

# Prosjekt

## Titanic katastrofen- ECON 3170

### *Introduksjon*

I denne oppgaven skal vi - ved hjelp av ulike maksinlæringsmodeller - forsøke å predikere om passasjerer på Titanic overlevde eller ikke. Datasettet er hentet fra kaggle, og oppgaven er inspirert av Kaggles maksinlæringskonkurranse (Kaggle). Vi innleder med å innspisere datasettet, og forsøke å få oversikt over de ulike variablene, og omfanget av manglende verdier. Vi vil så forsøke å behandle de manglende verdiene, og tilpasse de slik at de blir enklere å bruke i treningen av maskinlæringsmodellene. Videre vil vi innspisere datasettet ved hjelp av plots for å forsøke å se sammenhengen mellom ulike variabler, og andelen overlevende. Vi vil så tilpasse parametere og hyperparametere for å forsøke å få så presise modeller som mulig. Til slutt vil vi implementere modellene i et dashboard hvor en kan finne overlevelsessansynlighet for en selvvalgt karakter.

### **Pakker**

I prosjektet kommer vi hovedsaklig til å benytte oss av to pakker: tidyverse og tidymodels. Begge pakkene er samlinger av mange ulike pakker. Vi kommer i hovedsak til å bruke Tidyverse til manipulering og visualisering av data, og Tidymodels til maskinlærning.

```
library(tidyverse)
library(tidymodels)
```

### **Data**

```
titanic <- suppressMessages(read_csv("Titanic-Dataset.csv"))
glimpse(titanic[1,])
```

```

Rows: 1
Columns: 12
$ PassengerId <dbl> 1
$ Survived    <dbl> 0
$ Pclass      <dbl> 3
$ Name        <chr> "Braund, Mr. Owen Harris"
$ Sex         <chr> "male"
$ Age         <dbl> 22
$ SibSp       <dbl> 1
$ Parch       <dbl> 0
$ Ticket      <chr> "A/5 21171"
$ Fare        <dbl> 7.25
$ Cabin       <chr> NA
$ Embarked    <chr> "S"

```

#NOTE: "suppressMessages", for å unngå output melding.

Datasettet består av 891 observasjoner med 12 variabler, og er en oversikt over et utvalg av passasjerene ombord på Titanic. Første kolonne er PassengerId som gir hver observasjon en unik ID. Den andre kolonnen er Survival; en binær variabel som indikerer om passasjerens overlevde (1) eller ikke (0). Tredje kolonne er Pclass som forteller hvilken klasse passasjerens var på. Fjerde kolonne, Name, viser navnene til hver enkelt passasjer. Kolonne fem, Sex, er en kategorisk variabel som betegner kjønn til passasjerens. Den sjette kolonnen, Age, er en kontinuerlig variabel som gir alderen til passasjerens. Kolonne sju, SibSp, er en variabel som viser antallet søsken eller ektefeller ombord på Titanic. Kolonne åtte, Parch viser antallet foreldre eller barn ombord. Niende kolonne, Ticket, viser bilettnummeret til passasjerens. Tiende kolonne, Fare, betegner prisen passasjerens brukte på billetten. Kolonne elleve, Cabin, er lugarnummeret til passasjerens. Embarked, den tolvte kolonnen er en kategorisk variabel som viser hvor passasjerens gikk på.

Kilde: (Kaggle)

```
head(titanic)
```

```

# A tibble: 6 x 12
  PassengerId Survived Pclass Name      Sex      Age SibSp Parch Ticket  Fare Cabin
    <dbl>      <dbl> <dbl> <chr>   <chr> <dbl> <dbl> <dbl> <chr>  <dbl> <chr>
1         1         0     3 Braund~ male    22     1     0 A/5 2~  7.25 <NA>
2         2         1     1 Cuming~ fema~   38     1     0 PC 17~ 71.3  C85
3         3         1     3 Heikki~ fema~   26     0     0 STON/~  7.92 <NA>
4         4         1     1 Futrel~ fema~   35     1     0 113803 53.1  C123
5         5         0     3 Allen,~ male    35     0     0 373450  8.05 <NA>
6         6         0     3 Moran,~ male    NA     0     0 330877  8.46 <NA>

```

```
# i 1 more variable: Embarked <chr>
```

Pclass og Fare virker veldig spennede, de fleste har nok sett filmen og der blir tydlig fremstilt at hvis du er i første klasse har du en større sjanse for å overleve. Samme med Fare, hvis du har betalt mye for billetten indikerer dette en høyere klasse som kan hinte til en større sannsynlighet for å overleve.

Cabin og Ticket er nok også verdier som har en sterk tilknytting med hvilken klasse du er i og hvor mye du betalte. Likevel blir det vanskelig å skulle si om noen overlever p.g.a hvilket billettnummer man har. Ved første øyekast ser det ut som at lugar variabelen har mange manglende verdier. Hva den faktiske lugaren var er nok kanskje ikke altfor interessant, men hvilket dekk den ligger på kunne vært av stor interesse. Ettersom at dekket har mye å si for om du klarte å komme deg ut i tide.

Alder og kjønn er igjen ganske interessante variabler. Vi kjenner jo alle til “Kvinner og barn først”. Her vil det nok være interessant å manipulere dataen for å fremmehve om vedkommende er et barn eller ikke. En annen dimensjon kan også være om personen blir ansett som gammel eller ikke, ettersom at disse gruppen antagligvis får prioritet under evakuering av skipet. Samt også kombinere disse verdiene med hvilken klasse de var i.

Navn er litt “tricky”. Man kan gjerne tenke seg at navet til noen ikke er av stor betydning når det kommer til overlevelse som for såvidt kan stemme. Likevel inkluderer navene tittler so f.eks mrs og master, og dette kan igjen være interessant.

Embarked er igjen variabel det blir litt vanskelig å si noe om, ettersom at det er kun tre forskjellige steder, som egentlig ikke burde ha særlig stor betydning. På den andre siden kan det være en korrelasjon mellom hvor man gikk om bord og hvilken klassen man er i.

Parch og Sibsp er en ganske interessante variabler. Kan det være noe sammenheng mellom hvor stor familie du har ombord og overlevelse.

PassengerId virker ikke veldig relevant, den virker litt mer som en variabel for å kunne referere til passasjer og den har nok mest sannsynlig ikke noe å gjøre med overlevelsen.

(Trenger vi denne gjennomgangen?)

*Inspirasjon (Allohvåk)*

## Utforsking av dataen

### Behandling av manglende verdier(NA's)

Bruker skim fra pakken skimr for å få en oversikt over dataen:

```
skimr::skim(titanic)
```

Table 1: Data summary

Name	titanic
Number of rows	891
Number of columns	12
Column type frequency:	
character	5
numeric	7
Group variables	
None	

#### Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
Name	0	1.00	12	82	0	891	0
Sex	0	1.00	4	6	0	2	0
Ticket	0	1.00	3	18	0	681	0
Cabin	687	0.23	1	15	0	147	0
Embarked	2	1.00	1	1	0	3	0

#### Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
PassengerId	0	1.0	446.00	257.35	1.00	223.50	446.00	668.5	891.00	
Survived	0	1.0	0.38	0.49	0.00	0.00	0.00	1.0	1.00	
Pclass	0	1.0	2.31	0.84	1.00	2.00	3.00	3.0	3.00	
Age	177	0.8	29.70	14.53	0.42	20.12	28.00	38.0	80.00	
SibSp	0	1.0	0.52	1.10	0.00	0.00	0.00	1.0	8.00	
Parch	0	1.0	0.38	0.81	0.00	0.00	0.00	0.0	6.00	
Fare	0	1.0	32.20	49.69	0.00	7.91	14.45	31.0	512.33	

Her får vi vite at det 891 rander og at 5 av de tolv kolonnene er av typen “character”. Det som er av mest interesse i outputen er kolonnen med “n\_missing”. Kolonnen “Cabin” har 687 manglende verdier. Det er rimelig å tenke seg at lugarnummeret ikke har så mye å si for overlevelsen, og at heller “Pclass” er viktigere for analysen. Med så mange manglende verdier

blir det også vanskelig å lage nye variabler med hvilket dekk man var på. Da måtte man eventuelt gjort noen antagelser om at første klasse var på dekk A og B, men det blir jo nesten det samme som pclass og derfor ikke hensiktsmessig. Ettersom at datasettet ikke er så stort er det mer hensiktsmessig å fjerne hele kolonnen fremfor å fjerne alle randene med manglende verdier.

Vetle:

Vi bruker funksjonen skim fra skimr pakken for å skaffe en oversikt over dataen. Det mest interessante i utskriften er kolonnen “n\_missing” som viser hvor mange manglende verdier det er per variabel. Dersom vi ser på variabelen “Cabin” ser vi at den har 687 manglende verdier. Det er rimelig å anta at selve lugarnummeret ikke er så viktig for å predikere overlevelse, og at variabler som Pclass da vil være til større hjelp. Dersom man hadde gjort noen antagelser om hvordan dekkene var organisert kunne en brukt for eksempel Pclass og Fare til å forsøke å estimere lugarnummeret, men ettersom dette vil bli noe upresist og kunne resultert i uheldige utslag på prediksjonene - som følge av et lite datasett - velger vi istedenfor å bare fjerne hele variabelen.

```
titanic <- titanic |>
  select(-Cabin)
```

Alder er også en variabel som mangler 177. Denne kolonnen er litt mer problematisk. Som drøftet ovenfor kan man tenke seg at alder har mye å si for analysen. Datasettet er lite så det blir dumt å fjerne disse radene. En muligheter er å kunne fylle disse verdiene med gjennomsnittsalderen. En annen metode som blir diskutert i denne artikkelen til *Allohv* er å se på tittler og klasser for og imputere gjennomsnittsalderne(*Allohv*).

