AI Explainability 360 toolkit for Time-Series and Industrial use cases

Presenters: Amit Dhurandhar, Giridhar Ganapavarapu, Kanthi Sarpatwar

Contributors: Sumanta Mukherjee, Natalia Martinez Gill, Amaresh Rajasekharan, Vijay Arya, Roman Vaculin

Agenda

Session I: Introduction and overview of new capabilities

1. Overview of AI Explainability and AIX360
2. Explainability for Industry 4.0 and the need for new explainers.
3. New features in AIX360 toolkit.
   a. New explainers for time-series modality.
   b. New explainers for tabular modality
4. API structure in AIX360 and modular installation for targeted explainers

Session II: Hands on applications using explainers from AIX360

1. Demo 1: Explain models for Engine fault detection using TS-Saliency, TSLIME.
2. Demo 2: Explain Energy load forecasting models using TSICE explainer.
3. Demo 3: Explain Concrete strength prediction model using NNContrastive, GroupedCE explainers.

4. Conclusion, Questions and Answers

Overview of AI Explainability and AIX360

WHAT DOES IT TAKE TO TRUST AI DECISIONS? (BEYOND ACCURACY)

AI is now used in many high-stakes decision making applications

Credit

Employment

Admission

Sentencing

Healthcare

Is it fair?

Is it easy to understand?

Did anyone tamper with it?

Is it accountable?

THE QUEST FOR “EXPLAINABLE AI”

CIO JOURNAL.

Companies Grapple With AI’s Opaque Decision-Making Process

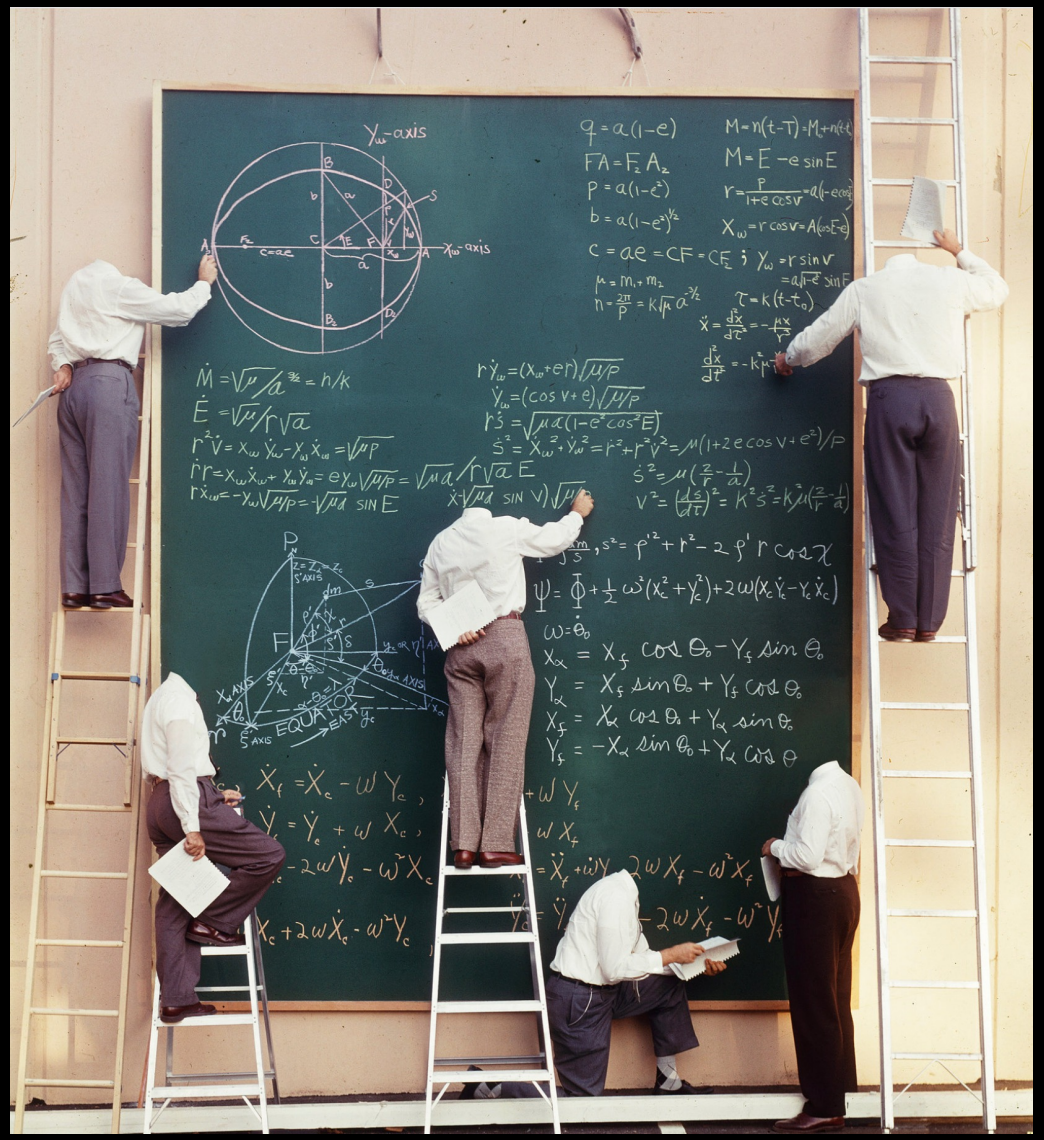
THE WALL STREET JOURNAL.

Why Explainable AI Will Be the Next Big Disruptive Trend in Business

AlleyWatch

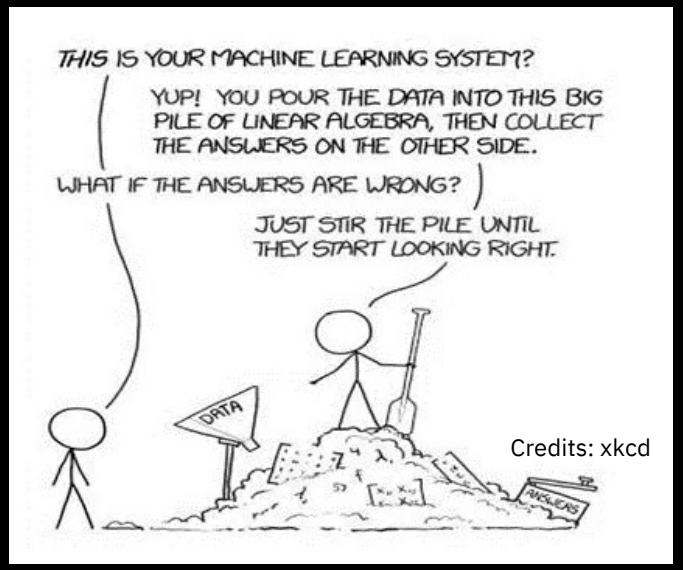
When a Computer Program Keeps You in Jail

Don't Trust Artificial Intelligence? Time To Open The AI 'Black Box'



PURPOSE OF EXPLAINABILITY

How do you *describe* the prediction strategy of a complex AI model to a human?



Credits: xkcd

Understand (model, inference, solution)

On what factors does my model base its predictions on

Assess impact of change (what-if analysis)

What is the prediction impact of changing some of the factors in my current instance

Suggest reasonable/action able alternatives

What should change in current instance state to change the model prediction (favorably)

ONE EXPLANATION DOES NOT FIT ALL

Different stakeholders require explanations for different purposes and with different objectives, and explanations will have to be tailored to their needs.

End users/customers (trust)

