Machine Learning Foundations
Lecture 2

Challenges of Machine Learning

- FITTING THE TRAINING DATA
- UTILIZES SUPERVISED AND UNSUPERVISED LEARNING TECHNIQUES TO STATISTICALLY ESTIMATE COMPLICATED FUNCTIONS

Challenges of Deep Learning

- FINDING PATTERNS THAT GENERALIZE TO NEW DATA
- IDENTIFIES HYPERPARAMETERS WHOSE VALUES NEED TO BE FIXED TO ALLOW LARGE NUMBER OF INTERCONNECTED NEURONS TO CAPTURE SUBTLE NUANCES AND VARIANCES IN DATA

Machine Learning

- Machine learning enables us to tackle tasks that are too difficult to be solved with fixed programs written by human beings.

Learning Algorithms

- Writing a Program that specifies a robot how to walk manually.

Vs

Writing a program that gives the ability to the robot to learn to walk gradually.

Learning Algorithms

- A machine learning algorithm is an algorithm that is able to learn from data. But What is Learning??

Learning Algorithms

- Improve their Performance, P
- At some task, T
- With experience, E

Common Machine Learning tasks

CLASSIFICATION TASKS

- *Goal*: Specify which of k categories some input belongs to.
- Learning Algorithm is asked to produce a function *y=f(x)*.
- This single function f outputs a probability distribution over classes mapping from a vector input to a categorical output.
- The model assigns an input described by x to a category identified by y.

Common Machine Learning tasks

EXAMPLE : CLASSIFICATION TASKS

+ *Task* : Object Recognition
+ *Input:* An image
+ *Output:* A numeric code identifying the object in the image.
+ *Example:* A robot that can act as a waiter that can recognize different kinds of drinks and deliver them to people on command.

Other Examples?

→ Face Recognition

1. *Tagging People in Photo Collections*
2. *Computers Recognizing their Users*

Common Machine Learning tasks

REGRESSION TASKS

- *Goal*: Predict a numerical value given some input.
- *Task*: Algorithmic Trading, Insurance Premiums etc.
- *Example :* Setting Insurance Premiums by Predicting of the expected claim amount that an insured person will make or the prediction of future prices of securities.

Common Machine Learning tasks

TRANSCRIPTION TASKS

- *Goal*: Observe a relatively unstructured representation of some data and transcribe the information into discrete textual form.
- *Task*: Optical Character Recognition, Google Street View, Speech Recognition etc.

Common Machine Learning tasks

EXAMPLE : TRANSCRIPTION TASKS

+ *Task*: Optical Character Recognition
+ *Input:* A photograph containing an image of text.
+ *Output:* The text from the image in the form of a sequence of characters.
+ ***Example 1:*** Google Street View identifies house numbers and addresses.
+ ***Example 2:*** Computer is provided with an audio waveform to return textual characters that were spoken in the audio recording.

Common Machine Learning tasks

MACHINE TRANSLATION TASKS

- *Goal*: Convert a sequence of symbols in one language to another language.
- *Task*: Translating a natural language say, English to French.

Common Machine Learning tasks

STRUCTURED OUTPUT TASKS

+ Goal: Program outputs several values that are all tightly interrelated.
+ Task: Mapping a certain input into different categorical structures.
+ Example: Parsing a natural language sentence into a tree that describes its grammatical structure by tagging nodes of trees as being verbs, nouns, adverbs etc.

Other Examples?

➔ Pixel Wise Segmentation of Images

1. *Computer Program Assigns every pixel in an image to a specific category*
2. *Annotating the locations of roads in aerial photographs*

Other Examples?

→ Image Captioning

1. *Computer Program observes an image and outputs a natural language sentence describing the image.*
2. *Structuring the sentence to form a valid meaningful phrase.*

Common Machine Learning tasks

ANOMALY DETECTION TASKS

+ *Goal*: Program sifts through a set of events or objects and flags some of them as being unusual or atypical..
+ *Task*: Credit Card Fraud Detection

Other Examples?

➔ **Detecting Misuse of Credit cards**

1. *Credit card companies detect misuse of cards by modeling customer's purchasing habits.*
2. *If a thief steals your card or card information, the thief's purchases will often reflect a different probability distribution over types of purchases than yours.*

Common Machine Learning tasks

SYNTHESIS AND SAMPLING TASKS

+ *Goal:* Generate new examples that are similar to those in the training data.
+ *Task*: Media Applications, Animated Graphics
+ *Example*: Generating textures for large objects or landscapes instead of requiring an artist to manually label each pixel.

Common Machine Learning tasks

DENSITY ESTIMATION TASKS

- *Goal:* The learning algorithm needs to learn the structure of the data it has seen so as to implicitly capture the structure of the probability distribution using density estimates.
- *Task*: Media Applications, Animated Graphics
- *Example*: Generating textures for large objects or landscapes instead of requiring an artist to manually label each pixel.

UNSUPERVISED LEARNING ALGORITHMS

+ No labels
+ There is no instructor or teacher and the algorithm must learn to make sense of the data without any explicit guide.
+ Dataset is divided into clusters of similar examples.

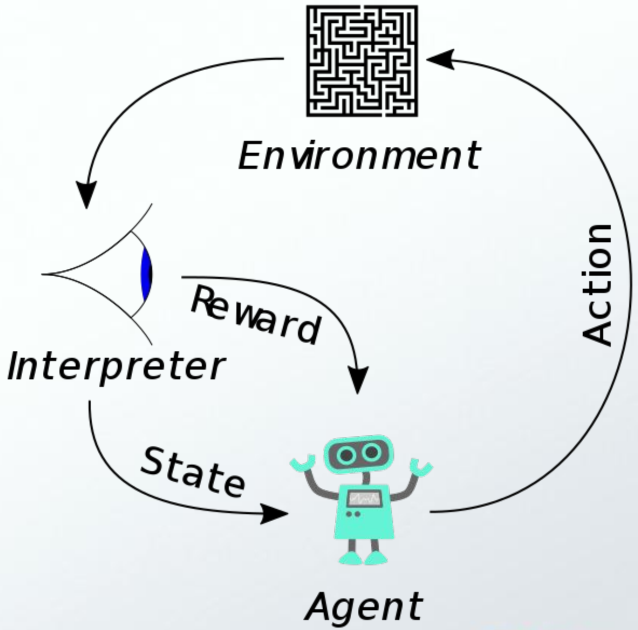
SUPERVISED LEARNING ALGORITHMS

+ Dataset contains both features and labels.
+ Observing several examples of a random vector x along with the target y to make conclusions from the data.
+ Dataset is divided into clusters of similar examples.

REINFORCEMENT LEARNING

- Do not work with a fixed dataset.
- Reinforcement Learning Algorithms interact with the environment, so there is a feedback loop between the learning system and its experiences.

REINFORCEMENT LEARNING



Capacity, Overfitting and underfitting

GENERALIZATION ERROR: The expected value of the error on a new input. The generalization error of a machine learning model is measured by its performance on a test set of examples that were collected separately from the training set.

Capacity, Overfitting and underfitting

CAPACITY: ML Algos perform best when their capacity is appropriate for their true complexity of the task they need to perform and the amount of training data they are provided with.

E.g., Generalizing linear regression to include polynomial functions increases its capacity.

CAPACITY, OVERFITTING AND UNDERFITTING

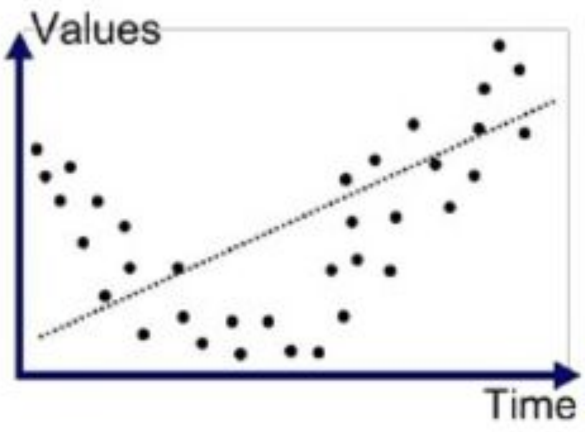
UNDERFITTING: Models with insufficient capacity are unable to solve complex tasks. The idea is to make the training error small.

- Underfitting occurs when the model is not able to obtain a sufficiently low error value on the training set.
- E.g., if A linear function is unable to capture the curvature of the true underlying problem, it underfits.

