AI200: APPLIED MACHINE LEARNING

SUPPORT VECTOR MACHINE (SVM)

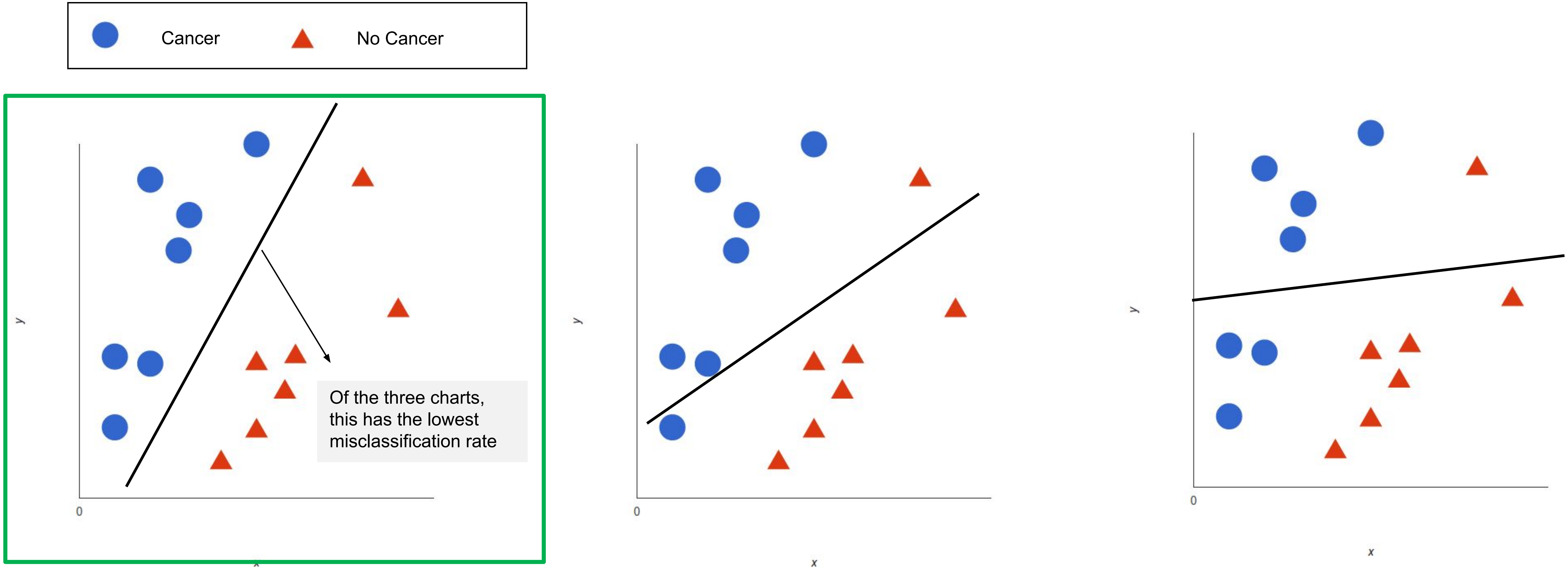
OVERVIEW & LITERATURE OF MACHINE LEARNING

Machine Learning

1. Spectrum of supervision	Supervised Learning		Unsupervised Learning			Reinforcement Learning
2. Types of problems	Regression	Classification	Clustering	Association	Dimensionality Reduction	
3. Types of algorithm / models	Linear regression Polynomial regression Neural Networks Decision Tree Random Forest XGBoost	Logistic Regression KNN Naïve Bayes $\boxed{\text{SVM}}$ Neural Networks Decision Tree Random Forest XGBoost	Neural Networks K-Means Hierarchical Clustering Hidden Markov Models Gaussian Mixture	Apriori Algorithm FP-Growth Algorithm Eclat Algorithm	Neural Networks SVD PCA	
4. Types of applications	Forecasting Predictions Process optimization New Insights	Fraud Detection Image Classification Customer Retention Diagnostics	Recommendations Targeted Marketing Customer Segmentation	Market Basket Analysis	Feature Elicitation Structure Discovery Meaningful Compression Big Data Visualisation	Real-time Decisions Game AI Learning Tasks Skill Acquisition Robot Navigation

