Sleep

April 25, 2022

1 Sleep

1.0.1 Deleja-Hotko Julian, Nou Sara, Zivojinovic Andrej

1.1 1. Goals

We will be analysing the results of a Self-experiment in which someone recorded their sleeping habits as well as the quality of sleep they experienced, and contrast it against a comprehensive dataset from a represenattive sample of the population of the United States of America.

1.2 2. Sources:

For our data, we mainly used the dataset "*Sleep Data*" from Dana Diotte on https://www.kaggle.com/datasets/danagerous/sleep-data, which is a comprehensive record by single user of their own sleeping habits.

In Addition, we used the dataset "*Time Americans spend sleeping*" from Makeover Monday on https://data.world/makeovermonday/2019w23 , which in itself is an extract from the much more comprehensive American Time Use Study https://www.bls.gov/tus/database.htm from 2017, providing an overview of average sleep times in the population of the United States distinguished between different age groups and genders.

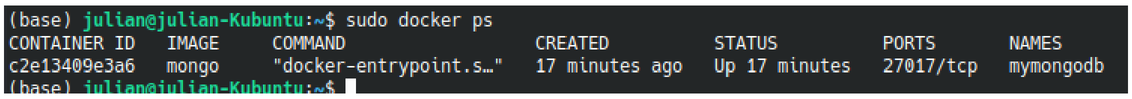
1.3 3. Infrastructure

for the purpose of working with the datasets, we created a jupyter notebook, from which we created multiple python scripts and csv output files to use with mapreduce.

For the purpose of collaboration, we stored said files, along with our data sources, any images and an export from out database in a GitHub repository.

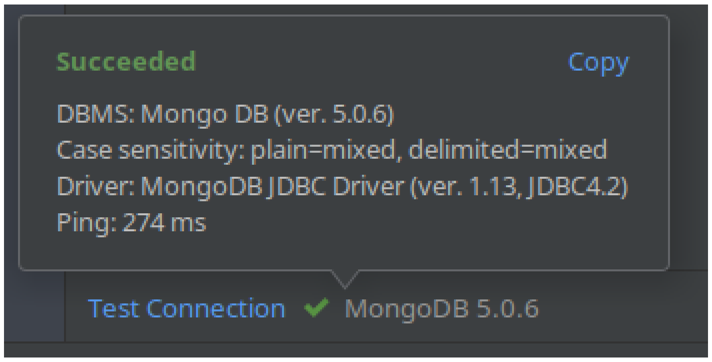
In order to store all of our data in an easily accessible way, we used a MongoDB Database. We chose this technology because it is easy to change data structure later on, for example if we want to add a second data source, because it is generally well-suited for Big Data Projects, and because some of us are already somewhat familiar with MongoDB Databases, allowing us to use the database much more efficiently.

We ran this database within a docker container:



