COMP 4432 Machine Learning

Lesson 2

Agenda

- The Data Science Project Process
  - Define the problem
  - Gather and get to know the data
  - Prepare data
  - Select and train a model
  - Tune the model
  - Productionalize
  - Interpret and explain the model

Define the Problem

- Translate the business problem into a technical problem
  - Almost never perfectly defined, so ask questions

Define the Problem

- Translate the business problem into a technical problem
  - Almost never perfectly defined, so ask questions
- Clearly define expectations
  - Is there another method currently in place?
    - If so, that is the benchmark. You must do better.

Define the Problem

- Translate the business problem into a technical problem
  - Almost never perfectly defined, so ask questions
- Clearly define expectations
  - Is there another method currently in place?
    - If so, that is the benchmark.
  - Always someone wanting perfection
    - “So-and-so is getting 99.9% accuracy…”

Define the Problem

- Translate the business problem into a technical problem
  - Almost never perfectly defined, so ask questions
- Clearly define expectations
  - Is there another method currently in place?
    - If so, that is the benchmark.
  - Always someone wanting perfection
    - “So-and-so is getting 99.9% accuracy…”
- Identify the potential methods and metrics

Gather and explore data

- Identify data sources and owners
- Descriptive statistics
  - df.describe().T
- Correlation Analysis
  - Code examples
- Visualizations
  - Histograms (Smaller feature sets)
  - First examination of skewness

Correlation Analysis

- Measures the strength and direction of the relationship between two variables
- Correlated features may not harm performance, but they won’t add extra information to the model, and will increase complexity.

Correlation Analysis

- Calculating correlation between two continuous variables

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

- Be careful when comparing continuous and categorical variables.
    - Cramer's V

Correlation Analysis

Correlation Analysis

	variable1	variable2	correlation
121	RAD	TAX	0.910228
32	INDUS	NOX	0.763651
62	NOX	AGE	0.731470
37	INDUS	TAX	0.720760
83	RM	PRICE	0.695360
...	...	...	...
82	RM	LSTAT	-0.613808
35	INDUS	DIS	-0.708027
181	LSTAT	PRICE	-0.737663
91	AGE	DIS	-0.747881
63	NOX	DIS	-0.769230

```
corr.loc[corr[corr.correlation.abs() > 0.7].index]
```

	variable1	variable2	correlation
121	RAD	TAX	0.910228
32	INDUS	NOX	0.763651
62	NOX	AGE	0.731470
37	INDUS	TAX	0.720760
35	INDUS	DIS	-0.708027
181	LSTAT	PRICE	-0.737663
91	AGE	DIS	-0.747881
63	NOX	DIS	-0.769230

Prepare data

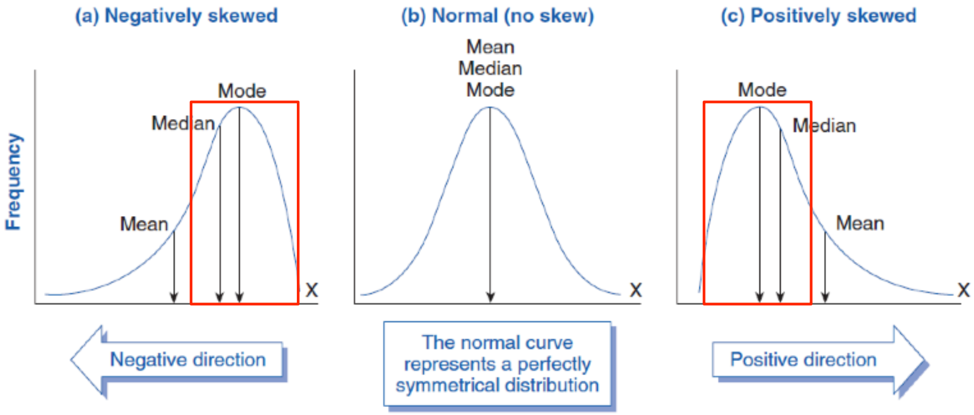
- Handle missing data
    - Imputation
        - Many methods to consider
            - Mean / Median replacement
            - Decision tree using feature with missing features as target
    - Removal

Prepare data

- Handle missing data
  - Imputation
    - Many methods to consider
      - Mean / Median replacement
      - Decision tree using feature with missing features as target
  - Removal
- Transforming
  - Skewed features

Transforming

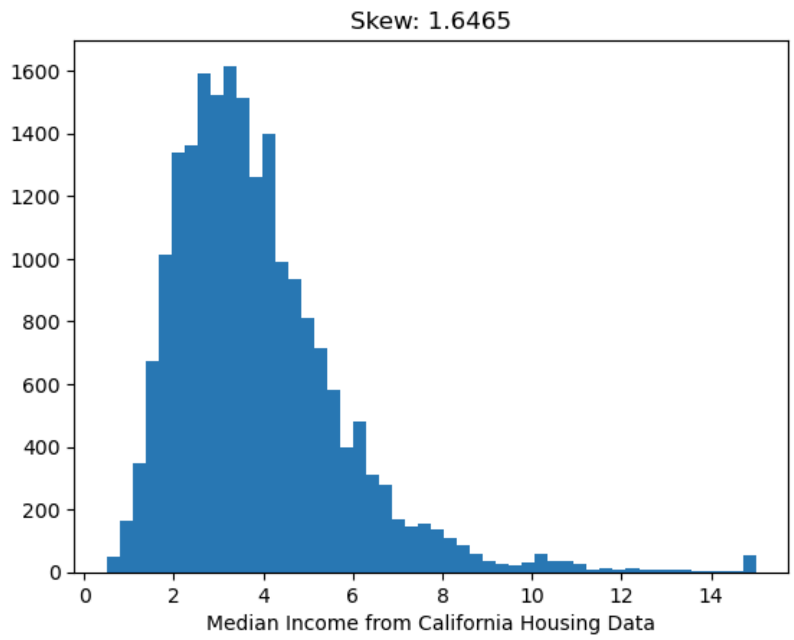
- Skewness (Measure of asymmetry)