Doctors: *Why did you recommend this treatment?*

Customers: *Why was my loan denied?*

Teachers: *Why was my teaching evaluated in this way?*

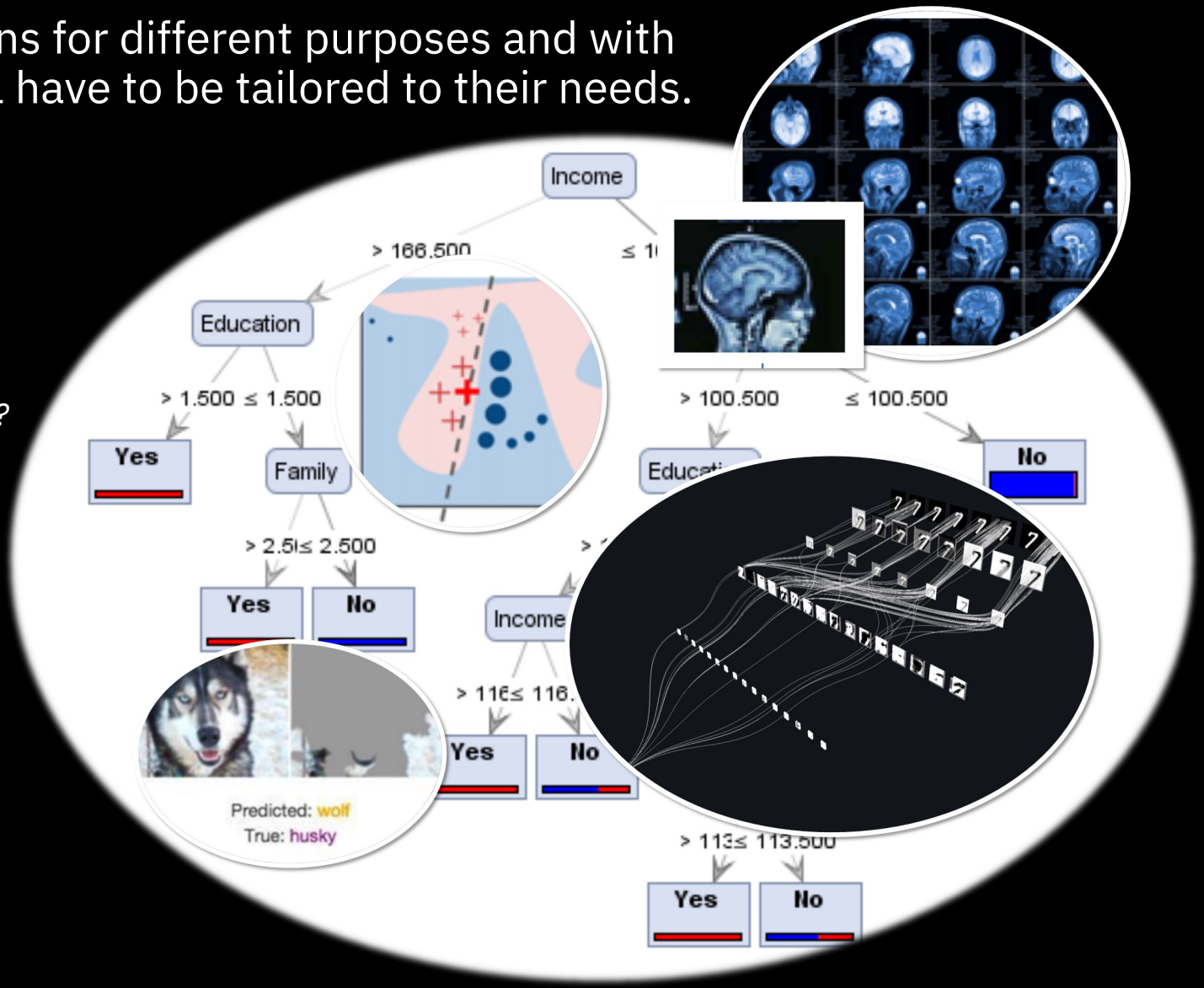
Gov't/regulators (compliance, safety)

Prove to me that you didn't discriminate.

Developers (quality, "debuggability")

Is our system performing well?

How can we improve it?



AI EXPLAINABILITY 360 (AIX360)

HTTPS://GITHUB.COM/TRUSTED-AI/AIX360

Sept. 2019

a) 9 algorithms to explain
   i) data, models
   ii) at local, global levels
   iii) for tabular, image, text

b) Interactive experience at aix360.mybluemix.net

c) 13 tutorial notebooks (finance, healthcare, lifestyle,...)

Aug. 2023 (~1500 stars, donated to THE LINUX FOUNDATION)

Additions

9 additional methods covering

a) directly interpretable

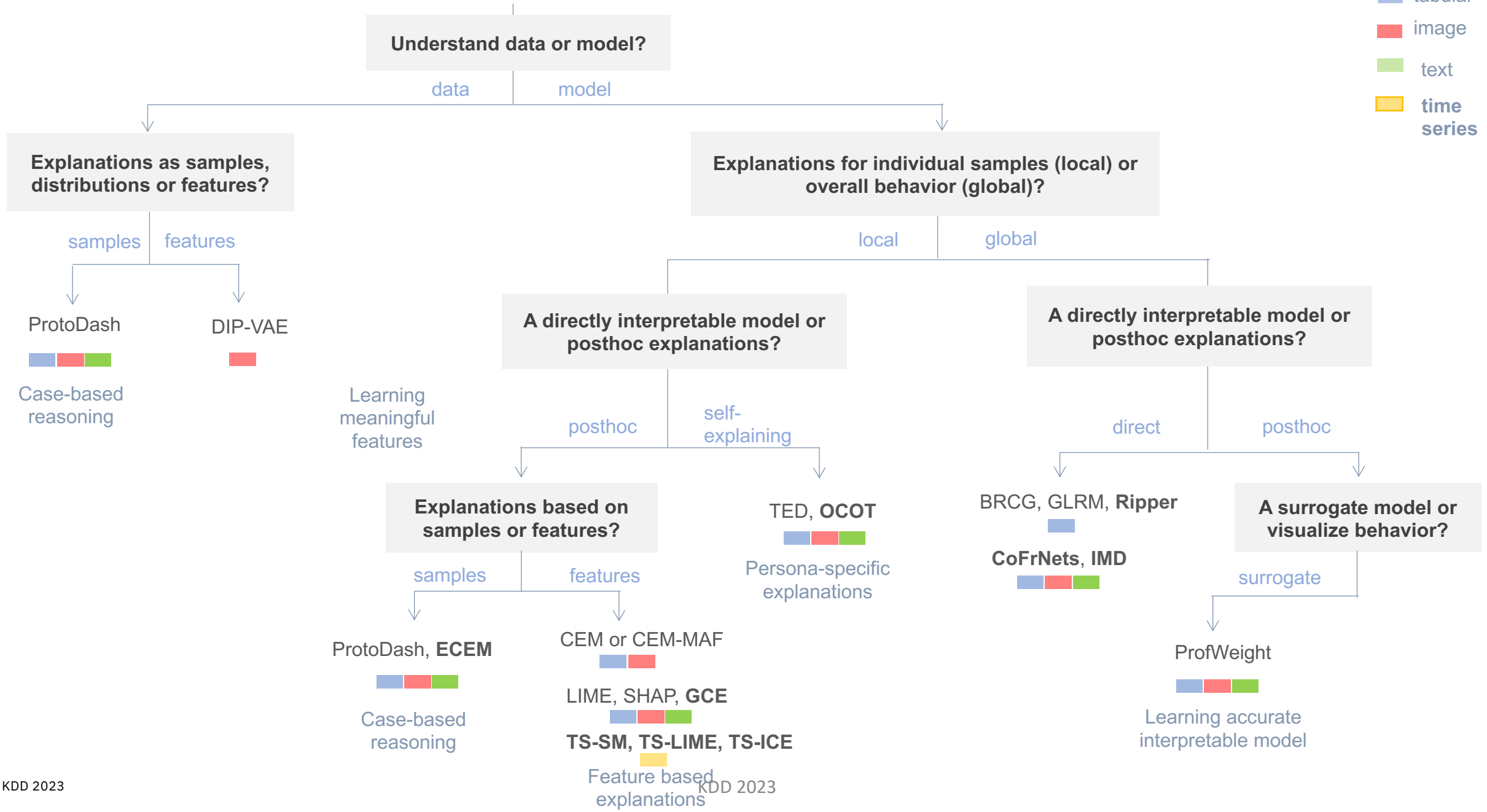
b) directly interpretable to compare models

c) local posthoc

d) time series modality

(many more in IBM internal)

New additions are **bolded**



KDD 2023

LIST OF ALGORITHMS

Data explanations

- ProtoDash (Gurumoorthy et al., 2019)
- Disentangled Inferred Prior VAE (Kumar et al., 2018)

Local post-hoc explanations

- ProtoDash (Gurumoorthy et al., 2019)
- Contrastive Explanations Method (Dhurandhar et al., 2018)
- Contrastive Explanations Method with Monotonic Attribute Functions (Luss et al., 2019)
- Exemplar based Contrastive Explanations Method
- Grouped Conditional Expectation (Adaptation of Individual Conditional Expectation Plots by Goldstein et al. to higher dimension)
- LIME (Ribeiro et al. 2016, Github)
- SHAP (Lundberg, et al. 2017, Github)

Time-Series local post-hoc explanations

- Time Series Saliency Maps using Integrated Gradients (Inspired by Sundararajan et al.)
- Time Series LIME (Time series adaptation of the classic paper by Ribeiro et al. 2016)
- Time Series Individual Conditional Expectation (Time series adaptation of Individual Conditional Expectation Plots Goldstein et al.)

Local direct explanations

- Teaching AI to Explain its Decisions (Hind et al., 2019)
- Order Constraints in Optimal Transport (Lim et al.,2022, Github)

Global direct explanations

- Interpretable Model Differencing (IMD) (Haldar et al., 2023)
- CoFrNets (Continued Fraction Nets) (Puri et al., 2021)
- Boolean Decision Rules via Column Generation (Light Edition) (Dash et al., 2018)
- Generalized Linear Rule Models (Wei et al., 2019)
- Fast Effective Rule Induction (Ripper) (William W Cohen, 1995)

Global post-hoc explanations

- ProfWeight (Dhurandhar et al., 2018)

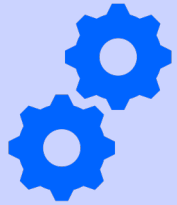
Explainability for Industry 4.0 and the need for new explainers.

