Time Series Prediction With Neural Networks

Scott Crespo
President, Konture Technology Services

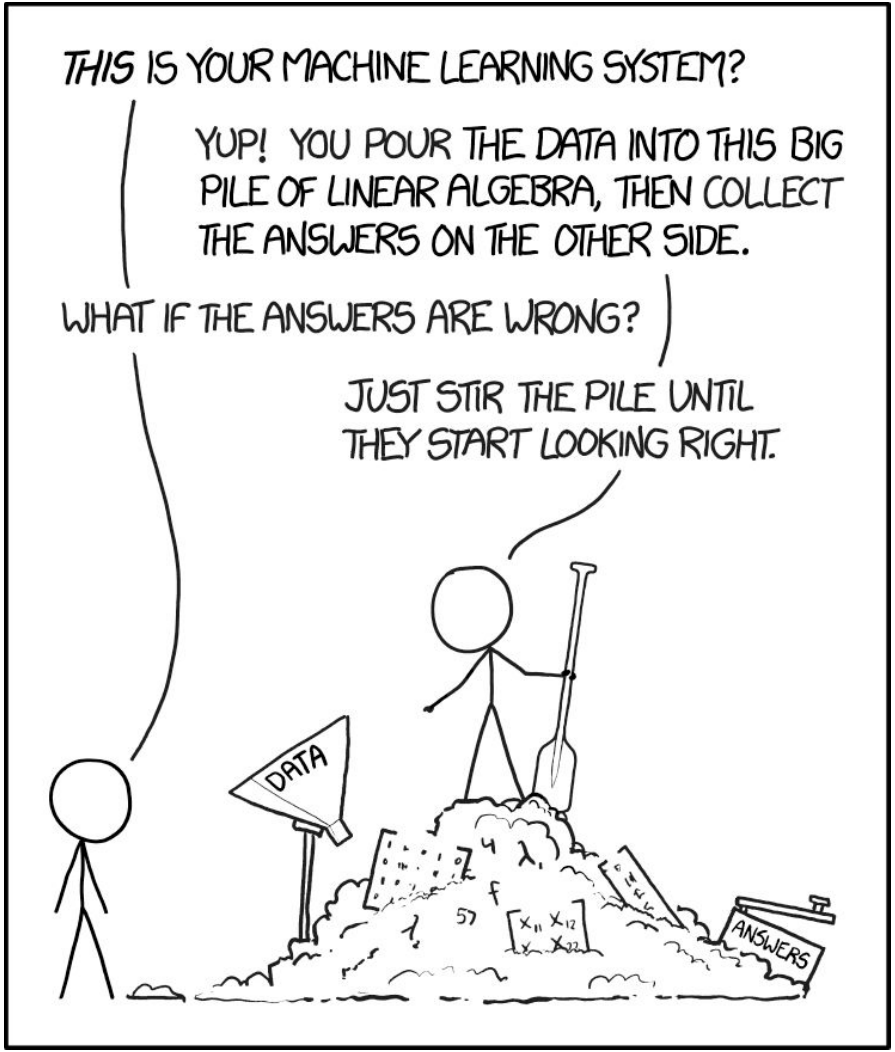
Konture Technology Services

Predictive Autoscaling Platform for AWS (Deprecated)

