Customer Propensity – Google Analytics Sample Data:

Introduction:

This assignment is to design, build, evaluate, and deploy an ML model to predict customer propensity to perform the “Add To Cart” action. The model should accurately predict whether this action will be performed or not based on the features selected or developed from the given dataset.

Dataset Information:

https://support.google.com/analytics/answer/7586738

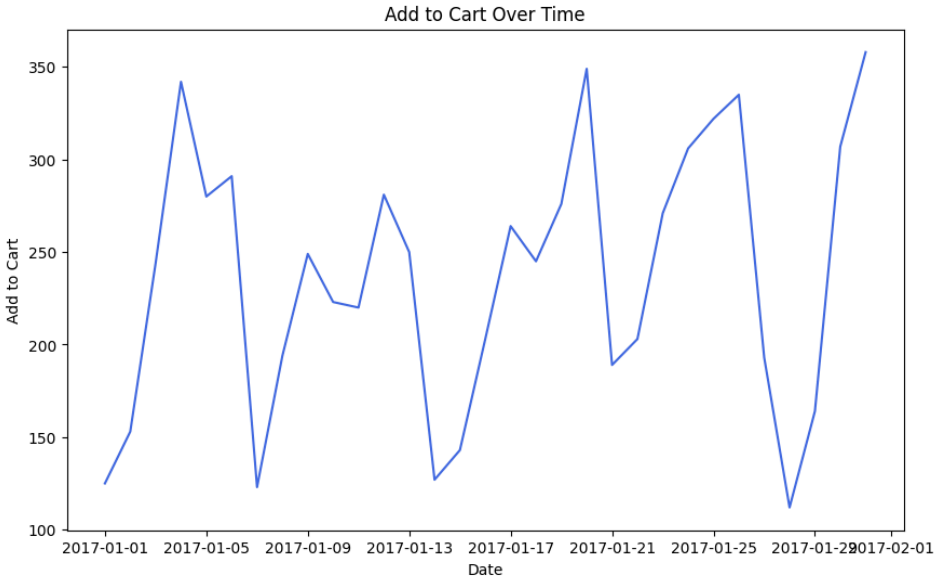
Data Analysis Report:

This report provides an analysis of Google Analytics data from January 2017. The analysis covers three main aspects:

Add to Cart Over Time: Examining the trend of add to cart actions over the month.
Number of Visitors by Channel Grouping: Understanding the distribution of visitors across different channel groupings.
Top 10 Countries by Customers: Identifying the top 10 countries with the highest number of customers.
Time Spent Before Add to Cart: Analyzing the time spent by customers before adding items to the cart.

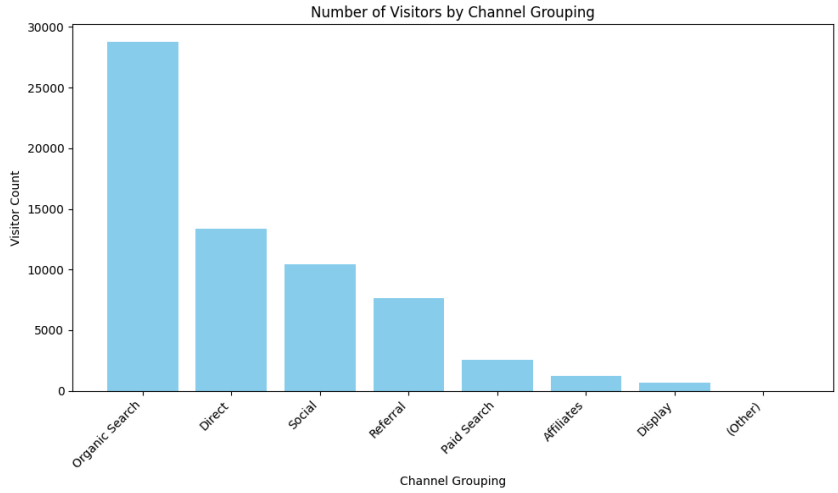
1. Add to Cart Over Time

The line plot below illustrates the total count of add to cart actions over the course of January 2017. The count appears to fluctuate, showing potential patterns or anomalies that may warrant further investigation.



