24. Работа с данни

(и още)

1 юни 2026

Честит празник, fam

Що е то data science, ML и AI?

- **Data Science** - Фокусът е извличане на ценна информация от наличните данни с цел вземане на информирани решения. (Финансови анализи, Класификация и таргетиране на клиенти, тендеции при клиенти)
- **Machine Learning** - Предоставя начин на самата машина да синтезира данни, да се учи от тях и да използва ключовата информация, за да се подобрява. (Рекомендър системи, търсачки, финансови модели)
- **Artificial Intelligence** - Фокуса е да разрешим на една машина да извършва комплексни интелектуални задачи, както би правил един човек (Чатботове, GO, Гласови асистенти, Co-pilot)

Примери

- https://chat.openai.com/chat
- https://github.com/features/copilot
- https://replit.com/site/ghostwriter
- https://v0.dev/
- https://claude.ai/
- https://bolt.new/
- https://lovable.dev/
- https://notebooklm.google.com/ <- Брутално за учене
- https://www.cursor.com/ -> VS Code Fork that is >>>> than copilot

Защо точно Питон?

- Изключително популярен с много голямо общество
- Разполага с множество интеграции
- Голямо разнообразие от библиотеки, за ДС, МЛ, АИ
- Ключови библиотеки:
    - NumPy
    - Pandas
    - Polars
    - Matplotlib
    - Seaborn
    - scikit-learn
    - PyTorch
    - Tensorflow
    - ... и много други

NumPy & Pandas 🐼

- Вашите най-добри приятели, когато става въпрос за боравене с данни
- **NumPy** - далеч по-скучно - Numerical Python
- **Pandas** - не е кръстено на панди за съжаление, а на Panel Data

NumPy

- NumPy е основната библиотека, когато говорим за всякакъв тип сметки в Python
- Дава възможност за супер ефективна работа с масиви от всякакви измерения
- `numpy.ndarray` – хомогенен n-мерен масив
  - 1 измерение – вектор
  - 2 измерения – матрица
  - по-общо – тензор
  - фиксирани дължини на размерностите
  - елементите са индексирани от tuples
- Обикновено се ползва като `import numpy as np`

NumPy array

- Като списък, ама с екстри
- Защо NumPy array вместо списък?
  - Всяка стойност в NumPy списъка има фиксиран тип
  - Използват contiguous memory -> непрекъсната памет
  - Позволяват векторни операции
  - Cache locality
- Къде се използва NumPy?
  - Като MATLAB, но в Python.
  - Pandas
  - Изображения (Matplotlib)
  - ML

Защо n-мерност?

- Колко-мерни са картинките
  - ... 2
  - ... ако са монохромни (черно-бели)
  - Цветовете обикновено са отделна размерност
- Колко-мерни са филмите?
- А звука?
  - ... а ако е стерео?

numpy broadcasting

- Операциите в numpy обикновено работят поелементно
- Но скаларите могат да се “разширяват” до тензори
- NumPy всъщност разширява всякакви размерности (не само скалари)
- Работи отдясно наляво
- масиви с различни форми могат да се комбинират, ако всяка ос е съвместима.

NumPy array операции

- `np.zeros()`, `np.ones()`, `np.identity()`, `np.random.randint()`
- `np.char.add()`, `np.char.multiply()`, `np.char.center()`
- `np.char.split()`, `np.char.splitlines()`, `np.char.strip()`
- `np.char.join()`, `np.char.replace()`
- `reshape()`, `transpose()`

NumPy array операции

- `reshape()`, `transpose()`
- `np.add()`, `np.subtract()`, `np.multiply()`, `np.divide()`
- `np.matmul()`
  - … или оператор `@`
- `np.nditer`

Въпрос?

```
array([[1., 1., 1., 1., 1.],
       [1., 0., 0., 0., 1.],
       [1., 0., 9., 0., 1.],
       [1., 0., 0., 0., 1.],
       [1., 1., 1., 1., 1.]])
```