Technology Services

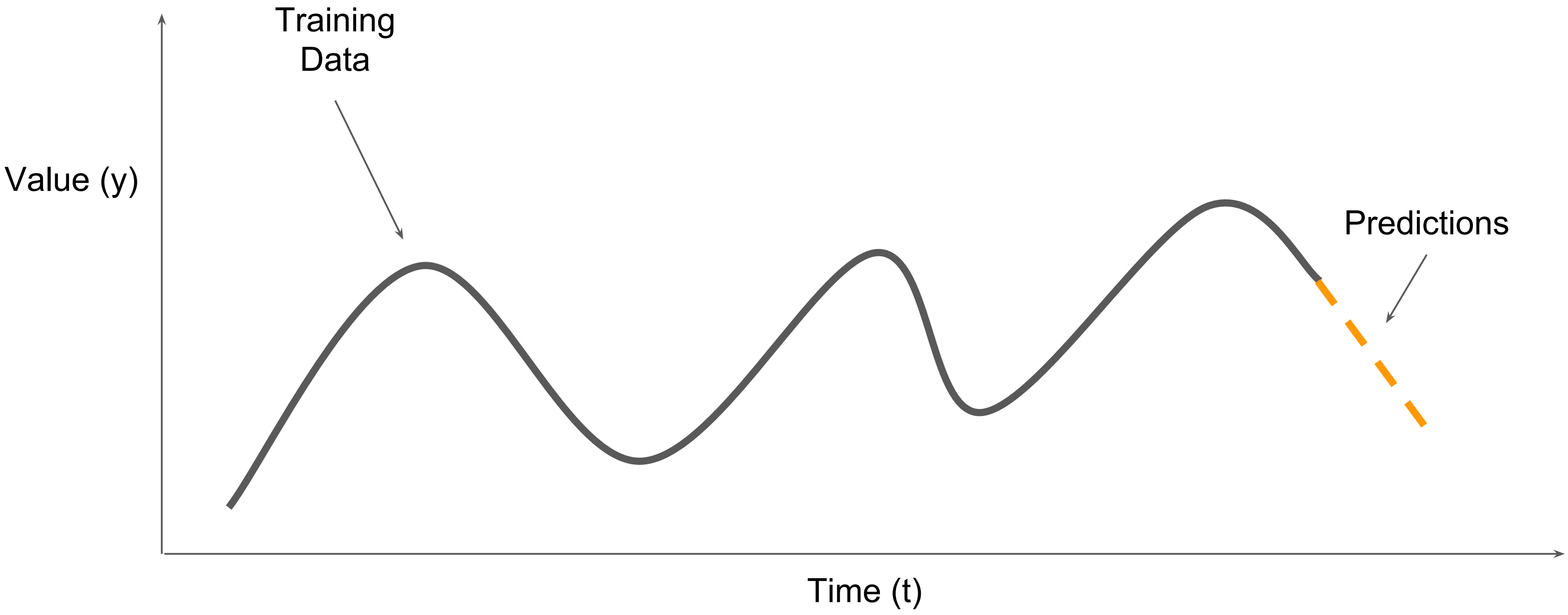
- Cloud Solutions Architecture
- Kubernetes
- DevOps
- Machine Learning
- Application Development

Section 1
Machine Learning Review



Credit: XKCD.com

Time Series Prediction



Some Application Areas

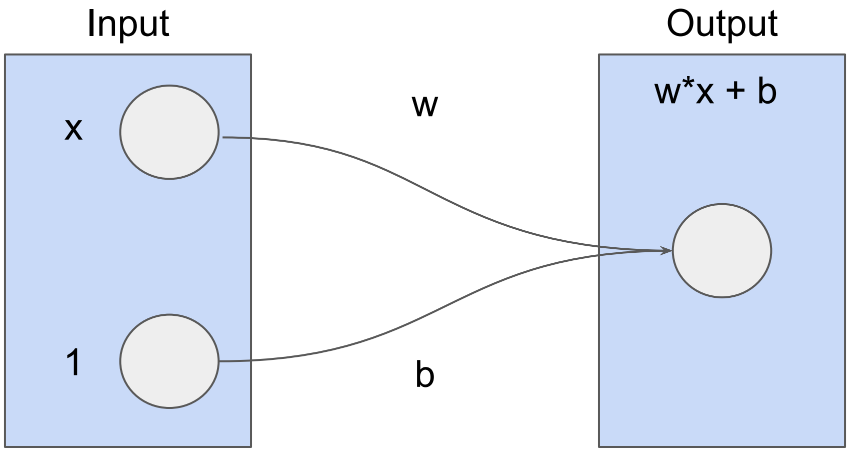
- Business Intelligence
  - Predict Revenue, Costs, Profit, etc
- Energy (Smart Grid)
  - Predict usage spikes, allocate energy efficiently
- High-Frequency Trading
  - Predict stock prices
- Resource Optimization
  - Predict resource requirements, such as server capacity

