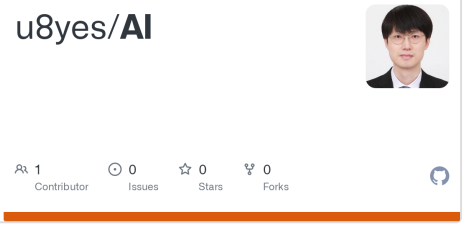
Day49; 20221115

날짜	@2022년 11월 15일
유형	@2022년 11월 15일
태그	

GitHub - u8yes/AI

You can't perform that action at this time. You signed in with another tab or window. You signed out in another tab or window. Reload to refresh your session. Reload to refresh your session.

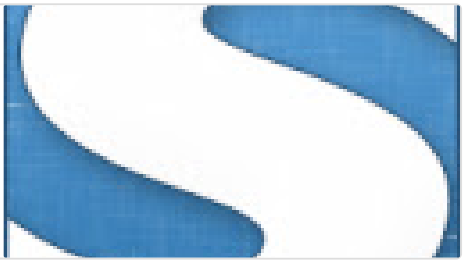
https://github.com/u8yes/ai



Take Webpage Screenshots Entirely - FireShot

Take FULL webpage screenshots. Capture, edit and save them to PDF/JPEG/GIF/PNG, upload, print, send to OneNote, clipboard or email.

https://chrome.google.com/webstore/detail/take-webpage-screenshots/mcbpblocgmgfnpjjppndjkmgjaogfceg/related



https://s3-us-west-2.amazonaws.com/secure.notion-static.com/3814fb60-3a4a-4100-b1ea-cc1fa1204d4d/02_%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D_%EC%95%8C%EA%B3%A0%EB%A6%AC%EC%A6%98.pdf

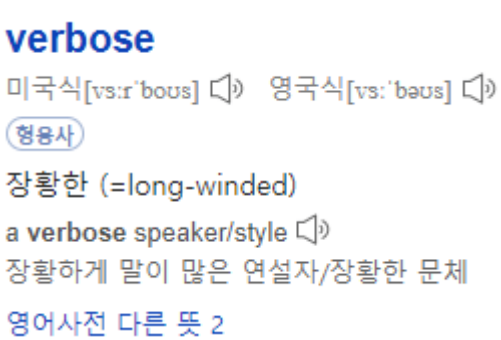
---

빅분기 실기 - 머신러닝)

▾ 2-05.GridSearchCV 사용

GridSearchCV : 최고의 hyperparameter를 찾기 위해 사용함

- sklearn.model_selection.GridSearchCV
- (estimator, param_grid, , *scoring=None, n_jobs=None, refit=True, cv=None, verbose=0, pre_dispatch='2n_jobs', error_score=nan,* return_train_score=False)
- estimator : 학습 모델
- param_grid : 실행해볼 hyperparameter 목록, dict 객체
- cv : CrossValication에 사용할 나누는 개수, 기본값 = 5
- verbose : 0(default) : 메시지 출력 안함, 1 : 간단한 메시지, 2 : 하이퍼 파라미터별 메시지 출력
- n_jobs : -1로 지정시 사용시 모든 코어를 다 사용, 속도가 빨라짐




```
# [13] 최고 점수가 나오는 모델 가져오기, 성능평가
model = gs.best_params_
```

[56]
```
# [13] 최고 점수가 나오는 모델 가져오기, 성능평가
model = gs.best_estimator_
print(model.score(x_test, y_test), gs.score(x_test, y_test), gs.best_params_)
```

```
0.9666666666666667 0.9666666666666667 {'n_neighbors': 5}
```

