Injection Molded Product Fault Detection and Diagnostic

2022-1 Digital Twin & Automation

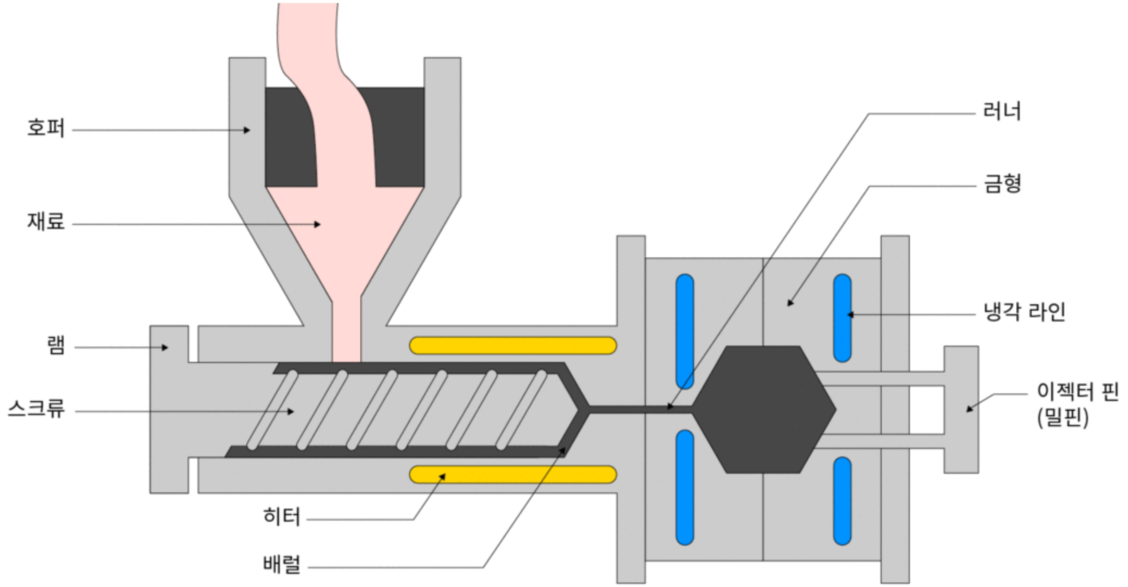
17 박정우 | 17 유진수 | 17 홍세현

한동대학교
HANDONG GLOBAL UNIVERSITY

Injection Molding Quality Prediction

What is the Injection Molding

- A method of manufacturing in which synthetic resins such as plastics are melted, injected into molds, and cooled to produce the desired form of products.



Injection Molding Process

1. Supply of material.
2. Melting plastic raw materials.
3. Injection of melt into mold at high pressure and constant rate.
4. Melt cooling
5. Product Completion.

Ref. https://creatable.com/molding/guide/design/0

Application of Machine Learning Techniques in Injection Molding Quality Prediction: Implications on Sustainable Manufacturing Industry

What is the problem in manufacturing site

- Injection molding can be produced defective due to pressure, temperature, injection time, etc.

- Increased quality costs due to higher wages.

- Efforts of manufacturing companies to improve production efficiency due to the Industry 4.0

