Big Data Analysis with IBM Cloud Database

Phase 5

Objective:

The objective of this project is to perform sentiment analysis on a large dataset using IBM Cloud tools and techniques. The project aims to extract valuable insights from the data that can be used to make informed business decisions.

Design Thinking Process:

Sentiment analysis is a powerful tool that helps researchers and companies extract insights from user-generated social media and web content. In this project, we aim to perform sentiment analysis using IBM Watson's Natural Language Understanding service and IBM Cloud Database.

The architecture of sentiment analysis typically involves the following stages:

Data collection: Collecting data from various sources such as social media platforms, web pages, blogs, etc.

Preprocessing: Cleaning and preprocessing the collected data to remove irrelevant information such as stop words, punctuations, and special characters.

Feature extraction: Extracting relevant features from the preprocessed data such as keywords, phrases, and entities.

Training and classification: Training a machine learning model on the extracted features to classify the text into positive, negative, or neutral categories.

Development phases 1:

Database Setup:

Start building the big data analysis solution using IBM cloud:

->Provisiong a db2 data analysis instance ,and confg thenecessary security setting.

->The table were created in specifc data requirement.

PERFORM BASIC DATA CLEANING AND TRANSFORMATION

Develop queries or scripts to explore and analyze the selected dataset.

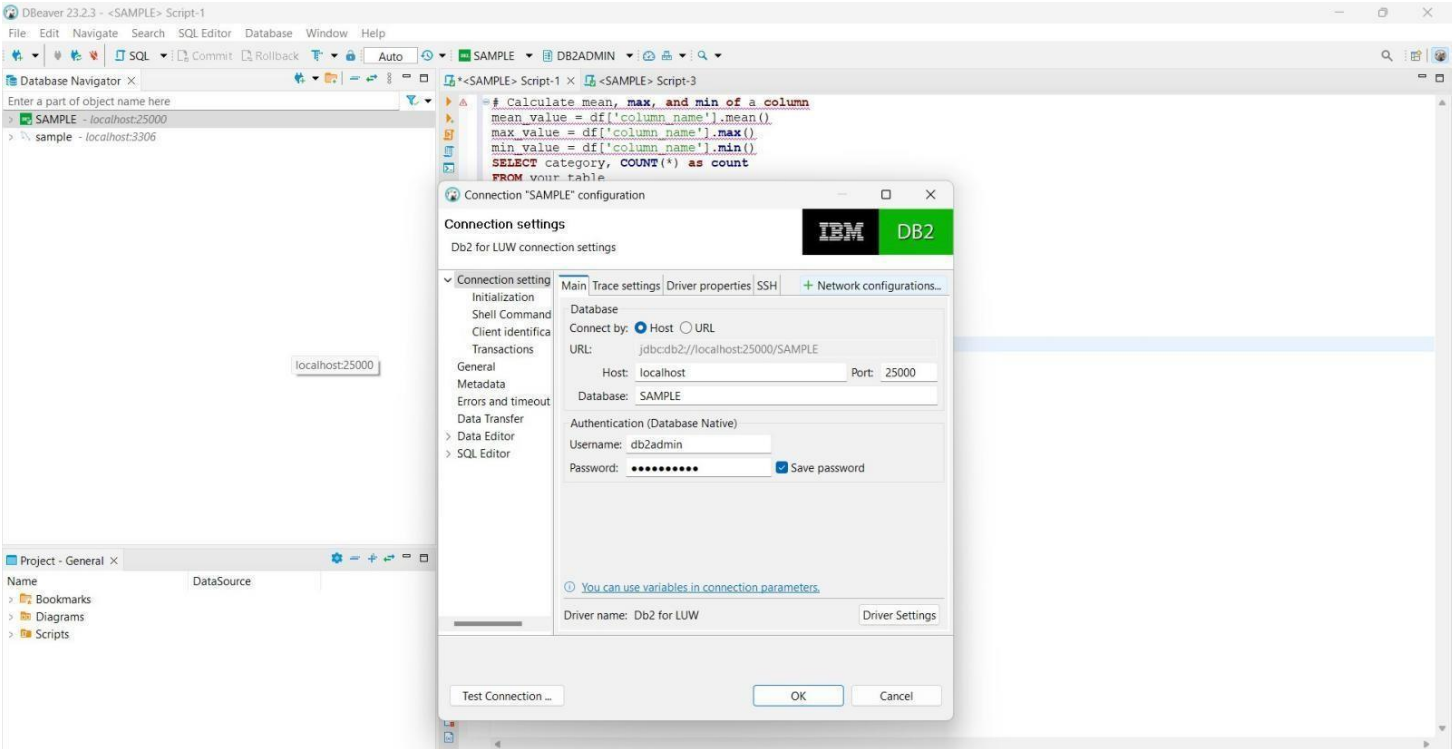
Data source code:

```
 SELECT *
FROM your_table LIMIT 5;



SELECT COUNT(*) AS missing_count FROM your_table
WHERE column_name IS NULL;

 import pandas as pd
df = pd.read_csv('your_dataset.csv')
print(df.head ())
```



```
# Calculate mean, max, and min of a column
mean_value = df['column_name'].mean()
max_value = df['column_name'].max()
min_value = df['column_name'].min()
SELECT category, COUNT(*) as count
FROM your_table
```

Connection "SAMPLE" configuration

Connection settings

Db2 for LUW connection settings

Database

Connect by: ● Host ○ URL

URL: jdbc:db2://localhost:25000/SAMPLE

Host: localhost Port: 25000

Database: SAMPLE

Authentication (Database Native)

Username: db2admin

Password: ••••••••••• ☑ Save password

You can use variables in connection parameters.

