Analysis Of ChatBot with Different Datasets using Python

Example 1 : By Export a WhatsApp Chat :

you'll have downloaded a TXT file that contains the chat history of a WhatsApp conversation. If you don't have a WhatsApp account or don't want to work with your own conversational data, then you can download a sample chat export below:

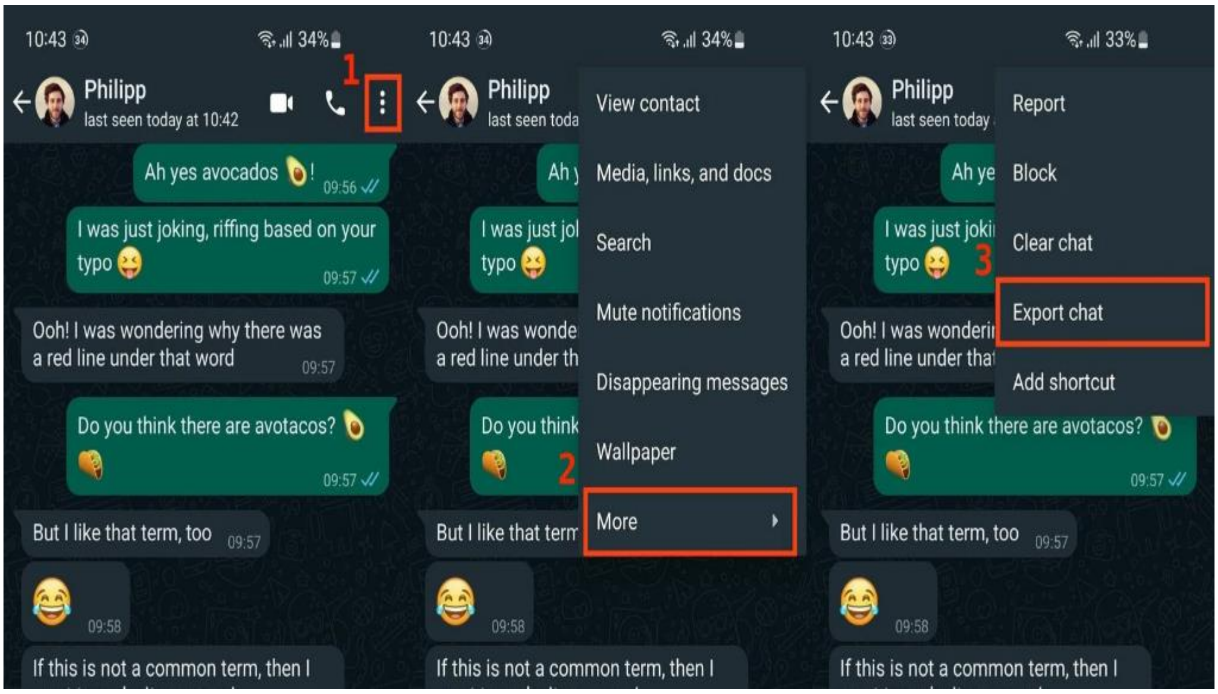
Python Program:

```
from chatterbot import ChatBot

from chatterbot.trainers import ListTrainer

chatbot = ChatBot("Chatpot")

trainer = ListTrainer(chatbot)

trainer.train(["Hi", "Welcome, friend 🤗"])

trainer.train(["Are you a plant?", "No, I'm the pot below the plant!"])

exit_conditions = (":q", "quit", "exit")

while True:

   query = input("> ")

   if query in exit_conditions:

      break

   else:

      printf("{chatbot.get_response(query)}")
```

To export the history of a conversation that you've had on WhatsApp, you need to open the conversation on your phone. Once you're on the conversation screen, you can access the export menu:

1. Click on the three dots (⁝) in the top right corner to open the main menu.
2. Choose More to bring up additional menu options.

Select Export chat to create a TXT export of your conversation.



