Algorithm Audit: Key concepts of Interpretability

By Waad ALMASRI

Let us start with an example

- data set: the cardiovascular disease dataset
- Logistic regression
- Fit then model and print its summary
- The summary function in statsmodel helps users understand which features X contributed the most to the target y (here the cardiovascular disease) using the model's coefficients
- Just like lineart regression, these coefficients are weights applied to the features
- But, their combination exponent is a logistic regression → difficult interpretation
- The coefficients of a logistic regression = the log odds → to get the odds, we apply the exponential function → better interpretability → Logistic regression = intrinsically interpretable model
- NB: the odds do not represent the features importance! They differ in scale & in statistical significance!
- To obtain what features matters the most, an **approximation** would be by multiplying the features' coefficients (the log odds) by their standard deviation: this method is called model-specific Feature Importance method, which is part of the global model interpretation methods

NB: the odds = the probability of a positive acs over the probability of a negative case; example: the probability of rain is 60% or the odds of rain are 3:2 or 1.5

Model interpretability method types

- Model-specific: a method that can only be used by a specific model and not by another
- Model-agnostic: a method that can work with any model

Model interpretability scopes

- Global holistic interpretation:
  - predictions are explainable because the model as a whole is simple to explain
  - Example: Linear regression
- Global modular interpretation:
  - Some parts of the model are explainable
  - Example: Feature importance
- Local single-prediction interpretation:
  - A single prediction can be explained
  - Example: Lime

What impedes interpretability?

- Dimensionality
- Non-linearity:
  - Replace the linear model by a non-linear one
  - Feature engineering with the help of domain experts (i.e., when dealing with height and weight features, we usually replace them by the body mass index (BMI), which is weight/(height)^2
- Outliers in data:
  - Explicit outliers must be removed
  - Hidden outliers: harder to detect because they are only perceived anomalous in the context of other features (i.e., age = 18 years, work experience = 20 years, here either of these values is erroneous)
- Interactivity:
  - Like the BMI
  - Interactions can add unnecessary complexity to the model by finding spurious correlations (meaningless & accidental)
- Non-monotonicity:
  - Monotonic functions are either increasing or decreasing throughout their entire domain
  - All linear relationships are monotonic
  - Not all monotonic relationship are necessary linear

Limitations of traditional model interpretation methods

They only cover surface-level questions about the model:

- The model's performance
- How hyperparameter tuning could impact the model's performance
- What patterns exist between the features & the target

They do not answer the questions of why & how is the model working?

Intrinsically (globally &/or locally) interpretable models

- Generalized linear models: global & local interpretability
- Decision trees: global & local interpretability
- K nearest neighbor: local interpretability
- Naive bayes: global & local interpretability

Generalized linear models - Linear regression

The coefficients & intercept. The coefficients are the features' weights; they tell how the target varies on every feature.
Model interpretation using the feature (f) ' coefficients (v) :

- Continuous feature: for every one unit increase in f, the target changes by v, if all other features stay the same
- Binary feature: if f=1, y changes by v, if all other features stay the same
- Categorical feature:
  - If converted to an ordinal feature: it becomes a continuous feature
  - If one-hot-encoded: it becomes a binary feature
- Intercept:
  - It is not a feature, but it means if all features were null, then the target takes its value.
  - When features are standardized with mean 0 → it means if all features have their mean values, then the target takes its value
- Feature importance = coefficients divided by their standard deviations or what is called t-statistic
  - Always look at the p-value to check the level of significance
  - In OLS, H0 (the null hypothesis): the feature is not significant → for p-value<=0.05, we can reject H0 → feature is significant

Decision trees

- Classification & Regression Tree (CART) is a white-box model
- It is expressed by a formula $\hat{y} = \sum_{m=1}^{M} \mu_m I\{x \epsilon R_m\}$; I identity function i.e., if $x \epsilon R_m$ then $I\{x \epsilon R_m\} = 1$, μ_m is the average of all elements in the subset (the leaf node R_m)
- It can be visualized as a set of rules dividing the tree into branches and finally leaves
- Decision trees are interpretable until a certain depth, the deeper the tree, the less interpretable it gets (usually trees with depth > 4 are no longer interpretable)
- Feature importance:
  - The features appearing more often in the tree are more important
  - The features are weighted by how much they contribute to the overall reduction in the Gini index compared to the previous node

