Interview Questions-2

Dimensionality Reduction

(Practice Projects)

PW SKILLS

Q1. You are working on a customer segmentation project for an e-commerce company. How can you reduce the dimensionality of the customer features and extract the most important information for clustering?

Answer: In the customer segmentation project, Principal Component Analysis (PCA) can be utilized to reduce the dimensionality of the customer features and extract the most important information for clustering. Firstly, I would preprocess the data by scaling the features to have zero mean and unit variance. Next, I would apply PCA to transform the feature space into a lower-dimensional space while retaining the maximum amount of variance. By analyzing the explained variance ratio, I can identify the principal components that capture the most significant information. These components represent the combinations of original features that explain the most variance in the dataset. By selecting a subset of the principal components that retain a significant amount of variance, we can effectively reduce the dimensionality of the customer features. This reduction not only simplifies the subsequent clustering process but also helps in visualizing the data and gaining insights into distinct customer segments.

Q2. You are working on a visualization project for high-dimensional data and want to uncover its underlying structure. How can you visualize the data in a lower-dimensional space while preserving the local and global relationships between data points?

Answer: In the visualization project for high-dimensional data, t-Distributed Stochastic Neighbor Embedding (t-SNE) can be utilized to uncover the underlying structure of the data. t-SNE is a dimensionality reduction technique that focuses on preserving both local and global relationships between data points. To use t-SNE, I would start by calculating pairwise similarities or distances between the data points. Then, I would employ a probability distribution to represent the similarities in the high-dimensional space and another distribution in the low-dimensional space. Next, t-SNE minimizes the Kullback-Leibler divergence between these two distributions, resulting in a lower-dimensional representation that reflects the underlying structure of the data. By visualizing the transformed data points in the lower-dimensional space, we can identify clusters, patterns, or relationships that were not apparent in the original high-dimensional space. t-SNE is particularly useful for revealing intricate structures and local neighborhoods within the data, making it a powerful tool for exploratory analysis and visualization.

Problem Statement

Context: RML Delhi Hospital, which serves thousands of patients across India, requires an advanced automated system to classify cancer as benign or malignant. As the data scientist, your mission is to develop a model that leverages dimensionality reduction techniques to enhance accuracy and efficiency in predicting cancer types. Objective: You need to create a system that can predict whether a cancer diagnosis is benign or malignant based on features such as mean radius, mean compactness, and worst area. The classification output will guide medical decisions, such as whether or not a patient should undergo surgery.

Dataset: Dataset

Question 3: What is the distribution of the Target Values? Check if data is balanced or not?

Answer:

