ML Unit-2 Notes by OMG

1. Explain Local Binary Pattern (LBP) feature extraction technique with suitable example. (CO2)

The **Local Binary Pattern (LBP)** is a powerful and simple feature extraction technique used primarily for texture classification in computer vision. It works by describing the local texture patterns around each pixel.

Process:

1. **Neighborhood Selection:** For a given pixel in an image (the center pixel), a neighborhood of surrounding pixels is selected (typically a 3x3 grid, meaning 8 neighbors).
2. **Thresholding:** The intensity of the center pixel is used as a threshold. Each neighbor's intensity value is compared to this threshold.
3. **Binary Code Generation:** A binary value is assigned to each neighbor. If the neighbor's intensity is greater than or equal to the center pixel's intensity, it gets a '1'; otherwise, it gets a '0'.
4. **LBP Code (Decimal Value):** The binary values are read in a sequence (usually clockwise, starting from the top-left) to form an 8-bit binary number. This binary number is then converted to its decimal equivalent, which is the LBP code for the center pixel. This value ranges from 0 to 255.
5. **Histogram Formation:** This process is repeated for every pixel in the image. The resulting LBP codes are then used to build a histogram, which represents the texture features of the entire image. This histogram can be used as a feature vector for machine learning models.

Example:
Let's use the pixel neighborhood from question 7:

10	12	18
7	**6**	**2**
1	**4**	**9**

1. The **center pixel** has a value of **6**. This is our threshold.
2. We compare each neighbor to 6, starting from the top-left and moving clockwise:
    - 10 ≥ 6 → **1**

   - 12 ≥ 6 → **1**
   - 18 ≥ 6 → **1**
   - 2 < 6 → **0**
   - 9 ≥ 6 → **1**
   - 4 < 6 → **0**
   - 1 < 6 → **0**
   - 7 ≥ 6 → **1**
3. The resulting binary sequence is **11101001**.
4. Converting this binary number to decimal:

   (1×27)+(1×26)+(1×25)+(0×24)+(1×23)+(0×22)+(0×21)+(1×20)
   =128+64+32+0+8+0+0+1=233

The LBP code for the central pixel (6) is **233**.

---

2. Explain the type of wrapper methods for feature selection. (CO2)

Wrapper methods are a type of feature selection technique that uses a specific machine learning model to evaluate the performance of different subsets of features. The core idea is to "wrap" the feature selection process around the model training process. The "best" feature subset is the one that results in the highest performance for that particular model.

These methods are generally more computationally expensive than filter methods but often lead to better model performance because they select features optimized for the chosen algorithm.

The main types of wrapper methods are:

1. **Forward Selection:**
   - **Process:** This is an iterative method that starts with an empty set of features. In each iteration, it adds the single feature from the remaining set that provides the most significant improvement to the model's performance. The process continues until adding new features no longer improves the model's performance beyond a certain threshold.
   - **Analogy:** It's like building a team from scratch, adding one player at a time based on who contributes the most to the team's overall score in a practice match.
2. **Backward Elimination:**
   - **Process:** This method works in the opposite direction of forward selection. It starts with the full set of features. In each iteration, it removes the single feature whose removal leads to the best improvement (or least degradation) in model performance. This process is repeated until no further improvement can be achieved by removing

features.
    - **Analogy:** It's like trimming a roster by removing the weakest player one at a time until you are left with the optimal team.
3. **Recursive Feature Elimination (RFE):**
    - **Process:** RFE is a more robust greedy optimization algorithm. It starts by training the model on the entire set of features and calculates an importance score for each feature (e.g., using model coefficients or feature importances). The least important feature is then removed. The process is repeated recursively on the smaller feature set until the desired number of features is reached.
    - **Analogy:** It's like ranking all players based on their overall contribution after a season and repeatedly cutting the lowest-ranked player until you reach your target team size.

---

3. How to handle missing values in a dataset that will be used for training the ML model? (CO2)

Handling missing values is a critical step in data preprocessing because most machine learning algorithms cannot function with null or NaN values in the dataset. The choice of method depends on the nature and amount of missing data.

Here are the common ways to handle missing values:

Deletion Methods

1. **Listwise Deletion (Row Deletion):**
    - **What it is:** The entire row (or record) containing a missing value is removed from the dataset.
    - **When to use:** This is the simplest method. It's effective when the number of missing values is very small compared to the total dataset size, so you don't lose significant information.
    - **Drawback:** Can lead to a significant loss of data if missing values are widespread, potentially introducing bias.
2. **Pairwise Deletion (Column Deletion):**
    - **What it is:** The entire column (or feature) is removed if it contains a large number of missing values.
    - **When to use:** Best used when a specific feature has a high percentage of missing data (e.g., > 60-70%) and is not critical for the analysis.