K-Nearest neighbors

- Lazy learner
- It takes the k closest training data points to this new data point and uses their labels to predict the new label
  - For classification: the mode of the labels
  - For regression: the mean of the labels
- Only a local interpretability because there is no fitted model

Naïve Bayes

- Based on the Bayes Theorem of conditional probability
- Assumption: it treats the probability of each feature impacting the model independently
- Exclusively classifiers
- $\hat{y} = p(y|X) = \ argmax_{C_k} P(y = C_k) \prod_{i=1}^{n} P(x_i|y = C_k)$
- $P(x_i|y = 1) = \frac{1}{\sqrt{2\pi\sigma_i}} e^{\frac{(x_i - \theta_i)^2}{2{\sigma_i}^2}}$
- With θ_i the mean of the feature and ${\sigma_i}^2$ its variance when y=1 → features have different means & variances for every class, which informs the classification
- The mean & variance of features are sufficient to interpret Naïve bayes
- We can use them to compute the conditional probabilities and rank the features by importance on a global & local level

Trade-off performance & interpretability

- High performance → higher complexity in models → less interpretability
- This complexity comes from 3 major sources: non-linearity, non-monoticity, & interactivity
- White box models → transparent models
- However, sometimes it is possible to accept non-transparent but interpretable models → post-hoc interpretability
- Regularization:
  - a good method to reduce complexity because it reduces the number of features→ better interpretability and usually better performance
  - Like lasso & ridge regression, pruning methods in decision trees, dropout in neural networks, etc,

Trade-off performance, interpretability, & excution speed performance

- Linear regression = gold standard for interpretability
- Black-box models are the most performant and least interpretable
- Other characteristics to consider: execution speed (computational time).
- High interpretability & performance comes with a significant loss in execution speed → glass-box models

White Box?	Model Class	Properties that Increase Interpretability					Task		Performance Rank	
		Expl.	Linear	Monotone	Non-Interactive	Regul.	Regr.	Classif.	Regr.	Classif.
✓	Linear Regression	🟢	🟢	🟢	🟢	🟢	✓	✗	6	
✓	Regularized Regression	🟢	🟢	🟢	🟢	🟢	✓	✓	7	8
✓	Logistic Regression	🟢	🟡	🟢	🟢	🟢	✗	✓		5
✓	Gaussian Naïve Bayes	🟢	🔴	🟢	🟢	🟢	✗	✓		7
✓	Polynomial Regression	🟡	🟡	🟢	🟡	🟢	✓	✓	2	
✓	RuleFit	🟢	🟢	🔴	🔴	🟢	✓	✓	8	
✓	Decision Tree	🟢	🔴	🟡	🔴	🟢	✓	✓	5	3
✓	k-Nearest Neighbors	🟡	🔴	🔴	🟢	🔴	✓	✓	9	6
✗	Random Forest	🔴	🔴	🔴	🔴	🟢	✓	✓	3	4
✗	Gradient Boosted Trees	🔴	🔴	🔴	🔴	🟢	✓	✓		2
✗	Multi-layer Perceptron	🔴	🔴	🔴	🔴	🟢	✓	✓	1	1

	White Box	Glass Box	Black Box
Interpretability	High	Mid-High	Low
Predictive Performance	Mid	High	High
Execution Speed Performance	High	Low	Mid

Model-specific interpretability approaches

Feature importance from models' intrinsic parameters

Feature importance for tree based models

- Features importance are computed using a weighted sum of decreases in node impurity
- Node impurity is a metric used to decide how to split a branch
- It tells us how much a node belongs to a class: its value ranges from 100% impure (the split is even) to 0% impure (when it all belongs to one single class)
- The feature importance coming from different tree-based model have different scales → incomparable
- → to compare them, we need to extract the ranks of features & compare accordingly
- Limitation:
    - Impurity-based feature importance methods are biased towards higher cardinality features, i.e., features having a lot of unique values

