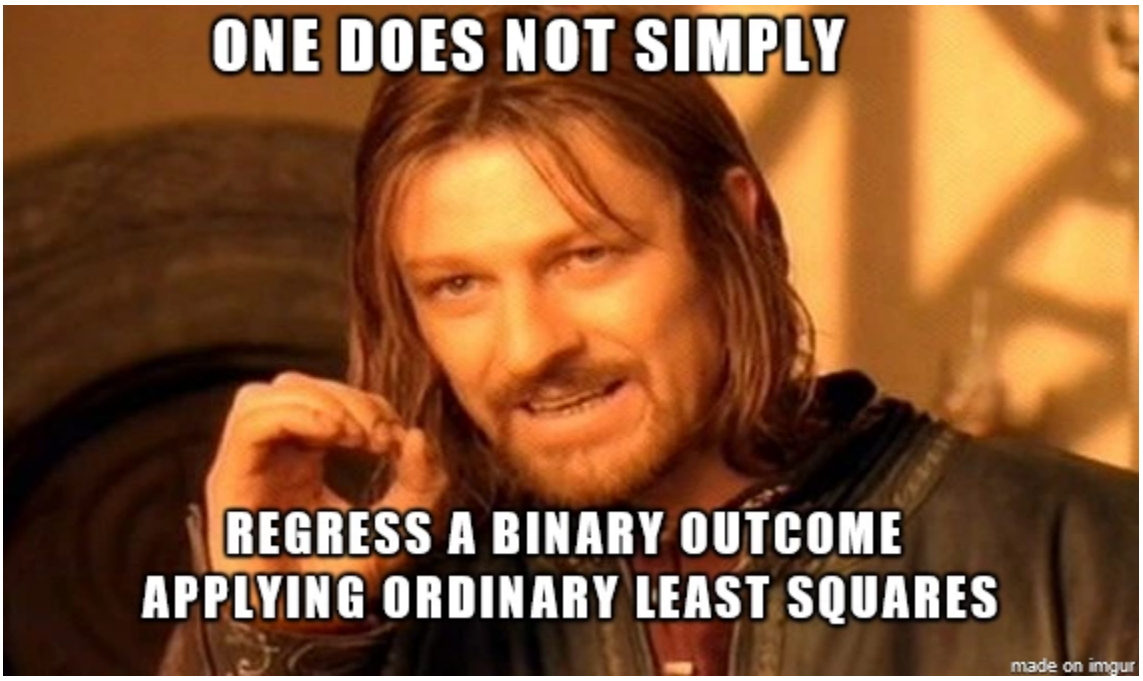
Seminar 4

Eric

3 9 2020

Velkommen til siste seminar! Nå har vi jo fått gjort nesten alt vi skal gå igjennom! Det siste vi skal se på her idag er logistisk regresjon. Som dere sikkert husker fra forelesning er logistisk regresjon en måte å analysere sansyneligheten for et dikotomt utfall, altså på data der vi kun har 0/1 observasjoner. Veldig mye data i statsvitenskap er organisert på denne måten. Vi kan være innteresert i hvorvidt en krig skjedde, eller ikke skjedde, om et individ stemte eller ikke, eller om en fredsavtale ble undertegnet eller ei. For disse typer data vil **OLS** gi feil estimater, med f.eks. sansyneligheter over eller under *0*. Logistisk regresjon gjør at vi kan ta høyde for nettopp det at vi her snakker om sansyneligheter, hvor *0* og *1* er de laveste og høyste verdiene vi kan få.