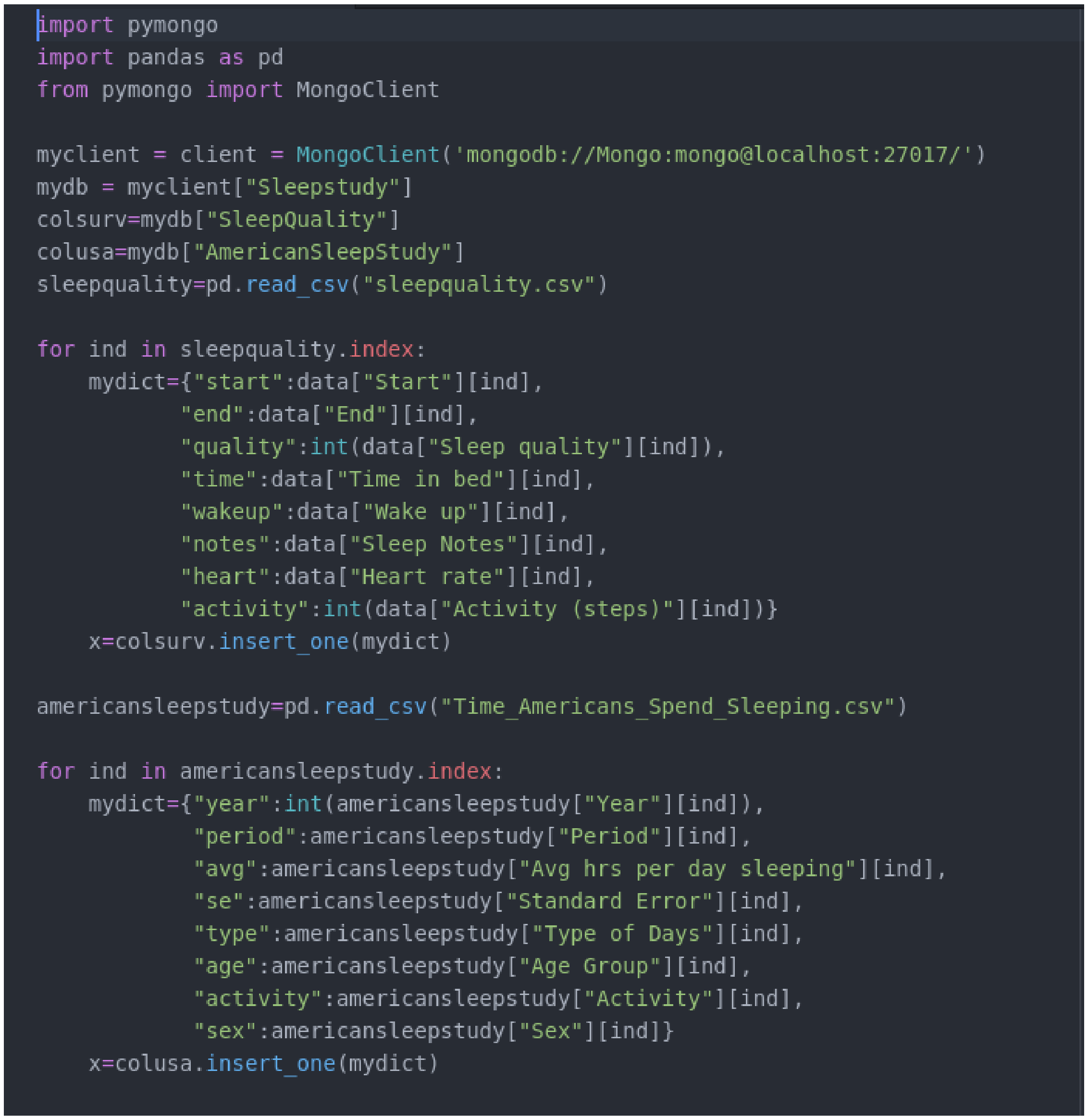
```
(base) julian@julian-Kubuntu:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS       NAMES
c2e13409e3a6   mongo     "docker-entrypoint.s…"   17 minutes ago   Up 17 minutes   27017/tcp   mymongodb
(base) julian@julian-Kubuntu:~$
```

Afterward we connected through DataGrip to insert our Data into the Database:



DBMS: Mongo DB (ver. 5.0.6)
Case sensitivity: plain=mixed, delimited=mixed
Driver: MongoDB JDBC Driver (ver. 1.13, JDBC4.2)
Ping: 274 ms

We ran a local python script to map and insert our data into the database



```python
import pymongo
import pandas as pd
from pymongo import MongoClient

myclient = client = MongoClient('mongodb://Mongo:mongo@localhost:27017/')
mydb = myclient["Sleepstudy"]
colsurv=mydb["SleepQuality"]
colusa=mydb["AmericanSleepStudy"]
sleepquality=pd.read_csv("sleepquality.csv")

for ind in sleepquality.index:
    mydict={"start":data["Start"][ind],
            "end":data["End"][ind],
            "quality":int(data["Sleep quality"][ind]),
            "time":data["Time in bed"][ind],
            "wakeup":data["Wake up"][ind],
            "notes":data["Sleep Notes"][ind],
            "heart":data["Heart rate"][ind],
            "activity":int(data["Activity (steps)"][ind])}
    x=colsurv.insert_one(mydict)

americansleepstudy=pd.read_csv("Time_Americans_Spend_Sleeping.csv")

for ind in americansleepstudy.index:
    mydict={"year":int(americansleepstudy["Year"][ind]),
             "period":americansleepstudy["Period"][ind],
             "avg":americansleepstudy["Avg hrs per day sleeping"][ind],
             "se":americansleepstudy["Standard Error"][ind],
             "type":americansleepstudy["Type of Days"][ind],
             "age":americansleepstudy["Age Group"][ind],
             "activity":americansleepstudy["Activity"][ind],
             "sex":americansleepstudy["Sex"][ind]}
    x=colusa.insert_one(mydict)
```

