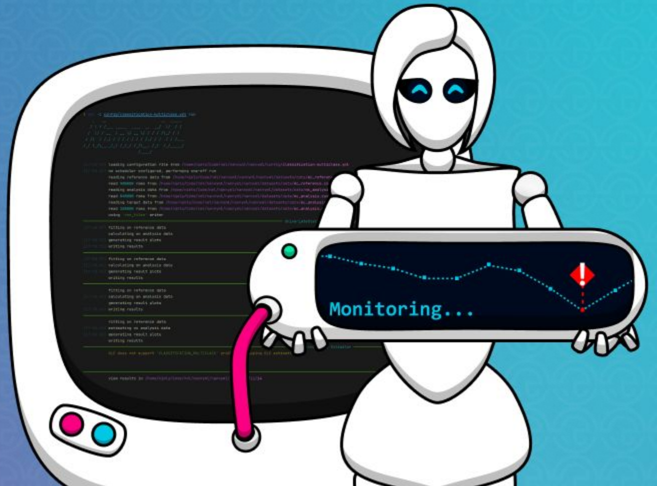
Notebooks to containers:

Setting up continuous ML model monitoring in production

Webinar with
Niels Nuyttens



Welcome! 😃

If you want to follow along:

https://github.com/NannyML/examples/tree/main/webinars

Feel free to post your questions in the Q&A section

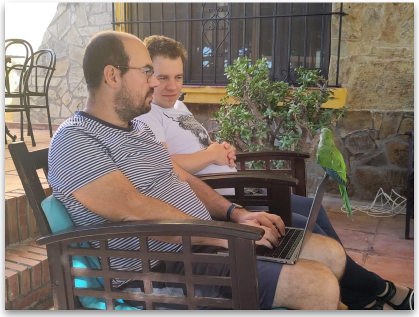
Hi! 👋

I'm Niels.

I've worked as a consultant for about a decade (shit I'm old) in software engineering, data engineering and DevOps tracks.

I joined NannyML in April 2021. I'm currently the lead engineer, because there are no other engineers.

I'm responsible for the architecture and implementation of our library, supporting tools and the exciting stuff we're about to build!



Just picking up some birds

Infecting others with my poor choice in clothing

(I had nothing to do with socks in sandals though)

nannyML

We're definitely a company! 🏢

Founded by Hakim, Wojtek and Wiljan in 2020.

The idea for a product grew from ML/AI consulting experiences.

Got VC backing.

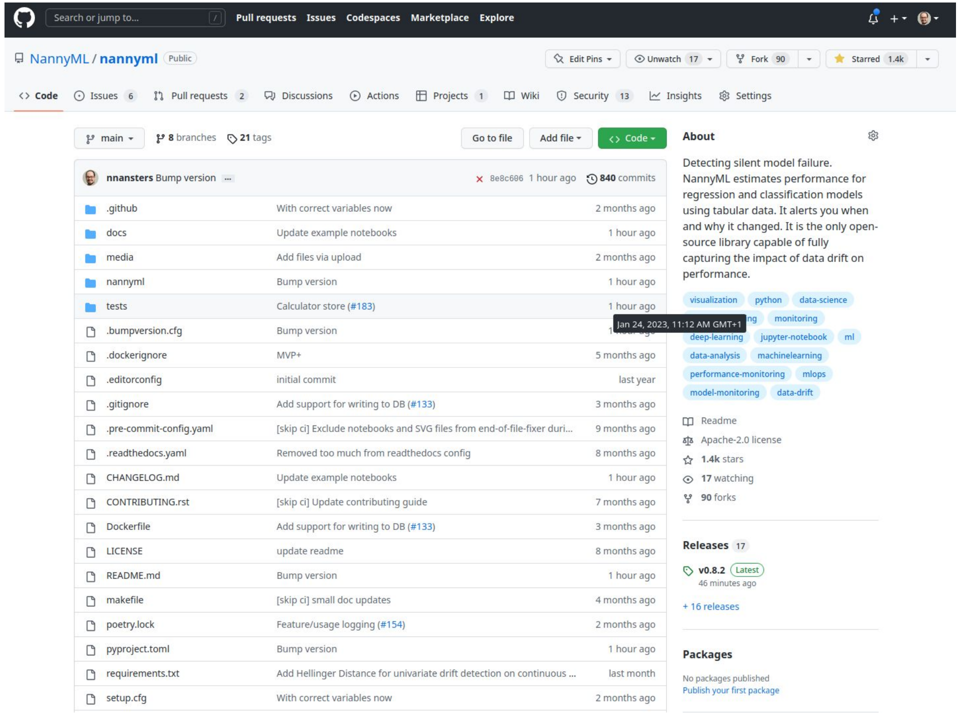
Grew the team with researchers, data scientists, operations and growth.

