Title: Comparing Classical and Quantum Methods for Supervised Machine Learning

Date: Aug 08, 2016 09:35 AM

URL: http://pirsa.org/16080001

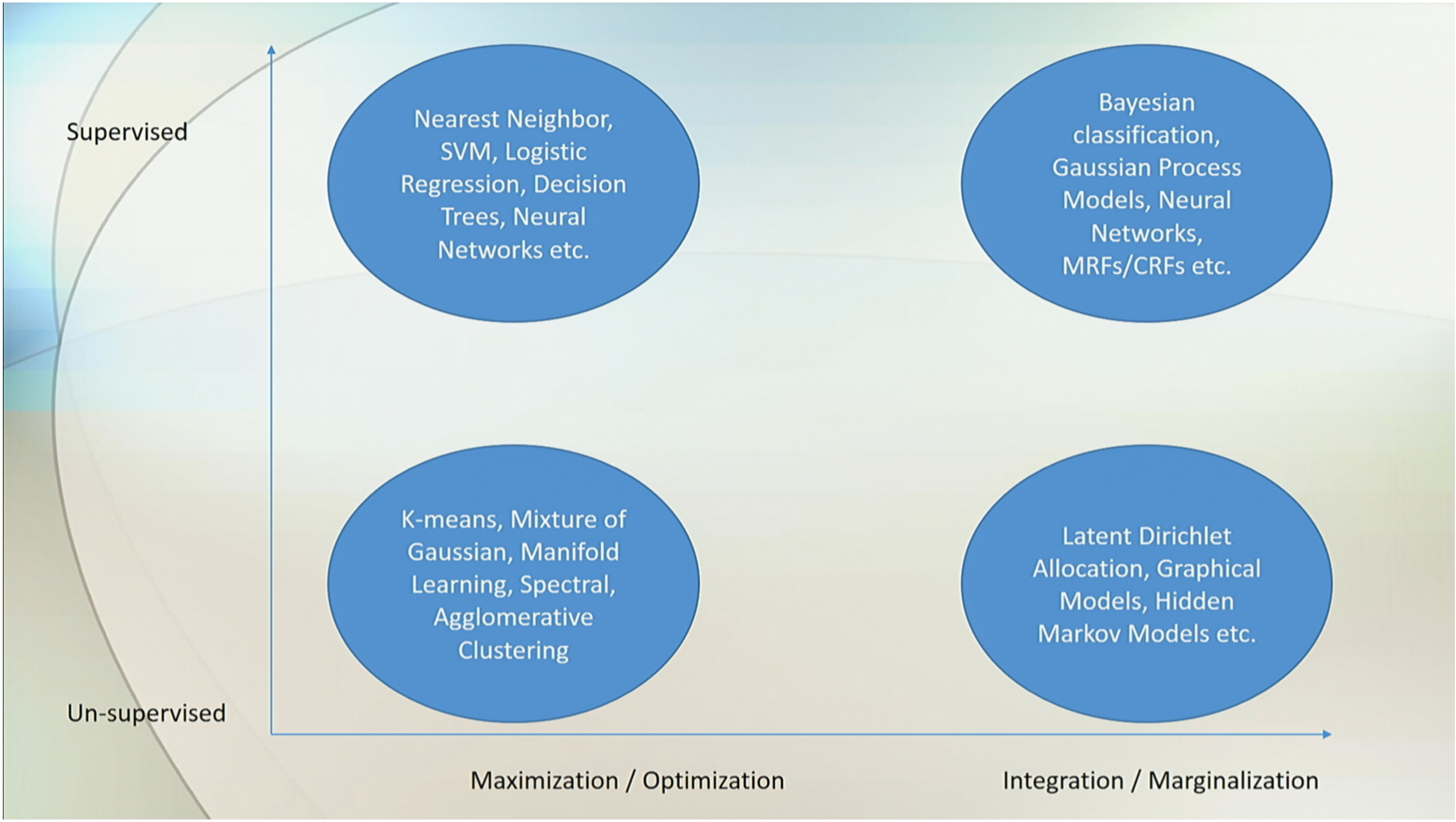
Abstract: Supervised Machine Learning is one of the key problems that arises in modern big data tasks. In this talk, I will first describe several different classical algorithmic paradigms for classification and then contrast them with quantum algorithmic constructs. In particular, we will look at classical methods such as the nearest neighbor rule, optimization based algorithms (e.g. SVMs), Bayesian inference based techniques (e.g. Bayes point machine) and provide a unifying framework so that we can get a deeper understanding about the quantum versions of the methods.

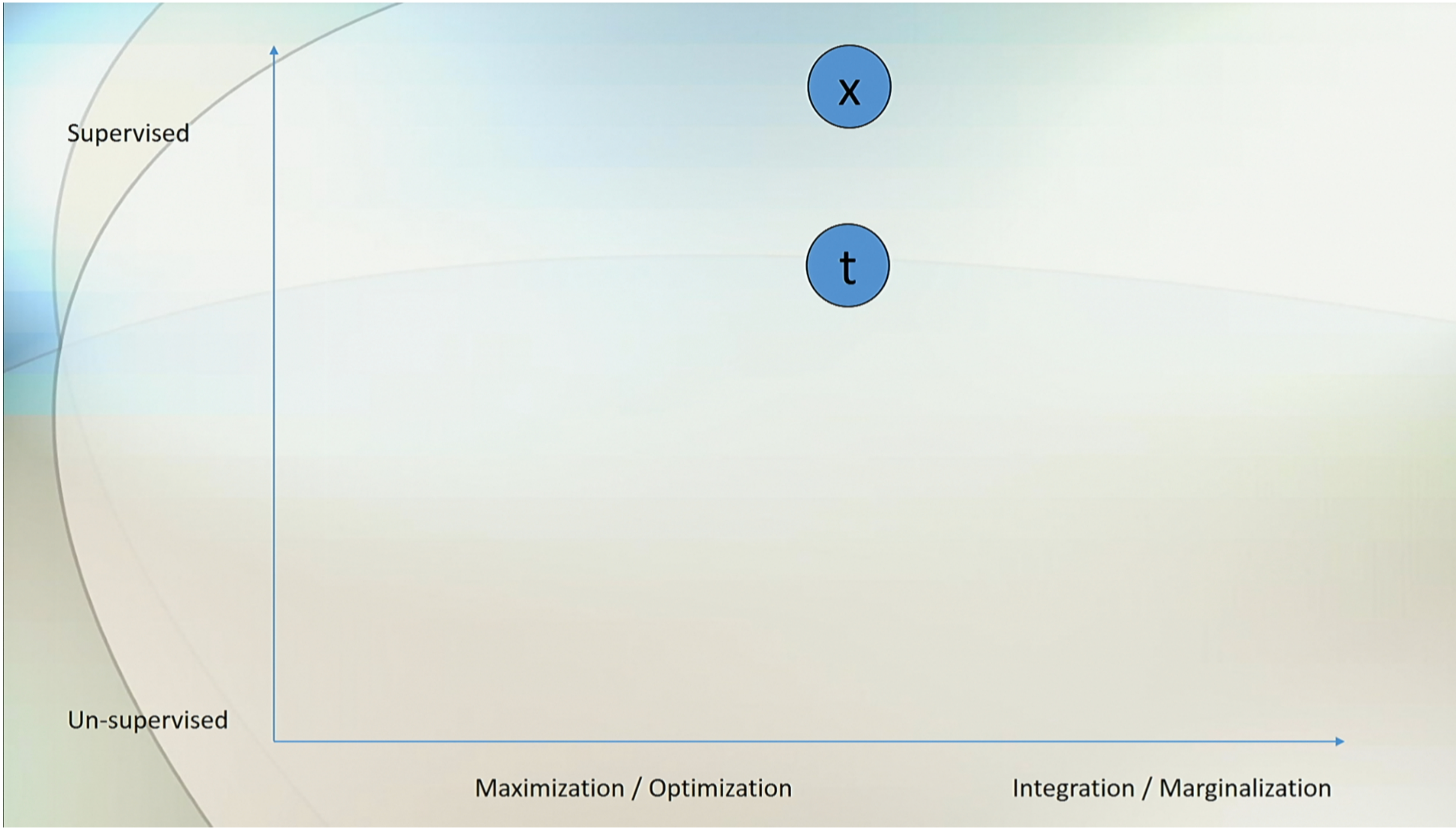
# Supervised Machine Learning

# Classical vs. Quantum

**Ashish Kapoor**
Microsoft Research

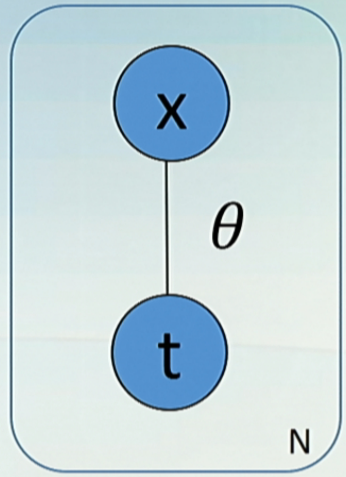
Joint work with Nathan Wiebe and Krysta Svore


Supervised
Nearest Neighbor, SVM, Logistic Regression, Decision Trees, Neural Networks etc.
Bayesian classification, Gaussian Process Models, Neural Networks, MRFs/CRFs etc.
K-means, Mixture of Gaussian, Manifold Learning, Spectral, Agglomerative Clustering
Latent Dirichlet Allocation, Graphical Models, Hidden Markov Models etc.
Un-supervised
Maximization / Optimization
Integration / Marginalization

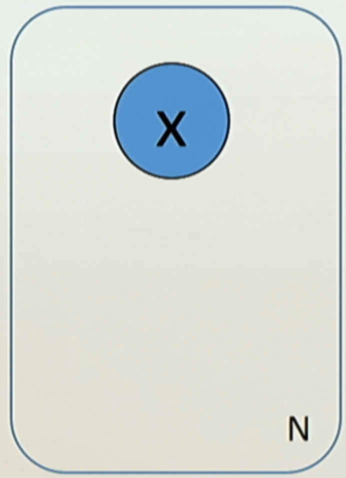

x
t
Supervised
Un-supervised
Maximization / Optimization
Integration / Marginalization