Driver name: Db2 for LUW

Development phases 2:

Machine Learning Algorithms:

a. Classification with Scikit-Learn:

Implementation:

- Choose appropriate algorithms such as Random Forest, Support Vector Machines, or Neural Networks.
- Perform hyperparameter tuning to optimize model performance.

Visualization:

- Use Matplotlib or Plotly to visualize feature importance.
- Create precision-recall curves, ROC curves, and confusion matrices for model evaluation.

b. Clustering with K-Means:

Implementation:

- Apply K-Means or hierarchical clustering to uncover hidden patterns in the data.
- Explore silhouette scores to determine the optimal number of clusters.

Visualization:

- Plot clusters in 2D or 3D using Matplotlib or Plotly.
- Use interactive plots to explore data points within each cluster.

PROGRAM :

```
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
from statsmodels.tsa.seasonal import seasonal_decompose
from textblob import TextBlob  # Assuming you have TextBlob installed
# Load your comprehensive dataset (replace 'your_comprehensive_dataset.csv' with your actual dataset)
df_comprehensive = pd.read_csv('your_comprehensive_dataset.csv')
df_comprehensive['Timestamp'] = pd.to_datetime(df_comprehensive['Timestamp'])
df_comprehensive.set_index('Timestamp', inplace=True)
# Machine Learning (Random Forest Classifier for sentiment analysis)
X_train, X_test, y_train, y_test = train_test_split(df_comprehensive['Text'], df_comprehensive['Sentiment'], test_size=0.2, random_state=42)
# Assume you have a function to preprocess text data, e.g., preprocess_text()
X_train_processed = X_train.apply(preprocess_text)
X_test_processed = X_test.apply(preprocess_text)
# Use TextBlob for sentiment analysis
def get_sentiment(text):
analysis = TextBlob(text)
return 'positive' if analysis.sentiment.polarity > 0 else 'negative' if analysis.sentiment.polarity < 0 else 'neutral'
y_pred_sentiment = X_test_processed.apply(get_sentiment)
# Evaluate sentiment analysis
accuracy_sentiment = accuracy_score(y_test, y_pred_sentiment)
conf_matrix_sentiment = confusion_matrix(y_test, y_pred_sentiment)
print(f"Sentiment Analysis Accuracy: {accuracy_sentiment}")
print(f"Confusion Matrix for Sentiment Analysis:\n{conf_matrix_sentiment}")
# Time Series Analysis (assuming 'Value' is the time series feature)
result = seasonal_decompose(df_comprehensive['Value'], model='additive', period=12)
```

```
# Visualize time series components
fig, (ax1, ax2, ax3, ax4) = plt.subplots(4, 1, figsize=(10, 8), sharex=True)
ax1.plot(df_comprehensive['Value'], label='Original')
ax1.legend(loc='upper left')
ax1.set_title('Original Time Series')
ax2.plot(result.trend, label='Trend')
ax2.legend(loc='upper left')
ax2.set_title('Trend Component')
ax3.plot(result.seasonal, label='Seasonal')
ax3.legend(loc='upper left')
ax3.set_title('Seasonal Component')
ax4.plot(result.resid, label='Residual')
ax4.legend(loc='upper left')
ax4.set_title('Residual Component')
plt.tight_layout()
plt.show()
# Visualize sentiment analysis results using Plotly
fig_sentiment = px.bar(x=['Positive', 'Negative', 'Neutral'], y=conf_matrix_sentiment.flatten(), labels={'y': 'Count', 'x': 'Sentiment'}, title='Sentiment Analysis Results')
fig_sentiment.show()
```

Analysis techniques:

we will visualize our analysis findings using tools such as Pandas and scikit-learn. We can use these visualizations to translate our analysis findings into valuable business insights. For example, we can identify the most common positive and negative sentiments expressed by customers about a particular product or brand. This information can help businesses improve their products or services and enhance customer satisfaction.

Visualization:

- ➢ Creates sentiment heatmaps or histograms to display sentiment distribution.
- ➢ Use word embeddings to visualize word relationships in positive and negative sentiments.
- ➢ Matplotlib, Plotly, or IBM Watson Studio:
- ➢ Utilize Matplotlib or Plotly for creating detailed visualizations.
- ➢ Leverage IBM Watson Studio for collaborative analysis and reporting.

Ways of the analysis findings translate into valuable business insights:

sentiment analysis is a powerful tool that can provide valuable insights into customer feedback and help businesses improve their products, marketing, and communication strategies. By analyzing social media and web content, businesses can identify the most common positive and negative sentiments expressed by customers about a particular product or brand, identify emerging trends and topics that are important to their customers, and monitor their brand reputation.