Explainable AI - Shapley values

Lecture at

“**48th Winter Conference in Statistics”**

March 11th, 2024

Agenda

- Shapley values in general
- Shapley regression
- Shapley values for prediction explanation

Shapley values in general

Example

- Amy, Bob and Claire are sharing a taxi.
- We imagine they go in the same direction
- If going alone
  - Amy would pay 6 \$ to go home
  - Bob would pay 12 \$ to go home
  - Claire would pay 42 \$ to go home
- What would be the optimal way of sharing the bill?

Shapley values

- Based on concepts from cooperative game theory.
- Originally invented for assigning payouts to players depending on their contribution towards the total payout.
- Consider a game with M players aiming at maximising a payoff
- Let $S \subseteq \mathcal{M} = \{1 \dots M\}$ be a subset of the M players
- Assume that we have a function v(S) that maps subsets of players to real numbers. v(S) is called the worth or contribution of S and describes the total expected payoff the members of S can obtain by cooperation.
- The Shapley value is one way of distributing the total gain to the players.

Shapley formula

- According to the Shapley value, the amount that player j gets is

$$\phi_j(v) = \phi_j = \sum_{\mathcal{S} \subseteq \mathcal{M} \setminus \{j\}} \frac{|\mathcal{S}|!(M - |\mathcal{S}| - 1)!}{M!} (v(\mathcal{S} \cup \{j\}) - v(\mathcal{S})), \quad j = 1, \dots, M,$$

- i.e. a weighted mean over all subsets S of players not containing player j.

The Shapley value is the average expected marginal contribution of one player after all possible combinations have been considered.

Example with 3 players

- Assume that we have a game with three players, i.e. $\mathcal{M} = \{1,2,3\}$.
- Then, there are 8 possible subsets:
  $\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\},$ and $\{1,2,3\}$.
- Using the Shapley formula, the Shapley values for the 3 players are:

$$\begin{aligned}
\phi_1 &= \tfrac{1}{3}\big(v(\{1,2,3\})-v(\{2,3\})\big)+\tfrac{1}{6}\big(v(\{1,2\})-v(\{2\})\big)+\tfrac{1}{6}\big(v(\{1,3\})-v(\{3\})\big)+\tfrac{1}{3}\big(v(\{1\})-v(\emptyset)\big),\\
\phi_2 &= \tfrac{1}{3}\big(v(\{1,2,3\})-v(\{1,3\})\big)+\tfrac{1}{6}\big(v(\{1,2\})-v(\{1\})\big)+\tfrac{1}{6}\big(v(\{2,3\})-v(\{3\})\big)+\tfrac{1}{3}\big(v(\{2\})-v(\emptyset)\big),\\
\phi_3 &= \tfrac{1}{3}\big(v(\{1,2,3\})-v(\{1,2\})\big)+\tfrac{1}{6}\big(v(\{1,3\})-v(\{1\})\big)+\tfrac{1}{6}\big(v(\{2,3\})-v(\{2\})\big)+\tfrac{1}{3}\big(v(\{3\})-v(\emptyset)\big).
\end{aligned}$$

- Summarizing the right hand sides, we get $\phi_0 + \phi_1 + \phi_2 + \phi_3 = v(\{1,2,3\})$.

Here, $\phi_0 = v(\emptyset)$

Properties

Efficiency: The total gain is distributed:

$$\sum_{j=0}^{M} \phi_j = v(\mathcal{M})$$

Symmetry: If i and j are two players who contribute equally to all possible coalitions, i.e.

$$v(\mathcal{S} \cup \{i\}) = v(\mathcal{S} \cup \{j\})$$

for every subset $\mathcal{S}$ which contains neither i nor j, then their Shapley values are identical:

$$\phi_i = \phi_j.$$

Dummy player: If $v(\mathcal{S} \cup \{j\}) = v(\mathcal{S})$ for a player j and all coalitions $\mathcal{S} \subseteq \mathcal{M} \setminus \{j\}$, then $\phi_j = 0$.

Linearity: If two coalition games described by gain functions v and w are combined, then the distributed gains correspond to the gains derived from v and the gains derived from w:

$$\phi_i(v + w) = \phi_i(v) + \phi_i(w),$$

for every i. Also, for any real number a we have that

$$\phi_i(a\,v) = a\phi_i(v).$$

The Shapley values are the only set of values satisfying these properties, see Shapley (1953) Young (1985) for proofs.

Example

- Amy, Bob and Claire are sharing a taxi.
- We imagine they go in the same direction
- If going alone
  - Amy would pay 6 $ to go home
  - Bob would pay 12 $ to go home
  - Claire would pay 42 $ to go home
- What would be the optimal way of sharing the bill?
- Use Shapley values to answer this question!