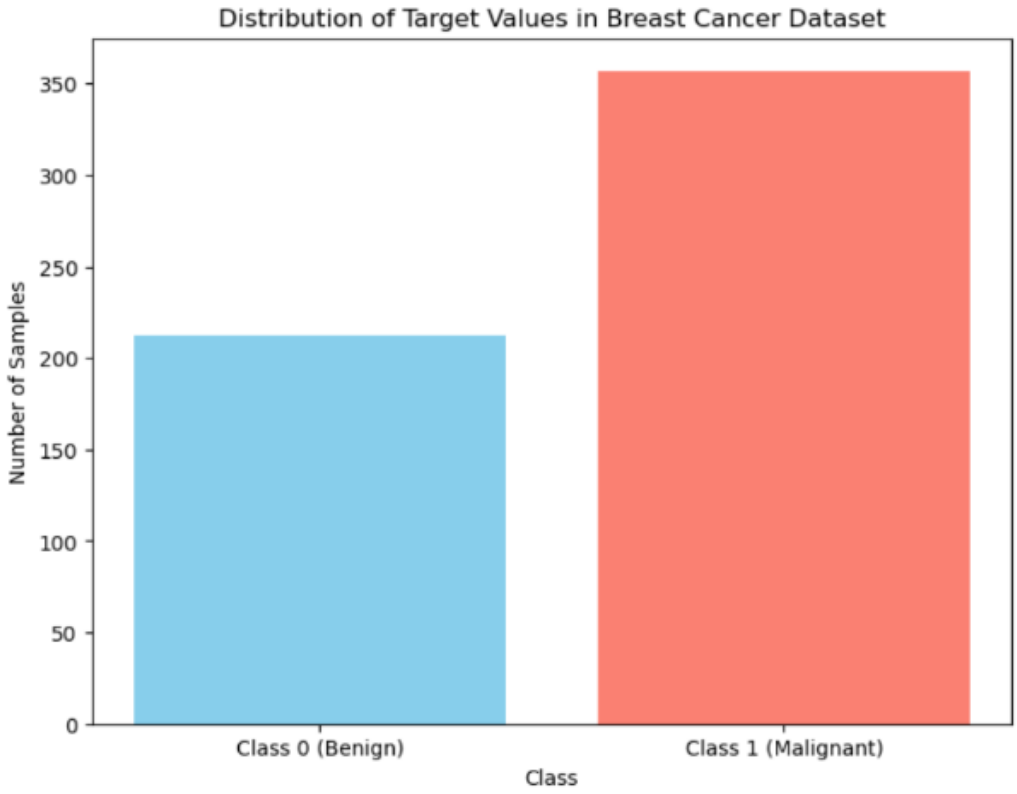
```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.datasets import load_breast_cancer

def plot_target_distribution():
    # Load the dataset
    data = load_breast_cancer()
    target = data.target

    # Calculate the distribution of target values
    unique, counts = np.unique(target, return_counts=True)
    distribution = dict(zip(unique, counts))

    # Create a bar plot
    plt.figure(figsize=(8, 6))
    plt.bar(distribution.keys(), distribution.values(),
color=['skyblue', 'salmon'])
    plt.xticks(ticks=[0, 1], labels=['Class 0 (Benign)', 'Class 1
(Malignant)'])
    plt.xlabel('Class')
    plt.ylabel('Number of Samples')
    plt.title('Distribution of Target Values in Breast Cancer
Dataset')
    plt.show()

plot_target_distribution()
```



Question4: Identifying Columns Least Related to the Target Class

Solution: Use ANOVA to determine the significance of the relationship between each numerical column and the target class. The columns that do not show a significant relationship (p-value ≥ 0.05) are considered least related.

Code:

```
import pandas as pd
from scipy import stats
from sklearn.datasets import load_breast_cancer

def identify_least_related_columns():
    # Load the dataset
    data = load_breast_cancer()
    df = pd.DataFrame(data.data, columns=data.feature_names)
    df['target'] = data.target

    # Get numerical columns
    num_cols = list(df.select_dtypes('float64').columns)
    unrelated_num_cols = []
    categorical_col = 'target'

    # Perform ANOVA for each numerical column
    for i in num_cols:
        # Perform ANOVA test
        grouped_data = [df[i][df[categorical_col] == category]
for category in df[categorical_col].unique()]
        statistic, p_value = stats.f_oneway(*grouped_data)

        # Set the significance level (alpha)
        alpha = 0.05

        # Print results
        print(f"ANOVA statistic for {i}: {round(statistic, 2)}")
        print(f"p-value: {p_value}")

        if p_value < alpha:
            print("\033[32m" + f"Reject the null hypothesis:
There is a significant relationship between {i} and
{categorical_col}")
            print("\033[0m")  # Reset text color to default
        else:
            print("\033[31m" + f"No significant relationship
between {i} and {categorical_col}")
            print("\033[0m")  # Reset text color to default
            unrelated_num_cols.append(i)

    print(f'The columns that failed the ANOVA test are
{unrelated_num_cols}. These columns don\'t help in making
predictions.')
    return unrelated_num_cols

# Run the function
unrelated_columns = identify_least_related_columns()
```

