library(wbstats)

library(dplyr)

library(data.table)

library(rio)

library(stringr)

library(magrittr)

library(purrr)

library(ggplot2)

library(latex2exp)

library(stargazer)

library(glmnet)

library(corrplot)

library(dplyr)

library(lubridate)

library(rio)

library(stringr)

library(tibble)

library(magrittr)

library(purrr)

library(data.table)

library(tidyr)

library(forecast)

library(gridExtra)

library(grid)

library(Amelia)

library(fastDummies)

library(gnm)

library(parallel)

library(xgboost)

library(rbenchmark)

#library(plotly)

library(e1071)

library(rlang)

library(caret)

library(DiagrammeR)

library(sophisthse)

library(DiagrammeRsvg)

library(rsvg)

library(neuralnet)

library(ggridges)

library(tseries)

library(xts)

library(DtD)

library(readr)

library(TTR)

library(zoo)

library(seqHMM)

library(hexbin)

library(sophisthse)

library(tseries)

library(caret)

library(sophisthse)

library(tseries)

library(MLmetrics)

library(caret)

library(xts)

library(randomForest)

library(spikeslab)

library(BigVAR)

library(xtable)

library(readxl)

library(httr)

library(ggpubr)

library(shiny)

library(scales)

library(DT)

library(lemon)

library(shinyWidgets)

library(dplyr)

library(multDM)

library(tables)

library(xtable)

library(animation)

library(gganimate)

library(multDM)

# import rawdata

load('raw.Rdata')

# # доходность ртс, процентные ставки на межбанковском рынке, спреды по облигациям,

# темпы роста номинального эффективного курса, реального эффективного курса, прирост цен на нефть,

# 4 разность логарифма ВВП и ИПЦ, отношение номинальных инвестиций к номинальному ввп + лаги

# extended series ----

eseries <- c('investment', 'mkr\_1d','mkr\_7d','gov\_6m','GKO',

'reer', 'neer', 'oil', 'rts',

'GDPEA\_C\_Q', 'INVFC\_Q', # служебные для invest2gdp

'CPI\_Q\_CHI',# ипц

'GDPEA\_Q\_DIRI', # ввп

'EMPLDEC\_Q', # заявленная потребность в работника

'UNEMPL\_Q\_SH',# безработица

'CONSTR\_Q\_NAT', # индекс строительно-монтажных работ,

'WAG\_Q', # зарплата

'CONI\_Q\_CHI', # индекс цен на строительно-монтажные работы

'CTI\_Q\_CHI', # индекс тарифов на грузовые перевозки

'AGR\_Q\_DIRI', # индекс сельхоз производства

#'CNSTR\_Q\_DIRI',# индекс работ в строительстве

'RTRD\_Q\_DIRI', # оборот розничной торговли

'HHI\_Q\_DIRI',# индекс реальных денежных доходов населения

'M0\_Q', # M0

'M2\_Q',# М2

'CBREV\_Q',# доходы конс. бюджета

'CBEX\_Q',# расходы конс. бюджета

'FBREV\_Q',# доходы фед. бюджета

'FBEX\_Q',# расходы фед. бюджета

'RDEXRO\_Q',# официальный курс доллара

'RDEXRM\_Q',# курс доллара на ммвб

'LIAB\_T\_Q',# кредиторская задолженность в среднем за период

'LIAB\_UNP\_Q',# просроченная кредиторская задолженность в среднем за период

'LIAB\_S\_Q',# кредиторская задолженность поставщикам в среднем за период

'LIAB\_B\_Q',# кредиторская задолженность в бюджет в среднем за период

'DBT\_T\_Q',#дебиторская задолженность в среднем за период

'DBT\_UNP\_Q',#просроченная дебиторская задолженность в среднем за период

# 'DBT\_P\_Q',# дебиторская задолженность покупателей в среднем за период

'EX\_T\_Q',# экспорт

'IM\_T\_Q',# импорт

'PPI\_EA\_Q' # (после 2004-01)

)

df <- rawdata[,eseries]