시각화_20221115

- 최적의 파라미터일때의 결정경계선과 다른 파라미터들일 때의 결정경계선을 비교

:
```
# 시각화를 하기 위해, 최적의 C와 다른 C를 후보로 저장한다.

C_candidaties = []
C_candidaties.append(clf.best_params_['C']* 0.01) # clf - classification
C_candidaties.append(clf.best_params_['C'])
C_candidaties.append(clf.best_params_['C']* 100) # Cost가 10일 때의 시각화

# 시각화를 하기 위해, 최적의 gamma와 다른 gamma를 후보로 저장한다.

gamma_candidaties = []
gamma_candidaties.append(clf.best_params_['gamma']* 0.01) # clf - classification
gamma_candidaties.append(clf.best_params_['gamma'])
gamma_candidaties.append(clf.best_params_['gamma']* 100) # gamma가 100일 때의 시각화
```

:
```
X = train[['3P','BLK']]
Y = train['Pos'].tolist() # 자료형을 tolist() - 리스트로 만듦

# 포지션에 해당하는 문자열 'SG'와 'C'를 벡터화한다.
position = []
for gt in Y:
    if gt == 'C':
        position.append(0)
    else:
        position.append(1)
```

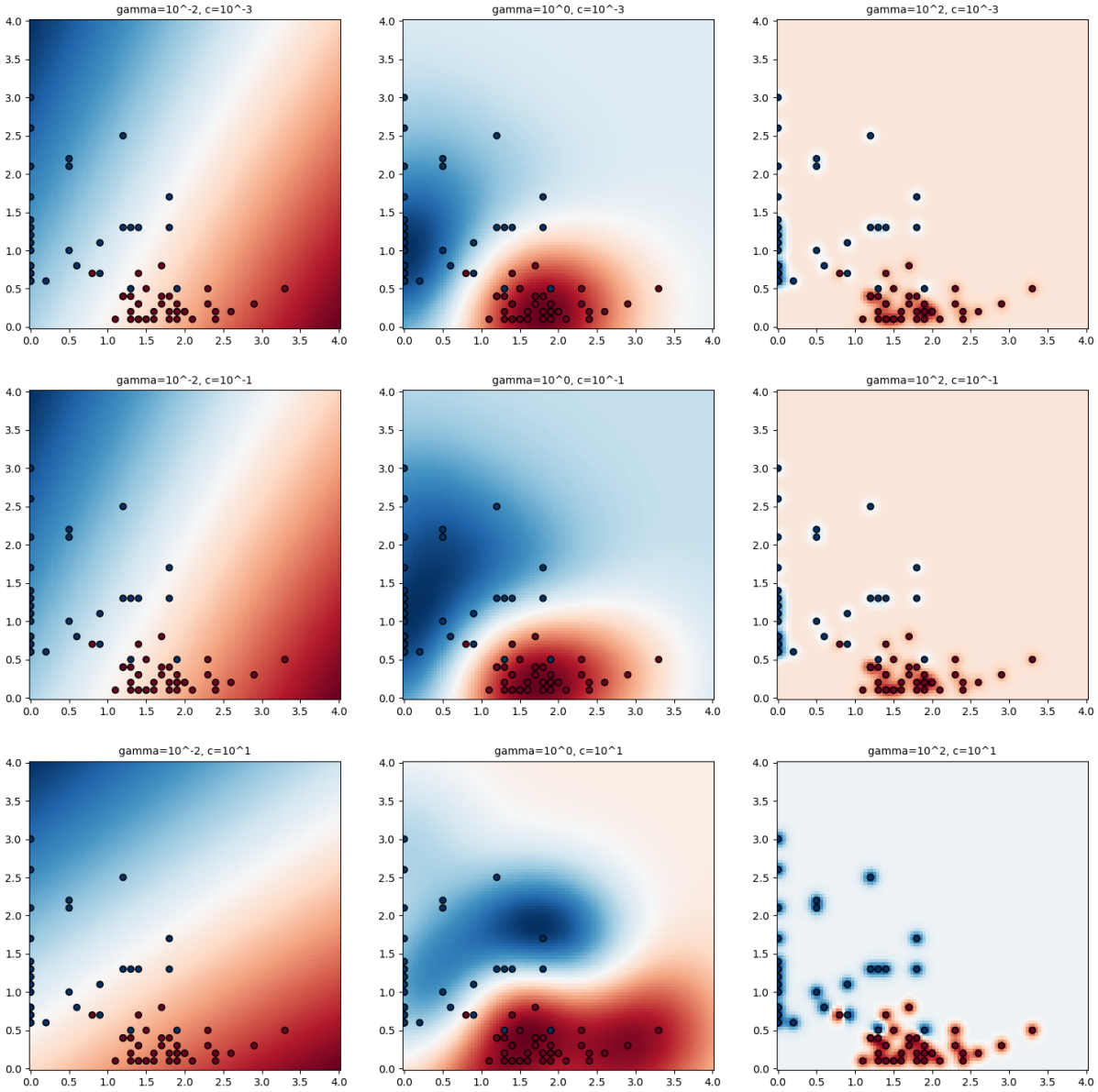
:
```
# 각각의 파라미터에 해당하는 SVM 모델을 만들어 classifiers에 저장한다.
classifiers = []
for C in C_candidaties:
    for gamma in gamma_candidaties:
        clf = SVC(C=C, gamma=gamma)
        clf.fit(X, Y)
        classifiers.append((C, gamma, clf))
```

