AI/ML Visualization

Shilpika
Argonne National Laboratory

Agenda

- Visualizing AI/ML
- AI explainability: SHAP, LIME, Captum (Pytorch), Explainable Boosting Machine (EBM), explAIner
- Quick plots and interactive dashboards : Plotly Dash, Voila jypyter, D3.js , Streamlit
- Training/experiment tracking: TensorBoard, Mlflow, W&B
- Embedding exploration: Facets (Google)

Why? Visualizing AI/ML

- Enhancing Model Interpretability
- Improving Debugging and Model Development
- Facilitating Decision-Making
- Identifying Patterns and Relationships
- Monitoring and Maintenance
- Reducing Cognitive Load

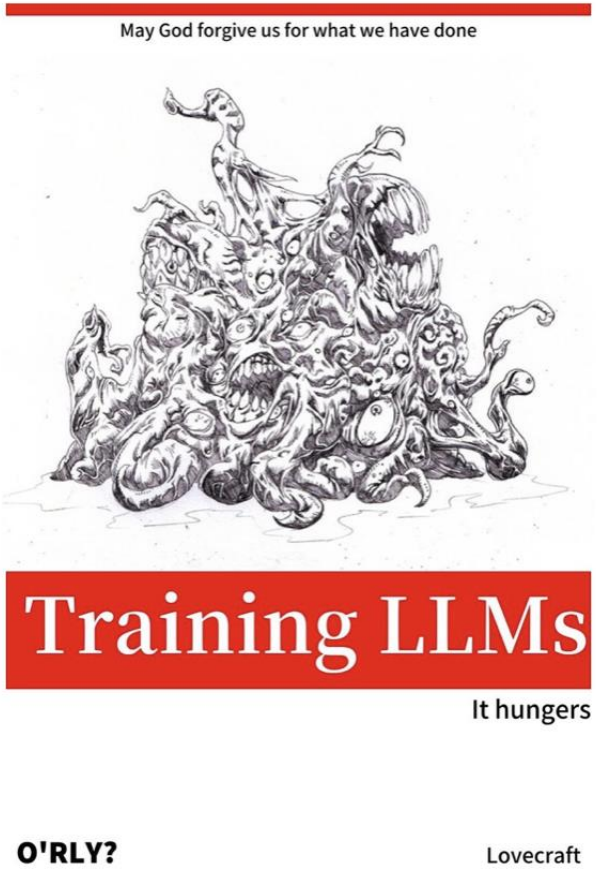


Image credit: https://raw.githubusercontent.com/saforem2/llm-lunch-talk/refs/heads/main/docs/assets/it_hungers.jpeg

Explainable AI

"Explainable AI (XAI)" founded by DARPA

Explainable AI (XAI) refers to the set of techniques, tools, and methods that make the **decision-making process of AI/ML models understandable to humans**

- **Transparency:** Shows the internal logic of the model or explains outputs in an interpretable way.
- **Interpretability:** Makes it easier for humans to understand the *meaning* of a model's predictions.
- **Justification:** Provides human-readable reasoning behind decisions.
- **Traceability:** Allows tracking the data and processing steps that led to a decision.

Local vs. global explanations

Model-agnostic vs. Model-specific

Explanation coverage = region in input space where the explanation applies

AI Explainability and Interpretability in ML Pipeline

"Explainable AI (XAI)" founded by DARPA

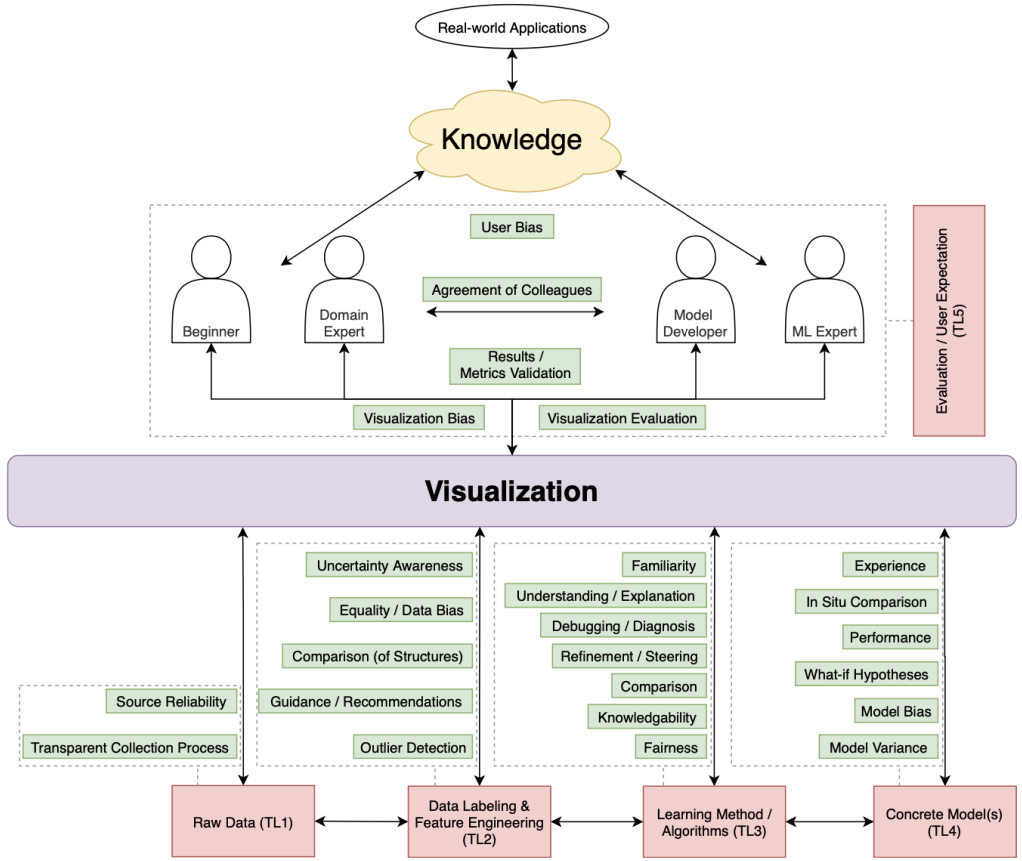


Figure 2: *A typical ML pipeline (depicted in red), assisted by visualization (in purple). Issues of trust permeate the complete shown pipeline, and we locate and categorize these issues in several* ***trust levels (TLs)****. The various categories proposed in this work are represented in green. The yellow "cloud" represents the knowledge created by the different target groups while they pursue their goals by using visualizations to explore the pipeline, the data and/or the ML models. Finally, at the very top, we encode the real-world applications with an ellipsoid.*

Topic 1
NNs hidden states and models' parameter spaces visualization (in time series applications)

keywords	papers
model view tree state hidden hidden state time parameter	[BAF*14, BAF*15, EASKC18, XCH*16, MCZ*17, MBD*11, MLMP18, Mad19, SBP19, SGPR18, TCE*19, TLKT09, ZTR16] 13 papers

Topic 2
network visualizations to investigate the behavior of projections and explore classes (for topic analysis)

keywords	papers
topic node network projection behavior class edge tree	[BWZ*20, CAA*19, CWS*17, Gle13, KKZE20, TLRB18, VKA*18, WSW*18, wv11] 9 papers

Topic 3
NNs models' hyper-parameters and reward visualization during training for image applications

keywords	papers
image model training input network hyper parameter reward	[AMJ18, SPBA19, SSS*18, WGZ*19, YCN*15] 5 papers

Topic 4
vector space, samples, and distributions visualization in projections and DR

keywords	papers
projection dimension view sample vector space point distribution	[ACD*15, AHH*14, CGZ*16, CD18b, EGG*12, GNRM08, GSS*20, IMI*10, JSR*19, KTC*19, LBT*18, LJLH19, SDMT16, SZL*18, WM18] 15 papers

Topic 5
models' predictions visualization in clustering (for text applications)

keywords	papers
model cluster clustering value view prediction document variable	[AYMW11, BDSF17, BPFG11, BvLBS, CD18a, CD19, CDS09, CWGW19, FMH16, FSJ13, GS14, HNH*12, HVP*19, JC17b, KEV*18, KKB19, KPB14, KPN16, LKC*12, LRL*18, MP13, MW10, PBK10, RG19, SBE*18, SKB*18, SMSL17, SNMM18, SRG*18, ZKM*19, ZWLC19, ZWRH14] 32 papers

Topic 6
ML models' explanations and visualization systems evaluation

keywords	papers
model learning system machine participant explanation instance machine learning	[ASW*19, BAL*15, BEF17, CEH*19, CHH*19, CS14, DCCE19, GHG*19, GSC16, GSK*20, HHC*19, HSD19, KPB16, JKM12, JSO19, JSS*18, JZF*09, KBW515, KDS*17, KFC16, KJR*18, KLTH10, KPB18, KSH18, MMD*19, MSM*17, MSW10, PSF17, RAL*17, SKKC19, SLT17, SSK10, SSSEA20, TKDB17, TMF*12, TSL*16, WMJ*19, WPB*20, ZYB*16] 39 papers

Topic 7
subspaces exploration and distances examination in clustering and DR

keywords	papers
cluster subspace dimension point view distance dimensional clustering	[AEM11, BAPB*16, BLBC12, CBB*19, CHAS18, CLKP10, CSG*18, FCS*20, FKM20, JHB*17, JJ09, KDFB16, LMZ*14, LWBP14, LWT*15, ML14, MvW11, PSPM12, RL14, RL15a, RL15b, RL16, TPRH11, VL09, WLN*17, WLS19, XYC*18, YZR*18, ZLH*16, dBD*12] 30 papers

Topic 8
models' predictions and design prototyping in ML algorithms

keywords	papers
model prediction instance view prototype algorithm learning design	[CPCS20, CSV*18, GDM*19, KAKC18, KCK*19, LGG*18, LLL*19, MQB19, MXC*20, MXLM20, MXQR19, SGB*19, SJS*17, SJS*18, SPG14, WGSY19, XXM*19, ZWM*19] 18 papers

