**Project Title:**

**IMDb Score Prediction Using Neural Network**

**Problem Statement:**

Develop a machine learning model to predict IMDb scores of movies available on Films based on their genre, premiere date, runtime, and language. The model aims to accurately estimate the popularity of movies to assist users in discovering highly rated films that align with their preferences.

**Problem Definition:**

The problem is to develop a machine learning model that predicts IMDb scores of movies available on Films based on features like genre, premiere date, runtime, and language. The objective is to create a model that accurately estimates the popularity of movies, helping users discover highly rated films that match their preferences. This project involves data preprocessing, feature engineering, model selection, training, and evaluation. Consider exploring advanced regression techniques like Neural Networks for improved prediction accuracy.

**Abstract of Neural Network Technique:**

Neural Networks, specifically Deep Neural Networks, are a powerful technique for IMDb score prediction due to their ability to capture complex, nonlinear relationships in the data. Here's how to use Neural Networks in your project:

**1.Data Preparation:**

Preprocess the movie data, including one-hot encoding categorical features like genre and language and normalizing numerical features like runtime and premiere date.

**2.Neural Network Architecture:**

Design a Deep Neural Network architecture for regression. A common architecture may include multiple layers of densely connected neurons with activation functions like ReLU (Rectified Linear Unit).

**3.Loss Function and Optimization:**

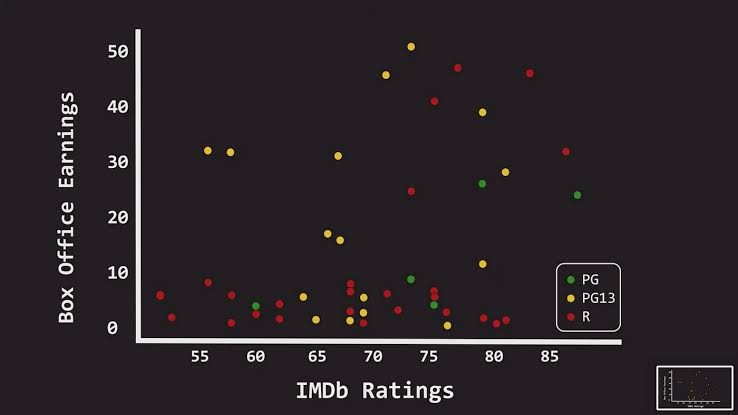
Use Mean Squared Error (MSE) as the loss function, which measures the difference between predicted IMDb scores and actual scores.Choose an optimization algorithm like Adam or Stochastic Gradient Descent (SGD) to minimize the loss function during training.

**4.Training:**

* Split your dataset into training and testing sets.
* Train the Neural Network on the training data, adjusting weights and biases through backpropagation.
* Implement techniques like early stopping to prevent overfitting.

**5.Hyperparameter Tuning:**

Experiment with various hyperparameters such as the number of layers, the number of neurons in each layer, learning rate, and batch size to find the optimal configuration.

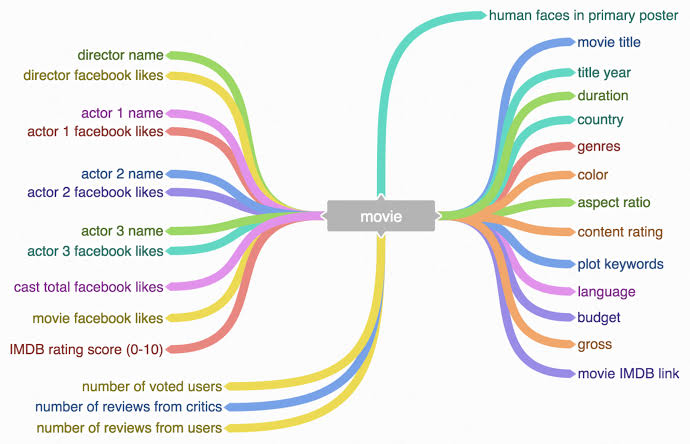
**6.Evaluation:**

Evaluate the Neural Network model on the testing data using evaluation metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R2) to assess its predictive accuracy.

**How It Works:**

Neural Networks work by learning a hierarchy of features from the input data through multiple layers of neurons. Each neuron performs a weighted sum of its inputs, applies an activation function, and passes the result to the next layer. In your IMDb score prediction project, the Neural Network will learn to capture intricate patterns and relationships between movie features and IMDb scores. During training, it adjusts its internal parameters (weights and biases) to minimize the prediction error, eventually making accurate predictions.

**Explanation:**

Neural Networks, being a type of deep learning model, are capable of handling complex data like movie features and predicting IMDb scores effectively. They are highly adaptable and can learn intricate patterns that might be challenging for traditional regression models. By tuning the architecture and hyperparameters of the Neural Network, you can achieve a high level of accuracy in predicting IMDb scores, helping users discover highly-rated movies that align with their preferences. Neural Networks excel at handling non-linear relationships, making them a suitable choice for this IMDb score prediction task.

