WEBSITE TRAFFIC ANALYSIS

DEVELOPMENT WITH PYTHON LIBRARIES

AGENDA

- Introduction
- Python
- Its libraries
- Data visualization
- Program code
- Conclusion

introduction

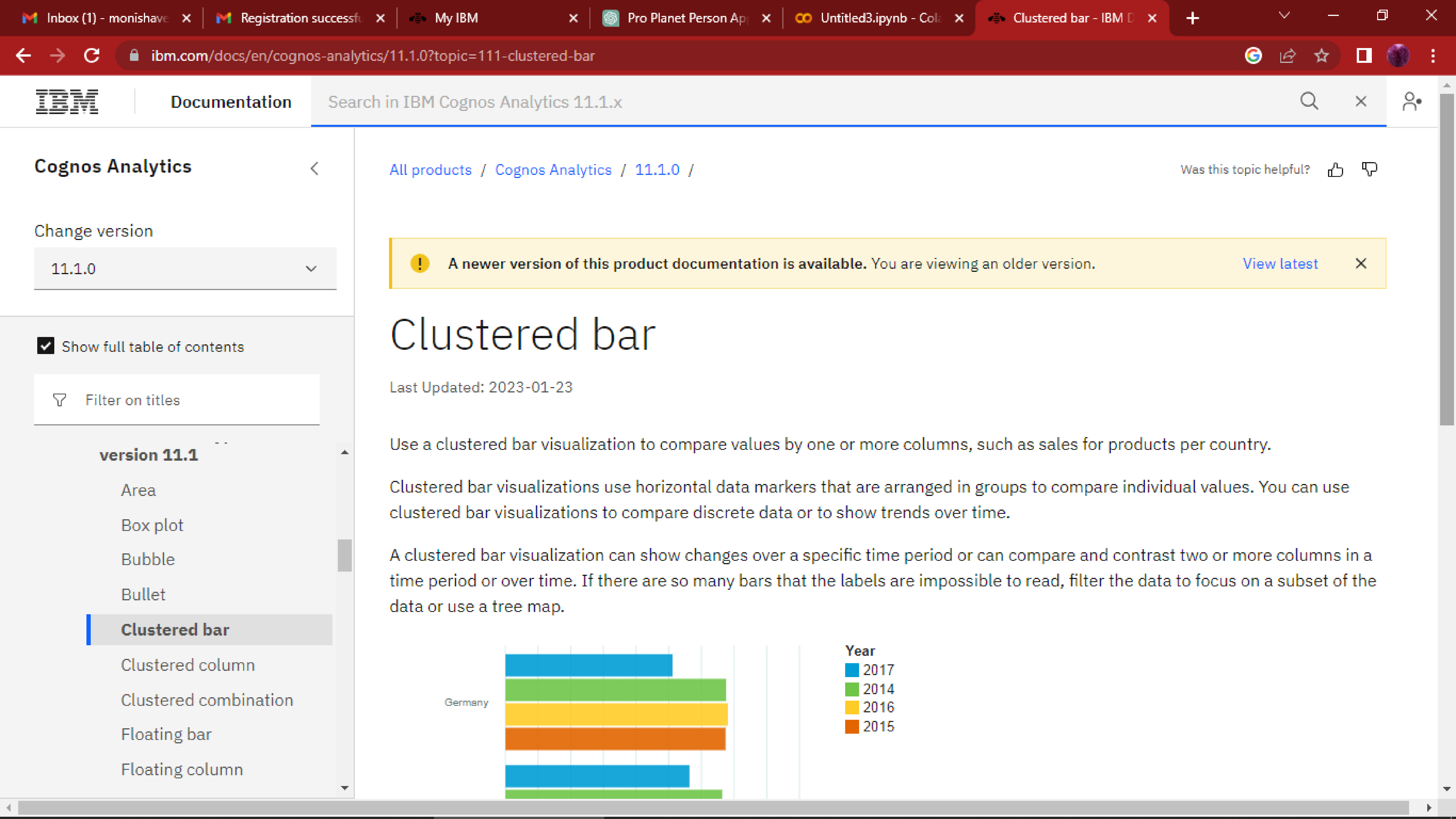
- Website traffic analysis is the process of evaluating and understanding the interactions and behaviors of visitors to a website. It provides valuable insights into how users engage with a website, what content they find most appealing, and how they navigate through the site. This information is essential for website owners and digital marketers, as it helps optimize the website, improve user experience, and achieve specific goals such as increasing conversion rates, driving more traffic, or boosting online revenue.

