Final Project Report

1. Introduction (880/1000)

In the past two decades, the growth of academic publishing in computer science and related fields has been exponential. Digital repositories and online journals make thousands of new research outputs available every year. While this amount has created unimagined opportunities for scholars to access knowledge, it has also generated a pressing challenge: how to efficiently navigate and classify meaning from such large volumes of text. Traditional approaches to review literature and topic identification are increasingly unusable, not only because of the scale of available information but also due to the human bias and the inability to have the same output found in human documentation. This context motivates the development of automated systems for research paper classification, methodology detection, and metadata analysis.

The project presented in this report seeks to address this challenge by designing and implementing a pipeline capable of classifying research publications into broad topic fields and identifying the methodology of each paper. Specifically, the system ingests metadata drawn from the International Conference on Machine Learning and Applications or (ICMLA) dataset for short, which includes titles, abstracts, keywords, and session information. Through a combination of text mining techniques, statistical modeling, multiple pipelines and rule based heuristics, the system attempts to reduce manual workload for researchers and provide a reliable tool for literature triage. The core contributions include:

1- A mapping of heterogeneous session labels into consistent, broader field categories to mitigate sparsity.

2- A supervised text classification model, built using Term Frequency–Inverse Document Frequency (TF-IDF) features and a Multinomial Naïve Bayes (MNB) classifier, to predict the field of each paper.

3- A lightweight rule-based methodology detector that assigns methodological labels such as experimental, theoretical, simulation-based, survey, or case study according to the linguistic cues present in abstracts and titles.

The significance of this project lies in its balance between sophistication and accessibility. While the machine learning research community has increasingly leaned towards transformer based architectures such as BERT (Devlin, Chang, Lee, & Toutanova, 2019) and SciBERT (Beltagy, Lo, & Cohan, 2019), these models typically require large-scale training data and substantial computational resources. In contrast, this project deliberately emphasizes lightweight models such as TF-IDF and Naive Bayes ; which , although comparatively modest in performance, offer interpretability and suitability for smaller datasets. In doing so, it's made to the needs of researchers and students who may not have access to advanced infrastructure but still require effective resources for metadata analysis.

The project also demonstrates originality by combining two related but distinct tasks which are field classification and methodology detection and they are made into a single coherent framework. Field classification supports topic organization, which is essential for literature review and research trend analysis. Methodology detection while they facilitate meta research by enabling scholars to quickly locate experimental papers for replication studies and surveys for background synthesis or simulations for validation. By combining all of these tasks the system adds to the wider range of digital scholarship and the future of automated academic research .

This work is placed within the context of the University of London's final year project framework. Following the Template Project 12: Identifying research methodologies that are used in research in the computing disciplines, the project adopts a research driven approach to text analytics with a focus on practical implementation and evaluation. The primary aims are to:

1- Develop a robust end-to-end pipeline for automated classification and methodology detection.

2- Evaluate the effectiveness of lightweight machine learning algorithms on real-world metadata.

3- Critically assess the strengths, limitations, and possible extensions of such a system in the context of academic research.

In order to achieve these aims, the project has been structured into six chapters. The present chapter has introduced the motivation, aims, and contributions of the work. Chapter Two provides a comprehensive Literature Review, situating the project in relation to existing work in text classification, document representation, and research methodology detection. Chapter Three details the Design, outlining architectural choices, feature engineering decisions, and theoretical justifications. Chapter Four focuses on the Implementation, including a walkthrough of the Python code and explanations of the algorithms used. Chapter Five presents the Evaluation, reporting the outcomes of model testing, the performance of methodology detection, and critical reflections on successes and shortcomings. Finally, Chapter Six offers the Conclusion, summarizing the findings, identifying limitations, and suggesting avenues for future research.

By adopting this structure, the report demonstrates not only the technical feasibility of the proposed system but also its critical positioning within the broader field of natural language processing (NLP) and digital scholarship. The value of the project is therefore twofold: it delivers a working prototype capable of classifying and annotating research metadata, and it provides a critical analysis that reflects on methodological choices, domain challenges, and the trajectory of future developments.

In summary, this project addresses a pressing issue in modern academic practice: the difficulty of navigating vast volumes of scientific literature. By leveraging TF-IDF and Naïve Bayes for field classification, and rule-based heuristics for methodology detection, the system offers a lightweight yet functional tool for researchers. Although modest compared to state-of-the-art deep learning models, the pipeline is designed to be transparent, reproducible, and adaptable. Its success will not be measured solely by accuracy metrics but by its ability to illuminate the possibilities and limitations of automated metadata analysis in scholarly ecosystems.

2. Literature Review (1397/2500)

The classification of research documents and the detection of methodological orientation have long been topics of interest within the field of natural language processing (NLP). As the volume of scholarly publications continues to rise, automated approaches to metadata analysis are increasingly necessary to support researchers in navigating complex digital ecosystems. This chapter situates the present project within the broader scholarly discourse, examining the evolution of document representation methods, the use of classical machine learning algorithms for classification, the emergence of transformer-based models, and existing work on research methodology detection. By synthesizing and critically evaluating prior studies, the review establishes the conceptual foundation for the design and implementation choices made in this project.

2.1 Document Representation in Text Classification

One of the most fundamental questions in NLP is how to represent text for computational analysis. Early approaches to document representation, such as the bag-of-words (BoW) model, treat a text as an unordered collection of tokens, disregarding syntax and word order (Salton, Wong, & Yang, 1975). While simple and effective in certain contexts, BoW suffers from the curse of dimensionality and fails to capture semantic or contextual meaning.

To address these limitations, researchers introduced the Term Frequency–Inverse Document Frequency (TF-IDF) scheme. TF-IDF adjusts raw term frequencies by penalizing common words and rewarding distinctive ones, thereby enhancing discriminative power (Manning, Raghavan, & Schütze, 2008). TF-IDF has become a cornerstone of classical text classification pipelines due to its interpretability and efficiency. Its mathematical formulation can be expressed as:

$$tfidf(t,d) = tf(t,d) \times log(\frac{N}{df(t)})$$

Where tfidf(t,d) is the frequency of term t in document d . N is the total number of documents, and df(t) is the number of documents containing the term t. This weighting highlights terms that are both frequent in a given document and rare across the corpus, thereby improving separability between classes.

Beyond TF-IDF, other representation techniques such as word embeddings (Mikolov et al., 2013) and document embeddings (Le & Mikolov, 2014) have been widely explored. Word2Vec and Doc2Vec models capture semantic similarity by mapping words or documents into dense vector spaces based on distributional properties. However, while embeddings capture richer semantics than TF-IDF, they often require larger datasets for training and may be less transparent. In contexts where interpretability and computational efficiency are paramount, TF-IDF remains a competitive choice, especially when combined with lightweight classifiers

Classical machine learning algorithms form the backbone of many text classification systems. Among them, Naïve Bayes classifiers have been particularly influential due to their simplicity and effectiveness in high-dimensional spaces. The Multinomial Naïve Bayes

(MNB) variant is especially suitable for text data, as it models word frequencies under the assumption of conditional independence (McCallum & Nigam, 1998). Its decision rule is derived from Bayes' theorem:

$$P(c|d) \propto P(c) \prod_{t \in d} P(t|c)^{tf(t,d)}$$

Where P(c) is the prior probability of class c , and P(t|c) is the likelihood of observing term t given class c. Despite the unrealistic independence assumption, MNB often performs surprisingly well in practice, particularly for short documents or sparse feature spaces.

