MIT-ADT UNIVERSITY
PUNE, INDIA
A leap towards World Class Education
(Established by MIT Art, Design and Technology University Act, 2015
(Maharashtra Act No. XXXIX of 2015)

MIT Art Design and Technology University

MIT School of Computing, Pune

Department of Computer Science and Engineering

Lab Manual

Course- Data Modeling & Visualization Laboratory

Class - T.Y. (SEM-VI) AIA

Name of the Course Coordinator

Prof. Dr. Rahesha Mulla

Team Members

1. Prof. Dr. Jayashree Prasad
2. Prof. Dr. Saiprasad Potharaju
3. Prof. Dr. Rashmi Nair
4. Prof. Dr. Ranjana Kale
5. Prof. Dr. Sunita Parinam

A.Y. 2023 - 2024

Lab Experiment List

Sr. No.	Name of Experiment	CO
1	Download any dataset from UCI or Data.org or any other data repositories and perform the basic data pre-processing steps using R. //RYM	CO1
2	Download any dataset from UCI or Data.org or any other data repositories and perform the basic visualization using R //RYM	CO1, CO4
3	Download any dataset from UCI or Data.org or any other data repositories and perform the basic transformation using R //RSK	CO4, CO5
4	Download any dataset from UCI or Data.org or any other data repositories and perform Exploratory data analysis using R //RSK	CO4
5	Download any dataset from UCI or Data.org or any other data repositories and perform suitable data modeling operation as per the list given below, with respect to the dataset. • Linear regression • Logistic regression //SUP	CO3
6	Download any dataset from UCI or Data.org or any other data repositories and perform suitable data modeling operation as per the list given below, with respect to the dataset. • K-nearest neighbors • K-means clustering// SUP	CO3
7	Introduction to Tableau and Installation. //RYM	CO1
8	Download any dataset from UCI or Data.org or any other data repositories and perform • Connecting to data and preparing data for visualization in Tableau. • Data Aggregation and Statistical functions in Tableau • Data Visualizations in Tableau • Basic Dashboards in Tableau //RN	CO2, CO4, CO5
9	Introduction to PowerBI and installation. //RYM	CO1
10	Download any dataset from UCI or Data.org or any other data repositories and perform • Connecting to data and preparing data for visualization in PowerBI. • Data Aggregation and Statistical functions in PowerBI • Data Visualizations in PowerBI • Basic Dashboards in PowerBI //RN	CO2, CO4, CO5
11	Download any dataset of CSV files, XML, JSON format and perform data modeling, and create a basic dashboard using Tableau and PowerBI. //JP	CO2, CO3, CO4, CO5
12	Download any dataset from NoSQL and perform data modeling and create a basic dashboard using Tableau and PowerBI. // PS	CO2, CO3, CO4, CO5

List of Hardware / Software Requirements

❖ Hardware requirements:

1. Laptop/Desktop machine
2. Internet facility

❖ Software Requirements

3. RStudio-Open-source IDE
4. Tableau
5. PowerBI
6. SQL, NoSQL

Experiment No 1

Experiment Title: Data Preprocessing

Problem Statement:

Download any dataset from UCI or Data.org or any other data repositories and perform the basic data pre-processing steps using R. [CO1]

Objective:

Acquire hands-on experience in data preprocessing on a dataset sourced from different data repositories such as UCI or data.org

Theory:

Data Preprocessing:

Introduction

Data cleaning and preprocessing are crucial steps in the data analysis process. They involve identifying and rectifying errors, inconsistencies, and missing values in the dataset to ensure accurate and reliable results. R is a popular programming language for statistical computing and data analysis and offers a wide range of tools and packages to effectively clean and preprocess data. Here, we will explore various techniques and methodologies in R for data cleaning and preprocessing.

Understanding Data Cleaning

Importance of Data Cleaning: Data cleaning is an essential step before conducting any analysis as it helps in improving data quality, reliability, and overall accuracy of the results. Unclean data may contain errors, outliers, or missing values, which can lead to biased or incorrect conclusions. Cleaning the data ensures that subsequent analyses are based on accurate and trustworthy information.

Common Data Cleaning Tasks

- **Handling Missing Data** − Missing data can significantly impact the analysis and interpretation of results. R provides functions like **is.na()** and **complete.cases()** to identify and handle missing values. Techniques such as imputation, where missing values are replaced with estimated values, can be performed using packages like **mice** or **missForest**.

- **Outlier Detection and Treatment** − Outliers are extreme values that deviate significantly from the rest of the data. R offers various methods, such as the use of **boxplots, z-scores**, or the **Mahalanobis distance** to detect outliers. Once identified, outliers can be treated by removing them or transforming them to more reasonable values.

- **Removing Duplicates** − Duplicate records in a dataset can introduce bias and affect the integrity of the analysis. R provides functions like **duplicated()** and **distinct()** to identify and remove duplicates based on specific columns or combinations of columns.

- **Data Validation** − Validating the integrity and consistency of data is crucial. R offers validation techniques like **cross-tabulation, data profiling**, and **summary statistics** to ensure data accuracy.

Data Preprocessing Techniques

- **Data Integration** − Data integration involves combining multiple datasets with similar variables or structures. R provides functions like **merge()** and **rbind()** to merge datasets based on common identifiers or variables. Proper data integration ensures a unified dataset for analysis.
- **Data Transformation** − Data transformation involves converting raw data into a suitable format for analysis. R provides functions like **scale()**, **log()** or **sqrt()** to normalize or transform skewed data distributions. These transformations help meet the assumptions of statistical models and improve interpretability.
- **Feature Selection** − Feature selection aims to identify the most relevant variables for analysis. R offers techniques like correlation analysis, stepwise regression, or regularization methods (e.g., Lasso or Ridge regression) to select informative features and avoid overfitting.
- **Encoding Categorical Variables** − Categorical variables often require encoding to numerical representations for analysis. R offers functions like **factor()** or **dummyVars()** to convert categorical variables into binary or numerical representations. This process enables the inclusion of categorical variables in statistical models.
- **Handling Imbalanced Data** − Imbalanced datasets, where one class dominates over others, can lead to biased predictions or model performance. R provides techniques such as oversampling (e.g., SMOTE) or under sampling to balance the dataset and improve model training.

R Packages for Data Cleaning and Preprocessing

- **Tidyverse** − Tidyverse is a collection of R packages, including **dplyr, tidyr**, and **stringr**, that provide powerful tools for data manipulation, cleaning, and tidying. These packages offer a consistent and intuitive syntax for transforming and cleaning data.
- **Caret** − The caret package (Classification and Regression Training) in R provides functions for data preprocessing, feature selection, and resampling techniques. It offers a comprehensive set of tools for preparing data for machine learning algorithms.
- **DataPreparation** − The DataPreparation package in R provides a wide range of functions for data cleaning, transformation, and preprocessing. It offers functionalities like missing value imputation, outlier detection, feature scaling, and more.

Conclusion: Data cleaning and preprocessing are vital steps in the data analysis workflow. R provides a rich set of tools, libraries, and packages that facilitate effective data cleaning and preprocessing. By employing these techniques, data scientists can ensure the accuracy, reliability, and validity of their analyses. A clean and preprocessed dataset forms the foundation for meaningful insights and successful data-driven decision-making.

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Useful Resource:

1. https://www.geeksforgeeks.org/data-preprocessing-in-r/
2. https://www.homeworkhelponline.net/blog/programming/data-preprocessing-r-studio-example-solutions
3. https://analyticsindiamag.com/data-preprocessing-with-r-hands-on-tutorial/
4. https://analyticsindiamag.com/data-preprocessing-with-r-hands-on-tutorial/

Exercise Questions

1. Explain the concept of packages in R and how to use them.
2. How can you import data in R?
3. Write an R function to remove missing values from a data frame.
4. Create an R function to perform one-hot encoding on categorical variables in a data frame.
5. Write an R function to scale numerical features in a data frame using Min-Max scaling.

Experiment No 2

Experiment Title: Data Visualization

Problem Statement: Download any dataset from UCI or Data.org or any other data repositories and perform the basic visualization using R

Objective:Use packages in R for understanding patterns and relationships within datasets through data visualization.

Theory:
In R, there are several packages available for data visualization, but one of the most popular and versatile ones is ggplot2. Here's a brief overview of data visualization using ggplot2:

1. Installation:

Before creating visualizations with ggplot2, make sure you have the package installed. If not, install it using:*install.packages("ggplot2")*

Load the package into your R environment: *library(ggplot2)*

2. Basic Scatter Plot:

Create a simple scatter plot using the *ggplot()* function and the *geom_point()* layer:

3. Line Plot:

Generate a line plot with the *geom_line()* layer.

4. Bar Chart:

Create a bar chart using the *geom_bar()* layer:

5. Histogram:

Build a histogram with the *geom_histogram()* layer.

6. Boxplot:

Construct a boxplot using the *geom_boxplot()* layer.

7. Scatter Plot with Regression Line:

Include a linear regression line in a scatter plot

8. Faceting:

Use *facet_wrap()* or *facet_grid()* to create multiple plots based on a variable:

Conclusion:

Data visualization is a technique used for the graphical representation of data. By using elements like scatter plots, charts, graphs, histograms, maps, etc., we make our data more understandable. Data visualization makes it easy to recognize patterns, trends, and exceptions in our data.

Source Code and Output /Screenshots:

(To be provided by the student)

```
//Source code and output/screenshot should be available here
```

Useful Resource:

1. https://www.geeksforgeeks.org/data-visualization-in-r/
2. https://rkabacoff.github.io/datavis/
3. https://www.analyticsvidhya.com/blog/2015/07/guide-data-visualization-r/
4. https://www.javatpoint.com/r-data-visualization

Exercise Questions

1. Create a bar plot to visualize the distribution of a categorical variable in a given dataset. Customize the plot to include proper axis labels, title, and color.
2. Generate a scatter plot to visualize the relationship between two numerical variables. Add a trendline or regression line to depict the overall trend. Choose appropriate colors and labels.
3. Use a boxplot to display the distribution of a numerical variable across different categories. Include outliers in the plot, and make sure to label the axes appropriately.
4. Create a histogram to visualize the distribution of a numeric variable. Adjust the number of bins for better clarity and provide a title and axis labels.
5. Create a 3D scatter plot to visualize the relationship between three numerical variables. Use a package like *plotly* or *rgl* for 3D plotting.

Experiment No 3

Experiment Title: Data Transforming

Problem Statement: Download any dataset from UCI or Data.org or from any other data repositories and perform the basic transformation using R. [CO1, CO4]

Objective:

Acquire hands-on experience in data transformation on a dataset sourced from different data repositories such as UCI or data.org

Theory:

Data Transformation

Introduction

Data Transformation is one of the key **aspects of working** for business data analysis, data science or even for the pre-work of artificial intelligence. **Data transformation** is the process of **converting, cleansing, and structuring** the data into a usable format that can be analyzed to support decision-making processes and to propel the growth of an organization. Data transformation is used when data needs to be converted to match that of the destination system. On a basic level, the data transformation process converts raw data into a usable format by removing duplicates, converting data types, and enriching the dataset. This data transformation process involves defining the structure, mapping the data, extracting the data from the source system, performing the transformations, and then storing the transformed data in the appropriate dataset. Organizations perform data transformation to ensure the compatibility of data with other types while combining it with other information or migrating it into a dataset. Through data transformations, organizations can gain valuable insights into the operational and informational functions.

The data transformation process is carried out in five stages.

1. **Discovery**- The first step is to identify and understand data in its original source format with the help of data profiling tools. Finding all the sources and data types that need to be transformed. This step helps in understanding how the data needs to be transformed to fit into the desired format.

2. **Mapping** - The transformation is planned during the data mapping phase. This includes determining the current structure, and the consequent transformation that is required, then mapping the data to understand at a basic level, the way individual fields would be modified, joined or aggregated.

3. **Code Generation** - The code, which is required to run the transformation process, is created in this step using a data transformation platform or tool.

4. Execution - The data is finally converted into the selected format with the help of the code. The data is extracted from the source(s), which can vary from structured to streaming, telemetry to log files. Next, transformations are carried out on data, such as aggregation, format conversion or merging, as planned in the mapping stage. The transformed data is then sent to the destination system which could be a dataset or a data warehouse.

Some of the transformation types, depending on the data involved, include:

- Filtering which helps in selecting certain columns that require transformation
- Enriching which fills out the basic gaps in the data set
- Splitting where a single column is split into multiple or vice versa
- Removal of duplicate data, and
- Joining data from different sources

5. Review - The transformed data is evaluated to ensure the conversion has had the desired results in terms of the format of the data.

It must also be noted that not all data will need transformation, at times it can be used as is.

Data Transformation Techniques

There are several data transformation techniques available. These techniques can be used to clean the data and structure it before it is stored in a data warehouse or analyze this data for business intelligence. Not all of these techniques work with all types of data, and sometimes more than one technique may be applied. Nine of the most common techniques are:

1. Revising - Revising ensures the data supports its intended use by organizing it in the required and correct way. It involves dataset normalization, Data cleansing, Format conversion, Deduplication, and Data validation.

2. Manipulation - This involves creation of new values from existing ones or changing current data through computation. Manipulation is also used to convert unstructured data into structured data that can be used by machine learning algorithms. It involves Derivation, Summarization that aggregates values, Sorting, ordering, and indexing of the data, Scaling, normalization and standardization and Vectorization.

3. Separating - This involves dividing up the data values into its parts for granular analysis. Splitting involves dividing up a single column with several values into separate columns with each of those values. This allows for filtering on the basis of certain values.

4. Combining/ Integrating - Records from across tables and sources are combined to acquire a more holistic view of activities and functions of an organization. It couples data from multiple tables and datasets and combines records from multiple tables.

5. Data Smoothing- This process removes meaningless, noisy, or distorted data from the data set. By removing outliers, trends are most easily identified.

6. Data Aggregation - This technique gathers raw data from multiple sources and turns it into a summary form which can be used for analysis. An example is the raw data providing statistics such as averages and sums.

7. Discretization -With the help of this technique, interval labels are created in continuous data in an attempt to enhance its efficiency and easier analysis. The decision tree algorithms are utilized by this process to transform large datasets into categorical data.

8. Generalization -Low level data attributes are transformed into high level attributes by using the concept of hierarchies and creating layers of successive summary data. This helps in creating clear data snapshots.

9. Attribute Construction- In this technique, a new set of attributes is created from an existing set to facilitate the mining process.

Benefits of Data Transformation

Data holds the potential to directly affect an organization's efficiencies and its bottom line. It plays a crucial role in understanding customer behavior, internal processes, and industry trends. While every organization has the ability to collect an immense amount of data, the challenge is to ensure that this is usable. Data transformation processes empower organizations to reap the benefits offered by the data.

Data Utilization - If the data being collected isn't in an appropriate format, it often ends up not being utilized at all. With the help of data transformation tools, organizations can finally realize the true potential of the data they have amassed since the transformation process standardizes the data and improves its usability and accessibility.

Data Consistency - Data is continuously being collected from a range of sources which increases the inconsistencies in metadata. This makes organization and understanding data a huge challenge. Data transformation helps making it simpler to understand and organize data sets.

Better Quality Data- Transformation process also enhances the quality of data which can then be utilized to acquire business intelligence.

Compatibility Across Platforms - Data transformation also supports compatibility between types of data, applications and systems.

Faster Data Access -It is quicker and easier to retrieve data that has been transformed into a standardized format.

R Packages

Tidyverse − Tidyverse is a collection of R packages, including dplyr, tidyr, and stringr, that provide powerful tools for data manipulation, cleaning, and tidying. These packages offer a consistent and intuitive syntax for transforming and cleaning data.

The dplyr package is a popular and powerful package in the R programming language for data manipulation and transformation. It provides a set of functions that allow users to perform various data manipulation tasks in a concise and readable manner.

Conclusion: **Data transformation is very important for data consistency, data quality improvement. R provides a rich set of tools, libraries, and packages that transform data.**

Source Code and Output /Screenshots:

(To be provided by the student)

```
//Source code and output/screenshot should be available here
```

Useful Resource:

https://towardsdatascience.com/data-transformation-in-r-288e95438ff9
https://www.slideshare.net/RsquaredIn/r-programming-transformreshape-data-transformation

Exercise Questions

Q1. Why Data transformation required?
Q2. What are the types of data transformation?
Q3. Explain Normalization techniques used for data transformation?
Q4. What is smoothing in data transformation?

Experiment No 4

Experiment Title: Exploratory Data Analysis

Problem Statement:Download any dataset from UCI or Data.org or from any other data repositories and perform the basic transformation using R. [CO1, CO4]

Objective:

Acquire hands-on experience in exploratory data analysis on a dataset sourced from different data repositories such as UCI or data.org

Theory:

Exploratory Data Analysis

Introduction

Data analysis involves different processes of cleaning, transforming, analyzing the data, and building models to extract specific, relevant insights. These are beneficial for making important business decisions in real-time situations. Exploratory Data Analysis is important for any business. It **lets data scientists analyze the data before reaching any conclusion.** Also, this makes sure that the results that are out are valid and applicable to business outcomes and goals. **Exploratory Data Analysis (EDA) is one of the techniques used for extracting vital features and trends used by machine learning and deep learning models in Data Science.** Thus, EDA has become an important milestone for anyone working in data science. The data analysis **approach involves summarizing and visualizing data to understand its key characteristics, uncover patterns, and identify anomalies.**

Types of Exploratory Data Analysis

There are three main types of EDA:

1. Univariate
2. Bivariate
3. Multivariate

In univariate analysis, the output is a single variable and all data collected is for it. There is no cause-and-effect relationship at all. For example, data shows products produced each month for twelve months. In bivariate analysis, the outcome is dependent on two variables, e.g., the age of an employee, while the relation with it is compared with two variables, i.e., his salary earned and expenses per month. In multivariate analysis, the outcome is more than two, e.g., type of product and quantity sold against the

product price, advertising expenses, and discounts offered. The analysis of data is done on variables that can be numerical or categorical. The result of the analysis can be represented in numerical values, visualization, or graphical form. Accordingly, they could be further classified as non-graphical or graphical.

1. Univariate Non-Graphical

It is the simplest of all types of data analysis used in practice. As the name suggests, **uni means only one variable is considered** whose data (referred to as population) is compiled and studied. The main aim of univariate non-graphical EDA is **to find out the details about the distribution of the population data and to know some specific parameters of statistics.** The significant parameters which are estimated from a distribution point of view are as follows:

Central Tendency: This term refers to values located at the data's central position or middle zone. The three generally estimated parameters of central tendency are **mean, median, and mode.** Mean is the average of all values in data, while the mode is the value that occurs the maximum number of times. The Median is the middle value with equal observations to its left and right.

Range: The range is the difference between the maximum and minimum value in the data, thus indicating how much the data is away from the central value on the higher and lower side.

Variance and Standard Deviation: Two more useful parameters are standard deviation and variance. Variance is a **measure of dispersion that indicates the spread of all data points** in a data set. It is the measure of dispersion mostly used and is the mean squared difference between each data point and mean, while **standard deviation is the square root value of it.** The larger the value of standard deviation, the farther the spread of data, while a low value indicates more values clustering near the mean.

2. Univariate Graphical

Stem-and-leaf Plots: This is a very simple but powerful EDA method **used to display quantitative data but in a shortened format.** It displays the values in the data set, keeping each observation intact but separating them as stem (the leading digits) and remaining or trailing digits as leaves. But histogram is mostly used in its place now.

Histograms (Bar Charts): These plots are used to display both grouped or ungrouped data. On the x-axis, values of variables are plotted, while on the y-axis are the number of observations or frequencies. Histograms are very simple to quickly understand your data, which tell about values of data like central tendency, dispersion, outliers, etc. The simplest fundamental graph is a histogram, which is a bar plot

with each bar representing the frequency, i.e., the count or proportion (the ratio of count to the total count of occurrences) for various values.

3. Multivariate Non-Graphical

The multivariate non-graphical exploratory data analysis technique is usually used **to show the connection between two or more variables** with the help of either cross-tabulation or statistics.

- For categorical data, an extension of tabulation called cross-tabulation is extremely useful. For two variables, cross-tabulation is preferred by making a two-way table with column headings that match the amount of one variable and row headings that match the amount of the opposite two variables, then filling the counts with all subjects that share an equivalent pair of levels.
- For each categorical variable and one quantitative variable, we can generate statistical information for quantitative variables separately for every level of the specific variable. We then compare the statistics across the number of categorical variables.

4. Multivariate Graphical

Graphics are used in multivariate graphical data to show the relationships between two or more variables. Here the outcome depends on more than two variables, while the change-causing variables can also be multiple.

Some common types of multivariate graphics include:

Scatter Plot

The essential graphical EDA technique for two quantitative variables is the scatter plot, so one variable appears on the x-axis and the other on the y-axis and, therefore, the point for every case in your dataset. This can be used for bivariate analysis.

Multivariate Chart

A Multivariate chart is a type of control chart used to monitor two or more interrelated process variables. This is beneficial in situations such as process control, where engineers are likely to benefit from using multivariate charts. These charts allow the monitoring of multiple parameters together in a single chart. A notable advantage of using multivariate charts is that they help minimize the total number of control charts for organizational processes. Pair plots generated using the Seaborn library are a good example of multivariate charts as they help visualize the relationships between all numerical variables in the entire dataset at once.

Run Chart

A run chart is a data line chart drawn over time. In other words, a run chart visually illustrates the process performance or data values in a time sequence. Rather than summary statistics, seeing data across time yields a more accurate conclusion. A trend chart or time series plot is another name for a run chart. The plot below depicts dummy values of sales over a period of time.

Bubble Chart

Bubble charts scatter plots that display multiple circles (bubbles) in a two-dimensional plot. These are used to assess the relationships between three or more numeric variables. In a bubble chart, every single dot corresponds to one data point, and the values of the variables for each point are indicated by different positions such as horizontal, vertical, dot size, and dot colors.

Heat Map

A heat map is a colored graphical representation of multivariate data structured as a matrix of columns and rows. The heat map transforms the **correlation matrix into color coding and represents these coefficients to visualize the strength of correlation among variables.** It assists in finding the best features suitable for building accurate Machine Learning models.

R Packages for EDA

Tidyverse − Tidyverse is a collection of R packages, including dplyr, tidyr, and stringr, that provide powerful tools for data manipulation, cleaning, and tidying. These packages offer a consistent and intuitive syntax for transforming and cleaning data.

The dplyr package is popular and powerful in the R

programming language for data manipulation and transformation. It provides a

set of functions that allow users to perform various data manipulation tasks in

a concise and readable manner.

soilDB is one of the Algorithms for Quantitative Pedology (AQP) suite of R packages and comprises a collection of functions for reading data from USDA-NCSS (National Cooperative Soil Survey) soil databases including SoilWeb, Series Extent Explorer, and Soil Data Explorer.

STEPS:

1. **Load the Dataset:** Load the loan dataset into R using functions like **read.csv()** or **read.table()**.

2. **Summary Statistics:** Use functions like **summary()** to get an overview of the dataset, including numerical summaries of each variable and counts of unique values for categorical variables.

3. **Data Cleaning:** Check for missing values: Use functions like **is.na()** or **complete.cases()** to identify missing values.

Handle missing values: Depending on the context, you may choose to remove missing values, impute them, or leave them as-is.

4. **Univariate Analysis:**Explore distributions of numerical variables using histograms, boxplots, or density plots. For categorical variables, examine frequency tables or bar plots to understand the distribution of categories.

5. **Bivariate Analysis:** Explore relationships between pairs of variables using scatter plots (for numerical variables) or stacked bar plots (for categorical variables). Use correlation analysis to quantify the strength and direction of relationships between numerical variables.

6. **Multivariate Analysis:** Explore relationships involving more than two variables using techniques like heatmaps (for correlation matrices), pair plots (for scatterplot matrices), or parallel coordinate plots.

7. **Visualization:**Create visualizations to summarize key insights from the data. Use libraries like **ggplot2** for creating customizable and visually appealing plots.

8. **Outlier Detection:** Identify outliers in the data using boxplots, scatter plots, or z-scores. Decide how to handle outliers based on domain knowledge and the objectives of your analysis.

9. **Feature Engineering:**Create new variables or transform existing ones to extract additional information from the dataset. For example, create new variables based on existing ones, such as calculating ratios or aggregating categorical variables.

10.**Final Insights:** Summarize key findings and insights from the EDA process. Identify patterns, trends, and relationships that may inform subsequent analyses or modeling efforts.

Conclusion: **Thus performed Exploratory Data Analysis using different library packages.**

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Useful Resource: **A Book "Exploratory Data Analysis with R" by *Roger D. Peng, 2020***

Exercise Questions

Q1. Which library is used for EDA?

Q2. What are the measures of central tendency to perform EDA?

Q3. What is the need of Exploratory data analysis?

Experiment No 5

Experiment Title: Regression

Problem Statement:

Download any dataset from UCI or Data.org or any other data repositories and perform suitable data modeling operation as per the list given below, with respect to the dataset.

- Linear regression
- Logistic regression

Objective: Use packages in R to perform regression for predicting dependent variable using independent variables

Theory:

Regression predicts the continuous output variables based on the independent input variable. E.g. Prediction of house prices based on different parameters like house age, distance from the main road, location, area, etc.

There are two types of regression

1) Linear regression
2) Logistic regression

1) Linear Regression- It is a supervised learning algorithm that computes a linear relationship between a dependent variable and one or more independent variables/features. If the number of linear features is 1,

it is known as Univariate or Simple Linear Regression, and if features are more than 1, it is known as Multivariate or Multiple Linear Regression.

Simple Linear Regression
The equation for simple linear regression is: □ = □0 + □1 □▄▄where:
Y is the dependent variable

X is the independent variable

β0 is the intercept

β1 is the slope

Multiple Linear Regression
The equation for multiple linear regression is □ = □0 + □1 □1 + □2 □2+..... □□ □□ where:

Y is the dependent variable

X1, X2, …, Xp are the independent variables

β0 is the intercept

β1, β2, …, βn are the slopes

Some other regression types are **Polynomial Regression, Ridge Regression, Lasso Regression**, **Elastic Net Regression** (explore more on these topics)

The goal of Linear regression is to find the best Fit Line equation that can predict the values based on the independent variables. This implies that the error between the predicted and actual values should be kept to a minimum.
The best Fit Line equation provides a straight line (Fig. Linear Regression) that represents the relationship between the dependent and independent variables. The slope of the line indicates how much the dependent variable changes for a unit change in the independent variable(s).

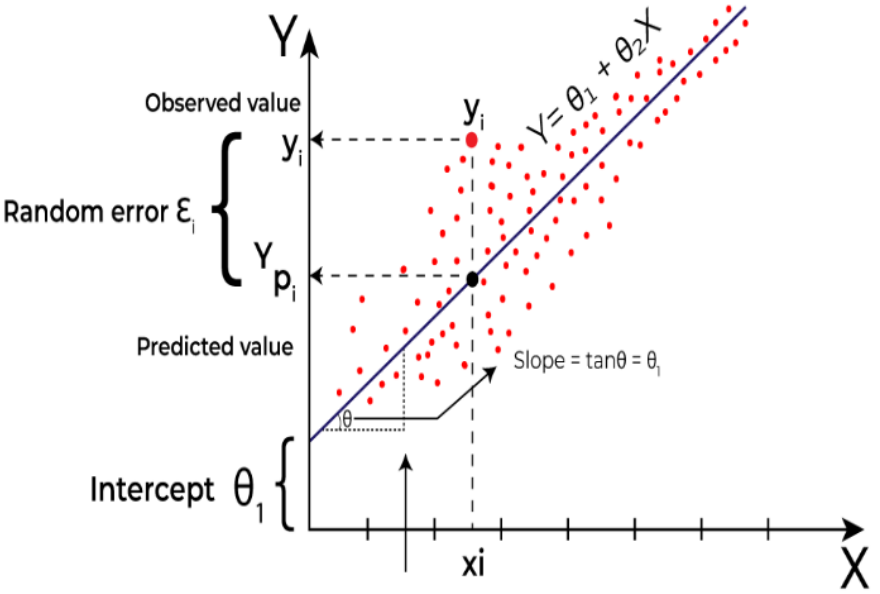


Fig. Linear Regression
Here Y is called a dependent or target variable and X is called an independent variable also known as the predictor of Y.

Cost function for Linear Regression

The cost function or the loss function is nothing but the error or difference between the predicted value and the true value.

- **Mean Squared Error (MSE) -** It calculates the average of the squared errors between the predicted values and the actual values. MSE function can be calculated as:

$$\text{MSE}(\theta) = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

- **Gradient Descent for Linear Regression -**The idea is to start with random θ1 and θ2 values and then iteratively update the values, reaching minimum cost (Fig. Gradient Descent). A gradient is nothing but a derivative that defines the effects on outputs of the function with a little bit of variation in inputs.

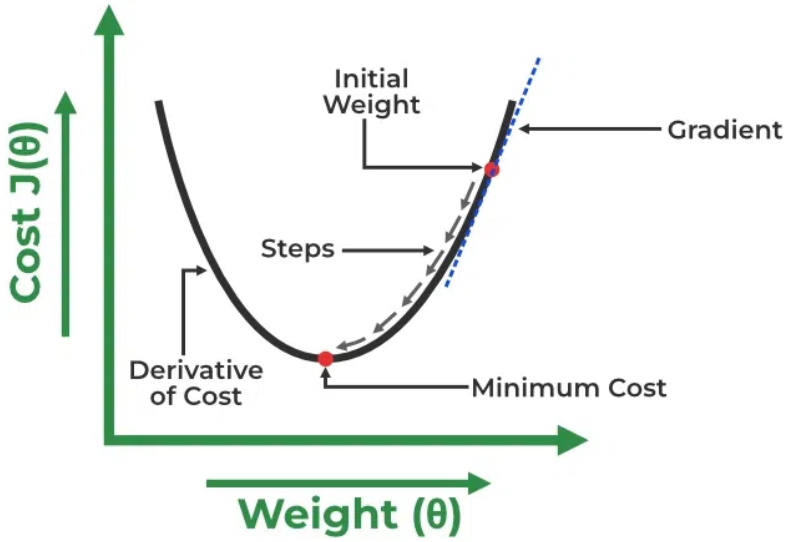


Fig. Gradient Descent

2) Logistic Regression

- Logistic regression is a supervised machine learning algorithm mainly used for classification tasks where the goal is to predict the probability that an instance belongs to a given class or not. For example, whether an email is spam or not.
- It's referred to as regression because it is the extension of linear regression but is mainly used for classification problems.
- The difference between linear regression and logistic regression is that linear regression output is the continuous value that can be anything while logistic regression predicts the probability that an instance belongs to a given class or not.
- It is used for predicting the categorical dependent variable using a given set of independent variables.
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

Logistic Function (Sigmoid Function):

- The sigmoid function (Fig. Sigmoid Function) is a mathematical function used to map the predicted values to probabilities.

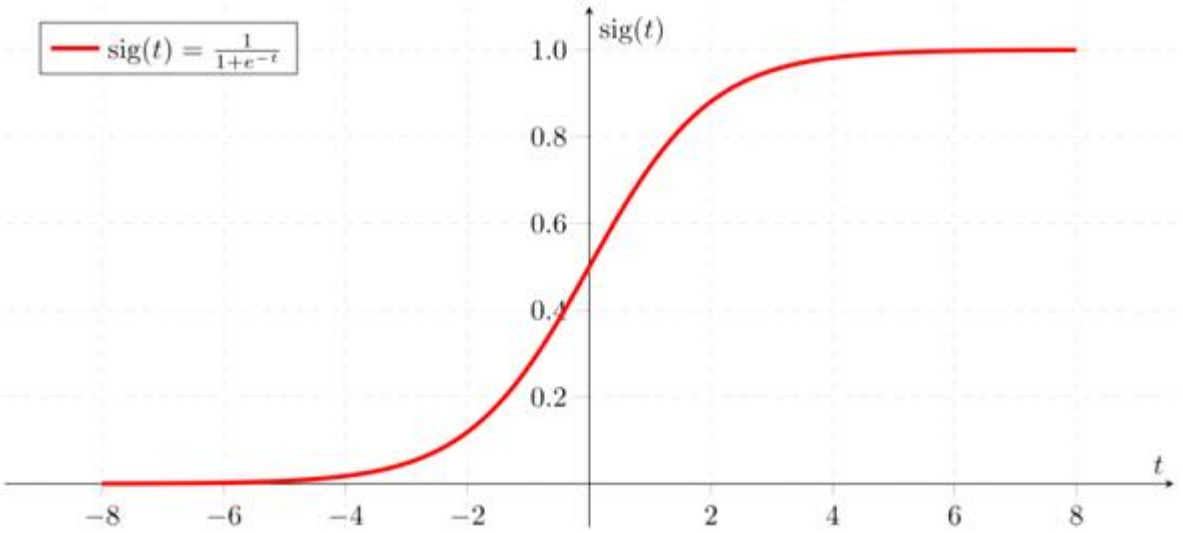


Fig. Sigmoid Function

- It maps any real value into another value within a range of 0 and 1. The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form.
- The S-form curve is called the Sigmoid function or the logistic function.
- Here, the concept of the threshold value is used, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.
- The logistic regression model transforms the linear regression function continuous value output into categorical value output using a sigmoid function, which maps any real-valued set of independent variables input into a value between 0 and 1. This function is known as the logistic function.
- Now we use the sigmoid function where the input will be z and we find the probability between 0 and 1. i.e predicted y.

$$\sigma(\square) = \frac{1}{1-\square}$$ where $x=e^{-z}$

Conclusion:

This lab provides students with a hands-on understanding of linear regression and logistic regression, covering essential concepts, practical implementation, and model evaluation. The exercises aim to reinforce theoretical knowledge with practical skills, preparing students for real-world data analysis and predictive modeling tasks.