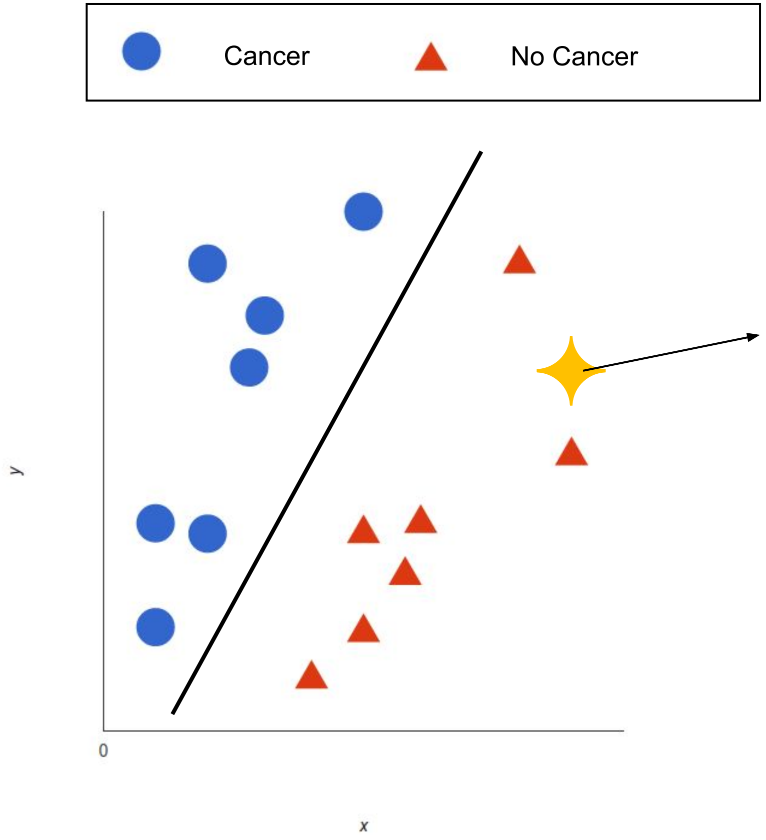
WHAT IS SVM: LAYMAN INTUITION

- Here is how a Support Vector Machine works:
  - ***[Training Step]*** We iteratively draw a line / plane and calculate the misclassification rate in each instance. The line / plane that with the **lowest misclassification rate** is chosen to serve as the classifier.



WHAT IS SVM: LAYMAN INTUITION

- Here is how a Support Vector Machine works:
  - ***[Training Step]*** We iteratively draw a line / plane and calculate the misclassification rate in each instance. The line / plane that with the **lowest misclassification rate** is chosen to serve as the classifier.
  - ***[Prediction Step]***
    To classify a new data point, we **identify which side of the classifier** this data falls on.



This new data point is classified as “No Cancer” as it falls on the other side of the SVM classifier.

