ML Modeling Pipelines in Production

Part 5: Interpretability

Ramin Toosi

Responsible AI

- Development of AI is creating new opportunities to improve lives of people
- Also raises new questions about the best way to build the following into AI systems:

Fairness

- Ensure working towards systems that are fair and inclusive to all users.
- Explainability helps ensure fairness.

Explainability

- Understanding how and why ML models make certain predictions.
- Explainability helps ensure fairness.

Privacy

- Training models using sensitive data needs privacy preserving safeguards.

Security

- Identifying potential threats can help keep AI systems safe and secure.

Explainable Artificial Intelligence (XAI)

- The field of XAI allow ML system to be more transparent, providing explanations of their decisions in some level of detail.
- These explanations are important:
  - To ensure algorithmic fairness.
  - Identify potential bias and problems in training data.
  - To ensure algorithms/models work as expected.

Need for Explainability in AI

- Models with high sensitivity, including natural language networks, can generate wildly wrong results
- Attacks
- Fairness
- Reputation and Branding
- Legal and regulatory concerns
- Customers and other stakeholders may question or challenge model decisions

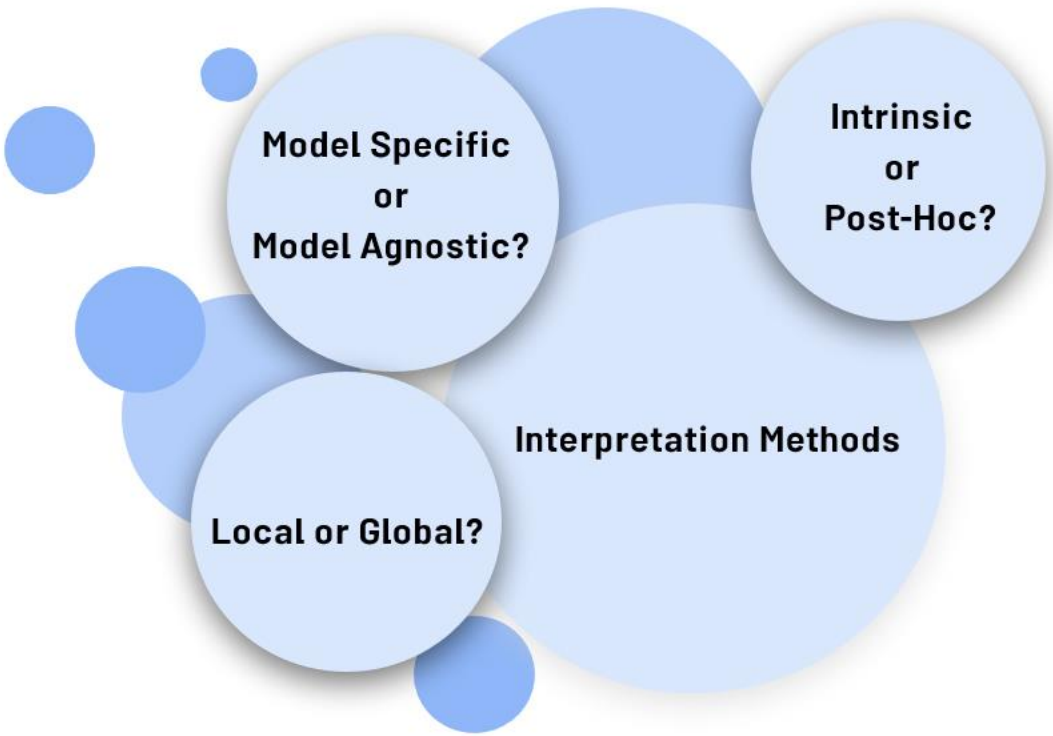
What is interpretability?

- "(Models) are interpretable if their operations can be understood by a human, either through introspection or through a produced explanation."
  - "Explanation and justification in machine learning: A survey" - O. Biran, C. Cotton

What are the requirements?