Example

- We have that v(1)=6, v(2)=12, v(3)=42, v(1,2)=12, v(1,3)=42, v(2,3)=42 v(1,2,3)=42.
- Amy should pay
- (1/3)*(42-42) + (1/6)*(12-12) + (1/6)*(42-42) + (1/3)*6 = 2
- Bob should pay
- (1/3)*(42-42) + (1/6)*(12-6) + (1/6)*(42-42) + (1/3)*12 = 5
- Claire should pay
- (1/3)*(42-12) + (1/6)*(42-6) + (1/6)*(42-12) + (1/3)*42 = 35

Example is from https://github.com/shapley-value-java/shapley-value-core

Example

- The Shapley solution says that:
  - Amy should pay 1/3 of the cost (6$) to her home
  - Bob should pay 1/3 of the cost (6$) to Amy's home plus 1/2 of the cost between her home and his home (6$) .
  - Claire should pay 1/3 of the cost to Amy's home (6$) , plus 1/2 of the cost between Amy's home and Bob's home (6$) plus the full cost between her home and Bob's home (30$) .

The airport problem

Source: freepik.com

- A famous example of the Shapley value in practice is the airport problem.
- In this problem, an airport needs to be built in order to accommodate a range of aircrafts which require different lengths of runway. The question is how to distribute the costs of the airport to all actors in an equitable manner.
- Simplified example:
  - An airport needs to build a runway for 4 different aircraft types.
  - The building costs associated with aircrafts A, B, C, D are 8, 11, 13, 18.
  - What are the Shapley values for A, B, C and D?

Source: https://www.investopedia.com/terms/s/shapley-value.asp

Solution

- Aircraft A should pay 1/4*8 = 2
- Aircraft B should pay 1/4*8 +1/3*3 = 3
- Aircraft C should pay 1/4*8 +1/3*3 + 1/2*2 = 4
- Aircraft D should pay 1/4*8 +1/3*3 + 1/2*2 + 1*5 = 9

Shapley regression

Shapley regression

- Global variable importance for linear regression
- Players = covariates $(x_1, \dots, x_M)$
- Contribution function $v(S) = R^2$
- Compute linear regression models for all possible combinations of covariates.
- Use the Shapley formula to decompose R^2 into the contribution for each covariate.

Example

- Three variables $X_1, X_2 \; X_3$

Model	Predictors	R^2
0	none	0
1	X_1	0.1064
2	X_2	0.1254
3	X_3	0.1288
1, 2	X_1, X_2	0.1807
1, 3	X_1, X_3	0.1700
2, 3	X_2, X_3	0.1873
1, 2, 3	X_1, X_2, X_3	0.2144

A model with intercept only

- Shapley values:
  - X_1 $\frac{1}{3}(R^2_{1,2,3} - R^2_{2,3}) + \frac{1}{6}(R^2_{1,2} - R^2_{2}) + \frac{1}{6}(R^2_{1,3} - R^2_{3}) + \frac{1}{3}(R^2_{1} - R^2_{0}) = 0.0606$
  - X_2 $\frac{1}{3}(R^2_{1,2,3} - R^2_{1,3}) + \frac{1}{6}(R^2_{1,2} - R^2_{1}) + \frac{1}{6}(R^2_{2,3} - R^2_{3}) + \frac{1}{3}(R^2_{2} - R^2_{0}) = 0.0788$
  - X_3 $\frac{1}{3}(R^2_{1,2,3} - R^2_{1,2}) + \frac{1}{6}(R^2_{1,3} - R^2_{1}) + \frac{1}{6}(R^2_{2,3} - R^2_{2}) + \frac{1}{3}(R^2_{3} - R^2_{0}) = 0.0751$

This is exactly the same formula for X1 as the one we obtained using the lmg method!

Relative weights

- Shapley regression and the LMG method are computationally heavy when we have many covariates.
- Relative Weights (Johnson, 2000) produces similar scores with much faster computation.
- See https://www.displayr.com/shapley-vs-relative-weights/

```
> library(relaimpo)
> library(tibble)
> library(tidyverse)
> library(rwa)
> library(ggplot)

> rwa(bikeTrain[,-6],"cnt",c("temp","hum","windspeed","days_since_2011"))$result$Rescaled.RelWeight

  46.480548 3.520444 3.630216 46.36879

> 100*calc.relimp(cnt~.,data=bikeTrain[,8:12],type ="lmg", rela = TRUE )$lmg
  temp hum windspeed days_since_2011
  46.512680 3.578442 3.470233 46.438645
```

The function rwa() does unfortunately only handle numeric predictors.