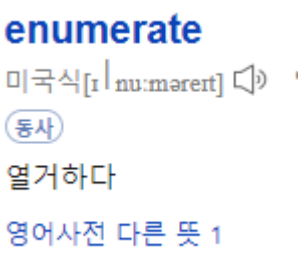
```
# 18, 18 사이즈의 차트를 구성 # 18inch 크기로 화면상에 시각화해줌.
plt.figure(figsize=(18,18))
xx, yy = np.meshgrid(np.linspace(0,4,100), np.linspace(0,4,100)) # np.linspace(0,4,100) 시작값, 끝값, 사이에 100개의 데이터 생성

# 각각의 모델들에 대한 결정 경계 함수를 적용하여 함께 시각화.
for(k,(C, gamma, clf)) in enumerate(classifiers): # 순수 시각화를 위한 위치를 잡아주고 있다.
    Z = clf.decision_function(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)

    # 최적의 모델을 포함한 다른 파라미터로 학습된 모델들을 함께 시각화
    plt.subplot(len(C_candidates), len(gamma_candidates), k + 1)
    plt.title("gamma=10^%d, c=10^%d" % (np.log10(gamma), np.log10(C)), size="medium")

    plt.pcolormesh(xx, yy, -Z, cmap=plt.cm.RdBu)
    plt.scatter(X['3P'], X['BLK'], c = position, cmap=plt.cm.RdBu_r, edgecolors='k')
    # cmap=plt.cm.RdBu_r 시각화를 위한 색상 셋팅(Red, Blue)
```






```
# 참조용)
# test_x, test_y = np.meshgrid([1,2,3],[1,2,3])
# print(test_x)
# print(test_y)
```

```
[[1 2 3]
 [1 2 3]
 [1 2 3]]
[[1 1 1]
 [2 2 2]
 [3 3 3]]
```

---

enumerate

```python
# 리스트
foods = ["치킨", "피자", "햄버거"]

# enumerate를 적용한 리스트 출력
print("enumerate를 적용한 리스트 :", list(enumerate(foods)))

# 인덱스에 해당하는 값을 순서대로 출력
for index, food in enumerate(foods):
    print("인덱스 : {}, 값 : {}".format(index, food))
```

```
enumerate를 적용한 리스트 : [(0, '치킨'), (1, '피자'), (2, '햄버거')]
인덱스 : 0, 값 : 치킨
인덱스 : 1, 값 : 피자
인덱스 : 2, 값 : 햄버거
```

