Practical 1

To perform Exploratory Data Analysis on Automobile data.

Aim: To perform Exploratory Data Analysis on Automobile data.

Prerequisites: Automobile data, Jupyter Notebook / Google Colab

Theory:

➢ What is Exploratory Data Analysis?

Exploratory Data Analysis (EDA) is **an approach to analyze the data using visual techniques**. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

An EDA is a thorough examination meant to uncover the underlying structure of a data set and is important for a company because **it exposes trends, patterns, and relationships that are not readily apparent**.

What are the types of exploratory data analysis?

The four types of EDA are

1. univariate non-graphical,
2. multivariate non- graphical,
3. univariate graphical,
4. multivariate graphical.

➢ Techniques and Tools:

There are a number of tools that are useful for EDA, but EDA is characterized more by the attitude taken than by particular techniques.
Typical graphical techniques used in EDA are:
Box plot
Histogram
Multi-vari chart
Run chart
Pareto chart
Scatter plot (2D/3D)
Stem-and-leaf plot
Parallel coordinates
Odds ratio
Heat map
Bar chart
Horizon graph
Dimensionality reduction:
Multidimensional scaling

Principal component analysis (PCA)
Multilinear PCA
Iconography of correlations

- ## Non-Graphical Exploratory Data Analysis

Non-graphical exploratory data analysis is the first step when beginning to analyze your data as part of the general data analysis approach.

- **Measures of central tendency**

i.e. the mean, the media and mode

- **Measures of spread,**

i.e. variability, variants and standard deviation, the shape of the distribution, and the existence of outliers.

➢ Conclusion

Analysis of the data set provides

- How the data set are distributed
- Correlation between different fields and how they are related
- Normalized loss of the manufacturer
- Mileage : Mileage based on City and Highway driving for various make and attributes
- Price : Factors affecting Price of the Automobile.
- Importance of drive wheels and curb weight

symboling	normalized	make	aspiration	num-of-do	body-style	drive-whe	engine-loc	wheel-bas	length
3	122	alfa-romer	std	two	convertible	rwd	front	88.6	0.811148
3	122	alfa-romer	std	two	convertible	rwd	front	88.6	0.811148
1	122	alfa-romer	std	two	hatchback	rwd	front	94.5	0.822681
2	164	audi	std	four	sedan	fwd	front	99.8	0.84863
2	164	audi	std	four	sedan	4wd	front	99.4	0.84863
2	122	audi	std	two	sedan	fwd	front	99.8	0.851994
1	158	audi	std	four	sedan	fwd	front	105.8	0.925997
1	122	audi	std	four	wagon	fwd	front	105.8	0.925997
1	158	audi	turbo	four	sedan	fwd	front	105.8	0.925997
2	192	bmw	std	two	sedan	rwd	front	101.2	0.849592
0	192	bmw	std	four	sedan	rwd	front	101.2	0.849592
0	188	bmw	std	two	sedan	rwd	front	101.2	0.849592
0	188	bmw	std	four	sedan	rwd	front	101.2	0.849592
1	122	bmw	std	four	sedan	rwd	front	103.5	0.908217
0	122	bmw	std	four	sedan	rwd	front	103.5	0.908217
0	122	bmw	std	two	sedan	rwd	front	103.5	0.931283
0	122	bmw	std	four	sedan	rwd	front	110	0.94666
2	121	chevrolet	std	two	hatchback	fwd	front	88.4	0.678039
1	98	chevrolet	std	two	hatchback	fwd	front	94.5	0.749159
0	81	chevrolet	std	four	sedan	fwd	front	94.5	0.763095
1	118	dodge	std	two	hatchback	fwd	front	93.7	0.755887
1	118	dodge	std	two	hatchback	fwd	front	93.7	0.755887
1	118	dodge	turbo	two	hatchback	fwd	front	93.7	0.755887
1	148	dodge	std	four	hatchback	fwd	front	93.7	0.755887
1	148	dodge	std	four	sedan	fwd	front	93.7	0.755887
1	148	dodge	std	four	sedan	fwd	front	93.7	0.755887
1	148	dodge	turbo	four	sedan	fwd	front	93.7	0.755887
-1	110	dodge	std	four	wagon	fwd	front	103.3	0.83902
3	145	dodge	turbo	two	hatchback	fwd	front	95.9	0.832292
2	137	honda	std	two	hatchback	fwd	front	86.6	0.694858
2	137	honda	std	two	hatchback	fwd	front	86.6	0.694858
1	101	honda	std	two	hatchback	fwd	front	93.7	0.720807
1	101	honda	std	two	hatchback	fwd	front	93.7	0.720807
1	101	honda	std	two	hatchback	fwd	front	93.7	0.720807
0	110	honda	std	four	sedan	fwd	front	96.5	0.785199
0	78	honda	std	four	wagon	fwd	front	96.5	0.754926
0	106	honda	std	two	hatchback	fwd	front	96.5	0.804901
0	106	honda	std	two	hatchback	fwd	front	96.5	0.804901
0	85	honda	std	four	sedan	fwd	front	96.5	0.842864
0	85	honda	std	four	sedan	fwd	front	96.5	0.842864
0	85	honda	std	four	sedan	fwd	front	96.5	0.842864
1	107	honda	std	two	sedan	fwd	front	96.5	0.81259
0	122	isuzu	std	four	sedan	rwd	front	94.3	0.820279
2	122	isuzu	std	two	hatchback	rwd	front	96	0.829409
0	145	jaguar	std	four	sedan	rwd	front	113	0.959154
0	122	jaguar	std	four	sedan	rwd	front	113	0.959154
0	122	jaguar	std	two	sedan	rwd	front	102	0.921192
1	104	mazda	std	two	hatchback	fwd	front	93.1	0.764536
1	104	mazda	std	two	hatchback	fwd	front	93.1	0.764536

width	height	curb-weigl	engine-typ	num-of-cy	engine-siz	fuel-syster	bore	stroke	compressi
0.890278	48.8	2548	dohc	four	130	mpfi	3.47	2.68	9
0.890278	48.8	2548	dohc	four	130	mpfi	3.47	2.68	9
0.909722	52.4	2823	ohcv	six	152	mpfi	2.68	3.47	9
0.919444	54.3	2337	ohc	four	109	mpfi	3.19	3.4	10
0.922222	54.3	2824	ohc	five	136	mpfi	3.19	3.4	8
0.920833	53.1	2507	ohc	five	136	mpfi	3.19	3.4	8.5
0.991667	55.7	2844	ohc	five	136	mpfi	3.19	3.4	8.5
0.991667	55.7	2954	ohc	five	136	mpfi	3.19	3.4	8.5
0.991667	55.9	3086	ohc	five	131	mpfi	3.13	3.4	8.3
0.9	54.3	2395	ohc	four	108	mpfi	3.5	2.8	8.8
0.9	54.3	2395	ohc	four	108	mpfi	3.5	2.8	8.8
0.9	54.3	2710	ohc	six	164	mpfi	3.31	3.19	9
0.9	54.3	2765	ohc	six	164	mpfi	3.31	3.19	9
0.929167	55.7	3055	ohc	six	164	mpfi	3.31	3.19	9
0.929167	55.7	3230	ohc	six	209	mpfi	3.62	3.39	8
0.943056	53.7	3380	ohc	six	209	mpfi	3.62	3.39	8
0.984722	56.3	3505	ohc	six	209	mpfi	3.62	3.39	8
0.8375	53.2	1488	l	three	61	2bbl	2.91	3.03	9.5
0.883333	52	1874	ohc	four	90	2bbl	3.03	3.11	9.6
0.883333	52	1909	ohc	four	90	2bbl	3.03	3.11	9.6
0.886111	50.8	1876	ohc	four	90	2bbl	2.97	3.23	9.41
0.886111	50.8	1876	ohc	four	90	2bbl	2.97	3.23	9.4
0.886111	50.8	2128	ohc	four	98	mpfi	3.03	3.39	7.6
0.886111	50.6	1967	ohc	four	90	2bbl	2.97	3.23	9.4
0.886111	50.6	1989	ohc	four	90	2bbl	2.97	3.23	9.4
0.886111	50.6	1989	ohc	four	90	2bbl	2.97	3.23	9.4
0.886111	50.6	2191	ohc	four	98	mpfi	3.03	3.39	7.6
0.897222	59.8	2535	ohc	four	122	2bbl	3.34	3.46	8.5
0.920833	50.2	2811	ohc	four	156	mfi	3.6	3.9	7
0.8875	50.8	1713	ohc	four	92	1bbl	2.91	3.41	9.6
0.8875	50.8	1819	ohc	four	92	1bbl	2.91	3.41	9.2
0.888889	52.6	1837	ohc	four	79	1bbl	2.91	3.07	10.1
0.888889	52.6	1940	ohc	four	92	1bbl	2.91	3.41	9.2
0.888889	52.6	1956	ohc	four	92	1bbl	2.91	3.41	9.2
0.888889	54.5	2010	ohc	four	92	1bbl	2.91	3.41	9.2
0.8875	58.3	2024	ohc	four	92	1bbl	2.92	3.41	9.2
0.905556	53.3	2236	ohc	four	110	1bbl	3.15	3.58	9
0.905556	53.3	2289	ohc	four	110	1bbl	3.15	3.58	9
0.905556	54.1	2304	ohc	four	110	1bbl	3.15	3.58	9
0.868056	54.1	2372	ohc	four	110	1bbl	3.15	3.58	9
0.905556	54.1	2465	ohc	four	110	mpfi	3.15	3.58	9
0.916667	51	2293	ohc	four	110	2bbl	3.15	3.58	9.1
0.858333	53.5	2337	ohc	four	111	2bbl	3.31	3.23	8.5
0.905556	51.4	2734	ohc	four	119	spfi	3.43	3.23	9.2
0.966667	52.8	4066	dohc	six	258	mpfi	3.63	4.17	8.1
0.966667	52.8	4066	dohc	six	258	mpfi	3.63	4.17	8.1
0.980556	47.8	3950	ohcv	twelve	326	mpfi	3.54	2.76	11.5
0.891667	54.1	1890	ohc	four	91	2bbl	3.03	3.15	9
0.891667	54.1	1900	ohc	four	91	2bbl	3.03	3.15	9

horsepow	peak-rpm	city-mpg	highway-m	price	city-L/100l	horsepow	diesel	gas
111	5000	21	27	13495	11.19048	Medium	0	1
111	5000	21	27	16500	11.19048	Medium	0	1
154	5000	19	26	16500	12.36842	Medium	0	1
102	5500	24	30	13950	9.791667	Medium	0	1
115	5500	18	22	17450	13.05556	Medium	0	1
110	5500	19	25	15250	12.36842	Medium	0	1
110	5500	19	25	17710	12.36842	Medium	0	1
110	5500	19	25	18920	12.36842	Medium	0	1
140	5500	17	20	23875	13.82353	Medium	0	1
101	5800	23	29	16430	10.21739	Low	0	1
101	5800	23	29	16925	10.21739	Low	0	1
121	4250	21	28	20970	11.19048	Medium	0	1
121	4250	21	28	21105	11.19048	Medium	0	1
121	4250	20	25	24565	11.75	Medium	0	1
182	5400	16	22	30760	14.6875	High	0	1
182	5400	16	22	41315	14.6875	High	0	1
182	5400	15	20	36880	15.66667	High	0	1
48	5100	47	53	5151	5	Low	0	1
70	5400	38	43	6295	6.184211	Low	0	1
70	5400	38	43	6575	6.184211	Low	0	1
68	5500	37	41	5572	6.351351	Low	0	1
68	5500	31	38	6377	7.580645	Low	0	1
102	5500	24	30	7957	9.791667	Medium	0	1
68	5500	31	38	6229	7.580645	Low	0	1
68	5500	31	38	6692	7.580645	Low	0	1
68	5500	31	38	7609	7.580645	Low	0	1
102	5500	24	30	8558	9.791667	Medium	0	1
88	5000	24	30	8921	9.791667	Low	0	1
145	5000	19	24	12964	12.36842	Medium	0	1
58	4800	49	54	6479	4.795918	Low	0	1
76	6000	31	38	6855	7.580645	Low	0	1
60	5500	38	42	5399	6.184211	Low	0	1
76	6000	30	34	6529	7.833333	Low	0	1
76	6000	30	34	7129	7.833333	Low	0	1
76	6000	30	34	7295	7.833333	Low	0	1
76	6000	30	34	7295	7.833333	Low	0	1
86	5800	27	33	7895	8.703704	Low	0	1
86	5800	27	33	9095	8.703704	Low	0	1
86	5800	27	33	8845	8.703704	Low	0	1
86	5800	27	33	10295	8.703704	Low	0	1
101	5800	24	28	12945	9.791667	Low	0	1
100	5500	25	31	10345	9.4	Low	0	1
78	4800	24	29	6785	9.791667	Low	0	1
90	5000	24	29	11048	9.791667	Low	0	1
176	4750	15	19	32250	15.66667	High	0	1
176	4750	15	19	35550	15.66667	High	0	1
262	5000	13	17	36000	18.07692		0	1
68	5000	30	31	5195	7.833333	Low	0	1
68	5000	31	38	6095	7.580645	Low	0	1

exploratory-data-analysis-on-automobile-dataset

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Data Loading

```
path='https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DA0
df_automobile = pd.read_csv(path)
df_automobile.head()
```

	symboling	normalized-losses	make	aspiration	num-of-doors	body-style	drive-wheels	engine-location
0	3	122	alfa-romero	std	two	convertible	rwd	front
1	3	122	alfa-romero	std	two	convertible	rwd	front
2	1	122	alfa-romero	std	two	hatchback	rwd	front
3	2	164	audi	std	four	sedan	fwd	front
4	2	164	audi	std	four	sedan	4wd	front

5 rows × 29 columns

```
df_automobile.shape
```

```
(201, 29)
```

```
df_automobile.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 201 entries, 0 to 200
Data columns (total 29 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   symboling          201 non-null    int64
 1   normalized-losses  201 non-null    int64
 2   make               201 non-null    object
 3   aspiration         201 non-null    object
 4   num-of-doors       201 non-null    object
 5   body-style         201 non-null    object
```

```
 6   drive-wheels       201 non-null    object
 7   engine-location    201 non-null    object
 8   wheel-base         201 non-null    float64
 9   length             201 non-null    float64
 10  width              201 non-null    float64
 11  height             201 non-null    float64
 12  curb-weight        201 non-null    int64
 13  engine-type        201 non-null    object
 14  num-of-cylinders   201 non-null    object
 15  engine-size        201 non-null    int64
 16  fuel-system        201 non-null    object
 17  bore               201 non-null    float64
 18  stroke             197 non-null    float64
 19  compression-ratio  201 non-null    float64
 20  horsepower         201 non-null    float64
 21  peak-rpm           201 non-null    float64
 22  city-mpg           201 non-null    int64
 23  highway-mpg        201 non-null    int64
 24  price              201 non-null    float64
 25  city-L/100km       201 non-null    float64
 26  horsepower-binned  200 non-null    object
 27  diesel             201 non-null    int64
 28  gas                201 non-null    int64
dtypes: float64(11), int64(8), object(10)
memory usage: 45.7+ KB
```

Data Cleaning

- Data contains "?" replace it with NAN

```
df_data = df_automobile.replace('?',np.NAN)
df_data.isnull().sum()
```

```
symboling            0
normalized-losses    0
make                 0
aspiration           0
num-of-doors         0
body-style           0
drive-wheels         0
engine-location      0
wheel-base           0
length               0
width                0
height               0
curb-weight          0
engine-type          0
num-of-cylinders     0
engine-size          0
fuel-system          0
bore                 0
stroke               4
compression-ratio    0
horsepower           0
peak-rpm             0
city-mpg             0
highway-mpg          0
price                0
```

```
city-L/100km         0
horsepower-binned    1
diesel               0
gas                  0
dtype: int64
```

```
df_automobile['stroke'].fillna('np.nan',  inplace= True)
```

```
df_automobile['horsepower-binned'].fillna('np.nan', inplace= True)
```

```
df_automobile.isnull().sum()
```

```
symboling            0
normalized-losses    0
make                 0
aspiration           0
num-of-doors         0
body-style           0
drive-wheels         0
engine-location      0
wheel-base           0
length               0
width                0
height               0
curb-weight          0
engine-type          0
num-of-cylinders     0
engine-size          0
fuel-system          0
bore                 0
stroke               0
compression-ratio    0
horsepower           0
peak-rpm             0
city-mpg             0
highway-mpg          0
price                0
city-L/100km         0
horsepower-binned    0
diesel               0
gas                  0
dtype: int64
```

```
df_automobile.head(10)
```

	symboling	normalized-losses	make	aspiration	num-of-doors	body-style	drive-wheels	engine-location
0	3	122	alfa-romero	std	two	convertible	rwd	front
1	3	122	alfa-romero	std	two	convertible	rwd	front
2	1	122	alfa-romero	std	two	hatchback	rwd	front
3	2	164	audi	std	four	sedan	fwd	front
4	2	164	audi	std	four	sedan	4wd	front
5	2	122	audi	std	two	sedan	fwd	front

Summary statistics of variable

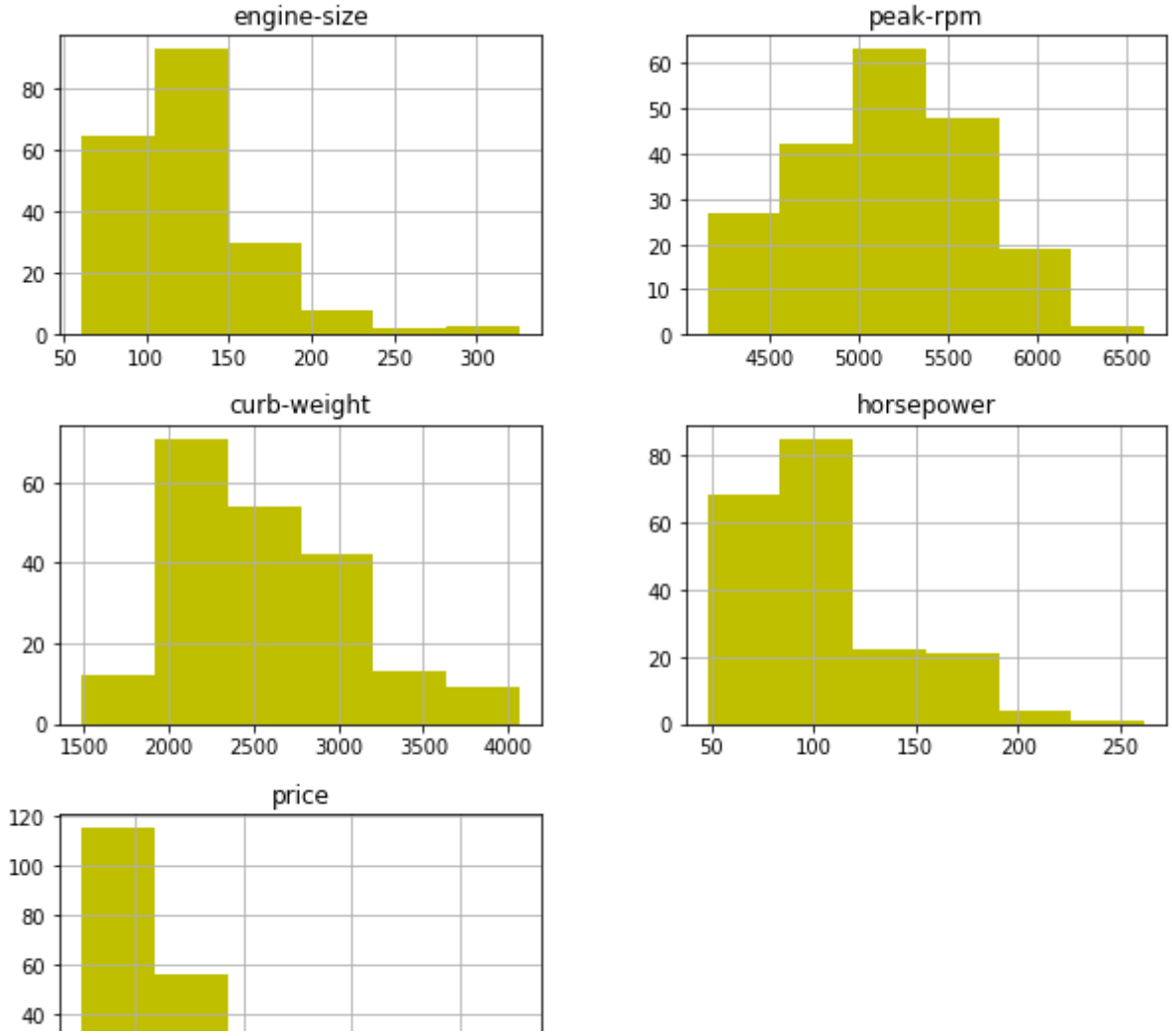
```
df_automobile.describe()
```

	symboling	normalized-losses	wheel-base	length	width	height	
count	201.000000	201.00000	201.000000	201.000000	201.000000	201.000000	201
mean	0.840796	122.00000	98.797015	0.837102	0.915126	53.766667	2555
std	1.254802	31.99625	6.066366	0.059213	0.029187	2.447822	517
min	-2.000000	65.00000	86.600000	0.678039	0.837500	47.800000	1488
25%	0.000000	101.00000	94.500000	0.801538	0.890278	52.000000	2169
50%	1.000000	122.00000	97.000000	0.832292	0.909722	54.100000	2414
75%	2.000000	137.00000	102.400000	0.881788	0.925000	55.500000	2926
max	3.000000	256.00000	120.900000	1.000000	1.000000	59.800000	4066

Univariate Analysis

```
plt.figure(figsize=(10,8))
df_automobile[['engine-size','peak-rpm','curb-weight','horsepower','price']].hist(figsize=
plt.figure(figsize=(10,8))
plt.tight_layout()
plt.show()
```

```
<Figure size 720x576 with 0 Axes>
```



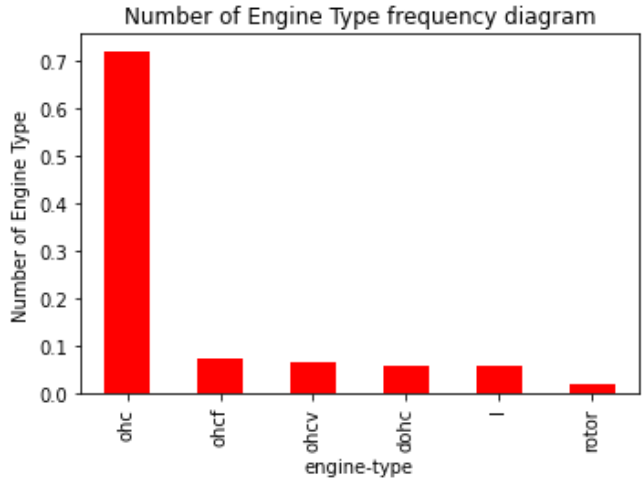
Findings

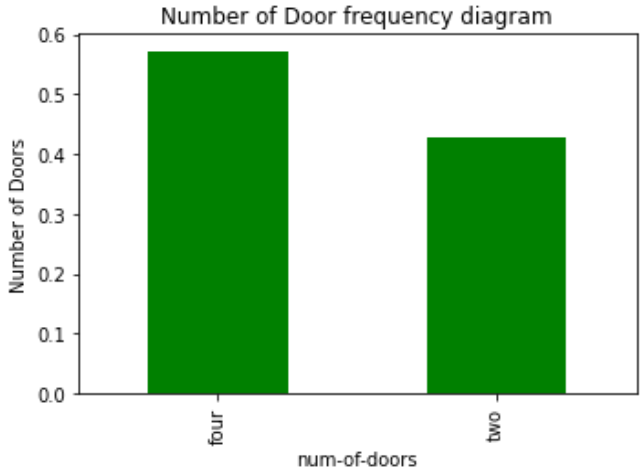
- Most of the car has a Curb Weight is in range 1900 to 3100
- The Engine Size is inrange 60 to 190
- Most vehicle has horsepower 50 to 125
- Most Vehicle are in price range 5000 to 18000
- peak rpm is mostly distributed between 4600 to 5700

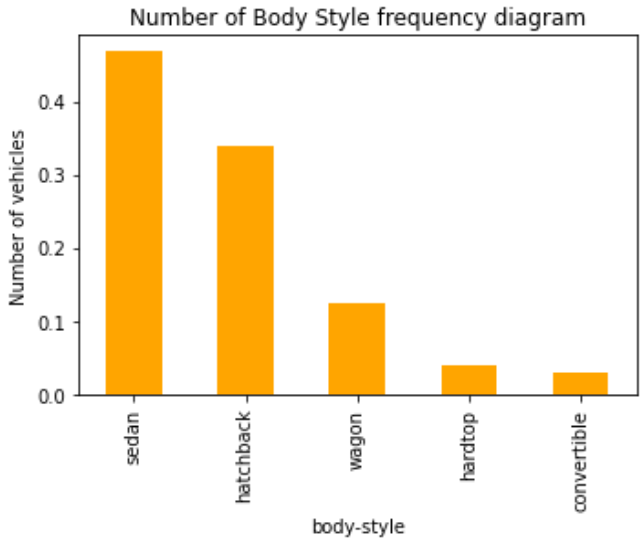
```
plt.figure(1)
plt.subplot(221)
df_automobile['engine-type'].value_counts(normalize=True).plot(figsize=(10,8),kind='bar',c
plt.title("Number of Engine Type frequency diagram")
plt.ylabel('Number of Engine Type')
plt.xlabel('engine-type');
```

```
plt.subplot(222)
df_automobile['num-of-doors'].value_counts(normalize=True).plot(figsize=(10,8),kind='bar',
plt.title("Number of Door frequency diagram")
plt.ylabel('Number of Doors')
plt.xlabel('num-of-doors');
```

```
plt.subplot(224)
df_automobile['body-style'].value_counts(normalize=True).plot(figsize=(10,8),kind='bar',co
plt.title("Number of Body Style frequency diagram")
plt.ylabel('Number of vehicles')
plt.xlabel('body-style');
plt.tight_layout()
plt.show()
```



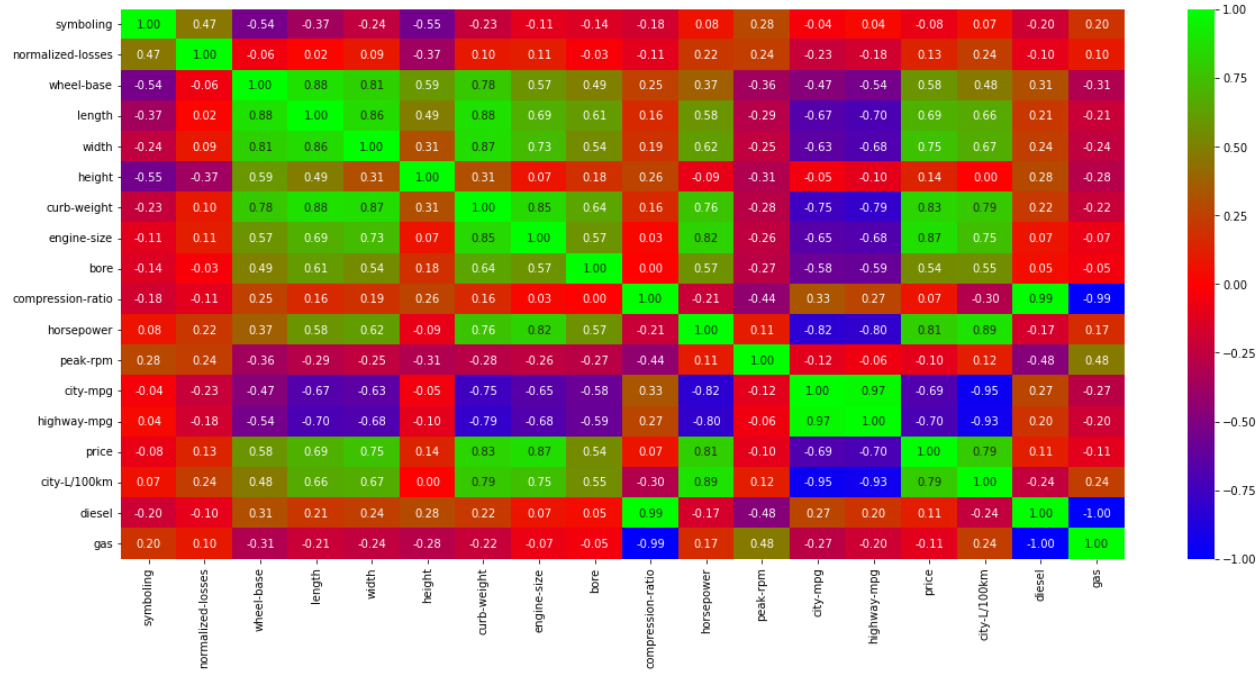




Findings

- More than 70 % of the vehicle has Ohc type of Engine
- 57% of the cars has 4 doors
- Most produced vehicle are of body style sedan around 48% followed by hatchback 32%

```
import seaborn as sns
corr = df_automobile.corr()
plt.figure(figsize=(20,9))
a = sns.heatmap(corr,cmap='brg', annot=True, fmt='.2f')
```



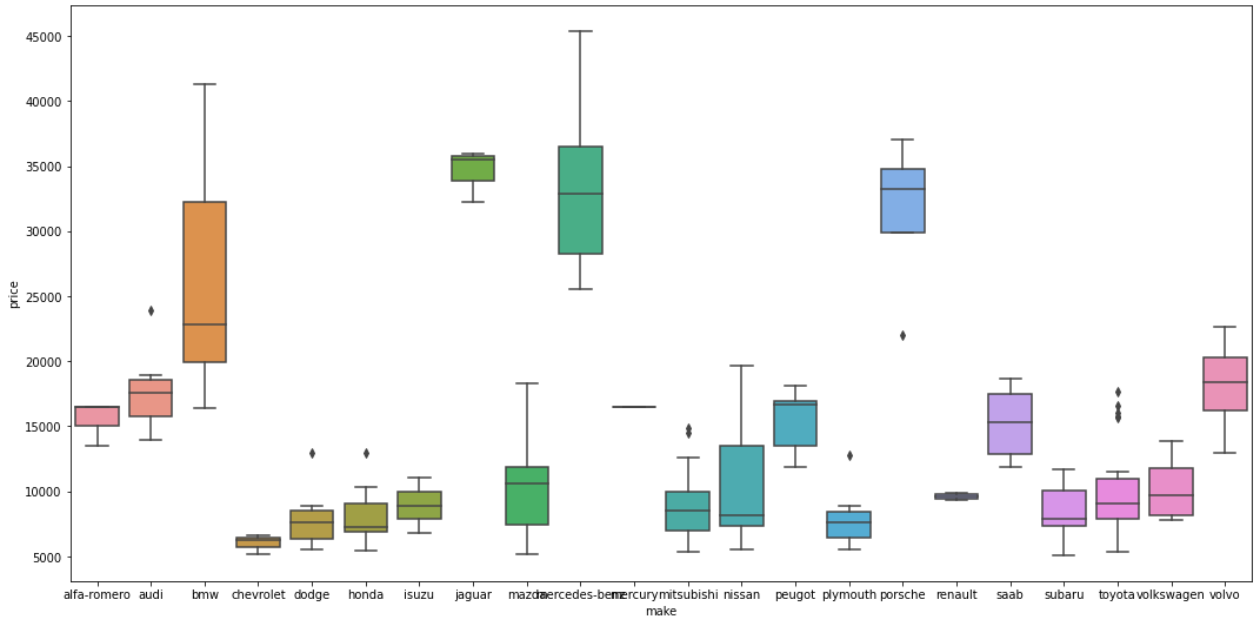
Findings

- curb-size, engine-size, horsepower are positively corelated
- city-mpg,highway-mpg are negatively corelated

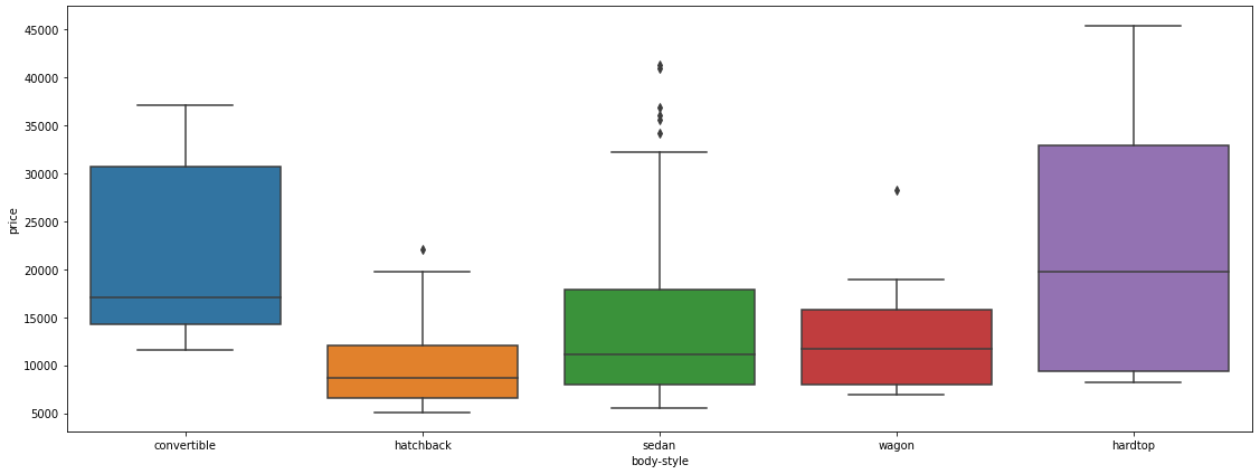
Bivariate Analysis

> Price Analysis

```
plt.rcParams['figure.figsize']=(18,9)
ax = sns.boxplot(x="make", y="price", data=df_automobile)
```

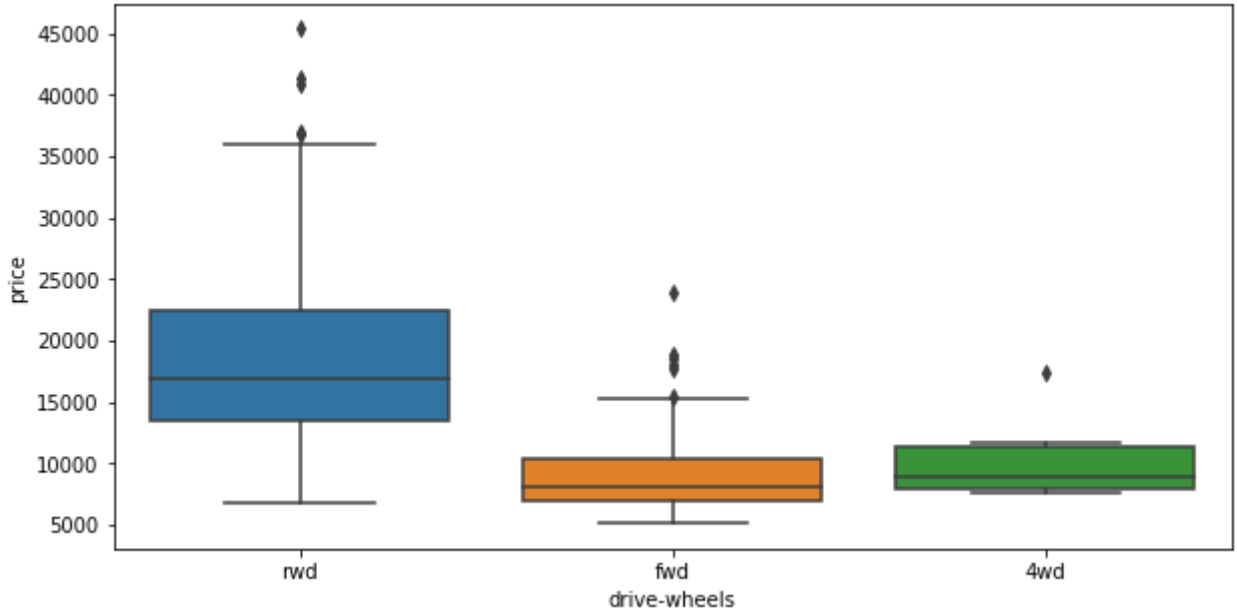



```
plt.rcParams['figure.figsize']=(19,7)
ax = sns.boxplot(x="body-style", y="price", data=df_automobile)
```



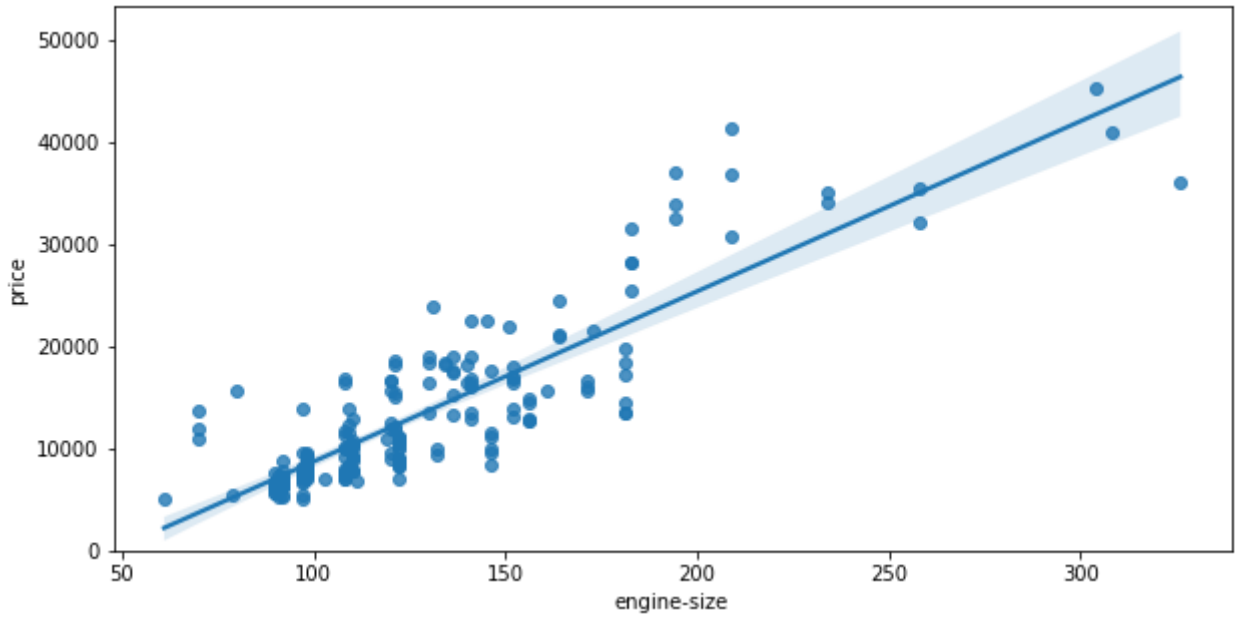
Positive linear relationship

```
plt.rcParams['figure.figsize']=(10,5)
ax = sns.boxplot(x="drive-wheels", y="price", data=df_automobile)
```




```
# Engine size as potential predictor variable of price
sns.regplot(x="engine-size", y="price", data=df_automobile)
plt.ylim(0,)
```

```
(0.0, 53379.500274433274)
```

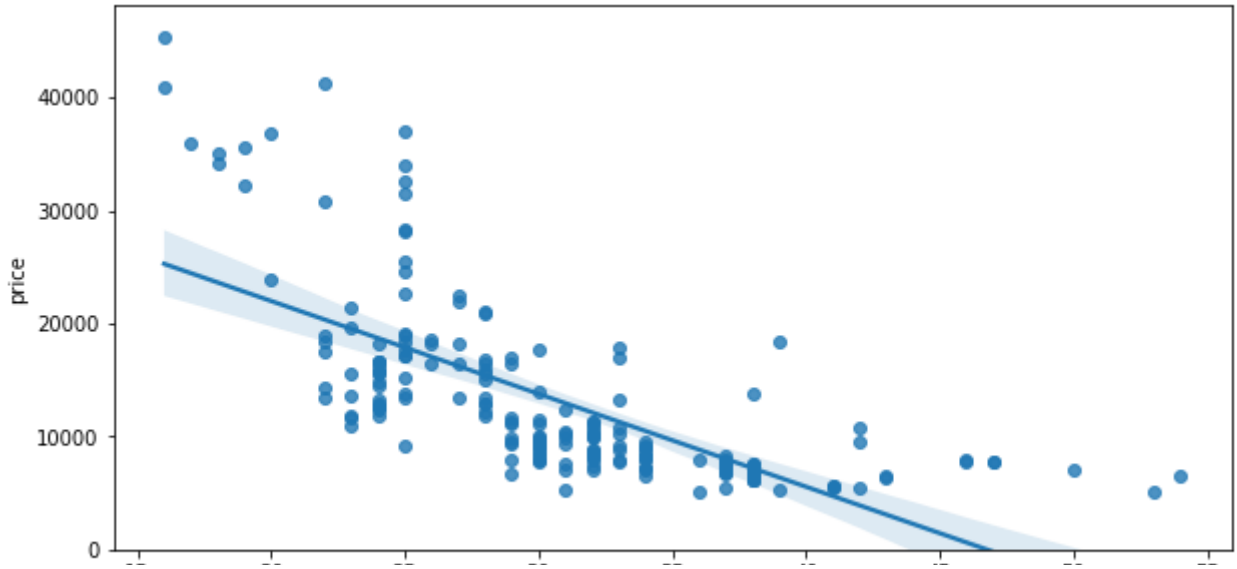



```
df_automobile[["engine-size", "price"]].corr()
```

	engine-size	price
engine-size	1.000000	0.872335
price	0.872335	1.000000

```
sns.regplot(x="highway-mpg", y="price", data=df_automobile)
plt.ylim(0,)
```

```
(0.0, 48181.81143636248)
```

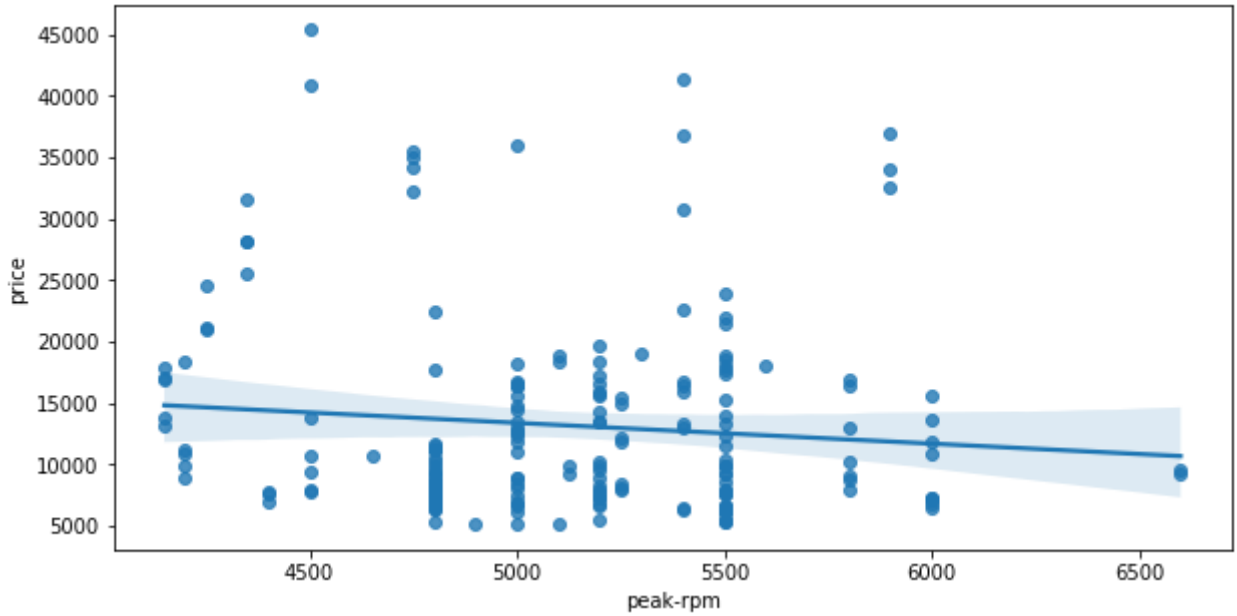



```
df_automobile[['highway-mpg', 'price']].corr()
```

	highway-mpg	price
highway-mpg	1.000000	-0.704692
price	-0.704692	1.000000

```
sns.regplot(x="peak-rpm", y="price", data=df_automobile)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe0c40154d0>
```




```
df_automobile[['peak-rpm','price']].corr()
```

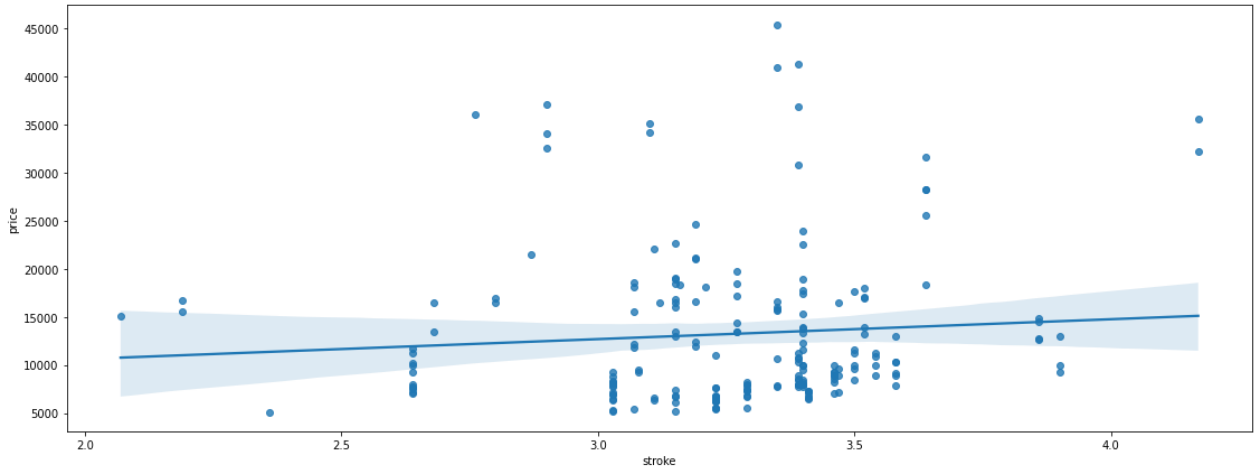
	peak-rpm	price
peak-rpm	1.000000	-0.101616
price	-0.101616	1.000000

```
df_automobile[["stroke","price"]].corr()
```

	stroke	price
stroke	1.00000	0.08231
price	0.08231	1.00000

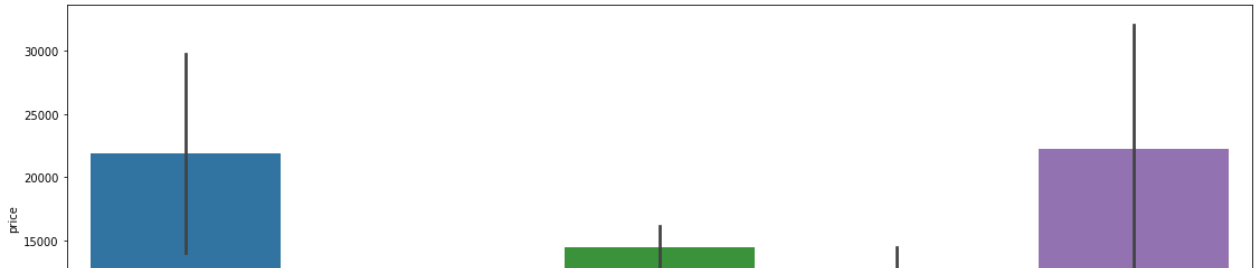
```
sns.regplot(x="stroke", y="price", data=df_automobile)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f266c730a10>
```



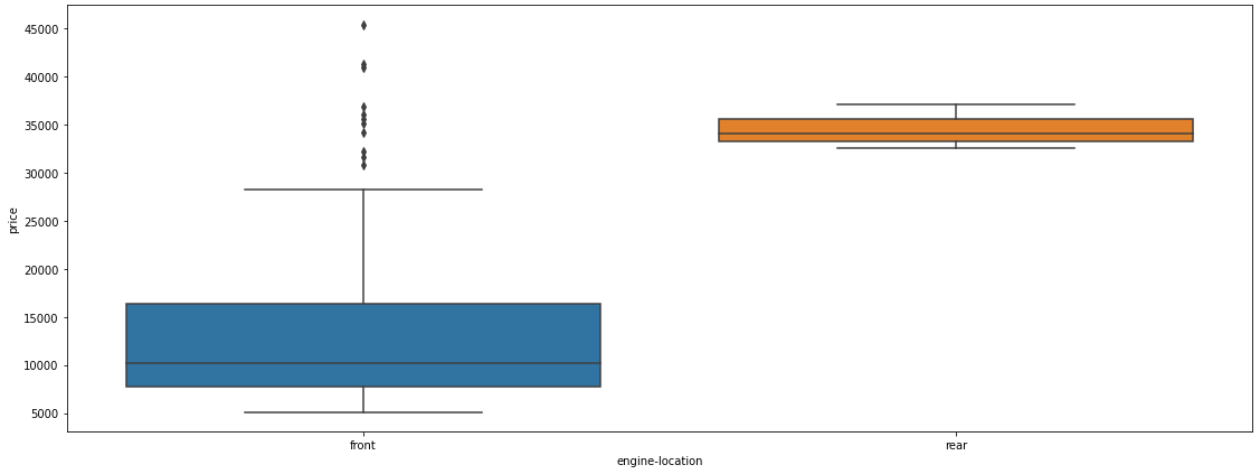

```
sns.barplot(x="body-style", y="price", data=df_automobile)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2667964ed0>
```



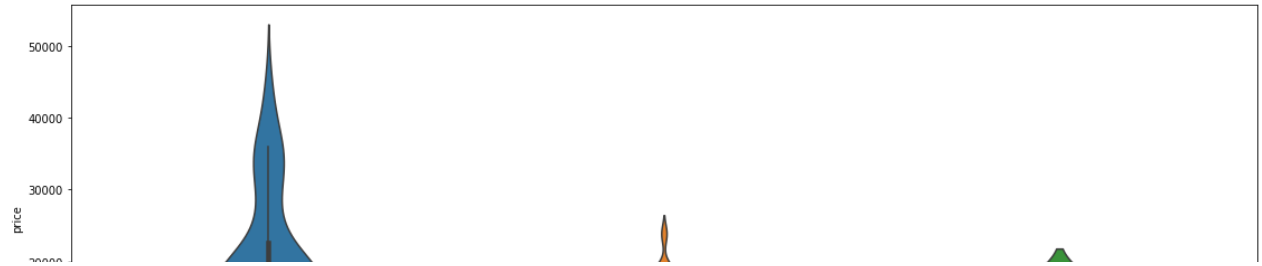

```
sns.boxplot(x="engine-location", y="price", data=df_automobile)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2667710b50>
```




```
# drive-wheels
sns.violinplot(x="drive-wheels", y="price", data=df_automobile)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2682358510>
```



Price Analysis

- engine size and curb-weight is positively co realted with price
- city-mpg is negatively corelated with price as increase horsepower reduces the mileage

Practical 2

To Perform Linear Regression study on the 3D printing dataset.

