



# EMPLOYEE ATTRITION CLASSIFICATION DATASET

An In-Depth Synthetic Simulation for Attrition  
Analysis and Prediction

## ABSTRACT

This report details the process of deploying a machine learning model using AWS SageMaker, using the Employee Attrition Classification dataset from Kaggle. The dataset which includes various features related to employee demographics, job roles, serves as a basis for predicting employee turnover within an organization. The report outlines the entire workflow, from data preparation to model training, evaluation, and deployment. Key aspects of AWS are explored, including configuring the SageMaker environment, training the model, and deploying it for inference. Additionally, the project required the management of data and model files, which were stored in an Amazon S3 bucket. The objective is to demonstrate a practical application of AWS SageMaker in implementing machine learning solutions for business problems, showcasing the effectiveness and efficiency of cloud-based machine learning deployments in addressing employee attrition and enhancing organizational retention strategies.

**Thapelo Lenzi**

Deploying Machine Learning Models in AWS  
SageMaker

## Dataset

This dataset consists of 74 498 entries with each including a unique Employee ID and features that influence employee attrition. The goal is to understand the factors contributing to attrition (Whether the employee has left the company, encoded as 0 for stayed and 1 for left) and develop predictive models to identify at-risk employees. The features include:

- Age
- Gender
- Years at company
- Monthly income
- Job role
- Work-life balance
- Education level
- And more

<https://www.kaggle.com/datasets/stealthtechnologies/employee-attritiondataset?resource=download&select=train.csv>

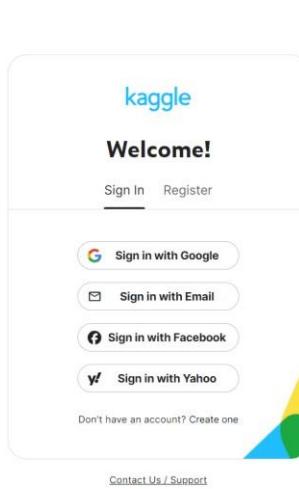
## Data Extraction

Obtaining the dataset from Kaggle:

Search for the Employee Attrition Classification dataset on Kaggle.

The screenshot shows the Kaggle website interface. On the left, there's a sidebar with navigation links: Create, Home, Competitions, Datasets (which is selected), Models, Code, Discussions, Learn, and More. Below that is a "View Active Events" section. The main content area shows a dataset page for "Employee Attrition Classification Dataset". At the top right are "Sign In" and "Register" buttons, along with a search bar and other navigation icons. The dataset title is "Employee Attrition Classification Dataset" with a subtitle "An In-Depth Synthetic Simulation for Attrition Analysis and Prediction". There's a preview image of an office interior. Below the title are tabs for "Data Card", "Code (15)", "Discussion (1)", and "Suggestions (0)". The "About Dataset" section contains detailed information about the dataset, including its size (74,498 samples), purpose (attrition analysis and prediction), and features (demographics, job-related features, personal circumstances). It also notes that the dataset is ideal for HR analytics, machine learning model development, and demonstrating advanced data analysis techniques. To the right of the dataset details are sections for "Usability" (rating 10.00), "License" (Apache 2.0), "Expected update frequency" (Annually), and "Tags" (Tabular, Finance).

Sign In:



Go to the dataset page and click on the download button to manually download the dataset:

The screenshot shows the Kaggle interface for the "Employee Attrition Classification Dataset". The left sidebar includes links for Home, Competitions, Datasets (selected), Models, Code, Discussions, Learn, More, Your Work, and View Active Events. The main content area is titled "Employee Attrition Classification Dataset" and contains a "Data Card" with details like "Code (15)", "Discussion (1)", and "Suggestions (0)". It also includes a detailed description of the dataset, such as "Age: The age of the employee, ranging from 18 to 60 years.", "Gender: The gender of the employee", and "Years at Company: The number of years the employee has been working at the company". Below this is a "View more" section. A central panel shows the "train.csv" file (7.64 MB) with options to "Detail", "Compact", or "Column" view, indicating it has 10 of 24 columns. The Data Explorer sidebar on the right shows a total size of 9.55 MB and lists "test.csv" and "train.csv".

## Data Preparation

The data preparation and exploratory data analysis stages of this project are done using Google Colab.

Loading Libraries, Importing necessary libraries for data manipulation and visualization:

```
Necessary Packages

[✓] [20] import pandas as pd
      import numpy as np
      import matplotlib.pyplot as plt
      import seaborn as sn
      pd.set_option('display.max_columns', None)
      import warnings
      warnings.filterwarnings('ignore')

      from sklearn.preprocessing import LabelEncoder
```

Loading the data into a pandas data frame:

```
[4] #Load the data
      data = pd.read_csv("/content/Attrition.csv")
```

# Exploratory Data Analysis

The overview of the dataset:

```
[5] #Display the top 5 rows of the dataset  
data.head()
```

	Employee ID	Age	Gender	Years at Company	Job Role	Monthly Income	Work-Life Balance	Job Satisfaction	Performance Rating	Number of Promotions	Overtime	Distance from Home	Education Level
0	8410	31	Male	19	Education	5390	Excellent	Medium	Average	2	No	22	Associate Degree
1	64756	59	Female	4	Media	5534	Poor	High	Low	3	No	21	Master's Degree
2	30257	24	Female	10	Healthcare	8159	Good	High	Low	0	No	11	Bachelor's Degree
3	65791	36	Female	7	Education	3989	Good	High	High	1	No	27	High School
4	65026	56	Male	41	Education	4821	Fair	Very High	Average	0	Yes	71	High School

```
▶ #Display the number of rows and columns  
data.shape
```

```
(8185, 24)
```

```
▶ #Data type summary  
data.info()
```

```
RangeIndex: 8186 entries, 0 to 8185  
Data columns (total 24 columns):  
 #   Column           Non-Null Count  Dtype     
 ---    
 0   Employee ID      8186 non-null   int64    
 1   Age              8186 non-null   int64    
 2   Gender            8186 non-null   object   
 3   Years at Company 8186 non-null   int64    
 4   Job Role          8186 non-null   object   
 5   Monthly Income    8186 non-null   int64    
 6   Work-Life Balance 8186 non-null   object   
 7   Job Satisfaction   8186 non-null   object   
 8   Performance Rating 8186 non-null   object   
 9   Number of Promotions 8186 non-null   int64    
 10  Overtime           8186 non-null   object   
 11  Distance from Home 8186 non-null   int64    
 12  Education Level    8186 non-null   object   
 13  Marital Status     8186 non-null   object   
 14  Number of Dependents 8186 non-null   int64    
 15  Job Level          8186 non-null   object   
 16  Company Size        8186 non-null   object   
 17  Company Tenure      8186 non-null   int64    
 18  Remote Work         8186 non-null   object   
 19  Leadership Opportunities 8186 non-null   object   
 20  Innovation Opportunities 8186 non-null   object   
 21  Company Reputation   8185 non-null   object   
 22  Employee Recognition 8185 non-null   object   
 23  Attrition           8185 non-null   object  
dtypes: int64(8), object(16)  
memory usage: 1.5 MB
```

```
[7] #Statistical summary  
data.describe()
```

	Employee ID	Age	Years at Company	Monthly Income	Number of Promotions	Distance from Home	Number of Dependents	Company Tenure
count	8186.000000	8186.000000	8186.000000	8186.000000	8186.000000	8186.000000	8186.000000	8186.000000
mean	37287.220743	38.615075	15.667237	7336.613609	0.827144	49.724408	1.649890	55.863059
std	21446.665299	12.135169	11.291536	2141.167892	0.996414	28.518282	1.556062	25.546607
min	8.000000	18.000000	1.000000	1855.000000	0.000000	1.000000	0.000000	2.000000
25%	18757.250000	28.000000	7.000000	5714.000000	0.000000	25.000000	0.000000	36.000000
50%	37088.500000	39.000000	13.000000	7373.000000	0.000000	50.000000	1.000000	56.000000
75%	55937.750000	49.000000	23.000000	8879.500000	1.000000	74.000000	3.000000	76.000000
max	74488.000000	59.000000	51.000000	15495.000000	4.000000	99.000000	6.000000	127.000000

The above snippets provide a sample of the data and understand the structure of the dataset, including column names and initial values. They provide the dimensions of the dataset, which can be observed as 8185 rows and 24 columns, which help in understanding the size of the dataset.