Ah yes avocados 🥑! 09:56
I was just joking, riffing based on your typo 😜 09:57
Ooh! I was wondering why there was a red line under that word 09:57
Do you think there are avotacos? 🥑 🌮 09:57
But I like that term, too 09:57
😂 09:58
If this is not a common term, then I

- ➢ Once you've clicked on Export chat, you need to decide whether or not to include media, such as photos or audio messages. Because your chatbot is only dealing with text, select WITHOUT MEDIA. Then, you can declare where you'd like to send the file.
- ➢ In this example, you saved the chat export file to a Google Drive folder named Chat exports. You'll have to set up that folder in your Google Drive before you can select it as an option. Of course, you don't need to use Google Drive.
- ➢ Once that's done, switch back to your computer. Find the file that you saved, and download it to your machine.

Specifically, you should save the file to the folder that also contains bot.py and rename it chat.txt. Then, open it with your favorite text editor to inspect the data that you received :

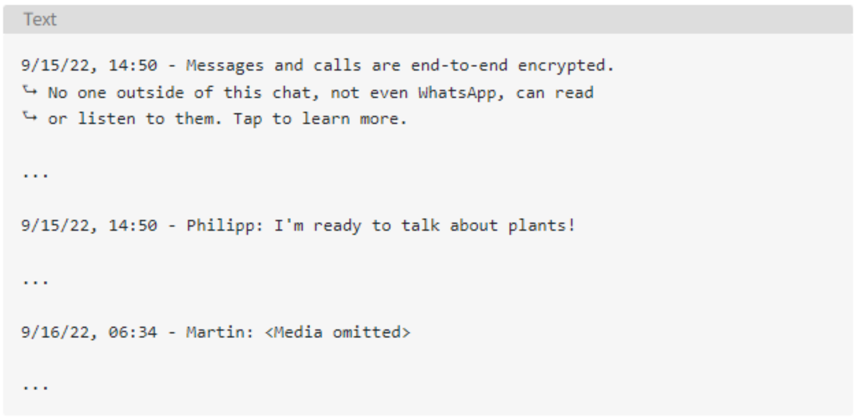
Text

```
9/15/22, 14:50 - Messages and calls are end-to-end encrypted.
↳ No one outside of this chat, not even WhatsApp, can read
↳ or listen to them. Tap to learn more.
9/15/22, 14:49 - Philipp: Hi Martin, Philipp here!
9/15/22, 14:50 - Philipp: I'm ready to talk about plants!
9/15/22, 14:51 - Martin: Oh that's great!
9/15/22, 14:52 - Martin: I've been waiting for a good convo about
↳ plants for a long time
9/15/22, 14:52 - Philipp: We all have.
9/15/22, 14:52 - Martin: Did you know they need water to grow?
...
```

- ➢ ChatterBot uses complete lines as messages when a chatbot replies to a user message.In the case of this chat export, it would therefore include all the message metadata. That means your friendly pot would be studying the dates, times, and usernames! Not exactly great conversation fertilizer.
- ➢ To avoid this problem, you'll clean the chat export data before using it to train your chatbot.

Clean Your Chat Export :

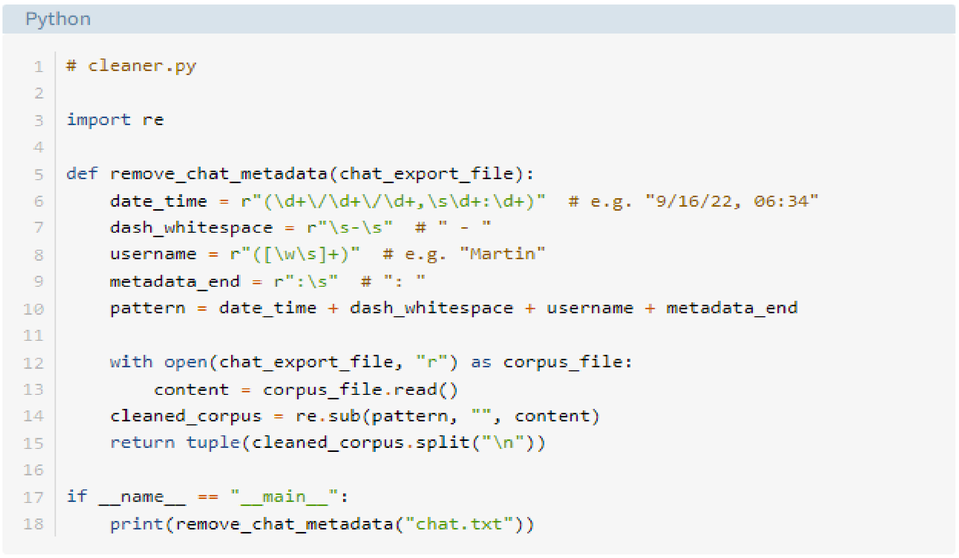
- ➢ Most data that you'll use to train your chatbot will require some kind of cleaning before it can produce useful results. It's just like the old saying goes:
  - ▪ Garbage in, garbage out (Source)
- ➢ Take some time to explore the data that you're working with and to identify potential issues:



Text

```
9/15/22, 14:50 - Messages and calls are end-to-end encrypted.
↳ No one outside of this chat, not even WhatsApp, can read
↳ or listen to them. Tap to learn more.

...

9/15/22, 14:50 - Philipp: I'm ready to talk about plants!

...

9/16/22, 06:34 - Martin: <Media omitted>

...
```

Open up a new Python file to preprocess your data before handing it to ChatterBot for training. Start by reading in the file content and removing the chat metadata:

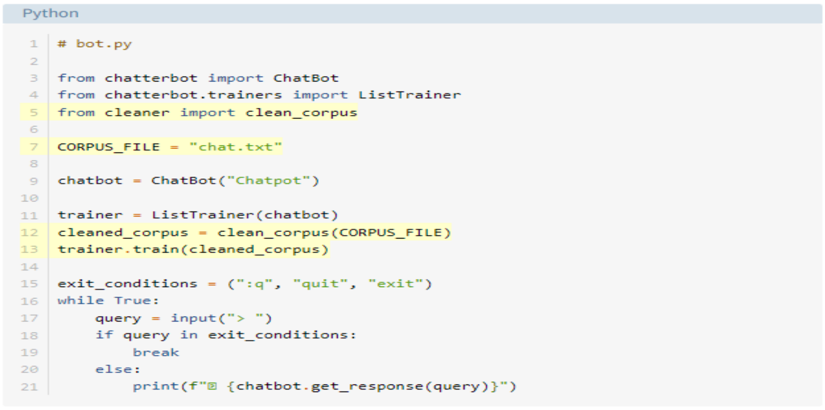


Python

```python
# cleaner.py

import re

def remove_chat_metadata(chat_export_file):
    date_time = r"(\d+\/\d+\/\d+,\s\d+:\d+)"  # e.g. "9/16/22, 06:34"
    dash_whitespace = r"\s-\s"  # " - "
    username = r"([\w\s]+)"  # e.g. "Martin"
    metadata_end = r":\s"  # ": "
    pattern = date_time + dash_whitespace + username + metadata_end

    with open(chat_export_file, "r") as corpus_file:
        content = corpus_file.read()
    cleaned_corpus = re.sub(pattern, "", content)
    return tuple(cleaned_corpus.split("\n"))

if __name__ == "__main__":
    print(remove_chat_metadata("chat.txt"))
```

Train Your Chatbot on Custom Data and Start Chatting :