[1]:
```
import pymongo
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from pymongo import MongoClient
```

[2]:
```
#datenbanksetup
myclient = client = MongoClient('mongodb://Mongo:mongo@192.168.50.25:27017/')
mydb = myclient["Sleepstudy"]
colqual=mydb["SleepQuality"]
colusa = mydb["AmericanSleepStudy"]
if type(colusa.find_one()) == type({}):
    print("Connection Sucessful!")
```

```
Connection Sucessful!
```

1.4 4. Data

first, let's take a closer look at the data we are using:

1.4.1 *American Sleep Study*

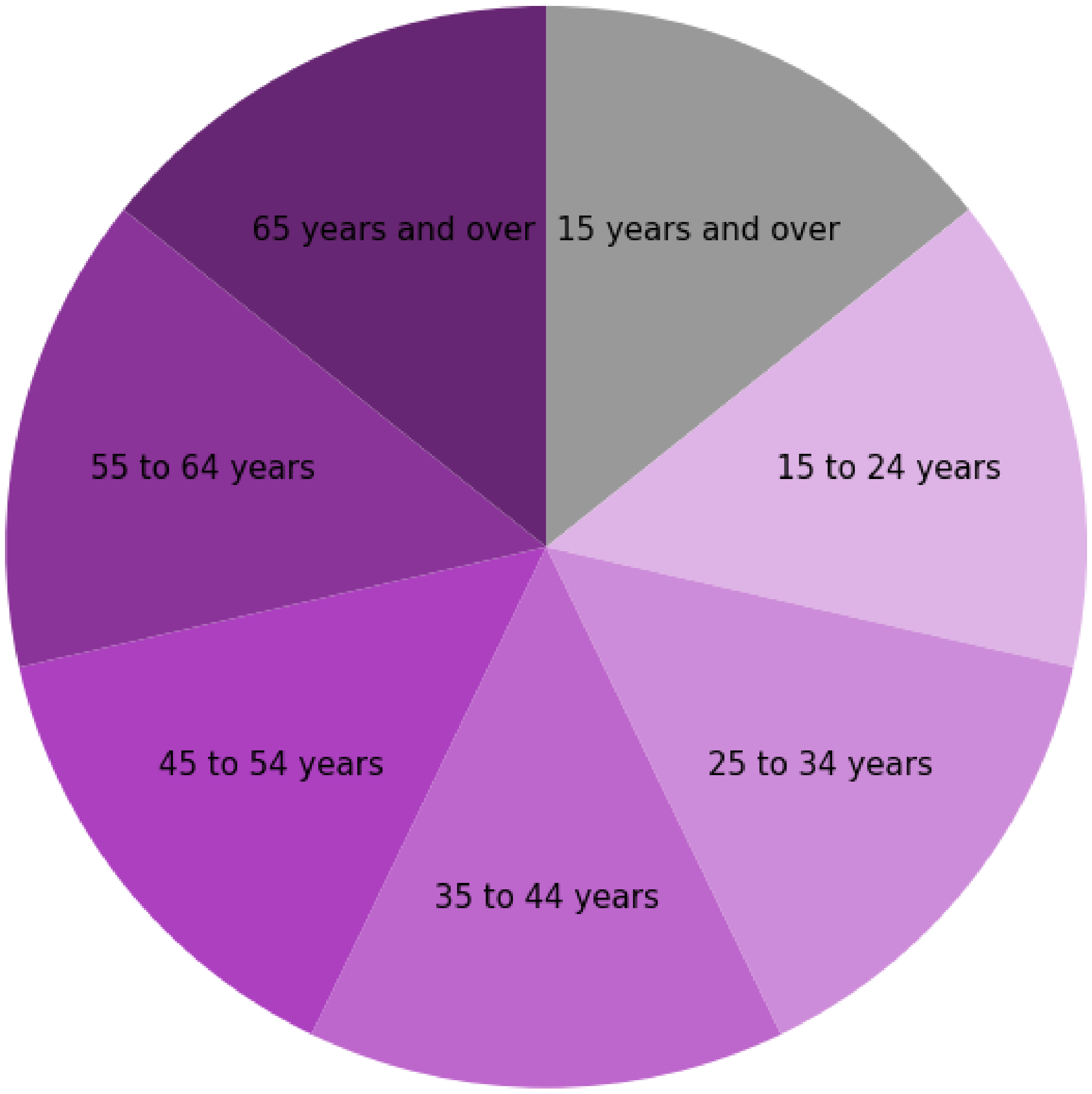
As previously mentioned, this dataset is an extract of a larger dataset, which contains yearly survey results on how Americans use their time polled by the US government.

