futurice

Deep Learning and Spark

WeAreDevelopers AI Congress in Vienna

Teemu Kinnunen / Data Scientist

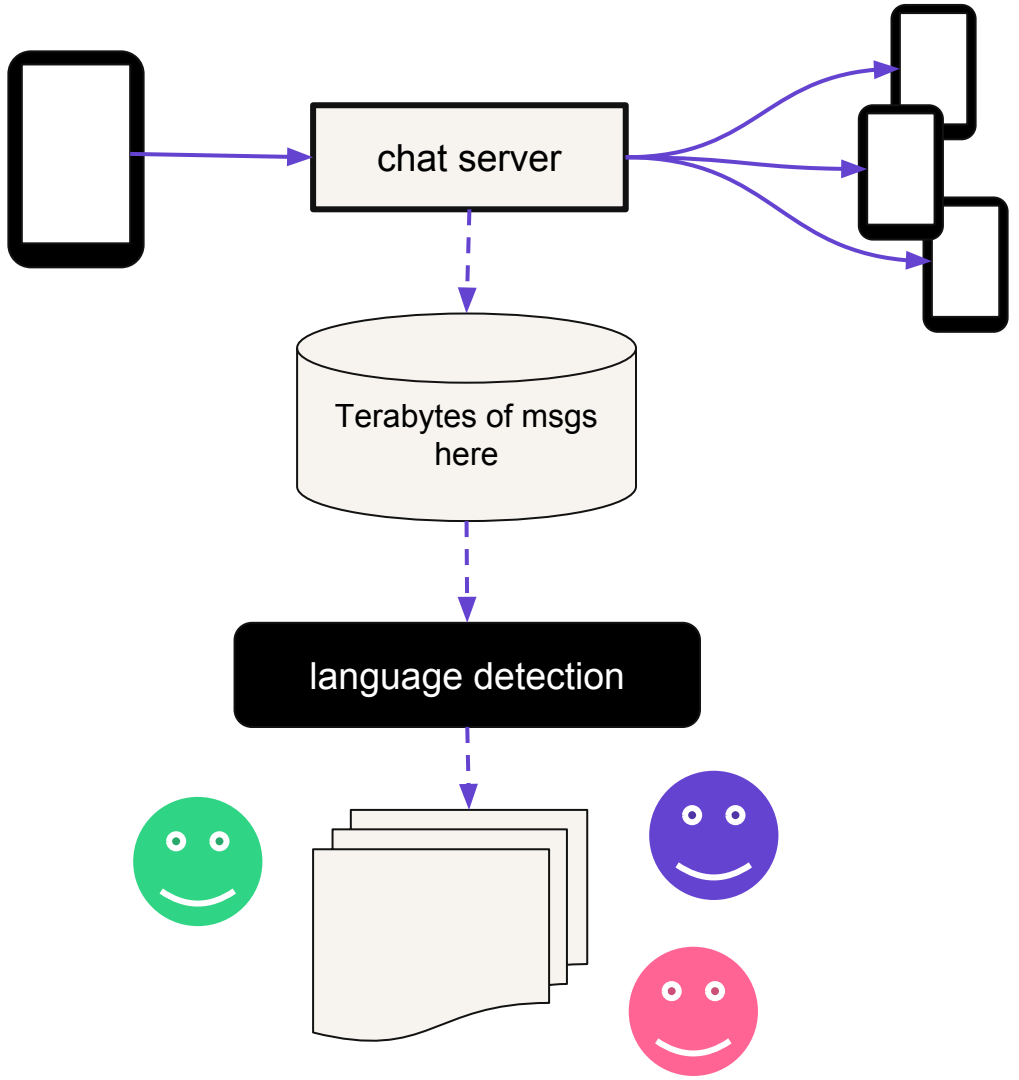
Problem description

- Giving some context about the problem -

Language classification

We have a chat application. We store messages to analyse them later.