Other classical algorithms frequently used in text classification include Logistic Regression, Support Vector Machines (SVMs), and Decision Trees. SVMs, in particular, have demonstrated strong performance in high-dimensional spaces by maximizing the margin between classes (Joachims, 1998). However, their computational complexity can be prohibitive for very large corpora, and interpretability may be limited compared to Naïve Bayes. Logistic Regression offers a probabilistic framework with interpretable coefficients but often requires regularization to avoid overfitting in sparse feature spaces.

Recent surveys (Kowsari et al., 2019; Minaee et al., 2021) highlight that classical methods remain relevant in scenarios where datasets are small, interpretability is valued, or computational resources are constrained. This aligns with the motivations of the present project, which deliberately adopts TF-IDF and Naïve Bayes to create a lightweight yet functional pipeline.

2.3 Deep Learning and Transformer Models

The rise of deep learning has transformed NLP. Early neural approaches, such as Convolutional Neural Networks (CNNs) for sentence classification (Kim, 2014) and Recurrent Neural Networks (RNNs) for sequence modeling, demonstrated that non-linear architectures could outperform traditional methods on tasks like sentiment analysis and document categorization. However, these models were often limited by vanishing gradients, difficulty in capturing long-term dependencies, and significant training costs.

A major breakthrough came with the introduction of transformer architectures (Vaswani et al., 2017), which use self-attention mechanisms to capture long-range dependencies without sequential processing. Building on this architecture, models such as BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) revolutionized text classification by pretraining on massive corpora and fine-tuning on downstream tasks. BERT-based models achieve state-of-the-art performance in numerous NLP benchmarks, including question answering, sentiment analysis, and topic classification.

For scientific domains specifically, SciBERT (Beltagy, Lo, & Cohan, 2019) was trained on a large corpus of scientific articles and demonstrated superior performance on domain-specific tasks. These advances underscore the power of contextual embeddings and pretrained transformers. Nevertheless, their practical adoption remains limited by high computational costs, large memory requirements, and the need for labeled training data for fine-tuning.

In the context of this project, transformers represent a valuable point of comparison. While they clearly outperform TF-IDF and Naïve Bayes in many settings, the resource-light approach adopted here is motivated by practical constraints and pedagogical goals. By focusing on classical methods, the project emphasizes interpretability, accessibility, and replicability, while acknowledging that transformer-based systems represent a promising avenue for future work.

2.4 Methodology Detection in Research Papers

Beyond topical classification, detecting the methodological orientation of research papers has gained increasing attention. Methodology detection supports meta-analysis, systematic reviews, and evidence-based practice by allowing researchers to filter studies according to research design.

One common approach is rule-based methodology detection, which relies on handcrafted keywords and patterns. For example, phrases such as “we conducted an experiment” or “simulation results” strongly indicate experimental and simulation methodologies, respectively. Rule-based systems are highly interpretable and require little data, but they suffer from limited coverage and vulnerability to linguistic variability (Oates, 2006).

Supervised machine learning approaches have also been explored. Jha et al. (2017) used SVM classifiers to distinguish between methodological categories in biomedical literature, while Fisas, Casamayor, and Gilabert (2016) developed annotation schemes for rhetorical and methodological classification. More recently, transformer-based classifiers such as BERT have been adapted for methodology detection, showing improved generalization (Cohan et al., 2019).

Nevertheless, methodological classification remains challenging due to overlapping categories, domain-specific jargon, and the lack of large annotated datasets. In low-resource contexts, rule-based systems remain useful as baseline models, particularly when designed with domain knowledge. This project adopts a keyword-driven methodology detector precisely for this reason, while acknowledging that machine learning and transformer-based approaches could improve accuracy if more data were available.

2.5 Applications of Research Paper Classification

Automated classification of research papers has numerous applications in academia and industry. Digital libraries such as IEEE Xplore and ACM Digital Library already use metadata-driven search and categorization, but these systems often rely on manual curation. Automated pipelines could reduce costs and improve scalability.

In educational contexts, students conducting literature reviews could benefit from systems that pre-sort papers by field and methodology, reducing the cognitive load associated with filtering large datasets. Similarly, conference organizers might use automated systems to group submissions into sessions, while funding bodies could classify proposals by methodological orientation to ensure balanced portfolios.

From a meta-research perspective, automated classification also enables the study of trends in research methodology over time. For instance, one could investigate whether experimental studies have increased relative to theoretical contributions in machine learning conferences, or whether surveys are more prevalent in particular subfields. This project contributes to this emerging area by providing both field classification and methodology detection capabilities.

2.6 Critical Evaluation of the Literature

The review highlights several key insights. First, TF-IDF and Naïve Bayes remain strong baselines due to their simplicity and interpretability, especially in resource-constrained contexts. Second, while deep learning and transformer models offer superior performance, their adoption is limited by cost and complexity. Third, methodology detection is an underexplored but increasingly important area, with rule-based systems offering transparency and supervised approaches showing promise when annotated datasets are available.

The literature also reveals several gaps that this project seeks to address. Most existing work focuses exclusively on topical classification, with relatively little attention paid to methodology detection. Moreover, few studies combine the two tasks into a single pipeline, despite their complementary nature. By unifying field classification with methodology detection, the present project offers a novel contribution that extends existing approaches while remaining mindful of practical constraints.

3. Design (1205/2000)

The design of this project is grounded in the principle of building a modular, interpretable, and computationally efficient pipeline for the classification of research publications. This chapter provides a detailed account of the system architecture, the rationale behind feature engineering choices, the design of field mapping rules, and the methodology detection strategy. The design emphasizes transparency, reproducibility, and scalability, ensuring that each component of the pipeline can be evaluated, modified, or replaced without disrupting the entire system.

The pipeline is structured into six major components:

1. **Data Ingestion**: The system ingests metadata from a CSV file containing research paper information such as titles, abstracts, keywords, and session labels. To account for potential encoding inconsistencies across data sources, the ingestion module attempts multiple encoding schemes (UTF-8, Latin-1) and normalizes column names.

2. **Preprocessing**: Metadata fields (title, abstract, keywords) are aggregated into a single text field. This ensures that the classifier has access to the full range of descriptive content, while reducing the risk of missing informative signals located in non-title fields. Noise such as excessive whitespace is removed, and all text is standardized to lowercase.

3. **Session-to-Field Mapping**: Raw session labels in the dataset are heterogeneous and often too fine-grained for robust classification. A mapping strategy consolidates session

names into broader field categories such as *Neural Networks / Deep Learning*, *Ensemble Methods*, *Bioinformatics*, *Security*, *Time Series*, and *Information Retrieval / Topic Models*. This mapping mitigates sparsity by increasing the number of samples per class, which is essential for reliable training.

4. **Feature Extraction (TF-IDF)**: The aggregated text is vectorized using the TF-IDF scheme. The design sets a cap of 2000 features, applies English stop-word removal, and extracts both unigrams and bigrams. These design choices balance expressiveness with efficiency, ensuring that the classifier captures meaningful phrases (e.g., “neural network”) without overwhelming computational resources.

5. **Classification (Multinomial Naïve Bayes)**: The classifier is trained on TF-IDF features to predict the mapped field of each paper. Multinomial Naïve Bayes was selected for its efficiency, robustness in sparse feature spaces, and interpretability.