2. Number of Visitors by Channel Grouping

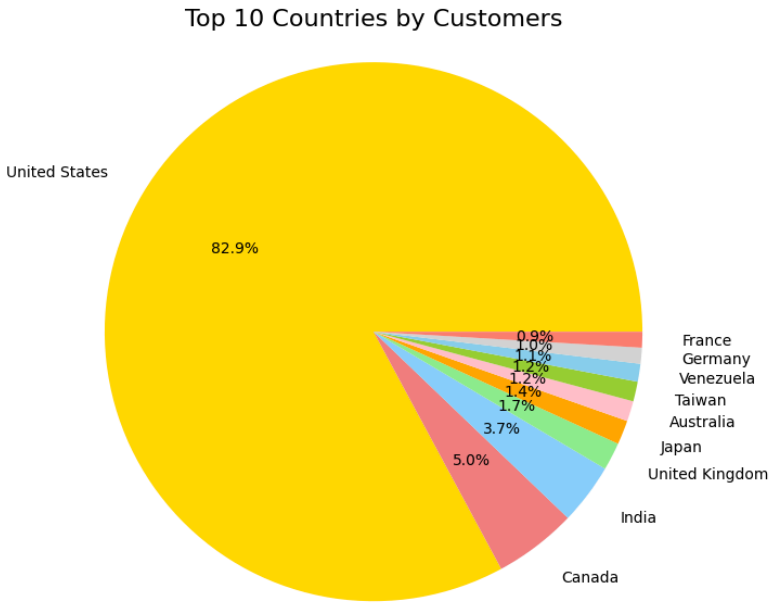
The bar plot displays the number of visitors categorized by different channel groupings. This analysis helps understand which channels are driving the most traffic to the website.



3. Top 10 Countries by Customers

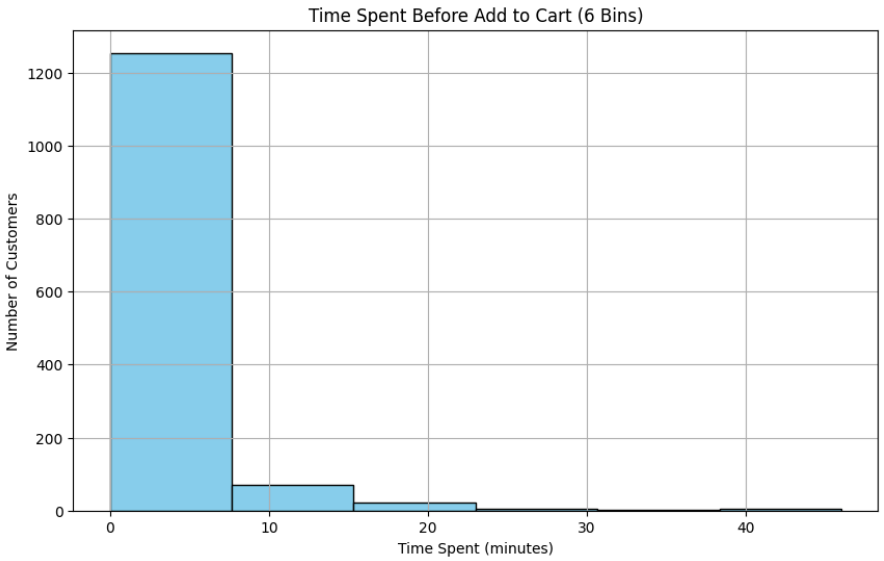
A pie chart showcases the distribution of customers across the top 10 countries with the highest customer counts. This information is crucial for understanding the

geographical spread of the customer base.



4. Time Spent Before Add to Cart

The histogram below represents the time spent by customers before adding items to the cart. This analysis helps in understanding customer behavior and preferences regarding the decision-making process leading to a purchase.

