Linear Modeling in R

Cheat Sheet

Model type	Probability distribution	Dependent variable	Variable types	Independent variable(s)	R

linear models	Normal	continuous	fixed effects (main effects, interactions, polynomials)	nominal, continuous	nominal IV
					`lm(y ~ factor(x))`
					continuous IV
					`lm(y ~ x)`
					interaction
					`lm(y ~ x*z)`

			random effects	nominal	`lmer(y ~ x + (1|τ))`

generalized linear models	Poisson	count	fixed effects (main effects, interactions, polynomials)	nominal, continuous	`glm(y ~ x, family=poisson)`
					model rates with offset
					`glm(y ~ x, offset=log(t), family=poisson)`
					model overdispersion with negative binomial
					`glm.nb(y ~ x)`

	binomial	binary			`glm(y ~ x, family=binomial)`
		count (*y* successes, *n* trials)			`glm(cbind(y, n-y) ~ x, family=binomial)`
			random effects	nominal	`glmer(y ~ x + (1|τ), family=family)`

Parameter interpretation & model comparison

- parameter interpretation
- ✓ possible fixes to failure to meet assumptions

LM with nominal IV's (ANOVA)

- extract ANOVA table to view sum-of-squares: `aov()`
- ✓ overall model test, *F*-test: `anova()`
- ✓ *post-hoc* tests to test for differences in factor levels: `tukeyHSD()`

LM with nominal and continuous IV's

- partial regression plots: `avPlots()`
- ✓ compare nested models with partial F-test, e.g., `anova()`
- ✓ compare non-nested models with AIC or adjusted R^2, e.g. `AIC()`

GLM with Poisson distribution (Poisson regression)

- uses log link, coefficients are outputted on log scale
- exponentiate coefficients to interpret them as ratios, e.g., `exp(coef())`
- ✓ compare nested models with likelihood ratio test: `lrtest(), anova(…, test="Chisq")`
- ✓ compare non-nested models with AIC, e.g. `AIC()`

GLM binary logistic regression

- uses logit link, coefficients are outputted as log-odds
- exponentiate coefficients to interpret them as odds, e.g., `exp(coef(mod))`
- take inverse logit to predict as probability, e.g., `inv.logit()`
- ✓ same model comparison as Poisson regression

GLM binomial count logistic regression

- same coefficient interpretation as binary logistic regression
- ✓ same model comparison as Poisson regression

G/LM assumptions and testing

- assumptions

✓ possible fixes to failure to meet assumptions

DV is Independently & Identically Distributed

- each sample must have the same distribution and all samples must be mutually independent
- assess study design: are observations correlated?
- plot data in order they were collected/measured: do not want patterns

✓ include random effects to account for nested design, spatial or temporal autocorrelation

DV & IV are linearly related

- evaluate graphically, e.g. `ggpairs()`
- residuals must be ~ normally distributed with a mean of 0; examine residuals, e.g., `plot(model)`, `qqnorm()`, `qqline()`

✓ transform DV, e.g. `log()`

✓ add interaction or polynomial to IV to account for non-linearity

No strong multicollinearity

- are IV's strongly correlated ($r \geq 0.70$)?
- assess graphically or check correlations among IV's
- test variance inflation factor after running model, e.g. `vif()`; VIF should be < 5

✓ remove 1 of the highly correlated IV's – often IV with lowest correlation with DV

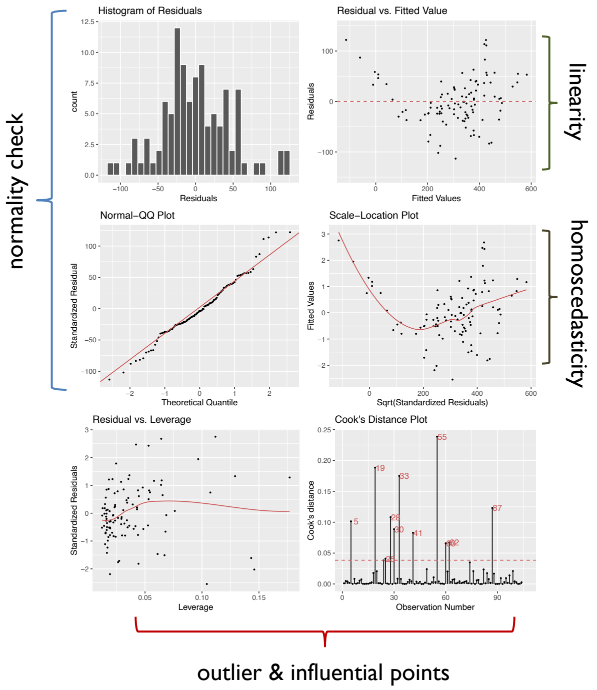
✓ combine correlated IV's into a single variable

Constant variance or homoscedasticity – LM's

- evaluate graphically: does spread of DV increase over continuous IV's?
- test equality of variance of DV for nominal IV's, e.g. `var.test()`
- assess with residual plot: do standardized residuals look like a cloud of points without pattern?

✓ transform DV with log or other transform, e.g. `log()`

✓ use a different model



Link function correctly specified – GLM's

- evaluate relationship between mean and variance; for Poisson and binomial count models no overdispersion, $\phi > 2$, e.g. `dispersiontest()`, `check_overdispersion()`?
- check model goodness of fit, e.g. `pchisq()`
- check relative fit of model, e.g. `pr2()`, `r.squaredGLMM()`

✓ use quasipoisson or quasibinomial models to scale standard errors of coefficients

✓ employ negative binomial model for counts, e.g. `glm.nb()`

✓ add an observation-level random effect to account for extra variance

No highly influential observations

- assess graphically with leverage plot; `plot(model)`
- calculate Cook's distance, e.g., `cooks.distance()`, observations with cook's distance > 4 times the mean may be influential
- check with `outlierTest()`

✓ check for data entry or transcription errors

✓ remove extreme or highly influential datapoints (must have a reason to do so!)

✓ present results with and without highly influential datapoints