기계학습 텀프로젝트 4

19011494 조국희

04. 산업환경상의 이상 상황 검출

- **프로젝트 목적**

  2D 영상 데이터 내 이상 발생 여부를 판단할 수 있음

- **이상 상황 데이터**
  - ✓ 정상 데이터와 다른 분포를 가진 데이터
  - ✓ MVTech AD 데이터 셋을 이용한 비지도 학습 기반의 이상 상황 검출
  - ✓ 이를 위해 PCA(차원 축소), SVM 분류기 사용

04. 산업환경상의 이상 상황 검출

- **Reconstruction based anomaly detection**
  - ✓ PCA로 이뤄진 encoder와 decoder를 이용
  - ✓ 정상 영상으로 구성된 train data의 주성분을 추출하여 차원 축소(feature 추출)
  - ✓ 정상/이상 영상이 포함된 test data에 train data에서 적용된 PCA로 feature f_test 추출(Encoding)
  - ✓ feature f_test를 복원(Decoding)
  - ✓ **원본 영상과 복원 영상의 차를 구함-> 이상-복원 차이↑, 정상-복원 차이↓**

```python
# 원하는 분산 비율 설정
n_components = 0.8

# [1] 정규화
sc = StandardScaler()
# train data 는 학습 및 변환하고 val data는 변환
train_sd = sc.fit_transform(train['imgs'])
val_sd = sc.transform(val['imgs'])

# [2] 차원 축소(주성분 추출)
pca = PCA(n_components=n_components, random_state=777)
pca.fit(train_sd)

# [3] train data에서 학습한 것을 바탕으로 val 주성분 추출
val_pca = pca.transform(val_sd)
```

```python
# val data 주성분 추출 한것 다시 복원한후 정규화 한것도 복원
# [num_of_img, width, height]
val_inverse = pca.inverse_transform(val_pca)
val_inverse = sc.inverse_transform(val_inverse)
Reconstruction_imgs = val_inverse.reshape(-1, 86, 86)

# [5]
# ori = Original imgs ~ reshape [n_sample, w, h]
# Reconstruction_error = Original imgs - Reconstruction_imgs
ori = val['imgs'].reshape(-1, 86, 86)
Reconstruction_error= ori - Reconstruction_imgs
```

bottle ROC_AUC : 0.514

04. 산업환경상의 이상 상황 검출

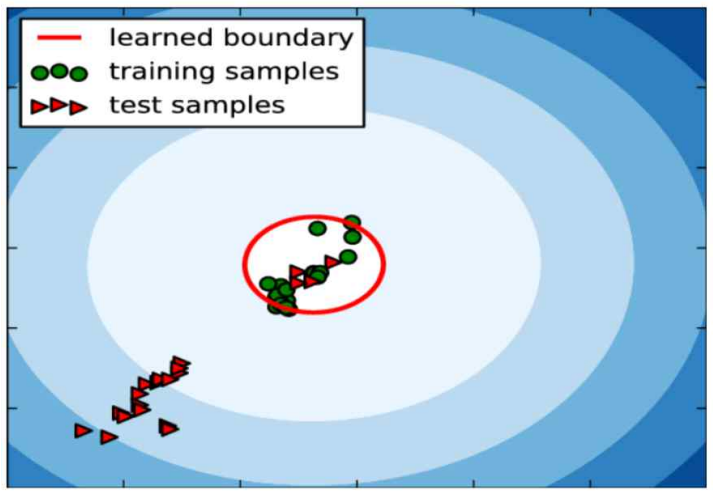
- **Embedding feature based anomaly detection**
  - ✓ 정상 영상의 특성을 학습한 feature extractor를 이용해 정상/이상 영상의 feature 추출
  - ✓ 정상 영상의 feature 분포 경계를 학습해 이에 벗어나는 경우를 이상치로 검출

이미지 사이즈, 축소할 차원의 수 설정

```
size = (86, 86)
n_components= 180
```

OneClassSVM

주어진 데이터를 잘 설명할 수 있는 최적의 support vector를 구하고 이 영역 밖의 데이터들은 outlier로 간주하는 방식




```
# [0] 모델 OneClassSVM 선언, PCA 초기화
clf = OneClassSVM(kernel="rbf", gamma=0.001, nu=0.01)
pca = PCA(n_components = n_components, random_state = 777)
```

04. 산업환경상의 이상 상황 검출

- **Embedding feature based anomaly detection**

정규화

```
# train data 는 학습 및 변환하고 val data는 변환
scaler = StandardScaler()
train_sd = scaler.fit_transform(train['imgs'])
val_sd = scaler.transform(val['imgs'])
```

train data로 주성분 학습한 것을 바탕으로 train, val 주성분 추출

```
pca.fit(train_sd)
train_pca = pca.transform(train_sd)
val_pca = pca.transform(val_sd)
```

추출된 주성분으로 모델 학습 및 예측

```
clf.fit(train_pca)
y_pred = clf.predict(val_pca)
cls_score_val = clf.score_samples(val_pca)
```

test.imgs도 동일한 과정을 적용
단, 추론 시 score에 대해서만 계산

```
scaler = StandardScaler()
train_sd = scaler.fit_transform(train['imgs'])
test_sd = scaler.transform(test['imgs'])

pca.fit(train_sd)
train_pca = pca.transform(train_sd)
test_pca = pca.transform(test_sd)

clf.fit(train_pca)
cls_score= clf.score_samples(test_pca)
```

03. 음성 감정 인식

- **성능을 높이기 위한 시도**
  - ✓ 축소하는 차원의 수를 조절(n_component)
  - ✓ 이미지의 사이즈 조절
  - ✓ 분류기의 하이퍼 파라미터 조절
  - ✓ 여러가지 PCA 방법론 사용

베이스라인	성능
PCA-OC-SVM	0.54373
RandomizedPCA-OC-SVM	0.67165

모델	성능
PCA180-OC-SVM, size(86,86)	0.57430
KernelPCA200-OC-SVM, size(86,86)	0.58998
RandomizedPCA180-OC-SVM, size(86,86)	0.71094
RandomizedPCA200-OC-SVM, size(86,86)	0.71695
RandomizedPCA200-OC-SVM, size(70,70)	0.72860
RandomizedPCA180-OC-SVM, size(60,60)	0.73728