Python

- Python is a high-level, versatile, and widely-used programming language known for its readability and ease of use. It was created by Guido van Rossum and first released in 1991. Python is an interpreted language, which means that you can write and execute code directly without the need for compilation.

LIBRARIES

- NumPy, pandas, Matplotlib, Seaborn, Scikit-Learn, TensorFlow, PyTorch, Keras, NLTK, Requests, Django, Flask, SQLAlchemy, Beautiful Soup, OpenCV, SciPy, NetworkX, Pillow, Statsmodels, SymPy.



A newer version of this product documentation is available. You are viewing an older version. View latest

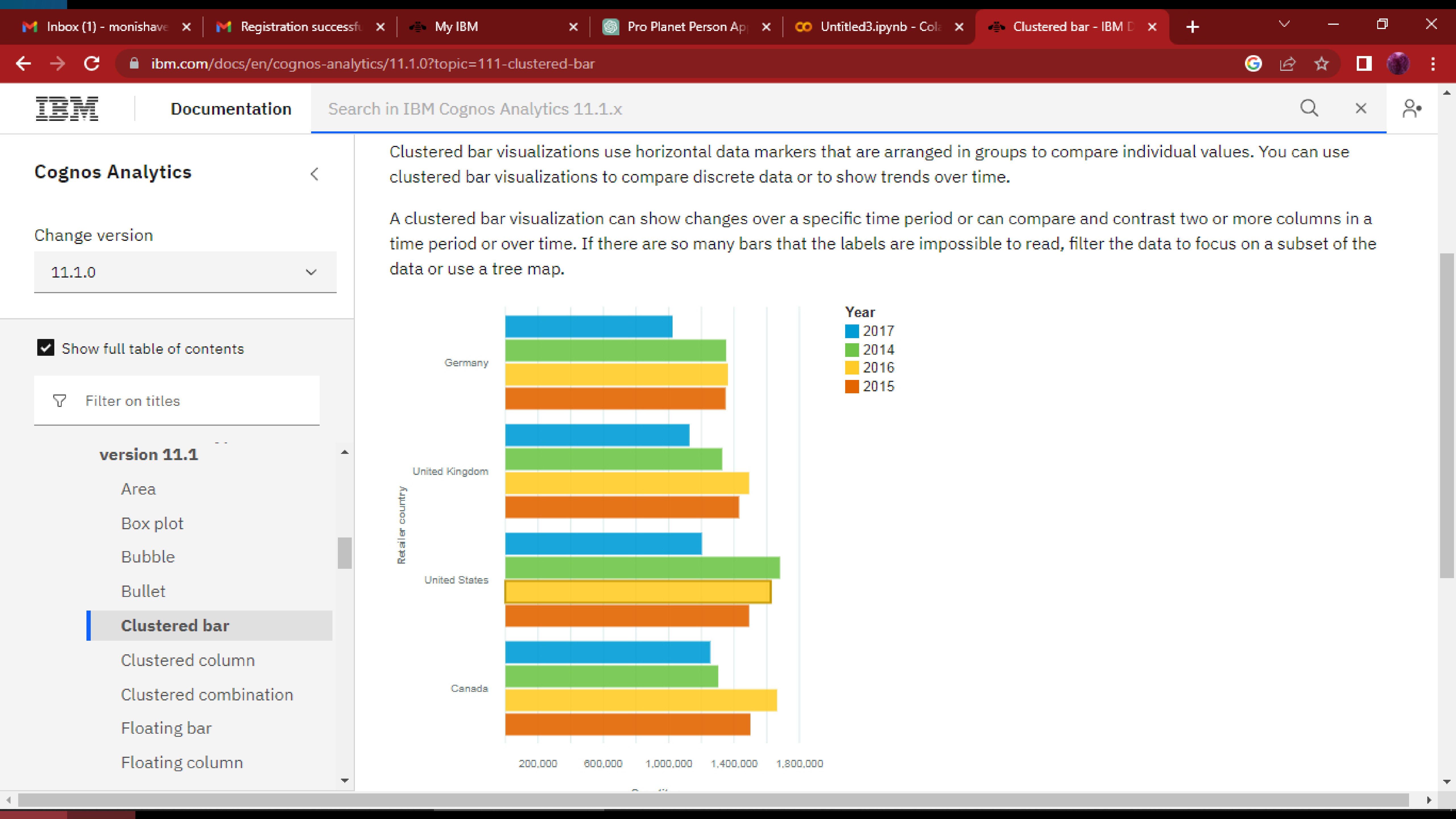
Clustered bar

Last Updated: 2023-01-23

Use a clustered bar visualization to compare values by one or more columns, such as sales for products per country.

Clustered bar visualizations use horizontal data markers that are arranged in groups to compare individual values. You can use clustered bar visualizations to compare discrete data or to show trends over time.

A clustered bar visualization can show changes over a specific time period or can compare and contrast two or more columns in a time period or over time. If there are so many bars that the labels are impossible to read, filter the data to focus on a subset of the data or use a tree map.