Aim: To Perform Linear Regression study on the 3D printing dataset.

Prerequisites: Automobile data, Jupyter Notebook / Google Colab

This dataset comes from research by TR/Selcuk University Mechanical Engineering department. The aim of the study is to determine how much of the adjustment parameters in 3d printers affect the print quality, accuracy and strenght. Where there are nine setting parameters and three measured output parameters.

Content

Setting Parameters:

- Layer Height (mm)
- Wall Thickness (mm)
- Infill Density (%)
- Infill Pattern ()
- Nozzle Temperature (Cº)

- Bed Temperature (C^{o})
- Print Speed (mm/s)
- Material ()
- Fan Speed (%)

Output Parameters: (Measured)

- Roughness (µm)
- Tension (ultimate) Strenght (MPa)
- Elongation (%)

In this notebook, we will perform simple linear regression analysis of the 3D printing dataset and study the various relationships existing between the target variables AKA labels and the predictor varibale AKA features.

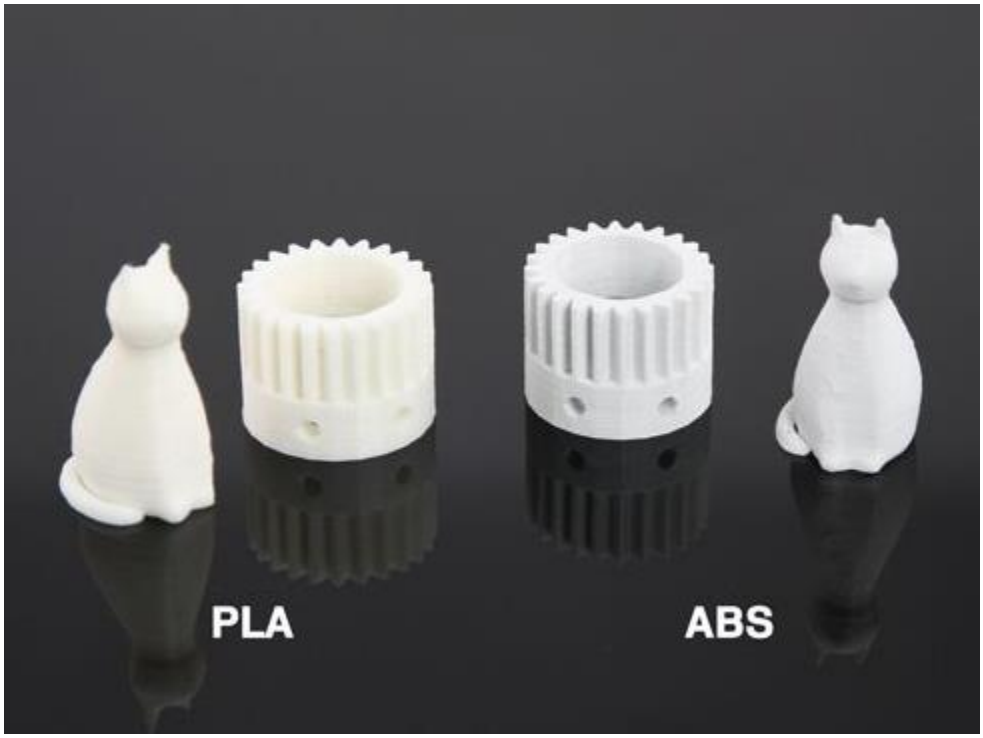
- The dataset contains **12 columns**.
- The **first 9 columns** i.e from layer_height to fan_speed are **features**.
- The **last 3 columns** i.i from roughness to elongation are **labels**. So, we will be **predicting** these **three** based on the **9 features**.
- The various **units** of the **nine features** are as follows:
- 1. Layer Height in mm
- 2. Wall Thickness in mm
- 3. Infill Density in %
- 4. Infill Pattern in either Grid or Honeycomb
- 5. Nozzle Temperature in Degree C
- 6. Bed Temperature in degree C
- 7. Print speed in mm/s
- 8. Material in either abs or pla

  9. Fan Speed in %

- The **units** of the **labels** are as follows
- 1. Roughness in micro metre
- 2. Tension Strength in MPa

  3. Elongation in %

Pictures of Infill patterns and Filament materials



Basic information

- The dataset contains **50 rows** of data.
- The columns `infill_pattern` and `material` consists of **categorical entries** (infill_pattern = grid or honeycomb and material = abs or pla) **instead** of **numerical** entries.
- In the **Machine Learning World** this is relatively a **very small dataset** interms of observations. **Still** we can **fit** a good **regression model** out of it and study them.

layer_heig	wall_thick	infill_dens	infill_patte	nozzle_ter	bed_temp	print_spee	material	fan_speed	roughness	tension_st	elongation
0.02	8	90	grid	220	60	40	abs	0	25	18	1.2
0.02	7	90	honeycom	225	65	40	abs	25	32	16	1.4
0.02	1	80	grid	230	70	40	abs	50	40	8	0.8
0.02	4	70	honeycom	240	75	40	abs	75	68	10	0.5
0.02	6	90	grid	250	80	40	abs	100	92	5	0.7
0.02	10	40	honeycom	200	60	40	pla	0	60	24	1.1
0.02	5	10	grid	205	65	40	pla	25	55	12	1.3
0.02	10	10	honeycom	210	70	40	pla	50	21	14	1.5
0.02	9	70	grid	215	75	40	pla	75	24	27	1.4
0.02	8	40	honeycom	220	80	40	pla	100	30	25	1.7
0.06	6	80	grid	220	60	60	abs	0	75	37	2.4
0.06	2	20	honeycom	225	65	60	abs	25	92	12	1.4
0.06	10	50	grid	230	70	60	abs	50	118	16	1.3
0.06	6	10	honeycom	240	75	60	abs	75	200	9	0.8
0.06	3	50	grid	250	80	60	abs	100	220	10	1
0.06	10	90	honeycom	200	60	60	pla	0	126	27	2.2
0.06	3	40	grid	205	65	60	pla	25	145	23	1.9
0.06	8	30	honeycom	210	70	60	pla	50	88	26	1.6
0.06	5	80	grid	215	75	60	pla	75	92	33	2.1
0.06	10	50	honeycom	220	80	60	pla	100	74	29	2
0.1	1	40	grid	220	60	120	abs	0	120	16	1.2
0.1	2	30	honeycom	225	65	120	abs	25	144	12	1.1
0.1	1	50	grid	230	70	120	abs	50	265	10	0.9
0.1	9	80	honeycom	240	75	120	abs	75	312	19	0.8
0.1	2	60	grid	250	80	120	abs	100	368	8	0.4
0.1	1	50	honeycom	200	60	120	pla	0	180	11	1.6
0.1	4	40	grid	205	65	120	pla	25	176	12	1.2
0.1	3	50	honeycom	210	70	120	pla	50	128	18	1.8
0.1	4	90	grid	215	75	120	pla	75	138	34	2.9
0.1	1	30	honeycom	220	80	120	pla	100	121	14	1.5
0.15	4	50	grid	220	60	60	abs	0	168	27	2.4
0.15	7	10	honeycom	225	65	60	abs	25	154	19	1.8
0.15	6	50	grid	230	70	60	abs	50	225	18	1.4
0.15	1	50	honeycom	240	75	60	abs	75	289	9	0.6
0.15	7	80	grid	250	80	60	abs	100	326	13	0.7
0.15	3	80	honeycom	200	60	60	pla	0	192	33	2.8
0.15	4	50	grid	205	65	60	pla	25	212	24	1.8
0.15	10	30	honeycom	210	70	60	pla	50	168	26	2.1
0.15	6	40	grid	215	75	60	pla	75	172	22	2.3
0.15	1	10	honeycom	220	80	60	pla	100	163	4	0.7
0.2	4	80	grid	220	60	40	abs	0	212	35	3.3
0.2	9	90	honeycom	225	65	40	abs	25	276	34	3.1
0.2	7	30	grid	230	70	40	abs	50	298	28	2.2
0.2	6	90	honeycom	240	75	40	abs	75	360	28	1.6
0.2	3	80	grid	250	80	40	abs	100	357	21	1.1
0.2	5	60	honeycom	200	60	40	pla	0	321	28	2.7
0.2	4	20	grid	205	65	40	pla	25	265	14	1.8
0.2	5	60	honeycom	210	70	40	pla	50	278	30	3.2
0.2	7	40	grid	215	75	40	pla	75	244	29	3.2
0.2	3	60	honeycom	220	80	40	pla	100	220	27	3.1

Linear Regression study on the 3D printing dataset

In this case study , we will perform simple linear regression analysis of the 3D printing dataset and study the various relationships existing between the target variables and the predictor varibale .

```
# Importing the libraries
!pip install pandas==1.2.0
!pip install xlrd==1.2.0
import pandas as pd
import·numpy·as·np
import·matplotlib.pyplot·as·plt
import·seaborn·as·sns
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting pandas==1.2.0
  Downloading pandas-1.2.0-cp37-cp37m-manylinux1_x86_64.whl (9.9 MB)
     |████████████████████████████████| 9.9 MB 4.5 MB/s
Requirement already satisfied: numpy>=1.16.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from
Installing collected packages: pandas
  Attempting uninstall: pandas
    Found existing installation: pandas 1.3.5
    Uninstalling pandas-1.3.5:
      Successfully uninstalled pandas-1.3.5
Successfully installed pandas-1.2.0
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting xlrd==1.2.0
  Downloading xlrd-1.2.0-py2.py3-none-any.whl (103 kB)
     |████████████████████████████████| 103 kB 5.3 MB/s
Installing collected packages: xlrd
  Attempting uninstall: xlrd
    Found existing installation: xlrd 1.1.0
    Uninstalling xlrd-1.1.0:
      Successfully uninstalled xlrd-1.1.0
Successfully installed xlrd-1.2.0
```

```
#To enable matplot visualization
%matplotlib inline
```

```
#Importing the Dataset using .read_csv() function of pandas
#We will call the dataset as 'printer'
from google.colab import drive
drive.mount('/content/drive',force_remount=True)
printer = pd.read_excel('/content/data.xlsx')
```

```
Mounted at /content/drive
```

```
#Let's check few rows of the dataset using the .head() function of pandas
printer.head()
```

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	be
0	0.02	8	90	grid	220	
1	0.02	7	90	honeycomb	225	
2	0.02	1	80	grid	230	
3	0.02	4	70	honeycomb	240	
4	0.02	6	90	grid	250	

```
# Basic information of the dataset using .info() function of pandas
printer.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   layer_height        50 non-null     float64
 1   wall_thickness      50 non-null     int64  
 2   infill_density      50 non-null     int64  
 3   infill_pattern      50 non-null     object 
 4   nozzle_temperature  50 non-null     int64  
 5   bed_temperature     50 non-null     int64  
 6   print_speed         50 non-null     int64  
 7   material            50 non-null     object 
 8   fan_speed           50 non-null     int64  
 9   roughness           50 non-null     int64  
 10  tension_strenght    50 non-null     int64  
 11  elongation          50 non-null     float64
dtypes: float64(2), int64(8), object(2)
memory usage: 4.8+ KB
```

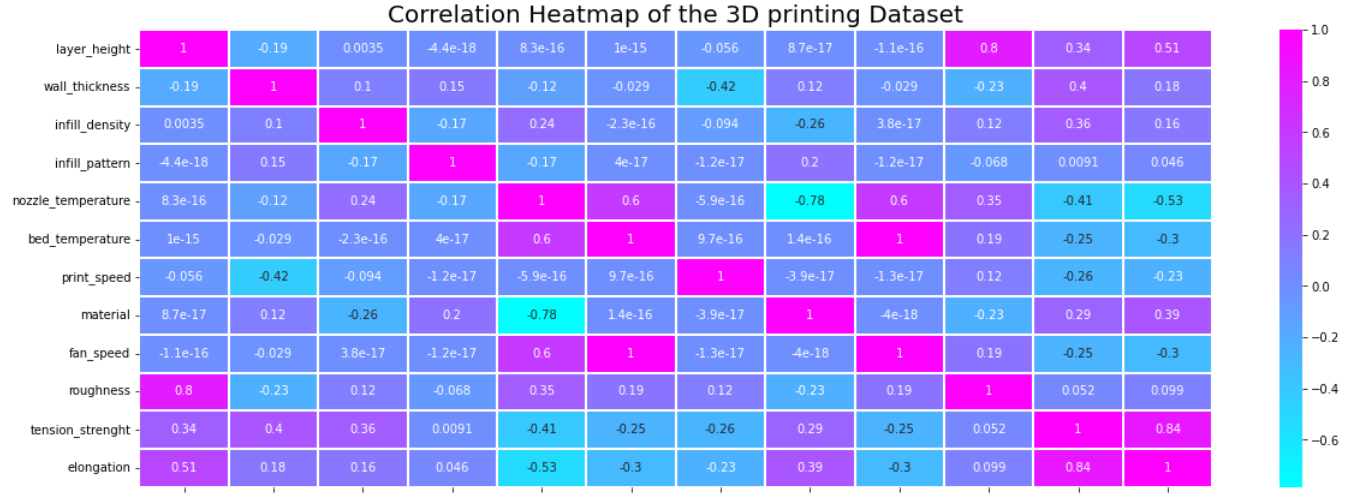
2. Data Visualization

```
printer['infill_pattern'].replace(['grid','honeycomb'], [0,1], inplace = True)
printer['material'].replace(['abs','pla'], [0,1], inplace = True)
```

```
#let's view the first 10 observations
printer.head(10)
```

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	be
0	0.02	8	90	0	220	
1	0.02	7	90	1	225	
2	0.02	1	80	0	230	
3	0.02	4	70	1	240	
4	0.02	6	90	0	250	
5	0.02	10	40	1	200	
6	0.02	5	10	0	205	
7	0.02	10	10	1	210	
8	0.02	9	70	0	215	
9	0.02	8	40	1	220	

```
# Creating the heatmap
# First create a space for the heatmap and then draw the heatmap inside the space

fig, ax = plt.subplots(figsize = (20,7))

#Title for the heatmap
title = 'Correlation Heatmap of the 3D printing Dataset'
plt.title(title, fontsize = 20)
ttl = ax.title

# Correlation heatmap using .heatmap() function of sns library
sns.heatmap(printer.corr(), cbar = True, cmap = 'cool', annot = True, linewidths = 1, ax = ax

#enable visualization using .show() function of matplot
plt.show()
```



Model 1 : Predicting roughness based on 9 features.

```
# Defining features and labels
X = printer.drop(['roughness','tension_strenght','elongation'], axis = 1)
y = printer['roughness']


X.head()
```

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	be
0	0.02	8	90	0	220	
1	0.02	7	90	1	225	
2	0.02	1	80	0	230	
3	0.02	4	70	1	240	
4	0.02	6	90	0	250	

```
#importing statsmodels library
import statsmodels.api as sm


# let's define a function for the multiple regression

def linear_Regression(x,y):

    x = sm.add_constant(x)

    #defining the model, fitting the model and printing the results
    multiple_model = sm.OLS(y,x).fit()
    print(multiple_model.summary())


#calling the linear regression function
```

```
linear_Regression(X,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              roughness   R-squared:                       0.875
Model:                            OLS   Adj. R-squared:                  0.851
Method:                 Least Squares   F-statistic:                     35.95
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           3.83e-16
Time:                        09:18:06   Log-Likelihood:                -248.19
No. Observations:                  50   AIC:                             514.4
Df Residuals:                      41   BIC:                             531.6
Df Model:                           8
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
const                  -0.9534      0.159     -6.006      0.000      -1.274      -0.633
layer_height         1269.4449     87.648     14.483      0.000    1092.437    1446.453
wall_thickness          2.3342      2.189      1.066      0.293      -2.087       6.756
infill_density         -0.0423      0.234     -0.181      0.857      -0.515       0.430
infill_pattern         -0.1255     11.280     -0.011      0.991     -22.907      22.656
nozzle_temperature     15.0562      2.529      5.953      0.000       9.948      20.164
bed_temperature       -55.6225      9.279     -5.995      0.000     -74.362     -36.883
print_speed             0.6496      0.206      3.153      0.003       0.233       1.066
material              298.4514     58.364      5.114      0.000     180.582     416.321
fan_speed               7.8989      1.238      6.379      0.000       5.398      10.399
==============================================================================
Omnibus:                        1.107   Durbin-Watson:                   1.467
Prob(Omnibus):                  0.575   Jarque-Bera (JB):                0.749
Skew:                           0.300   Prob(JB):                        0.688
Kurtosis:                       3.018   Cond. No.                     3.59e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 2.48e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

```
#check X
X.head()
```

```
X = X.drop(['wall_thickness','infill_density','infill_pattern'], axis = 1)
```

0	0.02	8	90	0		220

```
X.head()
```

	layer_height	nozzle_temperature	bed_temperature	print_speed	material	fan_speed
0	0.02	220	60	40	0	0
1	0.02	225	65	40	0	25
2	0.02	230	70	40	0	50
3	0.02	240	75	40	0	75
4	0.02	250	80	40	0	100

```
#calling the linear regression function
linear_Regression(X,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              roughness   R-squared:                       0.872
Model:                            OLS   Adj. R-squared:                  0.857
Method:                 Least Squares   F-statistic:                     59.78
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           1.67e-18
Time:                        09:19:12   Log-Likelihood:                -248.88
No. Observations:                  50   AIC:                             509.8
Df Residuals:                      44   BIC:                             521.2
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const                 -0.9307      0.151     -6.172      0.000      -1.235      -0.627
layer_height        1246.5353     83.178     14.986      0.000    1078.901    1414.169
nozzle_temperature    14.7774      2.398      6.163      0.000       9.945      19.610
bed_temperature      -54.3045      8.815     -6.160      0.000     -72.070     -36.539
print_speed            0.5538      0.180      3.070      0.004       0.190       0.917
material             294.1610     56.159      5.238      0.000     180.981     407.341
fan_speed              7.6993      1.177      6.542      0.000       5.328      10.071
==============================================================================
Omnibus:                        0.850   Durbin-Watson:                   1.367
Prob(Omnibus):                  0.654   Jarque-Bera (JB):                0.720
Skew:                           0.285   Prob(JB):                        0.698
Kurtosis:                       2.857   Cond. No.                     3.59e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 2.37e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

```
X.head()
```

	layer_height	nozzle_temperature	bed_temperature	print_speed	material	fan_speed
0	0.02	220	60	40	0	0
1	0.02	225	65	40	0	25
2	0.02	230	70	40	0	50
3	0.02	240	75	40	0	75
4	0.02	250	80	40	0	100

```
X.head()
```

	layer_height	nozzle_temperature	bed_temperature	print_speed	material	fan_speed
0	0.02	220	60	40	0	0
1	0.02	225	65	40	0	25
2	0.02	230	70	40	0	50
3	0.02	240	75	40	0	75
4	0.02	250	80	40	0	100

```
#get the interaction terms by multiplying values

inter_mn = X['material']*X['nozzle_temperature']
inter_bn = X['bed_temperature']*X['nozzle_temperature']
inter_fn = X['fan_speed']*X['nozzle_temperature']
inter_fb = X['fan_speed']*X['bed_temperature']


#adding these interaction terms to dataset using .concat() function of pandas
#we will call this dataset as interaction

interaction = pd.concat([X,inter_mn,inter_bn,inter_fn,inter_fb], axis = 1)

#chenge column names of this interaction terms
interaction = interaction.rename(columns = {0:'interct_mn', 1:'interact_bn', 2:'interact_fn',
                            3:'interact_fb'})

interaction.head(10)
```

	layer_height	nozzle_temperature	bed_temperature	print_speed	material	fan_speed
0	0.02	220	60	40	0	0
1	0.02	225	65	40	0	25
2	0.02	230	70	40	0	50
3	0.02	240	75	40	0	75
4	0.02	250	80	40	0	100
5	0.02	200	60	40	1	0
6	0.02	205	65	40	1	25
7	0.02	210	70	40	1	50
8	0.02	215	75	40	1	75
9	0.02	220	80	40	1	100

```
# NOw let's fit this model to the linear regression function

linear_Regression(interaction,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:              roughness   R-squared:                       0.925
Model:                            OLS   Adj. R-squared:                  0.910
Method:                 Least Squares   F-statistic:                     63.22
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           1.31e-20
Time:                        09:20:49   Log-Likelihood:                -235.45
No. Observations:                  50   AIC:                             488.9
Df Residuals:                      41   BIC:                             506.1
Df Model:                           8
Covariance Type:            nonrobust
==========================================================================================
                             coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                    -1.0735      0.923     -1.163      0.252      -2.938       0.791
layer_height           1246.5353     65.870     18.924      0.000    1113.508    1379.562
nozzle_temperature        0.0050      0.004      1.146      0.259      -0.004       0.014
bed_temperature         -57.8032     50.004     -1.156      0.254    -158.788      43.182
print_speed               0.5538      0.143      3.877      0.000       0.265       0.842
material               5516.1791   2836.453      1.945      0.059    -212.153    1.12e+04
fan_speed                33.0427     27.526      1.200      0.237     -22.546      88.632
interct_mn              -25.7036     12.956     -1.984      0.054     -51.869       0.462
interact_bn               0.2588      0.227      1.138      0.262      -0.201       0.718
interact_fn              -0.2032      0.174     -1.169      0.249      -0.554       0.148
interact_fb               0.1662      0.138      1.206      0.235      -0.112       0.444
==============================================================================
Omnibus:                        0.934   Durbin-Watson:                   1.345
Prob(Omnibus):                  0.627   Jarque-Bera (JB):                0.333
Skew:                           0.132   Prob(JB):                        0.847
Kurtosis:                       3.300   Cond. No.                     4.80e+20
==============================================================================
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 9.58e-32. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

Model 2: Predicting Tension strength based on features

```
X = printer.drop(['roughness','tension_strenght','elongation'], axis = 1)
X.head()
```

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	be
0	0.02	8	90	0	220	
1	0.02	7	90	1	225	
2	0.02	1	80	0	230	
3	0.02	4	70	1	240	
4	0.02	6	90	0	250	

```
y = printer['tension_strenght']
y
```

```
0     18
1     16
2      8
3     10
4      5
5     24
6     12
7     14
8     27
9     25
10    37
11    12
12    16
13     9
14    10
15    27
16    23
17    26
18    33
19    29
20    16
21    12
22    10
23    19
```

```
24     8
25    11
26    12
27    18
28    34
29    14
30    27
31    19
32    18
33     9
34    13
35    33
36    24
37    26
38    22
39     4
40    35
41    34
42    28
43    28
44    21
45    28
46    14
47    30
48    29
49    27
Name: tension_strenght, dtype: int64
```

```
linear_Regression(X,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       tension_strenght   R-squared:                       0.673
Model:                            OLS   Adj. R-squared:                  0.609
Method:                 Least Squares   F-statistic:                     10.55
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           6.91e-08
Time:                        09:22:30   Log-Likelihood:                -151.94
No. Observations:                  50   AIC:                             321.9
Df Residuals:                      41   BIC:                             339.1
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const                  0.0663      0.023      2.863      0.007       0.020       0.113
layer_height          55.5972     12.788      4.348      0.000      29.772      81.423
wall_thickness         1.0687      0.319      3.346      0.002       0.424       1.714
infill_density         0.1629      0.034      4.769      0.000       0.094       0.232
infill_pattern        -1.1427      1.646     -0.694      0.491      -4.466       2.181
nozzle_temperature    -1.0468      0.369     -2.837      0.007      -1.792      -0.302
bed_temperature        3.8647      1.354      2.855      0.007       1.131       6.599
print_speed           -0.0156      0.030     -0.519      0.607      -0.076       0.045
material             -17.3051      8.515     -2.032      0.049     -34.502      -0.108
fan_speed             -0.5719      0.181     -3.166      0.003      -0.937      -0.207
==============================================================================
```

```
Omnibus:                        0.265   Durbin-Watson:                   1.461
Prob(Omnibus):                  0.876   Jarque-Bera (JB):                0.456
Skew:                           0.060   Prob(JB):                        0.796
Kurtosis:                       2.548   Cond. No.                     3.59e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 2.48e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

```
X = X.drop(['infill_pattern','print_speed','material'], axis = 1)
X.head()
```

	layer_height	wall_thickness	infill_density	nozzle_temperature	bed_temperature	f
0	0.02	8	90	220	60	
1	0.02	7	90	225	65	
2	0.02	1	80	230	70	
3	0.02	4	70	240	75	
4	0.02	6	90	250	80	

```
linear_Regression(X,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       tension_strenght   R-squared:                       0.634
Model:                            OLS   Adj. R-squared:                  0.593
Method:                 Least Squares   F-statistic:                     15.27
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           1.07e-08
Time:                        09:23:02   Log-Likelihood:                -154.73
No. Observations:                  50   AIC:                             321.5
Df Residuals:                      44   BIC:                             332.9
Df Model:                           5
Covariance Type:            nonrobust
==========================================================================================
                             coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                     0.0191      0.004      4.358      0.000       0.010       0.028
layer_height             56.6997     12.886      4.400      0.000      30.730      82.669
wall_thickness            1.1478      0.290      3.965      0.000       0.564       1.731
infill_density            0.1543      0.034      4.537      0.000       0.086       0.223
nozzle_temperature       -0.3028      0.073     -4.141      0.000      -0.450      -0.155
bed_temperature           1.1021      0.255      4.318      0.000       0.588       1.617
fan_speed                -0.2052      0.043     -4.812      0.000      -0.291      -0.119
==============================================================================
Omnibus:                        0.429   Durbin-Watson:                   1.310
```

```
Prob(Omnibus):                  0.807   Jarque-Bera (JB):                 0.404
Skew:                           0.202   Prob(JB):                         0.817
Kurtosis:                       2.824   Cond. No.                     3.43e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 2.54e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

```
interaction = pd.concat([X,inter_mn,inter_bn,inter_fn,inter_fb], axis = 1)

interaction = interaction.rename(columns = {0:'interct_mn', 1:'interact_bn', 2:'interact_fn',
                            3:'interact_fb'})

interaction.head()
```

	layer_height	wall_thickness	infill_density	nozzle_temperature	bed_temperature	f
0	0.02	8	90	220	60	
1	0.02	7	90	225	65	
2	0.02	1	80	230	70	
3	0.02	4	70	240	75	
4	0.02	6	90	250	80	

```
linear_Regression(interaction,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       tension_strenght   R-squared:                       0.718
Model:                           OLS   Adj. R-squared:                  0.663
Method:                Least Squares   F-statistic:                     13.03
Date:               Sat, 03 Sep 2022   Prob (F-statistic):           4.06e-09
Time:                       09:23:31   Log-Likelihood:                -148.27
No. Observations:                 50   AIC:                             314.5
Df Residuals:                     41   BIC:                             331.7
Df Model:                          8
Covariance Type:           nonrobust
==============================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const                 -0.0870      0.068     -1.280      0.208      -0.224       0.050
layer_height          56.6017     11.739      4.822      0.000      32.895      80.309
```

```
wall_thickness         1.1373      0.268      4.242      0.000       0.596       1.679
infill_density         0.1587      0.032      4.928      0.000       0.094       0.224
nozzle_temperature     0.0004      0.000      1.319      0.194      -0.000       0.001
bed_temperature       -4.7917      3.832     -1.251      0.218     -12.530       2.946
fan_speed              2.1355      1.273      1.678      0.101      -0.435       4.706
interct_mn             0.1191      0.096      1.235      0.224      -0.076       0.314
interact_bn            0.0221      0.017      1.263      0.214      -0.013       0.057
interact_fn           -0.0138      0.007     -2.022      0.050      -0.028   -1.34e-05
interact_fb            0.0077      0.004      1.830      0.075      -0.001       0.016
==============================================================================
Omnibus:                        0.134   Durbin-Watson:                   1.472
Prob(Omnibus):                  0.935   Jarque-Bera (JB):                0.207
Skew:                          -0.114   Prob(JB):                        0.902
Kurtosis:                       2.782   Cond. No.                     5.90e+20
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 6.34e-32. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

Model 3 : Predicitng Elongation based on the features

```
printer.head()
```

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	be
0	0.02	8	90	0	220	
1	0.02	7	90	1	225	
2	0.02	1	80	0	230	
3	0.02	4	70	1	240	
4	0.02	6	90	0	250	

```
X = printer.drop(['roughness','tension_strenght','elongation'], axis = 1)
X.head()
```

	layer_height	wall_thickness	infill_density	infill_pattern	nozzle_temperature	be
0	0.02	8	90	0	220	
1	0.02	7	90	1	225	
2	0.02	1	80	0	230	
3	0.02	4	70	1	240	

```
linear_Regression(X,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       tension_strenght   R-squared:                       0.673
Model:                            OLS   Adj. R-squared:                  0.609
Method:                 Least Squares   F-statistic:                     10.55
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           6.91e-08
Time:                        09:24:56   Log-Likelihood:                -151.94
No. Observations:                  50   AIC:                             321.9
Df Residuals:                      41   BIC:                             339.1
Df Model:                           8
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
const                   0.0663      0.023      2.863      0.007       0.020       0.113
layer_height           55.5972     12.788      4.348      0.000      29.772      81.423
wall_thickness          1.0687      0.319      3.346      0.002       0.424       1.714
infill_density          0.1629      0.034      4.769      0.000       0.094       0.232
infill_pattern         -1.1427      1.646     -0.694      0.491      -4.466       2.181
nozzle_temperature     -1.0468      0.369     -2.837      0.007      -1.792      -0.302
bed_temperature         3.8647      1.354      2.855      0.007       1.131       6.599
print_speed            -0.0156      0.030     -0.519      0.607      -0.076       0.045
material              -17.3051      8.515     -2.032      0.049     -34.502      -0.108
fan_speed              -0.5719      0.181     -3.166      0.003      -0.937      -0.207
==============================================================================
Omnibus:                        0.265   Durbin-Watson:                   1.461
Prob(Omnibus):                  0.876   Jarque-Bera (JB):                0.456
Skew:                           0.060   Prob(JB):                        0.796
Kurtosis:                       2.548   Cond. No.                     3.59e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 2.48e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

```
X = X.drop(['infill_pattern','print_speed'], axis = 1)
X.head()
```

	layer_height	wall_thickness	infill_density	nozzle_temperature	bed_temperature	m
0	0.02	8	90	220	60	
1	0.02	7	90	225	65	
2	0.02	1	80	230	70	
3	0.02	4	70	240	75	
4	0.02	6	90	250	80	

```
linear_Regression(X,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       tension_strenght   R-squared:                       0.667
Model:                            OLS   Adj. R-squared:                  0.620
Method:                 Least Squares   F-statistic:                     14.33
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           6.78e-09
Time:                        09:25:44   Log-Likelihood:                -152.42
No. Observations:                  50   AIC:                             318.8
Df Residuals:                      43   BIC:                             332.2
Df Model:                           6
Covariance Type:            nonrobust
=====================================================================================
                        coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------------
const                  0.0646      0.023      2.840      0.007       0.019       0.111
layer_height          56.3670     12.448      4.528      0.000      31.262      81.472
wall_thickness         1.1117      0.280      3.967      0.000       0.547       1.677
infill_density         0.1664      0.033      4.985      0.000       0.099       0.234
nozzle_temperature    -1.0289      0.363     -2.833      0.007      -1.761      -0.296
bed_temperature        3.7661      1.330      2.831      0.007       1.084       6.449
material             -17.1036      8.392     -2.038      0.048     -34.028      -0.180
fan_speed             -0.5565      0.177     -3.140      0.003      -0.914      -0.199
==============================================================================
Omnibus:                        0.578   Durbin-Watson:                   1.501
Prob(Omnibus):                  0.749   Jarque-Bera (JB):                0.710
Skew:                           0.195   Prob(JB):                        0.701
Kurtosis:                       2.566   Cond. No.                     5.42e+18
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
[2] The smallest eigenvalue is 1.02e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

```
interaction = pd.concat([X,inter_mn,inter_bn,inter_fn,inter_fb], axis = 1)

interaction = interaction.rename(columns = {0:'interct_mn', 1:'interact_bn', 2:'interact_fn',
                            3:'interact_fb'})
```

```
interaction.head()
```

	layer_height	wall_thickness	infill_density	nozzle_temperature	bed_temperature	m
0	0.02	8	90	220	60	
1	0.02	7	90	225	65	
2	0.02	1	80	230	70	
3	0.02	4	70	240	75	
4	0.02	6	90	250	80	

```
linear_Regression(interaction,y)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       tension_strenght   R-squared:                       0.718
Model:                            OLS   Adj. R-squared:                  0.654
Method:                 Least Squares   F-statistic:                     11.30
Date:                Sat, 03 Sep 2022   Prob (F-statistic):           1.54e-08
Time:                        09:26:13   Log-Likelihood:                -148.26
No. Observations:                  50   AIC:                             316.5
Df Residuals:                      40   BIC:                             335.6
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const                -0.0781      0.164     -0.477      0.636      -0.409       0.253
layer_height         56.6034     11.884      4.763      0.000      32.585      80.622
wall_thickness        1.1375      0.271      4.191      0.000       0.589       1.686
infill_density        0.1587      0.033      4.868      0.000       0.093       0.225
nozzle_temperature    0.0004      0.001      0.482      0.633      -0.001       0.002
bed_temperature      -4.3166      8.863     -0.487      0.629     -22.229      13.595
material            -29.9509    502.401     -0.060      0.953   -1045.341     985.440
fan_speed             1.8551      4.876      0.380      0.706      -8.000      11.711
interct_mn            0.2557      2.295      0.111      0.912      -4.382       4.893
interact_bn           0.0199      0.040      0.494      0.624      -0.062       0.101
interact_fn          -0.0120      0.031     -0.389      0.700      -0.074       0.050
interact_fb           0.0062      0.024      0.256      0.800      -0.043       0.056
==============================================================================
Omnibus:                        0.143   Durbin-Watson:                   1.471
Prob(Omnibus):                  0.931   Jarque-Bera (JB):                0.217
Skew:                          -0.118   Prob(JB):                        0.897
Kurtosis:                       2.779   Cond. No.                     5.64e+20
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifi
```

```
[2] The smallest eigenvalue is 6.94e-32. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
  x = pd.concat(x[::order], 1)
```

Practical 3

To determine mechanical properties from stress strain curve data.

Aim: To determine mechanical properties from stress strain curve data.

Prerequisites: Tensile Test data, Jupyter Notebook / Google Colab

Theory:

Tensile Test:

A tensile test is a type of mechanical test performed by engineers used to determine the mechanical properties of a material. Engineering metal alloys such as steel and aluminium alloys are tensile tested in order to determine their strength and stiffness. Tensile tests are performed in a piece of equipment called a mechanical test frame.

Fig: Tensile Test Set up for Data Acquisition

After a tensile test is complete, a set of data is produced by the mechanical test frame. Using the data acquired during a tensile test, a stress-strain curve can be produced.

In this post, we will create a stress-strain curve (a plot) from a set of tensile test data of a steel 1045 sample and an aluminium 6061 sample. The stress strain curve we construct will have the following features:

- A descriptive title
- Axes labels with units
- Two lines on the same plot. One line for steel 1045 and one line for aluminum 6061
- A legend

Procedure:

1. Install Python

We are going to build our stress strain curve with Python and a Jupyter notebook. I suggest engineers and problem-solvers download and install the Anaconda distribution of Python. See this post to learn how to install Anaconda on your computer. Alternatively, you can download Python form Python.org or download Python the Microsoft Store.

2. Open a Jupyter notebook

We will construct our stress strain curve using a Jupyter notebook. See this post to see how to open a Jupyter notebook.

Make sure to save your Jupyter notebook with a recognizable name.

3. Download the data and move the data into the same folder as the Jupyter notebook

Next, we need to download the two data files that we will use to build our stress-strain curve. You can download sample data using the links below:

steel1045.xls

aluminum6061.xls

After these .xls files are downloaded, both .xls files need to be moved into the same folder as our Jupyter notebook.

4. Install Jupyter, NumPy, Pandas, and Matplotlib

Once Python is installed, the next thing we need to do is install a couple of Python packages. If you are using the Anaconda distribution of Python, the packages we are going to use to build the plot: Jupyter, NumPy, Pandas, and Matplotlib come pre-installed and no additional installation steps are necessary.

5. Import NumPy, Pandas, and Matplotlib

Now that our Jupyter notebook is open and the two .xls data files are in the same folder as the Jupyter notebook, we can start coding and build our plot.

At the top of the Jupyter notebook, import NumPy, Pandas and Matplotlib. The command %matplotlib inline is included so that our plot will display directly inside our Jupyter notebook. If you are using a .py file instead of a Jupyter notebook, make sure to comment out %matplotlib inline as this line is not valid Python code.

We will also print out the versions of our NumPy and Pandas packages using the .__version__ attribute. If the versions of NumPy and Pandas prints out, that means that NumPy and Pandas are installed and we can use these packages in our code.

6. Ensure the two .xls data files are in the same folder as the Jupyter notebook
7. Before we proceed, let's make sure the two .xls data files are in the same folder as our running Jupyter notebook. We'll use a Jupyter notebook magic command to print out the contents of the folder that our notebook is in. The %ls command lists the contents of the current folder.

8. Create stress and strain series from the **FORCE**, **EXT**, and **CH5** columns

- **FORCE** Force measurements from the load cell in pounds (lb), force in pounds
- **EXT** Extension measurements from the mechanical extensometer in percent (%), strain in percent
- **CH5** Extension readings from the laser extensometer in percent (%), strain in percent

Next we'll create a four Pandas series from the ['CH5'] and ['FORCE'] columns of our al_df and steel_df dataframes. The equations below show how to calculate stress, σ and strain, ϵ, from force F and cross-sectional area A. Cross-sectional area A is the formula for the area of a circle. For the steel and aluminium samples we tested, the diameter d was 0.506 in.

$$\sigma = \frac{F}{A_0}$$

$$F\,(kip) = F\,(lb) \times 0.001$$

$$A_0 = \pi(d/2)^2$$

$$d = 0.506\ in$$

$$\epsilon\,(unitless) = \epsilon\,(\%) \times 0.01$$

Calculate ductility

The ductility of a metal is calculated from a stress strain curve by drawing a line down from the fracture point on the curve, parallel to the linear elastic region. Where that line crosses the strain axis is the ductility.

Point-Slope Formula for a line:

$$y-y1=m(x-x1)$$

Where m= slope in the linear elastic region (elastic modulus),

x1= last strain point on the stress strain curve,

y1= last stress point on the stress strain curve.

