Regression Analysis

RESAN S. LIPAR

# install.packages(`tidyverse`)  
# install.packages(`gapminder`)  
# install.packages(`finalfit`)  
# install.packages(`broom`)  
  
  
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.4 v purrr 0.3.4  
## v tibble 3.1.2 v dplyr 1.0.6  
## v tidyr 1.1.3 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(gapminder) # for the dataset  
library(finalfit)  
library(broom)  
  
theme\_set(theme\_bw())

# Load the sample data  
gapdata <- gapminder

#Check the data first  
dim(gapdata)

## [1] 1704 6

names(gapdata)

## [1] "country" "continent" "year" "lifeExp" "pop" "gdpPercap"

head(gapdata)

## # A tibble: 6 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 Afghanistan Asia 1952 28.8 8425333 779.  
## 2 Afghanistan Asia 1957 30.3 9240934 821.  
## 3 Afghanistan Asia 1962 32.0 10267083 853.  
## 4 Afghanistan Asia 1967 34.0 11537966 836.  
## 5 Afghanistan Asia 1972 36.1 13079460 740.  
## 6 Afghanistan Asia 1977 38.4 14880372 786.

glimpse(gapdata) # inspect each variable as line, variable type, first values

## Rows: 1,704  
## Columns: 6  
## $ country <fct> "Afghanistan", "Afghanistan", "Afghanistan", "Afghanistan", ~  
## $ continent <fct> Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, ~  
## $ year <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992, 1997, ~  
## $ lifeExp <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, 38.438, 39.854, 40.8~  
## $ pop <int> 8425333, 9240934, 10267083, 11537966, 13079460, 14880372, 12~  
## $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 786.1134, ~

missing\_glimpse(gapdata) # inspect for missing data for each variable

## label var\_type n missing\_n missing\_percent  
## country country <fct> 1704 0 0.0  
## continent continent <fct> 1704 0 0.0  
## year year <int> 1704 0 0.0  
## lifeExp lifeExp <dbl> 1704 0 0.0  
## pop pop <int> 1704 0 0.0  
## gdpPercap gdpPercap <dbl> 1704 0 0.0

ff\_glimpse(gapdata) # obtain summary statistics for each variable

## $Continuous  
## label var\_type n missing\_n missing\_percent mean  
## year year <int> 1704 0 0.0 1979.5  
## lifeExp lifeExp <dbl> 1704 0 0.0 59.5  
## pop pop <int> 1704 0 0.0 29601212.3  
## gdpPercap gdpPercap <dbl> 1704 0 0.0 7215.3  
## sd min quartile\_25 median quartile\_75 max  
## year 17.3 1952.0 1965.8 1979.5 1993.2 2007.0  
## lifeExp 12.9 23.6 48.2 60.7 70.8 82.6  
## pop 106157896.7 60011.0 2793664.0 7023595.5 19585221.8 1318683096.0  
## gdpPercap 9857.5 241.2 1202.1 3531.8 9325.5 113523.1  
##   
## $Categorical  
## label var\_type n missing\_n missing\_percent levels\_n  
## country country <fct> 1704 0 0.0 142  
## continent continent <fct> 1704 0 0.0 5  
## levels  
## country -  
## continent "Africa", "Americas", "Asia", "Europe", "Oceania"  
## levels\_count levels\_percent  
## country - -  
## continent 624, 300, 396, 360, 24 36.6, 17.6, 23.2, 21.1, 1.4

## Dependent variable : Our target is the life Expectanc variable

## Predictor variables: country, continent, year, pop, gdpPercap

# Objective of Regression Analysis:

Find the best line that fits the data well

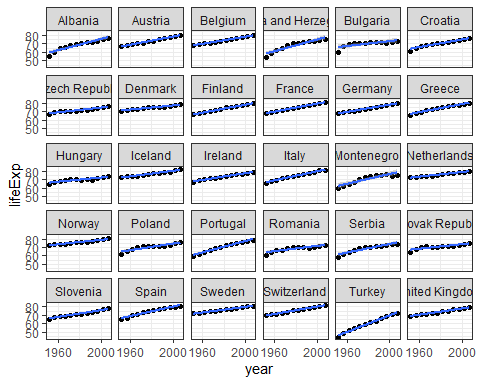
Let’s plot the life expectancies in European countries over the past 60 years, focussing on the UK and Turkey. We can add in simple best fit lines using

geom\_smooth()

