Introduction to AI

Introduction to AI

- Definition and History of AI
- Types of AI: Narrow AI, General AI, and Super AI
- Different fields of AI : Machine learning, Deep learning, Natural language processing, Computer vision, Robotics, Speech recognition etc
- AI applications in various industries & Real-world use cases
- Common Terminologies in AI
- Differences and relationships among AI, ML, and DL.

Python Basics

Python Basics

- Installation: Anaconda with Jupyter notebook
- Syntax and Semantics
- Data Types: Lists, Tuples, Dictionaries, and Sets
- Control Flow: Loops and Conditionals
- Operators: logical airthmatic
- Functions and Modules
- File Handling
- Error Handling
- Use most known feature/funtions of these packages
    - nltk, numpy, pandas, Matplotlib/Seaborn, sckitlearn

Math for AI

Math for AI

- Statistics
    - <https://www.w3schools.com/ai/ai_statistics.asp>
- Min and Max
- Variance
- Deviation
- Distribution
- Skewness
- Kurtosis
- Probability
- Hypothesis testing

Data preprocessing

Data preprocessing

- Understanding Data:
- Types: Structured (tables), Unstructured (images, text).
- Terminology: Features, target variable, training/testing data.
- Data Cleaning:
    - Handling missing values (mean, median, mode, drop rows/columns).
    - Removing duplicates.
    - Handling outliers using IQR or z-score.
    - Feature Engineering:
- Encoding categorical variables (One-Hot Encoding, Label Encoding).
- Scaling numerical data (StandardScaler, MinMaxScaler).
- Exploratory Data Anacondalysis (EDA):
- Visualizations: histograms, boxplots, scatterplots.
- Pairwise feature relationships (pairplot in Seaborn).

NLP (Natural Language Processing)

NLP (Natural Language Processing)

- Word Embeddings
    - Introduction to Word2Vec, GloVe.
    - Learn how embeddings represent words in vector space.
    - Exercises
        - Train Word2Vec on a sample text to find realated words.
        - Identify the top 5 similar words to “food” using embeddings trained on restaurant reviews.
    - Sentiment analysis, given feedback of customer is positive, neutral, negative
- Named Entity Recognition (NER)
    - Use spaCy or nltk to extract entities (e.g., person names, locations) from a sample text.
    - Text classification is a fundamental task in NLP, widely used in spam detection, topic categorization, etc.
    - Exercise:
        - Classify restaurant reviews by categories (e.g., service, food, ambiance).

ML (Machine Learning)

ML (Machine Learning)

- Introduction to Machine Learning
    - What is Machine Learning?
    - Types of Machine Learning: Supervised, Unsupervised, and Reinforcement Learning.
    - Common terminologies: Features, Labels, Training Data, Testing Data, Overfitting, Underfitting.
- Supervised Learning
    - Linear Regression:
        - Overview and use cases.
        - How it predicts continuous values.
    - Logistic Regression:
        - Overview and use cases.
        - Binary classification and sigmoid function.
    - Decision Trees:
        - Overview and use cases.
        - How they split data based on features.
- Unsupervised Learning
    - Clustering:
        - Overview of K-Means Clustering.
        - Applications in customer segmentation.
    - Dimensionality Reduction:
        - Introduction to Principal Component Analysis (PCA).
        - Visualizing high-dimensional data in 2D.
- Exercises
    - Supervised Learning:
        - Use a pre-trained Linear Regression model to predict housing prices using the Boston Housing Dataset.
            - Load the dataset using Scikit-Learn.
            - Split the data into training and testing sets.
            - Use the provided Linear Regression model from Scikit-Learn.
            - Evaluate performance using Mean Squared Error (MSE).
        - Use a pre-trained Logistic Regression model to classify whether an email is spam or not using the SMS Spam Dataset
            - Load the dataset and preprocess text using libraries like NLTK.
            - Fit the provided Logistic Regression model and evaluate using accuracy and confusion matrix.
    - Unsupervised Learning
        - Perform K-Means Clustering on the Iris Dataset to group flowers based on features.
            - Load the Iris dataset from Scikit-Learn.
            - Use the provided K-Means implementation.
            - Visualize clusters using scatter plots.
        - Use PCA to reduce dimensions of the MNIST Dataset and visualize it in 2D.
            - Load the MNIST dataset.
            - Apply PCA from Scikit-Learn.
            - Visualize the first two principal components.
- Projects
    - Predict Loan Approval (Supervised Learning):
        - Dataset: Bank loan data from Kaggle.
        - Goal: Predict if a loan will be approved based on customer details.
        - Steps:
            - Preprocess the dataset (handle missing values and encode categorical variables).
            - Use the pre-built Decision Tree model from Scikit-Learn.