Feature importance for a logistic regression model

- The coefficients (or the log odds)
- When dealing with a multiclass classification, we have C sets of coefficients with C being the number of classes, each set tells us the most important features to predict this specific class
- If the one-vs-rest (OvR) flag is set:
  - then the model behind the scene has built C binary models to predict independently each of the C classes
  - Finally, it compares the predicted probabilities for each class, for each observation, & chooses the class with the highest probability
  - NB: if features are not normalized, it is recommended to multiply them by their standard deviations to **approximate** feature importance
  - There is no consensus to the best method of feature importance computation for logistic regression, thus, the term approximation
  - We can weight the feature importance with the priors (the % of representation of each class in the dataset)

Feature importance of the Linear Discriminant Analysis (LDA)

- Assumptions: normality & homoscedasticity
- It is a dimensionality reduction technique
- It computes the distance between the mean of different classes, the between-class variance, and the variance within each class, the within-class variance
- Then, it projects the data to a lower-dimensional space while:
  - Minimizing the distance within classes
  - Maximizing the distances between classes
- It is used for classification, dimensionality reduction, & visualization of class separation
- Feature importance = coefficients; they refer to how much each feature weights in the separability of the class.
  - The higher the value of the coefficient, the more the feature assists in separating the class
  - Unlike PCA, it decomposes features into separateness & not corrolatedness
  - If data is not normalized, the coefficients need to be multiplied by the features' standard deviation and can be weighted by the class priors
- Limitations: No multicollinearity assumption between features & multivariate normality
- Advantage: more robust to assumption violation than logistic regression → better when dealing with noisy data + QDA or quadratic discriminant analysis (no normality assumption, classes split by a quadratic decision boundary instead of linear)

Feature importance for neural networks

There is no consensus on how to extract feature importance from the intrinsic parameters of neural networks

Model agnostic interpretability methods

Model agnostic interpretability methods

- Permutation Feature importance (PFI)
- Partial dependence plots (PDP)
- Individual conditional expectation (ICE)

Permutation Feature importance (PFI)

- The theory for PFI: if the feature has a relationship with the target variable, shuffling will disrupt it & increase the error. On the other side, if the feature does not have a strong relationship with the target variable, the prediction error won't increase by much, if at all.
- It ranks features intuitively & dependently
- Feature importance: ranking the features by those whose shuffling increases the error the most
- It can be used on the training dataset: the ranking will represent what the model thinks is important according to what was learnt from the training dataset
- It can be used with the test dataset
- Limitation: multicollinearity, if existent, will upstage feature importance. In other terms, when one feature is shuffled, its correlated features will remain unshuffled, keeping error rates relatively unaltered, which means clusters of correlated features will have lower importance values than they should

Partial dependence plots (PDP)

- A PDP carries the marginal effect of a feature on the prediction throughout all possible values for that feature
- It can demonstrate visually the feature's impact & the nature of the relationship with the target (linear, exponential, monotonic, etc.)
- A PDP examines the marginal impact of the most important feature to the outcome, showing which feature values correlate the most with the predictions
- It can be extended to 2 features; it illustrates the effect of their interaction on the model
- One feature plot: y axis = predicted outcome, x axis= all possible values of the feature, the plotted line is computed by changing the value of the feature to the one value in the x axis for all the observations, predicting the targets, & averaging these predictions to get the y coordinate
- Limitations:
    - The display is limited up to 2 features at a time
    - Features are assumed independent when there might be a correlation between each others→ solution is Accumulated Local Effect (ALE) plots

Partial dependence plots (PDP)

- A PDP carries the marginal effect of a feature on the prediction throughout all possible values for that feature
- It can demonstrate visually the feature's impact & the nature of the relationship with the target (linear, exponential, monotonic, etc.)
- A PDP examines the marginal impact of the most important feature to the outcome, showing which feature values correlate the most with the predictions
- It can be extended to 2 features; it illustrates the effect of their interaction on the model
- One feature plot: y axis = predicted outcome, x axis= all possible values of the feature, the plotted line is computed by changing the value of the feature to the one value in the x axis for all the observations, predicting the targets, & averaging these predictions to get the y coordinate
- Limitations:
    - The display is limited up to 2 features at a time
    - Features are assumed independent when there might be a correlation between each others→ solution is Accumulated Local Effect (ALE) plots

