In [15]:
```
# Librerias
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import plotly.express as px
import plotly.graph_objects as go

import seaborn as sns
from sklearn.preprocessing import OneHotEncoder
```

Datos

Los datos consta de los siguientes archivos:

- PdM_telemetry.csv: Las condiciones de funcionamiento de una máquina y datos recopilados de los sensores.
- PdM_errors.csv: El historial de errores que ha detectado la máquina, estos errors no siempre conllevan a un fallo.
- PdM_failures.csv: El historial de fallas de una máquina o componente dentro de la máquina.
- PdM_main.csvt: el historial de reparación de una máquina, p. códigos de error, actividades de mantenimiento anteriores o reemplazos de componentes.
- PdM_machines.csv: Las características de una máquina como el identificador, el modelo y la edad.

In [2]:
```
# Lectura de datos
DATA_DIR = "C:/Users/NetRunner/OneDrive/UOC/Semestre 6/TFM/Data"

telemetry_df = pd.read_csv(f"{DATA_DIR}/PdM_telemetry.csv")
errors_df = pd.read_csv(f"{DATA_DIR}/PdM_errors.csv")
maint_df = pd.read_csv(f"{DATA_DIR}/PdM_maint.csv")
failures_df = pd.read_csv(f"{DATA_DIR}/PdM_failures.csv")
machines_df = pd.read_csv(f"{DATA_DIR}/PdM_machines.csv")

tables = [telemetry_df, maint_df, failures_df, errors_df]
for df in tables:
    df["datetime"] = pd.to_datetime(df["datetime"], format="%Y-%m-%d %H:%M:%S")
    df.sort_values(["datetime", "machineID"], inplace=True, ignore_index=True)
```

In [4]:
```
# Funciones
def create_date_features(source_df, target_df, feature_name):
    '''
    Create new features related to dates

    source_df : DataFrame consisting of the timestamp related feature
    target_df : DataFrame where new features will be added
    feature_name : Name of the feature of date type which needs to be decomposed.
    '''
    target_df.loc[:, 'year'] = source_df.loc[:, feature_name].dt.year.astype('uint16')
    target_df.loc[:, 'month'] = source_df.loc[:, feature_name].dt.month.astype('uint8')
    target_df.loc[:, 'quarter'] = source_df.loc[:, feature_name].dt.quarter.astype('uint8')
    target_df.loc[:, 'weekofyear'] = source_df.loc[:, feature_name].dt.isocalendar().week.astype('uint8')

    target_df.loc[:, 'hour'] = source_df.loc[:, feature_name].dt.hour.astype('uint8')

    target_df.loc[:, 'day'] = source_df.loc[:, feature_name].dt.day.astype('uint8')
    target_df.loc[:, 'dayofweek'] = source_df.loc[:, feature_name].dt.dayofweek.astype('uint8')
    target_df.loc[:, 'dayofyear'] = source_df.loc[:, feature_name].dt.dayofyear.astype('uint8')
    target_df.loc[:, 'is_month_start'] = source_df.loc[:, feature_name].dt.is_month_start
    target_df.loc[:, 'is_month_end'] = source_df.loc[:, feature_name].dt.is_month_end
    target_df.loc[:, 'is_quarter_start']= source_df.loc[:, feature_name].dt.is_quarter_start
    target_df.loc[:, 'is_quarter_end'] = source_df.loc[:, feature_name].dt.is_quarter_end
    target_df.loc[:, 'is_year_start'] = source_df.loc[:, feature_name].dt.is_year_start
    target_df.loc[:, 'is_year_end'] = source_df.loc[:, feature_name].dt.is_year_end

    # This is of type object
    target_df.loc[:, 'month_year'] = source_df.loc[:, feature_name].dt.to_period('M')

    return target_df

def plot_boxh_groupby(df, feature_name, by):
    """
    Box plot with groupby

    df: DataFrame
    feature_name: Name of the feature to be plotted
    by: Name of the feature based on which groups are created
    """
    df.boxplot(column=feature_name, by=by, vert=False,
                               figsize=(15, 8))
    plt.title(f'Distribution of {feature_name} by {by}')
    plt.show()

def check_duplicate(df, subset):
```

```
    """
    Returns if there are any duplicate rows in the DataFrame.

    df: DataFrame under consideration
    subset: Optional List of feature names based on which
            duplicate rows are being identified.
    """
    if subset is not None:
        return df.duplicated(subset=subset, keep=False).sum()
    else:
        return df.duplicated(keep=False).sum()
```

EDA

¿Cuál es el rango de fechas de los datos que disponemos?

In [5]:
```
print(f"Primera fecha: {telemetry_df.datetime.min()} \n")
print(f"Última  fecha: {telemetry_df.datetime.max()} \n")
```

```
Primera fecha: 2015-01-01 06:00:00

Última  fecha: 2016-01-01 06:00:00
```

¿Qué tamaños tienen las tablas?

