

Lecture 3. White- and black-box supervised models. Causality

Regression models • Decision trees • Tree ensembles •
Neural networks

Part 1. Causal inference

Studying causality. RCTs vs A/B testing

Randomized controlled trials is a type of scientific experiments where test subject is split into several groups. One or more groups receive randomly the test treatment, while the control group does not. The independent variable that is responsible for the effect should be fully isolated.

“**Why**” does something happen?

Changes in conditions are minimal so that you are able to control variables and only test one (the independent variable).








RCTs are considered to be a gold standard for learning about causal relationships.

A/B testing (aka bucket testing or split testing) compares two variants of a product to see which performs more successfully. These variants are presented to a random sample from a population and can be tested simultaneously across the population.

Which **version** works **best**?

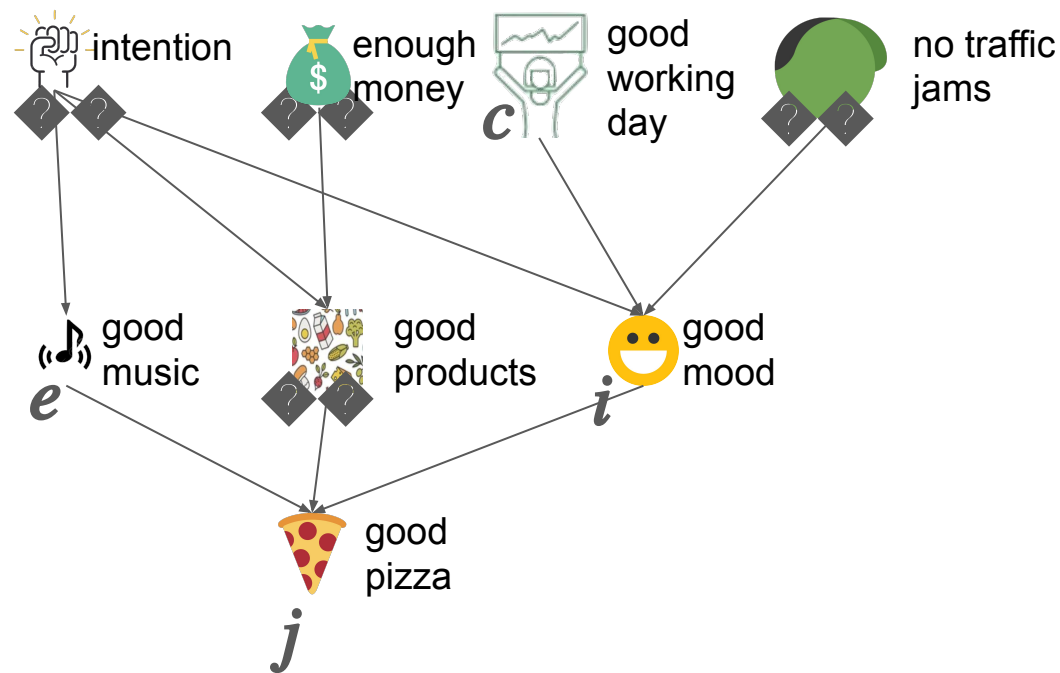
Many different variables can be changed at the same time.

Example

Alex is going to cook a pizza. He has an **intention**  to make his best pizza. He got a very **good working day**  where he got a good **bonus** . Then he quickly got home **without traffic jams** . He had time to buy a very **good products** . When he got home in a **good mood** , he turned on his **favorite music**  and started making pizza.

What is the probability that the pizza will be the **best**  he has ever made?

Causal (DAG) model and structural equation model



$$a = f_1(\varepsilon_1)$$

$$b = f_2(\varepsilon_2)$$

$$c = f_3(\varepsilon_3)$$

$$d = f_4(\varepsilon_4)$$

$$e = f_5(a, \varepsilon_5)$$

$$g = f_6(a, b, \varepsilon_6)$$

$$i = f_7(a, c, d, \varepsilon_7)$$

$$j = f_8(e, g, i, \varepsilon_8)$$

“Intention” is a **confounder** or common cause for “good music”, “good products”, “good mood”

Counterfactuals

What would happen if...?

Alex has made his best pizza, and given that he **in a very good mood** day, and given everything else we know about the circumstances of cooking the pizza, what can we say about the probability to make his best pizza, had he **been in a bad mood**?

Intervention query:

$$P(\text{🍕} \mid \text{do}(\text{😊} = 0))$$

Counterfactuals. Two parallel worlds

observed factual

...	«♪»			
	0	1	1	1
	0	0	0	0
	1	0	1	0
	0	1	0	1
	1	0	1	1

$P(A, B, C, D, E, G, I, J)$



$$a = f_1(\varepsilon_1)$$



$$b = f_2(\varepsilon_2)$$



$$c = f_3(\varepsilon_3)$$



$$d = f_4(\varepsilon_4)$$



$$e = f_5(a, \varepsilon_5)$$



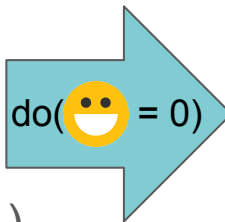
$$g = f_6(a, b, \varepsilon_6)$$



$$i = f_7(a, c, d, \varepsilon_7)$$



$$j = f_8(e, g, i, \varepsilon_8)$$



imagined counterfactual

...	«♪»			
	1	1	0	0
	0	0	0	0
	0	0	0	0
	0	1	0	1
	1	0	0	1

$P(A^*, B^*, C^*, D^*, E^*, G^*, I^*, J^*)$



$$a = f_1(\varepsilon_1)$$



$$b = f_2(\varepsilon_2)$$



$$c = f_3(\varepsilon_3)$$



$$d = f_4(\varepsilon_4)$$



$$e = f_5(a, \varepsilon_5)$$



$$g = f_6(a, b, \varepsilon_6)$$



$$i = 0$$

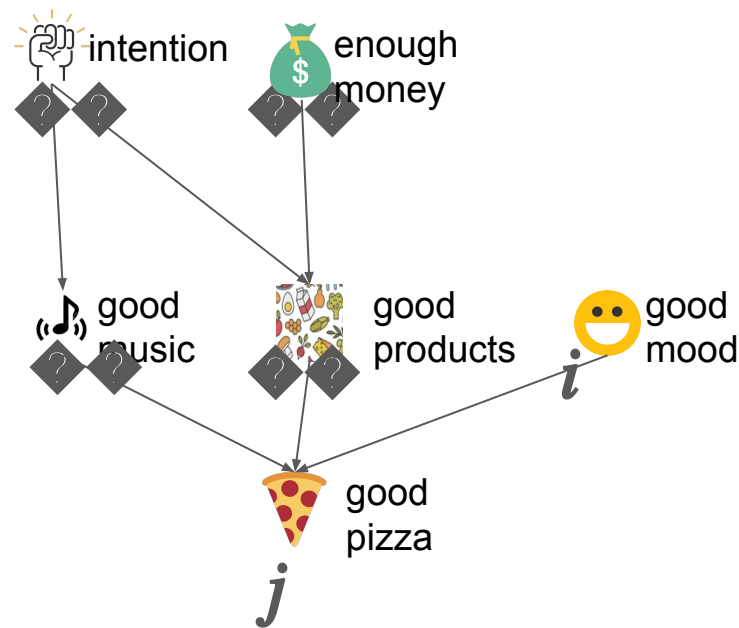
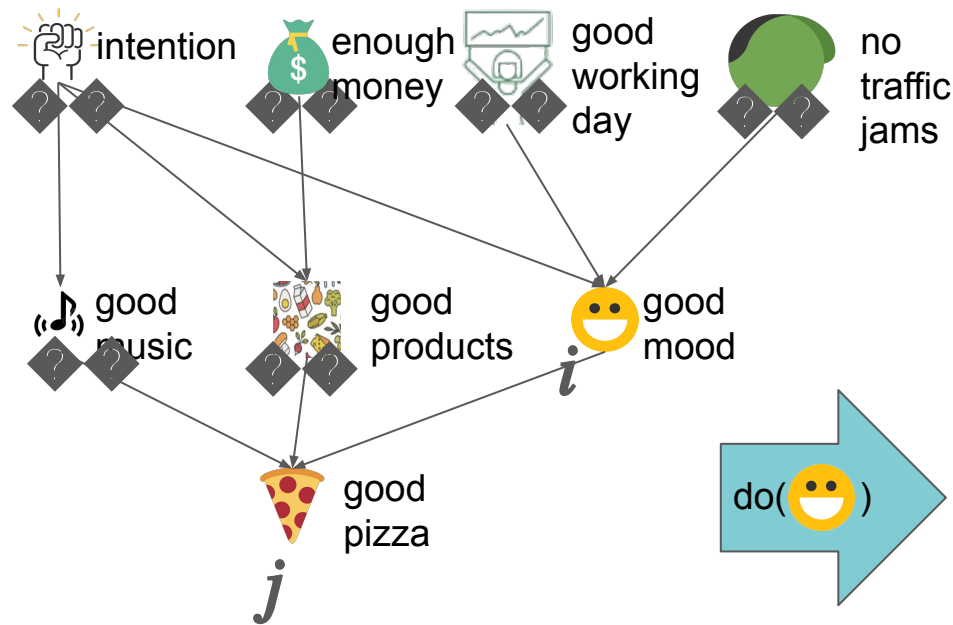


$$j = f_8(e, g, i, \varepsilon_8)$$

$P(j^* | I^* = \hat{I}, A=a, B=b, C=c, D=d, E=e, G=g, I=i, J=j)$ which is contrafactual prediction

Causal and intervention models

$$P(\text{🍕} \mid \text{do}(\text{😊} = 0))?$$

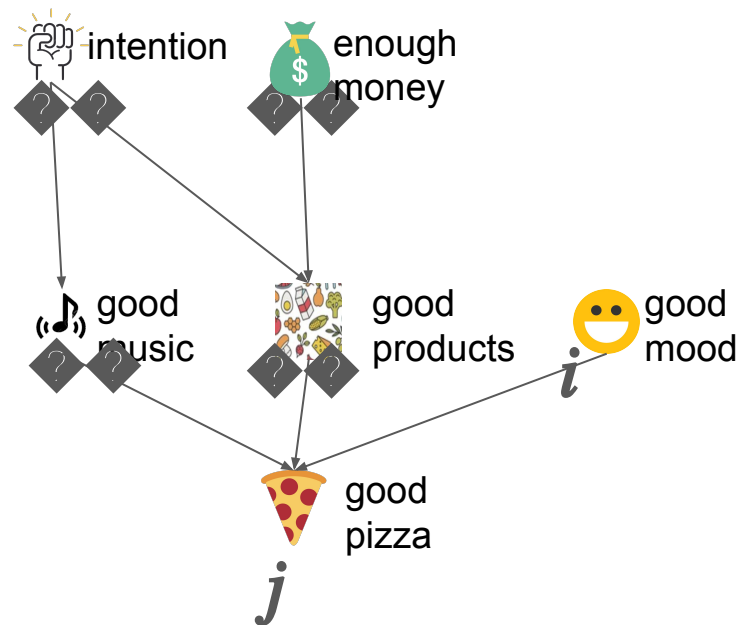


“Intention” is a **confounder** or common cause for “good music”, “good products”, “good mood”

Causal effect difference (average causal effect)

$$P(Y \mid \text{do}(X = 1)) - P(Y \mid \text{do}(X = 0))$$

$$P(\text{🍕} \mid \text{do}(\text{😊} = 1)) - P(\text{🍕} \mid \text{do}(\text{😊} = 0))$$



Causal models. Main components

Chains

1. A and B are dependent: $P(B|A) \neq P(B)$
2. B and C are dependent: $P(C|B) \neq P(C)$
3. A and C are likely dependent: $P(C|A) \neq P(C)$
4. A and C are independent conditional on B: $P(C|A,B) = P(C|B)$

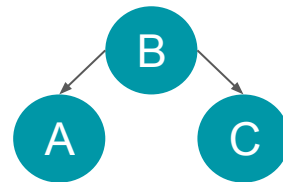


Conditional independence on chains: two variables, A and C, are conditionally independent given B, if there is only **one unidirectional path between** A and C and B is any set of variables that intercepts that path.

