicability, evaluating it under diverse machine learning models, integrating it with explainability and fairness methods, and scaling it to real-world applications would significantly enhance its role as a comprehensive classification performance metric. Future studies in these areas will further validate MCC as a reliable and interpretable measure for handling class imbalance across a wide range of machine learning domains.

7. Conclusion

This study explored the effectiveness of the MCC as a classification performance metric, particularly in imbalanced datasets. Traditional evaluation metrics such as accuracy, precision, recall, and F1-score may fail to provide a comprehensive assessment when class distributions are skewed, as they can be biased toward the majority class. MCC, in contrast, considers all four components of the confusion matrix, offering a more balanced evaluation of classification performance. The research demonstrated that MCC remains a robust and reliable metric across different levels of class imbalance, as confirmed through both synthetic data simulations and real-world evaluation using the Credit Card Fraud Dataset.

By quantifying class imbalance using Imbalance Ratio and Shannon entropy, the study analysed the relationship between class distribution and classification performance. The findings reinforced that accuracy tends to overestimate performance in imbalanced datasets, while MCC aligns more closely with dataset diversity and class balance. The results showed that MCC provides a clearer measure of model reliability in detecting minority class instances, making it particularly valuable in applications such as fraud detection, medical diagnosis, and anomaly detection.

Despite its strengths, this research also highlighted certain limitations, such as the exclusive focus on binary classification and the reliance on SMOTE for class balancing. Future work should extend MCC to multi-class settings, explore alternative imbalance-handling techniques, and assess MCC's effectiveness across a wider range of machine learning models. Additionally, integrating MCC into explainable AI frameworks and real-time classification systems could further enhance its practical impact.

Ultimately, this study reaffirms MCC as an effective metric for evaluating models in imbalanced scenarios, and when incorporated into the standard classifier evaluation regime, it can help to provide better insights into a model’s performance. Future research should continue refining its applications, ensuring that machine learning models are evaluated using metrics that truly reflect their ability to generalise across all classes, particularly in critical real-world domains.

8. Epilogue

Writing this paper has been an invaluable learning experience that deepened my understanding of classification evaluation metrics and handling class imbalance in machine learning. As a data analytics student, I have worked with traditional performance measures like accuracy, precision, and recall, but this research has shown me their limitations, particularly in highly imbalanced datasets. Exploring MCC as a more reliable metric provided a clearer perspective on why balanced evaluation is necessary to prevent misleading model assessments.

A key takeaway was learning how Shannon entropy and IRs can quantify dataset skewness, providing a mathematical foundation for understanding class imbalance. Implementing SMOTE to resample data and performing simulations on both synthetic and real-world datasets allowed me to experiment with different approaches to mitigating class skewness. These experiences enhanced my ability to handle imbalanced datasets effectively, a critical skill for real-world applications where minority class instances are often underrepresented.

Beyond technical skills, this research reinforced my ability to think critically about model performance and evaluate results beyond surface-level accuracy scores. The findings demonstrated that accuracy often overestimates performance in imbalanced scenarios, whereas MCC provides a more comprehensive measure by considering both true positives and true negatives. I also encountered challenges related to structuring my research methodology. For example, I explored polynomial fitting and planned to express classification metric scores as a function of imbalance quantifying variables. However, results from polynomial fitting were inconclusive and did not align with expectations. Overcoming these problems improved my problem-solving skills and ability to optimise machine learning models.

The knowledge gained from this research will be directly applicable to my future career in data analytics and machine learning. Understanding alternative evaluation metrics, resampling techniques, and classification model validation equips me to work on high-stakes domains such as fraud detection, healthcare analytics, and financial risk assessment. Additionally, learning how to incorporate entropy-based imbalance measurements into model evaluation has provided me with tools for developing more interpretable, fair, and ethical machine learning solutions.

This research has not only strengthened my technical expertise in machine learning evaluation but also enhanced my ability to analyse, interpret, and optimise classification models in real-world settings. These insights will serve as a strong foundation for my future work, ensuring that I can confidently tackle challenges in predictive modelling, AI fairness, and data-driven decision-making while maintaining a well-rounded approach to model assessment and evaluation.

9. References

Boughorbel, S., Jarray, F., & El-Anbari, M. (2017). Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. *PLOS ONE, 12*(6), e0177678. https://doi.org/10.1371/journal.pone.0177678

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research, 16*, 321-357.

Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics, 21*(1), 1-13. https://doi.org/10.1186/s12864-019-6413-7

Cover, T. M., & Thomas, J. A. (2006). *Elements of information theory* (2nd ed.). Wiley.

Draper, N. R., & Smith, H. (1998). *Applied regression analysis* (3rd ed.). Wiley.

Fernández, A., Garcia, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). *Learning from imbalanced data sets*. Springer International Publishing. https://doi.org/10.1007/978-3-319-98074-4

Ferrer, L. (2022). Analysis and comparison of classification metrics. arXiv preprint arXiv:2209.05355

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*. https://arxiv.org/abs/2008.05756

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering, 21*(9), 1263-1284. https://doi.org/10.1109/TKDE.2008.239

Islam, T. U., & Rizwan, M. (2022). Comparison of Correlation Measures for Nominal Data. *Communications in Statistics - Simulation and Computation, 51*(3), 698-714. https://doi.org/10.1080/03610918.2020.1869984

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis, 6*(5), 429-449.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

Luque, A., Carrasco, A., Martín, A., & de las Heras, A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition, 91*, 216-231. https://doi.org/10.1016/j.patcog.2019.02.023

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems, 30*, 4765-4774.

Machine Learning Group - ULB. (2016). *Credit Card Fraud Detection Dataset*. Kaggle. Retrieved from https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Matthews, B. W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure, 405*(2), 442-451. https://doi.org/10.1016/0005-2795(75)90109-9

Powers, D. M. W. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Technologies, 2*(1), 37-63.

Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE, 10*(3), e0118432. https://doi.org/10.1371/journal.pone.0118432

Seber, G. A. F., & Wild, C. J. (2003). *Nonlinear regression*. Wiley.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal, 27*(3), 379-423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management, 45*(4), 427-437. https://doi.org/10.1016/j.ipm.2009.03.002

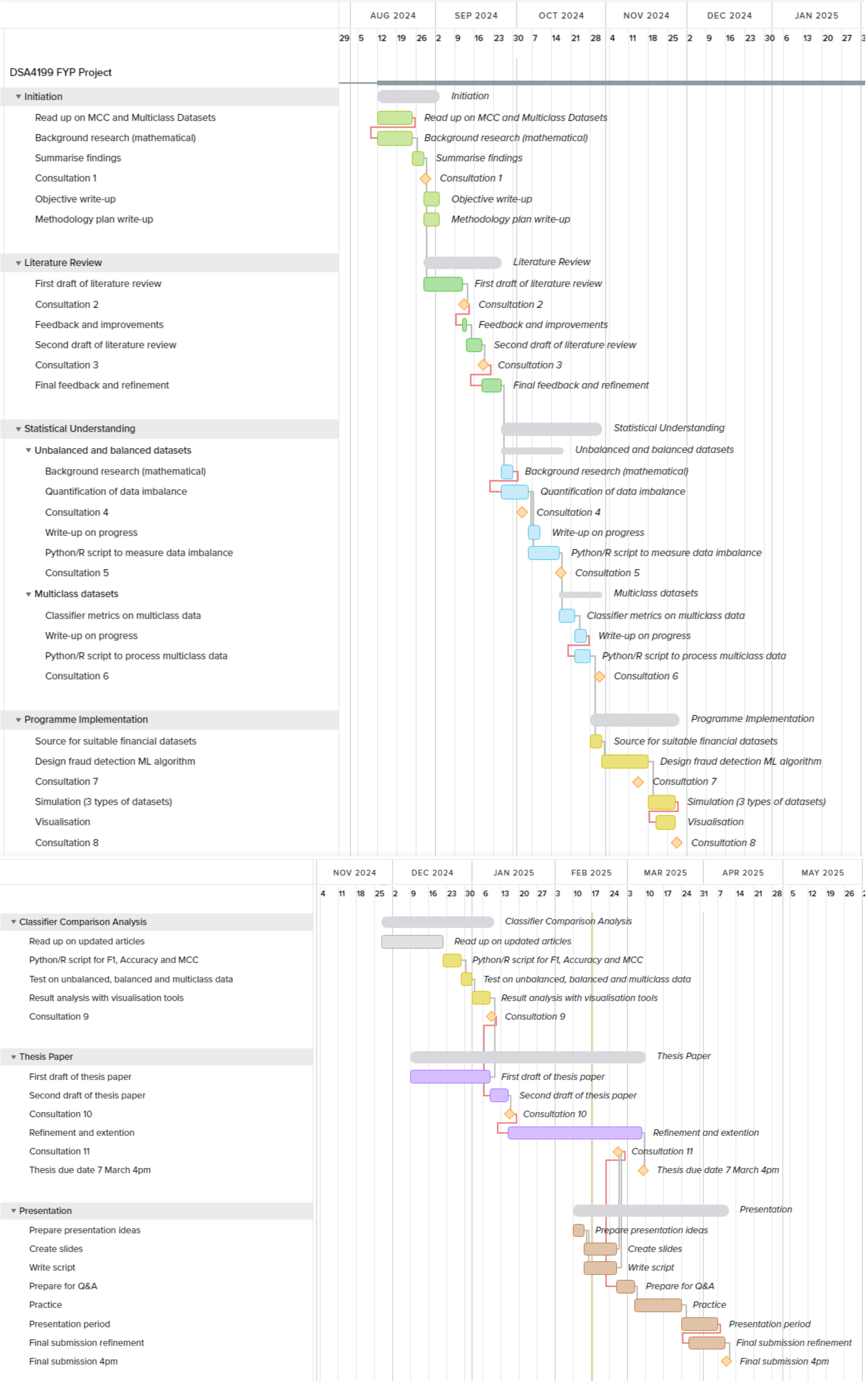
Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research, 9*(86), 2579-2605.

Van Hulse, J., Khoshgoftaar, T. M., & Napolitano, A. (2007). Experimental perspectives on learning from imbalanced data. *Proceedings of the 24th International Conference on Machine Learning*, 935-942.