In [6]:
```
print(f"Medida de la tabla Telemetry:\t {telemetry_df.shape} \n")
print(f"Medida de la tabla Errors:\t {errors_df.shape} \n")
print(f"Medida de la tabla Maintenance:\t {maint_df.shape} \n")
print(f"Medida de la tabla Failures:\t {failures_df.shape} \n")
print(f"Medida de la tabla Machines:\t {machines_df.shape} \n")
```

```
Medida de la tabla Telemetry:    (876100, 6)

Medida de la tabla Errors:       (3919, 3)

Medida de la tabla Maintenance:  (3286, 3)

Medida de la tabla Failures:     (761, 3)

Medida de la tabla Machines:     (100, 3)
```

¿Hay algún valor nulo?

In [7]:
```
print(f"Falta algún valor en Telemetry?\t\t {telemetry_df.isnull().values.any()}")
print(f"Falta algún valor en Errors?\t\t {errors_df.isnull().values.any()}")
print(f"Falta algún valor en Maintenance?\t {maint_df.isnull().values.any()}")
print(f"Falta algún valor en Failures?\t\t {failures_df.isnull().values.any()}")
print(f"Falta algún valor en Machines?\t\t {machines_df.isnull().values.any()}")
```

```
Falta algún valor en Telemetry?          False
Falta algún valor en Errors?             False
Falta algún valor en Maintenance?        False
Falta algún valor en Failures?           False
Falta algún valor en Machines?           False
```

¿Hay algún dato duplicado?

In [8]:
```
print(f"Hay un total de {check_duplicate(telemetry_df, ['datetime', 'machineID','volt','rotate','pressure','vib
print(f"Hay un total de {check_duplicate(errors_df, ['datetime','machineID','errorID'])} errores duplicados")
print(f"Hay un total de {check_duplicate(machines_df, ['machineID','model','age'])} maquinas duplicadas")
print(f"Hay un total de {check_duplicate(failures_df, ['datetime','machineID','failure'])} fallos duplicados")
print(f"Hay un total de {check_duplicate(maint_df, ['datetime','machineID','comp'])} mantenimientos duplicados"
```

```
Hay un total de 0 telemetria duplicadas
Hay un total de 0 errores duplicados
Hay un total de 0 maquinas duplicadas
Hay un total de 0 fallos duplicados
Hay un total de 0 mantenimientos duplicados
```

In [9]:
```
maint_df.head()
```

Out[9]:

	datetime	machineID	comp
0	2014-06-01 06:00:00	1	comp2
1	2014-06-01 06:00:00	6	comp2
2	2014-06-01 06:00:00	9	comp1
3	2014-06-01 06:00:00	9	comp2
4	2014-06-01 06:00:00	11	comp2

Telemetry

Se trata de datos de series temporales de telemetría. Consiste en mediciones de voltaje, rotación, presión y vibración recolectadas de 100 máquinas en tiempo real promediadas sobre cada hora recolectada durante el año 2015.

Los principales campos són:

- datetime: Fecha y hora de adquisición de los datos
- machineID: Identificador de la máquina
- volt: Voltage de la máquina detectado por el sensor
- rotate: Rotación de la máquina detectado por el sensor
- pressure: Presión de la máquina detectado por el sensor
- vibration: Vibración de la máquina detectado por el sensor

In [10]:
```
telemetry_df.head()
```

Out[10]:

	datetime	machineID	volt	rotate	pressure	vibration
0	2015-01-01 06:00:00	1	176.217853	418.504078	113.077935	45.087686
1	2015-01-01 06:00:00	2	176.558913	424.624162	76.005332	43.767049
2	2015-01-01 06:00:00	3	185.482043	461.211137	87.453199	28.216864
3	2015-01-01 06:00:00	4	169.710847	463.646727	95.929877	38.400372
4	2015-01-01 06:00:00	5	165.082899	452.283576	84.591722	40.298803

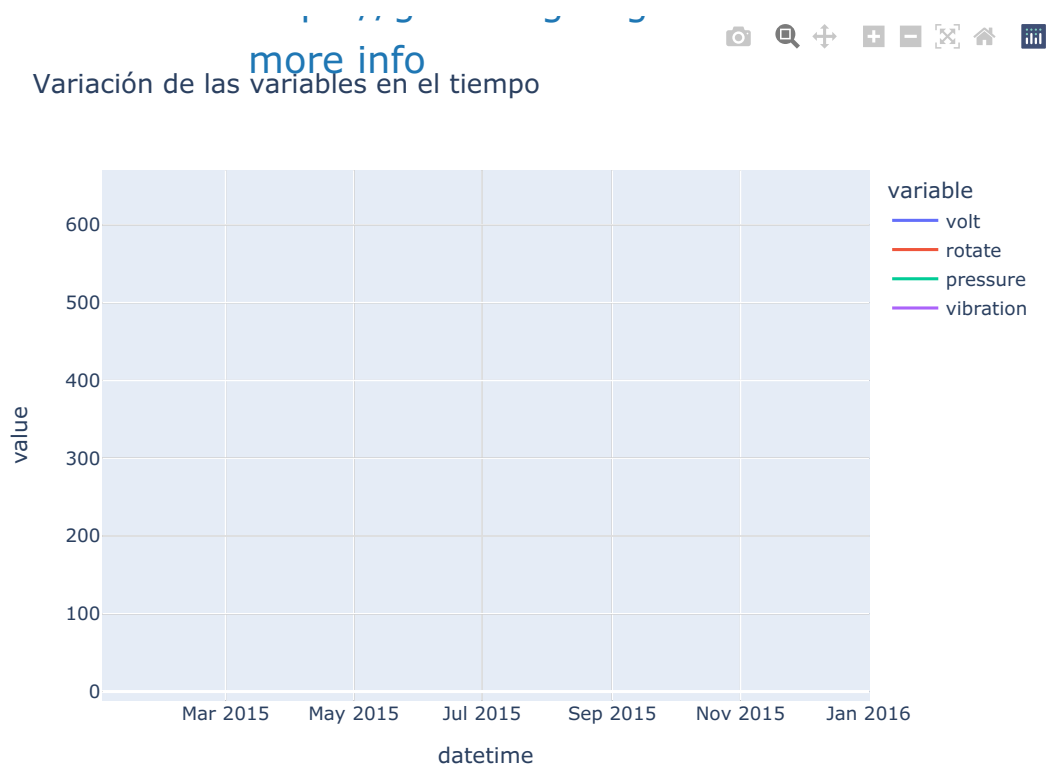
¿Cuantas maquinas hay?

