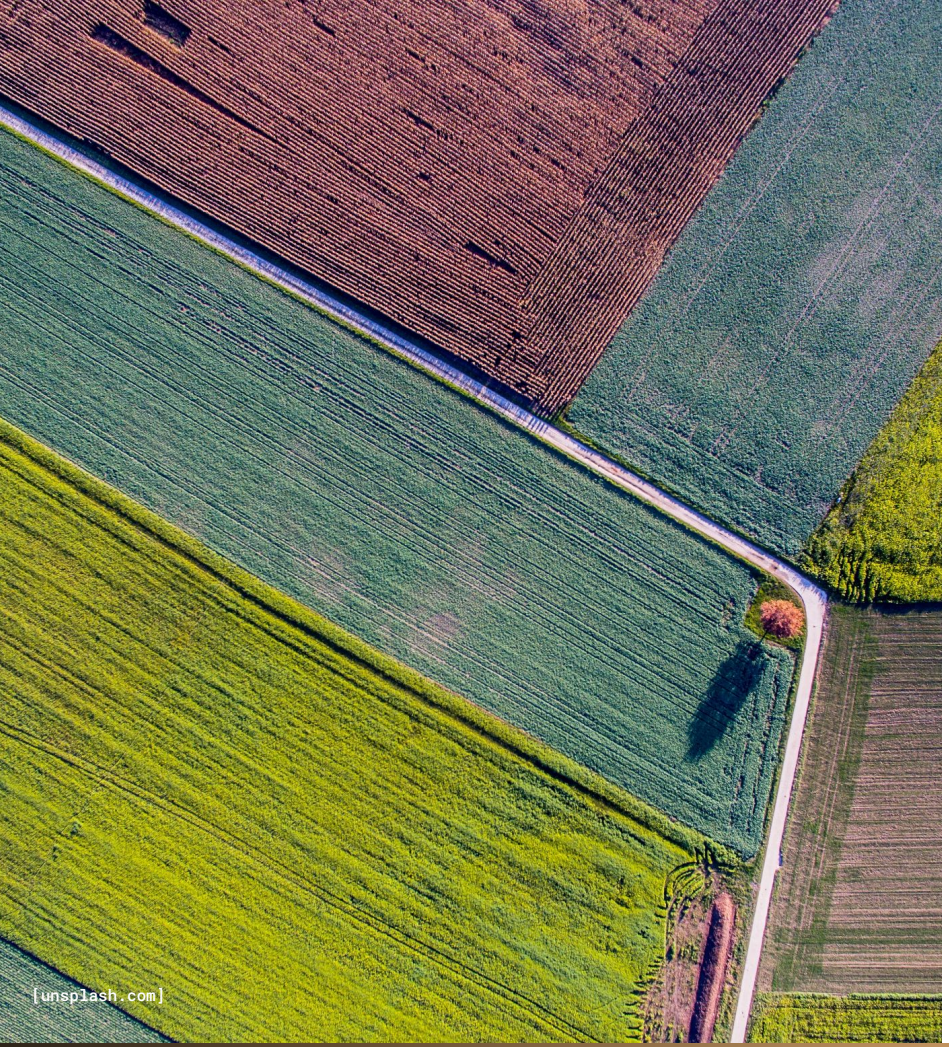
Machine Learning & AI Technology in Plant Sciences

Mary-Ann Blätke,
Jędrzej Jakub Szymański

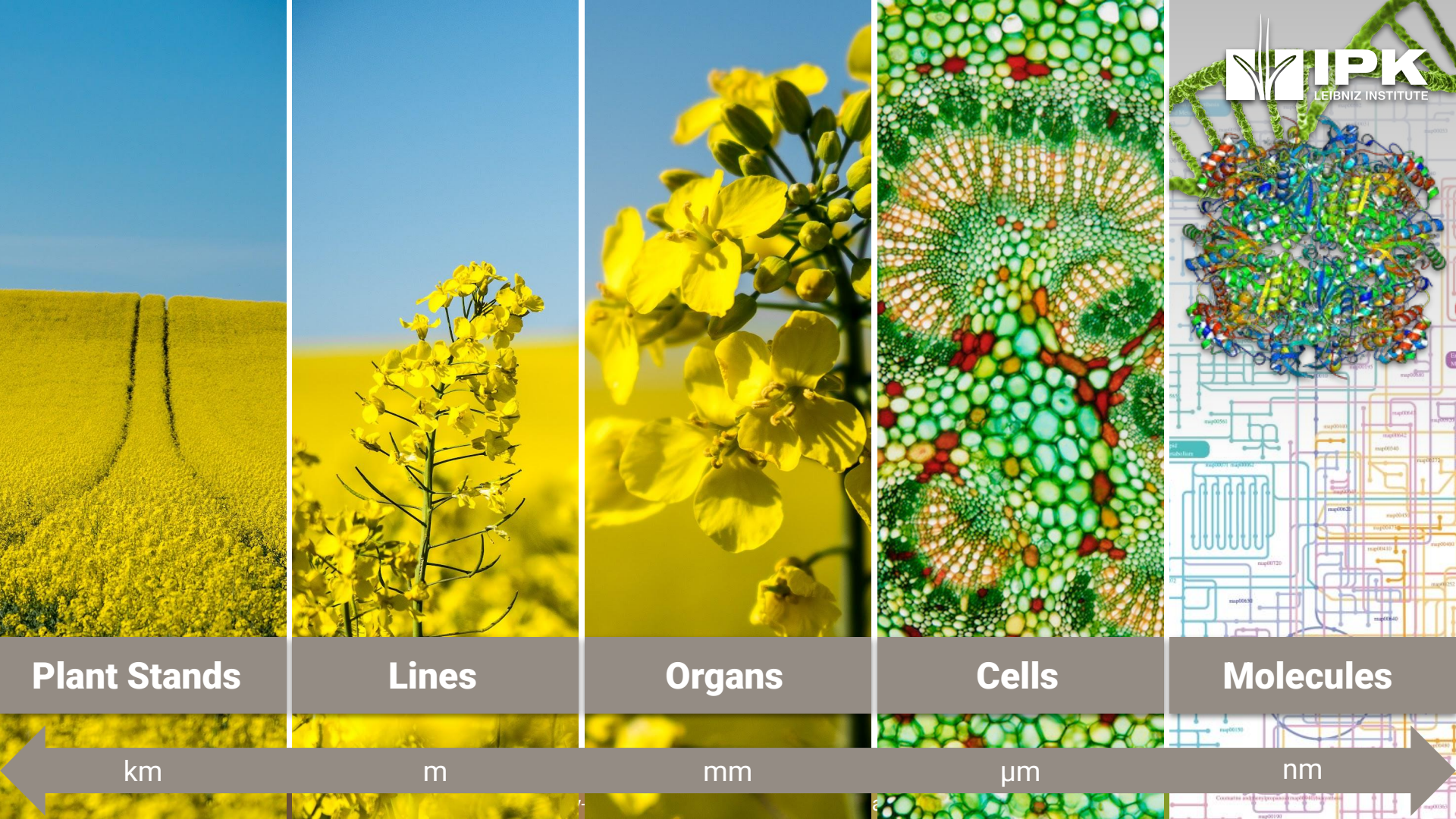

Agriculture is
a major driver
for innovations
in technology
[unsplash.com]

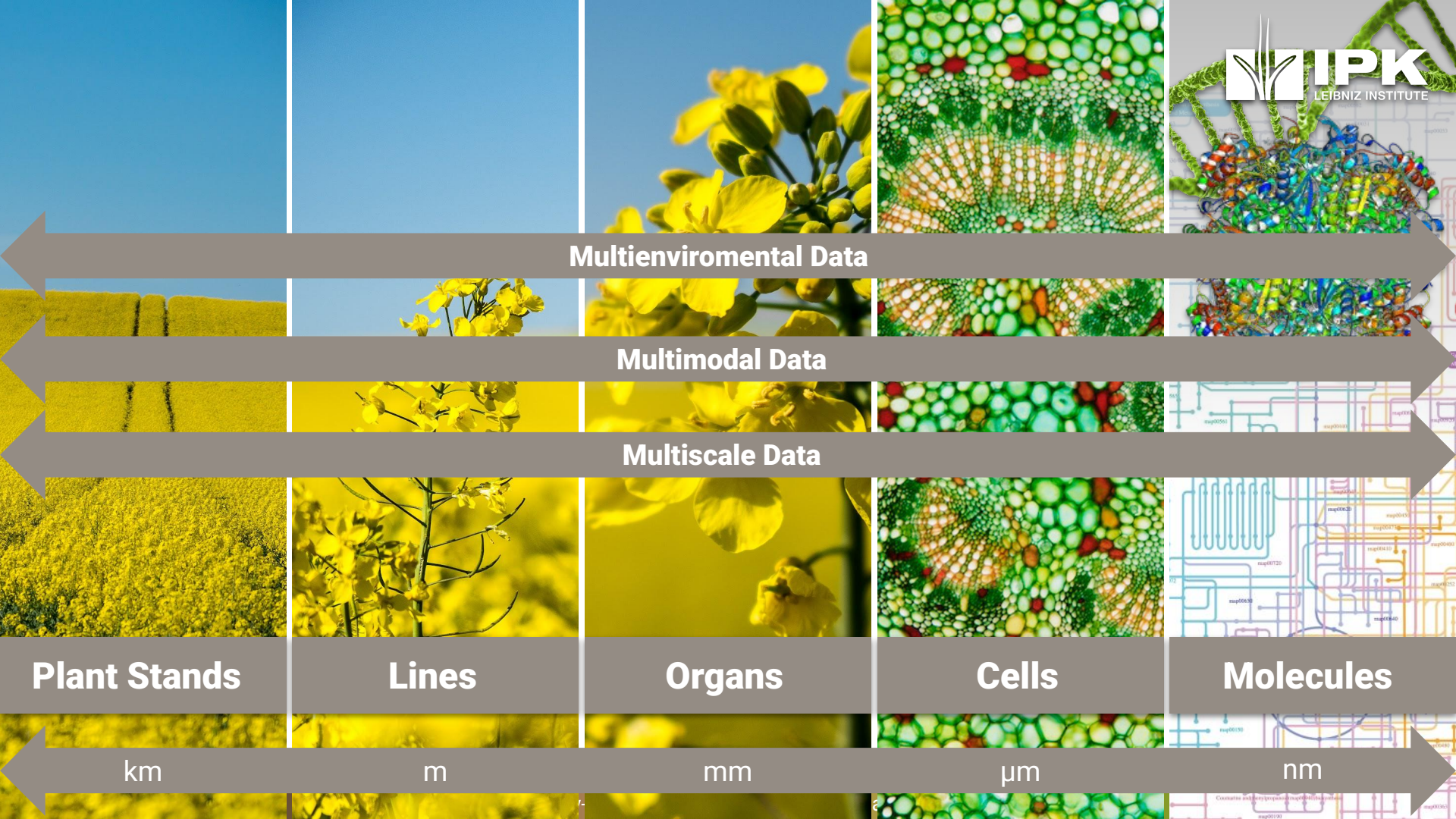


Challenges in Agriculture

- ➔ Food Security
- ➔ Climate Change
- ➔ Environment Protection
- ➔ Biodiversity Conservation
- ➔ Sustainability


Plant science supports the development and breeding of climate-adapted, resistant, and resilient crops to enhance agriculture productivity and sustainability
[unsplash.com]


IPK
LEIBNIZ INSTITUTE
Plant Stands
Lines
Organs
Cells
Molecules
km
m
mm
µm
nm


IPK
LEIBNIZ INSTITUTE
Multienviromental Data
Multimodal Data
Multiscale Data
Plant Stands
Lines
Organs
Cells
Molecules
km
m
mm
µm
nm


Tremendous increase in Data Volume and Diversity
[unsplash.com]

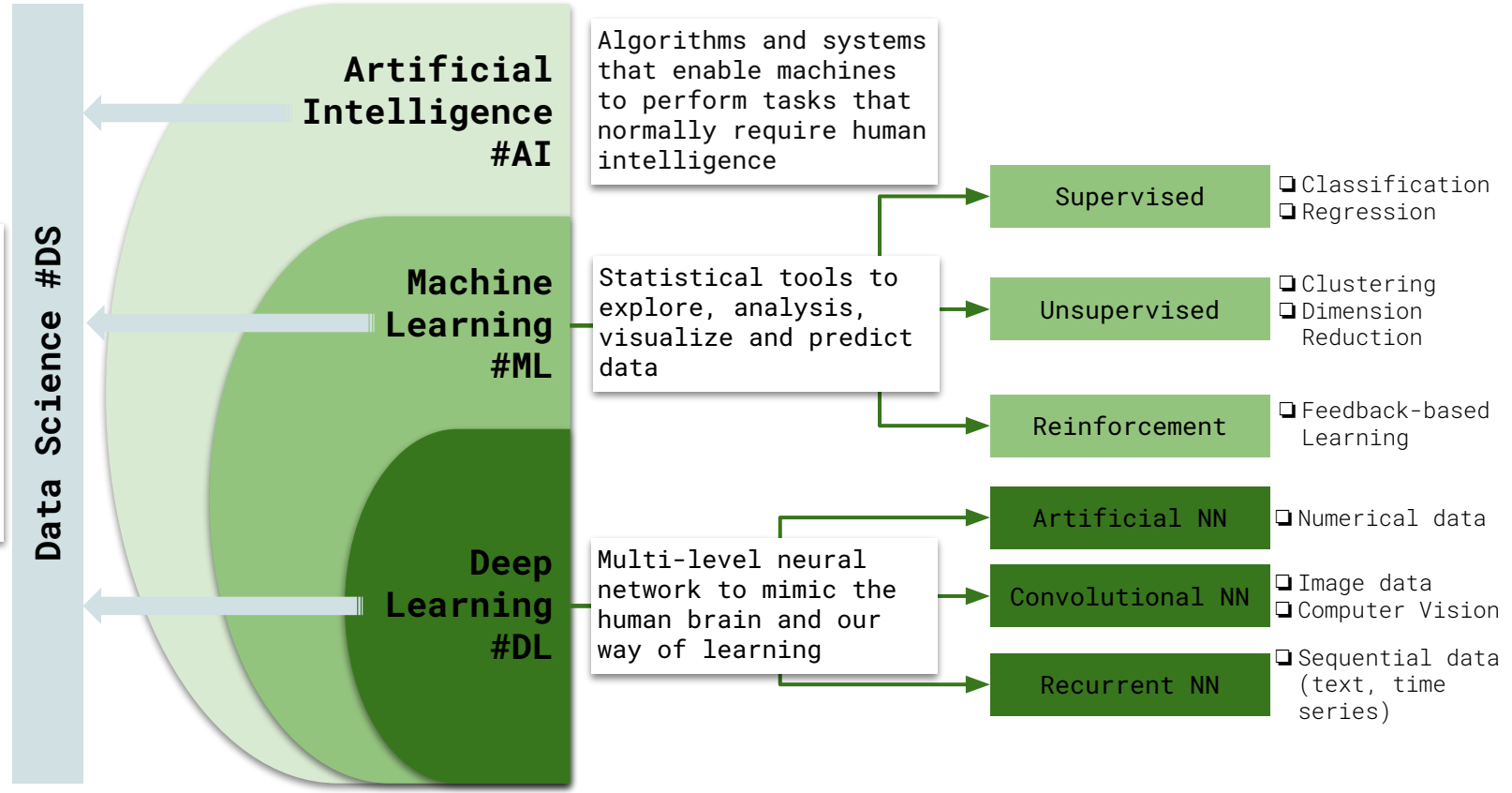
Complexity Demands AI

Data

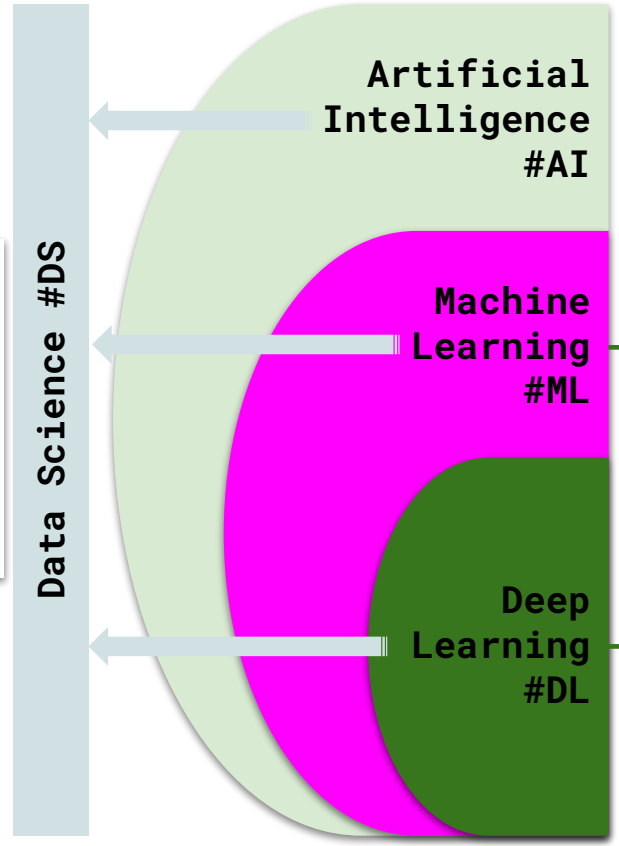
Information

Actionable Knowledge

#AI vs. #ML vs. #DL vs. #DS



#AI vs. #ML vs. #DL vs. #DS



Integration of AI, ML, DL, and maths to extract insights, make predictions, and support decision-making from complex data.

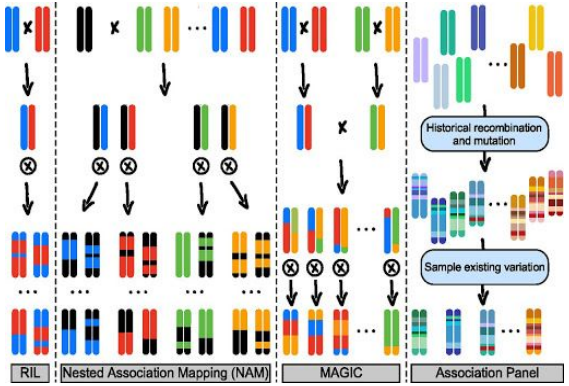
Algorithms and systems that enable machines to perform tasks that normally require human intelligence

Statistical tools to explore, analysis, visualize and predict data

Multi-level neural network to mimic the human brain and our way of learning

- Supervised
  - ❑ Classification
  - ❑ Regression
- Unsupervised
  - ❑ Clustering
  - ❑ Dimension Reduction
- Semisupervised/ reinforced
  - ❑ Feedback-based Learning
- Artificial NN
  - ❑ Numerical data
- Convolutional NN
  - ❑ Image data
  - ❑ Computer Vision
- Recurrent NN
  - ❑ Sequential data (text, time series)

#ML Application in Plant Sciences



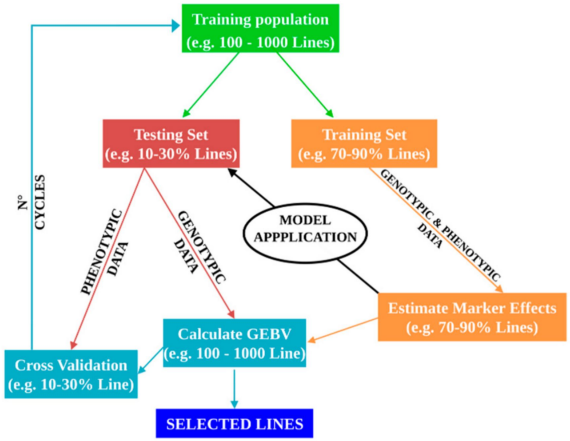
Gage et al. 2020 Plant Cell 32 (7): 2083–93

Genome-Wide Association Study (GWAS), is a method scientists use to find which genes are linked to specific traits in plants comparing the DNA of individuals with and without the trait of interest.

Mapping populations are groups of plants with known genetic differences used by scientists how genes control certain traits.

Gehan MA et al. 2017. *PeerJ* 5:e4088

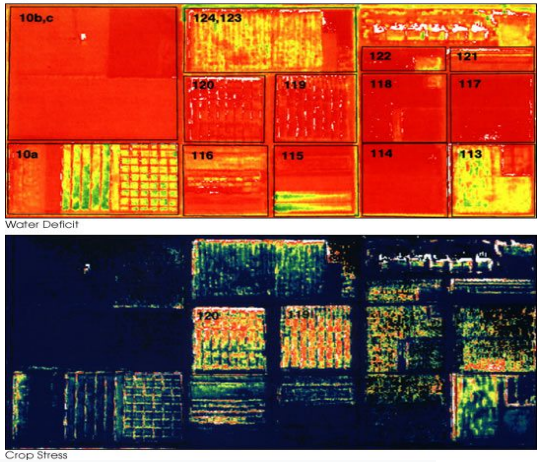
Image analysis of plants involves using computer software to study pictures of plants to measure things like their size, shape, or health.



Cappetta et at. 2020 Plants 9 (9): 1–14.

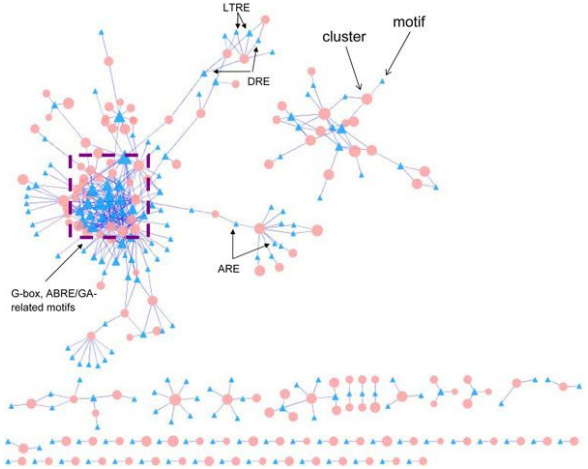
Genomic selection employs #ML to accelerate plant breeding by predicting traits, optimizing parental selection, and speeding up breeding cycles. By analyzing genomic data, #ML enables breeders to make informed decisions, leading to the production of superior crop varieties with desired traits more efficiently.

#ML Application in Plant Sciences



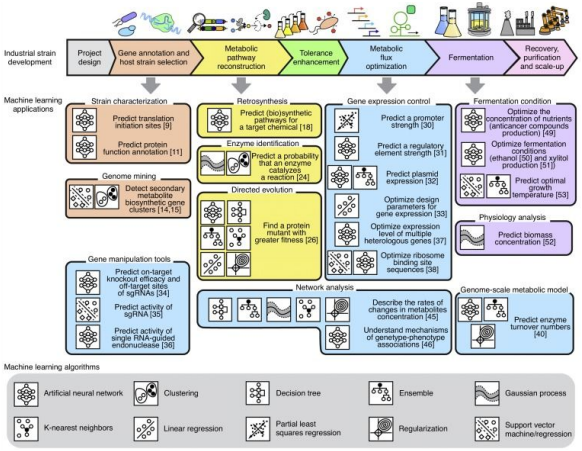
from: NASA Earth Observatory, Image of the Day 2001-01-30 "Precision Farming"

Precision farming optimizes crop production by analyzing data from various sources to make informed decisions, thereby improving efficiency and sustainability in agriculture.



Ruan et al. 2011, BMC Bioinformatics. 12 Suppl 12(Suppl 12):S2.

Regulatory network interference involves predicting how different genes interact and influence each other's activity, giving an idea about complex regulatory mechanisms governing plant growth, development, and responses to environmental changes.