- Need an efficient text classifier
- Need to distribute computation to multiple computers



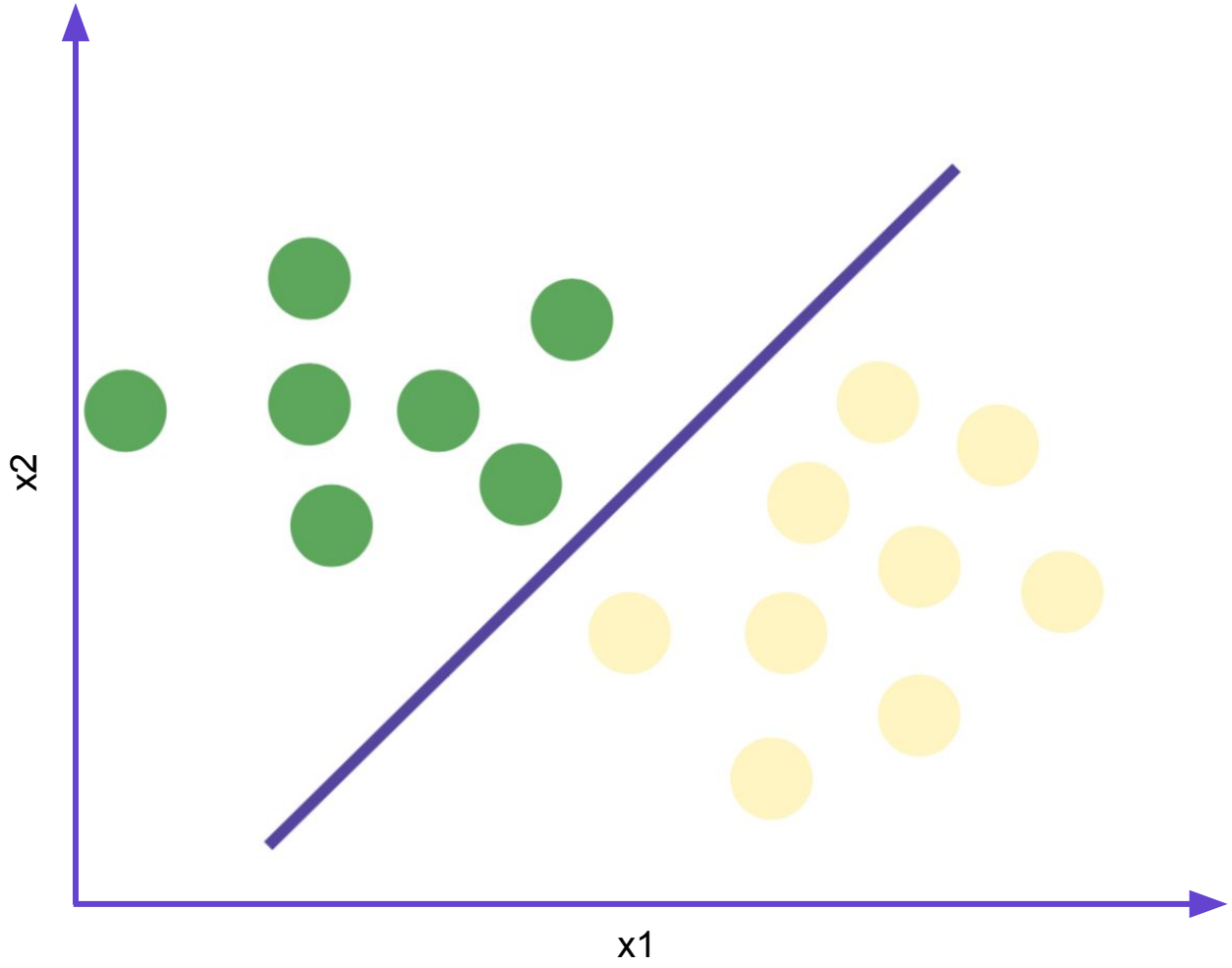
futurice

Machine Learning

- introducing main concepts -

Supervised learning

- Learns a relationship between **features** and **labels/targets**
- Regression (predict numerical value)
- **Classification (predicting a category)**

