Extract Transform Select

INTRODUCTION TO SPARK SQL IN PYTHON

Mark Plutowski
Data Scientist

ETS

Extract
Transform
Select

Extract, Transform, and Select

- Extraction

- Transformation

- Selection

Built-in functions

```python
from pyspark.sql.functions import split, explode
```

The length function

```python
from pyspark.sql.functions import length
```

```python
df.where(length('sentence') == 0)
```

Creating a custom function

- User Defined Function

- UDF

Importing the udf function

```python
from pyspark.sql.functions import udf
```

Creating a boolean UDF

```
print(df)
```

```
DataFrame[textdata: string]
```

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

```
from pyspark.sql.types import BooleanType
```

Creating a boolean UDF

```python
short_udf = udf(lambda x:
                        True if not x or len(x) < 10 else False,
                        BooleanType())
```

```python
df.select(short_udf('textdata')\
    .alias("is short"))\
    .show(3)
```

```
+--------+
|is short|
+--------+
|   false|
|    true|
|   false|
+--------+
```

Important UDF return types

```python
from pyspark.sql.types import StringType, IntegerType, FloatType, ArrayType
```

Creating an array UDF

```
df3.select('word array', in_udf('word array').alias('without endword'))\
    .show(5, truncate=30)
```

```
+----------------------------+------------------------+
|                  word array|        without endword|
+----------------------------+------------------------+
|[then, how, many, are, there]|[then, how, many, are]|
|                  [how, many]|                  [how]|
|             [i, donot, know]|             [i, donot]|
|                  [quite, so]|                [quite]|
|    [you, have, not, observed]|    [you, have, not]|
+----------------------------+------------------------+
```

Creating an array UDF

```python
from pyspark.sql.types import StringType, ArrayType
```

```python
# Removes last item in array
in_udf = udf(lambda x:
    x[0:len(x)-1] if x and len(x) > 1
    else [],
    ArrayType(StringType())))
```

Sparse vector format

1. Indices

2. Values

Example:

- Array: `[1.0, 0.0, 0.0, 3.0]`

- Sparse vector: `(4, [0, 3], [1.0, 3.0])`

Working with vector data

- `hasattr(x, "toArray")`

- `x.numNonzeros())`

Let's practice!

DataCamp

Creating feature data for classification

INTRODUCTION TO SPARK SQL IN PYTHON

Mark Plutowski
Data Scientist

DataCamp

Transforming a dense array

```python
from pyspark.sql.functions import udf
from pyspark.sql.types import IntegerType
bad_udf = udf(lambda x:
                x.indices[0]
                if (x and hasattr(x, "toArray") and x.numNonzeros())
                else 0,
                IntegerType())
```

Transforming a dense array

```python
try:
    df.select(bad_udf('outvec').alias('label')).first()
except Exception as e:
    print(e.__class__)
    print(e.errmsg)
```

```
<class 'py4j.protocol.Py4JJavaError'>
An error occurred while calling o90.collectToPython.
```

UDF return type must be properly cast

```python
first_udf = udf(lambda x:
                int(x.indices[0])
                if (x and hasattr(x, "toArray") and x.numNonzeros())
                else 0,
                IntegerType())
```

The UDF in action

```
+-------+------------------+-----+--------------------+------------------+
|endword|               doc|count|            features|            outvec|
+-------+------------------+-----+--------------------+------------------+
|     it|[please, do, not,...| 1149|(12847,[15,47,502...|  (12847,[7],[1.0])|
| holmes|[start, of, the, ...|  107|(12847,[0,3,183,1...|(12847,[145],[1.0])|
|      i|[the, adventures,...|  103|(12847,[0,3,35,14...| (12847,[11],[1.0])|
+-------+------------------+-----+--------------------+------------------+
```

```python
df.withColumn('label', k_udf('outvec')).drop('outvec').show(3)
```

```
+-------+------------------+-----+--------------------+-----+
|endword|               doc|count|            features|label|
+-------+------------------+-----+--------------------+-----+
|     it|[please, do, not,...| 1149|(12847,[15,47,502...|    7|
| holmes|[start, of, the, ...|  107|(12847,[0,3,183,1...|  145|
|      i|[the, adventures,...|  103|(12847,[0,3,35,14...|   11|
+-------+------------------+-----+--------------------+-----+
```

CountVectorizer

- ETS : Extract Transform Select

- CountVectorizer is a Feature **Extractor**

- Its input is an array of strings

- Its output is a vector

Fitting the CountVectorizer

```python
from pyspark.ml.feature import CountVectorizer

cv = CountVectorizer(inputCol='words',
                        outputCol="features")
model = cv.fit(df)
result = model.transform(df)
print(result)
```

```
DataFrame[words: array<string>, features: vector]

# Dense string array on left, dense integer vector on right
+-----------------------+----------------------------------------+
|words                  |features                                |
+-----------------------+----------------------------------------+
|[Hello, world]         |(10,[7,9],[1.0,1.0])                     |
|[How, are, you?]       |(10,[1,3,4],[1.0,1.0,1.0])               |
|[I, am, fine, thank, you]|(10,[0,2,5,6,8],[1.0,1.0,1.0,1.0,1.0])|
+-----------------------+----------------------------------------+
```

Let's practice!

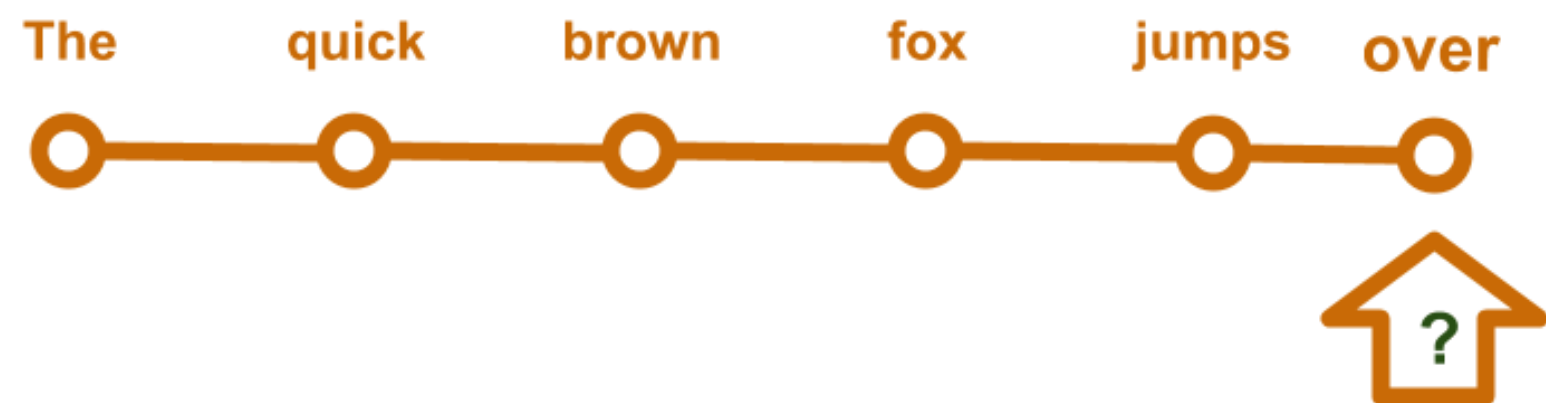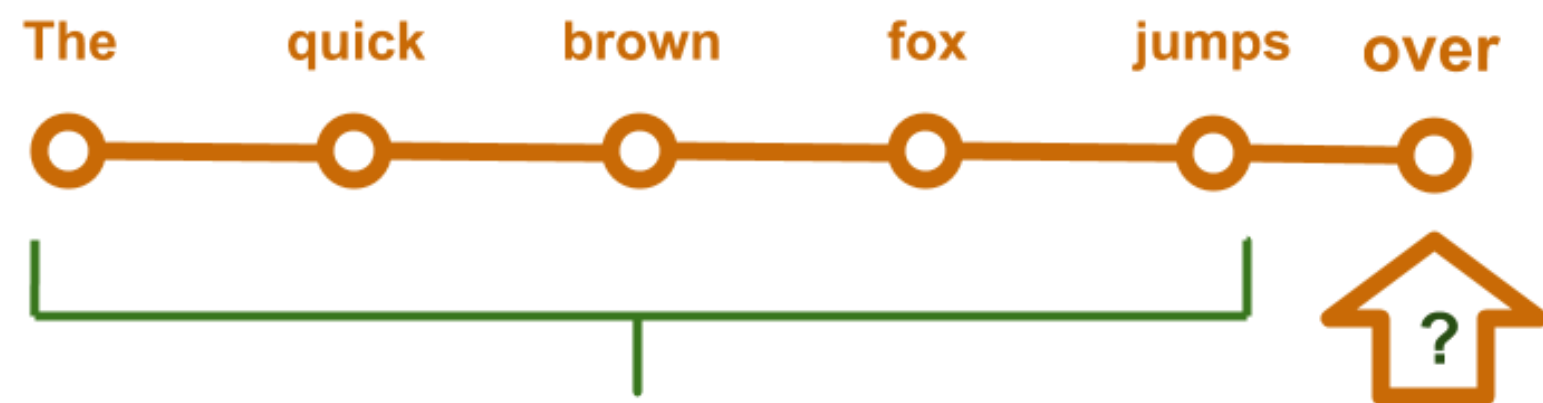INTRODUCTION TO SPARK SQL IN PYTHON

Text Classification

INTRODUCTION TO SPARK SQL IN PYTHON
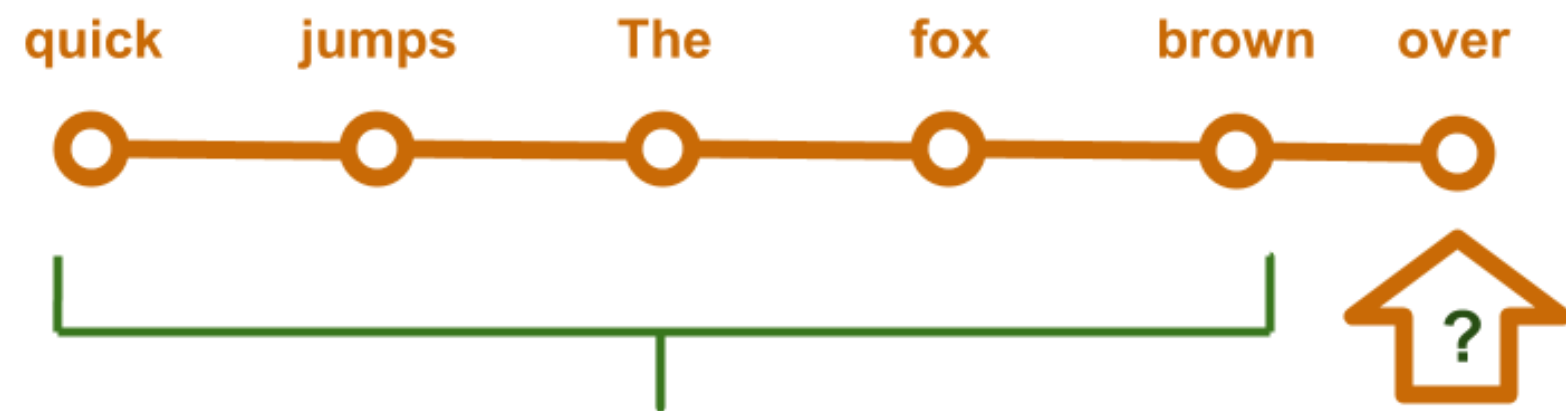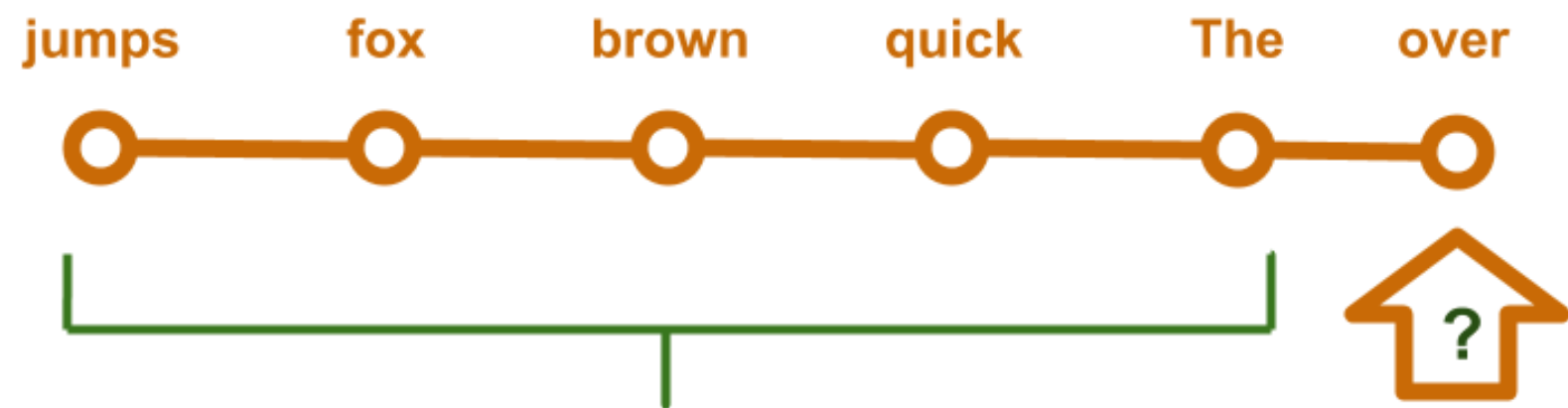
Mark Plutowski
Data Scientist