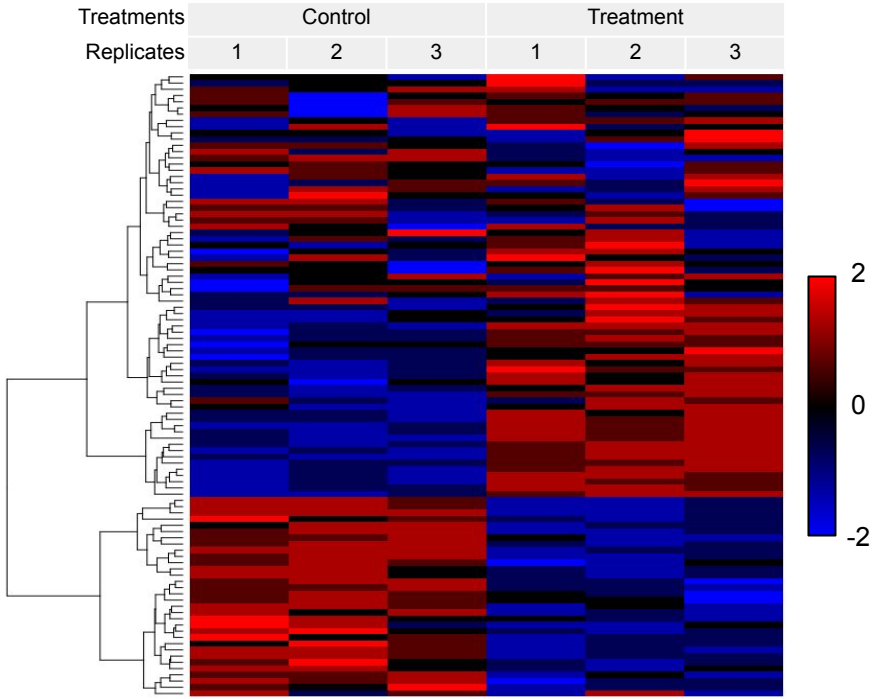


Kim et al. 2020 Current Opinion in Biotechnology 64, 1-9

Metabolic Engineering uses #ML for predicting and optimizing biochemical pathways within plants to enhance desired traits such as yield or nutritional content by identifying key genes/enzymes and aiding the exp. design process

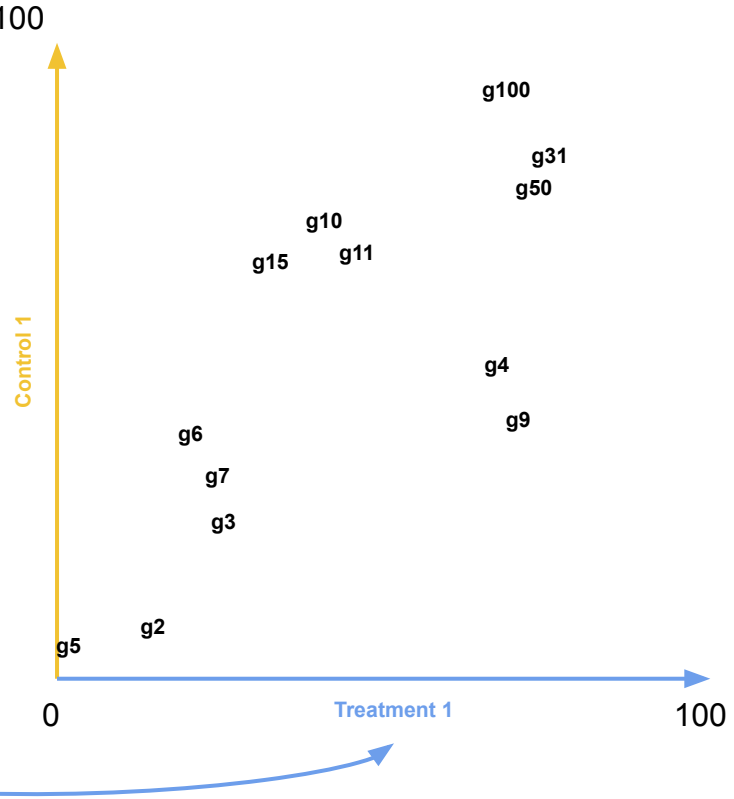
Gene Expression Analysis using Clustering

Treatments	Control			Treatment		
Replicates	1	2	3	1	2	3
g1	345	234	432	311	294	312
g2	2	23	11	12	45	3
g3	32	36	30	46	51	43
g4	79	85	67	54	67	66
g5	1	0	0	4	3	2
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
g100	24	22	17	17		18



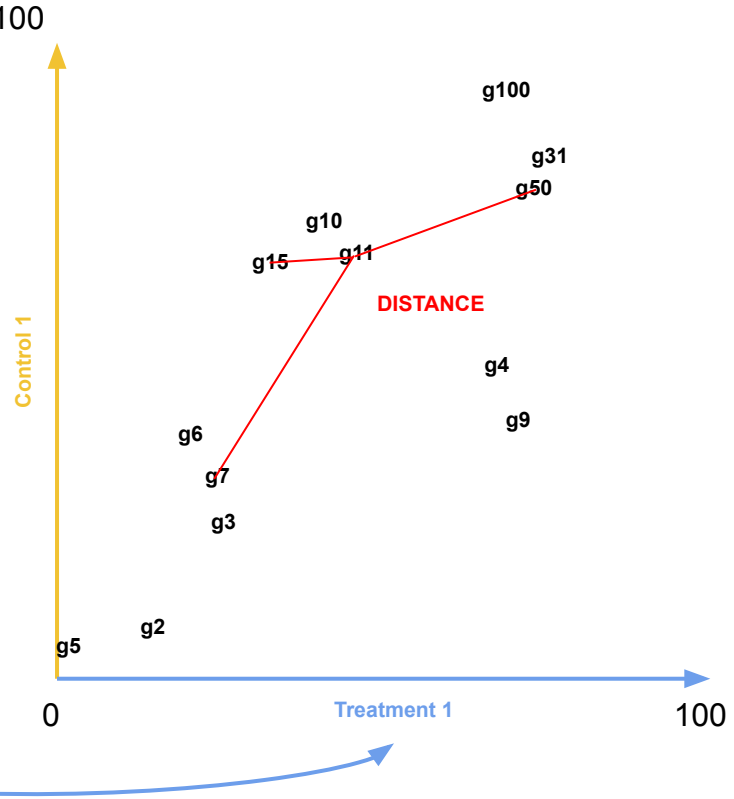
Gene Expression Analysis using Clustering

Treatments	Control			Treatment		
Replicates	1	2	3	1	2	3
g1	345	234	432	311	294	312
g2	2	23	11	12	45	3
g3	32	36	30	46	51	43
g4	79	85	67	54	67	66
g5	1	0	0	4	3	2
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
g100	24	22	17	17		18



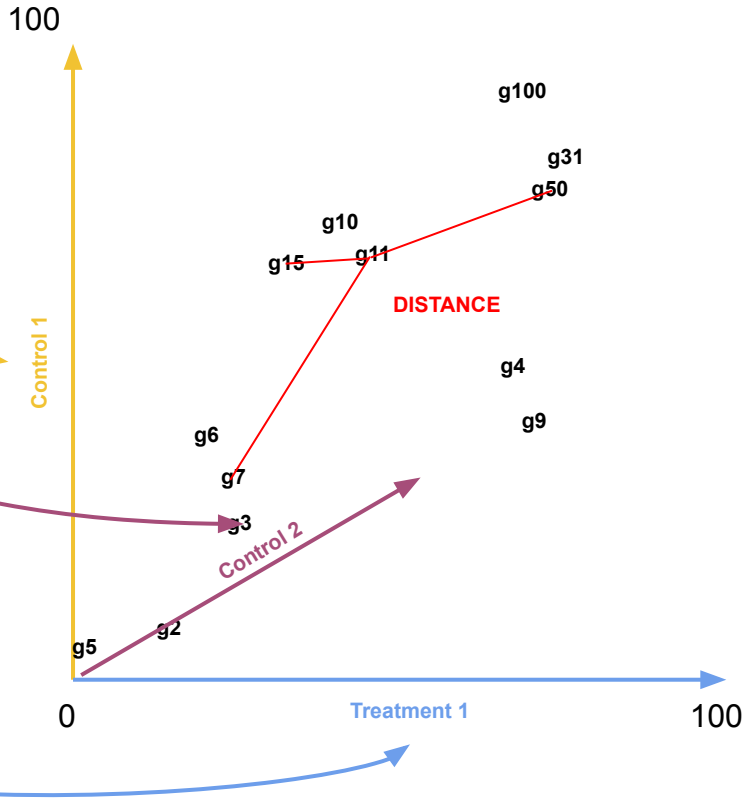
Gene Expression Analysis using Clustering

Treatments	Control			Treatment		
Replicates	1	2	3	1	2	3
g1	345	234	432	311	294	312
g2	2	23	11	12	45	3
g3	32	36	30	46	51	43
g4	79	85	67	54	67	66
g5	1	0	0	4	3	2
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
g100	24	22	17	17		18



Gene Expression Analysis using Clustering

Treatments	Control			Treatment		
Replicates	1	2	3	1	2	3
g1	345	234	432	311	294	312
g2	2	23	11	12	45	3
g3	32	36	30	46	51	43
g4	79	85	67	54	67	66
g5	1	0	0	4	3	2
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
g100	24	22	17	17		18



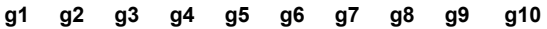
Gene Expression Analysis using Clustering

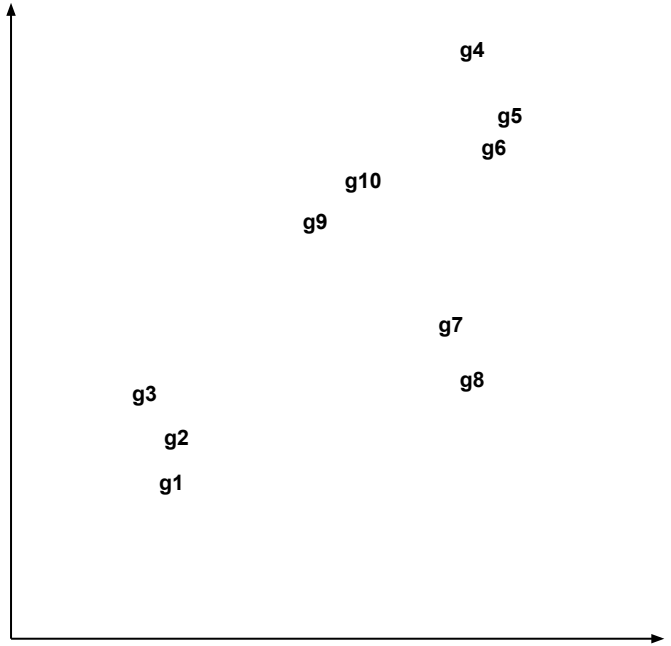
Treatments	Control			Treatment		
Replicates	1	2	3	1	2	3
g1	345	234	432	311	294	312
g2	2	23	11	12	45	3
g3	32	36	30	46	51	43
g4	79	85	67	54	67	66
g5	1	0	0	4	3	2
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
g100	24	22	17	17		18

Distance matrix

Replicates	g1	g2	g3	g4	g5	…	…	…	…	g100
g1	0	0.1	0.3	0.7	0.2	…	…	…	…	0.5
g2	0.1	0	0.5	0.2	0.5	…	…	…	…	0.2
g3	0.3	0.5	0	0.1	0.4	…	…	…	…	0.5
g4	0.7	0.2	0.1	0	0.9	…	…	…	…	0.7
g5	0.2	0.5	0.4	0.9	0	…	…	…	…	0.2
…	…	…	…	…	…	0	…	…	…	0.1
…	…	…	…	…	…	…	0	…	…	0.2
…	…	…	…	…	…	…	…	0	…	0.1
…	…	…	…	…	…	…	…	…	0	0.9
g100	0.5	0.2	0.5	0.7	0.2	0.1	0.2	0.1	0.9	0

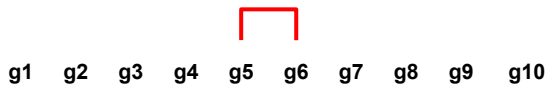
Gene Expression Analysis using Clustering

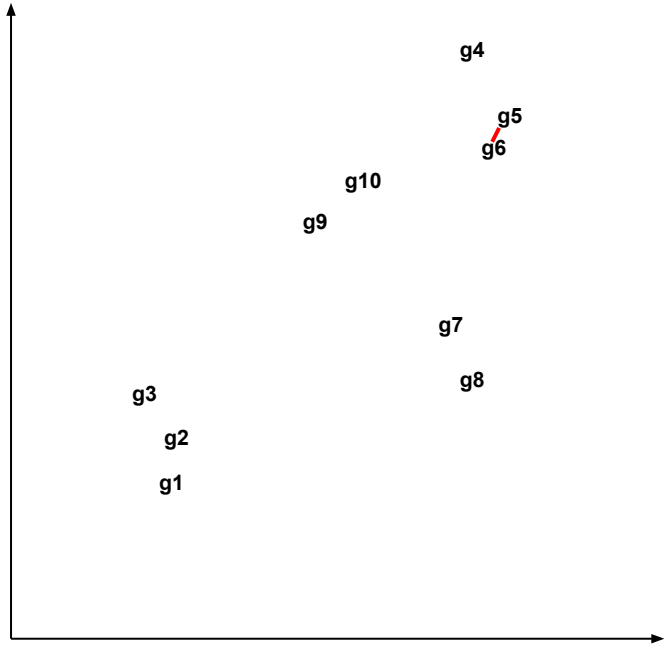




Gene Expression Analysis using Clustering

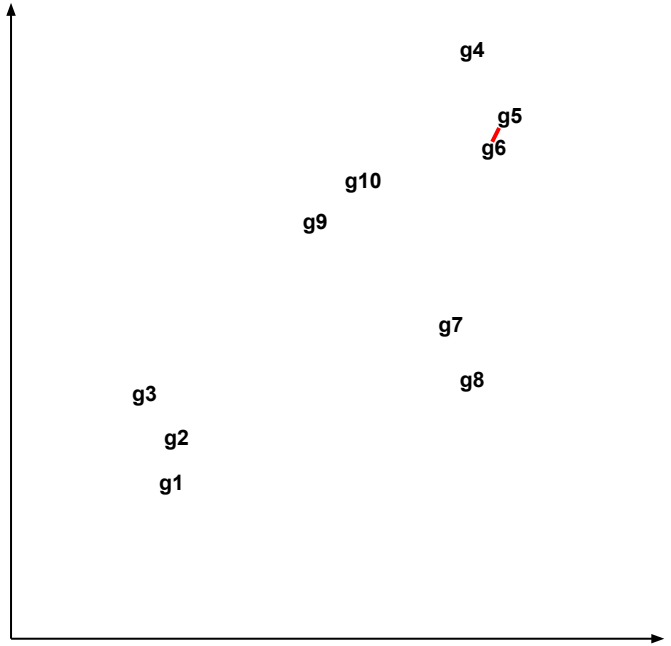
From the Distance Matrix find genes with the minimal distance and link them!



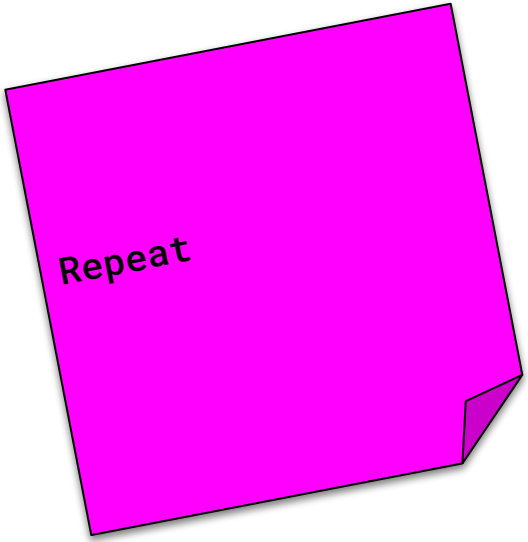


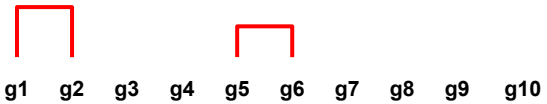
Gene Expression Analysis using Clustering

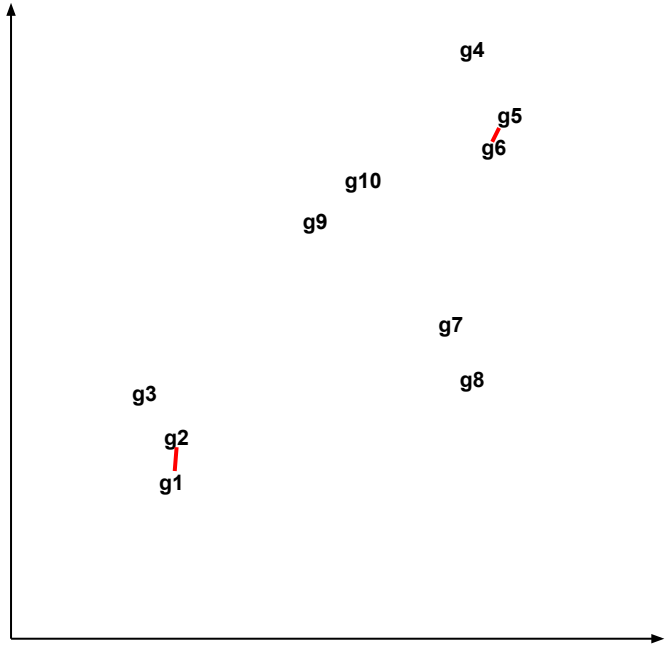
Recompute the distance matrix by agglomerating the linked genes.



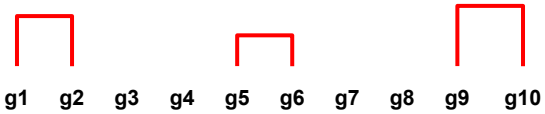
Gene Expression Analysis using Clustering

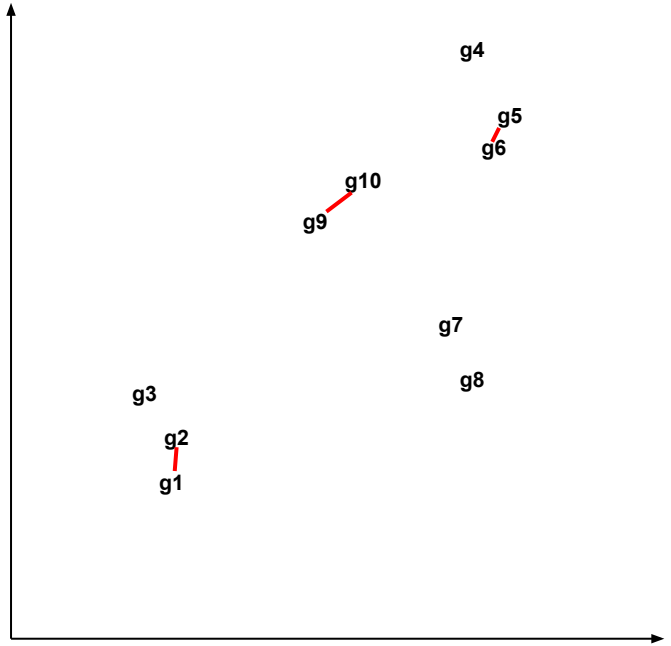




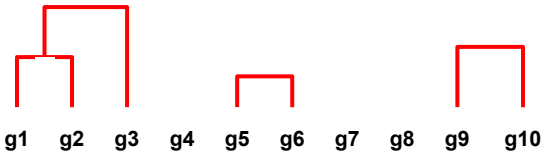


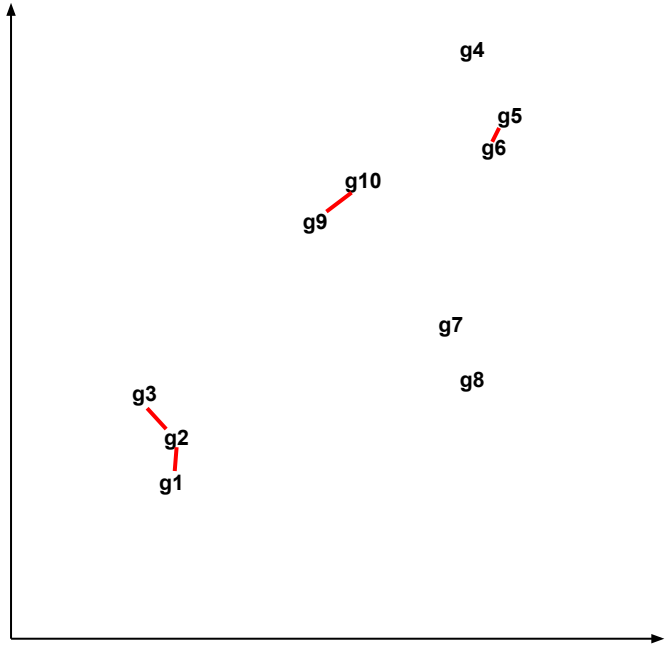
Gene Expression Analysis using Clustering



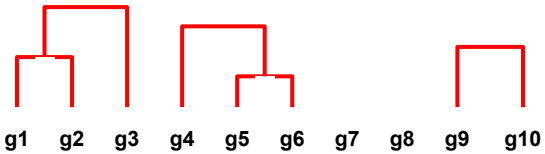


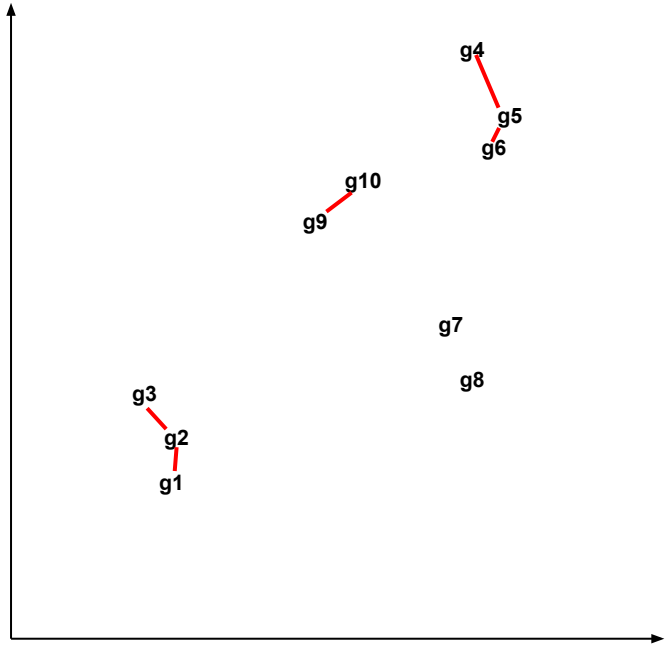
Gene Expression Analysis using Clustering



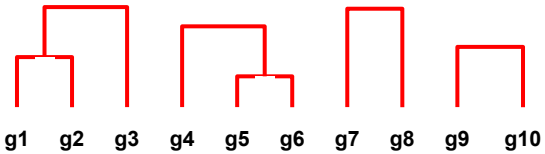


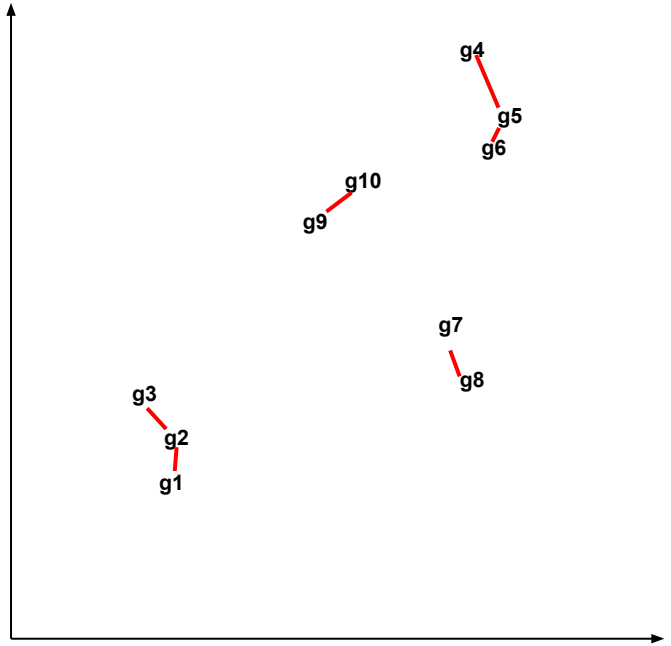
Gene Expression Analysis using Clustering



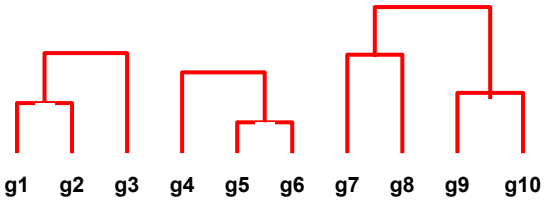


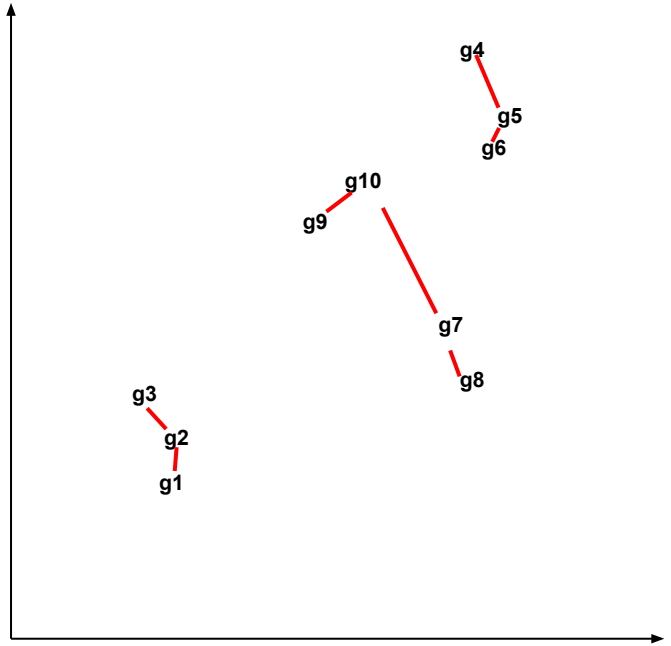
Gene Expression Analysis using Clustering



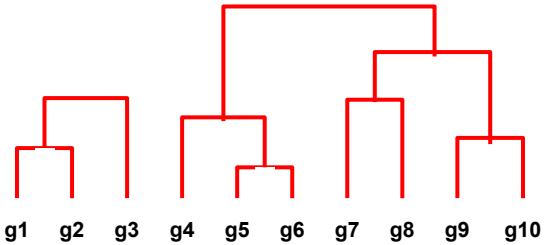


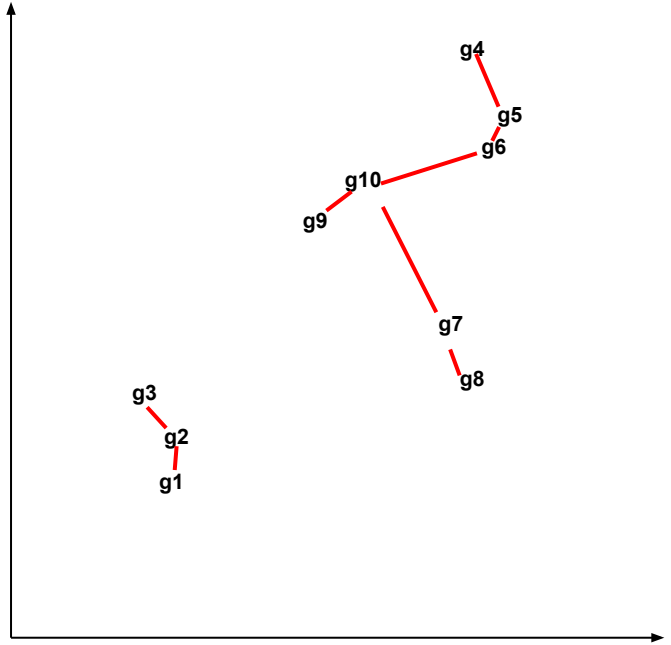
Gene Expression Analysis using Clustering





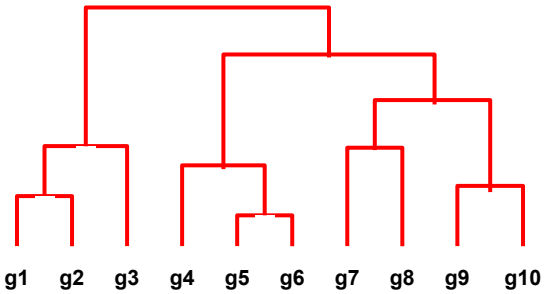
Gene Expression Analysis using Clustering