This dataset consists of 945 entries with the fields: - **Year:** the year the survey results were polled - **Avg hrs per day sleeping:** how much time the pariticipants of the survey spent sleeping on average - **Standard Error:** how much the individual results of the survey deviate from the average - **Type of Days:** what kind of weekdays were questioned in the survey (all days, only workdays, only weekends, etc.) - **Age Group:** what age group the praticipants of the survey fall under - **Sex:** the sex of the pariticipants - **there were two additional columns:** "Period" and "Activity", which are remainders of the original dataset and are identical for all rows.

Since the results are already curated and filtered, there are no missing datapoints, and the fields are very evenly distriubuted, especially the age:

[3]:
```
sizes=[0,0,0,0,0,0,0]
labels=["15 years and over","15 to 24 years", "25 to 34 years","35 to 44␣
 ↪years","45 to 54 years","55 to 64 years","65 years and over"]
colorsages=["#999999","#deb3e6","#cd8cd9","#bd66cc","#ac40bf","#8a3399","#672673"]
for x in range(0,7):
    myquery={"age":str(labels[x])}
    mydoc = colusa.find(myquery)
    for y in mydoc:
        sizes[x]+=1

plt.figure(facecolor='white')
plt.pie(sizes,labels=labels, textprops={'fontsize': 15,"ha":"center"},radius=3,␣
↪colors=colorsages, startangle=90,counterclock=False,labeldistance=0.65)

plt.show
```

```
[3]: <function matplotlib.pyplot.show(close=None, block=None)>
```



This equal distribution means that the data has been pre-selected to contain an equal amount of surveys for each individual age group.

1.4.2 *Sleep quality*

This dataset is a daily recording by a single person over almost 4 years (from late 2014 to early 2018) consisting of various data about their own sleep quality and related information.

It contains 887 rows with the following fields: - **Start:** the timestamp when the author fell asleep - **End:** the timestamp import seaborn as snswhen the author woke up - **Sleep quality:** in percent, how the author rtates the quality of sleep that night - **Time in bed:** how much time the author spent sleeping - **Wake up:** what mood the author was in upon wakeup, indicated by a happy (:)), neutral (:|) or sad (:() smiley - **Sleep Notes:** additional notes the author had about the night, like whether they consumed caffeine or worked out - **Heart rate:** the (presumably) average heart rate during sleep - **Activity(steps):** how many steps the author walked the day before

As this is a personal recording there are quite a few missing datapoints: - the author only stareted tracking their steps after about half the time, in august of 2016. - in contrast, heart rate data is only available until early january 2016, and there arew also holes before. - similarily,l the mood upon wakeup is also only recorded until early january 2016 - there are also no notes present past june 2017, with a few notes missing before too.

