Procedure for Fuel Efficiency Prediction

Step 1: Load the Dataset

- Read the dataset from the CSV file using pandas.
- Display basic information to understand its structure.

Step 2: Data Preprocessing

- **Drop unnecessary columns**: Remove `make` and `model` as they are too specific.
- **Separate features and target variable**:
    - Features (`X`): All columns except `combination_mpg`.
    - Target (`y`): `combination_mpg` (fuel efficiency).
- **Identify feature types**:
    - **Numerical features**: `cylinders`, `displacement`, `city_mpg`, `highway_mpg`, `year`.
    - **Categorical features**: `class`, `drive`, `fuel_type`, `transmission`.

Step 3: Define Data Processing Pipelines

- **For numerical features**:
    - Handle missing values using mean imputation.
    - Scale values using `StandardScaler`.
- **For categorical features**:
    - Convert categorical variables into numerical format using `OneHotEncoder`.

Step 4: Build and Train the Model

- Use a `RandomForestRegressor` as the prediction model.
- Split the dataset into **80% training** and **20% testing**.

- Train the model using the preprocessed data.

Step 5: Make Predictions

- Use the trained model to predict `combination_mpg` on the test set.

Step 6: Evaluate Model Performance

- **Mean Absolute Error (MAE)**: Measures average prediction error.
- **R^2 Score**: Measures how well the model explains variance in fuel efficiency.

Step 7: Display Results

- Print MAE and R^2 score to assess model accuracy.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, r2_score

# Load dataset
file_path = (r"c:\Users\balaj\Downloads\car_data (1).csv")
df = pd.read_csv(file_path)

# Drop unnecessary columns
df = df.drop(columns=['make', 'model'])

# Define features and target variable
X = df.drop(columns=['combination_mpg'])
y = df['combination_mpg']

# Identify numerical and categorical features
num_features = ['cylinders', 'displacement', 'city_mpg', 'highway_mpg',
'year']
cat_features = ['class', 'drive', 'fuel_type', 'transmission']
```

```
# Define preprocessing pipelines
num_pipeline = Pipeline([
    ('imputer', SimpleImputer(strategy='mean')),
    ('scaler', StandardScaler())
])

cat_pipeline = Pipeline([
    ('encoder', OneHotEncoder(handle_unknown='ignore'))
])

preprocessor = ColumnTransformer([
    ('num', num_pipeline, num_features),
    ('cat', cat_pipeline, cat_features)
])

# Define model pipeline
model = Pipeline([
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(n_estimators=100,
random_state=42))
])

# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Train model
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate model
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error: {mae}")
print(f"R² Score: {r2}")
```

OUTPUT:

```
PS C:\Users\balaj> & C:/Users/balaj/AppData/Local/Microsoft/WindowsApps/python3.10.exe
"c:/Users/balaj/OneDrive/Desktop/fuel efficiency.py"
Mean Absolute Error: 0.29918181818181816
R² Score: 0.9903228002638288
```