Topic 9
individual points, projection space, and outliers' exploration

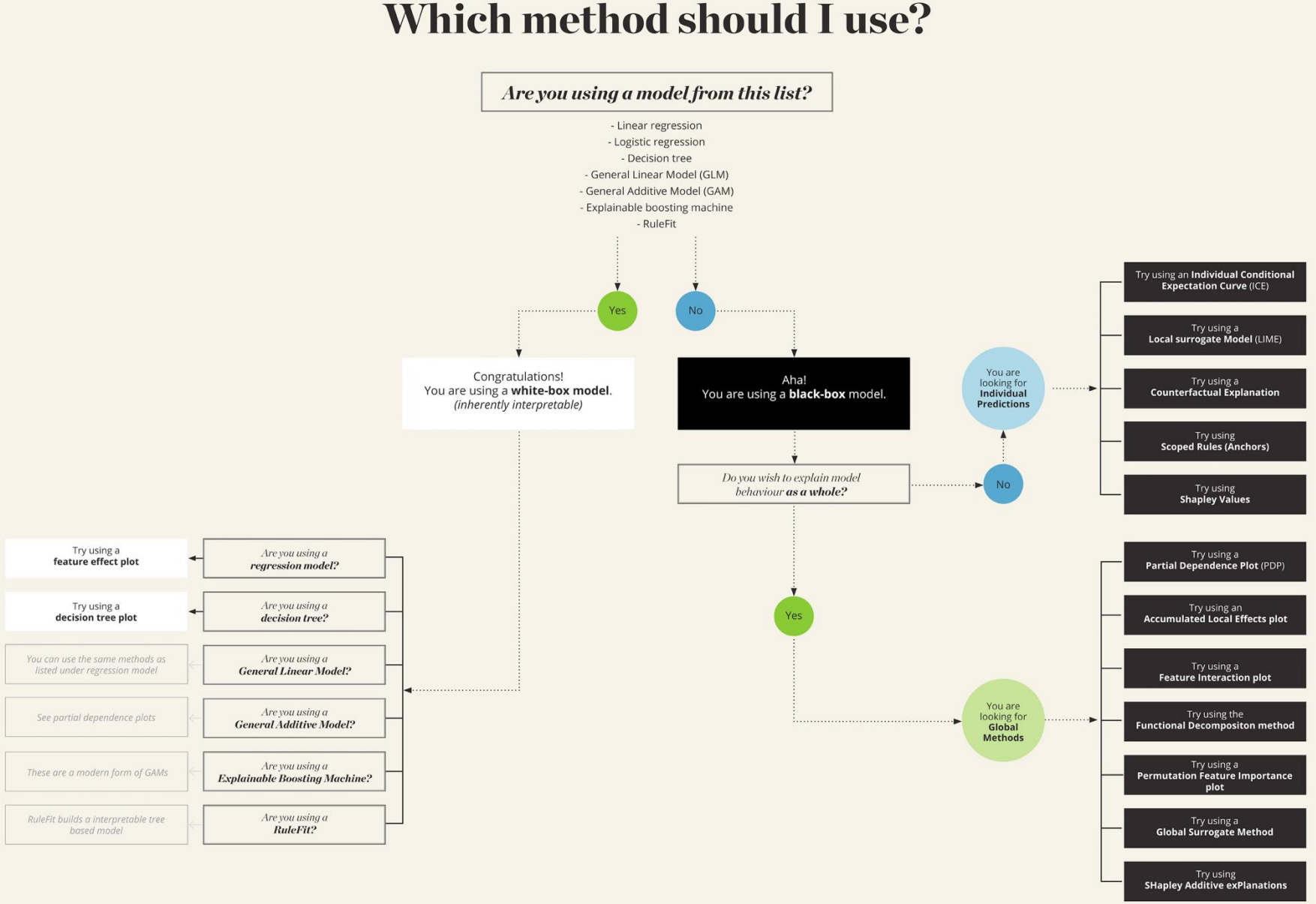
keywords	papers
point class projection space view instance value outlier	[AL20, Aup07, BHR*19, BZL*18, CMN*16, RSF*15, JPN15, KKW*17, KS12, LA11, MCMT14, PLHL19, RFT18, SNLH09, SRM*15, SvLB10, YMT17, ZCW*19, ZWC*18] 19 papers

Topic 10
NNs neurons' activations visualization during training for image applications

keywords	papers
layer network neuron model training activation class image	[AJY*18, CGR*17, CPM*18, HDK*19, HLW*20, HPRC20, LCJ*18, LLS*18, LSC*18, LSL*17, LXL*18, NHP*18, OSJ*18, PHY*18, PSMD14, RFFT17, SW17, WGYS18, ZF14, ZHP*17] 20 papers

Credit: The State of the Art in Enhancing Trust in Machine Learning Models with the Use of Visualizations. Computer Graphics Forum, 39: 713-756. https://doi.org/10.1111/cgf.14034

What Method Should I Use?



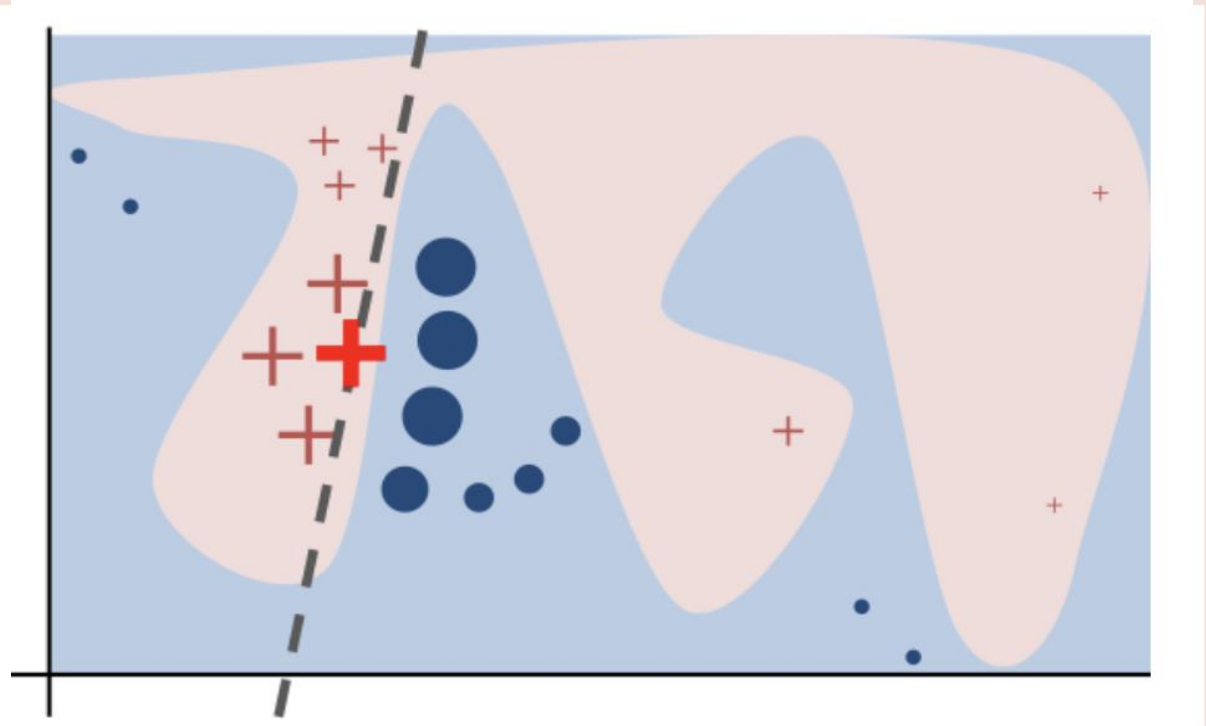
Source: https://medium.com/@okonstantinova/explainable-ai-xai-the-easy-guide-for-beginners-41e0753e03a7

LIME : Local Interpretable Model-Agnostic Explanations

LIME system generates a locally approximate explanatory model of an arbitrary learned model

- Mapping the complex model to a simpler one
- Model-agnostic
- The overall goal of LIME is to identify an interpretable model over the interpretable representation that is locally faithful to the classifier
- Produces an explanation for an individual prediction

"The black-box model's complex decision function, *f*, (unknown to LIME) is represented by the blue/pink background, which cannot be approximated well by a linear model. The bold red cross is the instance being explained. LIME samples instances, gets predictions using *f*, and weighs them by the proximity to the instance being explained (represented here by size). The dashed line is the learned explanation that is locally (but not globally) faithful."

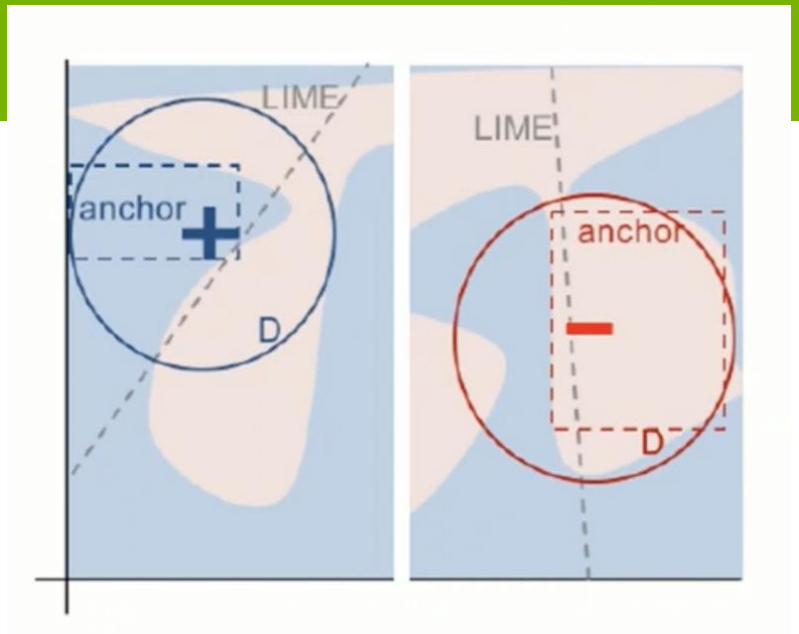


- M. Ribeiro, S. Singh, and C. Guestrin. "Why Should I Trust You?". In *Int. Conf. Knowl. Discov. Data Min.*, 2016.

ANCHORS: High Precision Model-Agnostic Explanations

Model-agnostic explanations based on if-then rules

- Local explanation
- An **anchor** explanation is a rule that sufficiently "anchors" the prediction locally – such that changes to the rest of the feature values of the instance do not matter. In other words, for instances on which the anchor holds, the prediction is (almost) always the same.



(a) Original image (b) Anchor for "beagle" (c) Images where Inception predicts $P(\text{beagle}) > 90\%$

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: high-precision model-agnostic explanations

Shapley Explanations

1. Based on cooperative game theory (Shapley values)
2. Treat model prediction as a cooperative game
3. Features collaborate to generate predictions
4. Shapley values distribute total prediction fairly among features

Advantages:

