#TASK #1: UNDERSTAND THE PROBLEM STATEMENT/CASE

- **Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning (DL) have been transforming finance and investing.**
- **"Artificial intelligence is to trading what fire was to the cavemen"!**
- **Electronic trades account for almost 45% of revenues in cash equities trading" U.K. research firm Coalition Report.**
- **AI powered robo-advisers can perform real-time analysis on massive datasets and trade securities at an extremely faster rate compared to human traders.**
- **AI-powered trading could potentially reduce risk and maximize returns.**
- **Check out the list of companies that leverage AI in trading:**
- https://builtin.com/artificial-intelligence/ai-trading-stock-market-tech

Photo Credit: https://www.pxfuel.com/en/free-photo-qaulj

- **In this project, we will train a ridge regression model and deep neural network model to predict future stock prices.**
- **By accurately predicting stock prices, investors can maximize returns and know when to buy/sell securities.**
- **The AI/ML model will be trained using historical stock price data along with the volume of transactions.**
- **We will use a type of neural nets known as Long Short-Term Memory Networks (LSTM).**
- **Disclaimer: Stock prices are volatile and are generally hard to predict. Invest at your own risk.**

#TASK #2: IMPORT DATASETS AND LIBRARIES

```
# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')
```

```
import pandas as pd
import plotly.express as px
from copy import copy
from scipy import stats
```

```
import matplotlib.pyplot as plt
import numpy as np
import plotly.figure_factory as ff
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from tensorflow import keras

# Read stock prices data

# Read the stocks volume data

# Sort the data based on Date

# Sort the volume data based on Date

# Check if Null values exist in stock prices data

# Check if Null values exist in stocks volume data

# Get stock prices dataframe info

# Get stock volume dataframe info
```

MINI CHALLENGE #1:

- **What is the average trading volume for Apple stock?**
- **What is the maximum trading volume for sp500?**
- **Which security is traded the most? comment on your answer**
- **What is the average stock price of the S&P500 over the specified time period?**
- **What is the maximum price of Tesla Stock?**

#TASK #3: PERFORM EXPLORATORY DATA ANALYSIS AND VISUALIZATION

```
# Function to normalize stock prices based on their initial price

# Function to plot interactive plots using Plotly Express

# plot interactive chart for stocks data
```

MINI CHALLENGE #2:

- **Plot the volume dataset for all stocks, list any observations you might see.**
- **Plot the normalized stock prices and volume dataset.**

TASK #4: PREPARE THE DATA BEFORE TRAINING THE AI/ML MODEL

```
# Function to concatenate the date, stock price, and volume in one dataframe

# Function to return the input/output (target) data for AI/ML Model
# Note that our goal is to predict the future stock price
# Target stock price today will be tomorrow's price

# Let's test the functions and get individual stock prices and volumes for
AAPL


# Remove the last row as it will be a null value

# Scale the data



# Create Feature and Target



# Spliting the data this way, since order is important in time-series
# Note that we did not use train test split with it's default settings since
it shuffles the data
```

```
# Define a data plotting function
```

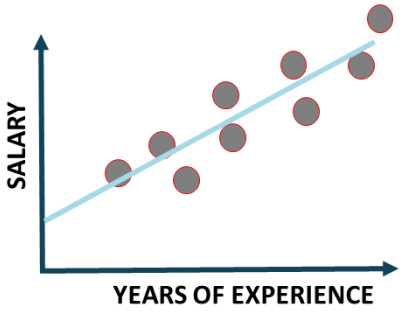
MINI CHALLENGE #3:

- **Test the created pipeline with S&P500 and Amazon datasets**

TASK #5: UNDERSTAND THE THEORY AND INTUITION BEHIND REGRESSION

SIMPLE LINEAR REGRESSION: INTUITION

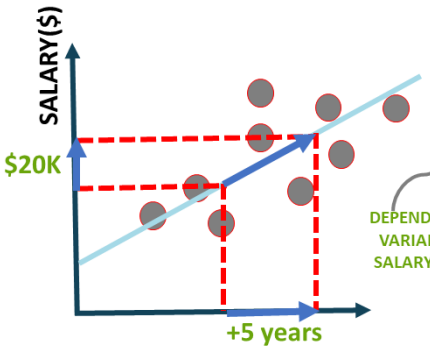
- In simple linear regression, we predict the value of one variable Y based on another variable X.
- X is called the independent variable and Y is called the dependant variable.
- Why simple? Because it examines relationship between two variables only.
- Why linear? when the independent variable increases (or decreases), the dependent variable increases (or decreases) in a linear fashion.



	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891
5	2.9	56642
6	3.0	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218
11	4.0	55794
12	4.0	56957
13	4.1	57081
14	4.5	61111

SIMPLE LINEAR REGRESSION: SOME MATH!

- Goal is to obtain a relationship (model) between employee salary and number of years of experience.



MODEL! (GOAL)

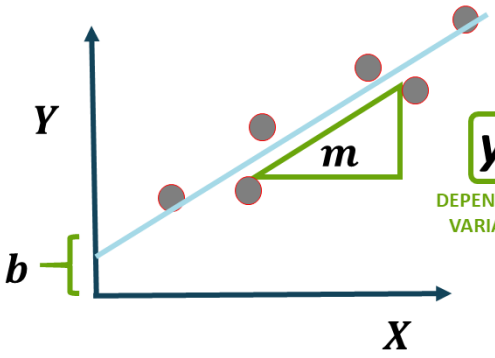
$$y = b + m * x$$

DEPENDANT VARIABLE SALARY ($)

INDEPENDENT VARIABLE NUMBER OF YEARS OF EXPERIENCE

WHAT'S M AND B?

- Once the coefficients m and b are obtained, you have obtained a simple linear regression model!
- This "trained" model can be later used to predict any salary based on the number of years of experience.



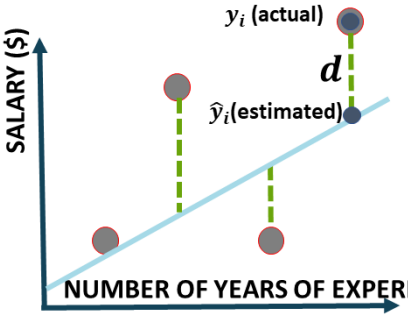
$$y = mX + b$$

DEPENDANT VARIABLE

INDEPENDANT VARIABLE

SIMPLE LINEAR REGRESSION: HOW TO OBTAIN MODEL PARAMETERS? LEAST SUM OF SQUARES

- Least squares fitting is a way to find the best fit curve or line for a set of points.
- The sum of the squares of the offsets (residuals) are used to estimate the best fit curve or line.
- Least squares method is used to obtain the coefficients m and b.



$$d = \hat{y}_i - y_i$$

$$min \sum (\hat{y}_i - y_i)^2$$

MINIMUM (LEAST) SUM OF SQUARES

TASK #6: UNDERSTAND THE CONCEPT OF REGULARIZATION & RIDGE REGRESSION

REGULARIZATION: INTUITION

- Regularization techniques are used to avoid networks overfitting
- Overfitting occurs when the model provide great results on the training data but performs poorly on testing dataset.
- Overfitting occurs when the model learns all the patterns of the training dataset but fails to generalize.
- Overfitted models generally provide high accuracy on training dataset but low accuracy on testing and validation (evaluation) datasets