Capacity, Overfitting and underfitting

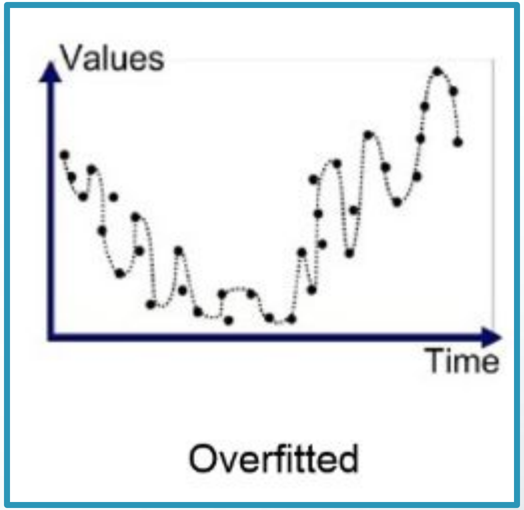
OVERFITTING: Models with higher capacity than needed to solve the present task. The idea is to make the gap between training error and test error small.

- Overfitting occurs when the gap between training error and test error is too large.
- E.g., A degree 9 predictor attempting to fit a problem where the true underlying problem is quadratic.



Underfitted

➔ A polynomial function fit to the data generalizing well to see unseen points.


Values
Time
Overfitted

OCCAM's RAZOR

This principle states that among competing hypothesis that explain known observations equally well, one should choose the simplest one that gives a reasonable accuracy.

PARAMETERS

- Parameters are values that control the behaviour of the system.
- Weights (w) determine how each feature affects the prediction.

PARAMETERS

- If a feature x_i receives a **positive** weight w_i then increasing the value of that feature Increase the value of our prediction.

PARAMETERS

- If a feature x_i receives a **negative** weight w_i then increasing the value of that feature decreases the value of our prediction.

PARAMETERS

- If a feature weight is **large** in magnitude then it has a large effect on the prediction.

PARAMETERS

- If a feature weight is **zero** it has no effect on the prediction.

REGULARIZATION

- Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.
- Find a way to penalize overly complex models.

HYPERPARAMETERS

- Hyperparameters are settings that we can use to control the algorithm's behaviour.
- Hyperparameters control model capacity.
- For example, the polynomial regression has a single parameter → the degree of its polynomial.

VALIDATION SET

SPLIT

- Split the training data into two disjoint subsets.

Learn

- About 80 percent of the training data is used to learn the parameters and optimize the hyper parameters.

Test

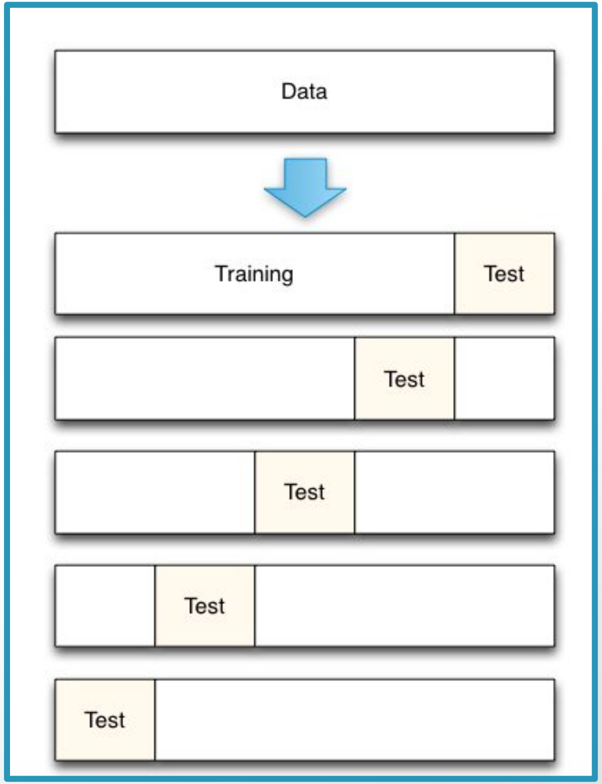
- Roughly 20 percent of the validation set is used to estimate the generalization error.

Cross Validation

- Dividing the data into a fixed training set and a fixed test set does not work with smaller datasets.
- Cross validation procedures are based on the idea of repeating the training and test computation on different randomly chosen subsets or splits of the original dataset.

K-fold Cross validation

- A partition of the dataset is formed by splitting it into k non overlapping subsets
- The test error os then estimated by taking the average test error across k trials.
- On trial i, the i-th subset of the data is used as the test set and the rest is used as training set.


Data
Training
Test
Test
Test
Test
Test

Types of Prediction Error

- Bias Error
- Variance Error
- Irreducible Error

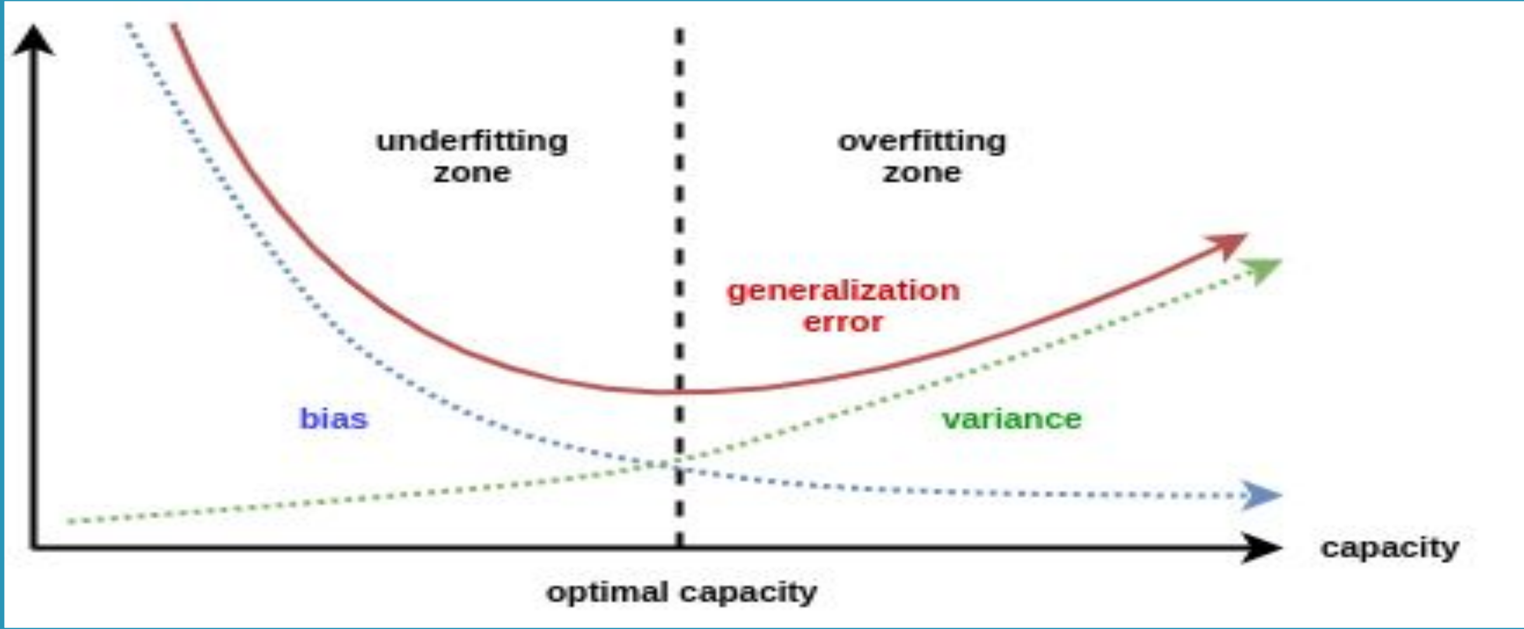
Types of Prediction Error

Variance Error
Variance is the amount that the estimate of the target function will be influenced if specifics of training data was changed.

Bias Error
Bias are the simplifying assumptions made by a model to make the target function easier to learn.

Irreducible Error
The irreducible error cannot be reduced regardless of what algorithm is used.
Possible Cause: Unknown factors

BIAS AND VARIANCE



PRACTICAL ML

Scikit Estimators : Supervised

- model.fit () – Accepts two arguments, the data(x) and the labels (y)
- model.predict () – Predicts the label of a new set of data. Accepts new data (X_new) and returns label for each object in the data.
- model.predict_proba () – Returns the probability of a new observation which has a categorical label.
- model.score () – Returns a score between 0 and 1. The larger the value the better the model fits.