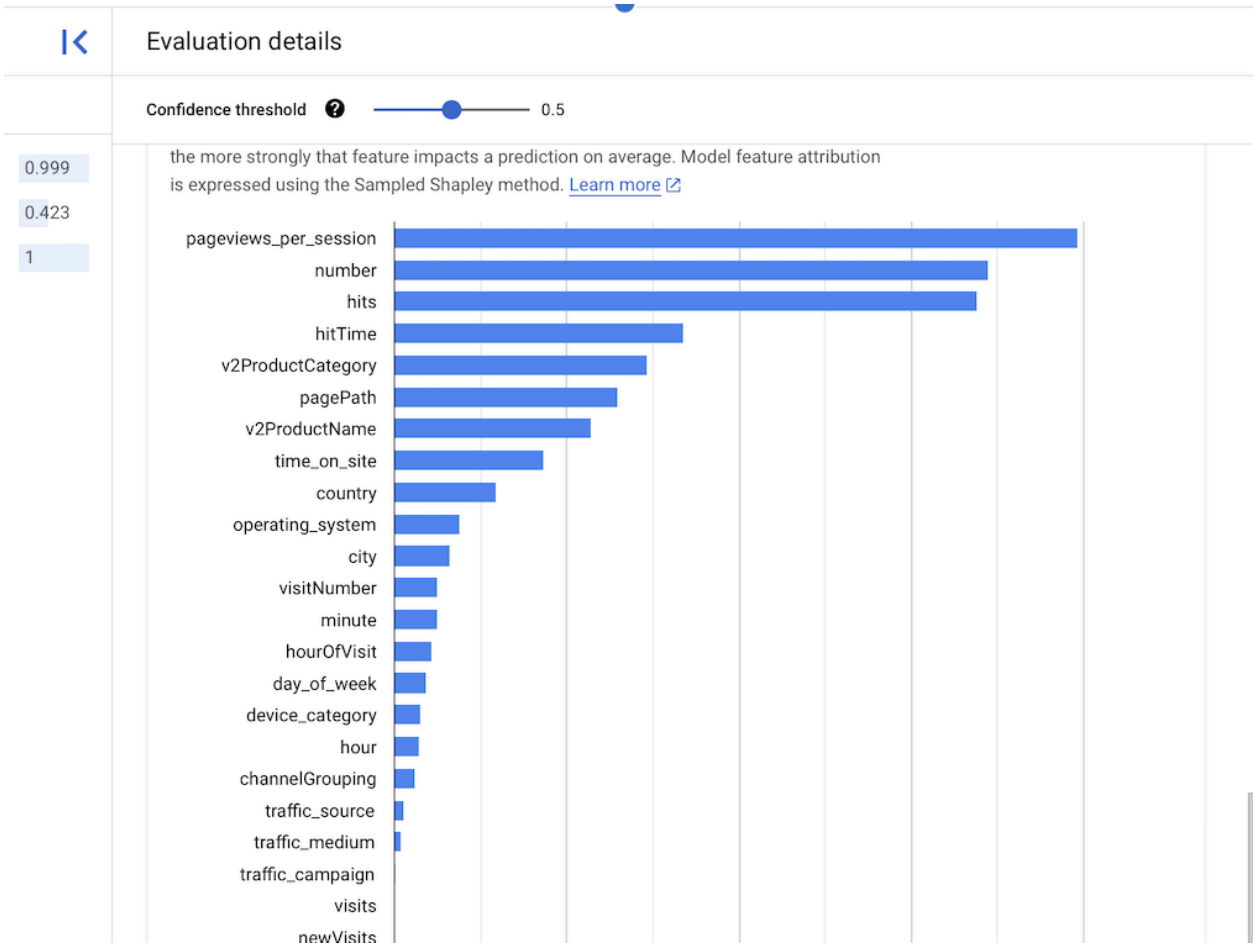


DataPreprocessing:

Checked for null values and omitted the columns that are mostly null. Omitted the columns that are deprecated in the schema. Features that contain “demo not available in demo set” are removed. After the initial cleaning, I decided to go with 17 features.

Feature Engineering:

Extracted day of weel and did pageviews_per_session / visits as Average Page View per visits. And hits/ visits as average hits per visit.



