

Support Vector Regression: Unleashing the Power of Non-Linear Predictive Modeling



Viswa · Follow

8 min read · Jul 28, 2023



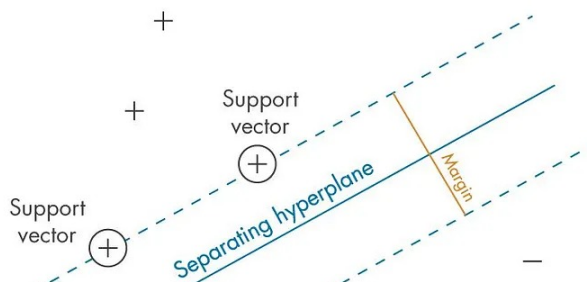
6



1



Support Vector Regression (SVR) is a powerful machine learning technique used for regression tasks, particularly in scenarios where linear regression may not be sufficient due to complex relationships or non-linear patterns in the data. SVR is an extension of the Support Vector Machine (SVM) algorithm, which is primarily used for classification tasks. SVR's ability to handle both linear and non-linear data makes it a tool for various real-world applications, including finance, economics, engineering, and more. In this article, we will explore about SVR.



Sign up to discover human stories that deepen your understanding of the world.

Free

- ✓ Distraction-free reading. No ads.
- ✓ Organize your knowledge with lists and highlights.
- ✓ Tell your story. Find your audience.

Sign up for free

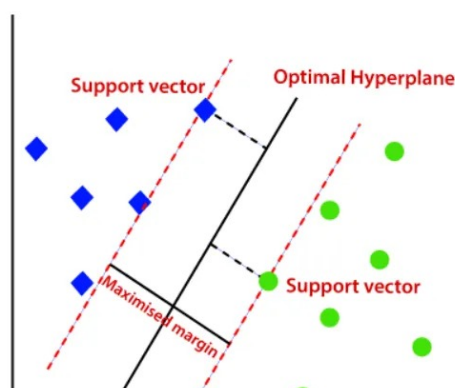
Membership

- ✓ Access the best member-only stories.
- ✓ Support independent authors.
- ✓ Listen to audio narrations.
- ✓ Read offline.
- ✓ Join the Partner Program and earn for your writing.

Try for \$5/month

5. Practical Example: Building a Predictive Model for Output Power in a Combined Cycle Power Plant

6. Conclusion





Introduction

At its core, SVR aims to find a function that maps input features to corresponding output values, making it a regression task. Unlike traditional linear regression, SVR allows for the identification of non-linear relationships by introducing a mapping into a higher-dimensional space through the use of kernel functions. The primary objective of SVR is to minimize the error between the predicted and actual output values while maximizing the margin around the regression line.

Key Components of Support Vector Regression

1. **Hyperplane:** In SVR, the hyperplane represents the regression line that best fits the data. However, in contrast to linear regression, SVR allows for a “margin” of tolerance, which is controlled by two hyperparameters: epsilon (ϵ) and C. Epsilon determines the width of the margin, while C controls the trade-off between maximizing the margin and minimizing the error.

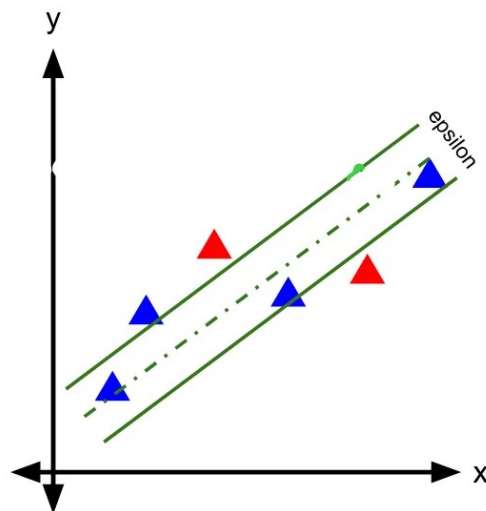
2. **Kernel Functions:** To handle non-linear relationships, SVR uses kernel functions to transform the data into a higher-dimensional space, where linear separation becomes possible. Commonly used kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.

3. **Support Vectors:** These are the data points closest to the hyperplane and play a vital role in defining the SVR model. Only support vectors significantly affect the model, while other data points have little or no impact.

Types of Support Vector Regression

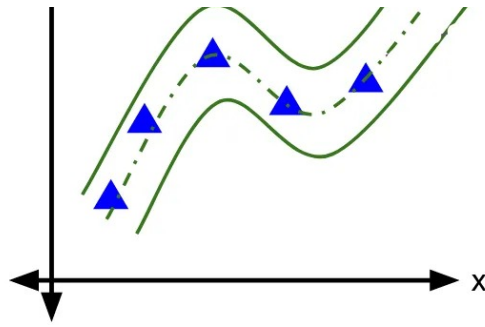
There are mainly three types of SVR models:

1. **Linear SVR:** This type of SVR uses a linear kernel function and aims to find a linear hyperplane that best fits the data.



