Python for R Users - Cheat Sheet

Quick correspondences for common data, wrangling, stats, viz, and modeling tasks. Updated Feb 18, 2026.

1) Core data structures

Concept	Python	R
Numeric vector	**list**, **numpy.array**	**c()**
Matrix	**numpy.array** (2D)	**matrix()**
Data frame	**pandas.DataFrame**	**data.frame()** / **tibble()**
1D labeled	**pandas.Series**	named vector
Categorical	**pandas.Categorical**	**factor()**
Missing value	**numpy.nan** / **pd.NA**	**NA**

2) Data wrangling: pandas ↔ dplyr

Task	Python (pandas)	R (dplyr/tidyr)
Select columns	df[['a','b']]	select(df, a, b)
Filter rows	df[df.a > 5]	filter(df, a > 5)
Add/transform column	df['c'] = ...	mutate(df, c = ...)
Group + summarize	df.groupby('a').agg(...)	df %>% group_by(a) %>% summarise(...)
Sort	df.sort_values('a')	arrange(df, a)
Join	pd.merge(x, y, on='id', how='left')	left_join(x, y, by = 'id')
Distinct	df.drop_duplicates()	distinct(df)
Rename	df.rename(columns={'old':'new'})	rename(df, new = old)
Pivot wider	df.pivot(index=..., columns=..., values=...)	pivot_wider(df, names_from=..., values_from=...)
Pivot longer	pd.melt(df, id_vars=[...])	pivot_longer(df, cols=..., names_to=..., values_to=...)

3) Stats: common tests & summaries

Thing	Python	R
Mean / SD	np.mean(x) / np.std(x, ddof=1)	mean(x) / sd(x)
Median / quantile	np.median(x) / np.quantile(x, q)	median(x) / quantile(x, probs=q)
Correlation	scipy.stats.pearsonr(x,y)	cor(x,y)
t-test	scipy.stats.ttest_ind(x,y)	t.test(x,y)
Chi-square	scipy.stats.chi2_contingency(tbl)	chisq.test(tbl)
Linear regression	statsmodels.formula.api.ols('y ~ x1 + x2', data=df).fit()	lm(y ~ x1 + x2, data=df)
Logistic regression	statsmodels.formula.api.logit('y ~ x', data=df).fit()	glm(y ~ x, data=df, family=binomial())

4) Visualization: matplotlib/seaborn ↔ ggplot2

Plot	Python	R
Scatter	sns.scatterplot(data=df, x='x', y='y')	ggplot(df, aes(x, y)) + geom_point()
Line	plt.plot(x, y)	ggplot(df, aes(x, y)) + geom_line()
Histogram	plt.hist(x, bins=30)	ggplot(df, aes(x)) + geom_histogram(bins=30)
Boxplot	sns.boxplot(data=df, x='g', y='y')	ggplot(df, aes(g, y)) + geom_boxplot()

5) Machine learning: scikit-learn ↔ caret/tidymodels

Task	Python	R
Train/test split	from sklearn.model_selection import train_test_split	rsample::initial_split() / caret::createDataPartition()
Fit model	model.fit(X_train, y_train)	fit(model, data=train) / train(...)
Predict	model.predict(X_test)	predict(model, newdata=test)
Random forest	RandomForestClassifier / RandomForestRegressor	ranger / randomForest
Cross-validation	cross_val_score(model, X, y, cv=5)	vfold_cv(data, v=5)

6) File I/O

Task	Python	R
Read CSV	pd.read_csv('file.csv')	read.csv('file.csv') / readr::read_csv()
Write CSV	df.to_csv('file.csv', index=False)	write.csv(df, 'file.csv', row.names=FALSE)
Read Excel	pd.read_excel('file.xlsx')	readxl::read_excel('file.xlsx')
Read JSON	json.load(open('file.json'))	jsonlite::fromJSON('file.json')
Save object	pickle.dump(obj, open('x.pkl','wb'))	saveRDS(obj, 'x.rds')
Load object	pickle.load(open('x.pkl','rb'))	readRDS('x.rds')

Handy mental mapping

• **%>%** pipe ≈ method chaining (**df.assign(...).query(...)**) or **.pipe()**
• **tibble** ≈ **DataFrame** (print-friendly, column-first)
• **formula** (**y ~ x**) ≈ statsmodels formula strings
• **NA** handling: in pandas watch out for **NaN** vs **pd.NA** and dtype casting

Tip: If you tell me what packages you use most in R (tidyverse, data.table, base, etc.), I can tailor a version that matches your exact workflow.