Solve the equation above for x, when y=0 in terms of x1, x2 and m

$$x= (- y1/ m)+ x1$$

Substitute in ductility, elastic modulus EE, and the last stress and strain points.

$$\%EL= (-stresslast / E) + strainlast$$

Results :

Sr. No	Mechanical property	Mechanical property from Program	
		Steel1045	Aluminium6061
1	Tensile Strength	851.7 KN/mm2	334.3 KN/mm2
2	Elastic Modulus	192235.2 KN/mm2	89631.5 KN/mm2
3	Ductility	17.7%	17.6%

Conclusion:

In this practical, we built a stress strain curve using Python. With the help of python libraries like Numpy, Panda, Matplotlib, scipy we have calculated the mechanical properties of metals steal1045 and aluminium6061 from tensile test data.

Curve_Table

TESTNUM	POINTNUM	TIME	POSIT	FORCE	EXT	CH5	CH6	CH7	CH8
761	1	6.532	0.01524	201.1585	0.018893	-0.02308			
761	2	6.702	0.016	205.9781	0.000265	-0.01302			
761	3	7.098	0.0172	219.2954	-0.00088	-0.02488			
761	4	8.697	0.0235	268.5059	0.001453	-0.0068			
761	5	10.196	0.03004	322.0282	0.001865	0.012563			
761	6	10.995	0.03312	375.0427	0.004141	0.015867			
761	7	11.697	0.03624	436.0469	0.006737	0.029149			
761	8	12.395	0.0389	488.5532	0.009003	0.045813			
761	9	12.993	0.04164	542.2004	0.011371	0.03416			
761	10	13.696	0.0443	599.1447	0.014223	0.053578			
761	11	14.297	0.04704	651.396	0.016873	0.045611			
761	12	14.898	0.04934	700.8568	0.019533	0.067568			
761	13	15.396	0.05166	753.7414	0.022385	0.072624			
761	14	15.994	0.05398	810.8106	0.025036	0.079838			
761	15	16.805	0.05746	878.6589	0.030904	0.081209			
761	16	17.299	0.0594	926.8498	0.033198	0.081636			
761	17	17.894	0.0617	977.8303	0.036279	0.080456			
761	18	18.396	0.06404	1029.571	0.038847	0.078333			
761	19	19.093	0.06674	1083.341	0.042055	0.085513			
761	20	19.597	0.06906	1137.237	0.044788	0.069972			
761	21	20.101	0.07098	1186.948	0.04785	0.065624			
761	22	20.599	0.07328	1249.085	0.051077	0.072107			
761	23	21.097	0.07522	1297.907	0.053746	0.072748			
761	24	21.597	0.07718	1346.222	0.054733	0.075456			
761	25	21.997	0.07914	1402.906	0.056607	0.072546			
761	26	22.493	0.08104	1458.701	0.058325	0.072748			
761	27	22.996	0.08296	1516.652	0.060437	0.067006			
761	28	23.495	0.08528	1583.859	0.063261	0.071276			
761	29	23.894	0.08684	1634.453	0.065172	0.074018			
761	30	24.299	0.08836	1682.765	0.06742	0.078557			
761	31	24.7	0.0903	1743.25	0.070043	0.063792			
761	32	25.199	0.0922	1803.861	0.073115	0.080614			
761	33	25.597	0.09378	1853.313	0.075336	0.080558			
761	34	25.993	0.0953	1902.638	0.077795	0.068703			
761	35	26.294	0.09678	1952.089	0.080089	0.075153			
761	36	26.694	0.09832	2008.133	0.082456	0.075231			
761	37	27.104	0.0999	2060.119	0.08518	0.079198			
761	38	27.497	0.1018	2125.165	0.088178	0.085153			
761	39	27.895	0.10336	2181.334	0.090884	0.093019			
761	40	28.294	0.1049	2234.461	0.093544	0.088862			
761	41	28.697	0.10682	2300.772	0.097044	0.097626			
761	42	29.095	0.1084	2357.194	0.099905	0.097919			
761	43	29.497	0.10992	2411.967	0.10262	0.102537			
761	44	29.9	0.11186	2483.222	0.106523	0.103324			
761	45	30.294	0.1134	2539.895	0.109594	0.101784			
761	46	30.696	0.11498	2596.441	0.112848	0.113459			
761	47	31.098	0.11648	2653.621	0.116029	0.116212			
761	48	31.499	0.11842	2728.55	0.120124	0.118078			
761	49	31.897	0.12	2787.63	0.12357	0.124224			
761	50	32.295	0.1215	2849.752	0.126806	0.125112			
761	51	32.699	0.12342	2922.143	0.130937	0.12291			
761	52	32.994	0.12458	2971.207	0.13367	0.130236			
761	53	33.396	0.12612	3030.665	0.137125	0.131056			
761	54	33.795	0.12768	3093.419	0.140407	0.133236			
761	55	34.094	0.12922	3155.538	0.143798	0.133551			
761	56	34.493	0.13074	3220.192	0.147509	0.137641			
761	57	34.897	0.13228	3278.508	0.150909	0.122247			

Curve_Table

TESTNUM	POINTNUM	TIME	POSIT	FORCE	EXT	CH5	CH6	CH7	CH8
762	1	5.969	0.01284	201.0308	0.00157222	-0.007133			
762	2	6.242	0.01392	215.2359	9.09E-06	-0.014581			
762	3	6.936	0.01646	246.8167	-0.0008318	0.006942			
762	4	8.632	0.023399999	371.8704	0.00220284	0.000776			
762	5	10.533	0.031099999	502.5019	0.00148073	0.018102			
762	6	11.835	0.03692	637.5695	0.00022851	0.031187			
762	7	13.035	0.041540001	761.9814	-0.0003383	0.04012			
762	8	13.838	0.045019999	885.5032	0.00096887	0.042809			
762	9	14.731	0.048859999	1017.266	0.00240393	0.046578			
762	10	15.735	0.053100001	1155.112	0.0041315	0.048074			
762	11	16.532	0.056200001	1285.982	0.00582248	0.042539			
762	12	16.631	0.056639999	1160.692	0.00541113	0.042539			
762	13	16.736	0.057080001	979.7283	0.00607841	0.041673			
762	14	17.036	0.058499999	1139.007	0.00848234	0.043484			
762	15	17.332	0.05954	1278.12	0.01095943	0.04039			
762	16	17.937	0.06188	1419.131	0.01297027	0.035215			
762	17	18.534	0.064560004	1561.027	0.01446024	0.046927			
762	18	19.132	0.0669	1691	0.01606889	0.048716			
762	19	19.639	0.069219999	1829.086	0.01793355	0.047838			
762	20	20.242	0.071520001	1966.028	0.01995362	0.049526			
762	21	20.831	0.073819987	2105.376	0.02211988	0.052811			
762	22	21.438	0.076520003	2245.355	0.0239937	0.058189			
762	23	21.941	0.078440003	2373.286	0.02639763	0.049852			
762	24	22.435	0.08072	2517.697	0.02835371	0.058864			
762	25	23.034	0.082979999	2662.484	0.03105015	0.048423			
762	26	23.532	0.084879987	2787.618	0.03275029	0.051292			
762	27	23.937	0.086759999	2919.722	0.03517248	0.050763			
762	28	24.433	0.088639997	3054.992	0.03764957	0.045656			
762	29	24.937	0.090939999	3210.67	0.04115954	0.05685			
762	30	25.452	0.092900001	3352.527	0.04399307	0.060225			
762	31	26.034	0.09516	3514.029	0.04694544	0.061114			
762	32	26.437	0.097059987	3659.555	0.04981555	0.067538			
762	33	26.935	0.099019997	3804.951	0.05241141	0.066593			
762	34	27.431	0.100959986	3952.879	0.05509876	0.07439			
762	35	27.833	0.102880001	4103.845	0.05771294	0.075313			
762	36	28.336	0.104819998	4249.738	0.06032705	0.078069			
762	37	28.733	0.106339999	4380.165	0.06256647	0.08222			
762	38	29.132	0.108319998	4536.698	0.06504357	0.087565			
762	39	29.639	0.11022	4691.453	0.06789543	0.096453			
762	40	30.031	0.111780003	4816.927	0.06996118	0.093674			
762	41	30.432	0.11372	4974.59	0.07241994	0.099434			
762	42	30.84	0.11524	5099.045	0.07419318	0.099445			
762	43	31.238	0.116779998	5225.272	0.07611267	0.100672			
762	44	31.533	0.118340001	5356.693	0.078279	0.099007			
762	45	31.941	0.119879998	5488.744	0.0801985	0.107006			
762	46	32.333	0.121420003	5619.906	0.08237392	0.106848			
762	47	32.732	0.123340003	5786.293	0.08485094	0.114983			
762	48	33.136	0.124899998	5917.956	0.08701727	0.112102			
762	49	33.539	0.126419991	6050.629	0.08940294	0.117053			
762	50	33.931	0.127959996	6186.848	0.09145036	0.113239			
762	51	34.233	0.129539996	6324.583	0.09358928	0.120889			
762	52	34.64	0.131080002	6461.43	0.0957647	0.120979			
762	53	35.036	0.132579997	6597.513	0.09802237	0.119089			
762	54	35.436	0.134539992	6769.83	0.1010479	0.12854			
762	55	35.835	0.136059999	6905.654	0.10313191	0.119303			
762	56	36.237	0.137639999	7042.361	0.10522507	0.130734			
762	57	36.539	0.139159992	7186.16	0.10752849	0.133018			
762	58	36.939	0.140699998	7333.124	0.1097405	0.13367			

Mechanical Properties from Stress-Strain Curves

```
!pip install pandas==1.2.0
!pip install xlrd==1.2.0
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting pandas==1.2.0
  Downloading pandas-1.2.0-cp37-cp37m-manylinux1_x86_64.whl (9.9 MB)
     |████████████████████████████████| 9.9 MB 8.7 MB/s
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: numpy>=1.16.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from
Installing collected packages: pandas
  Attempting uninstall: pandas
    Found existing installation: pandas 1.3.5
    Uninstalling pandas-1.3.5:
      Successfully uninstalled pandas-1.3.5
Successfully installed pandas-1.2.0
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting xlrd==1.2.0
  Downloading xlrd-1.2.0-py2.py3-none-any.whl (103 kB)
     |████████████████████████████████| 103 kB 8.0 MB/s
Installing collected packages: xlrd
  Attempting uninstall: xlrd
    Found existing installation: xlrd 1.1.0
    Uninstalling xlrd-1.1.0:
      Successfully uninstalled xlrd-1.1.0
Successfully installed xlrd-1.2.0
```

```
from google.colab import drive
drive.mount('/content/drive',force_remount=True)
df_steel = pd.read_excel('/content/steel1045.xls')
df_al = pd.read_excel('/content/aluminum6061.xls')
```

```
Mounted at /content/drive
WARNING *** OLE2 inconsistency: SSCS size is 0 but SSAT size is non-zero
```

```
df_steel.head()
```

	TESTNUM	POINTNUM	TIME	POSIT	FORCE	EXT	CH5	CH6	CH7	
0	762	1	5.969	0.01284	201.030792	0.001572	-0.007133	NaN	NaN	N
1	762	2	6.242	0.01392	215.235886	0.000009	-0.014581	NaN	NaN	N
2	762	3	6.936	0.01646	246.816742	-0.000832	0.006942	NaN	NaN	N

```
df_al.head()
```

	TESTNUM	POINTNUM	TIME	POSIT	FORCE	EXT	CH5	CH6	CH7	
0	761	1	6.532	0.01524	201.158508	0.018893	-0.023081	NaN	NaN	N
1	761	2	6.702	0.01600	205.978119	0.000265	-0.013024	NaN	NaN	N
2	761	3	7.098	0.01720	219.295441	-0.000877	-0.024879	NaN	NaN	N
3	761	4	8.697	0.02350	268.505890	0.001453	-0.006798	NaN	NaN	N
4	761	5	10.196	0.03004	322.028168	0.001865	0.012563	NaN	NaN	N

We see a number of columns in each dataframe. The columns we are interested in are FORCE, EXT, and CH5. Below is a description of what these columns mean.

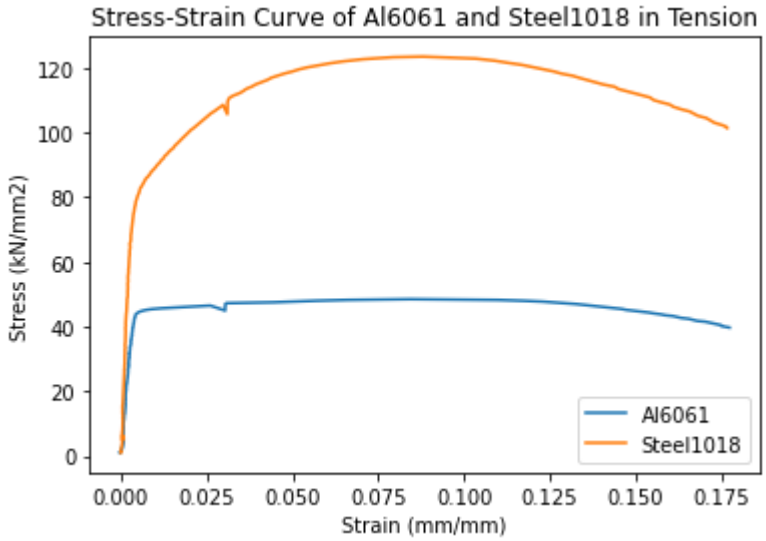
FORCE Force measurements from the load cell in pounds (lb), force in pounds EXT Extension measurements from the mechanical extensometer in percent (%), strain in percent CH5 Extension readings from the laser extensometer in percent (%), strain in percent

```
d = 0.506
r = d/2
A = np.pi*r**2

stress_al = (df_al['FORCE']/A)*0.001
strain_al = df_al['CH5']*0.01

stress_steel = (df_steel['FORCE']/A)*0.001
strain_steel = df_steel['CH5']*0.01
```

Plot the full stress strain curve

```
fig,ax = plt.subplots()
ax.plot(strain_al, stress_al)
ax.plot(strain_steel, stress_steel)
ax.set_xlabel('Strain (mm/mm)')
ax.set_ylabel('Stress (kN/mm2)')
ax.set_title('Stress Strain Curve of Al6061 and Steel1018 in Tension')
```

```
ax.set_title( 'Stress-Strain Curve of Al6061 and Steel1018 in Tension ')
ax.legend(['Al6061','Steel1018'])

plt.show()
```



Calculate tensile strength

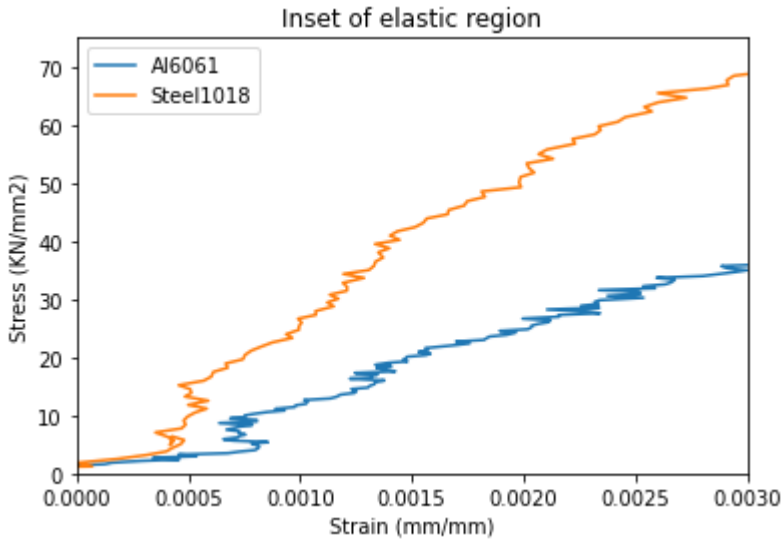
```
# Calculate the tensile strength
ts_al = np.max(stress_al)
ts_steel = np.max(stress_steel)
final_ts_steel=6.8976*ts_steel
print(f'The tensile strength of Steel1018 in KN/mm2: {round(final_ts_steel,1)} KN/mm2')
final_ts_steel=6.8976*ts_al
print(f'The tensile strength of Al6061 in KN/mm2: {round(final_ts_steel,1)} KN/mm2')
```

```
The tensile strength of Steel1018 in KN/mm2: 851.7 KN/mm2
The tensile strength of Al6061 in KN/mm2: 334.3 KN/mm2
```

Caclulate elastic modulus

```
fig,ax = plt.subplots()
ax.plot(strain_al, stress_al)
ax.plot(strain_steel, stress_steel)

ax.set_title('Inset of elastic region')
ax.set_xlabel('Strain (mm/mm)')
ax.set_ylabel('Stress (KN/mm2)')
ax.legend(['Al6061','Steel1018'])

ax.set_xlim([0,0.003])
```

```
ax.set_ylim([0,75])

plt.show()
```




```
# Find the elastic modulus of Al6061
# use stress and strain values from stress=0 to stress=35 N/mm2
linear_stress_al_mask = stress_al < 35
linear_stress_al = stress_al[linear_stress_al_mask]
linear_strain_al = strain_al[linear_stress_al_mask]
from scipy.stats import linregress

linear_regression_output = linregress(linear_strain_al, linear_stress_al)
E_al = linear_regression_output[0]
final_E_al=6.8976*E_al
print(f'The elastic modulus of Al6061 is {round(final_E_al,1)} KN/mm2')
```

```
The elastic modulus of Al6061 is 89631.5 KN/mm2
```

```
# Find the elastic modulus of Steel1018
# use stress and strain values from stress=0 to stress=55 N/mm2
linear_stress_steel_mask = stress_steel < 55
linear_stress_steel = stress_steel[linear_stress_steel_mask]
linear_strain_steel = strain_steel[linear_stress_steel_mask]

linear_regression_output_steel = linregress(linear_strain_steel, linear_stress_steel)
E_steel = linear_regression_output_steel[0]
final_E_steel=6.8976*E_steel
print(f'The elastic modulus of Steel1018 is {round(final_E_steel,1)} KN/mm2')
```

```
The elastic modulus of Steel1018 is 192235.2 KN/mm2
```

Calculate ductility

```
# Find the ductility for Al6061
stress_al_array = np.array(stress_al)
stress_al_last = stress_al_array[-1]
strain_al_array = np.array(strain_al)
strain_al_last = strain_al_array[-1]
EL_al = -stress_al_last/final_E_al + strain_al_last
print(f'The ductility of Al6061 is {round(EL_al*100,1)}%')
```

```
The ductility of Al6061 is 17.7%
```

```
# Find the ductility of Steel1018
stress_steel_array = np.array(stress_steel)
stress_steel_last = stress_steel_array[-1]
strain_steel_array = np.array(strain_steel)
strain_steel_last = strain_steel_array[-1]
EL_steel = -stress_steel_last/final_E_steel + strain_steel_last
print(f'The ductility of Steel1018 is {round(EL_steel*100,1)}%')
```

```
The ductility of Steel1018 is 17.6%
```

Practical 4

To perform Machine Predictive Maintenance

Aim: To perform Machine Predictive Maintenance

Prerequisites: Maintenance data, Jupyter Notebook / Google Colab

Theory:

Machine Predictive Maintenance Classification Dataset

The dataset consists of 10 000 data points stored as rows with 14 features in columns
UID: unique identifier ranging from 1 to 10000
product ID: consisting of a letter L, M, or H for low (50% of all products), medium (30%) and high (20%) as product quality variants and a variant-specific serial number
air temperature [K]: generated using a random walk process later normalized to a standard deviation of 2 K around 300 K
process temperature [K]: generated using a random walk process normalized to a standard deviation of 1 K, added to the air temperature plus 10 K.
rotational speed [rpm]: calculated from a power of 2860 W, overlaid with a normally distributed noise
torque [Nm]: torque values are normally distributed around 40 Nm with a Ïƒ = 10 Nm and no negative values.
tool wear [min]: The quality variants H/M/L add 5/3/2 minutes of tool wear to the used tool in the process. and a
'machine failure' label that indicates, whether the machine has failed in this particular datapoint for any of the following failure modes are true.

The machine failure consists of five independent failure modes
tool wear failure (TWF): the tool will be replaced of fail at a randomly selected tool wear time between 200 â€“ 240 mins (120 times in our dataset). At this point in time, the tool is replaced 69 times, and fails 51 times (randomly assigned).
heat dissipation failure (HDF): heat dissipation causes a process failure, if the difference between air- and process temperature is below 8.6 K and the toolâ€™s rotational speed is below 1380 rpm. This is the case for 115 data points.
power failure (PWF): the product of torque and rotational speed (in rad/s) equals the power required for the process. If this power is below 3500 W or above 9000 W, the process fails, which is the case 95 times in our dataset.
overstrain failure (OSF): if the product of tool wear and torque exceeds 11,000 minNm for the L product variant (12,000 M, 13,000 H), the process fails due to overstrain. This is true for 98 datapoints.
random failures (RNF): each process has a chance of 0,1 % to fail regardless of its process parameters. This is the case for only 5 datapoints, less than could be expected for 10,000 datapoints in our dataset.
If at least one of the above failure modes is true, the process fails and the 'machine failure' label is set to 1. It is therefore not transparent to the machine learning method, which of the failure modes has caused the process to fail

Important : There are two Targets - Do not make the mistake of using one of them as feature, as it will lead to leakage.

- Target : Failure or Not
- Failure Type : Type of Failure

UDI	Product ID	Type	Air temper	Process ten	Rotational	Torque [Nn	Tool wear [	Target	Failure Type
1	M14860	M	298.1	308.6	1551	42.8	0	0	No Failure
2	L47181	L	298.2	308.7	1408	46.3	3	0	No Failure
3	L47182	L	298.1	308.5	1498	49.4	5	0	No Failure
4	L47183	L	298.2	308.6	1433	39.5	7	0	No Failure
5	L47184	L	298.2	308.7	1408	40	9	0	No Failure
6	M14865	M	298.1	308.6	1425	41.9	11	0	No Failure
7	L47186	L	298.1	308.6	1558	42.4	14	0	No Failure
8	L47187	L	298.1	308.6	1527	40.2	16	0	No Failure
9	M14868	M	298.3	308.7	1667	28.6	18	0	No Failure
10	M14869	M	298.5	309	1741	28	21	0	No Failure
11	H29424	H	298.4	308.9	1782	23.9	24	0	No Failure
12	H29425	H	298.6	309.1	1423	44.3	29	0	No Failure
13	M14872	M	298.6	309.1	1339	51.1	34	0	No Failure
14	M14873	M	298.6	309.2	1742	30	37	0	No Failure
15	L47194	L	298.6	309.2	2035	19.6	40	0	No Failure
16	L47195	L	298.6	309.2	1542	48.4	42	0	No Failure
17	M14876	M	298.6	309.2	1311	46.6	44	0	No Failure
18	M14877	M	298.7	309.2	1410	45.6	47	0	No Failure
19	H29432	H	298.8	309.2	1306	54.5	50	0	No Failure
20	M14879	M	298.9	309.3	1632	32.5	55	0	No Failure
21	H29434	H	298.9	309.3	1375	42.7	58	0	No Failure
22	L47201	L	298.8	309.3	1450	44.8	63	0	No Failure
23	M14882	M	298.9	309.3	1581	30.7	65	0	No Failure
24	L47203	L	299	309.4	1758	25.7	68	0	No Failure
25	M14884	M	299	309.4	1561	37.3	70	0	No Failure
26	L47205	L	299	309.5	1861	23.3	73	0	No Failure
27	L47206	L	299.1	309.5	1512	39	75	0	No Failure
28	H29441	H	299.1	309.4	1811	24.6	77	0	No Failure
29	L47208	L	299.1	309.4	1439	44.2	82	0	No Failure
30	L47209	L	299	309.4	1693	30.1	84	0	No Failure
31	M14890	M	299.1	309.5	1339	48.2	86	0	No Failure
32	L47211	L	299	309.4	1798	25.5	89	0	No Failure
33	L47212	L	299	309.4	1419	48.3	91	0	No Failure
34	L47213	L	298.9	309.3	1665	32.5	93	0	No Failure
35	M14894	M	298.8	309.1	1559	34.7	95	0	No Failure
36	M14895	M	298.8	309.2	1452	48.6	98	0	No Failure
37	M14896	M	298.9	309.2	1581	36.7	101	0	No Failure
38	L47217	L	298.8	309.1	1439	39.2	104	0	No Failure
39	H29452	H	298.9	309.2	1379	50.7	106	0	No Failure
40	L47219	L	298.8	309.1	1350	52.5	111	0	No Failure
41	L47220	L	298.8	309.1	1362	45.4	113	0	No Failure
42	L47221	L	298.8	309.1	1368	50.8	115	0	No Failure
43	M14902	M	298.8	309.1	1368	49.1	117	0	No Failure
44	H29457	H	298.8	309.2	1372	48.5	120	0	No Failure
45	M14904	M	298.8	309.1	1472	47.5	125	0	No Failure
46	L47225	L	298.8	309.1	1489	49.1	128	0	No Failure
47	M14906	M	298.7	309	1843	25.8	130	0	No Failure
48	L47227	L	298.8	309.1	1418	46.3	133	0	No Failure
49	H29462	H	298.8	309.2	1425	53.9	135	0	No Failure
50	M14909	M	298.9	309.2	1412	44.1	140	0	No Failure
51	L47230	L	298.9	309.1	2861	4.6	143	1	Power Failure
52	L47231	L	298.9	309.1	1383	54.9	145	0	No Failure
53	H29466	H	298.8	309	1497	43.8	147	0	No Failure
54	L47233	L	298.7	309	1565	35.1	152	0	No Failure
55	L47234	L	298.7	309	1691	30.1	154	0	No Failure
56	L47235	L	298.8	309.1	1512	40.7	156	0	No Failure
57	L47236	L	298.8	309.1	1477	42.4	158	0	No Failure
58	L47237	L	298.8	309.1	1513	40.3	160	0	No Failure

Machine Predictive Maintenance

Importing Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set_style('darkgrid')
```

Importing Dataset

```
data = pd.read_csv('/content/predictive_maintenance.csv')
```

```
data.head(10)
```

	UDI	Product ID	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Target	Fail T
0	1	M14860	M	298.1	308.6	1551	42.8	0	0	Fa
1	2	L47181	L	298.2	308.7	1408	46.3	3	0	Fa
2	3	L47182	L	298.1	308.5	1498	49.4	5	0	Fa
3	4	L47183	L	298.2	308.6	1433	39.5	7	0	Fa
4	5	L47184	L	298.2	308.7	1408	40.0	9	0	Fa
5	6	M14865	M	298.1	308.6	1425	41.9	11	0	

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 10 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   UDI                      10000 non-null  int64
 1   Product ID               10000 non-null  object
```

```
 2   Type                    10000 non-null  object
 3   Air temperature [K]     10000 non-null  float64
 4   Process temperature [K] 10000 non-null  float64
 5   Rotational speed [rpm]  10000 non-null  int64
 6   Torque [Nm]             10000 non-null  float64
 7   Tool wear [min]         10000 non-null  int64
 8   Target                  10000 non-null  int64
 9   Failure Type            10000 non-null  object
dtypes: float64(3), int64(4), object(3)
memory usage: 781.4+ KB
```

*Data Preprocessing : Drop unwanted *

```
data = data.drop(["UDI",'Product ID'],axis=1)
data.head(3)
```

	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Target	Failure Type
0	M	298.1	308.6	1551	42.8	0	0	No Failure
								No

EDA(Exploratory Data Analysis)

Exploratory Data Analysis (EDA) is an approach to perform initial investigations on data to discover patterns, spot anomalies, test hypothesis and check assumptions with the help of statistics and graphical representations.

```
data.describe()
```

	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Target
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	300.004930	310.005560	1538.776100	39.986910	107.951000	0.033900
std	2.000259	1.483734	179.284096	9.968934	63.654147	0.180981
min	295.300000	305.700000	1168.000000	3.800000	0.000000	0.000000
25%	298.300000	308.800000	1423.000000	33.200000	53.000000	0.000000
50%	300.100000	310.100000	1503.000000	40.100000	108.000000	0.000000
75%	301.500000	311.100000	1612.000000	46.800000	162.000000	0.000000

```
data.groupby(['Failure Type','Target']).count().drop(['Process temperature [K]',
                                                      'Rotational speed [rpm]',
                                                      'Torque [Nm]',
                                                      'Tool wear [min]',
                                                      'Air temperature [K]'],axis=1).rename(c
```

		count
Failure Type	**Target**	
Heat Dissipation Failure	**1**	112
No Failure	**0**	9643
	1	9
Overstrain Failure	**1**	78
Power Failure	**1**	95
Random Failures	**0**	18
Tool Wear Failure	**1**	45

```
data.groupby(['Target','Failure Type']).median()
```

		Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]
Target	**Failure Type**					
0	**No Failure**	300.00	310.0	1507.0	39.80	107.0
	Random Failures	300.75	311.1	1490.0	44.60	142.0
1	**Heat Dissipation Failure**	302.45	310.7	1346.0	52.35	106.0
	No Failure	300.50	309.9	1438.0	45.20	119.0
	Overstrain					

```
data.groupby(['Type','Target']).median()
```

Type	Target	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]
H	**0**	299.7	309.9	1502.0	40.2	106.0
L	**0**	300.1	310.1	1508.0	39.7	107.0

Skewness Analysis

Skewness measures the deviation of a random variable's given distribution from the normal distribution, which is symmetrical on both sides. Skewness Analysis are performed to see whether the numerical features are severely skewed or not and this will help us in creating better linear models. If the skewness is between -0.5 & 0.5, the data are nearly symmetrical. If the skewness is between -1 & -0.5 (negative skewed) or between 0.5 & 1(positive skewed), the data are slightly skewed. If the skewness is lower than -1 (negative skewed) or greater than 1 (positive skewed), the data are extremely skewed.

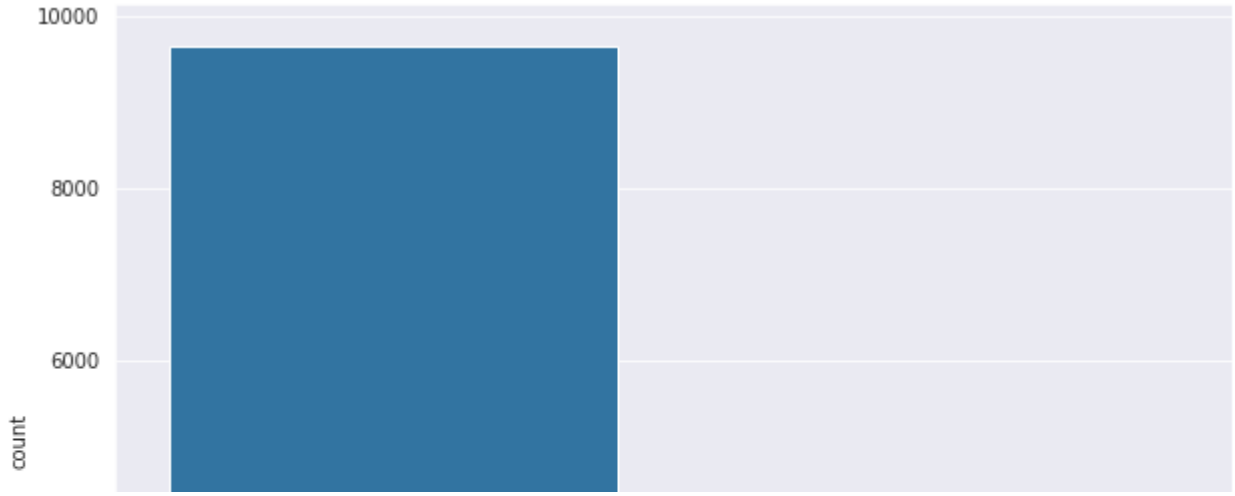
```
data_numeric = data.loc[:,['Air temperature [K]','Process temperature [K]','Rotational speed
data_numeric.skew()
```

```
Air temperature [K]        0.114274
Process temperature [K]    0.015027
Rotational speed [rpm]     1.993171
Torque [Nm]               -0.009517
Tool wear [min]            0.027292
dtype: float64
```

Data Visualisation

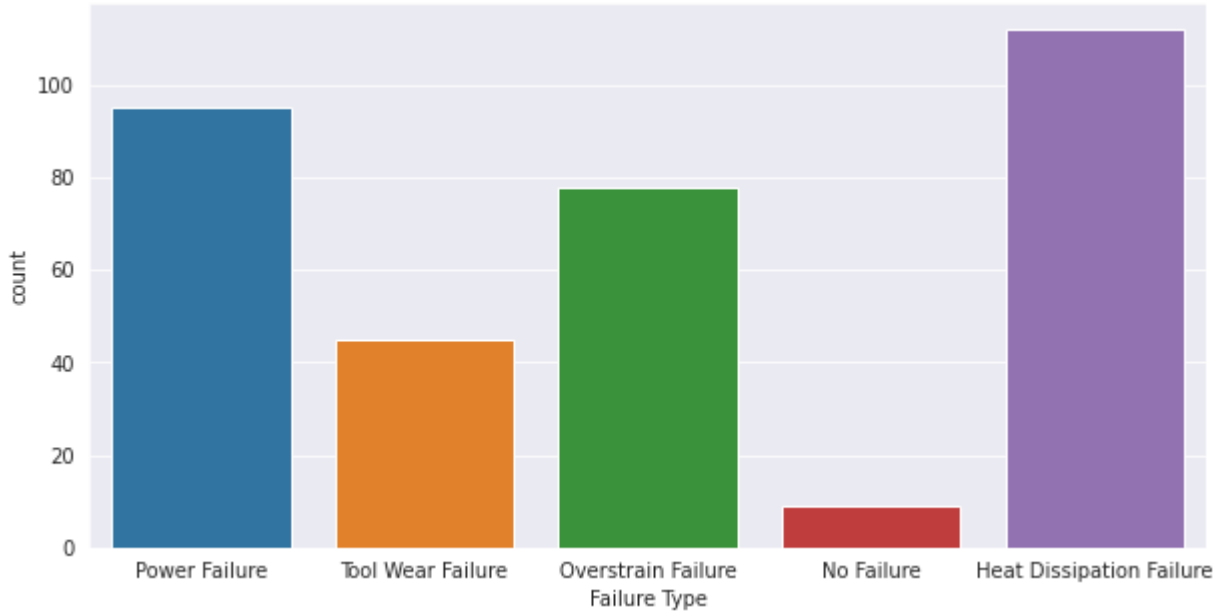
```
# Observe distribution of "Target : Failure or not" in a bar graph.
plt.figure(figsize=(10,8))
sns.countplot(data=data,x="Target")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f17a2b49850>
```

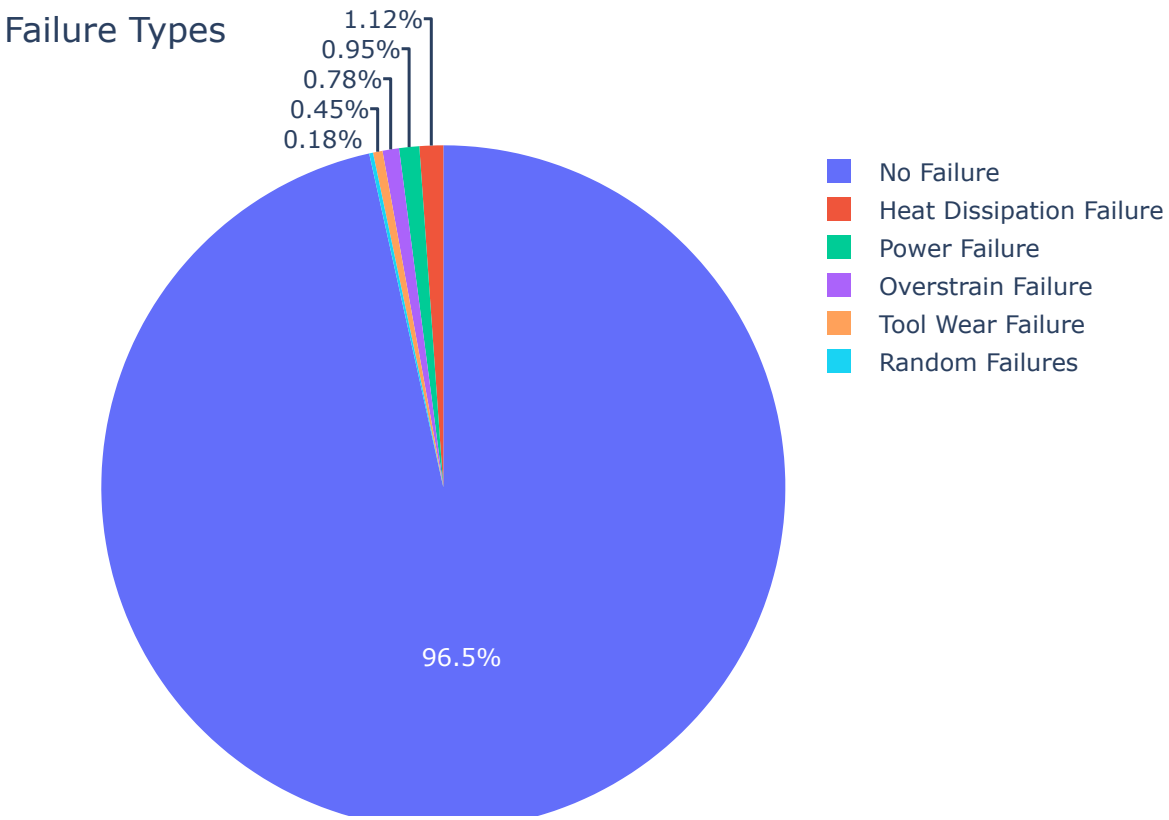



```
# Observe distribution of "Target Failure Type : Type of Failure" in a bar graph.
plt.figure(figsize=(10,5))
sns.countplot(data=data[data['Target']==1],x="Failure Type")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f17a2619fd0>
```

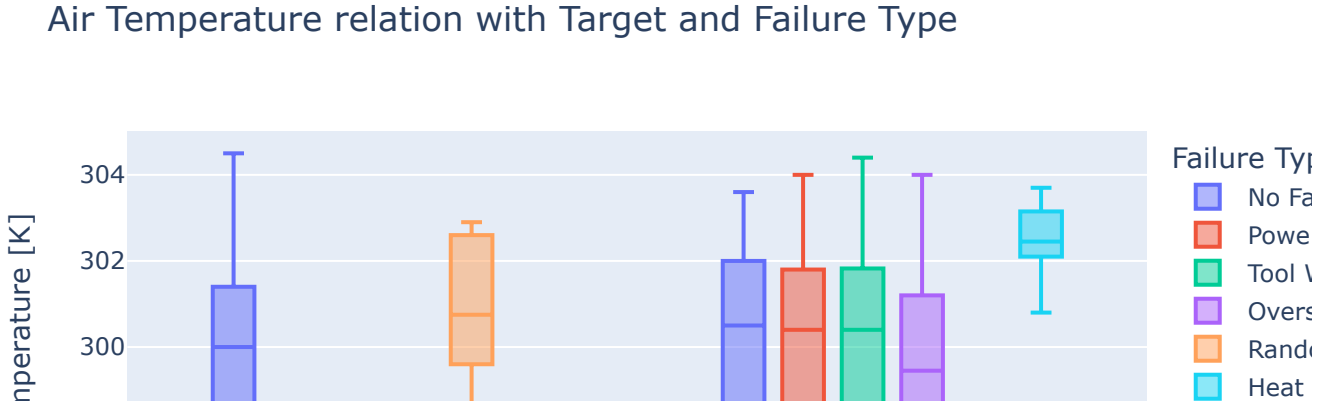



```
# Observe distribution of failures in a pie chart
import plotly.graph_objects as go
import plotly.express as px
fig = px.pie(data,
             title = 'Failure Types',
             names = 'Failure Type')
fig.show()
```

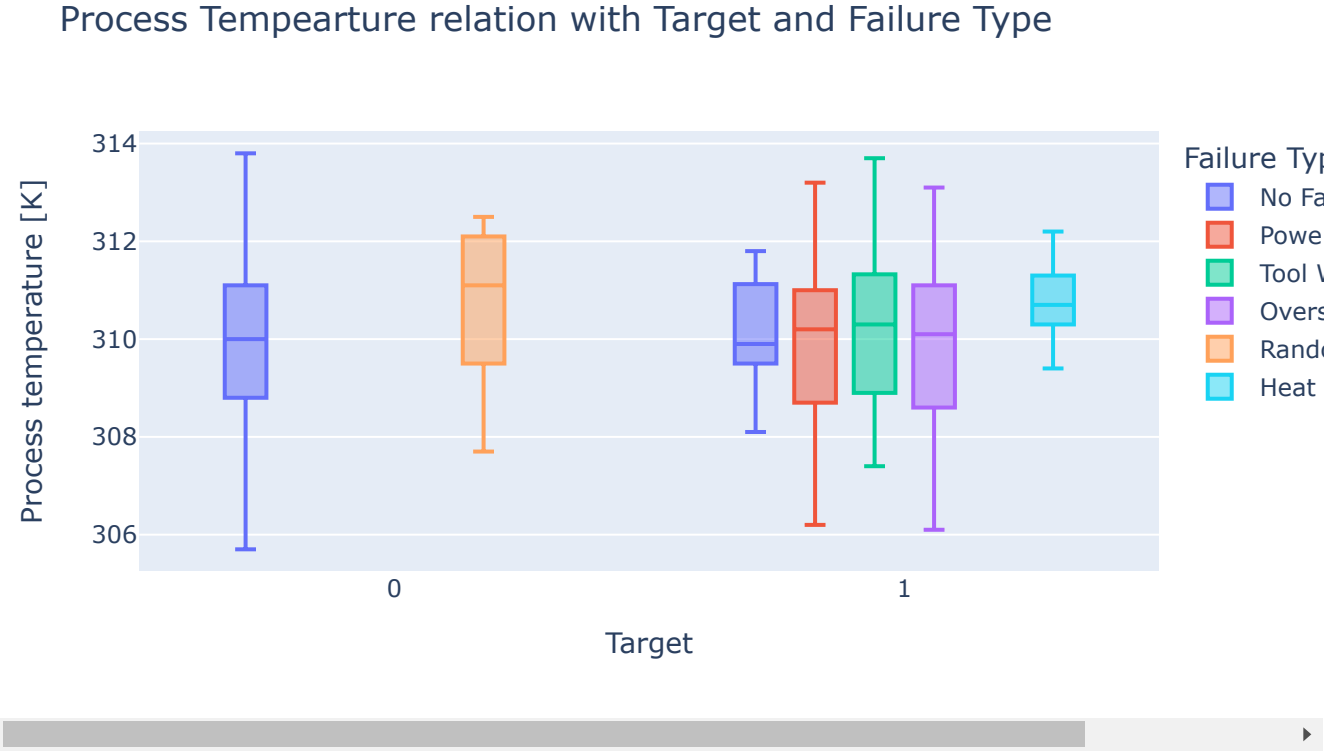


The dataset is highly imbalanced where the machine failure consist only 3.5% of the whole dataset. Next, Box plot are generated to observe the relationship between categorical features with the Target and Failure Type Here, q1 : 25th Percentile / the middle value between the median and the lowest value q3 : 75th Percentile / the middle value between the median and the highest value. Interquartile range (IQR): the difference from q3 to q1

```
# Air Temperature relation with Target/Failure Type
fig = px.box(data,
             y      =  "Air temperature [K]",
             x      =  "Target",
             title  =  "Air Temperature relation with Target and Failure Type",
             color  =  "Failure Type",
             width  =  800,
             height =  400)
fig.show()
```