Shapley values for prediction explanation

- Introduction
- Linear model
- Approximations to Shapley formula
  - KernelSHAP
  - FastSHAP
  - GroupSHAP
- Methods for estimating contribution function
  - Via conditional distribution
  - Directly

$$\phi_j(v) = \phi_j = \sum_{\mathcal{S} \subseteq \mathcal{M} \setminus \{j\}} \frac{|\mathcal{S}|!(M - |\mathcal{S}| - 1)!}{M!} (v(\mathcal{S} \cup \{j\}) - v(\mathcal{S})), \quad j = 1, \dots, M,$$

Introduction

- Players = covariates $(x_1, \ldots, x_M)$
- The instance to be explained = $\boldsymbol{x}^*$
- Payoff = prediction $f(\boldsymbol{x}^*)$
- Contribution function: $v(S) = E[f(\boldsymbol{x}) | \boldsymbol{x}_S = \boldsymbol{x}_S^*]$
- Properties:
  - $f(\boldsymbol{x}^*) = E[f(\boldsymbol{x})] + \sum_{j=1}^{M} \phi_j$
  - f independent of $x_j \Rightarrow \phi_j = 0$
  - x_i, x_j same contribution $\Rightarrow \phi_i = \phi_j$

The Shapley values explain the difference between the prediction $f(\boldsymbol{x}^*)$ and the average prediction.

19

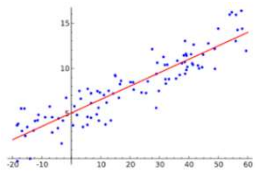
Linear model

- If we have a linear regression model $f(\boldsymbol{x}) = \beta_0 + \sum_{j=1}^{M} \beta_j x_j$ with independent features, the Shapley values take the simple form

$$\phi_0 = \beta_0 + \sum_{i=1}^{M} \beta_j E[x_j], \quad \text{and} \quad \phi_j = \beta_j (x_j^* - E[x_j]), \quad j = 1, \ldots, M.$$

- See Aas et. al. (2021) for a proof.
- For the general case with a non-linear model and dependent features, no such explicit formula exists.

Remember that $v(S) = E[f(\boldsymbol{x}) | \boldsymbol{x}_S = \boldsymbol{x}_S^*]$



Two main challenges

- The computational complexity of the Shapley formula
  - With M features, the number of possible subsets is 2^M

$$\phi_j(v) = \phi_j = \sum_{\mathcal{S} \subseteq \mathcal{M} \setminus \{j\}} \frac{|\mathcal{S}|!(M - |\mathcal{S}| - 1)!}{M!}(v(\mathcal{S} \cup \{j\}) - v(\mathcal{S})), \quad j = 1, \dots, M,$$

- Estimating the contribution function $v(S)$
  - Not trivial if model is non-linear features are dependent

$$v(S) = E[f(\boldsymbol{x})|\boldsymbol{x}_S = \boldsymbol{x}_S^*]$$

Computational complexity

Computational complexity

- Possible solutions
  - Kernel SHAP
  - FastSHAP
  - treeSHAP
  - Group Shapley

Kernel SHAP (Lundberg & Lee, 2017)

- In Kernel SHAP, the Shapley values are defined as the solution of a certain weighted least squares (WLS) problem.
- In its simplest form the WLS problem can be stated as minimizing

$$\sum_{\mathcal{S}\subseteq\mathcal{M}} (v(\mathcal{S}) - (\phi_0 + \sum_{j\in\mathcal{S}} \phi_j))^2 k(M,\mathcal{S}),$$

Remember that $v(S) = E[f(\boldsymbol{x})|\boldsymbol{x}_S = \boldsymbol{x}_S^*]$

- with respect to $\phi_0, \ldots, \phi_M$ where $k(M,\mathcal{S}) = (M-1)/(\binom{M}{|\mathcal{S}|}\,|\mathcal{S}|\,(M-|\mathcal{S}|))$ are denoted the Shapley kernel weights
- See the Supplementary Material of Lundberg & Lee (2017) for a proof.

We have that k(M,M)=k(M,0)=∞. We set these weights to a large constant C to avoid numerical problems.

Remember that $v(S) = E[f(\boldsymbol{x})|\boldsymbol{x}_S = \boldsymbol{x}_S^*]$

Example: Dimension 3

- Linear weighted regression where we have 2^3 =8 “observations” and 4 explanatory variables and want to determine ϕ_0, ϕ_1, ϕ_2 and ϕ_3
  - Response v(1,2,3) covariate vector 1,1,1,1 weight 10^6 Want to have $f(\boldsymbol{x}^*) = \sum_{j=0}^{p} \phi_j$
  - Response v(1,2) covariate vector 1,1,1,0 weight 1/3
  - Response v(1,3) covariate vector 1,1,0,1 weight 1/3
  - Response v(2,3) covariate vector 1,0,1,1 weight 1/3
  - Response v(1) covariate vector 1,1,0,0 weight 1/3
  - Response v(2) covariate vector 1,0,1,0 weight 1/3
  - Response v(3) covariate vector 1,0,0,1 weight 1/3
  - Response v(0) covariate vector 1,0,0,0 weight 10^6 Want to have $f(0) = \phi_0$