- **Drawback:** You lose all information associated with that feature.

Imputation Methods

Imputation is the process of replacing missing data with substituted values.

1. **Mean/Median/Mode Imputation:**
   - **What it is:** Missing values in a numerical column are replaced with the **mean** or **median** of the non-missing values in that column. For a categorical column, they are replaced with the **mode** (most frequent value).
   - **When to use:** A quick and easy method. Median is preferred over mean when the data has outliers.
   - **Drawback:** Can distort the variance and relationships between variables.
2. **End of Distribution Imputation:**
   - **What it is:** Missing values are replaced by a value at the far end of the feature's distribution, such as (mean + 3 * standard deviation) or (mean - 3 * standard deviation).
   - **When to use:** Useful when the fact that a value is missing is itself informative. It helps the model recognize that the value was originally missing.
3. **Arbitrary Value Imputation:**
   - **What it is:** All missing values are replaced with an arbitrary constant, like 0, 999, or -1.
   - **When to use:** Similar to end-of-distribution imputation, it's used when the missingness is considered a meaningful feature.
4. **Model-Based Imputation (e.g., Regression):**
   - **What it is:** A machine learning model is used to predict the missing values. The feature with missing values is treated as the target variable, and the other features are used as predictors. For example, a regression model can predict a missing numerical value.
   - **When to use:** More sophisticated and often more accurate than simple imputation, as it considers the relationships between variables.
   - **Drawback:** Computationally more expensive.

---

4. Explain the process of Principal Component Analysis (PCA) in brief. (CO2)

Principal Component Analysis (PCA) is a widely used unsupervised dimensionality reduction technique. Its main goal is to reduce the number of variables (features) in a dataset while preserving as much of the original information (variance) as possible. It achieves this by transforming the original variables into a new set of uncorrelated variables called **principal components**.

The process of PCA can be summarized in the following steps:

1. **Standardize the Data:**
   - The first step is to standardize the range of the continuous initial variables so that each one of them contributes equally to the analysis. This is done by subtracting the mean and dividing by the standard deviation for each feature (Z-score normalization).
2. **Compute the Covariance Matrix:**
   - PCA aims to identify the relationships between variables. The covariance matrix expresses these correlations. For a dataset with *n* features, this will be an *n x n* matrix where the diagonal elements represent the variance of each feature and the off-diagonal elements represent the covariance between pairs of features.
3. **Calculate Eigenvectors and Eigenvalues:**
   - The eigenvectors and eigenvalues are computed from the covariance matrix.
     - **Eigenvectors** represent the directions of the new feature space (the principal components). They are orthogonal to each other, meaning they are uncorrelated.
     - **Eigenvalues** represent the magnitude or importance of the corresponding eigenvectors. A higher eigenvalue indicates that its corresponding eigenvector explains more of the variance in the data.
4. **Select Principal Components:**
   - The eigenvectors are sorted in descending order based on their corresponding eigenvalues. The eigenvector with the highest eigenvalue is the first principal component (PC1), which captures the most variance in the data. The second highest is PC2, and so on.
   - You decide how many principal components to keep based on the cumulative explained variance. For example, you might decide to keep enough components to explain 95% of the total variance.
5. **Transform the Data:**
   - The final step is to transform the original standardized data onto the new subspace defined by the selected principal components. This is done by taking the dot product of the original data and the selected eigenvectors. The result is a new dataset with fewer dimensions.

---

5. Consider a vector x = (23, 29, 52, 31, 45, 19, 18, 27). Apply feature scaling and find out min-max scaled values as well as z-score values. (CO2)

Given vector: x = (23, 29, 52, 31, 45, 19, 18, 27)

a) Min-Max Scaling

This technique scales the data to a fixed range, usually [0, 1].
The formula is:

Xscaled=Xmax−XminX−Xmin

1. Find the minimum and maximum values:
   - Xmin=18
   - Xmax=52
2. Calculate the range:
   - Range=Xmax−Xmin=52−18=34
3. Apply the formula to each element:
   - 23: (23−18)/34=5/34≈0.147
   - 29: (29−18)/34=11/34≈0.324
   - 52: (52−18)/34=34/34=1.0
   - 31: (31−18)/34=13/34≈0.382
   - 45: (45−18)/34=27/34≈0.794
   - 19: (19−18)/34=1/34≈0.029
   - 18: (18−18)/34=0/34=0.0
   - 27: (27−18)/34=9/34≈0.265

