10-701 INTRODUCTION TO MACHINE LEARNING (PHD)
LECTURE 1: INTRO TO ML AND PERCEPTRON

LEILA WEHBE
CARNEGIE MELLON UNIVERSITY
MACHINE LEARNING DEPARTMENT

Lecture based on chapter 4 (http://ciml.info/dl/v0_99/ciml-v0_99-ch04.pdf) from Hal Daumé III, on Kilian Weinberger's lecture 3 (https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html) and on Tom Mitchell's lecture 1 (http://www.cs.cmu.edu/~tom/10701-S20/Intro-DTreesAndOverfitting-1-13-2020.pdf).

LECTURE OUTCOMES

- Definition of linear separator
- Perceptron algorithm
- Perceptron algorithm guarantees

LINKS (USE THE VERSION YOU NEED)

- Notebook (https://github.com/lwehbe/10701/blob/F22/Lecture_01_perceptron.ipynb)
- PDF slides (https://github.com/lwehbe/10701/raw/F22/Lecture_01_perceptron.slides.pdf)

WELCOME TO 10-701 INTRO TO MACHINE LEARNING

Lectures: MW, 10:10-11:30am, PH 100
Recitations: F, 10:10-11:30am, PH 100

Instructors:

Pradeep Ravikumar

Leila Wehbe

WELCOME TO 10-701 INTRO TO MACHINE LEARNING

Teaching Assistants:

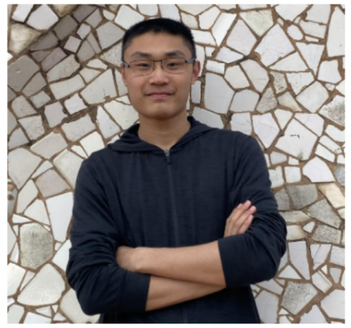
So Yeon (Tiffany) Min

Don Dennis

Runtian Zhai

Xinyue Chen

Dennis Li



Aryan Mehra

Varun Ursekar

LINKS

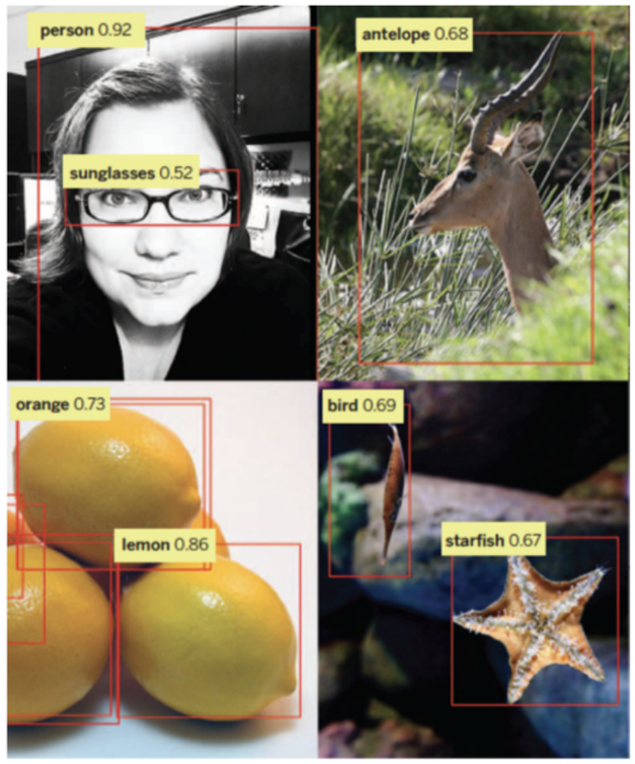
- Website (https://www.cs.cmu.edu/~lwehbe/10701_F22/) (includes schedule (https://www.cs.cmu.edu/~lwehbe/10701_F22/#schedule) and links to lectures)
- Piazza (https://piazza.com/cmu/fall2022/10701)
- Syllabus (https://www.cs.cmu.edu/~lwehbe/10701_F22/files/Syllabus.pdf)

WHAT IS MACHINE LEARNING?

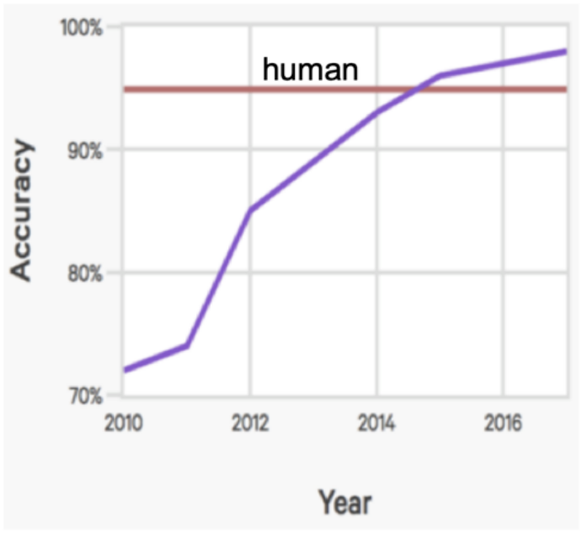
- "How can we build computer programs that automatically improve their performance through experience?"
    - Study of algorithms that
        - improve their performance **P**
        - at some task **T**
        - with experience **E**
    - well-defined learning task: (**P**,**T**,**E**)

- How can we learn from data?
- How robust is what we learn? What types of assumptions do we make with different approaches? What are the guarantees? How do we pick an approach?

