# 서포트 벡터 머신 회귀 (Support Vector Regressor, SVR)

- 본래 분류를 위한 알고리즘이지만, 개념을 확장하여 회귀 문제에도 적용 가능
    - 이를 SVR(Support Vector Regressor)이라고 부름
- SVR의 핵심 아이디어는, 선형 회귀처럼 오차(잔차)를 최소화하는 것이 아니라, **오차를 일정 수준 (epsilon, $\epsilon$)까지 허용**하는 것

- **마진(Margin)과 도로(Street)**: SVR은 예측값($y_{pred}$)을 중심으로 너비가 $2\epsilon$인 '도로'를 설정합니다.
- **오차 허용**: 이 도로 폭($\epsilon$) 안에 들어오는 데이터 포인트(샘플)들의 오차는 무시합니다 (비용=0). 즉, 모델이 어느 정도의 오차는 신경 쓰지 않도록 합니다.
- **마진 위반(Margin Violation)**: 도로 밖에 있는 데이터 포인트들에 대해서만 오차(슬랙 변수, slack variable)를 계산하고, 이 오차를 최소화하려고 노력합니다.
- **서포트 벡터(Support Vectors)**: 도로의 경계선이나 도로 밖에 위치하여 모델의 학습에 직접적인 영향을 주는 데이터 포인트들을 의미합니다.

SVR은 **커널 트릭(Kernel Trick)**을 사용하여 선형적으로 분리할 수 없는 데이터도 고차원 특성 공간으로 매핑하여 비선형 회귀를 수행할 수 있습니다.

### 적용 가능한 상황

- 특성(feature)이 매우 많은 고차원 데이터셋에서 좋은 성능을 보입니다.
- 커널 트릭을 통해 복잡한 비선형 관계를 모델링할 수 있습니다.
- 다른 모델과 달리, 모델의 성능이 데이터의 차원 수에 크게 의존하지 않습니다.
- 이상치에 비교적 덜 민감한 모델을 만들고 싶을 때 (epsilon 마진 내의 작은 오차는 무시하므로).

## 구현 방법

`scikit-learn`의 `svm` 모듈에 있는 `SVR` 클래스를 사용합니다.

### 용도

- 선형 및 비선형 관계를 가지는 연속형 타겟 값을 예측합니다.

### 주의사항

- **특성 스케일링 필수**
    - SVR은 거리 기반 알고리즘이므로, 특성들의 스케일이 다르면 큰 값을 가진 특성이 모델을 지배함
    - **`StandardScaler` 등으로 스케일링 필수**
- **하이퍼파라미터 튜닝**
    - SVR은 성능에 큰 영향을 미치는 여러 하이퍼파라미터를 가지고 있어, `GridSearchCV` 등을 통한 튜닝이 매우 중요
    - `kernel`: 사용할 커널 함수. 'linear', 'poly', 'rbf' 등이 있으며, 'rbf'가 일반적으로 많이 사용됩니다.

