Banks DB descStats

Emi

30/03/2021

R Markdown

Descriptive statistics for a probit model. Data is from 1997q4 to 2001q12.

```
dbX <- haven::read_dta('C:/Users/emi.ABLE-22868/OneDrive/UWA PhD/bankFailure/data/failures-1997-2001-quarterly.dta') %>%
       dplyr::mutate(.,
         FECHAdata = quarter(ymd("1960-01-01")+months(FECHA_Q*3), with_year=TRUE),
         FECHAdataAnio = year(ymd("1960-01-01")+months(FECHA_Q*3)),
         .after = IDENT) %>%
    filter(FECHAdata >= 1997.4 & FECHAdata <= 2001.4)
#summary(cars)
```

FECHAdata contains the quarter in R format. It is built from the quarterly variable for Stata (FECHA_Q).

Descriptive statistics

Missing values as %

```
dbX %>%
  group_by(FECHAdataAnio) %>%
    summarise(across(c(ActivoN, C8Est_w, CAR_IRR_3A6, P_ROA, P_DEP_ARS_RATE, P_LOANS_ARS_RATE_W, APRSpNF_RATE_W, APR_USD_RAT
E, APR_RATE_W, GDP_D_Q, ARG_YTM),
                  list(M = ~round(( sum(is.na(.x))/sum(!is.na(IDENT)))*100)) )) %>%
  #filter(., Activo_N_MISSING > 0) %>%
  #kable(., caption='Missinv values for Activo')
  DT::datatable(., rownames=FALSE, filter='top',  caption='Percentage of missing values (%)',
          options = list(columnDefs = list(list(
                      targets = c(1,2,3,4,5),
                      render = JS(
                        "function(data, type, row, meta) {",
                        "return type === 'display' && data.length > 8 ?",
                        "'<span title=\"' + data + '\">' + data.substr(0, 8) + '...</span>' : data;",
                        "}")
))))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

Show 10 **entries** **Search:**

Percentage of missing values (%)

FECHAdataAnio	ActivoN_M	C8Est_w_M	CAR_IRR_3A6_M	P_ROA_M	P_DEP_ARS_RATE_M	P_LOANS_ARS_RATE_W_M
All	Al	All	All	A	All	All
1997	9	9	11	11	18	2
1998	3	3	3	3	10	1
1999	0	0	1	1	8	1
2000	0	0	23	2	20	1
2001	1	1	38	1	49	4

Showing 1 to 5 of 5 entries Previous 1 Next

Banks desc stats table

Sample features

```
tibble('N' = length(unique(dbX$IDENT)),
                    'T' = length(unique(dbX$FECHA_Q)),
                    'NxT' = N*T,
                    'Avg n' = dbX %>%
                                select(FECHA_Q, IDENT) %>%
                                group_by(IDENT) %>% summarise(n = n()) %$% mean(.$n)) %>%
  t(.) %>%
  DT::datatable(.) %>% formatRound(columns=c('V1'))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

Show 10 entries Search:

	V1
N	159.00
T	17.00
NxT	2,703.00
Avg n	12.81

Showing 1 to 4 of 4 entries Previous 1 Next

```
varsList <- c('ActivoN', 'C8Est_w', 'CAR_IRR_3A6', 'P_ROA', 'P_DEP_ARS_RATE', 'P_LOANS_ARS_RATE_W', 'APRSpNF_RATE_W', 'APR_U
SD_RATE', 'APR_RATE_W', 'GDP_D_Q', 'ARG_YTM')
dSVals <- dbX %>%
  summarise(across(all_of(varsList), list(
    'min' = ~min(.x, na.rm=TRUE),
    'median' = ~round(median(.x, na.rm=TRUE)),
    'mean' = ~mean(.x, na.rm=TRUE),
    'max' = ~round(max(.x, na.rm=TRUE)),
    'SD' = ~round(sd(.x, na.rm=TRUE)) )))

descStatsTibble <- tibble('min' = as_vector(select(dSVals, ends_with('min'))),
                    'median' = as_vector(select(dSVals, ends_with('median'))),
                    'mean' = as_vector(select(dSVals, ends_with('mean'))),
                    'sd' = as_vector(select(dSVals, ends_with('sd'))),
                    'max' = as_vector(select(dSVals, ends_with('max'))))

rownames(descStatsTibble) <- c('Assets $', 'Equity/Assets (%)', 'Non-perfomring loans/Loans (%)', 'ROA (%)', 'Deposits inter
st rate (%)', 'Loans interest rate (%)', 'Public sector loans/Loans (%)', 'USD loans/Loans (%)', 'Loans/Assets (%)', 'Seas-a
djusted quarterly chg GDP (%)', 'Country risk (%)')
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
  datatable(descStatsTibble) %>%
    formatRound(columns=c('min', 'median', 'mean', 'sd', 'max'))
```