Pandas

- Основата му е NumPy
- Манипулации на данни, анализиране, почистване
- Позволява свеждане във вид за използване на някакъв тип ML модели или алгоритъм

Основни структури в Pandas

- **Pandas Series**
  - едноизмерен масив с индекс
  - индексът обикновено е целочислен (но може да бъде от друг тип)
  - Хомогенен
- **Pandas Dataframe**
  - Двуизмерен масив (практически просто таблица)
  - Всяка колона има собствен тип
  - Можем да достъпваме колоните по име
  - Всяка колона е series
  - ... но индексът е общ за всички колони

Series

- Създаване - от списъци, от NumPy масиви, от речници
- Можем да променяме индекса
- Манипулации

DataFrame

- Dataframe - практически колкото искате Series обекта в един. Създаваме таблица от такива
- Начини за създаване
- Манипулации

DataFrame ключови операции

- `columns`, `index`
- `head()`, `tail()`
- `describe()`
- `iloc`, `loc`
- `groupby()`, `sort_values()`
- `fillna()`, `dropna()`, `isna()`

Видове файлови формати (основните)

- .xlsx, .csv
- Pickle
- Parquet
- Feather
- hdf/json
- msgpack

Защо pandas?

- Добре се припокрива с идеята
- Всеки ред е отделно наблюдение (експеримент)
- Всяка колона - харакреистика (feature) на наблюдението

scikit-learn

- Библиотека с алгоритми за машинно самообучение
- ... и още

sklearn.datasets

- Модул с примерни данни
  - `load_wine` — classic wine classification dataset
  - `load_breast_cancer` — binary classification
  - `load_digits` — small 8×8 handwritten digits
  - `fetch_20newsgroups` — text classification
  - `make_classification` — synthetic classification data
  - `make_blobs` — synthetic clustering data

sklearn.datasets пример

```
>>> from sklearn.datasets import load_iris
>>> import pandas as pd
>>> iris_data = load_iris()
>>> iris_data.feature_names
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
>>> type(iris_data.data)
numpy.ndarray
>>> pd.DataFrame(iris_data.data, columns=iris_data.feature_names).head()
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2
```

sklearn пример

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

iris = load_iris() # Load dataset
X, y = iris.data, iris.target

X_train, X_test, y_train, y_test = train_test_split( # Train/test split
    X, y, test_size=0.2, random_state=42
)

clf = RandomForestClassifier() # Train a model
clf.fit(X_train, y_train)