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Useful Resource:

R Programming - Linear Regression

Linear regression using R programming

Logistic Regression in R, Clearly Explained!!!!

Exercise Questions

1. What are the dependent and independent variables in linear regression?
2. Attempt the following questions
    a. Write the equation of a simple linear regression model.
    b. What do the slope and intercept represent in the linear regression equation?
    c. How is the slope coefficient estimated in linear regression?
3. Attempt the following analysis questions
    a. What does a coefficient of determination (R-squared) value of 0.8 signify?
    b. If the Mean Squared Error (MSE) is 0, what does it indicate about the model?
4. Write the logistic regression equation.
5. What are common metrics used to evaluate the performance of a logistic regression model? Explain the Receiver Operating Characteristic (ROC) curve and its interpretation.

Experiment No 6

Experiment Title: Classification and clustering

Problem Statement:

Download any dataset from UCI or Data.org or any other data repositories and perform suitable data modeling operation as per the list given below, with respect to the dataset.

- k-nearest neighbors
- k-means clustering

Objective: Use packages in R to perform classification and clustering to understand the distribution of the data in dataset

Theory:

6.1 k-Nearest Neighbors

K-Nearest Neighbors (k-NN) algorithm is a popular supervised machine learning technique for classification and regression tasks. It is a type of instance-based learning where the prediction for a new data point is based on the majority class or average value of its k nearest neighbors.

The term "k" represents the number of neighbors considered in the prediction. k-NN is commonly used for both classification and regression tasks.

1. **Key Concepts:**
   - **Instance-Based Learning:** k-NN belongs to the category of instance-based learning, where the model memorizes the training instances and uses them for predictions.
   - **Decision Boundaries:** In classification, decision boundaries are formed based on the distribution of classes in the feature space.
2. **Distance Metrics:**
   - k-NN relies on distance metrics to determine the proximity between data points.
   - Common distance metrics include
     a. Euclidean distance,
     b. Manhattan distance, and
     c. Minkowski distance.
   - The choice of distance metric can impact the performance of the algorithm.

a) Euclidean Distance:

- **Formula:**
  - For two points (x_1, y_1) and (x_2, y_2) in a two-dimensional space:
  - Euclidean Distance = $\sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)}$
  - In general, for n-dimensional space: Euclidean Distance = $\sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2 + ... + (z_n - z_1)^2)}$
- **Interpretation:**
  - Represents the straight-line distance between two points in Euclidean space.
  - The shorter the Euclidean distance, the closer the points are to each other.

b) Manhattan Distance:

- **Formula:**
  - For two points (x_1, y_1) and (x_2, y_2) in a two-dimensional space:
  - Manhattan Distance = $|x_2 - x_1| + |y_2 - y_1|$

- In general, for n-dimensional space: Manhattan Distance = $|x_2 - x_1| + |y_2 - y_1| + ... + |z_n - z_1|$

- **Interpretation:**
  - Represents the distance between two points measured along the axes at right angles (like navigating through the streets of Manhattan).
  - Sum of absolute differences along each dimension.

c) Minkowski Distance:

- **Formula:**
  - For two points (x_1, y_1) and (x_2, y_2) in a two-dimensional space:
  - Minkowski Distance = $(|x_2 - x_1|^p + |y_2 - y_1|^p)^{(1/p)}$
  - In general, for n-dimensional space: Minkowski Distance = $(|x_2 - x_1|^p + |y_2 - y_1|^p + ... + |z_n - z_1|^p)^{(1/p)}$
- **Interpretation:**
  - Generalization of both Euclidean and Manhattan distances.
  - When p=2, it is equivalent to Euclidean distance; when p=1, it is equivalent to Manhattan distance.
  - Parameter p allows adjusting the sensitivity to different dimensions.

d) Comparison:

- **Euclidean Distance:**
  - Sensitive to magnitudes and scales in all dimensions.
  - Measures "as-the-crow-flies" distance.
- **Manhattan Distance:**
  - Less sensitive to outliers.
  - Suitable when movement is constrained to grid lines.
- **Minkowski Distance:**
  - Provides a general framework where Euclidean and Manhattan distances are special cases.
  - Allows adjusting the sensitivity to different dimensions through the parameter p.

In machine learning, the choice between these distances depends on the characteristics of the data and the problem at hand. Euclidean distance is common for continuous features, Manhattan distance for grid-based structures, and Minkowski distance offers flexibility.

3. Choosing the Value of k:

- The value of k significantly influences the model's performance.
- A small k can lead to noise sensitivity, while a large k may result in oversmoothing.
- Various methods, such as cross-validation, can be employed to determine the optimal k.

4. Handling Categorical Data:

- k-NN can handle categorical data by using appropriate distance metrics.
- One-hot encoding or label encoding may be necessary for categorical features.

5. Evaluation Metrics:

Common evaluation metrics for k-NN for regression tasks include

Accuracy:

- **Definition:**
  - Accuracy is the ratio of correctly predicted instances to the total instances in a classification problem.
  - Mathematically, Accuracy = (True Positives + True Negatives) / Total Predictions.
- **Interpretation:**
  - Represents the overall correctness of the model.
  - High accuracy suggests a reliable model, but it might not be sufficient for imbalanced datasets.

Precision:

- **Definition:**
  - Precision is the ratio of correctly predicted positive observations to the total predicted positives.
  - Mathematically, Precision = True Positives / (True Positives + False Positives).
- **Interpretation:**
  - Focuses on the accuracy of positive predictions.
  - High precision indicates a low false-positive rate.

Recall (Sensitivity or True Positive Rate):

- **Definition:**
  - Recall is the ratio of correctly predicted positive observations to all the actual positives.
  - Mathematically, Recall = True Positives / (True Positives + False Negatives).
- **Interpretation:**
  - Focuses on capturing all actual positive instances.
  - High recall indicates a low false-negative rate.

F1-Score:

- **Definition:**
  - F1-Score is the harmonic mean of precision and recall.
  - Mathematically, F1-Score = 2 * (Precision * Recall) / (Precision + Recall).
- **Interpretation:**
  - Balances precision and recall.
  - High F1-Score indicates a well-performing model, considering both false positives and false negatives.

Mean Squared Error (MSE):

- **Definition:**
  - MSE is a metric used in regression problems to measure the average squared difference between the predicted and actual values.
  - Mathematically, MSE = $(1/n) * \Sigma(y_i - \hat{y}_i)^2$ for all data points, where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of data points.
- **Interpretation:**
  - Quantifies the average magnitude of errors.

- Lower MSE values indicate better model performance.

Comparison:

- **Classification Metrics (Accuracy, Precision, Recall, F1-Score):**
  - Applicable to classification problems.
  - Evaluate the performance of models in predicting categories.
  - Trade-offs between precision and recall need to be considered based on the specific goals of the application.
- **Regression Metric (MSE):**
  - Applicable to regression problems.
  - Measures the accuracy of continuous predictions.
  - Lower MSE values signify better model accuracy, with zero indicating a perfect fit.

When selecting evaluation metrics, it's crucial to consider the nature of the problem and the specific goals of the model. Different metrics may be more appropriate depending on the application, and a balance between precision and recall is often necessary.

6.2 k-means Clustering

k-Means clustering is an unsupervised machine learning algorithm used for partitioning a dataset into distinct, non-overlapping groups or clusters. The goal is to group similar data points together and separate dissimilar ones. Each cluster is represented by a centroid, which is the average of all data points in that cluster. The algorithm iteratively refines the assignment of data points to clusters and adjusts the centroids to minimize the overall intra-cluster variance.

1. **Key Components:**
   1. **k (Number of Clusters):**
      - The parameter "k" represents the number of clusters the algorithm aims to identify in the dataset. It is a user-defined parameter, and choosing the right value for k is crucial.
   2. **Cluster Centroids:**
      - Each cluster is characterized by a centroid, which is the mean position of all data points in that cluster. Centroids serve as the representatives of their respective clusters.

2. Goal of k-Means Clustering:

The primary goal of k-Means clustering is to minimize the within-cluster sum of squares, often referred to as the **inertia** or **objective function**. The inertia is the sum of the squared distances between each data point and its assigned cluster centroid. Mathematically, for a dataset with nn data points and kk clusters:

Inertia=∑i=1n∑j=1k| | xi−μj| | 2Inertia=∑i=1n∑j=1k| | xi−μj| | 2

where xixi is a data point, μjμj is the centroid of the cluster to which xixi is assigned, and | | ⋅ | | | | ⋅ | | denotes the Euclidean distance.

3. Algorithm Workflow:

1. **Initialization:**
   - Select kk initial centroids, either randomly or through a more systematic method.
2. **Assignment:**
   - Assign each data point to the nearest centroid, forming kk clusters.
3. **Update Centroids:**

   - Recalculate the centroids based on the mean position of the data points in each cluster.
4. **Iteration:**
   - Repeat the assignment and centroid update steps until convergence, where the assignments no longer change significantly.
5. **Final Clusters:**
   - The algorithm produces kk clusters, each represented by its centroid.

4. Minimizing Inertia: By assigning data points to clusters and updating centroids iteratively, the algorithm minimizes the inertia, leading to clusters with tight internal cohesion and clear separation between clusters. The final result is a partitioning of the data into groups where points within the same group are more similar to each other than to those in other groups.

5. Steps of the k-Means Algorithm:

- The steps involved in the k-Means algorithm are:
  1. **Initialization:** Select k initial centroids.
  2. **Assignment:** Assign each data point to the nearest centroid.
  3. **Update Centroids:** Recalculate the centroids based on the assigned data points.
  4. **Iteration:** Repeat the assignment and centroid update steps until convergence.

6. Choosing the Number of Clusters (k):

Several methods can be employed to determine the optimal number of clusters, and here are some common approaches: **Elbow Method, Silhouette Score,Gap Statistics and Davies-Bouldin Index**

- **Elbow Method:**
  - **Concept:**
    - The Elbow Method involves running the k-Means clustering algorithm on the dataset for a range of values of k and plotting the sum of squared distances from each point to its assigned center (inertia) against k.
  - **Procedure:**
    - Execute k-Means clustering for a range of k values.
    - Compute the inertia for each k.
    - Plot the inertia values against the corresponding k values.
    - Identify the "elbow" point on the plot where the rate of decrease in inertia slows down.
  - **Interpretation:**
    - The "elbow" is a point where adding more clusters does not significantly reduce the inertia, suggesting diminishing returns in terms of clustering improvement.

7. Distance metrics play a fundamental role in the k-Means clustering algorithm, influencing how data points are assigned to clusters and how centroids are updated during the iterative process. The choice of distance metric affects the shape, size, and characteristics of the clusters formed. Common distance metrics used in k-Means clustering include Euclidean distance, Manhattan distance, and other variations.

8. Handling categorical data in k-Means clustering poses challenges because the algorithm relies on distance metrics, and categorical variables do not have a natural notion of distance.

- **Challenges:** No Inherent Distance Measure, Sensitivity to Encoding, Curse of Dimensionality
- **Strategies:** Ordinal Encoding, Target Encoding, Frequency-Based Encoding, Embedding Techniques, K-Prototypes Algorithm, Hybrid Approaches, Dimensionality Reduction, Feature Engineering

Conclusion:

This lab provides students with hands-on experience in implementing and understanding the k-Nearest Neighbors and k-means clustering algorithms. It covers fundamental concepts, practical aspects, and considerations for optimal model performance. Students will gain insights into the importance of parameter tuning, distance metrics, and the impact of k on the algorithm's behavior. By focusing on the theoretical aspects, this lab equips students with the foundational knowledge necessary for successful implementation and application of the algorithm in practical scenarios.

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Useful Resource:

K-nearest neighbor (KNN) Algorithm in Machine Learning using R Programming
K-Nearest Neighbors (KNN) with R | Classification and Regression Examples
Machine Learning for Beginners :: Session 9 - K-means Clustering
K Means Clustering Algorithm | K Means Solved Numerical Example Euclidean Distance by Mahesh Huddar

Exercise Questions

1. How does KNN handle the training data during the prediction phase?
2. Explain how a small value of "k" might lead to overfitting in KNN.
3. Define the "curse of dimensionality" and its relevance to KNN.
4. Why is selecting the appropriate number of clusters ("k") crucial in k-Means clustering?
5. In what scenarios is feature scaling particularly important in k-Means?
6. How does the centroid update step contribute to the convergence of the algorithm?

Experiment No 7

Experiment Title: Tableau

Problem Statement: To do installation of Tableau

Objective:

To acquire hands-on experience in setting up Tableau.

Theory:

Tableau is a powerful and widely used data visualization and business intelligence software that helps people see and understand their data. It allows users to connect to various data sources, create interactive and shareable dashboards, and gain insights from their data through visually appealing charts and graphs.

Tableau is known for its user-friendly interface and ability to handle large datasets, making it a popular choice for analysts, data scientists, and business professionals.

Here's a brief introduction to Tableau and steps for its installation:

Introduction to Tableau:

Data Connection:

- Tableau can connect to various data sources such as Excel, SQL databases, cloud-based sources like Google Analytics, and many others.
- It allows users to import, clean, and transform data for analysis.

Visualization:

- Tableau excels in creating interactive and dynamic visualizations like charts, graphs, maps, and dashboards.
- Users can drag and drop fields to create visualizations without the need for complex coding.

Dashboards:

- Dashboards in Tableau allow users to combine multiple visualizations into a single interactive view.
- Filters and actions can be applied to enable dynamic exploration of data.

Sharing and Collaboration:

- Tableau supports sharing of interactive dashboards and reports with others.
- The Tableau Server and Tableau Online platforms allow for collaboration and real-time updates.

Tableau Installation:

System Requirements:

- Ensure that your system meets the minimum requirements for Tableau installation. These requirements can be found on the official Tableau website.

Download Tableau:

- Visit the official Tableau website (https://www.tableau.com/) and navigate to the "Products" section.
- Choose the version of Tableau that suits your needs (Tableau Desktop, Tableau Server, or Tableau Online).

Installation Steps:

- Follow the installation instructions provided by Tableau during the download process.
- Provide the necessary information and select the installation location.
- Complete the installation process.

Activation:

- After installation, you will need to activate Tableau using your Tableau account.
- If you don't have an account, you may need to sign up for one.

License Key:

- Depending on your Tableau version, you may need to enter a license key during the activation process.

Getting Started:

- Once activated, launch Tableau and start exploring your data.
- Connect to your data source, create visualizations, and build dashboards.

Conclusion:

Tableau is a versatile tool that empowers users to gain insights from their data through intuitive and visually appealing analytics. Whether you're a data analyst, business professional, or a decision-maker, Tableau can help you make sense of your data and communicate your findings effectively.

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Useful Resource:

Exercise Questions

Experiment No 8

Experiment Title:

Download any dataset from UCI or Data.org or any other data repositories and perform
• Connecting to data and preparing data for visualization in Tableau.
• Data Aggregation and Statistical functions in Tableau
• Data Visualizations in Tableau
• Basic Dashboards in Tableau

Objective:

Students should be in a position to understand Tableau and be able to create visualizations and create dashboards.

Theory:

The Data

This project's data set is maintained transparently with the Creative Commons 4.0 license by Fernando Silva through the Mendeley data repository. The data set consists of roughly 180k transactions from supply chains used by the company DataCo Global for three years. The data set can be downloaded for a supply chain company. Making sure orders are delivered on time and preventing fraud is essential.

Data Cleaning & Modelling

The data set consists of 52 variables. All the variables that are not needed are dropped, and outliners are removed. Out of the remaining 40 variables, optimal variables to predict fraud and late deliveries are selected using forward selection. The optimal variables for predicting late deliveries are the Order Country and shipment days scheduled for the order. And for predicting fraud, the optimal variables are real days taken to ship the order and shipment days scheduled along with the order country. Common optimal variables are selected for ease of building dashboard.

Deploying the models to the Tableau

Python 3 and Anaconda should be installed on the device. If you don't have them installed, you can install them directly from their websites Anaconda and Python, for free. To connect the python environment with Tableau, there is an excellent library called Tabpy, which we need to install first. TabPy runs in an anaconda environment and is very easy to install. To install TabPy, open the command prompt and run the following command pip install tabpy

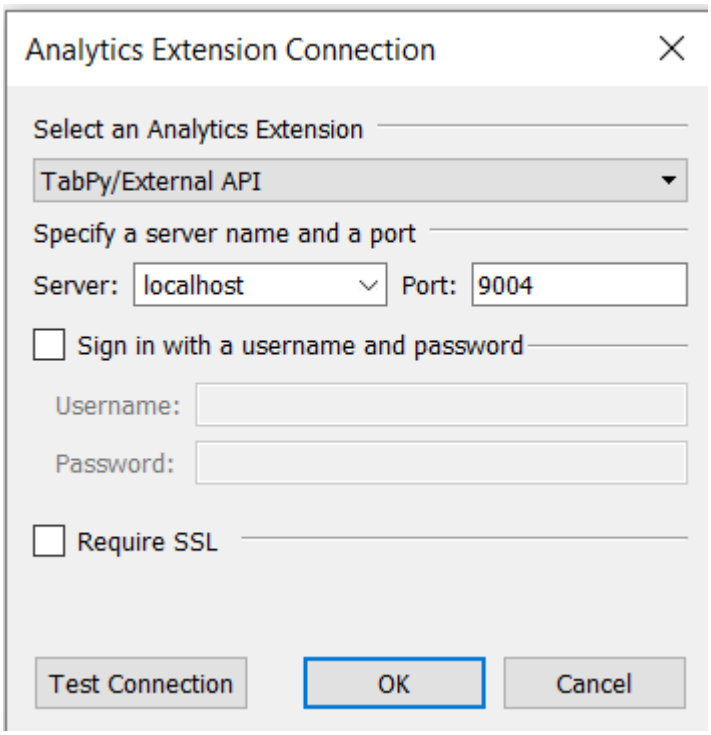
Once tabpy is installed. We also need an extra library called tabpy_client to connect the Jupyter notebook environment to tableau and deploy machine learning models directly. Even this library can be installed like above using the following command pip install tabpy_client

Once tabpy is installed successfully, you can start the tabpy environment by running tabpy

After running this, you should see a message saying the tabpy server started successfully, and the port is set to '9004' something similar to the image shown below.

Connecting the TabPy server to Tableau

After successfully installing the TabPy library, you can easily connect the tabpy server to Tableau in Tableau application settings Help > Settings and Performance > Manage Analytics Extension Connection in Tableau Desktop. Select the localhost in the server and enter the port number.



Click on 'Test Connection' to see if your server connection is successful. If the connection is successful. Then you can start deploying your machine learning models directly to Tableau.

1. To deploy models into the tableau, start by connecting the notebook to the Tableau server using tabpy_client.
2. Individual functions should be defined in a format suitable to Tableau. New variables as _arg1,_arg2 are declared with respect to the count of variables in the trained machine learning model.
3. Suppose any of the variables we are using in the dashboard are in the string. They should be converted into numeric format.

Conclusion:

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Sample Code

```
import pandas as pd
```

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import preprocessing
dataset=pd.read_csv("DataCoSupplyChainDataset.csv",header=
0,encoding= 'unicode_escape')


# Adding first name and last name together to create new
column
dataset['Customer Full Name'] = dataset['Customer
Fname'].astype(str)+dataset['Customer Lname'].astype(str)
data=dataset.drop(['Customer Email','Product
Status','Customer Password','Customer Street','Customer
Fname','Customer Lname',
          'Latitude','Longitude','Product
Description','Product Image','Order Zipcode','shipping date
(DateOrders)'],axis=1)
data['Customer Zipcode']=data['Customer Zipcode'].fillna(0)
def outlier_treatment(datacolumn):
sorted(datacolumn)
Q1,Q3 = np.percentile(datacolumn , [25,75])
IQR = Q3 - Q1
lower_range = Q1 - (1.5 * IQR)
upper_range = Q3 + (1.5 * IQR)
return lower_range,upper_range
lower_range,upper_range=outlier_treatment(data['Product
Price'])
data.drop(data[(data['Product Price'] < lower_range) |
(data['Product Price'] > upper_range)].index ,
inplace=True)
train_data=data.copy()
train_data['fraud'] = np.where(train_data['Order Status']
== 'SUSPECTED_FRAUD', 1, 0)
train_data['late_delivery']=np.where(train_data['Delivery
Status'] == 'Late delivery', 1, 0)
#Dropping columns with repeated values
train_data.drop(['Delivery
Status','Late_delivery_risk','Order Status', 'order date
(DateOrders)'], axis=1, inplace=True)
```

```
le = preprocessing.LabelEncoder()
train_data['Order Country']  =
le.fit_transform(train_data['Order Country'])
train_data['Order State']    =
le.fit_transform(train_data['Order State'])


#Fraud Prediction
Xf=train_data[['Days for shipping (real)','Days for
shipment (scheduled)','Order Country']]
yf=train_data['fraud']
train_x,test_x,train_y,test_y =
train_test_split(Xf,yf,test_size = 0.2, random_state = 42)
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(train_x, train_y.values.ravel())
random_forest.score(train_x, train_y)


#Late Delivery Prediction
Xl=train_data[['Days for shipment (scheduled)','Order
Country']]
yl=train_data['late_delivery']
train_xl,test_xl,train_yl,test_yl =
train_test_split(Xl,yl,test_size = 0.2, random_state = 42)
random_forest_l = RandomForestClassifier(n_estimators=100)
random_forest_l.fit(train_xl, train_yl.values.ravel())
random_forest_l.score(train_xl, train_yl)
```

Connecting the TabPy server to Tableau

```
import tabpy_client

from tabpy.tabpy_tools.client import Client
client = tabpy_client.Client('http://localhost:9004/')


def fraud_predictor5( _arg1, _arg2,_arg3):
    import pandas as pd
    row = {'shipping': _arg1,
```

```
            'shipping scheduled': _arg2,
           'country_str':_arg3}
    #Convert it into a dataframe
    test_data = pd.DataFrame(data = row,index=[0])
    from sklearn import preprocessing
    le = preprocessing.LabelEncoder()
    test_data['country_str']  =
le.fit_transform(test_data['country_str'])
    #Predict the Fraud
    predprob_survival = random_forest.predict_proba(test_data)
    #Return only the probability
    return [probability[1] for probability in predprob_survival]

def late_delivery( _arg1, _arg2):
    import pandas as pd
    row = {'shipping scheduled': _arg1,
          'country_str':_arg2}
    #Convert it into a dataframe
    test_data = pd.DataFrame(data = row,index=[0])
    from sklearn import preprocessing
    le = preprocessing.LabelEncoder()
    test_data['country_str']  =
le.fit_transform(test_data['country_str'])
    #Predict the late delivery probabilites
    predprob_late = random_forest_l.predict_proba(test_data)
    #Return only the probability
    return [probability[1] for probability in predprob_late]