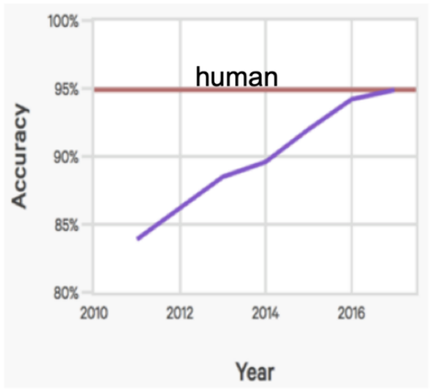
COMPUTER VISION



Imagenet Visual Recognition Challenge

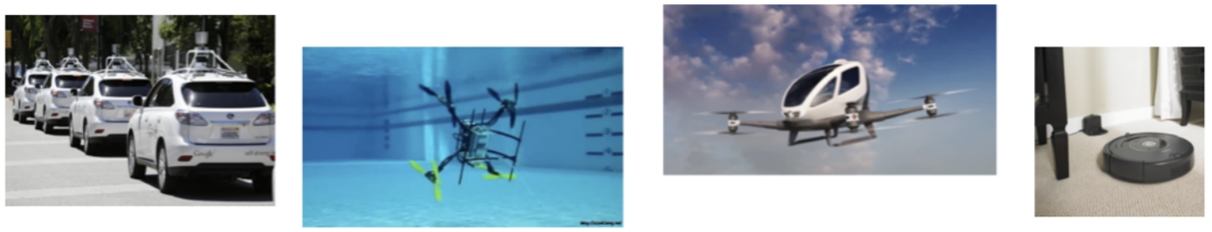


SPEECH RECOGNITION



ROBOTS

Factories, Land, Air, Sea, Mines, Homes



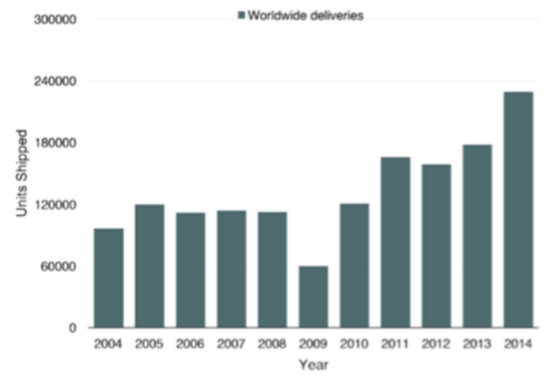


FIGURE 2.4 Worldwide shipping of robots over time. SOURCE: International Federation of Robotics, 2015.

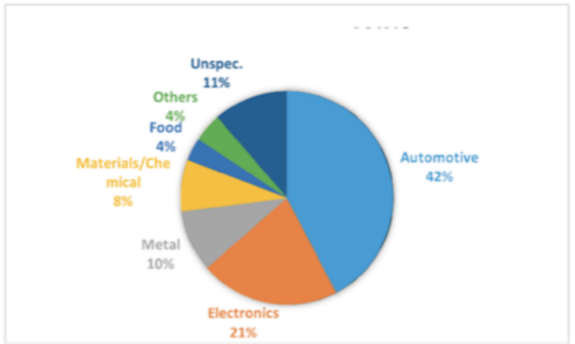
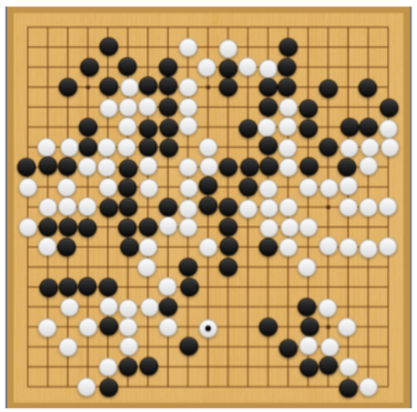


FIGURE 2.5 Robot application areas in 2015. SOURCE: Data from International Federation of Robotics, 2015.

GAMES AND REASONING

Chess

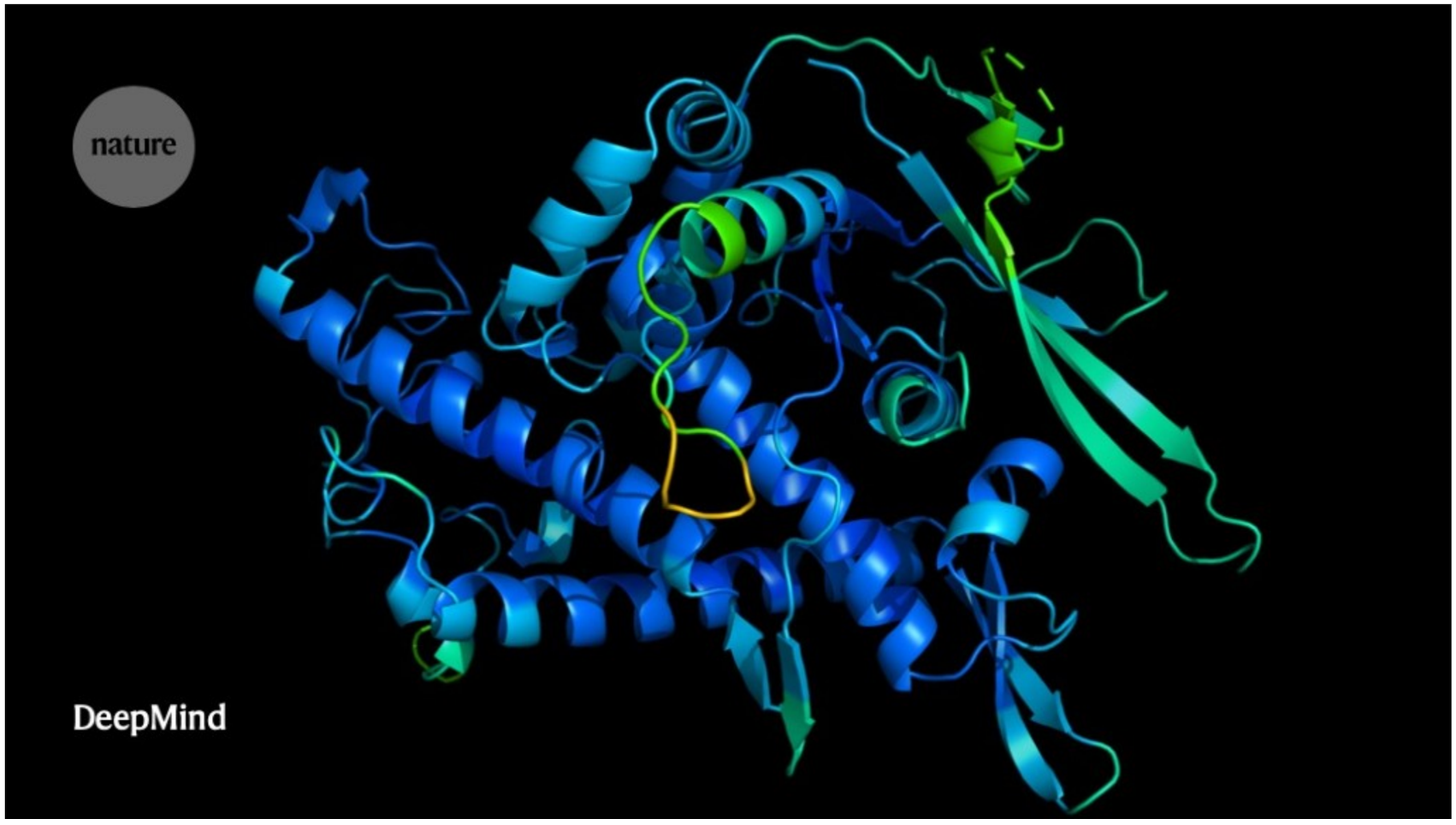


Go

Jeopardy

Poker

PROTEIN FOLDING



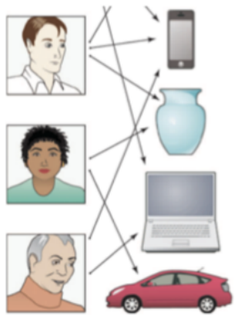
THE KEY: MACHINE LEARNING

conversational agents

medical diagnosis

fraud detection

translation



recommendations

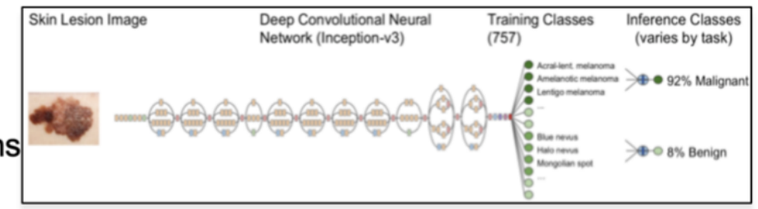
Many algorithms:

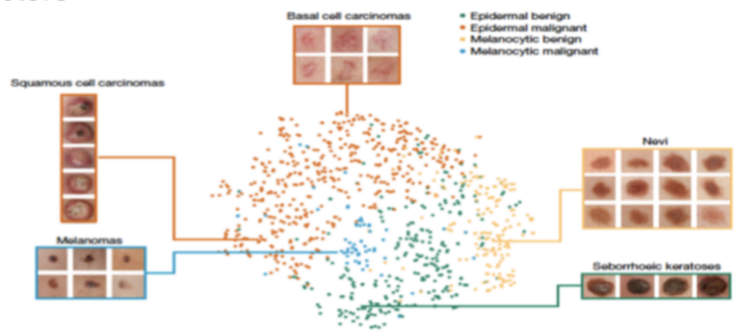
- Deep neural networks
- Bayesian networks
- Hidden Markov models
- Support Vector Machines
- Gaussian mixture model
- Expectation maximization
- ....

SKIN CANCER DIAGNOSIS

[Esteva et al., *Nature* 2017]

Trained on 129,450 skin images
plus 1.4 million standard photographs
Deep net Inception v3 architecture
Outperforms doctors



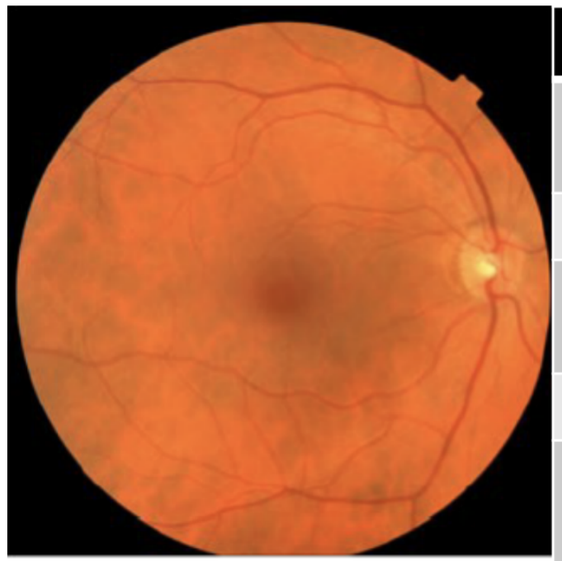


tsne visualization of final hidden layer

PREDICT CARDIOVASCULAR RISK FROM RETINAL PHOTOGRAPHS

[Poplin et al., *Nature Biomed Eng.* 2018]

Trained deep net on 284,335 retinal images
New approach to detecting risk factors and biometrics



	Accuracy
Age	within 3.26 years on average
Smoker?	71%
Systolic blood pressure	within 11 mmHg on average
Gender	97%
Major cardiac event within past 5 years?	70%

MACHINE LEARNING THEORY

PAC Learning Theory
(supervised concept learning)

examples (*m*)

representational complexity (*H*)

error rate (ε)

failure probability (δ)

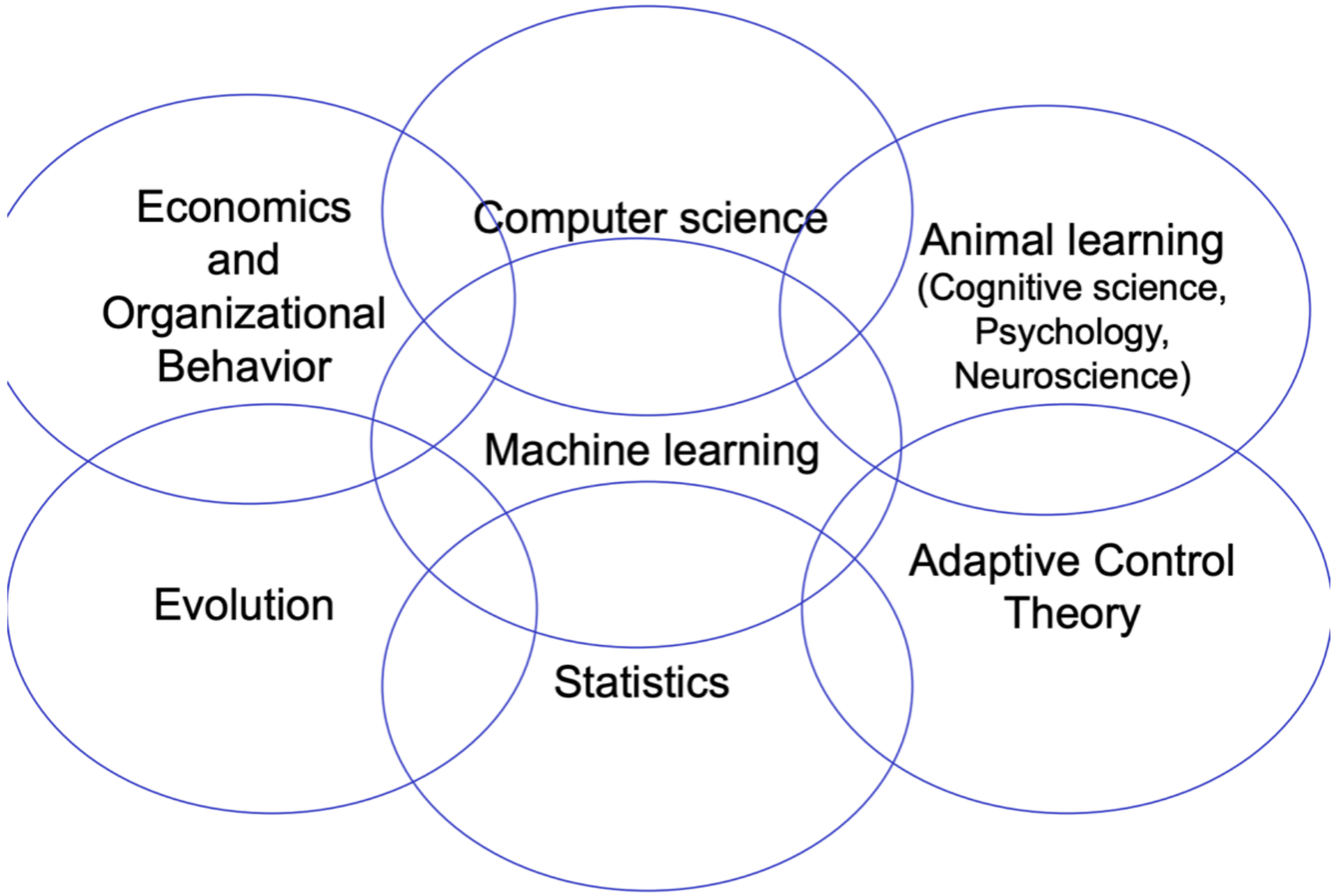
$$m \geq \frac{1}{\epsilon}(\ln |H| + \ln(1/\delta))$$