Released the open-source NannyML library earlier this year.

We're trying to make the world a better place 🌈 🦄

The NannyML library

- Calculate realized performance when target data is available
- Estimate performance when target data is not available
- Detect multivariate drift using data reconstruction error
- Detect univariate drift using KS, Chi2, Jensen-Shannon, EMD, Wasserstein, L-Infinity, …
- Calculate correlation between drift and performance using the Ranker
- Currently supporting classification and regression use cases on tabular data
- Plotting functionality
- Read from / write to local and cloud storage



nnansters Bump version … 8e8c606 1 hour ago 840 commits

.github	With correct variables now	2 months ago
docs	Update example notebooks	1 hour ago
media	Add files via upload	2 months ago
nannyml	Bump version	1 hour ago
tests	Calculator store (#183)	1 hour ago
.bumpversion.cfg	Bump version	1 hour ago
.dockerignore	MVP+	5 months ago
.editorconfig	initial commit	last year
.gitignore	Add support for writing to DB (#133)	3 months ago
.pre-commit-config.yaml	[skip ci] Exclude notebooks and SVG files from end-of-file-fixer duri…	9 months ago
.readthedocs.yaml	Removed too much from readthedocs config	8 months ago
CHANGELOG.md	Update example notebooks	1 hour ago
CONTRIBUTING.rst	[skip ci] Update contributing guide	7 months ago
Dockerfile	Add support for writing to DB (#133)	3 months ago
LICENSE	update readme	8 months ago
README.md	Bump version	1 hour ago
makefile	[skip ci] small doc updates	4 months ago
poetry.lock	Feature/usage logging (#154)	2 months ago
pyproject.toml	Bump version	1 hour ago
requirements.txt	Add Hellinger Distance for univariate drift detection on continuous …	last month
setup.cfg	With correct variables now	2 months ago

Jan 24, 2023, 11:12 AM GMT+1

About

Detecting silent model failure. NannyML estimates performance for regression and classification models using tabular data. It alerts you when and why it changed. It is the only open-source library capable of fully capturing the impact of data drift on performance.

visualization python data-science …ng monitoring deep-learning jupyter-notebook ml data-analysis machinelearning performance-monitoring mlops model-monitoring data-drift

Readme
Apache-2.0 license
1.4k stars
17 watching
90 forks

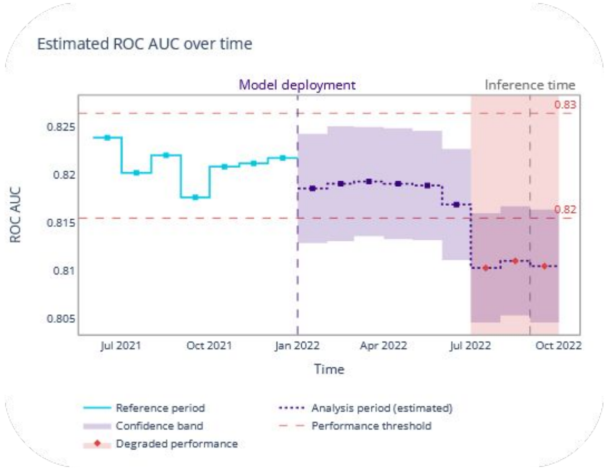
Releases 17

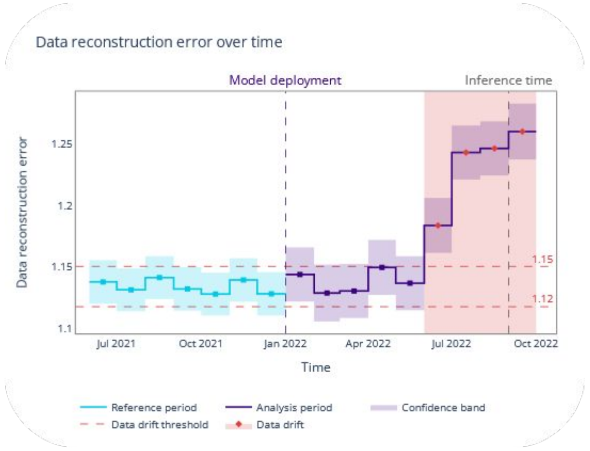
v0.8.2 Latest
46 minutes ago
+ 16 releases