#Deploying
client.deploy('fraud_predictor5', fraud_predictor5,'fraud_predictor
probability',override = True)
client.deploy('late_delivery',
late_delivery,'late_delivery_prop',override = True)
```

Useful Resource:

Exercise Questions

Experiment No 9

Experiment Title: PowerBI

Problem Statement: To do installation of PowerBI

Objective:
To acquire hands-on experience in setting up Tableau.

Theory:

Power BI, developed by Microsoft, is a business analytics service that provides interactive visualizations and business intelligence capabilities with an interface simple enough for end users to create their own reports and dashboards. It's widely used for data analysis, sharing insights across an organization, and making informed business decisions. Power BI integrates with a variety of data sources and offers a user-friendly environment for data preparation and visualization.

Here's a brief introduction to Power BI along with steps for its installation:

Introduction to Power BI:

Data Connection:

- Power BI allows users to connect to a wide range of data sources, including Excel, databases, cloud-based services (like Azure and AWS), and many others.
- Users can import, transform, and clean data using Power Query, a part of the Power BI suite.

Data Modeling:

- Power BI includes a robust data modeling engine that enables users to create relationships between different data tables.
- Users can define measures, calculated columns, and hierarchies to enhance data analysis.

Visualization:

- The core strength of Power BI lies in its interactive and customizable visualizations. Users can create a variety of charts, maps, tables, and other visual elements.
- Visualizations can be customized and arranged on a canvas to create comprehensive dashboards.

Dashboards:

- Power BI dashboards allow users to combine multiple visualizations and reports into a unified and interactive view.
- Dashboards can be shared and accessed across the organization for collaborative decision-making.

Power Query and Power Pivot:

- Power Query is used for data transformation and shaping.
- Power Pivot allows users to create data models and perform advanced data analysis.

Power BI Installation:

Download Power BI Desktop:

- Visit the official Power BI website (https://powerbi.microsoft.com/) and navigate to the "Downloads" section.
- Download the Power BI Desktop application.

Installation Steps:

- Run the installer and follow the installation wizard.
- Choose the appropriate options for your installation, including the installation location.

Launching Power BI Desktop:

- Once installed, launch Power BI Desktop.
- You can sign in with your Microsoft account or use Power BI without signing in for basic functionality.

Getting Started:

- Connect to your data source by importing data or connecting to an existing dataset.
- Use the Power BI Desktop interface to create visualizations, design reports, and build dashboards.

Power BI Service:

- In addition to Power BI Desktop, you can also explore the Power BI service (https://app.powerbi.com/) for cloud-based collaboration, sharing, and publishing of reports and dashboards.

Conclusion:

Power BI is a powerful tool for business intelligence, enabling users to transform data into actionable insights through intuitive visualizations and analytics. The combination of Power BI Desktop for report creation and the Power BI service for collaboration provides a comprehensive solution for organizations seeking to harness the full potential of their data.

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Useful Resource:

Exercise Questions

Experiment No 10

Experiment Title:

Download any dataset from UCI or Data.org or any other data repositories and perform

• Connecting to data and preparing data for visualization in PowerBI.

• Data Aggregation and Statistical functions in PowerBI

• Data Visualizations in PowerBI

• Basic Dashboards in PowerBI

Objective:

Students should be in a position to understand PowerBIand be able to create visualizations and create dashboards.

Theory:

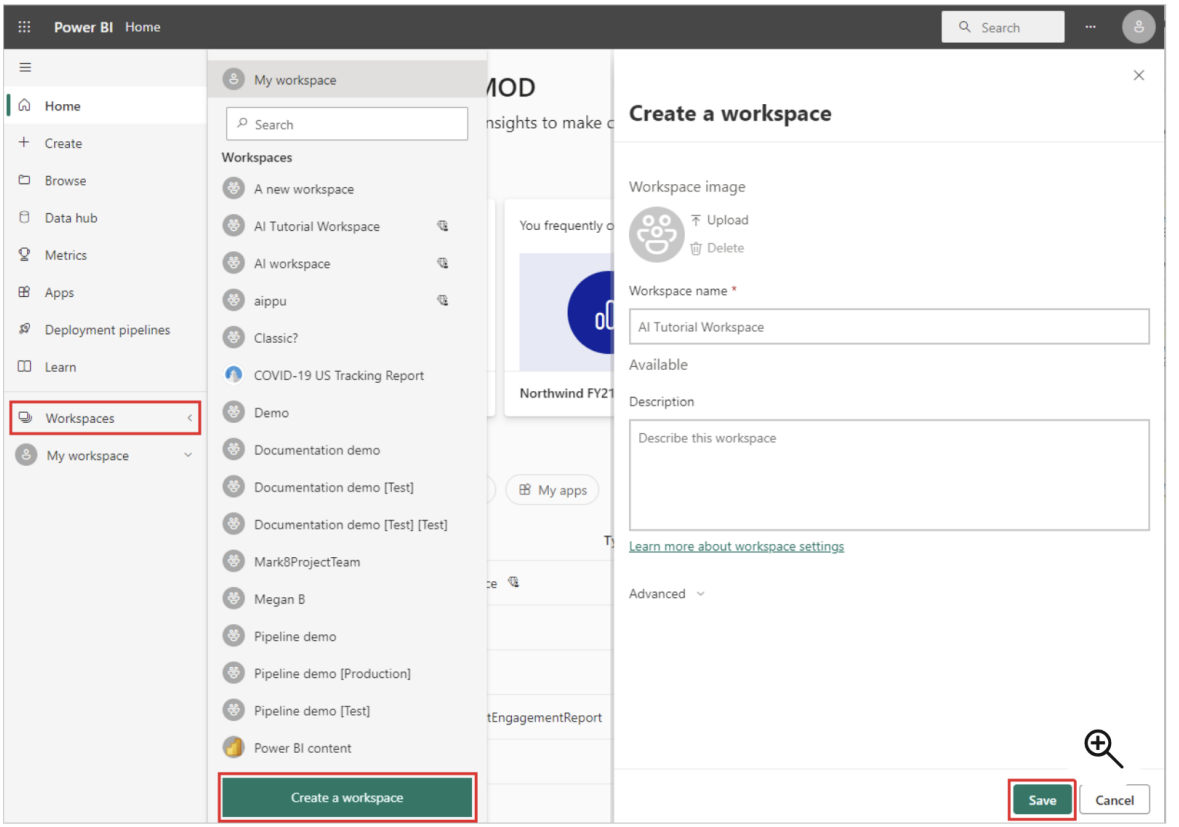
Data

You can download the semantic model from the UC Irvine website or by downloading the online_shoppers_intention.csv

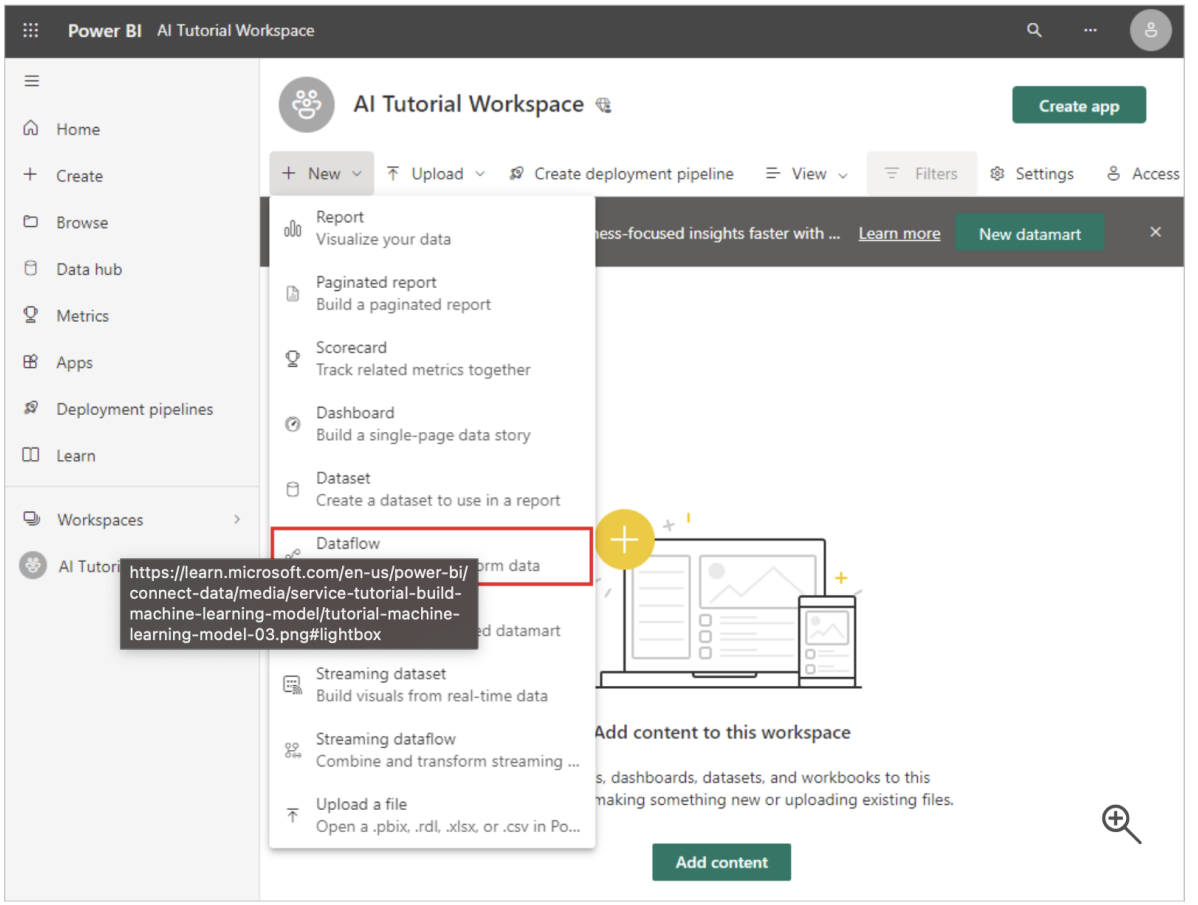
Create the tables

To create the entities in your dataflow, sign into the Power BI service and navigate to a workspace.

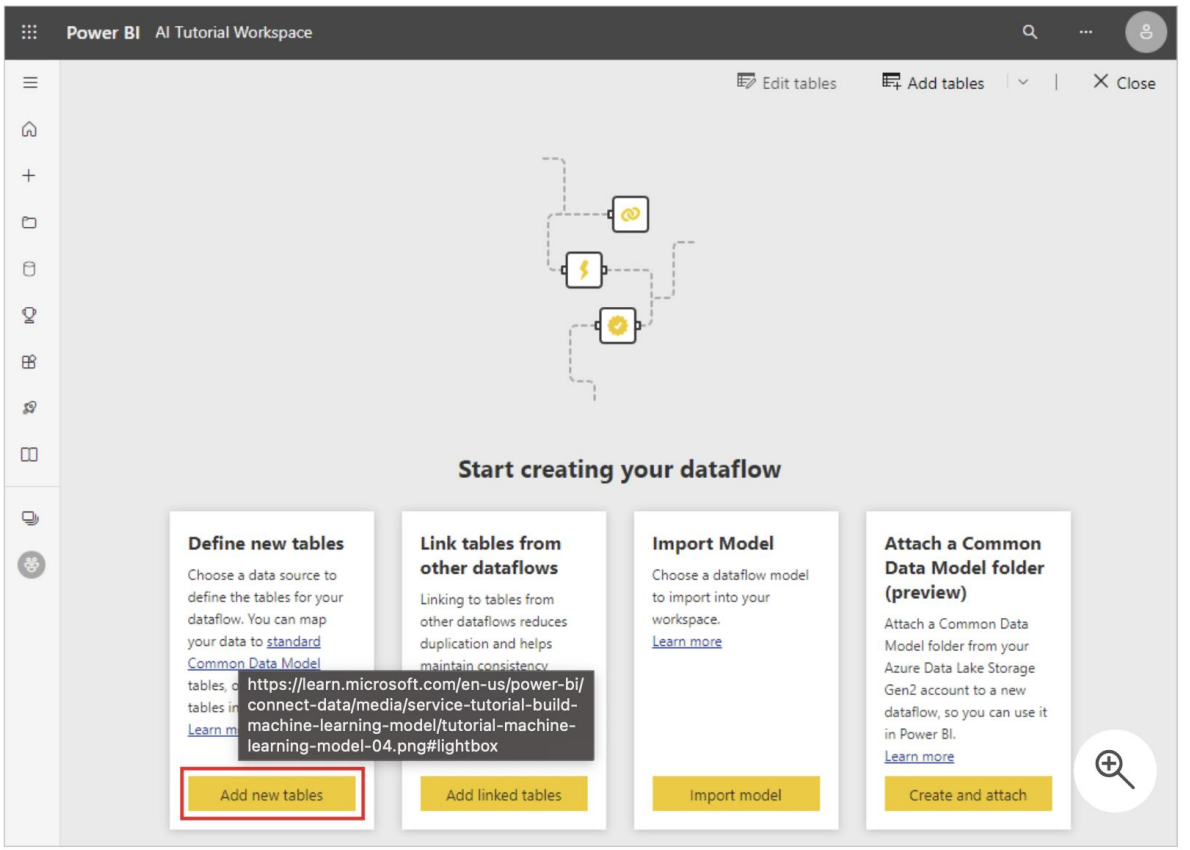
1. If you don't have a workspace, create one by selecting Workspaces in the Power BI left navigation pane and selecting Create a workspace. In the Create a workspace panel, enter a workspace name and select Save.



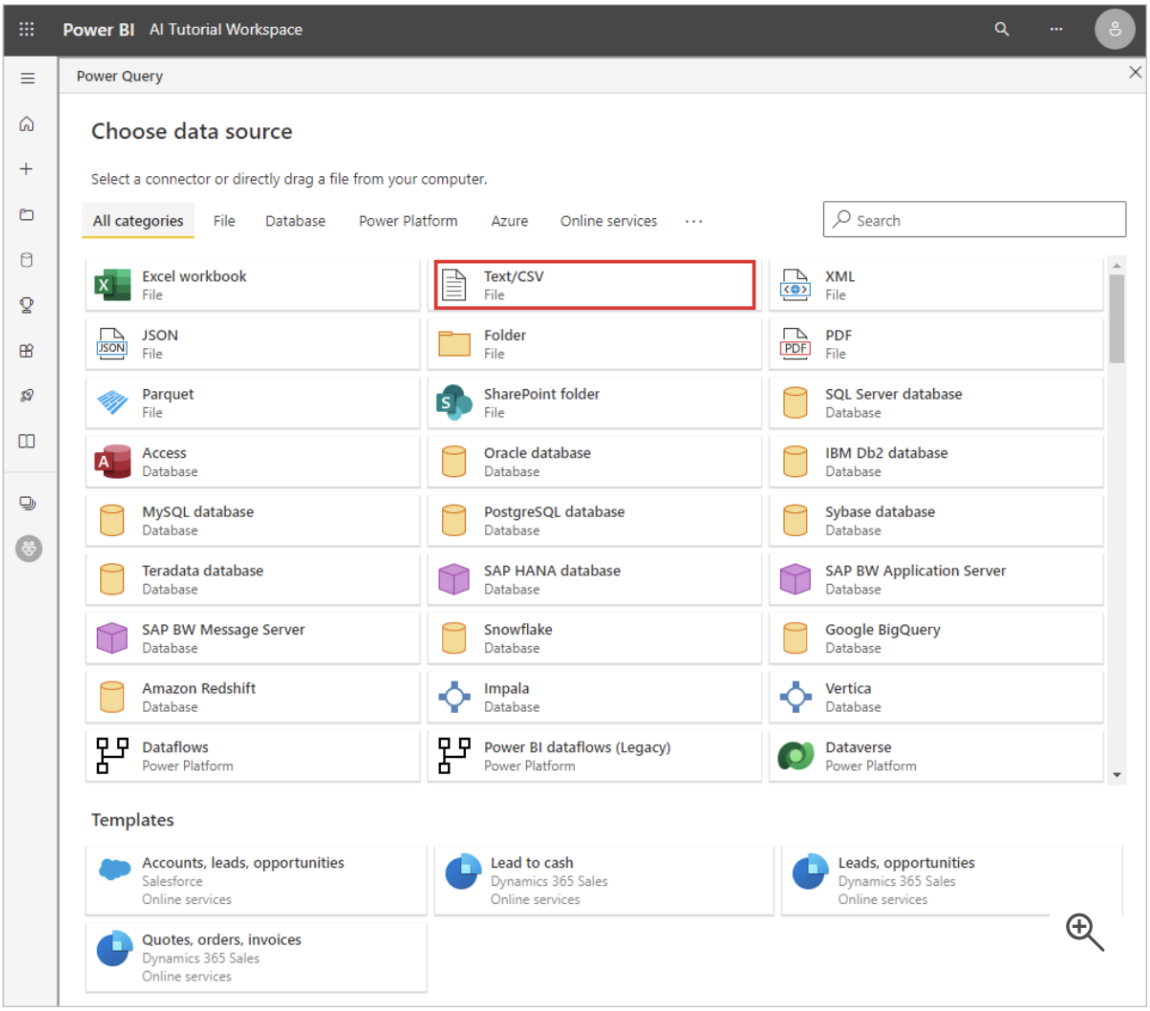
2. Select New at the top of the new workspace, and then select Dataflow.



3. Select Add new tables to launch a Power Query editor in the browser.



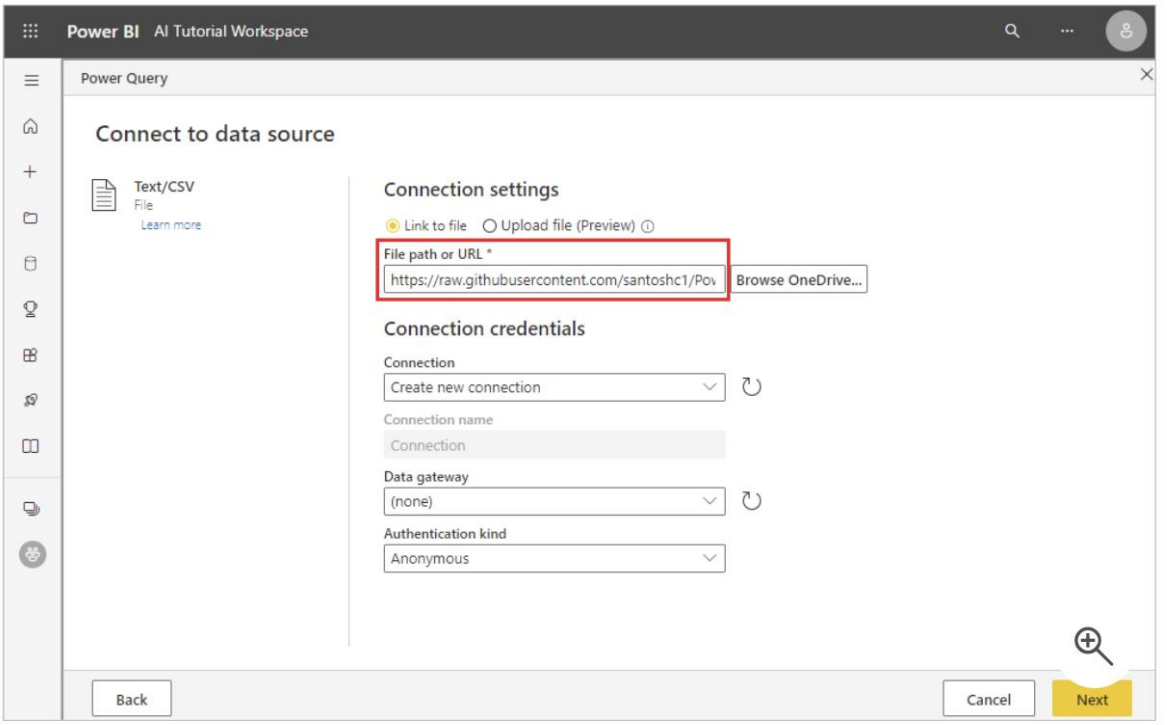
4. On the Choose data source screen, select Text/CSV as the data source.



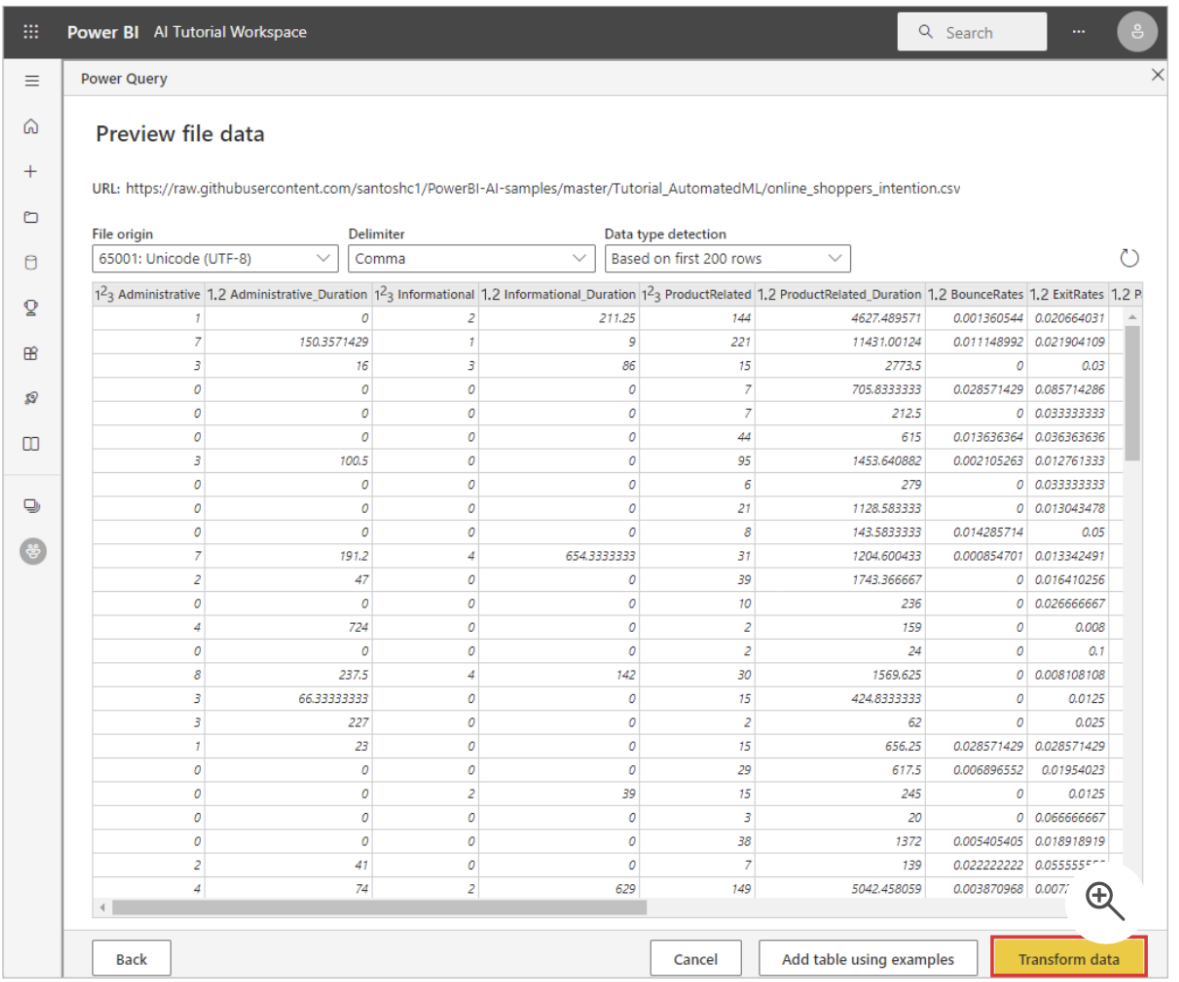
5. On the Connect to a data source page, paste the following link to the *online_shoppers_intention.csv* file into the File path or URL box, and then select Next.

https://raw.githubusercontent.com/santoshc1/PowerBI-AI-

samples/master/Tutorial_AutomatedML/online_shoppers_intention.csv

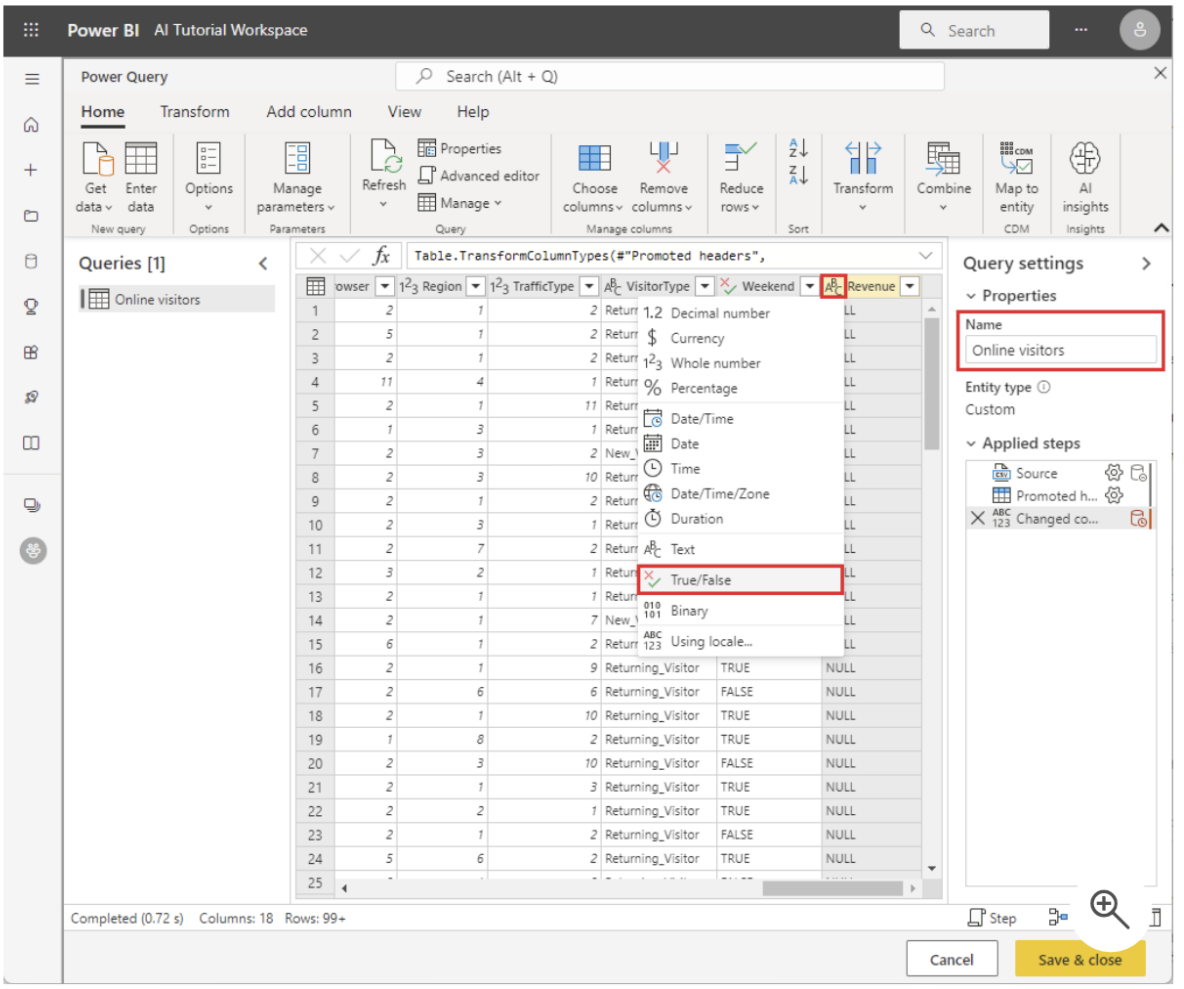


6. The Power Query Editor shows a preview of the data from the CSV file. To make changes in the data before loading it, select Transform data.

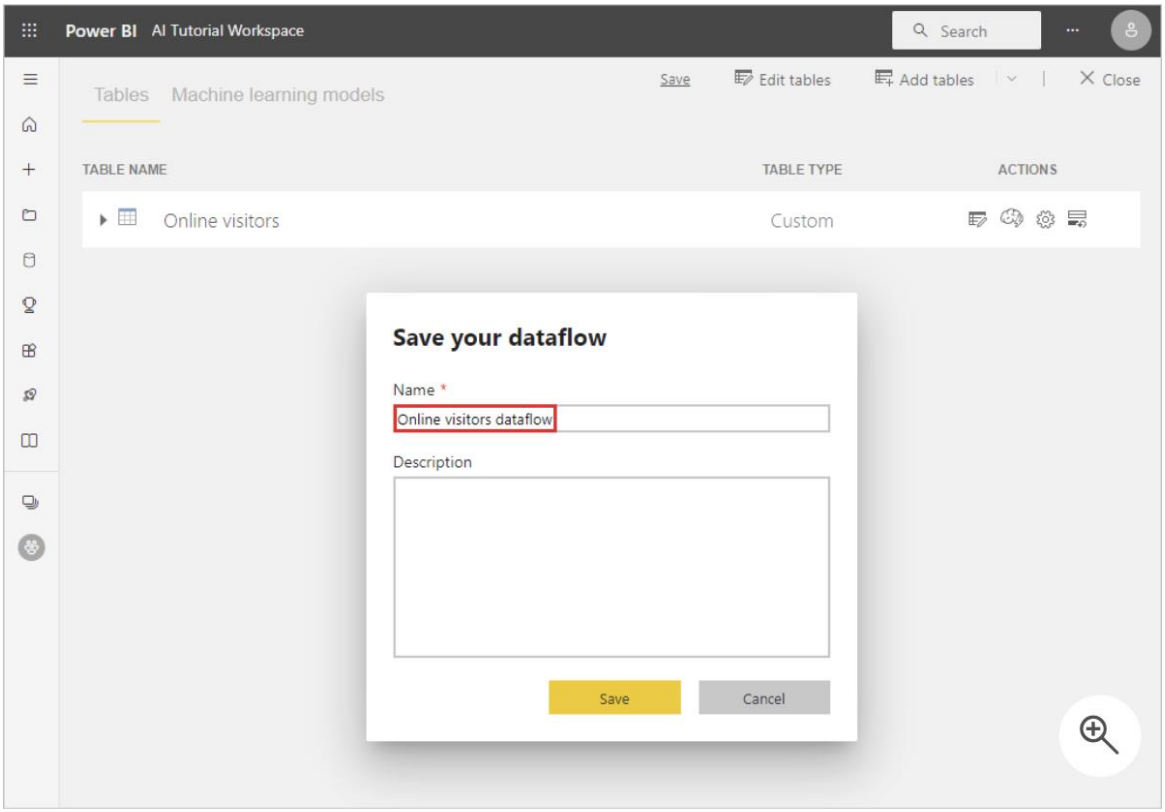


7. Power Query automatically infers the data types of the columns. You can

change the data types by selecting the attribute type icon at the tops of the column headers. Change the type of the Revenue column to True/False.You can rename the query to a friendlier name by changing the value in theName box in the right pane. Change the query name to *Online visitors*.

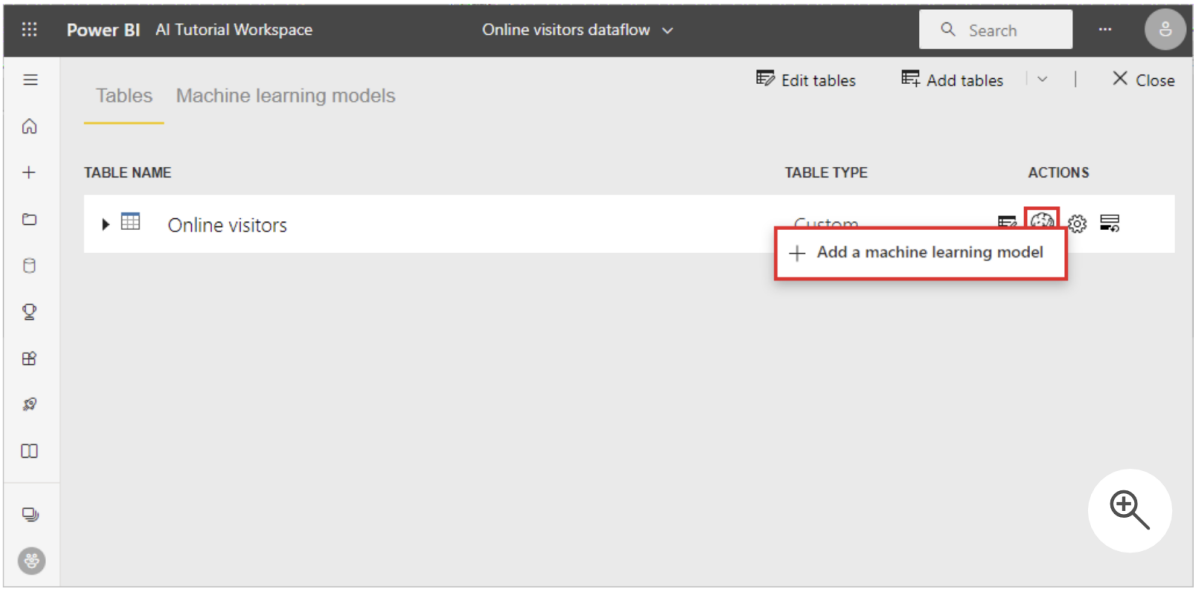


8. Select Save & close, and in the dialog box, provide a name for the dataflow and then select Save.

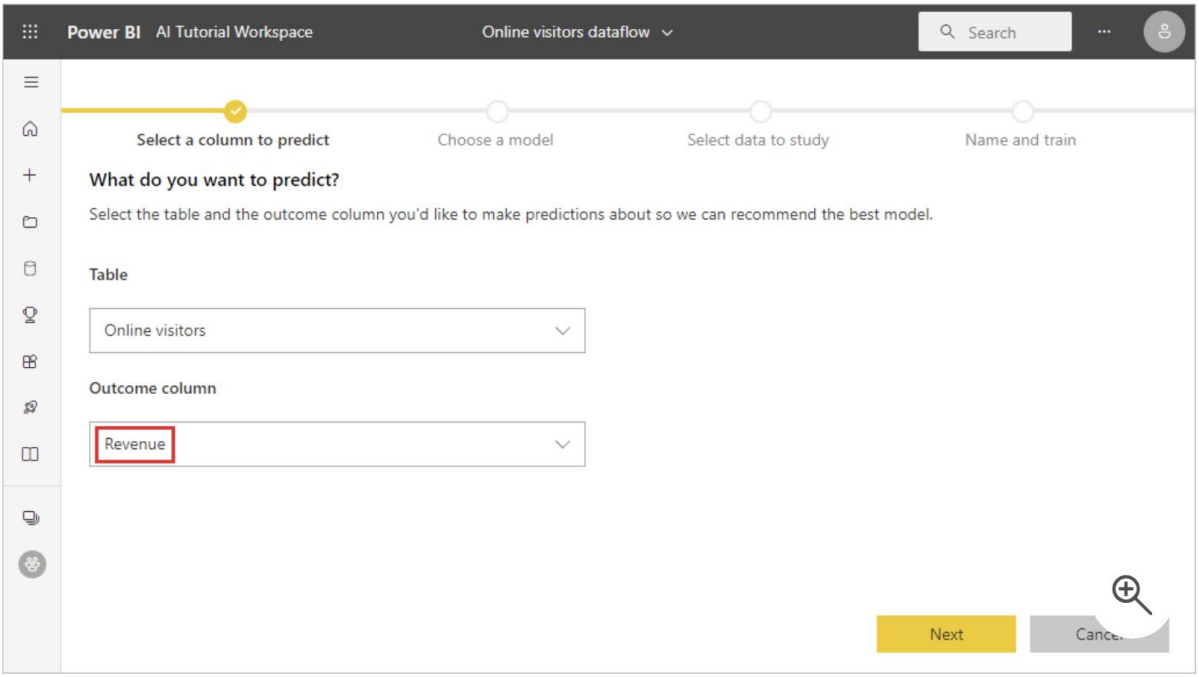


Create and train a machine learning model
To add a machine learning model:

1. Select the Apply ML model icon in the Actions list for the table that contains your training data and label information, and then select Add a machine learning model.

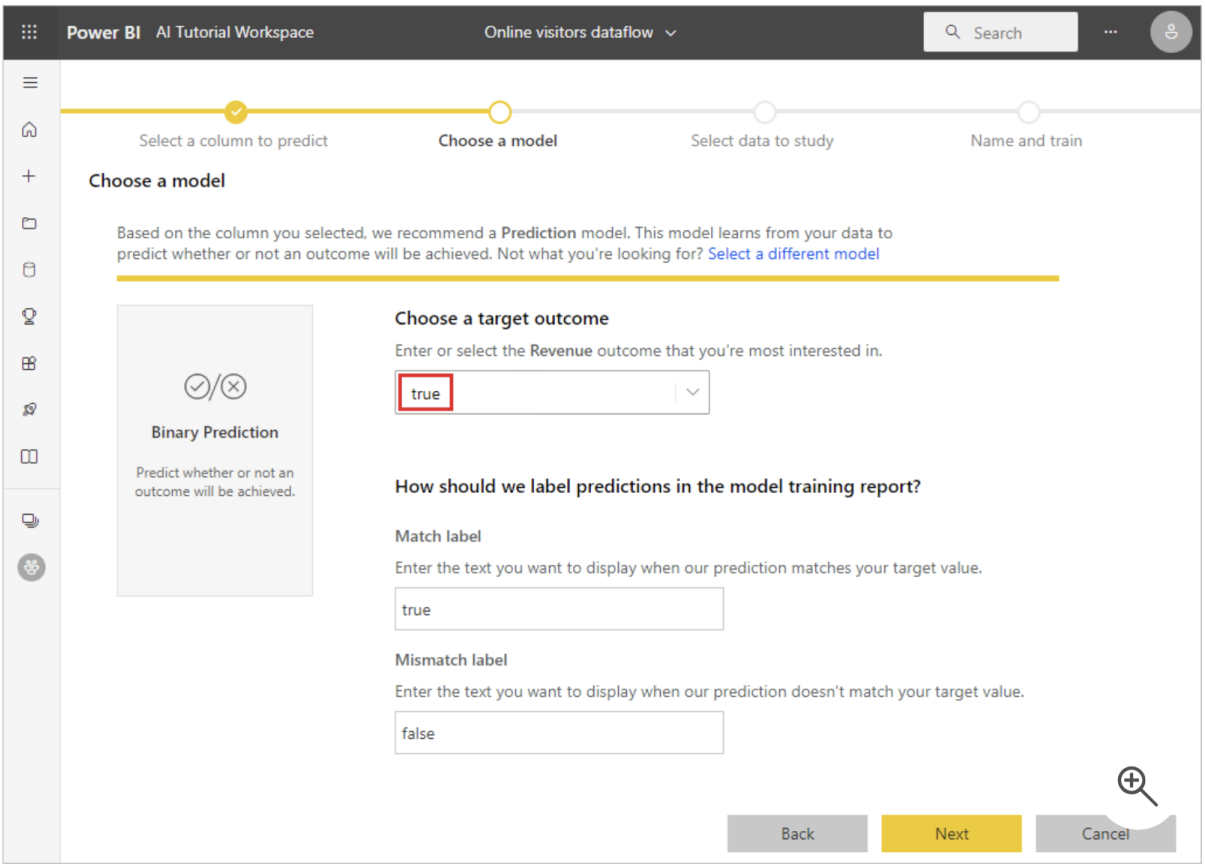


2. The first step to create your machine learning model is to identify the historical data, including the outcome field that you want to predict. The model is created by learning from this data. In this case, you want to predict whether or not visitors are going to make a purchase. The outcome you want to predict is in the Revenue field. Select Revenue as the Outcome column value, and then select Next.

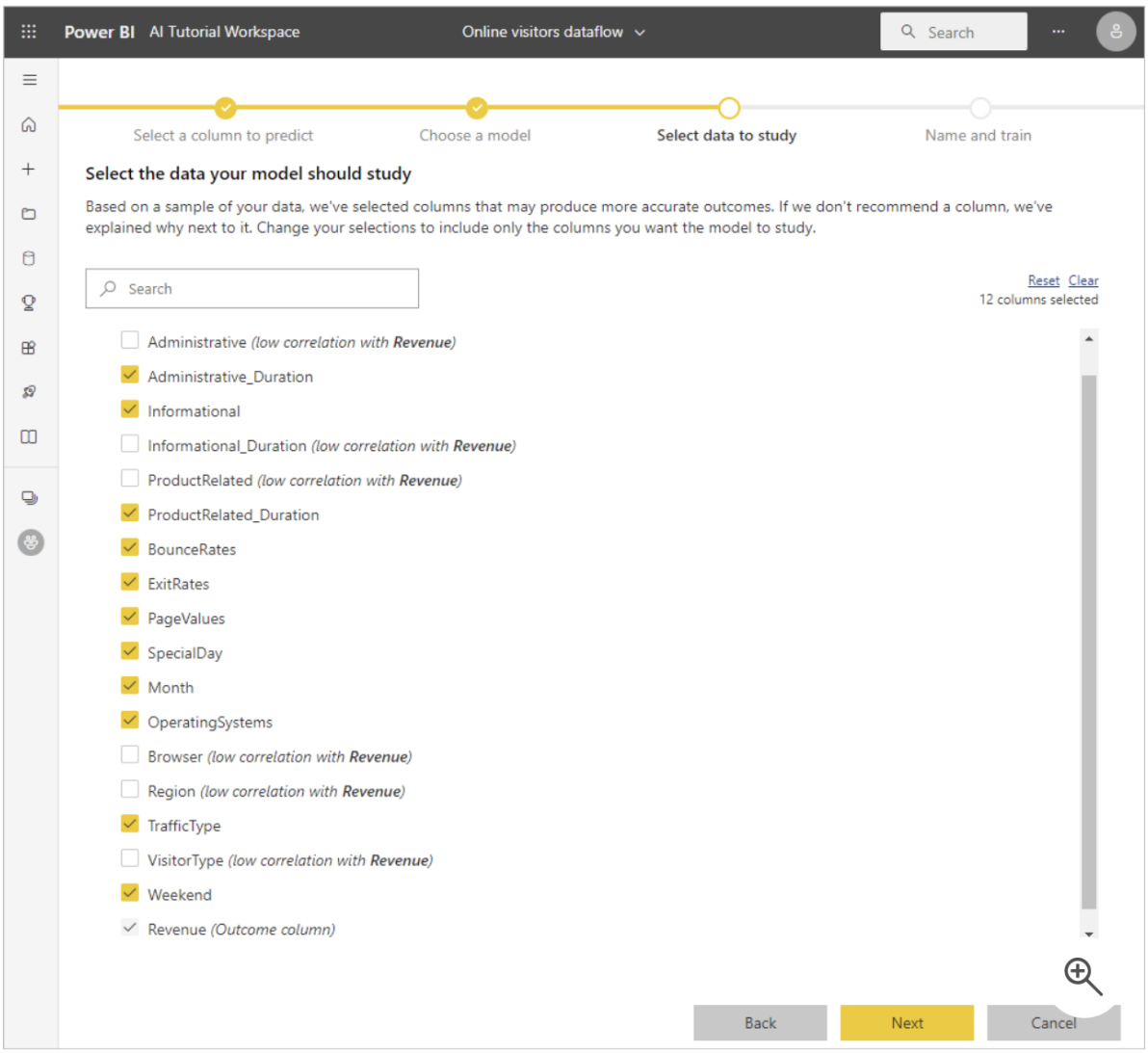


3. Next, you select the type of machine learning model to create. Power BI analyzes the values in the outcome field that you identified, and suggests the types of machine learning models that it can create to predict that field.

In this case, since you want to predict a binary outcome of whether or not a visitor is going to make a purchase, Power BI recommends Binary Prediction. Because you're interested in predicting visitors who are going to make a purchase, select true under Choose a target outcome. You can also provide different labels to use for the outcomes in the automatically generated report that summarizes the model validation results. Then select Next.



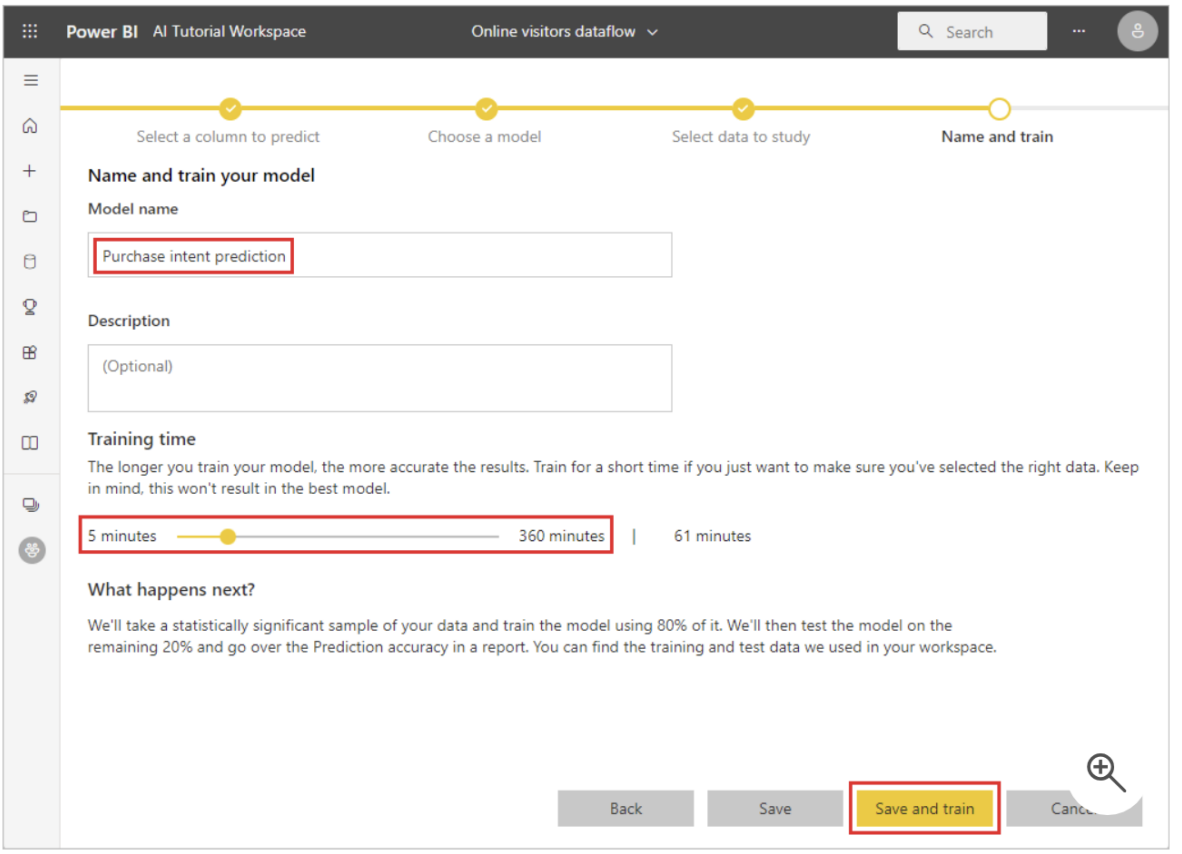
4. Power BI does a preliminary scan of a sample of your data and suggests inputs that might produce more accurate predictions. If Power BI doesn't recommend a column, it explains why not next to the column. You can change the selections to include only the fields you want the model to study by selecting or deselecting the checkboxes next to column names. Select Next to accept the inputs.

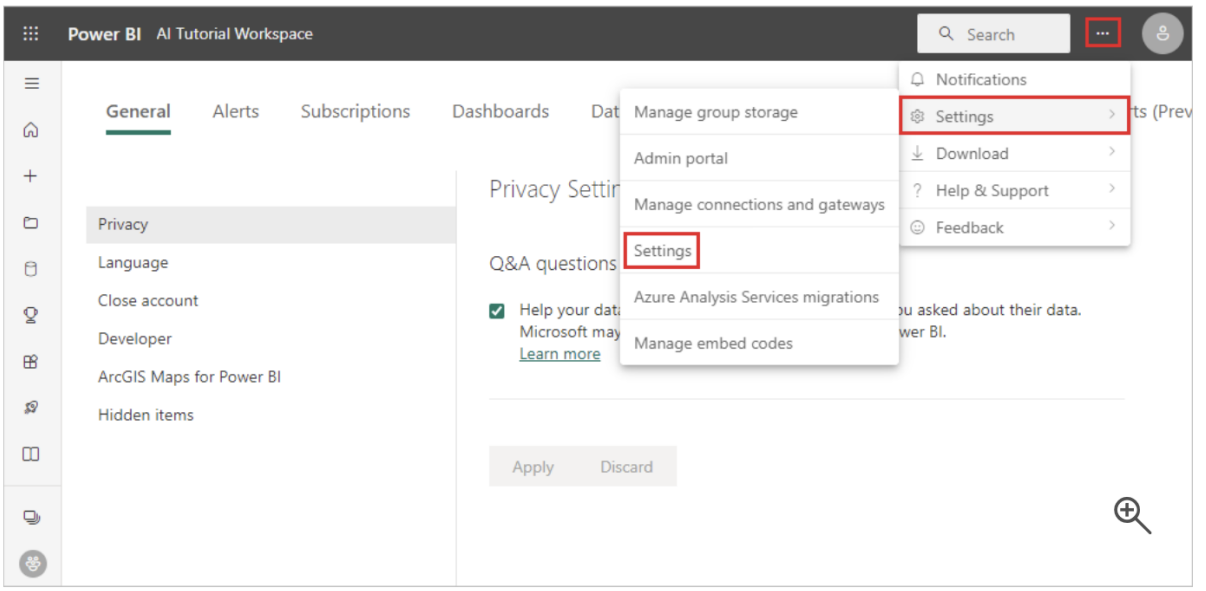


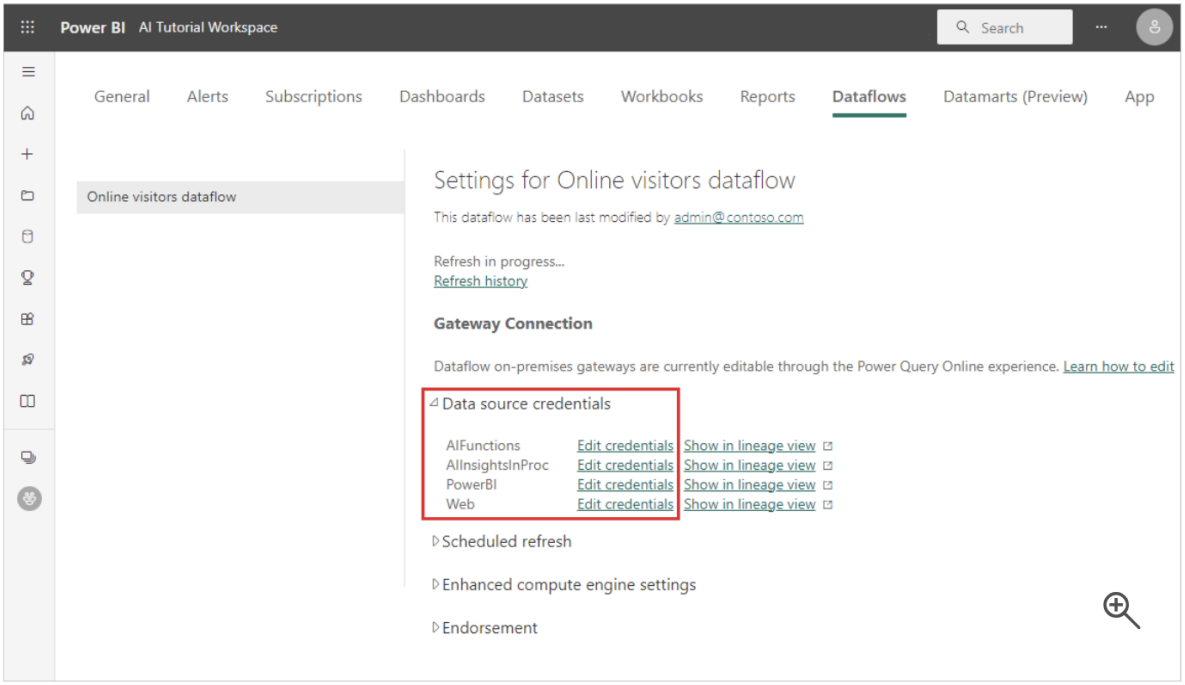
5. In the final step, name the model *Purchase intent prediction*, and choose the amount of time to spend in training. You can reduce the training time to see quick results or increase the time to get the best model. Then select Save and train to start training the model.

If you get an error similar to Credentials not found for data source, you need to update your credentials so Power BI can score the data. To update your credentials, select More options ... in the header bar and then select Settings > Settings.

Select your dataflow under Dataflows, expand Data source credentials, and then select Edit credentials.


Power BI
AI Tutorial Workspace
Online visitors dataflow
Search
Select a column to predict
Choose a model
Select data to study
Name and train
Name and train your model
Model name
Purchase intent prediction
Description
(Optional)
Training time
The longer you train your model, the more accurate the results. Train for a short time if you just want to make sure you've selected the right data. Keep in mind, this won't result in the best model.
5 minutes
360 minutes
61 minutes
What happens next?
We'll take a statistically significant sample of your data and train the model using 80% of it. We'll then test the model on the remaining 20% and go over the Prediction accuracy in a report. You can find the training and test data we used in your workspace.
Back
Save
Save and train

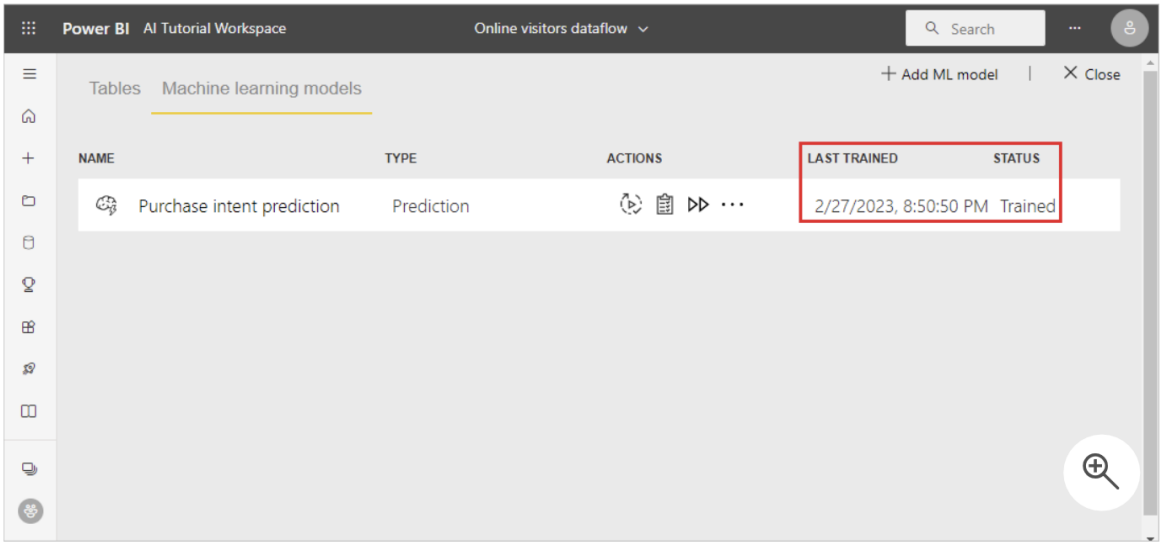

Power BI
AI Tutorial Workspace
Search
Notifications
Settings
Download
Help & Support
Feedback
General
Alerts
Subscriptions
Dashboards
Manage group storage
Admin portal
Manage connections and gateways
Settings
Azure Analysis Services migrations
Manage embed codes
Privacy
Language
Close account
Developer
ArcGIS Maps for Power BI
Hidden items
Q&A questions
Learn more
Apply
Discard

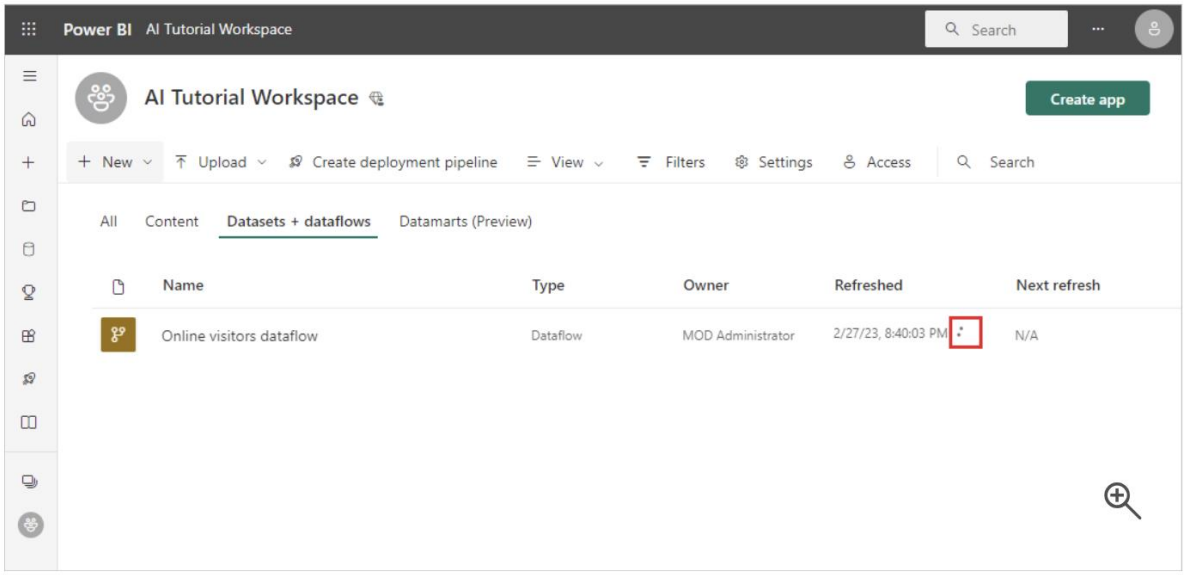


