# Predictive Maintenance of Hard-Drive Disks

**Fabio Cecchi**

Springboard

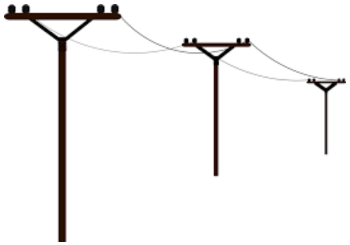
**Joint work with Paolo Lucchino**

June 19, 2018

# We constantly rely on fallible equipment

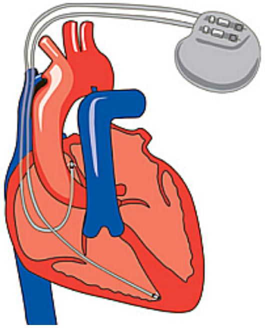
The car we use to commute

The laptop we use to work



The electric lines that serve our home

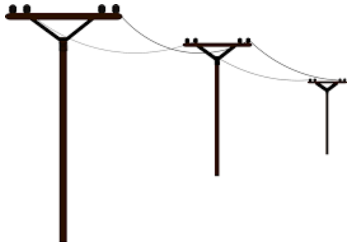
The pacemaker that keeps our heart beating



# When a failure happens, the cost is very high

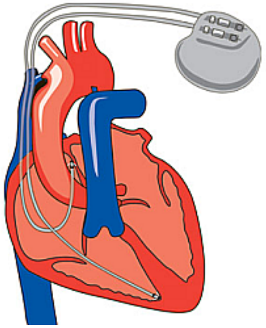
Emergency repair / Car accident

Loss of data / Impossibility to work



Network overcharge / Black outs

Urgent recovery



# Can we forecast a failure and prevent it?

## 1. Preventive replacement policy

Equipment is checked and replaced regularly,
thus minimizing the cost due to unplanned events.

- The policy largely **depends on the time-to-failure distribution** of the equipment.

- The policy is **very easy to implement**.

- The policy is **offline**: it does not cope well with unforeseen events.

# Can we forecast a failure and prevent it?

## 2. Predictive replacement policy

Place sensors in the equipment and measure real-time performance.
Use the collected data to predict whether the equipment is approaching a failure

- The policy **adapts to different the time-to-failure distribution** of the equipment.

- The policy is **more complicated to implement** and require data collection and analysis.

- The policy is **online**: it cope very well with unforeseen events.

# Can we forecast a failure and prevent it?

## 2. Predictive replacement policy

Place sensors in the equipment and measure real-time performance.
Use the collected data to predict whether the equipment is approaching a failure

- The policy **adapts to different the time-to-failure distribution** of the equipment.

- The policy is **more complicated to implement** and require data collection and analysis.

- The policy is **online**: it cope very well with unforeseen events.

Predictive replacement should not replace preventive replacement,
these policies should run in parallel

# Hard-Drive Disks (HDDs), the bricks of data centers

By 2020 **70% of all data** and **90% of all data center data** will be stored in HDDs (ref)

- Preventing the failure of HDDs plays a key role in designing efficient data centers

- In large data centers, failures of HDDs are taken into account by adding redundancy (ref), the same data is copied in multiple HDDs.

