1차 내부 프로젝트

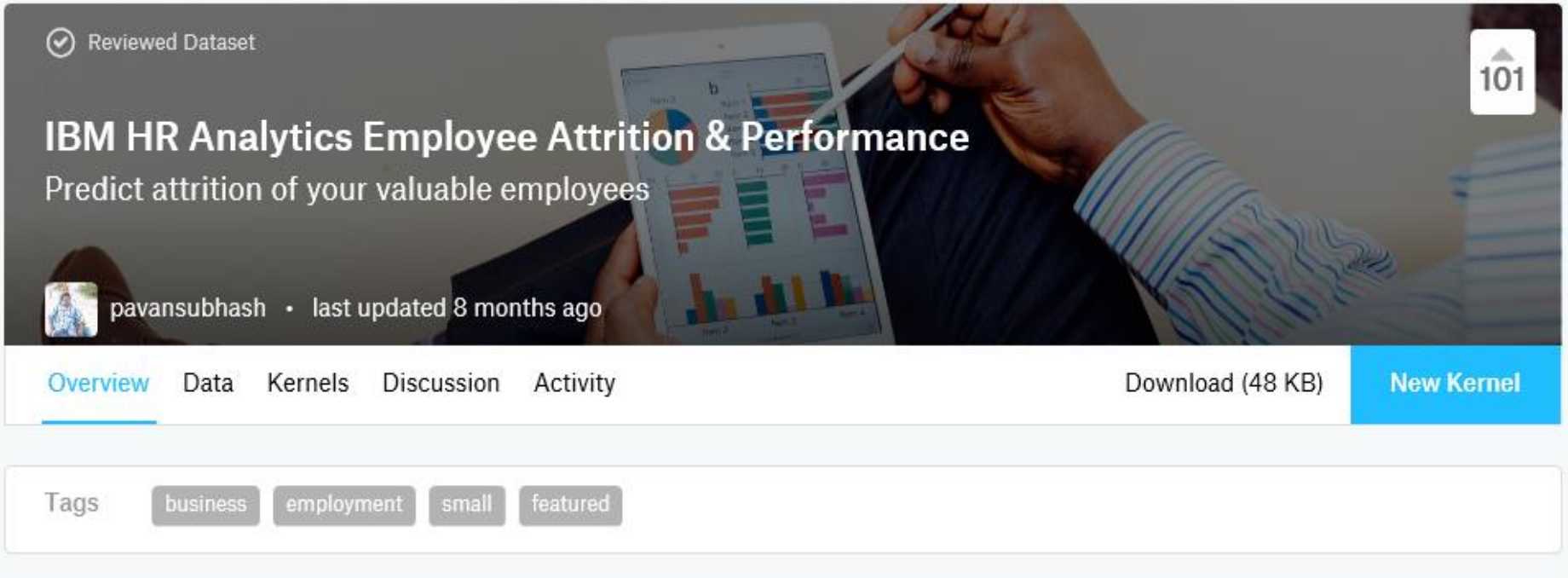
Team_ 김기석 윤성원 손원희 심규민 조연진

프로젝트 목표

HR 데이터를 바탕으로 '직원 이탈'과 관련된 요소를 규명

➔ 직원 이탈 가능성을 낮추기 위한 인사이트 획득

DATA : IBM HR Analytics Employee Attrition & Performance



출처: Kaggle
https://www.kaggle.com/pavansubhasht/ibm-hr-analytics-attrition-dataset

1. PCA (Principal Component Analysis)

[Goal]
차원 축소(Dimension reduction): numerical 변수들의 수를 줄임
➔ 가장 많은 정보를 내포하는 적은 수의 변수(주성분)를 도출함

'주성분'을 찾는 방법은?

