PAP_OS02

Machine Learning Model for Risk Qualification of R Libraries

Felipe Jimenez, Fortrea, San Jose, Costa Rica
Maripaz Montero, Fortrea, San Jose, Costa Rica

ABSTRACT

Regulatory compliance in clinical trials reporting requires the use of reliable, well-documented, and validated tools. While agencies such as the FDA do not mandate specific software for statistical analysis , organizations must ensure that their tools meet validation standards. To support a more objective and scalable qualification of R open-source libraries, we developed a machine learning framework to assess their risk in the context of clinical reporting. The model classifies libraries as high or low risk based on 14 data quality metrics collected using the `riskmetric` and `testthis` libraries. Data were grouped into threshold, training, and validation sets to build and evaluate models performance. After applying feature selection techniques, three machine learning algorithms were tested. The Support Vector Machine (SVM) model outperformed the others, achieving an F1 score of 0.97. This framework offers a robust, data-driven alternative to traditional scoring systems for qualifying R libraries in regulatory submissions.

INTRODUCTION

Are traditional scoring systems—based on expert-assigned weights—reliable enough to support highly regulated clinical trial statistical analyses produced using open-source technologies?

The adoption of open-source technologies is gaining significant momentum in the pharmaceutical industry, particularly for clinical trial statistical deliverables. According to a 2023 industry survey by the pharmaverse consortium (Siggaard Knoph, A., & Farrugia, R. 2024), approximately one-third of clinical trial statisticians now use R regularly —up from one in ten five years ago.

This shift opens new possibilities for innovation and efficiency, but also introduces challenges, especially around infrastructure and software validation. From a regulatory standpoint,open-source tools are acceptable in trials,provided that organizations implement robust validation procedures. The R Consortium has developed a comprehensive ecosystem to address these challenges focusing on accuracy, reproducibility and traceability perspective. However, when it comes to determining a software's "risk score", the R Validation Hub white paper acknowledges that this remains an "opaque process" that is outside the scope of its framework (R Validation Hub 2020).

Historically, scoring systems have relied on subjective weighting—expert-driven judgments assigning importance to various software quality metrics. While straightforward, this approach introduces bias and lacks the adaptability needed for today's rapidly evolving open-source ecosystem.

This paper proposes a supervised machine learning framework to objectively assess and qualify the risk of R libraries used in highly regulated clinical trial analyses. By leveraging metadata from open-source R libraries and building on the methodology of the R Validation Hub, the model evaluates risk using 14 predefined quality metrics. Rather than relying on manually assigned weights, the machine learning algorithm identifies the most predictive variables and assigns weights accordingly. The result is a dynamic, data-driven process that generates a reproducible risk score, enabling classification of libraries into high- or low-risk categories.

LIMITATIONS OF TRADITIONAL SUBJECTIVE RISK SCORING

Traditionally, software risk assessment systems—including those used in clinical trials—rely on identifying risk criteria and assigning weights to each element based on expert judgment to produce an overall risk score. While this method can provide directional insights, it presents several limitations (Kumar Arundhati 2025) :

- **Subjective Weighting & Bias:** Weights or thresholds are often determined by consensus rather than empirical evidence.This introduces expert bias, and may lead to non reproducible results if the experts, company or context changes.
- **Inconsistency and Limited Defensibility:** Because the weights are subjective,the rationale behind them can be difficult to defend. In regulated environments, auditors may question why a particular metric was assigned a specific weight (e.g., 20% instead of 25%).
- **Limited Detection of Emerging Risks:** Static scoring systems may fail to identify new signals in evolving open-source ecosystems. For example, they might overlook sudden maintainer dropouts or surges in unresolved bugs.

MACHINE LEARNING APPROACHES TO SOFTWARE RISK ASSESSMENT

Machine learning (ML) methods have been widely adopted for risk assessment across various industries, including IT project management, cybersecurity, and regulated sectors. These models offer the ability to learn from historical data and identify patterns that may not be evident through traditional expert-based scoring. Below are several relevant use cases:

- **Predictive Modeling of Project Risk:** Supervised models have been used to predict success or failure of software projects based on early risk indicators, such as unrealistic objectives, lack of user involvement or technology related issues,etc. These models provide evidence-based risk predictions, assuming access to quality historical data (Ibraigheeth et al. 2022).
- **Bug vulnerability prediction:** In software maintenance, random forest or regression trees have been applied to predict which components are likely to have defects or vulnerabilities. Inputs include code complexity, commit history, developer activity, and static analysis results.
- **Risk scoring in regulated industries:** A 2025 industry analysis on AI-enhanced risk scoring reported that AI/ML models can process a broader range of risk indicators in real time, improving detection rates by up to 72% compared to traditional methods (Kumar Arundhati 2025). These models continuously learn from new data, offering faster adaptation than manual assessments.
- **Open-Source Library Risk Assessment:** While few public examples exist, ML can be applied to assess the risk of R libraries. Similar to credit scoring, a dataset of R libraries labeled as “high” or “low” risk can be used to train a model. The algorithm can then learn which combinations of metrics (e.g., lack of maintainers and low test coverage) are most predictive of risk, while down-weighting less informative features (e.g., presence of a website).