Track training status

The training process begins by sampling and normalizing your historical data and splitting your semantic model into two new entities: Purchase Intent Prediction Training Data and Purchase Intent Prediction Testing Data.

Depending on the size of the semantic model, the training process can take anywhere from a few minutes up to the training time you selected. You can confirm that the model is being trained and validated through the status of the dataflow. The status appears as a data refresh in progress in the Semantic models + dataflows tab of the workspace.





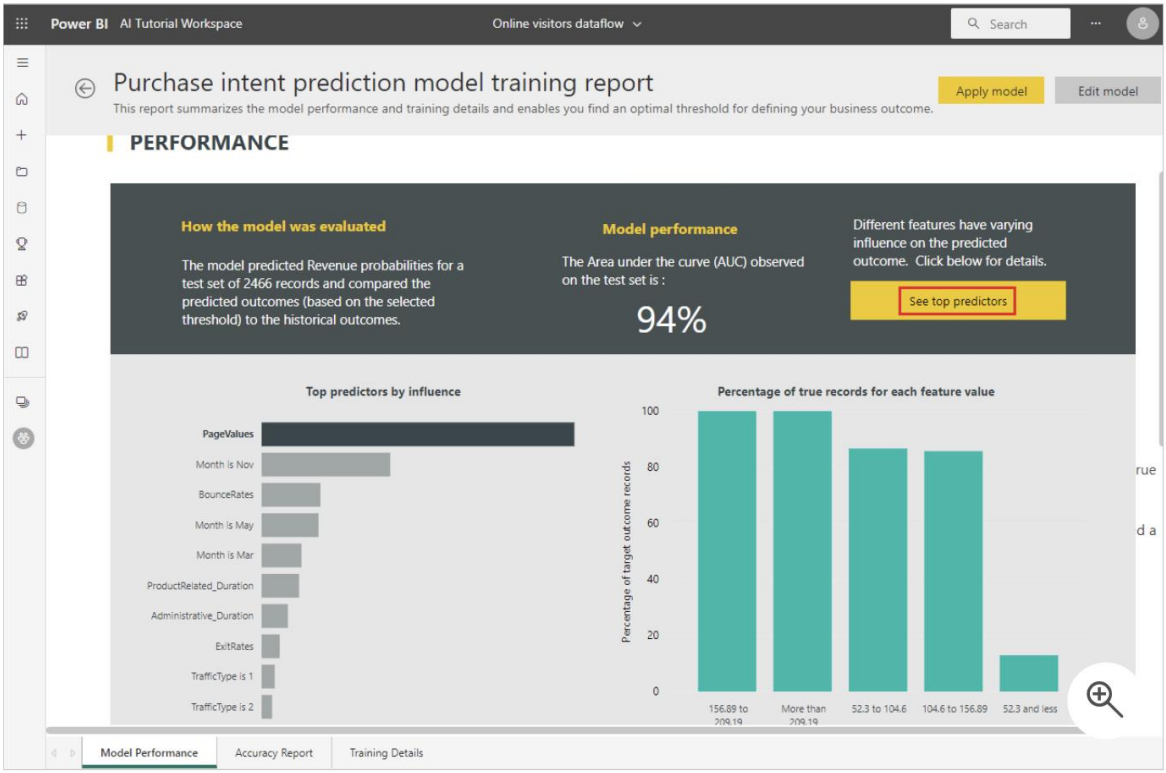
You can see the model in the Machine learning models tab of the dataflow. Status indicates whether the model has been queued for training, is under training, or is trained. Once the model training is completed, the dataflow displays an updated Last trained time and a status of Trained.

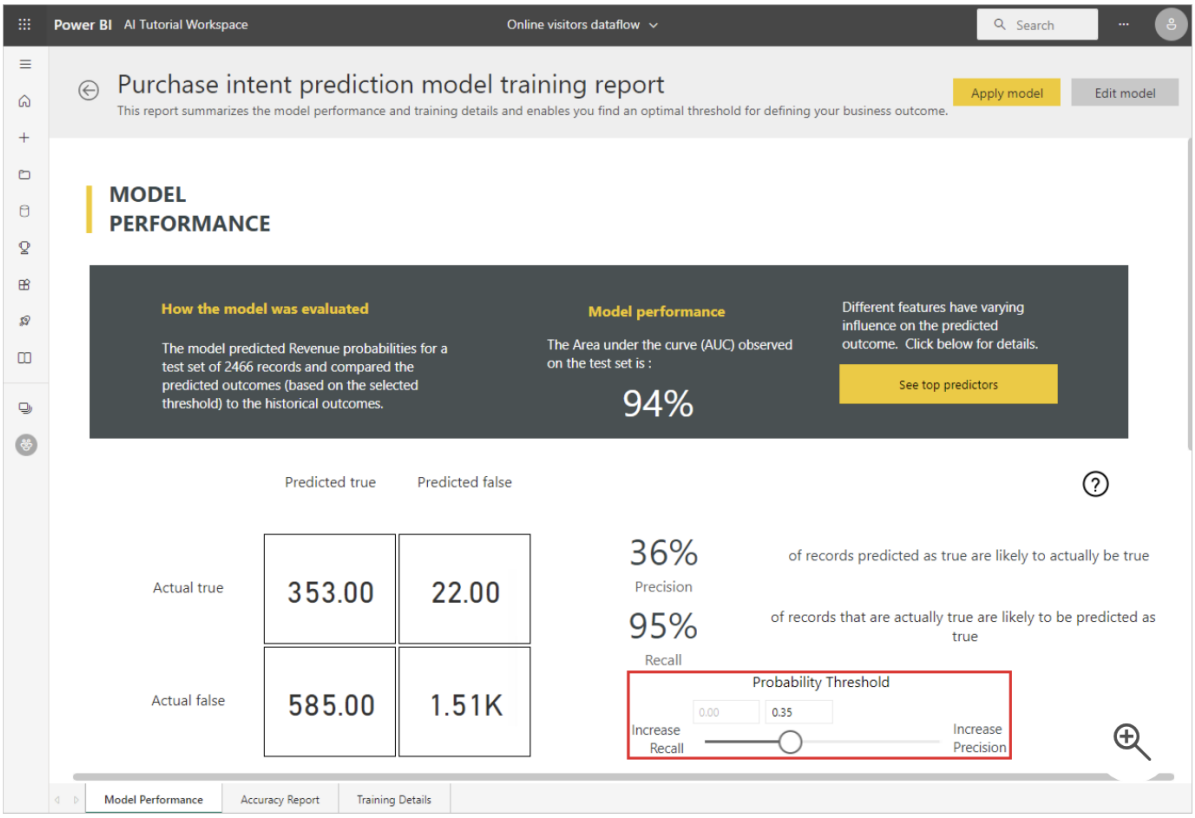
Review the model validation report

To review the model validation report, in the Machine learning models tab, select the View training report icon under Actions. This report describes how your machine learning model is likely to perform.

In the Model Performance page of the report, select See top predictors to view the top predictors for your model. You can select one of the predictors to see how the outcome distribution is associated with that predictor.

You can use the Probability Threshold slicer on the Model Performance page to examine the influence of model Precision and Recall on the model.



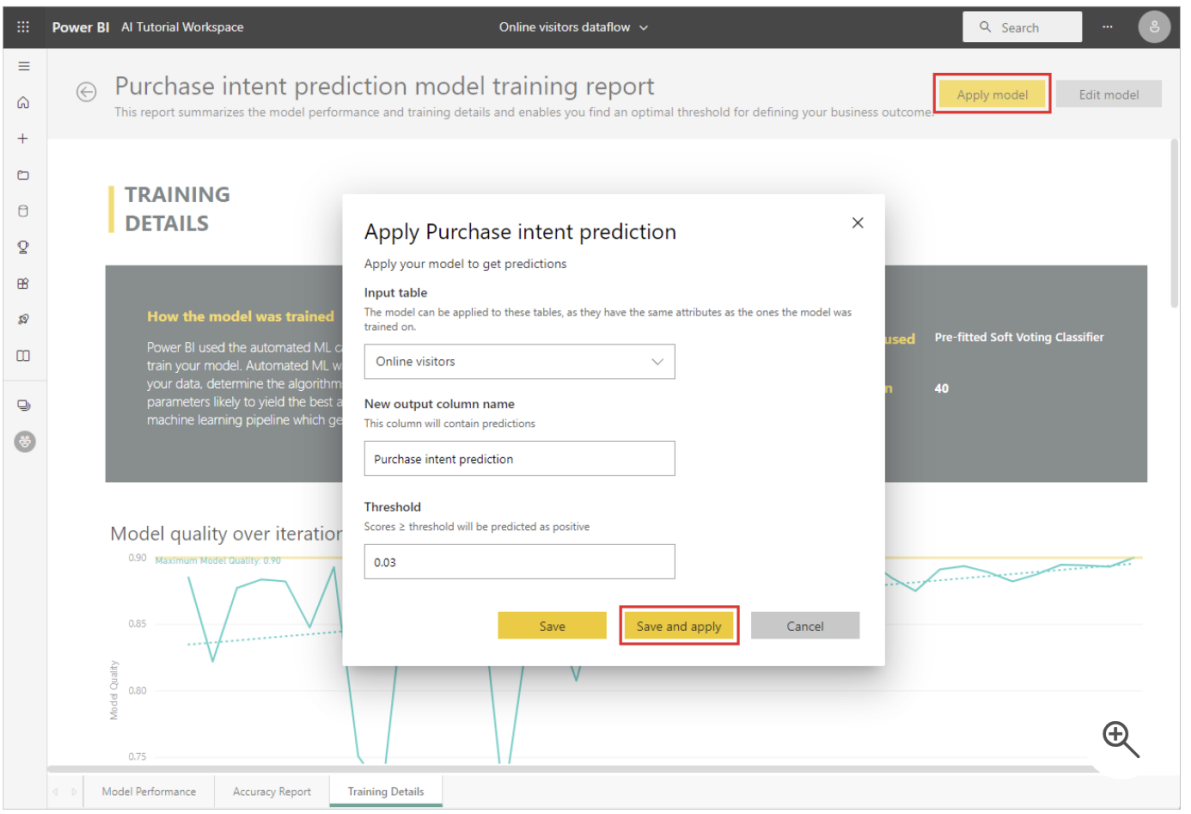


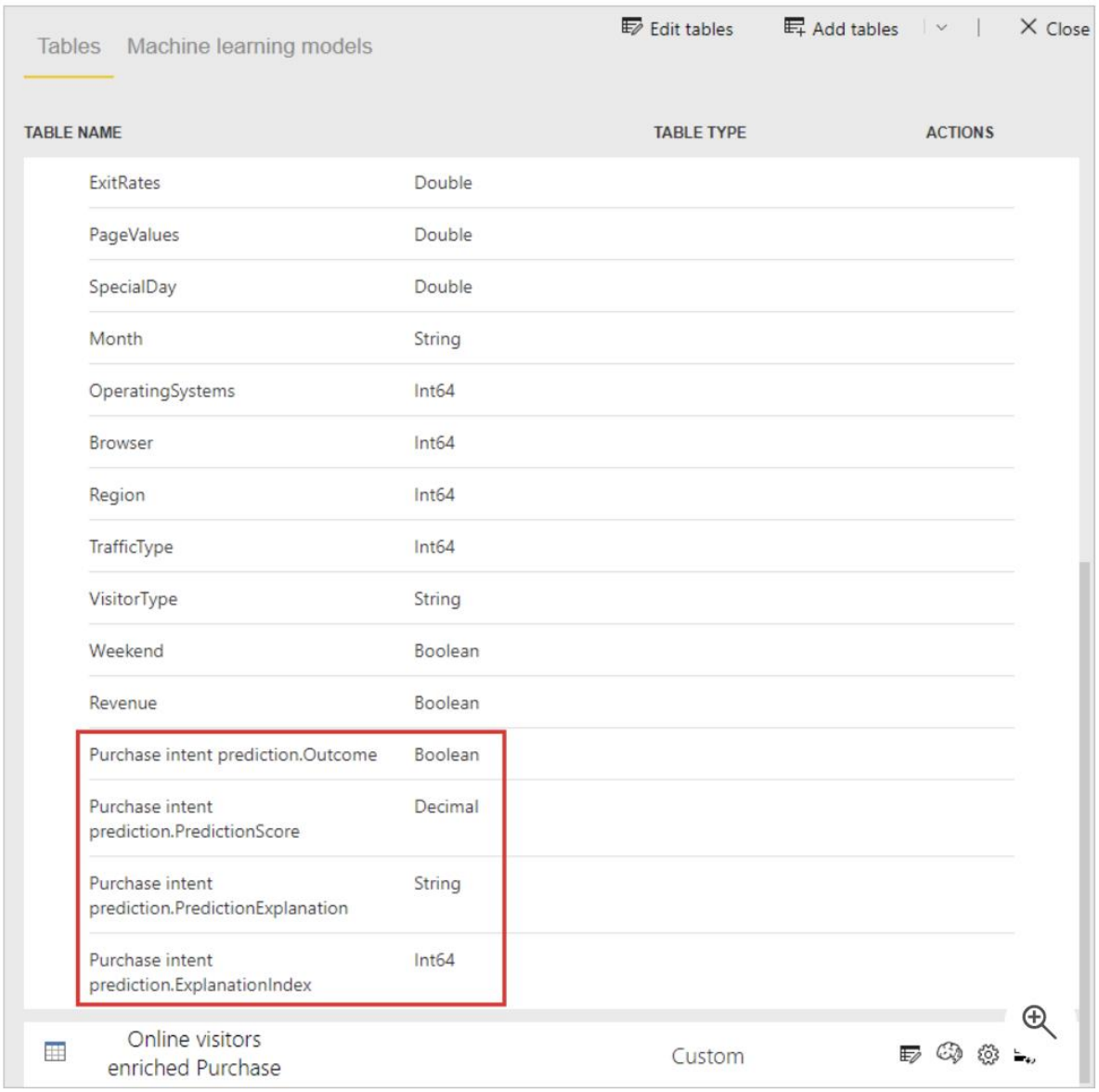
The other pages of the report describe the statistical performance metrics for the model.

The report also includes a Training Details page that describes the Iterations run, how features were extracted from the inputs, and the hyperparameters for the Final model used.

Apply the model to a dataflow entity

Select the Apply model button at the top of the report to invoke this model. In the Apply dialog, you can specify the target entity that has the source data to apply the model to. Then select Save and apply.

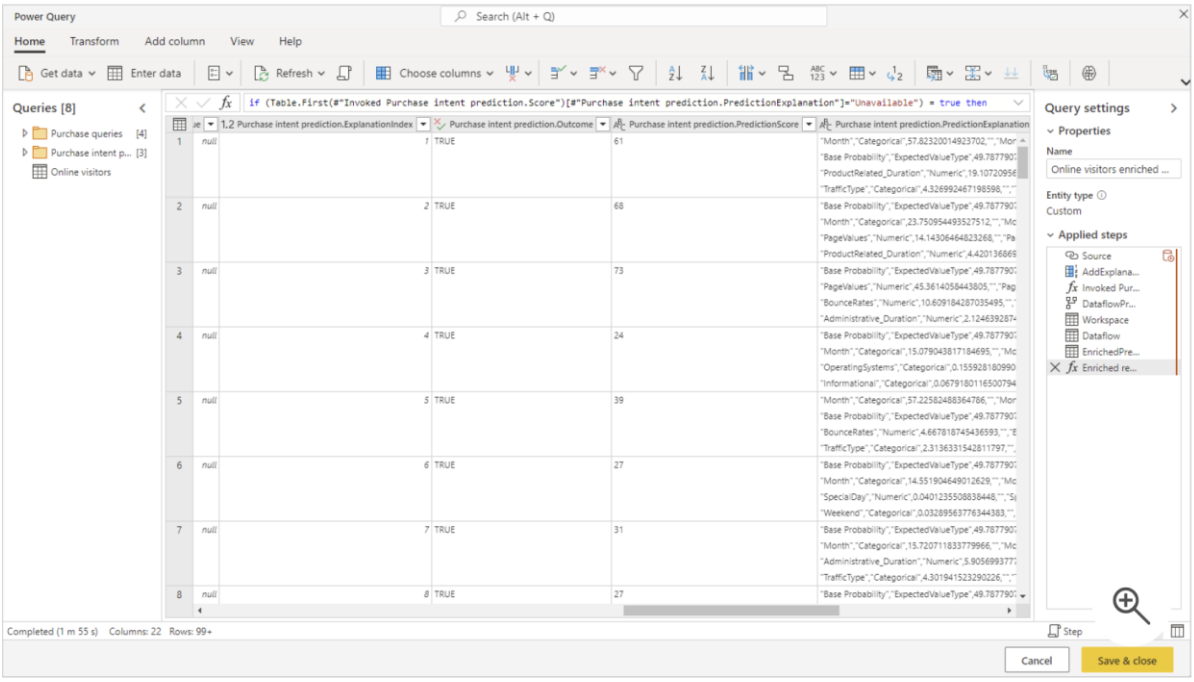

Power BI
AI Tutorial Workspace
Online visitors dataflow
Search
Purchase intent prediction model training report
This report summarizes the model performance and training details and enables you find an optimal threshold for defining your business outcome
Apply model
Edit model
TRAINING DETAILS
How the model was trained
Pre-fitted Soft Voting Classifier
40
Model quality over iteration
Apply Purchase intent prediction
Apply your model to get predictions
Input table
The model can be applied to these tables, as they have the same attributes as the ones the model was trained on.
Online visitors
New output column name
This column will contain predictions
Purchase intent prediction
Threshold
Scores ≥ threshold will be predicted as positive
0.03
Save
Save and apply
Cancel
Model Performance
Accuracy Report
Training Details



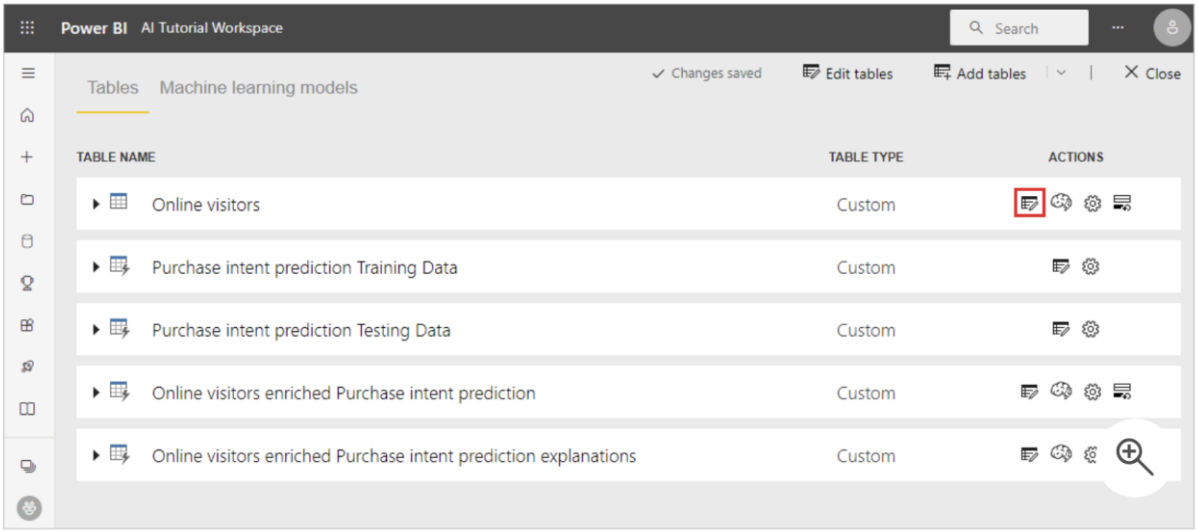
Applying the model creates two new tables, with the suffixes enriched <model_name> and enriched <model_name> explanations. In this case, applying the model to the Online visitors table creates:

Online visitors enriched Purchase intent prediction, which includes the predicted output from the model. Online visitors enriched Purchase intent prediction explanations, which contains top record-specific influencers for the prediction.

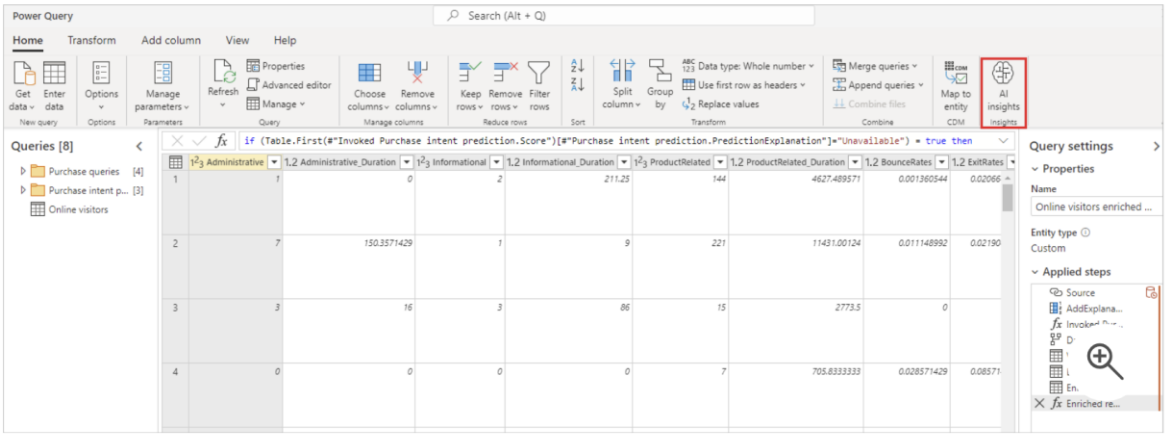
Applying the binary prediction model adds four columns: Outcome, PredictionScore, PredictionExplanation, and ExplanationIndex, each with a Purchase intent prediction prefix.



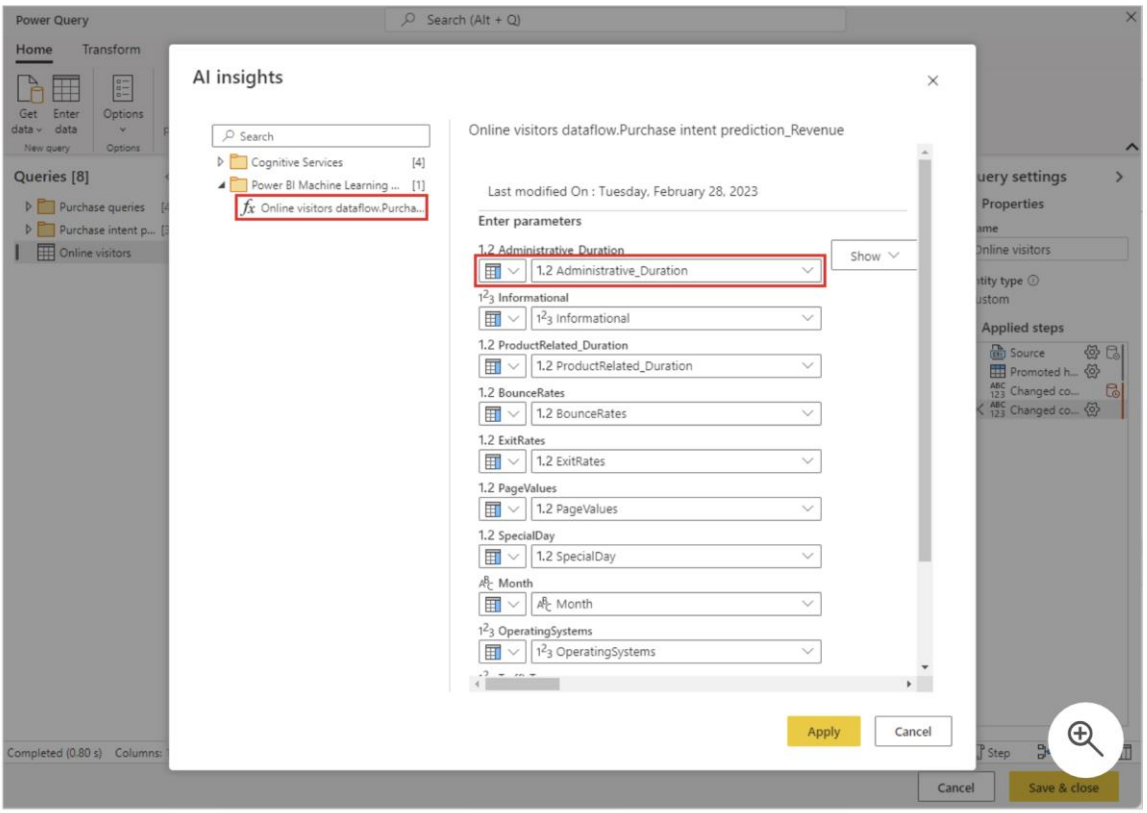
Once the dataflow refresh completes, you can select the Online visitors enriched Purchase intent prediction table to view the results.



You can also invoke any automated machine learning model in the workspace directly from the Power Query Editor in your dataflow. To access the automated machine learning models, select Edit for the table that you want to enrich with insights from your automated machine learning model.

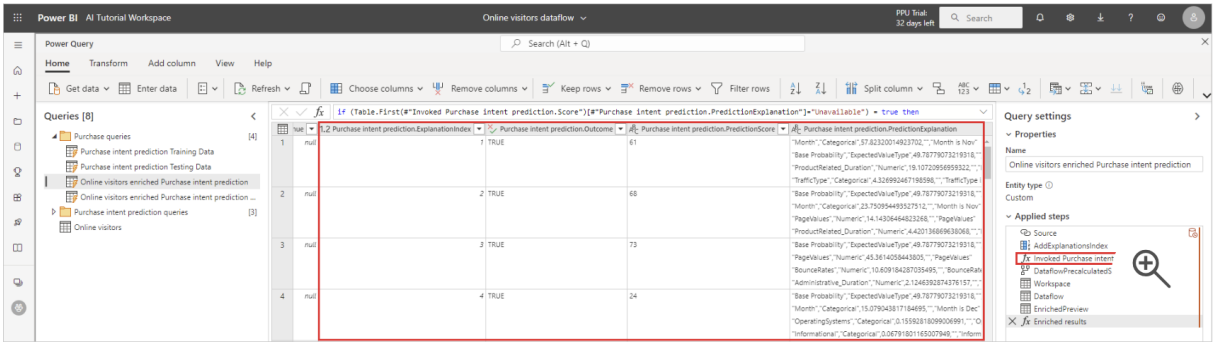


In the Power Query Editor, select AI insights in the ribbon.



On the AI insights screen, select the Power BI Machine Learning Models folder from the navigation pane. The list shows all the machine learning models you have access to as Power Query functions. The input parameters for the machine learning model automatically map as parameters of the corresponding Power Query function. The automatic parameter mapping happens only if the names and data types of the parameter are the same.

To invoke a machine learning model, you can select any of the selected model's columns as an input in the dropdown list. You can also specify a constant value to use as an input by toggling the column icon next to the input line.



Select Apply to view the preview of the machine learning model output as new columns in the table. You also see the model invocation under Applied steps for the query.

After you save your dataflow, the model automatically invokes when the dataflow refreshes, for any new or updated rows in the entity table.

Using the scored output from the model in a Power BI report

To use the scored output from your machine learning model, you can connect to your dataflow from Power BI Desktop by using the Dataflows connector. You can now use the Online visitors enriched Purchase intent prediction table to incorporate the predictions from your model in Power BI reports.

Limitations

There are some known issues with using gateways with automated machine learning. If you need to use a gateway, it's best to create a dataflow that imports the necessary data via the gateway first. Then create another dataflow that references the first dataflow to create or apply these models.

If your AI work with dataflows fails, you may need to enable Fast Combine when using AI with dataflows. Once you have imported your table and *before* you begin to add AI features, select Options from the Home ribbon, and in the window that appears select the checkbox beside *Allow combining data from multiple sources* to enable the feature, then select OK to save your selection. Then you can add AI features to your dataflow.

Conclusion:

Source Code and Output /Screenshots:

(To be provided by the student)

//Source code and output/screenshot should be available here

Useful Resource:

Exercise Questions

Experiment No 11

Experiment Title: Download any dataset of CSV files, XML, JSON format and perform data modeling, and create a basic dashboard using Tableau and PowerBI.

Objective:

1. Perform data cleaning to handle missing values, duplicates, and any inconsistencies in the dataset.
2. Pre-process the data to make it suitable for modeling, including data type conversions and normalization.
3. Define and implement a data model that reflects the relationships between different entities in the dataset.
4. Design and create a basic dashboard in Tableau that presents key insights from the dataset.
5. Design and create a basic dashboard in PowerBI with similar key insights as the Tableau dashboard.
6. Present the Tableau and PowerBI dashboards, highlighting the differences and similarities.

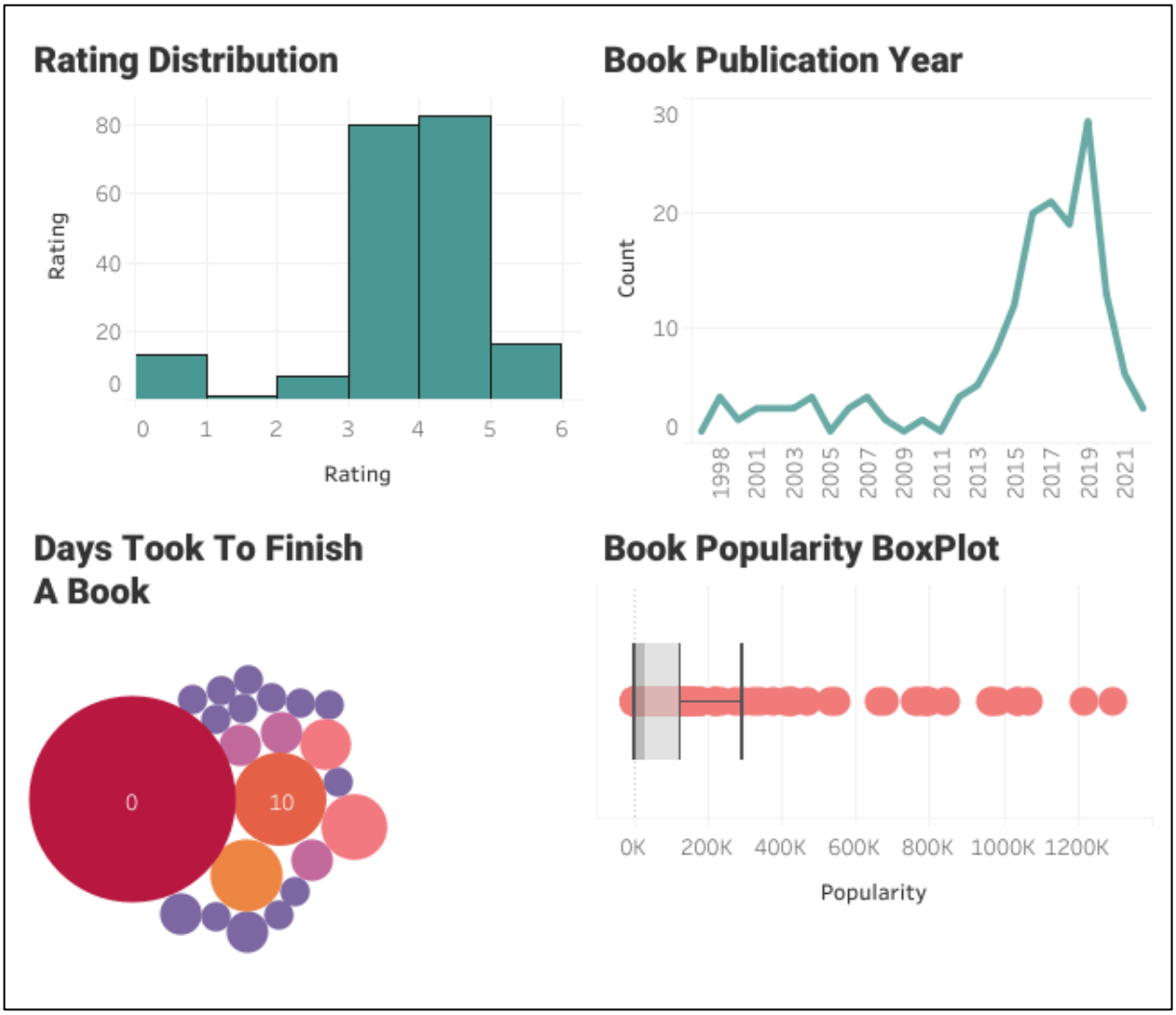
Theory: Visualizing Data with Python and Tableau Tutorial (Learn how we can use Python to extend Tableau's data visualization capabilities).

Tableau provides several options to augment and create new data fields. You can perform arithmetic, logical, spatial, and predictive modeling functions using calculated fields. Tableau is a powerful Business Intelligence (BI) tool, but there are limitations; that's where Python language comes to the rescue.

Python is popular programming among the data community. You can use it to extract, clean, process, and apply complex statistical functions to the data. It provides you with machine learning frameworks, data orchestrations, multiprocessing, and rich libraries to perform almost any task possible.

Python is a multipurpose language, and using it with Tableau gives us the freedom to perform highly complex tasks. In this tutorial, we are going to use Python for extracting and cleaning the data. Then, we will be using clean data to create data visualization on Tableau.

We will not be using **Tabpy** to create a Tableau Python server and execute Python scripts within Tableau. Instead, we will first extract and clean the data in Python (Jupyter Notebook) and then use Tableau to create interactive visualization.



Goodreads Data Viz | Tableau Public

This is a code-based step-by-step tutorial on Goodreads API and creating complex visualization on Tableau. Check out the link below to access the code and the Tableau dashboard.

- **DataCamp Workspace**
- **Tableau Public**

Data Ingestion and Processing with Python

In the first part of the tutorial, we will learn to use Goodreads API to access public data. In our case, we will be focusing on the user profile and converting it into a readable Pandas dataframe. Furthermore, we will clean the data and export it into CSV file format.

Getting Started

We will be using DataCamp's Workspace for running the Python code. It comes with the necessary Python packages for data science tasks.

If you are new to Python and want to set up the environment on your local machine, install **Anaconda**. It will install Python, Jupyter Notebook, and necessary Python Packages.

Before we start writing the code, we have to install the xmltodict package as it is not part of the Workspace or Anaconda data stack. We will use `pip` to install the missing Python package.

Note: The `!` symbol only works in Jupyter Notebooks. It lets us access the terminal within the Jupyter code cell.

