Modeling Returns

Kannan Singaravelu, CQF

June 2023

1 Normal Distribution

Normal Distribution, also known as Gaussian distribution is one of the most widely assumed distribution in Data Science. A normal distribution has a bell-shaped density curve described by its mean μ and standard deviation σ. The density curve is symmetrical, centered about its mean, with its spread determined by its standard deviation.

The probability distribution function of a normal density curve with mean μ and standard deviation σ at a given point x is given by:

$$f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

1.1 Import Libraries

We'll import the required libraries that we'll use in this example.

[ ]:
```
# Import Pandas, Numpy
import pandas as pd
import numpy as np

# Import matplotlib for visualization
import matplotlib
import matplotlib.pyplot as plt

# Plot settings
plt.style.use('ggplot')
matplotlib.rcParams['figure.figsize'] = [12.0,8.0]
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['lines.linewidth'] = 2.0

# ignore warnings
import warnings
warnings.filterwarnings('ignore')
```

1.2 Load GBPUSD Data

[ ]:
```
# Load the CSV file
df = pd.read_csv('data/gbpusd.csv', index_col=0, parse_dates=True,␣
↪dayfirst=True)['2011':'2023']
df
```

[ ]:
```
# Visualize the plot to verify the data
plt.plot(df.index, df['Adj Close']);
```

[ ]:
```
# Verify the datetime format
df.index
```

[ ]:
```
# get last 300 index values
df.index[-300:]
```

1.3 Calculate return

[ ]:
```
# Calculate returns and add it to existing DataFrame as a column
df['Return'] = df['Adj Close'].pct_change().fillna(0)

# Get first 5 rows
df.head()
```

1.4 Calculate Mean & Sigma

[ ]:
```
# Calculate mean and sigma
mu = np.mean(df['Return'])
sigma = np.std(df['Return'])

mu, sigma
```

[ ]:
```
def zscore(returns):
    zs = (returns - np.mean(returns))/np.std(returns)
    return zs
```

1.5 Calculate Scaled Returns

[ ]:
```
# Calculate the scaled return : zscore
df['Scaled_Return'] = df['Return'].apply(lambda x: (x-mu)/sigma)

# Check the output
df.head()
```

1.6 Calculate Bin Range

[ ]:
```
# Calculate minimum and maximum bin range
sr_min = np.min(df['Scaled_Return'])
sr_max = np.max(df['Scaled_Return'])

sr_min, sr_max
```

[ ]:
```python
# Define bins - x
x = np.linspace(sr_min, sr_max, 200)

# Calculate normal probability density function - y
y = (1/np.sqrt(2*np.pi)*np.exp(-0.5*x**2))

# Plot histogram of scaled returns
plt.hist(df['Scaled_Return'], bins=200, density=True, label='Empirical', alpha=1)

# Plot norm pdf
plt.plot(x,y, color='green', label='Normal', alpha=1)

# Set x and y axis limits
# plt.xlim(-4,4)
# plt.ylim(0,0.7)

# Set title
plt.title('Empirical Vs Normal Distribution')

# Set legends
plt.legend()
plt.show()
```

1.7 References

- Numpy Documentation
- Scipy Documentation
- Paul Wilmott (2007), Paul Wilmott introduces Quantitative Finance
- Python Resources

June 2023, Certificate in Quantitative Finance.