Explainable Artificial Intelligence

USE CASES FOR DATA TYPES

GAY PENG REND, JONATHAN EDUARD CHUA LIM, TEO MENG YUAN, ZHANG JIE XIONG
REPUBLIC POLYTECHNIC-NVIDIA ARTIFICIAL INTELLIGENCE INNOVATION AND TECHNOLOGY PROGRAMME | COMPANY SPONSOR: AI2LABS (PART OF YOOZOO SINGAPORE)

Table of Contents

INTRODUCTION

With the rapid technology advancement in IT, computation power has increase exponentially in the recent years. Such accelerated pace of change allows the rapid adoption and integration of Artificial intelligence into society - what seems like an impossible feat just a few decades ago. We have seen the development of expert system, integrating deep domain knowledge of expert into the rule engines that users can turn to, much like how user were to seek advice from the expert themselves, however it is limited to the expert's input which can varies from various domain. AI is the development of machine learning to encompass rational problem solving thru model training on historical dataset to obtain sufficient accuracy to predict and recommend results. By leveraging on the ever increasing of computational power of computer, we are able to develop various new techniques and model to push for higher accuracy through the means of training larger datasets. For the practicality of AI deployment, there is the need to fill the explainabilty gap to support the decision process.

The field of eXplainable Artificial Intelligence (XAI) creates that possibility to increase our visibility and understanding on how the AI models or systems make the prediction that they do. It helps us in many and various aspects to understand the black box algorithms and bridge that gap of knowledge, which in turn not only creates more trust in AI but also gives developers, engineers and other AI practitioners more insights that can help improve the work that they do.

In this write up, we present some of the XAI techniques amidst the countless number of them that are useful immediately for simpler applications onto AI models or systems, covering all four datatypes namely; Tabular, Image and Video, Time-Series, and Natural Language Processing. The below tests and application of the XAI techniques are not representative of the best or ideal methods to use, but what we strive to achieve is the usage of some of the techniques we found to provide good results of the explainability and the visualization that would be directly applicable to stakeholders namely the AI engineer, the end-user and the auditors.

DATA TYPES AND XAI METHODOLOGIES

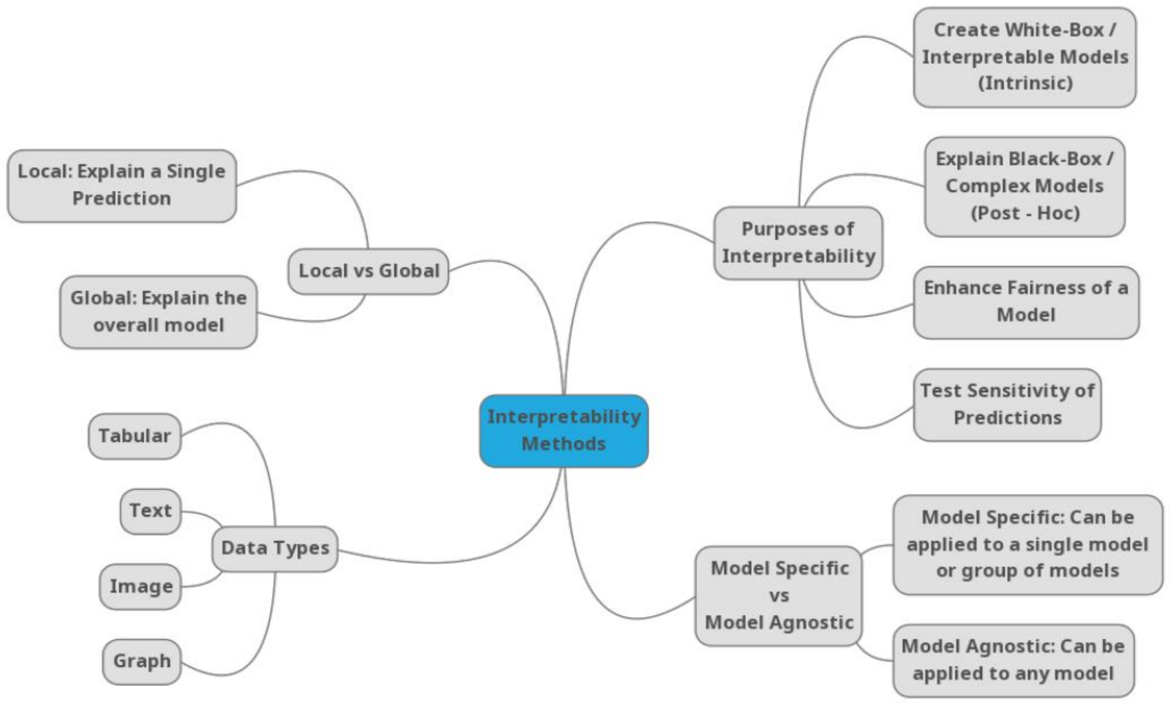

Interpretability Methods
Local vs Global
Local: Explain a Single Prediction
Global: Explain the overall model
Purposes of Interpretability
Create White-Box / Interpretable Models (Intrinsic)
Explain Black-Box / Complex Models (Post - Hoc)
Enhance Fairness of a Model
Test Sensitivity of Predictions
Data Types
Tabular
Text
Image
Graph
Model Specific vs Model Agnostic
Model Specific: Can be applied to a single model or group of models
Model Agnostic: Can be applied to any model

Figure i.1 Taxonomy mind-map of Machine Learning Interpretability Techniques

Before we approach any given task, there is a need to have a clear direction to what we are supposed to achieve, understanding the stakeholders as well as the limitation due to the different data type present, hence limiting to the type of models involved.

It can be said that XAI is a branch of machine learning, hence there is a similar interpretation method. The above mind-map highlight the 4 common direction of interpretating any given task, which can be by the area of scoop (local vs global), the functionality, specification, and requirements (the purpose), model suitability (specific or agnostic) be it for black box or white box model, and lastly by the various data types that is present.

With that being said, the intention of using XAI methodology is to simplify the understanding to various stakeholder and mass whom might not be equipped with deep domain knowledge, to understand and justify the prediction or outcome base on the “cause-and-effect”, and it is that reason XAI interpretability are contain within the local perimeter, for the explanation of the single prediction, one at a time given that model that are less complex are “easier” to interpret without going too deep into the justification.

The type of data presented will also plays an important role in the decision making process of choosing the right model, from model specific white box to black box model, such as the use of local surrogate model (LIME).

FRAMEWORK

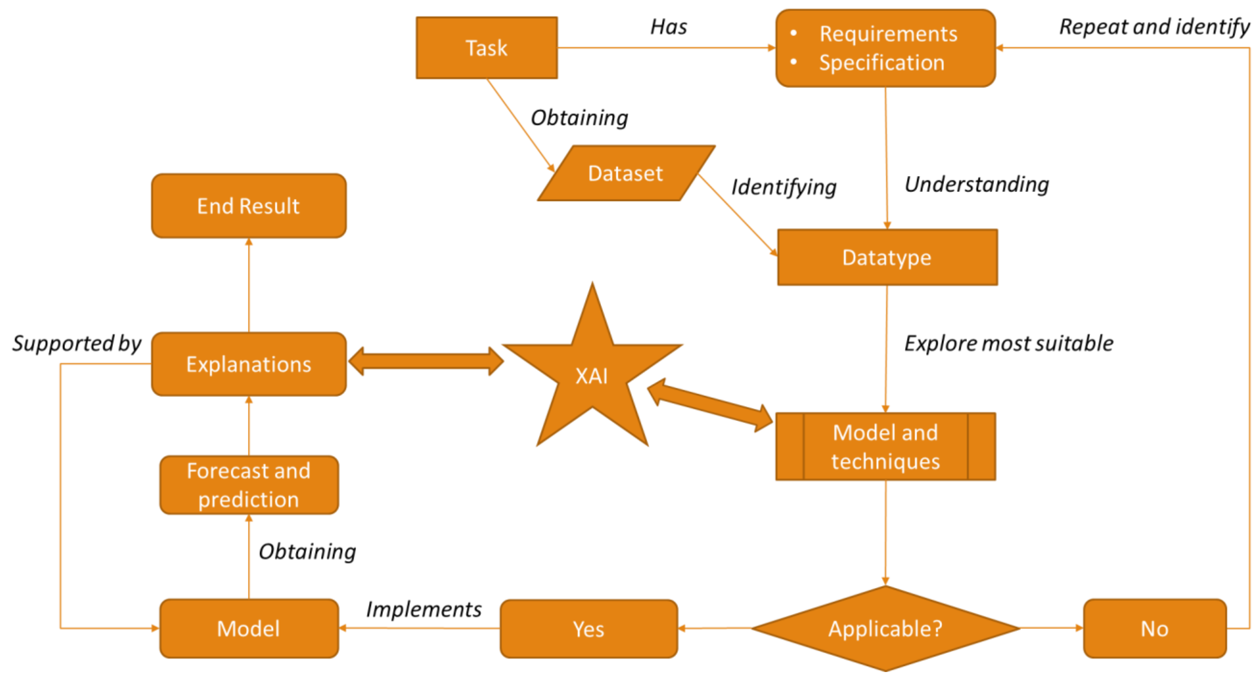


Figure i.2 Proposed framework on the team XAI workflow

We have come up with a framework on how the team approaches any given task. It illustrates the workflow and the thinking process at different stages. At the initial stage, we place great emphasis on the stakeholders' need in the requirements and work towards the target, as well as understanding and knowing the datatypes involved in the various tasks, namely NLP, tabular, time series and imaging.

As different models yield different results, there is the need to explore and decide models that provide higher accuracy that are appliable to the stakeholders' interest without compromising on the portion of justification and explanation at the prediction. This might also be the most crucial stage as the team might be prone to steering away from the objective in pursuit of perfection in the result.

TABULAR DATA

In the processing of tabular data, for smaller datasets, often Machine Learning (ML) models are used instead of the more complex Deep Learning models. These models, namely, linear regression, logistic regression decision trees, kNN, k-Means, random forest and the various gradient boosting algorithms to name a few, are deployed to solve a regression or a classification problem.

The benefit of using most ML models is that they are generally intrinsically interpretable. To give a simple insight on what is meant by intrinsically interpretable, let’s look at a typical linear regression model as shown by Molnar [1]

Linear models can be used to model the dependence of a regression target y on some features x. The learned relationships are linear and can be written for a single instance i as follows:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \epsilon$$

The predicted outcome of an instance is a weighted sum of its p features. The betas (βj) represent the learned feature weights or coefficients. The first weight in the sum (β0) is called the intercept and is not multiplied with a feature. The epsilon (ϵ) is the error we still make, i.e. the difference between the prediction and the actual outcome.

Here we can see that, for the value of y to change, all the betas carry a direct effect on each of the x features that also changes the value of y. It can be said that y is directly related to the changes in beta0, beta1, …, and betaP, and here we can directly interpret the model accordingly with this linear relationship.

The same interpretability can be said for the Logistic Regression Model except the relationship changes to y is directly related to the logarithmic coefficient of the x-features. Decision trees, random forests all carry the same idea of intrinsically interpretability (though with different complexity) but as we move into gradient boosted algorithms, the intrinsic nature of the interpretability decreases.

The next subsection will seek to gather insight into some of the ML models followed by actual XAI techniques as a black-box approach.

MODEL BEHAVIOR THROUGH ITS INTRINSIC INTERPRETABILITY NATURE

In this subsection, we study five models and the behaviors it present, Quadratic Discriminant Analysis (QDA), Logistic Regression (LR), Random Forest (RF), Decision Trees (DT), and XGboost (XGB). Each of these models are created using PyCaret library and the dataset is from Sklearn’s Breast Cancer binary classification of Malignant/Benign. The dataset consists of 30 features and 569 entries.

Decision Boundary

By condensing all the features into a 2-Dimensional space, we can see the behavior of each of the models through the decision boundary.

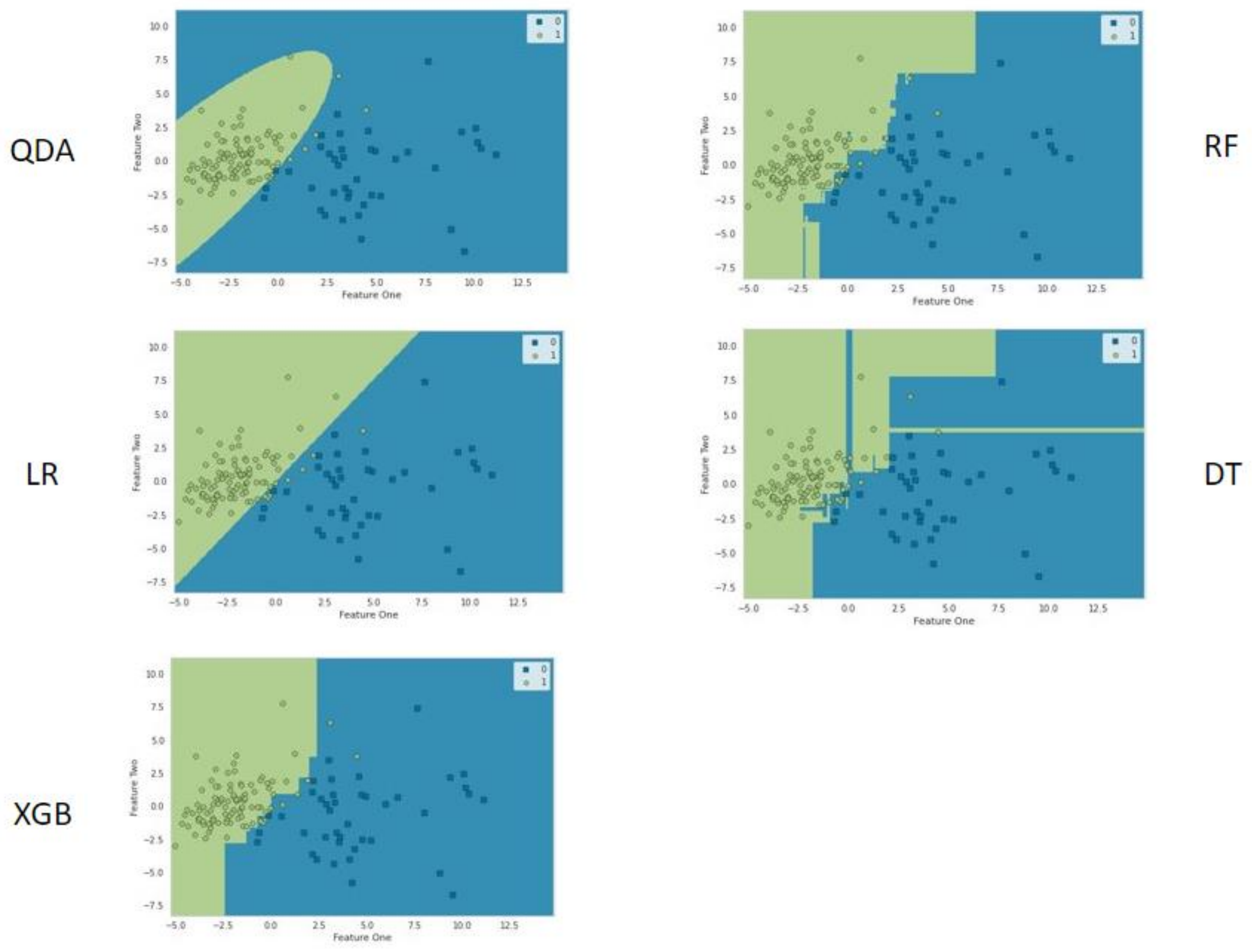


Figure 1.1: Decision Boundary for each of the models showing how each model's classification works

As observed in fig 1.1, we can understand intuitively from QDA and LR since the quadratic and linear nature of decision making will result in plots that are like what is shown. But as we go into RF, DT, and XGB, that intuitiveness of understanding decreases but is still generally acceptable. Should the engineer disagree with the way the model behaves, this plot provides that insight into how the model behaves.

RadViz for 5 features

The RadViz plot allows more features to be represented should greater understanding be required.

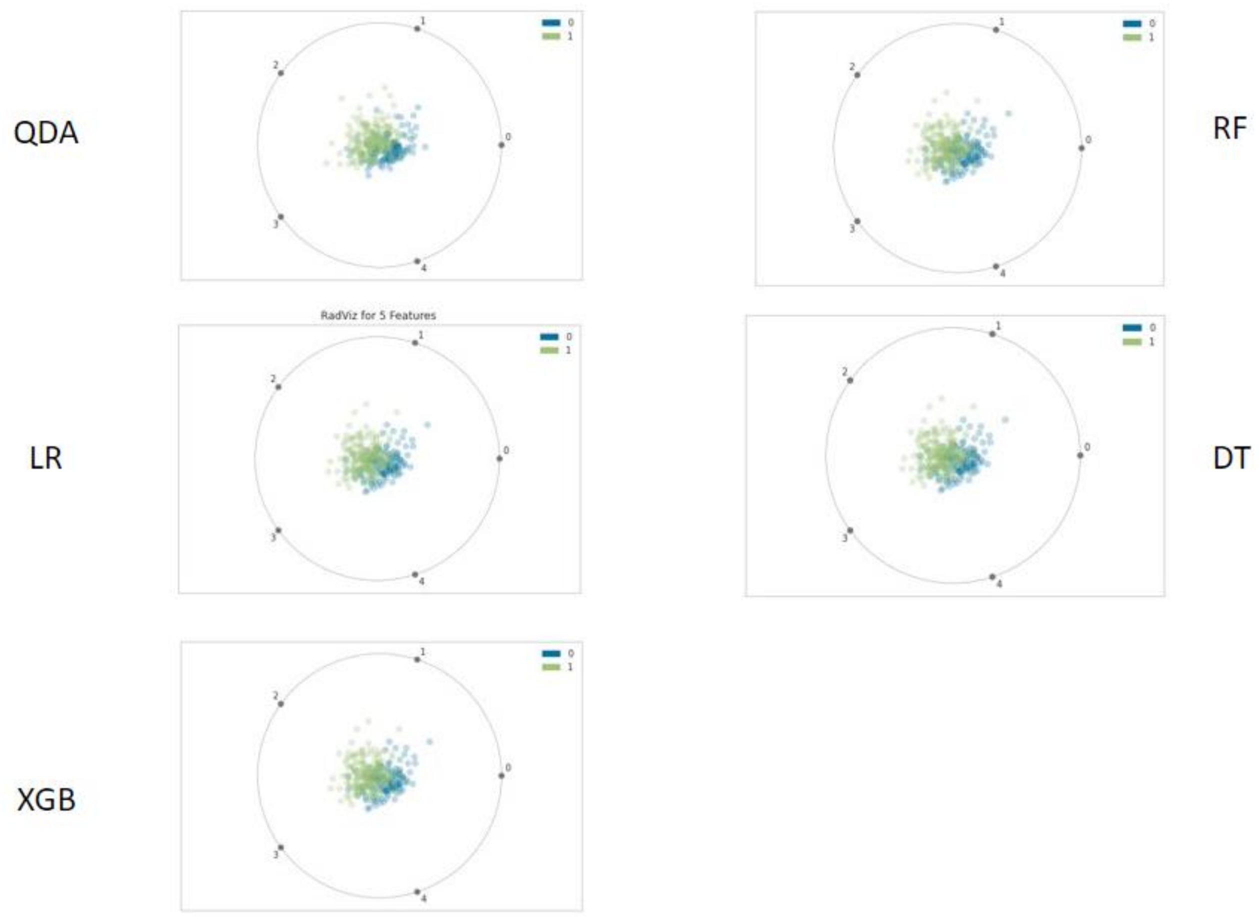


Figure 1.2: 5-feature RadViz plot

In fig 1.2, the radviz plots in each feature dimension uniformly around the circumference of a circle. It then plots points on the interior of the circle such that the point normalizes its values on the axes from the center to each arc. In a static image like this does not give much understanding into the model, but when the radviz is plotted with an interactive and adjustable feature space, there will be interesting results on how each of the decision point changes.

Model Insights

Having access to the way the model makes its prediction, there are further insights that can be taken from the model. All of these provide a look into the model to understand deeper how it behaves. The next few plots are examples of such.

