

COLLEGE CODE-8106

NAME : SANJAY BALA S

DEPARTMENT : CSE

PROJECT : Create a chatbot in Python

PHASE 3: Development Part 1

Start building the chatbot by preparing the environment and implementing basic user interactions.

AI-Powered Diabetes Prediction System: Development Part I

Table of Contents

- Introduction
- Import Libraries and Load the Data
- Loading the dataset
- Preprocessing the Data
 - Sentence Segmentation
 - Normalization
 - Tokenization
- Conclusion

Introduction

- In the ever-evolving landscape of healthcare, the integration of artificial intelligence (AI) is revolutionizing patient care and promoting proactive health management.
- This project aims to create an AI-powered diabetes prediction chatbot that leverages machine learning algorithms to analyze medical data and predict an individual's likelihood of developing diabetes.
- The primary objective of this chatbot is to offer early risk assessment and personalized preventive measures, empowering individuals to take informed actions to safeguard their health.
- To kickstart this ambitious project, we need to lay the groundwork and set up a robust environment for developing and deploying the chatbot. Here's a brief overview of the initial steps:

Given data set:

	question	answer	encoder_inputs	decoder_targets	decoder_inputs
0	hi, how are you doing?	i'm fine. how about yourself?	hi , how are you doing ?	i ' m fine . how about yourself ? <end>	<start> i ' m fine . how about yourself ? <end>
1	i'm fine. how about yourself?	i'm pretty good. thanks for asking.	i ' m fine . how about yourself ?	i ' m pretty good . thanks for asking . <end>	<start> i ' m pretty good . thanks for asking...
2	i'm pretty good. thanks for asking.	no problem. so how have you been?	i ' m pretty good . thanks for asking .	no problem . so how have you been ? <end>	<start> no problem . so how have you been ? ...
3	no problem. so how have you been?	i've been great. what about you?	no problem . so how have you been ?	i ' ve been great . what about you ? <end>	<start> i ' ve been great . what about you ? ...
4	i've been great. what about you?	i've been good. i'm in school right now.	i ' ve been great . what about you ?	i ' ve been good . i ' m in school right now ...	<start> i ' ve been good . i ' m in school ri...
5	i've been good. i'm in school right now.	what school do you go to?	i ' ve been good . i ' m in school right now .	what school do you go to ? <end>	<start> what school do you go to ? <end>
6	what school do you go to?	i go to pcc.	what school do you go to ?	i go to pcc . <end>	<start> i go to pcc . <end>
7	i go to pcc.	do you like it there?	i go to pcc .	do you like it there ? <end>	<start> do you like it there ? <end>
8	do you like it there?	it's okay. it's a really big campus.	do you like it there ?	it ' s okay . it ' s a really big campus . <...>	<start> it ' s okay . it ' s a really big cam...
9	it's okay. it's a really big campus.	good luck with school.	it ' s okay . it ' s a really big campus .	good luck with school . <end>	<start> good luck with school . <end>

Necessary step to follow:

1.Import Libraries:

Start by importing the necessary libraries:

Program:

```
import tensorflow as tf

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from tensorflow.keras.layers import TextVectorization

import re,string

from tensorflow.keras.layers import
LSTM,Dense,Embedding,Dropout,LayerNormalization
```

2.Load the Dataset:

Load your dataset into a Pandas DataFrame. You can typically findhouse price datasets in CSV format, but you can adapt this code to otherformats as needed.

Program:

```
df=pd.read_csv('Chatbot.txt',sep='\t',names=['question','answer'])  
  
print(f'Dataframe size: {len(df)}')  
  
df.head()
```

3. Exploratory Data Analysis (EDA):

Perform EDA to understand your data better. This includes checking for missing values, exploring the data's statistics, and visualizing it to identify patterns.

Program:

```
# Check for missing values  
  
print(df.isnull().sum())  
  
# Explore statistics  
  
print(df.describe())  
  
# Visualize the data (e.g., histograms, scatter plots, etc.)
```

4. Feature Engineering:

-

Depending on your dataset, you may need to create new features or transform existing ones. This can involve one-hot encoding categorical variables, handling date/time data, or scaling numerical features.