Supervised

$$\theta^* = \arg\max_\theta Fitness(X,T;\theta)$$



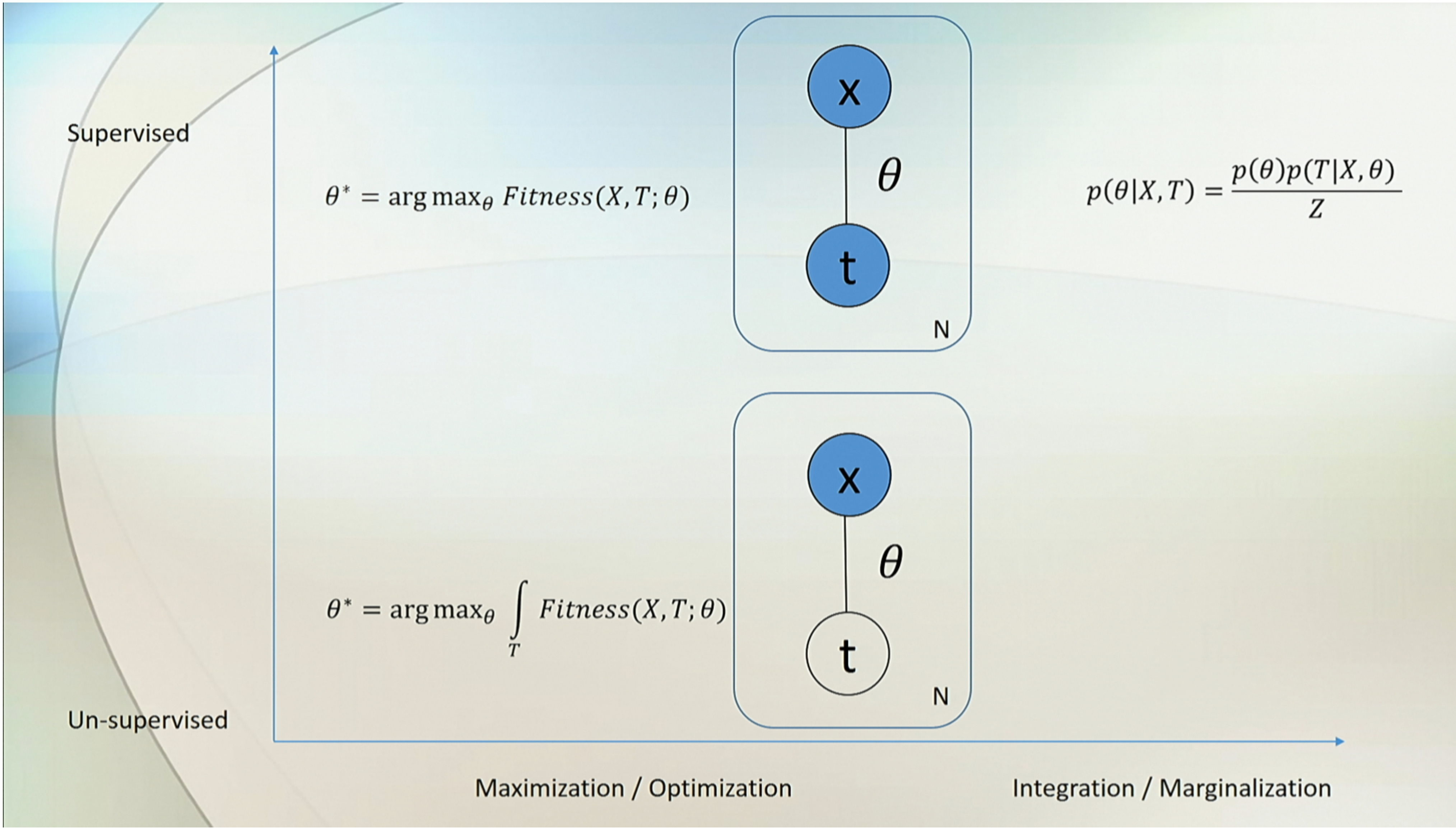

$$p(\theta|X,T) = \frac{p(\theta)p(T|X,\theta)}{Z}$$




Un-supervised

Maximization / Optimization

Integration / Marginalization


Supervised
$\theta^* = \arg\max_\theta Fitness(X,T;\theta)$
x
$\theta$
t
N
$p(\theta|X,T) = \frac{p(\theta)p(T|X,\theta)}{Z}$
$\theta^* = \arg\max_\theta \int_T Fitness(X,T;\theta)$
x
$\theta$
t
N
Un-supervised
Maximization / Optimization
Integration / Marginalization

# Supervised Machine Learning == Learning a Function

- Given: $D = \{(\mathbf{x}_i, t_i)\}_{i=1}^{n}$
- Learn: $t = f(\mathbf{x})$
- Includes, Classification, Regression, Ranking etc.

# Supervised Machine Learning == Learning a Function

- Given: $D = \{(\mathbf{x}_i, t_i)\}_{i=1}^{n}$
- Learn: $t = f(\mathbf{x})$
- Includes, Classification, Regression, Ranking etc.

# Supervised Learning – Nearest Neighbor

$$f(\mathbf{x}) = t_i \ \text{ s.t. } i = \arg\min \text{Distance}(\mathbf{x}, \mathbf{x}_i)$$

# Supervised Learning – Nearest Neighbor

$$f(\mathbf{x}) = t_i \ \text{ s.t. } i = \arg\min \text{Distance}(\mathbf{x}, \mathbf{x}_i)$$

- Two key ingredients
  - Assumptions about function being learnt
    - Function values at nearby/similar points are same
  - Observed Data
    - Determining the actual values

# Performance Metrics

# Performance Metrics

- Computational
  - How hard to train?
  - How hard to test?

# Performance Metrics

- Computational
  - How hard to train?
  - How hard to test?
- Statistical
  - How much training data?

# Performance Metrics

- Computational
  - How hard to train?
  - How hard to test?
- Statistical
  - How much training data?
- Classification Accuracy
  - How well do you do on the task of interest?

# Performance Metrics

***Most Quantum ML methods attempt to improve upon computational efficiency:***

***NN -> Quantum NN***

***Regularized Least Square -> Quantum RLS***

***<u>What about statistical efficiency?</u>***

## Numerical Methods - Optimization

## Numerical Methods - Integration

# Numerical Methods - Optimization

$$\theta^* = \arg\max_\theta Fitness(X, T; \theta)$$

# Numerical Methods - Integration

# Numerical Methods - Optimization

$$\theta^* = \arg\max_\theta Fitness(X, T; \theta)$$

e.g. Support Vector Machines, Regularized Least Square, Logistic Regression etc.

# Numerical Methods - Integration

# Numerical Methods - Optimization

$$\theta^* = \arg\max_\theta Fitness(X, T; \theta)$$

e.g. Support Vector Machines, Regularized Least Square, Logistic Regression etc.

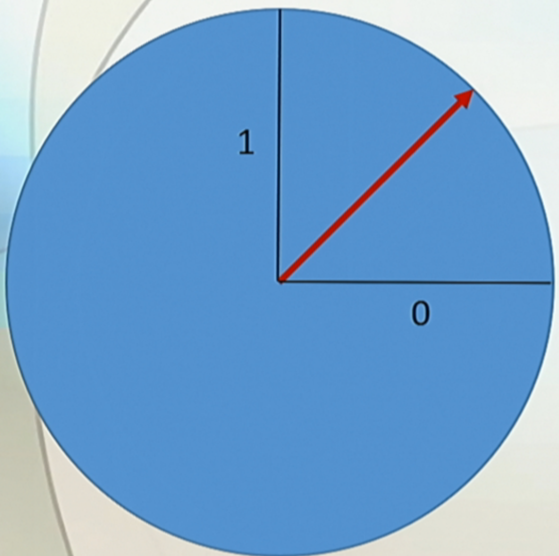
# Numerical Methods - Integration

$$p(\theta|X, T) = \frac{p(\theta)p(T|X, \theta)}{Z}$$

# ~~Quantum Machine Learning~~
# Machine Learning as Rotations

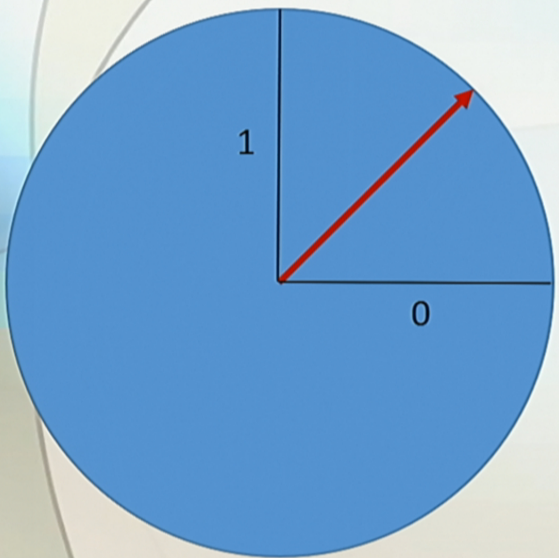
**Ashish Kapoor**
Microsoft Research

# Structure of quantum algorithms

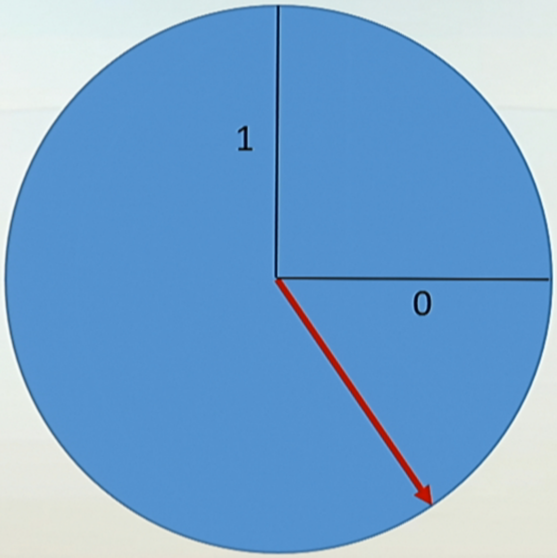



Prepare quantum state vector
(2 dimensional subspace shown here)

# Structure of quantum algorithms

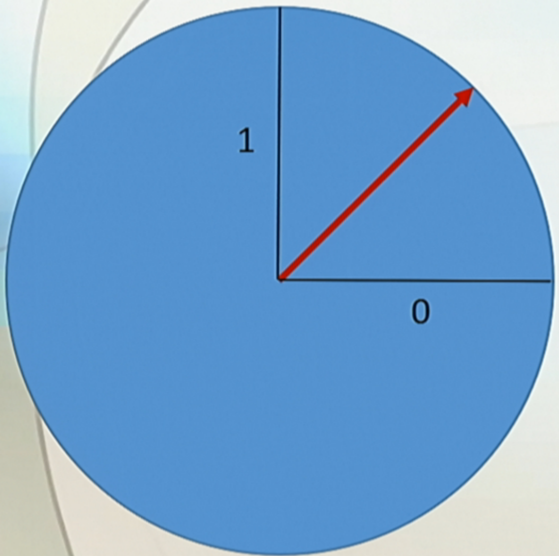



Prepare quantum state vector
(2 dimensional subspace shown here)

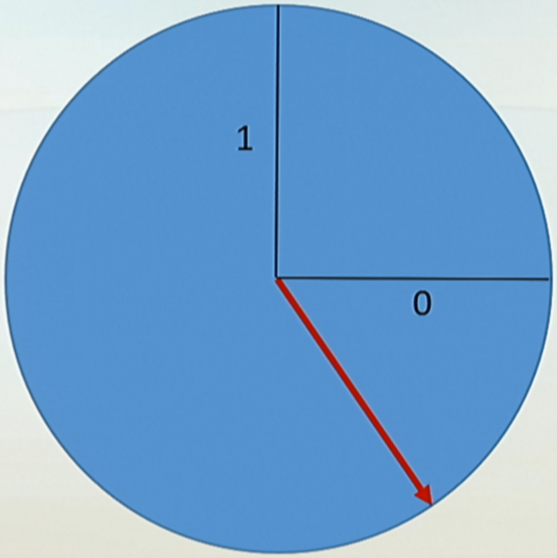



Transform state vector using
quantum gates

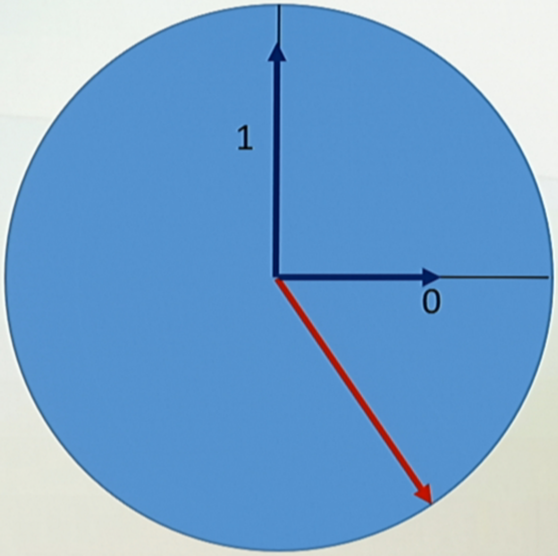
# Structure of quantum algorithms




Prepare quantum state vector
(2 dimensional subspace shown here)