Model Insights: Calibration Plots (Reliability Curve)

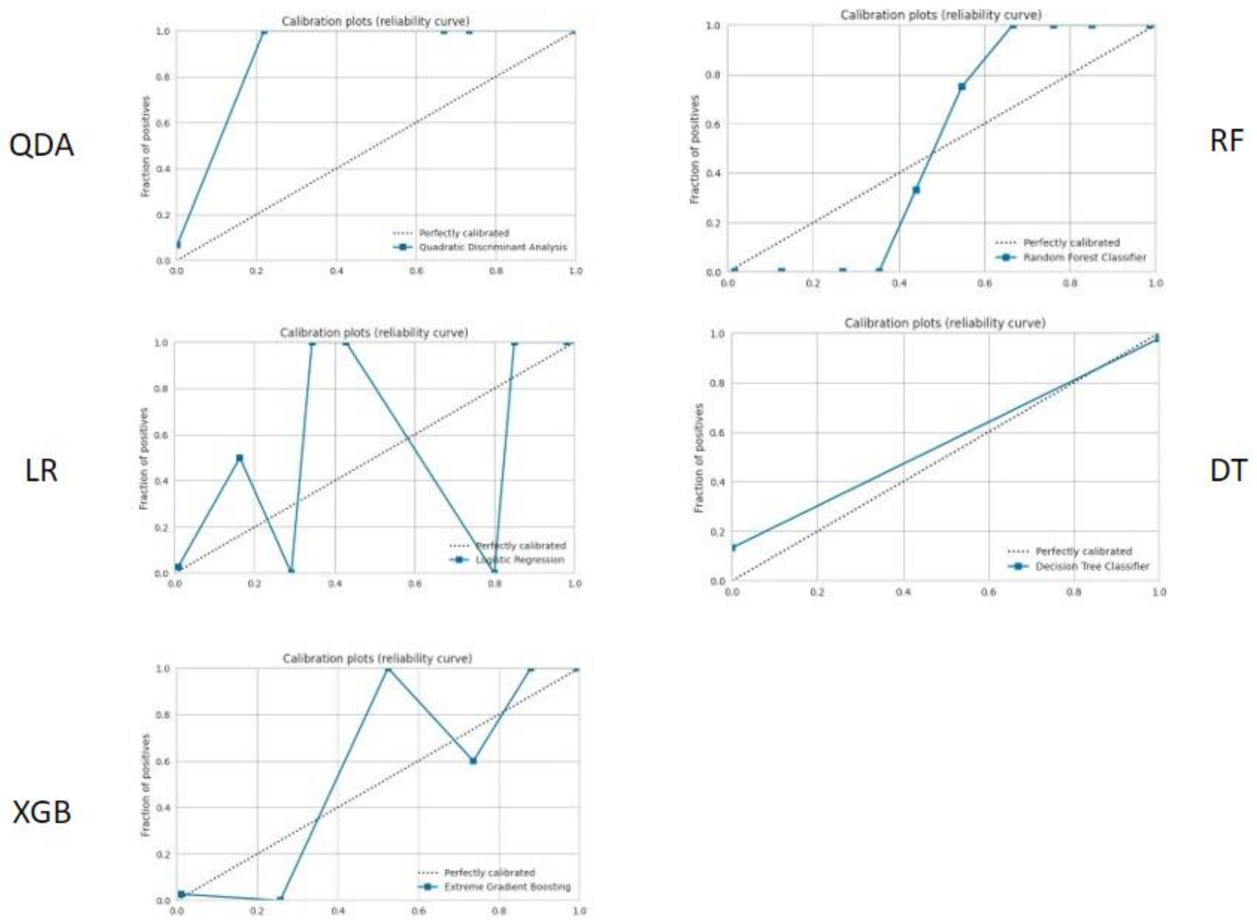


Figure 1.3: Calibration plots/reliability diagram showing the line plot of the relative frequency of what was observed (y-axis) versus the predicted probability frequency (x-axis)

Instead of predicting class values directly for a classification problem, it can be convenient to predict the probability of an observation belonging to each possible class. Predicted probabilities that match the expected distribution of probabilities for each class are referred to as calibrated.

A better calibrated model is one that follows the dotted line closer. Fig 1.3 shows the degree of calibration for each of the models.

Model Insights: The Kolmogorov-Smirnov (KS) plot

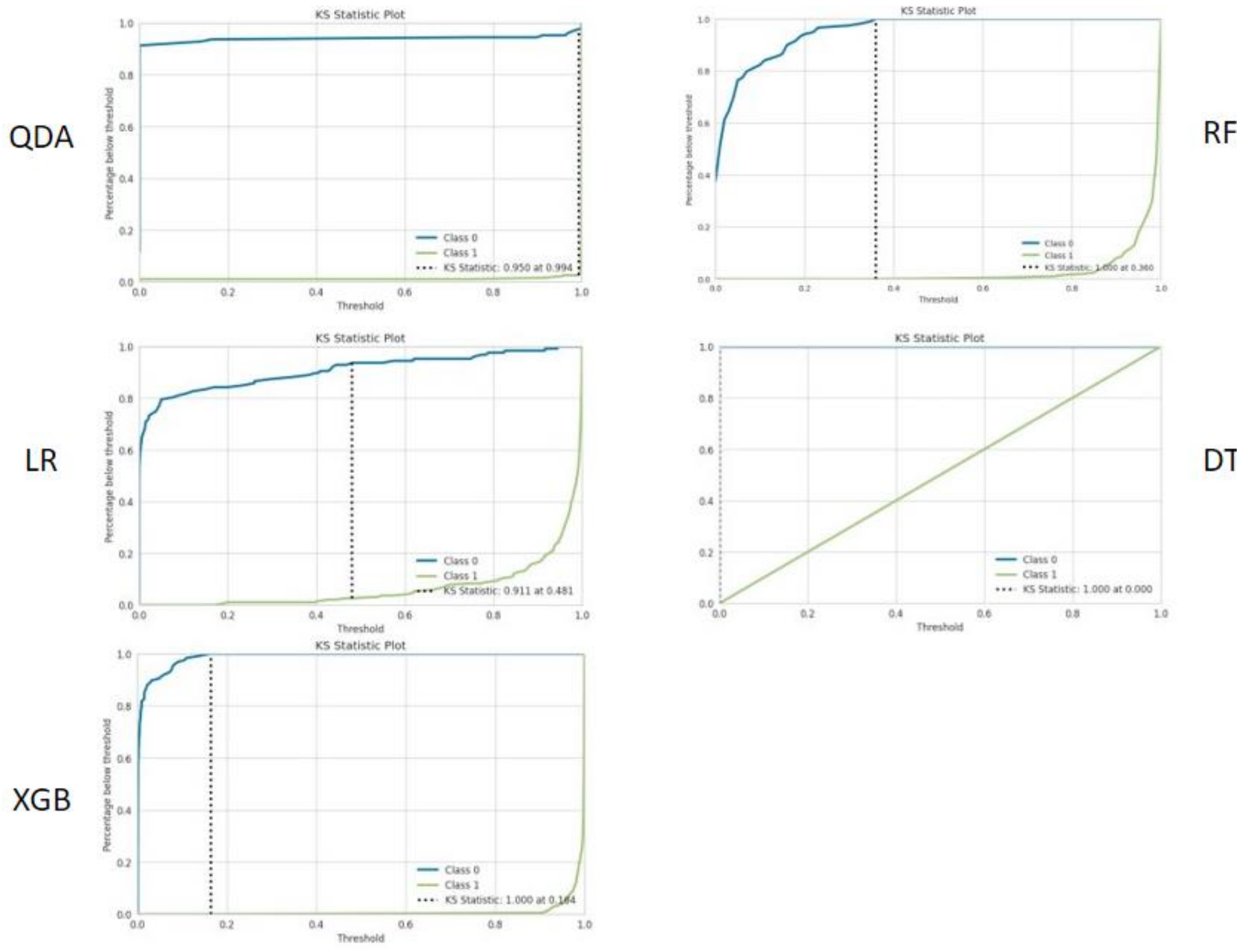


Figure 1.4: The KS plot for each model assessing the hypothesis that a random sample (of numerical data) came from a continuous distribution that was completely specified without referring to the data.

As with most tabular data, there is the base assumption that the data follows a certain cumulative distribution function (CDF) which would then fit into an ML model without issue. However, sometimes that assumption is wrong, and the ML model performs badly or worse even inaccurately.

The KS plot is a statistical method to analyze any data and determine whether it fits into that CDF without taking reference to the data itself. Hence, the KS statistic for two samples is simply the highest distance between their two CDFs, so if we measure the distance between the positive and negative class distributions, we can have another metric to evaluate the classifier. The closer the value of the KS statistic is to 1, the better the model behaves in separating the classification.

Model Insights: Feature Importance

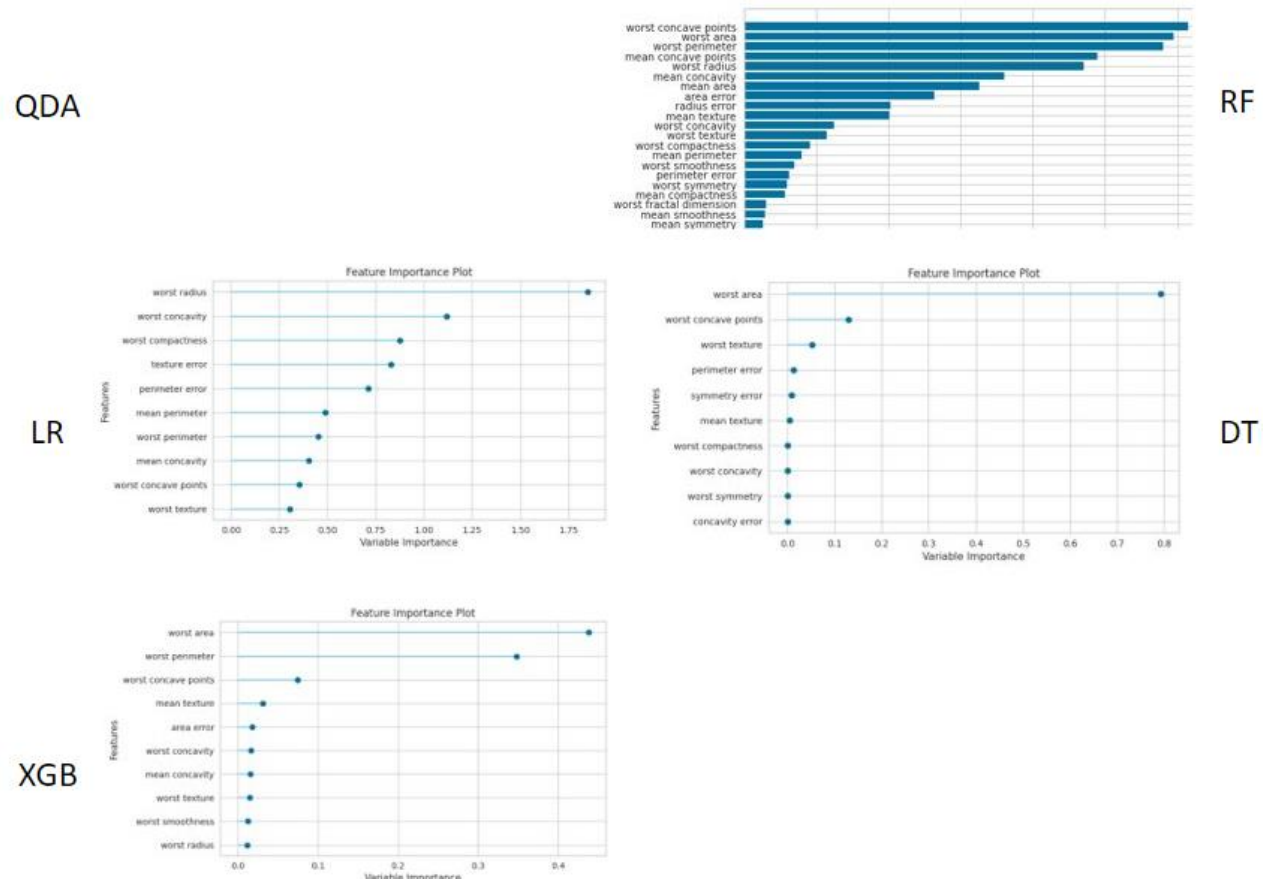


Figure 1.4: Feature importance charts based on each model's attribution

Not unfamiliar to many who have used ML models, the feature importance is an insight into how the model determines which feature carries the heaviest weight to the prediction result.

Model Insights: Partial Dependence Plot

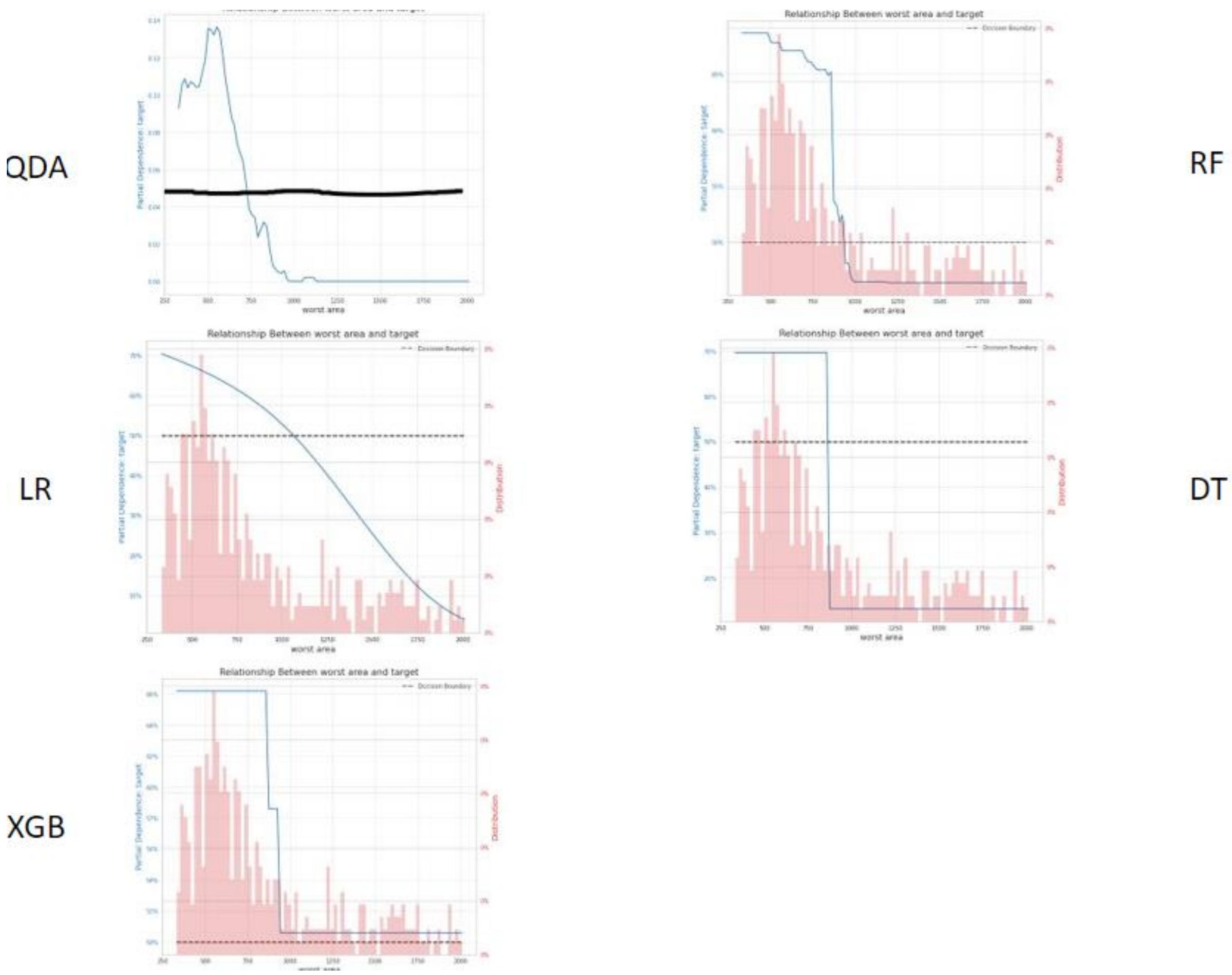


Figure 1.5: Partial Dependence Plot (PDP) of the relationship between the worst area and the target value. Dotted lines show the boundary line when the prediction flips.

As shown in fig 1.5, the PDP plots can be generated for each of the features in the feature space to understand how the changes of the feature value affects the target/classification value.

In fig 1.5, the histogram is the value of the 'worst area' feature, the blue line is the PDP curve, and the dotted line is the decision boundary. Note that for a linear model, the PDP behaves linearly, and for the quadratic model, the PDP behaves in a quadratic way as well. This is reflective of the model.

However, PDP assumes that features are independent of each other, and should there be domain knowledge that does not agree with that assumption, other techniques can be used to supplement the PDP plot. One such plot would be to use the Accumulated Local Effects (ALE) plots.

As a short summary of intrinsically interpretability, the deep dive into each model's intrinsic behaviours is useful for ML practitioners/engineers to generate greater understanding into the models that they used, these insights can help the engineer tweak, calibrate and improve the model.

Next, we can look at more directed XAI techniques so that both engineers and end-users can have a more human-friendly understanding into the data.

<u>MODEL BEHAVIOR THROUGH XAI TECHNIQUES - SHAP</u>

Instead of studying the models intrinsic features, XAI global agnostic techniques can be used to wrap around the whole model as well, one common technique is to use SHAP to generate the SHAPely Values for further analysis.

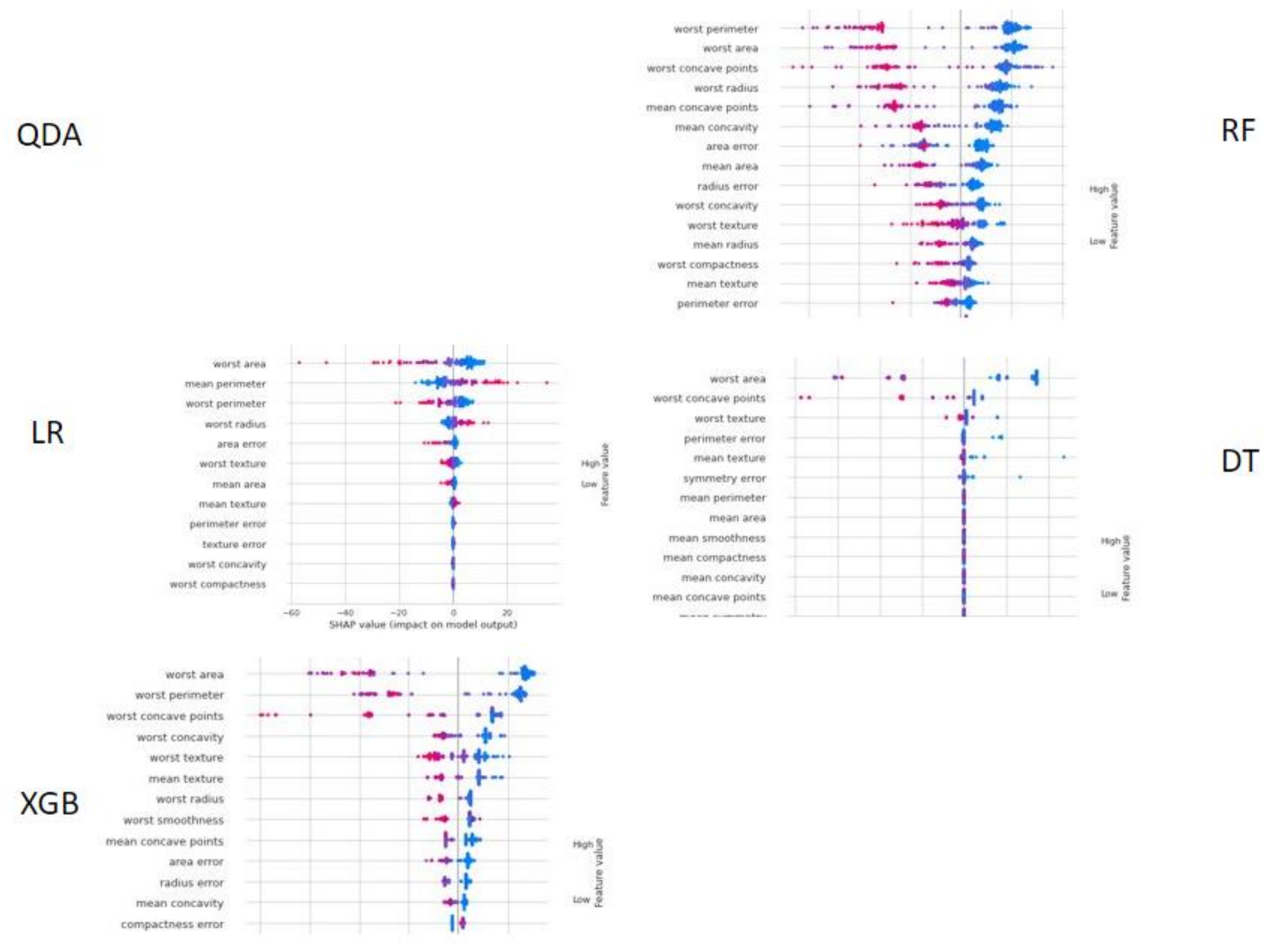


Figure 1.6: SHAP summary plot shows each feature's impact on the classification outcome

To understand the summary provided in fig 1.6, target class 0 represents malignant, while target class 1 represents benign.

Each dot represents each data point and the darker the BLUE dot, the lower the value of that feature and the darker the RED dot, the larger the value accordingly.

Hence, as a generic interpretation of say “worst area”, the lower the value of the “worst area” (BLUE), the outcome leans closer to the right, which is towards class 1 - benign classification, and the higher the value of the “worst area” (RED), the outcome leans closer to the left which is towards class 0 – malignant classification.

