# Explabox: A Python Toolkit for Standardized Auditing and Explanation of Text Models

Marcel Robeer [1,2], **Michiel Bron** [1,2], **Elize Herrewijnen** [1,2], **Riwish Hoeseni**[2], **and Floris Bex** [1,3]

1 National Police Lab AI, Utrecht University, The Netherlands ROR **2** Netherlands National Police, The Netherlands **3** School of Law, Utrecht University, The Netherlands ROR

Summary

Developed to meet the practical machine learning (ML) auditing requirements of the Netherlands National Police, `Explabox` is an open-source Python toolkit that implements a standardized four-step analysis workflow: *explore*, *examine*, *explain*, and *expose*. The framework transforms models and data (*ingestibles*) into interpretable reports and visualizations (*digestibles*), covering everything from data statistics and performance metrics to local and global explanations, and sensitivity testing for fairness, robustness, and security. Designed for developers, testers, and auditors, `Explabox` operationalizes the entire audit lifecycle in a reproducible manner. The initial release is focused on text classification and regression models, with plans for future expansion. Code and documentation are available open-source at https://explabox.readthedocs.io.

Statement of need

In high-stakes environments like law enforcement, machine learning (ML) models are subject to intense scrutiny and must comply with emerging regulations like the EU AI Act (Edwards, 2022). `Explabox` was developed to address the operational challenges of ML auditing at the Netherlands National Police, where models for text classification and regression require standardized, reproducible, and holistic evaluation to satisfy diverse stakeholders—from developers and internal auditors to legal and ethical oversight bodies. Existing tools, while powerful, were often fragmented, focusing on a single aspect of analysis (e.g., only explainability or testing) and lacking a unified framework for conducting a complete audit from data exploration to final reporting.

To solve this workflow problem, we developed Explabox around a four-step analysis strategy—*explore*, *examine*, *explain*, and *expose*—inspired by similar conceptualizations of the analytical process (Biecek & Burzykowski, 2021). While comprehensive libraries like `OmniXAI` (Yang et al., 2022) offer a broad, multi-modal collection of explainers and `dalex` (Baniecki et al., 2021) provides a mature, research-driven framework for model exploration, `Explabox` was developed to fill a specific operational gap. Practitioners seeking to conduct a full audit in a model-agnostic manner often have to combine multiple, highly-specialized libraries, such as `AIF360` (Bellamy et al., 2018) for fairness metrics, `alibi explain` (Klaise et al., 2021) or `AIX360` (Arya et al., 2019) for local explanations, and `CheckList` (Ribeiro et al., 2020) for behavioral testing.

This fragmentation introduces significant challenges, particularly regarding *reproducibility* and *flexibility in communicating results*. `Explabox` addresses the reproducibility challenge by providing a unified pipeline that not only offers centralized control over random seeds, but also tracks the specific data subsets and parameters used for each function call, ensuring full traceability. Furthermore, it provides flexibility through our *digestible* object system, which

is designed to generate outputs tailored to diverse stakeholders. By integrating these critical components into a single, cohesive workflow, `Explabox` provides a practical framework that enhances the efficiency and methodological rigor of the ML auditing lifecycle, making it directly applicable to other high-stakes domains where model validation is critical, such as finance, healthcare, and law.

Explore, Examine, Explain & Expose your ML models

`Explabox` transforms opaque *ingestibles* into transparent *digestibles* through four types of *analyses* to enhance explainability and aid fairness, robustness, and security audits.

Ingestibles

Ingestibles provide a unified import interface for data and models, where layers abstract away access (Figure 1) to allow optimized processing. `Explabox` uses `instancelib` (Bron, 2023) for fast model and data encapsulation. The model can be any Python `Callable` containing a regression or (binary and multi-class) classification model. While this interface is model-agnostic, the current release provides data handling and analysis modules optimized specifically for text-based tasks. `scikit-learn` or `ONNX` models (e.g., `PyTorch`, `TensorFlow`, or `Keras`) import directly with optimizations and automatic input/output interpretation. Data can be automatically downloaded, extracted and loaded. Data inputs include `NumPy`, `Pandas`, `Hugging Face`, raw files (e.g., HDF5, CSV, or TSV), and (compressed) file folders. Data can be subdivided into named splits (e.g., train-test-validation), and instance vectors and tokens can be precomputed and optionally saved for fast inferencing.
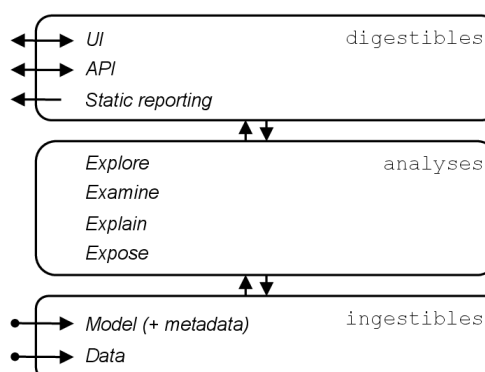

Figure 1: Logical separation of `Explabox` into layers with interfaces.