Transform state vector using
quantum gates




Measure result

# Example: Grover's search

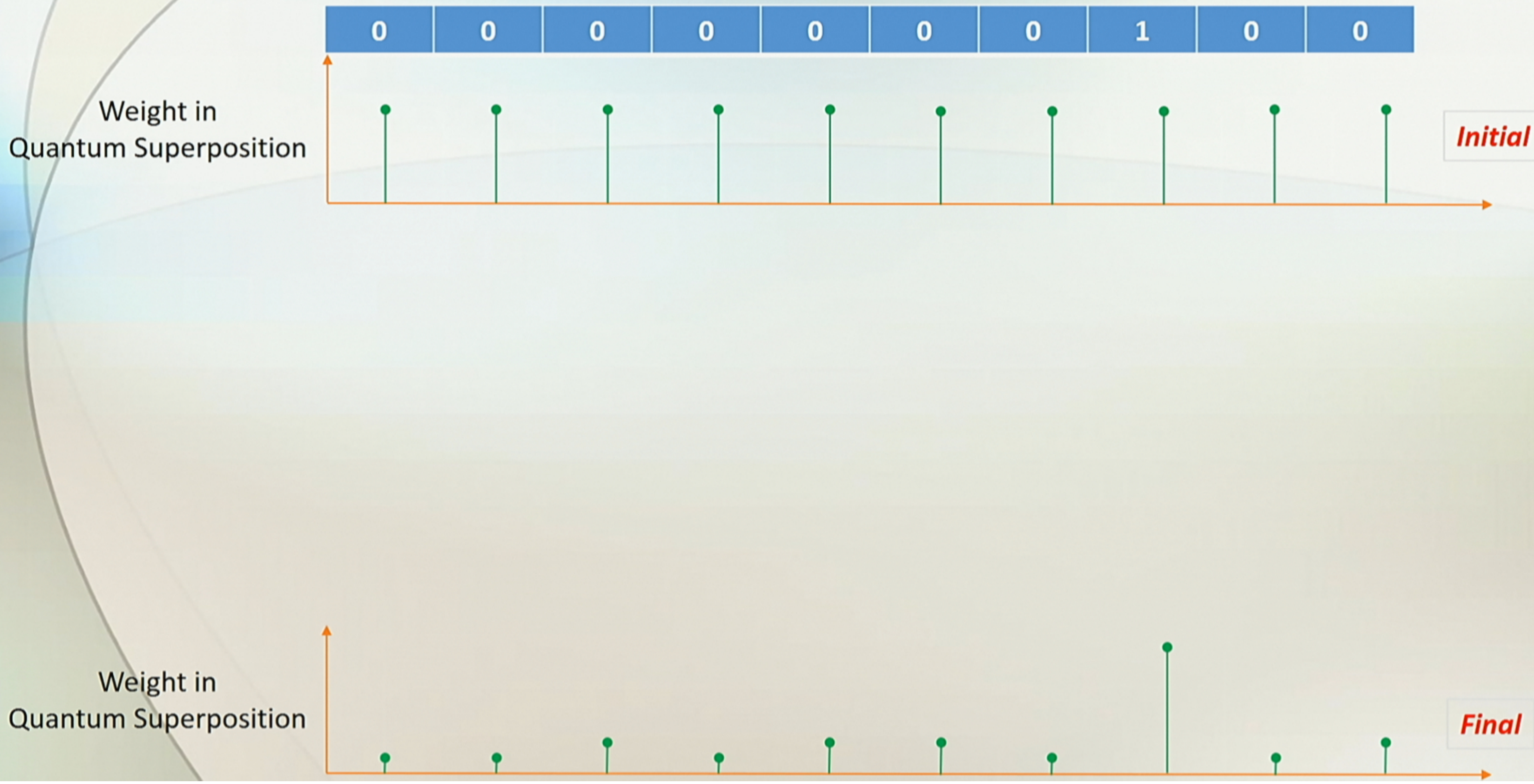
Problem: You have a list of $N$ numbers and one of them satisfies f(x) = 1.
Find that index.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| f(x) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

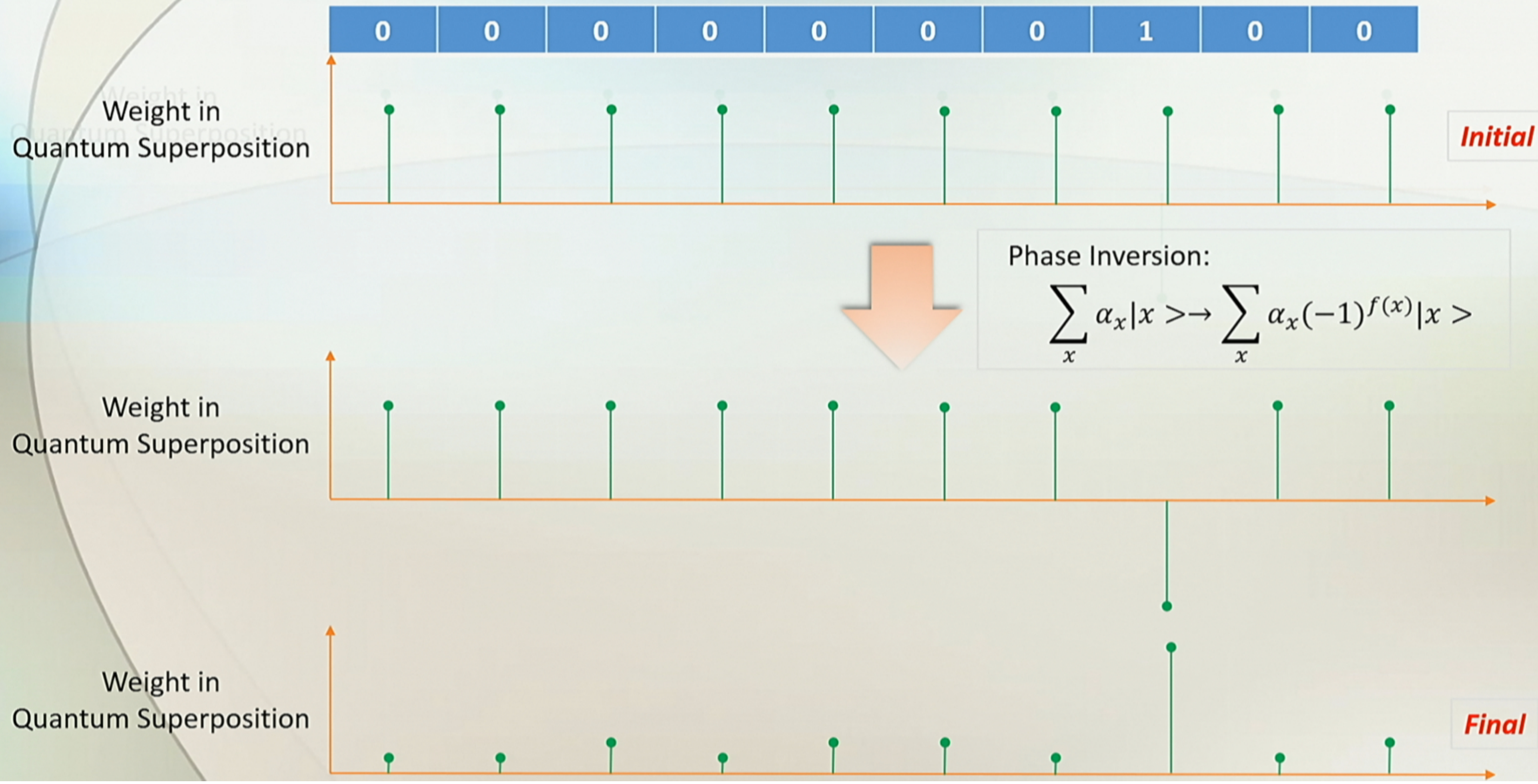
Classical: $O(N)$ queries needed to find it.

Quantum: $O(\sqrt{N})$queries needed to find it.