Taxi example revisited

- Let v(1)=6, v(2)=12, v(3)=42, v(1,2)=12, v(1,3)=42, v(2,3)=42 v(1,2,3)=42.

```
y <- c(0,6,12,42,12,42,42,42)
m <- 3
xMat <- NULL
for (i in 1:m)
  { # compute all possible combinations of i features
    coalitions <- combn(m, i)
    tmpMat <- matrix(0,ncol=m,nrow=ncol(coalitions))
    for(j in 1:ncol(coalitions))
        tmpMat[j,coalitions[,j]] <- 1
    xMat <- rbind(xMat,tmpMat) }
 #Add row for intercept
 xMat <- rbind(rep(0,m),xMat)

 d <- dim(xMat)[1]
 w <- array(0,d)
 for(i in 1:d)
   { s <- length(which(xMat[i,]==1))
     w[i] = (m-1)/(mCooseS(m,s)*s*(m-s))}
  w[1] <- 10^6
  w[d] <- 10^6
lm(y ~ ., data=as.data.frame(cbind(y,xMat)),weights=w)
```

```
Coefficients:
(Intercept)          V2          V3          V4
  5.333e-06   2.000e+00   5.000e+00   3.500e+01
```

which is exactly the same answer we got using the standard Shapley formula!

Kernel SHAP

- The formula at page 21 may be rewritten to

$$(\boldsymbol{v} - \boldsymbol{Z}\boldsymbol{\phi})^T \boldsymbol{W} (\boldsymbol{v} - \boldsymbol{Z}\boldsymbol{\phi})$$

- where
- ***Z*** is a matrix with 2^M rows and *M+1* columns representing all possible subsets of the *M* features (the first column is 1 for every row).
- **v** is a vector containing v(*S*) for each subset *S*.
- **W** is a 2^M x 2^M diagonal matrix containing $k(M, \mathcal{S})$

Kernel SHAP

- The solution of the regression problem is

$$\boldsymbol{\phi} = \left(\boldsymbol{Z}^T \boldsymbol{W} \boldsymbol{Z}\right)^{-1} \boldsymbol{Z}^T \boldsymbol{W} \boldsymbol{v}.$$

- When M is large, the computation of the formula above is expensive.
- In the Kernel SHAP method, one uses the approximation

$$\boldsymbol{\phi} = \left[\left(\boldsymbol{Z}_{\mathcal{D}}^T \boldsymbol{W}_{\mathcal{D}} \boldsymbol{Z}_{\mathcal{D}}\right)^{-1} \boldsymbol{Z}_{\mathcal{D}}^T \boldsymbol{W}_{\mathcal{D}}\right] \boldsymbol{v}_{\mathcal{D}}$$

- where only a subset *D* of the rows in ***Z*** are used.

Kernel SHAP

- To select the subset *D* we utilise the fact that the Shapley kernel weights have very different sizes.
- We sample with replacement a subset *D* of $\mathcal{M}$ from a probability distribution following the Shapley weighting kernel.
- As the kernel weights are used in the sampling, the sampled subsets are weighted equally in the new least squares problem.

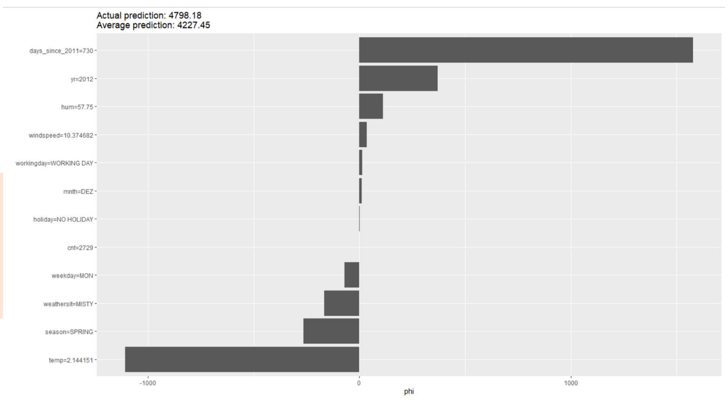
Shapley weighting kernel:

$$k(M, \mathcal{S}) = (M-1)/(\binom{M}{|\mathcal{S}|}\,|\mathcal{S}|\,(M-|\mathcal{S}|))$$

Example

- Bike data set

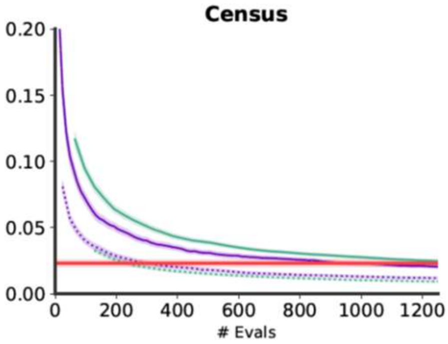
```
library("iml")
pfun <- function(object, newdata)
  predict(object, data = newdata)$predictions
mod <- Predictor$new(model = model, data = bikeTrain, predict.fun = pfun)
x.interest <- bikeTest[1, ]
shapley <- Shapley$new(mod, x.interest = x.interest)
plot(shapley)
```