Show 10 entries Search:

	min	median	mean	sd	max
Assets $	1,051.00	229,710.00	1,181,350.25	2,740,041.00	18,187,022.00
Equity/Assets (%)	0.14	13.00	22.17	21.00	96.00
Non-perfomring loans/Loans (%)	0.00	10.00	13.31	12.00	100.00
ROA (%)	-84.21	0.00	-0.52	6.00	31.00
Deposits interst rate (%)	0.00	5.00	5.26	4.00	30.00
Loans interest rate (%)	0.00	22.00	21.71	11.00	49.00
Public sector loans/Loans (%)	0.00	0.00	6.59	12.00	67.00
USD loans/Loans (%)	0.00	56.00	54.83	26.00	100.00
Loans/Assets (%)	0.00	55.00	53.10	19.00	93.00
Seas-adjusted quarterly chg GDP (%)	-5.69	0.00	-0.67	2.00	2.00

Showing 1 to 10 of 11 entries

APR_RATE_W is the ratio of Loans to Assets in percentage (%).

Failure descriptive statistics

Create quarterly dates and choose observations for banks alive by 1997q4.

```
dbEnts <- haven::read_dta('C:/Users/emi.ABLE-22868/OneDrive/UWA PhD/bankFailure/data/failures/failure_time.dta') %>%
       dplyr::mutate(.,
         EXIT_DATE_Q = if_else(is.na(EXIT_DATE),
                      9999,
                      quarter(as_date(EXIT_DATE, origin=ymd('1960-01-01')), with_year=TRUE) ),
         EXIT_DATE_Y = year(as_date(EXIT_DATE, origin=ymd('1960-01-01'))),
         START_Q = quarter(as_date(FIRST_DATE, origin=ymd('1960-01-01')), with_year=TRUE),
         .after = IDENT) %>%
  # Select only banks alive on 1997.4
         filter(START_Q <= 1997.4 & EXIT_DATE_Q > 1997.4)
```

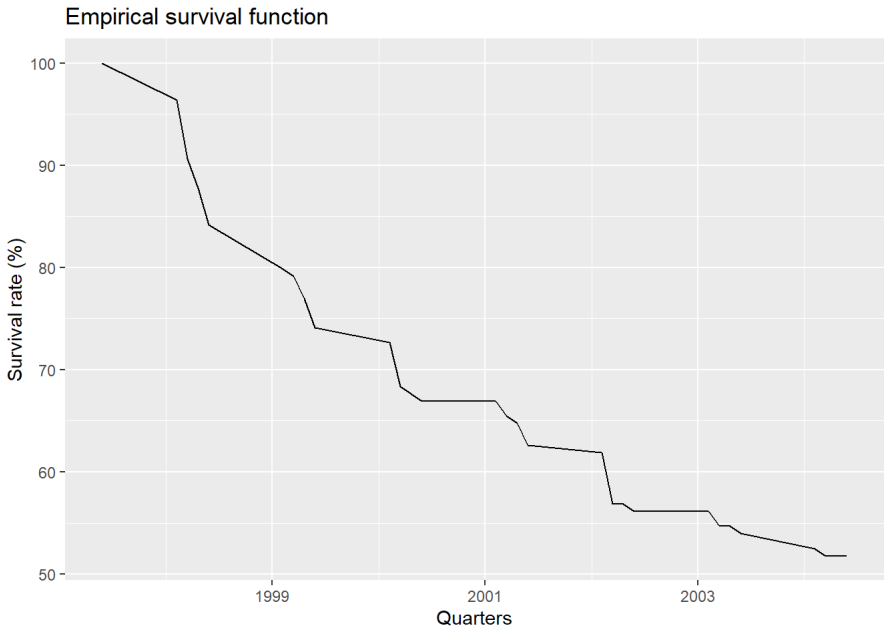
This give 139 entities

For each quarter I count how many banks die and are alive.

```
# Count failures by quarter
failuresByQ <- dbEnts %>%
               # TODO:  plot of exit by types group_by(EXIT_DATE_Q, EXIT_TYPE) %>%
               group_by(EXIT_DATE_Q) %>%
               count(., name='N_FAILS') %>%
               rename(Q = EXIT_DATE_Q)

allQs <- seq.Date(from=ymd('1997-10-01'), to=ymd('2004-12-31'), by='quarter') %>%
         quarter(., with_year = TRUE)
# For each quarter, count alives at the beginning of the quarter
alives <- map_int(allQs, function(thisQ)
    {
      dbEnts %>% arrange(IDENT) %>%
      filter(., START_Q <= thisQ & (EXIT_DATE_Q > thisQ) ) %>%
      nrow()

    })

failsByTime <- left_join( tibble('Q' = allQs,'TOTAL_ENTS' = alives),
                            failuresByQ,
                          by='Q') %>%
             replace(., is.na(.), 0) %>%
             mutate(., HAZ_RATE = (N_FAILS/TOTAL_ENTS)*100,
                       SUR_RATE = (TOTAL_ENTS/139)*100)
```

