**Econometrics Exam (05.02.2024)**

*Name: Nicolas Röver*

*Student-ID: 7354299*

**Theoretical Questions**

1. Correct. If we have the same significance level (e.g. 5%), the one sided test will have a critical value of about 1.64 (95% quantile) and the two sided test will have a value of about 1.96 (97.5% quantile).

2. False. Multicollinearity is a problem of the data, not the estimator. OLS is still unbiased. However, the estimator has a large variance. Dropping a variable should not be done! Dropiing a variable which is part of the true model leads to omitted variable bias!

3. Correct.

4. Wrong. The long run effect is given by beta/0.4. Thus the adjustment is given by beta\* 2,5. This is not 60%.

5. Correct, this is called the exogeneity requirement.

6. False. Adding an instrument with high relevance (i.e. high correlation with the outcome variable) can decrease the variance of the estimator.

7. False. The LASSO estimator is a shrinkage estimator which minimizes the mean square error subject to a size constraint on the parameters. While related, this is not tied one -to-one to the significance level of the parameter.

**CASE 2**

**Code:**

**################################################################################**

**##Code for Question "Unemployment" in the Exam on January 5th, 2024 in Advanced Econometrics###**

**################################################################################**

**################################################################################**

**##Set Root Directory**

**library(here)**

**library(fs)**

**MAIN\_DIRECTORY\_PATH <- here()**

**################################################################################**

**##Install Packages and Source Code**

**library(rio)**

**library(tidyverse)**

**library(lmtest)**

**library(car)**

**library(tseries)**

**library(dynlm)**

**library(pscl)**

**source(path(MAIN\_DIRECTORY\_PATH, "functions\_exam\_2024.R"))**

**################################################################################**

**##Load and Clean Data**

**Data\_Path = list.files(path = path(MAIN\_DIRECTORY\_PATH, "Data", "Question\_2"),**

**pattern = ".csv", full.names = TRUE)**

**df <- import(Data\_Path)**

**check\_if\_columns\_are\_numeric\_or\_integer(df)**

**check\_for\_nas\_in\_df(df)**

**constant <- 5/(1+5)**

**df = df %>%**

**mutate(age = age + constant) %>%**

**na.omit()**

**################################################################################**

**##Task a)**

**lpm <- lm(unemployed ~ age + age^2+ educ + health +**

**female + migback, data = df)**

**#I assume heteroskedasticity for the error dependent on the value of unemployed.**

**#The reason is that unemployed follows a bernoulli distribution, which means the**

**#variance of the error depends on the values of the dependent variables.**

**coeftest(lpm)**

**################################################################################**

**##Task b)**

**estimations <- lpm$fitted.values**

**hist(estimations, breaks = 50, main = "Histogram of Estimations", xlab = "Estimations")**

**#In the plot, we observe negativ values for the probability to be unemployed. The reason**

**#is that the linear probability model does not restrict the values of the dependent variable**

**#to be between 0 and 1. This is a problem, because the dependent variable is a probability**

**#which can only be between 0 and 1 (see kolmogorovs definition of probabilities.)**

**#A solution is to use a probit or logit model, which restricts the range of values**

**#through the choice of the link function.**

**################################################################################**

**##Task c)**

**mean(df$unemployed)**

**#Unconditionally, the probability is 5%.**

**coefficient\_age = lpm$coefficients['age']**

**coefficient\_age \* 5**

**# the probability decreases by -0.008623624.