In [1]:
```
import pandas as pd
import statsmodels.api as sm
```

1.) Import Data from FRED

In [2]:
```
data = pd.read_csv("TaylorRuleData.csv", index_col = 0)
```

In [3]:
```
data.index = pd.to_datetime(data.index)
```

In [4]:
```
data = data.dropna()
```

2.) Do Not Randomize, split your data into Train, Test Holdout

In [5]:
```
split1 = int(len(data) * .6)
split2 = int(len(data) * .9)
data_in = data[:split1]
data_out = data[split1:split2]
data_hold = data[split2:]
```

In [6]:
```
X_in = data_in.iloc[:,1:]
y_in = data_in.iloc[:,0]
X_out = data_out.iloc[:,1:]
y_out = data_out.iloc[:,0]
X_hold = data_hold.iloc[:,1:]
y_hold = data_hold.iloc[:,0]
```

In [7]:
```
# Add Constants
X_in = sm.add_constant(X_in)
X_out =  sm.add_constant(X_out)
X_hold =  sm.add_constant(X_hold)
```

3.) Build a model that regresses FF~Unemp, HousingStarts, Inflation

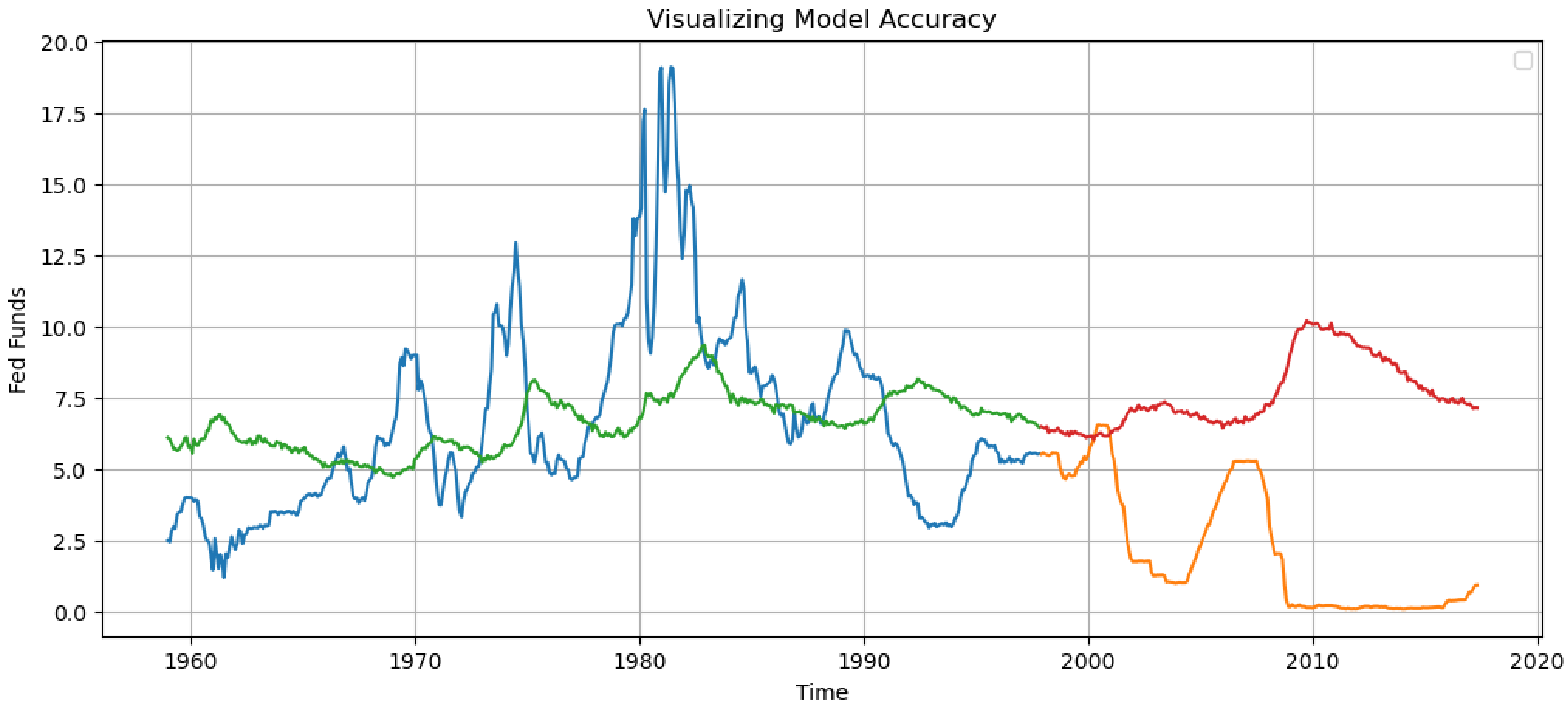
In [9]:
```
model1 = sm.OLS(y_in,X_in).fit()
```