- **You should be able to query the model to understand:**
  - **Why did the model behave in a certain way?**
  - **How can we trust the predictions made by the model?**
  - **What information can model provide to avoid prediction errors?**

Categorizing Model Interpretation Methods



Types of results produced by Interpretation Methods

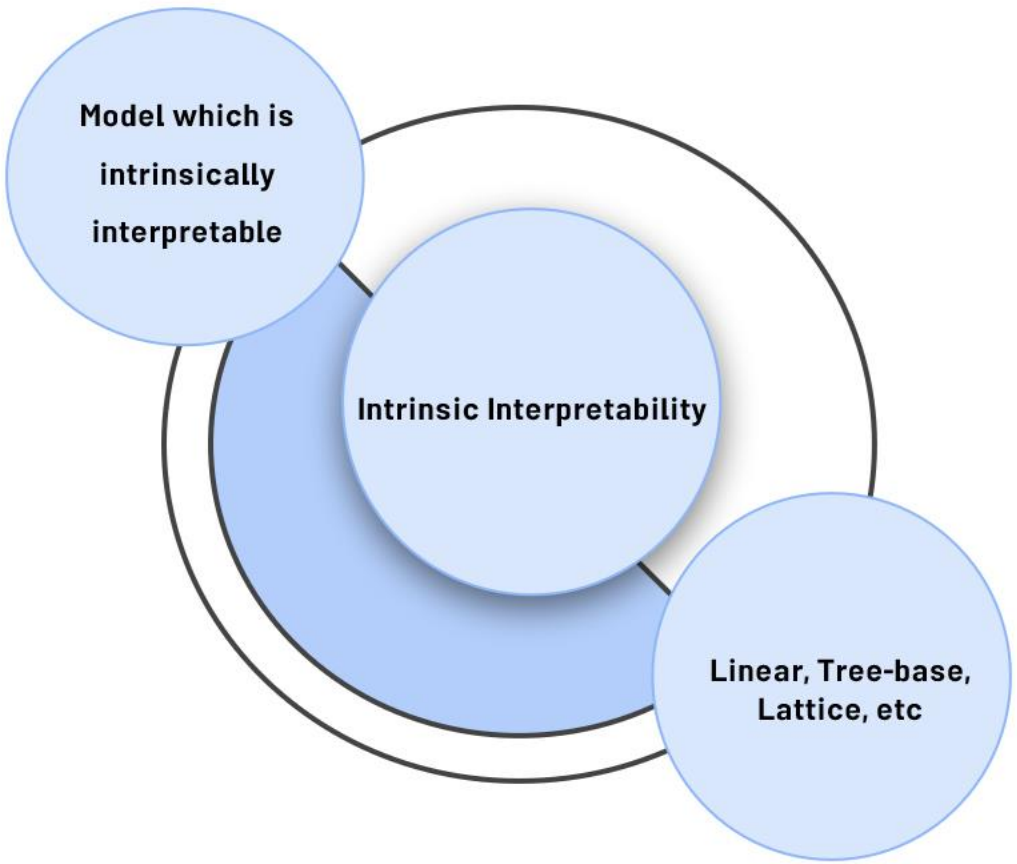
Feature Summary Statistics

Feature Summary Visualization

Model Internals

Data point

Intrinsic or Post-Hoc?



Intrinsic or Post-Hoc?