1. PCA (Principal Component Analysis)

- 변수 간 선형결합을 도출하여 종속변수를 가장 잘 설명하는 새로운 변수를 만든다
  X_1, X_2 중의 택일 보다는 a = $X_1 + X_2$ 라는 선형 조합이 더욱 유용하다
  WHY?
  A의 분산이 $X_1, X_2, X_1 - X_2$ 의 분산보다 크다

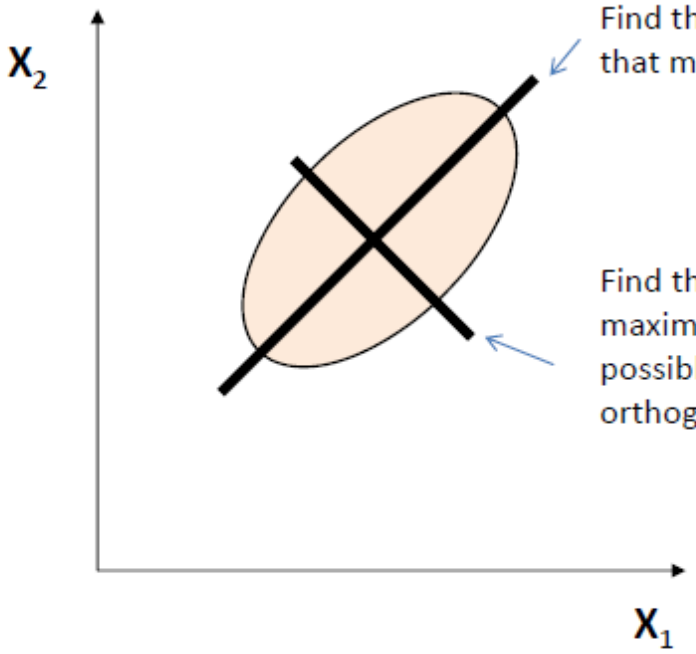
- 따라서 선형 결합들은 서로 비상관성을 가진다.
  오버랩되는 정보의 양은 변수들의 분산을 비교함으로써 파악한다
  ➔ 오버랩 되는 성분들은 웨이트로 차등을 둬서 결합시킨다

- 이렇게 만들어진 변수들을 주성분(Principal Component)이라고 부른다.

1. PCA (Principal Component Analysis)

1) 주성분을 찾는다

$$z_i = u_{i1}X_1 + u_{i2}X_2$$



Find the linear combination of X's that maximizes the variance

First Principal Component

Find the second linear combination that maximizes the variance, among all possible linear combinations that are orthogonal to the first one