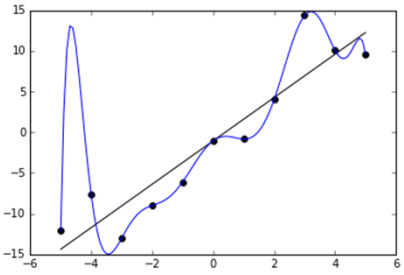
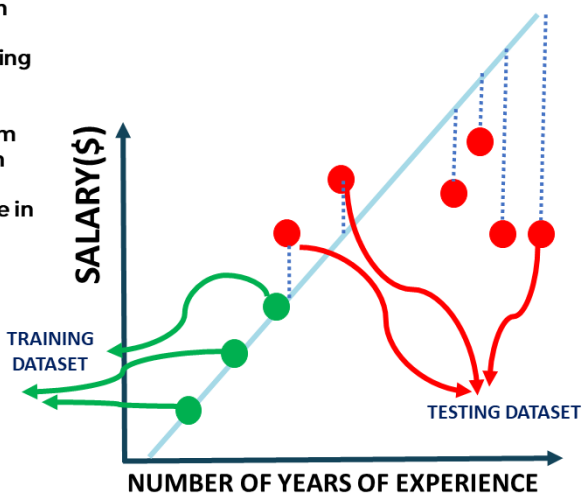
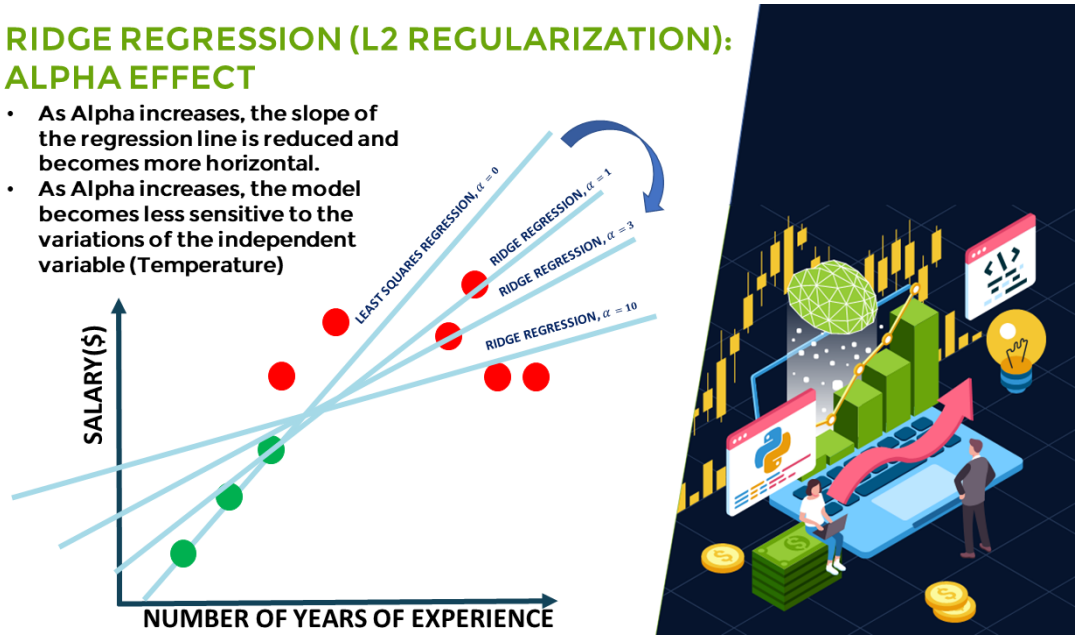


Photo Credit: https://commons.wikimedia.org/wiki/File:Overfitted_Data.png

RIDGE REGRESSION (L2 REGULARIZATION): INTUITION

- Least sum of squares is applied to obtain the best fit line
- Since the line passes through the 3 training dataset points, the sum of squared residuals = 0
- However, for the testing dataset, the sum of residuals is large so the line has a high variance.
- Variance means that there is a difference in fit (or variability) between the training dataset and the testing dataset.
- This regression model is overfitting the training dataset





TASK #7: BUILD AND TRAIN A RIDGE LINEAR REGRESSION MODEL

```
from sklearn.linear_model import Ridge
# Note that Ridge regression performs linear least squares with L2
regularization.
# Create and train the Ridge Linear Regression  Model

# Test the model and calculate its accuracy

# Make Prediction

# Append the predicted values into a list


# Append the close values to the list


# Create a dataframe based on the dates in the individual stock data

# Add the close values to the dataframe

# Add the predicted values to the dataframe

# Plot the results
```