            - Evaluate metrics like accuracy and F1-score.
    - Customer Segmentation (Unsupervised Learning):
        - Dataset: E-commerce transaction data.
        - Goal: Group customers into segments using clustering.
        - Steps:
            - Preprocess data (normalize numerical features).
            - Use the pre-configured K-Means clustering model.
            - Visualize clusters using PCA and interpret segments.
            -

Deep Learning

- **Introduction to Deep Learning**
  - **Definition**: What is Deep Learning and how it differs from Machine Learning.
  - **Key Concepts**: Neural networks, deep architectures, representation learning.
  - **Applications**: Image recognition, natural language processing, autonomous vehicles, etc.
  - **Common Terminologies**:
  - **Neural Network**
  - **Epochs**
  - **Batch Size**
  - **Learning Rate**
  - **Overfitting vs. Underfitting**
  - **Gradient Descent**
  - **Backpropagation**
  - **Activation Function**
- **Basics of Neural Networks**
  - **Architecture**:
    - Input layer, hidden layers, output layer.
    - Concept of perceptrons and multi-layer perceptrons (MLPs).
  - **Activation Functions**:
    - Sigmoid, ReLU, Tanh (with examples and when to use them).
  - **Forward Propagation & Backpropagation**:
    - High-level explanation of how weights are updated.
  - **Exercise**:
    - Implement a simple MLP using **TensorFlow/Keras** on the MNIST dataset to classify digits.
- **Optimizers and Loss Functions**
  - **Loss Functions**:
    - Mean Squared Error (MSE), Cross-Entropy Loss, Hinge Loss.
  - **Optimizers**:
    - Gradient Descent, Adam, SGD, RMSProp (with examples).
  - **Importance of Learning Rate**: Explanation with visualizations of convergence.
  - **Exercise**:
    - Train a pre-configured neural network with different optimizers on the CIFAR-10 dataset and observe accuracy.
- **Convolutional Neural Networks (CNN)**
  - **Concept**: Layers (Convolution, Pooling, Flattening, Fully Connected).
  - **Use Cases**: Image recognition and classification.
  - **Popular Architectures**: VGG, ResNet (high-level overview).
  - **Exercise**:
    - Use a **pre-trained CNN model (e.g., VGG16)** in TensorFlow to classify images from the Cats vs. Dogs dataset.
    - Modify the last layer for binary classification.
- **Recurrent Neural Networks (RNN)**
  - **Concept**: Sequential data processing, hidden states, time steps.
  - **Use Cases**: Text generation, language translation, speech recognition.
  - **Variants**: LSTM and GRU (basic explanation).
  - **Exercise**:
    - Use a pre-built **LSTM model** to generate text based on Shakespeare's works.
- **Generative Adversarial Networks (GANs)**
  - **Concept**: Generator and Discriminator networks.
  - **Use Cases**: Image generation, data augmentation.
  - **Challenges**: Mode collapse, stability issues.
  - **Exercise**:

        - Use a **pre-trained GAN** (e.g., DCGAN) to generate synthetic images from the Fashion-MNIST dataset.
- **Transformers**
    - **Concept**: Attention mechanisms, encoder-decoder structure.
    - **Use Cases**: Language translation, text summarization, question answering.
    - **Popular Architectures**: BERT, GPT (overview only).
    - **Exercise**:
        - Use the Hugging Face library to load a **pre-trained BERT model** for sentiment analysis on the IMDb movie reviews dataset.
- **Real-Time Projects**
    - **Project 1: Image Classification (CNN)**
        - **Dataset**: Fashion-MNIST.
        - **Goal**: Classify images into categories like shirts, shoes, etc.
        - **Steps**:
            - Preprocess data.
            - Use a pre-trained CNN model.
            - Evaluate performance metrics.
    - **Project 2: Text Generation (RNN)**
        - **Dataset**: Any public domain text corpus (e.g., Shakespeare's plays).
        - **Goal**: Generate text sequences resembling the corpus.
        - **Steps**:
            - Preprocess text (tokenization, padding).
            - Use a pre-trained LSTM model.
            - Evaluate the generated tex

Transfer Learning Module

Transfer Learning Module

- **What is Transfer Learning?**
  - Definition: Reusing a pre-trained model on a new, related task.
  - Benefits:
  - Saves computational resources and time.
  - Requires less data compared to training from scratch.
  - Applications: Image classification, object detection, text classification, etc.
- **Types of Transfer Learning**
  - **Feature Extraction:**
    - Use pre-trained models as feature extractors without modifying the architecture.
    - Exercises
      - Use the **ResNet50** pre-trained model to classify the CIFAR-10 dataset.
      - Steps:
        - Load the CIFAR-10 dataset and preprocess it.
        - Use ResNet50 to extract features from the images.
        - Train a simple classifier (e.g., Logistic Regression) on top of these features.
  - **Fine-Tuning:**
    - Unfreeze specific layers of the pre-trained model and train them on the new dataset.
    - Exercises
      - Fine-tune the pre-trained **BERT** model to classify movie reviews as positive or negative (Sentiment Analysis).
      - Steps:
        - Use a labeled dataset like IMDB Reviews.
        - Tokenize the text using a BERT tokenizer.
        - Fine-tune the BERT model on the dataset.
        - Goal: Learn how to unfreeze and train specific layers of a pre-trained model.
- **Commonly Used Pre-Trained Models, for your kind information**
  - **Image Data:**
    - VGG16, ResNet, Inception, EfficientNet.
  - **Text Data:**
    - BERT, GPT, DistilBERT, T5.
- **Projects**
  - **Image Classification: Dog vs. Cat**
    - **Dataset**: Dog vs. Cat dataset from Kaggle.
    - **Goal**: Build a model to classify images as either “dog” or “cat.”
    - **Steps**:
      - Use a pre-trained model like EfficientNet or Inception for feature extraction.
      - Fine-tune the last layers on the Dog vs. Cat dataset.
      - Evaluate accuracy and visualize predictions.
      - **Outcome**: Hands-on experience with both feature extraction and fine-tuning.
  - **Text Classification: Fake News Detection**
    - **Dataset**: Fake News Dataset from Kaggle.
    - **Goal**: Build a model to classify news articles as “real” or “fake.”
    - **Steps**:
      - Use a pre-trained model like DistilBERT.
      - Fine-tune it on the Fake News dataset.
      - Evaluate metrics like accuracy and F1-score.
    - **Outcome**: Understanding how Transfer Learning can be applied to NLP tasks.

Introduction to LLMs (Large Language Model) (beta)

Introduction to LLMs (Large Language Model)

Note: Now you need to create a python file not in a notebook. You can use https://pypi.org/project/pipenv/ which provide python virtual environment

- What are LLMs?
  - Definition: What are large language models, and why are they important?
  - Key Concepts: slackTokens, embeddings, pre-training, and fine-tuning.
- Examples of LLMs
  - OpenAI GPT Models (GPT-3, GPT-4).
  - Google BERT and LaMDA.
  - Meta LLaMA.
  - Open-source models: BLOOM, Falcon, Mistral, etc.
  - Domain-specific LLMs: BioGPT, FinGPT.

