Why Stata is the best programming language to start data analysis

Milós Koren Márton Fleck

What are we comparing?

1. Programming language
2. Software application
3. Documentation
4. Community

Programming language

- human readable syntax
- designed for data analysis
- good default options

Two-Column Slide

Brattle

A typical day for Brattle RAs includes:

- Combining economic theory and industry knowledge to solve real problems
- Diving into data, using statistical analyses to extract information from messy data
- Constructing models from a blend of theoretical concepts to answer complex questions
- Reviewing literature and industry trends to understand the debate around key developments
- Conducting statistical analysis and working with data using tools such as Stata, R, Excel or Python
- Auditing and contributing to the creation of financial, economic, and operational models

COMPASS LEXECON

Key responsibilities:

- Interacting extensively with clients to gain insight into their industry
- Contributing to development of theoretical and empirical approach
- Utilising literature to support economic arguments
- Efficiently conducting empirical analysis using Excel and Stata
- Overseeing the day-to-day running of the project
- Drafting reports summarising analysis
- Delivering an accurate and high-quality work product
- Participating actively in client meetings and conference calls
- Extensive mentoring and supervising of junior staff

Code Example

```
/* Hotel price data */
use "hotels-europe_price.dta", clear
/* Add hotel features (location,
  stars, ratings, etc.) */
merge m:1 hotel_id using
  "hotels-europe_features.dta"
/* Censor prices that are too high */
replace price = 1000 if price > 1000
/* Regress price on ratings, stars,
  plus month, weekend dummies */
regress price rating stars i.month
  i.weekend, vce(cluster country)
```

Code Example

```
/* Hotel price data */
use "hotels-europe_price.dta", clear
/* Add hotel features (location,
   stars, ratings, etc.) */
merge m:1 hotel_id using
   "hotels-europe_features.dta"
/* Censor prices that are too high */
replace price = 1000 if price > 1000
/* Regress price on ratings, stars,
   plus month, weekend dummies */
regress price rating stars i.month
   i.weekend, vce(cluster country)
```

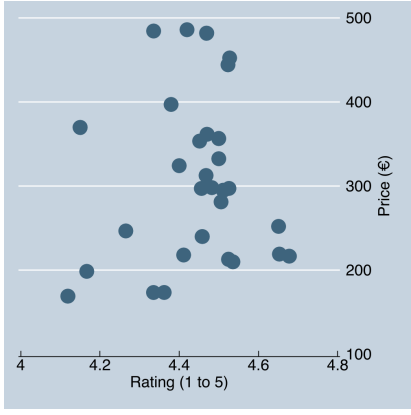
```
Linear regression                               Number of obs     =    115,367
                                                F(10, 30)         =     272.88
                                                Prob > F          =     0.0000
                                                R-squared         =     0.2577
                                                Root MSE          =     146.52

                                     (Std. Err. adjusted for 31 clusters in country)
------------------------------------------------------------------------------
             |               Robust
       price |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      rating |     21.5814   7.861631     2.75   0.010     5.52581    37.63699
       stars |    52.54748   8.304822     6.33   0.000     35.58677    69.50819
             |
       month |
           2 |    6.944091   5.554252     1.25   0.221    -4.399204    18.28739
           3 |    22.07722   5.573216     3.96   0.000     10.6952    33.45925
           4 |     29.2734   4.929571     5.94   0.000     19.20587    39.34093
           5 |    40.27256   4.755351     8.47   0.000     30.56084    49.98428
           6 |    40.54402   5.855406     6.92   0.000     28.58568    52.50235
          11 |    9.108877   4.401348     2.07   0.047     .1201249    18.09763
          12 |    187.1044   15.04021    12.44   0.000     156.3882    217.8206
             |
   1.weekend |    1.828793   6.036309     0.30   0.764    -10.49899    14.15658
       _cons |   -142.8199   16.73315    -8.54   0.000    -176.9935   -108.6462
------------------------------------------------------------------------------
```

Code Example

```
/* keep only 5-star hotels */
keep if stars == 5
/ * mean price and rating by country */
collapse (mean) price (mean) rating,
  by(country)
label variable price "Price (€)"
label variable rating "Rating (1 to 5)"
scatter price rating, scheme(economist)
```

Code Example

```
/* keep only 5-star hotels */
keep if stars == 5
/ * mean price and rating by country */
collapse (mean) price (mean) rating,
  by(country)
label variable price "Price (€)"
label variable rating "Rating (1 to 5)"
scatter price rating, scheme(economist)
```



Stata vs R

```
scatter price rating, scheme(economist)
```

```
ggplot(five_s
```

Stata vs Python

```
replace price = 1000 if price > 1000
```

```
data.loc[data
```

Same in Python

```
import pandas as pd
import matplotlib.pyplot as plt

# load hotel price data
price_data = pd.read_stata("hotels-europe_price.dta")

# add hotel features (location, stars, ratings, etc.)
features = pd.read_stata("hotels-europe_features.dta")
data = price_data.merge(features, on="hotel_id", how="left")

# replace high prices with 1000
data.loc[data["price"] > 1000, "price"] = 1000

# regress price on ratings, stars, plus month, weekend dummies
data = pd.get_dummies(data, columns=["month", "weekend"])
result = sm.OLS(data["price"], data[["rating", "stars"] + list(data.columns[data.columns.str.startswith("month_")])
  + list(data.columns[data.columns.str.startswith("weekend_")])]).fit(cov_type="cluster", cov_kwds={"groups": data["country"]})

# keep only 5-star hotels
data = data[data["stars"] == 5]

# calculate mean price and rating by country
data = data.groupby("country").mean()[["price", "rating"]]

# label variables
data.rename(columns={"price": "Price (€)", "rating": "Rating (1 to 5)"}, inplace=True)

# scatterplot
data.plot(x="Price (€)", y="Rating (1 to 5)", kind="scatter", colormap="tab10", figsize=(8, 6))
plt.show()
```

Same in R

```r
library(tidyverse)
library(ggplot2)

# load hotel price data
price_data <- read_dta("hotels-europe_price.dta")

# add hotel features (location, stars, ratings, etc.)
features <- read_dta("hotels-europe_features.dta")
data <- left_join(price_data, features, by="hotel_id")

# replace high prices with 1000
data <- data %>% mutate(price=if_else(price > 1000, 1000, price))

# regress price on ratings, stars, plus month, weekend dummies
data <- data %>% mutate(month=factor(month), weekend=factor(weekend)) %>% nest(-country)
result <- data %>% mutate(model=map(data, ~ lm(price ~ rating + stars + month + weekend, data=.)),
                          summ=map(model, broom::tidy)) %>%
                  unnest(summ)

# subset data for 5-star hotels only
five_star_data <- data %>% filter(stars == 5) %>%
                          group_by(country) %>%
                          summarize(mean_price=mean(price), mean_rating=mean(rating))

# create scatterplot
ggplot(five_star_data, aes(x=mean_price, y=mean_rating)) +
  geom_point() +
  labs(x="Price (€)", y="Rating (1 to 5)") +
  scale_color_economist()
```