Paper- RM03

RBQM and Failure Prediction Using Python

Vidya Muthukumar, IQVIA Biotech, USA

Summary

CROs are looking to reduce costs where trials can be effectively monitored remotely. Risk-based Quality Management (RBQM), for remote device trials has emerged as a strategic approach using advanced data analytics. RBQM optimizes error detection to facilitate replacement of even remote sites. RBQM aims to focus on remote monitoring for device trials likely to affect patient safety and data quality so that investigators address errors before they compromise trial quality. This paper showcases how RBQM along with failure prediction for health monitoring devices using Python are used to monitor patients' parameters at homes. However, verifying if device measurements are not impacted by their technical fault, is a challenging problem. With statistical analyses and a logistic regression model for RBQM, the attempt is to predict device failure with a daily aggregated sample device telemetry readings from a clinical trial using Python to illustrate risk-mitigating approaches and fix errors for an effective RBQM device trial.

Data Structure Specifications At-a-Glance

The data structure is a log flat file of telemetry data
date: YYYY-MM-DD
Device_id: device_id
Failure 0 - no, 1 – yes
Attribute1/attribute9: daily aggregated
telemetry log data
Test data: device_failure.csv.zip

	A	B	C	D	E	F	G	H	I	J	K	L
1	date	device	failure	attribute1	attribute2	attribute3	attribute4	attribute5	attribute6	attribute7	attribute8	attribute9
2	1/1/2015	S1F01085	0	215630672	56	0	52	6	407438	0	0	7
3	1/1/2015	S1F0166B	0	61370680	0	3	0	6	403174	0	0	0
4	1/1/2015	S1F01E6Y	0	173295968	0	0	0	12	237394	0	0	0
5	1/1/2015	S1F01JE0	0	79694024	0	0	0	6	410186	0	0	0
6	1/1/2015	S1F01R2B	0	135970480	0	0	0	15	313173	0	0	3
7	1/1/2015	S1F01TD5	0	68837488	0	0	41	6	413535	0	0	1
8	1/1/2015	S1F01XDJ	0	227721632	0	0	0	8	402525	0	0	0
9	1/1/2015	S1F023H2	0	141503600	0	0	1	19	494462	16	16	3
10	1/1/2015	S1F02A0J	0	8217840	0	1	0	14	311869	0	0	0
11	1/1/2015	S1F02DZ2	0	116440096	0	323	9	9	407905	0	0	164
12	1/1/2015	S1F02EVN	0	112348104	0	0	0	7	388146	0	0	1
13	1/1/2015	S1F02L38	0	223938928	0	0	0	2	215169	0	0	8
14	1/1/2015	S1F02MGA	0	44399688	0	266	1	6	399286	0	0	2269
15	1/1/2015	S1F02P76	0	104131304	1536	0	175	11	301679	0	0	0
16	1/1/2015	S1F02VAX	0	61019512	168	2	521	3	380496	0	0	3
17	1/1/2015	S1F02WFT	0	44348552	5160	14	1074	11	249515	0	0	21
18	1/1/2015	S1F0318A	0	35018688	0	0	0	9	394890	0	0	5
19	1/1/2015	S1F0322R	0	34540712	0	0	0	9	411399	0	0	0
20	1/1/2015	S1F0330P	0	125539768	0	0	12	14	297284	0	0	5
21	1/1/2015	S1F035SJ	0	220392160	0	0	0	9	389730	0	0	0