Result:

```
ANOVA statistic for mean radius: 646.98
p-value: 8.46594057226382e-96
Reject the null hypothesis: There is a significant relationship
between mean radius and target

ANOVA statistic for mean texture: 118.1
p-value: 4.0586360478986136e-25
Reject the null hypothesis: There is a significant relationship
between mean texture and target

ANOVA statistic for mean perimeter: 697.24
p-value: 8.43625103617395e-101
Reject the null hypothesis: There is a significant relationship
between mean perimeter and target

ANOVA statistic for mean area: 573.06
p-value: 4.7345643103077244e-88
Reject the null hypothesis: There is a significant relationship
between mean area and target

ANOVA statistic for mean smoothness: 83.65
p-value: 1.051850359203376e-18
Reject the null hypothesis: There is a significant relationship
between mean smoothness and target

ANOVA statistic for mean compactness: 313.23
p-value: 3.938263105887389e-56
Reject the null hypothesis: There is a significant relationship
between mean compactness and target

ANOVA statistic for mean concavity: 533.79
p-value: 9.966555755074321e-84
Reject the null hypothesis: There is a significant relationship
between mean concavity and target

ANOVA statistic for mean concave points: 861.68
p-value: 7.101150161059154e-116
Reject the null hypothesis: There is a significant relationship
between mean concave points and target

ANOVA statistic for mean symmetry: 69.53
p-value: 5.73338402846732e-16
Reject the null hypothesis: There is a significant relationship
between mean symmetry and target

ANOVA statistic for mean fractal dimension: 0.09
p-value: 0.7599368037255626
No significant relationship between mean fractal dimension and
target

ANOVA statistic for radius error: 268.84
p-value: 9.738948656462521e-50
Reject the null hypothesis: There is a significant relationship
between radius
```

```
error and target

ANOVA statistic for texture error: 0.04
p-value: 0.8433320287670788
No significant relationship between texture error and target

ANOVA statistic for perimeter error: 253.9
p-value: 1.6519051758498877e-47
Reject the null hypothesis: There is a significant relationship
between perimeter error and target

ANOVA statistic for area error: 243.65
p-value: 5.895521392606798e-46
Reject the null hypothesis: There is a significant relationship
between area error and target

ANOVA statistic for smoothness error: 2.56
p-value: 0.11029660865790443
No significant relationship between smoothness error and target

ANOVA statistic for compactness error: 53.25
p-value: 9.975994654075776e-13
Reject the null hypothesis: There is a significant relationship
between compactness error and target

ANOVA statistic for concavity error: 39.01
p-value: 8.260176167970819e-10
Reject the null hypothesis: There is a significant relationship
between concavity error and target

ANOVA statistic for concave points error: 113.26
p-value: 3.072308768818453e-24
Reject the null hypothesis: There is a significant relationship
between concave points error and target

ANOVA statistic for symmetry error: 0.02
p-value: 0.8766418183860331
No significant relationship between symmetry error and target

ANOVA statistic for fractal dimension error: 3.47
p-value: 0.06307355082240175
No significant relationship between fractal dimension error and
target

ANOVA statistic for worst radius: 860.78
p-value: 8.482291921685478e-116
Reject the null hypothesis: There is a significant relationship
between worst radius and target

ANOVA statistic for worst texture: 149.6
p-value: 1.0780574879494634e-30
Reject the null hypothesis: There is a significant relationship
between worst texture and target
```

```
ANOVA statistic for worst perimeter: 897.94
p-value: 5.771397139669512e-119
Reject the null hypothesis: There is a significant relationship
between worst perimeter and target

ANOVA statistic for worst area: 661.6
p-value: 2.8288477042872213e-97
Reject the null hypothesis: There is a significant relationship
between worst area and target

ANOVA statistic for worst smoothness: 122.47
p-value: 6.575143633985019e-26
Reject the null hypothesis: There is a significant relationship
between worst smoothness and target

ANOVA statistic for worst compactness: 304.34
p-value: 7.069816352539104e-55
Reject the null hypothesis: There is a significant relationship
between worst compactness and target

ANOVA statistic for worst concavity: 436.69
p-value: 2.464663956782974e-72
Reject the null hypothesis: There is a significant relationship
between worst concavity and target

ANOVA statistic for worst concave points: 964.39
p-value: 1.9690997072169153e-124
Reject the null hypothesis: There is a significant relationship
between worst concave points and target

ANOVA statistic for worst symmetry: 118.86
p-value: 2.9511205771542106e-25
Reject the null hypothesis: There is a significant relationship
between worst symmetry and target

ANOVA statistic for worst fractal dimension: 66.44
p-value: 2.31643244998312e-15
Reject the null hypothesis: There is a significant relationship
between worst fractal dimension and target

The columns that failed the ANOVA test are ['mean fractal
dimension', 'texture error', 'smoothness error', 'symmetry
error', 'fractal dimension error']. These columns don't help in
making predictions.
```