The data type and statistical summaries, include the number of non-null entries in each column and datatypes together with metrics such as the mean, minimum and maximum. These help with understanding the distribution and range of the numeric data and identifying any missing values. From the above, it can be noted that the dataset has two data types (int64 and object) and the last three columns have some null values.

## Handling Missing Data

The missing values can be explicitly displayed, as shown below. Since there is only one value missing in each of the three columns and there are a lot of entries, the missing values can be dropped and will not make any significant change in the analysis.

```
[11] #Get number of duplicated data
     data.duplicated().sum()

→ 0

[8] #Get number of empty rows in each column
     data.isnull().sum()

→ Employee ID      0
   Age              0
   Gender           0
   Years at Company 0
   Job Role          0
   Monthly Income    0
   Work-Life Balance 0
   Job Satisfaction    0
   Performance Rating 0
   Number of Promotions 0
   Overtime           0
   Distance from Home 0
   Education Level    0
   Marital Status      0
   Number of Dependents 0
   Job Level           0
   Company Size         0
   Company Tenure       0
   Remote Work          0
   Leadership Opportunities 0
   Innovation Opportunities 0
   Company Reputation    1
   Employee Recognition    1
   Attrition            1
dtype: int64

[12] #Drop the empty values
     data.dropna(inplace = True)

[14] #verify the empty values are removed
     data.isnull().values.any()

→ False
```

```
[19] #Display the count summary for each column
for column in data.columns:
    if data[column].dtype == object:
        print(f'{column}: {data[column].unique()}' )
        print(data[column].value_counts())
        print('.....')

[19]
Innovation Opportunities: ['No' 'Yes']
Innovation Opportunities
No      6873
Yes     1312
Name: count, dtype: int64
.....
Company Reputation: ['Excellent' 'Fair' 'Poor' 'Good']
Company Reputation
Good      4936
Poor       1667
Fair       1655
Excellent   827
Name: count, dtype: int64
.....
Employee Recognition: ['Medium' 'Low' 'High' 'Very High']
Employee Recognition
Low       3267
Medium     2439
High       2862
Very High   417
Name: count, dtype: int64
.....
Attrition: ['Stayed' 'Left']
Attrition
Stayed    4282
Left      3903
Name: count, dtype: int64
.....
```

The above is a summary of each column. In the dependent column (Attrition), there are two unique values with 'Stayed' having the greater count.

## Feature Engineering

The dependent column is originally in text form. In order for it to be used in the model, it is converted to numerical form using the LabelEncoder from the scikit-learn library, as seen in the snippet below.

```
le = LabelEncoder()
data['Attrition'] = le.fit_transform(data['Attrition'])
data.head(5)

Number of Promotions Overtime Distance from Home Education Level Marital Status Number of Dependents Job Level Company Size Company Tenure Remote Work Leadership Opportunities Innovation Opportunities Company Reputation Employee Recognition Attrition
2      No      22 Associate Degree Married          0 Mid Medium      89 No      No      No      No Excellent Medium 1
3      No      21 Master's Degree Divorced         3 Mid Medium      21 No      No      No      No Fair Low 1
0      No      11 Bachelor's Degree Married         3 Mid Medium      74 No      No      No      No Poor Low 1
1      No      27 High School Single            2 Mid Small       50 Yes     No      No      No Good Medium 1
0      Yes     71 High School Divorced         0 Senior Medium      68 No      No      No      No Fair Medium 1

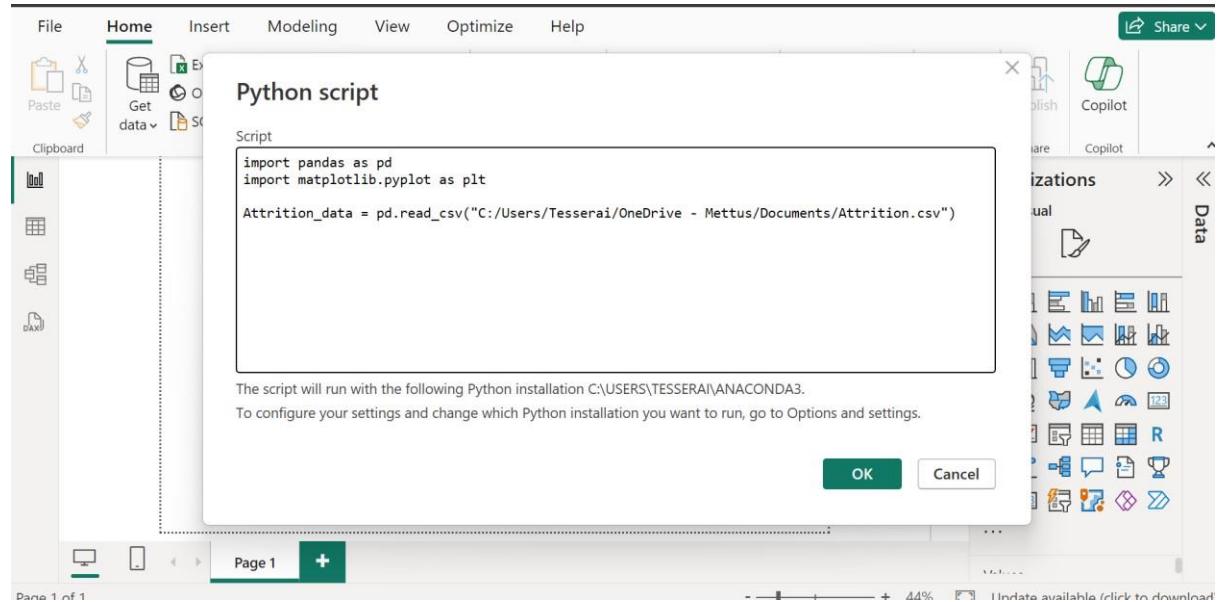
le.classes_
array(['Left', 'Stayed'], dtype=object)

[23] Employee_Attrition = le.classes_
print(Employee_Attrition)

['Left' 'Stayed']
```

## Data Visualization using PowerBI Desktop

Connecting to Python Script in PowerBI Desktop:



The screenshot shows the PowerBI Desktop interface with the 'Home' tab selected. A 'Navigator' pane is open on the left, showing a tree view with 'Python [1]' expanded, revealing 'Attrition\_data'. On the right, a preview of the 'Attrition\_data' table is displayed, showing the following data:

Employee ID	Age	Gender	Years at Company	Job Role	Monthly
8410	31	Male		19	Education
64756	59	Female		4	Media
30257	24	Female		10	Healthcare
65791	36	Female		7	Education
65026	56	Male		41	Education
24368	38	Female		3	Technology
64970	47	Male		23	Education
36999	48	Male		16	Finance
32714	57	Male		44	Education
15944	24	Female		1	Healthcare
29972	30	Female		12	Education
9063	29	Female		6	Healthcare
21896	47	Female		38	Technology
28098	31	Male		22	Healthcare
22068	40	Female		30	Technology
17696	40	Female		4	Finance

At the bottom right of the preview area are 'Load', 'Transform Data', and 'Cancel' buttons.