Causal models. Main components

Forks

1. A and B are dependent: $P(A|B) \neq P(A)$
2. C and B are dependent: $P(B|C) \neq P(C)$
3. A and C are likely dependent: $P(A|C) \neq P(A)$, $P(C|A) \neq P(C)$
4. A and C are independent, conditional on B: $P(A|B,C) = P(A|B)$, $P(C|B,A) = P(C|B)$

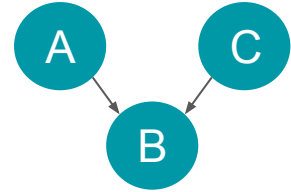


Conditional independence in forks: if a variable B is a common cause of variables A and C, and there is only **one path** between A and C, then A and C are independent conditional on B.

Causal models. Main components

Colliders

1. A and B are dependent: $P(B|A) \neq P(B)$
2. A and B are dependent: $P(B|A) \neq P(B)$
3. A and C are independent: $P(A|C) = P(A)$
4. A and C are dependent conditional on B: $P(A|B,C) \neq P(A|B)$



Conditional independence in colliders: if a variable B is the collision node between two variables A and C, and there is only **one path** between A and C, then A and C are unconditionally independent but are dependent conditional on B and any descendants of B.

More on dependencies

Note that for the given dependency $A \rightarrow C$, not only $P(C|A) \neq P(C)$, but also $P(A|C) \neq P(A)$

Why so?

According to the definition of conditional probability: $P(C|A) = P(AC) / P(A)$

Thus, $P(AC) = P(C|A)P(A) = P(A|C)P(C)$ (*)

Form the other hand, knowing that $P(C|A) \neq P(C)$, it follows that $P(AC) \neq P(A)P(C)$, putting it into (*) we get $P(AC) = P(A|C)P(C) \neq P(A)P(C)$

Thus, $P(A|C) \neq P(A)$

That is why, causal relations are not the same that correlation



Dependency identification. Do-calculus.

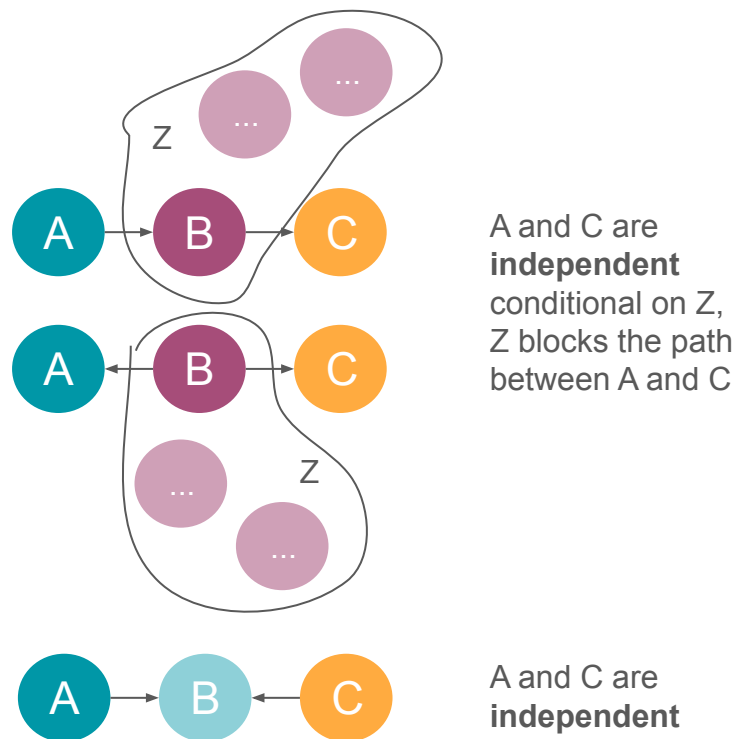
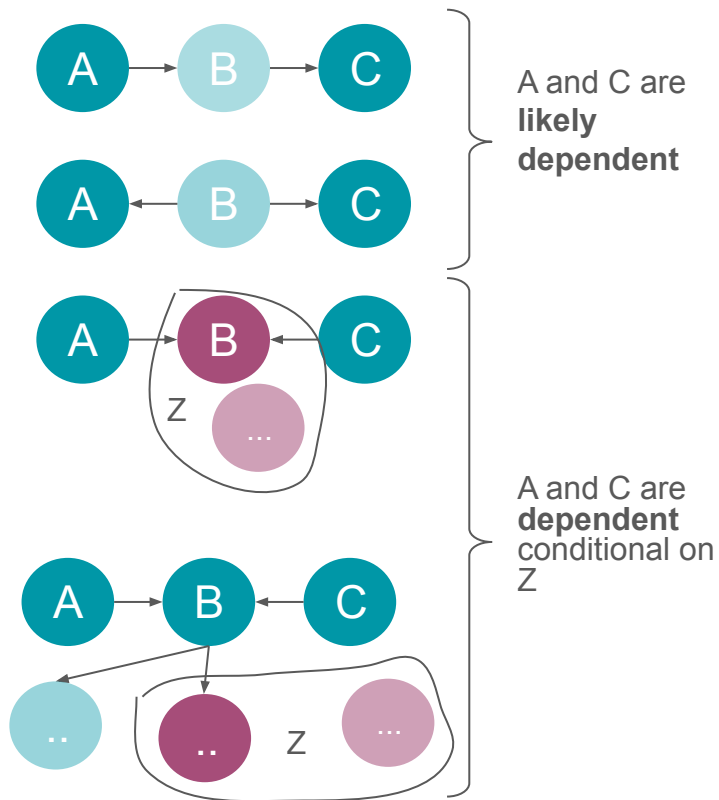
d-separation: A path p is blocked by a set of nodes Z , if and only if:

- p contains a chain of nodes $A \rightarrow B \rightarrow C$, or a fork $A \leftarrow B \rightarrow C$ such that the middle node B is in Z , or
- p is a collider $A \rightarrow B \leftarrow C$ such that the collision node B is not in Z , and no descendent of B is in Z .

If Z blocks every path between two nodes A and C , then A and C are d-separated, conditional on Z , and thus are independent conditional on Z

Any two nodes A , and B that are d-separated conditional on Z are necessarily **independent conditional on Z**

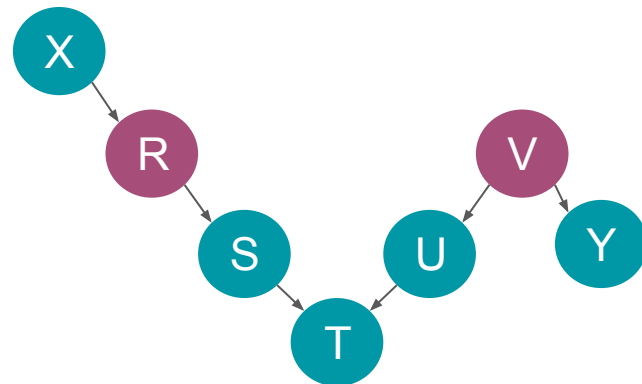
D-separation. Summary



Example

Which pairs of variables are independent conditional on the set $Z=\{R, V\}$?

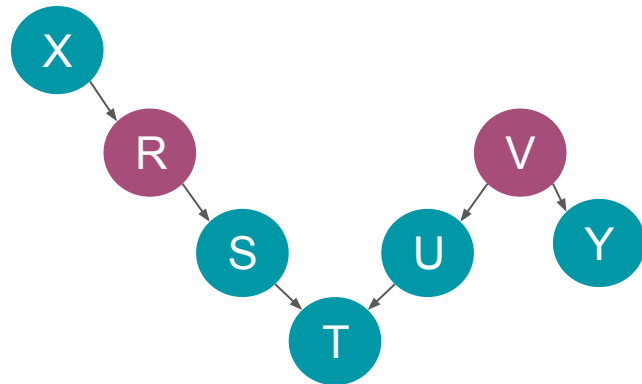
- (S, T)
- (U, T)
- (X, S)
- (X, T)
- (X, Y)
- (X, U)
- (S, Y)
- (U, Y)
- (T, Y)
- (S, U)



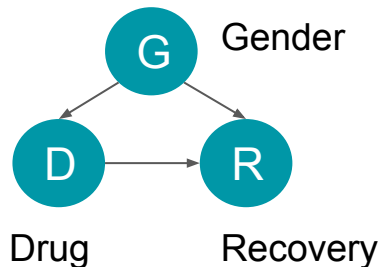
Example

Which pairs of variables are independent conditional on the set $Z=\{R, V\}$?

- (S, T) , (U, T) are **dependent** conditional on R and V
- (X, S) , (X, T) are **independent** conditional on R
- (X, Y) , (X, U) and (S, Y) are **independent** (since R and V are blocking the only path away from X and to Y , then the pairs)
- (U, Y) and (T, Y) are **independent** conditional on V
- (S, U) are **independent**



Computing post-intervention distribution

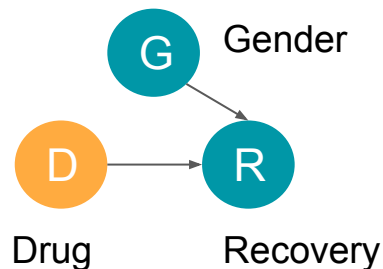


Joint distribution

$$P(G,D,R) = P(G)P(D|G)P(R|G,D)$$

The post-intervention distribution

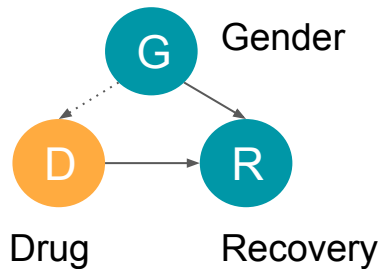
$$P(G,R|\text{do}(D = \mathbf{d}_0)) = P(G) \cancel{P(D|G)} P(R|G, D = \mathbf{d}_0)$$



Causal effects of D on R is computed by marginalizing over the remaining variables:

$$P(R|\text{do}(D = \mathbf{d}_0)) = \sum_g P(G = g) \cancel{P(D|G)} P(R|G = g, D = \mathbf{d}_0)$$

Example

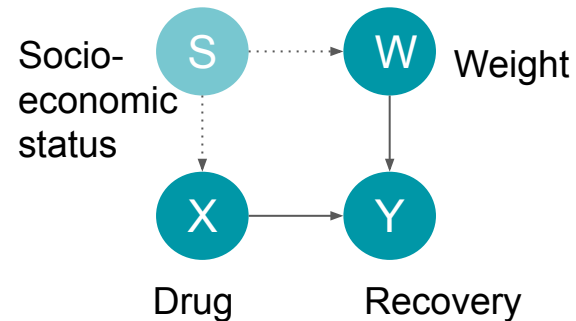


recovery	drug	no drug	total
men	111/ 130 (85%)	206/ 298 (69%)	• /428
women	217/ 270 (80%)	61/ 102 (60%)	• /372
total	• /400	• /400	• / 800

$$P(\text{Recovery} \mid \text{do}(\text{Drug} = 1)) = P(\text{Recovery} \mid d, m)P(m) + P(\text{Recovery} \mid d, f)P(f)$$

$$P(\text{Recovery} \mid \text{do}(\text{Drug} = 1)) = 0.85 (130+298)/800 + 0.80 (270+102)/800$$

Backdoor criterion. Motivation example

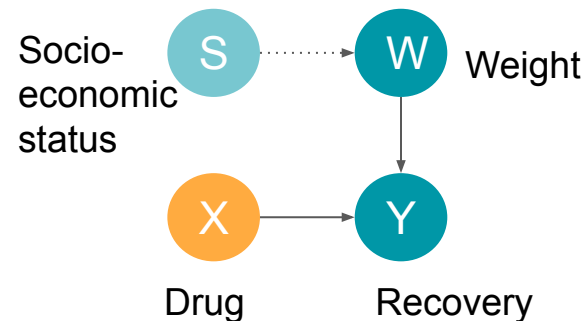


Joint distribution

$$P(X, Y, W, S) = P(S)P(W|S)P(X|S)P(Y|X, W)$$

The post-intervention distribution

$$P(Y, W, S | \text{do}(X = \mathbf{x}_0)) = P(S)P(W|S)\mathbf{P(X|S)}P(Y|X = \mathbf{x}_0, W)$$



Causal effects of X on Y:

$$P(Y | \text{do}(X = \mathbf{x}_0)) = \sum_{w,z} P(S=s)P(W=w|S=s)P(Y|X=\mathbf{x}_0, W=w)$$

What if Z is unmeasurable? Could we use a simpler conditional distribution?