4.) Recreate the graph fro your model

In [10]:
```
import matplotlib.pyplot as plt
```

In [11]:
```
plt.figure(figsize = (12,5))
```

```
###
plt.plot(y_in)
plt.plot(y_out)
plt.plot(model1.predict(X_in))
plt.plot(model1.predict(X_out))

###

plt.ylabel("Fed Funds")
plt.xlabel("Time")
plt.title("Visualizing Model Accuracy")
plt.legend([])
plt.grid()
plt.show()
```



"All Models are wrong but some are useful" - 1976 George Box

5.) What are the in/out of sample MSEs

In [12]:
```
from sklearn.metrics import mean_squared_error
```

In [13]:
```
in_mse_1 = mean_squared_error(model1.predict(X_in),y_in)
out_mse_1 = mean_squared_error(model1.predict(X_out),y_out)
```

In [14]:
```
print("Insample MSE : ", in_mse_1)
print("Outsample MSE : ", out_mse_1)
```

```
Insample MSE :  10.071422013168643
Outsample MSE :  40.36082783566727
```

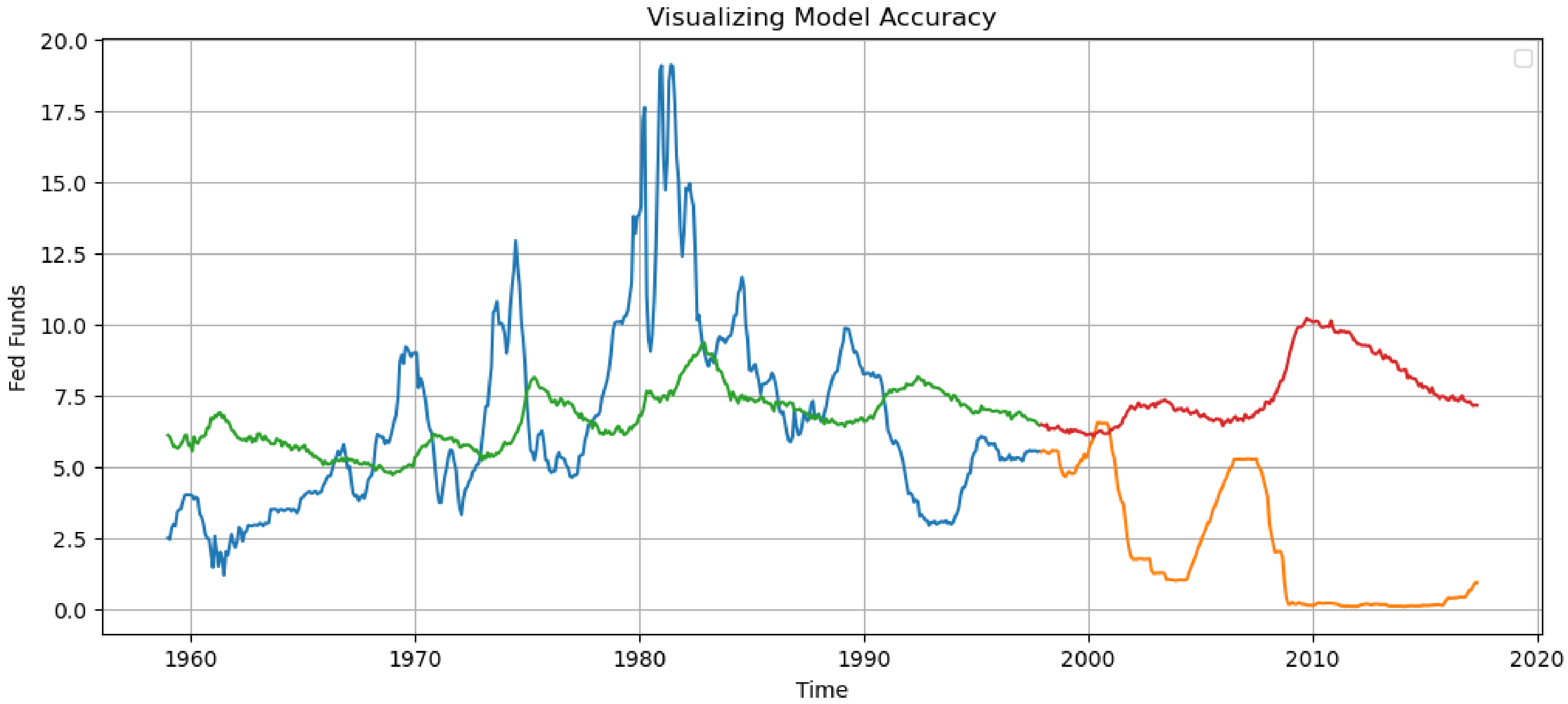
6.) Using a for loop. Repeat 3,4,5 for polynomial degrees 1,2,3