	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	hum	windspeed	days_since_2011
SHAPVAL	-265.50	370.88	9.90	0.50	-70.56	13.45	-165.51	-1107.93	111.29	35.56	1578.26

FastSHAP

- Estimate a parametric function g(**x**;θ) for computing the Shapley values.
- The parametric function is a neural network
- Can only be used for classification problems, typically when the response is either 0 or 1.
- Need an estimate for $v(S) = E[f(\boldsymbol{x})|\boldsymbol{x}_S = \boldsymbol{x}_S^*]$.
  - FastSHAP uses the method by Frye et. al. (2021), but any of the methods to be discussed might be used.



Error as function of the size of D.
Green line is KernelSHAP
Red line is fastSHAP.

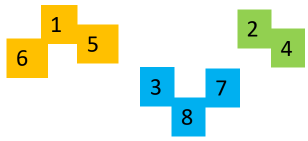
See https://arxiv.org/pdf/2107.07436.pdf for more details.

treeSHAP

- An algorithm for computing Shapley values for tree ensemble models, such as XGBoost and Random forests.
- Makes use of the structure of the tree algorithm to extract and generate Shapley values much faster than Kernel SHAP.
- Two methods for computing $v(S) = E[f(\boldsymbol{x})|\boldsymbol{x}_S = \boldsymbol{x}_S^*]$
  - Interventional: Assumes feature independence
  - Tree_path_dependent: Incorporates some feature dependence.

Group Shapley (Jullum et. al, 2021)

- As the number of features grows, the number of sampled subsets in the Kernel SHAP method also needs to grow (exponentially) in order to retain an acceptable accuracy for the approximated Shapley values.
- Jullum et. al. (2021) have proposed a method where one moves away from computing Shapley values for single features and rather compute Shapley values for groups of features.
- The groups may e.g. be based on
  - Feature type
  - Feature dependence



Group Shapley

- Assume that we have G groups $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_G\}$
- The Shapley value for the i'th group is then given by

$$\phi_{\mathcal{G}_i} = \sum_{\mathcal{T} \subseteq \mathcal{G} \setminus \mathcal{G}_i} \frac{|\mathcal{T}|_g!(G - |\mathcal{T}|_g - 1)!}{G!} (v(\mathcal{T} \cup \mathcal{G}_i) - v(\mathcal{T}))$$

- where the sum runs over all possible subsets $\mathcal{T}$ of the groups, and the contribution function is given by

$$v(\mathcal{T}) = \mathbb{E}[f(\boldsymbol{x}) | \boldsymbol{x}_{\mathcal{T}} = \boldsymbol{x}^*_{\mathcal{T}}]$$

- Here, $\boldsymbol{x}_{\mathcal{T}}$ denotes the subvector of $\boldsymbol{x}$ containing all features in group $\mathcal{T}$.

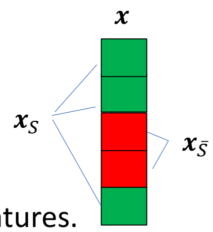
Group Shapley

- The group Shapley values possess all the regular Shapley value properties.
- The computational complexity of the Shapley formula reduces from
- 2^{M-1} to 2^{G-1}. With M = 50 features and G = 5 groups, the relative cost reduction is > 10^{13}.
- Let $\phi_{\text{post}-\mathcal{G}_i} = \sum_{j \in \mathcal{G}_i} \phi_j,$
- Then, under certain conditions* we have that $\phi_{post-\mathcal{G}_i} = \phi_{\mathcal{G}_i}$, i.e. summing the featurewise Shapley values for each group gives the same result as computing groupwise Shapley values.

*Partially additively separable prediction function and independent groups.

Contribution function

Introduction



- As previously stated the contribution function is defined as
$$v(S) = E[f(\boldsymbol{x}) | \boldsymbol{x}_S = \boldsymbol{x}_S^*]$$
- Most models $f(\boldsymbol{x})$ do not support making predictions without all the features.
- Hence, we need an approximation for $E[f(\boldsymbol{x}) | \boldsymbol{x}_S = \boldsymbol{x}_S^*]$
- Using the definition of the conditional expectation we have
$$E[f(\boldsymbol{x}) | \boldsymbol{x}_S = \boldsymbol{x}_S^*] = E[f(\boldsymbol{x}_{\bar{S}}, \boldsymbol{x}_S) | \boldsymbol{x}_S = \boldsymbol{x}_S^*] = \int f(\boldsymbol{x}_{\bar{S}}, \boldsymbol{x}_S^*)\, p(\boldsymbol{x}_{\bar{S}} | \boldsymbol{x}_S = \boldsymbol{x}_S^*) d\boldsymbol{x}_{\bar{S}}.$$
- Two main methods:
  - Estimate the conditional distribution and use Monte Carlo integration.
  - Estimate the expectation directly using a regression method or neural net

