Types of Data

Data

1. Numerical
2. catagorical
3. Text
4. Image
5. video
6. Audio

Text data

1. Tokenization: Split text into words or tokens.
2. Removing punctuation and special characters.
3. Lowercasing: Convert all text to lowercase.
4. Stopword Removal: Remove common words that do not carry much meaning.
5. Stemming and Lemmatization: Reduce words to their root form.
6. Vectorization: Convert text into numerical representations (Bag of Words, TF-IDF, Word Embeddings).

Image data

1. Resizing: Make images uniform in size
2. Normalization: Scale pixel values to a range (e.g., [0, 1] or [-1, 1])
3. Data Augmentation:Generate additional training samples through transformations like rotation, flipping, zooming, etc
4. Feature extraction:Use pre-trained CNN models for feature extraction.

Audio data

1. Resampling:Convert audio samples to a common sampling rate.
2. Feature extraction:Extract features like Mel-frequency cepstral coefficients (MFCCs), spectrograms, etc
3. Normalization:Scale audio features to a common range
4. Data Augmentation:Apply transformations such as pitch shifting, time stretching, adding noise, etc.

VideoFrame extraction (Extract frames from videos)

1. Resizing: Make frames uniform in size
2. Feature extraction: Use pre-trained CNN models to extract features from video frames.
3. Normalization :Scale pixel values to a common range.
4. Data Augmentation:Apply transformations like flipping, rotation, cropping, etc., to augment the data.

Catagorical data

1. one hot encoding:Convert categorical variables into binary matrices.
2. Label encoding:Encode categorical variables into ordinal integers.
3. Target encoding:Encode categorical variables based on the target variable.

Numerical data

1. Standardization:Min-Max scaling, standardization, robust scaling.
2. Normalization
3. Feature engineering:Polynomial features, interaction terms, binning.

Data Preprocessing steps

Data Preprocessing

1. Data Cleaning
2. Data Integration
3. Data Reduction
4. Data Transformation
5. Data Discretization
6. Data Normalisation

The steps involved for Data Cleaning

1. After the loading the data set check the missing values and null values. If the missing values are less than 20% delete those rows.
2. If the missing values are more than 20% then try to fill the values using median. because median is the best measure even when the outliers are present.
3. When the Outliers are present try to remove those outliers based on the use case or else treat the outliers using Inter Quartile range, z-score etc.
4. In the dataset if duplicates appears remove it avoid bias and redundancy.

Data Integration

1. Combine data from multiple sources or datasets into a single dataset. This may involve merging, joining, or concatenating datasets based on common attributes or keys.

Data Reduction

1. his involves reducing the size of the dataset while preserving the important information. Data reduction can be achieved through techniques such as feature selection and feature extraction. Feature selection involves selecting a subset of relevant features from the dataset, while feature extraction involves transforming the data into a lower-dimensional space while preserving the important information.

Data Transformation

1. This involves converting the data into a suitable format for analysis. Common techniques used in data transformation include normalization, standardization, and discretization. Normalization is used to scale the data to a common range, while standardization is used to transform the data to have zero mean and unit variance. Discretization is used to convert continuous data into discrete categories.

Data Discretization

1. This involves dividing continuous data into discrete categories or intervals. Discretization is often used in data mining and machine learning algorithms that require categorical data. Discretization can be achieved through techniques such as equal width binning, equal frequency binning, and clustering.

Data Normalisation

1. Normalize the data to reduce skewness or ensure that it follows a specific distribution. Techniques like log transformation, Box-Cox transformation, or quantile normalization can be used for data normalization.

4.

Required Libraries for text data

NLTK is an essential library that **supports tasks such as classification, stemming, tagging, parsing, semantic reasoning, and tokenization in Python**

```
#import nltk
#from nltk.tokenize import word_tokenize
```

- **TextBlob most basic NLP tasks like sentiment analysis, pos-tagging, or noun phrase extraction.Features like spelling correction and translation allow developers to perform NLP tasks without wading deep into intricate processes.**

# Use-case: Sentiment analysis of a sentence.

```
# from textblob import TextBlob
    #text = "I love using this product. It's fantastic!"
#blob = TextBlob(text)
#sentiment = blob.sentiment.polarity
#print(sentiment)
```

- CoreNLP **accurate natural language parsing and rich linguistic annotations.It's renowned for its robustness and supports various tasks, including named entity recognition and coreference resolution.**

Use-case: Named Entity Recognition (NER).

```
# Note: This requires running the CoreNLP server and its Python wrapper.
#from pycorenlp import StanfordCoreNLP
#nlp = StanfordCoreNLP('http://localhost:9000')
#text = "Barack Obama was the president of the United States."
#result = nlp.annotate(text, properties={'annotators': 'ner', 'outputFormat': 'json'})
# print(result['sentences'][0]['entitymentions'])
```

