MOD500 Decision Analysis with Artificial Intelligence Support

Enrico Riccardi[1]

Department of Energy Resources, University of Stavanger (UiS).[1]

Oct 27, 2024

Decision trees learning

It is a simple model for supervised classification

Each decision nodes performs a Boolean test (binary split version)

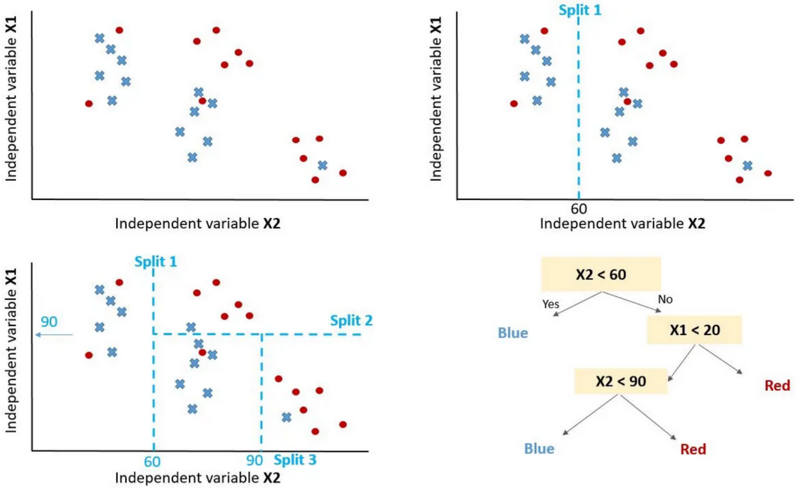
They are build out of DATA!

At each split, we perform the slip that reduce entropy the most.

REMINDER

We need to provide a label!

Decision tree outcome



Decision trees

Pseudo-code

- Compute the entropy of each feature (myopic approach)
- Pick the feature with the maximum entropy
- For each value of the selected feature, compute the entropy of the new population
- Compute the Information Gain by splitting the dataset
- Repeat for the number of desired splits

Decision trees in Python

```python
"""
MOD500 tutorial: Decision tree minimal example

"""
import numpy as np
from matplotlib import pyplot as plt

from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier, plot_tree

iris = load_iris()
X = iris.data
y = iris.target

clf = DecisionTreeClassifier(max_leaf_nodes=10,
                             criterion='entropy')
clf.fit(X, y)

plot_tree(clf, proportion=True, filled=True)

plt.show()
```

Tutorial [4]

Generate (at least) 4 different probability distributions

Make a meaningful label, and then make a decision tree from the data generated

(Use the given template to sort out Python programming part if you need)

A language model is a probability distribution over sequences of words [1].

Jurafsky and Martin: Speech and Language Processing, 2023

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | x < i)$$

P(Twinkle twinkle little star, how I wonder what you are.) = 0.99
P(Twinkle twinkle little moon, how I wonder what you are.) = 0.75
P(Twinkle twinkle little star, how I what you are.) = 0.3
P(Are you what I wonder I how star, little twinkle, twinkle.) = 0.02

Language models

A language model is a probability distribution over sequences of words [1].

Jurafsky and Martin: Speech and Language Processing, 2023

$$p(x_1, \dots, x_n) = \prod_{i=1}^{n} p(x_i | x_{< i})$$

P(Twinkle twinkle little star, how I wonder what you are.) = 0.99
P(Twinkle twinkle little moon, how I wonder what you are.) = 0.75
P(Twinkle twinkle little star, how I what you are.) = 0.3
P(Are you what I wonder I how star, little twinkle, twinkle.) = 0.02

Language models

A language model is a probability distribution over sequences of words [1].