In [11]:
```
telemetry_df.machineID.nunique()
```

Out[11]:
```
100
```

¿Como ha sido el voltage, rotación, pressión y vibración de la maquina 1?

In [12]:
```
# Dataframe maquina 1
m1_telemetry_df = telemetry_df[telemetry_df.machineID == 1]

fig = px.line(m1_telemetry_df, x='datetime', y=['volt', 'rotate', 'pressure', 'vibration'], title="Variación de
fig.show()
```

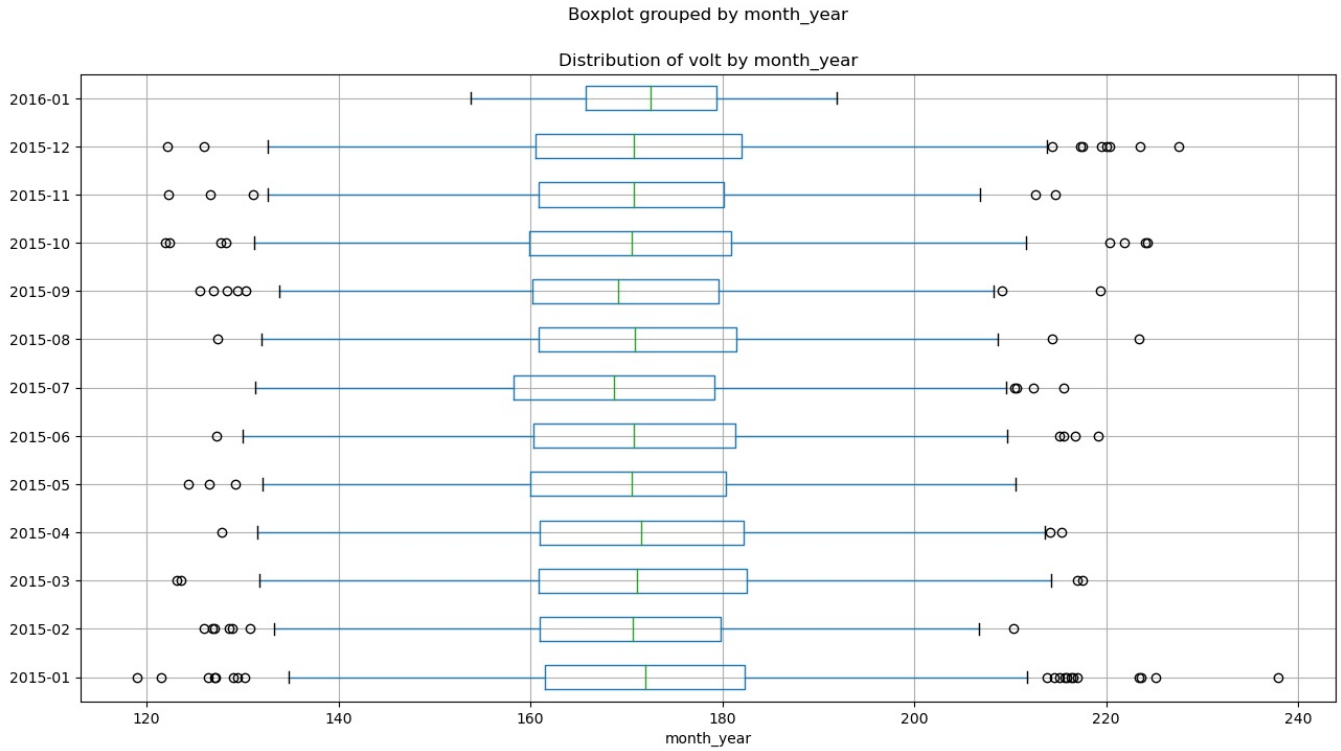


De entrada observamos que hay picos de valores irregulares en diferentes fechas lo que puede ser un indicio de actividad irregular de la máquina.
Por ejemplo en la fecha del 19 de Abril, vemos que hay un pico negativo en la rotación donde es considerablemente inferior a lo normal junto a la presión donde unos dias anteriores tenia un valor más alto en comparación a otros dias.

¿Como varia el voltage al lo largo de los meses?