Predictive Analytics for Preventive Maintenance

Preventive Maintenance

Early detection and prevention of events that

- Cause unexpected downtime
- Cause deterioration in quality of the final product
- Result in suboptimal performance of the asset, consuming more resources / energy, impact environment



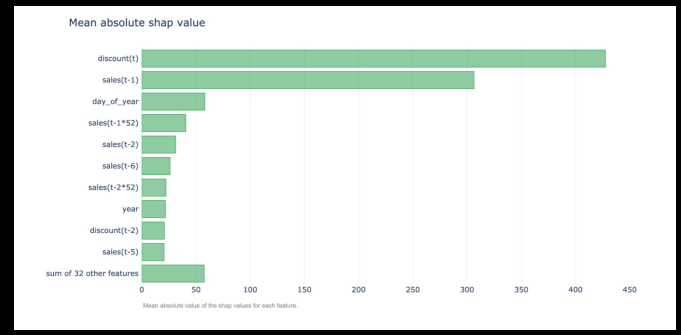
Usecase 1 (Pumps): Different kinds of pumps are used widely in various applications. Failure of pump means unexpected downtime in the operation. By monitoring operational parameters such as velocity, current (amperage), pressure, flowrate, the vibration patterns of the motor, and the temperature profile of the motor, the reliability engineer can detect the operational conditions, and predict an early failures of motor involving bearing failure and misalignment.

Usecase 2 (Turbines): Components in power turbines could fail for different reasons. By monitoring the active power, turbine speed, temperatures of the coolant at entry and exit points, and flow rate, the RE can identify potential problems with the bearing pads, and possible downstream issues such as leaks

Explainability in Preventive Maintenance

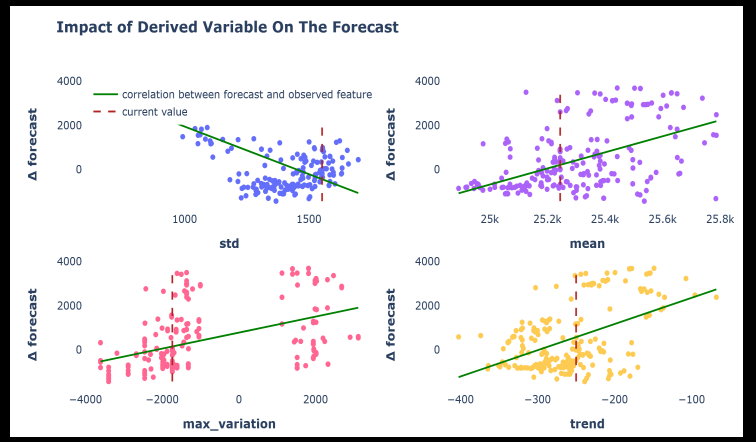
Feature Importance

What is the impact of velocity, current (amperage), pressure, flowrate, the vibration patterns on the failure predictions



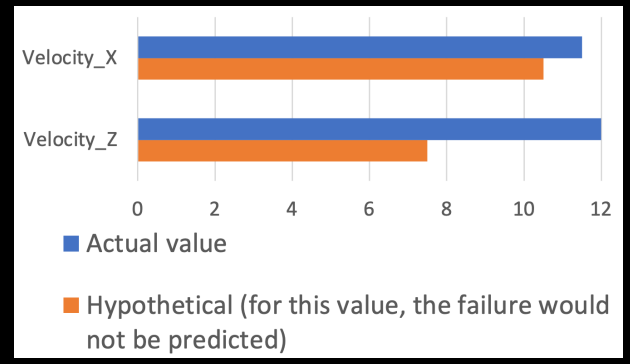
Feature Exploration

Explore the impact of different factors on the predictions: feature exploration techniques such as individual conditional plots



Suggest an actionable recourse

How can we potentially avoid a failure: suggest alterations to the current state to prevent failures using counterfactual explanations



Need for New Capabilities

Support for time-series modality:

- Often industrial datasets are time-series
- For example, pump dataset contains readings such as velocity etc. over time from sensors
- Time-series is high-dimensional and long-term correlations.

Support for Model-Agnostic Explainers:

- Industry scale model deployments are based on large scale pipelines where the best estimator is typically unknown
- MLaaS models are typically hidden for proprietary and privacy reasons
- Support for a wider-range of modeling libraries such as PyTorch, TensorFlow, SPSS etc. is highly desirable.

New features in AIX360

- ✓ Explainability tools for time-series modality
  - TS-ICE
  - TS-Saliency
  - TS-LIME

- ✓ Explainability tools for tabular-modality
  - NNContrastive
  - GroupedCE

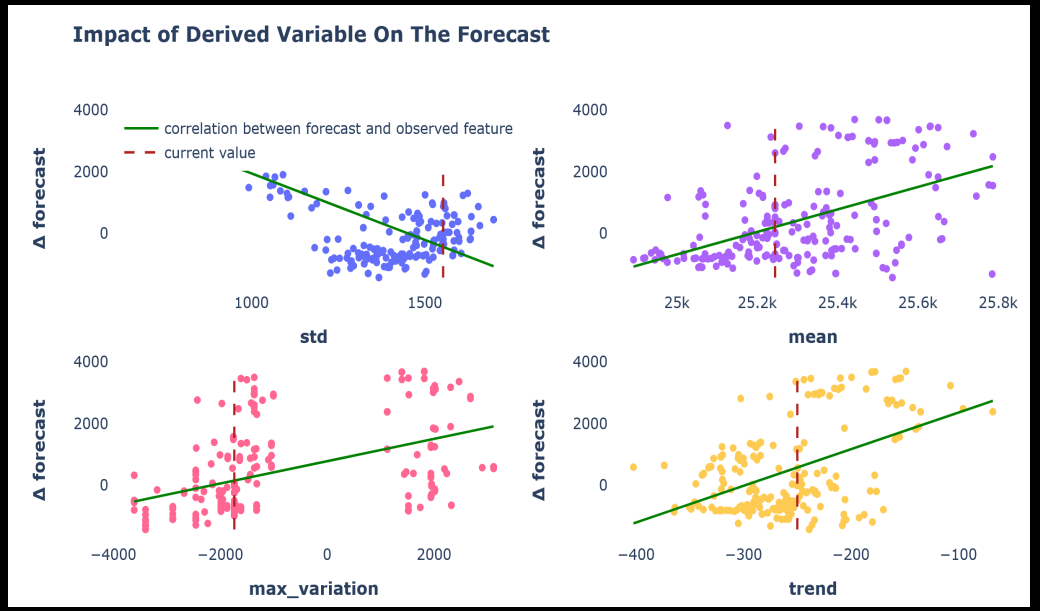
TS-ICE: Individual Conditional Expectation Plots for Time Series

Individual Conditional Expectation (ICE) - Goldstein et al. 2014

- captures how the model response varies for a given instance when a particular feature is changed.
- ICE analysis assumes features are independent and simulates new data point by varying one feature at a time keeping other features fixed.
- ICE provides more accurate picture to non-linear model response.

ICE for Time Series (TS-ICE)

- an adaptation of ICE algorithm for time series data.
- supports application of different time series perturbation techniques to construct *in-distribution time series* samples
- and uses these time series samples to capture the correlation of derived feature variations with model response.



The Approach

Generates in-distribution samples using a perturbation mechanism. We currently support the following:

1. *Frequency based perturbation*
2. *Blockbootstrap*
3. *Moving Average perturbation*

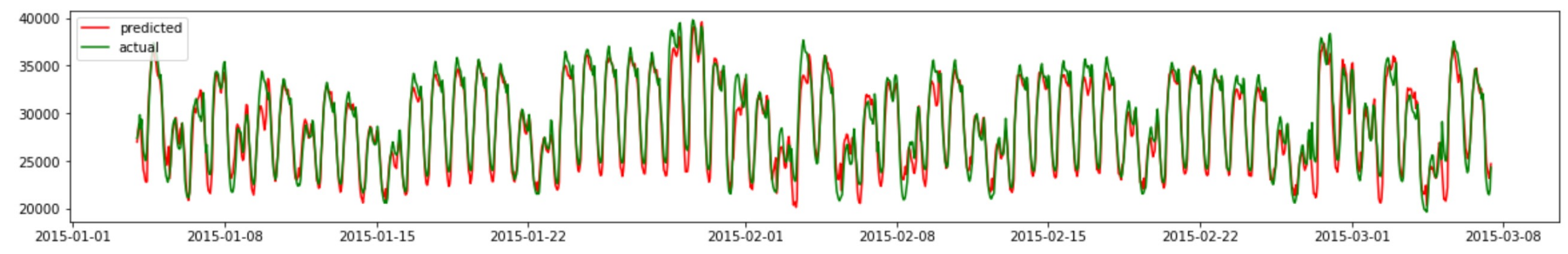
On each of these samples, TS-ICE constructs a standard, interpretable, and extensible, set of derived features (attributes), viz. mean, local variation, trends etc., as well as model forecasts.