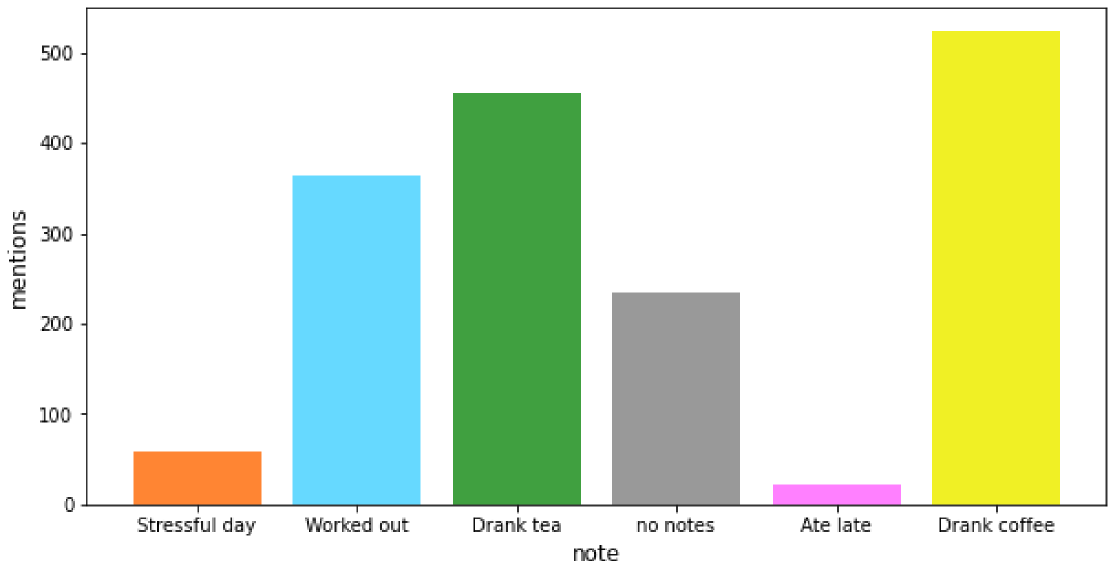
[4]:
```
%%file MRsleepnotesval.py
from mrjob.job import MRJob
class MRsleepnotesval(MRJob):
    def mapper(self, _, line):
        dt=line.split(",")
        if not dt[0]=="Start":
            if len(dt)>5 and len(dt[5])>0:
                notes=dt[5].split(":")
                for x in notes:
                    yield x,1
            else:
                yield "no notes",1



    def reducer(self, key, values):
        yield key,sum(values)

if __name__ == '__main__':
    MRsleepnotesval.run()
```

```
Writing MRsleepnotesval.py
```

[5]:
```
! python MRsleepnotesval.py sleepquality.csv > MRsleepnotesresults.csv
```

```
No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/MRsleepnotesval.julian.20220425.164454.843691
Running step 1 of 1…
job output is in /tmp/MRsleepnotesval.julian.20220425.164454.843691/output
Streaming final output from
/tmp/MRsleepnotesval.julian.20220425.164454.843691/output…
Removing temp directory /tmp/MRsleepnotesval.julian.20220425.164454.843691…
```

[6]:
```
colnamesnotes=["note","amount"]
MRsleepnotesresults=pd.read_csv("MRsleepnotesresults.
↪csv",sep="\t",names=colnamesnotes,header=None)
notes=MRsleepnotesresults["note"].tolist()
amounts=MRsleepnotesresults["amount"].tolist()
colorsnotes=["#ff8533","#66d9ff","#40a040","#999999","#ff80ff","#f0f025"]

fig = plt.figure(figsize = (10, 5))

plt.xlabel("note", fontsize="large")
plt.ylabel("mentions", fontsize="large")

plt.bar(notes,amounts,color=colorsnotes)
plt.show()
```



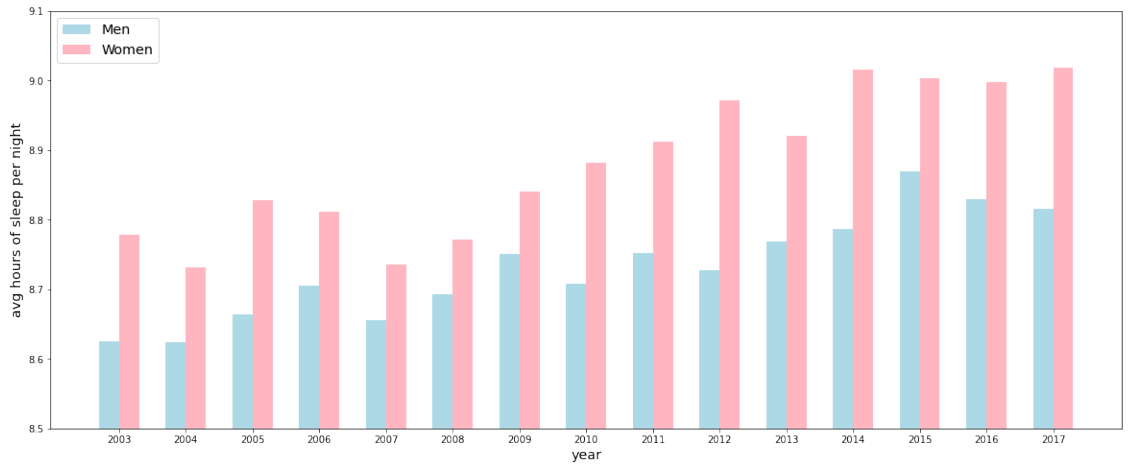
Here is how often the author made each single note in their recordings. There are multiple notes added to some lines.