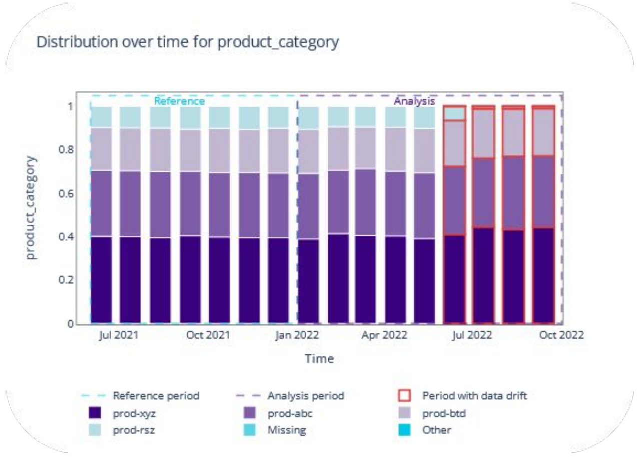
Packages

No packages published
Publish your first package

We obsess about performance (impact)







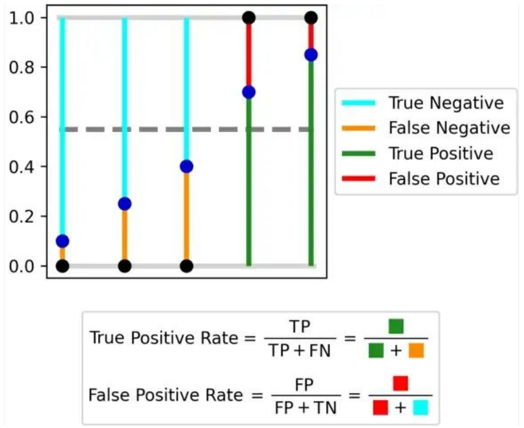
We only care when performance is impacted.

Look at multivariate shift to identify more complex covariate shift patterns.

Look at univariate covariate shift to find the culprits!

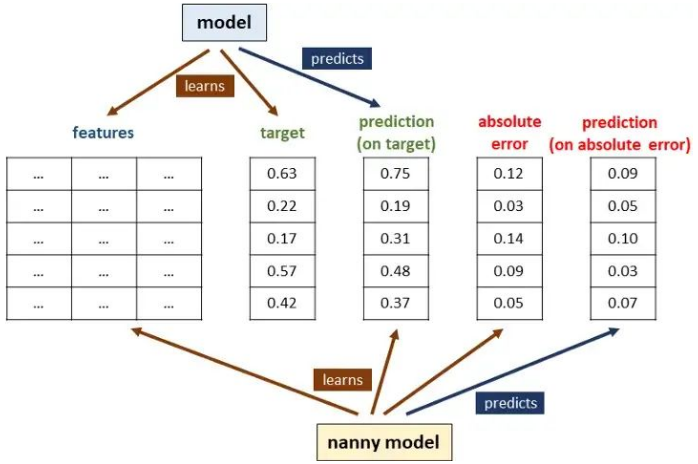
nannyML

Performance without ground truth?



$$\text{True Positive Rate} = \frac{TP}{TP + FN} = \frac{🟩}{🟩 + 🟧}$$

$$\text{False Positive Rate} = \frac{FP}{FP + TN} = \frac{🟥}{🟥 + 🟦}$$



features			target	prediction (on target)	absolute error	prediction (on absolute error)
...	...	...	0.63	0.75	0.12	0.09
...	...	...	0.22	0.19	0.03	0.05
...	...	...	0.17	0.31	0.14	0.10
...	...	...	0.57	0.48	0.09	0.03
...	...	...	0.42	0.37	0.05	0.07