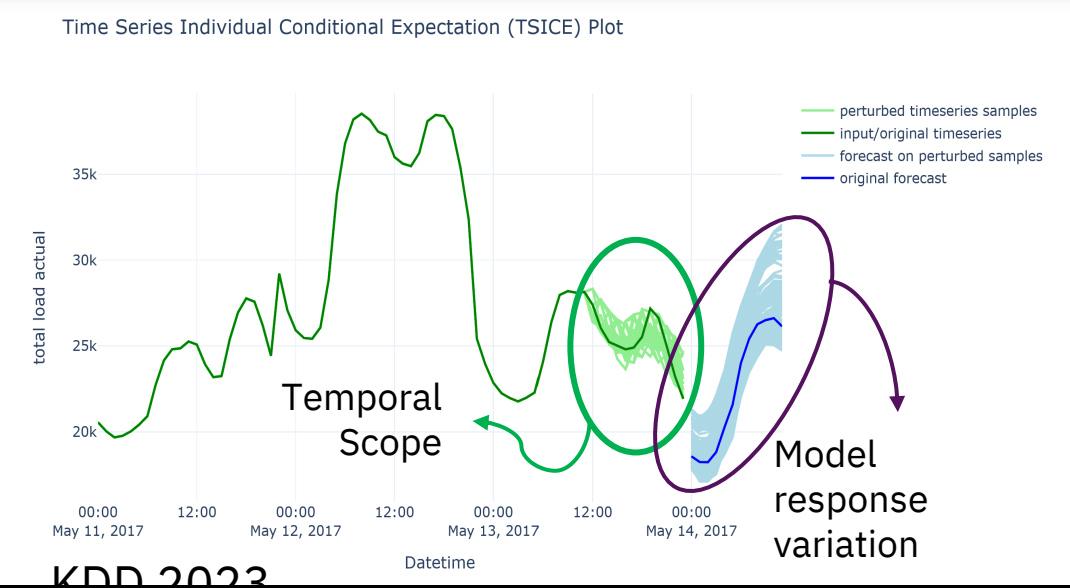
TS-ICE trains a linear model to capture model forecast in terms of the above interpretable derived properties. This enables more interpretable understanding of the model response.

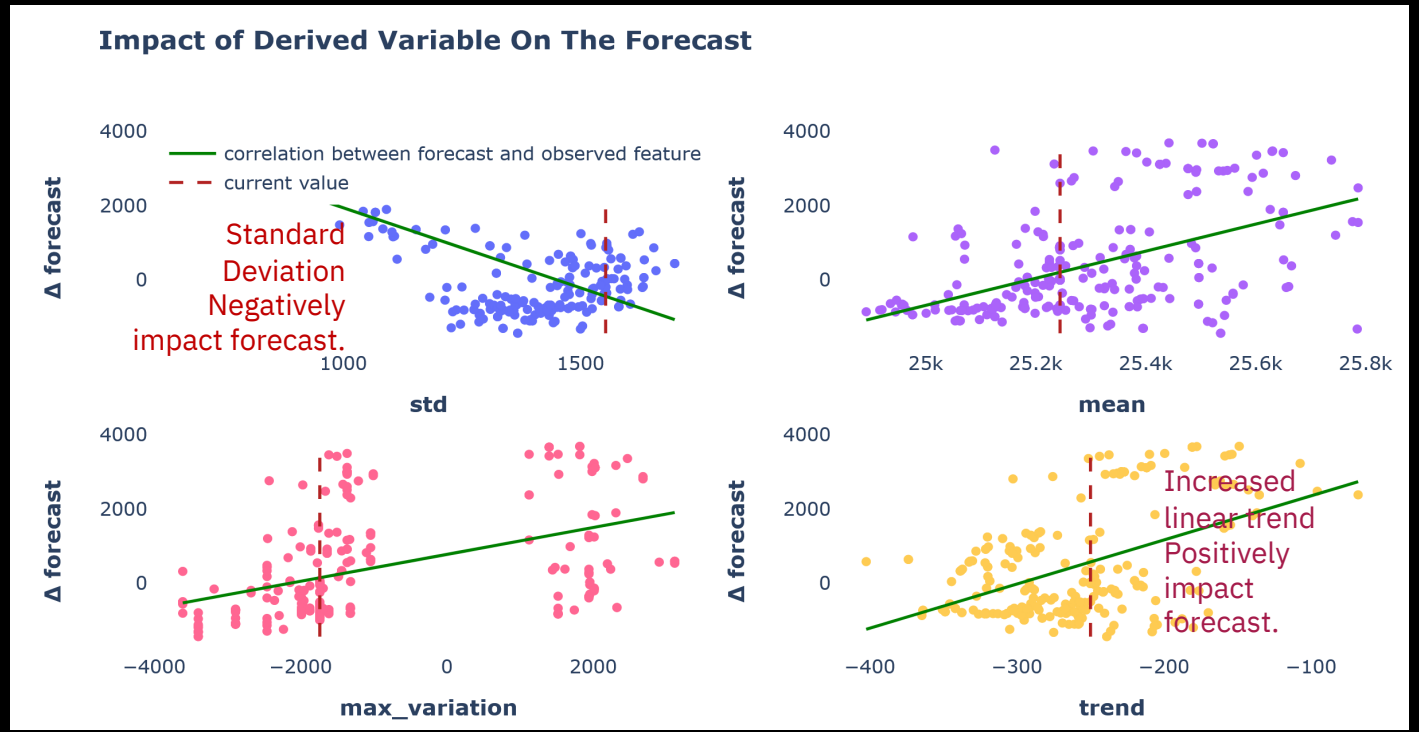
TS-ICE also analyses impact of perturbations on the predictions on a temporal resolution. This helps the end user to identify potential intervention point, and policy for model driven control.

TS-ICE: Sample Explanations

Univariate forecaster for hourly energy demand generation. Demonstrate the nature of explanation produced by tsICE algorithm



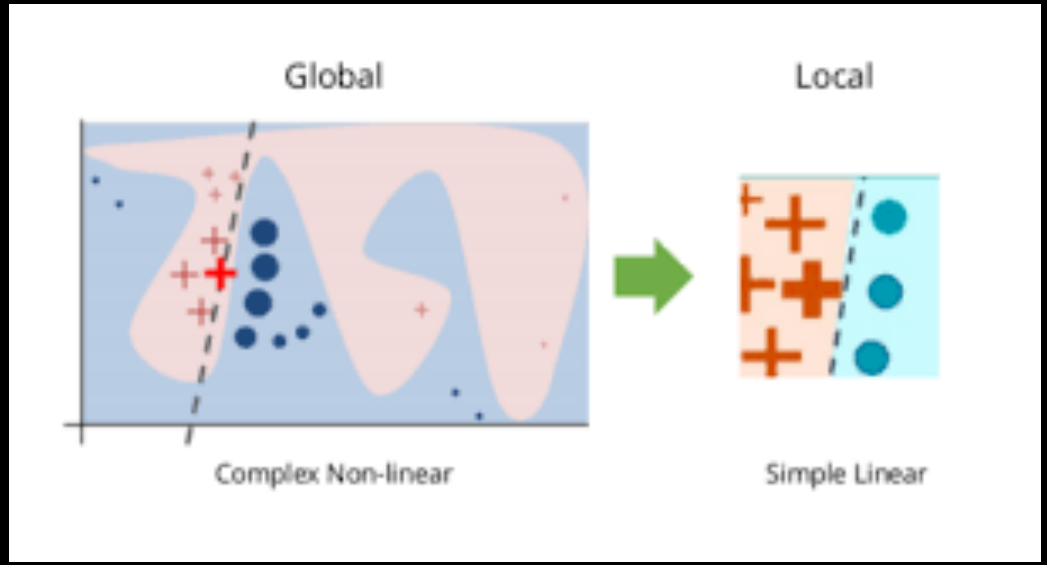




TS-LIME

LIME (Local, Interpretable, Model-Agnostic Explanations) - Ribeiro et al. 2016

- Local feature importance method
- Approximate local decision boundary by simple linear models.
- Local approximate model is trained on a neighborhood based on predefined binary interpretable features.