Jurafsky and Martin: Speech and Language Processing, 2023

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | x{<}\ i)$$

P(Twinkle twinkle little star, how I wonder what you are.) = 0.99
P(Twinkle twinkle little moon, how I wonder what you are.) = 0.75
P(Twinkle twinkle little star, how I what you are.) = 0.3
P(Are you what I wonder I how star, little twinkle, twinkle.) = 0.02

Vector representations

Vector representation

- tokenization
- word2vec

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Vector representations

Vector representation

- tokenization
- word2vec

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Vector representations

Vector representation

- tokenization
- word2vec

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Sparse Vector representations

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Table of co-occurrences of the words in Wikipedia

- One dimension for each word $->$ long
- Many values are 0 $->$ sparse

Cherry picking

pointing to individual cases that seem to confirm a particular position while ignoring a significant portion of similar cases or data that may contradict that position.

Sparse Vector representations

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Table of co-occurrences of the words in Wikipedia

- One dimension for each word $->$ long
- Many values are 0 $->$ sparse

Cherry picking

pointing to individual cases that seem to confirm a particular position while ignoring a significant portion of similar cases or data that may contradict that position.

Sparse Vector representations

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Table of co-occurrences of the words in Wikipedia

- One dimension for each word $->$ long
- Many values are 0 $->$ sparse

Cherry picking

pointing to individual cases that seem to confirm a particular position while ignoring a significant portion of similar cases or data that may contradict that position.

Sparse Vector representations

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Table of co-occurrences of the words in Wikipedia

- One dimension for each word $->$ long
- Many values are 0 $->$ sparse

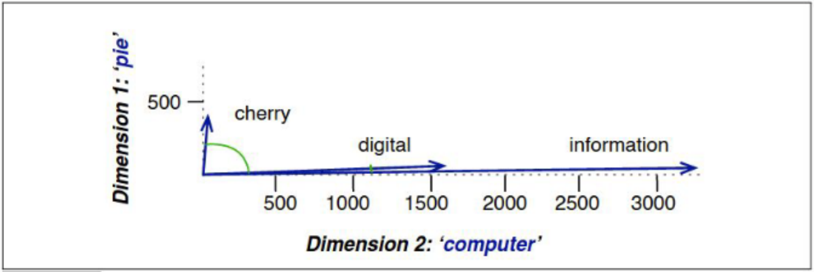
Cherry picking

pointing to individual cases that seem to confirm a particular position while ignoring a significant portion of similar cases or data that may contradict that position.

Vector similarity

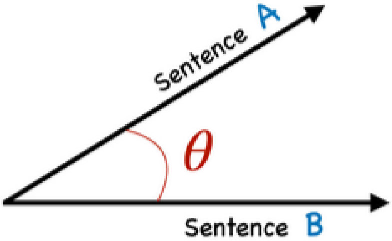
Metric alert

How close are two words?



Cosine similarity

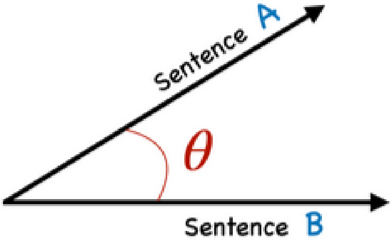
A popular metric to measure the similarity between sentences



$$cosinesimilarity = S_C(A, B) = cos(\theta) = \frac{A \cdot B}{||A||\ ||B||}$$

Cosine similarity

A popular metric to measure the similarity between sentences



$$cosinesimilarity = S_C(A, B) = cos(\theta) = \frac{A \cdot B}{||A|| \; ||B||}$$

Transformers

- A neural network designed to explicitly take into account the long-range dependencies between words
- Sequence-to-sequence models that transform an input vectors $(x_1, ..., x_n)$ to some output vectors $(y_1, ..., y_n)$ of the same length
- Transformers are made up of stacks of transformer blocks.
- Attention allows to directly extract and use information from arbitrarily long contexts

Transformers

- A neural network designed to explicitly take into account the long-range dependencies between words
- Sequence-to-sequence models that transform an input vectors (x_1, ..., x_n) to some output vectors (y_1, ..., y_n) of the same length
- Transformers are made up of stacks of transformer blocks.
- Attention allows to directly extract and use information from arbitrarily long contexts