Scikit Estimators : Unsupervised

model.fit () – Accepts a single argument, the data(x)

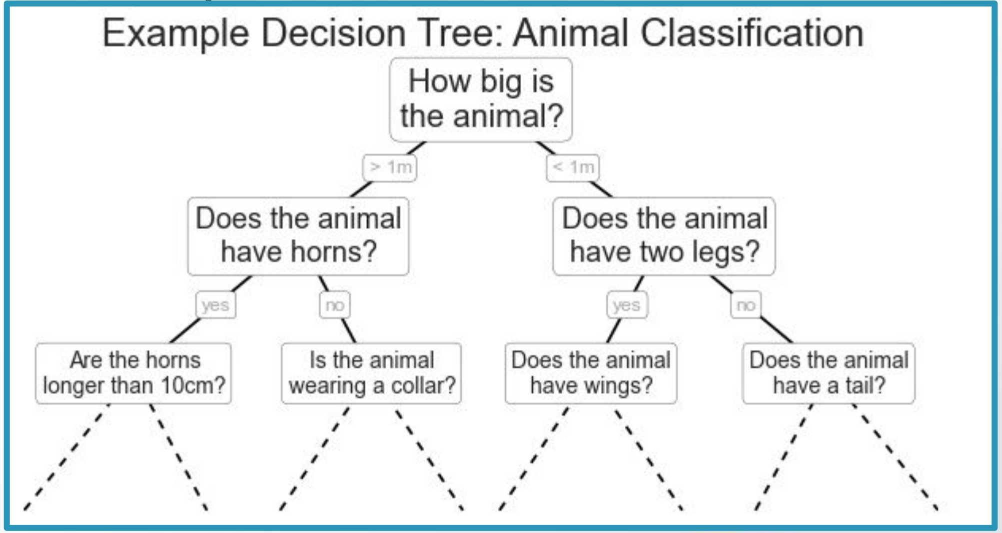
model.predict () – Predicts labels in clustering algorithm

model.transform () – takes one argument x_new and scales the data and returns a new representation of the data.

model.fit_transform () – Performs a fit and the transforms the same input data.

Random Forests :Decision trees

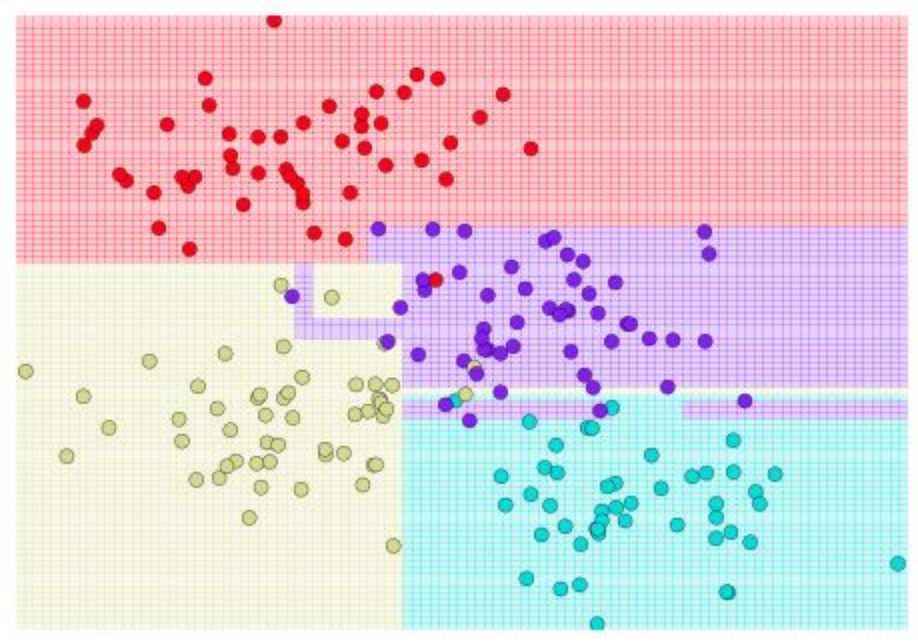
+ A series of questions to zero on a classification



Decision Trees: Binary Splits

+ Ask the Right Questions !!
+ Question = Split
+ Training a Classifier : Algorithm looks at the features and decides which question contains the most information

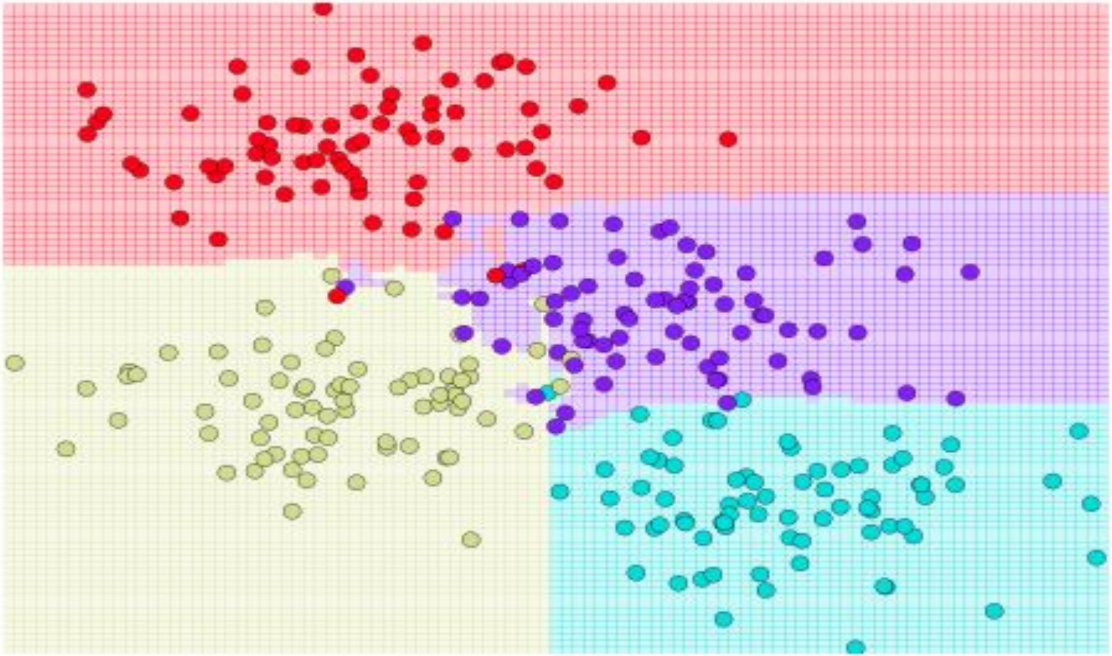
Decision Trees: Overfitting



Random Forests : Ensemble Method

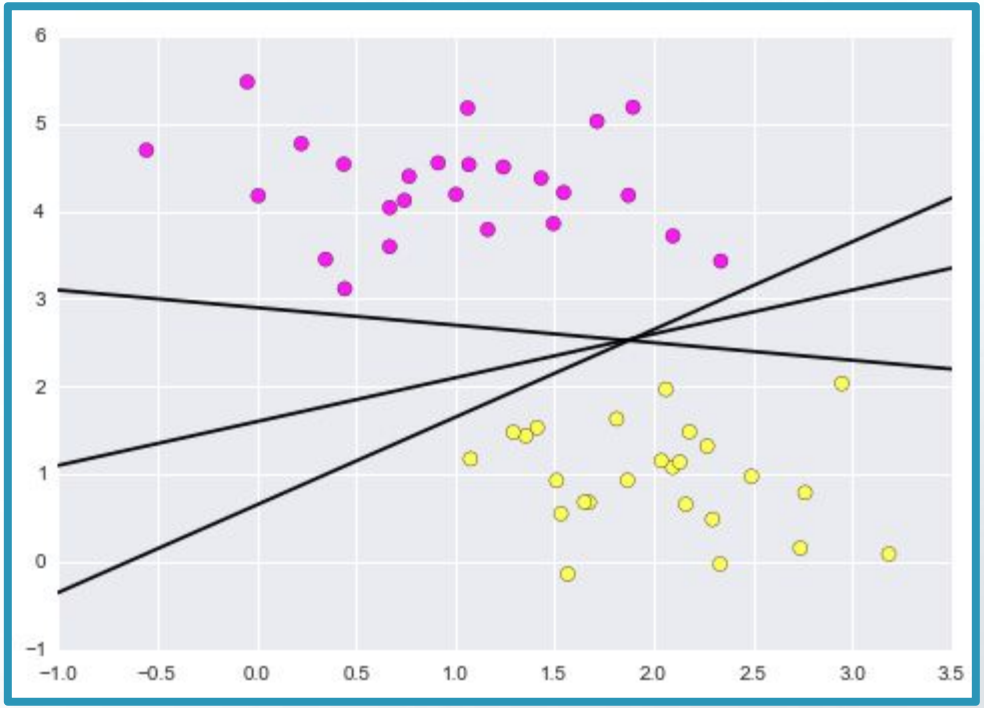
- Avoids Overfitting, by averaging the result of many individual decision trees.
- Classifier : Uses a combined version of all the trees to arrive at a final answer.

```python
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(n_estimators=100, random_state=0)
visualize_tree(clf, X, y, boundaries=False);
```



