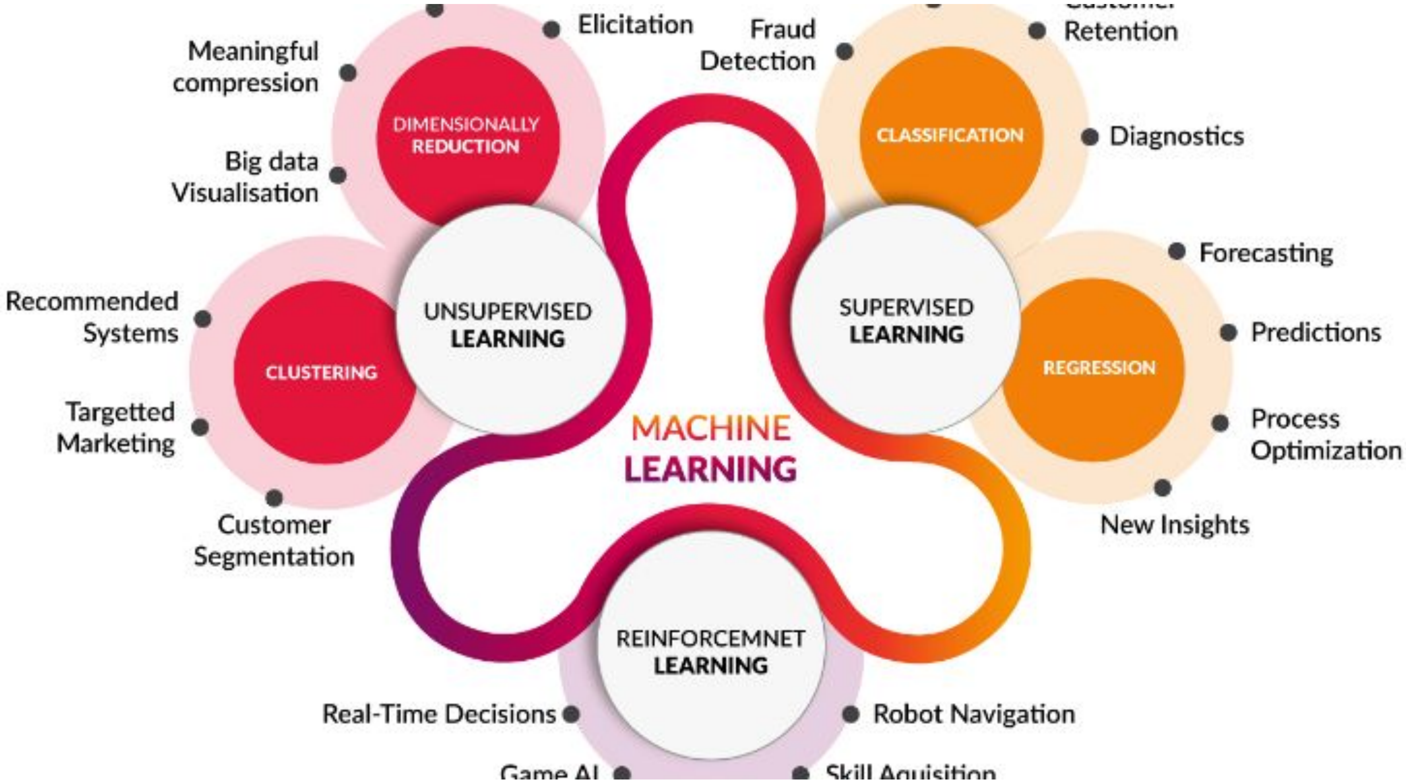
Source: **towardsdatascience.com**

Machine learning types



Fitting a model to noise

Can we hypothesize a nice parametric function for this kind of plot?

Or..

Fitting a model to noise

Short answer.

Yes. Using neural networks.

All it takes is 4 lines of code to design neural nets.

Fitting a model to noise

ML/DL may find you the right hypothesis. Here is a ***hello world*** neural network.

```
input_layer=Input(shape=(4,)) #<--4 inputs/features

output_layer=Dense(1)(input_layer) #<--1 output

model=Model(input_layer,output_layer)

model.compile(optimizer="adam",loss="mse")
```

Types of optimizers: Adam, Stochastic Gradient Descent, Adagrad, Adadelta

Types of losses: mse, binary_crossentropy(yes/no problems), categorical_crossentropy(>2 classes)

