**🔍 What is Support Vector Regression (SVR)?**

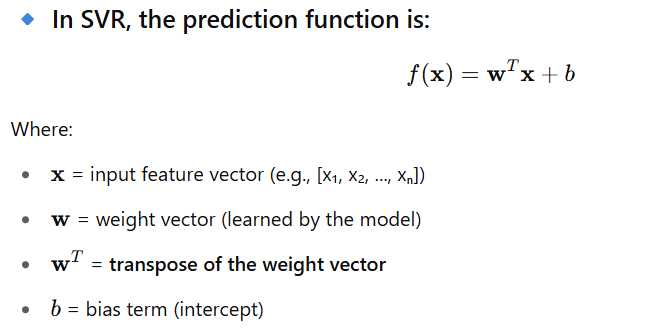
* + 1.1.1.2 Support Vector Regression (SVR)

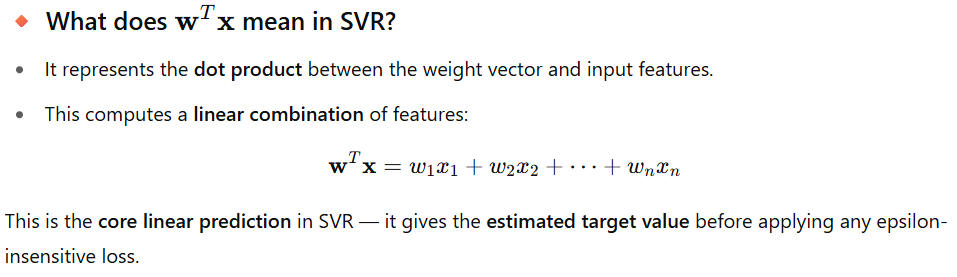
**ε (epsilon) – Epsilon-Insensitive Tube**

* **Name:** Epsilon (Greek letter)
* **Meaning:** Defines a **margin of tolerance** where no penalty is given.
* **Role:** The model does not care (i.e., is *insensitive*) to errors within ±ε range from the actual value.
* **Also called:** Epsilon-insensitive loss margin or epsilon-tube.

👉 Think of it as a "safe zone" — predictions within this margin are not penalized.

**🔹 w – Weight Vector**

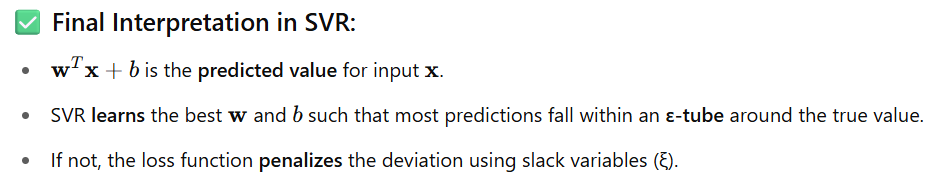




**Why the transpose?**

* Vectors w and x are typically column vectors.
* To compute the dot product, we need one to be a **row vector** — so we use the **transpose** of





**🔹 ξ and ξ\* (xi and xi-star) – Slack Variables**

* **Name:** Slack variables (Greek letter xi: ξ)
* **Meaning:** Measure how far the predictions are **outside** the epsilon margin.
* **Role:**
  + **ξi** : deviation above ε
  + **ξ\***​: deviation below -ε
* **Also called:** Loss variables, error distances, or constraint violations

👉 They represent how much a prediction **violates** the margin — these values are **penalized** during training.

**Support Vector Regression (SVR) – A Deep Dive**

**✅ Category:**

* Supervised Learning
* Regression Algorithm (predicts continuous values)
* Based on **Support Vector Machines (SVM)**

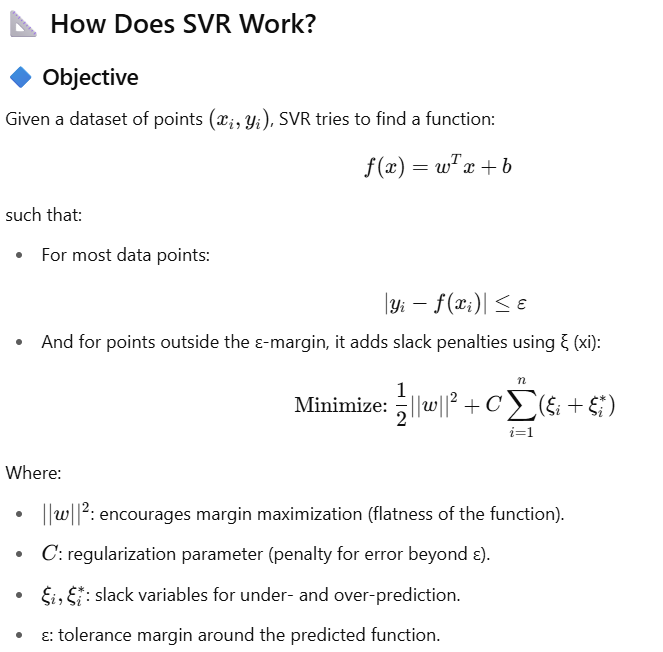
**What is Support Vector Regression (SVR)?**

SVR is a type of regression algorithm that tries to find a function that approximates the data within a certain acceptable error margin (ε).

