ACM AI Recruitment Task S3

Task 1: Build an Unsupervised Learning Model on a Dataset(Clustering Model)

In this task, you are expected to implement a clustering model on a dataset of your choice. Clustering is a form of unsupervised learning that groups similar data points based on inherent patterns and structures in the data, without any labels.

Steps to Follow:

1. Dataset Selection:
    - Choose a dataset suitable for clustering. You can use well-known datasets like the Iris dataset, customer segmentation data, or any other dataset with unlabeled data. Ensure the dataset has enough features and variety for meaningful clustering.
2. Preprocessing:
    - Clean the dataset: Handle missing data, if any.
    - Normalize/standardize the dataset to bring all features to the same scale.
    - Apply dimensionality reduction techniques (like PCA) if needed to reduce computational complexity or improve clustering performance.
3. Model Development:
    - Apply a clustering algorithm (e.g., K-means, DBSCAN, Agglomerative clustering) using Python libraries such as Scikit-learn.
    - Evaluate the quality of the clusters using metrics such as the Silhouette Score, Davies-Bouldin Index, or Inertia.
    - Visualization: Use data visualization techniques to plot the clusters (e.g., using Matplotlib or Seaborn).
4. Results and Insights:
    - Discuss the clusters you obtained.
    - Try to analyze the characteristics of each cluster and any patterns found in the data.
    - Optionally, different clustering algorithms can be applied, and their performances can be compared.

Bonus (Optional):

- Build a Clustering Algorithm from Scratch: Implement a basic clustering method (e.g., K-means) without relying on external libraries, using only basic Python libraries like NumPy.
- Implement advanced clustering methods (e.g., hierarchical clustering, Gaussian Mixture Models) and explore their differences compared to traditional methods like K-means.
- Apply clustering to real-world problems like customer segmentation or image segmentation.

Task 2: Build a Classifier or Regressor from Scratch and Compare it with Standard Libraries

In this task, you need to build a classifier or regressor from scratch (without using machine learning libraries) and compare its performance with a similar model built using standard Python libraries (such as Scikit-learn).

Steps to Follow:

1. Dataset Selection:
    - Choose a labeled dataset suitable for classification or regression tasks. Examples include the Iris dataset (for classification), the housing price dataset (for regression), or any dataset you choose.
2. Preprocessing:
    - Clean the dataset: Handle missing values, normalize features, and split the dataset into training and test sets.
    - Perform feature selection or dimensionality reduction if required.
3. Building the Classifier/Regressor from Scratch:
    - Implement a machine learning algorithm from scratch using basic Python libraries such as NumPy. You can choose one of the following algorithms:
        - For Classification: Implement algorithms like Logistic Regression, K-nearest Neighbors (KNN), or Decision Trees.
        - For Regression: Implement algorithms like Linear Regression, Polynomial Regression, or K-nearest neighbors (KNN) regression.

    - Ensure to implement forward and backward propagation if you choose gradient-based methods like logistic regression.
    - Calculate evaluation metrics (such as accuracy for classification and mean squared error for regression).
4. Building the Same Model Using Standard Libraries:
    - Implement the same classifier or regressor using standard machine learning libraries like Scikit-learn.
    - Train the model on the same dataset and compare the performance metrics (like accuracy, precision, recall, F1-score, or R^2).
5. Comparison of Results:
    - Compare the performance of your scratch-built model with the model created using libraries in terms of:
        - Accuracy/performance
        - Training time
        - Code complexity
        - Scalability
    - Explain the differences in performance and discuss why the model from scratch may or may not perform as well as the library-based model.

Task 3: Build a Basic Deep Learning Network for Binary Classification

In this task, you are required to implement a basic deep learning network with 3-4 layers to perform a binary classification task. You can use either PyTorch or TensorFlow for this task.

Steps to Follow:

1. Dataset Selection:
    - Choose or create a dataset suitable for binary classification. Examples include the Breast Cancer Wisconsin dataset, Iris dataset (binary classification version), or any other dataset where the task is to classify samples into one of two classes.
2. Preprocessing:
    - Clean the dataset: Handle missing values, normalize or standardize features, and split the dataset into training and test sets.
    - Convert labels into a binary format if needed (0 and 1).

3. Model Development:
    - Define the Network Architecture:
        - Build a neural network with 3-4 layers. Model Training:
    - Define the loss function (e.g., Binary Cross-Entropy Loss).
    - Choose an optimizer (e.g., SGD, Adam) and set learning rate parameters.
    - Train the model using the training set and validate it using the validation set.
    - Implement metrics like accuracy, precision, recall, and F1-score to evaluate the model performance.
4. Evaluation and Results:
    - Test the model on the test set and analyze the results.
    - Plot the loss and accuracy curves over epochs to observe model convergence and performance.
    - Discuss any observations regarding the performance, such as potential overfitting or underfitting.
5. Documentation:
    - Document the code and model architecture.
    - Provide insights into the model's performance and any challenges faced during the implementation.

Task 4: Using trained models

OBJECTIVE

- Select a well-known or preferred LLM model (GPT, LLaMa, Mistral, Gemini, etc.) from any of the open sources (Hugging Face, Git, Kaggle, etc.) (Offline / API is accepted)
- Create a chat interface using Streamlit or any preferred GUI tool.
- Develop an interactive Chatbot utilizing the selected model and GUI. (Clearly elaborate on the model details, description, and source in a PDF within the project folder.)

QUERIES :

Lokesh Yarramallu - SIG AI Lead - 7095419591