`gingado`: a machine learning library focused on economics and finance

Douglas K. G. Araujo[*]

This version: 16 June 2022
First version: 8 February 2022.

1 Introduction

Conceptually, every machine learning (ML) application entails the combination of a specific dataset, algorithm, cost function and optimisation procedure - and each of these components can be replaced more or less independently from the others (Goodfellow, Bengio, and Courville 2016). The set of possibilities is particularly wide in many economics and finance use cases (Athey and Imbens 2019, Mullainathan and Spiess 2017, Varian 2014, Doerr, Gambacorta, and Garralda 2021 Chakraborty and Joseph 2017). And there is often no definite answer as to which alternative is best (Mullainathan and Spiess 2017). Hence, in practice creating a ML application can amount to a fairly artisanal process that requires multiple iterations and attempted improvements until a result considered to be satisfactory is achieved. In fact, as of writing this paper different steps of the process of creating ML models in economics and finance could be more streamlined, ranging from selection and use of the dataset, comparison of different ML models, and crucially, also the model's documentation.

`gingado` is a free, open source library that seeks to facilitate the use of ML in economics and finance, while promoting good modelling practices.[1] It offers three main contributions. First, `gingado` facilitates data augmentation of the original user dataset with statistical series from official sources, in a way that is relevant for each case and testable on its ability to improve the model's performance. Second, `gingado` provides automatic benchmark models

[*]Economist, Bank for International Settlements. douglas.araujo@bis.org. The views expressed in this paper are those of the author and do not necessarily represent the views of the Bank for International Settlements. The author is thankful to Rodrigo Coelho, Ryan Defina, Emanuel Kohlscheen, Jermy Prenio, Bruno Sabóia and participants at the Irving Fisher Committee-Banca D'Italia workshop "Data science in central banking - Part 2: Data Science in Central Banking: Applications and tools" in February 2022 and at the Royal Statistical Society Finance and Economics Special Interest Group talk in May 2022. All errors and omissions are attributable to the author.

[1]`gingado`'s instructions, documentation, interactive notebooks and source code are available at `https://dkgaraujo.github.io/gingado/`.

that perform well on a broad set of situations and that can train quickly to achieve a reasonable if not the best performance for the dataset at hand; users can also make use of a generic benchmark object to create their own automatic benchmarks. And third, `gingado` promotes documentation of the ML model as part of the development work flow, automatising some documentation steps to leave users room for concentrating on more value-added documentation items such as ethical considerations. Also here `gingado` offers users the possibility to create their own model documentation templates that are easy to embed in the modelling practice.

Development of the `gingado` library follows three design principles:

1. **Flexibility**: users can use `gingado` out of the box or use it to build custom processes and objects that are more suitable for their use cases;

2. **Compatibility**: `gingado` works well with other widely used libraries in data science and machine learning; and

3. **Responsibility**: model documentation and ethical considerations are considered key steps in the modelling process and should be facilitated.

In addition, `gingado` seeks to be a parsimonious library that complements, rather than redoes, existing functionalities of other widely used libraries. `gingado`'s application programming interface (API) is compatible in particular with `scikit-learn` (Pedregosa et al. 2011) and it can generally be made compatible with other Python machine learning libraries with minimal adjustment. Also, it can be used in a modular way: users might prefer to just augment their datasets; create automatic benchmark models; or document their models.

These characteristics make `gingado` a useful tool across many domains in economics and finance. Machine learning algorithms are already amply used in economics for prediction problems (in the sense of Kleinberg et al. 2015), causal inference[2] (Chernozhukov et al. 2018 for example in randomised experiments), model estimation (L. Maliar, S. Maliar, and Winant 2021, Fernández-Villaverde, Hurtado, and Nuno 2019, Duarte 2018), estimation of models with non-traditional data (Ferreira et al. 2021) and even being the subject of study themselves (Fuster et al. 2022, Giannone, Lenza, and Primiceri 2021). `gingado`'s functionalities can be used as appropriate in each instance.

To showcase practical applications, the online documentation includes one end-to-end example: the automatic benchmark and model documentation functionalities are illustrated with Barro and Lee 1994's cross-country dataset with over 60 variables used to analyse drivers of GDP growth per capita.[3] The example illustrates one way in which `gingado` can help economists' workflow. The dataset was also recently used by Giannone, Lenza, and Primiceri 2021 to study the different predictive abilities of sparse vs dense models.