## Transforming the Attrition data column from categorical to numerical:

The screenshot shows the Power Query Editor interface. A dialog box titled "Replace Values" is open, prompting the user to replace one value with another in the selected columns. The "Value To Find" field contains "Left" and the "Replace With" field contains "1". The "Attrition\_data" query is selected in the left pane, and the main preview area shows a sample of the data.

Table: Attrition\_data (8,186 rows) Column: Attrition (3 distinct values) Update available (click to download)

The screenshot shows Power BI desktop. At the top, there is a Python script editor window containing the following code:

```

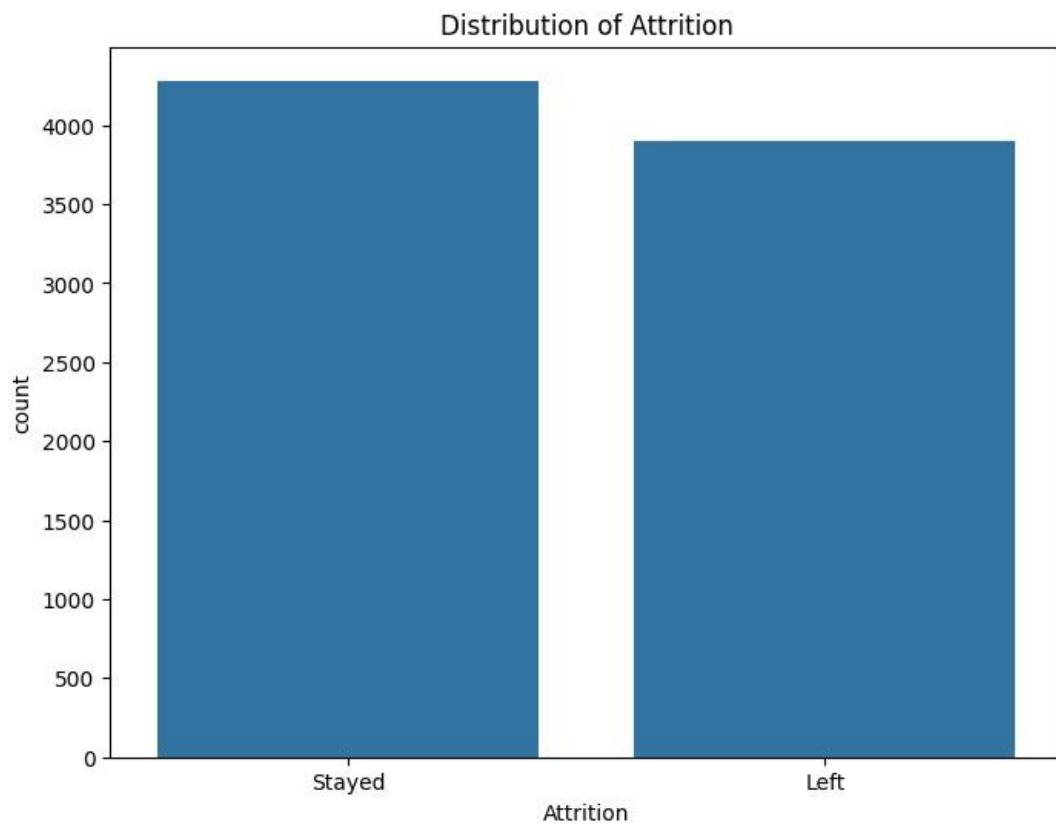
14 Attrition_data = pd.read_csv("C:/Users/Tesserai/OneDrive - Mettus/Documents/Attrition.csv")
15
16 plt.figure(figsize=(8, 6))
17 sns.countplot(x='Attrition', data=Attrition_data)
18 plt.title('Distribution of Attrition')
19 plt.show()
20

```

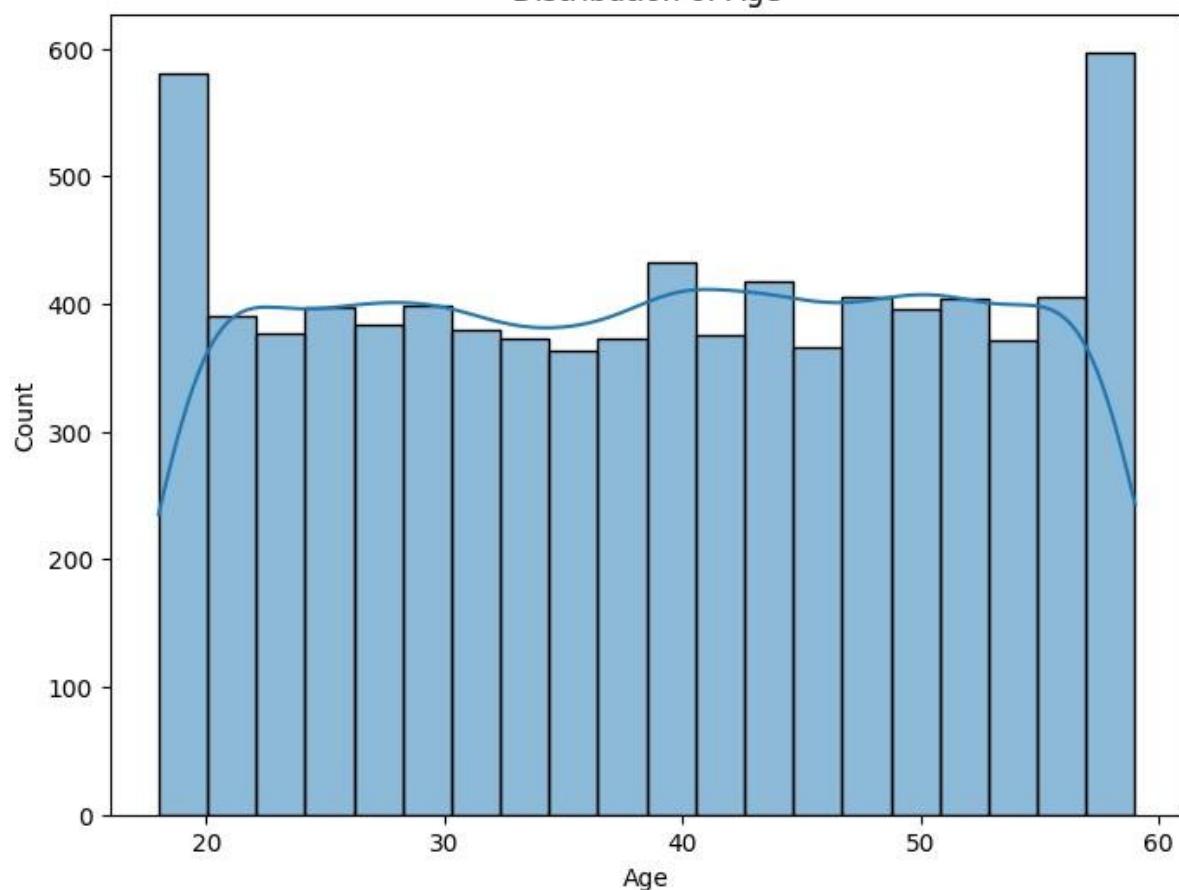
Below the script is a bar chart titled "Distribution of Attrition". The x-axis has two categories: "Stayed" and "Left". The y-axis represents the count, ranging from 0 to 4000. The "Stayed" bar is blue and reaches approximately 3800. The "Left" bar is orange and reaches approximately 3500.

