

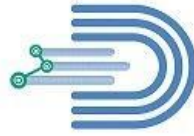


DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

Customer Churn

Use Case Implementation - GCP



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

Table of Contents

1. Customer Churn Use Case Implementation	3
1.1. Disclaimer	3
1.2. DeepSphere.AI and Google Cloud	3
1.3. Executive Summary	4
1.4. Problem Statement	4
1.5. Business Challenges	5
1.6. Model Selection	5
1.7. Feature Engineering	6
1.7.1. Advantages of Feature Engineering	7
1.8. Data Management	7
1.8.1. What is a Training Data Set	7
1.8.2. What is a Test Data Set	8
1.9. Learning Algorithm	10
1.9.1. Machine Learning Libraries Used	10
1.9.2. Classification Models Used	10
1.10. Model Building Blocks	10
1.11. Model Implementation High-level Steps	11
1.12. Model Building Steps	12
1.12.1. Create a Google Storage Bucket	12
1.12.2. Moving Files into Google Storage Bucket	18
1.12.3. Create Service account and Private Key	19
1.12.4. Access data from Google Storage Bucket using Python	25
1.12.5. Model Building Code Block	36

A Machine Learning Laboratory Manual

1. Customer Churn Use Case Implementation

1.1. Disclaimer

- We share this information for learning purposes only. We developed this material based on our prior experience, skills, knowledge, and expertise. Our perspective on the tools, technologies, systems, applications, processes, methodologies, and others used in these materials may differ from others. We advise the users to use these materials at their own risk.
- The sample programs used in this material are developed by us based on system and data assumptions. These examples may or may not work for others. If there are any issues in following this material, please feel free to contact our support services. We will help you based on our support resource availability.
- The respective vendors own all the hardware, software, tools, technologies, processes, methodologies, and others used in this material. Users may use these learning resources at their own risk. Under any circumstance, DeepSphere.AI is not liable for any of these vendor's products, services, and resources.

1.2. DeepSphere.AI and Google Cloud

- DeepSphere.AI (DS.AI) is a global leader in providing an advanced and higher educational platform for schools. DS.AI provides an intelligent learning management system (iLMS) to learn applied artificial intelligence, data science, and data engineering at a personalized level. DS.AI iLMS platform hosted on Amazon web services (AWS) and the learning resources developed on Google Cloud Platform(GCP) and SAP Litmos.

A Machine Learning Laboratory Manual

- To create social readiness and awareness about applied AI, DS.AI continues to develop learning resources to educate and empower schools, colleges, universities, organizations, and public entities. This article is part of a series of learning resources. There will be several articles in the future which will be published to master applied AI on Google Cloud. We use several GCP services to develop these learning resources, including storage services, compute services, network services; and other products and services.
- Our goal is to go beyond concepts, ideas, visions, and strategies to provide practical problem-solving applied AI skills, knowledge, and expertise that will result in on-the-job learning experience. To achieve our goals and objectives, we use GCP products and services, including BigQuery, AutoML, AutoML Tables, Dataproc, Dataflow, Data Studio, etc.

1.3. Executive Summary

- The purpose of this document is to provide adequate information to users to implement Customer Churn in Google Cloud Platform. In order to achieve this, we are using supervised machine learning models like Logistic Regression or XGBoost.

1.4. Problem Statement

- Companies or Organizations often face huge customer attrition or churn. When customers leave the company, they not only lose the revenue but also lose the resources spent to acquire these customers in the first place. This is a serious concern for the companies.
- In this Implementation we are trying to predict customers who are mostly likely to churn using machine learning modelling. This implementation helps the companies to know in

A Machine Learning Laboratory Manual

advance the customers who are more likely to leave the business at some point in time.

With this prediction, the companies can come up with retention strategies and policies.

1.5. Business Challenges

Companies need to build and deploy effective customer churn prediction models to succeed in today's complex business scenarios. Acquiring new customers always costs heavily. Following are the challenges companies face when there is no customer prediction modelling in place

- No Sustainable and robust strategy for customer retention.
- No formula plan to reacquire the customers who have moved to other competitors.
- Issues in converting low revenue earning customers into highly profitable ones.
- Reducing customer defections and improving profits.
- Tracking customer satisfaction by product, segment and cost to serve.

All these business challenges are solved by the predictive churn models that aim at retaining customers and maximizing profits.

1.6. Model Selection

Model selection is the process of choosing between different machine learning approaches, e:g Decision Tree, Logistic Regression, etc, or choosing between different hyperparameters or sets of features for the same machine learning approach, e:g deciding between the polynomial degrees/complexities for linear regression.

A Machine Learning Laboratory Manual

The choice of the actual machine learning algorithm (e.g. SVM or logistic regression) is less important than one would think. There could be a "best" algorithm for any given problem, but often its performance is hardly better than other well-performing approaches for the same problem.

There may be certain qualities you might look for in a model:

- Interpretable - can we see or understand why the model is making the decisions it makes?
- Simple - easy to explain and understand
- Accurate
- Fast (to train and test)
- Scalable (it can be applied to a large dataset)

Our Problem here is a Supervised Classification Problem. The Problem is to predict customers who are more likely to churn. This type of problem can only be solved by the following models.

1. Logistic Regression.
2. XGBoost.

1.7. Feature Engineering

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. If feature engineering is done correctly, it increases the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning process. Feature Engineering is an art.

Feature engineering is the most important art in machine learning which creates a huge difference between a good model and a bad model.

A Machine Learning Laboratory Manual

1.7.1. Advantages of Feature Engineering

- Good features provide you with the flexibility of choosing an algorithm; even if you choose a less complex model, you get good accuracy.
- If you choose good features, then even simple ML algorithms do well.
- Better features will lead you to better accuracy. You should spend more time on features engineering to generate the appropriate features for your dataset. If you derive the best and appropriate features, you have won most of the battle.

1.8. Data Management

- There are three types of data sets: Training, Test and Dev that are used at various stages of implementation. Training dataset is the largest of the three, while test data functions as a seal of approval and you don't need to use it till the end of the development.

1.8.1. What is a Training Data Set

- The training data set is the actual dataset used to train the model for performing various Machine Learning Operations (Regression, Classification, Clustering etc.). This is the actual data with which the models learn with various API and algorithms to train the machine to work automatically.

A Machine Learning Laboratory Manual

CustomerID	SUM(CustomerBuyingPattern.Average yearly purchase)	SUM(CustomerBuyingPattern.Last year purchase)	SUM(CustomerBuyingPattern.Quantity(in lots))	SUM(CustomerBuyingPattern.average Monthly wise purchase)
18928	317	32	71	30
18811	481	2	32	25
19651	437	34	55	39
18649	499	22	92	36
18056	329	3	90	32
...
19034	0	0	0	0
19732	0	0	0	0
18764	386	18	38	28
18836	1489	94	151	132
19654	0	0	0	0

Figure 1 - Training Data

The following section describes the data training data sets and its field level characteristics

- Customer
- Average Yearly Purchase
- Last Year Purchase
- Quantity
- Customer Lifetime in Years
- Price Amount
- Last Year Unit Price
- Product Average Unit Price
- Amount Spent in Lifetime
- Service Call
- Service Failure Rate

1.8.2. What is a Test Data Set

A Machine Learning Laboratory Manual

- Test data set helps you to validate that the training has happened efficiently in terms of either accuracy, or precision and so on. Such data is used in testing the models to analyze whether the model is responding and working appropriately.

CustomerID	SUM(CustomerBuyingPattern.Average yearly purchase)	SUM(CustomerBuyingPattern.Last year purchase)	SUM(CustomerBuyingPattern.Quantity(in lots))	SUM(CustomerBuyingPattern.average Monthly wise purchase)
19092	0	0	0	0
19787	1215	51	149	91
19440	1022	45	233	94
18746	1151	84	236	110
18821	318	7	65	32
...
19171	373	0	62	36
18939	462	11	64	39
18949	972	55	96	70
19653	863	33	114	79
19342	467	2	38	35

Figure 2 - Test Data

The following section describes the features that's used in the model.

- Customer
- Average Yearly Purchase
- Last Year Purchase
- Quantity
- Customer Lifetime in Years
- Price Amount
- Last Year Unit Price
- Product Average Unit Price
- Amount Spent in Lifetime
- Service Call
- Service Failure Rate

A Machine Learning Laboratory Manual

1.9. Learning Algorithm

- A Self Learning (not a human developed code) code, performs data analysis and extracts patterns (business characteristics) in data for business application development - A Modern approach to application/software development.
- Automatically understands and extracts data patterns when data changes (change in business circumstance) and performs data analysis based on the new/changed data set. No code change required to implement changes that took place in the data (change in business)

1.9.1. Machine Learning Libraries Used

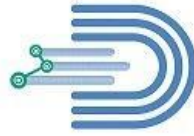
- **Sklearn (Scikit Learn)**
- **Pandas**

1.9.2. Classification Models Used

- **Logistic Regression**
- **XGBoost**

1.10. Model Building Blocks

- There are several technical and functional components involved in implementing this model. Here are the key building blocks to implement this model.