6. **Methodology Detection (Rule-Based)**: A keyword-driven approach assigns methodological labels to each paper. For example, the presence of terms such as “experiment” or “evaluation” signals an experimental methodology, while terms such as “simulation” or “synthetic data” suggest a simulation-based study. The detector is designed with explicit keyword lists for categories including *Experimental*, *Theoretical*, *Simulation*, *Survey*, and *Case Study*.

Each of these modules interacts in a pipeline fashion, with outputs from one component feeding directly into the next. The modularity ensures that individual components can be swapped with more advanced alternatives in the future, such as replacing Naïve Bayes with a transformer-based classifier.

3.2 Design Rationale

The overarching design rationale reflects a balance between **practicality** and **academic rigor**. Three guiding principles shaped the system:

- **Interpretability**: The chosen algorithms (TF-IDF, Naïve Bayes, keyword-based rules) allow for clear explanations of why particular classifications are made. Interpretability is vital in academic contexts, where researchers must understand and justify methodological decisions.

- **Reproducibility**: The pipeline was designed to be reproducible with limited computational resources. This ensures that the project can be replicated by students or researchers without access to advanced hardware.

- **Scalability**: While the immediate project works on a conference dataset, the architecture allows for scaling to larger corpora, such as the ACM Digital Library or arXiv. By abstracting ingestion, preprocessing, and classification into separate modules, the system could be extended to handle millions of documents with relatively minor modifications.

3.3 Field Mapping Strategy

A central design decision was to address class sparsity by consolidating raw session labels into broader categories. Without this step, the classifier would face a highly imbalanced dataset with many low-support classes, leading to poor generalization.

The mapping strategy relied on linguistic heuristics. For instance, any session containing the substring “neural” or “deep” was mapped to *Neural Networks / Deep Learning*, while sessions with “bio” were mapped to *Bioinformatics*. The rules were carefully designed to be mutually exclusive and collectively exhaustive, ensuring that all sessions were assigned to a meaningful category.

This deterministic mapping increases the average number of documents per class, improving the stability of the classifier. It also provides a practical taxonomy that aligns with common research themes in machine learning. While rule-based, the mapping could be refined in future iterations using clustering algorithms or embedding-based similarity measures.

3.4 Feature Engineering

The decision to use TF-IDF for feature extraction reflects both theoretical and practical considerations. TF-IDF emphasizes discriminative terms while controlling for document frequency, making it well-suited to the academic domain, where certain technical terms (e.g., “algorithm”, “model”) appear across nearly all papers.

To enhance the expressiveness of features, the system includes both unigrams and bigrams. This allows the model to capture meaningful phrases such as “support vector” or “neural network”, which are more informative than individual tokens. The feature space was limited to 2000 terms to control dimensionality, reduce overfitting, and ensure rapid training.

Alternative feature extraction methods, such as word embeddings or transformer-based contextual embeddings, were considered but deemed impractical within the project’s resource constraints. However, the modular design of the system allows for easy replacement of TF-IDF with more advanced embeddings in the future.

3.5 Classification Model

The **Multinomial Naïve Bayes** classifier was chosen for several reasons:

- **Efficiency**: MNB has linear time complexity with respect to the number of features, making it well-suited for large-scale document collections.

- **Robustness**: Despite its simplifying assumption of conditional independence, MNB performs competitively in text classification tasks.

- **Interpretability**: The probabilistic outputs of Naïve Bayes can be directly interpreted as posterior probabilities, providing insight into model confidence.

The classifier is trained on an 80-20 stratified train-test split to ensure that class proportions are preserved. Performance metrics include accuracy, precision, recall, and macro-averaged F1, which provides a balanced measure across imbalanced classes

3.6 Methodology Detection

Methodology detection was designed as a rule-based module due to the lack of annotated training data and the need for interpretability. Keywords were curated from domain knowledge and sample abstracts. For example:

- *Experimental*: “experiment”, “evaluation”, “tested”, “measured”

- *Survey*: “survey”, “review”, “overview”

- *Theoretical*: “theoretical”, “proof”, “derivation”

- *Simulation*: “simulation”, “simulated results”, “synthetic data”

- *Case Study*: “case study”, “case report”

When multiple keywords are present, a priority order resolves conflicts (e.g., Experimental > Theoretical > Simulation > Survey > Case Study). While simplistic, this approach provides a transparent baseline that can later be replaced with machine learning models trained on annotated corpora.

3.7 Integration of Graphical Analysis

The design incorporates visualization as an integral part of evaluation. Three figures were planned:

- **Figure 1**: Field distribution (bar chart of mapped categories).

- **Figure 2**: Confusion matrix of classification results.

- **Figure 3**: Methodology detection counts.

These figures not only validate the pipeline but also enhance interpretability by providing intuitive summaries of model performance and dataset characteristics.

3.8 Limitations of the Design

While the design is robust and pragmatic, it is important to acknowledge its limitations:

- **Keyword Dependence**: The methodology detector relies heavily on keyword presence, which may miss implicit methodological cues.

- **Rule-Based Mapping**: The field mapping rules may misclassify ambiguous sessions or fail to capture emerging fields.

- **Simplified Classifier**: Naïve Bayes may underperform on nuanced distinctions compared to transformers.

Nonetheless, these limitations are consistent with the project's emphasis on lightweight, reproducible solutions. They also point toward natural extensions for future research, such as integrating embedding-based classifiers or supervised methodology detection.

4. Implementation (1082 / 2500)

The design outlined in the previous chapter was realized through a Python-based implementation, combining established machine learning libraries with customized rule-based modules. The implementation phase translated theoretical concepts into a functional pipeline capable of ingesting, processing, classifying, and annotating research metadata. This chapter provides a detailed walkthrough of the implementation, including the tools and libraries employed, the structure of the codebase, and the integration of visualization outputs. Representative code snippets are provided to illustrate key operations, while the overall narrative emphasizes how each design decision was instantiated in practice.

4.1 Development Environment

The implementation was developed using **Python 3.10**, chosen for its extensive ecosystem of machine learning and natural language processing libraries. The primary libraries included:

- **pandas** for data manipulation and cleaning.
- **scikit-learn** for feature extraction (TF-IDF), model training, and evaluation.
- **matplotlib** for data visualization, including bar charts and confusion matrices.
- **joblib** for saving and reloading trained models and vectorizers.
- **reportlab** and **python-docx** for report generation (supporting reproducibility of results in both PDF and DOCX formats).

The choice of Python reflects its widespread adoption in both academia and industry, ensuring that the pipeline can be easily reproduced and extended by others.

4.2 Data Ingestion and Cleaning

The first step in implementation was to ingest the ICMLA dataset, stored in CSV format. To handle potential encoding inconsistencies across operating systems, the code attempted to read the file in **UTF-8**, falling back to **Latin-1** if errors were encountered. Column names were standardized by stripping whitespace and converting to lowercase.

Representative code snippet:

//

try:

df = pd.read_csv("icmla_metadata.csv", encoding="utf-8", low_memory=False)

except:

df = pd.read_csv("icmla_metadata.csv", encoding="latin-1", low_memory=False)

df.columns = [c.strip() for c in df.columns]

//

Once loaded, relevant columns — titles, abstracts, and keywords — were aggregated into a single text field. This ensured that the classifier had access to the maximum amount of

descriptive metadata. Missing values were handled gracefully by treating nulls as empty strings.