Via conditional distribution

- Estimate the conditional distribution $p(\boldsymbol{x}_{\bar{S}} | \boldsymbol{x}_S = \boldsymbol{x}_S^*)$
- Generate K samples from this distribution
- Estimate the value of
$$\int f(\boldsymbol{x}_{\bar{S}}, \boldsymbol{x}_S^*)\, p(\boldsymbol{x}_{\bar{S}} | \boldsymbol{x}_S = \boldsymbol{x}_S^*) d\boldsymbol{x}_{\bar{S}}$$
- using $\frac{1}{K} \sum_{k=1}^{K} f(\boldsymbol{x}_{\bar{S}}^k, \boldsymbol{x}_S^*)$

- Assuming independence
- Gaussian
- Copula
- ctree
- VAEAC

Assuming independence

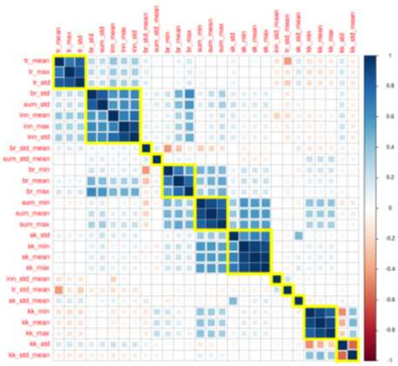
- Original KernelSHAP:
  - Assume that the features in the subset S are independent of the remaining features, meaning that we may replace $p(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^*)$ by $p(\boldsymbol{x}_{\bar{\mathcal{S}}})$

$$v(\mathcal{S}) = \mathbb{E}\left[f(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}})|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^*\right] \approx \hat{v}(\mathcal{S}) = \frac{1}{K}\sum_{k=1}^{K} f(\boldsymbol{x}_{\bar{\mathcal{S}}}^{(k)}, \boldsymbol{x}_{\mathcal{S}}^*), \quad \boldsymbol{x}_{\bar{\mathcal{S}}}^{(k)} \sim p(\boldsymbol{x}_{\bar{\mathcal{S}}}), \; k = 1, 2, \dots, K,$$

If the features are highly dependent, this may give a wrong answer!

Correlations

- If a group of variables are highly correlated, why not select one of the variables in this group ignore the other?
- Example: Mortgage robot
  - XGBoost model which predicts mortgage default
  - 28 features extracted from 6 transaction time series
  - Many features are highly correlated
  - We chose one variable from each group estimated a new model.
  - The AUC for this model was much lower than the AUC for the full model.



Parametric approaches

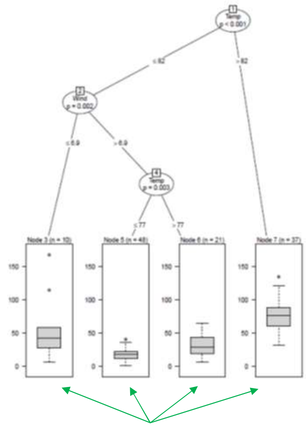
- Aas et al. (2019): Assume $p(\boldsymbol{x})$ Gaussian => analytical $p(\boldsymbol{x}_{\bar{S}}|\boldsymbol{x}_S = \boldsymbol{x}_S^*)$.
  - **Real data are seldom Gaussian.**
  - **Does not handle categorical variables**
- Aas et al. (2019): Assume a Gaussian copula approach.
  - **Real dependence structure is seldom Gaussian**
  - **Does not handle categorical variables**

Non-parametric approaches

- Aas et al. (2019): Use an empirical (conditional) approach where training observations at $\boldsymbol{x}_{\bar{S}}^k$ are weighted by proximity of $\boldsymbol{x}_S^k$ to $\boldsymbol{x}_S^*$.
  - **Does not handle categorical features**
- Redelmeier et al (2020): Use the conditional inference tree (ctree) approach to estimate $p(\boldsymbol{x}_{\bar{S}}|\boldsymbol{x}_S = \boldsymbol{x}_S^*)$
  - **Handles both numerical and categorical features**.
- Aas et al. (2021) Use vine copulas to estimate $p(\boldsymbol{x}_{\bar{S}}|\boldsymbol{x}_S = \boldsymbol{x}_S^*)$
  - **Does not handle categorical features**
- Olsen et al. (2022): Use the variational autoencoder with arbitrary conditioning (VAEC) to estimate $p(\boldsymbol{x}_{\bar{S}}|\boldsymbol{x}_S = \boldsymbol{x}_S^*)$.
  - **Handles both numerical and categorical features**.

ctree-approach

- Fit a multivariate regression tree with response $\boldsymbol{x}_{\bar{S}}$ and covariates $\boldsymbol{x}_S$ using the training data.
- Determine the terminal node in this tree to which $\boldsymbol{x}_S^*$ belongs.
- Approximate $p(\boldsymbol{x}_{\bar{S}}|\boldsymbol{x}_S = \boldsymbol{x}_S^*)$ by sampling K times from the training observations that also attained this node number.



ctree is implemented in the R-package partykit.