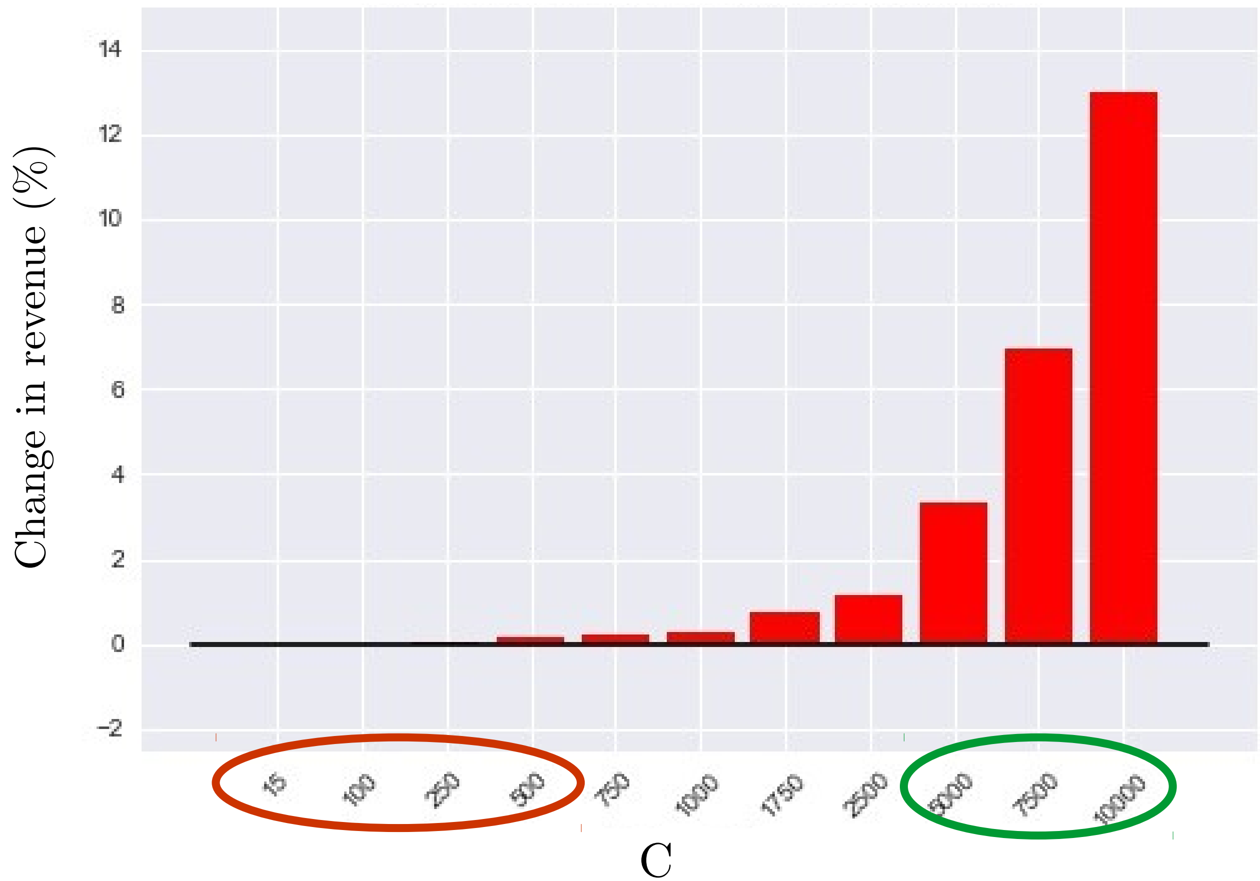
- Being able to forecast the failure of an HDD is convenient:
  - Data centers: Reduce the HDDs needed. More reliability = Less redundancy needed
  - Personal HDD: Reduce the risk of losing important data stored in it

**Can predictive maintenance be used to forecast whether HDDs are approaching failure?**

# Predictive maintenance may increase the revenue

Change in revenue as a function of the ratio C

C = (Cost per failure)/(Daily revenue of a working HDD)




**Low ratio C**

**Predictive maintenance does not help!**

**High ratio C**

**Predictive maintenance has a huge positive impact!**

# Table of contents

1) Description of the dataset

2) Exploratory data analysis & Inferential statistic

3) Optimization problem & Machine learning

4) Conclusion

# The dataset we use was released by Backblaze

Consider the data collected in 2015-17. It consists of **72 millions entries**.

Each entry describes the observation of an HDD on a **specific day**.

The sensors on an HDD capture the "Self-Monitoring, Analysis, and Reporting Technology" (**SMART**) metrics, proxies for the HDD performance.