Individual conditional expectation (ICE)

- ICE plots answer the question: what if my PDP plots obscure the variance my feature-target relationships?
  - PDP plots average predictions → with averaging, a lot of interesting information is lost
  - Looking at the PDP plots for individual features, we notice many thin lines that are not only distant of the average thick line but also not following its general direction
  - These variations can provide insightful information
  - These lines are essentially what ICE plots are about
  - They excel at looking for clues in the variation of the feature-target relationship & not on its aggregate(e.g., average)
- Limitations:
  - Assumption of features independency
  - Interactions with continuous or high-cardinality features is unfeasible
  - It is hard to ascertain the average relationship between a feature & a target, but this is the PDP's job

Shapley Addititive exPlanations (SHAP)

Desired characteristics of explainers

- Explanation must be interpretable taking into account the user's limitations
  - Not too bulky: not too many features to look at
  - Understandable: the input variables in the explanation may need to be different from the features used by the model
- Local fidelity: the explanation must correspond to how the model behaves in the vicinity of the instance being predicted

Shapley Additive exPlanations (SHAP)

- SHAP through a basketball analogy
- Coalitional or cooperative game theory:
  - The different combinations of players are coalitions
  - The differences in scores are marginal contributions
  - The **Shapley value** is the average of these contributions over many simulations
- For a model:
  - The players are the features
  - The coalitions of players are the different subset of features
  - The marginal contributions are the differences in predictive error
  - The blindfolded analogy refers to the model being a black box

Shapley Additive exPlanations (SHAP)

- Shapley's algorithm calculates the features' expected value over the entire dataset
- This value would be the best guess of the value of a feature
- It is not perfect but is enough to compare the contributions of the feature
- Shapley values have several properties derived from the coalitional game theory:
  - Dummy: if a feature i never contributes any marginal value → $Shapley_i = 0$
  - Substitutability: if 2 given features i & j contribute equally to all their possible subsets → $Shapley_i = Shapley_j$
  - Additivity: if a model p is an ensemble of k sub-models, the contributions of a feature i in the sub-models should add up → $Shapley_i^p = \sum_{n=1}^{k} Shapley_i^n$
  - Efficiency: all Shapley values must add up as the difference between predictions & expected values

