CAD-IT IoT Centre

Henry Wirawan Halim

Answer – Machine Learning Application engineer

1. The data Q1.csv contains the two temperature and two vibration sensors data for two chiller pumps with a column of to indicate the condition of normal or fault (0 indicating normal and 1 indicating fault).
   a. Insight of the datasets:

      Summary of the datasets

	CP1T1	CP1T2	CP2T1	CP2T2	CP1V1	CP1V2	CP2V1	CP2V2	Fault
count	13847.000000	13847.000000	13847.000000	13847.000000	13847.000000	13847.000000	13847.000000	13847.000000	13847.000000
mean	38.894831	38.372753	37.049863	37.083295	0.998612	1.345662	0.934057	1.763645	0.393082
std	1.657766	1.818556	2.927990	3.476318	0.616521	0.853053	0.619965	1.268073	0.488452
min	34.272883	33.652465	31.602933	31.247236	0.046510	0.045864	0.068607	0.082864	0.000000
25%	38.336334	37.233618	35.394773	34.523069	0.070278	0.067707	0.091273	0.108075	0.000000
50%	39.569812	39.364623	38.227170	37.395959	1.322656	1.778066	0.935697	1.670248	0.000000
75%	39.970923	39.671286	39.413810	40.465354	1.473305	1.967611	1.525421	2.952790	1.000000
max	41.325293	40.638881	43.187444	41.962697	1.610700	2.390570	1.785126	3.950299	1.000000

Counts of the faulty condition

```
1 df.Fault.value_counts()
```

executed in 22ms, finished 15:50:05 2021-04-14

```
0    8404
1    5443
```

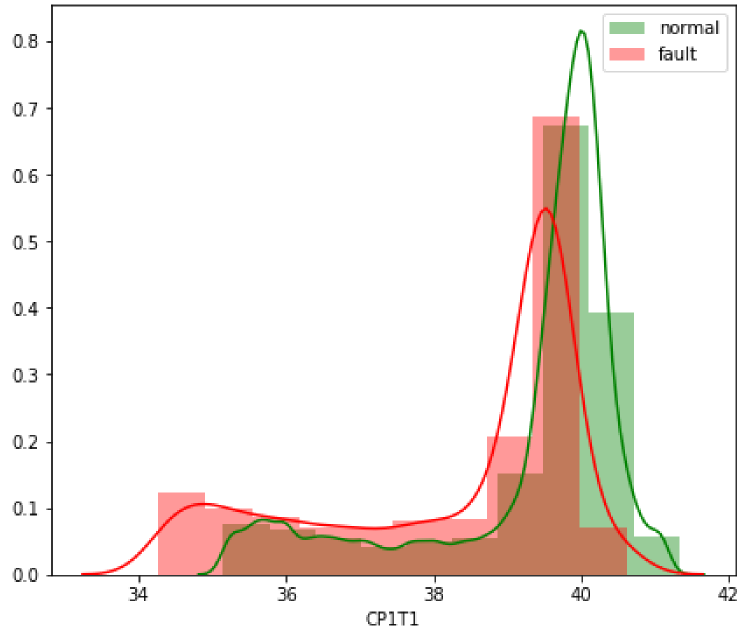
The dataset is balance (giving data of each target's condition)

Correlation of the temperature sensor 1 on chiller pump 1 and the pump faulty

From the figure below we can see that the sensor doesn't affect too much on the pump faulty.