device_failure

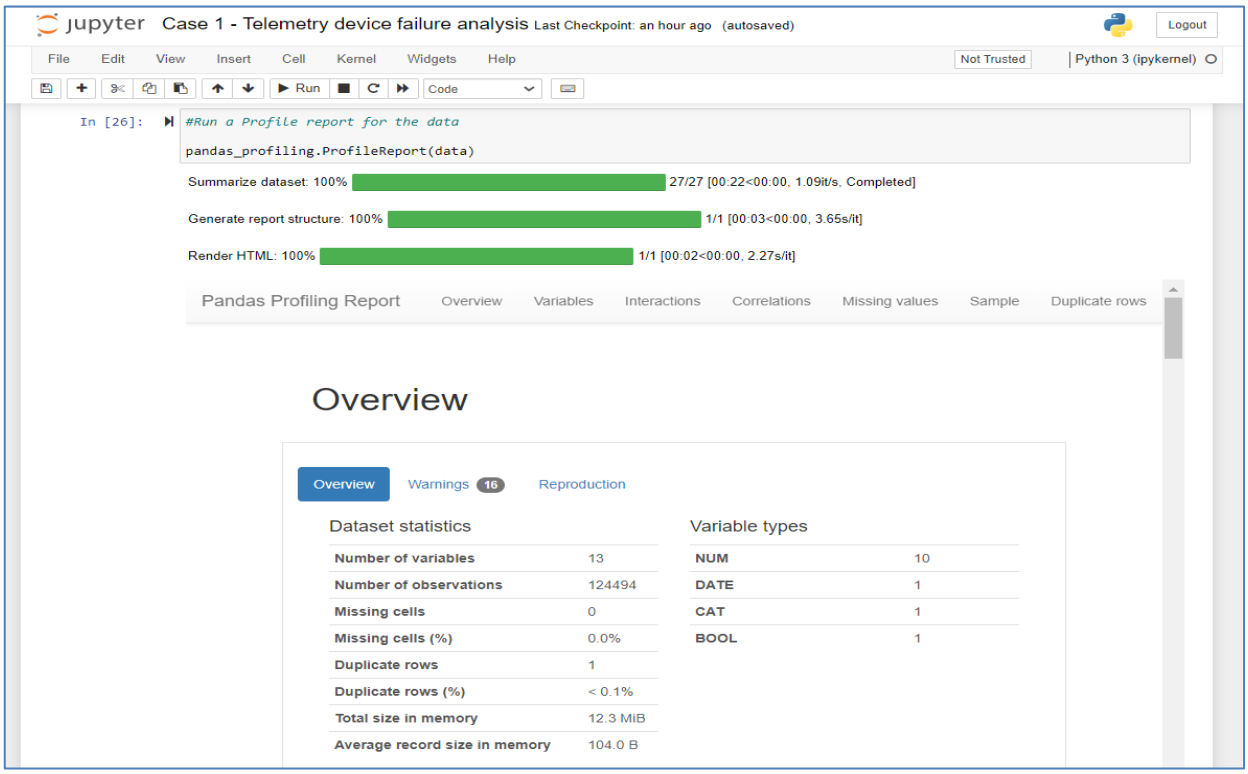
Analysis: Model selection and effectiveness

Technology: Python

Problem Statement: Predict device failure given the daily aggregated device telemetry attribute log readings

Assumptions: Each device is being used by one-person only. Device can be for repeat use at different dates by same person. This is why we can use the values of the attributes as explanatory variables. Otherwise, the values of the attributes can change if two different people use the same device, and it may not be because the device failed.

Data Analysis for Model



The objective is to understand what the data is and how it is structured. The analysis is about device failure, so we needed data at the device level.

- The first step was data cleaning by identifying duplicate observations, duplicate variables, large outlier values etc. I used Python tools such as ProfileReport, Box plots.
- Using logistic regression to model device failure (dependent variable) based upon the given attributes (independent variables).
- To understand what the telemetry data is and how it is structured, relevant to the device telemetry log data analysis.
- Validation with the p-value and classification tables and see if the results are significant and not due to random fluctuation

The next step is to run the ProfileReport

Profile Report Summary

- No missing values in telemetry dataset and the data is integer type.
- The dataset is not balanced since < 0.1% of the records are failures. Data is not at the device level, it is at the device-date level, so the same device is used over many days, there are multiple observations for each device.
- Attributes 2,3,4,7,9 are highly skewed.
- Attributes 7 and 8 are the exact same data, so I will drop attribute 8.
- There is one duplicate row which I dropped

jupyter Case 1 - Telemetry device failure analysis Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Logout

Run Code

In [1]:

```
####Case 1 uses aggregated telemetry data for 9 attributes with date, device and failure

####Objective is to predict failure by device given the current 9 attributes, deviceID, date and Boolean failure data.

####Import key packages and load Case 1 data
import pandas as pd
import os
import seaborn as sns
import numpy as np
import pandas_profiling
from pandas_profiling import ProfileReport
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report
%matplotlib inline

###Quick read of the head/tail data
data=pd.read_csv('Case_1_Data_device_failure.csv')
data.head(5)
```