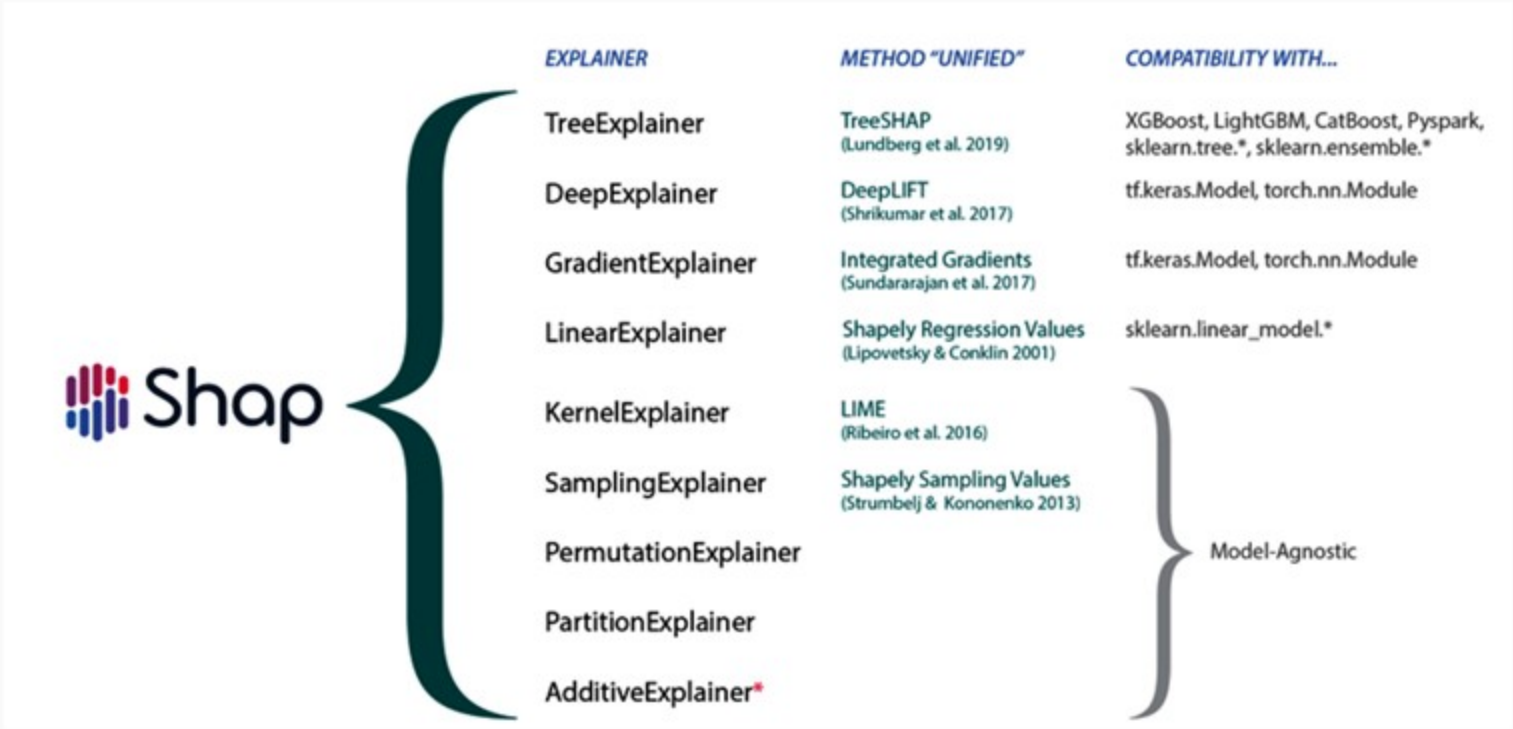
Shapley regression values for linear models in the presence of multicollinearity

- It is an additive feature attribution method
- This method $g(z')$ attributes an effect $\phi_i \epsilon \mathbb{R}$ to each feature such that summing the effects of all the features attributions approximates the output $f(x)$ of the original method
- $g(z') = f(h_x(z'))whenever\ z' \simeq x'$
- $x' = a\ simplified\ input\ such\ that\ x = h_x(x')$
- with $h\ is\ a\ mapping\ from\ the\ simplified\ input\ to\ its\ original\ version$
- $g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z_i'$, with $z' \epsilon \{0,1\}^M$ & $M = nbr\ of\ features$
- $\phi_i = \sum_{S \subseteq F \backslash \{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!} [f_{S \cup \{i\}}(x_{S \cup \{i\}}) - f_S(x_S)]$
- $F = the\ set\ of\ all\ features\ (columns\ in\ the\ dataset), |F| = nbr\ of\ columns/features$
- $S = all\ the\ features\ subsets \rightarrow S \subseteq F, |S| = total\ nbr\ of\ subsets$
- $S \subseteq F \backslash \{i\} = all\ the\ features\ subsets\ without\ the\ feature\ i$
- $f_{S \cup \{i\}}(x_{S \cup \{i\}}) = trained\ model\ with\ the\ features\ in\ the\ subset\ S\ and\ the\ evaluated\ feature\ i$
- $f_S(x_S) = trained\ model\ with\ the\ features\ in\ the\ subset\ S\ without\ the\ evaluated\ feature\ i$
- h_x maps 1 to the input existent and 0 to the input missing from the model
- Computational costs: $2^{|F|} differences\ to\ compute\ \&\ C_M^1 + C_M^2 + \cdots + C_M^M\ models\ to\ train$