Variabelen Age mangler 177 verdier, noe som kan være problematisk ettersom vi tenker at alder kan være viktig en viktig indikator for om noen overlevde eller ikke. Siden datasettet ikke er så stort velger vi å forsøke å fylle de manglende verdiene med gjennomsnittsalderen til passasjerer som deler tittel, og reiser med samme klasse (Pclass). Ved å dele inn etter både tittel og klasse forsøker vi å unngå at for mange observasjoner får den samme alderen, noe som kan bidra til å påvirke presisjonen til modellene.

```
#Lager lister til å fylle inn verdiene
master.mean <- list()
mr.mean <- list()
mrs.mean <- list()
miss.mean <- list()

#Setter opp en for-loop til å finne gjennomsnittet for alle med en gitt tittel, i en gitt kl
for (i in 1:3){
  for (j in c("Master", "Mr.", "Mrs.", "Miss")) {
```

```

    if (j == "Master"){
      master.mean[[i]] <- titanic |>
        filter(grepl("Master.", Name, fixed = TRUE), Pclass == i, Sex == "male") |>
        summarise(mean_age = mean(Age, na.rm = TRUE)) |>
        pull(mean_age)
    } else if(j == "Mr."){
      mr.mean[[i]] <- titanic |>
        filter(grepl("Mr.", Name, fixed = TRUE), Pclass == i, Sex == "male") |>
        summarise(mean_age = mean(Age, na.rm = TRUE)) |>
        pull(mean_age)
    } else if (j == "Mrs.") {
      mrs.mean[[i]] <- titanic |>
        filter(grepl("Mrs.", Name, fixed = TRUE), Pclass == i, Sex == "female") |>
        summarise(mean_age = mean(Age, na.rm = TRUE)) |>
        pull(mean_age)
    } else if(j == "Miss") {
      miss.mean[[i]] <- titanic |>
        filter(grepl("Miss.", Name, fixed = TRUE), Pclass == i, Sex == "female") |>
        summarise(mean_age = mean(Age, na.rm = TRUE)) |>
        pull(mean_age)
    }
  }
}

#Fyller inn de manglende verdiene
for (i in 1:3){
  titanic <- titanic |>
    mutate(Age = ifelse(grepl("Mr.", Name, fixed = T) & Pclass == i & is.na(Age), mr.mean[[i]],
    mutate(Age = ifelse(grepl("Miss", Name, fixed = T) & Pclass == i & is.na(Age), miss.mean[[i]],
    mutate(Age = ifelse(grepl("Mrs.", Name, fixed = T) & Pclass == i & is.na(Age), mrs.mean[[i]],
    mutate(Age = ifelse(grepl("Master", Name, fixed = T) & Pclass == i & is.na(Age), master.mean[[i]],
}

```

Vi fyller inn de manglende verdiene ved å ta utgangspunkt i tittelene - Master, Mr. , Mrs. , Miss. - og finner gjennomsnittsalderen for klassen de tilhører, og fyller dette inn i datasettet. Ettersom det eksisterer flere titler som vi ikke dekket sjekker vi om vi finner flere manglende verdier, og forsøker samme metode for å sette inn verdiene.

```

#Sjekker om det er fler manglende verider
titanic |>
  filter(is.na(Age))

```

```
# A tibble: 1 x 11
  PassengerId Survived Pclass Name          Sex    Age SibSp Parch Ticket  Fare
      <dbl>     <dbl>  <dbl> <chr>          <chr> <dbl> <dbl> <dbl> <chr>  <dbl>
1         767         0      1 Brewe, Dr. A~ male    NA     0     0 112379  39.6
# i 1 more variable: Embarked <chr>
```

```
#Finner gjennomsnittet for tittel gitt klasse
mean.dr.p1 <- titanic |>
  filter(grepl("Dr.", Name, fixed = TRUE) & Pclass == 1) |>
  summarise(round(mean(Age, na.rm = TRUE))) |>
  pull()

#Setter inn for verdien
titanic <- titanic |>
  mutate(Age = ifelse(is.na(Age), mean.dr.p1, Age))
```

Fra utskriftet til skimr funksjonen ser vi at det nå kun mangler 2 verdier for variabelen Embarked. Vi undersøker disse nærmere:

```
titanic |>
  filter(is.na(Embarked))
```

```
# A tibble: 2 x 11
  PassengerId Survived Pclass Name          Sex    Age SibSp Parch Ticket  Fare
      <dbl>     <dbl>  <dbl> <chr>          <chr> <dbl> <dbl> <dbl> <chr>  <dbl>
1         62         1      1 Icard, Miss.~ fema~    38     0     0 113572   80
2        830         1      1 Stone, Mrs. ~ fema~    62     0     0 113572   80
# i 1 more variable: Embarked <chr>
```

Vi ser at de manglende verdiene er for to kvinner, som begge overlevde, og som reiste med første klasse. For å opprettholde variasjonen i datasettet, og bevare flest mulig av observasjonene vi ønsker å predikere, vil vi forsøke å beholde observasjonene ved å fylle inn en verdi for Embarked.

```
titanic |>
  filter(Sex == "female" & Pclass == 1) |>
  group_by(Embarked, Pclass) |>
  summarise(mean_survived = mean(Survived), antall = n())
```

```
# A tibble: 4 x 4
```

```
# Groups:   Embarked [4]
  Embarked Pclass mean_survived antall
  <chr>     <dbl>         <dbl> <int>
1 C         1         0.977     43
2 Q         1         1         1
3 S         1         0.958     48
4 <NA>      1         1         2
```

Det vi kan tolke fra output er at de fleste kvinner som er i førseklasse overlevde, uavhengig av hvor de kom ombord. Derfor tenker vi det er rimelig å fylle de manglende verdiene med tilfeldig trekk mellom "C" og "S", ettersom at de er de verdien som er mest sannsynlige.

Det vi kan tolke fra utskriften er at de fleste kvinnene som reiste med førsteklasse overlevde, uavhengig av hvor de gikk ombord. Det vil derfor være rimelig å fylle de manglende verdiene med et tilfeldig trekk mellom "C" og "S", ettersom disse var de vanligste stedene å gå ombord.

```
tilfeldig_embarked <- sample(c("C", "S"), size = 1)

titanic <- titanic |>
  mutate(Embarked = ifelse(is.na(Embarked),
                           yes = tilfeldig_embarked,
                           no = Embarked))

paste("Det er", sum(is.na(titanic)), "NAs datasettet")
```

```
[1] "Det er 0 NAs datasettet"
```

## Visualisering av data

Nå som vi har bearbeidet datarammen har vi lyst til å utforske dataen litt mer. Dette gjør vi for å få en bedre forståelse for ulike sammenhenger i dataen.

Når vi nå har bearbeidet dataene ønsker vi å utforske de ytterligere, for å få et bedre inntrykk av sammenhengene i dataene.

```
titanic |>
  group_by(Survived) |>
  summarise(Antall = n())
```

```
# A tibble: 2 x 2
  Survived Antall
```



	<dbl>	<int>
1	0	549
2	1	342

Vi ser her at det kun er 342 passasjerer som overlevde, og at det var 549 passasjerer som omkom. Vi ønsker først å se på hvordan antall overlevende fordeler seg på klassen de reiste med.

```
titanic |>
  ggplot(aes(x = Survived, fill = as.factor(Survived))) +
  geom_bar() +
  ylab("Antall") +
  labs(fill = "Survived")+
  ggtitle("Overlevelse i klasser")
```



```
facet_wrap(vars(Pclass))
```

```
<ggproto object: Class FacetWrap, Facet, gg>
  compute_layout: function
  draw_back: function
  draw_front: function
  draw_labels: function
```

```

draw_panels: function
finish_data: function
init_scales: function
map_data: function
params: list
setup_data: function
setup_params: function
shrink: TRUE
train_scales: function
vars: function
super: <ggproto object: Class FacetWrap, Facet, gg>

```

Antallet overlevende ser ut til å være ganske likt mellom klassene. Å se på andelen overlevende fordelt på klasse kan derfor gi et bedre inntrykk.

```

titanic |>
  group_by(Pclass) |>
  summarise(Overlevelsesandel = sum(Survived)/n())

```

```

# A tibble: 3 x 2
  Pclass Overlevelsesandel
  <dbl>         <dbl>
1     1           0.630
2     2           0.473
3     3           0.242

```

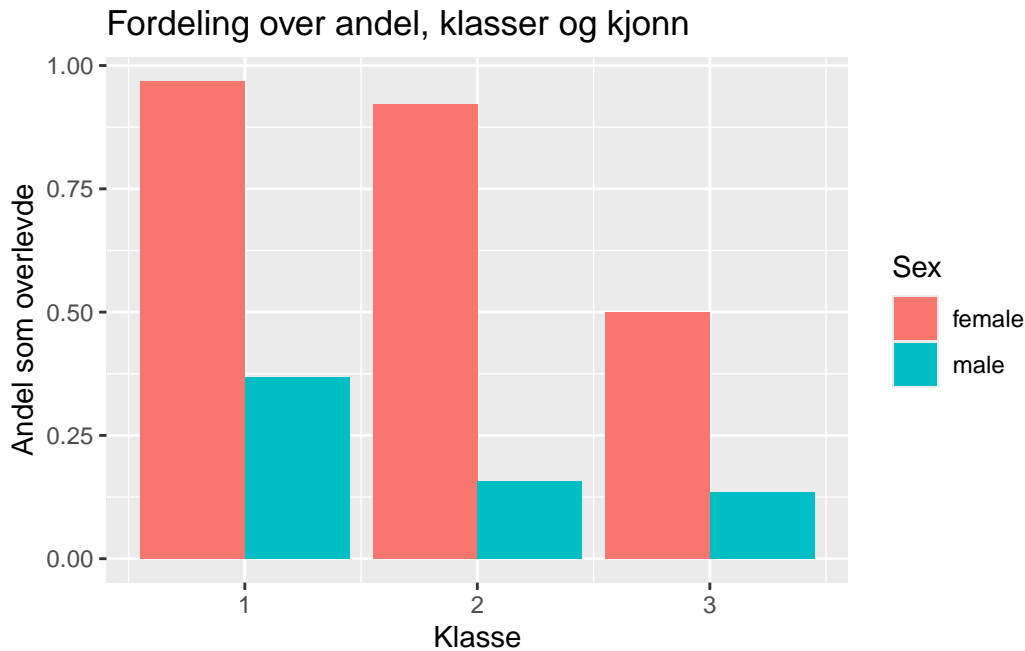
Her får vi et tydeligere bilde over overlevelse og hvilken klasse du er i. Vi ser at andel som overlevde blir betraktlig større desto høyere klasse klasse man er i. Kan kjønn ha noe betydning?

```

library(tidyverse)

titanic |>
  group_by(Pclass, Sex) |>
  summarise(Overlevelsesandel = sum(Survived)/n()) |>
  ggplot(aes(x = Pclass, y = Overlevelsesandel, fill = Sex)) +
  geom_col(position = position_dodge()) +
  ylab("Andel som overlevde") +
  xlab("Klasse") +
  ggtitle("Fordeling over andel, klasser og kjønn")

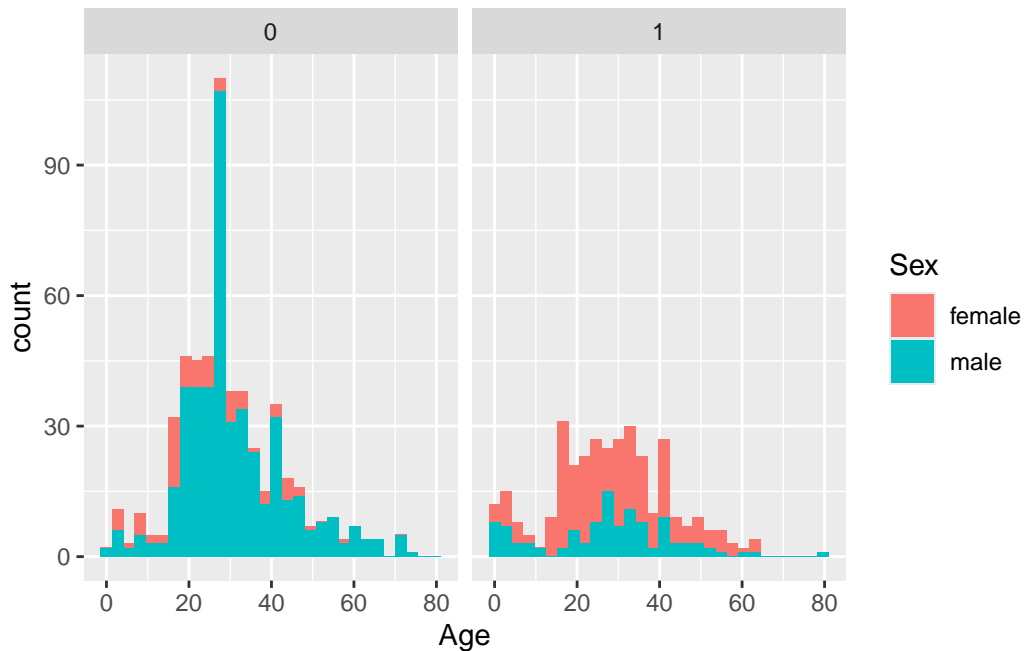
```



Her ser vi at det er en tydelig sammenheng mellom klasse og kjønn for overlevelse. Det ser ut som at kvinner i første og andre klasse har realt stor sannsynlighet for å overleve. Kvinner i tredje klasse ligger på rundt femti prosent. Andelen menn som overlevde er betraktlig lavere i alle klasser sammenlignet med kvinner. Andel menn som overlevde i andre og tredje klasse er omtrent halvparten av det den er i første klasse, likevel er det å være mann i første klasse “dårligere” enn å være kvinne i tredjeklasse med tanke på overlevelse. Det viser seg at kjønn har større påvirkning på overlevelse fremfor hvilken klasse du er i, dette stemmer jo overens med at kvinner og barn går først når det evakueres. Det er kanskje da naturlig å se på hvordan alder spiller en rolle.