Zakariah, M. (2014). Classification of large datasets using Random Forest Algorithm in various applications: Survey. International Journal of Engineering and Innovative Technology, 4(3), 189-196.

Appendix

Research Progress Gantt Chart



Summary of Other Research Papers

#	Authors	Origin	Title	Summary	Major Themes
1	Gana N. S. Pradeep	Unknown	A comprehensive review on ensemble deep learning: Opportunities and challenges	Provides analysis of deep learning methods, exploring their benefits in performance, robustness, and generalisation. Discusses various techniques, applications, and challenges in deep learning.	Deep Learning, Model Performance, Generalisation, Machine Learning Challenges
2	Good, P.	USA	Permutations, Parametrics and Bootstrap Tests of Hypotheses	Explores statistical hypothesis testing using permutation, parametric, and bootstrap methods.	Statistical Testing, Bootstrap Methods, Hypothesis Testing
3	Islam, T. U., & Rizwan, M.	Pakistan	Comparison of Correlation Measures for Nominal Data	Compares various correlation measures for nominal data, analysing their strengths and weaknesses.	Correlation Measures, Nominal Data, Statistical Analysis
4	Powers, D. M. W.	Australia	Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation	Explores evaluation metrics for binary classification, discussing ROC, precision-recall, and correlation measures.	Evaluation Metrics, Classification Performance, ROC, Precision-Recall

Python Code Implementation - MCC.ipynb

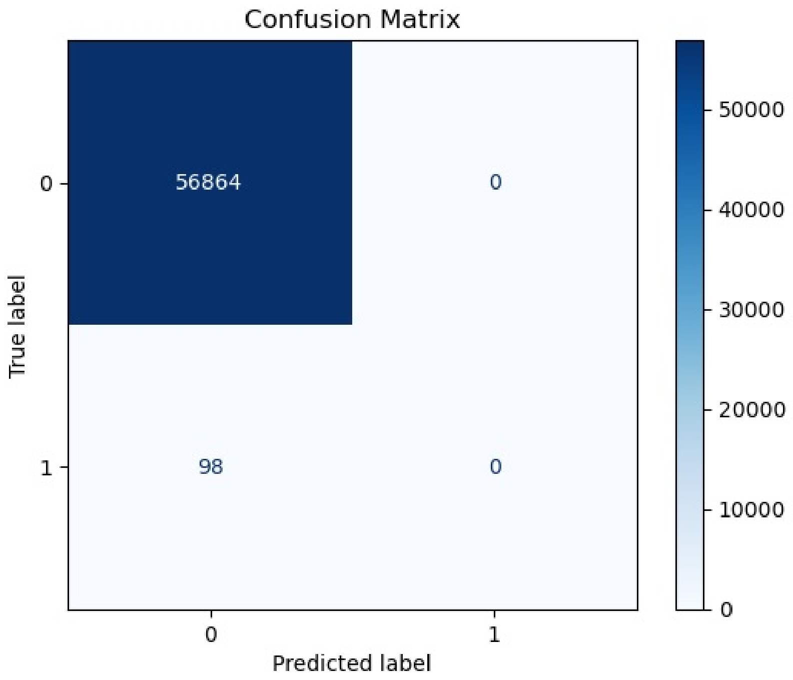
In [2]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import matthews_corrcoef, make_scorer, confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import classification_report, accuracy_score, precision_score, recall_score, f1_score
import matplotlib.pyplot as plt
from scipy.stats import entropy
from imblearn.over_sampling import SMOTE
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
df = pd.read_csv('creditcard.csv')
print(df["Class"].value_counts())

X = df.drop(columns=["Class"])
y = df["Class"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

y_pred_naive0 = np.zeros_like(y_test)  # Predict all as 0 (non-fraud)

# Compute Accuracy and MCC for both models
accuracy_naive = accuracy_score(y_test, y_pred_naive0)
mcc_naive = matthews_corrcoef(y_test, y_pred_naive0)

print("Naive Model Performance:")
print(f"\nAccuracy: {accuracy_naive:.4f}, MCC: {mcc_naive:.4f}\n")

cm = confusion_matrix(y_test, y_pred_naive0)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[0, 1])
print("True Positive:\t", cm[1][1])
print("False Positive:\t", cm[0][1])
print("True Negative:\t", cm[0][0])
print("False Negative:\t", cm[1][0])
disp.plot(cmap=plt.cm.Blues)
plt.title("Confusion Matrix")
plt.show()
```

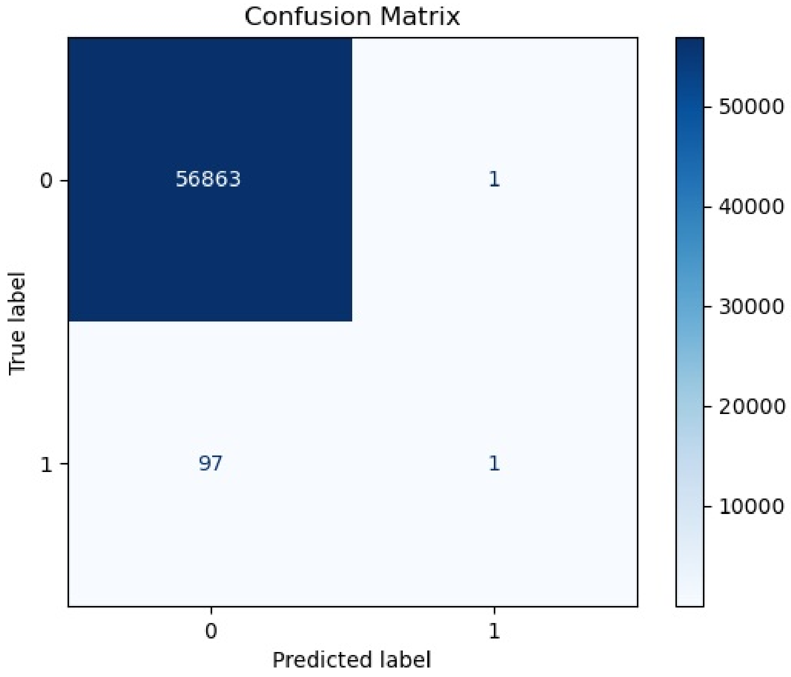
```
Class
0    284315
1       492
Name: count, dtype: int64
Naive Model Performance:

Accuracy: 0.9983, MCC: 0.0000

True Positive:   0
False Positive:  0
True Negative:   56864
False Negative:  98
```



In [4]:

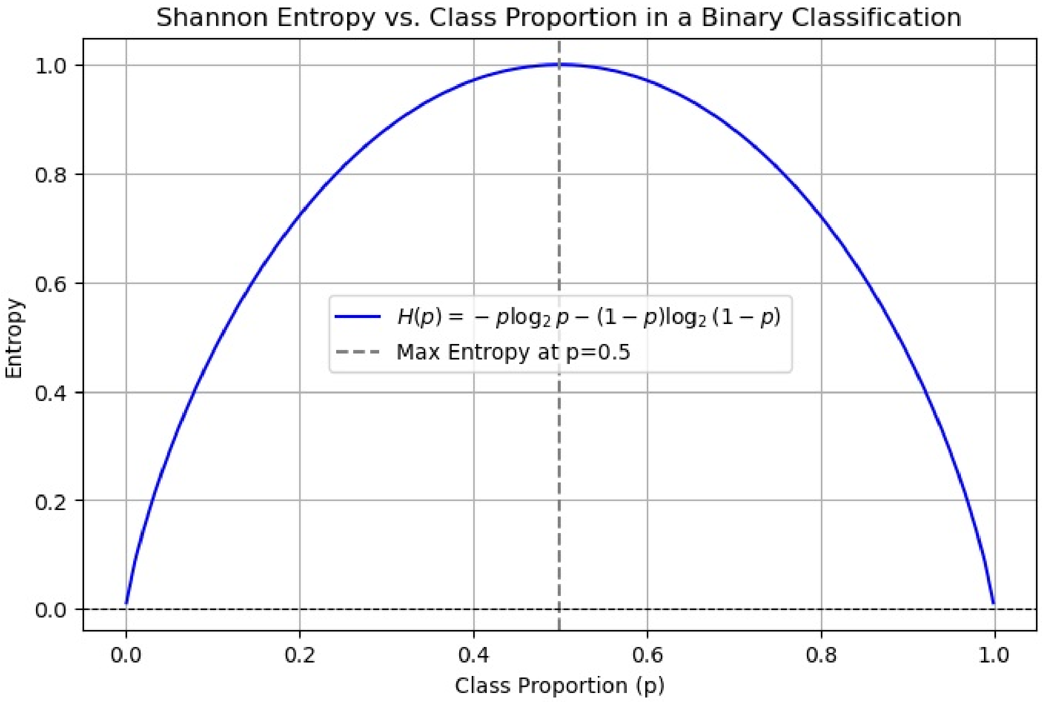
```
y_pred_naive1 = np.zeros_like(y_test)  # Predict all as 0 (non-fraud)
y_test_array = y_test.to_numpy()

fraud_indices = np.where(y_test_array == 1)[0]
non_fraud_indices = np.where(y_test_array == 0)[0]

# Hardcode TP and FP values
if len(fraud_indices) > 0:
    y_pred_naive1[fraud_indices[0]] = 1  # Set one fraud case as TP

if len(non_fraud_indices) > 0:
    y_pred_naive1[non_fraud_indices[0]] = 1  # Set one non-fraud case as FP

accuracy_naive = accuracy_score(y_test, y_pred_naive1)
mcc_naive = matthews_corrcoef(y_test, y_pred_naive1)

print("Naive Model Performance:")
print(f"\nAccuracy: {accuracy_naive:.4f}, MCC: {mcc_naive:.4f}\n")

cm = confusion_matrix(y_test, y_pred_naive1)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=[0, 1])

print("True Positive:\t", cm[1][1])
print("False Positive:\t", cm[0][1])
print("True Negative:\t", cm[0][0])
print("False Negative:\t", cm[1][0])

disp.plot(cmap=plt.cm.Blues)
plt.title("Confusion Matrix")
plt.show()
```

```
Naive Model Performance:

