ASSIGNMENT 03

Amazon Academy Challenge Lab 03

Machine Learning Pipeline Report
ITAI 1371: Intro to Machine Learning
Houston Community College
Prof. Machuria Johnson
Nov. 3, 2024

Andrew Badzioch

Introduction:

Machine Learning Pipelines are foundational to the success and scalability to ML applications and are designed to automate the process of how models are developed, deployed, and maintained. By standardizing tasks such as data preprocessing, feature engineering, model training, validation, and deployment, ML pipelines enable more reliable and repeatable results, which is important in production environments where models need to be updated frequently and managed at scale.
Amazon SageMaker Pipelines is a fully managed service provided by AWS, designed to simplify the creation, management, and scaling of ML pipelines. SageMaker Pipelines allows users to build, automate, and monitor ML workflows in a secure, scalable environment, making it easier for teams to bring models from concept to production. Through its integration with other AWS services such as Amazon S3 for data storage, AWS Lambda for automation, and AWS Identity and Access Management (IAM) for security, SageMaker becomes an efficient and accessible option for companies looking to simplify ML workflows.

Efficiency:

One of the core advantages of ML pipelines is the ability to automate repetitive tasks throughout the machine learning workflow. Tasks like data preprocessing, model training, and evaluation can be easily automated, minimizing manual intervention. By handling these steps consistently, pipelines reduce the risk of human error and ensure that each stage in the process is executed according to predefined configurations. This can be beneficial in environments with strict accuracy requirements, as it allows models to maintain a steady workflow, ensuring reliable results every time.

Scalability:

As data and model complexity grow, ML pipelines play a key role in managing this scale by leveraging computer resources, ensuring that large datasets and complex models con be handled efficiently. SageMaker allocates the necessary resources depending on the requirements for the task. This allows teams to manage large data volumes without compromising speed, adapting to the demands of the workload, maximizing performance while minimizing costs.

Reproducibility:

ML pipelines facilitate version control for data and models, enabling teams to keep track of each modification and iteration throughout the lifecycle. Teams can manage and trace every version of a dataset, model configuration, and parameter setting, providing a clear line for each result. This is beneficial as it allows identification of specific versions that produced desired results, diagnose issues, and revert to previous versions when needed.

This is essential for maintaining a stable workflow and ensures models can be redefined without compromising the accuracy or reliability of previous results.

Business Impact:

ML pipelines also contribute to cost efficiency by optimizing resources across the entire ML lifecycle. This optimization not only lowers costs but also allows businesses to handle high workloads or sudden spikes in demand without overcommitting to a fixed infrastructure. With SageMaker, companies can schedule batch jobs to run during off-peak hours, which often come at a lower. Cost than running them during peak times. By automating and fine-tuning resources, these pipelines contribute to a more predictable cost structure and enable better budgeting and forecasting for ML projects.

Sample Basic ML Workflow:
Data Preprocessing and Visualization:

```
df_temp =
pd.read_csv(f"{csv_base_path}On_Time_Reporting_Carrier_On_Time_Performance_(1987_prese
nt)_2018_9.csv")
```

input:

```
print("The #rows and #columns are ", data.shape[0], " and ", data.shape[1])
print("The years in this dataset are: ", list(data.Year.unique()))
print("The months covered in this dataset are: ", sorted(list(data.Month.unique())))
print("The date range for data is :" , min(data.FlightDate.unique()), " to ",
max(data.FlightDate.unique()))
print("The airlines covered in this dataset are: ",
list(data.Reporting_Airline.unique()))
print("The Origin airports covered are: ", list(data.Origin.unique()))
print("The Destination airports covered are: ", list(data.Dest.unique()))
```

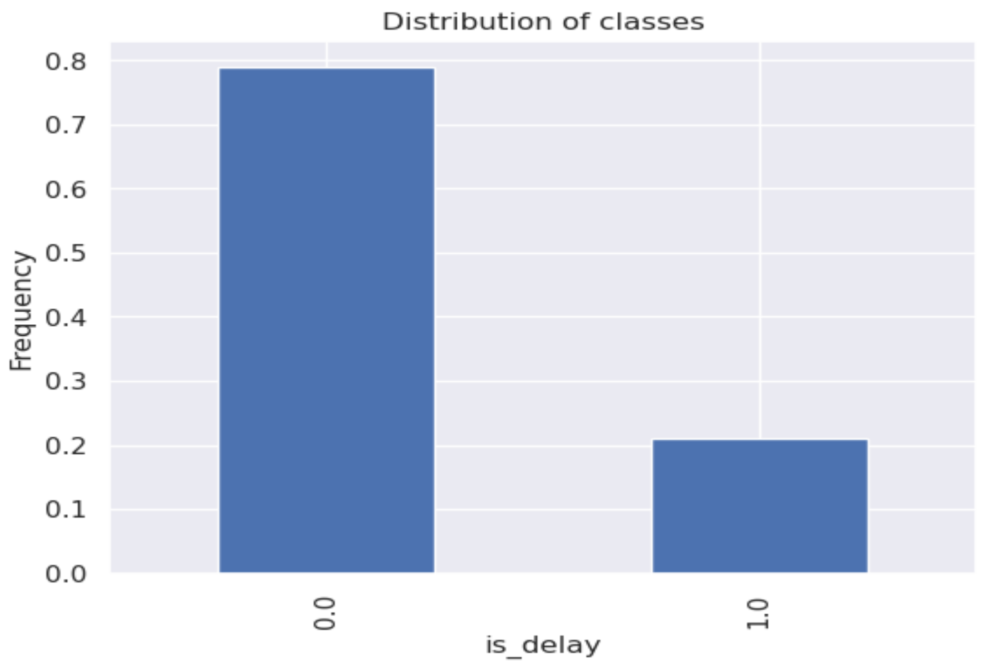
output:

```
The #rows and #columns are  1658130  and  20
The years in this dataset are:  [2015, 2018, 2017, 2014, 2016]
The months covered in this dataset are:  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
The date range for data is : 2014-01-01  to  2018-12-31
The airlines covered in this dataset are:  ['AA', 'DL', 'UA', 'WN', 'OO']
The Origin airports covered are:  ['CLT', 'DFW', 'ORD', 'LAX', 'SFO', 'PHX', 'IAH',
'DEN', 'ATL']
The Destination airports covered are:  ['DFW', 'ORD', 'ATL', 'PHX', 'SFO', 'LAX',
'IAH', 'DEN', 'CLT']
```