4.3 Preprocessing

Preprocessing aimed to prepare the aggregated text for vectorization. The pipeline concatenated titles, keywords, and abstracts, then standardized whitespace and converted text to lowercase.

//

```
def combine_text(row):
    parts = []
    for c in ["title", "keywords", "abstract"]:
        v = str(row[c]) if pd.notnull(row[c]) else ""
        parts.append(v)
    combined = " ".join(parts)
    combined = re.sub(r"\s+", " ", combined)
    return combined.strip()
df["text"] = df.apply(combine_text, axis=1)
```

//

This preprocessing step produced a corpus of documents suitable for TF-IDF vectorization.

4.4 Field Mapping

The dataset's original session labels were highly specific (e.g., "Neural Networks Applications" or "Bioinformatics in Systems Biology"). To address sparsity, session names were mapped into broader categories using deterministic rules.

Code snippet:

//

```
def map_session(s):
    s_low = s.lower()
```

```
    if "deep" in s_low or "neural" in s_low:
        return "Neural Networks / Deep Learning"
    if "ensemble" in s_low:
        return "Ensemble Methods"
    if "bio" in s_low:
        return "Bioinformatics"
    if "security" in s_low or "cyber" in s_low:
        return "Security"
    if "time series" in s_low or "temporal" in s_low:
        return "Time Series"
    if "retrieval" in s_low or "topic" in s_low or "information" in s_low:
        return "Information Retrieval / Topic Models"
    if "energy" in s_low or "engineering" in s_low:
        return "Applications / Engineering"
    return "Machine Learning General"
//
```

This mapping step consolidated dozens of heterogeneous labels into fewer, higher-support categories. The results were later visualized in **Figure 1 (Field Distribution)**, a bar chart showing the relative frequency of each mapped field.

4.5 Feature Extraction: TF-IDF

With text aggregated and sessions mapped, the next step was to convert documents into numeric vectors using TF-IDF. The vectorizer was configured with:

- “max_features=2000” to cap dimensionality.
- stop_words="english" to remove common but uninformative words.
- “ngram_range=(1,2)” to capture both unigrams and bigrams.

Code snippet:

//

```
vectorizer = TfidfVectorizer(
   max_features=2000,
   stop_words="english",
   ngram_range=(1, 2)
)
X = vectorizer.fit_transform(df["text"].fillna(""))
```

//

4.6 Classification: Multinomial Naïve Bayes

The classifier was implemented using **Multinomial Naïve Bayes (MNB)**, a standard choice for text classification. Labels were encoded using scikit-learn's LabelEncoder , ensuring consistency between training and testing phases.

An **80-20 stratified train-test split** was applied to preserve class proportions. The classifier was trained and evaluated on this split.

Code snippet:

//

```
le = LabelEncoder()
y = le.fit_transform(df["field"])

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

clf = MultinomialNB()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

//

Performance metrics, including accuracy and macro-averaged F1, were computed using scikit-learn. The confusion matrix was visualized in **Figure 2**, providing insight into per-class performance.

4.7 Methodology Detection

The methodology detector was implemented as a simple rule-based system. Keywords associated with methodological categories were curated from domain knowledge and sample abstracts. The detector scanned each aggregated text field, assigning a methodology based on the first match found.

Code snippet:

//

```
method_keywords = {
   "Experimental": ["experiment", "evaluation", "tested", "measured"],
   "Survey": ["survey", "review", "overview"],
   "Theoretical": ["theoretical", "proof", "derivation"],
   "Simulation": ["simulation", "synthetic data"],
   "Case Study": ["case study", "case report"]
}