Plots:

```
#ggplot(data=failuresByQ, mapping=aes(x=Q, y=N_FAILS, fill=EXIT_TYPE)) +
# ggplot(data=failuresByQ, mapping=aes(x=Q)) +
# geom_col(aes(y=N_FAILS)) +
# scale_x_continuous(name='Quarters') +
# theme(axis.text.x = element_text(angle=90, vjust=0.5)) +
# scale_y_continuous(name='Failure rate (%)') +
# coord_cartesian(ylim=20) +
# labs(title='Instantaneous failure rate (hazard)')
#


ggplot(data=failsByTime, mapping=aes(x=Q, y=SUR_RATE)) +
  geom_line() +
  #geom_curve(aes(xend=)) +
  scale_x_continuous(name='Quarters') +
  scale_y_continuous(name='Survival rate (%)') +
  labs(title='Empirical survival function')
```



Export data to Matlab

Merge EXIT_DATE_Q with covariates X

In dbX, there are `nrow(dbX)` observations, and `length(unique(dbX$IDENT))` entities; and `ncol(dbX)` variables. There should be $n = 2037$, $np = 159$, and $k = 18$. In dbEnts, there are `nrow(dbEnts)` observations, and `length(unique(dbEnts$IDENT))` entities; and `ncol(dbEnts)` variables. There should be $n = 2037$, $np = 159$, and $k = 18$. All entities in dbEnts should have a match in dbX

Entitines in dbEnts without a match in dbX

```
xo <- anti_join(dbEnts, dbX, by='IDENT')
```

Choose X data on 1997-12.

```
XExt <- inner_join(select(dbEnts, IDENT, EXIT_DATE_Q),
                   select(dbX, IDENT, FECHAdata, ActivoN, C8Est_w, CAR_IRR_3A6, P_ROA, P_DEP_ARS_RATE, P_LOANS_ARS_RATE_W, A
PRSpNF_RATE_W, APR_USD_RATE, APR_RATE_W),
                   by='IDENT') %>%
          mutate('Y' = if_else(EXIT_DATE_Q <= 2004.12, 0, 1), .after=EXIT_DATE_Q) %>%
        filter_all(all_vars(!is.na(.)))
Xt <- XExt %>%
      filter(FECHAdata == 1997.4) %>%
      select(!c(IDENT, Y, EXIT_DATE_Q, FECHAdata))

Yt <- XExt %>%
        filter(FECHAdata == 1997.4) %>%
        select(Y)
```

```
X <- Xt %>%
        mutate('constant' = 1, .before=ActivoN) %>%
        write_csv(., 'probit/X.csv')

Y <- Yt %>%
    write_csv(., 'probit/Y.csv')
# solve(t(X) %*% X) %*% t(X) %*% Y

# For the probit the macro vars have no role
```

Try to estimate here

```
#h <- rms::orm(Y ~ Xt$ActivoN + Xt$C8Est_w + Xt$CAR_IRR_3A6 + Xt$P_ROA + Xt$P_DEP_ARS_RATE + Xt$P_LOANS_ARS_RATE_W + Xt$APRS
pNF_RATE_W + Xt$APR_USD_RATE + Xt$APR_RATE_W, family=probit)
p <- glm(as.matrix(Y) ~ X$constant + X$ActivoN + X$C8Est_w + X$CAR_IRR_3A6 + X$P_ROA + X$P_DEP_ARS_RATE + X$P_LOANS_ARS_RATE
_W + X$APRSpNF_RATE_W + X$APR_USD_RATE + X$APR_RATE_W, family=binomial)

p
```

```
##
## Call:  glm(formula = as.matrix(Y) ~ X$constant + X$ActivoN + X$C8Est_w +
##     X$CAR_IRR_3A6 + X$P_ROA + X$P_DEP_ARS_RATE + X$P_LOANS_ARS_RATE_W +
##     X$APRSpNF_RATE_W + X$APR_USD_RATE + X$APR_RATE_W, family = binomial)
##
## Coefficients:
##          (Intercept)            X$constant             X$ActivoN
##            1.823e+00                    NA             2.579e-07
##            X$C8Est_w         X$CAR_IRR_3A6               X$P_ROA
##            1.861e-02            -4.717e-02             1.346e-01
##     X$P_DEP_ARS_RATE  X$P_LOANS_ARS_RATE_W      X$APRSpNF_RATE_W
##            9.762e-02            -9.006e-03             1.824e-02
##       X$APR_USD_RATE          X$APR_RATE_W
##           -1.417e-02            -2.402e-02
##
## Degrees of Freedom: 110 Total (i.e. Null);  101 Residual
## Null Deviance:       152.4
## Residual Deviance: 129.1     AIC: 149.1
```