# отношение номинальных инвестиций к номинальному ВВП

df$invest2gdp <- df$INVFC\_Q/df$GDPEA\_C\_Q

# удалить лишнее

df$GDPEA\_C\_Q <- df$INVFC\_Q <- NULL

# вставить gko в 6m gov bond yield

df$gov\_6m[which(is.na(df$gov\_6m))] <-

df$GKO[which(is.na(df$gov\_6m))]

df$GKO <- NULL

# нестационарные ряды без сезонности (разность) ----

for(i in c('reer','neer','oil','rts')){

df[,i] %<>% diff.xts(log=TRUE)

}

# нестационарные ряды с сезонностью (4-ая разность) ----

for(i in c('investment', 'CPI\_Q\_CHI',

'invest2gdp',

# 'deflator', только с 1996

'GDPEA\_Q\_DIRI',

'EMPLDEC\_Q',

'UNEMPL\_Q\_SH',

'CONSTR\_Q\_NAT',

###### 'TRP\_Q\_PASS\_DIRI',

'WAG\_Q',

'CONI\_Q\_CHI',

'CTI\_Q\_CHI',

'AGR\_Q\_DIRI',

'RTRD\_Q\_DIRI',

'HHI\_Q\_DIRI',

'M0\_Q',

'M2\_Q',

#### 'IR\_Q',

#### 'ICR\_Q',

'CBREV\_Q',

'CBEX\_Q',

'FBREV\_Q',

'FBEX\_Q',

'RDEXRO\_Q',# официальный курс доллара

'RDEXRM\_Q',# курс доллара на ммвб

'LIAB\_T\_Q',# кредиторская задолженность в среднем за период

'LIAB\_UNP\_Q',# просроченная кредиторская задолженность в среднем за период

'LIAB\_S\_Q',# кредиторская задолженность поставщикам в среднем за период

'LIAB\_B\_Q',# кредиторская задолженность в бюджет в среднем за период

'DBT\_T\_Q',#дебиторская задолженность в среднем за период

'DBT\_UNP\_Q',#просроченная дебиторская задолженность в среднем за период

########## 'DBT\_P\_Q',# дебиторская задолженность покупателей в среднем за период

'EX\_T\_Q',# экспорт

'IM\_T\_Q',# импорт

'PPI\_EA\_Q' # (после 2004-01)

)){

df[,i] %<>% diff.xts(lag = 4, log=TRUE)

}

do.call.pipe <- function(args,what){

do.call(what,args)

}

create\_lag <- function(df, lag){

out <- df

if(lag > 0){

for(i in 1:lag){

x <- lag.xts(df, k = i)

colnames(x) <- paste0(colnames(x), "\_lag", i)

out %<>% merge.zoo(x)

}

}

return(out %>% as.xts)

}

train.model <- function(startdt= as.Date('1996-01-01'),

enddt = as.Date('2016-10-01'),

model,

# series parameter

# only for regularisation and machine learning models

series='e',

lag = 0L,

h = 1L,

target = c('investment','RTRD\_Q\_DIRI','GDPEA\_Q\_DIRI','UNEMPL\_Q\_SH', 'CPI\_Q\_CHI'),

# parameters for durable evaluations with function arguments from expand.grid table

i = NULL,

N = NULL

){

message(paste0(i, '/', N))

# import df

# seed

set.seed(2019)

if(series == ''){

load('~/investment\_forecasting/data/stationary\_data.RData')

} else if(series == 'e') {

load('~/investment\_forecasting/data/stationary\_data\_ext.RData')

} else if(series == 'oil'){

load('~/investment\_forecasting/data/stationary\_data\_oil.RData')

}

target <- match.arg(target)

if(!model %in% c('arima', 'rw')){

if(!series %in% c('', 'e', 'oil')){

df %<>% df[, c(target,

series)]

}

# create lag

if(lag != 0 |! model %in% c('arima', 'rw')){

df %<>% create\_lag(lag)

}

if(!is.integer(h)){

message('h must be integer')

return(NULL)

}

df$y <- lag.xts(df[,target], k = -h)

if(h == 0){

if(target == 'investment'){

df$investment <-df$invest2gdp <- df$GDPEA\_Q\_DIRI <- NULL

} else{

df %<>% .[, colnames(.) != target]