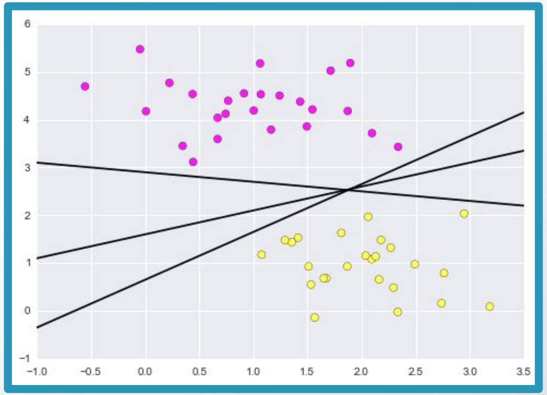
Support Vector Machines

+ Supervised Learning Algorithm used for classification and regression.
+ Discriminative Classifier, used to draw boundaries between clusters of data.
+ Several possibilities to discriminate between the data



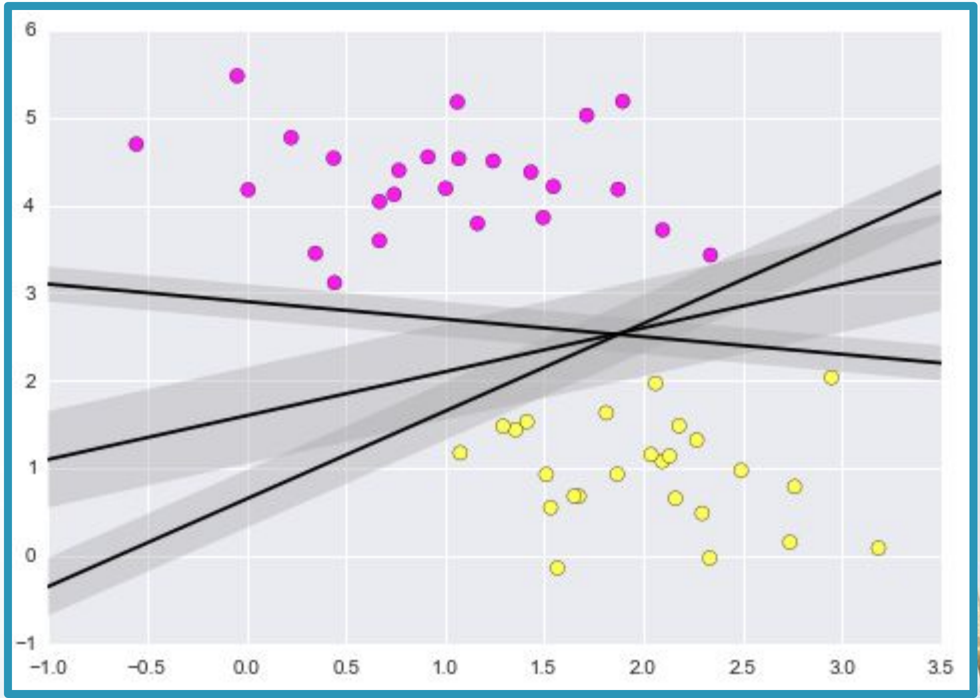
6
5
4
3
2
1
0
−1
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5

How to Improve??



