



BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

REFERENCES

THANK YOU

BSDS-206 : TEXT ANALYTICS

COURSE INSTRUCTOR - DR. AASHIMA BANGIA

Gauri Sharan - BSc Data Science, Semester 4

June 11, 2024



TABLE OF CONTENTS

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

REFERENCES

THANK YOU

1 TEXT PROCESSING AND MINING

2 RAW TEXTUAL DATA TO FEATURES

3 ADVANCED PROCESSES

4 WORD EMBEDDINGS AND MORE

5 NEURAL NETWORKS

6 THANK YOU



EXTRACTING DATA

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING
NLP CONCEPTS

DATA HANDLING
PREPROCESSING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

EXTRACTING DATA

Extracting data from various sources such as PDFs, Word files, JSON, HTML, etc.

```
import re
text = "Hello, World!"
pattern = r'\w+'
matches = re.findall(pattern, text)
print(matches)
```



COLLECTING DATA

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING
NLP CONCEPTS
DATA HANDLING
PREPROCESSING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

COLLECTING DATA

Collecting data from various sources using libraries and tools.

```
import json
with open('data.json') as f:
    data = json.load(f)
print(data)
```



PARSING DATA

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING
NLP CONCEPTS

DATA HANDLING
PREPROCESSING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

PARSING

Reading and interpreting data using regular expressions.

```
import re
text = "Hello, World!"
pattern = r'\w+'
matches = re.findall(pattern, text)
print(matches)
```



STRUCTURING DATA

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING
NLP CONCEPTS
DATA HANDLING
PREPROCESSING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

STRUCTURING DATA

Handling strings, scraping data from the web, and exploring and preprocessing text data.

```
import pandas as pd
df = pd.DataFrame({
    'Name': ['John', 'Mary', 'Jane'],
    'Age': [25, 30, 35]
})
print(df)
```



EXPLORING AND PREPROCESSING TEXT DATA

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING
NLP CONCEPTS
DATA HANDLING

PREPROCESSING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

NLP Python Libraries:

NLTK Natural Language Toolkit: A comprehensive library for NLP tasks such as tokenization, stemming, tagging, parsing, and semantic reasoning. It is widely used for text preprocessing and machine learning tasks.

spacy A modern NLP library that focuses on production-ready models for NLP tasks such as tokenization, entity recognition, and language modeling. It is known for its high-performance and ease of use.

Gensim A library for topic modeling and document similarity analysis. It is particularly useful for processing large volumes of text data and extracting insights from it.



PYTHON CODE EXAMPLE - EXPLORING AND PREPROCESSING DATA

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING
NLP CONCEPTS
DATA HANDLING
PREPROCESSING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

```
import re
import requests
from bs4 import BeautifulSoup

# Scraping example
url = 'http://example.com'
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
text = soup.get_text()

# Regular expression example
pattern = re.compile(r'\b[A-Za-z]+\b')
matches = pattern.findall(text)
print(matches)
```



CONVERTING TEXT DATA TO LOWERCASE

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

RAW TEXT TO
FEATURES

TEXT PROCESSING
PIPELINE

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

WHY LOWERCASE?

Lowercase text is easier to process and compare.

```
import string
text = "Hello, World!"
text = text.lower()
print(text)
```



REMOVING PUNCTUATION

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

RAW TEXT TO
FEATURES

TEXT PROCESSING
PIPELINE

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

WHY REMOVE PUNCTUATION?

Punctuation can affect the accuracy of text processing.

```
import string
text = "Hello, World!"
text = text.translate(str.maketrans('', '', string.punctuation))
print(text)
```



REMOVING STOP WORDS

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

RAW TEXT TO
FEATURES

TEXT PROCESSING
PIPELINE

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

WHY REMOVE STOP WORDS?

Stop words are common words that do not carry much meaning.

```
import nltk
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
text = "Hello, World!"
words = text.split()
filtered_words = [word for word in words if word not in stop_words]
print(filtered_words)
```



STANDARDIZING TEXT

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

RAW TEXT TO
FEATURES

TEXT PROCESSING
PIPELINE

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

TOKENIZATION

Breaking text into individual words or tokens.

