MARKET BASKET INSIGHTS

Includes:

1. *Feature engineering*
2. *Visualization*
3. *Evaluation*

FEATURE ENGINEERING

Feature engineering typically involve creating new features or transforming Existing once to improve the performance of a machine learning model this specific code for feature engineering can vary widely depending on data set .

some of the main reasons include:

1.Improve User Experience: The primary reason we engineer features is to enhance the user experience of a product or service. By adding new features, we can make the product more intuitive, efficient, and user-friendly, which can increase user satisfaction and engagement.

2.Competitive Advantage: Another reason we engineer features is to gain a competitive advantage in the marketplace. By offering unique and innovative features, we can differentiate our product from competitors and attract more customers.

3.Meet Customer Needs: We engineer features to meet the evolving needs of customers. By analyzing user feedback, market trends, and customer behavior, we can identify areas where new features could enhance the product's value and meet customer needs

4.Increase Revenue: Features can also be engineered to generate more revenue. For example, a new feature that streamlines the checkout process can increase sales, or a feature that provides additional functionality could lead to more upsells or cross-sells.

5.Future-Proofing: Engineering features can also be done to future-proof a product or service. By anticipating future trends and potential customer needs, we can develop features that ensure the product remains relevant and useful in the long term.

DATACOLLECTION

Gather transaction data that includes information on what items were purchased together. This can be obtained from point-of-sales system or e-commerce platforms.

Data collection is a systematic process of gathering observations or measurements. Whether you are performing research for business, governmental or academic purposes, data collection allows you to gain first-hand knowledge and original insights into your research

While methods and aims may differ between fields, the overall process of data collection remains largely the same. Before we begin collecting data, you need to consider

i)The **aim** of the research

ii)The **type of data** that you will collect

iii)The **methods and procedures** you will use to collect, store, and process the data

```
import pandas as pd
data={
    "BillNo":[536365,536366,536367.....]
    "Itemname":["WHITE HANGING HEART T-LIGHT HOLDER","WHITE METAL LATERN","CREAM CUPIDS HEARTS COAT HANGER"...)]
    "Quantity":[6,6,8.....]
    "Date":[########,########,########....]
    "Price":[2.55,3.39,2.75....]
    "CustomerID":[17850,1850,17850....]
    "Country":[United Kingdom,United Kingdom,United Kingdom.....]
}
```

FEATURE ENGINEERING PROCESS

We engineer features to improve the performance of machine learning models by providing them with relevant and informative input data. Raw data may contain noise, irrelevant information, or missing values, which can lead to inaccurate or biased model predictions. By engineering features, we can extract meaningful information from the raw data, create new variables that capture important patterns and relationships, and transform the data into a more suitable format for machine learning algorithms.

Feature engineering can also help in addressing issues such as overfitting, underfitting, and high dimensionality. For example, by reducing the number of features, we can prevent the model from becoming too complex or overfitting to the training data. By selecting the most relevant features, we can improve the model's accuracy and interpretability.

In addition, feature engineering is a crucial step in preparing data for analysis and decision-making in various fields, such as finance, healthcare, marketing, and social sciences. It can help uncover hidden insights, identify trends and patterns, and support data-driven decision-making.

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder
```

```
data=pd.read_csv("Assignment-1_Data.csv")
```

```
numeric_features = ['bill no','date', 'customer id']
scaler = StandardScaler()
data[numeric_features] = scaler.fit_transform(data[numeric_features])
```