Accuracy: 0.9983, MCC: 0.0712

True Positive:   1
False Positive:  1
True Negative:   56863
False Negative:  97
```



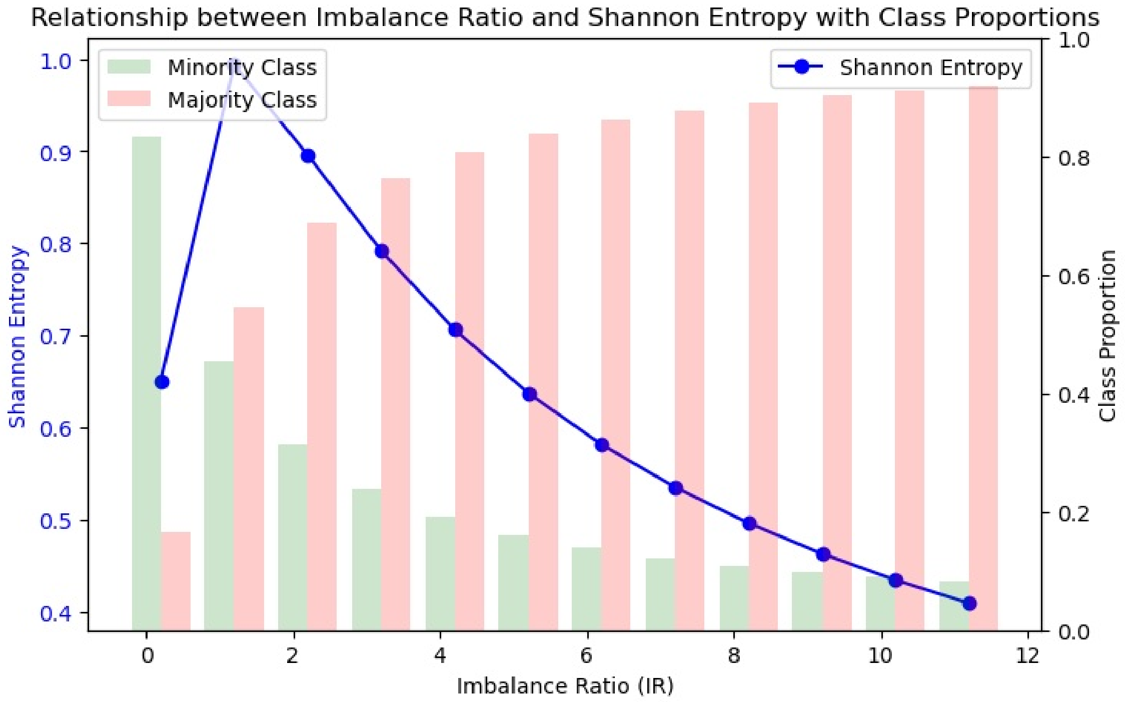
In [5]:

```
def entropy(p):
    return - (p * np.log2(p) + (1 - p) * np.log2(1 - p))

p_values = np.linspace(0.001, 0.999, 100)
entropy_values = entropy(p_values)

plt.figure(figsize=(8, 5))
plt.plot(p_values, entropy_values, label=r'$H(p) = - p \log_2 p - (1 - p) \log_2 (1 - p)$', color='b')
plt.axvline(x=0.5, linestyle="--", color="gray", label="Max Entropy at p=0.5")
plt.axhline(y=0, linestyle="--", color="black", linewidth=0.8)
plt.xlabel("Class Proportion (p)")
plt.ylabel("Entropy")
plt.title("Shannon Entropy vs. Class Proportion in a Binary Classification")
plt.legend()
plt.grid(True)
plt.show()
```



In [6]:

```
IR_values = np.arange(0.2, 12.2, 1)

minority_probs = 1 / (1 + IR_values)
majority_probs = IR_values / (1 + IR_values)

shannon_entropy = - (majority_probs * np.log2(majority_probs + 1e-9) +
                     minority_probs * np.log2(minority_probs + 1e-9))  # Avoid log(0)

fig, ax1 = plt.subplots(figsize=(8, 5))

ax1.plot(IR_values, shannon_entropy, marker='o', linestyle='-', color='b', label="Shannon Entropy")
ax1.set_xlabel("Imbalance Ratio (IR)")
ax1.set_ylabel("Shannon Entropy", color='b')
ax1.tick_params(axis='y', labelcolor='b')
ax1.set_title("Relationship between Imbalance Ratio and Shannon Entropy with Class Proportions")

ax2 = ax1.twinx()
width = 0.4

ax2.bar(IR_values - width/2, minority_probs, width=width, color='green', alpha=0.2, label="Minority Class")
ax2.bar(IR_values + width/2, majority_probs, width=width, color='red', alpha=0.2, label="Majority Class")

ax2.set_ylabel("Class Proportion")
ax2.tick_params(axis='y')
ax2.set_ylim(0, 1)

# Add legends
ax1.legend(loc='upper right')
ax2.legend(loc='upper left')

# Show plot
plt.show()
```



Measuring Class Imbalance in Datasets

In [8]:

```
def measure_class_imbalance(y):
    class_counts = np.bincount(y)
    n_samples = len(y)
    n_classes = len(class_counts)

    # Imbalance Ratio (IR)
    majority = class_counts.max()
    minority = class_counts.min()
    imbalance_ratio = majority/ minority if minority > 0 else np.inf  # Avoid division by zero

    # Shannon Entropy
    class_probabilities = class_counts / n_samples
    entropy = -np.sum(class_probabilities * np.log2(class_probabilities + 1e-9))  # Avoid log(0)
    shannon_entropy = entropy / np.log2(n_classes) if n_classes > 1 else 0