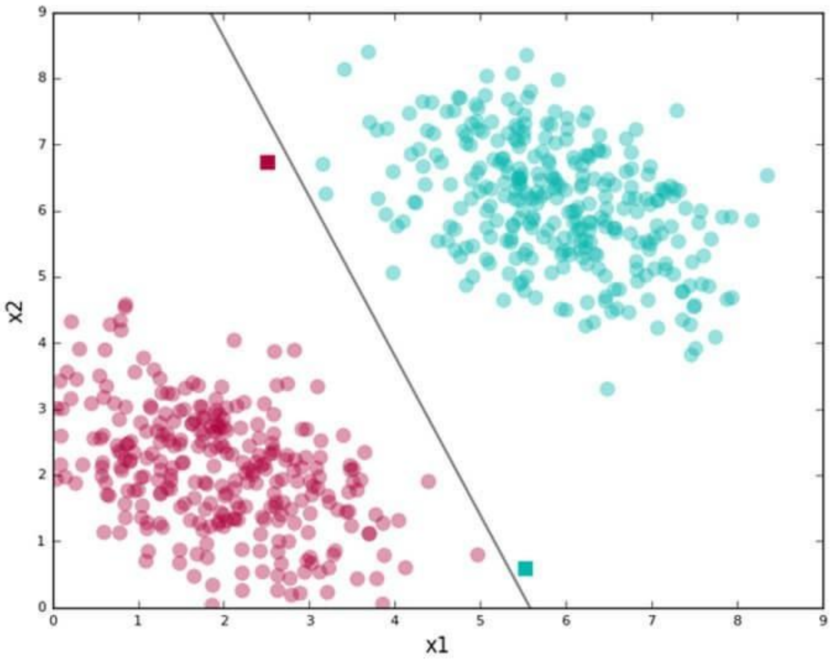
SUPPORT VECTOR MACHINES (SVM)

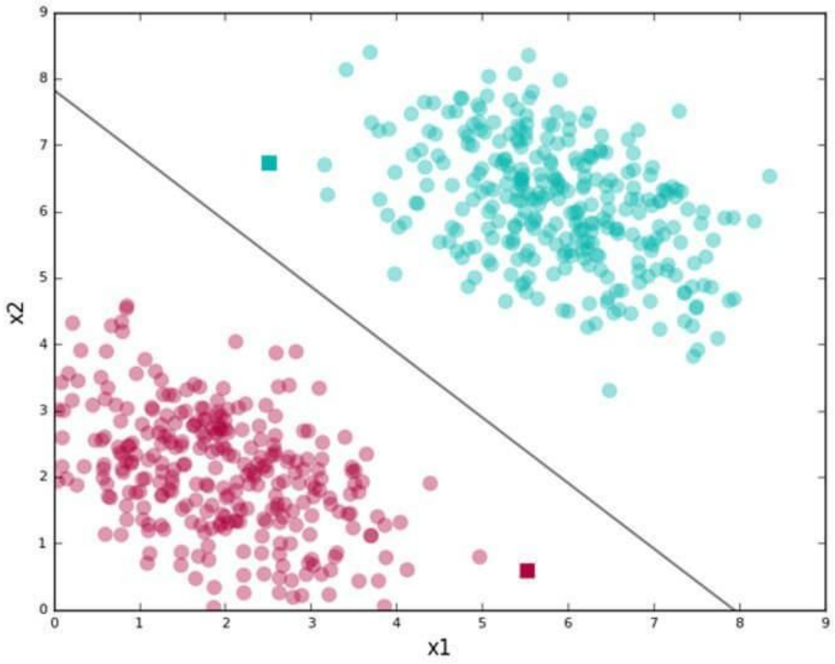
MECHANISM BEHIND MODEL

MECHANISM BEHIND MODEL: SUPPORT VECTORS

- But **surely there are many ways** to draw a line/plane such we can perfectly classify our training data?
- In the image below, both lines separate the red and green clusters.

