Q 1) Consider the following sentences related to data mining theory, and assume that each of the below sentences corresponds to a document dd:

- Data refers to characteristics that are collected through observation.
- A dataset can be viewed as a collection of objects.
- Data objects are described by a number of attributes.
- An attribute is a characteristic or feature of an object.

A. Construct and display the document-term matrix for the above documents. Remove all stop words (here consider as stop words: articles, prepositions, conjunctions, pronouns, and common verbs) and punctuation marks; convert any plural nouns/adjectives to their singular form; and convert verbs to the present tense and first person singular form, before you construct the matrix. [1 mark out of 5]

B. Using the above constructed document-term matrix, calculate the inverse document frequency $\mathrm{idf}(w)$idf(w) for all words ww you have identified from question 1(a). [0.5 marks out of 5]

Answer :- Document:

- Data refers to characteristics that are collected through observation.
- A dataset can be viewed as a collection of objects.
- Data objects are described by a number of attributes.
- An attribute is a characteristic or feature of an object.

Document-term matrix:

	characteristic	object	attribute	refer	Data	Dataset	Describe	Number
Data refers to characteristics that are collected through observation	1	1	0	1	1	0	0	0
A dataset can be viewed as a collection of objects	0	0	0	0	0	1	0	0
Data objects are described by a number of attributes	0	1	1	0	1	0	1	1
An attribute is a characteristic or feature of an object	1	1	1	0	0	0	0	0

Inverse document frequency for all terms:

IDF(t) = log_e(Total number of word in document / Number of term t in it)

Total number of words in document= 38

characteristic= log(38/2) = 1.14

object= lg(38/3)= 1.10 (P.S- log = lg)

attribute= lg(38/2) = 1.14

refer= lg(38/1)= 1.57

Data = lg(38/2) = 1.14

Dataset= lg(38/1)= 1.57

Describe= lg(38/1)= 1.57

Number= lg(38/1)= 1.57

View= lg(38/1)= 1.57

Collect= lg(38/2) = 1.14

feature= lg(38/1)= 1.57

Q 2) Consider a timeseries yy were we have obtained values of the timeseries for the following times tt, as shown in the below table. Using linear interpolation, calculate the values yy of the timeseries for times t=3t=3 and t=5t=5. [0.5 marks out of 5]

tt	**1**	**4**	**6**
yy	2	8	5

Answer :- The **formula** for the **linear interpolation** process is **y−y1=((y2−y1)/(x2−x1))×(x−x1)**, where x is the known value, y is the unknown value, x1 and y1 are the coordinates that are below the known x value, and x2 and y2 are the coordinates that are above the x value.

y−y1=((y2−y1)/(x2−x1))×(x−x1)
from the question : t: 1 4 6
y: 2 8 5

y= y1+ ((y2−y1)/(x2−x1))×(x−x1)

therefore for t=3: putting x=3 and x1=1, y1=2, x2=4,y2=8 then:

y= 2+((8-2)/(4-1))*(3-1)= 2+((6/3)*2)=2+(4)=6 is the ANS

therefor for t=5 : putting x=5 and x1=4, y1=8, x2=6,y2=5 then:

y=8+((5-8)/(6-4))*(5-4)=8+((-3/2)*1)=8-1.5=6.5 is the ANS

Q 1) Consider the following timeseries y={0.1,0.15,0.2,0.2,0.3,0.4,0.25,0.6,0.5}. Perform binning on the timeseries using k=3 values per bin, and show the resulting timeseries after binning.

Answer:- The timeseries on which we have to perform binning as per the question is y = {0.1,0.15,0.2,0.3,0.2,0.4,0.6,0.25.0.5}. In the question it is asked to perform the process of binning on this timeseries data using the value k=3 value on each bin. We perform binning on a set of data to bin it into a group of small intervals in order to make the data short.

So after binning the resultant timeseries would be:

Multiplying each value by 3. So, y = {0.3, 0.45, 0.6, 0.6, 0.9, 1.2, 0.75, 1.8, 1.5}

Q 2) Load CSV file "timeseries.csv", which contains a univariate timeseries. Once loaded, convert the timeseries into a numpy array and use the numpy flatten() function to ensure that the loaded timeseries is one-dimensional. Compute the Discrete Fourier Transform (DFT) of the timeseries, and display plots for both the original timeseries and the magnitude of its DFT. How many predominant frequency components does the timeseries have?

