Sec 1 Homework #1

January 12, 2024

[23]:
```
import pandas as pd
import statsmodels.api as sm
from fredapi import Fred
import statsmodels.stats.api as sms
import statsmodels.api as sm
```

1 1.) Import Data from FRED

[24]:
```
data = pd.read_csv("TaylorRuleData.csv", index_col = 0)
```

[25]:
```
data.index = pd.to_datetime(data.index)
```

[26]:
```
data = data.dropna()
```

[27]:
```
data
```

[27]:

	FedFunds	Unemployment	HousingStarts	Inflation
1959-01-01	2.48	6.0	1657.0	29.010
1959-02-01	2.43	5.9	1667.0	29.000
1959-03-01	2.80	5.6	1620.0	28.970
1959-04-01	2.96	5.2	1590.0	28.980
1959-05-01	2.90	5.1	1498.0	29.040
…	…	…	…	…
2023-07-01	5.12	3.5	1451.0	304.348
2023-08-01	5.33	3.8	1305.0	306.269
2023-09-01	5.33	3.8	1356.0	307.481
2023-10-01	5.33	3.8	1359.0	307.619
2023-11-01	5.33	3.7	1560.0	307.917

[779 rows x 4 columns]

[28]:
```
#from fredapi import Fred
#fred = Fred(api_key='e67121f20c15f2fa5dc000b94fb6bdb0')

# Pull data
#fed_funds = fred.get_series('FEDFUNDS')  # Example series ID for Federal Funds␣
 ↪Rate
```

```
#unemployment = fred.get_series('UNRATE')  # Unemployment Rate
#housing_starts = fred.get_series('HOUST')  # Housing Starts
#inflation = fred.get_series('CPIAUCSL')  # CPI for All Urban Consumers

# Combine data into a single DataFrame
#data = pd.DataFrame({
#    'FedFunds': fed_funds,
#    'Unemployment': unemployment,
#    'HousingStarts': housing_starts,
#    'Inflation': inflation
#})
```

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

[29]:
```
split_1 = int(len(data) * 0.6)
split_2 = int(len(data) * 0.9)
data_in = data[:split_1]
data_out = data[split_1:split_2]
data_hold = data[split_2:]
```

[30]:
```
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]
```

[31]:
```
# Add Constants
X_in = sm.add_constant(X_in)
X_out =  sm.add_constant(X_out)
X_hold =  sm.add_constant(X_hold)
```

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

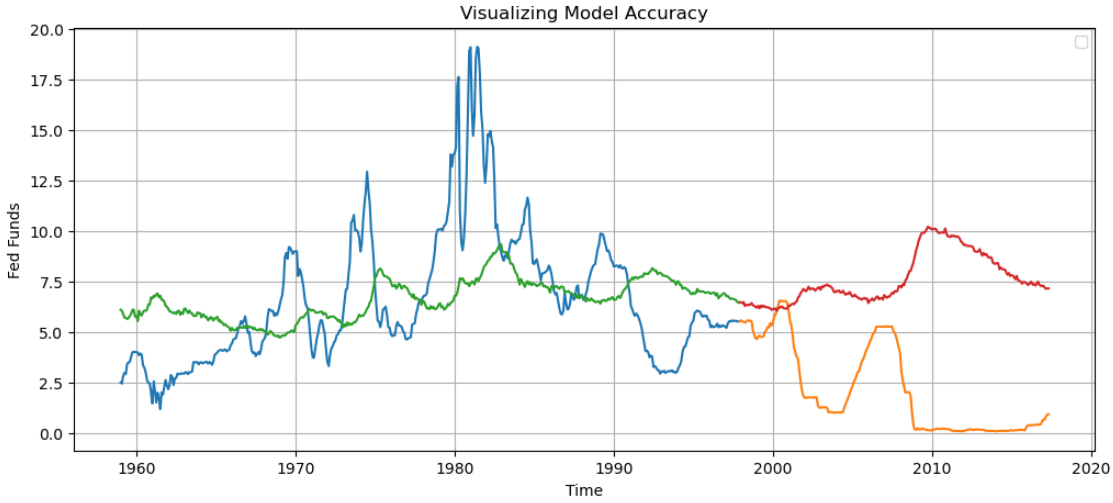
[32]:
```
model1 = sm.OLS(y_in, X_in).fit()
```