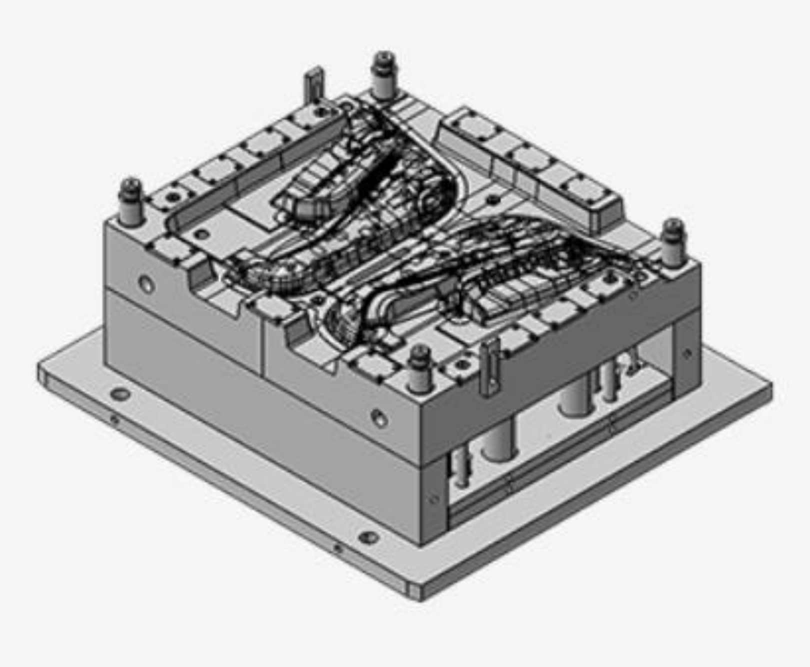
What is the goal of paper

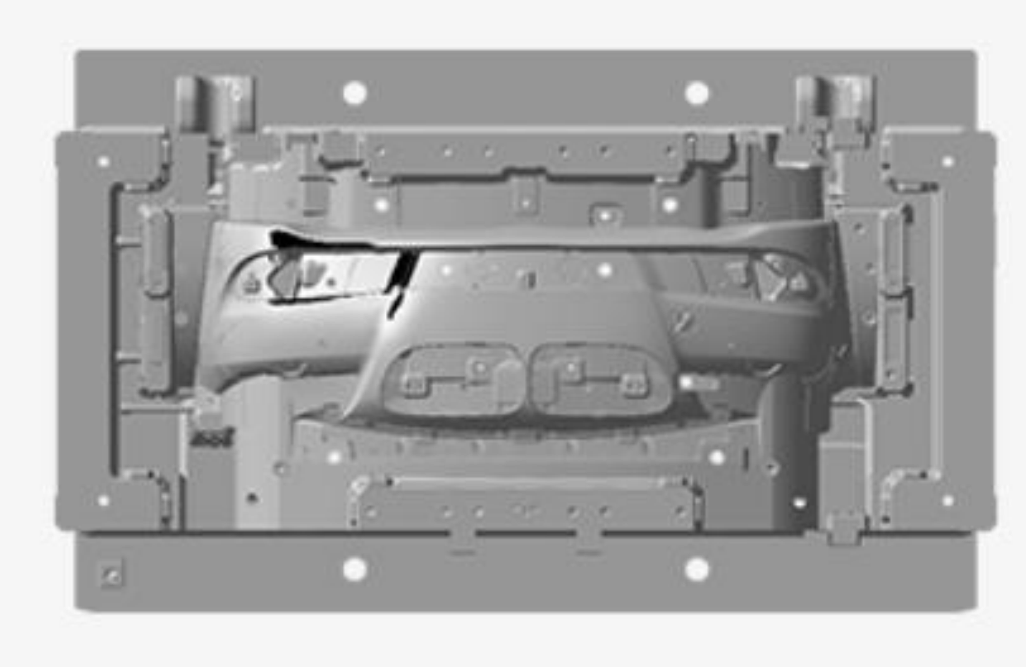
- Comparison of performance of quality prediction using algorithms of machine learning and deep learning.

Injection Molding Quality Prediction

Data Information

- Data Set: Injection Molding Production and Quality Dataset(Good: 8024, Defect: 125)
- Data Source: Korea Mold (Automotive Parts Manufacturer)
- Data Configuration: Injection Time(s), Maximum injection rate(mm/s), Maximum injection pressure(MPA), Mold temperature(μC) 50 and more..





Injection Molding Quality Prediction

Pre-processing

- they are selected variables that are considered more important in the manufacturing sites
- The Defect ratio is relatively low.
- They did over-sampling using SMOTE(synthetic minority oversampling technique method)
  *Defect N: 125 -> 5655

Variable Name	
Injection Time(s)	Max Screw RPM(RPM)
Filling Time(s)	Average Screw RPM(RPM)
Plasticizing Time(s)	Max Injection Pressure(MPa)
Cycle Time(s)	Max Switch Over Pressure(MPa)
Clamp Close Time(s)	Max Back Pressure(MPa)
Switch Over Position(mm)	Average Back Pressure(MPa)
Plasticizing Position(mm)	Barrel Temperature(◦C)
Clamp Open Position(mm)	Mold Temperature(◦C)
Max Injection Speed(mm/s)	

Injection Molding Quality Prediction

Feature Extraction

- **They got Statistical Features**

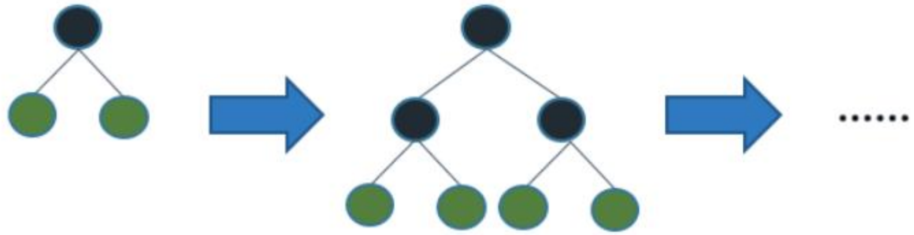
All Observations		Good		Defect		Difference in Means
Mean	Std	Mean	Std	Mean	Std	T-Test

Injection Molding Quality Prediction

Model selection

- Regression based, Tree based, Autoencoder
- Accuracy, precision, Recall and F1 score comparison of each model.