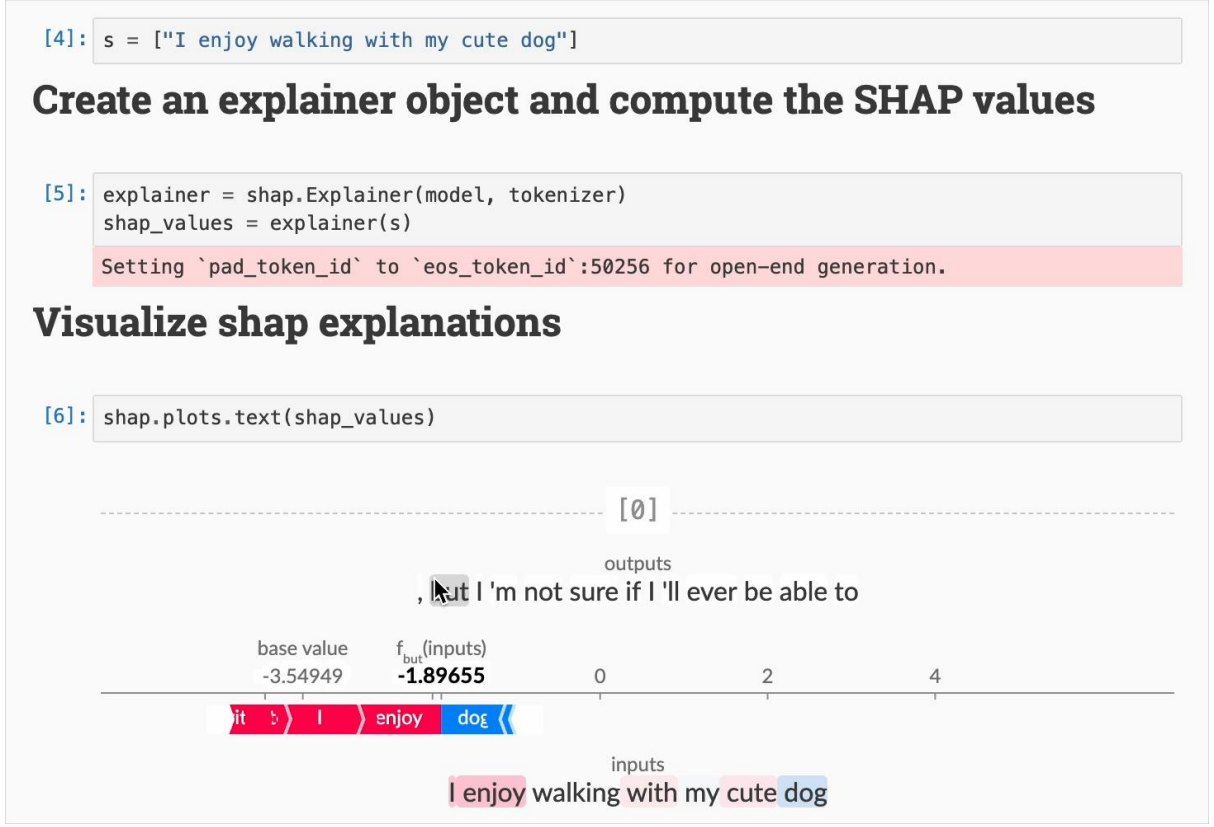
- Model-agnostic and theoretically grounded
- Consistent and locally accurate explanations
- Provides precise quantification of feature impact

Challenges:

- Computationally expensive for large feature sets
- Approximation methods (e.g., SHAP) often used

Applications:

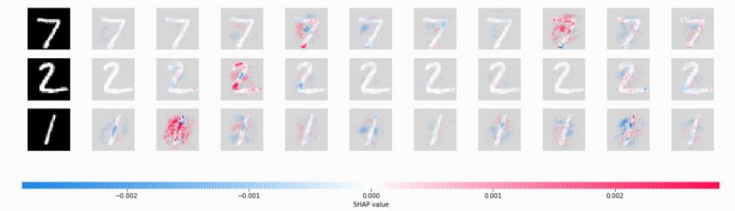
- Feature importance analysis
- Debugging and validating AI models
- Regulatory compliance and transparency

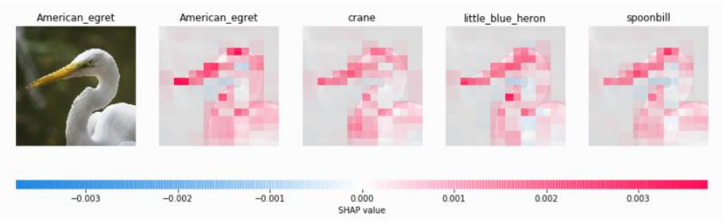


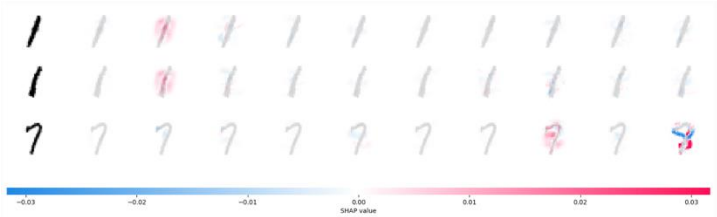
Source: https://shap.readthedocs.io/

Shapley Explanations

Explainer Type	Suitable For	Mechanism	Uses
GradientExplainer	Deep learning models (TF, Keras)	Uses model gradients and background dataset	Fast for differentiable models
PartitionExplainer	Tree-based models or structured data	Hierarchical clustering of features	Efficient with structured dependencies
DeepExplainer	Deep neural networks	Approximates Shapley using DeepLIFT-style gradients	Fast, tailored for DNNs







CAPTUM

- **Captum** (Latin for "comprehension") is a model-agnostic interpretability library for PyTorch
- Developed by Facebook AI and integrated with the PyTorch ecosystem

- **Gradient-based methods:** Integrated Gradients, Saliency Maps, Grad-CAM.
- **Perturbation-based methods:** Feature Ablation, Occlusion.
- **Layer-wise attribution:** Layer Conductance, Neuron Conductance.
- **Model-agnostic API:** Supports any PyTorch nn.Module.

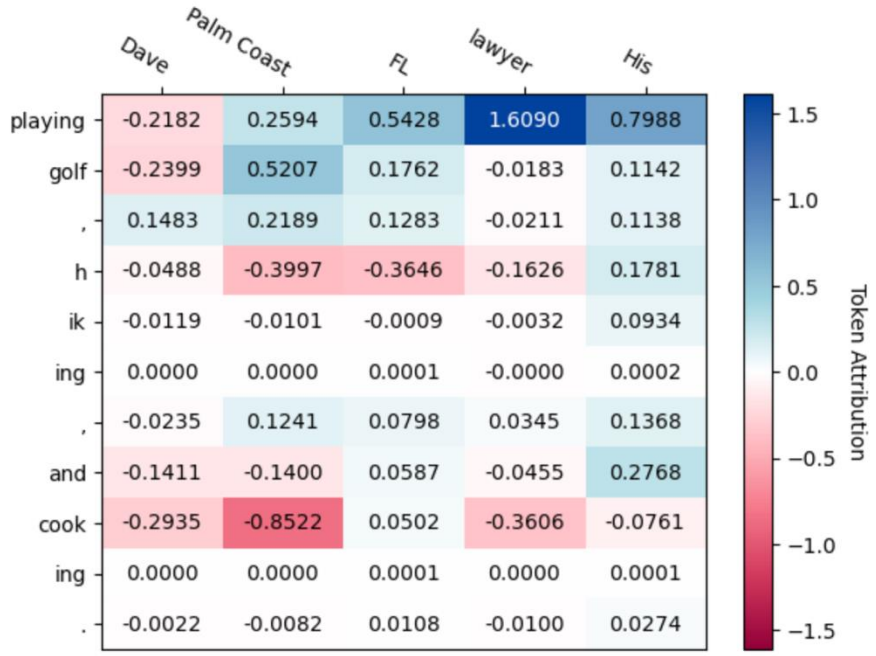
Task: Classification into ImageNet-1k categories.
Model: A ResNet18 trained on ImageNet-1k.
Data to inspect: Samples from PASCAL VOC (Visual Object Classes) 2012.
Ablation based on: Segmentation masks.

Source: http://captum.ai/

CAPTUM

Understanding Llama2 with Captum LLM Attribution

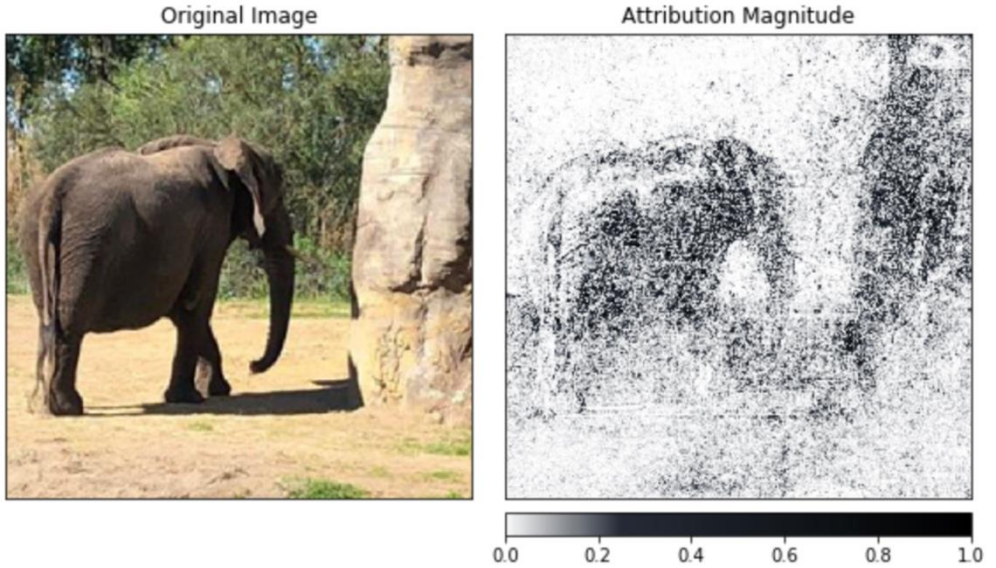
Prompt = "Dave lives in Palm Coast, FL and is a lawyer. His personal interests include"



Target= playing guitar, hiking, and spending time with his family.

Model interpretation for Visual Question Answering (VQA)

Target Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
elephant	elephant (0.55)	elephant	8.21	what is on the picture