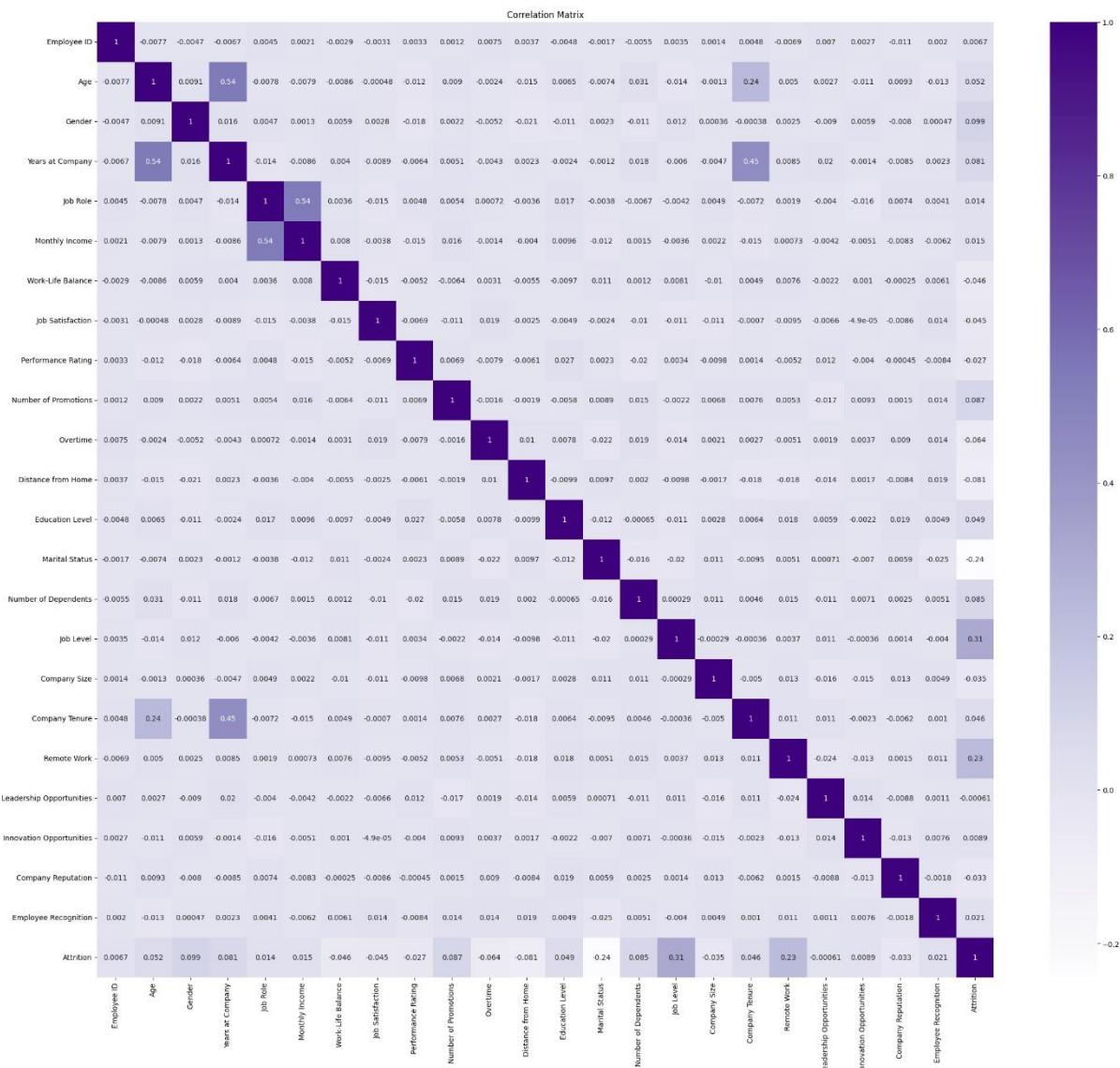
Page 1 of 1 119% Update available (click to download)

## More EDA Visuals



Distribution of Age





## Data Splitting

```
[29] #Libraries
from sklearn.model_selection import train_test_split

[30] x = data.drop('Attrition', axis=1)
y = data['Attrition']

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=42)

[31] #Shape of the train and test data
x_train.shape

(6548, 23)

[32] y_train.shape

(6548,)

[33] x_test.shape

(1637, 23)

[34] y_test.shape

(1637,)
```

## Model Training

The primary goal of the dataset is to classify employees into two categories: those who stayed and those who left the company. Logistic regression is specifically designed for binary classification tasks, making it ideal for distinguishing between these two outcomes. It estimates the probability that an employee falls into one of the two categories

```
[25] #Libraries
      from sklearn.linear_model import LogisticRegression
      from sklearn import metrics

[26] model = LogisticRegression()

[27] model.fit(x_train, y_train)
      ↗ LogisticRegression
      LogisticRegression()

[28] expected = y_train
      predicted = model.predict(x_train)
```

## Model Evaluation

```
[29] print(metrics.classification_report(expected, predicted))
      precision    recall  f1-score   support
      0       0.66     0.65     0.65     3126
      1       0.68     0.70     0.69     3422

      accuracy                           0.67
      macro avg       0.67     0.67     0.67     6548
      weighted avg    0.67     0.67     0.67     6548

[30] print(metrics.confusion_matrix(expected, predicted))
      ↗ [[2820 1196]
      [1859 2383]]
```

From the above metrics snippets, the model shows a balanced performance in classifying between the two classes. For the '0', the precision is 66% and the recall is 65%, indicating a moderate performance in identifying the '0' class. The '1' class, however, there is a better performance in detecting this class, with a precision and recall of 68% and 70%, respectively. From this, it can be observed that the model has more difficulty predicting the stayed class compared to the left class.

# Amazon Web Service

## Creating an S3 Bucket:

An Amazon S3 bucket named ‘sagemakerteebucket’ is created to store the datasets and model files used in the project. The configuration is set to general purpose, which is recommended for most use cases and access patterns. This type of bucket allows for the distribution of storage across multiple availability zones, which ensures redundancy and reliability. The bucket is created in the US East N. Virginia) region. The bucket will be used for managing the input and output data for the AWS SageMaker model training and deployment process.

On the search bar in the aws console dashboard, search for S3:

The screenshot shows the AWS console search results for 's3'. The search bar at the top contains 's3'. Below it, the sidebar shows 'Amazon Sage' and a 'Create' section with various options like 'Notebooks' and 'Marketplace'. The main search results are under 'Services' and 'Features'. Under 'Services', there are three items: 'S3' (Scalable Storage in the Cloud), 'S3 Glacier' (Archive Storage in the Cloud), and 'AWS Snow Family' (Large Scale Data Transport). Under 'Features', there is one item: 'Imports from S3' (DynamoDB feature). A 'See all 8 results' link is also present.