Second Principal Component

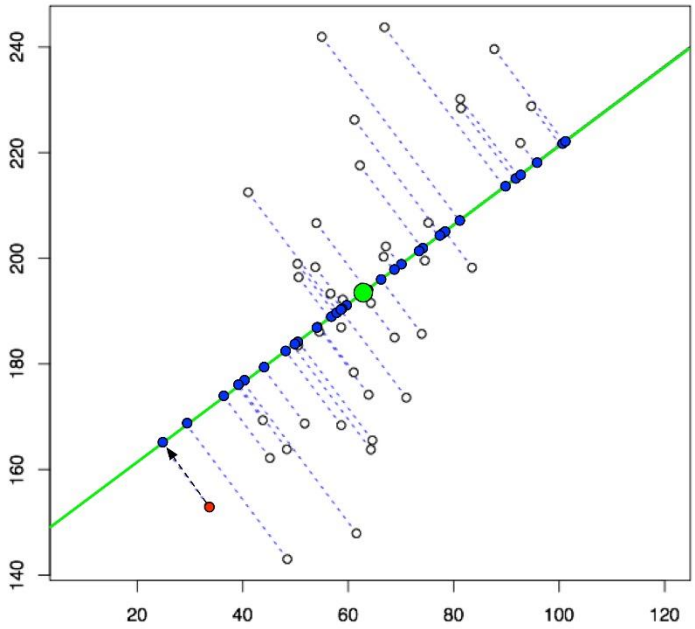
Inseong Song, SNU Business School

1. PCA (Principal Component Analysis)

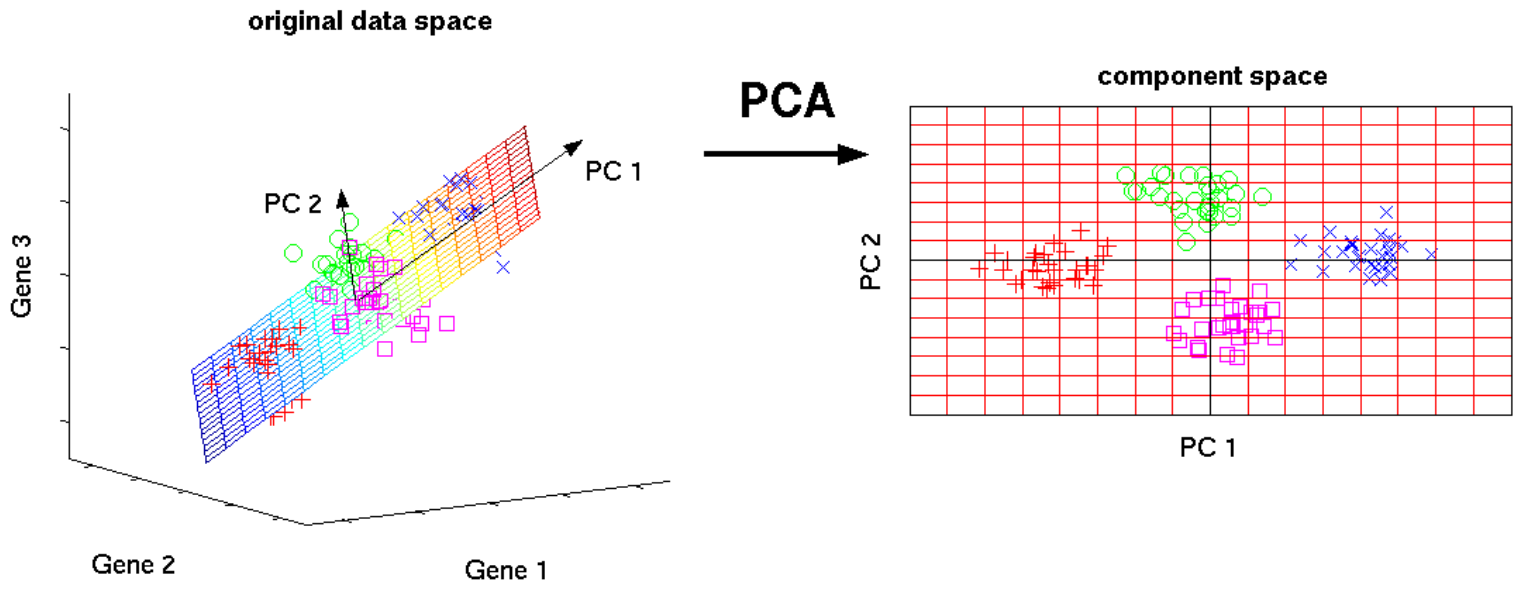
2) 데이터를 변환한다

- 각 성분에 원본 데이터를 사영(projection)시킨다
- 각 성분들을 새로운 축으로 하는 데이터 공간이 만들어진다
- n개 성분을 선택한다면 n차원 데이터 공간이 만들어진다

사영이란?



1. PCA (Principal Component Analysis)



1. PCA (Principal Component Analysis)

[코드소개]

```
pca = PCA(n_components=2)
pca.fit(X_train)
var = pca.explained_variance_ratio_
var_cum=np.cumsum(np.round(var, decimals=4)*100)
var
#var_cum

transformed_data = pca.transform(X_train)
transformed_data
```

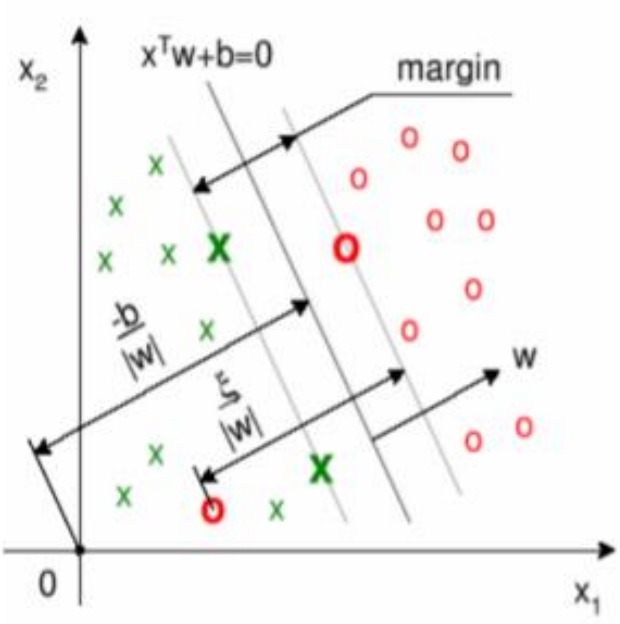
2. SVC (Support Vector Classification)

차원은 충분히 줄어들었다.
그렇다면 어떻게 퇴직자와 퇴직하지 않은 구성원을 구분해낼 수 있을까?