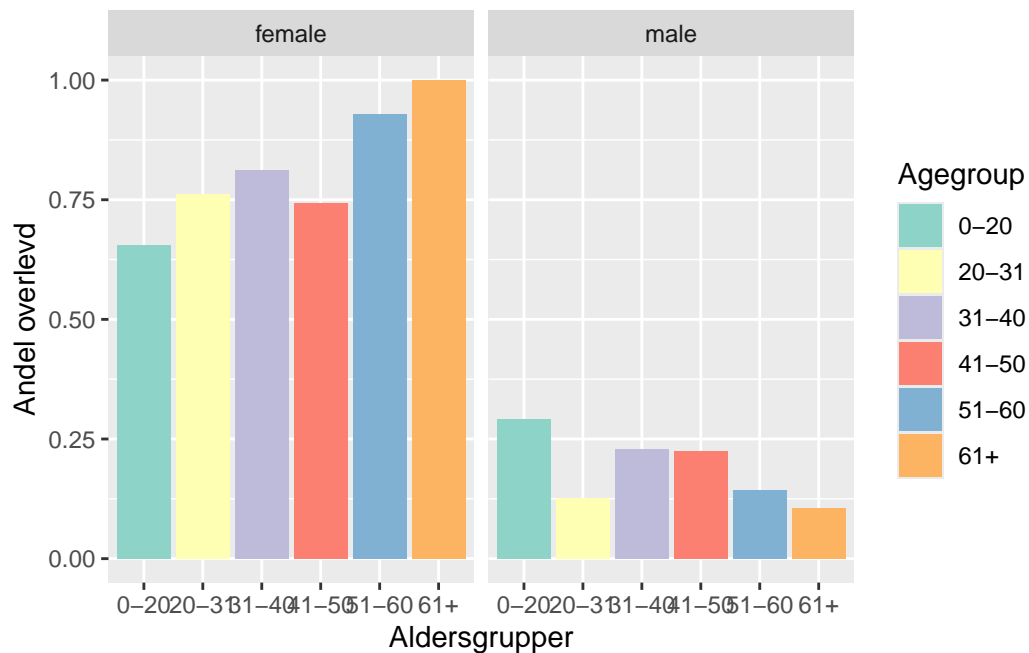
```
titanic |>
  ggplot(aes(x = Age, fill = Sex)) +
  geom_histogram() +
  facet_wrap(vars(Survived))
```

``stat_bin()` using `bins = 30`. Pick better value with `binwidth`.`



Det fremkommer ingen tydelig sammenheng, vi deler inn i aldersgrupper, og ser på andelen overlevne i ulike aldersgrupper fordelt på kjønn

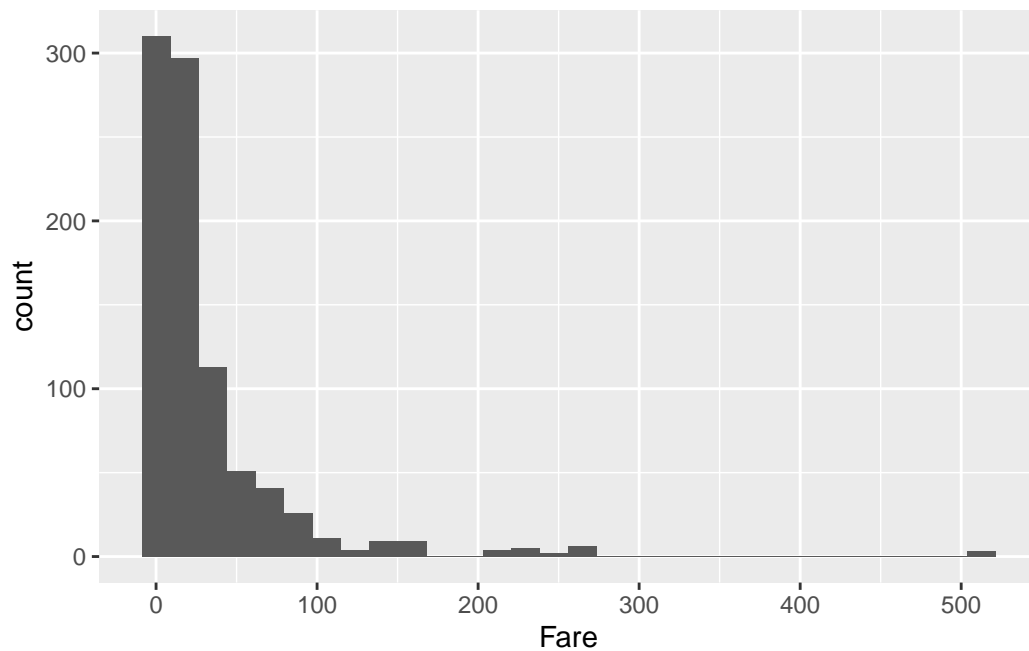
```
titanic |>
  mutate(
    Agegroup = cut(
      Age,
      breaks = c(-Inf, 20, 30, 40, 50, 60, Inf),
      labels = c("0-20", "20-31", "31-40", "41-50", "51-60", "61+")
    )
  ) |>
  group_by(Agegroup, Sex) |>
  summarise(Andel = sum(Survived)/n()) |>
  ggplot(aes(x = Agegroup, y = Andel, fill = Agegroup)) +
  geom_col() +
  scale_fill_brewer(palette = "Set3") +
  xlab("Aldersgrupper") +
  ylab("Andel overlevd") +
  facet_wrap(vars(Sex))
```



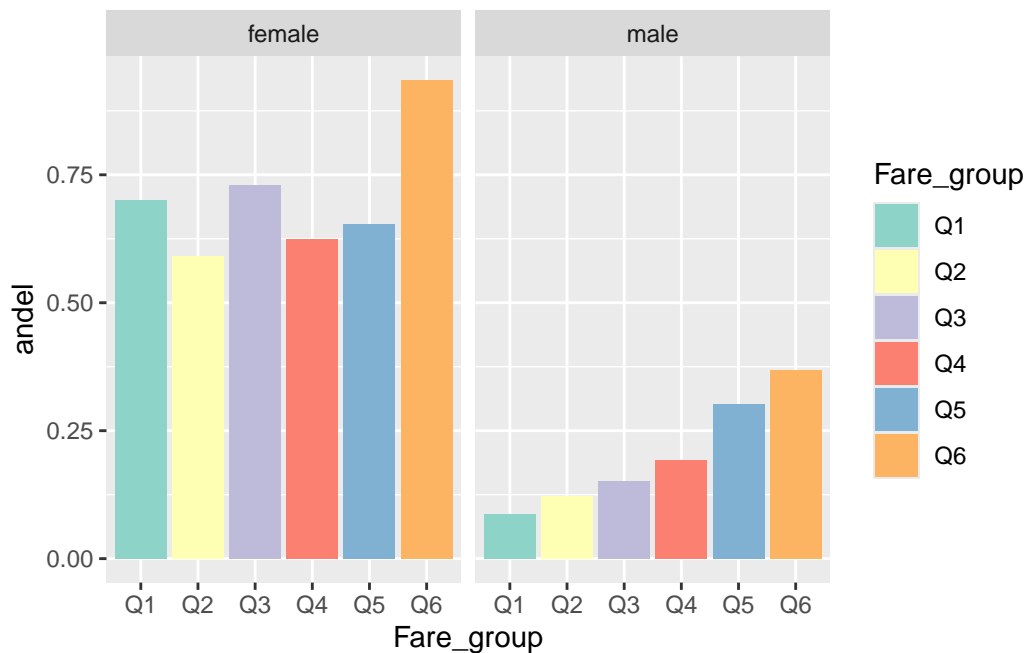
Vi ser at det er en svak sammenheng mellom alder og overlevelse. Den laveste overlevelsesraten er for menn i alderen 20-31, noe som gir mening ettersom disse trolig måtte vente lengst på redningsbåt. For kvinner er tendensen noe lik, men her er det heller ingen sterk sammenheng. Det er også viktig å nevne at sammenhengen vi ser kan være noe spuriøs. Her kan for eksempel Pclass være viktig.

Hva med prisen hvor mye de betalte, har det noe å si for overlevelsen.

```
titanic |>
  ggplot(aes(x = Fare)) +
  geom_histogram(bins = 30)
```

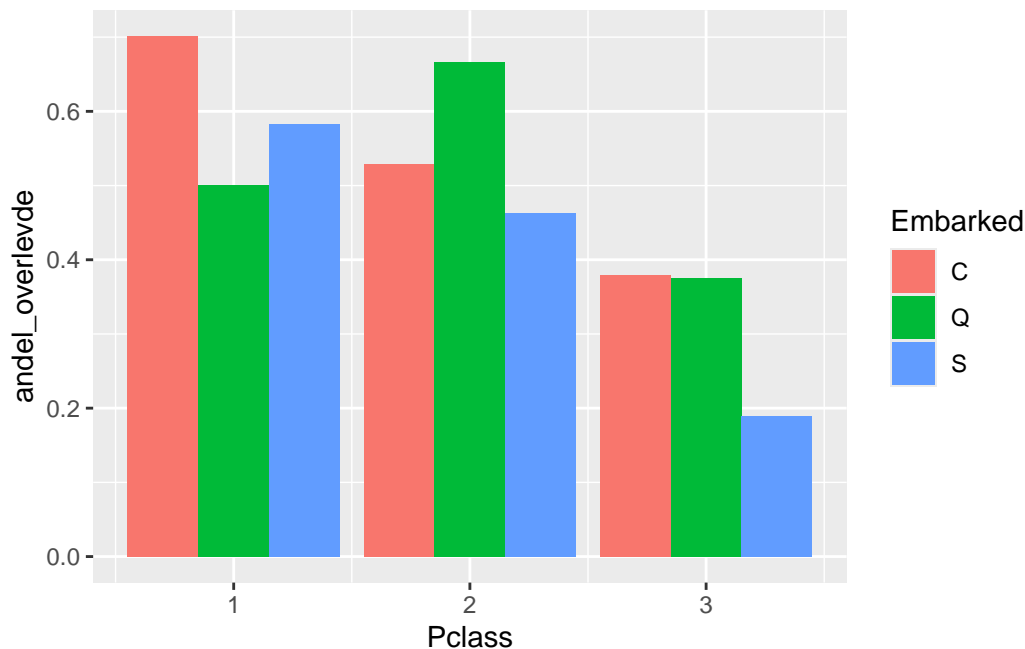


```
titanic |>
  filter(Fare < 200) |>
  mutate(Fare_group = ntile(Fare, 6)) |>
  mutate(Fare_group = factor(Fare_group, labels = c("Q1", "Q2", "Q3", "Q4", "Q5", "Q6"))) |>
  group_by(Fare_group, Sex) |>
  summarise(andel = sum(Survived)/n()) |>
  ggplot(aes(x = Fare_group, y = andel, fill = Fare_group)) +
  geom_col() +
  scale_fill_brewer(palette = "Set3") +
  facet_wrap(vars(Sex))
```