Introducing **Confidence Based Performance Estimation (CBPE)** to estimate performance metrics for classification use cases

Introducing **Direct Loss Estimation (DLE)** to estimate performance metrics for regression use cases

NannyML run modes

Exploration mode

- Use NannyML as a library
- Interactive, using a notebook
- Run once, maybe repeat with low frequency
- Typically analyse a single large dataset (spanning a period of weeks, months or years)
- Result: time series of metric values (generated all at once)

Production mode

- Using NannyML as a CLI tool or container
- Run automated, configuration based
- Run repeatedly, as high frequency as the data volume allows
- Typically analysing multiple smaller datasets (spanning hours to days)
- Result: time series of metric values (generated sequentially)

MLOps CMM levels

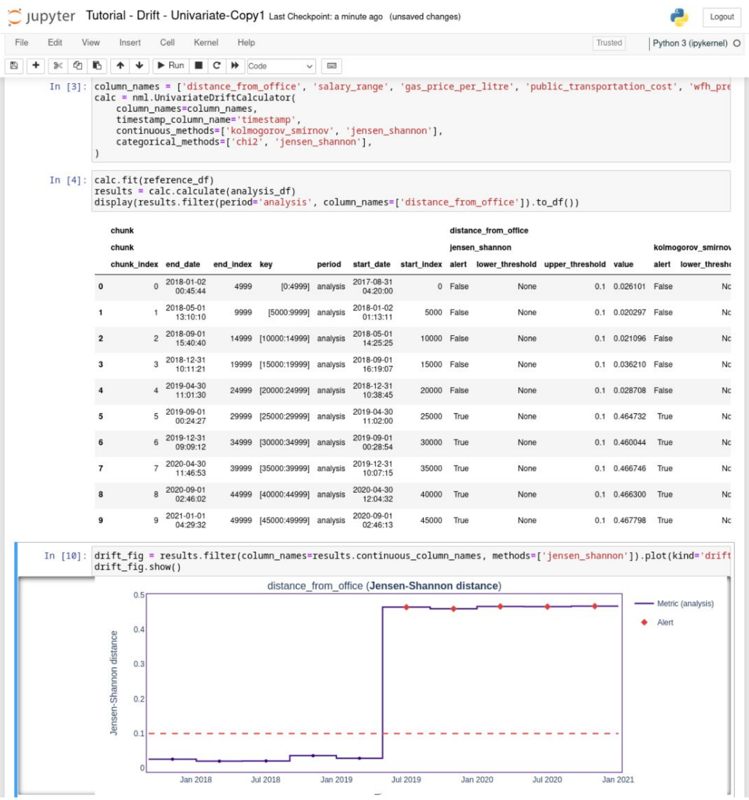
Manual	Repeatable	Reproducible	Automated	Improving
Level 1	Level 2	Level 3	Level 4	Level 5
Model development and deployment is fully manual and has limited documentation or tracking	Repeatable model development means others can repeat a documented process, which brings improved consistency and quality to the result	Fully reproducible model development means the result itself is exactly reproducible, which enables both efficient collaboration as well as low-effort maintenance	Automated model development and deployment brings increased efficiency for the AI team and organization	Model development, model performance, and maintenance is optimized to bring continuous improvement

MLOps CMM levels

Manual	Repeatable	Reproducible	Automated	Improving
Level 1	Level 2	Level 3	Level 4	Level 5
Model development and deployment is **fully manual** and has limited documentation or tracking	Repeatable model development means **others can repeat a documented process**, which brings improved consistency and quality to the result	Fully reproducible model development means the **result itself is exactly reproducible**, which enables both efficient collaboration as well as low-effort maintenance	**Automated model development and deployment** brings increased efficiency for the AI team and organization	Model development, model performance, and maintenance is optimized to bring **continuous improvement**

Check out https://radix.ai/

Level 1 - Manual


jupyter Tutorial - Drift - Univariate-Copy1 Last Checkpoint: a minute ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3 (ipykernel)
Run
Code