Cognos Analytics

Change version

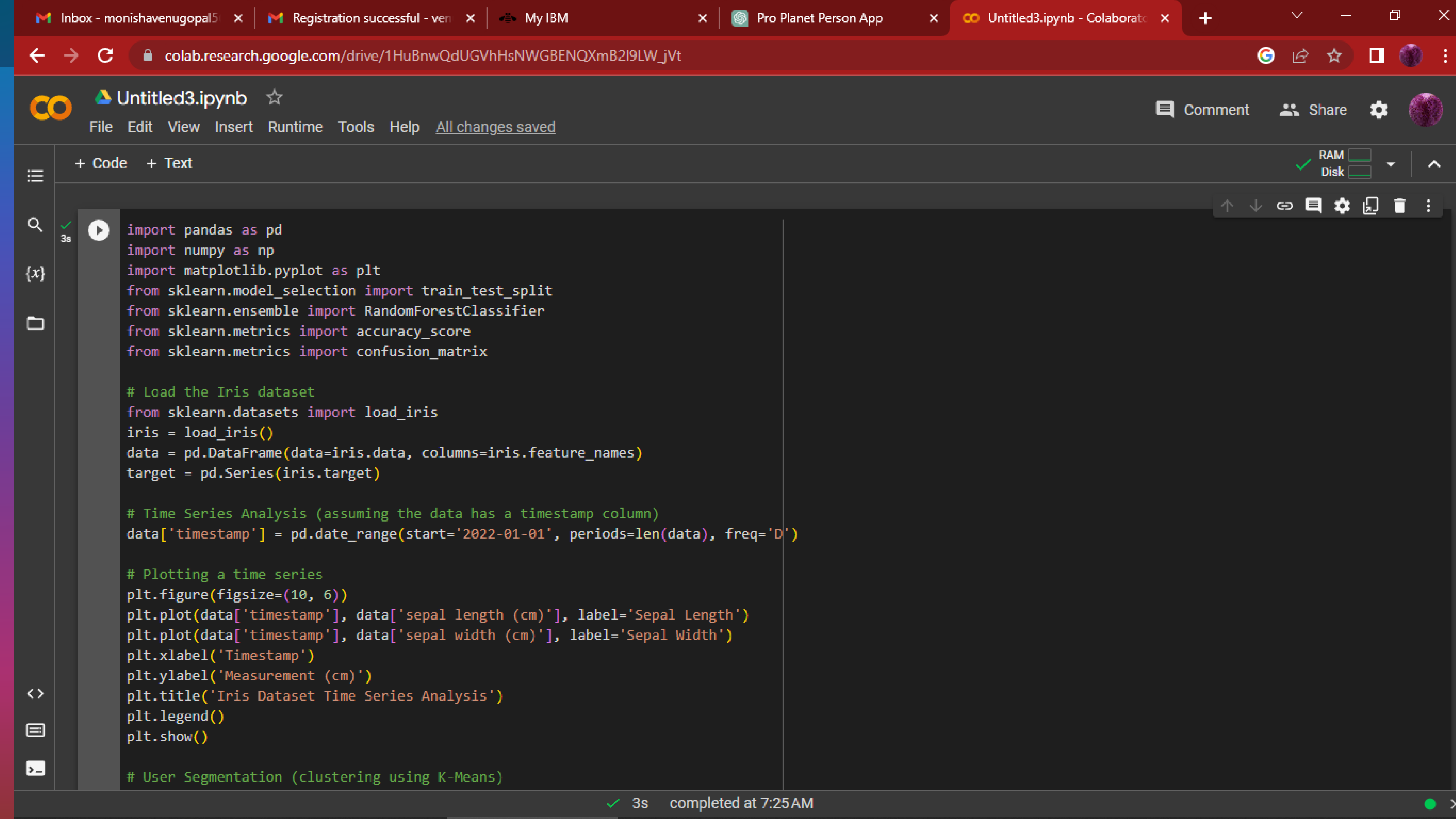
11.1.0

☑ Show full table of contents

Filter on titles

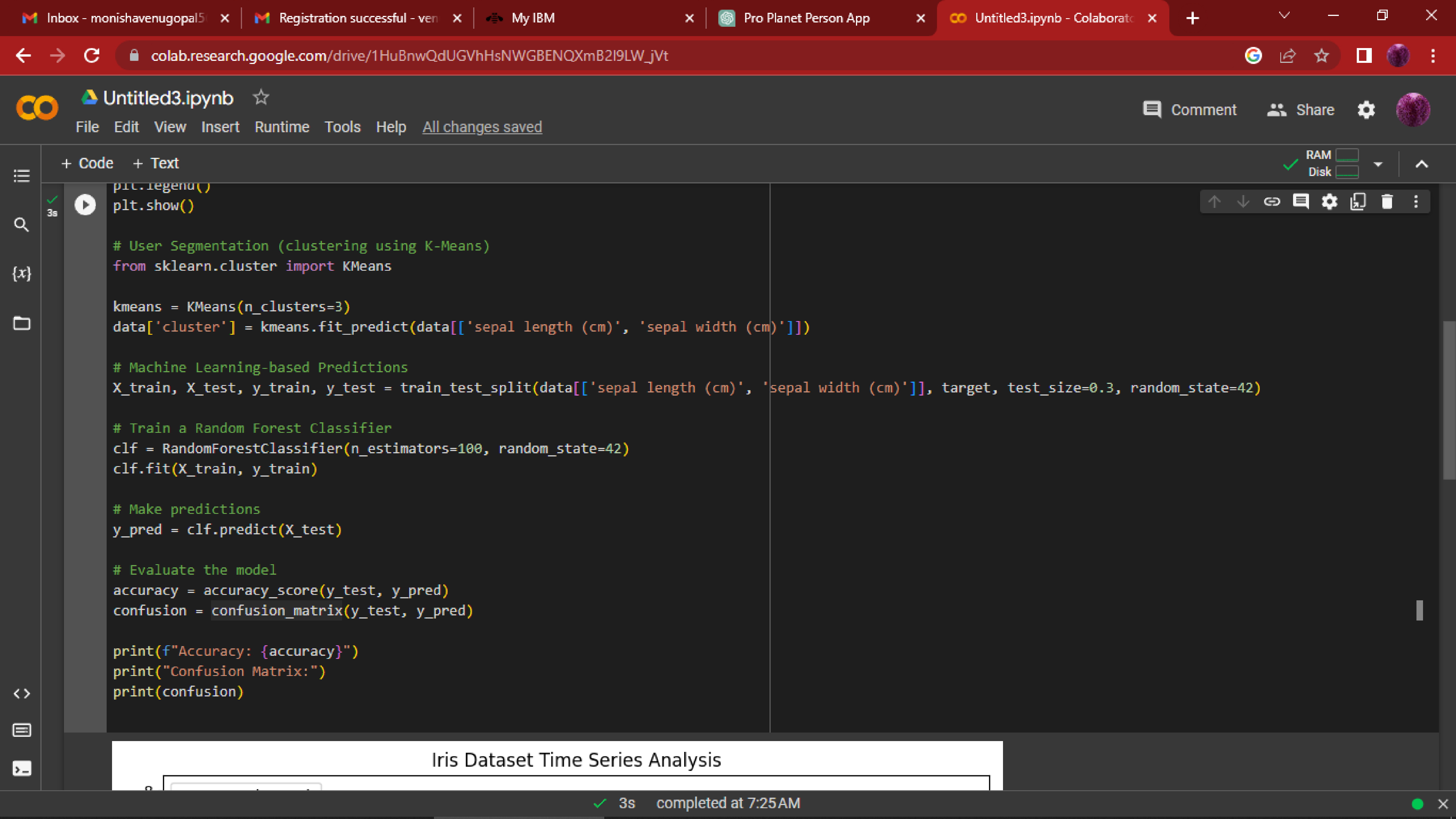
Clustered bar visualizations use horizontal data markers that are arranged in groups to compare individual values. You can use clustered bar visualizations to compare discrete data or to show trends over time.