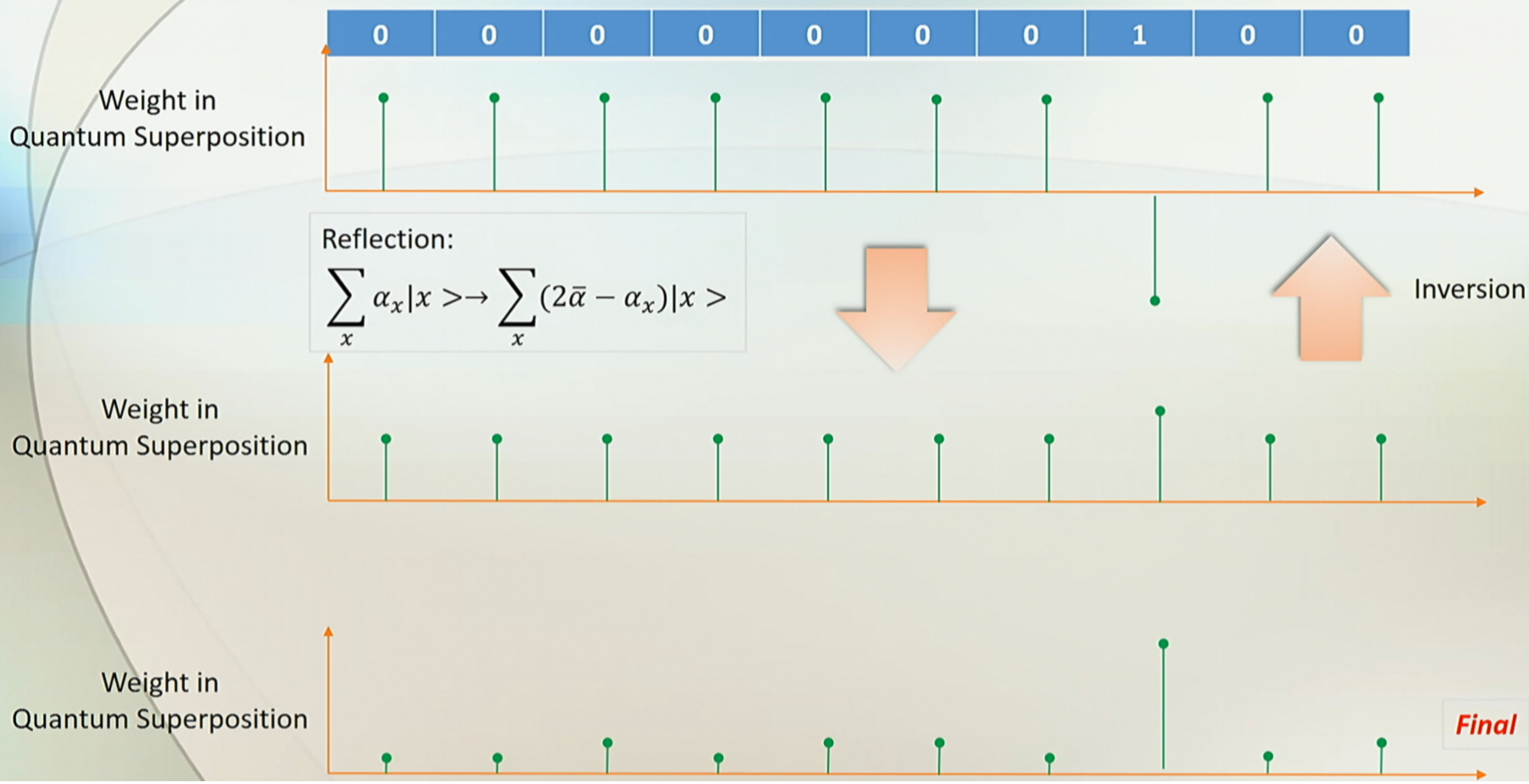
# Example: Grover's search



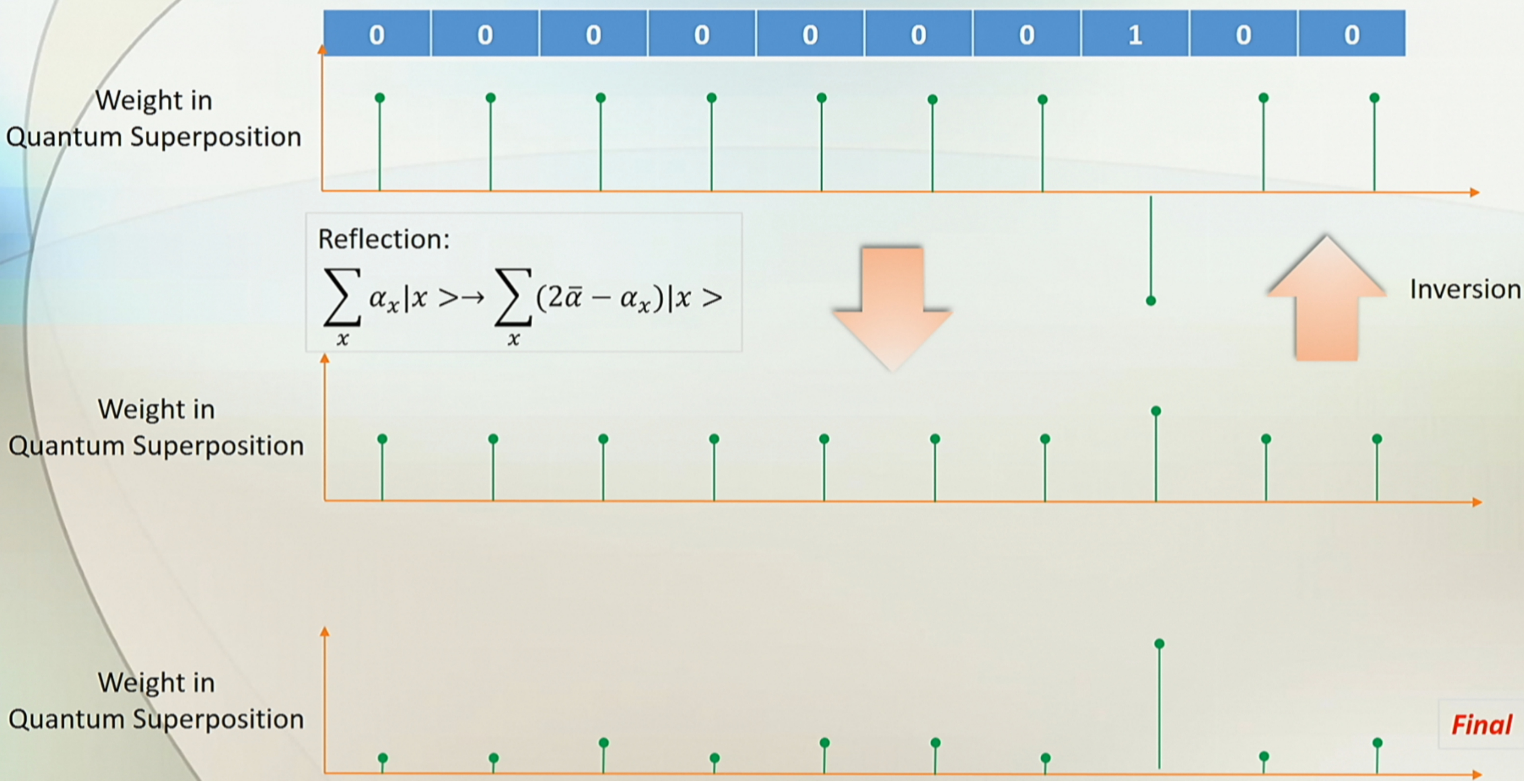
# Example: Grover's search



# Example: Grover's search



# Example: Grover's search



# Perceptron

Given separable training data: $\{(x_1, y_1), \ldots, (x_n, y_n)\}$

# Perceptron

Given separable training data: $\{(x_1, y_1), .., (x_n, y_n)\}$

Learn $w$ such that $y = sign(w^T x)$

# Perceptron

Given separable training data: $\{(x_1, y_1), .., (x_n, y_n)\}$

Learn $w$ such that $y = sign(w^T x)$

Step 1: Initialize $w = 0$

Step 2: For $i = 1\ to\ n$

If correctly classify $x_i$

do nothing

Else

$w = w + y_i . x_i$

# Perceptron

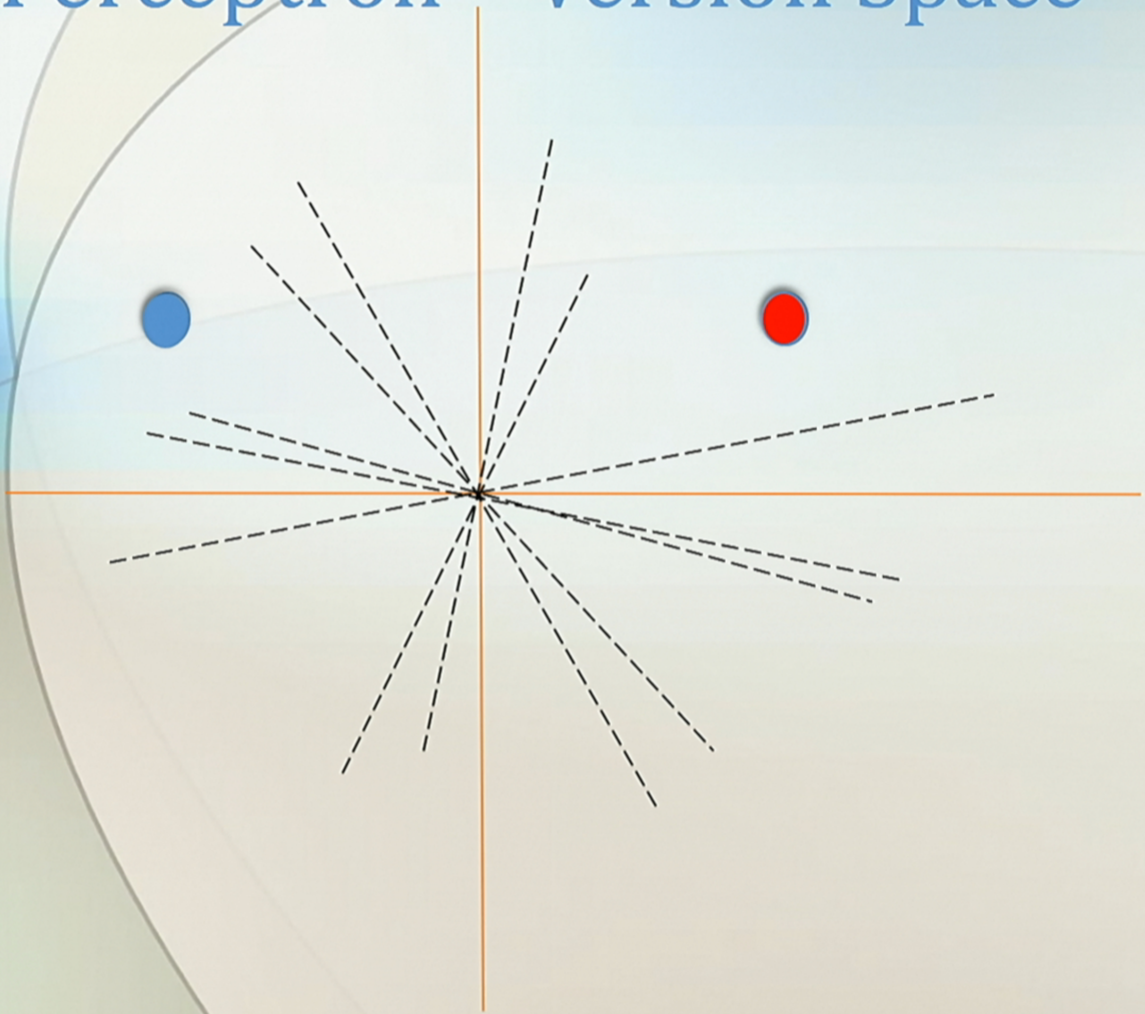
Given separable training data: $\{(x_1, y_1), .., (x_n, y_n)\}$

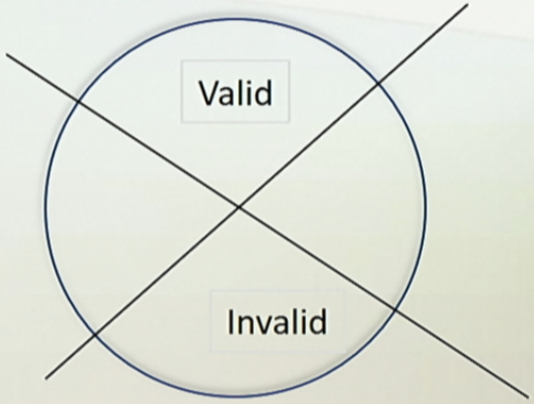
Learn $w$ such that $y = sign(w^T x)$

Step 1: Initialize $w = 0$
Step 2: For $i = 1\ to\ n$
If correctly classify $x_i$
do nothing
Else
$w = w + y_i . x_i$

**Block 1962, Novikoff 1962: If the margin is $\gamma$ then number of operations: $O(\frac{1}{\gamma^2})$**

# Perceptron – Version Space



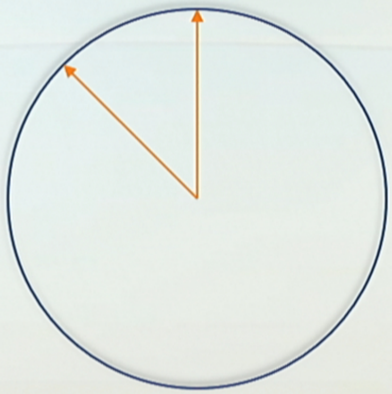



Unit Circle

All Possible Classifiers

# Perceptron – ML as Search

Step 1: Generate hypothesis



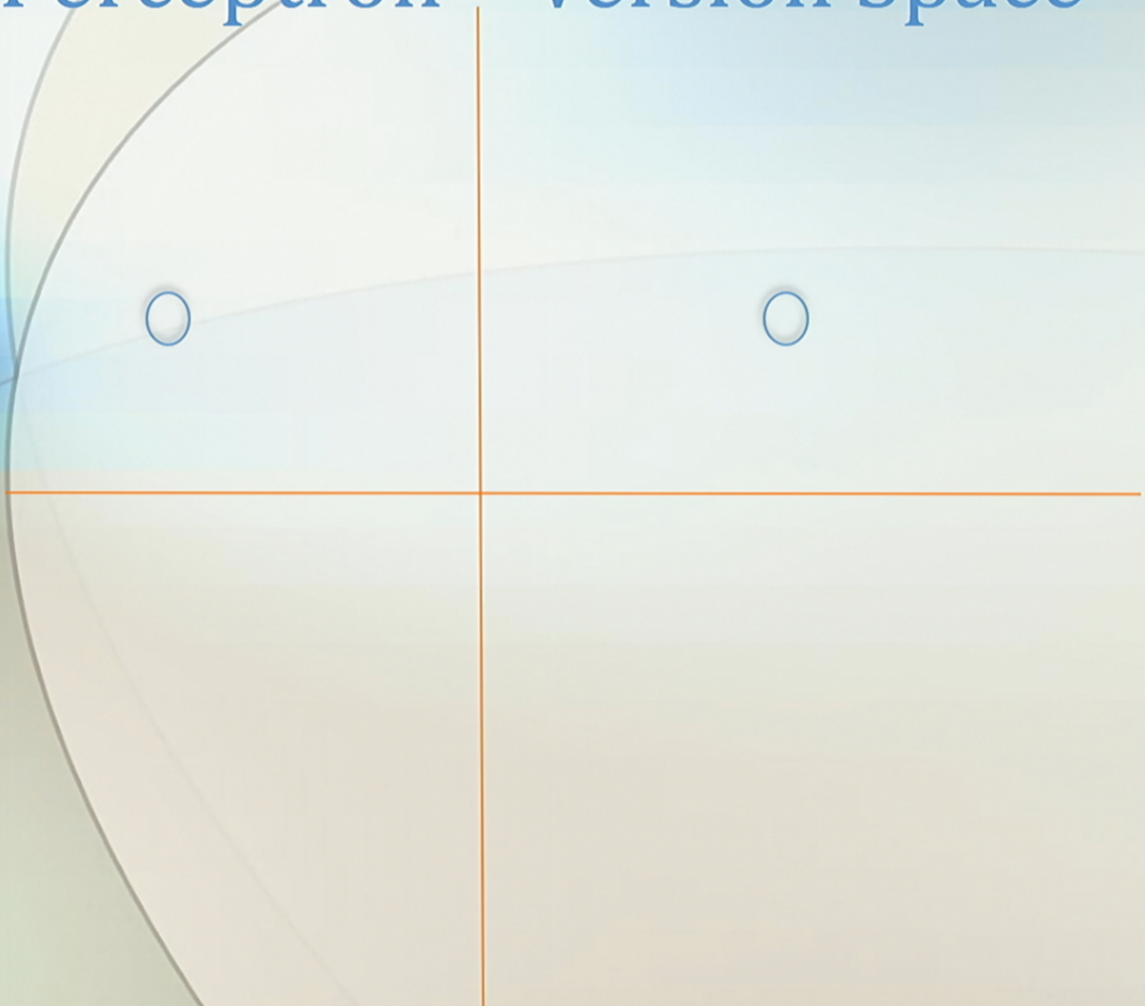
# Proof Sketch:

Search Space is $O(\frac{1}{\gamma})$

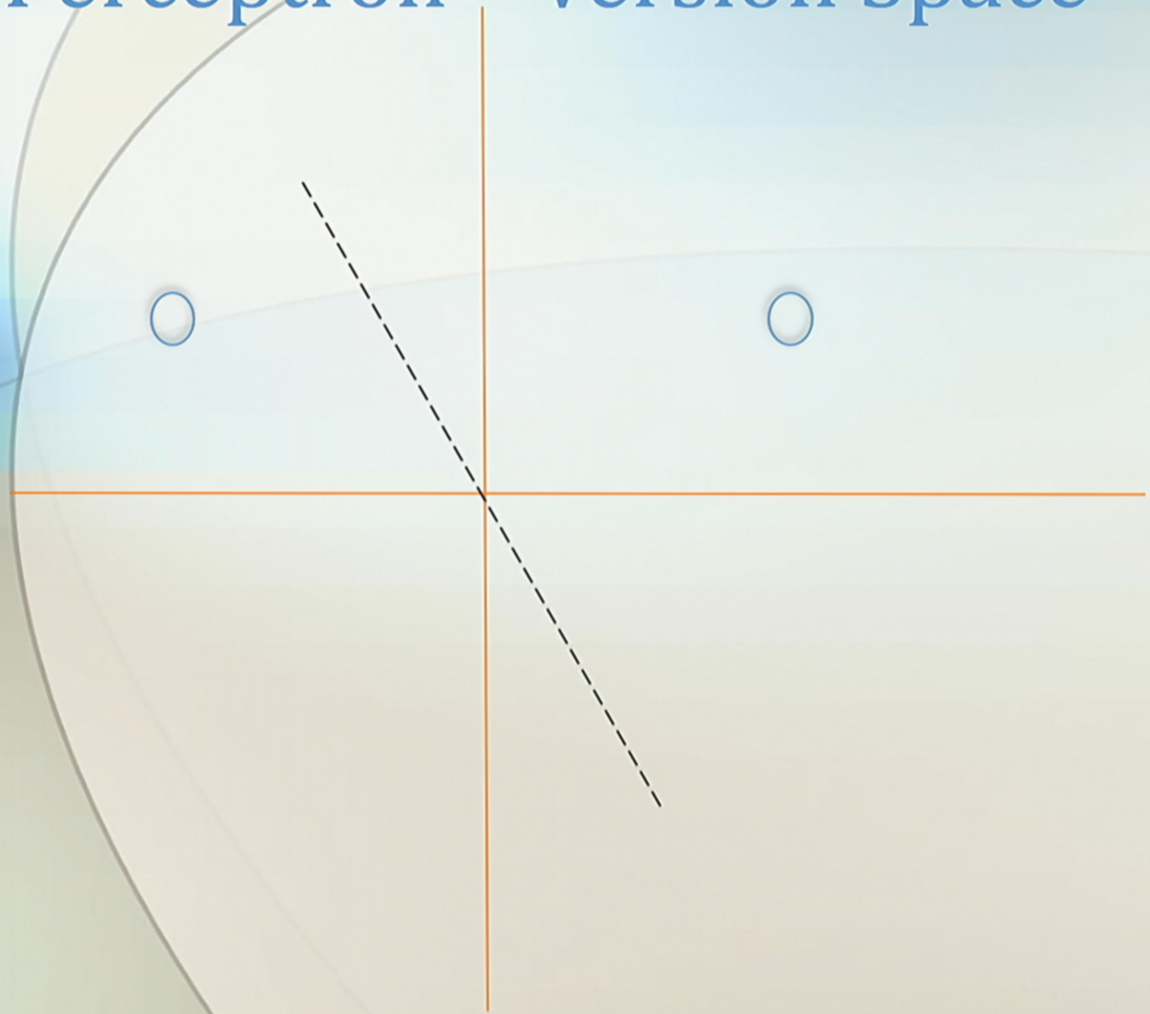
# Perceptron – Version Space

# Perceptron – Version Space

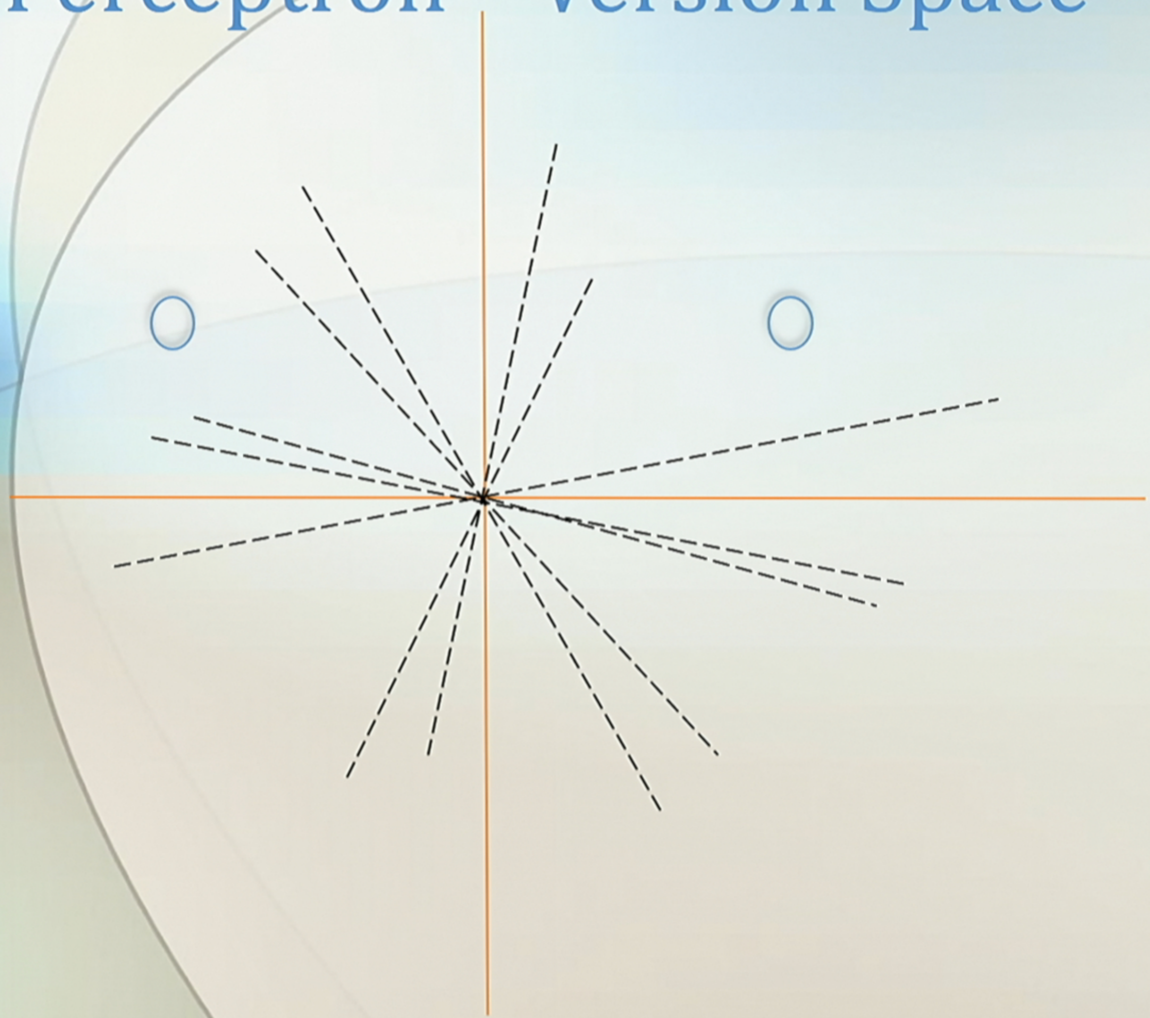
# Perceptron – Version Space



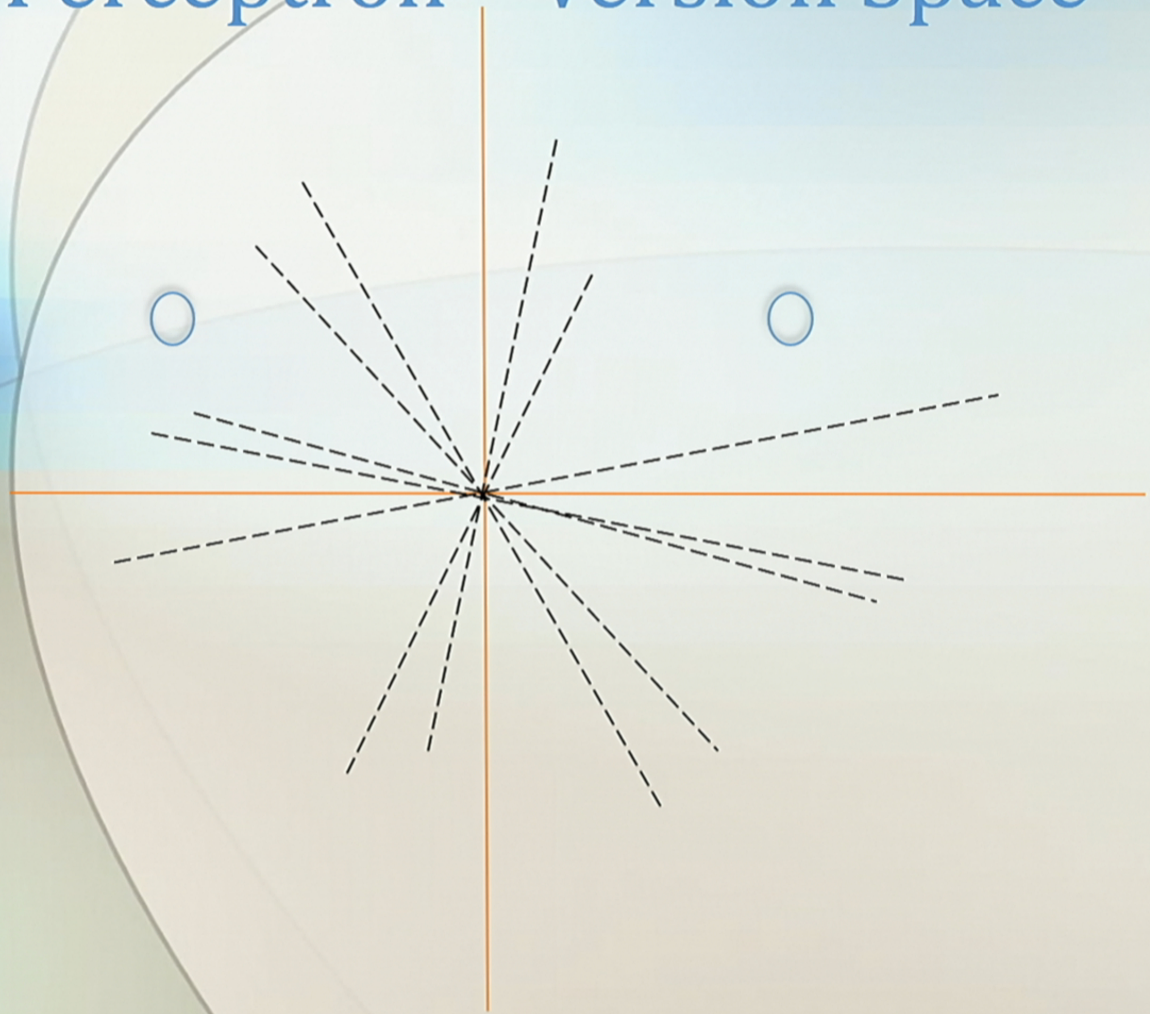
# Perceptron – Version Space

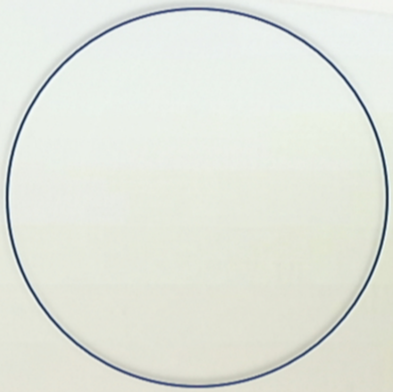


# Perceptron – Version Space



# Perceptron – Version Space

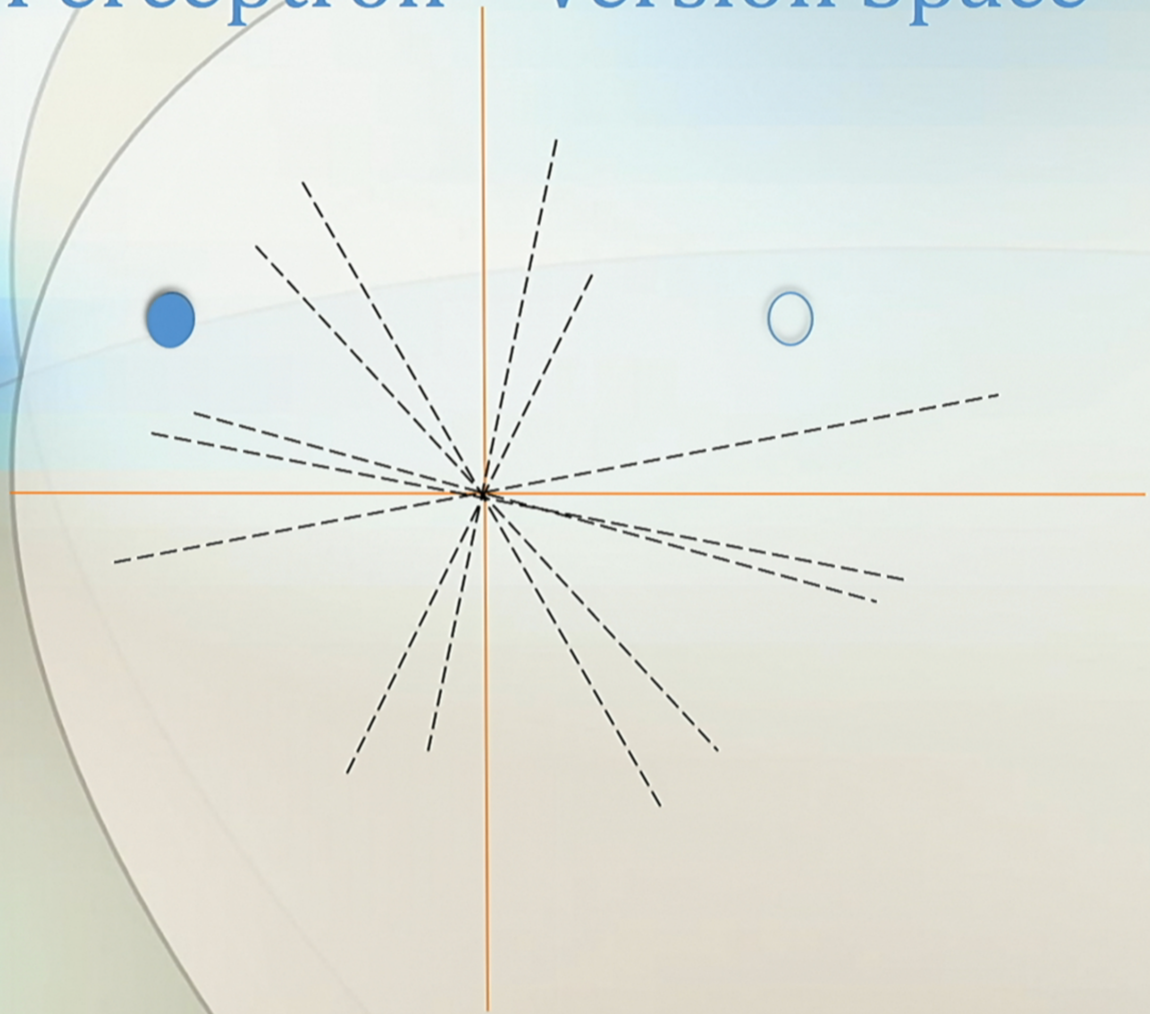