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

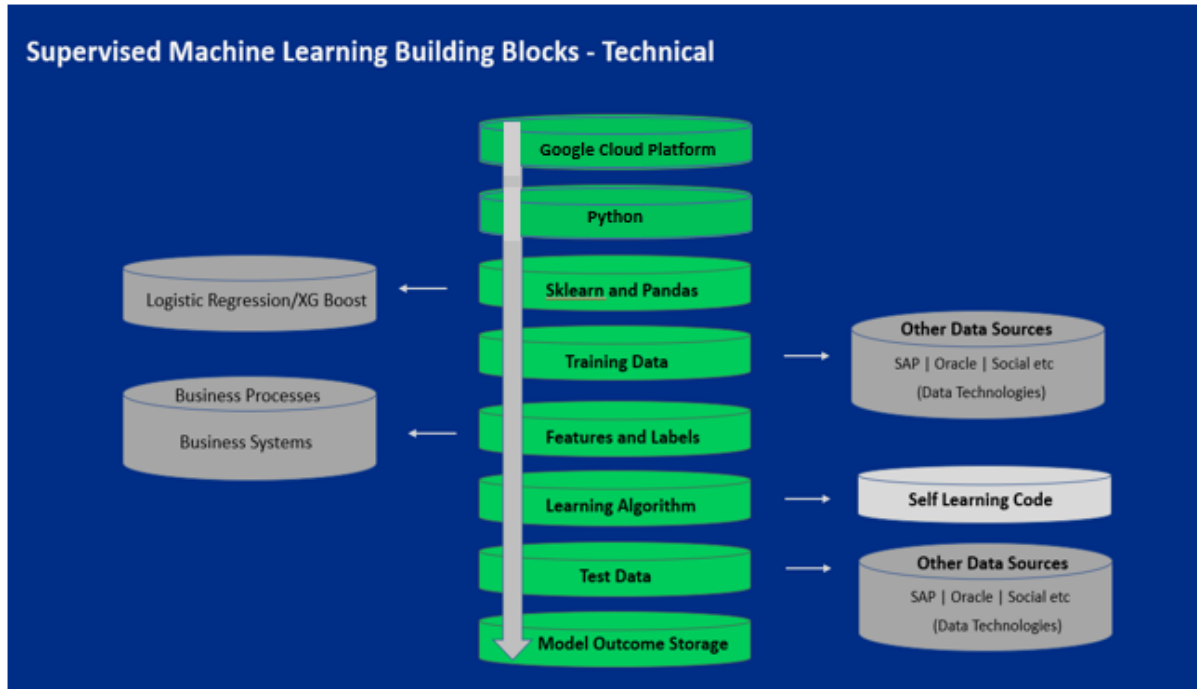
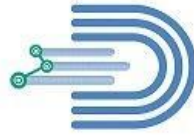


Figure 3 – Supervised Learning Building Blocks

1.11. Model Implementation High-level Steps

- A model implementation, to address a given problem involves several steps. Here are the key steps that are involved to implement a model. You can customize these steps as needed. We have developed these steps for learning purposes only.



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

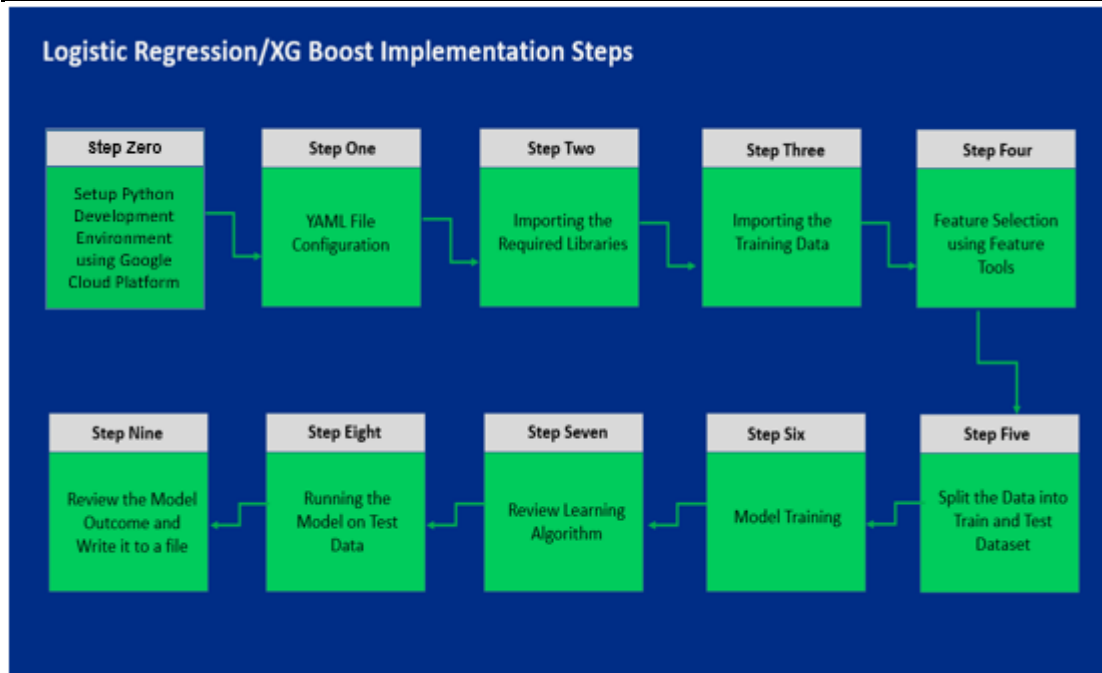


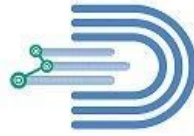
Figure 4 – Model Building Implementation Steps

1.12. Model Building Steps

- As we are implementing this use case using Google Cloud Platform, let's see how to upload data and access it through Python.

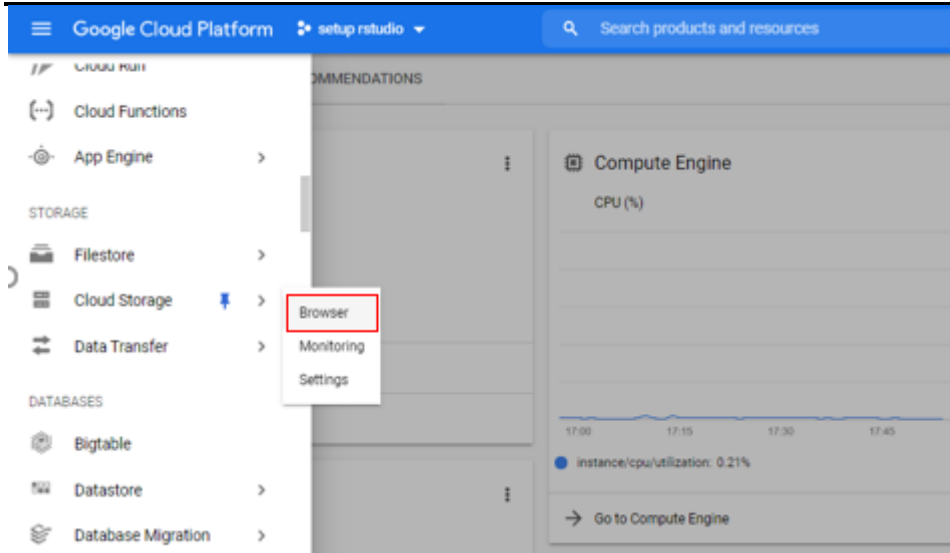
1.12.1. Create a Google Storage Bucket

- First, we'll create a storage bucket using the below steps.
- Navigation Menu > Storage > Cloud Storage > Browser

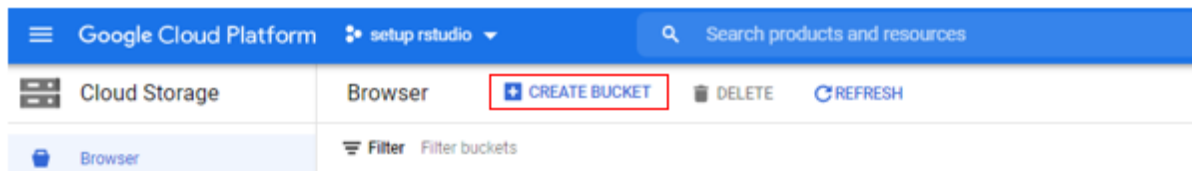


DeepSphere.AI
Enterprise AI and IIoT for Analytics

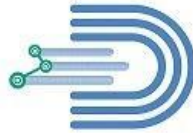
A Machine Learning Laboratory Manual



- Click on 'Create New Bucket' to open the bucket creation form.



- Enter a unique Name for your bucket and click on Continue.



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

Google Cloud Platform setup rstudio Search products and resources

Cloud Storage

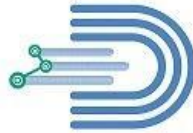
Browser Monitoring Settings

Create a Bucket

- Name your bucket**
Pick a globally unique, permanent name. [Naming guidelines](#)
Ex: 'example', 'example_bucket-1' or 'example.com'
Tip: Don't include any sensitive information
CONTINUE
- Choose where to store your data
- Choose a default storage class for your data
- Choose how to control access to objects
- Advanced settings (optional)

CREATE CANCEL

- Choose Region and location type, click on continue



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

Google Cloud Platform setup studio Search products and resources

Cloud Storage

Browser

Monitoring

Settings

Release notes

Create a Bucket

✓ Name your bucket

• Choose where to store your data

This permanent choice defines the geographic placement of your data and affects cost, performance and availability. [Learn more](#)

Location type

☒ Multi-region
Highest availability across largest area

☐ Dual-region
High availability and low latency across 2 regions

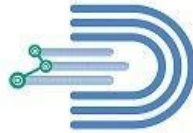
☐ Region
Lowest latency within a single region

Location

us (multiple regions in United States)

CONTINUE

- Choose Standard for default storage class. and click on continue

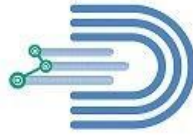


DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

The screenshot shows the Google Cloud Platform interface for creating a new bucket. The left sidebar contains navigation links for Cloud Storage, Browser, Monitoring, and Settings. The main content area is titled 'Create a Bucket' and shows a progress bar with three steps: 'Name your bucket', 'Choose where to store your data', and 'Choose a default storage class for your data'. The 'Standard' storage class is selected, with a description: 'Best for short-term storage and frequently accessed data'. Other options include Nearline, Coldline, and Archive. A red box highlights the 'CONTINUE' button at the bottom.

