💥 Day 14 – Dive into Neural Networks and TensorFlow Practice

▯ What are Neural Networks?

Neural Networks are a special class of machine learning models inspired by how the brain processes information. These models consist of layers made up of neurons (nodes) that are interconnected. Each connection carries a *weight*, which gets adjusted during training to improve accuracy. Neural networks are particularly good at solving complex tasks like recognizing images, understanding speech, and even powering self-driving vehicles — areas where traditional ML methods often fall short.

🆚 ML vs Neural Networks vs Deep Learning – Key Differences

Aspect	Machine Learning	Neural Networks	Deep Learning
Definition	Uses classic algorithms (e.g., SVM, Decision Trees)	Brain-inspired models with neurons and layers	Advanced NN with many layers (e.g., CNN, RNN)
Data Needs	Small to moderate datasets	Moderate	Large-scale datasets
Performance	Best on structured/tabular data	Good at recognizing patterns	Excels with high-dimensional data like images
Feature Engineering	Manual work needed	Somewhat automatic	Fully automatic through layers
Examples	Logistic Regression, SVM	Perceptron	CNNs, Transformers, RNNs
Computation	Lightweight	Medium	Heavy (often needs GPU/TPU support)

🔗 Implementation – Linear Regression Using TensorFlow

We implemented a basic neural network in TensorFlow to predict **MPG (Miles Per Gallon)** of cars based on **horsepower** — a classic linear regression use case.

🛠 Code Walkthrough:

1. Importing Required Libraries

```
import tensorflow as tf

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns
```

We use TensorFlow for the model, Pandas for data loading and processing, and Matplotlib/Seaborn for visualizing the results.

---

2. Loading the Auto MPG Dataset

```
url = "http://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data"

column_names = [...]
```

raw_dataset = pd.read_csv(url, names=column_names, ...)

We fetch the dataset from UCI's repository, assign meaningful column names, and handle missing entries using na_values.

---

3. Selecting and Normalizing the Features

X = dataset[['horsepower']].astype('float32')

y = dataset[['mpg']].astype('float32')

X = (X - X.mean()) / X.std()

We're using **horsepower** as our only input. Normalization ensures better convergence during training.

---

4. Splitting into Training and Test Sets

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(...)

We split the data 80-20 to evaluate generalization performance.

---

5. Building the Neural Network Model

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(units=1, input_shape=[1])
])
```

A minimal model with one neuron — essentially representing a linear regression.

---

6. Compiling and Training the Model

```
model.compile(optimizer=tf.keras.optimizers.SGD(0.01),
              loss='mean_squared_error')
model.fit(X_train, y_train, epochs=100, verbose=0)
```

We use **Stochastic Gradient Descent** to minimize **MSE** across 100 epochs.

---

7. Evaluating the Model's Performance

loss = model.evaluate(X_test, y_test)

This returns the **mean squared error** on the test set — a direct metric of model accuracy.

---

8. Checking Learned Weights

weights = model.get_weights()

We retrieve the learned slope and bias values to understand the model's internal parameters.

---

9. Visualizing Predictions

y_pred = model.predict(X).flatten()

Plot using seaborn

We overlay the predicted regression line over actual MPG data. This visual confirms how well the model fits the pattern.

📌 **Key Takeaways:**

- We explored the fundamentals of neural networks and how they relate to broader ML and DL concepts.
- Built a simple TensorFlow model using one dense layer to predict car fuel efficiency.
- Applied preprocessing, trained the model, and evaluated it visually and quantitatively.