데이터 분석을 통한 유방암 진단

제작 : 문경환

01 CONTENTS

Overview

Purpose

- AI를 기반으로 한 유방암 진단

Summary

- 데이터 분석을 통한 주요 feature 추출로 ML 학습 Score 향상(91% -> 94% & Overfitting 방지)

  -> barplot, violinplot, heatmap, correlation 사용

- 이미지 데이터를 통한 최적의 DL 모델 구축과 고찰(최대 91%)

  -> tensorflow, keras 사용

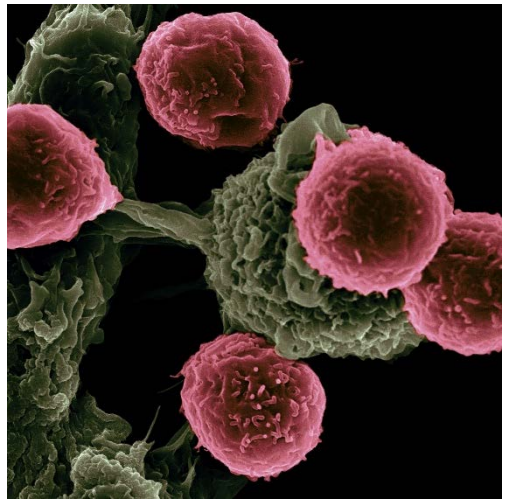
CONTENTS

01

Breast Cancer diagnosis with ML

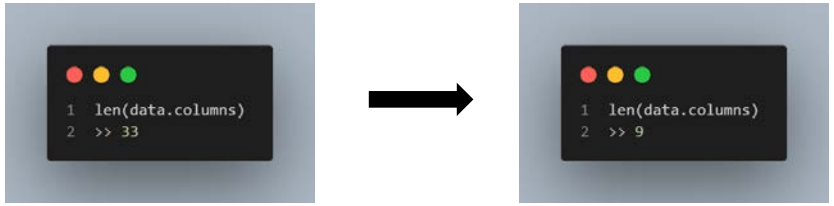
01 CONTENTS

Inspection & Target



Breast Cancer

- 세계에서 가장 흔한 여성 암
- 생활습관이 세계적으로 바뀜에 따른 큰 증가 폭



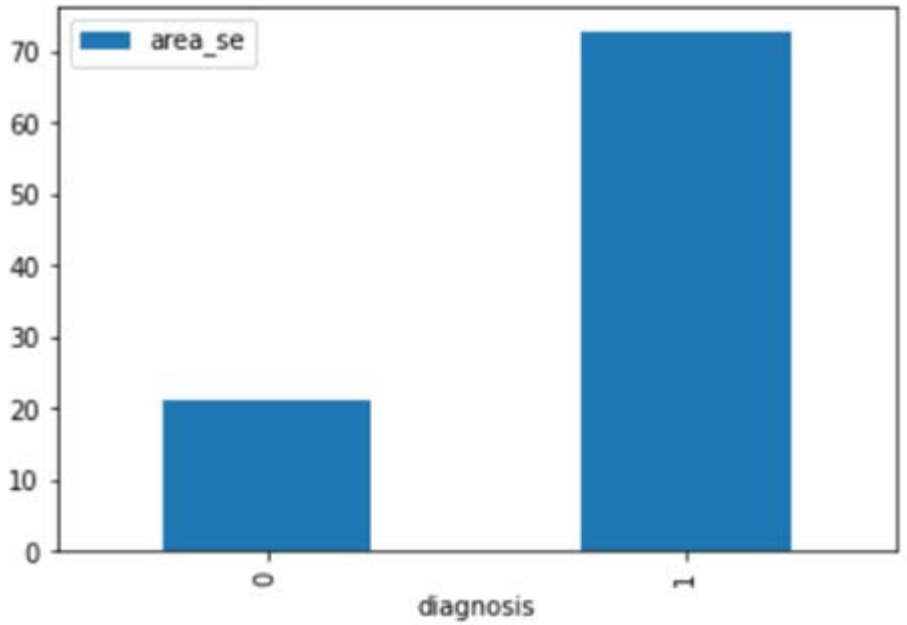
- Target : feature reduction을 통한 머신러닝의 성능 최적화

01 CONTENTS

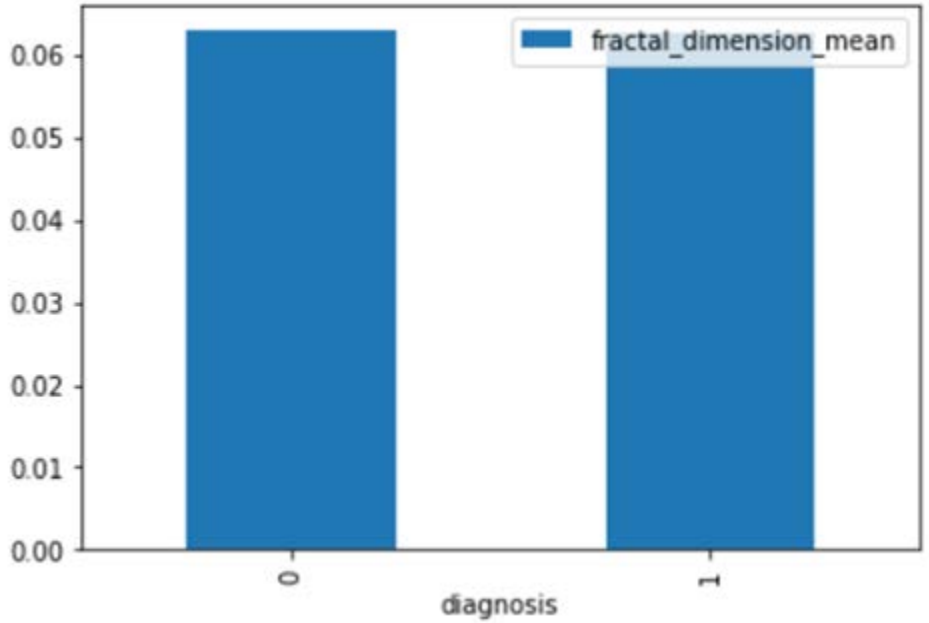
Mean by target

- 데이터의 target과 각 feature의 평균값을 구하였다.