Fitting a model to noise

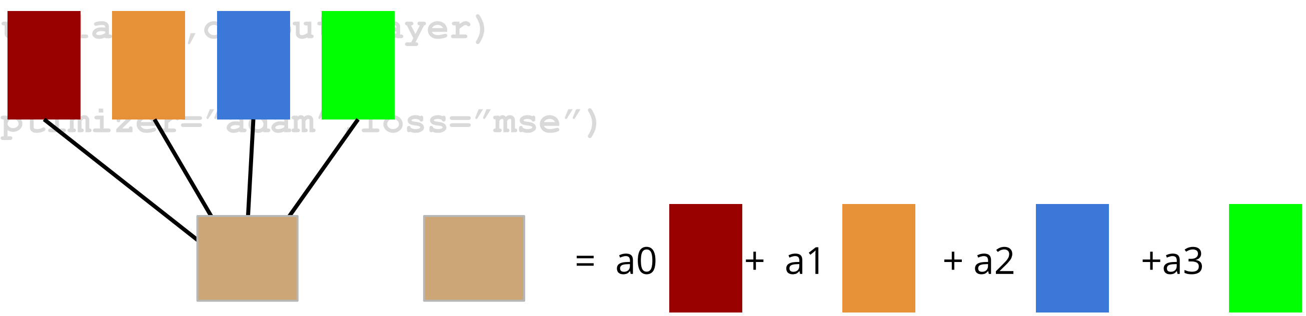
ML/DL may find you the right hypothesis. Here is a ***hello world*** neural network.

```
input_layer=Input(shape=(4,)) #<--4 inputs/features

output_layer=Dense(1)(input_layer) #<--1 output

model=Model(input_layer,output_layer)

model.compile(op[illegible]mi[illegible]="[illegible]m[illegible]oss="mse")
```

Fitting a model to noise

ML/DL may find you the right hypothesis. Here is a ***hello world*** neural network.

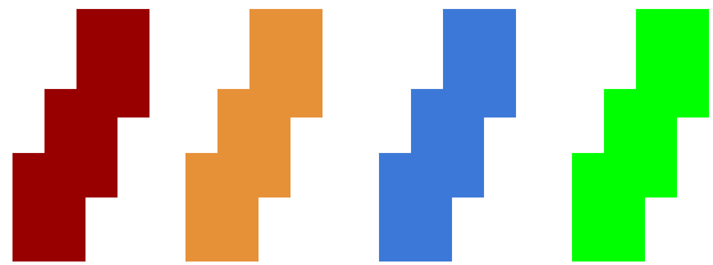
```
input_layer=Input(shape=(4,)) #<--4 inputs/features

output_layer=Dense(1)(input_layer) #<--1 output

model=Model(input_layer,output_layer)

model.compile(optimizer="adam",loss="mse")
```



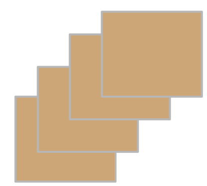
Fitting a model to noise

ML/DL may find you the right hypothesis. Here is a ***hello world*** neural network.

```
input_layer=Input(shape=(4,)) #<--4 inputs/features

output_layer=Dense(1)(input_layer) #<--1 output

model=Model(input_layer,output_layer)

model.compile(optimizer=”adam”,loss=”mse”)
```



```
#Run prediction on several datasets at a time
```



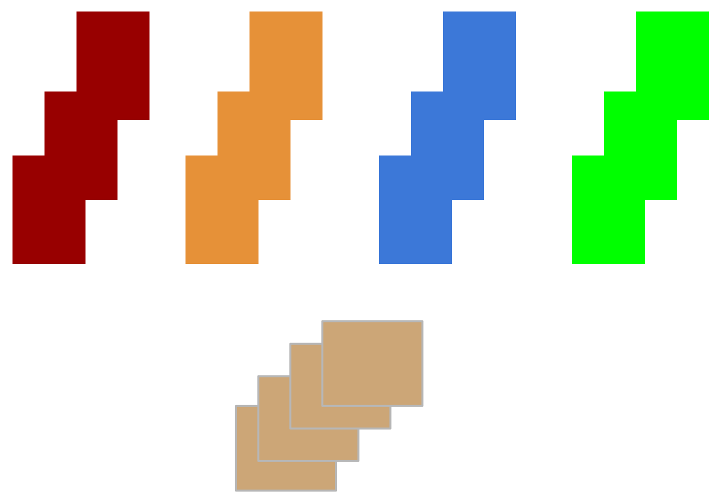
Fitting a model to noise

Shape of input data when passed to a model:

[**how many samples of n features aka batch size, n features**]

Ex. if I want to run a prediction on 1 image of size 20x20, the shape is [1,20,20]

Two audio files of 100 features -> [2,100]



Fitting a model to noise

Differences between the “normal” hypothesis and DL hypothesis

The usual Hypothesis

$y=1/(1+e^{-ax}) + \sin(bx)$

Requires a complete hessian

Deep Learning Hypothesis

Here is a minimalist representation:

$y=f_1(f_2(f_3(f_4(x))))$

where $f_n(x) = 1/(1+e^{-ax})$

Just a sparse Hessian? Guess why? (*Read 2.4*)

Fitting a model to noise

X	**Y**	**Z**	Answer
3	4	5	7
1	2	8	-6
9	3	5	22
3	4	5	7
2	3	4	5

..., guess the operators? X **?** Y **?** Z **=** Answer

Fitting a model to noise

X	Y	Z	Answer
3	4	5	7
1	2	8	-6
9	3	5	22
3	4	5	7
2	3	4	5

..., neural network finds the operators for you.

Exercise Fitting a model to noise

- Standardizing the input enhances the training. Skim through 2.2 to see how data is standardized
- **Ex. 2.2:**
  - Visualize features using iplot
  - Minimize using full hessian method, and plot the prediction, ground truth
- **Ex. 2.4:**
  - Create a custom 4 layer neural network
  - Optimize using backprop
  - Compare the results with full-Hessian approach in 2.2

    What do you infer?

DISCUSSION Fitting a model to noise

- Training vs. testing (70-15-15 rule)