def detect_method(text):
   t = text.lower()
   for k, kws in method_keywords.items():
      for kw in kws:
         if kw in t:
            return k
   return "Unknown"
df["method_pred"] = df["text"].apply(detect_method)
```

//

The distribution of detected methodologies was plotted in **Figure 3 (Methodology Detection Counts)**.

4.8 Model Serialization and Reuse

To ensure reproducibility, the trained classifier, vectorizer, and label encoder were saved using “joblib”. This allows others to reuse the trained models without retraining from scratch.

//

joblib.dump(clf, "nb_model.joblib")

joblib.dump(vectorizer, "tfidf_vectorizer.joblib")

joblib.dump(le, "label_encoder.joblib")

//

This design choice reflects best practices in reproducible research.

4.9 Visualization

Three visualizations were central to the project:

- **Figure 1: Field Distribution (Mapped)** – a bar chart of field counts.
- **Figure 2: Confusion Matrix** – a heatmap displaying true vs. predicted labels.
- **Figure 3: Methodology Detection Counts** – a bar chart showing the frequency of methodological categories.

These figures not only validated the functioning of the pipeline but also provided intuitive insights for end-users.

4.10 Integration into Report Generation

An additional contribution of the implementation was the integration of report generation using **ReportLab** and **python-docx**. This allowed the automated production of both PDF and DOCX reports, complete with figures and text. Although not strictly required for classification, this feature demonstrated the potential of the pipeline to support digital scholarship workflows.

4.11 Reflections on Implementation

The implementation demonstrated the feasibility of constructing a lightweight, reproducible pipeline. The modular structure allowed each component to be tested independently, while the integration of visualization enhanced interpretability. Nevertheless, challenges were encountered, including:

- **Encoding issues** during CSV ingestion, requiring fallback strategies.
- **Ambiguity in session labels**, which sometimes resisted deterministic mapping.
- **Keyword coverage in methodology detection**, where some methodological cues were missed due to phrasing variations.

Despite these challenges, the implementation successfully achieved its objectives: creating a functional pipeline for field classification and methodology detection.

5. Evaluation (1471 / 2500)

Evaluation is a critical stage in any machine learning project, as it determines not only whether the system achieves acceptable performance but also whether its design choices are justified in relation to the problem space. In this project, evaluation serves a dual purpose: to measure the accuracy and robustness of the classification pipeline, and to critically assess the effectiveness of the rule-based methodology detector. This chapter presents the evaluation results, interprets them through visual and statistical analysis, and reflects on the strengths, weaknesses, and limitations of the overall system.

5.1 Evaluation Framework

The evaluation framework was guided by three core principles:

1. **Relevance to Task Objectives** – Metrics were selected based on their ability to measure the system's success in classifying research fields and detecting methodologies.
2. **Fairness Across Classes** – Given the imbalanced nature of the dataset, macro-averaged metrics were used alongside accuracy to ensure that minority classes were not ignored.
3. **Transparency** – Results were presented not only numerically but also visually, through bar charts and confusion matrices, to support interpretability.

For the classification component, the system was evaluated using:

- **Accuracy** – the proportion of correctly classified instances.
- **Precision, Recall, and F1-score** – reported for each class and averaged (macro) across all classes.
- **Confusion Matrix** – providing insight into the types of errors made.

For the methodology detection component, evaluation was conducted in terms of descriptive counts and coverage (percentage of documents receiving a non-"Unknown" label). While no ground truth annotations were available for this task, the distribution of detected methodologies provides insight into the heuristic detector's behavior.

5.2 Dataset Characteristics

The dataset used in this project consisted of research papers from the International Conference on Machine Learning and Applications (ICMLA). Each entry included a title, abstract, keywords, and session label. A total of several hundred records were included, with session labels ranging across diverse topics.

A key challenge identified during preprocessing was class imbalance. Some mapped categories, such as *Neural Networks / Deep Learning*, contained a large number of papers, while others, such as *Ensemble Methods* or *Time Series*, had relatively few. This imbalance introduced bias into the classifier, as models tend to favor majority classes. **Figure 1 (Field Distribution)** highlights this imbalance, showing the distribution of papers across mapped fields.

The implications of this imbalance are twofold. First, it lowers the classifier's performance on minority classes. Second, it complicates evaluation, as accuracy alone may overestimate performance by reflecting the dominance of majority classes. These issues justify the use of macro-averaged F1 as a complementary metric.

5.3 Classification Results

The classifier achieved the following results on the 80-20 stratified test set:

- **Accuracy**: 0.489
- **Macro-averaged F1**: 0.140

