Data Analyst Portfolio

최서윤

Data Analyst

최서윤

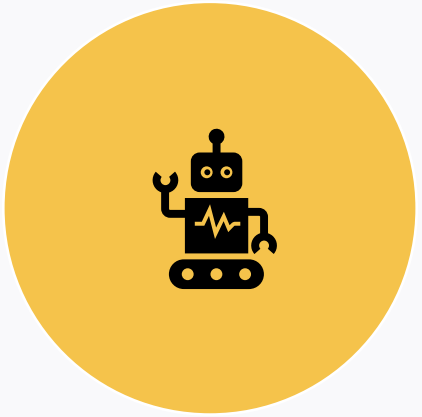
Email : choisawyou@gmail.com

Phone : +82 10 7161 7297

Index

실무 데이터 분석 프로젝트

- 제주도 내 인기장소 찾기
- 제주도 방문 관광객 이동경로 찾기



머신러닝 & 딥러닝 프로젝트

- 출산율 예측
- 텍스트와 이미지를 통한 감정 분류
- 재활용품 분류

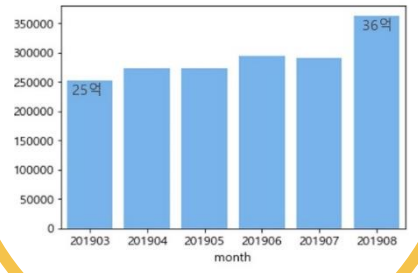
데이터 분석가 최서윤

Hot Places in Jeju

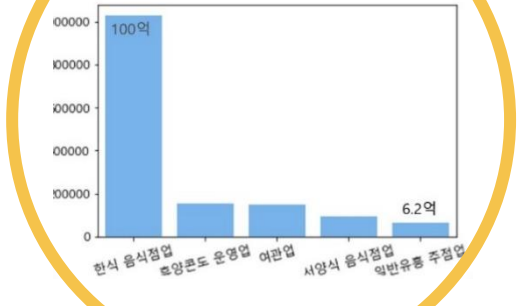
신용카드 데이터

신용카드 사용명세 데이터 + 소셜 데이터를 이용하여 제주도 내 인기장소 도출 프로젝트

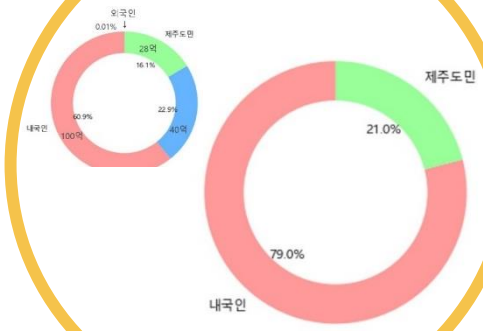
EDA



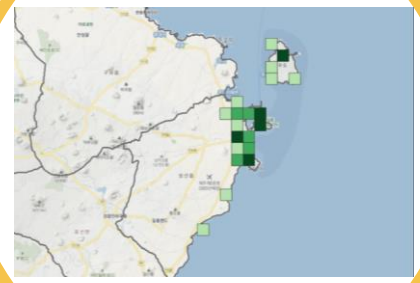
EDA



Pre-processing



