SVM 실습 보고서

Y2023011 윤요섭

Ⅰ. 자료 및 분석 방법

본 보고서는 bankruptcy.xlsx 데이터를 이용하여 LibSVM을 통해 SVM실습과제를 수행하였다. bankruptcy.xlsx 데이터 변수와 이에 대한 설명은 다음과 같다.

변수명	변수설명
CLASS	데이터셋 구분 (1=학습, 0=검증)
X13_1	금융비용대부채비율
X18_1	매출원가비율
X36_1	자기자본비율
X42_1	금융비용부담율증가분
X43_1	매입채무회전율
X74_1	지급여력도
X76_1	분식계수2
X98_1	기업경상이익율
X107_1	현금흐름7대전기총부채
X118_1	총자산변동계수
X129_1	자산대비금융비용증가율
X130_1	자산대비영업외비용증가율
X133_1	매출원가율*매출원가평균증가비
X149_1	매출대비재고자산증가율
X157_1	총자본회전율*매출액증가비
Y	부실여부(1=부실, 0=건전)

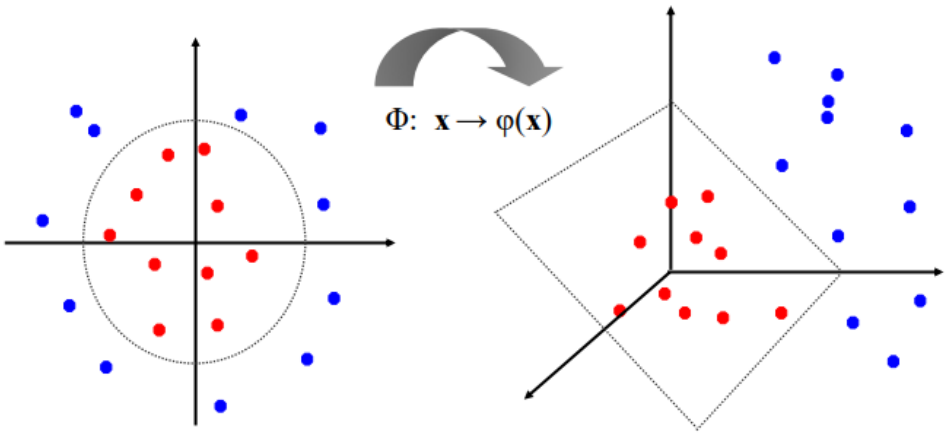
bankruptcy.xlsx 데이터를 통해 SVM실습을 실시하였으며, 이 때, CLASS 변수가 1인 경우 학습용 데이터로 0인 경우 테스트용 데이터로 활용하였다. 종속변수로는 Y(부실여부)로 설정하였으며, CLASS변수와 Y변수 외의 변수를 독립변수로 설정한 뒤, 학습을 진행하였다.

Ⅱ. SVM

본격적인 분석 결과와 해석에 앞서서, 오늘 실습하게 될 SVM의 특징은 다음과 같다. SVM의 발상은 마진을 최대화하는 결정 초평면을 찾는 것이라고 할 수 있다. SVM은 복잡한 분류 문제에 잘 맞으며, 데이터의 양이 많지 않아도 잘 맞추는 특성이 있다. 이러한 SVM의 기본 아이디어는 최대 마진 분류, 소프트 마진 분류, 고차원으로의 사상이다.

SVM에서 최대 마진 분류는 부등식 제약 최적화 문제로, 조건 하에 최대 마진을 갖는 결정 초평면을 찾는 과정이다. 초평면을 찾는 과정에서 마진의 경우, Hard Margin과 Soft Margin으로 나눌 수 있는데, Hard Margin은 매우 엄격하게 초평면을 정의하므로, 모든 입력값은 초평면을 사이로 무조건 한 클래스에 속해야 한다. 그러나 이렇게 할 경우, 몇 개의 이상치로 인해 마진이 매우 작아질 수 있으며, 과적합의 가능성이 크다. 이러한 Hard Margin의 문제점을 해결하기 위해, 여유변수를 도입한 것이 Soft Margin이다. Soft Margin은 오류를 어느 정도 허용함으로써 마진의 크기를 최대로 하는 방법이며, 하이퍼파라미터 C를 사용한다. 하이퍼파라미터 C는 오분류를 허용하는 정도로, C의 값이 작을수록 제약이 작아 오분류에 관대하고 클수록 제약이 커서 오분류의 엄격한 결정 경계를 생성한다. Soft Margin은 마진의 크기를 최대로 하여 여유변수의 크기를 최소로 하는 결정 경계를 찾는 것을 목적으로 한다.

SVM에서 고차원으로의 사상은 저차원에서 선형 분리가 잘 안되는 분류 문제가 있을 때, 데이터를 고차원으로 사상(mapping)시켜 그 공간에서 선형 분리되는 초평면을 찾는 기법이다.



2차원에서 선형 분리되지 않는 문제를 3차원으로 사상하여, 선형 분리하는 초평면(2차원 평면)을 찾는 예시

입력 데이터를 고차원으로 사상(mapping)할 수 있도록 만들어 주는 함수가 커널 함수(kernel function)이다. 대표적인 커널 함수로 선형함수, 다항식함수, 가우시안 RBF, 시그모이드 함수가 있다.

LibSVM(A Library for Support Vector Machines)는 SVM 공개 소프트웨어로 누구나 사용할 수 있으나, LibSVM이 요구하는 입력 데이터 양식이 일반적인 데이터 양식과 많이 다르다.

따라서, 본 실습을 시작하기 전에, bankruptcy.xlsx 데이터를 LibSVM이 요구하는 데이터 양식으로 변환하고, 학습에 사용할 데이터(CLASS=1)을 svm_train으로 변환하여 저장하였으며, 검증에 사용할 데이터(CLASS=0)을 svm_test로 변환하여 저장한 뒤, 실습을 진행하였다.