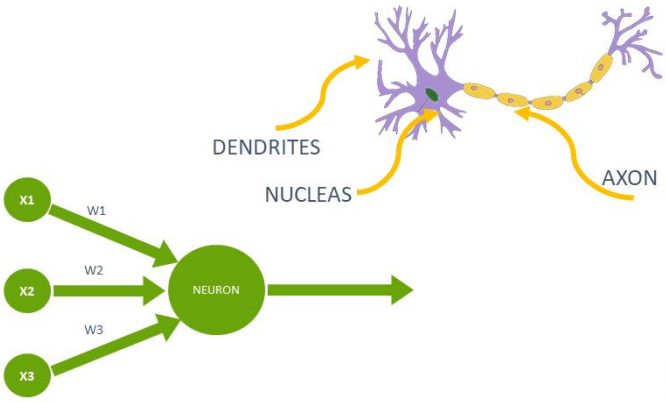
MINI CHALLENGE #4:

- **Experiment with various regularization values for alpha**
- **What is the impact of increasing alpha?**
- **Note: default value for alpha is = 1**

TASK #8: UNDERSTAND THE THEORY AND INTUITION BEHIND NEURAL NETWORKS

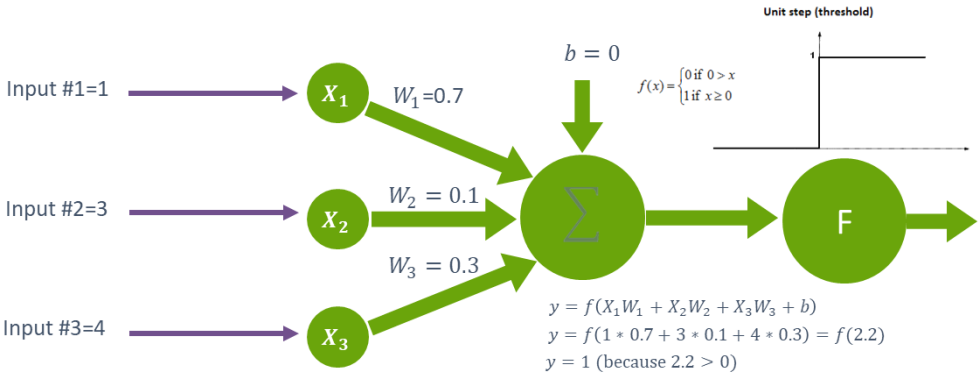
NEURON MATHEMATICAL MODEL

- **Artificial Neural Networks are information processing models that are inspired by the human brain.**
- **The neuron collects signals from input channels named dendrites, processes information in its nucleus, and then generates an output in a long thin branch called axon.**



SINGLE NEURON MODEL IN ACTION!

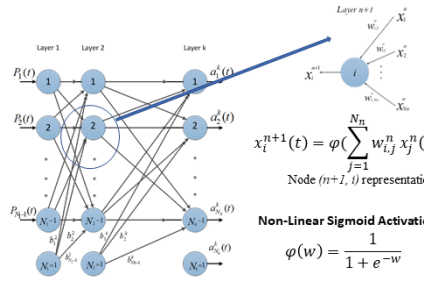
- **Let's assume an activation function of Unit Step.**
- **The activation functions is used to map the input between (0, 1).**



MULTI-LAYER PERCEPTRON NETWORK

- Let's connect multiple of these neurons in a multi-layer fashion.
- The more hidden layers, the more "deep" the network will get.