**

**################################################################################**

**##Task d)**

**probit\_model <- glm(unemployed ~ age + age^2+ educ + health +**

**female + migback, data = df, family = binomial(link = "probit"))**

**summary(probit\_model)**

**fit\_measures <- pR2(probit\_model, type = "McFadden")**

**fit\_measures["McFadden"]**

**df["probit\_estimations"] <- predict(probit\_model, type = "response")**

**extra\_regression <- lm(unemployed ~ probit\_estimations, data = df)**

**summary(extra\_regression)**

**#The pseudo R^2 is higher for the probit model than when using the approach of**

**#first estimating the predicted values and then using a normal R^2 from a linear**

**#regression. This highlights that pseudo-R^2 and R^2 are different measures to**

**#assess model fit. Both measures focus on different aspects of the model fit and**

**#thus do not give equal results. However, results for both measures should**

**#always be in the same ballpark, as these measures try to assess the same thing**

**#and are not that different.**

**Output:**

**################################################################################**

**> ##Task a)**

**>**

**> lpm <- lm(unemployed ~ age + age^2+ educ + health +**

**+ female + migback, data = df)**

**>**

**>**

**> #I assume heteroskedasticity for the error dependent on the value of unemployed.**

**> #The reason is that unemployed follows a bernoulli distribution, which means the**

**> #variance of the error depends on the values of the dependent variables.**

**>**

**> coeftest(lpm)**

**t test of coefficients:**

**Estimate Std. Error t value Pr(>|t|)**

**(Intercept) 0.32108376 0.01365356 23.5165 < 2.2e-16 \*\*\***

**age -0.00172472 0.00012329 -13.9890 < 2.2e-16 \*\*\***

**educ -0.00940957 0.00071835 -13.0989 < 2.2e-16 \*\*\***

**health -0.01065565 0.00095066 -11.2087 < 2.2e-16 \*\*\***

**female 0.00048538 0.00394107 0.1232 0.902**

**migback 0.01940547 0.00451371 4.2992 1.728e-05 \*\*\***

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**>**

**> ################################################################################**

**> ##Task b)**

**>**

**> estimations <- lpm$fitted.values**

**>**

**> hist(estimations, breaks = 50, main = "Histogram of Estimations", xlab = "Estimations")**

**>**

**> #In the plot, we observe negativ values for the probability to be unemployed. The reason**

**> #is that the linear probability model does not restrict the values of the dependent variable**

**> #to be between 0 and 1. This is a problem, because the dependent variable is a probability**

**> #which can only be between 0 and 1 (see kolmogorovs definition of probabilities.)**

**> #A solution is to use a probit or logit model, which restricts the range of values**

**> #through the choice of the link function.**

**>**

**> ################################################################################**

**> ##Task c)**

**>**

**> mean(df$unemployed)**

**[1] 0.05058528**

**>**

**> #Unconditionally, the probability is 5%.**

**>**

**> coefficient\_age = lpm$coefficients['age']**

**>**

**> coefficient\_age \* 5**

**age**

**-0.008623624**

**>**

**> # the probability decreases by -0.008623624.**

**>**

**>**

**> ################################################################################**

**> ##Task d)**

**>**

**> probit\_model <- glm(unemployed ~ age + age^2+ educ + health +**

**+ female + migback, data = df, family = binomial(link = "probit"))**

**>**

**> summary(probit\_model)**

**Call:**

**glm(formula = unemployed ~ age + age^2 + educ + health + female +**

**migback, family = binomial(link = "probit"), data = df)**

**Deviance Residuals:**

**Min 1Q Median 3Q Max**

**-1.0146 -0.3593 -0.2513 -0.1575 3.4912**

**Coefficients:**

**Estimate Std. Error z value Pr(>|z|)**

**(Intercept) 1.183774 0.151198 7.829 4.91e-15 \*\*\***

**age -0.018432 0.001369 -13.463 < 2e-16 \*\*\***

**educ -0.118621 0.009684 -12.249 < 2e-16 \*\*\***

**health -0.099082 0.009301 -10.652 < 2e-16 \*\*\***

**female 0.026366 0.041633 0.633 0.527**

**migback 0.177272 0.043828 4.045 5.24e-05 \*\*\***

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**(Dispersion parameter for binomial family taken to be 1)**