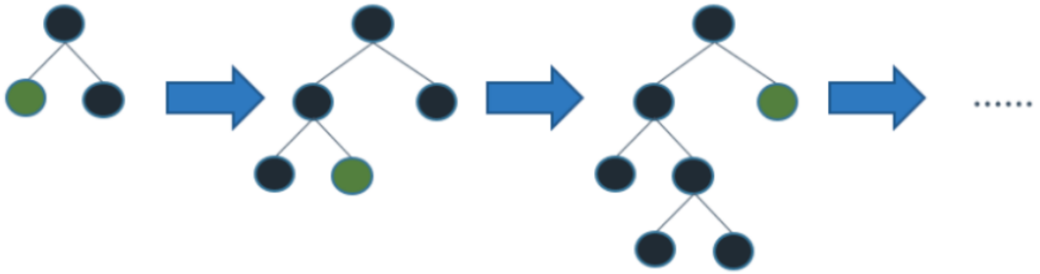
XGboost



Level-wise tree growth

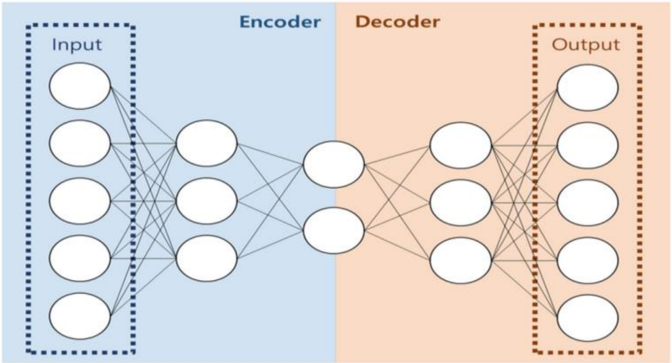
Catboost

LightGBM



Leaf-wise tree growth

Auto encoder



Injection Molding Quality Prediction

Model Evaluate

Panel A. Regression-based models				
	Accuracy	**Precision**	**Recall**	**F1-Score**
Logistic Regression	0.8449	0.0833	0.8947	0.1521
Support Vector Machine	0.8642	0.0961	0.9210	0.1741
Panel B. Tree-based models				
	Accuracy	**Precision**	**Recall**	**F1-Score**
Random Forest	0.9918	0.7647	0.6841	0.7222
Gradient Bootsing	0.9862	0.5576	0.7638	0.6444
XGBoost	0.989366	0.6761	0.6052	0.6388
CatBoost	0.9905	0.6923	0.7105	0.7012
LightGBM	0.9914	0.7575	0.6578	0.7042
Panel C. Autoencoder model				
	Accuracy	**Precision**	**Recall**	**F1-Score**
Autoencoder	0.9959	0.9469	1.0000	0.9727

Abnormal product diagnosis about Wind Shield Side Molding

What is Wind Shield Side Molding ?

[1] External molding that finishes both ends of the front glass prevents noise and contamination during driving.

[2] detachable part during front glass repair or replacement

[3] **Using a gas injection molding method in manufacturing**

Injection Molded Product Fault Detection and Diagnostic

Problem Definition

[1] Difficult to check the defect of the molded product with naked eye

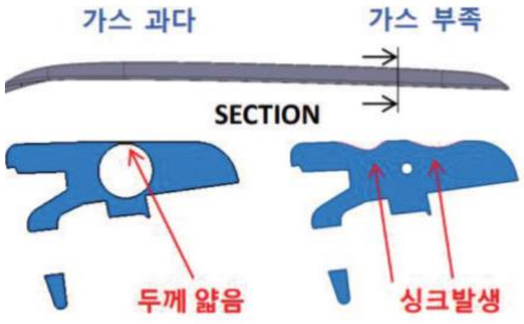
└ **Destruction inspection must be accompanied** to determine defective products.

└ Destruction inspections are performed at regular intervals,
and if a defect is determined, a certain number of molded products before and after are recycled

[2] After the molded product cools and contracts, defects occur and can be confirmed only then.

└ **Difficult to immediately control the process** in the event of a defect

Economic and Time Loss is Huge !



Injection Molded Product Fault Detection and Diagnostic

Problem Definition

How to diagnose defective products in real time without destruction inspection?

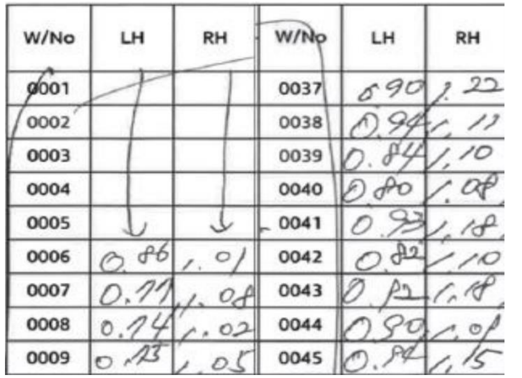
IDEA :: the temperature of the cross-section is distributed differently during the process of cooling the product.

IDEA	Expected Effect
Diagnose abnormal product with thermal image	No destruction test required
Full inspection of all products immediately after injection molding through thermal image	Enables real-time process control and avoids unnecessary disposal

Data Overview

Source	KAMP AI Manufacturing AI dataset [Machine Vision AI Dataset]
Data Collection equipment	IR camera

Data Type	Data Information	Data Segmentation	Number of data
Raw Train Data	Thermal Image	Left Train Data	414 * 256 * 320 pixels
		Right Train Data	423 * 256 * 320 pixels
Raw Label Data	Handwritten thickness information	Left Label Data	414 labels
		Right Label Data	423 labels

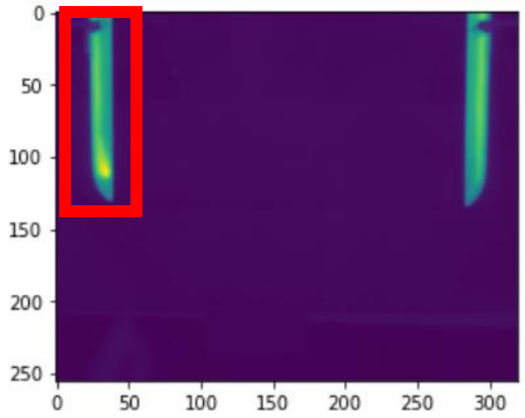


W/No	LH	RH	W/No	LH	RH
0001			0037	0.90	1.22
0002			0038	0.94	1.11
0003			0039	0.84	1.10
0004			0040	0.80	1.08
0005			0041	0.93	1.18
0006	0.86	1.01	0042	0.82	1.10
0007	0.77	1.08	0043	0.82	1.18
0008	0.74	1.02	0044	0.80	1.01
0009	0.73	1.05	0045	0.84	1.15

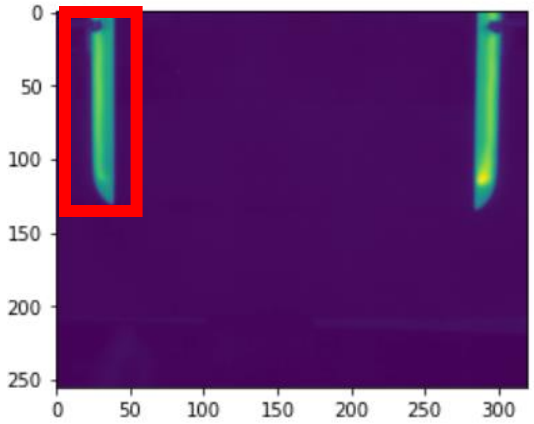
Each Thermal Image = 256*320 resolution → 81,920 pixels per image