Best fit



Worst fit

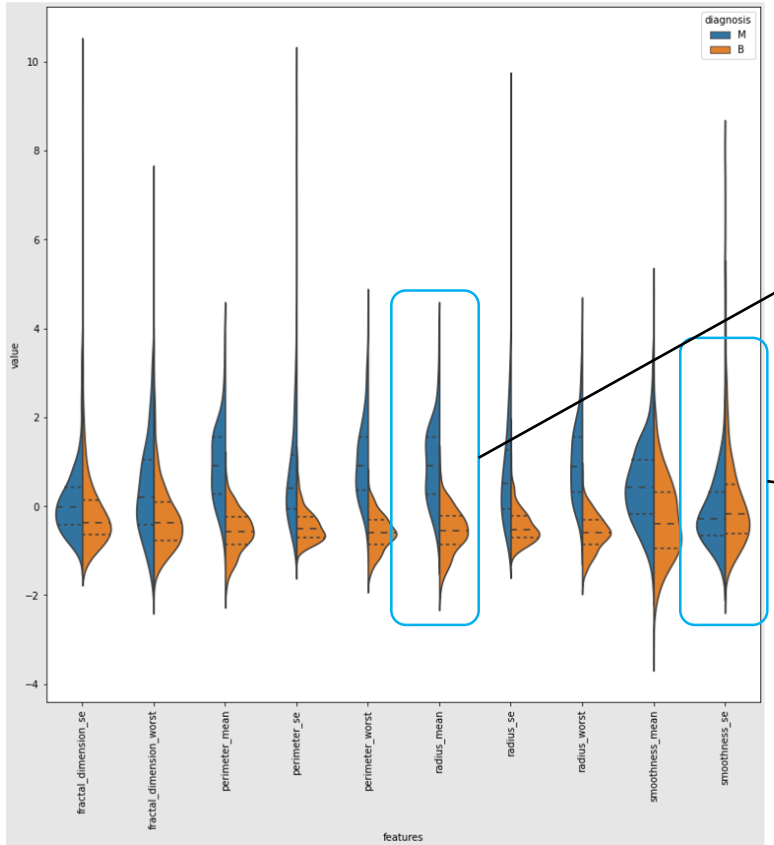


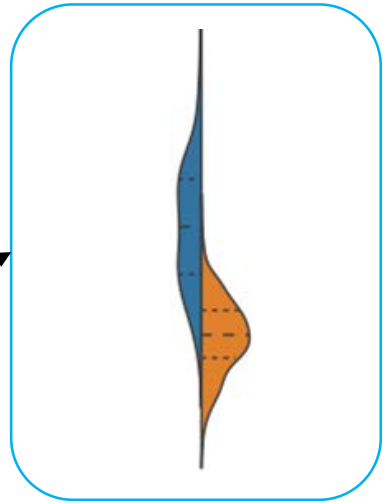
- 데이터의 label과 각 feature의 평균값이 비슷할수록 학습에 방해가 될 수 있다 -> feature 삭제

01 CONTENTS

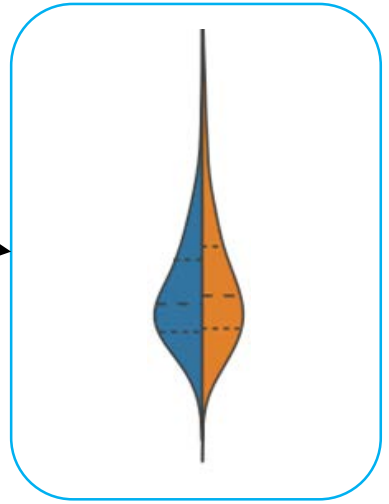
Distribution by target

- 데이터의 target에 따른 각 feature 값의 분포를 구하였다.





Best fit



Worst fit

01 CONTENTS

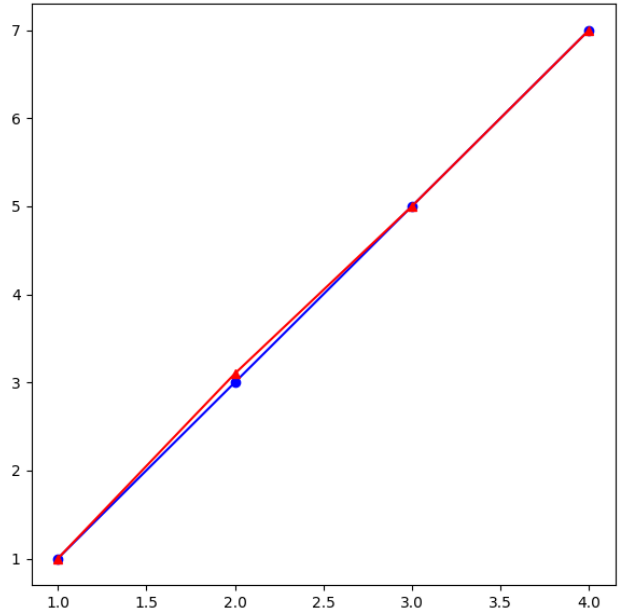
Why distribution important?

```python
import matplotlib.pyplot as plt

feature_1 = [1, 3, 5, 7]
feature_2 = [1, 3.1, 5, 7]

label = [1, 2, 3, 4]

f, ax = plt.subplots(figsize=(7,7))
ax.plot(label, feature_1, 'bo-')
ax.plot(label, feature_2, 'r^-')
plt.show()
```

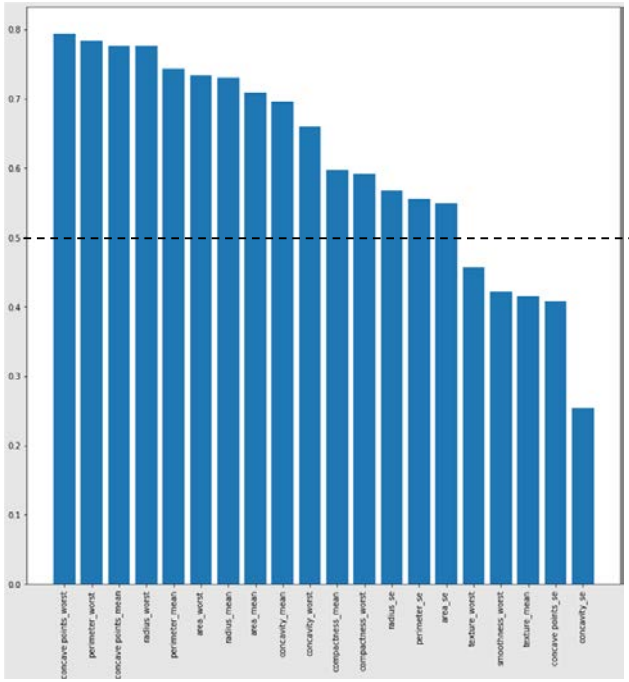


- 데이터의 feature의 값 분포가 비슷하니 그어진 선이 비슷할 수 있음을 알 수 있다 -> 따라서 삭제가 필요하다.