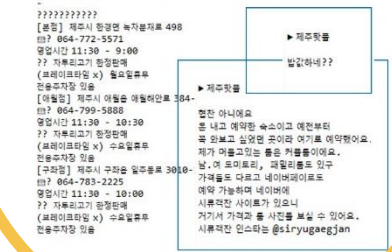
QGIS



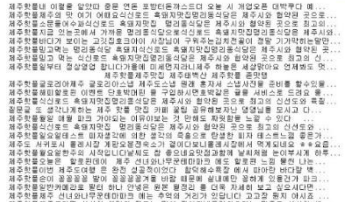
소셜 데이터

네이버 블로그&카페,인스타그램 크롤링

Crawling

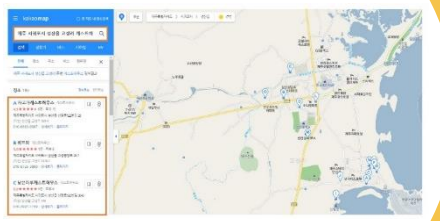


Pre-processing



Word cloud

Kakao Map API



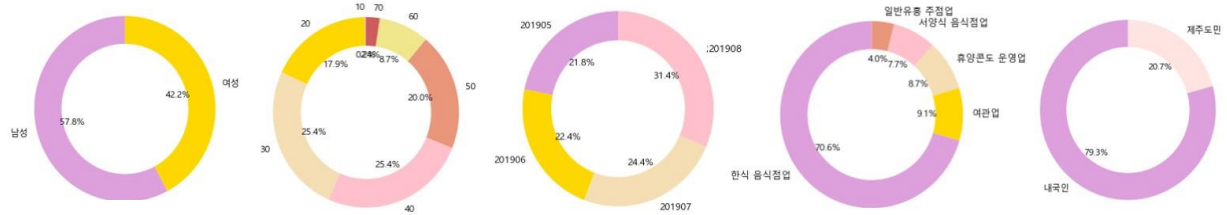
Hot Places in Jeju

우도

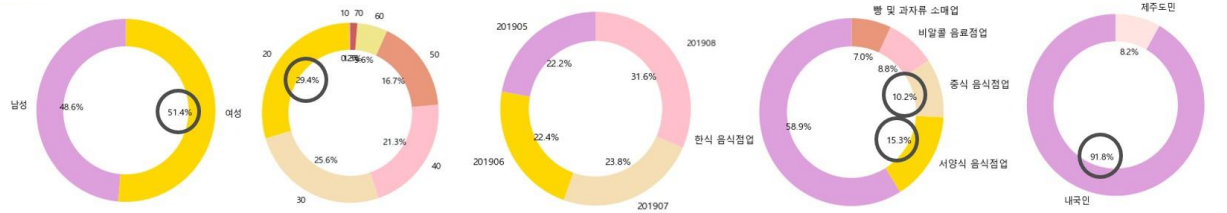
20대·여성·내국인
서양·중국 음식

- 하하호호 우도본점
- 섬소나이
- 우돈
- 우도해녀식당

전체지역



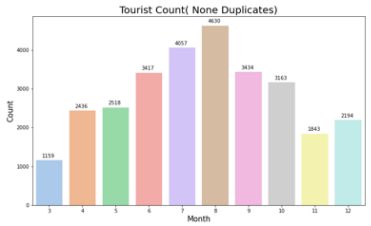
우도



Tourist route in Jeju

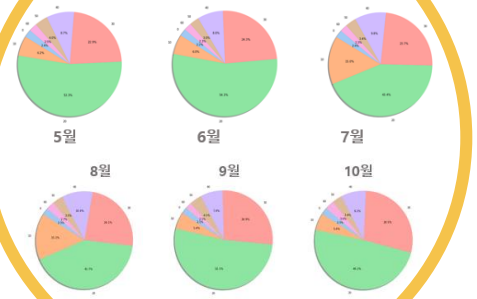
WIFI 접속 데이터를 이용하여 제주도내 중국인 관광객 이동 패턴 도출 프로젝트