Ettersom at fordelingen over hvor mye hver person betalte er høyrevridd har vi prøvd å dele dem inn i seks grupper, slik som for alder. Vi ser også at det er en del uteliggere, derfor har vi prøvd å filtrere disse bort. Det vi tolker fra grafene er at jo mer folk betalte for reisen desto større andel overlevde, spesielt hos menn. Funnet samsvarer også med grafen over at andelen menn i første klasse hadde en større andel overlevende. Tendensen er ikke like sterk hos kvinner. Noe som tyder på at det å være kvinner taler for at billettprisen har mindre betydning for dem enn det den har hos menn.

```
titanic |>
  group_by(Embarked, Pclass) |>
  summarise(andel_overlevde = sum(Survived)/ n()) |>
  ggplot(aes(x = Pclass, y = andel_overlevde, fill = Embarked )) +
  geom_col(position = position_dodge())
```



Grafen viser ikke noe særlig sammenheng mellom hvor du gikk ombord og overlevelse. Dette gir gi også mening ettersom at, det egentlig ikke burde ha noe å si.

Vi kan se av grafene at det i hovedsak er kjønn og hvilken klasse en tilhører som har sterkest sammenheng med overlevelsen. Vi ser at hvor mye billetten kostet har noe å si for overlevelse, men denne sammenhengen er sterkest for menn, og at hvor en gikk på ikke har så mye å si. Alder har mest å si for menn.

## Nye variabler

Vi ser at den kan være hensiktsmessig å formalisere noen av variablene, og lage nye variabler. Dette kan forbedrer modellens ytelse, gjør resultatene lettere å forstå og kan gi ny innsikt.

Vi velger å lage en dummy variabel for om man er mindreåring eller ikke, en for hvor stor familie personen reiser med, og antall personer per billett. Inspirasjon hentet (Donges, 2018).

```
titanic <- titanic |>
  add_count(Ticket, name = "Person_per_ticket") |>
  mutate(Minor = ifelse(Age < 18, 1, 0),
         Family_size = SibSp+Parch,
         Alone = ifelse(Family_size == 0, 1, 0)) |>
  select(-c(Name, Ticket, SibSp, Parch, PassengerId)) |>
  mutate(Pclass = as.factor(Pclass),
         Alone = as.factor(Alone),
```



```
Survived = as.factor(Survived),  
Minor = as.factor(Minor),  
Sex = as.factor(Sex),  
Embarked = as.factor(Embarked))
```

## Maskinlæring

Før vi tilpasser modellene vi ønsker å bruke, deler vi datasettet inn i trening og test sett. Dataene deles inn i en 80/20 fordeling hvor variabelen Survived skal være jevnt fordelt.

```
set.seed(3170)  
titanic_split<- initial_split(titanic, prop = .8, strata = Survived)  
titanic_train <- training(titanic_split)  
titanic_test  <- testing(titanic_split)  
titanic_split
```

```
<Training/Testing/Total>  
<712/179/891>
```

Lager en recepie:

```
titanic_recipe <-  
  recipe(Survived ~ ., data = titanic_train) |>  
  step_dummy(Sex, Embarked, Alone, Pclass, Minor)
```

## Lasso

```
lasso_model <-  
  logistic_reg(penalty = tune(), mixture = 1) |>  
  set_engine("glmnet") |>  
  set_mode("classification")  
  
wflow_lasso <-  
  workflow() |>  
  add_recipe(titanic_recipe) |>  
  add_model(lasso_model)
```

```

folds <- vfold_cv(titanic_train, 5, strata = Survived)

penalty_grid <- grid_regular(penalty(range = c(-4, -1)), levels = 100)

doParallel::registerDoParallel() # Parallelprogramering for raskere tune

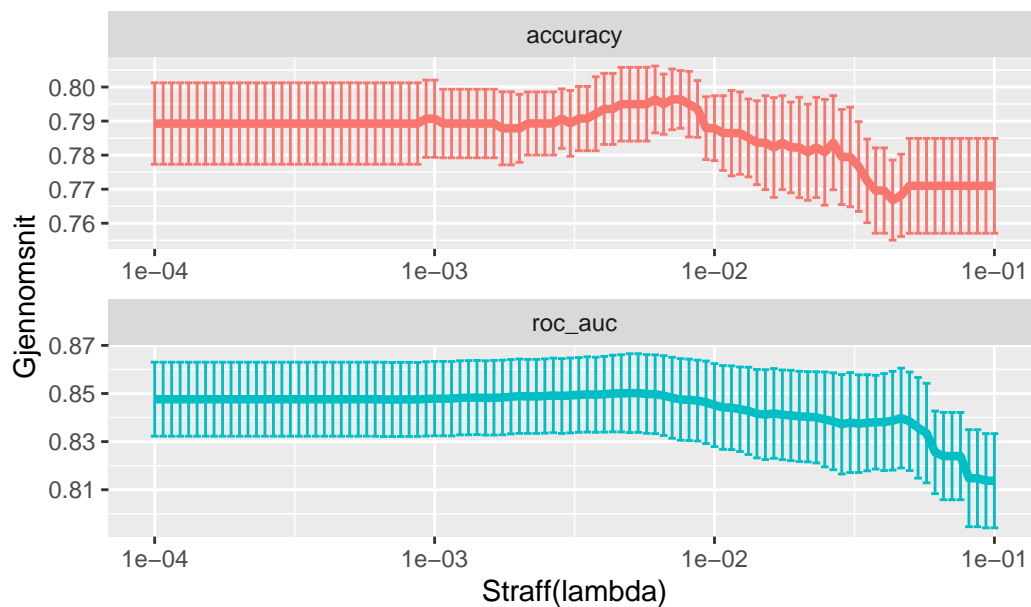
lasso_tune <-
  wflow_lasso |>
  tune_grid(resamples = folds, grid = penalty_grid)

lasso_tune |>
  collect_metrics() |>
  filter(.metric != "brier_class") |>
  ggplot(aes(penalty, mean, color = .metric)) +
  geom_errorbar(aes(
    ymin = mean - std_err,
    ymax = mean + std_err
  )) +
  geom_line(size = 1.5) +
  facet_wrap(~.metric, scales = "free", nrow = 2) +
  scale_x_log10() +
  theme(legend.position = "none") +
  ggtitle("Modellens prestasjon for ulike verdier av lambda")+
  xlab("Straff(lambda)") +
  ylab("Gjennomsnitt")

```

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
 i Please use `linewidth` instead.

## Modellens prestasjon for ulike verdier av lambda



Vi setter opp lasso modellen og tuner den. Plottet viser hvordan accuracy og roc\_auc utvikler seg for ulike verdier av lambda.

```
beste_lambda <- select_best(lasso_tune, metric = "roc_auc")
beste_lambda
```

```
# A tibble: 1 x 2
  penalty .config
  <dbl> <chr>
1 0.00534 Preprocessor1_Model058
```

```
lasso_fit <- finalize_workflow(wflow_lasso, beste_lambda) |>
  fit(data = titanic_train)
```

Bruker den beste roc\_auc verdien i modellen og tilpasser modellen med denne verdien.

```
predict(lasso_fit, titanic_test) |>
  bind_cols(titanic_test) |>
  accuracy(truth = Survived, estimate = .pred_class)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>    <chr>         <dbl>
1 accuracy binary      0.838
```

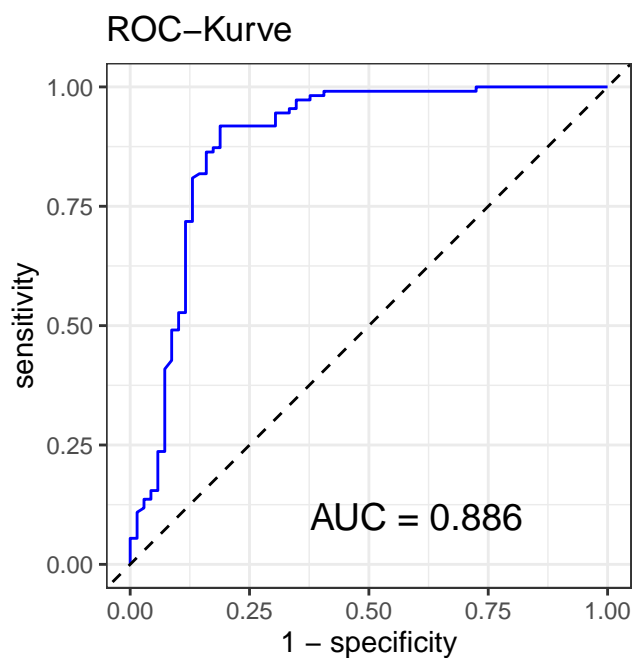
```

auc_lasso <-
  predict(lasso_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
  roc_auc(Survived, .pred_0) |>
  pull(.estimate)

lasso_roc <- predict(lasso_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
  roc_curve(Survived, .pred_0)

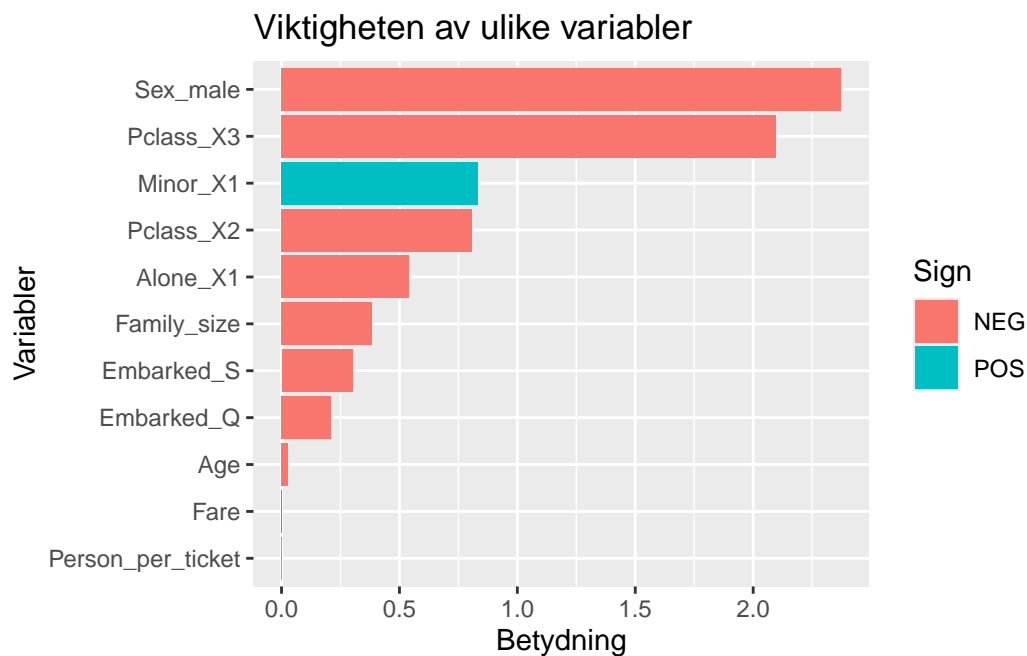
lasso_roc |>
  ggplot(aes(x = 1 - specificity, y = sensitivity)) +
  geom_path(col = "blue") +
  geom_abline(slope = 1, linetype = "dashed") +
  coord_equal()+
  annotate("text", x = 0.6, y = 0.1,
          label = paste("AUC =", round(auc_lasso,3) ), size = 5) +
  theme_bw() +
  ggtitle("ROC-Kurve")