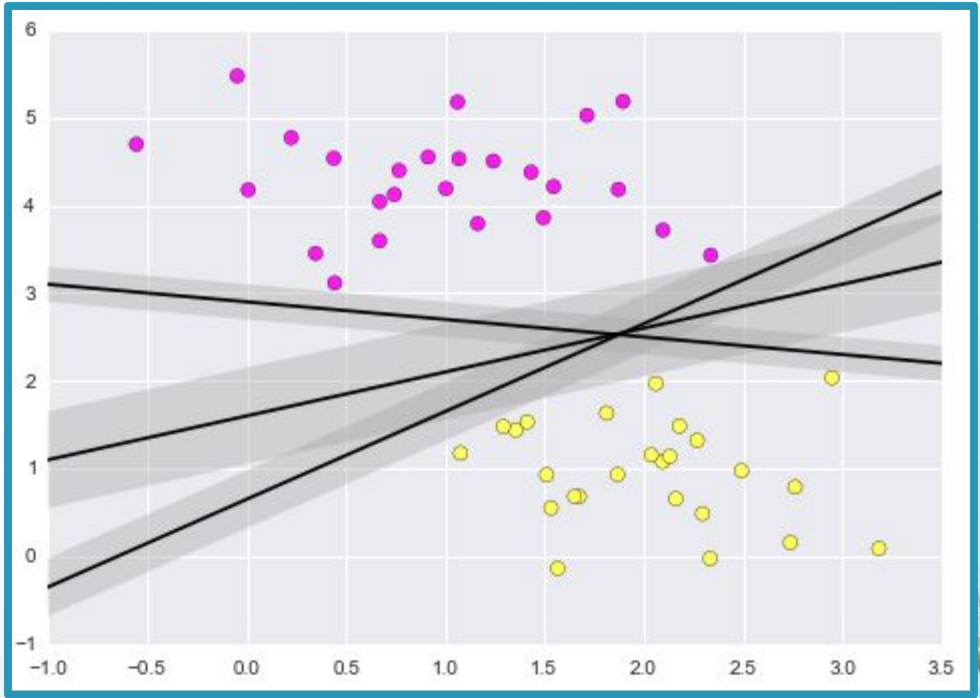
SVM : Maximizing the Margin

→ Consider a Region, Instead of a Line



Choose Region with Maximal Marginal Width

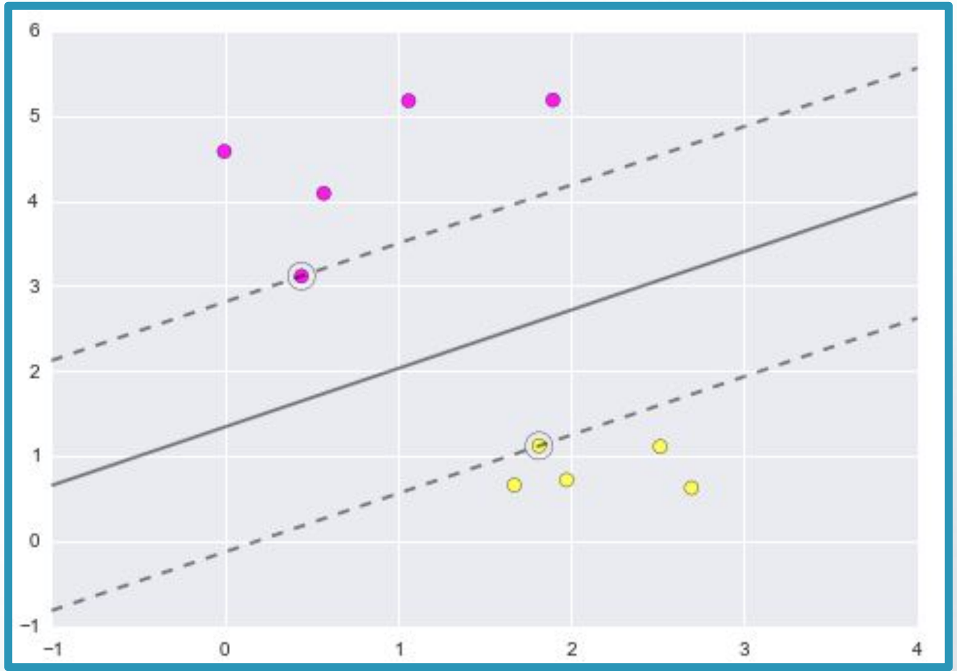
→ Middle fit is best



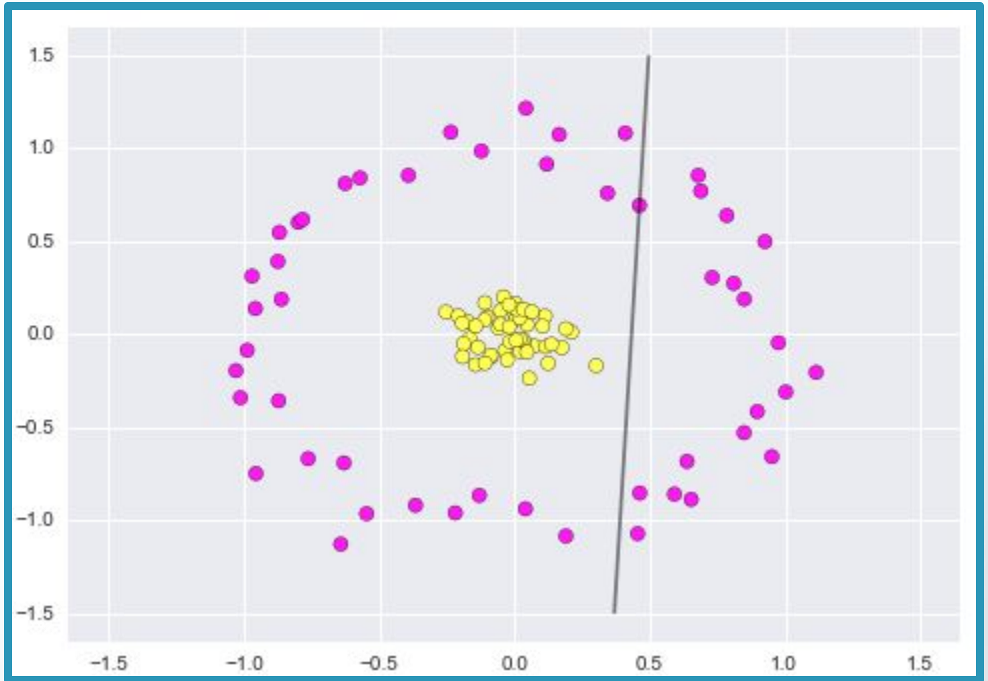
Principle of SVM

- Optimize a Linear Discriminant Model in conjunction with a <u>*margin*</u> . . .
- Representing the perpendicular distance between datasets.

Fitting a Support Vector Machine Classifier



Non Linearly Separable Data



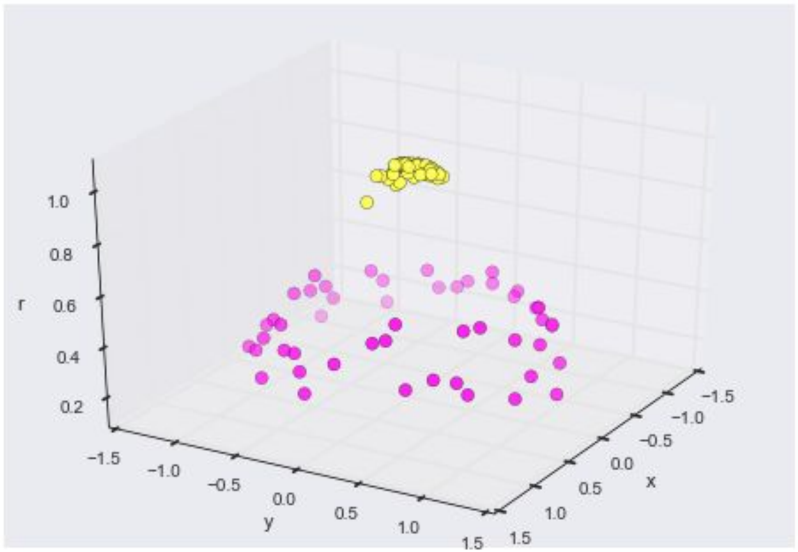
Applying Kernel: rbf Functional Transformation

✗

```
clf = SVC(kernel='linear').fit(X, y)
```

✓

```
clf = SVC(kernel='rbf')
clf.fit(X, y)
```



K-Nearest Neighbour

+ Pick a value for K.
+ Search for the K observations in the training data that are "nearest" to the measurements of the unknown data
+ Use the most popular response value from the K nearest neighbors as the predicted response value for the unknown data
+ KNN would search for one nearest observation and find that exact same observation

K-Nearest Neighbour

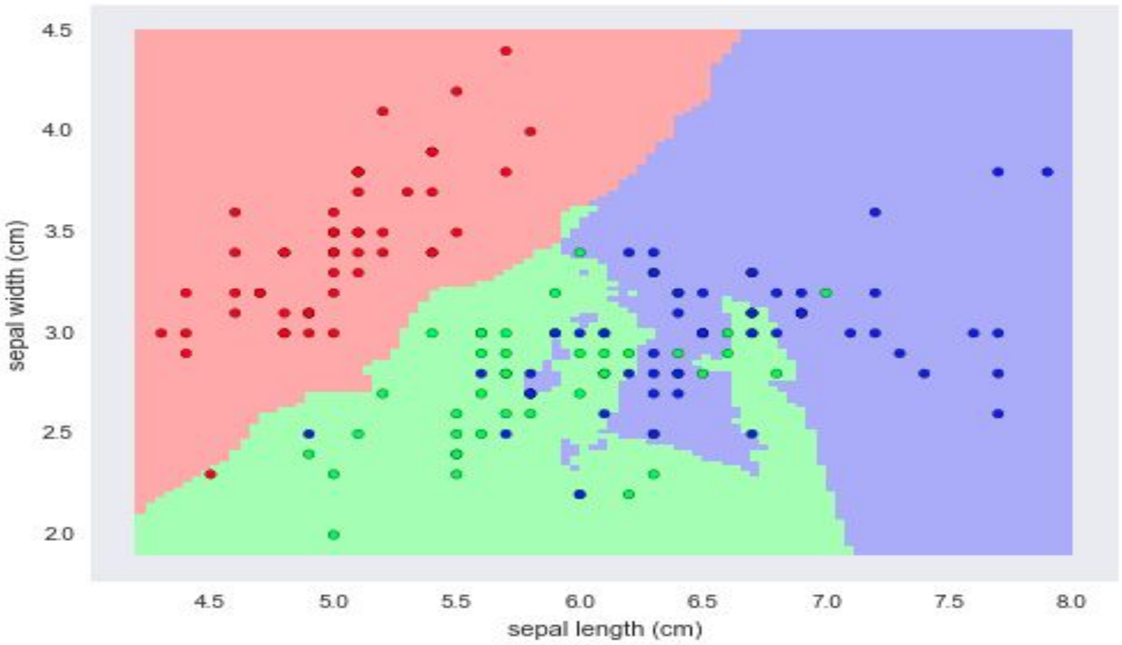
	feature 1	feature 2	response	
	1	2	2	
X_train	3	4	12	y_train
X_test	5	6	30	y_test
	7	8	56	
	9	10	90	

```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print(metrics.accuracy_score(y_test, y_pred))
```

K-Nearest Neighbour - Iris Dataset

```
from sklearn import neighbors, datasets

iris = datasets.load_iris()
X, y = iris.data, iris.target

# create the model
knn = neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform')

# fit the model
knn.fit(X, y)

# What kind of iris has 3cm x 5cm sepal and 4cm x 2cm petal?
X_pred = [3, 5, 4, 2]
result = knn.predict([X_pred, ])

print(iris.target_names[result])
print(iris.target_names)
print(knn.predict_proba([X_pred, ]))

from fig_code import plot_iris_knn
plot_iris_knn()
```

```
['versicolor']
['setosa' 'versicolor' 'virginica']
[[ 0.   0.8  0.2]]
```