Injection Molded Product Fault Detection and Diagnostic

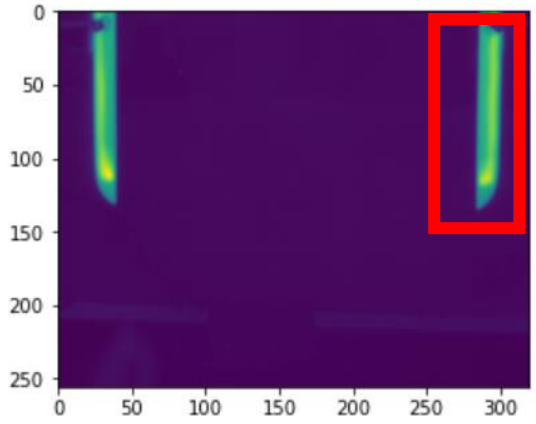
Data Overview



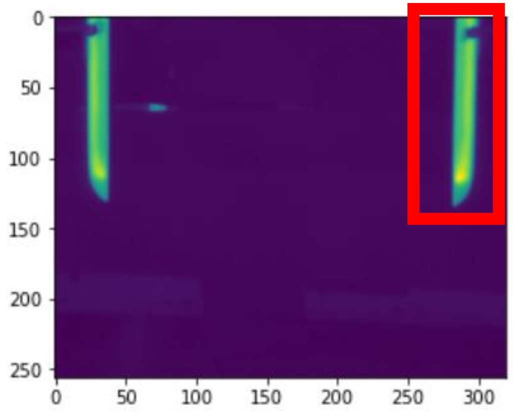
Raw Left Data (Normal)



Raw Left Data (Abnormal)



Raw Right Data (Normal)

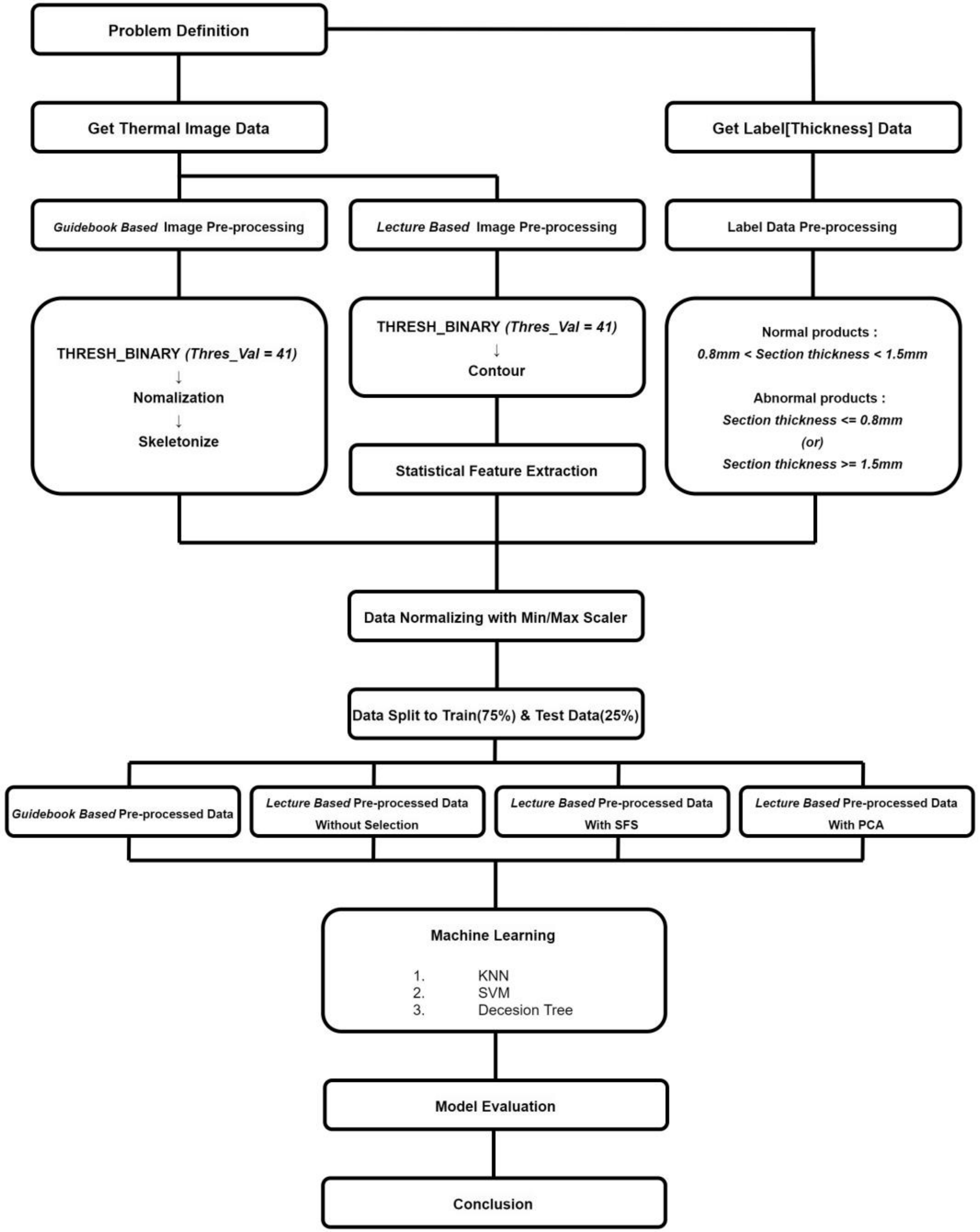


Raw Right Data (Abnormal)

