Open in Colab

In [33]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from urllib.request import urlopen
from bs4 import BeautifulSoup
```

ECS766 Coursework 4 - Elliot Linsey

Q1.A

- \<!DOCTYPE html>, this tag declares to the browser what type of information to expect. All html documents must start with this tag.
- \<html>, this tag represents the root of the html document and is the tag in which all of the information within the document is contained (apart from the \<!DOCTYPE html> tag).
- \<body>, this tag defines where the contents such as headings, paragraphs, tables, images and anything else that the user will actually see or interact with are placed within the document. There can be only one body within an html document but it can be split into sections with div elements.
- \<h1>, this tag is used to define the heading of the html document and is usually used as the main title. This is because it is the largest heading available in comparison to lesser headings like h2, h3 and so on.
- \<p>, this tag defines a paragraph element which is a block of text. If the \<h> tag is the title, then the \<p> tag will usually be the paragraph below it.
- \<table>, this tag defines a table, it must contain one of more of the \<tr>, \<td> and \<th> elements.
- \<thead>, this defines the head of the table which is where the column headings will be located.
- \<tbody>, this defines the body of the table where the data will be located.
- \<tr>, this indicates a table row which can contain either data \<td> or headings \<th> in this instance.
- \<th>, this indicates a table heading which will contain the column headings, usually a child of \<tr>.
- \<td>, this indicates table data that will be placed within a row, usually a child of \<tr> also.

Q1.B

In [34]:
```python
url = 'http://eecs.qmul.ac.uk/~emmanouilb/income_table.html'
html = urlopen(url)
```

In [35]:
```python
soup = BeautifulSoup(html, 'lxml')
```

In [36]:
```python
th = soup.find_all('th')
headers = []
for header in th:
  headers.append(header.get_text())
headers
```

Out[36]:
```
['Region', 'Age', 'Income', 'Online Shopper']
```

In [37]:
```python
td = soup.find_all('td')
rows = []
for row in td:
  rows.append(row.get_text())
rows = np.array(rows)
rows = np.reshape(rows,(10,4))
rows
```

Out[37]:
```
array([['India', '49', '86400', 'No'],