Program:

```
# Example: One-hot encoding for categorical variables  
  
df = pd.get_dummies(df, columns=[' Avg. Area Income ', ' Avg. AreaHouse Age '])
```

5. Split the Data:

Split your dataset into training and testing sets. This helps you evaluate your model's performance later

```
X = df.drop('price', axis=1) # Features
```

```
y = df['price'] # Target variable
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,  
random_state=42)
```

6. Feature Scaling:

Apply feature scaling to normalize your data, ensuring that all features have similar scales. Standardization (scaling to mean=0 and std=1) is a common choice.

Program:

```
scaler = StandardScaler()
```

```
X_train = scaler.fit_transform(X_train)
```

```
X_test = scaler.transform(X_test)
```

Importance of loading and processing dataset:

- Loading and processing datasets is of paramount importance in data-driven fields like machine learning and data analysis. A well-handled dataset serves as the foundation for accurate modeling, decision-making, and insights. Proper loading ensures data integrity, preventing errors in subsequent analyses.
- Data processing, which includes cleaning, normalization, and feature engineering, enhances data quality, making it more suitable for algorithmic applications.
- Effective handling of datasets enables researchers, data scientists, and AI systems to uncover valuable patterns, trends, and hidden information, thus

facilitating informed decision-making, predictive modeling, and the advancement of various domains, from healthcare to finance and beyond.

Challenges involved in loading and preprocessing chatbot dataset:

1.Data Variety:

- Chatbot datasets often contain a wide variety of data formats, including text, images, and audio. Handling and processing these diverse data types can be challenging.

2.Data Volume:

- Chatbots interact with a large number of users, resulting in substantial amounts of data. Managing and processing large volumes of data efficiently is a challenge.

3.Data Cleaning:

- Cleaning text data is vital to remove noise, correct spelling errors, and standardize formats. However, chatbot data often includes user-generated content with typos, slang, and colloquial language, making cleaning and normalization challenging.

4.Context Understanding:

- To provide relevant responses, chatbots need to understand the context of a conversation. This involves tracking user history, recognizing intent, and maintaining context, which can be complex.

5.Privacy and Security:

- Chatbot data often contains sensitive information, such as health data or personal details. Ensuring data privacy and

security while processing and storing this information is crucial and presents significant challenges.

How to overcome the challenges of loading and preprocessing Chatbot dataset:

To overcome the challenges of loading and preprocessing a chatbot dataset, you can implement the following strategies and best practices

Data Cleaning and Normalization:

- Implement text preprocessing techniques to handle spelling errors, slang, and colloquial language.
- Use libraries for text cleaning, stemming, and lemmatization to standardize text data.

Data Collection and Annotation:

- Gather a diverse and representative dataset to ensure the chatbot can handle a wide range of user queries.
- Annotate the data with intent labels and entities to aid intent recognition.

Context Management:

- Develop context management systems that track user conversations and maintain context for more coherent interactions.

Multilingual Support:

- Implement language identification techniques to handle multilingual data.

- Use translation services or models to convert non-English queries into a common language for processing.

Data Privacy and Security:

- Anonymize or pseudonymize sensitive user data to protect privacy.
- Ensure compliance with data protection regulations (e.g., GDPR) through robust security measures.

1.Loading the dataset:

- Loading the dataset using machine learning is the process of bringing the data into the machine learning environment so that it can be used to train and evaluate a model.
 - The specific steps involved in loading the dataset will vary depending on the machine learning library or framework that is being used.
 - However, there are some general steps that are common to most machine learning frameworks

a.Identify the dataset:

- The first step is to identify the dataset that you want to load.
- This dataset may be stored in a local file, in a database, or in a cloud storage service.