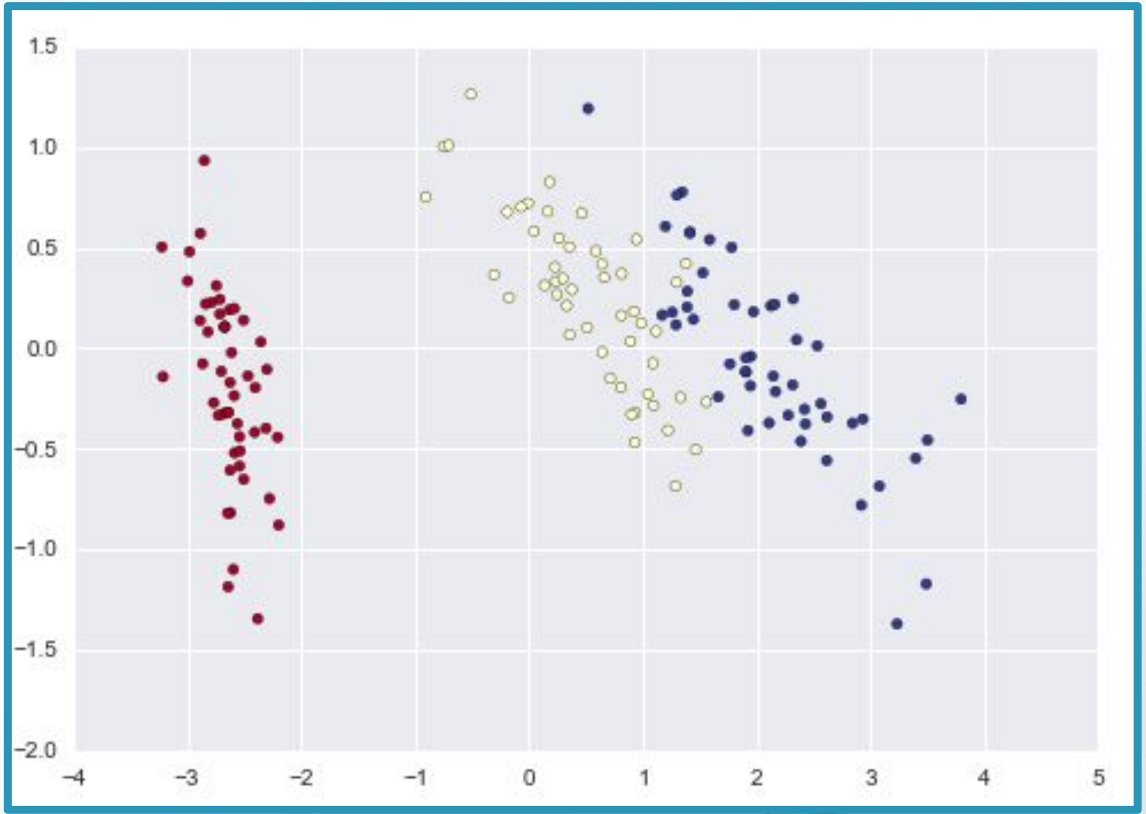
Principal component analysis

Unsupervised Dimensionality Reduction Technique

Objective : Find the combination of variables that explain the most variance. .

```
X, y = iris.data, iris.target
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit(X)
X_reduced = pca.transform(X)
print("Reduced dataset shape:", X_reduced.shape)

import pylab as pl
pl.scatter(X_reduced[:, 0], X_reduced[:, 1], c=y,
           cmap='RdYlBu')

print("Meaning of the 2 components:")
for component in pca.components_:
    print(" + ".join("%.3f x %s" % (value, name)
                     for value, name in zip(component,
                                            iris.feature_names)))
```

```
Reduced dataset shape: (150, 2)
Meaning of the 2 components:
0.362 x sepal length (cm) + -0.082 x sepal width (cm) + 0.857 x petal length (cm) + 0.359 x petal
width (cm)
-0.657 x sepal length (cm) + -0.730 x sepal width (cm) + 0.176 x petal length (cm) + 0.075 x petal
width (cm)
```


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−4
−3
−2
−1
0
1
2
3
4
5

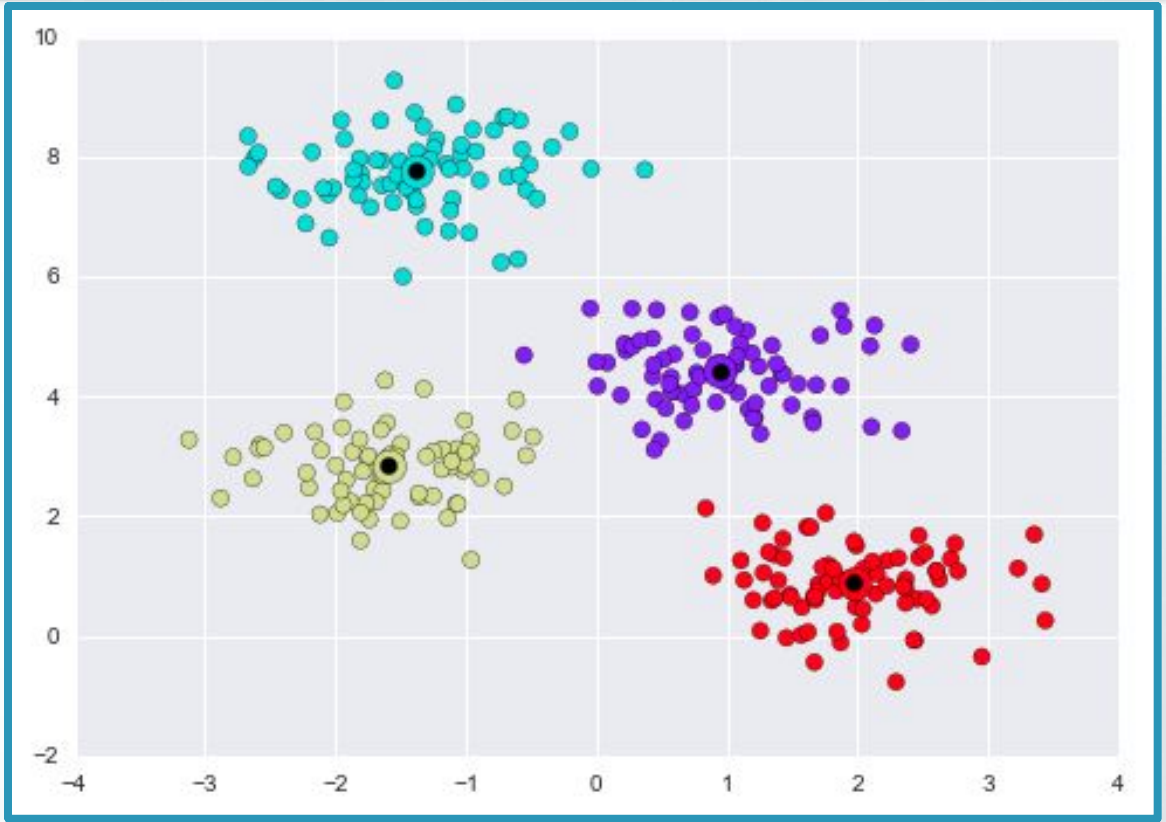
K-Means Clustering

- Unsupervised Learning Algorithm
- Searches for cluster centers such that every point is closest to the cluster center it is assigned to.
- K-Means uses expectation maximization (EM) approach to arrive at the solution.

Expectation Maximization

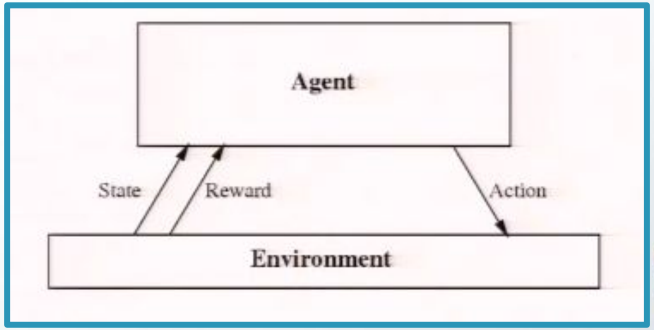
Process : -

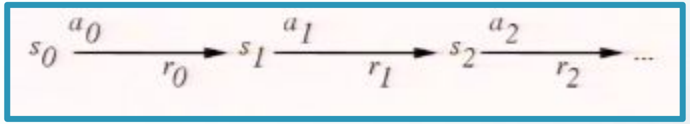
- Guess some cluster centers
- Assign Points to the nearest cluster center
- Set the cluster centers to the mean
- Repeat until Converged.


10
8
6
4
2
0
−2
−4
−3
−2
−1
0
1
2
3
4

Reinforcement Learning

- <u>Goal :</u> Learn to choose actions that maximize Rewards.





Reward Function?

Markov Decision Process

- The future is dependent on the past given the Present action!!

Task: : Learn a policy $\pi : S \rightarrow A$ for choosing actions that maximizes expected value of rewards for every possible states.