Other theories for

- Reinforcement skill learning
- Semi-supervised learning
- Active student querying
- …

… also relating:

- # of mistakes during learning
- learner's query strategy
- convergence rate
- computational demands
- asymptotic performance
- bias, variance, Bayesian priors
- VC dimension


Economics
and
Organizational
Behavior
Computer science
Animal learning
(Cognitive science,
Psychology,
Neuroscience)
Machine learning
Evolution
Adaptive Control
Theory
Statistics

SOCIAL IMPACTS OF MACHINE LEARNING

- Better, evidence-based, decision making in many domains
    - Medical diagnosis, Credit card fraud detection, Online tutoring, Anticipating equipment failures, Marketing, Legal sentencing, …
- Created breakthroughs in AI, with huge impact on society
    - Computer vision, speech, text processing, self-driving cars, games, …
- Raises new issues
    - Explainability
    - Bias
    - Privacy
    - If big data is key to successful ML, who controls access to the data?
    - …

WE WILL COVER IN THIS COURSE

Algorithms:

- Decision trees
- Bayes classifiers
- Logistic regression
- Deep neural networks
- Graphical models
- Expectation maximization
- Support Vector Machines
- Kernel regression
- PCA
- Reinforcement learning

Concepts:

- Statistical estimation
- Overfitting
- Representation learning
- Probabilistic models
- Maximum margin models
- Probably approximately correct learning
- VC dimension
- Role of unlabeled data
- Optimization

HIGHLIGHTS OF COURSE LOGISTICS

- 4 HW assignments (36%)
- 2 exams (see schedule, 20% each)
- Project (24%)

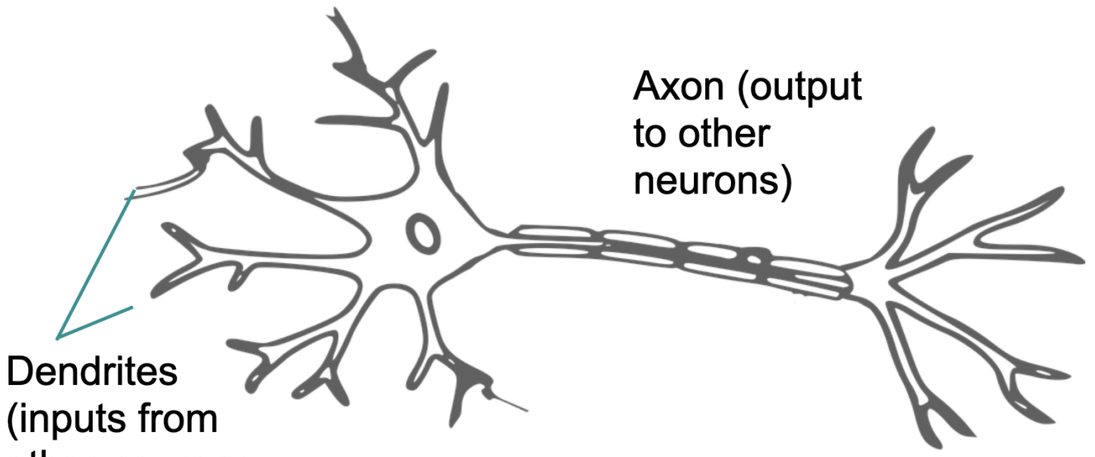
- Homework assignments will be submitted on gradescope. HW 1 out on the 7th.
- 8 late days in total, maximum 3 per assignment.
- Collaboration is ok if you only talk to each other, and then write / implement separately.
- **Collaboration should be disclosed.** There is a dedicated section for each homework assignment.
    - What happens if you disclose / don't disclose?
- What happens if you copy code from someone else (even if you change it)?

HIGHLIGHTS OF COURSE LOGISTICS

- Projects will be in groups of 2 or 3
- We will propose a set of project topics for you, you can also propose your own
- Project proposal, milestone report, and final report.
- Use the GCC (Global Communication Center) for this and other writing tasks you have!
- Project proposals will be due on September 28, we will release a list of project topics soon.

THE PERCEPTRON

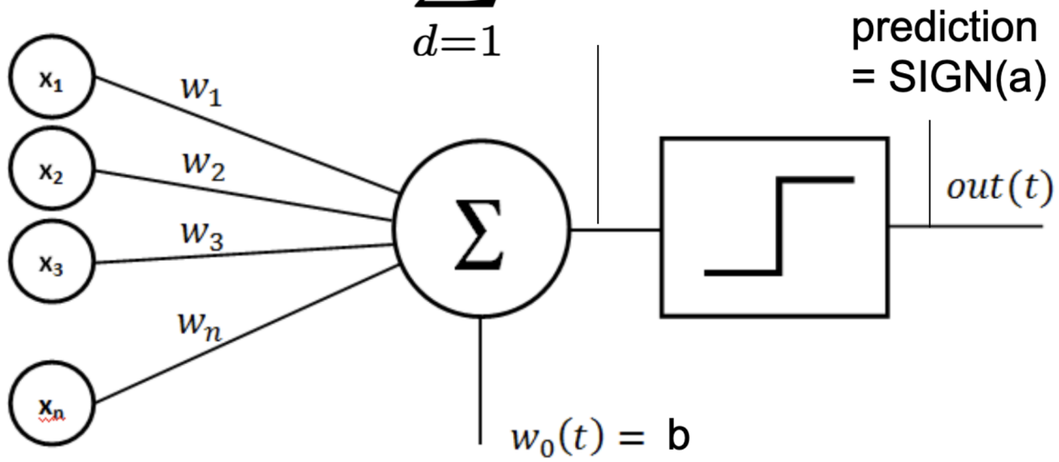
- Introduced by Rosenblatt in 1958
- Inspired by real neurons