       ['Brazil', '32', '57600', 'Yes'],
       ['USA', '35', '64800', 'No'],
       ['Brazil', '43', '73200', 'No'],
       ['USA', '45', '', 'Yes'],
       ['India', '40', '69600', 'Yes'],
       ['Brazil', '', '62400', 'No'],
       ['India', '53', '94800', 'Yes'],
       ['USA', '55', '99600', 'No'],
       ['India', '42', '80400', 'Yes']], dtype='<U6')
```

In [38]:
```python
df = pd.DataFrame(rows,columns=headers)
df
```

Out[38]:

	Region	Age	Income	Online Shopper
0	India	49	86400	No
1	Brazil	32	57600	Yes
2	USA	35	64800	No
3	Brazil	43	73200	No
4	USA	45		Yes
5	India	40	69600	Yes
6	Brazil		62400	No
7	India	53	94800	Yes
8	USA	55	99600	No
9	India	42	80400	Yes

Q2

In [39]:
```python
url = 'http://eecs.qmul.ac.uk/postgraduate/programmes/'
html = urlopen(url)
```

In [40]:
```python
soup = BeautifulSoup(html, 'lxml')
```

In [41]:
```python
th = soup.find_all('th')
headers = []
for header in th:
  headers.append(header.get_text())
headers
```

Out[41]:
```
['Postgraduate degree programmes', 'Part-time(2 year)', 'Full-time(1 year)']
```

In [42]:
```python
td = soup.find_all('td')
rows = []
for row in td:
  rows.append(row.get_text())
rows = np.array(rows)
rows = np.reshape(rows,(14,3))
rows2 = []
for x in rows:
  rows2.append(np.char.strip(x,'\xa0'))
rows = rows2
#rows
```

In [43]:
```python
df2 = pd.DataFrame(rows, columns=headers)
df2
```

Out[43]:

	Postgraduate degree programmes	Part-time(2 year)	Full-time(1 year)
0	Advanced Electronic and Electrical Engineering	H60C	H60A
1	Artificial Intelligence	I4U2	I4U1
2	Big Data Science	H6J6	H6J7
3	Computer Games		I4U4
4	Computer Science	G4U2	G4U1
5	Computer Science by Research	G4Q2	G4Q1
6	Computing and Information Systems	G5U6	G5U5
7	Data Science and Artificial Intelligence by Co...		I4U5
8	Electronic Engineering by Research	H6T6	H6T5
9	Internet of Things (Data)	I1T2	I1T0
10	Machine Learning for Visual Data Analytics	H6JZ	H6JE
11	Sound and Music Computing	H6T4	H6T8
12	Telecommunication and Wireless Systems	H6JD	H6JA
13	Digital and Technology Solutions (Apprenticeship)	I4DA	

In [44]:
```python
links = []
for row in soup.find_all('td'):
  #print(row.find('a'))
  try:
    links.append(row.find('a').get('href'))
  except AttributeError:
    links.append('')
```

In [45]:
```python
links = np.array(links)