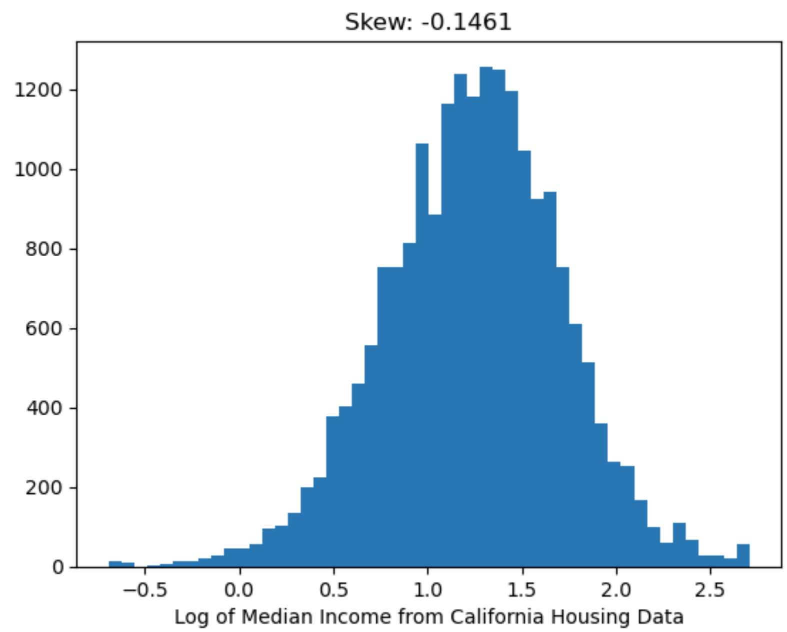


Model will perform better with the more abundant, values than with the rarer

- Reduces the model's predictability of the model to describe common cases because of extreme values in the tail
- Not a problem with decision trees

Transforming





Prepare data

- Handle missing data
  - Imputation
    - Many methods to consider
      - Mean / Median replacement
      - Decision tree using feature with missing features as target
  - Removal
- Transforming
  - Skewed features
- Scaling
  - Always fit to training data and use fitted scaler to transform validation and testing data

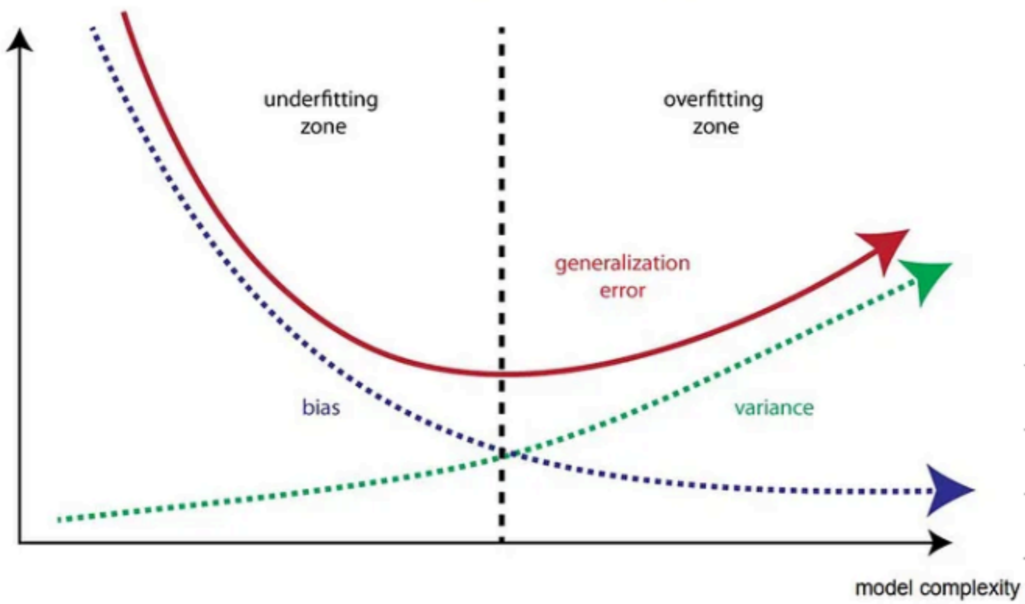
Scaling

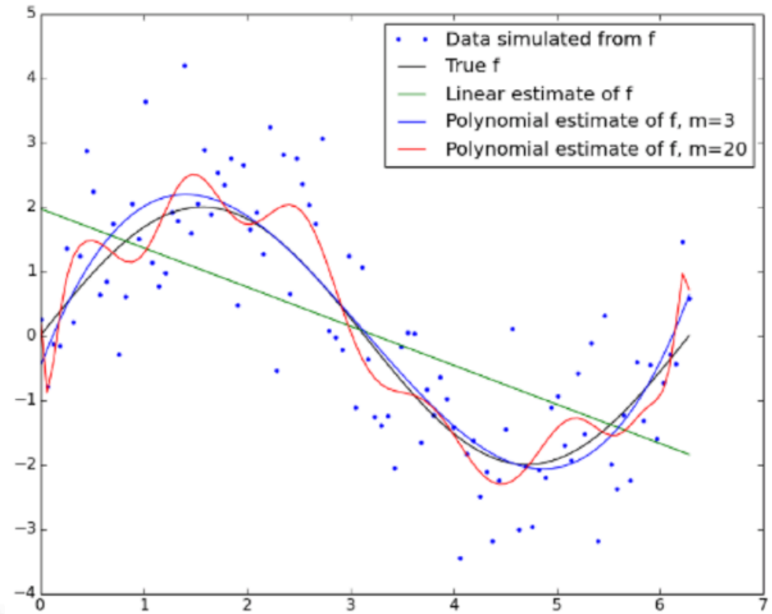
- Normalization
- Min-Max Scaling
- Small entropy in column ?

Select and Train Model

- More than one algorithm to consider
- Champion-Challenger approach
- Metrics to employ in decision
- Exercise caution with complexity
  - Models need to perform well on unseen (test) and future (out-of-time) data
  - Trade off between performance and explainability / interpretability

Bias-Variance Tradeoff





Overfitting

- Fits the training data too closely and gives very accurate predictions to only training data
- Model does not generalize to new data
- Reducible
  - Regularization, Early Stopping

Decision Tree Regressor

- By default, will construct if-then-else logic to explain each instance in training data

splitter : *{"best", "random"}, default="best"*

The strategy used to choose the split at each node. Supported strategies are "best" to choose the best split and "random" to choose the best random split.

max_depth : *int, default=None*

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

min_samples_split : *int or float, default=2*

The minimum number of samples required to split an internal node:

- If int, then consider `min_samples_split` as the minimum number.
- If float, then `min_samples_split` is a fraction and `ceil(min_samples_split * n_samples)` are the minimum number of samples for each split.

Changed in version 0.18: Added float values for fractions.

min_samples_leaf : *int or float, default=1*

The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least `min_samples_leaf` training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

- If int, then consider `min_samples_leaf` as the minimum number.
- If float, then `min_samples_leaf` is a fraction and `ceil(min_samples_leaf * n_samples)` are the minimum number of samples for each node.