```



Her er et plott som viser roc\_auc til lasso modellen.

```
lasso_fit |>
  extract_fit_parsnip() |>
  vip::vi() |>
  ggplot(aes(y = reorder(Variable, Importance),
               x = Importance, fill = Sign)) +
  geom_col() +
  theme() +
  ggtitle("Viktigheten av ulike variabler") +
  xlab("Betydning") +
  ylab("Variabler")
```



```
library(cvms)

prediksjoner_lasso <- predict(lasso_fit, titanic_test) |>
  bind_cols(titanic_test)

forvirringsmatrise_lasso <- prediksjoner_lasso |>
  conf_mat(Survived, .pred_class)

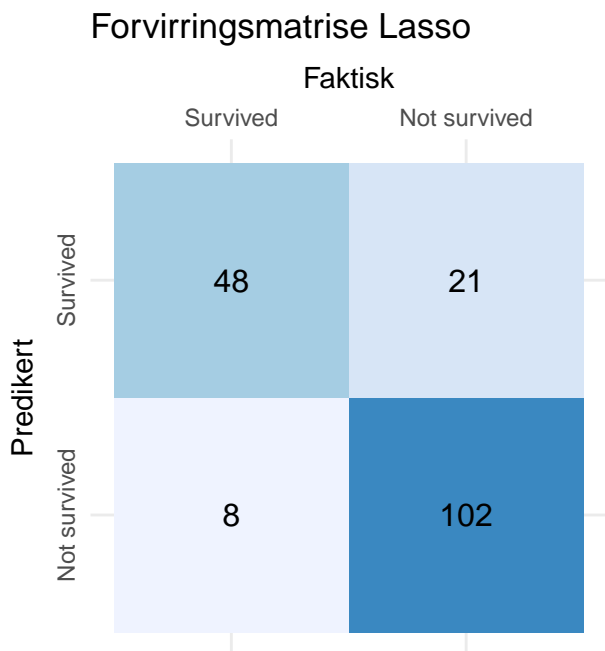
riktigmatrise_lasso <- as.tibble(forvirringsmatrise_lasso$table) |>
  mutate(Prediction = ifelse(Prediction == 0, "Not survived", "Survived")) |>
  mutate(Truth = ifelse(Truth == 0, "Not survived", "Survived"))
```

Warning: `as.tibble()` was deprecated in tibble 2.0.0.  
i Please use `as\_tibble()` instead.  
i The signature and semantics have changed, see `?as\_tibble`.

```
plot_confusion_matrix(riktigmatrise_lasso, "Prediction", "Truth", "n",  
                      add_normalized = F,  
                      add_row_percentages = F,  
                      add_col_percentages = F) +  
  xlab("Faktisk") +  
  ylab("Predikert") +  
  ggtitle("Forvirringsmatrise Lasso")
```

Warning in plot\_confusion\_matrix(riktigmatrise\_lasso, "Prediction", "Truth", :  
'ggimage' is missing. Will not plot arrows and zero-shading.

Warning in plot\_confusion\_matrix(riktigmatrise\_lasso, "Prediction", "Truth", :  
'rsvg' is missing. Will not plot arrows and zero-shading.



Konklusjon: Lasso oser godt.

## Random forest

```
set.seed(3170)

rf_mod <- rand_forest(
  mtry = tune(),
  trees = tune(),
  min_n = tune()
) |>
  set_engine("ranger", importance = "impurity") |>
  # For å kunne bruke Vip pakken
  set_mode("classification")

rf_wflow <- workflow() |>
  add_recipe(titanic_recipe) |>
  add_model(rf_mod)

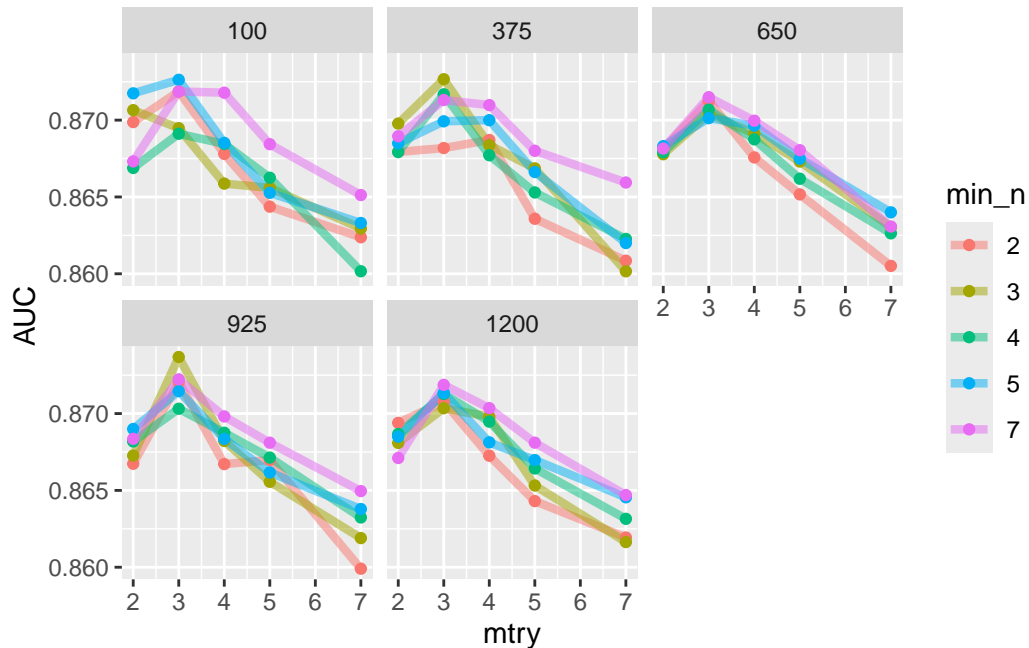
rf_grid <- grid_regular(
  mtry(range = c(2, 7)),
  min_n(range = c(2,7)),
  trees(range = c(100,1200)),
  levels = 5
)

doParallel::registerDoParallel()

tune_rf <- tune_grid(
  rf_wflow,
  resamples = folds,
  grid = rf_grid
)

tune_rf |>
  collect_metrics() |>
  filter(.metric == "roc_auc") |>
  mutate(min_n = factor(min_n)) |>
  ggplot(aes(mtry, mean, color = min_n)) +
  geom_line(alpha = 0.5, size = 1.5) +
```

```
geom_point() +
labs(y = "AUC") +
facet_wrap(vars(trees))
```



Tune rf, er en flaskehals. Vet ikke helt hv avi skal gjøre for å gjøre den raskere. Likevel leverer den stor os

```
set.seed(3170)
beste_mtry_min_tree <- select_best(tune_rf, metric = "roc_auc")

rf_fit <- finalize_workflow(rf_wflow, beste_mtry_min_tree) |>
  fit(titanic_train)

predict(rf_fit, titanic_test) |>
  bind_cols(titanic_test) |>
  accuracy(Survived, .pred_class)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>         <dbl>
1 accuracy binary      0.844
```



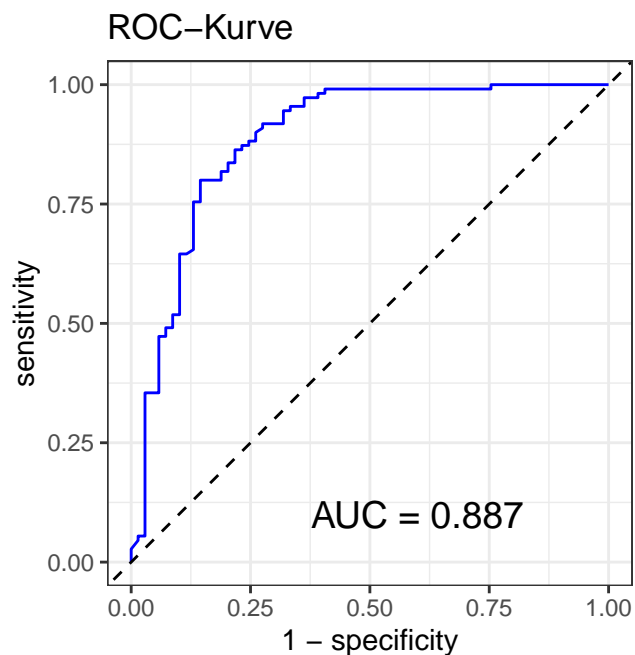
```

auc_rf <-
  predict(rf_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
  roc_auc(Survived, .pred_0) |>
  pull(.estimate)

rf_roc <- predict(rf_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
  roc_curve(Survived, .pred_0)

rf_roc|>
  ggplot(aes(x = 1 - specificity, y = sensitivity)) +
  geom_path(col = "blue") +
  geom_abline(slope = 1, linetype = "dashed") +
  coord_equal()+
  annotate("text", x = 0.6, y = 0.1, label = paste("AUC =", round(auc_rf,3) ), size = 5) +
  theme_bw() +
  ggtitle("ROC-Kurve")

