Certamente! Ecco la presentazione MARP completa che include tutte le slide del modulo pratico basato sull'articolo di Real Python, con gli esempi di codice pertinenti per ogni sezione e le note del presentatore.

```
---
marp: true
theme: default
paginate: true
---

# Elaborazione del Linguaggio Naturale (NLP) 🤖
## Modulo Pratico con NLTK 🛠️

<!--
Nota per il presentatore: Slide di titolo. Presenta il modulo come un'immersione pratica nell'NLP usando NLTK in Python. Sottolinea che si vedranno esempi concreti per capire come funziona il preprocessing e l'analisi del testo.
-->

---

# Cosa Impareremo Oggi 🗺️

- **Installazione** ⚙️: Come preparare l'ambiente.
- **Tokenizzazione** 🔪: Dividere il testo in frasi e parole.
- **Stop Words** 🚫: Rimuovere parole comuni.
- **Stemming & Lemmatizzazione** 🌱: Normalizzare le parole.
- **POS Tagging** 🏷️: Identificare le parti del discorso.
- **Chunking** 🧩: Estrarre gruppi sintattici (es. sintagmi nominali).

<!--
Nota per il presentatore: Questa è la roadmap del modulo. Elenca i punti chiave che verranno trattati. Sottolinea che si partirà dalle basi (installazione, tokenizzazione) per poi passare a tecniche di analisi più complesse come il POS tagging e il chunking. Ogni punto sarà accompagnato da codice Python funzionante.
-->

---

# Installazione e Configurazione ⚙️

Prima di tutto, installiamo NLTK e scarichiamo le risorse necessarie.

```bash
# 1. Installa NLTK usando pip (in terminale)
$ pip install nltk
```
```

```python
# 2. Scarica i dati necessari (in uno script Python)
import nltk

# Scarica solo i pacchetti usati in questo modulo
# (esegui una volta sola)
try:
    nltk.data.find('tokenizers/punkt')
except nltk.downloader.DownloadError:
    nltk.download('punkt') # Tokenizer pre-addestrato
```

Tokenizzazione: Frasi e Parole 🔪📄➡️📄

Dividere il testo in unità significative.

```
from nltk.tokenize import sent_tokenize, word_tokenize

# Testo di esempio (da Dune)
testo = ("Muad'Dib learned rapidly because his first training was in how to learn. "
         "And the first lesson of all was the basic trust that he could learn. "
         "It's shocking to find how many people do not believe they can learn, "
         "and how many more believe learning to be difficult.")

# 1. Tokenizzazione in Frasi
lista_frasi = sent_tokenize(testo)
print("Frasi:")
print(lista_frasi)
# Output: ['...', '...', '...']

# 2. Tokenizzazione in Parole (include punteggiatura)
lista_parole = word_tokenize(testo)
print("\nParole:")
print(lista_parole)
# Output: ['Muad'Dib', 'learned', 'rapidly', ..., 'difficult', '.']
```

Rimozione delle Stop Words 🚫🗣️

Eliminare parole comuni (es. "il", "e", "a") che non aggiungono molto significato.

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

frase = "Sir, I protest. I am not a merry man!" # Esempio da Star Trek
parole = word_tokenize(frase)
print("Parole originali:", parole)

# Ottieni la lista di stop words in inglese
stop_words_eng = set(stopwords.words('english'))
# print(stop_words_eng) # Mostra alcune stop words

# Filtra le parole: mantieni solo quelle NON stop words
# Confronta in minuscolo per gestire maiuscole/minuscole
parole_filtrate = [w for w in parole if w.lower() not in stop_words_eng]

print("Parole filtrate:", parole_filtrate)
# Output: ['Sir', ',', 'protest', '.', 'merry', 'man', '!']
```

Stemming: Alla Radice della Parola 🌱✂️

Ridurre le parole alla loro radice (stem), anche se non è una parola reale. Utile per raggruppare termini correlati.

```
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize

stemmer = PorterStemmer() # Un algoritmo di stemming comune
testo = ("The crew of the USS Discovery discovered many discoveries. "
         "Discovering is what explorers do.")
parole = word_tokenize(testo)

print("Parola -> Stem (radice)")
for w in parole:
    print(f"{w} -> {stemmer.stem(w)}")

# Output parziale:
# Discovery -> discoveri
# discovered -> discov
```

Lemmatizzazione: Alla Forma Base 📚✅

Ridurre le parole alla loro forma base (lemma) usando un dizionario. Produce parole reali.

