Introduction to Machine Learning

UG Summer School CSA, IISc 2017

Harit Vishwakarma

Blue Scholar,
IBM Research, Bangalore

July 3, 2017

Disclaimer

All images have been shamelessly downloaded from Google images.

Outline

- Motivation and Applications
- ML Methods or paradigms
    - Supervised Learning
    - Unsupervised Learning
    - Reinforcement Learning
    - Deep Learning
- Further Readings and Career Options.

What to Expect

- It's ok if you don't get the technical/math details now.
- A broad view of problems and solution techniques.
- Get a feel of the field.

Motivating Example

Problem 1

Given a collection of e-mails, search emails containing word "Lottery" at least once.

Motivating Example

Problem 1

Given a collection of e-mails, search emails containing word "Lottery" at least once.

- Use pattern matching algorithms to solve it.

Motivating Example

Problem 1

Given a collection of e-mails, search emails containing word "Lottery" at least once.

- Use pattern matching algorithms to solve it.
- Given a bunch of test cases our algorithm must be 100% correct.

Motivating Example

Problem 2

Given a collection of e-mails, tell which emails are spam and which are not.

Motivating Example

Problem 2

Given a collection of e-mails, tell which emails are spam and which are not.

- Can you use any standard algorithm ??

Motivating Example

Problem 2

Given a collection of e-mails, tell which emails are spam and which are not.

- Can you use any standard algorithm ??
- Given a bunch of test cases our algorithm **may not** be 100% correct.

Motivating Example

Problem 2

Given a collection of e-mails, tell which emails are spam and which are not.

- Can you use any standard algorithm ??
- Given a bunch of test cases our algorithm **may not** be 100% correct.
- Machine needs to know what is spam and what is not.

Motivating Example

Problem 2

Given a collection of e-mails, tell which emails are spam and which are not.

- Can you use any standard algorithm ??
- Given a bunch of test cases our algorithm **may not** be 100% correct.
- Machine needs to know what is spam and what is not.
- Definition of spam may keep evolving.

More Problems

Speech Recognition

Convert spoken language to text.

More Problems

Objection Recognition

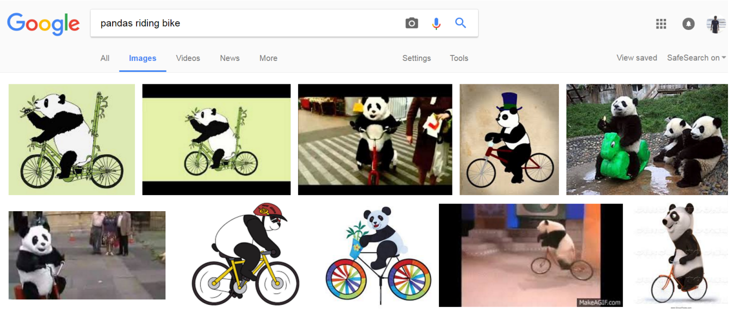
Who is it?

Objection Recognition

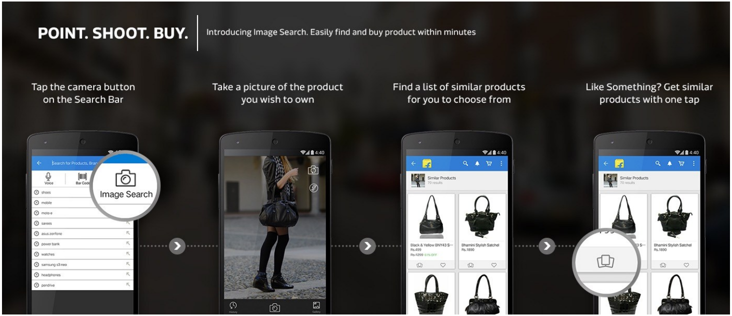
Who is it?

Object Recognition Applications

Google Image Search



Product Search in e-commerce



Product Recommendations

Recommend products to the user that he/she might be interested in buying.

Solving Spam Classification

Approach 1

- Define some rules/criteria. e.g. it should contain "Lottery or Prize" or "it is from an unknown sender" etc.

Solving Spam Classification

Approach 1

- Define some rules/criteria. e.g. it should contain "Lottery or Prize" or "it is from an unknown sender" etc.