In [22]:
```
from sklearn.preprocessing import PolynomialFeatures
```

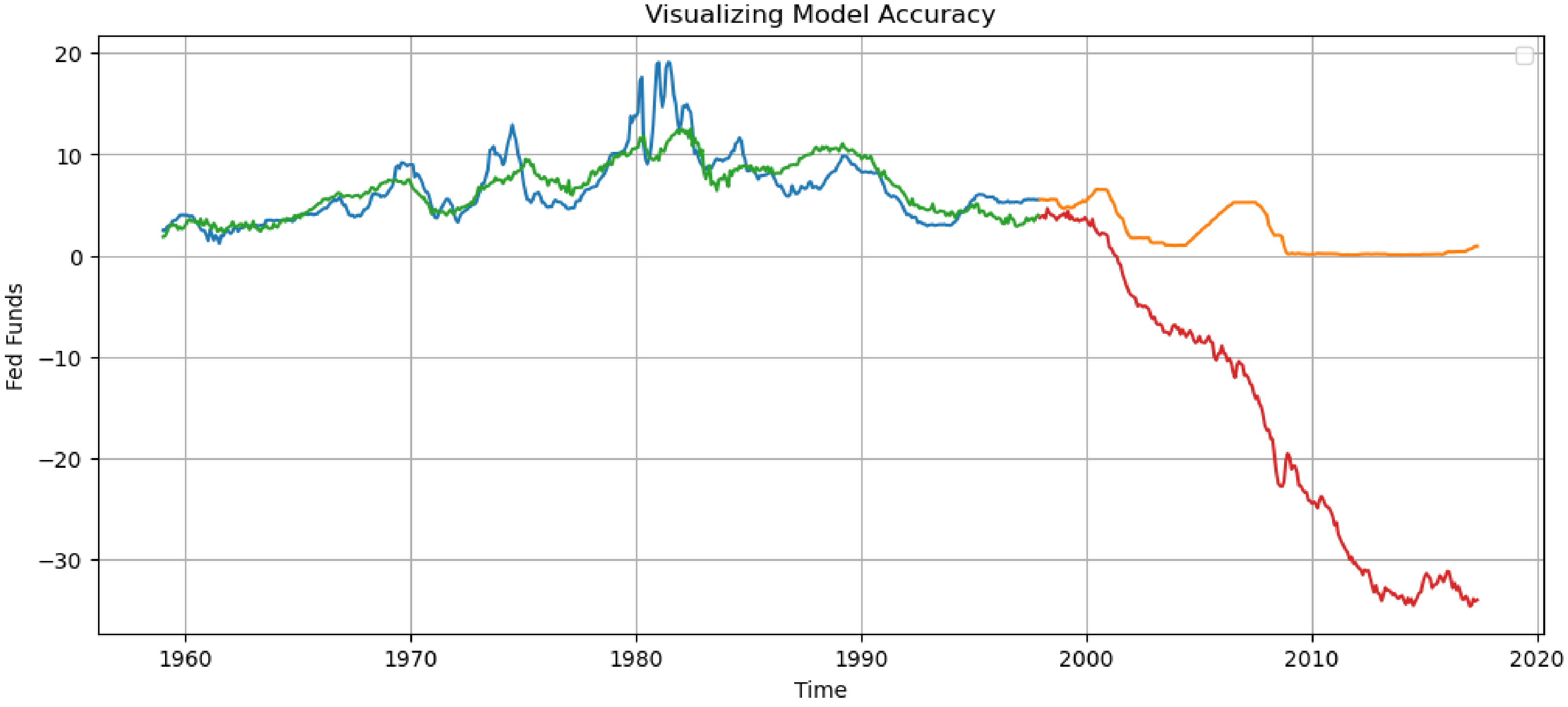
In [23]:
```
max_degrees = 3
```

In [25]:
```
for degrees in range(1, max_degrees+1):
    print('degrees: ', degrees)
    poly = PolynomialFeatures(degree = degrees)
    X_in_poly = poly.fit_transform(X_in)
    X_out_poly = poly.fit_transform(X_out)
    ###Q3
    model1 = sm.OLS(y_in, X_in_poly).fit()
    ###Q4
    plt.figure(figsize = (12,5))

    in_preds = model1.predict(X_in_poly)
    in_preds = pd.DataFrame(in_preds, index = y_in.index)
    out_preds = model1.predict(X_out_poly)
    out_preds = pd.DataFrame(out_preds, index = y_out.index)

    plt.plot(y_in)
    plt.plot(y_out)
    plt.plot(in_preds)
    plt.plot(out_preds)

    plt.ylabel("Fed Funds")
    plt.xlabel("Time")
    plt.title("Visualizing Model Accuracy")
    plt.legend([])
    plt.grid()
    plt.show()

    in_mse_1 = mean_squared_error(y_in, in_preds)
    out_mse_1 = mean_squared_error(y_out, out_preds)
    print("Insample MSE : ", in_mse_1)
    print("Outsample MSE : ", out_mse_1)
```

