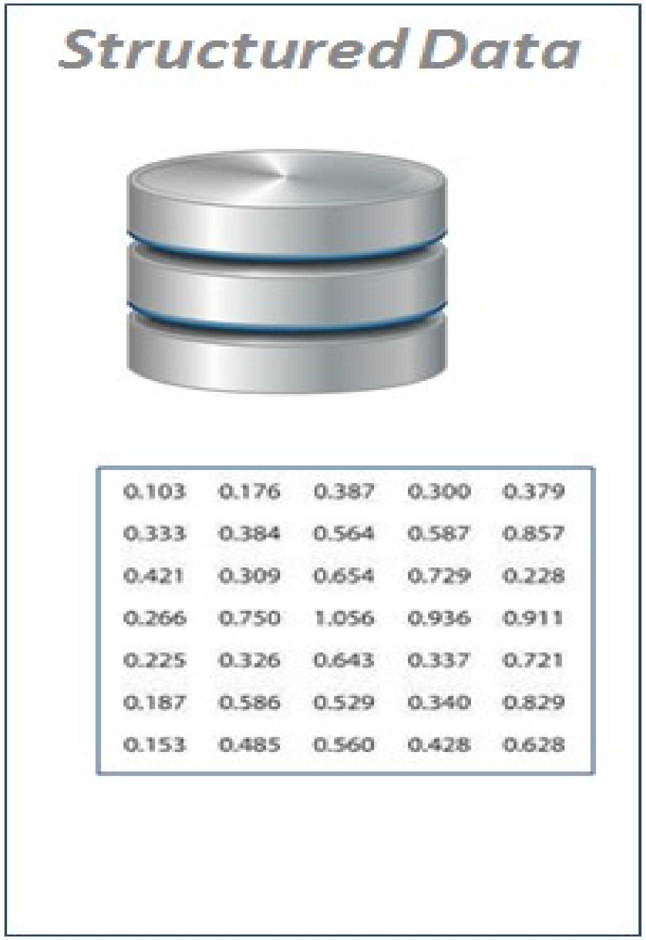
Text Mining
HR Appraisal Process Data
Using Python

DATA SCIENCE
INSTITUTE

Structured Vs. Unstructured Data



0.103	0.176	0.387	0.300	0.379
0.333	0.384	0.564	0.587	0.857
0.421	0.309	0.654	0.729	0.228
0.266	0.750	1.056	0.936	0.911
0.225	0.326	0.643	0.337	0.721
0.187	0.586	0.529	0.340	0.829
0.153	0.485	0.560	0.428	0.628

Unstructured Data
PDF

What Is Text Analysis ?

- Text Mining is also known as **Text Data Mining (TDM)** and **Knowledge Discovery in Textual Database (KDT)**
- It is a process of identifying novel information from a collection of texts (Also known as a 'Corpus')
- **Corpus is a collection of 'documents' containing natural language text.** Here, documents, generally, are sentences. Each document is represented as a separate line.

Case Study – HR Appraisal Process Feedback

Background

- The company XYZ carried out Annual Performance Appraisal process which is a routine HR process.
- The employees were asked to give feedback about the overall process and questions used for assessing their performance level.

Objective

- To understand the employee sentiments and incorporate recommendations in the current performance appraisal process.