[2]A recent compilation of causal inference techniques from various domains is `https://neurips.cc/virtual/2021/workshop/21871`.

[3]Available at: `https://dkgaraujo.github.io/gingado/BarroLee1994.html`. Other examples will be added over time.

The next section describes how `gingado` facilitates data augmentation. Section 3 outlines the automatic benchmark process, and section 4 describes `gingado`'s model documentation framework. The last section concludes. The online documentation describes how to install the most up-to-date version of `gingado` and also more advanced topics like how to create customised automatic benchmark models and model documentation templates.

2 Data augmentation

Publicly available data have a long tradition in economics and finance research and practice. For example, the Federal Reserve Bank of St. Louis' Federal Reserve Economic Data (FRED)[4] system has developed in tandem with the internet itself (see Stierholz 2014 for an interesting narrative of FRED's history). Similarly, numerous other central banks and statistical agencies, as well as the international financial institutions, put datasets in the public domain in one form or the other. A number of statistics organisations created in the early 2000's an initiative to promote the collection, compilation and dissemination of statistical data, the Statistical Data and Metadata Exchange (SDMX), which is now in its version 3.0.[5] In addition, other data aggregators such as DBnomics[6] host an incredible amount of economic and financial series.

Many of these services allow users to access data in a programmatic way, ie setting up a computer programme to download the data instead of the user manually accessing the website, selecting the data, downloading it in a file and incorporating the data in the analyses. Thanks to SDMX and to the broader availability of user-friendly data application programming interfaces (APIs) like the one offered by FRED, querying data from trusted sources to augment the user's original dataset has become much easier than in the past. Accessing data programmatically also allows any numeric transformations, consistency checks and data imputation routines that are applied on the dataset to be done in a reproducible and transparent way.

In addition, programmatic access to data can also ensure that any newly added datasets are consistent with each other. For example, SDMX includes the concept of "codelists", which are standardised definitions that apply across dataset domains. For example, one specific codelist contains all possible realisations of the frequency of a dataset (ie, daily, weekly, monthly, ...), and another codelist encompasses all possible codes for specific currencies. This technology ensures that the user has greater control over the datasets to be incorporated.

The considerations above are more important hold when models are in the production stage. In economics and finance, ML models are occasionally designed to be run in production settings instead of a one-off execution; and this is often performed by users that were not involved during development and there-

[4]Available at: `https://fred.stlouisfed.org`.

[5]Available at: `https://sdmx.org`. A technical description of version 3.0 is found here: `https://sdmx.org/wp-content/uploads/SDMX_3-0-0_SECTION_1_FINAL-1_0.pdf`.

[6]Available at: `https://db.nomics.world`

fore are oblivious to the model's internal functioning. For example, a model forecasting stock prices will be used extensively over time - and by stock traders or portfolio managers, not the developers. These situations tend to take place in situations where the ML model will require future observations of the datasets used for training. Therefore, automating the process of data augmentation in a way that can ensure consistency between datasets helps to insulate users from worrying about these processes related to the use of additional data.

`gingado` aims to facilitate this process of finding and adding new datasets, leveraging the public availability of reliable data from official sources and automatically ensuring that the augmented dataset is consistent with the original, having the same frequency and time period. With this, when the model is run in the future, potentially by other people or automatically by systems, the same publicly available dataset(s) used during model training are downloaded and used each time with the appropriate time period. The current version of `gingado` achieves this with by means of its data augmentation object `AugmentSDMX`; the idea is to gradually add to the public codebase other "data augmenters" that can scan and download publicly available datasets in a way that is consistent with the original dataset.

2.1 Basic data augmentation process

Similar to other parts of `gingado`, the API for data augmentation is designed for compatibility with `scikit-learn`. Specifically, the user decides which specific sources and dataflows will be scanned for data upon instantiation of the data augmenter object. So far, the object does not perform any activity other than store this and other parameters. It is only when one of the methods `fit`, `transform`, or `fit_transform` are called that the data augmenter will (a) read characteristics of the data and (b) look in the named sources and dataflows what would be adequate datasets corresponding to the same frequency and time period. This process is shown in listing 1, where the data augmenter will look for the European Central Bank's Composite indicator of systemic stress (CISS) and the Bank for International Settlement's dataflow on central bank policy rates.

```python
from gingado.augmentation import AugmentSDMX

src = {'ECB': ['CISS'], 'BIS': ['WS_CBPOL_D']}
AugmentSDMX(sources=src).fit_transform(X_train)
```

Listing 1: Basic example of the `AugmentSDMX` API

The process described above will result in a dataset that contains the original data, but is now augmented by potentially numerous other series. The data is indexed by time, and for this reason every combination of permissible codelists in the augmented dataset will create a different variable (ie, a different column).