Decision Tree Regressor

$$J(k, t_k) = \frac{m_{\text{left}}}{m}\text{MSE}_{\text{left}} + \frac{m_{\text{right}}}{m}\text{MSE}_{\text{right}} \quad \text{where} \begin{cases} \text{MSE}_{\text{node}} = \sum_{i \in \text{node}} \left(\hat{y}_{\text{node}} - y^{(i)}\right)^2 \\ \hat{y}_{\text{node}} = \frac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)} \end{cases}$$

Data is sorted by feature

Calculate Mean as Cut Point

RM	PRICE
3.561	27.5
4.138	11.9
4.368	8.8
4.519	7.0
4.628	17.9
...	...
8.375	50.0
8.398	48.8
8.704	50.0
8.725	50.0
8.780	21.9

Calculate Mean Squared Error for each set

Decision Tree Regressor

$$J(k, t_k) = \frac{m_{\text{left}}}{m}\text{MSE}_{\text{left}} + \frac{m_{\text{right}}}{m}\text{MSE}_{\text{right}} \quad \text{where} \begin{cases} \text{MSE}_{\text{node}} = \sum_{i \in \text{node}} \left(\hat{y}_{\text{node}} - y^{(i)}\right)^2 \\ \hat{y}_{\text{node}} = \frac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)} \end{cases}$$

RM	PRICE
3.561	27.5
4.138	11.9
4.368	8.8
4.519	7.0
4.628	17.9
...	...
8.375	50.0
8.398	48.8
8.704	50.0
8.725	50.0
8.780	21.9

Calculate Mean as Cut Point

Calculate Mean Squared Error for each set

Decision Tree Regressor

$$J(k, t_k) = \frac{m_{\text{left}}}{m}\text{MSE}_{\text{left}} + \frac{m_{\text{right}}}{m}\text{MSE}_{\text{right}} \quad \text{where} \begin{cases} \text{MSE}_{\text{node}} = \sum_{i \in \text{node}} \left(\hat{y}_{\text{node}} - y^{(i)}\right)^2 \\ \hat{y}_{\text{node}} = \frac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)} \end{cases}$$

RM	PRICE
3.561	27.5
4.138	11.9
4.368	8.8
4.519	7.0
4.628	17.9
...	...
8.375	50.0
8.398	48.8
8.704	50.0
8.725	50.0
8.780	21.9

Calculate Mean as Cut Point

Calculate Mean Squared Error for each set

Decision Tree Regressor

Decision Tree Regressor

- Will easily overfit without guidance
- Easy to reduce overfitting

splitter : *{"best", "random"}, default="best"*

The strategy used to choose the split at each node. Supported strategies are "best" to choose the best split and "random" to choose the best random split.

max_depth : *int, default=None*

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

min_samples_split : *int or float, default=2*

The minimum number of samples required to split an internal node:

- If int, then consider `min_samples_split` as the minimum number.
- If float, then `min_samples_split` is a fraction and `ceil(min_samples_split * n_samples)` are the minimum number of samples for each split.

> *Changed in version 0.18:* Added float values for fractions.

min_samples_leaf : *int or float, default=1*

The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least `min_samples_leaf` training samples in each of the left and right branches. This may have the effect of smoothing the model, especially in regression.

- If int, then consider `min_samples_leaf` as the minimum number.
- If float, then `min_samples_leaf` is a fraction and `ceil(min_samples_leaf * n_samples)` are the minimum number of samples for each node.

