Chapter - V

Feature Engineering

Syllabus: Curse of Dimensionality, Feature Elimination Techniques - PCA, LDA, Feature Selection - Wrapper, Embedded Techniques, Concept of Multicollinearity

Feature Engineering is the process of **creating, transforming, or selecting variables (features)** to improve model performance and enhance insights. During EDA, feature engineering helps understand data better and prepare it for modeling.

5.1. Curse of Dimensionality

The curse of dimensionality is the exponential increase in data sparsity and computational complexity as the number of dimensions (features) grows, which negatively affects distance-based learning, generalization, and model performance.

5.2. Importance of the Curse of Dimensionality

1. **Helps Improve Model Performance**: Understanding it helps prevent models from becoming inaccurate due to sparsity and meaningless distances in high dimensions.

2. **Guides Feature Selection:** It highlights the need to remove irrelevant or redundant features to avoid overfitting.

3. **Necessary for Distance-Based Algorithms:** Algorithms like KNN, K-Means, SVM (RBF kernel) fail in high dimensions; knowing this helps choose the right modeling technique.

4. **Supports the Use of Dimensionality Reduction:** It justifies applying methods like PCA, t-SNE, UMAP, Autoencoders to reduce dimensions and improve efficiency.

5. **Reduces Computational Cost**: High dimensionality increases memory and processing time; awareness helps optimize resource usage.

6. **Ensures Better Generalization:** By reducing dimensions, we prevent models from memorizing noise, leading to improved performance on unseen data.

7. **Avoids Data Sparsity Issues:** Sparsity makes pattern detection difficult; recognizing it ensures better dataset preparation.

5.3.Feature Elimination Techniques:

- ✓ Feature Elimination Techniques are methods used to remove irrelevant, redundant, or unimportant features from a dataset to improve model performance, reduce complexity, and avoid overfitting.
- ✓ These techniques help in choosing the minimal but most important set of features.

5.4. PCA (Principal Component Analysis)

PCA is a statistical technique that converts correlated features into a set of uncorrelated components called principal components, arranged in order of decreasing variance.

5.4.1. Core Idea Behind PCA

PCA tries to find **new axes (directions)** in the dataset such that:

- Each new axis captures **maximum possible variance**
- New axes are **orthogonal (uncorrelated)**
- Data projected onto these axes retains **maximum information** with fewer dimensions

These new axes are called **Principal Components (PCs)**.

5.4.2. Why Variance Is Important?

Variance represents **information** or **spread** in data.

- High variance = meaningful information
- Low variance = noise or redundancy

So PCA tries to **maximize variance** to capture the most useful information.

5.4.3. Steps to Solve PCA Manually

Step 1: Collect and Organize the Data

- Arrange data points into a matrix form.
- If there are *n* samples and *d* features → matrix size = **n × d**.

Step 2: Standardize the Data (if required)

- Subtract the mean of each feature from every value.
- Purpose: to center the data around zero.
- Formula: $x' = x - \bar{x}$

Step 3: Form the Covariance Matrix

- Compute covariance between every pair of features.
- For *d* features → covariance matrix is **d × d**.

$$\text{Cov}(X) = \frac{1}{n-1}(X^T X)$$

Step 4: Calculate Eigenvalues and Eigenvectors

- Find eigenvalues (λ) from the covariance matrix.
- Find eigenvectors corresponding to each eigenvalue.
- Eigenvalues tell **how much variance** each principal component captures.

Step 5: Rank Eigenvalues

- Sort eigenvalues in **descending order**.
- The eigenvector with highest eigenvalue = **1st Principal Component (PC1)**.
- Next highest = **2nd Principal Component (PC2)**, and so on.

Step 6: Select Number of Principal Components

- Choose top *k* eigenvalues based on:
  - Cumulative variance rule (e.g., 95%)
  - Scree plot
  - Domain knowledge
- Form a **projection matrix** with corresponding eigenvectors.