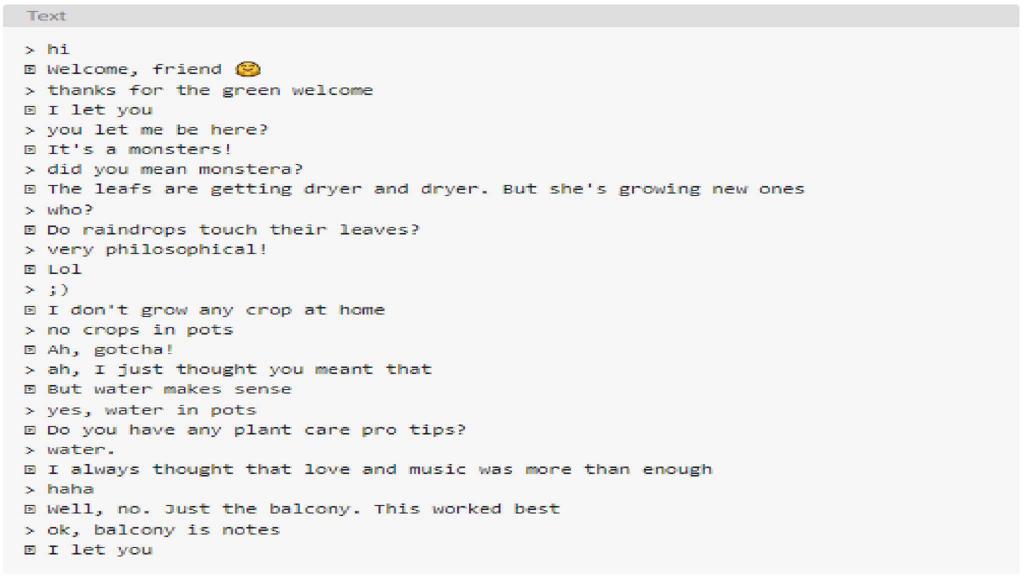
- ➢ Open up bot.py and include calls to your cleaning functions in the code:



Python

```python
# bot.py

from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer
from cleaner import clean_corpus

CORPUS_FILE = "chat.txt"

chatbot = ChatBot("Chatpot")

trainer = ListTrainer(chatbot)
cleaned_corpus = clean_corpus(CORPUS_FILE)
trainer.train(cleaned_corpus)

exit_conditions = (":q", "quit", "exit")
while True:
    query = input("> ")
    if query in exit_conditions:
        break
    else:
        print(f"⊡ {chatbot.get_response(query)}")
```

Your chatbot has increased its range of responses based on the training data that you fed to it. As you might notice when you interact with your chatbot, the responses don’t always make a lot of sense.



```
Text

> hi
⊡ Welcome, friend 😊
> thanks for the green welcome
⊡ I let you
> you let me be here?
⊡ It's a monsters!
> did you mean monstera?
⊡ The leafs are getting dryer and dryer. But she's growing new ones
> who?
⊡ Do raindrops touch their leaves?
> very philosophical!
⊡ Lol
> ;)
⊡ I don't grow any crop at home
> no crops in pots
⊡ Ah, gotcha!
> ah, I just thought you meant that
⊡ But water makes sense
> yes, water in pots
⊡ Do you have any plant care pro tips?
> water.
⊡ I always thought that love and music was more than enough
> haha
⊡ Well, no. Just the balcony. This worked best
> ok, balcony is notes
⊡ I let you
```