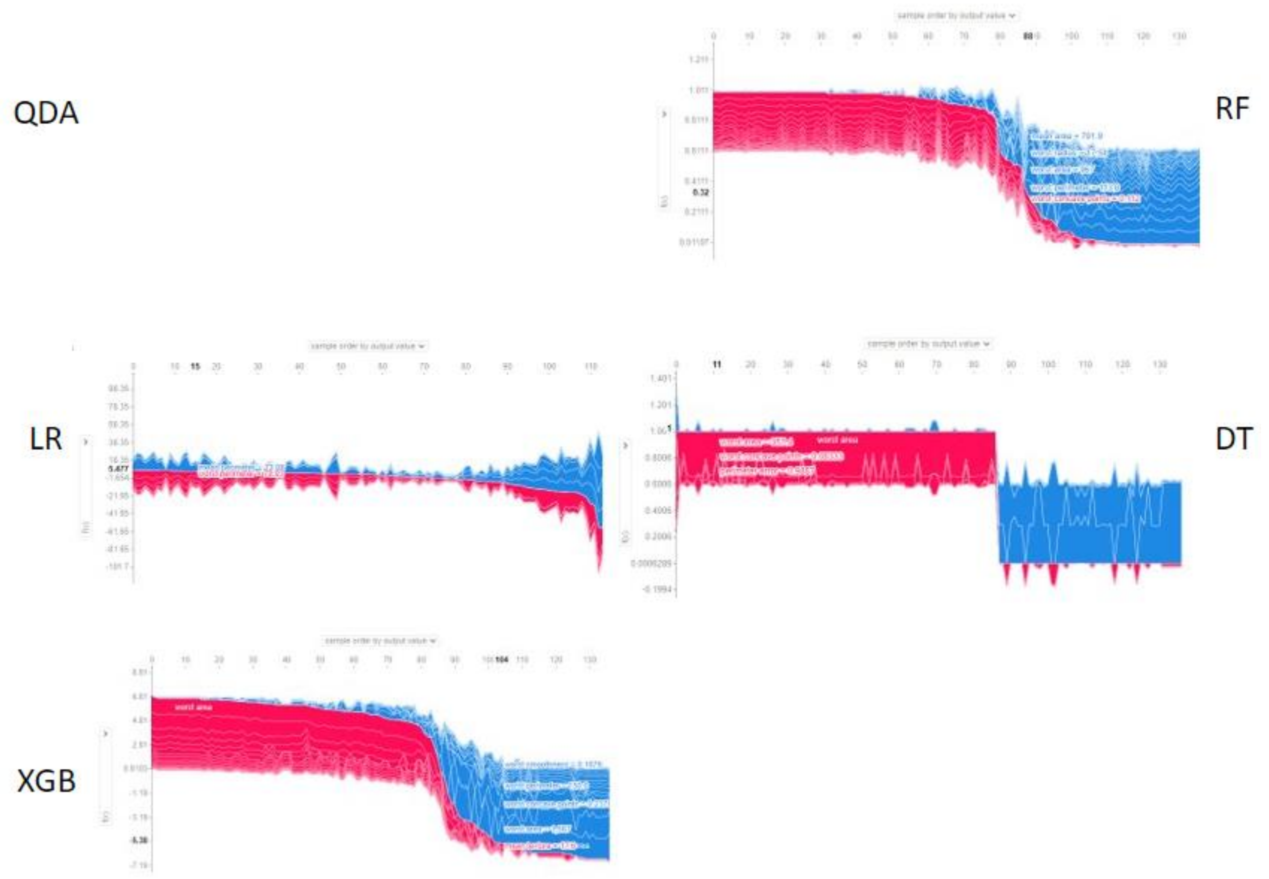


Figure 1.7: SHAP force/reason plot shows each data point and the reason for the associated classification

The reason plot can also be generated as an interactive display to study each data point and to find and study each instance and get insights into the whys of the model’s classification result. This result can then be used in the event where the user (doctor/patient) requires a reason for the model’s prediction.

The analysis can be further developed from just having a reason as to why to an actionable step to answer this question: “What can be adjusted/changed to flip the classification?” and that will be discussed later with another technique in XAI called the counterfactuals.

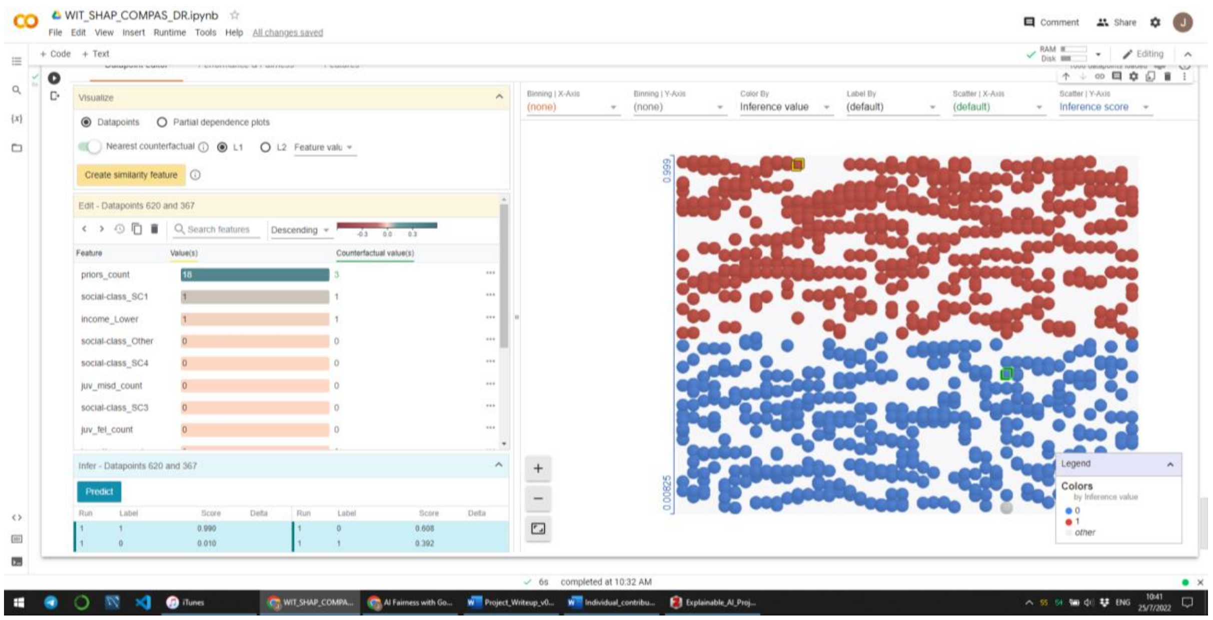
Google What-If Tool for Counterfactual Datapoints

This subsection covers the Javascript based Google’s What If Tool (WIT) and how it can be used to show Counterfactual Datapoints. This can be used as an extension in Jupyter or Google Colaboratory, where a subset of data can be visualised [2].

Counterfactual Datapoint is a method of displaying a datapoint belonging to a different classification label, using a distance function. Counterfactual Explanations is defined by Wachter et. al. as "Score p was returned because variables V had values (v1,v2,...) associated with them. If V instead had values (v1',v2',...), and all other variables had remained constant, score p' would have been returned." [3] The distance function defines the distance between two different datapoints, and the aim is to find the closest datapoint with a different score or classification as the original datapoint. [3]

As discussed by Rothman [4] , he uses an example of how Google What If tool can illustrate AI Fairness. At Chapter 6 of his book, he uses the dataset from Correctional Offender Management Profiling for Alternative Sanctions (COMPAS). COMPAS is a commercial algorithm created by Northpointe Inc. used to assist judges and police officers to predict recidivism. Recidivism is a measurement of a criminal defendant's likelihood to re-offend. The original dataset obtained by ProPublica the same variables used in the COMPAS algorithm [5].

The Google What If Tool can show selected datapoints, and how a trained model classified these datapoints. Once the "Nearest Counterfactual" radio button is enabled and a datapoint is selected, it will attempt to find a closest distance datapoint in a different classification class. Google WIT will then put these datapoints' features side by side, and identify which features, if changed, can change the classification class. Once these feature values are changed, a new prediction can be generated. This method would be a good tool to show how the prediction can change.


WIT_SHAP_COMPAS_DR.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
Visualize
Legend
Colors by Inference value
0
1
other

[● Datapoints] Partial dependence plots

Nearest counterfactual [✓ L1] L2 Feature valu

Binning | X-Axis (none)
Binning | Y-Axis (none)
Color By Inference value
Label By (default)
Scatter | X-Axis (default)
Scatter | Y-Axis Inference score

Edit - Datapoints 620 and 367

Feature	Value(s)	Counterfactual value(s)
priors_count	18	3
social-class_SC1	1	1
income_Lower	1	1
social-class_Other	0	0
social-class_SC4	0	0
juv_misd_count	0	0
social-class_SC3	0	0
juv_fel_count	0	0

Infer - Datapoints 620 and 367

Run	Label	Score	Delta	Run	Label	Score	Delta
1	1	0.990		1	0	0.608	
1	0	0.010		1	1	0.392	

6s completed at 10:32 AM

Figure 1.8: Google What If Tool Example for showing Counterfactual points

Using Google WIT, it is a good counterfactual "what-if" tool to see how recidivism can change when changing the feature values. For supervised machine learning problems, it can also display the label column and check whether the label is intuitive. However, these are some features that are not easily changeable at the point of time. Examples of these are features relating to a person's age, gender or ethnicity.

InterpretML

As mentioned earlier that some ML models are intrinsically interpretable and some more complex models along with deep learning models can only be interpreted with black-box approach. This also means that less complex models although they are more interpretable have lesser accuracy and more complex models, although having higher accuracy is less interpretable, which means there is a trade-off in accuracy vs interpretability[6]

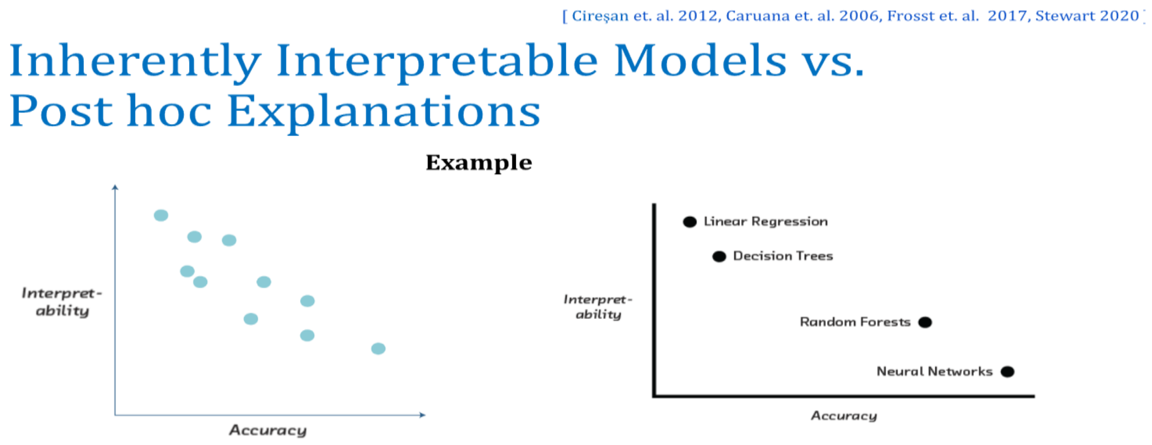


Figure 1.9.1: Interpretability vs Accuracy tradeoff [6]

One of the solution is using an open source package, InterpretML, developed at Microsoft Research that comes with popular XAI techniques such as LIME and SHAP but also their own developed model, Explainable Boosting Machine, which can explain in both global,local and have higher accuracy while maintaining interpretability.[7]

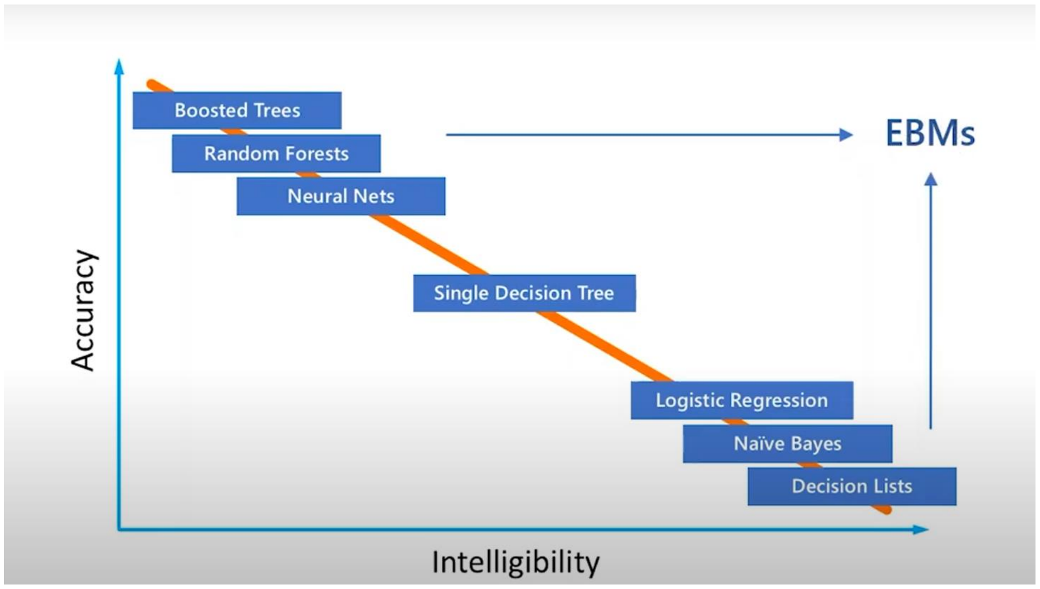
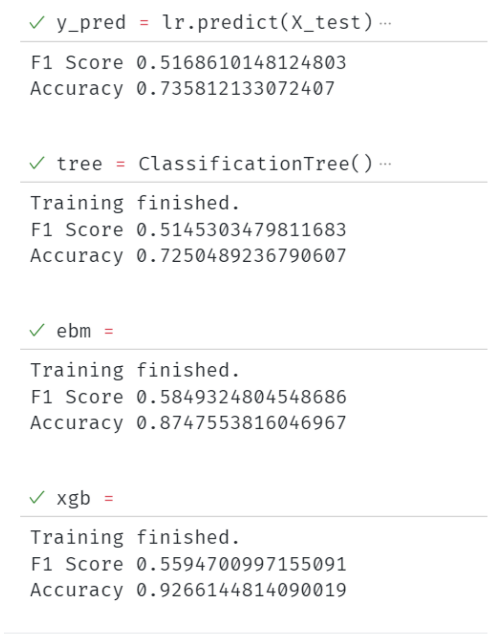


Figure 1.9.2: EBM`s performance[7]



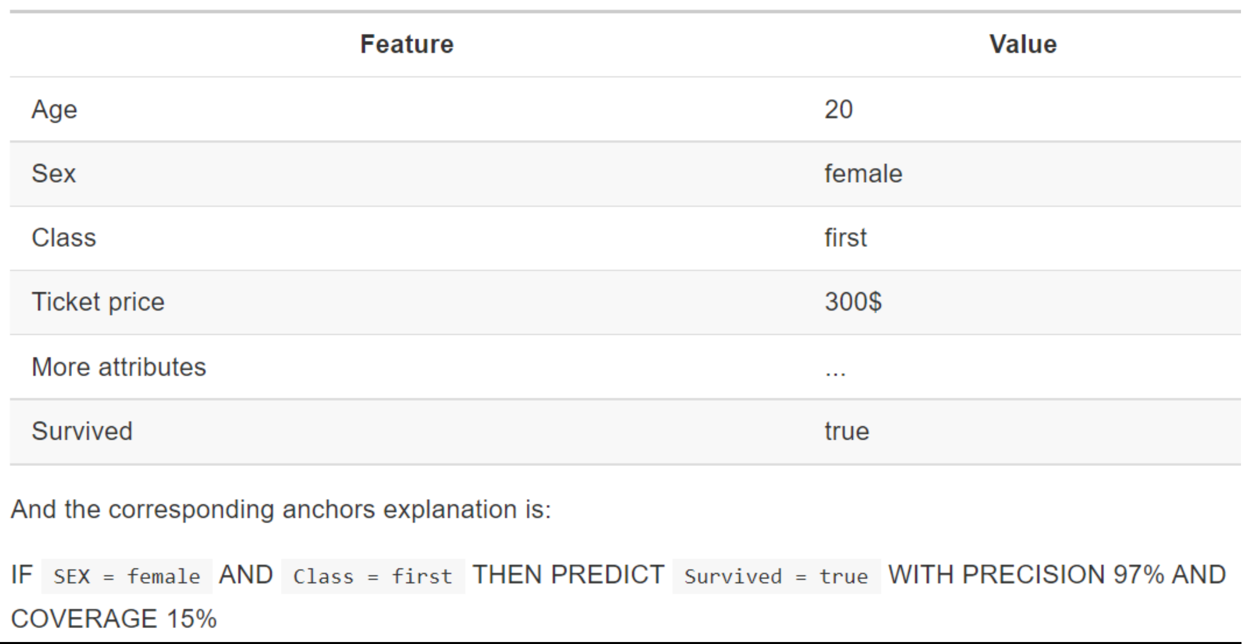
```
✓ y_pred = lr.predict(X_test) ···

F1 Score 0.5168610148124803
Accuracy 0.735812133072407

✓ tree = ClassificationTree() ···

Training finished.
F1 Score 0.5145303479811683
Accuracy 0.7250489236790607

✓ ebm =

Training finished.
F1 Score 0.5849324804548686
Accuracy 0.8747553816046967

✓ xgb =

Training finished.
F1 Score 0.5594700997155091
Accuracy 0.9266144814090019
```

Figure 1.9.3 EBM`s performance vs other ML models[8]

Scoped rules(Anchor)

This subsection covers the XAI method anchors which explains individual predictions of any black box model by finding a decision rule that causes the prediction. The resulting explanations are expressed easy-to-understand IF-THEN rules, called anchors.[1]

In an example in Interpretable Machine Learning by Christoph Molnar shows how anchor explanation works.



Feature	Value
Age	20
Sex	female
Class	first
Ticket price	300$
More attributes	...
Survived	true

And the corresponding anchors explanation is:

IF `SEX = female` AND `Class = first` THEN PREDICT `Survived = true` WITH PRECISION 97% AND COVERAGE 15%

Figure 1.9.4 Anchor example[1]

The explanation shows what features and what reasoning is used to cause that model`s prediction which can be used to check if the model`s prediction is using the wrong features or features that cause bias.

Now that we have explored the several XAI methods this is our recommended approach in implementing XAI for tabular data.

Our recommend approach is to use a intrinsically interpretable model if the accuracy does not meet the required target then proceed on to black box models as interpretability gives the machine learning engineers the information required to fine tune or troubleshoot the model.

We also recommend using different XAI methods to display different type of information and function for different stakeholders.

For example, SHAP is an extremely informative method but it is not useful to auditors or domain experts as they are concerned with if the model`s prediction is robust and biased, which counterfactuals and anchors can display more appropriately compared to SHAP.

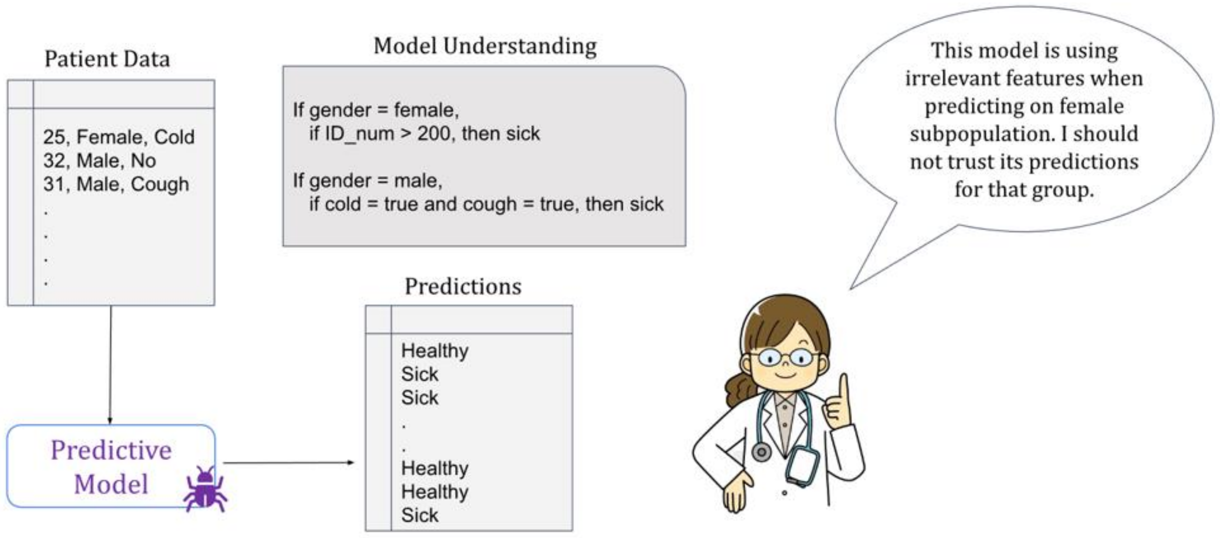

Patient Data
25, Female, Cold
32, Male, No
31, Male, Cough
Model Understanding
If gender = female,
if ID_num > 200, then sick
If gender = male,
if cold = true and cough = true, then sick
Predictions
Healthy
Sick
Sick
Healthy
Healthy
Sick
Predictive Model
This model is using irrelevant features when predicting on female subpopulation. I should not trust its predictions for that group.

