⭐ REGRESSION: Linear, Ridge, Lasso, Elastic Net

---

1. Linear Regression

Goal: Predict y using a straight line.
Model: $h(x) = θ0 + θ1x$

Cost Function (MSE):
$J(θ) = (1 / 2m) * Σ (h(x_i) – y_i)^2$

Why MSE?

- Differentiable
- Penalizes larger errors more

Problem:

- Overfits when data is noisy
- Struggles with correlated features

🎯 The Goal of Linear Regression

The goal of linear regression is:

To find the best values of weights (m, w_1, w_2...) and bias (c/b) that minimize the error between actual Y and predicted Y.

This is done by minimizing **cost function**, usually:

MSE=n1∑(Ytrue−Ypred)2

We choose parameters **so that MSE becomes minimum**.

---

2. Ridge Regression (L2 Regularization)

Purpose: Reduce overfitting by shrinking weights.

Cost Function:
$J(\theta) = (1 / 2m) * \Sigma (h(x_i) - y_i)^2 + \lambda \Sigma \theta_{\square}^2$

Key Points:

- $\lambda \geq 0$ controls penalty
- $\lambda = 0 \rightarrow$ same as linear regression
- Shrinks weights but never makes them zero
- Good when all features are important

Effect:

- Reduces overfitting
- Gives smooth, stable weights

---

3. Lasso Regression (L1 Regularization)

Purpose: Overfitting reduction + feature selection.

Cost Function:
$J(\theta) = (1 / 2m) * \Sigma (h(x_i) - y_i)^2 + \lambda \Sigma |\theta_{\square}|$

Key Points:

- L1 penalty can make weights exactly zero
- Removes irrelevant features
- Produces sparse models
- Useful when many features are irrelevant

Effect:

- Model becomes simpler
- Automatic feature selection

---

4. Elastic Net (L1 + L2)

Purpose: Combine Ridge + Lasso benefits.

Cost Function:
$J(\theta) = (1 / 2m) * \Sigma (h(x_i) – y_i)^2 + \lambda 1 \Sigma |\theta_\square| + \lambda 2 \Sigma \theta_\square^2$

Key Points:

- L1 → feature selection
- L2 → stability
- Works best when features are correlated

---

5. When to Use What

- **Linear Regression:** Small, clean data
- **Ridge Regression:** Overfitting + all features matter
- **Lasso Regression:** Need feature selection
- **Elastic Net:** Correlated features + selection

---

6. Super Quick Differences

Linear → No penalty → Overfits
Ridge (L2) → Shrinks weights → No zeros

Lasso (L1) → Shrinks + deletes weights → Feature selection
Elastic Net → Mix of both → Balanced

✅ Relation Between λ (Lambda) and θ (Theta)

Theta (θ) = model parameters (weights)

Lambda (λ) = regularization strength

Lambda controls what happens to Theta.
They are directly connected through the **regularization term** in the cost function.

🔹 How λ Affects θ (Relationship)

1️⃣ If λ = 0

No penalty → θ values can become large.
Model may overfit.

2️⃣ If λ increases

Penalty increases → θ values shrink (become smaller).
Model becomes simpler → less overfitting.

3️⃣ If λ → very large

θ values shrink very close to zero.
Model may underfit.

🔥 Key Relationship (Mathematically)

In Ridge Regression:

J(θ)=MSE+λ∑θ2J(\theta) = MSE + \lambda \sum \theta^2J(θ)=MSE+λ∑θ2

- Here λ multiplies the θ^2 term.
- So **larger λ** → **stronger penalty** → **θ becomes smaller**.

In Lasso Regression:

J(θ)=MSE+λ∑∣θ∣J(\theta) = MSE + \lambda \sum |\theta|J(θ)=MSE+λ∑∣θ∣

- Here λ controls how many θ become exactly **0**.

🎯 Final One-line Relation (Viva Answer)

👉 **Lambda controls the size and magnitude of theta; higher lambda forces theta to shrink, while lower lambda allows theta to grow.**

⭐ FINAL NOTES: Types of Cross Validation (BEST VERSION)

(Definition + Concept + Use + Example)

1️⃣ Hold-Out Validation

✔️ Definition

Split the dataset into one **training set** and one **test set** (ex: 80% train, 20% test).

✔️ Concept

Model trains on training set → performance checked on test set.

✔️ Why Needed

Quick evaluation, but **accuracy depends heavily on random split**, may be unstable.

✔️ Where Used

Basic ML tasks, when dataset is large.

✔️ Example

Dataset = 1000
 Train = 800
 Test = 200

Train → predict → test accuracy.

2️⃣ Leave-One-Out Cross Validation (LOOCV)

✔️ Definition

Each sample becomes the test set once.
Train on **N–1 samples**, test on **1 sample**.

✔️ Concept

Repeat this for all samples → average accuracy.

✔️ Why Needed

- Maximum training data every time
- Very good for **small datasets**

✔️ Where Used

Medical datasets, research datasets with < 500 samples.

✔️ Example

Dataset size = 5
Experiments = 5
Exp1: Test = sample1
Exp2: Test = sample2
…
Exp5: Test = sample5

Final score = average of 5 accuracies.

