# FSM Online Internship Completion Report
on

# INTP23-ML-5: Equipment Failure Prediction for Predictive Maintenance

In

Machine Learning

Submitted by

Jehan Patel
Vellore Institute of Technology, Bhopal

Under Mentorship of
Dhairya Sir

IITD-AIA Foundation for Smart Manufacturing

[1-June-2021 to 31-July-2021]

# Coding for Predictive Maintenance and Machine Learning


## Abstract

Over the decade, technology has seen a rapid boom and so has robotics seen advancement drastically. During this period, we have seen machines on the rise and its involvement in our day-to-day life. Machines have started to play a crucial role in sustaining humanity and one of the primary locations where machines have a crucial role is in the industrial sector. Machines now have the capacity to process and automate tasks which doesn't require the need of humans. When using machines in the industrial world, we need to make sure that the machine functions adequately and there's regular maintenance done. In order to minimize damage and downtime, we can implement predictive maintenance on the machine to get the machine working efficiently.



Keywords: downtime, damage, predictive, automate, industrial

# Table of Content

## 1. Introduction

Predictive maintenance (PdM) is maintenance that monitors the performance and condition of equipment during normal operation to reduce the likelihood of failures. The goal of predictive maintenance is the ability to first predict when equipment failure could occur (based on certain factors), followed by preventing the failure through regularly scheduled and corrective maintenance. There are various advantages of predictive maintenance from a cost-savings perspective and include minimizing planned downtime, maximizing equipment lifespan, optimizing employee productivity and increasing revenue.

## 2. Problem Definition

Developing a machine learning model to predict equipment failure and enable proactive maintenance. By analysing historical sensor data, maintenance records and operational parameters of industrial equipment, the model will identify maintenance strategies, reduce downtime and optimize the reliability and performance of critical machinery.

## 3. Existing Solution

Developing a machine learning model to predict equipment failure and enable proactive maintenance. By analysing historical sensor data, maintenance records and operational parameters of industrial equipment, the model will identify maintenance strategies, reduce downtime and optimize the reliability and performance of critical machinery. There are various ways that we can predict maintenance which includes through various factors such as heat generated, sounds made by the machine or even through certain abnormalities.

## 4. Proposed Development

Throughout the course of 2 months, I went ahead and proposed to build the dataset properly by making it work with prediction model using data preprocessing and then used the various prebuilt machine learning models to get hold of the predicted values. We will also use various boosting algorithms.

## 5. Functional Development:

## Breast Malignant Dataset

### IMPORT MODULES AND DATA

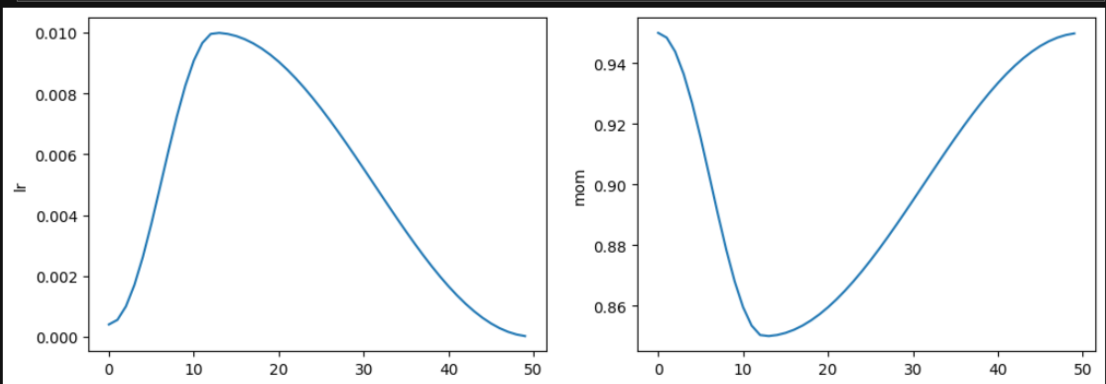
```
In [5]:  import os
         import numpy as np
         import pandas as pd
         from sklearn.metrics import classification_report, log_loss, accuracy_score
         from fastai.tabular.all import *
         pd.options.display.float_format = '{:.2f}'.format
         set_seed(42)

In [6]:  data = pd.read_csv("C:\\Users\\91923\\Downloads\\data (1).csv")
         display(data[0:3].T)
         print(data.columns.tolist())
```