K-Fold Cross Validation

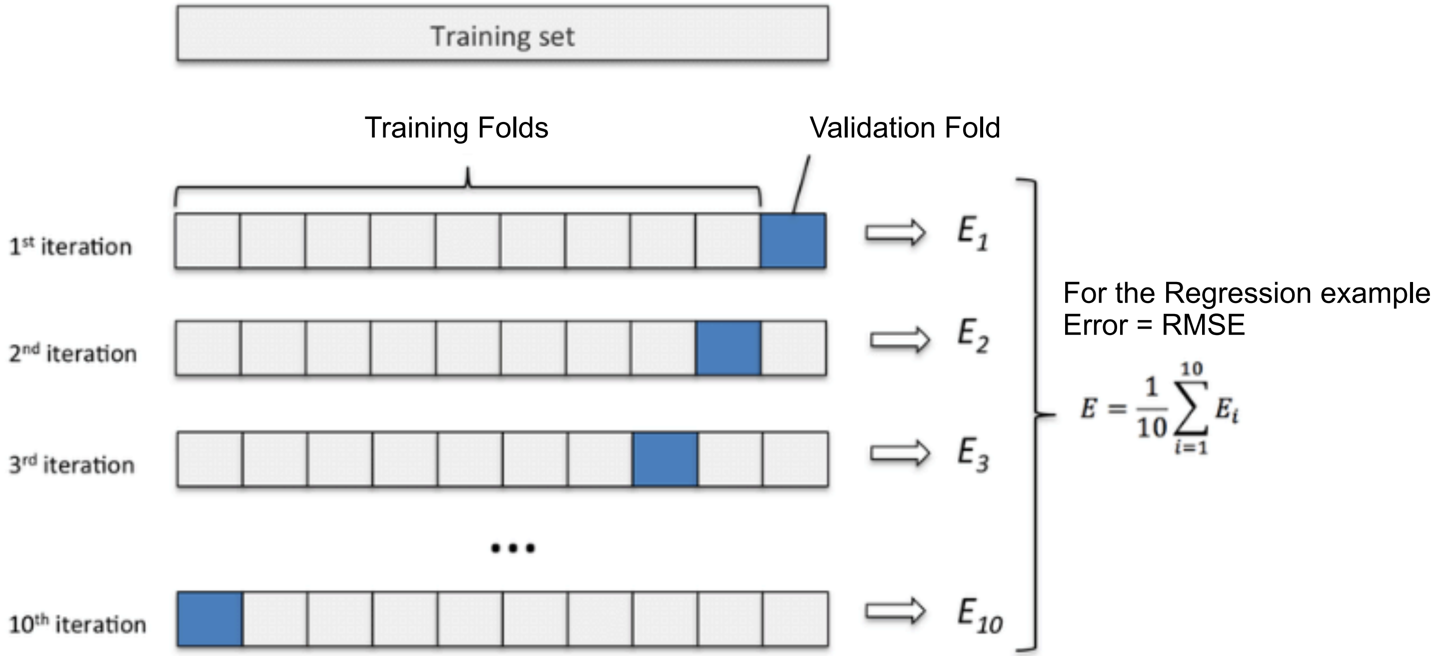
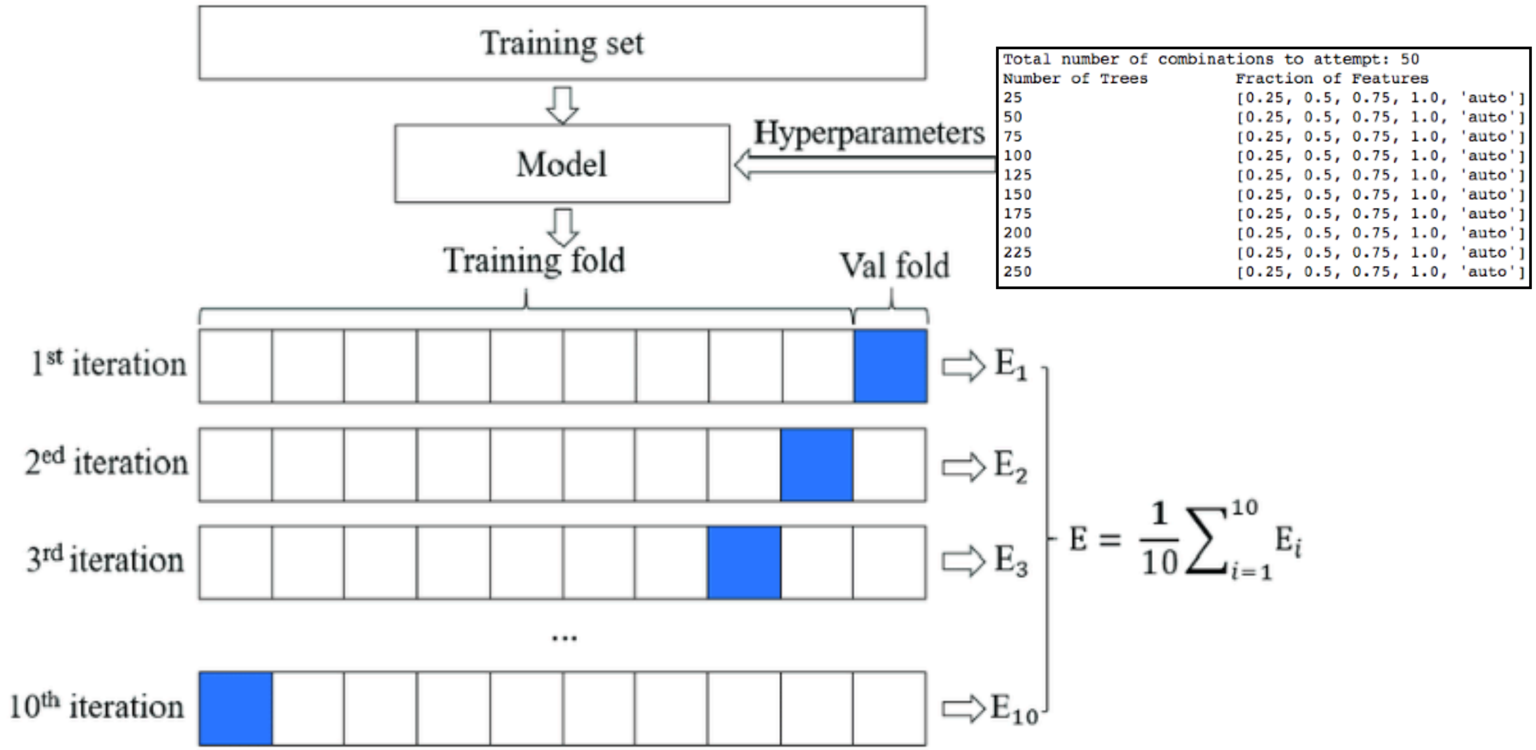


Diagram of k-fold cross-validation with k = 10. Image from Karl Rosaen Log http://karlrosaen.com/ml/learning-log/2016-06-20/

K-Fold CV for Tuning



Interpretability & Explainability

- Recent job requirement
  - The days of the *black box* are coming to an end

Interpretability & Explainability

- Recent job requirement
  - The days of the *black box* are coming to an end
  - “Why did you choose that algorithm?”

Interpretability & Explainability

- Recent job requirement
  - The days of the *black box* are coming to an end
  - “Why did you choose that algorithm?”
  - “How do you handle *some obscure buzzword*?”

Interpretability & Explainability

- Recent job requirement
  - The days of the *black box* are coming to an end
  - “Why did you choose that algorithm?”
  - “How do you handle *some obscure buzzword*?”
  - “My intuition says…”

Interpretability & Explainability

- Recent job requirement
  - The days of the *black box* are coming to an end
  - “Why did you choose that algorithm?”
  - “How do you handle *some obscure buzzword*?”
  - “My intuition says…”
- Ability to articulate complicated, technical processes to non-technical stakeholders

Interpretability & Explainability

- Recent job requirement
  - The days of the *black box* are coming to an end
  - “Why did you choose that algorithm?”
  - “How do you handle *some obscure buzzword*?”
  - “My intuition says…”
- Ability to articulate complicated, technical processes to non-technical stakeholders
- Visualizations offer simplicity
  - Picture… 1000 words…

Interpretability & Explainability

- Model Selection
  - Tree Based
  - Linear and Logistic Regression
  - Deep Learning
- Feature Selection
- Individual Conditional Expectation
- Partial Dependence Plot
- SHAP Analysis