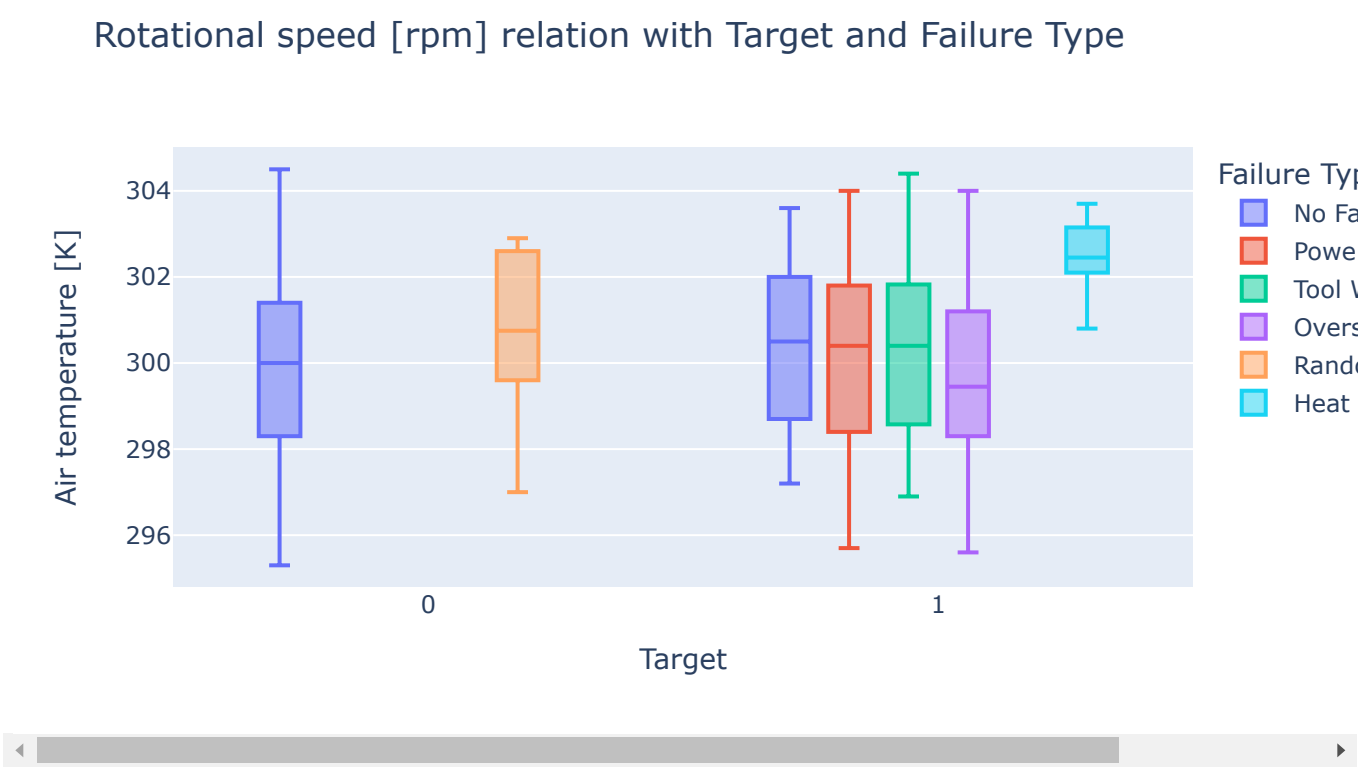



```
# Process Tempearture relation with Target/Failure Type
fig = px.box(data,
             y      =  "Process temperature [K]",
             x      =  "Target",
             title  =  "Process Tempearture relation with Target and Failure Type",
             color  =  "Failure Type",
             width  =  800,
             height =  400)
fig.show()
```

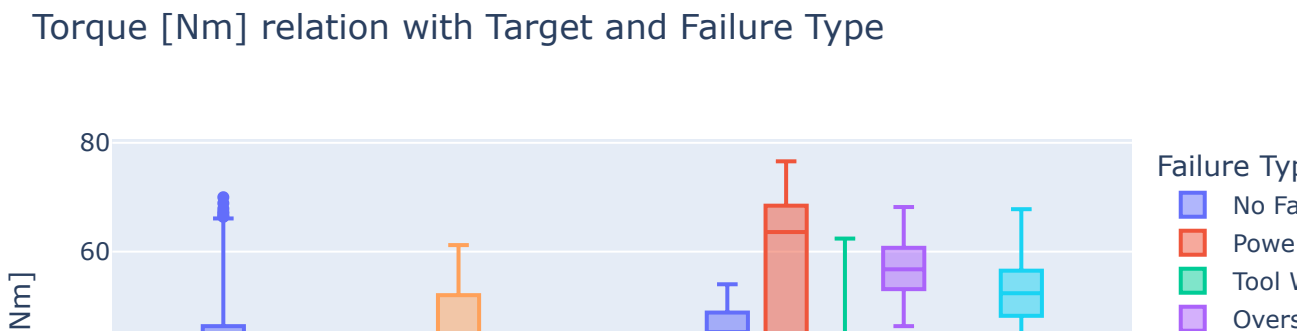



```
# Rotational speed [rpm] relation with Target/Failure Type
fig = px.box(data,
             y      =  "Air temperature [K]",
             x      =  "Target",
             title  =  "Rotational speed [rpm] relation with Target and Failure Type",
```

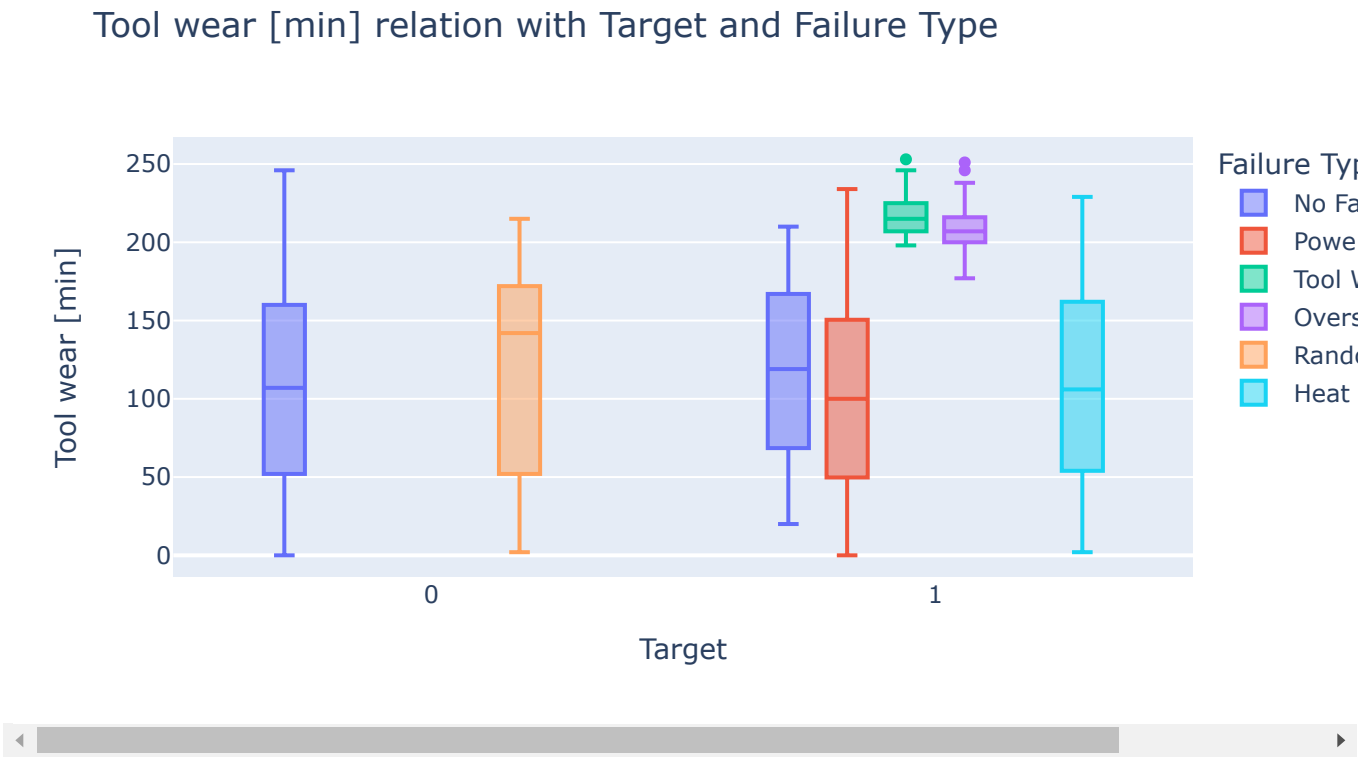
```
              color  =  "Failure Type",
              width  =  800,
              height =  400)
fig.show()
```




```
# Torque [Nm] relation with Target/Failure Type
fig = px.box(data,
              y      =  "Torque [Nm]",
              x      =  "Target",
              title  =  "Torque [Nm] relation with Target and Failure Type",
              color  =  "Failure Type",
              width  =  800,
              height =  400)
fig.show()
```




```
# Tool wear [min] relation with Target/Failure Type
fig = px.box(data,
             y      =  "Tool wear [min]",
             x      =  "Target",
             title  =  "Tool wear [min] relation with Target and Failure Type",
             color  =  "Failure Type",
             width  =  800,
             height =  400)
fig.show()
```



Now, we will try to observe correlation of certain features with Failure Type using pandas Pivot Table. The pivot table takes simple column-wise data as input, and groups the entries into a two-dimensional table that provides a multidimensional summarization of the data. In short, PivotTable is an interactive way to quickly summarize large amounts of data.

```
# Correlation with Product ID with Failure
pd.pivot_table(data,
               index   = 'Failure Type',
               columns = 'Type',
               aggfunc ='count')
```

	Air temperature [K]			Process temperature [K]			Rotational speed [rpm]			Target			Tool we	
Type	H	L	M	H	L	M	H	L	M	H	L	M	H	L
Failure Type														
Heat Dissipation Failure	8	74	30	8	74	30	8	74	30	8	74	30	8	
No Failure	979	5757	2916	979	5757	2916	979	5757	2916	979	5757	2916	979	57
Overstrain Failure	1	73	4	1	73	4	1	73	4	1	73	4	1	

```
pd.pivot_table(data,
               index   = 'Target',
               columns = 'Type',
               aggfunc ='count')
```

	Air temperature [K]			Failure Type			Process temperature [K]			Rotational speed [rpm]			Tool wear [	
Type	H	L	M	H	L	M	H	L	M	H	L	M	H	L
Target														
0	982	5765	2914	982	5765	2914	982	5765	2914	982	5765	2914	982	5765

From, these two tables we can see that machine Type L has higher tendency to fail.

Data Preprocessing for Prediction

Before using ML model the data is processed again in 3 steps Encoding categorical features Splitting test & train data Feature Scaling Encoding refers to converting the labels into a numeric form so as to convert them into the machine-readable form.We use it on our training data, and using "t" it will gure out the unique values and assign a value to it, returns the encoded labels.

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
label_encoder.fit(data['Type'])
```

```
data['Type'] = label_encoder.transform(data['Type'])
label_encoder.fit(data['Target'])
data['Target'] = label_encoder.transform(data['Target'])
```

Practical 5

To develop Manufacturing Cost Model

Aim: To develop Manufacturing Cost Model

Prerequisites: Maintenance data, Jupyter Notebook / Google Colab

Theory:

The objective of manufacturing cost models is to predict the manufacturing cost of a product or system. The manufacturing cost estimate is a function of application-specific details (e.g., the size of the product), technology/material details (e.g., the technology and materials required to meet thermal and electrical performance requirements), processing details (e.g., the manufacturing facilities available to fabricate and assemble the product), and accounting realities (e.g., applicable labor and overhead rates). Manufacturing costs include three basic activities: fabrication and/or assembly, recurring functional test/inspection, and diagnosis and rework (when relevant)…

Let's assume that you work as a consultant to a start-up company that was looking to develop a model to estimate the cost of goods sold as they vary the production volume (number of units produced). The startup gathered data and has asked you to develop a model to predict its cost vs. the number of units sold.

Number of Units	Manufacturing Cost
1	95.06605578
1.185993649	96.53174997
1.19149864	73.66131056
1.204771398	95.5668425
1.298772823	98.77701266
1.307435033	100
1.339385699	94.75975637
1.379043602	67.18538349
1.419999515	72.88604061
1.473948344	61.96769605
1.540898452	69.28409692
1.574599858	64.55249643
1.620309789	77.67937742
1.631997227	58.42664546
1.65268704	51.4409698
1.695801032	60.92903611
1.704214274	81.86775617
1.739201389	60.5725967
1.760146268	74.12260141
1.767001878	71.61417311
1.777280346	77.98340868
1.784341505	57.68193975
1.802089167	52.96440828
1.815917415	69.1787313
1.82328911	70.42005203
1.831097216	36.08580039
1.832406608	81.49469151
1.842875742	59.64067983
1.845840644	63.40954945
1.865227752	55.03613926
1.87414418	58.06045968
1.895730774	52.11412489
1.943028599	61.27727337
1.987770404	62.5960052
1.991902366	61.23299391
2.004803791	60.02154297
2.052316741	47.88555562
2.064920557	54.90274827
2.07583577	54.17817589
2.105273495	60.21890139
2.113298687	44.71285113
2.118589549	45.64983585
2.141908761	57.80822022
2.151590015	52.02926301
2.17131692	54.10994815
2.189511901	73.77810164

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load
!pip install xlrd==1.2.0
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns  #visualization
import matplotlib.pyplot as plt

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files und

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting xlrd==1.2.0
  Downloading xlrd-1.2.0-py2.py3-none-any.whl (103 kB)
     |████████████████████████████████| 103 kB 5.3 MB/s
Installing collected packages: xlrd
  Attempting uninstall: xlrd
    Found existing installation: xlrd 1.1.0
    Uninstalling xlrd-1.1.0:
      Successfully uninstalled xlrd-1.1.0
Successfully installed xlrd-1.2.0
```

```
from google.colab import drive
drive.mount('/content/drive',force_remount=True)
data = pd.read_excel('/content/EconomiesOfScale.xlsx')
```

```
Mounted at /content/drive
```

```
data.head()
```

	Number of Units	Manufacturing Cost

```
data.describe()
```

	Number of Units	Manufacturing Cost
count	1000.000000	1000.000000
mean	4.472799	40.052999
std	1.336241	10.595322
min	1.000000	20.000000
25%	3.594214	32.912036
50%	4.435958	38.345781
75%	5.324780	44.531822
max	10.000000	100.000000

```
nans=pd.isnull(data).sum()
nans[nans>0]
```

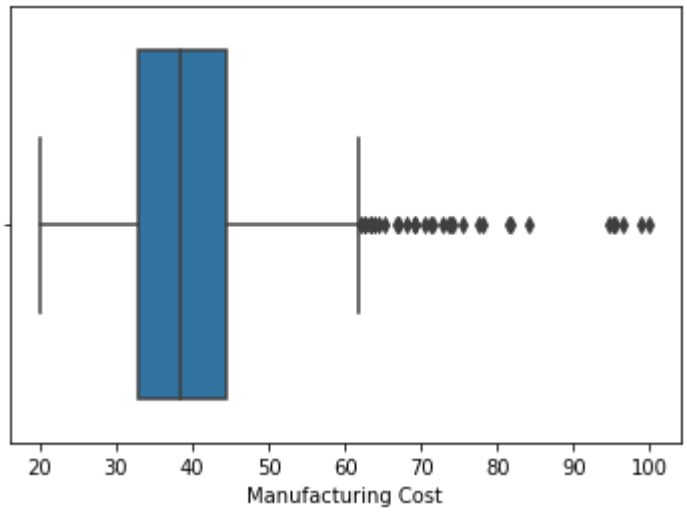
```
Series([], dtype: int64)
```

```
data.shape[0]
```

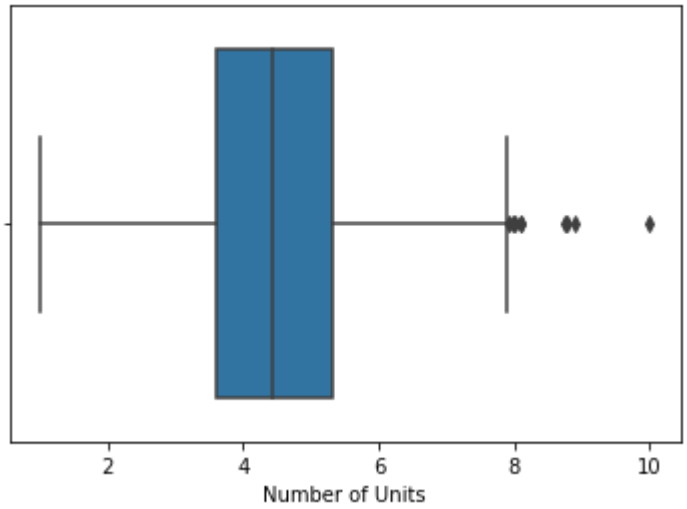
```
1000
```

Let's carry out some visualizations using Seaborn

```
ax = sns.boxplot(x=data["Manufacturing Cost"])
```

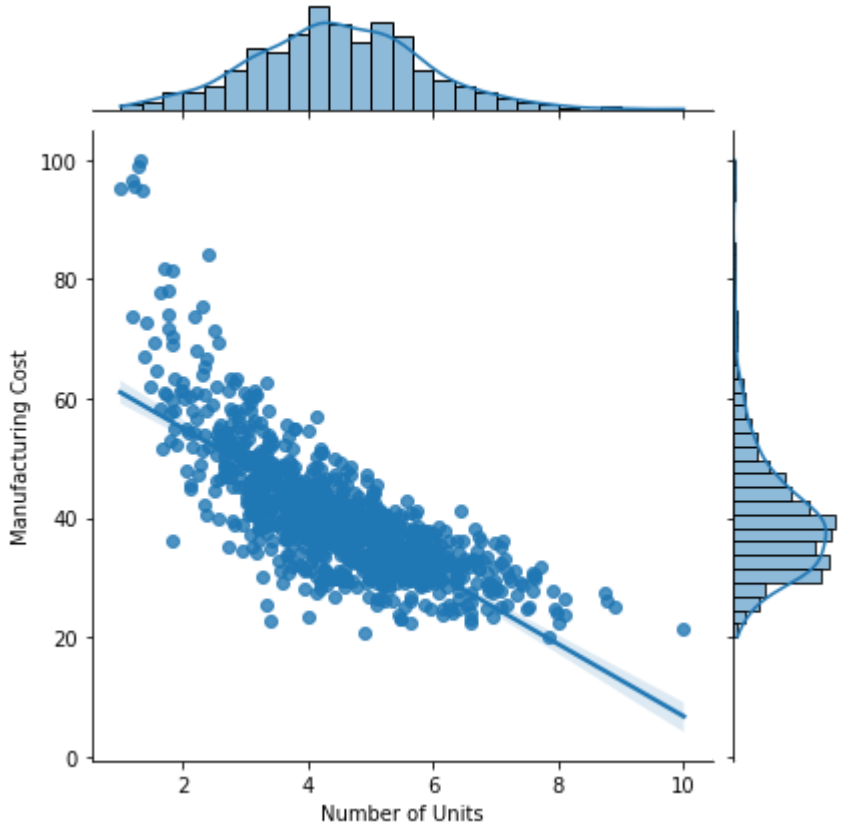


```
ax = sns.boxplot(x=data['Number of Units'])
```




```
sns.jointplot(data=data, x="Number of Units", y="Manufacturing Cost", kind="reg")
```

```
<seaborn.axisgrid.JointGrid at 0x7ff18e2c63d0>
```



Let's build model

```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
scaled_data = sc.fit_transform(data)
scaled_data
```

```
array([[0.        , 0.9383257 ],
       [0.02066596, 0.95664687],
       [0.02127763, 0.67076638],
       ...,
       [0.86454312, 0.07467234],
       [0.87752219, 0.06422889],
       [1.        , 0.01934721]])
```

```
scaled_data = pd.DataFrame(scaled_data, columns = ['Number of Units', 'Manufacturing Cost'])
```

```
scaled_data
```

	Number of Units	Manufacturing Cost
0	0.000000	0.938326
1	0.020666	0.956647
2	0.021278	0.670766
3	0.022752	0.944586
4	0.033197	0.984713
...	...	...
995	0.788857	0.048188
996	0.859972	0.094207
997	0.864543	0.074672
998	0.877522	0.064229
999	1.000000	0.019347

1000 rows × 2 columns

```
y = scaled_data.pop('Manufacturing Cost')
y
```

```
0      0.938326
1      0.956647
2      0.670766
3      0.944586
4      0.984713
         ...
995    0.048188
996    0.094207
997    0.074672
998    0.064229
999    0.019347
Name: Manufacturing Cost, Length: 1000, dtype: float64
```

```
X = scaled_data.values


"""
Split the dataset into training set and test set with an 80-20 ratio
"""
from sklearn.model_selection import train_test_split
seed=1
X_train, X_test, \
y_train, y_test = train_test_split(X, y, test_size=0.2, \
                                   random_state=42)


from sklearn.svm import SVR


regr= SVR(C=1.0, epsilon=0.2)
regr.fit(X_train,y_train.ravel())
```

```
SVR(epsilon=0.2)
```

```
y_pred = regr.predict(X_test)
y_pred
```

```
array([0.23476998, 0.19564829, 0.19552012, 0.20350118, 0.25761209,
       0.20152073, 0.20868388, 0.23718671, 0.20304453, 0.35180452,
       0.19570875, 0.41292374, 0.20709024, 0.23183355, 0.22813452,
       0.21300845, 0.28898906, 0.20898637, 0.19462914, 0.27781544,
       0.22914077, 0.33378322, 0.26540822, 0.23374456, 0.317575  ,
       0.30691577, 0.38566758, 0.22948237, 0.21358697, 0.22992041,
       0.27087766, 0.35108264, 0.20962335, 0.24059563, 0.26569672,
       0.32181694, 0.20065724, 0.22099506, 0.21344844, 0.45573471,
       0.27529935, 0.38878804, 0.27968523, 0.23177926, 0.28450419,
       0.29023651, 0.22285173, 0.2161231 , 0.25313191, 0.19790806,
       0.2250116 , 0.20793397, 0.29438255, 0.22256105, 0.60464413,
       0.58535064, 0.21132973, 0.70114225, 0.31394558, 0.19646868,
       0.28550935, 0.46230228, 0.23054964, 0.3178442 , 0.21526703,
       0.20025726, 0.20334417, 0.2047899 , 0.41879638, 0.23031656,
       0.37798776, 0.24097249, 0.22005137, 0.19537004, 0.28626906,
       0.28661788, 0.20441796, 0.5164567 , 0.22013279, 0.28129508,
       0.20147005, 0.42688768, 0.29052581, 0.42638852, 0.27804957,
       0.22929735, 0.22977071, 0.19732925, 0.19497802, 0.27538463,
       0.2633768 , 0.25082279, 0.23453376, 0.31433468, 0.19467478,
       0.25353211, 0.19456455, 0.39709234, 0.43353714, 0.19685745,
       0.1971824 , 0.26981381, 0.22775389, 0.29543255, 0.20733488,
       0.20571743, 0.22817938, 0.23200193, 0.33702665, 0.22039579,
       0.23372122, 0.19661119, 0.22950817, 0.19593826, 0.39894177,
       0.25449299, 0.32918061, 0.23097683, 0.2329515 , 0.28591771,
       0.2110974 , 0.25549309, 0.19736443, 0.29471497, 0.30469764,
       0.22748822, 0.22111173, 0.25231059, 0.2048678 , 0.19524356,
       0.21631618, 0.20047142, 0.29288796, 0.22982797, 0.23354053,
       0.46075801, 0.36576062, 0.31622165, 0.59955219, 0.41707837,
       0.49931476, 0.29957463, 0.19680618, 0.28884013, 0.21141338,
```

```
       0.21829503, 0.19566166, 0.30106666, 0.19659288, 0.27753212,
       0.20592251, 0.34307049, 0.23110011, 0.25478658, 0.22832641,
       0.24935498, 0.28069803, 0.19986757, 0.45245932, 0.20906462,
       0.21711763, 0.19448755, 0.20520075, 0.205105  , 0.20446054,
       0.20282835, 0.2849949 , 0.20061093, 0.27949631, 0.28022113,
       0.26771314, 0.24630999, 0.37452241, 0.2290563 , 0.2430412 ,
       0.26757145, 0.29705325, 0.2948896 , 0.19535961, 0.20187177,
       0.24093193, 0.20266503, 0.26391159, 0.2341701 , 0.19928893,
       0.35138871, 0.26921077, 0.26709635, 0.22872646, 0.19508277,
       0.23028492, 0.19709374, 0.20392945, 0.32127952, 0.31694141,
       0.25830457, 0.27435644, 0.31842808, 0.21207435, 0.40956217])
```

```
from sklearn.metrics import r2_score,mean_squared_error
```

```
mse = mean_squared_error(y_test,y_pred)
rmse= np.sqrt(mse)
rmse
```

```
0.0822903295452384
```

```
mse
```

```
0.006771698336663936
```

```
r2_score(y_test, y_pred)
```

```
0.4700318323625766
```

The R-square value is approximately 0.5 which is quite encouraging. We can improve this work by trying out algorithms as well.

Practical 6

To perform Thermal Analysis from IOT Devices temperature readings

Aim: To perform Thermal Analysis from IOT Devices temperature readings.

Prerequisites: IOT Temp data, Jupyter Notebook / Google Colab

Theory:

`IIoT 4.0` is coming to cover all enterprise monitoring and maintenance system. Thus, we need bold and sustainable algorithms and approaches to *analyze the IOT sensor data* and find hidden patterns and insights. Heat Index (*temperature + humidity*) is one common data recorded on these IOT readers. The frequency of the upcoming data is very fast. The sensor reads *hundreds to millions of data per second*. There is a huge and versatile application of this data in real world. like:- Agriculture, weather forecasting, soil monitoring and treatment, enterprise maintenance, Data centres, and many more…

This dataset contains the temperature readings from IOT devices installed outside and inside of an anonymous Room (say - admin room). The device was in the alpha testing phase. So, It was uninstalled or shut off several times during the entire reading period (28-07-2018 to 08-12-2018). This random interval recordings and few mis-readings (outliers) makes it more challanging to perform analysis on this data. Let's see, what you can present in the plate out of this messy data.

Technical Details:
columns = 5 | Rows = 97605
`id` : unique IDs for each reading
`room_id/id` : room id in which device was installed (inside and/or outside) -> currently 'admin room' only for example purpose.
`noted_date` : date and time of reading
`temp` : temperature readings
`out/in` : whether reading was taken from device installed inside or outside of room?

From this dataset , it would be intersting to find out:

- what was the max and min temperature?
- How outside temperature was related to inside temperature? any relation between the two?
- What was the variance of temperature for inside and outside room temperature?
- What is the trend in the data?
- Can you use Time Series Forecast algo to predict the next scenario?
- which was the hottest/coolest month ?
- any warning signals fro climate disaster ?
- and many more…

id	room_id/id	noted_date	temp	out/in
__export_	Room Admin	8/12/2018 9:30	29	In
__export_	Room Admin	8/12/2018 9:30	29	In
__export_	Room Admin	8/12/2018 9:29	41	Out
__export_	Room Admin	8/12/2018 9:29	41	Out
__export_	Room Admin	8/12/2018 9:29	31	In
__export_	Room Admin	8/12/2018 9:29	31	In
__export_	Room Admin	8/12/2018 9:28	29	In
__export_	Room Admin	8/12/2018 9:28	29	In
__export_	Room Admin	8/12/2018 9:26	29	In
__export_	Room Admin	8/12/2018 9:26	29	In
__export_	Room Admin	8/12/2018 9:25	42	Out
__export_	Room Admin	8/12/2018 9:25	42	Out
__export_	Room Admin	8/12/2018 9:24	29	In
__export_	Room Admin	8/12/2018 9:24	29	In
__export_	Room Admin	8/12/2018 9:22	29	In
__export_	Room Admin	8/12/2018 9:22	29	In
__export_	Room Admin	8/12/2018 9:21	41	Out
__export_	Room Admin	8/12/2018 9:21	41	Out
__export_	Room Admin	8/12/2018 9:20	29	In
__export_	Room Admin	8/12/2018 9:20	29	In
__export_	Room Admin	8/12/2018 9:19	42	Out
__export_	Room Admin	8/12/2018 9:19	42	Out
__export_	Room Admin	8/12/2018 9:18	29	In
__export_	Room Admin	8/12/2018 9:18	29	In
__export_	Room Admin	8/12/2018 9:17	41	Out
__export_	Room Admin	8/12/2018 9:17	41	Out
__export_	Room Admin	8/12/2018 9:16	29	In
__export_	Room Admin	8/12/2018 9:16	29	In
__export_	Room Admin	8/12/2018 9:14	29	In
__export_	Room Admin	8/12/2018 9:14	29	In
__export_	Room Admin	8/12/2018 9:12	29	In
__export_	Room Admin	8/12/2018 9:12	29	In
__export_	Room Admin	8/12/2018 9:09	42	Out
__export_	Room Admin	8/12/2018 9:09	42	Out
__export_	Room Admin	8/12/2018 9:08	29	In
__export_	Room Admin	8/12/2018 9:08	29	In
__export_	Room Admin	8/12/2018 9:07	41	Out
__export_	Room Admin	8/12/2018 9:07	41	Out
__export_	Room Admin	8/12/2018 9:05	29	In
__export_	Room Admin	8/12/2018 9:04	42	Out
__export_	Room Admin	8/12/2018 9:04	30	In
__export_	Room Admin	8/12/2018 9:04	29	In
__export_	Room Admin	8/12/2018 9:03	42	Out
__export_	Room Admin	8/12/2018 9:03	30	In
__export_	Room Admin	8/12/2018 9:01	30	In
__export_	Room Admin	8/12/2018 9:00	29	In
__export_	Room Admin	8/12/2018 9:00	29	In
__export_	Room Admin	8/12/2018 8:58	29	In
__export_	Room Admin	8/12/2018 8:58	29	In

Temperature_Analysis

Step 1: Importing Liabraries

```
!pip install xlrd==1.2.0
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import os
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting xlrd==1.2.0
  Downloading xlrd-1.2.0-py2.py3-none-any.whl (103 kB)
     |████████████████████████████████| 103 kB 17.0 MB/s
Installing collected packages: xlrd
  Attempting uninstall: xlrd
    Found existing installation: xlrd 1.1.0
    Uninstalling xlrd-1.1.0:
      Successfully uninstalled xlrd-1.1.0
Successfully installed xlrd-1.2.0
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
---------------------------------------------------------------------------
MessageError                                Traceback (most recent call last)
<ipython-input-6-21dc3c638f66> in <module>
      1 from google.colab import files
----> 2 uploaded = files.upload()

3 frames
/usr/local/lib/python3.7/dist-packages/google/colab/_message.py in
read_reply_from_input(message_id, timeout_sec)
    100         reply.get('colab_msg_id') == message_id):
    101       if 'error' in reply:
--> 102         raise MessageError(reply['error'])
    103       return reply.get('data', None)
    104

MessageError: TypeError: Cannot read properties of undefined (reading '_uploadFiles')
```

Step 2: Importing Dataset and Read insights

```
data = pd.read_csv("/content/IOT-temp.csv")
data.sample(5)
```

	id	room_id/id	noted_date	temp	out/in
88379	__export__.temp_log_92096_185c6f2c	Room Admin	07-09-2018 14:17	28	In
62842	__export__.temp_log_121623_31ce9b10	Room Admin	12-09-2018 01:10	27	Out
7985	__export__.temp_log_170444_68b89df4	Room Admin	30-11-2018 07:50	42	Out
40697	__export__.temp_log_54025_f910bbd5	Room Admin	17-10-2018 14:04	40	Out
3289	__export__.temp_log_186152_08633488	Room Admin	05-12-2018 17:49	35	Out

```
data.head()
```

	id	room_id/id	noted_date	temp	out/in
0	__export__.temp_log_196134_bd201015	Room Admin	08-12-2018 09:30	29	In
1	__export__.temp_log_196131_7bca51bc	Room Admin	08-12-2018 09:30	29	In
2	__export__.temp_log_196127_522915e3	Room Admin	08-12-2018 09:29	41	Out
3	__export__.temp_log_196128_be0919cf	Room Admin	08-12-2018 09:29	41	Out
4	__export__.temp_log_196126_d30b72fb	Room Admin	08-12-2018 09:29	31	In

```
data.tail()
```

	id	room_id/id	noted_date	temp	out/in
97601	__export__.temp_log_91076_7fbd08ca	Room Admin	28-07-2018 07:07	31	In
97602	__export__.temp_log_147733_62c03f31	Room Admin	28-07-2018 07:07	31	In
97603	__export__.temp_log_100386_84093a68	Room Admin	28-07-2018 07:06	31	In
97604	__export__.temp_log_123297_4d8e690b	Room Admin	28-07-2018 07:06	31	In
97605	__export__.temp_log_133741_32958703	Room Admin	28-07-2018 07:06	31	In

```
print("Shape of our data is : ",data.shape)
```

```
Shape of our data is :  (97606, 5)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 97606 entries, 0 to 97605
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
```

```
---  ------      --------------  -----
 0   id          97606 non-null  object
 1   room_id/id  97606 non-null  object
 2   noted_date  97606 non-null  object
 3   temp        97606 non-null  int64
 4   out/in      97606 non-null  object
dtypes: int64(1), object(4)
memory usage: 3.7+ MB
```

```
print("Unique values in every column \n"+'-'*25)
for i in data.columns:
    print("\t"+i+" = ",len(set(data[i])))
```

```
Unique values in every column
-------------------------
        id =  97605
        room_id/id =  1
        noted_date =  27920
        temp =  31
        out/in =  2
```

```
data.describe()
```

	temp
count	97606.000000
mean	35.053931
std	5.699825
min	21.000000
25%	30.000000
50%	35.000000
75%	40.000000
max	51.000000

Step 3: Data-Preprocessing

```
df = data.drop(['room_id/id'],axis=1)
df.head()
```

	id	noted_date	temp	out/in
0	__export__.temp_log_196134_bd201015	08-12-2018 09:30	29	In
1	__export__.temp_log_196131_7bca51bc	08-12-2018 09:30	29	In
3	__export__.temp_log_196128_be0919cf	08-12-2018 09:29	41	Out

Step 4: Data Analysis

```
# Check for Missing Values
data.isnull().sum()
```

```
id            0
room_id/id    0
noted_date    0
temp          0
out/in        0
dtype: int64
```

```
## Separate the date and time
date=[]
time=[]
for i in df['noted_date']:
    date.append(i.split(' ')[0])
    time.append(i.split(' ')[1])
df['date']=date
df['time']=time
df.drop('noted_date',axis=1,inplace=True)
df.head()
```

	id	temp	out/in	date	time
0	__export__.temp_log_196134_bd201015	29	In	08-12-2018	09:30
1	__export__.temp_log_196131_7bca51bc	29	In	08-12-2018	09:30
2	__export__.temp_log_196127_522915e3	41	Out	08-12-2018	09:29
3	__export__.temp_log_196128_be0919cf	41	Out	08-12-2018	09:29
4	__export__.temp_log_196126_d30b72fb	31	In	08-12-2018	09:29

```
df[['outside','inside']]=pd.get_dummies(df['out/in'])
df.rename(columns = {'out/in':'location'}, inplace = True)
print('Total Inside Observations  :',len([i for i in df['inside'] if  i == 1]))
print('Total Outside Observations :',len([i for i in df['inside'] if  i == 0]))
```

```
Total Inside Observations  : 77261
Total Outside Observations : 20345
```

```
## Let's separate date further into days,months and year
try:
```

```
    df['date'] = pd.to_datetime(df['date'])
    df['year'] = df['date'].dt.year
    df['month'] = df.date.dt.month
    df['day'] = df.date.dt.day
    df.drop('date',axis=1,inplace=True)
except:
 print('Operations already performed')
df.head()
```

	id	temp	location	time	outside	inside	year	mont
0	__export__.temp_log_196134_bd201015	29	In	09:30	1	0	2018	
1	__export__.temp_log_196131_7bca51bc	29	In	09:30	1	0	2018	
2	__export__.temp_log_196127_522915e3	41	Out	09:29	0	1	2018	
3	__export__.temp_log_196128_be0919cf	41	Out	09:29	0	1	2018	
4	__export__.temp_log_196126_d30b72fb	31	In	09:29	1	0	2018	

```
print("Days of observation   : ",sorted(df['day'].unique()))
print("Months of observation : ",sorted(df['month'].unique()))
print("Year of observation   : ",sorted(df['year'].unique()))
```

```
Days of observation   :  [8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23,
Months of observation :  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
Year of observation   :  [2018]
```

```
print ("Temperature")
print("\tTotal Count    = ",df['temp'].shape[0])
print("\tMinimum Value  = ",df['temp'].min())
print("\tMaximum Value  = ",df['temp'].max())
print("\tMean Value     = ",df['temp'].mean())
print("\tStd dev Value  = ",df['temp'].std())
print("\tVariance Value = ",df['temp'].var())
```

```
Temperature
        Total Count    =  97606
        Minimum Value  =  21
        Maximum Value  =  51
        Mean Value     =  35.05393111079237
        Std dev Value  =  5.699825337585307
        Variance Value =  32.48800887897946
```

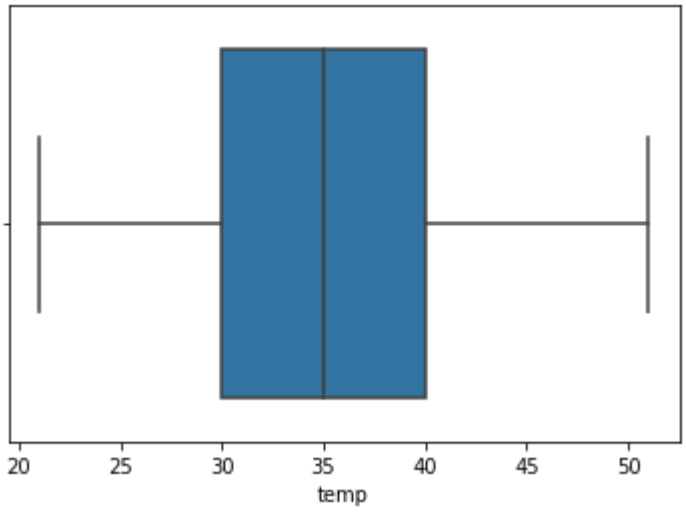
```
## Reassemble whole dataframe and print the new detailed dataframe
df = df[['day','month','year','time','temp','location','outside','inside']]
df.head()
```

	day	month	year	time	temp	location	outside	inside
0	12	8	2018	09:30	29	In	1	0
1	12	8	2018	09:30	29	In	1	0
2	12	8	2018	09:29	41	Out	0	1
3	12	8	2018	09:29	41	Out	0	1
4	12	8	2018	09:29	31	In	1	0

Step 5: Data Visualation

```
sns.boxplot(df['temp'])
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
```

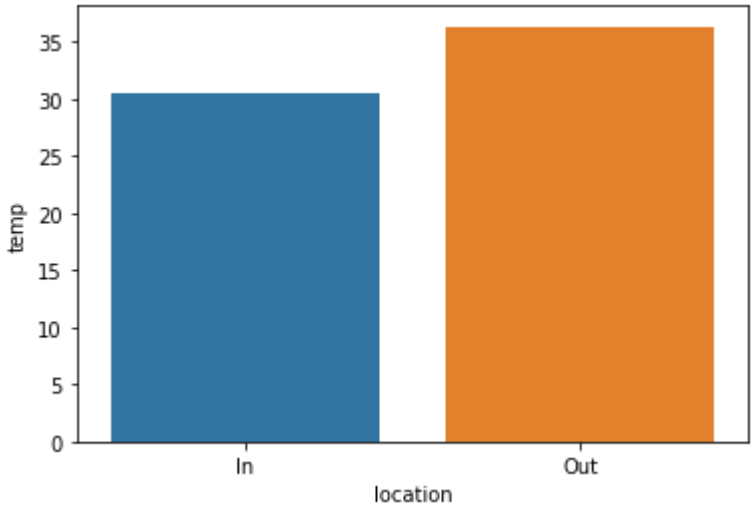



```
sns.countplot(df['location'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f858e94d950>
```

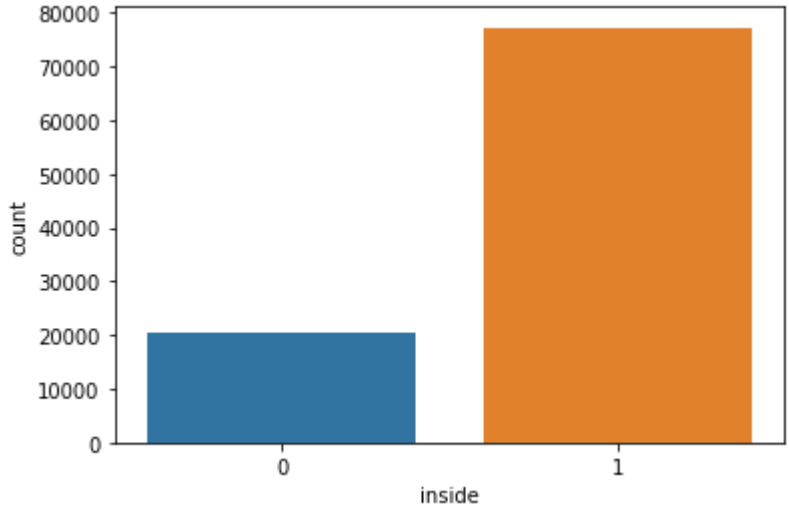
```
sns.barplot(df['location'],df['temp'])
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
```



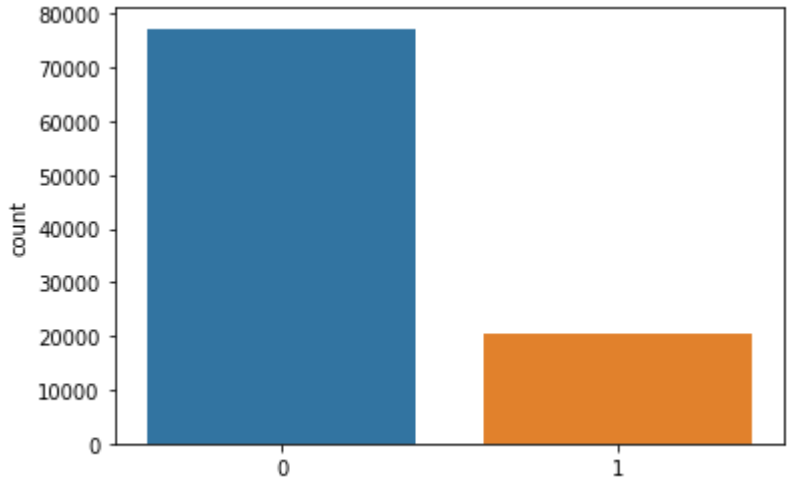

```
sns.countplot(df['inside'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f858e94d610>
```



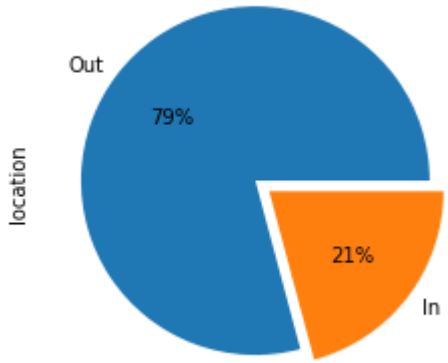

```
sns.countplot(df['outside'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f858e3e99d0>
```



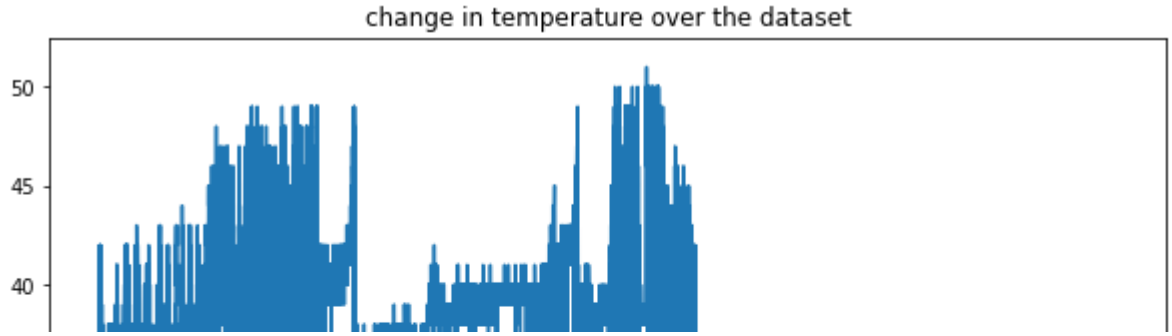

```
df['location'].value_counts().plot.pie(explode=[0,0.1],  autopct='%1.0f%%')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f858e335750>
```