Backdoor criterion

Given an ordered pair of variables (X, Y) in a directed acyclic graph G , a set of variables Z satisfies the **backdoor criterion** relative to (X, Y) if

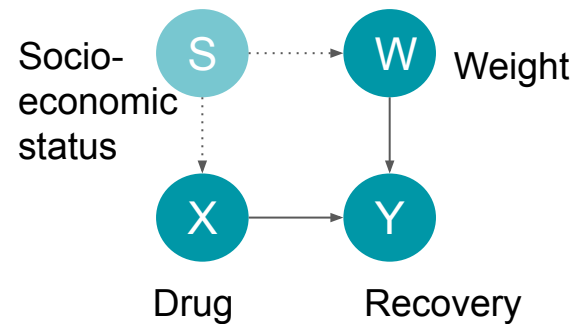
1. no node in Z is a **descendant** of X , and
2. Z blocks every path between X and Y that contains an **arrow into** X .

We measure the direct effect that X has on Y by isolating the effect from any other spurious correlations that might be present, i.e., to make sure all non-causal paths between X and Y are blocked off

Backdoor adjustment: If a set of variable Z satisfy a backdoor criterion relative to (X, Y) , then the causal effect of X on Y is identifiable and is given by the formula

$$P(Y|do(X = x_0)) = \sum_z P(Y|X = x_0, Z = z)P(Z = z)$$

Motivation example



The causal path between X and Y: $X \rightarrow Y$

The backdoor path between X and Y: $X \leftarrow S \rightarrow W \rightarrow Y$

Using the d-separation rule it follows that

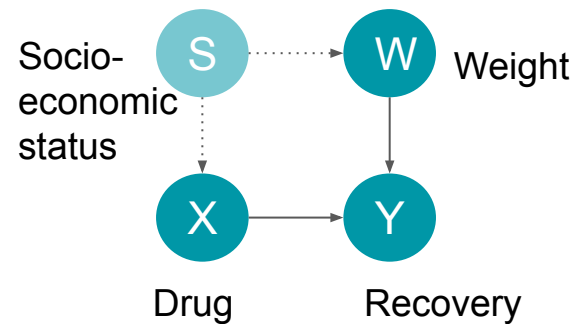
- S **blocks** the path between X and Y,
- W **blocks** the path between X and Y
- S and W **block** the path between X and Y

Thus, $P(Y | \text{do}(X=x)) = P(Y | X=x, W)P(W)$

Which is equivalent to

$$P(Y = y | \text{do}(X = \mathbf{x})) = \sum_w P(Y = y | X = \mathbf{x}, W = w)P(W = w)$$

Motivation example



probability to be recovered	weight			total
	underweighted uw	normal weight nw	overweighted ow	
drug	40/80	200/220	30/300	-/600
no drug	40/130	50/90	4/380	-/600

How to compute some probabilities

From the table we see that the total number of observation is 1200 (600 taking drugs and 600 without drugs)

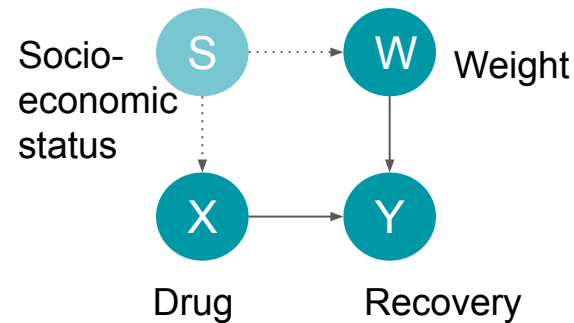
$P(\text{recovery}|\text{uw}, \text{drug}) = 40/80$ (the same for the remaining weight categories)

do not mix with $P(\text{recovery}, \text{uw}, \text{drug}) = 40/1200$, $P(\text{uw}, \text{drug}) = 80/1200$, thus $P(\text{recovery}|\text{uw}, \text{drug}) = P(\text{recovery}, \text{uw}, \text{drug}) / P(\text{uw}, \text{drug}) = 40/80$

$P(\text{uw}) = (80 + 130)/1200$

we simply compute in the numerator the number of the observation that are “underweighted” and taken “drug” (80, among them 40 are recovered) and the number of the observation that are “underweighted” and do not taken “drug” (130, among them 40 are recovered), in the denominator we take the total number of observation, i.e., 1200

Motivation example



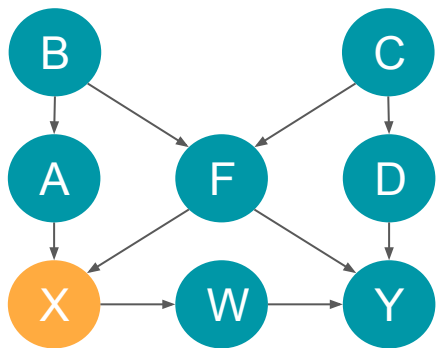
recovered	weight			total
	underweighted uw	normal weight nw	overweighted ow	
drug	40/80	200/220	30/300	-/600
no drug	40/130	50/90	4/380	-/600

Without backdoor adjustment $P(\text{recovery}|\text{do}(\text{drug})) = (40+200+30)/(80+220+300) = 270/600 = 0.45$

Applying the backdoor adjustment $P(\text{recovery} | \text{do}(\text{drug})) = \sum_{w \in \{uw, nw, ow\}} P(\text{recovery} | \text{drug}, W = w)P(W = w) = 40/80 \cdot (80+130)/1200 + 200/220 \cdot (220+90)/1200 + 30/300 \cdot (300+380)/1200 = 0.38$

Thus, W may affect both X and Y and without taking into account it, we slightly overestimate the effect of treatment

One more exercise



Joint distribution

$$P(B, C, A, F, D, X, W, Y) =$$

$$P(B)P(C)P(A|B)P(F|B, C)P(D|C)P(X|A, F)P(W|X)P(Y|W, F, D)$$

The post-intervention distribution

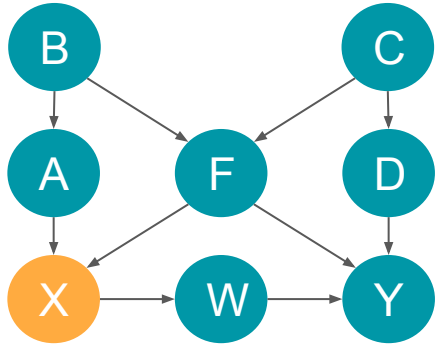
$$P(B, C, A, F, D, W, Y | \text{do}(X = \mathbf{x}_0)) =$$

$$P(B)P(C)P(A|B)P(F|B, C)P(D|C)\cancel{P(X|A, F)}P(W|\mathbf{x}_0)P(Y|W, F, D)$$

Causal effects of X on Y is computed by marginalizing over the remaining variables:

$$P(Y | \text{do}(X = \mathbf{x}_0)) = \sum_{a, b, c, d, f, w} P(b)P(c)P(a|b)P(f|b, c)P(d|c)P(w|\mathbf{x}_0)P(Y|w, f, d)$$

Example. Estimating the effect of X on Y

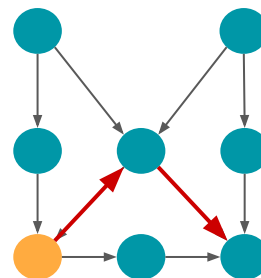
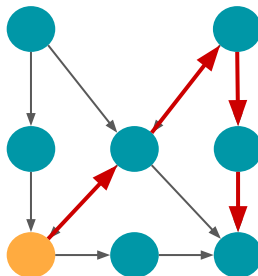
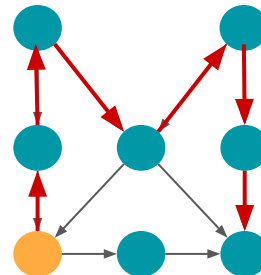
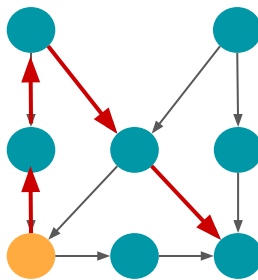
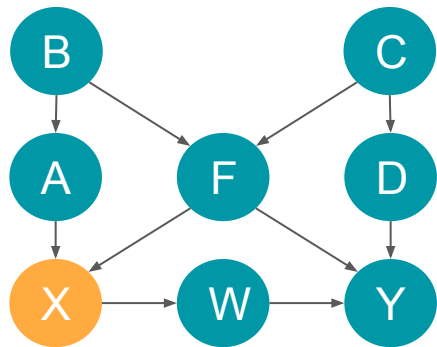


To apply the backdoor criterion we apply the following steps:

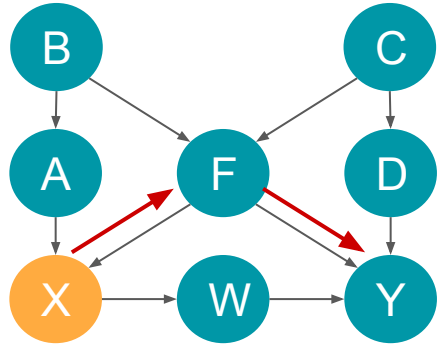
1. Find all possible backdoor paths from X to Y
2. Block them all
3. Apply the formula of the backdoor adjustment

Example. Estimating the effect of X on Y

1. Find all possible backdoor paths:

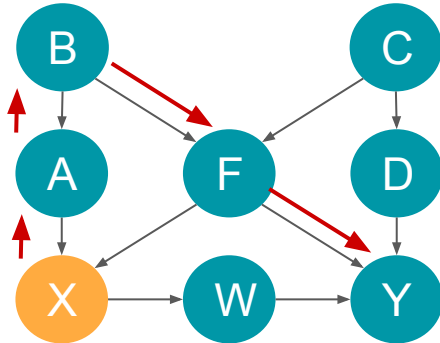


Example. Estimating the effect of X on Y



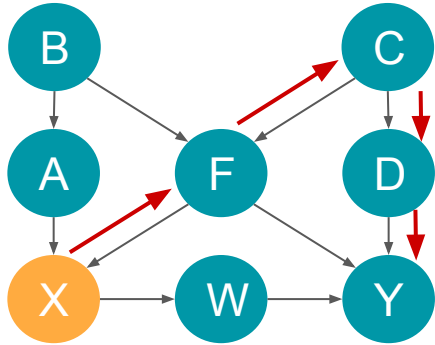
2. Start blocking them starting from the smallest one

- the path should be conditional on F

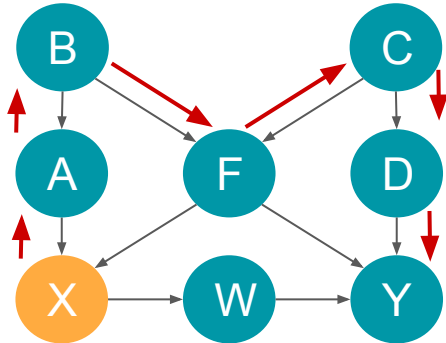


- being conditional on F, the path is blocked, since B and Y are independent conditional on F

Example. Estimating the effect of X on Y



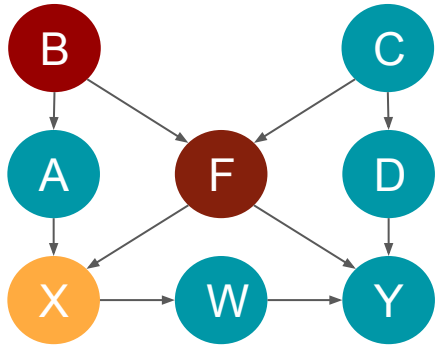
- the path is blocked being conditional on F, since the path between X and C is blocked conditional on F



- being conditional on F, the path is unblocked, since B and C are dependent conditional on F, thus we must also condition on one of (A, B, C, or D).