STEMMING

Reducing words to their base form.

LEMMATIZING

Reducing words to their base form using a dictionary.

```
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
text = "running"
print(lemmatizer.lemmatize(text))
```



BUILDING A TEXT PROCESSING PIPELINE - CONVERTING TEXT TO FEATURES

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

RAW TEXT TO
FEATURES

TEXT PROCESSING
PIPELINE

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

WHY PIPELINE?

Pipelines help to streamline text processing tasks.

- One-hot encoding
- Count Vectorizing
- TF-IDF
- Generating N-grams, Co-occurrence Matrix, Hash Vectorizing
- Implementing Word Embeddings



PYTHON CODE EXAMPLE - PIPELINE

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

RAW TEXT TO
FEATURES

TEXT PROCESSING
PIPELINE

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
```

```
# Example text
```

```
texts = ["This is a sentence.", "This is another sentence."]
```

```
# Count Vectorizer
```

```
count_vectorizer = CountVectorizer()
```

```
X_count = count_vectorizer.fit_transform(texts)
```

```
print(X_count.toarray())
```

```
# TF-IDF Vectorizer
```

```
tfidf_vectorizer = TfidfVectorizer()
```

```
X_tfidf = tfidf_vectorizer.fit_transform(texts)
```

```
print(X_tfidf.toarray())
```



POS TAGGING

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

POS TAGGING

TOPIC MODELING

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

REFERENCES

WHAT IS POS TAGGING?

Part-of-speech tagging is the process of identifying the part of speech of each word in a sentence.

WHY POS TAGGING?

POS tagging is useful for tasks such as sentiment analysis and named entity recognition.

```
import nltk
from nltk import pos_tag
text = "Hello, World!"
tokens = word_tokenize(text)
pos_tags = pos_tag(tokens)
print(pos_tags)
```



TOPIC MODELING

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

POS TAGGING
TOPIC MODELING

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

REFERENCES

WHAT IS TOPIC MODELING?

Topic modeling is the process of identifying topics or themes in a large corpus of text.

WHY TOPIC MODELING?

Topic modeling is useful for tasks such as text classification and clustering.

TOPIC MODELLING TECHNIQUES

- Latent Dirichlet Allocation (LDA)
- Non-Negative Matrix Factorization (NMF)
- Latent Semantic Analysis (LSA)
- Parallel Latent Dirichlet Allocation (PLDA)



PYTHON CODE EXAMPLE - TOPIC MODELLING

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

POS TAGGING
TOPIC MODELING

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

REFERENCES

```
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import LatentDirichletAllocation
def topic_modeling(text):
    tokens = word_tokenize(text)
    stop_words = set(stopwords.words('english'))
    filtered_tokens = [token for token in tokens if token not in stop_words]
    lemmatizer = WordNetLemmatizer()
    lemmatized_tokens = [lemmatizer.lemmatize(token) for token in filtered_tokens]
    vectorizer = TfidfVectorizer()
    tfidf = vectorizer.fit_transform(lemmatized_tokens)
    lda = LatentDirichletAllocation(n_components=2)
    topics = lda.fit_transform(tfidf)
    return topics

text = "Hello, World!" ; topics = topic_modeling(text)
print(topics)
```



WORD EMBEDDINGS

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

WHAT ARE WORD EMBEDDINGS?

Word embeddings are dense vector representations of words.

WHY WORD EMBEDDINGS?

Word embeddings are useful for tasks such as text classification and clustering.

WORD EMBEDDING TECHNIQUES

- Vectorizing, Continuous Bag of Words (CBoW), N-Gram, Word2Vec
- Training a Word Embedding through Skip-Grams, CBoW
- Global Vectors for Word Representations (GloVe)
- Using trained Word Embeddings with LSTMs
- Paragraph2Vec: Distributed Memory of Paragraph Vectors (PV-DM)



PYTHON CODE EXAMPLE - WORD EMBEDDING

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