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP1T1[df.Fault==0], bins=10, color='g', label='normal')
sns.distplot(df.CP1T1[df.Fault==1], bins=10, color='r', label='fault')
plt.legend();
```

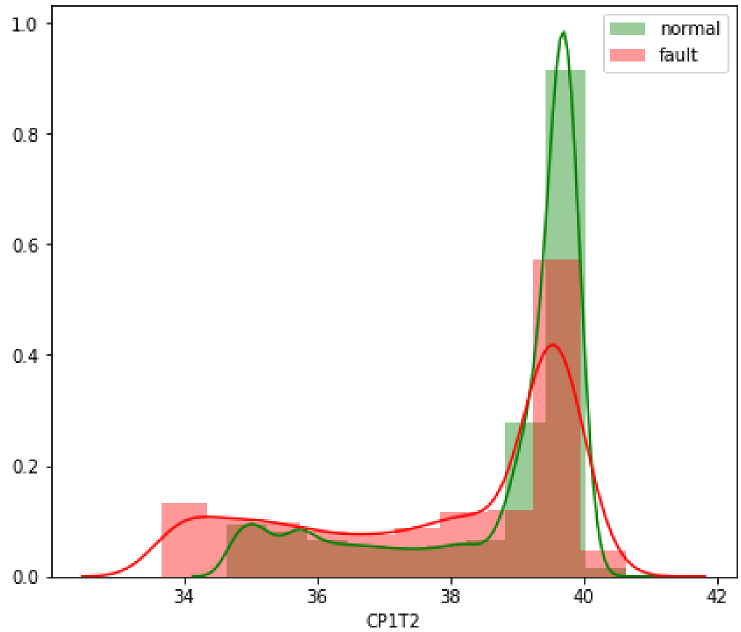
executed in 935ms, finished 15:50:06 2021-04-14



Correlation of the temperature sensor 2 on chiller pump 1 and the pump faulty

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP1T2[df.Fault==0], bins=10, color='g', label='normal')
sns.distplot(df.CP1T2[df.Fault==1], bins=10, color='r', label='fault')
plt.legend();
```

executed in 815ms, finished 15:50:07 2021-04-14

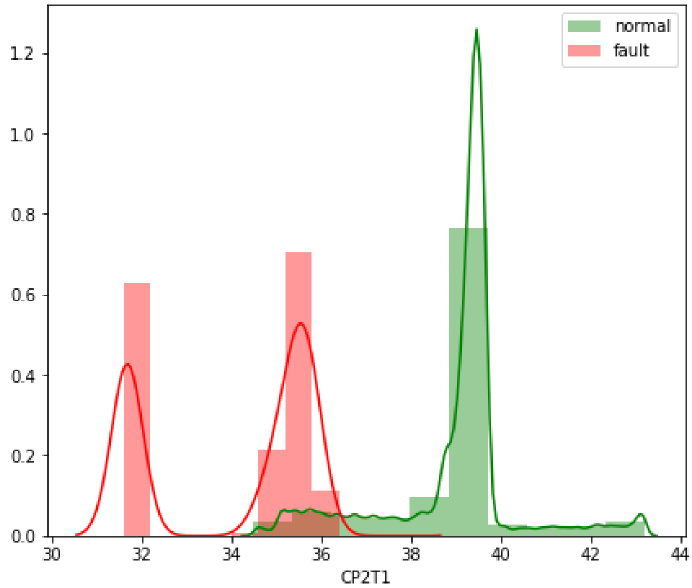


From the figure above we can see that the sensor doesn't affect too much on the pump faulty.