Feature Selection

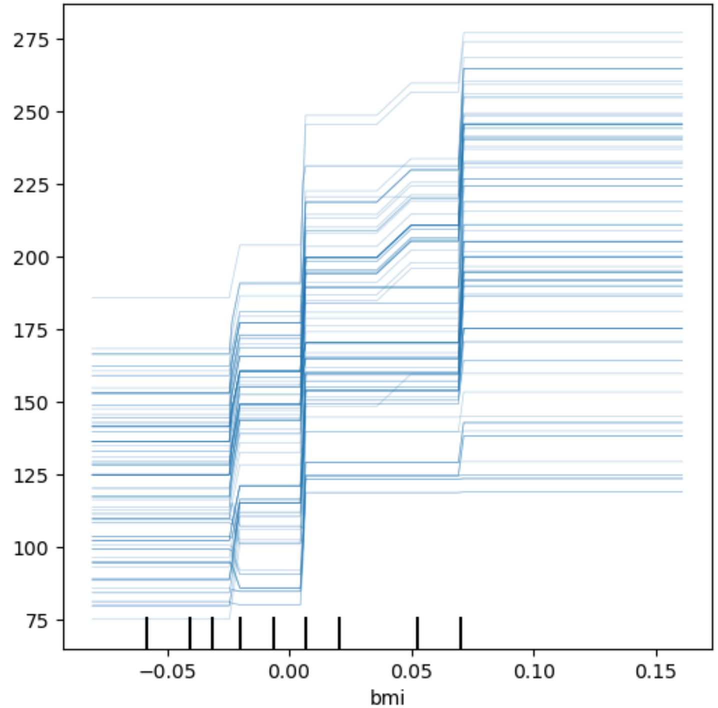
- Parsimony
  - Employ the fewest number of features necessary

s5	0.329587
bmi	0.178243
bp	0.033039
s6	0.029752
s3	0.005360
sex	0.000000
s1	0.000000
s4	0.000000
age	-0.001817
s2	-0.004315

```
XGB Test RMSE (all features): 59.34169589490085
XGB Test RMSE (reduced features): 58.923764600353856
```

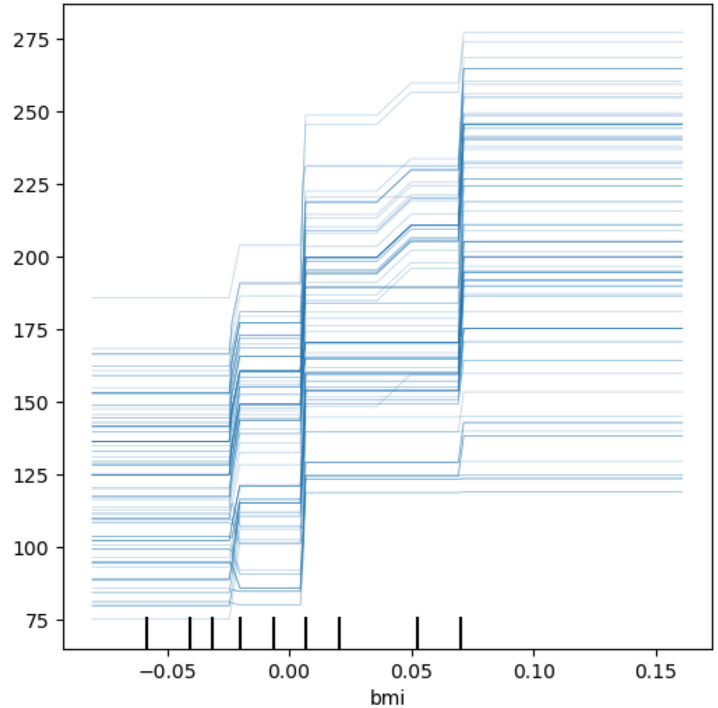
Individual Conditional Expectation

Demonstrates the marginal effect of altering a feature has on the predictions from a machine learning model.



Individual Conditional Expectation

Demonstrates the marginal effect of altering a feature has on the predictions from a machine learning model.

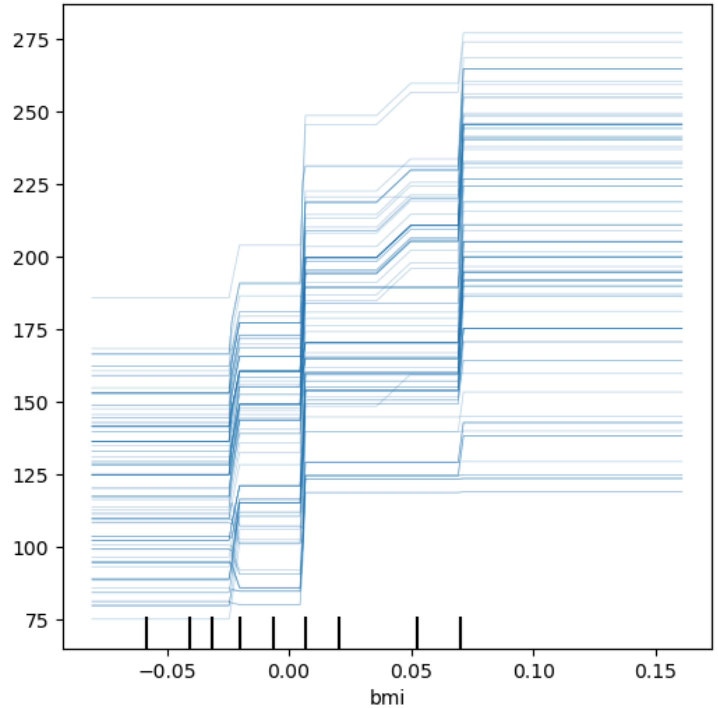


The values for each instance are computed by keeping all other features the same, creating variants of this instance by replacing the given feature's value with values from a grid, and making predictions with the model for these newly created instances.

Individual Conditional Expectation

Demonstrates the marginal effect of altering a feature has on the predictions from a machine learning model.

Updated predictions are shown in the y-axis.

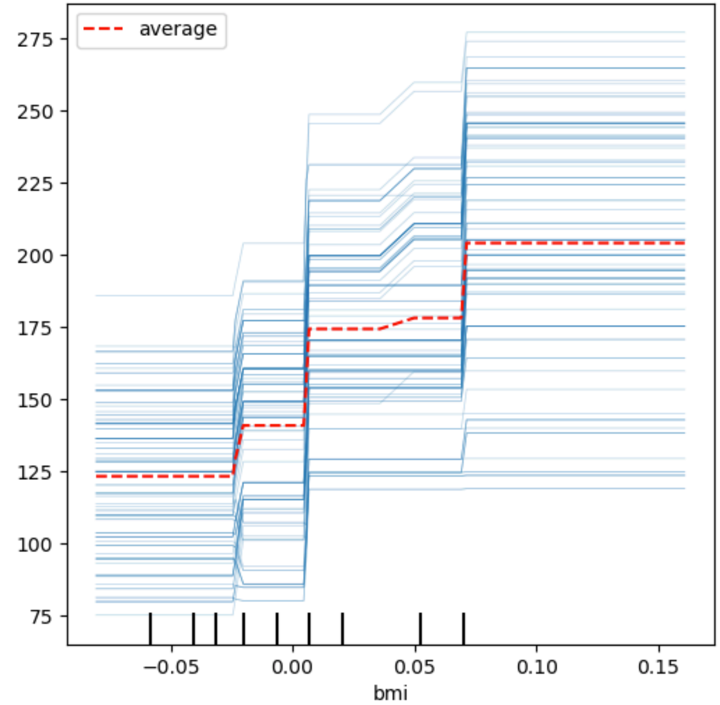


Default is a grid of 100 equally spaced points for the feature given.