EDA



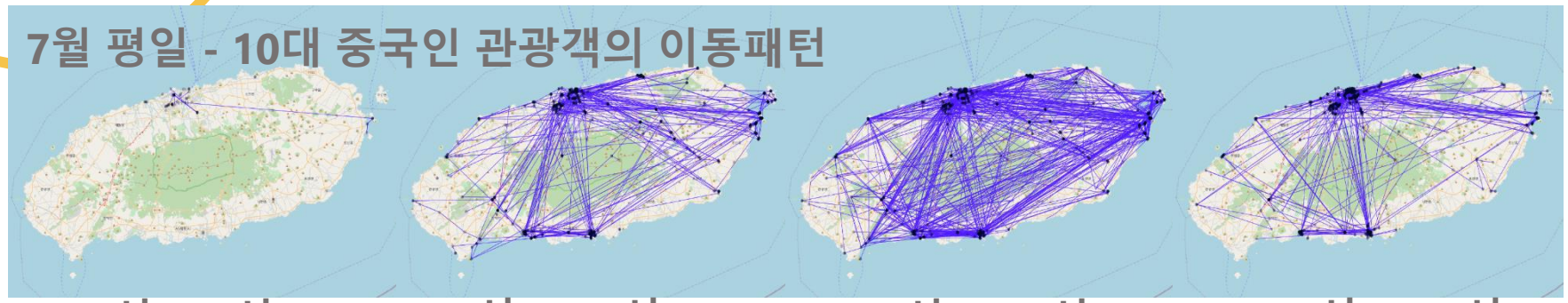
5716867 rows

EDA



WIFI 접속 이력 데이터

7월 평일 - 10대 중국인 관광객의 이동패턴



0시 ~ 6시 | 6시 ~ 12시 | 12시 ~ 18시 | 18시 ~ 24시

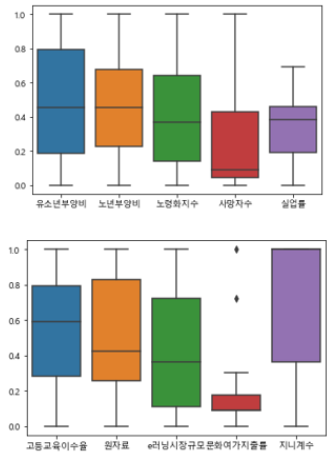
7월 주말 - 10대 중국인 관광객의 이동패턴

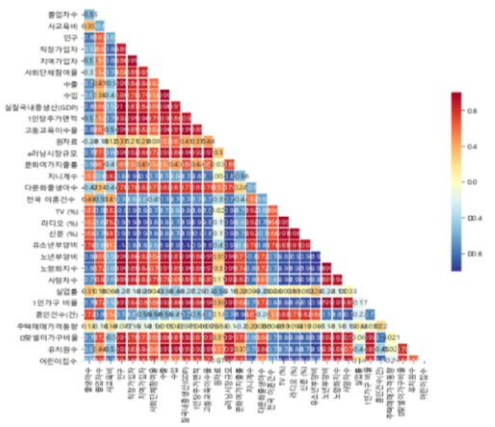
0시 ~ 6시 | 6시 ~ 12시 | 12시 ~ 18시 | 18시 ~ 24시

Birth-rate Regression

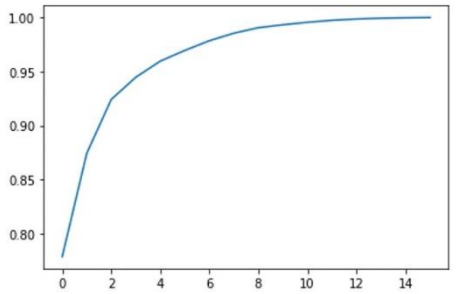
출산율 예측 프로젝트

EDA

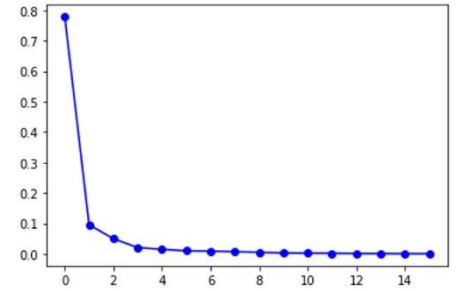




PCA 차원축소



전처리 된 변수들의 조합으로 이루어진 새로운 파생변수 생성이 모델에 도움이 될 것으로 판단하여 pca를 이용하여 주성분 생성



영향력 있는 주성분만을 변수로 사용하여 모델링 진행

Modeling

	Linear	* SVR	D.T	R.T	XGB
Train MSE	0.19581	0.011588	0.0	0.00379	9.4991
Test MSE	0.23230	0.009546	0.01739	0.00666	0.0151
상관계수	0.95	0.96	0.94	0.95	0.93

Grid Search CV

*최적의 파라메터 조합

0.988474

SVR (C=0.1, cache_size=200, coef0=0.0, degree=3,
Epsilon=0.1, gamma=1, kernel=' rbf ', max_iter=-1, shrinking=True,
Tol=0.001, verbose=False)