```
import numpy as np
from sklearn.preprocessing import Normalizer

def word_embeddings(word):
    # Define word embeddings
    word_embeddings = {
        'hello': np.array([1, 2, 3]),
        'world': np.array([4, 5, 6])
    }
    return word_embeddings.get(word, np.zeros(3))

word = "hello"
embedding = word_embeddings(word)
print(embedding)
```



PYTHON CODE EXAMPLE - WORD2VEC

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

```
from gensim.models import Word2Vec

# Example sentences
sentences = [["This", "is", "a", "sentence"], ["This", "is", "another", "sentence"]]

# Train Word2Vec model
model = Word2Vec(sentences, vector_size=100, window=5, min_count=1, workers=4)
print(model.wv['sentence'])
```



TEXT GENERATION

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

WHAT IS TEXT GENERATION?

Text generation is the process of generating new text based on a given input.

WHY TEXT GENERATION?

Text generation is useful for tasks such as chatbots and language translation.

METHODS:

- Text Generation with LSTMs
- Creating NER tagging
- Sequence-to-sequence Models (Seq2Seq)



PYTHON CODE EXAMPLE 1 - TEXT GENERATION

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

```
import numpy as np
from sklearn.preprocessing import Normalizer

def text_generation(input_text):
    # Define a simple text generation model
    model = {
        'hello': 'world',
        'world': 'hello'
    }
    return model.get(input_text, 'Unknown')

input_text = "hello"
output_text = text_generation(input_text)
print(output_text)
```



PYTHON CODE EXAMPLE 2 - TEXT GENERATION

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

```
from keras.models import Sequential
from keras.layers import LSTM, Dense

model = Sequential()
model.add(LSTM(50, return_sequences=True, input_shape=(100, 1)))
model.add(LSTM(50))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')

print(model.summary())
```



MACHINE TRANSLATION

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

WHAT IS MACHINE TRANSLATION?

Machine translation is the process of translating text from one language to another.

WHY MACHINE TRANSLATION?

Machine translation is useful for tasks such as language translation and localization.



PYTHON CODE EXAMPLE - MACHINE TRANSLATION

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

WORD EMBEDDINGS

TEXT GENERATION

MACHINE
TRANSLATION

NEURAL
NETWORKS

```
import numpy as np
from sklearn.preprocessing import Normalizer

def machine_translation(input_text):
    # Define a simple machine translation model
    model = {
        'hello': 'bonjour',
        'world': 'monde'
    }
    return model.get(input_text, 'Unknown')

input_text = "hello"
output_text = machine_translation(input_text)
print(output_text)
```



UNDERSTANDING ACTIVATION FUNCTIONS IN NEURAL NETWORKS

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS

LOSS FUNCTION

DEFINITION

Activation functions introduce non-linearity into the neural network.

COMMON ACTIVATION FUNCTIONS

- Sigmoid
- ReLU (Rectified Linear Unit)
- Tanh (Hyperbolic Tangent)



ACTIVATION FUNCTIONS GRAPHS

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS

LOSS FUNCTION

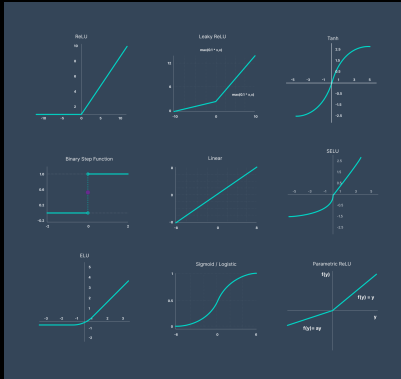


FIGURE: Activation Functions Graphs



LOSS FUNCTION

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS

LOSS FUNCTION

DEFINITION

The loss function measures the difference between the predicted output and the actual output.

COMMON LOSS FUNCTIONS

- Mean Squared Error (MSE)
- Cross-Entropy



PERCEPTRON:- SINGLE LAYERED AND MULTIPLE LAYERED

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS
LOSS FUNCTION

SINGLE LAYER PERCEPTRON

A single layer neural network with a linear activation function.

MULTI-LAYER PERCEPTRON

A neural network with multiple layers, allowing for more complex representations.



BUILDING A SINGLE NEURON NEURAL NETWORK FROM SCRATCH IN PYTHON

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS
LOSS FUNCTION