시작 인덱스 변경

```python
# 리스트
foods = ["치킨", "피자", "햄버거"]

# 인덱스에 해당하는 값을 순서대로 출력
# - start : 시작 인덱스
for index, food in enumerate(foods, start=1):
    print("인덱스 : {}, 값 : {}".format(index, food))
```

```
인덱스 : 1, 값 : 치킨
인덱스 : 2, 값 : 피자
인덱스 : 3, 값 : 햄버거
```

테스트_20221115

- sklearn의 gridsearch로 얻어진 최적의 파라미터로 학습된 clf를 이용하여 테스트 진행.

```
]: # 테스트에 사용될 특징을 지정
X_test = test[['3P','BLK']]

# 특징으로 예측할 값(농구선수 포지션)을 지정.
y_test = test[['Pos']]

# 최적의 파라미터로 완성된 SVM에 테스트 데이터를 주입하여, 실제값과 예측값을 얻는다.
y_true, y_pred = y_test, clf.predict(X_test)

print(classification_report(y_true, y_pred)) # classification_report() - 분류 모델 평가 지표
print('accuracy : ' + str(accuracy_score(y_true, y_pred)))
```

```
              precision    recall  f1-score   support

           C       1.00      0.88      0.93         8
          SG       0.92      1.00      0.96        12

    accuracy                           0.95        20
   macro avg       0.96      0.94      0.95        20
weighted avg       0.95      0.95      0.95        20

accuracy : 0.95
```

```
comparision = pd.DataFrame({'prediction':y_pred, 'target':y_true.values.ravel()})
comparision
```

	prediction	target
0	SG	SG
1	SG	SG
2	C	C
3	SG	SG
4	SG	SG
5	SG	SG
6	C	C
7	SG	SG
8	SG	SG
9	C	C
10	SG	C
11	SG	SG
12	SG	SG
13	SG	SG
14	SG	SG
15	C	C
16	C	C
17	C	C
18	SG	SG
19	C	C

서포트 벡터머신 – 장/단점

➢ 장점

- 커널 트릭을 사용함으로써 특성이 다양한 데이터를 분류하는 데 강하다.
    - N개의 특성을 가진 데이터는 N차원 공간의 데이터 포인트로 표현되고, N차원 공간 또는 그 이상의 공간에서 초평면을 찾아 데이터 분류가 가능하기 때문.
- 예측 속도가 빠르다.
- 파라미터(C, gamma)를 조정해서 과대적합 및 과소적합에 대처할 수 있다.
- 적은 학습 데이터로도 딥러닝만큼 정확도가 높은 분류를 기대할 수 있다.

➢ 단점

- 데이터 전처리 과정(data preprocessing)이 상당히 중요하다.
    - 특성이 비슷한 수치로 구성된 데이터 같은 경우에는 SVM을 쉽게 활용할 수 있지만, 특성이 다양하거나 혹은 확연히 다른 경우에는 데이터 전처리 과정을 통해 데이터 특성 그대로 벡터 공간에 표현해야 함.
- 특성이 많을 경우 결정 경계 및 데이터의 시각화가 어려움 – SVM의 분류 결과를 이해하기 힘듦.
- 커널 트릭 오사용시 과대적합되기 쉽다.

matplotlib 시각화는 특성이 다양할수록 다차원으로 표현하기가 어려워짐.

의사결정트리(Decision Tree)