2. **Nonlinear SVR:** Nonlinear SVR employs various kernel functions (e.g., polynomial, Gaussian radial basis function) to capture complex nonlinear relationships between variables.





3. **Epsilon-Support Vector Regression:** Epsilon-SVR introduces an additional parameter called epsilon, which controls the width of the margin and allows for a certain tolerance of errors.

Advantages of Support Vector Regression

1. **Robustness to Outliers:** SVR is less sensitive to outliers compared to traditional regression techniques. The margin around the hyperplane is determined by support vectors, and outliers are often not considered as support vectors, making the model more resilient to noisy data.
2. **Flexibility in Handling Non-Linear Data:** The ability to use different kernel functions allows SVR to capture complex, non-linear relationships in the data. This flexibility makes it suitable for a wide range of regression tasks with varying complexities.
3. **Regularization:** SVR incorporates regularization through the C parameter, preventing overfitting and promoting a more generalizable model.

Support Vector Regression Practical

This is the section where you'll find out how to perform the support vector regression in Python.

We will analyze data from a combined cycle power plant to attempt to build a predictive model for output power.

Step 1: Importing Python Libraries

The first step is to start your Jupyter notebook and load all the prerequisite libraries in your Jupyter notebook. Here are the important libraries that we will be needing for this linear regression.

- **NumPy** (to perform certain mathematical operations)
- **pandas** (to store the data in a pandas Data Frames)
- **matplotlib.pyplot** (you will use matplotlib to plot the data)

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Step 2: Loading the Dataset

Let us now import data into a DataFrame. A DataFrame is a data type in Python. The simplest way to understand it would be that it stores all your data in tabular format.

```
df = pd.read_csv('Data[1].csv')
df.head()
X = df.iloc[:, :-1].values
y = df.iloc[:, -1].values
```

Step 3 : Splitting the dataset into the Training and Test set

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25,random_state=42)
```

This line imports the function `train_test_split` from the `sklearn.model_selection` module. This module provides various methods for splitting data into subsets for model training, evaluation, and validation.

Here, `X` and `y` represent your input features and corresponding target values, respectively. The `test_size` parameter specifies the proportion of the data that should be allocated for testing. In this case, `test_size=0.25` means that 25% of the data will be reserved for testing, while the remaining 75% will be used for training.

The `random_state` parameter is an optional argument that allows you to set a seed value for the random number generator. By providing a specific `random_state` value (e.g., `random_state=42`), you ensure that the data is split in a reproducible manner.

The `train_test_split` function returns four separate arrays: `X_train`, `X_test`, `y_train`, and `y_test`. `X_train` and `y_train` represent the training data, while `X_test` and `y_test` represent the testing data.

Step 4 : Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc_x = StandardScaler()
sc_y = StandardScaler()
X_train = sc_x.fit_transform(X_train)
y_train = sc_y.fit_transform(y_train)
```

Standardization is a common preprocessing technique used in machine learning to transform data into a standard scale. The scikit-learn library provides a `StandardScaler` class that performs standardization on numerical data.

In the code snippet you provided, the `StandardScaler` class is used to standardize both the input features (`X_train`) and the target variable (`y_train`).

Standardization involves subtracting the mean and dividing by the standard deviation for each feature. This process ensures that all features have a mean of zero and a standard deviation of one. Standardizing the data can be beneficial for certain machine learning models and algorithms as it helps to bring the features to a comparable scale and prevents any single feature from dominating the learning process.

By creating an instance of the `StandardScaler` class, such as `sc_x` for features

and `sc_y` for the target variable, you can fit the scalers to the training data using the `fit_transform()` method. This method calculates the mean and standard deviation for each feature or the target variable and applies the standardization formula.

Once the standardization is applied, the transformed features and target variable are stored back into `X_train` and `y_train`, respectively. The standardized data can now be used for training machine learning models.

Step 5: Training the Support vector regression model on the Training set

```
from sklearn.svm import SVR
regressor = SVR(kernel = 'rbf')
regressor.fit(X_train,y_train)
```

The first line of code imports the SVR class from the `sklearn.svm` module, which provides implementation for Support Vector Regression.

Next, you create an instance of the SVR class and assign it to the variable `regressor`. The kernel parameter is set to “rbf”, indicating that you want to use the radial basis function (RBF) kernel for the SVR model. The RBF kernel is a popular choice for SVR as it is capable of modeling complex nonlinear relationships between the features and the target variable.

The `fit()` method of `regressor` is then called, with `X_train` (the input features) and `y_train` (the target variable) as the arguments. This method trains the SVR model on the provided training data, using the RBF kernel to capture the underlying patterns and relationships between the features and the target variable.