The values for each instance are computed by keeping all other features the same, creating variants of this instance by replacing the given feature's value with values from a grid, and making predictions with the model for these newly created instances.

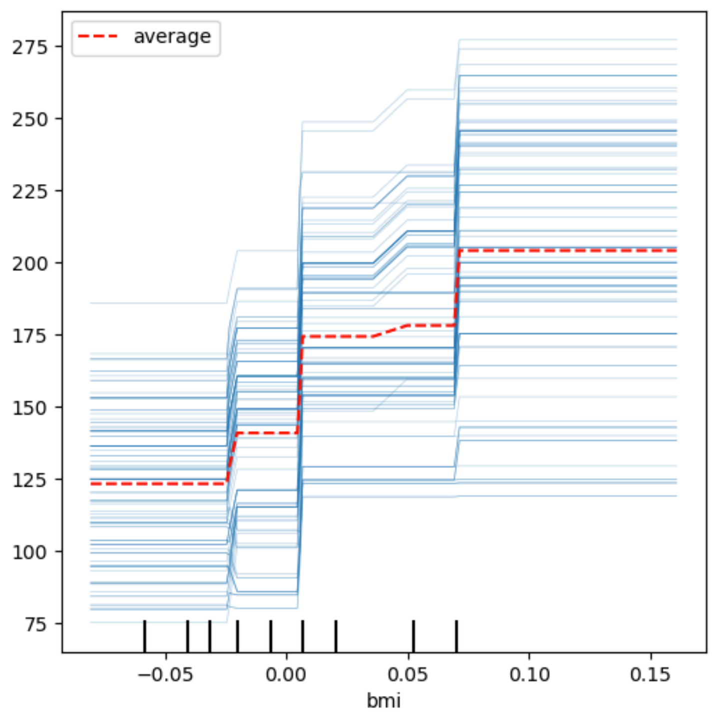
Partial Dependence Plot

Shows the complexity in the average relationship between the target and the feature.

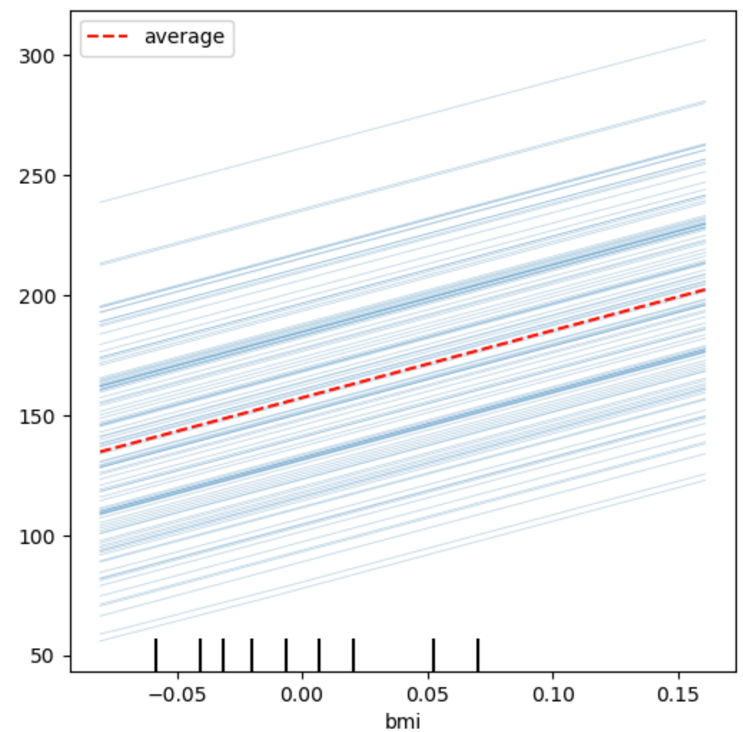


Partial Dependence Plot

Shows the complexity in the average relationship between the target and the feature.