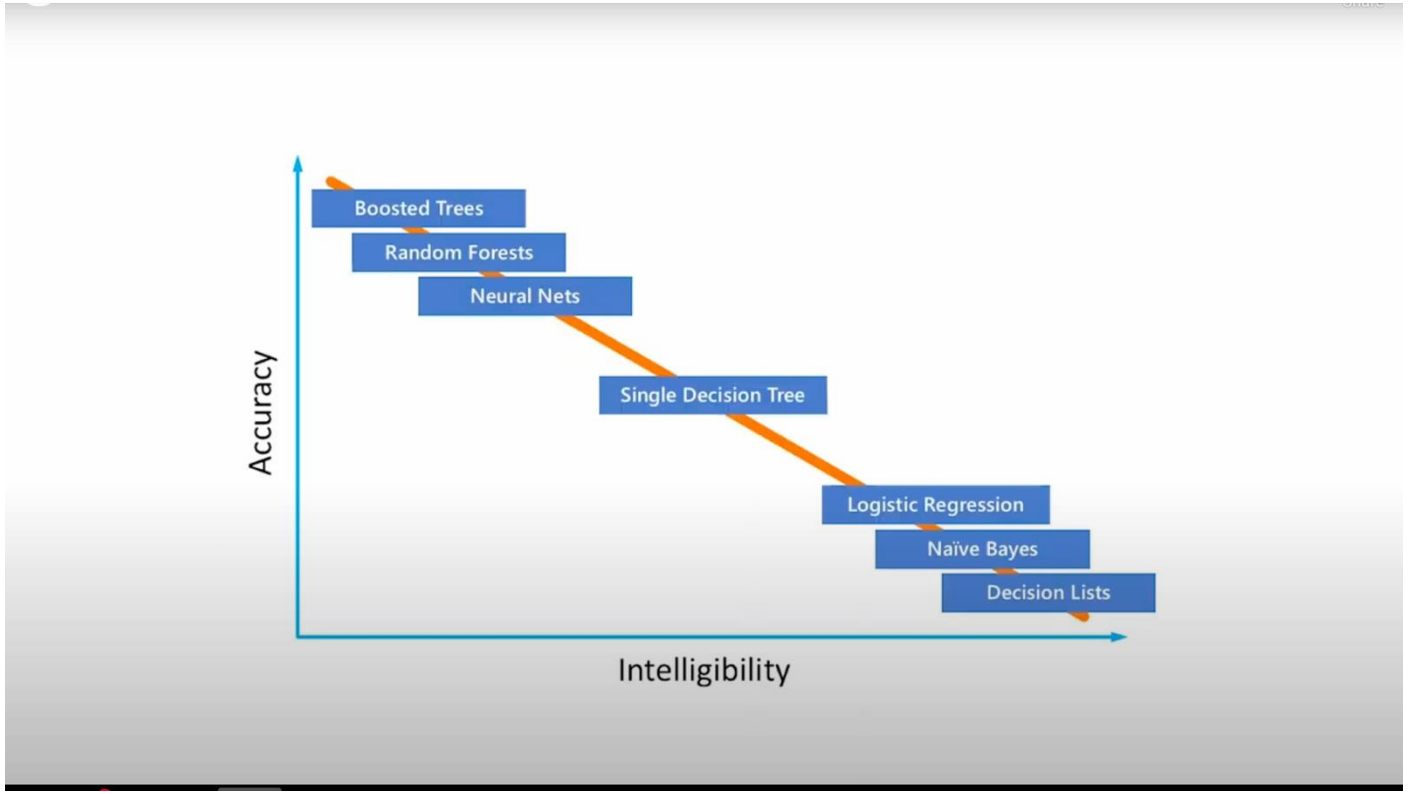
Source: http://captum.ai/

Comparison between Tools/Libraries

	Captum	LIME	SHAP
Integration	Native PyTorch support	Framework-agnostic (black-box)	Framework-agnostic (black-box)
Methodology	Gradient- & layer-based, plus perturbation	Local surrogate models (regression)	Shapley value estimation
Interaction effects	Captures via IntegratedGradients and interaction modules	Limited (requires feature grouping)	Captures interactions but computationally heavy
Efficiency	Fast for gradient-friendly models	Slower (requires many perturbations)	Slow for high-dimensional data
Extensibility	Add custom explainers easily	Extendable via custom surrogates	Extendable via custom kernel SHAP

Explainable Boosting Machine (EBM)

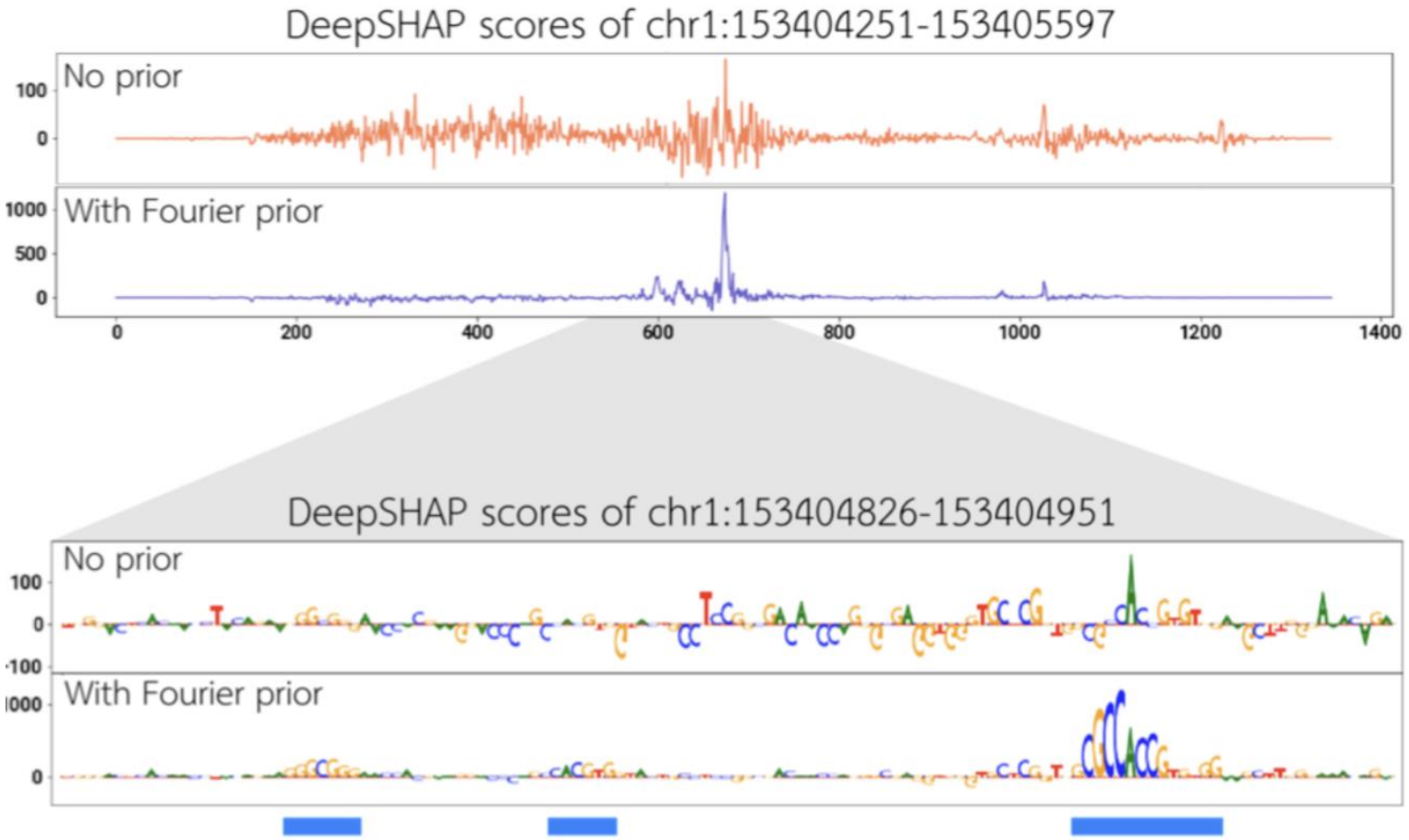
- Model-specific
- Interpretability part of training phase
- Source reliability



"InterpretML: A Unified Framework for Machine Learning Interpretability" (H. Nori, S. Jenkins, P. Koch, and R. Caruana 2019)

Fourier-transform-based attribution priors improve the interpretability and stability of deep learning models for genomics

- Deep learning model to map genomic DNA sequences to protein-DNA binding data
- Model-agnostic (without deeper knowledge of the network properties)
- Local explanation method
- Explanations correspond to pieces of DNA sequences,'motifs', which are responsible for the DNA discoverable properties



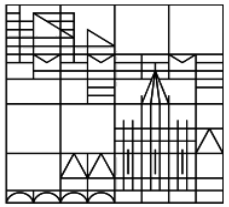
Alex M. Tseng, Avanti Shrikumar, and Anshul Kundaje. 2020. Fourier-transform-based attribution priors improve the interpretability and stability of deep learning models for genomics

explAIner: A Visual Analytics Framework for Interactive and Explainable Machine Learning

Thilo Spinner, Udo Schlegel, Hanna Schäfer, and Mennatallah El-Assady

University Konstanz

Universität Konstanz



Explainability Toolboxes

Toolbox	Publication	Code repository
InterpretML	Nori et al. (2019)	https://github.com/interpretml/interpret
iNNvestigate	Alber et al. (2019)	https://github.com/albermax/innvestigate
AI Fairness 360	Arya et al. (2019)	https://github.com/Trusted-AI/AIF360
explAIner	Spinner et al. (2020)	https://github.com/dbvis-ukon/explainer
FAT Forensics	Sokol et al. (2020)	https://github.com/fat-forensics/fat-forensics
Alibi	Klaise et al. (2021)	https://github.com/SeldonIO/alibi

List of Explainability Techniques

Level of explainability: Global and local.
Global: they explain the model in general, noting its generic operating rules.
Local: They explain for every single data, how the model reasoned and the rules that led to a certain output.

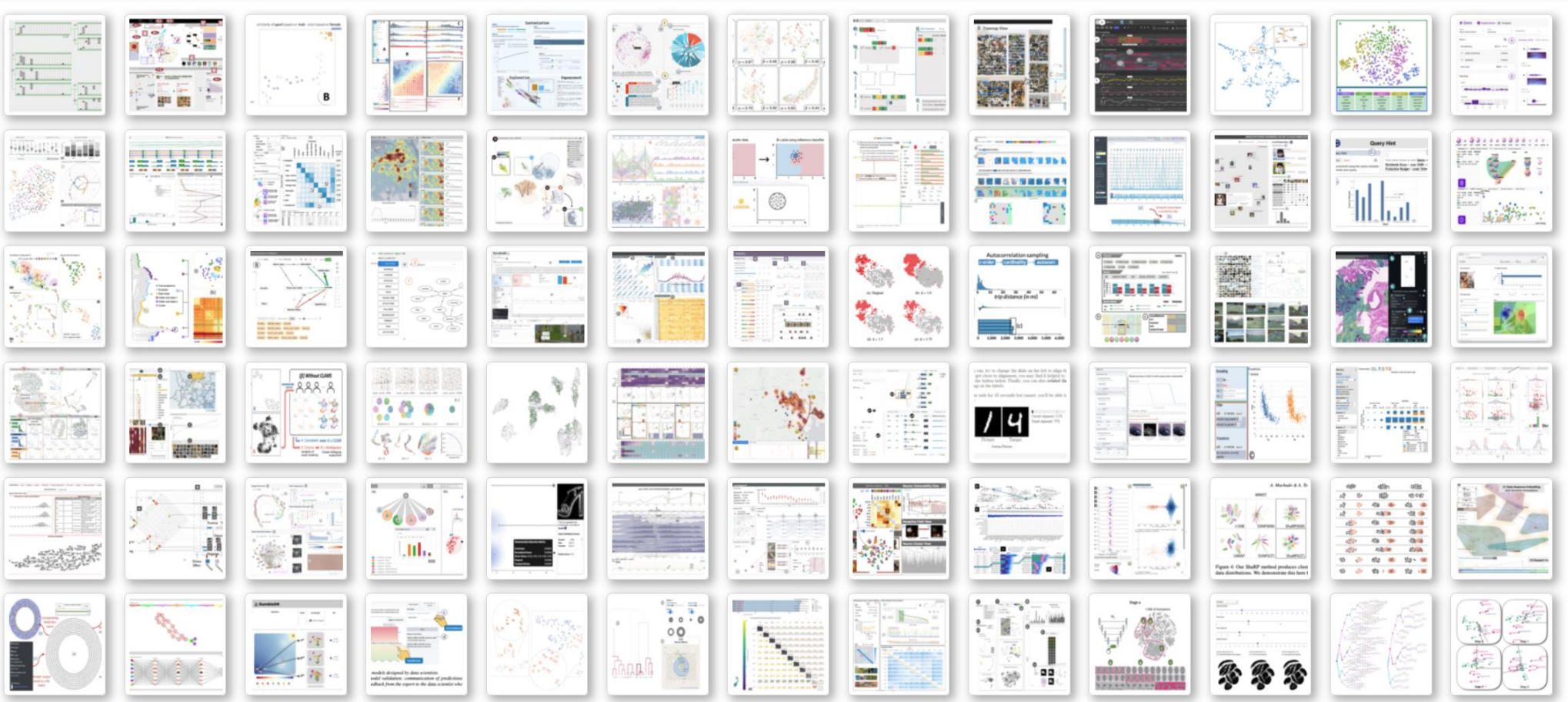
- SHAP - **(SHapley Additive exPlanations)** is a framework that explains the output of any model using Shapley values, a game-theoretic approach often used for optimal credit allocation
- LIME - Local interpretable model-agnostic explanations (LIME) is a method that fits a surrogate glass-box model around the decision space of any black-box model's prediction.
- Permutation Importance - **feature importance** can be measured by looking at **how much the score** (accuracy, F1, R^2, etc. — any score we're interested in) **decreases when a feature is not available**.
- Partial Dependence Plot - **PDP or PD plot** shows the marginal effect one or two features have on the predicted outcome of a machine learning model.
- Morris Sensitivity Analysis - **One-step-at-a-time (OAT) global sensitivity analysis** where only one input has its level (discretized value) adjusted per run.
- Accumulated Local Effects (ALE) - **Accumulated Local Effects (ALE)** is a method for computing feature effects. The algorithm provides model-agnostic (black box) global explanations for classification and regression models on tabular data.
- Anchors - explain the behaviour of complex models with **high-precision rules called anchors**. These anchors are locally sufficient conditions to ensure a certain prediction with a high degree of confidence.
- Contrastive Explanation Method (CEM) - **CEM** generates instance-based local black box explanations for classification models in terms of Pertinent Positives (PP) and Pertinent Negatives (PN).