- **Gensim** is a Python library that **identifies semantic similarity between two documents through vector space modeling and topic modeling toolkits.**

# Use-case: Topic modeling with LDA (Latent Dirichlet Allocation).

```
import gensim
from gensim import corpora
documents = ["Human machine interface for lab abc computer applications",
"A survey of user opinion of computer system response time",
...]
texts = [doc.split() for doc in documents]
dictionary = corpora.Dictionary(texts)
corpus = [dictionary.doc2bow(text) for text in texts]
```

```
lda = gensim.models.ldamodel.LdaModel(corpus, num_topics=3, id2word=dictionary,
passes=15)
print(lda.print_topics(num_topics=3, num_words=3))
```

- spaCy **spaCy offers the fastest syntactic parser available on the market today.**

#Use-case: Dependency parsing of a sentence.

```
import spacy

nlp = spacy.load("en_core_web_sm")
sentence = "The cat chased the mouse"
doc = nlp(sentence)
for token in doc:
print(token.text, "-->", token.dep_)
```

- Polyglot is more than just an efficient library; it's a multilingual NLP library. It offers **word embeddings for over 130 languages** and supports tasks like named entity recognition and morphological analysis in multiple languages

#Use-case: Language detection.

```
from polyglot.detect import Detector

text = "Bonjour le monde!"
detector = Detector(text)
language = detector.language.code
print(language)
```

- scikit-learn provides developers with a **wide range of algorithms for building machine-learning models.** It offers many functions for the **bag-of-words method of creating features to tackle text classification problems.**

# Use-case: Text classification using TF-IDF and Support Vector Machine.

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.pipeline import make_pipeline

X_train = ["This is a positive sentence.", "This is a negative statement.", ...]
y_train = ["positive", "negative", ...]
model = make_pipeline(TfidfVectorizer(), SVC())
model.fit(X_train, y_train)
```

1. The Pattern allows **part-of-speech tagging, sentiment analysis, vector space modeling, SVM, clustering, n-gram search, and WordNet. You can use a DOM parser, a web crawler, and helpful APIs like Twitter or Facebook.**