**Null deviance: 4789.6 on 11959 degrees of freedom**

**Residual deviance: 4254.9 on 11954 degrees of freedom**

**AIC: 4266.9**

**Number of Fisher Scoring iterations: 6**

**>**

**> fit\_measures <- pR2(probit\_model, type = "McFadden")**

**fitting null model for pseudo-r2**

**>**

**> fit\_measures["McFadden"]**

**McFadden**

**0.1116336**

**>**

**> df["probit\_estimations"] <- predict(probit\_model, type = "response")**

**>**

**> extra\_regression <- lm(unemployed ~ probit\_estimations, data = df)**

**>**

**> summary(extra\_regression)**

**Call:**

**lm(formula = unemployed ~ probit\_estimations, data = df)**

**Residuals:**

**Min 1Q Median 3Q Max**

**-0.41922 -0.06310 -0.03015 -0.01049 1.00006**

**Coefficients:**

**Estimate Std. Error t value Pr(>|t|)**

**(Intercept) -0.002423 0.002758 -0.878 0.38**

**probit\_estimations 1.047997 0.038659 27.108 <2e-16 \*\*\***

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**Residual standard error: 0.2127 on 11958 degrees of freedom**

**Multiple R-squared: 0.0579, Adjusted R-squared: 0.05782**

**F-statistic: 734.9 on 1 and 11958 DF, p-value: < 2.2e-16**

**>**

**> #The pseudo R^2 is higher for the probit model than when using the approach of**

**> #first estimating the predicted values and then using a normal R^2 from a linear**

**> #regression. This highlights that pseudo-R^2 and R^2 are different measures to**

**> #assess model fit. Both measures focus on different aspects of the model fit and**

**> #thus do not give equal results. However, results for both measures should**

**> #always be in the same ballpark, as these measures try to assess the same thing**

**> #and are not that different.**

**Histogram:**

**A graph of a graph of a person

Description automatically generated with medium confidence**

**CASE 1**

**################################################################################**

**> ##Code for Question "Milk" in the Exam on January 5th, 2024 in Advanced Econometrics###**

**> ################################################################################**

**>**

**> ################################################################################**

**> ##Set Root Directory**

**>**

**> library(here)**

**> library(fs)**

**>**

**> MAIN\_DIRECTORY\_PATH <- here()**

**>**

**> ################################################################################**

**> ##Install Packages and Source Code**

**>**

**> library(rio)**

**> library(tidyverse)**

**> library(lmtest)**

**> library(car)**

**> library(tseries)**

**> library(dynlm)**

**> source(path(MAIN\_DIRECTORY\_PATH, "functions\_exam\_2024.R"))**

**>**

**> ################################################################################**

**> ##Load and Clean Data**

**>**

**> Data\_Path = list.files(path = path(MAIN\_DIRECTORY\_PATH, "Data", "Question\_1"),**

**+ pattern = ".csv", full.names = TRUE)**

**>**

**> df <- import(Data\_Path)**

**>**

**> check\_if\_columns\_are\_numeric\_or\_integer(df)**

**>**

**> check\_for\_nas\_in\_df(df)**

**No NA values found in the data frame.