For example, even if the original dataset represents only data from Brazil, the augmentation process shown in 1 will append the CISS for every country in the list as a different column, as well as the central bank policy rates. `gingado` makes an explicit choice to allow for this, instead of filtering by individual (in this example, retrieving only the dataset about Brazil), because the newly added data can have some level of information on the target variable, and if that is the case the model would probably uncover it.

2.2 Data augmentation with SDMX

`gingado` explores and fetches available datasets from official sources to augment user data using SDMX, an initiative by international organisations (Bank for International Settlements - BIS, European Central Bank - ECB, Organisation for Economic Cooperation and Development - OECD, International Monetary Fund - IMF, World Bank - WB, Eurostat, and United Nations- UN) that develop and publish statistics from these sources. Since its inception in 2001, SDMX has grown to be used by other sources as well - primarily central banks and statistics agencies - as a standard to disseminate their data.

Downloading data from SDMX sources can be advantageous to users because of the variety of sources and the consistency of the concepts describing the datasets. Instead of dealing with the specific data descriptions and download processes of each of the SDMX participant institutions, the user can rely on an API based on standardised concepts to fetch the data. For example, using SDMX to look across multiple sources for data of quarterly frequency related to the countries of Argentina, Brazil and South Africa for the period spanning the first quarter of 2015 to the fourth quarter of 2021 would use the codelists (introduced above) for frequency and for reference area. These are the same across the SDMX sources and dataflows, which facilitates the process of finding relevant datasets.

`AugmentSDMX` is based on the `pandasdmx` python package. The backend code searches all listed SDMX sources in this package, and retrieves the dataflows for those it is able to get (some sources may timeout). Given that downloading all the relevant data from all sources could be an expensive operation, users define the SDMX source(s) from which to get the data, as well as the specific dataflows if they wish.

As of the time of writing, the data providers available for automatic download of data using `AugmentSDMX` are:[7]

- Australian Bureau of Statistics
- Bundesbank
- Bank for International Settlements
- Countdown 2030

[7]Users can get the up-to-date list with the `gingado` method `list_SDMX_sources` in `gingado.utils`.

- European Central Bank
- Eurostat
- International Labour Organization
- Internationa Monetary Fund
- National Institute of Statistics and Geography (Mexico)
- National Institute of Statistics and Economic Studies (France)
- National Institute of Statistics (Italy)
- National Institute of Statistics (Lithuania)
- Norges Bank (Norway)
- National Bank of Belgium
- Organisation for Economic Cooperation and Development
- Pacific Data Hub
- Statistics Estonia
- United Nations Statistics Division
- UN International Children's Emergency Fund (UNICEF)
- World Bank Group's "World Integrated Trade Solution"
- World Bank Group's "World Development Indicators"

Each of those sources offers a number of dataflows, which are closely related datasets. For example, one dataflow related to foreign exchange rates could include the time series of multiple individual exchange rate pairs, and each pair can be downloaded in their nominal or real exchange rate. The dataflows from all of these sources could be available to train the ML model at hand. In total, the sources listed above result in 9,110 dataflows.

One potential drawback of the process is that the computation time might increase as the number of SDMX series are added. This is an important consideration to have, as production version models will also need to incorporate this data - depending on how expensive these transactions are, it could affect. However, for cases where immediate response is not required, the quantity of data could be considered reasonable.

2.3 Data augmentation with other datasets

There are many other official agencies that offer freely (and easily) accessible datasets that would certainly be useful to the proposed data augmentation. For example, the Federal Reserve Bank of St Louis maintains the Federal Reserve Economic Data (FRED)[8] and related APIs, and the Brazilian Institute for Geography and Statistics (IBGE) has a range of APIs[9] that offer datasets that could be of use. It would be useful to also be able to use these tools programatically.