Out[1]:

	date	device	failure	attribute1	attribute2	attribute3	attribute4	attribute5	attribute6	attribute7	attribute8	attribute9
0	1/1/2015	S1F01085	0	215630672	56	0	52	6	407438	0	0	7
1	1/1/2015	S1F0166B	0	61370680	0	3	0	6	403174	0	0	0
2	1/1/2015	S1F01E6Y	0	173295968	0	0	0	12	237394	0	0	0
3	1/1/2015	S1F01JE0	0	79694024	0	0	0	6	410186	0	0	0
4	1/1/2015	S1F01R2B	0	135970480	0	0	0	15	313173	0	0	3

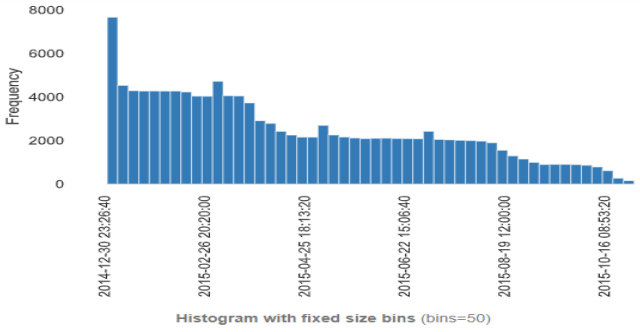
In [2]:

```
data.tail()
```

Out[2]:

	date	device	failure	attribute1	attribute2	attribute3	attribute4	attribute5	attribute6	attribute7	attribute8	attribute9
124489	11/2/2015	Z1F0MA1S	0	18310224	0	0	0	10	353705	8	8	0
124490	11/2/2015	Z1F0Q8RT	0	172556680	96	107	4	11	332792	0	0	13
124491	11/2/2015	Z1F0QK05	0	19029120	4832	0	0	11	350410	0	0	0
124492	11/2/2015	Z1F0QL3N	0	226953408	0	0	0	12	358980	0	0	0
124493	11/2/2015	Z1F0QLC1	0	17572840	0	0	0	10	351431	0	0	0

Pandas Profiling Report Overview Variables Interactions Correlations Missing values Sample Duplicate rows



Histogram with fixed size bins (bins=50)

jupyter Case 1 - Telemetry device failure analysis Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

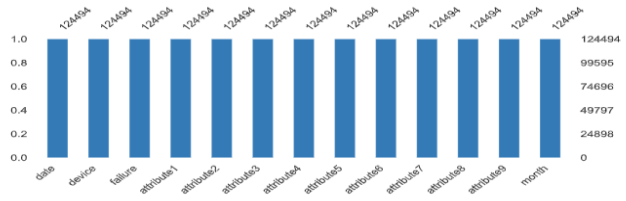
Logout

Run Code

Pandas Profiling Report Overview Variables Interactions Correlations Missing values Sample Duplicate rows

