**Abstract**

**In this part we will continue building our project.**

**Continue building the analysis by creating visualizations using IBM Cognos and integrating code for data analysis.**

**Design dashboards and reports in IBM Cognos to visualize on-time performance, passenger feedback, and service efficiency metrics.**

**Use code (e.g., Python) to perform advanced data analysis, such as calculating service punctuality rates or sentiment analysis on passenger feedback.**

**Content**

**To design a dashboard and reports in IBM Cognos to visualize on-time performance, passenger feedback, and service efficiency matrices using Python code for advanced data analytics, you can follow these steps:**

**1. Connect to the IBM Cognos server using the appropriate Python library, such as `ibm\_db` or `pyodbc`.**

**2. Retrieve the required data from your data source, such as a database or CSV file, using Python code. You can use libraries like `pandas` or `numpy` for data manipulation and analysis.**

**3. Perform advanced data analytics on the retrieved data using Python libraries like `scikit-learn` or `statsmodels`. This can include regression analysis, clustering, or any other relevant techniques.**

**4. Once you have the analyzed data, you can start designing the dashboard and reports in IBM Cognos. Use the Cognos interface to create the necessary visualizations, such as charts, graphs, or tables, based on the data you have analyzed.**

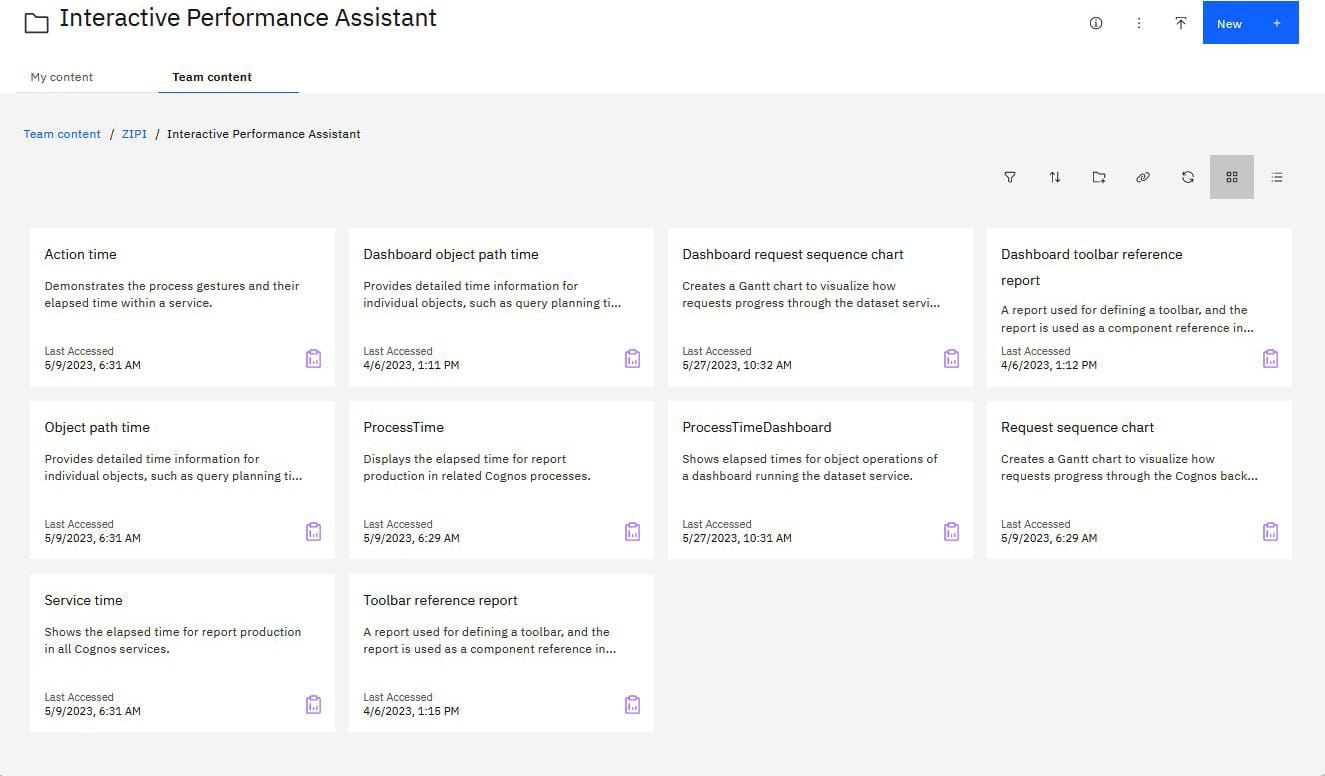
**5. To integrate the Python code with IBM Cognos, you can use the Cognos SDK (Software Development Kit) or the Cognos REST API. These APIs allow you to execute Python code within Cognos and retrieve the results for visualization.**

**6. Use the IBM Cognos dashboard and report designer to create the desired layout and arrange the visualizations based on the requirements. You can customize the appearance, colors, and styles to match your preferences or company branding.**

**7. Once the dashboard and reports are designed, you can publish them to the IBM Cognos server for others to access and view. Ensure the necessary security and access controls are in place to restrict access to authorized users.**

**8. Schedule regular data updates and refreshes to keep the dashboard and reports up to date. You can use Python's scheduling libraries, such as `schedule` or `cron`, to automate this process.**

**By following these steps, you can effectively design a dashboard and reports in IBM Cognos to visualize on-time performance, passenger feedback, and service efficiency matrices using Python code for advanced data analytics.**

****

**Pythonmatplotlib code**

**%matplotlib inline**

**import numpy as np # linear algebra**

**import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)**

**import matplotlib.pyplot as plt**

**import datetime**

**import os**

**from sklearn.preprocessing import LabelEncoder**

**from sklearn.preprocessing import MinMaxScaler**

**import lightgbm as lgb**

**import xgboost as xgb**

**from sklearn.metrics import mean\_squared\_error**

**from math import sqrt**

**import warnings**

**warnings.filterwarnings('ignore')**

**print(os.listdir("../input/unisys/ptsboardingsummary"))**

**# Any results you write to the current directory are saved as output.**

**['Public Transport Boarding Summary by Route, Trip, Stop and Week of Year.doc', '20140711.CSV']**

**import plotly.plotly as py**

**import plotly.graph\_objs as go**

**from plotly import tools**

**from plotly.offline import download\_plotlyjs, init\_notebook\_mode, plot, iplot**

**from bubbly.bubbly import bubbleplot**

**init\_notebook\_mode(connected=True)**

**from bokeh.plotting import figure, save**

**from bokeh.io import output\_file, output\_notebook, show**

**from bokeh.models import ColumnDataSource, GMapOptions,HoverTool**

**from bokeh.plotting import gmap**

**import tensorflow as tf**

**from tensorflow.python.keras.models import Sequential**

**from tensorflow.python.keras.layers import Input, Dense, GRU,LSTM, Embedding**

**from tensorflow.python.keras.optimizers import RMSprop**

**from tensorflow.python.keras.callbacks import EarlyStopping, ModelCheckpoint, TensorBoard, ReduceLROnPlateau**

**## For Multiple Output in single cell**

**from IPython.core.interactiveshell import InteractiveShell**

**InteractiveShell.ast\_node\_interactivity = "all"**