Step 7: Transform Original Data

- Multiply the standardized data with the chosen eigenvector matrix.
- $Z=XW$

  where
  X = standardized data
  W = eigenvector matrix
  Z = transformed principal components

Step 8: Interpret the Results

- PC1, PC2, … represent new features capturing maximum variance.
- Use these new components for:
  - Visualization
  - Classification
  - Clustering
  - Regression
  - Noise reduction

PCA – Solved Example

Given Data:

Observation	X1	X2
1	2	0
2	0	2
3	3	1
4	4	3

Step 1: Standardize the Data (Mean-center)

Compute Means:

$$\bar{X}_1 = \frac{2+0+3+4}{4} = 2.25$$

$$\bar{X}_2 = \frac{0+2+1+3}{4} = 1.5$$

Subtract Means:

Obs	X1	X2	X1_centered	X2_centered
1	2	0	2 − 2.25 = −0.25	0 − 1.5 = −1.5
2	0	2	0 − 2.25 = −2.25	2 − 1.5 = 0.5
3	3	1	3 − 2.25 = 0.75	1 − 1.5 = −0.5
4	4	3	4 − 2.25 = 1.75	3 − 1.5 = 1.5

Step 2: Form Covariance Matrix

Compute variances:

$$\text{var}(X1) = \frac{1}{4-1}[(-0.25)^2 + (-2.25)^2 + (0.75)^2 + (1.75)^2]$$

$$= \frac{1}{3}(0.0625 + 5.0625 + 0.5625 + 3.0625)$$

$$= \frac{8.75}{3} = 2.9167$$

$$\text{var}(X2) = \frac{1}{3}(2.25 + 0.25 + 0.25 + 2.25) = \frac{5}{3} = 1.6667$$

Compute covariance:

$$\text{cov}(X1, X2) = \frac{1}{3}[(-0.25)(-1.5) + (-2.25)(0.5) + (0.75)(-0.5) + (1.75)(1.5)]$$

Calculate:

$$= \frac{1}{3}[0.375 - 1.125 - 0.375 + 2.625]$$

$$= \frac{1.5}{3} = 0.5$$

Covariance Matrix

$$\Sigma = \begin{bmatrix} 2.9167 & 0.5 \\ 0.5 & 1.6667 \end{bmatrix}$$

Step 3: Compute Eigenvalues & Eigenvectors

Solve:

$$\det(\Sigma - \lambda I) = 0$$

$$\begin{vmatrix} 2.9167 - \lambda & 0.5 \\ 0.5 & 1.6667 - \lambda \end{vmatrix} = 0$$

Characteristic equation:

$$(2.9167 - \lambda)(1.6667 - \lambda) - 0.25 = 0$$

Solve (values rounded):

Eigenvalues

$$\lambda_1 = 3.0$$

$$\lambda_2 = 1.5834$$

So PC1 explains more variance.

Step 4: Find Eigenvectors

For $\lambda_1 = 3.0$:

Solve:

$$(\Sigma - 3I)v = 0$$

$$\begin{bmatrix} -0.0833 & 0.5 \\ 0.5 & -1.3333 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 0$$

Solution (normalized):

$$v_1 = \begin{bmatrix} 0.316 \\ 0.949 \end{bmatrix}$$

For $\lambda_2 = 1.5834$:

$$v_2 = \begin{bmatrix} 0.949 \\ -0.316 \end{bmatrix}$$

Step 5: Principal Components

PC1 = 0.316·X1 + 0.949·X2

(captures highest variance)

PC2 = 0.949·X1 − 0.316·X2

(secondary direction of variation)

Step 6: Final PCA Transformation

To get transformed data (scores):

$$Z = X_{\text{centered}} \cdot V$$

Where

$V = [v_1\ v_2]$

After projection, PC1 and PC2 values for each observation are obtained.

- **Covariance matrix**

$$\begin{bmatrix} 2.9167 & 0.5 \\ 0.5 & 1.6667 \end{bmatrix}$$

- **Eigenvalues**

  PC1: **3.0**

  PC2: **1.5834**
- **Eigenvectors**

  PC1: $[0.316,\ 0.949]$

  PC2: $[0.949,\ -0.316]$
- **Principal Components**

$$PC1 = 0.316X_1 + 0.949X_2$$

$$PC2 = 0.949X_1 - 0.316X_2$$

4.5. Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is a supervised dimensionality reduction and classification technique used to project high-dimensional data onto a lower-dimensional space while maximizing the separation between multiple classes.