Create a new bucket:

The screenshot shows the AWS S3 service dashboard. The top navigation bar includes 'CloudShell', 'Feedback', and links for 'Privacy', 'Terms', and 'Cookie preferences'. The main area displays an 'Account snapshot - updated every 24 hours' with a link to 'All AWS Regions'. Below this, there are tabs for 'General purpose buckets' and 'Directory buckets', with 'General purpose buckets' selected. A table lists nine buckets: 'aws-cloudtrail-logs-339712748200-6238cd9a' (created July 19, 2024) and 'demobucketsfrancis' (created August 19, 2024). The table includes columns for Name, AWS Region, IAM Access Analyzer, and Creation date. Action buttons for 'Copy ARN', 'Empty', 'Delete', and 'Create bucket' are available. A search bar at the bottom left and a footer with copyright information are also visible.

AWS Services Search [Alt+S] N. Virginia Lwazi

### General configuration

AWS Region  
US East (N. Virginia) us-east-1

Bucket type [Info](#)

General purpose  
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

Directory - New  
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - optional  
Only the bucket settings in the following configuration are copied.  
[Choose bucket](#)

Format: s3://bucket/prefix

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Services Search [Alt+S] N. Virginia Lwazi

Successfully created bucket "sagemakerteebucket"  
To upload files and folders, or to configure additional bucket settings, choose [View details](#).

### General purpose buckets (10) [Info](#) All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
<a href="#">nontobekobucket</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 19, 2024, 09:04:28 (UTC+02:00)
<a href="#">romeodiabetebucket</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 19, 2024, 09:03:37 (UTC+02:00)
<a href="#">sagemakeremployeeattrition</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 18, 2024, 02:26:30 (UTC+02:00)
<a href="#">sagemakerteebucket</a>	US East (N. Virginia) us-east-1	<a href="#">View analyzer for us-east-1</a>	August 19, 2024, 23:21:21 (UTC+02:00)

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

## Amazon SageMaker

The following captures the process of setting up a new notebook instance in AWS SageMaker. The instance is named TeeAttrition and is configured to use the ‘ml.t3.medium’ instance type, which provides a balance of compute, memory and networking resources. This notebook instance is the environment where the model training and evaluation will be conducted.

On the search bar in the aws console dashboard, search for sagemaker:

A screenshot of the AWS Console search results for 'sagemaker'. The search bar at the top contains 'sagemaker'. Below it, the 'Services' section shows a single result: 'Amazon SageMaker' with a star icon, described as 'Build, Train, and Deploy Machine Learning Models'. Other services listed include EMR, Amazon Kinesis, S3, EC2, AWS Lambda, and CloudWatch. The right side of the screen shows a sidebar with 'Recent' items like 'CloudWatch Metrics' and 'Amazon CloudWatch Metrics Insights'.

On the side bar, in the Amazon SageMaker dashboard, navigate to notebooks under Applications and IDEs:

A screenshot of the Amazon SageMaker Domains page. The left sidebar shows 'Applications and IDEs' with 'Studio' selected. The main content area shows a message about Studio being the default landing UI. Below it, the 'Domains' section has a sub-header 'Domains Info'. It explains that a domain is an environment for your team to access SageMaker resources. A table titled 'Domains (0) Info' shows a single row: 'No domains'. A note says 'To add a domain, choose Create domain.' The bottom of the page includes standard AWS footer links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

## Create a notebook instance:

The screenshot shows the Amazon SageMaker console interface. On the left, a sidebar menu includes 'Getting started', 'Applications and IDEs' (with 'Notebooks' selected), 'Admin configurations', and links to 'CloudShell' and 'Feedback'. The main content area displays 'Notebook instances' and 'Git repositories'. A table lists existing notebook instances: 'TsehsDiabetesInstance' (ml.t2.medium, InService) and 'Diabetes' (ml.t3.medium, InService). Below this, a 'Create notebook instance' button is visible. The URL in the browser bar is 'Amazon SageMaker > Notebook instances > Create notebook instance'.

**Notebook instances**

Name	Instance	Creation time	Status	Actions
TsehsDiabetesInstance	ml.t2.medium	8/19/2024, 8:50:59 AM	InService	Open Jupyter   Open JupyterLab
Diabetes	ml.t3.medium	8/19/2024, 8:36:54 AM	InService	Open Jupyter   Open JupyterLab
		8/19/2024, 8:34:42		Open Jupyter   Open

**Create notebook instance**

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. [Learn more](#)

**Notebook instance settings**

Notebook instance name: TeeAttrition  
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type: ml.t3.medium

Platform identifier [Learn more](#): Amazon Linux 2, Jupyter Lab 3

Additional configuration

## Creating an IAM role:

The IAM role allows the SageMaker notebook instance to access the selected S3 bucket. The role is restricted to a specific bucket, which ensures that the SageMaker instance can read and write only to the designated storage location. This is important for handling input datasets and storing the model outputs.

AWS Services Search [Alt+S] N. Virginia ▾ Lwazi ▾

### Permissions and encryption

IAM role  
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the [AmazonSageMakerFullAccess](#) IAM policy attached.

AmazonSageMaker-ExecutionRole-20240819T085077

Create a new role  
Enter a custom IAM role ARN  
Use existing role

- AmazonSageMaker-ExecutionRole-20240818T022737
- AmazonSageMaker-ExecutionRole-20240819T083104
- AmazonSageMaker-ExecutionRole-20240819T083116
- AmazonSageMaker-ExecutionRole-20240819T083153
- AmazonSageMaker-ExecutionRole-20240819T083170
- AmazonSageMaker-ExecutionRole-20240819T083176
- AmazonSageMaker-ExecutionRole-20240819T083332

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

**Create an IAM role**

The IAM role you create will provide access to:

- S3 buckets you specify - *optional*
  - Any S3 bucket
 

Allow users that have access to your notebook instance access to any bucket and its contents in your account.
  - Specific S3 buckets
 

sagemakerteebucket

Comma delimited. ARNs, "\*" and "/" are not supported.
  - None
- Any S3 bucket with "sagemaker" in the name
- Any S3 object with "sagemaker" in the name
- Any S3 object with the tag "sagemaker" and value "true"
 

See Object tagging
- S3 bucket with a Bucket Policy allowing access to SageMaker
 

See S3 bucket policies

[Cancel](#) [Create role](#)

**Permissions and encryption**

IAM role  
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the [AmazonSageMakerFullAccess](#) IAM policy attached.