- Choose Uniform for Access control, click on continue.

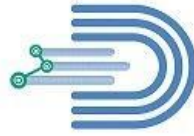


DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

The screenshot shows the Google Cloud Platform interface for creating a new storage bucket. The left sidebar contains navigation links for Cloud Storage, Browser, Monitoring, and Settings. The main content area is titled 'Create a Bucket' and shows a progress bar with three steps: 'Choose where to store your data', 'Choose a default storage class for your data', and 'Choose how to control access to objects'. The third step is currently active. Under 'Choose how to control access to objects', there are two sections: 'Prevent public access' and 'Access control'. The 'Access control' section has two radio button options: 'Uniform' (which is selected and highlighted with a red box) and 'Fine-grained'. Below the 'Uniform' option, there is a description: 'Ensure uniform access to all objects in the bucket by using only bucket-level permissions (IAM). This option becomes permanent after 90 days. [Learn more](#)'. Below the 'Fine-grained' option, there is a description: 'Specify access to individual objects by using object-level permissions (ACLs) in addition to your bucket-level permissions (IAM). [Learn more](#)'. A 'CONTINUE' button is visible at the bottom of the 'Access control' section. The 'Advanced settings (optional)' section is partially visible at the bottom.

- Click on Create and your storage bucket will be created.



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

The screenshot shows the 'Create a Bucket' wizard in the Google Cloud Platform console. The left sidebar shows 'Cloud Storage' with options for 'Browser', 'Monitoring', and 'Settings'. The main content area has two steps: 'Choose a default storage class for your data' and 'Choose how to control access to objects'. Under the second step, there are options for 'Prevent public access' (checked) and 'Access control' (Uniform selected, Fine-grained unselected). At the bottom, the 'CONTINUE' button is highlighted with a red box, and there are 'CREATE' and 'CANCEL' buttons below it.

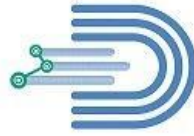
1.12.2. Moving Files into Google Storage Bucket

- Now, the storage bucket has been created. Click on the bucket name as shown below:

The screenshot shows the 'Browser' view of the Google Cloud Platform console. It displays a table of buckets. The bucket 'churndata20210721' is highlighted with a red box. The table has columns for Name, Created, Location type, Location, Default storage class, Updated, and Public access.

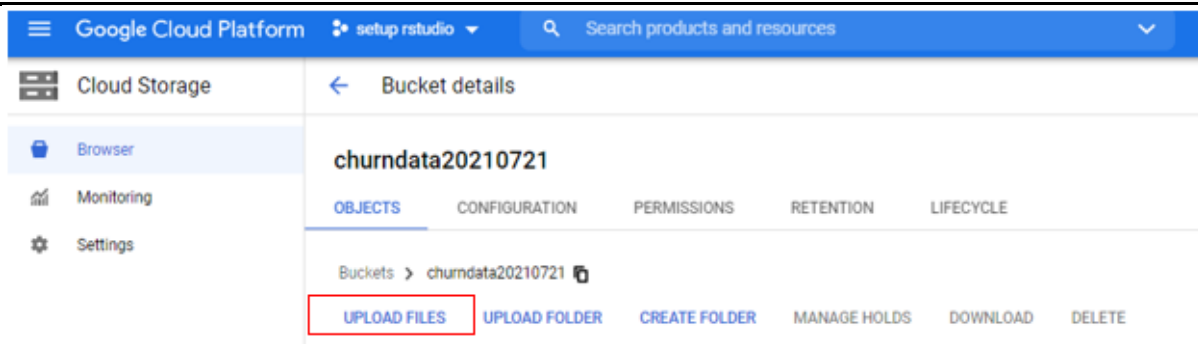
Name	Created	Location type	Location	Default storage class	Updated	Public access
churndata20210721	21 Jul 2021, 18:10:17	Multi-region	us (multiple re...)	Standard	21 Jul 2021, 18:10:17	Not public

- To upload data/files into your bucket, click on 'Upload Files'



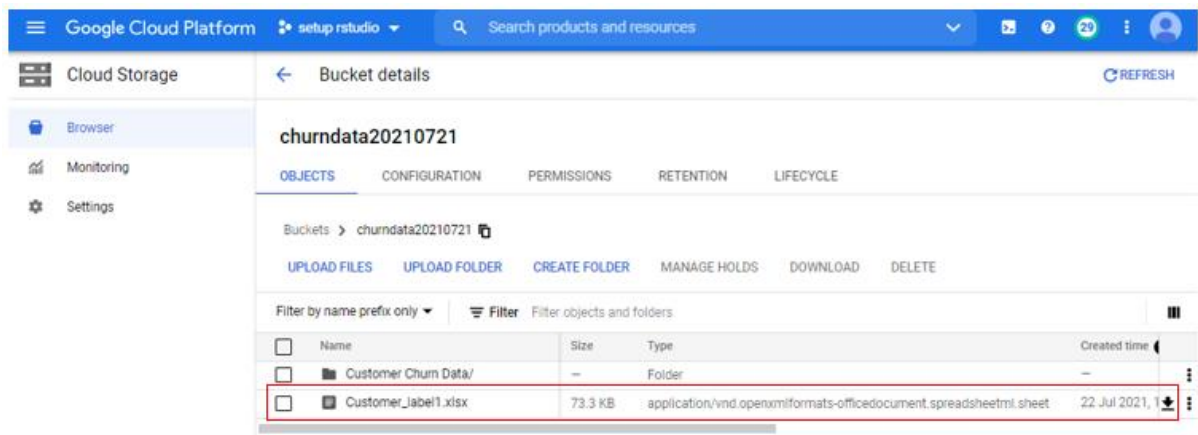
DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual



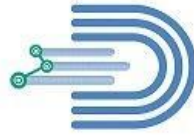
- **NOTE:** We can also upload a folder by clicking on 'Upload Folder'. In my case, I've uploaded a folder which contains all the required data to implement this use case.

- In the file dialog, go to the files that you want to upload and select them. After the upload completes, you should see the file name and information about the file, such as its size and type.



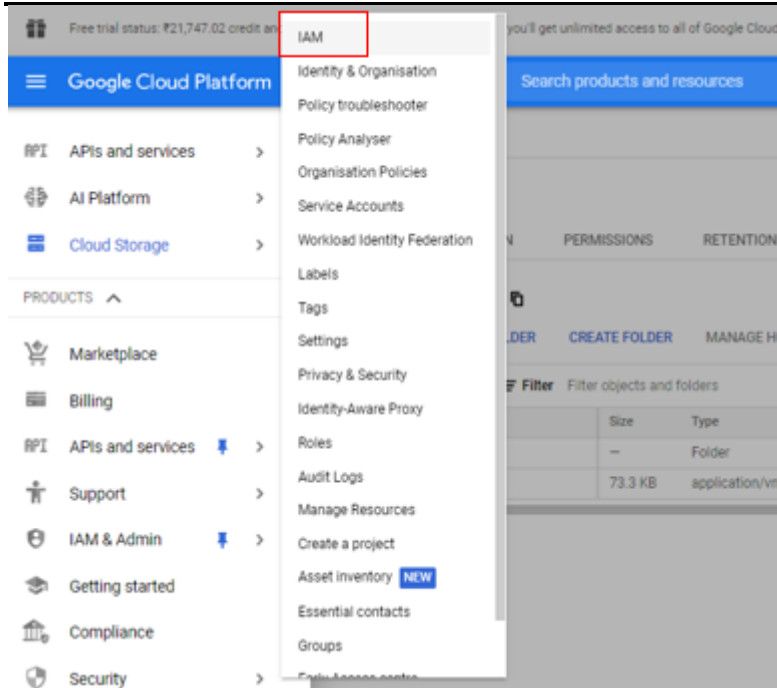
1.12.3. Create Service account and Private Key

- We'll first set up authentication by creating a service account and setting an environment variable.
- Navigation menu > Products > IAM & Admin > IAM