preds = clf.predict(X_test) # Evaluate
print("Accuracy:", accuracy_score(y_test, preds))
```

sklearn пример (2)

- Зарждаме данни
- Разделяме ги...
  - Тренировъчни (с тях ще тренираме модела)
  - Тестови (с тях ще оценим как работи модела)
  - Прави се, за да избегнем претрениране (overfitting)
  - (изчислително мощни модели могат да научат данните наизуст)
- Трениране модела
- Печатаме метрика за точността му
  - Ползваме accuracy
  - Верни_предсказвания / всички_предсказвания
  - ... има и други метрики

Класификация VS регресия

- Двата най-често срещани типове модели
- Ние направихме класификатор
- Класификация
  - Дискретно (една измежду няколко категории)
  - Например - коя е следващата буква в текст
- Регресията
  - Непрекъснато предсказване
  - Например - как ще изглежда сигнал в бъдещето

Малко за НЛП-то

- Клон в ИИ, който се фокусира в това да разреши на машините да отговарят на “човешкия” език по ценен начин.
- Защо това е важно?
- Примери: Преводачи, Виртуални асистенти, Сърч Енджини, Сентимент Анализ, Филтриране на имейли, Чатботове и тн.

Embeddings

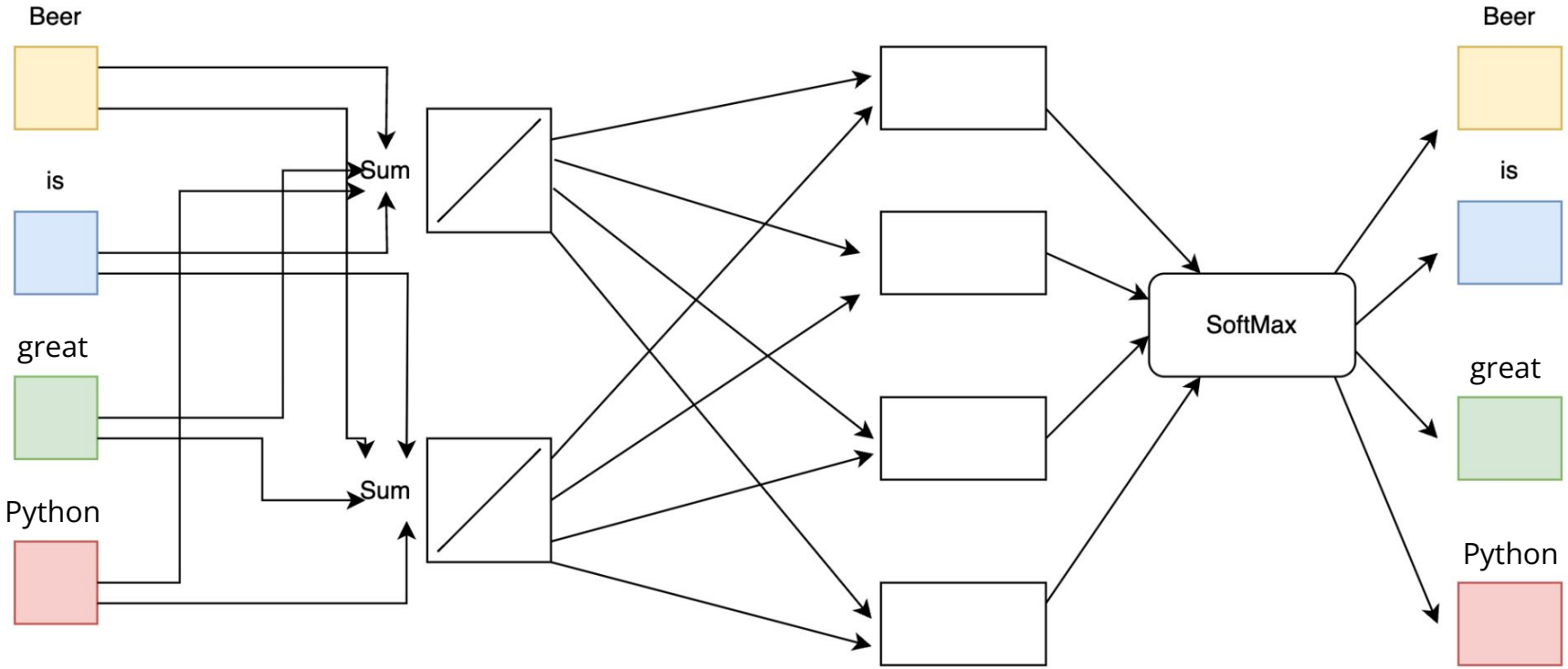
- Числено представяне на думи, където всяка дума е репрезентирана от вектор в непрекъснато векторно пространство. Обикновено се състоят от десетки или стотици измерения.
- Улавянето на семантична връзка между думите.