Solving Spam Classification

Approach 2

- Suppose you have a collection of emails which are labeled as spam or not spam.

Solving Spam Classification

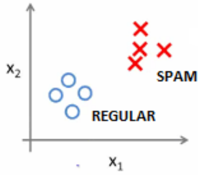
Approach 2

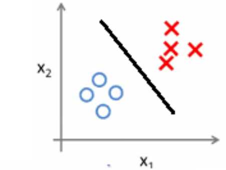
- Suppose you have a collection of emails which are labeled as spam or not spam.
- Write programs to make it figure out the rules from this data.

Solving Spam Classification

Approach 2

- Suppose you have a collection of emails which are labeled as spam or not spam.
- Write programs to make it figure out the rules from this data.
- Such approaches fall under **Supervised Learning**





Solving Spam Classification

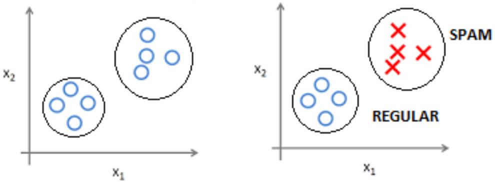
Approach 3

- Find different groups/clusters of emails and analyze them.

Solving Spam Classification

Approach 3

- Find different groups/clusters of emails and analyze them.
- This comes in **Unsupervised Learning**



Solving Spam Classification

Approach 4

- Suppose you have an oracle which gives a reward each time your program makes a correct prediction.

Solving Spam Classification

Approach 4

- Suppose you have an oracle which gives a reward each time your program makes a correct prediction.
- Such approaches fall under **Reinforcement Learning**
- e.g. Chess playing etc.

Supervised Learning

General Setup

We have a set of training examples each with a target label.
Goal is to learn a function which takes an example as input and outputs **accurate** label for it.

Supervised Learning

General Setup

We have a set of training examples each with a target label.
Goal is to learn a function which takes an example as input and outputs **accurate** label for it.

Types of Problems

- Classification Problems
  - Binary : e.g. spam classification
  - Multi-Class : Digit Recognition

Supervised Learning

General Setup

We have a set of training examples each with a target label.
Goal is to learn a function which takes an example as input and outputs **accurate** label for it.

Types of Problems

- Classification Problems
  - Binary : e.g. spam classification
  - Multi-Class : Digit Recognition
- Regression
  - e.g. Cricket Score Prediction.

Supervised Learning Algorithms

- Naive Bayes
- Logistic Regression
- Support Vector Machines (SVMs)
- Decision Trees
- ...

Notations and Prelim

- vectors $\boldsymbol{x}$, $\boldsymbol{w}$
- $\boldsymbol{x} =< x_1, x_2....x_n >$ e.g. $\boldsymbol{x} =< 1, 0.2, 3, 8 >$
- Dot Product of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2 = \boldsymbol{x}_1^T \boldsymbol{x}_2$
- Equation of hyper-plane $\boldsymbol{w}^T \boldsymbol{x} = 0$
- $\|\boldsymbol{w}\| = (w_1^2 + w_2^2 + ... + w_n^2)^{\frac{1}{2}}$

Naive Bayes

- For new instance x we want an estimate of $Pr(y = 1|\boldsymbol{x})$.

Naive Bayes

- For new instance x we want an estimate of $Pr(y = 1|\boldsymbol{x})$.
- From data we can estimate $Pr(\boldsymbol{x}|y)$
  e.g. Fraction of spam emails having word "Lottery".

Naive Bayes

- For new instance x we want an estimate of $Pr(y = 1|\boldsymbol{x})$.
- From data we can estimate $Pr(\boldsymbol{x}|y)$
  e.g. Fraction of spam emails having word "Lottery".
- Now by Bayes rule we have
  $Pr(y = 1|\boldsymbol{x}) = \frac{Pr(\boldsymbol{x}|y=1)Pr(y=1)}{Pr(\boldsymbol{x})}$

Naive Bayes

- For new instance x we want an estimate of $Pr(y = 1|\boldsymbol{x})$.
- From data we can estimate $Pr(\boldsymbol{x}|y)$
  e.g. Fraction of spam emails having word "Lottery".
- Now by Bayes rule we have
  $Pr(y = 1|\boldsymbol{x}) = \frac{Pr(\boldsymbol{x}|y=1)Pr(y=1)}{Pr(\boldsymbol{x})}$
- Suppose $\boldsymbol{x}$ is d dimensional binary vector, then how many parameters we need to estimate?