43

VAEAC approach

- The approaches just mentioned need to estimate the conditional distributions $p(x_{\bar{S}}|x_S = x_S^*)$ for all feature combinations $\mathcal{S} \in \mathcal{P}(\mathcal{M})$
- Olsen et. al (2022) use the variational autoencoder with arbitrary conditioning (VAEAC). This method can model all 2^M conditional distributions simultaneously using a single variational variational autoencoder.
- The VAEC consists of three fully connected neural networks; the encoder, the masked encoder and the decoder.

For more details see: https://www.jmlr.org/papers/volume23/21-1413/21-1413.pdf

Estimating expectation directly

- Estimate expectation directly instead of going via conditional distribution and Monte Carlo Integration

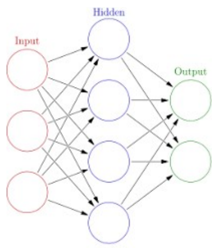
$$E[f(\boldsymbol{x})|\boldsymbol{x}_S = \boldsymbol{x}_S^*] = E[f(\boldsymbol{x}_{\bar{S}}, \boldsymbol{x}_S)|\boldsymbol{x}_S = \boldsymbol{x}_S^*] \cancel{= \int f(\boldsymbol{x}_{\bar{S}}, \boldsymbol{x}_S^*)\, p(\boldsymbol{x}_{\bar{S}}|\boldsymbol{x}_S = \boldsymbol{x}_S^*) d\boldsymbol{x}_{\bar{S}}}$$

- Any regression method might be used, e.g.
  - GAM, Projection pursuit, Random forest, XGBoost, neural net….

For a survey see: https://arxiv.org/pdf/2305.09536.pdf (accepted for publication in Data Mining and Knowledge Discovery).

Method by Frye et. al. (2021):

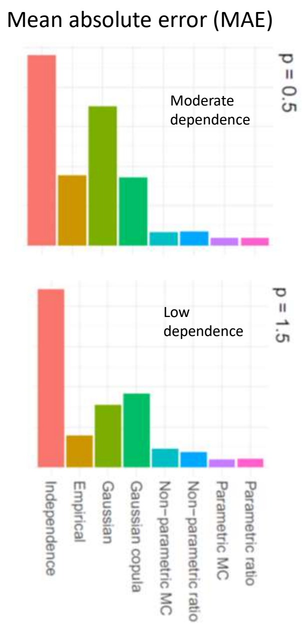
- Train one single neural network g(**x**,m(**x**,S)) to estimate $v(S) = E[f(\boldsymbol{x})|\boldsymbol{x}_S = \boldsymbol{x}_S^*]$ for all coalitions S.
- Takes as input a vector of masked features m(**x**,S), where the mask replaces the elements of **x** that are not in the subset S by a value not in the distribution of **x**.
- The loss function of the neural net is the expected value of
- |f(**x**)-g(**x**,m(**x**,S))|^2
- where the expectation is over all values **x** of **X** and "all" subsets S



See https://arxiv.org/pdf/2006.01272.pdf for more information

Evaluation – simulated data

- No ground truth ➡ not obvious how to evaluate the different approaches.
- We have compared Shapley with and without taking dependence into account in several controlled experiments.
  - Linear and non-linear models
  - Gaussian and non-Gaussian distributions



47

Observations

- It is important to take the dependence between the features into account when computing the Shapley values.
- If the data is close to Gaussian, the Gaussian and Gaussian copula methods show the best performance.
- If we have non-Gaussian data with high dependence VAEAC and ctree show the best performance.
- If estimating the expectation directly, using a regression model of the same form as the predictive model often provides more precise Shapley values.