Answer:- import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

data=pd.read_csv(r"timeseries.csv")

data.head()

data=np.array(data)

data.flatten('F')

import numpy as np

import matplotlib.pyplot as plt

from pyts.approximation import DiscreteFourierTransform

# Parameters

n_samples, n_timestamps = 100, 48

# Toy dataset

```
rng = np.random.RandomState(41)
X = rng.randn(n_samples, n_timestamps)

# DFT transformation
n_coefs = 30
dft = DiscreteFourierTransform(n_coefs=n_coefs, norm_mean=False,
                               norm_std=False)
X_dft = dft.fit_transform(X)

# Compute the inverse transformation
if n_coefs % 2 == 0:
    real_idx = np.arange(1, n_coefs, 2)
    imag_idx = np.arange(2, n_coefs, 2)
    X_dft_new = np.c_[
        X_dft[:, :1],
        X_dft[:, real_idx] + 1j * np.c_[X_dft[:, imag_idx],
                                        np.zeros((n_samples, ))]
    ]
else:
    real_idx = np.arange(1, n_coefs, 2)
    imag_idx = np.arange(2, n_coefs + 1, 2)
    X_dft_new = np.c_[
        X_dft[:, :1],
        X_dft[:, real_idx] + 1j * X_dft[:, imag_idx]
    ]
X_irfft = np.fft.irfft(X_dft_new, n_timestamps)

# Show the results for the first time series
plt.figure(figsize=(6, 4))
plt.plot(X[0], 'o--', ms=4, label='Original')
plt.plot(X_irfft[0], 'o--', ms=4, label='DFT - {0} coefs'.format(n_coefs))
plt.legend(loc='best', fontsize=10)
plt.xlabel('Time', fontsize=14)
plt.title('Discrete Fourier Transform', fontsize=16)

plt.show()
```

Q 3) Using the daily births dataset from this lab tutorial, smooth the timeseries using trailing moving average smoothing and a window size that corresponds to one week; then replace any NaN values with zeros. Perform timeseries forecasting using the smoothed dataset in order to predict daily births for the first 5 days of 1960, using the models below. Show your forecasting results. [0.75 marks out of 5]

- AR model with $p=2$p=2
- ARMA model with $p=2$p=2 and $q=2$q=2

Answer:-
```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

data=pd.read_csv(r"births.csv")

data.head()

# Perform trailing moving average smoothing

rolling = data.rolling(window=3) # using a window of 3 samples: t, t-1, t-2

rolling_mean = rolling.mean()

print(rolling_mean.head(10))


# plot original and transformed dataset

data.plot(figsize=(15,4),title='original timeseries')

rolling_mean.plot(color='red', figsize=(15,4),title='smoothed timeseries')

data.replace({'NaN':0})

data.head()


from statsmodels.tsa.ar_model import AR

model = AR(data)

model_fit = model.fit()

print('Lag: %s' % model_fit.k_ar)

print('Coefficients: %s' % model_fit.params)
```

```
from statsmodels.tsa.arima_model import ARIMA

model = ARIMA(data,order=(2,0,2))

results_MA = model.fit()

print('Lag: %s' % results_MA.k_ar)

print('Coefficients: %s' % results_MA.params)
```

Q 4) Using a similar process used in section 1 of this lab notebook, perform document clustering using k-means on the following wikipedia articles: supervised learning, unsupervised learning, semi-supervised learning, association rule learning, anomaly detection, cluster analysis, dimensionality reduction, regression analysis, statistical classification, data mining, data warehouse, online analytical processing. As with section 1, use the elbow metric to find an appropriate number of clusters. Discuss and display the document clustering results.

Answer:-

```
import pandas as pd

import wikipedia


articles=['supervised learning','unsupervised learning','semi-supervised learning','association rule learning','anomaly detection','cluster analysis','dimensionality reduction','regression analysis','statistical classification','data mining','data warehouse','data mining','data warehouse','online analytical processing']

wiki_lst=[]

title=[]


# Load wikipedia articles

for article in articles:

  print("loading content: ",article)

  wiki_lst.append(wikipedia.page(article).content)

  title.append(article)


# Printing the first retrieved entry

print(wiki_lst[0])


from sklearn.feature_extraction.text import TfidfVectorizer


vectorizer = TfidfVectorizer(stop_words={'english'})

X = vectorizer.fit_transform(wiki_lst) # Create tf-idf feature of the wikipedia dataset
```

```
print(X.shape) # Print dimensions of tf-idf feature

import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

Sum_of_squared_distances = []
K = range(2,7)

for k in K:
   km = KMeans(n_clusters=k, max_iter=200, n_init=10)
   km = km.fit(X)
   Sum_of_squared_distances.append(km.inertia_)

plt.plot(K, Sum_of_squared_distances, 'bx-')
plt.xlabel('k')
plt.ylabel('Sum_of_squared_distances')
plt.title('Elbow Method')
plt.show()

# Fit k-means model with k=3
true_k = 3
model = KMeans(n_clusters=true_k, init='k-means++', max_iter=200, n_init=10)
model.fit(X)

# Print list of documents and associated clusters
labels=model.labels_
wiki_cl=pd.DataFrame(list(zip(title,labels)),columns=['title','cluster'])
print(wiki_cl.sort_values(by=['cluster']))
```

From the Elbow Plot it will be k=3