```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt

# Step 1: Load the data
data = pd.read_excel('1553768847_housing.xlsx')
print(data.head())

# Extract input (X) and output (Y) data from the dataset
X = data.drop('median_house_value', axis=1)
Y = data['median_house_value']

# Step 2: Handle missing values
X.fillna(X.mean(), inplace=True)

# Step 3: Encode categorical data
X_encoded = pd.get_dummies(X, columns=['ocean_proximity'])

# Step 4: Split the dataset
X_train, X_test, Y_train, Y_test = train_test_split(X_encoded, Y, test_size=0.2,
random_state=42)

# Step 5: Standardize data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Step 6: Perform Linear Regression
linear_reg = LinearRegression()
linear_reg.fit(X_train_scaled, Y_train)
Y_pred_linear = linear_reg.predict(X_test_scaled)
linear_rmse = mean_squared_error(Y_test, Y_pred_linear, squared=False)
print('Linear Regression RMSE:', linear_rmse)

# Step 7: Perform Decision Tree Regression
dt_reg = DecisionTreeRegressor()
dt_reg.fit(X_train_scaled, Y_train)
Y_pred_dt = dt_reg.predict(X_test_scaled)
dt_rmse = mean_squared_error(Y_test, Y_pred_dt, squared=False)
```

```
print('Decision Tree Regression RMSE:', dt_rmse)

# Step 8: Perform Random Forest Regression
rf_reg = RandomForestRegressor()
rf_reg.fit(X_train_scaled, Y_train)
Y_pred_rf = rf_reg.predict(X_test_scaled)
rf_rmse = mean_squared_error(Y_test, Y_pred_rf, squared=False)
print('Random Forest Regression RMSE:', rf_rmse)

# Step 9: Bonus exercise - Linear Regression with one independent variable
X_train_income = X_train_scaled[:, X_encoded.columns.tolist().index('median_income')]
X_test_income = X_test_scaled[:, X_encoded.columns.tolist().index('median_income')]

linear_reg_income = LinearRegression()
linear_reg_income.fit(X_train_income.reshape(-1, 1), Y_train)
Y_pred_income = linear_reg_income.predict(X_test_income.reshape(-1, 1))

# Plot the fitted model for training data and test data
plt.scatter(X_train_income, Y_train, color='blue', label='Training Data')
plt.scatter(X_test_income, Y_test, color='green', label='Test Data')
plt.plot(X_test_income, Y_pred_income, color='red', linewidth=2, label='Fitted Model')
plt.xlabel('Median Income')
plt.ylabel('Median House Value')
plt.legend()
plt.show()
```