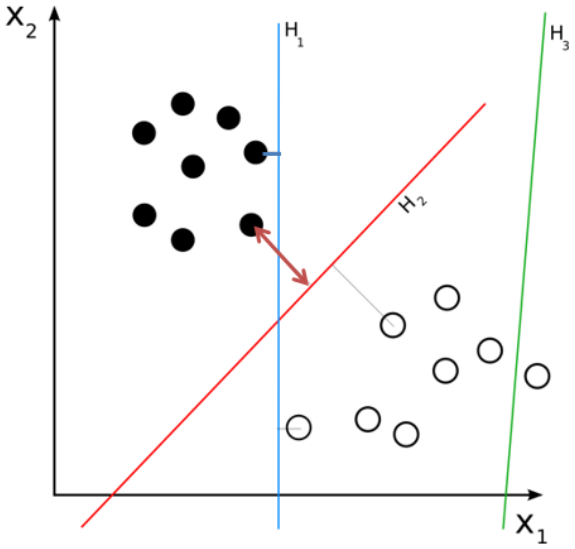
Decision Tree, Naïve Bayes, K-NN method, SVC, else…

2. SVC (Support Vector Classification)

- N차원의 공간에서 경계(초평면)을 그을 때, 경계와 점 간의 간격을 최대화하는 경계를 구한다.
- 이 때 초평면과 가장 가까운 거리에 있어 간격의 기준이 되는 점을 Support Vector로 부른다.
- 이 때 Support Vector와 간격을 라그랑주 승수법을 통해 구한다.