AmazonSageMaker-ExecutionRole-20240819T232511

**Success! You created an IAM role.**  
AmazonSageMaker-ExecutionRole-20240819T232511

Create role using the role creation wizard

Root access - *optional*

- Enable - Give users root access to the notebook
- Disable - Don't give users root access to the notebook
 

Lifecycle configurations always have root access

Encryption key - *optional*  
Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

[View details](#)

**Amazon SageMaker**

Getting started

**Applications and IDEs**

- Studio
- Canvas
- RStudio
- TensorBoard
- Profiler
- Notebooks

**Admin configurations**

- Domains
- Role manager

[View details](#)

Name	Instance	Creation time	Status	Actions
TeeAttrition	ml.t3.medium	8/19/2024, 11:25:54 PM	Pending	-
TsehsDiabetesInstance	ml.t2.medium	8/19/2024, 8:50:59 AM	InService	Open Jupyter   Open JupyterLab
Diabetes	ml.t3.medium	8/19/2024, 8:36:54 AM	InService	Open Jupyter   Open JupyterLab
EmployeeAttrition	ml.t3.medium	8/19/2024, 8:34:42 AM	InService	Open Jupyter   Open JupyterLab

The screenshot shows the Amazon SageMaker console with the 'Notebook instances' page open. The left sidebar includes sections for 'Getting started', 'Applications and IDEs' (Studio, Canvas, RStudio, TensorBoard, Profiler, Notebooks), and 'Admin configurations' (Domains, Role manager). The main area displays a table of notebook instances:

Name	Instance	Creation time	Status	Actions
TeeAttrition	mLt3.medium	8/19/2024, 11:25:54 PM	InService	Open Jupyter   Open JupyterLab
TsehsDiabetesInstance	mLt2.medium	8/19/2024, 8:50:59 AM	InService	Open Jupyter   Open JupyterLab
Diabetes	mLt3.medium	8/19/2024, 8:36:54 AM	InService	Open Jupyter   Open JupyterLab
EmployeeAttrition	mLt3.medium	8/19/2024, 8:34:42 AM	InService	Open Jupyter   Open JupyterLab
Diabeticsdeploy	mLt3.medium	8/19/2024, 8:34:38 AM	InService	Open Jupyter   Open JupyterLab

At the bottom of the page, there are links for CloudShell, Feedback, and a footer with copyright information.

Uploading the attrition dataset together with the google colab notebook to be edited:

The screenshot shows the Jupyter interface with a sidebar menu (Files, Running, Clusters, Conda, SageMaker Examples) and a main content area. The content area displays a file list:

0	/	Name	Last Modified	File size
<input type="checkbox"/>	<input type="checkbox"/> Attrition.ipynb		seconds ago	847 kB
<input type="checkbox"/>	<input type="checkbox"/> Attrition.csv		2 minutes ago	1.05 MB

At the bottom right, there is a 'Share this window' button.

Continuing from the data splitting portion of the project. The x and y, test and train data is combined, respectively, for better display.

## Feature Engineering

```
In [149]: #lets convert this attrition to label
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['Attrition']= le.fit_transform(data['Attrition'])
data.head(20)
```

```
In [150]: stayed = data.Attrition == 0
left = data.Attrition ==1
```

```
In [151]: le.classes_
```

```
Out[151]: array([0, 1])
```

```
In [152]: Attrition_Employee = le.classes_
print(Attrition_Employee)
```

```
[0 1]
```

```
In [154]: #Define the columns to be label encoded
labels_cols = ['Gender','Job Role','Overtime','Education Level','Marital Status','Company Size','Remote
'Leadership Opportunities', 'Innovation Opportunities','Work-Life Balance', 'Job Satisfaction',
'Company Reputation','Job Level', 'Employee Recognition']

#initialize Label encoders
label_encoders = {col: LabelEncoder() for col in labels_cols}

#Apply the Label Encoding
for col in labels_cols:
    data[col] = label_encoders[col].fit_transform(data[col])
```

Share this window

```
In [159]: data.head(10)
```

```
Out[159]:
```

	Employee ID	Age	Gender	Years at Company	Job Role	Monthly Income	Work-Life Balance	Job Satisfaction	Performance Rating	Number of Promotions	Overtime	Distance from Home	Ed
0	8410	31	1	19	0	5390	0	2	0	2	0	0	22
1	64756	59	0	4	3	5534	3	0	3	3	0	0	21
2	30257	24	0	10	2	8159	2	0	3	0	0	0	11
3	65791	36	0	7	0	3989	2	0	2	1	0	0	27
4	65026	56	1	41	0	4821	1	3	0	0	1	1	71
5	24368	38	0	3	4	9977	1	0	1	3	0	0	37
6	64970	47	1	23	0	3681	1	0	2	1	1	1	75
7	36999	48	1	16	1	11223	0	3	2	2	0	0	5
8	32714	57	1	44	0	3773	2	2	2	1	1	1	39

```
In [160]: x=data.iloc[:, :-1]
#remove the last column Attrition
```

```
In [161]: x.head()
```

```
Out[161]:
```

	Employee ID	Age	Gender	Years at Company	Job Role	Monthly Income	Work-Life Balance	Job Satisfaction	Performance Rating	Number of Promotions	Overtime	Distance from Home	Ed
0	8410	31	1	19	0	5390	0	2	0	2	0	0	22
1	64756	59	0	4	3	5534	3	0	3	3	0	0	21
2	30257	24	0	10	2	8159	2	0	3	0	0	0	11
3	65791	36	0	7	0	3989	2	0	2	1	0	0	27
4	65026	56	1	41	0	4821	1	3	0	0	1	1	71

```
In [162]: y=data.iloc[:, -1]
```

```
In [163]: y.head()
```

```
Out[163]: 0    1
1    1
2    1
3    1
4    1
Name: Attrition, dtype: int64
```

## Data Splitting

### Data Splitting

```
In [23]: #Libraries
from sklearn.model_selection import train_test_split

In [24]: x = data.drop(columns = ['Employee ID', 'Attrition'], axis=1)
y = data['Attrition']

x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=42)
```

## Saving the Files:

The screenshot shows a Jupyter Notebook interface with three code cells and their corresponding outputs.

**In [29]:** `#Displaying the training data  
TrainData = x_train.join(y_train)  
TrainData.head()`

**Out[29]:**

	Age	Gender	Years at Company	Job Role	Monthly Income	Work-Life Balance	Job Satisfaction	Job Performance Rating	Number of Promotions	Overtime	Distance from Home	Education Level
2926	50	1	38	4	7026	2	1	0	1	0	3	3
3781	23	1	6	0	5002	0	0	0	2	1	14	1
4498	47	1	26	2	8037	2	3	0	0	0	35	1
4141	50	1	23	2	7252	2	2	3	0	0	96	2
5245	49	1	28	4	11314	1	0	0	0	0	14	1

**In [30]:** `#Display all the test data  
TestData=x_test.join(y_test)  
TestData.head()`

**Out[30]:**

	Age	Gender	Years at Company	Job Role	Monthly Income	Work-Life Balance	Job Satisfaction	Job Performance Rating	Number of Promotions	Overtime	Distance from Home	Education Level
5065	28	0	12	0	4638	1	2	0	0	0	61	1
5871	18	0	2	1	7769	1	3	0	0	0	39	1
3011	21	1	3	2	8030	3	2	1	2	0	71	4
3065	21	0	1	0	4268	2	2	0	0	1	54	0
1320	55	1	16	2	7527	2	1	0	1	0	10	3

**In [37]:** `# Saving the trained data  
TrainData.to_csv('TrainData.csv', index=False, index_label='Row', header=False)`

**In [38]:** `#Saving the test data  
TestData.to_csv('TestData.csv', index=False, index_label='Row', header=False)`

[Files](#) [Running](#) [Clusters](#) [Conda](#) [SageMaker Examples](#)

Select items to perform actions on them.

[Upload](#) [New](#) [↻](#)

		Name	Last Modified	File size
<input type="checkbox"/>	0	/		
<input type="checkbox"/>	 Attrition.ipynb		Running a minute ago	910 kB
<input type="checkbox"/>	 Attrition.csv		23 minutes ago	1.05 MB
<input type="checkbox"/>	 TestData.csv		seconds ago	86.2 kB
<input type="checkbox"/>	 TrainData.csv		a minute ago	345 kB

The saved data can be seen on the notebook home list.