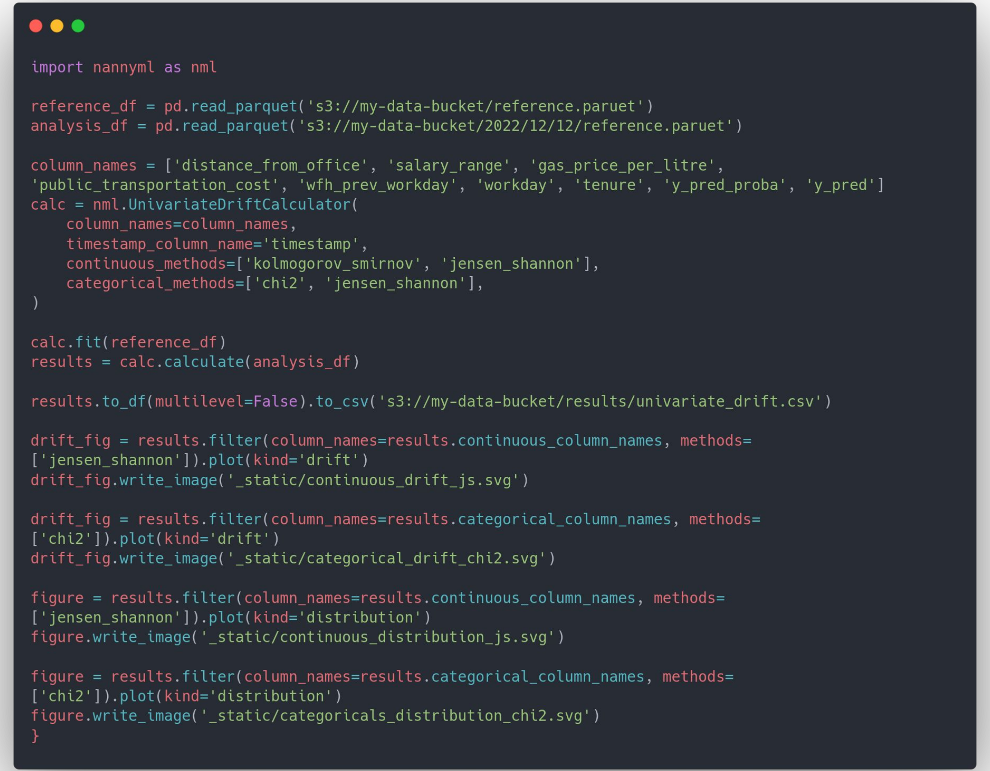
```
In [3]: column_names = ['distance_from_office', 'salary_range', 'gas_price_per_litre', 'public_transportation_cost', 'wfh_pre
        calc = nml.UnivariateDriftCalculator(
            column_names=column_names,
            timestamp_column_name='timestamp',
            continuous_methods=['kolmogorov_smirnov', 'jensen_shannon'],
            categorical_methods=['chi2', 'jensen_shannon'],
        )
```

```
In [4]: calc.fit(reference_df)
        results = calc.calculate(analysis_df)
        display(results.filter(period='analysis', column_names=['distance_from_office']).to_df())
```

	chunk							distance_from_office					
	chunk							jensen_shannon				kolmogorov_smirnov	
	chunk_index	end_date	end_index	key	period	start_date	start_index	alert	lower_threshold	upper_threshold	value	alert	lower_thresh
0	0	2018-01-02 00:45:44	4999	[0:4999]	analysis	2017-08-31 04:20:00	0	False	None	0.1	0.026101	False	Nc
1	1	2018-05-01 13:10:10	9999	[5000:9999]	analysis	2018-01-02 01:13:11	5000	False	None	0.1	0.020297	False	Nc
2	2	2018-09-01 15:40:40	14999	[10000:14999]	analysis	2018-05-01 14:25:25	10000	False	None	0.1	0.021096	False	Nc
3	3	2018-12-31 10:11:21	19999	[15000:19999]	analysis	2018-09-01 16:19:07	15000	False	None	0.1	0.036210	False	Nc
4	4	2019-04-30 11:01:30	24999	[20000:24999]	analysis	2018-12-31 10:38:45	20000	False	None	0.1	0.028708	False	Nc
5	5	2019-09-01 00:24:27	29999	[25000:29999]	analysis	2019-04-30 11:02:00	25000	True	None	0.1	0.464732	True	Nc
6	6	2019-12-31 09:09:12	34999	[30000:34999]	analysis	2019-09-01 00:28:54	30000	True	None	0.1	0.460044	True	Nc
7	7	2020-04-30 11:46:53	39999	[35000:39999]	analysis	2019-12-31 10:07:15	35000	True	None	0.1	0.466746	True	Nc
8	8	2020-09-01 02:46:02	44999	[40000:44999]	analysis	2020-04-30 12:04:32	40000	True	None	0.1	0.466300	True	Nc
9	9	2021-01-01 04:29:32	49999	[45000:49999]	analysis	2020-09-01 02:46:13	45000	True	None	0.1	0.467798	True	Nc

```
In [10]: drift_fig = results.filter(column_names=results.continuous_column_names, methods=['jensen_shannon']).plot(kind='drift
         drift_fig.show()
```