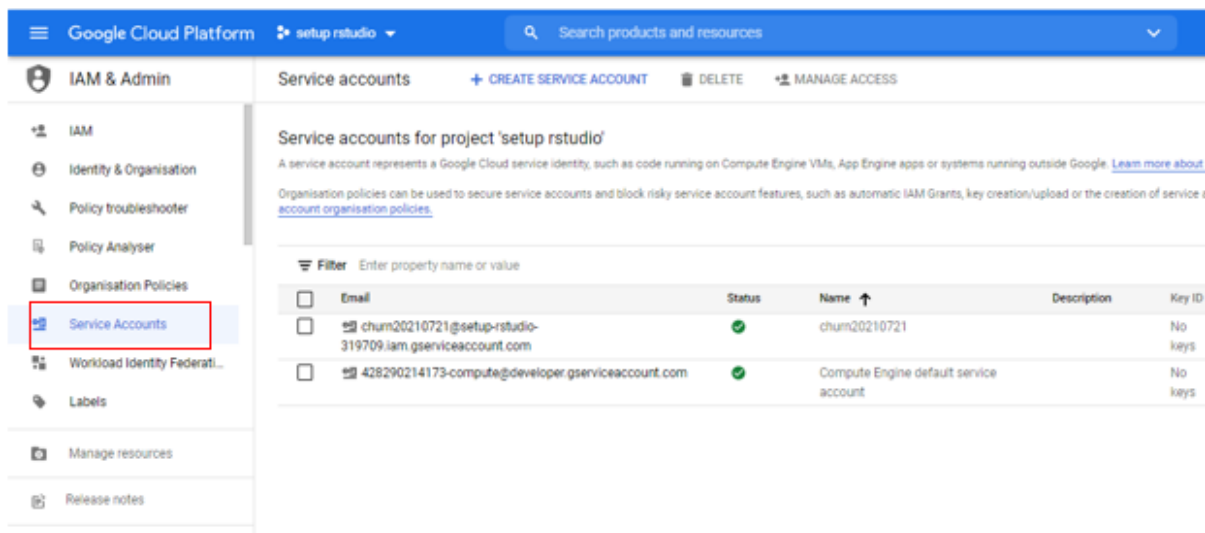


DeepSphere.AI
Enterprise AI and IIoT for Analytics

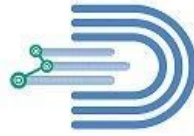
A Machine Learning Laboratory Manual



- Click on Service accounts



- Click on Create Service account



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

Service accounts for project 'setup-rstudio'

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps or systems running outside Google. [Learn more about service accounts.](#)

Organisation policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload or the creation of service accounts entirely. [Learn more about service account organisation policies.](#)

Filter: Enter property name or value

<input type="checkbox"/>	Email	Status	Name ↑	Description	Key ID	Key creation date	Actions
<input type="checkbox"/>	chum20210721@setup-rstudio-319709.iam.gserviceaccount.com	✓	chum20210721		2136b36a96695b657fc02e67ba886d2f9dde4d48	21 Jul 2021	

- Enter a name for the service account, click on Create and Continue

Create service account

1 Service account details

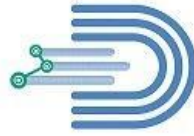
Service account name
Display name for this service account

Service account email: @setup-rstudio-319709.iam.gserviceaccount.com

Service account description
Describe what this service account will do

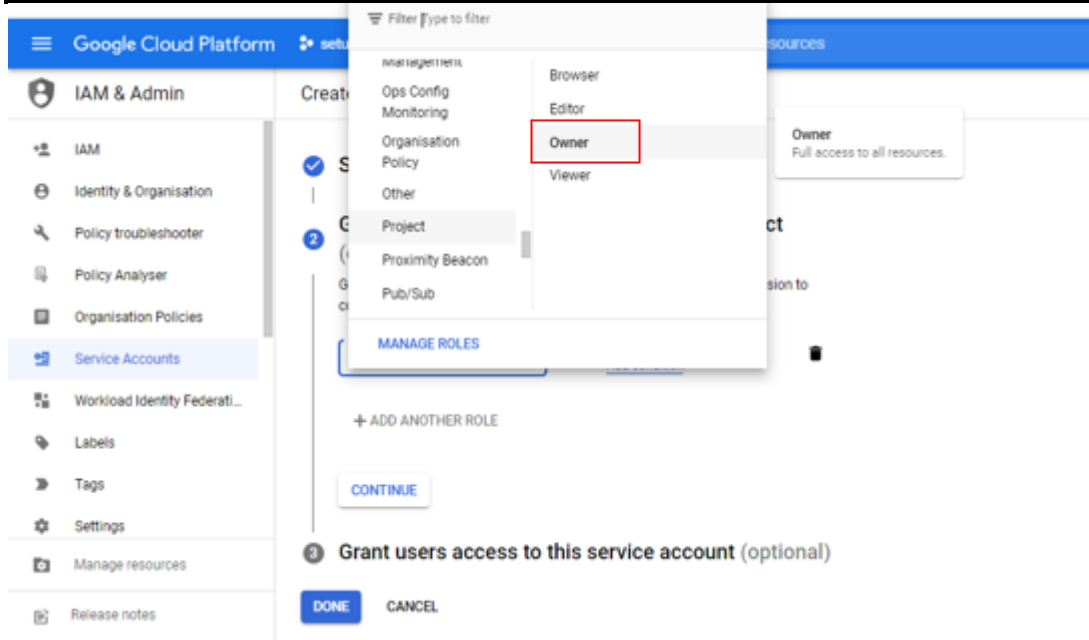
CREATE AND CONTINUE

- Grant this service account access to your project so that it has permission to complete specific actions on the resources in your project.
- Under Project, select 'Owner' as your role. Click on continue.

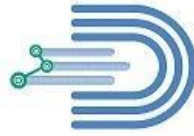


DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

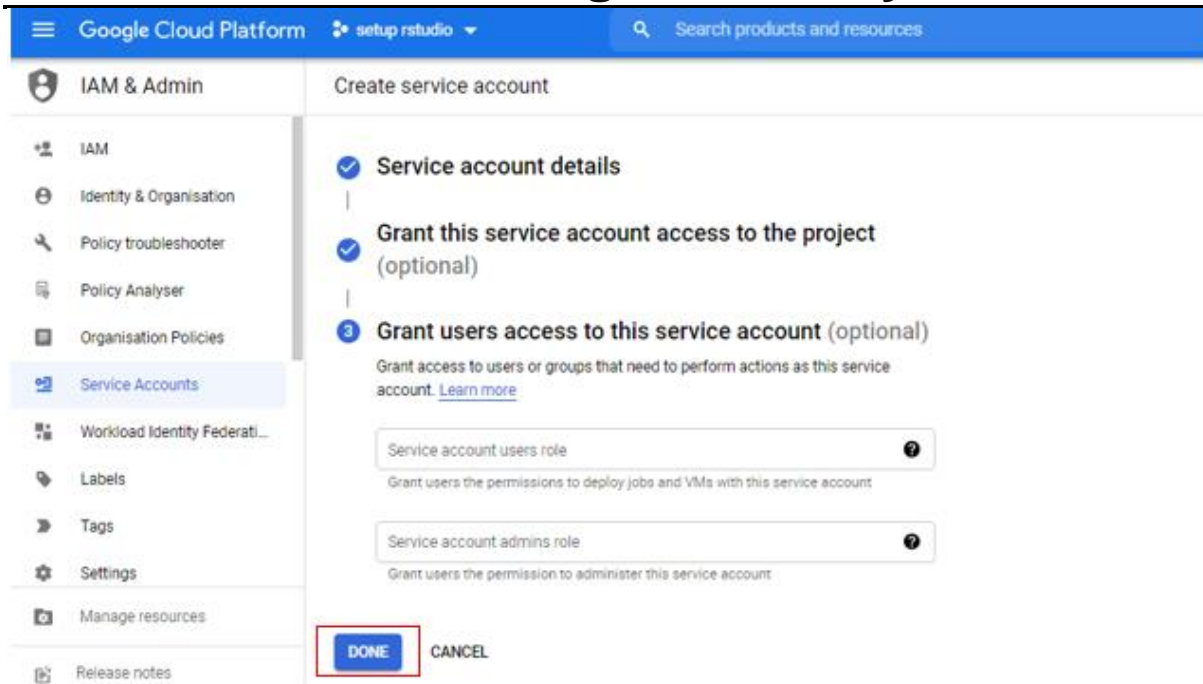


- Click on Done

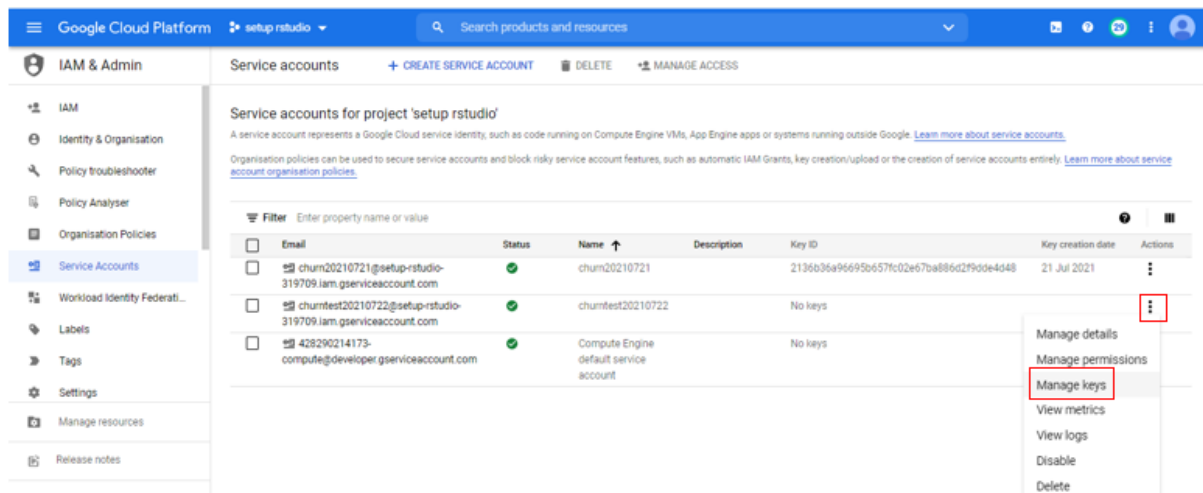


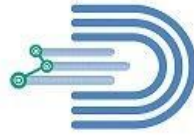
DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual



- Now, the service account has been created. Click on More Options (3 dots) and then Manage Keys as shown below:

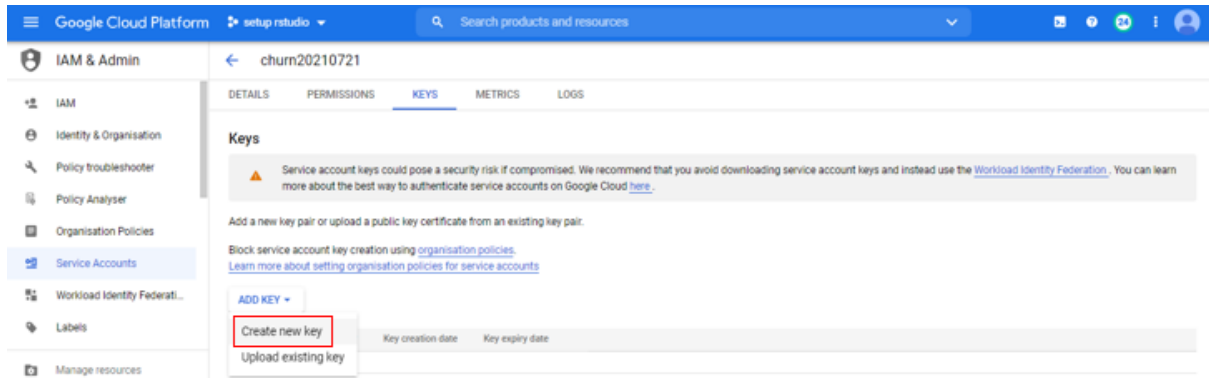




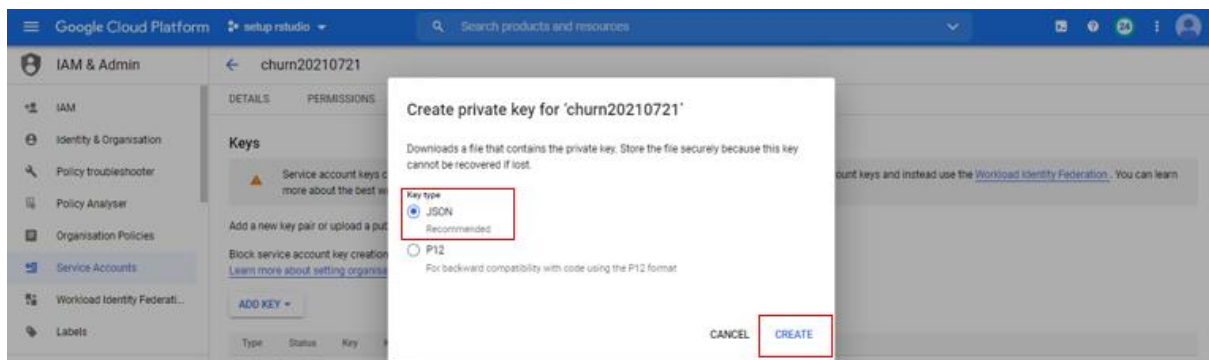
DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

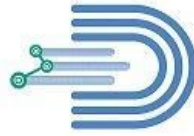
- Under Keys tab, click on Create New Key from the Add Key dropdown.



- Select the key type as JSON and click on Create

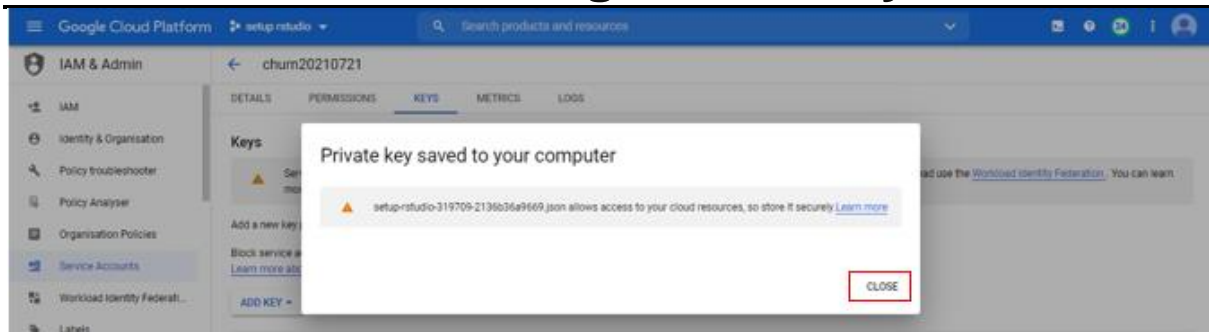


- Now, the key will be created and downloaded to your local system. You'll also get a pop up to indicate the key is saved to your computer as shown below.
- Click on Close



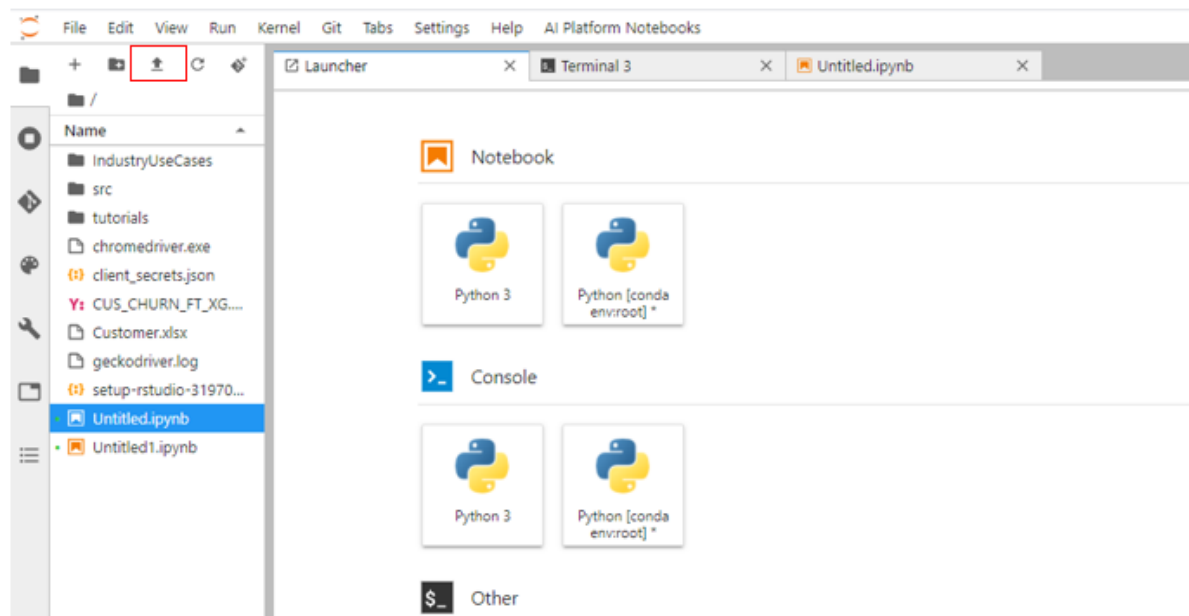
DeepSphere.AI
Enterprise AI and IIoT for Analytics

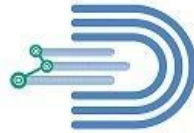
A Machine Learning Laboratory Manual



1.12.4. Access data from Google Storage Bucket using Python

- Please follow the steps instructed [here](#) to open the Jupyter Notebook using Notebook API.
- After you opened the notebook, you need to upload the downloaded key to the working environment by following the below steps:
- Click on the Upload icon:

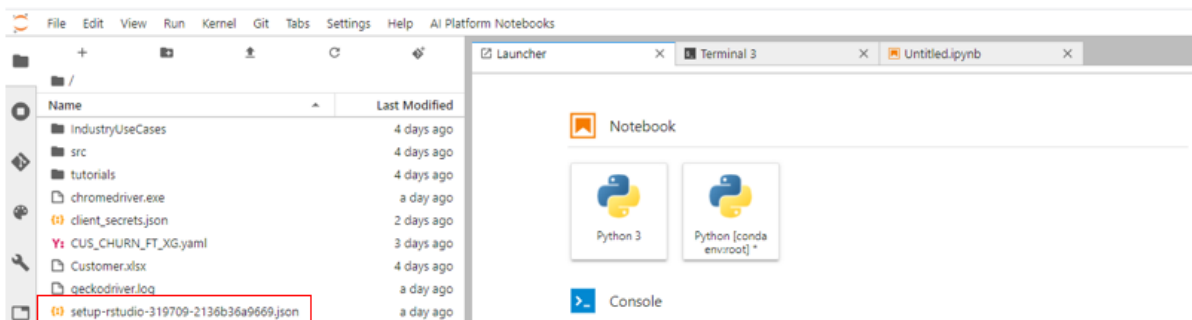




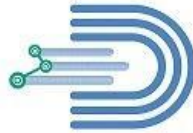
DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

- In the file dialog, go to the JSON file (key) you've downloaded and select it and it will be uploaded in the environment.




