Graphical and Statistical Packages

Jamie Saxon

Introduction to Programming for Public Policy

November 13, 2017

Outline

Focus thus far: basic programming, finding resources, and manipulating (new) file formats (html, json, SQL, etc.).

This week: anchor these data skills in exploratory data analysis.

- **seaborn**: fast and attractive plotting library (matplotlib+).
- **scipy.stats**: basic statistical tests (correlation, *t*-tests, &c.).
- **statsmodels**: 'stata for python' – OLS up to complex models.

I will cover a tiny fraction of these libraries. Check them out!

Time permitting: pointers for classification problems and fancier methods.

- **scikit-learn**: clustering, classification, and machine learning toolkit.

Seaborn

```
import seaborn as sns
```

Seaborn does two things for us:

1. It changes the matplotlib defaults and thus the **aesthetics** of any plot through pandas.
2. It provides methods for making **attractive plots**. Among others:
   - **distplot**: histograms with KDE or PDF fit.
   - **regplot**: scatter plot and regression with error bands.
   - **boxplot** or **violinplot**: Tukey's IQR + whiskers, and modern variants.
   - **pointplot**: point estimates with bootstrap CIs.
   - **jointplot**: simultaneous 1D and 2D – scatter, hex grid, KDE, etc. (Good alternative for scatter with too many observations.)
   - ...

Seaborn Aesthetics

Seaborn will change the aesthetics. You can manipulate them further:

- `sns.set_style("whitegrid")` changes total look to one of seaborn's (good) defaults.
- `sns.set_context("notebook")` changes the sizing of the text.

Both of these allow you to fine-tune any further parameters. (This actually directly wraps matplotlib.)

```
sns.set_style("whitegrid",
              rc={'axes.linewidth': 2.5})

sns.set_context('notebook', font_scale=1.45,
                rc={"lines.linewidth": 3,
                    "figure.figsize" : (7, 3)})
```

Seaborn Plotting

- Each Seaborn method has a ~dozen arguments. Basically, you have to provide the data.
- Most methods return a (manipulable) axis.

```
ax = sns.boxplot(data = ipums,
                 x = "EDUC5", y = "INCTOTK",
                 hue = "SEX", linewidth = 2)

ax.get_legend().set_bbox_to_anchor((1.3, 1))
ax.set_ylim(0, 300)
ax.set_ylabel("Income [Thousands]")
ax.set_xlabel("Education")

ax.figure.savefig("income_box.pdf",
                  bbox_inches='tight',
                  pad_inches=0.05)
```

Scipy

```
from scipy.stats import pearsonr, ttest_ind
```

- Scipy is one-stop shopping for statistical tests:
  - ***t*-test**: `ttest_ind(a, b, equal_var = False)`.
  - **Pearson's *r***: `pearsonr(x, y)`
  - **Spearman's rank correlation**: `spearmanr(x, y)`
  - **Linear regression**: `linregress(x, y)`
- It provides methods for many **PDFs** (e.g., **norm**).
  - Among others: `rvs()` (random variables), `pdf()`, `cdf()`.
  - If you want to mock up a model, these can be extremely helpful.

Statsmodels (OLS and WLS)

Statsmodels provides intuitive model building (patsy). You'll need:

```
import statsmodels.formula.api as smf

formula = "np.log(INCTOT) ~ AGE + RACE + EDUC + SEX"
ols = smf.wls(formula = formula, data = ipums,
                          weights = ipums["PERWT"])
model = ols.fit()
model.summary()
```

- Patsy understands numpy functions (log, pow, exp, etc.).
- Strings are interpreted as categories. To treat a numerical value as a category (sex, state, etc.), you can use `C(state)`.
- Interactions and variables are specified `A*B`, while `A:B` is the interactions only.
- After fitting, check `model.summary()` or retrieve `model.params` or `model.resid`.