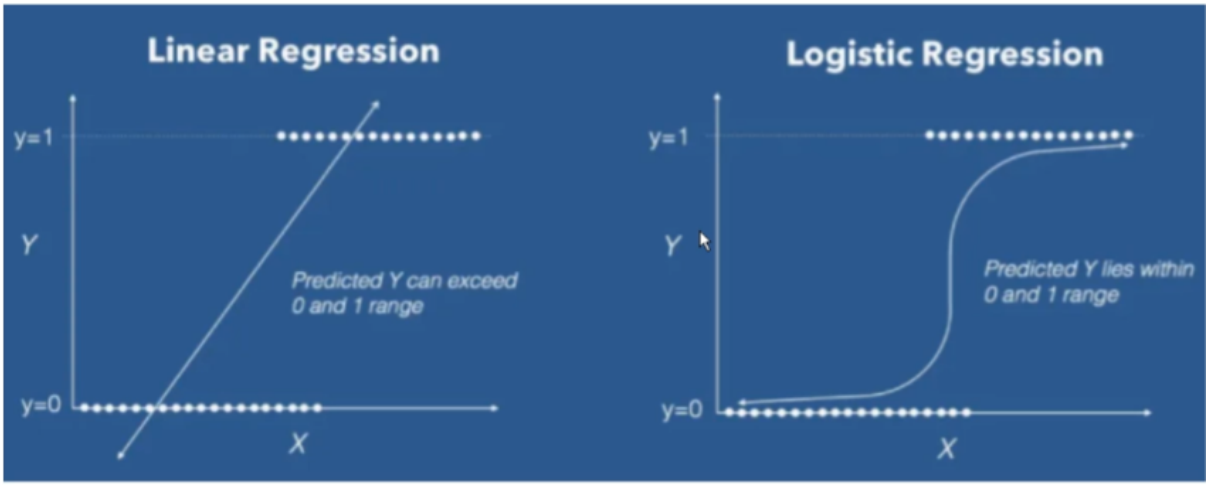
의사결정 트리(Decision Tree)

- 데이터 분류 및 회귀에 사용되는 지도학습 알고리즘.
- **데이터의 특징 속에서 분류에 큰 영향을 끼치는 특징을 발견하고, 상위 노드로 선택하는 알고리즘.**
- 머신러닝에서 의미 있는 질문이란 데이터의 특징 중 의미 있는 특징에 해당하며, 의사결정 트리의 핵심은 바로 영향력이 큰 특징을 상위 노드로, 영향력이 작은 특징은 하위 노드로 선택하는 것.
- 데이터가 주어졌을 때 의사결정 트리는 어떻게 특징별 영향력의 크고 작음을 비교할 수 있을까?
  - 크고 작음을 비교하기위해 수치적인 결과가 필요.

회귀에서 분류가 파생돼 나왔다.

분류모델과 회귀를 같이 사용할 수 있는 이유.

빅분기 실기 - 머신러닝)



엔트로피 - 불확실성(처음에는 100%)

불확실성을 제일 많이 줄여주는 것을 제일 상위에 놓음.

- **엔트로피(Entropy)** : 정보 이론(information Theory)에서는 이 불확실성을 수치적으로 표현한 값.
- **정보 이득(information gain)** : 불확실성이 줄어 든 정도. 즉 질문 이전의 엔트로피에서 질문 후의 엔트로피를 뺀 값.
  - $Gain(T, X) = Entropy(T) - Entropy(T, X)$
- 확률을 바탕으로 정보 엔트로피를 구하는 공식
  - $Entropy = \sum_{i=1}^{n} -p_i \, log_2 p_i$

Pi - 확률

시그마 - n반복

Target이 정답(Y)의 카테고리를 말함.

tar·get

1. 목표; 대상 2. 목표로 삼다 3. 대상으로 삼다

발음 미국식 [ˈtɑ:rgɪt] 영국식 [ˈtɑ:gɪt]

Target을 알 수 있게 영향을 주는 feature들

의사결정 트리 사례로 정보 엔트로피 구하기

이름	군필	긴생머리	성별
김철수	네	아니오	남자
이영희	아니오	아니오	여자
홍길동	네	아니오	남자
최빛나	아니오	네	여자
강감찬	네	아니오	남자
유관순	아니요	아니오	여자

- 엔트로피 = -p(남자) * log(p(남자)) – p(여자) * log(p(여자))
- 의사결정 트리의 최초 엔트로피는 1이며, 계산 방법은
  - – (3/6) * log(3/6) – (3/6) * log(3/6) = 1

군필로 남자를 판단하는 게 더 상위 노드로 간다. 왜냐하면 불확실성을 더 많이 제거할 수 있기 때문에.

