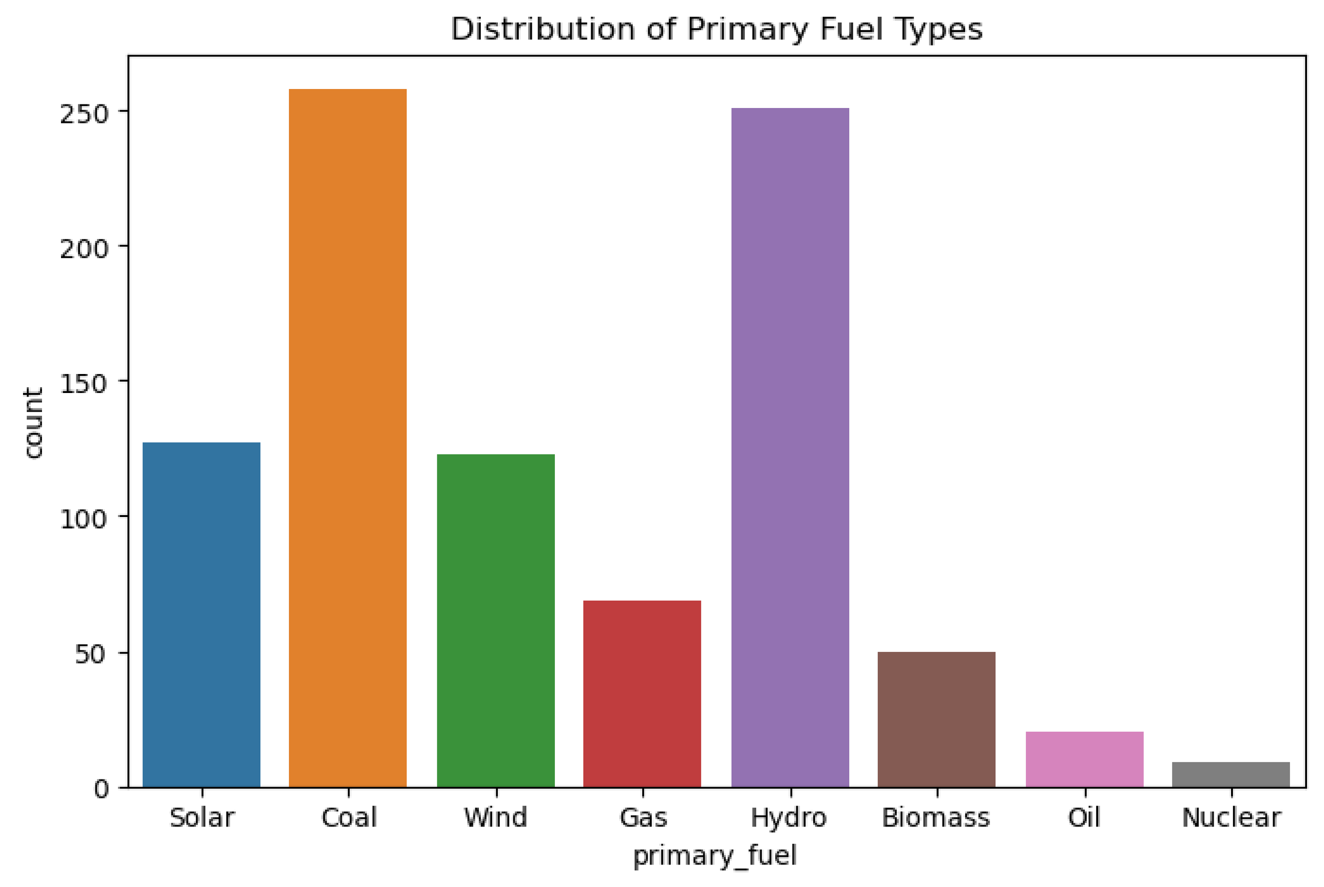
In [1]:
```python
# Importing required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, make_scorer
import joblib
```

In [2]:
```python
# Load the dataset
link = 'https://raw.githubusercontent.com/wri/global-power-plant-database/master/source_databases_csv/database_IND.csv'
data = pd.read_csv(link)
```