Min-Max Scaled Values: (0.147, 0.324, 1.0, 0.382, 0.794, 0.029, 0.0, 0.265)

b) Z-Score Normalization (Standardization)

This technique rescales data to have a mean (μ) of 0 and a standard deviation (σ) of 1.
The formula is:

Xscaled=σX−μ

1. Calculate the mean (μ):
   - μ=(23+29+52+31+45+19+18+27)/8=244/8=30.5
2. Calculate the standard deviation (σ):
   - Variance σ2=NΣ(Xi−μ)2

   - $\sigma2=[(-7.5)2+(-1.5)2+(21.5)2+(0.5)2+(14.5)2+(-11.5)2+(-12.5)2+(-3.5)2]/8$
   - $\sigma2=[56.25+2.25+462.25+0.25+210.25+132.25+156.25+12.25]/8=1032/8=129$
   - $\sigma=129\approx11.358$
3. Apply the formula to each element:
   - 23: $(23-30.5)/11.358=-7.5/11.358\approx$-0.660
   - 29: $(29-30.5)/11.358=-1.5/11.358\approx$-0.132
   - 52: $(52-30.5)/11.358=21.5/11.358\approx1.893$
   - 31: $(31-30.5)/11.358=0.5/11.358\approx0.044$
   - 45: $(45-30.5)/11.358=14.5/11.358\approx1.277$
   - 19: $(19-30.5)/11.358=-11.5/11.358\approx$-1.012
   - 18: $(18-30.5)/11.358=-12.5/11.358\approx$-1.091
   - 27: $(27-30.5)/11.358=-3.5/11.358\approx$-0.308

Z-Score Values: (-0.660, -0.132, 1.893, 0.044, 1.277, -1.012, -1.091, -0.308)

---

6. What is feature selection? Explain filtering technique. (CO2)

Feature selection is the process of automatically or manually selecting a subset of the most relevant features from a dataset to be used in training a machine learning model. The primary goals are to improve model performance, reduce overfitting, decrease training time, and create simpler, more interpretable models.

Filtering Technique

Filter methods are a type of feature selection that ranks features based on their intrinsic statistical properties and their relationship with the target variable, independent of any machine learning algorithm. They "filter" out irrelevant or redundant features before the model training process begins.

Characteristics:

- **Fast:** They are computationally very fast and efficient.
- **Model-Agnostic:** The feature selection is done without involving a specific learning model.
- **Univariate:** They typically evaluate each feature independently.

Common Filtering Techniques:

1. **Pearson's Correlation:**
   - **What it does:** Measures the linear relationship between two continuous variables. The correlation coefficient ranges from -1 to +1.
   - **How it's used:** Features that are highly correlated with the target variable are considered good candidates. Also, if two features are highly correlated with each other, one might be removed to reduce redundancy.
2. **Chi-Squared Test:**
   - **What it does:** Used to test the relationship between two categorical variables. It determines whether there is a significant association between the feature and the target class.
   - **How it's used:** It is applied to categorical features to see how dependent they are on the target variable. A higher Chi-Squared value indicates a stronger relationship.
3. **Information Gain (Mutual Information):**
   - **What it does:** Measures the reduction in uncertainty (entropy) about the target variable given the knowledge of a feature. A higher information gain means a feature is more useful for predicting the target.
   - **How it's used:** It's often used with decision tree algorithms but can be used as a general filter method. It ranks features based on how much information they provide about the class.

---

7. Calculate LBP code generated value for the central point in the neighborhood of 8 pixels as shown below. (CO2)

This is the same example used to explain LBP in question 1.

Pixel Neighborhood:

10	12	18
7	**6**	**2**
1	**4**	**9**

1. **Center Pixel (Threshold):** 6
2. **Comparison and Binary Generation (Clockwise from top-left):**
   - 10 ≥ 6 ⟶ **1**
   - 12 ≥ 6 ⟶ **1**
   - 18 ≥ 6 ⟶ **1**
   - 2 < 6 ⟶ **0**

   - 9 ≥ 6 → **1**
   - 4 < 6 → **0**
   - 1 < 6 → **0**
   - 7 ≥ 6 → **1**
3. **Binary String:** 11101001
4. Decimal Conversion:

   (1·27)+(1·26)+(1·25)+(0·24)+(1·23)+(0·22)+(0·21)+(1·20)
   =128+64+32+0+8+0+0+1
   =233

The generated LBP code value is **233**.