2. SVC (Support Vector Classification)



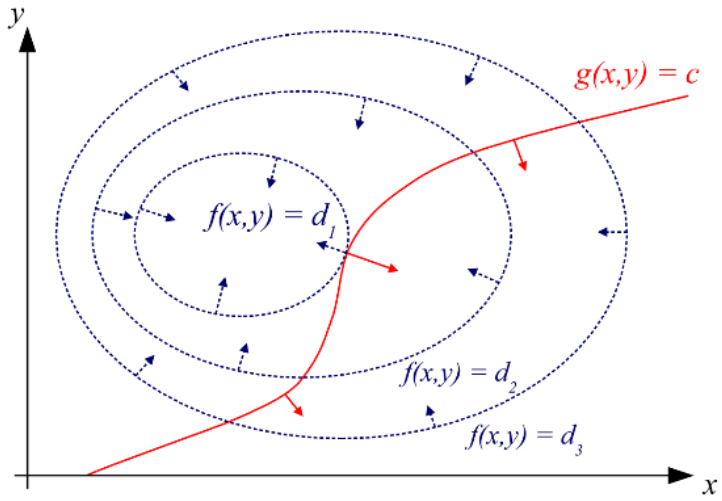
Lagrangian Multiplication
알고리즘

⇩

가장 큰 폭을 가진 경계

⇩

카테고리 구별은 H_1, H_2 모두 가능하지만, H_2 경계가 가장 일반화에 가까운 경계



$$\max U(x,y)$$

subject to

$$g(x,y) = c$$

Then

$$\mathcal{L} = U(x,y) - \lambda(g(x,y) - c)$$

$$[x] \quad \partial_x U = \lambda \partial_x g(x,y)$$

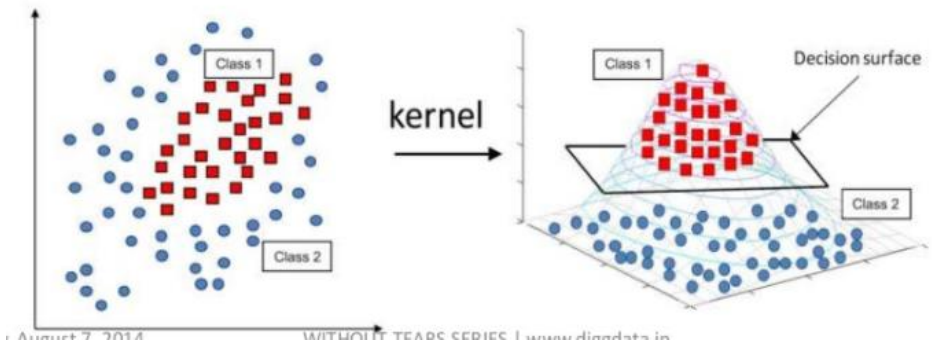
$$[y] \quad \partial_y U = \lambda \partial_y g(x,y)$$

$$[\lambda] \quad g(x,y) = c$$

2. SVC (Support Vector Classification)

SVC의 장점:

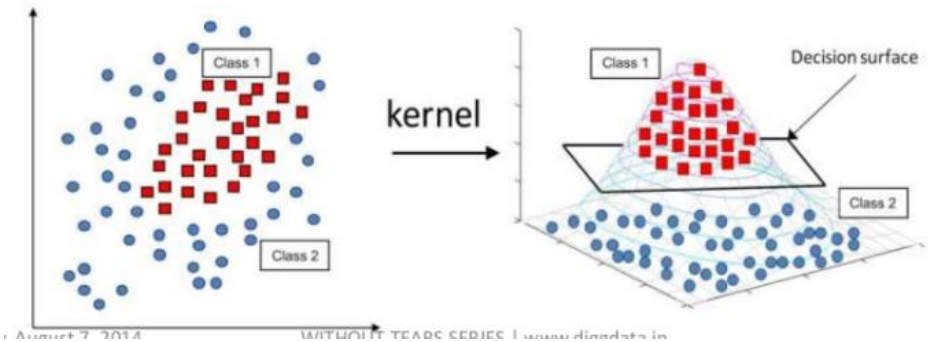
- Kernel을 활용하여 선형이 아닌 초평면으로 경계를 구분할 수 있다. 다양한 Kernel의 활용에 따라 다양한 분석이 가능하다.
- r(또는 감마, Gaussian Kernel에서 활용), C 등의 값을 조절하여 특이값에 대한 조정이 가능하다.
- 라그랑주 승수법으로 최적화를 하게 되므로 최적의 평면을 결정할 수 있다.
- 무엇보다, 정확성이 높다!



2. SVC (Support Vector Classification)

SVC 구현:

- Python, Scikit-Learn을 활용
- PCA로 축소한 차원의 데이터를 Scaling(평균, 분산을 일정하게 균등화)
- Gaussian Kernel의 활용
- C와 gamma 변수값을 0.1, 1, 10으로 변화해 가며 변수에 따른 경계선의 변화에 주목
- Scikit-Learn 내에 있는 변수 최적화 모듈(GridSearchCV)을 사용하여 최적의 C와 gamma를 도출



$$K(\vec{x}, \vec{x}_j) = \exp(-\gamma \|\vec{x} - \vec{x}_j\|^2)$$

```python
def SVM_ex(X_train, y_train):
    C_ex= [1e-1, 1, 1e+1]
    gamma_ex= [1e-1, 1, 1e+1]
    clsfic= []
    for C in C_ex:
        for gamma in gamma_ex:
            clsfic.append((C, gamma, SVM_one(X_train, y_train, C= C, gamma= gamma)))
    return clsfic

def best_parameter(X, y):
    C_range = np.logspace(-2, 4, 7)
    gamma_range = np.logspace(-9, 3, 7)
    param_grid = dict(gamma=gamma_range, C=C_range)
    cv = StratifiedShuffleSplit(n_splits=5, test_size=0.2)
    grid = GridSearchCV(SVC(kernel= 'rbf'), param_grid=param_grid, cv=cv)
    grid.fit(X, y)
    return grid

def SVM_one(X_train, y_train, **kwargs):
    if kwargs.get("grid"):
        grid= kwargs.get("grid")
        clf= SVC(C=grid.best_params_['C'], gamma=grid.best_params_['gamma'], kernel='rbf')
        clf.fit(X_train, y_train)
    elif kwargs.get("C") and kwargs.get("gamma"):
        C, gamma= kwargs.get("C"), kwargs.get("gamma")
        clf= SVC(C=C, gamma= gamma, kernel='rbf')
        clf.fit(X_train, y_train)
    else:
        raise IOError
    return clf
```