Analyses

`Explabox` turns these *ingestibles* into *digestibles* (transparency-increasing information on ingestibles) through four *analyses* types: **explore**, **examine**, **explain**, and **expose**.

Explore allows data slicing, dicing and sorting, and provides descriptive statistics (dataset sizes, label distributions, and text string/token lengths).

Examine shows model performance metrics, summarized in a table or shown graphically, with computation and interpretation references. For further analysis, **examine** also supports drilling down into (in)correct predictions.

Explain uses model-agnostic techniques (Ribeiro et al., 2016a) to explain model behavior (*global*) and individual predictions (*local*). It summarizes model-labelled data, through prototypes (K-Medoids), prototypes with criticisms (`MMDCritic` (Kim et al., 2016)), token distributions

(`TokenFrequency`), and token informativeness (`TokenInformation`). Local explanations use popular techniques: feature attribution scores (`LIME` (Ribeiro et al., 2016b), KernelSHAP (Lundberg & Lee, 2017)), feature subsets (Anchors (Ribeiro et al., 2018)), local rule-based models (`LORE` (Guidotti et al., 2018)), and counterfactual or contrastive explanations (`FoilTrees` (Waa et al., 2018)). Built from generic components separating global and local explanation steps, these methods allow customization and enable scientific advances to be quickly integrated into operational processes (e.g., combine `KernelShap` sampling with `imodels` (Singh et al., 2021) surrogate rules). Example configurations, such as `LIME` with default hyperparameters, ease adoption. **Explain** is provided by subpackage `text_explainability` (Robeer, 2021a), which doubles as a standalone tool.

Expose gathers sensitivity insights via local and global testing regimes. These insights can be used to, through relevant attributes, assess the *robustness* (e.g., the effect of typos on model performance), *security* (e.g., if inputs containing certain characters crash the model), and *fairness* (e.g., subgroup performance for protected attributes such as country of origin, gender, race or socioeconomic status) of the model. Relevant attributes can either be observed in the current data or generated from user-provided templates (Ribeiro et al., 2020) filled with multi-lingual data generation (Faraglia & Other Contributors, 2021). These attributes are then either summarized in performance metrics, compared to expected behavior (Ribeiro et al., 2020), or assessed with fairness metrics for classification (Mehrabi et al., 2021) and regression (Agarwal et al., 2019). Like **explain**, **expose** is also made from generic components, which allows users to customize data generation and tests. **Expose** is provided by the `text_sensitivity` subpackage (Robeer, 2021b), which also doubles as a standalone tool.

Digestibles

Digestibles serve stakeholders—such as creators, auditors, applicants, end-users, or clients (Tomsett et al., 2018)—via a Jupyter Notebook or Web UI (Figure 2) (using `plotly` (Plotly Technologies Inc., 2015) visuals), integrated API, and static reporting.
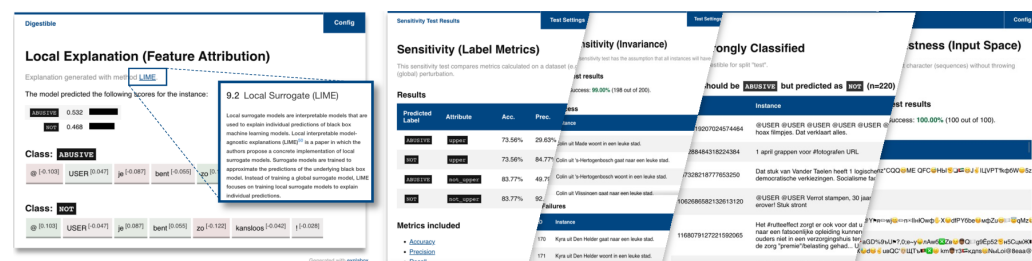

Figure 2: UI elements from the Jupyter Notebook interface, designed to present audit results to diverse stakeholders.

Acknowledgements

References

Agarwal, A., Dudik, M., & Wu, Z. S. (2019). Fair Regression: Quantitative Definitions and Reduction-Based Algorithms. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (Vol. 97, pp. 120–129). PMLR.