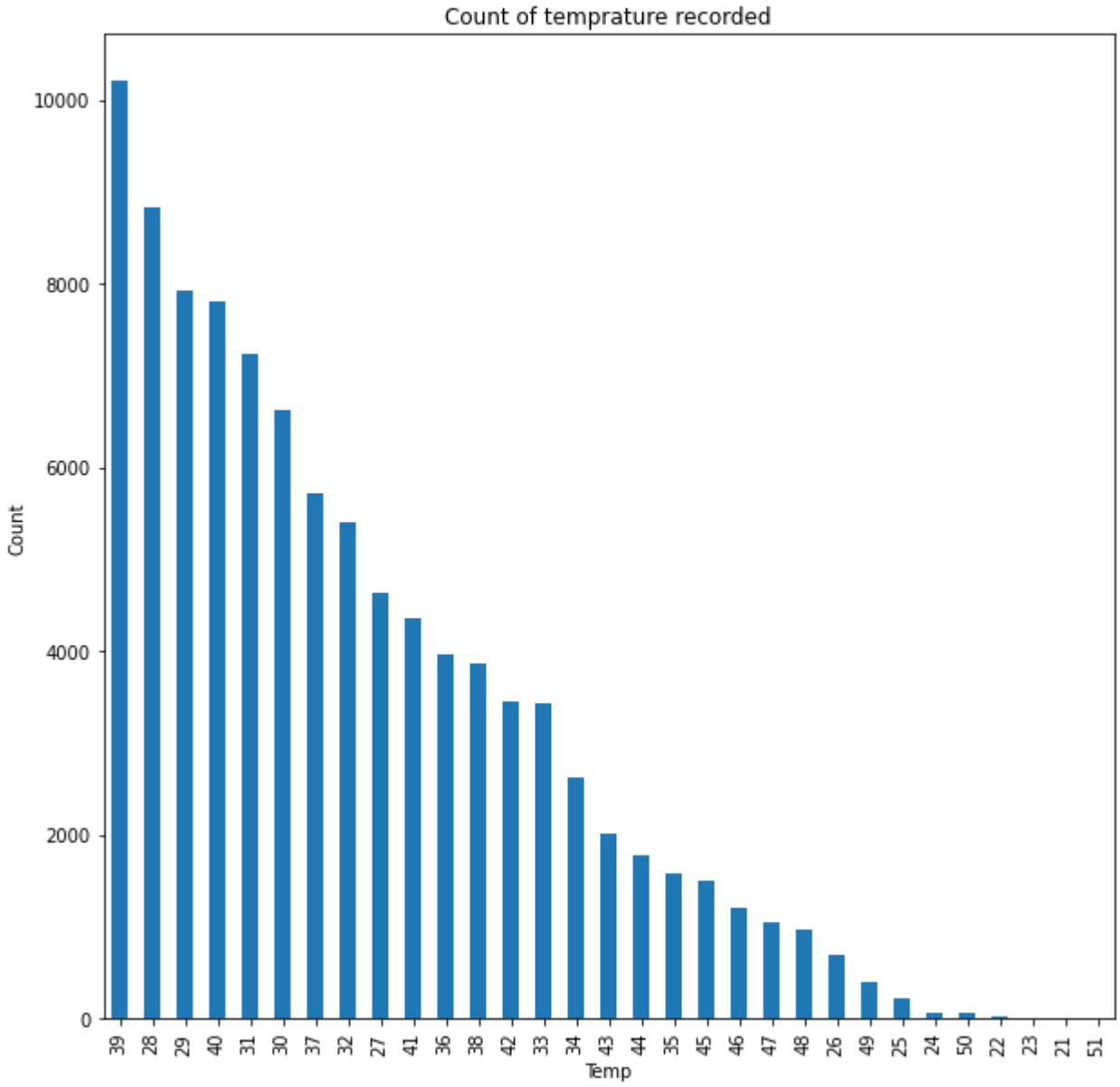



```
plt.figure(figsize=(10,6))
plt.plot(data['temp'])
plt.ylabel('temp')
plt.title('change in temperature over the dataset')
```

```
Text(0.5, 1.0, 'change in temperature over the dataset')
```

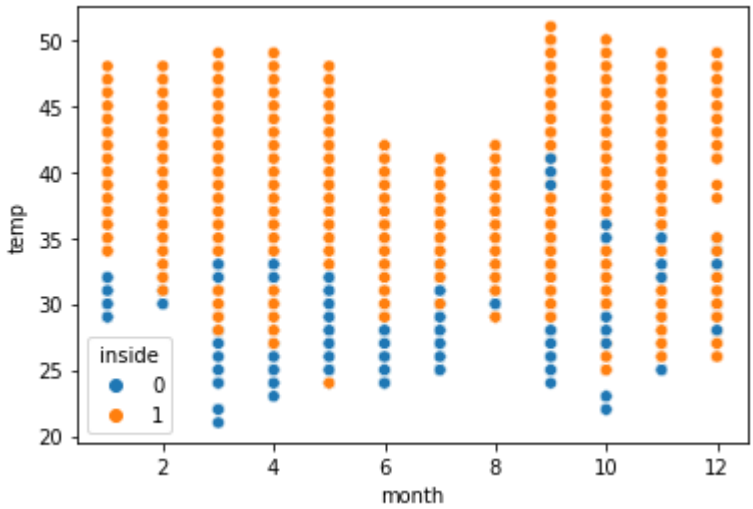



```
# histogram of the various values of temp recorded
plt.figure(figsize=(10,10))
data['temp'].value_counts().plot.bar()
plt.xlabel('Temp')
plt.ylabel('Count')
plt.title('Count of temprature recorded')
plt.show()
```



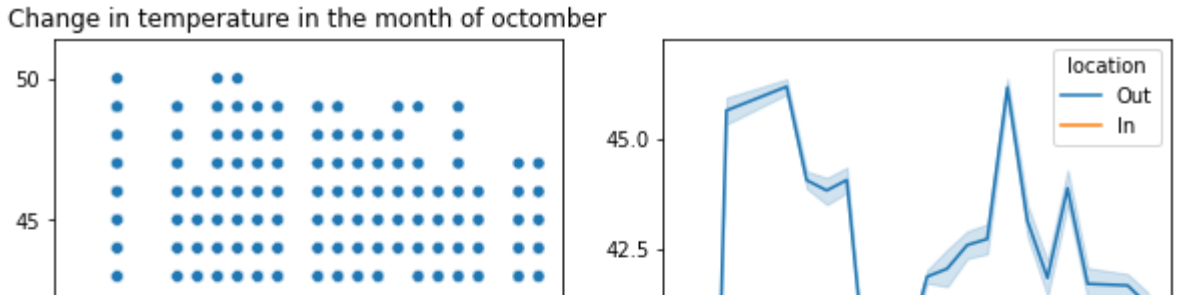
```
sns.scatterplot(df['month'],df['temp'],hue=df['inside'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f858e2cff50>
```



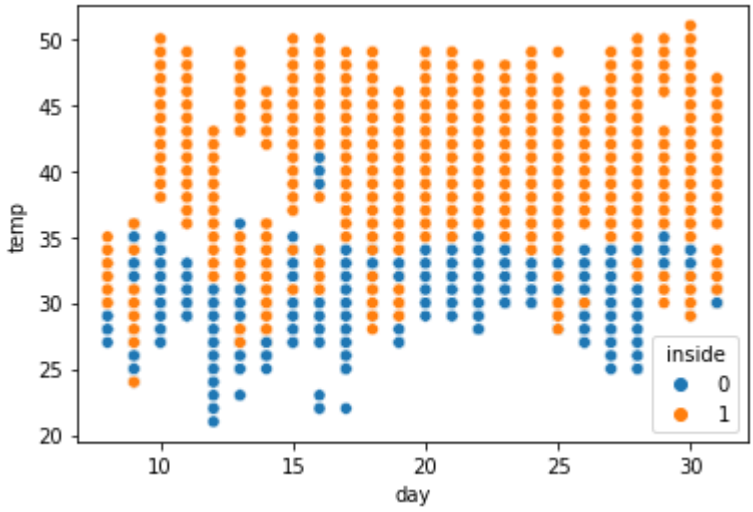

```
df['month'].value_counts()
x = df[df['month'] == 10]
f, ax = plt.subplots(1,2, figsize=(10,8))
sns.scatterplot(x['day'], x['temp'], hue=x['location'], ax=ax[0])
sns.lineplot(x['day'], x['temp'], hue=x['location'], ax=ax[1])
plt.xlabel('days')
plt.ylabel('temp')
ax[0].set_title('Change in temperature in the month of octomber')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
Text(0.5, 1.0, 'Change in temperature in the month of octomber')
```



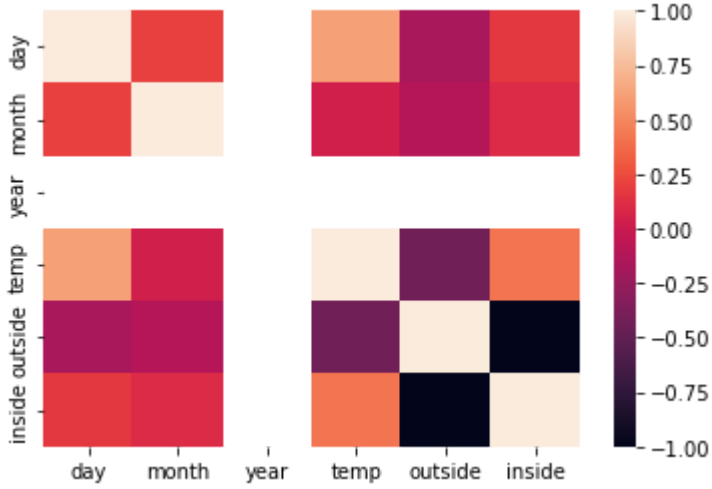

```
sns.scatterplot(df['day'],df['temp'],hue=df['inside'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f858d74f610>
```



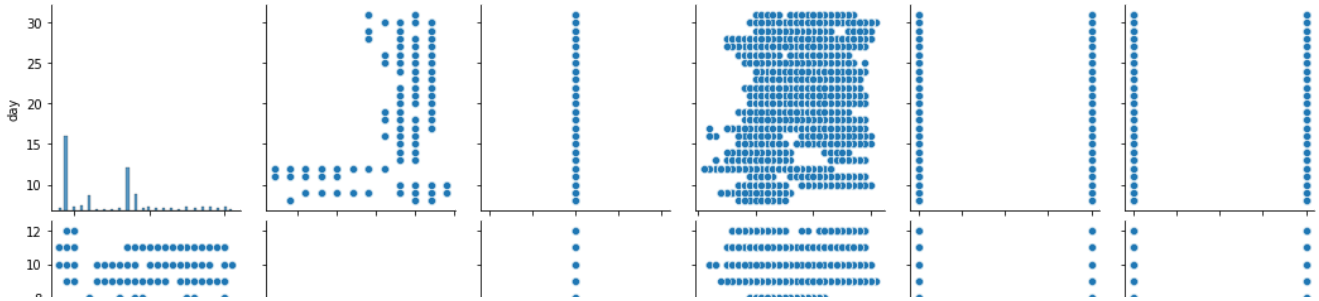

```
sns.heatmap(df.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f858dc48750>
```



```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f858dc42610>
```




```
## Tasks
arr = df['inside']
x=[]
y=[]
for i in arr:
    if i==1:
        x.append(i)
    else :
        y.append(i)
x=pd.Series(x)
y=pd.Series(y)
type(arr)
```

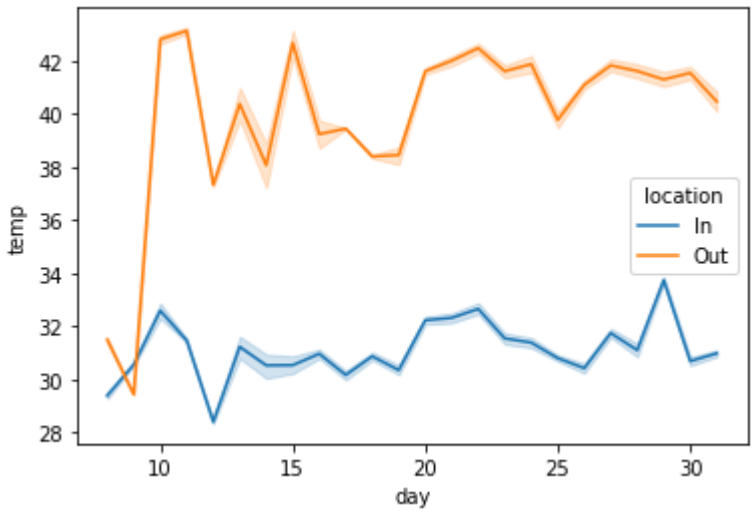
```
pandas.core.series.Series
```


```
## Variance of temp for inside - outside room temp ?
##   Outcome : The temperature outside has larger variance than inside temperature.
fig,axes = plt.subplots(1,3,figsize=(18,5))
sns.violinplot(x,df['temp'],ax=axes[0],color='b').set_title("Inside v/s Temp")
sns.violinplot(y,df['temp'],ax=axes[1],color='r').set_title("Outside v/s Temp")
sns.violinplot(df['location'],df['temp'],ax=axes[2]).set_title("Location v/s Temp")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
Text(0.5, 1.0, 'Location v/s Temp')
```

```
## How outside temp was related to inside temp ?
## Outcome:  Inside temp is free from any variations in data so follows a flat/linear trend,
sns.lineplot(df['day'],df['temp'],hue=df['location'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f8588a65c90>
```



Practical 7

Case Study on

Energy Conservation & Management

International Journal of RF Technologies xx (20xx) x–xx
DOI 10.3233/RFT-171671
IOS Press

Big data analytics methodologies applied at energy management in industrial sector: A case study

Maurizio Bevilacqua[a], Filippo Emanuele Ciarapica[a], Claudia Diamantini[b], Domenico Potena[b] and Maurizio Bevilacqua[a,*]

[a]*Dipartimento di Ingegneria Industriale e Scienze Matematiche (DIISM), Università Politecnica delle Marche, Ancona, Italy*

[b]*Dipartimento di Ingegneria dell'Informazione (DII), Università Politecnica delle Marche, Ancona, Italy*

Abstract. In this work, a framework is developed to integrate IoT-based energy management and company's existing information systems. This framework is a multi-layer model that includes three layers: 1) data collection layer, 2) data management layer and 3) data analytics layer. In order to test the proposed approach and assess its impact on improving energy efficiency, a pilot study was carried out in an Italian manufacturing company. Several smart meters have been installed at machine level to collect energy consumption data in real time, and then this data have been analyzed and provided to decision makers to improve energy efficiency by integrating them in production management decisions. When a company aims at analyzing the energy characteristics of its production system, data provided by different sources and geographically dispersed repositories must be taken into consideration. These characteristics bring several problems to develop a data analytic architecture. In this paper, we propose a data analytic model for IoT, in order to integrate the data collected from different sources and to improve energy-aware decision-making. Improving the overall equipment effectiveness of machine tools will improve resource-efficiency and productivity in manufacturing and support the development of smart factories from an energy point of view.

Keywords: Internet of things, data analytics, data mining, energy management, big data management, industrial energy consumption

1. Introduction

Worldwide about 50% of the total electricity consumption is made in industry by conversion using electric motor-driven systems of workstations (Waide & Brunner, 2011). There have been significant efforts over the last decade to define appropriate standards and best practices and implement consistent energy management systems to increase and maintain energy savings. Company energy management systems seek approaches to reduce their energy consumption without declining the production outputs (Hadera et al., 2015). According to Vikhorev et al. (2013), companies should define an energy management framework to promote energy awareness in manufacturing processes.

In this context, this work aims at developing a framework to improve the energy efficiency of a production system using concepts of Internet of Thinks (IoT) and Data Analytics. Relationships between IoT and Data Analytics is described by Feller et al.

*Corresponding author: Prof. Ing. Maurizio Bevilacqua, Dipartimento di Ingegneria Industriale e Scienze Matematiche, Università Politecnica delle Marche, Via Brecce Bianche, 60100 – Ancona, Italy. Tel.: +39 071 2204874; Fax: +39 071 2204770; E-mail: m.bevilacqua@univpm.it.

(2015): IoT collects data from different sources (in this work from power and energy sensors embedded in workstations), Data Analytics is responsible for extracting patterns or generating models from the output of the data processing step and then feeding them into the decision-making step, which takes care of transforming its input into useful knowledge.

The first step towards reducing energy consumption in machine tools and manufacturing systems is to devise methods to understand and characterize their energy consumption (Herrman et al., 2007). For this reason, an energy monitoring system is necessary. IoT can provide useful tools in order to develop a more detailed analysis of machine consumption. Kees et al. (2015) define IoT as "the connectivity of physical objects or industrial products, equipped with sensors and actuators, to the Internet via data communication technology, enabling interaction with and/or among these objects" or products. Moreover, electrical energy metering in complex manufacturing facilities is necessary to provide industrial enterprises higher levels of quantification and visibility in their energy consumption. Both voltage and current need to be measured at either low or high sampling rates, in order to calculate power consumption and to produce more complex power quality statistics such as sags, peaks, and harmonics (O'Driscoll & O'Donnell, 2013). On the basis of the measured power, empirical energy models can be built for estimating the energy consumption related to the production.

There has been a compelling need to adopt data management systems in industrial operational processes and product-development principles in order to enhance IoT applications, while the development of big data is already lagging behind in integration with cloud computing. It has been widely recognized that these two technologies are interdependent and should be jointly developed: meaning that the widespread deployment of IoT drives the high growth of data (Min et al. 2014). Big data management is a complex process, particularly when abundant data originating from heterogeneous sources are to be used for business intelligence and decision-making (Baker, 2014). Furthermore, big data management has become a key to the success of many enterprises, science, industries, engineering fields and government ventures (Chaudhuri et al., 2011). The main objective is to enhance data quality and accessibility for decision-making and improve productivity. Therefore, "big data" could be defined as a fast-growing amount of data from various sources that increasingly poses a challenge to industrial organizations and also presents them with a complex range of valuable-use, storage and analysis issues.

Traditional data storage and processing are typically fed with relatively clean datasets generated by limited sources; hence, the results tend to be accurate. However, the recent introduction of Industry 4.0 paradigm leads to the collection of massive, heterogeneous and frequently generated data (Bevilacqua et al., 2017). This has revealed a serious management problem not only due to the growth in the volumes of datasets but also to their complexity and volatility that makes processing and analysis very hard to achieve. These aspects are very important when a company aims at analyzing the energy characteristics of its production system. In this case, data provided by different sources and geographically dispersed repositories must be taken into consideration. Important information must be collected from administrative office (i.e. electricity accounts), production sites (i.e. workstations energy consumption,

marking data of production progress), from suppliers (i.e. material delivery date), from production planning department (i.e. master production schedule) and from technical department (i.e. products codes, bill of materials, working times and manufacturing cycle).

These characteristics bring several problems to develop data analytics architecture. In this paper, we propose a data analytics framework for IoT, in order to integrate the data collected from different sources and to improve energy-aware decision-making.

The remainder of this paper is organized as follows. Section 2 presents a literature review regarding energy consumption models developed for production systems, with a special focus on IoT, Big Data Management and energy management in the industrial sector in Section 2.1. Section 3 introduces the research approach and the case study. Results of the study are shown in section 4 while discussion and conclusions are reported in Section 5.

2. Energy consumption models for production systems: Literature review

The development of energy models at the level of unit process is a research topic quite analyzed in literature.

Some authors focused on machining process level while other works were carried out on the machine-tool level. Regarding papers on machining process level, Srinivasan and Sheng (1999) developed an approach for macro and microplanning of feature-based machining. Microplanning looked at selecting process parameters, tooling, and cutting fluid based on process energy use, waste streams, process quality, and machining time. Draganescu et al. (2003) used experimental data and Response Surface Methodology (RSM) to establish a statistic model of machine-process efficiency and specific energy consumption in machining. Sarwar et al. (2009) chose Specific Cutting Energy (SCE) as the evaluation parameter of measuring the efficiency of the metal cutting process, and the variation of SCE regarding different workpiece materials provided valuable information for bandsaw manufacturers and end users to estimate machinability characteristics for selected workpieces.

Regarding works on machine-tool level, we can highlight Diaz et al. (2011) work. They carried out a characterization on the energy consumption of milling machine tools during their use stage. They studied the effect of workpiece material on power demand. Dahmus and Gutowski (2004) developed an experimental research on machine tool energy consumption and categorized the total energy of the system in three main activities, namely "Constant start-up operations", "Run-time operations" and "Material removal operations". Herrmann et al. (2010) addressed the energetic consumption of the machine tools and extended the perspective by considering the ecological aspects beyond the economic input and output flows.

Assessment methods of energy consumption are another aspect quite analyzed in literature. Some authors highlighted the importance of carrying out a real-time monitoring system while other authors proposed theoretical models for analyzing energy consumptions.

For instance, in Abele et al. (2012), work power measurements of a single machine tool is described by several functional modules that further consist of various

components. Within their Hardware-in-the-Loop-Simulation (HiLSimulation), a physical machine controller is connected to the simulation model so that the programmable logic control (PLC) or numerical control (NC) signals, which contain power-on states, axis speeds, machine tool movement path, process operations, etc., are coupled with the functional modules and components to enable continuous energy simulation of a machine tool. In addition to estimating the machine energy requirement within the work of Abele et al. described above, Eberspacher et al. (2014) further developed the HiL-Simulation model for real-time monitoring of the energy demand of a machine and its functional modules in production environments. Dietmair and Verl (2009) introduced a generic method to model the energy consumption behavior of machines based on a statistical discrete event formulation. The parameter information required to characterize the discrete events can be obtained with a small number of simple measurements or with a degree of uncertainty from the machine and component documentation. Vijayaraghavan and Dornfeld (2010) pointed out that, in order to decrease energy consumption, energy data has to be placed in the context of the manufacturing activity. They developed a monitoring system, in which MTConnect_standard, as an XML-based standard, for data exchange is selected for data collection from manufacturing equipment.

The automated monitoring system can help attach contextual processing-related information to the raw data. Therefore, it is very important for reducing energy consumption in order to develop a real-time energy efficiency monitoring system of machine tools. On the other hand, several authors highlighted that generally manufacturing machines and equipment are not metered permanently (Müller & Loffler, 2009; Garetti and Taisch (2012). Lack of sub-metering is highlighted as the main barrier to improving energy efficiency in non-energy-intensive manufacturing by Rohdin and Thollander (2006), as well for energy intensive industry by Trianni et al. (2013).

2.1. IoT and data analtics for energy management in production systems

Recently, some authors tried to integrate IoT and Big Data Management concepts in order to manage energy consumption data and improve energy-aware decision-making. Tao et al. (2014) developed a new method for Energy-saving and emission-reduction based on Internet of Things (IoT) and bill of material (BOM). Shrouf et al. (2014) proposed an approach for energy management in smart factories based on the IoT paradigm. They developed a guideline and highlighted expected benefits of this approach.

Always Shrouf and Miragliotta (2015) contributed to the understanding of energy-efficient production management practices that are enhanced and enabled by the Internet of Things technology. Moreover, they presented a framework to support the integration of gathered energy data into a company's information technology tools and platforms.

Regarding Data Management in literature data analytics methods have been used by various researchers in energy system applications. With the emergence of the data mining approach for predictive modeling, different types of models can be built on a

unified platform: to implement various modeling techniques, assess the performance of different models and select the most appropriate model for future prediction. Tso and Yau (2007) have used regression analysis, decision tree and neural networks models for the prediction of electricity energy consumption. Model selection is based on the square root of the average squared error. Figueiredo et al. (2005) presented an electricity consumer characterization framework based on a knowledge discovery in databases procedure, supported by data mining techniques. Lu et al. (2013) presented a framework for predicting the electricity price, the price spike, the level of spike and the associated forecast confidence level. The proposed model is based on a mining database including market clearing price, trading hour, electricity demand, electricity supply and reserve.

Other efforts (Seem, 2007; Berglund et al., 2011) first extract the features from daily energy consumption then use statistical methods to identify abnormally high or low energy use. However, these methods relied on the assumption that the data is sampled from a particular distribution, which may not hold true.

3. Energy management framework

Literature in the energy management research field focused attention on developing methods for reducing energy consumption and improving energy-aware decision-making. No many works developed integration methods of complex data sets from multiple information sources such as energy system, production system and enterprise information systems. These data sets must be integrated with data streams collected from wired and wireless sensors and meters in order to perform N-dimensional analysis of energy performance data and to support the decision-making process of the end users. Moreover, modern energy management systems incorporate data archival but energy managers need assistance in extracting useful information from a large volume of data compiled. In this context, this paper addresses this deficit by introducing a new method for designing a Data Warehouse that can be useful to a production system that aims at collecting big data. In our case study, data comes from many sources affected by veracity problems and are provided with a different velocity. The energy management framework proposed in this work is based on a data warehouse to store, integrate and analyse the complex data sets in order to support multi-dimensional analysis of energy performance data.

An approach is developed to integrate IoT-based energy management and company's information technology tools and platforms. This approach has been used in the case study presented in the next section but it is a general framework that can be used in every manufacturing company. The framework (Fig. 1) is a multi-layer model that includes three layers: 1) data collection layer, 2) data management layer and 3) data analytics layer. According to Haller et al. (2008), Internet of Things model has generally been recognized as three layers. The bottom level is used to perceive sensory data; the second layer is the network layer for data transmission; the top is the application layer. In this perspective, Data Collection layer implements the first two layers of the Haller architecture, while Data Management and Data Analytics layers belong to the Haller's application layer.

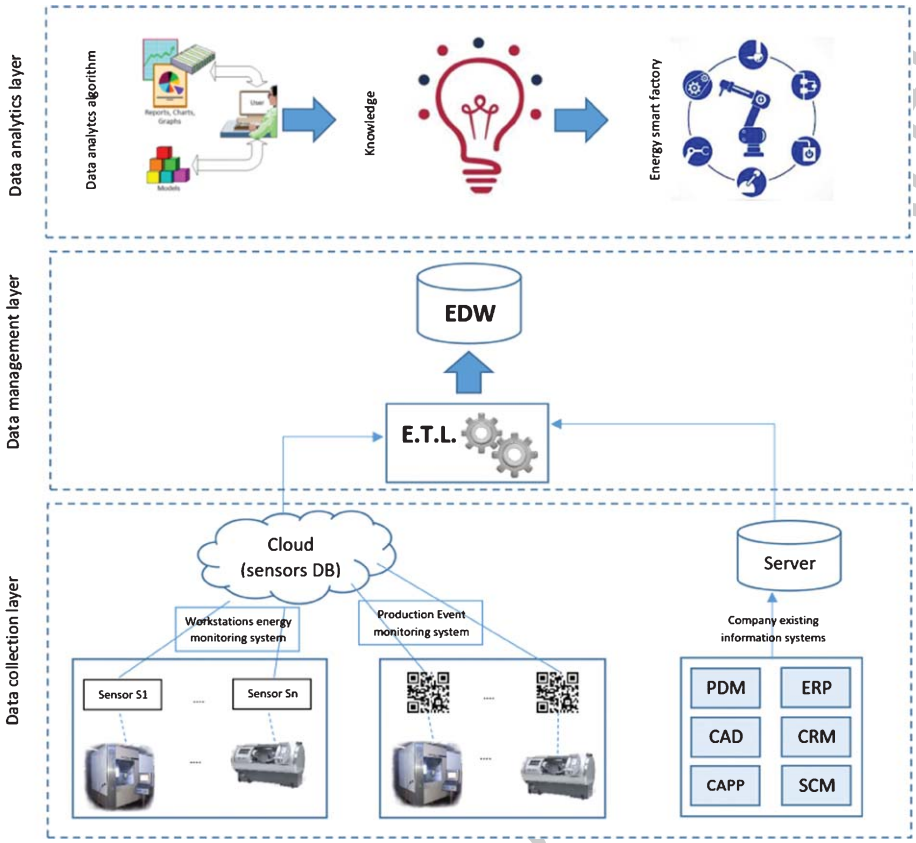


Fig. 1. Muti-layer model.

3.1. Data collection layer

Data collection layer adopts company existing information systems and devices, e.g. RFID Reader (Manufacturing Execution System) and sensors (Workstations), to collect various smart object's data. With the fast development and pervasive application of information technology in manufacturing, enterprise information systems such as product data management (PDM), enterprise resource planning (ERP), computer-aided design (CAD), computer aided process planning (CAPP), customer relationship management (CRM), and supply chain management (SCM) have been widely accepted and applied by manufacturing enterprises. Different types of data require different data collection strategies. Moreover, designing the layer require to define: (1) the machines that will be monitored; (2) a list of required measures (active power, reactive power, etc.); (3) the monitoring devices for each machine and its specifications; (4) the communication system; (5) where and how the data will be stored and analyzed. Furthermore, the production processes have to be identified (e.g. production sequence, the processing time for each product under different machine configuration), so as to link and

understand the energy consumption behavior and make the efficient decision. In the case study proposed in next section, it is assumed that each individual sensor will be assigned its own unique IP address and that the data will be transported through the Internet infrastructure to cloud-based applications. One of the pillars of IoT is cloud computing. According to Cook and Das (2004), cloud computing platforms provide two types of benefits: managed infrastructure services and a software framework that simplifies the development of large-scale applications. Cloud computing builds on the virtualization capabilities of modern computer systems to provide organizations with on-demand computing and storage capabilities. Through the use of fault tolerant systems and geographically distributed data centers, it can ensure high availability. Most importantly, it ensures that updates and patches can be applied in a timely manner.

Companies may benefit from cloud-based solutions without compromising security using a multi-layered approach and private clouds. They may choose to manage their own internally hosted data acquisition applications to collect data from their own private network of devices and use some type of gateway application to push the data to a cloud-based application for processing.

3.2. Data management layer

In the process of data collection, a series of problems, e.g., energy-efficiency, misreading, repeated reading, fault tolerance, data filtering and communications etc., must be solved. Data management layer applies an ETL (Extraction, Transformation and Loading) process for pulling data out of the source systems and placing it into a Data Warehouse (DW). After the extraction of interesting data from sources, they are cleaned and transformed for reconciling possible semantic heterogeneities (e.g., synonyms and homonyms, different representation of semantically equivalent concepts). Finally, data are saved in the DW. For instance, RFID data are collected by sensors as a stream of EPC format, which is the universal identifier for a physical object (Bevilacqua et al., 2013). After transformation, data are structured in a table where each record contains EPC, location, time_in and time_out. A DW is a system used in the enterprise to support decision at strategic level, hence it provides a unique view of the entire organization, including also external data. Furthermore, DW provides the access to current and historical data, which are typically stored in aggregated form. Data stored in a DW are used for creating analytical reports for knowledge workers throughout the enterprise. Data in a DW are organized on the basis of the multidimensional model (Golfarelli et al., 1998).

3.3. Data analytics layer

Data analytics layer is built based on data management and event processing. Various object-based or event-based data mining services, such as classification, forecasting, clustering, outlier detection, association analysis or pattern mining, are provided for applications, e.g., supply chain management, inventory management and optimization etc. The architecture of this layer is service-oriented. Smart meters and sensors enable

remote monitoring of energy consumption data across the factory. The data can then be stored and analyzed. The results and warning messages can be delivered through mobile applications to shop floor supervisor. Also, energy management experts can make real-time assessment by having a clear picture of energy consumption in real time.

After collecting and analyzing data, in this phase data are exploited into energy management tools (e.g. energy decision support system, simulation tools) to enable the decision makers to enlighten possible waste of energy, where improvement can be achieved, or select the most sustainable configuration mode of machines by considering the production planning in order to improve energy efficiency. In this phase, decision makers can also define strategies and practices to improve the energy efficiency of the smart factory "by design", for example by integrating energy data in production management practices.

4. Case study

A case study was conducted to show the proposed method. The company analyzed is an Italian medium enterprise specialized in the production of turned metal parts and precision mechanical components.

The production system of the company is strongly oriented to high technology products thanks to the presence of numerical control single and multiple spindle machines. The company performs automatic CNC turning operations on different types of metals: Aluminum, Copper, Brass, steel ETG 100, quenched and tempered steel, Stainless Steel, Alloy Steel. The company buildings cover an area of 5548 m^2 and have an electrical total annual consumption equal to 2,094,936 Mwh. Figure 2 shows the layout of the production site. The company is made up of four operating areas: raw material warehouse, processing departments, finishing and washing department, finished products warehouse; represented by spots 2, 3, 4 and 5 in Fig. 2. Company departments work 230 days per year in three shifts of eight hours/day. Machines set-up is carried out exclusively during the morning shift.

Various sensors have been installed on production machines since 2015 to acquire electrical measurements through single-phase and three-phase multimeters. The position of the sensors is shown by the spots in Fig. 2. Low-cost sensors have been used in order reduce the payback period of the investment. The meters are connected to a gateway that links the Modbus network with Ethernet. A web platform has been designed for acquisition, processing and presentation of energy and production data according to Big Data paradigms. Indeed, the case study is a typical Vs-problem. The velocity and volume is due to the number of acquisitions; in addition to the 10 meters shown in the Figure, we have one sensor for each active machine. Each sensor collects three measures: active power, reactive power and energy consumptions. Although in the case study the sensors sampling rate is low, we like to note that, the proposed framework is independent on the velocity of acquisition. Furthermore, as shown in next subsection, we take into account different forms of data (variety): energy sensors' measures, production data from the ERP and manually inserted production phases.

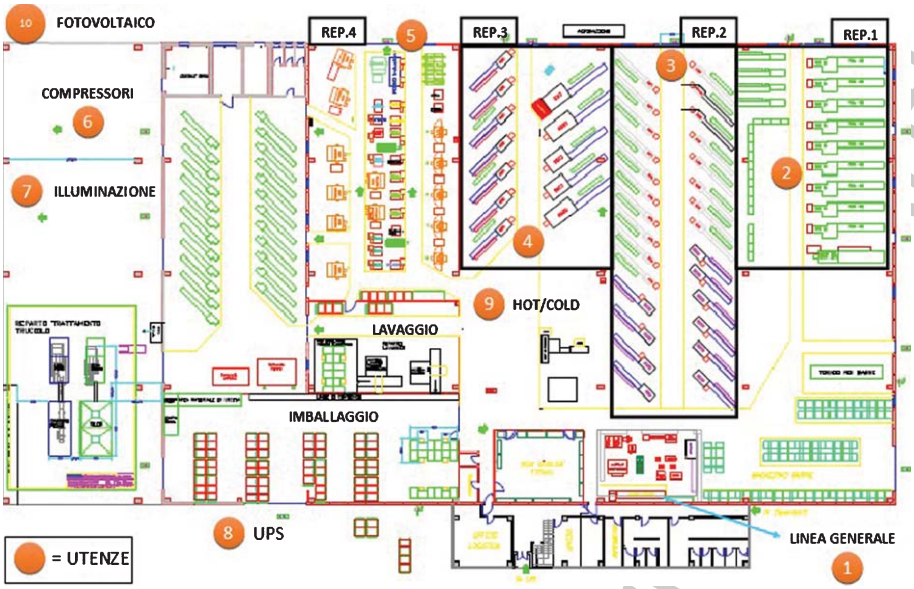


Fig. 2. Company layout.

4.1. Data sources

In the case study, both information from the ERP and data from sensors has been considered. From the former, the production plan has been extracted in order to be aware of what has been produced and characteristics of the product, like name, description and raw material. Sensors have been used to collect actual data: production events and energy consumption.

A production event is characterized by an operator, which manages a specific production phase, in a given time interval. Data about events has been collected using several QR codes placed next to each machine; a QR code for both the beginning and the end of each specific phase. In this case study, four phases have been identified: set-up, warming-up, production and technical stoppage. Furthermore, stoppages are further classified in sub-phases, namely machine tuning, changing tool, production completed, separator replacement, plank manual replacement, and loading jam. The operator, by scanning the QR code through a common reader, easily identifies the phase or sub-phase and the system assigns the actual timestamp, so all information about the event are collected.

Data about energy consumptions are collected by means of sensors which are able to measure active power, reactive power and energy consumptions. Nine machines have been coupled with an energy sensor. In order to ensure the economic sustainability of the experiment, while ensuring the quality, the company has chosen to adopt sensors that return accurate measurements but with low frequencies; the sampling time is 15 minutes. Chosen sensors return the cumulative values in the sampling interval of active power, reactive power and energy consumption.

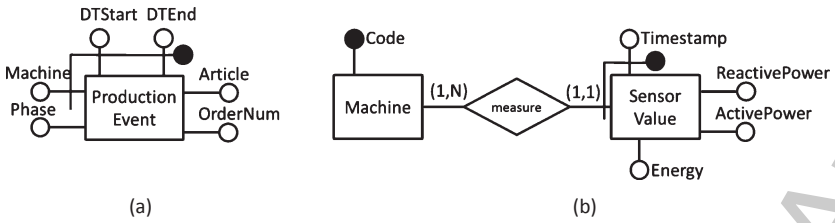


Fig. 3. ER representation of sensor data schema: (a) production event, (b) energy consumption.

All data from sensors are sent to and stored in an enterprise data cloud. Figure 3 shows the conceptualization made for data in the cloud using the Entity-Relationship (ER) model, a reference model for conceptual design (Chen, 1976). Similarly, Fig. 4 shows the portion of the ERP schema representing needed information about the production plan. From Fig. 3(a), a production event is identified by the triple formed by code of the machine, start and end date/time, and is characterized by the code of the article that is being produced, the related order and the phase to which the event refers. As regards energy consumption in Fig. 3(b), each sensor value is identified by the related machine and the instant the measurement refers to; each measurement is formed by three values: energy, active power and reactive power.

In Fig. 4, a production order describes the number of products ("quantity") of a given article that will be produced on a specific machine on the scheduled date ("SchedDate"). A production order is identified by its number ("OrdNum"). Each article is characterized by a textual description, the code, and the raw material of which is made the cylinder used in the lathe and its diameter. Finally, a production order is assigned to a given machine, which is located in just one department.

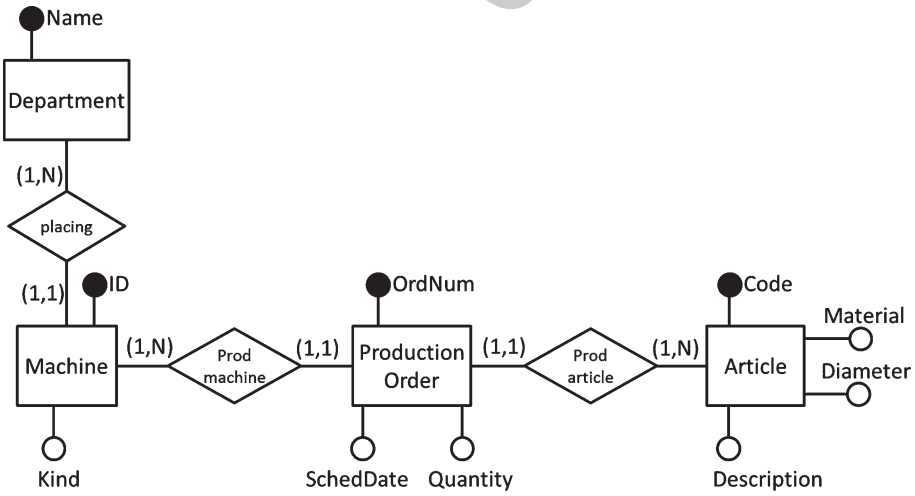


Fig. 4. ER representation of the portion of the ERP schema representing the production plan.

4.2. Data Warehouse design

The first step of Data Warehouse design (Golfarelli et al., 1998; Kimball & Ross, 2002) is the integration of relevant portions of different data schemas. Conceptualization in the form of ER schemas greatly simplify this step, allowing to easily identify overlapping elements, conflicts in the representation of the same element, and/or missed relationships between different schema fragments. For instance, it can be seen that a machine is represented as a separate entity with its own attributes in ERP schema (Fig. 4) while it is a simple attribute describing the name of the machine where a Production Event takes place (Fig. 3(a)). The general strategy is to choose, among the different conflicting representations, the most general one, that is the one that allows accommodating the others as special cases (in the example, the machine as an entity). Similar reasoning applies to other conflicting representation. A more critical issue is the integration of Production Events (Fig. 3(a)) with their energy-related measures (Sensor Value in Fig. 3(b)). Indeed, events refers to highly-variable (manually defined) time intervals, while measurements are acquired every 15 minutes from 00 : 00 of each day. Figure 5 shows an example of production event and measurement interval (i.e., the time between two subsequent measurement instants). In the Figure, the start and end time of a production event (i.e., a production interval) is represented using subscript *c*, while the subscript p is used for measurements. In order to integrate data, we have intersected intervals hence defining new events which inherit the production phase from the corresponding production event and consumption values from measurement intervals. In particular, energy (active power, reactive power) consumption for each interval is computed as a fraction of the measured energy (active power, reactive power) that is proportional to the length of the interval itself. The following formula is used to estimate consumption values of the interval $I_i = [t_i; t_{i+1}]$:

$$\begin{pmatrix} \hat{E}(t_i) \\ \widehat{AP}(t_i) \\ \widehat{RP}(t_i) \end{pmatrix} = \begin{pmatrix} E(t_{c(x+1)}) - E(t_{cx}) \\ AP(t_{c(x+1)}) - AP(t_{cx}) \\ RP(t_{c(x+1)}) - RP(t_{cx}) \end{pmatrix} . \frac{(t_{i+1} - t_i)}{15}$$

where $\hat{E}(t_i)$, $\widehat{AP}(t_i)$ and $\widehat{RP}(t_i)$ are estimated energy, active power and reactive power in the interval I_i respectively, $E(t_{cx})$, $AP(t_{cx})$ and $RP(t_{cx})$ are energy, active power and reactive power measured by a sensor at the x-th instant, and $t_{c(x+1)} =$

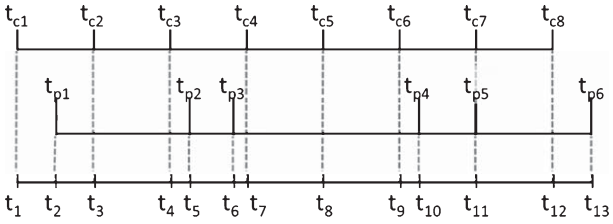


Fig. 5. An example of integration of consumption (t_{ci}) and production event (t_{pj}) timelines.

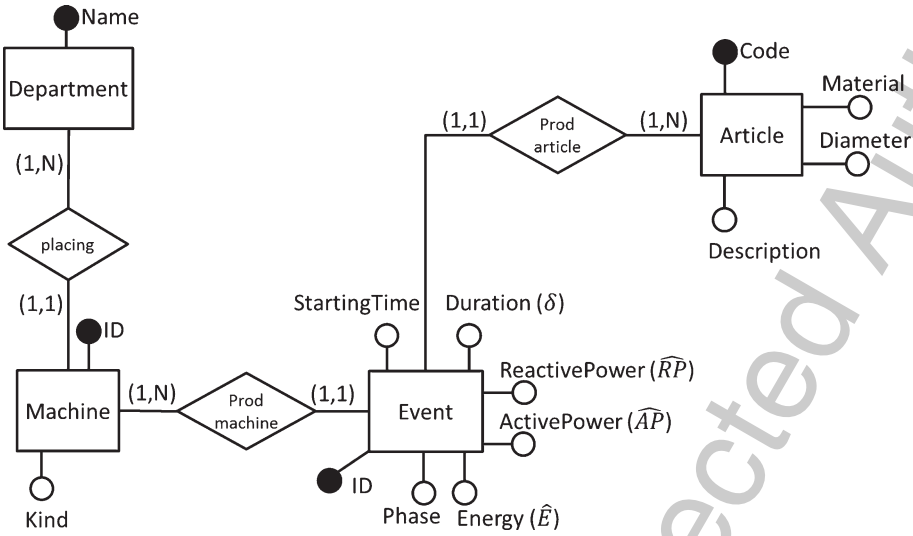


Fig. 6. The integrated schema.

$\left\{k \cdot 15,\ k \in \mathbb{N} | k \geq \frac{t_{i+1}}{15}\right\}$. All time values are in minutes. For instance, in Fig. 5, the interval $[t_9,\ t_{10}]$ inherits the phase of the production event in $[t_{p3};\ t_{p4}]$ and energy estimate $\hat{E}(t_9) = (E(t_{c7}) - E(t_{c6})) \cdot \frac{(t_{10}-t_9)}{15}$. Noteworthy that a simpler integration strategy, such as assigning to a production event the average of all measurements gathered during its interval (e.g., in $[t_{p1};\ t_{p2}]$ there are two measurements in t_{c2} and t_{c3}), would lead to a loss of information; indeed, during a fast production event (, like $[t_{p2};\ t_{p3}]$), no consumption values could be collected, hence we would not able to assign estimated energy, active power and reactive power to the event. In these cases, we would not be able to correctly perform all analysis (e.g. to evaluate energy consumption with respect to production phases), hence resulting in low-quality DW.

The resulting integrated schema is shown in Fig. 6, where "Article" and "Machine" attributes, of "Production Event" and "Sensor Value" respectively, have been reificated by using the extended descriptions provided by the ERP schema.

Following the approach proposed by Golfarelli et al. (1998), next steps in Data Warehouse design are related to the definition of multidimensional model elements, namely the *fact* to be analysed, the analysis perspectives (or *dimension*s), the granularity *level*s at which data are shown, and the *measure*s by which the fact is evaluated. The pivotal concept in the integrated schema, which is chosen as fact, is the "Event". Note that, one might think to keep the "Machine" at the center, but since multiple events are related to the same machine, this design choice would lead to a DW where each machine will have several phases and just aggregated consumption values (e.g., average), and the analyze of consumption with respect to the phases will be prevented.

Four dimensions have been selected: the machine where the production occurred, the time when the event begins, the product and the production phase. Each dimension

is structured in a hierarchy of levels, where a part-of relation exists between members of a level and members of a higher level (e.g., a machine is part of a department). In this way, moving from a level to the higher one, the fact represents a wider portion of data and corresponding measures' values are aggregation of values returned at lower level. The following hierarchies have been selected:

- Machine: *department*→ *machine*;
- Time: *year*→ *month*→ *day*→ *time slot*→ *starting time*;
- Product: *article*→ *raw material*;
- Phase: *phase*→ *sub-phase*;

In order to allow analysis at different granularity levels, the "Phase" and "Starting-Time" attributes have been suitably expanded. Measures have been selected among quantitative attributes of the fact. In particular:

- Total Energy: = sum($\hat{E}$)
- Total ActivePower: = sum($\widehat{AP}$)
- Total ReactivePower: = sum($\widehat{RP}$)
- Time interval: = sum(δ)
- Average Power: = $\frac{\text{Total Energy}}{\text{Duration}}$

The final star schema of the Data Warehouse is depicted in Fig. 7.