## geom\_smooth: na.rm = FALSE, orientation = NA, se = TRUE  
## stat\_smooth: na.rm = FALSE, orientation = NA, se = TRUE  
## position\_identity

gapdata %>%  
 filter(continent == "Europe") %>% # Europe only  
 ggplot(aes(x = year, y = lifeExp)) + # lifeExp~year  
 geom\_point() + # plot points  
 facet\_wrap(~ country) + # facet by country  
 scale\_x\_continuous(  
 breaks = c(1960, 2000)) + # adjust x-axis  
 geom\_smooth(method = "lm") # add regression lines

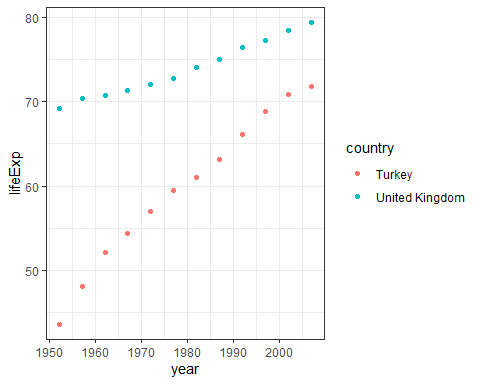
## `geom\_smooth()` using formula 'y ~ x'



# Remarks about the plot of life expectation:

Regression line uses the formula ’y ~ x Plot shows the life expectancy pattern for all countries in the dataset for the time period 1960 up to 2000 Lowest expectancy rate occurs in Turkey on the year 1960

# We focus our attention to Turkey and United Kingdom  
gapdata %>%  
 filter(country %in% c("Turkey", "United Kingdom")) %>%  
 ggplot(aes(x = year, y = lifeExp, colour = country)) +  
 geom\_point()



# Remarks

It is clear now that the highest life expectancy of Turkey is just higher than 70 years which is the lowest rate among the UK people

# We fit a regression line for United Kingdom and obtain the model summary  
fit\_uk <- gapdata %>%  
 filter(country == "United Kingdom") %>%  
 lm(lifeExp~year, data = .)  
fit\_uk %>% summary()

##   
## Call:  
## lm(formula = lifeExp ~ year, data = .)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.69767 -0.31962 0.06642 0.36601 0.68165   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -2.942e+02 1.464e+01 -20.10 2.05e-09 \*\*\*  
## year 1.860e-01 7.394e-03 25.15 2.26e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.4421 on 10 degrees of freedom  
## Multiple R-squared: 0.9844, Adjusted R-squared: 0.9829   
## F-statistic: 632.5 on 1 and 10 DF, p-value: 2.262e-10

# We fit a regression line for Turkey and obtain the model summary  
fit\_turkey <- gapdata %>%  
 filter(country == "Turkey") %>%  
 lm(lifeExp~year, data = .)  
fit\_turkey %>% summary()

##   
## Call:  
## lm(formula = lifeExp ~ year, data = .)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.4373 -0.3457 0.1653 0.9008 1.1033   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -924.58989 37.97715 -24.35 3.12e-10 \*\*\*  
## year 0.49724 0.01918 25.92 1.68e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.147 on 10 degrees of freedom  
## Multiple R-squared: 0.9853, Adjusted R-squared: 0.9839   
## F-statistic: 671.8 on 1 and 10 DF, p-value: 1.681e-10

## WE compare regression coefficients  
fit\_uk$coefficients

## (Intercept) year   
## -294.1965876 0.1859657

fit\_turkey$coefficients

## (Intercept) year   
## -924.5898865 0.4972399

# the y intercepts are negative and far from each other

## To make the intercepts meaningful, we will add in a new column called

## year\_from1952 and re-run fit\_uk and fit\_turkey using year\_from1952 instead of year.

head(gapdata) # dataframe before new column

## # A tibble: 6 x 6  
## country continent year lifeExp pop gdpPercap  
## <fct> <fct> <int> <dbl> <int> <dbl>  
## 1 Afghanistan Asia 1952 28.8 8425333 779.  
## 2 Afghanistan Asia 1957 30.3 9240934 821.  
## 3 Afghanistan Asia 1962 32.0 10267083 853.  
## 4 Afghanistan Asia 1967 34.0 11537966 836.  
## 5 Afghanistan Asia 1972 36.1 13079460 740.  
## 6 Afghanistan Asia 1977 38.4 14880372 786.

