

## Session 14

Monday, March 27, 2023 11:15 PM

**Which of the measures of central tendency will always change if a single value in the data changes?**

Mean  
Median  
Mode

The measure of central tendency that will always change if a single value in the data changes is the mean. The mean is calculated by adding up all the values in a data set and dividing by the number of values. Since a single value affects the sum of the data, it will also affect the mean. On the other hand, the median and mode may or may not change depending on which value changes. If the value that changes is the median or mode itself, then they will obviously change. However, if a value that is not the median or mode changes, they may or may not change depending on the values in the data set.

**If a positively skewed distribution has a median of 50, which of the following statement is true?**

1. Mean > 50 ✓

2. Mean < 50



If a positively skewed distribution has a median of 50, then the mean of the distribution will generally be greater than 50.

In a positively skewed distribution, the tail of the distribution will be longer on the right-hand side, indicating that there are more high values than low values. This means that the mean will be pulled towards the higher values, which will result in a higher mean value than the median.

Therefore, we can conclude that the statement "Mean > 50" is true.

**What is Adaptive Moment Estimation (or Adam), what all it combines?**

Momentum ✓

+ Rmsprop ✓

Adaptive Moment Estimation (Adam) is an optimization algorithm used in machine learning to update the weights of the neural network during the training process. Adam combines two techniques - Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp) to estimate the first and second moments of the gradients and use them to update the parameters.

Adam is designed to combine the benefits of both AdaGrad and RMSProp. AdaGrad adapts the learning rate for each parameter based on the historical gradient information, but it can also decay the learning rate too quickly. RMSProp, on the other hand, uses a moving average of the squared gradients to scale the learning rate, which can adapt the learning rate more smoothly. Adam combines these two approaches to adaptively estimate the learning rate for each parameter while also smoothing out the learning rate adaptation.

In Adam, the first moment of the gradient is estimated as the exponential moving average of the gradient, and the second moment is estimated as the exponential moving average of the squared gradient. These estimates are then used to update the learning rate for each parameter, which is then used to update the parameter values. Overall, Adam is an efficient and effective optimization algorithm that combines the benefits of AdaGrad and RMSProp to adaptively estimate the learning rate and update the parameters of the neural network during training.

A. Create a function in Python that accepts two parameters. They'll both be numbers. The first will be the month as a number, and the second will be the four-digit year. The function should parse the parameters and return True if the month contains a Friday the 13th and False if it doesn't.

(N1, N2)  
(28, 2001)

Here's a Python function that takes in two parameters, month and year, and checks if there is a Friday the 13th in that month:

python

import datetime

```
def has_friday_13th(month, year):
```

```
# Create a datetime object for the 13th of the given month and year
```

```
date = datetime.datetime(year, month, 13)
```

```
# Check if the 13th is a Friday
```

```
return date.weekday() == 4
```

Here's an explanation of how the function works:

We first import the datetime module to work with dates and times.

We define a function called `has_friday_13th` that takes in two parameters, month and year.

We create a datetime object called `date` that represents the 13th of the given month and year.

We use the `weekday()` method of the datetime object to get the day of the week as an integer, where Monday is 0 and Sunday is 6.

We check if the day is 4, which corresponds to Friday.

If the 13th is a Friday, the function returns `True`. Otherwise, it returns `False`.

Here's an example of how you could use the function:

pythonCopy code

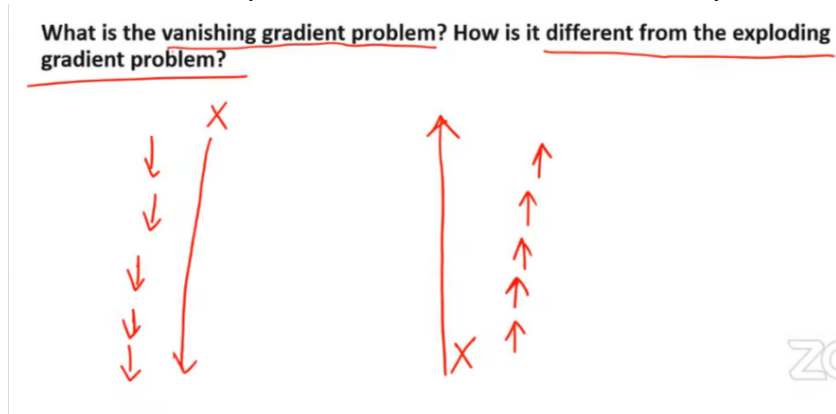
```
>>> has_friday_13th(3, 2023)
```

```
True
```

```
>>> has_friday_13th(4, 2023)
```

```
False
```