Correlation of the temperature sensor 1 on chiller pump 2 and the pump faulty

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP2T1[df.Fault==0], bins=10, color='g', label='normal')
sns.distplot(df.CP2T1[df.Fault==1], bins=10, color='r', label='fault')
plt.legend();
```

executed in 821ms, finished 15:50:08 2021-04-14

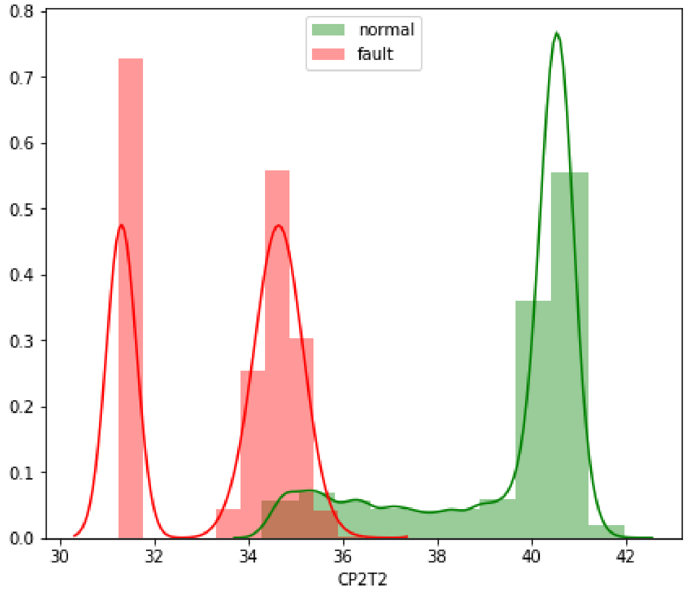


From the figure above we can see that most of pump fault happen when the sensor's temperature is below 37°C.

Correlation of the temperature sensor 2 on chiller pump 2 and the pump faulty

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP2T2[df.Fault==0], bins=10, color='g', label='normal')
sns.distplot(df.CP2T2[df.Fault==1], bins=10, color='r', label='fault')
plt.legend();
```

executed in 782ms, finished 15:50:09 2021-04-14



From the figure above we can see that most of pump fault happen when the sensor's temperature is below 37°C.

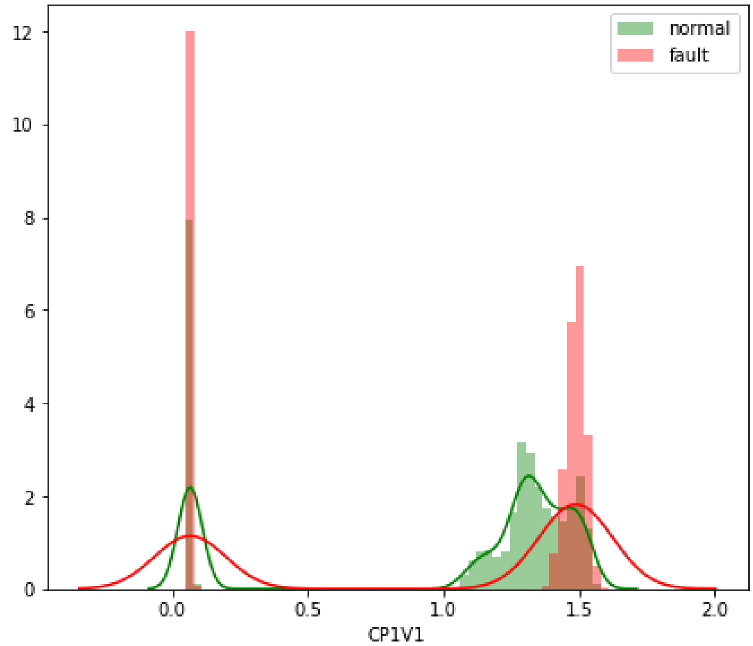
Correlation of temperature sensor of chiller pumps and the pump faulty

Both temperature sensor on chiller pump 2 have a high correlation with pumps faulty and both temperature sensor on chiller pump 1 don't have a high correlation with pumps faulty.