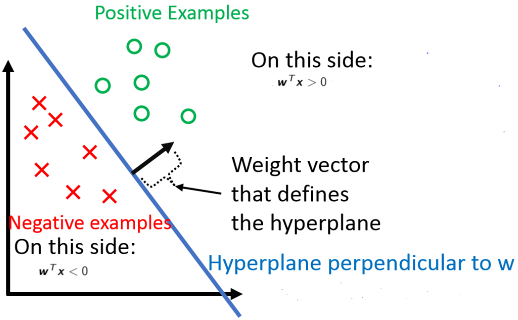
Naive Bayes

- For new instance x we want an estimate of $Pr(y = 1|\boldsymbol{x})$.
- From data we can estimate $Pr(\boldsymbol{x}|y)$
  e.g. Fraction of spam emails having word "Lottery".
- Now by Bayes rule we have
  $Pr(y = 1|\boldsymbol{x}) = \frac{Pr(\boldsymbol{x}|y=1)Pr(y=1)}{Pr(\boldsymbol{x})}$
- Suppose $\boldsymbol{x}$ is d dimensional binary vector, then how many parameters we need to estimate?
- $\mathcal{O}(2^d)$

Naive Bayes

- For new instance x we want an estimate of $Pr(y = 1|\boldsymbol{x})$.
- From data we can estimate $Pr(\boldsymbol{x}|y)$
  e.g. Fraction of spam emails having word "Lottery".
- Now by Bayes rule we have
  $Pr(y = 1|\boldsymbol{x}) = \frac{Pr(\boldsymbol{x}|y=1)Pr(y=1)}{Pr(\boldsymbol{x})}$
- Suppose $\boldsymbol{x}$ is d dimensional binary vector, then how many parameters we need to estimate?
- $\mathcal{O}(2^d)$
- Assume features are mutually independent(**Naive**), then
  $Pr(\boldsymbol{x}) = \prod_{i=1}^{d} Pr(x_i)$ Now we need $\mathcal{O}(d)$ parameters.

Linear Classifier



- $f(\boldsymbol{x}) = sign(\boldsymbol{w}^T \boldsymbol{x})$

Logistic Regression

- We want to learn a linear classifier i.e. $\hat{y} = sign(\boldsymbol{w}^T \boldsymbol{x})$

Logistic Regression

- We want to learn a linear classifier i.e. $\hat{y} = sign(\boldsymbol{w}^T \boldsymbol{x})$
- Assume $P(y_i | \boldsymbol{w}, \boldsymbol{x}_i) = \frac{1}{1+e^{-y_i \boldsymbol{w}^T \boldsymbol{x}_i}}$

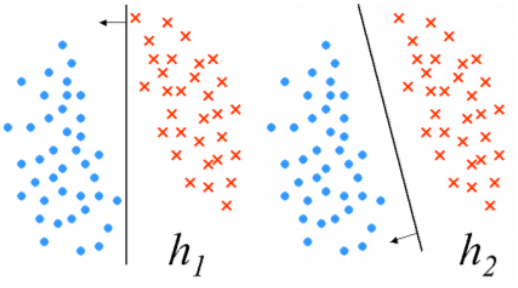
Logistic Regression

- We want to learn a linear classifier i.e. $\hat{y} = sign(\boldsymbol{w}^T \boldsymbol{x})$
- Assume $P(y_i|\boldsymbol{w}, \boldsymbol{x}_i) = \frac{1}{1+e^{-y_i \boldsymbol{w}^T \boldsymbol{x}_i}}$
- $\mathcal{L}(w) = P(y_1, y_2..., y_m|\boldsymbol{x}_1, \boldsymbol{x}_2, ...\boldsymbol{x}_m; \boldsymbol{w})) = \prod_{i=1}^{m} P(y_i|\boldsymbol{x}_i; \boldsymbol{w})$