    return imbalance_ratio, shannon_entropy
```

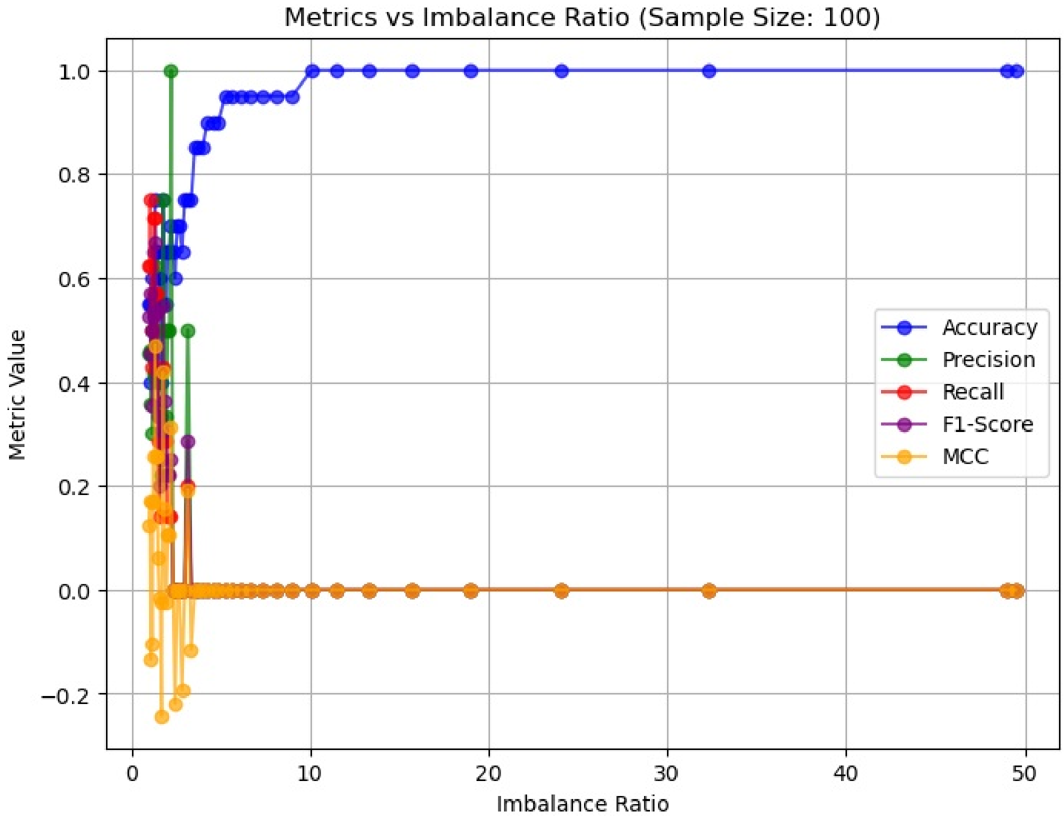
Simulating Various Class Proportions Fitted Against Different Metrics (Synthetically Generated Data)

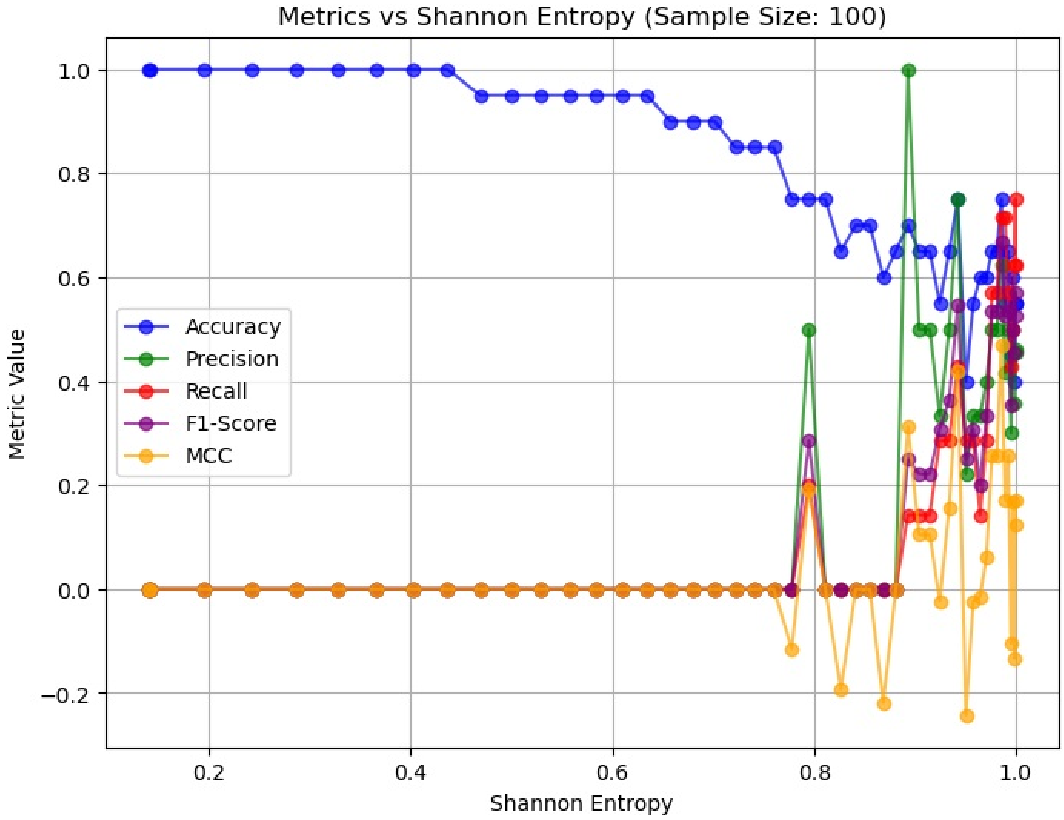
In [10]:

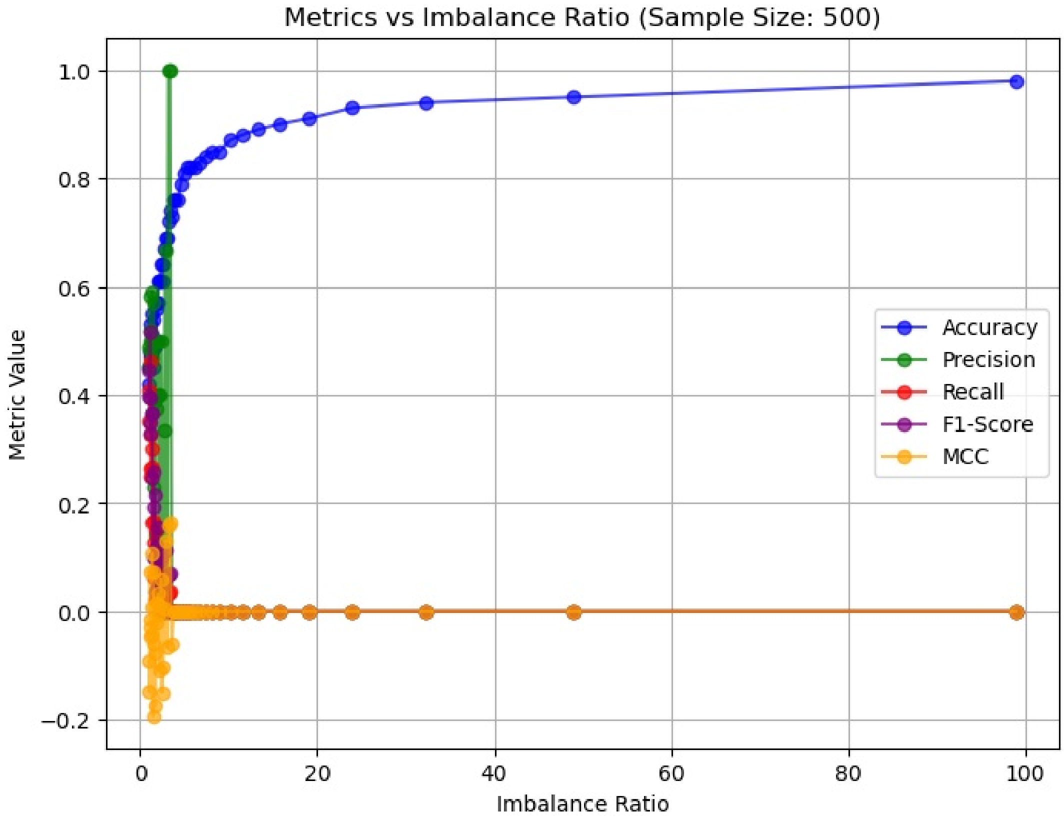
```
sample_sizes = [100, 500, 1000, 5000, 10000]
```

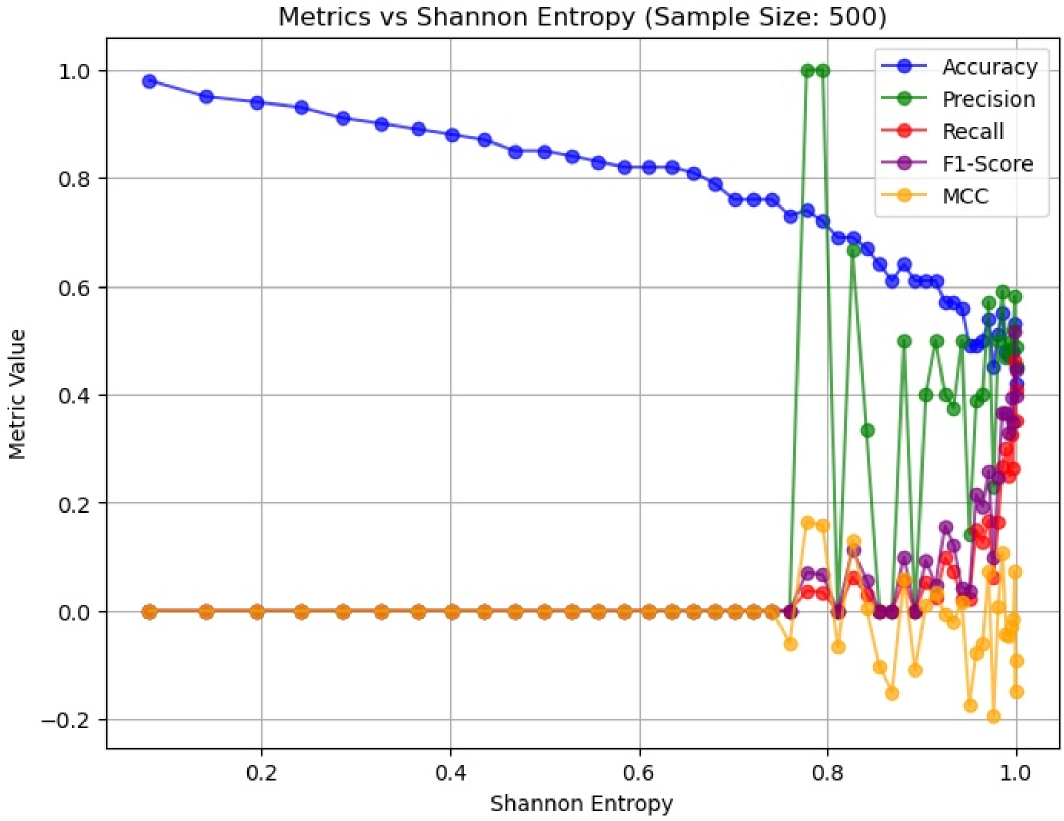
In [11]:

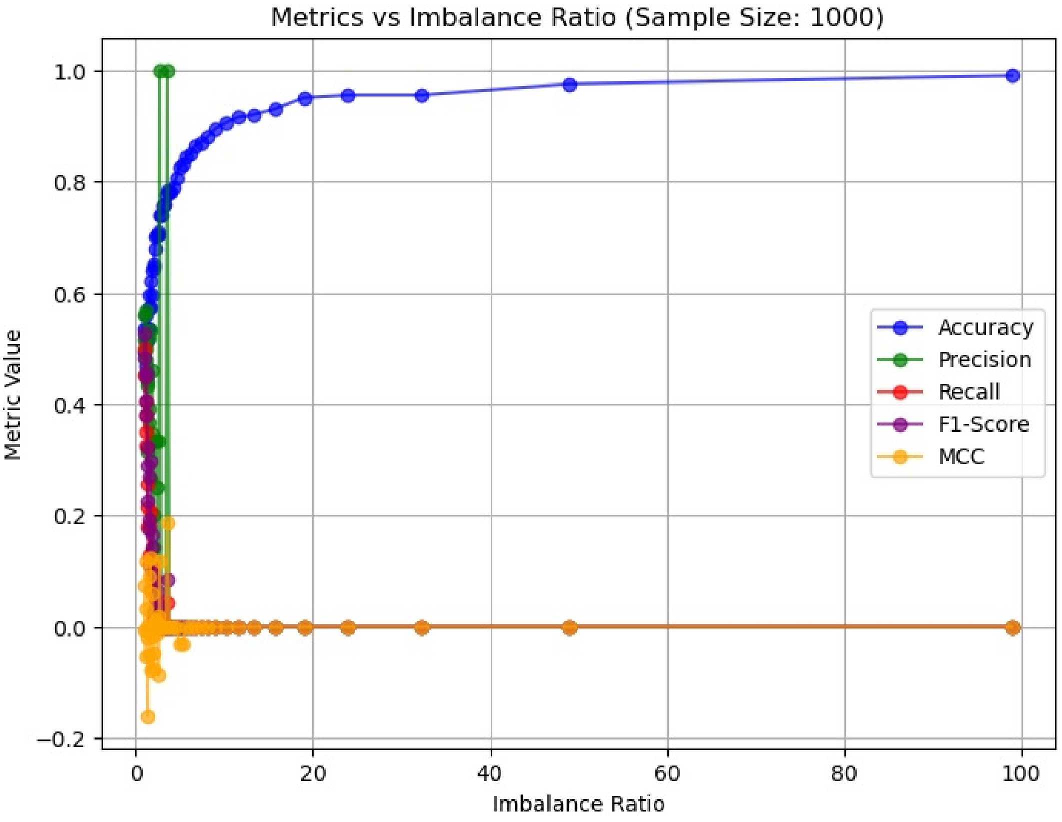
```
results = []
for sample_size in sample_sizes:
    imbalance_ratios = []
    shannon_entropies = []
    accuracy_values = []
    precision_values = []
    recall_values = []
    f1_values = []
    mcc_values = []

    for i in range(1,51):
        num_majority = max(2, int(sample_size * (100 - i) / 100))
        num_minority = max(2, int(sample_size * i / 100))

        if num_majority < 2 or num_minority < 2:
            continue  # Skip if we can't have both classes with at least 2 samples

        y = np.array([0] * num_majority + [1] * num_minority)
        X = np.random.rand(len(y), 10)

        ir, entropy = measure_class_imbalance(y)
        imbalance_ratios.append(ir)
        shannon_entropies.append(entropy)