While `gingado` doesn't include these sources above off-the-shelf as automatic augmenters, work on the next versions will concentrate on more flexibly adding data augmenters and crucially, offering users the possibility to do the same. The code already allows for that, for the users willing to adjust the class `AugmentSDMX` but it is not yet an intuitive process.

2.4 Is it worth adding more and more data?

While an increasing amount of data can lead to better results by machine learning models, there are situations where users might want to consider limiting the amount of data being fed to a model. Therefore, the answer to this subsection's title will depend on each use case, and `gingado` can help the user answer it.

`AugmentSDMX`'s API is compatible with `scikit-learn` in a specific way that allows it to be included in a *pipeline*. Pipelines are objects that apply on data a specific sequence of transformations, and possibly a final step consisting of an estimator (such as a predictor). These pipelines exist to allow the cross-validation of the sequence of steps as a whole, and to allow for a consistent way to estimate different versions of the same model without losing control of the data pipeline.

What this means in practice is that the user can apply standard parameter search algorithms such as grid search to test whether or not the inclusion of a particular dataset will impact the model, eg by improving its performance, changing the importance of regressors, etc. This process is exemplified in listing 2: the user created with a few lines a model that, when fitted, will estimate the model without augmentation (ie, will "pass through" `AugmentSDMX`) and with augmentation, in this case using all dataflows made available by the BIS. Beyond that, the user can even jointly search a combination of different parameters governing data augmentation, data transformation and model estimation by combining `AugmentSDMX` with `scikit-learn`'s `Pipeline`.

3 Automatic benchmark

`gingado` offers users the possibility of automatically developing a benchmark machine learning model fine-tuned for their particular case, in a short time and

[8]`https://fred.stlouisfed.org`

[9]`https://servicodados.ibge.gov.br/api/docs/`

```python
from gingado.augmentation import AugmentSDMX
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV
from sklearn.pipeline import Pipeline

pipeline = Pipeline([
    ('augmentation', AugmentSDMX(sources={'BIS': 'all'})),
    ('forest', RandomForestRegressor())
])

grid = GridSearchCV(
    estimator=pipeline,
    param_grid={
        'augmentation': ['passthrough', AugmentSDMX(sources={'BIS': 'all'})]
    })
```

Listing 2: Use of `AugmentSDMX` in a `scikit-learn` pipeline

with no input from the user other than the data (although users can also fine tune all aspects of the benchmark). This is achieved by means of an embedded parameter grid search mechanism that evaluates different versions of the underlying algorithm on the user's data, and selects that one that performs better as the benchmark.

The objective is to help the user during the exploratory phase of the development of a machine learning model. The practice of establishing a baseline model is common in machine learning practice. Without a goalpost, a baseline model that is relatively simple to understand and benchmark against, it can be difficult in practice to realise if one's model is performing well or just improving from a low base. So `gingado` allows users to quickly create a fully functioning model that can serve as benchmark, as shown in listing 3.

```python
from gingado.benchmark import ClassificationBenchmark

benchmark = ClassificationBenchmark().fit(X_train, y_train)

y_pred = ClassificationBenchmark().predict(X_test)
```

Listing 3: Creation of an automatic benchmark model

The off-the-shelf implementations of this automatic benchmark model are based on random forests (Breiman 1996, Breiman 2001), one type for regression tasks and another one for classification. Random forests, one of the most widely used machine learning methods according to industry practitioners (J. Howard and S. Gugger 2020) present several advantages that make them good candidates for benchmark models. They tend to have a very good out-of-sample fit, and require little data preparation work compared to other machine learning models. Random forests also provides an intuitive way to measure the individual importance of regressors, although they don't lend themselves to interpreting the channels by which the regressors contribute to the prediction (Varian 2014). These regressor importance measures are the mean reduction in impurity occurring in the trees that use that particular variable. The values are then typically scaled so that the sum of all feature importance measures is always one. Practitioners use this measurement as one possibility for variable selection (Géron 2019, Kohlscheen 2021).

Whenever a benchmark model is fitted to the data, the model automatically creates a documentation. In the off-the-shelf implementations, this documentation is done via the `ModelCard` object, described in more detail in section 4. From then on, the model documentation can be accessed - and most importantly, filled out, from that object.