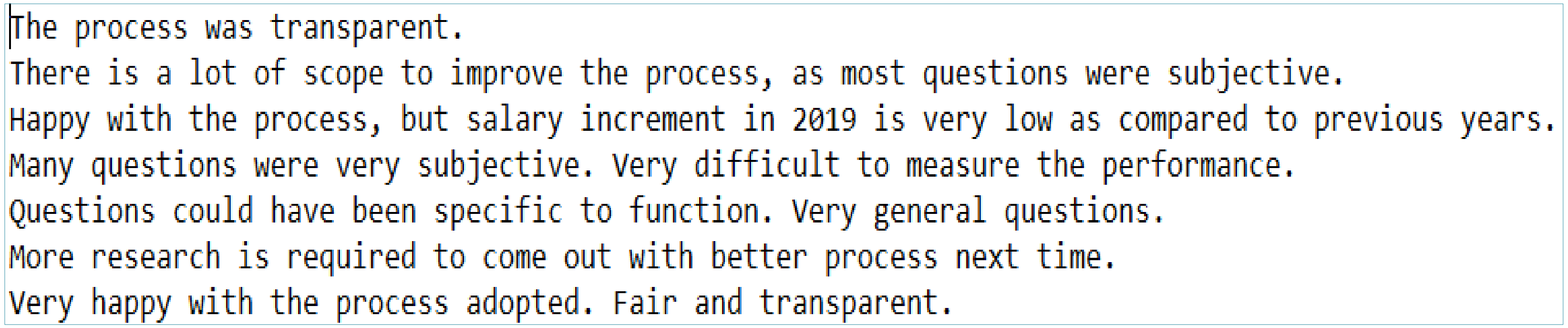
Available Information

- Feedback and comments from the employees were stored in a text document.

Data Snapshot

HR Appraisal process

Text Observations



```
The process was transparent.
There is a lot of scope to improve the process, as most questions were subjective.
Happy with the process, but salary increment in 2019 is very low as compared to previous years.
Many questions were very subjective. Very difficult to measure the performance.
Questions could have been specific to function. Very general questions.
More research is required to come out with better process next time.
Very happy with the process adopted. Fair and transparent.
```

*

These are the comments received from employees.
Note that, data is not in structured format.

Text Mining In Python

#Install NLTK library in Anaconda Prompt

```
pip install nltk
```

#Import NLTK library
#Import data and convert into 'Corpus'

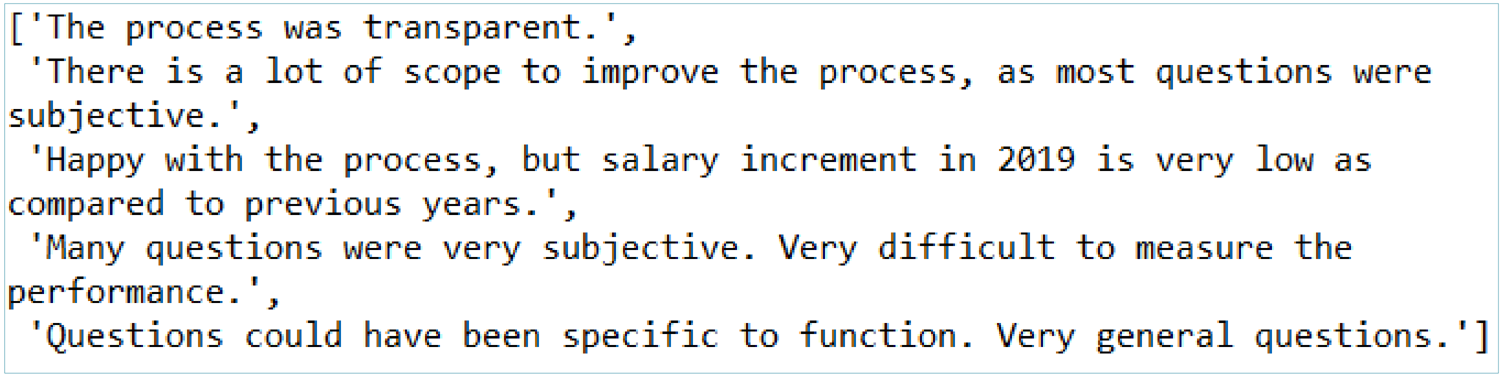
```
import nltk
nltk.download()
from nltk.book import *
text = [line.rstrip() for line in open("HR Appraisal
process.txt")]
text[0:5]
```

- Install and load **NLTK**(**Natural Language Toolkit**) library.
- **open("HR Appraisal process.txt"): Opens the file "HR Appraisal process.txt" for reading.**
- **line.rstrip():** For each line in the file, the rstrip() method is called to remove any trailing whitespace characters, including the newline character \n at the end of each line.
- **[line.rstrip() for line in open("HR Appraisal process.txt")]:** This is a list comprehension that iterates over each line in the file, applies rstrip() to each line, and stores the result in a list.
- The resulting list, text, will contain all the lines from the file "HR Appraisal process.txt",

*** Note : When imported nltk, nltk.download() will download the required libraries from NLTK for text mining. Run nltk.download() only for the first time**