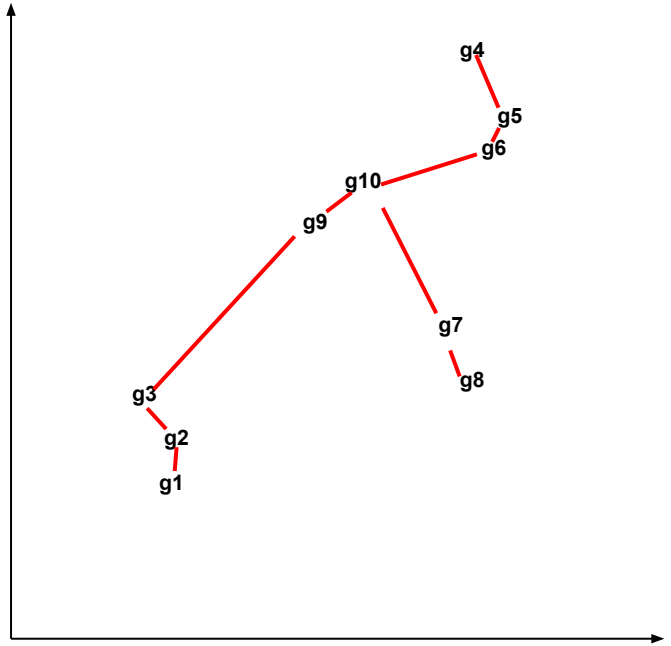




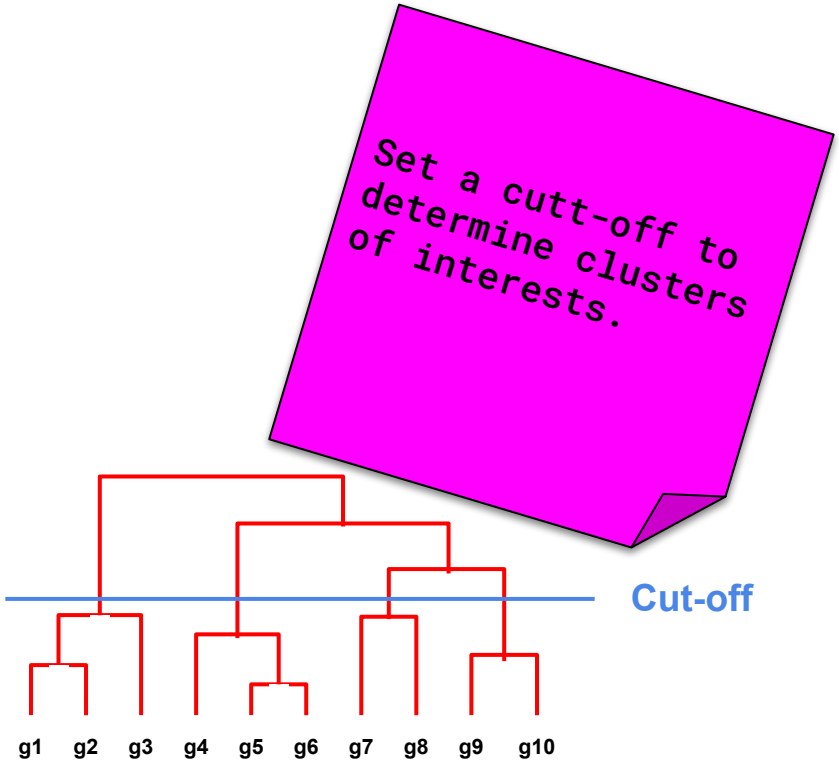
Gene Expression Analysis using Clustering

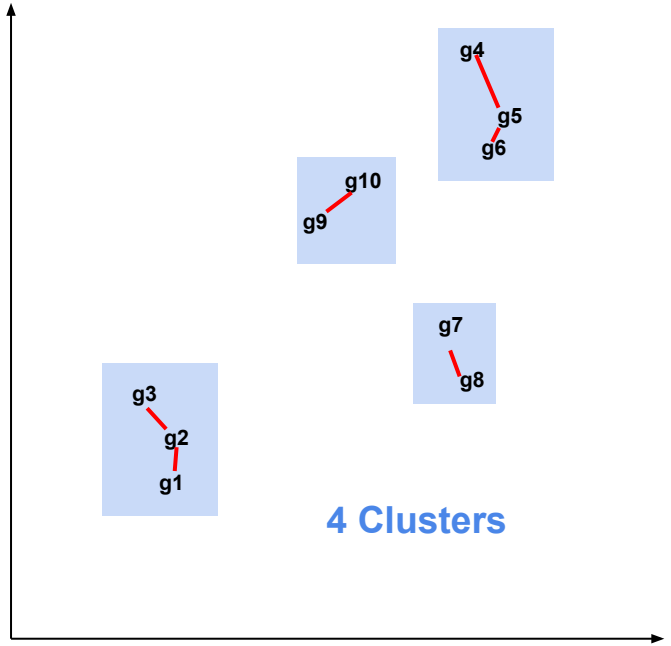
Hierarchical **C**lustering **A**nalysis
HCA (Agglomerative)





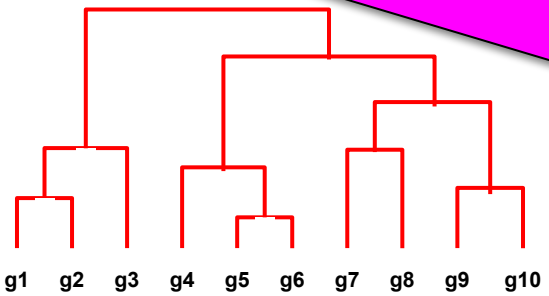
Gene Expression Analysis using Clustering

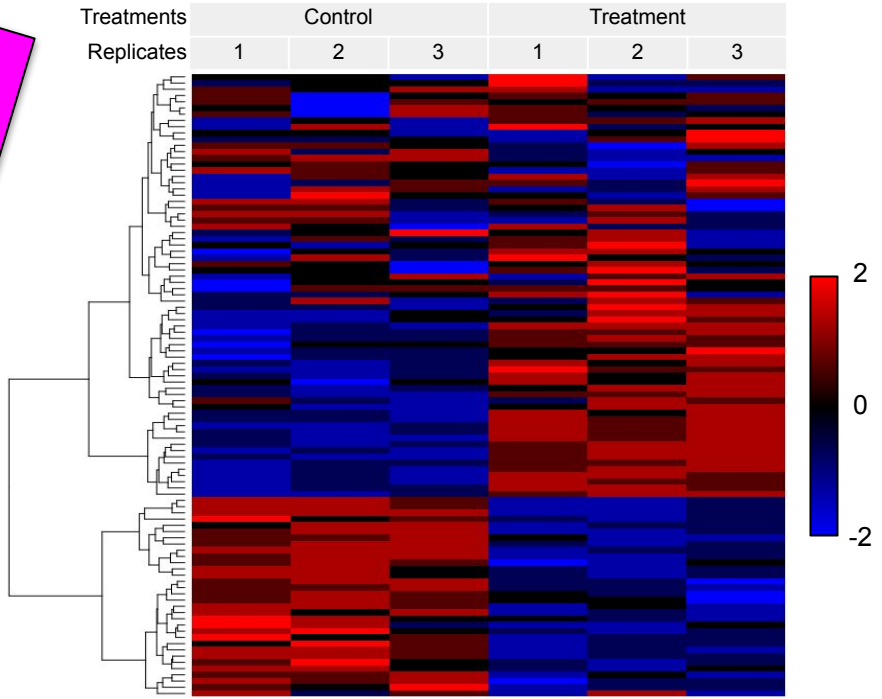




Gene Expression Analysis using Clustering

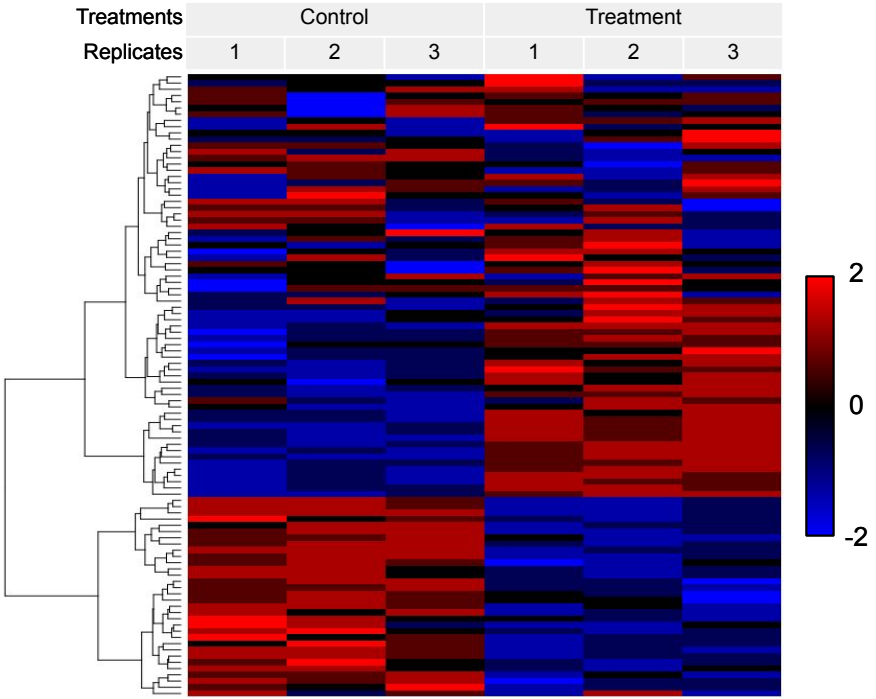
Using clustering tree to structure other data visualisation like heatmaps.





Gene Expression Analysis using Clustering

- ➔ Identification of genes that are:
  - ◆ Coexpressed (similar expression patterns)
  - ◆ Differentially expressed (divergent expression patterns)
- ➔ Provide insights about:
  - ◆ Gene function
  - ◆ Regulation



Sequence Analysis using Classification

Predict Gene Structure

```
>NC_000932.1:152806-154312 Arabidopsis thaliana chloroplast
ATGGCGATACATTTATACAAAACTTCTACCCCGAGCACACGCAATGGAGCCGTAGACAGTCAAGTGAAAT
CCAATCCACGAAATAATTTGATCTGTGGGCAGCATCATTGTGGTAAAGGTCGTAATGCCAGAGGAATAAT
TACCGCAAGGCATAGAGGGGGAGGTCATAAGCGTCTATACCGTAAAATAGATTTTCGACGAAATGCAAAA
GACATATATGGTAGAATCGTAACCATAGAATACGACCCTAATCGAAATGCATACATTTGTCTCATACACT
ATGGGGATGGTGAGAAGAGATATATTTTACATCCCAGAGGGGCTATAATTGGAGATACCATTGTTTCTGG
TACAGAAGTTCCTATAAAAATGGGAAATGCCCTACCTTTGAGTGCGGTTTGAACTATTTGATTTACGTAA
TTGGAAGTAACCAATTAGGTTTACGACGAAACCTAGAAATCGATCACTGATCCAATTTGAGTACCTCTAC
AGGATAGACCTCAACAGAAAACTGAAGAGTAACGGCAGCAAGTGATTGAGTTCAGTAGTTCCTCATATAA
AATTATTGACTCTAGAGATATAGTAATATGGAGAAGACAAAATTGTTTCAAGCACCGACAGAACCATAAG
CGCCCCTTGTTTCAAAGAGAGGAGGACGGGTTATTCACATTTCATTTGATGGTCAGAGGCGAATTGAAAG
CTAAGCAGTGGTAATTCTAAAGATTCCCCCGGGGAAAAATAGAGATGTCTCCTACGTTACCCATAATATG
TGGAAGTATCGACGTAATTTCATAGAGTCATTCGGTCTGAATGCTACATGAAGAACATAAGCCAGATGAC
GGAACGGGAAGACCTAGGATGTAGAAGATCATAACATAAGTTATTCGGCAGATTTTGATTCCTATATATC
CACTCGTGTGGTACTTCTACCATATATAGAAGAATTCTACGATATATATAAGATCCATCCGTATAGATAT
CATCATCTACATCCAGAAAGCCGTATGCTTTGGAAGAAGCTTGTACAGTTTGGGAAGGGGTTTTGATTGA
TCAAAAAGAAGAATCTACTTCAACCGATATGCCCTTAGGCACGGCCATACATAATATAGAAATCACACTT
GGAAGGGGTGGACAATTAGCTAGAGCAGCGGGTGCTGTAGCGAAACTGATTGCAAAAGAGGGGAAATCGG
CCACATTAAAATTACCTTCTGGAGAGGTCCGTTTGATATCCAAAAACTGCTCAGCAACAGTCGGACAAGT
GGGAAATGTTGGGGTAAACCAGAAAAGTTTGGGTAGAGCCGGATCGAAATGTTGGCTAGGTAAACGTCCT
GTAGTAAGAGGAGTAGTTATGAACCCTGTCGACCATCCCCATGGAGGTGGTGAAGGGAGGGCTCCAATTG
GTAGAAAAAAACCCGTAACCCCCTGGGGTTATCCTGCACTTGGAAGAAGAACTAGAAAAAGGAAAAAATA
TAGTGAGACTTTGATTCTTCGTCGCCGTAGTAAATAG
```

Predict Weather based on Activity

Day 1 🎮

🛒

🎮

🏃

🛒

🎮

🎮

🏃

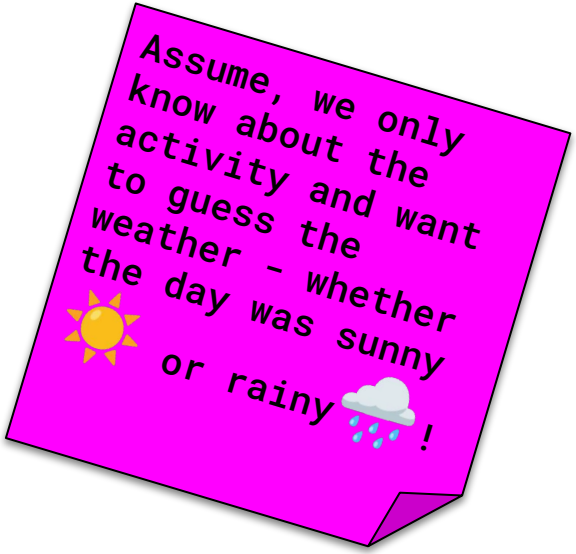
🏃

🛒

🎮

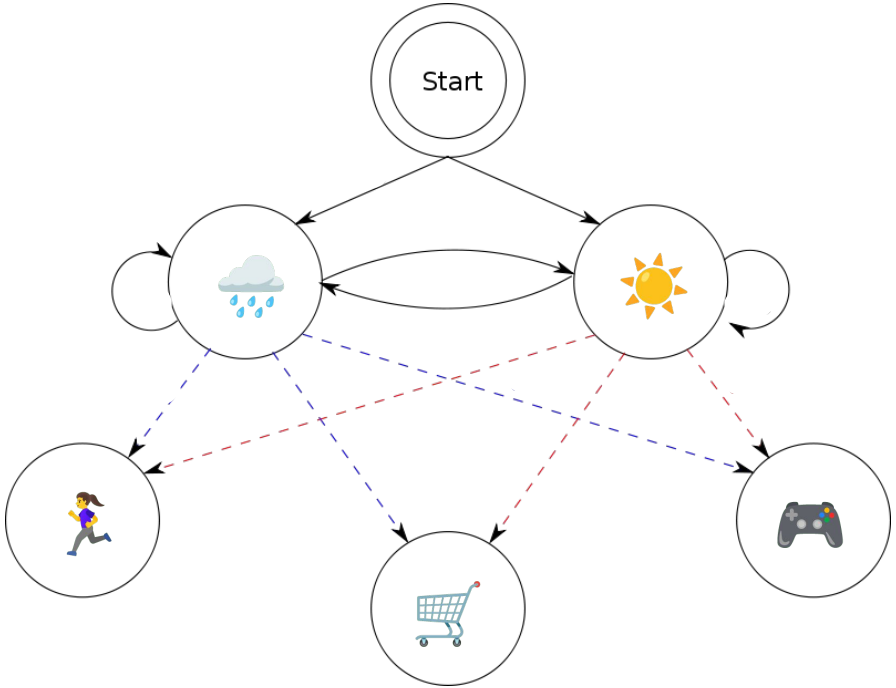
🎮

Day 14 🎮



Sequence Analysis using Classification

Hidden Markov Model (HHM)



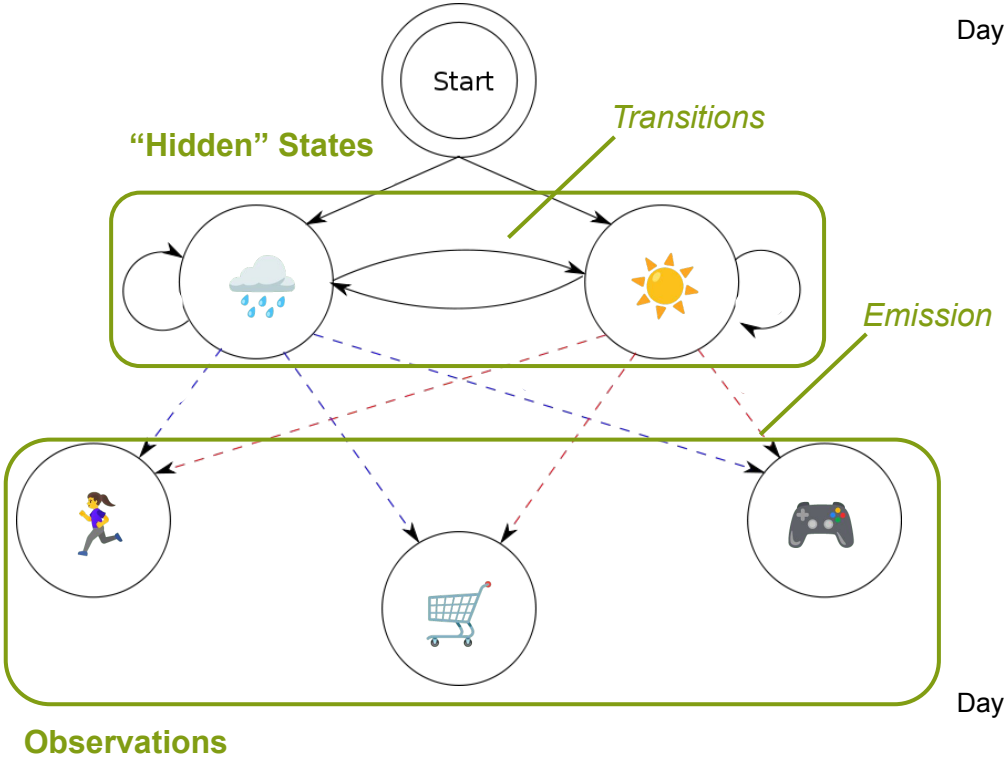
Predict Weather based on Activity

Day 1 🎮 🌧️
🛒 🌧️
🎮 🌧️
🚶 🌧️
🛒 🌧️
🎮 🌧️
🎮 🌧️
🚶 ☀️
🚶 ☀️
🚶 ☀️
🛒 ☀️
🎮 🌧️
🎮 🌧️
Day 14 🎮 🌧️

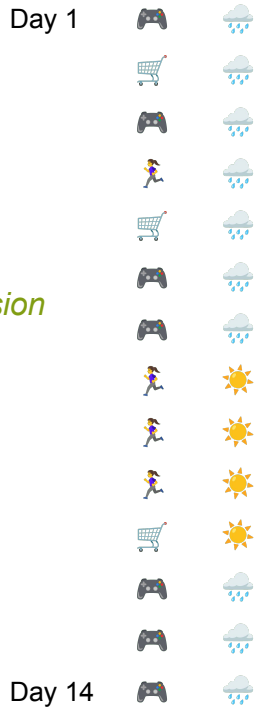
Assume, we only know about the activity and want to guess the weather - whether the day was sunny ☀️ or rainy 🌧️ !

Sequence Analysis using Classification

Hidden Markov Model (HHM)



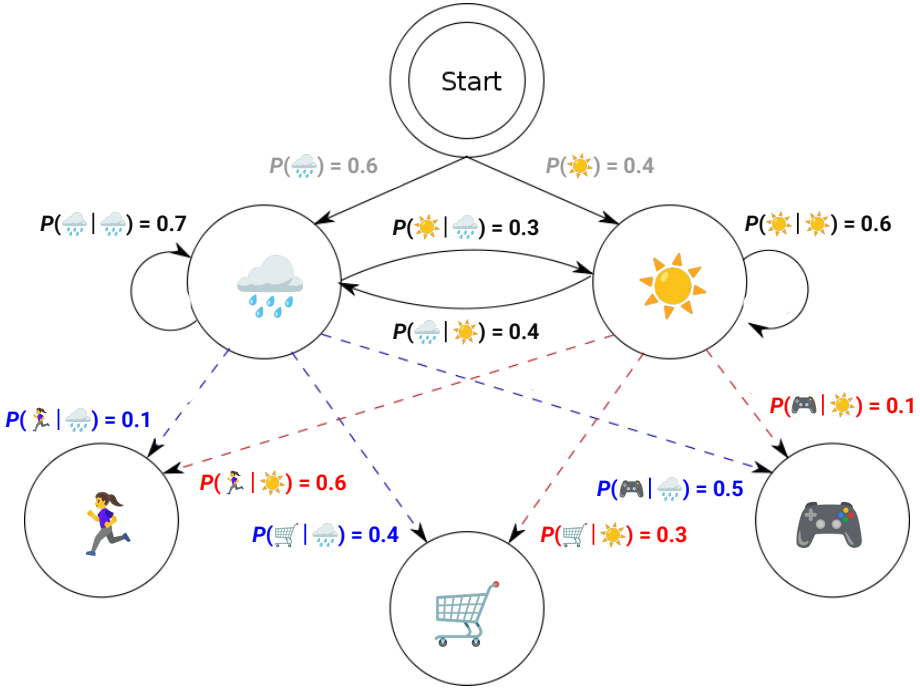
Predict Weather based on Activity



Assume, we only know about the activity and want to guess the weather - whether the day was sunny or rainy !

Sequence Analysis using Classification

Hidden Markov Model (HHM)



P(x|y) - Probability of x given y (conditional probability)
P(🌧|☀️) - Probability of being rainy when it was sunny before
P(🏃|☀️) - Probability of running if it's sunny

Predict Weather based on Activity

Day 1 🎮 🌧

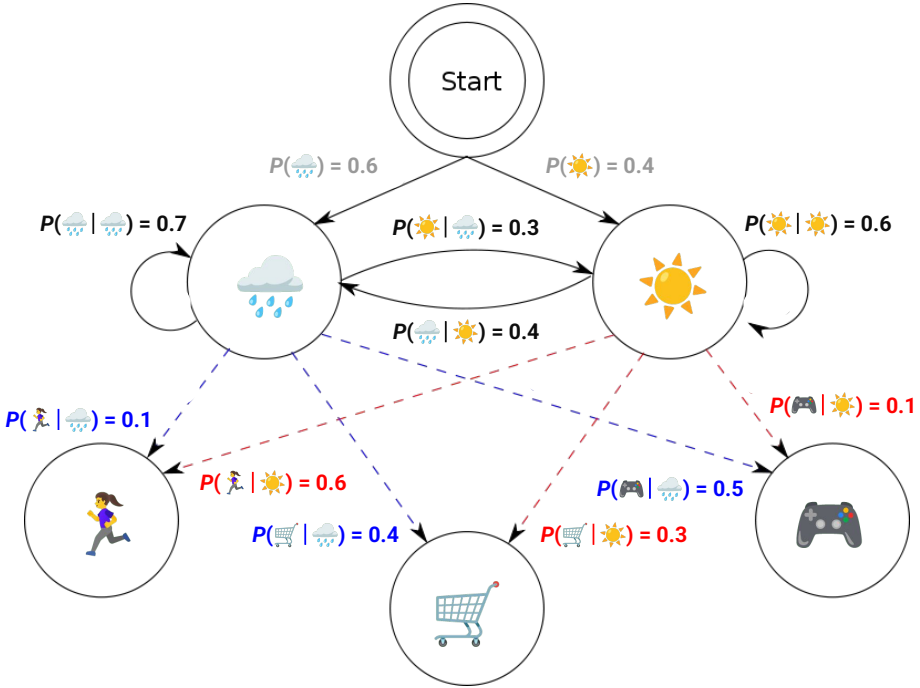
$P(🌧 | 🎮) = P(🌧) \times P(🎮 | 🌧) = 0.6 \times 0.5 = 0.30$

🛒 🌧
🎮 🌧
🏃 🌧
🛒 🌧
🎮 🌧
🎮 🌧
🏃 ☀️
🏃 ☀️
🏃 ☀️
🛒 ☀️
🎮 🌧
🎮 🌧

Day 14 🎮 🌧

Sequence Analysis using Classification

Hidden Markov Model (HHM)



P(x|y) - Probability of x given y (conditional probability)
P(🌧|☀️) - Probability of being rainy when it was sunny before
P(🏃|☀️) - Probability of running if it's sunny

Predict Weather based on Activity

Day 1 🎮 🌧

$P(🌧 | 🎮) = P(🌧) \times P(🎮 | 🌧) = 0.6 \times 0.5 = 0.30$

🛒 🌧

$P(🌧 | 🛒) = P(🌧 | 🌧) \times P(🛒 | 🌧) = 0.7 \times 0.4 = 0.28$

🎮 🌧

🏃 🌧

🛒 🌧

🎮 🌧

🎮 🌧

🏃 ☀️

🏃 ☀️

🏃 ☀️

🛒 ☀️

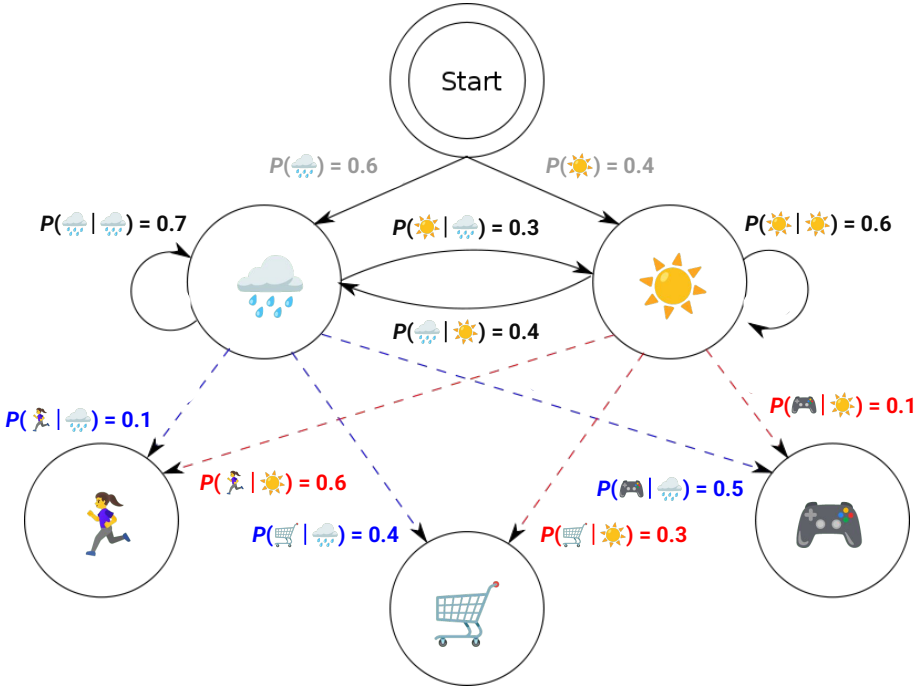
🎮 🌧

🎮 🌧

Day 14 🎮 🌧

Sequence Analysis using Classification

Hidden Markov Model (HHM)