Question 5: Is data scaling necessary for this dataset, and how does dimensionality reduction come into the picture?

Answer:
Data scaling is crucial for many machine learning algorithms, especially those that are sensitive to the scale of the data. In the context of this dataset:

Need for Data Scaling:
Normalization (Min-Max Scaling) and Standardization (Z-score Normalization) are two common data scaling techniques. They adjust the data to a common scale, which can significantly impact the performance of many algorithms.

Normalization scales the data to a fixed range, usually [0, 1]. This technique is useful when features have different units or ranges, and you need to preserve the relative distances between data points.
Standardization rescales the data to have a mean of 0 and a standard deviation of 1. This method is less sensitive to outliers compared to normalization and is commonly used when features are assumed to follow a Gaussian distribution.

Given the dataset's characteristics (e.g., features with different scales or ranges), data scaling is important. For algorithms like Principal Component Analysis (PCA), scaling ensures that all features contribute equally to the analysis, preventing features with larger ranges from dominating the results.

Dimensionality Reduction and Data Scaling:
Dimensionality Reduction techniques like PCA benefit from data scaling because they are sensitive to the scale of the data. PCA identifies the directions (principal components) along which the variance of the data is maximized. If the data is not scaled, features with larger ranges will disproportionately influence the principal components.

Scaling the data ensures that each feature contributes equally to the computation of the principal components. This helps in obtaining meaningful and interpretable results from PCA.
Normalization and Standardization are preprocessing steps that can improve the effectiveness of PCA. They help in standardizing the data to a common scale, making the dimensionality reduction process more robust and accurate.

In summary: Scaling the data is essential for effective dimensionality reduction, particularly when using PCA. It ensures that each feature contributes equally and that the results of the dimensionality reduction are reliable and meaningful.

Code:

```
scaler = MinMaxScaler()
scaler.fit(inputs_df[input_cols])
inputs_df[input_cols] = scaler.transform(inputs_df[input_cols])
inputs_df[input_cols].head()
```

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	mean_compactness	mean_concavity	mean_concave_points	r
0	0.521037	0.022658	0.545989	0.363733	0.593753	0.792037	0.703140	0.731113	
1	0.643144	0.272574	0.615783	0.501591	0.289880	0.181768	0.203608	0.348757	
2	0.601496	0.390260	0.595743	0.449417	0.514309	0.431017	0.462512	0.635686	
3	0.210090	0.360839	0.233501	0.102906	0.811321	0.811361	0.565604	0.522863	
4	0.629893	0.156578	0.630986	0.489290	0.430351	0.347893	0.463918	0.518390	

Question 6: Why do we calculate the covariance matrix in Principal Component Analysis (PCA)?

Answer:
The covariance matrix plays a critical role in Principal Component Analysis (PCA) because it provides essential information about the relationships between the features in a dataset. Here's why it is calculated and how it is used in PCA:

Purpose of the Covariance Matrix:
The covariance matrix captures both the variances of individual features and the covariances between pairs of features.

- **Diagonal Elements:** Represent the variances of the individual features. Variance measures how much a feature deviates from its mean.

- **Off-Diagonal Elements:** Represent the covariances between different features. Covariance indicates the direction of the linear relationship between pairs of features. Positive covariance means that the features tend to increase together, while negative covariance indicates that one feature tends to decrease when the other increases.