In the dataset, the following features are present:

- **serial_number**: The unique ID of the HDD observed
- **model**: The model of the HDD observed
- **capacity**: The capacity of the HDD observed
- **date**: The day of the observation
- **failure**: A binary value, if equal to 0 the observed HDD did not fail on the day, if equal to 1 it did.
- **smart_n_raw**: The raw value of the SMART metric $n$ of the HDD
- **smart_n_normalized**: The normalized value of the SMART metric $n$ of the HDD

# What are the SMART metrics?

- The SMART metrics measure the performance of HDDs facing diverse tasks

**Which performance worsen as the HDD approaches failure?**

- The SMART metrics are **model-dependent**: a SMART metric can be absent for a specific model ($NaN$) and the interpretation of the same SMART metric may differ for the various HDDs producers

**We will analyze each model by itself**

- The normalized value of a SMART metric takes value in between 1 and 253, where 253, 100, 1 denotes optimal, typical, and terrible performance, respectively. The normalized values are normalized version of the raw metrics

**We will use only the normalized values and drop the raw values**

# Let's focus on a specific model

The code provided requires to specify which model we aim to analyze, here we present the result for **model Hitachi HDS722020ALA330**

| # Entries | # Unique HDDs | # Failures | % Failed HDDs |
|---|---|---|---|
| 2,608,197 | 4683 | 152 | 3.2% |

- We chose this model being average among the models present, both in terms of number of unique HDDs and percentage of failures observed

- Huge class imbalance!

In Notebook 2, we read the dataset on the Backblaze website,
we identify the **relevant SMART metrics** for a specific model,
and we store their value for the following analysis

# Relevant SMART metrics: a first feature selection

What makes a SMART metric relevant and worth being looked at?

(1) It has to be **present** for the specified model

(2) It has to show some **variation**, i.e., satisfy either of the following conditions:

- The normalized range has to be above a predefined range threshold (ex: 15)
- The variance of the normalized values above a variance threshold (ex: 1)

(3) It's **minimum** has to be below the typical normalized value (ex: 100)

**We apply the filter to the model Hitachi HDS722020ALA330**
**and only 7 (out of 45) SMART metrics remain**

| SMART metric | 1 | 5 | 7 | 8 | 192 | 193 | 196 |
|---|---|---|---|---|---|---|---|
| Minimum | 1 | 5 | 84 | 46 | 68 | 68 | 4 |
| Maximum | 100 | 100 | 100 | 123 | 100 | 100 | 100 |
| Standard dev. | 1.64 | 1.19 | 0.01 | 6.03 | 0.62 | 0.62 | 1.22 |

1) Description of the dataset

2) Exploratory data analysis & Inferential statistic

3) Optimization problem & Machine learning

4) Conclusion

# EDA: Relations between SMART metrics and failures

Exploratory Data Analysis (EDA) attempt to answer the following questions:

(1) Which SMART metrics vary as HDDs approach failure?

**For the model considered, only metrics 1, 5, and 196 provide useful information**

(2) How many failures can be prevented?

**Most HDDs fail without displaying any change in the metrics, roughly 10% of the failures appear to be detectable**

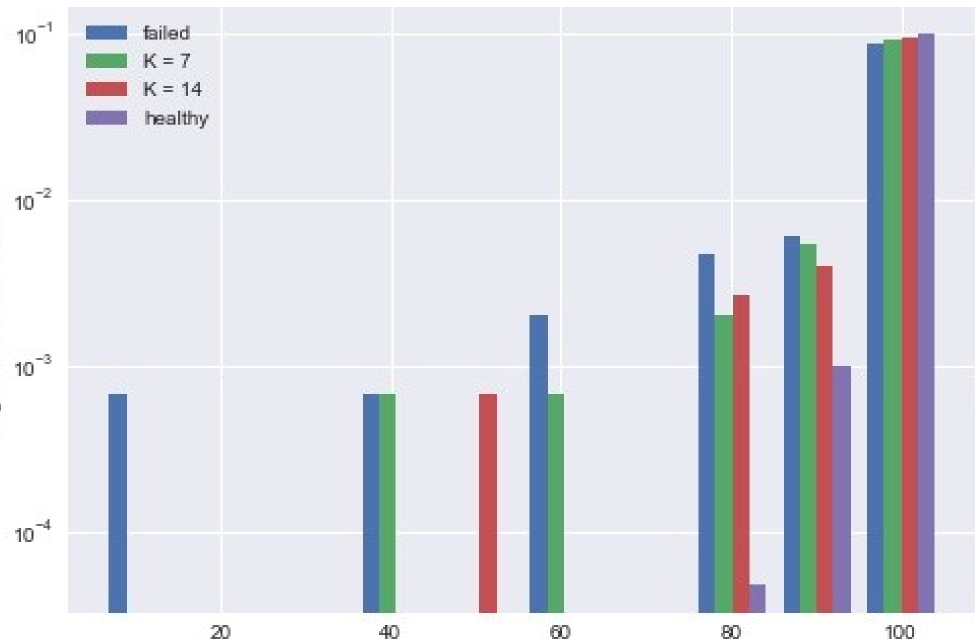
Here we display the results relative to few metrics, more details can be found in Notebook 3