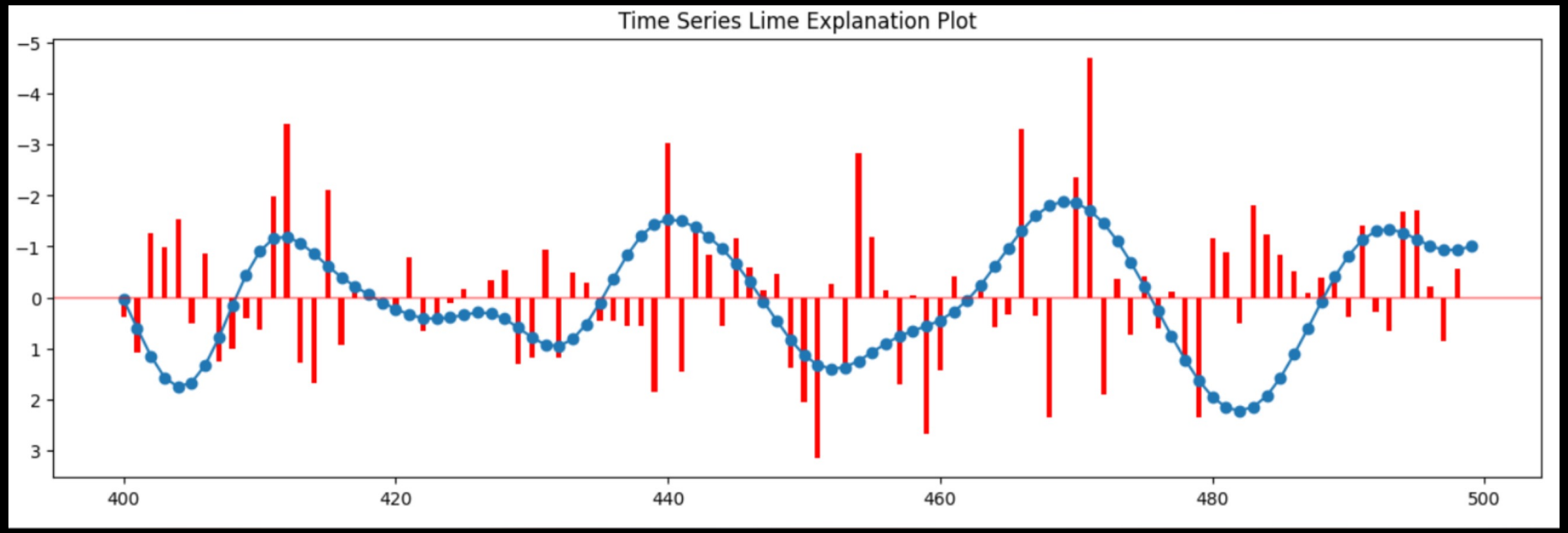
Ribeiro et al. 2016

- Defining appropriate neighborhood is important for the quality of explanations.
- For time-series modality, vanilla LIME does not work well

TS-LIME

LIME for Time Series (TS-LIME)

- TS-LIME uses time series perturbation to generate multiple time series samples. These samples are used to generate simulated data for local linear model training.
- TS-LIME allow user to specify a temporal window and explains the model output with a linear model. That describes the local response of the model in terms of few observation instance only
- Linear coefficients of the local model is the explanation generated by TS-LIME algorithm.

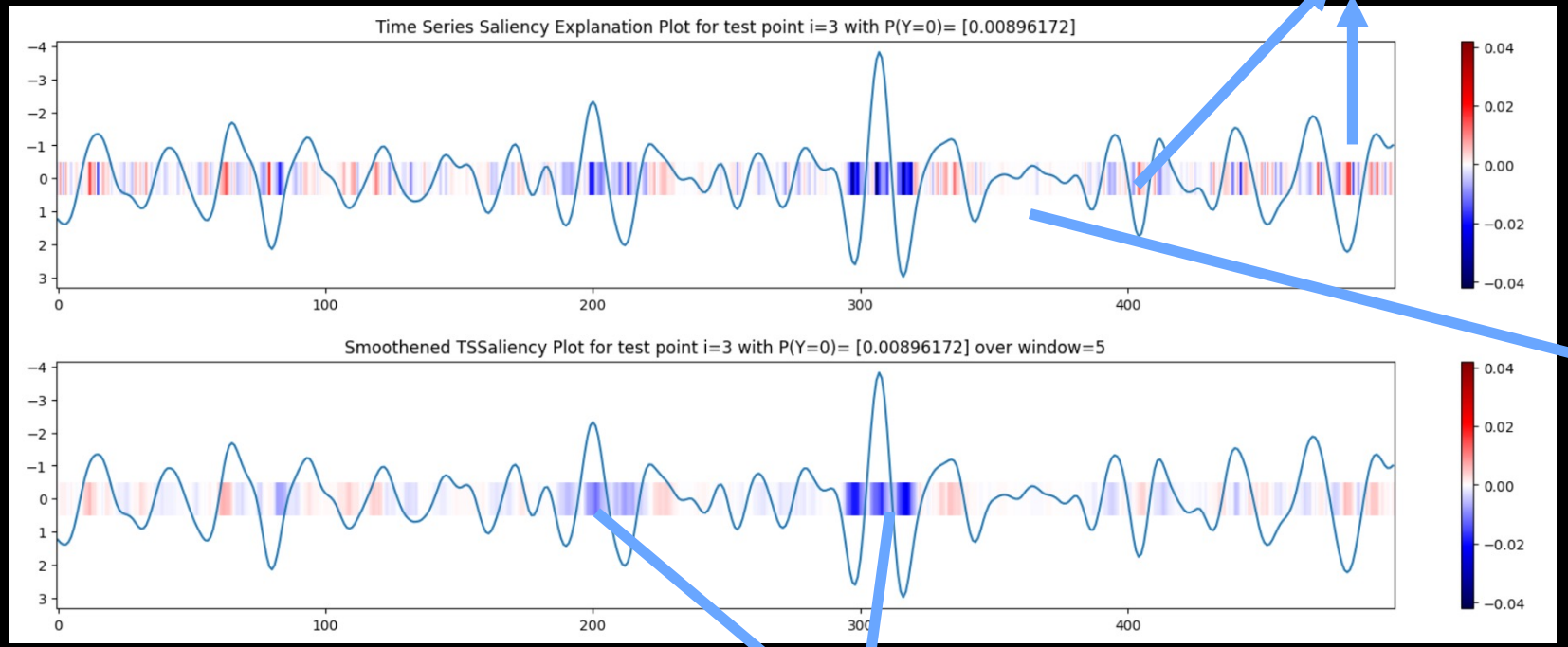


Time Series Saliency Explainer

Computing Temporal Feature Sensitivity

- ❑ Temporal impact of variates on the model predictions
- ❑ Integrated gradient based approach that computes saliency/sensitivity.
- ❑ Supports univariate and multi-variate scenarios

Highest positive impact on the model prediction



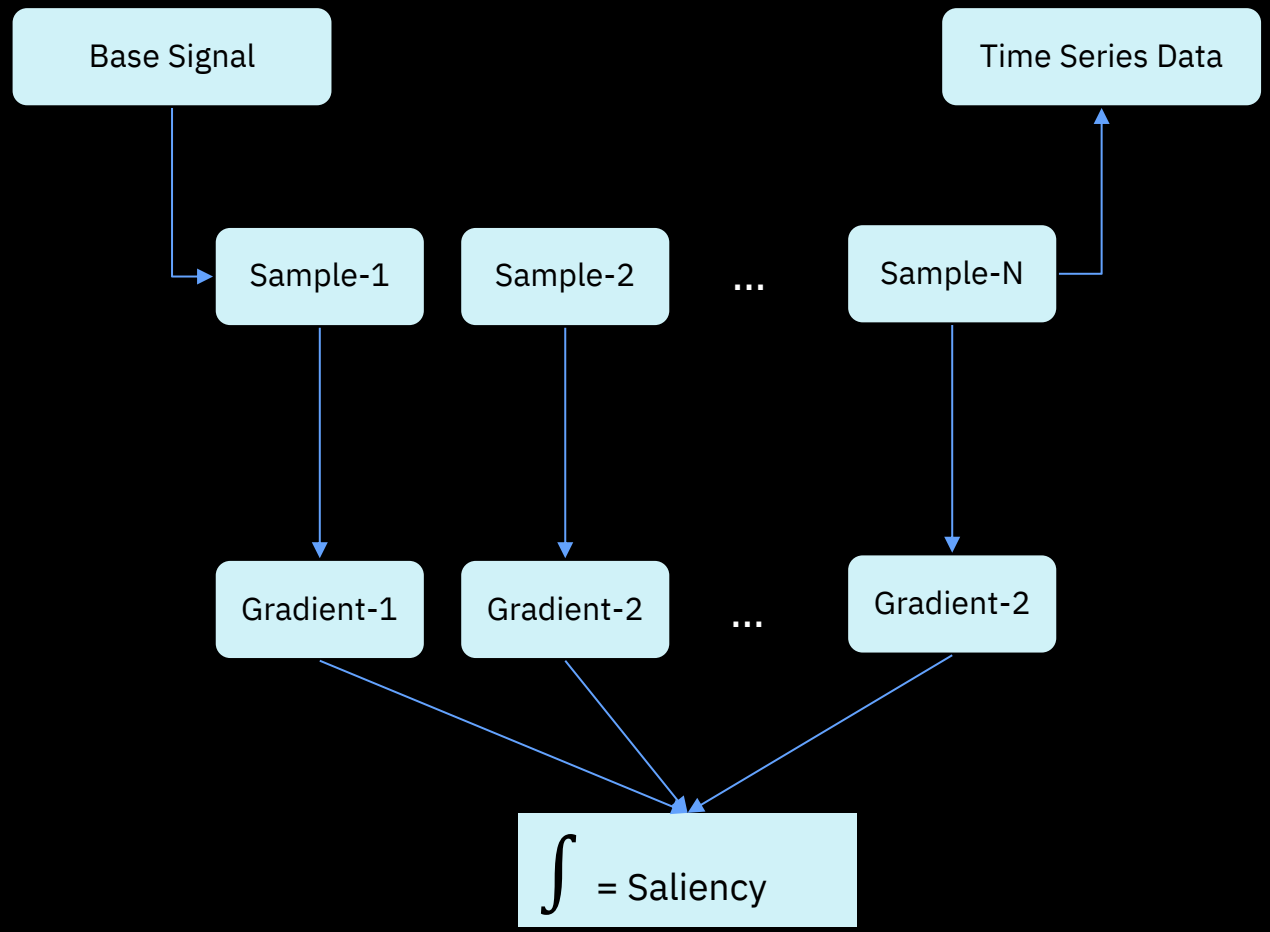
Regions of no impact on the model prediction

Highest negative impact on the model prediction

TS-Saliency

- We use Integrated Gradient as a saliency measure.
- It measures the impact of change in temporal feature values impacts model estimations.
- Integrated gradient assumes a base line definition, which is often the average feature values for tabular data, blank image for an image data.
- Integrated gradient uses a continuous feature path from the base value to current instance value and integrates the gradients over the path. These axiomatically provides each feature contribution to the model estimation from the base line.