From there, the boto3 library is used to upload the files to the S3 bucket. The paths to the training and testing datasets are defined, their S3 paths are constructed, then they are uploaded to the designated S3 bucket locations.

```
In [38]: #Saving the test data
TestData.to_csv('TestData.csv', index=False, index_label='Row', header=False)

In [ ]: #Uploading the trained and test data to Amazon S3 bucket

In [39]: #Necessary packages
import boto3
import re

In [40]: bucketName = 'sagemakerteebucket'
TrainFile = r'EmployeeAttritionData/TrainData/TrainData.csv'
TestFile = r'EmployeeAttritionData/TestData/TestData.csv'
ValFile = r'EmployeeAttritionData/ValData/Val.csv'
ModelFolder = r'EmployeeAttrition/model/'

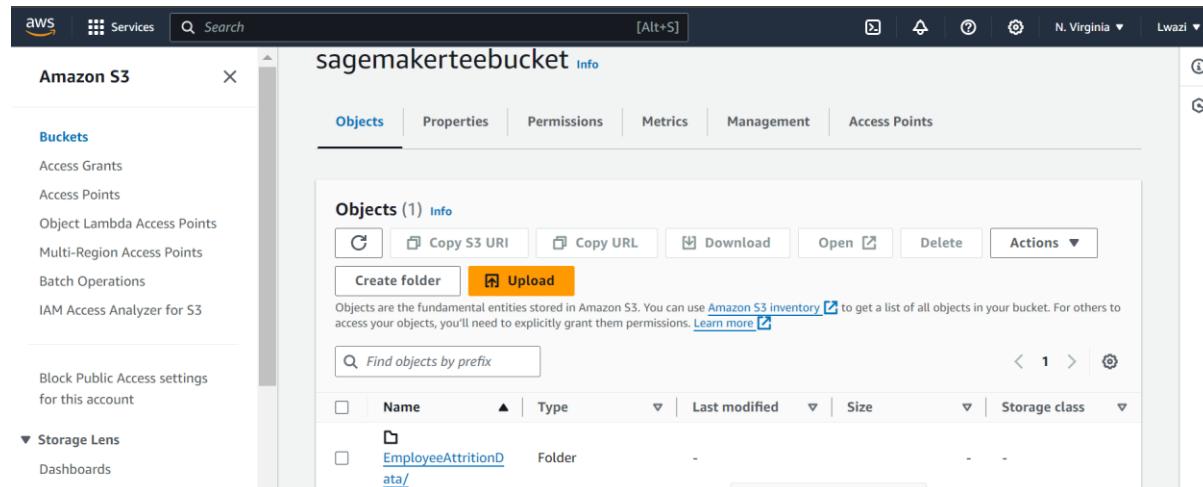
In [41]: s3ModelOutput = r's3://{{0}}/{{1}}'.format(bucketName,ModelFolder)
s3Train = r's3://{{0}}/{{1}}'.format(bucketName,TrainFile)
s3Test = r's3://{{0}}/{{1}}'.format(bucketName,TestFile)
s3Val = r's3://{{0}}/{{1}}'.format(bucketName,ValFile)

In [42]: s3ModelOutput
Out[42]: 's3://sagemakerteebucket/EmployeeAttrition/model/'

In [43]: with open('TrainData.csv', 'rb') as f:
    boto3.Session().resource('s3').Bucket(bucketName).Object(TrainFile).upload_fileobj(f)

In [44]: with open('TestData.csv', 'rb') as f:
    boto3.Session().resource('s3').Bucket(bucketName).Object(TestFile).upload_fileobj(f)
```

From the following snippets, it can be observed that the folders and datasets, have indeed been uploaded on the specified S3 bucket.



The screenshot shows the AWS S3 console interface. The left sidebar has 'Amazon S3' selected under 'Services'. The main area shows the 'sagemakerteebucket' bucket. The 'Objects' tab is active, showing one object named 'EmployeeAttritionD ata/'. Below the object list are buttons for 'Create folder' and 'Upload'. A note says 'Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions.' At the bottom, there's a table with columns: Name, Type, Last modified, Size, and Storage class.

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	EmployeeAttritionD ata/	Folder	-	-	-

aws Services Search [Alt+S] N. Virginia Lwazi

### Amazon S3 EmployeeAttritionData/

Buckets Access Grants Access Points Object Lambda Access Points Multi-Region Access Points Batch Operations IAM Access Analyzer for S3

Block Public Access settings for this account

Storage Lens Dashboards Storage Lens groups

Objects (2) info

Copy S3 URI Copy URL Download Open Delete Actions

Create folder Upload

Find objects by prefix

Name	Type	Last modified	Size	Storage class
TestData/	Folder	-	-	-
TrainData/	Folder	-	-	-

Amazon S3 TestData/

Buckets Access Grants Access Points Object Lambda Access Points Multi-Region Access Points Batch Operations IAM Access Analyzer for S3

Block Public Access settings for this account

Storage Lens Dashboards Storage Lens groups

Objects (1) info

Copy S3 URI Copy URL Download Open Delete Actions

Create folder Upload

Find objects by prefix

Name	Type	Last modified	Size	Storage class
TestData.csv	csv	August 20, 2024, 00:07:37 (UTC+02:00)	84.2 KB	Standard

Amazon S3 TrainData/

Buckets Access Grants Access Points Object Lambda Access Points Multi-Region Access Points Batch Operations IAM Access Analyzer for S3

Block Public Access settings for this account

Storage Lens Dashboards Storage Lens groups

Objects (1) info

Copy S3 URI Copy URL Download Open Delete Actions

Create folder Upload

Find objects by prefix

Name	Type	Last modified	Size	Storage class
TrainData.csv	csv	August 20, 2024, 00:07:36 (UTC+02:00)	336.9 KB	Standard

## Deploying:

```
In [83]: import sagemaker
from sagemaker import get_execution_role

sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/con-
fig.yaml
```

```
In [84]: sagemakerSess=sagemaker.Session()
role=get_execution_role()
```

```
In [85]: sagemakerSess.boto_region_name
```

```
Out[85]: 'us-east-1'
```

```
In [86]: gemaker.amazon.amazon_estimator.get_image_uri(sagemakerSess.boto_region_name,'linear-learner','latest')
        ↪
The method get_image_uri has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
Defaulting to the only supported framework/algorithm version: 1. Ignoring framework/algorithm versio-
n: latest.
```

```
In [87]: LogisticModel=sagemaker.estimator.Estimator(image_uri=ECRdockercontainer,
                                                    role=role,
                                                    train_instance_count=1,
                                                    train_instance_type='ml.m4.xlarge',
                                                    output_path=s3ModelOutput,
                                                    sagemaker_session=sagemakerSess,
                                                    base_job_name = 'Logistic-Demo-v1'
                                                    )
```

```
train_instance_count has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
train_instance_type has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```

```
In [56]: LogisticModel.set_hyperparameters(predictor_type='binary_classifier', mini_batch_size=100)
```

```
In [57]: LogisticModel.hyperparameters()
```

```
Out[57]: {'predictor_type': 'binary_classifier', 'mini_batch_size': 100}
```

```
In [58]: trainConfig=sagemaker.session.s3_input(s3_data=s3Train,content_type='text/csv')
```