# SMART metric distribution as approaching failure

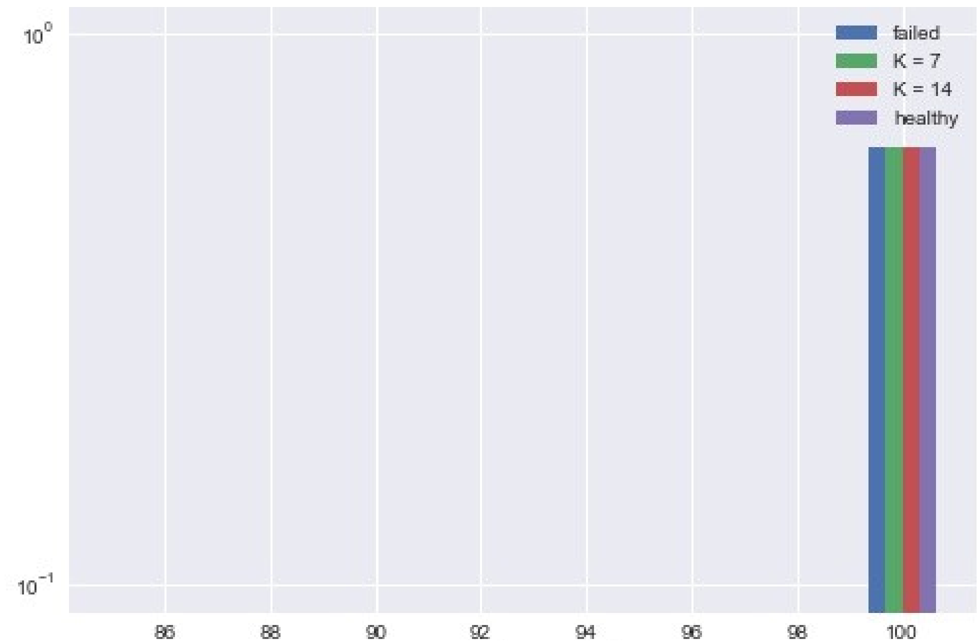
Compare the distribution of the SMART metrics extracted from:

- Healthy HDDs (more than 1 year from last observation in the dataset)
- HDDs at K days days from failure, for K = 0 (failing day), 7, 14