- `C`: 규제 파라미터. 마진 위반에 대한 페널티를 조절합니다. `C`가 클수록 마진 위반을 엄격하게 금지하여 모델이 훈련 데이터에 더 적합(과적합 위험)되고, 작을수록 마진을 더 허용(일반화)합니다.
- `epsilon` ($\epsilon$): 오차를 허용하는 도로의 폭 절반. 이 값에 따라 모델의 복잡도와 서포트 벡터의 개수가 달라집니다.
- `gamma`: 'rbf', 'poly' 커널에 사용되는 파라미터로, 하나의 데이터 샘플이 미치는 영향의 범위를 결정합니다. `gamma`가 크면 영향의 범위가 좁아져 모델이 더 복잡해지고 과적합될 수 있습니다.

```
import numpy as np
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error

# 1. 데이터 준비 및 전처리
housing = fetch_california_housing()
X = housing.data
y = housing.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# SVR은 스케일링이 매우 중요
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# 2. SVR 모델 학습 (RBF 커널)
# 주요 하이퍼파라미터
# kernel: 커널 종류 ('linear', 'poly', 'rbf', 'sigmoid')
# C: 규제 파라미터
# epsilon: 오차 허용 범위
# gamma: 커널 계수
svr_rbf = SVR(kernel='rbf', C=1.0, epsilon=0.1)
svr_rbf.fit(X_train_scaled, y_train)
svr_pred = svr_rbf.predict(X_test_scaled)
print(f"SVR (RBF Kernel) MSE: {mean_squared_error(y_test, svr_pred):.3f}") # 0.357

# 3. GridSearchCV를 이용한 최적 하이퍼파라미터 탐색
param_grid = {
    'C': [0.1, 1, 10],
    'gamma': ['scale', 'auto', 0.1],
    'epsilon': [0.1, 0.5]
}

grid_svr = GridSearchCV(SVR(kernel='rbf'), param_grid, cv=3, # cv=5는 시간이 오래
걸릴 수 있음
                        scoring='neg_mean_squared_error', n_jobs=-1)
grid_svr.fit(X_train_scaled, y_train)

print("\nGridSearchCV 최적 파라미터 (SVR):", grid_svr.best_params_) # {'C': 10,
'epsilon': 0.1, 'gamma': 'scale'}
print(f"최적 파라미터 적용 시 MSE: {-grid_svr.best_score_:.3f} (교차검증 평균)") #
```

```
0.325

# 최적 모델로 예측
best_svr = grid_svr.best_estimator_
best_pred = best_svr.predict(X_test_scaled)
print(f"Best SVR MSE: {mean_squared_error(y_test, best_pred):.3f}") # 0.324
```

### 결과 해석 방법

- SVR은 선형 회귀 모델처럼 각 특성의 영향력을 나타내는 직관적인 회귀 계수를 제공하지 않습니다. 모델의 해석이 어렵다는 것이 SVR의 주요 단점 중 하나입니다.
- 모델의 성능은 MSE, $R^2$ 등 일반적인 회귀 평가지표를 통해 평가합니다.
- `support_vectors_` 속성을 통해 어떤 데이터 포인트가 서포트 벡터로 사용되었는지 확인할 수 있습니다.
- `dual_coef_` 속성은 서포트 벡터에 대한 라그랑주 승수 값을 나타내며, 모델의 내부적인 가중치를 의미합니다.

## 장단점 및 대안

- **장점**:
  - 커널 트릭을 통해 복잡한 비선형 관계를 효과적으로 모델링할 수 있습니다.
  - 특성 수가 많은 고차원 데이터에서도 좋은 성능을 보입니다.
  - 규제 파라미터 `C`와 오차 허용 범위 `epsilon`을 통해 모델의 복잡도를 조절하고 과적합을 제어할 수 있습니다.
- **단점**:
  - 데이터의 양이 매우 클 경우 학습 속도가 현저히 느려집니다.
  - 어떤 커널과 하이퍼파라미터를 사용해야 할지 결정하기 위해 교차 검증을 통한 튜닝 과정이 거의 필수적이며, 시간이 많이 소요됩니다.
  - 모델의 내부 동작을 직관적으로 해석하기 어렵습니다.
- **대안**:
  - **랜덤 포레스트, 그래디언트 부스팅**: 대용량 데이터셋에서 SVR보다 빠른 속도와 높은 성능을 보이는 경우가 많으며, 특성 중요도 등 해석을 위한 부가 정보를 제공합니다.
  - **가우시안 프로세스 회귀 (Gaussian Process Regression)**: 예측의 불확실성을 측정할 수 있는 강력한 비선형 회귀 모델이지만, SVR보다도 계산 비용이 더 큽니다.
  - **신경망 (Neural Networks)**: 매우 복잡한 패턴 학습에 강력하지만, 더 많은 데이터와 튜닝 노력이 필요합니다.