Day 3: Session Guide

1. Removing Special Characters

**LOAD**

**DocumentID,**

**PurgeChar(Lower(TextContent), '!"#$%&''()\*+,-./:;<=>?@[\]^\_’{|}~') AS NormalizedText**

**RESIDENT YourTextData;**

1. Stop words removal

**table1:**

**Load \* Inline**

**[**

**prodid,prodname**

**1,the chair**

**2,table with**

**3,efan in**

**];**

**stopwords:**

**Load \* Inline**

**[**

**sw, value**

**in,''**

**as,''**

**the,''**

**with,''**

**of,''**

**if,''**

**];**

**newtable1:**

**NoConcatenate**

**Load**

**prodid,**

**MapSubString('stopwords',prodname) as prodname**

**Resident table1;**

**drop table stopwords;**

**drop table table1;**

1. Stemming and lemmatization

To perform stemming and lemmatization on all the columns in a CSV dataset, you can use Python along with popular natural language processing libraries like NLTK (for stemming) and spaCy (for lemmatization).

**import pandas as pd**

**import nltk**

**import spacy**

**from nltk.stem import PorterStemmer**

**# Load NLTK and spaCy models**

**nltk.download('punkt')**

**nlp = spacy.load('en\_core\_web\_sm')**

**# Load the CSV dataset**

**df = pd.read\_csv('sample\_superstore.csv') # Replace with your dataset file path**

**# Function for stemming using NLTK**

**def stem\_text(text):**

**stemmer = PorterStemmer()**

**words = nltk.word\_tokenize(text)**

**stemmed\_words = [stemmer.stem(word) for word in words]**

**return ' '.join(stemmed\_words)**

**# Function for lemmatization using spaCy**

**def lemmatize\_text(text):**

**doc = nlp(text)**

**lemmatized\_words = [token.lemma\_ for token in doc]**

**return ' '.join(lemmatized\_words)**

**# Apply stemming and lemmatization to all columns**

**for column in df.columns:**

**if df[column].dtype == 'O': # Check if the column contains text data**

**df[f'{column}\_stemmed'] = df[column].apply(stem\_text)**

**df[f'{column}\_lemmatized'] = df[column].apply(lemmatize\_text)**

**# Save the processed dataset to a new CSV file**

**df.to\_csv('sample\_superstore\_processed.csv', index=False) # Change the output file path as needed**

1. Entity recognition

Qlik Sense does not have a native Named Entity Recognition (NER) function. One workaround is to use a third-party NER API. There are a number of NER APIs available, such as Google Cloud Natural Language API and Amazon Comprehend. You can use these APIs to extract named entities from your data and then load the extracted entities into Qlik Sense for analysis.

**# example using python**

**import spacy**

**# Load the English NER model**

**nlp = spacy.load('en\_core\_web\_sm')**

**# Sample text**

**text = "Apple Inc. was founded in April 1976 by Steve Jobs and Steve Wozniak. The company is headquartered in Cupertino, California."**