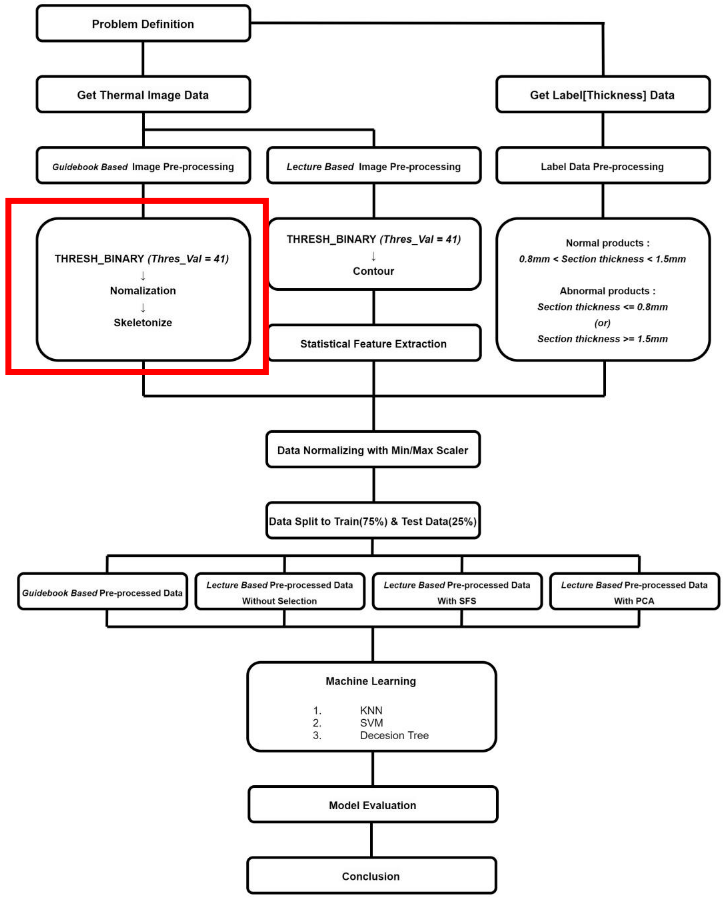
Thermal image plotting and data pre-processing were handled by Python OpenCV in the Jupiter

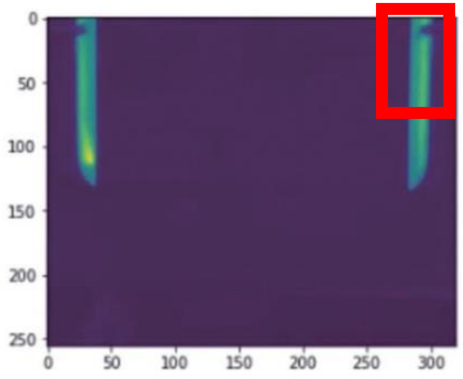
Injection Molded Product Fault Detection and Diagnostic

Flow Chart

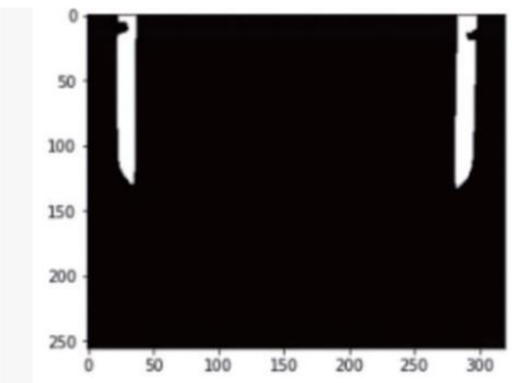


Algorithm[1] Pre-Processing *(Guide Based)*

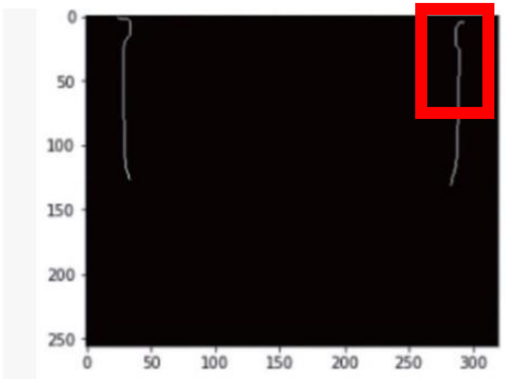




Raw Right Data



Threshold(THRESH_BINARY)