In [3]:
```python
# EDA Analysis
# Display basic statistics
print(data.describe())

# Check for missing values
print(data.isnull().sum())
```

```
       capacity_mw    latitude   longitude  other_fuel3  commissioning_year  \
count   907.000000  861.000000  861.000000          0.0          527.000000   
mean    326.223755   21.197918   77.464907          NaN         1997.091082   
std     590.085456    6.239612    4.939316          NaN           17.082868   
min       0.000000    8.168900   68.644700          NaN         1927.000000   
25%      16.725000   16.773900   74.256200          NaN         1988.000000   
50%      59.200000   21.780000   76.719500          NaN         2001.000000   
75%     385.250000   25.512400   79.440800          NaN         2012.000000   
max    4760.000000   34.649000   95.408000          NaN         2018.000000   

       wepp_id  year_of_capacity_data  generation_gwh_2013  \
count      0.0                  519.0                  0.0   
mean       NaN                 2019.0                  NaN   
std        NaN                    0.0                  NaN   
min        NaN                 2019.0                  NaN   
25%        NaN                 2019.0                  NaN   
50%        NaN                 2019.0                  NaN   
75%        NaN                 2019.0                  NaN   
max        NaN                 2019.0                  NaN   

       generation_gwh_2014  generation_gwh_2015  generation_gwh_2016  \
count           398.000000           422.000000           434.000000   
mean           2431.823590          2428.226946          2467.936859   
std            4026.440035          4194.596959          4162.884308   
min               0.000000             0.000000             0.000000   
25%             223.557672           176.381063           188.285252   
50%             801.123775           711.181225           737.205450   
75%            3035.306250          3084.121250          3282.861313   
max           28127.000000         30539.000000         30015.000000   

       generation_gwh_2017  generation_gwh_2018  generation_gwh_2019  \
count           440.000000           448.000000                  0.0   
mean           2547.759305          2600.804099                  NaN   
std            4196.991169          4314.880456                  NaN   
min               0.000000             0.000000                  NaN   
25%             177.874930           193.378250                  NaN   
50%             817.977250           751.644375                  NaN   
75%            3275.690475          3143.535900                  NaN   
max           35116.000000         35136.000000                  NaN   

       estimated_generation_gwh  
count                       0.0  
mean                        NaN  
std                         NaN  
min                         NaN  
25%                         NaN  
50%                         NaN  
75%                         NaN  
max                         NaN  
country                       0
country_long                  0
name                          0
gppd_idnr                     0
capacity_mw                   0
latitude                     46
longitude                    46
primary_fuel                  0
other_fuel1                 709
other_fuel2                 906
other_fuel3                 907
commissioning_year          380
owner                       565
source                        0
url                           0
geolocation_source           19
wepp_id                     907
year_of_capacity_data       388
generation_gwh_2013         907
generation_gwh_2014         509
generation_gwh_2015         485
generation_gwh_2016         473
generation_gwh_2017         467
generation_gwh_2018         459
generation_gwh_2019         907
generation_data_source      458
estimated_generation_gwh    907
dtype: int64
```

In [4]:
```python
# Visualize the distribution of the target variable (primary fuel)
plt.figure(figsize=(8, 5))
sns.countplot(x='primary_fuel', data=data)
plt.title('Distribution of Primary Fuel Types')
plt.show()
```



In [5]:
```python
# Preprocessing and Feature Engineering
# Handling missing values
data.fillna(0, inplace=True)
```

In [6]:
```python
# Encoding categorical variables
label = LabelEncoder()
data['primary_fuel_encoded'] = label.fit_transform(data['primary_fuel'])
```

In [7]:
```python
# Feature selection
features = ['latitude', 'longitude', 'commissioning_year', 'generation_gwh_2013', 'generation_gwh_2014', 'generation_gwh_2015']
target = 'primary_fuel_encoded'
```

In [8]:
```python
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(data[features], data[target], test_size=0.2, random_state=42)
```

In [9]:
```python
# Build/Test Multiple Models
# Random Forest Model
model = RandomForestRegressor()
model.fit(X_train, y_train)
predictions = model.predict(X_test)
```

In [10]:
```python
# Check for overfitting/underfitting
train_predictions = model.predict(X_train)
print(f"Train MAE: {mean_absolute_error(y_train, train_predictions)}")
print(f"Test MAE: {mean_absolute_error(y_test, predictions)}")

# Cross-validation
cv_mae = cross_val_score(model, data[features], data[target], scoring=make_scorer(mean_absolute_error), cv=5)
print(f"Cross-Validation MAE: {np.mean(cv_mae)}")
```

```
Train MAE: 0.3046011494252873
Test MAE: 0.7303663003663005
Cross-Validation MAE: 0.8454618207759091
```

In [11]:
```python
# Hyperparameter Tuning
param_grid = {'n_estimators': [50, 100, 200], 'max_depth': [None, 10, 20], 'min_samples_split': [2, 5, 10]}
grid_search = GridSearchCV(estimator=model, param_grid=param_grid, scoring=make_scorer(mean_absolute_error), cv=5)
grid_search.fit(data[features], data[target])
```

Out[11]:

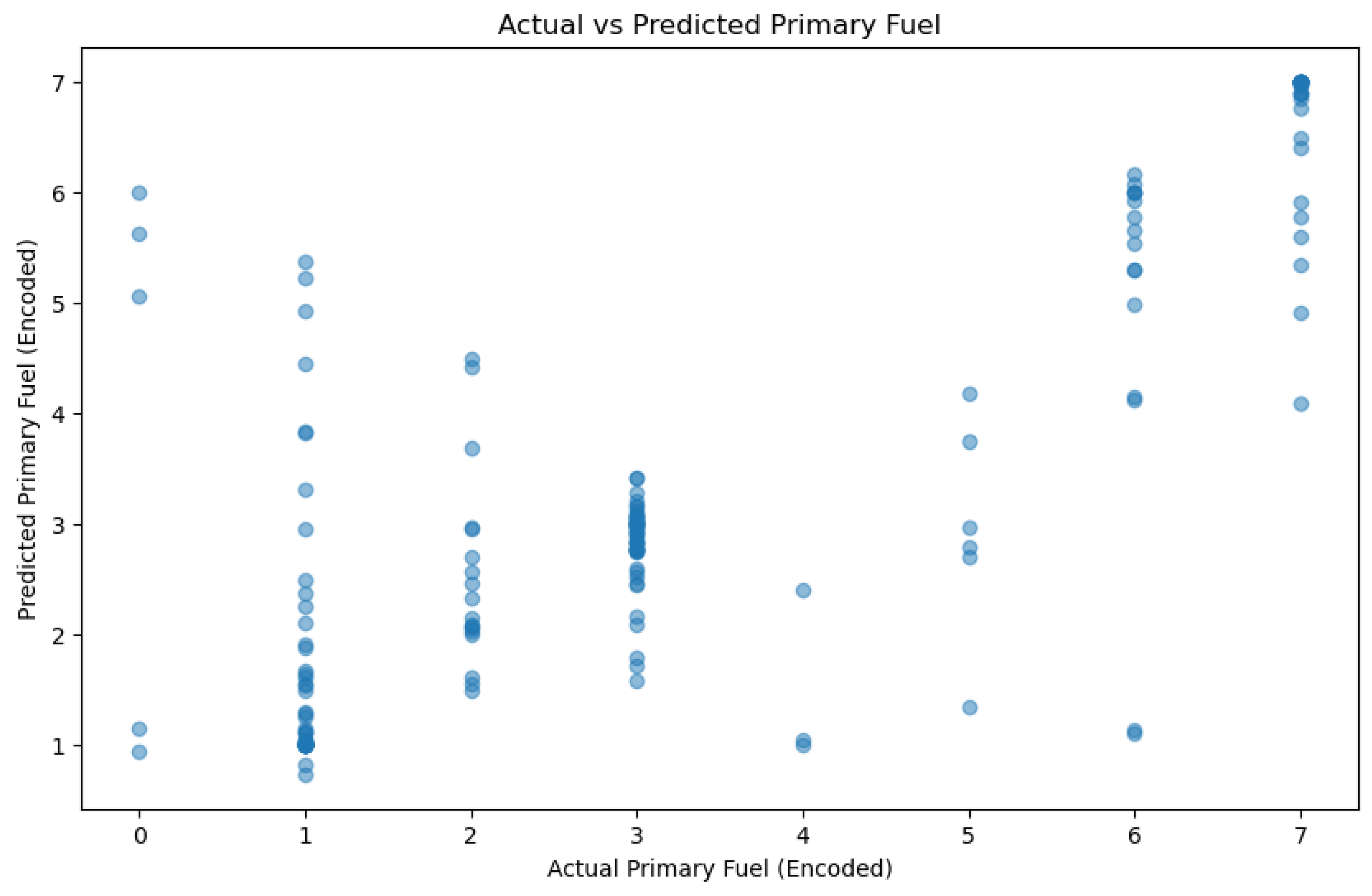
```
▸ GridSearchCV
▸ estimator: RandomForestRegressor
  ▸ RandomForestRegressor
```

In [12]:
```python
# Select the Best Model
best_model = grid_search.best_estimator_

# Save the Best Model for Production
joblib.dump(best_model, 'best_model.pkl')
```

Out[12]:

```
['best_model.pkl']
```

In [13]:
```python
# Additional Visualizations
# Scatter Plot for Predicted vs Actual
plt.figure(figsize=(10, 6))
plt.scatter(y_test, predictions, alpha=0.5)
plt.xlabel('Actual Primary Fuel (Encoded)')
plt.ylabel('Predicted Primary Fuel (Encoded)')
plt.title('Actual vs Predicted Primary Fuel')
plt.show()
```



Importing Libraries: You start by importing the necessary Python libraries for data analysis, visualization, and machine learning.

Loading the Dataset: You load a dataset related to the global power plant database in India from a given link using pandas.

Exploratory Data Analysis (EDA): You conduct exploratory data analysis by displaying basic statistics, checking for missing values, and visualizing the distribution of the target variable (primary fuel).

Preprocessing and Feature Engineering: You handle missing values by filling them with zero. Categorical variables are encoded using LabelEncoder. You select specific features and the target variable for your model.

Train-Test Split: You split the dataset into training and testing sets using the train_test_split function.

Building and Testing Multiple Models: You use a Random Forest Regressor, fit the model, and make predictions. You check for overfitting/underfitting by comparing the Mean Absolute Error (MAE) on both training and testing sets. Cross-validation is performed to obtain a more robust evaluation.

Hyperparameter Tuning: You perform a grid search to find the best hyperparameters for the Random Forest model.

Selecting the Best Model: The best model is selected based on the results of the hyperparameter tuning.

Saving the Model: The best model is saved using the joblib library for potential use in production.

Explanation for Model Selection: A brief explanation is provided on how the best model is selected based on the lowest Mean Absolute Error (MAE) obtained during cross-validation.

Additional Visualization: You create a scatter plot to visualize the relationship between the actual and predicted primary fuel types.