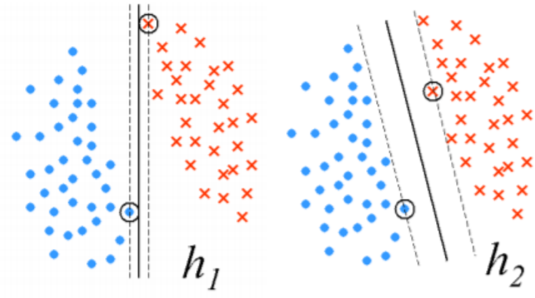
Logistic Regression

- We want to learn a linear classifier i.e. $\hat{y} = sign(\boldsymbol{w}^T \boldsymbol{x})$
- Assume $P(y_i|\boldsymbol{w}, \boldsymbol{x}_i) = \frac{1}{1+e^{-y_i \boldsymbol{w}^T \boldsymbol{x}_i}}$
- $\mathcal{L}(w) = P(y_1, y_2..., y_m|\boldsymbol{x}_1, \boldsymbol{x}_2, ...\boldsymbol{x}_m; \boldsymbol{w})) = \prod_{i=1}^{m} P(y_i|\boldsymbol{x}_i; \boldsymbol{w})$
- $\ln \mathcal{L}(\boldsymbol{w}) = -\sum_{i=1}^{m} \ln(1 + e^{-y_i \boldsymbol{w}^T \boldsymbol{x}_i})$
- $$\underset{\boldsymbol{w}}{\text{maximize}} \quad -\sum_{i=1}^{m} \ln(1 + e^{-y_i \boldsymbol{w}^T \boldsymbol{x}_i})$$

Support Vector Machines



Support Vector Machines



- Distance of a point $\boldsymbol{x}_i$ from the hyper-plane $\gamma_i = \frac{|\boldsymbol{w}^T \boldsymbol{x}_i|}{\|\boldsymbol{w}\|}$

- Distance of a point $\boldsymbol{x}_i$ from the hyper-plane $\gamma_i = \frac{|\boldsymbol{w}^T \boldsymbol{x}_i|}{\|\boldsymbol{w}\|}$
- Margin of a hyper-plane $\gamma = \min \gamma_i$

- Distance of a point $\boldsymbol{x}_i$ from the hyper-plane $\gamma_i = \frac{|\boldsymbol{w}^T \boldsymbol{x}_i|}{\|\boldsymbol{w}\|}$
- Margin of a hyper-plane $\gamma = \min \gamma_i$
- $\min \gamma_i = \min\limits_i y_i \boldsymbol{w}^T \boldsymbol{x}_i = \hat{\gamma}$

- Distance of a point $\boldsymbol{x}_i$ from the hyper-plane $\gamma_i = \frac{|\boldsymbol{w}^T \boldsymbol{x}_i|}{\|\boldsymbol{w}\|}$
- Margin of a hyper-plane $\gamma = \min \gamma_i$
- $\min \gamma_i = \min_i y_i \boldsymbol{w}^T \boldsymbol{x}_i = \hat{\gamma}$
- $$\begin{aligned} & \underset{\boldsymbol{w}}{\text{maximize}} && \frac{\hat{\gamma}}{\|\boldsymbol{w}\|} \\ & \text{subject to} && y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq \hat{\gamma}, \; i = 1, \ldots, m. \end{aligned}$$

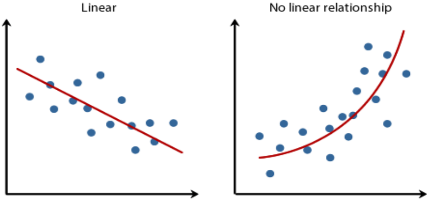
- Distance of a point $\boldsymbol{x}_i$ from the hyper-plane $\gamma_i = \frac{|\boldsymbol{w}^T \boldsymbol{x}_i|}{\|\boldsymbol{w}\|}$
- Margin of a hyper-plane $\gamma = \min \gamma_i$
- $\min \gamma_i = \min_i y_i \boldsymbol{w}^T \boldsymbol{x}_i = \hat{\gamma}$
- $$\begin{array}{ll} \underset{\boldsymbol{w}}{\text{maximize}} & \dfrac{\hat{\gamma}}{\|\boldsymbol{w}\|} \\ \text{subject to} & y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq \hat{\gamma},\ i = 1, \ldots, m. \end{array}$$
- $$\begin{array}{ll} \underset{\boldsymbol{w}}{\text{minimize}} & \dfrac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{subject to} & y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1,\ i = 1, \ldots, m. \end{array}$$
- Dual of this problem is more interesting and popular.