- Post-hoc methods treat models as black boxes
- Agnostic to model architecture
- Extracts relationships between features and model predictions, agnostic of model architecture
- Applied after training

Model Specific or Model Agnostic

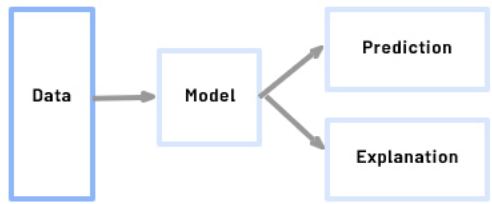
Model Specific

These tools are limited to specific model classes

Example: Interpretation of regression weights in linear models

Intrinsically interpretable model techniques are model specific

Tools designed for particular model architectures

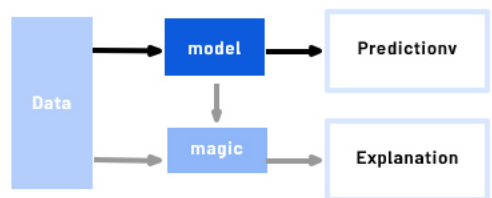


Model Agnostic

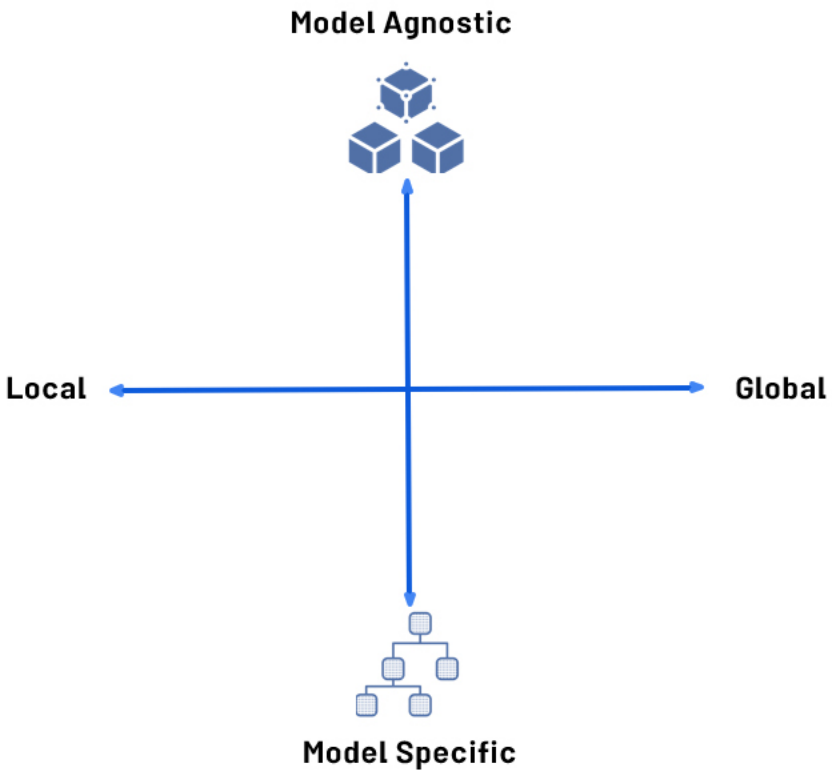
Applied to any model after it is trained