Ⅲ. 분석 과정과 결과 및 해석

SVM을 통해 학습할 때, 옵션값과 실습시 사용한 값을 나타낸 결과이다.

```
C:₩Temp₩libsvm>svm-train
Usage: svm-train [options] training_set_file [model_file]
options:
-s svm_type : set type of SVM (default 0)
        0 -- C-SVC              (multi-class classification)
        1 -- nu-SVC             (multi-class classification)
        2 -- one-class SVM
        3 -- epsilon-SVR        (regression)
        4 -- nu-SVR             (regression)
-t kernel_type : set type of kernel function (default 2)
        0 -- linear: u'*v
        1 -- polynomial: (gamma*u'*v + coef0)^degree
        2 -- radial basis function: exp(-gamma*|u-v|^2)
        3 -- sigmoid: tanh(gamma*u'*v + coef0)
        4 -- precomputed kernel (kernel values in training_set_file)
-d degree : set degree in kernel function (default 3)
-g gamma : set gamma in kernel function (default 1/num_features)
-r coef0 : set coef0 in kernel function (default 0)
-c cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
-n nu : set the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)
-p epsilon : set the epsilon in loss function of epsilon-SVR (default 0.1)
-m cachesize : set cache memory size in MB (default 100)
-e epsilon : set tolerance of termination criterion (default 0.001)
-h shrinking : whether to use the shrinking heuristics, 0 or 1 (default 1)
-b probability_estimates : whether to train a SVC or SVR model for probability estimates, 0 o
r 1 (default 0)
-wi weight : set the parameter C of class i to weight*C, for C-SVC (default 1)
-v n: n-fold cross validation mode
-q : quiet mode (no outputs)

C:₩Temp₩libsvm>svm-train -t 2 -g 0.5 -c 50 svm_train
...*.*
optimization finished, #iter = 4277
nu = 0.019202
obj = -1024.469267, rho = -0.078498
nSV = 1536, nBSV = 0
Total nSV = 1536
```

실습에서는 가우시안 RBF 커널 함수를 사용하였고, gamma는 0.5, 파라미터 C는 50으로 설정하여 진행하였다.

서포트 벡터 머신의 파라미터 중에 파라미터 C가 오분류의 허용정도를 나타내는 파라미터라면, gamma는 단일 데이터 샘플이 행사하는 영향력의 정도를 나타낸다. gamma 값이 작을수록 과소적합의 가능성이 커지고, gamma 값이 클수록 과적합의 가능성이 커진다.

```
C:₩Temp₩libsvm>svm-predict svm_train svm_train.model tout.txt
Accuracy = 100% (2134/2134) (classification)
```

학습한 데이터를 토대로 학습 데이터를 예측시켜본 결과 100%의 결과가 나온 것을 확인 할 수 있다.

```
C:₩Temp₩libsvm>svm-predict svm_test svm_train.model vout.txt
Accuracy = 81.1567% (435/536) (classification)
```

이를 토대로, 테스트 데이터를 예측 시켜본 결과 81.1567%의 결과가 나온 것을 확인 할 수 있다.

커널별로 파라미터를 조정하여 Validation을 측정 한 결과는 아래와 같다.

구 분		Training	Validation
선형커널	C=1	88.47%	86.94%
	10	88.43%	86.94%
	33		86.94%
	55	88.47%	86.57%
	78	89.50%	86.38%
	100	89.03%	86.38%

선형커널의 경우, C=1 or 10 or 33으로 조정하였을 때, Validation이 86.94%로 제일 높은 것을 확인할 수 있었다.

다항식 커널	C=1	d=1	88.99%	86.19%
		2	88.24%	86.19%
		3	88.05%	86.19%
		4	88.85%	86.01%
		5	90.16%	85.82%
	C=10	d=1	89.46%	85.82%
		2	87.91%	85.82%
		3	87.49%	85.63%
		4	91.38%	85.26%
		5	90.39%	85.26%
	C=33	d=1	89.41%	85.07%
		2	91.89%	84.89%
		3	92.69%	84.70%
		4	89.22%	84.70%
		5	87.11%	84.70%
	C=55	d=1	92.60%	84.51%
		2	88.85%	84.51%
		3	88.85%	84.33%
		4	86.41%	84.14%

다항식 커널에서는 파라미터를 C=1, d=1or 2 or 3으로 했을 때, Validation이 86.19%로 제일 높은 것을 확인할 수 있었다.

RBF 커널	C=1	σ^2=1	85.15%	82.65%
		25	85.15%	82.65%
		50	85.15%	82.65%
		75	85.15%	82.65%
		100	85.15%	82.65%
	C=10	σ^2=1	85.15%	82.65%
		25	85.15%	82.65%
		50	85.15%	82.65%
		75	100.00%	81.16%
		100	100.00%	81.16%
	C=33	σ^2=1	100.00%	81.16%
		25	100.00%	81.16%
		50	99.91%	80.97%
		75	100.00%	80.22%
		100	100.00%	80.22%
	C=55	σ^2=1	100.00%	80.22%
		25	100.00%	80.22%
		50	100.00%	80.22%
		75	100.00%	80.22%
		100	100.00%	79.48%

마지막으로, RBF 커널에서는 C=1, δ^2=1 or 25 or 50 or 75 or 100일 때, 82.65%로 가장 높은 것을 확인 할 수 있었다.

이를 통해, 선형 커널에서의 86.94%로 Validation 값이 가장 높은 것을 확인 할 수 있었다. 이는 SVM이 아닌 선형 문제를 접근할 수 있는 다른 분석 방법으로 접근할 수 있다는 점을 시사한다고 볼 수 있다.