CoGrammar

DS PORTFOLIO SESSION 10

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

Data Science Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(FBV: Mutual Respect.)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Open Classes. You can submit these questions here: **Open Class Questions**

Data Science Session Housekeeping cont.

- For all **non-academic questions**, please submit a query:

  www.hyperiondev.com/support

- Report a **safeguarding** incident:

  www.hyperiondev.com/safeguardreporting

- We would love your **feedback** on lectures: **Feedback on Lectures**

CoGrammar

Progression Criteria

✅ **Criterion 1: Initial Requirements**

- Complete 15 hours of Guided Learning Hours and the first four tasks within two weeks.

✅ **Criterion 2: Mid-Course Progress**

- Software Engineering: Finish 14 tasks by week 8.
- Data Science: Finish 13 tasks by week 8.

✅ **Criterion 3: Post-Course Progress**

- Complete all mandatory tasks by 24th March 2024.
- Record an Invitation to Interview within 4 weeks of course completion, or by 30th March 2024.
- Achieve 112 GLH by 24th March 2024.

✅ **Criterion 4: Employability**

- Record a Final Job Outcome within 12 weeks of graduation, or by 23rd September 2024.

CoGrammar

Recap of Week 11: Predictive Maintenance for Manufacturing Machinery

Machine Learning Application/System

- Uses ML algorithms in order to analyse sensor data and predict machine failure.

Supervised Machine Learning

- A category of ML where the model is trained on a labeled dataset, ultimately learning the relationship between input and corresponding target labels.

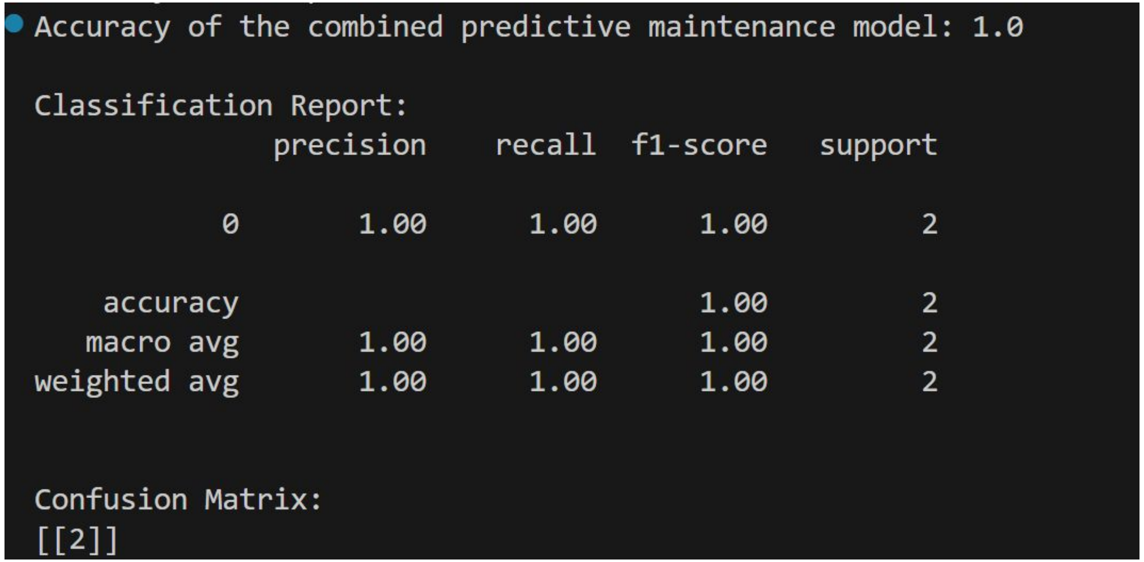
Feature Selection

- Picking relevant features from the sensor data is crucial for accurate predictions.

Evaluation Metrics

- Factors such as accuracy, precision, recall, and the confusion matrix.

Predictive Maintenance (Output)



```
Accuracy of the combined predictive maintenance model: 1.0

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         2

    accuracy                           1.00         2
   macro avg       1.00      1.00      1.00         2
weighted avg       1.00      1.00      1.00         2


Confusion Matrix:
[[2]]
```

CoGrammar

Back to FutureTech Industries

- **Background:** In the heart of the digital age, the "Daily Byte" stands as one of the most renowned online news platforms. With a vast array of articles spanning various topics, readers are spoilt for choice. However, with such an abundance of information, many readers feel overwhelmed, often missing out on articles that might resonate with their interests.

- **Challenge:** The editorial team, led by the visionary editor-in-chief, Ms. Penelope Wordsworth, has a challenge for the students. They've noticed a trend: readers who enjoy articles on specific topics tend to have a high likelihood of enjoying other articles with semantic similarities. With this observation, Ms. Wordsworth poses a question: "Can we guide our readers to articles they'll love, based on what they've previously enjoyed?"