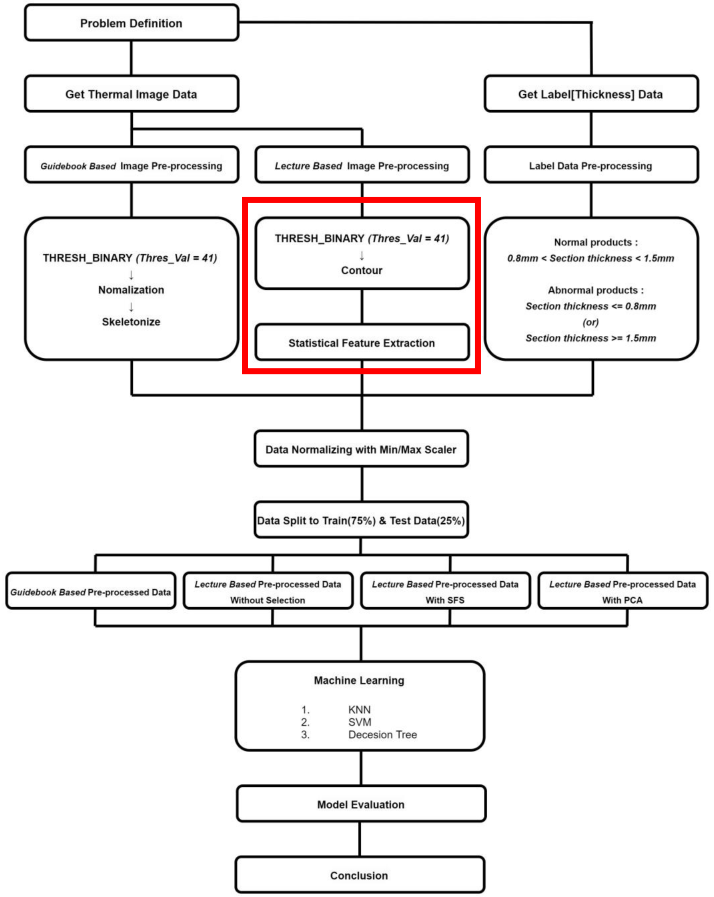
Skeleton Right Image

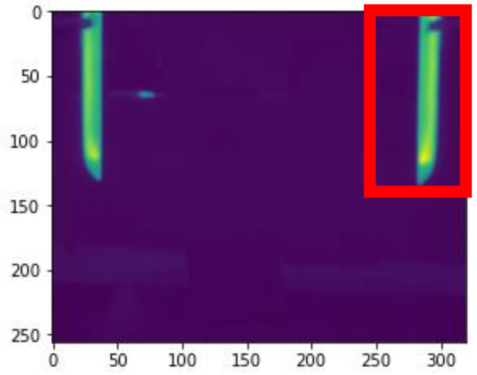
Get data location about region of interest

[Extract only indexes in rows 1 to 80]

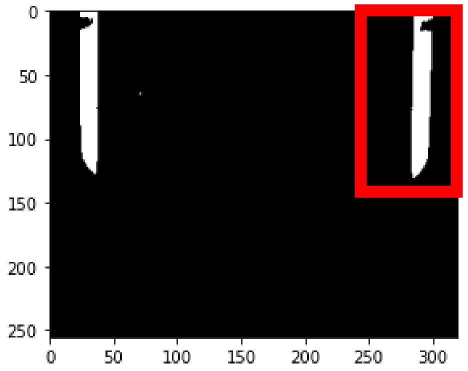
Injection Molded Product Fault Detection and Diagnostic

Algorithm[1] Pre-Processing *(Lecture Based)*





Raw Right Data



Threshold(THRESH_BINARY)

Injection Molded Product Fault Detection and Diagnostic

Output :: 171st image, 50st row

In [18]:
```python
# Verifying that the preprocessing was performed successfully
# Gets all the values in the 10th row (out of 256X320) of the first image.
cont_image[170][50]
```

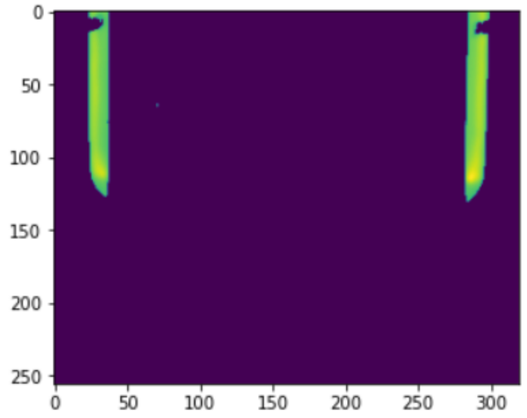
Out[18]:
```
array([  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0., 255., 255., 255., 255., 255., 255., 255., 255., 255.,
       255., 255., 255., 255., 255.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., 255., 255.,
       255., 255., 255., 255., 255., 255., 255., 255., 255., 255., 255.,
       255.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.])
```

Verified that **Contour**
was successfully applied to 'BINARY' image

In [19]:
```python
Mask_image = cv.bitwise_and(flir_image, cont_image)

plt.imshow(Mask_image[170])
Mask_image[170].max()
```

Out[19]: 61.0



Masking Contour results to raw images

Injection Molded Product Fault Detection and Diagnostic

Output :: 171st image, 50st row

In [20]:
```python
flir_left = np.zeros((423, 255, 160));
flir_right = np.zeros((423, 255, 160));

for i in range(0,423):
    flir_left[i]  = Mask_image[i][0:255, 0:160]
    flir_right[i] = Mask_image[i][0:255,160:320]

plt.figure(figsize = (10,10))
plt.subplot(1,2,1)
plt.imshow(flir_left[170])
plt.subplot(1,2,2)
plt.imshow(flir_right[170])
```