gapdata <- gapdata %>%  
 mutate(year\_from1952 = year - 1952)  
head(gapdata) # dataframe after new column inserted

## # A tibble: 6 x 7  
## country continent year lifeExp pop gdpPercap year\_from1952  
## <fct> <fct> <int> <dbl> <int> <dbl> <dbl>  
## 1 Afghanistan Asia 1952 28.8 8425333 779. 0  
## 2 Afghanistan Asia 1957 30.3 9240934 821. 5  
## 3 Afghanistan Asia 1962 32.0 10267083 853. 10  
## 4 Afghanistan Asia 1967 34.0 11537966 836. 15  
## 5 Afghanistan Asia 1972 36.1 13079460 740. 20  
## 6 Afghanistan Asia 1977 38.4 14880372 786. 25

# Now fit the regression line for each country  
fit\_uk <- gapdata %>%  
 filter(country == "United Kingdom") %>%  
 lm(lifeExp ~ year\_from1952, data = .)  
  
fit\_turkey <- gapdata %>%  
 filter(country == "Turkey") %>%  
 lm(lifeExp ~ year\_from1952, data = .)  
  
  
# Compare again the coefficients  
fit\_uk$coefficients

## (Intercept) year\_from1952   
## 68.8085256 0.1859657

fit\_turkey$coefficients

## (Intercept) year\_from1952   
## 46.0223205 0.4972399

# the y intercepts are now positive and closer

#We use the tidy() function from library(broom) to get the variable(s) and specific  
#values in a nice tibble:  
fit\_uk %>% tidy()

## # A tibble: 2 x 5  
## term estimate std.error statistic p.value  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 (Intercept) 68.8 0.240 287. 6.58e-21  
## 2 year\_from1952 0.186 0.00739 25.1 2.26e-10

#And we use the glance() function to get overall model statistics (mostly the r.squared).  
fit\_uk %>% glance()

## # A tibble: 1 x 12  
## r.squared adj.r.squared sigma statistic p.value df logLik AIC BIC  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0.984 0.983 0.442 633. 2.26e-10 1 -6.14 18.3 19.7  
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>

# Comaparison of 3 different regression models for the life expectancy

#Simple linear regression (exactly one predictor variable):  
myfit = lm(lifeExp ~ year, data = gapdata)  
  
#Multivariable linear regression (additive) two predictor variables   
myfit = lm(lifeExp ~ year + country, data = gapdata)  
  
#Multivariable linear regression (interaction):  
myfit = lm(lifeExp ~ year \* country, data = gapdata)  
  
  
# The 3 regression models given above can be combined using one command   
myfit = lm(lifeExp ~ year + country + year:country, data = gapdata)

# We will now create three different linear regression models to further illustrate

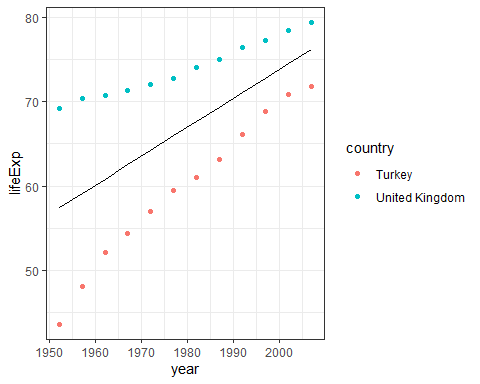
# the difference between a simple model, additive model, and multiplicative model.

# Model 1: year only

# UK and Turkey dataset  
gapdata\_UK\_T <- gapdata %>% filter(country %in% c("Turkey", "United Kingdom"))  
fit\_both1 <- gapdata\_UK\_T %>% lm(lifeExp ~ year\_from1952, data = .)  
fit\_both1

##   
## Call:  
## lm(formula = lifeExp ~ year\_from1952, data = .)  
##   
## Coefficients:  
## (Intercept) year\_from1952   
## 57.4154 0.3416

# Plot the regression line for both countries  
gapdata\_UK\_T %>%  
 mutate(pred\_lifeExp = predict(fit\_both1)) %>%  
 ggplot() +  
 geom\_point(aes(x = year, y = lifeExp, colour = country)) +  
 geom\_line(aes(x = year, y = pred\_lifeExp))