links = np.reshape(links,(14,3))
#links
```

A note to marker, the course table (on the website) has changed since we were set this coursework. Below is the table as displayed from 16/12/2021.

In [46]:
```python
df3 = pd.DataFrame(links, columns=['drop', 'Part-time Link', 'Full-time Link'])
df3 = df3.drop('drop',axis=1)
df3 = df2.join(df3)
df3
```

Out[46]:

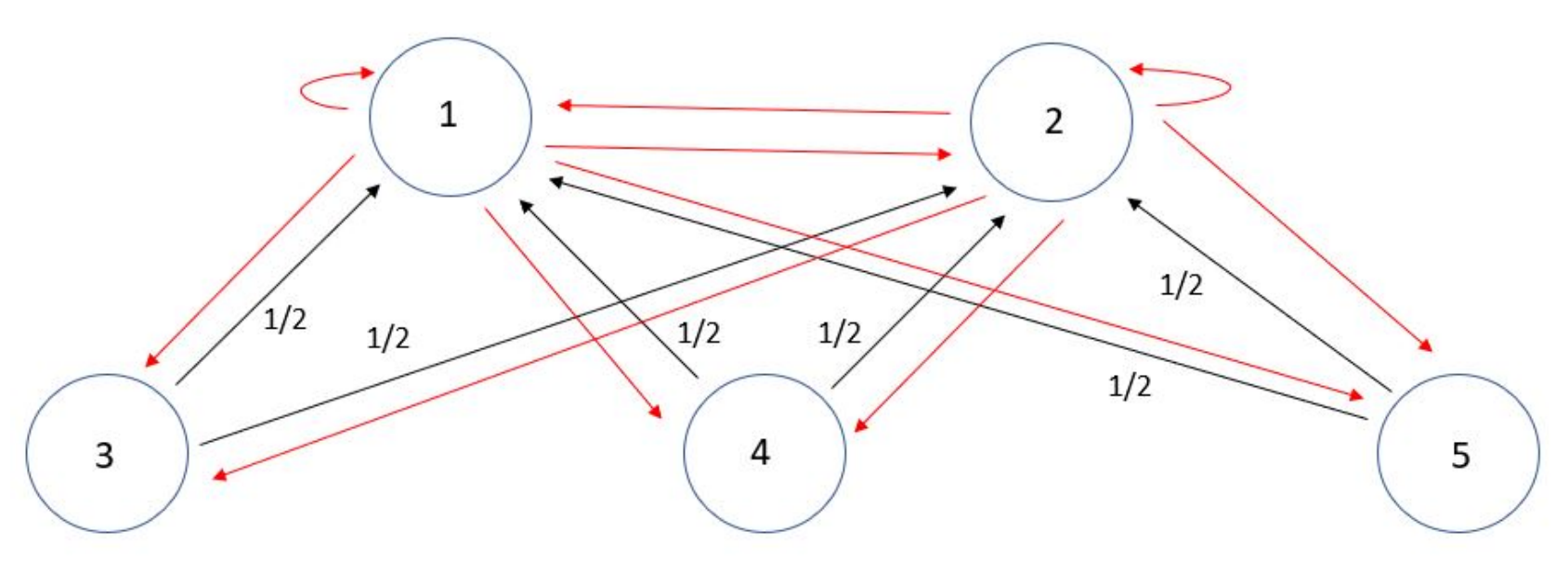
	Postgraduate degree programmes	Part-time(2 year)	Full-time(1 year)	Part-time Link	Full-time Link
0	Advanced Electronic and Electrical Engineering	H60C	H60A	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
1	Artificial Intelligence	I4U2	I4U1	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
2	Big Data Science	H6J6	H6J7	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
3	Computer Games		I4U4		https://www.qmul.ac.uk/postgraduate/taught/cou...
4	Computer Science	G4U2	G4U1	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
5	Computer Science by Research	G4Q2	G4Q1	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
6	Computing and Information Systems	G5U6	G5U5	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
7	Data Science and Artificial Intelligence by Co...		I4U5		https://www.qmul.ac.uk/postgraduate/taught/cou...
8	Electronic Engineering by Research	H6T6	H6T5	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
9	Internet of Things (Data)	I1T2	I1T0	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
10	Machine Learning for Visual Data Analytics	H6JZ	H6JE	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
11	Sound and Music Computing	H6T4	H6T8	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
12	Telecommunication and Wireless Systems	H6JD	H6JA	https://www.qmul.ac.uk/postgraduate/taught/cou...	https://www.qmul.ac.uk/postgraduate/taught/cou...
13	Digital and Technology Solutions (Apprenticeship)	I4DA		https://www.qmul.ac.uk/postgraduate/taught/cou...	

Q3.A:

We would consider nodes 1 and 2 to be authorities and nodes 3, 4 and 5 to be hubs.

Q3.B:

The black lines have a probability of 1/2, whereas all the red lines are the teleportation probabilities as they are dead-end nodes and therefore have all red lines have a probability of 1/5 as there are 5 nodes in total.