Text Mining In Python

```
# Output:
```



```
['The process was transparent.',
 'There is a lot of scope to improve the process, as most questions were
subjective.',
 'Happy with the process, but salary increment in 2019 is very low as
compared to previous years.',
 'Many questions were very subjective. Very difficult to measure the
performance.',
 'Questions could have been specific to function. Very general questions.']
```

Interpretation:

- **text[0:5]** prints first 5 text lines from the data with each line as one set of strings.

```
# Display a particular document from corpus.
```

```
text[2]
```

```
'Happy with the process, but salary increment in 2019 is very low as
compared to previous years.'
```

- **text[2]** prints text line of specified number in []. Here it is printing 3rd li
- Python indexing starts from 0, thus 2 represents 3rd data point (sentenc

Text Mining In Python

```
# Clean the Corpus for further analysis
```

```
corp = [item.lower() for item in text]
corp [2]
```

```
'happy with the process, but salary increment in 2019 is very low as
compared to previous years.'
```

```
from string import punctuation
remove_punc = str.maketrans('','', punctuation)
corp = [item.translate(remove_punc) for item in corp]
corp[2]
```

```
'happy with the process but salary increment in 2019 is very low as
compared to previous years'
```

```
from string import digits
remove_digits = str.maketrans('', '', digits)
cor
cor
```

```
'hap
comp
```

- **lower()** converts text to lowercase.
- **maketrans(x,y,z)**

Where:
x is a string specifying the characters to be replaced.
y is a string specifying the characters with which to replace x.
z is a string specifying the characters to be removed.

- **maketrans(“,”,punctuation)** removes punctuation
- **maketrans(“,”,digits)** removes digits

Text Mining In Python

```
# Clean the Corpus for further analysis
```

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
stop_words = nltk.corpus.stopwords.words('english')
fs=[]
for item in corp:
    word_tokens = word_tokenize(item)
    filtered_sentence = [word for word in word_tokens if word not in
stop_words]
    fs.append(filtered_sentence)
fs[2]
```

```
['happy', 'process', 'salary', 'increment', 'low', 'compared’,
'previous', 'years']
```

- **stopwords("english")** remove stop words like: i, me, our, and, the, is, etc.
- There are more than 100 in-built English Stopwords in Python NLTK. Use stopwords("english") to view the list of these stopwords.

Text Mining In Python

```
# Clean the Corpus for further analysis
```

```
newStopWords = ['process']
stop_words.extend(newStopWords)
fs=[]
for item in corp:
    word_tokens = word_tokenize(item)
    filtered_sentence = [word for word in word_tokens if word not in
stop_words]
    fs.append(filtered_sentence)
fs[2]
```

```
['happy', 'salary', 'increment', 'low', 'compared', 'previous',
'years']
```

- If you wish to remove specific words from the corpus use **.extend("word")** to add the word in list of stopwords. Here "**process**" word is removed.

Text Mining In Python

```
import itertools
filtered_text = list(itertools.chain.from_iterable(fs))
fdist = nltk.FreqDist(filtered_text)
```

- **itertools.chain.from_iterable()** function is used to flatten a list of lists (fs) into a single list.
- **FreqDist()** gives frequency of each word in the list

```
fdist.most_common(10)
```

```
[('questions', 13),
 ('hr', 12),
 ('happy', 10),
 ('subjective', 8),
 ('fair', 7),
 ('performance', 6),
 ('work', 6),
 ('difficult', 5),
 ('measure', 5),
 ('salary', 4)]
```

Interpretation:

- “questions”, “hr”, “happy”, “subjective”, “fair”, “performance”, “work”, “difficult”, “measure”, ”salary” are the top 10 words by frequency.
- The frequencies of the words are listed besides them.

- **fdist.most_common(n)** gives the list of top n words sorted highest to lowest by frequency

Word Cloud In Python

Word cloud, as the name suggests, is an **image showing compilation of words**, in which, **size of words indicates its frequency or importance**.