Do not have access to the internals of the model

Work by analyzing feature input and output pairs

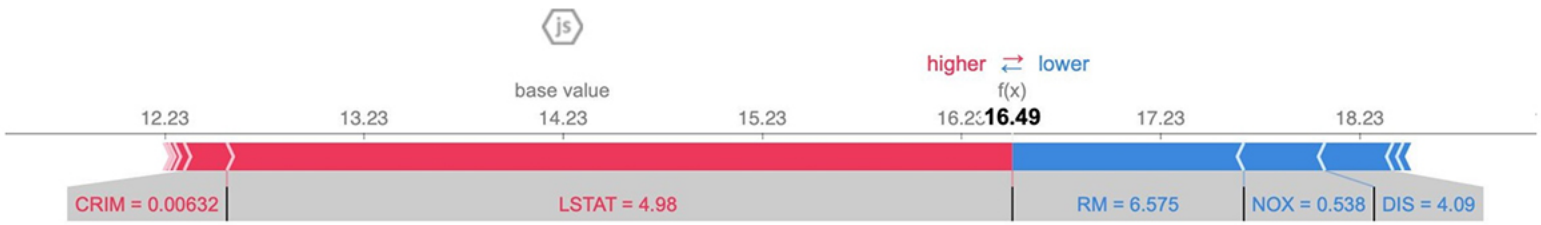


Interpretability of ML Models



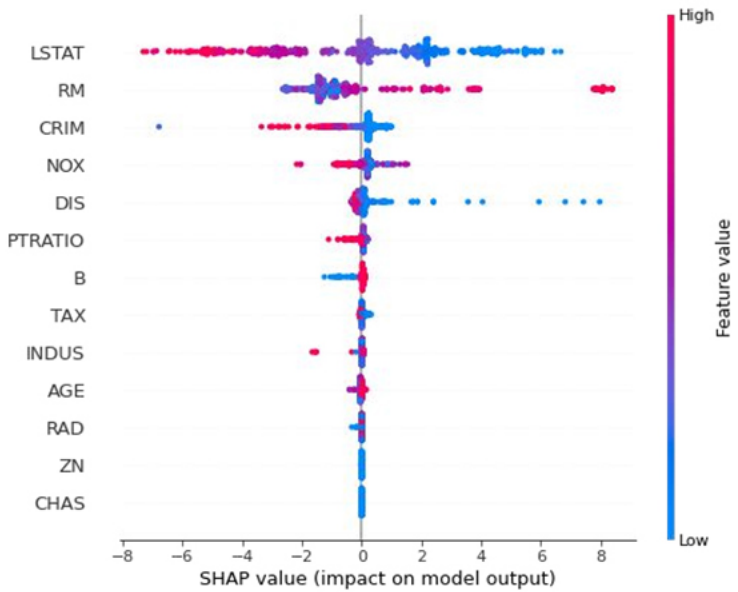
Local or Global?

- Local: interpretation method explains an individual prediction.
- Feature attribution is identification of relevant features as an explanation for a model.

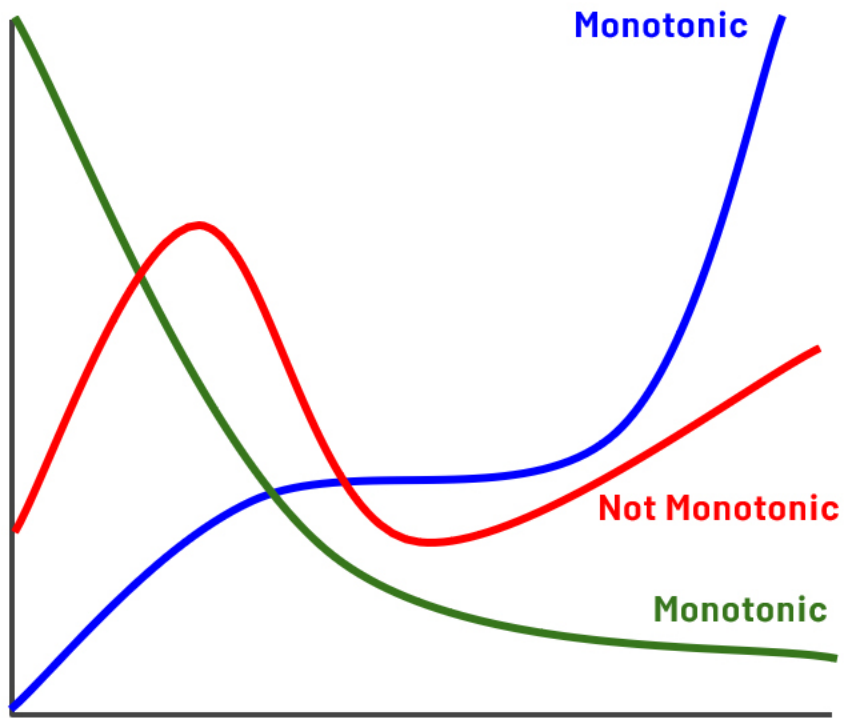


Local or Global?