Z in addition to any combinations of these 4 nodes will fulfill the back-door criteria. So we could condition on (F, A), (F, B), (F, A, C), (F, A, B, C, D), etc.

Example. Estimating the effect of X on Y



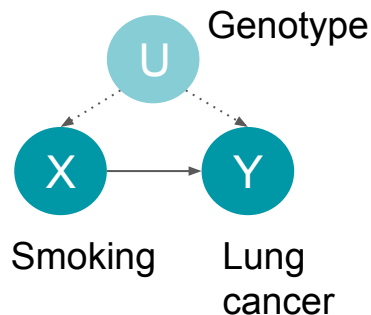
3. Applying the backdoor adjustment formula.

As a set Z we take $Z = \{F, B\}$, then

$$P(Y|\text{do}(X = \mathbf{x}_0)) = \sum_{f,b} P(Y|X = \mathbf{x}_0, F=f, B=b)P(F=f, B=b)$$

where we sum up over all pairs of values (f,b)

Front-door criterion. Motivation example



	smoker	non-smoker
no cancer	341	39
cancer	59	361

$P(\text{no cancer} \mid \text{smokers}) = 85\%$

$P(\text{no cancer} \mid \text{non-smokers}) = 9.75\%$

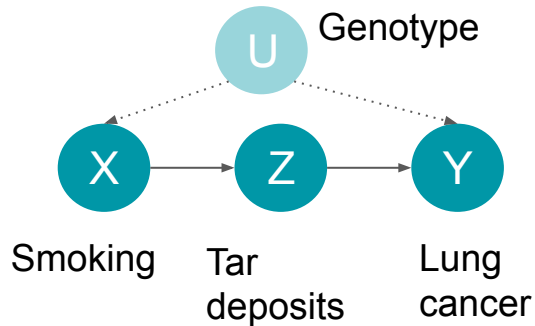
$P(\text{cancer} \mid \text{smokers}) = 15\%$

$P(\text{cancer} \mid \text{non-smokers}) = 90.25\%$



Is it better to smoke in order to avoid lung cancer?

Front-door criterion. Motivation example



	Tar 400		No tar 400		All subjects 800	
	Smokers	Nonsmokers	Smokers	Nonsmokers	Smokers	Nonsmokers
No cancer	380 323 (85%)	20 1 (5%)	20 18 (90%)	380 38 (10%)	400 341 (85%)	400 39 (9.75%)
Cancer	57 (15%)	19 (95%)	2 (10%)	342 (90%)	59 (15%)	361 (90.25%)

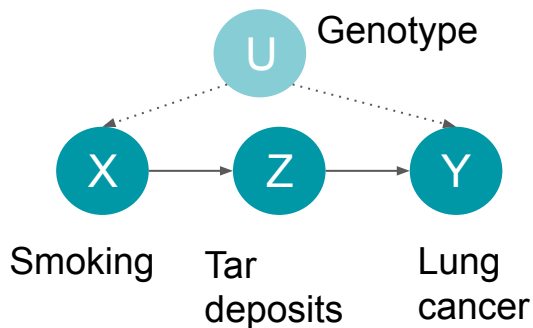
$$P(\text{no cancer} \mid \text{smokers, tar}) = 85\%$$

$$P(\text{no cancer} \mid \text{non-smokers, tar}) = 5\%$$

$$P(\text{cancer} \mid \text{smokers, tar}) = 15\%$$

$$P(\text{cancer} \mid \text{non-smokers, tar}) = 95\%$$

Front-door criterion. Motivation example



We need to compute $P(\text{cancer}|\text{do}(\text{smoking}))$? Here, the backdoor criterion is inapplicable since U is unobservable.

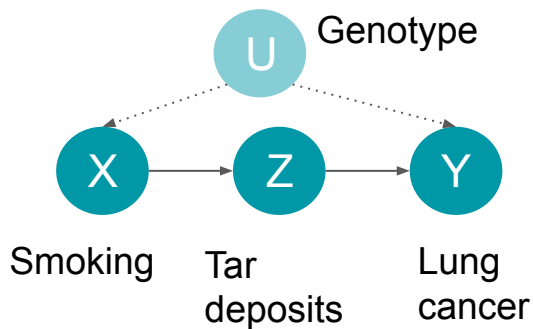
Frontdoor criterion can be applied in this case, by using the front-door paths.

A **front-door path** between X and Y is a path containing an edge that outcome from X .

A front-door criterion is a consecutive combination of the backdoor criteria. Thus, in our case we do the following:

1. Apply the backdoor criterion to $X \rightarrow Z$, i.e., to compute $P(Z|\text{do}(X=x_0))$:
 - a. The only backdoor path $Z \rightarrow Y \leftarrow U \rightarrow X$ is blocked since it contains a collider, thus, we have nothing to add to Z
 - b. Since the set Z is empty, then $P(Z = z|\text{do}(X = x)) = P(Z = z|X = x)$ (*)
2. Apply the backdoor criterion to $Z \rightarrow Y$:
 - a. The only backdoor path $Y \leftarrow U \rightarrow X \rightarrow Z$ is unblocked. It can be blocked by conditioning on X , i.e. $Z = \{X\}$
 - b. Applying the backdoor adjustment, we get $P(Y = y|\text{do}(Z = z)) = \sum_x P(Y = y|Z = z, X = x)P(X = x)$ (**)

Front-door criterion. Motivation example



Form previous slide:

Application the backdoor adjustment to $X \rightarrow Z$ we get

$$P(Z = z | \text{do}(X = x)) = P(Z = z | X = x) (*)$$

Application it to $Z \rightarrow Y$ gives us

$$P(Y = y | \text{do}(Z = z)) = \sum_{x'} P(Y = y | Z = z, X = x') P(X = x') (**)$$

Thus, we combine two consecutive backdoor adjustment:

$$P(Y = y | \text{do}(X = x)) = \sum_z P(Z = z | \text{do}(X = x)) P(Y = y | \text{do}(Z = z))$$

Incorporating the formulas (*) and (**) we get:

$$P(Y = y | \text{do}(X = x)) = \sum_z P(Z = z | X = x) \sum_{x'} P(Y = y | Z = z, X = x') P(X = x')$$

Front-door criterion

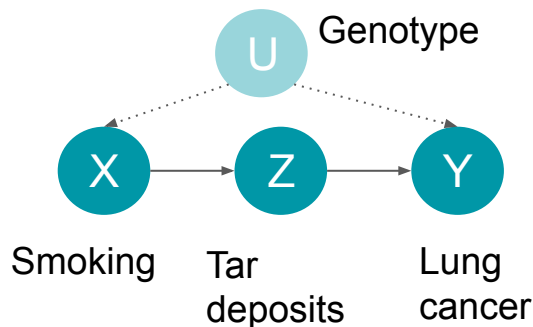
A set of variables Z is said to satisfy the front-door criterion relative to an ordered pair of variables (X, Y) , if:

1. Z intercepts all directed paths from X to Y
2. there is no unblocked backdoor path from X to Z
3. all backdoor paths from Z to Y are blocked by X

If Z satisfies the front-door criterion relative to (X, Y) and if $P(x, z) > 0$, then the causal effect of X on Y is identifiable and is given by:

$$P(Y|\text{do}(X)) = \sum_Z P(Z|X) \sum_{X'} P(Y|X', Z) P(X')$$

Front-door criterion. Motivation example

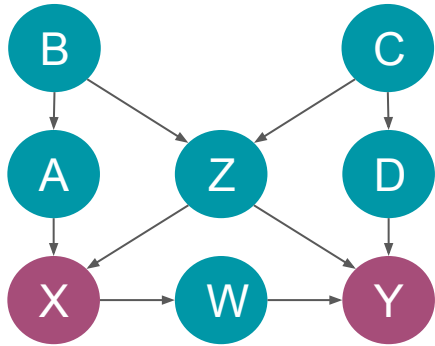


	Tar 400		No tar 400		All subjects 800	
	Smokers	Nonsmokers	Smokers	Nonsmokers	Smokers	Nonsmokers
No cancer	380 323 (85%)	20 1 (5%)	20 18 (90%)	380 38 (10%)	400 341 (85%)	400 39 (9.75%)
Cancer	57 (15%)	19 (95%)	2 (10%)	342 (90%)	59 (15%)	361 (90.25%)

$$\begin{aligned}
 P(\text{cancer} \mid \text{smokers}) &= \sum_{z \in \{\text{tar, no tar}\}} P(Z = z \mid \text{smokers}) \sum_{x \in \{\text{non-smokers, smokers}\}} P(\text{cancer} \mid Z = z, X = x) P(X = x) \\
 &= P(t \mid \mathbf{s}) [P(c \mid t, \text{ns}) P(\text{ns}) + P(c \mid t, \text{s}) P(\text{s})] + P(\text{nt} \mid \mathbf{s}) [P(c \mid \text{nt}, \text{ns}) P(\text{ns}) + P(c \mid \text{nt}, \text{s}) P(\text{s})] \\
 &= 380/400 [19/20 \times 400/800 + 57/380 \times 400/800] + 20/400 [342 / 380 \times 400/800 + 2 / 20 \times 400/800] \\
 &= 0.5225 + 0.025 = 0.5475
 \end{aligned}$$

Pay attention that this example is handcrafted just to demonstrate the effect of application of the criterion

Example. Estimating the direct effect of X on Y



Front-door criterion for the causal effect of X on Y:

- W intercepts the only direct path between X and Y
- there are no unblocked backdoor paths between W and X (as they must all pass through the collider at Z)
- all backdoor paths between W and Y are blocked by X

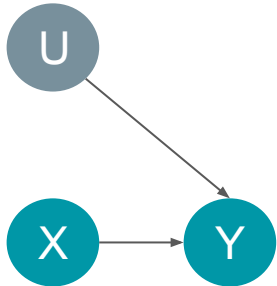
As we had previously seen, estimating the causal effect of X on Y using the back-door criterion requires conditioning on at least 2 variables (Z and B, for example) while the front-door approach requires only W.