```
!pip install xmltodict
```

```
>>> Collecting xmltodict
```

```
>>>   Using cached xmltodict-0.13.0-py2.py3-none-any.whl (10.0 kB)
```

```
>>> Installing collected packages: xmltodict
```

```
>>> Successfully installed xmltodict-0.13.0
```

In the next step, we will import the necessary packages.

```
import pandas as pd
```

```
import xmltodict
```

```
import urllib.request
```

Parsing the Profile Link

To extract user data, we need both user id and user name. In this section, we will parse the user (**Abid**) profile link.

Note: You can use your friend's profile or use your profile link, and run this script.

1. It extracts user_id by filtering digitals within the link and returns "73376016".
2. To extract user_name, we will split out the string on user_id and then split it on "-" to get the user. After replacing "-" with a space, we get the user name "abid".
3. Finally, we will concatenate user_id with user_name. This unique id will be used in the next section to access user data.

```
Goodread_profile = "https://www.goodreads.com/user/show/73376016-abid"
```

```
user_id = ''.join(filter(lambda i: i.isdigit(), Goodread_profile))
```

```
user_name = Goodread_profile.split(user_id, 1)[1].split('-', 1)[1].replace('-', ' ')
```

```
user_id_name = user_id+'-'+user_name
```

```
print(user_id_name)
```

```
>>> 73376016-abid
```

Goodreads Data Extraction

At the end of 2020, Goodreads will stop providing developer API. You can read the full report **here**. To overcome this issue, we will be using API keys from old projects such as **streamlit_goodreads_app**. The project explains how to access the Goodreads user data using API.

Goodreads also provides you the option to download the data in CSV file format without an API key, but it is limited to a user, and it doesn't give us the freedom to extract real-time data.

In this section, we will be creating functions that will take user_id_name, version, shelf, per_page, and apiKey.

- apiKey: is to get access to the public data
- version: to specify the latest data type.
- shelf: There are multiple shelves in the user profile but mostly read, to-read, and currently-reading.
- per_page: Number of books entries per page

The function takes user inputs to prepare the URL and then downloads the data using urllib.request. Finally, we get the data in XML format.

```
apiKey = "ZRnySx6awjQuExO9tKEJXw"
version = "2"
shelf = "read"
per_page = "200"
def get_user_data(user_id, apiKey, version, shelf, per_page):
    api_url_base = "https://www.goodreads.com/review/list"
    final_url = (
        api_url_base
        + user_id
        + ".xml?key="
        + apiKey
        + "&v="
        + version
        + "&shelf="
        + shelf
        + "&per_page="
        + per_page
    )
    contents = urllib.request.urlopen(final_url).read()
    return contents
contents = get_user_data(user_id_name,apiKey,version, shelf, per_page)
print(contents[0:100])
```

```
>>> b'<?xml version="1.0" encoding="UTF-8"?>\n<GoodreadsResponse>\n <Request>\n <authentication>true</aut'
```

Converting XML to JSON

Our initial data is in XML format, and there is no direct way to convert it into a structured database. So, we will transform it into JSON using the **xmltodict** Python package.

The XML data is converted into nested JSON format, and to display the first entry in book reviews data, we will use square brackets to access encapsulated data.

You can experiment with metadata and explore more options, but in this tutorial, we will be focusing on users reviewing data.

```
contents_json = xmltodict.parse(contents)
```

```
print(contents_json["GoodreadsResponse"]["reviews"]["review"][:1])
```

```
>>> [{'id': '4626706284', 'book': {'id': {'@type': 'integer', '#text': '57771224'}, 'isbn': '1250809606', 'isbn13': '9781250809605', 'text_reviews_count': {'@type': 'integer', '#text': '150'}, 'uri': 'kca://book/amzn1.gr.book.v3.tcNoY0o7ErAhczdQ', 'title': 'Good Intentions', 'title_without_series': 'Good Intentions', 'image_url': .....
```

Converting JSON to Pandas Dataframe

To convert JSON data type to Pandas dataframe, we will use the json_normalize function. The review data is present at the third level, and to access it, we will access GoodreadsResponse, reviews, and review.

Before we display the data frame, we will filter out irrelevant data by dropping the books with missing date_updated column.

Learn various ways to ingest CSV files, spreadsheets, JSON, SQL databases, and APIs using Pandas by taking **Streamlined Data Ingestion with pandas** course.

```
df = pd.json_normalize(contents_json["GoodreadsResponse"]["reviews"]["review"])
```

```
df = df[df["date_updated"].notnull()]
```

```
df.head()
```

	id	rating	votes	spoiler_flag	spoilers_state	recommended_for	recommended_by	started_at	read_at	date_added	...
0	4626706284	3	0	false	none	None	None	Wed Mar 23 00:00:00 -0700 2022	Thu Mar 24 00:00:00 -0700 2022	Wed Mar 23 21:56:32 -0700 2022	...
1	4617282277	5	0	false	none	None	None	Sat Mar 19 00:00:00 -0700 2022	Mon Mar 21 13:12:00 -0700 2022	Sat Mar 19 07:18:42 -0700 2022	...
2	4611790134	3	0	false	none	None	None	Mon Mar 28 00:35:23 -0700 2022	Fri Apr 08 12:12:46 -0700 2022	Wed Mar 16 07:14:39 -0700 2022	...
3	4539885289	4	0	false	none	None	None	Wed Feb 09 00:00:00 -0800 2022	Mon Mar 07 10:50:51 -0800 2022	Wed Feb 09 11:20:31 -0800 2022	...
4	4386936522	5	0	false	none	None	None	Mon Dec 13 06:32:37	Sun Dec 26 09:51:35	Mon Dec 13 06:32:36	...

Data Cleaning

The raw dataframe looks reasonably clean, but we still need to reduce the number of columns.

As we can see, there are 61 columns.

```
df.shape
```

```
(200, 61)
```

Let's drop the empty ones.

```
df.dropna(axis=1, how='all', inplace=True)
```

```
df.shape
```

```
(200, 58)
```

We have successfully dropped 3 columns with missing values.

We will now check all column names by using `df.columns` and select the most useful columns.

```
final_df = df[
    [
        "rating",
        "started_at",
        "read_at",
        "date_added",
        "book.title",
        "book.average_rating",
```

```
    'book.ratings_count',
    "book.publication_year",
    "book.authors.author.name"
  ]
]
final_df.head()
```

As we can observe, the final dataframe looks clean with the relevant data fields.

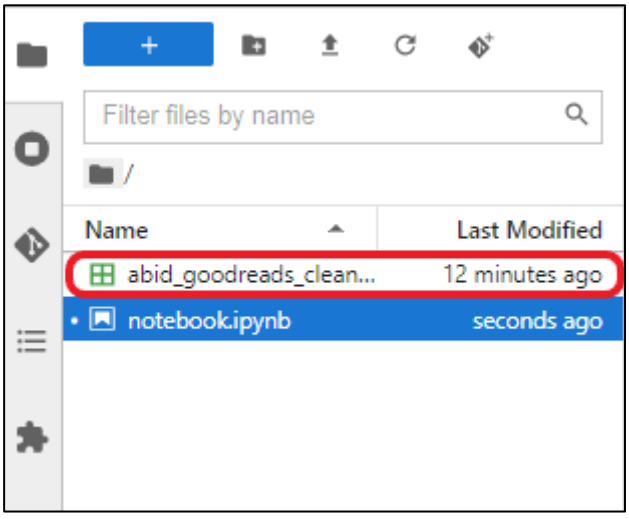
	rating	started_at	read_at	date_added	book.title	book.average_rating	book.ratings_count	book.publication_year	book.authors.author.name
0	3	Wed Mar 23 00:00:00 -0700 2022	Thu Mar 24 00:00:00 -0700 2022	Wed Mar 23 21:56:32 -0700 2022	Good Intentions	3.53	655	2022	Kasim Ali
1	5	Sat Mar 19 00:00:00 -0700 2022	Mon Mar 21 13:12:00 -0700 2022	Sat Mar 19 07:18:42 -0700 2022	The One	4.12	100020	2018	John Marrs
2	3	Mon Mar 28 00:35:23 -0700 2022	Fri Apr 08 12:12:46 -0700 2022	Wed Mar 16 07:14:39 -0700 2022	Nine Lives	3.60	11972	2022	Peter Swanson
3	4	Wed Feb 09 00:00:00 -0800 2022	Mon Mar 07 10:50:51 -0800 2022	Wed Feb 09 11:20:31 -0800 2022	Out of Office: The Big Problem and Bigger Prom...	3.79	1505	2021	Charlie Warzel
4	5	Mon Dec 13 06:32:37 -0800 2021	Sun Dec 26 09:51:35 -0800 2021	Mon Dec 13 06:32:36 -0800 2021	Supernova (Renegades, #3)	4.42	49612	2019	Marissa Meyer

Exporting CSV File

In the last section, we will export the dataframe into a CSV file that is compatible with Tableau. In the to_csv function, add the name of the file with the extension type and drop the index by changing the index argument to **False**.