Transformers

- A neural network designed to explicitly take into account the long-range dependencies between words
- Sequence-to-sequence models that transform an input vectors $(x_1, \ldots, x_n)$ to some output vectors $(y_1, \ldots, y_n)$ of the same length
- Transformers are made up of stacks of transformer blocks.
- Attention allows to directly extract and use information from arbitrarily long contexts

Transformers

- A neural network designed to explicitly take into account the long-range dependencies between words
- Sequence-to-sequence models that transform an input vectors (x_1, ..., x_n) to some output vectors (y_1, ..., y_n) of the same length
- Transformers are made up of stacks of transformer blocks.
- Attention allows to directly extract and use information from arbitrarily long contexts

Transformers

- A neural network designed to explicitly take into account the long-range dependencies between words
- Sequence-to-sequence models that transform an input vectors (x_1, ..., x_n) to some output vectors (y_1, ..., y_n) of the same length
- Transformers are made up of stacks of transformer blocks.
- Attention allows to directly extract and use information from arbitrarily long contexts

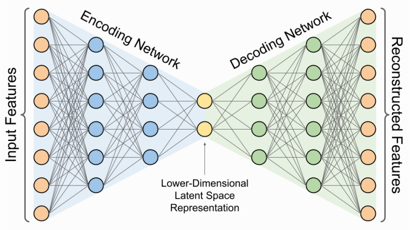
Encode & decode

Encoder model

From an input sequence to a contextualised representation of each input element

Decoder model

From contextualised representations to a task-specific output sequence



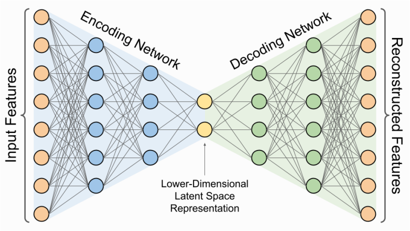
Encode & decode

Encoder model

From an input sequence to a contextualised representation of each input element

Decoder model

From contextualised representations to a task-specific output sequence



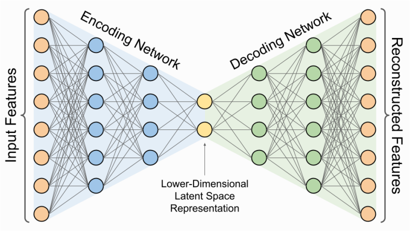
Encode & decode

Encoder model

From an input sequence to a contextualised representation of each input element

Decoder model

From contextualised representations to a task-specific output sequence



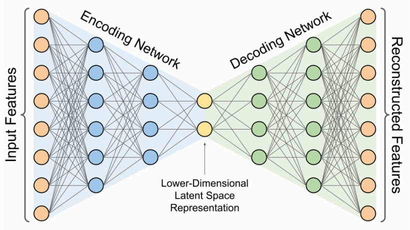
Encode & decode

Encoder model

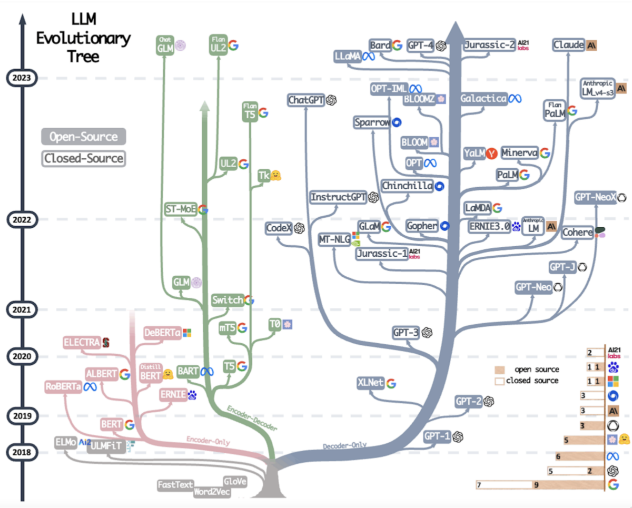
From an input sequence to a contextualised representation of each input element