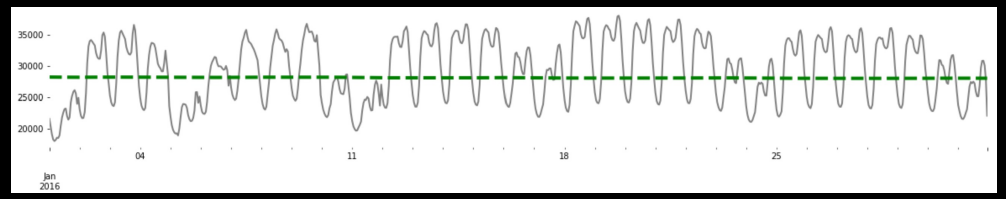
$$IG_i(x) = (x - x') \int_{\alpha=0} \frac{\partial F\left(x' + \alpha(x - x')\right)}{dx_i} d\alpha$$



TS-Saliency

Base Definition

- In image context the definition of base in uniquely defined as a null image, i.e., image containing all zero values at all pixels.
- For tabular data, data mean is used as base value.
- We use constant signal with mean strength is used as base signal.



- It must be noted this strategy make the base value data dependent. For a consistent explanation user must provide a choice proper base value while generating the explanation.

- Setting baseline as the query signal produces saliency same to model sensitivity.

Gradient Estimation

- For model agnostic implementation, we use unit sphere sampling based gradient estimation.

$$\frac{dF(x)}{dx} \approx \sum_i \frac{\left(F(x_i) - F(x)\right)}{x_i - x}; \forall\, |x_i - x| = 1$$

- User also has the option of providing a gradient function

Pseudo code

1. $X = (X_1, X_2, \ldots X_T)$ //INPUT
2. $\bar{X} = \frac{1}{T}\sum_{t=1}^{T} X_t$
3. $X_{base} = (\bar{X}, \bar{X}, \ldots \bar{X})$ //BASE VALUE
4.
5. $S = 0$
6.
7. for $\alpha \in \left[0, \frac{1}{N}, \frac{2}{N}, \ldots 1\right]$:
8. $X_{sample} = \boldsymbol{SamplePath}(X, X_{base}, \alpha)$ //Trajectory Sampling
9. $g = \textbf{ComputeGradient}(\text{Model}, X_{sample})$ //Gradient
10. $S = S + g * \frac{1}{N}$ //Integration
11. $S = S * (X - X_{base})$
12. REPORT $\frac{S}{||S||_\infty}$

NNContrastive Explainer

Exemplar Based Contrastive Explanations

Counterfactuals/Contrastive Explanations using Pertinent Negatives (PNs)

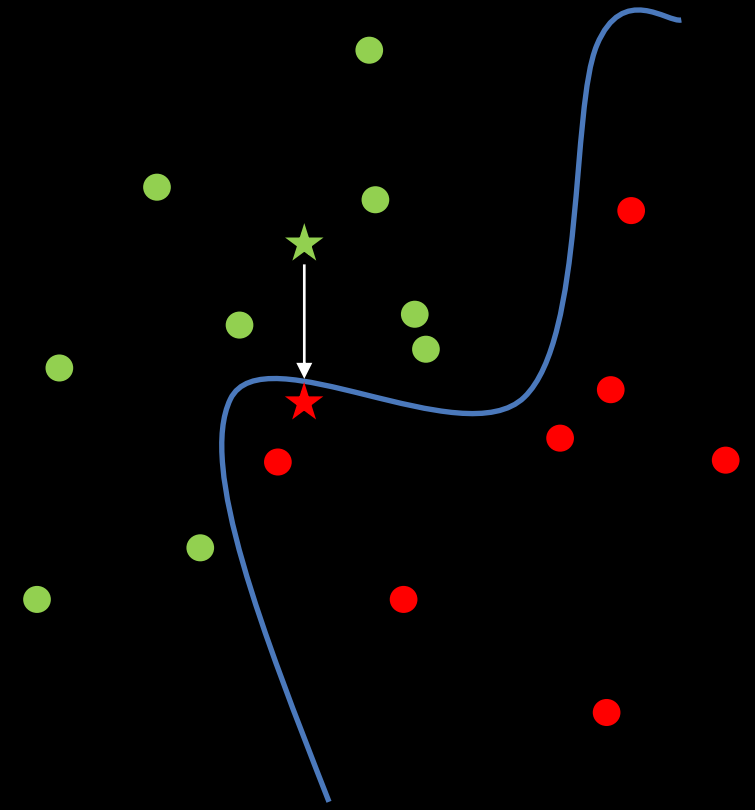
- What is the smallest change to the current input needed to *flip* the predicted label.

Key Issues:

- **Missing examples:** Existing counterfactual explainers do not always converge to a reasonable example.
- **Meaningless examples:** Further, we have found that generated PNs are often meaningless in the business context.

New approach:

- Exemplar Guided Approach: Use training examples to guide the counterfactual search.
- Especially useful when you have access to the training data



The Approach

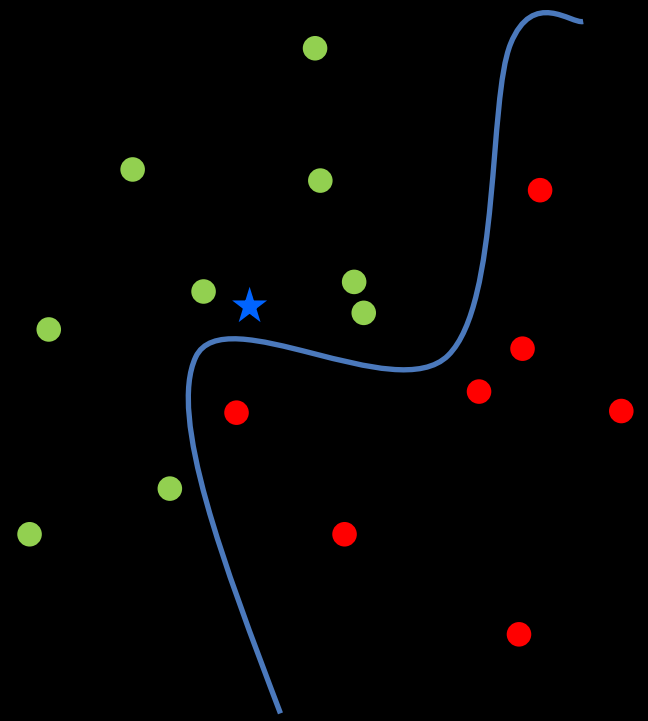
Nearest Neighbor Contrastive explanation method is an exemplar based contrastive explanation method which provides feasible or realizable contrastive instances.

For a given model, exemplar/representative dataset, and query point, it computes the closest point, within the representative dataset, that has a different prediction compared to the query point (with respect to the model).

The closeness is defined using an AutoEncoder and ensures a robust and faithful neighborhood even in case of high-dimensional feature space or noisy datasets.

This explanation method can also be used in the model-free case, where the model predictions are replaced by (user provided) ground truth.