Example 2 : Import a Sample Dialog Document 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.
i've been good. i'm in school right now.	what school do you go to?
what school do you go to?	i go to pcc.
i go to pcc.	do you like it there?
do you like it there?	it's okay. it's a really big campus.
it's okay. it's a really big campus.	good luck with school.
good luck with school.	thank you very much.
how's it going?	i'm doing well. how about you?
i'm doing well. how about you?	never better, thanks.
never better, thanks.	so how have you been lately?
so how have you been lately?	i've actually been pretty good. you?
i've actually been pretty good. you?	i'm actually in school right now.
i'm actually in school right now.	which school do you attend?
which school do you attend?	i'm attending pcc right now.
i'm attending pcc right now.	are you enjoying it there?
are you enjoying it there?	it's not bad. there are a lot of people there.
it's not bad. there are a lot of people there.	good luck with that.
```

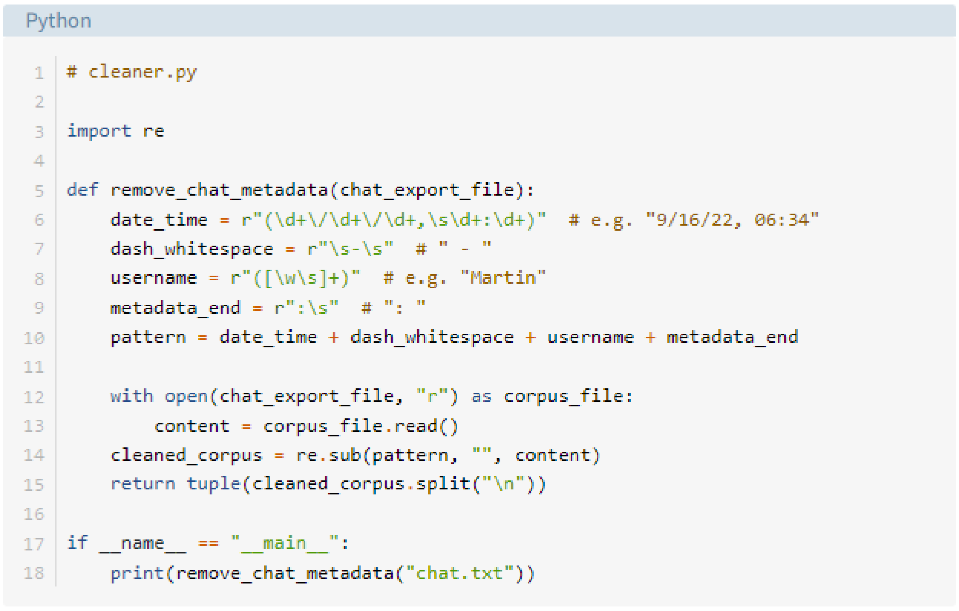
you can download a sample chat export below:

Python Program:

```
from chatterbot import ChatBot

from chatterbot.trainers import ListTrainer

chatbot = ChatBot("Chatpot")

trainer = ListTrainer(chatbot)

trainer.train(["hi, how are you doing?"])

trainer.train(["i'm fine. how about yourself?”,”i'm pretty good. thanks for asking."])

exit_conditions = (":q", "quit", "exit")

while True:

  query = input("> ")

  if query in exit_conditions:

    break

  else:

    printf("{chatbot.get_response(query)}")
```

Clean Your Chat Export :

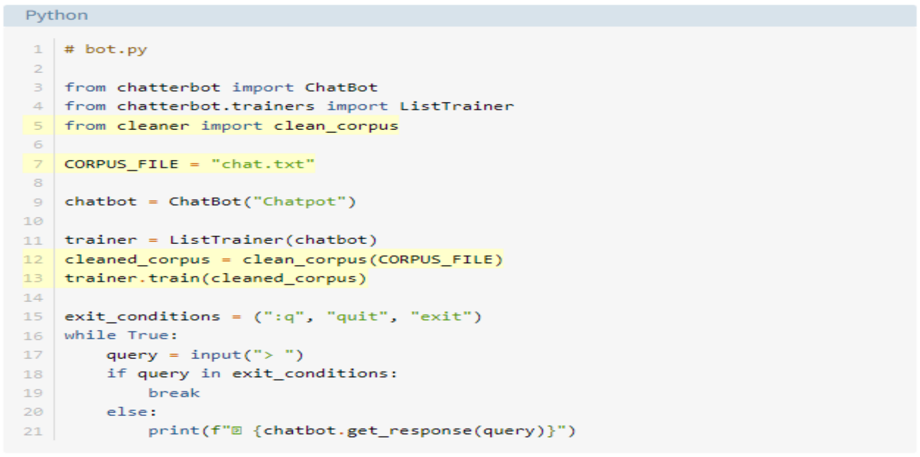
Open up a new Python file to preprocess your data before handing it to ChatterBot for training. Start by reading in the file content and removing the chat metadata:



```python
# cleaner.py

import re

def remove_chat_metadata(chat_export_file):
    date_time = r"(\d+\/\d+\/\d+,\s\d+:\d+)"  # e.g. "9/16/22, 06:34"
    dash_whitespace = r"\s-\s"  # " - "
    username = r"([\w\s]+)"  # e.g. "Martin"
    metadata_end = r":\s"  # ": "
    pattern = date_time + dash_whitespace + username + metadata_end

    with open(chat_export_file, "r") as corpus_file:
        content = corpus_file.read()
    cleaned_corpus = re.sub(pattern, "", content)
    return tuple(cleaned_corpus.split("\n"))

if __name__ == "__main__":
    print(remove_chat_metadata("chat.txt"))
```

Train Your Chatbot on Custom Data and Start Chatting :

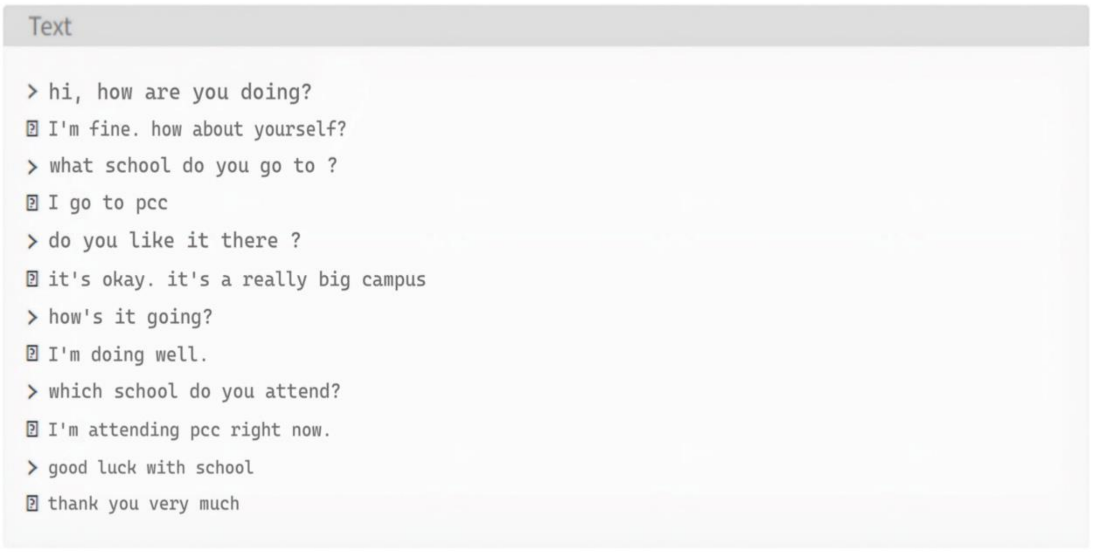
Open up bot.py and include calls to your cleaning functions in the code:



Python

```python
# bot.py

from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer
from cleaner import clean_corpus

CORPUS_FILE = "chat.txt"

chatbot = ChatBot("Chatpot")

trainer = ListTrainer(chatbot)
cleaned_corpus = clean_corpus(CORPUS_FILE)
trainer.train(cleaned_corpus)

exit_conditions = (":q", "quit", "exit")
while True:
    query = input("> ")
    if query in exit_conditions:
        break
    else:
        print(f"□ {chatbot.get_response(query)}")
```

Your chatbot has increased its range of responses based on the training data that you fed to it. As you might notice when you interact with your chatbot, the responses don’t always make a lot of sense.



Text