4 4.) Recreate the graph fro your model

[33]:
```
import matplotlib.pyplot as plt
```

[34]:
```
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()
```



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

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

[35]:
```
from sklearn.metrics import mean_squared_error
```

[36]:
```
in_mse_1 = mean_squared_error(model1.predict(X_in), y_in)
out_mse_1 = mean_squared_error(model1.predict(X_out), y_out)
```

[37]:
```
print("Insample MSE : ", in_mse_1)
print("Outsample MSE : ", out_mse_1)
```

```
Insample MSE :  10.071422013168641
Outsample MSE :  40.36082783566856
```

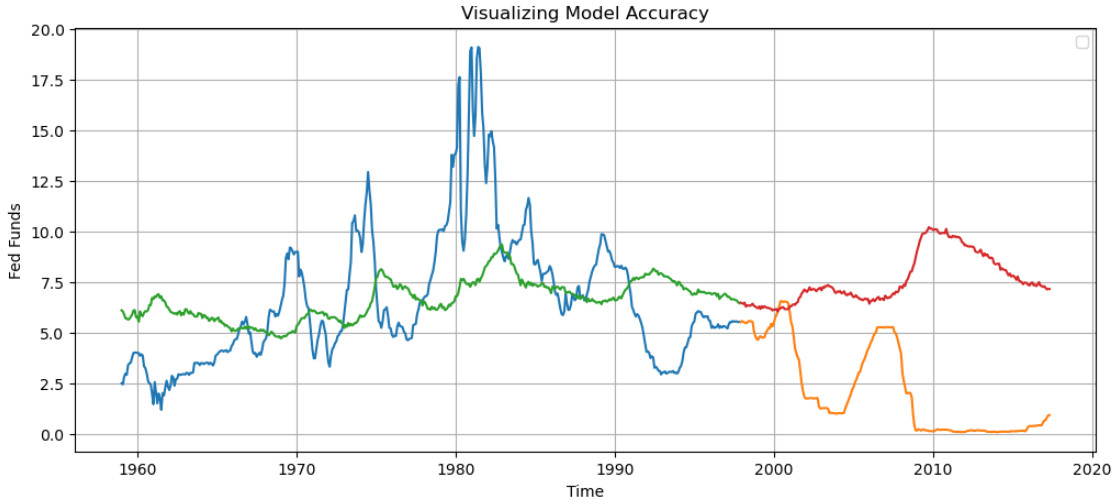
6 6.) Using a for loop. Repeat 3,4,5 for polynomial degrees 1,2,3