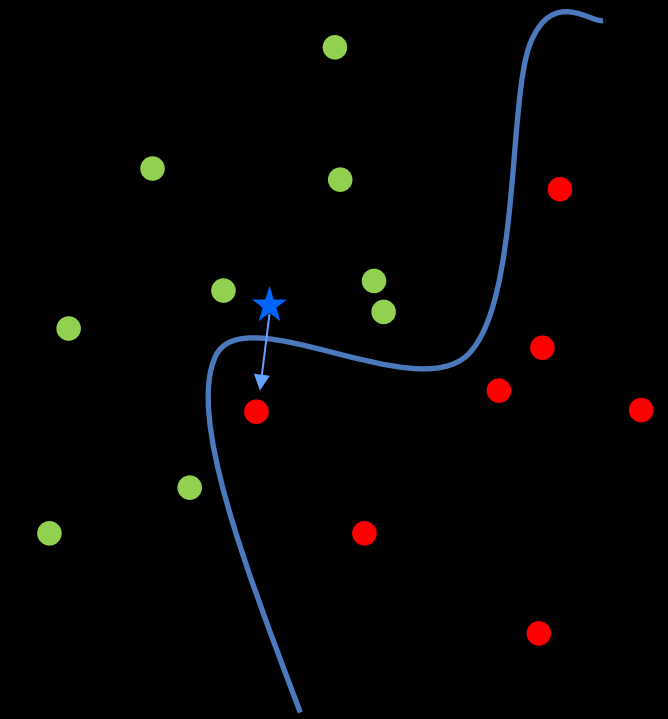
Exemplar Guided Approach



Class 1
Class 2
Query

Goal: Find a pertinent negative, i.e., a point that is closest to *query* but of a different label

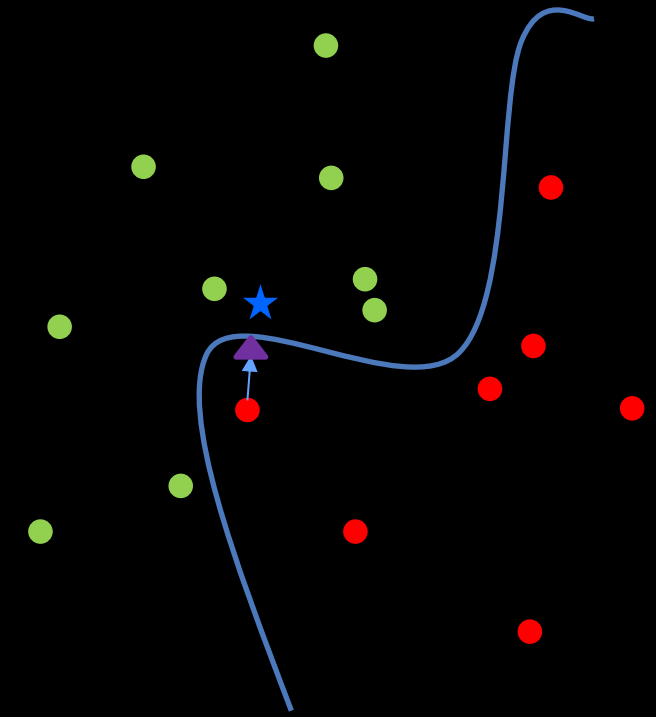
Step I



Class 1
Class 2
Query

Step I: Find a point in the training dataset that is closest to query but of a different label

Step II



Class 1
Class 2
Query

Step II: Using the *query* as base value, we find a Pertinent positive (not necessarily in the training set), i.e., closest point to *query* but of different label.

Grouped Conditional Expectation (GroupedCE)

GroupedCE is a local, model-agnostic explainer that generates grouped Conditional Expectation (CE) plots for a given instance and a set of features.

GroupedCE extends Individual Conditional Expectation (ICE) to a group of variables, by exploring the impact of varying pairs of input features jointly on the model prediction.

ICE plots show how the output of a model changes when the input features vary in a range of values.

The set of features can be either a subset of the input covariates defined by the user or the top K features based on the importance provided by a global explainer (e.g., SHAP).

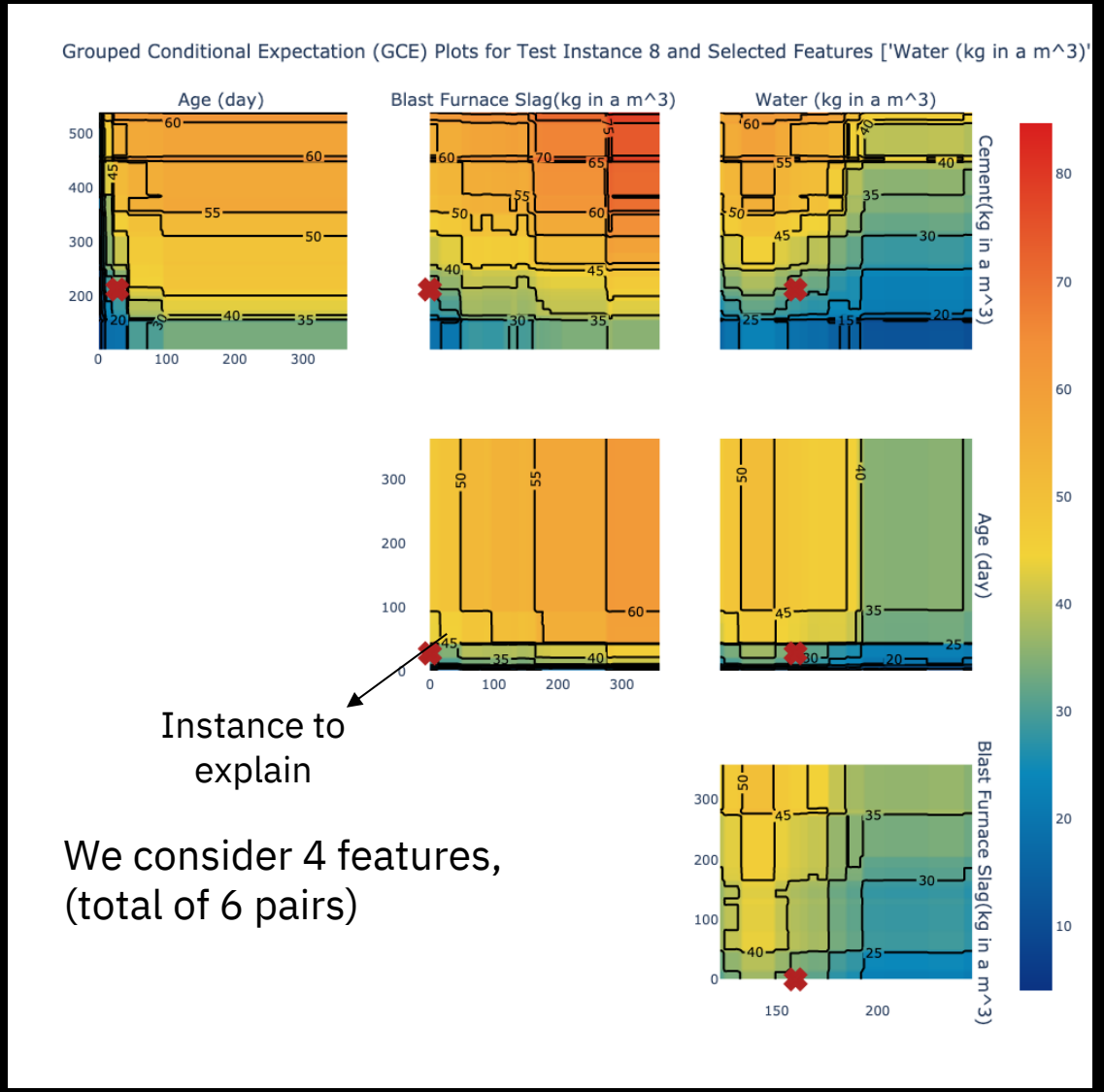
Algorithmic Description

- If no list of features is provided the top K important features are obtained with a global explainer
- For each pair of features a mesh-grid with feasible values is generated.
- The instance to explain is perturbed by replacing the pair of features values with those in the mesh-grid.
- The model prediction function is evaluated on each perturbation and stored.
- The output can be interpreted as a 3D ICE plot.

Grouped Conditional Expectation (GroupedCE)

Example: UCI Cement Dataset

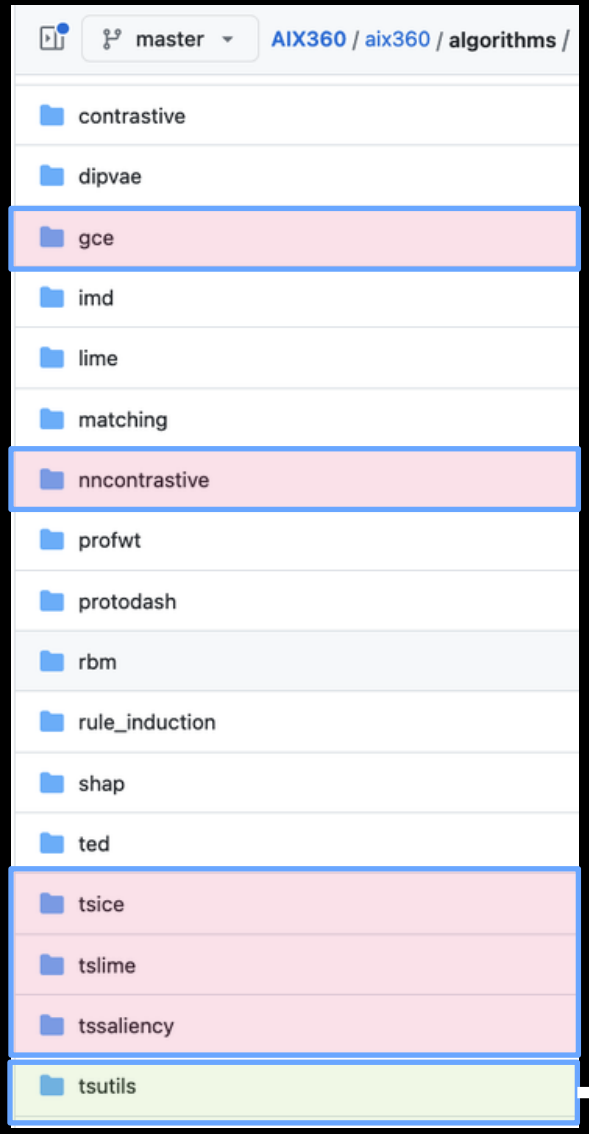
- Explain a GradientBoostRegression model that predicts the concrete compressive strength (MPa).
- The Explainer produces 6 3D-ICE plots containing the model prediction for the Top 4 important features.
- We observe that the instance to be explained (red x) has a compressive strength of ~25MPa
- According to the regression model if the Cement and/or Blast Furnace Slag in the mixture volume was higher the compressive strength would improve.
- The model also captures a dependency with Age.
- The model captures a correlation between between Water and Cement in mixture.