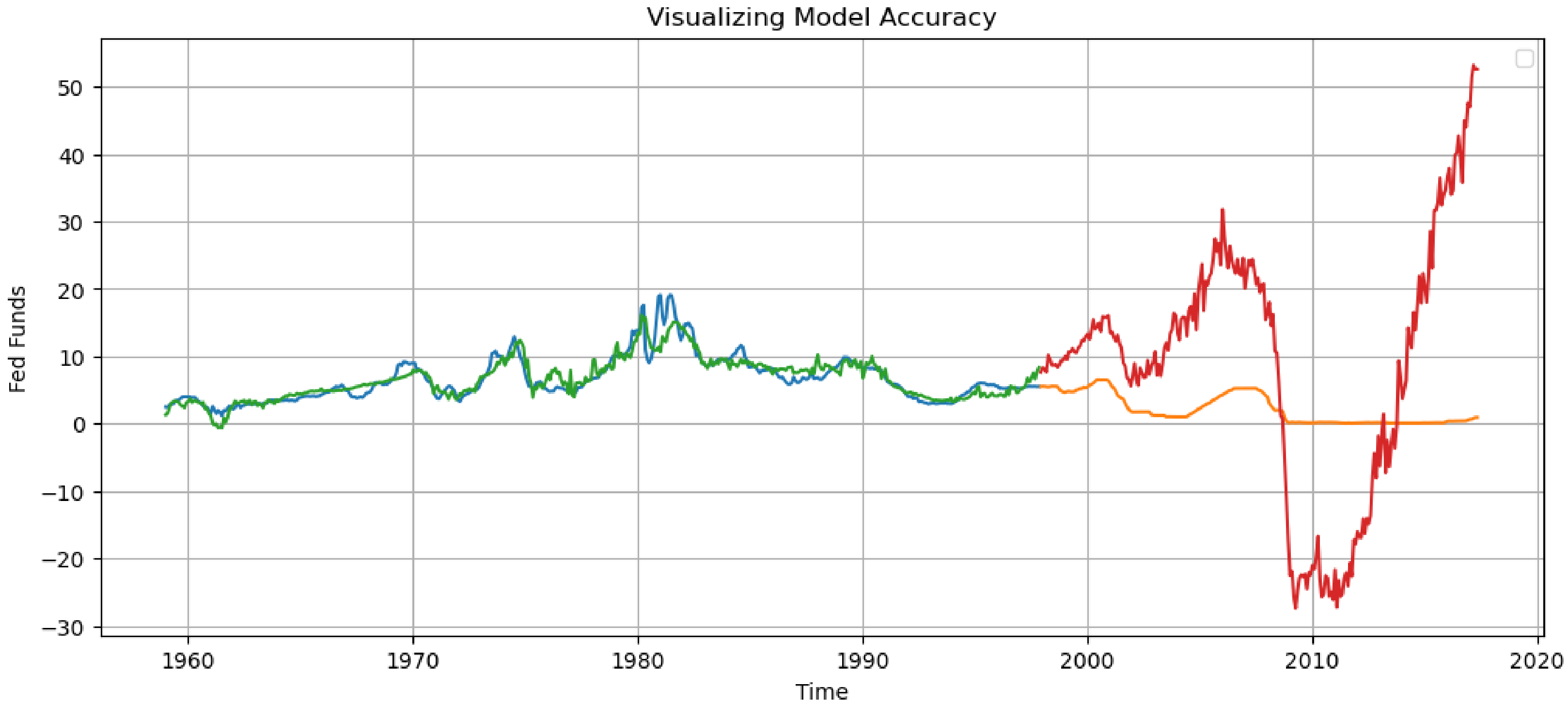
```
degrees:  1
```




```
Insample MSE :  10.07142201316864
Outsample MSE :  40.36082783566782
degrees:  2
```




```
Insample MSE :  3.8634771392760685
Outsample MSE :  481.4465099294859
degrees:  3
```




```
Insample MSE :  1.872363626650644
Outsample MSE :  371.7680409381005
```

7.) State your observations :

As the degrees of fitting increases, the model tends to overfit, and it fits in sample data perfectly while it has high variance toward out of the sample data.

In [ ]: