Data2_Q4

November 29, 2021

1 Question 4

1.1 (i)

Map reduce is a way to process data in paraell on a large, distributed dataset

For example, there is a large text file and you must count the occurences of certain words in the text. You can do this by counting the occurences on each line then combining these totals together for the final count

1. Split data of the text file into chunks, in this example I am using each line as an individual chunck
2. Map each chunk to the number of occurences of each specified words in the line:
    1. First the chunk is filtered, such that only relavent words are left
    2. Then the occurences of the relevant words are counted. This is stored as a Counter object
3. Reduce the multiple Counter objects into a total count by combining pairs of Counter objects together until only 1 object is left

[ ]:
```
from functools import reduce
from collections import Counter

counted_words = ["dog", "cat", "horse", "puppy"]

text = """fsdkljfsdl dog sajf dfjksd horse dog
fsd fsd cat fjsdks dog fsd dog
dog dog horse dsfsd dfs
fsdjfklf puppy cat fds fdsf
puppy"""


def filter_counted_words(word):
    if word in counted_words:
        return True
    return False


def mapper(line):
    words = line.split(" ")
```

```
    filtered_words = filter(filter_counted_words, words) # Unessecary words are␣
 ↪filtered from the line
    return Counter(filtered_words)


def reducer(cnt1, cnt2):
    cnt1.update(cnt2)
    return cnt1

chunks = text.split("\n") # Each chunk is just a line of the file
mapped = map(mapper, chunks) # A Counter object is mapped to each chunk
reduced = reduce(reducer, mapped) # Counter objects are reduced to a single␣
 ↪object

print(reduced)
```

```
Counter({'dog': 6, 'horse': 2, 'cat': 2, 'puppy': 2})
```

1.2 (ii)

1.2.1 (a)

mapper.py:

[ ]:
```
import sys
import pandas as pd

csv = [line.split(",") for line in sys.stdin.read().splitlines()]
df = pd.DataFrame(
    csv, columns=["salesperson", "capp_size", "tea_size", "capp_price",␣
 ↪"tea_price"]
)
cap_df = df[["capp_size", "capp_price"]].set_index("capp_size")
tea_df = df[["tea_size", "tea_price"]].set_index("tea_size")
df = pd.concat([cap_df, tea_df]).fillna(0)
df[""] = df.capp_price.astype(float) + df.tea_price.astype(float)
df.drop(columns=["capp_price", "tea_price"], inplace=True)
print(df)
```

```
$ cat ../exam-resources/hot_drinks.csv | python3 mapper.py | sort
```

```
capp_L   3.50
capp_L   3.50
capp_L   3.50
capp_L   3.50
capp_M   3.00
capp_M   3.00
capp_M   3.00
capp_S   2.50
```

```
capp_S   2.50
capp_S   2.50
capp_S   2.50
capp_XL  4.50
capp_XL  4.50
capp_XL  4.50
tea_L    2.75
tea_L    2.75
tea_L    2.75
tea_L    2.75
tea_M    2.00
tea_M    2.00
tea_M    2.00
tea_S    1.50
tea_S    1.50
tea_S    1.50
tea_S    1.50
tea_S    1.50
tea_XL   3.25
tea_XL   3.25
```

1.2.2 (b)

reducer.py

[ ]:
```
import sys
import pandas as pd

csv = [line.split(",")[0].split() for line in sys.stdin.read().splitlines()][1:]
df = pd.DataFrame(csv, columns=["type_size", "price"]).set_index("type_size")
df.price = pd.to_numeric(df.price)


df = df.groupby(level=0).sum()
df.rename(columns={"price": ""}, inplace=True)
df.index.name = None
print(df)
```

```
$ cat ../exam-resources/hot_drinks.csv | python3 mapper.py | sort |
python3 reducer.py
```

```
capp_L   14.0
capp_M    9.0
capp_S   10.0
capp_XL  13.5
tea_L    11.0
tea_M     6.0
tea_S     7.5
tea_XL    6.5
```