List of Explainability Techniques

Level of explainability: Global and local.
Global: they explain the model in general, noting its generic operating rules.
Local: They explain for every single data, how the model reasoned and the rules that led to a certain output.

- Counterfactual Instances - Counterfactual explanations 'interrogate' a model to show how much individual feature values would have to change in order to flip the overall prediction.
- Integrated Gradients - Integrated Gradients aims to attribute an importance value to each input feature of a machine learning mo del based on the gradients of the model output with respect to the input.
- Global Interpretation via Recursive Partitioning (GIRP) - A compact binary tree that interprets ML models globally by representing the most important decision rules implicitly contained in the model using a contribution matrix of input variables.
- Protodash - A new approach for finding "prototypes" in an existing machine learning program. A prototype can be thought of as a subset of the data that have a greater influence on the predictive power of the model.
- Scalable Bayesian Rule Lists - Learn from the data and create a decision rule list. They have a logical structure that is a sequence of IF-THEN rules, identical to a decision list or one-sided decision tree.
- Tree Surrogates - Tree Surrogates are an interpretable model that is trained to approximate the predictions of a black-box model. We can draw conclusions about the black-box model by interpreting the surrogate model. The policy trees are easily human interpretable and provide quantitative predictions of future behaviour.
- Explainable Boosting Machine (EBM) - EBM is an interpretable model developed at Microsoft Research. It uses modern machine learning techniques like bagging, gradient boosting, and automatic interaction detection to breathe new life into traditional GAMs (Generalized Additive Models)

A Visual Survey in Enhancing Trust in Machine Learning (ML) Models with Visualization (CGF survey article (2020) and IEEE CG&A survey article (2024))

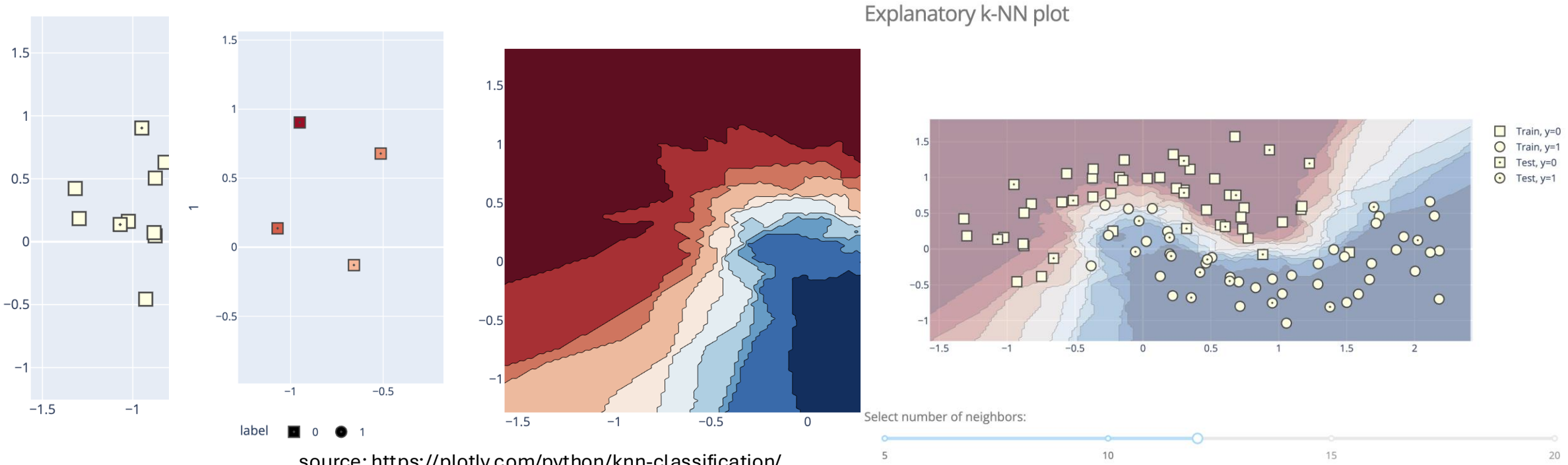


https://trustmlvis.lnu.se/

Quick plots and Interactive Dashboards

Plotly Dash

- ***Python framework for building ML & data science web apps***
- With Dash Open Source, Dash apps run on your local laptop or workstation, but cannot be easily accessed by others in your team.

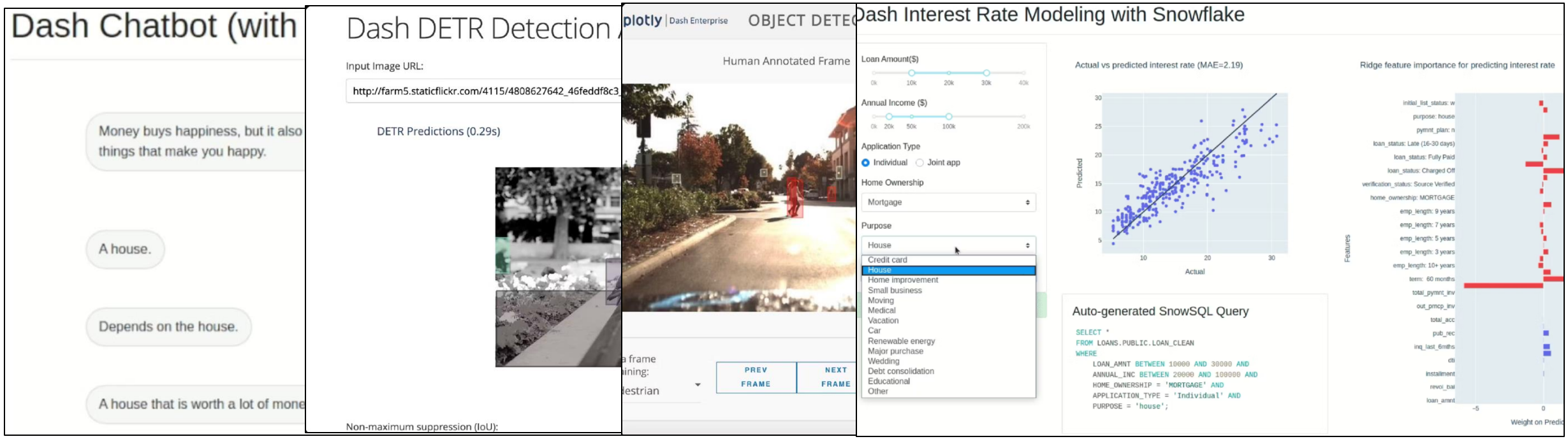


source: https://plotly.com/python/knn-classification/

Plots and interactive dashboards

Plotly Dash

- ***Python framework for building ML & data science web apps***
- With Dash Open Source, Dash apps run on your local laptop or workstation, but cannot be easily accessed by others in your organization.

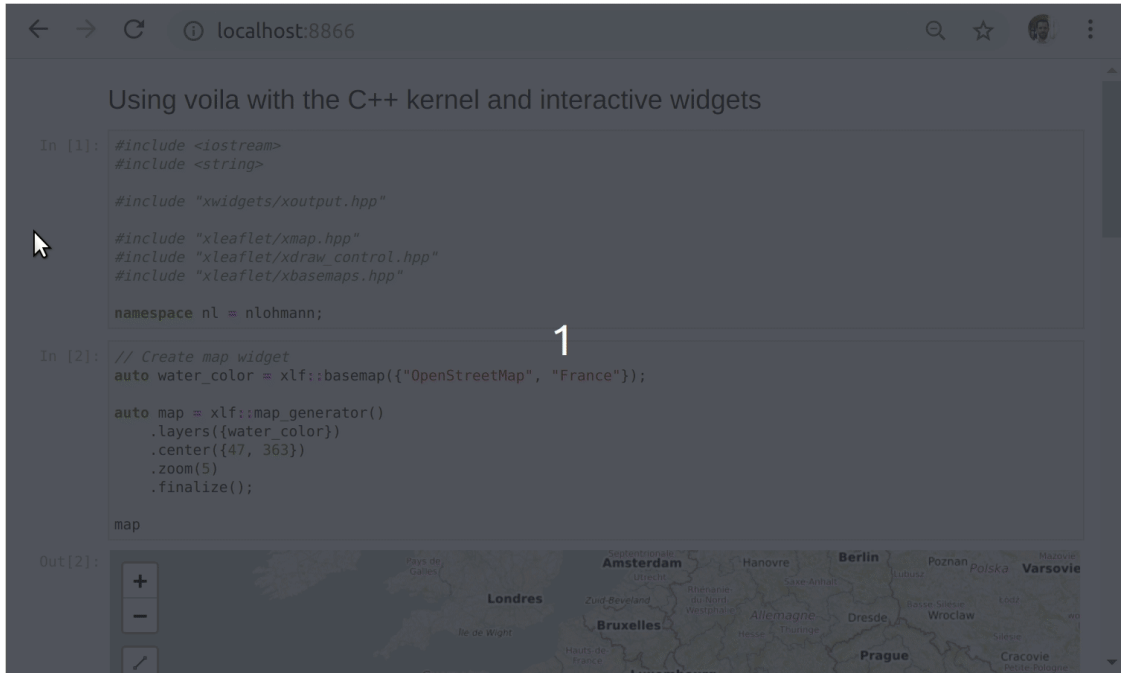


source: https://plotly.com/building-machine-learning-web-apps-in-python/

Plots and interactive dashboards

Voila jypyter, Custom D3.js

- Voilà : Jupyter notebooks into standalone web applications and allows you to share your work with others.
- Voilà can be used as a standalone application, or as a Jupyter server extension