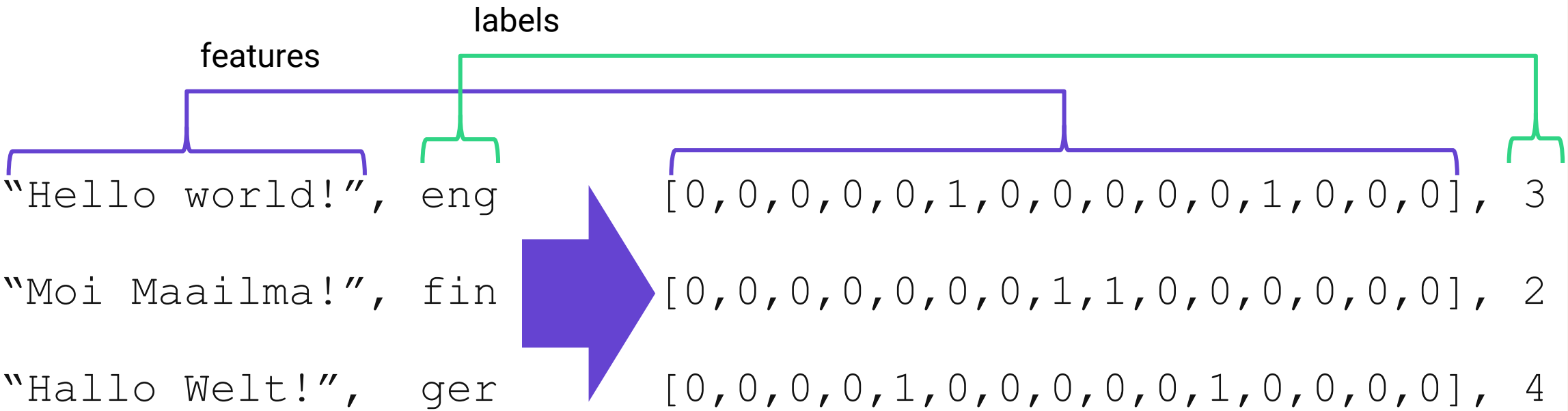


Text classification

- diving into the basics -

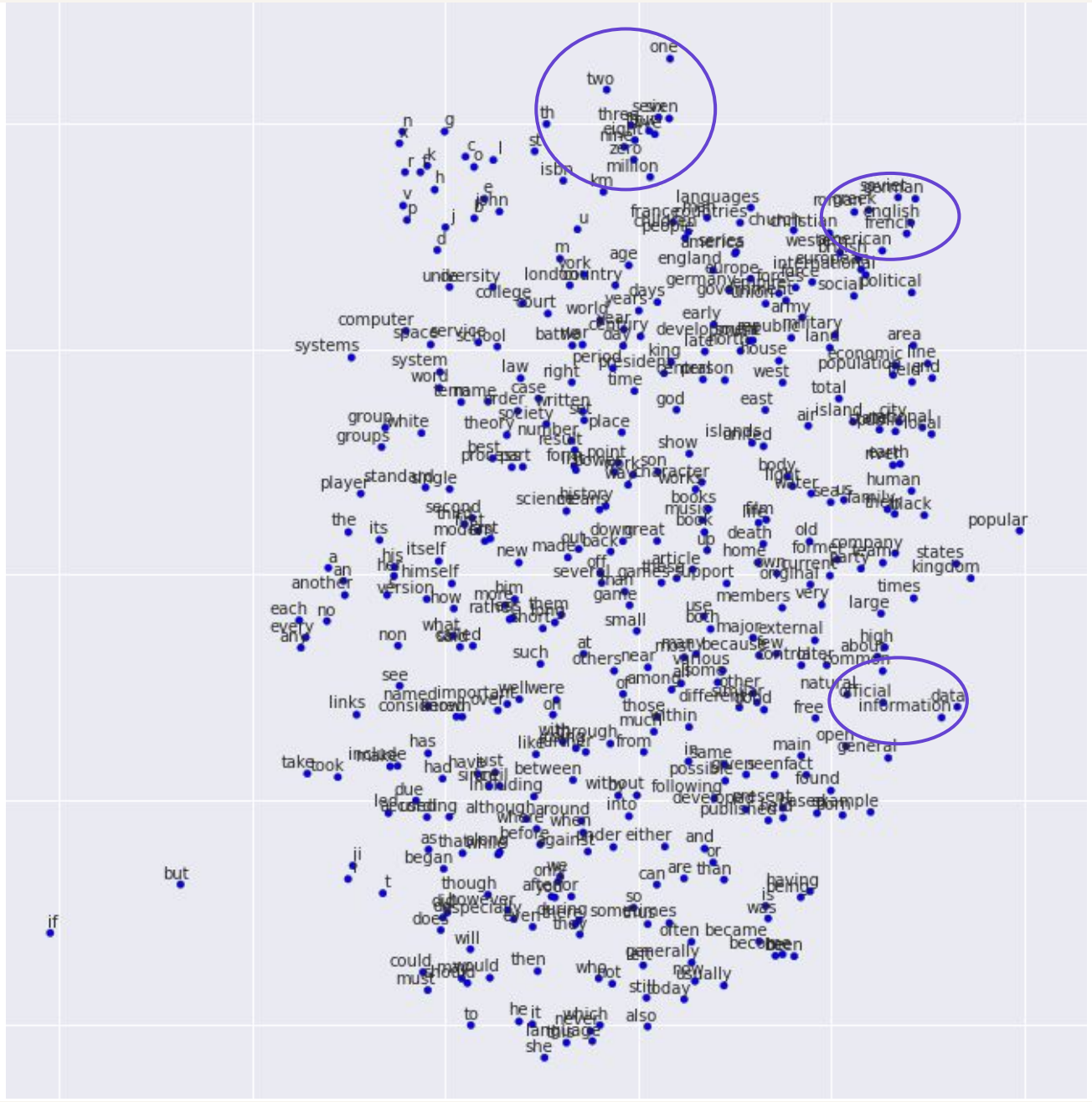
The basics of text classification

A traditional approach is to count **word histograms** and feed them to a classifier (aka Bag-of-words approach)



Words embeddings

The goal is to find a (lower dimensional) space where semantically similar words are close to each other