01 CONTENTS

Correlation with target

- 데이터의 target과 각 feature간의 correlation을 구하였다.
- Correlation이 낮다는 것은 비교 대상의 관계가 크지 않다고 볼 수 있다.

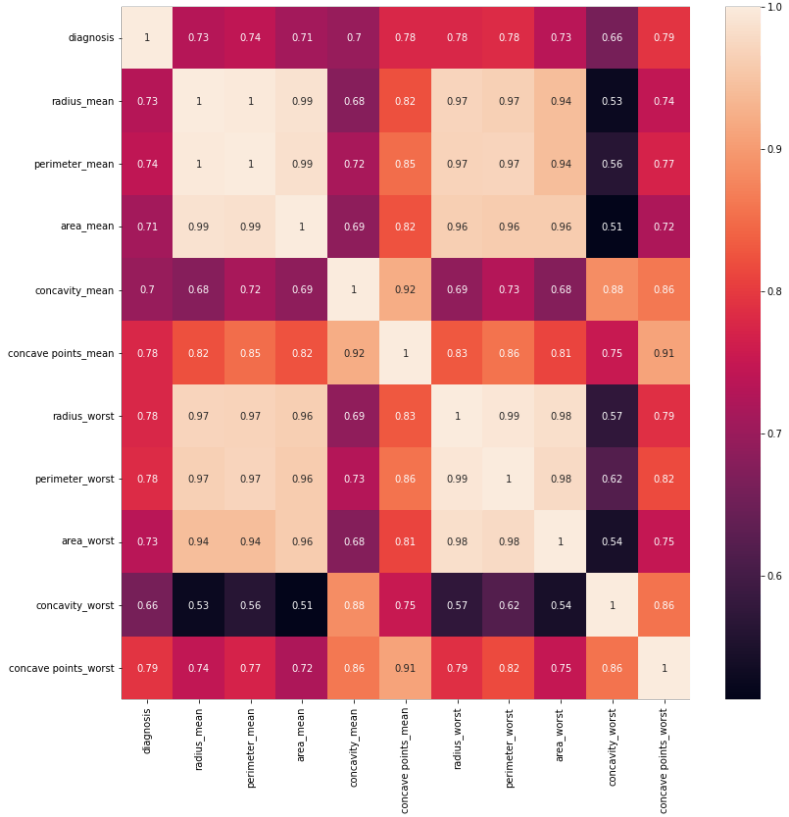


Correlation이 0.5 내외인 feature은
학습에 방해가 될지도 모르니 삭제하겠다.

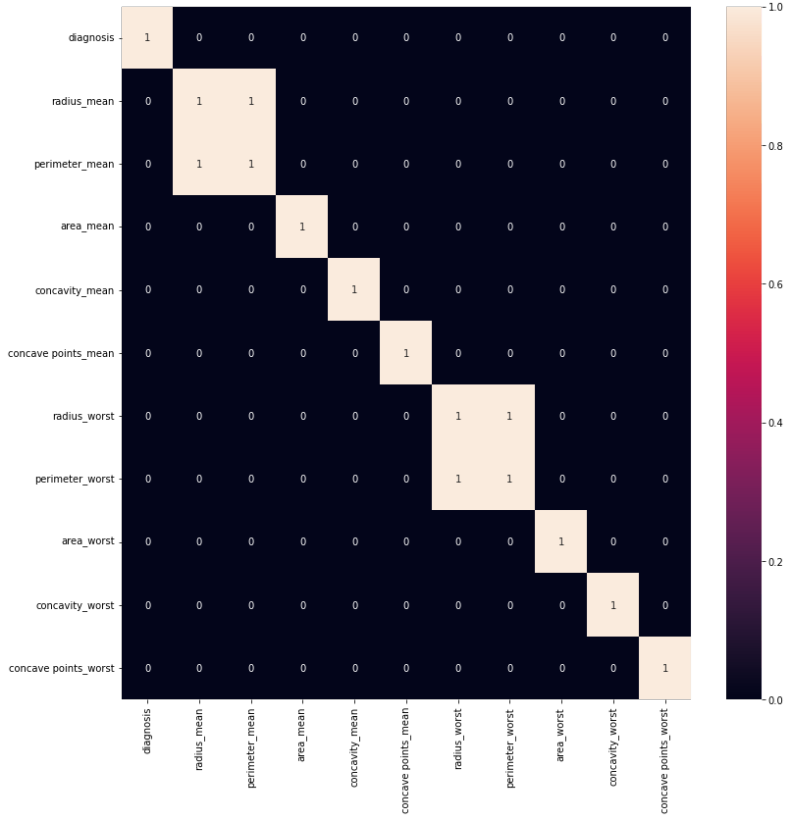
01 CONTENTS

Draw heatmap

- 전체 데이터의 상관관계를 heatmap으로 나타내었다.



Correlation > 0.99



01 CONTENTS

How to create new feature with original data

```
->creat new feature
data['radXper'] = data['radius_mean'] * data['perimeter_mean']
data['radXper']

->creat new feature
data['radXperW'] = data['radius_worst'] * data['perimeter_worst']
data['radXperW']

->drop used data
data.drop(columns=['radius_mean', 'perimeter_mean', 'radius_worst', 'perimeter_worst'], inplace=True)
```

- 데이터의 correlation이 1에 가깝다는 것은 두 feature의 관계가 두텁다는 것이다.
  - ➢ Correlation이 비슷한 데이터 끼리 곱 연산을 하여 새 feature을 창조한다.

