⚠️ Limitations of Ordinary Linear Regression

1. Multicollinearity

- Multicollinearity occurs when two or more independent variables in a regression model are highly correlated with each other.
- When independent variables are **highly correlated**, it becomes hard to estimate the coefficients accurately.
- Leads to **unstable** and **unreliable** models.
- Multicollinearity denotes when independent variables in a linear regression equation are correlated.

📏 Example: Height in Inches vs. Height in Meters

Person	Height (inches) x_1	Height (meters) x_2	Weight (kg) y
A	60	1.524	45
B	65	1.651	55
C	70	1.778	65
D	72	1.829	70
E	75	1.905	75

🔁 Relationship

$$x_2 = x_1 \times 0.0254$$

This means **Height in meters is linearly dependent** on Height in inches → **perfect multicollinearity**.

🧠 Why This Is a Problem in Regression

Suppose we try to model:

$$y = \beta_0 + \beta_1 \cdot \text{Height\_inches} + \beta_2 \cdot \text{Height\_meters} + \epsilon$$

The model will **fail to assign unique values** to β_1 and β_2, since both inputs contain the **same information**.

🧮 Correlation

$$\text{Corr}(x_1, x_2) \approx 1$$

Meaning they are **perfectly correlated**, causing:

- Inflated **standard errors**
- Unstable **coefficients**
- High **VIF values**

📌 2. High Variance When Number of Predictors is Large

- When the number of **predictors (features/input/independent variables)** is close to or exceeds the number of **observations (samples)**, the model becomes **unstable**.
- This results in a model that:
  - Fits the **training data too well**
  - Performs **poorly on test/unseen data**
- This is a classic case of **overfitting**, where the model captures noise rather than true patterns.

📌 3. Overfitting: Model Fits Noise Instead of Signal

- **Overfitting** happens when a model learns not only the **underlying patterns** in the data (signal), but also the **random fluctuations or errors** (noise).
- The model performs **very well on training data**, but **fails to generalize** to new or unseen data.

🔍 Why It Happens:

- Too many predictors/features
- Very flexible models
- Small training datasets
- High variance in the data

📉 Example:

- Imagine fitting a **complex curve** to just a few data points. It may pass through all the points (perfect accuracy), but perform poorly on future data.

📌 4. Poor Generalization to Unseen Data

- **Generalization** refers to how well a model performs on **new, unseen data** — not just the data it was trained on.
- **Ordinary Linear Regression** assumes:
  - A **linear relationship** between predictors and output
  - **Constant variance** of errors (homoscedasticity)
  - **Normally distributed** residuals
  - **Independence** of observations

❌ When these assumptions are violated:

- The model may perform well on training data but **poorly on test data**.
- This leads to **low predictive accuracy** in real-world applications.

💡 Example:

If the true relationship is **non-linear**, a linear model will **underfit**, missing key patterns — resulting in **poor generalization**.

Limitation	Explanation	Example
High Variance When Number of Predictors is Large	Model becomes unstable when there are too many predictors compared to data size.	Predicting house prices with 10 features but only 8 houses in the dataset.
Multicollinearity Causes Unstable Coefficient Estimates	Highly correlated predictors distort the influence of individual variables.	Predicting salary using both `years of experience` and `age`, which are strongly correlated.
Overfitting: Fits Noise Instead of Signal	Model learns random noise rather than actual patterns in data.	A model gives perfect predictions on training data but fails completely on test data.

Limitation	Explanation	Example
Poor Generalization to Unseen Data	Model doesn't perform well on new or unseen data.	Linear regression used to predict non-linear relationships like stock prices or weather trends.

📈 2. Polynomial Regression

✔️ Concept

- Polynomial Regression is an extension of **Linear Regression** that allows for **non-linear** relationships between the **independent variable(s)** and the **dependent variable**.
- It does so by including **powers (exponents)** of the predictors as additional features.
- Despite modeling non-linear relationships, it is still considered a **linear model** in terms of the coefficients.

🧠 Why Use It?

- When a scatterplot of the data shows a **curved trend** instead of a straight line.
- It helps capture **quadratic**, **cubic**, or even higher-order patterns in the data.

🧮 Equation

For one predictor x, a **degree-2 (quadratic)** polynomial regression model looks like:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \varepsilon$$

More generally, for a degree-d polynomial:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_d x^d + \varepsilon$$

Where:

- y: Dependent variable (target)
- x: Independent variable (predictor)
- $\beta_0, \beta_1, \ldots, \beta_d$: Coefficients
- ε: Error term

📌 When to Use Polynomial Regression

✔️ 1. Residual Plots Show Non-Random Patterns

- In a good linear model, residuals (errors) should be **randomly scattered** around zero.
- If the residual plot shows a **curve or systematic pattern**, it indicates that a **linear model is not appropriate**, and a **polynomial regression** may be needed.

✔️ 2. R^2 Improves Significantly with Higher-Order Terms

- Add polynomial terms (e.g., x^2, x^3) to the model and observe **R^2** (coefficient of determination).
- If **R^2 increases noticeably** with added terms, it suggests the model is better capturing the underlying **non-linear relationship**.
- But beware: A very high degree may lead to **overfitting**.

🧪 Example Workflow:

1. Fit a linear regression.
2. Plot the residuals.
    - If they curve → try a degree-2 polynomial.
3. Check **R^2**:
    - If R^2 increases from, say, **0.65 to 0.91**, polynomial terms are likely adding real value.

🔒 Regularization Techniques in Regression

Regularization helps prevent **overfitting** by adding a **penalty** to the model's complexity—specifically, to the **size of the coefficients**.

📘 1. Ridge Regression (L2 Regularization)

- **Penalty**: Sum of squared coefficients

$$\text{Loss} = \text{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2$$

- Shrinks all coefficients **toward zero** but **does not eliminate any**.
- **Use case**: When all predictors are important but you want to reduce model complexity.

📘 2. Lasso Regression (L1 Regularization)

- **Penalty**: Sum of absolute coefficients

$$\text{Loss} = \text{RSS} + \lambda \sum_{j=1}^{p} |\beta_j|$$

- Can shrink **some coefficients exactly to zero**, effectively performing **feature selection**.
- **Use case**: When you expect **only a few variables** to be truly important.

📘 3. Elastic Net Regression

- **Penalty**: Combination of L1 and L2

$$\text{Loss} = \text{RSS} + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2$$

- Balances the **feature selection ability** of Lasso with the **stability** of Ridge.
- **Use case**: When predictors are highly correlated or you need both **shrinkage** and **selection**.

🎛️ Tuning the Regularization Parameter λ

- λ controls the **strength of the penalty**:
  - **High** λ → More penalty → Coefficients are **shrunk more** → Model is **simpler**.
  - **Low** λ → Less penalty → Coefficients are closer to **OLS estimates**.
- Typically chosen using **cross-validation** (e.g., k-fold CV).

📊 Summary Table

Technique	Penalty Type	Effect on Coefficients	Feature Selection?
Ridge Regression	L2 (squared values)	Shrinks all coefficients, none set to 0	❌ No
Lasso Regression	L1 (absolute values)	Some coefficients shrunk to 0	✔️ Yes
Elastic Net	L1 + L2 (combined penalty)	Shrinks and selects variables	✔️ Yes