P(x | y) - Probability of x given y (conditional probability)
P(🌧 | ☀) - Probability of being rainy when it was sunny before
P(🏃 | ☀) - Probability of running if it's sunny

Predict Weather based on Activity

Day 1 🎮 🌧

$P(🌧 \mid 🎮) = P(🌧) \times P(🎮 \mid 🌧) = 0.6 \times 0.5 = 0.30$

🛒 🌧

$P(🌧 \mid 🛒) = P(🌧 \mid 🌧) \times P(🛒 \mid 🌧) = 0.7 \times 0.4 = 0.28$

🎮 🌧

$P(🌧 \mid 🎮) = P(🌧 \mid 🌧) \times P(🎮 \mid 🌧) = 0.7 \times 0.5 = 0.35$

🏃 🌧

🛒 🌧

🎮 🌧

🎮 🌧

🏃 ☀

🏃 ☀

🏃 ☀

🛒 ☀

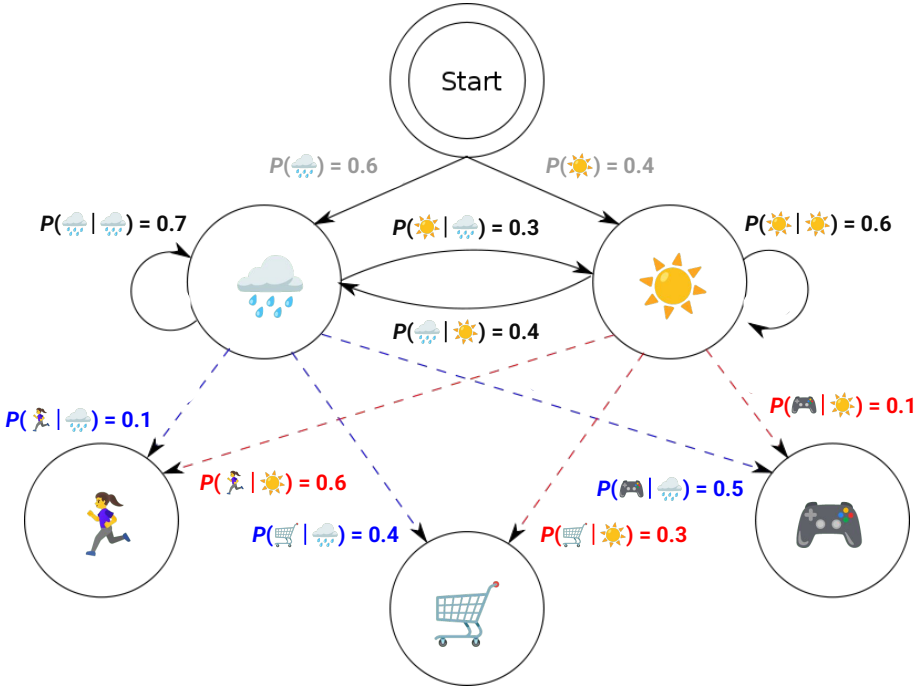
🎮 🌧

🎮 🌧

Day 14 🎮 🌧

Sequence Analysis using Classification

Hidden Markov Model (HHM)



P(x | y) - Probability of x given y (conditional probability)
P(🌧 | ☀) - Probability of being rainy when it was sunny before
P(🏃 | ☀) - Probability of running if it's sunny

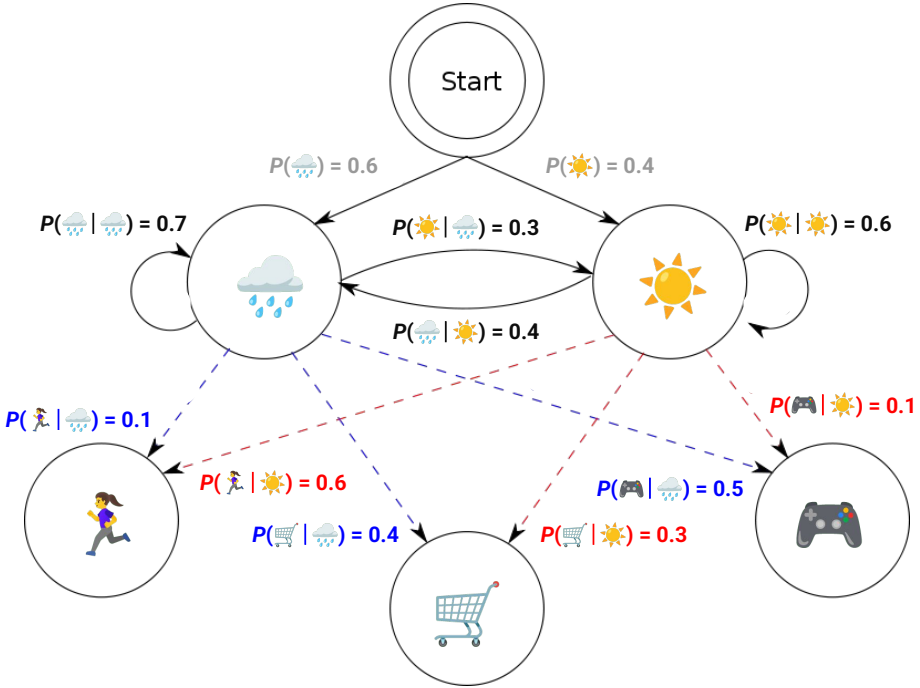
Predict Weather based on Activity

Day 1	🎮	🌧	$P(🌧 \mid 🎮) = P(🌧) \times P(🎮 \mid 🌧) = 0.6 \times 0.5 = 0.30$
	🛒	🌧	$P(🌧 \mid 🛒) = P(🌧 \mid 🌧) \times P(🛒 \mid 🌧) = 0.7 \times 0.4 = 0.28$
	🎮	🌧	$P(🌧 \mid 🎮) = P(🌧 \mid 🌧) \times P(🎮 \mid 🌧) = 0.7 \times 0.5 = 0.35$
	🏃	🌧	$P(🌧 \mid 🏃) = P(🌧 \mid 🌧) \times P(🏃 \mid 🌧) = 0.7 \times 0.1 = 0.07$
	🛒	🌧	$P(🌧 \mid 🛒) = P(🌧 \mid 🌧) \times P(🛒 \mid 🌧) = 0.7 \times 0.4 = 0.28$
	🎮	🌧	$P(🌧 \mid 🎮) = P(🌧 \mid 🌧) \times P(🎮 \mid 🌧) = 0.7 \times 0.5 = 0.35$
	🎮	🌧	$P(🌧 \mid 🎮) = P(🌧 \mid 🌧) \times P(🎮 \mid 🌧) = 0.7 \times 0.5 = 0.35$
	🏃	☀	$P(☀ \mid 🏃) = P(☀ \mid 🌧) \times P(🏃 \mid ☀) = 0.3 \times 0.6 = 0.18$
	🏃	☀	$P(☀ \mid 🏃) = P(☀ \mid ☀) \times P(🏃 \mid ☀) = 0.6 \times 0.6 = 0.36$
	🏃	☀	$P(☀ \mid 🏃) = P(☀ \mid ☀) \times P(🏃 \mid ☀) = 0.6 \times 0.6 = 0.36$
	🛒	☀	$P(☀ \mid 🛒) = P(☀ \mid ☀) \times P(🛒 \mid ☀) = 0.6 \times 0.3 = 0.36$
	🎮	🌧	$P(🌧 \mid 🎮) = P(🌧☀ \mid ☀) \times P(🎮 \mid 🌧) = 0.4 \times 0.5 = 0.2$
	🎮	🌧	$P(🌧 \mid 🎮) = P(🌧 \mid 🌧) \times P(🎮 \mid 🌧) = 0.7 \times 0.5 = 0.35$
Day 14	🎮	🌧	$P(🌧 \mid 🎮) = P(🌧 \mid 🌧) \times P(🎮 \mid 🌧) = 0.7 \times 0.5 = 0.35$

Multi all -> Path Probability = **7.26 × 10^(-9)**

Sequence Analysis using Classification

Hidden Markov Model (HHM)



Predict Weather based on Activity

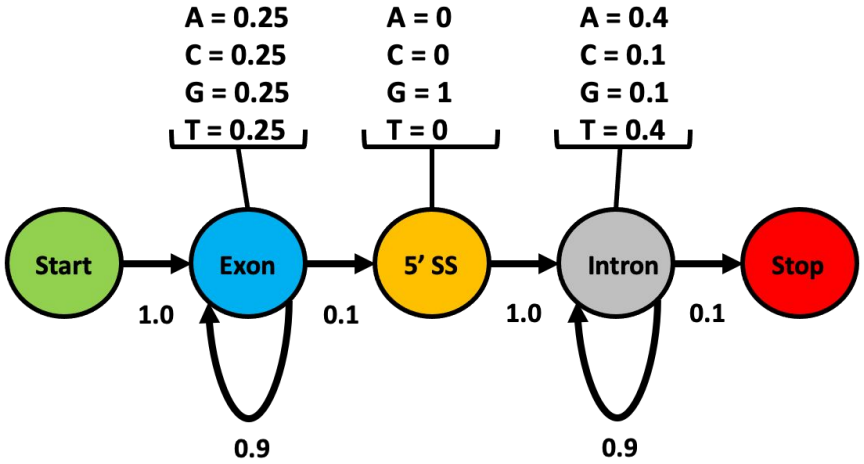
Day 1	🎮	🌧	🌧	🌧	🌧	🌧
	🛒	🌧	🌧	🌧	☀	🌧
	🎮	🌧	🌧	🌧	☀	🌧
	🏃	🌧	☀	🌧	🌧	☀
	🛒	🌧	🌧	🌧	🌧	☀
	🎮	🌧	🌧	🌧	🌧	☀
	🎮	🌧	🌧	🌧	🌧	🌧
	🏃	☀	☀	☀	☀	🌧
	🏃	☀	☀	☀	☀	🌧
	🏃	☀	☀	☀	☀	🌧
	🛒	☀	🌧	🌧	☀	☀
	🎮	🌧	☀	🌧	🌧	🌧
	🎮	🌧	🌧	🌧	🌧	🌧
Day 14	🎮	🌧	🌧	🌧	🌧	🌧
		7.26 × 10^(-9)	8.13 × 10^(-10)	**1.13× 10^(-8)**	2.29 × 10^(-10)	8.65× 10^(-12)

Sequence Analysis using Classification

Predict Gene Structure

```
>NC_000932.1:152806-154312 Arabidopsis thaliana chloroplast
ATGGCGATACATTTATACAAAACTTCTACCCCGAGCACACGCAATGGAGCCGTAGACAGTCAAGTGAAAT
CCAATCCACGAAATAATTTGATCTGTGGGCAGCATCATTGTGGTAAAGGTCGTAATGCCAGAGGAATAAT
TACCGCAAGGCATAGAGGGGGAGGTCATAAGCGTCTATACCGTAAAATAGATTTTCGACGAAATGCAAAA
GACATATATGGTAGAATCGTAACCATAGAATACGACCCTAATCGAAATGCATACATTTGTCTCATACACT
ATGGGGATGGTGAGAAGAGATATATTTTACATCCCAGAGGGGCTATAATTGGAGATACCATTGTTTCTGG
TACAGAAGTTCCTATAAAAATGGGAAATGCCCTACCTTTGAGTGCGGTTTGAACTATTTGATTTACGTAA
TTGGAAGTAACCAATTAGGTTTACGACGAAACCTAGAAATCGATCACTGATCCAATTTGAGTACCTCTAC
AGGATAGACCTCAACAGAAAACTGAAGAGTAACGGCAGCAAGTGATTGAGTTCAGTAGTTCCTCATATAA
AATTATTGACTCTAGAGATATAGTAATATGGAGAAGACAAAATTGTTTCAAGCACCGACAGAACCATAAG
CGCCCCTTGTTTCAAAGAGAGGAGGACGGGTTATTCACATTTCATTTGATGGTCAGAGGCGAATTGAAAG
CTAAGCAGTGGTAATTCTAAAGATTCCCCCGGGGAAAAATAGAGATGTCTCCTACGTTACCCATAATATG
TGGAAGTATCGACGTAATTTCATAGAGTCATTCGGTCTGAATGCTACATGAAGAACATAAGCCAGATGAC
GGAACGGGAAGACCTAGGATGTAGAAGATCATAACATAAGTTATTCGGCAGATTTTGATTCCTATATATC
CACTCGTGTGGTACTTCTACCATATATAGAAGAATTCTACGATATATATAAGATCCATCCGTATAGATAT
CATCATCTACATCCAGAAAGCCGTATGCTTTGGAAGAAGCTTGTACAGTTTGGGAAGGGGTTTTGATTGA
TCAAAAAGAAGAATCTACTTCAACCGATATGCCCTTAGGCACGGCCATACATAATATAGAAATCACACTT
GGAAGGGGTGGACAATTAGCTAGAGCAGCGGGTGCTGTAGCGAAACTGATTGCAAAAGAGGGGAAATCGG
CCACATTAAAATTACCTTCTGGAGAGGTCCGTTTGATATCCAAAAACTGCTCAGCAACAGTCGGACAAGT
GGGAAATGTTGGGGTAAACCAGAAAAGTTTGGGTAGAGCCGGATCGAAATGTTGGCTAGGTAAACGTCCT
GTAGTAAGAGGAGTAGTTATGAACCCTGTCGACCATCCCCATGGAGGTGGTGAAGGGAGGGCTCCAATTG
GTAGAAAAAAACCCGTAACCCCCTGGGGTTATCCTGCACTTGGAAGAAGAACTAGAAAAAGGAAAAAATA
TAGTGAGACTTTGATTCTTCGTCGCCGTAGTAAATAG
```

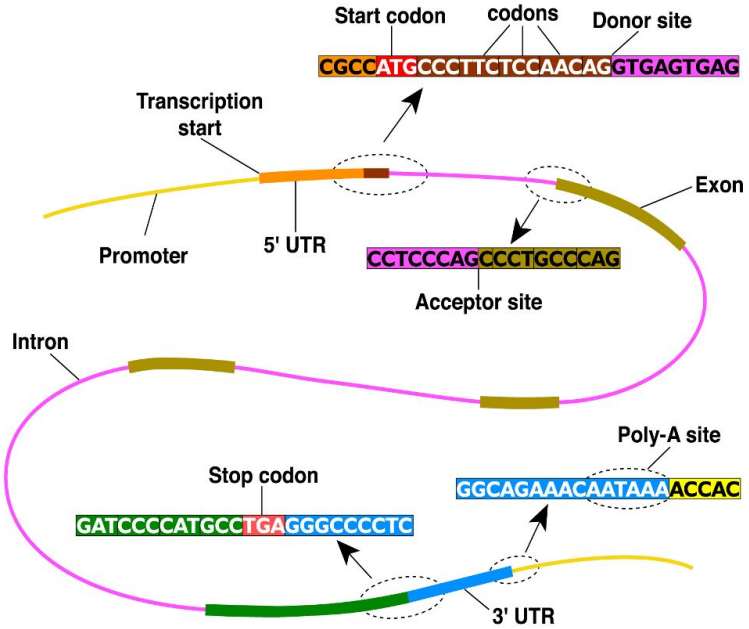
Hidden Markov Model (HHM)



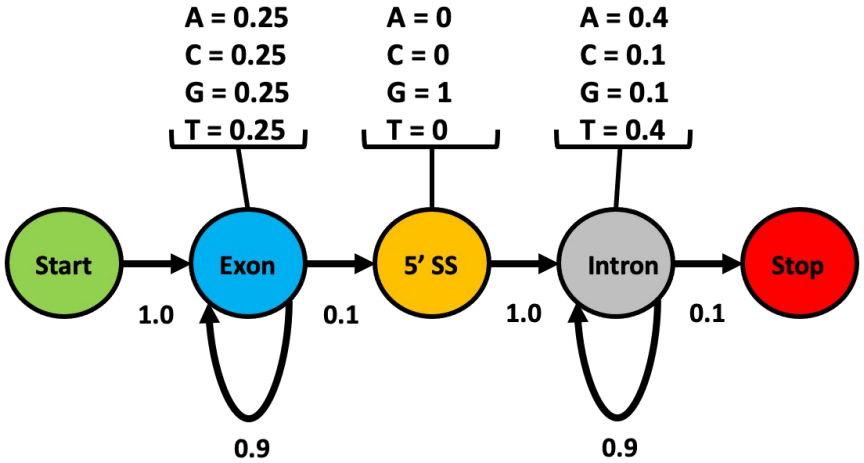
[Weisstein et al. A Hands-on Introduction to Hidden Markov Models]

Sequence Analysis using Classification

Predict Gene Structure



Hidden Markov Model (HHM)



[Weisstein et al. A Hands-on Introduction to Hidden Markov Models]

Sequence Analysis using Classification

Predict Gene Structure

>NC_000932.1:152806-154312 Arabidopsis thaliana chloroplast

ATGGCGATACATTTATACAAAACTTCTACCCCGAGCACACGCAATGGAGCCGTAGACAGTCAAGTGAAAT
CCAATCCACGAAATAATTTGATCTGTGGGCAGCATCATTGTGGTAAAGGTCGTAATGCCAGAGGAATAAT
TACCGCAAGGCATAGAGGGGGAGGTCATAAGCGTCTATACCGTAAAATAGATTTTCGACGAAATGCAAAA
GACATATATGGTAGAATCGTAACCATAGAATACGACCCTAATCGAAATGCATACATTTGTCTCATACACT
ATGGGGATGGTGAGAAGAGATATATTTTACATCCCAGAGGGGCTATAATTGGAGATACCATTGTTTCTGG
TACAGAAGTTCCTATAAAAATGGGAAATGCCCTACCTTTGAGTGCGGTTTGAACTATTTGATTTACGTAA
TTGGAAGTAACCAATTAGGTTTACGACGAAACCTAGAAATCGATCACTGATCCAATTTGAGTACCTCTAC
AGGATAGACCTCAACAGAAAACTGAAGAGTAACGGCAGCAAGTGATTGAGTTCAGTAGTTCCTCATATAA
AATTATTGACTCTAGAGATATAGTAATATGGAGAAGACAAAATTGTTTCAAGCACCGACAGAACCATAAG
CGCCCCTTGTTTCAAAGAGAGGAGGACGGGTTATTCACATTTCATTTGATGGTCAGAGGCGAATTGAAAG
CTAAGCAGTGGTAATTCTAAAGATTCCCCCGGGGAAAAATAGAGATGTCTCCTACGTTACCCATAATATG
TGGAAGTATCGACGTAATTTCATAGAGTCATTCGGTCTGAATGCTACATGAAGAACATAAGCCAGATGAC
GGAACGGGAAGACCTAGGATGTAGAAGATCATAACATAAGTTATTCGGCAGATTTTGATTCCTATATATC
ACCTCGTGTGGTACTTCTACCATAATAGAAGAATTCTACGATATATATAAGATCCATCCGTATAGATAT
CATCATCTACATCCAGAAAGCCGTATGCTTTGGAAGAAGCTTGTACAGTTTGGGAAGGGGTTTTGATTGA
TCAAAAAGAAGAATCTACTTCAACCGATATGCCCTTAGGCACGGCCATACATAATATAGAAATCACACTT
GGAAGGGGTGGACAATTAGCTAGAGCAGCGGGTGCTGTAGCGAAACTGATTGCAAAAGAGGGGAAATCGG
CCACATTAAAATTACCTTCTGGAGAGGTCCGTTTGATATCCAAAAACTGCTCAGCAACAGTCGGACAAGT
GGGAAATGTTGGGGTAAACCAGAAAAGTTTGGGTAGAGCCGGATCGAAATGTTGGCTAGGTAAACGTCCT
GTAGTAAGAGGAGTAGTTATGAACCCTGTCGACCATCCCCATGGAGGTGGTGAAGGGAGGGCTCCAATTG
GTAGAAAAAAACCCGTAACCCCCTGGGGTTATCCTGCACTTGGAAGAAGAACTAGAAAAAGGAAAAAATA
TAGTGAGACTTTGATTCTTCGTCGCCGTAGTAAATAG

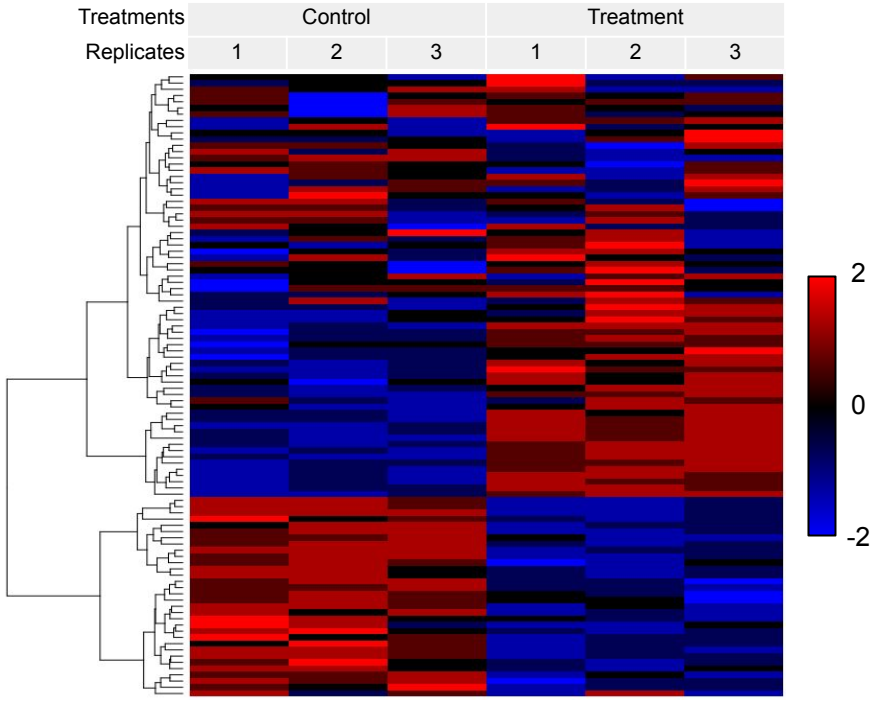
GENE
CDS (Coding Sequence)
ATG (Start Codon of CDS)
TAG (Stop Codon of CDS)
Intron

Hidden Markov Model - AUGUSTUS

```
# This output was generated with AUGUSTUS (version 3.3.3).
# AUGUSTUS is a gene prediction tool written by M. Stanke (mario.stanke@uni-greifswald.de),
# O. Keller, S. König, L. Gerischer, L. Romoth and Katharina Hoff.
# Please cite: Mario Stanke, Mark Diekhans, Robert Baertsch, David Haussler (2008),
# Using native and syntenically mapped cDNA alignments to improve de novo gene finding
# Bioinformatics 24: 637-644, doi 10.1093/bioinformatics/btn013
# No extrinsic information on sequences given.
# arabidopsis version. Using default transition matrix.
# We have hints for 0 sequences and for 0 of the sequences in the input set.
#
# ----- prediction on sequence number 1 (length = 1507, name = NC_000932.1:152806-154312) -----
#
# Predicted genes for sequence number 1 on forward strand
#start gene g1
NC_000932.1:152806-154312          AUGUSTUS    gene         1          575
NC_000932.1:152806-154312          AUGUSTUS    transcript   1          575
NC_000932.1:152806-154312          AUGUSTUS    start_codon  1          3
NC_000932.1:152806-154312          AUGUSTUS    single       1          402
NC_000932.1:152806-154312          AUGUSTUS    CDS          1          402
NC_000932.1:152806-154312          AUGUSTUS    exon         1          575
NC_000932.1:152806-154312          AUGUSTUS    stop_codon   400        402
NC_000932.1:152806-154312          AUGUSTUS    tts          575        575
# coding sequence = [atggcgatacatttatacaaaacttctaccccgagcacacgcaatggagccgtagacagtcaagtgaaatccaatccac
# gaaataatttgatctgtgggcagcatcattgtggtaaaggtcgtaatgccagaggaataatttaccgcaaggcatagaggggggaggtcataagcgtcta
# taccgtaaaatagattttcgacgaaatgcaaaagacatatatggtagaatcgtaaccatagaatacgaccctaatcgaaatgcatacatttgtctcat
# acactatgggatggtgagaagagatatattttacatcccagaggggctataattggagataccattgtttctggtacagaagttcctataaaaatgg
# gaaatgccctacctttgagtgcggtttga]
# protein sequence = [MAIHLYKTSTPSTRNGAVDSQVKSNPRNNLICGQHHCGKGRNARGIITARHRGGGHKRLYRKIDFRRNAKDIYGRIVT
# IEYDPNRNAYICLIHYGDGEKRYILHPRGAIIGDTIVSGTEVPIKMGNALPLSAV]
# end gene g1
###
# start gene g2
NC_000932.1:152806-154312          AUGUSTUS    gene         936        1507
NC_000932.1:152806-154312          AUGUSTUS    transcript   936        1507
NC_000932.1:152806-154312          AUGUSTUS    tss          936        936
NC_000932.1:152806-154312          AUGUSTUS    exon         936        1507
NC_000932.1:152806-154312          AUGUSTUS    start_codon  1079       1081
NC_000932.1:152806-154312          AUGUSTUS    single       1079       1507
NC_000932.1:152806-154312          AUGUSTUS    CDS          1079       1507
NC_000932.1:152806-154312          AUGUSTUS    stop_codon   1505       1507
# coding sequence = [atgcccttaggcacggccatacataatatagaaatcacacttggaaggggtggacaattagctagagcagcgggtgctg
# tagcgaaactgattgcaaaagaggggaaatcggccacattaaaattaccttctggagaggtccgtttgatatccaaaaactgctcagcaacagtcggga
# caagtgggaaatgttggggtaaaccagaaaagtttgggtagagccggatcgaaatgttggctaggtaaacgtcctgtagtaagaggagtagttatgaa
# ccctgtcgaccatccccatggaggtggtgaaggggagggctccaattggtagaaaaaaacccgtaacccccctggggttatcctgcacttggaagaagaa
# ctagaaaaaggaaaaaatatagtgagactttgattcttcgtcgccgtagtaaatag]
# protein sequence = [MPLGTAIHNIEITLGRGGQLARAAGAVAKLIAKEGKSATLKLPSGEVRLISKNCSATVGQVGNVGVNQKSLGRAGSKC
# WLGKRPVVRGVVMNPVDHPHGGGEGRAPIGRKKPVTPWGYPALGRRTRKRKKYSETLILRRRSK]
# end gene g2
```

Gene Expression Analysis using Dimensionality Reduction

Treatments	Control			Treatment		
Replicates	1	2	3	1	2	3
g1	345	234	432	311	294	312
g2	2	23	11	12	45	3
g3	32	36	30	46	51	43
g4	79	85	67	54	67	66
g5	1	0	0	4	3	2
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
g100	24	22	17	17		18



Gene Expression Analysis using Dimensionality Reduction

Treatments	Control			Treatment		
Replicates	1	2	3	1	2	3
g1	345	234	432	311	294	312
g2	2	23	11	12	45	3
g3	32	36	30	46	51	43
g4	79	85	67	54	67	66
g5	1	0	0	4	3	2
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
…	…	…	…	…	…	…
g100	24	22	17	17		18

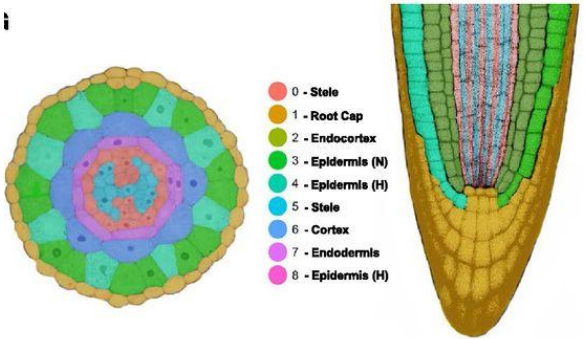
- ➔ Not, only one treatment, but
  - ◆ multiple genotypes, lines, origins
  - ◆ multiple treatments
  - ◆ multiple time points
  - ◆ …
- ➔ How to determine, which samples cluster together?
- ➔ How to explain the variance within the samples?
  …Reduce Dimensionality using Principal Component Analysis (PCA)

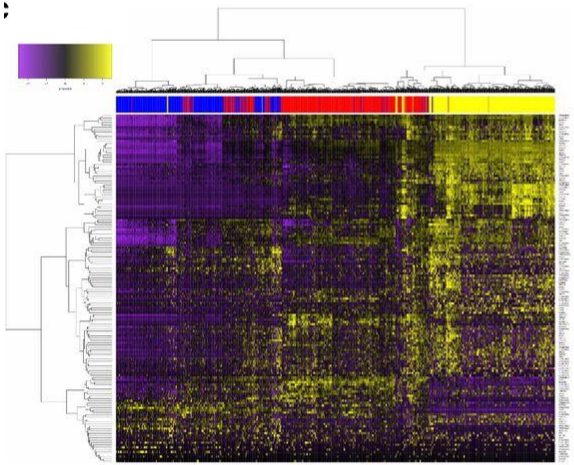
Gene Expression Analysis using Dimensionality Reduction

Single Cell RNA sequencing
on Arabidopsis thaliana roots

Cluster Analysis
Hierarchical Cluster Analysis (HCA)

Dimensionality Reduction
T-distributed Stochastic Neighbour Embedding (t-SNE)





How to know which cells behave similarly?