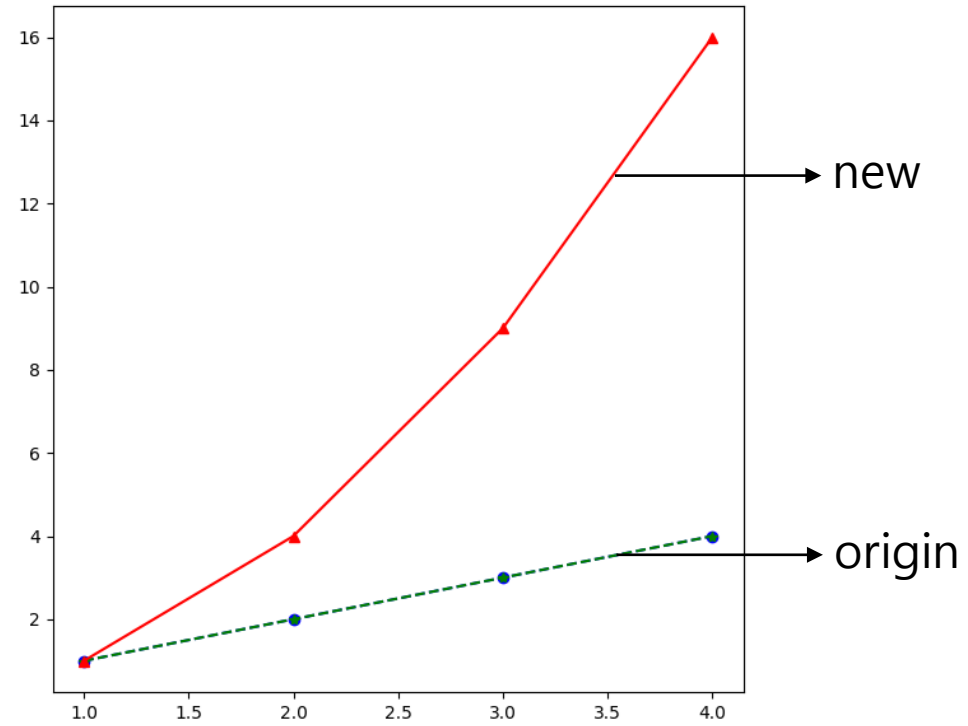
01 CONTENTS

Is new feature useful?

```python
import matplotlib.pyplot as plt

origin1 = [1, 2, 3, 4]
origin2 = [1, 2, 3, 4]
new = [1*1, 2*2, 3*3, 4*4]

f, ax = plt.subplots(figsize=(7,7))
ax.plot([1,2,3,4], origin1, 'bo--')
ax.plot([1,2,3,4], origin2, 'g*--')
ax.plot([1,2,3,4], new, 'r^-')
plt.show()
```



- 새 데이터의 x축이 증가할수록 기존의 데이터에비해 큰 폭으로 차이가 남을 알 수 있다

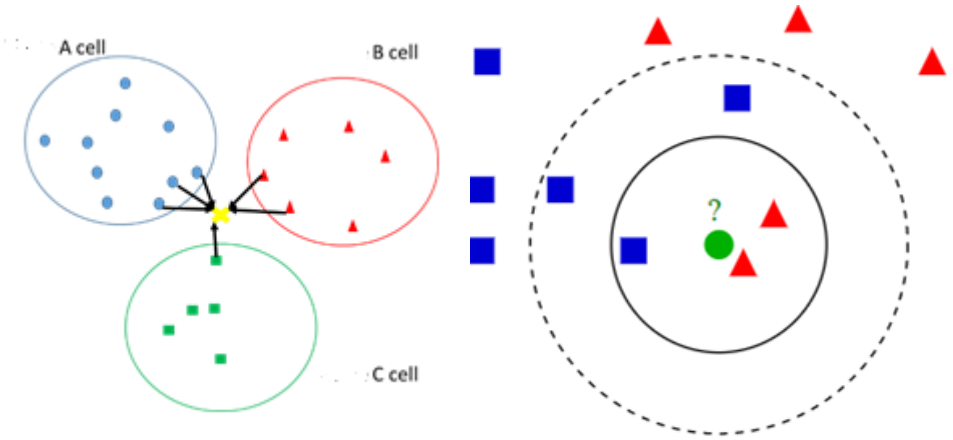
-> 분류에 도움이 될 수 있다.

01 CONTENTS

Machine Learning

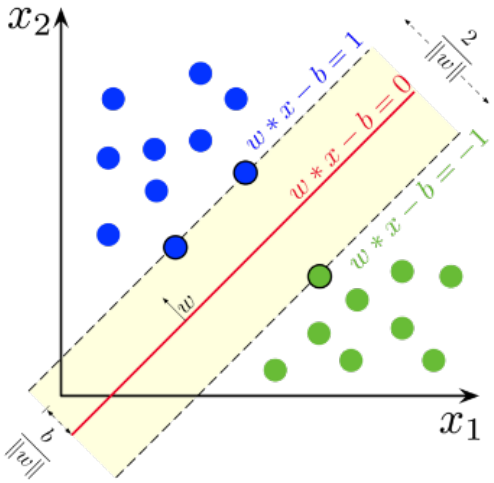
- 머신러닝의 KNN과 SVM을 이용하여 데이터를 분류하려고 한다.

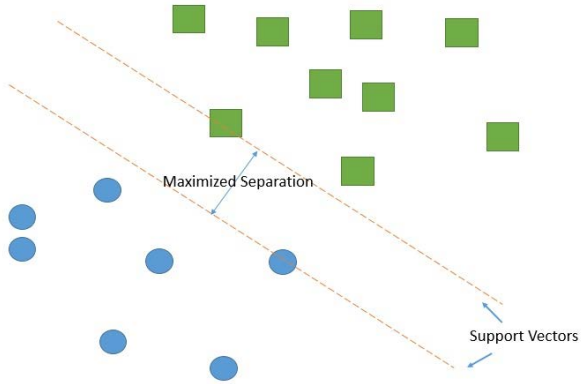
K - Nearest Neighbor



출처: Wikimedia commons

Support Vector Machine





출처: Wikimedia commons

01 CONTENTS

Score

```
KNN
knn = KNeighborsClassifier()
knn.fit(X_train, y_train)
knn.score(X_test, y_test)
>> 0.9440559440559441

SVM
svmc = SVC(kernel='linear')
svmc.fit(X_train, y_train)
svmc.score(X_test, y_test)
>> 0.951048951048951
```

