Department of Mathematical and Computational Sciences
National Institute of Technology Karnataka, Surathkal

https://sam.nitk.ac.in/ sam@nitk.edu.in

MA722 - Foundations of Machine Learning Algorithms

Lab Assignment Questions - 1

---

1. For the following equations of two parallel lines in the form:

$$ax + by + c_1 = 0$$

$$ax + by + c_2 = 0$$

where a, b, c_1, and c_2 are real numbers, write a Python programme that

- takes inputs c_1, and c_2, and
- calculates the **distance between the two parallel lines**.

Note:

- Take a and b as the digits of your "Register Number" in unit's and ten's places respectively. For example, if your "Register Number" is 25203972520397, then $a = 7$ and $b = 9$. If a or b is 0 in your "Register Number", take a or b is 1.
- Include input validation to ensure the lines are parallel (i.e., coefficients a and b must be the same). If the lines are not parallel, raise an exception or return an appropriate message.

2. For a given set of D vectors, write a Python programme to check if the vectors are **orthonormal**.

3. For any two $n \times 1$ vectors v and w, write a Python code to compute the **Euclidean Distance** between them.

4. For two sparse matrices (i.e., the number of zero entries is significantly greater than the number of non-zero entries) $A_{m\times p}$ and $B_{p\times n}$, compute AB (**Sparse Multiplication**) by:

- Skipping the zero entries,
- Only using positions where both A and B have non-zero values.

5. For two random variables X and Y, each with N data points:

- Calculate **Expectation, Variance, Covariance and Correlation Coefficients**.
- Check whether the correlation of two variables are positive, negetive or zero. If the covariance is zero, print 'No linear relationship between the two features'.

6. Find the **Covariance Matrix** for D features, each with N data points.

7. Write a programme to solve the optimization problem:

$$f(x, y) = x^2 + y^2$$

subject to constraint

$$g(x, y) = x + y - 1 = 0$$

using the **Lagrange Multiplier Method** and display the solution as output.

8. For a given set of results of m independent coin tosses, where

   - 1 represents heads (success)
   - 0 represents tails (failure).

   Assume that each toss follows a **Bernoulli Distribution** with an unknown probability p of success. Write a Python code that uses the Maximum Likelihood Estimation (MLE) method to estimate the value of p, the probability of getting heads.

9. Given a list of real-valued data assumed to come from a **Normal Distribution**, estimate the following parameters using Maximum Likelihood Estimation (MLE).

   - The MLE of the **mean** μ
   - The MLE of the **variance** σ^2

   Input : A list of real numbers (e.g., `[1.2, 0.9, 1.5, 2.0, 1.7]`)

10. Write a Python programme to compute the **Singular Value Decomposition (SVD)** of a real-valued matrix A. The SVD decomposes the matrix into three components

$$A = U\Sigma V^T,$$

where

   - $U \in \mathbb{R}^{m \times m}$ is an orthogonal matrix containing the left singular vectors,
   - Σ is a diagonal matrix (or a vector) containing the singular values in descending order,
   - $V^T \in \mathbb{R}^{n \times n}$ is the transpose of an orthogonal matrix containing the right singular vectors.

   Note:

   - `Input:` A 2D NumPy array $A \in \mathbb{R}^{m \times n}$, containing real numbers.
   - `Output:` A tuple `(U, S, VT)`, where:
     - `U`: $m \times m$ matrix of left singular vectors,
     - `S`: 1D array of singular values (length = $\min(m, n)$),
     - V^T: $n \times n$ matrix, the transpose of right singular vectors.

11. Consider the **Iris dataset**, which contains 150 samples of Iris flowers, each described by four numerical features:

    - Sepal length

- Sepal width
- Petal length
- Petal width

Each sample belongs to one of three species: *setosa*, *versicolor*, or *virginica*. Apply **Principal Component Analysis (PCA)** to reduce the dimensionality of the dataset from 4 to 2 in order to visualize how well the data separates by species.

12. A university wants to predict whether students will be admitted based on their scores from two exams. The dataset contains:

Exam 1	Exam 2	Admitted (0 = No, 1 = Yes)
34	78	0
45	85	0
50	43	1
65	72	1
70	90	1

Build a **Logistic Regression Model** that predicts admission based on exam scores.

13. Write a Python program to implement **Linear Regression**, **Ridge Regression**, and **Lasso Regression without using** `scikit-learn` **or any other machine learning library**, using a dataset loaded from a `.csv` file.

Requirements:

(a) Implement Regression Models:
- **Linear Regression** using the normal Equation
- **Ridge Regression** using **L2 regularization**
- **Lasso Regression** using **L1 regularization** and an iterative optimization method.

(b) Train All Models on the dataset and compute their respective coefficients.

(c) Plot the Model Predictions:
- For single-feature models: plot the data points and regression lines.
- For multi-feature models: optionally visualize prediction errors or compare MSE/R^2.

(d) Print the Coefficients of all models for comparison.

14. Using the Iris dataset from `sklearn.datasets.load_iris`, implement **Linear Discriminant Analysis (LDA)** from scratch with the following steps:

- Compute the **within-class scatter matrix** S_W.
- Compute the **between-class scatter matrix** S_B.
- Solve the eigenvalue problem for the matrix $S_W^{-1}S_B$.
- Project the data onto the top discriminant components (up to two components).
- Visualize the resulting 2D projection with distinct colors for each class.

Note: Do not use any built-in LDA functions. You may use libraries only for data loading, linear algebra operations, and visualization.

15. The following dataset consist of two features:

    - Number of times the word "buy" appears in the email.
    - Number of hyperlinks in the email.

    Write a python programme to classify whether an email is **spam** (label 1) or **not spam** (label 0).

    Dataset:

buy_words	hyperlinks	spam
10	5	1
1	1	0
5	3	0
3	2	0
15	7	1
2	1	0
7	4	0
9	6	1
20	10	1
0	0	0

    Instructions:

    (a) Load the provided dataset containing the features (number of "buy" words and hyperlinks) and labels (spam: 1, not spam: 0).

    (b) Split the dataset into a training set and a testing set.

    (c) Train a **Support Vector Machine (SVM)** classifier with a **linear kernel**.

    (d) Evaluate the classifier's performance on the test set.

    (e) Print the accuracy of the classifier.

    (f) Plot the dataset points with different colors for spam and not spam emails. Also, plot the decision boundary (the separating hyperplane) learned by the SVM.

*****