Least Squares Regression

- labels and predictions are real values.
- e.g. Cricket score prediction.

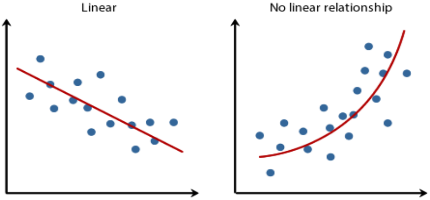
Least Squares Regression

- labels and predictions are real values.
- e.g. Cricket score prediction.



Least Squares Regression

- labels and predictions are real values.
- e.g. Cricket score prediction.



- Fitting a Linear Function:
  Suppose we have m data points of the form $(\boldsymbol{x}_i, y_i)$

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{w}^T\boldsymbol{x}_i - y_i)^2$$

Regularization

- To avoid over-fitting to the training data.

Regularization

- To avoid over-fitting to the training data.
- Enforce some structural constraints on the parameters such as sparsity.

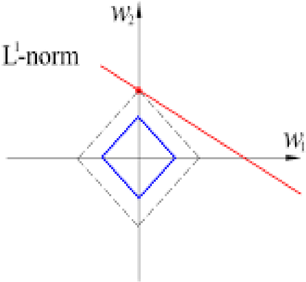
Regularization

- To avoid over-fitting to the training data.
- Enforce some structural constraints on the parameters such as sparsity.
- Some common choices
    - L_1-regularizer: $\|\boldsymbol{w}\|_1 = |w_1| + |w_2| + ... + |w_n|$

Regularization

- To avoid over-fitting to the training data.
- Enforce some structural constraints on the parameters such as sparsity.
- Some common choices
  - L_1-regularizer: $\|\boldsymbol{w}\|_1 = |w_1| + |w_2| + ... + |w_n|$
  - L_2-regularizer: $\|\boldsymbol{w}\|_2 = (w_1^2 + w_2^2 + ... + w_n^2)^{\frac{1}{2}}$
-

Regularization L_1 vs L_2



- L_1 gives sparse solutions. Good for feature selection.
- Optimization is easier with L_2 then L_1.

Regularization of some models

- Lasso (Linear regression with L_1 regularizer)

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{w}^T\boldsymbol{x}_i - y_i)^2 + \lambda_s\|\boldsymbol{w}\|_1$$

Regularization of some models

- Lasso (Linear regression with L_1 regularizer)

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{w}^T\boldsymbol{x}_i - y_i)^2 + \lambda_s\|\boldsymbol{w}\|_1$$

- SVM (with L_2 regularizer)

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}(1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i)_+ + \lambda_s\|\boldsymbol{w}\|_2$$

Regularization of some models

- Lasso (Linear regression with L_1 regularizer)

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}(\boldsymbol{w}^T\boldsymbol{x}_i - y_i)^2 + \lambda_s\|\boldsymbol{w}\|_1$$

- SVM (with L_2 regularizer)

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \frac{1}{m}\sum_{i=1}^{m}(1 - y_i\boldsymbol{w}^T\boldsymbol{x}_i)_+ + \lambda_s\|\boldsymbol{w}\|_2$$

- In general we come across problems of the form:

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \underbrace{\mathcal{L}(\boldsymbol{w})}_{\text{Loss Function}} + \underbrace{\mathcal{R}(\boldsymbol{w})}_{\text{Regularizer}}$$

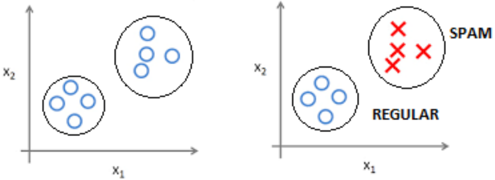
Supervised Learning Workflow

- Get labeled data. Create different features.
- Split the dataset into train and test.
- Choose appropriate algorithm/model and train it on the training data.
- Get the predictions for the test data from the learnt model and measure the performance.
- Cross-Validation is used to tune any hyper-parameters of the model.

