VISTORA ASSIGNMENT

RISHIKESH DWIVEDI
(BT22CSA027)

1. Introduction to Feature Engineering

Feature engineering is the process of selecting, modifying, or creating new features (columns) from raw data to help a machine learning model learn better. Think of it like preparing the ingredients before cooking — better prep usually leads to better results.

Why is it important?
Because even the best model won't perform well if the data it learns from isn't well-prepared. Good features make it easier for the model to understand patterns and make accurate predictions.

Types of Feature Engineering Techniques:

- - Normalization/Scaling
- - Encoding (e.g., One-hot, Label)
- - Time-Based Aggregations
- - Handling Missing Data
- - Binning/Bucketing

2. Using Snowflake for Data Storage & Processing

Snowflake is a cloud-based data platform used for storing and processing both structured and semi-structured data. It allows you to work with tables and also JSON-like data using the VARIANT type.

3. Feature Store Concepts

A Feature Store is a system that stores and manages machine learning features. It helps data scientists and ML engineers share and reuse features efficiently.

Why is it needed?

- - Keeps features consistent across training and inference
- - Avoids duplication and errors
- - Promotes collaboration by sharing features

Comparison of Feature Stores:

- - AWS SageMaker Feature Store: Fully managed, integrates with AWS
- - Snowflake Feature Store: SQL-based, good for analysts
- - Databricks Feature Store: Works well with Spark and MLflow

4. Implementing Feature Engineering with Snowflake & Feature Store

Step 1: Extract

```
import snowflake.connector
import pandas as pd


conn = snowflake.connector.connect(
    user='',
    password='',
    account='',
    warehouse='',
    database='',
    schema=''
)


cursor = conn.cursor()


query = "SELECT * FROM TITANIC LIMIT 1000"
cursor.execute(query)
df = pd.DataFrame(cursor.fetchall(), columns=[desc[0] for desc in cursor.description])
```

Step 2: Transform

```
from sklearn.preprocessing import LabelEncoder
df['SEX'] = LabelEncoder().fit_transform(df['SEX'])
df = pd.get_dummies(df, columns=['EMBARKED'], prefix='EMBARKED')
df['FAMILYSIZE'] = df['SIBSP'] + df['PARCH'] + 1
df['ISALONE'] = (df['FAMILYSIZE'] == 1).astype(int)
df['TITLE'] = df['NAME'].str.extract(' ([A-Za-z]+)\.', expand=False)
df['TITLE'] = df['TITLE'].replace(
    ['Lady', 'Countess', 'Capt', 'Col', 'Don', 'Dr', 'Major', 'Rev', 'Sir', 'Jonkheer', 'Dona'],
    'Rare'
)
df['TITLE'] = df['TITLE'].replace({'Mlle': 'Miss', 'Ms': 'Miss', 'Mme': 'Mrs'})
df['TITLE'] = LabelEncoder().fit_transform(df['TITLE'])
df = df.drop(columns=['PASSENGERID', 'NAME', 'TICKET', 'CABIN'])
```

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df[['AGE', 'FARE', 'FAMILYSIZE']] = scaler.fit_transform(df[['AGE', 'FARE', 'FAMILYSIZE']])
```

Step 3: Load into Feature Store

Once features are created, you can save them back into Snowflake as a table or push them into an external Feature Store like SageMaker or Databricks using their respective APIs.

```
[70] from snowflake.connector.pandas_tools import write_pandas

     success, nchunks, nrows, _ = write_pandas(
         conn=conn,
         df=df_to_upload,
         table_name='TITANIC_FEATURES_FINAL',  # use an existing or new table name
         auto_create_table=True
     )
```

Step 4: ML Model

```
[79] from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LogisticRegression
     from sklearn.metrics import accuracy_score


     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


     model = LogisticRegression(max_iter=1000)
     model.fit(X_train, y_train)

     y_pred = model.predict(X_test)


     accuracy = accuracy_score(y_test, y_pred)
     print(f"Model accuracy: {accuracy * 100:.2f}%")
```

```
Model accuracy: 80.34%
```