Correlation of the vibration sensor 1 on chiller pump 1 and the pump's faulty

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP1V1[df.Fault==0], bins=50, color='g', label='normal')
sns.distplot(df.CP1V1[df.Fault==1], bins=50, color='r', label='fault')
plt.legend();
```

executed in 1.39s, finished 15:50:10 2021-04-14



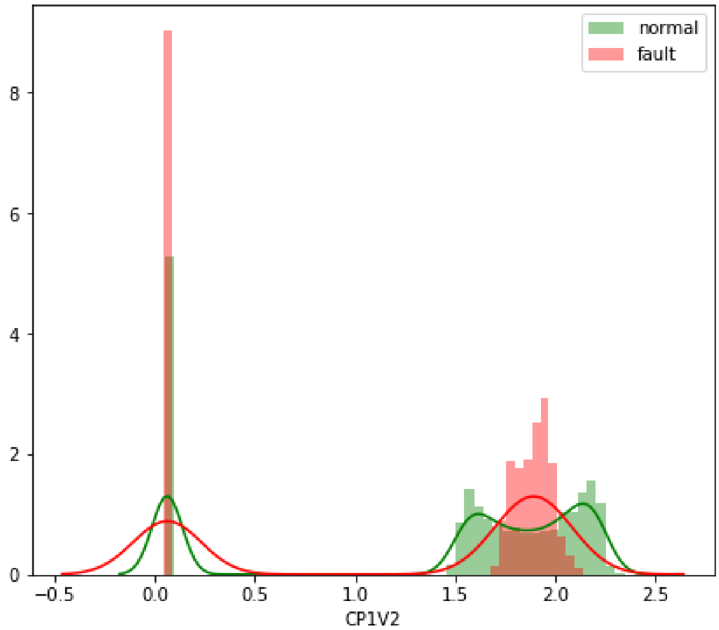
From the figure above we can see that the sensor doesn't affect too much on the pump faulty.

Correlation of the vibration sensor 2 on chiller pump 1 and the pump's faulty

From the figure below we can see that the sensor doesn't affect too much on the pump faulty.

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP1V2[df.Fault==0], bins=50, color='g', label='normal')
sns.distplot(df.CP1V2[df.Fault==1], bins=50, color='r', label='fault')
plt.legend();
```

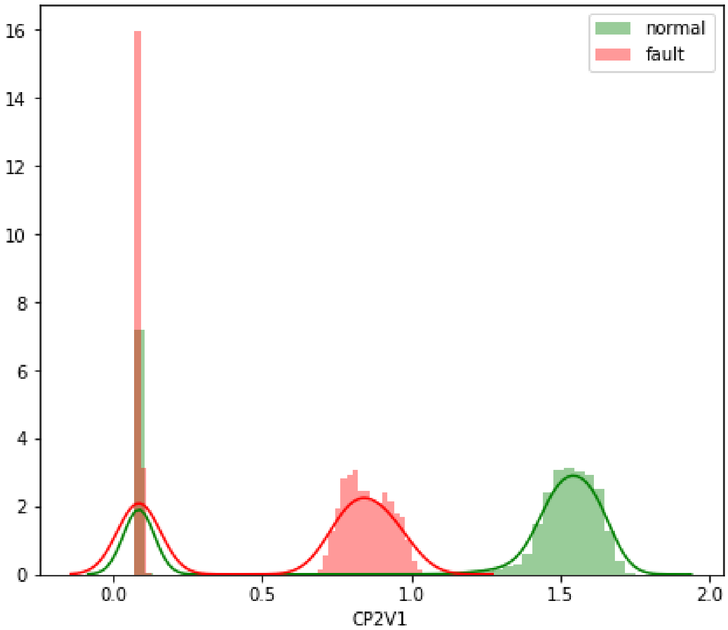
executed in 1.23s, finished 15:50:11 2021-04-14



Correlation of the vibration sensor 1 on chiller pump 2 and the pump's faulty

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP2V1[df.Fault==0], bins=50, color='g', label='normal')
sns.distplot(df.CP2V1[df.Fault==1], bins=50, color='r', label='fault')
plt.legend();
```

executed in 1.46s, finished 15:50:13 2021-04-14

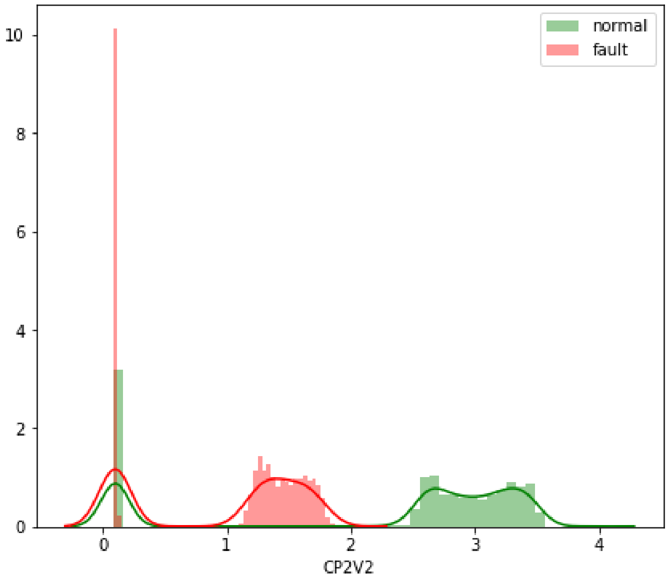


From the figure above we can see that most of pump fault happen when the sensor's vibration is about 0.6 – 1 mm/s.

Correlation of the vibration sensor 2 on chiller pump 2 and the pump's faulty