# Model 2: year + country

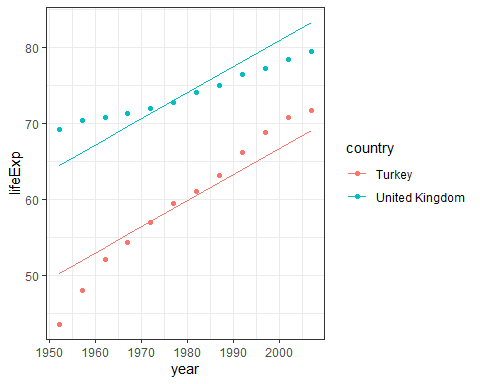
gapdata\_UK\_T %>%  
 mutate(pred\_lifeExp = predict(fit\_both1)) %>%  
 select(country, year, lifeExp, pred\_lifeExp) %>%  
 group\_by(country) %>%  
 slice(1, 6, 12)

## # A tibble: 6 x 4  
## # Groups: country [2]  
## country year lifeExp pred\_lifeExp  
## <fct> <int> <dbl> <dbl>  
## 1 Turkey 1952 43.6 57.4  
## 2 Turkey 1977 59.5 66.0  
## 3 Turkey 2007 71.8 76.2  
## 4 United Kingdom 1952 69.2 57.4  
## 5 United Kingdom 1977 72.8 66.0  
## 6 United Kingdom 2007 79.4 76.2

# Note how the slice() function recognises group\_by() and in this case shows  
# us the 1st, 6th, and 12th observation within each group.  
  
fit\_both2 <- gapdata\_UK\_T %>%  
 lm(lifeExp ~ year\_from1952 + country, data = .)  
fit\_both2

##   
## Call:  
## lm(formula = lifeExp ~ year\_from1952 + country, data = .)  
##   
## Coefficients:  
## (Intercept) year\_from1952 countryUnited Kingdom   
## 50.3023 0.3416 14.2262

gapdata\_UK\_T %>%  
 mutate(pred\_lifeExp = predict(fit\_both2)) %>%  
 ggplot() +  
 geom\_point(aes(x = year, y = lifeExp, colour = country)) +  
 geom\_line(aes(x = year, y = pred\_lifeExp, colour = country))



# Remark

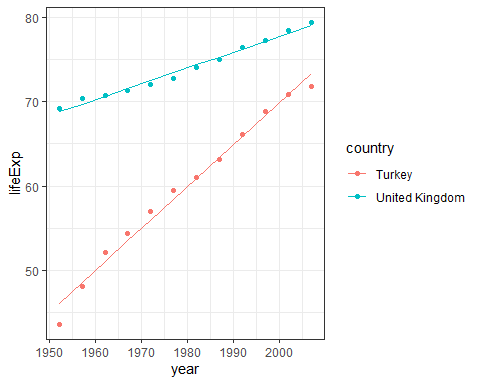
This is better since a regression line was fitted for each country, by including country in the model, we now have fitted lines more closely representing the data. However, the lines are constrained to be parallel.

#Model 3: year \* country

fit\_both3 <- gapdata\_UK\_T %>%  
 lm(lifeExp ~ year\_from1952 \* country, data = .)  
fit\_both3

##   
## Call:  
## lm(formula = lifeExp ~ year\_from1952 \* country, data = .)  
##   
## Coefficients:  
## (Intercept) year\_from1952   
## 46.0223 0.4972   
## countryUnited Kingdom year\_from1952:countryUnited Kingdom   
## 22.7862 -0.3113

gapdata\_UK\_T %>%  
 mutate(pred\_lifeExp = predict(fit\_both3)) %>%  
 ggplot() +  
 geom\_point(aes(x = year, y = lifeExp, colour = country)) +  
 geom\_line(aes(x = year, y = pred\_lifeExp, colour = country))



# This fits the data much better than the previous two models. You can check

# the R-squared using summary(fit\_both3).

# Advanced tip

we can apply a function on multiple objects at once by putting them in a list() and using a map\_() function from the purrr package. library(purrr) gets installed and loaded with library(tidyverse), but it is outside the scope of this book

model\_stats1 <- glance(fit\_both1)  
model\_stats2 <- glance(fit\_both2)  
model\_stats3 <- glance(fit\_both3)  
  