- **Would you prefer the line drawn on the left chart, or the right chart?**



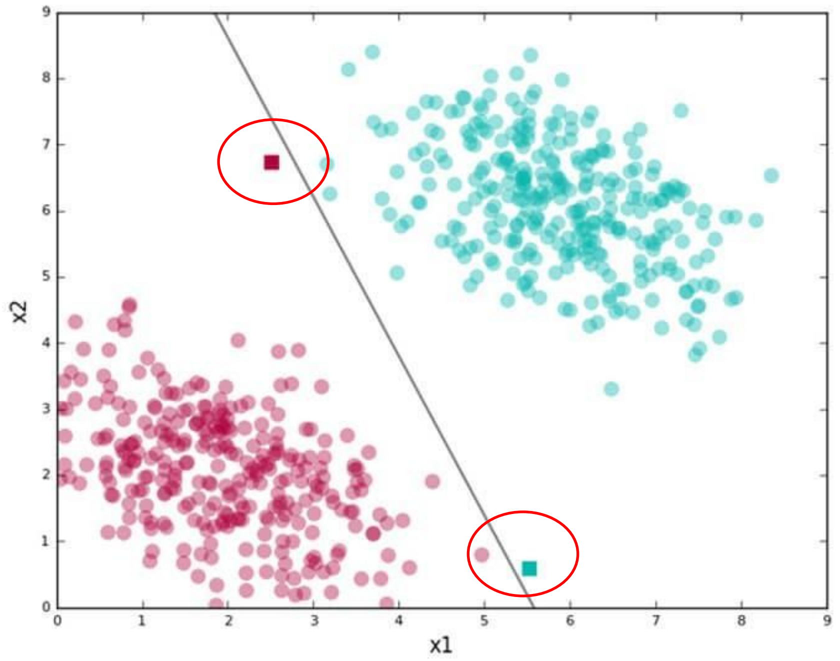


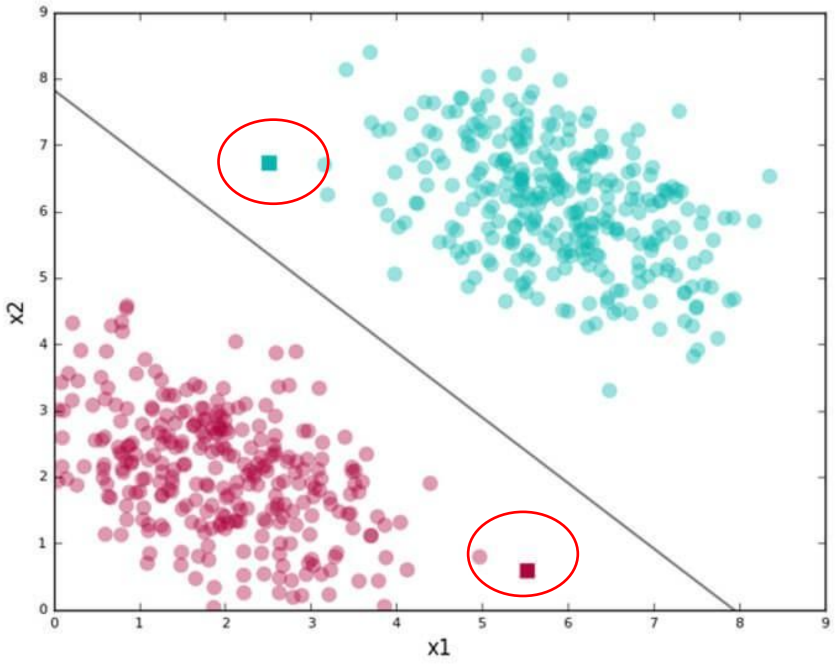
MECHANISM BEHIND MODEL: SUPPORT VECTORS

- Remember that **the worth of a classifier is not in how well it separates the training data**. We eventually want it to classify yet-unseen data points, so we must choose a line that *captures the general pattern* in the training data so that it does well on the future datasets.

- The line on the left chart is more skewed; its lower half runs too close to the red cluster, and in its upper half it runs too close to the green cluster. If it sees a test point that is a little farther out from the clusters, it will likely get the label wrong.

- The second chart on the right, however, maximizes distance from both the clusters while getting the training data separation right. By being right in the middle of the two clusters, it gives the data distributions for each class some "wiggle room", generalizing better on test data.