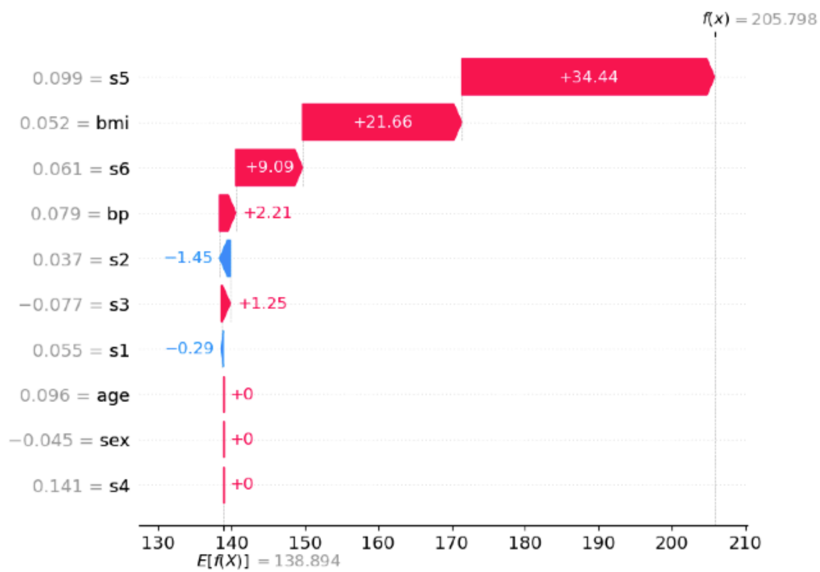
XGBoost Regressor

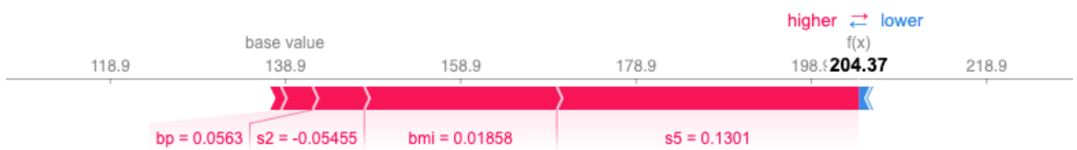


MLP Regressor

SHAP Analysis

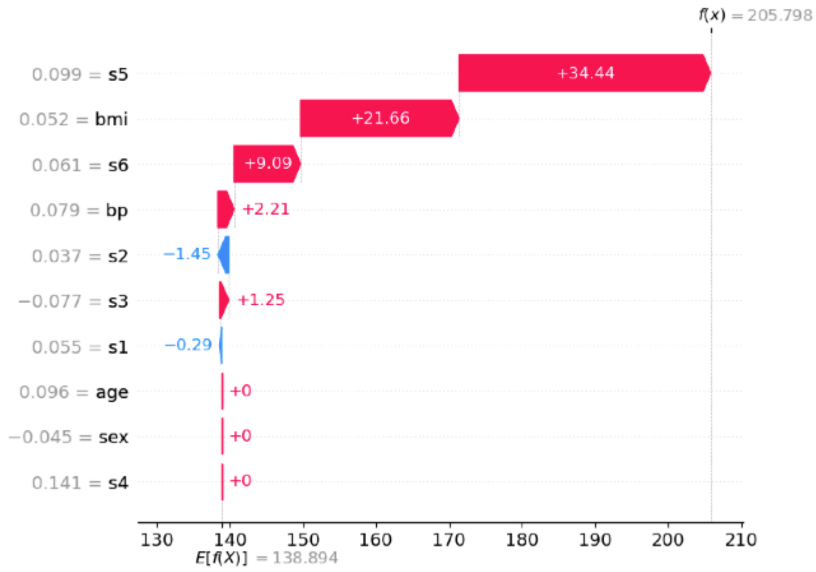
Explain the prediction of an instance by computing the contribution of each feature to the prediction. Shapley values demonstrate how to fairly distribute the the prediction among the features.

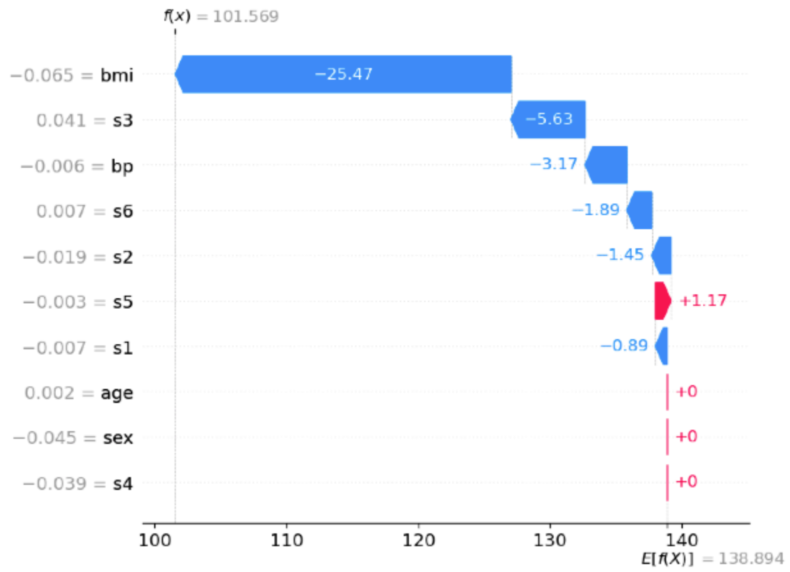




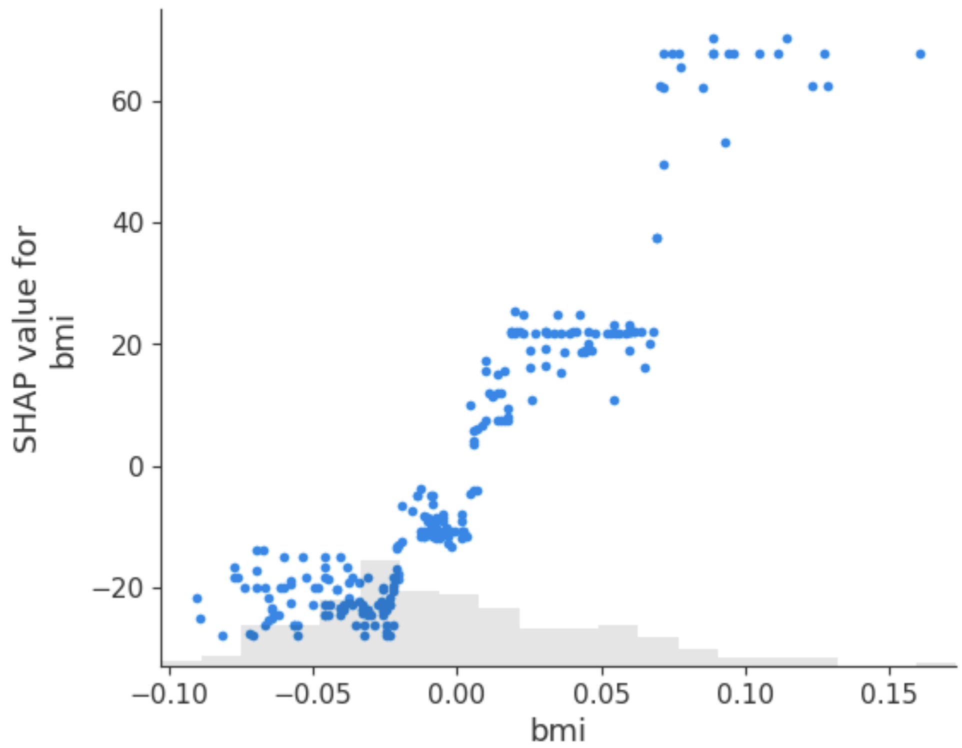
SHAP Analysis (Single Instance)

Explain the prediction of an instance by computing the contribution of each feature to the prediction. Shapley values demonstrate how to fairly distribute the the prediction among the features.