```
final_df.to_csv("abid_goodreads_clean_data.csv",index=False)
```

The CSV file will show in the current directory.



Goodreads Clean CSV File

You can also check out the Python Jupyter Notebook: **Data Ingestion using Goodreads API**. It will help you debug your code and if you want to skip the Python programming part, you can simply download the file by clicking on the **Copy & Edit** button and running the script.

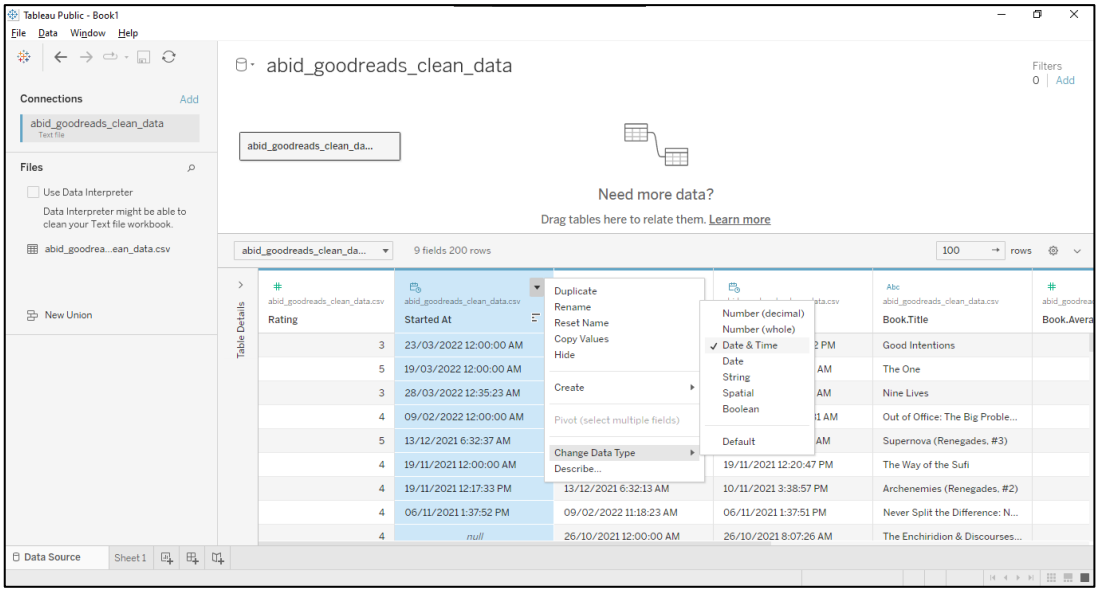
Data Visualization in Tableau

In the second part, we will use clean data and create simple and complex data visualization in Tableau. Our goal is to plot interactive charts which will help us understand the user's book reading behavior.

Connecting the Data

We will connect the CSV file by selecting the **Text** file option and selecting the abid_goodreads_clean_data.csv file. After that, we will change the Started At, Read At, and Date Added data fields to **Date & Time**, as shown below.

Note: It is a good practice to modify your data fields at the start.

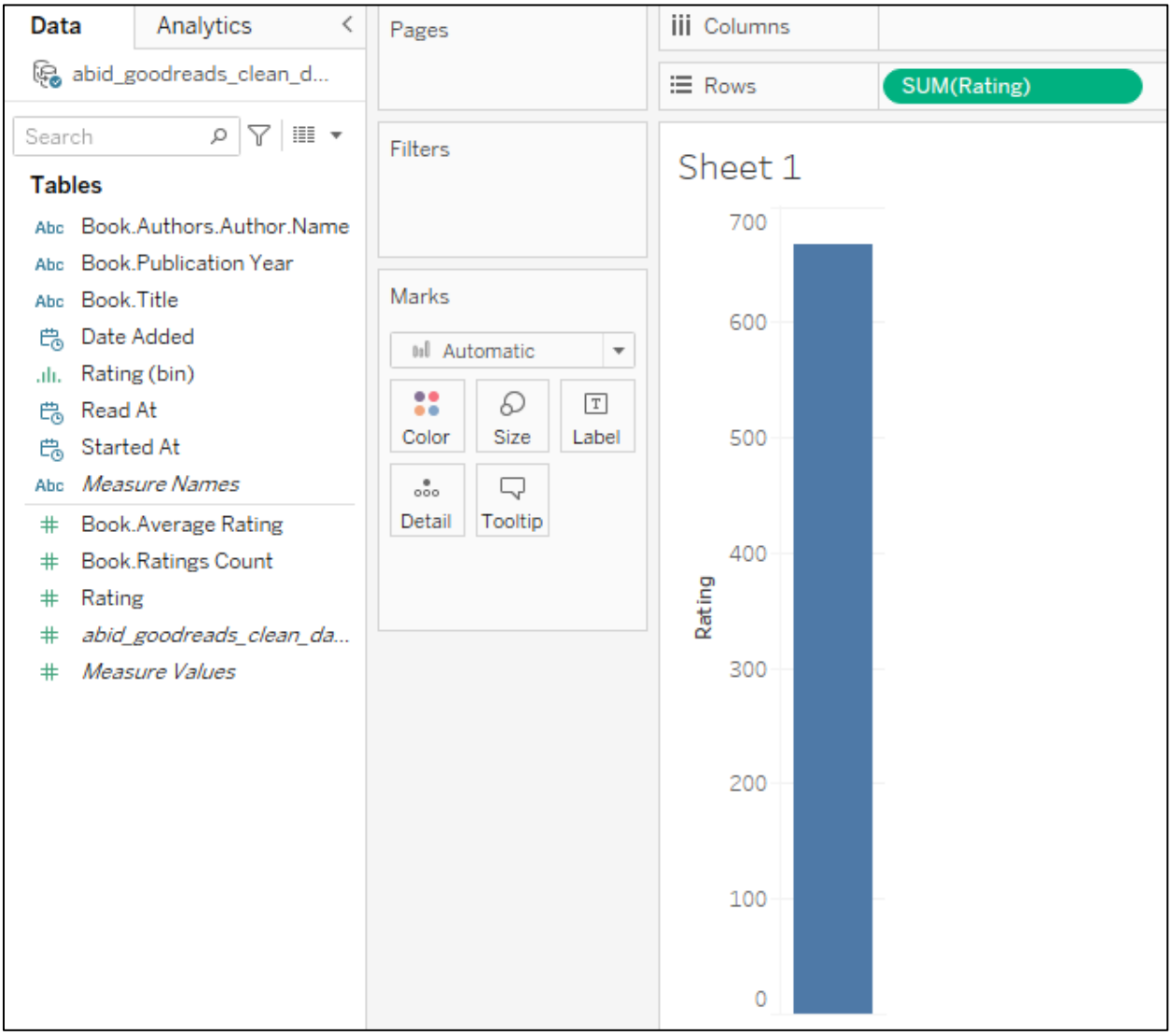


Connecting Data and Modifying Data Types

Creating Rating Histogram

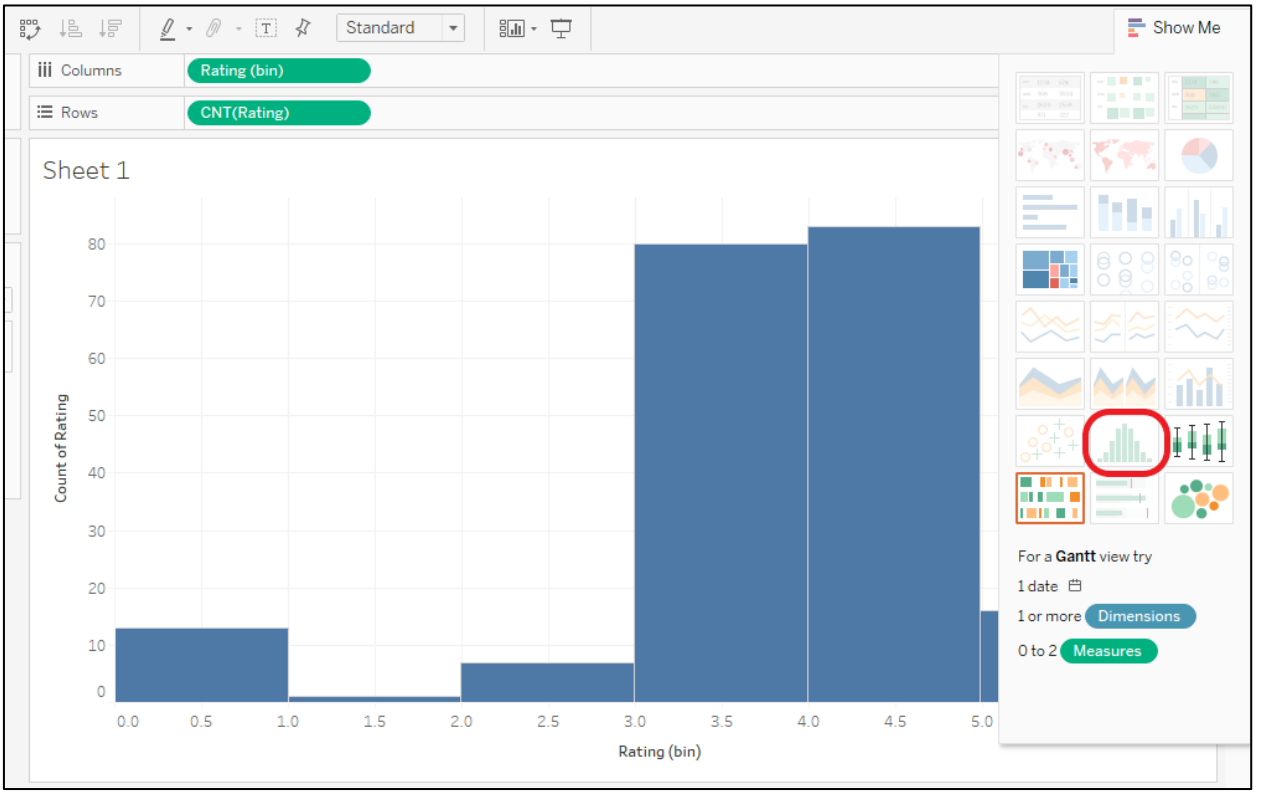
In this section, we will create the user book rating histogram.

- First, drag and drop the **Rating** field to the **Rows** shelf.



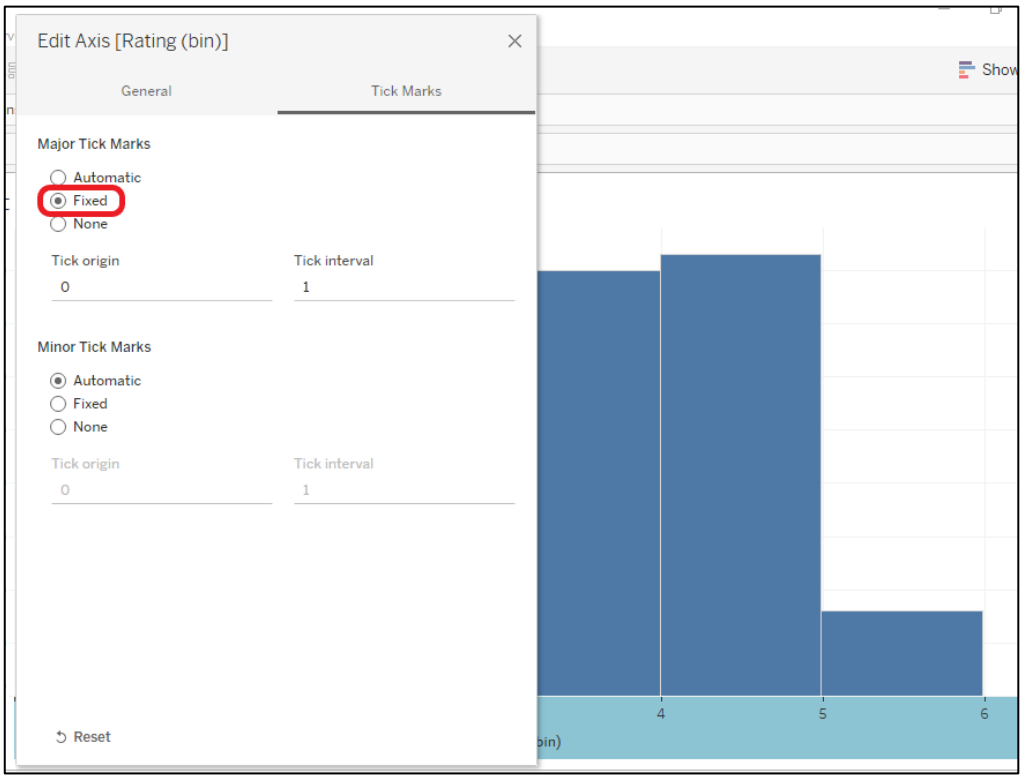
User Rating Histogram Part 1

- Click on the **Show Me** drop-down button to access the visualization templates. We will convert the bar chart to a histogram by clicking on the **Histogram** option.



User Rating Histogram Part 2

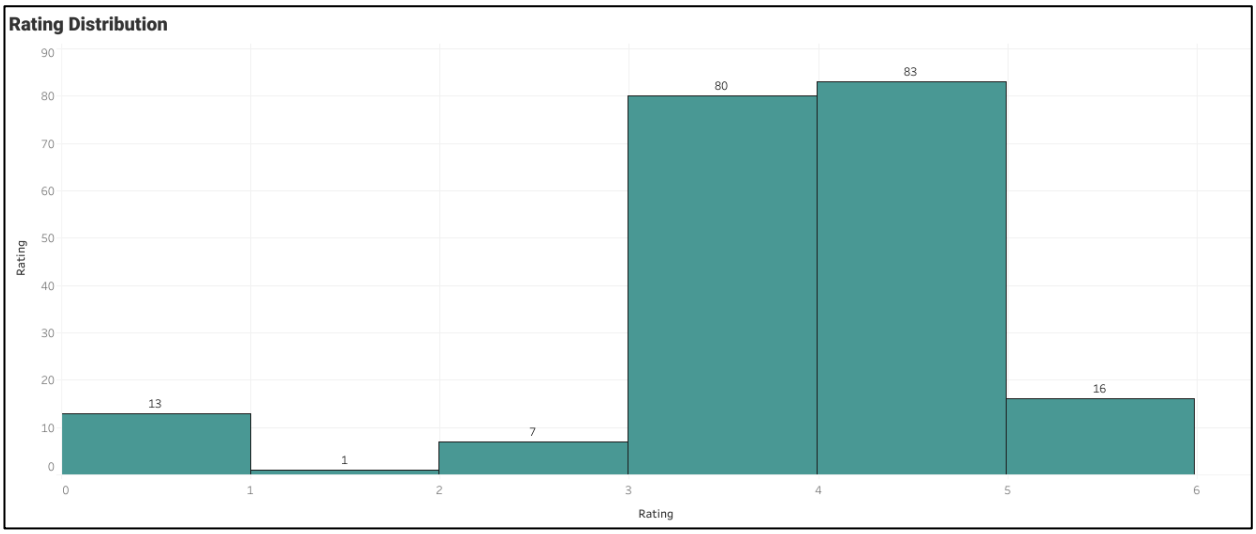
- The Rating axis has 0.5 interval tick marks. Change the tick marks by right-clicking on the bottom axis and selecting **Edit Axis**. After that click on the **Tick Marks** tab and change the **Major Tick Marks** to **Fixed**. Make sure the **Tick Origin** is 0 and the **TickInterval** is 1.



User Rating Histogram Part 3

- We will customize the histogram by cleaning axis labels, changing the colors and borders of the bar, and adding mark labels. You can do all of this by accessing the options on the **Marks** panel.

You can find it in the middle-left section.



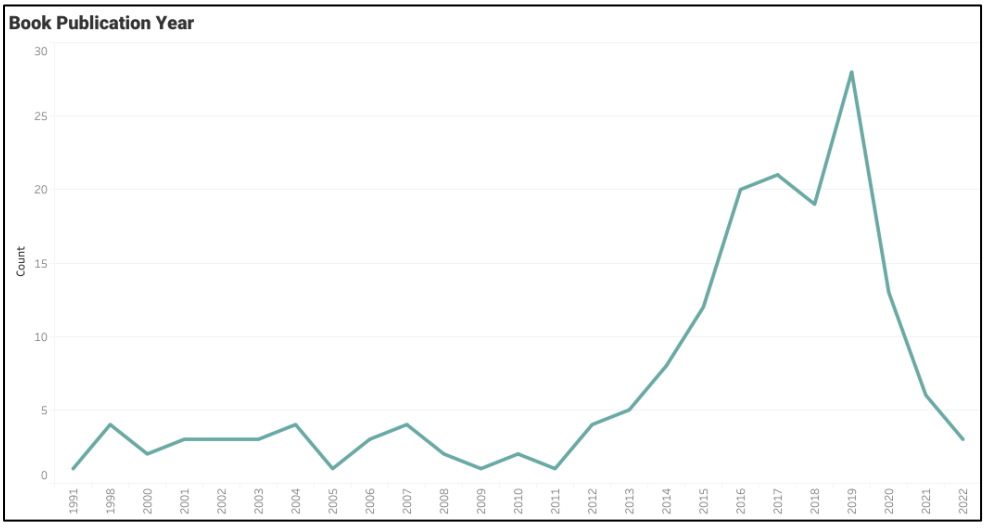
User Rating Histogram Part 4

The user had typically given ratings between 3 and 4. The zero ratings are the books that are not rated.

Line Plot

To plot line chart:

1. Drag and drop *Book.Publication Year* field to **Rows** and **Columns** Shelf.
2. Change the **Rows** data field to count by right-clicking on it and selecting **Measure** > **Count**.
3. Change the **Columns** data field to dimensions by right-clicking on it and selecting **Dimensions**.
4. Go to the **Marks** panel, click the Automatic dropdown option, and change it to **Line**.
5. Clean the axis label, customize the chart, add title, and remove null values.



Book Publication Year Line Plot

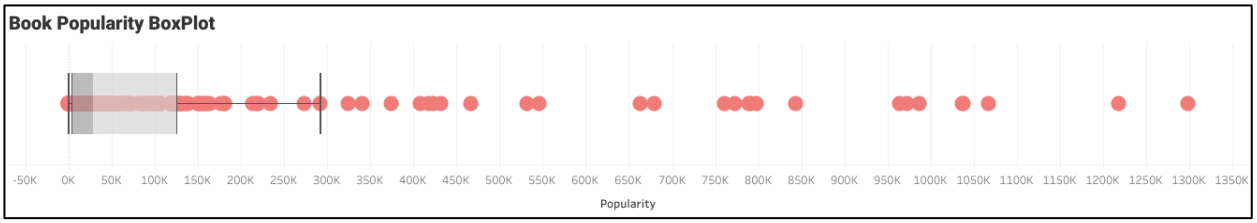
The user has read some old books, but they are particularly interested in the books that are published between the years 2015 and 2020.

If you are feeling overwhelmed and want to learn the fundamentals of Tableau, you might find **Tableau Tutorial for Beginners** by Eugenia Anello helpful.

Box Plot

To plot box and whisker plot:

1. Drag and drop *Books.Ratings Count* data field to **Rows** shelf. Change it from **Discrete** to **Continuous**.
2. Drag and drop *Books.Ratings Count* data field to **Detail** option at **Marks** panel. Change it from **Measure** to **Dimension**.
3. Click on the **Show Me** drop-down option and select the box-and-whisker plots option.
4. Move the data field from **Rows** to **Columns** shelf to make it horizontal.
5. The last part is all about customizing. We will increase the size, change the color, add a title, and rename the axis to "Popularity".



Number of Reviews per Book Boxplot

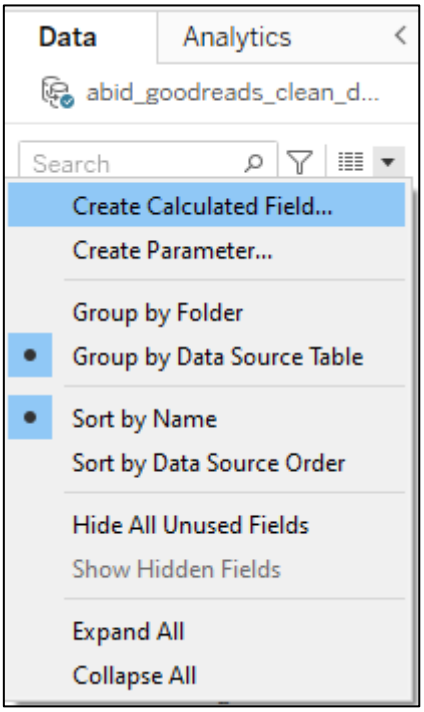
It seems like the user is reading less popular books and a few famous books. It means that the user has a unique taste based on the content, not on popularity.

Bubble Plot

We have created a simple but amazing visualization to learn about user books reading behavior. Next, we will learn more complex data visualization which includes creating a new calculated field, editing bins, and creating multiple layers.

In bubble plots, the labels represent the number of days it took a user to finish a book, and the size of the bubble represents the number of occurrences. We don't have a data field for the duration, but we can create it using Started At and Read At.

In the first step, we have to create a new calculated field by clicking the down arrow in the **Data** panel and selecting **Create Calculated Field**.



Creating Calculated Field

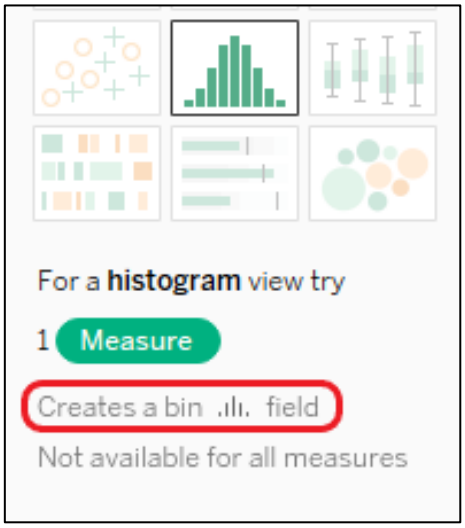
The new window will pop out, and you have to:

1. Rename the title field to “Read Duration”
2. Use the DATEDIFF function to get the difference between Read At and Started At.
3. Make sure the first argument of the functions is “day”.
4. Drag and drop or type the field name in the square bracket as second and third arguments.

Read Duration in Days

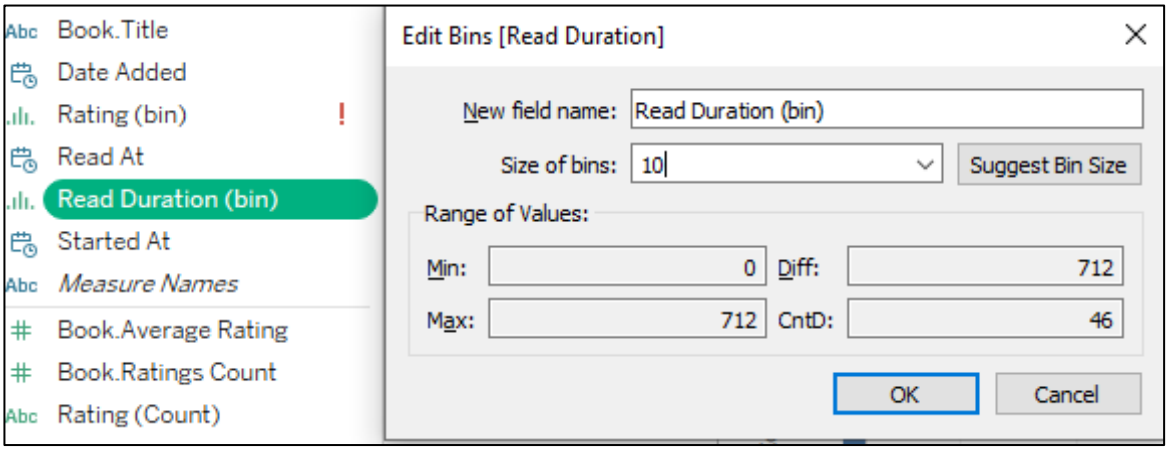
The *Read Duration* data field is continuous, and to plot packet bubbles visualization, we have to divide the data field into smaller chunks known as bins.

1. Drag and drop the newly created `Read Duration` field to the **Rows** shelf.
2. Click on **Show Me** and select **histogram**. It will automatically create a bin field for you.



Creates a Bin Field

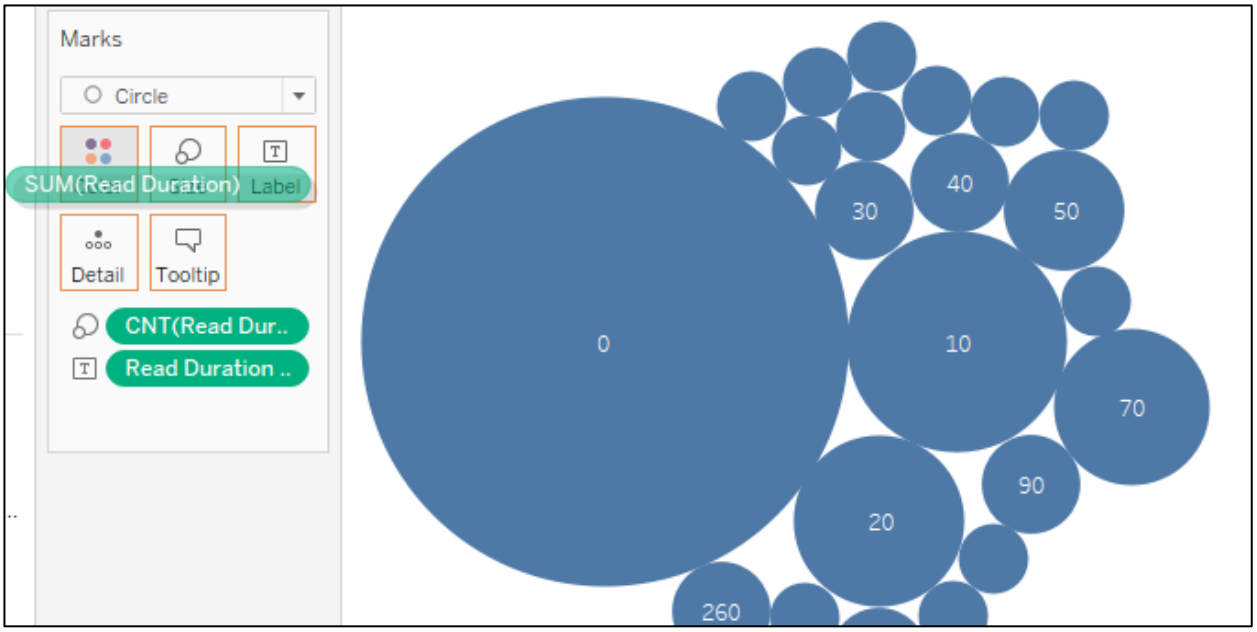
1. Right-click on the recently created Read Duration (bin) field and select the Read Duration in Days option.
2. Change the **Size of Bins** to 10. It will create multiple smaller chunks of data which will help us create a more refined version of the packed bubbles plot.



Editing Bins

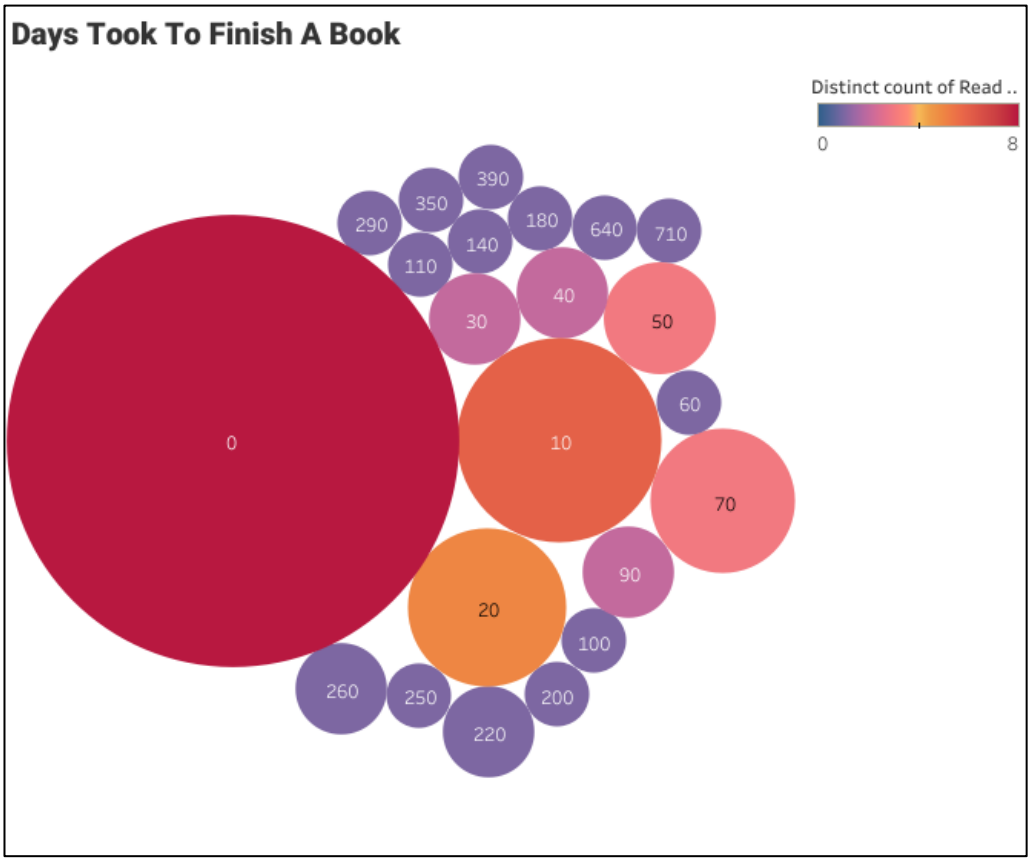
We have crossed the hard path, and now it's time to see the fruits of our labors.

1. To create the simple visualization, click on **Show Me** and select the **packed bubbles option**. You will see unicolor circles of different sizes.
2. To add some colors, we will drag and drop the *Read Duration* field onto the **Color** option in the **Marks** panel.
3. Change the color field to Count (Distinct) by right-clicking on the field and selecting **Measure** > **Count** (**Distinct**). It will give a unique color to each bin or label.



Unicolor Packed Bubbles Plot

1. Click on the **Color** option in the **Marks** panel and select **Edit Colors…** > **Sunrise-Sunset Diverging**. You can pick any gradient color that suits your taste.
2. The last part is all about customizing and making sure your visualization is appealing and conveys the right message.



Packed Bubbles Plot

It took the user less than a day to finish most of the books. You can also see a few outliers above 300. We can also create a Tableau dashboard by combining these visualizations.

Part B: Visualize CSV Data in the Power BI

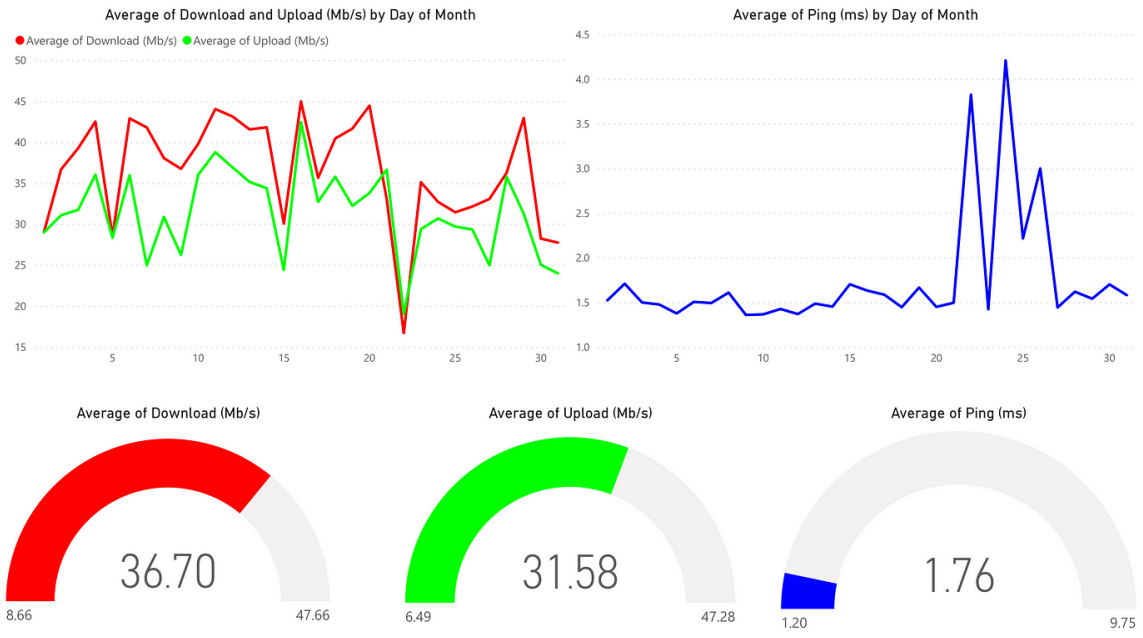
Now that you have Power BI and the dataset in your computer, let's look into the data. This dataset contains internet speeds I've collected at home from January 1st 2019 to March 31st 2019, once a day, using Python along with the speedtest-cli and pandas libraries (to perform the internet speed tests and to write the results to a CSV file, respectively). A link to the code is available at the end of this article.
As you can see in the following sample screenshot from Power BI,

Date	Ping (ms)	Download (Mb/s)	Upload (Mb/s)
Tuesday, January 1, 2019	1.23	45	34.06
Wednesday, January 2, 2019	1.82	44.61	36.32
Thursday, January 3, 2019	1.47	44.99	32.61
Friday, January 4, 2019	1.45	45.69	22.94
Saturday, January 5, 2019	1.68	20.24	19.65
Sunday, January 6, 2019	1.49	47.61	42.63
Monday, January 7, 2019	1.34	36.46	8.47
Tuesday, January 8, 2019	1.3	45.35	44.11
Wednesday, January 9, 2019	1.2	47.03	47.28
Thursday, January 10, 2019	1.3	47.66	46.41

Sample of the dataset viewed in the Data tab of Power BI
our dataset has four columns:

- Date: the date of the test
- Ping (ms): the ping, collected in miliseconds
- Download (Mb/s): the download speed, collected in megabits per second
- Upload (Mb/s): the upload speed, collected in megabits per second

Pretty straightforward. And now for the dashboard we'll create:



Resulting dashboard

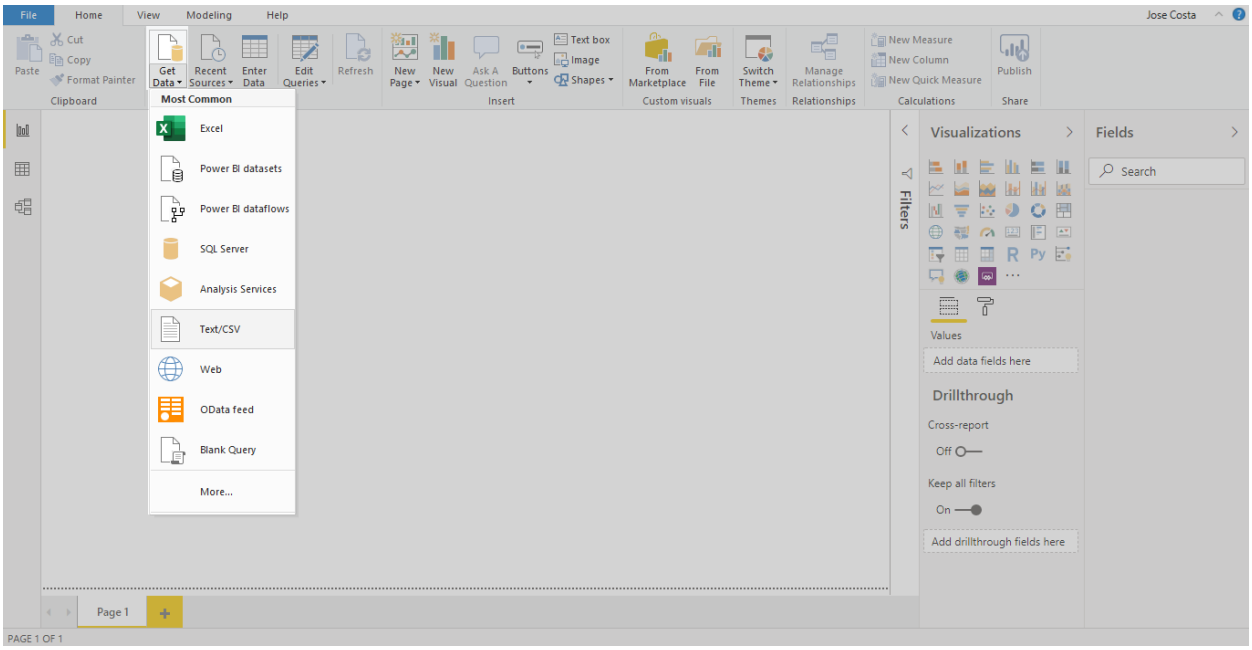
We'll create a dashboard to visualize the average of each speed metric per day of the month, which will include line charts to see the evolution over the course of the days (1–31) and gauges for a quick look at the performance of each metric.

The intent of this dashboard is to create visualizations of the evolution and performance of the internet speeds collected, with the least amount of effort. To create this dashboard in Power BI, it's pretty much a matter of adding line charts and gauges to the canvas and then dragging and dropping the data into each visualization.

Load the data into Power BI

Let's start creating our dashboard. The first thing we'll need is to load the dataset.

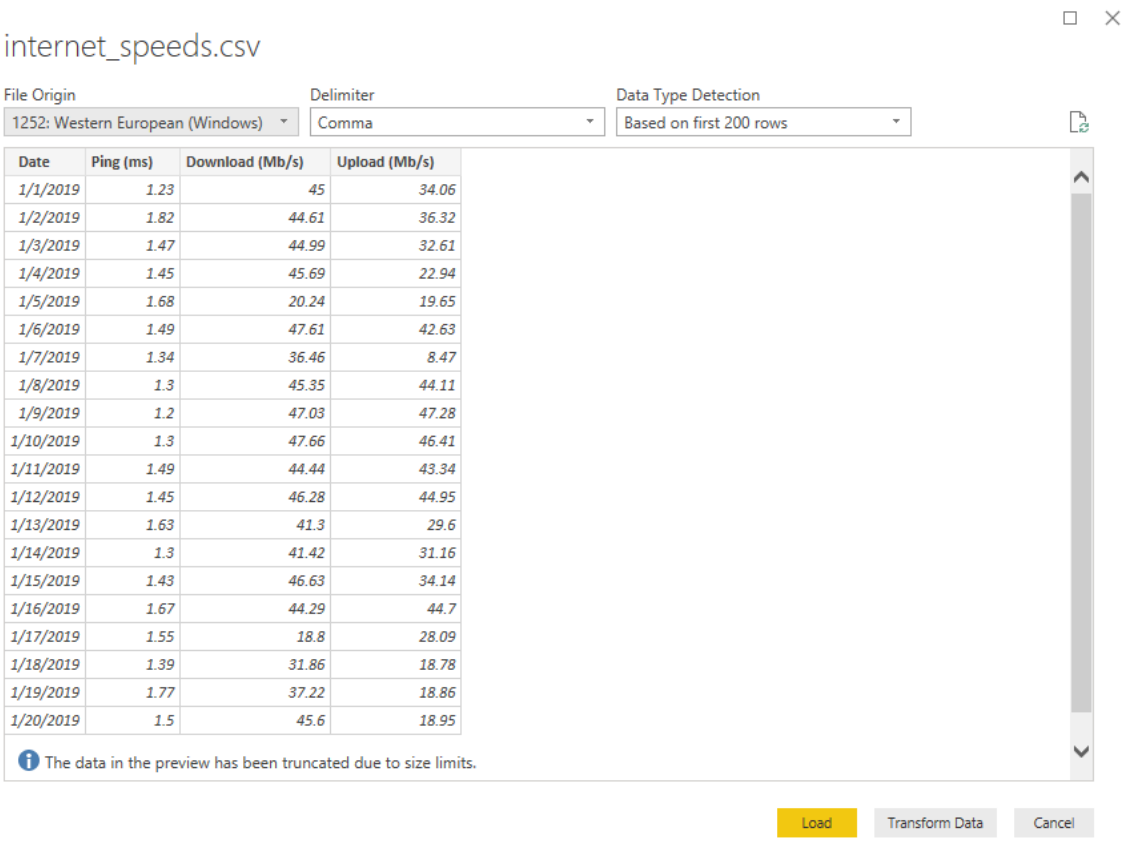
For that, open Power BI and then, in the Home tab of the ribbon bar (the top menu bar), click the Get Data dropdown and choose the Text/CSV option.



Get data from external source

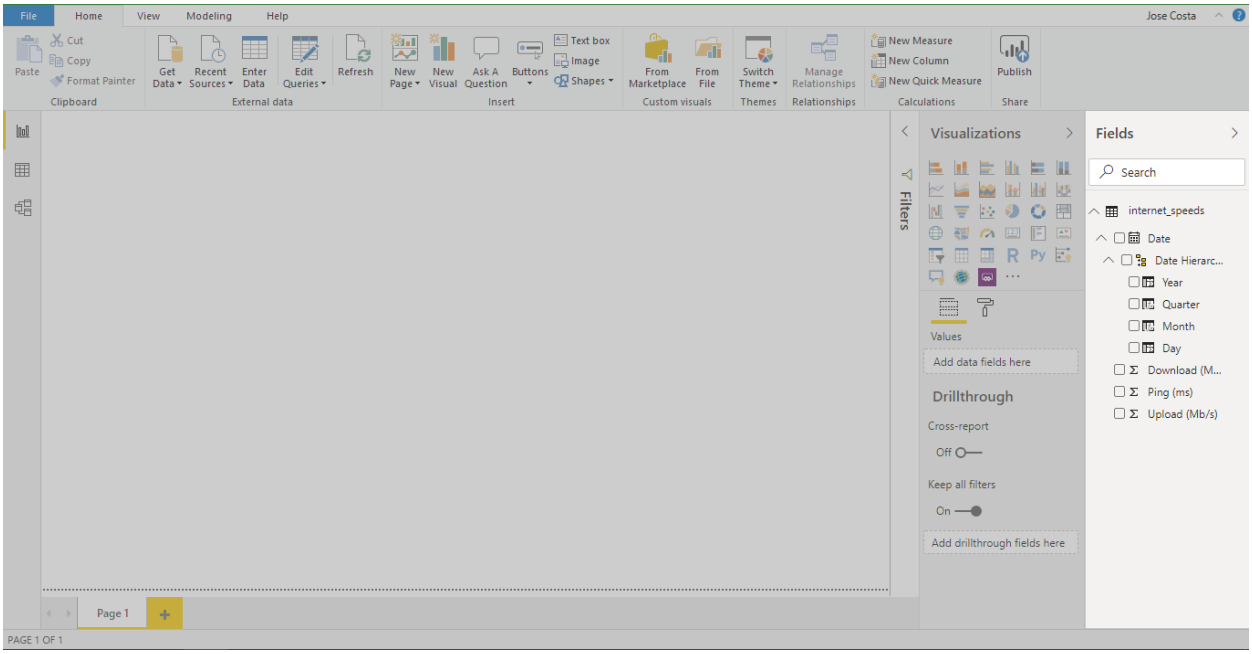
This will prompt you for a text (.txt) or CSV (.csv) file, i.e., the dataset you want to load. Choose the internet_speeds.csv file you have downloaded before.

In the following screen, we can preview the chosen dataset and either choose to load the data as is or modify it before loading it into the report. Since the data has been prepared beforehand, we only need to load it.



Loaded dataset preview window

If we turn our attention to the Fields pane, we can see the internet_speeds dataset and its columns have been loaded.

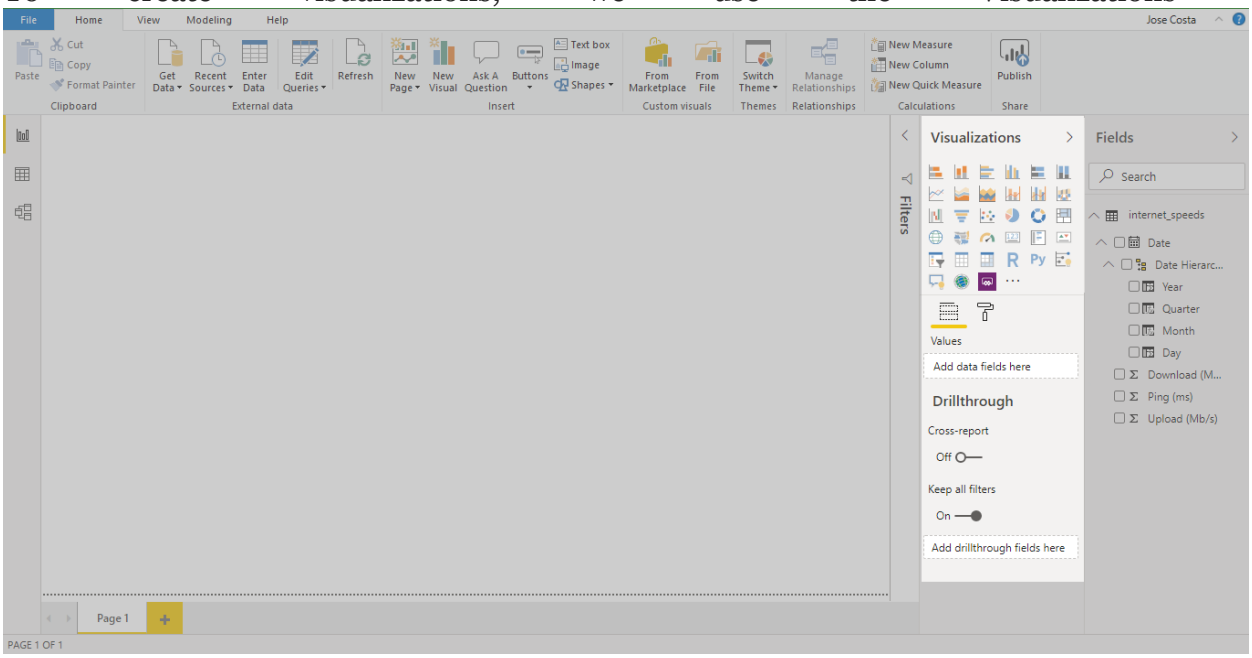


Fields pane

By default, because the “Date” column has values of type date, Power BI has created a hierarchy to separate each component of a date (year, quarter, month and day). We can either use each component separately or use the complete “Date”.
Now that we have our data loaded, we can start working on the dashboard!

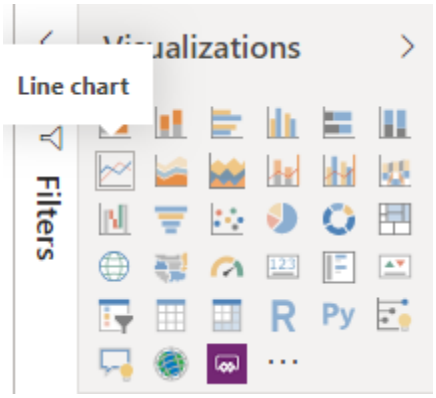
Create the line charts

To create visualizations, we use the Visualizations pane.

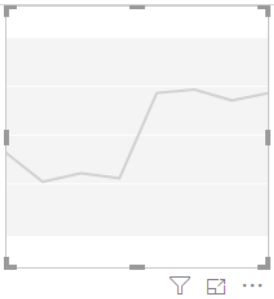


Visualizations pane

The top half shows the available visualizations and that’s where we go to add new visualizations to the canvas. As you can see, Power BI comes with a lot of visualizations and it’s possible to get even more from the Marketplace. The bottom half, we’ll get to it in a second.
Then let’s create our first visualization. Clicking the line chart options creates, as expected, a new empty line chart on the canvas like so:



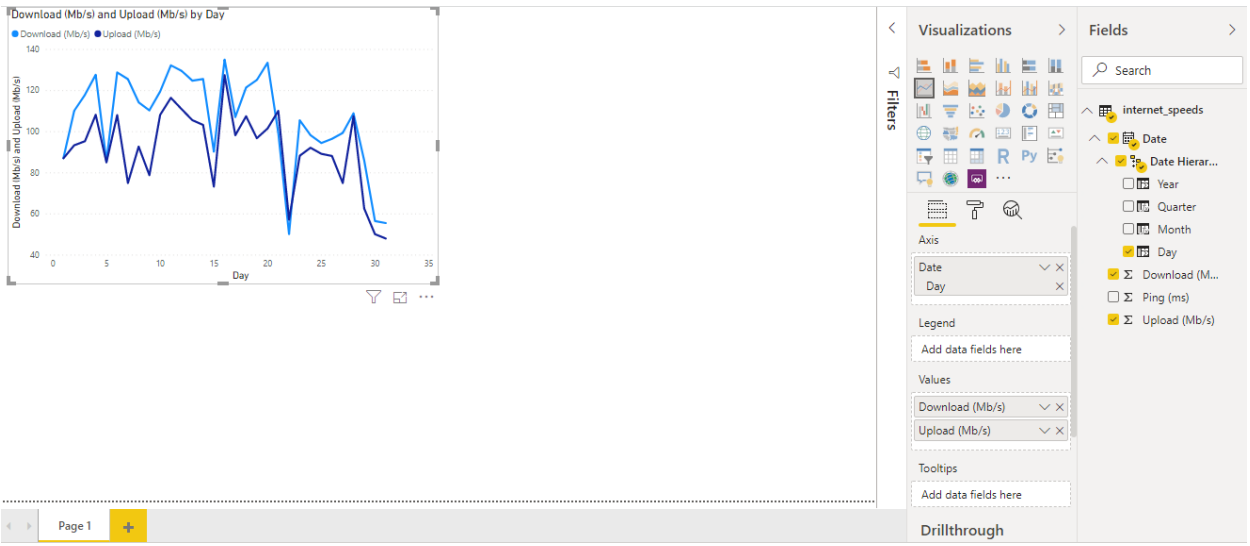
Line chart location



Canvas with empty line chart

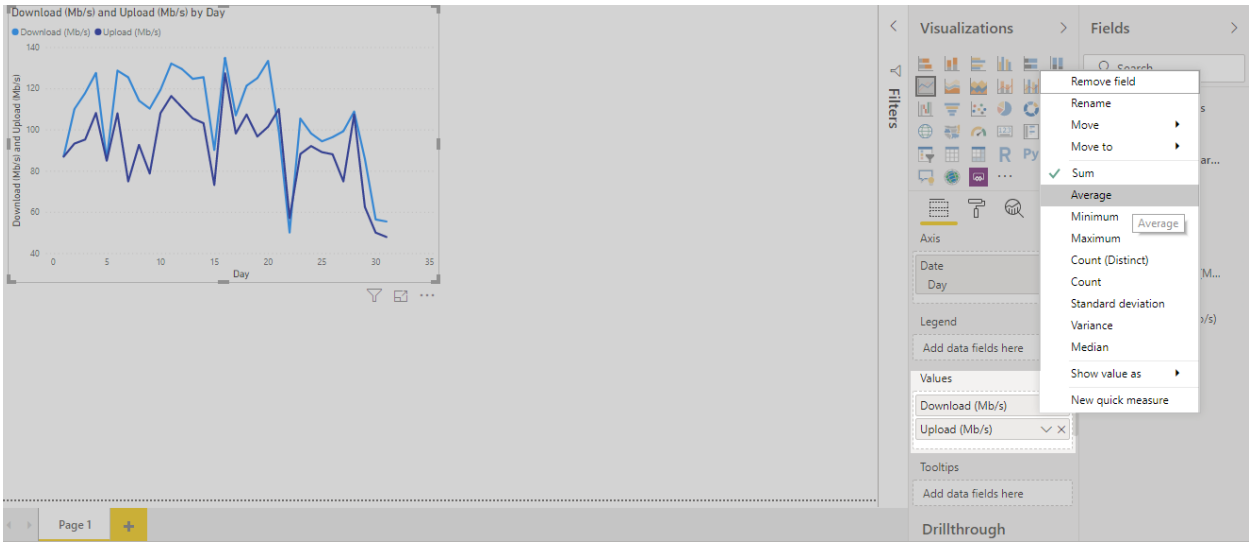
Before adding data to the visualization let me just clarify the purpose of the bottom half of the Visualizations pane. When you click on a visualization on your canvas, you can then access various options and settings for it, separated into three main categories: Fields, Format and Analytics. The first category is about the data used in the visualization, while the second category is used to change its style and format. The third category won't be tackled in this article, but it's very helpful as it can add new information to the charts and even forecast values (not advised for short datasets though).

Back to our first line chart. With this new information, click the line chart to open its options in the Visualization pane. Now we can drag and drop the columns from our dataset into each Field of the chart. Let's add the "Day" from the "Date" column to the Axis of our visualization, and both the "Download (Mb/s)" and "Upload (Mb/s)" to the Values. Oh, and do increase the size of the chart, as the default size is too small (use the resize markings on the border of the chart).



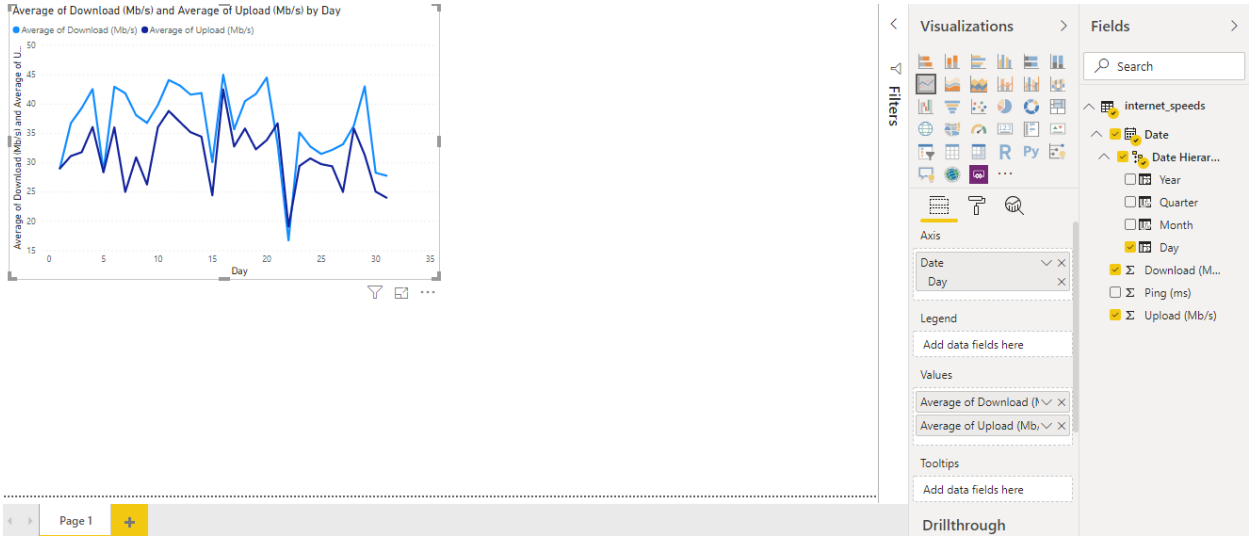
First attempt at the line chart

Ok, the line chart finally displays data but, judging by the values shown on the vertical axis, it doesn't seem like it's showing the average of the download and upload speeds on each day. Actually, by default for numeric values, Power BI displays the sum of the values. But we can easily change that with a right-click of the mouse on the Values of the line chart:



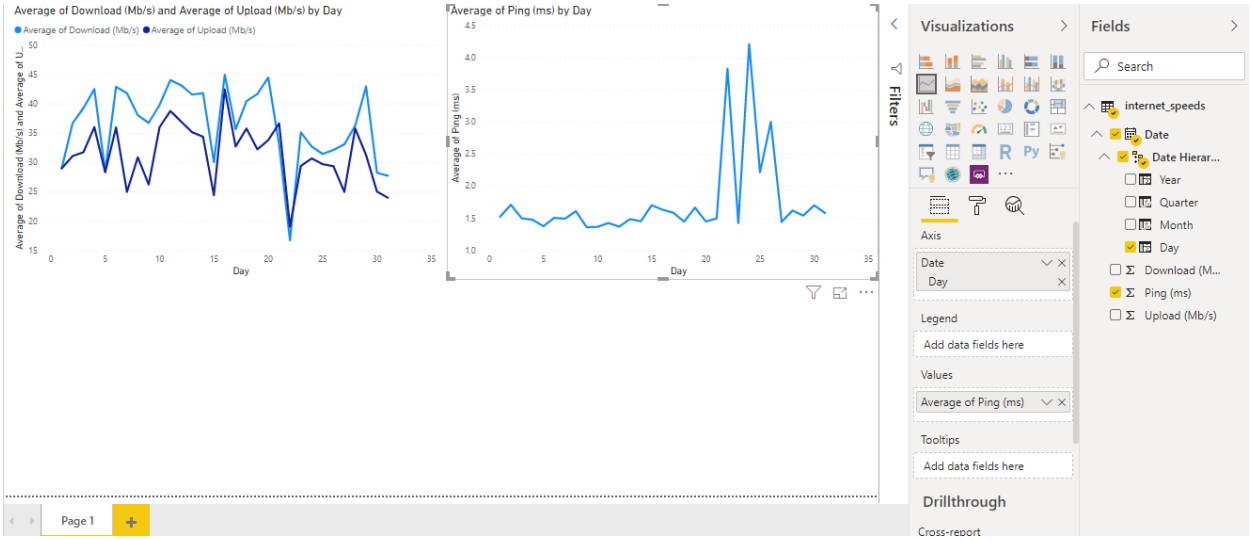
Change the display of the visualization's values

Change it for both the "Download (Mb/s)" and the "Upload (Mb/s)" and our first line chart is finished:



Download and Upload (Mb/s) by Day of Month line chart

Now create a new line chart, to plot the "Ping (ms)" by "Day" again:



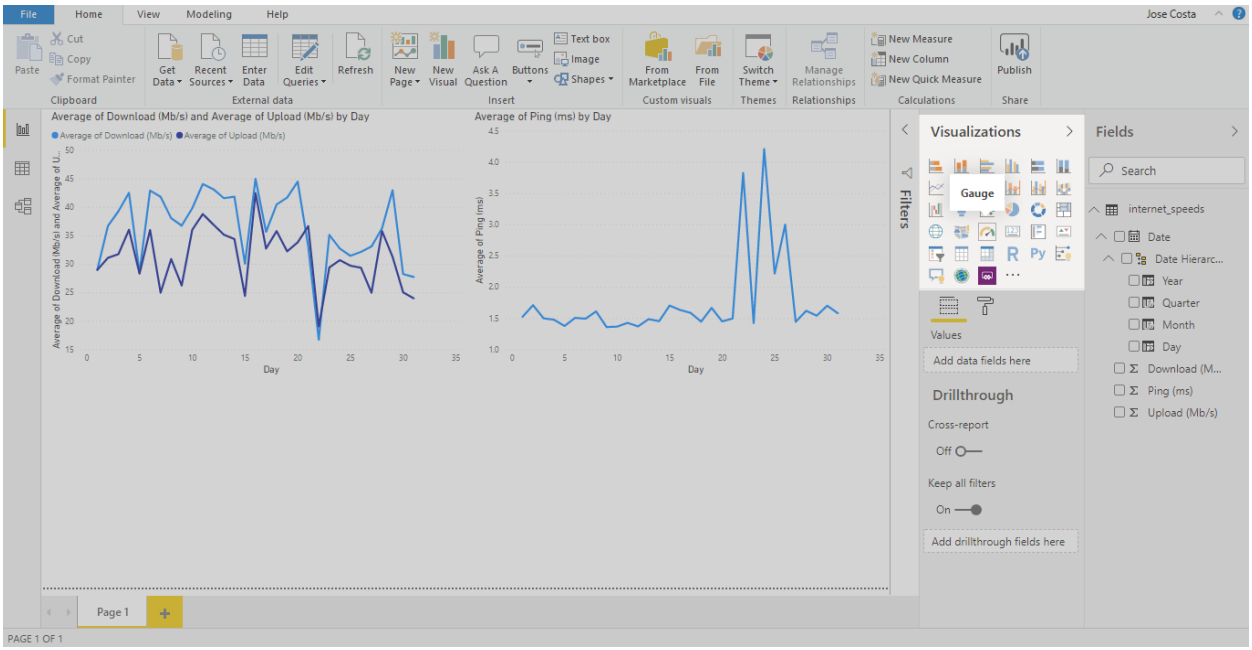
Ping (ms) by Day line chart

Notice how the data available on the Fields pane is ticked on or off depending on the visualization you have selected on the canvas. It's a quick way of finding out which values are being used for each visualization.With this, we've reached the end of the first part. We've created two line charts, one to visualize the evolution of the average download and upload speeds over the course of a month and another for the average ping.

We'll return to these charts later to format them. For now, let's create the gauges.

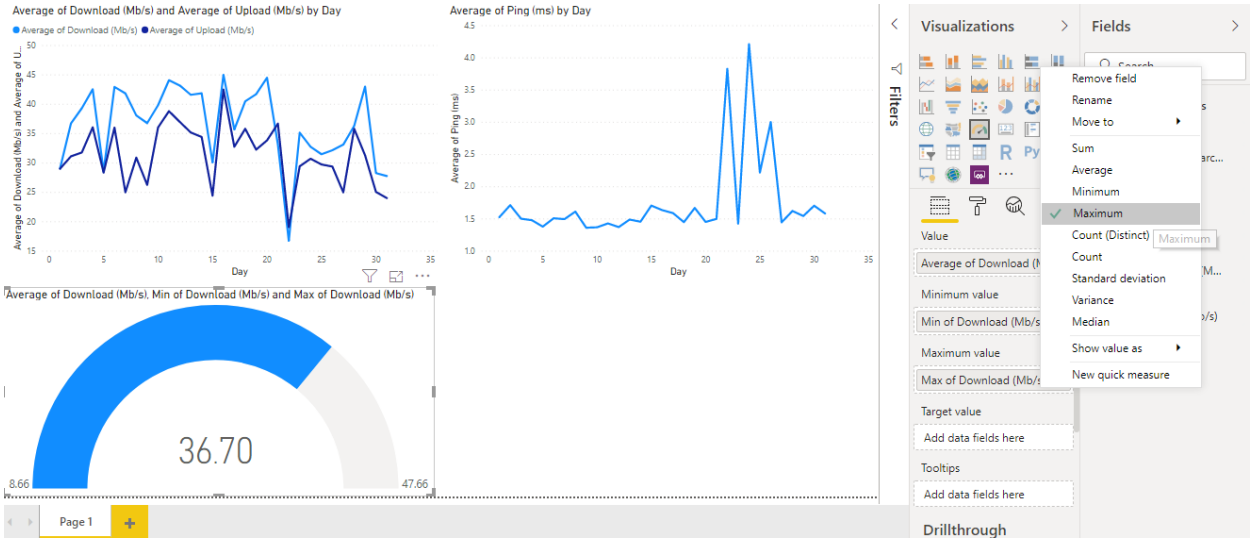
Create the gauges

Now we'll create three gauges, one for each metric of the internet speeds. These will be useful to see where the average download, upload and ping, respectively, stand compared to the minimum and maximum values of each metric.As before, create the gauge visualization for the "Download (Mb/s)" using the option in the top half of the Visualizations pane.



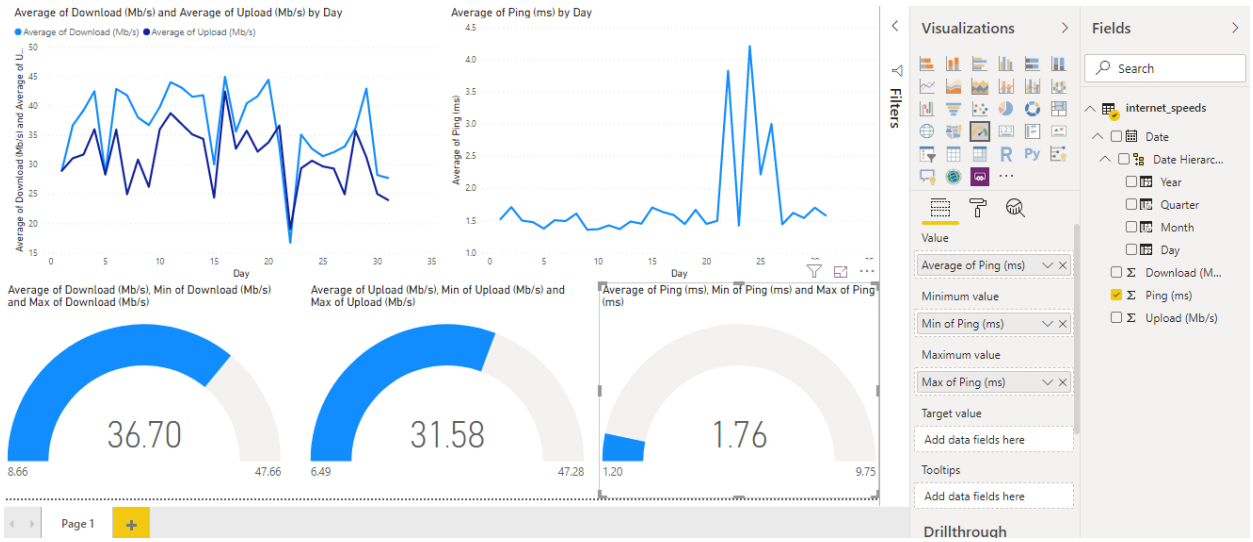
Create a new gauge visualization

And again, like before, we'll drag and drop the data into the Fields of the gauge. Since this first gauge is for the download speeds, we'll use that column for the three Fields of the visualization: Value, Minimum Value and Maximum Value. The difference will be in how we display the data on each Field: just like we changed sum to average in the Values of the line charts, we'll display the average, minimum and maximum values of "Download (Mb/s)", respectively, on each Field.



Change the values of the first gauge

And that's it, this first gauge is ready. Now create two more identical gauges, one to display the "Upload (Mb/s)" and another for the "Ping (ms)".



Gauge visualizations finished

And that's it. All the visualizations are now created, we just need to format them to make the dashboard more readable and its design consistent.

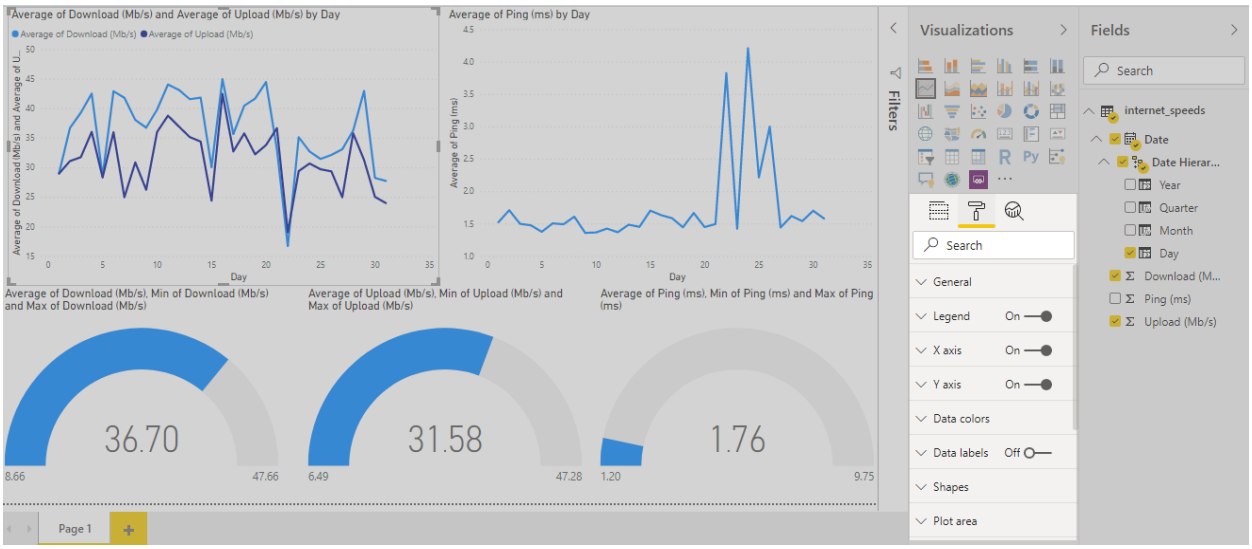
Format the visualizations

This last part of the article is somewhat optional. I've already showed you how to create the visualizations, now I'm going to show you how to make them look better.

I'll show you three main points for this example and then you're free to play around with the dashboard and the data as much as you want to: the title, the data colors and the axes. Just keep in mind, format these visualizations as much as you want to, the objective is for you to learn more about Power BI, the following are only my suggestions.

To keep the dashboard consistent, we need to use the same colors for each metric, that is, we should be using the same color for every instance of "Download (Mb/s)", "Upload (Mb/s)" and "Ping (ms)". Plus, we can also adjust the limits of the axes of the line charts slightly. For example, a month doesn't have more than 31 days, so the upper limit of the X axis should be 31. Lastly, the title of the visualizations is the last thing we'll change as the default title is too verbose.

For any format changes we want to make, we need to first select our target visualization and turn our attention to the second category in the bottom half of the Visualizations pane: Format.

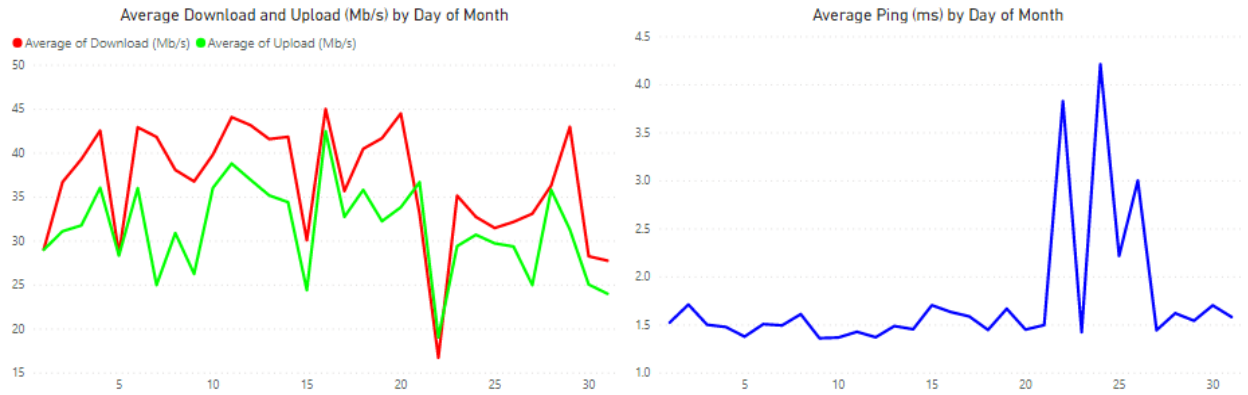


Format options for visualizations

In here, there are plenty of changes one can make to visualizations and I do encourage you to try them out and see what you like best for this dashboard.

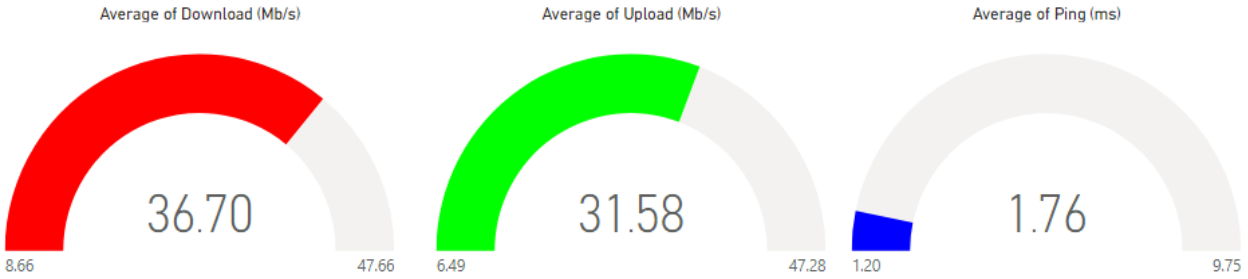
My suggestions, in the case of the line charts, are to change the Start and End values of the X axis to 1 and 31, respectively, and disable the title of both the X and Y axes. In the Data colors options, choose different colors for each metric (three metrics, three colors). Lastly, in the Title options, change the text displayed and change the alignment to center. I think "Average Download and Upload (Mb/s) by Day of Month" and "Average Ping (ms) by Day of Month" are good titles.

If you were following along with the changes suggested in the previous paragraph, your line charts should look something like this now:



Formatted line chart visualizations

For the gauges, there's not really much I want to suggest for format changes. Just change the Data colors to the ones used in the line charts as well as the titles.



Formatted gauge visualizations

Conclusions

And that's it from me. Now go experiment with the Format options, try out different visualizations (tip: select one of the visualizations created and then in the Visualizations pane select a different visualization), try out different combinations of the metrics, do whatever your curiosity asks for to learn more about Power BI. P.S.: the graphical interface of this application is in line with that of the other Microsoft Office products, like Word and Excel, so a lot of the functionality in Power BI is extremely intuitive :)

Conclusion:

In this assignment, we have learned the importance of Python and how to use it with Tableau. In the first section, we extracted Goodreads user data using developer API and converted the XML data into a clean and structured Pandas dataframe. In the second part, we used clean data to create both simple and complex data visualizations.

The combination of Python with Tableau opens a whole new world of possibilities. You can integrate data pipelines, implement machine learning models, run complex statistical analyses, and perform various tasks that are impossible to run on Tableau alone.

You can run the Python code on DataCamp's **Workspace** for free, and it comes with all the necessary packages for you to run this example. To create Tableau visualization, we have used a free version of Tableau called **Tableau Public**.

If you are new to Python and want to learn more about functionalities and syntax, check out **Introduction to Data Science in Python** course. Also, you can master the basics of Tableau visualization and customization by taking **Tableau Fundamentals** skill track. It consists of 5 courses that cover an **introduction to Tableau**, data analysis, creating interactive dashboards, working on a case study, and connecting multiple data sources.

Source Code and Output /Screenshots:

(To be provided by the student)