In [13]:
```
m1_telemetry_df = create_date_features(m1_telemetry_df, m1_telemetry_df, "datetime")
plot_boxh_groupby(m1_telemetry_df, feature_name="volt", by="month_year")
```

```
C:\Users\NetRunner\anaconda3\envs\TFM\lib\site-packages\pandas\core\indexing.py:1596: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\NetRunner\anaconda3\envs\TFM\lib\site-packages\pandas\core\indexing.py:1745: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```



Vemos que el voltaje hay algunas irregularidades del voltaje a lo largo los meses veamos la distribución de este junto con las otras maquinas

Distribución del voltage

In [13]:
```
telemetry_df.volt.describe()
```

Out[13]:
```
count    876100.000000
mean        170.777736
std          15.509114
min          97.333604
25%         160.304927
50%         170.607338
75%         181.004493
max         255.124717
Name: volt, dtype: float64
```

In [15]:
```
telemetry_df.rotate.describe()
```

Out[15]:
```
count    876100.000000
mean        446.605119
std          52.673886
min         138.432075
25%         412.305714
50%         447.558150
75%         482.176600
max         695.020984
Name: rotate, dtype: float64
```

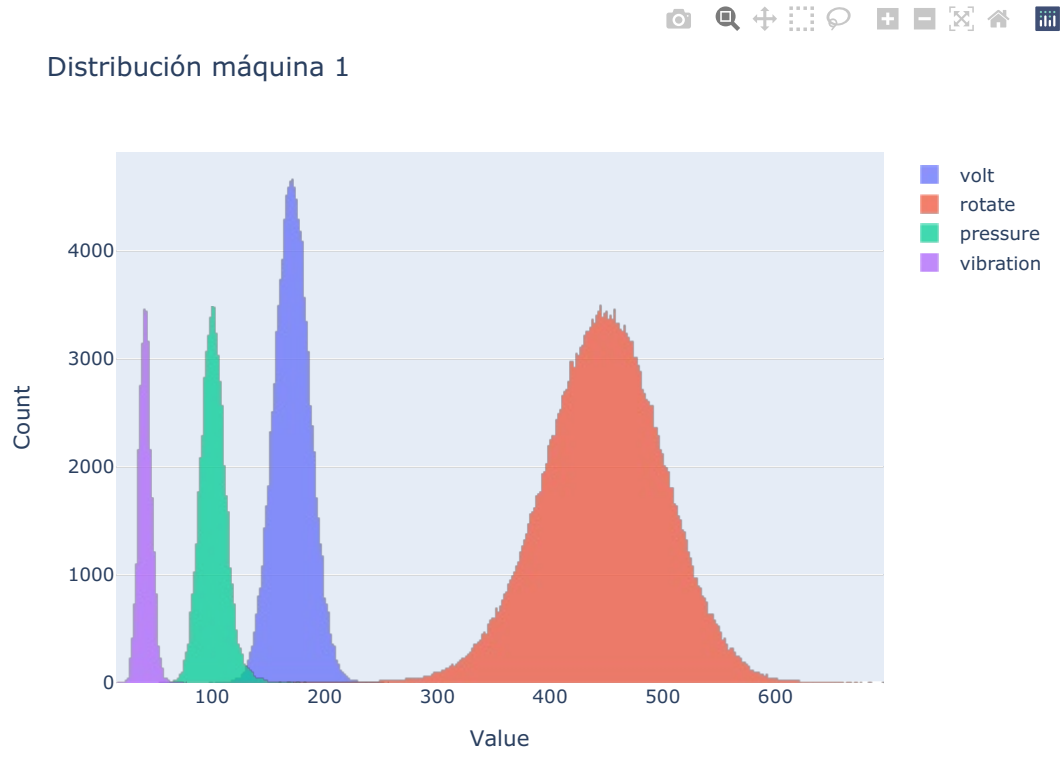
In [17]:
```
telemetry_df.pressure.describe()
```

Out[17]:
```
count    876100.000000
mean        100.858668
std          11.048679
min          51.237106
25%          93.498181
50%         100.425559
75%         107.555231
max         185.951998
Name: pressure, dtype: float64
```

In [19]:
```
telemetry_df.vibration.describe()
```

Out[19]:
```
count    876100.000000
mean         40.385007
std           5.370361
min          14.877054
25%          36.777299
50%          40.237247
75%          43.784938
max          76.791072
Name: vibration, dtype: float64
```

In [22]:
```
fig = go.Figure()
fig.add_trace(go.Histogram(x=telemetry_df['volt'], name='volt'))
fig.add_trace(go.Histogram(x=telemetry_df['rotate'], name='rotate'))
fig.add_trace(go.Histogram(x=telemetry_df['pressure'], name='pressure'))
fig.add_trace(go.Histogram(x=telemetry_df['vibration'], name='vibration'))