---

8. What is Matrix factorization? Explain content based filtering with an example. (CO2)

Matrix Factorization

Matrix factorization is a powerful technique used in recommender systems to uncover latent (hidden) features that explain user-item interactions. The core idea is to decompose the large, sparse user-item interaction matrix (e.g., a matrix of user ratings for movies) into the product of two smaller, dense matrices:

1. A **user-feature matrix**, where rows represent users and columns represent latent features (e.g., genres like action, comedy; or more abstract concepts).
2. An **item-feature matrix**, where rows represent latent features and columns represent items.

By multiplying these two smaller matrices, we can reconstruct the original user-item matrix, but now with predicted values for the empty cells (the items the user has not yet rated). This allows the system to recommend items a user is likely to enjoy.

Content-Based Filtering

Content-based filtering is a type of recommender system that recommends items to a user based on the similarity of those items to ones the user has liked in the past. It relies entirely on the properties (or "content") of the items and a profile of the user's preferences.

How it works:

1. **Item Profile:** Features are extracted from the items. For movies, these could be genre, director, actors, plot keywords, etc.
2. **User Profile:** A profile is built for the user that summarizes their preferences based on the items they have rated. If a user likes movies with a specific actor, their profile will reflect this preference.
3. **Recommendation:** The system matches the item profiles with the user profile to find and recommend items that are most similar to the user's known preferences.

Example:

- **User's History:** A user watches and gives high ratings to the movies *Iron Man*, *The Avengers*, and *Captain America: Civil War*.
- **Item Profiles:** The system analyzes the content of these movies and identifies key features:
  - **Genre:** Action, Sci-Fi, Superhero
  - **Actors:** Robert Downey Jr., Chris Evans
  - **Studio:** Marvel
- **User Profile Creation:** The system creates a profile for this user that indicates a strong preference for "Marvel Superhero movies" and "movies starring Robert Downey Jr."
- **Recommendation:** The system then searches its database for other movies with similar item profiles. It might recommend *Spider-Man: Homecoming* because it is also a Marvel Superhero movie and features Robert Downey Jr. It would not recommend a romantic comedy, as its content features do not match the user's profile.

---

9. Given following data for attribute AGE calculate Z- score normalization. (CO2)

Given data: AGE = {18, 22, 25, 42, 28, 43, 33, 35, 56, 28}

Z-score normalization transforms data to have a mean of 0 and a standard deviation of 1.
The formula is:

$Z=\sigma X-\mu$

1. **Calculate the Mean (μ):**
   - $\mu=(18+22+25+42+28+43+33+35+56+28)/10$

- $\mu=330/10=33$

2. **Calculate the Standard Deviation (σ):**
   - Variance $\sigma2=N\Sigma(Xi-\mu)2$
   - $\sigma2=[(-15)2+(-11)2+(-8)2+(9)2+(-5)2+(10)2+(0)2+(2)2+(23)2+(-5)2]/10$
   - $\sigma2=[225+121+64+81+25+100+0+4+529+25]/10$
   - $\sigma2=1174/10=117.4$
   - Standard Deviation $\sigma=117.4\approx10.835$
3. **Calculate the Z-score for each value:**
   - 18: $(18-33)/10.835=-15/10.835\approx-1.384$
   - 22: $(22-33)/10.835=-11/10.835\approx-1.015$
   - 25: $(25-33)/10.835=-8/10.835\approx-0.738$
   - 42: $(42-33)/10.835=9/10.835\approx0.831$
   - 28: $(28-33)/10.835=-5/10.835\approx-0.461$
   - 43: $(43-33)/10.835=10/10.835\approx0.923$
   - 33: $(33-33)/10.835=0/10.835=0.0$
   - 35: $(35-33)/10.835=2/10.835\approx0.185$
   - 56: $(56-33)/10.835=23/10.835\approx2.123$
   - 28: $(28-33)/10.835=-5/10.835\approx-0.461$

Z-Score Normalized Values: {-1.384, -1.015, -0.738, 0.831, -0.461, 0.923, 0.0, 0.185, 2.123, -0.461}

---

10. Why do you need categorical variable encoding? With an example, explain one-hot encoding. (CO2)

Why Categorical Variable Encoding is Needed

Most machine learning algorithms are based on mathematical equations and statistical calculations. They require input data to be numerical to perform these operations. Categorical variables, which are textual or qualitative (e.g., "Red", "Green", "Blue" or "High", "Medium", "Low"), cannot be directly used by these algorithms.

