

Attribution 4.0 International (CC BY 4.0)

<https://creativecommons.org/licenses/by/4.0/> (<https://creativecommons.org/licenses/by/4.0/>)

```
date; find . | grep csv | while read full_path; do cat $full_path | awk -F',' 5 == "1" {print
```

```
0}' >> all_failures_all_smart_column_count_varies.log; done; date
```

count the number of columns in the file by counting commas per line

```
while read p; do echo $p | tr -cd ',' | wc -c; done <
```

all_failures_all_smart_column_count_varies.log Because the columns are inconsistent, check which columns are aligned date; find . | grep csv | while read full_path; do head -n1

full_path >> all_column_headers.log; done; date Now we can determine which columns are consistent for i in 1..130; do echo column i;

cat all_column_headers.log | cut -d',' -f\$i | sort | uniq -c; done answer: columns 1 to 31 are present in all CSVs. Reduce failure list to first 31 columns cat all_failures_all_smart_column_count_varies.log | cut -d',' -f1-31 > all_failures_smart_columns1to31.log

Smart_9_raw is power-on hours

https://en.wikipedia.org/wiki/S.M.A.R.T.#Known_ATA_S.M.A.R.T._attributes

(https://en.wikipedia.org/wiki/S.M.A.R.T.#Known_ATA_S.M.A.R.T._attributes)

```
In [1]: import pandas
print('pandas', pandas.__version__)
import glob
import pickle
import numpy
import time
import matplotlib.pyplot as plt
```

pandas 0.23.4

get the headers for the dataframe from a CSV

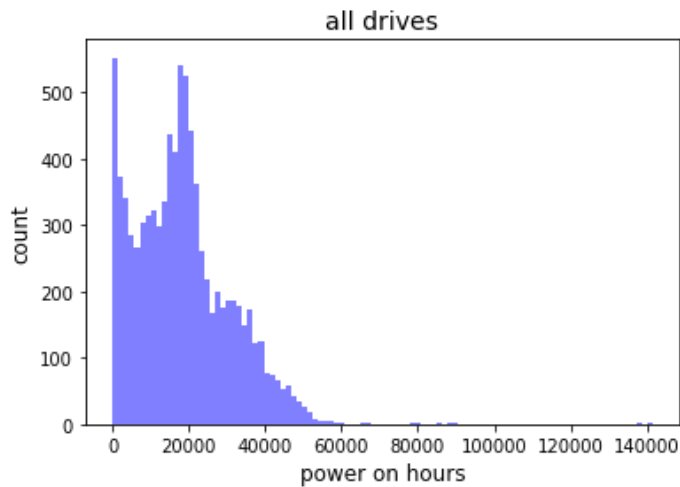
```
In [2]: df_header_only=pandas.read_csv('zipped_data/data_Q2_2018.zip_folder/2018-04-01.csv',
nrows=3)
#df_header_only.head()
```

```
In [3]: df = pandas.read_csv('data_synthesized_from_csvs/all_failures_smart_columns1to31.log', header=None)
df.columns=df_header_only.columns[0:31]
df.shape
```

Out[3]: (8743, 31)

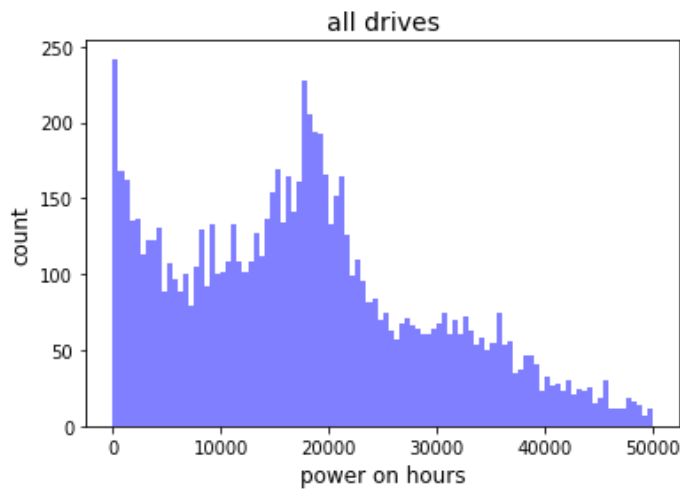
histogram of power-on hours at time of failure

```
In [4]: n, bins, patches = plt.hist(df['smart_9_raw'].dropna(how='any'), bins=100, facecolor='blue', alpha=0.5)
plt.xlabel('power on hours',fontsize=12)
plt.ylabel('count',fontsize=12);
plt.title('all drives',fontsize=14);
```