Source: https://github.com/voila-dashboards/

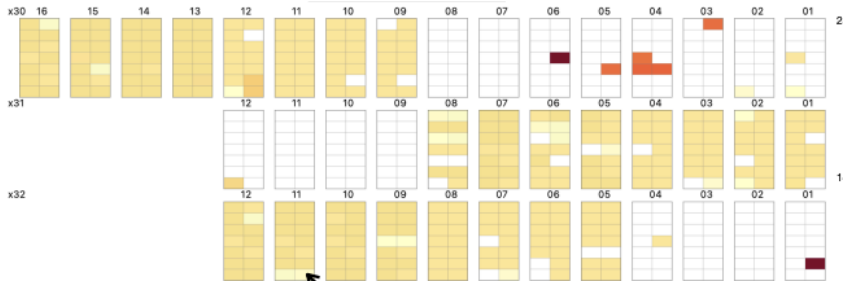
Plots and interactive dashboards

D3.js in jupyter notebook

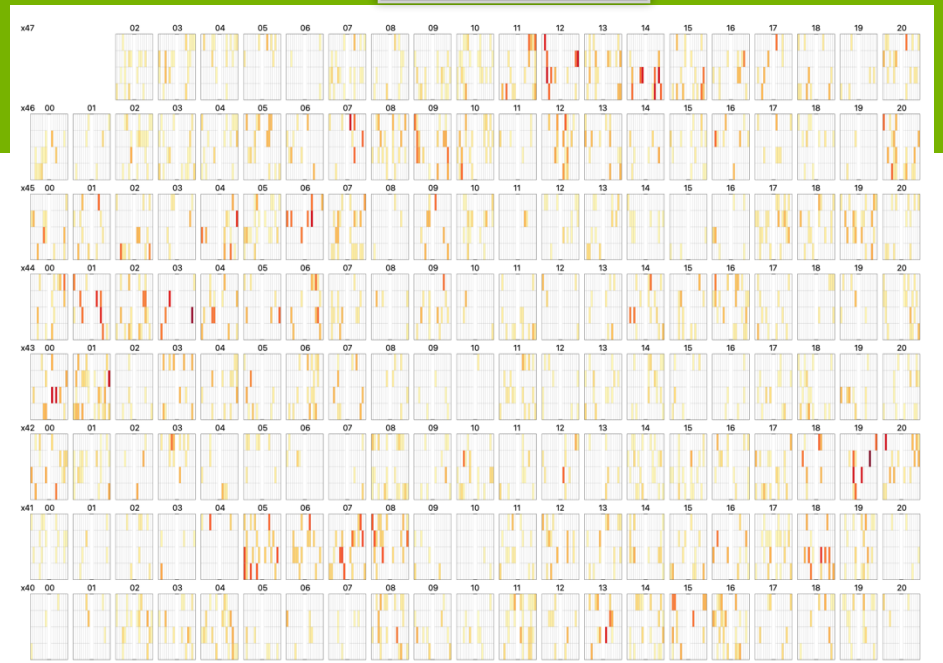
```
[5]: sysl = HPCLayout(dummy_data, path="../")
     template1,template2 = sysl.get_javascript_templates()

     display(template1)
     display(template2)
```

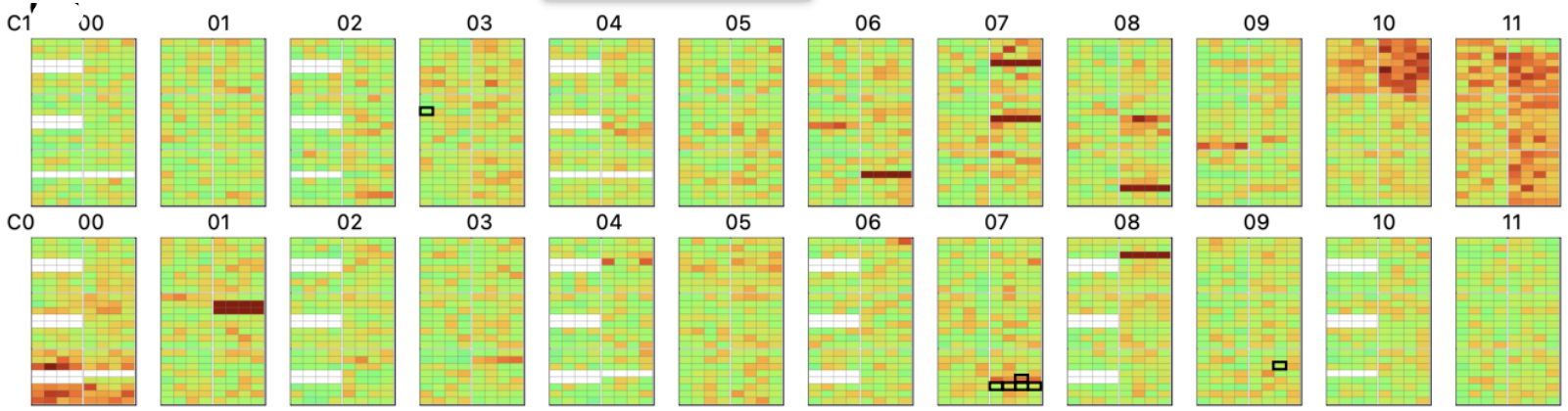
Polaris

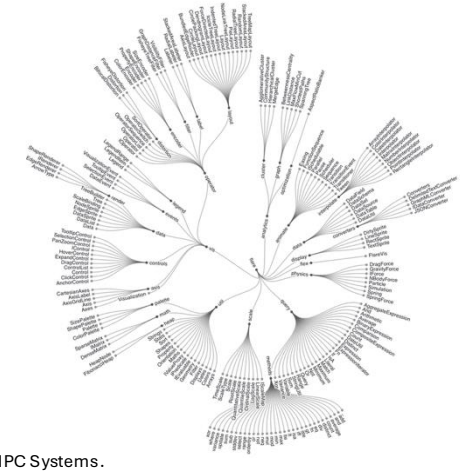


Aurora



Theta





https://medium.com/@stallonejacob/d3-in-juypter-notebook-685d6dca75c8

An Incremental Multi-Level, Multi-Scale Approach to Assessment of Multifidelity HPC Systems.

Plots and interactive dashboards

Streamlit

Streamlit is an open-source Python framework for data scientists and AI/ML engineers to deliver interactive data apps

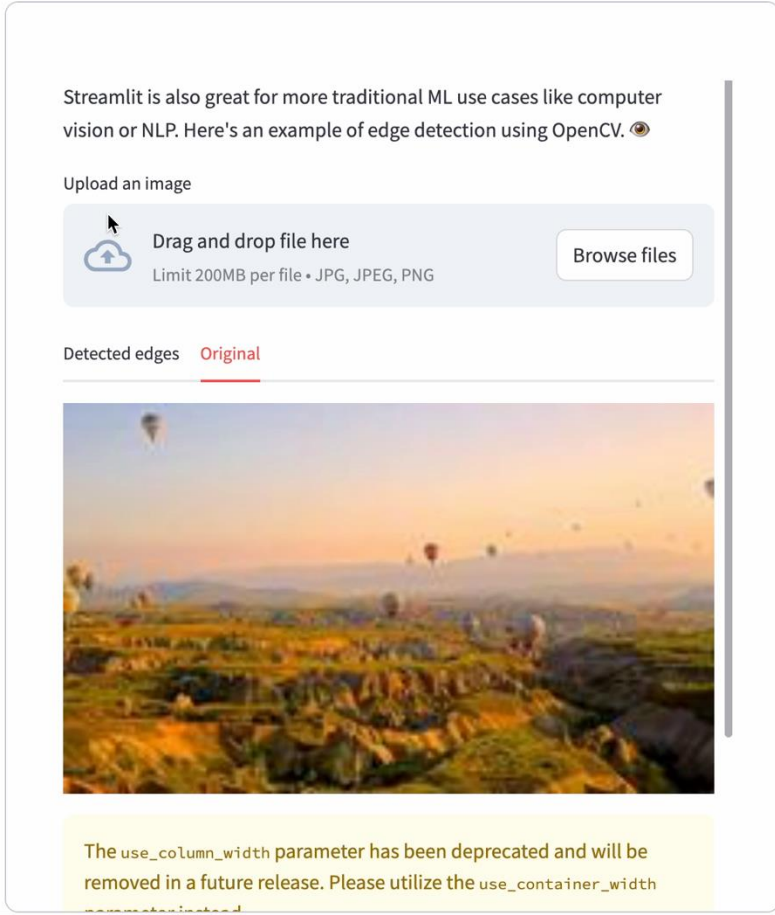
- Quickly prototyping data and ML apps
- Interactive model demonstrations
- Simple deployment of ML models for stakeholder presentations
- Intuitive and accessible UI for non-developers

```python
import streamlit as st
import cv2
import numpy as np
from PIL import Image
import requests

st.write("Streamlit is also great for more traditional ML use cases

uploaded_file = st.file_uploader("Upload an image", type=["jpg", "j
if uploaded_file:
    image = Image.open(uploaded_file)
else:
    image = Image.open(requests.get("https://picsum.photos/200/120"

edges = cv2.Canny(np.array(image), 100, 200)
tab1, tab2 = st.tabs(["Detected edges", "Original"])
tab1.image(edges, use_column_width=True)
tab2.image(image, use_column_width=True)
```



Source: https://streamlit.io/

Monitoring AI/ML Training

- TensorBoard
- Mlflow
- Weights & Biases

Monitoring AI/ML Training

TensorBoard

TensorBoard provides the visualization and tooling needed for machine learning experimentation:

- Tracking and visualizing metrics such as loss and accuracy
- Visualizing the model graph (ops and layers)
- Viewing histograms of weights, biases, or other tensors as they change over time
- Projecting embeddings to a lower dimensional space
- Displaying images, text, and audio data
- Profiling TensorFlow programs