- Global: interpretation method explains entire model behaviour
- Feature attribution summary for the entire test data set



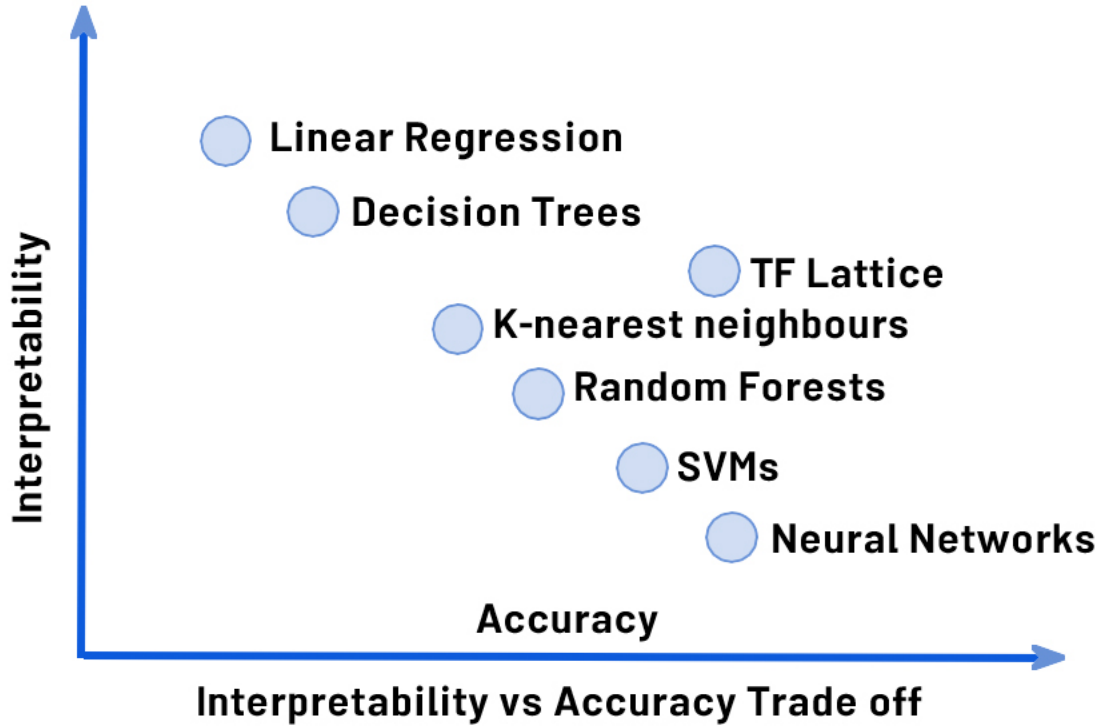
Monotonicity improves interpretability



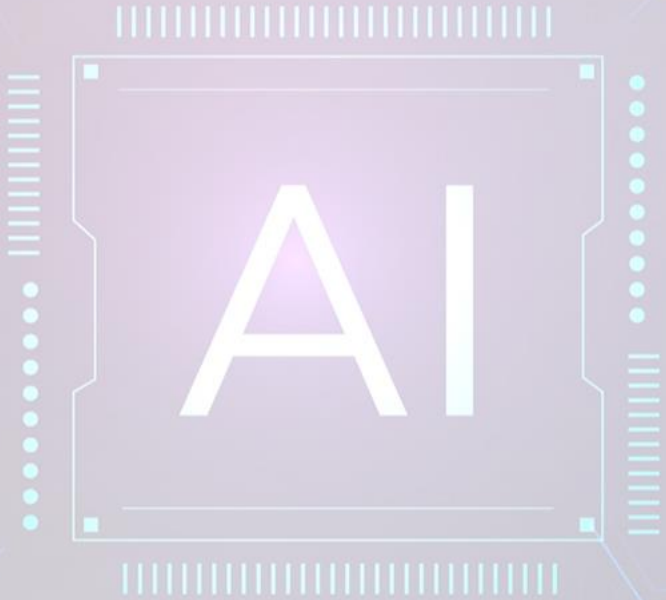
Interpretable Models

Algorithm	Linear	Monotonic	Feature Interaction	Task
Linear regression	Yes	Yes	No	regr
Logistic regression	No	Yes	No	class
Decision trees	No	Some	Yes	class, regr
RuleFit	Yes*	No	Yes	class, regr
K-nearest neighbors	No	No	No	class, regr
TF Lattice	Yes*	Yes	Yes	class, regr