```
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize

lemmatizer = WordNetLemmatizer() # Usa il dizionario WordNet

# Esempio 1: Parola singola
print(f"Lemmatizzazione di 'scarves': {lemmatizer.lemmatize('scarves')}")
# Output: scarf (corretto!) vs stemmer -> scarv

# Esempio 2: Frase intera
frase = "The friends of DeSoto love scarves."
parole = word_tokenize(frase)
lemmi = [lemmatizer.lemmatize(w) for w in parole]
print(f"Lemmi della frase: {lemmi}")
# Output: ['The', 'friend', 'of', 'DeSoto', 'love', 'scarf', '.']
```

Lemmatizzazione con Contesto Grammaticale (POS) 🤔🏷️

La lemmatizzazione migliora specificando la parte del discorso (Part-of-Speech).

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

# Senza specificare il POS (assume sostantivo 'n' di default)
print(f"lemmatize('worst'): {lemmatizer.lemmatize('worst')}")
# Output: worst

# Specificando che è un aggettivo ('a')
# NOTA: WordNet usa codici specifici: 'a'=aggettivo, 'v'=verbo, 'n'=nome, 'r'=avverbio
print(f"lemmatize('worst', pos='a'): {lemmatizer.lemmatize('worst', pos='a')}")
# Output: bad (corretto! 'worst' è il superlativo di 'bad')
```

Part-of-Speech (POS) Tagging 🏷️✍️

Assegnare a ogni parola la sua categoria grammaticale (nome, verbo, aggettivo, ecc.).

```
from nltk import pos_tag, word_tokenize

# Frase di esempio (Carl Sagan)
frase = "If you wish to make an apple pie from scratch, you must first invent the universe."
parole = word_tokenize(frase)

# Esegui il POS tagging
taggati = pos_tag(parole) # Usa il tagger pre-addestrato di NLTK

print(taggati)
# Output: [('If', 'IN'), ('you', 'PRP'), ('wish', 'VBP'), ('to', 'TO'),
#          ('make', 'VB'), ('an', 'DT'), ('apple', 'NN'), ('pie', 'NN'), ...,
#          ('universe', 'NN'), ('.', '.')]
```

Chunking: Estrarre Sintagmi 🧩🏗️

Identificare gruppi di parole con significato sintattico (es. sintagmi nominali) usando regole sui POS tag.

```
import nltk
from nltk.tokenize import word_tokenize

frase = "It's a dangerous business, Frodo, going out your door." # Esempio da LoTR
parole = word_tokenize(frase)
pos = nltk.pos_tag(parole) # Prima facciamo il POS tagging

# Definiamo una grammatica per i Sintagmi Nominali (NP)
# Regola: (Opzionale:DT) seguito da (Zero o più:JJ) seguito da (Uno:NN)
grammatica = "NP: {<DT>?<JJ>*<NN>}"
parser_chunk = nltk.RegexpParser(grammatica) # Crea il parser

# Applica il parser alla frase taggata
tree = parser_chunk.parse(pos)
#tree.draw() # Apre una finestra con l'albero (utile in locale)

# Estrai e stampa i chunk NP trovati
print("Chunk NP trovati:")
for subtree in tree.subtrees(filter=lambda t: t.label() == 'NP'):
    chunk_parole = [token for token, tag in subtree.leaves()]
```

Ricapitolando 🏁

Abbiamo visto come usare NLTK per:

1. **Installare** e configurare l'ambiente.
2. **Tokenizzare** testo in frasi e parole (`sent_tokenize` , `word_tokenize`).
3. **Filtrare Stop Words** (`stopwords.words`).
4. **Stemming** (es. `PorterStemmer`).
5. **Lemmatizzazione** (`WordNetLemmatizer` , opz. con `pos=`).
6. **POS Tagging** (`pos_tag`).
7. **Chunking** con grammatiche (`RegexpParser`).

Questi sono blocchi fondamentali per costruire applicazioni NLP più complesse! 🧱

Prossimi Passi & Risorse 🚀

- **Sperimenta!** Prova con altri testi, anche in italiano (`stopwords.words('italian')`).
- **Approfondisci NLTK:** NER, Parsing completo, Classificazione...
- 📖 **NLTK Book:** nltk.org/book/ (Gratuito!)
- 📄 **Articolo Real Python:** Link all'articolo originale
- ❓ **Domande?**