}

} else{

df$gdplag <- df$investmentlag <- df$invest2gdplag <- NULL

}

df %<>% .[rowSums(is.na(.[,colnames(.)!='y']))==0,]

# проверка на start и end

if(startdt >= enddt){

message('start must be greater then end')

return(NULL)

}

startdt <- max(startdt %>% as.yearqtr,

first(time(df)) %>% as.yearqtr) %>%

as.Date

enddt <- min(enddt %>% as.yearqtr,

last(time(df)) %>% as.yearqtr) %>%

as.Date

if(df[paste0(startdt, "/", enddt)] %>% nrow < 48){

message('train set length must be greater then 48 quarters')

return(NULL)

}

if(which((time(df) %>% as.Date())==(enddt%>% as.yearqtr%>% as.Date))+1>length(df$y %>% na.omit)){

#message('enddt must not be greater then last date when investment data is avaliable minus h quarters')

return(NULL)

}

train\_n <- which(time(df) %>% as.Date()==(startdt %>% as.yearqtr %>% as.Date)):

which((time(df) %>% as.Date())==

(enddt %>% as.yearqtr%>% as.Date))

test\_n <- (which((time(df) %>% as.Date())==(enddt%>% as.yearqtr%>% as.Date))+1):nrow(df)

df$y %<>% na.fill(0)

X.matrix <- model.matrix(y~0+., data = df)

X.train <- X.matrix[train\_n,]

X.test <- X.matrix[test\_n,]

y.train <- df$y[train\_n] %>% as.numeric

y.test <- df$y[test\_n] %>% as.numeric

# tc <- trainControl(method = 'cv', number = 10)

tc <- trainControl(method = "timeslice",

initialWindow = 40,

horizon = 1,

skip=0,

fixedWindow = TRUE)

lambda <- seq(-8,-1,length = 200) %>% exp()

if(model == 'lasso'){

train.out <- train(x=X.train,

y=y.train,

method = "glmnet",

metric = "RMSE",

trControl = tc,

tuneGrid =

expand.grid(.alpha = c(1),

.lambda = lambda))

model\_fit <- glmnet(X.train,

y.train,

alpha = 1,

lambda = train.out$bestTune[1,2])

pred <- predict(model\_fit,

newx = rbind(X.train, X.test)) %>% as.numeric

}

else if(model == 'elnet'){

train.out <- train(x=X.train,

y=y.train,

method = "glmnet",

metric = "RMSE",

trControl = tc,

tuneGrid =

expand.grid(.alpha = c(0.5),

# new

.lambda = lambda))

model\_fit <- glmnet(X.train,

y.train,

alpha = 0.5,

lambda = train.out$bestTune[1,2])

pred <- predict(model\_fit,

newx = rbind(X.train, X.test)) %>% as.numeric

}

else if(model == 'ridge'){

train.out <- train(x=X.train,

y=y.train,

method = "glmnet",

metric = "RMSE",trControl = tc,

tuneGrid = expand.grid(.alpha = 0,

.lambda = lambda))

model\_fit <- glmnet(X.train,

y.train,

alpha = 0,

lambda = train.out$bestTune[1,2])

pred <- predict(model\_fit,newx = rbind(X.train, X.test)) %>% as.numeric

}

else if (model == 'adalasso'){

train.ridge <- train(x=X.train,

y=y.train,

method = "glmnet",

metric = "RMSE",trControl = tc,

tuneGrid = expand.grid(.alpha = 0,.lambda = lambda))

m\_ridge <- glmnet(X.train,

y.train,

alpha = 0,

lambda = train.ridge$bestTune[1,2])

w3 <- 1/abs(as.numeric(coef(m\_ridge))

