Simple Linear Regression - Viva Questions

Conceptual Questions

1. What is Principal Component Analysis (PCA)?

- PCA is a dimensionality reduction technique used to reduce the number of features in a dataset while retaining as much variability as possible.

2. What is the goal of PCA?

- To find a set of uncorrelated principal components that maximize the variance in the dataset.

3. What are eigenvectors and eigenvalues in the context of PCA?

- Eigenvectors represent the direction of the principal components, while eigenvalues indicate the magnitude of variance explained by each principal component.

4. How is PCA different from linear regression?

- PCA focuses on reducing dimensionality and finding patterns in the data, while linear regression models the relationship between independent and dependent variables.

5. What is the covariance matrix, and why is it important in PCA?

- The covariance matrix represents the relationships (covariance) between features. PCA uses it to compute eigenvalues and eigenvectors.

Mathematical and Practical Questions

6. How are principal components calculated?

- Steps:

    - Compute the covariance matrix of the data.

    - Calculate eigenvalues and eigenvectors of the covariance matrix.

    - Sort eigenvectors by decreasing eigenvalues.

    - Select the top k eigenvectors to form the principal components.

7. What is the explained variance ratio?

- It indicates the proportion of total variance explained by each principal component.

8. How do you decide the number of components to retain in PCA?

- Use the explained variance ratio, scree plot, or cumulative variance threshold (e.g., 95%).

9. What happens if the original data is not standardized in PCA?

- Features with larger ranges may dominate the principal components, leading to biased results.

10. How does PCA handle correlated features?

- PCA creates uncorrelated principal components by projecting the data onto orthogonal axes.

Implementation Questions

11. How would you implement PCA in Python?

- Steps:

- Import libraries: `from sklearn.decomposition import PCA`
- Standardize the data using `StandardScaler()`.
- Initialize PCA: `pca = PCA(n_components=k)`.
- Fit and transform the data using `pca.fit_transform()`.

12. What are some common applications of PCA?

- Noise reduction, feature extraction, data visualization, and speeding up machine learning models.

13. Can PCA be used for categorical data?

- Not directly; categorical data must first be encoded into numerical values.

14. How do you interpret the output of PCA?

- The principal components are linear combinations of the original features, and their coefficients indicate feature contributions.

15. What is the role of the singular value decomposition (SVD) in PCA?

- PCA can be computed using SVD, which decomposes the data matrix into orthogonal components.

Limitations and Scenario-Based Questions

16. What are the limitations of PCA?

- Loss of interpretability of original features, assumption of linearity, sensitivity to outliers, and reliance on mean-centered data.

17. How does PCA affect classification or regression tasks?

- PCA can improve performance by removing multicollinearity but may also lead to loss of important information if too many components are discarded.

18. How would you handle missing data in PCA?

- Impute missing values before applying PCA, using methods like mean imputation or KNN imputation.

19. How does PCA differ from t-SNE?

- PCA is linear and focuses on variance, while t-SNE is non-linear and focuses on preserving local structure for visualization.

20. In what scenarios would you not use PCA?

- When interpretability of features is critical or when the dataset is small and already has low-dimensional features.