Figure 1.9.5 example of information for domain expert [6]

In the case for consumers applying for a loan, the information from anchor or counterfactuals can be used to give a recourse on how the applicant can improve their chances of getting their loan approved

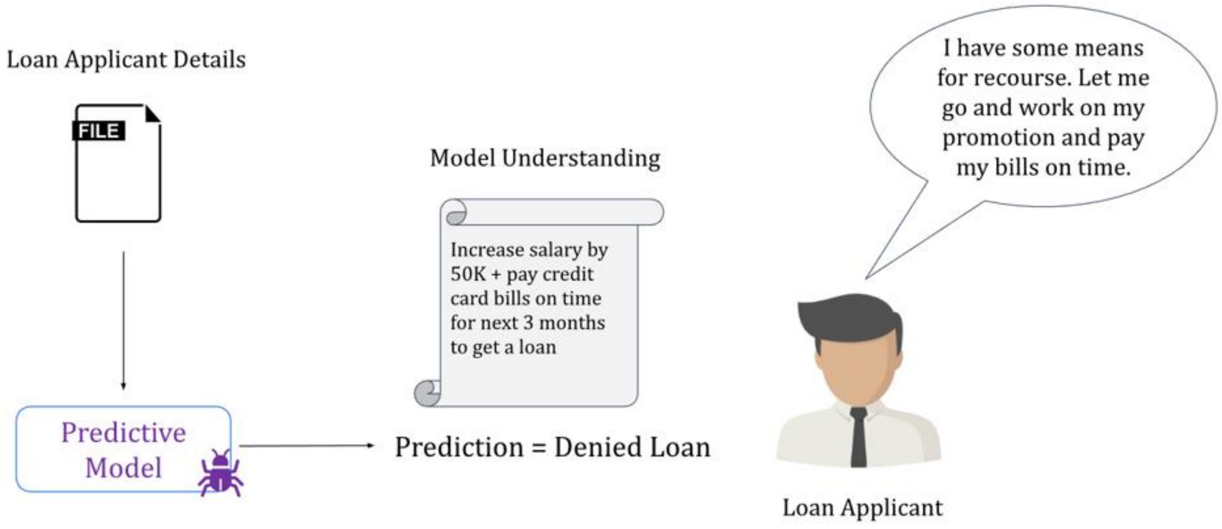

Loan Applicant Details
FILE
Model Understanding
Increase salary by 50K + pay credit card bills on time for next 3 months to get a loan
Predictive Model
Prediction = Denied Loan
I have some means for recourse. Let me go and work on my promotion and pay my bills on time.
Loan Applicant

Figure 1.9.6 example of information for consumers[6]

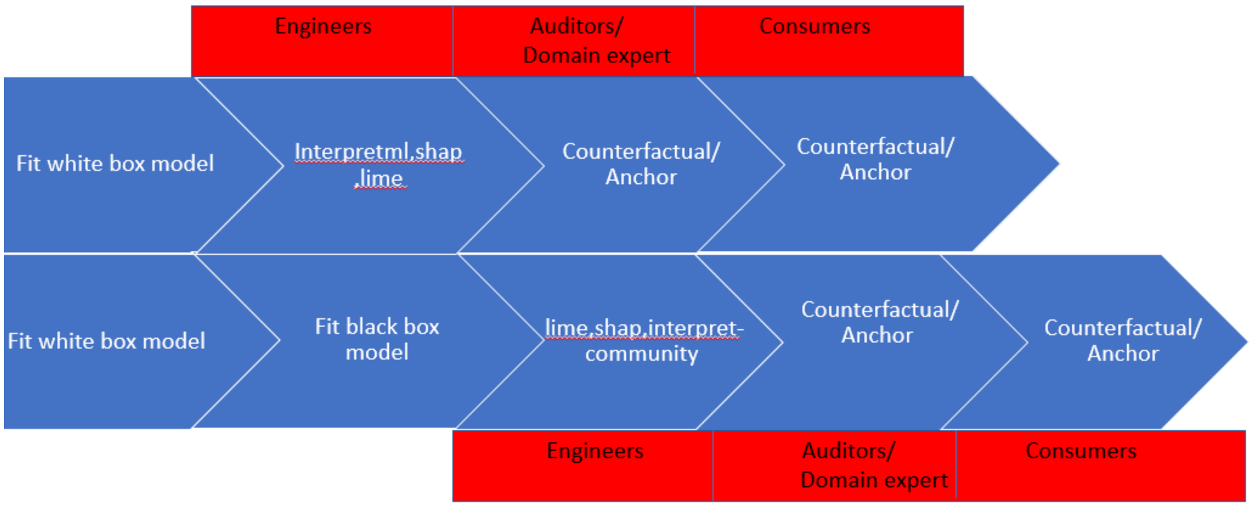


Figure 1.9.7 Recommend Approach

IMAGE DATA

This section will cover a few of the methods used to create interpretations for image data. There are mainly 2 popular types of XAI methods for computer vision. The first one is Layer-wise Relevance Propagation (LRP) and the second is Grad-cam. We`ve explored 3 different XAI package and find that Grad-cam gives the better explanation. Let`s briefly show the 2 XAI package we`ve explored before diving into Grad-cam package.

LRP toolbox

This package uses Layer-wise Relevance Propagation (LRP) as the XAI method. It explains the model`s prediction by highlighting what the neural network thinks are the important pixels.

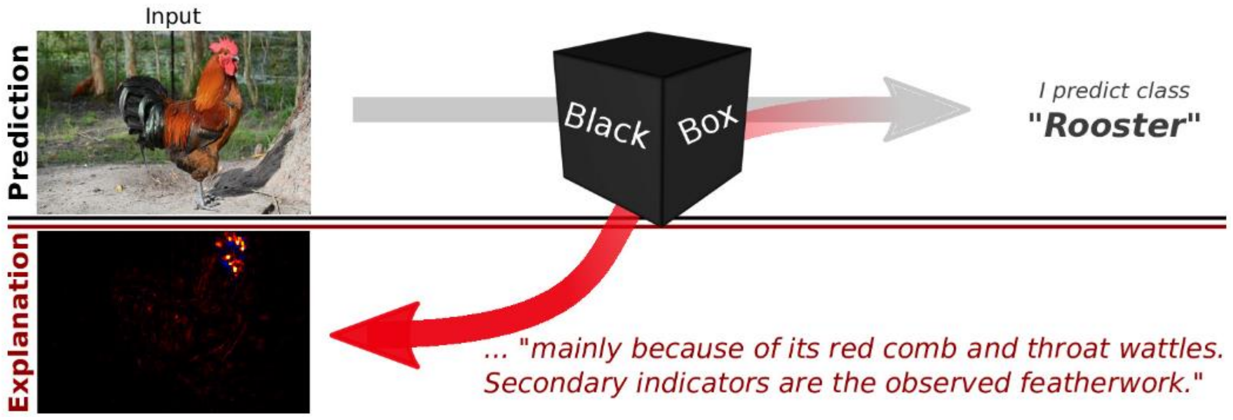

Input
Prediction
Explanation
Black Box
I predict class
"Rooster"
... "mainly because of its red comb and throat wattles.
Secondary indicators are the observed featherwork."

Figure 2.1 LRP toolbox example [9]

Do note that this package is for only limited to scikit-learn and there is 2 variant of this package which is investigate(Keras) and zennit(Pytorch) however it is hard to code and use the package and the documentation is not beginner-friendly.

Captum

The second package we`ve explored is Captum. It is an open-sourced package developed by Meta it has gradient-based methods including Grad-cam and LRP and also perturbation based methods such as LIME and kernel SHAP however the visualization against Grad-cam still comes up short. You will see the difference between the 2 packages when you compared the images shown below.

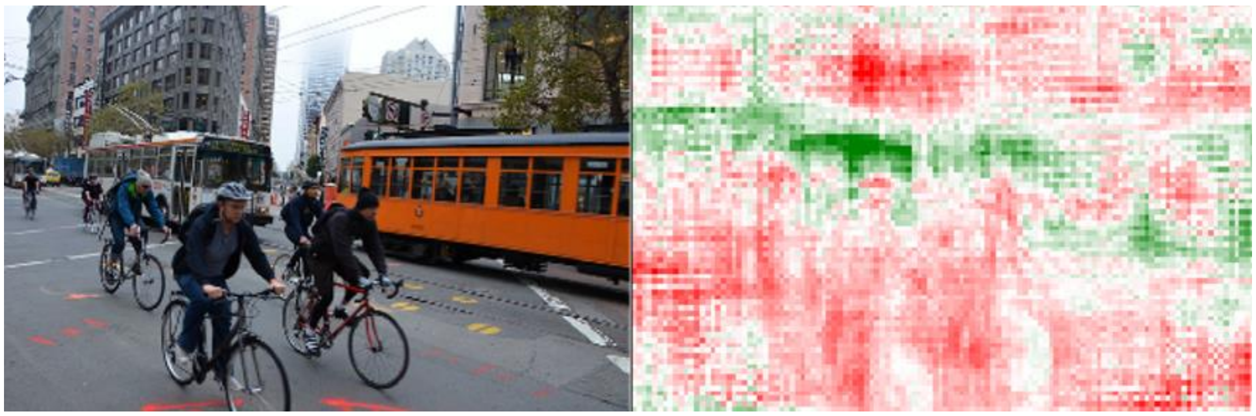


Figure 2.2 Semantic Segmentation with Gradcam on Captum[10]

<u>Grad-cam</u>

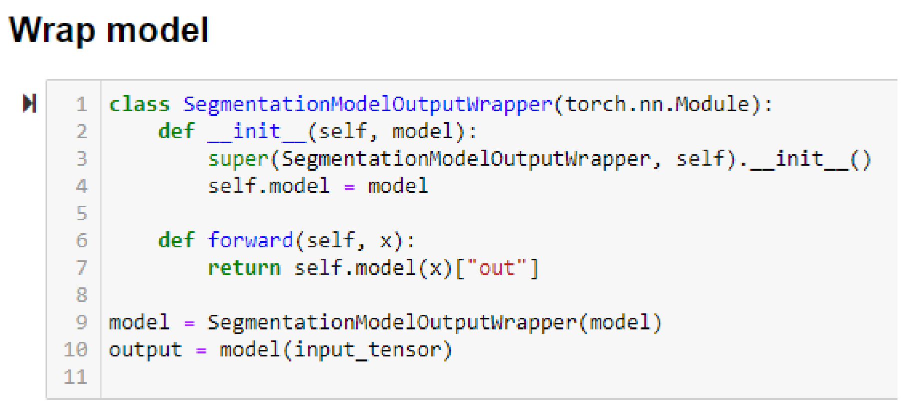
This package comes with many class activation maps methods, it is also easy to use and the visual explanation is our opinion the best out of the 3. We will go through one of the example of this package and see its visualization and ease of use. The demo notebook of this package has been broken down into simpler steps so the user can play around with the parameters.

Example: Semantic Segmentation

1. To apply this XAI method you have to code a wrapper to wrap the pytorch model`s output

2. Define the target class for semantic segmentation

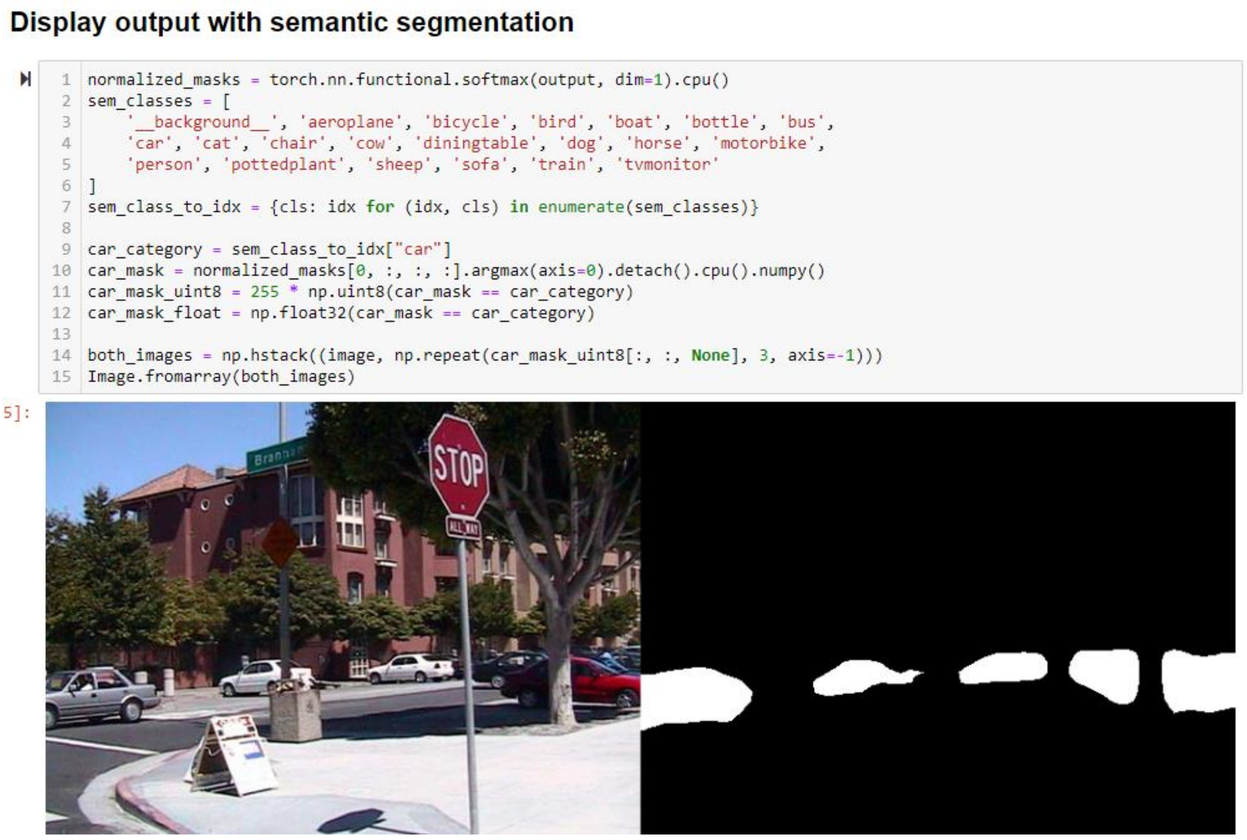
3.Apply one of the class activation maps methods



Wrap model

```python
class SegmentationModelOutputWrapper(torch.nn.Module):
    def __init__(self, model):
        super(SegmentationModelOutputWrapper, self).__init__()
        self.model = model

    def forward(self, x):
        return self.model(x)["out"]

model = SegmentationModelOutputWrapper(model)
output = model(input_tensor)

```

Figure 2.3 Wrap pytorch model output code



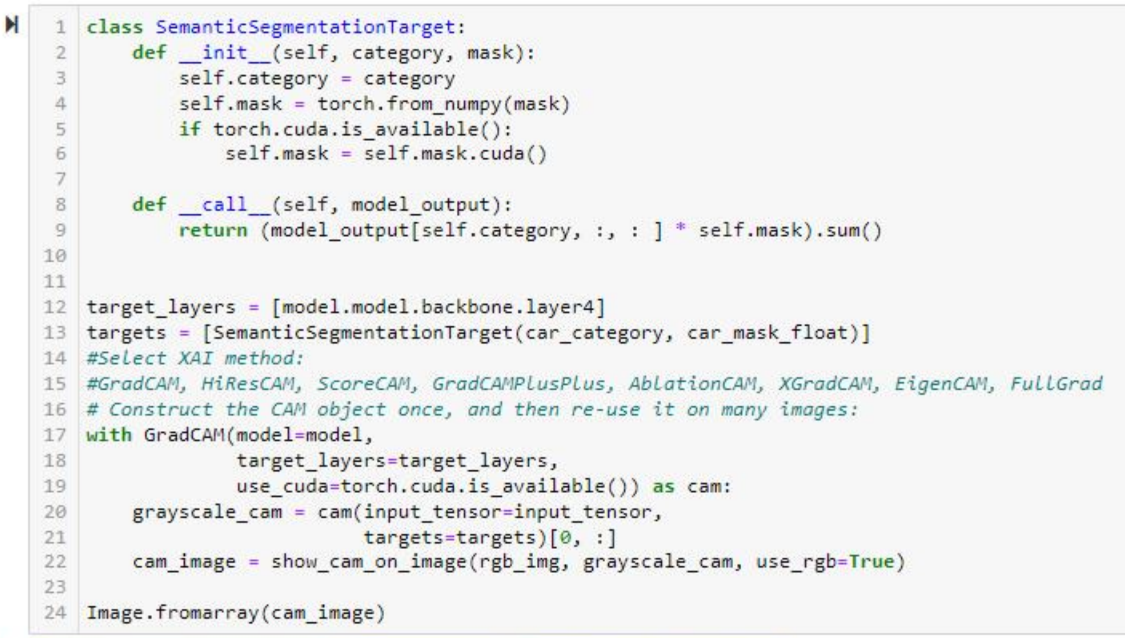
Display output with semantic segmentation

```python
normalized_masks = torch.nn.functional.softmax(output, dim=1).cpu()
sem_classes = [
    '__background__', 'aeroplane', 'bicycle', 'bird', 'boat', 'bottle', 'bus',
    'car', 'cat', 'chair', 'cow', 'diningtable', 'dog', 'horse', 'motorbike',
    'person', 'pottedplant', 'sheep', 'sofa', 'train', 'tvmonitor'
]
sem_class_to_idx = {cls: idx for (idx, cls) in enumerate(sem_classes)}

car_category = sem_class_to_idx["car"]
car_mask = normalized_masks[0, :, :, :].argmax(axis=0).detach().cpu().numpy()
car_mask_uint8 = 255 * np.uint8(car_mask == car_category)
car_mask_float = np.float32(car_mask == car_category)

both_images = np.hstack((image, np.repeat(car_mask_uint8[:, :, None], 3, axis=-1)))
Image.fromarray(both_images)
```

5]:

Figure 2.4 Define class and display output

Select target image to apply XAI



```python
class SemanticSegmentationTarget:
    def __init__(self, category, mask):
        self.category = category
        self.mask = torch.from_numpy(mask)
        if torch.cuda.is_available():
            self.mask = self.mask.cuda()

    def __call__(self, model_output):
        return (model_output[self.category, :, : ] * self.mask).sum()


target_layers = [model.model.backbone.layer4]
targets = [SemanticSegmentationTarget(car_category, car_mask_float)]
#Select XAI method:
#GradCAM, HiResCAM, ScoreCAM, GradCAMPlusPlus, AblationCAM, XGradCAM, EigenCAM, FullGrad
# Construct the CAM object once, and then re-use it on many images:
with GradCAM(model=model,
             target_layers=target_layers,
             use_cuda=torch.cuda.is_available()) as cam:
    grayscale_cam = cam(input_tensor=input_tensor,
                        targets=targets)[0, :]
    cam_image = show_cam_on_image(rgb_img, grayscale_cam, use_rgb=True)

Image.fromarray(cam_image)
```

16]:

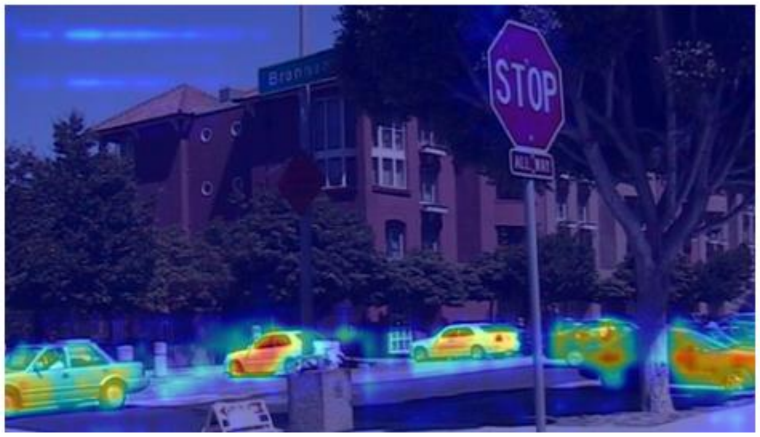


Figure 2.5 Applying XAI method

As you can see its easy to apply the XAI with this package but with so many types of methods, which to use? Unfortunately we cant give you the solution to this answer but the only way is to implement several and see which gives the best explanation.