Som vanlig nå vi skal arbeide med data må vi først laste inne de datene vi ønsker, samt pakker vi vil bruke. La oss denne gangen se på Frankrike i runde 8.

```
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts -------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()    masks stats::lag()
```

```
library(stargazer)
```

```
##
## Please cite as:
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

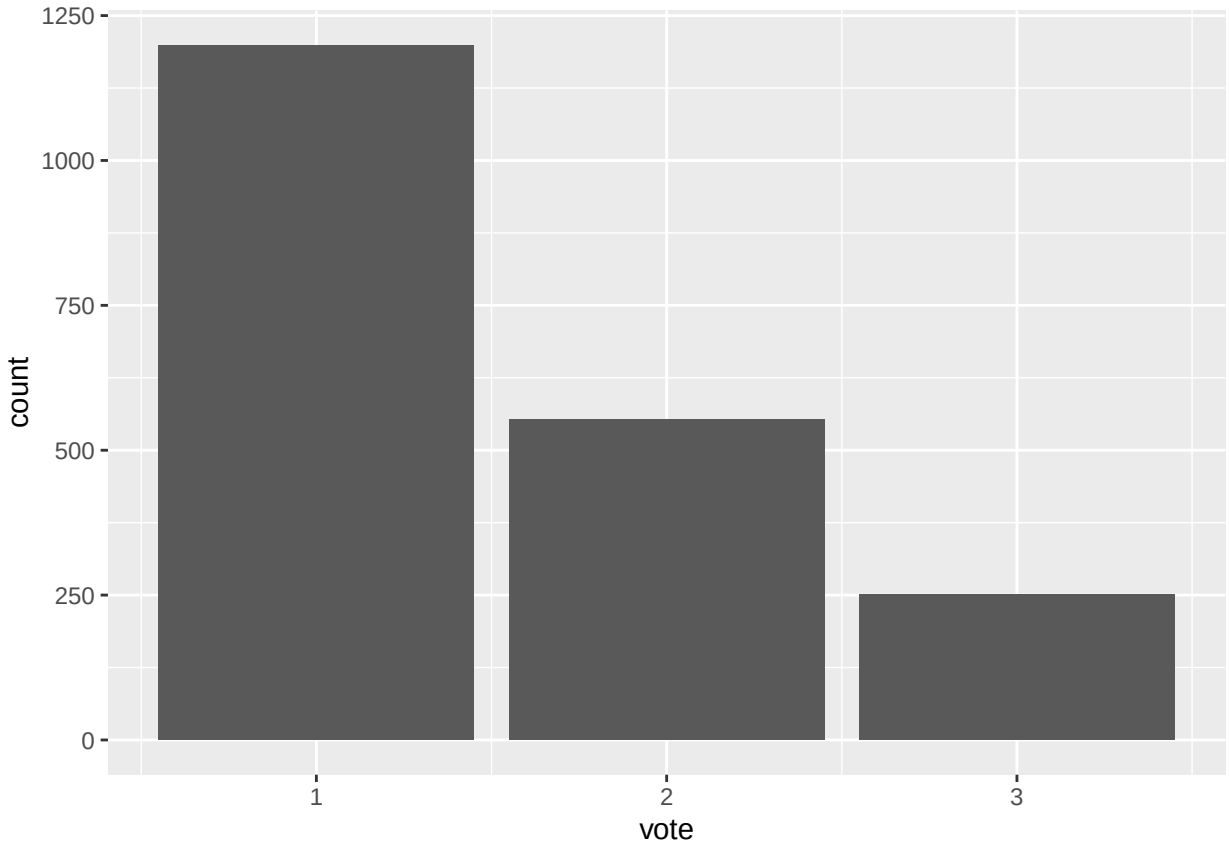
```
ESS <- read.csv("https://raw.githubusercontent.com/egen97/4020A_RSeminar/master/ESS_Selected.csv")

#Nå kan vi hente ut Frankrike runde 8

France_8 <- ESS %>%
  filter(essround == 8 & Country == "FR")
```

Som vi kan se har vi nå et datasett med rett over 2000 observasjoner, og 24 variabler. Om du har glemt hva variablene betyr kan du sjekke ut scriptet til 1. seminar. Den uavhengige variabelen vi ønsker å undersøke er hvorvidt folk i Frankrike valgte å stemme, eller ikke. Dette er logisk nok en dikotom variabel, så vi kan først se hvor mange som har stemt gjennom et stolpediagram:

```
ggplot(France_8, aes(vote)) +
  geom_bar()
```



Her er det åpenbart noe feil, hva skal 3 bety? I ess har de i tilegg til hvorvidt de har stemt eller ikke, valgt å ha med hvorvidt de ikke hadde stemmerett i siste valg, f.eks. pga. alder eller statsborgerskap. For oss er dette egentlig ganske uinteresant informasjon, så vi kan derfor heller bare sette disse til `NA`. Sånn som dataene

er nå er det også *1* som er stemt, og *2* som er ikke stemt, her vil vi nok forandre det så *1* betyr stemt, og *0* betyr ikke stemt.