Out[20]: <matplotlib.image.AxesImage at 0x1c9ba77be80>

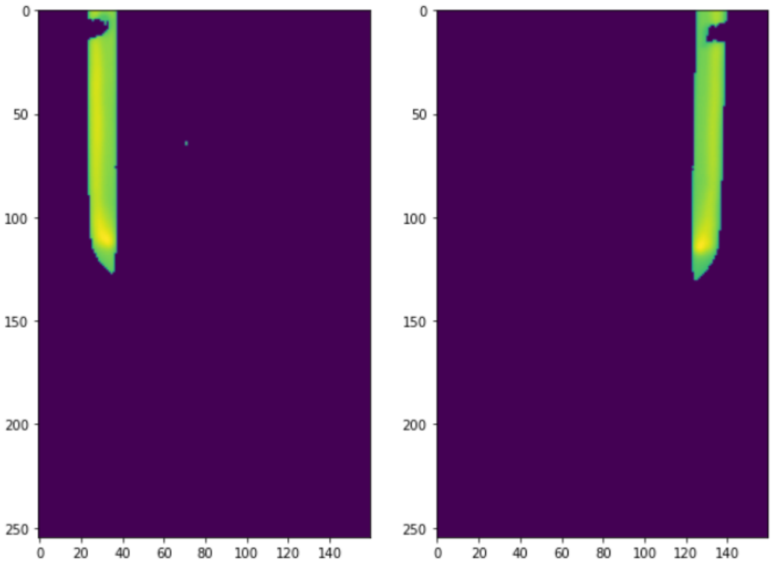


Photo Segmentation, Dividing masked images to left images and right images

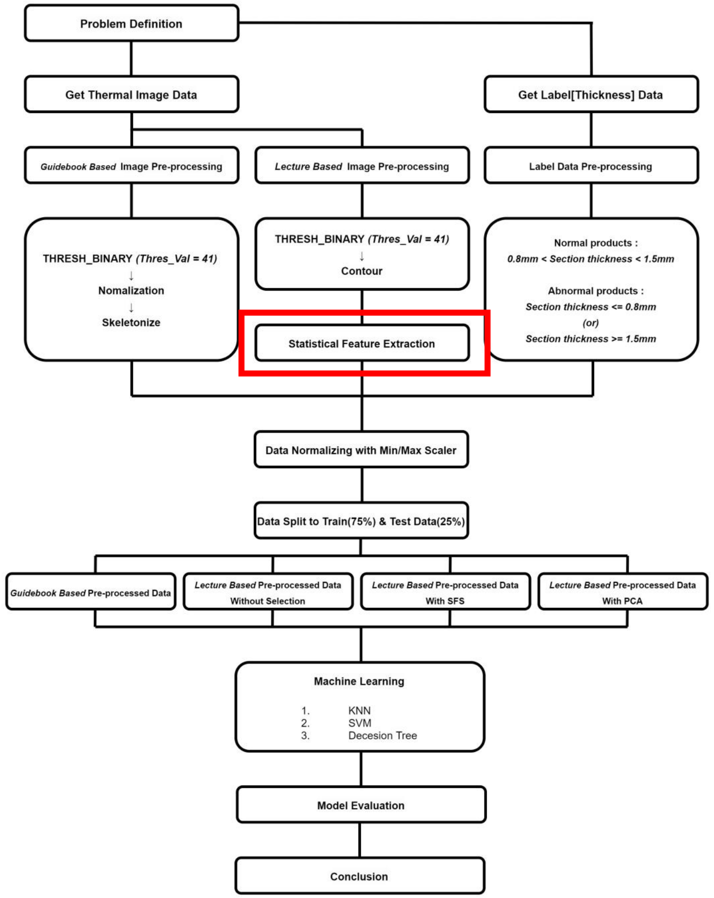
In [21]:
```python
flir_right[170][50]
```

Out[21]:
```
array([ 0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  , 44.  , 47.  ,
       48.  , 48.75, 49.  , 49.5 , 50.75, 52.25, 53.5 , 54.  , 53.75,
       52.  , 49.75, 46.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,
        0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ,  0.  ])
```

Extracted only the intensity value of the region of interest through Contour

Injection Molded Product Fault Detection and Diagnostic

Algorithm[2] Feature Extraction



Features	The Meaning of Each Features
Skewness Value	For Defective Product, data distribution might be biased to the right or to the left
Kurtosis Value	For Defective Product, an Outlier is generated and the data distribution is spread
Peak to Peak	For Defective Product, an Outlier is generated and P2P value might be increase
Marginal Factor	For Defective Product, an Outlier is generated and MF value might gradually decrease
Min	For Defective Product, an Outlier is generated and Min/Max Value might be change greatly
Max	
Mean	For Defective Product, the overall temperature will be high or low
Square Root Average	
Impulse Factor	Impulse Factor is to find the most prominent value

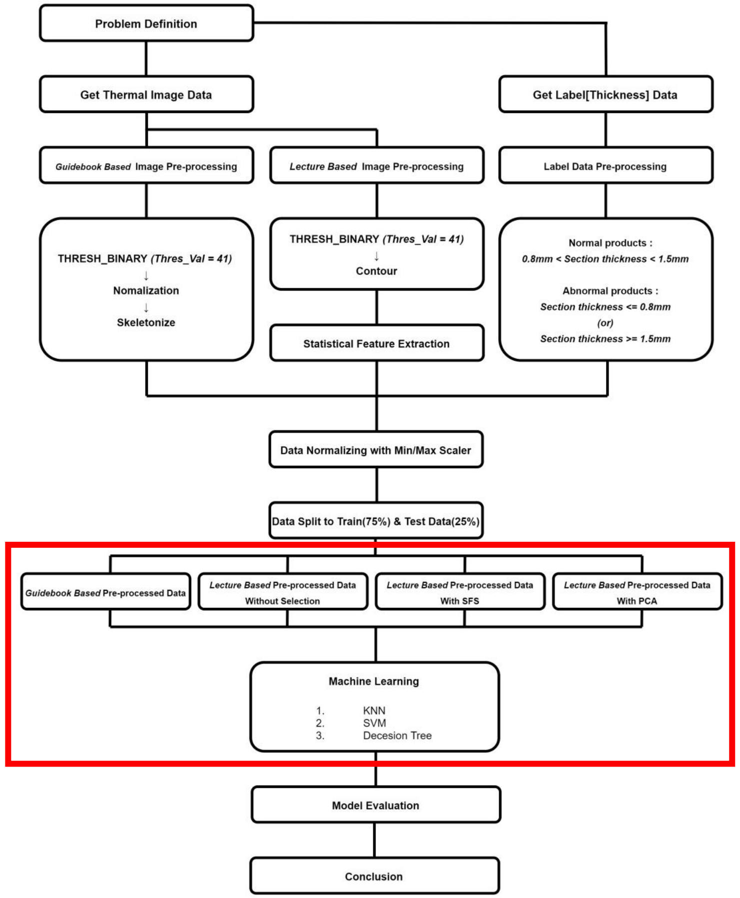
RMS Value

Calculating for frequency data such as vibration signals.
The data we have is Image Intensity data, so RMS value was not used.