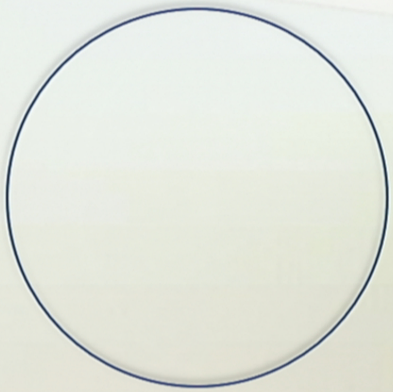


Unit Circle

All Possible Classifiers

# Perceptron – Version Space

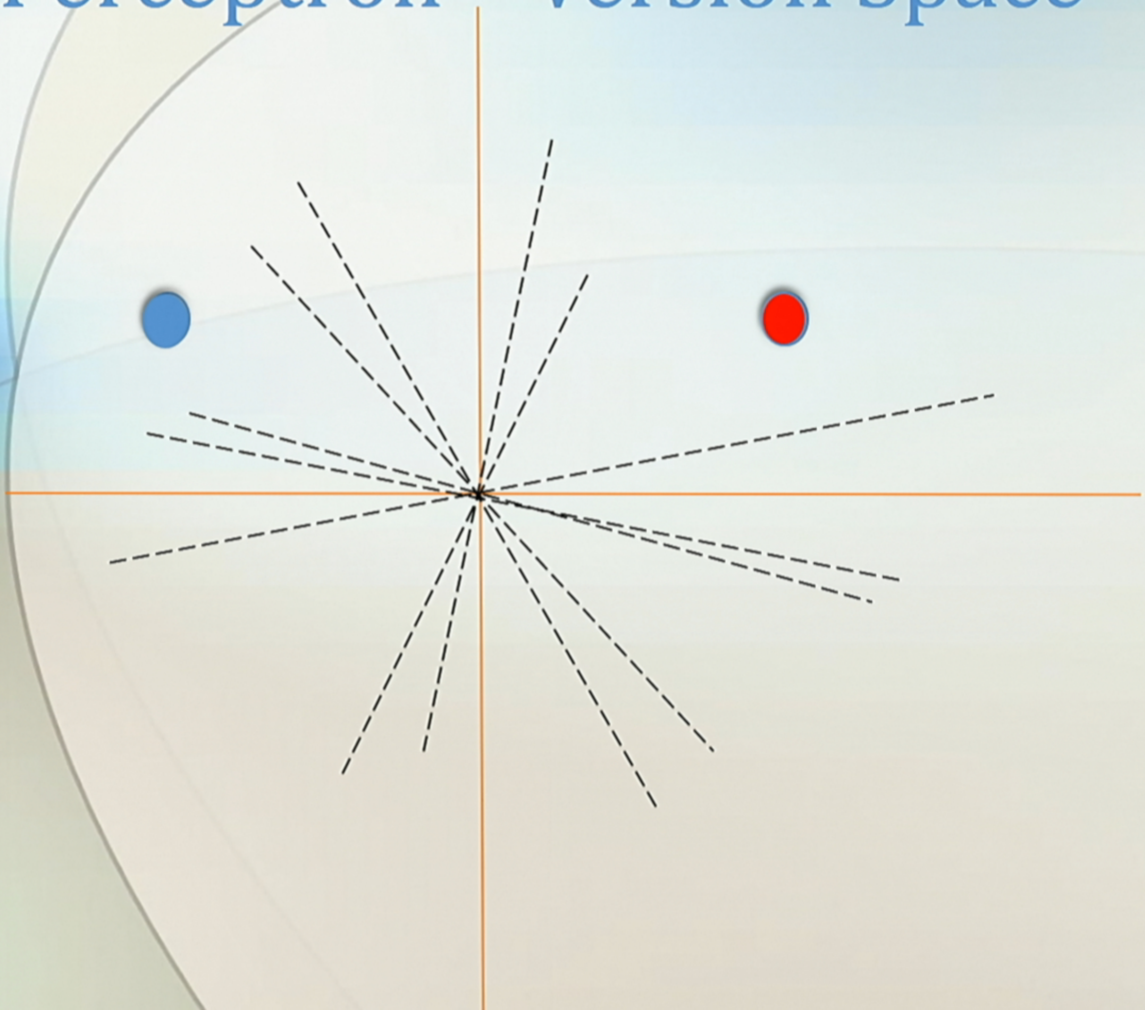


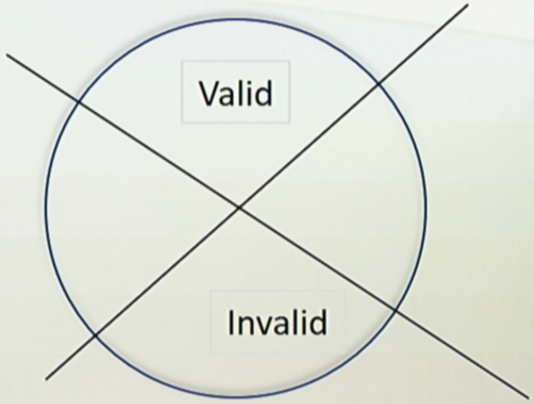


Unit Circle

All Possible Classifiers

# Perceptron – Version Space

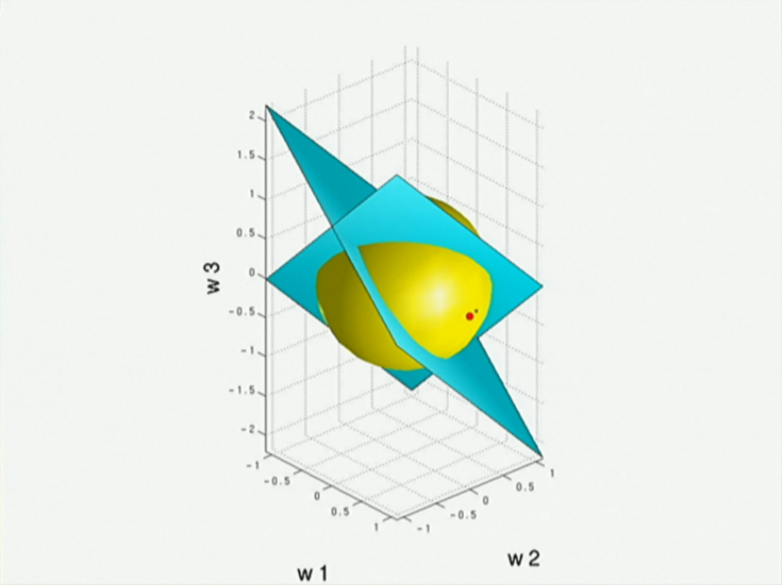





Unit Circle

All Possible Classifiers

# Perceptron – Version Space



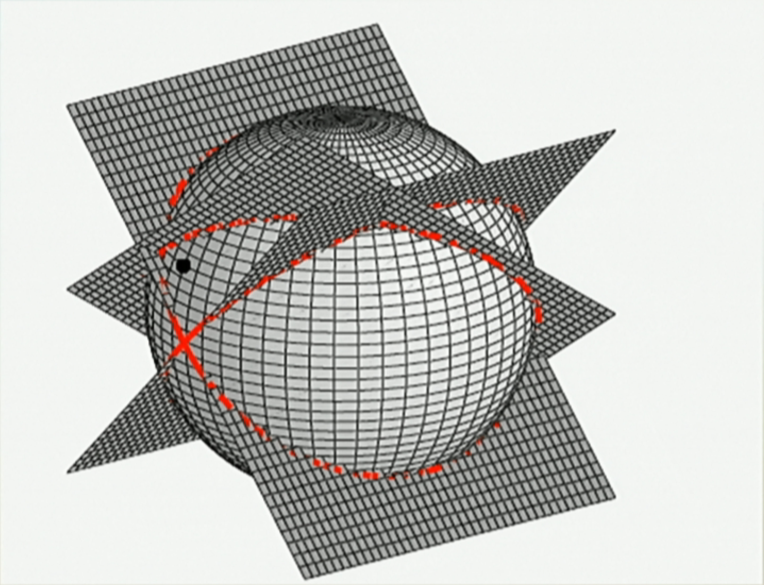
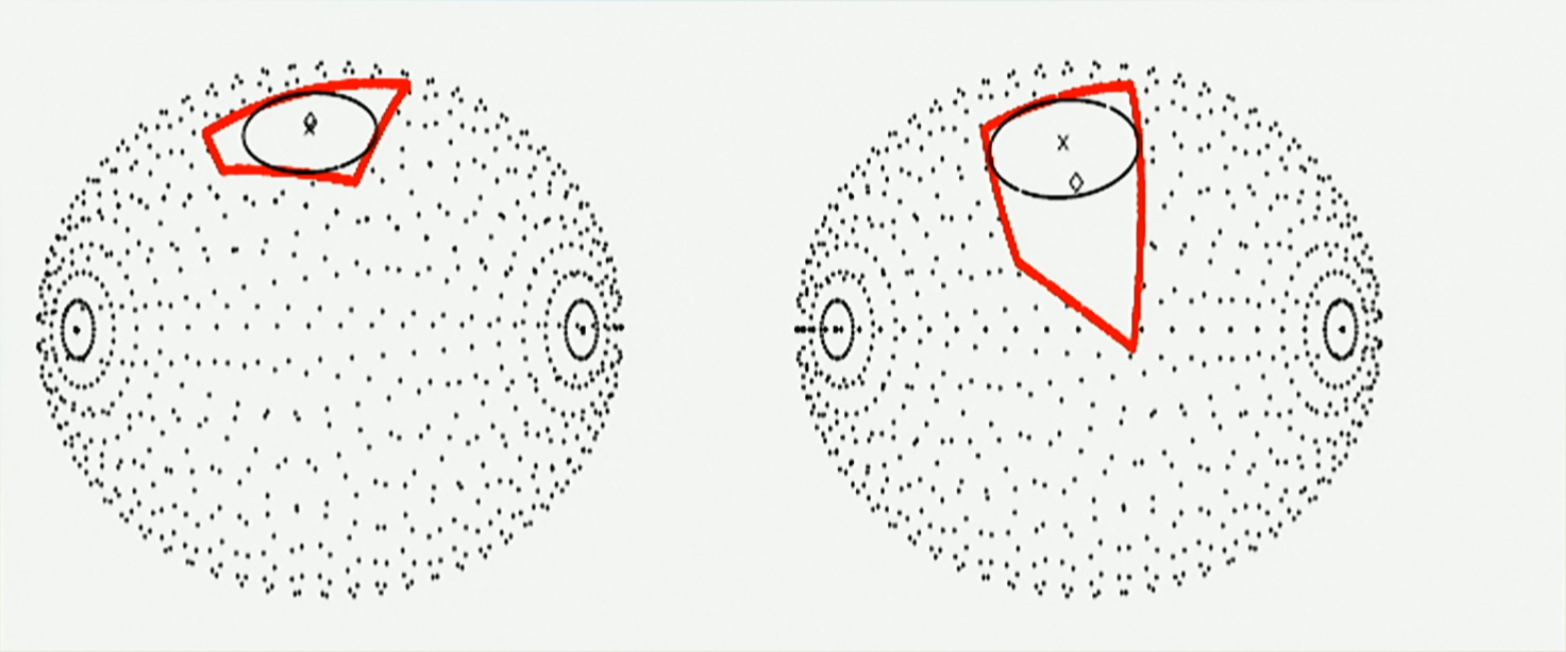