```
class Neuron:
    def __init__(self, inputs, bias):
        self.weights = [random.random() for _ in range(inputs)]
        self.bias = bias

    def forward(self, inputs):
        output = sum([x * y for x, y in zip(inputs, self.weights)]) + self.bias
        return self.sigmoid(output)

    def sigmoid(self, x):
        return 1 / (1 + math.exp(-x))

neuron = Neuron(2, 1)
inputs = [0, 0]
output = neuron.forward(inputs)
print(output)
```



PRACTICAL IMPLEMENTATION OF NEURAL NETWORK TRAINING IN PYTHON

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS
LOSS FUNCTION

```
import numpy as np
# Define the neural network
class NeuralNetwork:
    def __init__(self, inputs, hidden, outputs):
        self.weights1 = np.random.rand(inputs, hidden)
        self.weights2 = np.random.rand(hidden, outputs)

    def forward(self, inputs):
        hidden_layer = np.maximum(np.dot(inputs, self.weights1), 0)
        output_layer = np.dot(hidden_layer, self.weights2)
        return output_layer

# Train the neural network
nn = NeuralNetwork(2, 2, 1)
inputs = np.array([[0, 0], [0, 1], [1, 0], [1, 1]])
outputs = np.array([[0], [1], [1], [0]])
for _ in range(1000):
    output = nn.forward(inputs)
    error = outputs - output
    # Update weights
```



CONVOLUTION AND RECURRENT NEURAL NETWORKS

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS
LOSS FUNCTION

WHAT ARE CONVOLUTIONAL NEURAL NETWORKS?

Convolutional neural networks are a type of neural network that uses convolutional and pooling layers to process data.

WHAT ARE RECURRENT NEURAL NETWORKS?

Recurrent neural networks are a type of neural network that uses recurrent layers to process sequential data.

WHY CONVOLUTIONAL AND RECURRENT NEURAL NETWORKS?

Convolutional and recurrent neural networks are useful for tasks such as image classification and text classification.



PYTHON CODE EXAMPLE 1 - CNN AND RNN

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS
LOSS FUNCTION

```
import numpy as np
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from keras.layers import LSTM, Embedding

# Define a convolutional neural network
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(10, activation='softmax'))

# Define a recurrent neural network
model = Sequential()
model.add(Embedding(input_dim=10000, output_dim=128, input_length=100))
model.add(LSTM(128, dropout=0.2))
model.add(Dense(64, activation='relu'))
model.add(Dense(10, activation='softmax'))
```



PYTHON CODE EXAMPLE 2 - CNN AND RNN

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

UNDERSTANDING
ACTIVATION
FUNCTIONS IN
NEURAL NETWORKS
LOSS FUNCTION

```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences

texts = ["This is a sentence.", "This is another sentence."]
tokenizer = Tokenizer(num_words=100)
tokenizer.fit_on_texts(texts)
sequences = tokenizer.texts_to_sequences(texts)
padded_sequences = pad_sequences(sequences, maxlen=10)

print(padded_sequences)
```



REFERENCES

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES






ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

REFERENCES

THANK YOU

-  Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, 2009.
-  Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Pearson, 2021.
-  Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
-  Yoav Goldberg. *Neural Network Methods for Natural Language Processing*. Morgan & Claypool Publishers, 2017.
-  Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.



THANK YOU

BSDS-206 :
TEXT
ANALYTICS

GAURI
SHARAN

TEXT
PROCESSING
AND MINING

RAW
TEXTUAL
DATA TO
FEATURES

ADVANCED
PROCESSES

WORD
EMBEDDINGS
AND MORE

NEURAL
NETWORKS

REFERENCES

THANK YOU

Hope you liked this presentation.

Gauri Sharan

Student, School of Data Science
AAFT Noida (Shobhit University)
BSc Data Science 2022-25
Semester 4, 2024

- LinkedIn: [linkedin.com/in/gauri-sharan](https://www.linkedin.com/in/gauri-sharan)
- GitHub: github.com/gaurisharan
- Mail: gaurisharan123@gmail.com