결과 및 한계점

* Random Forest Importance 를 이용한 주요 변수 선택

*유소년부양비

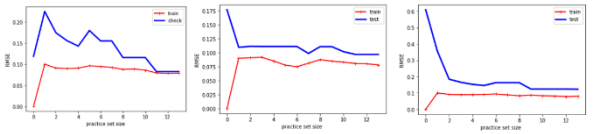
*교육 이수율

*노령화지수

*1인 가구

*문화 여가 지출

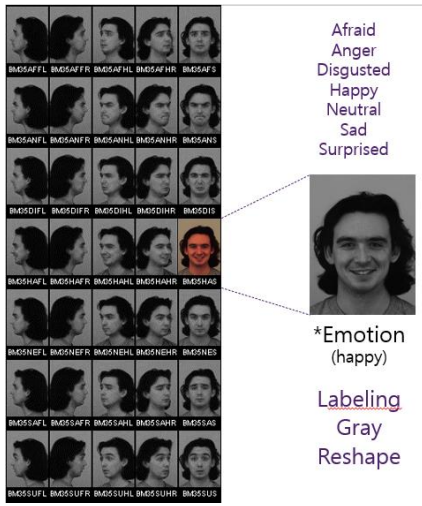
과소적합:



Emotion Classification

텍스트와 이미지를 통한 감정분류 프로젝트

Image pre-treatment



DCNN

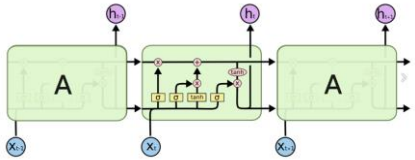
```
with K.tf_ops.device('/device:GPU:0'):
    model5 = Sequential()

    model5.add(Conv2D(32, (3,3), padding="same", input_shape=X_train.shape[1:], activation='relu'))
    model5.add(MaxPooling2D(pool_size=(2,2)))
    model5.add(Dropout(0.25))

    model5.add(Conv2D(64, (3,3), padding="same", activation='relu'))
    model5.add(MaxPooling2D(pool_size=(2,2)))
    model5.add(Dropout(0.25))

    model5.add(Conv2D(128, (3,3)
    model5.add(MaxPooling2D(pool
    model5.add(Dropout(0.25))

    model5.add(Conv2D(256, (3,3)
    model5.add(MaxPooling2D(pool
    model5.add(Dropout(0.25))

    model5.add(Conv2D(512, (3,3)
    model5.add(MaxPooling2D(pool
    model5.add(Dropout(0.25))

    model5.add(Flatten())  # fla
    model5.add(Dense(256, activa
    model5.add(Dropout(0.5))
    model5.add(Dense(nb_classes,
    model5.compile(loss='categorical_cros

    model_dir = 'C:/Users/ICT01_09/Docume

    if not os.path.exists(model_dir):
        os.mkdir(model_dir)

    model_path5 = model_dir + '/multi_img
    checkpoint = ModelCheckpoint(filepath
    early_stopping = EarlyStopping(monito
```

ACC 35.5%

```
model.evaluate(X_test, y_test)
11882/11882 [==============================] - 7s 618us/step
[1.6016808201058348, 0.35499075055122375]
print("정확도 : %.4f" % (model.evaluate(X_test, y_test)[1]))
print("정확도 : %.4f" % (model.evaluate(X_test, y_test)[1]))
11882/11882 [==============================] - 7s 613us/step
정확도 : 0.3550
```

Text Emotion Classification

KoNLPy



1400	함께하다	1
1401	헤어지다	3
1402	끼어들다	0
1403	반하다	1
1404	사과하다	3
1405	위로하다	1
1406	화해하다	1
1407	방해하다	0
1408	되지 않아 불편하다	0
1409	재밌다 보고 싶다	1
1410	안쓰럽다	3
1411	감동적이다	1
1412	감사하다	1
1413	든든하다	1
1414	만족스럽다	1
1415	반갑다	1
1416	사랑스럽다	1
1417	설레다	1