        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

        clf = RandomForestClassifier(n_estimators=100, random_state=42, class_weight="balanced")
        clf.fit(X_train, y_train)
        y_pred = clf.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        precision = precision_score(y_test, y_pred, zero_division=0)
        recall = recall_score(y_test, y_pred, zero_division=0)
        f1 = f1_score(y_test, y_pred, zero_division=0)
        mcc = matthews_corrcoef(y_test, y_pred)

        accuracy_values.append(accuracy)
        precision_values.append(precision)
        recall_values.append(recall)
        f1_values.append(f1)
        mcc_values.append(mcc)

    if len(imbalance_ratios) < 3:
        continue

    # Store results
    results.append({
        "sample_size": sample_size,
        "imbalance_ratios": imbalance_ratios,
        "shannon_entropies": shannon_entropies,
        "accuracy": accuracy_values,
        "precision": precision_values,
        "recall": recall_values,
        "f1": f1_values,
        "mcc": mcc_values
    })

    x_ir = np.linspace(min(imbalance_ratios), max(imbalance_ratios), 100)
    x_entropy = np.linspace(min(shannon_entropies), max(shannon_entropies), 100)

    plt.figure(figsize=(8, 6))
    plt.plot(imbalance_ratios, accuracy_values, label="Accuracy", color="blue", linestyle='-', marker='o', alpha
    plt.plot(imbalance_ratios, precision_values, label="Precision", color="green", linestyle='-', marker='o', al
    plt.plot(imbalance_ratios, recall_values, label="Recall", color="red", linestyle='-', marker='o', alpha=0.7
    plt.plot(imbalance_ratios, f1_values, label="F1-Score", color="purple", linestyle='-', marker='o', alpha=0.7
    plt.plot(imbalance_ratios, mcc_values, label="MCC", color="orange", linestyle='-', marker='o', alpha=0.7)

    plt.xlabel("Imbalance Ratio")
    plt.ylabel("Metric Value")
    plt.title(f"Metrics vs Imbalance Ratio (Sample Size: {sample_size})")
    plt.legend()
    plt.grid()
    plt.show()

    plt.figure(figsize=(8, 6))
    plt.plot(shannon_entropies, accuracy_values, label="Accuracy", color="blue", linestyle='-', marker='o', alph
    plt.plot(shannon_entropies, precision_values, label="Precision", color="green", linestyle='-', marker='o', a
    plt.plot(shannon_entropies, recall_values, label="Recall", color="red", linestyle='-', marker='o', alpha=0.7
    plt.plot(shannon_entropies, f1_values, label="F1-Score", color="purple", linestyle='-', marker='o', alpha=0
    plt.plot(shannon_entropies, mcc_values, label="MCC", color="orange", linestyle='-', marker='o', alpha=0.7)