Based on the previous cited sources, the differences between traditional methods and machine learning models for open-source libraries risk validation can be summarized here:

Table 1: Comparison of Traditional vs. Machine Learning-Based Risk Scoring

Dimension	Traditional Scoring	ML-Based Scoring
Methodology	Expert-defined weights based on experience.	Learns weights from historical data.
Transparency	Rationale may be undocumented or informal.	Transparent and reproducible logic.
Bias & Subjectivity	Prone to human bias and variability.	Reduces bias through data-driven decisions; still dependent on training data quality.
Reproducibility	Results may vary across reviewers or organizations.	Consistent scoring when model and data are version-controlled.
Scalability	Suitable for small-scale assessments; limited for large-scale evaluations.	Efficient for evaluating hundreds of libraries automatically.
Regulatory Defensibility	Requires detailed justification and expert rationale.	Auditable and supported by performance metrics; may require explanation of model logic.
Emerging Risk Detection	Limited ability to detect new or evolving risk patterns.	Capable of identifying novel risk signals through continuous learning.
Implementation Complexity	Simple to implement; minimal technical infrastructure required.	Requires data infrastructure, model training, and validation expertise.

Sources: R Validation Hub (2020); Kumar Arundhati (2025); Ibraigheeth et al. (2022)

METHODOLOGY AND DATA

According to (R Validation Hub 2020), the FDA defines validation as the process of establishing documented evidence that provides a high degree of assurance that a specific process consistently produces results meeting predetermined specifications. This includes accuracy, reproducibility, and traceability. This paper focuses primarily on the accuracy dimension of validation.

The R Validation Hub distinguishes between two types of R libraries: core (base and recommended) and contributed (open source) libraries.

- **Core libraries** (base and recommended), developed by the R Foundation, follow a rigorous software development lifecycle. This includes structured testing, controlled release management, and community feedback mechanisms. These practices ensure that core libraries are considered low risk for regulatory use.
- **Contributed libraries**, developed by the broader community and submitted to CRAN, undergo basic technical checks (e.g., successful code execution, test passing, compatibility with other libraries). However, these checks do not guarantee accuracy, making additional risk assessment necessary.

Organizations often categorize libraries into low, medium, or high risk based on their type and intended use. Core libraries are typically whitelisted as low risk, with some organizations extending this to include the Tidyverse (PharmaR 2023). Packages are further classified based on whether they are directly loaded by users ("intended for use") or used as dependencies ("imports"). While imports are accessed indirectly, their accuracy is typically ensured through the intended-for-use libraries (R Validation Hub 2020).

Each intended to use library is evaluated under a series of metrics that assess the reliability of the library to be used in validated environments. As per the ICH (International Council on Harmonization of Technical Requirements for Pharmaceuticals for Human Use) and its Statistical Principles for Clinical Trials, the "software used should be reliable, and documentation of appropriate software testing procedures should be available" (ICH 1998). Following this principle, the libraries included in these protocols must follow these principles:

- Package has a well-defined purpose/scope.
- Package has been exposed to the community for several years.
- Package is developed to a high standard with suitable tests covering the key functionality.

To assess risk, the R Validation Hub outlines four key evaluation categories:

- **Purpose:** There are many ways to categorize the library purpose, for simplicity, two classifications are proposed: Statistical or Non-Statistical.
- **Good Practice in Maintenance:** Evaluates software development practices such as presence of vignettes, websites, bug tracking, source control, release frequency, and codebase size.
- **Testing:** Assesses the presence and quality of testing practices, which support functional reliability and maintainability.
- **Community usage:** Considers indicators such as time since first release and download volume, reflecting real-world exposure and informal validation.

R VALIDATION FRAMEWORK OPERATIONALIZATION WITH `riskmetric`

To operationalize the R Validation Hub framework, the proposed machine learning model utilizes the `riskmetric` R library to extract metadata from contributed R libraries. These metrics evaluate development best practices, community engagement, and the sustainability of library maintenance.

The `riskmetric` library and internally developed scripts include a set of metrics designed to assess the reliability of libraries intended for use, with these metrics being largely based on fundamental SDLC principles.

METRICS SELECTION PROCESS

Each metric extracted from the `riskmetric` and `testthis` libraries returned either a binary value (e.g., 1 for "yes", 0 for "no") or a normalized numeric score. During the data quality review, only metrics with 100% completeness across the selected libraries were retained for modeling.