THE PERCEPTRON

- Introduced by Rosenblatt in 1958
- Inspired by real neurons

$$a = b + \sum_{d=1}^{D} w_d x_d$$



THE PERCEPTRON

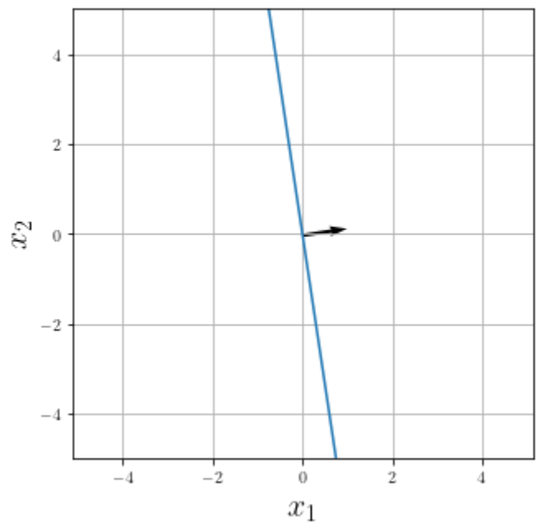
- Assume data is binary
- Assume data is linearly separable:
    - there exist a hyperplane that perfectly divides the two classes

REFRESHER:

- Recall how to define a hyperplane:
    - a subspace whose dimension is one less than that of its ambient space
    - if x is d-dimensional:
        - $\langle \mathbf{x}, w \rangle + b = 0$

In [34]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
plt.rcParams['text.usetex'] = True
import warnings
warnings.filterwarnings('ignore')
```

In [44]:
```
def plot_line(ax,xlims, w, do_norm=True):
    x1 = np.linspace(xlims[0],xlims[1],1000)
    x2_plot = (- w[2] - w[0]*x1)/w[1]  #  w[0]*x1 + w[1]*x2 + w[2] = 0
    ax.plot(x1,x2_plot)
    origin = x1[np.array(x1.shape[0]/2).astype(int)], x2_plot[int(x1.shape[0]/2)]
    nn = np.linalg.norm(w) if do_norm else 1
    ax.quiver(*origin, w[0]/nn,w[1]/nn, color='k',angles='xy', scale_units='xy', scale=1)
    ax.axis('equal')
    ax.axis([xlims[0],xlims[1],xlims[0],xlims[1]])
    ax.set_xlabel(r'$x_1$',fontsize=20); ax.set_ylabel(r'$x_2$',fontsize=20)
    ax.grid()

w = [4,0.6]
b = 0
f, ax = plt.subplots(figsize=(5,5))
plot_line(ax, [-5,5], [w[0],w[1],b])
```

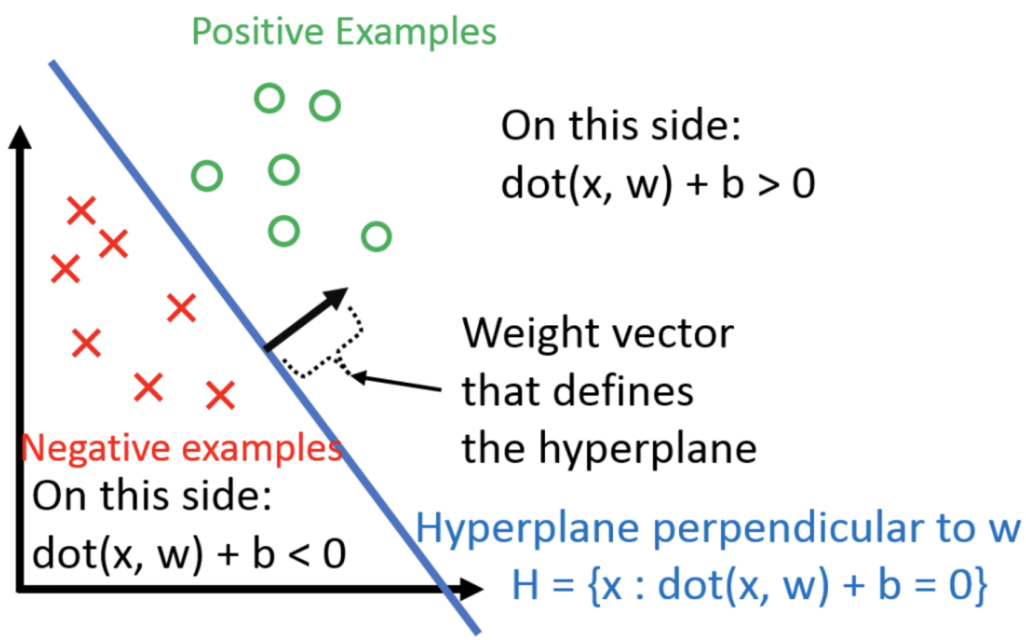


THE PERCEPTRON

- Assume data is binary
- Assume data is linearly separable:
    - there exist a hyperplane that perfectly divides the two classes

$$\exists \mathbf{w}, b \quad \text{s.t.} \quad \forall (\mathbf{x}_i, y_i) \in \boldsymbol{D},$$
$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) > 0$$

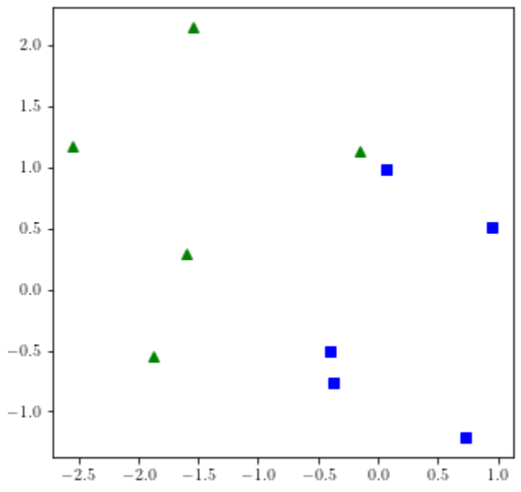
$$\exists \mathbf{w}, b \quad \text{s.t.} \quad \forall (\mathbf{x}_i, y_i) \in \boldsymbol{D}, \\ y_i(\mathbf{w}^\top \mathbf{x}_i + b) > 0$$



source (https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html)

What is a separable dataset?

In [61]:
```
from sklearn import datasets