```

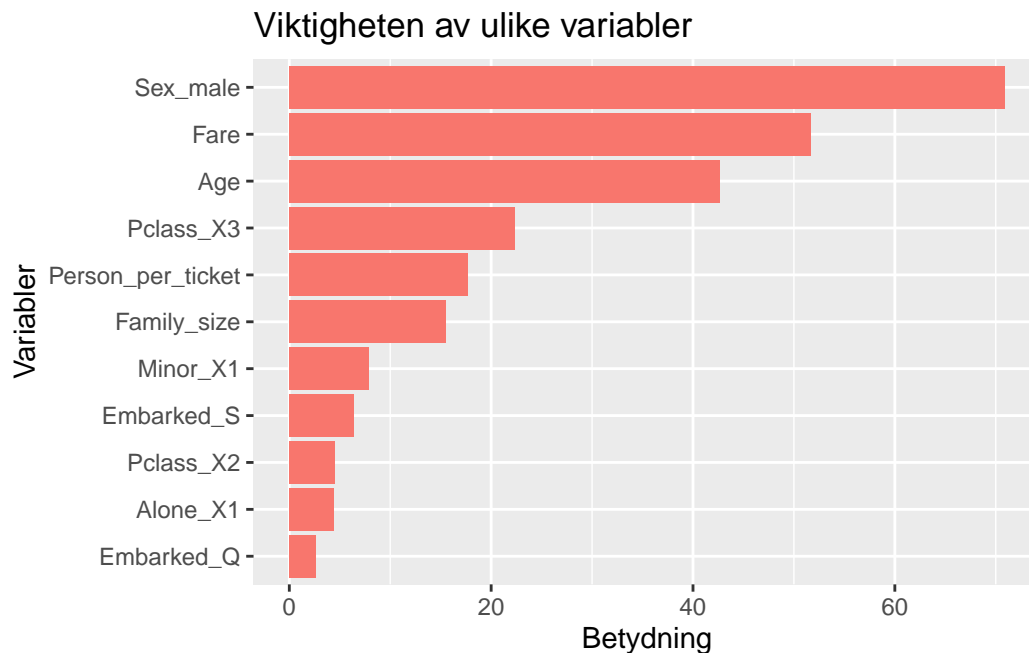


```

rf_fit |>
  extract_fit_parsnip() |>
  vip::vi() |>
  ggplot(aes(y = reorder(Variable, Importance), x = Importance, fill = "#F8766D")) +

```

```
geom_col() +
theme(legend.position = "none") +
ggtitle("Viktigheten av ulike variabler") +
xlab("Betydning") +
ylab("Variabler")
```



```
prediksjoner_rf <- predict(rf_fit, titanic_test) |>
  bind_cols(titanic_test)

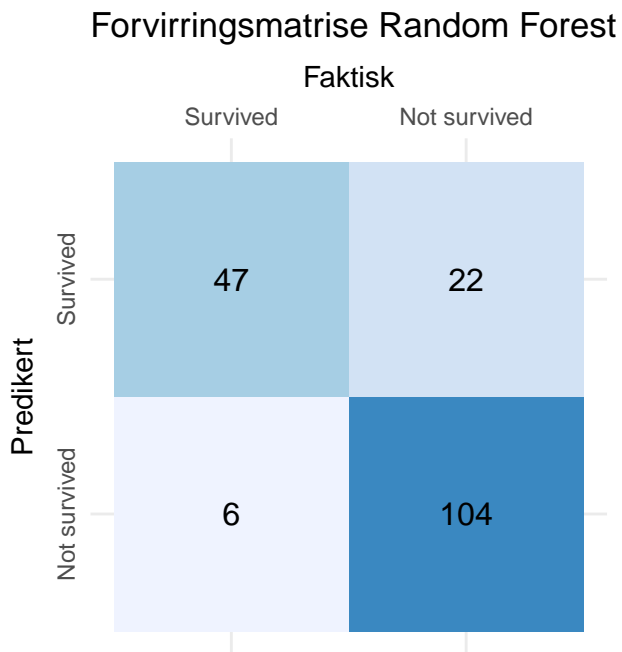
forvirringsmatrise_rf <- prediksjoner_rf |>
  conf_mat(Survived, .pred_class)

riktigmatrise_rf <- as.tibble(forvirringsmatrise_rf$table) |>
  mutate(Prediction = ifelse(Prediction == 0, "Not survived", "Survived")) |>
  mutate(Truth = ifelse(Truth == 0, "Not survived", "Survived"))

plot_confusion_matrix(riktigmatrise_rf, "Prediction", "Truth", "n",
  add_normalized = F,
  add_row_percentages = F,
  add_col_percentages = F) +
  xlab("Faktisk") +
  ylab("Predikert") +
  ggtitle("Forvirringsmatrise Random Forest")
```

Warning in plot\_confusion\_matrix(riktigmatrise\_rf, "Prediction", "Truth", :  
'ggimage' is missing. Will not plot arrows and zero-shading.

Warning in plot\_confusion\_matrix(riktigmatrise\_rf, "Prediction", "Truth", :  
'rsvg' is missing. Will not plot arrows and zero-shading.



## Decision tree

```
decision_model <- decision_tree(  
  cost_complexity = tune(),  
  tree_depth = tune(),  
  min_n = tune()) |>  
  set_engine("rpart") |>  
  set_mode("classification")  
  
decision_wflow <- workflow() |>  
  add_recipe(titanic_recipe) |>  
  add_model(decision_model)
```

```

decision_tree_params <- parameters(decision_model) |>
  update(
    cost_complexity = cost_complexity(),
    tree_depth = tree_depth(),
    min_n = min_n()
  )

```

Warning: `parameters.model\_spec()` was deprecated in tune 0.1.6.9003.  
 i Please use `hardhat::extract\_parameter\_set\_dials()` instead.

```

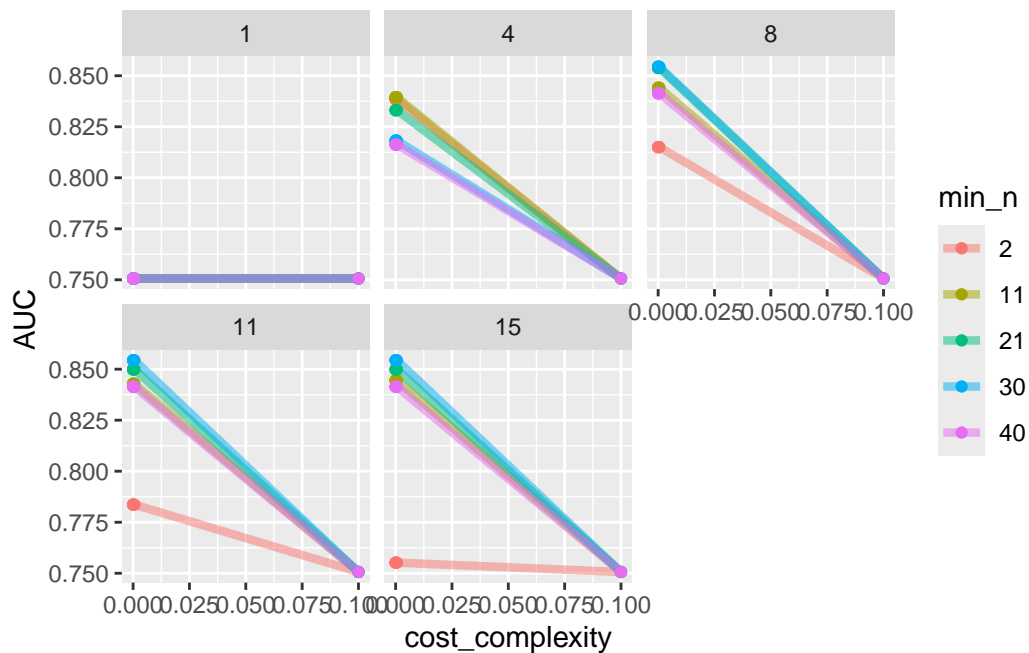
decision_grid <- grid_regular(
  decision_tree_params,
  levels = 5
)

doParallel::registerDoParallel()

tune_decision <- tune_grid(
  decision_wflow,
  resamples = folds,
  grid = decision_grid,
)

tune_decision |>
  collect_metrics() |>
  filter(.metric == "roc_auc") |>
  mutate(min_n = factor(min_n)) |>
  ggplot(aes(cost_complexity, mean, color = min_n)) +
  geom_line(alpha = 0.5, size = 1.5) +
  geom_point() +
  labs(y = "AUC") +
  facet_wrap(vars(tree_depth))

```



Lar tidymodels ta seg av ranges på ulike parameterer. Noe som vi ikke gjorde for random forest eller lasso.

```
best_cost_depth_min <- select_best(tune_decision, metric = "roc_auc")
best_cost_depth_min
```

```
# A tibble: 1 x 4
  cost_complexity tree_depth min_n .config
      <dbl>         <int> <int> <chr>
1 0.00000000001         8    30 Preprocessor1_Model086
```

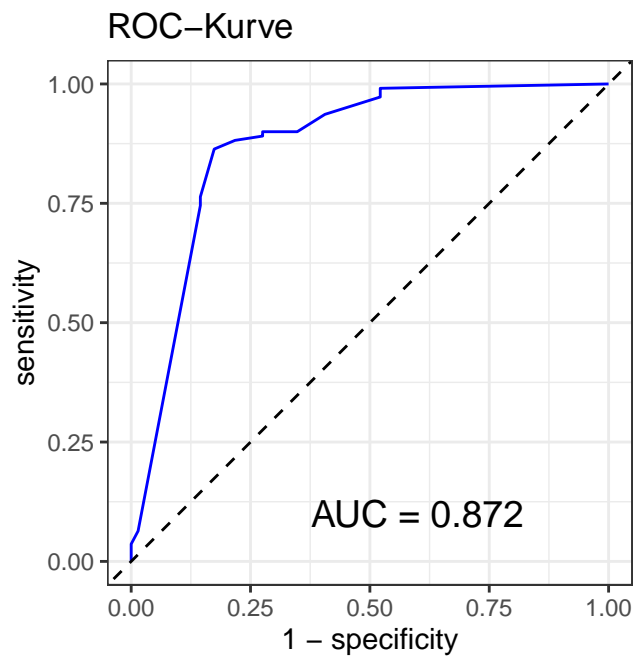
```
decision_fit <- finalize_workflow(decision_wflow, best_cost_depth_min) |>
  fit(titanic_train)
```

```
predict(decision_fit, titanic_test) |>
  bind_cols(titanic_test) |>
  accuracy(Survived, .pred_class)
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>    <chr>         <dbl>
1 accuracy binary      0.827
```