Logscale histogram of the normalized SMART metrics distribution




SMART metric 1




SMART metric 7

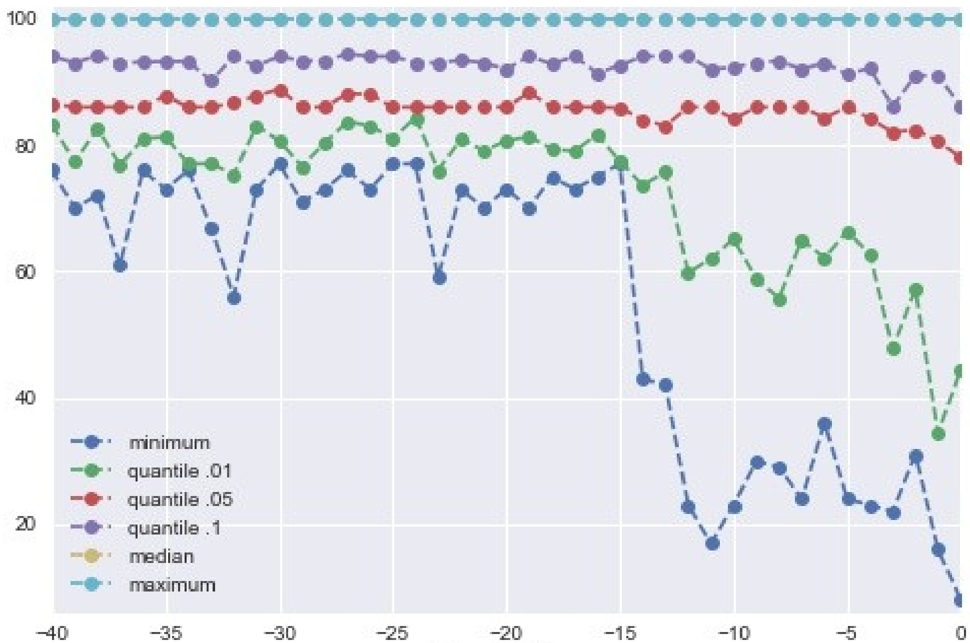
**Useful**: The metric value is lower as the HDDs approach failure

**Not useful**: No evident change in the metric value

# SMART metric distribution as approaching failure

Quantiles of the SMART metric values as a a function of the days to failure

SMART metric 1




SMART metric 5


- SMART metrics 1 and 5 worsen as the failure approaches

- Different way in which they decrease (fluctuations or neat jumps). SMART 196 has a behavior similar to SMART 5. The other metrics display flat plots

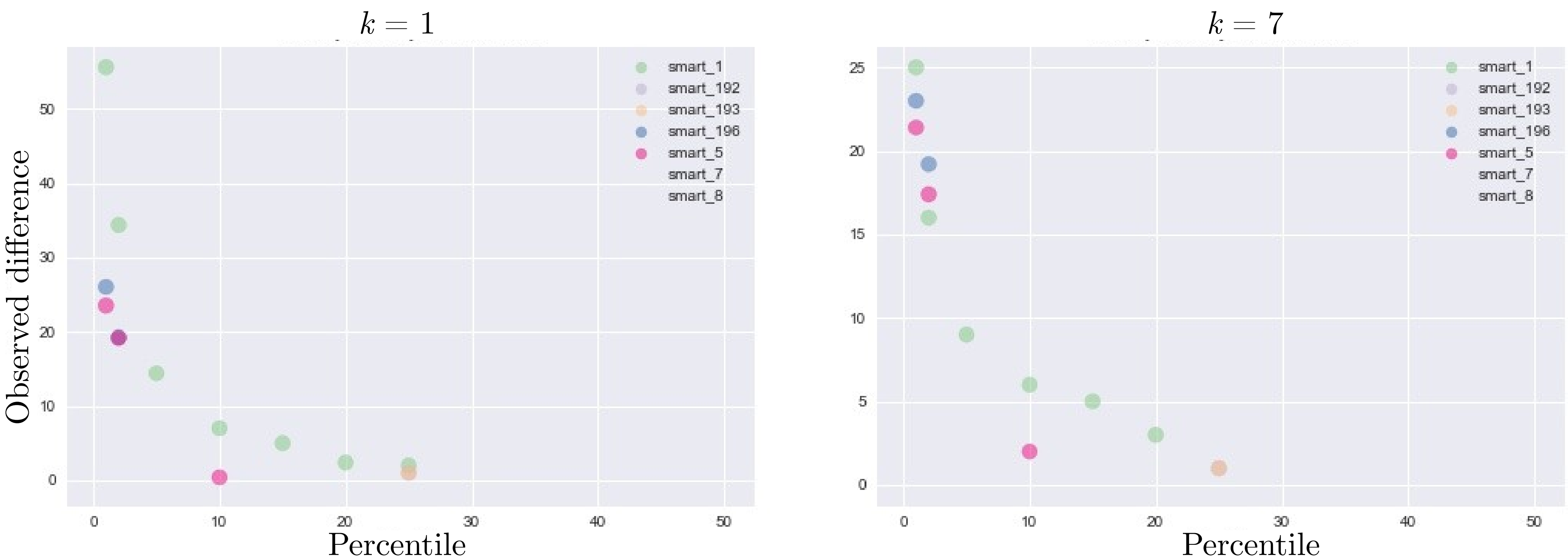
- The decrease starts a couple of weeks before the failure