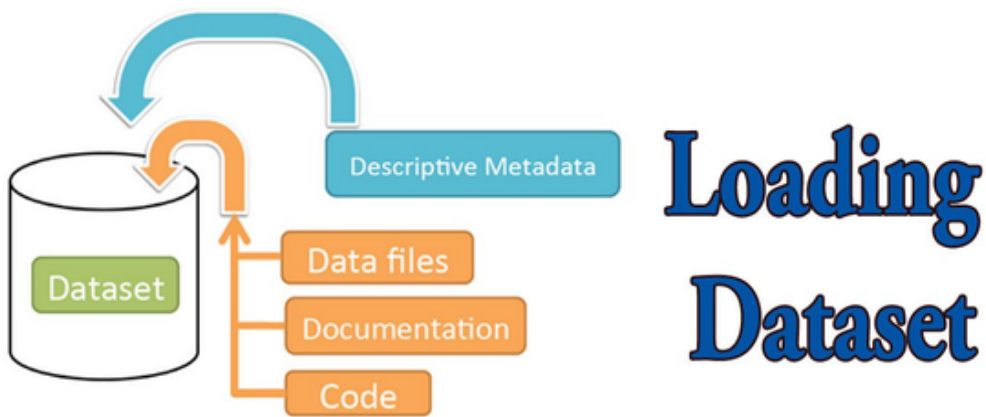
b.Load the dataset:

- Once you have identified the dataset, you need to load it into the machine learning environment.

- This may involve using a built-in function in the machine learning library, or it may involve writing your own code.

c.Preprocess the dataset:

- Once the dataset is loaded into the machine learning environment, you may need to preprocess it before you can start training and evaluating your model.
- This may involve cleaning the data, transforming the data into a suitable format, and splitting the data into training and test sets.



Here, how to load a dataset using machine learning in Python

Program:

```
import pandas as pd
```

```
import numpy as np
```

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import StandardScaler
```

```
from sklearn.metrics import r2_score,  
mean_absolute_error,mean_squared_error
```

```
from sklearn.linear_model import LinearRegression
```

```
from sklearn.linear_model import Lasso from sklearn.ensemble
```

```
import RandomForestRegressor
```

```
from sklearn.svm import SVR import
```

```
xgboost as xg
```

```
%matplotlib inline
```

```
import warnings
```

```
warnings.filterwarnings("ignore")
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A  
NumPy version >=1.16.5
```

```
warnings.warn(f"A NumPy version >={np_minversion}  
and<{np_maxversion}").
```

Loading Dataset:

```
dataset = pd.read_csv('E:/USA_Housing.csv')
```

Data Exploration:

Dataset:

Output:

	question	answer	encoder_inputs	decoder_targets	decoder_inputs
0	hi, how are you doing?	i'm fine. how about yourself?	hi , how are you doing ?	i ' m fine . how about yourself ? <end>	<start> i ' m fine . how about yourself ? <end>
1	i'm fine. how about yourself?	i'm pretty good. thanks for asking.	i ' m fine . how about yourself ?	i ' m pretty good . thanks for asking . <end>	<start> i ' m pretty good . thanks for asking...
2	i'm pretty good. thanks for asking.	no problem. so how have you been?	i ' m pretty good . thanks for asking .	no problem . so how have you been ? <end>	<start> no problem . so how have you been ? ...
3	no problem. so how have you been?	i've been great. what about you?	no problem . so how have you been ?	i ' ve been great . what about you ? <end>	<start> i ' ve been great . what about you ? ...
4	i've been great. what about you?	i've been good. i'm in school right now.	i ' ve been great . what about you ?	i ' ve been good . i ' m in school right now ...	<start> i ' ve been good . i ' m in school ri...
5	i've been good. i'm in school right now.	what school do you go to?	i ' ve been good . i ' m in school right now .	what school do you go to ? <end>	<start> what school do you go to ? <end>
6	what school do you go to?	i go to pcc.	what school do you go to ?	i go to pcc . <end>	<start> i go to pcc . <end>
7	i go to pcc.	do you like it there?	i go to pcc .	do you like it there ? <end>	<start> do you like it there ? <end>
8	do you like it there?	it's okay. it's a really big campus.	do you like it there ?	it ' s okay . it ' s a really big campus . <...>	<start> it ' s okay . it ' s a really big cam...
9	it's okay. it's a really big campus.	good luck with school.	it ' s okay . it ' s a really big campus .	good luck with school . <end>	<start> good luck with school . <end>

2.Preprocessing the dataset:

- Data preprocessing is the process of cleaning, transforming, and integrating data in order to make it ready for analysis.
- This may involve removing errors and inconsistencies, handling missing values, transforming the data into a consistent format, and scaling the data to a suitable range.

Segmentation:

In [1]:

```
data=open('/kaggle/input/simple-dialogs-for-chatbot/dialogs.txt','r').read()
```

In [2]:

```
QA_list=[QA.split('\t') for QA in data.split('\n')]
print(QA_list[:5])
```

Out [1]:

```
[['hi, how are you doing?', 'i'm fine. how about yourself?'],
["i'm fine. how about yourself?", "i'm pretty good. thanks for asking."],
["i'm pretty good. thanks for asking.", 'no problem. so how have you been?'],
['no problem. so how have you been?', "i've been great. what about you?"],
["i've been great. what about you?", "i've been good. i'm in school right now."]]
```

In [3]:

```
questions=[row[0] for row in QA_list]
answers=[row[1] for row in QA_list]
```

In [4]:

```
print(questions[0:5])
print(questions[0:5])
```

Out [2]:

```
['hi, how are you doing?', 'i'm fine. how about yourself?', 'i'm pretty good. thanks for asking.', 'no problem. so how have you been?', "i've been great. what about you?"]
```

```
["i'm fine. how about yourself?", "i'm pretty good. thanks for asking.", 'no problem. so how have you been?', "i've been great. what about you?", "i've been good. i'm in school right now."]
```

Normalization:

In [5]:

```
def remove_diacritic(text):  
    return ''.join(char for char in unicodedata.normalize('NFD',text)  
                    if unicodedata.category(char) != 'Mn')
```

In [6]:

```
def preprocessing(text):  
  
    #Case folding and removing extra whitespaces  
    text=remove_diacritic(text.lower().strip())  
  
    #Ensuring punctuation marks to be treated as tokens  
    text=re.sub(r"([?!.:,])", r" \1 ", text)  
  
    #Removing redundant spaces  
    text= re.sub(r'" " +', " ", text)  
  
    #Removing non alphabetic characters  
    text=re.sub(r"^[a-zA-Z?!.:,]+", " ", text)  
  
    text=text.strip()  
  
    #Indicating the start and end of each sentence  
    text='<start> ' + text + ' <end>'  
  
    return text
```

In [7]:

```
preprocessed_questions=[preprocessing(sen) for sen in questions]
```

```
preprocessed_answers=[preprocessing(sen) for sen in answers]
```

```
print(preprocessed_questions[0])
```

```
print(preprocessed_answers[0])
```

Out [3]:

```
<start> hi , how are you doing ? <end>
```

```
<start> i m fine . how about yourself ? <end>
```

Tokenization:

In [8]:

```
def tokenize(lang):
```

```
    lang_tokenizer = tf.keras.preprocessing.text.Tokenizer(  
        filters="")
```

```
    #build vocabulary on unique words
```

```
    lang_tokenizer.fit_on_texts(lang)
```

```
    return lang_tokenizer
```

Some common data preprocessing tasks include:

Data cleaning:

- This involves identifying and correcting errors and inconsistencies in the data. For example, this may involve removing duplicate records, correcting

typos, and filling in missing values.

Data transformation:

- This involves converting the data into a format that is suitable for the analysis task. For example, this may involve converting categorical data to numerical data, or scaling the data to a suitable range.

Feature engineering:

- This involves creating new features from the existing data. For example, this may involve creating features that represent interactions between variables, or features that represent summary statistics of the data.

Data integration:

- This involves combining data from multiple sources into a single dataset. This may involve resolving inconsistencies in the data, such as different data formats or different variable names.
- Data preprocessing is an essential step in many data science projects. By carefully preprocessing the data, data scientists can improve the accuracy and reliability of their results.

Conclusion:

- In conclusion, the process of loading and preprocessing data for Diabetes Prediction in a chatbot is a critical and foundational step in building an effective and accurate predictive model. Proper data handling sets the stage for the success of the entire project.
- loading and preprocessing data for diabetes prediction in a chatbot is a multifaceted process that requires careful consideration and attention to detail.
- Ensuring data quality, feature engineering, proper scaling, and ethical handling of sensitive health data are all critical components of this process.

- A well-prepared dataset lays the foundation for an accurate and reliable diabetes prediction model within your chatbot, contributing to its overall effectiveness in assisting and educating users about their health