```text
> hi, how are you doing?
□ I'm fine. how about yourself?
> what school do you go to ?
□ I go to pcc
> do you like it there ?
□ it's okay. it's a really big campus
> how's it going?
□ I'm doing well.
> which school do you attend?
□ I'm attending pcc right now.
> good luck with school
□ thank you very much
```

Example 3: Import a Sample Dialog Document 2:

this friday, i'm throwing a party.	oh really? i didn't know that.
oh really? i didn't know that.	are you serious?
are you serious?	i haven't heard anything about it.
i haven't heard anything about it.	can you make it?
can you make it?	what time does it start?
what time does it start?	the party starts at 8.
the party starts at 8.	yeah, i think i'll go.
yeah, i think i'll go.	am i going to see you there?
what's going on?	nothing really, you?
nothing really, you?	i'm throwing a party next saturday.
i'm throwing a party next saturday.	is that right?
is that right?	yeah, are you going to come?
yeah, are you going to come?	i'm sorry, i can't.
i'm sorry, i can't.	why not?
why not?	i don't really want to.
i don't really want to.	well, why don't you?
well, why don't you?	i hate going to parties.
i hate going to parties.	well, that's okay.
well, that's okay.	yeah, sorry.
what's up?	nothing, how about you?
nothing, how about you?	next saturday, i'm going to have a party.

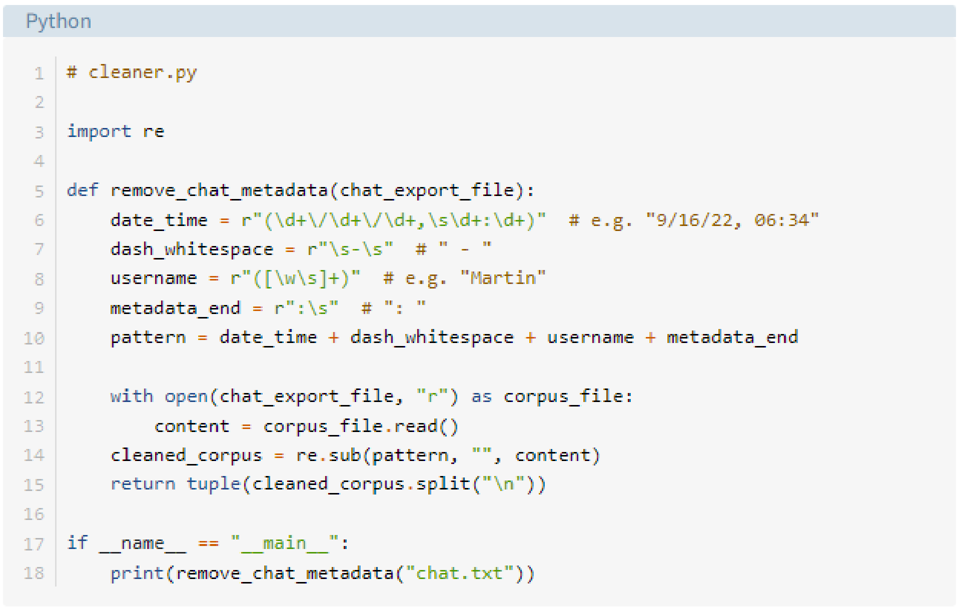
you can download a sample chat export below:

Python Program:

```
from chatterbot import ChatBot

from chatterbot.trainers import ListTrainer

chatbot = ChatBot("Chatpot")

trainer = ListTrainer(chatbot)

trainer.train(["this friday, i'm throwing a party."])

trainer.train(["are you serious?”,”i haven't heard anything about it."])

exit_conditions = (":q", "quit", "exit")
while True:

   query = input("> ")

   if query in exit_conditions:
        break
    else:
        printf("{chatbot.get_response(query)}")
```

Clean Your Chat Export :

Open up a new Python file to preprocess your data before handing it to ChatterBot for training. Start by reading in the file content and removing the chat metadata:

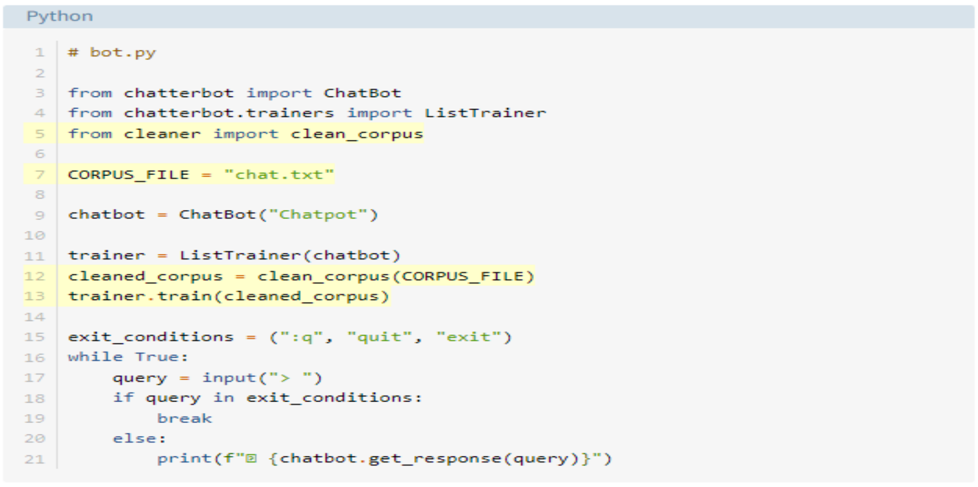


Python