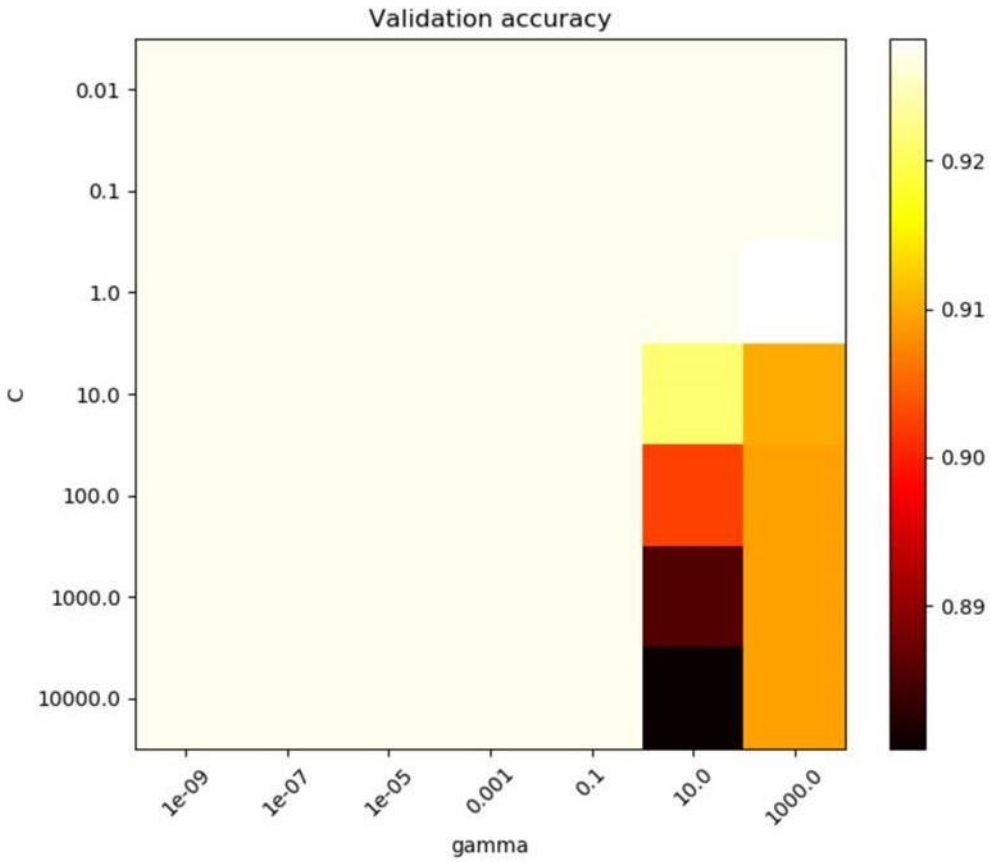
각 C, gamma 별로 SVM을 시행

GridSearchCV로 최적의 변수 추출

입력값이 scikit-learn의 최적 변수인 경우의 SVC 구현

별도로 C, gamma를 설정한 경우의 SVC 구현

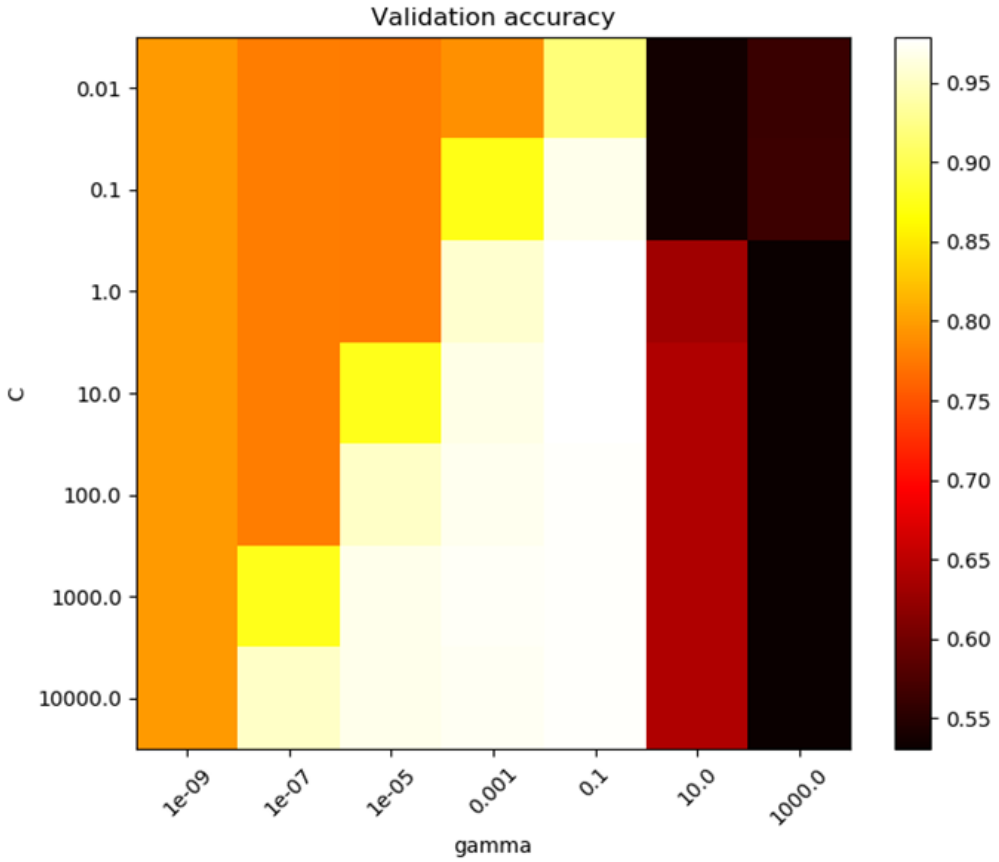
결론



C = 10.0, gamma = 0.1
The Score of Prediction :
0.839590

03 분석 – 실행결과 공유

결론



C = 10.0, gamma = 0.1
The Score of Prediction :
0.9841772151898734

인사이트

1. PCA 알고리즘이 데이터에서 설명력이 낮은 변수를 직접적으로 제거하거나 질적인 해석을 하는 데에는 큰 도움이 되지 않음. 하지만 개별 성분이 큰 의미를 갖지 않는 경우에는 충분히 활용할 수 있음.
   (실제 각 주성분에 거의 모든 변수들이 기여)

2. 특정 집단의 모비율이 매우 낮은 경우 Classification 테스트 정확도가 과대평가되는 경향이 있으므로 테스트 케이스의 분포를 조정해줄 필요가 있어 보임