- Улавяне на контекстуална информация в изречения.
- Позволяват ни да намалим нужните пространства, за да работим с думи.
- Използват Невронни мрежи.

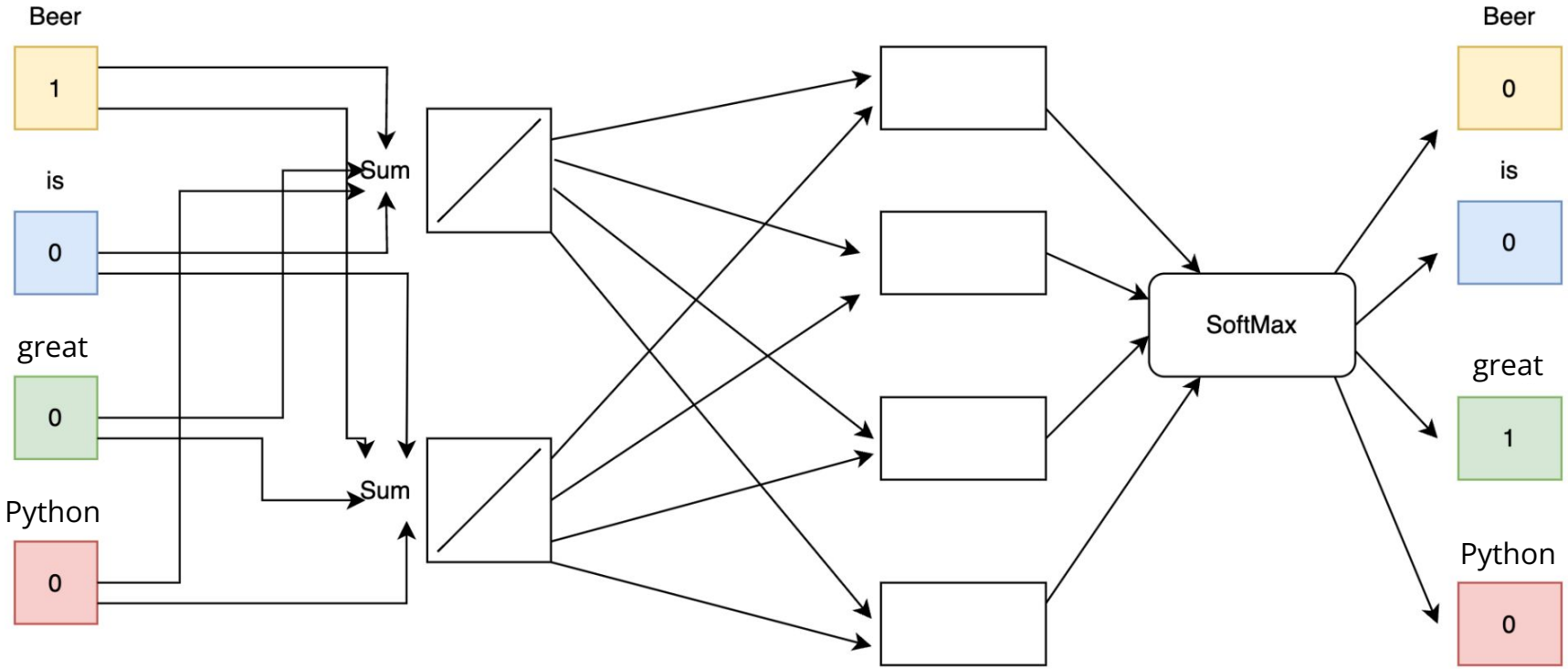
Word2Vec

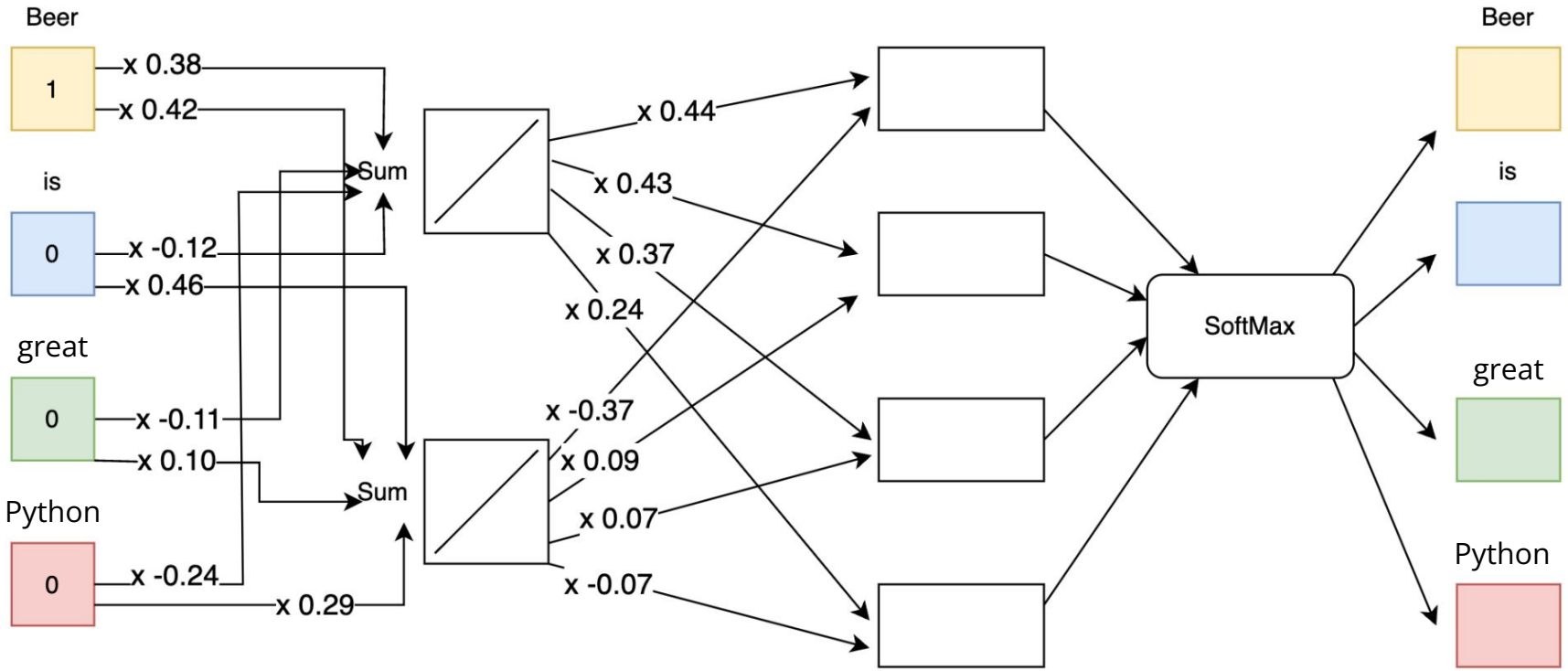
- Появява се през 2013 - Естествено от Гугъл (Томас Миколов и неговия екип)
- Continuous bag of words:
  - Дадено ни е изречението: Пайтън е много ___ език.
  - Искаме да попълним празното място с подходяща дума, например- “як” или “готин” (може и “труден”).
- Skip-Gram:
  - Като CBOW, ама наобратно. Дадена ни е думата “бира” .
  - Искаме да предскажем възможните думи наоколо - “крафт”, “лагер”, “пия”.