Ryu, Kook Hui, Ling Huang, Hyun Min Kang, and John Schiefelbein. 2019. “Single-Cell RNA Sequencing Resolves Molecular Relationships among Individual Plant Cells.” Plant Physiology 179 (4): 1444–56. https://doi.org/10.1104/pp.18.01482.

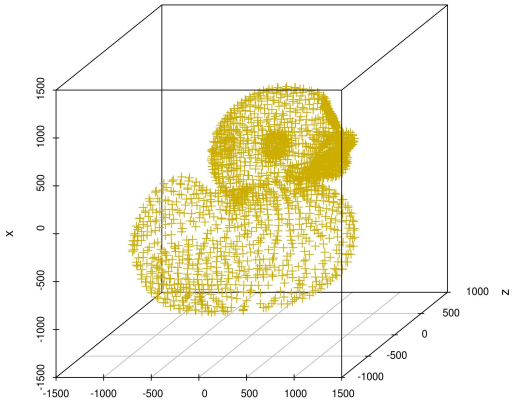
Principal Component Analysis

Gene Expression Data

Cluster Analysis
Hierarchical Cluster Analysis (HCA)

Dimensionality Reduction
(Principal Component Analysis)

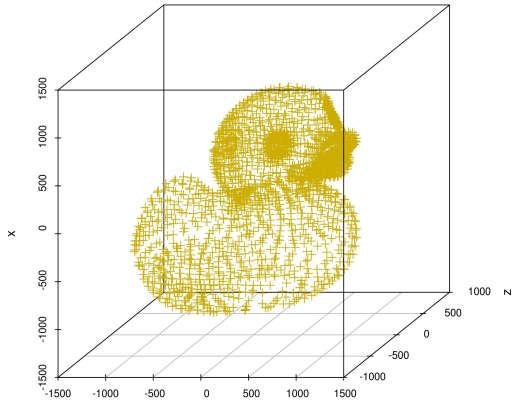
	X	Y	Z
g1	345	234	312
g2	2	23	3
g3	32	36	43
g4	79	85	66
g5	1	0	2
…	…	…	…
g1000	24	22	18



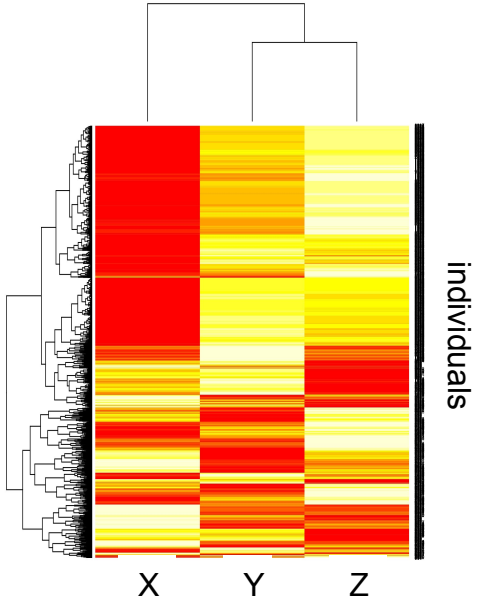
Principal Component Analysis

Gene Expression Data

	X	Y	Z
g1	345	234	312
g2	2	23	3
g3	32	36	43
g4	79	85	66
g5	1	0	2
…	…	…	…
g1000	24	22	18



Cluster Analysis
Hierarchical Cluster Analysis (HCA)

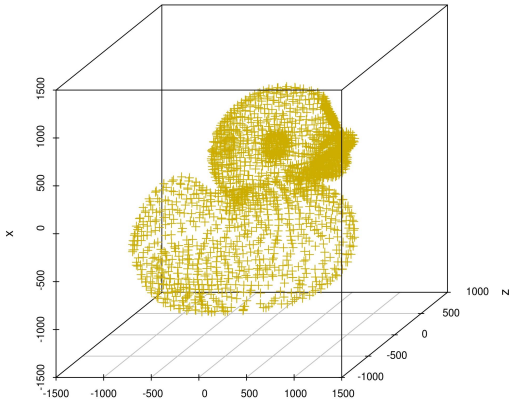


Dimensionality Reduction
(Principal Component Analysis)

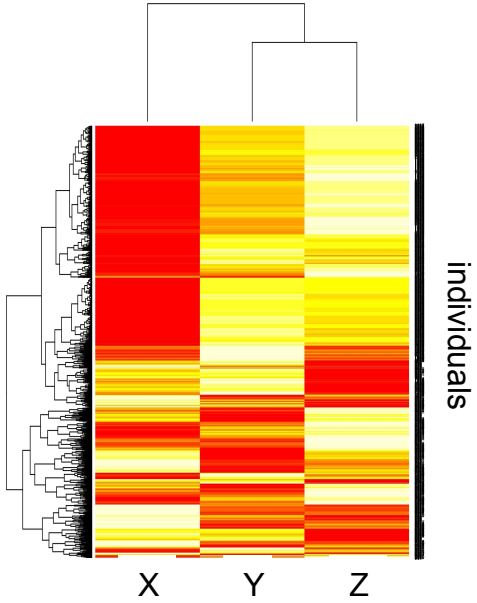
Principal Component Analysis

Gene Expression Data

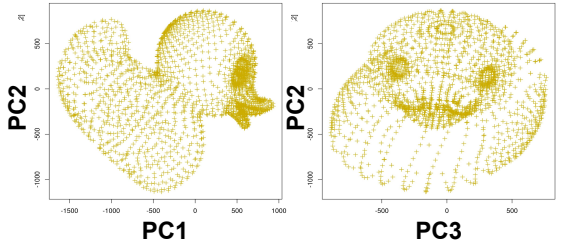
	X	Y	Z
g1	345	234	312
g2	2	23	3
g3	32	36	43
g4	79	85	66
g5	1	0	2
…	…	…	…
g1000	24	22	18



Cluster Analysis
Hierarchical Cluster Analysis (HCA)



Dimensionality Reduction
(Principal Component Analysis)



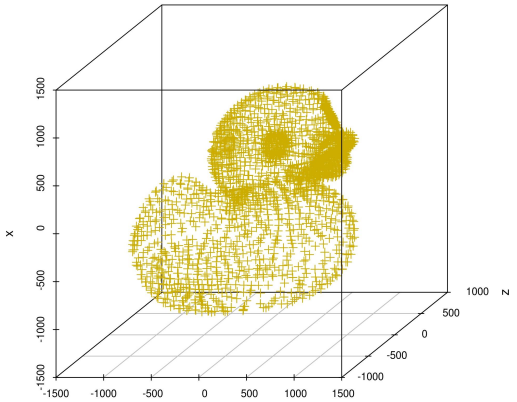
	PC1	PC2	PC3
X	**0.49**	**0.40**	0.11
Y	**0.49**	**0.50**	0.01
z	0.03	0.10	**0.87**

Loadings – how much each PC contributes to the sample

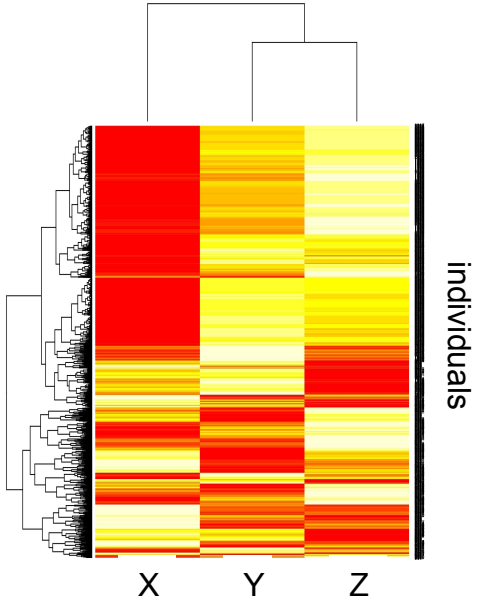
Principal Component Analysis

Gene Expression Data

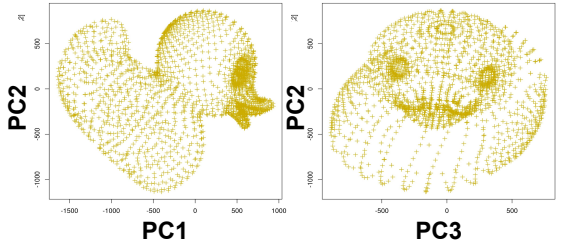
	X	Y	Z
g1	345	234	312
g2	2	23	3
g3	32	36	43
g4	79	85	66
g5	1	0	2
…	…	…	…
g1000	24	22	18



Cluster Analysis
Hierarchical Cluster Analysis (HCA)



Dimensionality Reduction
(Principal Component Analysis)



	PC1	PC2	PC3
X	**0.49**	**0.40**	0.11
Y	**0.49**	**0.50**	0.01
z	0.03	0.10	**0.87**

Loadings – how much each PC contributes to the sample

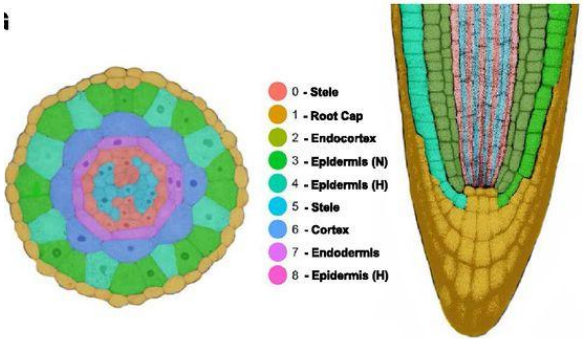
PC1 and PC2 together explain a significant portion of the variance in X and Y. PC3 is crucial for explaining the variance in Z. To reduce the data dimensionality, one could consider dropping PC3.

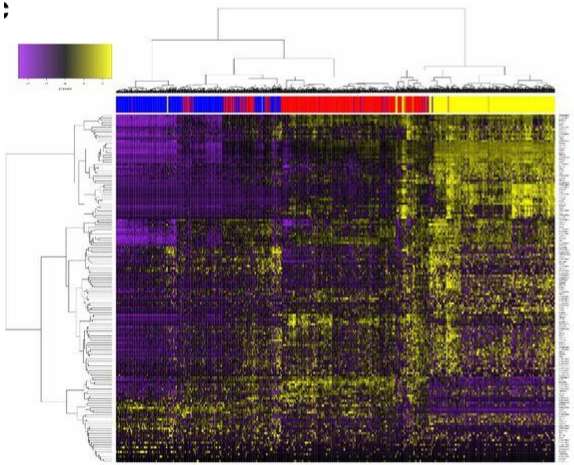
Gene Expression Analysis using Dimensionality Reduction

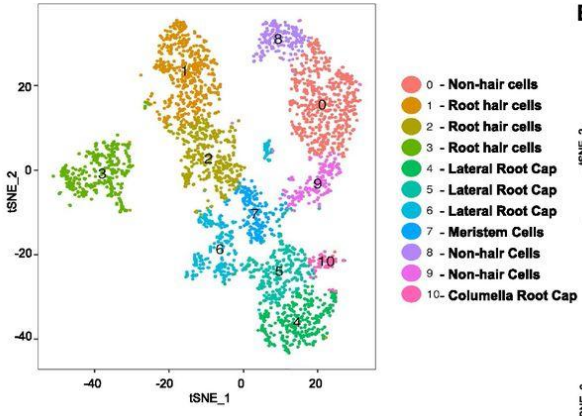
Single Cell RNA sequencing
on Arabidopsis thaliana roots

Cluster Analysis
Hierarchical Cluster Analysis (HCA)

Dimensionality Reduction
T-distributed Stochastic Neighbour Embedding (t-SNE)







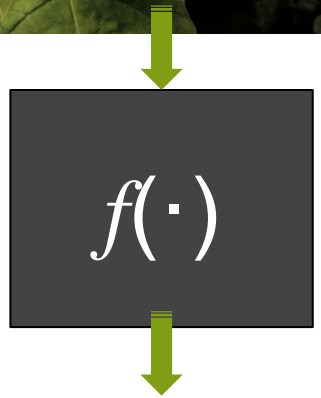
Ryu, Kook Hui, Ling Huang, Hyun Min Kang, and John Schiefelbein. 2019. “Single-Cell RNA Sequencing Resolves Molecular Relationships among Individual Plant Cells.” Plant Physiology 179 (4): 1444–56. https://doi.org/10.1104/pp.18.01482.

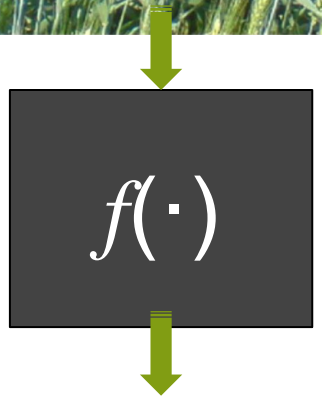
What about less obvious questions?

$f(\cdot)$

Infection with *Botrytis cinerea (gray mold)*?

What about less obvious questions?





Infection with *Botrytis cinerea (gray mold)*?

How many spikes?

What about less obvious questions?

$f(\cdot)$ $f(\cdot)$ $f(\cdot)$

Infection with *Botrytis cinerea (gray mold)*?

How many spikes?

Is the gene expressed under drought?

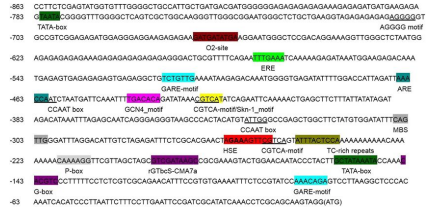
What about less obvious questions?

$f(\cdot)$

Infection with *Botrytis cinerea (gray mold)*?

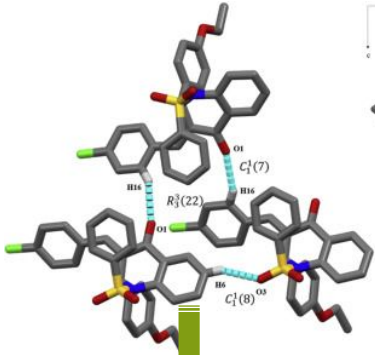
$f(\cdot)$

How many spikes?



$f(\cdot)$

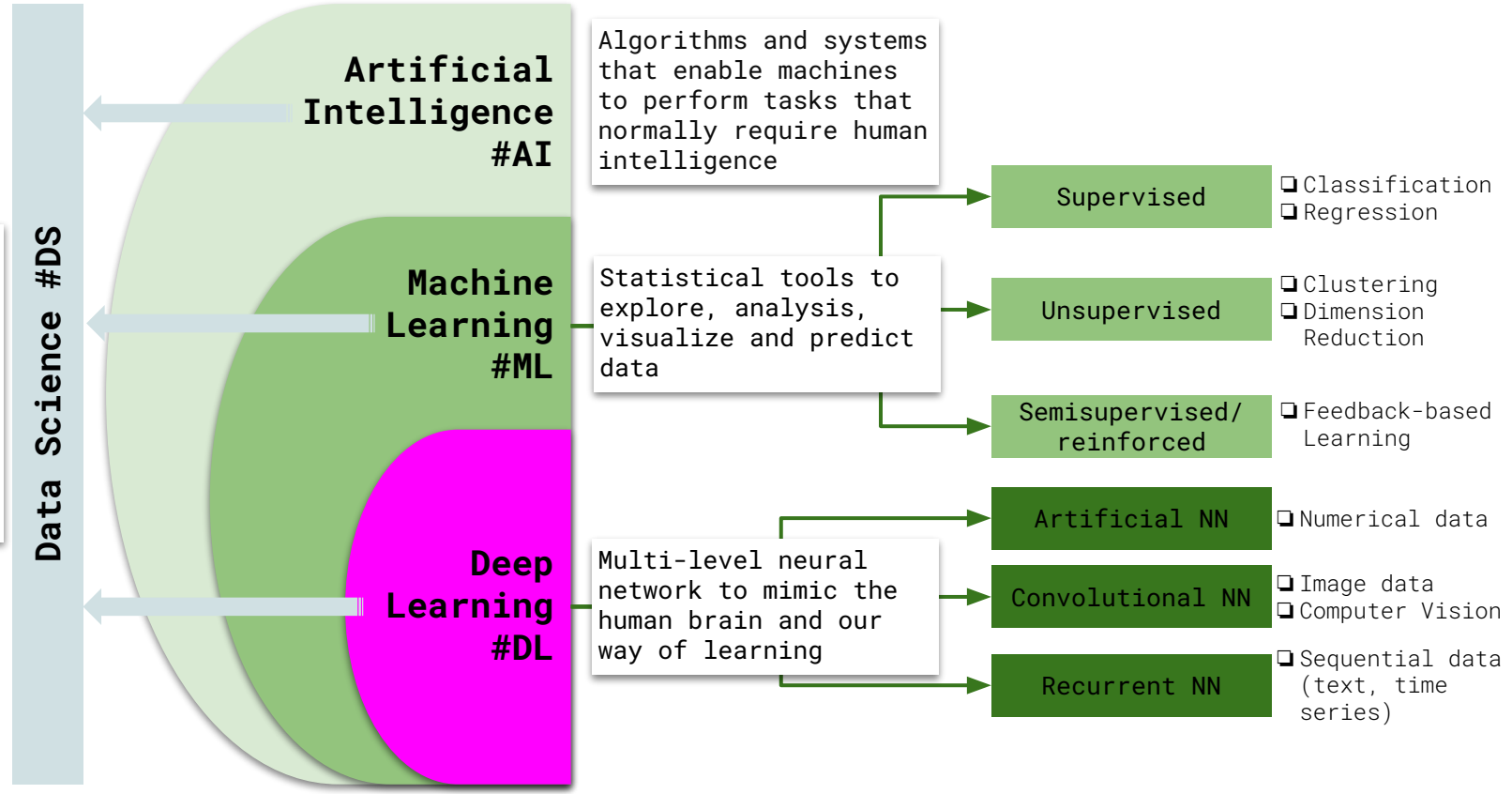
Is the gene expressed under drought?



$f(\cdot)$

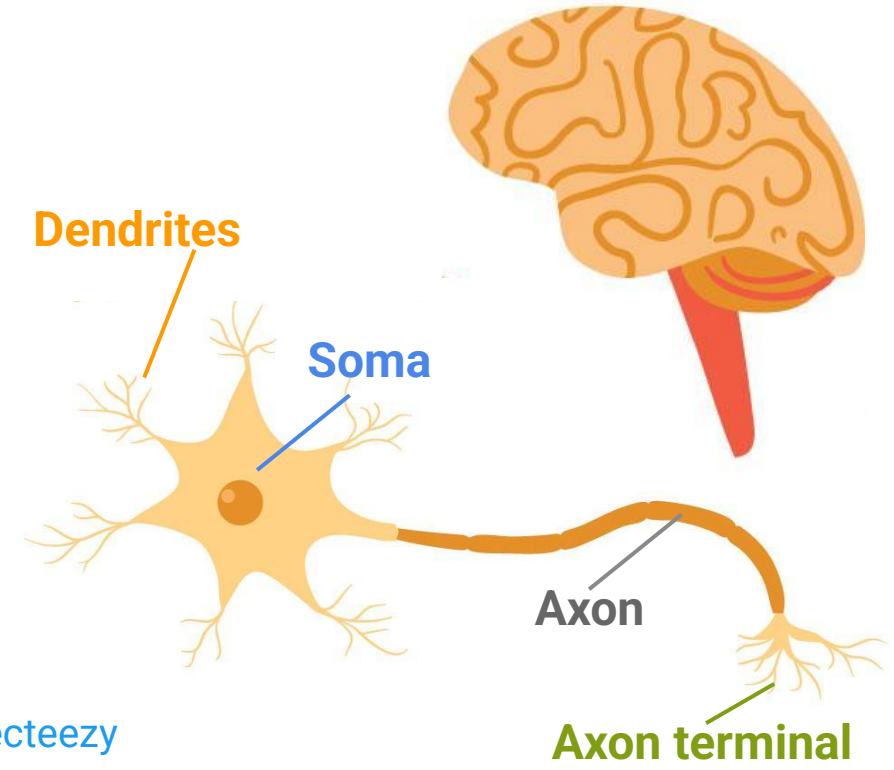
Is this chemical a potential pesticide?