Input:

```
(data.groupby('is_delay').size()/len(data) ).plot(kind='bar')
plt.ylabel('Frequency')
plt.title('Distribution of classes')
plt.show()
```

output:



Model Training and Evaluation:

input:

```
from sklearn.model_selection import train_test_split
def split_data(data):
    train, test_and_validate = train_test_split(data, test_size=0.2, random_state=42,
stratify=data['target'])
    test, validate = train_test_split(test_and_validate, test_size=0.5,
random_state=42, stratify=test_and_validate['target'])
    return train, validate, test
```

```
train, validate, test = split_data(data)
print(train['target'].value_counts())
print(test['target'].value_counts())
print(validate['target'].value_counts())
```

output:

```
0.0    1033806
1.0     274666
Name: target, dtype: int64
0.0    129226
1.0     34333
Name: target, dtype: int64
0.0    129226
1.0     34333
Name: target, dtype: int64
```

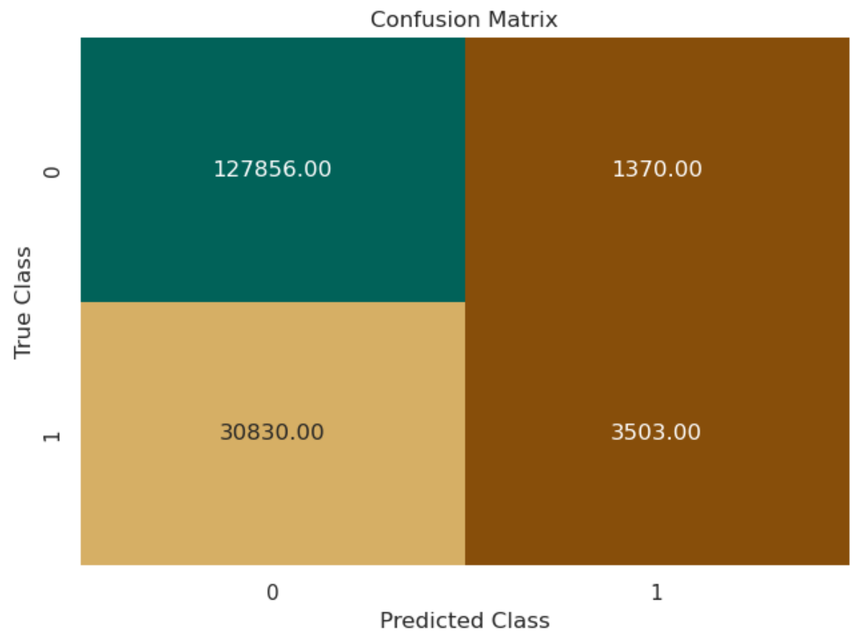
Using the XGBoost Model:

input:
```
bucket='c134412a3409745l8003998t1w11504374297-flightbucket-zyuwb8c4zacf'
prefix='flight-xgb'
train_file='flight_train.csv'
test_file='flight_test.csv'
validate_file='flight_validate.csv'
whole_file='flight.csv'
s3_resource = boto3.Session().resource('s3')

def upload_s3_csv(filename, folder, dataframe):
    csv_buffer = io.StringIO()
    dataframe.to_csv(csv_buffer, header=False, index=False )
    s3_resource.Bucket(bucket).Object(os.path.join(prefix, folder,
filename)).put(Body=csv_buffer.getvalue())

upload_s3_csv(train_file, 'train', train)
upload_s3_csv(test_file, 'test', test)
upload_s3_csv(validate_file, 'validate', validate)
```

output:
```
INFO:botocore.credentials:Found credentials from IAM Role:
BaseNotebookInstanceEc2InstanceRole
```

input:
```
plot_confusion_matrix(test_labels, target_predicted)
```

output:



Conclusion:

Machine Learning pipelines are integral to developing reliable, scalable, and efficient ML and AI applications. They enable data science teams to work with higher consistency, lower operational costs, and deliver models that can be trusted and updated as new data becomes available. As the demand for AI continues to rise, the importance of these pipelines will grow, paving the way for rapid innovation and operational excellence in ML powered organizations.

References:

AWS Academy:

https://docs.aws.amazon.com/sagemaker/