- TOKENS -> https://tokens-lpj6s2duga-ew.a.run.app/

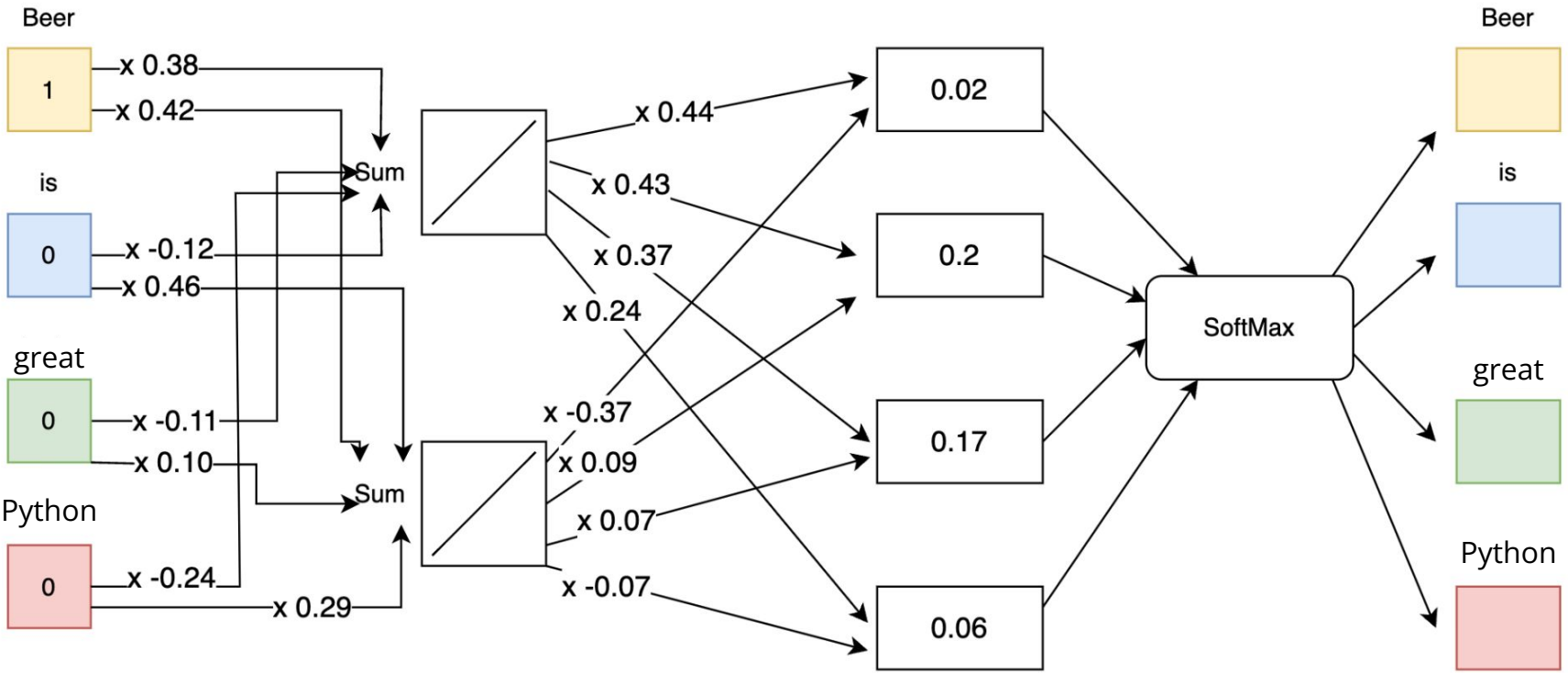
Нека направим един много прост Embedding

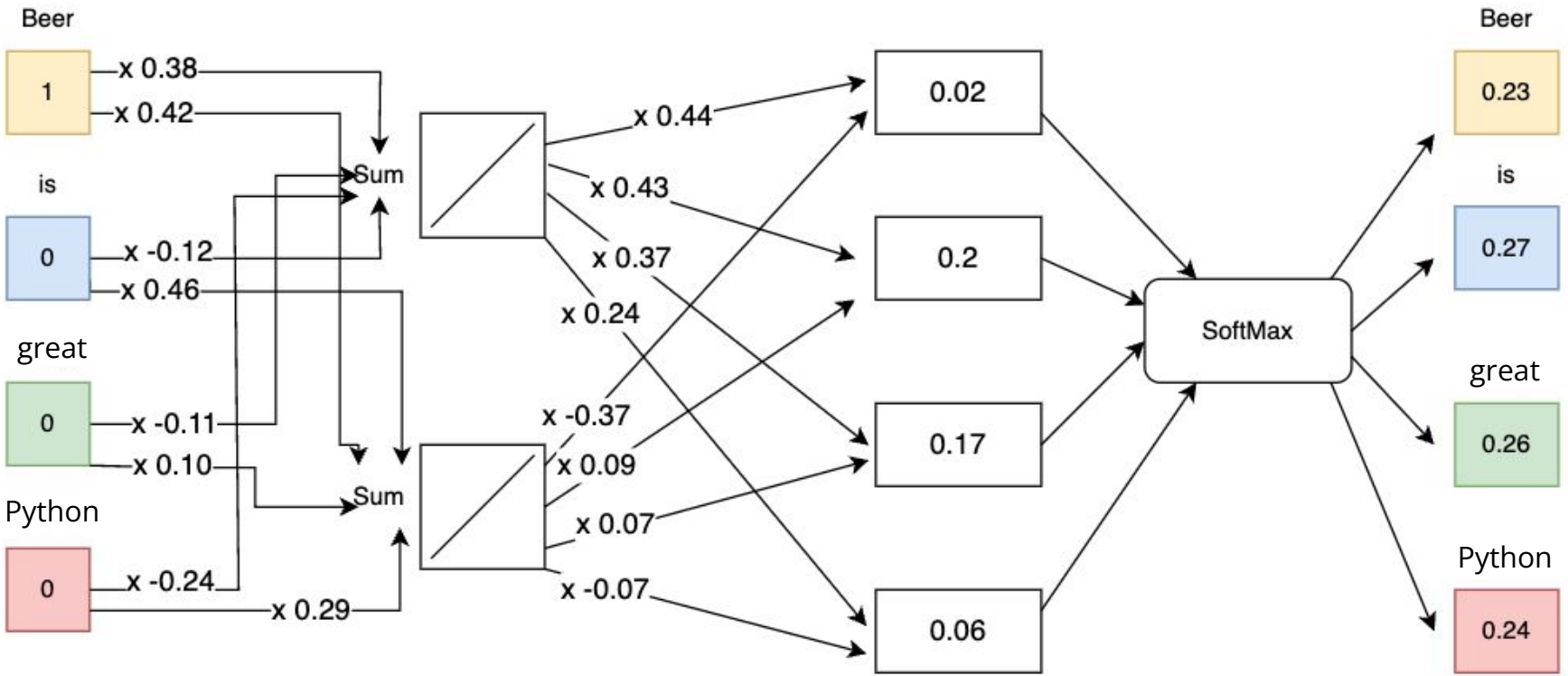
- Взимаме следните 2 изречения:
  - Beer is great
  - Python is great
- Tokenization (Много важна стъпка на обработване):
  - `[“Beer”, “is”, “great”]`
  - `[“Python”, “is”, “great”]`
- Нека сега направим един наш Embeddings Модел
- Извинявам се за следващите слайдове

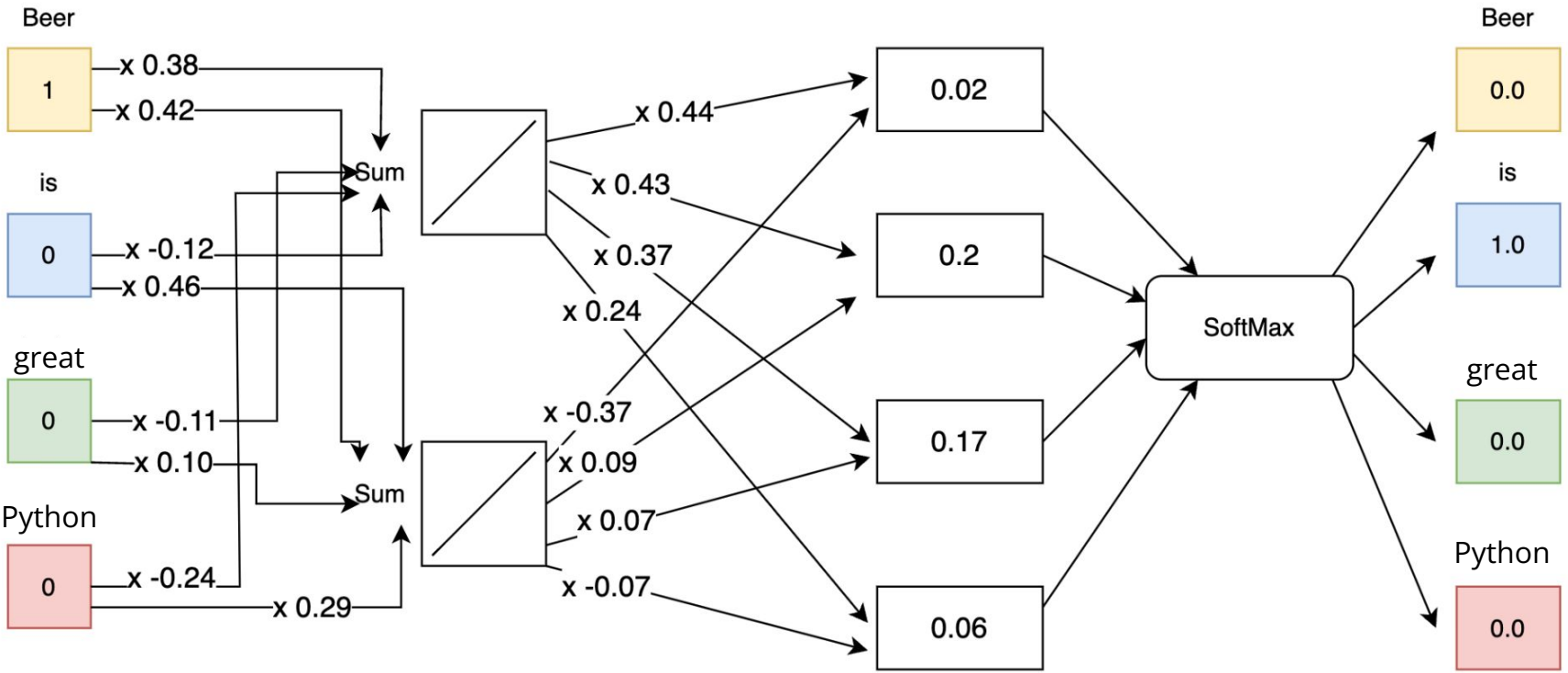

Beer
is
great
Python
Sum
Sum
SoftMax
Beer
is
great
Python


Beer
1
is
0
great
0
Python
0
Sum
Sum
SoftMax
Beer
0
is
0
great
1
Python
0


Beer
1
x 0.38
x 0.42
is
0
x -0.12
x 0.46
great
0