**

**>**

**> df <- df %>%**

**+ mutate(log\_Sales = log(SALE \* (7354299/7123456)))**

**>**

**>**

**> ################################################################################**

**> ##Task a)**

**>**

**> demand\_model <- lm(log\_Sales ~ log(PRICE) + log(PRICE\_FM) + log(PROMOTION), data = df)**

**>**

**> car::durbinWatsonTest(demand\_model, max.lag= 4)**

**lag Autocorrelation D-W Statistic p-value**

**1 0.5167077 0.9517446 0**

**2 0.5027356 0.9772383 0**

**3 0.5283984 0.9207446 0**

**4 0.4596939 1.0420174 0**

**Alternative hypothesis: rho[lag] != 0**

**>**

**> #The errors are highly autocorrelated! Every lag up until the fourth has a p-value**

**> #very close to 0.**

**>**

**> coeftest(demand\_model)**

**t test of coefficients:**

**Estimate Std. Error t value Pr(>|t|)**

**(Intercept) 5.535578 0.080603 68.6769 < 2e-16 \*\*\***

**log(PRICE) -3.281473 0.174393 -18.8166 < 2e-16 \*\*\***

**log(PRICE\_FM) 3.452079 0.241428 14.2986 < 2e-16 \*\*\***

**log(PROMOTION) 0.020251 0.010048 2.0155 0.04495 \***

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**>**

**> > ################################################################################**

**> ##Task b)**

**>**

**> approximate\_effect <- demand\_model$coefficients["log(PRICE)"]**

**>**

**> approximate\_effect**

**log(PRICE)**

**-3.281473**

**>**

**> #Approximately, sales decrease by -3.281473%**

**>**

**> exact\_effect = exp(demand\_model$coefficients["log(PRICE)"]\*log(1.01))**

**>**

**> 1- exact\_effect**

**log(PRICE)**

**0.03212443**

**>**

**> #Exact effect, is given by -0.03212443**

**> ################################################################################**

**> ##Task c)**

**>**

**> linearHypothesis(demand\_model, c("log(PRICE) + log(PRICE\_FM) = 0"))**

**Linear hypothesis test**

**Hypothesis:**

**log(PRICE) + log(PRICE\_FM) = 0**

**Model 1: restricted model**

**Model 2: log\_Sales ~ log(PRICE) + log(PRICE\_FM) + log(PROMOTION)**

**Res.Df RSS Df Sum of Sq F Pr(>F)**

**1 244 6.5072**

**2 243 6.4825 1 0.024702 0.926 0.3369**

**>**

**> #The p-value is 0.3369, so I cannot reject the null-hypothesis on a reasonable**

**> #significance level.**

**>**

**> #I would reformulate the following way: Only include the variable log(PRICE) - log(PRICE\_FM);**

**> #so I would not include log(PRICE) and log(PRICE\_FM) separately. This way, the model**

**> #represents our hypothesis that the coeffecients sum up to zero (this can be derived**

**> #using simple algebra, just plugging in -beta\_{log\_Price} for beta\_{log\_Price\_FM} in**

**> #the model specifcitation.**

**>**

**> ################################################################################**

**> ##Task d)**

**>**

**> df <- df %>%**

**+ mutate(log\_price\_squared = log(PRICE)^2,**

**+ log\_price = log(PRICE))**

**>**

**> quadratic\_model <-lm(log\_Sales ~ log\_price + log\_price\_squared, data = df)**

**>**

**> summary(quadratic\_model)**

**Call:**

**lm(formula = log\_Sales ~ log\_price + log\_price\_squared, data = df)**

**Residuals:**

**Min 1Q Median 3Q Max**

**-0.53448 -0.13218 -0.02574 0.08488 0.89267**

**Coefficients:**

**Estimate Std. Error t value Pr(>|t|)**

**(Intercept) 4.56615 0.01954 233.67 < 2e-16 \*\*\***

**log\_price -1.52557 0.17968 -8.49 2.03e-15 \*\*\***

**log\_price\_squared 8.79063 1.43862 6.11 3.90e-09 \*\*\***

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**Residual standard error: 0.2215 on 244 degrees of freedom**