- You need to copy and paste the below data in the YAML file as shown below and import it in the working environment.



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

 CUS_CHURN_FT_XG.yaml - Notepad

File Edit Format View Help

```
# Date       : 04/05/2021
# Version    : v1
# (C)        : Deep Sphere, Inc.
#
```

```
#####
```

```
#####
```

```
[Data Source]
```

```
#####
```

```
DATA_SOURCE1= FILE
```

```
DATA_SOURCE2= HDFS
```

```
DATA_SOURCE3= SAP
```

```
DATA_SOURCE4= ORACLE
```

```
DATA_SOURCE5= MS
```

```
#####
```

```
[Data Source Connection String]
```

```
#####
```

```
SAP_CONNECTION_STRING=""
```

```
HDFS_CONNECTION_STRING=""
```

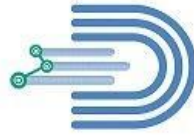
```
ORACLE_CONNECTION_STRING=""
```

```
MS_CONNECTION_STRING=""
```

```
#####
```

```
[FILE PATH]
```

```
#####
```

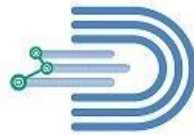


DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

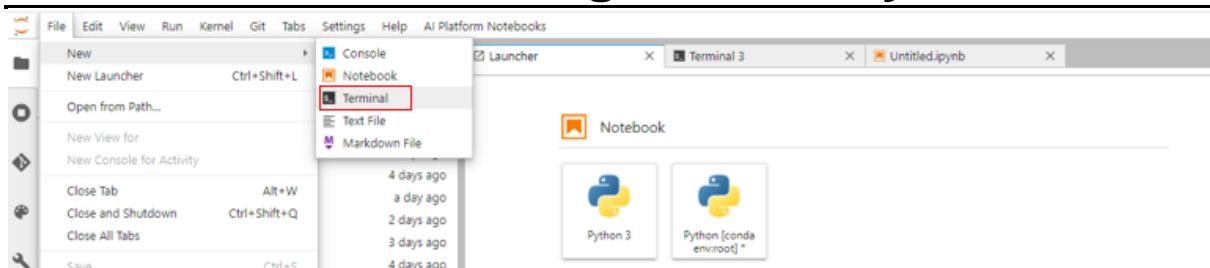
```
#####  
[FILE PATH]  
#####  
  
TRAINING_DATA = /Customer Churn Data/Customer_label.xlsx  
  
TRAINING_DATA_EXCEL_WORKSHEET = Customer_label  
|  
TRAINING_DATA(CUS) = /Customer Churn Data/Customer.xlsx'  
  
TRAINING_DATA_EXCEL_WORKSHEET(CUS) = CUSTOMER  
  
TRAINING_DATA(CBP) = /Customer Churn Data/customer_buying_pattern.xlsx'  
  
TRAINING_DATA_EXCEL_WORKSHEET(CBP) = customer_buying_pattern  
  
TRAINING_DATA(CPP) = /Customer Churn Data/Customer_Product_Price.xlsx'  
  
TRAINING_DATA_EXCEL_WORKSHEET(CPP) = Customer_Product_Price  
  
TRAINING_DATA(CSP) = /Customer Churn Data/Customer_Spending_Power.xlsx'  
  
TRAINING_DATA_EXCEL_WORKSHEET(CSP) = Customer_Spending_Power  
  
TRAINING_DATA(CSQ) = /Customer Churn Data/Customer_Service_Quality.xlsx'  
  
TRAINING_DATA_EXCEL_WORKSHEET(CSQ) = Customer_Service_Quality  
  
TRAINING_DATA(CS) = /Customer Churn Data/Customer_Satisfaction.xlsx'  
  
TRAINING_DATA_EXCEL_WORKSHEET(CS) = Customer_Satisfaction  
  
#####
```

- Now, we'll see how to install the required libraries. Open the Terminal by following the below steps:
- Click on File > New > Terminal

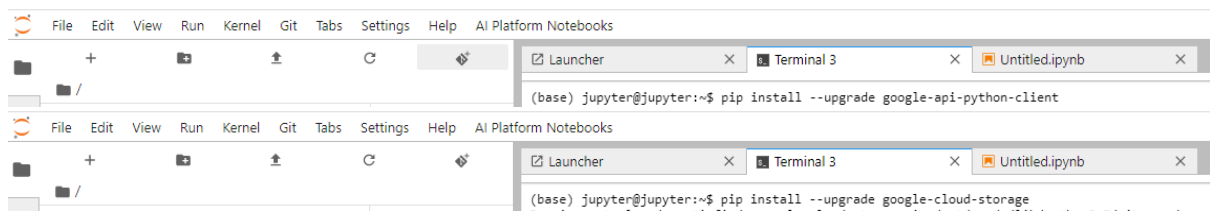


DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual



- Run these codes in the terminal to install the libraries



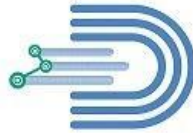
- Import the required libraries

```
try:
    from google.cloud import storage
    import google.cloud.storage
    import json
    import os
    import sys
    import pandas as pd
    import io
    from io import BytesIO
except Exception as e:
    print("Error : {}".format(e))
```

- Provide authentication credentials to your application code by setting the environment variable GOOGLE_APPLICATION_CREDENTIALS.

```
PATH = os.path.join(os.getcwd(), 'setup-rstudio-319709-2136b36a9669.json')
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = PATH
```

- Create a client object



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
storage_client = storage.Client(PATH)
storage_client
```

<google.cloud.storage.client.Client at 0x7f4aff19e810>

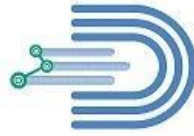
- Getting all files from the Google Storage Bucket which we created

```
bucket = storage_client.get_bucket('churndata20210721')
```

```
filename = [filename.name for filename in list(bucket.list_blobs(prefix='')) ]
filename
```

```
['Customer Churn Data/CUS_CHURN_FT_XG.yaml',
 'Customer Churn Data/Customer.xlsx',
 'Customer Churn Data/Customer_Product_Price.xlsx',
 'Customer Churn Data/Customer_Satisfaction.xlsx',
 'Customer Churn Data/Customer_Service_Quality.xlsx',
 'Customer Churn Data/Customer_Spending_Power.xlsx',
 'Customer Churn Data/Customer_label.xlsx',
 'Customer Churn Data/customer_buying_pattern.xlsx',
 'Customer_label1.xlsx']
```

- YAML file configuration



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
import configparser
import os

VAR_Config = configparser.ConfigParser(allow_no_value=True)

VAR_YAML_FILE_PATH = 'CUS_CHURN_FT_XG.yaml'

VAR_YAML_FILE_PATH

VAR_Config.read(VAR_YAML_FILE_PATH)

VAR_Data = VAR_Config.sections()

VAR_Config.sections()

VAR_Train_Data = VAR_Config['FILE PATH']['TRAINING_DATA']

VAR_Training_Data_Excel_Worsheet = VAR_Config['FILE PATH']['TRAINING_DATA_EXCEL_WORKSHEET']
print(VAR_Training_Data_Excel_Worsheet)

VAR_Train_Data_CUS = VAR_Config['FILE PATH']['TRAINING_DATA(CUS)']

VAR_Training_Data_Excel_Worsheet_CUS = VAR_Config['FILE PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CUS)']

print(VAR_Training_Data_Excel_Worsheet_CUS)

VAR_Training_Data_CBP = VAR_Config['FILE PATH']['TRAINING_DATA(CBP)']

VAR_Training_Data_Excel_Worsheet_CBP = VAR_Config['FILE PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CBP)']
print(VAR_Training_Data_Excel_Worsheet_CBP)

VAR_Training_Data_CPP = VAR_Config['FILE PATH']['TRAINING_DATA(CPP)']

VAR_Training_Data_Excel_Worsheet_CPP = VAR_Config['FILE PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CPP)']

print(VAR_Training_Data_Excel_Worsheet_CPP)

VAR_Training_Data_CSP = VAR_Config['FILE PATH']['TRAINING_DATA(CSP)']

VAR_Training_Data_Excel_Worsheet_CSP = VAR_Config['FILE PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CSP)']

print(VAR_Training_Data_Excel_Worsheet_CSP)

VAR_Training_Data_CSQ = VAR_Config['FILE PATH']['TRAINING_DATA(CSQ)']

VAR_Training_Data_Excel_Worsheet_CSQ = VAR_Config['FILE PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CSQ)']

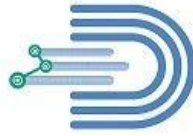
print(VAR_Training_Data_Excel_Worsheet_CSQ)

VAR_Training_Data_CS = VAR_Config['FILE PATH']['TRAINING_DATA(CS)']

VAR_Training_Data_Excel_Worsheet_CS = VAR_Config['FILE PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CS)']

print(VAR_Training_Data_Excel_Worsheet_CS)
```

- Import the Training data



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
import pandas as VAR_pd

import xgboost as VAR_xgb

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from xgboost import XGBClassifier
import featuretools as ft
import warnings

warnings.filterwarnings('ignore')

Customer_Label = VAR_pd.read_excel(VAR_Train_Data)

Customer = VAR_pd.read_excel(VAR_Train_Data_CUS)

Customer_Buying_Pattern = VAR_pd.read_excel(VAR_Training_Data_CBP)

Customer_Product_Price = VAR_pd.read_excel(VAR_Training_Data_CPP)