https://www.tensorflow.org/tutorials/representation/word2vec

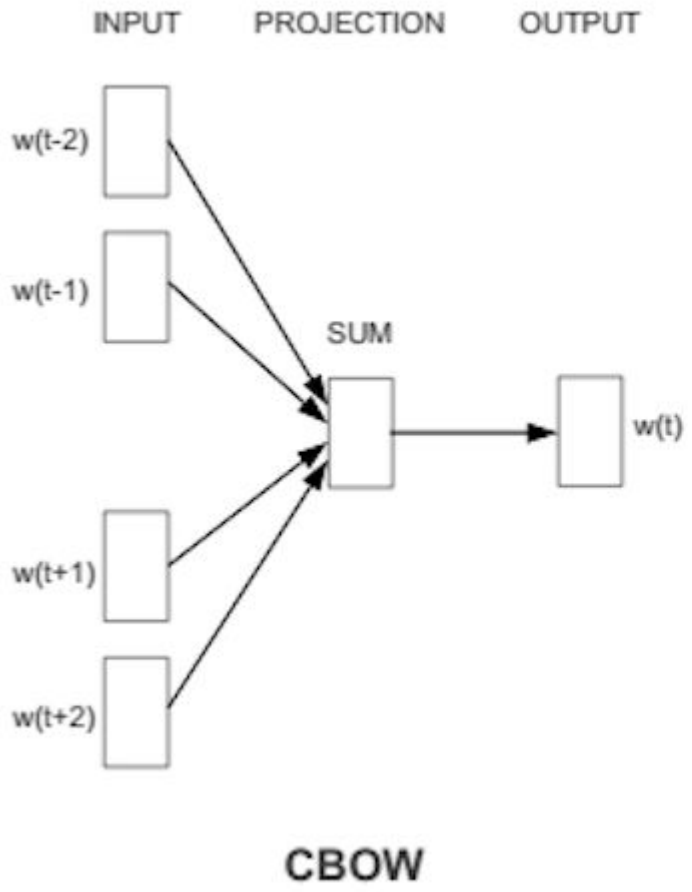
futurice

Word embeddings

Unsupervised learning method to create word embeddings

It scans through the text and tries to predict the middle word (CBOW)

Hidden layer is being used as an embedding and decision layer is taken away since not needed.



Mikolov et al. 2013

fastText

- intro and practice -

Why fastText?

Efficient, **easy-to-use** and **open-source** tool to classify text.

Tokenizer:

- N-grams for text and characters

Word embeddings:

- Skip-gram and CBOW

Text classification

- Softmax, Hierarchical softmax

fastText in practice

Command line:

```
./fasttext supervised -input train.txt -output model.bin
```

In Python (using fasttext.py)

```
import fasttext

fasttext.supervised('train.txt','model.bin')
```

Training file (train.txt):

```
__label__eng Hello world!

__label_ger Hallo Welt!

__label__fin Moi Maailma!
```

Apache Spark

- hadoop / map-reduce (mostly) in memory -

Why Apache Spark?

More data than we can fit into memory or on the hard-drive

→ We need distributed computing

Distributed computing is complex

→ We need an efficient and easy to use framework / library to distribute computation

Spark is both **efficient** and **easy-to-use**

Python + Spark + Jupyter Notebooks = Ultimate toolbox for a data scientist

Just run:

```
PYSPARK_PYTHON=python3 \

PYSPARK_DRIVER_PYTHON=jupyter \

PYSPARK_DRIVER_PYTHON_OPTS='notebook‘ \

pyspark # Need internets?
```

Loading data to a dataframe using Spark

```
# Define schema
schema = StructType([
    StructField("sentence_id", IntegerType(), True),
    StructField("language_code", StringType(), True),
    StructField("text", StringType(), True)])
```

```
# Read CSV into spark dataframe
spark_df = spark.read.csv('data/sentences.csv', schema=schema, sep='\t')
```

Displaying the data

```
# Show 10 samples
spark_df.limit(10).toPandas()
```

	sentence_id	language_code	text
0	1	cmn	我們試試看!
1	2	cmn	我该去睡觉了。
2	3	cmn	你在干什麼啊?
3	4	cmn	這是什麼啊?
4	5	cmn	今天是６月１８号，也是Muiriel的生日!
5	6	cmn	生日快乐，Muiriel!

only showing top 10 rows

fastText in Spark

- combining fastText and Spark -

Training a fastText classifier

```
train_test_df = spark_df.randomSplit([0.8,0.2], 42)
```

```
TRAIN_FILE = 'data/fasttext_train.txt'
TEST_FILE = 'data/fasttext_test.txt'
MODEL_FILE = 'data/fasttext_language'
```

```
with open(TRAIN_FILE, 'w') as fp:
    for i, row in enumerate(train_test_df[0].toLocalIterator()):
        fp.write('__label__%s %s\n' % (row['language_code'],
                                       row['text']))
```

```
model = fasttext.supervised(TRAIN_FILE, MODEL_FILE)
```

Running fastText classifier in Spark using UDF

One can use a **User Defined Function** (UDF) to run custom function/code in Spark

UDF takes a column (or columns) as an input and returns a column (can consist of many sub-columns)