Source: https://www.tensorflow.org/tensorboard

Monitoring AI/ML Training

TensorBoard

TensorBoard's **Graphs dashboard**

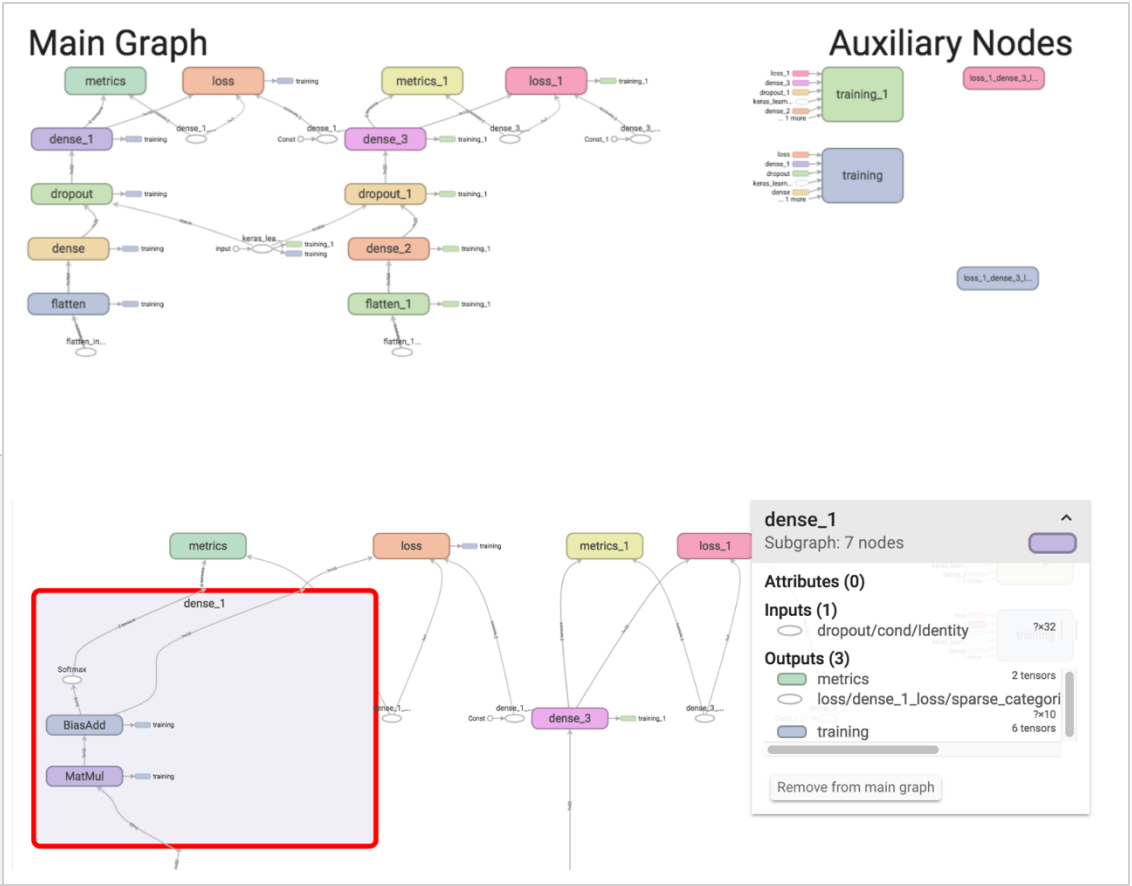
- View a conceptual graph of your model's structure and ensure it matches your intended design.
- View op-level graph to understand how TensorFlow understands your program.
- Examining the op-level graph can give you insight as to how to change your model.

```
# Define the model.
model = keras.models.Sequ
    keras.layers.Flatten(
    keras.layers.Dense(32
    keras.layers.Dropout(
    keras.layers.Dense(10
])

model.compile(
    optimizer='adam',
    loss='sparse_categori
    metrics=['accuracy'])
```

```
# Define the Keras TensorBoard callback.
logdir="logs/fit/" + datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = keras.callbacks.TensorBoard(log_dir=logdir)

# Train the model.
model.fit(
    train_images,
    train_labels,
    batch_size=64,
    epochs=5,
    callbacks=[tensorboard_callback])
```



Source: https://www.tensorflow.org/tensorboard/graphs

Monitoring AI/ML Training

TensorBoard

TensorFlow Profiler: Profile model performance

View the performance profiles by navigating to the **Profile** tab.

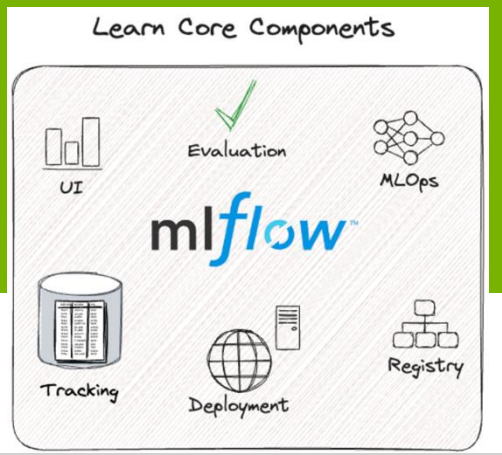
The **Profile** tab opens the Overview page which shows you a high-level summary of your model performance.

The Trace Viewer shows you a timeline of the different events that occurred on the CPU and the GPU during the profiling period.

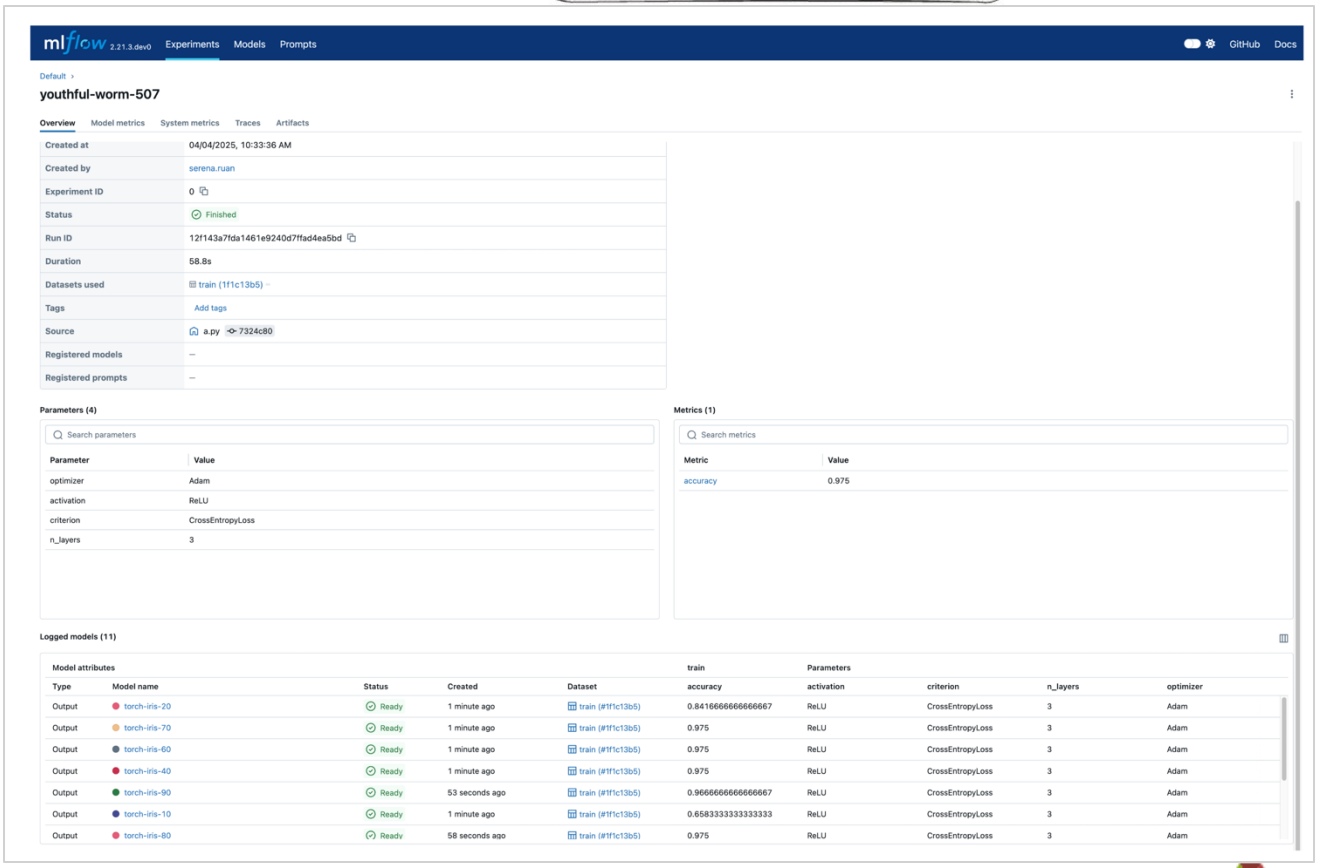
Source: https://www.tensorflow.org/tensorboard/tensorboard_profiling_keras

Monitoring AI/ML Training

Mlflow



- MLflow is an open-source platform to assist machine learning practitioners and teams in handling the complexities of the machine learning process.
- MLflow **focuses on the full lifecycle** for machine learning projects
  - manageable, traceable, and reproducible.



MLflow Tracking
- experiment logging
- parameter tracking
- metrics visualization
- artifact management.

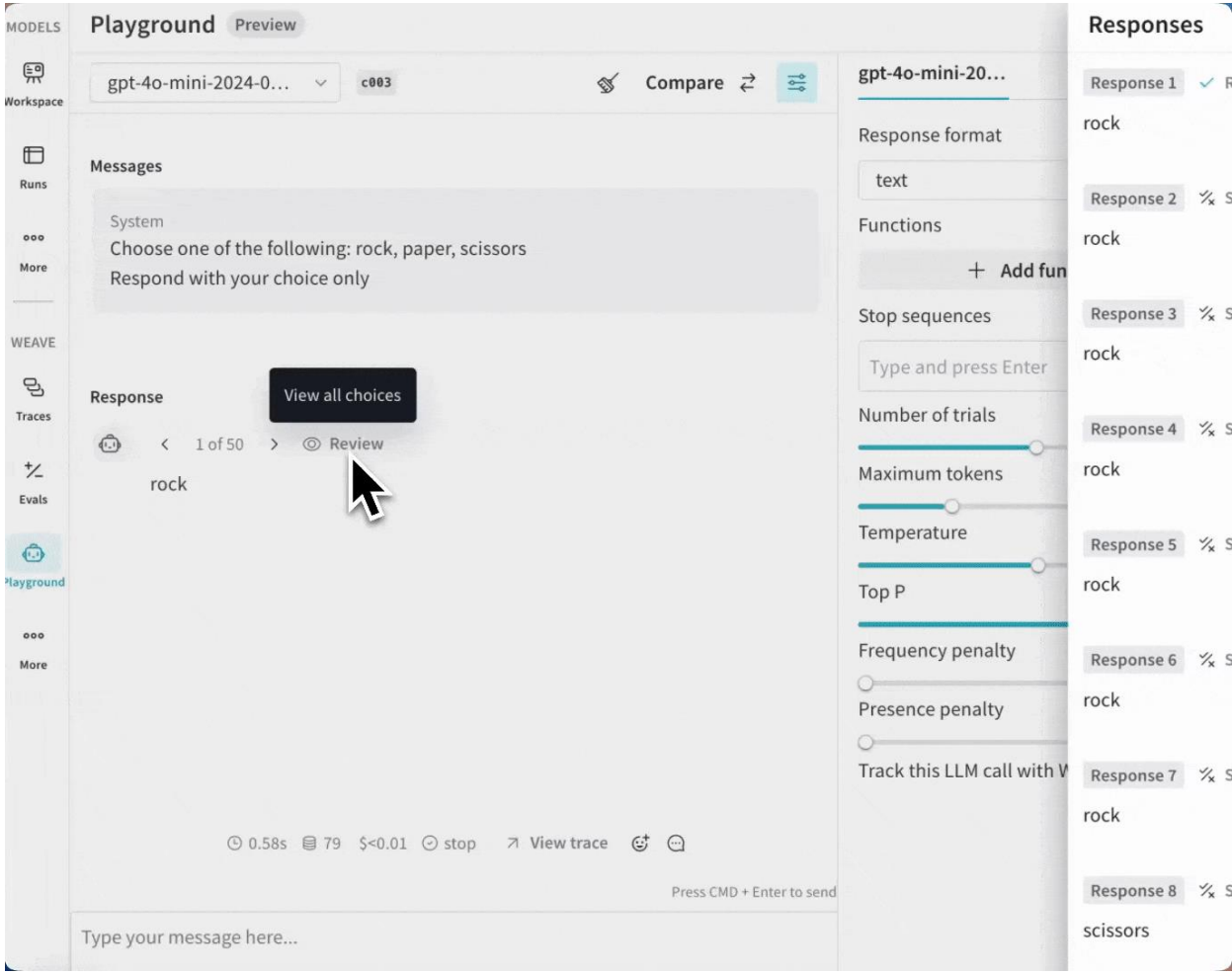
Support DL Frameworks
- Keras
- PyTorch
- TensorFlow
- Transformers
- SpaCy

Source: https://mlflow.org/docs/latest/ml/

Monitoring AI/ML Training

Weights & Biases

- Experiment with prompts and models
- Compare different models to find the best fit. Easily test the same prompt across multiple models to determine which performs best for your use case.
- Get statistical results
  - Generate multiple outputs for the same prompt to assess response consistency. Review individual trials to identify inconsistencies or outliers, then use these insights to fine-tune your LLM settings and implement effective guardrails.
- Refine prompting techniques
  - Load production traces into the playground to troubleshoot prompt issues. Iterate on system prompts until the models return optimal responses. Adjust model settings such as temperature and maximum tokens to further optimize prompts specifically for your application.