In order to populate the DW, an ETL process has been defined, which is able (a) to extract useful data from sources (Figs. 3 and 4), (b) to transform them removing errors, defining new time intervals and estimating new energy consumption values (as defined above), (c) to load data into the DW (Fig. 7).

In particular, some errors could occur in source data due to failures of energy consumption sensors and to the manual gathering of data about production events. In the first case, measures detected by the sensor could be missing or out-of-the-range. In the ETL process, they are replaced with values obtained by the same sensor in the previous sampling interval. As regards production data, there are two main issues: 1) the operator wrongly scans the begin of a phase before the end of the previous one; 2) the same system is used to record the end of a work shift, in this case we have a production event which begins and ends at the same time. In the first case, usually the operator quickly recognizes the error and scans the right code; hence, these kind of issues are handled by reversing overlapping timestamps. Issues of the second kind are simply removed, because they do not represent production events.

4.3. Analysis

OLAP analysis (Fig. 8) allows the company to divide the total machine tool power in four power levels:

- Fixed power: power demand of all activated machine components ensuring the operational readiness of the machine.
- Operational power: power demand to operate components.
- Tool tip power: power demand at tool tip to remove the workpiece material.

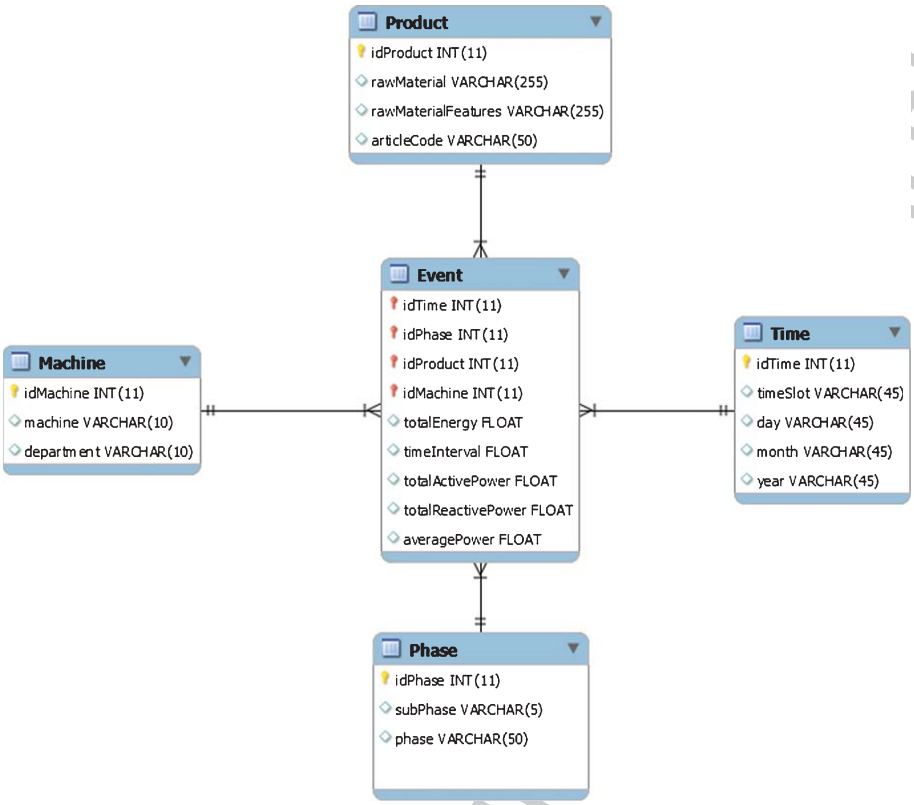


Fig. 7. The Data Warehouse star schema.

- Unproductive power: power converted to heat mainly due to friction during the material removal.

During the implementation of the proposed approach, several operations management practices have been enhanced by integrating energy data. For example, energy consumption data has been collected from the machines under different configurations (e.g. machine speed). Then, based on the flexibility of production schedule, this data enabled the production manager to select the most efficient configuration of the machines. Moreover, the analysis of energy consumption data provided the decision makers with a clear picture on the energy waste at production level (e.g. idle time), and it provided precise information about the energy needed to produce one piece. Figure 9 illustrates a typical energy consumption profile of a turning machine.

The load profiles of single machines add up to a cumulative load profile for the process chain and determine the embodied energy of a product. The specific energy and resource consumption behavior of a process chain is significantly influenced by its specific technical configuration (design) and control. This includes the individual selection/combination of processes/machines and their inter linkage (e.g. process chain

				Measures		
Product	Time	Machine	Phase	Time Interval (m)	Total Energy	Average Power (kW)
-All Products	+All Dates	+All Machines	-All Events	10.877,6666	1.011,7419	5,5807
			+Production	10.105,5832	954,2617	5,6657
			+Set-up	452,6667	28,8159	3,8195
			+Stoppage	269,2834	24,0406	5,3566
			+Warminig-up	50,1334	4,6237	5,5337
-BARRA TRAF. OTTONE CW614N D.23.00h11	+All Dates	-All Machines	-All Events	6.659,9999	498,7990	4,4937
			+Production	6.575,1499	493,3865	4,5023
			+Stoppage	62,2333	4,5144	4,3524
			+Warminig-up	22,6167	,8981	2,3827
		+REP-3	-All Events	6.659,9999	498,7990	4,4937
			+Production	6.575,1499	493,3865	4,5023
			+Stoppage	62,2333	4,5144	4,3524
			+Warminig-up	22,6167	,8981	2,3827
PF3673-J	+All Dates	-All Machines	+All Events	6.659,9999	498,7990	4,4937
		+REP-3	+All Events	6.659,9999	498,7990	4,4937
-BARRA TRAF. OTTONE CW614N D.31.00h11	+All Dates	+All Machines	+All Events	4.217,6667	512,9429	7,2971
PF3686-E	+All Dates	+All Machines	+All Events	4.217,6667	512,9429	7,2971

Fig. 8. A screenshot of the OLAP analysis implemented using Mondrian.

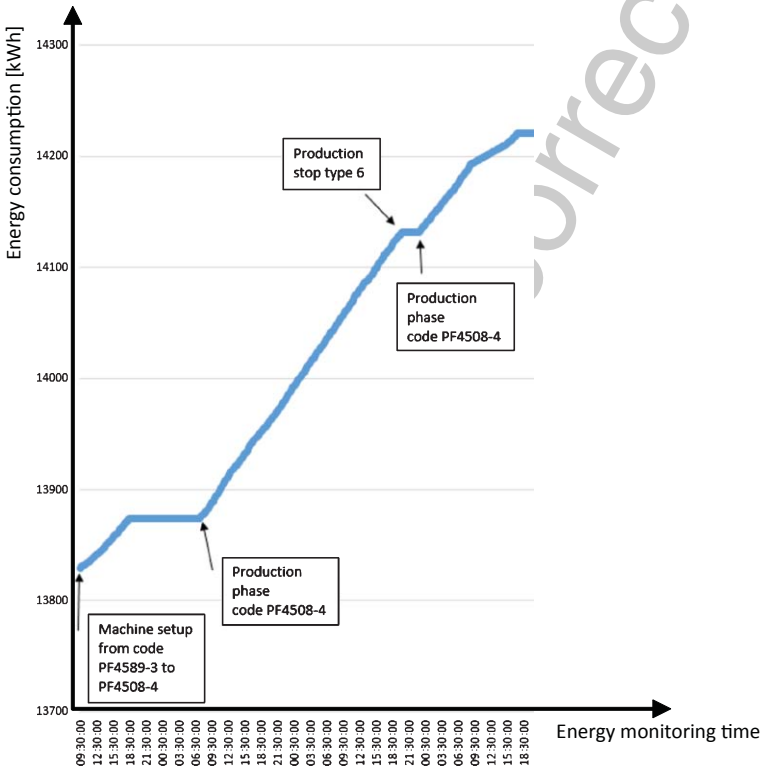


Fig. 9. Energy consumption profile of a turning machine.

structure, buffers) as well as aspects like batch sizes, scheduling of orders (e.g. start time, capacity allocation) or speed of production.

In this perspective, the electrical work can be reduced, for example by optimal utilization of equipment and avoiding energy waste in idling machines or the selection of

appropriate machines for the specific manufacturing task. The avoidance of consumption peaks is another important issue. From an economic perspective, peaks should be avoided since they may cause cost surcharges in an electricity bill and production interruptions. Other important aspects concern the possibility to shift the energy consumption from day to night, because of less expensive energy price rates (e.g. base time at night).

5. Discussion and conclusion

The paper presented a general methodology and practical issues involved in the development of a data analytics framework for IoT in an energy management setting.

The results of this study can facilitate various applications for supporting an energy aware design and control of process chains. Energy efficiency improvement requires awareness of energy consumption behavior at production line and machine level. In this context, smart meters provide real time data, and take decisions in collaboration with company information systems. In the case study, combined with other information systems, Internet of things played the role of data aggregation, improving management capacity and efficiency. The core technology is the energy sensor network and computer information processing, for building an advanced, powerful information acquisition and processing platform. New data analytic model has been developed to integrate the data collected from different sources: workstations energy consumption and marking data of production progress from production site, electricity accounts from administrative office, material delivery date from suppliers, master production schedule from production planning department and products codes, bill of materials, working times and manufacturing cycle from technical department. The process commences by transforming big data from original format to computer formats. It progresses with applying big data operations towards achieving decision-making. We propose a big data management process flow as a layered component diagram that shows all steps big data must undergo in order to accomplish the management process. The journey begins with big data being transmitted from different sources to storage devices and continues with the implementation of pre-processing, integration and analysis, amounting to the decision-making endpoint.

There are some advantages of the proposed data analytics based method. First, the machine energy patterns and characteristics can be discovered by this method, while traditional methods can only show the statistical characteristics of the entire data set. Second, different analysis perspectives can be adopted, focusing on products or product types, or the whole production process, besides machines. Finally the approach is flexible enough to allow the introduction of new sensors and/or measures, or new kinds of analyses. We plan to exploit the system deployed to obtain further insights about energy consumption profiles and patterns, also by applying different data analysis techniques like clustering and predictive analytics techniques.

References

Abele, E., Eisele, C., & Schrems, S. (2012). Simulation of the energy consumption of machine tools for a specific production task. In: Dornfeld, D.A., Linke, B.S. (Eds.), Leveraging Technology for a Sustainable World. Springer, Berlin Heidelberg, pp. 233-237.

Baker, T. (2014). Designing and managing Big Data – How are you researching your outcomes. *In SimTecT*.

Berglund, J., Michaloski, J., Leong, S., Shao, G., Riddick, F., Arinez, J., & Biller, S. (2011), "Energy Efficiency Analysis for A Casting Production System," in *Winter Simulation Conference, IEEE*, 2011, pp. 1060-1071. Proceedings of the 2014 IEEE IEEM 701.

Bevilacqua, M., Ciarapica, F.E., Mazzuto, G., & Paciarotti, C. (2013). The impact of RFID technology in hospital drug management: An economic and qualitative assessment. *International Journal of RF Technologies*, *4*(3-4), 181-208.

Bevilacqua, M., Ciarapica, F.E., & De Sanctis, I. (2017). Lean practices implementation and their relationships with operational responsiveness and company performance: An Italian study. *International Journal of Production Research*, *55*(3), 769-794.

Candela, L., Castelli, D., & Pagano, P. (2012). Managing big data through hybrid data infrastructures. *ERCIM News*, *89*, 37-38.

Chaudhuri, S. (2012). What next?: A half dozen data management research goals for big data and the cloud. In: *Proceedings of the 31st Symposium on Principles of Database Systems*. ACM.

Chaudhuri, S., Dayal, U., & Narasayya, V. (2011). An overview of business intelligence technology. *Commun ACM*, *54*(8), 88-98.

Chen, P. (1976). The entity-relationship model - toward a unified view of data. *ACM Transactions on Database Systems*, *1*(1), 9-36.

Cook, D.J., & Das, S.K. (2004). Smart Environments: Technologies, Protocols, and Applications, in: Wiley Series on Parallel and Distributed Computing, Wiley-Interscience, 2004.

Dahmus, J., & Gutowski, T. (2004). An Environmental Analysis of Machining. *Proceedings of ASME International Mechanical Engineering Congress and R&D Exposition*, 13-19.

Diaz, N., Redelsheimer, E., & Dornfeld, D. (2011). Energy Consumption Charaterization and Reduction Strategies for Milling Machine Tool Use. *18th CIRP LCE Conference*, Braunschweig, 263-267.

Dietmair, A., & Verl, A. (2009). A generic energy consumption model for decision making and energy efficiency optimisation in manufacturing. *Int J Sustain Eng*, *2*, 123-133.

Draganescu, F., Gheorghe, M., & Doicin, C.V. (2003). Models of machine tool efficiency and specific consumed energy. *Journal of Materials Processing Technology*, *141*(1), 9-15.

Eberspacher, P., Schraml, P., Schlechtendahl, J., Verl, A., & Abele, E. (2014). A model- and signal-based power consumption monitoring concept for energetic optimization of machine tools. *Procedia CIRP*, *15*, 44-49.

Feller, E., Ramakrishnan, L., & Morin, C. (2015). Performance and energy efficiency of big data applications in cloud environments: A hadoop case study. *Journal of Parallel and Distributed Computing*, *79-80*, 80-89.

Figueiredo, V., Rodrigues, F., & Gouveia, J.G. (2005). An Electric Energy Consumer Characterization Framework Based on Data Mining Techniques. *IEEE Transactions on Power Systems*, *20*(2005), 596-602.

Garetti, M., & Taisch, M., (2012). Sustainable manufacturing: Trends and research challenges. *Prod Plan Control Manag Oper*, 83-104.

Golfarelli, M., Maio, D., & Rizzi, S. (1998). The dimensional fact model: A conceptual model for data warehouses. *International Journal of Cooperative Information Systems*, *7*(2-3), 215-247.

Hadera, H., Harjunkoski, I., Sand, G., Grossmann, I.E., & Engell, S. (2015). Optimization of steel production scheduling with complex time-sensitive electricity cost. *Comput Chem Eng*, *76*, 117-136.

Haller, S., Karnouskos, S., & Schroth, C. (2008). "The Internet of Things in an Enterprise Context," in First Future Internet Symposium, FIS 2008 Vienna, Austria, September 29-30, 2008 Revised Selected Papers, J. Domingue, D. Fensel, and P. Traverso, Eds. Springer-Verlag, Berlin, Heidelberg, 2009, 2009, pp. 14-28.

Herrman, C., Bergmann, L., Thiede, S., & Zein, A. (2007). Energy Labels for Production Machines – An Approach to Facilitate Energy Efficiency in Production Systems. *Proceedings of 40th CIRP International Seminar on Manufacturing Systems Location*, Liverpool, UK.

Herrmann, C., & Thiede, S. (2009). Process chain simulation to foster energy efficiency in manufacturing. *CIRP Journal of Manufacturing Science and Technology*, *1*(4), 221-229.

Herrmann, C., Kara, S., Thiede, S., & Luger, T. (2010). Energy Efficiency in Manufacturing Perspectives from Germany and Australia. *17th CIRP LCE Conference*, Hefei, 23-28.

Kees, A., Oberländer, A., Röglinger, M., & Rosemann, M. (2015). Understanding the Internet of Things: A Conceptualisation of Business-To-Thing (B2T) Interactions. *In the Proceedings of Twenty-Third European Conference on Information Systems (ECIS)*, pp. 1-16. Münster, Germany.

Kimball, R., & Ross, M. (2002). The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, 2nd ed., New York, NY, USA: John Wiley & Sons, Inc.

Lu, S.-M., Lu, C., Tseng, K.-T., Chen, F., & Chen, C.-L. (2013). Energy-saving potential of the industrial sector of Taiwan. *Renew Sustain Energy Rev*, *21*, 674-683.

Min, C., Shiwen, M., & Yunhao, L. (2014). Big data: A survey. *Mobile Network Applications*, *19*, 171-209.

Müller, E., & Loffler, T. (2009). Improving energy efficiency in manufacturing plants e case studies and guidelines. In: *16th CIRP International Conference on Life Cycle Engineering (LCE 2009).* Cairo, Egypt, pp. 465-471.

O'Driscoll, E., & O'Donnell, G.E. (2013). Industrial power and energy metering e a state of the art review. *J Clean Prod*, *41*, 53-64.

Philip Chen, C.L., & Zhang, C.-Y. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on big data. *Inf Sci*, *275*(0), 314-347.

Rohdin, P., & Thollander, P. (2006). Barriers to and driving forces for energy efficiency in the non-energy intensive manufacturing industry in Sweden. *Energy*, *31*, 1836-1844.

Trianni, A., Cagno, E., Thollander, P., & Backlund, S. (2013). Barriers to industrial energy efficiency in foundries: A European comparison. *J Clean Prod*, *40*, 161-176.

Tso, G.K.F., & Yau, K.K.W. (2007). Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural Networks, *Energy*, *32*, 1761-1768.

Sarwar, M., Persson, M., Hellbergh, H., & Haider, J. (2009). Measurement of specific cutting energy for evaluating the efficiency of bandsawing different workpiece materials. *International Journal of Machine Tools and Manufacture*, *49*(12-13), 958-965.

Seem, J.E. (2007). Using intelligent data analysis to detect abnormal energy consumption in buildings. *Energy and Buildings, 39*(2007), 52-58.

Shrouf F., Ordieres J., & Miragliotta G. (2014). Smart factories in industry 4.0: A review of the concept and of energy management approached in production based on the internet of things paradigm. *Proceedings of the 2014 IEEE IEEM*.

Shrouf F., & Miragliotta G. (2015). Energy management based on Internet of Things: Practices and framework for adoption in production management. *Journal of Cleaner Production*, *100*(2015), 235-246.

Srinivasan, M., & Sheng, P. (1999). Feature-based process planning for environmentally conscious machining – part 1: Microplanning. *Robotics and Computer-Integrated Manufacturing*, *15*(3), 257-270.

Tao, F., Zuo, Y., Xu, L.D., Lv, L., & Zhang L. (2014). Internet of things and BOM-based life cycle assessment of energy-saving and emission-reduction of products. *IEEE Transactions on Industrial Informatics*, *10*(2).

Vijayaraghavan, A., & Dornfeld, D. (2010) Automated energy monitoring of machine tools. *CIRP Annals – Manufacturing Technology*, *59*(1), 21-24.

Vikhorev, K., Greenough, R., & Brown, N. (2013). An advanced energy management framework to promote energy awareness. *J Clean Prod*, *43*, 103-112.

Waide P., & Brunner C.U. (2011). Energy-Efficiency Policy Opportunities for Electric Motor-Driven Systems. International Energy Agency, 2011.

Practical 8

Case Study on Metallurgy & Material Science

A SURVEY ON DATA MINING IN STEEL INDUSTRIES

S. Umeshini[1], C.PSumathi[2]

[1]ResearchScholar, Department of Computer Science, SDNB Vaishnav College for Women, Chennai, India
[2]Associate Professor and Head, Department of Computer science, SDNB Vaishnav College for Women, Chennai India

ABSTRACT

In Industrial environments, huge amount of data is being generated which in turn collected indatabase anddata warehouses from all involved areas such as planning, process design, materials, assembly, production, quality, process control, scheduling, fault detection,shutdown, customer relation management, and so on. Data Mining has become auseful tool for knowledge acquisition for industrial process of Iron and steel making. Due to the rapid growth in Data Mining, various industries started using data mining technology to search the hidden patterns, which might further be used to the system with the new knowledge which might design new models to enhance the production quality, productivity optimum cost and maintenance etc. The continuous improvement of all steel production process regarding the avoidance of quality deficiencies and the related improvement of production yield is an essential task of steel producer. Therefore, zero defect strategy is popular today and to maintain it several quality assurance techniques areused. The present report explains the methods of data mining and describes its application in the industrial environment and especially, in the steel industry.

KEYWORDS

Repository, Explanatory variables,Clusters, Dependent variables, Ensemble methods, Decision making, patterns.

1.INTRODUCTION

In most steel sectors, [1] manufacturing is extremely competitive and financial margins that differentiate between success and failure are very tight with established industries needed to compete, produce and sell at global level. To master trans-continental challenges, a company must achieve low cost production yet still maintain highly skilled, flexible and efficient workforces who could consistently design and produce high quality, low cost products. This can be achieved by using data mining techniques to improve decision making.

Data Mining can be generally said as a technique to find patterns (extraction) or interesting information in large amount of data[2]. This technique has been widely used in research areas like engineering, marketing, business, education and now especially in industries like Iron and Steel, Rubber etc. However, knowledge can take many forms and it is necessary to identify the kind of knowledge to be mined when testing the huge amount of data generated during manufacturing.
Data Mining is the process of interesting patterns and knowledge from large amount of data. The data source can include databases, data warehouses, the web, other information repositories, data that streamed into the system automatically [3].

DOI:10.5121/ijcses.2017.8201

2.NINE LAWS OF DATA MINING [4]:

First Law: "Business goals Laws" -Business objectives are the origin of every data mining solution.

Second law: "Business knowledge Law" - Business knowledge is control to every step of the data mining solution.

Third law: "Data preparation Law" - Data preparation is more than every data mining process.

Fourth law: "NFL – DM" - The right model for a given application can only be discovered by experiment.

Fifth law: "Watkins Law" - There are always patterns

Sixth law: "Insight Law" - Data mining amplifies perception in the business domain.

Seventh law:"prediction Law" - Prediction increases information locally by generalization.

Eighth Law:"Value Law" - The value of data mining results is not determined by the accuracy (or) stability of predictive models.

Ninth Law:"Law of Change" - All patterns are subject to change.

3.DATA MINING STRUCTURE

The architecture of a typical data mining system [3] has the following major components (Han &Kamber)

- Database, Data warehouse (or) other information repository.
- Database (or) Data warehouse server
- Knowledge base
- Data mining engine
- Pattern Evolution module
- Graphics user interface

Fayyad (1996) stated that data mining algorithms consist of some specific mix of these components.

3.1. Sources of data:

We get lot of data through internet and other sources. The data available over internet may be in different types. When we have huge amount of data, then it is big data, which creates another challenge in every aspect. The sources of data may be internal or external. The data may be structured, unstructured and semi-structured(Figure1). Such data are analyzed by predefined tools and technique referred to as data mining. It creates trends and patterns in data.

The data may be summarized as follows [5]:

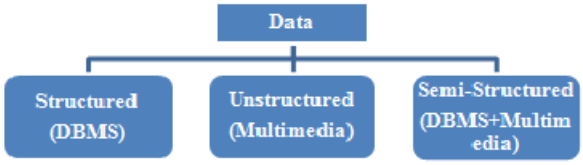


Figure1. Data Summarization

4. THE ITERATIVE PROCESS IN KDD

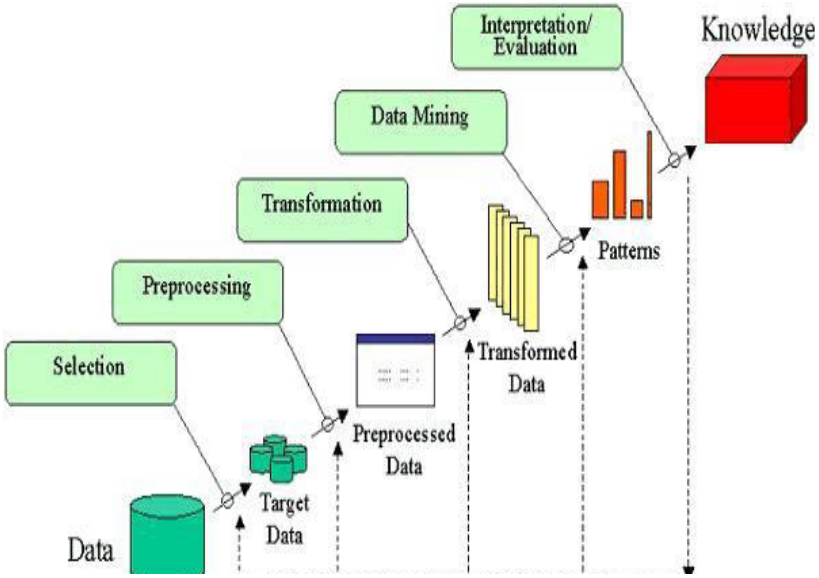


Figure2. Iterative Process in KDD

According, to the Figure2 the following are the iterative steps in Knowledge Discovery [3]:
Data selection (where data relevant to the analysis task are retrieved from the database)
Data cleaning (to remove noise and inconsistent data)

Data integration (where multiple data sources may be combined)

Data transformation (where data are transformed and consolidated into forms appropriate for mining by performing summary or aggregation operations)

Data mining (an essential process where intelligent methods are applied to extract data patterns)
Pattern evaluation (to identify the truly interesting patterns representing knowledge based on interesting measures)

Knowledge presentation (where visualization and knowledge representation techniques are used to present mined knowledge to users)

5.DATA MINING FUNCTIONALITIES

In general data mining functionalities used to specify kinds of patterns to be found in data mining tasks [3].Such tasks are classified into two categories:

5.1. Descriptive:

Descriptive mining tasks characterize properties of the data set. Clustering, summarization, association rules and sequence discovery are usually viewed as descriptive in nature.

5.2. Predictive:

Predictive mining tasks perform induction on the current data to make predictions. Predictive modeling may be made on the use of other historical data. Classification, Regression, Time series analysis and prediction are predictive in nature. The Figure-3 shows task in data mining:

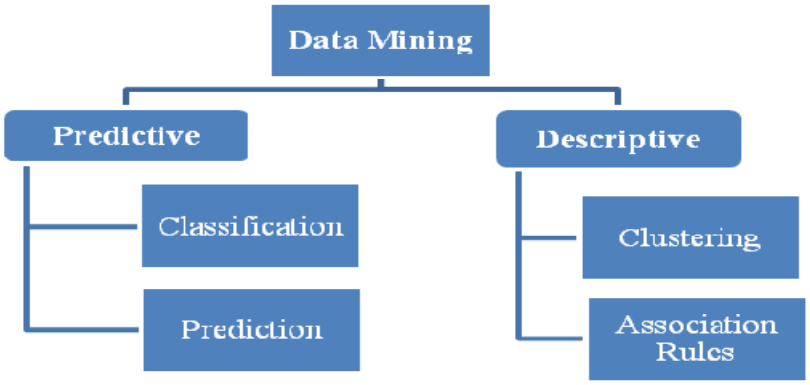


Figure 3. Task in Data Mining

5.2.1. In data mining the data is mined using two learning approaches [6]. They are

A. Supervised learning:

Supervised learning is also called directed data mining. The variables under investigation are split in two groups.1. Explanatory variables, 2.one or more dependent variables.The goal of analysis is to specify a relationship between the dependent variables and explanatory variables as we do it in regression analysis. To proceed with directed data mining techniques, the value of the dependent variable must be known for a sufficiently large part of dataset.

B. Unsupervised learning:

In unsupervised learning, all the variables are tested in the same waythere is no distinction between dependent and explanatory variables. However, in contrast to the name undirected data mining, still there is some target to achieve. This target might be as data reduction as general or more specific like clustering. The divided line between unsupervised learning and supervised learning, in the same way distinguishes discriminate analysis from cluster analysis. Supervised learning requires target variable which should be well defined and sufficientnumber ofits value are given. In unsupervised learning target variable, has only been recorded for the small number of cases as the target variable is unknown.

5.3. Tasks in data mining:

Data mining was divided for the specific classes of six activities or tasks as follows. In this classification, estimation, prediction belongs to supervised learning and association rule, clustering, description& visualization belongs to unsupervised learning.

5.3.1. Classification: classification consists of examining the features of a newly presented objectset consist of reclassified examples. The tasks build a model that can be applied to unclassifieddata which classify it.

5.3.2. Estimation: Estimation deals with continuous value outcomes. Given some input datawe use estimation to come up with a value for some unknown continuous variables.

5.3.3. Prediction: Any prediction can be thought of as classification or estimation [7]. Predictive tasks feel different because the records are classified according, to some predicted future behavior or estimated future value. With prediction, the only way to check the accuracy of the classification is to wait and see. The historical data is used to build a model that explains the current observed behavior. When this model is applied to current inputs, the result is prediction of future behavior.

5.3.4. Association: An association rule is a rule which implies certain association relationships among a set of objects (such as "occur together"(or) "one implies the other") in a database. Given a set of transactions, where each transaction is a set of literals (called items), an association rule as an expression of the form x, y, where x and y are set of items. The intuitive meaning of such a rule is that transactions of the database which contains x tends to contain y.

5.3.5. Clustering: Clustering analysis can be used as a standalone data mining tool to gain insight in data distribution, or a preprocessing step for the other data mining algorithms operating on the detected clusters. Many clustering algorithm have been developed and are categorized from several aspects such as partitioning methods, hierarchical methods and grid based methods. Further data set can be numeric or categorical. Clustering is the task of segmenting a diverse group into number of similar subgroup (or) clusters. The difference between clustering & classification is that cluster does not rely on predefined classes. In clustering, there is nopredefined class, the records are grouped together based onself-similarity.

5.3.6. Description &Visualization: Data visualization is a powerful form of descriptive data mining. It is not always ease to come up with meaningful visualizations, but the right picture really can be worth a thousand association rules, since the human beings are extremely practiced at extracting meaning from visual scenes.

Knowledge discovery goals are defined by the intended use of the system. There are two types of goals. They are verification and discovery. With verification, the system is limited to verifying the user's hypothesis. With discovery goal is further divided into prediction, where the system finds patterns for predicting the future behavior of some entities and description, where the system finds patterns for presentation to user in human understandable form.

6. REVIEW

Steel consists of alloy iron, carbon and manganese with small amount of silicon, phosphorous and Sulphur. Steel production stages: Heating, cooling, melting, solidification. India is 3rd largest of crudesteel(up from eighth in 2003) and is expected to become 2nd largest producer in near future. Most industrial applications of data mining in steel industries is system modeling,approaching new manufacturing technologies and to improve the quality of products,anti-corrosive properties, the galvanized steelis a product experiencing an increasing demand in multiple sector.Nonlinear modeling comprises important techniques that have reached broad applicability due to efficiency in speed. The aim of data mining analysis was to predict the yield strength,elongation as functions of large number of variables, including the chemical composition, heat treatment, strip speed in

the annealing process.It is also being used in industrial process optimization and control. This extensive study is to analysesexisting techniques used in blast furnace in steel industries.

Themain objective of a blast furnace is to reduce chemically and convert physically iron oxides into liquid iron, which is called hot metal. Hot metal manufacturing process consumes about 70% of the entire energy of steel manufacturing integrated route. Besides, social and industrial needs for iron steel, high prices of raw materials and reducing agents have also increased the necessity to model this complex process to increase productivity and reduce cost[2]. To implement this, data mining techniques have been tried in many stages of blast furnace operations.

According, to G. J. Zheng, W. Zhang, P. Hu& D.Y Shi, [8] in this process of hot forming, many design variables have effect on the resultdifferent and complex way such as geometry features and forming process parameters. It is difficult to understand the relationship between design variables and results, which is very important to guide the design. In this paper data mining was introduced to explore the influence the past geometric feature and the hot forming parameters on hot forming results of an automobile B-pillar model and the optimum parameter ranges were determined. First several variable parameters were selected and 100 groups of experimental data were generated with Super LatinMethod and then finite element method analysis results were calculated respectively. Next analysis and evolution of simulation were carried by making use of Decision Trees(DT) algorithms. Finally, a series of B-pillar hot forming rules were refined, such as initial temperature of the sheet metal should be controlled between 720.c to 800 c. Therefore, a real b-pillar model was designed to test the rules and result was correct and effective.

In this work [9] authors developed a decision support system that can determine corrosion and project the time for corrosion growth for maintenance using probabilistic modelling approach. In this work, Data Mining based corrosion control process is introduced, the process is aimed at preventive and predictive maintenance instead of the existing practice which uses fixed scheduled inspection and using corrosion sensors continuously to facilitate preemptive actions. The use of “Data Mining” process includes analyzing historical data collected over time about known metal thickness previously exposed to environment factors such as rainfall, temperature, humidity & soforth. The results of the analysis provide support for decision making by taking proactive and knowledge driven decisions on corrosion control. Classification method like Bayes theorem approach, Neural networks were used to model uncertainties in corrosion occurrence considering both knowledge uncertainties and data uncertainties to make information decisions. The probabilistic modeling is used to develop a data mining method that can be applied to obtain information from a database on metals previously exposed to an atmosphere environment.

The authors Mohamad Saraee, MehdiMoghimi&AyoubBagherisuggests [10],theannealing process is one of the important operations in production of cold rolled steel sheets, which influences the final product quality of cold rolling mills. In their process, cold rolling coils are heated slowly to a desired temperature and then cooled. Modelling of annealing process (prediction of heating and cooling time and trend prediction of coil core temperature) is very sophisticated and expensive work. Modeling of annealing process can be done by using thermal models. In this paper modelling of annealing process is proposed by data mining techniques due to high speed in data processing, acceptable results is obtained and its simplicity to use it. In this study, they proposed a method for modeling the annealing process by using data mining techniques. After testing different techniques of data mining, the feed forward back propagation neural network is selected to predict heating and cooling times and temperature of coil core during annealing process. The result of predictors applied neural networks for modeling annealing process were accurate enough, but while we use a larger data set accuracy of predictors will be improved. The present method is applicable to predict the behavior of processes that cannot be described byany analytical or physical equations. The other techniques such as

classification, different regressions or clustering method can be applied to optimize the input parameters of annealing process for maximizing productivity of annealing operations.

According, to Sayed Mehran sharafi & Hamid Reza Esmaeily[11], applying data mining methods to predict defects on steel surface was challenging one. In the steel industry, especially alloy steel, creating different defected product can impose a high cost for steel producers. One common defect in producing low carbon steel grades is Pits &blisters defect. Its drawback is waste of time and cost to eliminate this drawback, we need to grind the surface of the product. In some cases, the severity of defects may lead to scrap part of the product. Grinding cause waste of time and cost of production will be increased. Incidence of defects is related to several factors including material analysis and production processes. In this study authors created a model to predict this fault with data mining methods including decision tree, neural network respectively. They applied these techniques to the data collected from Iran alloy steel company. The model created using decision tree has higher accuracy. The benefits of this model are

* Reduces the process time and energy cost by omitting grinding step.
* There is an optimal tool for predicting defect and applying for producing products with no defects.

According, to the authors [12] this paper revealed a data mining approach for variable selection and knowledge extraction from the dataset. The approach is based on unguided symbolic regression (every variable present in the dataset is treated as the target variable in multiple regression runs) and a novel variable relevance metric for genetic programming. The relevance of each input variable is calculated and a model approximating the target variable is created. The genetic programming configurations with different target variables are executed multiple times to reduce stochastic effects and the aggregated results are displayed as a variable interaction network. This interaction network highlights important system components and implicit relations between variables. The whole approach is tested on a blast furnace dataset, because of the complexity of the blast furnace and the many interrelations between the variables.

Many variables in the blast furnace process are implicitly related, either because of the underlying physical relations or because of the external control of blast furnace parameters. Examples for variables with implicit relation to other variables are the flame temperature or hot blast parameters. Usually suchimplicit relation was not known a-priori in data-based modeling scenarios but could be extracted from the variable relevance information collected from multiple GP runs.

Using an unguided symbolic regression data mining approach several models have been identified that approximate the observed values in the blast furnace process rather accurately. The experiments also lead to many number of model describing several components of the blast furnace. The generated model is used to extract information about implicit relations in the dataset to further reduce and disambiguate the set of relevant input variables.

The next paper also describes application of symbolic regression on blast furnace using temper mill dataset[13]. It reveals the application of an adapted symbolic regression system on two different datasets, the first one contains measurements from blast furnace process which is most common to produce hot metal(liquid iron). Although chemical and physical process are well understood the heat loss in certain areas of the furnace are not completely understood. The knowledge about such relationships can be used to optimize the blast furnace process and therefore modeling the blast furnace process based on collected real world data is of specific interest. Regression analysis is a sub field of data mining attempting to reveal knowledge contained in given set. In this work, symbolic regression is performed using a tree-based on genetic programming system to evolve mathematical formulas. Genetic programming [14] is an

evolutionary algorithm that produces programs to solve a given problem. In this modeling approach three algorithmic aspects are incorporated and compared to a standard symbolic regression approach. Precisely the authors tested the effects of off-spring selection, using the coefficient of correlation R^2 as fitness function and additionally sample the evaluated samples or fitness cases of individual. Off-spring selection is an additional selection setup in genetic algorithms and genetic programming that is applied after recombination and mutation. The fitness of an individual in symbolic regression analysis is commonly calculated as a Mean Square Error (MSE) between the predicted value of the model and the observed value of the target variable. The genetic algorithm is faster than the genetic algorithm without off spring selection. In this contribution three algorithmic adaptations, off spring selection, coefficient of determination R^2 as fitness function and sampling, to a rather standard symbolic regression system have been investigated. The effects of combining these adaptations have been demonstrated on real world data sets from two steel production processes. The best improvements in terms of quality were achieved due to the use of R^2as fitness function.

According, to Jong-Hag Jeon [15], this presentation is an example of Data Mining Applications at steel making factory. Engineers who are responsible for quality control encounter two kinds of difficulties in solving problems using statistical methods. Hot blast stoves are important to energy consumers. Hot blast sensible heat supplies 10%-15% of the total energy consumed by the blast furnaces. Reduction in energy consumption for hot blast heating has important impact on hot metal production cost.

The productions of industrial blast furnaces require the constant delivery of hot blast air from blast furnace stoves. The blast air must be supplied with temperature and flow-rate as close as set point. In the long term, the control must also meet the conflicting objectives of thermal efficiency, cyclic stability and load response speed. Optimizing combustion conditions contributes largely not only to the prolongation of hot stove service life, but also increases in thermal efficiency of the stove during operation. This paper shows the hot stove combustion analysis both theoretically and experimentally to optimize combustion condition preventing excessive development of combustion flames as well as the abnormal heating effect mainly after burning. In addition, the development feed forward guidance simulator is based on Neural Network Models and multi-stage combustion pattern control systems developed through Data Mining. The introduction of these systems required only minimal operation costs, decreasing the hot blast stove energy cost approximately 15%. The author had developed this system with SAS for no.3 blast Furnace in Pohang steel works, Korea.

High productivity is achieved at Pohang No.3 blast furnace through the following process:

*Optimization of heat efficiency: automatic combustion control system, calorie control system.
*Application of new control concept: Multi stage combustion control system.
*Optimization of operation condition using data mining.
*Increase of model accuracy using Neural Network Model

POSCO blast furnaces will focus on high productivity operation including high efficiency and long campaign life through consistent technology development.

The following paper revealed [16] the use of data analytics tools for predicting the fatigue strength of steels. Several physics-based as wellas data-driven approaches have been used to arrive at correlations between various properties of alloys and their compositions and manufacturing process parameters. Data-driven approaches are of significant interest to material engineers amount to Neural Networks,Reduced error pruning trees, MS model trees(Reconstruction of Quinlan's algorithm). In this study, a range of advanced data analytics techniques, typically involving a combination of feature selection and regression methods, have

been successfully employed and critically evaluated for the problem of fatigue prediction of different grades of steels.

They tested with 12 predictive modeling techniques in this study, which includes the following: Linear regression, pace regression, regression post non-linear transformation of select input variables, Robust fit regression, multivariate polynomial regression, K-nearest neighbor (KNN) modeling, Decision table, Support Vector Machines, Artificial neural networks, Reduced error pruning trees. Thus in particular, neural networks, decision trees and multi-variate polynomial regression were found to achieve a high R^2 value of greater than 0.97, which is significantly better than what has been previously reported in the literature. It is very encouraging to see that despite the limited amount of data available in this dataset, the data-driven analytics models could achieve a reasonably high degree of accuracy.

The authors suggest that [17] Iron and steel melting is an extremely complex physical and chemical process. Blast furnace is main method of iron and steel melting .To obtain the knowledge has become more and more difficult task on manual analysis of the data. Matured theory like data mining and knowledge discovery are applied to acquire new knowledge. Data mining methods are used in the blast furnace production control. Firs the authors introduced clustering algorithms. Then clustering analysis of blast furnace operation parameters was carried out by K-means clustering. Analysis and comparison of practical data was conducted to determine the optimal cluster number of this algorithm for blast furnace parameters analysis, and this yielded the ideal operating value for the parameters. The optimal threshold for blast furnace parameters were determined through statistical analysis, repeated experiments and field assessment, and the difference between blast furnace state as estimated and the practical one analyzed. Finally, the factor analysis method to reduce the dimension of parameters was successful and mining test of Tangshan iron and steel shows that the method is effective in practical application.

According, to the authors [18] the blast furnace for steel making process is examined. As the process is conducted at extreme conditions, it is impossible to observe what is occurring inside the blast furnace. Nevertheless, optimizing the process would greatly improve the overall process and quality of the final product. By implementing the ANFIS (Artificial Neural Fuzzy Inference System) model on collected blast furnace data, they aimed at optimizing the blast furnace process more specially predicting the performance indicator $nc_{o.}$ which describes gas utilization rate in the furnace. These studies used linear modeling techniques to study the effects of several explanatory variables of the silicon content in the hot metal part of the output. The results were promising but were outmaneuvered by nonlinear soft computing techniques. The nonlinear soft computing techniques have been applied to blast furnace process modeling. These studies have used neural network-based hybrids to model furnace performance indicators with good results. A well-known data mining framework, CRISP-DM, was used to guide the analysis and modeling process. The ANFIS model was completely data- driven except for some expert knowledge that was utilized in the pre-processing stages and in the construction of final input series, were several series consists of mathematical combinations of two or more raw input series. The adequate performance of ANFIS combined with a proposed pre-processing approach resulted in a system which is feasible in real world applications.

According, to the authors this study [19] described one of possible approaches to deoxidant cost optimization in steel production based on the expert system. Education of the system is based on successfully completed heats. Decision support Systems(DSS) that uses intelligent conclusion (including fuzzy)are used in industry, particularly in the ferroalloys production. The authors used Naïve Bayes network and decision tree(ID3). But decision tree did not go well as naïve Bayes. So, authors used Naïve Bayes network for proposed production model due to its speed, simplicity and ease of interpretation of results. However, this method does not allow the direct processing

of continuous variables – they must be divided into number of intervals to discretize values.DSS on Bayesian network shows higher accuracy of approximation at the training and test sample.
According, to the authors [20] the products manufactured by manufacturing company products may vary in quality than expected quality in terms of many parameters. Due to rapid development in data mining and machine learning industries started to do analysis in data mining. In steel quality of product starts from blast furnace process, so they started using data mining techniques to improve the product quality. This existing work used random forest method to detect mild steel defect diagnosis. The Random forest technique is not used widely in steel industry. This paper taken initiative to do it and they found that random forest technique gives more accuracy (97%) than ANN and single decision tree. This paper focused on enhancing the accuracy and efficiency [21] with which operating set points are calculated for a Continuous Annealing Furnace (CAF) on a Hot Dip Galvanizing Lines(HDGL). Recent papers have proved that data driven models based on artificial Intelligence are a good alternative for developing accurate prediction models in the steel industry. To take advantage of these techniques in the modelling of the galvanizing process, the main requirements are the availability of historical data, detailed knowledge of the chemical composition of steel and time to train the models. Now Ensemble Methods give high overall performance in predicting system set point. Ensemble Methods are built using a set of model and the final output is a combination of the outputs of each individual model.

In this paper these Ensemble Methods Additive Regression(AR), bootstrap aggregating (bagging) are developed to adjust CAF temperature settings in an HDGL. Additionally, five data driven models least median squared linear regression, linear regression, Quinlan's improved M5 algorithm(M5P), multilayer perceptron neural network, Support Vector Machine(SVM) are chosen from literature as basic components of ensembles. The results for furnace temperature set point predictions obtained with this method highlighted the benefits of using EM rather than other data driven models. Multilayer perceptron with basic learner (MLP with BP) worked well in this model.