$$P = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_{N_1} \end{bmatrix}$$



$$x_i^{n+1}(t) = \varphi\left(\sum_{j=1}^{N_n} w_{i,j}^n \, x_j^n(t)\right)$$

Node $(n+1, i)$ representation

Non-Linear Sigmoid Activation function

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

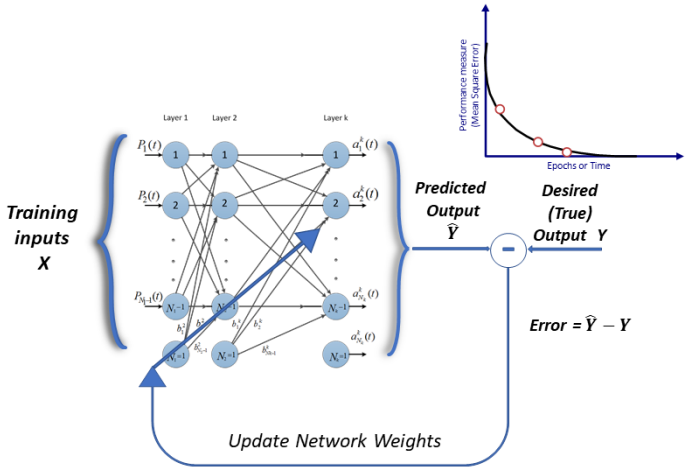
$$\begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1,N_1} \\ W_{21} & W_{22} & & W_{2,N_1} \\ \vdots & & \ddots & \vdots \\ W_{m-1,1} & W_{m-1,2} & \cdots & W_{m-1,N_1} \\ W_{m,1} & W_{m,2} & & W_{m,N_1} \end{bmatrix}$$

m: number of neurons in the hidden layer

N_1 : number of inputs

TASK #9: UNDERSTAND HOW DO ARTFICIAL NEURAL NETWORKS TRAIN

HOW DO ANN TRAIN? PROCESS OVERVIEW

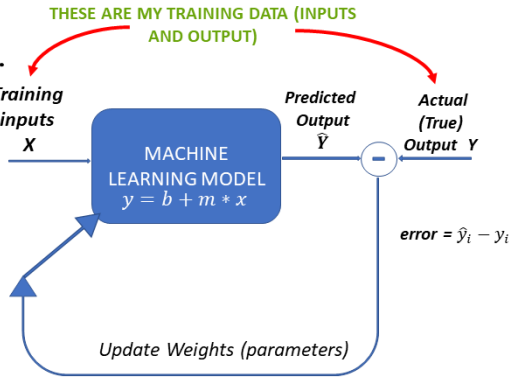


HOW DO ANN TRAIN? GRADIENT DESCENT

- Let's assume that we want to obtain the optimal values for parameters 'm' and 'b'.

$$y = b + m * x$$

GOAL IS TO FIND BEST PARAMETERS



- We need to first formulate a loss function as follows:

$$Cost\ Function\ f(m,b) = \frac{1}{N}\sum_{i=1}^{n}(error)^2 = \frac{1}{N}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

HOW DO ANN TRAIN? GRADIENT DESCENT

$$Loss\ Function\ f(m,b) = \frac{1}{N}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

GRADIENT DESCENT WORKS AS FOLLOWS:

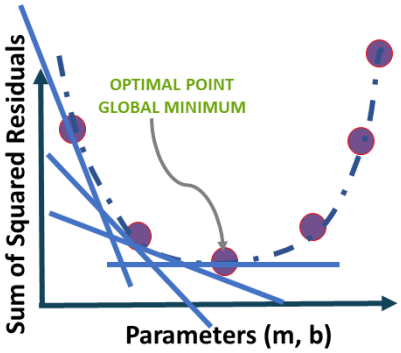
1. Calculate the gradient (derivative) of the Loss function $\frac{\partial loss}{\partial w}$
2. Pick random values for weights (m, b) and substitute
3. Calculate the step size (how much are we going to update the parameters?)

$$Step\ size\ = learning\ rate * gradient = \alpha * \frac{\partial loss}{\partial w}$$