```
France_8 <- France_8 %>%
  mutate(vote = ifelse(vote == 3, NA, vote),
         vote = ifelse(vote == 2, 0, vote))
```

Samtidig kan det være innteresant å vite hvor mange som ikke har svart! Selvom vi har over 2K observasjoner, er det ikke dette mye verdt om 90 % er `NA`. En fin ting med `summary()` funksjonen er at den rapporterer nettopp NA, i tilegg til statistikk som gjennomsnitt, og median også viser antall `NA`i variabelen!

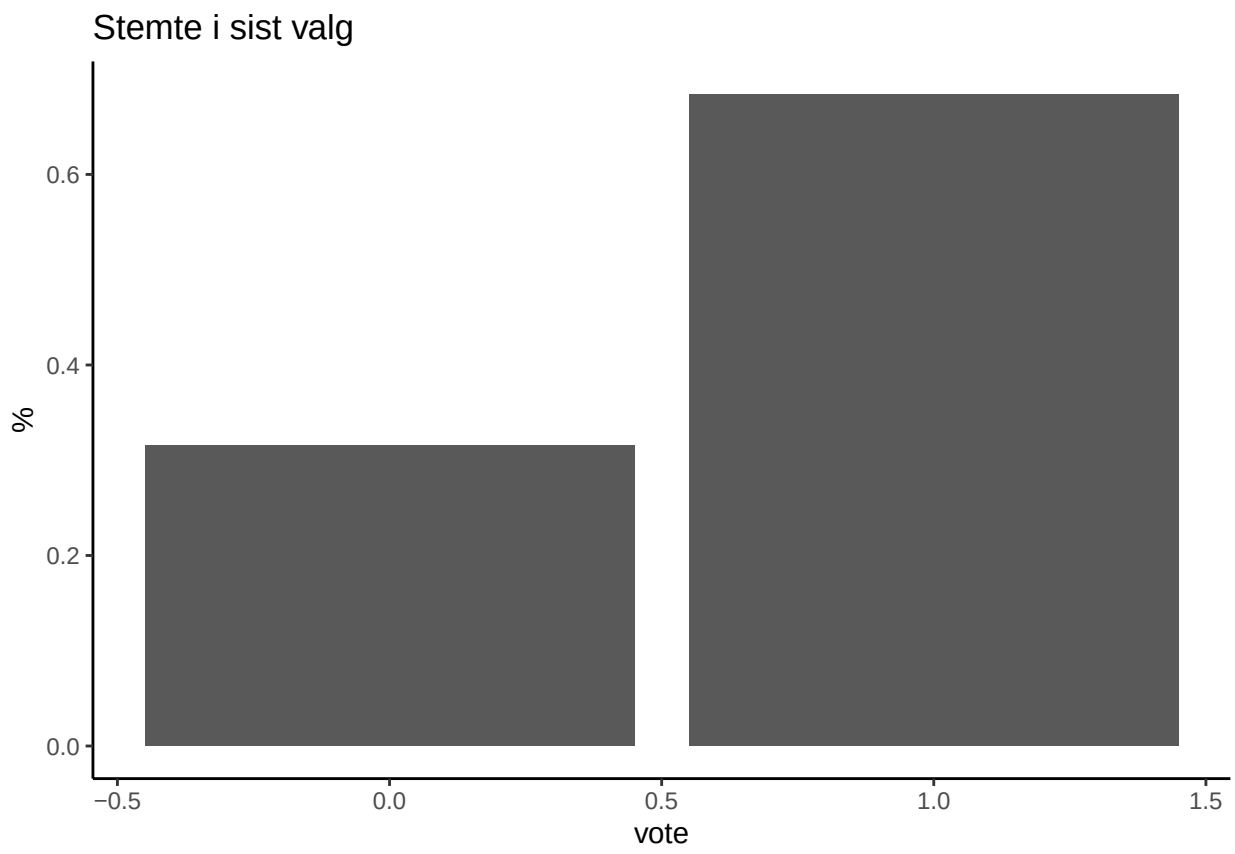
```
summary(France_8$vote)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.0000  0.0000  1.0000  0.6844  1.0000  1.0000     318
```

Her kan vi se at vi har 318 `NA`, det er nok så lite at det ikke har noe særlig å si for resultatet vårt. Vi kan også se at gjennomsnittet er ca. *0,68*, kan vi tolke det på noen måte? Vel, det betyr faktisk at *68%* av de som er i datasettet stemte, **og** at at om vi trekker ett tilfeldig individ er det **68%** sansynelighet for at de har stemt! La oss nå se om vi kan lage et sånt plott igjen:

```
ggplot(France_8, aes(vote)) +
  geom_bar(aes(y = (..count..)/sum(..count..))) + #Ved å legge til dette kan vi % istedenfor
  theme_classic() +                               #reele tall! Det er ofte mer innteresant.
  ggtitle("Stemte i sist valg") +
  ylab("%")
```

```
## Warning: Removed 318 rows containing non-finite values (stat_count).
```



Logistisk regresjon

Nå som vi har en avhengig variabel, må vi bestemme oss for hva tror påvirker denne. Her kommer vi til å se på alder, tilitt til politikere, inntekt, og kjønn. Tenk gjerne over hvilke resultater du forventer før du gjennomfører regresjonen! Når vi faktisk gjør en analyse til en oppgave, eller selve masteroppgaven vil du først måtte sette en hypotese før du i det hele tatt samler inn data!

Å gjøre selve regresjonen er relativt enkelt, og dere vil kjenne igjen mye fra når vi jobbet med OLS modeller. Det er to vikige forskjeller:

- For det første må vi nå bruke det som heter en "generalized linear model", med funksjonen `glm()`heller enn `lm()`. En generalisert linærmodell er en regresjonsmodell som kan ha ikke-linære variabler.
- Det andre vi må gjøre er å spesifisere hvilken type regresjon vi ønsker. Her vil vi gjøre en binomisk logistisk regresjon, og det må vi spesifisere i modellen.

Resten vil være ganske likt med hvordan vi gjorde det i OLS. Den virkelige forskjellen kommer når vi skal tolke resultatene! La oss nå først lage en modell med kun alder som predikator.