Policy : Recommended State-Action Pair

Markov Decision Process

Set of States S

Set of Actions A

At each time agent observes a state $s_t \in S$,

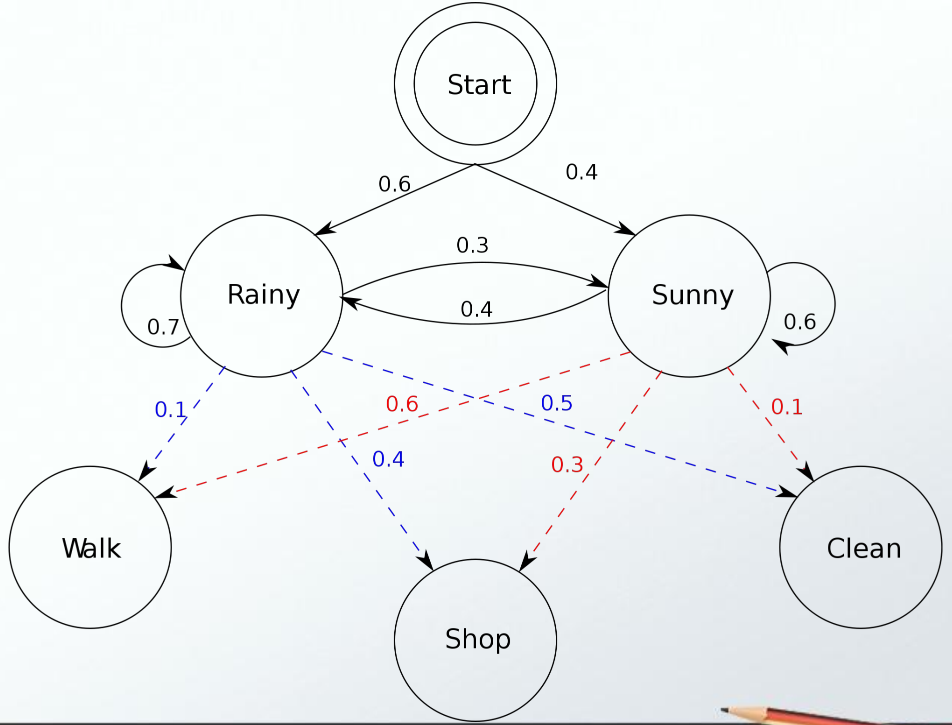
it chooses action $a_t \in A$,

then receives award r_t and state changes to s_{t+1}.

Markov Assumption: $P(s_{t+1} | s_t, a_t, s_{t-1}, a_{t-1}, s_{t-2}, a_{t-2} \ldots) = P(s_{t+1} | s_t, a_t)$

Reward Markov assumption: $P(r_t | s_t, a_t, s_{t-1}, a_{t-1}, s_{t-2}, a_{t-2} \ldots) = P(r_t | s_t, a_t)$

HIDDEN MARKOV PROCESS



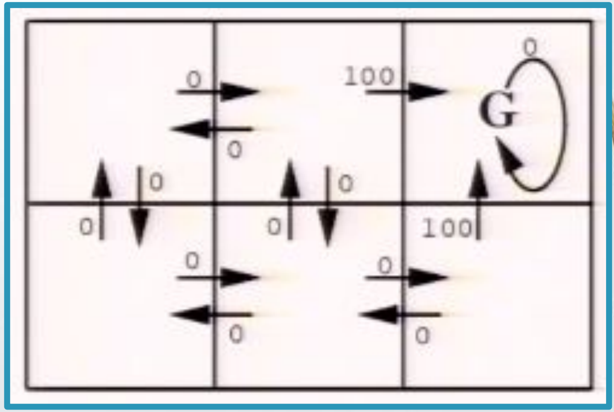
Reinforcement Learning Task For Autonomous Agents

- Executes Action in Environment
- Observes Results
- Learns Policy

Reward, r(s,a) = 0, if not Goal State

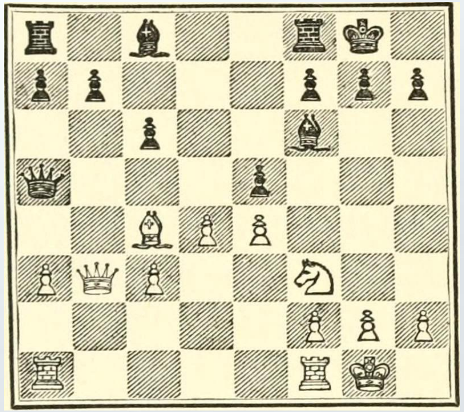
Reward, r(s,a) = 100, if Goal State

Immediate Reward vs Delayed Rewards



Delayed Reward

- When you do not know what your immediate action would lead to later on.
- Example in Chess
- You make many moves and wait until the end for final reward



Value Function & Optimal Policy

- For every policy $\pi : S \rightarrow A$, define a Value Function

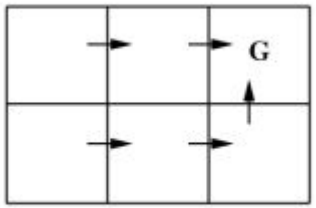
$$\boxed{V^{\pi}(s) = E[\sum_{t=0}^{\infty} \gamma^t r_t]}$$

- If we knew the value of $V(\pi)$ for every state, then we could label the value of each of those states by the sum of expected rewards.

Value Function & Optimal Policy

- While playing chess, if we knew the value (expected reward) of being in every other state, we can compute the value function for each of the subsequent states and find the max(V*(s)).

$$\pi^* = \arg\max_{\pi} V^{\pi}(s), \quad (\forall s)$$



One optimal policy

- Optimal policy = π^*

Autonomous Helicopter Flight Training project

(a)

(b)

Figure 1: (a) Autonomous helicopter. (b) Helicopter hovering under control of learned policy.

Autonomous helicopter flight via reinforcement learning

Andrew Y. Ng
Stanford University
Stanford, CA 94305

H. Jin Kim, Michael I. Jordan, and Shankar Sastry
University of California
Berkeley, CA 94720

Abstract

Autonomous helicopter flight represents a challenging control problem, with complex, noisy, dynamics. In this paper, we describe a successful application of reinforcement learning to autonomous helicopter flight. We first fit a stochastic, nonlinear model of the helicopter dynamics. We then use the model to learn to hover in place, and to fly a number of maneuvers taken from an RC helicopter competition.

CLICK HERE FOR FULL PAPER

Google's DEEPMIND AI – Teaching How To Walk

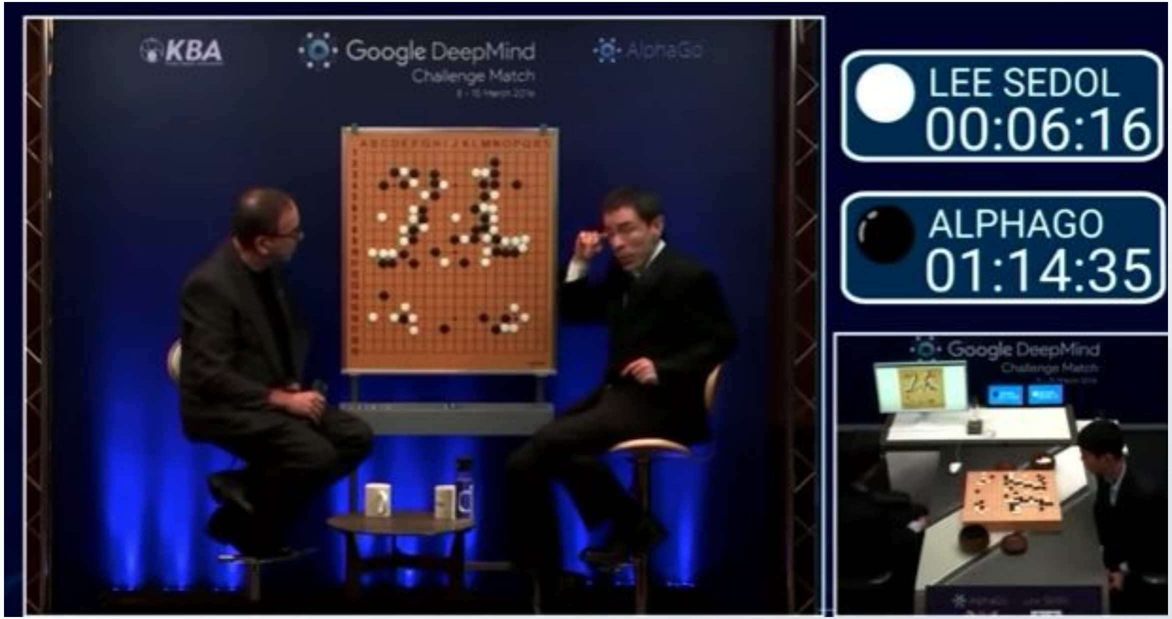
HOW??