4. Update the parameters and repeat

$$new\ weight\ =\ old\ weight - step\ size$$

$$w_{new} = w_{old} - \alpha * \frac{\partial loss}{\partial w}$$

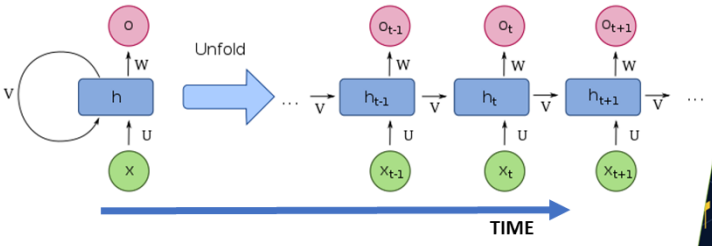


Note: in reality, this graph is 3D and has three axes, one for m, b and sum of squared residuals

TASK #10: UNDERSTAND THE THEORY AND INTUITION BEHIND RECURRENT NEURAL NETWORKS

RNN ARCHITECTURE

- **A RNN contains a temporal loop in which the hidden layer not only gives an output but it feeds itself as well.**
- **An extra dimension is added which is time!**
- **RNN can recall what happened in the previous time stamp so it works great with sequence of text.**



"We'll train RNNs to generate text character by character and ponder the question "how is that even possible?"

Source: The Unreasonable Effectiveness of Recurrent Neural Networks by Andrej Karpathy
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Photo Credit: https://fr.wikipedia.org/wiki/Fichier:Recurrent_neural_network_unfold.svg

LET'S WATCH THIS MOVIE WRITTEN BY AN RNN!

- Let's watch a movie written by AI!
https://arstechnica.com/gaming/2016/06/an-ai-wrote-this-movie-and-its-strangely-moving/
- The movie was written by an LSTM recurrent neural network
- The LSTM network was trained with a corpus of dozens of sci-fi screenplays from movies from the 1980s and 90s.

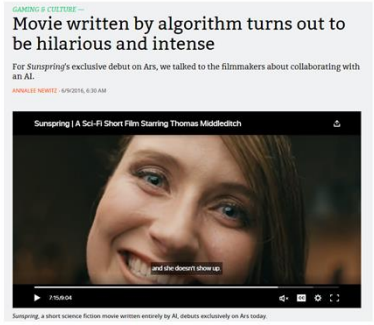


Photo Credit: https://fr.wikipedia.org/wiki/Fichier:Recurrent_neural_network_unfold.svg

TASK #11: UNDERSTAND THE THEORY AND INTUITION BEHIND LONG SHORT TERM MEMORY NETWORKS

VANISHING GRADIENT PROBLEM

- LSTM networks work much better compared to vanilla RNN since they overcome the vanishing gradient problem.
- The error has to propagate through all the previous layers resulting in a vanishing gradient.
- As the gradient goes smaller, the network weights are no longer updated.
- As more layers are added, the gradients of the loss function approaches zero, making the network hard to train.

EACH LAYER DEPENDS ON THE OUTPUT FROM THE PREVIOUS LAYERS, THE "V" IS MULTIPLIED SEVERAL TIMES RESULTING IN VANISHING GRADIENT

0.1 * 0.1 * 0.1 *.....*0.1 =1e-10

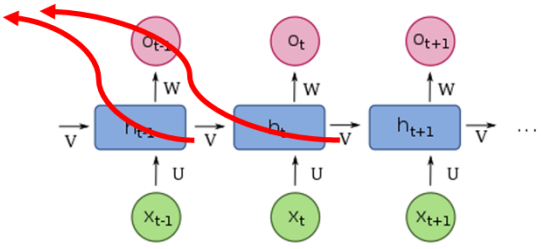
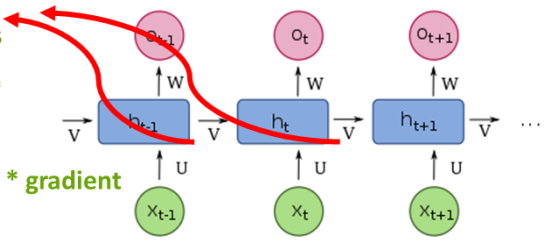