Artificial Neural Networks

Mimic the way neurons in the brain work

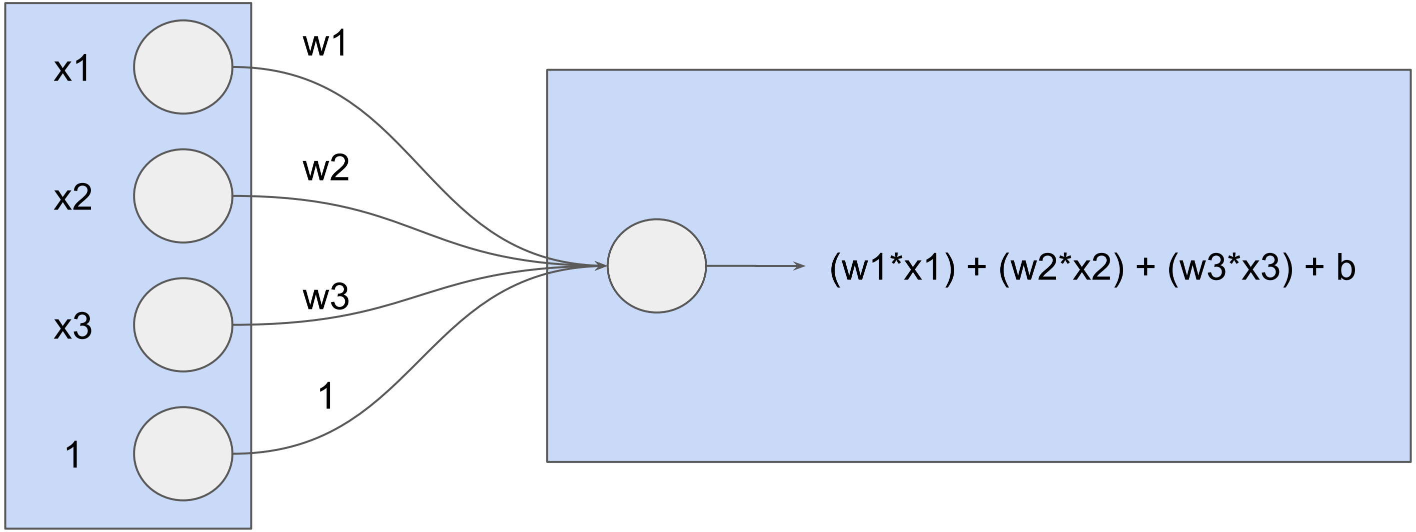
Outputs and weights (parameters) from one neuron form the inputs of another - forming a directed weighted graph

Linear Function
f(x) = w*x + b



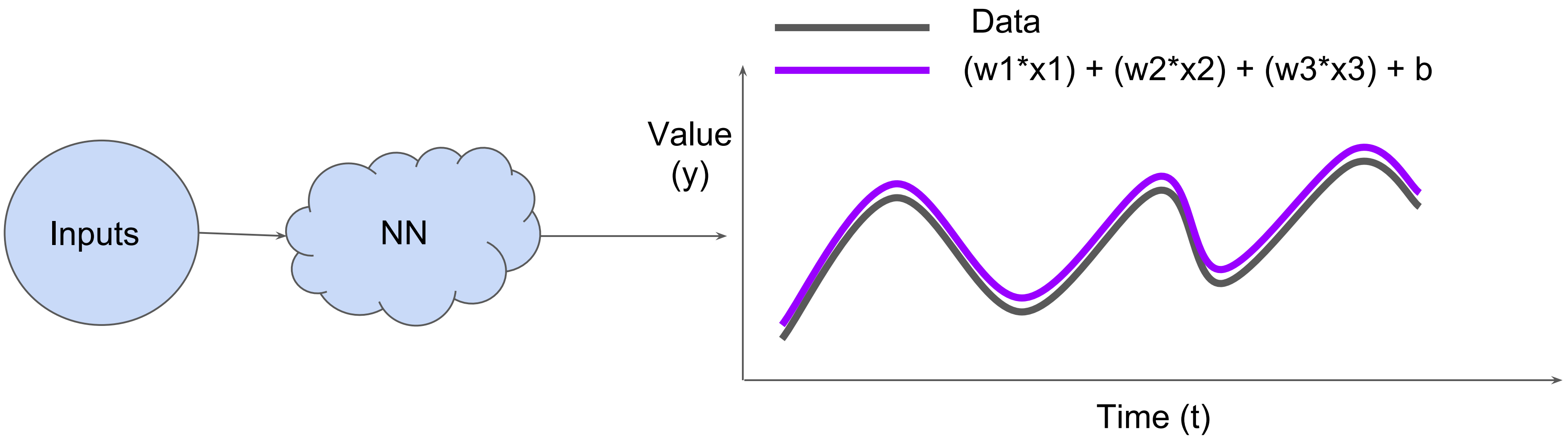
Source: Nishant Shukla (2017), *Machine Learning With Tensorflow.* Manning Publications