Figure from Herbrich et al. 2001

# Perceptron – Version Space

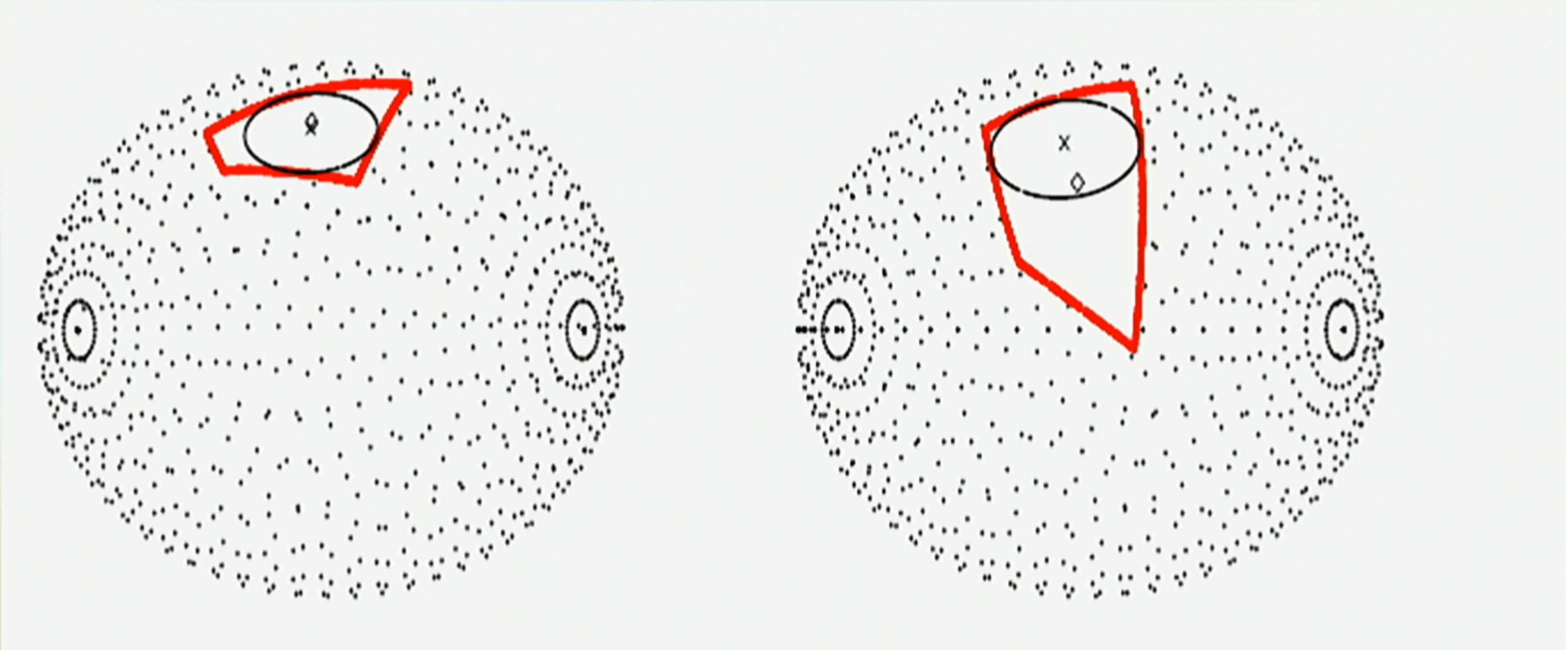


**SVM is the center of maximally inscribable ball in the version space**

**Radius of the ball = Margin!**

Figure from Herbrich et al. 2001

# Perceptron – Version Space

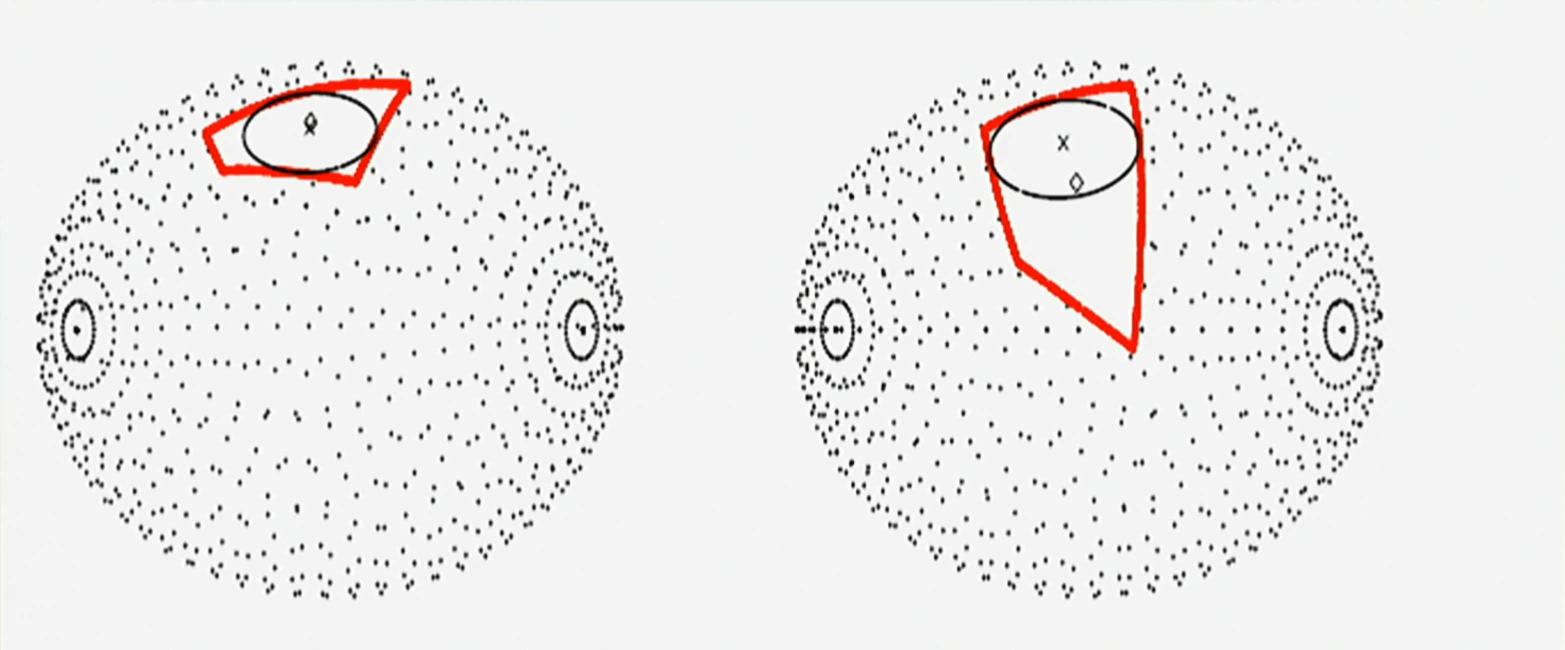


**SVM is the center of maximally inscribable ball in the version space**

**Radius of the ball = Margin!**

Figure from Herbrich et al. 2001

# Perceptron – Version Space



**SVM is the center of maximally inscribable ball in the version space**

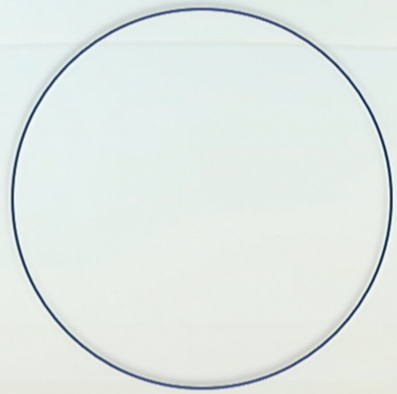
**Radius of the ball = Margin!**

Figure from Herbrich et al. 2001

# Perceptron – ML as Search

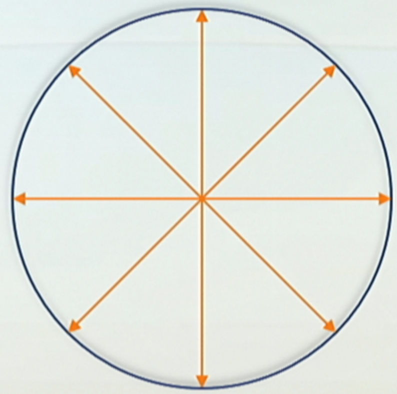
# Perceptron – ML as Search

Step 1: Generate hypothesis



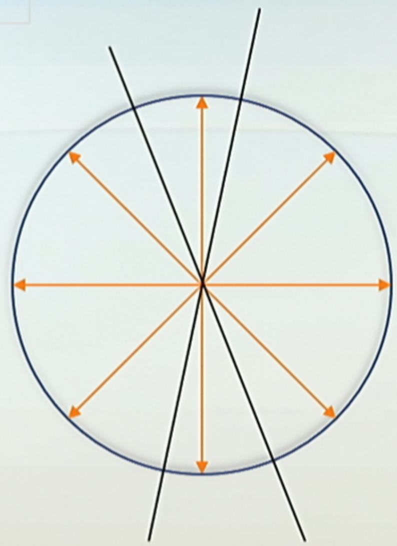
# Perceptron – ML as Search

Step 1: Generate hypothesis



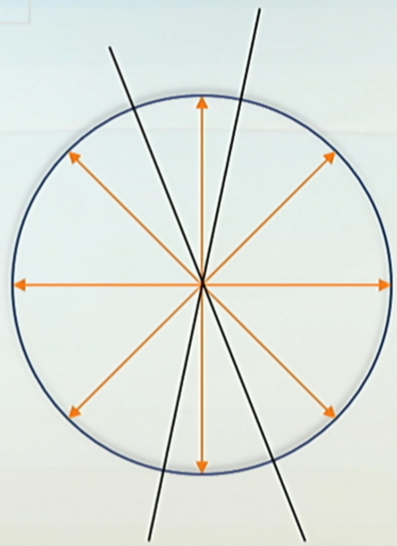
# Perceptron – ML as Search

Step 1: Generate hypothesis



# Perceptron – ML as Search

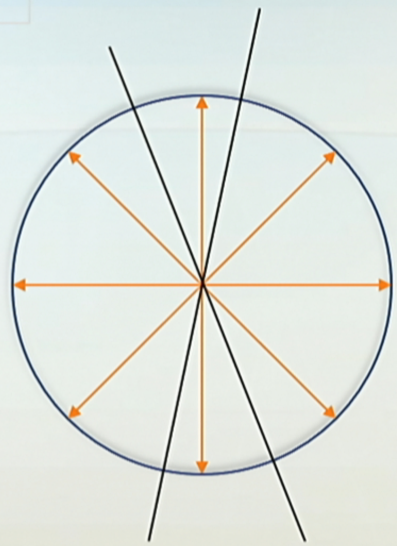
Step 1: Generate hypothesis



Search Space is $O(\frac{1}{\gamma})$

# Perceptron – ML as Search

Step 1: Generate hypothesis

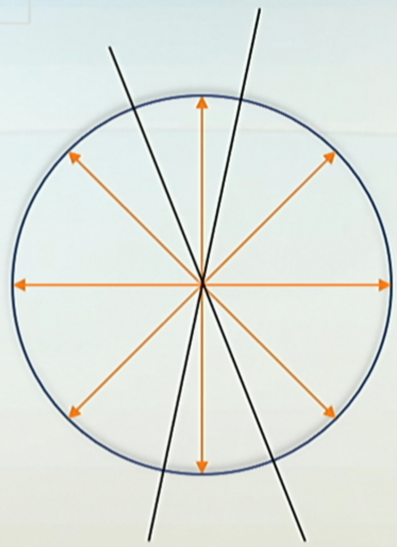


Search Space is $O(\frac{1}{\gamma})$

Step 2: Grover Search

# Perceptron – ML as Search

Step 1: Generate hypothesis



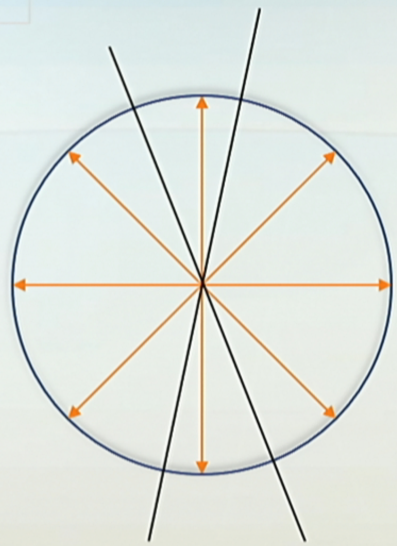
Search Space is $O(\frac{1}{\gamma})$

Step 2: Grover Search

Number of Operations: $O(\frac{1}{\gamma^{1/2}})$

# Perceptron – ML as Search

Step 1: Generate hypothesis



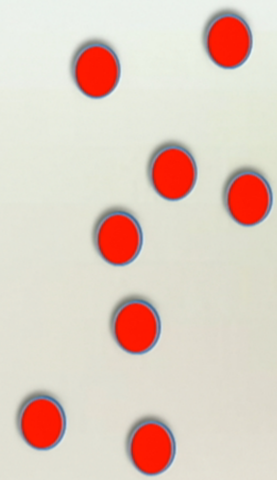
Search Space is $O(\frac{1}{\gamma})$

Step 2: Grover Search

Number of Operations: $O(\frac{1}{\gamma^{1/2}})$

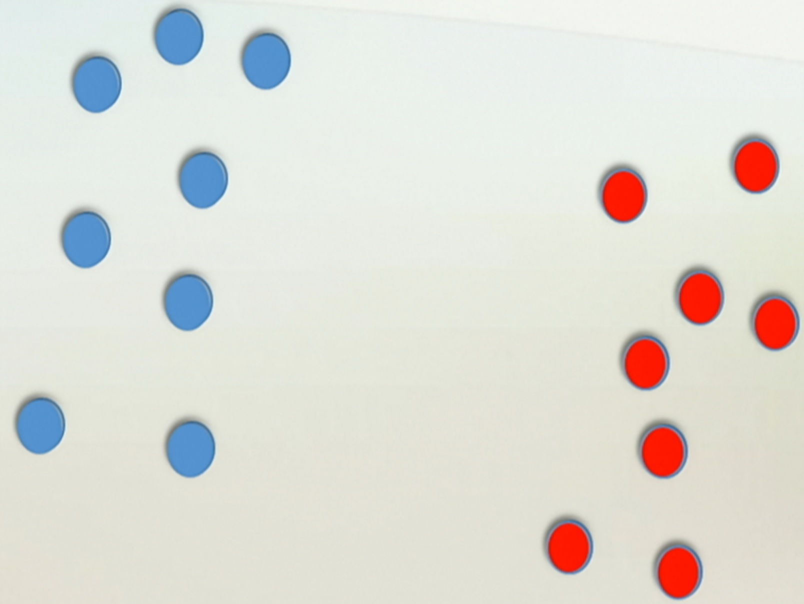
# Proof Sketch:

Search Space is $O(\frac{1}{\gamma})$



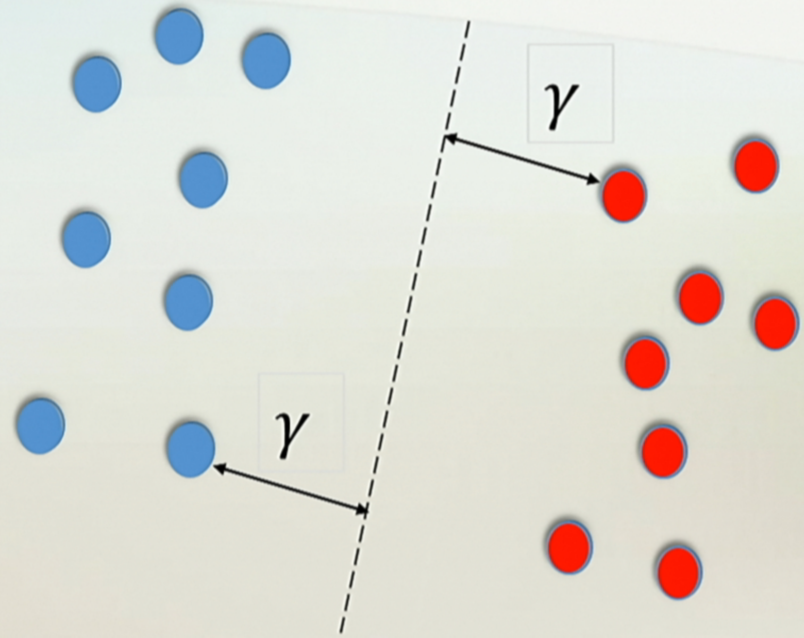
# Proof Sketch:

Search Space is $O(\frac{1}{\gamma})$



# Proof Sketch:

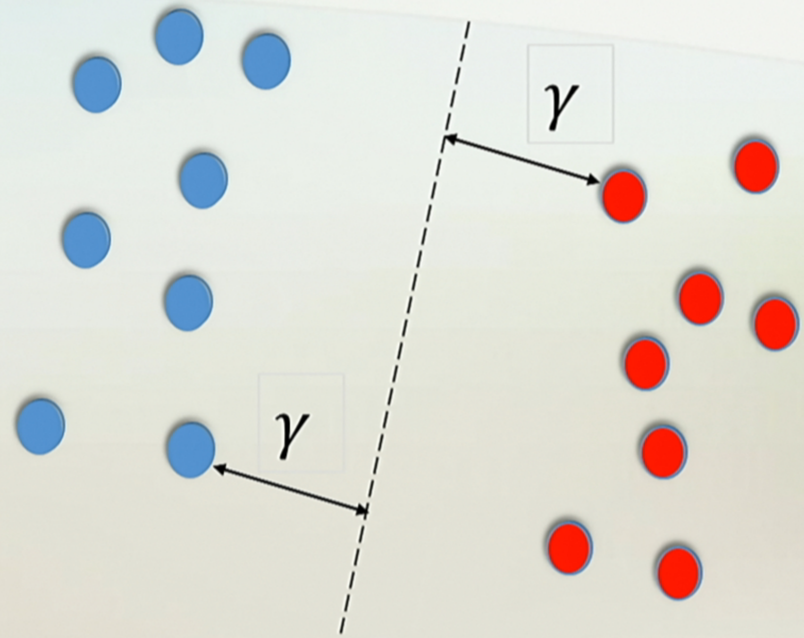