Dårligste Accuarcy så langt

```
auc_decision <-  
  predict(decision_fit, titanic_test, type = "prob") |>  
  bind_cols(titanic_test) |>  
  roc_auc(Survived, .pred_0) |>  
  pull(.estimate)  
  
decision_roc <- predict(decision_fit, titanic_test, type = "prob") |>  
  bind_cols(titanic_test) |>  
  roc_curve(Survived, .pred_0)  
  
decision_roc|>  
  ggplot(aes(x = 1 - specificity, y = sensitivity)) +  
  geom_path(col = "blue") +  
  geom_abline(slope = 1, linetype = "dashed") +  
  coord_equal()+  
  annotate("text", x = 0.6, y = 0.1,  
          label = paste("AUC =", round(auc_decision,3) ), size = 5) +  
  theme_bw() +  
  ggtitle("ROC-Kurve")
```



```

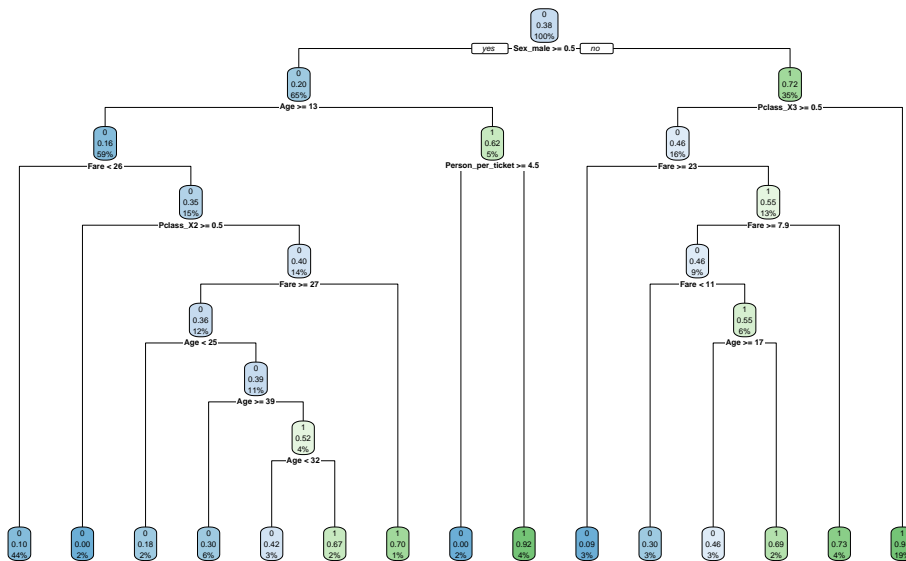
library(rpart)
library(rpart.plot)

decision_final_fit <- finalize_workflow(decision_wflow, best_cost_depth_min) |>
  last_fit(titanic_split)

final_tree <- extract_workflow(decision_final_fit)

final_tree |>
  extract_fit_engine() |>
  rpart.plot(roundint = FALSE)

```

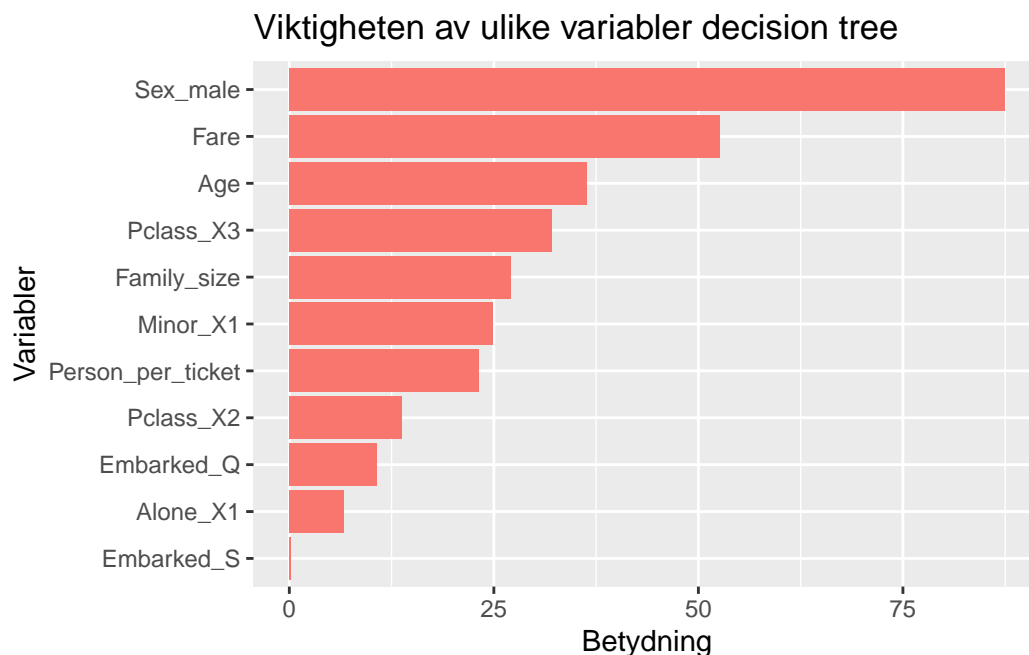


For å bruke funksjonen til rpart, må vi fitte på en annen måte, vi kunne ha brukt denne metodene med last\_fit gjennom hele, men vi syntes det var greiere å kun jobbe med fit, isteden for å kjøre last\_fit

```

decision_fit |>
  extract_fit_parsnip() |>
  vip::vi() |>
  ggplot(aes(y = reorder(Variable, Importance), x = Importance, fill = "#F8766D")) +
    geom_col() +
    theme(legend.position = "none") +
    ggtitle("Viktigheten av ulike variabler decision tree") +
    xlab("Betydning") +
    ylab("Variabler")

```



```

prediksjoner_decision <- predict(decision_fit, titanic_test) |>
  bind_cols(titanic_test)

forvirringsmatrise_decision <- prediksjoner_decision |>
  conf_mat(Survived, .pred_class)

riktigmatrise_decision <- as.tibble(forvirringsmatrise_decision$table) |>
  mutate(Prediction = ifelse(Prediction == 0, "Not survived", "Survived")) |>
  mutate(Truth = ifelse(Truth == 0, "Not survived", "Survived"))

plot_confusion_matrix(riktigmatrise_decision, "Prediction", "Truth", "n",
  add_normalized = F,
  add_row_percentages = F,
  add_col_percentages = F) +
  xlab("Faktisk") +
  ylab("Predikert") +
  ggtitle("Forvirringsmatrise Decision tree")

```

Warning in plot\_confusion\_matrix(riktigmatrise\_decision, "Prediction", "Truth",  
: 'ggimage' is missing. Will not plot arrows and zero-shading.

Warning in plot\_confusion\_matrix(riktigmatrise\_decision, "Prediction", "Truth",  
: 'rsvg' is missing. Will not plot arrows and zero-shading.



## Forvirringsmatrise Decision tree

		Faktisk	
		Survived	Not survived
Predikert	Survived	50	19
	Not survived	12	98

## Xgboost

```
xg_model <- boost_tree(  
  trees = 1000,  
  tree_depth = tune(),  
  learn_rate = tune(),  
  mtry = tune(),  
  min_n = tune(),  
  loss_reduction = tune(),  
  sample_size = tune()  
)|>  
set_engine("xgboost") |>  
set_mode("classification")  
  
xg_wflow <- workflow() |>  
  add_recipe(titanic_recipe) |>  
  add_model(xg_model)  
  
xg_grid <- grid_latin_hypercube(  
  tree_depth(),  
  learn_rate(),
```

```

    finalize(mtry(), titanic_train),
    min_n(),
    loss_reduction(),
    sample_size = sample_prop(),
    size = 50
  )

```

Warning: `grid\_latin\_hypercube()` was deprecated in dials 1.3.0.  
 i Please use `grid\_space\_filling()` instead.

```
doParallel::registerDoParallel()
```

```

tune_xg <- tune_grid(
  xg_wflow,
  resamples = folds,
  grid = xg_grid,
)

```

```

best_param_xg <- select_best(tune_xg, metric = "roc_auc")
best_param_xg

```

```

# A tibble: 1 x 7
  mtry min_n tree_depth learn_rate loss_reduction sample_size .config
<int> <int>    <int>      <dbl>      <dbl>      <dbl> <chr>
1     6     9      11      0.0991      0.00000154  0.483 Preprocessor1_Mo~

```

grid: <https://juliasilge.com/blog/xgboost-tune-volleyball/>, blir litt vanskelig å plotte

```

xg_fit <- finalize_workflow(xg_wflow, best_param_xg) |>
  fit(titanic_train)

```

```

predict(xg_fit, titanic_test) |>
  bind_cols(titanic_test) |>
  accuracy(Survived, .pred_class)

```

```

# A tibble: 1 x 3
  .metric .estimator .estimate
<chr>    <chr>      <dbl>
1 accuracy binary      0.855

```

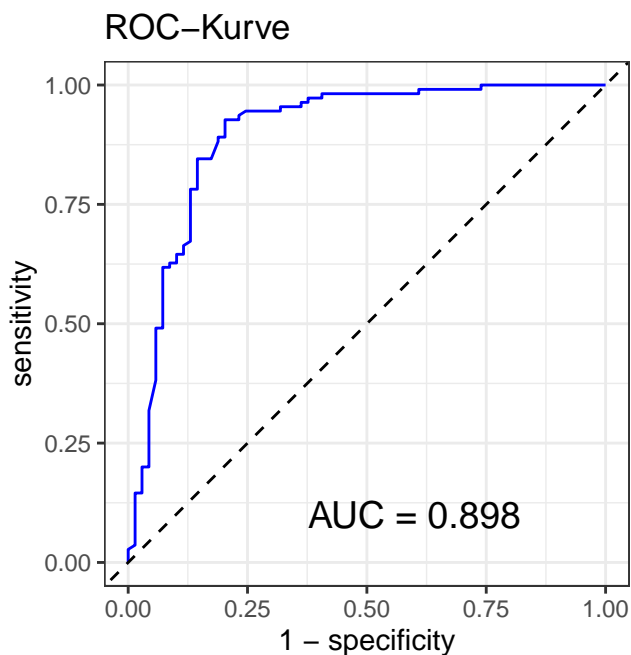
```

auc_xg <-
  predict(xg_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
  roc_auc(Survived, .pred_0) |>
  pull(.estimate)

xg_roc <- predict(xg_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
  roc_curve(Survived, .pred_0)

xg_roc|>
  ggplot(aes(x = 1 - specificity, y = sensitivity)) +
  geom_path(col = "blue") +
  geom_abline(slope = 1, linetype = "dashed") +
  coord_equal()+
  annotate("text", x = 0.6, y = 0.1,
          label = paste("AUC =", round(auc_xg,3) ), size = 5) +
  theme_bw() +
  ggtitle("ROC-Kurve")

```

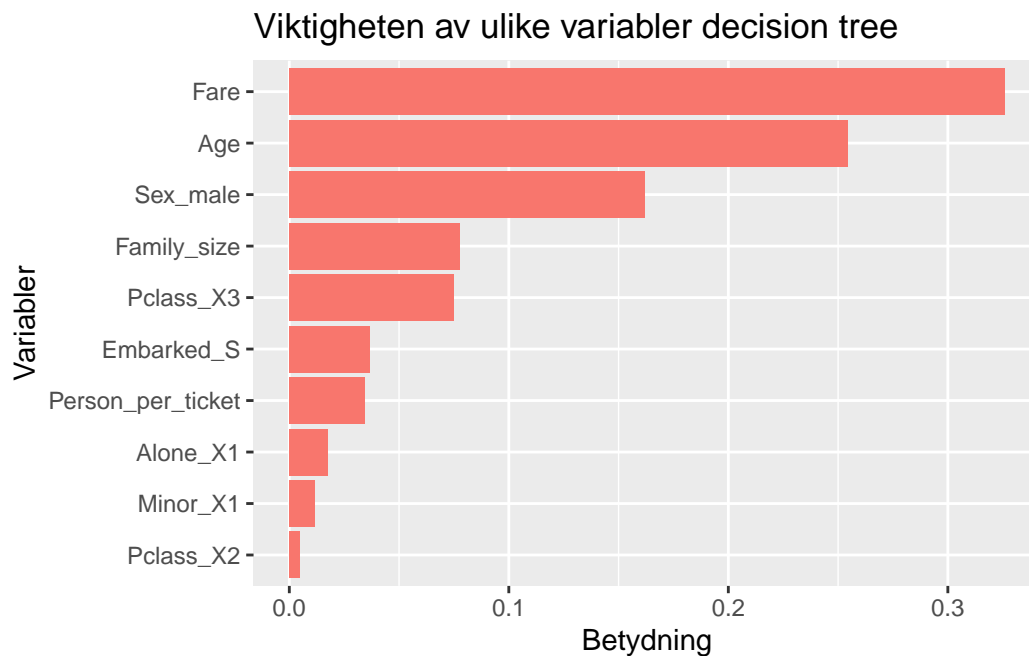


```

xg_fit |>
  extract_fit_parsnip() |>
  vip::vi() |>