Source: https://wandb.ai/site

Monitoring AI/ML Training

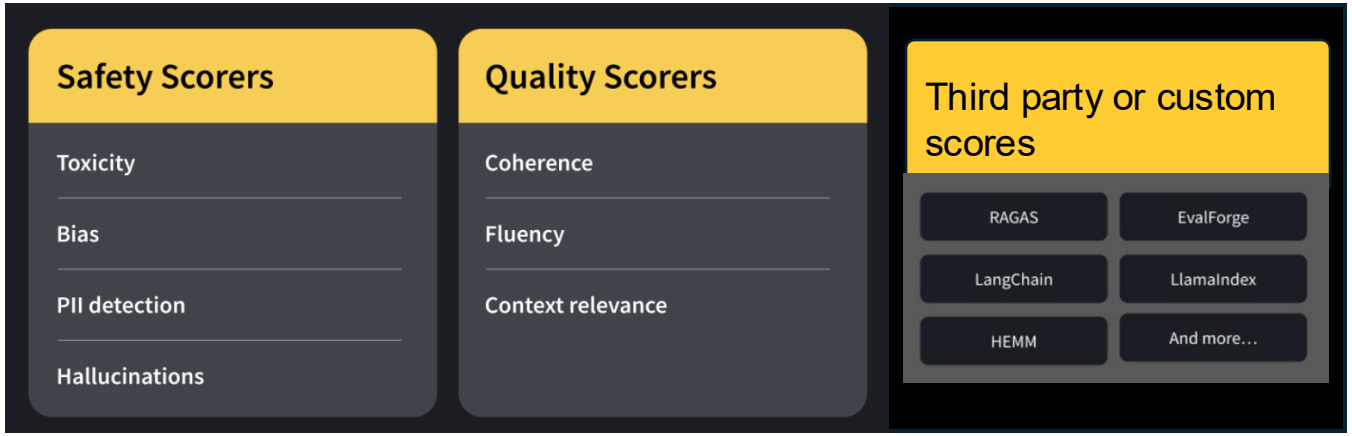
Weights & Biases

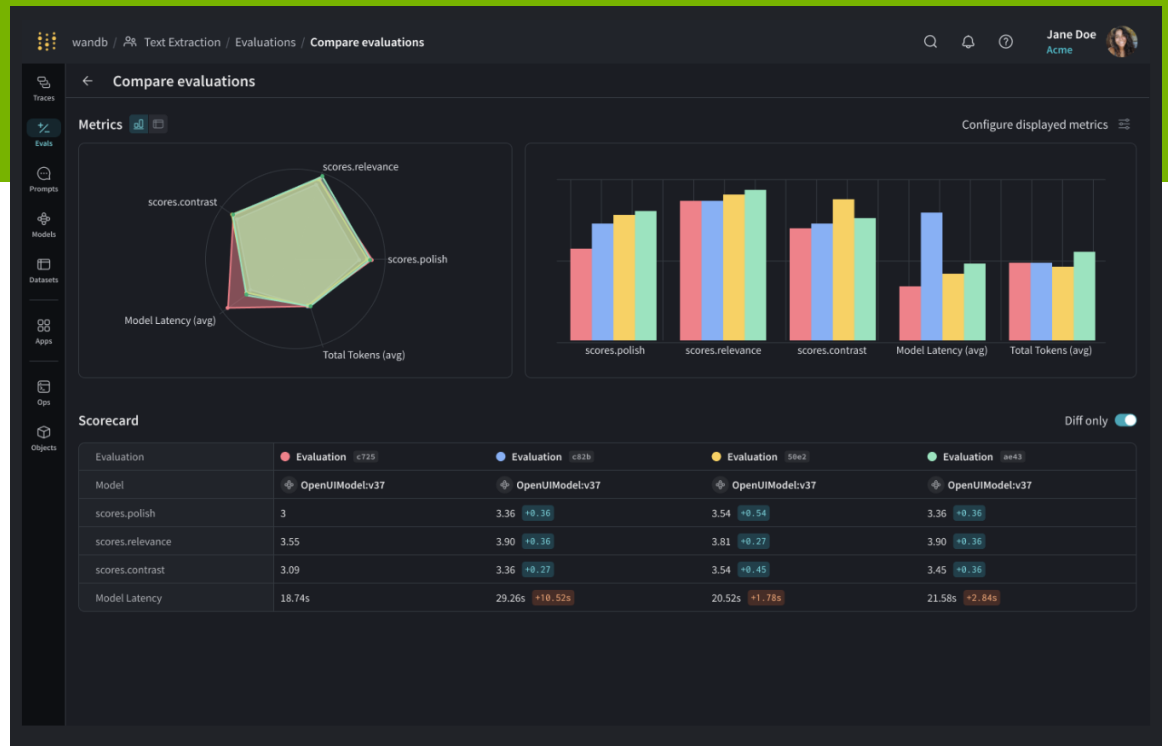
Guardrails:

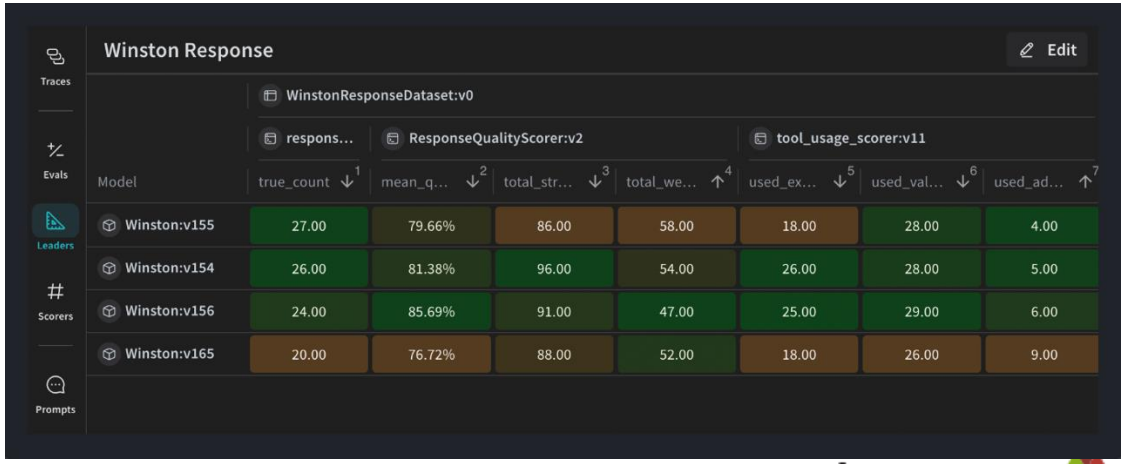
Guardrails offers pre-built scorers for safety and quality to support responsible AI.

Scorers:

Run scorers on any LLM call—on user inputs they detect and mitigate malicious activities such as prompt injection, and on AI outputs they identify and prevent hallucinations or inappropriate content.







Source: https://wandb.ai/site

Embedding Exploration

Facets (Google)

Facets is a visualization tool developed by Google Research, specifically created for exploring large datasets, particularly in machine learning and data analysis contexts.
Helps identify issues or interesting patterns within data, like missing values, skewed distributions, or anomalies.

"Takes input feature data from any number of datasets, analyzes them feature by feature and visualizes the analysis"

- Robust handling of large-scale, structured data.
- Quick identification of data quality issues and distribution anomalies.
- Interactive filtering and exploration, useful in preprocessing and exploratory data analysis (EDA).

- Quickly assessing training dataset characteristics, ensuring data quality before model training, and debugging model errors by inspecting problematic instances.

Source:https://pair-code.github.io/facets/

References & Reading

1. Chatzimparmpas, A., Martins, R.M., Jusufi, I., Kucher, K., Rossi, F. and Kerren, A. (2020), The State of the Art in Enhancing Trust in Machine Learning Models with the Use of Visualizations. Computer Graphics Forum, 39: 713-756. https://doi.org/10.1111/cgf.14034
2. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). Association for Computing Machinery, New York, NY, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778
3. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: high-precision model-agnostic explanations. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'18/IAAI'18/EAAI'18). AAAI Press, Article 187, 1527–1535.
4. "InterpretML: A Unified Framework for Machine Learning Interpretability" (H. Nori, S. Jenkins, P. Koch, and R. Caruana 2019)
5. Scott M. Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 4768–4777.
6. Alex M. Tseng, Avanti Shrikumar, and Anshul Kundaje. 2020. Fourier-transform-based attribution priors improve the interpretability and stability of deep learning models for genomics. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20). Curran Associates Inc., Red Hook, NY, USA, Article 161, 1913–1923.
7. T. Spinner, U. Schlegel, H. Schäfer and M. El-Assady, "explAIner: A Visual Analytics Framework for Interactive and Explainable Machine Learning," in IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 1, pp. 1064-1074, Jan. 2020, doi: 10.1109/TVCG.2019.2934629. keywords: {Data models;Analytical models;Computational modeling;Pipelines;Machine learning;Monitoring;Explainable AI;Interactive Machine Learning;Deep Learning;Visual Analytics;Interpretability;Explainability}
8. Shilpika, Bethany Lusch, Venkatram Vishwanath, and Michael E. Papka. 2025. An Incremental Multi-Level, Multi-Scale Approach to Assessment of Multifidelity HPC Systems. In Proceedings of the SC '24 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis (SC-W '24). IEEE Press, 1576–1587. https://doi.org/10.1109/SCW63240.2024.00197

- https://trustmlvis.lnu.se/
- http://captum.ai/
- https://plotly.com/building-machine-learning-web-apps-in-python/
- https://github.com/voila-dashboards/
- https://d3js.org/
- https://streamlit.io/
- https://www.tensorflow.org/tensorboard/
- https://mlflow.org/docs/latest/ml/
- https://wandb.ai/site

ARGONNE ATPESC2025 EXTREME-SCALE COMPUTING

ARGONNE TRAINING PROGRAM ON EXTREME-SCALE COMPUTING

Produced by Argonne National Laboratory, a U.S. Department of Energy Laboratory managed by UChicagoArgonne, LLC under contract DE-AC02-06CH11357.

Special thanks to the National Energy Research Scientific Computing Center (NERSC) and Oak Ridge Leadership Computing Facility (OLCF) for the use of their resources during the training event.