3️⃣ Leave-P-Out Cross Validation (LPOCV)

✔️ Definition

Leave **P samples** out for testing, train on the rest.

✔️ Concept

All possible combinations of P test samples are checked.

✔️ Why Needed

Very high accuracy but **extremely expensive** → rarely used.

✔️ Where Used

Research, not used in production.

✔️ Example

Data = 20 samples
P = 5
Test = any 5 samples
Train = remaining 15
Repeat for all sample combinations.

4️⃣ K-Fold Cross Validation

✔️ Definition

Divide data into **K equal folds**.
Each fold becomes the test set once.

✔️ Concept

K experiments:

- Train on K−1 folds
- Test on 1 fold
  Final score = mean accuracy of K folds.

✔️ Why Needed

- More stable than hold-out
- Eliminates random split bias
- Best for medium-size datasets

✔️ Where Used

Almost all ML tasks (Regression + Classification).

✔️ Example

n = 500, K = 5
Each fold = 100 samples
Exp1: Test = Fold1
Exp2: Test = Fold2
…
Exp5: Test = Fold5

Average accuracy = final model score.

5️⃣ Stratified K-Fold Cross Validation

✔️ Definition

Same as K-fold, but **maintains class ratios** in every fold.

✔️ Concept

If dataset has 60% class 0 and 40% class 1 →
each fold keeps the same 60/40 ratio.

✔️ Why Needed

Prevents model from training on **imbalanced class distribution**.

✔️ Where Used

Classification tasks:

- Fraud detection
- Cancer detection
- Sentiment analysis

✔️ Example

Data: 100 samples
Class: 60 zeros, 40 ones
K = 5 → each fold:

- 12 class 0
- 8 class 1

6️⃣ Time-Series Cross Validation (Rolling/Forward CV)

✔️ Definition

Split data based on **time order**
Train on past → test on future.

✔️ Concept

Cannot shuffle time-based data.
You progressively expand training period.

✔️ Why Needed

Keeps **temporal order** → avoids data leakage.

✔️ Where Used

- Stock prices
- Weather forecasting
- Sales prediction
- Sensor readings

✔️ Example

Data = Jan → Dec
Exp1: Train = Jan–Mar, Test = Apr
Exp2: Train = Jan–Apr, Test = May
Exp3: Train = Jan–May, Test = Jun

⭐ SHORT EXAM-VIVA SUMMARY

Type	Definition	Best Use
Hold-Out	One train-test split	Large datasets
LOOCV	Leave 1 sample out	Small data
LPOCV	Leave P out	Rare, research
K-Fold	K splits, rotate test	General ML
Stratified K-Fold	K-Fold + class ratio preserved	Imbalanced classification
Time-Series CV	Train on past, test on future	Time-dependent Problem

Apple

⭐ Now: What Does `LassoCV()` Do in sklearn?

✔️ Definition

`LassoCV()` performs **Lasso Regression with automatic lambda (α) selection using Cross Validation**.

✔️ Concept

- Lasso requires choosing α (regularization strength)
- Instead of you choosing α manually
- `LassoCV()` tries many values of α
- Uses **K-Fold Cross Validation**
- Picks the **best α** that gives lowest error

✔️ So LassoCV gives you:

✔️ Best λ value
✔️ Best model
✔️ Automatically selected features
✔️ No overfitting
✔️ High accuracy

✔️ Step 1 — LassoCV creates a grid of alpha values

Example:

```
[0.001, 0.01, 0.1, 1, 10, 100]
```

✔️ Step 2 — Performs K-Fold Cross Validation

Default is **cv=5**, so data is split into 5 folds.

✔️ Step 3 — For each alpha

LassoCV:

1. Trains model on Fold1–Fold4
2. Tests on Fold5
3. Computes validation error
4. Repeats 5 times
5. Computes average error

✔️ Step 4 — Compare validation errors

Alpha with **lowest average error** is the winner.

✔️ Step 5 — Train final Lasso model on full dataset

Using the best alpha.

✔️ Step 6 — Save alpha in `alpha_`

That is the optimal regularization strength.

⭐ Diagram Explanation (Very simple)

For alpha = 0.01 → error = 0.5
For alpha = 0.1 → error = 0.3
For alpha = 1 → error = 0.8

Lowest error = **0.3** → best alpha = **0.1**

⭐ After Finding Alpha, What Does the Model Do?

Once `LassoCV()` discovers the best alpha, it does the following:

✅ 1. It Trains the Final Lasso Model Using That Alpha

The model re-trains on the entire training dataset with the selected alpha.

Why this matters:

- Before finding alpha → model was only trained on folds (partial data)
- After selecting alpha → it trains on full training data
- This gives the best generalization

✅ 2. It Learns the Final Coefficients (θ values)

Using the chosen alpha, the model decides:

- Which coefficients shrink
- Which coefficients become 0 (feature removed)
- Which coefficients stay large (important features)

So it creates the final version of the regression equation.