Search Space is $O(\frac{1}{\gamma})$



# Proof Sketch:

Search Space is $O(\frac{1}{\gamma})$

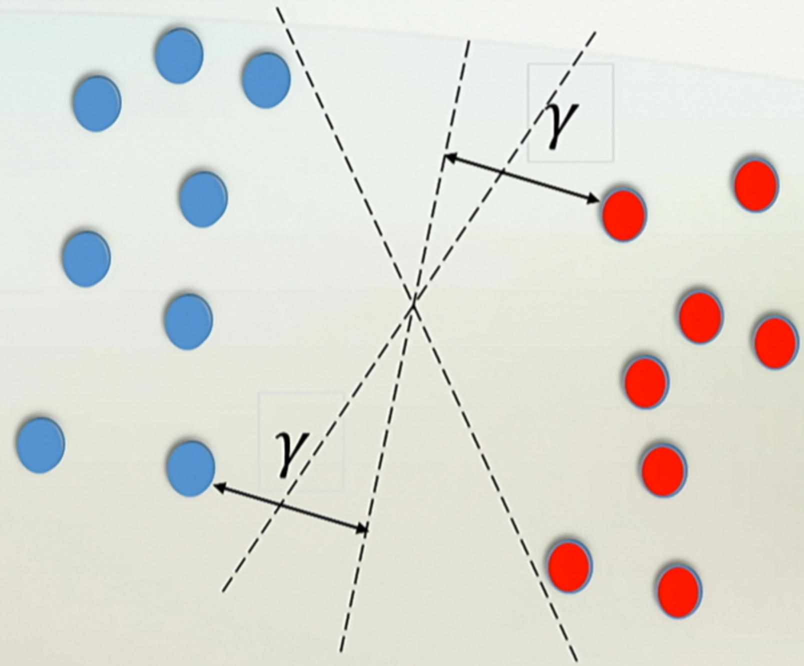
Perturb the max margin classifier by adding $w \sim N(0, I)$



# Proof Sketch:

Search Space is $O(\frac{1}{\gamma})$

Perturb the max margin classifier by adding $w \sim N(0, I)$

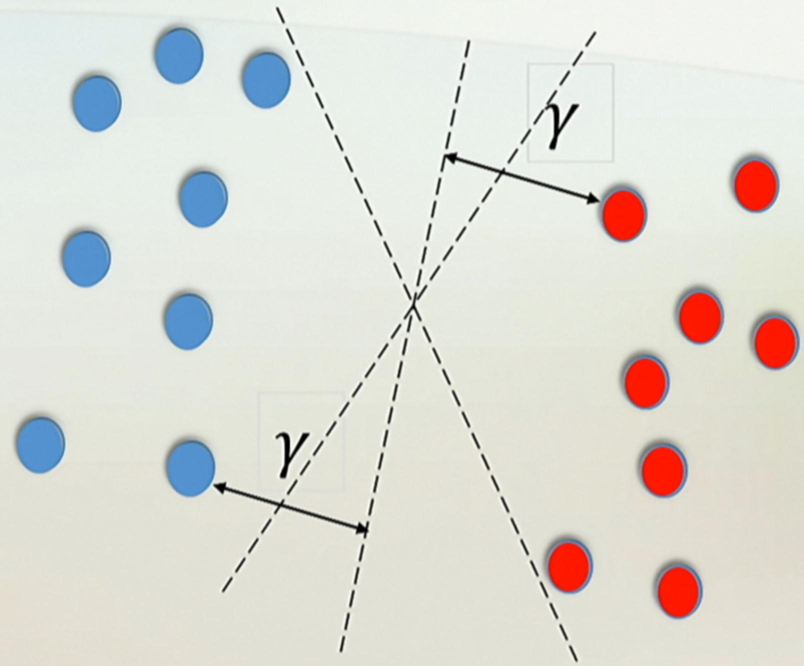


# Proof Sketch:

Search Space is $O(\frac{1}{\gamma})$

Perturb the max margin classifier by adding $w \sim N(0, I)$

Probability that perturbations that result in perfect separation is $O(\gamma)$



# Perceptron – ML as Search

- Data Input and Output
  - How to prepare training data?
  - How to use the classifier?
- We need an Oracle
  - Testing the separability of the classifier
- Do we really need the perfectly correct solution?

# Concluding Thoughts

- How to go beyond computational efficiency with quantum methods?

# Concluding Thoughts

- How to go beyond computational efficiency with quantum methods?
  - Statistical Efficiency

# Concluding Thoughts

- How to go beyond computational efficiency with quantum methods?
  - Statistical Efficiency
  - Performance on the Classification Task?

# Concluding Thoughts

- How to go beyond computational efficiency with quantum methods?
  - Statistical Efficiency
  - Performance on the Classification Task?

- Does quantum give us the ability to ask entirely new questions?

# Concluding Thoughts

- How to go beyond computational efficiency with quantum methods?
  - Statistical Efficiency
  - Performance on the Classification Task?

- Does quantum give us the ability to ask entirely new questions?
  - New Representations

# Concluding Thoughts

- How to go beyond computational efficiency with quantum methods?
  - Statistical Efficiency
  - Performance on the Classification Task?

- Does quantum give us the ability to ask entirely new questions?
  - New Representations
  - How can we take advantage of the “quantumness” of Hamiltonians for learning?