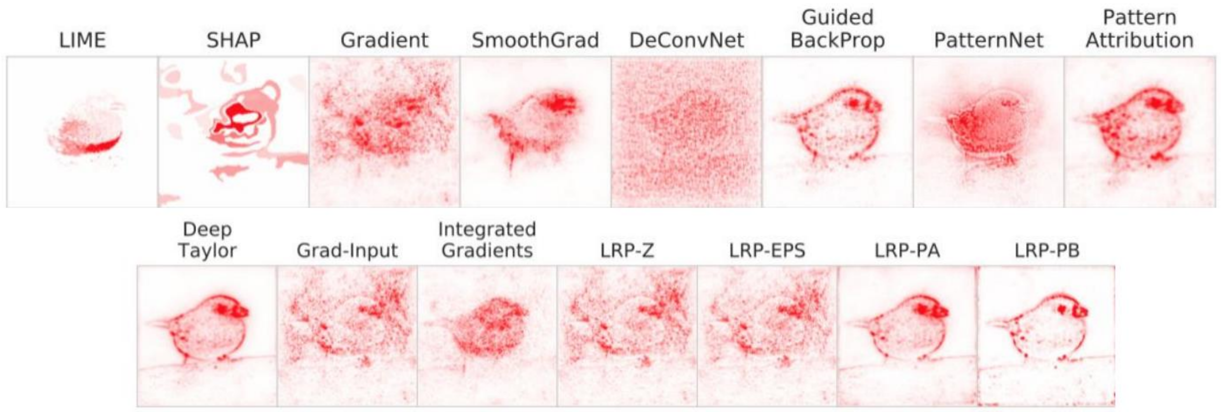


Figure 2.6 Different XAI method example 1 [6]

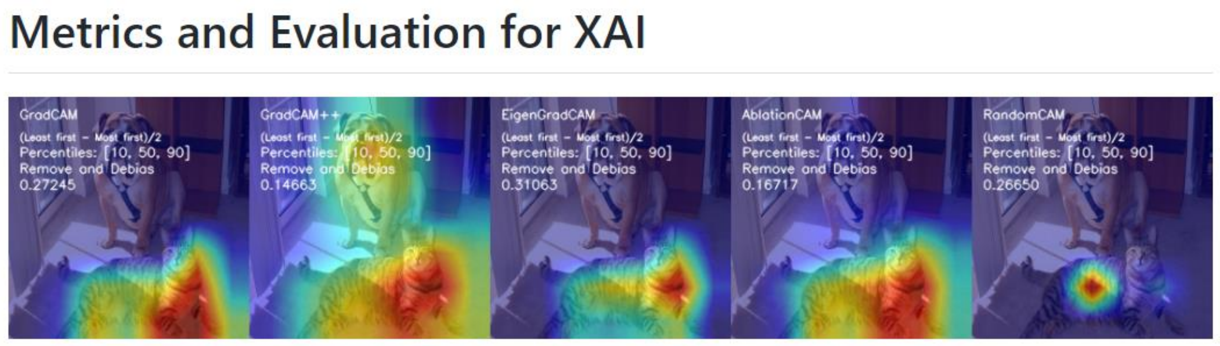

Metrics and Evaluation for XAI
GradCAM
(Least first - Most first)/2
Percentiles: [10, 50, 90]
Remove and Debias
0.27245
GradCAM++
(Least first - Most first)/2
Percentiles: [10, 50, 90]
Remove and Debias
0.14663
EigenGradCAM
(Least first - Most first)/2
Percentiles: [10, 50, 90]
Remove and Debias
0.31063
AblationCAM
(Least first - Most first)/2
Percentiles: [10, 50, 90]
Remove and Debias
0.16717
RandomCAM
(Least first - Most first)/2
Percentiles: [10, 50, 90]
Remove and Debias
0.26650

Figure 2.7 Different XAI method example 2[11]

The metric seen in Fig 2.6 is part of the grad-cam package which uses different metric to help with parameter tuning, which XAI method to use and the performance of each XAI method[11]

Discussions: Image XAI Methods

As mentioned earlier, there is no solution to pick the best visualization, the only way is to code it out and see which is best, we've picked the grad-cam package due to its better visualization, ease of use and documentation. This is not to say that the other 2 packages is not worth exploring.

Even though it was shown earlier using the same method Captum displays Grad-cam method in a different way, Captum does have a lot more libraries and methods inside which Grad-cam does not have.

For LRP toolbox, it is also worth exploring as the above 2 package works based off pytorch and LRP toolbox is able to work on Keras(iNNvestigate) however if you do want to explore LRP on pytorch you can use its pytorch variant package, Zennit, as it may still show different insights and visualization that you need.

TIME SERIES DATA

This section will consist of the available methodologies using Time Series Data. It will focus on solving multivariate time series problems, or 2 to more features used. This section comprises of Vector Autoregression Model (VAR) that is a statistical methodology for time series modelling, LIONETS for interpreting Encoder-Decoder Neural network architecture sub-type, and Neural Prophet, a bridge between Statistical and Neural Network method for time series problems.

VECTOR AUTOREGRESSION MODEL (VAR)

Vector Autoregression (VAR) model is a Traditional Statistics Time Series for bivariate/multivariate data, or 2 or more features. This method was popularized by Christopher Sims in Economics using multivariate macroeconomic data [12]. The model will look at current Variable A's values with its lagged values, and Variable B's lagged values to form an equation to forecast Variable A's future values.

The case study featured is the monthly Google Search Trend data for Heater and Ice Cream from years 2004 to 2011, and the desired outcome is to use both trend data to forecast out-of-data search trend values of Heater.

Before fitting the model, one important assumption for VAR model to work is the concept of stationary data. Stationary Data is having the datapoint values change in a constant way, or stationary data have a constant variance. Figure 3.1 below illustrates a non-stationary time series value, while Figure 3.2 will illustrate how the values look like once the time series is stationary.

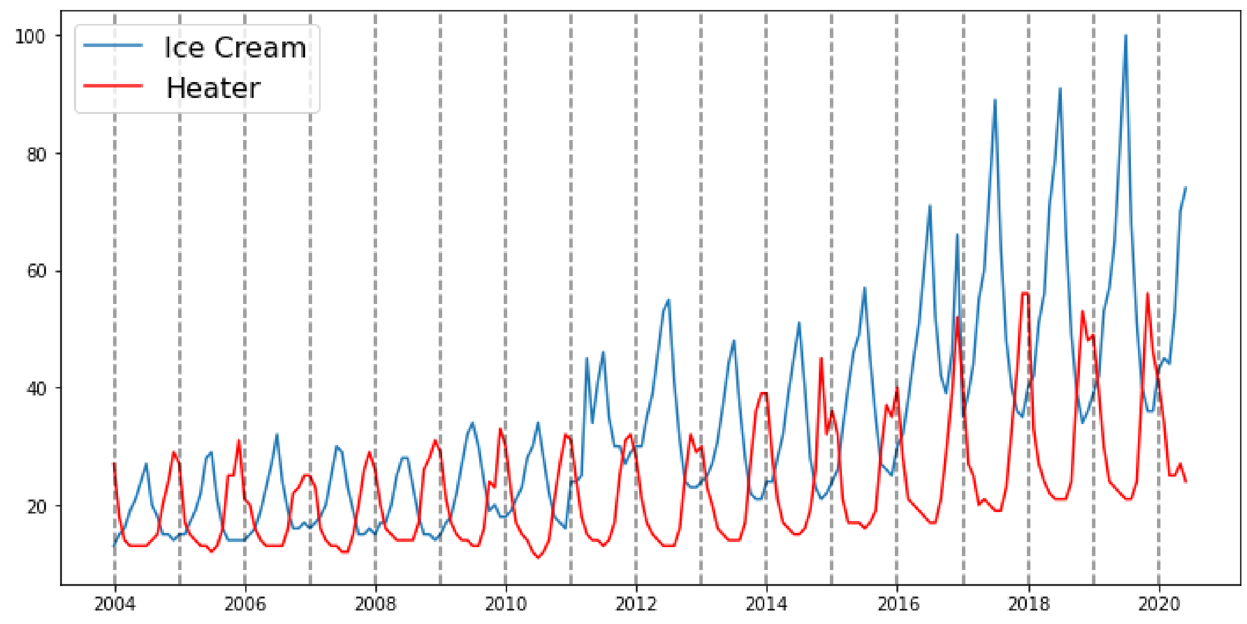


Figure 3.1: Example of Non-stationary Time-Series before pre-processing

Should the time series not be stationary, pre-processing steps such as eliminating the effects of trend, seasonality, and volatility. In this case, both Heater and Ice Cream time series will need to be stationary, before fitting these values. Figure 3.2 below showcases the plots for Heater and Ice Cream when it is stationary.

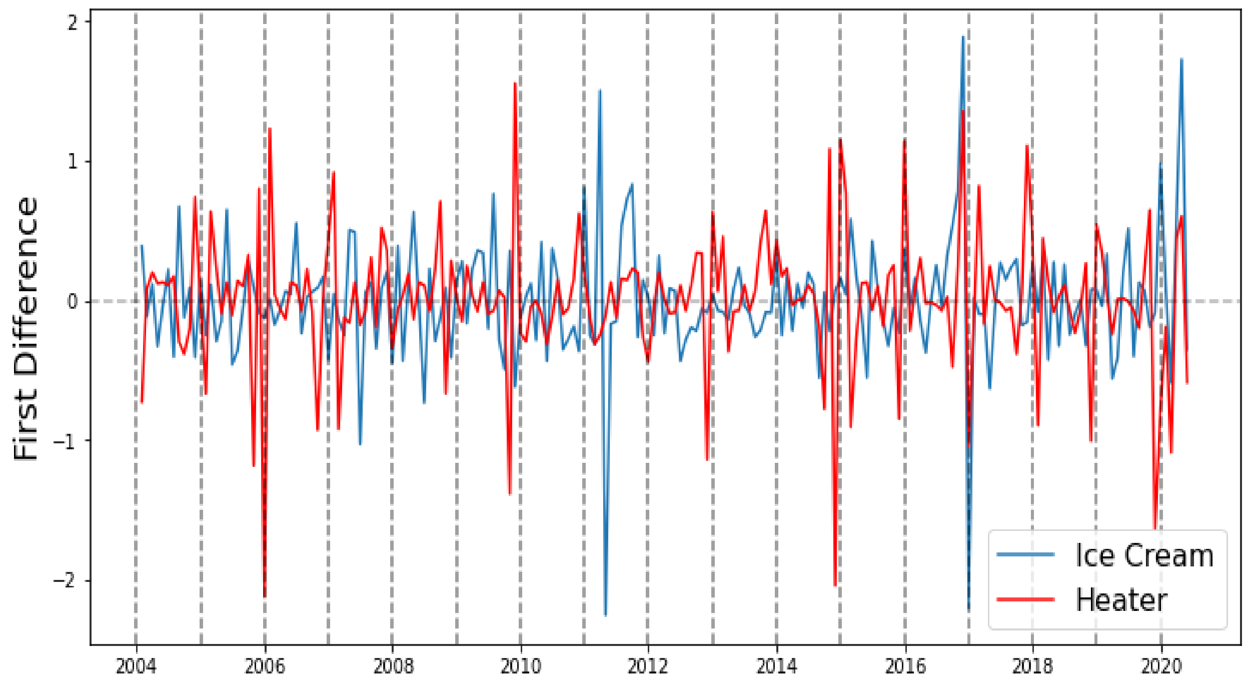


Figure 3.2: Example of a Stationary Time-Series after pre-processing

After fitting the VAR model, we can extract the coefficients from the lagged values of both Heater and Ice Cream by setting a p-value threshold of below 0.05. The coefficients can then be used to construct an equation to forecast future Heater search trends. Since the data is a bivariate monthly time series, the lagged values represent one month lagged values, up to the maximum available data.

Variable and Lag Value	Coefficient	p-value
Lag 1 Heater	-0.41	0.000
Lag 2 Heater	-0.19	0.022
Lag 13 Ice Cream	0.20	0.027

Table 3.1: Relevant Coefficients below the p-value threshold.

The lags above represent 1 month before, 2 months before and 13 months before respectively. After the model fitting, the final equation used to predict Heater can be found below:

$$\hat{h}_t = -0.41h_{t-1} - 0.19h_{t-2} + 0.2i_{t-13}$$

After exploring VAR model, it is debatable if this can be considered an Explainable AI methodology, or even be grouped as an interpretable model together with linear regression. This is because the VAR model has specific assumptions such as “stationary data” to be done before fitting it to the VAR model. Also, some metrics to measure model performance can be model specific and metrics are not usable across models, as compared to a typical Neural Network’s accuracy score, confusion metrics receiver operating characteristic (ROC) curve.

Overall, Vector Autoregression is an appropriate model to be used for multivariate data, despite the challenges that come with it. This is a good method for models that only work with univariate or one feature type of data, however the model being interpretable could be debatable.

LOCAL INTERPRETATION OF NEURAL NETWORKS THROUGH PENULTIMATE LAYER DECODING (LIONETS)

LioNets is a local surrogate XAI method that uses the penultimate layer before the output layer, to generate a local neighbourhood closely comparable to the original penultimate layer. This is to understand how a particular prediction can be made. After combining this information, a linear interpretable model is trained and its explanations can be extracted [13].

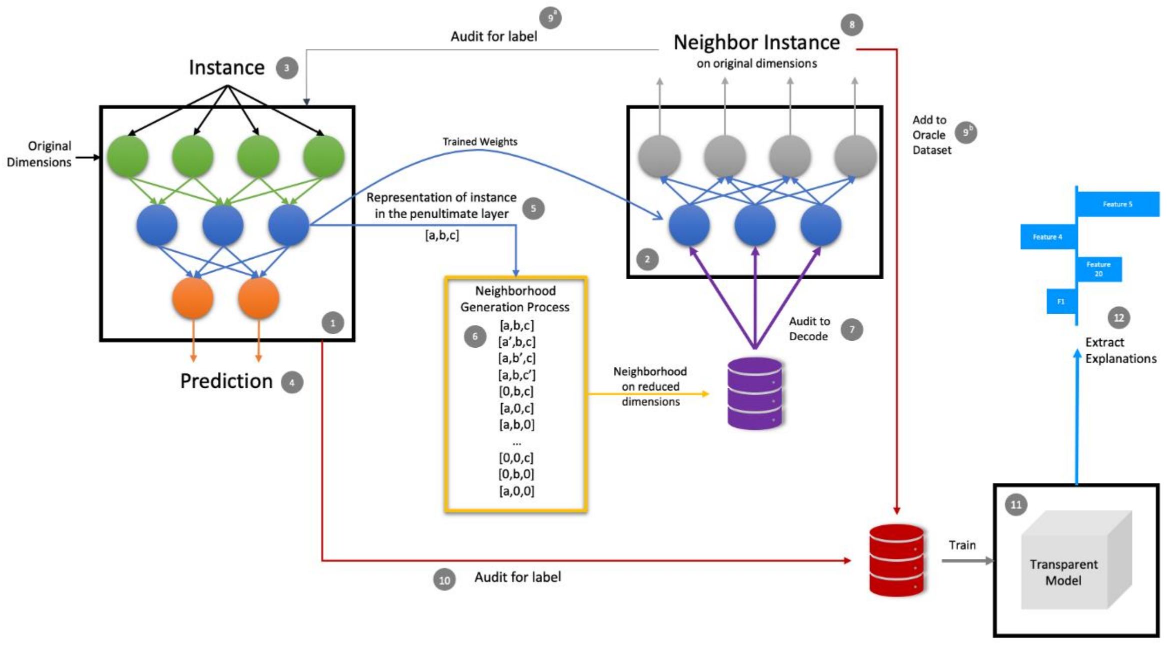


Figure 3.3: Summary of LioNETS XAI Methodology for Time Series Data Type

It is a model specific approach that requires a certain neural network model architecture to be able to use LioNets. The components needed to create a LioNets are model architecture(s) with an Encoder-Decoder architecture. The Encoder can be extracted from the predictor model, while the Decoder will need to be trained separately. For this model specific example, a Long Short Term Memory (LSTM) neural network model architecture was used.

The reason this XAI methodology requires a decoder is due to the abstract representation of the linear model post-training that affects prediction, and it needs to be converted back to its original representation. The local neighbourhood representation will have abstract representation. Thus, in order to get the predictions, the decoder will need to be trained to inverse transform from the abstract representation to the original representation [13]. This process can be affected by the data, the original neural network architecture and the encoder.

For the XAI method's case study, a Predictive Maintenance dataset was used from the NASA Turbofan Degradation. This dataset shows the Remaining Useful Life for a particular Turbofan engine, given numerous settings and multiple sensor readings [14]. The Remaining Useful Life for this NASA Turbofan dataset can be expanded to changing operational settings to prolong the life of the engine, or planning future maintenance to facilitate logistics arrangement from fixing a faulty component to fully operational engine [14].

Using the NASA Turbofan dataset and LioNets, a visualisation could be extracted on how a particular sensor influences the RUL. Looking at a particular sensor called "Sensor_11", it can be seen as negatively affecting the RUL. Using the example by Mollas et. al. [13], changing the values of "Sensor_11" improved the RUL from 31 to 49, extending the RUL. See Figure 3.4 and Figure 3.5 below for the visualisation before and after the "Sensor_11" values were changed.

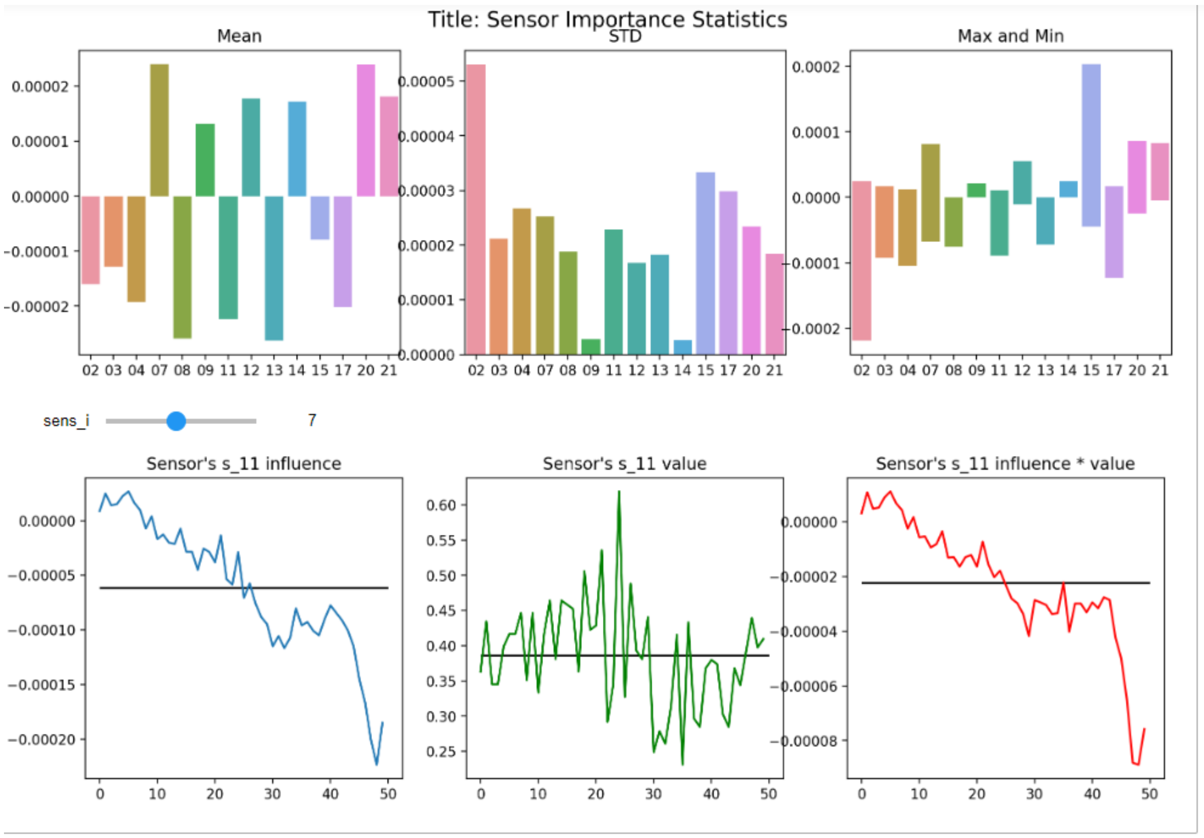


Figure 3.4: Original "Sensor_11" values having negative influence to the RUL

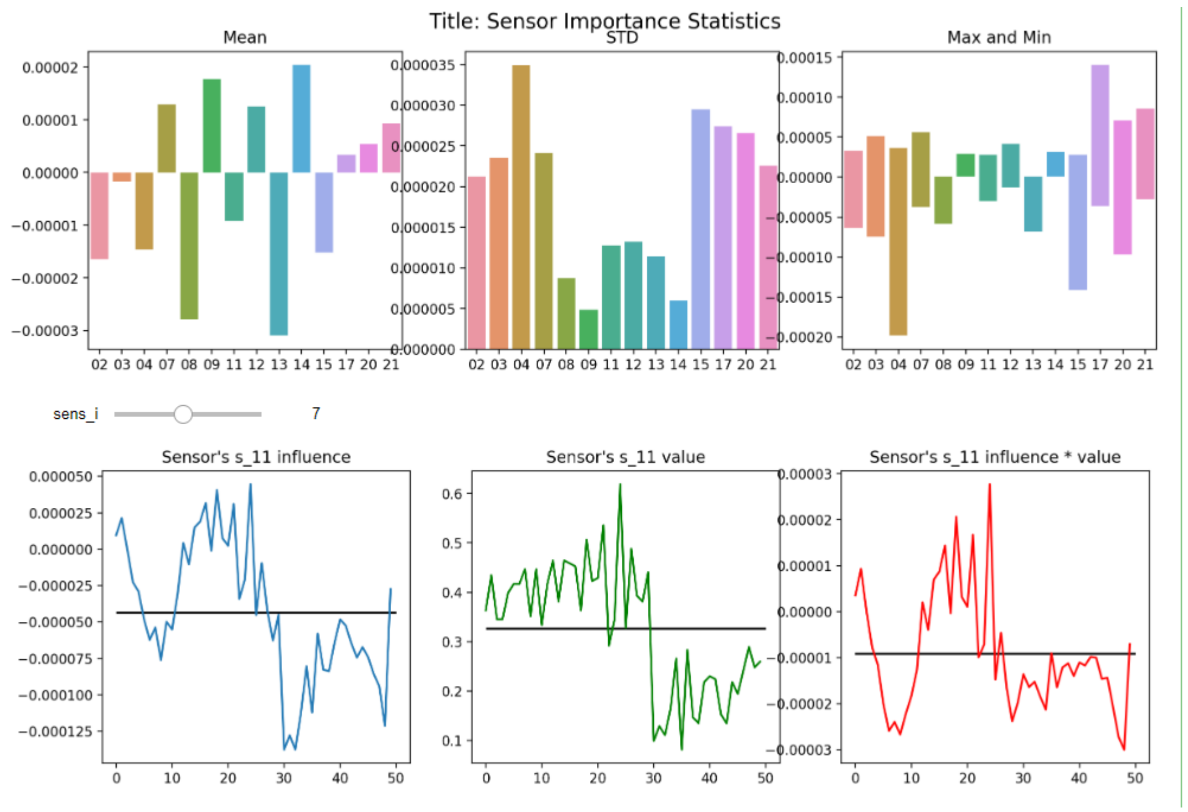


Figure 3.5: Changing “Sensor_11” values Improved Influence on the RUL

Looking at this RUL example, we can better explain which sensor can positively or negatively affect the RUL of a particular turbofan. This allows the LSTM model to be explainable to different stakeholders, and allow its RUL predictions to plan an appropriate downtime for predictive maintenance.