# Overlay both histograms
fig.update_layout(barmode='overlay',
                  title_text='Distribución máquina 1',
                  xaxis_title_text='Value',
                  yaxis_title_text='Count',)
# Reduce opacity to see both histograms
fig.update_traces(opacity=0.75)
fig.show()
```



Machines

Este conjunto de datos incluye metadatos sobre las máquinas: tipo de modelo y edad (años en servicio).

Los principales campos són:

- machineID: Identificador de la máquina
- moddel: Modelo de la máquina
- age: Edad/años en servicio

In [21]:
```
machines_df.head()
```

Out[21]:

	machineID	model	age
0	1	model3	18
1	2	model4	7
2	3	model3	8
3	4	model3	7
4	5	model3	2

¿Cuantos modelos hay?

In [22]:
```
print(machines_df.model.unique(),'\n')
print(machines_df.groupby('model').size())
```

```
['model3' 'model4' 'model2' 'model1']

model
model1    16
model2    17
model3    35
model4    32
dtype: int64
```

Distribución de los modelos por edad

In [23]:
```
fig = px.histogram(machines_df, x="age", color="model", title="Modelos-Edad", nbins=30)
fig.update_layout(xaxis={"dtick":1},bargap=0.2)
fig.show()
```

Es interesante destacar que pràcnicamente no hay maquinas con una edad de 13 años. En cuanto a la distribución en si parece ser bastante aleatoria

In [24]:
```
fig = px.box(machines_df, x="model", y="age")
fig.show()
```

Observamos que las máquinas del modelo 4 tienden a tener menos edad en comparación a los otors modelos.

Errors

Estos son errores encontrados por las máquinas mientras están en condiciones de funcionamiento. Dado que estos errores no apagan las máquinas, no se consideran fallas. La fecha y hora del error se redondean a la hora más cercana, ya que los datos de telemetría se recopilan a una tasa por hora.

Los principales campos són:

- datetime: Fecha y hora de adquisición de los datos
- machineID: Identificador de la máquina
- errorID: Identificador del error

In [25]:
```
errors_df.head()
```

Out[25]:

	datetime	machineID	errorID
0	2015-01-01 06:00:00	24	error1
1	2015-01-01 06:00:00	73	error4
2	2015-01-01 06:00:00	81	error1
3	2015-01-01 07:00:00	43	error3
4	2015-01-01 08:00:00	14	error4

¿Cuantos errores hay en total?

In [26]:
```
print(f"Total de {errors_df.shape[0]} errores\n")
print(errors_df.errorID.unique(),'\n')
```

```
Total de 3919 errores

['error1' 'error4' 'error3' 'error5' 'error2'] 
```

¿Como estan distribuidos estos errors?

In [27]:
```
print(errors_df.groupby('errorID').size())
fig = px.histogram(errors_df, x="errorID", title="Errors")
fig.show()
```

```
errorID
error1    1010
error2     988
error3     838
error4     727
error5     356
dtype: int64
```

In [28]:
```
fig = px.histogram(errors_df, x="machineID", color="errorID", title="Errors by machine", nbins=100)
fig.update_layout(xaxis={"dtick":1},bargap=0.1)
fig.show()
```

Aqui desca la máquina con el ID 22 por ser la que tiene más errores, un total de 60 errores, en comparación a las demás. Por otro lado, tenemos a la con ID 9 y la máquina con ID 77 que són las que tienen menos errors siendo 23 i 22 errores respectivamente

¿Cuantos errores se registran por dia?

In [29]:
```
date_errors_df = errors_df.copy()
date_errors_df['datetime'] = pd.to_datetime(date_errors_df['datetime']).dt.date
date_errors_df = date_errors_df.groupby('datetime').size()
```

```
date_errors_df
fig = px.line(date_errors_df, title="Variación de las variables en el tiempo")
fig.update_layout(showlegend=False)
fig.show()
```

¿Que dia del mes y meses se registra mas errores?

In [30]:
```
date_errors_df = errors_df.copy()
date_errors_df = create_date_features(errors_df, errors_df, "datetime")
fig = px.histogram(date_errors_df, x='month', title="Errores por mes")
fig.update_layout(xaxis={"dtick":1},bargap=0.2)
fig.show()
```

In [31]:
```
date_errors_df = errors_df.copy()
date_errors_df = create_date_features(errors_df, errors_df, "datetime")
fig = px.histogram(date_errors_df, x='day', title="Errores por dia del mes")
fig.update_layout(xaxis={"dtick":1},bargap=0.2,showlegend=False)
fig.show()
```

Failures

Cada registro representa el reemplazo de un componente debido a una falla. Estos datos son un subconjunto de los datos de mantenimiento. Estos datos se redondean a la hora más cercana ya que los datos de telemetría se recopilan a una tasa por hora.

Los principales campos són:

- datetime: Fecha y hora de adquisición de los datos
- machineID: Identificador de la máquina
- failure: Componente/causa del fallo

In [32]:
```
failures_df.head()
```

Out[32]:

	datetime	machineID	failure
0	2015-01-02 03:00:00	16	comp1
1	2015-01-02 03:00:00	16	comp3
2	2015-01-02 03:00:00	17	comp4
3	2015-01-02 03:00:00	22	comp1
4	2015-01-02 03:00:00	35	comp1

¿Cuantos fallos ha hbido en total?

In [33]:
```
print(f"Total de {failures_df.shape[0]} fallos\n")
print(failures_df.failure.unique(),'\n')
```