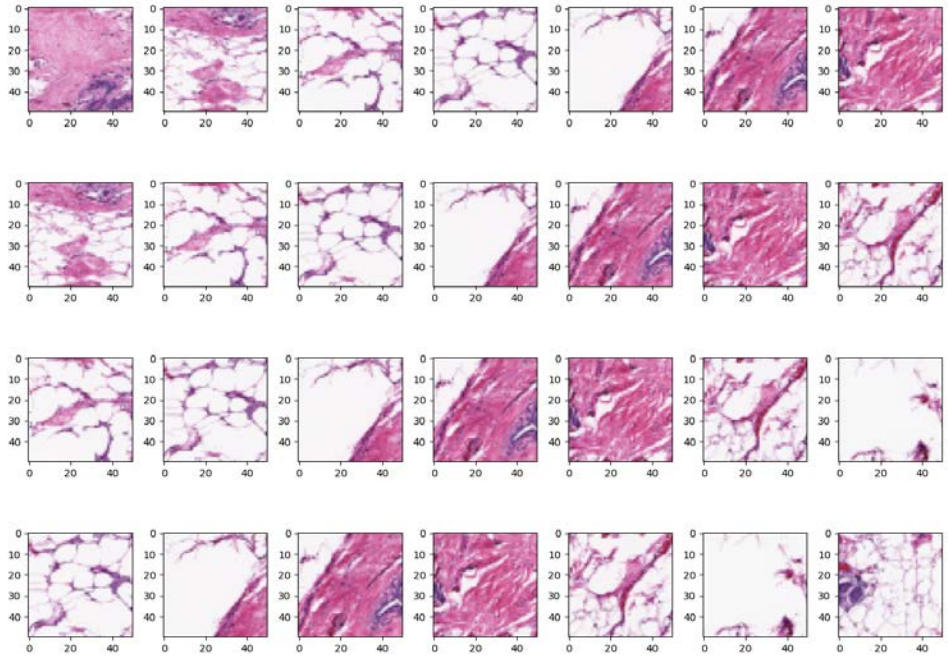
- KNN과 SVM을 사용한 결과 약 94%, 95%의 score을 달성하게 되었다.
  - ➢ 따라서 위에서 진행한 process가 적절했다고 생각할 수 있다.
- 예측에 사용된 feature을 돌아보면 모두 크기와 형태에 관련된 것만 남았음을 알 수 있다.
  - ➢ 따라서 도메인 지식이 없어도 암을 판별하는 기준을 예상 할수있다.