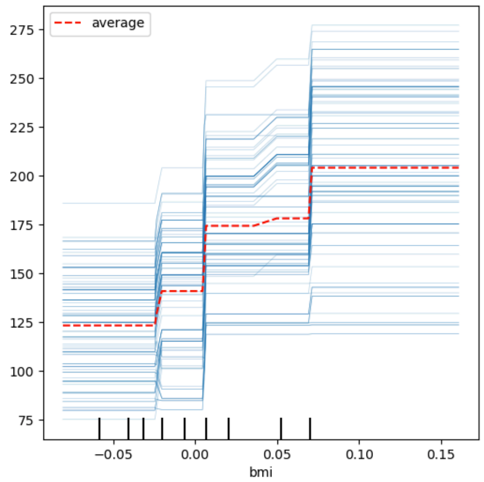


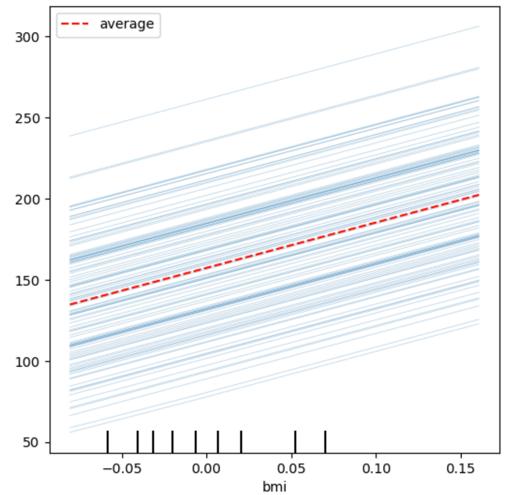


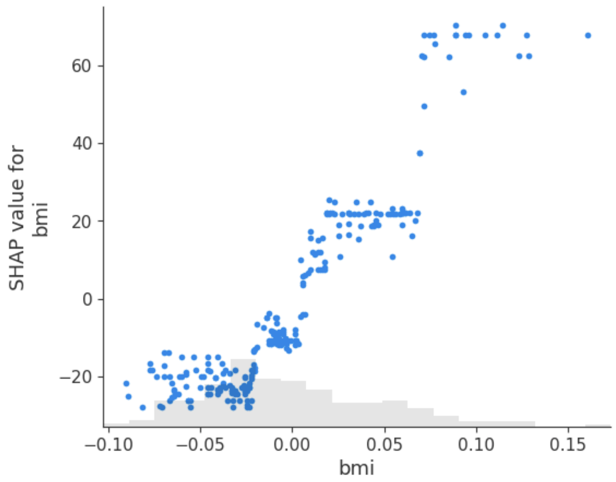
SHAP Dependence (Single Feature)



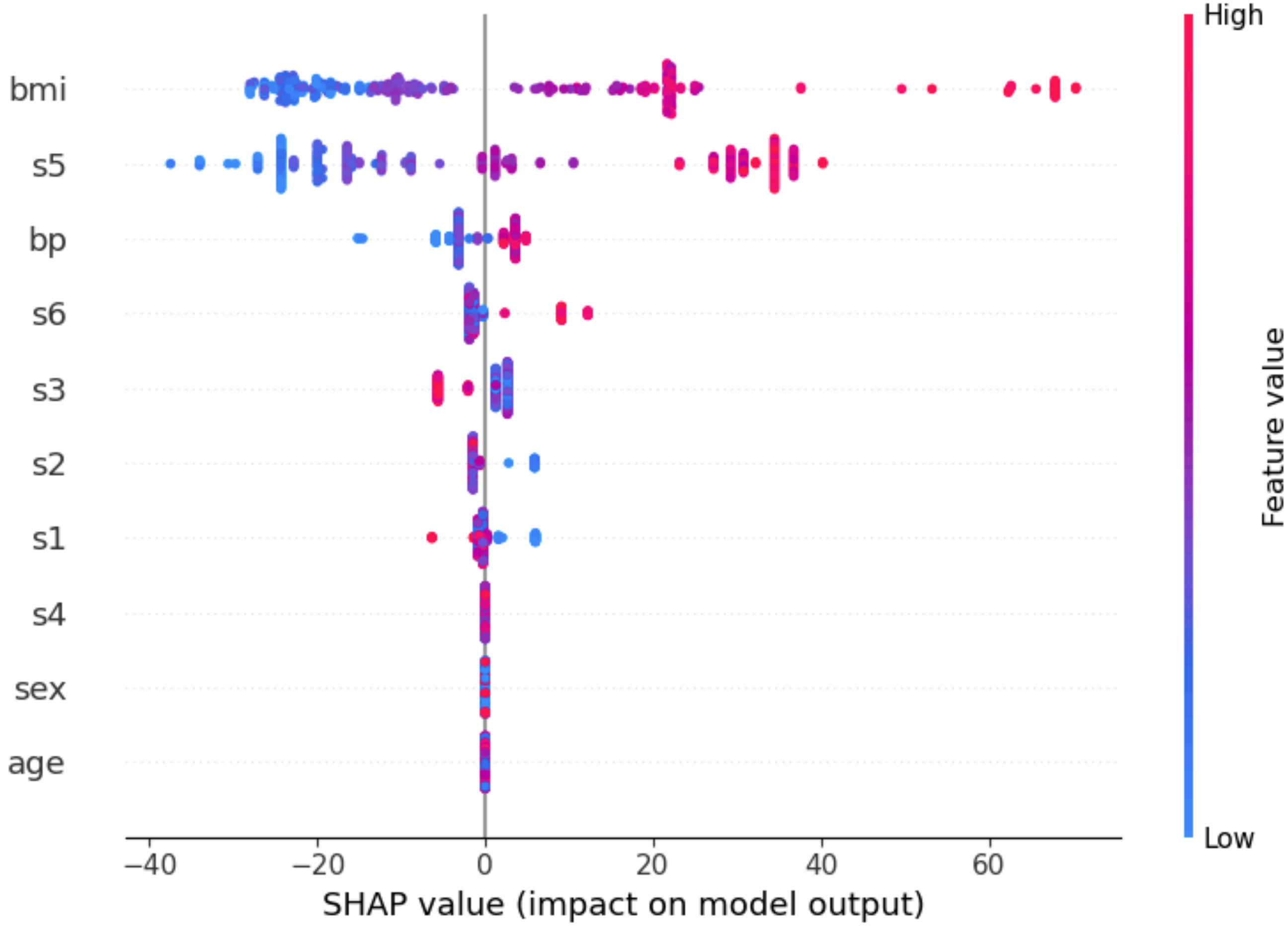
Dependence Plot Comparison







SHAP Summary Plot (Entire Dataset)



SHAP Analysis

Explain the prediction of an instance by computing the contribution of each feature to the prediction. Shapley values demonstrate how to fairly distribute the the prediction among the features.

In terms of coalition game theory, the “game” is predicting an outcome.
The prediction will have a combination of features, called a “coalition”.
The “gain” is the difference between the predicted outcome against the average predicted outcome for all combinations of features.
The “players” are the feature values employed as input into the model which work together to create the gain (or difference from the average value).

Machine learning can solve this problem by building a model to predict an outcome, taking all the variables into account. A SHAP analysis of that model will give an indication of how significant each feature is in determining the final prediction the model outputs. This is done by running a large number of predictions comparing the impact of a variable against the other features.