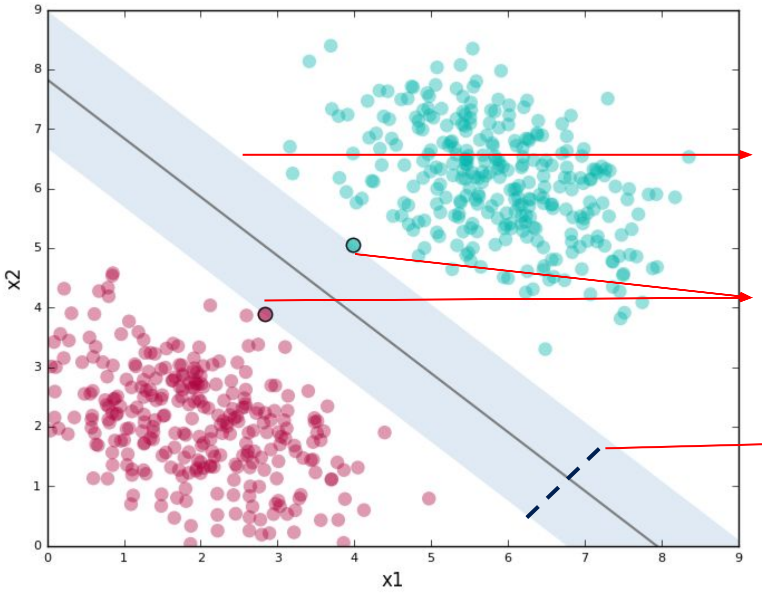
MECHANISM BEHIND MODEL: SUPPORT VECTORS

- SVMs try to find the second kind of line. We selected the better classifier visually, but let’s define the underlying philosophy more precisely to apply it in the general case. Here’s a simplified version of what SVMs do:

  1. Find lines that correctly classify the training data
  2. Among all such lines, ***pick the line with the greatest distance from the points closest to it.***

- The closest points that identify this line are known as **support vectors**, and the region they define around the line is known as **the margin**.

  We can visualise this using the right chart from the previous slide, this time with
  - the support vectors: points with black edges (there are two of them), and
  - the margin (the shaded region).



Lines adjacent to support vectors

These points are known as **support vectors**

The distance between 2 margins are known as **maximum margin**

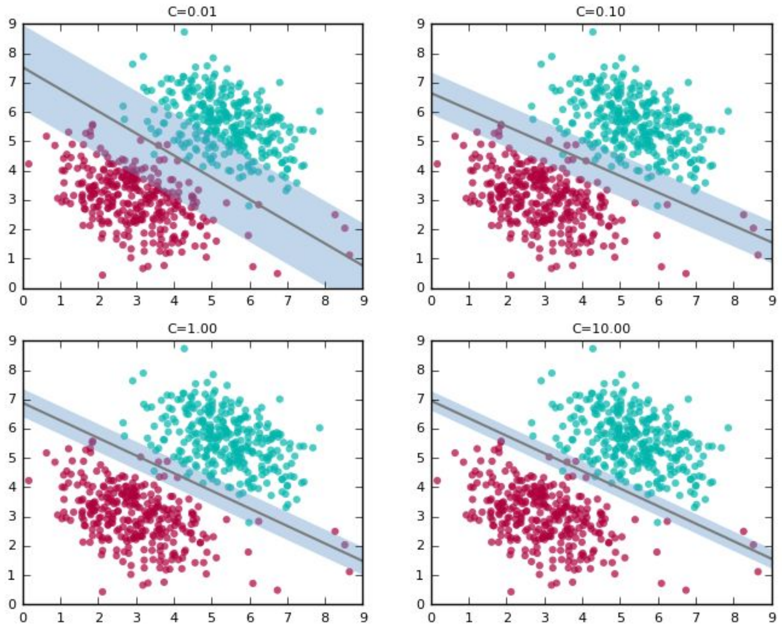
MECHANISM BEHIND MODEL: C VALUE

- Earlier, we looked at the easy case of **perfectly separable data**. However real-world data is typically messy.
  You will almost always have a few instances that a linear classifier can't get right.

- Clearly, if we are using a linear classifier, we are never going to be able to perfectly separate the labels. How do SVMs deal with this:

- They allow you to specify how many errors you are willing to accept. You can provide a parameter called "C" to your SVM; this allows you to dictate the **tradeoff between**:
  1. Correctly classifying training data. A high value of C focuses more on **having less errors on the training data**
  2. Having a wider margin, or "wiggle room" for future unseen data.