Endword Prediction

The   quick   brown   fox   jumps   over

?

quick   jumps   The   fox   brown   over

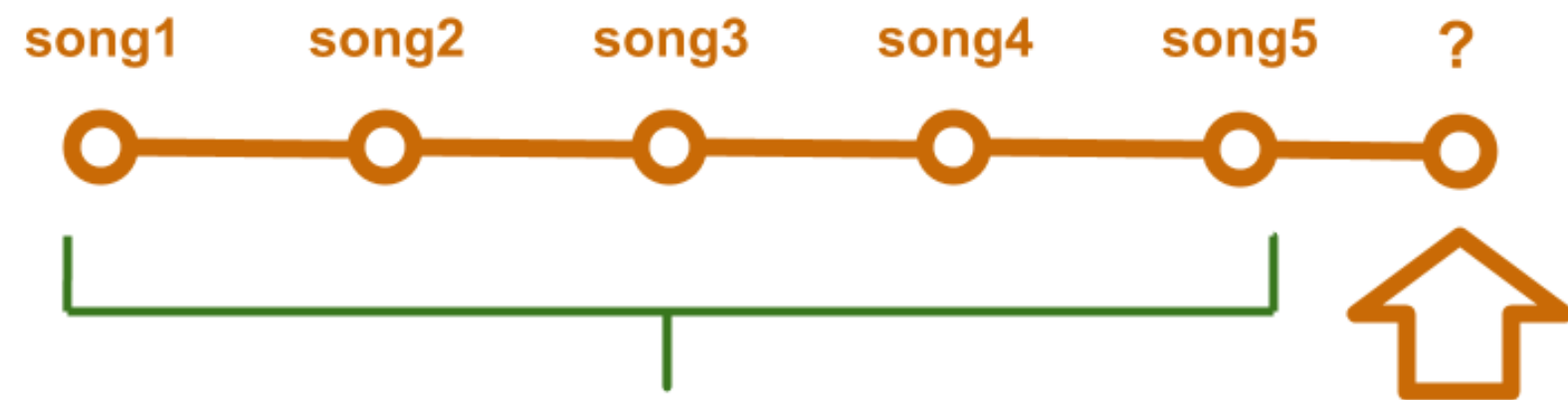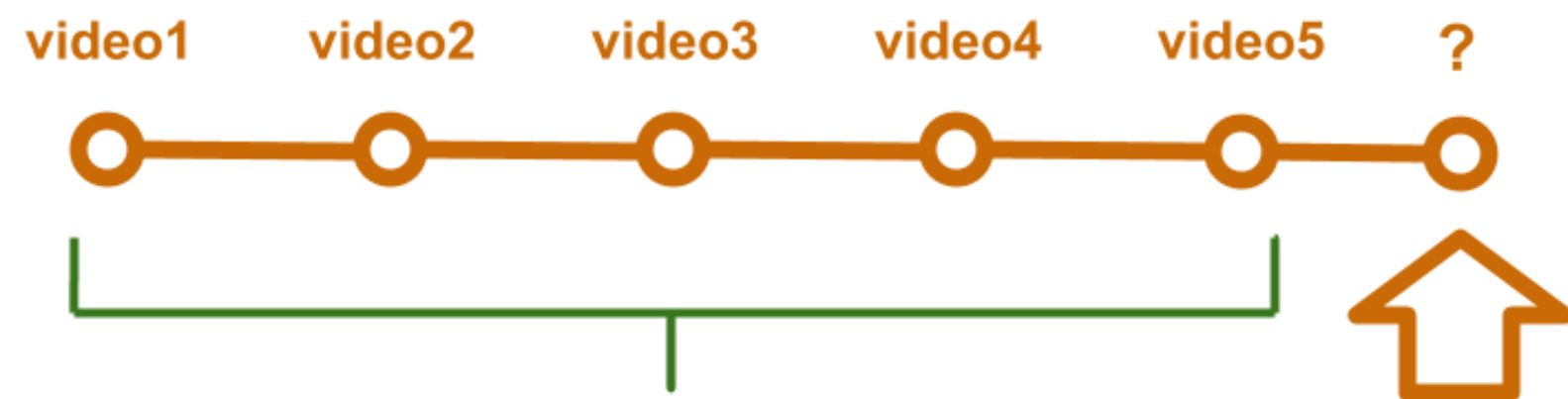jumps   fox   brown   quick   The   over