```
# Use-case: Part-of-speech tagging.
from pattern.en import parse

text = "The sun shines brightly."
tagged_text = parse(text, relations=True, lemmata=True)
print(tagged_text)
```

Required Libraries for Audio Preprocessing Data

- **Librosa**

Librosa is a Python library for audio and music analysis. It provides functions for loading audio files, extracting features (e.g., mel spectrograms, MFCCs), and performing various preprocessing tasks.

- **PyDub**

PyDub is a Python library for audio manipulation. It provides functions for reading, manipulating, and saving audio files in various formats.

- **Soundfile**

Soundfile is a Python library for reading and writing sound files using the libsndfile library. It provides a simple interface for working with audio files in various formats

Required Libraries for image and video data.

OpenCV
OpenCV is often deployed for computer vision tasks like face detection, object detection, face recognition, image segmentation, and much more.(https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjE4LyBla2EAxXgb2wGHfwABxMQFnoECBUQAQ&url=https%3A%2F%2Fdocs.opencv.org%2F4.x%2Findex.html&usg=AOvVaw12FSpNqUqsz6MvWxXnCN3R&opi=89978449)

- **Scikit-Image**

Scikit-Image, which uses NumPy arrays as image objects, offers many different algorithms for segmentation, color space manipulation, geometric transformation, analysis, morphology, feature detection, and much more(https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwihol_gla2EAxWBXGwGHauVCWQQFnoECBMQAQ&url=https%3A%2F%2Fdevdocs.io%2Fscikit_learn%2F&usg=AOvVaw1uhnCYTl3aNWuNTESyTArZ&opi=89978449)

- **SciPy**

This image processing library is another great option if you're looking for a wide range of applications like image segmentation, convolution, reading images, face detection, feature extraction, and more.
(https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjR0InUIq2EAxWgUGwGHbpFAUYQFnoECBsQAQ&url=https%3A%2F%2Fdocs.scipy.org%2Fdoc%2Fscipy%2Freference%2Fstats.html&usg=AOvVaw10tRCONK2ao9I4hTVRvwig&opi=89978449)
4.PIL
python Imaging Library. With Pillow, you can carry out many processes in image processing like point operations, filtering, and manipulating.
5.Matplotlib
The image processing library is usually used for 2D visualizations like scatter plots, histograms, and bar graphs, but it has proven to be useful for image processing by effectively pulling information out of an image. It's important to note that Matplotlib doesn't support all file formats.
(https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiNvqS3ma2EAxVIfGwGHWvpASUQFnoECB8QAQ&url=https%3A%2F%2Fdevdocs.io%2Fmatplotlib~3.1%2F&usg=AOvVaw3-I3V1RV9k6MBgvctSgIZb&opi=89978449)
6.NUMPY
While NumPy is an open-source Python library used for numerical analysis, it can also be used for image processing tasks like image cropping, manipulating pixels, masking of pixel values, and more. NumPy contains a matrix and multi-dimensional arrays as data structures.
Reference (https://numpy.org/doc/1.26/reference/typing.html#mypy-plugin)

7.Pytorch
*PyTorch*library TorchVision contains the most common image transformations for computer vision. It also contains datasets and model
(https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwi98LaPmK2EAxUcTWwGHT-uDGIQFnoECBMQAQ&url=https%3A%2F%2Fpytorch.org%2Fdocs%2Fstable%2Ftorch.html&usg=AOvVaw04NQHgz0_vr9vrNT-BiRxn&opi=89978449)

- 8.Keras

Keras is an open-source deep learning library written in Python and designed for fast prototyping and experimentation with neural networks.(https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiuns71mK2EAxVTZWwGHUAVDsoQFnoECBUQAQ&url=https%3A%2F%2Fwww.tensorflow.org%2Fguide%2Fkeras&usg=AOvVaw3akioP6CTfXvcSruCMOeIG&opi=89978449
9.Tensorflow
TensorFlow represents computations as directed graphs called dataflow graphs, where nodes represent mathematical operations and edges represent tensors

https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiuns71mK2EAxVTZWwGHUAVDsoQFnoECBUQAQ&url=https%3A%2F%2Fwww.tensorflow.org%2Fguide%2Fkeras&usg=AOvVaw3akioP6CTfXvcSruCMOeIG&opi=89978449

Required libraries for numerical and categorical data Preprocessing

NumPy:

NumPy is a fundamental package for scientific computing in Python. It provides support for multidimensional arrays, mathematical functions, linear algebra operations, random number generation, and more.

.

- Pandas

Pandas is a powerful library for data manipulation and analysis in Python. It provides data structures like DataFrame and Series, which are ideal for working with structured data. (https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjUuby2mq2EAxWbxzgGHYMYBRAQFnoECBQQAQ&url=https%3A%2F%2Fdevdocs.io%2Fpandas~0.25%2F&usg=AOvVaw2EQup2IuXLuEkSzFsPqAbv&opi=89978449)

- **scikit-learn**

scikit-learn is a versatile machine learning library in Python that provides tools for data preprocessing, modelling, evaluation, and more.

- **SCIPY**

SciPy is a scientific computing library that builds on top of NumPy. It provides additional functionality for optimization, integration, interpolation, signal processing, and more.

- **Statsmodels:**

Statsmodels is a Python library for estimating and interpreting statistical models. It provides tools for regression analysis, time series analysis, hypothesis testing, and more.
(https://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjSuJz_mq2EAxUIn2MGHXUvBhMQFnoECBMQAQ&url=https%3A%2F%2F

devdocs.io%2Fstatsmodels%2F&usg=AOvVaw2kXmsY_wtsugSlUyfO0qVm&opi=89978449)

Data Augumentation

Data Augumentation
Data augmentation is a technique of artificially increasing the training set by creating modified copies of a dataset using existing data. It includes making minor changes to the dataset or using deep learning to generate new data points.
Augmented data is driven from original data with some minor changes. In the case of image augmentation, we make geometric and color space transformations (flipping, resizing, cropping, brightness, contrast) to increase the size and diversity of the training set.
Synthetic data is generated artificially without using the original dataset. It often uses DNNs (Deep Neural Networks) and GANs (Generative Adversarial Networks) to generate synthetic data.
Data Augumentation Techniques

1. Audio Data Augumentation
2. Text data Augumentation
3. Image data Augumentation

- Audio Data Augumentation

Noise injection: add gaussian or random noise to the audio dataset to improve the model performance.
Shifting: shift audio left (fast forward) or right with random seconds.
Changing the speed: stretches times series by a fixed rate.
Changing the pitch: randomly change the pitch of the audio.

- Text data Augumentation

Word or sentence shuffling: randomly changing the position of a word or sentence.
Word replacement: replace words with synonyms.
Syntax-tree manipulation: paraphrase the sentence using the same word.
Random word insertion: inserts words at random.
Random word deletion: deletes words at random.

- Image Data AugumentationI

Geometric transformations: randomly flip, crop, rotate, stretch, and zoom images. You need to be careful about applying multiple transformations on the same images, as this can reduce model performance.
Color space transformations: randomly change RGB color channels, contrast, and brightness.
Kernel filters: randomly change the sharpness or blurring of the image.
Random erasing: delete some part of the initial image.
Mixing images: blending and mixing multiple images.

https://research.aimultiple.com/wp-content/uploads/2021/04/data-augmentation-1648x928.png.webp
https://research.aimultiple.com/wp-content/uploads/2021/04/dataaugmention_image_alletranitons-1224x632.png.webp