6.1According to thereview [22] the Figure4 and figure 5 reflects the percentage of application of data mining in steel industries :

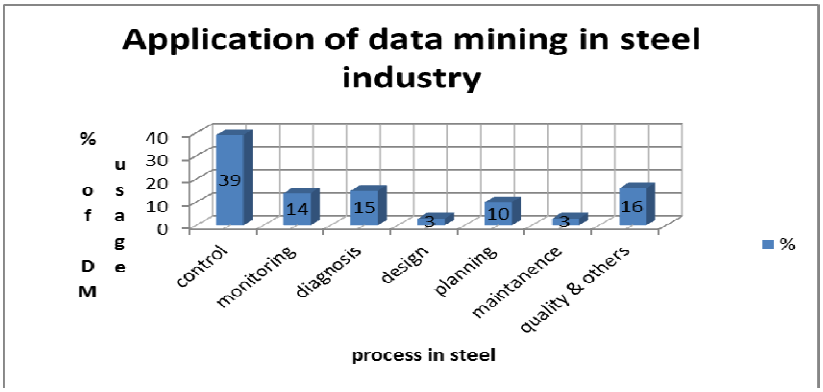


Figure 4. Application of data mining in steel industry

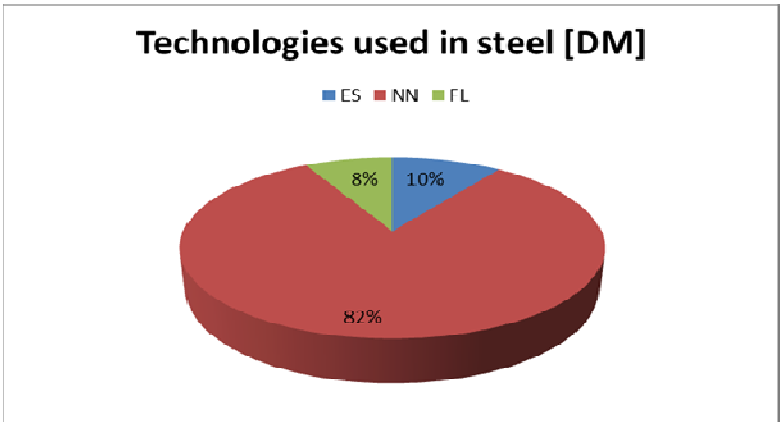


Figure5. Technologies used in Steel [DM]

Abbreviations used in this chart-

DM – Data Mining	NN-Neural Network
ES – Expert System	FL-Fuzzy Logic

7.CONCLUSION

Data mining techniques in steel industries shows excellent results when it is used for the following purposes

- Product quality monitoring
- Process monitoring
- Maintenance strategies

Also, it seems to be clear that data mining can be improved by using numbered modelling simulation where real time data are not available. In all cases, the impact of these technologies in modern industrial process make a requirement for companies to be aware of this to extract knowledge from the process and can improve them. The intension of thissurvey is to present a review of current work related to usage of data mining techniquesin steel industry in Iron making (blastfurnace) and in steel making process. Further research directions will continue in this field. Especially Random Forest and yet more machine learning techniques can be tried,since these techniques are applied in less frequencies than Artificial neural networks, Fuzzy logic techniques in steel industries modelling process.

REFERENCES

[1] A.K. Choudhury, M.K. Tiwari and J.A Harding, (2009), 'Data Mining in Manufacturing: A review based on the kind of knowledge'. Wolfson school mechanical andmanufacturing engineering, Loughborough university, Loughborough, Leicestershire, UK, Journal of Intelligent manufacturing, 20(5), pp. 501-521.

[2] Rosiane Mary Rezende Faleiro, Claudio Musso Velloso, Luiz Fernando Andrade De Castro, Ronaldo Santos Sampaio, (2013), 'Statistical modeling of charcoal consumption of blast furnace based on historical data: Journal of Materials research and technology',2(4), 303 – 307.

[3] Jiawei Han, Micheline Kamber, Jian Pei, (2012), Data Mining: Concepts and Techniques, Third Edition, USA, Morgan Kaufmann Publishers.

[4] Nine law of Data Mining by Tom Khabaza (http://wwwkdnuggets.com/2015/16/nine-laws-datamining-part-1:html).

[5] Hand D. J, Manila H, & Smyth, (2001): Principlesof Data mining, MIT press, Cambridge, Massachusetts. ISN-262-08290-X

[6] Aastha Joshi and Rajneet kaur, (2013), A review: 'Comparative study of various clustering techniques in data mining', International Journal of Advanced Research in Computer Science and Software Engineering, Vol 3,2277 128x.

[7]Manisha Verma, Mauly Srivastava, Neha Chack, Abul Kumar Diswar, Nidhi Gupta, (2012), 'Comparative study of various clustering algorithms in data mining'. International journal for engineering research and applications, Vol 2, Issue 3, pp. 1379-1384.

[8]G.J Zheng, W. Zhang, P. Hu & D.Y SHI, (2015), Optimization of hot forming process using DMT and Finite element method, International Journal of Automative Technology, Vol 16, no.2, pp: 329-337.

[9] Stephen Dapiap, Gregory Wajiga, Michael Egwurube, Musa Kadzai, Nathaniel Oye &ThankGodAnazodo, (June2015), Corrosion Control Approach using Data Mining, International Journal of Computer Science & Information Technology(IJCSIT), vol 7, No 3.

[10] Mahamad saraee school of computing, science and Eng., university of Salford, greater Manchester, UK, Mehdi Moghimi, Dept. of Elec. & computer Eng., Islamic Azad university, Najafabad branch, Isfahan, Iran, Ayoub bagheri, Dept. of Elec and computer Eng, Isfahan university of technology, Isfahan Iran, (2011), Modeling Batch Annealing Process using Data Mining Techniques. ACM journal.

[11] Sayed Mehran Sharafi, Hamid Reza Esamaely, (2005-2010), Applying data mining methods to predict defects of steel surface, Journal of theoretical and applied information technology, [www.jatit.org].

[12]Michael Kommenda, Gabriel Kronberger Christoph Feilmayr and Michael Affenzeller, (23 Sep 2013), Data mining using unguided symbolic regression on a blast furnace dataset, arXiv;1309.5931v1 [cs.NE].

[13] Michael Kommenda, Gabriel Kronberger, Christoph Feilmayr, Leonhard Schickmair, Michael Affenzeller, Stephan Winkler and Stefan Wagner, Application of symbolic regression on blast furnace and temper mill datasets, [n.d].

[14] John R. Koza, Consulting Associate Professor in computer science department Stanford university, Genetic programming: On the programming of computers by means of natural selection, the MIT press [1992].

[15] Jong-Hag Jeon, POSCO, Pohang South Korea, Data mining application of six-sigma project, SUGI 29 solutions, paper 186-29.

[16] Ankit Agarwal, Parjit D Deshpande, Ahmet Cecen, Gautham P Basavarsu, Alok N Choudary and Surya R Kalidindi, (2014), Exploration of data science techniques to predict fatigue strength of steel from composition and processing parameters, Integrating materials and manufacturing innovation, 3:8, A springer open journal.

[17] Fuxing Yu, Yina Suo, Xin Zang, Aidind Yan, Fulong Liu, (2013), Data mining in blast furnace smelting parameter, Applied mechanics and materials, vol. 303-306, pp 1093-1096.

[18] Bjork, Holopainen, Wikstron, Saxen, Carelsson and Sihdonen, technical report number 1094, (Nov 2013), Analysis of blast furnace time series data with ANFIS: Turku center for computer science [TUCS].

[19] Zheldak T.A, Slesarev V.V, Volovenko D.O, (2013), Knowledge-based intellectual DSS of steel deoxidation in BOF production process, American Journal of Mining and Metallurgy, Vol. 1, no.1, 7-10.

[20] Veena Jokhakar, S.V Patel Ph.D., (March 2015), A Review of Business Intelligence Techniques for Mild Steel Defect diagnosis, International Journal of Computer applications (0975 – 8887), volume 113 – No 10.

[21] Sanz-Garcia, F. Antonanzas-Torres, J. Fernandez-Ceniceros & F.J. Martinez-De-Pison (2014), Overall models based on ensemble methods for predicting continuous annealing furnace temperature settings, Iron and Steel Making, vol. 41, issue no 1.

[22] Radu Platon & Mouloud Amazouz,From Report CETC – Varennes September 2007 -141 (TR),Application of data mining techniques in Industrial Process Optimization, Prepared by CANMET energy Technology Centre, http//www.nrcan.gc.ca>2007-141e.

Practical 9

Case Study on Quality Control

Diagnosis of quality management systems using data analytics – a case study in the manufacturing sector

Sanchez-Marquez R[1], Albarracin Guillem JM[2], Vicens-Salort E[3], Jabaloyes Vivas J[4]

Abstract: The main objective is to improve customer satisfaction by developing and testing a method to study quality management systems by analysing the key performance indicators of balanced scorecards in manufacturing environments. The methodology focuses on the identification and quantification of relationships between internal and external metrics that allow moving from performance measurement to effective performance management. It has been tested as a case study approach using real data from two complete years of the balanced scorecard of a leading manufacturing company. The results provided a new understanding of how the quality management system works that was used to make systemic and strategic decisions to improve the long-term performance of the company. Industry practitioners with a moderate level of data analytical skill can use it to help managers and executives improve management systems.

Keywords: Manufacturing, quality management system, data analytics, balanced scorecard, key performance indicators

The final version of this article with full bibliographic details is available online at: https://doi.org/10.1016/j.compind.2019.103183

1. Introduction

Although performance measurement is not an end in itself, the literature identifies it as an essential part of performance management, since a lack of appropriate performance measurement can be a barrier to change and improvement [1]. Bititci et al. [2] claim that reviewing and prioritising internal goals when changes in the internal and

[1] Rafael Sanchez-Marquez (email: rsanch18@ford.com)
Doctorate Student (corresponding author)
Production Management and Engineering Research Centre
Universitat Politecnica de Valencia
Camino de Vera, s/n. 46021 Valencia. Spain
ORCID: 0000-0001-9071-9550

[2] José Miguel Albarracín Guillem (email: jmalbarr@omp.upv.es)
Departamento de Organización de Empresas
Universitat Politècnica de València

[3] Eduardo Vicens-Salort (email: evicens@cigip.upv.es)
Production Management and Engineering Research Centre
Universitat Politècnica de València
Camino de Vera, s/n. 46021 Valencia. Spain
ORCID: 0000-0002-8111-5269

[4] José Jabaloyes Vivas (email: jabaloye@eio.upv.es)
Department of Statistics, Operational Research and Quality
Universitat Politecnica de Valencia
ORCID: 0000-0003-3411-2062

Declarations of interest: none

external environment are significant is an important feature of effective performance measurement systems. As a performance management system, the balanced scorecard (BSC) establishes the importance of knowing and using the cause-and-effect relationships between internal and external metrics to move from performance measurement to effective performance management [3].

This paper develops a methodology that allows practitioners to identify and quantify those relationships by using key performance indicators (KPIs) of the BSC. Since the objective is to develop a practical methodology that uses real BSC data, its development focuses on the integration of appropriate methods and tools for data analytics.

The present method was developed and tested in a leading multinational manufacturing company, which had implemented a balanced scorecard for the production facilities composed of seven management/operating systems [4]: safety; quality; delivery; cost; people; maintenance; and environment. The quality management system (QMS) was selected by the directors of the company to develop and test the validity of the method, since it was the system with the highest level of complexity. Nevertheless, with small adjustments the method can be applied in the other six management systems in the same way as in quality.

According to the international standard ISO 9001:2015 that specifies the requirements for a quality management system, industrial products and their manufacturing processes must be designed to meet customer expectations through the specific engineering specifications of critical product characteristics. These are typically specified in terms of a nominal (ideal) value and a tolerance interval (upper specification limit - lower specification limit). Controlling and managing these critical characteristics is a fundamental task of the quality measurement system and, therefore, of the quality management system. When critical characteristic measurements meet engineering specifications, they also meet customer expectations, leading to customer satisfaction. These measurements are summarised in the internal KPIs of the QMS. Therefore, the QMS includes internal KPIs, which summarise compliance with engineering specifications, and external KPIs, which include customer complaint indicators. Consequently, if the quality management system works well, internal and external KPIs must reflect customer satisfaction and, therefore, both sets of indicators must be highly correlated.

Identifying which internal KPIs drive customer satisfaction (external KPIs) and quantifying such relationships allows executives and managers to design strategies to

improve customer satisfaction, which is the main objective of this research work. In addition, the results serve as a start point to reduce the complexity of the quality management system (QMS). Simplification of performance management systems (PMSs) is a recurring topic in the literature [5, 6, 7]. Therefore, the following two main research questions were established:

- How do the KPIs of the QMS relate to each other?
- How can these KPIs help improve customer satisfaction?

There is some research on the development of analytical methods based on the key performance indicators of balanced scorecards in the manufacturing environment. However, the results of these works [8, 9, 10, 11] are qualitative rather than quantitative (which should be the nature of any analytical method). Therefore, the development of robust analytical methods for manufacturing systems based on proven scientific tools is an issue that has not been covered in the literature. This paper focuses on the diagnosis of a management system to improve its capabilities and this implies a novel approach.

This work was carried out as part of a collaborative research project between the company (which requested to keep its identity and data confidential) and the Centre for Research and Production Management of the Polytechnic University of Valencia (Spain) to improve management methods in manufacturing environments.

The company decided to use the findings of the present study to make changes in the balanced scorecards of all production facilities worldwide. Although these changes are detailed in the results section of this paper, they can be summarised as a reduction in the complexity of the operating system and the inclusion of new KPIs, as well as the elimination of some existing indicators that have shown less strategic weight. The new insight provided by this study was used to prioritise some strategies over others and start new strategies to improve customer perception about the quality of company products.

The method was validated using real data from two complete years of key quality performance indicators as a case study approach.

2. Literature review

The literature review was structured to cover the relevant topics:

- Analytical methods applied to key performance indicators using actual data

    - Regression, multiple linear regression (MLR), partial least squares (PLS), principal component analysis (PCA), time series, artificial neural networks (ANN), data mining
- Analytical methods applied to building balanced scorecards as a proactive tool
    - Fuzzy logic, analytic network process (ANP)
- The balanced scorecard in the manufacturing environment
- Limitations of the analytical tools mentioned above
- Limitations of the balanced scorecard model
- Quality management systems in the manufacturing environment

The main objective of the literature review was to identify the best possible approach and the strengths and limitations of each method available in the literature. As discussed in the introduction section, the present method covers a new objective, although to some extent it is based on improvements in existing methods developed by other authors and applied for other purposes. In addition, it addresses the limitations already commented by the authors themselves.

2.1. Analytical methods applied to KPIs using actual data

The available works use analytical tools such as MLR [12], PCA and PLS [13, 14, 15], and graphic methods [16], to assess the effectiveness of the strategies in place and quantify their impact on the output metrics. Sanchez-Marquez et al. [16] suggest previously selecting the output metrics among all the key performance indicators (KPIs) included in the scorecard to streamline the method as a key step in any method that addresses the KPIs. While some comments are made about the need for more perspective to understand how the system works, this goal is beyond the scope of those works.

2.2. Analytical methods applied to the BSC as a proactive tool

Other works focus on proactive methods to build a balanced scorecard by selecting the best key performance indicators when sufficient data is not yet available. These works use other techniques – such as ANP [17] or fuzzy logic [18, 19]. Although the effectiveness of these methods proves that it works in the construction of new information systems as a proactive approach, this document focuses on making the most of the data available from existing information systems.

2.3. Regression methods

2.3.1. Multiple linear regression

MLR has been used to quantify the effect of input metrics on the output [12, 15] with good results in terms of model predictability (R^2). However, the main objective of the present study, which is to discover systemic relationships, can be compromised by the effect of collinearity. MLR when affected by collinearity, which can be measured by the variance inflation factor (VIF), can produce an unstable model since coefficients are overestimated when VIF > 5. In addition, the MLR, as a regression technique, must assume cause and effect relationships between the variables before evaluating the model, which are not sufficiently clear in this case, at least as a starting point.

2.3.2. Partial least squares

For complex models (e.g., high-order constructs) or cases with multi-collinearity, PLS is more appropriate [20]. Moreover, PLS can be used even if the number of observations is smaller than the number of variables to study [13]. However, the uncertainty of the construct in the initial stages of the study is the most difficult obstacle to overcome [20]. Rodriguez-Rodriguez et al. [13] highlighted this uncertainty in a study where the research team had to evaluate different constructs together with the team of the board of the company where the study was made.

Although PLS is generally the preferred method when a regression analysis is required, MLR also has some points in its favour, such as the possibility of evaluating non-linear relationships between predictors and dependent variables. PLS is a multivariate technique, so it uses linear algebra, and although the transformations of the variables can be used to explain nonlinear relationships, it is not recommended, since the number of variables increases exponentially, and multivariate techniques are not adequate for such models in practical terms [21].

2.3.3. Simple linear regression

Simple linear regression (SLR) can also be an option when the problem is to understand the relationships between different levels or dimensions and only two variables are being studied. However, depending on the nature of the problem, several regression techniques can be applied, and the practitioner will always have to consider the principle of parsimony (which is to keep the model as simple as possible). The principle of parsimony can generally be considered a good guide when applying statistical tools [22, 23]. However, in social sciences, Gunitsky [24] recommends distinguishing between three different views of the concept according to the objective. He emphasises the epistemological conception of parsimony – abstract from reality – to

highlight recurring patterns and construct verifiable propositions. Therefore, Gunitsky [24] suggests that to prove a specific hypothesis, the principle of parsimony is justified, coinciding fundamentally with Coelho et al. [22] and Nalborczyk et al. [23].

2.4. Principal component analysis

Several studies [13, 14, 15] have shown that PCA is an effective tool for selecting KPIs. Bi-dimensional plots of principal components can be used to screen the main KPIs for their weight, but also to perform a more comprehensive correlation analysis than just looking at the table with the loads of each variable for each component. Rencher [25] pointed out that this analysis can be an integral result by itself if a qualitative analysis is carried out together with the quantitative analysis.

2.5. ANN and other data mining techniques

ANN and other data mining techniques are more suitable in big data contexts [26], since these methods work well when number of instances is much bigger than the number of variables (KPIs), which in principle is not the case when dealing with KPIs of the BSC. In addition, ANN does not provide an explicit regression equation compared to other regression techniques, which was considered essential for the purpose of this research. Therefore, the present methodology does not use data mining methods such as ANN.

2.6. Quality management systems in manufacturing and the BSC

The main studies on QMSs are more qualitative than empirical and analytical [27, 28, 29], mainly in the manufacturing sector [30]. Although the quantitative analysis was performed in the QMS, the approach was to generate a construct using PLS-SEM or CB-SEM techniques based on established theoretical frameworks [30, 20].

Norreklit [31] points out that one of the main problems of the balanced scorecard model is the assumption of fixed cause and effect relationships between variables of different dimensions. Instead, she proposes a model with systemic relations where the different dimensions do not have a defined hierarchy or a fixed model. She also mentions the problem of potential delayed effects on the system of some variables. Kaplan [5] recognises that these problems can be present in the model and invites the

scientific community to study how they can be discovered and thereby improve the model using analytical techniques and empirical systems dynamics. Hoque [6], in a comprehensive review of the use and limitations of the balanced scorecard, suggests that the existence of potential trade-offs between KPIs from different dimensions or levels is among the most cited unresolved problems.

2.7. Time series techniques

Time series techniques should be applied to address and solve the problems that this type of data tends to have. The most common problems are autocorrelation or working with non-stationary time series. A hybrid method that combines analytical and graphical tools is the most convenient in these cases [15].

2.8. Synthesis of the literature review

Table 1 summarises the literature review on the existing methods explained in detail in the previous sections. The tools and techniques selected for the proposed methodology are underlined. This selection is based on the characteristics of each tool and those of the problem addressed in this study.

Tool / technique	Type of data	Multivariate/ univariate	Suitable for variable selection	Typical applications
SLR	Actual data	Univariate	Yes	Hypothesis testing, causal models
MLR	Actual data	Univariate	Yes	Medium-complexity models
PLS	Actual data	Multivariate	No	High-complexity models / machine learning
PCA	Actual data	Multivariate	Yes	Feature extraction
ANN	Actual data	Multivariate	No	Deep learning / data mining
Fuzzy logic	Subjective data (from experts)	Multivariate	Yes	Proactive methods / decision support systems
ANP	Subjective data (from experts)	Multivariate	Yes	Decision support systems
Time series	Actual data (time domain)	Both	Yes	Data pre-processing, econometrics, forecasting

Table 1. Analytical methods

In the next section, the method used to carry out the study is presented as a multi-phase model. This method was designed to include all the characteristics and, as far as

possible, improve the limitations of the techniques selected from those identified in the literature review.

3. Data and methods

The methodology developed has been tested in a case study approach using real data from two full years of the balanced scorecard of a leading manufacturing company. The company where this work was done considers the raw data used to be confidential and its representatives and the university research team signed a confidentiality agreement. For this reason, this paper only shows the result of the statistical analyses, but not specific values of the key performance indicators of the QMS. To preserve its confidentiality, the scale of the original data has been changed by dividing all data points in the entire original dataset by the same figure. It has been confirmed that by dividing by the same number, all analyses give the same result with the original and the transformed data, since the scales change, but not the ratios between the KPIs. This paper provides the reference to the transformed dataset to allow replication of the main results shown in section 4.1. To ease interpretation, Table 2 shows detailed definitions for all the KPIs used in the study.

KPI	Designation	Units	Definition
D1000 or D1000 ONLINE	Online defects per thousand units	# of defects / 1000 units	Number of defects detected online at any production stage every 1000 units produced
EL D1000	End of line defects per thousand	# of defects / 1000 units	Number of defects detected at the end of the production line every 1000 units produced
EL FTT	End of line first time through	%	Proportion of units produced without defects that need offline repairs detected at the end of the line
FTT	First time through	%	Proportion of faultless produced units that need offline repairs detected at any stage of the production line
ONLINE or ONLINE %	Online percentage	%	Proportion of units repaired online with at least one defect
PA D1000	Final product audit defects per thousand	# of defects / 1000 units	Number of defects detected in the final product audit every 1000 units
PA FTT	Final product audit first time through	%	Proportion of faultless units needing offline repairs detected in the final product audit
PA ONLINE	Final product audit online percentage	%	Proportion of units repaired online with at least one defect detected in the final product audit
PA TGW	Final product audit things gone wrong	# of claims / 1000 units	Number of customer claims per thousand units predicted based on the severity and probability of defects detected in the final product audit
PA TGW A	Final product audit things gone wrong type A	# of claims / 1000 units	Number of customer type A claims per thousand units predicted based on the severity and probability of defects detected in the final product audit
PA TGW AB	Final product audit things gone wrong type A and B	# of claims / 1000 units	Number of customer claims of type A and B per thousand units estimated based on the severity and probability of defects detected in the final product audit
PA TGW B	Final product audit things gone wrong type B	# of claims / 1000 units	Number of customer type B claims per thousand units estimated based on the severity of defects detected in the final product audit
R1000 0MIS	Repair per thousand at zero months in service	# of claims / 1000 units	Number of customer claims per 1000 units due to repairs at zero months in service after product sale
R1000 1MIS	Repair per thousand at one month in service	# of claims / 1000 units	Number of customer claims per 1000 units due to repairs at one month in service after product sale
R1000 3MIS	Repair per thousand at three months in service	# of claims / 1000 units	Number of customer claims per 1000 units due to repairs at three months in service after product sale

Table 2. Definition of the quality management system KPIs

The multi-phase methodology is shown in Figure 1 and the details of each phase are explained below.

The statistical analyses were performed using the statistical software packages Minitab, Stata, and the data analysis tool of Excel.

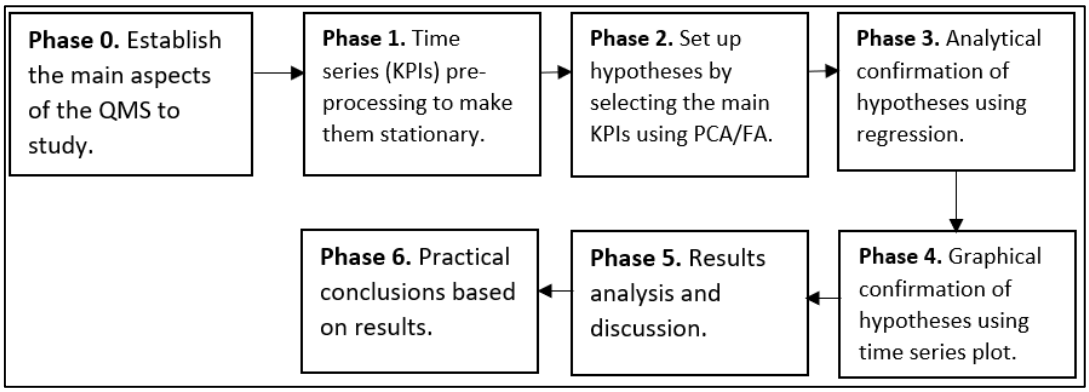


***Figure 1**. Multiphase methodology of the study*

In **phase 0**, the research team together with company experts established that the main aspects of the study were the 'predictability of the quality system' and the 'feedback capability of the quality system'. The predictability of the quality system can also be understood as the ability to control customer satisfaction through internal KPIs. If there were internal KPIs with good predictability, causality, or correlation with external KPIs (related to customers), it would be easy to implement strategies to improve customer satisfaction indexes.

Quality feedback is the ability of the system to recalibrate internal controls in an environment of continuous improvement. The ability to recalibrate quality inspection is vital to ensure the system continues predicting, reacting, and preventing future customer complaints.

In **phase 1**, the raw data must be processed before starting statistical analyses [15]. The main problems when dealing with time series (KPIs) are the autocorrelation and the seasonality of the data. The time series must be stationary before performing statistical analyses that use correlation or regression [32, 33]. Sanchez-Marquez et al. [15] use the Dickey-Fuller analytic t-test augmented for stationary time series [34, 35] complemented by a graphical analysis of the time series with the time series chart, the autocorrelation function (ACF), and the partial autocorrelation function (PACF) [32, 33]. If any sign of non-stationarity is observed, a transformation of the unprocessed data must be performed to obtain a stationary time series. The most common transformation is to take differences, but in some cases, other transformations are needed, such as the logarithmic ones [32, 33, 35, 15].

The main objective of **phase 2** is to select the main KPIs that explain most of the variability observed. Rodriguez-Rodriguez et al. [13] use the two-dimensional plot of the PCA to select those KPIs with the highest loadings (coefficients) before performing

regression analyses (PLS). This eliminates the noise produced in the system by the discarded variables, which results in a more precise estimation of the regression coefficients. In this paper, this quantitative analysis is complemented with a qualitative analysis using the vector view of the two-dimensional plot. As shown in the results section, the closer the vector direction, the more similar are the variables explained by those vectors. This means that there is a high correlation between variables represented by vectors with close directions. These variables with high correlation together with the known manufacturing flow (Fig. 2), which establishes the cause-and-effect relationships between the variables, are used to carry out the regression analyses in **phase 3**, which will quantify the variables relationships in terms of strength (regression coefficients) and stability (R^2-predictive).

As mentioned by Rencher [25], the PCA can be a result in itself when the objective is a descriptive or qualitative analysis. Starting with the data matrix (multidimensional observations), the variance-covariance matrix (usually called the covariance matrix as its shortest form) can be computed as follows:

$$\boldsymbol{S} = \frac{1}{n-1}\widetilde{\boldsymbol{X}}'\widetilde{\boldsymbol{X}} \quad (1)$$

where:

- $\boldsymbol{S}$ is the covariance matrix.
- n is the number of observations or multidimensional instances
- $\widetilde{\boldsymbol{X}}$ is the data matrix centred by subtracting from each data point the mean of each variable (column). Therefore, $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{1}\overline{\boldsymbol{x}}'$, where $\boldsymbol{X}$ is the raw data matrix, $\boldsymbol{1}$ is a column vector composed of n observations or instances, and $\overline{\boldsymbol{x}}'$ is the row vector composed of the means of the m variables in the study. Therefore, since $\boldsymbol{X}$ is an n x m matrix, $\widetilde{\boldsymbol{X}}$ is also an n x m matrix, where n is the number of multidimensional instances or observations, and m is the number of variables considered in the study.

Since $\boldsymbol{S}$ is a square and symmetric matrix, the Eigen analysis can be performed to obtain the eigenvalues and eigenvectors. According to Peña [21], this can be shown in its matrix form as follows:

$$\boldsymbol{SU} = \boldsymbol{UD} \quad (2)$$

where:

- $\boldsymbol{S}$ is the covariance matrix
- $\boldsymbol{U}$ is a square matrix where each value u_{nm} represent the loadings or coefficients of the original m variables in each principal component (p components). The principal components (also known as latent variables) are the column vectors.
- $\boldsymbol{D}$ is a diagonal matrix where each diagonal value (λ_p) represents the eigenvalue of each p component.

Initially, from the Eigen analysis, we obtain the same number of components as original variables (p = m), since $\boldsymbol{U}$ is square. In practical terms, the eigenvalues of some of the components are almost zero ($\lambda \approx 0$), because some variables are not linearly independent of others (high correlation between the variables), so $p \leq m$ and this implies a reduction of complexity.

Since $\boldsymbol{U}$ is a square matrix composed of orthogonal vectors [21], then $\boldsymbol{U'U=U^{-1}U=I}$. If one pre-multiplies (2) by $\boldsymbol{U'}$ on each side of the equation, then

$$\boldsymbol{U'SU = D} \tag{3}$$

and therefore

$$\boldsymbol{S = UDU'} \tag{4}$$

Equation (4) is known as the spectral decomposition of the covariance matrix [21]. The covariance matrix is decomposed into orthogonal vectors (principal components) where each explains a certain amount of variance (λ_p). Therefore, all the variance observed in the original data can be explained by these new variables (components/dimensions).

To obtain the value of the new variables in each observation (principal component scores), the original variables must be projected in the new space, which normally has fewer dimensions due to the reduction in complexity explained above, therefore

$$\boldsymbol{T = \tilde{X}U} \tag{5}$$

where $\boldsymbol{T}$ is a matrix n x p that represents the projected observations in the new space. Note that, as explained above, $p \leq m$ due to the reduction in complexity.

Rodriguez-Rodriguez et al. [13] only use the coefficients ($\boldsymbol{U}$) as the weight to select the variables. Since an original variable can be projected in more than one component, the original variables are characterised by their coefficients and by its direction when they are projected. Therefore, the present method uses the vector view as a graphical method, not only the coefficients.

Peña [21] and Rencher [25] recommend using the correlation matrix instead of the covariance to perform PCA when the variables have different scales, which is a way of standardising the scale of the variables. The balanced scorecard, including each of its operating systems, is composed of heterogeneous groups of variables; therefore, this method must use the correlation matrix as follows:

$$\boldsymbol{C = PLP'} \tag{6}$$

where

- $\boldsymbol{C}$ is the correlation matrix, where the elements outside the diagonal are the correlation coefficients between the variables and the elements of the diagonal are all equal to one.

- ***P*** is a square m x p matrix (square since initially p=m), which represents the standardised loadings / coefficients.
- ***L*** is the diagonal matrix where the values in the diagonal (eigenvalues) represent the amount of variance explained by each principal component. In this case, the variance is standardised as well.

Therefore, using ***C*** instead of ***S*** also changes the scores of the principal components (the new projected variables) from absolute to standardised units. To compare and select variables, which is a qualitative analysis, it is recommended to use the standardised units (***C*** instead of ***S***) when scales are different as already mentioned [21, 25]. Since the KPI scales are typically different, the present method should use the correlation matrix (***C***) to extract the principal components. However, once the selection is made (**phase 2**), start with the regression analysis **(phase 3**) as the objective is usually to interpret the coefficients in absolute terms – rather than just the statistical significance (p-value vs. α) and the predictive power (predictive R^2). The study must be done with the original variables and so their original units must be used (original scales). The present method uses regression analysis in this sense, and therefore the original scales of the variables are used. However, other methods use, for instance, multivariate regression analysis as PLS for qualitative analysis. In these cases, the dichotomy of standardised versus non-standardised is present, and researchers have to decide on the objectives of the study and the nature of the variables. Marin-Garcia & Alfalla-Luque [20] make an in-depth analysis on this topic and propose a series of recommendations for researchers using the PLS analysis.

Since a two-dimensional vector chart can only represent two dimensions, the method uses the two first principal components, $\boldsymbol{u_1}$ and $\boldsymbol{u_2}$. A verification of the variability explained by these two components is needed to ensure that the variance is at least 80% of the total [21]. For practical reasons, if the variation is not 80%, but is close, it is advisable to use the first two components. As part of this method, when more than two components are needed, factor analysis (FA) can be used instead of PCA [36]. First, according to Joliffe & Morgan [36], it is necessary to select the number of components (explaining at least 80% of the total variance) and rotate the vectors, usually using the 'varimax' rotation method, which facilitates the interpretation of the results. However, wherever possible, bi-dimensional vector visualisation is recommended, since a graphical method is always more intuitive, mainly, considering that the results are interpreted not only by the researchers, but also by company staff. The use of the 'varimax' rotation, which maximises the variance explained by the new projected variables (called factors instead of components in FA), is equivalent to using the

direction of the vectors when using the two-dimensional plot. These new coefficients are maximised when they are rotated and so the effect of having the original variables explained by several components or factors is solved, or at least minimised [36].

From the two-dimensional plot, the variables are selected according to weight criteria and correlation (vectors in the same direction, regardless of the sense) and considering which hypotheses are related to the aspects established in phase 0 – predictability and feedback of the QMS.

Once the variables are selected, a regression analysis is performed in **phase 3**. Following the principle of parsimony, the simplest regression technique is selected to test the hypotheses. The hypotheses related to the predictability of quality will always be a cause and effect relationship between the internal and external variables in the direction from inside the company towards the customers (outwards). The quality feedback hypotheses go in the other direction (inwards).

In this phase, the principle of parsimony is not the only aspect to select the simplest technique. Simple linear regression (SLR) models can be represented graphically; however, when there is more than one predictor in the model, the graphical representation is not clear or is not possible.

The practical application of the principle of parsimony is to select the simplest possible model, i.e. with as few variables as possible. The application of this principle will ensure that the selected model is the easiest to interpret, which is essential for the objectives of the methodology. On the other hand, a good quality of the model in terms of a high R^2 must be ensured. Therefore, if two regression models are comparable in terms of predictability (R^2), the simplest will be selected.

Akoglu [37] provides guidance for deciding the strength of the relationship between variables based on the correlation coefficient (ρ). Since it is well known that in simple linear regression $\rho^2 = R^2$, for each value of ρ we can compute an equivalent for R^2. Although this relationship can only be proven mathematically for SLR, the same interpretation of R^2, at least in terms of strength (quality), can be used for any regression model. Table 3 summarizes the criteria to decide between different models.

Strength	**Correlation**	**Regression**
Very strong	$0.8 < \lvert\rho\rvert \leq 1$	$64\% < R^2 \leq 100\%$
Strong	$0.7 < \lvert\rho\rvert \leq 0.8$	$49\% < R^2 \leq 64\%$
Moderate	$0.5 < \lvert\rho\rvert \leq 0.7$	$25\% < R^2 \leq 49\%$
Weak	$0.3 < \lvert\rho\rvert \leq 0.5$	$9\% < R^2 \leq 25\%$
Negligible	$0 \leq \lvert\rho\rvert \leq 0.3$	$0\% \leq R^2 \leq 9\%$

Table 3. Interpretation of ρ and R^2

In **phase 4**, the hypotheses proven/disproven by the regression models are confirmed by graphically comparing the behaviour of the time series of the variables included in

the regression models. If there is correlation, the regression model is significant (p-value < α) and the predictability power of the regression model is at least moderate according to table 3. It can then be said that there is a good model. If there is a good model, the behaviour of the variables and, therefore, of the time series should be similar. For each significant regression model whose strength is moderate to very strong, we will confirm that the behaviours of KPIs are similar by comparing the trends of the time series charts of each KPI included in the model – see figures 9, 16, and 17. This will help make the decision that the strength of the relationship is not only mathematical, but practical. Like any graphical analysis, it is essentially qualitative, since the confirmation of the quality (strength) of the regression model will depend on the nature of each KPI and its practical meaning. Therefore, the management team will conduct the analysis with the support of data analysts.

To graphically compare KPIs that have different scales, the size of the chart bars should be the same regardless of the range shown by the data, so KPIs can be compared in terms of trends regardless of the scale of the data. In practice, this can be done using automatic chart scaling that most computer packages with graphical tools incorporate.

In **phase 5**, researchers together with subject matter experts (SME) from the company discuss the results in detail. The main objective is to develop a specific statement for each significant regression model confirmed in the previous phase. This declaration should include an interpretation of the regression coefficient and the strength of the model based on Table 3.

For example, if we have the following regression model with p-value < 0.05 and R^2-pred = 76.86%:

R1000 0MIS=15.52-0.1966 *FTT*, the team will present the following statement:

'It has been found that there is a very strong relationship between the internal KPI of first time through (*FTT*) and the external KPI for warranty repairs at zero months in service (*R1000 0MIS*). A 1% increase in FTT causes a decrease of approximately 0.2 warranty repairs per 1000 units sold.'

Finally, in **phase 6**, these discussions are summarised in solid and practical conclusions with the aim of proposing strategic changes to improve customer satisfaction – which is the ultimate goal of the QMS.

The team must draw at least two main types of conclusions, one based on the confirmed regression models between internal and external KPIs, and one based on KPIs of the same stage, either external or internal – see figure 2 for process stages. The latter will be based on strong or very strong relationships based on the correlation

coefficient. For example, if a strong relationship is confirmed between two external metrics, as they belong to the same stage (the customer's), it is not a causal relationship, but a correlation. The same could happen for internal metrics of the same stage.

Only those KPIs that appear in the confirmed regression models will be considered as strategic, therefore the management team should exclude the rest. In addition, the team will reduce the complexity of the QMS by choosing only one KPI for each strong or very strong correlation between metrics of the same stage. These decisions will lead to a simplified QMS composed with KPIs with a strong impact on customer satisfaction.

4. Results of the case study and discussion

The aspects that were selected in **phase 0** of the study, which were the predictability of the quality management system and its feedback capability, have been explained in the previous section. In this phase, it was also decided to separate the study into two sub-studies, one with variables that include all the models produced in the company and the other, by the model.

In the hybrid analysis (graphical and analytical) of the time series [15], corresponding to **phase 1**, the conclusion was that they were stationary series and, therefore, the transformation of the data was not necessary.

Figure 2 shows the process flow of the case study and locates each group of KPIs (internal and external). The process flow is necessary to establish input and output variables for the regression analyses of **phase 3**, which is carried out on the KPIs previously selected in **phase 2** (see section 3).

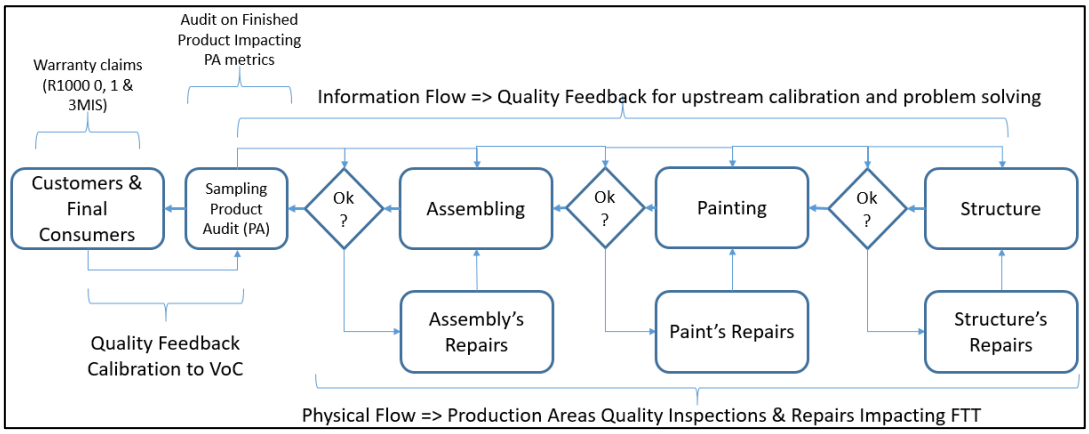


Figure 2. High-level process flow of the quality management system

Phases from **2** to **6** are detailed in the following sections.

4.1. Results including all models

4.1.1. Quality predictability

The predictability of quality is the relationship between the internal metrics and the voice of the customer as measured by warranty repairs at 0 months in service (*R1000 0MIS*), *R1000 1MIS* and *R1000 3MIS*.

Figures 3 and 5 are the bi-dimensional plots of the principal component analysis (PCA). Figures 4 and 6 show the amount of variance explained by each principal component (eigenvalues). In both study periods, bi-dimensional plots could explain about the 80% of the total variance observed [25]. By comparing the period from August 2017 to January 2018 (Fig. 3) to the period from January 2017 to January 2018 (Fig. 5), it can be seen that the relationship between the variables 'online product auditing' (*PA ONLINE*) and 'repairs per thousand at 0 months in service' (*R1000 0MIS*) is not maintained. The more orthogonal the vectors are, the less correlation there is between the variables. It is also denoted by the fact that the predictive R^2 (R^2-pred) was low (<30%) in the period beginning in August 2017. Therefore, when more data points are taken, that relationship disappears because the model is unstable.

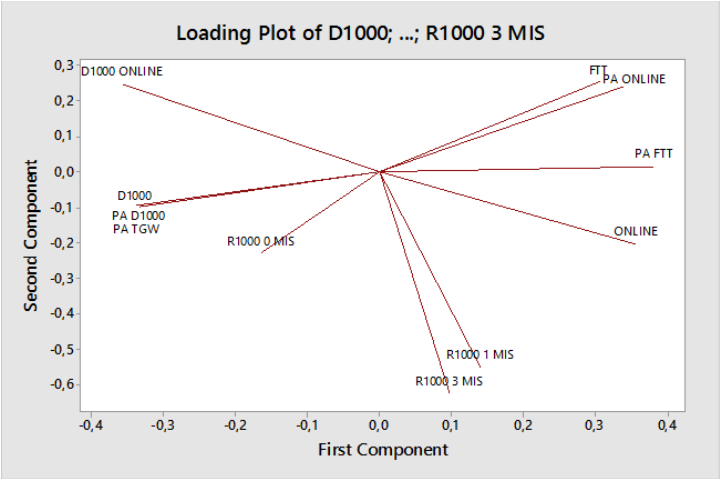


Figure 3. *PCA for all models (data from Aug 2017 to Jan 2018)*

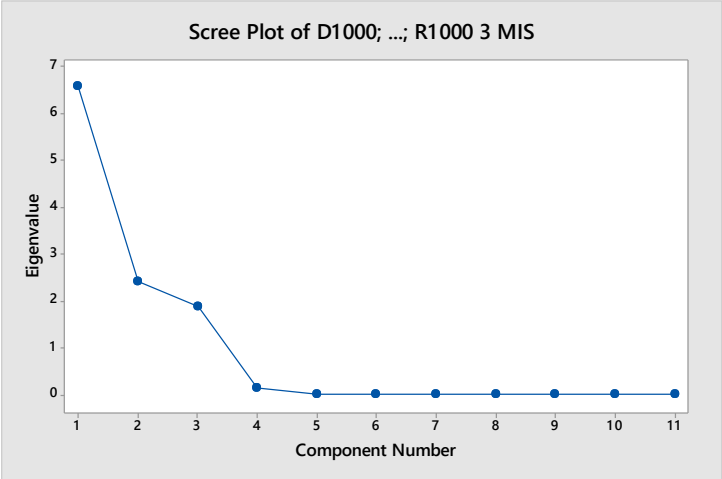


***Figure 4**. Scree plot (Aug 2017 to Jan 2018). 82% of variance in the two first components*

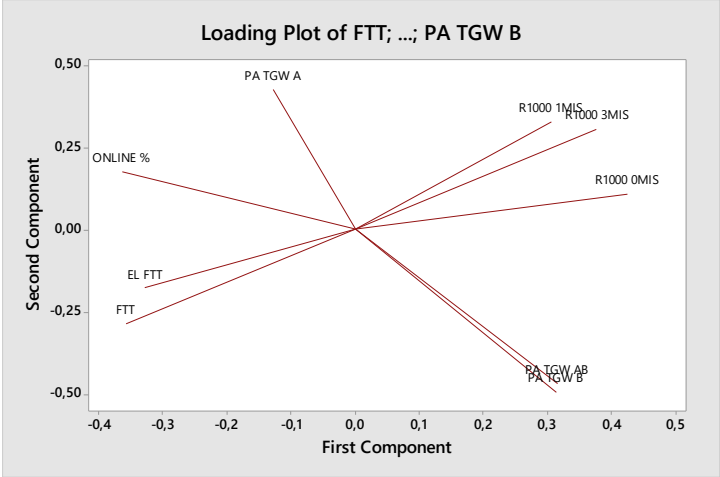


***Figure 5**. PCA for all models from Jan 2017 to Jan 2018*

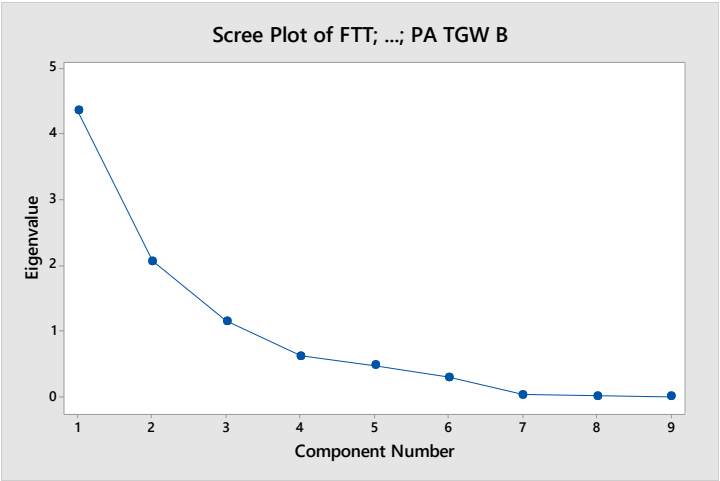


***Figure 6**. Scree plot (Jan'17-Jan'18). 72% of variance in the two first components*

The most powerful relationship that appears is that of warranties with almost all internal metrics – first time through (*FTT*), *end-of-line FTT* (*EL FTT*) and even with

on-line metrics, but especially with *FTT*, with a predictive R^2 for the period from August 2017 to January 2018 of 89.3%. For the period beginning in January 2017, R^2-pred was 75%. These values of R^2-pred mean a high predictive power and a high stability of the model.