1.5 5. Analysis

1.5.1 Averages

To start, let's compare the average sleeping duration of the general population to the author of our *Sleep quality* Dataset.

[7]:
```
men={}
women={}
##
for i in range(2003,2018):
    marr=[]
    mquery = { "year": i,"sex":"Men"}
    mdoc = colusa.find(mquery)
    for x in mdoc:
        marr.append(x["avg"])
    mavg=np.array(marr)
    men[str(i)]=np.mean(mavg)

myears=list(men.keys())
maverages=list(men.values())
##

##
for i in range(2003,2018):
    farr=[]
    fquery = { "year": i,"sex":"Women"}
    fdoc = colusa.find(fquery)
    for x in fdoc:
        farr.append(x["avg"])
    favg=np.array(farr)
    women[str(i)]=np.mean(favg)

fyears=list(women.keys())
faverages=list(women.values())
##

fig = plt.figure(figsize = (20, 8))
plt.ylim(8.5,9.1)

plt.xlabel("year", fontsize="x-large")
plt.ylabel("avg hours of sleep per night", fontsize="x-large")

X_axis = np.arange(15)
plt.bar(myears, maverages, align="edge", width=-0.3, color ='lightblue',␣
 ↪label="Men")
plt.bar(fyears, faverages, align="edge", width=0.3, color ='lightpink',␣
 ↪label="Women")
plt.legend(loc="upper left", fontsize="x-large")
plt.show()
```



Now, we add combine these general results with those of the (male) author of the second dataset:

To do this, we first need to extract the yearly averages from this daily dataset.

In order to achive this, we use a Mapreduce process on the *Sleep quality* dataset, which first parses the time in hh:mm format from the original dataset into a float and extracts the year from the timestamps. Then, using the years as keys, the time is averaged.

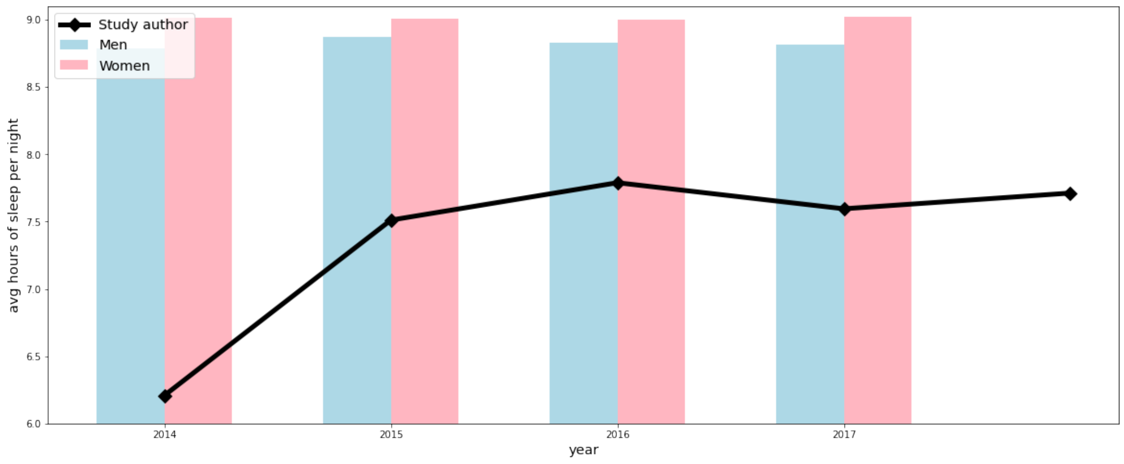
[20]:
```
%%file MRsleepavg.py
from mrjob.job import MRJob
class MRsleepavg(MRJob):
    def mapper(self, _, line):
        dt=line.split(",")
        if not dt[0]=="Start":
            year=dt[0].split("-")[0]
            if len(dt)>=4:
                time=dt[3].split(":")
                h=int(time[0])
                m=int(time[1])/60
                yield int(year), h+m


    def reducer(self, key, values):
        c=0
        s=0
        for x in values:
            s=s+x
            c=c+1
        avg=s/c
        yield key,avg

if __name__ == '__main__':
    MRsleepavg.run()
```

```
Overwriting MRsleepavg.py
```