```
plt.figure(figsize=(7, 6))
sns.distplot(df.CP2V2[df.Fault==0], bins=50, color='g', label='normal')
sns.distplot(df.CP2V2[df.Fault==1], bins=50, color='r', label='fault')
plt.legend();
```

executed in 1.51s, finished 15:50:14 2021-04-14



From the figure above we can see that most of pump fault happen when the sensor's vibration is about 1.1 – 1.8 mm/s

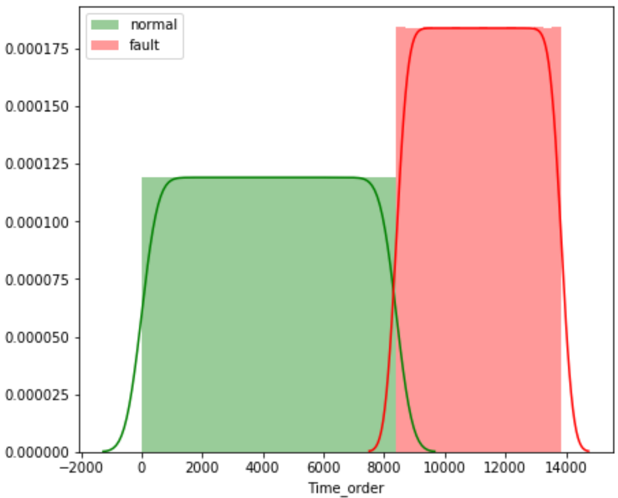
Correlation of vibration sensor of chiller pumps and the pump faulty

Both vibration sensor on chiller pump 2 have a high correlation with pumps faulty and both vibration sensor on chiller pump 1 don't have a high correlation with pumps faulty.

Correlation of timestamps and the pump faulty