**# Process the text with spaCy**

**doc = nlp(text)**

**# Extract and print named entities**

**for ent in doc.ents:**

**print(f"Entity: {ent.text}, Type: {ent.label\_}")**

**# extract email address from column using python**

**import pandas as pd**

**import re**

**# Load the CSV dataset**

**input\_file = 'your\_dataset.csv' # Replace with the path to your CSV dataset**

**output\_file = 'output\_dataset.csv' # Specify the path for the output CSV**

**# Function to extract email addresses from a text**

**def extract\_email(text):**

**email\_pattern = r'\S+@\S+'**

**email\_addresses = re.findall(email\_pattern, text)**

**return ', '.join(email\_addresses)**

**# Read the CSV into a DataFrame**

**df = pd.read\_csv(input\_file)**

**# Apply the email extraction function to the specified column and create a new column**

**df['ExtractedEmails'] = df['TextColumn'].apply(extract\_email) # Replace 'TextColumn' with your column name**

**# Save the DataFrame to a new CSV file with the extracted email addresses**

**df.to\_csv(output\_file, index=False)**

**print(f"Email addresses extracted and saved to {output\_file}")**

1. Binning and categorization

**// Load the Sample Superstore dataset**

**LOAD \***

**FROM [lib://SampleSuperstore/SampleSuperstore.xlsx]**

**(ooxml, embedded labels, table is Orders);**

**// Binning and Categorization**

**Orders:**

**LOAD \*,**

**If(Sales >= 0 and Sales < 100, 'Low Sales',**

**If(Sales >= 100 and Sales < 500, 'Medium Sales',**

**If(Sales >= 500, 'High Sales', 'Undefined'))) as SalesCategory**

**RESIDENT Orders;**

**// Drop the original "Sales" field**

**DROP FIELD Sales;**

**// using class function**

**superstore:**

**LOAD**

**\***

**FROM [lib://DataFiles/storeTransactions.csv]**

**(txt, codepage is 28591, embedded labels, delimiter is ',', msq);**

**Orders:**

**NoConcatenate**

**LOAD \*,**

**Class(Sales, 500) as SalesClass**

**RESIDENT superstore;**

**drop table superstore;**

1. Aggregation

**// example 1**

**superstore:**

**LOAD**

**\***

**FROM [lib://DataFiles/storeTransactions.csv]**

**(txt, codepage is 28591, embedded labels, delimiter is ',', msq);**

**regiontotals:**

**Load**

**Region as Regions,**

**Sum(Sales) as TotalSales,**

**Sum(Profit) as TotalProfit**

**Resident superstore**

**group by Region;**

**// example 2**

**superstore:**

**LOAD**

**\***

**FROM [lib://DataFiles/storeTransactions.csv]**

**(txt, codepage is 28591, embedded labels, delimiter is ',', msq);**

**regiontotals:**

**Load**

**Region as Regions,**

**State as States,**

**Sum(Sales) as TotalSales,**

**Sum(Profit) as TotalProfit**

**Resident superstore**

**group by Region,State**

**order by Region;**

1. Using chart functions

**Aggr(Sum(Sales), Category, Region)**

**//example 1: total sales for each customer**

**CustomerSales:**

**LOAD**

**CustomerName,**

**Sum(Sales) as TotalSales**

**RESIDENT SampleSuperstore**

**GROUP BY CustomerName;**

**CustomerTotalSales:**

**LOAD**

**CustomerName,**

**Sum(Aggr(Sum(Sales), CustomerName)) as AggregatedTotalSales**

**RESIDENT SampleSuperstore;**

**DROP TABLE CustomerSales;**

**//example 2: find the category with the highest total sales**

**FirstSortedValue(Category,-Aggr(sum(Sales),Category))**

**//example 3: calculate the top N products by sales**

**TopProducts:**

**LOAD**

**ProductName,**

**Sum(Sales) as TotalSales**

**RESIDENT SampleSuperstore**

**GROUP BY ProductName;**

**TopNProducts:**

**First 10**

**LOAD**

**ProductName,**

**TotalSales**

**RESIDENT TopProducts**

**ORDER BY TotalSales DESC;**

// use aggr() in chart

1. Create a table or chart object:
2. Add a table or chart object to your sheet.

* In the "Dimensions" section, add the "Product Sub-Category" field.
* In the "Measures" section, add a measure using the Aggr function to calculate the total sales.

1. Use the Aggr function:

* In the "Measures" section of your chart, create a new measure.
* Define the measure using the Aggr function like this:

**Aggr(Sum(Sales), [Product Sub-Category])**

// use aggr() with filter conditions

**Aggr(Sum({<Category={'Technology'}>} Sales), Product)**

\*note: the chart script above will not change even if sheet filters are applied

1. Using Chart Set Analysis

**// example 1: total sales for a selected year and region**

**Sum({<Year={'2020'}, Region={'West'}>} Sales)**

**// example 2: calculate Total Sales Excluding a Category**

**Sum({<Category-={'Technology'}>} Sales)**

**// example 3: total sales for the top N customers, where N is a variable.**

**Sum({<CustomerName={"=Rank(Sum(Sales)) <= $(vTopN)"}>} Sales)**

**// example 4: calculate the sales growth over time by comparing the current year's sales to the previous year's sales**

**(Sum({<Year={'2022'}>} Sales) - Sum({<Year={'2021'}>} Sales)) / Sum({<Year={'2021'}>} Sales)**

**// example 5: variables in set analysis**

**LET vThreshold = 1000;**

**Sum({<Sales={"<=$(vThreshold)"}>} Sales)**

1. Measures of central tendency

**Avg(Sales)**

**Median(Sales)**

**Mode(Country)**

**Aggr(Mode(Sales), Country)**

1. Measures of dispersion

Variance measures how much individual data points deviate from the mean. It's the average of the squared differences between each data point and the mean.

* A high variance indicates that data points are spread out widely from the mean.
* A low variance suggests that data points are clustered closely around the mean.
* Variance is always non-negative; it can't be negative.

**// get variance in data load script**

**// Load the Sample Superstore data**

**Load \* Inline [**

**OrderID, Category, Sales**

**1, Office Supplies, 100**

**2, Furniture, 200**

**3, Technology, 300**

**];**

**// Calculate the mean (average) of Sales**

**MeanSales:**

**Load Avg(Sales) as MeanSales**

**Resident YourData;**

**// Calculate the variance of Sales**

**Variance:**

**Load Sum((Sales - MeanSales) \* (Sales - MeanSales)) / Count(Sales) as Variance**

**Resident YourData;**

**// Drop the intermediate table with MeanSales**

**Drop Table MeanSales;**

**//example 2: in a chart expression**

**Sum((Sales-vAverageSales) \* (Sales-vAverageSales)) / Count(Sales)**

**//where vAverageSales is a variable**

**=Avg(Sales)**

The standard deviation is the square root of the variance. It's expressed in the same units as the data and provides a more intuitive measure of dispersion.

* A high standard deviation means that data points are dispersed over a wide range.
* A low standard deviation indicates that data points are concentrated near the mean.

Use: Standard deviation is a commonly used measure of variability and is often preferred over variance because it's in the same units as the data.

**// standard deviation**

**Stdev(Sales)**

**// range**

**RangeMax(Sales) - RangeMin(Sales)**

1. Quantiles and percentiles

**// chart expression**

**Aggr(Fractile(Sales, 0.25), Category) // 25th percentile**

**Aggr(Fractile(Sales, 0.5), Category) // Median (50th percentile)**

**Aggr(Fractile(Sales, 0.75), Category) // 75th percentile**

1. Visualizing data distribution

Visualizing data distributions in Qlik Sense can provide insights into the characteristics of your data, including its shape, central tendency, and spread. Histograms and density plots are commonly used visualizations for exploring data distributions.

1. **Histograms:**

A histogram is a graphical representation of the distribution of numerical data. It divides the data into bins or intervals and shows the frequency or count of data points in each bin. To create a histogram in Qlik Sense:

1. Open your Qlik Sense app and add a chart object, such as a bar chart.
2. Dimension:

**=Class(Sales, 5) // where 5 is the interval**

1. In the "Measures" area, add the following expression to count the data points in each bin:

**Count(Sales)**

1. Customize your chart by adding titles, labels, and styling as needed.
2. The resulting chart will display a histogram showing the distribution of the "Sales" variable.
3. **Density Plots:**

Density plots, such as kernel density plots, provide a smooth representation of the data distribution. They are particularly useful for visualizing the shape and central tendency of the data. To create a density plot in Qlik Sense:

1. Open your Qlik Sense app and add a chart object, such as a Scatter plot.
2. Bubble: Category or Sub-category or Region

X-Axis: Count(Sales)

Y-Axis: Count(Sales)

1. Customize your chart by adding titles, labels, and styling as needed.
2. The resulting chart will display a smooth density plot illustrating the data distribution.
3. Skewness and kurtosis

// Skewness measures the asymmetry of the distribution

* positive skewness indicates that the distribution is skewed to the right (tail on the right),
* negative skewness indicates a leftward skew (tail on the left).
* A perfectly symmetrical distribution has a skewness of zero.

**Skew(Sales)**

// kurtosis measures the heaviness of the tails and the presence of outliers.

* positive kurtosis indicates heavy tails and potential outliers,
* negative kurtosis indicates light tails and a lack of outliers.
* normal distribution has a kurtosis of 3 (excess kurtosis).

**Kurtosis(Sales)**

1. Normality testing

**Visual Normality Assessment:**

Create a histogram or density plot to visualize the data distribution. A bell-shaped curve in the chart suggests a normal distribution.

**Normality Test Expressions:**

These tests provide p-values that indicate whether the data significantly deviates from a normal distribution.

In python:

**import pandas as pd**

**from scipy import stats**

**# Load your CSV dataset**

**df = pd.read\_csv('your\_dataset.csv') # Replace 'your\_dataset.csv' with the actual file path**

**# Select the column for which you want to perform the normality tests**

**column\_to\_test = df['ColumnName'] # Replace 'ColumnName' with the name of your column**

**# Shapiro-Wilk Test**

**shapiro\_stat, shapiro\_p = stats.shapiro(column\_to\_test)**

**print("Shapiro-Wilk Test:")**

**print(f"Statistic: {shapiro\_stat}")**

**print(f"P-value: {shapiro\_p}")**

**if shapiro\_p > 0.05:**

**print("Data looks normally distributed (fail to reject H0)")**

**else:**

**print("Data does not look normally distributed (reject H0)")**

**# Anderson-Darling Test**

**anderson\_stat, anderson\_critical, anderson\_significance = stats.anderson(column\_to\_test)**

**print("\nAnderson-Darling Test:")**

**print(f"Statistic: {anderson\_stat}")**

**print(f"Critical Values: {anderson\_critical}")**

**if anderson\_stat < anderson\_critical[2]:**

**print("Data looks normally distributed (fail to reject H0)")**

**else:**

**print("Data does not look normally distributed (reject H0)")**

**# Kolmogorov-Smirnov Test (against a normal distribution)**

**ks\_stat, ks\_p = stats.kstest(column\_to\_test, 'norm')**

**print("\nKolmogorov-Smirnov Test (against a normal distribution):")**

**print(f"Statistic: {ks\_stat}")**

**print(f"P-value: {ks\_p}")**

**if ks\_p > 0.05:**

**print("Data follows a normal distribution (fail to reject H0)")**

**else:**

**print("Data does not follow a normal distribution (reject H0)")**

**Interpretation**

1. Select a Significance Level (Alpha):

In hypothesis testing, the significance level (often denoted as alpha, α) is pre-defined and represents the threshold at which you consider a result statistically significant. A common significance level is 0.05, which corresponds to a 5% chance of making a Type I error (incorrectly rejecting a true null hypothesis).

2. Compare the P-Value to the Significance Level:

After performing a normality test in Qlik Sense (e.g., using the Shapiro-Wilk, Anderson-Darling, or Kolmogorov-Smirnov test), you'll obtain a p-value.

Here's a practical guideline for interpreting p-values in the context of normality testing:

**p ≤ α: Reject the null hypothesis.**

**The data significantly deviates from a normal distribution.**

**p > α: Fail to reject the null hypothesis.**

**There's no significant evidence that the data deviates from a normal distribution.**

1. Correlation analysis

**Pearson correlation coefficient**

The Pearson correlation coefficient, often denoted as "r," is a statistical measure that quantifies the linear relationship between two continuous variables in Qlik Sense. It ranges from -1 to 1, where:

r = 1 indicates a perfect positive linear relationship (as one variable increases, the other increases proportionally).

r = -1 indicates a perfect negative linear relationship (as one variable increases, the other decreases proportionally).

r = 0 indicates no linear relationship; the variables are uncorrelated.

To calculate the Pearson correlation coefficient in Qlik Sense, follow these steps:

1. Open your Qlik Sense app and create a chart or table object.

2. Drag and drop the two continuous variables you want to calculate the correlation for into the visualization. For this example, let's assume you want to calculate the correlation between the "Sales" and "Profit" variables.

3. Create a new expression to calculate the correlation. You can use the ‘Correl()’ function in Qlik Sense. The syntax is as follows:

Correl(Sales, Profit)

4. Customize your chart or table as needed. You can add titles, labels, and formatting options to make the visualization more informative.

5. The resulting chart or table will display the Pearson correlation coefficient ("r") between the two variables. The value can range from -1 to 1, indicating the strength and direction of the linear relationship between the variables.

**Interpreting the Pearson correlation coefficient:**

1. Perfect Positive Correlation (1):

- A correlation coefficient of 1 indicates a perfect positive correlation.

- It means that the two variables move in the same direction. As one variable increases, the other also increases in a linear fashion.

2. High Positive Correlation (0.7 to 0.99):

- A correlation coefficient between 0.7 and 0.99 suggests a strong positive correlation.

- It indicates that there is a strong linear relationship between the two variables. As one variable increases, the other tends to increase.

3. Moderate Positive Correlation (0.3 to 0.69):

- A correlation coefficient between 0.3 and 0.69 indicates a moderate positive correlation.

- It suggests that there is a moderate linear relationship between the two variables. An increase in one variable is associated with an increase in the other.

4. No Correlation (0):

- A correlation coefficient of 0 suggests no linear correlation between the two variables.

- It means that there is no discernible relationship between the variables, or any relationship is non-linear.

5. Moderate Negative Correlation (-0.3 to -0.69):

- A correlation coefficient between -0.3 and -0.69 indicates a moderate negative correlation.

- It suggests that there is a moderate linear relationship between the two variables, but as one variable increases, the other tends to decrease.

6. High Negative Correlation (-0.7 to -0.99):

- A correlation coefficient between -0.7 and -0.99 suggests a strong negative correlation.

- It indicates a strong linear relationship in the opposite direction. As one variable increases, the other tends to decrease.

7. Perfect Negative Correlation (-1):

- A correlation coefficient of -1 indicates a perfect negative correlation.

- It means that the two variables move in opposite directions. As one variable increases, the other decreases in a linear fashion.

**Spearman rank correlation (Non-qliksense)**

The Spearman rank correlation coefficient, often denoted as "ρ" or "rho," is a non-parametric measure of association between two variables in Qlik Sense. It assesses the strength and direction of a monotonic relationship between two continuous or ordinal variables. Spearman's rank correlation is particularly useful when the relationship between variables is not strictly linear. Here's how to calculate the Spearman rank correlation coefficient in Qlik Sense:

1. Open your Qlik Sense app and create a chart or table object.

2. Drag and drop the two variables you want to calculate the Spearman rank correlation for into the visualization. For this example, let's assume you want to calculate the correlation between the "Sales" and "Profit" variables.

3. Create a new expression to calculate the Spearman rank correlation. You can use the ‘Spearman()’ function in Qlik Sense. The syntax is as follows:

Spearman(Sales, Profit)

In this expression, replace "Sales" and "Profit" with the names of your two variables.

4. Customize your chart or table as needed. You can add titles, labels, and formatting options to make the visualization more informative.

5. The resulting chart or table will display the Spearman rank correlation coefficient (ρ) between the two variables. The value of ρ can range from -1 to 1, indicating the strength and direction of the monotonic relationship between the variables.

Interpreting the Spearman rank correlation coefficient:

If ρ is close to 1, there is a strong positive monotonic relationship between the two variables.

If ρ is close to -1, there is a strong negative monotonic relationship between the two variables.

If ρ is close to 0, there is little to no monotonic relationship between the two variables.

Spearman's rank correlation is robust to outliers and does not assume linearity. It works by converting the values of the two variables into ranks and then calculating the Pearson correlation coefficient on the ranks. This makes it a valuable measure for analyzing relationships between variables when the relationship isn't strictly linear, or when you want to assess associations between ordinal data.

**Correlation matrices (non-qliksense)**

Creating a correlation matrix in Qlik Sense allows you to examine the relationships between multiple variables, helping you understand how they are related to each other. A correlation matrix displays the correlation coefficients between pairs of variables, and it can provide valuable insights into your data.

1. Open your Qlik Sense app and create a new table object.

2. Choose the variables you want to include in the correlation matrix. Typically, these should be continuous or ordinal variables, as correlation is primarily used for assessing relationships between such variables. You can select these variables and drag them into the visualization.

3. Create the correlation matrix expression. In Qlik Sense, you can use the ‘CorrelationMatrix()’ function to calculate the correlation coefficients between the selected variables. Add this expression to the table:

CorrelationMatrix([Variable1, Variable2, Variable3, ...])

Replace ‘[Variable1, Variable2, Variable3, ...]’ with the list of variables you've selected. This expression will generate a matrix of correlation coefficients between these variables.

4. Customize your table object as needed. You can add titles, labels, and formatting options to make the correlation matrix more informative.

5. The resulting table object will display the correlation matrix, showing the correlation coefficients between all pairs of selected variables. The values in the matrix will range from -1 to 1, indicating the strength and direction of the relationships between the variables.

**Interpreting the correlation matrix:**

Positive values (close to 1) indicate a positive correlation (as one variable increases, the other tends to increase).

Negative values (close to -1) indicate a negative correlation (as one variable increases, the other tends to decrease).

Values close to 0 indicate little to no linear correlation between the variables.

The correlation matrix is a valuable tool for exploring multivariate relationships in your data. It can help identify which variables are strongly related and which are not. Keep in mind that correlation does not imply causation, and the matrix measures linear associations. Non-linear relationships may not be accurately represented by correlation coefficients.

**Heatmaps for correlation visualization (non-qliksense)**

Heatmaps are an effective way to visualize a correlation matrix in Qlik Sense, making it easier to identify patterns and relationships between variables. In a heatmap, colors represent the strength and direction of correlations, allowing you to quickly spot strong and weak associations. Here's how to create a heatmap for correlation visualization in Qlik Sense:

1. Open your Qlik Sense app and create a new sheet or visualization sheet where you want to display the heatmap.

2. Add a table object to the sheet. This table will be the foundation for the heatmap.

3. Select the variables for which you want to create a correlation heatmap. Typically, these should be continuous or ordinal variables.

4. Create the correlation matrix using the ‘CorrelationMatrix()’ function. In Qlik Sense, you can calculate the correlation matrix and display it as a heatmap. Add the expression to the table object:

CorrelationMatrix([Variable1, Variable2, Variable3, ...])

Replace ‘[Variable1, Variable2, Variable3, ...]’ with the list of variables you've selected for the heatmap.

5. Customize the table object to make it visually resemble a heatmap:

* Row and Column Styling: Make the row and column headers invisible or minimal to emphasize the heatmap cells.
* Cell Styling: Adjust the cell background color to represent correlation values using color gradients. The colors can vary from, for example, dark blue for negative values to dark red for positive values.
* Cell Labels: You can choose to display correlation values in the heatmap cells or use color intensity alone to convey the information.
* Conditional Formatting: You can use conditional formatting rules to assign colors to specific correlation ranges, such as a gradient from -1 to 1.
* Labels and Titles: Add labels, titles, and descriptions to the heatmap to help viewers understand the content.

6. The resulting table object will display a heatmap representation of the correlation matrix. Cells will be colored based on the strength and direction of the correlation between the selected variables.

**Interpreting the heatmap:**

Dark blue or colors close to blue represent strong negative correlations.

Dark red or colors close to red represent strong positive correlations.

Light colors, such as white, represent correlations close to zero or little to no linear relationship.

Using a heatmap for correlation visualization in Qlik Sense provides an intuitive way to explore the relationships between multiple variables. It's a powerful tool for identifying patterns and trends in your data, especially when dealing with a large number of variables.

1. Hypothesis testing

**T-tests for mean comparison**

In Qlik Sense, you can perform a t-test for mean comparison using the built-in `ttest` function. Here's how you can do it with the Sample Superstore dataset:

1. Ensure you have the Sample Superstore dataset loaded into Qlik Sense.
2. Create a table or pivot table in your Qlik Sense app where you want to display the results of the t-test.
3. In the chart, add an expression that uses the `ttest` function to compare the means of the "Furniture" and "Technology" product categories. The `ttest` function takes three parameters: the field you want to test (Sales), the first dimension (ProductCategory), and the second dimension (ProductCategory) for comparison.

**Ttest\_t(Sales, 'Furniture', 'Technology')**

1. Qlik Sense will display the results of the t-test, including the t-statistic, degrees of freedom, p-value, and whether the test is significant. These results will indicate whether there is a significant difference between the means of the "Furniture" and "Technology" product categories.
2. To interpret the results, look at the p-value. If the p-value is less than your chosen significance level (e.g., 0.05), you can conclude that there is a significant difference between the means of the two categories.

* If the p-value is less than or equal to alpha (p ≤ α), you reject the null hypothesis (H0). This suggests there is a significant difference in means between the two groups.
* If the p-value is greater than alpha (p > α), you fail to reject the null hypothesis, indicating that there is no significant difference in means.

**# demo ttest in python**

**import pandas as pd**

**from scipy import stats**

**# Load the Sample Superstore dataset**

**df = pd.read\_excel("Sample\_Superstore\_Sales.xlsx")**

**# Select the two groups for the t-test (e.g., "Consumer" and "Corporate")**

**group1 = df[df["Category"] == "Consumer"]["Sales"]**

**group2 = df[df["Category"] == "Corporate"]["Sales"]**

**# Perform the t-test**

**t\_statistic, p\_value = stats.ttest\_ind(group1, group2)**

**# Display the results**

**print("T-Statistic:", t\_statistic)**

**print("P-Value:", p\_value)**

**# Interpret the results**

**alpha = 0.05 # Set your desired significance level (e.g., 0.05)**

**if p\_value < alpha:**

**print("Reject the null hypothesis: There is a significant difference between the groups.")**

**else:**

**print("Fail to reject the null hypothesis: There is no significant difference between the groups.")**

**Chi-squared tests for categorical data**

A Chi-squared test is a statistical test used to assess the independence or association between two categorical variables in Qlik Sense. It helps determine whether there is a significant relationship between the variables. Chi-squared tests are commonly used in fields such as market research, social sciences, and quality control.

**// Sample\_1 data is pre-aggregated... Note: make sure you set your DecimalSep='.' at the top of the script.**

**Sample\_1:**

**LOAD \* inline [**

**Grp,Grade,Count**

**I,A,15**

**I,B,7**

**I,C,9**

**I,D,20**

**I,E,26**

**I,F,19**

**II,A,10**

**II,B,11**

**II,C,7**

**II,D,15**

**II,E,21**

**II,F,16**

**];**

**Chi2\_table:**

**LOAD Grp,**

**Chi2Test\_chi2(Grp, Grade, Count) as chi2,**

**Chi2Test\_df(Grp, Grade, Count) as df,**

**Chi2Test\_p(Grp, Grade, Count) as p**

**resident Sample\_1 group by Grp;**

**Results**

**Grp chi2 df p**

**I 16.00 5 0.007**

**II 9.40 5 0.094**

To interpret the result, look at the p-value. If the p-value is less than your chosen significance level (e.g., 0.05), you can conclude that there is a significant association between "Region" and "Category."

* If the p-value is less than or equal to alpha (p ≤ α), you reject the null hypothesis (H0). This suggests a significant association or relationship between the categorical variables.
* If the p-value is greater than alpha (p > α), you fail to reject the null hypothesis, indicating that there is no significant association.
* Chi (χ²): The chi-squared statistic is a measure of the difference between the observed and expected frequencies in a contingency table. It is calculated by summing the squared differences between the observed and expected frequencies, divided by the expected frequencies.
* Degrees of freedom (df): The degrees of freedom of a chi-squared test is the number of independent comparisons that are being made in the test. It is calculated by multiplying the number of rows in the contingency table minus one by the number of columns in the contingency table minus one.
* P-value: The p-value is the probability of obtaining a chi-squared statistic as large as or larger than the observed value, assuming that the null hypothesis is true. The p-value is used to determine whether or not to reject the null hypothesis.

1. Exploring categorical data

**Frequency tables**

[using Sample Superstore Dataset]

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the frequency table.

3. Add a table object to the sheet:

4. Select the "Category" variable:

5. Create the frequency table:

* Drag the "Category" variable into either the "Rows" or "Columns" area of the "Pivot Table" object, depending on how you want to structure the table. For this example, let's place it in the "Rows" area.

6. Customize the frequency table:

* To display the counts of each category, you don't need to add any expressions in the "Measures" area. The table will automatically count the occurrences of each category.

**Bar charts for categorical variables**

[using Sample Superstore Dataset]

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the bar chart.

3. Add a chart object to the sheet:

4. Select "Bar Chart" as the chart type:

5. Choose the dimension and measure:

* For the "Dimension," select the categorical variable you want to analyze. Ex. "Category" variable.
* For the "Measure," you can select any field that represents a numerical value. Ex. "Sales" or "Profit"

6. Customize the bar chart:

* You can customize the bar chart by adding titles, labels, and formatting options to make it more informative.
* You can also choose to display counts, sums, or percentages as the measure in the chart, depending on your analysis needs.

**Stacked bar charts**

[using Sample Superstore Dataset]

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the stacked bar chart.

3. Add a chart object to the sheet:

4. Select "Bar Chart" as the chart type:

5. Choose the dimension and measures:

* For the "Dimension," select the categorical variable you want to analyze. In this case, choose the "Category" variable.
* For the "Measures," you need two variables to create a stacked bar chart. Select the "Region" variable to break down the categories by region.

6. Customize the stacked bar chart:

* You can customize the chart by adding titles, labels, and formatting options to make it more informative.
* Ensure that the chart type is set to "Stacked" to create a stacked bar chart.

**Pie charts**

[using Sample Superstore Dataset]

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the pie chart.

3. Add a chart object to the sheet:

4. Select "Pie Chart" as the chart type:

5. Choose the dimension and measure:

* For the "Dimension," select the categorical variable you want to analyze. In this case, choose the "Category" variable.
* For the "Measure," you can select any field that represents a numerical value. For example, you can choose "Sales" to display the total sales for each category.

6. Customize the pie chart:

* You can customize the chart by adding titles, labels, and formatting options to make it more informative.
* Choose the "Donut Chart" option if you want a donut chart instead of a traditional pie chart.

1. Time series analysis

\*Note: We’ll use sample superstore dataset, let us add a continuous date/time column in excel

A screenshot of a computer

Description automatically generated

**1. Load your time series data:**

**Ensure that your data includes a field representing time, such as a date or timestamp, and at least one other numeric field that you want to analyze over time.**

**2. Open your Qlik Sense app and go to the sheet where you want to create the time series plot.**

**3. Add a chart object to the sheet:**

**4. Select "Line Chart" as the chart type:**

**5. Choose the dimension and measure:**

* **For the "Dimension," select the time-related field. In our case, it's the "OrderDate" field.**
* **For the "Measure," select the numeric field you want to analyze over time. Here, we'll choose the "Sales" field.**

**6. Customize the time series plot:**

* **You can customize the chart by adding titles, labels, and formatting options to make it more informative.**
* **Make sure to set the "Dimension Axis" to represent time, and the "Measure Axis" to represent your numeric data, like sales.**

**Seasonal Decomposition**

Calculate seasonal decomposition in Qlik Sense with the stl\_trend() and stl\_seasonal() functions using the sample Superstore dataset:

**// chart 1**

1. Create a new Qlik Sense app.
2. Load the Superstore dataset.
3. Create a new sheet in the app.
4. Add a table visualization to the sheet.
5. In the table visualization, add the following fields:

* Order Date
* Sales

1. Click the Add Measure button and select the stl\_trend() function.

**STL\_Trend(Sum(Sales),12) // 12 = means yearly or every 12 months**

1. Click the Add Measure button and select the stl\_seasonal() function.

**STL\_Seasonal(Sum(Sales),12) // 12 = means yearly or every 12 months**

1. The table visualization will now display the original sales data, the seasonal component, and the trend component.

The trend component of the sales data shows the overall trend of sales over time. In the case of the Superstore dataset, the trend component shows that sales are increasing over time.

**// chart 2**

1. Add a line chart
2. Add YearMonth as a dimension and label it Date.
3. Add the following measure and label it Sales per month:

**=Sum(Sales)**

1. Under Data, expand the Sales per month measure and click Add trend line.
2. Set the Type to Linear.
3. You will compare this trend line to the smoothed output of the trend component.
4. Add the following measure to plot the trend component and label it Trend:

**STL\_Trend(SUM(Sales), 12)**

1. Next, add the following measure to plot the seasonal component and label it Seasonal:

**STL\_Seasonal(SUM(Passengers), 12)**

1. Under Appearance > Presentation, set Scroll bar to None.
2. Keep the default colors, or change them to fit your preferences.
3. Bivariate analysis

**Scatter plots with regression lines**

In Qlik Sense, you can perform bivariate analysis using scatter plots with regression lines to visualize the relationship between two numeric variables and understand how they are related. This analysis is particularly useful for examining correlations or trends in your data.

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the scatter plot with a regression line.

3. Add a chart object to the sheet:

4. Select "Scatter Plot" as the chart type:

5. Choose the dimension and measures:

* For the "Dimension," select the first numeric variable you want to analyze. In the Sample Superstore dataset, let's use "Sales."
* For the "Measure," select the second numeric variable you want to analyze. For this example, let's use "Profit."

6. Customize the scatter plot:

7. Add a trendline (regression line):

* In Qlik Sense, you can add a trendline to the scatter plot to visualize the regression line. To do this, click on the "Add Expression" button and create a new expression.

8. Define the regression line expression:

To add a linear regression line, you can use the "linest\_m" function. The syntax for the expression is as follows:

linest\_m(Sales, Profit)

This expression calculates the slope of the linear regression line.

9. Visualize the scatter plot with the regression line:

* After defining the expression for the regression line, you will see the scatter plot with the regression line displayed.

10. Customize the chart to enhance the visualization and insights:

**Grouped bar charts**

In Qlik Sense, you can perform bivariate analysis using grouped bar charts with regression lines to visualize the relationship between multiple categorical variables and a numeric variable. This type of analysis can help you understand how different categories relate to a numeric measure, and regression lines can provide insights into trends within each category.

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the grouped bar charts with regression lines.

3. Add a chart object to the sheet:

4. Select "Grouped Bar Chart" as the chart type:

5. Choose the dimensions and measures:

* In a grouped bar chart with regression lines, you typically have multiple dimensions, such as two categorical variables, and one measure, which is a numeric variable. For the Sample Superstore dataset, let's select "Region" and "Category" as dimensions, and "Sales" as the measure.

6. Customize the grouped bar chart:

7. Add a trendline (regression line) for each group:

* In Qlik Sense, you can add trendlines to visualize regression lines for each group in the grouped bar chart. To do this, you can use the "Add Trend Line" option in the chart properties.

8. Define the regression line expression:

* Configure the trendline properties and specify that you want a linear regression line. The trendline should be calculated for each group (e.g., each region or category).

9. Visualize the grouped bar chart with regression lines:

10. Customize the chart to enhance the visualization and insights:

**2D density plots**

Creating a bivariate analysis with 2D density charts and regression lines in Qlik Sense using the Sample Superstore dataset can provide insights into the relationship between two numeric variables. In this example, we'll use "Sales" and "Profit" as the two numeric variables to demonstrate how to create a 2D density chart with regression lines:

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the 2D density chart with regression lines.

3. Add a chart object to the sheet:

4. Select "Scatter Plot" as the chart type:

5. Choose the dimensions and measures:

* For the "Dimension," select one of the numeric variables you want to analyze, such as "Sales."
* For the "Measure," select the other numeric variable you want to analyze, such as "Profit."

6. Customize the scatter plot:

7. Add a 2D density chart:

* In Qlik Sense, you can add a 2D density chart to your scatter plot. To do this, click on the "Add Expression" button and create a new expression.

8. Define the 2D density chart expression:

**kde(Sales, Profit)**

This expression calculates the density of data points in the scatter plot.

9. Visualize the scatter plot with the 2D density chart:

10. Add regression lines:

* To add regression lines to the chart, go to the "Add Expression" menu and create new expressions for linear regression lines or other regression models. You can use functions like "linest\_m" as shown in a previous response.

11. Customize the chart to enhance the visualization and insights:

**Contour plots**

Creating a bivariate analysis with contour plots and regression lines in Qlik Sense using the Sample Superstore dataset can provide valuable insights into the relationship between two numeric variables. In this example, we'll use "Sales" and "Profit" as the two numeric variables to demonstrate how to create a contour plot with regression lines:

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to create the contour plot with regression lines.

3. Add a chart object to the sheet:

4. Select "Scatter Plot" as the chart type:

5. Choose the dimensions and measures:

* For the "Dimension," select one of the numeric variables you want to analyze, such as "Sales."
* For the "Measure," select the other numeric variable you want to analyze, such as "Profit."

6. Customize the scatter plot:

7. Add a contour plot:

* In Qlik Sense, you can add a contour plot to your scatter plot to visualize the density of data points. To do this, click on the "Add Expression" button and create a new expression.

8. Define the contour plot expression:

To add a contour plot, you can use the "kde" function to calculate the density. The syntax for the expression is as follows:

kde(Sales, Profit)

This expression calculates the density of data points in the scatter plot.

9. Visualize the scatter plot with the contour plot:

10. Add regression lines:

* To add regression lines to the chart, go to the "Add Expression" menu and create new expressions for linear regression lines or other regression models. You can use functions like "linest\_m" as shown in previous responses.

11. Customize the chart to enhance the visualization and insights:

1. Moving average

Statistical forecasting using a moving average is a simple but effective method to smooth out variations in time series data and make short to medium-term predictions. In Qlik Sense, you can apply moving average forecasting to the Sample Superstore dataset to predict future values of a numeric variable.

1. Load the Sample Superstore dataset. Add years.

**superstore:**

**LOAD**

**"Ship Mode",**

**Segment,**

**Country,**

**City,**

**State,**

**"Postal Code",**

**Region,**

**Category,**

**"Sub-Category",**

**Sales,**

**Quantity,**

**Discount,**

**Profit,**

**OrderDate,**

**Year(OrderDate) as year**

**FROM [lib://DataFiles/storeTransactions.csv]**

**(txt, codepage is 28591, embedded labels, delimiter is ',', msq);**

**Concatenate**

**Load**

**2002 + IterNo() as year**

**AutoGenerate (1)**

**while 2002 + IterNo() <= 2005;**

2. Open your Qlik Sense app and go to the sheet where you want to perform moving average forecasting.

3. Add a chart object to the sheet

4. Select "Line Chart" as the chart type

5. Choose the dimension and measure:

* For the "Dimension," select the time-related field you want to analyze, such as "Order Year"
* For the "Measure," select the numeric variable you want to forecast. For this example, let's use "Sales."
* Add another measure (2nd param is offset, 3rd param is interval count, the ff is good for 5-year period if year was used as dimension)

RangeAvg(

Above(sum(Sales), 0, 1),

Above(sum(Sales), 1, 1),

Above(sum(Sales), 2, 1),

Above(sum(Sales), 3, 1),

Above(sum(Sales), 4, 1)

)

7. Visualize the moving average:

8. Customize the chart:

9. Use the moving average for forecasting:

* The moving average line represents the smoothed trend in the data. You can use this line to make short to medium-term predictions. For example, if you want to forecast the next three months of sales, look at the moving average value for the current month, and assume that the trend will continue.