A clustered bar visualization can show changes over a specific time period or can compare and contrast two or more columns in a time period or over time. If there are so many bars that the labels are impossible to read, filter the data to focus on a subset of the data or use a tree map.

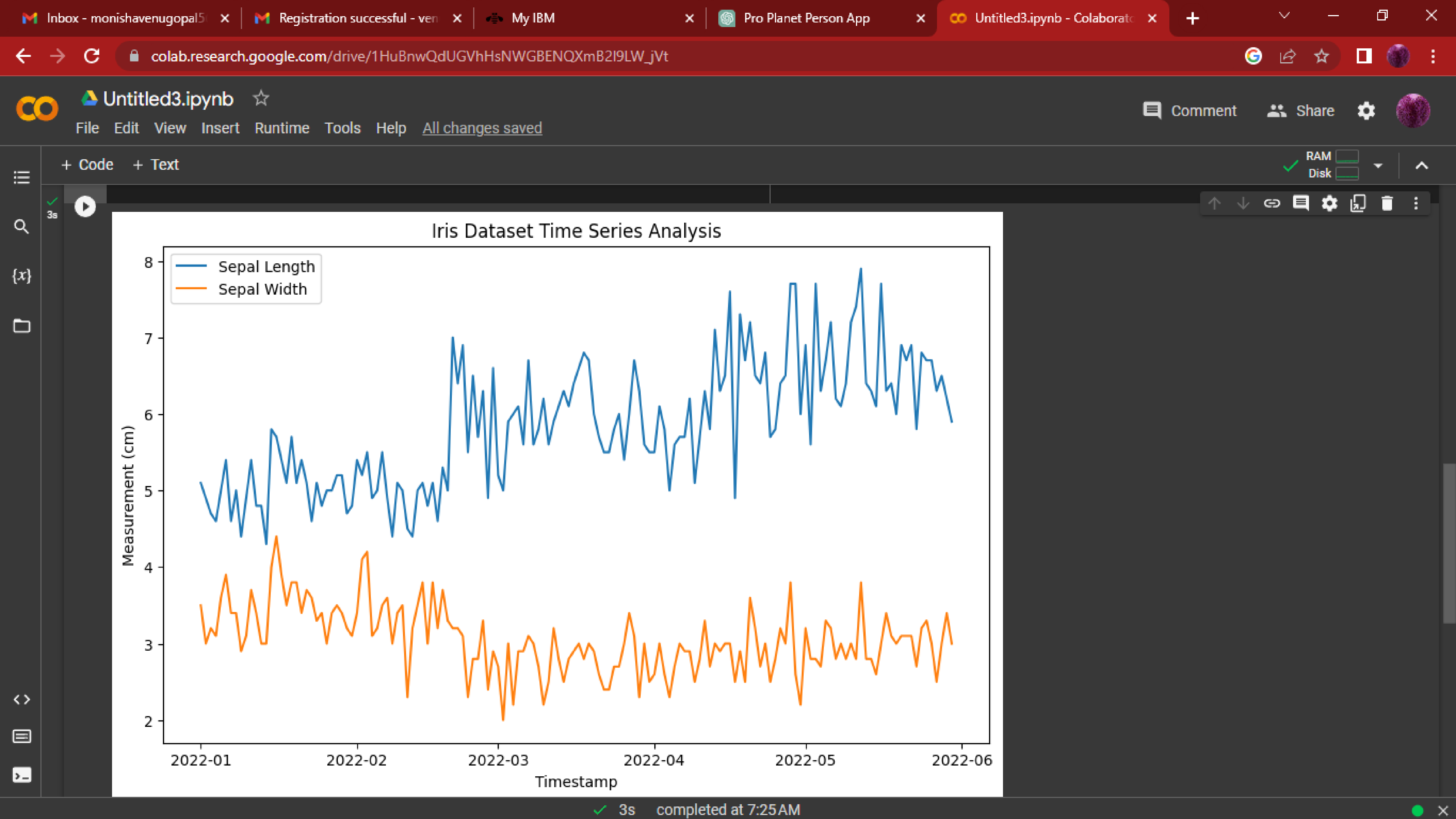

Inbox - monishavenugopal5
Registration successful - ven
My IBM
Pro Planet Person App
Untitled3.ipynb - Colaborat
colab.research.google.com/drive/1HuBnwQdUGVhHsNWGBENQXmB2I9LW_jVt
Untitled3.ipynb
File
Edit
View
Insert
Runtime
Tools
Help
All changes saved
Comment
Share
+ Code
+ Text
RAM
Disk
3s

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix

# Load the Iris dataset
from sklearn.datasets import load_iris
iris = load_iris()
data = pd.DataFrame(data=iris.data, columns=iris.feature_names)
target = pd.Series(iris.target)

# Time Series Analysis (assuming the data has a timestamp column)
data['timestamp'] = pd.date_range(start='2022-01-01', periods=len(data), freq='D')

# Plotting a time series
plt.figure(figsize=(10, 6))
plt.plot(data['timestamp'], data['sepal length (cm)'], label='Sepal Length')
plt.plot(data['timestamp'], data['sepal width (cm)'], label='Sepal Width')
plt.xlabel('Timestamp')
plt.ylabel('Measurement (cm)')
plt.title('Iris Dataset Time Series Analysis')
plt.legend()
plt.show()

# User Segmentation (clustering using K-Means)
```

✓ 3s completed at 7:25 AM


Inbox - monishavenugopal5
Registration successful - ven
My IBM
Pro Planet Person App