Instrumental variables

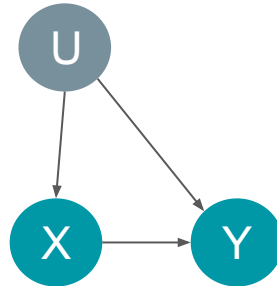
Let us consider the regression problem: $y = X'b + u$

Thus, we assume Case a, however, when Case b, we have $y = X + u(x)$ we get the following estimates: $\frac{dy}{dx} = \beta + \frac{du}{dx}$

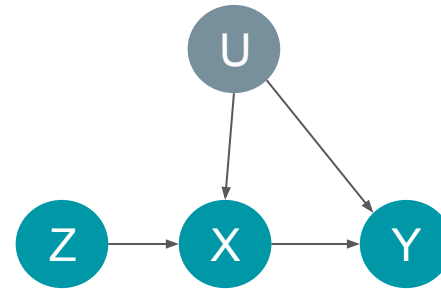
More formally, a variable z is called an instrument or instrumental variable for the regressor x in the scalar regression model $y = x+u$ if (1) z is uncorrelated with the error u ; and (2) z is correlated with the regressor



Case a



Case b: Error-in-variables



Case c: instrumental variable z

Dataset bias and propensity scores

Assessing the effect of treatment, it is important that the groups of instances that have obtained the treatment be quite similar to the group of instances that do not obtain it. In real settings, it is almost impossible. Thus, in practice, the groups are built based on propensity score.

Let T be a treatment, and C be a set of background characteristics (observed baseline covariates*). The propensity score $P(T|C)$ is the probability that an individual with a given set of background characteristics C will be assigned to a particular treatment group (Rosenbaum, 1995; Rubin, 2006)

In practice, the propensity score is most often estimated using a **logistic regression model**, in which T is regressed on C

The estimated propensity score is the predicted probability of treatment derived from the fitted regression model

Dataset bias and propensity scores

treatment	check last 3 months	BMI	age	...	physical activity
1	1	24.3	27	...	1
1	1	31.7	37		3
1	0	18.9	43		5
1	1	21.4	29		3
0	0	32.5	18		0
0	0	20.1	25		4
0	1	25.2	41		1
0	1	17.8	35		2

Dataset bias and propensity scores

treatment	check last 3 months	BMI	age	...	physical activity
1	1	24.3	27	...	1
1	1	31.7	37		3
1	0	18.9	43		5
1	1	21.4	29		3
0	0	32.5	18		0
0	0	20.1	25		4
0	1	25.2	41		1
0	1	17.8	35		2

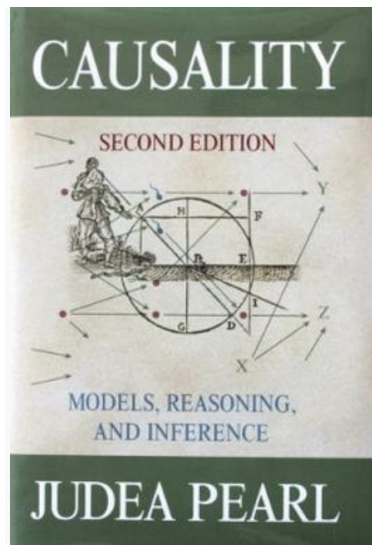
Dataset bias and propensity scores

treatment	check last 3 months	BMI	age	...	physical activity
1	1	24.3	27	...	1
1	1	31.7	37		3
1	0	18.9	43		5
1	1	21.4	29		3
0	0	32.5	18		0
0	0	20.1	25		4
0	1	25.2	41		1
0	1	17.8	35		2

Dataset bias and propensity scores

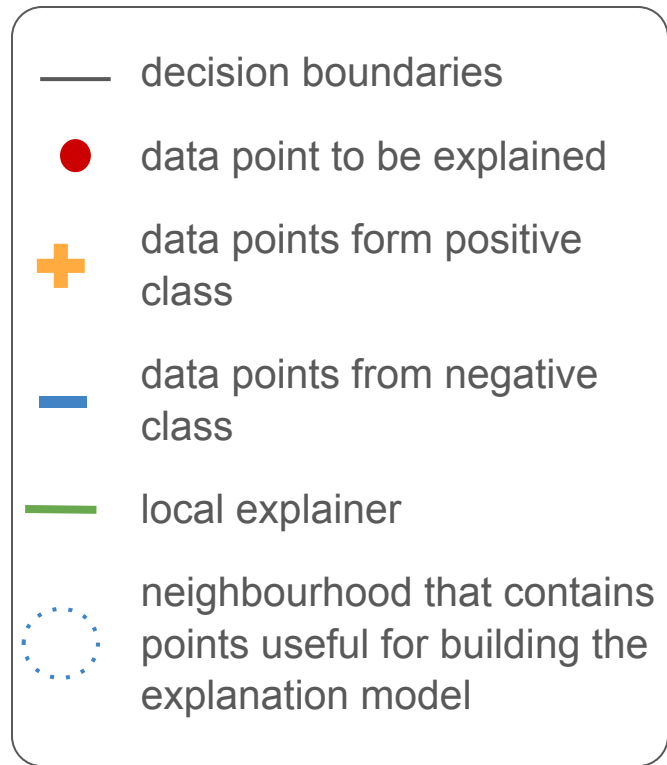
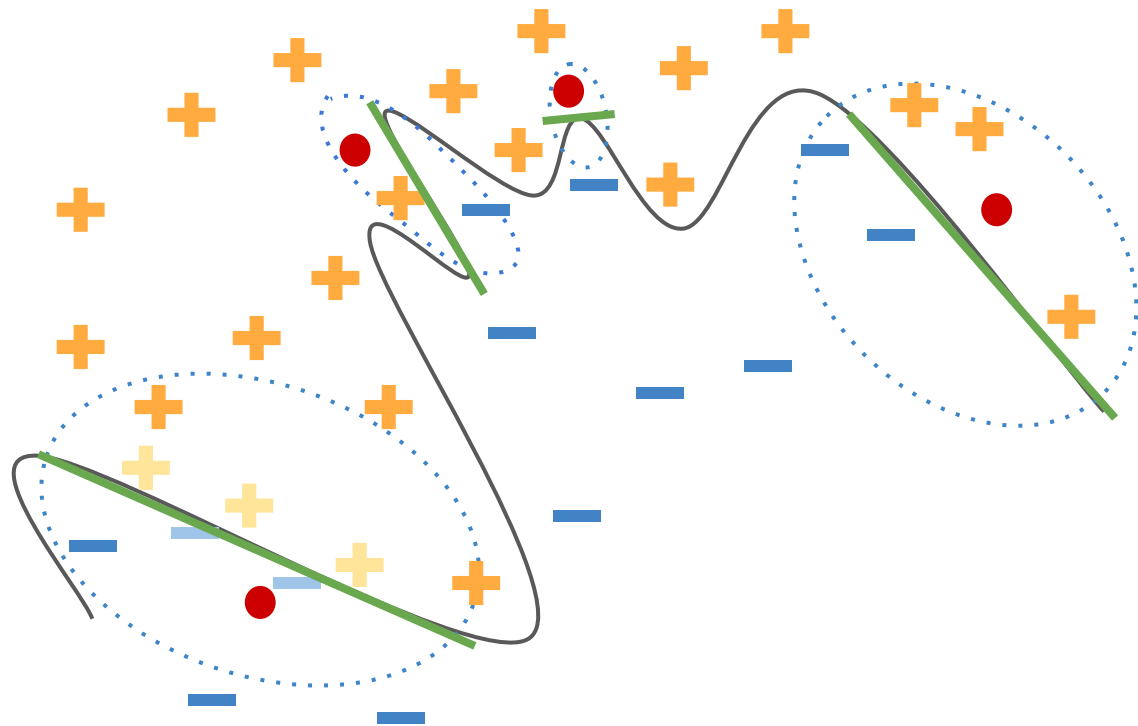
treatment	check last 3 months	BMI	age	...	physical activity	propensity scores
1	1	24.3	27	...	1	0.71
1	1	31.7	37		3	0.97
1	0	18.9	43		5	0.91
1	1	21.4	29		3	0.88
0	0	32.5	18		0	0.93
0	0	20.1	25		4	0.75
0	1	25.2	41		1	0.02
0	1	17.8	35		2	0.01

See the details in



Part 2. White-box models in supervised settings

Regression-based explanation



Linear regression

The linear regression model has the form $f(X) = \beta_0 + \sum_{j=1}^p X_j \beta_j$

One of the most popular method to estimates the coefficients is to minimize the residual sum of squares, i.e.,
$$\text{RSS}(\beta) = \sum_{i=1}^N (y_i - f(x_i))^2$$

Notation: X – matrix, X_j – j -th column, x_i – i -th row, N is the number of rows, and p is the number of columns

Multicollinearity and its consequences

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2$$

Collinearity: $X_1 = kX_2$

Equivalent expression: $f(X) = \beta_0 + (\beta_1 + \beta^*)X_1 + (\beta_2 - \beta^*k)X_2$

Regularized linear regression

Ridge regression penalizes by the sum-of-squares (aka “weight decay” in neural nets)

$$f(X) = \beta_0 + \sum_{j=1}^p X_j \beta_j + \lambda \sum_{j=1}^p \beta_j^2$$

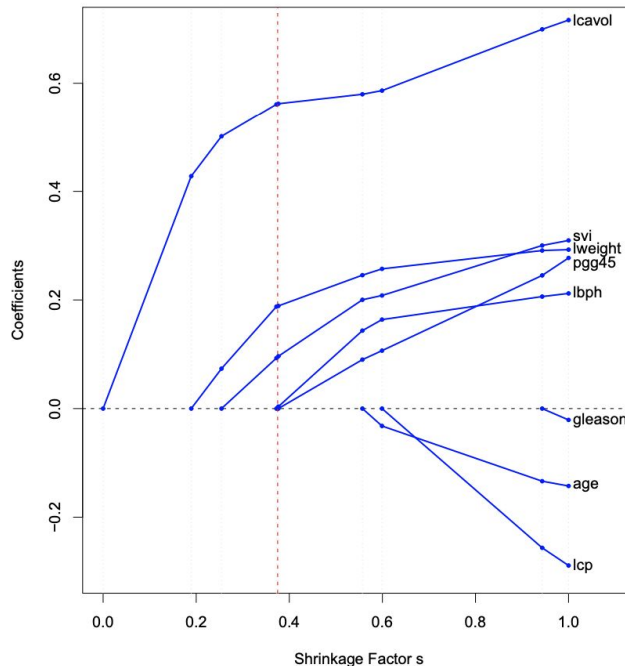
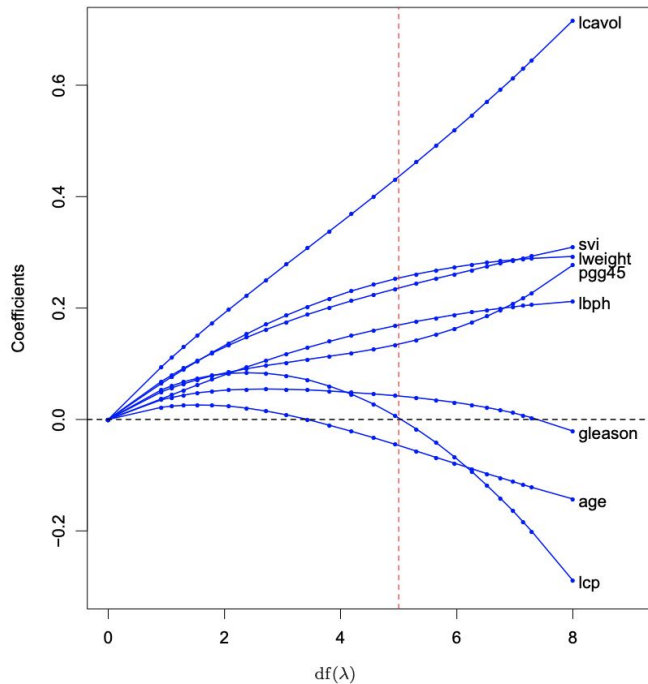
Lasso regression penalizes by the sum-of-absolute-values

$$f(X) = \beta_0 + \sum_{j=1}^p X_j \beta_j + \lambda \sum_{j=1}^p |\beta_j|$$

Elastic net

$$f(X) = \beta_0 + \sum_{j=1}^p X_j \beta_j + \lambda \sum_{j=1}^p (\alpha \beta_j^2 + (1 - \alpha) |\beta_j|)$$