Missing values

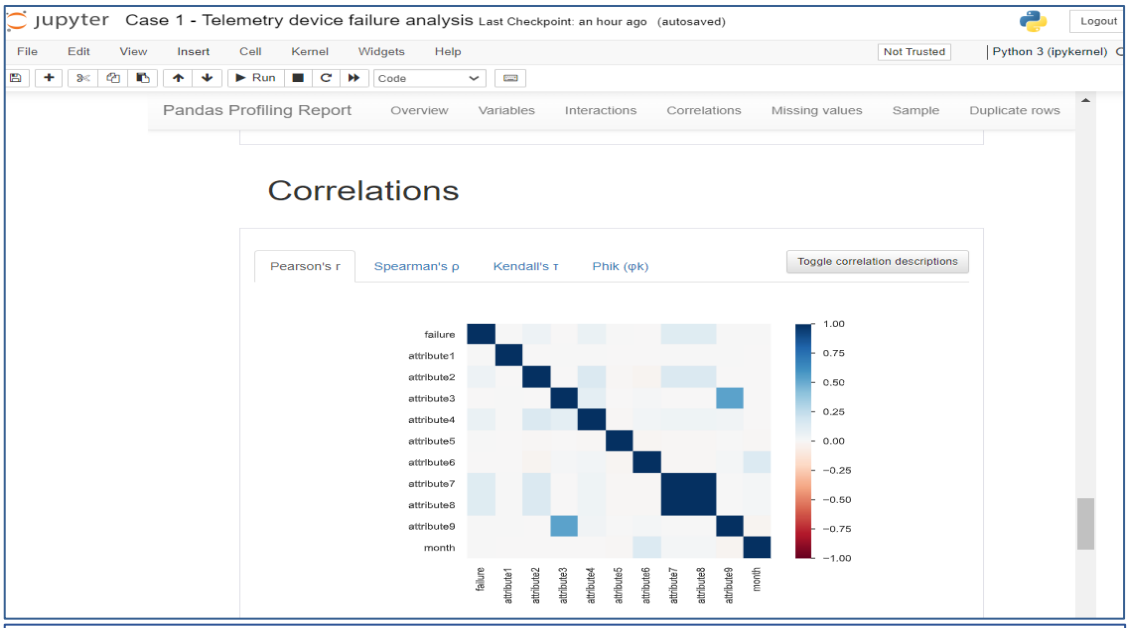
Count Matrix

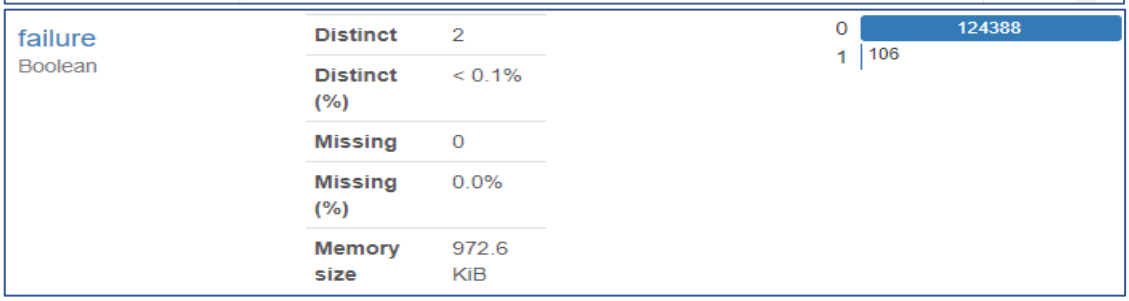


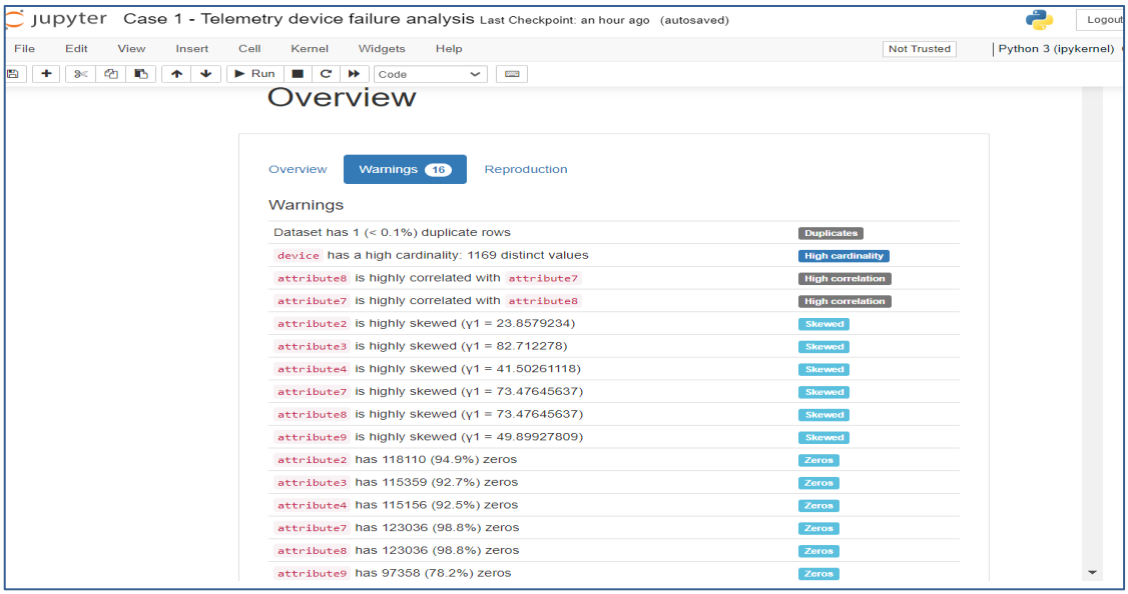
Duplicate rows