02

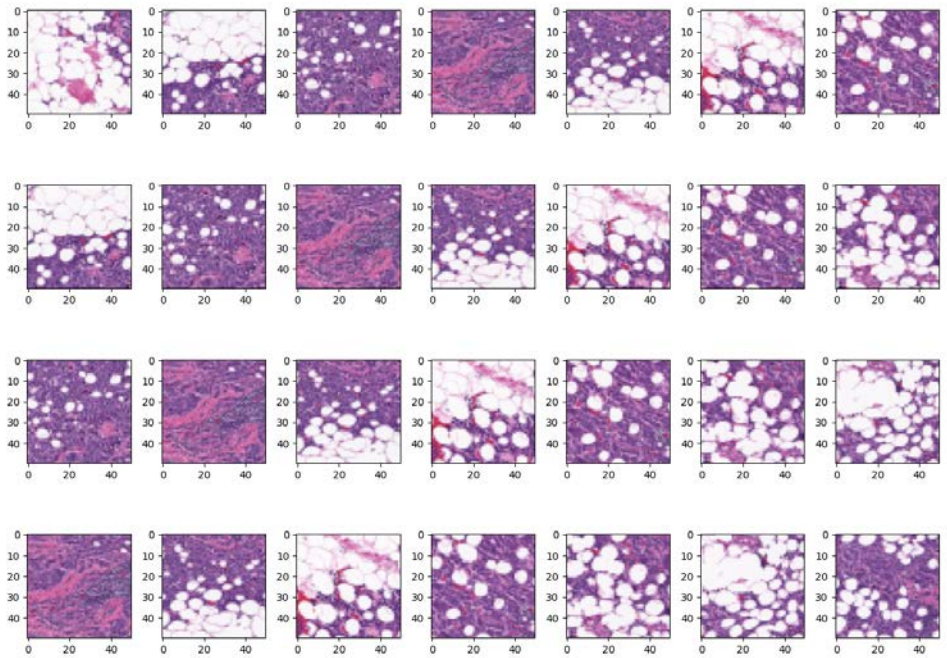
Breast Cancer diagnosis with DL

02 CONTENTS

Inspection & Target



- Normal Image

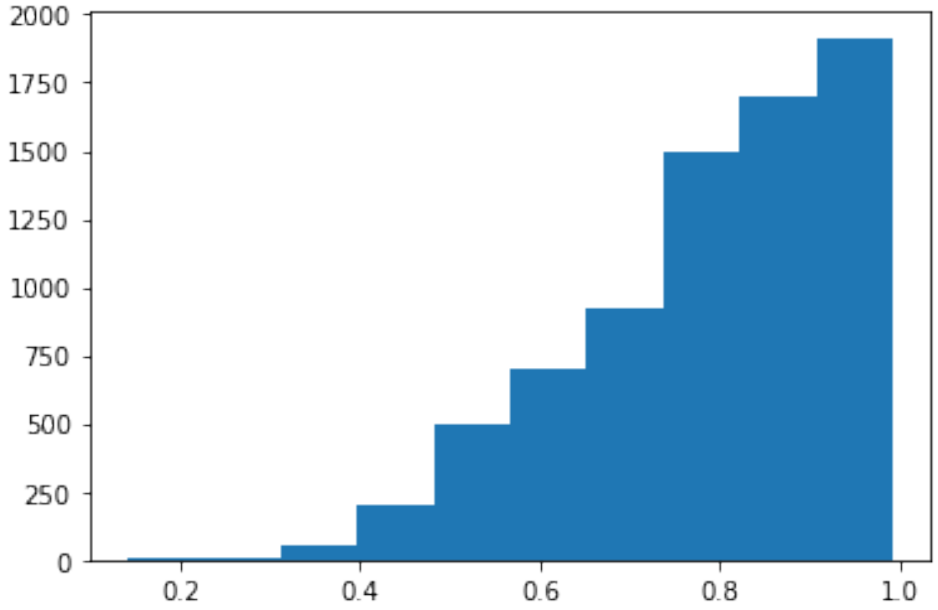


- Cancer Image

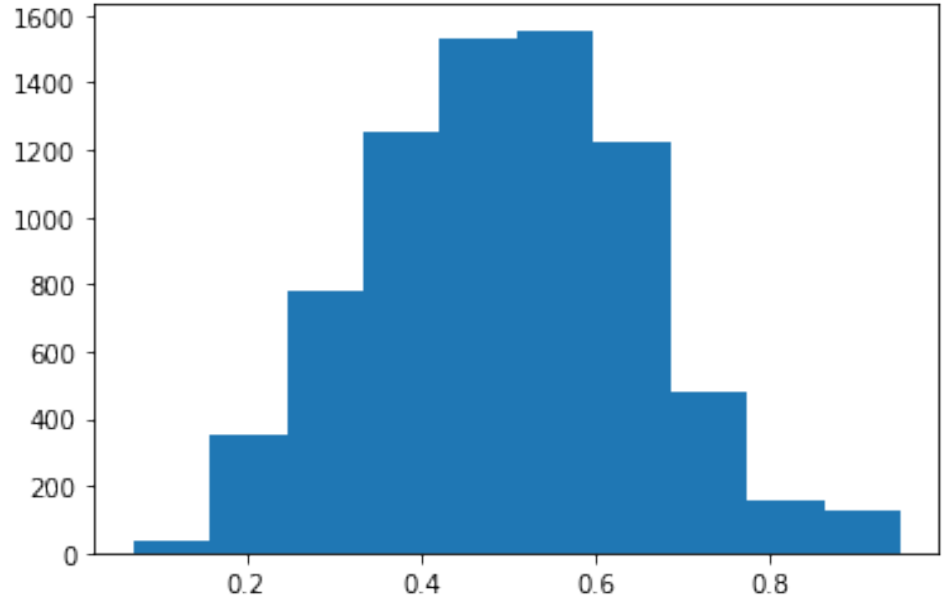
- Target : 딥러닝을 이용하여 암 환자 진단하기

02 CONTENTS

Brightness



- Normal Image

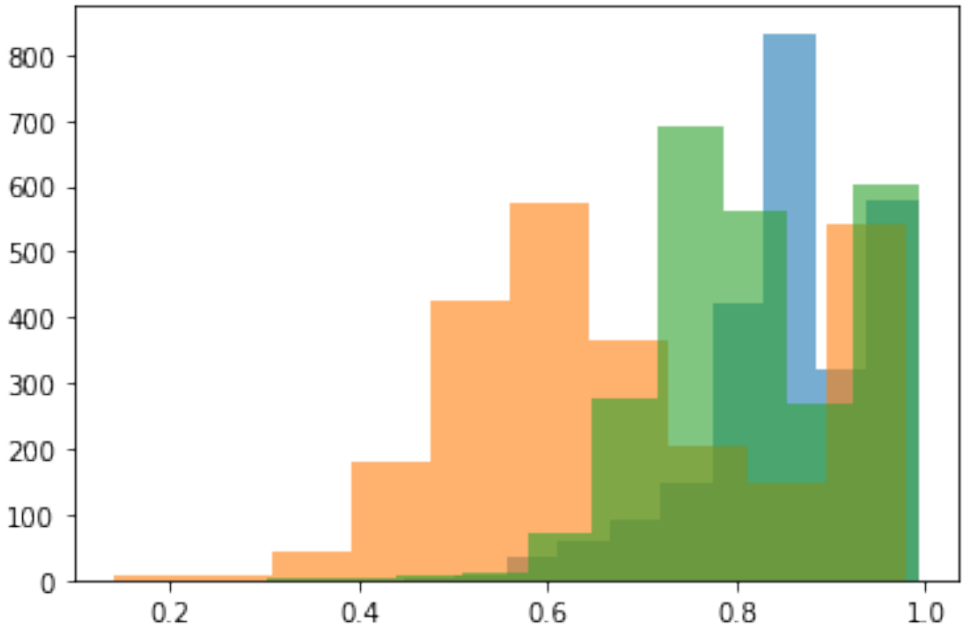


- Cancer Image

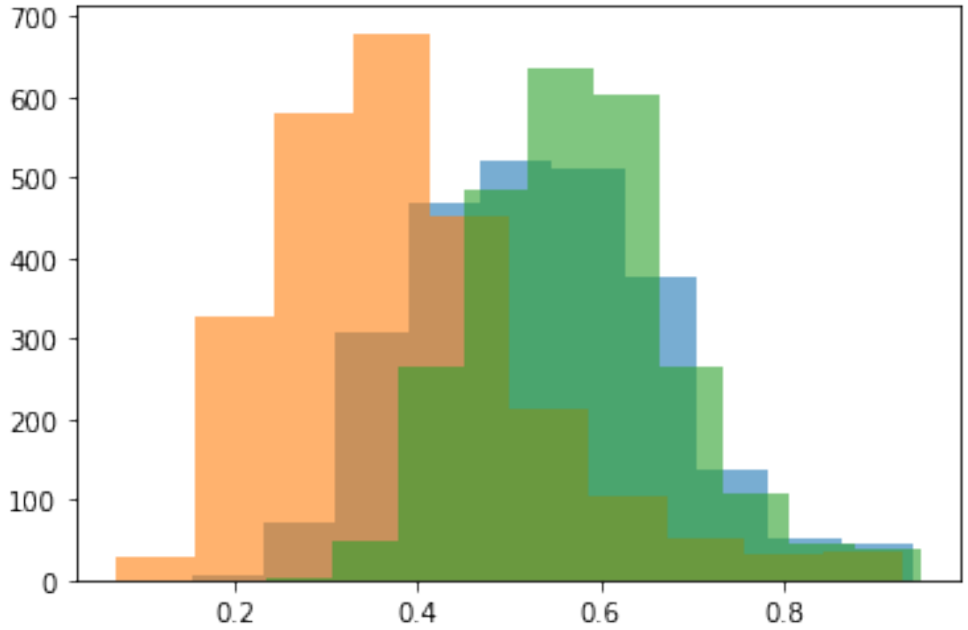
- 암 환자의 세포가 정상인에 비해 어두운 부분이 많다는 것을 알 수 있다.