The table below shows all the transition probabilities, for example node 1 to any other node (including itself) is 0.2. Node 3 to nodes 1 or 2 has a probability of 0.5 but into any other nodes (including itself) it is 0.

In [47]:
```python
matrix = np.array([[0.2,0.2,0.2,0.2,0.2],
                   [0.2,0.2,0.2,0.2,0.2],
                   [0.5,0.5,0,0,0],
                   [0.5,0.5,0,0,0],
                   [0.5,0.5,0,0,0]])
trans_probs = pd.DataFrame(matrix, columns=[1,2,3,4,5],index=[1,2,3,4,5])
trans_probs
```

Out[47]:

	1	2	3	4	5
1	0.2	0.2	0.2	0.2	0.2
2	0.2	0.2	0.2	0.2	0.2
3	0.5	0.5	0.0	0.0	0.0
4	0.5	0.5	0.0	0.0	0.0
5	0.5	0.5	0.0	0.0	0.0

Q3.C:

The pagerank equations are as follows:

$$\pi(1) = \alpha/n + (1-\alpha)\cdot(\pi(1)/5 + \pi(2)/5 + \pi(3)/2 + \pi(4)/2 + \pi(5)/2)$$

$$\pi(2) = \alpha/n + (1-\alpha)\cdot(\pi(1)/5 + \pi(2)/5 + \pi(3)/2 + \pi(4)/2 + \pi(5)/2)$$

$$\pi(3) = \alpha/n + (1-\alpha)\cdot(\pi(1)/5 + \pi(2)/5)$$

$$\pi(4) = \alpha/n + (1-\alpha)\cdot(\pi(1)/5 + \pi(2)/5)$$

$$\pi(5) = \alpha/n + (1-\alpha)\cdot(\pi(1)/5 + \pi(2)/5)$$

Part 2

Q1

Removing all the stop words results in these sentences:

- data refer characteristic collect observation
- dataset view collection object
- data object describe number attribute
- attribute characteristic feature object

---	data	refer	characteristic	collect	observation	dataset	view	collection	object	describe	number	attribute	feature
Doc. 1	1	1	1	1	1	0	0	0	0	0	0	0	0
Doc. 2	0	0	0	0	0	1	1	1	1	0	0	0	0
Doc. 3	1	0	0	0	0	0	0	0	1	1	1	1	0
Doc. 4	0	0	1	0	0	0	0	0	1	0	0	1	1

To calculate the inverse document frequency, we use this formula: idf(w) = log10(|D|/|Dw|)

For example, data appears in two documents. The idf(data) = log10(4/2) = 0.30

In [48]:
```python
np.log10(4/2)
```

Out[48]:
```
0.3010299956639812
```

word	idf
data	0.30
refer	0.60
characteristic	0.30
collect	0.60
observation	0.60
dataset	0.60
view	0.60
collection	0.60
object	0.12
describe	0.60
number	0.60
attribute	0.30
feature	0.60

Q2

For the timeseries $y = \{0.1, 0.15, 0.2, 0.2, 0.3, 0.4, 0.25, 0.6, 0.5\}$, a binned version with k=3 would equal:

$y = \{0.15, 0.3, 0.45\}$

Q3

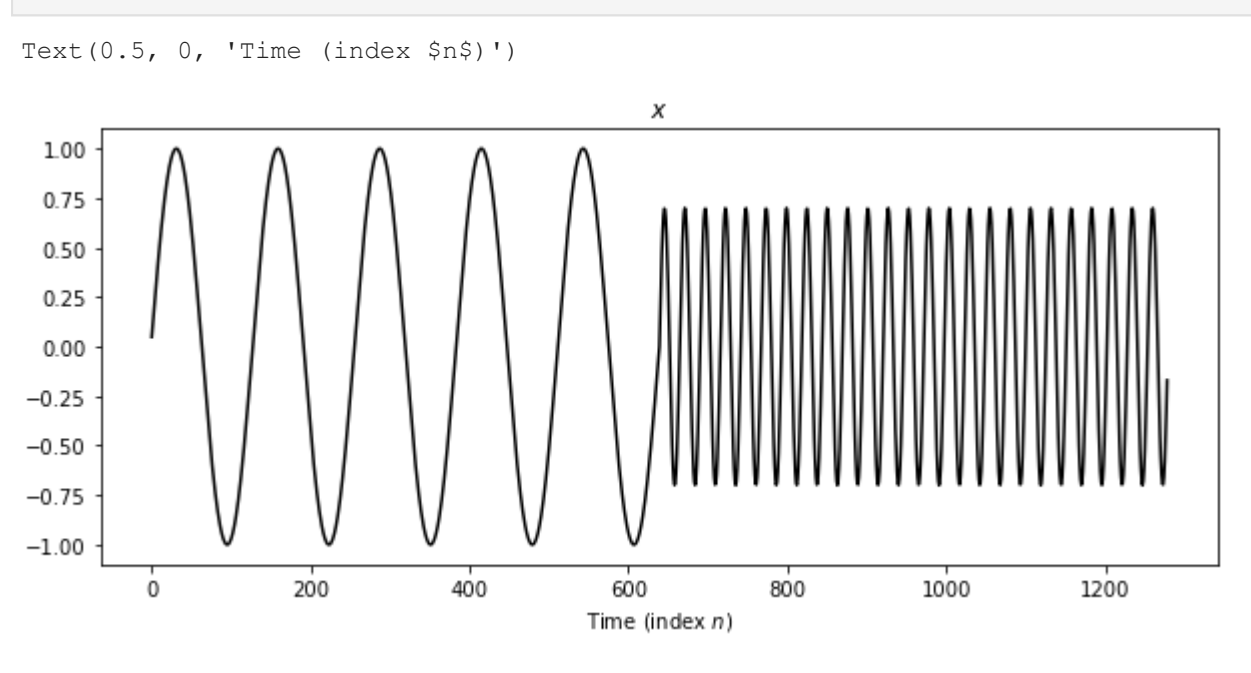
In [49]:
```python
import pandas as pd
time = pd.read_csv('timeseries.csv').to_numpy()
time = time.flatten()
time.shape
```

Out[49]:
```
(1279,)
```

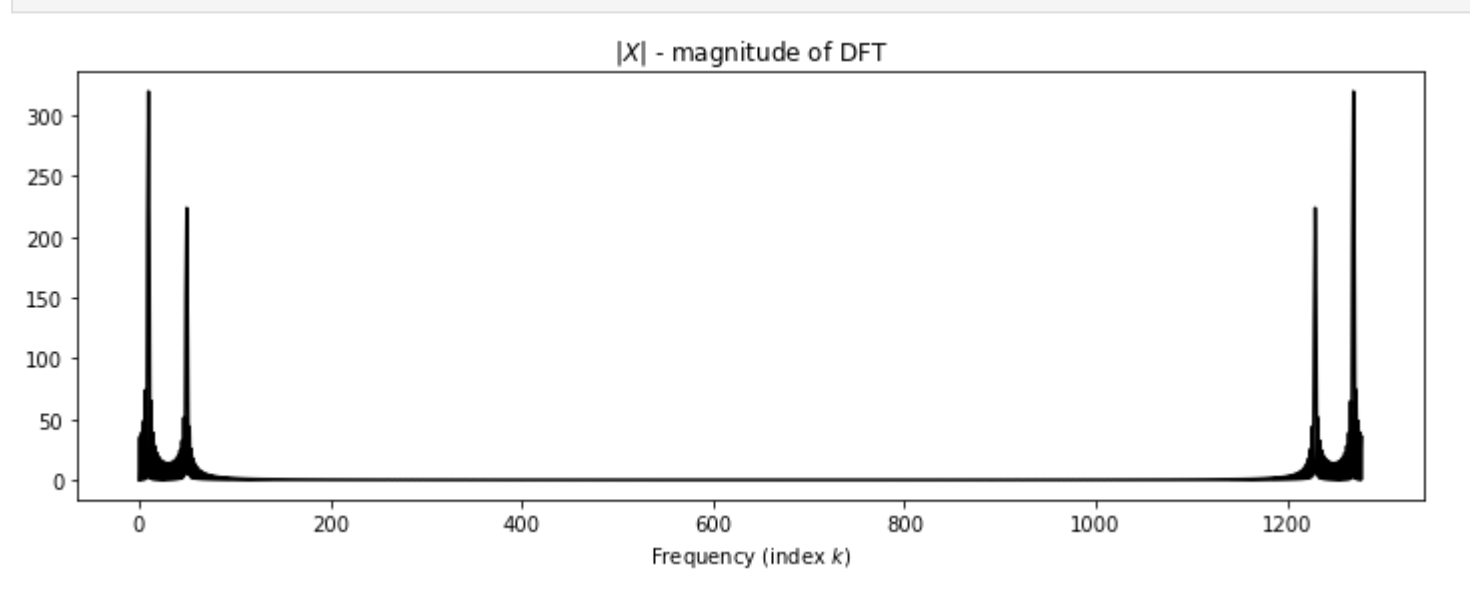
In [50]:
```python
from matplotlib import pyplot as plt