```
Mod1 <- glm(vote ~ Age, data = France_8, family = binomial(link = "logit"))
stargazer(Mod1, type = "text")
```

```
##
## ===============================================
##                         Dependent variable:
##                     ---------------------------
##                                vote
## -----------------------------------------------
## Age                          0.034***
##                               (0.003)
##
## Constant                     -1.033***
##                               (0.176)
##
## -----------------------------------------------
## Observations                   1,751
## Log Likelihood              -1,032.815
## Akaike Inf. Crit.            2,069.629
## ===============================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

Hva kan vi egentlig tolke ut av denne modellen? Vel, vi har en positiv og signifikant koeffisient for alder. Det er alt vi kan si utifra denne tabellen. Når vi analyserer logistiske regresjoner er de vikig å huske hvilke skalaer du ser dataene på, i denne tabellen er koeffisienten på logoddsskalaen. Ønsker vi å tolke den substansielt bør vi heller gjør den om til odds, eller sannsynelighet. Før vi gjør det kan vi prøve å lage en ny modell med alle variablene i.

```
Mod2 <- glm(vote ~ Age + Trust_Politicians +
              Income + Gender, data = France_8, family = binomial(link = "logit"))
stargazer(Mod2, type = "text")
```

```
##
## ===============================================
##                         Dependent variable:
##                     ---------------------------
##                                vote
## -----------------------------------------------
## Age                          0.040***
```

```
##                               (0.004)
##
## Trust_Politicians           0.131***
##                               (0.029)
##
## Income                      0.155***
##                               (0.022)
##
## Gender                       -0.219*
##                               (0.118)
##
## Constant                    -2.074***
##                               (0.298)
##
## ---------------------------------------------
## Observations                   1,601
## Log Likelihood              -891.050
## Akaike Inf. Crit.           1,792.099
## =============================================
## Note:              *p<0.1; **p<0.05; ***p<0.01
```

Hvordan kan vi vite om denne modellen er bedre enn den andre? Disse to modellene er "nøstede", eller *nested* som det heter på nynorsk. Det betyr at alle variablene som er i den "lille" modellen også er i den "større". Siden det er tilfellet kan vi sammenligne dem ved hjelpe av *AIC*, hvor vi et at en mindre *AIC* betyr at modellen er bedre. For modell *1* kan vi lese at $AIC = 2069$, og for modell *2* kan vi se at $AIC = 1792$, altså vet vi at den med flere variabler er en bedre modell.

Tolke logistisk regresjon

Logistisk regresjon kan være noe vanskeligere å tolke, enn en vanlig OLS-modell. Et viktig poeng er at økningen i Y for en gitt økning i X er avhengig **både** av verdien på alle andre uavhengige variabler, **og** verdien på X. Altså må vi på en eller annen måte ta hensyn til dette når vi skal tolke modellen vår. Her skal vi se på to muligheter, først å se på endring i odds, altså oddsratio, og så på endring i sansynelighet gitt et case.

Oddsratio

Tilitt til politikere er målt i en skala fra 1 (ingen tilitt) til 10 (fullstendig tilitt). La oss si at vi her er innteresert i hvor mye oddsen for å stemme øker når du har ett skalapunkt høyere tilitt til politikere. Det vi da vil ha er antilogaritmen til koeffisienten, ganger med hundre minus 1. La oss ta det trinnvis, vi brukes `exp()` for å få antilogaritmen, og `coef()` for å hente ut koeffisienten vi er innteresert i.

```
exp(coef(Mod2)[3]) #Trust_Politicians er den 3. koefissienten, derfor [3]
```

```
## Trust_Politicians
##          1.139969
```

Her kan vi se at vi får resultatet *1.13*, alså hos respondenter med en skalaenhet høyere tillitt er oddsen for å stemme *1,14* ganger høyere, *ceteris paribus.*

```
(exp(coef(Mod2)[3])*100)
```

```
## Trust_Politicians
##          113.9969
```

Her kan vi se at når vi ganger med hundre får vi tallet *114*. Altså er oddsen for å stemme *114* høyere hos de med en skalaenhet mer tilitt til politikere, *ceteris paribus.*