Level 2 - Repeatable



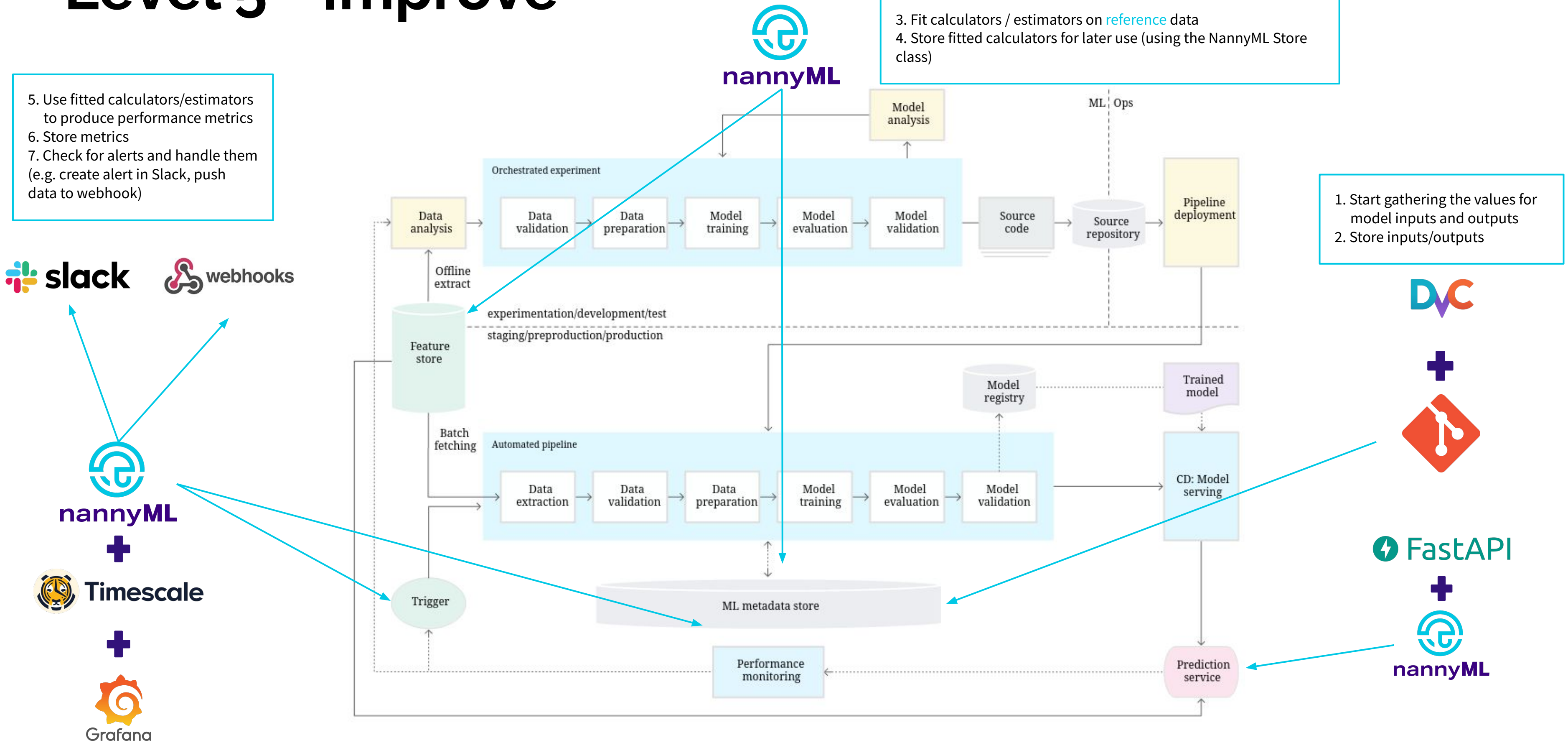
```python
import nannyml as nml

reference_df = pd.read_parquet('s3://my-data-bucket/reference.paruet')
analysis_df = pd.read_parquet('s3://my-data-bucket/2022/12/12/reference.paruet')

column_names = ['distance_from_office', 'salary_range', 'gas_price_per_litre',
'public_transportation_cost', 'wfh_prev_workday', 'workday', 'tenure', 'y_pred_proba', 'y_pred']
calc = nml.UnivariateDriftCalculator(
    column_names=column_names,
    timestamp_column_name='timestamp',
    continuous_methods=['kolmogorov_smirnov', 'jensen_shannon'],
    categorical_methods=['chi2', 'jensen_shannon'],
)

calc.fit(reference_df)
results = calc.calculate(analysis_df)

results.to_df(multilevel=False).to_csv('s3://my-data-bucket/results/univariate_drift.csv')

drift_fig = results.filter(column_names=results.continuous_column_names, methods=
['jensen_shannon']).plot(kind='drift')
drift_fig.write_image('_static/continuous_drift_js.svg')

drift_fig = results.filter(column_names=results.categorical_column_names, methods=
['chi2']).plot(kind='drift')
drift_fig.write_image('_static/categorical_drift_chi2.svg')

figure = results.filter(column_names=results.continuous_column_names, methods=
['jensen_shannon']).plot(kind='distribution')
figure.write_image('_static/continuous_distribution_js.svg')

figure = results.filter(column_names=results.categorical_column_names, methods=
['chi2']).plot(kind='distribution')
figure.write_image('_static/categoricals_distribution_chi2.svg')
}
```