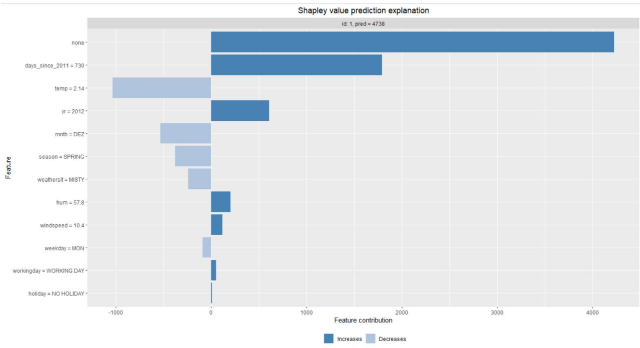
For more details see: https://arxiv.org/pdf/2305.09536.pdf

Example: Bike rental data

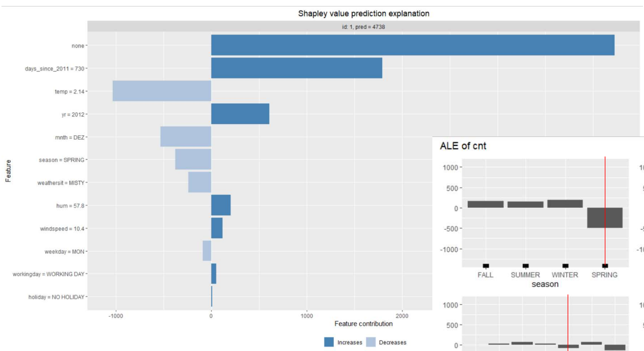
- Since we have both numeric and categorical variables, we use the ctree approach.

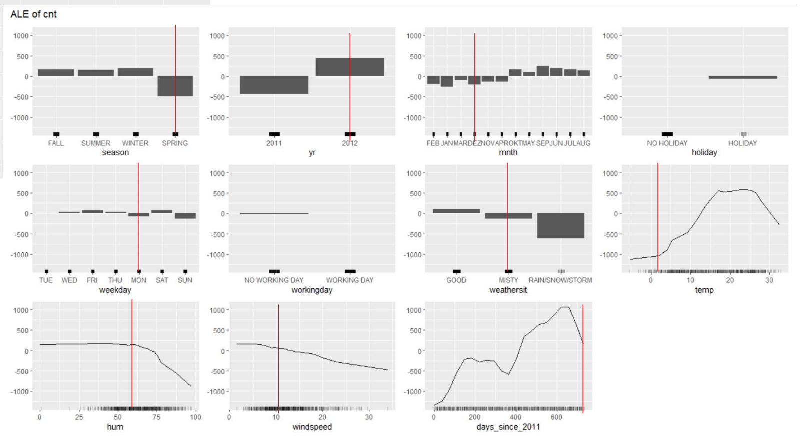
```
library(shapr)
explainer <- shapr(bikeTrain[,-11], model)
p <- mean(bikeTrain[,11])
explain <- shapr::explain(bikeTest[1,],
            explainer,
            approach = "ctree",
            prediction_zero = p,
            mincriterion = 0.95,
            minsplit = 20,
            minbucket = 7,
            sample = TRUE)
print(explain$dt)
if (requireNamespace("ggplot2", quietly = TRUE))
{plot(explain)}
```

```
model<- ranger(cnt ~ ., data = bikeTrain,
num.trees = 50, num.threads = 6,
         verbose = TRUE,
         probability = FALSE,
         importance = "impurity",
         mtry = sqrt(27))
```

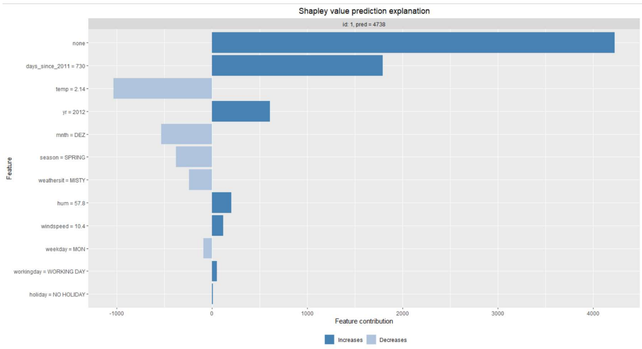


Shapley values ALE plots

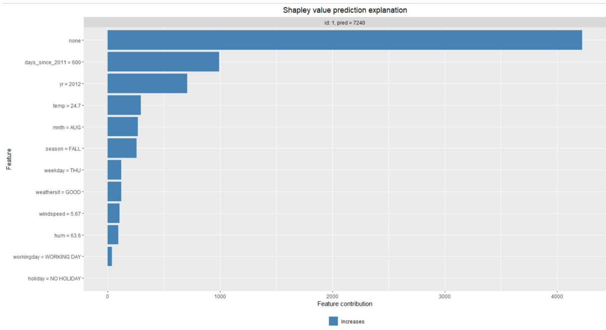




The Shapley values are in good accordance with what we observed in the ALE plot



Low prediction



High prediction

51

Software

- For a python implementation of the kernel SHAP method, see https://github.com/slundberg/shap
- For a python implementation of the fastSHAP approach, see https://github.com/iancovert/fastshap
- The "independence","gaussian","copula","empirical", and "ctree”methods have been implemented in the R-package shapr: https://cran.r-project.org/web/packages/shapr/shapr.pdf
- For a python implementation of the VAEAC approach, see https://github.com/LHBO/ShapleyValuesVAEAC