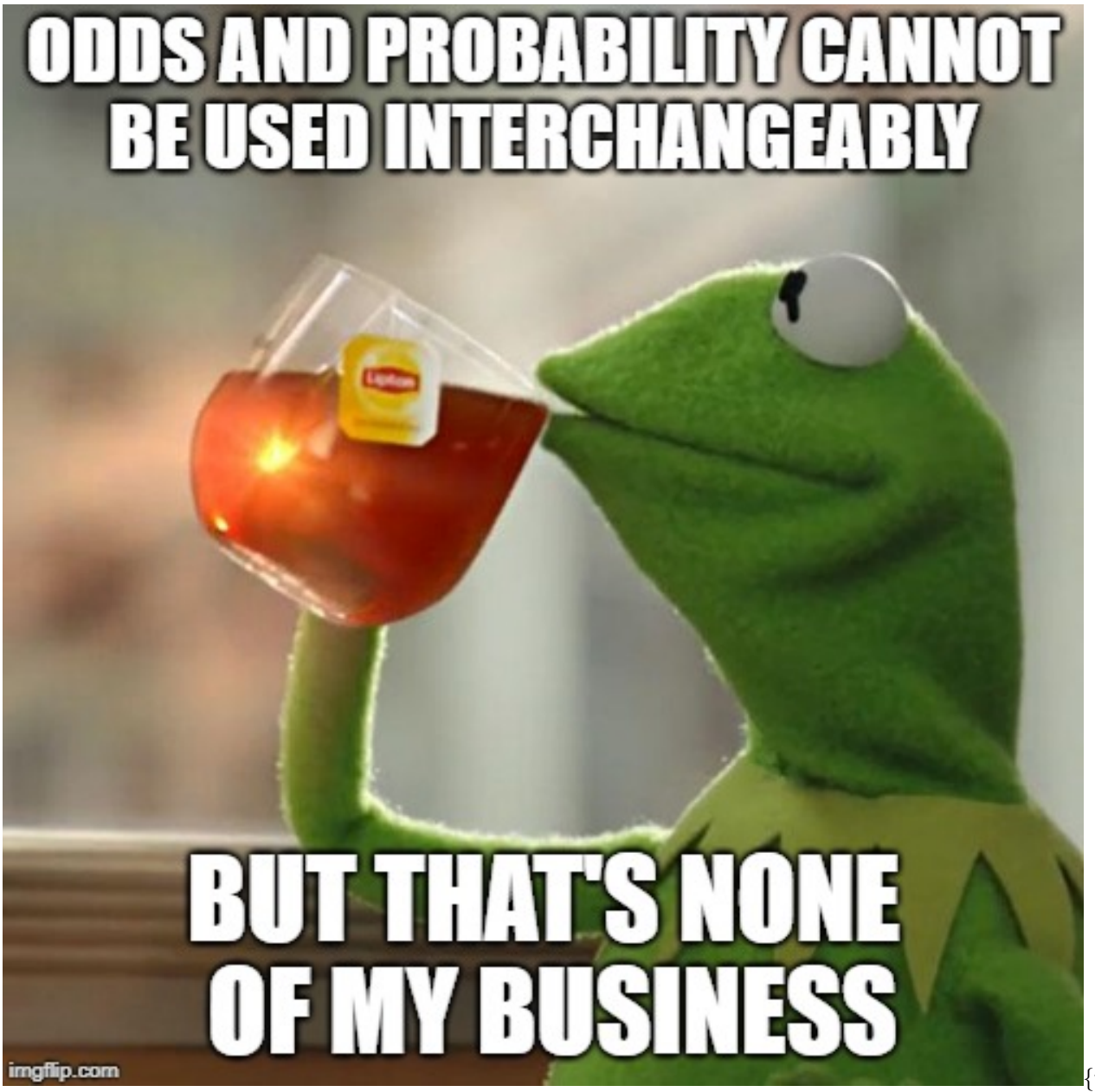
Til slutt kan vi trekke fra *1*:

```
(exp(coef(Mod2)[3])-1)*100
```

```
## Trust_Politicians
##          13.99689
```

Og her ser vi altå at oddsen for å stemme øker med *14%* når tillit til politikere øker med en skalaenhet, *ceteris paribus*.

Omregne til sansynelighet



{width = 50%}

Selvom vi nå har det som odds kan denne ofte være noe vanskelig å forstå, da den ikke gir helt intiuitiv mening. Den beste, om du spør meg, måten å presentere resultatene på er ved å vise et case, og hvordan sansyneligheten øker eller synker for $Y = 1$ gitt dette caset og forskjellige verdier på den uavhengige variabelen

vi er innteresert i. For å gjøre dette må vi logisk nok først sette opp et case, her er det flere forskjellige måter å gjøre det på. Noen velger å ta case de er reelt innteresert i, mens andre bare setter alt på gjennomsnittet. En viktig ting er at alt bør gi substansielt mening, så ikke sett kjønn på *0,8* f.eks.

For å lage et case lager vi rett og slett et datasett som inneholder de variablene vi ønsker, dette kan vi enkelt gjøre med `tibble()` funksjonen.

```
Data_Case <- tibble(
  Age = mean(France_8$Age, na.rm = TRUE),
  Trust_Politicians = seq(min(France_8$Trust_Politicians, na.rm = TRUE),
                          max(France_8$Trust_Politicians, na.rm = TRUE)
                          ), #Her kan dere se at jeg lar tilitt variere fra minimum til max-verdien
  Income = median(France_8$Income, na.rm = TRUE),
  Gender = 1
)
```

Her har jeg altså laget et datasett hvor alt er konstant, unntatt tilitt til politikere som varierer fra minimum *0* til maksimum *10*. Nå ønsker vi å se hvordan sansyneligheten endrer seg for disse gitte verdiene. For å undersøke dette kan vi bruke `predict()` funksjonen, som nettopp predikerer hva sansyneligheter for *Y=1* er for en gitt *X* verdi. I `predict()` må vi også si hva slags data vi ønsker å få ut, her logodds, og om vi vil ha med standardfeil.

```
Predicted_Data <- predict(Mod2, newdata = Data_Case, type = "link", se = TRUE)
```

Nå som vi har de predikerte verdiene kan det være nyttig å legge til disse som kolonner i dataene til caset, det kan vi gjøre med `cbind()` som står for "collum-bind":