[1:(ncol(X.train))] )^0.5 ## Using gamma = 1

w3[which(w3 == Inf)] <- 999999999 ## Replacing values estimated as Infinite for 999999999

train.out <- train(x=X.train,

y=y.train,

penalty.factor = w3,

method = "glmnet",

metric = "RMSE",

trControl = tc,

tuneGrid = expand.grid(.alpha = 1,.lambda = lambda))

model\_fit <- glmnet(X.train,

y.train,

alpha = 1,

lambda = train.out$bestTune[1,2],

penalty.factor = w3)

pred <- predict(model\_fit,newx = rbind(X.train, X.test)) %>%

as.numeric

} else if (model == 'postlasso'){

train.out <- train(x=X.train,

y=y.train,

method = "glmnet",

metric = "RMSE",trControl = tc,

tuneGrid = expand.grid(.alpha = c(1),.lambda = lambda))

model\_lasso <- glmnet(X.train,

y.train,

alpha = 1,

lambda = train.out$bestTune[1,2])

nzvars <- predict(model\_lasso, type = "nonzero") %>% .[[1]]

train\_post <- as.data.frame(X.train) %>%

select(nzvars) %>%

mutate(y = y.train)

if(class(X.test)=='numeric'){

X.test %<>% as.tibble %>% t %>% set\_colnames(colnames(X.train))

}

test\_post <- as.data.frame(X.test) %>%

select(nzvars) %>%

mutate(y = 0)

model\_fit <- lm(y~., data = train\_post)

pred <- predict(model\_fit,newdata = rbind(train\_post,test\_post ))

} else if (model == 'rf'){

train.out <- NULL

model\_fit <- randomForest(x = X.train,

y = y.train,

ntree = 100,

nodesize = 5)

pred <- predict(model\_fit, newdata = rbind(X.train, X.test)) %>%

as.numeric

} else if (model =='ss'){

train.out <- NULL

model\_fit <- spikeslab(x = X.train,

y = y.train,

n.iter2 = 500,

bigp.smalln = ncol(X.train)>=nrow(X.train),

intercept = TRUE)

pred <- predict(model\_fit, newdata = rbind(X.train, X.test))$yhat.gnet

} else if (model == 'boost'){

tune\_grid <- expand.grid(nrounds = 100,

max\_depth = c(5),

eta = c(0.3),

gamma = 0,

colsample\_bytree = 0.3,

min\_child\_weight = 1,

subsample = 1)

train.out <- NULL

model\_fit <- train(x = X.train,

y = y.train,

method = "xgbTree",

metric = "RMSE",

tuneGrid = tune\_grid)

pred <- predict(model\_fit, newdata = rbind(X.train, X.test)) %>%

as.numeric

}

} else

{

if(!is.integer(h)){

message('h must be integer')

return(NULL)

}

df %<>% .[,'investment']

df$y <- df$investment

df %<>% na.omit

# проверка на start и end

if(startdt >= enddt){

message('start must be greater then end')

return(NULL)

}

startdt <- max(startdt %>% as.yearqtr,

first(time(df)) %>% as.yearqtr) %>%

as.Date

enddt <- min(enddt %>% as.yearqtr,

last(time(df)) %>% as.yearqtr) %>%

as.Date

if(df[paste0(startdt, "/", enddt)] %>% nrow < 48){

message('train set length must be greater then 48 quarters')

return(NULL)

}

train\_n <- which(time(df) %>% as.Date()==(startdt %>% as.yearqtr %>% as.Date)):

which((time(df) %>% as.Date())==

(enddt %>% as.yearqtr%>% as.Date))

test\_n <- (which((time(df) %>% as.Date())==(enddt%>% as.yearqtr%>% as.Date))+1):nrow(df)

y.train <- df$y[train\_n] %>% as.numeric

y.test <- df$y[test\_n] %>% as.numeric

y.full <- c(y.train, y.test)

if (model == 'arima'){

train.out <- NULL

model\_fit <- auto.arima(y.train,seasonal = FALSE, stationary = TRUE)

maxord <- arimaorder(model\_fit) %>% sum

if(h == 0){

pred <- Arima(y.full, model=model\_fit) %>% fitted() %>% as.numeric()

} else {

if(maxord == 1){

pred <- c()

} else{

pred <- rep(NA, maxord-1)

}

for(i in maxord:length(y.full)){

fit <- Arima(y.full[1:i],

model=model\_fit)

pred <- c(pred, forecast(fit, h) %>% as.data.frame %>% .[h,1])

