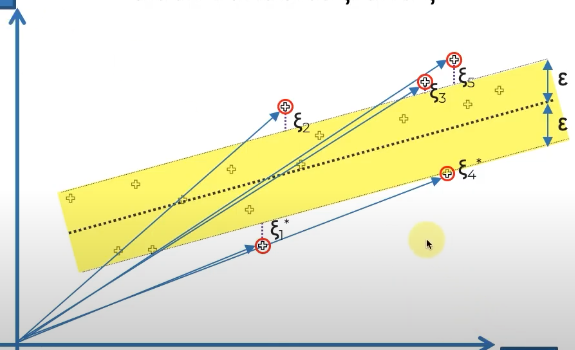
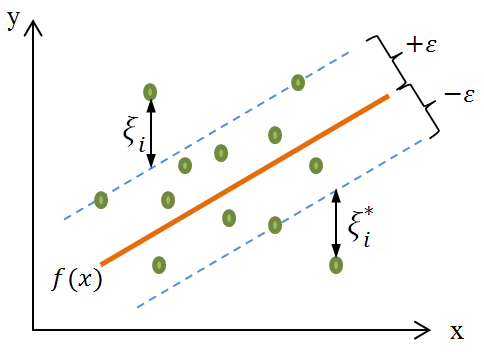
**הסבר SVR**

SVM (Support Vector Machines) הם אלגוריתמי למידת מכונה המשמשים לקלסיפיקציה, רגרסיה ומציאת רעש.

SVR (Support Vector Regressor) הוא מודל ניבוי העובד בצורה הבאה:

* מימד = פיצ'ר / עמודה



תחילה הוא ממפה את הדאטה במרחב n מימדי (בהתאם למספר הפיצ'רים) אבל כמובן לשם נוחות והבנה אנחנו נסתכל על המרחב כדו-מימדי (Y – משתנה המטרה, X – שאר הפיצ'רים).

הוא מותח קו רגרסיה כללי (הנקרא Hyperplane – הכתום – במימדים גדולים יותר מדובר במישור ולא בקו) ובהתאם לאפסילון מוגדר, הוא יוצר "צינור" שהרדיוס שלו הוא אפסילון (הנקרא -

Epsilon Insensitive Tube), השאיפה שרוב הדאטה יפול בתוך הצינור. בעצם המודל "מתעלם" מכל נקודות הדאטה שכעת נמצאות בתוך הצינור כדי למזער את השגיאה.

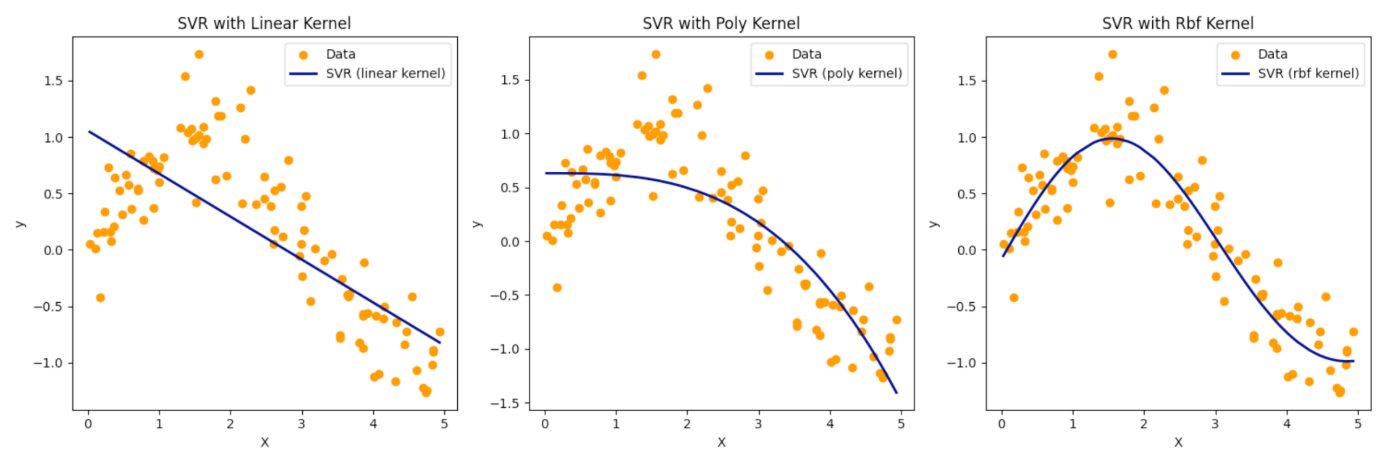
אפשר לחשוב על זה שקו הרגרסיה שלנו הוא פשוט הצינור (קו עווה מאוד) וכעת אנחנו מודדים את המרחקים של הדאטה מדפנות הצינור כשהמטרה שהמרחק (השגיאה) יהיה מינימלי.