#AI vs. #ML vs. #DL vs. #DS



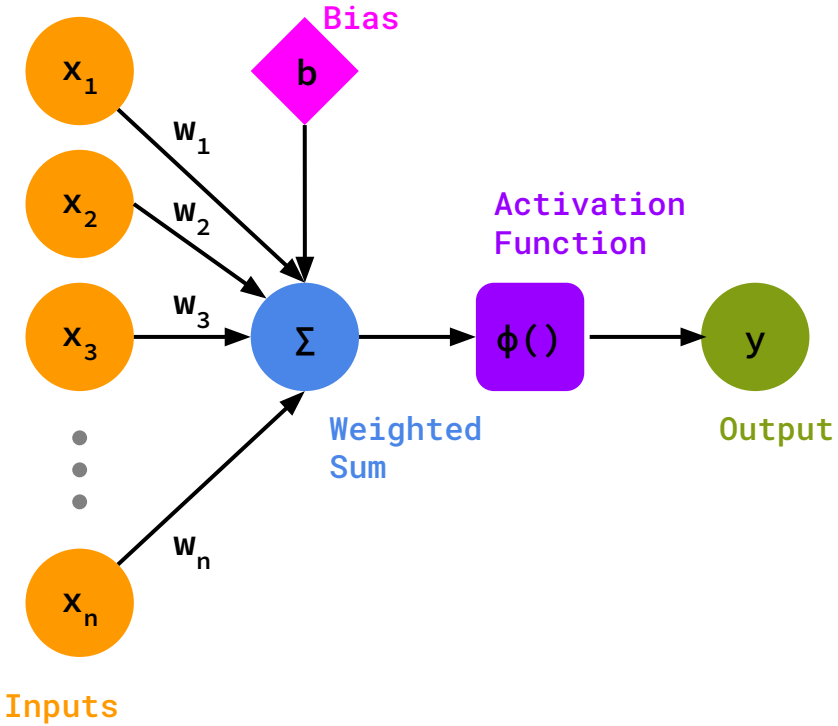
Neurons - Base Unit of our Brains

1. **Dendrites** receive input signals
2. **Soma** integrates input signals and checks whether a threshold is reached before sending an “action potential”
3. The action potential travels along the **Axon**
4. **Axon terminal** provides the final output signal

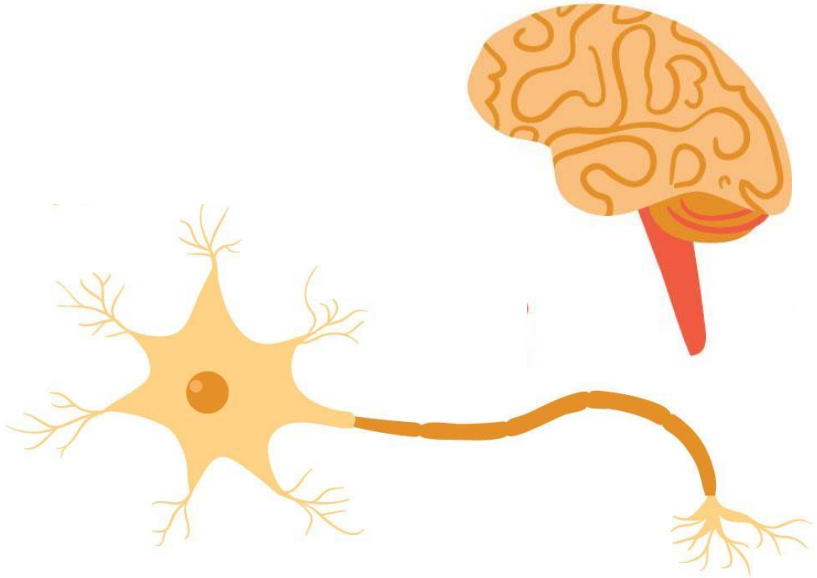


Perceptron - Base Unit of Artificial Neural Networks

Perceptron

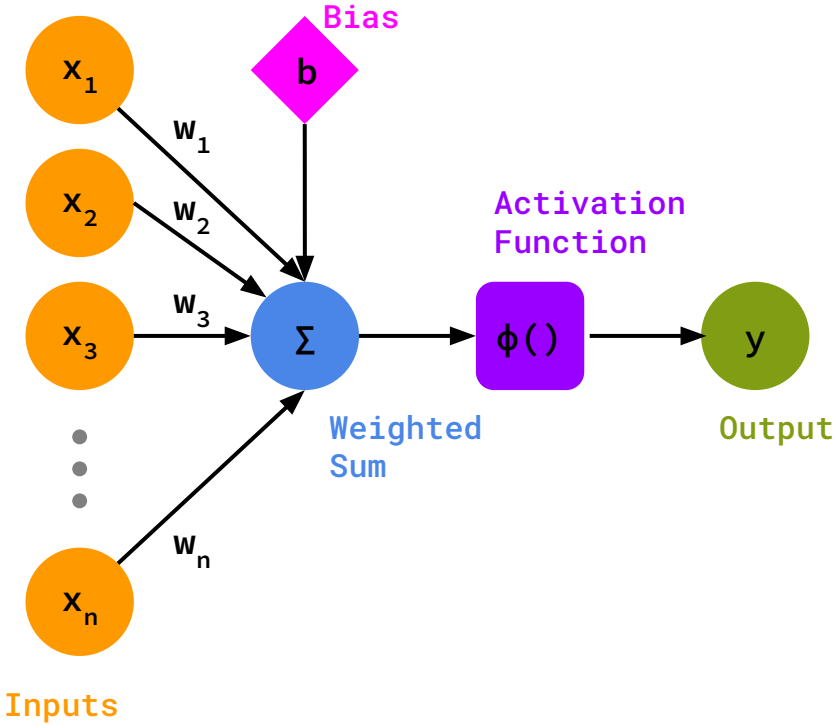


Neuron



Perceptron - Base Unit of Artificial Neural Networks

Perceptron

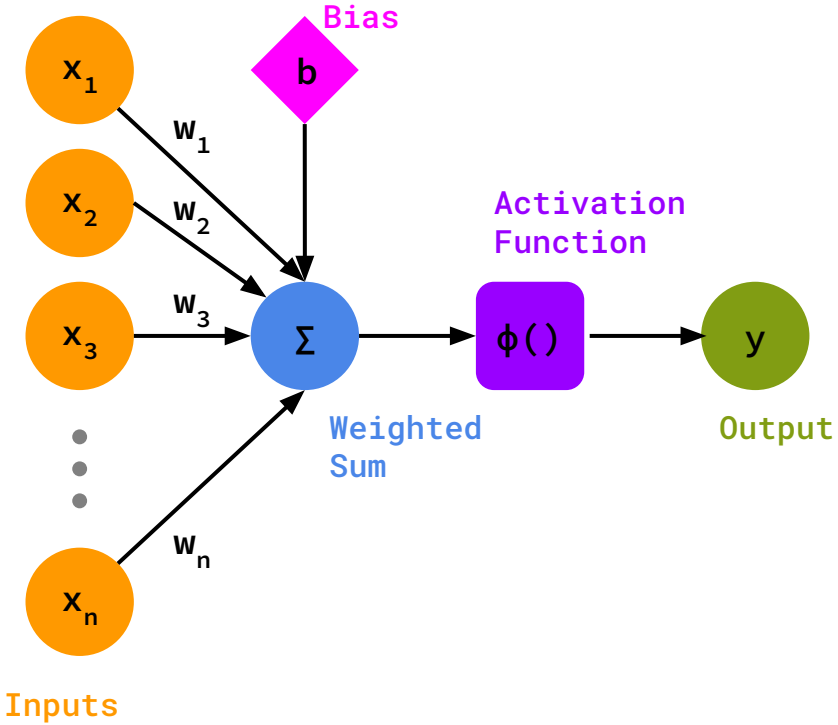


Inputs and Weights:

- Receives multiple numeric inputs (e.g., features of data).
- Each input has an associated weight indicating its importance.

Perceptron - Base Unit of Artificial Neural Networks

Perceptron



Inputs and Weights:

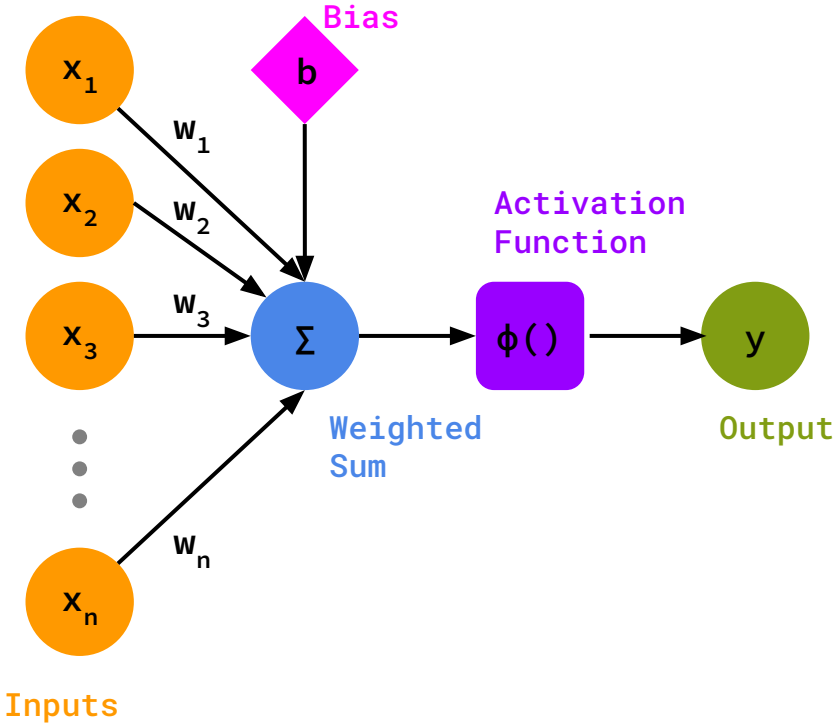
- Receives multiple numeric inputs (e.g., features of data).
- Each input has an associated weight indicating its importance.

Summation:

- Multiplies each input by its weight and sums these values.

Perceptron - Base Unit of Artificial Neural Networks

Perceptron



Inputs and Weights:

- Receives multiple numeric inputs (e.g., features of data).
- Each input has an associated weight indicating its importance.

Summation:

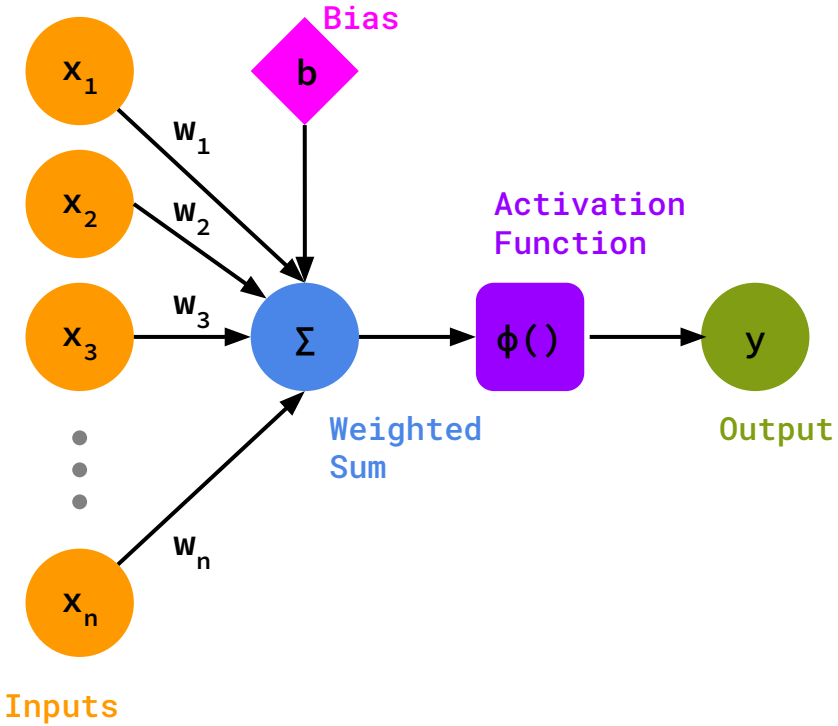
- Multiplies each input by its weight and sums these values.

Activation:

- Sum passes through an activation function (commonly a step function).
- Activation function outputs either 0 or 1, based on a threshold.

Perceptron - Base Unit of Artificial Neural Networks

Perceptron



Inputs and Weights:

- Receives multiple numeric inputs (e.g., features of data).
- Each input has an associated weight indicating its importance.

Summation:

- Multiplies each input by its weight and sums these values.

Activation:

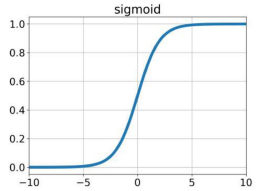
- Sum passes through an activation function (commonly a step function).
- Activation function outputs either 0 or 1, based on a threshold.

Output:

- The output from the activation function is the perceptron's decision (e.g., Yes/No, On/Off).

Activation Functions

$$\varphi(x) = \frac{1}{1 + e^{-x}}$$

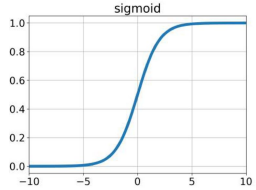


Sigmoid Function

- Output Range: Between 0 and 1, suitable for predicting probabilities.
- Characteristics:
  - Smooth and continuous curve.
  - As the input becomes very large or very small, the output approaches 1 or 0, respectively, but never exactly reaches those extremes.

Activation Functions

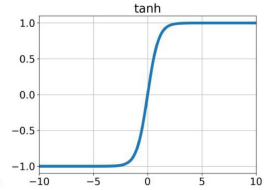
$$\varphi(x) = \frac{1}{1 + e^{-x}}$$



Sigmoid Function

- Output Range: Between 0 and 1, suitable for predicting probabilities.
- Characteristics:
  - Smooth and continuous curve.
  - As the input becomes very large or very small, the output approaches 1 or 0, respectively, but never exactly reaches those extremes.

$$\varphi(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

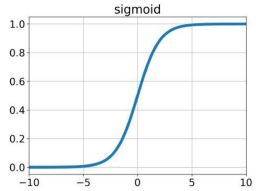


Hyperbolic Tangent Function (tanh)

- Output Range: Between -1 and 1,
- Characteristics:
  - smooth and continuous curve.
  - Shifted version of the sigmoid function, providing outputs that are symmetric around zero.
  - Faster convergence

Activation Functions

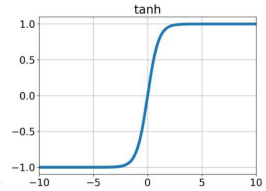
$$\varphi(x) = \frac{1}{1 + e^{-x}}$$



Sigmoid Function

- Output Range: Between 0 and 1, suitable for predicting probabilities.
- Characteristics:
  - Smooth and continuous curve.
  - As the input becomes very large or very small, the output approaches 1 or 0, respectively, but never exactly reaches those extremes.

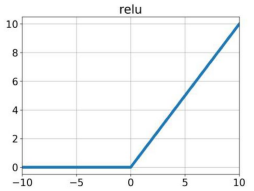
$$\varphi(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$



Hyperbolic Tangent Function (tanh)

- Output Range: Between -1 and 1,
- Characteristics:
  - smooth and continuous curve.
  - Shifted version of the sigmoid function, providing outputs that are symmetric around zero.
  - Faster convergence

$$\varphi(x) = \max(0, x)$$

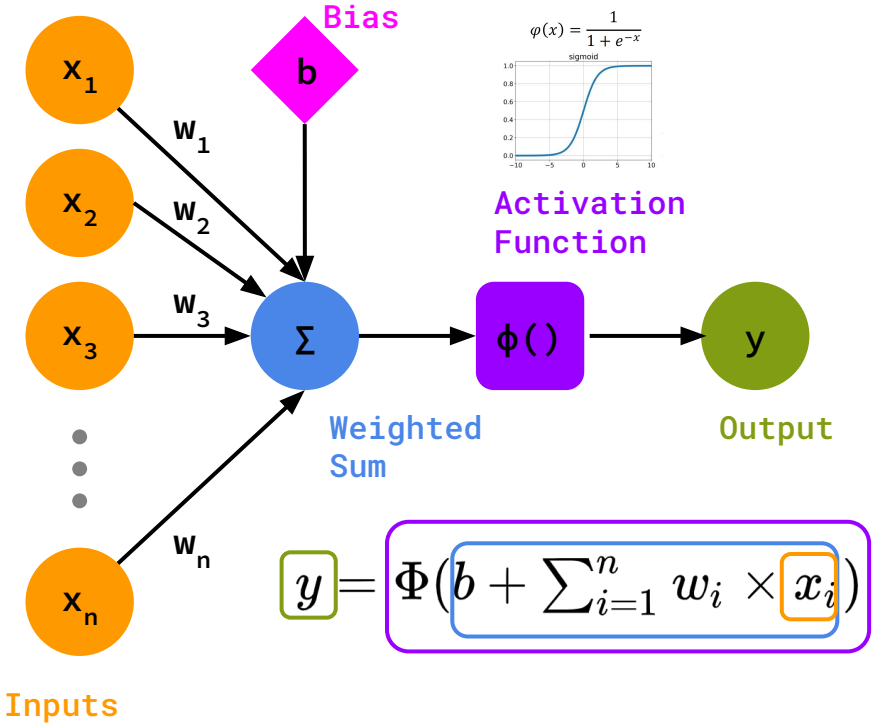


Rectified Linear Unit (ReLU)

- Output Range: From 0 to ∞
- Characteristics:
  - Non-linear, which allows for complex decision boundaries.
  - Computationally efficient as it involves simple thresholding at zero.
  - Faster convergence

Perceptron - Base Unit of Artificial Neural Networks

Perceptron



Inputs and Weights:

- Receives multiple numeric inputs (e.g., features of data).
- Each input has an associated weight indicating its importance.

Summation:

- Multiplies each input by its weight and sums these values.

Activation:

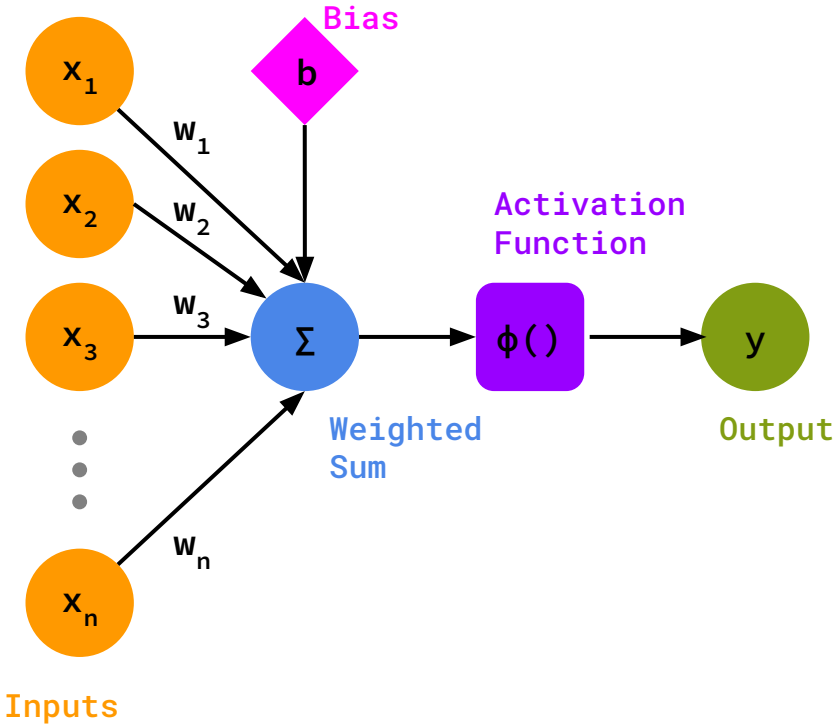
- Sum passes through an activation function (commonly a step function).
- Activation function outputs either 0 or 1, based on a threshold.

Output:

- The output from the activation function is the perceptron's decision (e.g., Yes/No, On/Off).

Perceptron - Base Unit of Artificial Neural Networks

Perceptron

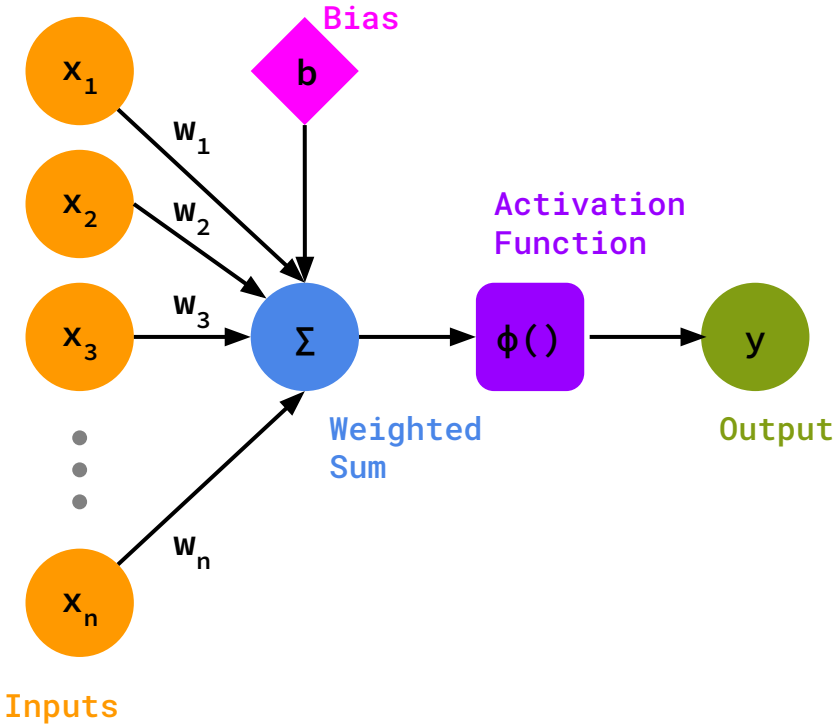


Learning Process:

- Begins with random weights.
- Adjusts weights based on errors in prediction (learning from mistakes).
- Repeats adjustments to improve accuracy over time.

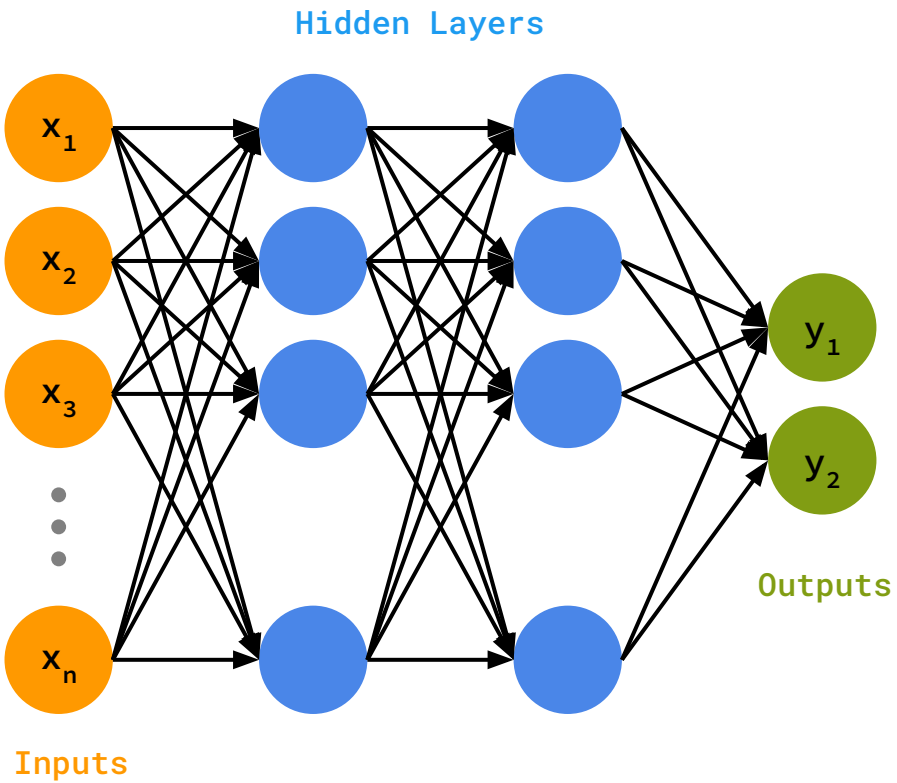
Perceptron - Base Unit of Artificial Neural Networks

Perceptron



- **Single Perceptron** is capable of making simple decisions by weighing inputs, applying a summation, and then passing the result through an activation function (like a step function) to produce an output.
- **Limitations:** A single perceptron can only solve problems that are linearly separable, meaning the data points can be divided by a straight line into classes.

Expansion to Deep Neural Networks (DNN)



DNN consists of multiple layers of neurons. The first layer receives the input data (like images, text, or sound), and each subsequent layer receives the output from the previous layer and transforms it further. The final layer produces the output of the network. This setup allows DNN to learn very complex patterns

Layer Types:

- **Input Layer:** Receives raw data.
- **Hidden Layers:** Intermediate processing layers; these can be numerous and are where most of the computation takes place.
- **Output Layer:** Produces the final decision or prediction of the network.

Split your Data

Dataset

Training Set

- **60 %** of the total data set
- Model learns to recognize patterns or features based on the training set
- Training sets should be balanced (each output class should be equally represented)
  - Real-world challenge
  - Be aware of possible model bias

Validation Set

- **20 %** of the total dataset
- Unbiased evaluation of the model fit while tuning model's hyperparameters (learning rates, number of hidden layers etc.)

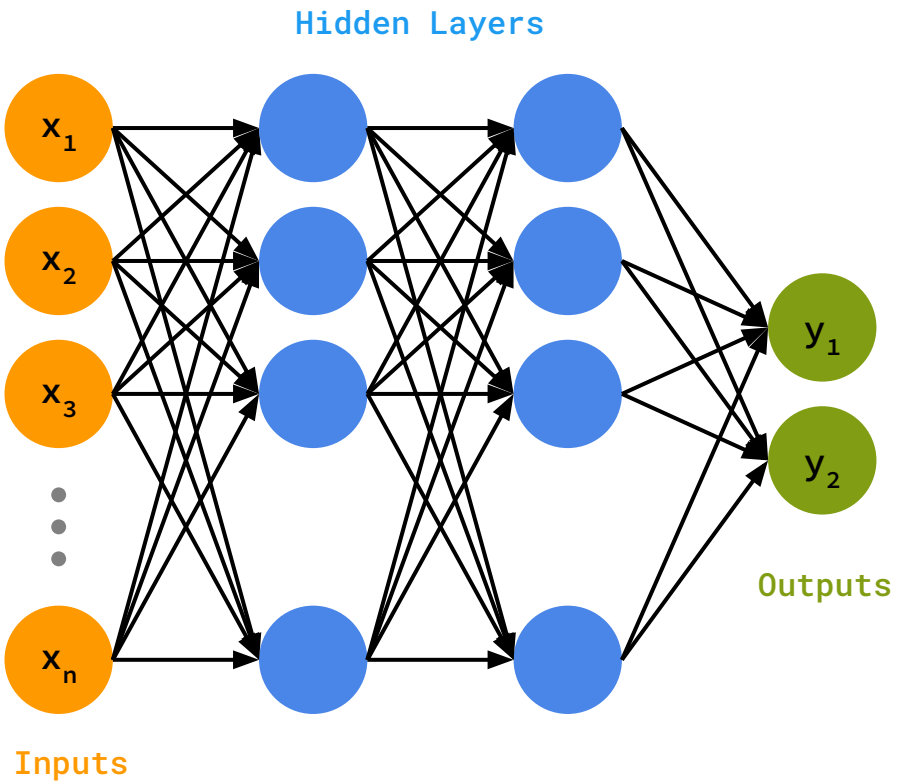
Fine-Tuning of Model Parameters

Testing Set

- **20 %** of the total dataset
- After training and validation, the model performance is assessed through the training set

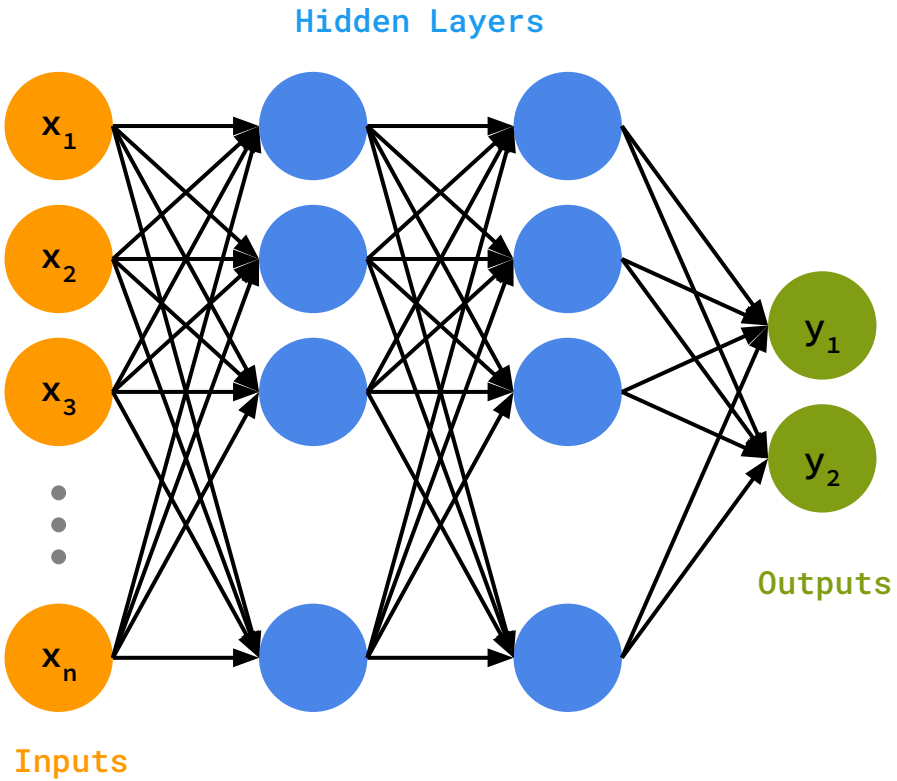
Avoid overfitting to ensure the model performs well on unseen data

Deep Neural Networks Training



- During training, the optimizer iterates over smaller subsets of training data, called **batches**
  - Efficient memory usage
  - Speed up training process
  - Avoid local minima during optimization
- One round of training, complete pass of training data is called an **epoch**.
  - Multiple rounds of epochs are required to learn.
- With each epoch, the prediction becomes more accurate by adjusting the weights by a process called **backpropagation**

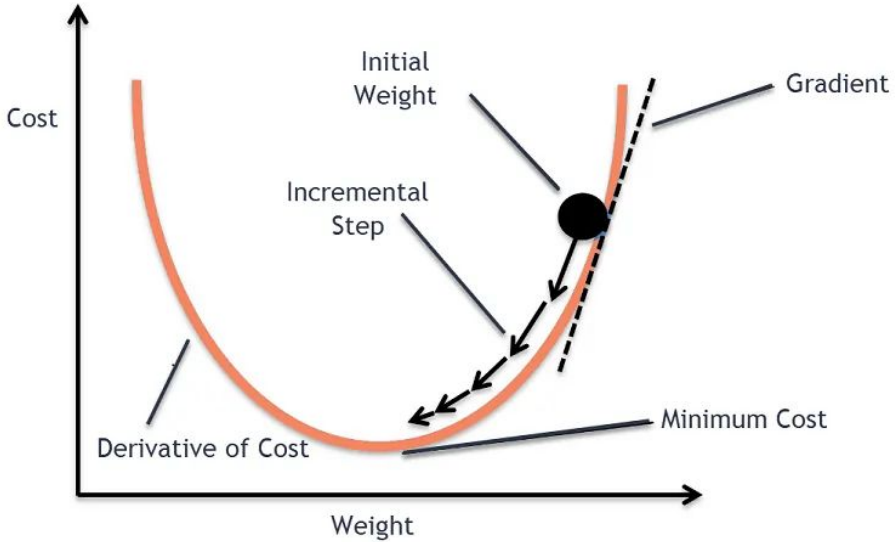
Deep Neural Networks Training



Process of Backpropagation:

- After a forward pass through the network (computing predictions from inputs), the error is calculated using a loss function.
- This error is then propagated backwards through the network (hence "backpropagation"), allowing the model to calculate the gradient of the loss function with respect to each weight in the network.
- Using these gradients, the model's weights are adjusted via an optimization algorithm (like gradient descent) to minimize the loss.