Benchmark models also have a specific method to compare themselves with other candidate models. This allows users to directly compare their own candidate models with the existing benchmark. When this is done, via the module `compare`, `gingado` also includes amongst the candidates to be tested an ensemble combination of all the candidates and the current benchmark. The inclusion of this candidate ensembles serves to leverage the advantage that ensemble models have over simpler models (Giannone, Lenza, and Primiceri 2021).

3.1 Other use cases for a benchmark model

In addition to serving as a baseline model during the development of machine learning, users can simply retain the automatic benchmark as their production model. Beyond benchmark-specific functionalities, these objects behave also as normal `scikit-learn` models and therefore can be applied as any normal model would.

Benchmarks can be used as a test to see what if any regressors differentiate the most between two or more groups, for example to see if one group of samples has out-of-domain data (as proposed by J. Howard and S. Gugger 2020). The test proceed as follows: a random forest classifier is trained on the dataset, with group identifiers serving as the target variable. The forest's calculated regressor importance can then uncover what are the variables that differentiate between groups. This test can be generalised to check what regressors, if any, vary substantially between two or more groups.

3.2 Custom automatic benchmarks

In spite of the empirical qualities of random forests, no single algorithm could plausibly cover all potential use cases. For example, random forests could potentially not perform as well as other established algorithms if the economic or financial data at hand contains multimodal data, ie images, videos, texts and other such data. In addition, random forests cannot extrapolate beyond the range of the training data that was fed to it. And there are some empirical settings in which gradient boosting trees are more used than random forests and other machine learning methods (Brotcke 2022). Similarly, Gorishniy et al. 2021 show that neural networks can achieve similar, and in some cases better, performance than tree-based methods in some cases. And Taylor and Letham 2018 describe Facebook's own tool (now open sourced) for automatic forecasting, without relying on random forests.

So random forests might not be the first choice for all cases, even though they are expected to work well in a wide array of situations. In addition to the off-the-shelf implementations described above, `gingado` offers two possibilities for users to set up their own benchmark models. The most straightforward one is to simply pass as argument a model object to the benchmark's method `set_benchmark`. This will cause the benchmark object to put this new model in place of the previous benchmark.

The second object involves the creation of a new benchmark object class altogether. `gingado` ships with a base class, `ggdBenchmark`, that contains all the necessary features for a benchmarker object. Users that wish to create their own benchmark models can sub-class from `ggdBenchmark` and implement the desired funcionalities. The user's benchmark will then work in the same way as the original `gingado` benchmark models. This user-created benchmark can include any algorithm or combination of algorithms, as long as the API is maintained.

4 Model documentation

Stakeholders often require some level of documentation of the machine learning models. From simple descriptions of the model to standardised model reports to fully-fledged evaluation of models, `gingado` provides a way for models to be more easily documented. The basic idea is that a *documentation template* outlines the specific items to be documented, and various methods allow the user to interact with this template. This template can be seen at any time by the user with the method `show_template`.A `gingado` documenter object also specifies which of those items are to be filled in automatically (eg, model description can be parsed from the machine learning algorithm itself), and how exactly this should be done.

After a documenter is instantiated, it can read information from an existing model as well. Listing 4 demonstrates how straightforward it is to automatically read information from a model, and then see what are the questions from the documentation template that were not answered automatically by the `ModelCard` object. Note that the model itself is not a `gingado` object.[10]

At any time, the user can view the current state of the document with the

[10]Currently the base documenter `ggdModelDocumentation`, from which all documenters in this library derive, can read models created using `gingado, scikit-learn and keras. Support for automatically reading information from models built with other libraries such as PyTorch is under development.`

```python
from gingado.model_documentation import ModelCard
from tensorflow import keras

keras_clf = keras.Sequential()
keras_clf.add(keras.layers.Dense(16, activation='relu', input_shape=(20,)))
keras_clf.add(keras.layers.Dense(8, activation='relu'))
keras_clf.add(keras.layers.Dense(1, activation='sigmoid'))
keras_clf.compile(optimizer='sgd', loss='binary_crossentropy')
keras_clf.fit(X, y, batch_size=10, epochs=10)

model_doc_keras = ModelCard()
model_doc_keras.read_model(keras_clf)
model_doc_keras.open_questions()
```