Unlike PCA (which is unsupervised and preserves maximum variance), LDA uses class labels and focuses on finding a feature subspace where different classes are as far apart as possible and data within the same class is as close as possible.

Core Idea of LDA

LDA finds a new axis (or multiple axes) such that:

- **Between-class variance is maximized**
  (Classes are well separated)
- **Within-class variance is minimized**
  (Points of same class are compact)

Thus, LDA increases class discriminability → making classification easier and more accurate.

Steps to solve LDA

Step 1: Prepare and Organize the Data

- Collect dataset with **features** and **class labels.**
- Separate the data class-wise.
- Example:
  - Class 1 → X_1
  - Class 2 → X_2
  - For *k* classes → $X_1, X_2, \ldots, X_k$

Step 2: Compute the Mean of Each Class

For each class i:

$$m_i = \frac{1}{n_i} \sum_{x \in X_i} x$$

- m_i = mean vector of class *i*
- n_i = number of samples in class *i*

Step 3: Compute the Overall Mean

$$m = \frac{1}{N} \sum_{i=1}^{k} n_i m_i$$

where

- m = mean of all samples
- N = total samples across all classes

Step 4: Compute the Within-Class Scatter Matrix S_w

$$S_w = \sum_{i=1}^{k} \sum_{x \in X_i} (x - m_i)(x - m_i)^T$$

- Measures **spread of data inside each class.**
- Should be **minimized** by LDA.

Step 5: Compute the Between-Class Scatter Matrix S_b

$$S_b = \sum_{i=1}^{k} n_i (m_i - m)(m_i - m)^T$$

- Measures **distance between class means.**
- Should be **maximized** by LDA.

Step 6: Compute the Projection Vector (Eigen Problem)

Solve the generalized eigenvalue equation:

$$S_w^{-1} S_b W = \lambda W$$

- W = eigenvectors
- λ = eigenvalues
- Largest eigenvalues correspond to **best discriminant directions.**

Step 7: Select Top Linear Discriminants

- Sort eigenvalues in descending order.
- Choose top **k − 1** discriminant vectors (because LDA gives at most *C − 1* dimensions for *C* classes).

Example:

- For 2 classes → 1 dimension
- For 3 classes → up to 2 dimensions

Step 8: Transform the Data

Project data onto new LDA components:

$$Z = XW$$

- Z = transformed (reduced) data
- Used for visualization or classification

Solved Problem on LDA:

Problem Statement

Perform LDA for the following two-class dataset:

Class 1:

(2, 3), (3, 4), (4, 5)

Class 2:

(5, 3), (6, 2), (7, 3)

Find the LDA projection direction vector **w**.

Step 1: Compute Mean of Each Class

Class 1 Mean

$$m_1 = \left(\frac{2+3+4}{3}, \frac{3+4+5}{3}\right) = (3, 4)$$

Class 2 Mean

$$m_2 = \left(\frac{5+6+7}{3}, \frac{3+2+3}{3}\right) = (6,\ 2.67)$$

Step 2: Compute Within-Class Scatter Matrix S_w

Class 1 Scatter

For point (2, 3): (-1, -1)

Outer product:

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

For (3, 4): (0, 0)

Outer product:

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

For (4, 5): (1, 1)

Outer product:

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

So,

$$S_1 = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$$

Class 2 Scatter

Mean = (6, 2.67)

For (5,3): (-1, 0.33)

Outer product:

$$\begin{bmatrix} 1 & -0.33 \\ -0.33 & 0.11 \end{bmatrix}$$

For (6,2): (0, -0.67)

Outer product:

$$\begin{bmatrix} 0 & 0 \\ 0 & 0.45 \end{bmatrix}$$

For (7,3): (1, 0.33)