Role in PCA:
- **Identify Principal Components:** PCA aims to find the directions (principal components) along which the data varies the most. These directions are determined by the eigenvectors of the covariance matrix.

- **Calculate Eigenvectors and Eigenvalues:** The eigenvectors of the covariance matrix represent the directions of maximum variance in the data, while the eigenvalues represent the magnitude of the variance along those directions. In other words, the eigenvectors give the principal components, and the eigenvalues tell us how much variance each principal component explains.

- **Dimensionality Reduction:** By projecting the data onto these principal components, PCA reduces the dimensionality of the data while retaining as much of the original variance as possible. This helps in simplifying the data while preserving important patterns.

Process:
- **Compute Covariance Matrix:** First, calculate the covariance matrix for the dataset. This matrix provides a summary of how the features in the dataset vary and how they are correlated.

- **Eigen Decomposition:** Perform eigen decomposition on the covariance matrix to find the eigenvectors and eigenvalues.

- **Select Principal Components:** The eigenvectors with the largest eigenvalues are chosen as the principal components. These components capture the most variance in the dataset.

Code:

```
# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# 2. Compute the covariance matrix
cov_matrix = np.cov(X_scaled, rowvar=False)

# Compute eigenvalues and eigenvectors
eigen_values, eigen_vectors = np.linalg.eig(cov_matrix)

# Sort eigenvalues in descending order
sorted_indices = np.argsort(eigen_values)[::-1]
eigen_values_sorted = eigen_values[sorted_indices]
eigen_vectors_sorted = eigen_vectors[:, sorted_indices]

# Calculate explained variance
explained_variance_ratio = eigen_values_sorted /
np.sum(eigen_values_sorted)
cumulative_explained_variance =
np.cumsum(explained_variance_ratio) * 100

# Display explained variance
print("Explained Variance Cumulative Percentage:")
for i, var in enumerate(cumulative_explained_variance):
    print(f"PC{i+1}: {var:.2f}%")
```

Result:

```
Explained Variance Cumulative Percentage:
PC1: 44.27%
PC2: 63.24%
PC3: 72.64%
PC4: 79.24%
PC5: 84.73%
PC6: 88.76%
PC7: 91.01%
PC8: 92.60%
PC9: 93.99%
PC10: 95.16%
PC11: 96.14%
PC12: 97.01%
PC13: 97.81%
PC14: 98.34%
PC15: 98.65%
PC16: 98.92%
PC17: 99.11%
PC18: 99.29%
PC19: 99.45%
PC20: 99.56%
PC21: 99.66%
PC22: 99.75%
PC23: 99.83%
PC24: 99.89%
PC25: 99.94%
PC26: 99.97%
PC27: 99.99%
PC28: 100.00%
PC29: 100.00%
PC30: 100.00%
```

Question 7: Determine the number of principal components required to explain at least 90% of the variance in the data.

Solution:

```
# 3. Determine the number of components explaining at least 90%
variance
num_components = np.argmax(cumulative_explained_variance ⩾ 90) +
1
print(f"Number of principal components to explain at least 90%
variance: {num_components}")

# Transform the dataset using the selected number of principal
components
pca = PCA(n_components=num_components)
X_pca = pca.fit_transform(X_scaled)
```

Result:
Number of principal components to explain at least 90% variance: 7

Question 8: As this dataset is imbalanced , So how to deal with that?

Answer: We can use Smote for imbalanced data.

```
from sklearn.datasets import load_breast_cancer
from sklearn.utils.class_weight import compute_class_weight
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE
import numpy as np

# Load dataset
data = load_breast_cancer()
X = data.data
y = data.target

# Handle class imbalance
smote = SMOTE()
X_resampled, y_resampled = smote.fit_resample(X, y)

# Scale data
scaler = StandardScaler()
X_resampled_scaled = scaler.fit_transform(X_resampled)

# Apply PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_resampled_scaled)

print("Class distribution after SMOTE:",
np.bincount(y_resampled))
```