Categorical variable encoding is the process of converting these non-numerical, categorical variables into a numerical format that machine learning models can understand and work with. Without this step, the model cannot be trained on the data.

One-Hot Encoding

One-Hot Encoding is one of the most common techniques for handling nominal categorical variables (where categories have no intrinsic order). It works by creating new binary (0 or 1) columns for each unique category in the original variable.

Example:
Imagine you have a dataset with a Color feature:

ID	Color
1	Red
2	Green
3	Blue
4	Red

1. **Identify Unique Categories:** The unique categories in the Color column are "Red", "Green", and "Blue".
2. **Create New Binary Columns:** One-Hot Encoding will create three new columns, one for each category: Color_Red, Color_Green, and Color_Blue.
3. **Populate the Columns:** For each row, a '1' is placed in the column corresponding to its original category, and '0's are placed in all other new columns.

The transformed dataset would look like this:

ID	Color_Red	Color_Green	Color_Blue
1	1	0	0
2	0	1	0
3	0	1	
4	1	0	0

This method avoids introducing a false order or relationship between categories, which could happen if we simply assigned numbers like Red=1, Green=2, and Blue=3.

---

11. Which statistical methods are used to describe the nature of data? (CO2)

The statistical methods used to describe and summarize the nature of data are part of **descriptive statistics**. They can be broadly categorized into two main types:

1. Measures of Central Tendency

These measures describe the center or typical value of a dataset.

- **Mean:** The average of all data points. It is calculated by summing all values and dividing by the count of values. It is sensitive to outliers.
- **Median:** The middle value in a dataset that has been sorted in ascending order. If there is an even number of values, it is the average of the two middle values. It is robust to outliers.
- **Mode:** The value that appears most frequently in a dataset. A dataset can have one mode, more than one mode (multimodal), or no mode.

2. Measures of Variability (or Dispersion)

These measures describe the spread or dispersion of the data points.

- **Range:** The difference between the highest and lowest values in a dataset. It is simple to calculate but highly sensitive to outliers.
- **Variance (σ2):** The average of the squared differences from the Mean. It measures how far the data points are spread out from the average value. A higher variance indicates greater spread.
- **Standard Deviation (σ):** The square root of the variance. It is the most common measure of spread and is expressed in the same units as the data, making it easier to interpret than variance.
- **Interquartile Range (IQR):** The range between the first quartile (25th percentile) and

the third quartile (75th percentile) of the data. It represents the spread of the middle 50% of the data and is robust to outliers.

---

12. What are the feature of multidimensional scaling? (CO2)

Multidimensional Scaling (MDS) is a dimensionality reduction and data visualization technique. Its primary purpose is to represent the similarity or dissimilarity between pairs of objects as distances in a lower-dimensional space (usually 2D or 3D).

The key features of MDS are:

1. **Input is a Proximity Matrix:** Unlike techniques like PCA which require a feature-based dataset, the primary input for MDS is a **dissimilarity matrix** (or similarity matrix). This matrix quantifies the pairwise "distances" or dissimilarities between all items in a set.
2. **Preservation of Pairwise Distances:** The core goal of MDS is to arrange points in a low-dimensional space such that the distances between them in that space reflect, as closely as possible, the original dissimilarities. If two objects were very dissimilar in the original data, their corresponding points in the MDS plot will be far apart.
3. **Visualization:** Its main application is visualization. By projecting complex, high-dimensional data into a 2D or 3D scatter plot, MDS helps in uncovering underlying structures, patterns, clusters, and relationships that are not apparent in the raw data.
4. **Types of MDS:**
   - **Classical MDS (Metric MDS):** This type assumes the input dissimilarities are actual distances (metric) and aims to preserve these distances precisely. It is mathematically equivalent to PCA when the input is Euclidean distance.
   - **Non-metric MDS:** This type is more flexible. It assumes the input dissimilarities are only an ordering (ordinal). It tries to preserve the *rank order* of the dissimilarities, not their exact values. For example, if item A is more dissimilar to B than C is to D, the distance between points A and B in the plot should be greater than the distance between C and D.