Model Building:

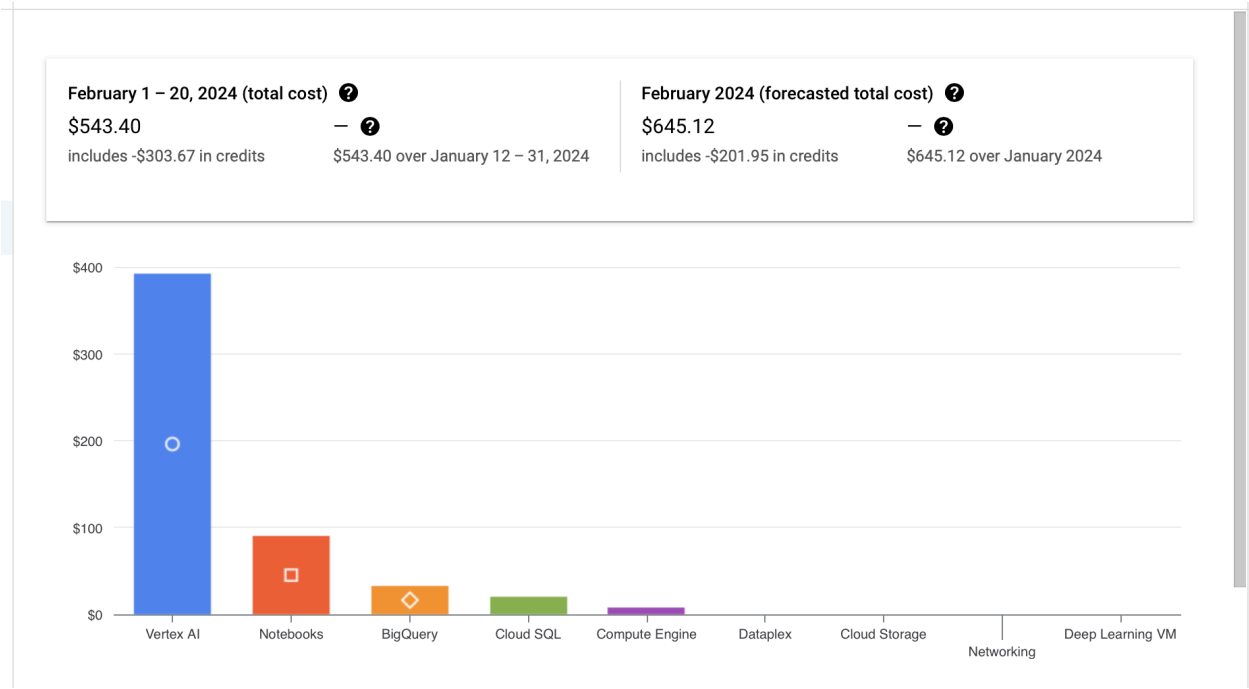
Used ML XGBoost Boosted tree classifier.Splitted the data into 80%Train 10% evaluation and 10% Test.Evaluation Matrix precision ,roc_auc,f1_score,accuracy.Also ran autoML model and did batch prediction on that.

Deploying:

- Ensure that your model is trained and saved in a format compatible with deployment.The model is already saved in BigQuery ML format.
- Use the `!bq` command to extract the model from BigQuery ML. Set the destination format to `ML_XGBOOST_BOOSTER` and specify the destination directory for the exported model.
- Define the serving container image URI where the model will be deployed. In this case, the image URI is specified as `'us-docker.pkg.dev/vertex-ai/prediction/xgboost-cpu.1-4:latest'`.
- Use the Vertex AI Python SDK to upload the model artifact to Vertex AI. Provide the display name for the model, artifact URI, and serving container image URI.
- Deploy the model to an endpoint using the `deploy()` method of the uploaded model object.
- Specify the traffic split, machine type, and any other necessary configurations.
- Prepare the data for prediction. This may involve preprocessing steps such as encoding categorical features.
- Define functions for preprocessing data and building feature encoders.
- Extract test data from BigQuery or any other data source.
- Convert the test data to a suitable format for prediction, such as a list of dictionaries.
- Apply the preprocessing functions defined earlier to transform categorical features and prepare numerical instances for prediction.
- Use the deployed endpoint to generate predictions for the test data.

Cost Analysis:

Overall the cost was $543:40



VertexAI Instances: $27

Node hours: $30

Total:$57