Shapley Additive exPlanations (SHAP) - Implementation

- SHAP is a collection of methods or explainers that approximate Shapley values while approximately adhering to its mathematical properties
- It has 3 properties:
  - Local accuracy:
    - like Shapley's efficiency property, the explanation model $g(x')$ must approximately match the original model's output $f(x)$ when $x = h_x(x')$ and $\phi_0 = f(h_x(0))$ representing the model's output when all simplified inputs are missing
    - $f(x) = g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x_i'$
  - Missingness:
    - if a feature is missing, its Shapley value is 0; this feature is needed for constant features in practice
    - $x_i' = 0 \Rightarrow \phi_i = 0$
  - Consistency: additivity & substitutivity axioms, and dummy in theory

SHAP library



Accumulated Local Effects (ALE)

- ALEs are like PDPs (Partial Dependence Plots) but unbiased & faster.
- Unbiased = they do not have the assumption of Collinearity, i.e., the uncorrelation between features
- PDPs make averages of predictions across all feature values regardless of whether they make sense while assuming independence of the features
- However, ALEs take the approach of factoring data distributions when computing the effects of a feature
    - They split the features into equally sized intervals (quantiles)
    - They compute how much the predictions change, on average, in each of these intervals; hence the word local in ALEs
    - They sum these effects across all intervals; hence the cord accumulated in ALEs
- The effects are relative to an average → they are zero centered

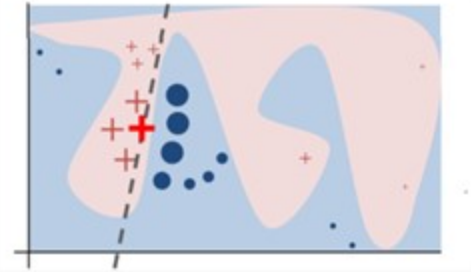
Local Interpretable Model-agnostic Explanations (LIME)

Local Interpretable Model-agnostic Explanations (LIME)

- **Reminder:** there is a difference between the features used by the model & the interpretable data representations; the latter must be understandable to humans
- We denote $x \epsilon \mathbb{R}^d$: $the\ original\ data\ representation\ of\ an\ instance\ being\ explained$
- $x' \epsilon \{0,1\}^{d'}$: $the\ binary\ vector\ of\ its\ interpretable\ representation$
- g : $the\ explanation\ model \epsilon G$;
- $G\ is\ the\ set\ of\ potentially\ interpretable\ models\ such\ as\ linear\ models, trees, rule\ lists, etc.$
- The domain of g is $\{0,1\}^{d'}$ => g acts over absence/presence of the interpretable components
- Since not every g ϵ G is enough interpretable, we define $\Omega(g)$ to be the complexity of the explanation model g,
  - for a decision tree, $\Omega(g)$ could be the depth of the tree
  - For a linear model, $\Omega(g)$ could be the number of non-zero weights
- Let the model being explained be $f : \mathbb{R}^d \rightarrow \mathbb{R}$; in classification, $f(x)$ is the probability that x belongs to a certain class (or a binary indicator)
- And let $\pi_x(z)$ be the proximity measure between an instance z to x, as to define locality around x
- Finally, let $L(f, g, \pi_x)$ be a measure of how unfaithful g is in approximating f in the locality defined by π_x
- To ensure fidelity & local interpretability, we minimize $L(f, g, \pi_x)$ while having a low $\Omega(g)$ to ensure interpretability: $\xi(x) = argmin_{g \epsilon G} L(f, g, \pi_x) + \Omega(g)$

Local Interpretable Model-agnostic Explanations (LIME)

- We sample instances around x' by sampling non-zero elements of x'
- Given a sample $z' \epsilon \{0,1\}^{d'}$; z' contains a fraction of the non-zero elements of x', the sample $z \epsilon \mathbb{R}^d$ in the original data representations is recovered, then we compute $f(z)$, which is used as a label for the explanation model
- Given a dataset Z of samples z' with their associated labels, we optimize the equation $argmin_{g \epsilon G} L(f, g, \pi_x) + \Omega(g)$ & get the explanation $\xi(x)$