- No more than 10% of the failing HDDs can be detected

# Inferential statistic: the quantiles of which SMART metrics are lower as HDDs approach failure

By means of bootsrapping: For every SMART metric s, test whether

$$\mathcal{H}_0 : \quad \begin{matrix} n\text{-th percentile} \\ k \text{ days from failure} \end{matrix} \quad - \quad \begin{matrix} n\text{-th percentile} \\ \text{healthy HDDs} \end{matrix} \quad \geq 0$$




- Display the dot only if p-value $< 0.1$, i.e., if we reject the hypothesis

- The difference is significant for SMART values 1, 5, and 196 even one week prior to the failure and up to percentile 10

1) Description of the dataset

2) Exploratory data analysis & Inferential statistic

3) Optimization problem & Machine learning

4) Conclusion

# What is our goal? A simple mathematical model

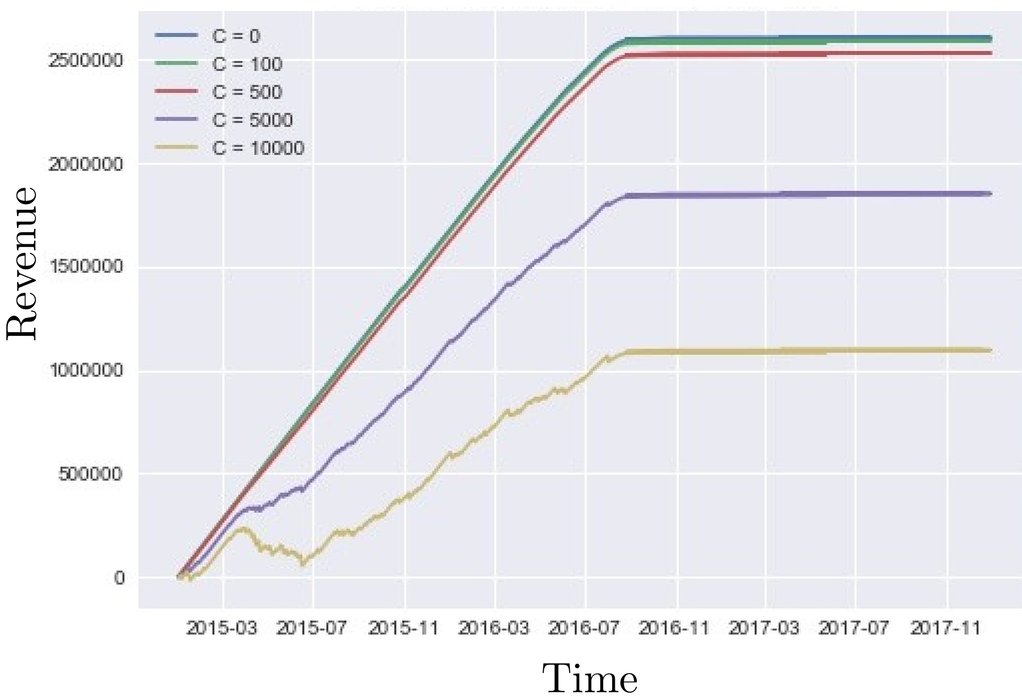
Let us make the following assumprions:

- An HDD generates a profit of 1 for everyday it is active

- When an HDD fails, we have to pay C

- The revenue R is determined by all the profits generated by active HDDs minus all the costs incurred

**Goal: maximize the total revenue R**




(Left) Benchmark revenue obtained by means of the preventive replacement policy currently implemented.