Customer_Spending_Power = VAR_pd.read_excel(VAR_Training_Data_CSP)

Customer_Service_Quality = VAR_pd.read_excel(VAR_Training_Data_CSQ)

Customer_Satisfaction = VAR_pd.read_excel(VAR_Training_Data_CS)
```

- Checking for Missing values

```
print(Customer_Label.isnull().sum())

print(Customer.isnull().sum())

print(Customer_Buying_Pattern.isnull().sum())

print(Customer_Product_Price.isnull().sum())

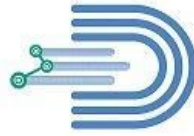
print(Customer_Spending_Power.isnull().sum())

print(Customer_Service_Quality.isnull().sum())

print(Customer_Satisfaction.isnull().sum())
```

```
CustomerID      0
Customer Code   0
CustomerRegion  0
CustomerLocation 0
CustomerChurn   1
dtype: int64
CustomerID      0
Customer Code   0
CustomerRegion  0
CustomerLocation 0
dtype: int64
CustomerID      0
```

- Handling Missing values



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
#Removing null values
Customer_Label.dropna(inplace=True)

#Re-Checking for null values
print(Customer_Label.isnull().sum())
```

```
CustomerID          0
Customer Code       0
CustomerRegion      0
CustomerLocation    0
CustomerChurn       0
dtype: int64
```

- Feature Selection using Feature tools

```
#DEFINING THE ENTITIES

es = ft.EntitySet(id="CUSTOMER_CHURN")

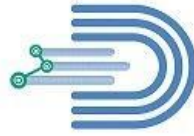
es1 = es.entity_from_dataframe(entity_id = 'Customer', dataframe = Customer, index='CustomerID')
es2 = es.entity_from_dataframe(entity_id = 'CustomerBuyingPattern', dataframe = Customer_Buying_Pattern, index='CBPID')
es3 = es.entity_from_dataframe(entity_id = 'CustomerProductPurchase', dataframe = Customer_Product_Price, index='CPPID')
es4 = es.entity_from_dataframe(entity_id = 'CustomerSpendingPower', dataframe = Customer_Spending_Power, index = 'CSPID')
es5 = es.entity_from_dataframe(entity_id = 'CustomerServiceQuality', dataframe = Customer_Service_Quality, index = 'CSQID')
es6 = es.entity_from_dataframe(entity_id = 'CustomerSatisfaction', dataframe = Customer_Satisfaction, index = 'CSID')

print(es)

#DEFINING THE RELATIONSHIPS

es.add_relationship(ft.Relationship(es['Customer']['CustomerID'], es['CustomerBuyingPattern']['CustomerID']))
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'], es['CustomerProductPurchase']['CustomerID']))
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'], es['CustomerSpendingPower']['CustomerID']))
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'], es['CustomerServiceQuality']['CustomerID']))
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'], es['CustomerSatisfaction']['CustomerID']))

#APPLYING DEEP SYNTHESIS
```



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

#APPLYING DEEP SYNTHESIS

```
feature_matrix_Customer, feature_defs = ft.dfs(entityset=es, target_entity="Customer", agg_primitives=["SUM"], max_depth=2)
```

```
feature_matrix_Customer
```

```
print(feature_matrix_Customer.shape)
```

#Removing Unwanted columns

```
VAR_Featuresft = feature_matrix_Customer.iloc[:,4:]
```

```
print(VAR_Featuresft.shape)
```

Entityset: CUSTOMER_CHURN

Entities:

Customer [Rows: 2103, Columns: 4]

CustomerBuyingPattern [Rows: 2103, Columns: 8]

CustomerProductPurchase [Rows: 2103, Columns: 6]

CustomerSpendingPower [Rows: 2103, Columns: 6]

CustomerServiceQuality [Rows: 2103, Columns: 6]

CustomerSatisfaction [Rows: 2103, Columns: 6]

Relationships:

No relationships

(2103, 19)

(2103, 15)

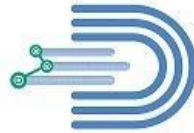
- Defining the label

```
VAR_label = Customer_Label.iloc[:,4:]
```

```
VAR_label
```

CustomerChurn	
0	1.0
1	1.0
2	0.0
3	0.0
4	1.0

- Split the Data into train and Test



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
VAR_X_TRAIN, VAR_X_TEST, VAR_Y_TRAIN, VAR_Y_TEST = train_test_split(VAR_Featuresft, VAR_label, test_size=0.20, random_state=0)
```

```
VAR_X_TRAIN
```

	SUM(CustomerBuyingPattern.Last year purchase)	SUM(CustomerBuyingPattern.Quantity(in lots))	SUM(CustomerBuyingPattern.average Monthly wise purchase)	SUM(CustomerBuyingPattern.c lifetime
CustomerID				
18929.0	57.0	120.0	74.0	
18811.0	2.0	32.0	25.0	
19652.0	0.0	0.0	0.0	
18649.0	22.0	92.0	36.0	
18056.0	3.0	90.0	32.0	
...	
19035.0	29.0	100.0	24.0	
19733.0	0.0	0.0	0.0	

- Training the model

```
#Training the Logistic regression model
VAR_Model1 = LogisticRegression()
VAR_Model1.fit(VAR_X_TRAIN,VAR_Y_TRAIN)
```

```
#Training the XGBoost model
VAR_Model2 = XGBClassifier(eta=0.01,gamma=10)
VAR_Model2.fit(VAR_X_TRAIN,VAR_Y_TRAIN)
```

```
[15:14:53] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
```

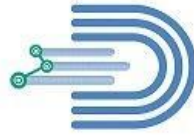
```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
               colsample_bynode=1, colsample_bytree=1, eta=0.01, gamma=10,
               gpu_id=-1, importance_type='gain', interaction_constraints='',
               learning_rate=0.00999999978, max_delta_step=0, max_depth=6,
               min_child_weight=1, missing=nan, monotone_constraints='()',
               n_estimators=100, n_jobs=1, num_parallel_tree=1, random_state=0,
               reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
               tree_method='exact', validate_parameters=1, verbosity=None)
```

- Review the Learning Algorithm

```
VAR_Model2.predict(VAR_X_TRAIN)
```

```
array([0., 0., 0., ..., 0., 0., 0.])
```

- Running the model on test data



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
#Prediction using Logistic regression
vAR_Labels_predLG = vAR_Model1.predict(vAR_X_TEST)

#Prediction using XGBoost
vAR_Labels_predXG = vAR_Model2.predict(vAR_X_TEST)
```

- Checking Accuracy of the Model Output

```
# Checking accuracy for Logistic Regression
from sklearn.metrics import accuracy_score
print(accuracy_score(vAR_Y_TEST, vAR_Labels_predLG))

# Checking accuracy for XGBoost
from sklearn.metrics import accuracy_score
accuracy_score(vAR_Y_TEST, vAR_Labels_predXG)

0.5534441805225653
0.5653206650831354
```

1.12.5. Model Building Code Block

- We need to implement these code blocks after completing the below steps (which are explained from [1.12.1](#) section)
 1. Creating Google Storage Bucket using Console ,
 2. Uploading Datasets to Google Storage Bucket using Console,
 3. Create and Importing the JSON key file into the working environment,
 4. Importing the YAML configuration file into the working environment.

Install the required libraries by running these commands in the terminal

```
pip install --upgrade google-api-python-client
```

```
pip install --upgrade google-cloud-storage
```

Import the required libraries

```
try:
```



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
from google.cloud import storage
import google.cloud.storage
import json
import os
import sys
import pandas as pd
import io
from io import BytesIO
except Exception as e:
    print("Error : {}".format(e))
```

Provide authentication credentials to your application code by setting the environment variable GOOGLE_APPLICATION_CREDENTIALS.

```
PATH = os.path.join(os.getcwd(), 'setup-rstudio-319709-2136b36a9669.json')
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = PATH
```

Create a client object

```
storage_client = storage.Client(PATH)
storage_client
```

Getting all files from the Google Storage Bucket which we created

```
bucket = storage_client.get_bucket('churndata20210721')

filename = [filename.name for filename in list(bucket.list_blobs(prefix='')) ]
filename
```

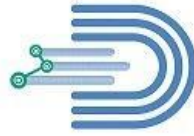
YAML file configuration

```
import configparser

import os

vAR_Config = configparser.ConfigParser(allow_no_value=True)

vAR_YAML_FILE_PATH = 'CUS_CHURN_FT_XG.yaml'
```



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
vAR_YAML_FILE_PATH
```

```
vAR_Config.read(vAR_YAML_FILE_PATH)
```

```
vAR_Data = vAR_Config.sections()
```

```
vAR_Config.sections()
```

```
vAR_Train_Data = vAR_Config['FILE PATH']['TRAINING_DATA']
```

```
vAR_Training_Data_Excel_Worsheet = vAR_Config['FILE  
PATH']['TRAINING_DATA_EXCEL_WORKSHEET']  
print(vAR_Training_Data_Excel_Worsheet)
```

```
vAR_Train_Data_CUS = vAR_Config['FILE PATH']['TRAINING_DATA(CUS)']
```

```
vAR_Training_Data_Excel_Worsheet_CUS = vAR_Config['FILE  
PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CUS)']
```

```
print(vAR_Training_Data_Excel_Worsheet_CUS)
```

```
vAR_Training_Data_CBP = vAR_Config['FILE PATH']['TRAINING_DATA(CBP)']
```

```
vAR_Training_Data_Excel_Worsheet_CBP = vAR_Config['FILE  
PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CBP)']
```

```
print(vAR_Training_Data_Excel_Worsheet_CBP)
```

```
vAR_Training_Data_CPP = vAR_Config['FILE PATH']['TRAINING_DATA(CPP)']
```

```
vAR_Training_Data_Excel_Worsheet_CPP = vAR_Config['FILE  
PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CPP)']
```

```
print(vAR_Training_Data_Excel_Worsheet_CPP)
```

```
vAR_Training_Data_CSP = vAR_Config['FILE PATH']['TRAINING_DATA(CSP)']
```

```
vAR_Training_Data_Excel_Worsheet_CSP = vAR_Config['FILE  
PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CSP)']
```