X, y = datasets.make_blobs(n_samples=10, centers=np.array([[-1,1],[1,-1]]),
                           n_features=2, center_box=(0, 10))
y[y==0] = -1

def plot_dataset(ax,X,y):
    ax.plot(X[:, 0][y == -1], X[:, 1][y == -1], 'g^')
    ax.plot(X[:, 0][y == 1], X[:, 1][y == 1], 'bs');

f, ax = plt.subplots(figsize=(5,5))
plot_dataset(ax,X,y)
```



SIMPLIFYING w AND b

- We can write $\mathbf{x}_i$ as:

  \begin{align} {\bf x_i}' &= \begin{bmatrix}

  ```
  {\bf x_i} \\
  1
  \end{bmatrix}
  ```

  \end{align}

- and incorporate b into $\mathbf{w}$:

$$\mathbf{w}' = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$$

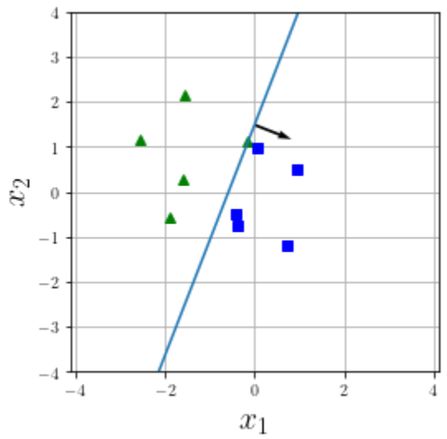
- The same hyperplane is now defined by $\mathbf{w}'^\top \mathbf{x}' = 0$
- Why does this work?

- We will use $\mathbf{w}$ and $\mathbf{x}$ to refer to these vectors in the rest of the lecture.

THE PERCEPTRON TRAINING ALGORITHM

Initialize $\vec{w} = \vec{0}$ // Initialize $\vec{w}$. $\vec{w} = \vec{0}$ misclassifies everything.
while TRUE **do** // Keep looping
  $m = 0$ // Count the number of misclassifications, m
  for $(x_i, y_i) \in D$ **do** // Loop over each (data, label) pair in the dataset, D
    if $y_i(\vec{w}^T \cdot \vec{x_i}) \leq 0$ **then** // If the pair $(\vec{x_i}, y_i)$ is misclassified
      $\vec{w} \leftarrow \vec{w} + y\vec{x}$ // Update the weight vector $\vec{w}$
      $m \leftarrow m + 1$ // Counter the number of misclassification
    end if
  end for
  if $m = 0$ **then** // If the most recent $\vec{w}$ gave 0 misclassifications
    break // Break out of the while-loop
  end if
end while // Otherwise, keep looping!

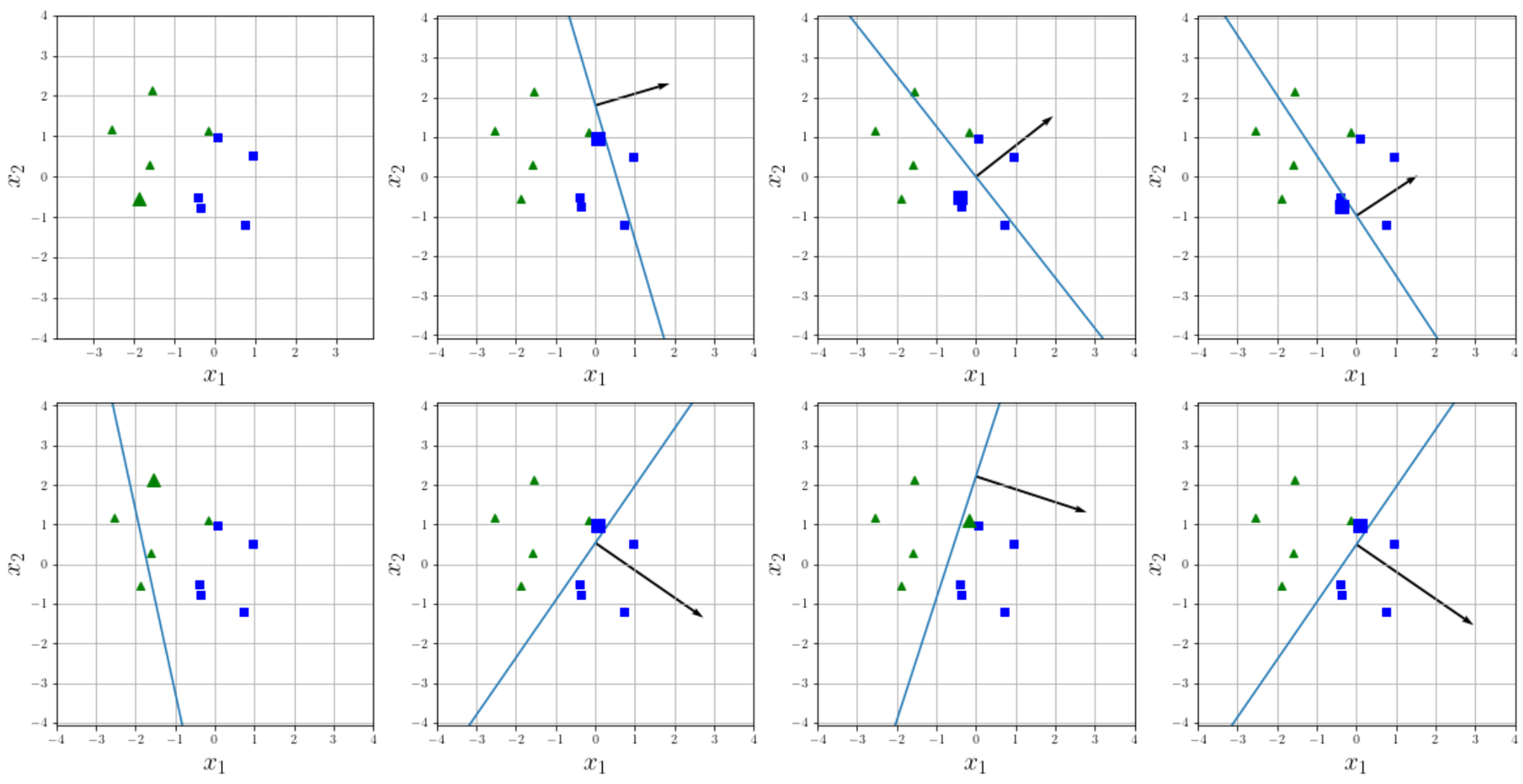
In [62]:
```
def perceptron_train(X,y,MaxIter=20):
    w = np.zeros((X.shape[1]))
    for i in range(MaxIter):
        m = 0
        for (xi,yi) in zip(X,y):
            if yi*w.T.dot(xi)<=0:
                w = w + yi*xi
                m = m + 1
        if m==0:
            break
    return w