For core R libraries (e.g., base and recommended libraries), `riskmetric` does not return valid results due to their pre-installed nature. To address this, metric values for core libraries were manually assigned the maximum score where applicable, reflecting their low-risk status. This ensured their correct classification within the validation framework.

The following table includes the metrics from `riskmetric` and `testthis` libraries, that are aligned to the R Validation Hub principles and were selected after the cleaning process:

Table 2: Risk Metrics classification and their association with corresponding functions or libraries

Metric category	Metric	Function or Package
Code coverage	Code coverage percentage	testthis::test_coverage
	Includes usage examples	riskmetric::has_examples
	Includes vignettes	riskmetric::has_vignettes
Good Software Practice	Has an Active maintainer	riskmetric::has_maintainer
	Public code repository available	riskmetric::has_source_control
	Number of library dependencies	riskmetric::dependencies
	Includes a NEWS file	riskmetric::has_news
	NEWS file is up to date	riskmetric::news_current
	Proper namespace exports	riskmetric::exported_namespace
Bug resolution	Proportion of resolved issues	riskmetric::bugs_status
	Provides a URL for reporting bugs	riskmetric::has_bugs_reports_url
Community Usage	Number of downloads in the last year	riskmetric::downloads_1yr
	Package has a dedicated website	riskmetric::has_website
Usability metrics	Documentation for exported functions	riskmetric::export_help

Source: Metrics derived from the riskmetric and testthis R libraries, aligned with the R Validation Hub framework.