Decoder model

From contextualised representations to a task-specific output sequence



LLMs



LLM Evolutionary Tree
2023
2022
2021
2020
2019
2018
Open-Source
Closed-Source
ChatGLM
Flan UL2
Flan T5
UL2
Tk
ST-MoE
GLM
Switch
mT5
T0
T5
BART
DeBERTa
ELECTRA
ALBERT
DistillBERT
RoBERTa
ERNIE
BERT
ELMo
ULMFiT
FastText
GloVe
Word2Vec
Encoder-Only
Encoder-Decoder
Decoder-Only
LLaMA
Bard
GPT-4
Jurassic-2
Claude
ChatGPT
OPT-IML
BLOOMZ
Galactica
Anthropic LM v4-s3
Flan PaLM
Sparrow
BLOOM
YaLM
Minerva
OPT
PaLM
Chinchilla
InstructGPT
GPT-NeoX
LaMDA
CodeX
GLaM
Gopher
ERNIE3.0
Anthropic LM
Cohere
MT-NLG
Jurassic-1
GPT-J
GPT-Neo
GPT-3
XLNet
GPT-2
GPT-1
open source
closed source

RAGs

Reducing hallucinations

Retrieval Augmented Generation

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | x < i; [Q:])$$

where [Q:] is additional information

Combining different information sources with different (assumed) reliability

RAGs

Reducing hallucinations

Retrieval Augmented Generation

$$p(x_1, ..., x_n) = \prod_{i=1}^{n} p(x_i | x{<}\ i; [Q :])$$

where [Q:] is additional information

Combining different information sources with different (assumed) reliability

RAGs

Reducing hallucinations

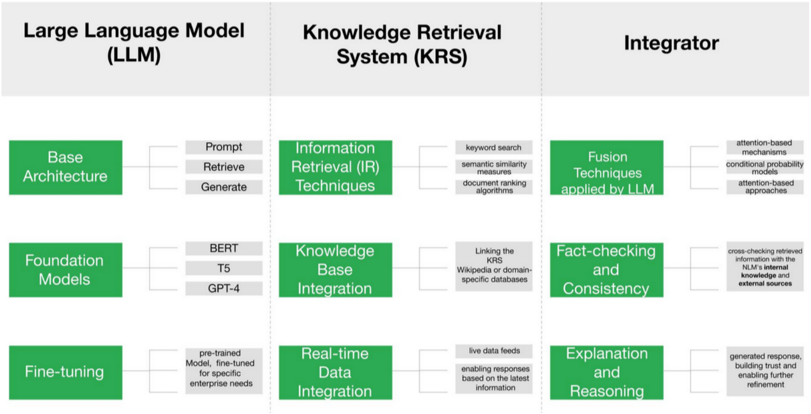
Retrieval Augmented Generation

$$p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x < i; [Q :])$$

where [Q:] is additional information

Combining different information sources with different (assumed) reliability

RAGS


Large Language Model (LLM)
Knowledge Retrieval System (KRS)
Integrator
Base Architecture
Prompt
Retrieve
Generate
Information Retrieval (IR) Techniques
keyword search
semantic similarity measures
document ranking algorithms
Fusion Techniques applied by LLM
attention-based mechanisms
conditional probability models
attention-based approaches
Foundation Models
BERT
T5
GPT-4
Knowledge Base Integration
Linking the KRS Wikipedia or domain-specific databases
Fact-checking and Consistency
cross-checking retrieved information with the NLM's internal knowledge and external sources
Fine-tuning
pre-trained Model, fine-tuned for specific enterprise needs
Real-time Data Integration
live data feeds
enabling responses based on the latest information
Explanation and Reasoning
generated response, building trust and enabling further refinement

Learn more!

- Speech and Language Processing, Chapter 9 (Transformers) and 10 (Large Language Models), Dan Jurafsky and James H. Martin 17
- The Illustrated Transformer, Jay Alammar