Lab : Communication Engineering by Python

Created by	Borhan
Last edited time	@January 6, 2025 4:21 PM
Tag	Year 3 Term 1

Theoretical Concepts

Entropy:

$$H(X) = -\sum P(x_i)\log_2 P(x_i)$$

Joint Entropy:

$$H(X,Y) = -\sum_{x\epsilon X}\sum_{y\epsilon Y} P(x,y)\log_2 P(x,y)$$

Mutual Information

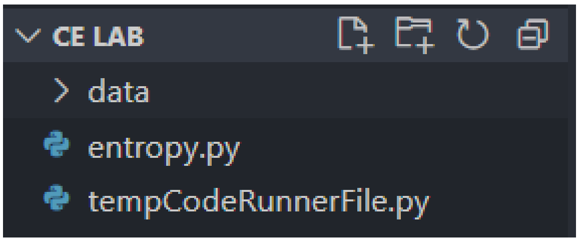
$$M(X,Y) = H(X) + H(Y) - H(X,Y)$$

Packages

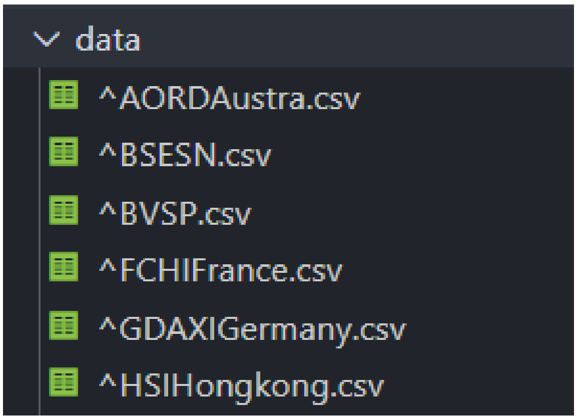
- Pandas
- NumPy
- SciPy

```
pip install pandas numpy scipy
```

My File Structure



CE LAB
data
entropy.py
tempCodeRunnerFile.py



data
^AORDAustra.csv
^BSESN.csv
^BVSP.csv
^FCHIFrance.csv
^GDAXIGermany.csv
^HSIHongkong.csv

Code

Reading a CSV file

```
import pandas as pd

data = pd.read_csv("data/^TWII.csv");
print(data.head());
```

Calculating Information Content

```
import pandas as pd
import numpy as np

data = pd.read_csv("data/^TWII.csv")
column = data['Open']
value, counts = np.unique(column, return_counts=True)

prob = counts / len(column)

for i in range(len(value)) :
    information = -np.log2(prob[i])
    print("Symbol: ", value[i], "Information: ", information)
```

Calculating Entropy

```
import pandas as pd
import numpy as np

data = pd.read_csv("data/^TWII.csv")
column = data['Open']

values, counts = np.unique(column, return_counts=True)
counts = counts[np.nonzero(counts)]

prob = []

for i in range(len(counts)):
    prob.append(counts[i] / len(column));
# Alternative : prob = counts / len(column)

def entropy(prob) :
    entropy = 0
    for i in range(len(prob)):
        entropy -= prob[i] * np.log2(prob[i]);
    return entropy

print(entropy(prob))
```

Calculating Entropy by Using Bins

```
import pandas as pd
import numpy as np

data = pd.read_csv("data/^TWII.csv")
column = data['Open']
column = column.dropna()
```

```
def entropy(column, bins) :
    binned_dist = np.histogram(column, bins)[0]
    probs = binned_dist / np.sum(binned_dist)
    probs = probs[np.nonzero(probs)]
    entropy = 0
    for i in range(len(probs)):
        entropy -= probs[i] * np.log2(probs[i]);
    return entropy

print(entropy(column, 100))
```

NOTE :

```
    for i in range(len(prob)):
        entropy -= prob[i] * np.log2(prob[i]);

    This portion may be written,

    entropy = -np.sum(probs * np.log2(probs));
```

Joint Entropy

```
import pandas as pd
import numpy as np

data = pd.read_csv("data/^TWII.csv")

#OPEN
X = data['Open']
X = X.dropna()

# Close
Y = data['Close']
Y = Y.dropna()

def jointEntropy(X, Y, bins) :
    binned_XY = np.histogram2d(X, Y, bins)[0]
    probsXY = binned_XY / np.sum(binned_XY)
    probsXY = probsXY[np.nonzero(probsXY)]
    jointEntropy = -np.sum(probsXY * np.log2(probsXY));
    return jointEntropy

print(jointEntropy(X, Y, 100))
```

Mutual Information

```
import pandas as pd
import numpy as np

data = pd.read_csv("data/^TWII.csv")

#X
X = data['Open']
X = X.dropna()

#Y
Y = data['Close']
Y = Y.dropna()

def entropy(X, bins):
    binned_dist = np.histogram(X, bins)[0]
    probs = binned_dist / np.sum(binned_dist)
    probs = probs[np.nonzero(probs)]
    entropy = -np.sum(probs * np.log2(probs))
    return entropy

def jointEntropy(X, Y, bins) :
    binned_XY = np.histogram2d(X, Y, bins)[0]
    probsXY = binned_XY / np.sum(binned_XY)
    probsXY = probsXY[np.nonzero(probsXY)]
    jointEntropy = -np.sum(probsXY * np.log2(probsXY));
    return jointEntropy

def mutualInformation(X, Y, bins) :
    return entropy(X, bins) + entropy(Y, bins) - jointEntropy

print(mutualInformation(X, Y, 100))
```

Using Library function

```
import pandas as pd
import numpy as np
from scipy.stats import entropy
from sklearn.metrics import mutual_info_score

data = pd.read_csv("data/^TWII.csv")
bins = 100;

#X
X = data['Open']
X = X.dropna()
X_binned = np.histogram(X, bins)[0];

#Y
Y = data['Close']
Y = Y.dropna()
Y_binned = np.histogram(Y, bins)[0];

print(entropy(X_binned, base=2))
print(entropy(Y_binned, base=2))

mutual_info = mutual_info_score(X_binned,Y_binned)
print(mutual_info)
```