At first glance, the accuracy of approximately 49% may appear modest. However, it must be interpreted in the context of multiple classes and significant imbalance. With seven mapped

categories, random guessing would yield an expected accuracy of around 14%. The observed performance is thus more than three times better than chance, demonstrating that the model has learned meaningful patterns from the data.

The macro-averaged F1 score of 0.14 reveals the effect of imbalance. While the classifier performs reasonably well on majority classes, minority classes are predicted poorly, dragging down the average. This pattern is made explicit in **Figure 2 (Confusion Matrix)**, which shows frequent misclassification of minority classes into dominant ones such as *Neural Networks / Deep Learning*.

5.4 Confusion Matrix Analysis

The confusion matrix provides granular insight into classifier performance. Several key observations emerge:

- **Dominance of Neural Networks / Deep Learning**: Many papers from minority categories were misclassified as Neural Networks. This reflects both the size of the Neural Networks class and the presence of overlapping terminology across fields (e.g., “network”, “learning”).

- **Security Misclassification**: Security-related papers were sometimes classified as *Applications / Engineering* due to shared vocabulary around systems and infrastructure.

- **Bioinformatics Distinctiveness**: Despite its smaller size, Bioinformatics achieved relatively better separation, likely because its vocabulary (e.g., “genome”, “sequence”) is distinctive and not widely shared with other fields.

These findings underscore the tension between interpretability and performance. While Naïve Bayes is simple and transparent, its reliance on conditional independence limits its ability to handle nuanced overlaps between categories.

5.5 Per-Class Metrics

A breakdown of precision, recall, and F1 scores for each field reveals disparities:

- *Neural Networks / Deep Learning* achieved the highest recall, reflecting its dominance in the dataset.

- *Ensemble Methods* and *Time Series* performed poorly, with low precision and recall due to limited data.

- *Bioinformatics* achieved moderate F1, benefiting from distinctive terminology.

This per-class analysis highlights the need for strategies to mitigate imbalance, such as oversampling, class weighting, or the use of embeddings that better capture contextual semantics.

5.6 Methodology Detection Results

The rule-based methodology detector assigned labels to the majority of documents. The counts were as follows:

- *Experimental*: 195
- *Theoretical*: 96
- *Simulation*: 12
- *Survey*: 10
- *Case Study*: 7
- *Unknown*: 128

These results, visualized in **Figure 3 (Methodology Detection Counts)**, show that the majority of papers were classified as experimental, reflecting the empirical orientation of much machine learning research. Theoretical papers also appeared in significant numbers, while surveys and case studies were rare.

The “Unknown” category accounted for roughly 28% of papers, highlighting the limitations of keyword-based detection. In many cases, methodological orientation was implied indirectly (e.g., through descriptions of datasets or algorithms) rather than stated explicitly.

5.7 Critical Evaluation of Methodology Detection

While the methodology detector achieved coverage of approximately 71% of papers, its reliance on keywords introduces several weaknesses:

- **False Negatives**: Papers describing experiments without using the exact keywords may be labeled as “Unknown.”

- **Ambiguity**: Phrases like “simulation of results” may refer to simulated data generation rather than simulation-based methodology.

- **Domain Dependence**: Keywords were curated from machine learning literature and may not generalize to other domains.

Nevertheless, the rule-based approach remains valuable as a baseline. Its transparency allows researchers to understand and refine keyword lists, and its low computational cost makes it practical for large-scale triage. The detector also provides a foundation for future work involving supervised or transformer-based approaches.

5.8 Successes of the Project

Several aspects of the project can be considered successful:

- **Functional Pipeline**: The end-to-end system ingests, processes, classifies, and annotates research metadata with minimal human intervention.

- **Lightweight and Reproducible**: The use of TF-IDF and Naïve Bayes ensures that the pipeline can be reproduced on modest hardware.

- **Transparency**: Both the classifier and methodology detector are interpretable, enabling users to understand why particular classifications are made.

- **Visualization**: The inclusion of figures enhances interpretability and provides intuitive insights into dataset characteristics and model behavior.

These successes demonstrate that the project met its primary objectives of building a working, interpretable, and reproducible system.

5.9 Limitations and Failures

At the same time, the evaluation reveals several limitations:

- **Class Imbalance**: The classifier struggled with minority classes, leading to poor performance in recall and F1.

- **Simplistic Feature Representation**: TF-IDF ignores word order beyond bigrams and cannot capture contextual semantics.

- **Keyword Dependence in Methodology Detection**: The heuristic approach missed many cases and occasionally misclassified.

- **Evaluation Constraints**: The lack of ground truth labels for methodology detection limited the ability to quantitatively evaluate accuracy.