3. 더미 변수 관련 인사이트
   - Lasso 회귀의 아이디어를 공유해 낮은 영향력을 보완할 필요가 있어 보임
   - 한 영역이 지나치게 많은 더미 변수를 요구하는 경우 의외로 설명력이 크지 않고, 차원만 늘리는 결과를 야기할 수 있음

4. 특정 문제를 해결하는 알고리즘은 고정된 것이 아니며, 방법론 마다 결과가 상이한 경우도 존재. 가령 의사결정 나무를 활용하면 보다 직관적인 설명이 가능. 따라서 장단점을 고려해 결정[Kaggle 타 프로젝트 참조]

활용 가능한 분야

1. PCA
   PCA의 차원 축소는 45개의 필드엔 큰 이점이 없었으나, 천 개 이상 필드(픽셀 등)를 활용하는 데이터의 경우 노이즈 제거에 큰 도움을 줄 수 있다.
   개별 성분의 식별이 중요하지 않은 경우에는 해석보다 예측력이 우선하므로 큰 이점을 지님

2. SVM
   비선형 Classification 지도학습에 활용할 수 있음.
   필드 수가 많아졌을 때(특히 시각화할 수 없는 경우) 직관적이지 않으나 PCA와 마찬가지로 예측력을 우선할 때 이점을 지님.
   (의사결정 나무와 비교하였을 때 bias가 낮음.)

Q&A

Thank you !