Listing 4: Example of `ModelCard` reading information from an existing neural network model

method `show_json`, and save or read it to file with `save_json` and `read_json` respectively. Additional information items that were not included automatically are filled with `fill_info` (the user can also override automatic entries). And the remaining items from the template that are not yet filled are shown to the user with the method `open_questions`. These methods (and others, not shown for brevity) aim to provide the user with a more direct, hands-on approach to documenting the model, compared to a more traditional setting where a separate document (ie, a MS Word file) need to be written and maintained. Hopefully allowing users to document their models from inside their machine learning development environment will help embed the documentation process as another step of the model development. Another advantage of `gingado`'s approach is that it is much easier to keep the documentation aligned with the current version of the model. This is particularly important in the settings where the user expects to iterate over different specifications until a suitable model is achieved.

The model documentation is stored as a JSON file, a flexible format that is easy for machines to read, and that can quickly be transformed into objects humans can read more easily, too. `gingado` uses JSON files because they are a common language to serialise this type of structured information that works across platforms (ie, a JSON file works in Windows machines the same way it works in MacOS, Linux or any other system). Users that want to automatically produce documents in other formats (eg, PDF files) can do so by using these JSON files with other libraries of their choice. And in settings where multiple models are developed and in production, JSON files are a more streamlined format to offer information on each model to comparison scripts.

`gingado` includes a ready-to-use documenter object, `ModelCard`. This documentation template is based on the work of Mitchell et al. 2018, which I believe strikes a good balance between being a general documentatio template while also prompting the user to answer questions about the model that are relevant from a broader stakeholder perspective. Similar to how users can create their own class of benchmark models, `gingado` enables users to create their own custom documenters, from the base class `ggdModelDocumentation`. This allows specific documentation templates to be used in a more automatic way, and can be of particular importance in the context of organisations using machine learning, since they might have their own documentation preferences.

4.1 Ethical issues in economics and finance machine learning models

One of the reasons why `gingado` facilitates model documentation is to promote a greater role for ethical considerations as part of the development of machine learning models in economics and finance: if the model documentation becomes part of development workflow and certain parts of the documentation are automated, then users are presumably more likely to consider these issues at development time, not *ex post*.

There seems to be a wide awareness of the implications from biased datasets feeding into complex, black-box machine learning models in economics and finance.[11] They are illustrated for example by Brotcke 2022, who present a practitioner view on using machine learning while seeking to originate loans without bias, and by Doerr, Gambacorta, and Garralda 2021, who describe central banks' concerns with fairness implications from greater use of machine learning. But the ethical implications of ML do not stem just from the datasets used for training. It is likely that similar issues are important in a variety of uses, in varying degrees. For example, while Bender et al. 2021 highlight ethical and societal issues stemming from the incredibly large size of datasets used to train large scale language models.

But Bender et al. 2021 also discuss model-related aspects: eg, the environmental impact of the large-size architecture of these models, and the risks originating from the higher (apparent) fluency of these models. Mullainathan and Obermeyer 2017 and Thomas and Uminsky 2022 discuss pitfalls and risks from the metrics chosen during machine learning development. Therefore, promoting opportunities for machine learning developers to think about the ethical implications from the complete model pipeline, from data to application, seems more than warranted. One recent evidence of the real-life impact of machine learning applications might materially impact stakeholder outcomes is studied by Fuster et al. 2022, in the context of mortgage lending.

[11] A related discussion where ethics and fairness in machine learning has made strides but needs more progress is related to explainable artificial intelligence (XAI) (for example, Barredo Arrieta et al. 2020).

5 Conclusion

`gingado` is a free, open source machine learning library focused on use cases in economics and finance - in both research and practice. Its compatibility with widely used machine learning frameworks, most notably `scikit-learn`, allows it to leverage on the wide familiarity with these tools and complement existing user codebases. And its flexible approach simultaneously affords users the ability to advance machine learning model development with few steps, while enabling users to tweak all tools to meet their goals, including adjusting models and their outcomes to better suit econometric use cases when necessary (Athey and Imbens 2019). The toolset provided by `gingado` was first created for my own use as a practitioner of machine learning in economics, and I expect to continue developing it over time to incorporate funcionalities that can be of most value-added to researchers and practitioners in economics and finance. At the same time, I hope that `gingado` can contribute to facilitate greater use of machine learning in economic and finance, while promoting good modelling practices including a greater role for model documentation and ethical considerations. The particular areas focused by `gingado` are data augmentation, automatic benchmark models, and model documentation.