    plt.xlabel("Shannon Entropy")
    plt.ylabel("Metric Value")
    plt.title(f"Metrics vs Shannon Entropy (Sample Size: {sample_size})")
    plt.legend()
    plt.grid()
    plt.show()
```

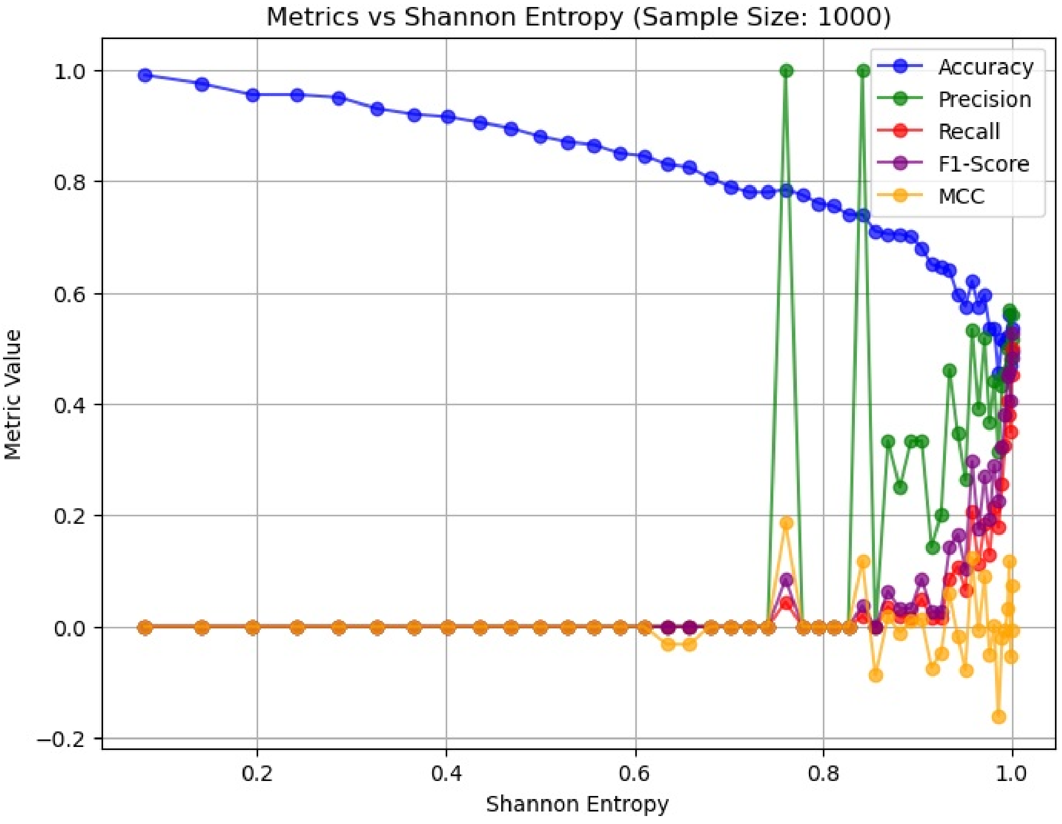

Metrics vs Imbalance Ratio (Sample Size: 100)
Metric Value
Imbalance Ratio
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
10
20
30
40
50
Accuracy
Precision
Recall
F1-Score
MCC

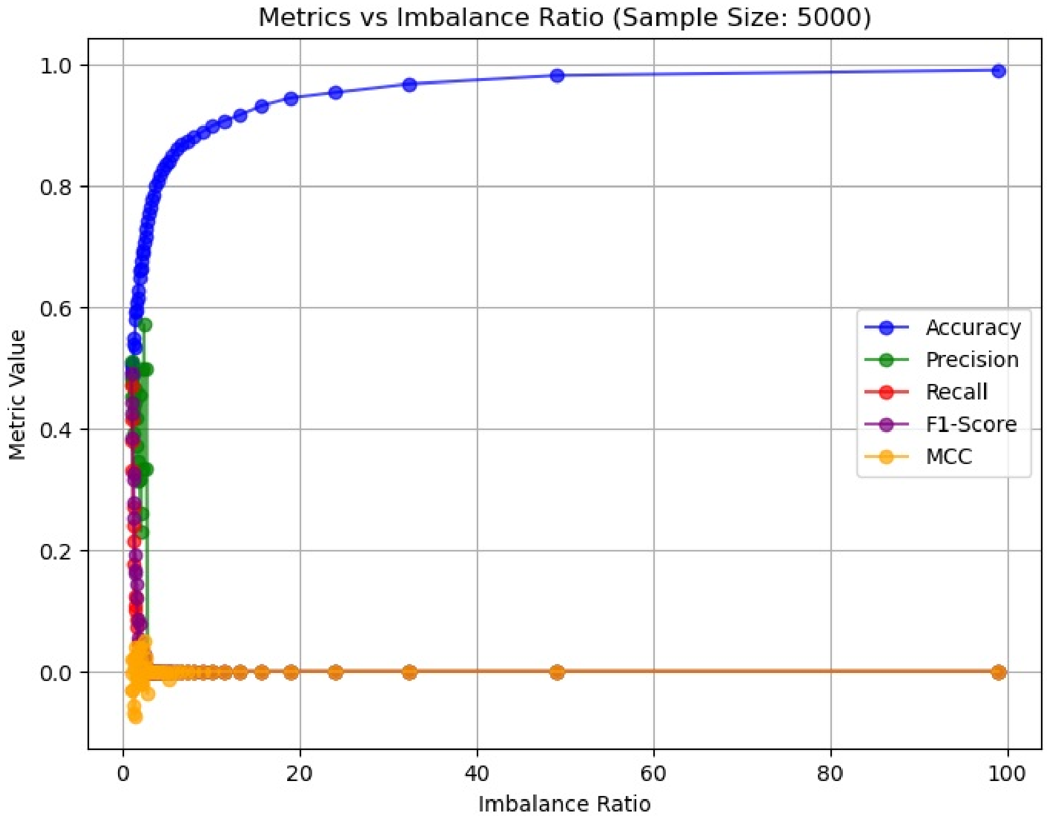

Metrics vs Shannon Entropy (Sample Size: 100)
Metric Value
Shannon Entropy
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0.2
0.4
0.6
0.8
1.0
Accuracy
Precision
Recall
F1-Score
MCC

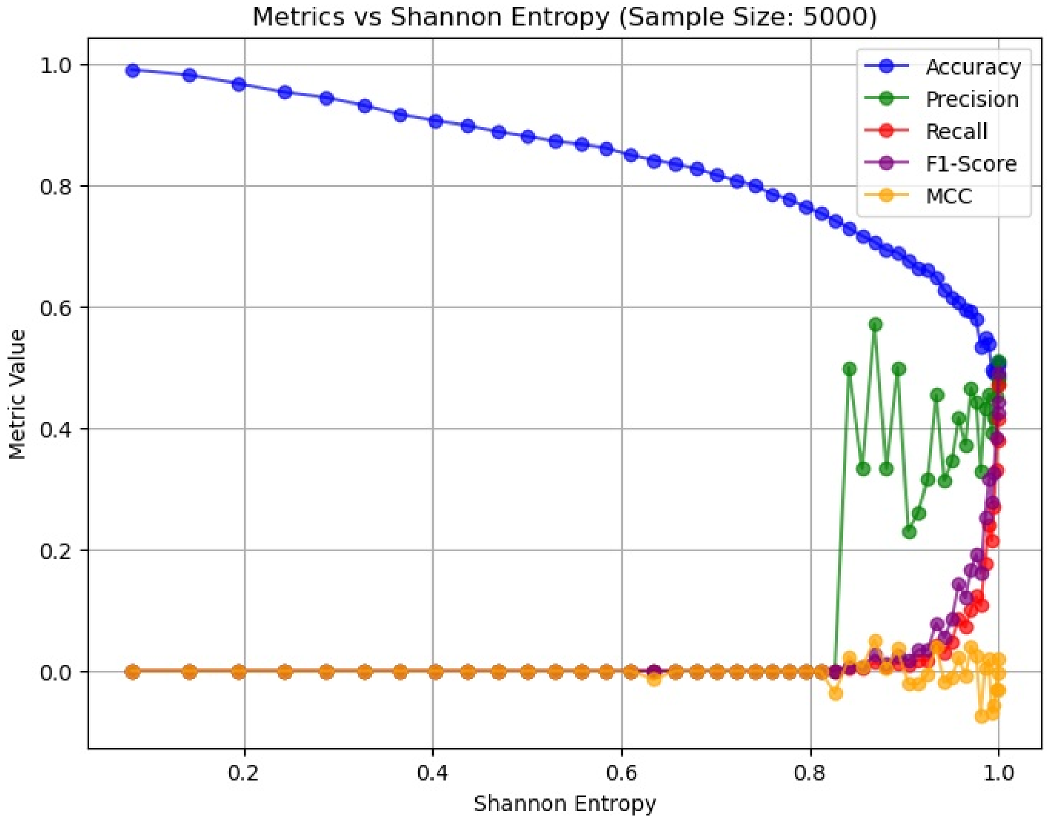

Metrics vs Imbalance Ratio (Sample Size: 500)
Metric Value
Imbalance Ratio
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100
Accuracy
Precision
Recall
F1-Score
MCC

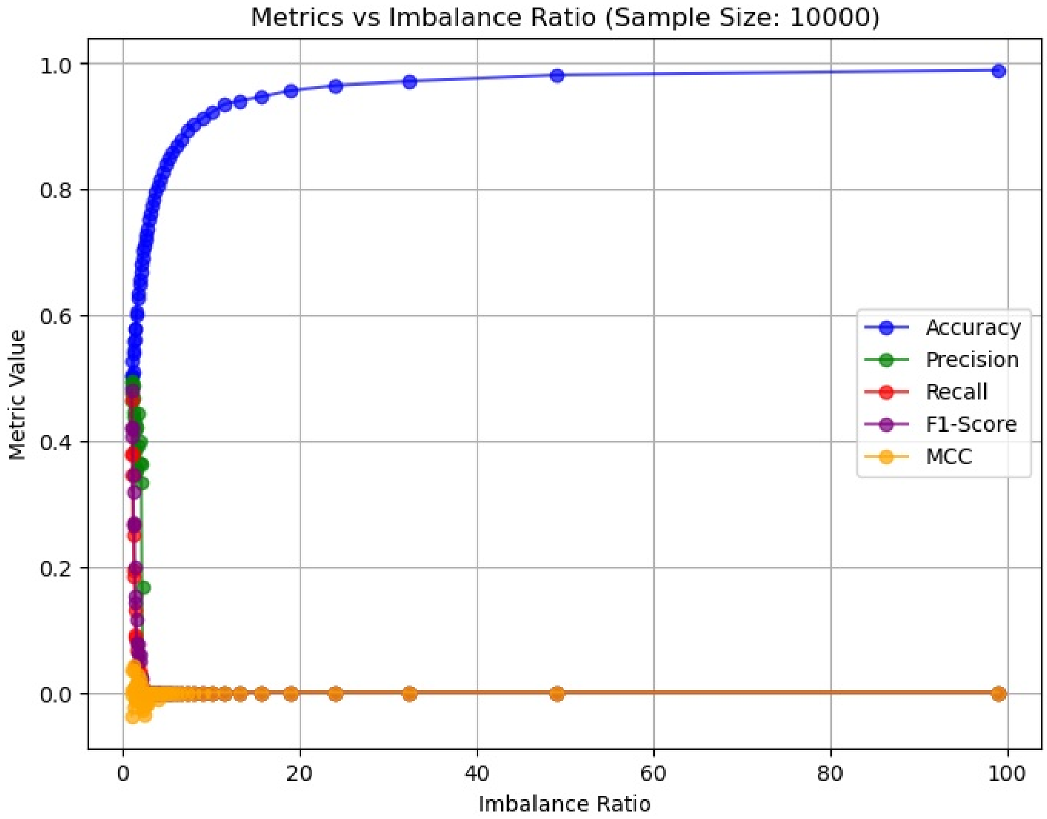

Metrics vs Shannon Entropy (Sample Size: 500)
Metric Value
Shannon Entropy
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
0.2
0.4
0.6
0.8
1.0
Accuracy
Precision
Recall
F1-Score
MCC

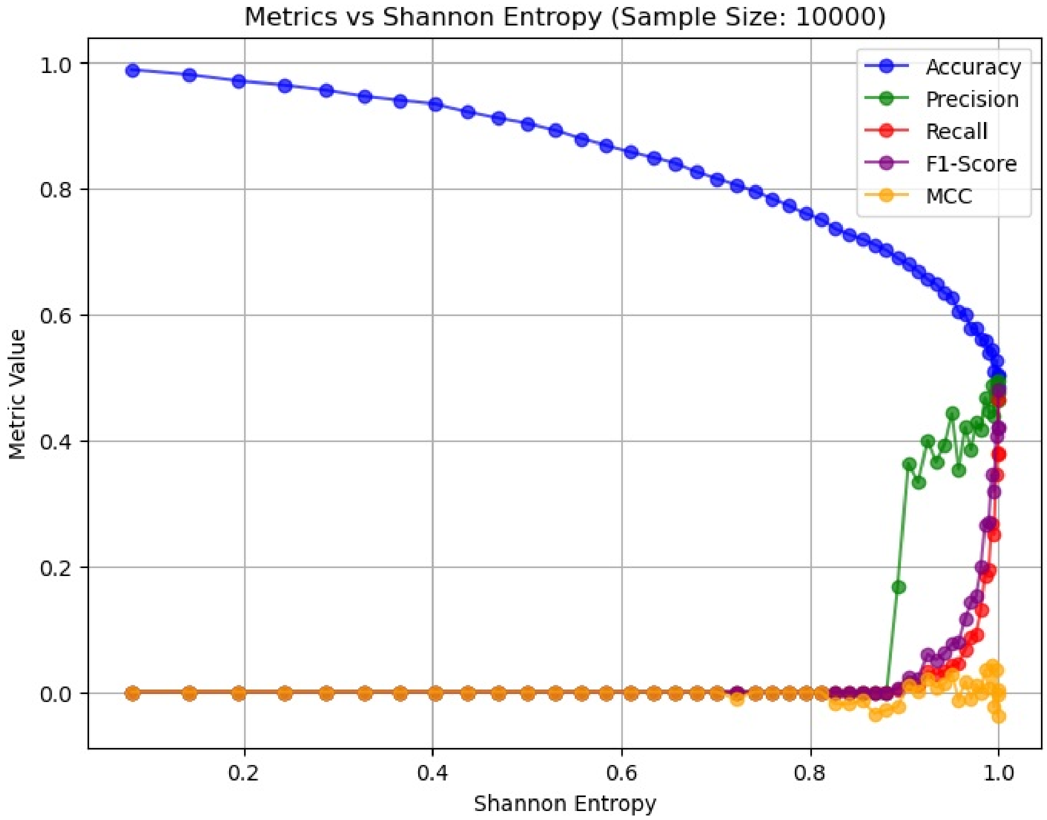

Metrics vs Imbalance Ratio (Sample Size: 1000)
Metric Value
Imbalance Ratio
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
20
40
60
80
100
Accuracy
Precision
Recall
F1-Score
MCC


Metrics vs Shannon Entropy (Sample Size: 1000)
Metric Value
Shannon Entropy
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0.2
0.4
0.6
0.8
1.0
Accuracy
Precision
Recall
F1-Score
MCC


Metrics vs Imbalance Ratio (Sample Size: 5000)
Metric Value
Imbalance Ratio
0
20
40
60
80
100
0.0
0.2
0.4
0.6
0.8
1.0
Accuracy
Precision
Recall
F1-Score
MCC


Metrics vs Shannon Entropy (Sample Size: 5000)
Metric Value
Shannon Entropy
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Accuracy
Precision
Recall
F1-Score
MCC





In [12]:
```
# Create DataFrame for the last dataset
df_last_sample = pd.DataFrame({
    "Minority Proportion (%)": list(range(1,51)),
    "Imbalance Ratio": imbalance_ratios,
    "Shannon's Entropy": shannon_entropies,
    "Accuracy": accuracy_values,
    "Precision": precision_values,
    "Recall": recall_values,
    "F1-Score": f1_values,
    "MCC": mcc_values
})
```

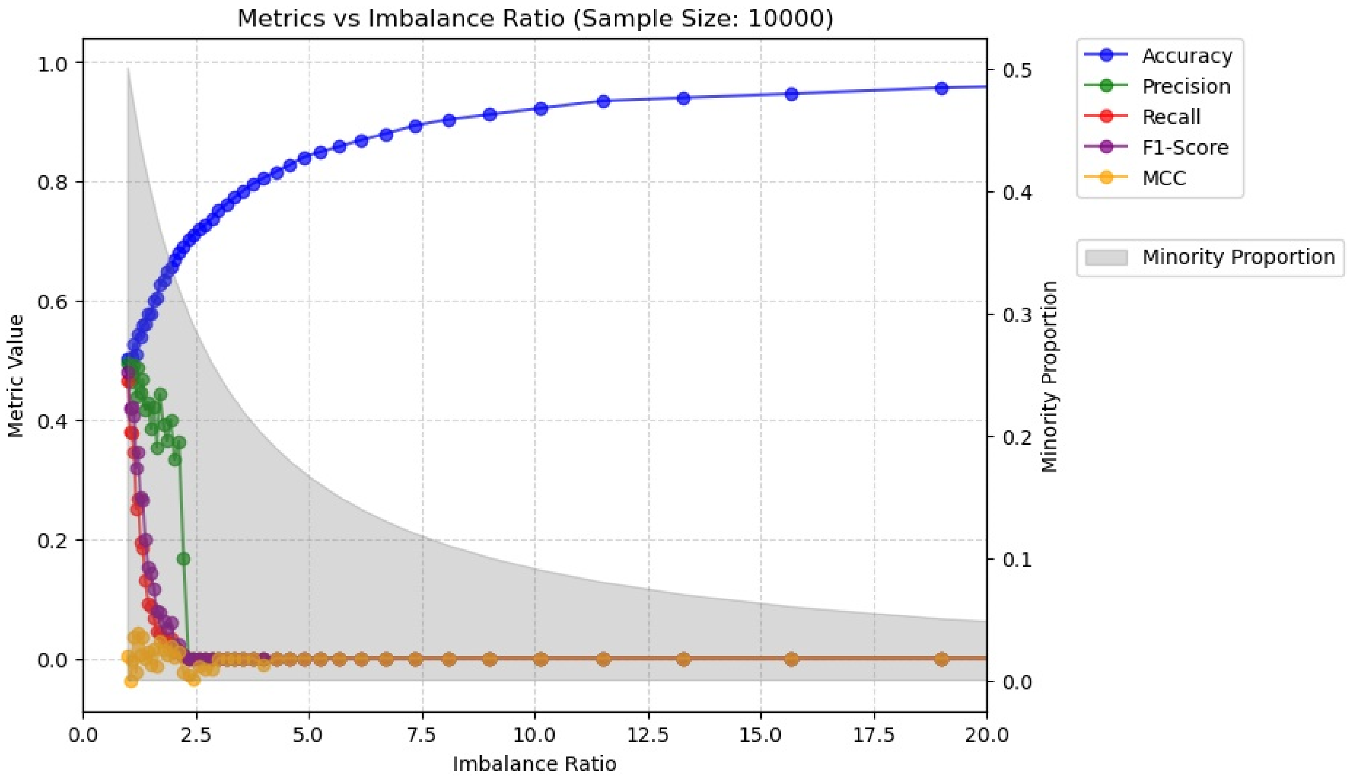
In [13]:
```
display(df_last_sample)
```

	Minority Proportion (%)	Imbalance Ratio	Shannon's Entropy	Accuracy	Precision	Recall	F1-Score	MCC
0	1	99.000000	0.080793	0.9895	0.000000	0.000000	0.000000	0.000000
1	2	49.000000	0.141441	0.9815	0.000000	0.000000	0.000000	0.000000

2	3	32.333333	0.194392	0.9715	0.000000	0.000000	0.000000	0.000000
3	4	24.000000	0.242292	0.9650	0.000000	0.000000	0.000000	0.000000
4	5	19.000000	0.286397	0.9570	0.000000	0.000000	0.000000	0.000000
5	6	15.666667	0.327445	0.9470	0.000000	0.000000	0.000000	0.000000
6	7	13.285714	0.365924	0.9405	0.000000	0.000000	0.000000	0.000000
7	8	11.500000	0.402179	0.9350	0.000000	0.000000	0.000000	0.000000
