AI Problem Statement 2: Titanic Survival Prediction

Objective

The goal is to develop a machine learning model that predicts whether a Titanic passenger survived based on various features provided in the dataset. This project focuses on feature engineering, data preprocessing, and building a neural network for classification.

Data Provided

1. **train.csv**: Training data with passenger details and their survival status.
2. **test.csv**: Testing data without survival status, used for predictions.
3. **gender_submission.csv**: A simple prediction benchmark to guide the submission format.

Data Preprocessing

1. **Libraries Imported**:
   - Essential libraries such as `pandas`, `numpy`, and `sklearn` were used for data manipulation and preprocessing.
   - TensorFlow and Keras were used for constructing the neural network model.
2. **Data Loading**:
   - The training dataset `train.csv` and the benchmark file `gender_submission.csv` were loaded into Pandas DataFrames.

- An initial preview using `.head()` helped explore the structure of the dataset and gain insights into the data.

3. **Handling Missing Values**:
   - Missing values in the `Age` column were filled with the median of the age distribution to avoid introducing bias.
   - Missing values in the `Embarked` column were filled with the most frequent value (mode), ensuring categorical consistency.
   - Any missing values in the `Fare` column were filled using the median fare.
   - A binary feature called `CabinAvailable` was created to indicate whether a cabin was assigned (1) or not (0) to each passenger, as the presence of cabin data might have predictive value.
4. **Feature Engineering**:
   - **Family Size**: Created by adding the number of siblings/spouses (`SibSp`) and parents/children (`Parch`) with the passenger included. This helped model group dynamics, which might affect survival chances.
   - **IsAlone**: A new binary feature was derived from `FamilySize`. If the passenger was alone (`FamilySize = 1`), `IsAlone` was set to 1; otherwise, it was set to 0.
5. **Categorical Encoding**:
   - One-hot encoding was applied to categorical variables such as `Sex`, `Embarked`, and `Pclass` to convert them into a format that can be used by machine learning algorithms.
   - This ensures the machine does not impose ordinal relationships on non-ordinal data like passenger class or gender.

6. **Data Splitting**:
   - The dataset was divided into feature set `X` and target variable `y` (`Survived`).
   - The dataset was then split into training and testing sets with an 80-20 split for model evaluation using `train_test_split()`.

Model Selection and Training

1. **Model Architecture**:
   - A simple neural network was built with the following layers:
     - **Input Layer**: A dense layer with 64 neurons and ReLU activation.
     - **Hidden Layer**: Another dense layer with 32 neurons, also using ReLU activation.
     - **Output Layer**: A single neuron with sigmoid activation to handle binary classification (`Survived` or `Not Survived`).
2. **Training**:
   - The neural network model was compiled using the **Adam optimizer**, known for efficient gradient-based optimization.
   - The loss function was set to **binary cross-entropy** to handle the binary classification task.
   - The model was trained for 50 epochs with a batch size of 32, using 20% of the training data for validation.

Model Evaluation

1. **Accuracy**:
    - The model achieved an accuracy of approximately **79%** on the test data.
2. **Classification Report**:
    - Precision, recall, and F1-score were calculated for both classes (survived and not survived), highlighting the model's performance across key metrics.
3. **Confusion Matrix**:
    - A confusion matrix was generated, detailing the true positives, true negatives, false positives, and false negatives, which provides deeper insight into the model's performance.

Predictions on Test Dataset

1. **Test Data Preprocessing**:
    - The same preprocessing steps (handling missing values, feature engineering, and encoding) were applied to the test dataset to maintain consistency with the training data.
    - Any missing columns in the test dataset were filled with zeros to ensure the test data aligns with the feature set of the training data.
2. **Making Predictions**:
    - Predictions were made using the trained neural network, where each test passenger was classified as either survived or not survived.

3. **Submission File**:
   - The predictions were saved in a file called `submission.csv` containing two columns: `PassengerId` and `Survived`.
   - This file was formatted for submission to Kaggle.

Result on Kaggle

After submitting the predictions to the Kaggle Titanic competition, the model achieved a score of **0.79665**, securing a rank of **736.** This indicates that the neural network performed well compared to many other models submitted in the competition.

This solution showcases the power of feature engineering combined with a neural network model to tackle a classic classification problem. Further tuning and experimentation could push the performance even higher!