API structure in AIX360 and modular installation for targeted explainers

Where is everything?

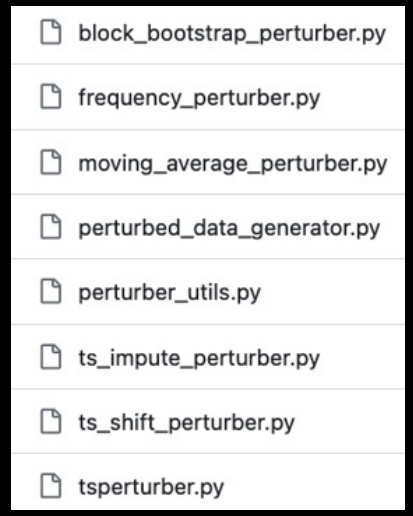
New Capabilities in AIX360



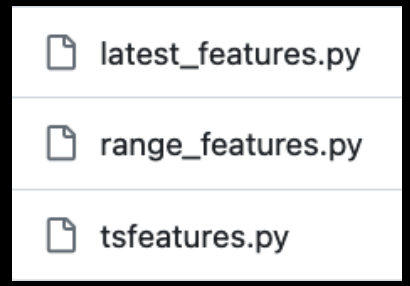
https://github.com/Trusted-AI/AIX360/tree/master/aix360/algorithms

Time Series Utilities

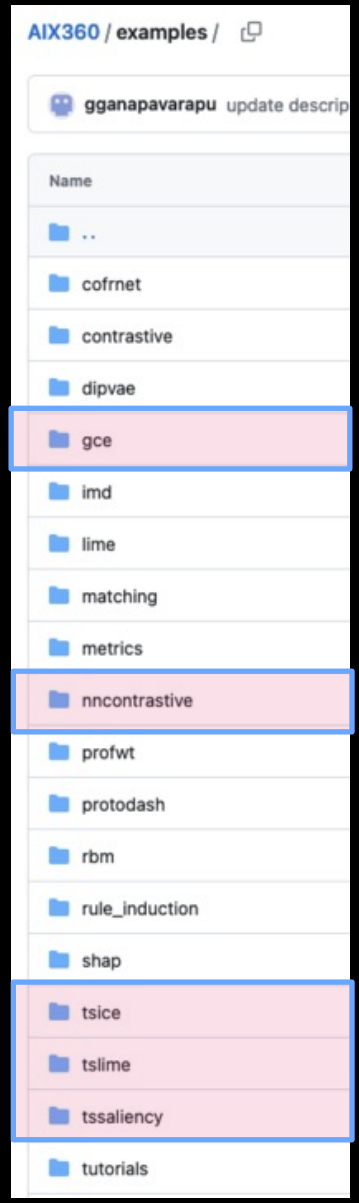
Perturbation Techniques



Feature Extraction Techniques



Example Notebooks for the new explainers



https://github.com/Trusted-AI/AIX360/tree/master/examples

API Structures for Time Series Explainers

```python
class LocalBBExplainer(ABC):

    """
    LocalBBExplainer is the base class for local post-hoc black-box explainers (LBBE).
    Such explainers are model agnostic and generally require access to model's predict function alone.
    Examples include LIME[#1]_, SHAP[#2]_, etc..

    References:
        .. [#1] "Why Should I Trust You?" Explaining the Predictions of Any Classifier, ACM SIGKDD 2016.
        Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin. https://arxiv.org/abs/1602.04938.
        .. [#2] A Unified Approach to Interpreting Model Predictions, NIPS 2017.
        Lundberg, Scott M and Lee, Su-In. https://arxiv.org/abs/1705.07874

    """
    def __init__(self, *argv, **kwargs):

        """
        Initialize a LocalBBExplainer object.
        ToDo: check common steps that need to be distilled here.
        """

    @abc.abstractmethod
    def set_params(self, *argv, **kwargs):
        """
        Set parameters for the explainer.
        """
        raise NotImplementedError


    @abc.abstractmethod
    def explain_instance(self, *argv, **kwargs):
        """
        Explain an input instance x.
        """
        raise NotImplementedError
```

AIX360 – Tabular/Image Base Class

```python
class TSLocalBBExplainer(LocalBBExplainer):
    """Base class for local black box explainers. This class extends :py:mod:``LocalBBExplainer``
    for local time series explanations. Such explainers generally are model agnostic and would
    require model's predict/forecast/scoring function."""

    def __init__(self, *argv, **kwargs):
        super(TSLocalBBExplainer, self).__init__(*argv, **kwargs)

    @abc.abstractmethod
    def _explain_instance(
        self, ts: tsFrame, ts_related: tsFrame = None, **explain_params
    ):
        raise NotImplementedError("This method is not implemented.")

    def explain_instance(
        self, ts: tsFrame, ts_related: tsFrame = None, **explain_params
    ):
        """Explain the prediction made by the time series model at a certain point in time
        (**local explanation**).

        Args
            ts (tsFrame): Input time series signal in ``tsFrame`` format. This can
                be generated using :py:mod:`aix360.algorithms.tsframe.tsFrame`.
                A ``tsFrame`` is a pandas ``DataFrame`` indexed by ``Timestamp`` objects
                (that is ``DatetimeIndex``). Each column corresponds to an input feature.
            ts_related (tsFrame, optional): The related time series ``tsFrame`` containing
                the external regressors. A ``tsFrame`` is a pandas ``DataFrame`` indexed by
                ``Timestamp`` objects (that is ``DatetimeIndex``). Each column corresponds to a
                related external regressor. Defaults to None.
            explain_params: Arbitrary explainer parameters.

        Returns:
            explanation (Union[List[Dict], Dict]): Returns a dict with explanation object.
        """

        return self._explain_instance(ts=ts, ts_related=ts_related, **explain_params)
```

AIX360 – Time-Series Base Class

Special Utilities for Inputs/Models

```python
def tsFrame(
    df=Union[pd.DataFrame, np.ndarray],
    timestamp_column: Union[str, int] = 0,
    columns: Union[List[str], List[int]] = None,
    freq: str = "infer",
    dt: Union[float, int] = None,  # is this required?
) -> pd.DataFrame:
    """Convert a pandas DataFrame to a time series data tsFrame.

    A time series associates values with points in time. We represent a time series as
    a tsFrame. A tsFrame is a pandas DataFrame that is indexed by Timestamp objects,
    that is, DatetimeIndex.
```

Utility Class for Data

```python
class Tensor_Based_Classification_Model(Classification_Model):
    def __init__(
        self,
        model: Callable,
        class_pos: int = 0,
        input_length: int = 2,
        n_features: int = 1,
    ):
        super(Tensor_Based_Classification_Model, self).__init__(
            model=model, class_pos=class_pos
        )
        self.input_length = input_length
        self.n_features = n_features

    def predict(self, X: np.ndarray, **kwargs):
        X = X.reshape(-1, self.input_length, self.n_features)
        return super(Tensor_Based_Classification_Model, self).predict(X, **kwargs)

    def predict_proba(self, X: np.ndarray, **kwargs):
        X = X.reshape(-1, self.input_length, self.n_features)
        predictions = self.model.predict(X, **kwargs)[:, self.class_pos]
        predictions = predictions.reshape(-1, 1)
        return predictions
```

Example Wrapper for Tensor based Time-Series Models

Modular installation of targeted explainers

AIX360 / setup.py

Code | Blame | 148 lines (143 loc) · 3.33 KB

```
8    extra_requires = {
81       ],
82       "nncontrastive": [
83           "pandas<2.0.0",
84           "tensorflow==2.9.3",
85       ],
86       "tsice": [
87           "pandas<2.0.0",
88           "scipy",
89           "plotly",  # required for units
90           "ipython",  # required for units
91           "kaleido",  # required for units
92           "requests",  # required for dataset and units
93       ],
94       "tssaliency": [
95           "pandas<2.0.0",
96           "requests",  # required for dataset and units
97       ],
```

```
        "tslime": [
            "pandas<2.0.0",
            "scipy",
            "requests",  # required for dataset and units
        ],
        "gce": [
            "pandas<2.0.0",
            "shap",
            "numba<=0.56",
            "requests",  # required for dataset and units
        ],
```

Example: To install NNContrastive and TS-Saliency explainer --

pip install aix360 [nncontrastive, tssaliency]

Demonstrations

Link to instructions