[38]:
```
from sklearn.preprocessing import PolynomialFeatures
```

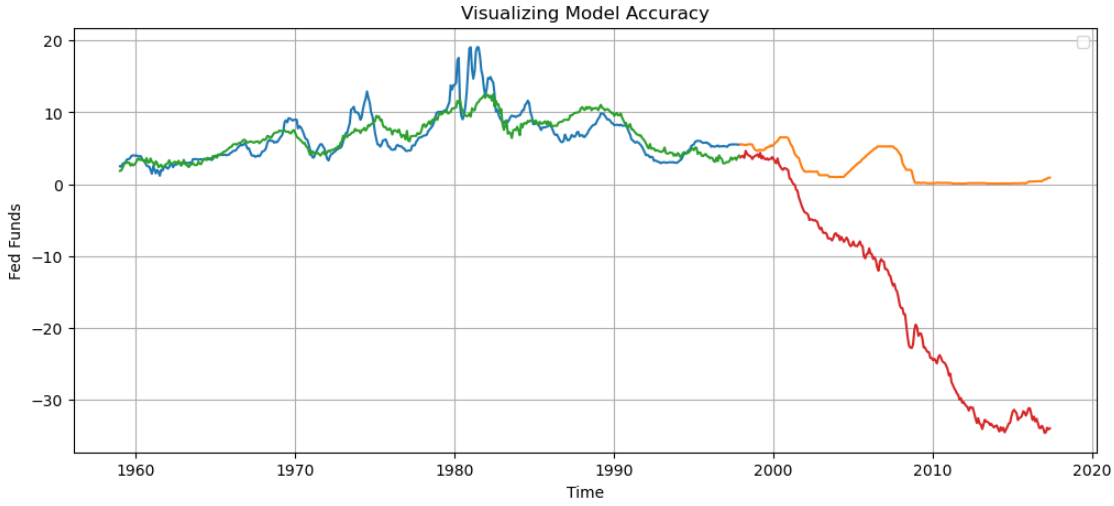
[39]:
```
max_degrees = 3
```

[41]:
```
for degrees in range(1, max_degrees+1):

    print("DEGREE : " , degrees)

    poly = PolynomialFeatures(degree = degrees)
    X_in_poly = poly.fit_transform(X_in)
    X_out_poly = poly.fit_transform(X_out)

    model1 = sm.OLS(y_in, X_in_poly).fit()



    plt.figure(figsize = (12,5))

    pred_in = model1.predict(X_in_poly)
    pred_in = pd.DataFrame(pred_in, index = y_in.index)

    pred_out = model1.predict(X_out_poly)
    pred_out = pd.DataFrame(pred_out, index = y_out.index)

    ###
    plt.plot(y_in)
    plt.plot(y_out)
    plt.plot(pred_in)
    plt.plot(pred_out)
    ###

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

    in_mse_1 = mean_squared_error(model1.predict(X_in_poly), y_in)
    out_mse_1 = mean_squared_error(model1.predict(X_out_poly), y_out)

    print("Insample MSE : ", in_mse_1)
    print("Outsample MSE : ", out_mse_1)
```

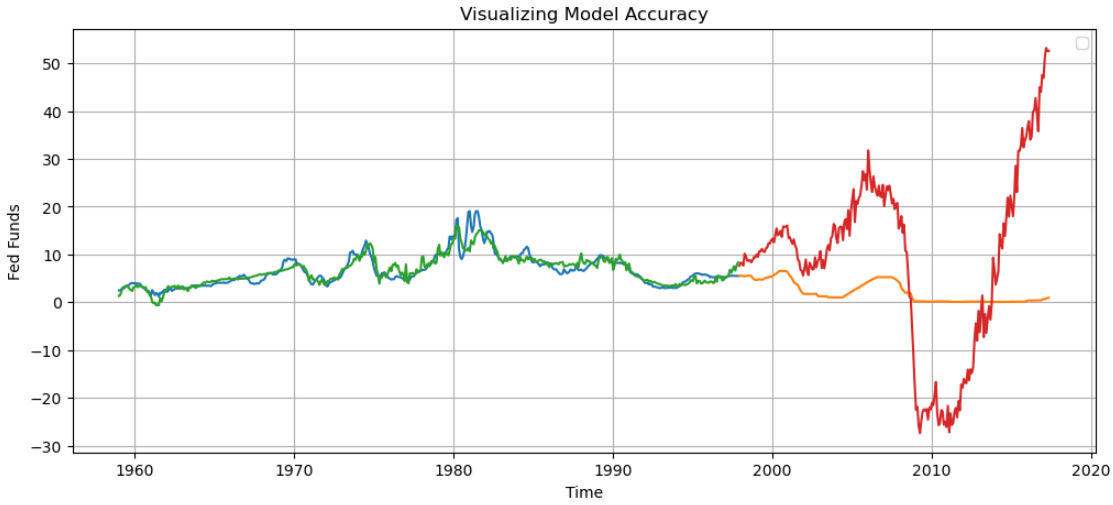
```
DEGREE :  1
```




```
Insample MSE :  10.071422013168641
Outsample MSE :  40.36082783566674
DEGREE :  2
```




```
Insample MSE :  3.863477139276067
Outsample MSE :  481.44650990363203
DEGREE :  3
```




```
Insample MSE :  1.872363627194615
Outsample MSE :  371.76618900618945
```

7 7.) State your observations :

- Insample MSE keeps decreasing as you add a regresssor with higher degrees
- If insample MSE keeps decreasing, that means its improving
- Outsample MSE performs inefficiently as you add regressors with higher degrees
- Models with degrees 2 and 3 are overfitting, therefore having high variance
- Model with degree 3 has the lowest Insample MSE