10. Interpret the results:

Keep in mind that a moving average is a relatively simple forecasting method. While it can help smooth out short-term fluctuations and identify trends, it may not capture more complex patterns in the data. Be cautious when making long-term predictions based solely on moving averages.

1. Exponential smoothing

Exponential smoothing is a widely used statistical forecasting method for time series data. It's a powerful technique for predicting future values based on historical data, and you can apply it to the Sample Superstore dataset in Qlik Sense. Exponential smoothing gives more weight to recent observations, making it especially useful when there is a trend or seasonality in the data.

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to perform exponential smoothing forecasting.

3. Add a chart object to the sheet:

4. Select "Line Chart" as the chart type:

5. Choose the dimension and measure:

* For the "Dimension," select the time-related field you want to analyze, such as "Order Date."
* For the "Measure," select the numeric variable you want to forecast. For this example, let's use "Sales."

6. Create an exponential smoothing expression:

In Qlik Sense, you can create an expression for exponential smoothing using the "Esmooth()" function. You need to specify the field to be smoothed (e.g., "Sales") and the alpha (α) parameter, which controls the smoothing level. The closer α is to 1, the more weight recent values have. A common choice for α is 0.2.

**Esmooth(Sales, 0.2)**

This expression calculates the exponentially smoothed values of "Sales" based on the specified alpha.

7. Visualize the smoothed values:

8. Customize the chart:

9. Use the smoothed values for forecasting:

10. Interpret the results:

* Exponential smoothing provides a way to capture the underlying trend in the data while reducing the impact of short-term fluctuations. It can be a valuable tool for forecasting when there is a clear trend or seasonality in the data.

11. Evaluate and fine-tune your forecasts:

* It's important to evaluate the accuracy of your forecasts and adjust the α parameter as needed. You may also want to use different levels of smoothing (e.g., double or triple exponential smoothing) if the data exhibits seasonality.

1. Autoregressive integrated moving average (ARIMA)

Performing statistical forecasting with an ARIMA (AutoRegressive Integrated Moving Average) model is a more advanced and powerful technique for time series analysis compared to simpler methods like moving averages and exponential smoothing. In Qlik Sense, you can use the ARIMA model to forecast future values based on historical data from the Sample Superstore dataset.

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to perform ARIMA forecasting.

3. Add a chart object to the sheet:

4. Select "Line Chart" as the chart type:

5. Choose the dimension and measure:

* For the "Dimension," select the time-related field you want to analyze, such as "Order Date."
* For the "Measure," select the numeric variable you want to forecast. For this example, let's use "Sales."

6. Create an ARIMA expression:

In Qlik Sense, you can create an expression for ARIMA forecasting using the "arima()" function. You need to specify the field to be forecasted (e.g., "Sales"), the order of the ARIMA model (p, d, q), and the number of forecast periods.

**arima(Sales, 1, 1, 1, 12)**

In this example, the ARIMA model has parameters (1, 1, 1) for the AR order, differencing, and MA order, respectively, and is set to forecast 12 periods ahead.

7. Visualize the ARIMA forecast:

8. Customize the chart:

9. Interpret the results:

* The ARIMA model captures both short-term and long-term patterns in the data. It's important to evaluate the model's accuracy and make any necessary adjustments to the model order.

10. Evaluate model accuracy:

* Use measures like Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) to assess the accuracy of the ARIMA forecasts.

11. Fine-tune the ARIMA model:

* You may need to experiment with different ARIMA model orders (p, d, q) to achieve the best forecasting results. Qlik Sense allows you to interactively adjust these parameters in the chart properties.

12. Consider seasonality:

* If your data exhibits seasonality, you should consider seasonal ARIMA models (SARIMA) or other advanced models to capture these patterns effectively.

1. Linear regression

Statistical forecasting using linear regression in Qlik Sense can be a powerful tool for predicting values based on historical data and understanding relationships between variables. In this example, we'll demonstrate how to use linear regression to forecast a numeric variable (e.g., "Sales") based on other predictor variables (e.g., "Profit" and "Discount") in the Sample Superstore dataset.

1. Load the Sample Superstore dataset:

2. Open your Qlik Sense app and go to the sheet where you want to perform linear regression forecasting.

3. Add a chart object to the sheet:

4. Select "Scatter Plot" as the chart type:

5. Choose the dimension and measures:

* For the "Dimension," select one of the predictor variables you want to use, such as "Profit."
* For the "Measure," select the numeric variable you want to forecast, such as "Sales."

6. Fit a linear regression model:

* In Qlik Sense, you can fit a linear regression model to the scatter plot by using the "Trend Line" functionality. Select "Trend Lines" in the chart properties and choose "Linear" as the trend line type.

7. Visualize the linear regression model:

* The scatter plot will display the original data points along with the linear regression line that represents the relationship between the predictor and target variables.

8. Customize the chart:

9. Use the linear regression model for forecasting:

* The linear regression model can be used to make predictions based on the predictor variables. For example, if you want to forecast sales based on a given profit and discount, enter those values and use the regression line equation.

10. Interpret the results:

* The linear regression line provides insights into the relationship between the predictor and target variables. The slope and intercept of the line are essential for making forecasts.

11. Evaluate model accuracy:

* Assess the accuracy of your linear regression model using metrics like the coefficient of determination (R-squared) and Mean Squared Error (MSE).

12. Fine-tune the model:

* You may need to refine the model by adding more predictor variables or experimenting with other regression techniques, such as multiple linear regression if you have multiple predictors.