**Multiple R-squared: 0.5232, Adjusted R-squared: 0.5193**

**F-statistic: 133.9 on 2 and 244 DF, p-value: < 2.2e-16**

**>**

**> minimum <- quadratic\_model$coefficients["log\_price"]/(2\*quadratic\_model$coefficients["log\_price\_squared"])**

**>**

**> minimum\_real\_number <- exp(minimum)**

**>**

**> minimum\_real\_number**

**log\_price**

**0.9168854**

**>**

**> #Note that this is a minimum, because the coefficient of the squared term is positive**

**>**

**> linearHypothesis(quadratic\_model , c("log\_price - log\_price\_squared = 0"))**

**Linear hypothesis test**

**Hypothesis:**

**log\_price - log\_price\_squared = 0**

**Model 1: restricted model**

**Model 2: log\_Sales ~ log\_price + log\_price\_squared**

**Res.Df RSS Df Sum of Sq F Pr(>F)**

**1 245 14.888**

**2 244 11.974 1 2.9143 59.385 3.26e-13 \*\*\***

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1**

**>**

**> #I can reject that the minimum sales are at 1 euro at any reasonable significance level.**

**File with functions (I did not use most of them!)**

**#Packages**

**library(roxygen2)**

**plot\_error\_distribution\_for\_variable <- function(original\_data, regression\_model,**

**heteroskedastic\_variable){**

**#'This function generates a graph with Boxplots for the empirical error distribution**

**#'given a value of a variable where it is supposed that the error is heteroskedastic**

**#'with respect to that variable, i.e. clustering could be a good idea.**

**#'**

**#'@args original\_data: A dataframe with the original data, all columns are numeric/int**

**#'@args regression\_model: A regression model object.**

**#'@args heteroskedastic\_variable: A string with the name of the variable where**

**#' heteroskedasticity is supposed to be present.**

**#'**

**#'@return A graph with Boxplots for the empirical error distribution**

**#'**

**check\_if\_columns\_are\_numeric\_or\_integer(original\_data)**

**check\_for\_nas\_in\_df(original\_data)**

**original\_data$heteroskedastic\_variable <- factor(original\_data[,heteroskedastic\_variable])**

**original\_data$residuals <- regression\_model$residuals**

**plot <- ggplot(original\_data, aes(x = heteroskedastic\_variable, y = residuals)) +**

**geom\_boxplot() +**

**labs(x = heteroskedastic\_variable, y = "Residuals") +**

**theme\_bw()**

**return(plot)**

**}**

**check\_if\_columns\_are\_numeric\_or\_integer <- function(my\_data) {**

**# Check if all columns are numeric or integer**

**if (!all(sapply(my\_data, function(x) is.numeric(x) || is.integer(x)))) {**

**stop("All columns must be numeric or integer.")**

**}**

**}**

**check\_for\_nas\_in\_df <- function(data) {**

**if (any(is.na(data))) {**

**message("Careful: The data frame contains NA values.")**

**} else {**

**message("No NA values found in the data frame.")**

**}**

**}**

**library(roxygen2)**

**compute\_probit\_confusion\_matrix <- function(model, data, outcome\_variable) {**

**#' This function takes in a probit model and corresponding data and returns a confusion matrix**

**#' @param model A probit model estimated with glm**

**#' @param data A data frame containing the data used to estimate the model**

**#' @param outcome\_variable str The name of the outcome variable as a string**

**#'**

**#' @return A confusion matrix for the model and the data**

**check\_for\_nas\_in\_df(data)**

**# Make Predictions**

**predicted\_probs <- predict(model, type = "response")**

**# Convert Probabilities to Binary Predictions**

**predicted\_classes <- ifelse(predicted\_probs > 0.5, 1, 0)**

**# Create Confusion Matrix**

**confusion\_matrix <- table(observed = data[, outcome\_variable], predicted = predicted\_classes)**

**return(confusion\_matrix)**

**}**

**#Packages**

**library(roxygen2)**

**run\_standardized\_regression<- function(data, regression\_formula){**

**#' This function takes a dataframe and the regression formula and returns**

**#' a standardized regression model**

**#'**

**#' @param data this is the data on which the regression is performed**

**#' @param regression\_formula this is the regression formula formulated as a string,**

**#' i.e. y ~ x1 + x2**

**#'**

**#' @return a regression model object**

**regression\_formula <- as.formula(regression\_formula)**

**standardized\_data <- as.data.frame(scale(data))**

**regression\_model <- lm(regression\_formula, data = standardized\_data)**

**return(regression\_model)**

**}**

**create\_lag\_of\_variable <- function(column, lag){**

**#' This function takes a column and creates a lag of that column**

**#'**

**#' @param column this is the column for which we want to create a lag**

**#' @param lag this is the number of lags we want to create -1 is creating one**

**#' lag back, -2 is creating 2 lags back etc.)**

**#'**

**#' @return a vector with the lagged values**

**if (lag > -1) {**

**stop("Invalid value for lag. Lag should be -1 or less.")**

**}**

**lagged\_column <- c(rep(NA, abs(lag)), column[1:(length(column) - abs(lag))])**

**return(lagged\_column)**

**}**

**check\_if\_columns\_are\_numeric\_or\_integer <- function(my\_data) {**

**# Check if all columns are numeric or integer**

**if (!all(sapply(my\_data, function(x) is.numeric(x) || is.integer(x)))) {**

**stop("All columns must be numeric or integer.")**

**}**

**}**

**check\_for\_nas\_in\_df <- function(data) {**

**#Check if there are NA values in the data**

**if (any(is.na(data))) {**

**message("Careful: The data frame contains NA values.")**

**} else {**

**message("No NA values found in the data frame.")**

**}**

**}**