```
Total de 761 fallos

['comp1' 'comp3' 'comp4' 'comp2']
```

¿Como estan distribuidos estos fallos?

In [34]:
```
print(failures_df.groupby('failure').size())
fig = px.histogram(failures_df, x="failure", title="Failures")
fig.show()
```

```
failure
comp1    192
comp2    259
comp3    131
comp4    179
dtype: int64
```

In [35]:
```
fig = px.histogram(failures_df, x="machineID", color="failure", title="Failures by machine", nbins=100)
fig.update_layout(xaxis={"dtick":1},bargap=0.1)
fig.show()
```

En comparación a los datos de errores, vemos que tanto la máquina con ID 6 e ID 77 no han tenido ningún fallo. Por otro lado, la máquina con ID 22 que préviamente se ha observado con muchos errores no parece ser la máquina con más fallos però si uno de los que estan por encima de la media.

¿Cuantos fallos se registran por dia?

In [36]:
```
date_failures_df = failures_df.copy()
date_failures_df['datetime'] = pd.to_datetime(date_failures_df['datetime']).dt.date
date_failures_df = date_failures_df.groupby('datetime').size()
date_failures_df
fig = px.line(date_failures_df, title="Variación de las variables en el tiempo")
fig.update_layout(showlegend=False)
fig.show()
```

¿Que dia del mes y meses se ha registrado más fallos?

In [37]:
```
date_failures_df = failures_df.copy()
date_failures_df = create_date_features(failures_df, failures_df, "datetime")
fig = px.histogram(date_failures_df, x='month', title="Fallos por mes")
fig.update_layout(xaxis={"dtick":1},bargap=0.2)
fig.show()
```

In [23]:
```
date_failures_df = failures_df.copy()
date_failures_df = create_date_features(failures_df, failures_df, "datetime")
fig = px.histogram(date_failures_df, x='day', title="Fallos por dia del dia", nbins=31)
fig.update_layout(xaxis={"dtick":1},bargap=0.2)
fig.show()
```

Maintenance

Son los registros de mantenimiento programado y no programado que corresponden tanto a la inspección periódica de componentes como a fallas. Se genera un registro si un componente se reemplaza durante la inspección programada o se reemplaza debido a una avería. Los registros que se crean debido a fallas se denominarán fallas, lo cual se explica en las secciones posteriores. Los datos de mantenimiento tienen registros de 2014 y 2015.

Los principales campos són:

- datetime: Fecha y hora de adquisición de los datos
- machineID: Identificador de la máquina
- comp: Componente del fallo

In [39]:
```
maint_df.head()
```

Out[39]:

	datetime	machineID	comp
0	2014-06-01 06:00:00	1	comp2
1	2014-06-01 06:00:00	6	comp2
2	2014-06-01 06:00:00	9	comp1
3	2014-06-01 06:00:00	9	comp2
4	2014-06-01 06:00:00	11	comp2

¿Cuantos componentes se le ha hecho mantenimiento en total?

In [40]:
```
print(f"Total de {maint_df.shape[0]} componentes que se le ha hecho mantenimiento\n")
print(maint_df.comp.unique(),'\n')
```

```
Total de 3286 componentes que se le ha hecho mantenimiento

['comp2' 'comp1' 'comp4' 'comp3']
```

¿Como estan distribuidos los mantenimientos?

In [41]:
```
print(maint_df.groupby('comp').size())
fig = px.histogram(maint_df, x="comp", title="Componentes mantenidos")
fig.show()
```

```
comp
comp1    804
comp2    863
comp3    808
comp4    811
dtype: int64
```

In [42]:
```
fig = px.histogram(maint_df, x="machineID", color="comp", title="Componentes reemplazado por máquina", nbins=10
fig.update_layout(xaxis={"dtick":1},bargap=0.1)
fig.show()
```

En comparación a los datos de errores, vemos que tanto la máquina con ID 6 e ID 77 no han tenido ningún fallo. Por otro lado, la máquina con ID 22 que préviamente se ha observado con muchos errores no parece ser la máquina con más fallos però si uno de los que estan por encima de la media.

¿Cuantos fallos se registran por dia?

In [43]:
```
date_maint_df = maint_df.copy()
date_maint_df['datetime'] = pd.to_datetime(date_maint_df['datetime']).dt.date
date_maint_df = date_maint_df.groupby('datetime').size()
```

```
fig = px.line(date_maint_df, title="Reemplazos en el tiempo")
fig.update_layout(showlegend=False)
fig.show()
```

¿Que dia del mes y meses se registra más reemplazos?

In [44]:
```
maint_df
```

Out[44]:

	datetime	machineID	comp
0	2014-06-01 06:00:00	1	comp2
1	2014-06-01 06:00:00	6	comp2
2	2014-06-01 06:00:00	9	comp1
3	2014-06-01 06:00:00	9	comp2
4	2014-06-01 06:00:00	11	comp2
...	...	...	...
3281	2016-01-01 06:00:00	35	comp2
3282	2016-01-01 06:00:00	42	comp4
3283	2016-01-01 06:00:00	43	comp1
3284	2016-01-01 06:00:00	55	comp3
3285	2016-01-01 06:00:00	78	comp3

3286 rows × 3 columns

In [45]:
```
date_maint_df = maint_df.copy()
date_maint_df = create_date_features(maint_df, maint_df, "datetime")
fig = px.histogram(date_maint_df, x='month', title="Reemplazos por mes")
fig.update_layout(xaxis={"dtick":1},bargap=0.2)
fig.show()
```

In [46]:
```
date_maint_df = maint_df.copy()
date_maint_df = create_date_features(maint_df, maint_df, "datetime")
fig = px.histogram(date_maint_df, x='day', title="Mantenimientos por dia del mes", nbins=31)
fig.update_layout(xaxis={"dtick":1},bargap=0.2)
fig.show()
```

Unificación de los datasets

Una vez visto todos los datos disponibles por encima, unificamos los diferentes datasets en uno solo dataset.

In [47]:
```
# Lectura de datos
DATA_DIR = "C:/Users/NetRunner/OneDrive/UOC/Semestre 6/TFM/Data"