Most frequent

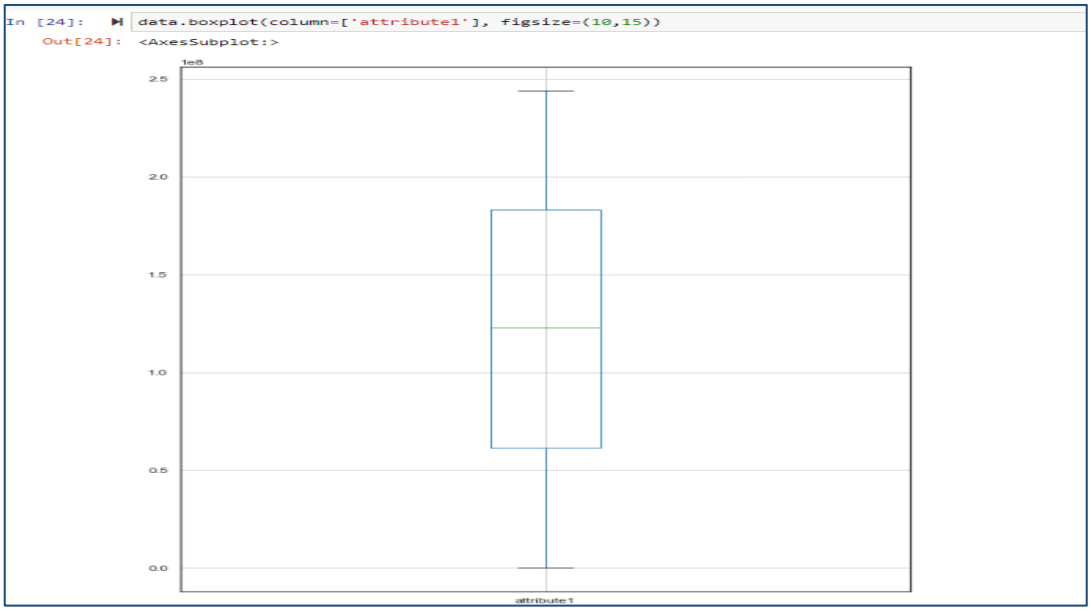
	date	device	failure	attribute1	attribute2	attribute3	attribute4	attribute5	attr
0	2015-07-10	S1F0R4Q8	0	192721392	0	0	0	8	213