?
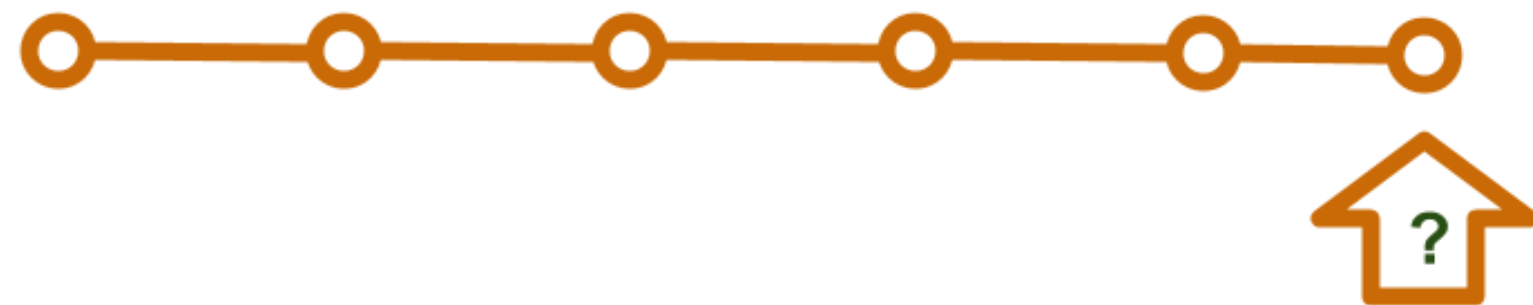
Selecting the data

```
df_true = df.where("endword in ('she', 'he', 'hers', 'his', 'her', 'him')")\
            .withColumn('label', lit(1))
```

```
df_false = df.where("endword not in ('she', 'he', 'hers', 'his', 'her', 'him')")\
            .withColumn('label', lit(0))
```

Combining the positive and negative data

```
df_examples = df_true.union(df_false)
```

Splitting the data into training and evaluation sets

```python
df_train, df_eval = df_examples.randomSplit((0.60, 0.40), 42)
```

Training

```python
from pyspark.ml.classification import LogisticRegression

logistic = LogisticRegression(maxIter=50, regParam=0.6, elasticNetParam=0.3)

model = logistic.fit(df_train)

print("Training iterations: ", model.summary.totalIterations)
```

Let's practice!

INTRODUCTION TO SPARK SQL IN PYTHON

Predicting and evaluating

INTRODUCTION TO SPARK SQL IN PYTHON

Mark Plutowski
Data Scientist

DataCamp

Applying a model to evaluation data

```
predicted = df_trained.transform(df_test)
```

- prediction column: double

- probability column: vector of length two

```
x = predicted.first
print("Right!" if x.label == int(x.prediction) else "Wrong")
```

Evaluating classification accuracy

```
model_stats = model.evaluate(df_eval)
```

```
type(model_stats)
```

```
pyspark.ml.classification.BinaryLogisticRegressionSummary)
```

```
print("\nAccuracy: %.2f" % model_stats.areaUnderROC)
```

Example of classifying text

- Positive labels:
  - ['**her**', '**him**', '**he**', '**she**', '**them**', '**us**', '**they**', '**himself**', '**herself**', '**we**']

- Number of examples: **5746**

- Number of examples: **2873 positive**, **2873 negative**

- Number of training examples: **4607**

- Number of test examples: **1139**

- training iterations: **21**

- Test AUC: **0.87**

Predicting the endword

- Positive label: **'it'**

- Number of examples: **438**

- Number of examples: **219 positive**, **219 negative**

- Number of training examples: **340**

- Number of test examples: **98**

- Test AUC: **0.85**

Let's practice!

INTRODUCTION TO SPARK SQL IN PYTHON

Recap

INTRODUCTION TO SPARK SQL IN PYTHON

Mark Plutowski
Data Scientist

Recap

- Window function SQL

- **E**xtract

- **T**ransform

- **S**elect

- Train

- Predict

- Evaluate

Congratulations!

INTRODUCTION TO SPARK SQL IN PYTHON