telemetry_df = pd.read_csv(f"{DATA_DIR}/PdM_telemetry.csv")
errors_df = pd.read_csv(f"{DATA_DIR}/PdM_errors.csv")
maint_df = pd.read_csv(f"{DATA_DIR}/PdM_maint.csv")
```

```
failures_df = pd.read_csv(f"{DATA_DIR}/PdM_failures.csv")
machines_df = pd.read_csv(f"{DATA_DIR}/PdM_machines.csv")

tables = [telemetry_df, maint_df, failures_df, errors_df]
for df in tables:
    df["datetime"] = pd.to_datetime(df["datetime"], format="%Y-%m-%d %H:%M:%S")
    df.sort_values(["datetime", "machineID"], inplace=True, ignore_index=True)
```

In [48]:
```
machines_df
```

Out[48]:

	machineID	model	age
0	1	model3	18
1	2	model4	7
2	3	model3	8
3	4	model3	7
4	5	model3	2
...	...	...	...
95	96	model2	10
96	97	model2	14
97	98	model2	20
98	99	model1	14
99	100	model4	5

100 rows × 3 columns

In [49]:
```
# Codificamos los errores
error_enc = OneHotEncoder()
error_labels = errors_df['errorID'].values
error_enc.fit(error_labels.reshape(-1,1))
error_enc_labels = error_enc.transform(error_labels.reshape(-1,1)).toarray()
print(error_enc.categories_)
mat_error = np.matrix(error_enc_labels)

# Aplicamos el encoder i añadimos los si se considera fallo
errors_df_enc = pd.concat([errors_df, pd.DataFrame(mat_error).astype(int)], axis=1)
errors_df_enc.columns = ['datetime', 'machineID', 'errorID', 'error1', 'error2', 'error3', 'error4', 'error5']
#errors_df_enc = errors_df_enc.drop(columns=['errorID'])
errors_df_enc = errors_df_enc.groupby(by=['datetime', 'machineID']).sum()
errors_df_enc
```

```
[array(['error1', 'error2', 'error3', 'error4', 'error5'], dtype=object)]
```

Out[49]:

		error1	error2	error3	error4	error5
datetime	**machineID**					
2015-01-01 06:00:00	**24**	1	0	0	0	0
	73	0	0	0	1	0
	81	1	0	0	0	0
2015-01-01 07:00:00	**43**	0	0	1	0	0
2015-01-01 08:00:00	**14**	0	0	0	1	0
...	**...**	...	...	...	...	...
2015-12-31 18:00:00	**91**	0	0	0	1	0
2015-12-31 20:00:00	**23**	0	0	1	0	0
2015-12-31 23:00:00	**94**	0	0	1	0	0
2016-01-01 05:00:00	**8**	0	0	1	0	0
	30	0	1	0	0	0

3616 rows × 5 columns

In [50]:
```
# Codificamos los fallos
failures_enc = OneHotEncoder()
filure_labels = failures_df['failure'].values
failures_enc.fit(filure_labels.reshape(-1,1))
filure_enc_labels = failures_enc.transform(filure_labels.reshape(-1,1)).toarray()
print(failures_enc.categories_)
mat_fail = np.matrix(filure_enc_labels)

# Aplicamos el encoder i añadimos los si se considera fallo
failures_df_enc = pd.concat([failures_df, pd.DataFrame(mat_fail).astype(int)], axis=1)
failures_df_enc.columns = ['datetime', 'machineID', 'failure', 'comp1', 'comp2', 'comp3', 'comp4']
failures_df_enc = failures_df_enc.drop(columns=['failure'])
failures_df_enc = failures_df_enc.groupby(by=['datetime', 'machineID']).sum()
failures_df_enc['failure'] = 1
```