Level 5 - Improve

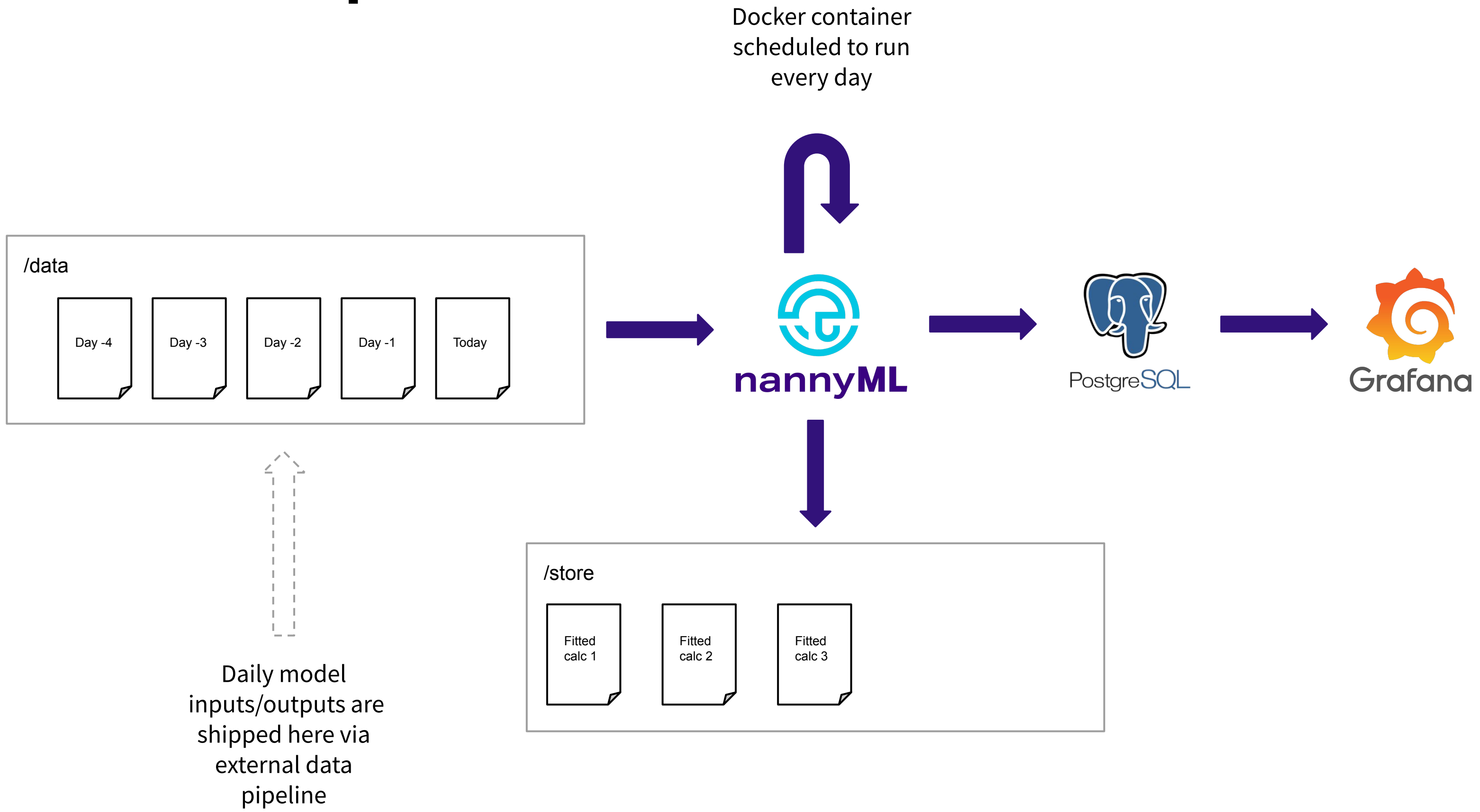
3. Fit calculators / estimators on reference data
4. Store fitted calculators for later use (using the NannyML Store class)