Learning Phase – Making numerous random moves and finding winner actions

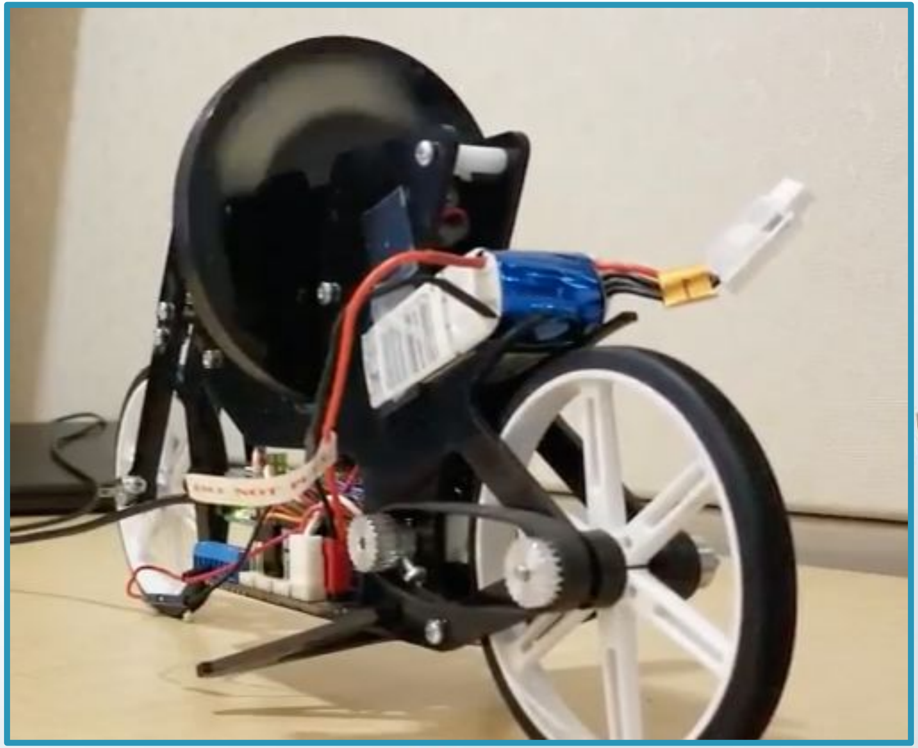
Learning from Every Mistake

Building up information

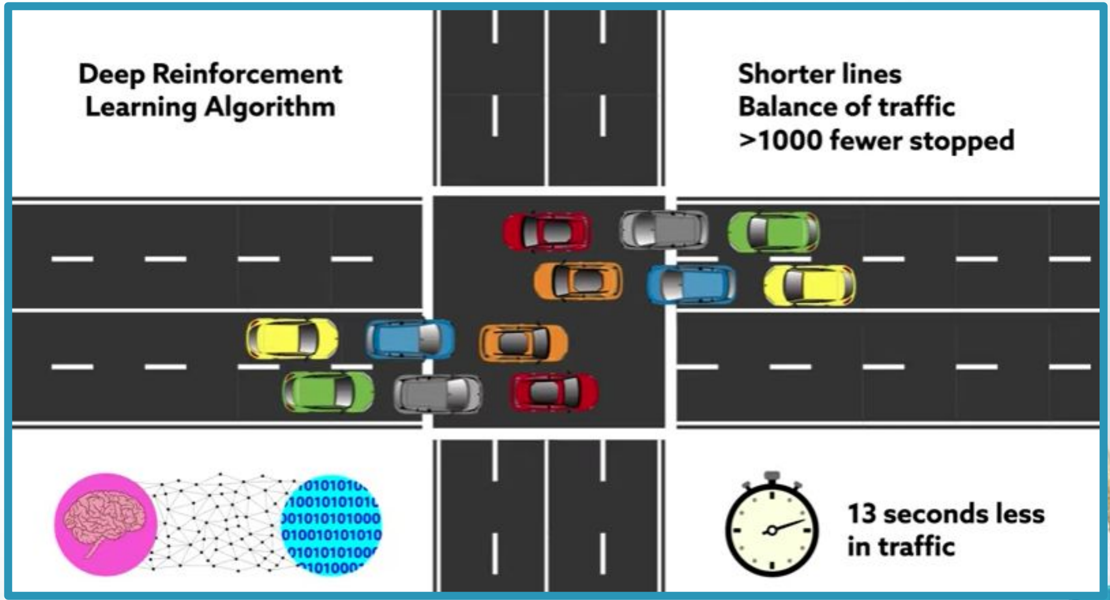
TURN BASED GAMES



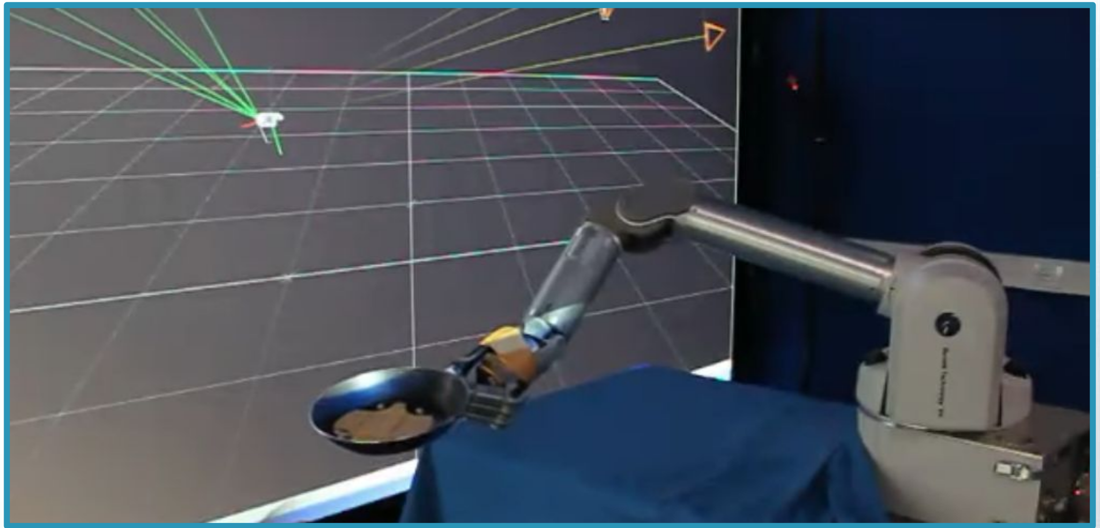
SELF BALANCING BIKE



DIRECTING TRAFFIC



FLIPPING PANCAKES

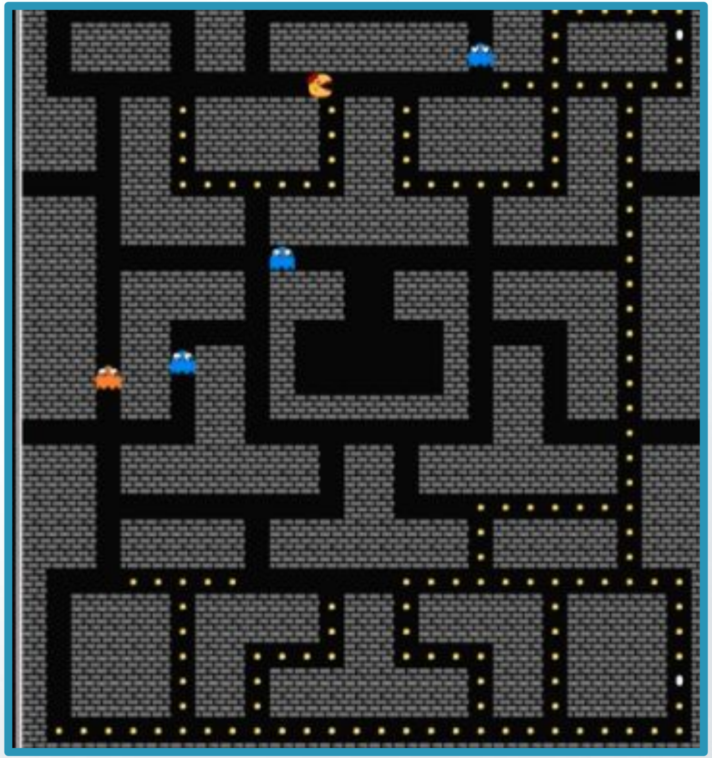


- Using Motion Capture To Evaluate the Rollouts.
- Successfully learned the skill after 50 trials.

LEARNING TENNIS

- Initial Policy (Success Rate 0%)
- First Success (16th Trial)
- Final Policy Rate after 60 trials (79%)

TRAINING PAC-MAN



Reading Material

+ IBM WATSON
  + https://www.theatlantic.com/magazine/archive/2013/03/the-robot-will-see-you-now/309216/

+ STANLEY THE ROBOT
  + https://www.theatlantic.com/magazine/archive/2013/03/the-robot-will-see-you-now/309216/

+ Using Large Scale Unsupervised Learning
  + http://www.cs.cornell.edu/courses/cs6700/2013sp/readings/04-b-Deep-Learning.pdf

Any questions?

You can find me at:

+ @ankita90

Thank You