GLMNET

Instructor: Davide Proserpio

What is GLMNET?

- **GLMNET** is an R package (part of the glmnet library) for fitting *regularized generalized linear models*.
- It efficiently handles **high-dimensional data** (many features, sparse matrices).
- Supports various models:
  - Linear Regression
  - Logistic Regression (binary or multinomial)
  - Poisson Regression, Cox, etc.
- Adds a **penalty term** to control model complexity → prevents overfitting.

Penalty

GLMNET minimizes:

$$\text{Loss}(\boldsymbol{\beta}) \;=\; \underbrace{\text{DataFit}(\boldsymbol{\beta})}_{\text{depends on model}} \;+\; \lambda \underbrace{\left[\alpha\|\boldsymbol{\beta}\|_1 + (1-\alpha)\|\boldsymbol{\beta}\|_2^2\right]}_{\text{Elastic Net penalty}}$$

Where:

- **λ (lambda)** = regularization strength
- **Penalty** = penalty term (Lasso, Ridge, or both)
- **Smaller λ** → more flexible, coefficients can take large values, can overfit
- **Larger λ** → more shrinkage (some coefficients are shrunk towards zero, simpler model)

Penalty: Choosing α

- The penalty itself depends on a parameter called α:
  - 1: Lasso Regression → sets some coefficients **exactly to 0** (feature selection)
  - 0: Ridge regression → **shrink** coefficients **towards 0**
  - $0 < \alpha < 1$**:** Elastic Net → mix of both

- Rule of thumb:
  - Start with **Elastic Net** (e.g., alpha = 0.5).
  - Prefer **Ridge** for collinearity + no need for selection.
  - Prefer **Lasso** when you value **sparse, interpretable** models and suspect many features are noise.

Penalty: Choosing λ

- GLMNET automatically fits models for many λ values along a **regularization path**.
- Typically, use **cross-validation** to pick the best one.
- cv.glmnet() returns:
  - **lambda.min** → λ giving the best performance (lowest error / highest AUC)
  - **lambda.1se** → largest λ within 1 standard error of the best (simpler, more stable)

Comparison with traditional glm

```
glm(target ~ x1 + x2 + x3, data = train, family =
binomial)
```

- Fits a **classical logistic regression**.
- Minimizes the **log-likelihood** (no penalty on coefficients).
- All variables are included unless removed manually.

Comparison with traditional glm

```
glmnet(X_train, y_train, family = "binomial",
     alpha = 1,
     lambda = 0.01)
```

- Fits a **penalized logistic regression**.
- Adds a **regularization term**
- **Key Parameters:**
  - Lambda: controls regularization strength.
  - Alpha: controls the type of penalty.
    - alpha = 1: Lasso (L1 → some coefficients = 0).
    - alpha = 0: Ridge (L2 → coefficients shrink but stay ≠ 0).
    - 0 < alpha < 1: Elastic Net (mix).

```
cv.glmnet(X_train, y_train, family = "binomial",
         type.measure = "auc",
         nfolds = 5,
         alpha = 1)
```

5-fold cross-validation with Lasso penalty that finds the optimal lambda that maximizes AUC