- Suppose we have a complex model f like the one presented in the figure by the blue/pink background
- This model is hardly approximated by a linear model
- The bold red cross is the instance we would like to explain
- What LIME does is sample in the vicinity of x and far away from x, such that samples in the vicinity get high weights and far away samples get low weights, and this is thanks to π_x; the weights are represented here by the size of the cross and dot
- It computes the predictions using the complex model f
- The dashed line is the learned explanation that is locally faithful, NOT globally
- Even when the original model is too complex to explain, LIME succeeds on producing a locally faithful explanation (linear in this case)
- It is important to note that it is fairly robust to sampling noise thanks to the weighting mechanism

Local Interpretable Model-agnostic Explanations (LIME)

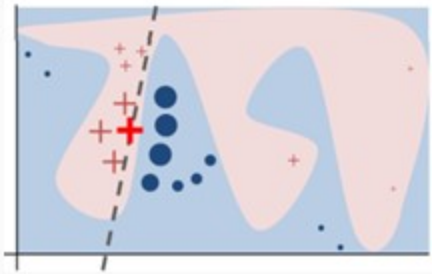
- Let G be the class of linear models, such that $g(z') = w_g . z'$, L the square loss, and $\Pi_x = \exp(-\frac{D(x,z)^2}{\sigma^2})$ the weighting exponential kernel with the distance function D (cosine distance for text, L_2 distance for images & tabular data) and width = σ
- $L(f, g, \pi_x) = \sum_{z,z' \in Z} \pi_x(z).(f(z) - g(z'))^2$
- For text classification, we only limit the words to K words for interpretability reasons
- For image classification, we only select the pixels that did influence the prediction
- Thus, the equation is approximated using K-LASSO

Algorithm 1 Sparse Linear Explanations using LIME

Require: Classifier f, Number of samples N
Require: Instance x, and its interpretable version x'
Require: Similarity kernel π_x, Length of explanation K
  $\mathcal{Z} \leftarrow \{\}$
  for $i \in \{1, 2, 3, ..., N\}$ **do**
    $z'_i \leftarrow sample\_around(x')$
    $\mathcal{Z} \leftarrow \mathcal{Z} \cup \langle z'_i, f(z_i), \pi_x(z_i) \rangle$
  end for
  $w \leftarrow$ K-Lasso$(\mathcal{Z}, K)$ ▷ with z'_i as features, $f(z)$ as target
  return w

Local Interpretable Model-agnostic Explanations (LIME)

- LIME trains local surrogates to explain a single prediction
- Suppose we have a complex model f like the one presented in the figure by the blue/pink background
- This model is hardly approximated by a linear model
- The **bold red cross** is the instance we would like to explain
- What LIME does is:
    1. Sample **in the vicinity of** x and far away from x, such that samples in the vicinity get high weights and far away samples get low weights, and this is thanks to; the weights are represented here by the size of the cross and dot. π_x an exponential kernel of width σ and distance function D, with D being the cosine distance for text and the L_2 distance for images & tabular data
    2. Compute the predictions using the complex model f
    3. Train a weighted intrinsically interpretable model (a sparse linear model with weighted regularization) using the sampled points & the black-box predictions in this neighborhood
    4. Finally interpret this surrogate model



The dashed line is the learned explanation that is locally faithful, NOT globally
Even when the original model is too complex to explain, LIME succeeds on producing a locally faithful explanation (linear in this case)
It is important to note that: it is fairly robust to sampling noise thanks to the weighting mechanism & this faithfulness is contingent on the neighborhood being defined correctly → choosing the right kernel & the assumption of local linearity holding true

SHAP vs LIME

SHAP	LIME
Grounded on game theory	Not grounded on consistent principles
A consistent method due to properties such as additivity, efficiency, & sustainability; the dummy property is violated	Its consistency is limited to the intuition that neighbors are alike
No need for parameter tuning	The number of neighbors to be selected is a tricky parameter to tune
More suited for global interpretation	Only suited for local interpretations
Its general-purpose model-agnostic method, KernelExplainer, is slow	speedy
SHAP explainers work with tabular data and DeepExplainer can do text but is restricted to deep learning models	Adaptable to all kinds of data

References

Masís, Serg. Interpretable machine learning with Python: Learn to build interpretable high-performance models with hands-on real-world examples. Packt Publishing Ltd, 2021.

Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "" Why should i trust you?" Explaining the predictions of any classifier." Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016.

Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).