VGG16

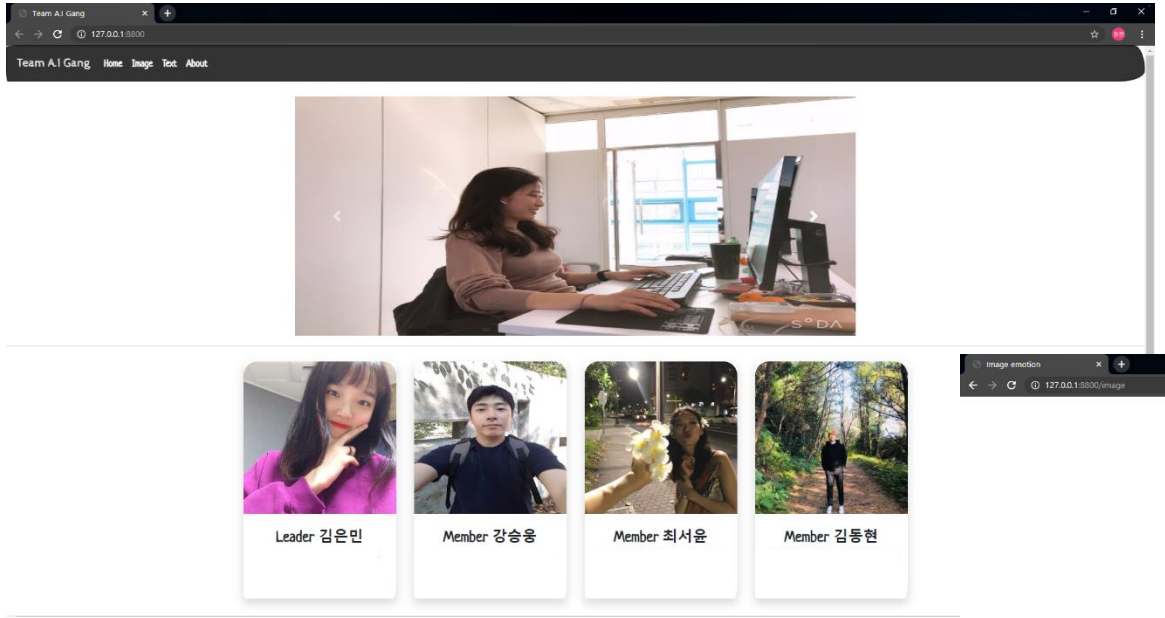
```
model =Sequential()
#1
model.add(Conv2D(256,(3,3), padding ='same',input_shape =(img_row,img_col,channel),activation = 'relu'))
model.add(Conv2D(64,(3,3),activation ='relu',padding='same'))
model.add(AveragePooling2D(2,2))

#2

model.add(Conv2D(128,(3,3),activation ='relu',padding='same'))
model.add(Conv2D(128,(3,3),activation ='relu',padding='same'))
model.add(AveragePooling2D(2,2))

#3
model.add(Conv2D(256,(3,3),activation ='relu',padding='same'))
model.add(Conv2D(256,(3,3),activation ='relu',padding='same'))
model.add(Conv2D(256,(3,3),activation ='relu',padding='same'))
model.add(AveragePooling2D(2,2))


#4

model.add(Conv2D(512,(3,3),activation ='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation ='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation ='relu',padding='same'))
model.add(AveragePooling2D(2,2))


#5

model.add(Conv2D(512,(3,3),activation ='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation ='relu',padding='same'))
model.add(Conv2D(512,(3,3),activation ='relu',padding='same'))
model.add(AveragePooling2D(2,2))


#Flatten
model.add(layers.Flatten())
```