Arya, V., Bellamy, R. K. E., Chen, P.-Y., Dhurandhar, A., Hind, M., Hoffman, S. C., Houde, S., Liao, Q. V., Luss, R., Mojsilović, A., Mourad, S., Pedemonte, P., Raghavendra, R., Richards, J., Sattigeri, P., Shanmugam, K., Singh, M., Varshney, K. R., Wei, D., & Zhang, Y. (2019). One Explanation Does Not Fit All: A Toolkit and Taxonomy of AI Explainability Techniques. *arXiv Preprint*. https://doi.org/10.48550/arXiv.1909.03012

Baniecki, H., Kretowicz, W., Piatyszek, P., Wisniewski, J., & Biecek, P. (2021). dalex: Responsible Machine Learning with Interactive Explainability and Fairness in Python. *Journal of Machine Learning Research*, *22*(214), 1–7. http://jmlr.org/papers/v22/20-1473. html

Bellamy, R. K. E., Dey, K., Hind, M., Hoffman, S. C., Houde, S., Kannan, K., Lohia, P., Martino, J., Mehta, S., Mojsilovic, A., Nagar, S., Ramamurthy, K. N., Richards, J., Saha, D., Sattigeri, P., Singh, M., Varshney, K. R., & Zhang, Y. (2018). AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias. *arXiv Preprint*. https://doi.org/10.48550/arXiv.1810.01943

Biecek, P., & Burzykowski, T. (2021). *Explanatory Model Analysis: Explore, Explain and Examine Predictive Models*. Chapman; Hall/CRC. https://doi.org/10.1201/9780429027192

Bron, M. P. (2023). *Python Package instancelib* (Version 0.5.0). Zenodo. https://doi.org/10. 5281/zenodo.8308017

Edwards, L. (2022). The EU AI Act: A summary of its significance and scope. *Ada Lovelace Institute, Expert Explainer Report*. https://www.adalovelaceinstitute.org/wp-content/ uploads/2022/04/Expert-explainer-The-EU-AI-Act-11-April-2022.pdf

Faraglia, D., & Other Contributors. (2021). *Python package Faker*. https://github.com/ joke2k/faker

Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., & Giannotti, F. (2018). Local Rule-Based Explanations of Black Box Decision Systems. *arXiv Preprint*. https: //doi.org/10.48550/arXiv.1805.10820

Kim, B., Khanna, R., & Koyejo, O. (2016). Examples are not Enough, Learn to Criticize! Criticism for Interpretability. *29th Conference on Neural Information Processing Systems (NIPS 2016)*.

Klaise, J., Looveren, A. V., Vacanti, G., & Coca, A. (2021). Alibi Explain: Algorithms for Explaining Machine Learning Models. *Journal of Machine Learning Research*, *22*(181), 1–7. http://jmlr.org/papers/v22/21-0017.html

Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 4765–4774.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A Survey on Bias and Fairness in Machine Learning. *ACM Computing Surveys (CSUR)*, *54*(6), 1–35. https://doi.org/10.1145/3457607

Plotly Technologies Inc. (2015). *Collaborative data science*. Plotly Technologies Inc. https: //plot.ly

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016a). Model-Agnostic Interpretability of Machine Learning. *2016 ICML Workshop on Human Interpretability in Machine Learning (WHI 2016)*, 91–95. https://doi.org/10.1145/2858036.2858529

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016b). "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *22nd ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD'16), Proceedings*, 1135–1144. https://doi.org/10.18653/ v1/n16-3020

Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). Anchors: High-Precision Model-Agnostic

Explanations. *AAAI Conference on Artificial Intelligence, Proceedings*. https://doi.org/10.1609/aaai.v32i1.11491

Ribeiro, M. T., Wu, T., Guestrin, C., & Singh, S. (2020). Beyond Accuracy: Behavioral Testing of NLP models with CheckList. *Association for Computational Linguistics (ACL)*. https://doi.org/10.18653/v1/2020.acl-main.442

Robeer, M. (2021a). *Python package text_explainability*. https://doi.org/10.5281/zenodo.14192126

Robeer, M. (2021b). *Python package text_sensitivity*. https://doi.org/10.5281/zenodo.14192940

Singh, C., Nasseri, K., Tan, Y. S., Tang, T., & Yu, B. (2021). imodels: A python package for fitting interpretable models. *Journal of Open Source Software*, *6*(61), 3192. https://doi.org/10.21105/joss.03192

Tomsett, R., Braines, D., Harborne, D., Preece, A., & Chakraborty, S. (2018). Interpretable to Whom? A Role-based Model for Analyzing Interpretable Machine Learning Systems. *2018 ICML Workshop on Human Interpretability in Machine Learning (WHI 2018)*.

Waa, J. van der, Robeer, M., Diggelen, J. van, Neerincx, M., & Brinkhuis, M. (2018). Contrastive Explanations with Local Foil Trees. *2018 Workshop on Human Interpretability in Machine Learning (WHI 2018)*.

Yang, W., Le, H., Savarese, S., & Hoi, S. (2022). OmniXAI: A Library for Explainable AI. *arXiv Preprint*. https://doi.org/10.48550/arXiv.2206.01612