Result will be stored to a current dataframe with a new column

User Defined Functions in Spark

```
from pyspark.sql.functions import col, udf

udf_len = udf(len)

messages = messages.withColumn('len', udf_len(col('text')))
```

```
messages.limit(5).toPandas()
```

	sentence_id	language_code	text	len
0	1	cmn	我們試試看!	6
1	2	cmn	我该去睡觉了。	7
2	3	cmn	你在干什麼啊?	7
3	4	cmn	這是什麼啊?	6
4	5	cmn	今天是６月１８号，也是Muiriel的生日!	22

fastText classifier UDF

```
1  # We need fasttext to load the model and make predictions
2  import fasttext
3
4  # Load model (loads when this library is being imported)
5  model = fasttext.load_model('data/model_fasttext.bin')
6
7  # This is the function we use in UDF to predict the language of a given msg
8  def predict_language(msg):
9      pred = model.predict([msg])[0][0]
10     pred = pred.replace('__label__', '')
11     return pred
```

```
import fasttext_lang_classifier
udf_predict_language = udf(fasttext_lang_classifier.predict_language)
```

```
messages = messages.withColumn('predicted_lang',
                               udf_predict_language(col('text')))
```

FASTTEXT + SPARK

fastText prediction results with Spark

```
%%time
messages.show()
```

```
+-----------+-------------+--------------------+--------------+
|sentence_id|language_code|                text|predicted_lang|
+-----------+-------------+--------------------+--------------+
|          1|          cmn|              我們試試看！|           cmn|
|          2|          cmn|             我该去睡觉了。|           cmn|
|          3|          cmn|             你在干什麼啊？|           cmn|
|          4|          cmn|              這是什麼啊？|           cmn|
|          5|          cmn|今天是６月１８号，也是Muirie...|           cmn|
|          6|          cmn|       生日快乐，Muiriel！|           cmn|
|          7|          cmn|     Muiriel现在20岁了。|           cmn|
|          8|          cmn|      密码是"Muiriel"。|           cmn|
|          9|          cmn|           我很快就會回來。|           cmn|
|         10|          cmn|              我不知道。|           cmn|
|         11|          cmn|        我不知道應該說什麼才好。|           cmn|
|         12|          cmn|           這個永遠完不了了。|           cmn|
|         13|          cmn|    我只是不知道應該說什麼而已……|           cmn|
|         14|          cmn|          那是一隻有惡意的兔子。|           cmn|
|         15|          cmn|             我以前在山里。|           cmn|
|         16|          cmn|             那是一张近照吗？|           cmn|
|         17|          cmn|          我不知道我有沒有時間。|           cmn|
|         18|          cmn|剛才我的麥克風沒起作用，不知道為什麼。|           cmn|
|         19|          cmn|    到了最後，大家一定要靠自己學習。|           cmn|
|         20|          cmn|          世界上的教育都讓我失望。|           cmn|
+-----------+-------------+--------------------+--------------+
only showing top 20 rows

CPU times: user 2.93 ms, sys: 3.76 ms, total: 6.69 ms
Wall time: 11.3 s
```

Accuracy: ~97%

Alternative approaches

spark.mllib provides word2vec and classifiers such as SVM

Elephas project combines Spark and Keras (http://maxpumperla.github.io/elephas/), but is outdated (supports Spark 1.5.x)

Thank you!
Kiitos!
Danke!
Tack!

futurice

Teemu Kinnunen

DATA SCIENTIST

@TKinnunen

teemu.kinnunen@futurice.com

+49 151 50 900 366

+358 40 961 3310

References

1. Efficient Estimation of Word Representations in Vector Space, Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, arXiv: 1301.3781 [cs.CL] (2013)

2. Bag of Tricks for Efficient Text Classification, Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov, arXiv: 1607.01759 [cs.CL] (2016)

THE LAST SLIDE

More information

Jupyter notebooks:

https://github.com/futurice/language_detection_in_spark