

Audio file

[Your Recording 13.wav](#)

Transcript

00:00:01 Speaker 2

In this lesson, we're going to.

00:00:04 Speaker 2

In this lesson, we're going to look at How much math do I need for machine learning?

00:00:12 Speaker 2

So we are going to look at an overview of the math needed to perform machine learning and understand where math is used in the machine learning process. We are going to look at some key techniques and see where they apply.

00:00:28 Speaker 2

But crucially, in this lesson, we want to get a depth of understanding that is appropriate for us to be able to demystify Sagemaker. In other words, how much math do I need to know about model training, and how much math do I need to know in order to prepare my data? This is all about having just enough math to perform the job.

00:00:46 Speaker 2

And we're going to look at specific targeted study. We're going to cover the topics that will help you create and produce a model and host it for inference with reasonable accuracy. So when we think about how much math is needed for machine learning, I know that I struggled when I first started looking at machine learning, that there seemed to be a barrier that was very high. All resources seemed to suggest that I needed to study a lot of mathematics and a lot of concepts in order to be remotely good at starting a machine learning. And I'm here to say that that's not true.

00:01:15 Speaker 2

No, I cannot sugarcoat this. If you are going to be working with machine learning, the more mathematical skills you have, the better at your job you are going to be. However, that doesn't mean that when starting out learning Sagemaker and machine learning, you need to learn it all upfront. You can learn just enough math to be dangerous, just enough math to be able to develop a model, test the model, and get some useful predictions.

00:01:37 Speaker 2

So what we're trying to say is that that barrier is not as high as you think it is. Don't need a year worth of mathematical skills before we get started. So we need to think about the different areas in machine learning where mathematics is employed.

00:01:50 Speaker 2

Firstly, there is heavy mathematics used during model training in the development of a new model. Now the more you understand about the topics of linear algebra, probabilities, statistics, differential calculus and numerical methods such as gradient descent, then the better data scientist you will be. And again to look at some other training resources out there, it might suggest that you need to learn all of them first. I would say you don't have to learn them first, you just need to be aware of how they fit and then later you can fill in your knowledge and develop your knowledge to get good in each of these areas and then.

00:02:20 Speaker 2

Will be able to build much better models through that knowledge.

00:02:23 Speaker 2

The other area where we need some awareness of mathematical concepts is the mathematics that is used for data preparation. So the mathematics used for data preparation would be statistical methods, which might mean something as simple as taking the average, finding the mean, or finding the median, the middle value, and working out all the values above or below that point. Understanding the basics of linear algebra and understanding encoding techniques. If we've got a grounding or a foundation in those areas and you apply them to your data set, you are going to get a better quality of model. Of course, if we know more about.

00:02:53 Speaker 2

Model development and probability and stats. We can take it even further, but this will be enough to get you started. While we study machine learning, there will come a point where we decide whether or not you are going to specialize in machine learning or be a General practitioner who does machine learning in addition with other activities.

00:03:08 Speaker 2

If we consider the path that we're going to go down as maybe the data scientist path, and we want to dedicate ourselves to machine learning, then as I mentioned before, you are going to need strong mathematical foundation, particularly linear algebra, statistical methods, and probability math.

00:03:21 Speaker 2

This will ultimately enable you to build better models because you are able to understand what is happening exactly during the training process and therefore be able to tune what is happening and understand that tuning process in great detail. You will be able to create better models through that knowledge.

00:03:37 Speaker 2

However, if you are a General practitioner, you are maybe going to be doing some data engineering, you're going to be doing some coding, you're going to be doing some ML. Then maybe we see ourselves more as a Sagemaker user who has the ability to create models, maybe for proof of concept, to determine if you can generate a model of good enough quality to solve a business problem or to determine if we need to engage a data scientist to take that model further in its development. So if I'm going down that path, then really we can say let's use the built in algorithms that are in Sagemaker like Xgboost, like linear learner, like LGBM.

00:04:07 Speaker 2

And have minimal need for deep mathematical knowledge. But if we've got enough mathematical knowledge to prepare our data well, then we might get good enough accuracy out of the models that we build.

00:04:18 Speaker 2

So in the remaining part of this lesson, I want to place a greater emphasis on the mathematical techniques that we can apply to our data set so that data can be in a shape that the training process of our model can make best use of that data. And we'll see why that's important in a few slides time.

00:04:35 Speaker 2

We saw in a previous lesson how when we are training our model that the training process is actually doing some maths behind the scenes. What is it doing? Well, we know we have input features in our tabular data set and we know that for each feature it will have a different contribution to the target value. And we saw that we could have, for example, with house prices, the different input features might be number of bedrooms, number of bathrooms, the square footage which we would refer to like $X_1X_2X_3$ as the input features. But each feature will have a coefficient awaiting what we call W_1 for feature one.

00:05:05 Speaker 2

W_2 for feature 2 and so on. Depending how many features you have, you have a greater emphasis of one particular feature then it be a greater weighting against that feature.

00:05:13 Speaker 2

So what's really happening during that training process is that algorithm is trying to adjust those weightings so that the function is produced kind of like our line of best fit when we saw our two-dimensional example or our surface of best fit when we saw three-dimensional example. But this is a multi dimensional space. So we're thinking about like a multi dimensional surface of best fit and how does the function get created for that. And we have that kind of formula and we are able to adjust the weights during that process.

00:05:39 Speaker 2

How does it know whether it's doing a good job or not? Well, remember during training your data set includes the input features and the desired target value. In other words, I might train my house price prediction model with a set of existing house price data where I know what the house sold for. I have the target value, so I could present data to the model based on the input features and say, well, how

close did it come to what the actual output was? And that will give me an idea of how close that was. If you remember, when we talked about linear regression, we said that in our simple 2 dimensional.

00:06:09 Speaker 2

Hope that when we're drawing a line of best fit, we could measure the distance between the actual data points and the line. We said that was the residual. We could sum up the square of those residuals, and that was our loss function. How well, how close were we to the actual desired output?

00:06:24 Speaker 2

So during training, we need to be aware of the math that is happening. We need to know that the algorithm will be using a method such as stochastic gradient descent. And again, if we were going deeper into the math here, we would go into exactly how that works. But it's using a technique of trying to determine what the appropriate weights should be for each of the input features. And it's trying to do that by reducing the loss function, OK. It's trying to get as close as possible to what the desired target value should be. Because you supplied that ground truth value in the training data set, you've got the answer right there, and we can determine.

00:06:54 Speaker 2

Effective our model is so far. Remember, as well, training is an iterative process, so it will be doing this over and over in these adjustments until it can get as close as it can to minimizing loss. For the remaining part of this lesson, I'd like to concentrate on the mathematical techniques that we can apply to our source data set. This is about trying to get our data into the best form that our algorithm can make best use of that data.

00:07:16 Speaker 2

So we have our data that we want to transform, and the techniques we want to think about are encoding techniques so that we can turn categorical data into numerical data, something called outlier management and scaling techniques. And we're going to look at 3 separate scaling techniques. So it's going to be up to you as the data scientist to decide which, if any, of these techniques you will apply to your data. You might apply them all, you might apply none, but this will become with experience based on the problems you're solving, and it will also be a function of what algorithm that you choose. Some algorithms are more sensitive to.

00:07:46 Speaker 2

Data-scale than others, for example.

00:07:49 Speaker 2

So we'll talk more about this as we go through them, but this will transform our data into a shape that the algorithm can make best use of. And I feel that we can gain a lot more productivity from being a beginner with Sagemaker and producing useful models by concentrating our maths effort in this space rather than going deep into understanding how gradient descent actually works. To help me apply these transformations, I'm going to make use of a number of Python libraries.

00:08:13 Speaker 2

Now the libraries I want to look at are pandas and scikit learn. Now existing data scientists likely have come across these packages before, but if we are new to the realm of data science and Sagemaker, this might be the first time you've heard of these. But I can guarantee that most machine learning projects you will work on will make use of one or both of these packages. If we look at pandas in a little bit more detail, what we can see here is this is how to import pandas with import statement. It is a convention to import pandas as PD for short. Now you don't have to, but it makes sense that we all do this as convention and makes.

00:08:42 Speaker 2

Us be able to read one another's code. Now by onboarding pandas I get access to a data construct called a data frame and a data frame makes it very easy to manipulate tabular data. Here I've got a CSV file, a comma separated value file and what I'm doing is I'm using the read CSV method to read that file into a data frame that I'm calling data. Now that I've got a data frame called data, I can use the pandas methods to perform actions upon it and transform it in any way I want. Now initially here I'm using the head method just to return the first few rows.

00:09:12 Speaker 2

But what I really want to do is manipulate that data. Now you can see in the next few lines here what I'm doing is saying missing values in each column. Ah, I'm doing an is null method against my data frame and then counting up how many values are missing. OK, so there is a null method on the data frame. It's going to help me identify that. I certainly do want to know if I have missing data. Now, if I did have missing data, maybe I would want to impute values for it. In other words, rather than throw that through other data away, why don't I invent or synthesize new data based on the average for that feature?

00:09:42 Speaker 2

That might be better than no data at all for that particular feature. So here I'm using the fill NA method on the data frame for the specific column that has the missing data. Then looks like I'm synthesizing a new column, so I'm adding another column to my data set. I'm adding another feature, and in this case here it's just taking an existing feature and its numeric value multiplying by two.

00:09:59 Speaker 2

Now of course you can do whatever arithmetic operation makes sense to you, but you can see that the ease at which I can add columns or take away columns or synthesize new data is very straightforward. These data frames and the methods supported against the data frames are there to empower user data scientist to do things quickly and easily without needing to write 20 lines of Python just to do 1 operation from a data scientist perspective.

00:10:21 Speaker 2

So we often find that data scientists will be using pandas for cleaning, for organizing, and generally transforming their data set into being a usable data set for training.

00:10:29 Speaker 2

The other package I want to talk about is scikit learn. You'll sometimes see this referred to as sklearn, but that's just how we type it in code. But if you say scikit learn or sklearn, we are talking about the same thing. Now when I started working in a large enterprise with data scientists, they were all working locally on their laptops and they were just using pandas with scikit learn and they were able to do complete model development just with those utilities. So this is a very powerful library. Now in the code example here, I can see that from SK learn, I'm importing the pre processing sublibrary here and specifically I'm just pulling in one.

00:11:00 Speaker 2

Feature is standard scaler, so this is a tool that's going to allow me to scale my data. And don't worry, we're going to talk about scaling in a few slides time. For now, I just want to concentrate on the fact that SK Learn is this complete package that helps us perform activities for machine learning.

00:11:14 Speaker 2

There are so many capabilities built in the SK learn for scaling, normalization, regularization, and model training itself. I can see here when I've created my scalar, I can simply apply the fit transform method to my scalar, passing in the value of my data. In this case here I've got a multidimensional list, but I could equally use a data frame, a pandas data frame that I'd obtained from CSV. But the key thing here is that this is about being able to do data preparation techniques like scaling, like standardization, like encoding.

00:11:43 Speaker 2

And not having to write all of that code yourself, but use proven methods that exist in these libraries that we can just call and then concentrate on deriving value from that feature rather than trying to write Python code ourselves to do it.

00:11:55 Speaker 2

Now let's turn our attention to outliers.

00:11:58 Speaker 2

Now, what is an outlier? Well, we define that as an extreme value that can deviate significantly from the rest of the data. Hmm, OK, I think we should be able to easily identify that. Here is a very simple data set, 257101530 Oh, 8953. OK, in that example, it seems pretty obvious which numeric value is the outlier. We can easily look at that with our eyes and see quite clearly that that is our outlier. But why is that important?

00:12:23 Speaker 2

It's important because when we start to apply statistical techniques to it, like calculating the mean.

00:12:28 Speaker 2

Just another word for average, that if we sum up all of the values here to get the mean, dividing it by the number of values in that list, OK, we're saying that the average value for that data set is 1288. Is it though? Is that really representative of the data set? In other words, outliers can skew your statistical properties of your data.

00:12:46 Speaker 2

If we got rid of the outlier altogether and now added up the items 2571015 and 30 and divided by 6, now there's only 6 elements in the list. If we've discarded the outlier, we get a more meaningful mean value of 11.5. But we discarded data, and we generally don't like discarding data.

00:13:03 Speaker 2

So we think it's a good idea to think about how we might handle outliers in our data set.

00:13:08 Speaker 2

So if we find that we have an outlier, here's a data set for car sales data and it looks like in 2019, it does look like we have an outlier here. But whenever we think we have outliers, we do need to go through a process to determine if this maybe a data entry issue, In other words, the data is not valid in the first place.

00:13:24 Speaker 2

Or maybe the data has been corrupted. Maybe if we go back to the golden source of our data, maybe the data came through a data extraction pipeline and at some point it got corrupted along the way.

00:13:33 Speaker 2

But of course this data might be valid, it's just as an extreme value in comparison with all the rest. But again, even if it is an outlier, and even if it is valid, why should I think about handling it?

00:13:44 Speaker 2

So if our data is valid here, we have verified that our data is valid, it's just an extreme outlier. What can we do?

00:13:52 Speaker 2

One approach we could take is to cap an upper threshold value to say, I'm not going to include any value greater than this, but instead substitute it for an upper threshold value. Now looking at this data set, I might say, well my next highest value is 30, so I could replace the 8953 value in this data set just with another instance of 30. And again you could determine what that threshold value would be. So that's one possible technique I could use.

00:14:17 Speaker 2

There are other techniques I could use. For example, I could use logarithm or square root, or simply just clipping that value altogether. These are all valid techniques for handling the outlier. But there's one technique which maybe gets more use than the others, and that is when we use something

called the interquartile range. And this is going to use the median value, the middle value of your range of values, and it will use that to divide up the data set into what we call quartiles.

00:14:41 Speaker 2

Let's take a look in a little bit more detail. But that IQR method.

00:14:45 Speaker 2

Interquartile range is interesting because we can take our data set and find the middle value. And when we find the middle value, we know 50% of data points are above it and 50% of data points are below it. That's our middle value, our median. And it's not affected by the actual values themselves because it's not the mean, it's the media, it's the middle value now, but you know further than that and say, OK, well, in the lower 50% of values, what's the median value there? What's the middle value of the lower 50%? And that gives you your first quartile. So you're dividing your data set up into 25%.

00:15:15 Speaker 2

Of data points are below this point or 50% of the data points are below Q2 medium and 75% of the data points would be below Q3 point. Why do we get Q3? Because the upper 50% of data points would also have a median value and that would be the Q3 value.

00:15:29 Speaker 2

So we call this range between Q1 and Q3. We call this the interquartile range. So if you know your values for Q3 and Q1, you can calculate what that number would be. Now that on its own won't be enough, but it does certainly define where 50% of your data points fall.

00:15:45 Speaker 2

So with interquartile range, we have our data set and we have it divided into four quartiles where we know that at quartile 125% of the data points will be below that value.

00:15:55 Speaker 2

A quartile 375% of the data points will be below that value.

00:15:59 Speaker 2

How do we know that? Because remember this is based on median, the middle value of a sorted list. Ah, OK. We got the middle value then we knew the lower 50% values and the upper 50% of values. And we got the Q1 and Q3 by taking the median of those 2, upper and lower bound 50% ranges. So that means between Q3 and Q1 we have the interquartile range, the middle 50%. So we can calculate that simply by subtracting Q1 from Q3 and you will literally get a number and that's your interquartile range.

00:16:28 Speaker 2

So the IQR is showing you the spread of the metadata and we're going to use that to help us identify what values are technically outliers.

00:16:37 Speaker 2

So to detect outliers, we're going to use IQR. Now a common industry standard way of using the IQR to detect outliers is to use a little formula for calculating a lower and an upper threshold. Let's take a look.

00:16:51 Speaker 2

For lower bound values, in other words an outlier that is far below the normal range of the rest of the data. And that would be if you take the value of Q1 and you subtract from it at 1.5 times multiplier of the IQR value and any value that falls below that number, we would class as an outlier.

00:17:07 Speaker 2

And we would do something similar for defining an upper threshold. So for the upper threshold, we would say the Q3 value plus a 1.5 times multiplier of the IQR. That will give me a number. Any number that is above that number would be an outlier and then subject to your handling of that outlier, which might be winsorization or might be simply to discard the data, whichever technique you choose.

00:17:28 Speaker 2

Let's see that in action.

00:17:30 Speaker 2

Here we have a slightly modified data set. You can see now that the data set is the same except for the upper value now is 90. So I look at that data set and think, do I have any outliers? Well, let's apply the math that we've learned.

00:17:40 Speaker 2

So if we look, how can we calculate our Q2 and Q3 values, Q2 would be the median value, the middle value of my sorted data. So 10 is the middle value, OK. That means in my lower 50% of values, I've got 2-5 and seven and in the upper I've got 1530 and 90. So that means again to work out my Q1 and my Q3 again, I just use those middle values. So Q1 is the middle of two, five and seven and Q3 would be the middle of 1530 and 90.

00:18:05 Speaker 2

So I have my numeric values for Q1Q2 and Q3.

00:18:09 Speaker 2

So that means I can calculate my interquartile range. I simply take away Q1 from Q3. So 5 from 30 = 25. OK, but remember, it's all about using the IQR to define an upper and lower threshold. So let's do that. Let's apply that new arithmetic formula we learned.

00:18:25 Speaker 2

So for the upper and lower bound thresholds we do this. For lower bound we take Q1 should be 5 and subtract from that at 1.5 multiplier of the IQR value which we have calculated to be 25. So we can see there that be $5 - 1.5 * 25$ which is $5 - 37.5$ giving you a lower bound of 32.5.

00:18:43 Speaker 2

In other words, when I look at that data now, do I have any values below -32.5? No. OK, so I have no outliers at the lower bound. OK. What about the upper bound? Well, for the upper bound calculation, we say that we take the Q3 value that's 30, and we add to it 1.5 multiplier times the IQR value. So that would be $1.5 * 25$. Added on to 30 for Q3 gives us $30 + 37.5$, giving us a total of 67.5 for our upper bound.

00:19:08 Speaker 2

So in this example using the IQR outlier detection method.

00:19:13 Speaker 2

Then we could say actually 90 is an outlier because it is above the upper bound threshold that we've calculated using the IQR.

00:19:21 Speaker 2

So IQR is kind of interesting, right? And the great thing about this is that we can use this in code.

00:19:27 Speaker 2

Here's a code example. Notice that importing pandas is PD Great. That will give us a data frame that we can work with. OK, we're also importing numpy. They're numpy is a very useful additional library that we sometimes need if we need to perform some, maybe more detailed math functions against some numeric data in pandas. I can see here we've also imported looks like scalars as well.

00:19:48 Speaker 2

But I want to jump down to the code section on IQR and look at that. It's saying Q1 equals my data frame of the particular feature that I'm working with. Remember, this will be feature by feature. And I'm saying quantile .25 lows 25% of values, Q3 for that same data frame feature quantile .75. So 75% of values are below this point. IQR is just a straight arithmetic operation as we saw on the lower bound and upper bound thresholds are calculated just as we did on the slide a moment ago. But to actually enforce this here, we're choosing to do a cap and we're choosing to use.

00:20:19 Speaker 2

And its clip function to clip the values at the lower and upper bound. So we can see in that feature cap feature of the data frame. This is where we're going to show that feature after it's been clipped based on the IQR method. Very, very simple code, but what's going on under the hood is really quite clever from a statistical perspective.

Agenda

- 01 Overview of Math in ML – Understanding its role in the ML process
- 02 Key Techniques – Essential math concepts for ML
- 03 Depth of Understanding – Where and how much math is needed
- 04 Focus Areas – Targeted study for SageMaker proficiency

© Copyright KodeKloud



Which Math for Which Purpose?



© Copyright KodeKloud



Which Math for Which Purpose?

Math for Model development and optimization

Linear Algebra

Probabilities

Statistics

Calculus
(e.g., Differentiation)

Numerical Methods
(e.g., Gradient Descent)

Math for Data preparation

Statistics

Linear Algebra

Encoding



How Much Math Do You Need for ML?



Data
Scientist
Path

SageMaker
User Path



Needs linear algebra, statistics,
and probability

Requires deeper ML math
knowledge

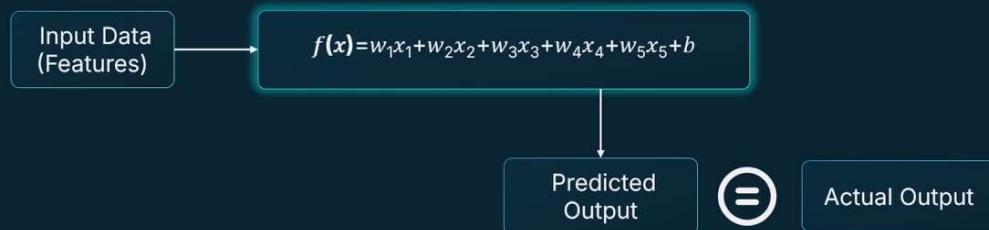
Can use built-in algorithms

Minimal math needed

Focus on data preparation and
model deployment



How Much Math Do You Need for ML?



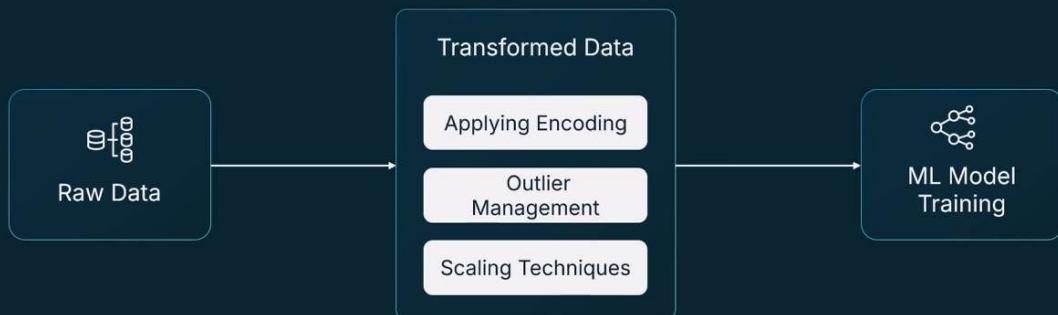
1 | Adjust weights and bias

2 | Repeat until loss is minimized

Copyright KodeCloud 10:15

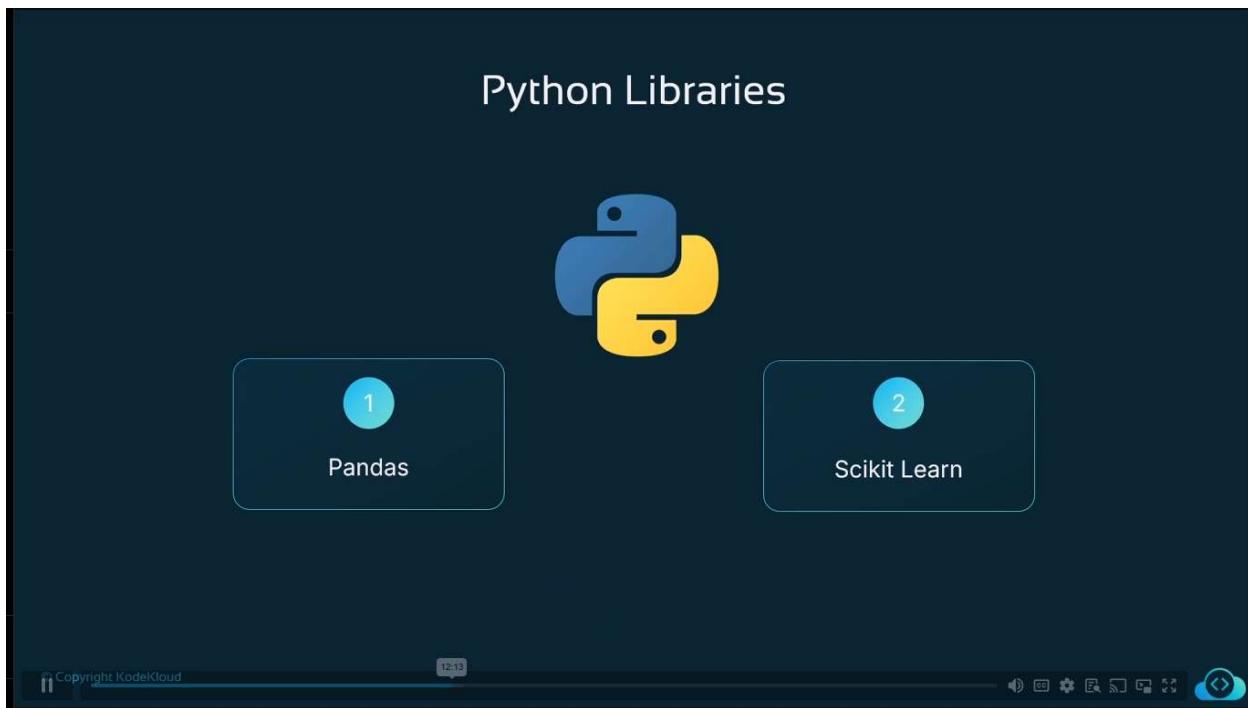


Data Preparation Math



Copyright KodeCloud 11:30





A detailed slide about Pandas. It features a title "Python Libraries – Pandas" and three bullet points: "Python library for data manipulation", "Works with structured data (DataFrames)", and "Used for cleaning, organizing, and analyzing data before ML". To the right is a code editor window showing Python code for data manipulation, with a status bar indicating "15:17".

Python Libraries – Scikit Learn

Machine Learning library in Python

Tool for building, training, and evaluating models

Supports common data preparation techniques

```
scikit-learn.py

from sklearn.preprocessing import StandardScaler

# Sample data: two features (e.g., height and weight)
data = [[1.8, 75], [1.6, 60], [1.7, 68], [1.5, 50]]

# Initialize the scaler
scaler = StandardScaler()

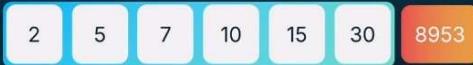
# Fit the scaler to the data and transform it
scaled_data = scaler.fit_transform(data)

print("Original data:")
print(data)

print("\nScaled data:")
print(scaled_data)
```

Handling Outliers

Extreme values that deviate significantly from the rest of the data



Misleading

$$\text{Mean} = (2+5+7+10+15+30+8953) / 7 = 1288$$

More accurate

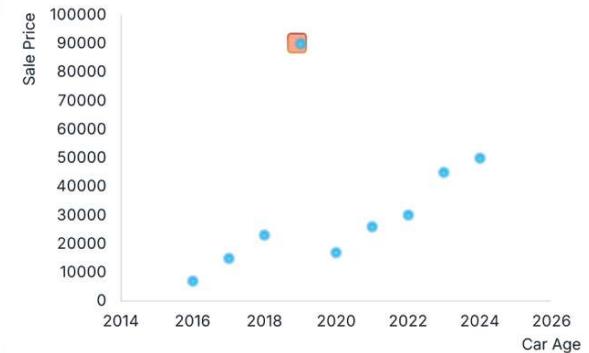
$$\text{Mean} = (2+5+7+10+15+30) / 6 = 11.5$$

Handling Outliers

1 Data entry issue?

2 Data corruption?

3 Valid, but extreme?



Handling Outliers

2 5 7 10 15 30 8953

Outlier is valid.

Transformation

Transform outliers with log, square root, or clipping to minimize their impact.

2 5 7 10 15 30 log(8953)

Handling Outliers



Outlier is valid.

Robust Scaling

Use median and IQR for scaling instead of mean and standard deviation.

Interquartile Range (IQR)



Copyright KodeCloud

23:15



Interquartile Range (IQR)

Detect Outliers

Lower Bound: Any value below $Q1 - 1.5 * IQR$ is an outlier.

Upper Bound: Any value above $Q3 + 1.5 * IQR$ is an outlier.

II P

25:37



Interquartile Range (IQR)

Data

2

5

7

10

15

30

90

Q1 = 5

Q2 = 10

Q3 = 30

IQR = Q3 - Q1 = 30 - 5 = 25

Lower Bound = $Q1 - 1.5 * IQR = 5 - (1.5 * 25) = 5 - 37.5 = -32.5$

Upper Bound = $Q3 + 1.5 * IQR = 30 + (1.5 * 25) = 30 + 37.5 = 67.5$

Outlier: 90 is above 67.5 (outlier detected)



```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler, RobustScaler

# Sample data with outliers
data = {'Feature': [1, 2, 3, 4, 5, 100]} # 100 is an outlier
df = pd.DataFrame(data)

# Detect outliers using IQR
q1 = df['Feature'].quantile(0.25)
q3 = df['Feature'].quantile(0.75)
iqr = q3 - q1
lower_bound = q1 - 1.5 * iqr
upper_bound = q3 + 1.5 * iqr

# Cap outliers
df['Feature_capped'] = np.clip(df['Feature'], lower_bound, upper_bound)
```