Model Architecture Influence on Interpretability



Model Agnostic Methods



Model Agnostic Methods

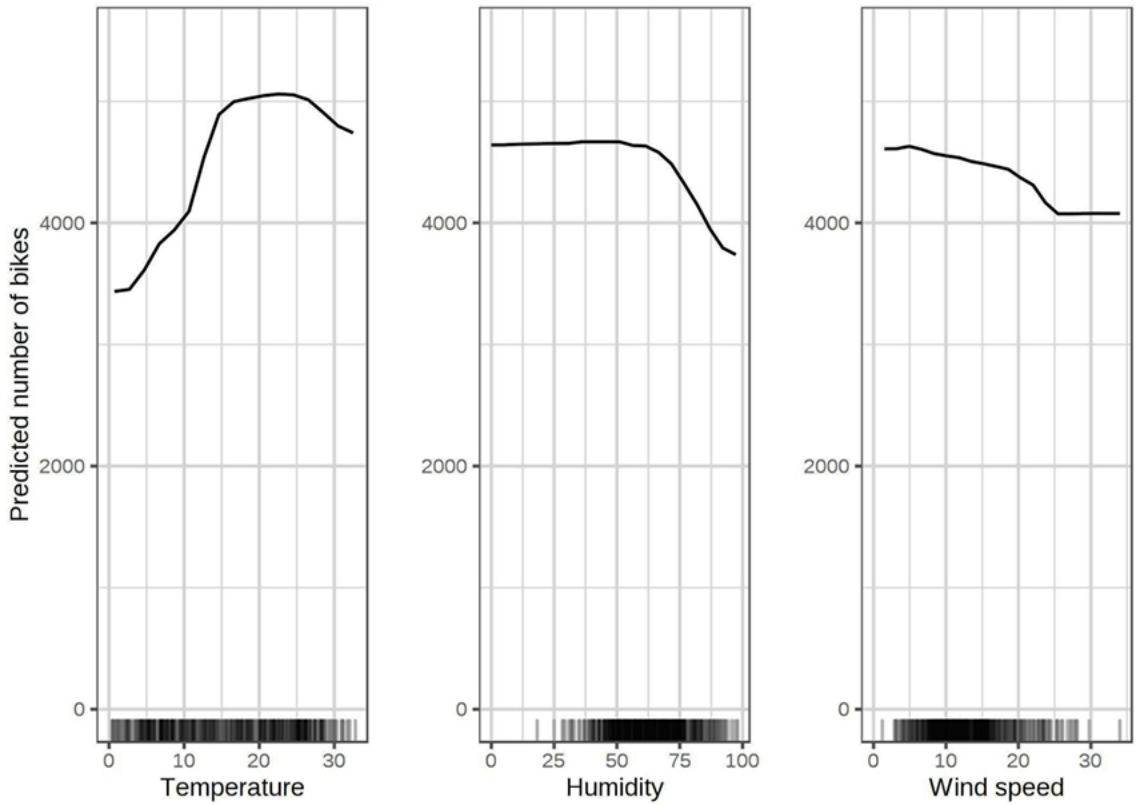
- Partial Dependence Plots
- Accumulated Local Effects
- Permutation Feature Importance
- Local Surrogate (LIME)
- SHAP

Partial Dependence Plots (PDP)