Outer product:

$$\begin{bmatrix} 1 & 0.33 \\ 0.33 & 0.11 \end{bmatrix}$$

Add them:

$$S_2 = \begin{bmatrix} 2 & 0 \\ 0 & 0.67 \end{bmatrix}$$

Total Within-Class Scatter

$$S_w = S_1 + S_2 = \begin{bmatrix} 4 & 2 \\ 2 & 2.67 \end{bmatrix}$$

Step 3: Compute Mean Difference Vector

$$m_1 - m_2 = (3 - 6,\ 4 - 2.67) = (-3,\ 1.33)$$

Step 4: Compute LDA Projection Vector

Formula for 2-class LDA:

$$w = S_w^{-1}(m_1 - m_2)$$

Invert S_w

$$S_w = \begin{bmatrix} 4 & 2 \\ 2 & 2.67 \end{bmatrix}$$

Determinant:

$$|S_w| = (4)(2.67) - (2)(2) = 10.68 - 4 = 6.68$$

Inverse:

$$S_w^{-1} = \frac{1}{6.68} \begin{bmatrix} 2.67 & -2 \\ -2 & 4 \end{bmatrix}$$

Multiply with ($m_1 - m_2$)

$$w = \frac{1}{6.68} \begin{bmatrix} 2.67 & -2 \\ -2 & 4 \end{bmatrix} \begin{bmatrix} -3 \\ 1.33 \end{bmatrix}$$

Compute:

First component:

$$(2.67)(-3) + (-2)(1.33) = -8.01 - 2.66 = -10.67$$

Second component:

$$(-2)(-3) + (4)(1.33) = 6 + 5.32 = 11.32$$

So,

$$w = \frac{1}{6.68} \begin{bmatrix} -10.67 \\ 11.32 \end{bmatrix}$$

$$w = \begin{bmatrix} -1.596 \\ 1.695 \end{bmatrix}$$

Final LDA Projection Direction

$$w = [-1.60,\ 1.70]^T$$

This is the direction that best separates the two classes.

Feature Selection

Feature Selection is the process of selecting the most relevant and significant features (variables) from a dataset to improve model performance, reduce overfitting, and decrease computational cost. It removes irrelevant, redundant, and noisy features while keeping the most informative ones.

There are three major categories of Feature Selection:

1. **Filter Methods**
2. **Wrapper Methods**
3. **Embedded Methods**

1. Filter Techniques

Filter methods select features **based on statistical measures**, independent of any machine learning model.

How Filter Methods Work

- Evaluate each feature individually for its relevance to the target.
- Rank features using a score.
- Select top-ranked features.

Common Filter Methods

1. **Correlation-Based Selection**
    - Remove features highly correlated with each other.
    - Keep features that are strongly correlated with the target but weakly with other features.
2. **Chi-Square Test** (for categorical features)
    - Measures dependency between a feature and the target.
    - Higher Chi-square → more important feature.
3. **ANOVA / F-Test** (for numerical features)
    - Measures variance between groups.
    - Features with low F-value can be removed.
4. **Mutual Information**
    - Measures how much information a feature shares with the target.
    - Features with low mutual information are discarded.

Advantages

- Fast and computationally cheap.
- Works well with high-dimensional datasets.
- Avoids overfitting since it does not use the learning algorithm.

Disadvantages

- Does not consider interaction between features.
- May not select the optimal subset for a specific model.

Example:

1. Filter Technique Example

Scenario: Predict whether a student passes or fails based on:

- Hours studied (numeric)
- Number of lectures attended (numeric)
- Student ID (numeric, just an identifier)

Step: Compute correlation with the target (Pass/Fail).

Feature	Correlation with Pass/Fail
Hours studied	0.85
Lectures attended	0.80
Student ID	0.02

Action: Remove **Student ID** because correlation is near 0.

Filter method selects features based purely on statistical measure.

2. Wrapper Techniques

Wrapper methods evaluate subsets of features by **actually training a model** and selecting the subset that gives the best performance.

How Wrapper Methods Work

- Choose a model (e.g., Logistic Regression, KNN, Decision Tree).
- Try different combinations of features.
- Evaluate model accuracy for each combination.
- Select the best subset.