More Complex Inputs



Source: Nishant Shukla (2017), *Machine Learning With Tensorflow.* Manning Publications

How Neural Networks Learn Time Series



Supervised Learning

We tell the algorithm what our classes are

Data	Class
	Dog
	Cat

Time Series as Supervised Learning

Raw	
X	Y
1	60
2	70
3	80
4	90
5	100

Supervised	
X	Y
?	60
60	**70**
70	**80**
80	**90**
90	**100**
100	?

Univariate, Single Step Prediction

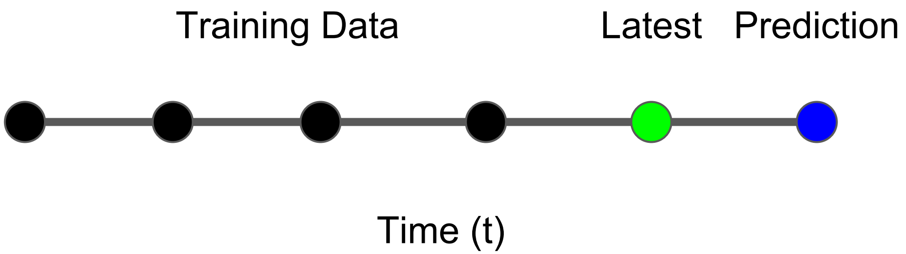
Definition

Univariate, Single Step Prediction

Predict a single, future value based on the previously observed value

Example

Given value of 100, what will the next value be?



Training Concept

Discard →

Training Data →

Supervised Dataset	
X	Y
?	60
60	70
70	80
80	90
90	100

Training Pseudocode

Input training data as supervised set into neural network

X is 3D array of (samples, timesteps, features)

Y is 2D array of(samples, class)

Keras Pseudocode:

```
model.fit(X, Y)
```

Prediction Concept

	Supervised Dataset	
	X	Y
Discard	?	60
Training Data	60	70
	70	80
	80	90
	90	100
Prediction Set	100	110

Prediction Pseudocode

```
def univariate_single_step_prediction():
  X = [100]
  Y = model.predict(X)
  return Y
```

Univariate, Multi-Step Prediction

Definition

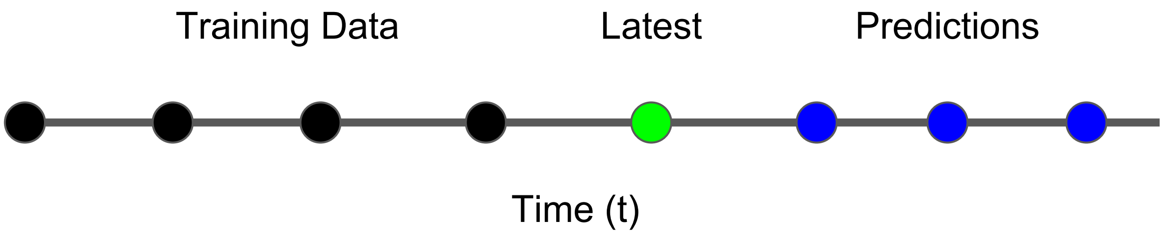
Given a previous value, what will the next *n* values be?

For each forward timestep:

Make a prediction

Take that prediction and feed back into model to emit the next prediction

etc.