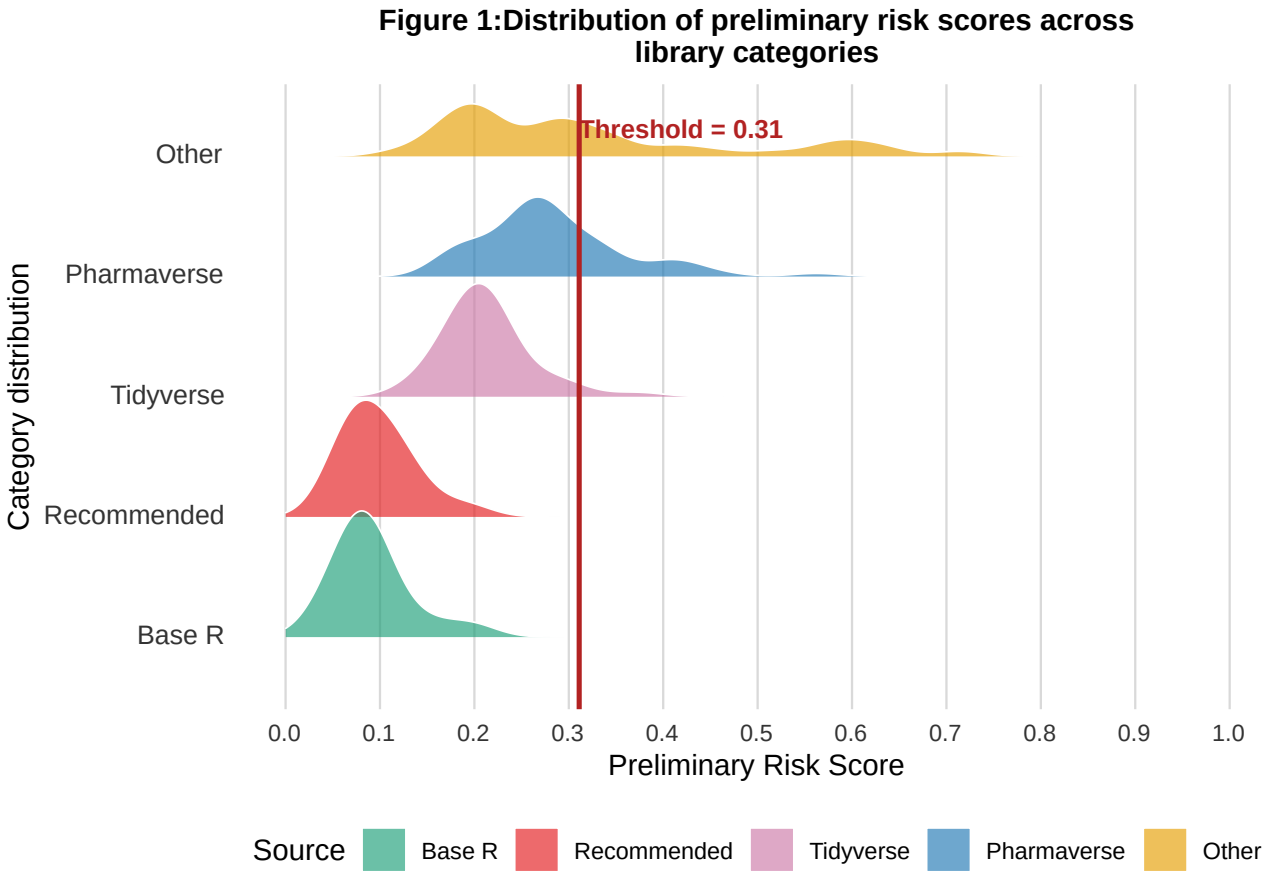
DATA COLLECTION

To train and evaluate the machine learning model, a binary classification system was adopted, labeling libraries as either "low risk" or "high risk".This could be accomplish in several ways, that will vary on data or resources availability. For this use case, the process to label the dataset was the following:

1. **Threshold Definition:** A set of 72 high-quality libraries—including Base R, Recommended, and Tidyverse libraries—was selected as a reference group. A preliminary risk score was calculated for each library using the 14 selected metrics. The highest score observed among these libraries (0.311) was established as the threshold. Libraries with scores at or below this value were labeled "low risk"; those exceeding it were labeled "high risk.".The preliminary risk score was calculated using the following formula:

$$\text{Preliminary risk score} = 1 - \sum(\text{metric's numeric value} \times \text{weight})$$

Figure 1 illustrates the distribution of preliminary risk scores across library categories, with the threshold clearly marked.



Note: The line indicates the threshold (0.311) used to classify libraries as low or high risk.

2. **Training and validation datasets:** The top 1,000 most-downloaded CRAN libraries (as of August–September 2024) were selected. Of these, 939 were successfully installed in the production environment, and 883 had complete data for all 14 metrics. These, along with the 29 core libraries, formed the modeling dataset. The "intended-for-use" libraries were excluded from training and reserved for external validation.

RESULTS

Before training the machine learning models, a exploratory and bivariate analyses was conducted to evaluate the relationship between each metric and the preliminary risk classification.

Figure 2 shows the percentage of libraries classified as low risk based on the presence or absence of each binary metric. Most metrics demonstrated strong discriminative power. For example, 80% of libraries with an up-to-date NEWS file were classified as low risk, compared to only 8% without one. The metric "has_maintainer" showed no meaningful variation, as nearly all libraries had a maintainer listed. Chi-squared tests confirmed that all binary metrics, except "has_maintainer," were significantly associated with risk classification ($p < 0.05$).