Examples of Wrapper Techniques

1. **Forward Selection**
   Start with no features → add features one by one.
2. **Backward Elimination**
   Start with all features → remove the least important feature each step.

3. **Recursive Feature Elimination (RFE)**
   Uses model coefficients (or importance) to recursively remove weakest features.

Advantages

- Usually gives better performance because it considers model behavior.
- Works well for small-to-medium feature sets.

Disadvantages

- Very slow and computationally expensive (multiple model training cycles).
- May cause overfitting in small datasets.

Example:

2. Wrapper Technique Example

Scenario: Predict house price using features: Size, Bedrooms, Age, Distance to city.

Method: Recursive Feature Elimination (RFE) with Linear Regression.

Steps:

1. Train model with all features → check performance (R^2).
2. Remove least important feature (e.g., Age) → retrain → performance improves.
3. Remove next least important (e.g., Distance to city) → retrain → performance drops.

Result: Keep **Size** and **Bedrooms** → gives best model performance.

Wrapper method uses the model to select features.

3. Embedded Techniques

Embedded methods perform feature selection **during the model training process**, not before or after.

How Embedded Methods Work

- Model itself identifies important and unimportant features.
- Feature selection is built into the training algorithm.

Examples of Embedded Techniques

1. **Lasso Regression (L1 Regularization)**
   Shrinks some coefficients to zero → automatically removes features.

2. **Ridge Regression (L2 Regularization)**
   Reduces coefficient magnitude but does not remove features completely.
3. **Elastic Net (L1 + L2 combination)**
   Performs feature shrinkage + selection simultaneously.
4. **Decision Trees / Random Forests Feature Importance**
   Trees automatically rank features by importance (Gini, Information Gain).

Advantages

- Faster than wrapper methods.
- Avoids overfitting better because regularization is built-in.
- Works well for high-dimensional data.

Disadvantages

- Depends on the chosen learning algorithm.
- Not as flexible as wrapper methods.

Example:

3. Embedded Technique Example

Scenario: Predict loan approval based on Income, Credit Score, Existing Loans.

Method: Lasso Regression (L1 regularization)

Steps:

- Train Lasso model with all features.
- Lasso shrinks coefficients:
  - Income → 0.5
  - Credit Score → 0.3
  - Existing Loans → 0 (coefficient becomes zero)

Result: Feature **Existing Loans** is automatically eliminated.

Embedded method does feature selection during model training.

Concept of Multicollinearity

Definition

Multicollinearity occurs when **two or more independent variables (features) are highly correlated** with each other in a dataset.

In other words:
One feature can be predicted from another with high accuracy.

Why It Is a Problem

- Coefficients become unstable and unreliable.
- Model may show high accuracy but individual feature importance becomes meaningless.
- Inflates standard errors → wrong interpretations in regression.

Symptoms of Multicollinearity

- High correlation values (> 0.8) between features.
- Large variance in model coefficients.
- High VIF (Variance Inflation Factor) values.
  VIF > 5 (or > 10) indicates multicollinearity.

Causes

- Derived variables (e.g., Height in cm and Height in m).
- Dummy variables created incorrectly.
- Including different transformations of the same feature.

How to Handle It

- Remove one of the correlated features.
- Use PCA or LDA for dimensionality reduction.
- Use Ridge Regression (helps reduce effect of multicollinearity).
- Combine correlated features.

Review Questions:

1. Define feature engineering.
2. What is curse of dimensionality? Explain importance of curse of dimensionality in detail.
3. Describe core idea behind PCA and explain why variance is important.
4. Explain the steps required to solve PCA manually.
5. Solved example of on PCA.
6. Explain LDA along with its core idea.
7. Describe the steps to follow to solve LDA.
8. Solved problems on LDA.
9. What is feature selection? List the categories in feature selection.
10. Describe filters techniques in detail. Also list common filter methods
11. Explain wrapper technique along with types and example.
12. Explain embedded technique along with types and example.
13. Describe the Concept of Multicollinearity