x = np.fft.fft(time)
# Plot
plt.figure(figsize=(10, 4))
plt.title('$x$')
plt.plot(time, 'k')
plt.xlabel('Time (index $n$)')
```

Out[50]:
```
Text(0.5, 0, 'Time (index $n$)')
```



In [51]:
```python
plt.figure(figsize=(10, 4))
plt.title('$|X|$ - magnitude of DFT')
plt.plot(np.abs(x), 'k')
plt.xlabel('Frequency (index $k$)')
plt.tight_layout()
```



There are 2 principal frequency components.

Q4

In [52]:
```python
series = pd.read_csv('births.csv', header=0, index_col=0)

print(series.head())
```

```
            Births
Date
1959-01-01      35
1959-01-02      32
1959-01-03      30
1959-01-04      31
1959-01-05      44
```

In [53]:
```python
rolling = series.rolling(window=7)
rolling_mean = rolling.mean().fillna(0)
print(rolling_mean.head(10))
```

```
               Births
Date
1959-01-01   0.000000
1959-01-02   0.000000
1959-01-03   0.000000
1959-01-04   0.000000
1959-01-05   0.000000
1959-01-06   0.000000
1959-01-07  35.142857
1959-01-08  36.285714
1959-01-09  37.142857
1959-01-10  36.714286
```

Autoregression (AR)

The errors are occurring due to using pandas series rather than numpy arrays, however the values should still be correct and it is useful to see the corresponding dates for the predicted values.

In [54]:
```python
from statsmodels.tsa.ar_model import AutoReg


model = AutoReg(rolling_mean, lags=2,old_names=False)
model_fit = model.fit()

yhat = model_fit.predict(len(rolling_mean), len(rolling_mean)+4)
print(yhat)
```

```
1960-01-01    45.380177
1960-01-02    44.960852
1960-01-03    44.590676
1960-01-04    44.271699
1960-01-05    43.997395
Freq: D, dtype: float64
C:\Users\ellio\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:536: ValueWarning: No frequency in
formation was provided, so inferred frequency D will be used.
  warnings.warn('No frequency information was'
C:\Users\ellio\anaconda3\lib\site-packages\statsmodels\tsa\deterministic.py:152: UserWarning: Only PeriodIndexe
s, DatetimeIndexes with a frequency set, RangesIndexes, and NumericIndex with a unit increment support extendin
g. The index is set will contain the position relative to the data length.
  warnings.warn(
```

Autoregression Moving Average (ARMA)

In [55]:
```python
from statsmodels.tsa.arima.model import ARIMA
model = ARIMA(rolling_mean, order=(2, 0, 2))
model_fit = model.fit()

yhat = model_fit.predict(len(rolling_mean), len(rolling_mean)+4) # arguments denote which dataset indices to pr
print(yhat)
```

