Prof. Dr. Natalie Packham
LV-Nr. 425093

Summer term 2019

Computational Finance and FinTech – Problem Set 4 with solutions

Use the data in `dax_and_spx.csv` in the following exercises. This data set contains one year of daily DAX and S&P500 index values.

Exercise 1. The goal of this exercise is to analyse the linear relationship between the DAX and the S&P 500.
Load the data set into a `DataFrame` and add columns with the log returns and create a scatter plot. Perform a linear regression of the DAX log-returns on the S&P log-returns. Comment on the ability of the model to forecast DAX returns. Is the model statistically significant?

Solution 1.

```
# import statements
import numpy as np
import pandas as pd
from pylab import mpl, plt
from arch import arch_model
import datetime as dt
import statsmodels.api as sm
plt.style.use('seaborn')

# read the data into a dataframe and insert columsn with log returns
df = pd.read_csv('./_src/dax_spx.csv', index_col=0)
df.insert(2, 'SPX', 100 * np.log(df['SPX Index'] / df['SPX Index'].shift(1)))
df.insert(3, 'DAX', 100 * np.log(df['DAX Index'] / df['DAX Index'].shift(1)))
df.dropna(inplace=True)

# scatter plot
plt.scatter(df['SPX'], df['DAX'])

# regression model
Y=df['DAX']
X=df['SPX']
X = sm.add_constant(X)
model = sm.OLS(Y,X)
results = model.fit()

print(results.summary())
```

Output:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    DAX   R-squared:                       0.213
Model:                            OLS   Adj. R-squared:                  0.210
Method:                 Least Squares   F-statistic:                     71.32
Date:                Fri, 03 May 2019   Prob (F-statistic):           2.08e-15
Time:                        15:43:11   Log-Likelihood:                -326.21
No. Observations:                 265   AIC:                             656.4
Df Residuals:                     263   BIC:                             663.6
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.0234      0.051     -0.458      0.647      -0.124       0.077
SPX            0.4558      0.054      8.445      0.000       0.350       0.562
==============================================================================
Omnibus:                       15.608   Durbin-Watson:                   2.233
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               26.223
Skew:                          -0.351   Prob(JB):                     2.02e-06
Kurtosis:                       4.372   Cond. No.                         1.07
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

The R^2 of 0.213 means that 21.3% of the variance in DAX returns are captured by SP returns. This would be considered too low for forecasting.
The SPX coefficient is statistically significant at the 1% level, so there is evidence that a linear relationship between the two indices is present.

Exercise 2. Given is the following data set:

u	v
26.00	6.25
11.00	5.50
7.50	5.00
18.50	6.00
29.00	6.50
15.00	6.00
30.00	6.50
31.50	6.50

Assume that the following non-linear model applies:

$$v = a + b \ln(u).$$

(a) Determine the regression parameters.

(b) Plot the data and the model with estimated coefficients.

Solution 2.

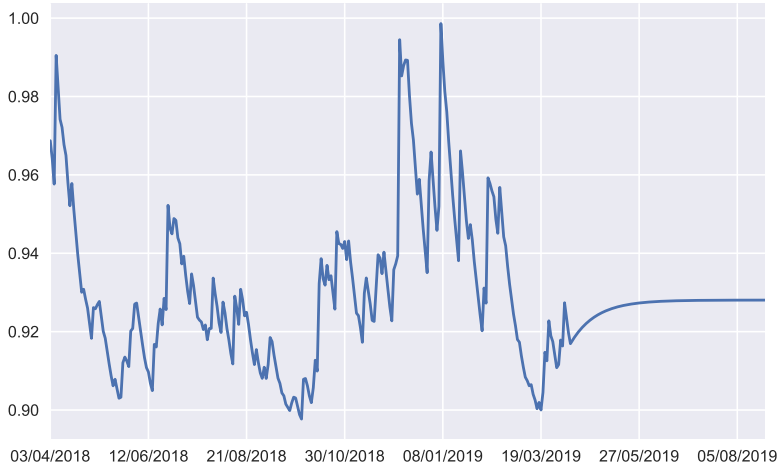
```
import statsmodels.api as sm

y=v
x = np.log(u)
x = sm.add_constant(x)

model = sm.OLS(y,x)
result = model.fit()
a=result.params[0]
b=result.params[1]
print("a = {:.2f}, b = {:.2f}".format(a,b))

plt.figure()
plt.scatter(u,v)
u2=np.arange(min(u), max(u), .5)
plt.plot(u2, a + b*np.log(u2))
```

Exercise 3. The goal of this exercise is to fit a GARCH model to the DAX and forecast DAX volatilities.
Load the data set into a `DataFrame` and add columns with the log returns and create a scatter plot. Fit a GARCH model on the DAX returns. What do the parameters of the GARCH model tell you about the variability in the DAX volatility? Next, forecast daily volatilities for 100 days and produce a plot of both the historical and the forecasted volatility. It should look similar to this:



Solution 3.

```
import numpy as np
import pandas as pd
from pylab import mpl, plt
from arch import arch_model
import datetime as dt
import statsmodels.api as sm
plt.style.use('seaborn')

df = pd.read_csv('./_src/dax_spx.csv', index_col=0)\end{solution}
df.insert(2, 'SPX', 100 * np.log(df['SPX Index'] / df['SPX Index'].shift(1)))
df.insert(3, 'DAX', 100 * np.log(df['DAX Index'] / df['DAX Index'].shift(1)))
df.dropna(inplace=True)

DAX_dem = df['DAX'] - df['DAX'].mean();  # de-mean the data (i.e., make mean
                                         zero)

am = arch_model(DAX_dem, mean = 'Zero')
res = am.fit()

print(res.summary()) # prints the GARCH output

nf = 100 # number of forecast periods
# forecasts in a pandas Series object
f = np.sqrt(res.forecast(horizon=nf).variance.iloc[-1])

# set the index in the forecast time series to the next 100 business days
start = dt.datetime.strptime(df.index[-1], '%d/%m/%Y')
dates = pd.date_range(start, periods = nf, freq = 'B')
f.index=dates.strftime('%d/%m/%Y')

# concatenate the historical volatilities with the forecasted volatilities
vols = res.conditional_volatility.append(f)

plt.figure(figsize=(8,5))
vols.plot()
plt.savefig('dax_vol.pdf')
```