FutureTech Industries

- **Objective:** You are tasked with creating ByteMatch, a recommendation system for the Daily Byte. Using the power of linear regression and natural language processing, they must sift through the vast sea of articles and find the hidden gems that each reader will cherish. By analysing the semantic content of articles, you will craft a system that ensures every reader feels like the Daily Byte was curated just for them.

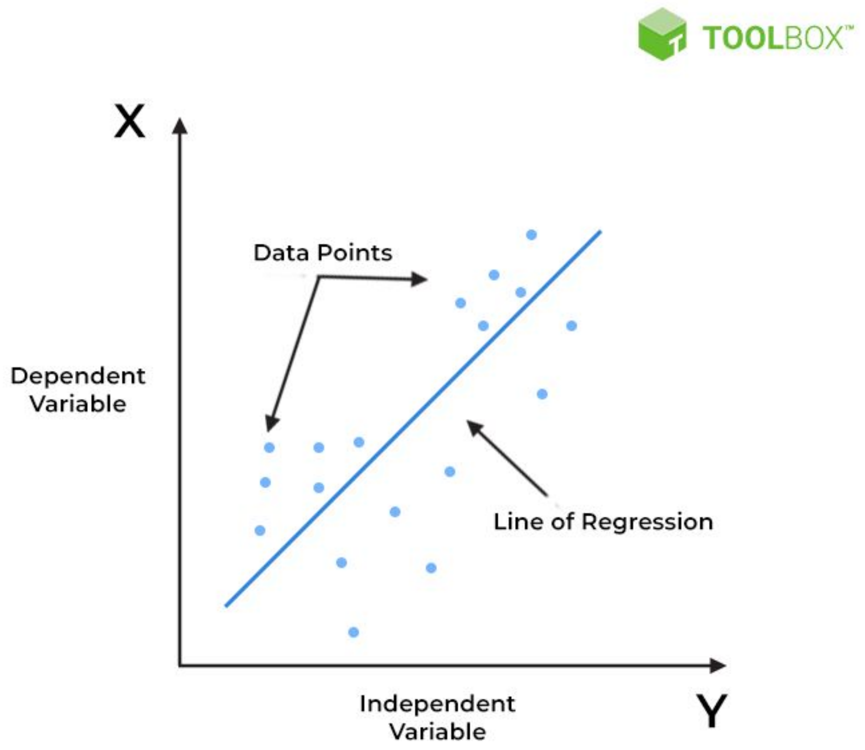
What is the primary goal of ByteMatch?

A. Enhancing graphics on the Daily Byte platform
B. Creating a recommendation system for the Daily Byte to guide readers to articles they'll love
C. Building a real-time news updating system
D. Developing an interactive user interface for article browsing

CoGrammar

Demo: Need for ByteMatch

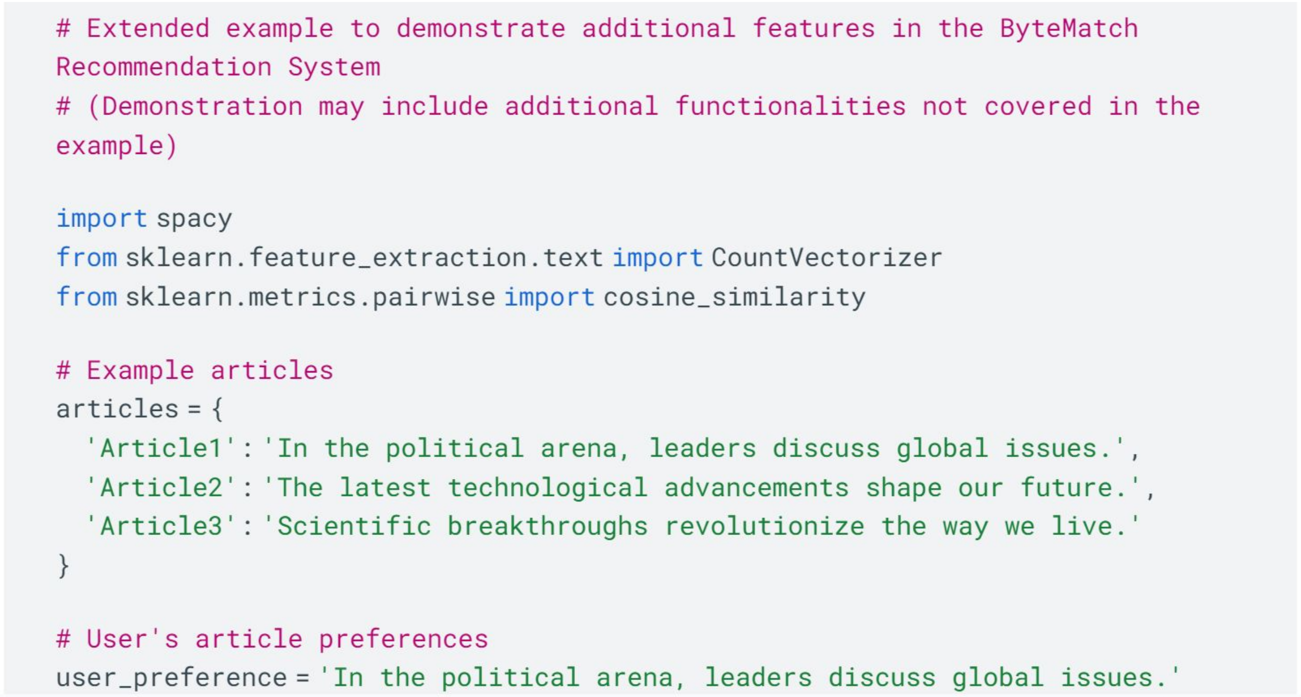
```
# Simple example to demonstrate the need for ByteMatch
reader_preferences = {'Politics': 0.8, 'Technology': 0.6, 'Science': 0.7}

def byte_match(recommendation_scores):
  sorted_recommendations = sorted(recommendation_scores.items(), key=lambda x: 
x[1], reverse=True)
  top_recommendation = sorted_recommendations[0][0]
  return f"Based on your preferences, we recommend checking out more articles on 
'{top_recommendation}'."

# Call the function
recommendation = byte_match(reader_preferences)
print(recommendation)
```