```
C:\Users\ellio\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:536: ValueWarning: No frequency in
formation was provided, so inferred frequency D will be used.
  warnings.warn('No frequency information was'
C:\Users\ellio\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:536: ValueWarning: No frequency in
formation was provided, so inferred frequency D will be used.
  warnings.warn('No frequency information was'
C:\Users\ellio\anaconda3\lib\site-packages\statsmodels\tsa\base\tsa_model.py:536: ValueWarning: No frequency in
formation was provided, so inferred frequency D will be used.
  warnings.warn('No frequency information was'
1960-01-01    45.810247
1960-01-02    45.818766
1960-01-03    45.728092
1960-01-04    45.564016
1960-01-05    45.347305
Freq: D, Name: predicted_mean, dtype: float64
```

Q5

In [56]:
```python
import wikipedia
articles=['anomaly detection', 'cluster analysis', 'k-means clustering', 'data mining', 'data warehouse', 'asso
wiki_lst=[]
title=[]

for article in articles:
   print("loading content: ",article)
   wiki_lst.append(wikipedia.page(article,auto_suggest=False).content)
   title.append(article)
```

```
loading content:  anomaly detection
loading content:  cluster analysis
loading content:  k-means clustering
loading content:  data mining
loading content:  data warehouse
loading content:  association rule learning
```

In [57]:
```python
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
```

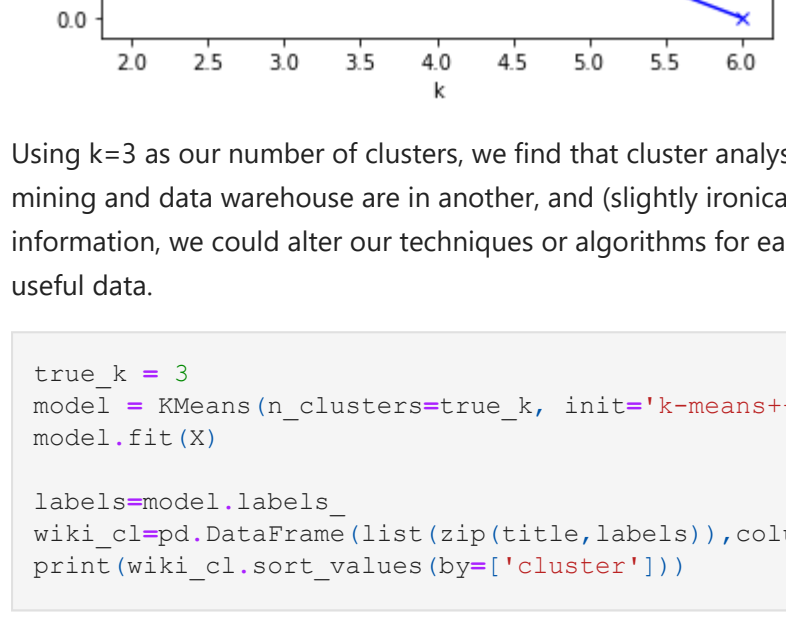
```
(6, 3511)
```

In [58]:
```python
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()
```



Using k=3 as our number of clusters, we find that cluster analysis, k-means clustering and association rule learning are in one cluster, data mining and data warehouse are in another, and (slightly ironically), that anomaly detection is in it's own separate cluster. Using this information, we could alter our techniques or algorithms for each wikipedia article depending on what cluster it belongs to to help extract useful data.

In [59]:
```python
true_k = 3
model = KMeans(n_clusters=true_k, init='k-means++', max_iter=200, n_init=10)
model.fit(X)

labels=model.labels_
wiki_cl=pd.DataFrame(list(zip(title,labels)),columns=['title','cluster'])
print(wiki_cl.sort_values(by=['cluster']))
```

```
                      title  cluster
1          cluster analysis        0
2        k-means clustering        0
5  association rule learning        0
3               data mining        1
4            data warehouse        1
0         anomaly detection        2
```

Thanks for reading!

Elliot Linsey, QMUL