Support vector machine

선형이나 비선형 분류, 회귀, 이상치 탐색에도 사용할 수 있는 머신 러닝 방법론으로 복잡한 분류 문제를 잘 해결하고 상대적으로 작거나 중간 크기를 가진 데이터에 적합하다.

Train data

서울시 마포구의 날짜별, 시간별 기상상황과 따릉이 대여 수 데이터

train.isnull().sum()

train.fillna(0,inplace=True)

> 결측치 확인하고 0으로 대체하기

def get\_outlier(x):

  Q1, Q3 = x.quantile([0.25,0.75])

  IQR = Q3-Q1

  upper\_bound = Q3+1.5\*IQR

  lower\_bound = Q1-1.5\*IQR

  outliers=x[(x>upper\_bound)|(x<lower\_bound)]

  return str(round(100\*len(outliers)/len(x),1))+'%'

print(train.apply(get\_outlier))

> IQR 이용하여 이상치 확인하기

x\_train=train.loc[:,['hour','hour\_bef\_temperature','hour\_bef\_precipitation','hour\_bef\_windspeed','hour\_bef\_humidity','hour\_bef\_visibility','hour\_bef\_ozone','hour\_bef\_pm10','hour\_bef\_pm2.5']]

y\_train=train.loc[:,'count']

> train 데이터 X와 Y로 구분하기

from sklearn.svm import SVR

svm\_reg=SVR(kernel='linear')

svm\_reg.fit(x\_train,y\_train)

y\_predict=svm\_reg.predict(x\_train)

> SVM 모델 적합하기

from sklearn.model\_selection import GridSearchCV

from sklearn.svm import SVR

param\_grid=[{'kernel':['linear'],'C':[10.0,30.0,100.0]},

            {'kernel':['rbf'],'C':[1.0,3.0,10.0],

             'gamma':[0.01,0.03,0.1]},

            ]

svm\_reg=SVR()

grid\_search=GridSearchCV(svm\_reg,param\_grid,cv=5,scoring='neg\_mean\_squared\_error',verbose=2,n\_jobs=-1)

grid\_search.fit(x\_train,y\_train)

negative\_mse=grid\_search.best\_score\_

rmse=np.sqrt(-negative\_mse)

rmse

: 53.23435422235155

> 최적의 rmse 찾기