Figure 2:Percentage of Libraries Classified as Low Risk by metric presence

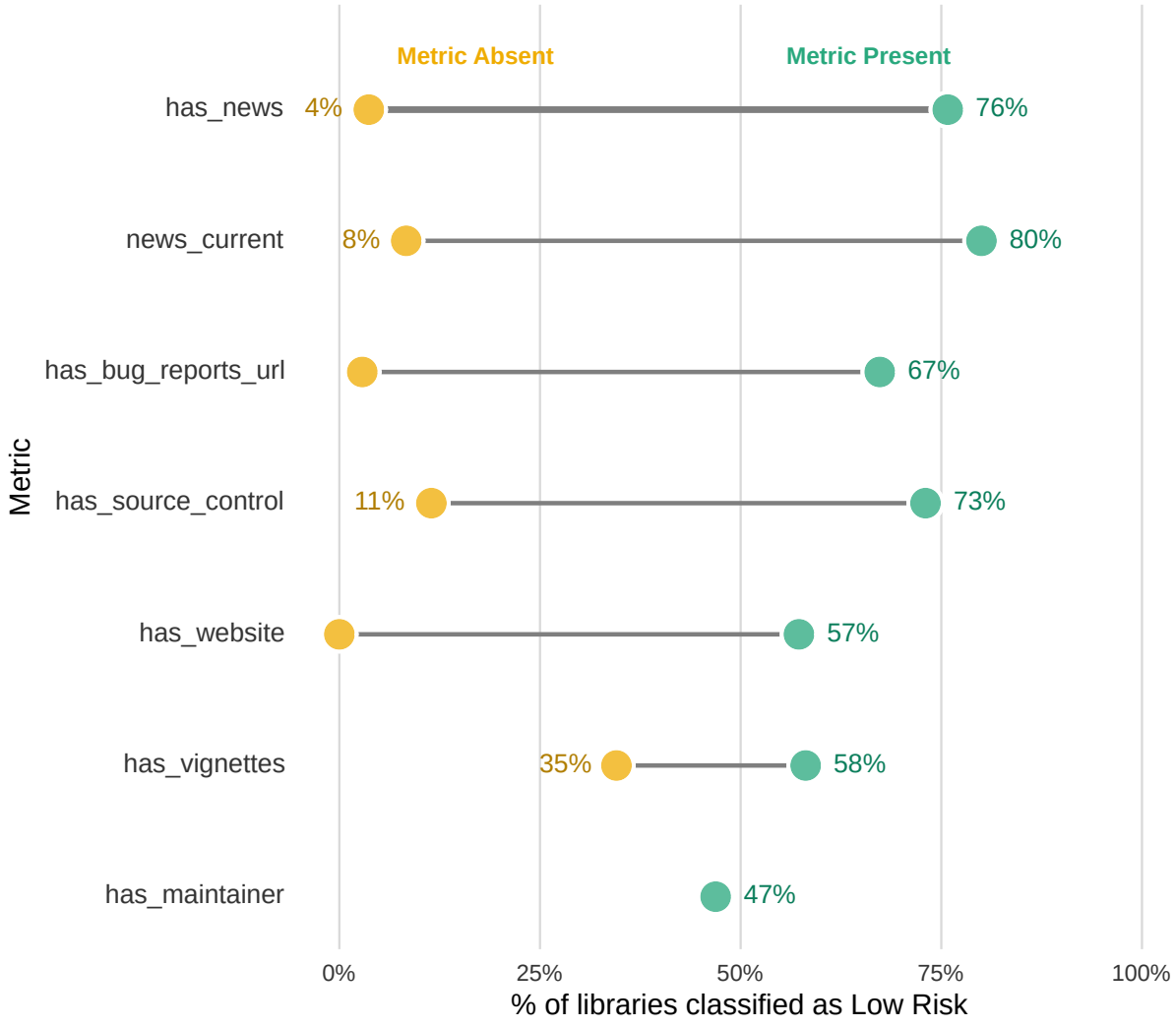
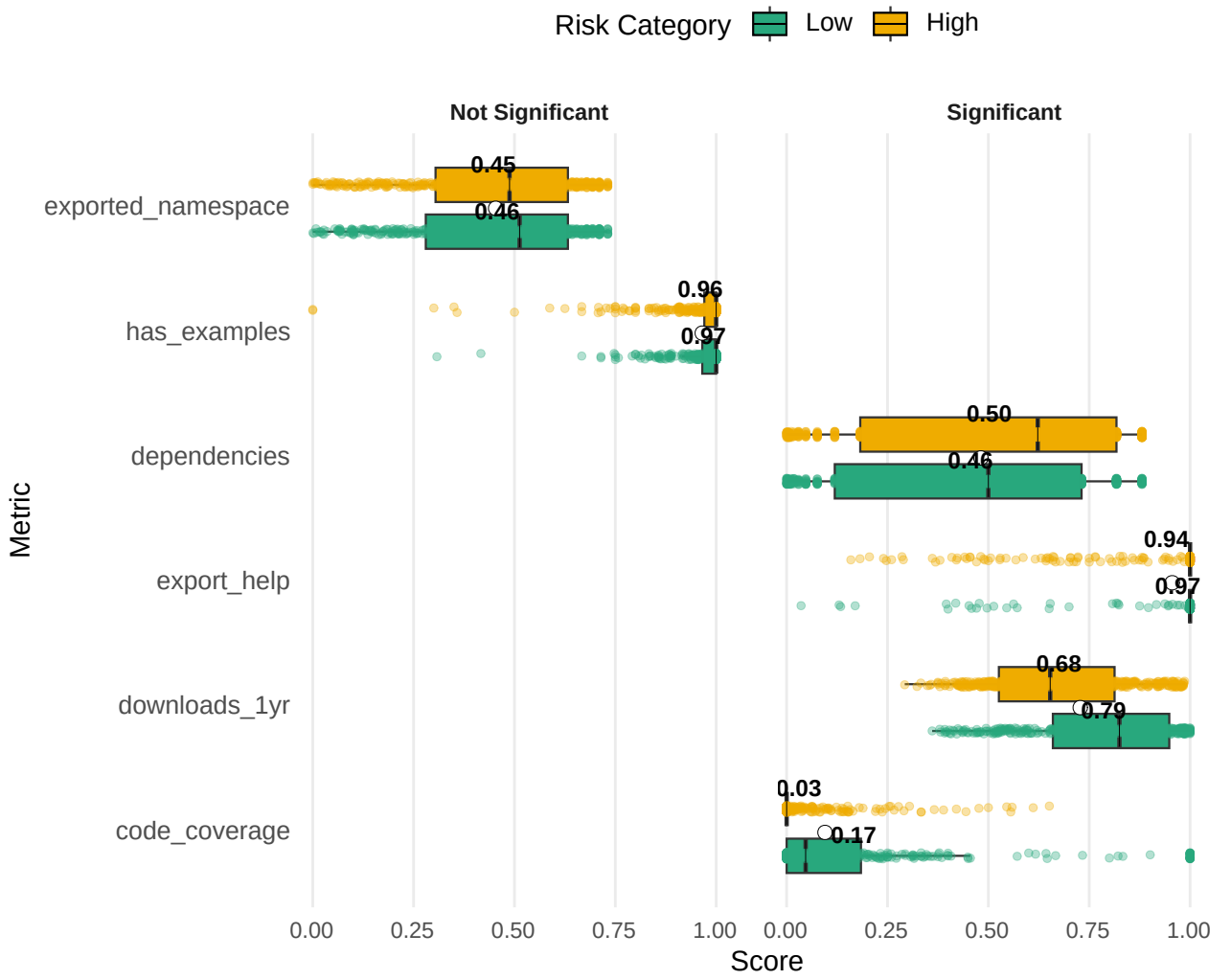