Adding more data to one's dataset can often be cumbersome from an operational perspective, and especially so when multiple sources are involved. This is of course much harder when the machine learning model is used in a production setting, instead of a one-off analysis. `gingado` addresses this by augmenting the user dataset through an object that fits nicely in standard machine learning pipelines. This also allows the user to test whether or not adding more data actually improves the model (according to any criteria defined by the user). In addition, `gingado`'s data augmentation method focuses on ensuring that the data provided to users is from trusted sources. When more data augmentation objects are added to the `gingado` codebase, the reliability of data sources will be a key criterion.

Automatic benchmark models are not new. Jeremy Howard and Sylvain Gugger 2020 and Taylor and Letham 2018 for example enable users to quickly set up models with a reasonably good performance for the vast majority of use cases in their respective domains. `gingado` builds on that insight and provides users with additional functionalities that to my knowledge are novel. Namely, a way to conveniently compare candidate models (and their ensemble) and pick the best one as the new benchmark, the automatic documentation of the benchmark model, and the ability to create one's own benchmark from a base class aim to push standard practice of how to set up baseline models.

And finally, model documentation. Mullainathan and Spiess 2017 mention the risk that machine learning models in economics and finance be applied naïvely or have their outputs misinterpreted. This risk increases with greater deployment of machine learning in important aspects of daily life, such as banking[12]. But the risk probably also grows as the technical preconditions for ma-

[12]For example, in March 2022 the Basel Committee on Banking Supervision affirmed that

chine learning, such as greater availability of higher-dimensional datasets and of compute power, facilitate model development by more people. As the popularity of machine learning models amongst economists grow, my goal with `gingado` is to contribute a small part to embed model documentation in the process of model development, automating some of the questions to afford the humans in control the opportunity to properly document their models and pay due consideration to ethical issues arising in each particular situation.

References

Athey, Susan and Guido W. Imbens (2019). "Machine Learning Methods That Economists Should Know About". In: *Annual Review of Economics* 11.1, pp. 685–725. DOI: `10.1146/annurev-economics-080217-053433`. eprint: `https://doi.org/10.1146/annurev-economics-080217-053433`. URL: `https://doi.org/10.1146/annurev-economics-080217-053433`.

Barredo Arrieta, Alejandro et al. (2020). "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI". In: *Information Fusion* 58, pp. 82–115. ISSN: 1566-2535. DOI: `https://doi.org/10.1016/j.inffus.2019.12.012`. URL: `https://www.sciencedirect.com/science/article/pii/S1566253519308103`.

Barro, Robert J. and Jong-Wha Lee (1994). "Sources of economic growth". In: *Carnegie-Rochester Conference Series on Public Policy* 40, pp. 1–46. ISSN: 0167-2231. DOI: `https://doi.org/10.1016/0167-2231(94)90002-7`. URL: `https://www.sciencedirect.com/science/article/pii/0167223194900027`.

Bender, Emily M. et al. (2021). "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. FAccT '21. Virtual Event, Canada: Association for Computing Machinery, pp. 610–623. ISBN: 9781450383097. DOI: `10.1145/3442188.3445922`. URL: `https://doi.org/10.1145/3442188.3445922`.

Breiman, Leo (1996). "Bagging predictors". In: *Machine learning* 24.2, pp. 123–140.

— (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.

Brotcke, Liming (2022). "Time to Assess Bias in Machine Learning Models for Credit Decisions". In: *Journal of Risk and Financial Management* 15.4. ISSN: 1911-8074. DOI: `10.3390/jrfm15040165`. URL: `https://www.mdpi.com/1911-8074/15/4/165`.

Chakraborty, Chiranjit and Andreas Joseph (2017). "Machine learning at central banks". In.

Chernozhukov, Victor et al. (2018). *Generic machine learning inference on heterogeneous treatment effects in randomized experiments, with an application to immunization in India.* Tech. rep. National Bureau of Economic Research.

"*Banks are increasingly exploring opportunities for using artificial intelligence, including machine learning*" (available at: `https://www.bis.org/publ/bcbs_nl27.htm`).

