IFRS9 Forward-Looking Modeling

OLS Regression and Predictor Importance

Andrija Djurovic

www.linkedin.com/in/andrija-djurovic

IFRS9 Forward-Looking Modeling

- With the introduction of IFRS9, one key requirement practitioners had to address was the inclusion of reasonable and supportable information available without undue cost or effort when measuring expected credit losses.
- Several regulatory paragraphs also emphasized incorporating such information when recognizing lifetime expected credit losses.
- After several years of implementation, the most common approach observed in practice quantifies the macroeconomic environment's effect on a bank's risk parameters, usually aggregated at the segment level. For example, practitioners calculate the default rate at the segment level and regress it against macroeconomic indicators.
- Many models rely on Ordinary Least Squares (OLS) regression but differ in their design and level of aggregation.
- Given the regression design, when analyzing forward-looking (FLI) models, practitioners often assess the importance of each predictor in the final model.
- In doing so, the most commonly used approach in practice is to assess predictor importance using standardized regression coefficients. This raises the question of whether this is the only meaningful way to interpret a predictor's importance, and what other approaches may offer additional insights.

OLS Regression and Predictor Importance

According to Achen (1982), in his book Interpreting and Using Regression, the concept of variable importance is ambiguous without clarifying: important for what purpose?
He classifies importance into three groups, each answering a different question:

1. Theoretical Importance
2. Level Importance
3. Dispersion Importance

The following slides briefly explain each type of importance and illustrate their calculations using `R` and `Python`. The data used for these examples can be imported using the code below.

`R` Code:

```r
url <- "http://andrija-djurovic.github.io/adsfcr/model_dev_and_vld/db_pi.csv"
db <- read.csv(url)
```

`Python` Code:

```python
import pandas as pd

url = "http://andrija-djurovic.github.io/adsfcr/model_dev_and_vld/db_pi.csv"
db = pd.read_csv(url)
```

Theoretical Importance

Conceptual Overview:

- Theoretical importance measures the potential effect of a variable, holding all else constant.
- It is captured by the unstandardized coefficient (β) in the regression and indicates how much the dependent variable changes per unit increase in the predictor.
- The interpretation of this type of importance is not sample-dependent.
- Meaningful comparisons between predictors can only be made if they are on the same scale.
- For IFRS9 FLI modeling, it helps answer questions such as: "If GDP increases by one percentage point, how much would we expect the default rate to change?"
- In this context, practitioners often refer to this importance as sensitivity.

R Code:

```
#ols regression
ols <- lm(formula = odr ~ gdp + unemployment,
          data = db)
#theoretical importance
coef(ols)[-1]
```

```
##          gdp unemployment
##   -0.2586861    0.1682093
```

Theoretical Importance cont.

Python Code:

```
import numpy as np
import statsmodels.formula.api as smf

#ols regression
ols = smf.ols(formula = "odr ~ gdp + unemployment",
              data = db).fit()
#theoretical importance
ols.params.drop("Intercept")
```

```
## gdp            -0.258686
## unemployment    0.168209
## dtype: float64
```

Level Importance

Conceptual Overview:

- Level importance measures a predictor's actual contribution to the mean level of the dependent variable in a given sample.
- It is computed as the product of the regression coefficient (β) and the mean of the predictor.
- It is sample-dependent and reflects historical influence, not abstract potential.
- Although it is rarely used in practice in IFRS9 FLI modeling, it helps answer important questions such as: "How much of the observed average default rate in the past is explained by average unemployment?"
- Despite reflecting historical influence, it plays a valuable role in this context by clarifying the contribution of predictors to the forecasted value, thereby supporting the assessment of forecast validity and the impact of predictors.

R Code:

```r
#ols regression
ols <- lm(formula = odr ~ gdp + unemployment,
          data = db)
#predictor coefficients
coef.p <- coef(ols)[-1]
#gdp mean
gdp.m <- mean(db$gdp)
#unemployment mean
unempl.m <- mean(db$unemployment)
```

Level Importance cont.

R Code cont:

```
#level importance
coef.p * c(gdp.m, unempl.m)
```

```
##           gdp  unemployment
## -0.0037307972  0.0002566554
```

Python Code:

```
import numpy as np
import statsmodels.formula.api as smf

#ols regression
ols = smf.ols(formula = "odr ~ gdp + unemployment",
              data = db).fit()
#predictor coefficients
coef_p = ols.params.drop("Intercept")
#gdp mean
gdp_m = db["gdp"].mean()
#unemployment mean
unempl_m = db["unemployment"].mean()
#level importance
coef_p * np.array([gdp_m, unempl_m])
```

```
## gdp           -0.003731
## unemployment   0.000257
## dtype: float64
```

Dispersion Importance

Conceptual Overview:

- Dispersion importance measures how much variation in the dependent variable is explained by a predictor.
- It is captured by the standardized coefficient.
- It shows only how much of the spread is explained rather than indicating the effect of a one-unit change or the average contribution.
- In the context of IFRS9, it is the most commonly used measure and helps answer questions such as: "How much of the spread of the dependent variable is explained by each predictor in a given sample?"
- The following code demonstrates two different ways to calculate standardized regression coefficients. Method 1 applies OLS regression to a standardized dataset, while Method 2 transforms unstandardized regression coefficients into standardized ones.

R Code for Method 1:

```
#ols regression on standardized variables
ols.s <- lm(formula = scale(odr) ~ scale(gdp) + scale(unemployment),
            data = db)
#dispersion importance
coef(ols.s)[-1]
```

```
##          scale(gdp) scale(unemployment)
##          -0.8344894           0.2612825
```

Dispersion Importance cont.

R Code for Method 2:

```
#ols regression
ols <- lm(formula = odr ~ gdp + unemployment,
          data = db)
#unstandardized coefficients
coef.p <- coef(ols)[-1]
#standard deviation of odr
sd.odr <- sd(db$odr)
#standard deviation of predictors
sd.p <- sapply(X = db[, c("gdp", "unemployment")],
               FUN = sd)
#dispersion importance
coef.p * sd.p / sd.odr
```

```
##          gdp unemployment
##   -0.8344894    0.2612825
```

Dispersion Importance cont.

Python Code for Method 1:

```
import numpy as np
import statsmodels.formula.api as smf

#variable standardization
db["odr_z"] = (db["odr"] - db["odr"].mean()) / db["odr"].std()
db["gdp_z"] = (db["gdp"] - db["gdp"].mean()) / db["gdp"].std()
db["unemployment_z"] = (db["unemployment"] - db["unemployment"].mean()) / \
                        db["unemployment"].std()
#ols regression on standardized variables
ols_s = smf.ols(formula = "odr_z ~ gdp_z + unemployment_z",
                data = db).fit()
#dispersion importance
ols_s.params.drop("Intercept")
```

```
## gdp_z           -0.834489
## unemployment_z   0.261282
## dtype: float64
```

Dispersion Importance cont.

Python Code for Method 2:

```
#ols regression
ols = smf.ols(formula = "odr ~ gdp + unemployment",
              data = db).fit()
#ustandardized coefficients
coef_p = ols.params.drop("Intercept")
#standard deviation of odr
sd_odr = db["odr"].std()
#standard deviation of predictors
sd_p = db[["gdp", "unemployment"]].std()
#dispersion importance
coef_p * sd_p / sd_odr
```

```
## gdp            -0.834489
## unemployment    0.261282
## dtype: float64
```