Figure 3 presents the distribution of continuous metric scores across low- and high-risk categories. Metrics such as code coverage and number of downloads showed clear separation between groups. For instance, the mean code coverage score was 0.17 for low-risk libraries and 0.03 for high-risk ones. In contrast, metrics like "has_examples" showed minimal difference between groups, suggesting limited discriminative value. These analyses informed the feature selection process and highlighted which metrics contributed most to risk differentiation.

Figure 3: Continuous metrics distribution by risk score category



Note: Numeric labels above each boxplot indicate the mean score.

MODEL RESULTS

For the 883 libraries, 414 (46.89%) are pre-classified as low risk and 469 (53.11%) as high risk. A random 3/4 split between training and testing was applied, ensuring the proportions were maintained in each dataset.

Three machine learning models—Random Forest, Support Vector Machine (SVM), and XGBoost—were trained using a 75/25 train-test split and evaluated via 10-fold cross-validation. All models demonstrated high performance, with F1 scores above 0.96.

SVM was selected as the final model due to its balanced performance and high precision (0.945) in identifying low-risk libraries. Table 3 summarizes the performance metrics across models. Table 4 presents the SVM confusion matrix, showing 103 true positives and only 6 false negatives for low-risk classification, yielding a recall of 0.99 and F1 of 0.967.

Table 3: Model Comparison metrics

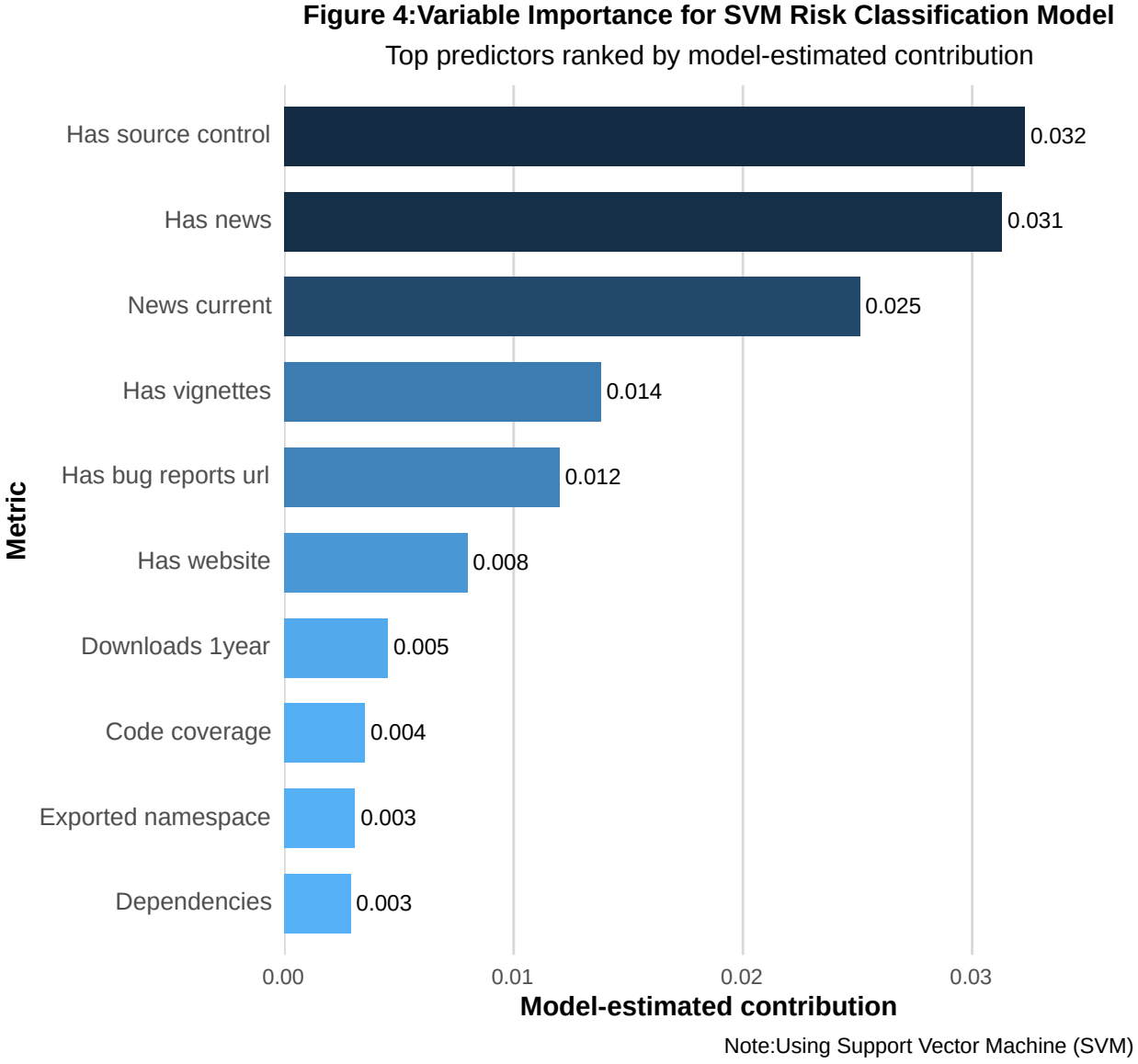
Model	Sensitivity	Specificity	Precision	Recall	F1
Random Forest	0.99	0.941	0.936	0.99	0.963
Support Vector Machine	0.99	0.949	0.945	0.99	0.967
XGBoost	0.99	0.941	0.956	0.99	0.963