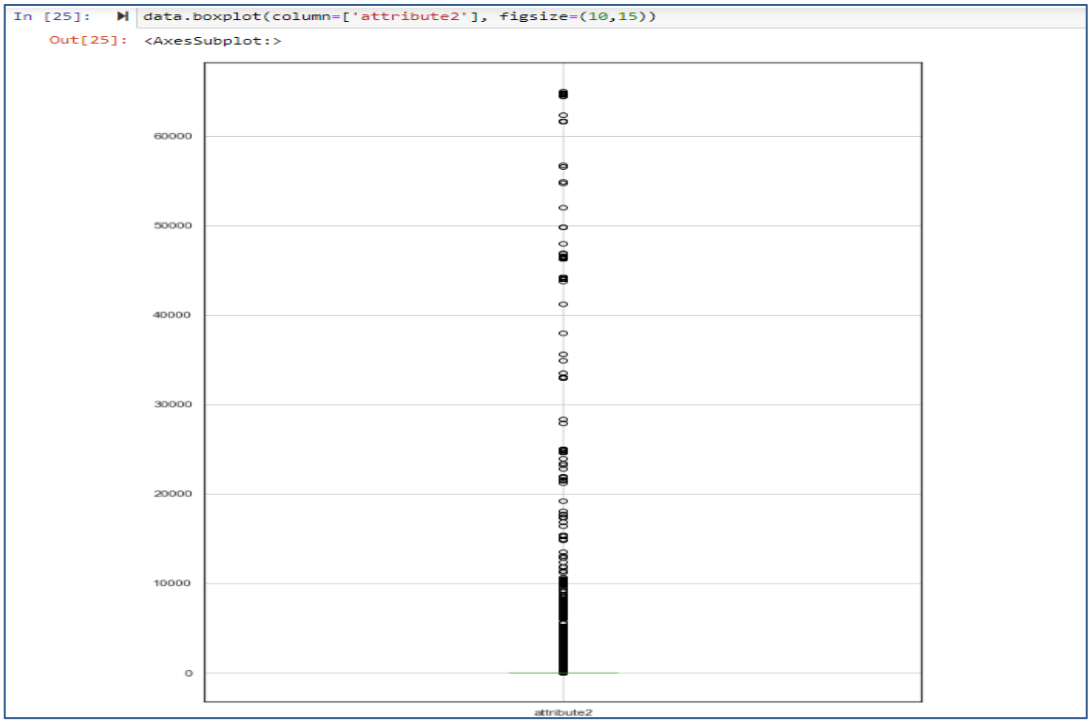

jupyter Case 1 - Telemetry device failure analysis Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Pandas Profiling Report
Overview
Variables
Interactions
Correlations
Missing values
Sample
Duplicate rows
Correlations
Pearson's r
Spearman's ρ
Kendall's τ
Phik (φk)
Toggle correlation descriptions
failure
attribute1
attribute2
attribute3
attribute4
attribute5
attribute6
attribute7
attribute8
attribute9
month
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


failure
Boolean
Distinct 2
Distinct (%) < 0.1%
Missing 0
Missing (%) 0.0%
Memory size 972.6 KiB
0 124388
1 106


jupyter Case 1 - Telemetry device failure analysis Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Overview
Overview
Warnings 16
Reproduction
Warnings
Dataset has 1 (< 0.1%) duplicate rows Duplicates
device has a high cardinality: 1169 distinct values High cardinality
attribute8 is highly correlated with attribute7 High correlation
attribute7 is highly correlated with attribute8 High correlation
attribute2 is highly skewed (γ1 = 23.8579234) Skewed
attribute3 is highly skewed (γ1 = 82.712278) Skewed
attribute4 is highly skewed (γ1 = 41.50261118) Skewed
attribute7 is highly skewed (γ1 = 73.47645637) Skewed
attribute8 is highly skewed (γ1 = 73.47645637) Skewed
attribute9 is highly skewed (γ1 = 49.89927809) Skewed
attribute2 has 118110 (94.9%) zeros Zeros
attribute3 has 115359 (92.7%) zeros Zeros
attribute4 has 115156 (92.5%) zeros Zeros
attribute7 has 123036 (98.8%) zeros Zeros
attribute8 has 123036 (98.8%) zeros Zeros
attribute9 has 97358 (78.2%) zeros Zeros

One can visualize the box plot for each attribute





Besides Attribute 1, 6 and 5 where the box plot could be visualized, there are too many data points for the box plots. After a point, the points overload and typically are not very informative. The Box Plots show the median of the dataset (the vertical line in the middle), as well as the interquartile ranges (the ends of the boxes) and the minimum and maximum values of the chosen dataset feature (the far end of the “whiskers”).

Model and analysis

I initially used the statistical tools (e.g., logistic regression) available in the sklearn.linear_model package. However, at a later stage, I realized that this package does not automatically provide the pvalues for logistic regressions, so I additionally used logistic regressions from the statsmodels.api package for the analysis.

Since the failure is at the device level (i.e., a device fails or not), I decided to do the analysis at the device level. For this purpose, I decided to use the mean values of the attributes for each device as the independent variables. The logic was that a device can fail and this can show up in the attributes data in one of two ways:

i) The device could be recording the correct levels of the data when it is working properly but could record a lower value when it fails. Under this scenario, a lower value for the attribute would be indicative of device failure.

ii) But, the device could also record higher values when it fails. It could go either way since we dont have information on how the device functions when it fails. Under this scenario, a higher value for the attribute would be indicative of failure.

Highlights of the analysis using logistic regression from the sklearn package:

a) The sklearn package by default has settings for data regularization for logistic regression and imposes a penalty as part of such regularization of the data.

b) Importantly, the package does not automatically report the p-values.

c) It also provides a few different algorithms for estimating the model.

d) But the regression coefficients were all very small, and given the lack of p-values, I did not assess the goodness of fit.

```
In [95]:
# Step 3: Create a model and train it
model = LogisticRegression(fit_intercept = True, C = 1e12)
model.fit(device_means2, np.ravel(max_result,order='C'))
model.fit(device_means2, max_result)
```

```
In [96]:
print (model.coef_)
#print (model.intercept)

[[-1.45465653e-08  9.79205536e-08 -1.01467323e-09  1.02154132e-09
   2.09212390e-10 -2.03640257e-06  2.61251056e-10 -1.44239983e-09]]
```

```
In [114]:
dev_fail = logit( "failure ~ attribute1 + attribute2 + attribute3 + attribute4 + attribute5 + attribute6 + attribute7 + attri
                  device_means3,).fit()

Optimization terminated successfully.
         Current function value: 0.295114
         Iterations 10
```