Photo Credit: https://fr.wikipedia.org/wiki/Fichier:Recurrent_neural_network_unfold.svg

VANISHING GRADIENT PROBLEM

- ANN gradients are calculated during backpropagation.
- In backpropagation, we calculate the derivatives of the network by moving from the outermost layer (close to output) back to the initial layers (close to inputs).
- The chain rule is used during this calculation in which the derivatives from the final layers are multiplied by the derivatives from early layers.
- The gradients keeps diminishing exponentially and therefore the weights and biases are no longer being updated.

EACH LAYER DEPENDS ON THE OUTPUT FROM THE PREVIOUS LAYERS, THE "V" IS MULTIPLIED SEVERAL TIMES RESULTING IN VANISHING GRADIENT, (ex: 0.1 * 0.1 * 0.1 *.....*0.1 =1e-10)



New Weight = Old Weight – Learning rate * gradient

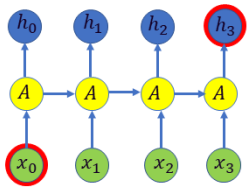
9.09999 = 10.1 – 1 * 0.001

Photo Credit: https://fr.wikipedia.org/wiki/Fichier:Recurrent_neural_network_unfold.svg

LSTM INTUITION

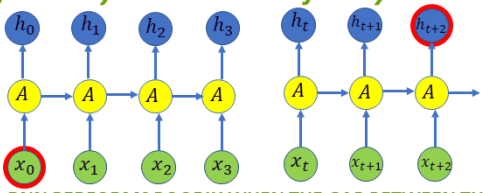
- LSTM networks work better compared to vanilla RNN since they overcome vanishing gradient problem.
- In practice, RNN fail to establish long term dependencies.
- Reference: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

The tree color is "green"



RNN PERFORMS WELL SINCE THE GAP BETWEEN THE PREDICTION "GREEN" AND THE NECESSARY CONTEXT INFORMATION "TREE" IS SMALL

I live in Quebec in Northern Canada.....where I live, the weather is generally "cold" most of the year

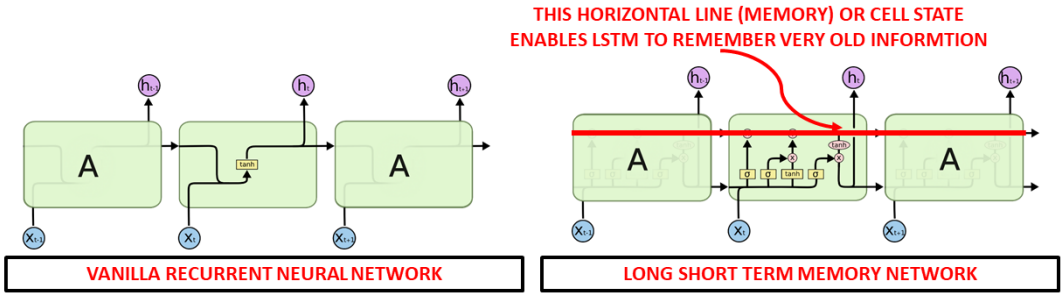


RNN PERFORMS POORLY WHEN THE GAP BETWEEN THE PREDICTION "COLD" AND THE NECESSARY CONTEXT INFORMATION "CANADA" IS LARGE

Reference: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM INTUITION

- **LSTM networks are type of RNN that are designed to remember long term dependencies by default.**
- **LSTM can remember and recall information for a prolonged period of time.**
- **Recall that each line represents a full vector.**



Reference and Photo Credit:
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