Untitled3.ipynb - Colaborato
colab.research.google.com/drive/1HuBnwQdUGVhHsNWGBENQXmB2I9LW_jVt
Untitled3.ipynb
File
Edit
View
Insert
Runtime
Tools
Help
All changes saved
Comment
Share
+ Code
+ Text
RAM
Disk
3s
Iris Dataset Time Series Analysis
3s
completed at 7:25 AM

```
plt.legend()
plt.show()

# User Segmentation (clustering using K-Means)
from sklearn.cluster import KMeans

kmeans = KMeans(n_clusters=3)
data['cluster'] = kmeans.fit_predict(data[['sepal length (cm)', 'sepal width (cm)']])

# Machine Learning-based Predictions
X_train, X_test, y_train, y_test = train_test_split(data[['sepal length (cm)', 'sepal width (cm)']], target, test_size=0.3, random_state=42)

# Train a Random Forest Classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

# Make predictions
y_pred = clf.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
confusion = confusion_matrix(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print("Confusion Matrix:")
print(confusion)
```



✓ 3s completed at 7:25 AM

CONCLUSION

- In conclusion, Python libraries like Pandas and Matplotlib offer powerful tools for conducting in-depth analyses of data, allowing us to uncover valuable insights and make informed decisions. Through time series analysis, we can understand trends and patterns in our data over time, enabling us to anticipate future developments. User segmentation helps us tailor our strategies to different user groups, optimizing the user experience and achieving our business objectives