```

```
ggplot(aes(y = reorder(Variable, Importance), x = Importance, fill = "#F8766D")) +
  geom_col() +
  theme(legend.position = "none") +
  ggtitle("Viktigheten av ulike variabler decision tree") +
  xlab("Betydning") +
  ylab("Variabler")
```



```
prediksjoner_xg <- predict(xg_fit, titanic_test) |>
  bind_cols(titanic_test)

forvirringsmatrise_xg <- prediksjoner_xg |>
  conf_mat(Survived, .pred_class)

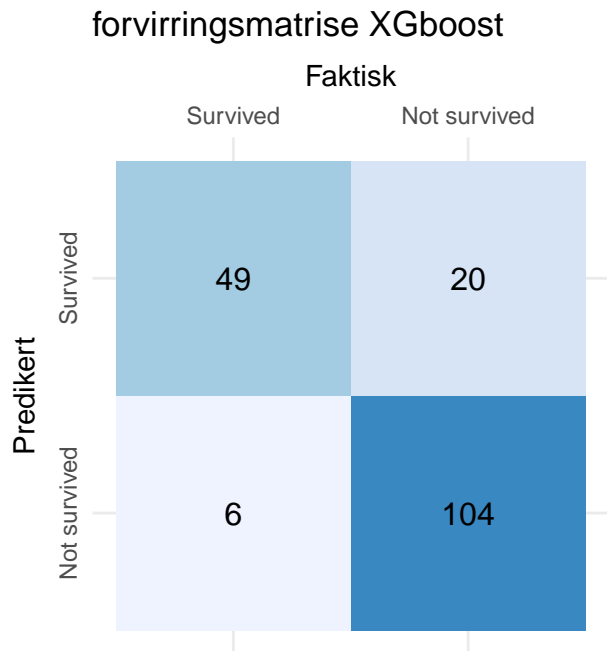
riktigmatrise_xg <- as.tibble(forvirringsmatrise_xg$table) |>
  mutate(Prediction = ifelse(Prediction == 0, "Not survived", "Survived")) |>
  mutate(Truth = ifelse(Truth == 0, "Not survived", "Survived"))

plot_confusion_matrix(riktigmatrise_xg, "Prediction", "Truth", "n",
  add_normalized = F,
  add_row_percentages = F,
  add_col_percentages = F) +
  xlab("Faktisk") +
```

```
ylab("Predikert") +
ggtitle("forvirringsmatrise XGboost")
```

Warning in plot\_confusion\_matrix(riktigmatrise\_xg, "Prediction", "Truth", :  
'ggimage' is missing. Will not plot arrows and zero-shading.

Warning in plot\_confusion\_matrix(riktigmatrise\_xg, "Prediction", "Truth", :  
'rsvg' is missing. Will not plot arrows and zero-shading.



## MLP

Starter med å definere modell, og lage en workflow.

```
mlp.model <- mlp(hidden_units = tune(), penalty = tune(), epochs = tune()) |>
  set_engine("nnet") |>
  set_mode("classification")

mlp_wflow <- workflow() |>
  add_model(mlp.model) |>
  add_recipe(titanic_recipe)
```

Henter så ut hyperparameterne og implepenterer de i en grid ved hjelp av `grid_latin_hypercube` med 50 nivåer

```
mlp_params <- extract_parameter_set_dials(mlp_wflow)
metrics <- metric_set(roc_auc, accuracy, brier_class)

grid <- grid_latin_hypercube(
  mlp_params,
  size = 50
)

doParallel::registerDoParallel()

mlp_tune <- tune_grid(
  mlp_wflow,
  resamples = folds,
  grid = grid,
  metrics = metrics,
  control = control_grid(save_pred = TRUE)
)
```

Henter deretter ut de beste parameterne, og tilpasser modellen

```
logistic_param.reg <- select_best(mlp_tune, metric = "roc_auc") |>
  select(-.config)

fn.mlp_wflow <- mlp_wflow |>
  finalize_workflow(logistic_param.reg)

fn.mlp_fit <- fn.mlp_wflow |>
  fit(titanic_train)
```

Kjører prediksjoner, og plotter ROC-AUC.

```
auc_mlp <-
  predict(fn.mlp_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
  roc_auc(Survived, .pred_0) |>
  pull(.estimate)

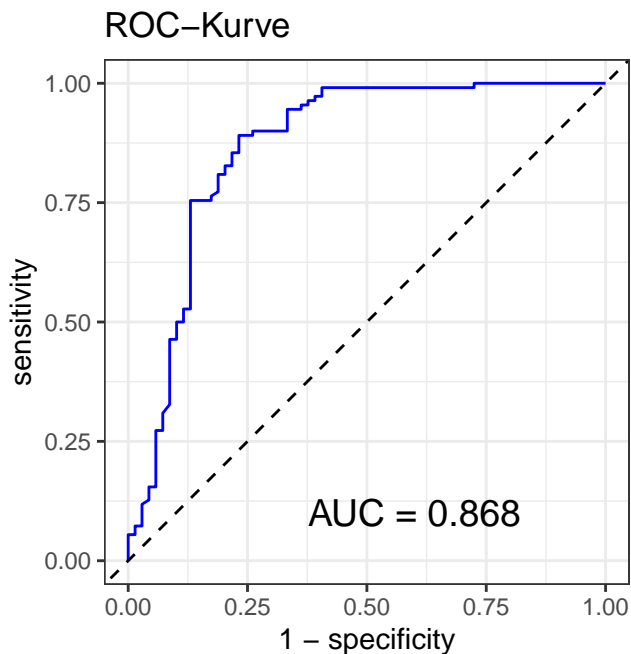
mlp_roc <- predict(fn.mlp_fit, titanic_test, type = "prob") |>
  bind_cols(titanic_test) |>
```

```

roc_curve(Survived, .pred_0)

mlp_roc |>
  ggplot(aes(x = 1 - specificity, y = sensitivity)) +
  geom_path(col = "blue") +
  geom_abline(slope = 1, linetype = "dashed") +
  coord_equal()+
  annotate("text", x = 0.6, y = 0.1,
          label = paste("AUC =", round(auc_mlp, 3) ), size = 5) +
  theme_bw() +
  ggtitle("ROC-Kurve")

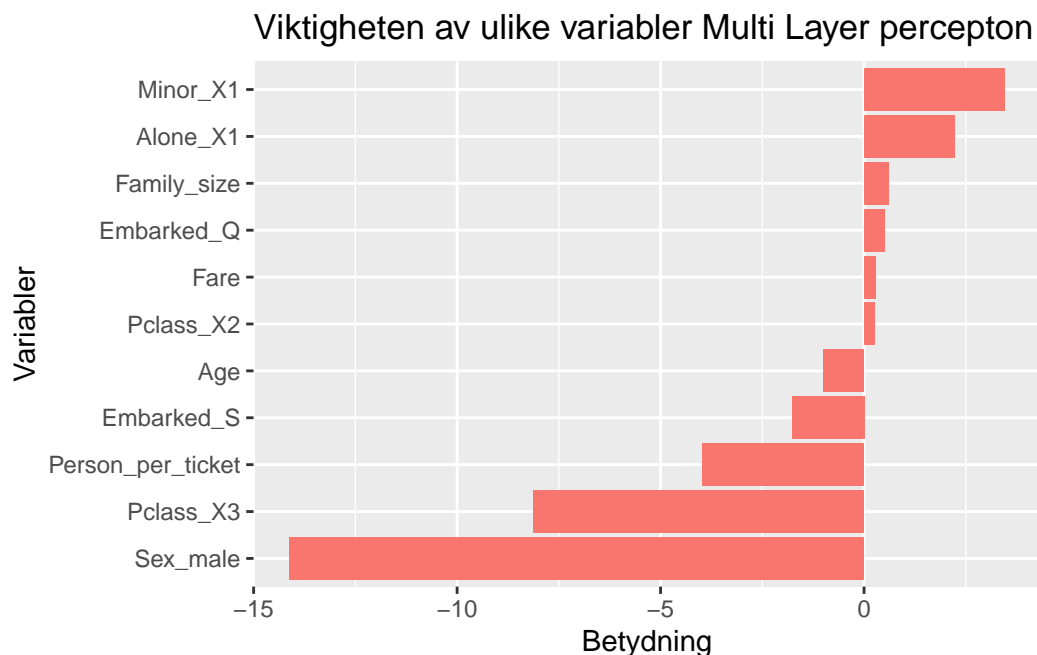
```



```

fn.mlp_fit |>
  extract_fit_parsnip() |>
  vip::vi() |>
  ggplot(aes(y = reorder(Variable, Importance), x = Importance, fill = "#F8766D")) +
  geom_col() +
  theme(legend.position = "none") +
  ggtitle("Viktigheten av ulike variabler Multi Layer perceptron") +
  xlab("Betydning") +
  ylab("Variabler")

```



```

prediksjoner_mlp <- predict(fn.mlp_fit, titanic_test) |>
  bind_cols(titanic_test)

forvirringsmatrise_xg <- prediksjoner_xg |>
  conf_mat(Survived, .pred_class)

riktigmatrise_xg <- as.tibble(forvirringsmatrise_xg$table) |>
  mutate(Prediction = ifelse(Prediction == 0, "Not survived", "Survived")) |>
  mutate(Truth = ifelse(Truth == 0, "Not survived", "Survived"))

plot_confusion_matrix(riktigmatrise_xg, "Prediction", "Truth", "n",
  add_normalized = F,
  add_row_percentages = F,
  add_col_percentages = F) +
  xlab("Faktisk") +
  ylab("Predikert") +
  ggtitle("Forvirringsmatrise Multi-Layer Perceptron")

```

Warning in plot\_confusion\_matrix(riktigmatrise\_xg, "Prediction", "Truth", :  
'ggimage' is missing. Will not plot arrows and zero-shading.

Warning in plot\_confusion\_matrix(riktigmatrise\_xg, "Prediction", "Truth", :  
'rsvg' is missing. Will not plot arrows and zero-shading.



## Forvirringsmatrise Multi-Layer Percepton

		Faktisk	
		Survived	Not survived
Predikert	Survived	49	20
	Not survived	6	104

### Kilder

- Kaggle. *Titanic - Machine Learning from Disaster*. Kaggle. Hentet 1. november 2024 fra <https://www.kaggle.com/competitions/titanic>
- Allohvk. *Titanic - Advanced EDA*. Kaggle. Hentet 1. november 2024 fra <https://www.kaggle.com/code/allohvk/titanic-advanced-eda?scriptVersionId=77739368>
- Donges, N. (2018). *Predicting the survival of Titanic passengers*. Towards Data Science. Hentet 4. november 2024 fra <https://towardsdatascience.com/predicting-the-survival-of-titanic-passengers-30870ccc7e8>