8	9	10.111111	0.436470	0.9225	0.000000	0.000000	0.000000	0.000000
9	10	9.000000	0.468996	0.9125	0.000000	0.000000	0.000000	0.000000
10	11	8.090909	0.499916	0.9040	0.000000	0.000000	0.000000	0.000000
11	12	7.333333	0.529361	0.8935	0.000000	0.000000	0.000000	0.000000
12	13	6.692308	0.557438	0.8800	0.000000	0.000000	0.000000	0.000000
13	14	6.142857	0.584239	0.8690	0.000000	0.000000	0.000000	0.000000
14	15	5.666667	0.609840	0.8585	0.000000	0.000000	0.000000	0.000000
15	16	5.250000	0.634310	0.8495	0.000000	0.000000	0.000000	0.000000
16	17	4.882353	0.657705	0.8405	0.000000	0.000000	0.000000	0.000000
17	18	4.555556	0.680077	0.8270	0.000000	0.000000	0.000000	0.000000
18	19	4.263158	0.701471	0.8160	0.000000	0.000000	0.000000	0.000000
19	20	4.000000	0.721928	0.8060	0.000000	0.000000	0.000000	-0.010955
20	21	3.761905	0.741483	0.7965	0.000000	0.000000	0.000000	0.000000
21	22	3.545455	0.760168	0.7840	0.000000	0.000000	0.000000	0.000000
22	23	3.347826	0.778011	0.7730	0.000000	0.000000	0.000000	0.000000
23	24	3.166667	0.795040	0.7610	0.000000	0.000000	0.000000	0.000000
24	25	3.000000	0.811278	0.7520	0.000000	0.000000	0.000000	0.000000
25	26	2.846154	0.826746	0.7370	0.000000	0.000000	0.000000	-0.018851
26	27	2.703704	0.841465	0.7275	0.000000	0.000000	0.000000	-0.019315
27	28	2.571429	0.855451	0.7200	0.000000	0.000000	0.000000	-0.013931
28	29	2.448276	0.868721	0.7110	0.000000	0.000000	0.000000	-0.034717
29	30	2.333333	0.881291	0.7025	0.000000	0.000000	0.000000	-0.028993
30	31	2.225806	0.893173	0.6895	0.166667	0.003263	0.006400	-0.023564
31	32	2.125000	0.904381	0.6800	0.363636	0.012618	0.024390	0.010570
32	33	2.030303	0.914926	0.6695	0.333333	0.010703	0.020741	0.001391
33	34	1.941176	0.924819	0.6560	0.400000	0.032496	0.060109	0.021855
34	35	1.857143	0.934068	0.6480	0.365385	0.027536	0.051213	0.007006
35	36	1.777778	0.942683	0.6350	0.393443	0.033473	0.061697	0.012924
36	37	1.702703	0.950672	0.6275	0.442857	0.042062	0.076828	0.029355
37	38	1.631579	0.958042	0.6045	0.354167	0.044561	0.079162	-0.012635
38	39	1.564103	0.964800	0.5995	0.420635	0.067862	0.116869	0.016017
39	40	1.500000	0.970951	0.5780	0.384615	0.087282	0.142276	-0.010578
40	41	1.439024	0.976500	0.5770	0.429379	0.092570	0.152305	0.011958
41	42	1.380952	0.981454	0.5605	0.416031	0.130539	0.198724	-0.001157
42	43	1.325581	0.985815	0.5590	0.467647	0.184884	0.265000	0.034415
43	44	1.272727	0.989588	0.5400	0.445312	0.194761	0.270998	0.006201
44	45	1.222222	0.992774	0.5440	0.487805	0.266667	0.344828	0.043405
45	46	1.173913	0.995378	0.5090	0.438931	0.250545	0.319001	-0.023996
46	47	1.127660	0.997402	0.5280	0.493151	0.346524	0.407035	0.035958
47	48	1.083333	0.998846	0.5045	0.474934	0.377754	0.420807	-0.002449
48	49	1.040816	0.999711	0.4845	0.463750	0.381295	0.418500	-0.037164
49	50	1.000000	1.000000	0.5020	0.495680	0.464575	0.479624	0.003121

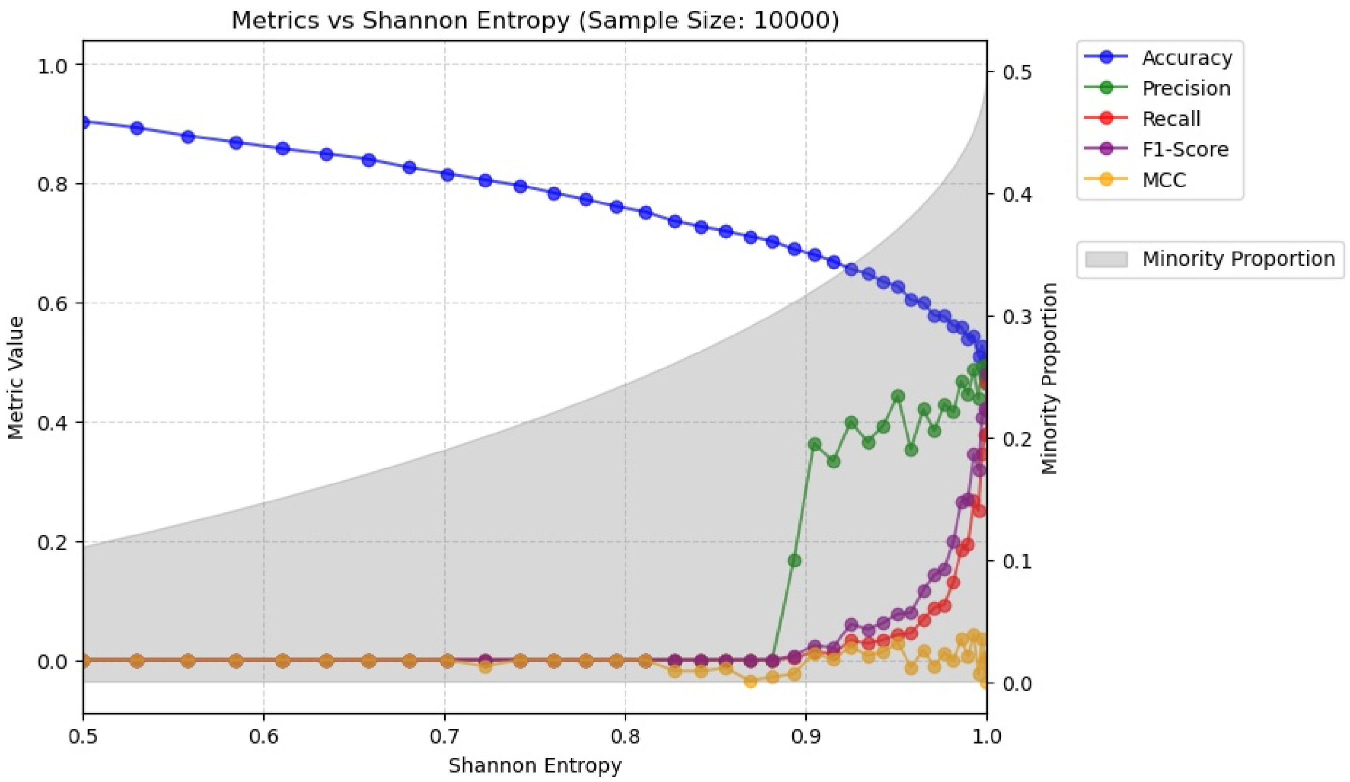
```
In [16]: x = df_last_sample["Imbalance Ratio"]
```