- Attention Mechanisms: (Theory, outside office hours)
  - https://www.ibm.com/think/topics/attention-mechanism
- Layer Structures: (Theory, outside office hours)
  - Encoder-decoder architecture vs. decoder-only models.

- **Terminology**
  - LLm, Tokenization, Hallucination, Tokens, Training, Fine tune, Embedding, Vectors, RAG, LoRA, Gen AI, Agents, ReAct, Tools, Prompt
  - GTP, BERT, LLama, Mistral, Claudem, DeepSeek
  - https://python.langchain.com/docs/concepts/
- **Transformer-based architecture**
  - *Goal: How to use huggingface models/dataset*
  - https://huggingface.co/docs/transformers/index
    - *You needs learn by your own (outside office hours)*
  - https://huggingface.co/learn/nlp-course/chapter0/1?fw=pt
    - https://streamlit.io/playground
    - https://huggingface.co/models
    - Chapter:0, 1, 2, 3, 4(outside office hours), 5, 6, 7

- **Prompt Engineering and Prompt Tuning**. *(outside office hours)*
  - https://www.promptingguide.ai/

- Introduction to langchain framework (wip)
  - ChatModels
    - https://python.langchain.com/docs/integrations/chat/
      - Use Ollama proviidocder
      - https://python.langchain.com/docs/integrations/chat/ollama/
    - What is LangChain, and why use it?
    - What is LangChain, and why use it?
    - Using LLaMa**, DeepSeek, or Mistral** with LangChain
    - Mini Project

            - Implement a chatbot that answers basic questions using DeepSeek Model via LangChain.
    - **Vector Embedding & Semantic Search**
        - What is Embedding and why is it important?
        - Different Embedding Models (OpenAI, Hugging Face, DeepSeek, LLaMa)
        - Hands-on: Storing text in a vector database and searching efficiently
        -
        -
    - **Document Loaders & Preprocessing**
        - What are **Document Loaders** in LangChain?
        - Loading **PDF, CSV, Word, HTML, JSON, TXT** files
        - Handling **large documents** efficiently (chunking techniques)
        - Integrating document loaders with **Vector Embedding & RAG**
            - Load **a document file (PDF, TXT, or CSV)** and use **vector embedding** to make it searchable.
            - Combine this with **RAG** to allow users to **ask questions about the document**.
            -
    - https://python.langchain.com/docs/tutorials/#orchestration
    - https://python.langchain.com/docs/how_to/
    - https://python.langchain.com/docs/concepts/
    - Agents/Tools

- Context and Knowledge-Based Training
    - What is context?: Providing relevant inputs for better LLM responses.
    - Fine-tuning vs. using external knowledge bases.
    - Examples: ChatGPT with memory, knowledge-based Q&A.

Mini Project

- Create simple chat application using DeepSeek Model
    - For the ui use https://streamlit.io/playground
    - https://huggingface.co/deepseek-ai/DeepSeek-R1
- Get data from document file and feed to model (semantic search)
    - Then ask model a question to reply answer from the doc
- Convert mysql data to vector and implement product search or product recomandation

- **Prompt Engineering and Prompt Tuning**.
    - https://www.promptingguide.ai/

Projects

- Use Cases
    - Chatbots
        - Building and deploying intelligent, task-specific chat applications.
    - Content Generation
        - Articles, stories, emails, marketing content.

    - Language Translation
        - Creating multi-language models.
    - Document Summarization
        - Generating summaries of long documents.
- AI Agents and Retrieval-Augmented Generation (RAG)
    - What are AI Agents?
        - Models that perform goal-oriented tasks (e.g., LangChain agents).
    - What is RAG?
        - Combining LLMs with search capabilities to retrieve real-time knowledge.
    - Example
        - LLM with vector databases (e.g., Pinecone, Weaviate) to create chatbots with live knowledge.

Introduction to LangChain Framework

- Prompt Chaining
    - Automating multi-step tasks with LLMs.
- Building an AI Agent / Tools
    - Integrating LLMs with APIs (e.g., search engines, calculators).
- Knowledge-Based Applications
    - Integrating LLMs with a vector database using LangChain.