```
The class sagemaker.session.s3_input has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```

```
In [58]: trainConfig=sagemaker.session.s3_input(s3_data=s3Train,content_type='text/csv')
The class sagemaker.session.s3_input has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.

In [59]: LogisticModel.fit({'train':trainConfig})
INFO:sagemaker:Creating training-job with name: Logistic-Demo-v1-2024-08-19-17-08-36-616
2024-08-19 17:08:36 Starting - Starting the training job...
2024-08-19 17:08:54 Starting - Preparing the instances for training...
2024-08-19 17:09:22 Downloading - Downloading input data...
2024-08-19 17:09:52 Downloading - Downloading the training image.....
2024-08-19 17:11:03 Training - Training image download completed. Training in progress....Docker en
trypoint called with argument(s): train
    r : 0.99, lr_scheduler_minimum_lr : 1e-05
[08/19/2024 17:12:34 INFO 140294227871552] Saved checkpoint to "/tmp/tmpeislakal/mx-mod-0000.param
s"
[08/19/2024 17:12:34 INFO 140294227871552] Test data is not provided.
#metrics {"StartTime": 1724087487.5590994, "EndTime": 1724087554.5685575, "Dimensions": {"Algorithm": "Linear Learner", "Host": "algo-1", "Operation": "training"}, "Metrics": {"initialize.time": {"sum": 614.1691207885742, "count": 1, "min": 614.1691207885742, "max": 614.1691207885742}, "epoch.time": {"sum": 15.0, "count": 1, "min": 15, "max": 15}, "check_early_stopping.time": {"sum": 2.2020339965820312, "count": 6, "min": 0.1728534698486328, "max": 0.7431507110595703}, "update.time": {"sum": 62414.03818130493, "count": 6, "min": 10283.617334365845, "max": 18773.866415023884}, "finalize.time": {"sum": 3954.5650482177734, "count": 1, "min": 3954.5650482177734, "max": 3954.5650482177734}, "setup.time": {"sum": 1.9104480743488283, "count": 1, "min": 1.9104480743488283, "max": 1.9104480743488283}, "totaltime": {"sum": 67109.06028747559, "count": 1, "min": 67109.06028747559, "max": 67109.06028747559}}}

2024-08-19 17:12:52 Uploading - Uploading generated training model
2024-08-19 17:12:52 Completed - Training job completed
Training seconds: 211
Billable seconds: 211

In [60]: #Deploying the Trained Model
predictmodel=LogisticModel.deploy(initial_instance_count=1, instance_type='ml.m4.xlarge',
                                    endpoint_name = 'LogisticRegression-demo-v1')
INFO:sagemaker:Creating model with name: Logistic-Demo-V1-2024-08-19-17-25-43-398
INFO:sagemaker:Creating endpoint-config with name LogisticRegression-demo-v1
INFO:sagemaker:Creating endpoint with name LogisticRegression-demo-v1
-----!
```

## Endpoints:

The image consists of three vertically stacked screenshots of the AWS SageMaker console, each with a red circle and checkmark highlighting specific sections.

**Screenshot 1: Models**

The left sidebar shows the navigation path: Ground Truth > Processing > Training > Inference > Models. The "Models" section is highlighted with a red circle and a checkmark. The main content area shows a table titled "Models" with one item:

Name	ARN	Creation time
Logistic-Demo-v1-2024-08-19-17-25-43-398	arn:aws:sagemaker:us-east-1:339712748200:model/Logistic-Demo-v1-2024-08-19-17-25-43-398	8/19/2024, 7:25:44 PM

**Screenshot 2: Endpoint configuration**

The left sidebar shows the navigation path: Processing > Training > Inference > Models > Endpoint configurations. The "Endpoint configurations" section is highlighted with a red circle and a checkmark. The main content area shows a table titled "Endpoint configuration" with one item:

Name	ARN	Creation time
LogisticRegression-demo-v1	arn:aws:sagemaker:us-east-1:339712748200:endpoint-config/LogisticRegression-demo-v1	8/19/2024, 7:25:44 PM

**Screenshot 3: Endpoints**

The left sidebar shows the navigation path: Ground Truth > Processing > Training > Inference > Models > Endpoint configurations > Endpoints. The "Endpoints" section is highlighted with a red circle and a checkmark. The main content area shows a table titled "Endpoints" with one item:

Name	ARN	Creation time	Status	Last updated
LogisticRegression-demo-v1	arn:aws:sagemaker:us-east-1:339712748200:endpoint/LogisticRegression-demo-v1	8/19/2024, 7:25:44 PM	InService	8/19/2024, 7:30:18 PM

Amazon SageMaker > Endpoints > LogisticRegression-demo-v1

## LogisticRegression-demo-v1

[Delete](#)

Endpoint summary		
Name LogisticRegression-demo-v1	Status <span style="color: green;">⌚ InService</span>	Type Real-time
ARN arn:aws:sagemaker:us-east-1:339712748200:endpoint/LogisticRegression-demo-v1	Creation time Mon Aug 19 2024 19:25:44 GMT+0200 (South Africa Standard Time)	Last updated Mon Aug 19 2024 19:30:18 GMT+0200 (South Africa Standard Time)
URL <a href="https://runtime.sagemaker.us-east-1.amazonaws.com/endpoints/LogisticRegression-demo-v1/invocations">https://runtime.sagemaker.us-east-1.amazonaws.com/endpoints/LogisticRegression-demo-v1/invocations</a>	Model container logs <a href="#">/aws/sagemaker/endpoints/LogisticRegression-demo-v1/invocations</a>	Alarms 0 alarms
<a href="#">Learn more about the API</a>		

[Open SageMaker Domain](#)

## Dashboard

Recent activity

Recent activity within the last 7 days					
Ground Truth	Notebook	Training	Inference	Processing	Canvas
Labeling jobs	Notebook instances	Training jobs	Models	Jobs	Endpoints
No recent activity.	<span style="background-color: green; color: white; padding: 2px 5px;">⌚ 7 In Service</span>	<span style="background-color: grey; color: red; padding: 2px 5px;">⌚ 1 Completed</span>	<span style="background-color: grey; color: red; padding: 2px 5px;">⌚ 1 Created</span>	No recent activity.	No recent activity.
	<span style="background-color: grey; color: red; padding: 2px 5px;">⌚ 7 Created</span>	<span style="background-color: grey; color: red; padding: 2px 5px;">⌚ 1 Created</span>			
		Hyperparameter tuning jobs			
		No recent activity.			
			Endpoints	<span style="background-color: green; color: white; padding: 2px 5px;">⌚ 1 In Service</span>	
				<span style="background-color: grey; color: red; padding: 2px 5px;">⌚ 1 Created</span>	
			Batch transform jobs		
			No recent activity.		

Recent activity within the last 7 days

Amazon SageMaker > Endpoints > LogisticRegression-demo-v1

## LogisticRegression-demo-v1

[Delete](#)

Endpoint summary		
Name LogisticRegression-demo-v1	Status <span style="color: green;">⌚ InService</span>	Type Real-time
ARN arn:aws:sagemaker:us-east-1:339712748200:endpoint/LogisticRegression-demo-v1	Creation time Mon Aug 19 2024 19:25:44 GMT+0200 (South Africa Standard Time)	Last updated Mon Aug 19 2024 19:30:18 GMT+0200 (South Africa Standard Time)
URL <a href="https://runtime.sagemaker.us-east-1.amazonaws.com/endpoints/LogisticRegression-demo-v1/invocations">https://runtime.sagemaker.us-east-1.amazonaws.com/endpoints/LogisticRegression-demo-v1/invocations</a>	Model container logs <a href="#">/aws/sagemaker/endpoints/LogisticRegression-demo-v1/invocations</a>	Alarms 0 alarms
<a href="#">Learn more about the API</a>		

## Monitoring