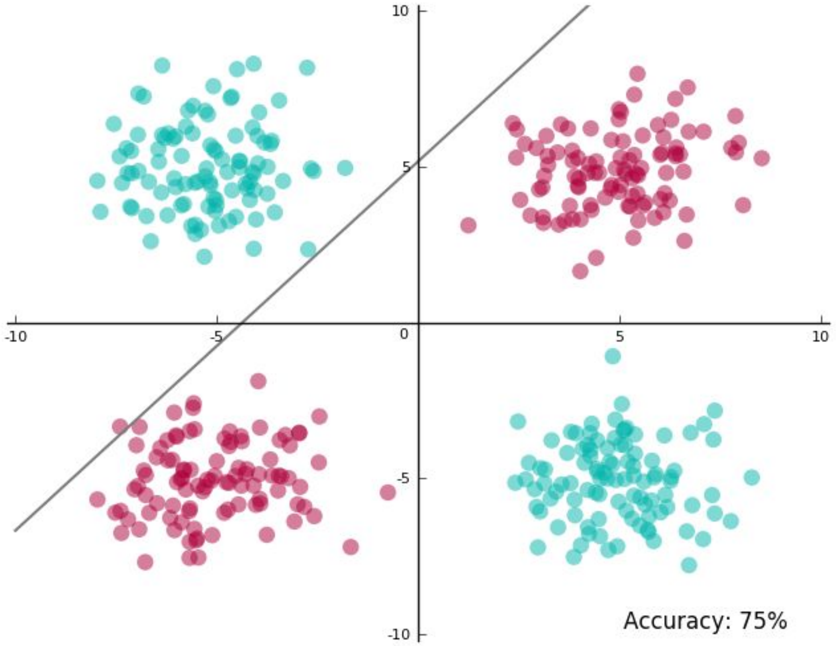
- While we strive to learn closely from trends in training data, there is always some risk of overfitting.. It bears repeating that this is a tradeoff.



The following plots show how the classifier and the margin vary as we increase the value of C. We will explore this through code and Sklearn later on.

MECHANISM BEHIND MODEL: KERNEL TRICK *(FUN FACT: SVM ISN'T ALWAYS LINEAR!)*

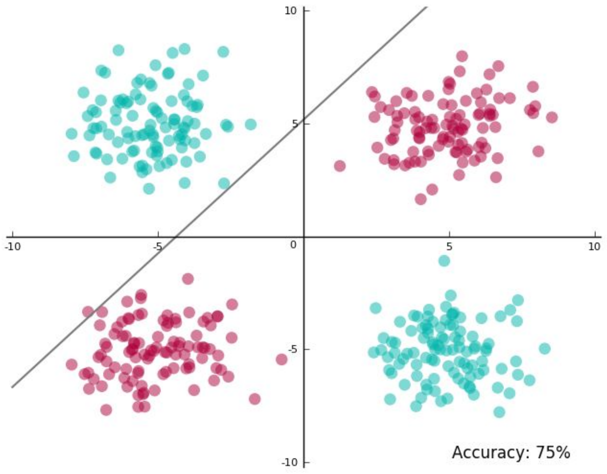
- Now what if your data is non-linear, like our image below?
  - There is a method known as a **kernel trick.** This involves applying mathematical functions such as polynomial functions to elevate the data to a higher dimensionality, such that it becomes easier to draw a linear plane.
  - **Video Demonstration: https://www.youtube.com/watch?time_continue=2&v=3liCbRZPrZA**
  - There are many kernel types, but these are the popular ones:
    - Polynomial Kernel
    - Radial Basis Function Kernel
    - Sigmoid Kernel