Ridge vs lasso



Left: ridge regression
shrinks the coefficients

Right: lasso regression
ensures feature
selection

Regularized regression

It helps to tackle overfitting, i.e., the situations when

- the number of attributes too large w.r.t. the number of objects
- some attributes are linearly dependent
- some attributes are non-informative

As an explanation model, linear regression can be applied to **any type** of black-box models since it approximate the true function in a neighbourhood of a certain object (instance/observation/example)

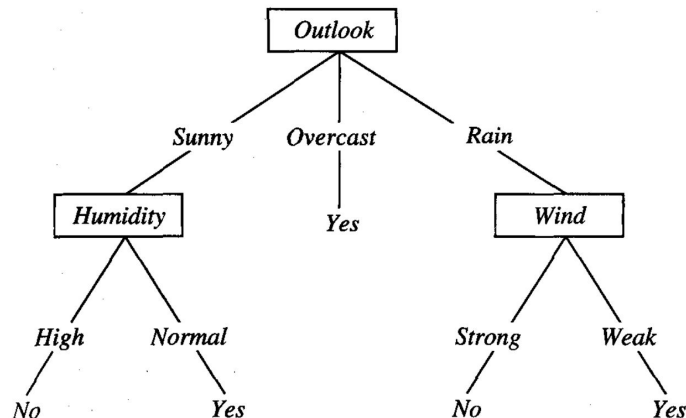
Further reading on regression models

- Best-subset selection
 - Forward-stepwise selection
 - Backward-stepwise selection
 - Forward-stagewise selection
- Ridge, lasso, and its generalization
- Elastic net
- Least angle regression
- Principal components regression
- Partial least squares

Decision trees

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision tree:



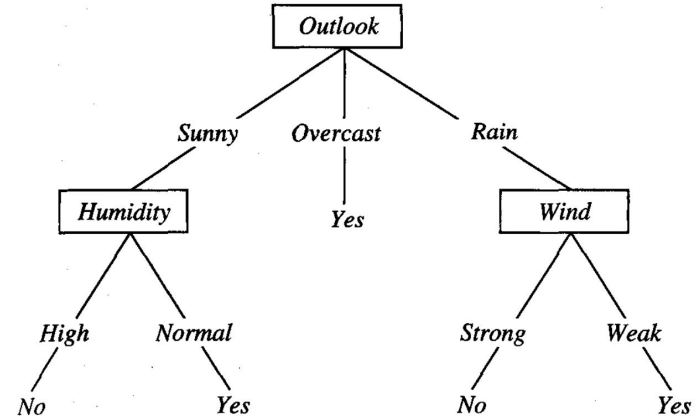
Each branch of the tree can be represented as a **rule**, e.g.,
IF "Outlook = sunny" **AND** "Humidity = high" **THEN** **NO**

The **number of rules** is equal to the **number of leafs**

Decision trees vs rule-based approaches

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision tree:



Equivalent rule list: IF "Outlook = sunny" AND "Humidity = high" THEN **NO**
ELSE IF "Outlook = rain" AND "Wind = strong" THEN **NO**
ELSE **YES**

They take into account interaction between features unlike regression models

Rule-based explanation models

The rule-based models on are often used to explain ensembles of trees

The idea consists in the following steps

1. to **extract all rules** from the trees, and then
2. to select the most **important** ones

Thus, often, rule-based explanation models are model-specific explainers, since they use the rule-like structure of trees

Examples of explainers:

- **RuleFit**
- **Skopec-rules**

RuleFit: explaining tree ensembles

Main steps:

- extract rules from decision trees
- define non-linear features based on the rules
- fit lasso regression on input features and the extracted ones

RuleFit. Step 1: extracting rules

An ensemble of trees has the form

$$f(x) = a_0 + \sum_{m=1}^M a_m f_m(X)$$

The extracted rules are the following

$$r_m(x) = \prod_{j \in T_m} I(x_j \in s_{jm})$$

if T_m is a set of features used in the m -th tree.

The number of rules extracted from M rules is

$$K = \sum_{m=1}^M 2(t_m - 1)$$

RuleFit. Step 2: building new features

- Selection of the best (or all) rules (by precision / recall)
- “Winsorisation” and normalization of the original attributes¹

1. <https://en.wikipedia.org/wiki/Winsorizing>

RuleFit. Step 3: fitting a model

A sparse linear model is given by

$$\hat{f}(x) = \hat{\beta}_0 + \sum_{k=1}^K \hat{\alpha}_k r_k(x) + \sum_{j=1}^p \hat{\beta}_j l_j(x_j)$$

where r_k is the k -th rule, and l_j is the j -th attribute

Lasso-based cost function

$$(\{\hat{\alpha}\}_1^K, \{\hat{\beta}\}_0^p) = \underset{\{\hat{\alpha}\}_1^K, \{\hat{\beta}\}_0^p}{\operatorname{argmin}} \sum_{i=1}^n L(y^{(i)}, f(x^{(i)})) + \lambda \cdot \left(\sum_{k=1}^K |\alpha_k| + \sum_{j=1}^p |b_j| \right)$$

Additionally we may estimate the feature importance

RuleFit. Example

1. Extracted rules:

IF “Outlook = sunny” AND “Humidity = high” THEN **NO**

IF “Outlook = sunny” AND “Humidity = normal” THEN **YES**

IF “Outlook = overcast” THEN **YES**

IF “Outlook = rain” AND “Wind = strong” THEN **NO**

IF “Outlook = rain” AND “Wind = weak” THEN **YES**

2. Selection of the best rules (precision = 1)

recall

3/5 **+**

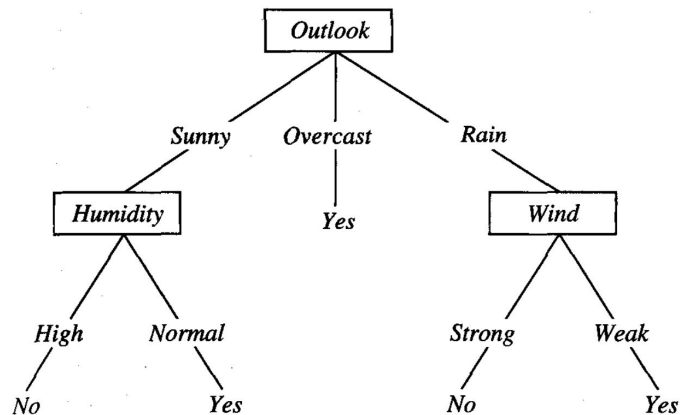
2/9

4/9 **+**

2/5 **+**

3/9

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No



RuleFit. Example

Augment the dataset with 3 selected rules:

Rule 1: IF “Outlook = sunny” AND “Humidity = high” THEN **NO**

Rule 2: IF “Outlook = overcast” THEN **YES**

Rule 3: IF “Outlook = rain” AND “Wind = strong” THEN **NO**

Then train the lasso regression based on the original attributes

	rule 1	rule 2	rule 3
D1	1		
D2	1		
D3		1	
D4			
D5			
D6			1
D7		1	
D8	1		
D9			
D10			
D11			
D12		1	
D13		1	
D14			1

RuleFits

Pros:

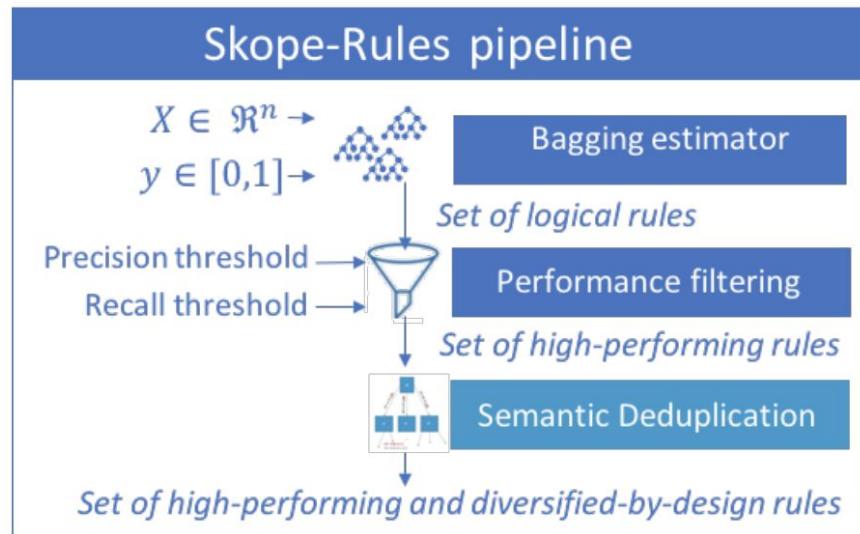
- takes into account feature interactions
- provides a simple interpretation
- supports feature importance, post-hoc analysis (PDP, ALE, ICE)

Cons:

- the interpretation becomes more complicated as more weights are non-zero
- may be an non-intuitive interpretation in case of overlapped rules
- performance may be not very high...

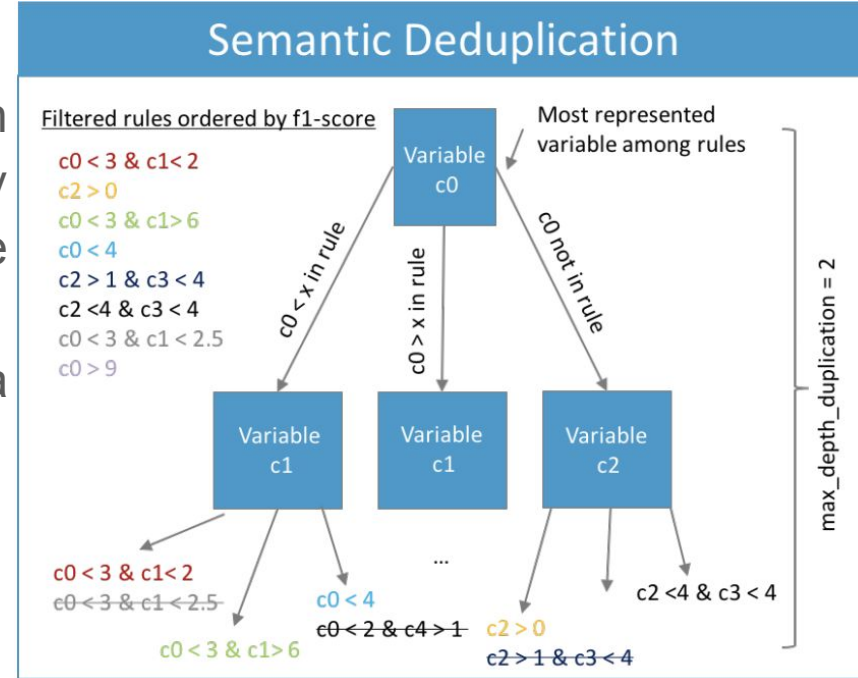
Skopec-rules

- **Bagging estimator training:** multiple decision tree classifiers are trained. Note that each node in this bagging estimator can be seen as a rule
- **Performance filtering:** out-of-bag precision and recall thresholds are applied to select best rules



Skopec-rules

- **Semantic deduplication:** a similarity filtering is applied to maintain enough diversity among the rules. The similarity measure of two rules is based on the number of their common terms. A term is a variable name combined with a comparison operator (< or >)



Computing rules directly from data. What's wrong?