Table 4: SVM Risk Libraries Model Confusion Matrix

	Actual	
Predicted	**Low risk**	**High risk**
Low risk	103	6
High risk	1	112

The Figure 4 illustrates each metric relevance and prediction power using the chosen SVM model, where the metrics “Has source control”, “has news” and “news current” provide most of the explanatory power of the risk category classification.

Figure 4:Variable Importance for SVM Risk Classification Model



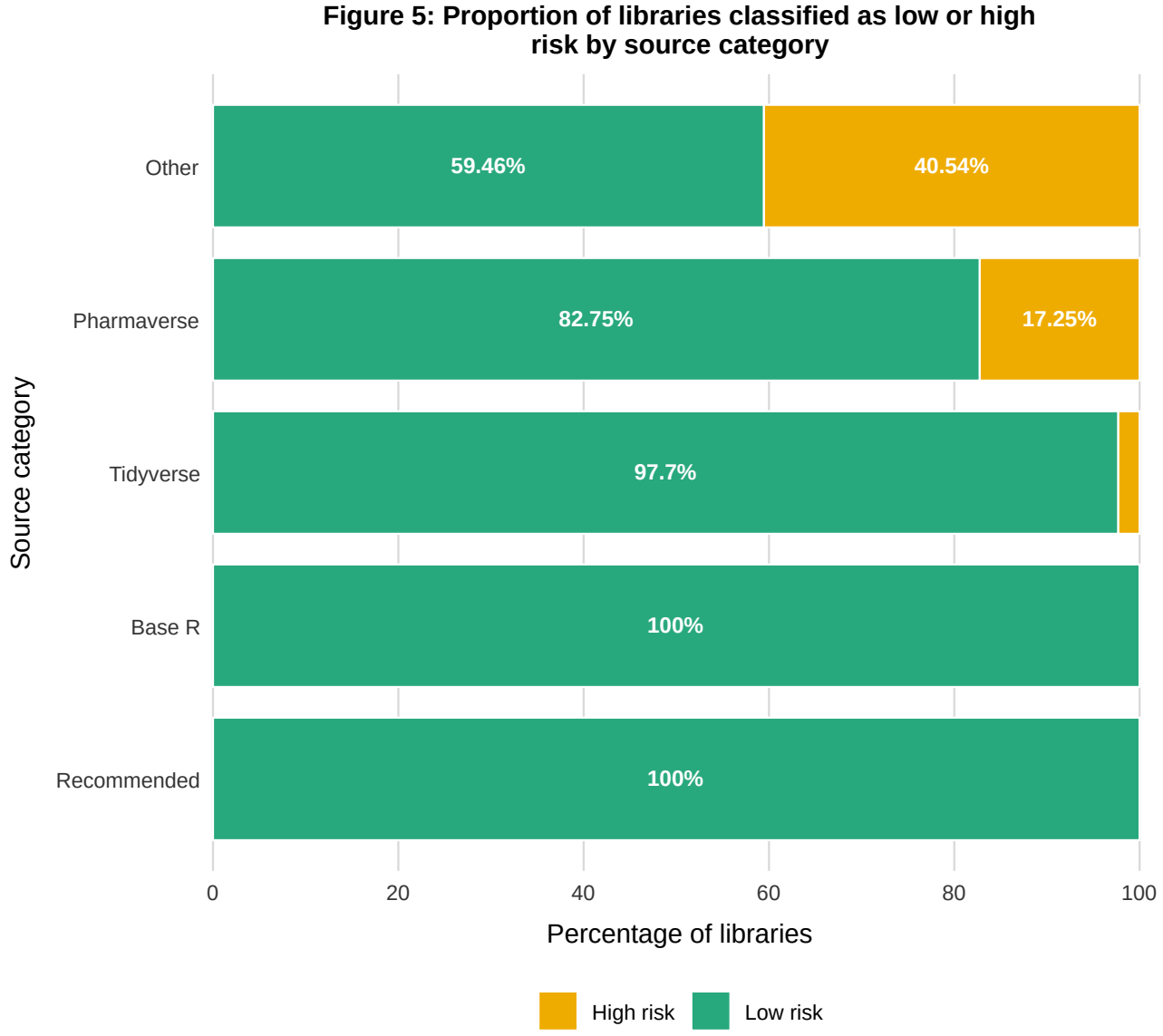
Note:Using Support Vector Machine (SVM)