# Combine all model statistics   
bind\_rows(model\_stats1, model\_stats2, model\_stats3)

## # A tibble: 3 x 12  
## r.squared adj.r.squared sigma statistic p.value df logLik AIC BIC  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0.373 0.344 7.98 13.1 1.53e- 3 1 -82.9 172. 175.   
## 2 0.916 0.908 2.99 114. 5.18e-12 2 -58.8 126. 130.   
## 3 0.993 0.992 0.869 980. 7.30e-22 3 -28.5 67.0 72.9  
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>

# Interpretation

The r.squared statistics is higher in the third model The Akaike Information Criterion (AIC) is smallest among the three models The p-value = 7.30e-22 < 0.05 which tells us that the regression model is significant

# another way to display summary results:  
 list(fit\_both1, fit\_both2, fit\_both3) %>%  
 map\_df(glance)

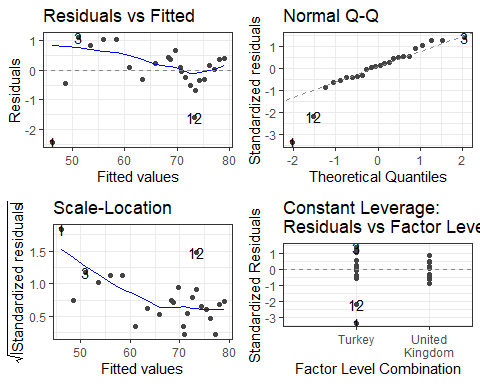
## # A tibble: 3 x 12  
## r.squared adj.r.squared sigma statistic p.value df logLik AIC BIC  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0.373 0.344 7.98 13.1 1.53e- 3 1 -82.9 172. 175.   
## 2 0.916 0.908 2.99 114. 5.18e-12 2 -58.8 126. 130.   
## 3 0.993 0.992 0.869 980. 7.30e-22 3 -28.5 67.0 72.9  
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>

# Check assumptions

The assumptions of linear regression can be checked with diagnostic plots, either by passing the fitted object (lm() output) to base R plot(), or by using the more convenient function below.

library(ggfortify)  
autoplot(fit\_both3)

## Warning: `arrange\_()` was deprecated in dplyr 0.7.0.  
## Please use `arrange()` instead.  
## See vignette('programming') for more help



#FIGURE 7.13: Diagnostic plots. Life expectancy in Turkey and the UK - # multivariable multiplicative model

# Fitting more complex models

# 7.3.1 The Question (3)

# Finally in this section, we are going to fit a more complex linear regression

# model. Here, we will discuss variable selection and introduce the Akaike Information Criterion (AIC).

# 7.3.2 Model fitting principles (This will be your guide on how to fit good model)

We suggest building statistical models on the basis of the following six pragmatic principles: 1. As few explanatory variables should be used as possible (parsimony); 2. Explanatory variables associated with the outcome variable in previous studies should be accounted for; 3. Demographic variables should be included in model exploration; 4. Population stratification should be incorporated if available; 5. Interactions should be checked and included if influential; 6. Final model selection should be performed using a “criterion-based #approach” 7. minimise the Akaike information criterion (AIC) 8. maximise the adjusted R-squared value.

# 7.3.3 AIC

The Akaike Information Criterion (AIC) is an alternative goodness-of-fit measure. In that sense, it is similar to the R-squared, but it has a different statistical basis. It is useful because it can be used to help guide the best model

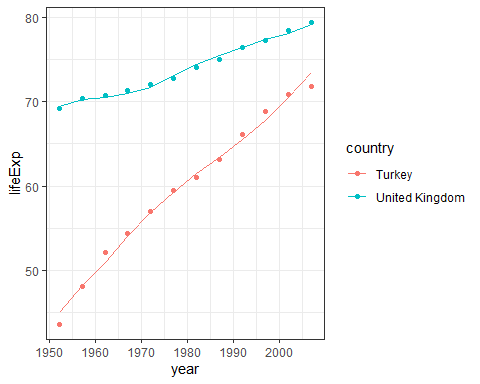
Question: What of we considered other variables?