- **A partial dependence plot shows:**
  - The marginal effect one or two features have on the model result
  - Whether the relationship between the targets and the feature is linear, monotonic, or more complex

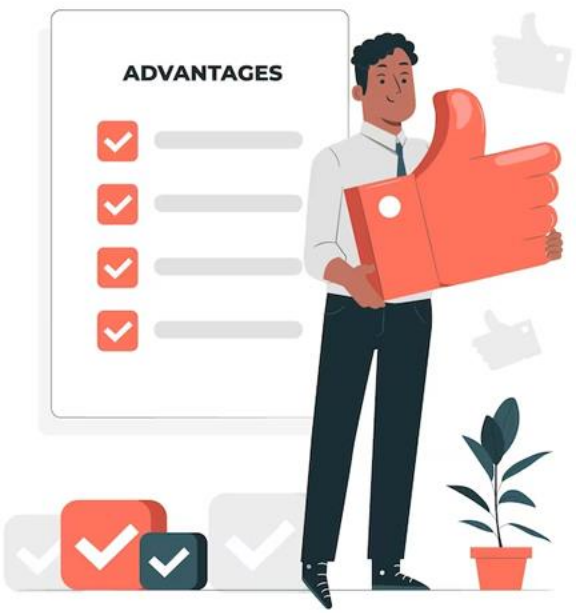
Partial Dependence Plots: Examples

- PDP plots for a linear regression model trained on a bike rentals dataset to predict the number of bikes rented



Advantages of PDP

- Computation is intuitive
- If the feature whose PDP is calculated has no feature correlations, PDP perfectly represents how feature influences the prediction on average
- Easy to implement



Disadvantages of PDP

- Realistic maximum number of features in PDP is 2
- PDP assumes that feature values have no interactions

Permutation Feature Importance

- Feature importance measures the increase in prediction error after permuting the features
- Feature is important if:
  - Shuffling its values increases model error
- Feature is unimportant if:
  - Shuffling its values leaves model error unchanged

Permutation Feature Importance

- Estimate the original model error
- For each feature:
  - Permute the feature values in the data to break its association with the true outcome
  - Estimate error based on the predictions of the permuted data
  - Calculate permutation feature importance
  - Sort features by descending feature importance

Example

```
>>> from sklearn.inspection import permutation_importance
>>> r = permutation_importance(model, X_val, y_val,
...                            n_repeats=30,
...                            random_state=0)
...
>>> for i in r.importances_mean.argsort()[::-1]:
...     if r.importances_mean[i] - 2 * r.importances_std[i] > 0:
...         print(f"{diabetes.feature_names[i]:<8}"
...               f"{r.importances_mean[i]:.3f}"
...               f" +/- {r.importances_std[i]:.3f}")
```

```
s5      0.204 +/- 0.050
bmi     0.176 +/- 0.048
bp      0.088 +/- 0.033
sex     0.056 +/- 0.023
```

Advantages of Permutation Feature Importance

- Nice interpretation: Shows the increase in model error when the feature's information is destroyed.
- Provides global insight to model's behaviour
- Does not require retraining of model

