Linear Regression Roadmap (Beginner to Advanced)

Phase 1: Basics

Goal: Understand what linear regression is and when to use it. - **Concepts:** - Predict continuous values (e.g., house price, salary) - Simple Linear Regression: one feature → y = mx + b - Multiple Linear Regression: multiple features → y = b0 + b1x1 + b2x2 + ... - Features (X) and Target (y) - **Metrics:** - Mean Squared Error (MSE) - Mean Absolute Error (MAE) - R^2 Score - **Practice:** - Predict salary from years of experience - Predict house price from size

---

Phase 2: Mathematical Understanding

Goal: Learn how Linear Regression works under the hood. - **Formulas:** - m = Σ(xi - x̄)(yi - ȳ) / Σ(xi - x̄)^2 - b = ȳ - m * x̄ - **Cost Function:** J(m,b) = (1/n) Σ(yi - (mxi + b))^2 - **Gradient Descent:** iterative way to find best m and b - m := m - α ∂J/∂m, b := b - α ∂J/∂b - **Practice:** - Implement simple linear regression from scratch - Plot regression line vs actual points

---

Phase 3: Python Implementation

Goal: Learn to use libraries to build regression models quickly. - **Scikit-learn:**

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

- **Metrics in scikit-learn:** mean_squared_error, r2_score - **Practice:** - Use multiple features (house size, bedrooms, location) - Compare scratch implementation vs scikit-learn

---

Phase 4: Model Evaluation & Improvement

Goal: Learn how to test and improve your model. - **Techniques:** - Train/Test split - Cross-validation - Feature scaling (normalization/standardization) - Detect multicollinearity - Check residual plots for patterns or outliers - **Practice:** - Evaluate model with R^2, MSE, MAE - Try different feature combinations

---

Phase 5: Advanced Topics

Goal: Handle complex data and improve model performance. - **Regularization:** - Ridge Regression (L2) - Lasso Regression (L1) - ElasticNet (combined) - **Polynomial Regression:** fit non-linear trends

```
from sklearn.preprocessing import PolynomialFeatures
poly = PolynomialFeatures(degree=2)
X_poly = poly.fit_transform(X)
```

- **Practice:** - Predict complex trends using polynomial features - Apply regularization to avoid overfitting

---

Summary Table

Phase	Topics	Practice
1	Basics, simple vs multiple regression	Salary, house price prediction
2	Formulas, gradient descent	Implement from scratch
3	Scikit-learn implementation	Multiple features datasets
4	Evaluation, cross-validation, residuals	Train/test split, error analysis
5	Regularization, polynomial regression	Ridge/Lasso, complex trends