Other statistical features derived from RMS,
such as Crest Factor, Shape Factor and Impulse Factor, were not used too.

Injection Molded Product Fault Detection and Diagnostic

Algorithm[3] Machine Learning

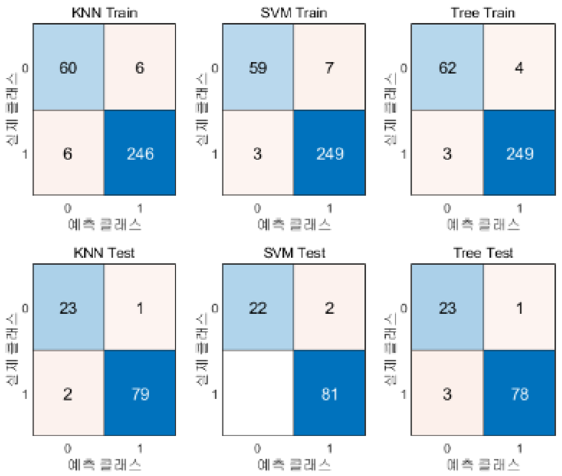


Classification Machine Learning about RIGHT dataset

KNN - SVM - TREE

```
[Knn_Test_Loss_R, SVM_Test_Loss_R, Tree_Test_Loss_R] = KNN_SVM_TREE(Right_Train_Data, Right_Train_Label, Right_Test_Data, Right_Test_Label, 5);
```

```
k = 3
Knn_CrossValication_Error = 0.0629
Knn_Train_Loss = 0.0377
Knn_Test_Loss = 0.0282
SVM_CrossValication_Error = 0.0660
SVM_Train_Loss = 0.0314
SVM_Test_Loss = 0.0173
Tree_CrossValication_Error = 0.0566
Tree_Train_Loss = 0.0220
Tree_Test_Loss = 0.0380
```



Confusion Matrix about

Right Train/Test Data without Selection

KNN	**K = 3**
SVM	**Gaussian Kernel SVM** was Selected
Decision Tree	**Default Value** was Used

Injection Molded Product Fault Detection and Diagnostic

Compare Accuracy

LEFT DATA

```
Accuracy_L = [Accuracy_Table_Guide_L, Accuracy_Table_Without_Selection_L, Accuracy_Table_With_Feature_Selection_L, Accuracy_Table_With_PCA_L]
```

```
Accuracy_L = 3x4 table
```

		GuideBook_Accuracy_L	Without_Selection_Accuracy_L	With_Sequential_Selection_Accuracy_L	With_PCA_Accuracy_L
1	KNN	$\boxed{93.1216}$	88.7521	85.6805	86.8376
2	SVM	83.1829	$\boxed{89.5896}$	88.6722	88.6722
3	Tree	83.9989	82.7686	82.9285	85.4408

RIGHT DATA

```
Accuracy_R = [Accuracy_Table_Guide_R, Accuracy_Table_Without_Selection_R Accuracy_Table_With_Feature_Selection_R Accuracy_Table_With_PCA_R]
```

```
Accuracy_R = 3x4 table
```

		GuideBook_Accuracy_R	Without_Selection_Accuracy_R	With_Sequential_Selection_Accuracy_R	With_PCA_Accuracy_R
1	KNN	$\boxed{95.3032}$	97.1785	98.1569	98.1569
2	SVM	87.3770	$\boxed{98.2704}$	97.4057	95.4490
3	Tree	93.0979	96.2002	96.2002	96.2002

No significant difference in accuracy from the pre-processed data provided in the guidebook !

→ Used fewer features, achieved good accuracy

Comparison with Previous Studies

<table>
<tr><th colspan="2" rowspan="2"></th><th rowspan="2">KAMP Guide Book</th><th colspan="2">DTA Project</th></tr>
<tr><th>KAMP Guide Book</th><th>Project Dataset</th></tr>
<tr><td colspan="2">Model</td><td>SVM Linear
SVM Polynomial
SVM RBF</td><td colspan="2">KNN
SVM Gaussian
Decision Tree</td></tr>
<tr><td colspan="2">Image Pre-Processing</td><td colspan="2">Skeletonize</td><td>Contour</td></tr>
<tr><td rowspan="2">Feature</td><td>Type</td><td colspan="2">Intensity Value</td><td>Statistical Features</td></tr>
<tr><td>Number</td><td colspan="2">80</td><td>9</td></tr>
<tr><td colspan="2">Max Accuracy for Left</td><td>-</td><td>93.12% at KNN</td><td>89.59% at SVM Gaussian</td></tr>
<tr><td colspan="2">Max Accuracy for Right</td><td>95.44% at SVM RBF</td><td>95.30% at KNN</td><td>98.27% at SVM Gaussian</td></tr>
</table>

Thank you ☺

Q & A

Reference

[1] Jung, H., Jeon, J., Choi, D., & Park, J. Y. (2021). Application of Machine Learning Techniques in Injection Molding Quality Prediction: Implications on Sustainable Manufacturing Industry. *Sustainability*, *13*(8), 4120.

[2] KAMP, 제조 AI 데이터셋, 머신비전 AI 데이터셋, 열화상 이미지를 이용한 양/불량 판정을 위한 머신비전 데이터, KAIST, 2020.12.14,