**Next word prediction:**

import random

from collections import defaultdict

class NextWordPredictor:

def \_\_init\_\_(self, n):

self.n = n # Order of the Markov chain

self.model = defaultdict(list)

def train\_model(self, text):

words = text.split()

for i in range(len(words) - self.n):

context = tuple(words[i:i + self.n])

next\_word = words[i + self.n]

self.model[context].append(next\_word)

def predict\_next\_word(self, context):

possible\_next\_words = self.model.get(context, [])

if possible\_next\_words:

return random.choice(possible\_next\_words)

else:

return None

if \_\_name\_\_ == "\_\_main\_\_":

# Example usage

text\_data = "This is a simple example of a next word prediction model. Implementing this model can be educational and fun."

nwp\_model = NextWordPredictor(n=2) # Use a trigram model (n=2)

# Train the model

nwp\_model.train\_model(text\_data)

# Predict the next word given a context

context = tuple("word prediction".split())

predicted\_word = nwp\_model.predict\_next\_word(context)

if predicted\_word:

print(f"Predicted next word: {predicted\_word}")

else:

print("No prediction available.")