**Can we do better by designing efficient predictive replacement policies?**

# Capture the trade off: is it worse to turn off an healthy HDD or to miss a failure?

The answer is “**it depends on the application**”, or better it depends on the value of C

In Notebook 4 we derive various predictive replacement policies
By means of machine learning techniques

These predictive replacement policies solve the trade off in an intuitive way:

- **Low C = Cautious replacement**. It is too risky to turn off healthy nodes, it is better to miss a failure if we are not very certain of an imminent failure.

A policy is efficient if the number off **false positive is low**

- **High C = Aggressive replacement**. Catching a failure pays back for the profit missed by turning off an healthy HDD.

A policy is efficient if the number off **true positive is high**

# When shall an HDD be considered close to failure?

From the EDA performed, the relevant metrics decrease 2 weeks before the actual failure:

**Label with 1 the observations of HDDs failing in the next 14 days**

- We now present four different learning techniques that aim to distinguish healthy HDDs and close to failure. For more details, see Notebook 4

- From each classifier we **derive a predictive replacement policy:** an HDD predicted to be close to failure is turned off for good

- **Each predictive replacement policy is evaluated on the same testing dataset** consisting of $\frac{1}{4}$ of the HDDs (not used in the training procedure) and we compute the revenue obtained by flanking the predictive replacement policy to the currently used preventive replacement one

# Method 1: Classifier – Daily

- Each observation is taken separately

- To account for temporal dependence, we add features denoting the change in the SMART metric values with respect to 1, 7, and 14 days prior the observation.

- Too many features and observations to use KN_neighbors or SVM, hence we use a **Random Forest classifier**

- We tune the **number of trees**, the **maximum number of features** per tree, and the acceptance **threshold probability** by means of a 3-fold cross validation.

| C | 15 | 100 | 250 | 500 | 1000 | 2500 | 5000 | 10000 |
|---|---|---|---|---|---|---|---|---|
| Change R | 0% | 0% | 0% | 0% | 0% | 0% | 2.43% | 5.09% |
| False Pos. | 0 | 0 | 0 | 0 | 0 | 0 | 12 | 12 |
| True Pos. | 1 | 1 | 1 | 1 | 1 | 1 | 3 | 3 |

**Slight increase in the revenue for large values of C**

# Method 2: Classifier – Time Series

- Every 5 days, we observe the last 15 days of the HDD evolution

- Use the **Dynamic Time Warping (DTW)** distance to compare time series. A flexible distance that allows to compare time series of different lengths

- We define 6 **benchmark time series** and compute their DTW distance from every SMART metric time series. The distances obtained are the features of the training dataset

- We use a **Random Forest Classifier**

- We tune the **number of trees**, the **maximum number of features** per tree, and the acceptance **threshold probability** by means of a 3-fold cross validation

| C | 15 | 100 | 250 | 500 | 1000 | 2500 | 5000 | 10000 |
|---|---|---|---|---|---|---|---|---|
| Change R | 0% | 0% | 0% | 0% | 0% | 1.13% | 3.03% | 10.62% |
| False Pos. | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 2 |
| True Pos. | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 |

**Method 2 outperforms Method 1, still no improvement for low C**

# Method 3: Survival Analysis

- Use the **Cox Proportional Hazard Regression** model from **package** lifelines to predict the impact of the SMART metric values on the survival function

- We predict the survival function for every observation in the test dataset. We predict the probability that an HDD initialized with the observed SMART metric values at time 0 will survive for $d$ days for different values of $d$

- We tune the parameters **days_to_fail** and **prob_threshold**. An observation is predicted as close to failure if the probability to survive after days_to_fail is lower than prob_threshold

| C | 15 | 100 | 250 | 500 | 1000 | 2500 | 5000 | 10000 |
|---|---|---|---|---|---|---|---|---|
| Change R | -0.01% | 0.01% | 0.06% | 0.14% | 0.31% | 0.88% | 2.14% | 7.28% |
| False Pos. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| True Pos. | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |

**This method shows no false positive, good for low values of C**

# Method 4: Ensemble Method

- Each predictive replacement policy derived identify HDDs close to failure by looking at **different patterns in the dataset**

- The predictive replacement policy derived by the ensemble method prescribes to turn off an HDD if **at least one** of the other policies prescribe to do it