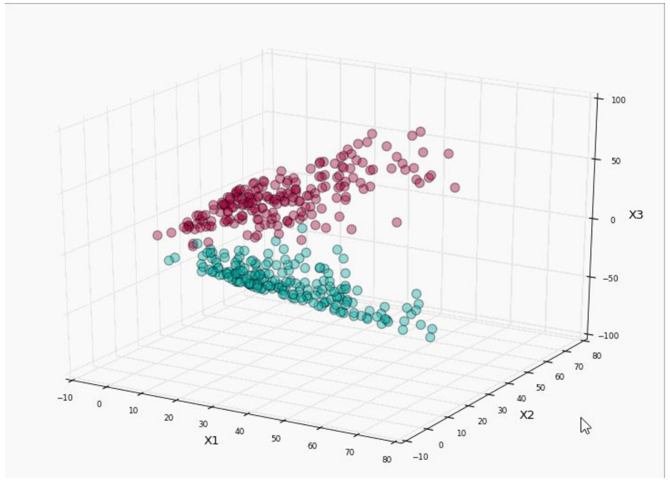
Here you can see that no matter how we cut the data points, we are always going to end up with very high misclassification rates

MECHANISM BEHIND MODEL: KERNEL TRICK *(FUN FACT: SVM ISN'T ALWAYS LINEAR!)*

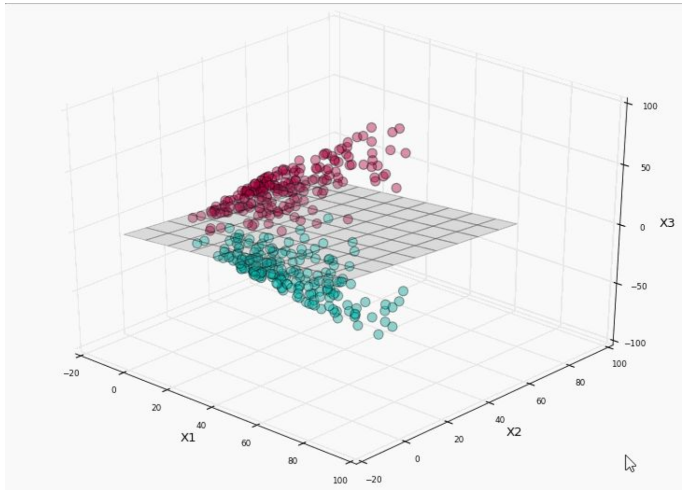
- Let's apply kernel trick to our data!



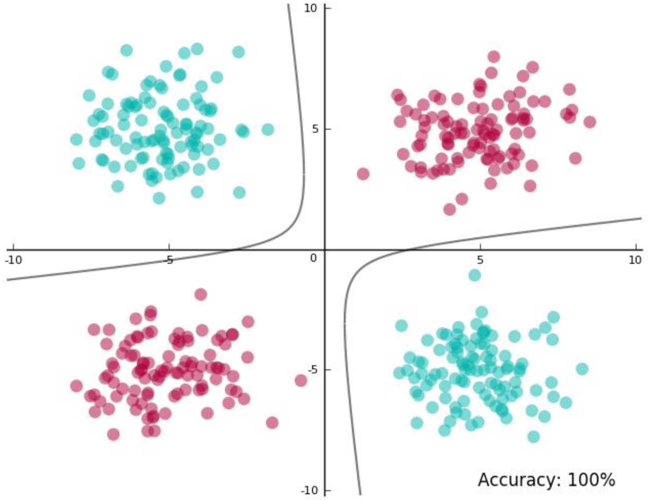
Apply kernel trick to elevate the data to a high dimensionality



Now do you see a space where you can slip a plane?



Map back the plane found in high dimensionality to the original dimensionality



MECHANISM BEHIND MODEL: KERNEL TRICK *(FUN FACT: SVM ISN'T ALWAYS LINEAR!)*

- When you map it back to the original space, the separating boundary is not a line anymore. This is also true for the margin and support vectors. As far as our visual intuition goes, they make sense in the projected space.

- Take a look at what they look like in the projected space, and then in the original space. The 3D margin is the region (not shaded to avoid visual clutter) between the planes above and below the separating hyperplane