엔트로피인 불확실성이 커야 더 많이 제거될 수 있다.

한 가지 특징에 대한 엔트로피 계산

$$Entropy = \sum_{c \in X} P(c)E(c)$$

- X : 선택된 특징.
- c : 선택된 특징에 의해 생성된 하위 노드.
- P(c): 선택된 특징에 의해 생성된 하위 노드에 데이터가 속할 확률.
- E(c): 선택된 특징에 의해 생성된 하위 노드의 엔트로피.

X - 군대냐 생머리냐 feature

P(c): 선택된 특징에 의해 생성된 하위 노드에 데이터가 속할 확률.

- 군대라는 특징으로 데이터를 분리했을 때의 엔트로피

  3/6 * E[3,0] + 3/6 * E[0,3]
  = 3/6 * (-(3/3) * log(3/3) – (0/3) * log(0/3)) + 3/6 * (-(0/3) * log(0/3) – (3/3) * log(3/3))
  = 0

- 긴 생머리라는 특징으로 데이터를 분리했을 때의 엔트로피

  1/6 * E[0,1] + 5/6 * E[3,2]
  = 1/6 * (-(0/1) * log(0/1) – (1/1) * log(1/1)) + 5/6 * (-(3/5) * log(3/5) – (2/5) * log(2/5))
  = 0.966

군대라는 특징으로 데이터를 분리했을 때의 엔트로피

3/6 * E[3,0] + 3/6 * E[0,3] 군대간3명 - 군대 간 사람 중 남자3여자0[3,0], 군대 안 간 3명 - 남자0여자3[0,3]

= 0 나오면 확실하다. → 상위 노드로 가게함.

긴 생머리라는 특징으로 데이터를 분리했을 때의 엔트로피
1/6 * E[0,1] + 5/6 * E[3,2]

= 0.966 불확실성이 높다.

"군대"로 데이터를 분리할 때의 정보 이득 : 1

"긴 생머리"로 데이터를 분리할 때의 정보 이득 : 0.034

군대로 분리: 0-1=1 이라서 이득: 1

긴 생머리로 분리하면 불확실성이 0.034만 해소된다. - 이득

트리를 랭킹으로 해서 가장 상위에 배치시킨다고 보면 이해가 쉬움.

계수는 고정된 값을 가짐, 기준이 되어지는 값을 가짐. 그래서 파생되는 것들이 기준을 참고함.

지니 계수가 높을수록 순도가 높음.

- 순도가 높다는 뜻: 한 그룹에 모여있는 데이터들의 속성들이 많이 일치한다는 뜻.
- 불순도가 높다는 뜻: 한 그룹에 여러 속성의 데이터가 많이 섞여 있다는 뜻.

지니 계수를 통한 의사결정 트리 노드 결정 순서

1. 특징으로 분리된 두 노드의 지니 계수를 구함(P^2 + Q^2)
2. 특징에 대한 지니 계수를 구함.

예) 군대라는 특징으로 "군대를 다녀옴", "다녀오지 않음"으로 구분할 경우 지니 계수

- 군대를 다녀옴 : (0/3)^2 + (3/3)^2 = 1
- 군대를 다녀오지 않음 : (3/3)^2 + (0/3)^2 = 1
- 군대 특징의 지니 계수 : 3/6 * 1 + 3/6 * 1 = 1

예) 긴 생머리를 특징으로 "긴 생머리", "긴 생머리 아님"으로 구분할 경우 지니 계수

- 긴생머리 : (0/1)^2 + (1/1)^2 = 1
- 긴생머리 아님 : (3/5)^2 + (2/5)^2 = 13/25
- 긴생머리 특징의 지니 계수 : 1/6 * 1 + 5/6 * 13/25 = 0.6

네라고 대답 - P^2

아니오 라고 대답 - Q^2

라이브러리는 '사이킷런' 적용