TASK #12: TRAIN AN LSTM TIME SERIES MODEL

```
# Let's test the functions and get individual stock prices and volumes for AAPL

# Get the close and volume data as training data (Input)

# Normalize the data

# Create the training and testing data, training data contains present day and previous day values


# Convert the data into array format
```

```
# Split the data

# Reshape the 1D arrays to 3D arrays to feed in the model




# Create the model

# Train the model

# Make prediction
```

MINI CHALLENGE #5:

- **Test the pipeline with at least 3 other stocks**
- **Experiment with various LSTM model parameters (Ex: Use 150 units instead of 50), print out the model summary and retrain the model**

MINI CHALLENGE SOLUTIONS

MINI CHALLENGE #1 SOLUTION:

- **What is the average trading volume for Apple stock?**
- **What is the maximum trading volume for sp500?**
- **Which security is traded the most? comment on your answer**

- **What is the average stock price of the S&P500 over the specified time period?**
- **What is the maximum price of Tesla Stock?**

```
# Get the statistical data for the stocks volume dataframe
# Average trading volume for Apple stock is 2.498238e+06
# Average trading volume for S&P500 is 3.680732e+09

# Why S&P500 is the most traded of all? Check out this article:
# https://www.investopedia.com/articles/personal-finance/022216/put-10000-sp-500-etf-and-wait-20-years.asp

# Text from the article above:
# "The S&P 500 index is a broad-based measure of large corporations traded on U.S. stock markets. Over long periods of time,
# passively holding the index often produces better results than actively trading or picking single stocks.
# Over long-time horizons, the index typically produces better returns than actively managed portfolios."

stock_vol_df.describe()

# Get the statistical data for the prices dataframe
stock_price_df.describe()

# Average price for S&P500 = 2218.749554
# Maximum Tesla Price = 1643.000000
```

MINI CHALLENGE #2 SOLUTION:

- **Plot the normalized stock prices and volume dataset.**

```
# Plot interactive chart for volume data
# Notice that S&P500 trading is orders of magnitude compared to individual stocks
interactive_plot(stock_vol_df, 'Stocks Volume')

# plot interactive chart for normalized stocks prices data
interactive_plot(normalize(stock_price_df), 'Stock Prices')

# Let's normalize the data and re-plot interactive chart for volume data
interactive_plot(normalize(stock_vol_df), 'Normalized Volume')
```

MINI CHALLENGE #3 SOLUTION:

- **Test the pipeline with S&P500 and AMZN datasets instead of AAPL**

```
# Let's test the functions and get individual stock prices and volumes for S&P500
price_volume_df = individual_stock(stock_price_df, stock_vol_df, 'sp500')
price_volume_df

# Let's test the functions and get individual stock prices and volumes for Amazon
```

```
price_volume_df = individual_stock(stock_price_df, stock_vol_df, 'AMZN')
price_volume_df
```

MINI CHALLENGE #4 SOLUTION:

- **Experiment with various regularization value for alpha**
- **What is the impact of increasing alpha?**
- **Note: default value for alpha is = 1**

```
from sklearn.linear_model import Ridge
# Note that Ridge regression performs linear least squares with L2
regularization.
# Create and train the Ridge Linear Regression  Model
regression_model = Ridge(alpha = 2)
regression_model.fit(X_train, y_train)
```

MINI CHALLENGE #5 SOLUTION:

- **Test the pipeline with at least 3 other stocks**
- **Experiment with various LSTM model parameters (Ex: Use 150 units instead of 50), print out the model summary and retrain the model**

```
# Create the model
inputs = keras.layers.Input(shape=(X_train.shape[1], X_train.shape[2]))
x = keras.layers.LSTM(150, return_sequences= True)(inputs)
x = keras.layers.Dropout(0.3)(x)
x = keras.layers.LSTM(150, return_sequences=True)(x)
x = keras.layers.Dropout(0.3)(x)
x = keras.layers.LSTM(150)(x)
outputs = keras.layers.Dense(1, activation='linear')(x)

model = keras.Model(inputs=inputs, outputs=outputs)
model.compile(optimizer='adam', loss="mse")
model.summary()
```