Unlike linear regression that tries to minimize the total error (like MSE), SVR:

* Tries to fit the best hyperplane (or curve) within an ε-insensitive tube.
* Only penalizes predictions that fall outside this ε margin.
* Focuses on support vectors: the data points that lie on the edge or outside the ε-tube.

[**https://www.geeksforgeeks.org/support-vector-regression-svr-using-linear-and-non-linear-kernels-in-scikit-learn/**](https://www.geeksforgeeks.org/support-vector-regression-svr-using-linear-and-non-linear-kernels-in-scikit-learn/)

****

**Graphical Intuition**

* SVR creates a tube of width 2ε around the regression line.
* Points inside the tube are considered correctly predicted.
* Only the points outside the tube influence the model — these are the support vectors.

**🧠 Key Concepts**

| **Term** | **Meaning** |
| --- | --- |
| **Epsilon (ε)** | ε-tube width: determines how much deviation from actual y is acceptable |
| **Support Vectors** | Data points that lie outside the ε margin |
| **Kernel Trick** | Maps input features into higher-dimensional space (for nonlinear regression) |
| **Regularization (C)** | Controls the trade-off between flatness and tolerance for outliers |

**🧮 Kernel Functions in SVR**

SVR can perform both **linear** and **non-linear** regression using kernels.

| **Kernel Type** | **Description** |
| --- | --- |
| Linear | Works well when data is linearly separable |
| Polynomial | Fits polynomial curves |
| RBF (Gaussian) | Popular for non-linear problems |
| Sigmoid | Behaves like a neural network |

**🎯 Use Cases of SVR**

| **Domain** | **Application** |
| --- | --- |
| Finance | Stock price forecasting |
| Healthcare | Predicting patient recovery time |
| Energy | Load forecasting in power grids |
| Engineering | Sensor signal estimation |
| Marketing | Forecasting ad campaign performance |

**⚙️ How SVR Works (Training Process)**

1. Choose a kernel (linear or nonlinear)
2. Define the ε margin
3. Use support vectors that fall outside the margin to define the model
4. Optimize using **Quadratic Programming** to find the best fit with minimum deviation

**📈 Assumptions of SVR**

* Data may or may not be linearly separable (kernels help with nonlinear data)
* It can tolerate some outliers (controlled via C parameter)
* Works best with **normalized or scaled data**

**Dataset: California Housing (predicting median house value)**

# 📌 Step 1: Import Libraries

import numpy as np

import matplotlib.pyplot as plt

from sklearn.datasets import fetch\_california\_housing

from sklearn.svm import SVR

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import mean\_squared\_error, r2\_score

from sklearn.pipeline import make\_pipeline

# 📌 Step 2: Load Dataset

data = fetch\_california\_housing()

X = data.data # Features

y = data.target # Target: Median house value

# Let's take only 1 feature for visualization (e.g., average rooms per household)

X = X[:, [3]] # 'AveRooms'

# 📌 Step 3: Train/Test Split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# 📌 Step 4: Define SVR Model Pipeline (with scaling)

svr\_model = make\_pipeline(

StandardScaler(), # Normalize data

SVR(kernel='rbf', C=100, epsilon=0.1) # RBF kernel

)

# 📌 Step 5: Train the Model

svr\_model.fit(X\_train, y\_train)

# 📌 Step 6: Make Predictions

y\_pred = svr\_model.predict(X\_test)

# 📌 Step 7: Evaluate the Model

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("✅ Mean Squared Error (MSE):", mse)

print("✅ R^2 Score:", r2)

# 📌 Step 8: Visualize

plt.figure(figsize=(10,6))

plt.scatter(X\_test, y\_test, color='blue', label='Actual')

plt.scatter(X\_test, y\_pred, color='red', alpha=0.6, label='Predicted')

plt.xlabel('Average Rooms per Household')

plt.ylabel('Median House Value')

plt.title('SVR - California Housing')

plt.legend()

plt.grid(True)

plt.show()

🔍 Result Explanation

* Mean Squared Error (MSE): Measures average squared difference between actual and predicted values (lower is better)
* R² Score: Indicates how well the predictions match the actual values (1 is perfect)

**✅ Pros and Cons**

**✅ Pros:**

* Works for both linear and non-linear data
* Can model complex relationships with kernel tricks
* Robust to outliers (due to ε margin)
* High generalization capability

**❌ Cons:**

* Computationally expensive for large datasets
* Choosing the right kernel and tuning parameters (C, ε, gamma) is critical
* Difficult to interpret compared to linear regression

**🔎 When to Use SVR?**

* When the relationship between input and output is **non-linear**
* When **outliers** are present in your dataset
* When you want **flexible and smooth fitting**
* When dataset is **small to medium-sized**

**Summary Table**

| **Concept** | **Meaning** |
| --- | --- |
| ε-tube | The margin where predictions are not penalized (penalties) |
| C | Penalty parameter — controls tolerance to errors |
| Support Vectors | Points outside the ε margin — drive the model |
| Kernels | Functions to transform data into higher dimensions |
| Slack variables | Allow some points to be outside the ε margin |