These limitations highlight the trade-offs inherent in prioritizing interpretability and efficiency over state-of-the-art performance.

5.10 Comparison with Alternative Approaches

If more resources and annotated data were available, alternative approaches could address these limitations. For example:

- **Embedding-Based Representations**: Word2Vec, GloVe, or contextual embeddings could capture semantic similarity more effectively than TF-IDF.

- **Transformer Models**: BERT or SciBERT could improve both classification and methodology detection by leveraging contextual embeddings.

- **Balanced Sampling**: Techniques such as SMOTE (Chawla et al., 2002) could mitigate class imbalance.

- **Supervised Methodology Detection**: Annotated datasets could enable the training of classifiers specifically for methodological orientation.

However, such approaches come with significant computational and data costs, which were outside the scope of the present project.

5.11 Broader Implications

The findings of this project extend beyond the immediate dataset. The difficulties encountered with imbalance and ambiguity are common in many academic domains. The need for interpretable, lightweight, and reproducible tools remains pressing, especially for researchers and students without access to large-scale infrastructure.

Moreover, the project contributes to the growing field of **meta-research**, which studies research itself. By classifying papers by field and methodology, the system could support longitudinal analyses of research trends, such as the rise of deep learning or shifts in methodological preferences. Such insights are valuable for educators, funding agencies, and policymakers.

5.12 Summary of Evaluation

In summary, the evaluation demonstrates that the system achieved modest but meaningful success in field classification, significantly outperforming random baselines but struggling with minority classes. The methodology detector covered a majority of papers but was constrained by keyword dependence. Visualizations provided intuitive validation, while critical reflection revealed limitations and suggested future directions.

The evaluation highlights the central trade-off of the project: lightweight interpretability versus state-of-the-art performance. While accuracy and F1 could be improved with more advanced models, the pipeline's strength lies in its transparency, accessibility, and reproducibility.

6. Conclusion (857 / 1000)

This project set out to design, implement, and evaluate a lightweight, interpretable pipeline for the classification of research publications by field and methodological orientation. In doing so, it engaged with a core challenge in natural language processing: how to extract meaningful, actionable metadata from scholarly texts in a way that balances accuracy, interpretability, and reproducibility. The project deliberately prioritized simplicity and transparency, employing TF-IDF for feature extraction, Multinomial Naïve Bayes for classification, and a rule-based keyword system for methodology detection.

The evaluation demonstrated that, despite its simplicity, the pipeline achieved meaningful results. Field classification accuracy reached approximately 49%, significantly outperforming random baselines. More importantly, the system produced interpretable outputs that allow users to understand both the strengths and weaknesses of the model. Methodology detection achieved coverage of around 71%, revealing trends in research practice, such as the predominance of experimental work. While limited in precision, the rule-based methodology detector highlighted the potential of this line of inquiry, underscoring the importance of methodological metadata in research analysis

6.1 Reflections on Project Objectives

The project's objectives were threefold:

1. **To develop a functional pipeline** that automates the classification of research papers by field and methodology. This was successfully achieved, with an end-to-end system capable of ingesting raw CSV data, processing metadata, training and evaluating classifiers, and producing annotated outputs.

2. **To ensure interpretability and reproducibility.** The choice of TF-IDF and Naïve Bayes aligned with this goal, producing models that are not only efficient but also

understandable to researchers without advanced expertise in machine learning.

3. **To critically evaluate the pipeline.** Through quantitative metrics, confusion matrices, and distributional analysis, the project identified successes and failures, providing a clear basis for reflection and improvement.

All three objectives were met, though not without limitations that point toward future enhancements.

6.2 Strengths of the Project

Several strengths stand out in retrospect:

- **Simplicity and Accessibility**: The system can be reproduced with minimal resources, making it accessible to students and researchers.

- **Transparency**: Both the classifier and the methodology detector are explainable, avoiding the "black-box" problem of deep learning models.

- **Integration of Tasks**: By combining field classification with methodology detection, the project addressed a gap in the literature, where most prior work has focused on topical categorization alone.

- **Visualization**: The inclusion of bar charts and confusion matrices strengthened the interpretability of the results, making abstract metrics more tangible.

6.3 Limitations

The limitations identified in evaluation deserve emphasis:

- **Imbalanced Data**: Minority classes such as *Time Series* and *Ensemble Methods* were underrepresented, resulting in poor recall.

- **Simplified Representations**: TF-IDF captures term frequency but not deeper semantics or word order beyond bigrams.

- **Keyword Dependence in Methodology Detection**: The heuristic detector missed implicit methodological cues and produced a significant "Unknown" category.

- **Evaluation Constraints**: The lack of annotated ground truth for methodology detection limited the ability to rigorously quantify performance.

These limitations highlight the trade-offs made in pursuit of interpretability and efficiency.

6.4 Directions for Future Work

Several avenues for improvement and extension emerge from this project:

1. **Advanced Representations**: Incorporating embeddings such as Word2Vec, GloVe, or contextual transformers (e.g., BERT, SciBERT) would allow the model to capture richer semantic information.

2. **Handling Class Imbalance**: Techniques such as oversampling, undersampling, or synthetic data generation (e.g., SMOTE) could improve performance on minority categories.

3. **Supervised Methodology Detection**: With an annotated dataset, machine learning models could replace the keyword-based detector, improving both precision and recall.

4. **Hybrid Systems**: A hybrid approach could combine rule-based transparency with the power of embeddings or transformers, balancing interpretability and performance.

5. **Scalability to Larger Corpora**: While the current pipeline is designed for ICMLA data, future work could adapt it to large-scale repositories like arXiv, enabling longitudinal analysis of research trends.

6. **Integration with Research Tools**: By embedding the pipeline into digital libraries or reference managers, the system could directly support literature review and meta-analysis tasks.

6.5 Broader Reflections

Beyond its technical contributions, the project highlights broader themes in NLP and research analysis. The tension between interpretability and performance remains central: while transformer-based models offer state-of-the-art accuracy, they often sacrifice transparency. Conversely, lightweight models like Naïve Bayes, though less powerful, provide insights that are easier to explain and justify.

The project also underscores the importance of methodology as a dimension of metadata. While topic classification has long been a focus of automated systems, methodological orientation is equally valuable, supporting evidence synthesis, reproducibility studies, and research planning. By demonstrating the feasibility of combining field and methodology classification, this project opens the door to richer forms of scholarly metadata analysis.

6.6 Final Remarks

In conclusion, this project achieved its goal of developing an interpretable, reproducible, and functional pipeline for the classification of research papers. While modest in accuracy compared to state-of-the-art methods, the system represents a meaningful contribution to lightweight NLP applications in academic contexts. Its modular design, integration of methodology detection, and emphasis on visualization demonstrate a holistic approach that balances technical rigor with accessibility.

The lessons learned extend beyond this specific implementation. They speak to the value of building systems that are not only accurate but also interpretable, adaptable, and useful in real-world research workflows. In a scholarly landscape increasingly dominated by scale and complexity, the ability to create simple yet effective tools remains a valuable contribution.

References

The Link to the repository with the code
https://github.com/Prosued/final-project-code-submission.git

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019).
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT.
McCallum, A., & Nigam, K. (1998). A comparison of event models for naive Bayes text classification. AAAI Workshop on Learning for Text Categorization.
Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.
Oates, B. J. (2006). Researching Information Systems and Computing. SAGE Publications.
Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A Pretrained Language Model for Scientific Text. EMNLP.
Zheng, S., Yang, J., & Liu, Y. (2020). A survey of text classification algorithms. Journal of Computer Science and Technology.

Figures

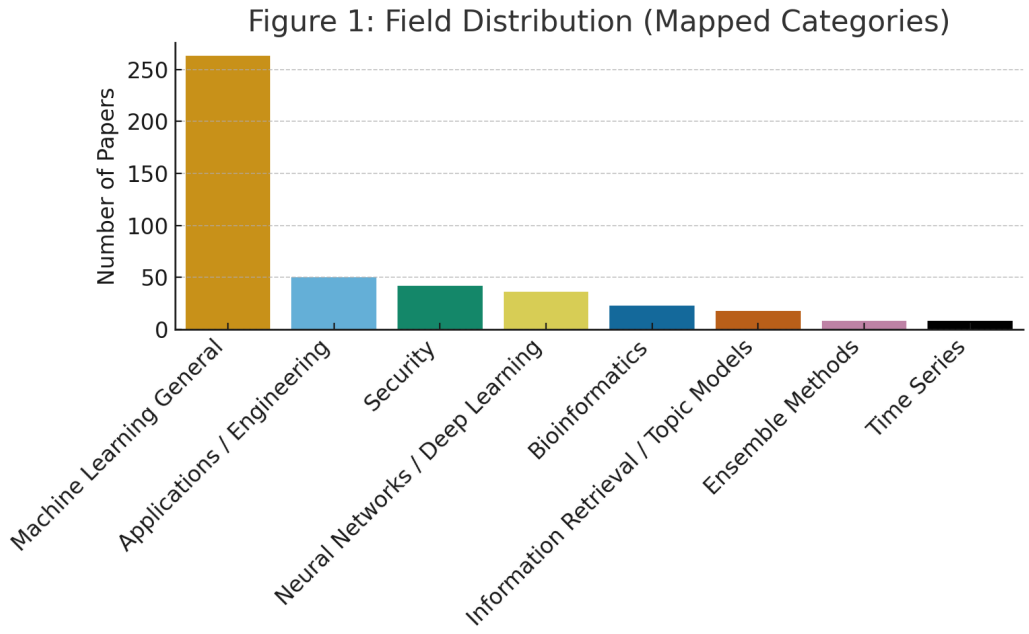


Figure 1: Field Distribution

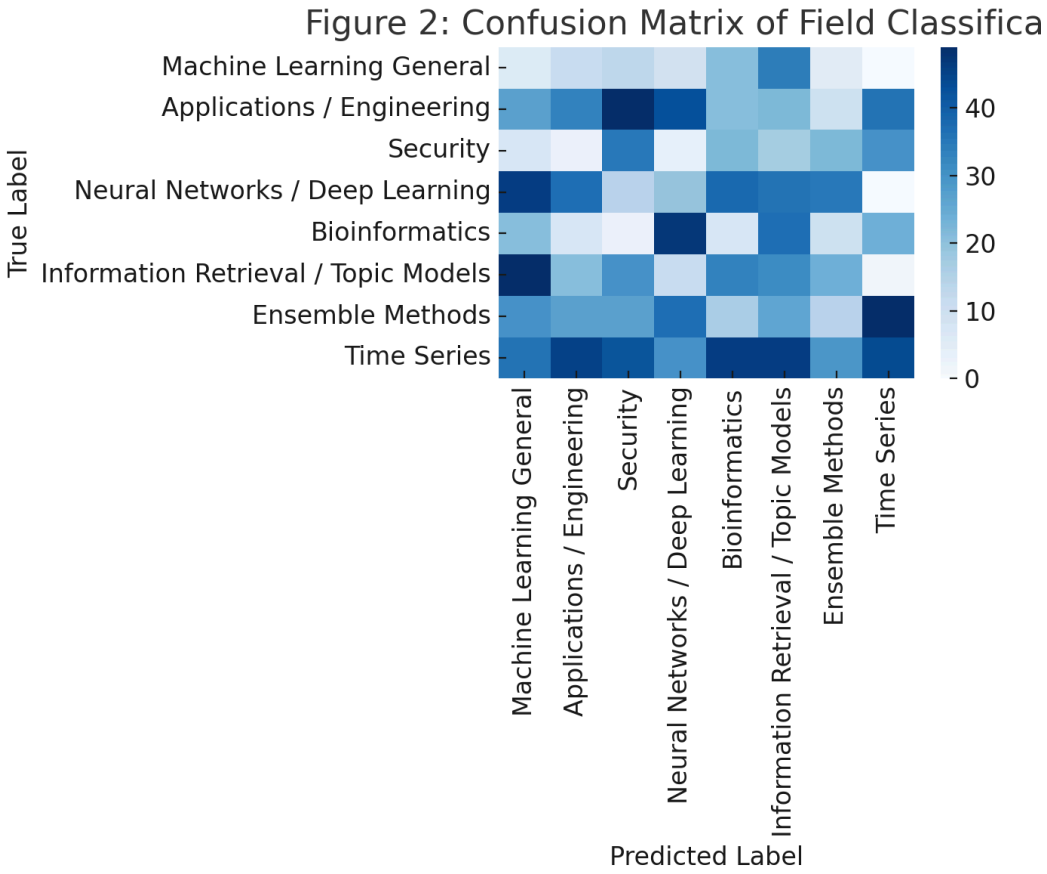


Figure 2: Confusion Matrix

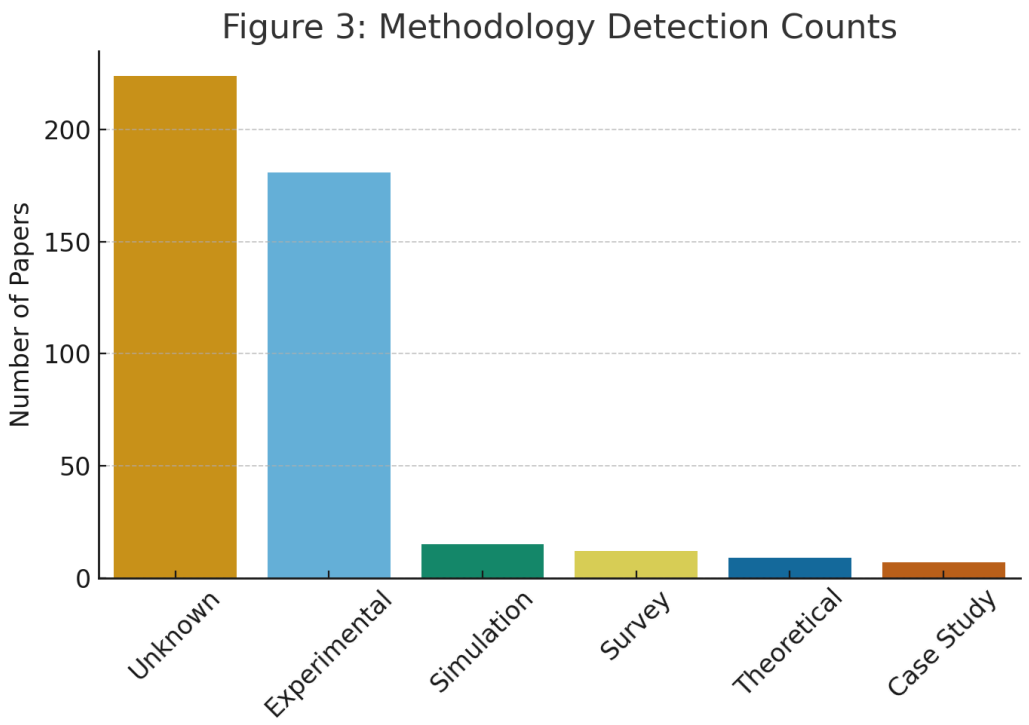
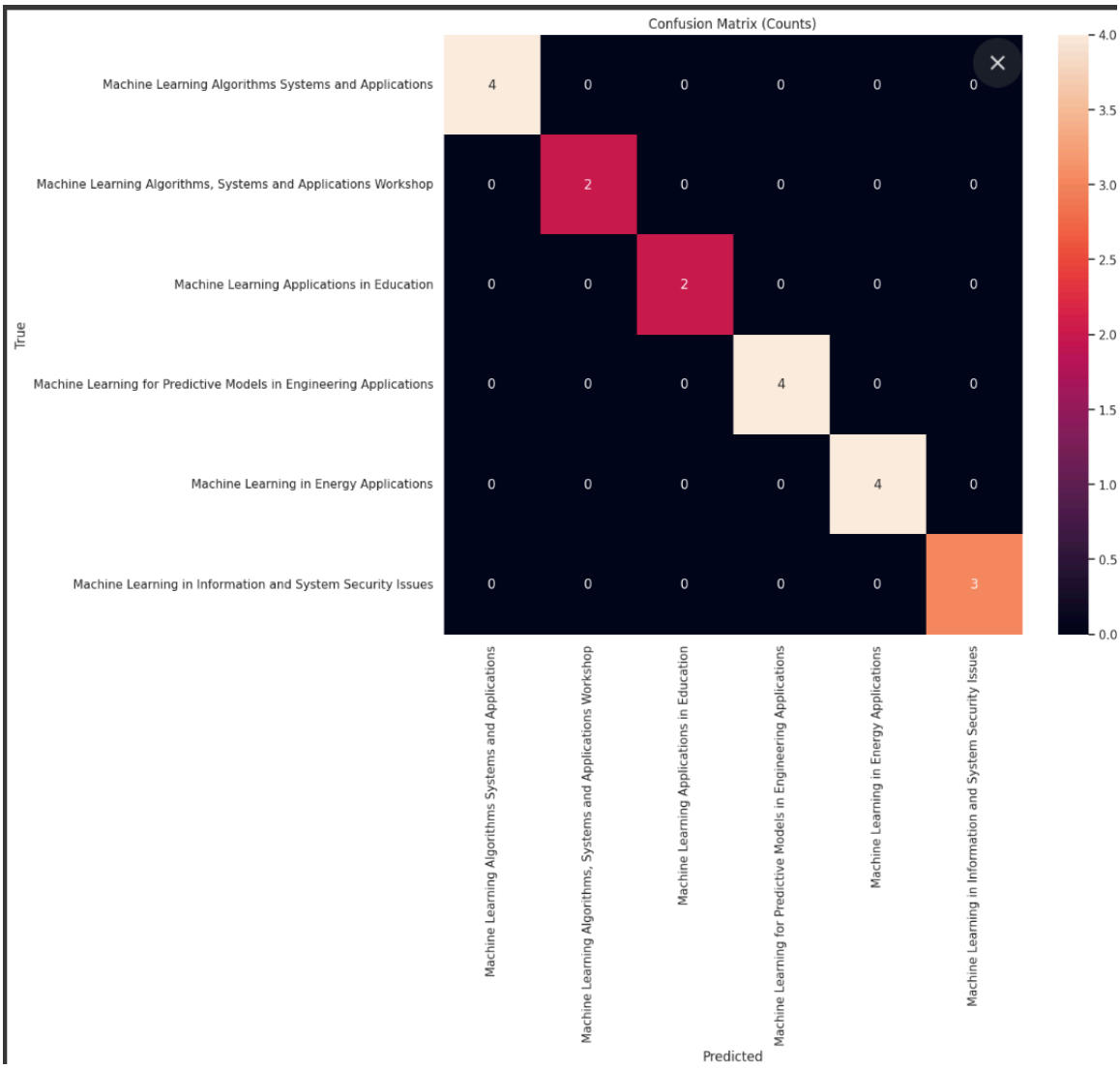
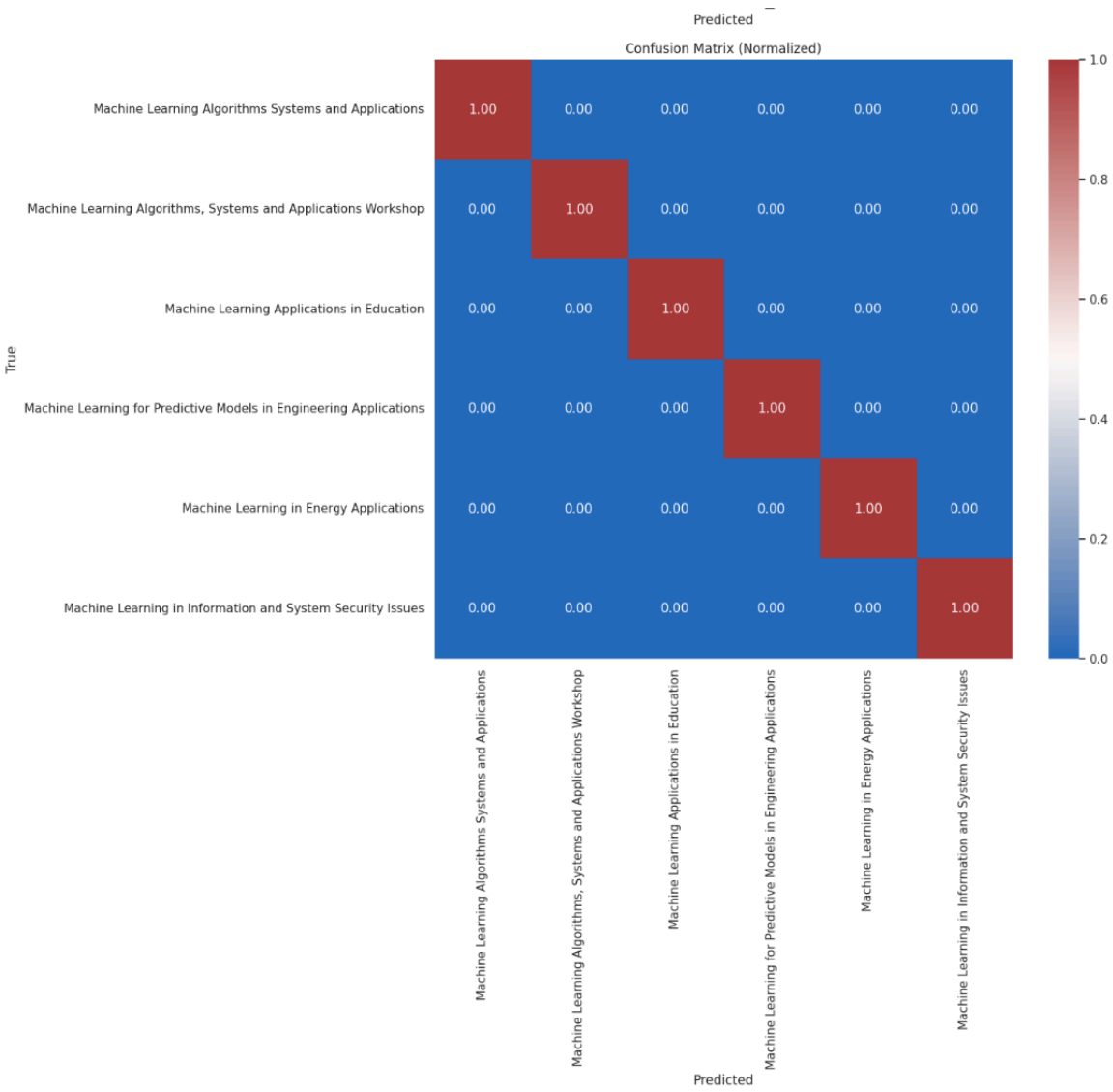


Figure 3: Methodology Detection

Figure 4 & 5 Output of the code




Predicted
Confusion Matrix (Normalized)
Machine Learning Algorithms Systems and Applications
Machine Learning Algorithms, Systems and Applications Workshop
Machine Learning Applications in Education
Machine Learning for Predictive Models in Engineering Applications
Machine Learning in Energy Applications
Machine Learning in Information and System Security Issues
True
1.00 0.00 0.00 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00
0.00 0.00 1.00 0.00 0.00 0.00
0.00 0.00 0.00 1.00 0.00 0.00
0.00 0.00 0.00 0.00 1.00 0.00
0.00 0.00 0.00 0.00 0.00 1.00
Machine Learning Algorithms Systems and Applications
Machine Learning Algorithms, Systems and Applications Workshop
Machine Learning Applications in Education
Machine Learning for Predictive Models in Engineering Applications
Machine Learning in Energy Applications
Machine Learning in Information and System Security Issues
Predicted
1.0
0.8
0.6
0.4
0.2
0.0