Example:

If alpha is high: many coefficients = 0
If alpha is small: more coefficients survive

✅ 3. It Becomes Ready For Prediction

After `.fit()` completes:

```
lassocv.predict(X_test)
```

Now prediction is possible because:

- **The best alpha is fixed**
- **The model is fully trained**
- **Coefficients are finalized**

⭐ Summary (Perfect Interview Answer)

👉 After finding alpha, LassoCV retrains the model on the full dataset using that alpha, computes the final coefficients, performs feature selection (shrinks some to zero), and becomes ready to make predictions.

Your three lines of code do the following:

✔️ Import RidgeCV

→ gives you a model that supports automatic alpha selection.

✔️ Initialize RidgeCV with 5-fold cross validation

→ tells sklearn to do 5-validation splits for each alpha.

✔️ Fit the model (MOST WORK DONE HERE)

During `.fit`:

1. **RidgeCV picks a list of alpha values**

2. **For each alpha:**
    - **trains 5 times**
    - **validates 5 times**
    - **computes 5 errors**
3. **Computes the average error**
4. **Chooses best alpha**
5. **Trains final Ridge model using that alpha**
6. **Stores alpha, coefficients, intercept**
7. **Model becomes ready to predict**

⭐ One-Line Viva Answer

👉 **RidgeCV finds the best alpha using K-fold CV, then trains the final Ridge model on full data using that alpha.**

📘 Combined Notes: RidgeCV vs LassoCV vs ElasticNetCV

⭐ 1. RidgeCV (Ridge Regression with Cross-Validation)

✔️ Definition

RidgeCV automatically finds the **best alpha** (L2 penalty strength) using K-fold cross validation.

✔️ What it does behind the scenes

1. Creates a list of alpha values
2. Splits data into K folds
3. For each alpha:
   - Train on K−1 folds
   - Validate on 1 fold
   - Compute validation MSE
4. Average the errors
5. Picks alpha with minimum average error
6. Trains final Ridge model on full training data

✔️ RidgeCV outputs

- `ridgecv.alpha_` → best L2 penalty

- `ridgecv.coef_` → final weights

✔️ When RidgeCV is used

- When features are correlated
- To reduce model variance
- When you DO NOT want coefficients to become 0
- Regression tasks with multicollinearity

⭐ Ridge strength

- Ridge shrinks coefficients but **never makes them zero**.

⭐ 2. LassoCV (Lasso Regression with Cross-Validation)

✔️ Definition

LassoCV automatically finds the **best alpha** (L1 penalty strength) using K-fold cross validation.

✔️ What it does internally

1. Creates many alphas
2. Runs K-fold CV for each alpha
3. Computes validation error
4. Selects alpha that gives lowest error
5. Fits final Lasso model on full data

✔️ Special property

Lasso **sets some coefficients exactly to ZERO** → feature selection.

✔️ LassoCV outputs

- `lassocv.alpha_` → best L1 penalty
- `lassocv.coef_` → many coefficients = 0
- `lassocv.mse_path_` → all CV errors

✔️ When LassoCV is used

- When dataset has **many irrelevant features**
- When you want **automatic feature selection**
- When model must be sparse and simple

⭐ 3. ElasticNetCV (ElasticNet + Cross Validation)

✔️ Definition

ElasticNetCV automatically finds:

- **best alpha** (penalty strength)
- **best l1_ratio** (how much L1 vs L2)

using K-fold CV.

✔️ What it does internally

1. Creates multiple alpha values

2. Creates multiple l1_ratio values
3. For every combination → performs K-fold CV
4. Computes MSE
5. Picks the pair with lowest validation error
6. Fits final ElasticNet model

✔️ ElasticNetCV outputs

- `elasticcv.alpha_` → best alpha
- `elasticcv.l1_ratio_` → best mix of Lasso + Ridge
- `elasticcv.coef_` → some coefficients shrink to 0

✔️ When ElasticNetCV is used

- When dataset has many features
- When features are correlated
- When you want some feature selection + some shrinkage
- Best for **high-dimensional datasets**

⭐ 4. Key Differences (VERY IMPORTANT TABLE)

Feature	RidgeCV	LassoCV	ElasticNetCV
Penalty type	L2	L1	L1 + L2
Removes features (coef=0)?	❌ No	✔️ Yes	✔️ Sometimes

Solves multicollinearity	✔️ Yes	❌ No	✔️ Yes
Handles many irrelevant features	❌ No	✔️ Yes	✔️ Yes
Finds best alpha	✔️ Yes	✔️ Yes	✔️ Yes
Finds best l1_ratio	❌ No	❌ No	✔️ Yes
Good for high-dimensional data	Medium	High	Highest
Output model complexity	Higher	Lowest	Balanced

⭐ 5. Final Viva Answers

✔️ RidgeCV

"RidgeCV uses K-Fold cross-validation to find the best L2 penalty and prevents overfitting by shrinking coefficients."

✔️ LassoCV

"LassoCV uses cross-validation to find the best L1 penalty, and it performs feature selection by making some coefficients zero."

✔️ ElasticNetCV

"ElasticNetCV selects both the best alpha and best l1_ratio using cross-validation, giving a balanced L1+L2 model suitable for correlated and high-dimensional datasets."