On the other hand, the usage of LioNets as an XAI tool is limited by its model specific archictecture and the challenges of training a decoder. As mentioned by Mollas et. al., LioNets require a specific model architecture that has an Encoder, which limits on the neural network architecture this can be used on. Also, training a decoder can be computationally expensive, and subject to other parameters such as the data, the original predictor and the encoder [13].

Meta (Facebook Research) Neural Prophet

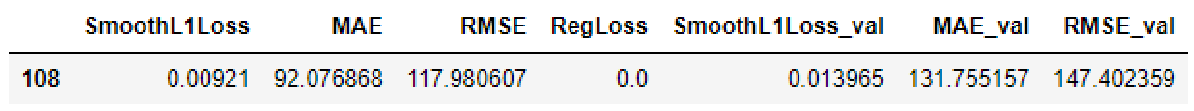
Neural Prophet aims to connect traditional statistical time series and deep learning neural network architecture in one Python Library. It is built using PyTorch Library, which allows a gradient descent optimisation method of model training [15].

At its building block, it utilises Facebook Prophet, a statistical time series tool. This can be seen in the multiple types of hyperparameters that runs along both traditional statistical and deep learning architecture for time series forecasting.

Another component of Neural Prophet is the Auto-Regressive (AR) Neural Network Architecture, a linear neural network architecture. The simplicity of this neural network architecture contributes to it being an interpretable model, while allowing scalability due to the usage of a forward feed neural network [16].

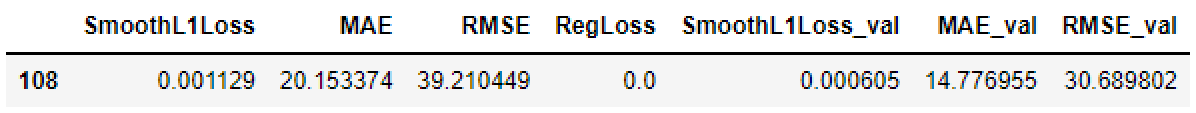
The examples used to showcase Neural Prophet's capability is looking at Solar Irradiance data at the city of San Francisco, a city at California State, United States of America for the year 2015. It is an univariate time series data, with hourly measurements of solar irradiance [17].

Looking at the Neural Prophet case study, the baseline traditional statistical model was able to decompose the hourly solar irradiance dataset to identify the trend, and yearly and hourly seasonality. It also has an extensive list of hyperparameters bridging both traditional statistical methods and neural network specific methods. Comparing between baseline statistics methods and neural network specific methods, Figure 3.6 showcases model evaluation metrics using baseline Statistical model, while Figure 3.7 below showcases the improvement using the neural network method. For the below values generally, the lower the values, the better the model performance is.



	SmoothL1Loss	MAE	RMSE	RegLoss	SmoothL1Loss_val	MAE_val	RMSE_val
108	0.00921	92.076868	117.980607	0.0	0.013965	131.755157	147.402359

Figure 3.6: Model Evaluation Metrics using baseline Statistical Model, taken from [18]



	SmoothL1Loss	MAE	RMSE	RegLoss	SmoothL1Loss_val	MAE_val	RMSE_val
108	0.001129	20.153374	39.210449	0.0	0.000605	14.776955	30.689802

Figure 3.7: Model Evaluation Metrics using Neural Network Architecture, taken from [18]

As of this writing, Neural Prophet is only capable of Univariate Time Series Forecasting, which severely limits the possible use cases. One type of problem that difficult to solve using this library is a Predictive Manufacturing use case, consisting of multiple sensors to predict the Remaining Useful Life (RUL) of a machine. RUL is a measure of which time-step a given machine or component could fail, and preventive maintenance could be decided before machine or component failure. Although Feature Engineering could reduce multiple sensors into one univariate feature, each sensor's influence to the RUL would not be captured fully to a single feature.

Discussions: Time Series XAI Methods

To summarise, there have been challenges in sourcing Time Series XAI related methodologies that includes both articles and publicly available code. This makes reproducibility of the paper's finding difficult, and assessment whether this XAI method is suitable as discussed in the articles or papers. Another potential reason for this could be the intent to commercialise the XAI methodology, hence not releasing the source code publicly.

Another reason could be that XAI on time series data is still on its infancy, and the potential XAI methods are still under development. As of this paper, XAI on time series is not as extensive as other data types. On this note, there is still a wide room for XAI time series to grow and have extensive XAI methodologies available for usage.

TEXT (NATURAL LANGUAGE PROCESSING) DATA

This section will cover a few of the methods used to create interpretations for text data specifically in the area of binary classification and an extension of the method to cover a multi-label use-case as well.

Apart from the most typical use of XAI methodologies such as LIME and SHAPely values, the aim is to assist the various stakeholders (engineers, users and auditors) with other methodologies that provide a more understandable and intuitive visualization as well.

The two methods explored so far are using τ-SS3 (insert reference) and the innvestigate (insert reference) library. The former is used as a surrogate model, training its own parameters with the given data, while the latter is used specifically with neural networks with an output O1 just before the softmax layer for analysis while applying the weights after the softmax layer O2 back to O1 and drawing inferences from there.

Both methods will eventually return a relevance score with each of the words, sentences or paragraphs which can be visually represented using a heatmap over each of the words, The methodologies though employed differently will give an insight into how the weights are assigned based on the models that were used.

τ-SS3 (A surrogate model method)

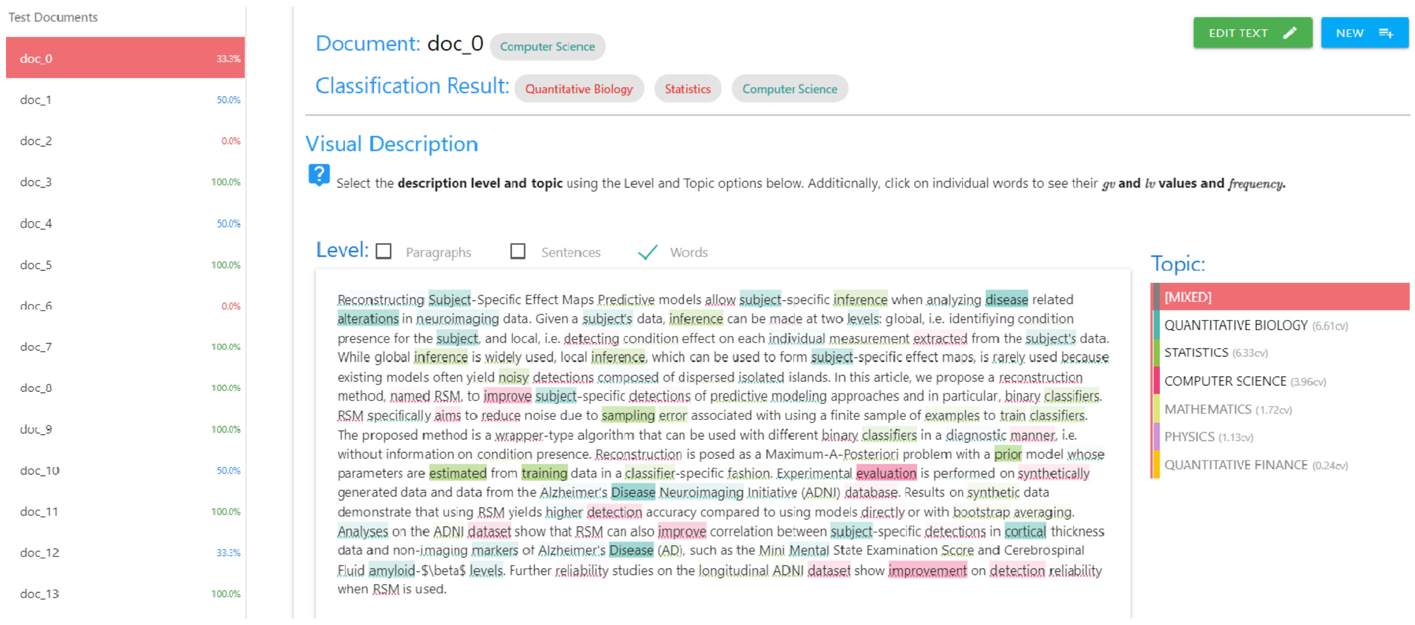
Using the τ-SS3 model, we can immediately get prediction results with the explanation element. This allows the engineer to immediately understand how the τ-SS3 model learns and eventually distributes the weights.

Two datatypes were explored namely a binary classification sentiment analysis, and a multi-label text classification. The details of the structure of the data shall not be discussed here and the XAI technique and visualizations will be the focus instead.

τ-SS3 visualization (multi-label classifier)

The following dataset is taken from kaggle, consisting of multiple titles and summaries labeled with different labels (one or more) spanning across 5 labels namely: "Computer Science, Physics, Mathematics, Statistics, Quantitative Biology, and Quantitive finance".

The visualization below shows how the τ-SS3 model assigns weights to the words heatmapped accordingly over the text. Each document in the evaluation set can be viewed separately.



Test Documents	
doc_0	33.3%
doc_1	50.0%
doc_2	0.0%
doc_3	100.0%
doc_4	50.0%
doc_5	100.0%
doc_6	0.0%
doc_7	100.0%
doc_8	100.0%
doc_9	100.0%
doc_10	50.0%
doc_11	100.0%
doc_12	33.3%
doc_13	100.0%

Document: doc_0 Computer Science

Classification Result: Quantitative Biology Statistics Computer Science

Visual Description

Select the **description level and topic** using the Level and Topic options below. Additionally, click on individual words to see their ***gv* and *lv* values and *frequency*.**

Level: ☐ Paragraphs ☐ Sentences ✓ Words

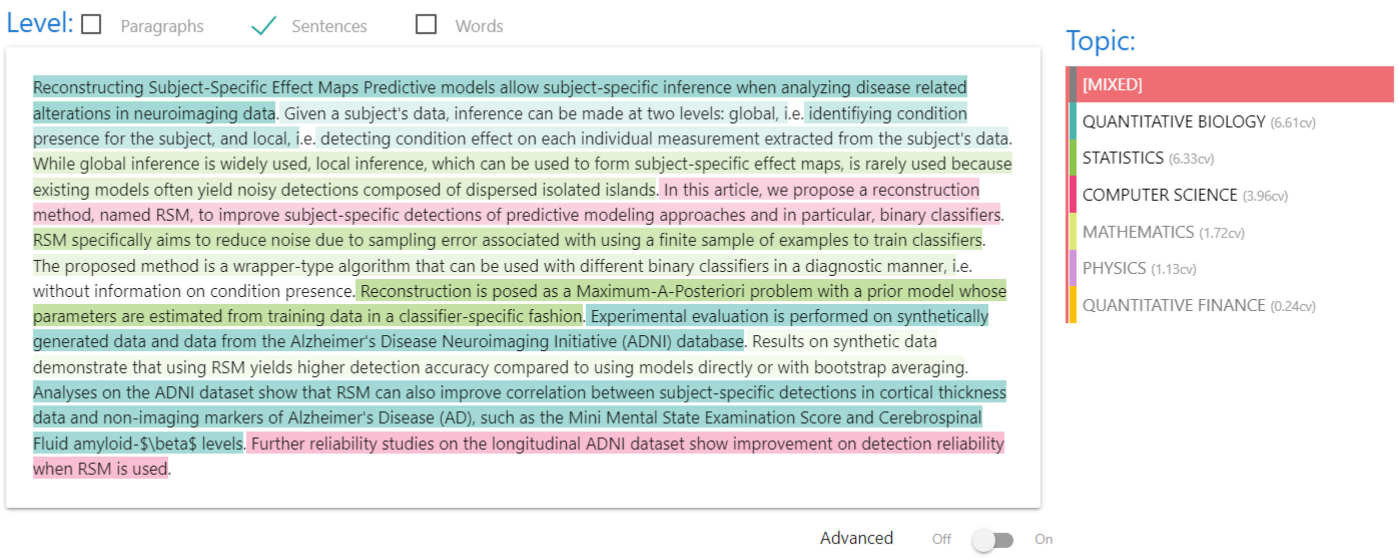
Reconstructing Subject-Specific Effect Maps Predictive models allow subject-specific inference when analyzing disease related alterations in neuroimaging data. Given a subject's data, inference can be made at two levels: global, i.e. identifiying condition presence for the subject, and local, i.e. detecting condition effect on each individual measurement extracted from the subject's data. While global inference is widely used, local inference, which can be used to form subject-specific effect maps, is rarely used because existing models often yield noisy detections composed of dispersed isolated islands. In this article, we propose a reconstruction method, named RSM, to improve subject-specific detections of predictive modeling approaches and in particular, binary classifiers. RSM specifically aims to reduce noise due to sampling error associated with using a finite sample of examples to train classifiers. The proposed method is a wrapper-type algorithm that can be used with different binary classifiers in a diagnostic manner, i.e. without information on condition presence. Reconstruction is posed as a Maximum-A-Posteriori problem with a prior model whose parameters are estimated from training data in a classifier-specific fashion. Experimental evaluation is performed on synthetically generated data and data from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database. Results on synthetic data demonstrate that using RSM yields higher detection accuracy compared to using models directly or with bootstrap averaging. Analyses on the ADNI dataset show that RSM can also improve correlation between subject-specific detections in cortical thickness data and non-imaging markers of Alzheimer's Disease (AD), such as the Mini Mental State Examination Score and Cerebrospinal Fluid amyloid-β levels. Further reliability studies on the longitudinal ADNI dataset show improvement on detection reliability when RSM is used.

Topic:

[MIXED]	
QUANTITATIVE BIOLOGY	(6.61cv)
STATISTICS	(6.33cv)
COMPUTER SCIENCE	(3.96cv)
MATHEMATICS	(1.72cv)
PHYSICS	(1.13cv)
QUANTITATIVE FINANCE	(0.24cv)

Figure 4.1: Multi-label classification for doc_0. 3 labels being predicted but the test document is a single labelled "COMPUTER SCIENCE". The word level relevances is represented here.

As seen in figure 4.1, each word associated with the label is being highlighted accordingly. Take pink for example, words with a pink highlight represent words that added weights to the label "COMPUTER SCIENCE", the darker the shade of the colour, the higher the weight the predictor gave it. The XAI technique can pick up the weights that the predictor gave it and displayed accordingly in the heat map above.



Reconstructing Subject-Specific Effect Maps Predictive models allow subject-specific inference when analyzing disease related alterations in neuroimaging data. Given a subject's data, inference can be made at two levels: global, i.e. identifiying condition presence for the subject, and local, i.e. detecting condition effect on each individual measurement extracted from the subject's data. While global inference is widely used, local inference, which can be used to form subject-specific effect maps, is rarely used because existing models often yield noisy detections composed of dispersed isolated islands. In this article, we propose a reconstruction method, named RSM, to improve subject-specific detections of predictive modeling approaches and in particular, binary classifiers. RSM specifically aims to reduce noise due to sampling error associated with using a finite sample of examples to train classifiers. The proposed method is a wrapper-type algorithm that can be used with different binary classifiers in a diagnostic manner, i.e. without information on condition presence. Reconstruction is posed as a Maximum-A-Posteriori problem with a prior model whose parameters are estimated from training data in a classifier-specific fashion. Experimental evaluation is performed on synthetically generated data and data from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database. Results on synthetic data demonstrate that using RSM yields higher detection accuracy compared to using models directly or with bootstrap averaging. Analyses on the ADNI dataset show that RSM can also improve correlation between subject-specific detections in cortical thickness data and non-imaging markers of Alzheimer's Disease (AD), such as the Mini Mental State Examination Score and Cerebrospinal Fluid amyloid-β levels. Further reliability studies on the longitudinal ADNI dataset show improvement on detection reliability when RSM is used.

🟥	[MIXED]
🟦	QUANTITATIVE BIOLOGY (6.61cv)
🟩	STATISTICS (6.33cv)
🟥	COMPUTER SCIENCE (3.96cv)
🟨	MATHEMATICS (1.72cv)
🟪	PHYSICS (1.13cv)
🟧	QUANTITATIVE FINANCE (0.24cv)

Figure 4.2: Multi-label classification for doc_0. The sentence level relevances is represented here

In figure 4.2, by selecting the sentence level, we see how the model identifies the labels by the sentences, in the same way, the XAI technique picks up the model's weight assignments and by means of a heatmap plotting to show the result. The darker the shade of the colour, the higher the weights.

The visualization can also provide the detailed value of the per-word, per-sentence, or per-paragraph weight-relevances which led to the prediction outcome as seen in fig 3 and fig 4 by turning on the UI with the "Advanced" button.

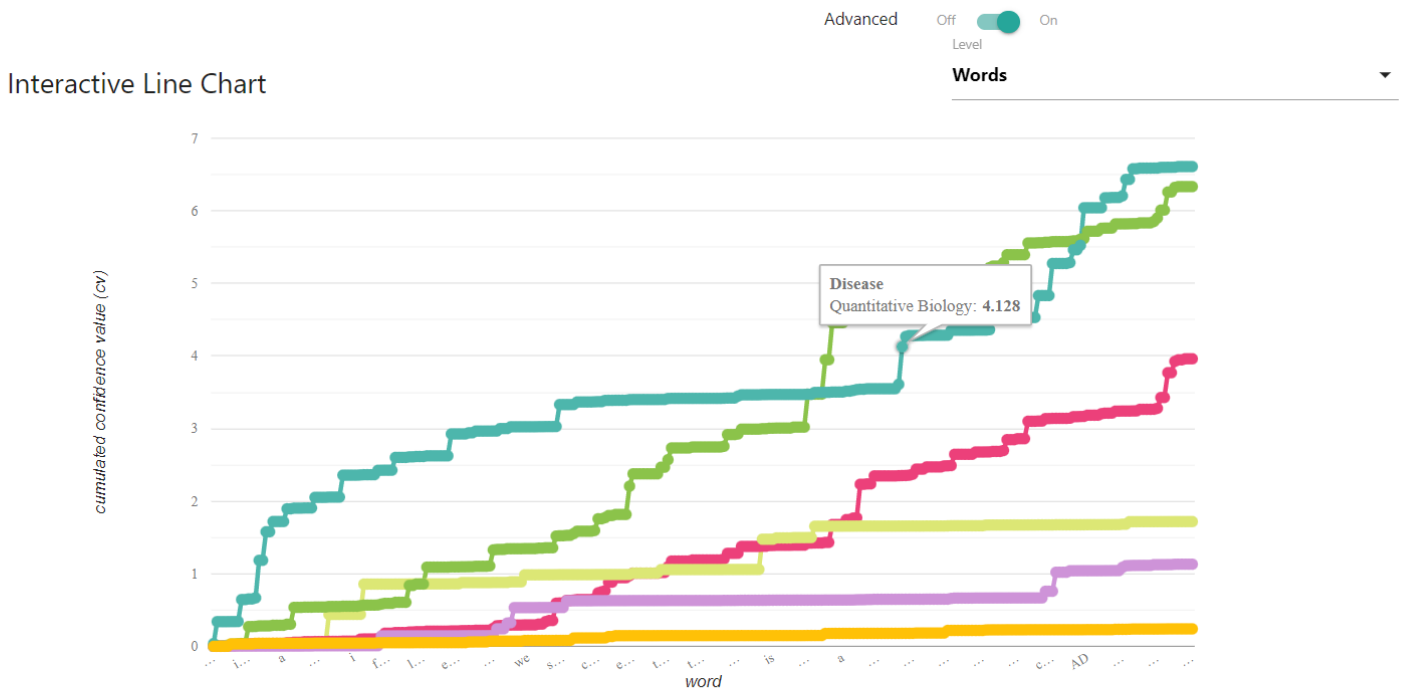


Figure 4.3: word-level weight relevance for each individual word. In the case of "QUANTITATIVE BIOLOGY", the word Disease is highlighted to show the associated value of 4.128 which raised the likelihood of the model's prediction by that quantum. 6 line charts are shown as a comparison to each other.