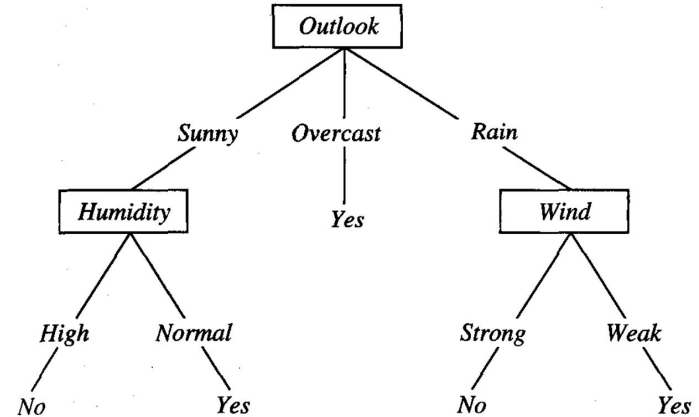
- Computationally **expensive search for splitting points** in each interval range (compare with a greedy dichotomous splitting of the decision trees)
- **Pre-discretization** is not always a good solution (in a few slides)
- **Potentially exponential number of rules**, the approaches for a greedy search for good rules are underdeveloped

The issues are even more acute in the unsupervised settings!

Decision trees vs rule-based approaches

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision tree:

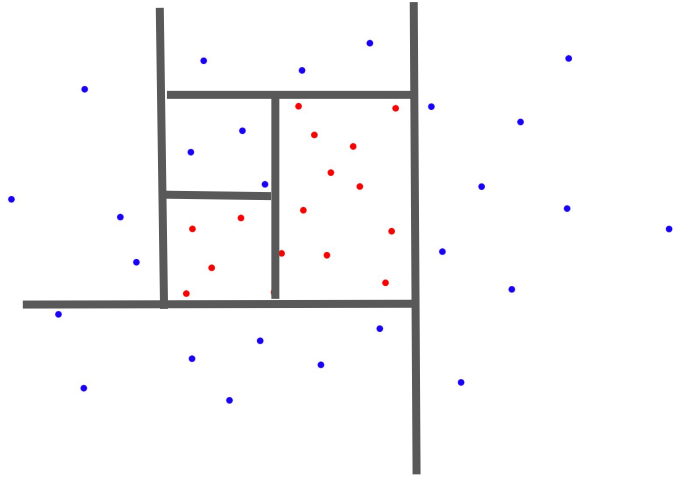


Equivalent rule list: IF "Outlook = sunny" AND "Humidity = high" THEN **NO**
ELSE IF "Outlook = rain" AND "Wind = strong" THEN **NO**
ELSE **YES**

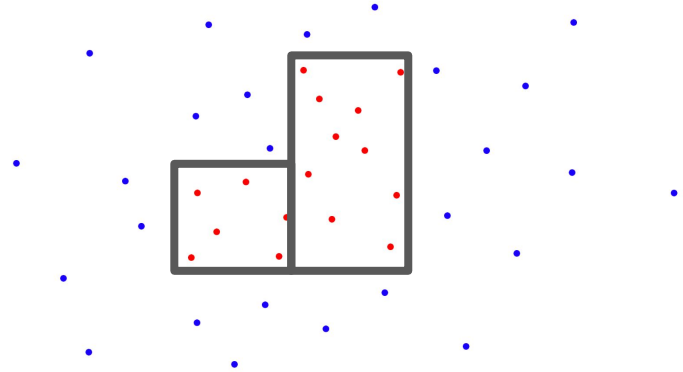
They take into account interaction between features unlike regression models

Principles of splitting

Trees: dichotomous search for boundaries



Rules: “arbitrary” search for boundaries



- + : computationally simpler than for rules
- : potentially more complicated boundaries

Methods for dealing with numerical data

UNSUPERVISED SETTINGS

- * Itemset mining

Preprocessing: discretization, binarization

- * Numerical pattern mining
- * Clustering

SUPERVISED SETTINGS

- * Rules in numerical data

Preprocessing: discretization

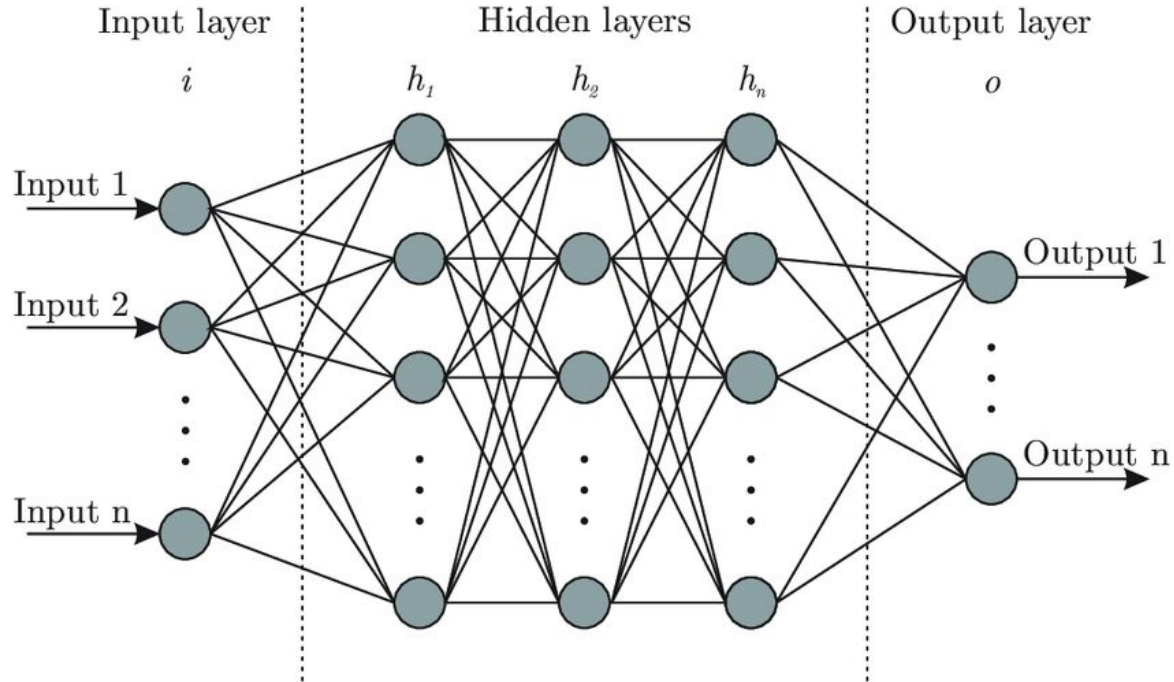
- * Tree-based methods
- * Regressions

Wrapping up

- Regularization is useful for tackling with overfitting problem and for feature selection
- Mining rules directly from (numerical) data is a computationally expensive task
- Rules may be used for concise description of (ensembles of) trees

Part 3. Neural networks. Basics

Artificial neural networks

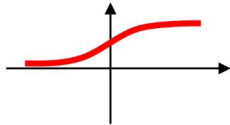
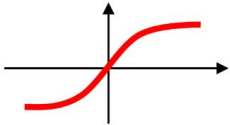
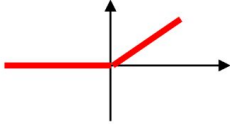


Neurons and its activation

A neuron has the following form

$$z_j = \sum \omega_{ij} x_i + b_j$$

Some of activation functions $\phi(z_j)$ for **hidden** layers:

Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN		Sigmoid and tanh suffer from saturation problem. The functions are only really sensitive to changes around their mid-point of their input, such as 0.5 for sigmoid and 0.0 for tanh. They also suffer from the vanishing gradient problem.
Hyperbolic tangent (tanh)	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks		
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = \max(0, z)$	Multi-layer Neural Networks		ReLU is free from these disadvantages

Last-layer activations and loss functions

Problem Type	Last-layer activation	Loss function
binary classification	sigmoid	binary cross entropy
multiclass, single-label classification	softmax	categorical cross entropy
multiclass, multilabel classification	sigmoid	binary cross entropy
regression to arbitrary values	-	MSE
regression to values between 0 and 1	sigmoid	MSE / binary cross entropy

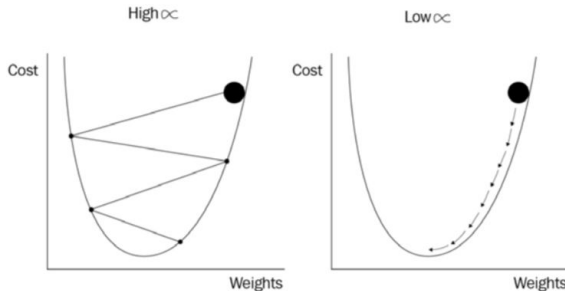
Loss vs cost functions

Generally cost and loss functions are synonymous but cost function can contain regularization terms in addition to loss function. although it is not always necessary:

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} \underbrace{\sum_j \left(t(\mathbf{x}_j) - \sum_i w_i h_i(\mathbf{x}_j) \right)^2}_{\text{loss function}} + \underbrace{\lambda \sum_{i=1}^k |w_i|}_{\text{cost function}}$$

Network training. Main steps

1. Draw a **batch** of training samples x and corresponding targets y .
2. **Forward pass**: run the network on x to obtain predictions y_{pred} .
3. Compute the **loss** of the network on the batch, a measure of the mismatch between y_{pred} and y .
4. **Backward pass**: compute the gradient of the loss with regard to the network's parameters
5. Move all weights of the network towards anti-gradient (that slightly reduces the loss on this batch) with a chosen **step**



Batch, mini-batch, and stochastic gradient descent

```
def gradientDescent(X, y, learning_rate = 0.001, batch_size = 32, n_epoch = 10):  
    theta = 0  
    error_list = []  
    for itr in range(n_epoch):  
        mini_batches = create_mini_batches(X, y, batch_size)  
        for mini_batch in mini_batches:  
            X_mini, y_mini = mini_batch  
            theta = theta - learning_rate * gradient(X_mini, y_mini, theta)  
            error_list.append(cost(X_mini, y_mini, theta))  
    return theta, error_list
```

Depending on the batch size, it can be:

- True SGD: `batch_size = 1`
- Mini-batch SGD: `1 < batch_size < #objects` (often, 32)
- Batch SGD: `batch_size = #objects`

Optimizers: setting a suitable batch size

- stochastic gradient descent: drawing a single instance
- mini-batch stochastic gradient descent: drawing a subset of instances
- batch stochastic gradient descent: using of the whole training set

Rules of thumb:

1. mini-batches of size 32 might be a good default
2. tune batch size and learning rate after all other hyperparameters
3. the size can be tuned by checking up learning curves (training and validation error vs amount of training time)
4. for BGD use relatively relatively larger learning rate and more training epochs, for SGD use a relatively smaller learning rate and fewer training epochs