Doerr, Sebastian, Leonardo Gambacorta, and José Maria Serena Garralda (2021). “Big data and machine learning in central banking”. In: *BIS Working Papers* 930.

Duarte, Victor (2018). “Machine learning for continuous-time economics”. In: *Available at SSRN 3012602*.

Fernández-Villaverde, Jesús, Samuel Hurtado, and Galo Nuno (2019). *Financial frictions and the wealth distribution.* Tech. rep. National Bureau of Economic Research.

Ferreira, Leonardo N et al. (2021). *Forecasting with VAR-teXt and DFM-teXt Models: exploring the predictive power of central bank communication.* Tech. rep.

Fuster, Andreas et al. (2022). “Predictably Unequal? The Effects of Machine Learning on Credit Markets”. In: *The Journal of Finance* 77.1, pp. 5–47. DOI: `https://doi.org/10.1111/jofi.13090`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1111/jofi.13090`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1111/jofi.13090`.

Géron, Aurélien (2019). *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems, 2nd edition.* Sebastopol, CA: O’Reilly Media. ISBN: 978-1491962299.

Giannone, Domenico, Michele Lenza, and Giorgio E Primiceri (2021). “Economic predictions with big data: The illusion of sparsity”. In: *Econometrica* 89.5, pp. 2409–2437.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning.* http://www.deeplearningbook.org. MIT Press.

Gorishniy, Yury et al. (2021). *Revisiting Deep Learning Models for Tabular Data.* arXiv: `2106.11959 [cs.LG]`. URL: `https://proceedings.neurips.cc/paper/2021/hash/9d86d83f925f2149e9edb0ac3b49229c-Abstract.html`.

Howard, J. and S. Gugger (2020). *Deep Learning for Coders with Fastai and Pytorch: AI Applications Without a PhD.* O’Reilly Media, Incorporated. ISBN: 9781492045526. URL: `https://books.google.no/books?id=xd6LxgEACAAJ`.

Howard, Jeremy and Sylvain Gugger (2020). “Fastai: A Layered API for Deep Learning”. In: *Information* 11.2. ISSN: 2078-2489. DOI: `10.3390/info11020108`. URL: `https://www.mdpi.com/2078-2489/11/2/108`.

Kleinberg, Jon et al. (May 2015). “Prediction Policy Problems”. In: *American Economic Review* 105.5, pp. 491–95. DOI: `10.1257/aer.p20151023`. URL: `https://www.aeaweb.org/articles?id=10.1257/aer.p20151023`.

Kohlscheen, Emanuel (2021). “What does machine learning say about the drivers of inflation?” In: *Available at SSRN 3949352*.

Maliar, Lilia, Serguei Maliar, and Pablo Winant (2021). “Deep learning for solving dynamic economic models.” In: *Journal of Monetary Economics* 122, pp. 76–101.

Mitchell, Margaret et al. (2018). “Model Cards for Model Reporting”. In: *CoRR* abs/1810.03993. arXiv: `1810.03993`. URL: `http://arxiv.org/abs/1810.03993`.

Mullainathan, Sendhil and Ziad Obermeyer (May 2017). “Does Machine Learning Automate Moral Hazard and Error?” In: *American Economic Review*

107.5, pp. 476–80. DOI: `10.1257/aer.p20171084`. URL: `https://www.aeaweb.org/articles?id=10.1257/aer.p20171084`.

Mullainathan, Sendhil and Jann Spiess (May 2017). "Machine Learning: An Applied Econometric Approach". In: *Journal of Economic Perspectives* 31.2, pp. 87–106. DOI: `10.1257/jep.31.2.87`. URL: `https://www.aeaweb.org/articles?id=10.1257/jep.31.2.87`.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Stierholz, Katrina (2014). "FRED®, the St. Louis Fed's force of data". In: *Review* 96.2, pp. 195–198. URL: `https://ideas.repec.org/a/fip/fedlrv/00023.html`.

Taylor, Sean J and Benjamin Letham (2018). "Forecasting at scale". In: *The American Statistician* 72.1, pp. 37–45.

Thomas, Rachel L and David Uminsky (2022). "Reliance on metrics is a fundamental challenge for AI". In: *Patterns* 3.5, p. 100476.

Varian, Hal R (2014). "Big data: New tricks for econometrics". In: *Journal of Economic Perspectives* 28.2, pp. 3–28.