A good quality of the model implies a good calibration of the internal quality controls with the voice of the customer (VoC). Therefore, the variability in the R^2 could mean differences in the level of calibration within different periods. These changes in the calibration of the internal controls require a recalibration of the quality controls, which is a key function of the quality improvement teams. Another highlight of this result is the potential use of the R^2 of this regression model to evaluate the level of calibration of internal controls in a given period. However, the limitation of sample size will always be present in this type of study, although the possibility of having more data points should also be explored, for example, by increasing the frequency of data points.

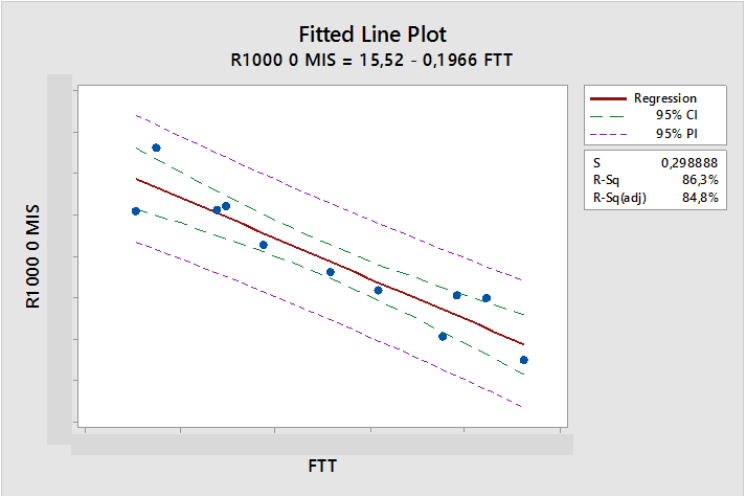


***Figure 7**. All models from Aug'17 to April'18 (R^2-pred=76.86%)*

The regression equation (also shown in Fig. 7) for this model is:

$$R1000\ 0MIS = 15.52 - 0.1966\ FTT \quad (7)$$

In Table 4, a complete analysis of variance and a model summary of the regression analysis of the Figure 7 is presented.

Analysis of variance							
Source	**DF**	**Seq SS**	**Contribution**	**Adj SS**	**Adj MS**	**F-Value**	**P-Value**
Regression	1	5.0753	86.32%	5.0753	5.0753	56.81	0.000
FTT	1	5.0753	86.32%	5.0753	5.0753	56.81	0.000
Error	9	0.8040	13.68%	0.8040	0.0893		
Total	10	5.8793	100.00%				
Model summary							
S	**R^2**	**R^2(adj)**	**PRESS**	**R^2**(pred)			
0.2988	86.32%	84.81%		1.3603	76.86%		
Coefficients							
term	**Coef**	**SE Coef**	**95% CI**	**T-Value**	**P-Value**	**VIF**	
Constant	15.52	1.60	(11.89; 19.14)	9.67	0.000		
FTT	-0.197	0.026	(-0.26; -0.14)	-7.54	0.000	1.00	

Table 4. Analysis of variance and model summary for the period Aug 2017 to April 2018

The coefficient of *FTT* means that an increase of one percentage point in the *FTT* equals a decrease of approx. 0.2 *R/1000 0MIS* and vice versa. However, the extrapolation of the linear function beyond the inference space should be used with caution even with such a high model quality, which would imply assuming that the linearity of the model remains beyond the inference space.

The model shows that there is no need to reach 100% of the *FTT* to eliminate warranty claims at *0MIS (R1000 0MIS)*. Although it is not entirely possible, since the probability model based on continuous distributions and product specifications is asymptotic, the linear approximation is good and thinking of a defect reduction very close to zero in the customer before 100% of *FTT* is not completely illogical. This objective, in relation to the transfer function of the regression model, was established at a certain *FTT* point (not shown due to confidentiality reasons) for this case study. The assumptions of normality, equal variance, and independence of the residuals have been verified to validate the model. The autocorrelation for the independent variables has also been verified by up to 12 lags to rule out the overestimation of the regression coefficient due to the time relationships (lack of independence of the estimators). The assumptions were verified for *FTT* and 'defects per thousand' KPIs (*D1000*) – see Figure 8.

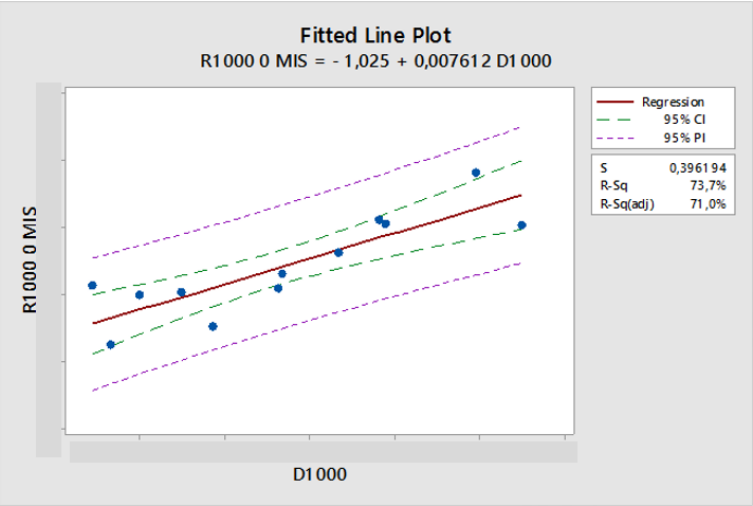


Figure 8. *Regression R1000 0MIS vs. D1000 (Aug'17 – Apr'18) (R^2-pred=57%)*

In Table 5, a complete analysis of variance and a model summary of the regression analysis of the Figure 8 is presented.

Analysis of variance							
Source	**DF**	**Seq SS**	**Contribution**	**Adj SS**	**Adj MS**	**F-Value**	**P-Value**
Regression	1	4.393	73.68%	4.393	4.3933	27.99	0.000
D1000	1	4.393	73.68%	4.393	4.3933	27.99	0.000
Error	10	1.570	26.32%	1.570	0.1570		
Total	11	5.963	100.00%				
Model summary							
S	**R^2**	**R^2(adj)**	**PRESS**	**R^2-pred**			
0.3962	73.68%	71.04%	2.5812	56.71%			
Coefficients							
Term	**Coef**	**SE Coef**	**95% CI**	**T-Value**	**P-Value**	**VIF**	
Constant	-1.025	0.848	(-2.915;0.864)	-1.21	0.254		
D1000	0.0076	0.0014	(0.0044;0.011)	5.29	0.000	1.00	

Table 5. Analysis of variance and model summary for the period Aug 2017 to April 2018

A likely interpretation of this result is that all failure modes at 0 MIS (impact on customer's warranty claims) are the same as those detected within the production facilities during internal verifications (those related to the KPIs of *FTT*, *EL* and *ONLINE %)*. Another possible reason is that the relationship between *R1000 0MIS* and *D1000* remains stable regardless of the chosen study period, which was also confirmed by a regression model. There was a slight fluctuation in the value of the regression coefficient that turned out to be between 0.008 and 0.01. It means that the quality leak can be estimated around that proportion, which is the Type-II error. An improvement strategy may be to reinforce internal quality controls based on objective measures using Gage R & R for both variables and attributes. However, a Type II error of less than 1%

is more than 10 times better (smaller) than the industry average, which is approximately 10%. Negative values of *R1000 0MIS* are not possible, but the negative coefficient of the equation implies that before *D1000* reaches zero we will have zero *R1000 0MIS*, which is the same conclusion as for the equation with *FTT*, due to the linear assumption.

Another point to consider is the relationship between *R1000 1MIS* and *R1000 3MIS*, which also remains constant with an R^2-pred of 80%. This means that both are, in fact, the same indicator, at least in their dynamic behaviour. Both indicators could be summarised – as one or one of them can be eliminated to reduce the complexity of the balanced scorecard.

In the following lines and figures (see Figure 9), as part of **phase 4**, it is graphically confirmed that when there is a good regression model or a high correlation, the dynamic behaviour of the variables on both sides of the equal sign of the equation is very similar, since this method uses time series as variables.

In Figure 9, where the warranties at 0MIS (*R1000 0MIS*) are compared with the complementary of the *FTT*, we can see the correlation between both KPIs in a more intuitive way.

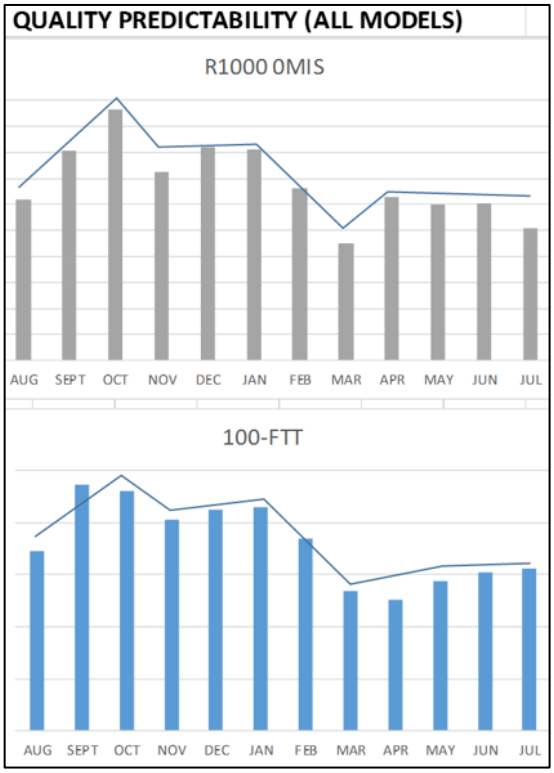


***Figure 9**. Graphical confirmation of the predictability of the quality system*

4.1.2. Quality feedback

While the quality predictability can be understood as the ability to predict customer warranties based on internal metrics, the quality feedback is the ability of the system to

feed customer claims back to production facilities in the form of quality controls during the audits of finished products (*PA*). These audits, since they are based on small samples, are designed to calibrate the upstream system, but not to predict the behaviour of the market.

To carry out this study, it was necessary to transform some variables, applying a certain time delay. The time series related to customer complaints were transformed with different delays of t-1, t-2 and t-3, which means delays of 1, 2, and 3 months. This transformation allowed the study of the hypothetical delayed correlation between the customer claims and the product audit KPIs (*PA*). Delays of more than three months were also tested in the study although they are not shown here for reasons of clarity. However, the results showed that there were no relationships between the variables with such delays.

In Figures 10 and 11, we can see a clear relationship between R1000 *0MIS t-3* and type B alerts of *PA* (*PA B*) with 70% of R^2-pred, slightly weaker than with *R1000 1MIS t-3* and *R1000 3MIS t-3*, which have an R^2-pred of 50%. With the time series with a delay of less than three months, which is t-1 and t-2, there was no significant relationship; as shown by the analysed data.

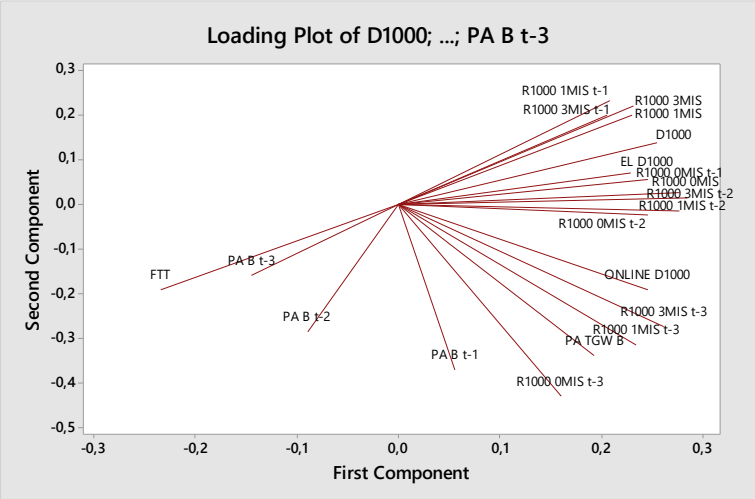


***Figure 10**. Quality feedback for all models. From Jan 2017 with lagged variables*

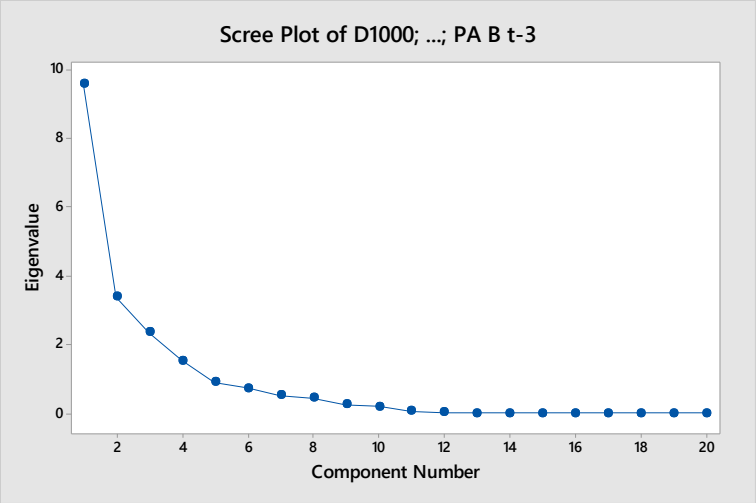


***Figure 11**. Scree Plot from Jan 2017. 70% of variance in the two first components*

Figures 12 to 14 show the relationship between customer claims and *PA* in terms of quality feedback.

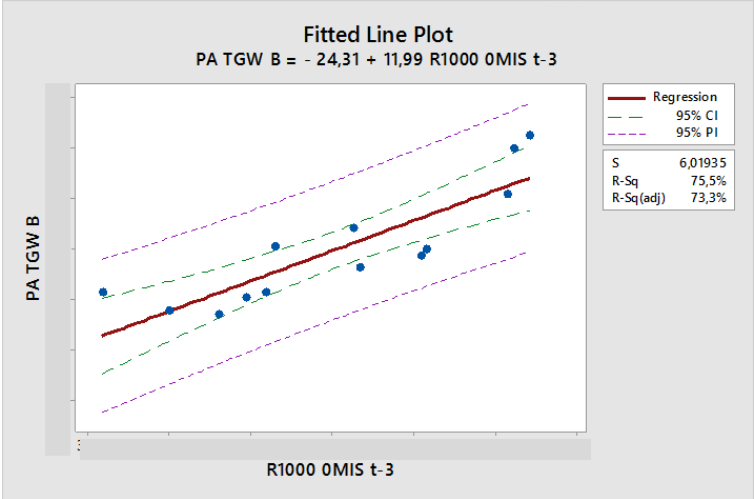


***Figure 12**. PA TGW B vs. customer claims at 0MIS after 3 months. R^2-pred = 62%*

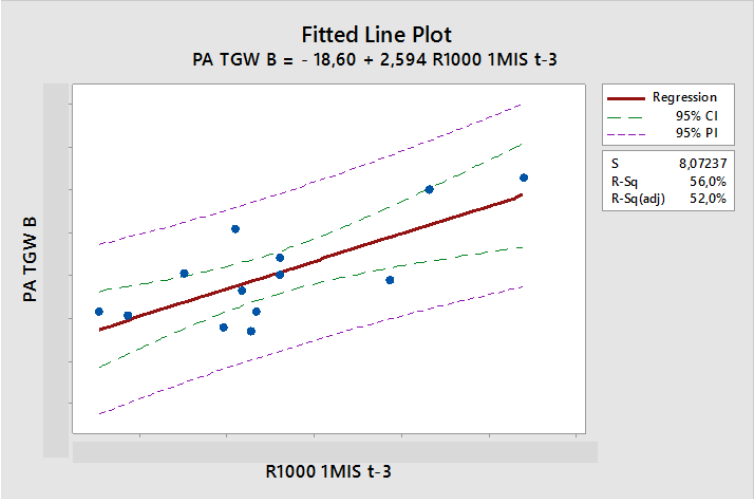


***Figure 13**. PA TGW B vs. customer claims at 1MIS after 3 months. R^2-pred = 41%*

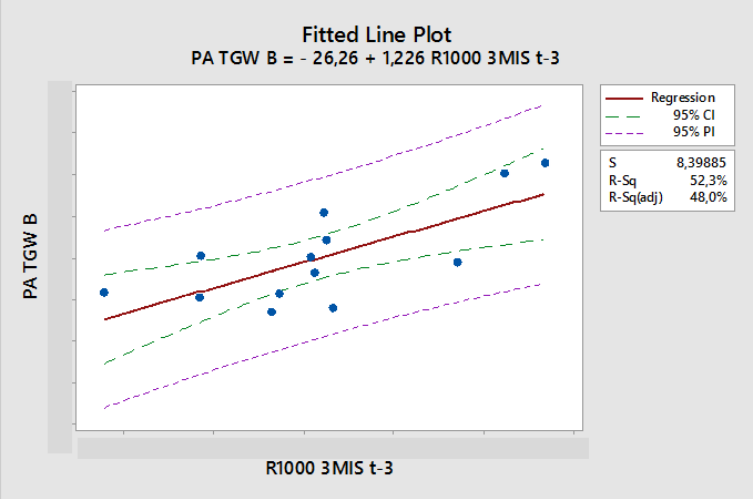


***Figure 14**. PA TGW B vs. customer claims at 3MIS after 3 months. R^2-pred=32%*

The main interpretation of these results is that it takes around three months to provide feedback to the product audits. In addition, failure modes claimed by customers at 1MIS and 3MIS do not feed back with the same efficiency to product audits as those at 0MIS. This could be because these failure modes are not based on verifications in the production plant, but in special actions to increase the robustness of the product or in verifications related to reliability. In addition, these failure modes are sometimes latent or functional problems that cannot be detected in regular internal inspections, but only in product audits.

Negative values of *PA TGW B* are not possible, but the negative coefficient tells us that before *R1000 0MIS* reaches zero, *PA* must be zero. This means that product audits do not capture all failure modes. Only after a certain value of *R1000 0MIS* do product audits detect those failure modes three months later.

Before adjusting the simple regression models, a multiple linear regression (MLR) model was tested that included all the variables in the three different MIS (R1000 *0MIS*, *R1000 1MIS* and *R1000 3MIS*) and the quadratic terms. This model was ruled out due to a much lower R^2-pred than the SLR models. In addition, the variance assumptions and the independence of the residuals were verified to validate the regression model.

The model *PA TGW B* = -24.31 + 11.99 *R1000 0MIS t-3* was chosen as the only model valid from a systemic and structural point of view. The reasons were the following:

- When applying MLR and reducing the model using the stepwise algorithm, only the *R1000 0MIS* term remains in the model. Such a result was replicated for the model with and without constant – as well as when using standardised variables and absolute scales. Therefore, the conclusion was always the same – only *R1000 0MIS* remained in the model.

- The coefficient of *R1000 0MIS* is greater than the others, which also means greater sensitivity and power of explanation. The same occurred when using standardised variables.
- It makes physical sense that the 0MIS warranty claims explain most of the *PA* defects.
- The direct correlation between the *PA* indicators and *1MIS* & *3MIS* is lost according to the study period, which is also supported by the evidence shown in Figures 2, 4 and 16. In Figure 16, we can see that there is no clear correlation between *PA* and the warranties, but the correlation between *1MIS* and *3MIS* is never lost regardless of the study period (see also Figures 2 and 4).

However, the fact that, although only in some specific periods, *PA* KPIs may have some relation with *R1000 1MIS* and *R1000 3MIS* could be interesting and may be the objective for a future study on this topic.

Figure 15 summarises the three models in one picture.

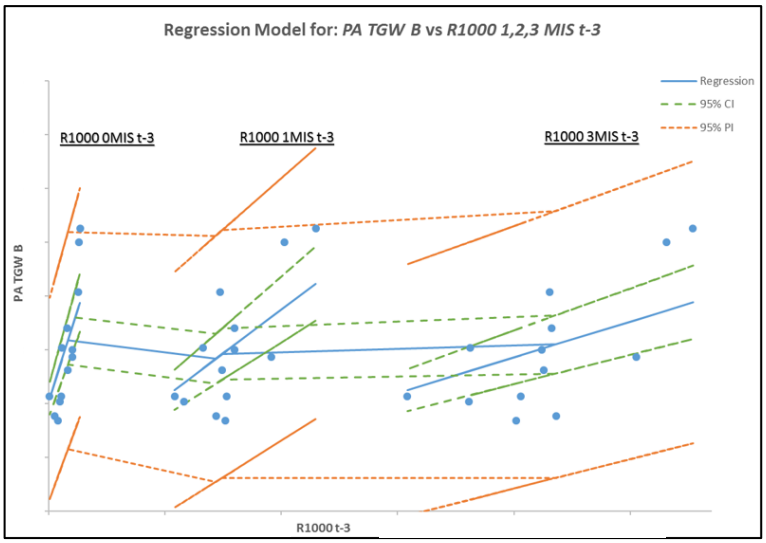


Figure 15. *Regression models for PA TGW B vs. R1000 at 1, 2 and 3MIS t-3*

Figure 16 shows the graphic confirmation of the correlation between *PA TGW B* and *R/1000 0 MIS*.

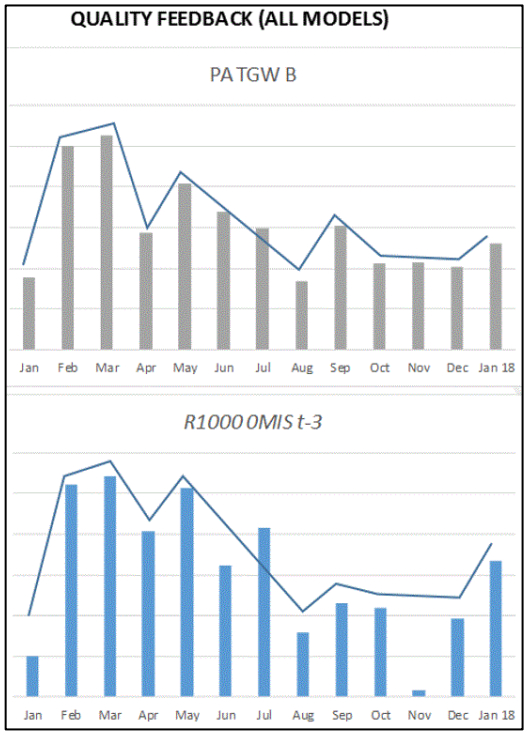


***Figure 16**. Correlation between PA TGW B & R1000 0MIS with 3-month delay (t-3)*

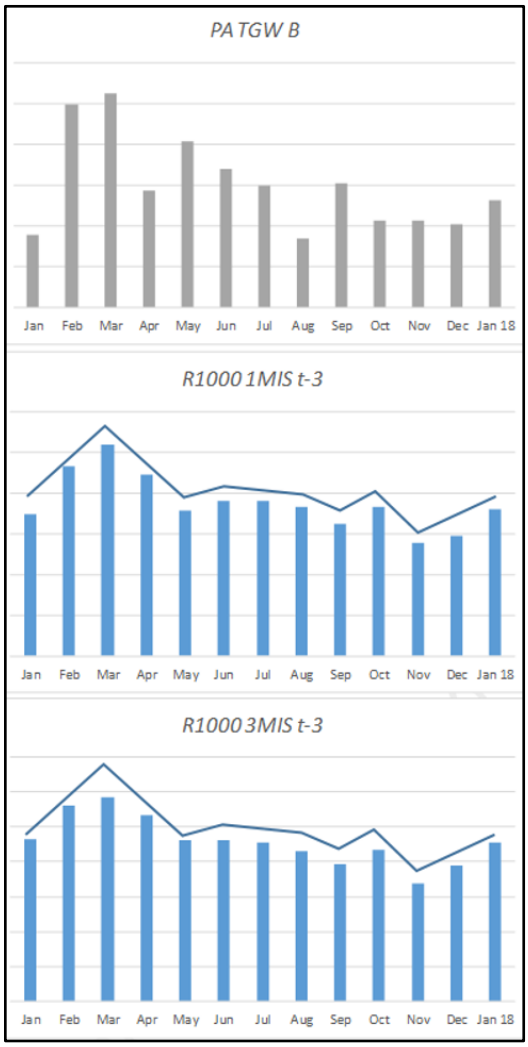


***Figure 17**. Correlation among PA TGW B, R1000 1MIS and R1000 3MIS with 3-month delay (t-3)*

Figure 17 clearly shows the absence of correlation between *PA* indicators and *R1000 1MIS* & *R1000 3MIS*. In addition, the correlation between *R1000 1MIS* and *R1000 3MIS* is again evident and has been confirmed in each study period, which means that it is a solid structural relationship.

To validate the models, the assumptions of independence and equality of variance of the residuals were verified. In addition, the presence of autocorrelation of up to 12 delays in the predictors was ruled out.

It is interesting to quantify in a time period the ability to capture the modes of failure of warranty claims. The time period has been estimated as approximately three months and the ability to capture faults per *PA* could be estimated at a rate of 12 for *R1000 0MIS*, 2.6 for *R1000 1MIS,* and 1.23 for *R1000 3MIS*, which are the coefficients of the regression models shown in Figures 11 to 13. The higher the MIS, the lower the detection capacity in PA. Such a conclusion derived from the models is logical, since the higher MIS failure modes are more difficult to detect within the inspections of the production plant.

4.2. Results by model

4.2.1. Quality predictability

Analysis by model gives similar results, although less consistent in terms of stability and the power of relationships between variables. This first unexpected result is probably because the uncertainty due to working with proportions of internal and external metrics is much greater than that of continuous variables. This uncertainty increases as the proportion or size of the sample decreases, so for models with small proportions (defect rate) and/or small production volumes (sample size), the uncertainty of the data increases. Therefore, more data points may be necessary to establish relationships based on regression / correlation techniques.

The above-mentioned characteristic, confirmed by the results, has meant that conclusions of the aspect of quality predictability were only obtained when the relationships between the variables were significant enough. Therefore, it was not possible to obtain any meaningful model for the aspect of quality feedback when the KPIs were split by model.

Figure 18 shows the regression model for the production model A. We can see a similar relationship between *R1000 0MIS* and *EL D1000*. Although there are more

relationships between internal and external metrics, the relationship shown is the strongest, regardless of the study period. The regression coefficient is around 0.0206.

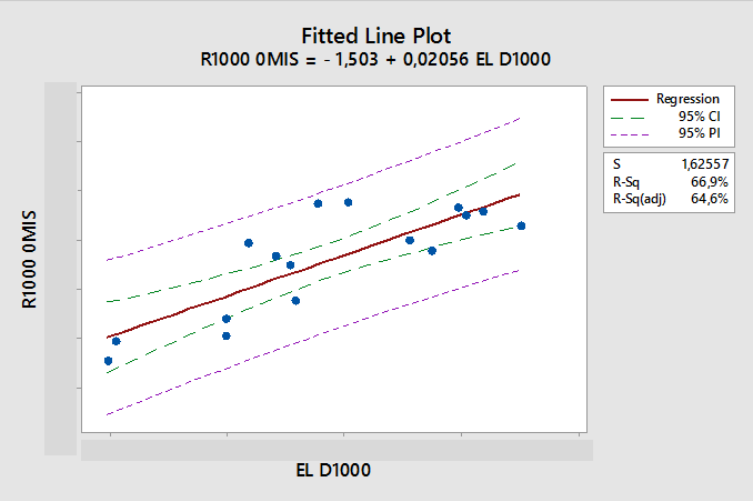


***Figure 18**. Data for the period from Jan'17 to Apr'18. R^2-pred ≈ 60%*

For production model B, it was not possible to confirm such relationships between internal and external metrics. Figure 19 shows some new metrics between different MIS, which, interestingly, were different from what was seen when working with all the models. *0MIS* warranties (*R1000 0MIS*) had a moderate to strong correlation with *1MIS* and *3MIS* (*R1000 0MIS* & *R1000 3MIS*), with a Pearson correlation coefficient of 0.8 ($R^2 \approx 64\%$) for the case of *1MIS* and 0.7 ($R^2 \approx 50\%$) for *3MIS*. A more detailed analysis of the failure mode could establish physical reasons for these relationships if it is confirmed that some related failure modes are appearing in different MIS (at least in this production model).

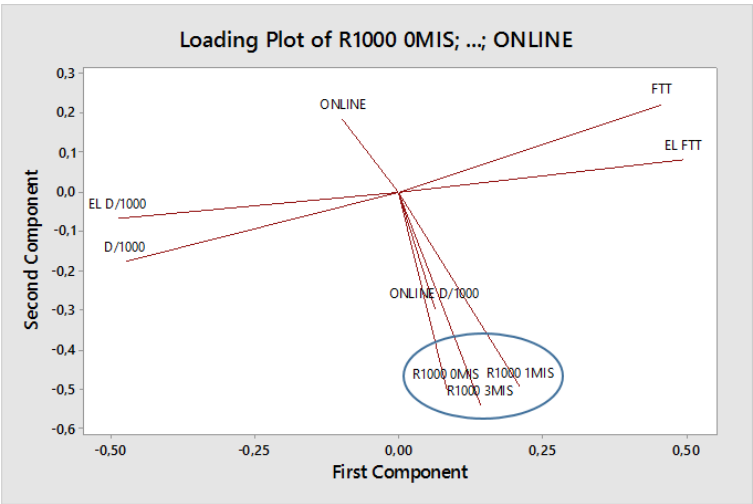


***Figure 19**. Production model B bi-plot of PCA for the period starting in Aug 2017*

Figure 20 illustrates the results for the production model C. A similar relationship was found between *R1000 0MIS* and *D1000*, although its coefficient was only 0.7% and its R^2-pred was slightly greater than 30%. Therefore, it seemed to confirm the relationship between internal and external metrics with a moderate quality of the regression model.

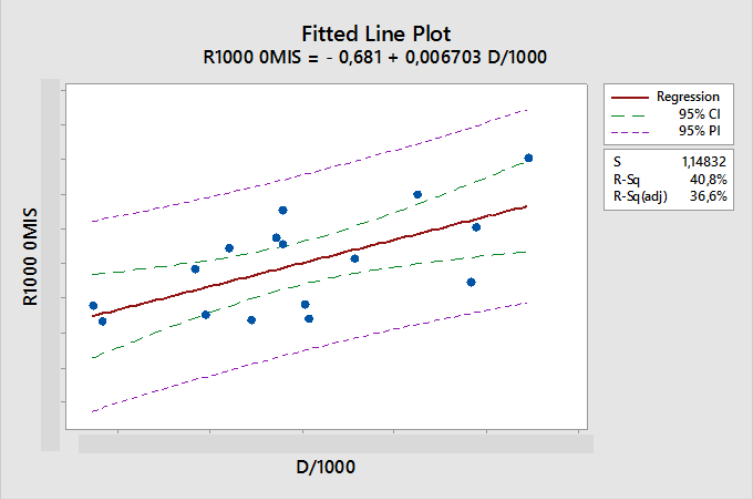


***Figure 20**. Regression of R1000 0MIS vs D1000 – R^2-pred ≈ 30%*

Figure 21 shows the results for the production model D. Two relationships between the metrics were found, although the most interesting is that this it is the only model that establishes a correlation of *R1000 3MIS* and an internal metric. It was the *ONLINE* metric expressed as a percentage. This relationship had a Pearson correlation coefficient of 0.636 (R^2-pred of 20%), which can be considered moderate to weak, but with a p-value of 0.019 – although its stability would not be particularly good, and it would have a high-risk level if used to make predictions. Despite this, there were additional correlations that, although weak, were present in other metrics: such as *EL* with a Pearson coefficient of -0.539 and a p-value of 0.057. Based on these findings, we could say that production model D would be the only model where it is possible to detect some failure modes that appeared after three months in service (*3MIS*).

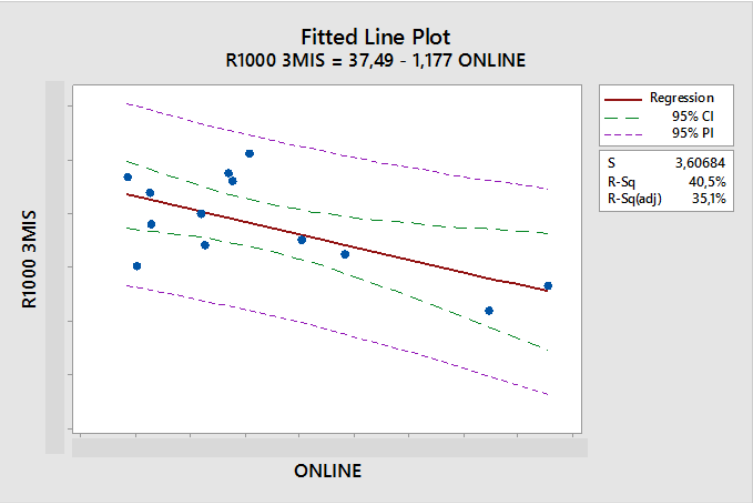


***Figure 21**. Regression of R1000 3MIS vs ONLINE – R^2-pred ≈ 21%*

5. Conclusions

Based on the results, the main conclusions are summarised in the following lines. Two different sections are presented for the aspects of quality predictability and quality feedback.

The executive board of the company followed most of the recommendations made in **phase 6** of this study, which are included in this section. For example, the *FTT* was included in the balanced scorecard for all production facilities around the world and strategies were initiated to improve the *FTT*. The improvement actions derived from these strategies caused the customer quality complaint metrics to improve within a few months. Due to this, the *FTT* was considered as a strategic KPI. In addition, the balanced scorecard was simplified by eliminating the KPIs of *R1000 3MIS* and the quality improvement teams began to only monitor *R1000 1MIS* and this implied a faster reaction time that also meant improvements in the quality KPIs related to customer satisfaction.

5.1. Conclusions on quality predictability

Conclusions on the aspect of the predictability of the QMS can be summarised as follows:

- The stable (structural) and powerful relationship between *FTT* and *R1000 0MIS* was confirmed regardless of the study period and even when using data from different model years.
- Such a strong correlation implies an excellent calibration between the internal quality controls and the VoC.

- Every 2% improvement in *FTT* equals approx. 0.4 *R1000 0MIS*. With *FTT* = 78.94% it is possible to reach the ideal zero *R1000* at *0MIS* (assuming the existence of a linear model).
- There was another strong and stable correlation between *R1000 1MIS* and *R1000 3MIS*. Since $\rho>0.9$, both indicators can be considered as different measures of almost the same thing. Therefore, it would make sense to use only one KPI for the balanced scorecard. The best option is to maintain *R1000 1MIS* and eliminate *R1000 3MIS*, since the KPIs of *R1000 1MIS* are obtained two months previously and the reaction to a deterioration of the metric would be faster.
- The general leakage of defects can be quantified as between 0.8% and 0.9%, which is much better than what is considered a good leak, namely 10% for a Type II error (β Risk).
- This study proved that statistical analyses of KPIs can be used to diagnose the predictability of quality systems in a manufacturing environment.
- Since this method uses statistical tools with real data, it has the limitation of needing a sufficiently sized sample. Future research may focus on changing the data period (measure more frequently) to overcome or minimise this limitation.
- Future research can focus on the generalisation of the method by applying it to the other six management systems.

5.2. Conclusions on quality feedback

Conclusions about the feedback ability of the quality system can be summarised as follows:

- It took three months to provide feedback to the product audits (60 days for data maturity plus 30 additional days for the feedback process itself).
- The strength of the relationships and their stability weakened as we increased MIS. Only the relationship between *PA* and *R1000 0MIS* remained independent of the study period. Therefore, the capacity and stability to capture warranties in product audits was reduced as MIS increased
- Product audits were working as a calibrator of the internal quality system but not as a predictor.
- It was recommended that *R1000 1MIS* appear in the balanced scorecard instead of *R1000 3MIS*. The reaction would be two months faster as *R1000 1MIS* and *R1000 3MIS* were strongly correlated.
- This study proved that the statistical analysis of KPIs can be used to diagnose how the quality management system works in terms of feedback.
- Future research may focus on the generalisation of the method by applying it to other sectors beyond the manufacturing environment.
- Since this method uses statistical tools with real data, it has the limitation of needing enough sample. Future research may focus on changing the data period (measuring more frequently) to overcome or minimise this limitation.

6. Abbreviations

- ACF: autocorrelation function
- ANN: artificial neural network
- ANP: analytical network process
- BSC: balanced scorecard
- CB-SEM: covariance-based structural equation modelling
- EL: end of line
- FA: factor analysis
- KPI: key performance indicator
- MIS: months in service
- MLR: multiple linear regression
- PA: product audit
- PACF: partial autocorrelation function
- PCA: principal component analysis
- PLS: partial least squares
- PLS-SEM: partial least squares structural equation modelling
- PMS: performance management system
- QMS: quality management system
- SLR: simple linear regression
- SME: subject matter expert
- VIF: variance inflation factor
- VoC: voice of customer

7. References

[1] Amaratunga, D., & Baldry, D. (2002). Moving from performance measurement to performance management. Facilities, 20(5/6), 217-223.
[2] Bititci, U. S., Turner, U., & Begemann, C. (2000). Dynamics of performance measurement systems. International Journal of Operations & Production Management, 20(6), 692-704.
[3] Robert S. Kaplan and David P. Norton (2001). Transforming the Balanced Scorecard from Performance Measurement to Strategic Management: Part I. Accounting Horizons: March 2001, Vol. 15, No. 1, pp. 87-104.
[4] Dennis P (2006). Getting the right things done: A learner's guide to planning and execution. The Lean Enterprise Institute, Cambridge, MA, USA.
[5] Kaplan, R. S. (2009). Conceptual foundations of the balanced scorecard. Handbooks of management accounting research, 3, 1253-1269.
[6] Hoque, Z. (2014). 20 years of studies on the balanced scorecard: trends, accomplishments, gaps and opportunities for future research. The British accounting review, 46(1), 33-59.
[7] Malbašić, I. & Marimon, F. (2019). A Simplified Balanced 'Balanced Scorecard'. European Accounting and Management Review, 5(2), 38-60.
[8] Malmi, T. (2001). Balanced scorecards in Finnish companies: a research note. Management Accounting Research, 12(2), 207-220.
[9] Anand, M., Sahay, B. S., & Saha, S. (2005). Balanced scorecard in Indian companies. Vikalpa, 30(2), 11-26.

[10] Junior, I. C. A., Marqui, A. C., & Martins, R. A. (2008). MULTIPLE CASE STUDY ON BALANCED SCORECARD IMPLEMENTATION IN SUGARCANE COMPANIES. Accessed 26 Dec 2016. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.572.3364&rep=rep1&type=pdf
[11] Ferenc A (2011). Balanced Scorecard Measurement applications at a car manufacturer supplier company. Accessed 8 May 2017. https://pdfs.semanticscholar.org/f10e/409533c49dd2934ace78405126978302ab96.pdf.
[12] Grillo-Espinoza, H., Campuzano Bolarin, F., & Mula, J. (2018). Modelling performance management measures through statistics and system dynamics-based simulation. Dirección y Organización, 65, 20-35.
[13] Rodriguez-Rodriguez R., Saiz, J. J. A., & Bas, A. O. (2009). Quantitative relationships between key performance indicators for supporting decision-making processes. Computers in Industry, 60(2), 104-113.
[14] Morard, B., Stancu, A., & Jeannette, C. (2013). Time evolution analysis and forecast of key performance indicators in a balanced scorecard. Global Journal of Business Research, 7(2), 9-27.
[15] Sanchez-Marquez, R., Guillem, J. M. A., Vicens-Salort, E., & Vivas, J. J. (2018b). Intellectual Capital and Balanced Scorecard: impact of Learning and Development Programs using Key Performance Indicators in Manufacturing Environment. Dirección y Organización, (66), 34-49.
[16] Sanchez-Marquez, R., Guillem, J. A., Vicens-Salort, E., & Vivas, J. J. (2018a). A statistical system management method to tackle data uncertainty when using key performance indicators of the balanced scorecard. Journal of Manufacturing Systems, 48, 166-179.
[17] Boj, J. J., Rodriguez-Rodriguez, R., & Alfaro-Saiz, J. J. (2014). An ANP-multi-criteria-based methodology to link intangible assets and organizational performance in a Balanced Scorecard context. Decision Support Systems, 68, 98-110.
[18] Gurrea V, Alfaro-Saiz JJ, Rodriguez-Rodriguez R, Verdecho MJ (2014). Application of fuzzy logic in performance management: a literature review. International Journal of Production Management and Engineering, 2(2), 93-100.
[19] Chytas, P., Glykas, M., & Valiris, G. (2011). A proactive balanced scorecard. International Journal of Information Management, 31(5), 460-468.
[20] Marin-Garcia, J., & Alfalla-Luque, R. (2019). Key issues on Partial Least Squares (PLS) in operations management research: A guide to submissions. Journal of Industrial Engineering and Management, 12(2), 219-240.
[21] Peña, D. (2002). Análisis de datos multivariantes. Retrieved July 5th, 2018, from: http://bida.uclv.edu.cu/bitstream/handle/123456789/12092/Daniel%20Pena%20-%20Analisis%20de%20datos%20multivariantes%20.pdf?sequence=1
[22] Coelho, M. T. P., Diniz-Filho, J. A., & Rangel, T. F. (2019). A parsimonious view of the parsimony principle in ecology and evolution. Ecography, 42(5), 968-976.
[23] Nalborczyk, L., Bürkner, P. C., & Williams, D. R. (2019). Pragmatism should not be a substitute for statistical literacy, a commentary on Albers, Kiers, and van Ravenzwaaij (2018). Collabra: Psychology, 5(1).
[24] Gunitsky, S. (2019). Rival Visions of Parsimony. International Studies Quarterly.
[25] Rencher, A. C. (2005). A review of "Methods of Multivariate Analysis". Retrieved June 30th, 2018, from: https://pdfs.semanticscholar.org/a83c/fec9c23390a10e5c215c375480b8cd3a1565.pdf
[26] He, Q. P., & Wang, J. (2018). Statistical process monitoring as a big data analytics tool for smart manufacturing. Journal of Process Control, 67, 35-43.
[27] Neely, A., Gregory, M., & Platts, K. (1995). Performance measurement system design: a literature review and research agenda. International journal of operations & production management, 15(4), 80-116.
[28] Akkerman, R., Farahani, P., & Grunow, M. (2010). Quality, safety and sustainability in food distribution: a review of quantitative operations management approaches and challenges. Or Spectrum, 32(4), 863-904.
[29] Goetsch, D. L., & Davis, S. B. (2014). Quality management for organizational excellence. Upper Saddle River, NJ: pearson.
[30] Molina-Azorín, J. F., Tarí, J. J., Claver-Cortés, E., & López-Gamero, M. D. (2009). Quality management, environmental management and firm performance: a review of empirical studies and issues of integration. International Journal of Management Reviews, 11(2), 197-222.
[31] Norreklit, H. (2000). The balance on the balanced scorecard a critical analysis of some of its assumptions. Management accounting research, 11(1), 65-88.
[32] Wu, J. P., & Wei, S. (1989). Time series analysis. Hunan Science and Technology Press, ChangSha. Retrieved January 20th, 2018, from: http://www2.geog.ucl.ac.uk/~mdisney/teaching/GEOGG121/time_series/GEOGG121_5_TimeSeries_Wu.pdf
[33] Box GE, Jenkins GC, Reinsel GC (2008). Time Series Analysis Forecasting and Control. New York: John Wiley and Sons.
[34] Dickey, D. A., & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. Journal of the American statistical association, 74(366a), 427-431.
[35] Becketti, S. (2013). Introduction to time series using Stata (pp. 176-182). College Station, TX: Stata Press.
[36] Joliffe, I. T. & Morgan, B. J. T. (1992). Principal component analysis and exploratory factor analysis. Statistical methods in medical research, 1(1), 69-95.
[37] Akoglu, H. (2018). User's guide to correlation coefficients. Turkish journal of emergency medicine, 18(3), 91-93.