```
//Source code and output/screenshot should be available here
```

Useful Resource:

https://www.datacamp.com/tutorial/visualizing-data-with-python-and-tableau-tutorial

https://help.tableau.com/current/pro/desktop/en-us/export.htm

https://hevodata.com/learn/power-bi-data-model/

https://www.simplilearn.com/tutorials/power-bi-tutorial/what-is-power-bi

https://www.linkedin.com/pulse/mastering-data-analytics-visualization-power-bi-tools-best/

Exercise Questions

1. How did you choose the dataset for this assignment? What criteria influenced your decision?
2. Provide a brief overview of the selected dataset, including its source, size, and a description of the variables.
3. What challenges did you encounter during the initial exploration of the dataset, and how did you address them?
4. Describe the steps you took to clean the dataset. How did you handle missing values and duplicates?
5. Explain the pre-processing techniques applied to make the data suitable for modeling. Give specific examples.
6. Discuss the decisions made during the data modeling phase. How did you determine relationships between different entities in the dataset?
7. What considerations did you take into account while choosing data types for variables, and how did it impact the overall data model?

8. Provide an overview of the normalized database schema you created. Why did you choose this specific normalization approach?
9. Walk through the process of importing the pre-processed data into Tableau.
10. Explain the design principles followed while creating the Tableau dashboard. What types of visualizations did you include, and why?
11. Discuss any challenges faced during the Tableau dashboard creation and how you overcame them.
12. Outline the steps involved in importing the pre-processed data into PowerBI.
13. Compare the features and functionalities of PowerBI used in the dashboard creation with those of Tableau.
14. How did you ensure consistency in the key insights presented in both the Tableau and PowerBI dashboards?
15. Reflect on the strengths and weaknesses of Tableau and PowerBI for this specific assignment. Which tool did you find more suitable for your tasks, and why?
16. Discuss any performance differences between Tableau and PowerBI during data import and visualization rendering.
17. How did you document the entire process, and why is documentation important in a data modeling and visualization project?
18. Share the key findings and insights presented in your brief presentation. How did you prioritize the information to include?
19. Describe a specific challenge you faced during the assignment and the strategies you used to overcome it.
20. Reflect on the most significant learnings and insights gained from performing data modeling and creating dashboards using Tableau and PowerBI.
21. Provide recommendations for additional analyses or improvements that could enhance the effectiveness of the dashboards.
22. Suggest potential features or enhancements that could be implemented in Tableau and PowerBI to improve the overall user experience.

These questions cover a wide range of topics related to dataset selection, data modeling, dashboard creation, tool comparison, and reflection on the overall learning experience.

Experiment No 12

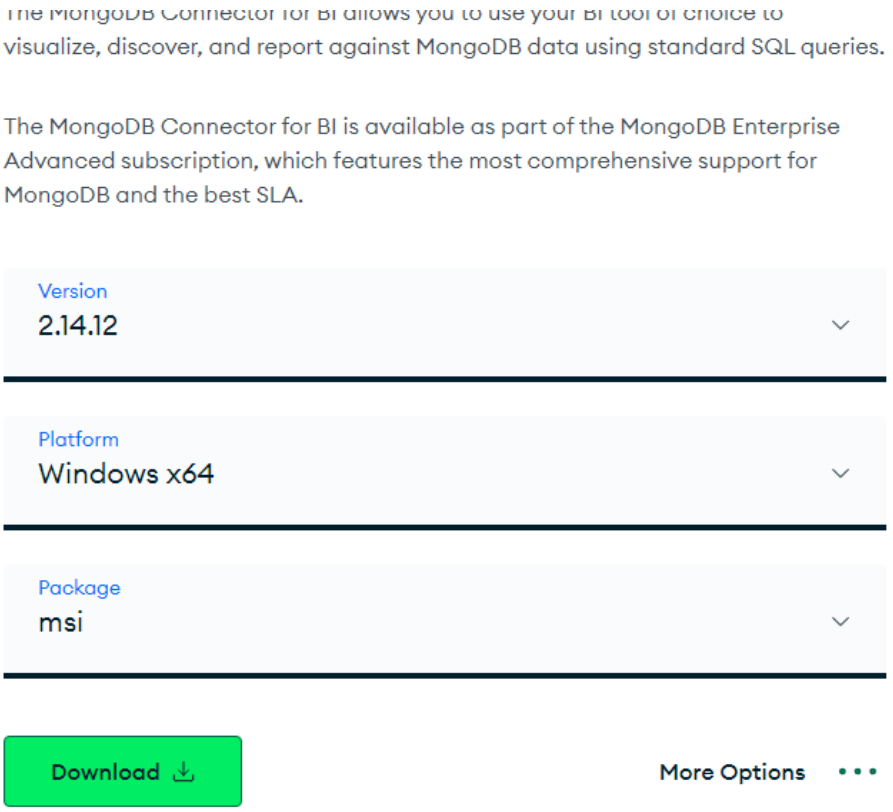
Experiment Title: Download any dataset from NoSQL and perform data modeling and create a basic dashboard using Tableau and PowerBI.

Objective: To Import the dataset from mongodb and load into the PowerBi then create the dashboard

Theory: Mongodb is the NoSQL DataBase Management System. We have to create the database then create the collection in it and insert a few documents in the collection. Import the documents into powerBi then create the basic dashboard for the collection created.

To perform the above tasks following tools/ drivers are required.

1. PowerBi Desktop
2. MongoDB Server
3. MongoDB Shell
4. MongoDB BI Connector (MongoDB BI Connector Download | MongoDB)



5. MongoDB ODBC Driver (Releases · mongodb/mongo-bi-connector-odbc-driver (github.com))

v1.4.5 Latest

Version 1.4.5 fixes a stack overflow happening when allocating more memory than available for the client application (Bug# 01141507).

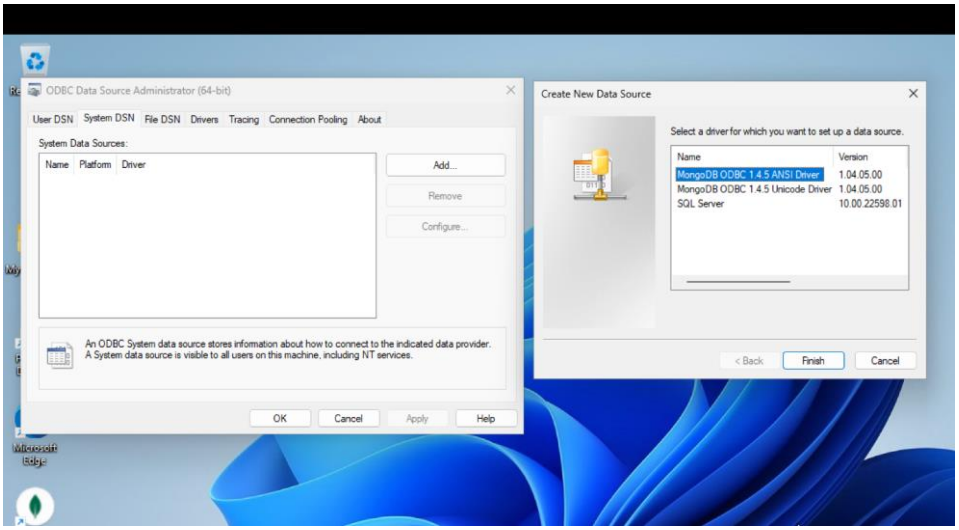
Please refer to the README and to the BI Connector reference documentation for usage instructions.

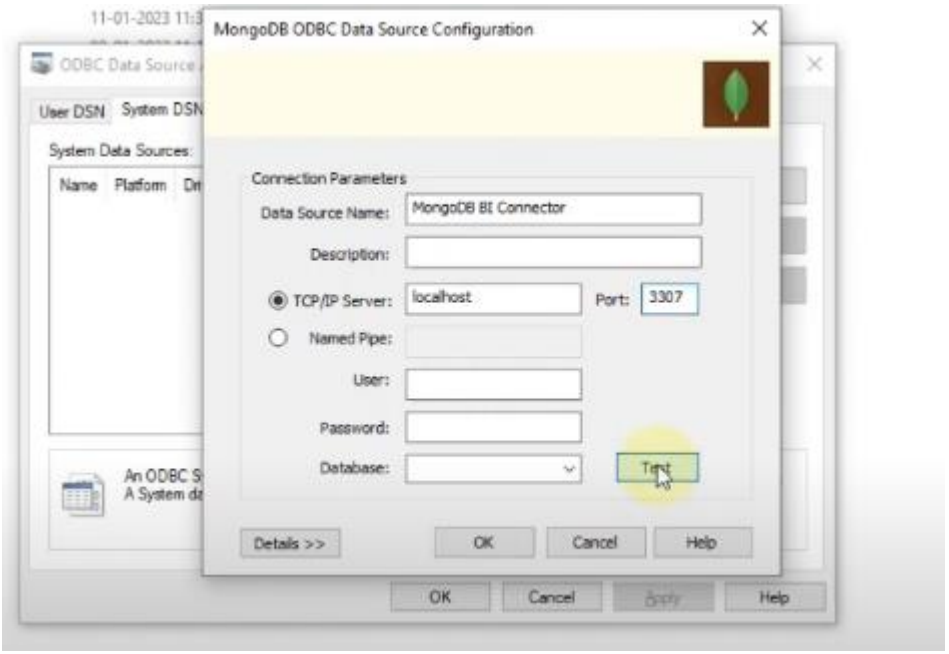
▾ Assets 8

mongodb-connector-odbc-1.4.5-macos-64-bit.dmg	27.3 MB	Aug 24, 2023
mongodb-connector-odbc-1.4.5-rhel-7.0-64.tar.gz	26 MB	Aug 24, 2023
mongodb-connector-odbc-1.4.5-ubuntu-14.04-64.tar.gz	25.9 MB	Aug 24, 2023
mongodb-connector-odbc-1.4.5-ubuntu-16.04-64.tar.gz	26 MB	Aug 24, 2023
mongodb-connector-odbc-1.4.5-win-32-bit.msi	26.7 MB	Aug 24, 2023
mongodb-connector-odbc-1.4.5-win-64-bit.msi	27.4 MB	Aug 24, 2023
Source code (zip)		Aug 24, 2023
Source code (tar.gz)		Aug 24, 2023

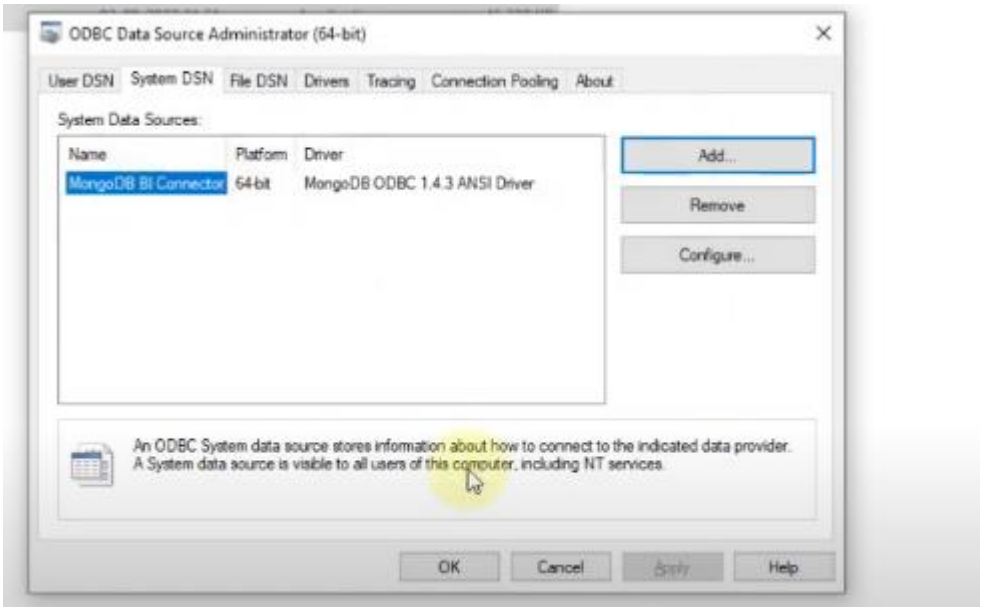
Follow the below steps to establish a connection between PowerBi and MongoDb

1. Run the ODBC Data source
2. Choose System DNS Tab Click on ADD select MongoDB ODBC ANSI Driver and Complete the Configuration



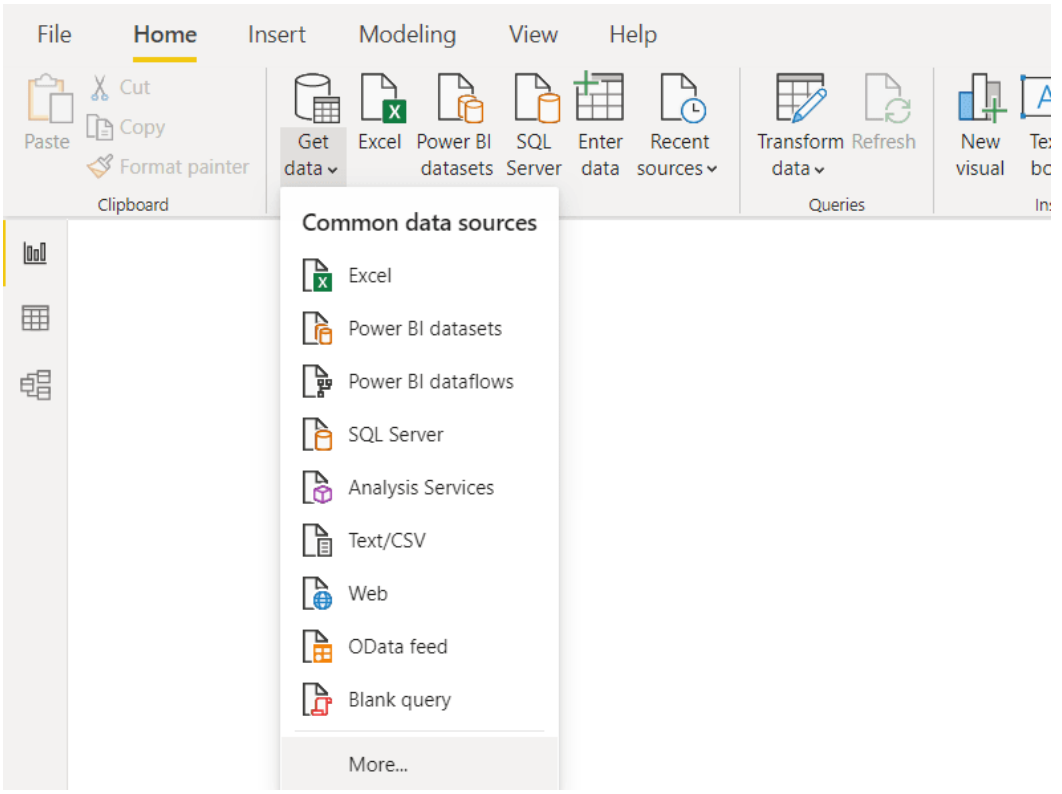


- Run the mongodb
- Test the Connection
- Port number should be the same on which mongodb is running.
- Upon successful connection the following screen will be displayed with the Data source name.

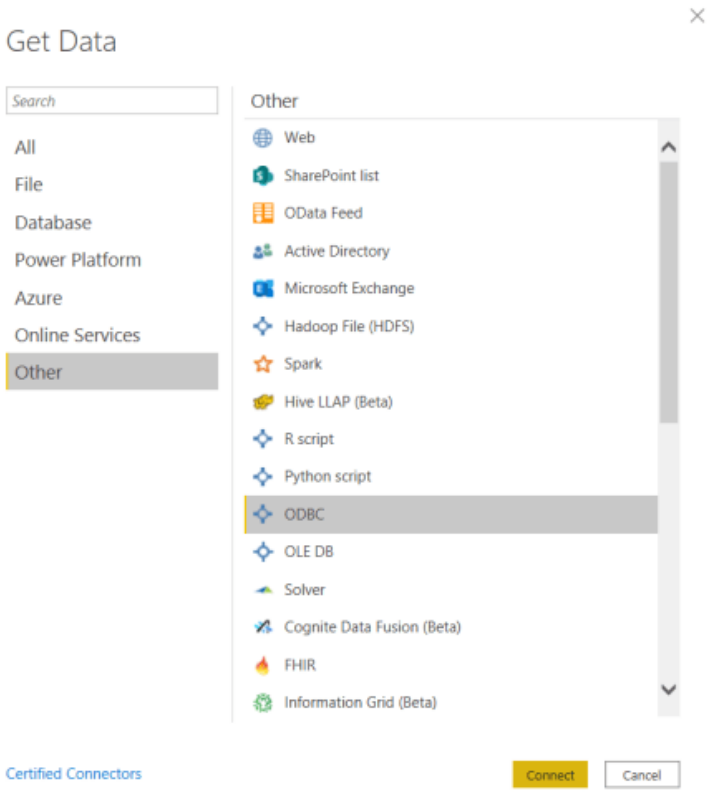


3. Create a database in mongodb

4. Launch the PowerBi application

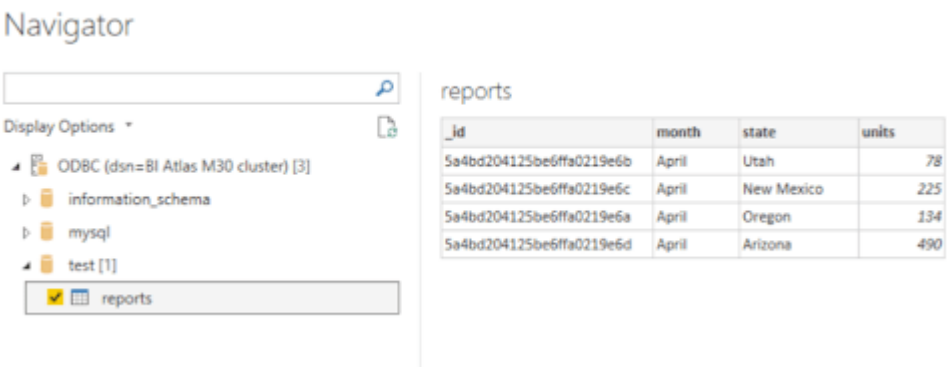


Click on Get Data then select More



Click on Other then select ODBC then click on Connect

Choose the Datasource name you created.



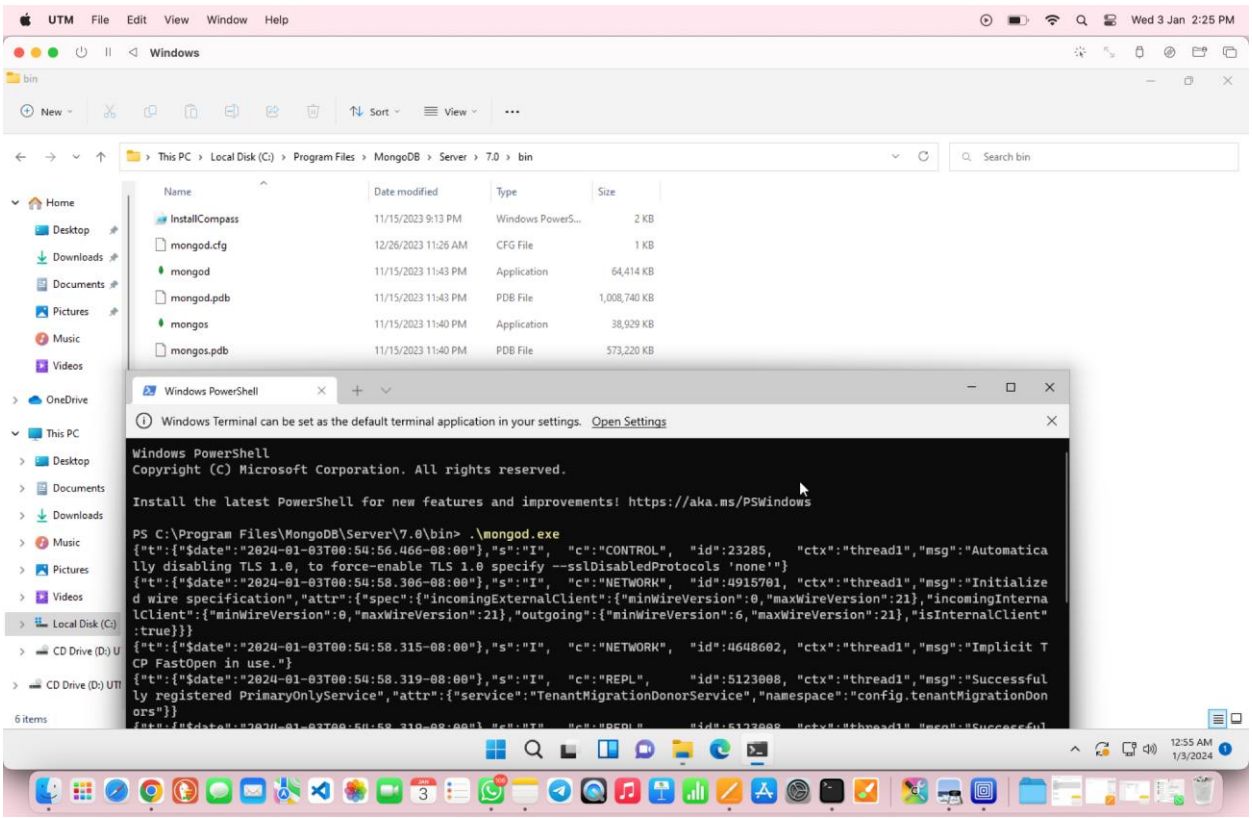
List of databases and its collections will be loaded.

Now perform the data visualization as per the requirements.

1. Create the Exam Database in mongodb
2. Create the collection (Student_Result)
3. Insert few documents (Name, ID, Different Course with Marks, Resul (Pass/Fail)
4. Create the Dashboard using powerBi (Barcharts, Pie charts etc which visualize the different courses results)

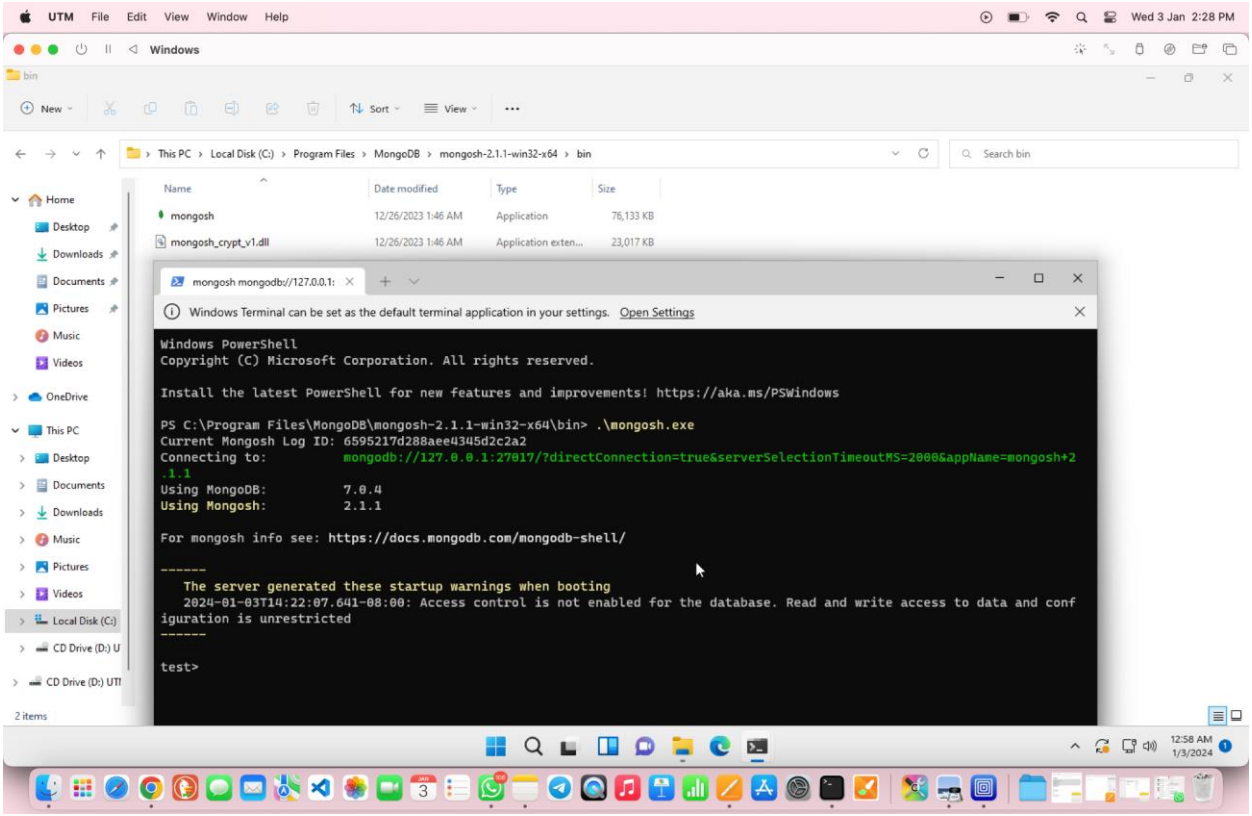
Steps for Execution

- Open Terminal
- Go to C:\Program Files\MongoDB\Server\7.0\bin\./mongo.exe
- Run ./mongo.exe (Run the Mongodb Server)



Run the Mongodb Shell

Go to C:\Program Files\MongoDB\mongosh-2.1.1-win32-x64\bin\.mongosh.exe



Create the Database (Exam) and Insert the Student_Result Collection

test> use exam;

test> use exam;
switched to db exam
exam> db.student.insert({Rno:1,Name:"Sai",Course:"DVM",Marks:45,Result:"Pass"});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
 acknowledged: true,
 insertedIds: { '0': ObjectId('65952255288aee4345d2c2a3') }
}
exam> db.student.insert({Rno:1,Name:"Sai",Course:"IoT",Marks:55,Result:"Pass"});
{
 acknowledged: true,
 insertedIds: { '0': ObjectId('65952399288aee4345d2c2a4') }
}
exam> db.student.insert({Rno:1,Name:"Sai",Course:"DBMS",Marks:65,Result:"Pass"});
{
 acknowledged: true,
 insertedIds: { '0': ObjectId('659523a7288aee4345d2c2a5') }
}
exam> db.student.insert({Rno:2,Name:"Rahul",Course:"DBMS",Marks:35,Result:"Fail"})
{
 acknowledged: true,
 insertedIds: { '0': ObjectId('659523ce288aee4345d2c2a6') }
}
exam> db.student.insert({Rno:2,Name:"Rahul",Course:"IoT",Marks:45,Result:"Pass"})