https://images.spiceworks.com/wp-content/uploads/2022/04/07040339/25-4.png

CoGrammar

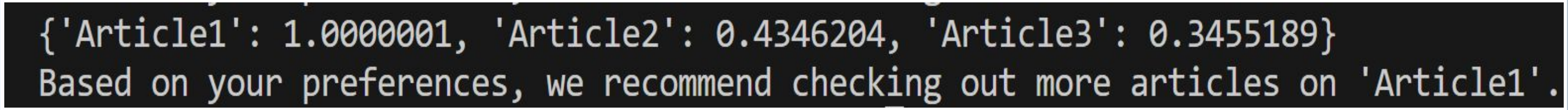
Demo: Recommendation system using vector similarities



```
# Extended example to demonstrate additional features in the ByteMatch
Recommendation System
# (Demonstration may include additional functionalities not covered in the
example)

import spacy
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

# Example articles
articles = {
  'Article1': 'In the political arena, leaders discuss global issues.',
  'Article2': 'The latest technological advancements shape our future.',
  'Article3': 'Scientific breakthroughs revolutionize the way we live.'
}

# User's article preferences
user_preference = 'In the political arena, leaders discuss global issues.'
```

```python
# Tokenization and word vector similarity calculation
nlp = spacy.load('en_core_web_sm')

def calculate_similarity(user_preference, articles):
  vectors = [nlp(article).vector for article in articles.values()]
  user_vector = nlp(user_preference).vector
  similarities = cosine_similarity([user_vector], vectors)[0]
  return dict(zip(articles.keys(), similarities))

# Call the function
recommendation_scores = calculate_similarity(user_preference, articles)
print(recommendation_scores)

# Call the ByteMatch function
recommendation = byte_match(recommendation_scores)
print(recommendation)
```

Demo: Recommendation System (Output)



```
{'Article1': 1.0000001, 'Article2': 0.4346204, 'Article3': 0.3455189}
Based on your preferences, we recommend checking out more articles on 'Article1'.
```

ByteMatch

By analysing the semantic content of articles, you will craft a system that ensures every reader feels like the "Daily Byte" was curated just for them.

Helpful sklearn modules:

spacy(for NLP)
scikit-learn
CountVectorizer
cosine_similarity

Important Concepts:

1. **Algorithm Efficiency:** Comprehending and leveraging the efficiency of the ByteMatch system will generally result in improved performance in text-processing tasks as opposed to complex algorithms.

2. **NLP:** Valuable for information extraction, summarizing and sentiment analysis, where patterns are taken from the meaning of the words instead of their byte sequences.

3. **Linear Regression:** In the context of NLP, this can be applied to understand patterns and relationships in text-based data, ultimately predicting outcomes based on features derived from the text.

Advanced Challenge:

- Apply NLP techniques, such as tokenization and sentiment analysis, to extract higher-level patterns and insights from the same text.

In the ByteMatch recommendation system, what is the primary goal?

A. Enhancing graphics on the Daily Byte platform
B. Creating a recommendation system to guide readers to articles they'll love
C. Building a real-time news updating system
D. Developing an interactive user interface for article browsing

How does ByteMatch use natural language processing to enhance the user experience on the Daily Byte platform?

A. By applying advanced graphics techniques
B. Through semantic analysis of articles and reader preferences
C. By incorporating real-time news updates
D. By providing an interactive user interface for article browsing.

Summary

Linear Regression

★ Linear regression is a statistical technique used in Natural Language Processing for analysing relationships between variables.

Natural Language Processing (NLP)

★ NLP involves the use of algorithms and computational models to process and understand human language.

Questions and Answers

Questions around the Case Study