02 CONTENTS

Density



- Normal Image

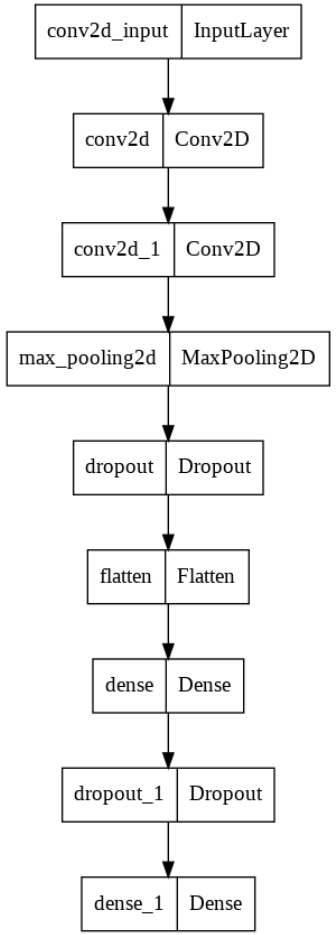


- Cancer Image

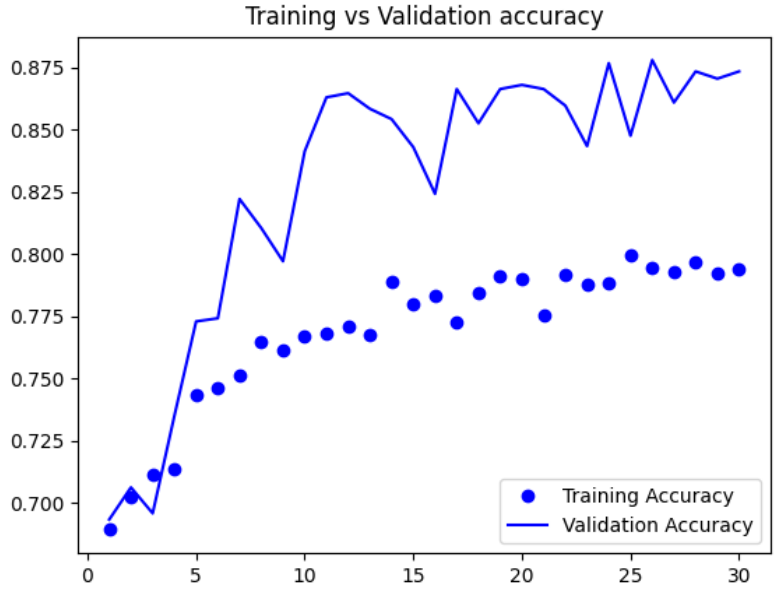
- 암 환자의 세포가 정상인에 비해 세포가 밀접하게 분포한 것을 알 수 있다.

02 CONTENTS

Model : CNN

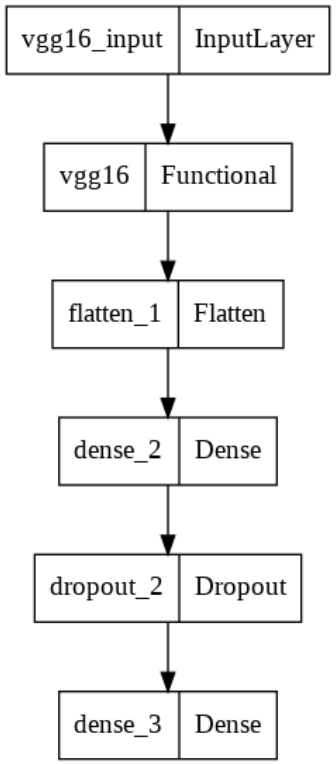



```
model = models.Sequential()
model.add(layers.Conv2D(32, 3, activation='relu', input_shape=(50,50,3)))
model.add(layers.Conv2D(64, 3, activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Dropout(0.25))
model.add(layers.Flatten())
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='sigmoid'))
model.compile(loss=tf.keras.losses.BinaryCrossentropy(),
              optimizer=tf.keras.optimizers.RMSprop(learning_rate=1e-5),
              metrics=['accuracy'])
```

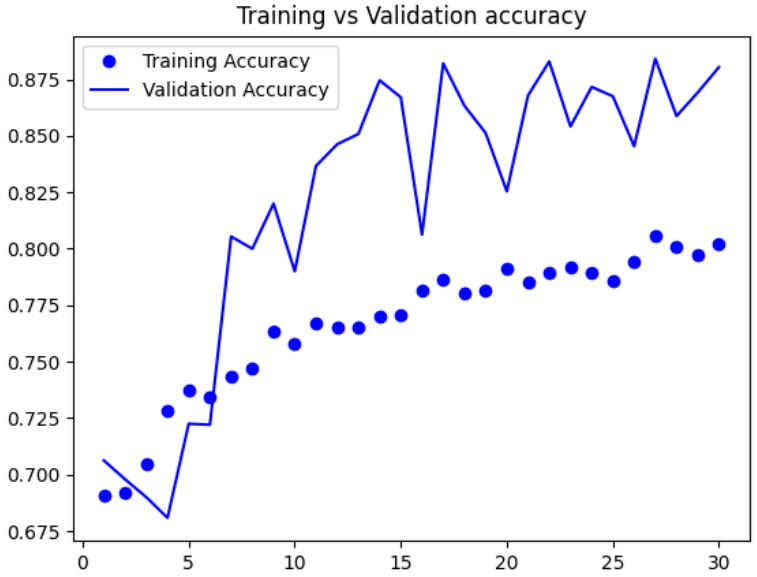


02 CONTENTS

Model : VGG16

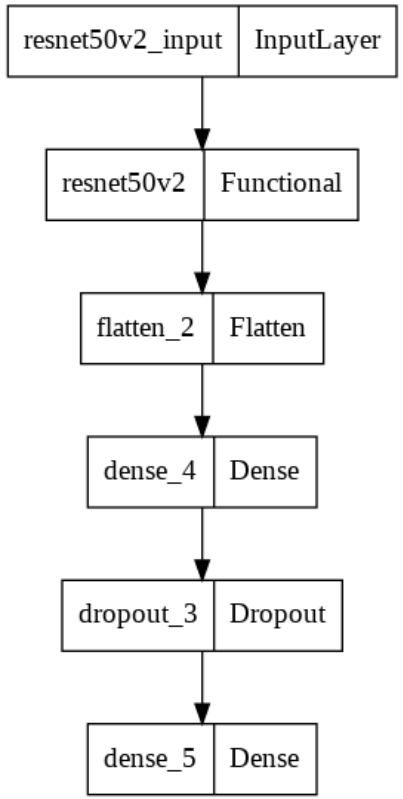



```
model = models.Sequential()
model.add(VGG16(weights='imagenet', include_top=False, input_shape=(50,50,3)))
model.add(layers.Flatten())
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='sigmoid'))
model.compile(loss=tf.keras.losses.BinaryCrossentropy(),
              optimizer=tf.keras.optimizers.RMSprop(learning_rate=1e-5),
              metrics=['accuracy'])
```