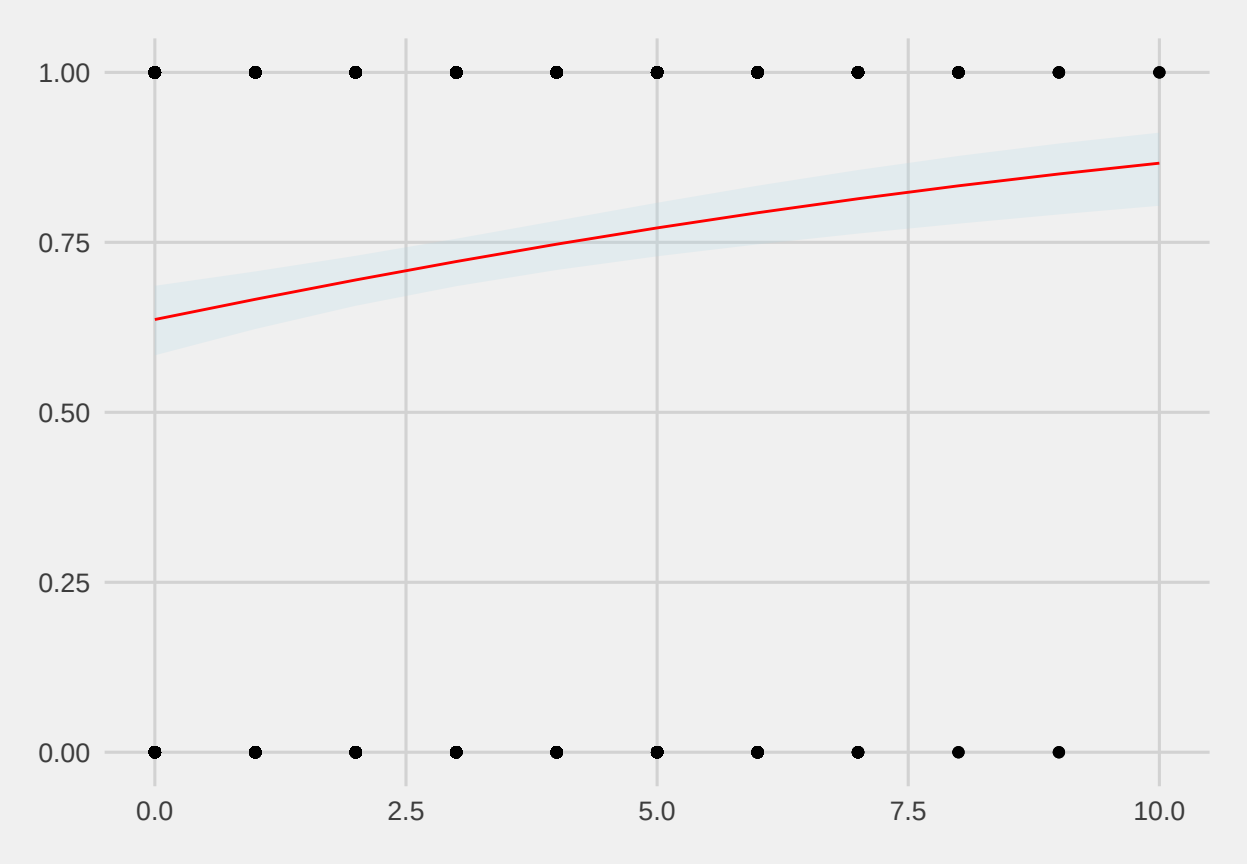
```
Plot_Data <- cbind(Predicted_Data, Data_Case)
```

I objektet vi nå har laget kan vi se at vi har verdiene for de forskjellige casene, i tilegg til tre nye rader som heter "fit", "se.fit", og "residual.scale." I tilegg til selve punkt-estimatet, ønsker vi å bruke standardfeilen til å regne ut usikkerheten til punktestimatet. Derfor vil vi legge til tre nye kolloner som heter "high", "low", and "fit." Fit er bare en omkoding fra logodds til sansynelighet.

```
Plot_Data$low <- exp(Plot_Data$fit - 1.96*Plot_Data$se)/(1+ exp(Plot_Data$fit - 1.96*Plot_Data$se))
Plot_Data$high <- exp(Plot_Data$fit + 1.96*Plot_Data$se)/(1+ exp(Plot_Data$fit + 1.96*Plot_Data$se))
Plot_Data$fit <- exp(Plot_Data$fit)/(1 + exp(Plot_Data$fit))
```

Nå som vi har data både på hva vi tror sansyneligheten er, og usikkerheten kan vi prøve å plotte denne!

```
library(ggthemes)
ggplot(France_8, aes(Trust_Politicians, vote)) +
  geom_rangeframe() +
  geom_point()+
  geom_ribbon(data = Plot_Data, aes(y = fit, ymin = low, ymax = high), alpha = .2, fill = "lightblue")+
  geom_line(data = Plot_Data, aes(y = fit), colour = "red") +
  theme_fivethirtyeight()
```



Assumptions for logistisk

Som all annen regresjon, har logistisk regresjon flere antagelser vi bør sjekke om stemmer for vår modell. Her kan vi først se på hvorvidt den forventede S-formen er tilstede, og om vi har inflytelsesrike uteliggere. Til slutt kan vi se på en ROC-kurve for å teste hvor god modellen vår er.