#Model 4: year \* country and population and gdpPercap

fit\_both4 <- gapdata\_UK\_T %>%  
 lm(lifeExp ~ year\_from1952 \* country + pop + gdpPercap, data = .)  
fit\_both4

##   
## Call:  
## lm(formula = lifeExp ~ year\_from1952 \* country + pop + gdpPercap,   
## data = .)  
##   
## Coefficients:  
## (Intercept) year\_from1952   
## 5.844e+01 1.055e+00   
## countryUnited Kingdom pop   
## 4.136e+01 -6.066e-07   
## gdpPercap year\_from1952:countryUnited Kingdom   
## 2.012e-05 -7.729e-01

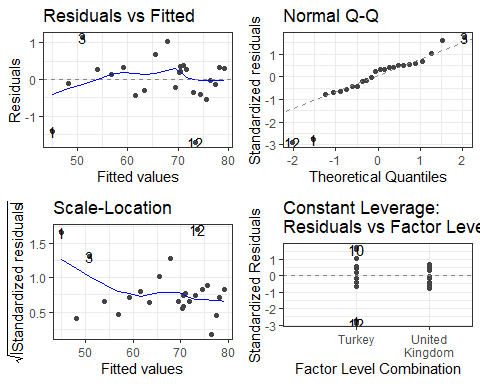
gapdata\_UK\_T %>%  
 mutate(pred\_lifeExp = predict(fit\_both4)) %>%  
 ggplot() +  
 geom\_point(aes(x = year, y = lifeExp, colour = country)) +  
 geom\_line(aes(x = year, y = pred\_lifeExp, colour = country))



# Display summary results:  
list(fit\_both1, fit\_both2, fit\_both3,fit\_both4 ) %>%  
 map\_df(glance)

## # A tibble: 4 x 12  
## r.squared adj.r.squared sigma statistic p.value df logLik AIC BIC  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0.373 0.344 7.98 13.1 1.53e- 3 1 -82.9 172. 175.   
## 2 0.916 0.908 2.99 114. 5.18e-12 2 -58.8 126. 130.   
## 3 0.993 0.992 0.869 980. 7.30e-22 3 -28.5 67.0 72.9  
## 4 0.996 0.995 0.723 852. 1.02e-20 5 -22.8 59.7 67.9  
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>

autoplot(fit\_both4)

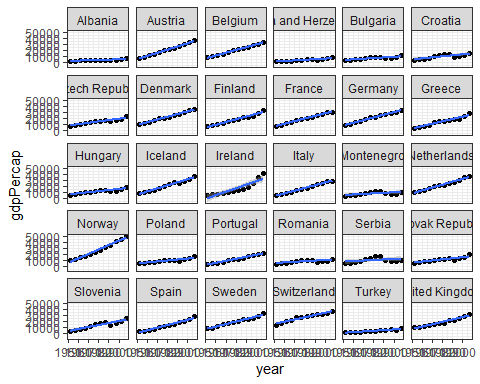


Note: Regression Analysis as applied here consists of Linear models and Non-Linear models

#Plot the GDP per capita by year for countries in Europe.

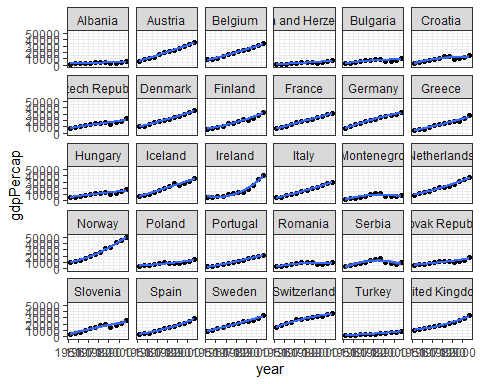
gapdata %>%  
 filter(continent == "Europe") %>%  
 ggplot(aes(x = year, y = gdpPercap)) +  
 geom\_point() +  
 geom\_smooth(method = "lm") +  
 facet\_wrap(country ~ .)

## `geom\_smooth()` using formula 'y ~ x'



What countries have non-linear models? Countries not linear: Ireland, Montenegro, Serbia.

# Add quadratic term  
gapdata %>%  
 filter(continent == "Europe") %>%  
 ggplot(aes(x = year, y = gdpPercap)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", formula = "y ~ poly(x, 2)") +  
 facet\_wrap(country ~ .)



# Try to model Albania and Austria (the first two countries)  
gapdata %>%  
 filter(country %in% c("Albania", "Austria")) %>%  
 ggplot() +  
 geom\_point(aes(x = year, y = gdpPercap, colour= country))