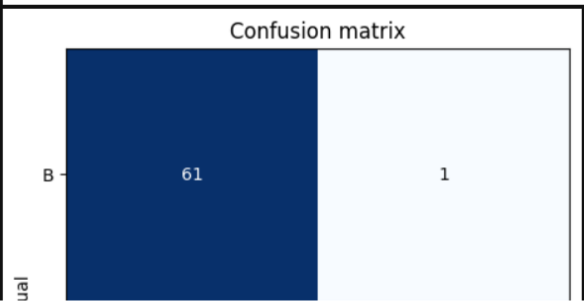
### DATA PREPROCESSING

```
In [7]:  Name=['B','M']
         Name2=['Benign','Malignant']
         N=list(range(len(Name)))
         normal_mapping=dict(zip(Name,N))
         reverse_mapping=dict(zip(N,Name))
         print(normal_mapping)

         {'B': 0, 'M': 1}

In [8]:  display(data['diagnosis'].value_counts())

         B    357
         M    212
         Name: diagnosis, dtype: int64
```

### DATA VISUALIZATION

```
In [18]: learn.recorder.plot_sched()
         interp = ClassificationInterpretation.from_learner(learn)
         interp.plot_confusion_matrix()
```



PREDICTION MODEL

```
In [9]:
m=len(data)
print(m)
M=list(range(m))
random.seed(2021)
random.shuffle(M)

569

In [10]:
train=data.iloc[M[0:(m//4)*3]]
test=data.iloc[M[(m//4)*3:]]
print(len(train),len(test))
testY=test['diagnosis'].map(normal_mapping)
testX=test.drop('diagnosis',axis=1)

426 143

In [11]:
splits = RandomSplitter(seed=42)(train)
display(splits)
print(len(splits[0]),len(splits[1]))

((#341) [251,355,165,159,120,193,413,391,207,16...],
 (#85) [198,168,246,34,158,91,46,294,412,227...])
341 85
```

# CNC Prediction

IMPORT MODULES AND DATA

```
In [24]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

import tensorflow
from tensorflow import keras

from sklearn.metrics import accuracy_score,f1_score,confusion_matrix

In [25]:
train=pd.read_csv("C:\Users\91923\Downloads\train (1).csv")
```

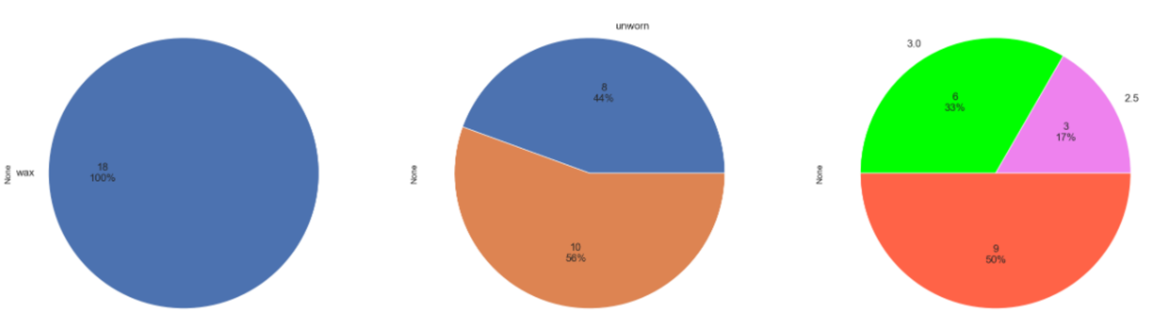
DATA PREPROCESSING

Data was already preprocessed

DATA VISUALIZATION

```
# feedrate
train.groupby('feedrate').size().plot(kind='pie',
                                      autopct=label_function,
                                      textprops={'fontsize': 15},
                                      ax=ax5)
ax5.set_xlabel('feedrate',size=15)

# showing the figure
fig.show()
```

```
C:\Users\91923\AppData\Local\Temp\ipykernel_4176\3352319648.py:53: UserWarning: Matplotlib is currently using module://matp
lotlib_inline.backend_inline, which is a non-GUI backend, so cannot show the figure.
  fig.show()
```




PREDICTION MODEL

```
from sklearn.ensemble import RandomForestClassifier
random_forest=RandomForestClassifier()
random_forest.get_params()
```

```
{'bootstrap': True,
 'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': None,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 100,
 'n_jobs': None,
 'oob_score': False,
 'random_state': None,
 'verbose': 0,
 'warm_start': False}
```

```
random_forest.fit(x_Train,y_Train)
```

```
RandomForestClassifier()
```

```
random_forest_acc=random_forest.score(x_Test,y_Test)
```

```
y_pred=random_forest.predict(x_Test)
accuracy_score(y_true=y_Test,
               y_pred=y_pred)
```

```
0.9911032028469751
```