```
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('659523e5288aee4345d2c2a7') }
}
exam> db.student.insert({Rno:2,Name:"Rahul",Course:"DVM",Marks:65,Result:"Pass"})
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('659523f4288aee4345d2c2a8') }
}
exam> db.student.find().pretty()
[
  {
    _id: ObjectId('65952255288aee4345d2c2a3'),
    Rno: 1,
    Name: 'Sai',
    Course: 'DVM',
    Marks: 45,
    Result: 'Pass'
  },
  {
    _id: ObjectId('65952399288aee4345d2c2a4'),
    Rno: 1,
    Name: 'Sai',
    Course: 'IoT',
    Marks: 55,
    Result: 'Pass'
  },
  {
    _id: ObjectId('659523a7288aee4345d2c2a5'),
    Rno: 1,
    Name: 'Sai',
    Course: 'DBMS',
    Marks: 65,
    Result: 'Pass'
  },
  {
    _id: ObjectId('659523ce288aee4345d2c2a6'),
    Rno: 2,
    Name: 'Rahul',
    Course: 'DBMS',
    Marks: 35,
    Result: 'Fail'
  },
  {
    _id: ObjectId('659523e5288aee4345d2c2a7'),
    Rno: 2,
    Name: 'Rahul',
    Course: 'IoT',
    Marks: 45,
    Result: 'Pass'
  },
```

```
  {
    _id: ObjectId('659523f4288aee4345d2c2a8'),
    Rno: 2,
    Name: 'Rahul',
    Course: 'DVM',
    Marks: 65,
    Result: 'Pass'
  }
]
exam>
```

1. Load the Above exam dataset into PowerBi
2. Create the Dashboard for the following tasks
    a. Visualize the Result of each course (Bar chart)
    b. Display the highest marks in each course (Pie Chart)
    c. Display the Pass percentage in each course (Bar Chart)

Conclusion:

In this assignment how to connect to the powerBi and mongodb database .

Useful Resource:

- https://www.mongodb.com/try/download/bi-connector
- https://github.com/mongodb/mongo-bi-connector-odbc-driver/releases/
- https://www.linkedin.com/learning/power-bi-essential-training-17362720/get-power-bi-for-mobile?u=157124841

Exercise Questions

XYZ Company, a retail business, is looking to gain deeper insights into its sales data to optimize business strategies, enhance decision-making, and improve overall performance. The company's sales data is currently stored in a MongoDB database. The goal is to leverage Power BI to connect to the MongoDB database, import relevant sales data, and create insightful visualizations.

Objectives:

Connect to MongoDB Database:

- Establish a connection between Power BI and the MongoDB database.
- Retrieve sales data from the MongoDB collection.

Visualize Sales Performance:

- Create visualizations that provide insights into sales performance.
- Explore key metrics such as total sales, sales by product category, and sales over time.