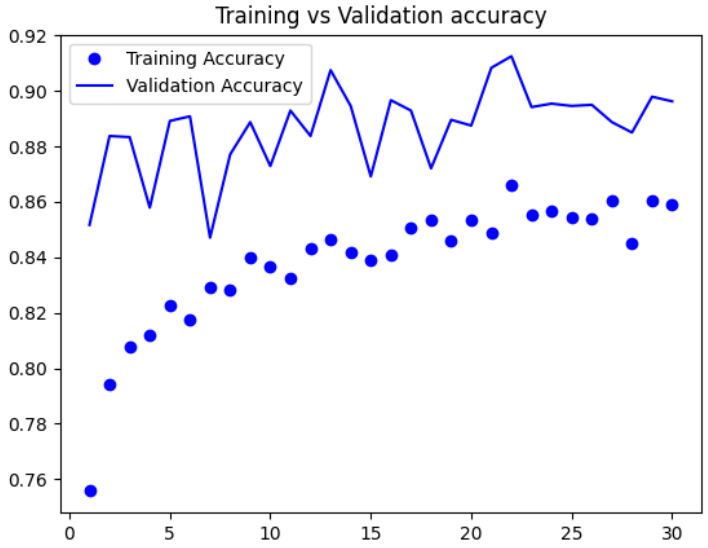


02 CONTENTS

Model : ResNet




```python
model = models.Sequential()
model.add(ResNet50V2(weights='imagenet', include_top=False, input_shape=(50,50,3)))
model.add(layers.Flatten())
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='sigmoid'))
model.compile(loss=tf.keras.losses.BinaryCrossentropy(),
              optimizer=tf.keras.optimizers.RMSprop(learning_rate=1e-5),
              metrics=['accuracy'])
```



03

Analysis

03 CONTENTS

Diagnosis Function

```python
def diagnosis(data, image):
  # 데이터 전처리
  data = pd.DataFrame(data).T
  data.drop(columns=['id', 'fractal_dimension_mean', 'texture_se', 'symmetry_se', 'Unnamed: 32',
                     'compactness_se', 'fractal_dimension_se', 'fractal_dimension_worst', 'smoothness_mean',
                         'smoothness_se', 'symmetry_mean', 'symmetry_worst','compactness_mean',
                     'compactness_worst', 'radius_se', 'perimeter_se', 'area_se', 'texture_worst', 'smoothness_worst',
                         'texture_mean', 'concave points_se', 'concavity_se'], inplace=True)
  data['radXper'] = data['radius_mean'] * data['perimeter_mean']
  data['radXperW'] = data['radius_worst'] * data['perimeter_worst']
  data.drop(columns=['radius_mean', 'perimeter_mean', 'radius_worst', 'perimeter_worst'], inplace=True)

  # 모델 로드
  svm_load = joblib.load('/content/drive/MyDrive/svm.pkl')
  resnet_load = models.load_model('/content/drive/MyDrive/ResNetTrain.h5')

  # 예측
  data_pred = svm_load.predict(data.T.to_numpy()[1::].reshape(-1, 8))[0]
  image_pred = resnet_load.predict(image.reshape(-1, 50, 50, 3))[0][0]

  if not data_pred:
    print(f'유방암일 확률은 {int(image_pred*100)}% 입니다.' if image_pred > 0.5 else '의사의 확인이 필요합니다.')
  else:
    print(f'유방암일 확률은 {int(image_pred*100)}% 입니다.' if image_pred <= 0.5 else '의사의 확인이 필요합니다.')
```

```
->정상인 진단
diagnosis(data_none, img_none)
>> 유방암일 확률은 1% 입니다.

->암환자 진단
diagnosis(data_cancer, img_cancer)
>> 유방암일 확률은 98% 입니다.

->정상인과 암환자의 데이터를 섞었을 때
diagnosis(data_none, img_cancer)
>> 의사의 확인이 필요합니다.
```

- 실제로 바로 사용 가능한 함수를 구성하였다. 데이터에 따른 예측도 오류 없이 도출되었다.

03 CONTENTS

Limit of Model

```python
base = ResNet50V2(weights='imagenet', include_top=False, input_shape=(50,50,3))

base.trainable = True
set_train = False

for layer in base.layers:
    if layer.name == 'conv5_block3_3_conv':
        set_train = True
        if set_train:
            layer.trainable = True
        else:
            layer.trainable = False

model = models.Sequential()
model.add(base)
model.add(layers.Flatten())
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='sigmoid'))
model.compile(loss=tf.keras.losses.BinaryCrossentropy(),
              optimizer=tf.keras.optimizers.RMSprop(learning_rate=1e-5),
              metrics=['accuracy'])
```

- Imagenet의 모델은 동결 해제시킨 학습이나 일반 학습이나 큰 차이가 없었다.

```
knn.predict(test_data)
>> breast cancer

svm.predict(test_data)
>> breast cancer

model.predict(test_data)
>> breast cancer
```

- 모든 모델이 92% 내외로 예측을 하고 있다.

03 CONTENTS

Limit & Apply

Limit

1. 적은 데이터의 양

2. validation accuracy > accuracy
   ->underfitting이나 validation data가 예측하기 쉬운 구성일 수 있음.

3. Accuracy를 높이려면 앙상블이나, 최적화된 layer 구현 필요 예상.

Apply

1. 데이터를 입력하면 진단결과를 출력해주는 웹사이트 제작

2. 의료현장의 유방암 진단 보조기구로 사용

Thank You