# Crypto prediction Dataset

## IMPORT MODULES AND DATA

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Activation, Dense, Dropout, LSTM
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.metrics import mean_absolute_error
```

```
data = pd.read_csv("C:\Users\91923\Downloads\12 (2) (Data).csv")
print(data.head(0))
data = data.loc[:,['date','high','low','open','Volume XRP','Volume USDT','close']]
print(data.head(5))
print(data.date)
```

## DATA PREPROCESSING

```
data = data.set_index('date')
data.index = pd.to_datetime(data.index,unit='ns')
print(data.index)
aim = 'close'
```

## DATA VISUALIZATION

```
import matplotlib.pyplot as plt
plt.plot(modelfit.history['loss'],'r',linewidth=2, label='Training loss')
plt.plot(modelfit.history['val_loss'], 'g',linewidth=2, label='Validation loss')
plt.title('LSTM Neural Networks - XRP Model')
plt.xlabel('Epochs numbers')
plt.ylabel('MSE numbers')
plt.show()
```

## PREDICTION MODEL

```
model = build_lstm_model(
    X_train, output_size=1, neurons=lstm_neurons, dropout=dropout, loss=loss,
    optimizer=optimizer)
modelfit = model.fit(
    X_train, y_train, validation_data=(X_test, y_test), epochs=epochs, batch_size=batch_size, verbose=1, shuffle=True)
```

```
import matplotlib.pyplot as plt
plt.plot(modelfit.history['loss'],'r',linewidth=2, label='Training loss')
plt.plot(modelfit.history['val_loss'], 'g',linewidth=2, label='Validation loss')
plt.title('LSTM Neural Networks - XRP Model')
plt.xlabel('Epochs numbers')
plt.ylabel('MSE numbers')
plt.show()
```

```
targets = test[aim][window_len:]
preds = model.predict(X_test).squeeze()
mean_absolute_error(preds, y_test)
```

```
from sklearn.metrics import mean_squared_error
SCORE_MSE=mean_squared_error(preds, y_test)
SCORE_MSE
from sklearn.metrics import r2_score
r2_score=r2_score(y_test, preds)
r2_score*100
```

```
preds = test[aim].values[:-window_len] * (preds + 1)
preds = pd.Series(index=targets.index, data=preds)
line_plot(targets, preds, 'actual', 'prediction', lw=3)
```

# Language Prediction Dataset

## IMPORT MODULES AND DATA

```
In [1]: import pandas as pd
        import numpy as np

        from sklearn.model_selection import train_test_split
        from sklearn.feature_extraction.text import CountVectorizer
        from sklearn.naive_bayes import MultinomialNB
        from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, plot_confusion_matrix
        import seaborn as sns
        import matplotlib.pyplot as plt
        from sklearn.metrics import confusion_matrix
```

```
In [2]: data = pd.read_csv("C:\\Users\\91923\\Downloads\\Language Detection.csv\\Language Detection.csv")
        data.head()
```

## DATA PREPROCESSING

Data is already preprocessed.

## DATA VISUALIZATION

Not required

## PREDICTION MODEL

```
y_pred = model.predict(X_test)

cm = confusion_matrix(y_test, y_pred)

cmap = sns.color_palette("coolwarm", as_cmap=True)

sns.heatmap(cm, annot=True, fmt="d", cmap=cmap,
            xticklabels=model.classes_, yticklabels=model.classes_)

plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')

plt.xticks(rotation=90)

plt.show()
```

# Laptop Cleaning Dataset

## IMPORT MODULES AND DATA

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

from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import (RandomForestRegressor,
                              AdaBoostRegressor ,
                                GradientBoostingRegressor,
                                VotingRegressor,
                                StackingRegressor)

from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error , r2_score
from sklearn.model_selection import GridSearchCV
from catboost import CatBoostRegressor
from xgboost import XGBRegressor
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split

import pickle
import os
```

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv("C:\\Users\\91923\\Downloads\\laptop_data_cleaned.csv")
df.drop_duplicates(inplace=True)
```