```
# Install the library “wordcloud” in Anaconda Prompt
```

```
pip install wordcloud
```

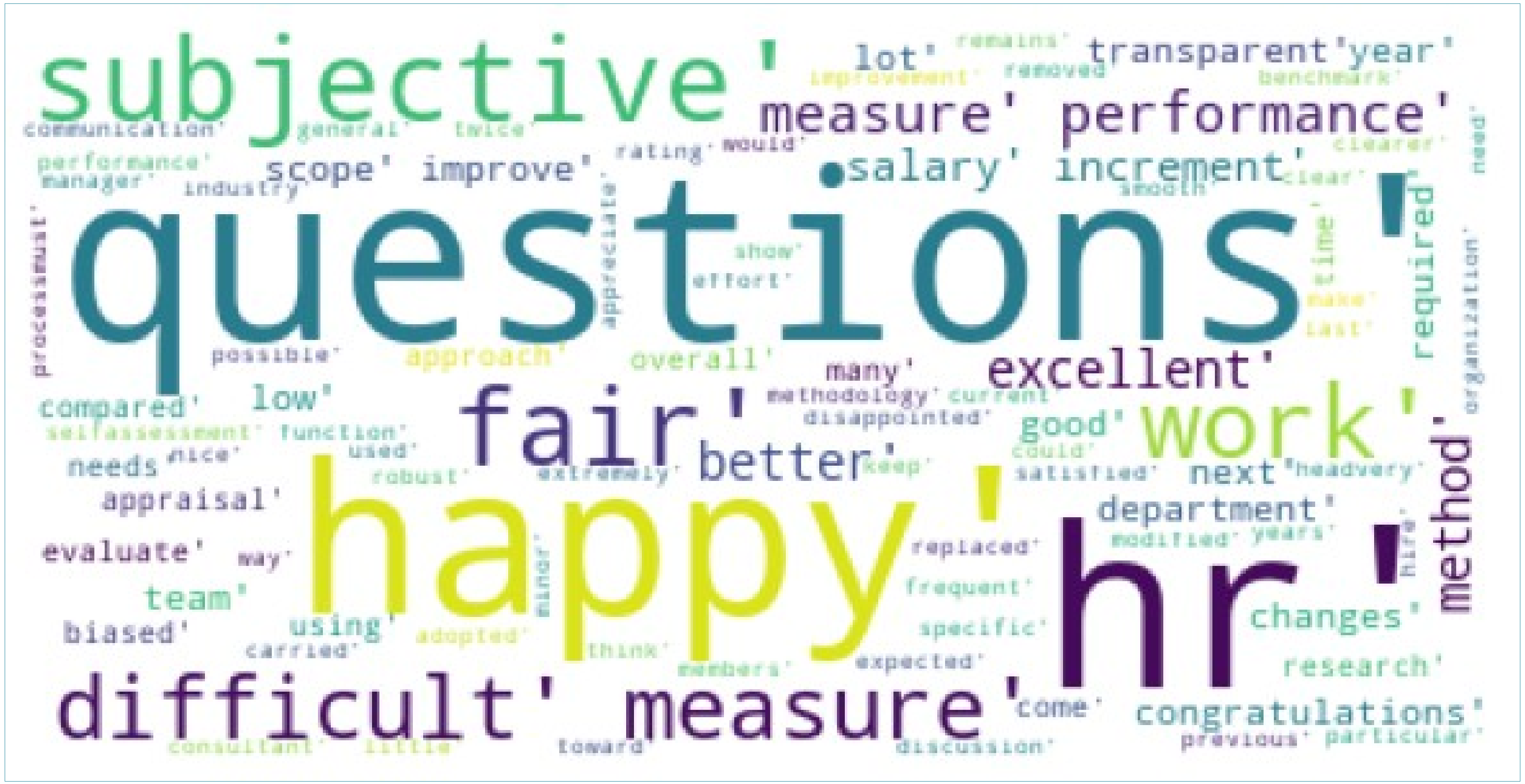
```
# Get Word Cloud
```

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt
wordcloud =
WordCloud(background_color="white").generate(str(filtered_text))
plt.figure(figsize = (8, 8))
plt.imshow(wordcloud); plt.axis("off")
plt.tight_layout(pad = 0); plt.show()
```

- **background.color** allows you to select the color of the background.
- **fig.size** allows you to adjust the size/dimensions of the wordcloud.
- **plt.imshow()** is used to display data as an image.
- **plt.axis("off")** means axis lines and labels are turned off.

Word Cloud In Python

```
# Output :
```



Interpretation:

- Word ‘questions’ has largest size, indicating most frequent word followed by ‘happy’ and ‘hr’ and so on..