In this example, `has_friday_13th(3, 2023)` returns `True` because March 2023 has a Friday the 13th, while `has_friday_13th(4, 2023)` returns `False` because April 2023 does not have a Friday the 13th.



The vanishing gradient problem and the exploding gradient problem are two common issues that can occur during the training of artificial neural networks, particularly in deep neural networks.

The vanishing gradient problem occurs when the gradients used to update the weights during backpropagation become very small as they are propagated backward through the layers of a neural network. This can make it difficult for the network to learn and adjust the weights in the earlier layers, because the gradients become so small that they have little or no impact on the weights. As a result, the earlier layers may not converge and the network may not learn the desired features.

On the other hand, the exploding gradient problem occurs when the gradients become very large as they are propagated backward through the layers of a neural network. This can lead to instabilities during training, as the large gradients can cause the weights to update too

much in one direction, making it difficult for the network to converge to an optimal solution.

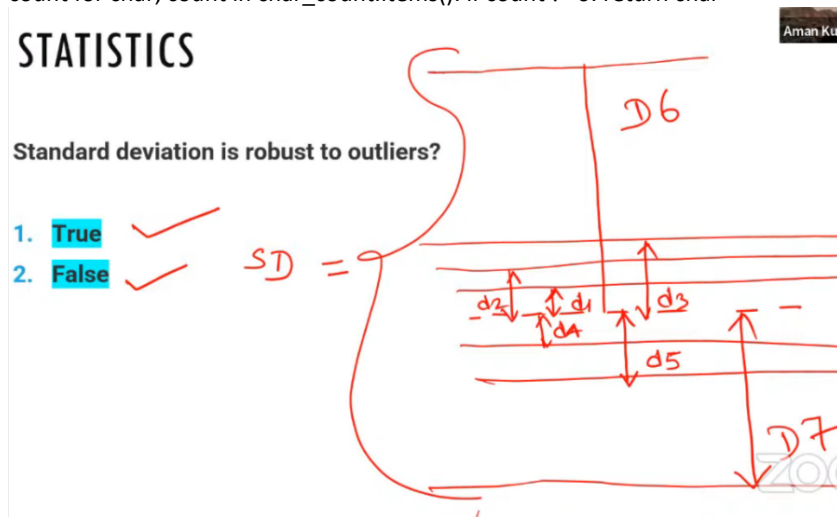
Both problems are caused by the way gradients are propagated backward through a deep neural network during training. The difference is that in the vanishing gradient problem, the gradients become too small, while in the exploding gradient problem, the gradients become too large.

In summary, the vanishing gradient problem and the exploding gradient problem are two common challenges that can occur during the training of deep neural networks. The vanishing gradient problem occurs when the gradients become too small, making it difficult to learn and adjust the weights in earlier layers, while the exploding gradient problem occurs when the gradients become too large, leading to instabilities during training.

Write a function in Python that accepts two string parameters. The first parameter will be a string of characters, and the second parameter will be the same string of characters, but they'll be in a different order and have one extra character. The function should return that extra character.

For example, if the first parameter is "eueieio" and the second is "iieoedue," then the function should return "d."

```
def find_extra_char(s1, s2):
    # Create a dictionary to count the frequency of characters in s1
    char_count = {}
    for char in s1:
        if char in char_count:
            char_count[char] += 1
        else:
            char_count[char] = 1
    # Loop through the characters in s2 and subtract their count in char_count
    for char in s2:
        if char in char_count:
            char_count[char] -= 1
        else:
            return char
    # Loop through the counts in char_count and return the character with a non-zero
    # count for char, count in char_count.items(): if count != 0: return char
```



The statement "standard deviation is robust to outliers" is not entirely true.