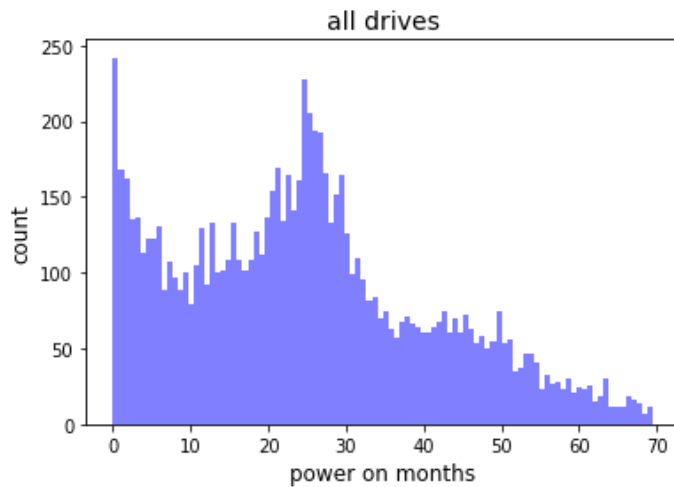


zoom in on relevant lifespans

```
In [5]: n, bins, patches = plt.hist(df[df['smart_9_raw'] < 50000]['smart_9_raw'].dropna(how='any'), bins=100, facecolor='blue', alpha=0.5)
plt.xlabel('power on hours',fontsize=12)
plt.ylabel('count',fontsize=12);
plt.title('all drives',fontsize=14);
```

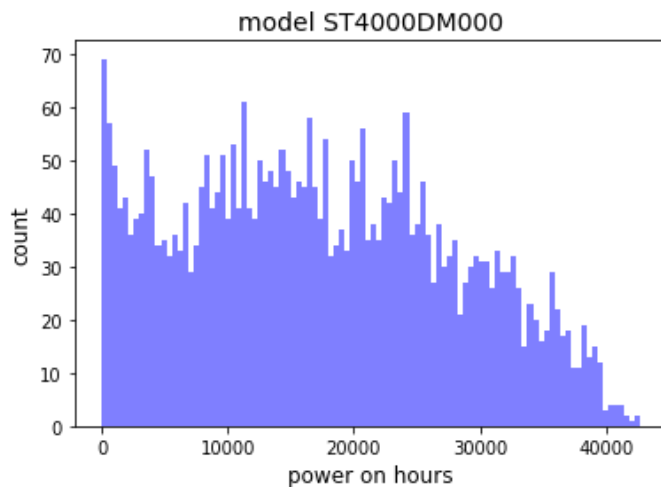


```
In [6]: n, bins, patches = plt.hist(df[df['smart_9_raw'] < 50000]['smart_9_raw'].dropna(how='any')/24/30, bins=100, facecolor='blue', alpha=0.5)
plt.xlabel('power on months',fontsize=12)
plt.ylabel('count',fontsize=12);
plt.title('all drives',fontsize=14);
```



Focus on most popular model, ST4000DM000

```
In [7]: n, bins, patches = plt.hist(df[df['model']=='ST4000DM000']['smart_9_raw'].dropna(how='any'), bins=100, facecolor='blue', alpha=0.5)
plt.xlabel('power on hours',fontsize=12)
plt.ylabel('count',fontsize=12);
plt.title('model ST4000DM000',fontsize=14);
```



```
In [8]: n, bins, patches = plt.hist(df[df['model']=='ST4000DM000']['smart_9_raw'].dropna(how='any')/24/30, bins=100, facecolor='blue', alpha=0.5)
plt.xlabel('power on months',fontsize=12)
plt.ylabel('count',fontsize=12);
plt.title('model ST4000DM000',fontsize=14);
```