MODEL VALIDATION

The final SVM model was applied to an external set of 167 intended-for-use libraries. These included 14 Base R, 15 Recommended, 43 Tidyverse, 58 Pharmaverse, and 37 internally used ("Other") libraries.

Figure 5 summarizes the classification results. As expected, 100% of Base R and Recommended libraries were classified as low risk. Tidyverse libraries showed a 97.7% low-risk rate, followed by Pharmaverse (82.7%) and Other (59.46%).

These results align with expectations that well-established, community-supported libraries are more likely to meet quality standards.



Following this validation, additional manual review was conducted for libraries near the risk threshold. Common causes for borderline classifications included incomplete metadata or information located in sources not captured by `riskmetric`. To address this, a temporary "quarantine" category was introduced to flag such libraries for further investigation before assigning a final risk classification.

IMPLEMENTATION AND DEPLOYMENT

Once the model was validated and reviewed, an API and shiny application was deployed and provided to the company's open source community portal. The application allows users to input a library name and receive a risk classification, along with the library metrics and other descriptive information.

CONCLUSION

This study presents a machine learning-based framework to assess the risk of R open-source libraries used in clinical trial environments. By leveraging 14 quality metrics and building on the R Validation Hub methodology, the model provides a reproducible, data-driven alternative to traditional expert-based scoring systems.

The Support Vector Machine (SVM) model demonstrated strong performance (F1 = 0.97), effectively classifying libraries into high- and low-risk categories. Its deployment as an API and Shiny application enables scalable, consistent, and transparent risk assessments across development teams.

This approach addresses a key gap in operationalize open-source risk evaluation by reducing subjectivity and enabling continuous adaptation as new data becomes available.

Limitations include the need for expert oversight in model development, the absence of publicly labeled datasets, and the potential for incomplete metadata. Future work may focus on expanding the framework to other ecosystems such as Python, integrating additional metrics, and refining the model with broader validation datasets.

CONTACT INFORMATION

Author Name: Felipe Jimenez
Company: Fortrea
Email: `felipe.jimenez@fortrea.com`

REFERENCES

Ibraigheeth et al. 2022. *Software Project Risk Assessment Using ML Approaches*. Northern Boarder University. https://ajmrd.com/wp-content/uploads/2022/02/E423541.pdf.

ICH. 1998. *Statistical Principles for Clinical Trials E9*. ICH. https://www.ema.europa.eu/en/ich-e9-statistical-principles-clinical-trials-scientific-guideline.

Kumar Arundhati. 2025. *AI-Enhanced Risk Scoring: A New Era in Governance and Compliance*. Analytics Insights. https://www.analyticsinsight.net/artificial-intelligence/ai-enhanced-risk-scoring-a-new-era-in-governance-and-compliance.

PharmaR. 2023. *Case Studies Summary*. PharmaR. https://www.pharmar.org/posts/news/case-studies-summary-march-2023/.

R Validation Hub. 2020. *Risk-Based Approach to Assessing r Package Accuracy, Reproducibility and Traceability*. R Validation Hub. https://pharmar.org/white-paper/.

Siggaard Knoph, A., & Farrugia, R. 2024. *How r Pharma? An Overview of How Our Industry Is Adopting the r Programming Language*. Vienna, Austria: PHUSE EU Connect. https://www.lexjansen.com/phuse/2024/os/PAP_OS01.pdf.