הנקודות הכי קרובות לצינור נקראות Support Vectors כי ניתן לחשוב עליהם כוקטורים "התומכים" בצינור שלנו.

ל SVR קיימים אופציות שונות לקרנלים בהתאם לקשר הנתונים שלך.

Linear – כמו בדוגמא למעלה, כאשר יש קשר ליניארי (קורלציה) חזקה בין משתנים.

RBF – מתאים את הצינור להתנהגות הדאטה שלך ולכן מאוד מדויק אך צורך המון מהמחשב ולא מיועד לכמויות גדולות של מידע.



**פרמטרים: (משפיעים על אופי הצינור ומעניקים שליטה על שגיאות)**

Epsilon – קובע את "עובי" הצינור.

C – קובע את מידת פיתול הצינור.

תחילה ניסיתי את גרסת הברירת מחדל של SVR (שהוא קרנל RBF) אך לאחר מעל 50 דקות של נסיון לאמן את המודל על הדאטה הבנתי שזה לא ריאלי. לאחר מכן ניסיתי להתחכם ולפצל את הדאטה לתתי קבוצות קטנות שעל כל אחד מהם מריצים את המודל ואז בסוף מריצים את המודל על כל ה Support Vectors שמצאנו בתתי קבוצות אך גם זה לא עבד. בסוף ניסיתי את הקרנל הליניארי שעבד מהר מאוד אך מכיון שהקשרים בין המשתנים כלל לא ליניאריים אז המדדי תוצאה היו נמוכים מאוד ללא מקום לשיפור משמעותי ולכן עברתי לחפש מודל שמתאים לקשרים לא ליניאריים ועל דאטה גדול.

**הסבר kNN**

K Nearest Neighbors – גם הוא ממפה את הדאטה למרחב n מימדי ומאפשר אופציות של קלסיפיקציה ורגרסיה.

בהקשר של רגרסיה, בהתאם ל K מוגדר, הוא לוקח את ה K שכנים הכי קרובים לנקודה ועושה ממוצע שלהם כדי לקבוע איפה יעמוד התוצאה.

זהו מודל פשוט יחסית אך יש מקום להרבה tweaking (כמו מספר ה – K, המרחק – אוקלידי, מנהטן..., משקלים – האם כל נקודה שוות ערך בחישוב הממוצע? או האם כל נקודה תקבל משקל מסויים בהתאם למרחק שלה מהתוצאה).

* כמובן שמספר K נמוך יגרום ל Overfitting ו K גבוה מדי יגרום ל Underfitting.

**דוגמה:**  
אם k=1 התחזית תהיה בדיוק כמו ערך הנקודה הקרובה ביותר, כלומר המודל יהיה מאוד רגיש לרעש.  
אם k=10 התחזית תבוסס על ממוצע 10 הנקודות הקרובות, מה שעשוי להקטין את ההשפעה של רעש.

* כל אלגוריתם שממפה את הדאטה שלו (כמו אלה) מצריך Scaling/Normalizing כדי שהטווחים יהיו שווים ובכך לא יהיו נקודות דאטה במרחק עצום מהשאר, ובכך אני מונעים הטיה.