Blueprinting a training process

1. Define a problem and get a dataset
2. Select suitable evaluation metrics (to monitor on validation data)
3. Decide an evaluation protocol
4. Prepare your data
 - a. use values between 0 and 1
 - b. have roughly the same ranges for all features
 - c. perform an additional normalization (with mean 0 and std 1)
 - d. do some feature engineering if you don't have enough features
 - e. use a special number, e.g., 0, for missing values so that a NN learn to ignore them
 - f. Develop a model that is better than a basic baseline
5. Develop a model that overfits
6. Regularizing your model and tuning your hyperparameters

Tackling with overfitting

To define the capacity of your model try to create one that overfits by

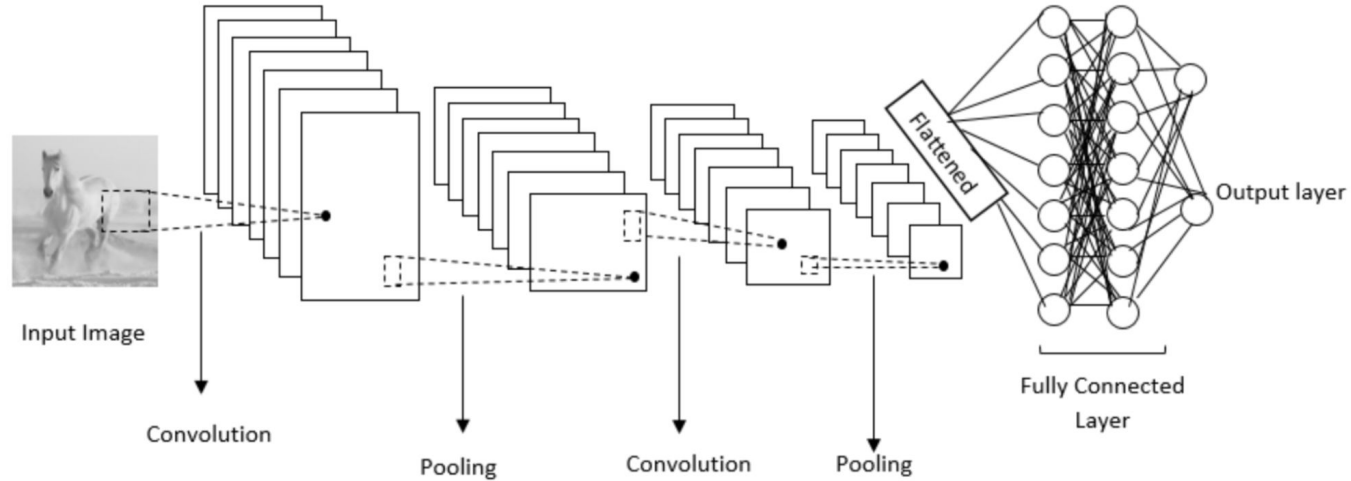
- adding layers
- making the layers bigger
- adding more epochs

Then just remove excessive elements...

To regularize your model:

- add dropout (to break neuron coalitions)
- get more data (including “data augmentation”)
- add weight regularization (L1 and L2, check the previous lecture)
- reduce the network capacity

Architecture of the CNN



Motivation: dense layers learn global patterns; while convolutional layers learn local patterns.

Key parameters: the **size** of the patches extracted from the inputs (kernel size), **depth** of the output feature map

Other parameters: strides and padding.

To understand better the aforementioned parameters, check <https://poloclub.github.io/cnn-explainer/>

Convolutional neural networks (continuation)

- **Convolutional layers:** searching local patterns
 - Application of kernels and an activation function
- **Pooling layers:** dimension reduction
 - Maximum pooling (better)
 - Average pooling

Convolution operation

25	0	1
3	9	15
0	20	5

Input image

	0
3	

	1
9	

	9
0	

	15
20	

0	1
1	0

Filter (kernel)

3	10
9	35



Feature map

35

Max pooling



Convolution operation

Edge detection


$$* \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} =$$


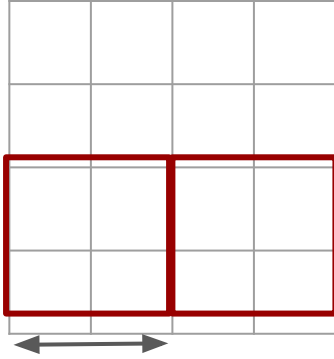
Kernel

Sharpen

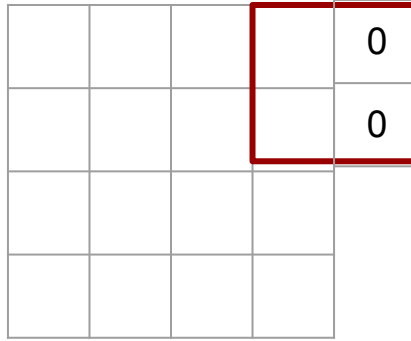

$$* \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} =$$


Application of filters

Strides

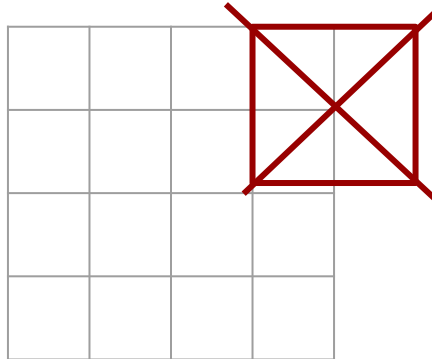


stride = 2



same (zero) padding

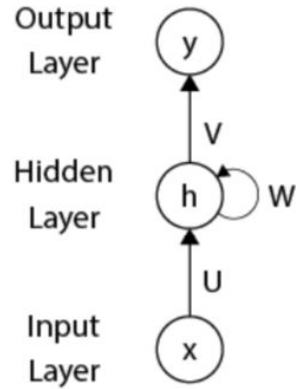
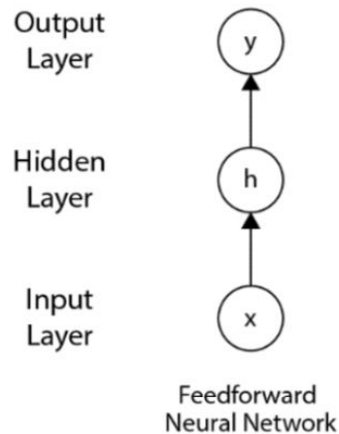
the padding ensures that the output has the same shape as the input data



valid (no) padding

Recurrent Neural Networks

For taking into account the context (dealing with time series)



$$h_t = \tanh(Ux_t + Wh_{t-1})$$

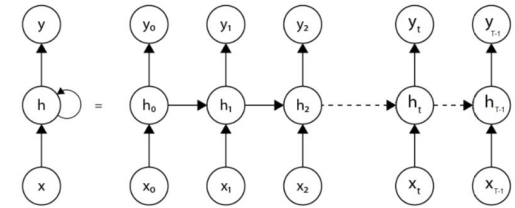


Figure 7.17: Unrolled RNN

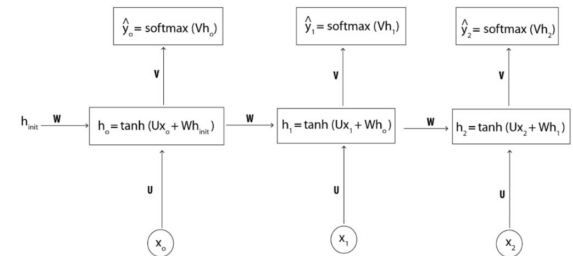
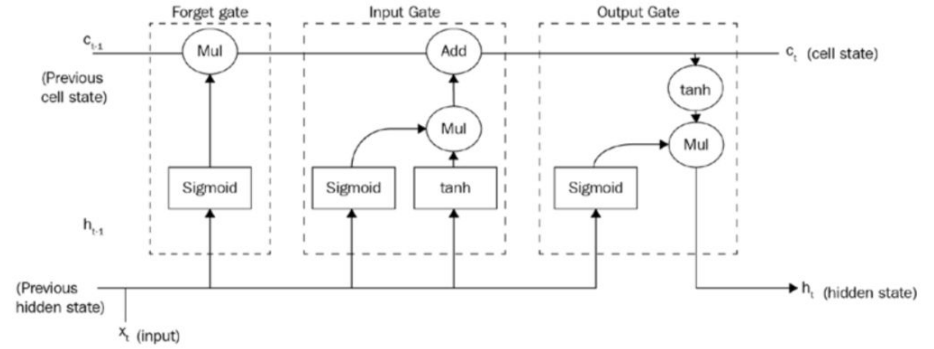
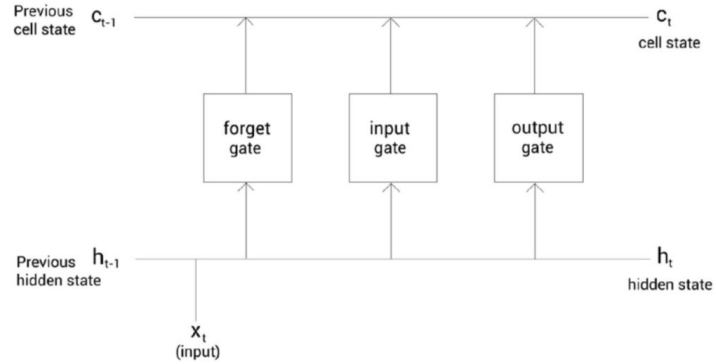


Figure 7.20: Unrolled version – forward propagation in an RNN

Long Short-Term Memory



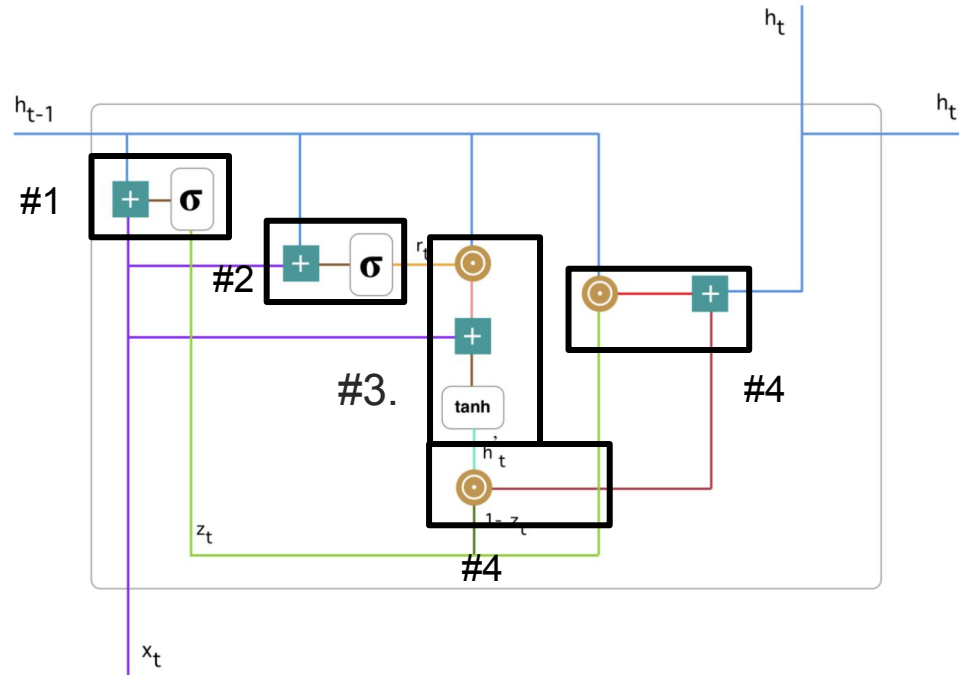
Gated Recurrent Units

#1. Update gate

#2. Reset gate

#3. Current memory content

#4. Final memory at current time



Source: <https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be>

Further reading on deep learning