Training Concept (Same as Single Step)

Supervised Dataset	
X	Y
?	60
60	70
70	80
80	90
90	100

Discard

Training Data

Multi-Step Prediction Concept

	Supervised	
	X	Y
Discard	?	60
Training Data	60	70
	70	80
	80	90
	90	100
Prediction Set	100	110
	110	120
	120	130

Multi-Step Prediction Pseudocode

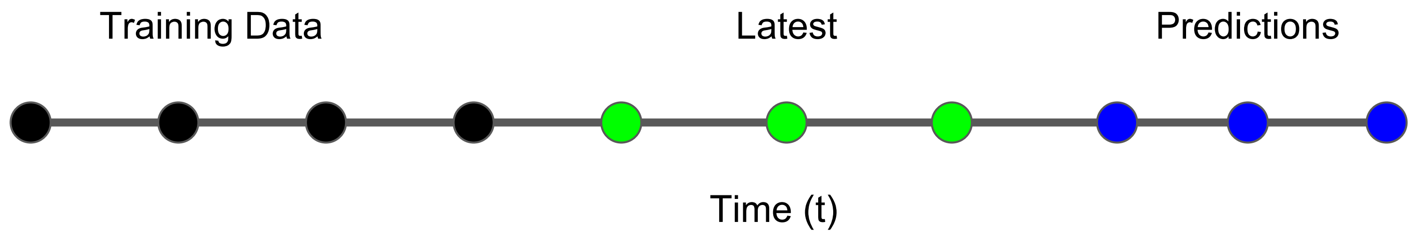
```
def multistep_prediction(timesteps=5):

  i = 0

  X = [100]

  Y = []

  while i < timesteps:

    Y[i] = model.predict(X[i])

    X[i+1] = Y[i]

    i+=1

  return Y
```

Multivariate, Multi-Step Time Series

Multivariate Time Series

Predict a sequence of future values based on *n* previously observed values

Given we observed [60,70,80,90,100], what will the next *m* values be in the sequence?



Multivariate Training Concept

X	Y
[60, 70, 80, 90, 100]	[110]
[70, 80, 90, 100, 110]	[120]
[80, 90, 100, 110, 120]	[130]
[90, 100, 110, 120, 130]	[140]
[100, 110, 120, 130, 140]	[?]

Training

```
raw = [60, 70, 80, 90, 100, 110, 120, 130, 140]

X = [[ 60, 70, 80, 90, 100 ],

        [ 70, 80, 90, 100, 110],

        [ 80, 90, 100, 110, 120 ],

        [ 90, 100, 110, 120, 130 ]]

Y = [ [110],

      [120],

      [130],

      [140]]


model.fit(X, Y)
```

Multivariate, Multi-Step Prediction Concept

p1 = model.predict([60, 70, 80, 90, 100])

p2 = model.predict([70, 80, 90, 100, **p1**])

p3 = model.predict([80, 90, 100. **p1, p2**])

P4 = model.predict([90, 100, **p1, p2, p3**])

Multi-Step, Multivariate Pseudocode

```
def multistep_multivariate_prediction(timesteps=5):

  i = 0

  X = [60,70,80,90,100]

  Y = []

  while i < timesteps:

    Y[i] = model.predict(X[i:i+timesteps])

    X[i+1] = Y[i]

    i+=1

  return Y
```

Multivariate Time Series Prediction
Part 2

Let’s Do Something With Our Timestamps!

Up until now, we’ve discarded our timestamps from the raw data. But there’s a lot of information encoded in the day, month, year, minute, etc of our observations. We shouldn’t let it go to waste!

Decomposing Timestamps into Multiple Attributes

Timestamp = 1543289298

- Year: 2018
- Month: 11
- Day of the Week: 1
- Hour: 22
- Minute: 28

Year	Month	Day (one-hot)	Hour	Minute	Prev. Value

X = [[2018], [11], [1,0,0,0,0,0,0], [22], [28], [60]]

Y = [[70]]

Grand Finale

Multivariate Time Series Prediction with Decomposed Timestamp Attributes

```
X = [[
        [[2018], [11], [1,0,0,0,0,0,0], [22], [28], [60]],
        [[2018], [11], [1,0,0,0,0,0,0], [23], [32], [70]],
        [[2018], [11], [1,0,0,0,0,0,0], [24], [28], [80]],
      ]]
Y = [[90]]
```

Resources

Machinelearningmastery.com

Nishant Shukla (2017), *Machine Learning With Tensorflow.* Manning Publications

Thank You!

Scott Crespo

scott@konture.io
linkedin.com/in/scottcrespo