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
print(vAR_Training_Data_Excel_Worsheet_CSP)
```

```
vAR_Training_Data_CSQ = vAR_Config['FILE PATH']['TRAINING_DATA(CSQ)']
```

```
vAR_Training_Data_Excel_Worsheet_CSQ = vAR_Config['FILE  
PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CSQ)']
```

```
print(vAR_Training_Data_Excel_Worsheet_CSQ)
```

```
vAR_Training_Data_CS = vAR_Config['FILE PATH']['TRAINING_DATA(CS)']
```

```
vAR_Training_Data_Excel_Worsheet_CS = vAR_Config['FILE  
PATH']['TRAINING_DATA_EXCEL_WORKSHEET(CS)']
```

```
print(vAR_Training_Data_Excel_Worsheet_CS)
```

Data to be updated in the YAML File (Copy and Paste it in your YAML file)

```
# Date : 23/07/2021
```

```
# Version : v1
```

```
# (C) : Deep Sphere, Inc.
```

```
#
```

```
#####
```

```
#####
```

```
[Data Source]
```

```
#####
```

```
DATA_SOURCE1= FILE
```

```
DATA_SOURCE2= HDFS
```

```
DATA_SOURCE3= SAP
```

```
DATA_SOURCE4= ORACLE
```

```
DATA_SOURCE5= MS
```

```
#####
```

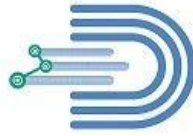
```
[Data Source Connection String]
```

```
#####
```

```
SAP_CONNECTION_STRING=""
```

```
HDFS_CONNECTION_STRING=""
```

```
ORACLE_CONNECTION_STRING=""
```



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
MS_CONNECTION_STRING=""
```

```
#####  
[FILE PATH]  
#####
```

```
TRAINING_DATA = /Customer Churn Data/Customer_label.xlsx
```

```
TRAINING_DATA_EXCEL_WORKSHEET = Customer_label
```

```
TRAINING_DATA(CUS) = /Customer Churn Data/Customer.xlsx'
```

```
TRAINING_DATA_EXCEL_WORKSHEET(CUS) = CUSTOMER
```

```
TRAINING_DATA(CBP) = /Customer Churn Data/customer_buying_pattern.xlsx'
```

```
TRAINING_DATA_EXCEL_WORKSHEET(CBP) = customer_buying_pattern
```

```
TRAINING_DATA(CPP) = /Customer Churn Data/Customer_Product_Price.xlsx'
```

```
TRAINING_DATA_EXCEL_WORKSHEET(CPP) = Customer_Product_Price
```

```
TRAINING_DATA(CSP) = /Customer Churn Data/Customer_Spending_Power.xlsx'
```

```
TRAINING_DATA_EXCEL_WORKSHEET(CSP) = Customer_Spending_Power
```

```
TRAINING_DATA(CSQ) = /Customer Churn Data/Customer_Service_Quality.xlsx'
```

```
TRAINING_DATA_EXCEL_WORKSHEET(CSQ) = Customer_Service_Quality
```

```
TRAINING_DATA(CS) = /Customer Churn Data/Customer_Satisfaction.xlsx'
```

```
TRAINING_DATA_EXCEL_WORKSHEET(CS) = Customer_Satisfaction
```

```
#####
```

```
# Import the Training data
```

```
import pandas as vAR_pd
```




DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
import xgboost as vAR_xgb

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from xgboost import XGBClassifier
import featuretools as ft
import warnings

warnings.filterwarnings('ignore')

Customer_Label = vAR_pd.read_excel(vAR_Train_Data)

Customer = vAR_pd.read_excel(vAR_Train_Data_CUS)

Customer_Buying_Pattern = vAR_pd.read_excel(vAR_Training_Data_CBP)

Customer_Product_Price = vAR_pd.read_excel(vAR_Training_Data_CPP)

Customer_Spending_Power = vAR_pd.read_excel(vAR_Training_Data_CSP)

Customer_Service_Quality = vAR_pd.read_excel(vAR_Training_Data_CSQ)

Customer_Satisfaction = vAR_pd.read_excel(vAR_Training_Data_CS)

# Checking for Missing values

print(Customer_Label.isnull().sum())

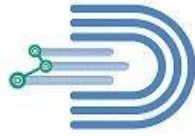
print(Customer.isnull().sum())

print(Customer_Buying_Pattern.isnull().sum())

print(Customer_Product_Price.isnull().sum())

print(Customer_Spending_Power.isnull().sum())

print(Customer_Service_Quality.isnull().sum())
```



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
print(Customer_Satisfaction.isnull().sum())
```

Handling Missing values

```
#Removing null values  
Customer_Label.dropna(inplace=True)
```

```
#Re-Checking for null values  
print(Customer_Label.isnull().sum())
```

Feature Selection using Feature tools

#DEFINING THE ENTITIES

```
es = ft.EntitySet(id="CUSTOMER_CHURN")
```

```
es1 = es.entity_from_dataframe(entity_id = 'Customer', dataframe = Customer,  
index='CustomerID')
```

```
es2 = es.entity_from_dataframe(entity_id = 'CustomerBuyingPattern', dataframe =  
Customer_Buying_Pattern, index='CBPID')
```

```
es3 = es.entity_from_dataframe(entity_id = 'CustomerProductPurchase', dataframe =  
Customer_Product_Price, index='CPPID')
```

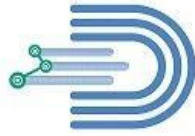
```
es4 = es.entity_from_dataframe(entity_id = 'CustomerSpendingPower', dataframe =  
Customer_Spending_Power, index = 'CSPID')
```

```
es5 = es.entity_from_dataframe(entity_id = 'CustomerServiceQuality', dataframe =  
Customer_Service_Quality, index = 'CSQID')
```

```
es6 = es.entity_from_dataframe(entity_id = 'CustomerSatisfaction', dataframe =  
Customer_Satisfaction, index = 'CSID')
```

```
print(es)
```

#DEFINING THE RELATIONSHIPS



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'],es['CustomerBuyingPattern']['CustomerID']))
```

```
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'],es['CustomerProductPurchase']['CustomerID']))
```

```
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'],es['CustomerSpendingPower']['CustomerID']))
```

```
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'],es['CustomerServiceQuality']['CustomerID']))
```

```
es.add_relationship(ft.Relationship(es['Customer']['CustomerID'],es['CustomerSatisfaction']['CustomerID']))
```

#APPLYING DEEP SYNTHESIS

```
feature_matrix_Customer, feature_defs = ft.dfs(entityset=es, target_entity="Customer",  
agg_primitives=["SUM"], max_depth=2)
```

```
feature_matrix_Customer
```

```
print(feature_matrix_Customer.shape)
```

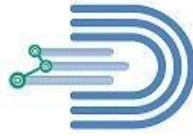
```
#Removing Unwanted columns
```

```
vAR_Featuresft = feature_matrix_Customer.iloc[:,4:]  
print(vAR_Featuresft.shape)
```

```
# Defining the label
```

```
vAR_label = Customer_Label.iloc[:,4:]  
vAR_label
```

```
# Split the Data into train and Test
```



DeepSphere.AI
Enterprise AI and IIoT for Analytics

A Machine Learning Laboratory Manual

```
vAR_X_TRAIN, vAR_X_TEST, vAR_Y_TRAIN, vAR_Y_TEST = train_test_split(vAR_Featuresft,  
vAR_label, test_size=0.20, random_state=0)
```

```
vAR_X_TRAIN
```

Training the model

```
#Training the logistic regression model  
vAR_Model1 = LogisticRegression()  
print(vAR_Model1.fit(vAR_X_TRAIN,vAR_Y_TRAIN))
```

```
#Training the XGBoost model  
vAR_Model2 = XGBClassifier(eta=0.01,gamma=10)  
vAR_Model2.fit(vAR_X_TRAIN,vAR_Y_TRAIN)
```

Review the Learning Algorithm

```
vAR_Model2.predict(vAR_X_TRAIN)
```

Running the model on test data

```
#Prediction using Logistic regression  
vAR_Labels_predLG = vAR_Model1.predict(vAR_X_TEST)
```

```
#Prediction using XGBoost  
vAR_Labels_predXG = vAR_Model2.predict(vAR_X_TEST)
```

Checking Accuracy of the Model Output

```
# Checking accuracy for Logistic Regression  
from sklearn.metrics import accuracy_score  
print(accuracy_score(vAR_Y_TEST, vAR_Labels_predLG))
```

```
# Checking accuracy for XGBoost  
from sklearn.metrics import accuracy_score  
accuracy_score(vAR_Y_TEST, vAR_Labels_predXG)
```