```
plt.figure(figsize=(7, 6))
sns.distplot(df.Time_order[df.Fault==0], color='g', label='normal')
sns.distplot(df.Time_order[df.Fault==1], color='r', label='fault')
plt.legend();
```

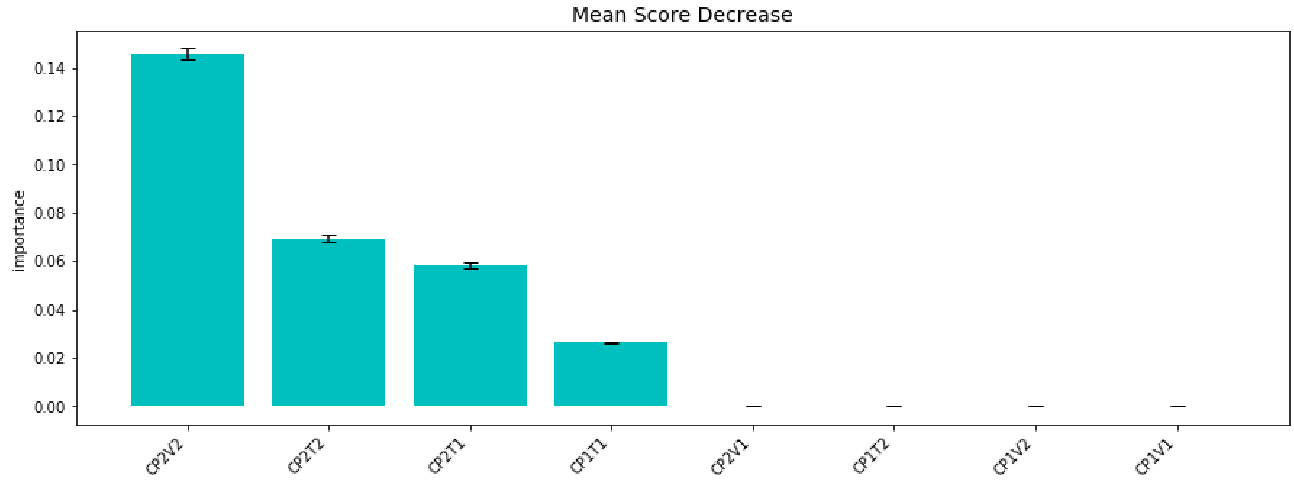
executed in 1.04s, finished 15:50:15 2021-04-14



The timestamp didn’t affect on the pump safety. From the data, the pump faulty is happen when the time is over 2018-10-17. So we must drop the timestamp data for the prediction.

Feature Importance

After the prediction model is made, each feature (sensor data) correlation is tested by changing its value and see if the score is decreasing. The less important feature is if we change the value of its data and the score accuracy doesn’t change much then we can assume that the feature has less correlation or less important of predicting the target (pump faulty) and vice versa.



And the result shows that each vibration sensor of chilling pump have the least importance on predicting the pump faulty.

b. Train a classification model

The pump faulty model is trained and has **99.9638989% accuracy** on the test data (split 20% random data of the datasets). You can see the code details on “Q1.ipynb” file.

c. Simple application to predict

First, we load the model and create the predicting function.

Bonus (predicting program)

```
pump_faulty_predictor = load_model("model/pump_faulty.pkl")
```

executed in 81ms, finished 15:52:55 2021-04-14

Assuming that the data input is in json format

Each timestamp send the data to machine to predict fault

```
def predict_pump_faulty(sensor_data):
    X_pred = pd.DataFrame([sensor_data], columns=['CP1T1', 'CP1T2', 'CP2T1', 'CP2T2', 'CP1V1', 'CP1V2', 'CP2V1', 'CP2V2'
    faulty = pump_faulty_predictor.predict(X_pred)[0]
    print("PUMP FAULT!") if faulty else print("Pump is fine")
    return faulty
```

executed in 15ms, finished 15:52:55 2021-04-14

Then we make the input (the two temperature and two vibration sensors data for the two chiller pumps).

```python
sensor_data0 = {
    "CP1T1":38.8948,
    "CP1T2":38.3727,
    "CP2T1":37.0498,
    "CP2T2":37.0832,
    "CP1V1":0.9986,
    "CP1V2":1.3456,
    "CP2V1":0.9340,
    "CP2V2":1.7636
}
sensor_data1 = {
    "CP1T1":38.8948,
    "CP1T2":38.3727,
    "CP2T1":32.0498, # change the vital sensor that affect the prediction
    "CP2T2":37.0832,
    "CP1V1":0.9986,
    "CP1V2":1.3456,
    "CP2V1":0.9340,
    "CP2V2":1.7636
}
```

executed in 25ms, finished 15:52:55 2021-04-14

Then we call the function to get the predicted output from the model.

Predicting

```python
predict_pump_faulty(sensor_data0)
```

executed in 154ms, finished 15:52:55 2021-04-14

```
Pump is fine
```

```
7]: 0
```

```python
predict_pump_faulty(sensor_data1)
```

executed in 160ms, finished 15:52:55 2021-04-14

```
PUMP FAULT!
```

```
3]: 1
```

Or you can try with open the "Q1.py" app by open the command prompt and type: python Q1.py –input "your input". With the input file format is in json.

Anaconda Prompt (miniconda3)