```
df
```

## DATA PREPROCESSING

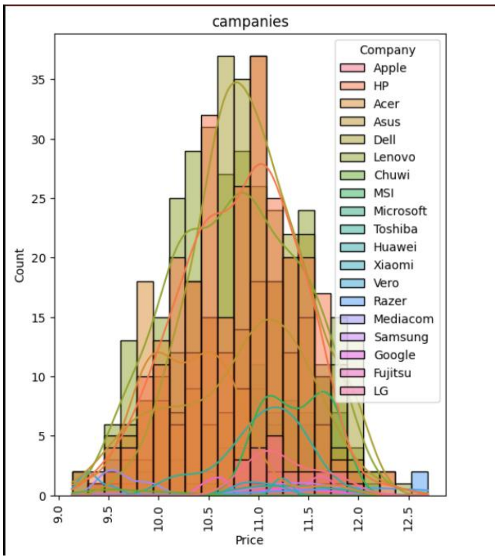
```
cat_cols = [features for features in df.columns if df[features].dtypes == 'O']
num_cols = [features for features in df.columns if df[features].dtypes != 'O']
print(f'cat_cols {len(cat_cols)}')
print(f'num_cols {len(num_cols)}')
```

```
cat_cols 5
num_cols 8
```

```
df[cat_cols]
df[cat_cols].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1272 entries, 0 to 1272
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Company    1272 non-null   object
 1   TypeName   1272 non-null   object
 2   Cpu_brand  1272 non-null   object
 3   Gpu_brand  1272 non-null   object
 4   Os         1272 non-null   object
dtypes: object(5)
memory usage: 59.6+ KB
```

## DATA VISUALIZATION




## PREDICTION MODEL

```
y_pred = model.predict(X_test)

cm = confusion_matrix(y_test, y_pred)

cmap = sns.color_palette("coolwarm", as_cmap=True)

sns.heatmap(cm, annot=True, fmt="d", cmap=cmap,
            xticklabels=model.classes_, yticklabels=model.classes_)

plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')

plt.xticks(rotation=90)

plt.show()
```

Here are a few more of the project code’s I have done during the 2 month period

1. Largest Universities
2. Marathon Runners
3. Medical Insurance
4. Taxi Fare

5. Weather Prediction

## 6. Final Deliverable

Throughout the course of 2 months, These are the project codes I have done for the internship

1. Breast Malignant Prediction: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Breast%20Malignant%20Prediction

2. CNC Prediction: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/CNC%20Prediction

3. Cryptocurrency Prediction: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Cryptocurrency%20Dataset

4. Language Prediction: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Language%20Prediction%20Dataset

5. Laptop Cleaning Prediction: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Laptop%20Cleaning%20Prediction

6. Largest Universities: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Largest%20Universities

7. Marathon Runners: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Marathon%20Runner%20Prediction

8. Medical Insurance: https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Medical%20Insurance

9. Taxi Fare:

https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Taxi%20Fare

10. Weather Prediction:
https://github.com/JehanPatel/FSM-INT-2023/tree/main/Practice%20Code%20and%20Datasets/Weather%20Prediction

## 7. Innovation In Implementation

One of the key implementations which I did in my code is the use of boosting algorithm. I went through various Boosting Algorithms and one of the major boosting algorithms which I found out was XG Boosting Algorithm and ADA Boosting Algorithm. Out of them, the one algorithm which I used was the ADA Boosting Algorithm. AdaBoost works by fitting one weak learner after the other. In subsequent fits, it gives more weight to incorrect predictions and less weight to correct predictions. In this way, the models learn to make predictions for the difficult classes.

## 8. Scalability to solve Industrial Problem

Predictive Maintenance is very scalable. We can scale the code and work done easily with the help of 3$^{rd}$ party tech parties. Since Predictive Maintenance is a rapidly evolving field, work will be prevalent and it will also help upskill. We can easily use the same model steps to increase the amount of data used in the dataset and gain even more clearer knowledge about the same. We can also scale it by implementing the same model into the ecosystem to save costs and increase revenue and decrease losses.

## 9. References


1. https://medium.com/analytics-vidhya/a-beginners-guide-for-getting-started-with-machine-learning-7ba2cd5796ae
2. https://www.youtube.com/watch?v=BrvwLVpwD6o&t=7s
3. https://www.youtube.com/watch?v=NWONeJKn6kc&t=17s
4. https://www.researchgate.net/publication/326837099_Design_and_Fabrication_of_Automated_Pneumatic_Shearing_Machine_to_Cut_Aluminium_Sheet