f,ax = plt.subplots(figsize=(4,4))
plot_dataset(ax,X,y)
Xprime = np.hstack([X,np.ones((X.shape[0],1))])
w = perceptron_train(Xprime,y)
plot_line(ax,[-4,4], w)
```



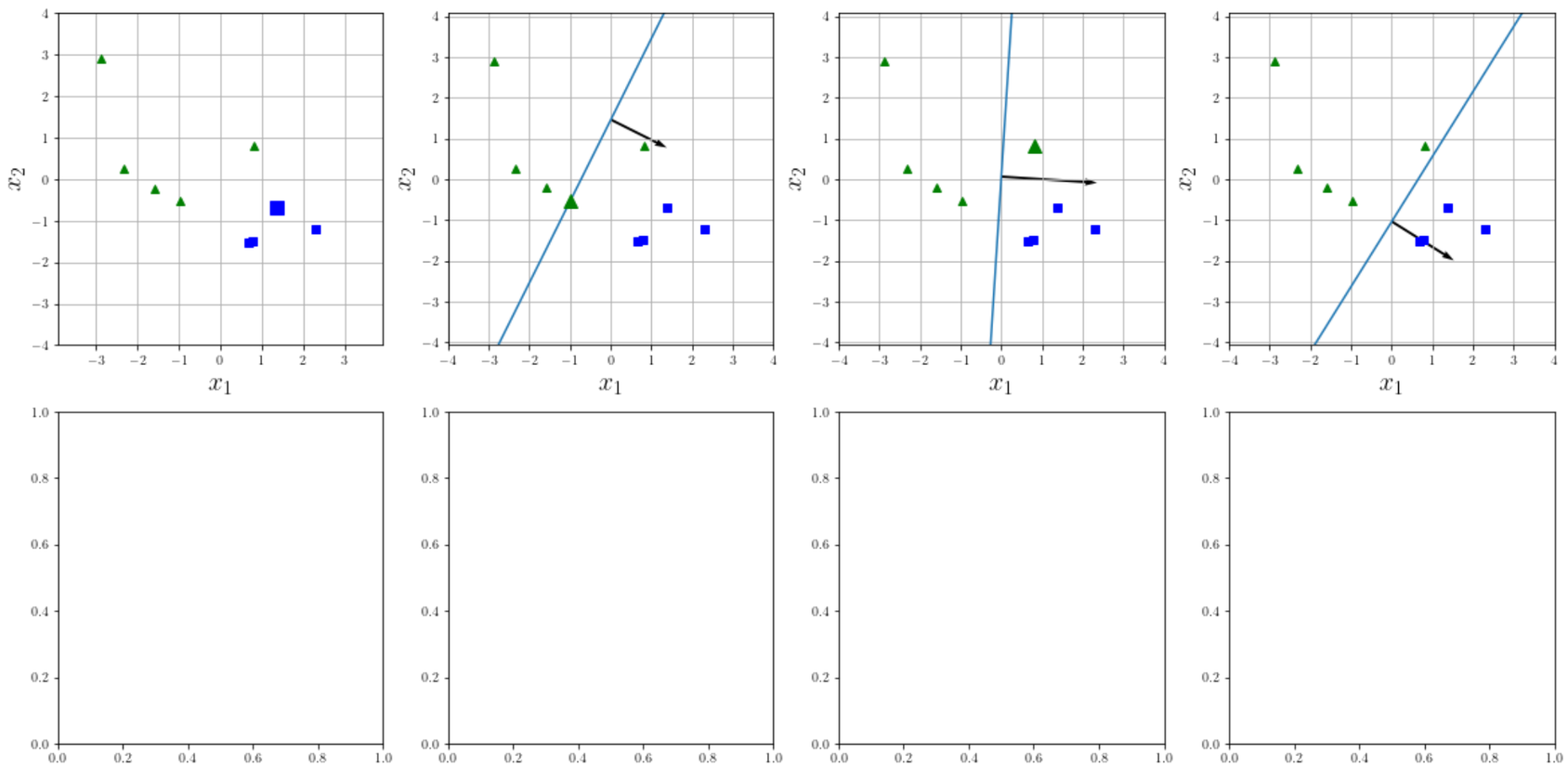
In [63]:
```
def perceptron_train_and_plot(axs,X,y,MaxIter=20):
    w = np.zeros((X.shape[1]))
    plot_dataset(axs[0],X,y)
    plot_line(axs[0],[-4,4], w)
    plt_cnt = 0
    for i in range(MaxIter):
        m = 0
        for (xi,yi) in zip(X,y):
            if yi*w.T.dot(xi)<=0:
                w = w + yi*xi
                m = m + 1
                plt_cnt = plt_cnt + 1
                try:
                    if yi == -1:
                        axs[plt_cnt-1].plot([xi[0]],[xi[1]],'g^',markersize=10)
                    else:
                        axs[plt_cnt-1].plot([xi[0]],[xi[1]],'bs',markersize=10)
                    plot_dataset(axs[plt_cnt],X,y)
                    plot_line(axs[plt_cnt],[-4,4], w, do_norm=False)
                except:
                    print('not enough subplots')
        if m==0:
            break
    return w
```

In [64]:
```
f,axs = plt.subplots(nrows=2,ncols=4,figsize=(20,10))
w = perceptron_train_and_plot(axs.reshape(-1),Xprime,y)
```