x -0.11
x 0.10
Python
0
x -0.24
x 0.29
Sum
Sum
x 0.44
x 0.43
x 0.37
x 0.24
x -0.37
x 0.09
x 0.07
x -0.07
SoftMax
Beer
is
great
Python


Beer
1
x 0.38
x 0.42
is
0
x -0.12
x 0.46
great
0
x -0.11
x 0.10
Python
0
x -0.24
x 0.29
Sum
Sum
x 0.44
x 0.43
x 0.37
x 0.24
x -0.37
x 0.09
x 0.07
x -0.07
0.02
0.2
0.17
0.06
SoftMax
Beer
is
great
Python


Beer
1
x 0.38
x 0.42
is
0
x -0.12
x 0.46
great
0
x -0.11
x 0.10
Python
0
x -0.24
x 0.29
Sum
Sum
x 0.44
x 0.43
x 0.37
x 0.24
x -0.37
x 0.09
x 0.07
x -0.07
0.02
0.2
0.17
0.06
SoftMax
Beer
0.23
is
0.27
great
0.26
Python
0.24


Beer
1
x 0.38
x 0.42
is
0
x -0.12
x 0.46
great
0
x -0.11
x 0.10
Python
0
x -0.24
x 0.29
Sum
Sum
x 0.44
x 0.43
x 0.37
x 0.24
x -0.37
x 0.09
x 0.07
x -0.07
0.02
0.2
0.17
0.06
SoftMax
Beer
0.0
is
1.0
great
0.0
Python
0.0

Накратко


Harvard Business Review
GETTING CONTROL OF BIG DATA
Harvard Business Review
DATA SCIENTIST
The Sexiest Job of the 21st Century
WWW.FACEBOOK.COM/EMCACADEMICALLIANCE
EMC²

Още ресурси

- Ноутбуци за NumPy & Pandas - Може да си направите копие и да екзекютвате в DeepNote
- https://www.kaggle.com/
- https://huggingface.co/
- Neural Network Playground
- https://bbycroft.net/llm
- https://tokens-lpj6s2duga-ew.a.run.app/
- https://drive.google.com/drive/folders/15Rt6yKom94WjaWIM0K7hBvK6hCLzBTyM?usp=sharing - Link to notebooks and data

Въпроси?