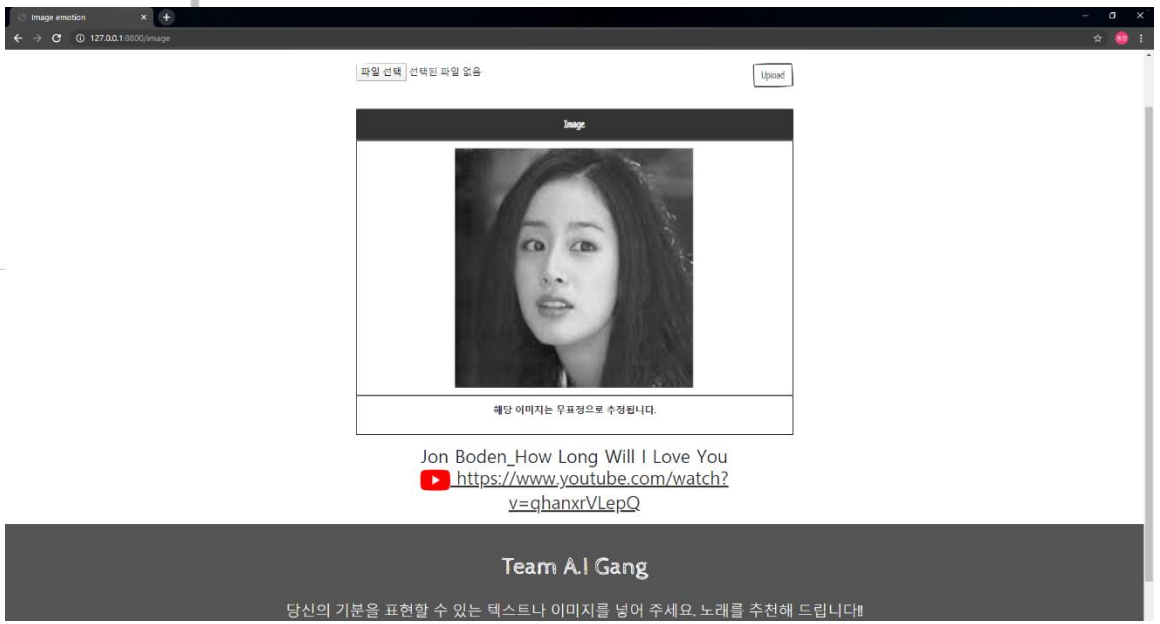
ACC : 79.93%
VAL_LOSS 57.69%

```
Epoch 00011: val_loss improved from 0.70415 to 0.57698, saving model to emotion_edit/model/vgg16avg.model
Epoch 12/32
22144/35324 [=================>............] - ETA: 1:04:14 - loss: 0.6133 - acc: 0.7993
```

Emotion Classification

Web 구축

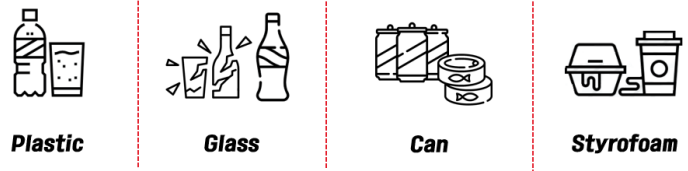




Recycle Classification

쓰레기 4종 분류 프로젝트

Pre-processing

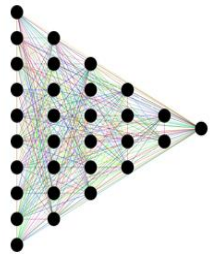


Selenium Crawling

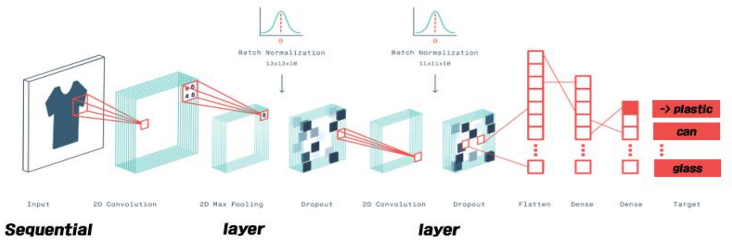
ImageDataGenerator & reshape

Modeling

Various Shapes Problem



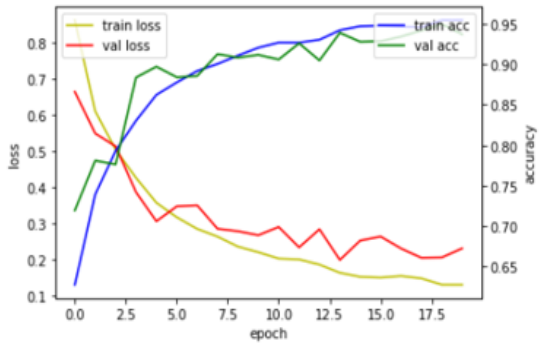
Feature extraction



CNN

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 64, 64, 32)        896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 32, 32, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 32, 32, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 16, 16, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 16, 16, 64)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 16384)             0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               4194560
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 4)                 1028
=================================================================
Total params: 4,214,980
Trainable params: 4,214,980
Non-trainable params: 0
_________________________________________________________________
```

결과




```
========] - 8s 810us/step

47266]

print("정확도 : %.4f" % (model.evaluate(X_test, y_test)[1]))

10000/10000 [==============================] - 8s 770us/step
정확도 : 0.9396
```

Thank you