```
fig, ax1 = plt.subplots(figsize=(8, 6))

ax1.plot(x, df_last_sample["Accuracy"], label="Accuracy", color="blue", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, df_last_sample["Precision"], label="Precision", color="green", linestyle='-', marker='o', alpha=0.7
ax1.plot(x, df_last_sample["Recall"], label="Recall", color="red", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, df_last_sample["F1-Score"], label="F1-Score", color="purple", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, df_last_sample["MCC"], label="MCC", color="orange", linestyle='-', marker='o', alpha=0.7)

ax1.set_xlabel("Imbalance Ratio")
ax1.set_ylabel("Metric Value")
ax1.set_xlim(0, 20)
ax1.set_title(f"Metrics vs Imbalance Ratio (Sample Size: {sample_size})")
ax1.legend()
ax1.grid(True, linestyle="--", alpha=0.5)

ax2 = ax1.twinx()
ax2.fill_between(x, 0, df_last_sample["Minority Proportion (%)"]/100, color='gray', alpha=0.3, label="Minority

ax2.set_ylabel("Minority Proportion")

ax1.legend(loc="upper left", bbox_to_anchor=(1.1, 1), borderaxespad=0.)
ax2.legend(loc="upper left", bbox_to_anchor=(1.1, 0.7), borderaxespad=0.)
plt.show()
```



In [20]:

```
x = df_last_sample["Shannon's Entropy"]

fig, ax1 = plt.subplots(figsize=(8, 6))

ax1.plot(x, df_last_sample["Accuracy"], label="Accuracy", color="blue", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, df_last_sample["Precision"], label="Precision", color="green", linestyle='-', marker='o', alpha=0.7
ax1.plot(x, df_last_sample["Recall"], label="Recall", color="red", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, df_last_sample["F1-Score"], label="F1-Score", color="purple", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, df_last_sample["MCC"], label="MCC", color="orange", linestyle='-', marker='o', alpha=0.7)

ax1.set_xlabel("Shannon Entropy")
ax1.set_ylabel("Metric Value")
ax1.set_xlim(0.5, 1)
ax1.set_title(f"Metrics vs Shannon Entropy (Sample Size: {sample_size})")
ax1.legend()
ax1.grid(True, linestyle="--", alpha=0.5)

ax2 = ax1.twinx()
ax2.fill_between(x, 0, df_last_sample["Minority Proportion (%)"]/100, color='gray', alpha=0.3, label="Minority

ax2.set_ylabel("Minority Proportion")

ax1.legend(loc="upper left", bbox_to_anchor=(1.1, 1), borderaxespad=0.)
ax2.legend(loc="upper left", bbox_to_anchor=(1.1, 0.7), borderaxespad=0.)
plt.show()
```



Simulating Various Class Proportions Fitted Against Different Metrics (Credit Card Fraud Data)

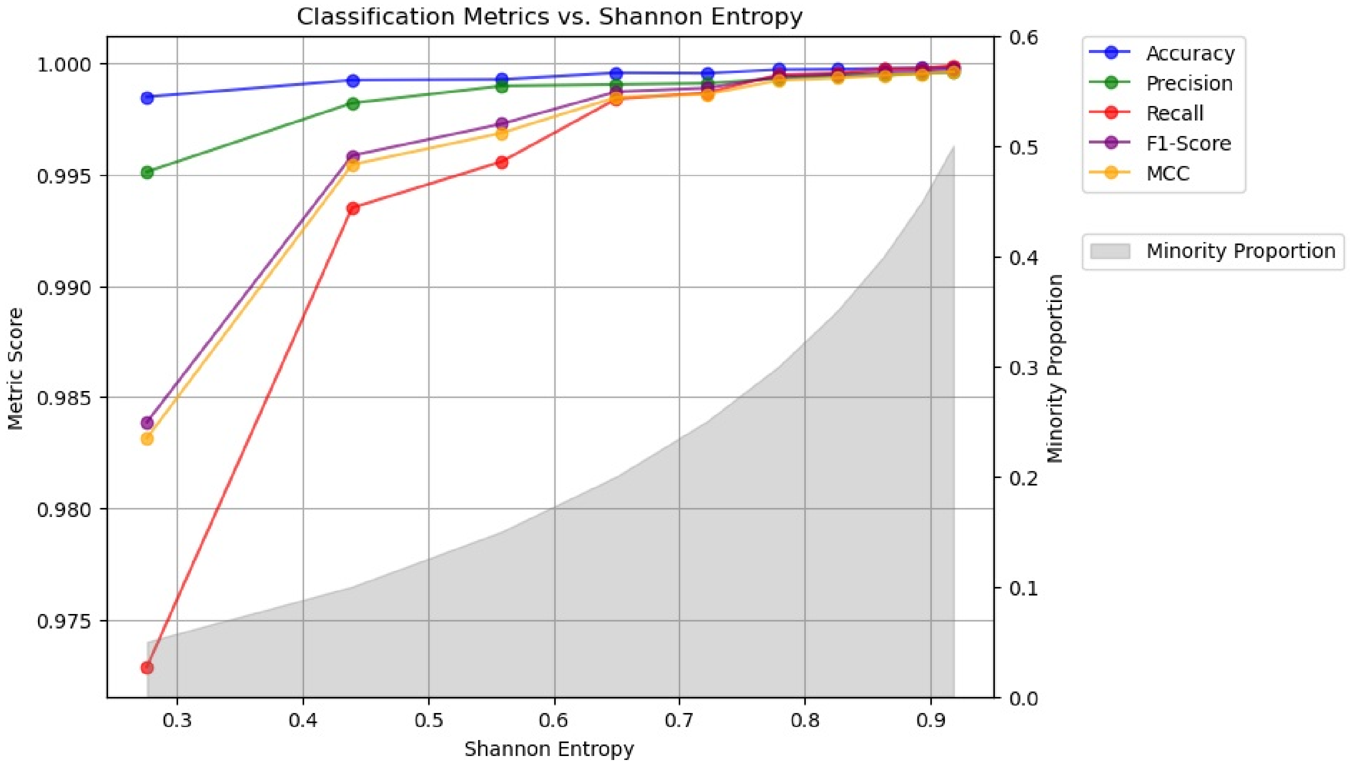
In [23]:

```
data = pd.read_csv('creditcard.csv')

X = data.drop(columns=['Class'])
y = data['Class']

minority_proportions = [0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5]

results = []
imbalance_ratios = []
shannon_entropies = []

for minority_proportion in minority_proportions:
    smote = SMOTE(sampling_strategy=minority_proportion, random_state=42)
    X_resampled, y_resampled = smote.fit_resample(X, y)

    ir, entropy = measure_class_imbalance(y_resampled)
    imbalance_ratios.append(ir)
    shannon_entropies.append(entropy)

    X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.3, random_state=4

    clf = RandomForestClassifier(n_estimators=100, random_state=42, class_weight="balanced")
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
    mcc = matthews_corrcoef(y_test, y_pred)

    results.append({
        'Minority Proportion': minority_proportion,
        'Imbalance Ratio': ir,
        'Shannon Entropy': entropy,
        'Accuracy': accuracy,
        'Precision': precision,
        'Recall': recall,
        'F1-Score': f1,
        'MCC': mcc
    })
```

In [24]:

```
results_df = pd.DataFrame(results)

x = results_df['Shannon Entropy']

fig, ax1 = plt.subplots(figsize=(8, 6))

ax1.plot(x, results_df["Accuracy"], label="Accuracy", color="blue", linestyle='-', marker='o', alpha=0.7)
```

```
ax1.plot(x, results_df["Precision"], label="Precision", color="green", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, results_df["Recall"], label="Recall", color="red", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, results_df["F1-Score"], label="F1-Score", color="purple", linestyle='-', marker='o', alpha=0.7)
ax1.plot(x, results_df["MCC"], label="MCC", color="orange", linestyle='-', marker='o', alpha=0.7)

ax1.set_xlabel('Shannon Entropy')
ax1.set_ylabel('Metric Score')
ax1.set_title('Classification Metrics vs. Shannon Entropy')
ax1.legend()
ax1.grid(True)

ax2 = ax1.twinx()
ax2.fill_between(x, 0, results_df["Minority Proportion"], color='gray', alpha=0.3, label="Minority Proportion")

ax2.set_ylabel("Minority Proportion")
ax2.set_ylim(0, max(minority_proportions) * 1.2)

ax1.legend(loc="upper left", bbox_to_anchor=(1.1, 1), borderaxespad=0.)
ax2.legend(loc="upper left", bbox_to_anchor=(1.1, 0.7), borderaxespad=0.)
plt.show()
```



```
In [25]: display(results_df)
```

	Minority Proportion	Imbalance Ratio	Shannon Entropy	Accuracy	Precision	Recall	F1-Score	MCC
0	0.05	20.001055	0.276185	0.998504	0.995133	0.972877	0.983879	0.983163
1	0.10	10.000176	0.439492	0.999254	0.998221	0.993510	0.995860	0.995453
2	0.15	6.666706	0.558628	0.999286	0.998990	0.995587	0.997285	0.996876
3	0.20	5.000000	0.650022	0.999580	0.999057	0.998410	0.998733	0.998482
4	0.25	4.000042	0.721925	0.999559	0.999107	0.998685	0.998896	0.998621
5	0.30	3.333353	0.779348	0.999729	0.999338	0.999494	0.999416	0.999240
6	0.35	2.857150	0.825626	0.999748	0.999465	0.999566	0.999516	0.999345
7	0.40	2.500000	0.863121	0.999782	0.999474	0.999766	0.999620	0.999468
8	0.45	2.222235	0.893570	0.999798	0.999557	0.999792	0.999674	0.999528
9	0.50	2.000007	0.918295	0.999836	0.999602	0.999906	0.999754	0.999631

```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js