[21]:
```
! python MRsleepavg.py sleepquality.csv > MRsleepavgresults.csv
```

```
No configs found; falling back on auto-configuration
No configs specified for inline runner
Creating temp directory /tmp/MRsleepavg.julian.20220425.171907.718252
Running step 1 of 1…
job output is in /tmp/MRsleepavg.julian.20220425.171907.718252/output
Streaming final output from
/tmp/MRsleepavg.julian.20220425.171907.718252/output…
Removing temp directory /tmp/MRsleepavg.julian.20220425.171907.718252…
```

[22]:
```
colnamesavg=["year","result"]
MRsleepavgresults=pd.read_csv("MRsleepavgresults.
 ↪csv",sep="\t",names=colnamesavg,header=None).sort_values("year")
years=MRsleepavgresults["year"].tolist()
results=MRsleepavgresults["result"].tolist()


fig = plt.figure(figsize = (20, 8))
plt.ylim(6,9.1)

plt.xlabel("year", fontsize="x-large")
plt.ylabel("avg hours of sleep per night", fontsize="x-large")

X_axis = np.arange(5)


plt.plot(results, label="Study author",marker="D",markersize=10,linewidth=5,␣
 ↪color="black")
plt.bar(myears[11:], maverages[11:], align="edge", width=-0.3, color␣
 ↪="lightblue", label="Men")
plt.bar(fyears[11:], faverages[11:], align="edge", width=0.3, color␣
 ↪="lightpink", label="Women")

plt.legend(loc="upper left", fontsize="x-large")
plt.show()
```



Since the *Sleep quality* dataset includes data from 2014 to 2018, whereas the *American Sleep Study* dataset conatins data from 2003 to 2017, we ingored the results of our general study until 2014.

From this we see that on average, our author slept significally(~1.5 hours) less per night han the general population.

It should also be noted that the *Sleep quality* Dataset only starts in late December of 2014, so the only values we have for 2014 are heavily skewed by the holidays.

1.5.2 Time versus Quality

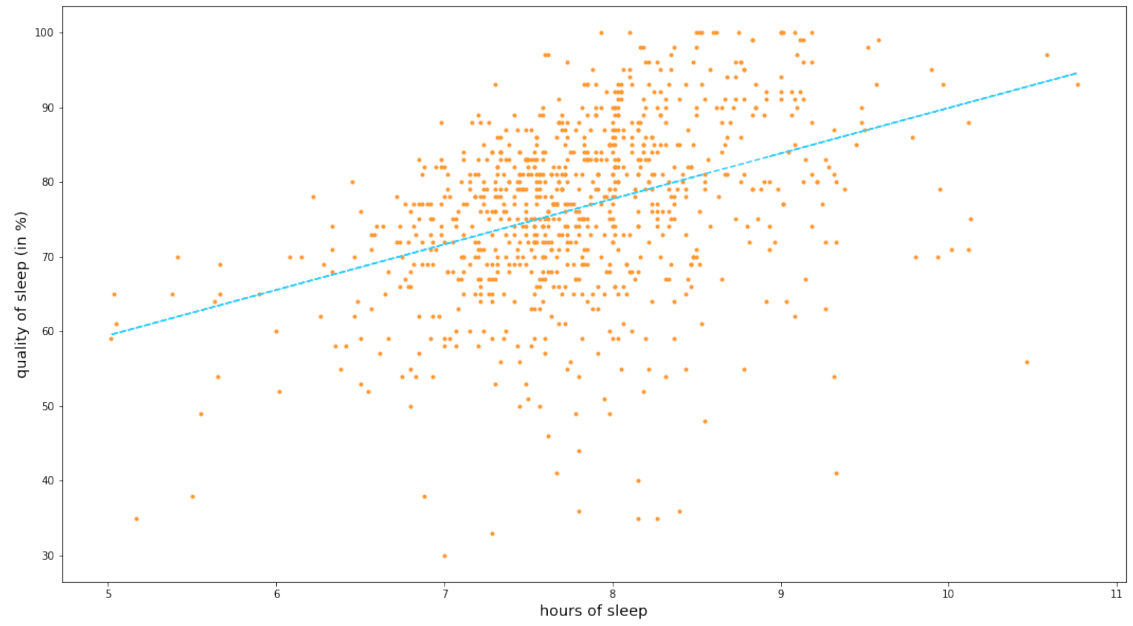
We can also try to find whether a correlation between the time spent sleeping and the quality of the sleep exists, and if so, how noticeable it is.