During the training process, the SVR model will determine the support vectors and learn the optimal hyperplane that best fits the training data, aiming to minimize the error between the predicted and actual target values.

Once the `fit()` method completes, the SVR model (`regressor`) will have learned from the training data and be ready to make predictions on new, unseen data.

Step 6: Predicting the Test set results

```
y_pred = sc_y.inverse_transform(regressor.predict(sc_x.transform(X_test)).reshape(-1,))
np.set_printoptions(precision=2)
print(y_pred)
```

The `transform` method of `sc_x` is applied to `X_test` to standardize the new data using the same scaling factors computed on the training data. Then, the `predict()` method of `regressor` is used to make predictions on the standardized `X_test`.

The predicted target variable (`y_pred`) is passed to the `inverse_transform()` method of `sc_y`. This method undoes the standardization performed on the training target variable, bringing the predicted values back to their original scale. The `reshape(-1,1)` is used to reshape the predicted values to a column vector.

The code ensures that the predicted values are transformed back to their original scale using the inverse transformation of the target variable's standard scaler (sc_y). This is important to provide predictions in the original units of the target variable rather than the standardized values.

Step 6 : Evaluating the Model Performance

```
from sklearn.metrics import r2_score
r2_score(y_pred,y_test)
```

This code imports the r2_score function from scikit-learn's metrics module. The r2_score function is commonly used as an evaluation metric for regression models, including linear regression. It measures the proportion of the variance in the target variable that is predictable from the input features.

A higher R-squared score indicates a better fit of the regression model to the data, where 1 represents a perfect fit and 0 represents no relationship between the predicted and actual values.

An R-squared score of 0.9438 indicates that approximately 94.38% of the variance in the target variable can be explained by the predictions of the SVR model. This suggests a strong fit of the model to the data.

Link to the code:

Regression/Support_Vector_Regression.ipynb at main · ViswaKiranAndraju/Regression

All codes of regression module. Contribute to ViswaKiranAndraju/Regression development by creating an...

github.com

anAndraju/
ion

ression module

0 Issues 0 Stars 0 Forks

Conclusion

Support Vector Regression is a valuable tool in predictive modeling, particularly when dealing with complex and non-linear datasets. By introducing a margin of tolerance and employing kernel functions, SVR can handle diverse real-world regression problems effectively. Its robustness to outliers, flexibility in handling non-linear data, and guarantee of a global optimum make it a preferred choice for many researchers and practitioners. As the field of machine learning continues to evolve, SVR remains an essential technique in the data scientist's toolkit for accurate and reliable regression analysis.

Support Vector Regression

Machine Learning

Support Vector Machine

Predictive Modeling

Artificial Intelligence

6 1

Written by Viswa

4 Followers

Follow

I am a passionate writer, I specialize in crafting engaging content at the intersection of data and technology. With a focus on data science, machine learning.

More from Viswa

Viswa

Unveiling Decision Tree Regression: Exploring its...

In the world of machine learning, decision tree regression is a powerful algorithm used...

7 min read · Jul 31, 2023



Viswa

Demystifying Polynomial Regression: Understanding and...

In the field of machine learning and statistical modeling, regression analysis plays a vital...

7 min read · Jul 26, 2023



Viswa

Logistic Regression

Logistic Regression is a statistical technique widely used in machine learning and...

7 min read · Mar 24, 2024



Viswa

Linear Regression Model Practical

This is the article where you'll find out how to perform the linear regression in Python.

4 min read · Jul 26, 2023



See all from Viswa

Recommended from Medium

NANDINI VERMA

An Introduction to Support Vector Regression (SVR) in Machine...

Support Vector Regression (SVR) is a machine learning technique used for...

3 min read · Nov 3, 2023



Suraj Singh Bisht

Mastering Gradient Descent: Math, Python, and the Magic Behind...

Implementing a Python class to auto-compute gradient values like PyTorch tensor

19 min read · Dec 26, 2023



Lists

Predictive Modeling w/ Python

20 stories · 1102 saves

AI Regulation


6 stories · 415 saves

Natural Language Processing

1377 stories · 870 saves

Practical Guides to Machine Learning

10 stories · 1316 saves


 Mahendra Nath Reddy E

Understanding Regression Models: A Comprehensive Guide

In predictive modeling, regression analysis stands tall as one of the most fundamental...

4 min read · Mar 24, 2024




 Ratan Kumar Sajja in Python in Plain English

Comparing Linear Regression and Random Forest Regression Using...

Power of Random Forest Regression

6 min read · Oct 21, 2023



 Yoann Mocquin in Towards Data Science

Linear Regression, Kernel Trick, and Linear-Kernel.

Sometimes the kernel trick is useless.

★ · 8 min read · Nov 5, 2023



 Viswa

Logistic Regression

Logistic Regression is a statistical technique widely used in machine learning and...

7 min read · Mar 24, 2024



See more recommendations