```
categorical_features = ['']
encoder = OneHotEncoder()
encoded_features = encoder.fit_transform(data[categorical_features]).toarray()
encoded_feature_names = encoder.get_feature_names(categorical_features)
data = pd.concat([data, pd.DataFrame(encoded_features, columns=encoded_feature_names)], axis=1)
data.drop(categorical_features, axis=1, inplace=True)
```

Visualization

Data visualization is the representation of data through use of common graphics, such as charts, plots, infographics, and even animations. These visual displays of information communicate complex data relationships and data-driven insights.

Types of Data Visualizations

Line charts and area charts: These visuals show change in one or more quantities by plotting a series of data points over time and are frequently used within predictive analytics. Line graphs utilize lines to demonstrate these changes while area charts connect data points with line segments, stacking variables on top of one another and using color to distinguish between variables.

Histograms: This graph plots a distribution of numbers using a bar chart (with no spaces between the bars), representing the quantity of data that falls within a particular range. This visual makes it easy for an end user to identify outliers within a given dataset.

Tables: This consists of rows and columns used to compare variables. Tables can show a great deal of information in a structured way, but they can also overwhelm users that are simply looking for high-level trends.

Pie charts and stacked bar charts: These graphs are divided into sections that represent parts of a whole. They provide a simple way to organize data and compare the size of each component to one other.

Scatter plots: These visuals are beneficial in reveling the relationship between two variables, and they are commonly used within regression data analysis. However, these can sometimes be confused with bubble charts, which are used to visualize three variables via the x-axis, the y-axis, and the size of the bubble.

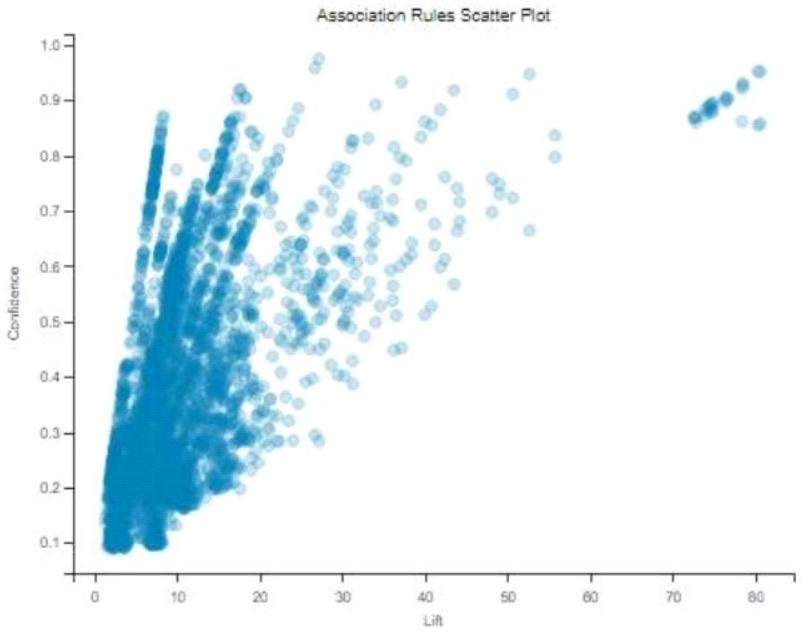
Heat maps: These graphical representation displays are helpful in visualizing behavioral data by location. This can be a location on a map, or even a webpage.

Tree maps, which display hierarchical data as a set of nested shapes, typically rectangles. Treemaps are great for comparing the proportions between categories via their area size

OPEN SOURCE TOOLS

Vega defines itself as “visualization grammar,” providing support to customize visualizations across large datasets which are accessible from the web.

```
import matplotlib.pyplot as plt
x=[0,10,20,30,40,50,60,70,80]
y=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0]
plt.scatter(x,y)
plt.xlabel("X AXIS LABEL")
plt.yaxis("Y AXIS LABEL")
plt.title("Scatter Plot")
plt.show()
```



EVALUATION

It is the performance of machine learning models is crucial for understanding how well they're doing .Here's is the basic example of how to evaluate a classification model using python and scikit-learn

Perspectives

The word "evaluation" has various connotations for different people, raising issues related to this process that include; what type of evaluation should be conducted; why there should be an evaluation process and how the evaluation is integrated into a program, for the purpose of gaining greater knowledge and awareness? There are also various factors inherent in the evaluation process, for example; to critically examine influences within a program that involve the gathering and analyzing of relative information about a program.

*Activities

*Characteristics

*Outcomes

*The making of judgments on a program

*Improving its effectiveness,

*Informed programming decisions

```python
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.ensemble import RandomForestClassifier
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
# Print the results
print(f'Accuracy: {accuracy:.2f}')
print('Confusion Matrix:')
print(conf_matrix)
print('Classification Report:')
print(class_report)
```