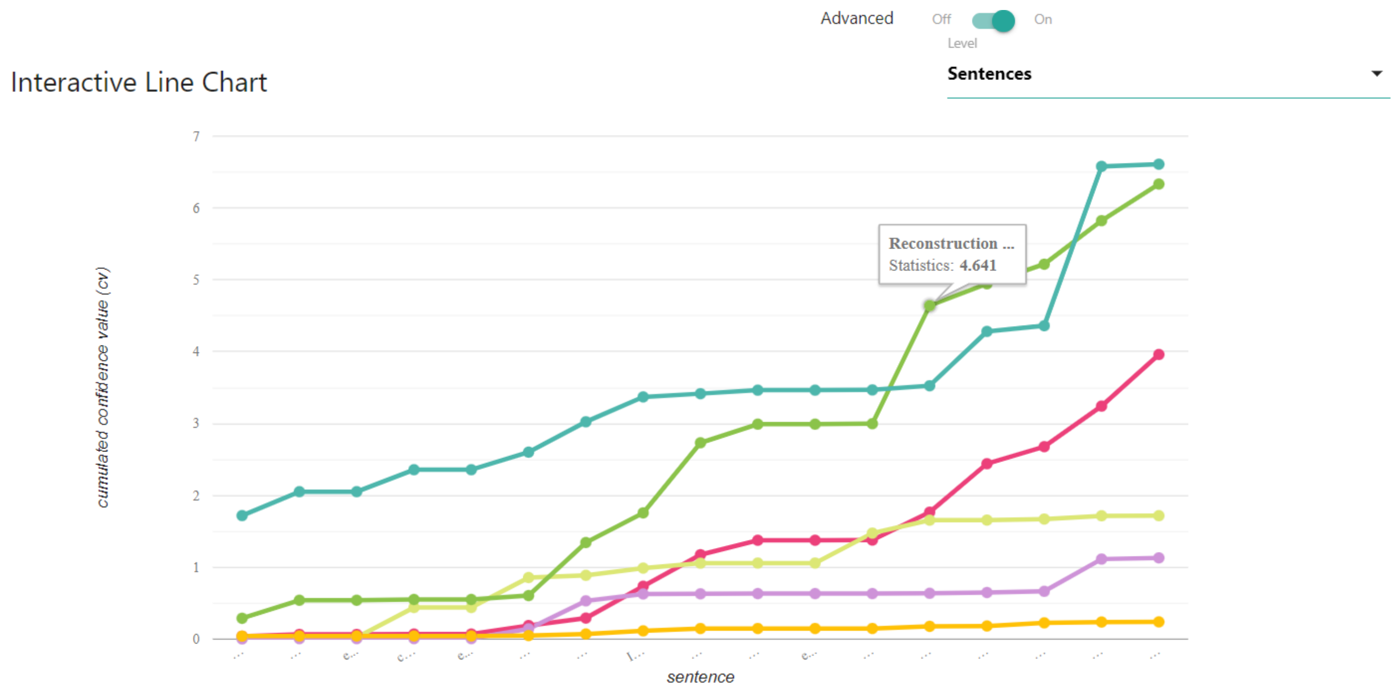


Figure 4.4: Sentence-level weight relevance for each individual sentence. In the case of "STATISTICS", the sentence beginning with Reconstruction is highlighted to show the associated value of 4.641 which raised the likelihood of the model's prediction by that quantum. 6 line charts are shown as comparison to each other.

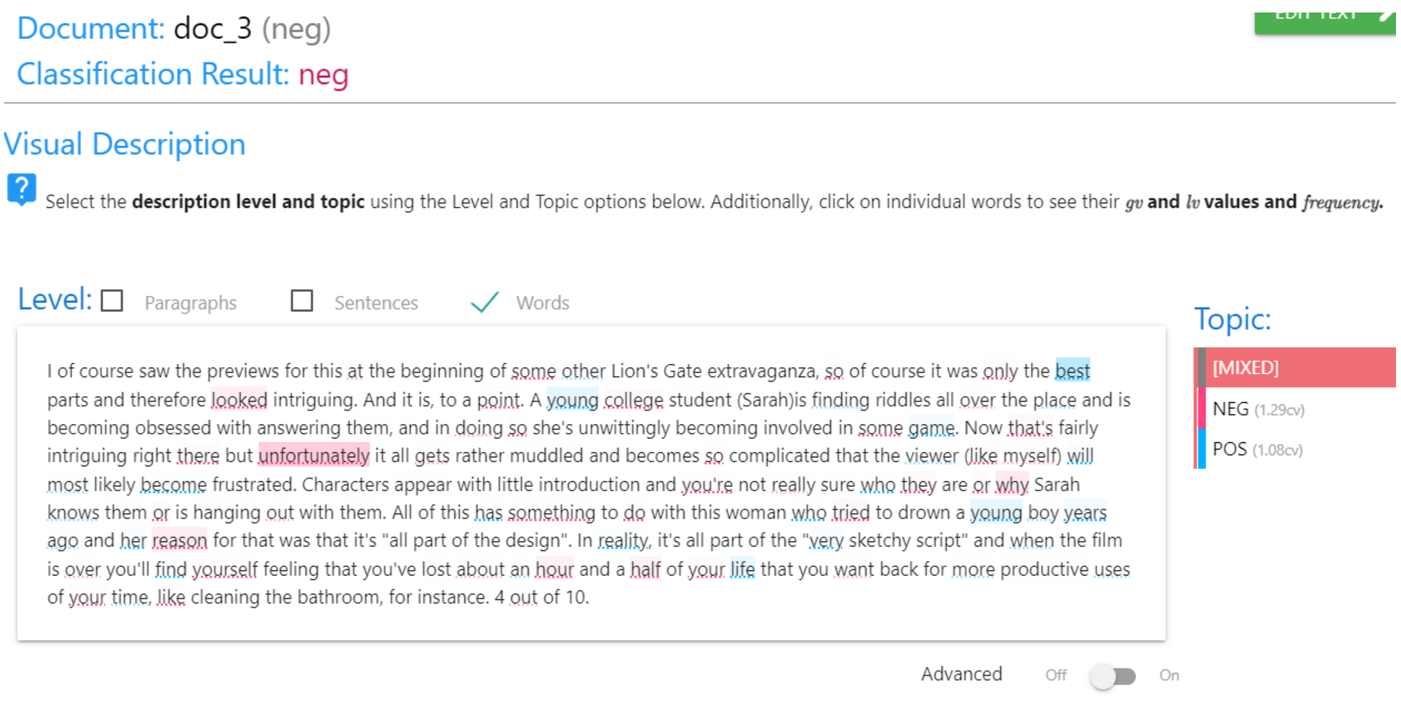
What is seen in figures 4.3 and 4.4 (and paragraph level but not shown) are considered in this model to give the total weight relevance value that eventually predicts the labels.

The engineer can also input a new text for the model to classify the text and provide the interpretation immediately as part of the interactive feature available to use.

τ-SS3 visualization (binary classifier)

The following dataset will cover binary classification for sentiment analysis of movie reviews being positive or negative reviews. It is a typical dataset that can be obtained from the IMDB dataset or the stanford dataset. Although multiclass can be analysed as well, to keep things simple and readable, we focused on just 2 classes to showcase the interpretation obtained by the technique. The prediction accuracy is approximately 80% but again, we are not building accurate models but studying the effects of the XAI techniques.

The following figures will show the same style of visualization but for a binary classifier. The methods to achieve such are called differently and looking into the documentation would provide further insights into their uses.



Document: doc_3 (neg)
Classification Result: neg

Visual Description

Select the **description level and topic** using the Level and Topic options below. Additionally, click on individual words to see their *gv* **and** *lv* **values and** *frequency***.**

Level: ☐ Paragraphs ☐ Sentences ✓ Words

I of course saw the previews for this at the beginning of some other Lion's Gate extravaganza, so of course it was only the best parts and therefore looked intriguing. And it is, to a point. A young college student (Sarah)is finding riddles all over the place and is becoming obsessed with answering them, and in doing so she's unwittingly becoming involved in some game. Now that's fairly intriguing right there but unfortunately it all gets rather muddled and becomes so complicated that the viewer (like myself) will most likely become frustrated. Characters appear with little introduction and you're not really sure who they are or why Sarah knows them or is hanging out with them. All of this has something to do with this woman who tried to drown a young boy years ago and her reason for that was that it's "all part of the design". In reality, it's all part of the "very sketchy script" and when the film is over you'll find yourself feeling that you've lost about an hour and a half of your life that you want back for more productive uses of your time, like cleaning the bathroom, for instance. 4 out of 10.

Topic:

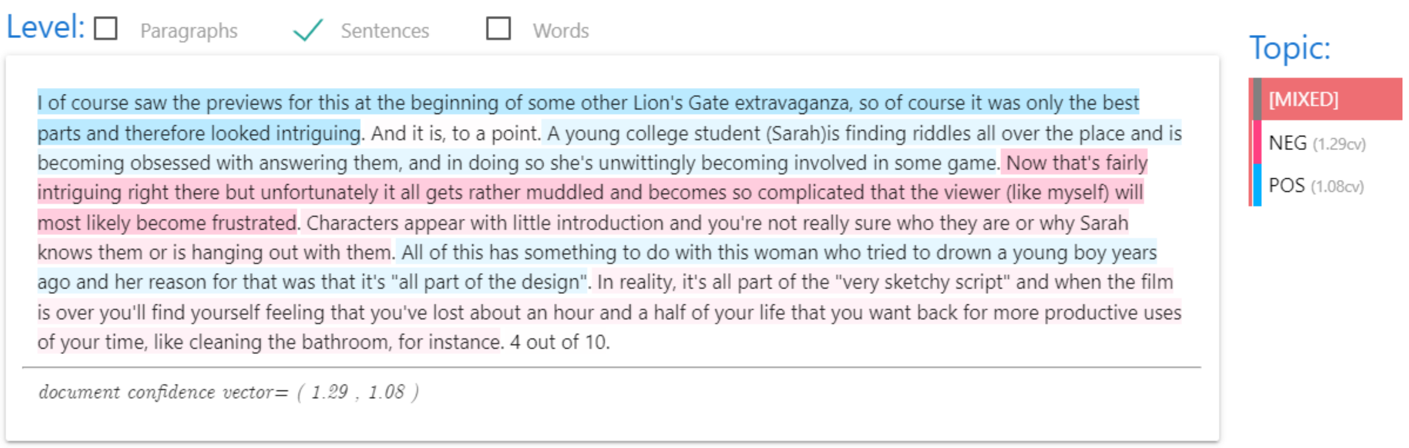
[MIXED]
NEG (1.29cv)
POS (1.08cv)

Advanced Off On

Figure 4.5: Word-level weight relevances for each word are provided. The heatmap colour red and blue are the weight-relevances given to a negative and positive sentiment respectively.

Figure 4.5 provides insight into the whole text, and the word "unfortunately" has the biggest effect on classifying this document to be negative, while the word "best" provided the weights that swing the prediction to the positive side.

At this point, we can arguably say after reading the whole text that it is insufficient to draw a conclusion regarding the sentiment of this document.



I of course saw the previews for this at the beginning of some other Lion's Gate extravaganza, so of course it was only the best parts and therefore looked intriguing. And it is, to a point. A young college student (Sarah)is finding riddles all over the place and is becoming obsessed with answering them, and in doing so she's unwittingly becoming involved in some game. Now that's fairly intriguing right there but unfortunately it all gets rather muddled and becomes so complicated that the viewer (like myself) will most likely become frustrated. Characters appear with little introduction and you're not really sure who they are or why Sarah knows them or is hanging out with them. All of this has something to do with this woman who tried to drown a young boy years ago and her reason for that was that it's "all part of the design". In reality, it's all part of the "very sketchy script" and when the film is over you'll find yourself feeling that you've lost about an hour and a half of your life that you want back for more productive uses of your time, like cleaning the bathroom, for instance. 4 out of 10.

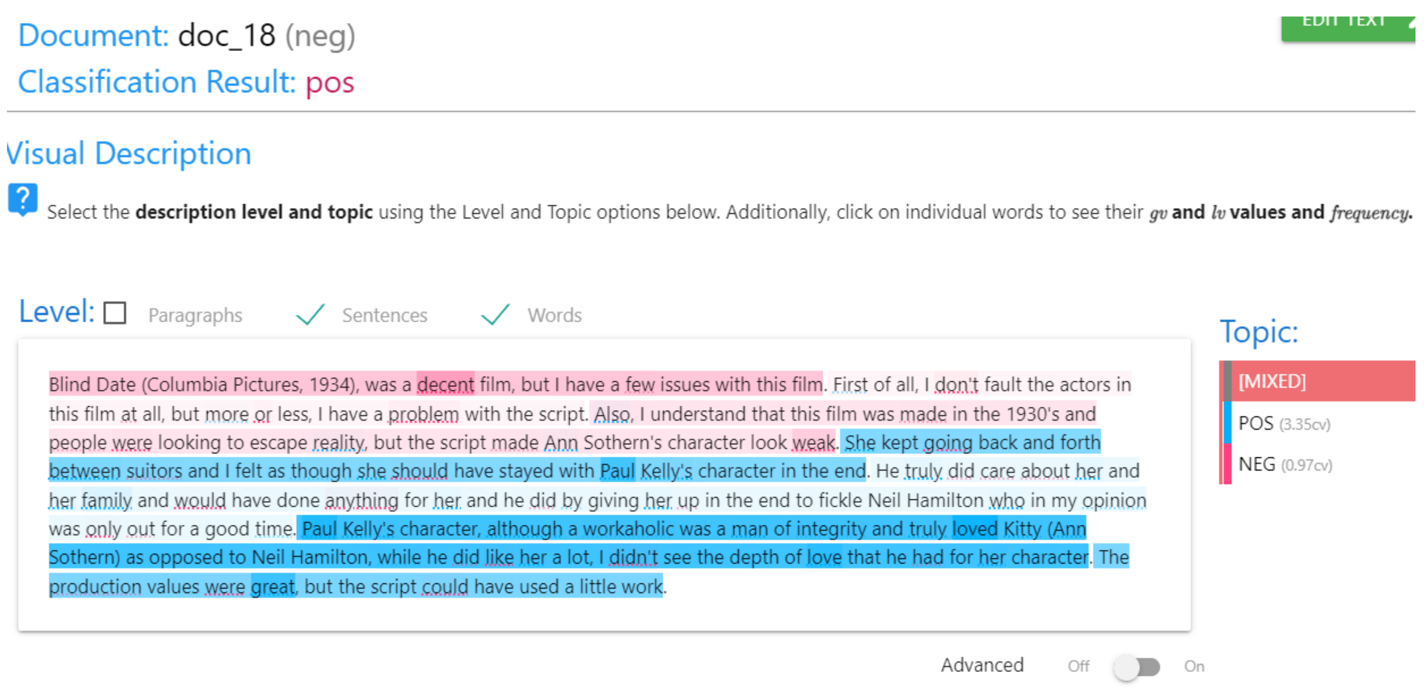
document confidence vector= (1.29 , 1.08)

Figure 4.6: Sentence-level weight relevances for the same document in figure 5.

We can draw better insights by looking at the sentence level weights, which is somewhat more consistent with the prediction, however, we can see that the first sentence that some might argue that it is negative but seen by the model as positive, because the word “best” was in it, and no other directly obvious negative word is in that sentence.

This is where employing XAI technique provides useful insight into how the model is performing, and whether as an engineer we can do better by improving the model, or as the end-user, agree or disagree with the prediction of the model.

We will further this investigation by looking into a wrongly predicted case by the model and what the XAI technique manages to bring out.



Document: doc_18 (neg)
Classification Result: pos

Visual Description

Select the **description level and topic** using the Level and Topic options below. Additionally, click on individual words to see their *gv* **and** *lv* **values and** *frequency*.

Level: ☐ Paragraphs ✓ Sentences ✓ Words

Blind Date (Columbia Pictures, 1934), was a decent film, but I have a few issues with this film. First of all, I don't fault the actors in this film at all, but more or less, I have a problem with the script. Also, I understand that this film was made in the 1930's and people were looking to escape reality, but the script made Ann Sothern's character look weak. She kept going back and forth between suitors and I felt as though she should have stayed with Paul Kelly's character in the end. He truly did care about her and her family and would have done anything for her and he did by giving her up in the end to fickle Neil Hamilton who in my opinion was only out for a good time. Paul Kelly's character, although a workaholic was a man of integrity and truly loved Kitty (Ann Sothern) as opposed to Neil Hamilton, while he did like her a lot, I didn't see the depth of love that he had for her character. The production values were great, but the script could have used a little work.

Topic:

[MIXED]	
POS	(3.35cv)
NEG	(0.97cv)

Advanced Off On

Figure 4.7: doc_18 is a negative sentiment document predicted positive by the model. The word and sentence-level weight relevances are both shown here.

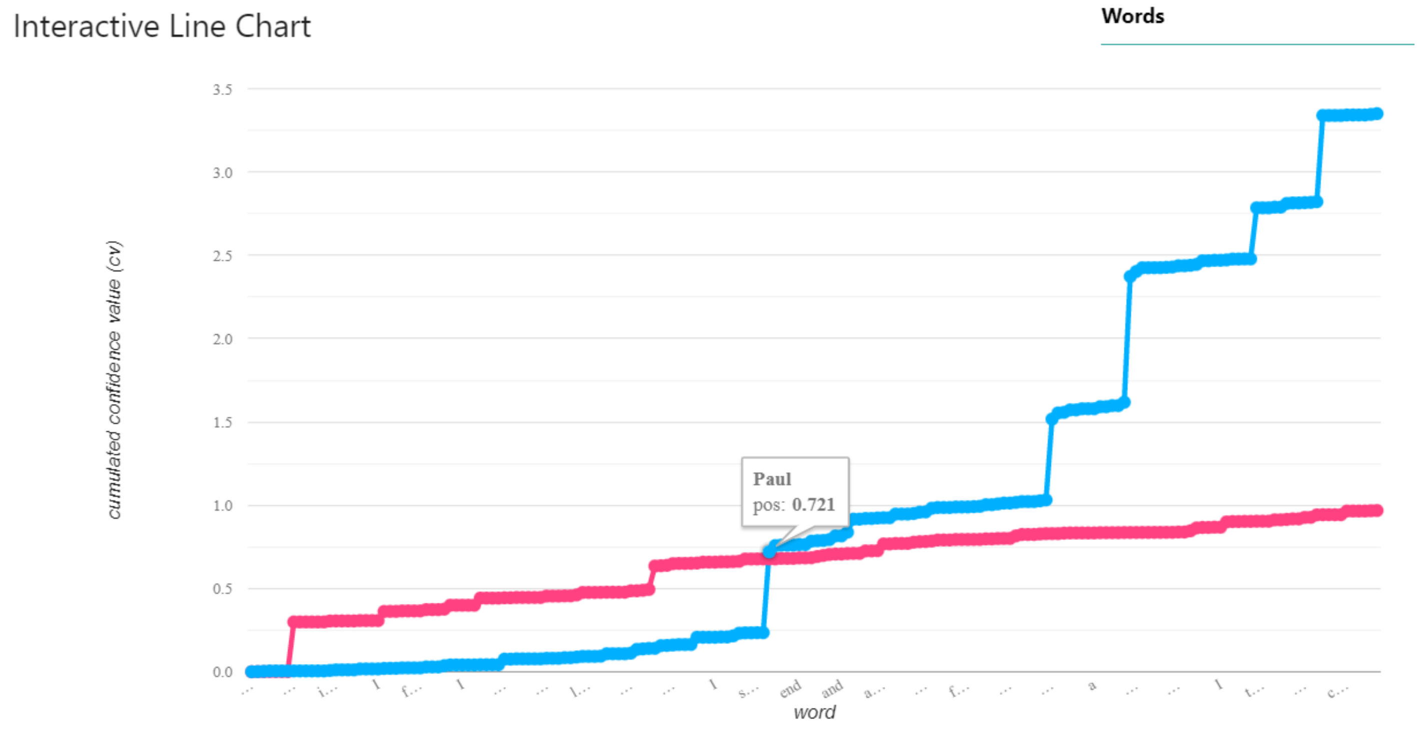


Figure 4.8: Word-level line chart for the document seen in figure 4.7, the word "paul" is highlighted to see the weight given to it.

Looking closely at figure 4.8, we can see that the prediction is confidently placed as positive, but as readers, we can also agree that the document leans towards the negative sentiment.

Both figure 4.7 and figure 4.8 provide good insight into the behavior of the model and gives the engineer an understanding of how to improve the model.

Looking at figure 4.7, we can see the word “decent” carries a heavy weight to predict the document as negative, while words like “Paul” and the sentence it lies in pushed the prediction to positive. Figure 8 gives us additional information that the word “Paul” gave a significant jump in the score towards positive.

Now, we can immediately draw a conclusion about the model, that it has started to learn that the word “Paul” is always positive, and the word “decent” is negative. While decent can be argued when used collectively in a sentence can be either negative or positive (study of chargrams will add better insights into this problem), the word “Paul” is definitely overfitted.

The XAI technique provided a means to understand the model’s shortcomings or in the event of a very good model, the basis of its accuracy.

Innvestigate library method

The innvestigate library is a tool that is usually used in CV applications as it covers many techniques that allow inference into the layers and the nodes of a neural network model. The same case can be applied to NLP text data. We will study the techniques and visualization that we achieved by doing so.

There are many different analysis methods to be applied, and each one should be applied according to how we believe the model behaves or applied altogether to simply study the differences.

The various methods are:

[**'input'**, **'random'**, **'gradient'**, '**gradient.baseline**', '**input_t_gradient**', '**deconvnet**', '**guided_backprop**', '**integrated_gradients**', **'smoothgrad'**, '**lrp**', '**lrp.z**', '**lrp.z_IB**', '**lrp.epsilon**', 'l**rp.epsilon_IB**', '**lrp.w_square**', '**lrp.flat**', '**lrp.alpha_beta**', '**lrp.alpha_2_beta_1**', '**lrp.alpha_2_beta_1_IB**', '**lrp.alpha_1_beta_0**', '**lrp.alpha_1_beta_0_IB**', '**lrp.z_plus**', '**lrp.z_plus_fast**', '**lrp.sequential_preset_a**', '**lrp.sequential_preset_b**', '**lrp.sequential_preset_a_flat**', '**lrp.sequential_preset_b_flat**', '**lrp.sequential_preset_b_flat_until_idx**', '**deep_taylor**', '**deep_taylor.bounded**', '**deep_lift.wrapper**', '**pattern.net**', '**pattern.attribution**']"

For the purposes of understanding, we will only explore a few of the methods above.