```python
# cleaner.py

import re

def remove_chat_metadata(chat_export_file):
    date_time = r"(\d+\/\d+\/\d+,\s\d+:\d+)"  # e.g. "9/16/22, 06:34"
    dash_whitespace = r"\s-\s"  # " - "
    username = r"([\w\s]+)"  # e.g. "Martin"
    metadata_end = r":\s"  # ": "
    pattern = date_time + dash_whitespace + username + metadata_end

    with open(chat_export_file, "r") as corpus_file:
        content = corpus_file.read()
    cleaned_corpus = re.sub(pattern, "", content)
    return tuple(cleaned_corpus.split("\n"))

if __name__ == "__main__":
    print(remove_chat_metadata("chat.txt"))
```

Train Your Chatbot on Custom Data and Start Chatting :

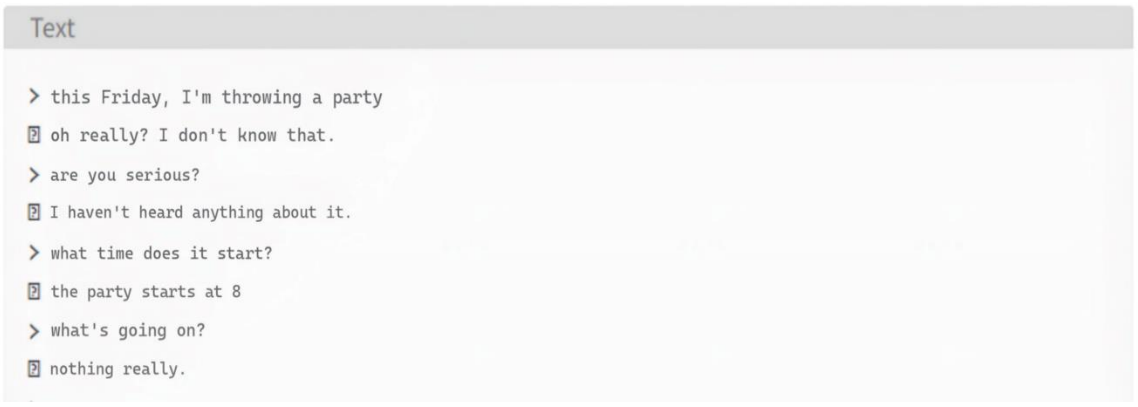
Open up bot.py and include calls to your cleaning functions in the code:



Python

```python
# bot.py

from chatterbot import ChatBot
from chatterbot.trainers import ListTrainer
from cleaner import clean_corpus

CORPUS_FILE = "chat.txt"

chatbot = ChatBot("Chatpot")

trainer = ListTrainer(chatbot)
cleaned_corpus = clean_corpus(CORPUS_FILE)
trainer.train(cleaned_corpus)

exit_conditions = (":q", "quit", "exit")
while True:
    query = input("> ")
    if query in exit_conditions:
        break
    else:
        print(f"⊡ {chatbot.get_response(query)}")
```

Your chatbot has increased its range of responses based on the training data that you fed to it. As you might notice when you interact with your chatbot, the responses don’t always make a lot of sense.



```
> this Friday, I'm throwing a party
? oh really? I don't know that.
> are you serious?
? I haven't heard anything about it.
> what time does it start?
? the party starts at 8
> what's going on?
? nothing really.
```