Rett line på logittskalen

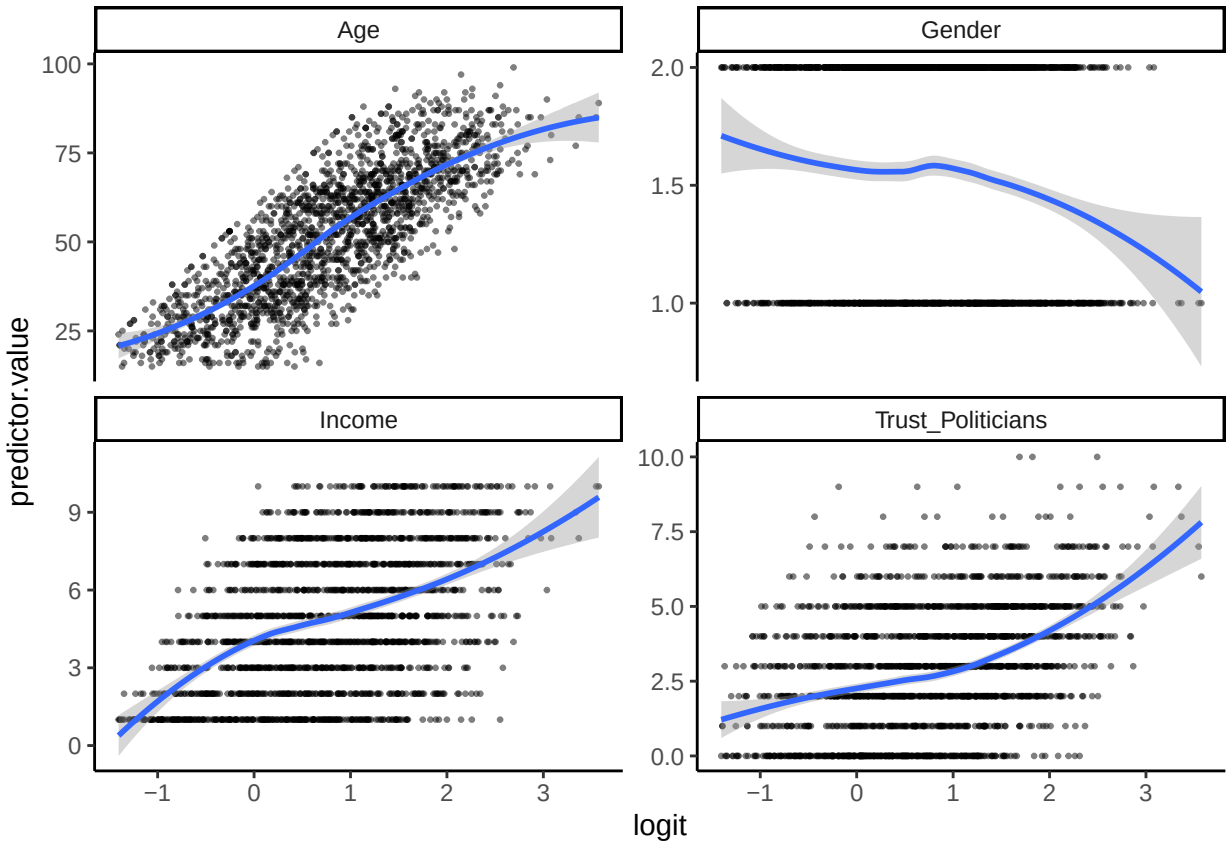
Her vil vi undersøke hvorvidt regresjonen sine uavhengige variabler har en rett linje på logittskalaen. For å gjøre det må vi først hente ut de predikerte verdiene. Disse kan vi så gjøre om til logodds.

```
France_8 %>%
  mutate(preds = predict(Mod2, France_8, type = "response")) %>%
  mutate(logit = log(preds/(1-preds))) %>%
  select(Age, Trust_Politicians, Income, Gender, logit) %>%
  pivot_longer(-logit, names_to = "predictors", values_to = "predictor.value") %>%
  ggplot(aes(logit, predictor.value)) +
  geom_point(size = 0.5, alpha = 0.5) +
  geom_smooth(method = "loess") +
  theme_classic() +
  facet_wrap(~predictors, scales = "free_y")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 784 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 784 rows containing missing values (geom_point).
```



Inflytelsesrike uteliggere

For å finne inflytelsesrike uteliggere kan vi bruke Cooks distance. En fin ting med base R er at `plot`funksjonen kan gjøre dette for oss! Gjennom å sette `which` og `id.n` argumentet får vi frem et plott som viser Cook's distance, og markerer de observasjonene som kan være en kraftig uteliggere.

```
#plot(Mod2, which = 4, id.n = 3)
#dev.off()
```