}

}

} else if (model == 'rw'){

if(h == 0){

pred <- lag.xts(y.full)

} else{

pred <- y.full

}

train.out <- model\_fit <- NULL

}

}

list(

model = model,

series = series,

lag = lag,

startdt = startdt,

enddt = enddt,

date = time(df)[c(train\_n,test\_n)] %>% as.Date,

h = h,

pred = pred,

train.out = train.out,

model\_fit = model\_fit

)

}

out <- expand.grid(startdt = c(as.Date('1996-01-01'), as.Date('2000-01-01')),

enddt = seq(as.Date('2012-10-01'), as.Date('2030-10-01'), by = 'quarter'),

lag = c(0L),

h=c(1L:8L),

model = c('arima',

'boost',

'adalasso',

'postlasso',

'lasso',

'elnet',

'rf',

'rw',

'ss',

'ridge'

)

) %T>%

(function(x) {assign('N', nrow(x), envir = globalenv())}) %>%

split(seq(1:nrow(.))) %>%

imap(function(x, i){

train.model(startdt=x$startdt,

enddt=x$enddt,

model = x$model,

lag=x$lag,

h=x$h,

i = i,

N = N

)

})

%>%

map\_dfr(function(x){

data.frame(model=x$model,

lag = x$lag,

startdt= x$startdt,

enddt= x$enddt,

h = x$h,

date = x$date,

pred=x$pred)

})

save(out, file = 'out.RData')

ggplot(df %>% na.omit) +

geom\_line(aes(y = investment, x = time(df %>% na.omit)))+

labs(title = "",

y = "Валовое накопление основного капитала",

x = "Дата") +

theme\_bw()

df %>%

na.omit %>%

as.data.frame %>%

rownames\_to\_column('date') %>%

melt(id.vars = 'date') %>%

group\_by(variable) %>%

mutate(value = scale(value),

date = as.Date(as.yearqtr(date))) %>%

ggplot() +

geom\_line(aes(date, value, group=variable,

alpha = ifelse(variable %in% c('investment'), 1, 0.3)), show.legend = F)+

labs(title = "",

y = "",

x = "") +

scale\_x\_date(limits = c('1996-01-01', '2019-01-01') %>% as.Date)+

geom\_vline(xintercept =c('1996-01-01','2000-01-01') %>% as.Date, linetype='dashed')+

scale\_y\_continuous(limits = c(-5, 3))+

labs(x = 'Дата',

y='')+

theme\_bw()

out %>% filter(enddt < as.Date(as.yearqtr(date)-h/4)) %>%

group\_by(date, h, model, startdt) %>%

filter(enddt == max(enddt)) %>%

ungroup %>%

mutate(forecastdate = as.Date(as.yearqtr(date) -h/4)) %>%

mutate(pred = ifelse(h == 0, true, pred)) %>%

mutate(giftime =as.numeric(forecastdate)+0.2\*((date -forecastdate) %>%

as.numeric())) %>%

filter(forecastdate <='2019-01-01') %>% mutate(true = ifelse(date <= forecastdate, true, NA))%>% filter(startdt ==

'2000-01-01',

h<5)%>%

mutate(true\_na = ifelse(date <= forecastdate, true, NA)) %>%

ggplot()+

geom\_path(data = fordata %>%

mutate(true\_na = ifelse(date <= forecastdate, true, NA)) %>% na.omit,

aes(date, true\_na,

alpha = 'Наблюдаемые\nзначения',

color = 'Наблюдаемые\nзначения',

size = 'Наблюдаемые\nзначения',

linetype = 'Наблюдаемые\nзначения'))+

geom\_line(aes(date, pred,

group = interaction(startdt,

forecastdate),

alpha = 'Прогноз',

color = 'Прогноз',

size = 'Прогноз',

linetype = 'Прогноз')

)+

facet\_wrap(vars(model))+

scale\_y\_continuous(limits = c(-0.2, 0.15))+

labs(x = 'Дата',

y = 'Квартальное изменение валового

накопления\nосновного капитала (разность логарифмов)')