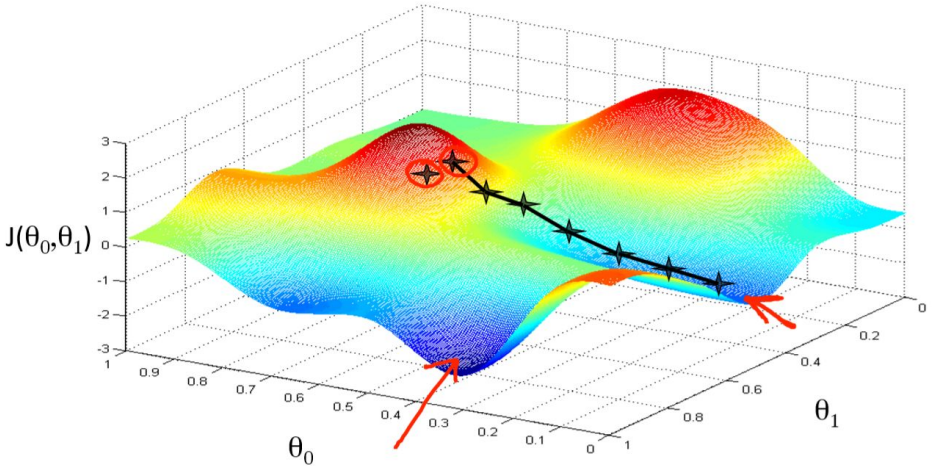
Deep Neural Networks Training



Process of Backpropagation:

- After a forward pass through the network (computing predictions from inputs), the error is calculated using a loss function.
- This error is then propagated backwards through the network (hence "backpropagation"), allowing the model to calculate the gradient of the loss function with respect to each weight in the network.
- Using these gradients, the model's weights are adjusted via an optimization algorithm (like gradient descent) to minimize the loss.

Deep Neural Networks Training



Process of Backpropagation:

- After a forward pass through the network (computing predictions from inputs), the error is calculated using a loss function.
- This error is then propagated backwards through the network (hence "backpropagation"), allowing the model to calculate the gradient of the loss function with respect to each weight in the network.
- Using these gradients, the model's weights are adjusted via an optimization algorithm (like gradient descent) to minimize the loss.

Evaluate Model Performance

Confusion Matrix

Not a score or ratio, but shows the counts for true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN)

Gene Expression	Predicted: Low	Predicted: High
Actual: Low	85 (TN)	15 (FP)
Actual: High	20 (FN)	80 (TP)

Evaluate Model Performance

Confusion Matrix

Not a score or ratio, but shows the counts for true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN)

Gene Expression	Predicted: Low	Predicted: High
Actual: Low	85 (TN)	15 (FP)
Actual: High	20 (FN)	80 (TP)

Accuracy

This measures how often the classifier makes the correct prediction. It's the sum of TP and TN divided by the total number of cases.

TP + TN / (TP + TN + FP + FN) = 82.5 %

Evaluate Model Performance

Confusion Matrix

Not a score or ratio, but shows the counts for true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN)

Gene Expression	Predicted: Low	Predicted: High
Actual: Low	85 (TN)	15 (FP)
Actual: High	20 (FN)	80 (TP)

Accuracy

This measures how often the classifier makes the correct prediction. It's the sum of TP and TN divided by the total number of cases.

TP + TN / (TP + TN + FP + FN) = 82.5 %

Precision

This is the proportion of positive identifications that were actually correct. It is calculated as TP divided by the sum of TP and FP.

➔ *Crucial when the cost of a false positive is high*

TP / (TP + FP) = 84.2 %

Recall (Sensitivity)

This is the ability of the model to find all the relevant cases (all actual Highs). It is TP divided by the sum of TP and FN

➔ *Crucial when the cost for false negative is high*

TP / (TP + FN) = 80 %

Evaluate Model Performance

Confusion Matrix

Not a score or ratio, but shows the counts for true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN)

Gene Expression	Predicted: Low	Predicted: High
Actual: Low	85 (TN)	15 (FP)
Actual: High	20 (FN)	80 (TP)

Accuracy

This measures how often the classifier makes the correct prediction. It's the sum of TP and TN divided by the total number of cases.

TP + TN / (TP + TN + FP + FN) = 82.5 %

Precision

This is the proportion of positive identifications that were actually correct. It is calculated as TP divided by the sum of TP and FP.

➔ *Crucial when the cost of a false positive is high*

TP / (TP + FP) = 84.2 %

Recall (Sensitivity)

This is the ability of the model to find all the relevant cases (all actual Highs). It is TP divided by the sum of TP and FN

➔ *Crucial when the cost for false negative is high*

TP / (TP + FN) = 80 %

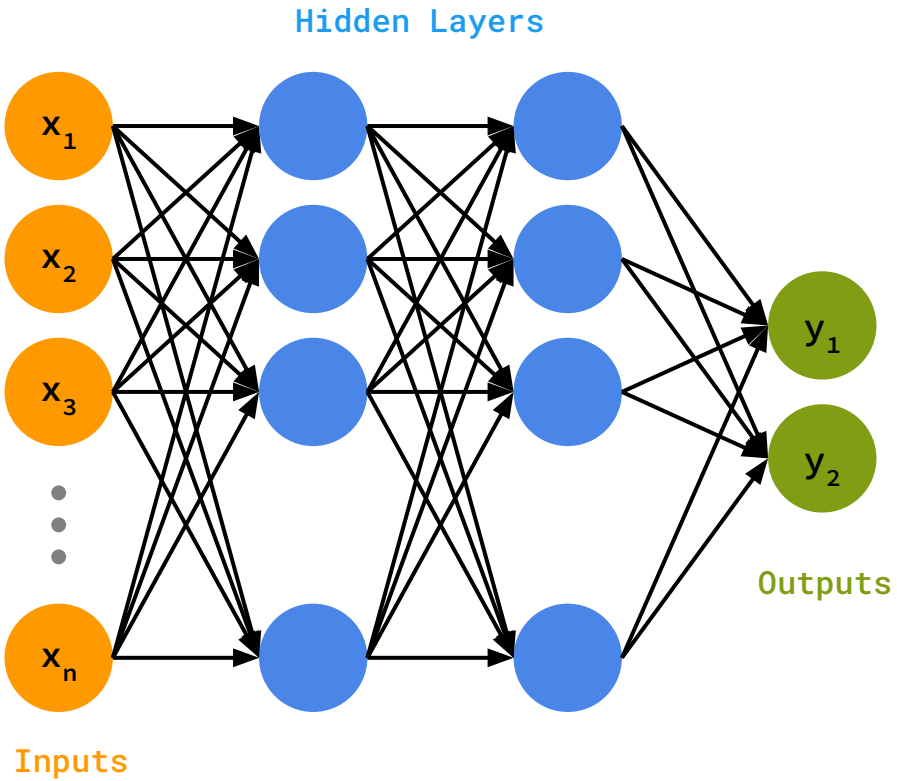
F1 Score

This is the harmonic mean of precision and recall, which provides a balance between them.

➔ *Particularly useful when the class distribution is uneven*

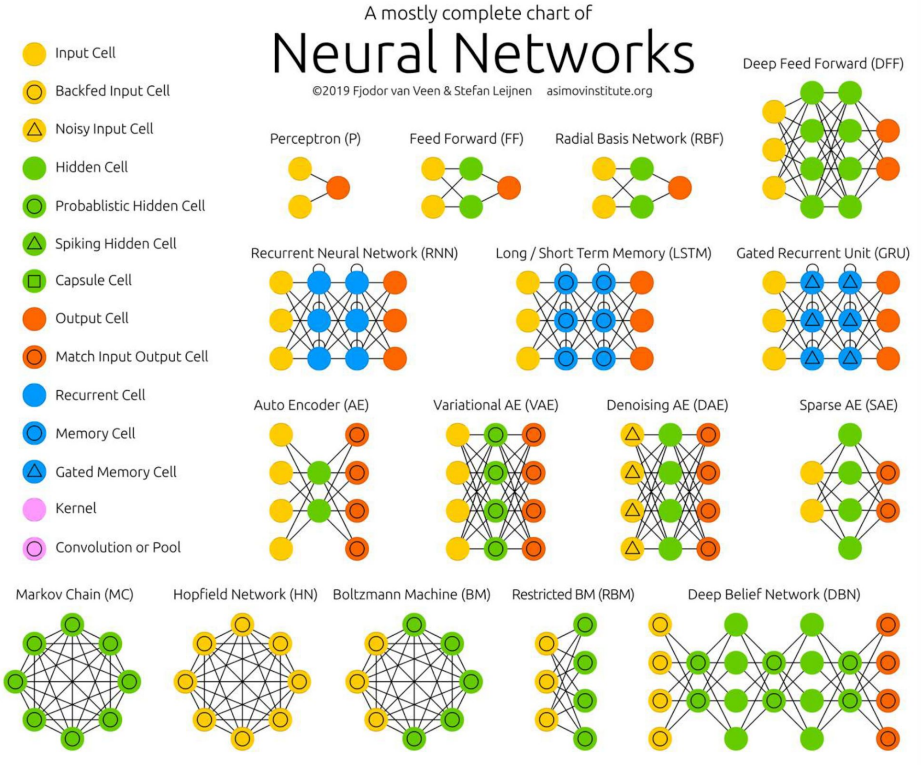
*2 * (Precision * Recall) /(Precision + Recall) = 0.821*

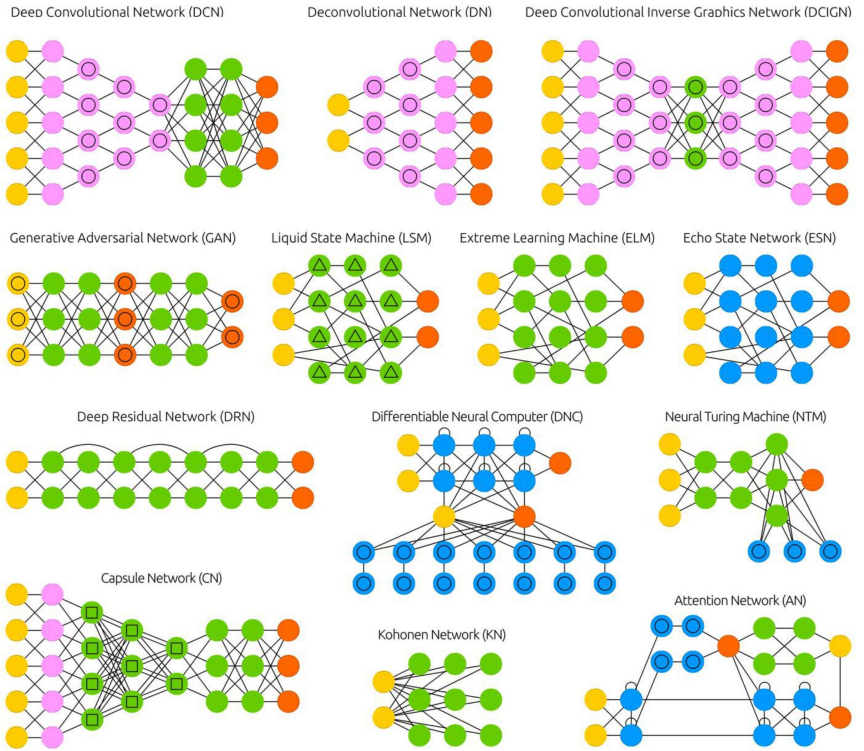
Expansion to Deep Neural Networks (DNN)



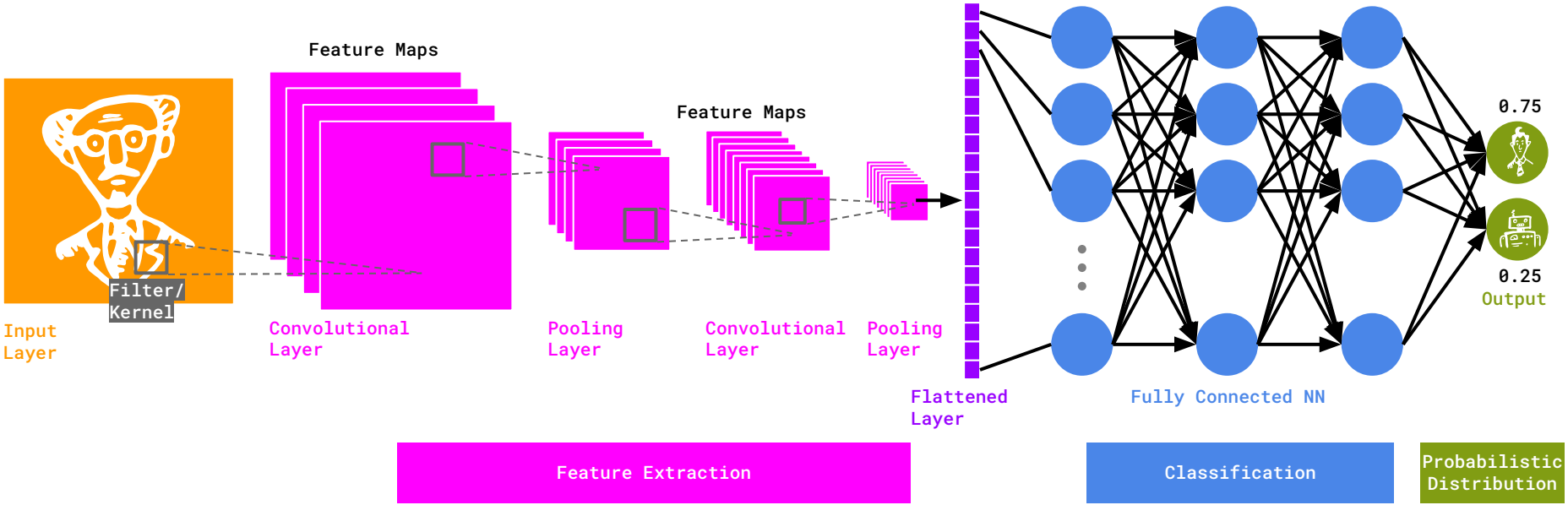
- **Versatility**:
  Deep neural networks are used across a wide range of applications from image and speech recognition to playing complex games like Go and autonomous driving.
- **Learning from Data:**
  They excel in environments where there are vast amounts of data to learn from, making them a cornerstone technology in AI.

Artificial Neural Networks ANN





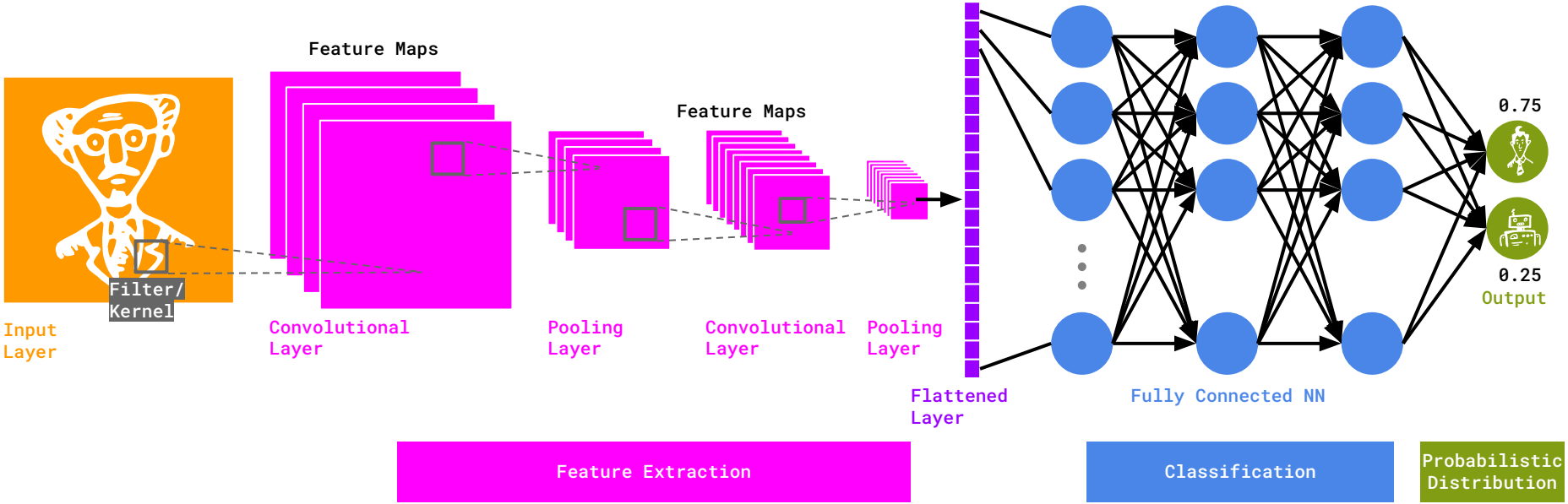
Convolutional Neural Networks (CNN)



Applications in Computer Vision

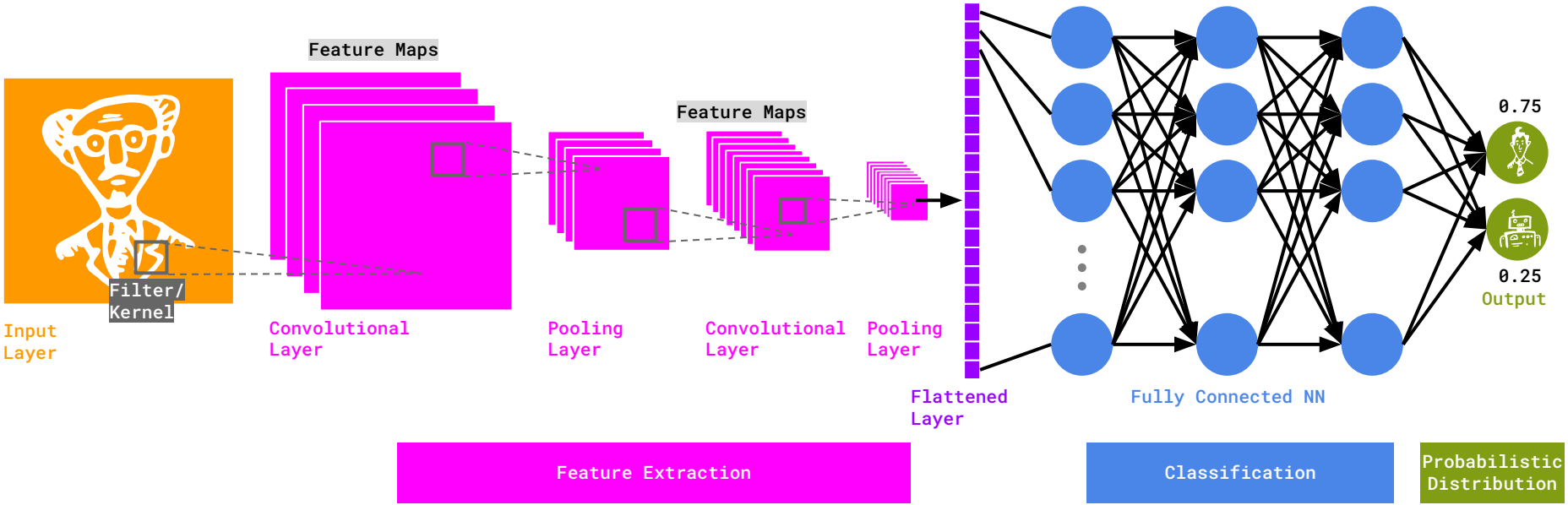
- Image and Video Recognition: CNNs are particularly powerful for tasks involving image and video recognition, spatial recognition for autonomous driving, and much more.
- Image Classification: Assigning a label to an image based on the visual content.
- Object Detection: Identifying objects within an image and drawing bounding boxes around them.

Convolutional Neural Networks (CNN)



Input Layers receive the raw input image with its height, width, and color depth (e.g., RGB images have three color channels).

Convolutional Neural Networks (CNN)

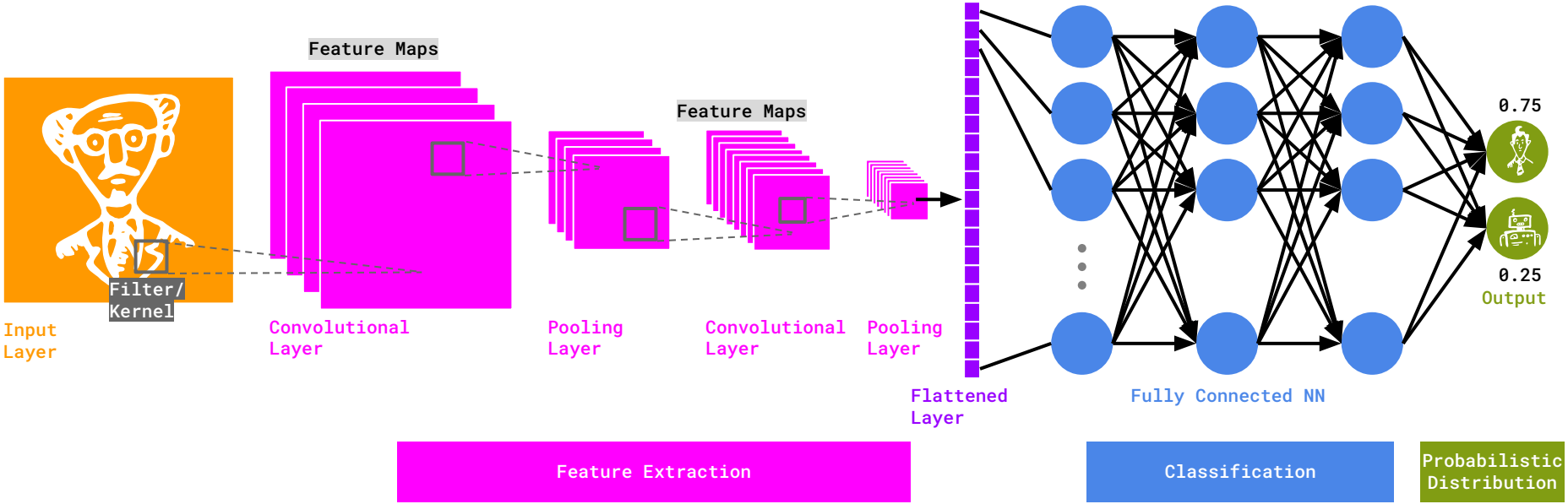


Convolutional Layers use filters (or kernels) that perform convolution operations as they are slid (across the input image to create feature maps. This process captures the local dependencies in the original image.

Filters: Small matrices that traverse through the image. As they move, they learn patterns such as edges, corners, colors

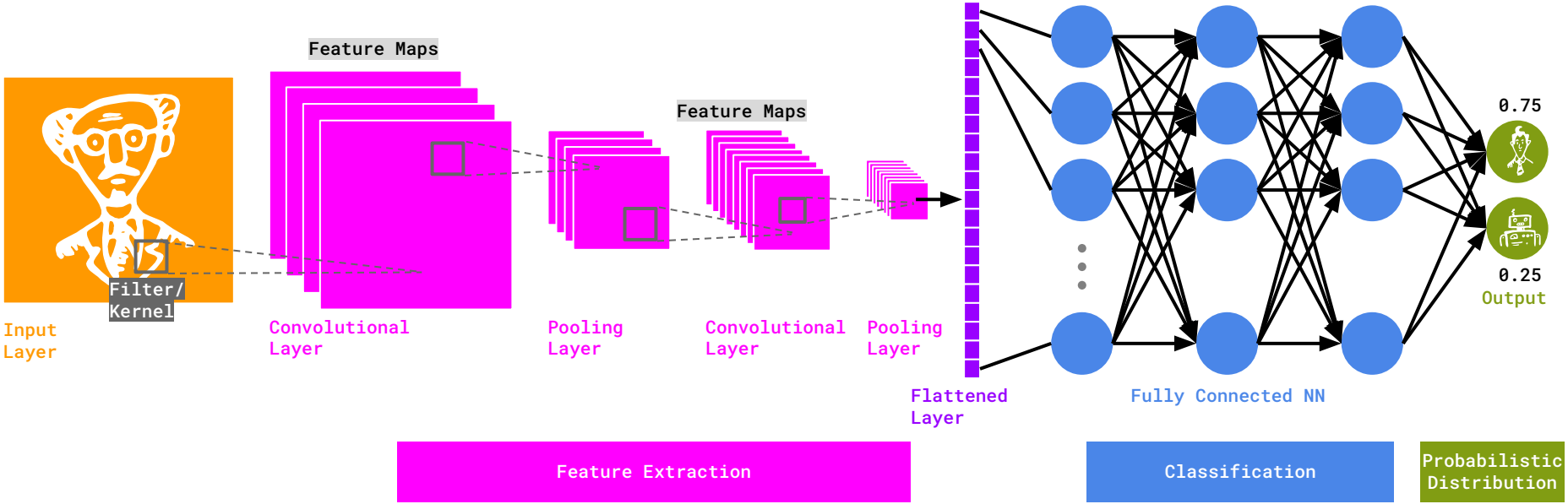
Feature Maps: Result from applying filters to the image. Each feature map represents a particular feature detected in the image.