```
not enough subplots
not enough subplots
not enough subplots
not enough subplots
not enough subplots
```



In [68]:
```
X, y = datasets.make_blobs(n_samples=10, centers=np.array([[-1,1],[1,-1]]),
                           n_features=2, center_box=(0, 10))
y[y==0] = -1
Xprime = np.hstack([X,np.ones((X.shape[0],1))])
f,axs = plt.subplots(nrows=2,ncols=4,figsize=(20,10))
w = perceptron_train_and_plot(axs.reshape(-1),Xprime,y)
```



WHY DOES THIS WORK?

- what happens if you missclassify a positive example? (i.e. $\mathbf{w^k}^\top \mathbf{x_i} < 0$)

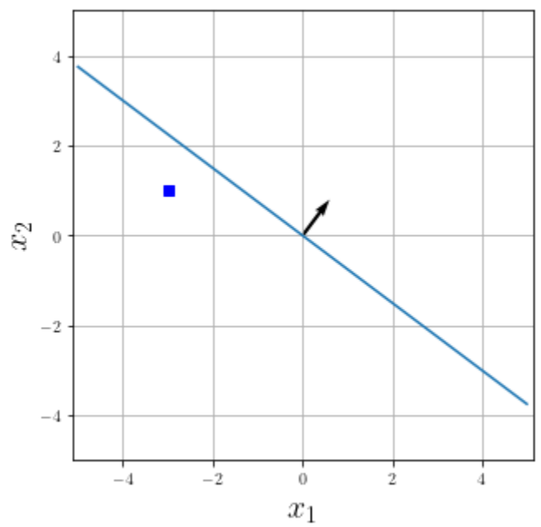
$$\begin{aligned}
\mathbf{w^{k+1}} &= \mathbf{w^k} + \mathbf{x_i} \\
\mathbf{w^{k+1}}^\top \mathbf{x_i} &= (\mathbf{w^k} + \mathbf{x_i})^\top \mathbf{x_i} \\
&= \mathbf{w^k}^\top \mathbf{x_i} + \mathbf{x_i}^\top \mathbf{x_i} \\
&> \mathbf{w^k}^\top \mathbf{x_i}
\end{aligned}$$

- The prediction becomes more positive (vice versa for a negative example).
- Thus the boundary is getting closer to correctly classiying $\mathbf{x_i}$.

HOW MANY ITERATIONS ARE NEEDED WITH A DATASET WITH ONE EXAMPLE?

In [55]:
```
f, ax = plt.subplots(figsize=(5,5))
plot_line(ax, [-5,5], [3,4,0])
ax.plot([-3],[1],'bs')
```

Out[55]:
```
[<matplotlib.lines.Line2D at 0x11c503c88>]
```



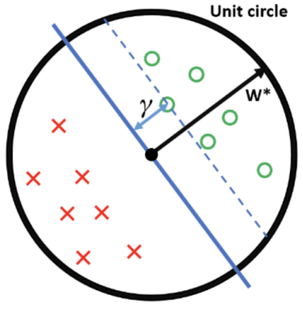
CONVERGENCE OF THE PERCEPTRON ALGORITHM

The perceptron algorithm converges in $\frac{1}{\gamma^2}$ updates if the data is linearly separable.

γ is the margin of the problem instance (defined on next slide).

NOTION OF MARGIN

- Assume there exists $\mathbf{w}^*$ such that $\forall(\mathbf{x_i}, y_i) \in D, \quad y_i(\mathbf{x_i}^\top \mathbf{w}^*) > 0$
- Also assume we rescale $\mathbf{w}^*$ and $\mathbf{x_i}$s such that:
    - $||\mathbf{w}^*|| = 1$ and $||\mathbf{x_i}|| \leq 1 \ \forall \mathbf{x_i}$ (how?)
- The margin γ of the hyperplane $\mathbf{w}^*$ is the minimum distance between one of the points and the hyperplane:
    - $\gamma = \min_{(\mathbf{x_i}, y_i) \in \mathcal{D}} |\mathbf{x_i}^\top \mathbf{w}^*|$ (since $\mathbf{w}^*$ is unit norm)



source (https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html)

THEOREM

- Given:
    - All $\mathbf{x}_i$ s are within the unit sphere
    - There exists a separating hyperplane $\mathbf{w}^*$, with $||\mathbf{w}^*|| = 1$
    - γ is the margin of hyperplane $\mathbf{w}^*$
- If all of the above holds, then the Perceptron algorithm makes at most $\frac{1}{\gamma^2}$ mistakes.

- Would we want a large margin or a small margin?
- What types of datasets will converge quickly?

PROOF

source (https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html)

Keeping what we defined above, consider the effect of an update ($\mathbf{w}$ becomes $\mathbf{w} + y\mathbf{x}$) on the two terms $\mathbf{w}^\top \mathbf{w}^*$ and $\mathbf{w}^\top \mathbf{w}$. We will use two facts:

- $y(\mathbf{x}^\top \mathbf{w}) \leq 0$: This holds because $\mathbf{x}$ is misclassified by $\mathbf{w}$ - otherwise we wouldn't make the update.
- $y(\mathbf{x}^\top \mathbf{w}^*) > 0$: This holds because $\mathbf{w}^*$ is a separating hyper-plane and classifies all points correctly.

1. Consider the effect of an update on $\mathbf{w}^\top \mathbf{w}^*$:

$$(\mathbf{w} + y\mathbf{x})^\top \mathbf{w}^* = \mathbf{w}^\top \mathbf{w}^* + y(\mathbf{x}^\top \mathbf{w}^*) \geq \mathbf{w}^\top \mathbf{w}^* + \gamma$$

The inequality follows from the fact that, for $\mathbf{w}^*$, the distance from the hyperplane defined by $\mathbf{w}^*$ to $\mathbf{x}$ must be at least γ (i.e. $y(\mathbf{x}^\top \mathbf{w}^*) = |\mathbf{x}^\top \mathbf{w}^*| \geq \gamma$).

This means that for each update, $\mathbf{w}^\top \mathbf{w}^*$ grows by **at least** γ.

2. Consider the effect of an update on $\mathbf{w}^\top \mathbf{w}$:

$$(\mathbf{w} + y\mathbf{x})^\top (\mathbf{w} + y\mathbf{x}) = \mathbf{w}^\top \mathbf{w} + \underbrace{2y(\mathbf{w}^\top \mathbf{x})}_{<0} + \underbrace{y^2(\mathbf{x}^\top \mathbf{x})}_{0 \leq \ \leq 1} \leq \mathbf{w}^\top \mathbf{w} + 1$$

The inequality follows from the fact that

- $2y(\mathbf{w}^\top \mathbf{x}) < 0$ as we had to make an update, meaning $\mathbf{x}$ was misclassified
- $0 \leq y^2(\mathbf{x}^\top \mathbf{x}) \leq 1$ as $y^2 = 1$ and all $\mathbf{x}^\top \mathbf{x} \leq 1$ (because $\|\mathbf{x}\| \leq 1$).

3. Now we know that after M updates the following two inequalities must hold:

   (1) $\mathbf{w}^\top \mathbf{w}^* \geq M\gamma$

   (2) $\mathbf{w}^\top \mathbf{w} \leq M$.

   We can then complete the proof:

$$
\begin{aligned}
M\gamma &\leq \mathbf{w}^\top \mathbf{w}^* && \text{By (1)} \\
&= \|\mathbf{w}\| \cos(\theta) && \text{by definition of inner-product, where } \theta \text{ is the angle between } \mathbf{w} \text{ and } \mathbf{w}^*. \\
&\leq ||\mathbf{w}|| && \text{by definition of cos, we must have } \cos(\theta) \leq 1. \\
&= \sqrt{\mathbf{w}^\top \mathbf{w}} && \text{by definition of } \|\mathbf{w}\| \\
&\leq \sqrt{M} && \text{By (2)} \\
\\
&\Rightarrow M\gamma \leq \sqrt{M} \\
&\Rightarrow M^2\gamma^2 \leq M \\
&\Rightarrow M \leq \frac{1}{\gamma^2} && \text{And hence, the number of updates } M \text{ is bounded from above by a constant.}
\end{aligned}
$$

- What happens if the data is not separable?
- Does the order matter?