```
failures_df_enc
```

```
[array(['comp1', 'comp2', 'comp3', 'comp4'], dtype=object)]
```

Out[50]:

		comp1	comp2	comp3	comp4	failure
datetime	machineID					
2015-01-02 03:00:00	16	1	0	1	0	1
	17	0	0	0	1	1
	22	1	0	0	0	1
	35	1	0	0	0	1
	45	1	0	0	0	1
...	...	...	...	...	...	...
2015-12-30 06:00:00	88	1	0	0	0	1
2015-12-31 06:00:00	15	0	0	0	1	1
	64	1	0	0	0	1
	90	1	0	0	0	1
	95	1	0	0	0	1

719 rows × 5 columns

In [51]:
```
dataset_tele_err = pd.merge(telemetry_df, errors_df_enc, how='left', left_on=['datetime', 'machineID'], right_o
dataset_tele_err[['error1', 'error2', 'error3', 'error4', 'error5']] = dataset_tele_err[['error1', 'error2', 'e
```

In [52]:
```
dataset_tele_err_fail = pd.merge(dataset_tele_err, failures_df_enc, how='left', left_on=['datetime', 'machineID
dataset_tele_err_fail[['comp1', 'comp2', 'comp3', 'comp4', 'failure']] = dataset_tele_err_fail[['comp1', 'comp2
```

In [53]:
```
dataset = pd.merge(dataset_tele_err_fail, machines_df, how='left', left_on=['machineID'], right_on=['machineID'
dataset
```

Out[53]:

	datetime	machineID	volt	rotate	pressure	vibration	error1	error2	error3	error4	error5	comp1	comp2	comp3	c
0	2015-01-01 06:00:00	1	176.217853	418.504078	113.077935	45.087686	0	0	0	0	0	0	0	0	
1	2015-01-01 06:00:00	2	176.558913	424.624162	76.005332	43.767049	0	0	0	0	0	0	0	0	
2	2015-01-01 06:00:00	3	185.482043	461.211137	87.453199	28.216864	0	0	0	0	0	0	0	0	
3	2015-01-01 06:00:00	4	169.710847	463.646727	95.929877	38.400372	0	0	0	0	0	0	0	0	
4	2015-01-01 06:00:00	5	165.082899	452.283576	84.591722	40.298803	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
876095	2016-01-01 06:00:00	96	157.157424	380.691266	102.885777	35.445739	0	0	0	0	0	0	0	0	
876096	2016-01-01 06:00:00	97	159.607756	458.143799	100.945230	40.011599	0	0	0	0	0	0	0	0	
876097	2016-01-01 06:00:00	98	165.717790	501.520194	114.553412	37.696504	0	0	0	0	0	0	0	0	
876098	2016-01-01 06:00:00	99	168.439623	427.990029	107.899979	44.193151	0	0	0	0	0	0	0	0	
876099	2016-01-01 06:00:00	100	171.336037	496.096870	79.095538	37.845245	0	0	0	0	0	0	0	0	

876100 rows × 18 columns

In [55]:
```
models = list(machines_df.model.value_counts().index)

dict_machines_by_models = {}
for model in models:
    dict_machines_by_models[str(model)] = []

machineID_machines = list(machines_df.machineID)
models_machines = list(machines_df.model)
```

```
for model in models:
    for ind_model_machine, model_machine in enumerate(models_machines):
        if model_machine == model:
            dict_machines_by_models[str(model)].append(machineID_machines[ind_model_machine])

print("Máquinas con modelo:")
for key in dict_machines_by_models:

    print(f"{key}: {dict_machines_by_models[key]}")
```

```
Máquinas con modelo:
model3: [1, 3, 4, 5, 6, 7, 8, 10, 12, 14, 15, 18, 19, 26, 27, 30, 33, 43, 45, 52, 53, 55, 59, 64, 65, 68, 70, 7
5, 79, 80, 82, 84, 86, 89, 93]
model4: [2, 9, 25, 28, 29, 32, 34, 36, 38, 39, 41, 44, 46, 48, 50, 51, 57, 60, 61, 62, 63, 66, 67, 72, 74, 77,
78, 81, 83, 88, 91, 100]
model2: [11, 20, 21, 40, 47, 54, 69, 71, 73, 76, 87, 90, 94, 95, 96, 97, 98]
model1: [13, 16, 17, 22, 23, 24, 31, 35, 37, 42, 49, 56, 58, 85, 92, 99]
```

In [56]:
```
dataset[(dataset['machineID']==1) & (dataset['failure']==1)]
```

Out[56]:

	datetime	machineID	volt	rotate	pressure	vibration	error1	error2	error3	error4	error5	comp1	comp2	comp3	c
9600	2015-01-05 06:00:00	1	179.303153	499.777962	111.833028	52.383097	0	0	0	0	0	0	0	0	
153600	2015-03-06 06:00:00	1	198.257975	456.862342	89.333995	38.671900	0	0	0	0	0	1	0	0	
261600	2015-04-20 06:00:00	1	180.050801	346.362480	105.661164	39.218055	0	0	0	0	0	0	1	0	
405600	2015-06-19 06:00:00	1	187.673963	493.005160	105.334392	53.963961	0	0	0	0	0	0	0	0	
585600	2015-09-02 06:00:00	1	144.094532	409.380150	106.720871	57.454990	0	0	0	0	0	0	0	0	
693600	2015-10-17 06:00:00	1	178.322428	383.715256	79.704008	43.213417	0	0	0	0	0	0	1	0	
837600	2015-12-16 06:00:00	1	137.701308	501.770653	108.557111	47.960218	0	0	0	0	0	0	0	0	

In [59]:
```
dataset.to_csv('C:/Users/NetRunner/OneDrive/UOC/Semestre 6/TFM/Data 2/Dataset.csv', encoding='utf-8', index=Fal
# df.to_csv('folder/subfolder/out.csv')
```