Convolutional Neural Networks (CNN)



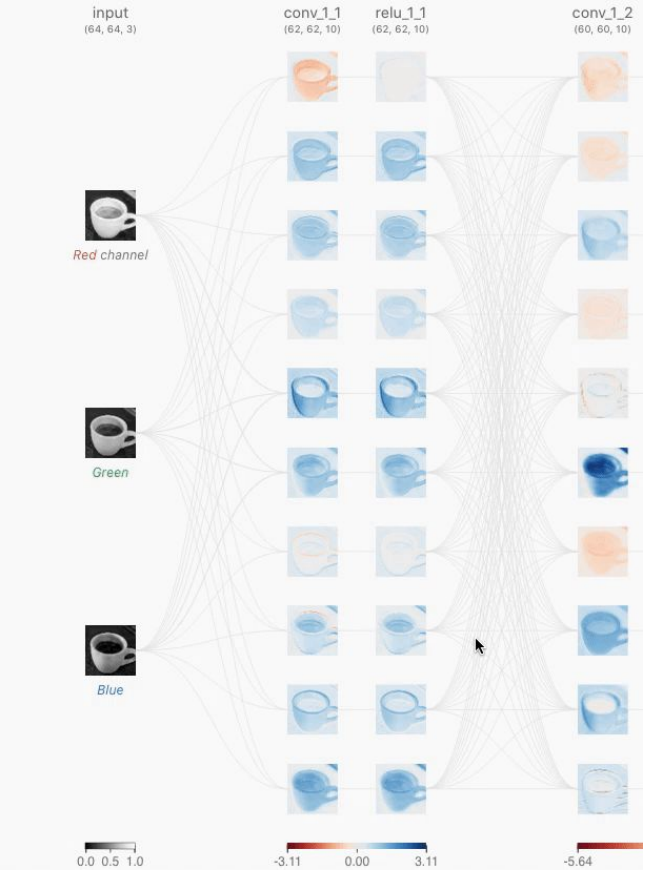
Pooling Layers are used intermittently between successive convolutional layers to reduce the spatial size of the representation, decrease the amount of parameters, and computation in the network, and hence to also control overfitting. Pooling layers summarize the features present in regions of the feature map generated by convolutions.

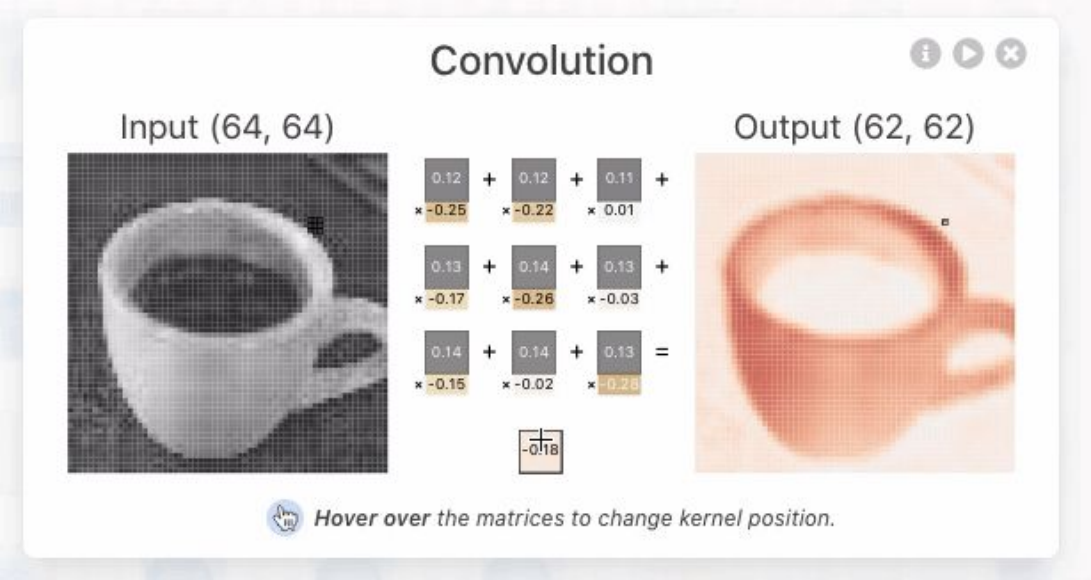
Convolutional Neural Networks (CNN)



Fully Connected Networks have neurons that have connections to all activations in the previous layer, as seen in regular neural networks. Their role is to output the predictions for the task based on the features extracted by the convolutional layers. The Fully Connected Network performs the actual reasoning, e.g. classification.

CNN Explainer

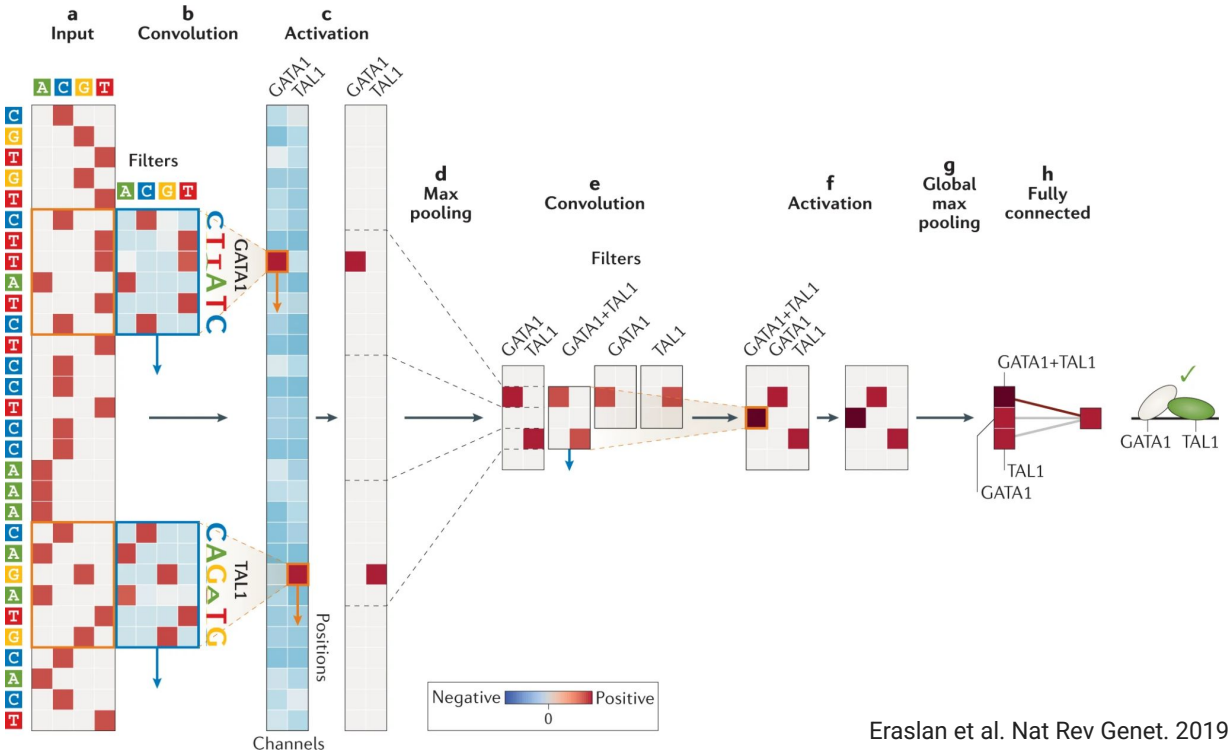




https://github.com/poloclub/cnn-explainer


IPK
LEIBNIZ INSTITUTE
Genetic Sequences as Images to detect elements, domains, regulatory sequences etc.
[https://media.sciencephoto.com/]

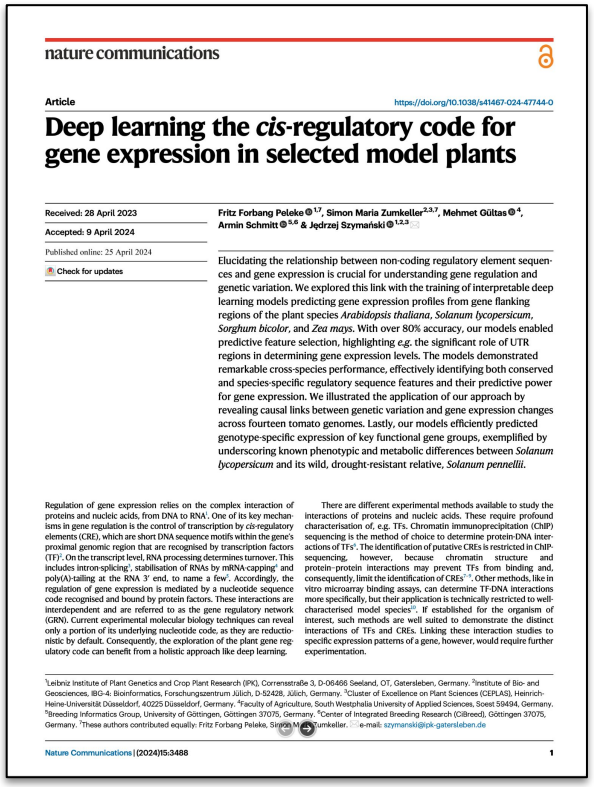
CNNs to predict Transcription Factor Binding



Eraslan et al. Nat Rev Genet. 2019

a | One-hot encoded representation of the DNA sequence.
b | The first convolutional layer scans the input sequence using filters.
c | Negative values are truncated to 0 using the rectified-linear unit (ReLU) activation function.
d | In the max pooling operation, contiguous bins of the activation map are summarized by taking the maximum value for each channel in each bin.
e | The second convolutional layer scans the sequence for pairs of motifs and for instances of individual motifs.
f | Similarly to that of the first convolution, ReLU activation function is applied.
g | The maximum value across all positions for each channel is selected.
h | A fully connected layer is used to make the final prediction.

CNNs to predict Gene Expression



nature communications

Article

https://doi.org/10.1038/s41467-024-47744-0

Deep learning the *cis*-regulatory code for gene expression in selected model plants

Received: 28 April 2023

Accepted: 9 April 2024

Published online: 25 April 2024

Check for updates

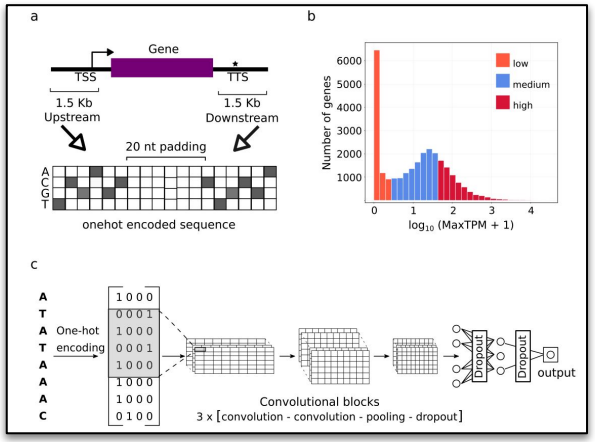
Fritz Forbang Peleke[1,7], Simon Maria Zumkeller[2,3,7], Mehmet Gültas[4], Armin Schmitt[5,6] & Jędrzej Szymański[1,2,3]

Elucidating the relationship between non-coding regulatory element sequences and gene expression is crucial for understanding gene regulation and genetic variation. We explored this link with the training of interpretable deep learning models predicting gene expression profiles from gene flanking regions of the plant species *Arabidopsis thaliana*, *Solanum lycopersicum*, *Sorghum bicolor*, and *Zea mays*. With over 80% accuracy, our models enabled predictive feature selection, highlighting *e.g.* the significant role of UTR regions in determining gene expression levels. The models demonstrated remarkable cross-species performance, effectively identifying both conserved and species-specific regulatory sequence features and their predictive power for gene expression. We illustrated the application of our approach by revealing causal links between genetic variation and gene expression changes across fourteen tomato genomes. Lastly, our models efficiently predicted genotype-specific expression of key functional gene groups, exemplified by underscoring known phenotypic and metabolic differences between *Solanum lycopersicum* and its wild, drought-resistant relative, *Solanum pennellii*.

Regulation of gene expression relies on the complex interaction of proteins and nucleic acids, from DNA to RNA[1]. One of its key mechanisms in gene regulation is the control of transcription by *cis*-regulatory elements (CRE), which are short DNA sequence motifs within the gene's proximal genomic region that are recognised by transcription factors (TF)[2]. On the transcript level, RNA processing determines turnover. This includes intron-splicing[3], stabilisation of RNAs by mRNA-capping[4] and poly(A)-tailing at the RNA 3' end, to name a few[5]. Accordingly, the regulation of gene expression is mediated by a nucleotide sequence code recognised and bound by protein factors. These interactions are interdependent and are referred to as the gene regulatory network (GRN). Current experimental molecular biology techniques can reveal only a portion of its underlying nucleotide code, as they are reductionistic by default. Consequently, the exploration of the plant gene regulatory code can benefit from a holistic approach like deep learning.

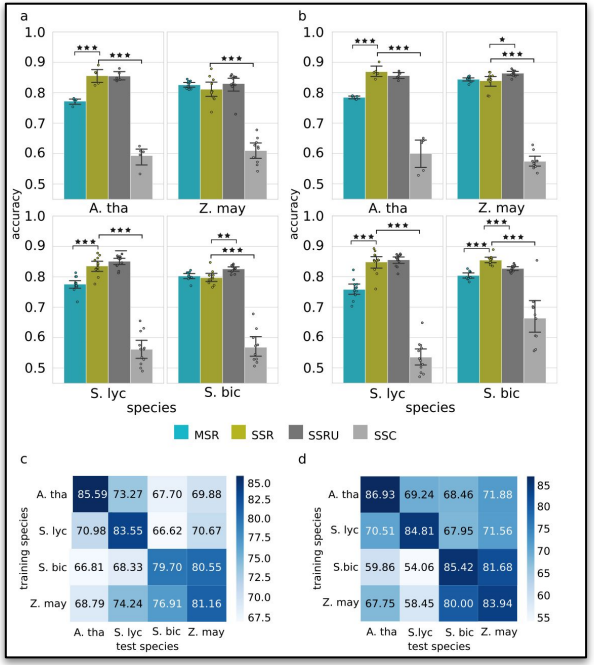
There are different experimental methods available to study the interactions of proteins and nucleic acids. These require profound characterisation of, e.g. TFs. Chromatin immunoprecipitation (ChIP)-sequencing is the method of choice to determine protein-DNA interactions of TFs[6]. The identification of putative CREs is restricted in ChIP-sequencing, however, because chromatin structure and protein–protein interactions may prevent TFs from binding and, consequently, limit the identification of CREs[7–9]. Other methods, like in vitro microarray binding assays, can determine TF-DNA interactions more specifically, but their application is technically restricted to well-characterised model species[10]. If established for the organism of interest, such methods are well suited to demonstrate the distinct interactions of TFs and CREs. Linking these interaction studies to specific expression patterns of a gene, however, would require further experimentation.

[1]Leibniz Institute of Plant Genetics and Crop Plant Research (IPK), Corrensstraße 3, D-06466 Seeland, OT, Gatersleben, Germany. [2]Institute of Bio- and Geosciences, IBG-4: Bioinformatics, Forschungszentrum Jülich, D-52428, Jülich, Germany. [3]Cluster of Excellence on Plant Sciences (CEPLAS), Heinrich-Heine-Universität Düsseldorf, 40225 Düsseldorf, Germany. [4]Faculty of Agriculture, South Westphalia University of Applied Sciences, Soest 59494, Germany. [5]Breeding Informatics Group, University of Göttingen, Göttingen 37075, Germany. [6]Center of Integrated Breeding Research (CiBreed), Göttingen 37075, Germany. [7]These authors contributed equally: Fritz Forbang Peleke, Simon Maria Zumkeller. e-mail: szymanski@ipk-gatersleben.de

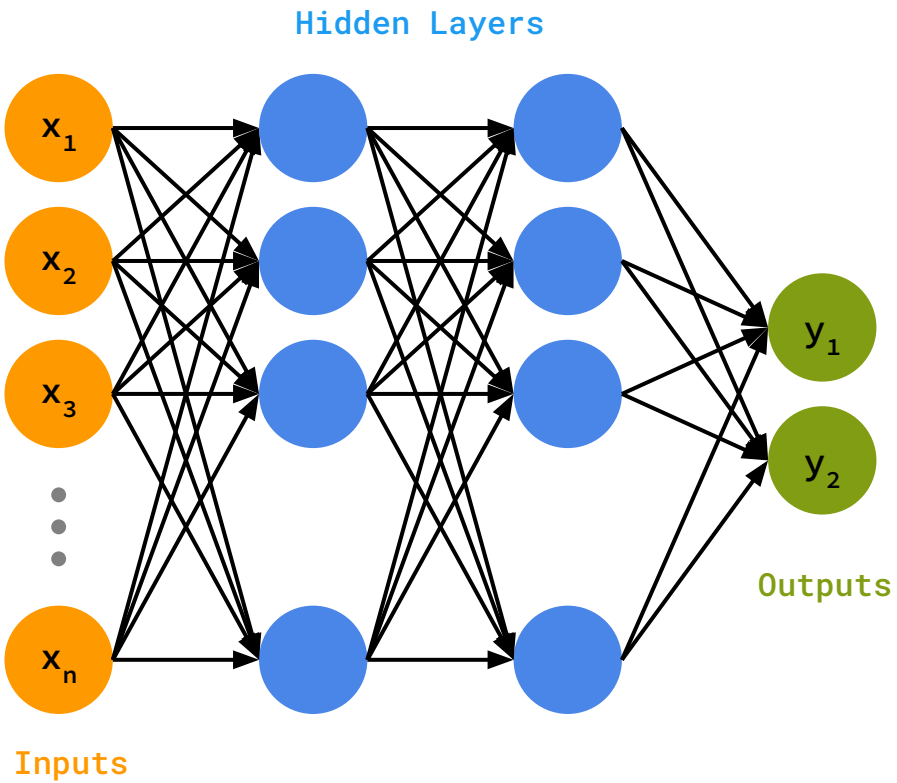


CNN for gene expression prediction required the extraction of proximal gene sequence, estimation and classification of transcript levels and nucleotide sequence via one-hot-encoding.

Comparison of predictive performance of deep learning CNN gene expression prediction models for crop plants under varying combinations of training data.

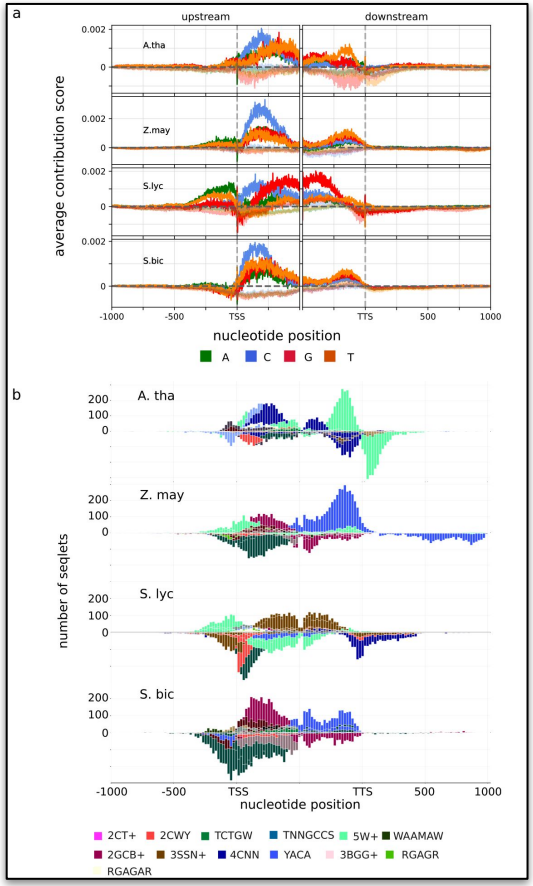


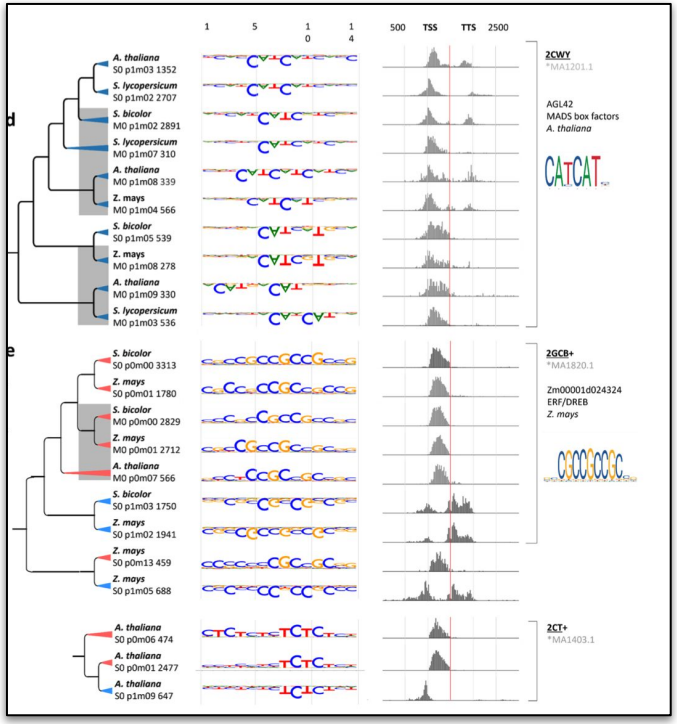
Expansion to Deep Neural Networks (DNN)



- **Deep Neural Networks**, are often considered "black boxes" because they involve complex calculations that are not easily understood or visualized.
- **Interpretable deep learning** involves creating models where the reasoning behind their decisions can be clearly understood by humans. These models provide insights into how and why they reach specific conclusions.
- **Feature Importance**: Techniques that allow evaluation of how different features influence the model's output directly, which can help in understanding the underlying mechanics in a DNN.

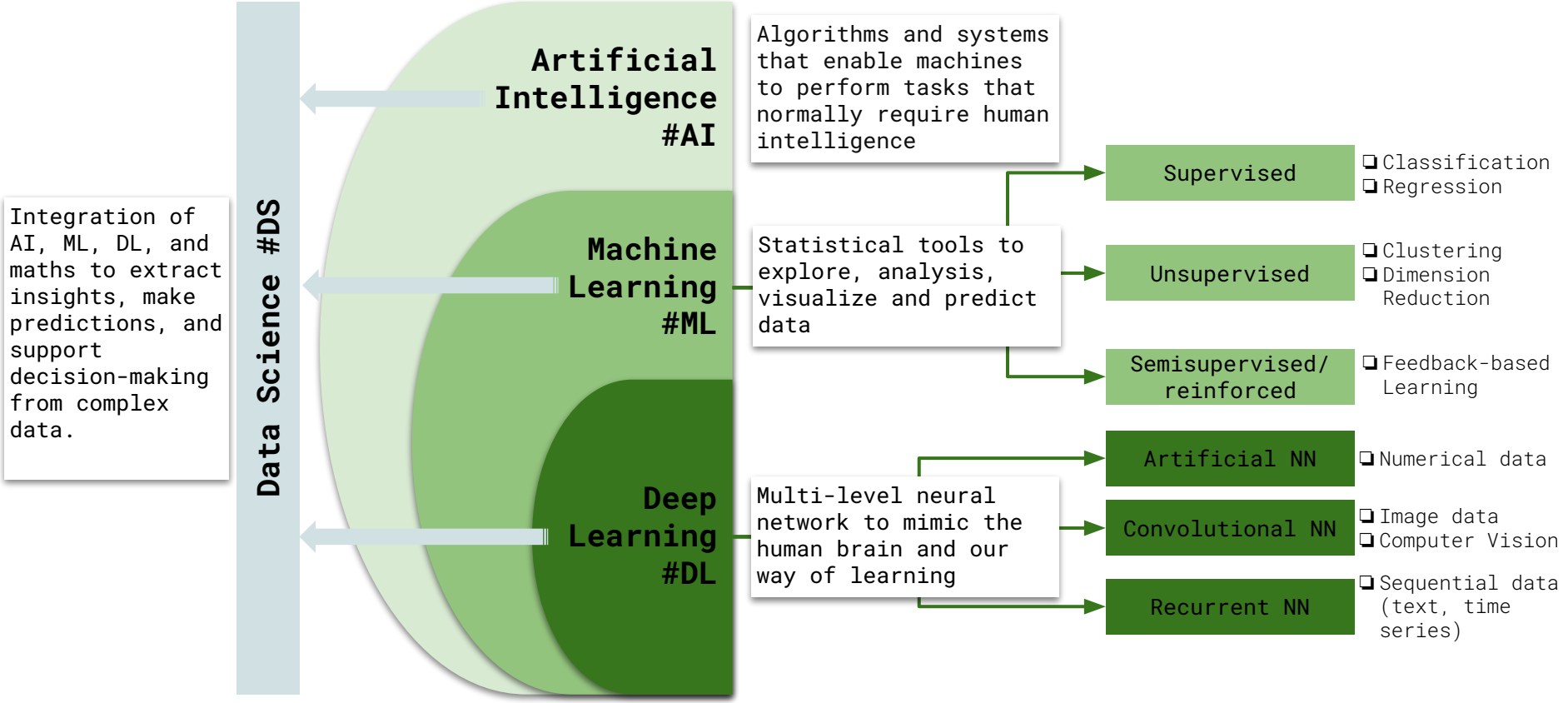
DeepLift and TF-MoDISco = Explainable DL Tools





- Predictive nucleotides are located mainly in the regions near the TSS and TTS
- Expression predictive motifs (EPMs) identified by DeepLIFT and TF-MoDISco

#AI vs. #ML vs. #DL vs. #DS



Harnessing AI Technology for Plant Science

- Importance of #AI and #ML in plant science to address the BigData challenge
- Differentiating between #AI, #ML and #DL
- #ML and #DL concepts
- Application of #ML and #DL in plant science and biological data analysis (gene expression)

Advancing Plant Science with #AI and #ML

- Transformation of plant research through advanced computational methods.
- Enhancing understanding of plant biology and accelerating breeding programs.
- Future prospects: integrating AI and ML for sustainable agriculture and global food security.


Questions?
[Generated by MidJourney]