```
(jcopml) C:\Python\Machine Learning Application Engineer Test 2021 V1.1>python Q1.py --input sensor_data_test0.json
Pump is fine
>>> Pump status: 0
```

2. Develop a simple application which extract the 'Description', 'Possible Root cause' and the page number as three columns to store into a database or a CSV file.

Name	Date modified	Type	Size
.git	4/15/2021 3:02 PM	File folder	
.ipynb_checkpoints	4/15/2021 3:17 PM	File folder	
model	4/14/2021 1:11 AM	File folder	
Q3	4/13/2021 3:41 PM	File folder	
Machine Learning Application Engineer T...	4/2/2021 6:44 PM	Adobe Acrobat D...	166 KB
Q1.csv	3/17/2020 10:00 AM	Microsoft Excel Co...	1,272 KB
Q1.ipynb	4/15/2021 3:40 AM	IPYNB File	1,343 KB
Q2.ipynb	4/15/2021 4:04 PM	IPYNB File	115 KB
Q2.pdf	2/1/2021 9:24 PM	Adobe Acrobat D...	314 KB
Q2.py	4/15/2021 4:03 PM	Python Source File	3 KB
Q3.ipynb	4/15/2021 2:50 AM	IPYNB File	18 KB

The application is named "Q2.py"

To start and test the application, you have to:

1. Run Command Prompt
2. Type python Q2.py --input "your input file like Q2.pdf" --output "output file"

   Examples:

```
Anaconda Prompt (miniconda3) - conda  install -c conda-forge poppler

(text) C:\Python\Machine Learning Application Engineer Test 2021 V1.1>python Q2.py --input Q2.pdf --output Q2.csv
Done.
```

3. Check your csv output file

Name	Date modified	Type	Size
.git	4/15/2021 3:02 PM	File folder	
.ipynb_checkpoints	4/15/2021 3:17 PM	File folder	
model	4/14/2021 1:11 AM	File folder	
Q3	4/13/2021 3:41 PM	File folder	
Machine Learning Application Engineer T...	4/2/2021 6:44 PM	Adobe Acrobat D...	166 KB
Q1.csv	3/17/2020 10:00 AM	Microsoft Excel Co...	1,272 KB
Q1.ipynb	4/15/2021 3:40 AM	IPYNB File	1,343 KB
Q2.csv	4/15/2021 4:08 PM	Microsoft Excel Co...	6 KB
Q2.ipynb	4/15/2021 4:04 PM	IPYNB File	115 KB
Q2.pdf	2/1/2021 9:24 PM	Adobe Acrobat D...	314 KB
Q2.py	4/15/2021 4:03 PM	Python Source File	3 KB
Q3.ipynb	4/15/2021 2:50 AM	IPYNB File	18 KB

```
df = pd.read_csv("Q2.csv")
df.head()
```

executed in 30ms, finished 16:02:57 2021-04-15

	Description	Possible_Root_cause	Page_number
0	INDOOR PCB ABNORMALITY	['Faulty indoor PCB.', 'Faulty connector conne...	35
1	ANTIFREEZE PROTECTION OR HIGH PRESSURE CONTROL	['Indoor air short circuit.', 'Indoor coil the...	36
2	INDOOR FAN MOTOR ABNORMALITY	['Indoor fan motor winding short, or the motor...	37
3	INDOOR HEAT EXCHANGER THERMISTOR ABNORMALITY	['Thermistor, connector faulty.', 'Indoor PCB ...	37
4	INDOOR ROOM THERMISTOR ABNORMALITY	['Thermistor, connector faulty.', 'Indoor PCB ...	38

3. Train a model to classify the sentences for all the text in ‘Q3’ folder.

    Step to handle a text data:

    1. Extract features and targets on text then combine into a dataframe.
    2. Because the data only has two columns which are the text and the label, we can directly split the datasets into feature and target.
    3. After splitting the data, we can just train the data using scikit-learn’s pipeline.
    4. For the preprocessor, use word tokenize to tokenize sentence using nltk.
    5. Fitting the model
    6. The model has 100% accuracy
    7. Save the model

    You can see the code details on “Q3.ipynb” file.