Nå kan vi f.eks. se på verdiene til observasjon 289. For å gjøre det kan vi bruke stargazer

```
stargazer(France_8[289,], type = "text")
```

```
##
## ===============================================================================
## Statistic                      N  Mean  St. Dev. Min Pctl(25) Pctl(75) Max
## -------------------------------------------------------------------------------
## Time_News                      1 20.000          20     20       20     20
## Trust_People                   1 2.000            2     2        2      2
## People_Fair                    1 8.000            8     8        8      8
## Pol_Interest                   1 3.000            3     3        3      3
## Trust_Police                   1 3.000            3     3        3      3
## Trust_Politicians              1 0.000            0     0        0      0
## vote                           1 0.000            0     0        0      0
## Left_Right                     1 8.000            8     8        8      8
## Satisfied_Gov                  1 2.000            2     2        2      2
## Gov_Reduce_IncomDif            1 4.000            4     4        4      4
```

```
## LGBT_Free                     1 1.000            1     1        1      1
## Religous                      1 0.000            0     0        0      0
## Climate_Human                 1 4.000            4     4        4      4
## Responsibility_Climate        1 5.000            5     5        5      5
## Goverment_Climate             1 2.000            2     2        2      2
## Basic_Income                  1 2.000            2     2        2      2
## Important_Rules               1 2.000            2     2        2      2
## Important_Equal_Oppurtunities 1 3.000            3     3        3      3
## Income                        1 10.000          10     10       10    10
## Gender                        1 1.000            1     1        1      1
## Age                           1 77.000          77     77       77    77
## essround                      1 8.000            8     8        8      8
## ------------------------------------------------------------------------
```

Her kan vi f.eks. se at denne mannen er blant de rikste i landet, og ganske gammel.

ROC-curve

Til slutt kan vi se på ROC-curven. Denne viser hvor god modellen vår er til å predikere hvorvidt en enhet er *1* eller *0* basert på forskjellige kuttpunkt på sansynelighetsskalen. For å få frem denne skal vi bruke pakken `pROC`.

```
#install.packages("pROC")
library("pROC")
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

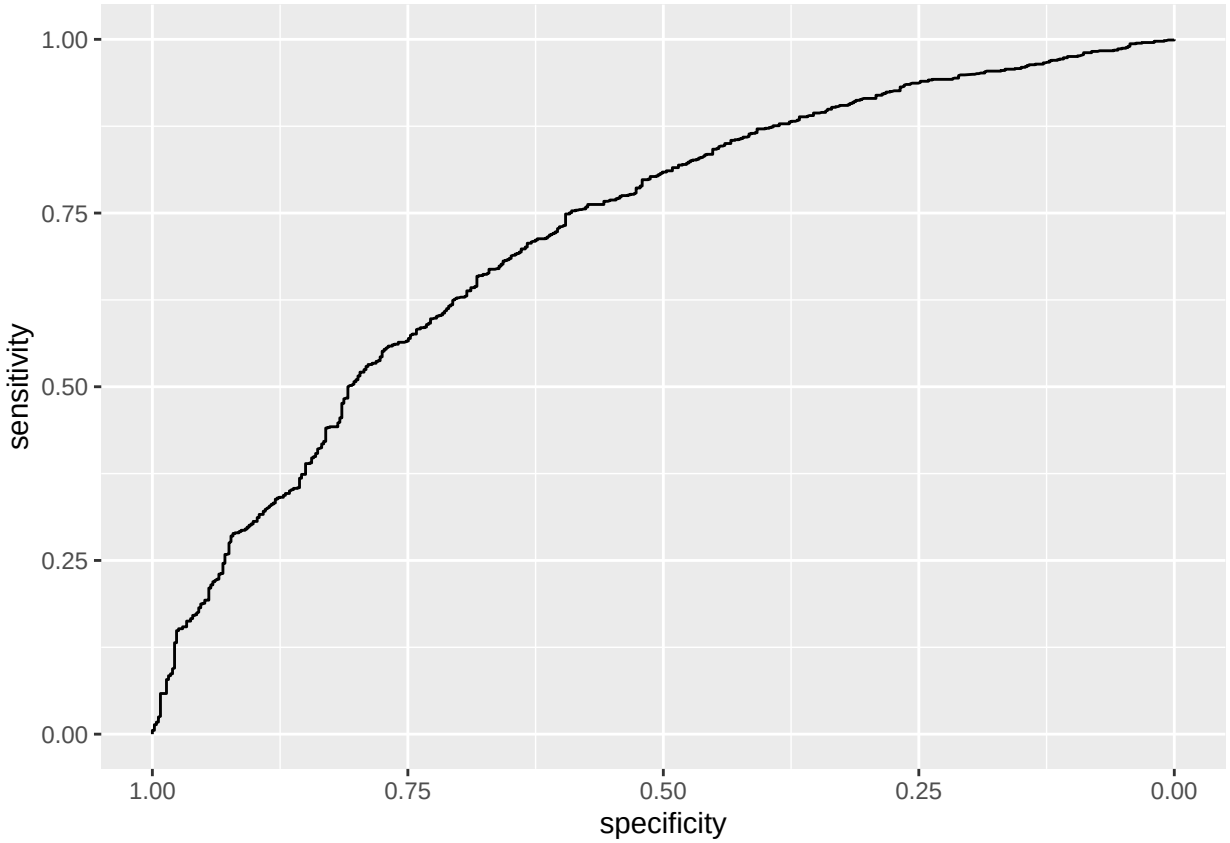
```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
Mod2_Roc <- roc(response = France_8$vote,
                predictor = predict(Mod2, France_8))
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
ggroc(Mod2_Roc)
```


1.00
0.75
0.50
0.25
0.00
sensitivity
1.00
0.75
0.50
0.25
0.00
specificity