Unsupervised Learning

- Labels are not available here.
- Focus is on understanding the data rather than on predictions.
- For example,
  Are there groups of customers?, how many are there?, what are the characteristics of each group? etc.
- Some of the common tasks are:
  - Clustering
  - Dimensionality reductions (PCA etc.)

Clustering



K-means Clustering

Problem

Given a set of observations $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m\}$, goal is to partition them into k clusters $\boldsymbol{C} = \{C_1, C_2 ... C_k\}$ such that the with-in cluster *distance* is minimized. Intuitively similar points should be put in the same cluster. Assume the euclidean distance and $\boldsymbol{\mu}_i$ is the mean of cluster C_i. Then we have the following problem:

K-means Clustering

Problem

Given a set of observations $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m\}$, goal is to partition them into k clusters $\boldsymbol{C} = \{C_1, C_2...C_k\}$ such that the with-in cluster *distance* is minimized. Intuitively similar points should be put in the same cluster. Assume the euclidean distance and $\boldsymbol{\mu}_i$ is the mean of cluster C_i. Then we have the following problem:

$$\underset{\boldsymbol{C}}{\operatorname{argmin}} \quad \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2$$

K-means clustering

Algorithm

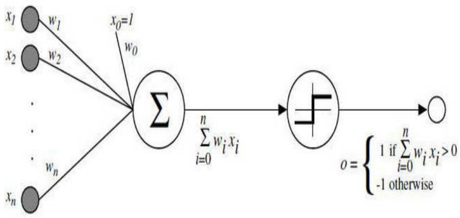
Initialize each $\boldsymbol{\mu}_i$. (randomly/some other way)

- Assignment Step:
  $C_i = \{\boldsymbol{x}_p : \|\boldsymbol{x}_p - \boldsymbol{\mu}_i\|^2 \leq \|\boldsymbol{x}_p - \boldsymbol{\mu}_j\|^2 \quad \forall j, 1 \leq j \leq k\}$
- Update Step:
  $\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x}_p \in C_i} \boldsymbol{x}_p$
- Continue until assignments keep changing

Comments

- Easy to implement, Good to start with.
- Although theoretically converges in $2^{\Omega(\sqrt{m})}$ iterations, but in practice converges in few iterations.
- More Clustering types e.g. Spectral Clustering, Hierarchical Clustering etc.

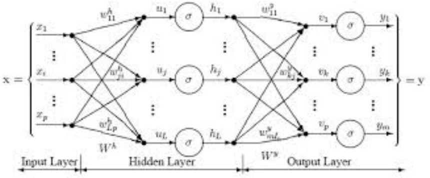
Deep Learning (Perceptron)



- update rule: $\boldsymbol{w} = \boldsymbol{w} + y_i \boldsymbol{x}_i$, if prediction is incorrect.
- Convergence is guaranteed when dataset is linearly separable.
- Simple, Online Algorithm.
- Not very powerful, doesn't work well on complex tasks.

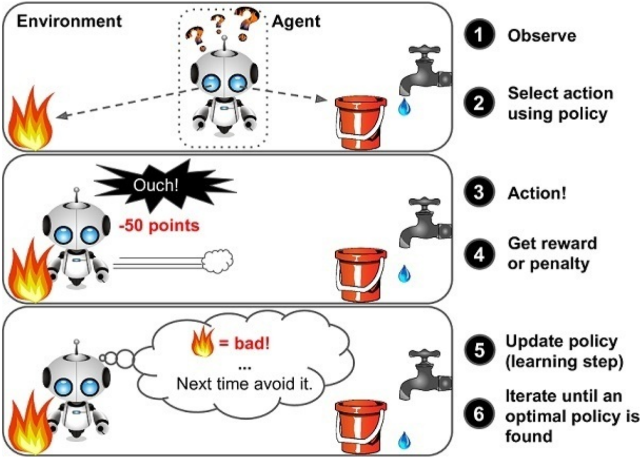
Deep Learning(Lets go deeper :D)

Deep Learning (Multi-Layer Percptrons)



- Performs very well on computer vision tasks (e.g. object-recognition), speech recognition, NLP tasks etc.
- Training time is huge, Hard to explain the predictions.