To do this, we query the "SleepQuality" document in the database and return a table with time and quality. Then we again parse the time from hh:mm format to a float.

We also remove the outliers with less than 4 hours of sleep per night, because those nights were most likely special circumstances which also affected sleep quality.

[13]:
```
quality=[]
time=[]
scatterdoc=colqual.find({},{"quality":1,"time":1})
for x in scatterdoc:
    temp=x["time"].split(":")
    if int(temp[0])>3:
        quality.append(x["quality"])
        time.append(int(temp[0])+int(temp[1])/60)

z = np.polyfit(time, quality, 1)
p = np.poly1d(z)


fig = plt.figure(figsize = (18, 10))
```

```
plt.xlabel("hours of sleep", fontsize="x-large")
plt.ylabel("quality of sleep (in %)", fontsize="x-large")

plt.scatter(time,quality,marker=".",color="#ff9933")
plt.plot(time,p(time),color="#33ccff",linestyle="--")

plt.show()
```



From this we can see a somewhat noticeable trend that the author generally experienced longer sleep as better in terms of quality. Interestingly, no night with less than 7 hours of sleep (the recommended minimum for healthy adults) was rated 100% for quality.

1.6 final Considerations

1.6.1 Challenges of Big Data

Challenge 1 (Volume) Since data Volume was not an issue, we stored our data locally in a MongoDB database inside a docker container, and for use with MapReduce also in .csv format. In case data size had increased beyond reasonable amounts, we would have switched to MongoDB's cloud service.

Challenge 2 (Variety) We used two different sources for our data, both in the form of discrete, structured datasets. Although a regular RDBMS would have probably sufficed, we decided on a document-oriented database in the form of MongoDB.

Challenge 3 (Velocity) The only factor that could slow down the database would be hardware or software bottlenecks of the PC where it is locally stored. Since it is a small volume of data high-end desktop, running off a modern SSD, delays were unnoticable. If Hardware bottlenecks were to become an issue though, it would be unlikely that any non-local solution would improve the situation, as a regular consumer-grade network connection would be far slower.

Challenge 4 (Veracity) The data quality from "Time Americans spend sleeping" from Makeover Monday (open data from US Government) and from "Sleep Data" from Dana Diotte was good enough to not have additional steps performed in relation to data cleaning.

The *American sleep study* dataset is anonymized, the *Sleep quality* dataset only includes personal data from it's creator.

Challenge 5 (Value) The *American sleep study* dataset is a very comprehensive view over a representative sample of the US population. The subject as a whole has gained more importance over the years as Sleep Science has been considered a significant value for productivity.

1.6.2 Layers

Data Source Layer:

- The data is structured
- 2 different soures (csv tables)
- 2 datasets all together
- collected by the US government and a single researcher

Data Storage Layer:

- Stored in a NoSQL database (MongoDB)
- Reason: It is easy to use for us
- Different collections for the different sources
- The data is checked via visual inspection before loading it into the database
  - Method #1: We checked the unique occuring values of each column in order to detect invalid fields
  - Method #2: We checked the float values of the sleep duration fields for impossible values like "smaller than 0" or "greater than 24")
- CAP theorem: what would be most important for us? (consistency, availability, partition tolerance):
  - cosnistency: priority #1, as multiple sections work with the same data. if it is changed, that will obfuscate our results.
  - availability: prority #2, after downloading the inital data source files, it is fully controlled by us
  - partition tolerance: priority #3, dataset size is far to low to require partition in the first place

Processing Layer:

- We access the storage via the pymongo python package
- The data is processed using python.

Data Output Layer:

- The data is visualized using the matplotlib python package.
- the MapReduce processes create .csv files to be interpreted by the script at a later time
- The output is used by people (us, our colleagues, possibly lecturers)