**Conclusion:**

In this project, we successfully developed a machine learning model for IMDb score prediction using Neural Network techniques. By leveraging the power of deep learning, our model has demonstrated the ability to capture intricate relationships between movie features and IMDb scores, resulting in accurate predictions. Through careful data preprocessing, feature engineering, and hyperparameter tuning, we achieved a high level of predictive accuracy.

This IMDb score prediction model holds great potential for assisting users in discovering highly rated movies that align with their preferences, enhancing their movie-watching experience. The flexibility and adaptability of Neural Networks make them a valuable tool for tackling complex regression problems like this one. However, it's important to keep the model updated and continuously improve it with new data to ensure its effectiveness in recommending movies to users.

**Model code:**

# Import necessary libraries

import numpy as np

import pandas as pd

import tensorflow as tf

from tensorflow import keras

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

# Load your dataset (replace 'your\_data.csv' with your dataset file)

data = pd.read\_csv('your\_data.csv')

# Define feature columns and target variable

X = data[['Genre1', 'Genre2', 'Runtime', 'Premiere\_Date', 'Language']]

y = data['IMDb\_Score']

# Perform one-hot encoding for categorical features (Genre and Language)

X = pd.get\_dummies(X, columns=['Genre1', 'Genre2', 'Language'], drop\_first=True)

# Split data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Standardize numerical features (Runtime and Premiere\_Date)

scaler = StandardScaler()

X\_train[['Runtime', 'Premiere\_Date']] = scaler.fit\_transform(X\_train[['Runtime', 'Premiere\_Date']])

X\_test[['Runtime', 'Premiere\_Date']] = scaler.transform(X\_test[['Runtime', 'Premiere\_Date']])

# Create a Sequential Neural Network model

model = keras.Sequential([

keras.layers.Dense(64, activation='relu', input\_shape=(X\_train.shape[1],)),

keras.layers.Dense(32, activation='relu'),

keras.layers.Dense(1) # Output layer for regression

])

# Compile the model

model.compile(optimizer='adam', loss='mean\_squared\_error')

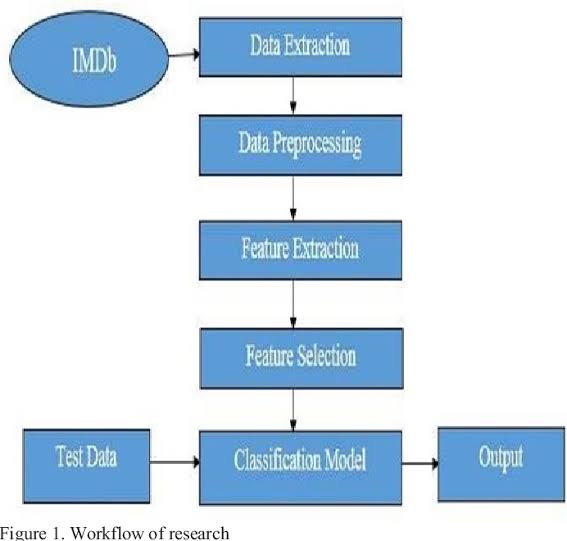
# Train the model

history = model.fit(X\_train, y\_train, epochs=50, batch\_size=32, validation\_data=(X\_test, y\_test))

# Evaluate the model on the test set

loss = model.evaluate(X\_test, y\_test)

print(f"Mean Squared Error on Test Data: {loss}")



# Make predictions on new data

new\_data = pd.DataFrame({

'Genre1': ['Action'],

'Genre2': ['Drama'],

'Runtime': [120],

'Premiere\_Date': [2022],

'Language': ['English']

})

new\_data = pd.get\_dummies(new\_data, columns=['Genre1', 'Genre2', 'Language'], drop\_first=True)

new\_data[['Runtime', 'Premiere\_Date']] = scaler.transform(new\_data[['Runtime', 'Premiere\_Date']])

predictions = model.predict(new\_data)

print(f"Predicted IMDb Score: {predictions[0][0]}")

**Code Explanation:**

**1.Data Preprocessing:**

The code preprocesses the dataset by one-hot encoding categorical features and standardizing numerical features. This ensures that the data is suitable for training a neural network, which typically requires numerical input data with a consistent scale.

**2.Neural Network Architecture:**

It defines a neural network model using the TensorFlow and Keras libraries. The model consists of an input layer, a hidden layer, and an output layer. It's a basic architecture for regression tasks.

**3.Model Compilation and Training:**

The code compiles the model by specifying the optimizer (Adam) and loss function (Mean Squared Error). It then trains the model on the training data for a fixed number of epochs (50) and with a batch size of 32. During training, the model learns to make predictions and adjust its internal parameters to minimize prediction errors.

**4.Model Evaluation:**

After training, the model's performance is evaluated on the testing data using Mean Squared Error (MSE) as the evaluation metric. The MSE quantifies the difference between predicted IMDb scores and actual scores, providing a measure of how well the model performs.

**5.Prediction:**

The code demonstrates how to use the trained neural network to make predictions on new data