To use the method, we would need to build a network consisting of the final softmax layer and a model without the softmax layer. Unfortunately, this can only be done by using keras directly with tensorflow 1.12, the library currently does not support tf2.

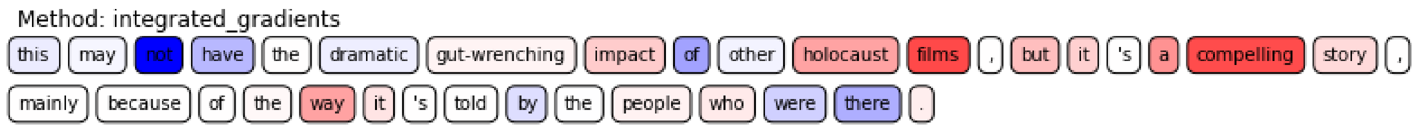
We then apply the weights obtained by the model with softmax onto the model without softmax and choose which layer we would like inference into.

Innvestigate visualization

The dataset is from the stanfordTreeSentimentBank and we are classifying positive and negative movie reviews.

The classifier is a convolutional neural network, which was experimented in Arras et al. (2017b). As shown below, the architecture has a convoluation layer, convolving word embeddings of every two words, followed by a max pooling layer and a softmax layer.

Let's look at a correctly predicted example:



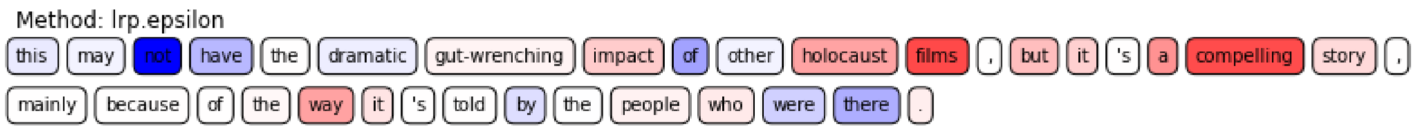
-0.10	this	-0.06	other	0.20	story	0.00	told
-0.05	may	0.51	holocaust	0.00	,	-0.18	by
-1.42	not	1.01	films	0.00	mainly	-0.01	the
-0.39	have	0.01	,	0.00	because	0.07	people
0.00	the	0.36	but	0.00	of	0.11	who
-0.09	dramatic	0.26	it	0.04	the	-0.26	were
0.06	gut-wrench	0.00	's	0.51	way	-0.45	there
0.28	impact	0.61	a	0.15	it	0.10	.
-0.52	of	0.99	compelling	0.00	's		

Figure 4.9: Review no. 97 predicted correctly as positive, heatmap of the method integrated gradients method, and the relevance scores for each word.

As the model is predicted as positive, the higher the positive value of the score, the more that word pushes the prediction to positive, while the more negative will be towards a negative prediction.

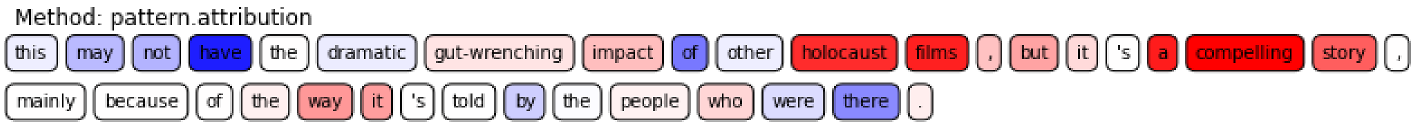
NOTE: The colour of the heatmap is based on the prediction, the darker the RED it is, the more weightage it provides to the prediction, the darker the BLUE it is, the more it pulls it away from the prediction.

Analyzing figure 4.9 alone gives us a certain idea on how the model determines the statement is positive, and perhaps we can draw a reasonable conclusion that it is somewhat, albeit not too confident that the words "a compelling story" does indeed suggest a positive review while the words "Holocaust films" as it stands by itself is not sufficient evidence.



-0.12	this	-0.06	other	0.20	story	0.00	told
-0.06	may	0.52	holocaust	0.00	,	-0.19	by
-1.44	not	1.02	films	0.00	mainly	-0.01	the
-0.40	have	0.01	,	0.00	because	0.08	people
0.00	the	0.36	but	0.00	of	0.11	who
-0.10	dramatic	0.26	it	0.04	the	-0.27	were
0.06	gut-wrench	0.00	's	0.52	way	-0.46	there
0.29	impact	0.62	a	0.15	it	0.10	.
-0.53	of	1.01	compelling	0.00	's		

Figure 4.10: Heatmap of the method lrp epsilon and its associated relevance scores.


Method: pattern.attribution

this may not have the dramatic gut-wrenching impact of other holocaust films , but it 's a compelling story , mainly because of the way it 's told by the people who were there .

-0.69	this	-0.69	other	6.68	story	-0.06	told
-2.90	may	8.74	holocaust	0.00	,	-2.03	by
-3.12	not	9.25	films	0.00	mainly	-0.14	the
-9.31	have	2.51	,	0.00	because	0.58	people
0.00	the	3.75	but	0.00	of	1.72	who
-0.81	dramatic	1.56	it	0.63	the	-1.48	were
1.14	gut-wrench	0.00	's	4.28	way	-4.85	there
2.69	impact	9.45	a	4.02	it	0.74	.
-5.62	of	10.64	compelling	0.00	's		

Figure 4.11: Heat map of the method pattern.attribution and its associated relevance scores.

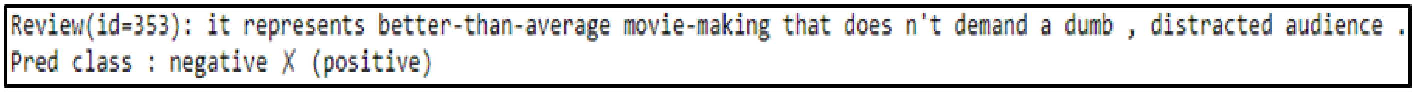
We compare further with figure 4.10 and figure 4.11, we notice that there is consistency with results except some difference in the pattern attribution method, this is also due the way the method is used to derive the relevance scores.

As such we can draw a decent conclusion about the model after looking into the XAI techniques, that a big part of the correct prediction falls with the words “holocaust films” and “a compelling story”, and the part where it says “not have” seems to suggest a negative review.

The benefit of such an analysis is to the stakeholder, which in this case is the modelling engineer, is to decide whether to improve the model by looking at the word embeddings and/or training dataset to see if we can further generalize the phrase “holocaust films” so that it does not simply get classified as positive.

Another insight as some might already have seen is that the dataset was not cleaned to remove the punctuations and weights were given to it. This is done deliberately to have greater insight into the state of the trained model, that certain improvement in the EDA could be performed to better generalize the model and not see that these punctuations are affecting the result.

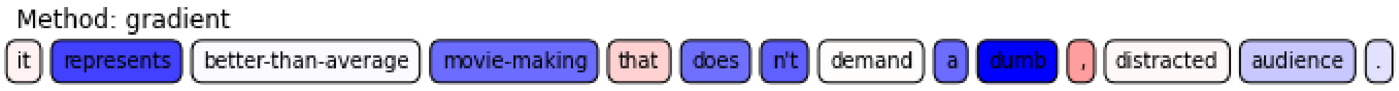
To gain better insights into the methods and understanding, let's investigate an example where the model predicted wrongly and what the XAI technique shows us:



```
Review(id=353): it represents better-than-average movie-making that does n't demand a dumb , distracted audience .
Pred class : negative X (positive)
```

Figure 4.12: Negative class was predicted but the sentiment is actually positive.

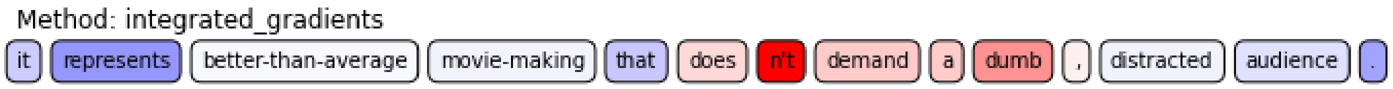
As someone reading this review, I had to re-read it in order to finally understand that indeed, this is a positive sentiment review, therefore, we can also already guess why the model is struggling to predict this correctly. Let's take a look.



0.13	it	0.46	that	-1.51	a	-0.57	audience
-1.94	represents	-1.46	does	-2.63	dumb	-0.31	.
-0.06	better-than-average	-1.63	n't	0.99	,		
-1.52	movie-making	0.04	demand	0.08	distracted		

Figure 4.13: Heatmap of the simple method gradient and its associated relevance scores.

The gradient method was particularly shown here to draw out certain concerns with using the innvestigate library, for many who knows the gradient method, it generally does not provide a good understanding of the model, in fact, it is known already that other methods are superior in getting the correct score (I.e.: integrated gradients). However, this also suggests that a deeper understanding of what each method and the application plays a big role in ensuring that the result is then not biased or achieved out of ignorance.



-0.36	it	-0.39	that	0.36	a	-0.21	audience
-0.75	represents	0.28	does	0.77	dumb	-0.67	.
-0.05	better-than-average	1.82	n't	0.11	,		
-0.08	movie-making	0.37	demand	-0.09	distracted		

Figure 4.14: Heatmap of the method integrated gradients and its associated relevance scores.

From figure 4.14, we can see clearer and make an inference. The issue with the punctuations is ever more so prominent as well but studying the heatmap, the whole phrase "does n't demand a dumb," suggest strongly that this statement is a negative sentiment.

Personally, I am inclined to agree based on that alone, however, the full sentence and its semantic meanings when pieced together is vaguely yet also clearly positive. Discounting the fact that the positive values represented by the words “represents”, “that”, “audience”, and “.” is not meaningful to predict positive, we can understand why the model struggles with this review.

Agreeably, this sentence is hard even for normal readers, but the greater benefit is that, as an engineer desiring to improve my model, I can think of ways to improve my model in catching the sematic meanings better and to clean my dataset more.

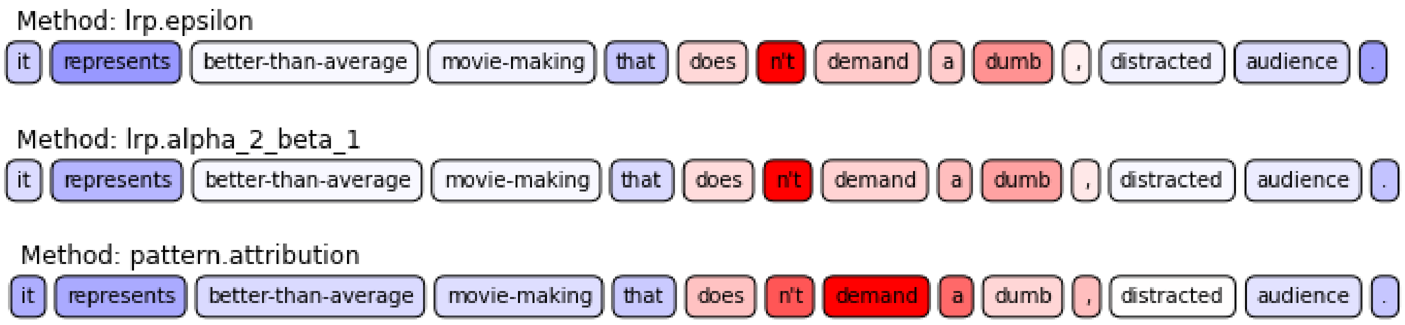


Figure 4.15: Heatmap of other methods showing similar difficulties faced by the model.

We can see from 3 other viable methods namely lrp.epsilon, lrp.a2b1, and the pattern.attribution all having similar difficulties when it comes to this text.

Application on foreign language dataset

Both methods mentioned above were tested with a live dataset in German language testing for profanity or offensive language. The purpose of which is to draw inferences using the XAI techniques on the model.

Three different word embeddings were used, fasttext[1], devmount[2], and self-trained word embeddings. However, the model results were all similar between 50%-65% accuracy.

The model used for the innvestigate method is a CNN model with a single convolution layer and a max pooling layer.

Analysis was done on a few test cases namely: “Hello”, “Hello, Good Morning”, and an obvious offensive case “Go die you son of a bitch”.

German Text	English Translated
Hallo	Hello
Hallo guten morgen	Hello good morning
Geh sterben du hurensoh	Go die you son of a bitch

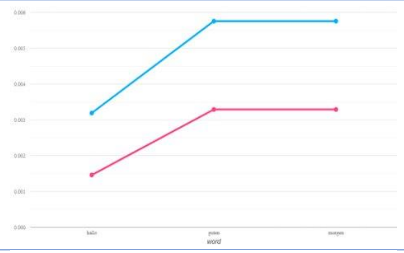
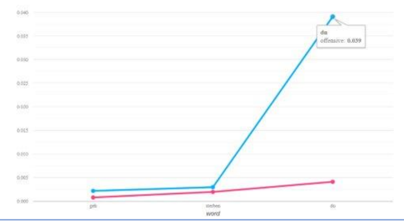
[1] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, *Bag of Tricks for Efficient Text Classification*

[2] https://devmount.github.io/GermanWordEmbeddings/

The results for the innvestivate method are as follows:

Prediction Result	Integrated Gradients	LRP.epsilon	Pattern Attribution
Offensive	hallo 0.41 hallo	hallo 0.43 hallo	hallo 0.88 hallo
Normal	hallo guten morgen 0.41 hallo 0.00 guten 0.50 morgen	hallo guten morgen 0.43 hallo 0.00 guten 0.50 morgen	hallo guten morgen 0.88 hallo 1.98 guten 4.78 morgen
Offensive	geh sterben du <UNK> 0.41 geh 0.00 sterben 0.50 du 0.11 <UNK>	geh sterben du <UNK> 0.43 geh 0.00 sterben 0.50 du 0.11 <UNK>	geh sterben du <UNK> 0.88 geh 1.98 sterben 4.78 du 0.41 <UNK>

The results for the t-ss3 model are as follows:

Prediction Result	Word Analyzed + Score	Relevances
Offensive	hallo Offensive: (1, 0.0031902480765139203) Normal: (0, 0.0014627126473799191)	
Offensive	hallo, guten morgen Offensive: (1, 0.0057524815813273155) Normal: (0, 0.003290131248245563)	
Offensive	geh sterben du hurensoh Offensive: (1, 0.039064584074413308) Normal: (0, 0.004099351978933172)	

Analysis of results

The result is telling of the performance of model not being able to capture the sentiment of a clearly normal phrase, and yet, even for the clearly offensive case, the XAI picked out that the determinant word for offensive fell on the word “du”. The same word “du” semantically can be used to describe positive attributes, but perhaps in the dataset, the usage has been far too often in the negative way.

On the first look, there might also be a sensing that shorter phrases are often associated with offensive, however, from the XAI relevances, the difference between offensive and normal is simply too close to each other and can conclude that the model is guessing already as opposed to having a very sure prediction.

FUTURE IMPROVEMENT

In the analysis of NLP using τ-SS3 model and innvestigate library, the insights obtained from the XAI techniques used led to the conclusion that the data that the model is trained on is not sufficient to differentiate normal from offensive, but also, that the cleaning of the data can be done further as some offensive words are still not recognized in the vocabulary.

Other techniques that can be used are to concatenate embeddings or to also create meta-embeddings between different pre-trained embedding sets to improve the training further.

Exploring the use of other packages/libraries such as Captum (which has started to incorporate the innvestigate methodologies within) can be a more user-friendly, intuitive and community-supported way forward as to approaching the XAI for NLP data. Surrogate models such as τ-SS3 has its place in terms of the specific classification problem that it handles yet limited to its own architecture that may produce prediction scores lower than another model. The best way forward to achieve similar visualization and interactability is to also code that portion of it in a more generic way that can be applied to another model.

Looking at time series data type, one potential improvement is to introduce more models to solve a Multivariate time series problem, and appropriate XAI methods to makes sense of these models. As of this writing, there is a lack of appropriate Time Series Models to solve a Multivariate Time series problem, and appropriate XAI methods to interpret or “un-box” these models. Generally, time series problems require different assumptions and solving methods as compared to other tabular data type problems. By having more model architectures to solve Multivariate Time Series, perhaps we can then proceed to determining if the model can be categorised as “interpretable model”, and if not think of a potential XAI method to understand the model’s decision-making process.

Another potential improvement would be supporting publicly available open source XAI methods or Machine Learning Models. Due to a lack of publicly available XAI methodologies available, assisting the open-source projects of existing XAI methodologies or Machine Learning Models. This could be in the form of doing up a documentation, developing new enhancements, or creating new case studies. Through these activities, we can improve the existing XAI methodologies and interpretable models.

FUTURE WORK

To further enhance the initial work we have done, the following are some of the potential enhancements that could value-add this XAI study:

- Include adversarial attacks, to test robustness of A.I model
- Code an open-sourced python package that includes all the XAI methods mentioned in this report
- Create a github page to continue tracking and compare new XAI methods or packages available to gauge which ones have the best explanations
- Code a web deployment which have a different dropdown box to select different XAI methods in order to display information for different stakeholders

SUMMARY OF XAI METHODOLOGIES

On 25th May 2022, Singapore Government Agencies Infocomm Media Development Authority (IMDA) and Personal Data Protection Commission (PDPC) launched A.I. Verify, a toolkit that contains XAI which aims to promote transparency between companies and stakeholders. Below table is a comparison of our findings versus the A.I. Verify toolkit.

	Tabular	Image	Text	Timeseries	Web deployment	Adversarial attacks
AI verify	Lime, Shap	Lime, Shap	✗	✗	✓	AIF360(IBM)
RP-AI2Labs - Nvidia	Lime, Shap, counterfactual, scoped rules(anchor)	Grad-cam	τ- SS3, iNNvestigate	VAR, neural prophet, LioNets	✗	✗

CONCLUSION

Making Artificial Intelligence understandable by all is essential to build trust and facilitate widespread adoption. This paper explored and evaluated some XAI techniques used for different data type. While this sub-field is researched mainly in academic circles, the desired outcome is to have XAI available and easily accessible to all users, whether that is an AI Engineer, Auditor or End User.

REFERENCES

[1] C. Molnar, *Interpretable Machine Learning*. Accessed: May 27, 2022. [Online]. Available: https://christophm.github.io/interpretable-ml-book/index.html#summary

[2] Google People+AI Research (PAIR), "What-If Tool." Google. Accessed: Jul. 22, 2022. [Online]. Available: https://pair-code.github.io/what-if-tool/

[3] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR." arXiv, Mar. 21, 2018. Accessed: Jul. 22, 2022. [Online]. Available: http://arxiv.org/abs/1711.00399

[4] D. Rothman, *Hands-On Explainable AI (XAI) with Python: Interpret, visualize, explain, and integrate reliable AI for fair, secure, and trustworthy AI apps*. Packt Publishing, 2020. [Online]. Available: https://books.google.com.sg/books?id=2f30DwAAQBAJ

[5] Jeff Larson, Surya Mattu, Lauren Kirchner and Julia Angwin, "How We Analyzed the COMPAS Recidivism Algorithm," May 23, 2016. https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm

[6] H. Lakkaraju, J. Adebayo, and S. Singh, "Explaining Machine Learning Predictions: State-of-the-art, Challenges, and Opportunities." [Online]. Available: https://explainml-tutorial.github.io/neurips20

[7] Microsoft Research AI, "Explainable Boosting Machine." Microsoft. [Online]. Available: https://interpret.ml/docs/ebm.html

[8] Microsoft Research AI, "InterpretML Github." [Online]. Available: https://github.com/interpretml/interpret/

[9] Fraunhofer Heinrich Hertz Institute and Technical University of Berlin, "LRP toolbox." [Online]. Available: https://github.com/sebastian-lapuschkin/lrp_toolbox

[10] META, "Captum." [Online]. Available: https://github.com/pytorch/captum

[11] Jacob Gildenblat and contributors, "Pytorch-gradcam." [Online]. Available: https://github.com/jacobgil/pytorch-grad-cam

[12] C. A. Sims, "Macroeconomics and reality," *Econometrica: journal of the Econometric Society*, pp. 1–48, 1980.

[13] I. Mollas, N. Bassiliades, and G. Tsoumakas, "LioNets: A Neural-Specific Local Interpretation Technique Exploiting Penultimate Layer Information," *Appl Intell*, May 2022, doi: 10.1007/s10489-022-03351-4.

[14] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *2008 International Conference on Prognostics and Health Management*, 2008, pp. 1–9. doi: 10.1109/PHM.2008.4711414.

[15] O. Triebe, H. Hewamalage, P. Pilyugina, N. Laptev, C. Bergmeir, and R. Rajagopal, "NeuralProphet: Explainable Forecasting at Scale." arXiv, Nov. 29, 2021. Accessed: Jul. 15, 2022. [Online]. Available: http://arxiv.org/abs/2111.15397

[16] O. Triebe, N. Laptev, and R. Rajagopal, "AR-Net: A simple Auto-Regressive Neural Network for time-series." arXiv, Nov. 27, 2019. Accessed: Jul. 15, 2022. [Online]. Available: http://arxiv.org/abs/1911.12436

[17] Department of Energy, United States of America (USA), "National Solar Radiation Database." https://nsrdb.nrel.gov/data-sets/tmy (accessed Jul. 25, 2022).

[18] Neural Prophet, "Renewable Energy: Forecasting hourly solar irradiance." https://neuralprophet.com/html/energy_solar_pv.html (accessed Jul. 25, 2022).