So, I decided to look for an alternative approach, and decided to use logistic regressions available in the **statmodels** package. The summary of the analysis is below:

a) After fitting the model, I printed the summary of the results.

b) Using a cutoff of 10% for assessing statistical significance, I found that attributes 2, 5, and 7 are positively associated with device failure – higher values indicate a higher likelihood that the device has failed. The other attributes were not statistically significant, suggesting that we cannot reject the base hypothesis that these other attributes had no ability to predict failure.

c) In terms of goodness of fit, the pseudo r-square was small, at less than 3%. This indicates that the fit of the regression is not very good.

d) To further analyze this lack of fit, I printed out the classification table. The model was good at predicting the true negatives (1060 out of 1063 non-events). But the model was not good at correctly classifying device failures or true positives – it correctly classified only 2 out of the 106 total device failures.

As an added check, I compared the results of this analysis with the output I got via SAS (not shown in this report). Since the numbers were very close, I am confident that there are no logical or data errors in the analysis I have done with python.

```
In [115]: print(dev_fail.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                failure   No. Observations:                 1169
Model:                          Logit   Df Residuals:                     1160
Method:                           MLE   Df Model:                            8
Date:                Wed, 17 Aug 2022   Pseudo R-squ.:                 0.02954
Time:                        11:50:30   Log-Likelihood:                -344.99
converged:                       True   LL-Null:                       -355.49
Covariance Type:            nonrobust   LLR p-value:                  0.007134
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -2.5583      0.686     -3.727      0.000      -3.904      -1.213
attribute1  1.641e-09   4.97e-09      0.330      0.741    -8.1e-09    1.14e-08
attribute2  2.806e-05   1.61e-05      1.744      0.081   -3.47e-06    5.96e-05
attribute3    -0.0006      0.002     -0.330      0.741      -0.004       0.003
attribute4     0.0023      0.001      1.609      0.108      -0.000       0.005
attribute5     0.0159      0.006      2.558      0.011       0.004       0.028
attribute6 -8.942e-07   1.07e-06     -0.838      0.402   -2.98e-06    1.2e-06
attribute7     0.0154      0.009      1.803      0.071      -0.001       0.032
attribute9  -6.39e-05      0.000     -0.229      0.819      -0.001       0.000
==============================================================================
```

```
In [119]: dev_fail.pred_table()
```

```
Out[119]: array([[1060.,    3.],
                 [ 104.,    2.]])
```

Conclusion and suggestions for further analysis

The logistic regression model seems to perform well in identifying true negatives but does not do as well in predicting true positives. The overall explanatory power of the model seems to be quite low.

There are several potential issues that could be contributing to the model's poor performance. These also point towards some additional analyses that could be done. I am highlighting a few key points:

i) We are assuming that each device is being used by one person only. But, if many people use the same device, then the telemetry aggregate values (values for the attributes1-9 in the dataset) can change not because the device failed but simply because different people used the same device.

ii) Further, it is possible that once a device has failed, it could be repaired and put back into service. While I have verified that no device failed a second time, it is quite possible that some of the data is from the dates AFTER the device failed (i.e., when it was repaired and put back into use). In predicting failure, such data that are after the failure date should be excluded from the analysis and could be contributing to the model's poor fit.

iii) Instead of the mean of the attribute values for each device, one could use the standard deviation and could possibly get a better fit. The reasoning is that if the device records a different value when it fails (compared to when it is working well), then the recorded values when the device fails would be different from when it was working. This would lead to a higher standard deviation of the values of each attribute, per device.

iv) Along similar lines, I could take the absolute value of the daily change in attribute value as the explanatory variable. The analysis could then be done at the device level using the last day of use for each device, or at the device-day level in which case the entire dataset could be used without having to aggregate it by device (like I have done in the current analysis).

v) If we have more information on how the device records data, one could try and transform the data in other sensible ways (e.g., convert some of the attributes to categorical data instead of continuous data and then do the analysis).

vi) Of course, it is always possible that the logistic regression is not a good model in this case, and one could use other techniques to do the classification such as nearest neighbor techniques, discriminant analysis, neural network-based methods, etc.