- No need to tune parameters (already done for the specific methods ensembled)

| C | 15 | 100 | 250 | 500 | 1000 | 2500 | 5000 | 10000 |
|---|---|---|---|---|---|---|---|---|
| Change R | -0.01% | 0.01% | 0.06% | 0.14% | 0.31% | 1.17% | 3.35% | 13.00% |
| False Pos. | 0 | 0 | 0 | 0 | 0 | 2 | 13 | 13 |
| True Pos. | 3 | 3 | 3 | 3 | 3 | 4 | 5 | 5 |

**This method captures 13% (5 out of 38) of the failing HDDs present!**

1) Description of the dataset

2) Exploratory data analysis & Inferential statistic

3) Optimization problem & Machine learning

4) Conclusion

# What has been done . . .

- We analyzed a **very imbalanced** data set. Only 3% of the HDDs present fail. Out of those, roughly 10% of the failing HDDs displays changes in the SMART metrics as approach failure, i.e., **0.3% of all HDDs**.

- We derived a **simple filter** to determine which SMART metrics could be relevant in forecasting failures of HDDs of each specific model

- We formulated a simple **optimization problem** which explains the trade off between having few false positives and many true positives

- We trained **four different learning models** and derived the relative predictive replacement policies

- We **increased the revenue** substantially in the regime with high values of C

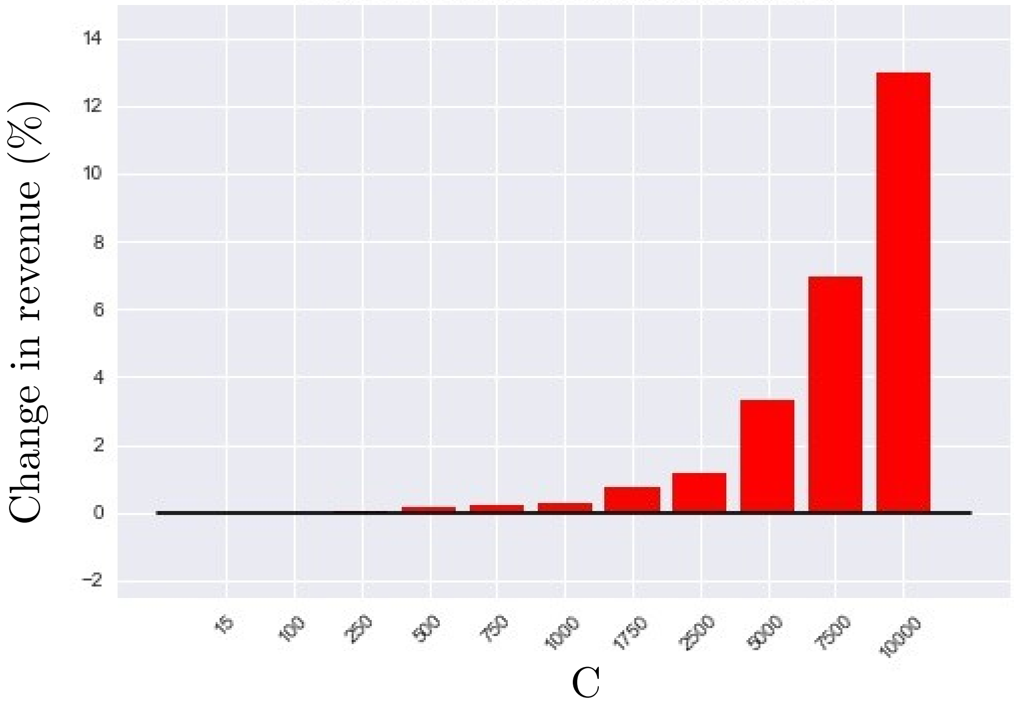
# . . . what we learned

- Most of the failing HDDs do not show any worsening in the SMART metrics performance. It is impossible to forecast most failures.

    - Sample observations **more frequently** (ex: hourly)?

    - Design **different SMART metrics**?

- This dataset is sufficient to achieve **significant increase in revenue for large values of C**




- To increase the revenue **for low values of C, we need a different dataset**