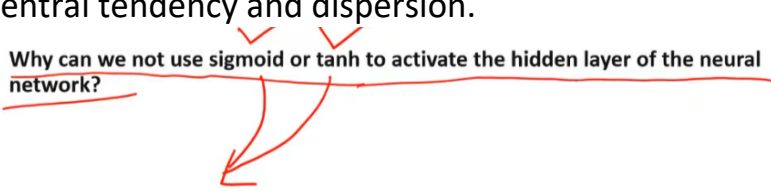
The standard deviation is a statistical measure that indicates how much the data values in a population or a sample differ from the mean value. It is calculated by taking the square root of the variance, which is the average of the squared differences between each data point and the mean.

However, the standard deviation can be affected by outliers, which are extreme values that lie far away from the bulk of the data. If a dataset contains outliers, the standard deviation will be larger than it would be without the outliers. This is because the squared differences between the data points and the mean are magnified by the outliers, which can result in a larger variance and a larger standard deviation.

In contrast, there are other measures of central tendency and dispersion, such as the median and the interquartile range, that are more robust to outliers. The median is the middle value in a dataset, and it is not affected by extreme values in the same way that the mean is. The interquartile range is a measure of dispersion that represents the range between the 25th and 75th percentiles of the data, and it is also less sensitive to outliers than the standard deviation.

Therefore, while the standard deviation is a useful measure of dispersion, it may not be the best choice when dealing with datasets that contain outliers. It is important to consider the nature of the data and the presence of outliers when choosing an appropriate measure of central tendency and dispersion.

Why can we not use sigmoid or tanh to activate the hidden layer of the neural network?



While sigmoid and tanh activation functions were commonly used in the past to activate the hidden layers of neural networks, they have certain limitations that have led to the development of other activation functions that are more commonly used today. Here are a few reasons why sigmoid and tanh are not always preferred for activation of the hidden layers:

**Vanishing Gradient Problem:** Sigmoid and tanh are both saturating activation functions, meaning that their output values get closer and closer to their asymptotic limits as the input values get larger in magnitude. This saturation behavior can cause the gradients to become very small during backpropagation, leading to the vanishing gradient problem. This problem can make it difficult to train deep neural networks with many layers.

**Bias Shift:** Both sigmoid and tanh are biased towards the positive side, meaning that their outputs are always positive or zero. This bias can cause the outputs of the hidden units to be shifted away from zero, which can lead to slower convergence during training.

**Output Range:** Sigmoid has an output range between 0 and 1, while tanh has an output range between -1 and 1. This limited output range can lead to a problem called "saturation", where the output of the activation function is pushed to the extremes, making it difficult for the network to learn.

As a result of these limitations, other activation functions such as ReLU (Rectified Linear Unit), Leaky ReLU, and ELU (Exponential Linear Unit) have become more popular for activation of the hidden layers of neural networks. These functions are non-saturating and allow for faster

training, while also addressing the vanishing gradient problem and the bias shift problem.

You are blindfolded and 10 coins are placed in front of you on the table. You are allowed to touch the coins but can't tell which way up they are by feel. You are told that there are 5 coins head up, and 5 coins tails up but not which ones are which.

Can you make two piles of coins each with the same number of heads up? You can flip the coins any number of times.



Yes, it is possible to make two piles of coins each with the same number of heads up. Here's one way to do it:

Divide the 10 coins into two piles of 5 coins each.

Flip all the coins in one of the piles so that they are all tails up.

Count the number of heads up in each pile.

If the number of heads up in each pile is the same, you're done!

If the number of heads up in each pile is different, take one coin from the pile with fewer heads up and flip it over.

Count the number of heads up in each pile again.

If the number of heads up in each pile is now the same, you're done!

If the number of heads up in each pile is still different, repeat steps 5-7 until you have two piles with the same number of heads up.

The idea behind this approach is to start with one pile in a known state (all tails up) and then adjust the other pile by flipping coins until the two piles have the same number of heads up. By always flipping a coin from the pile with fewer heads up, we ensure that we're making progress towards the goal of having two piles with the same number of heads up.