Text Mining Using Matplotlib

```
# Plotting frequent terms as a bar plot
```

```
a = fdist.most_common(10)
```

```
# Transform as a dataframe
```

```
import pandas as pd
b = pd.DataFrame(a)
b = b.rename(columns={0:'Words',1:'Freq'})
```

```
 # Horizontal bar plot
```

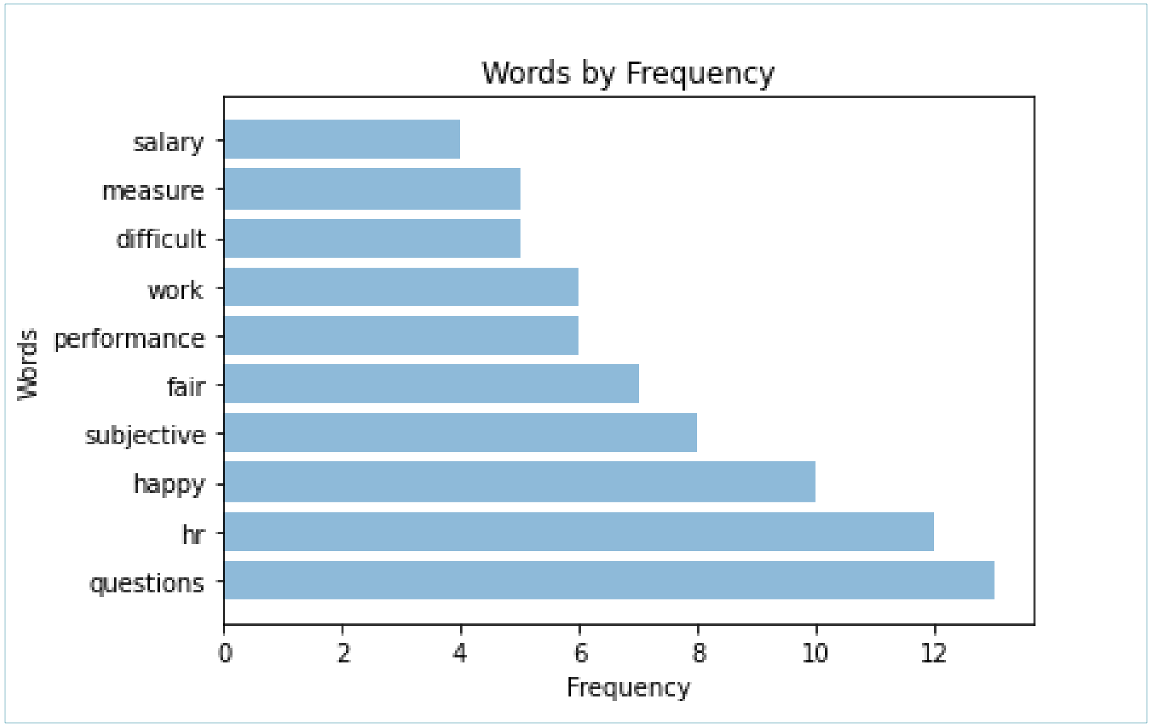
```
import numpy as np
c=b.Words
y=np.arange(len(c))
x=b.Freq

plt.barh(y, x, align='center', alpha=0.5)
plt.yticks(y, c);plt.ylabel('Words')
plt.xlabel('Frequency');plt.title('Words by Frequency')

plt.show()
```

Text Mining Using Matplotlib

```
# Output :
```



Interpretation:

- Graph shows the frequency of the top 10 words by frequency on a horizontal bar graph. “questions” is the most frequent word with frequency 13.

Quick Recap

In this session, we learnt **Text Mining in Python** :

Unstructured Data	• Does not reside in traditional databases and data warehouses. • Example: emails, tweets, feedback, blogs, webpages, etc.
Text Analysis	• Process of identifying novel information from a collection of texts. (Also known as a ‘Corpus’)
Text mining in Python	• Install ‘**nltk**’ library. Convert data into corpus. • Clean the corpus: convert all words to lowercase/uppercase, remove punctuation, numbers, stopwords, words.
Word Cloud in Python	• An image showing compilation of words, in which, size of words indicates its frequency or importance. • Install ‘**wordcloud**’ library.

THANK YOU!