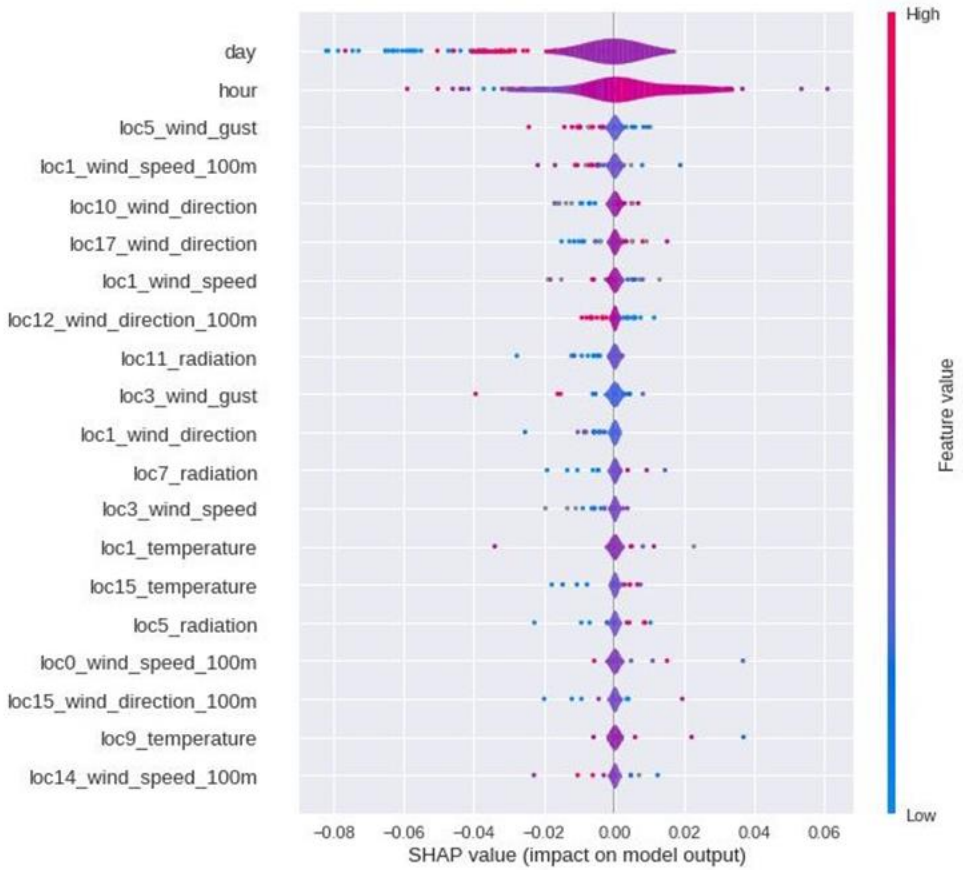
SHAP

- SHAP (SHapley Additive exPlanations) is a framework for Shapley Values which assigns each feature an importance value for a particular prediction
- Includes extensions for:
  - TreeExplainer: high-speed exact algorithm for tree ensembles
  - DeepExplainer: high-speed approximation algorithm for SHAP values in deep learning models
  - GradientExplainer: combines ideas from Integrated Gradients, SHAP, and SmoothGrad into a single expected value equation
  - KernelExplainer: uses a specially-weighted local linear regression to estimate SHAP values for any model

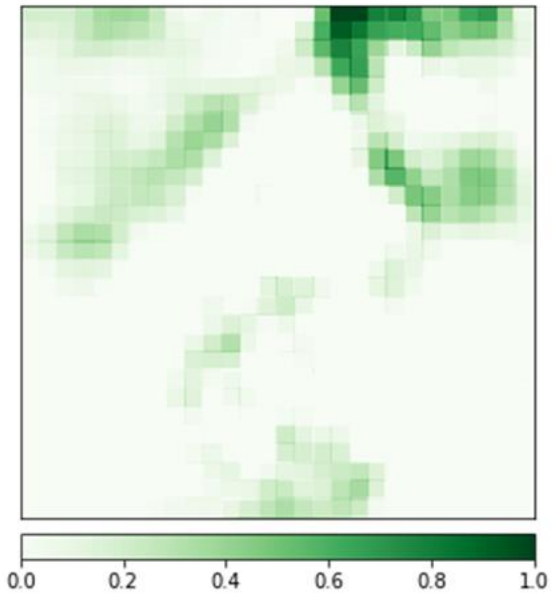
SHAP Explanation Force Plots

- Shapley Values can be visualized as forces
- Prediction starts from the baseline (Average of all predictions)
- Each feature value is a force that increases (red) or decreases (blue) the prediction

SHAP Summary Plot



Captum: Model Interpretability for PyTorch



ResNet 50

Goose (0.45)