5. Use fitted calculators/estimators to produce performance metrics
6. Store metrics
7. Check for alerts and handle them (e.g. create alert in Slack, push data to webhook)

1. Start gathering the values for model inputs and outputs
2. Store inputs/outputs



Demo setup


Docker container scheduled to run every day
/data
Day -4
Day -3
Day -2
Day -1
Today
nannyML
PostgreSQL
Grafana
/store
Fitted calc 1
Fitted calc 2
Fitted calc 3
Daily model inputs/outputs are shipped here via external data pipeline

Some useful links

- https://nannyml.readthedocs.io/en/stable/quick.html#installing-nannyml
- https://nannyml.readthedocs.io/en/stable/tutorials/persisting_calculators.html
- https://nannyml.readthedocs.io/en/stable/cli/configuration_file.html#input-section
- https://nannyml.readthedocs.io/en/stable/cli/configuration_file.html#output-section
- https://nannyml.readthedocs.io/en/stable/cli/configuration_file.html#templating-paths

Conclusion

🥡

1. It's never too early to start monitoring.
2. Always collect your model inputs and outputs.
3. Use NannyML 🚀

Like what you see? Check us out on https://github.com/NannyML/nannyml

(leave a ⭐ or the kitty gets it)

Thank you!

We're looking for Design Partners

Free open source onboarding
In exchange for product feedback

Sign Up ⬇️⬇️

https://go.nannyml.com/design-partnership-form

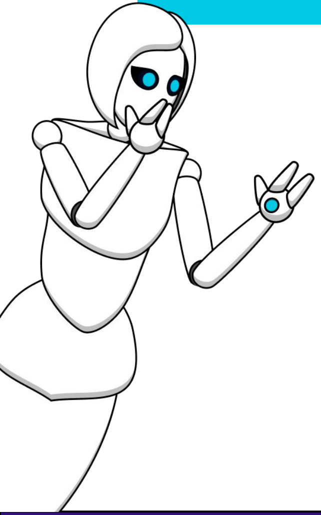
Let us know how you liked this webinar

1-minute feedback form 😃

https://go.nannyml.com/webinar-feedback-2401

nannyML

Join us again next Wednesday?



How to estimate the ML performance of deployed models?

Webinar with Wojtek Kuberski
Co-founder @NannyML

https://go.nannyml.com/webinar-01-feb

⭐GitHub/NannyML/nannyml NannyML