Reinforcement Learning



Reinforcement Learning

- Successfully applied in robot control, game playing such as (checkers, go etc.)
- Very Interesting and Promising, but too slow to be applied on large scale problems.
- OpenAI gym https://gym.openai.com/

Studying ML

- Probability and Statistics (E0232)
- Linear Algebra (E0219)
- Convex Optimization (E0230)
- Machine Learning (E0270)

Studying ML

- Probability and Statistics (E0232)
- Linear Algebra (E0219)
- Convex Optimization (E0230)
- Machine Learning (E0270)
- Reinforcement Learning
- Deep Learning
- Natural Language Understanding
- Computer Vision

Packages / Tools

- Python: scikit-learn, scipy, numpy, pandas etc.
- Matlab, R, Octave etc.
- At Scale: Apache Spark
- visualizations: d3,
- IDE: Jupyter Notebook

Cloud APIs

- IBM Watson
- Amazon ML
- Microsoft Azure ML

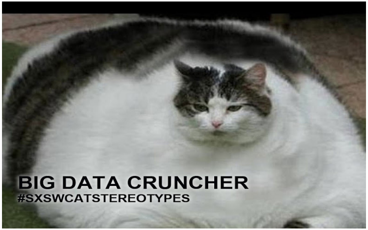
Career Options

Practitioner

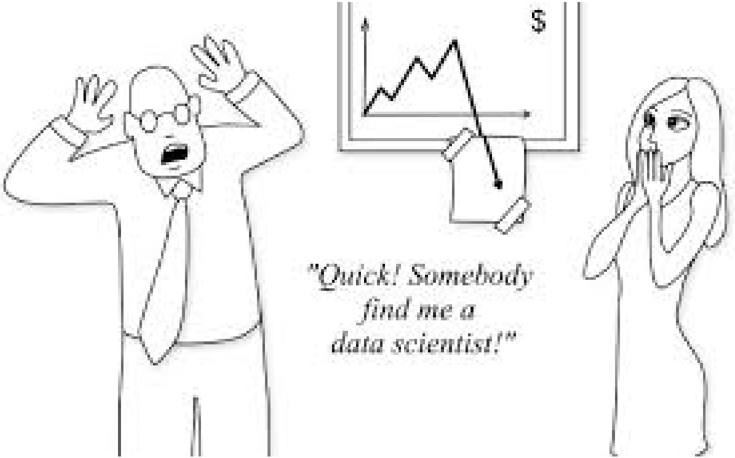
- Work as Data Scientists/ ML Engineer etc.
- You will have real data and real problems. e.g. e-commerce reviews, customer purchase data etc.
- Often data will be huge and have to tackle engineering problems as well.
- Its cool if you enjoy building systems to solve real problems.

Practitioner

- Work as Data Scientists/ ML Engineer etc.
- You will have real data and real problems. e.g. e-commerce reviews, customer purchase data etc.
- Often data will be huge and have to tackle engineering problems as well.
- Its cool if you enjoy building systems to solve real problems.



Huge Demand of Data Scientists



Researcher

- I guess significant progress has been made but we are still far from the so-called Turing test.
- It is a field which is heavily influenced by the application areas. e.g. NLP, Computer Vision, Speech etc.

Researcher

- I guess significant progress has been made but we are still far from the so-called Turing test.
- It is a field which is heavily influenced by the application areas. e.g. NLP, Computer Vision, Speech etc.
- You can choose to be in any application area and /or focus more on theoretical side (optimization etc.)

Researcher

- I guess significant progress has been made but we are still far from the so-called Turing test.
- It is a field which is heavily influenced by the application areas. e.g. NLP, Computer Vision, Speech etc.
- You can choose to be in any application area and /or focus more on theoretical side (optimization etc.)
- Look at papers in conferences such as KDD, WWW, ICML, NIPS, CVPR, ACL etc.

Researcher

- I guess significant progress has been made but we are still far from the so-called Turing test.
- It is a field which is heavily influenced by the application areas. e.g. NLP, Computer Vision, Speech etc.
- You can choose to be in any application area and /or focus more on theoretical side (optimization etc.)
- Look at papers in conferences such as KDD, WWW, ICML, NIPS, CVPR, ACL etc.

Thank You

References I

- Machine Learning Course Link (E0270)
- Probablity and Statistics Course Link (E0232)
- Linear Algebra Course Link (NPTEL)
- Optimization Course (E0230)
- Optimization Course on NPTEL