Lab Details | loan_cleaned_data/loan_

https://loan-prediction-notebook-m90d.notebook.us-east-1.sagemaker.aws/notebooks/Question.ipynb#

jupyter **Question** Last Checkpoint: 14 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run | Markdown | nbdiff

In [3]:
```
#### Import the dataset from S3
import boto3
import numpy as np
import pandas as pd
from sklearn import datasets
import sagemaker
from sagemaker import get_execution_role
role=get_execution_role()
bucket= "loan-data525763749793711"
folder_name = "loan_cleaned_data"
data_key = "loan_cleaned_data.csv"
data_location = "s3://loan-data525763749793711/loan_cleaned_data/loan_cleaned_data.csv"
```

In [5]:
```
##### Load the dataset

data = pd.read_csv(data_location)
data.head()
```

Out[5]:

	sl_no	credit_policy	purpose	int_rate	installment	log_annual_inc	dti	fico	days_with_cr_line	revol_bal	revol
0	1	1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	5639.958333	28854	
1	2	1	credit_card	0.1071	228.22	11.082143	14.29	707	2760.000000	33623	

Search

loan_cleaned_data/loan_ × Home

https://loan-prediction-notebook-m90d.notebook.us-east-1.sagemaker.aws/notebooks/Question.ipynb#

jupyter Question Last Checkpoint: 17 minutes ago (unsaved changes)

Logout

Edit View Insert Cell Kernel Widgets Help

Trusted conda_python3

Run Markdown nbdiff

In [ ]:

In [ ]:

In [6]:
```
#### Store the updated dataframe below

data = pd.get_dummies(data, columns=['purpose'],dtype=int)
data.head()
```

Out[6]:

	sl_no	credit_policy	int_rate	installment	log_annual_inc	dti	fico	days_with_cr_line	revol_bal	revol_util	...	delinq_2yrs	pub_rec	not_fully_paid	purpc
0	1	1	0.1189	829.10	11.350407	19.48	737	5639.958333	28854	52.1	...	0	0	0	
1	2	1	0.1071	228.22	11.082143	14.29	707	2760.000000	33623	76.7	...	0	0	0	
2	3	1	0.1357	366.86	10.373491	11.63	682	4710.000000	3511	25.6	...	0	0	0	
3	4	1	0.1008	162.34	[illegible]350407	8.10	712	2699.958333	33667	73.2	...	0	0	0	
4	5	1	0.1426	102.92	[illegible]99732	14.97	667	4066.000000	4740	39.5	...	1	0	0	

5 rows × 21 columns

Task III - Data Preprocessing

Search

DELL

jupyter Question Last Checkpoint: 23 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Code nbdiff

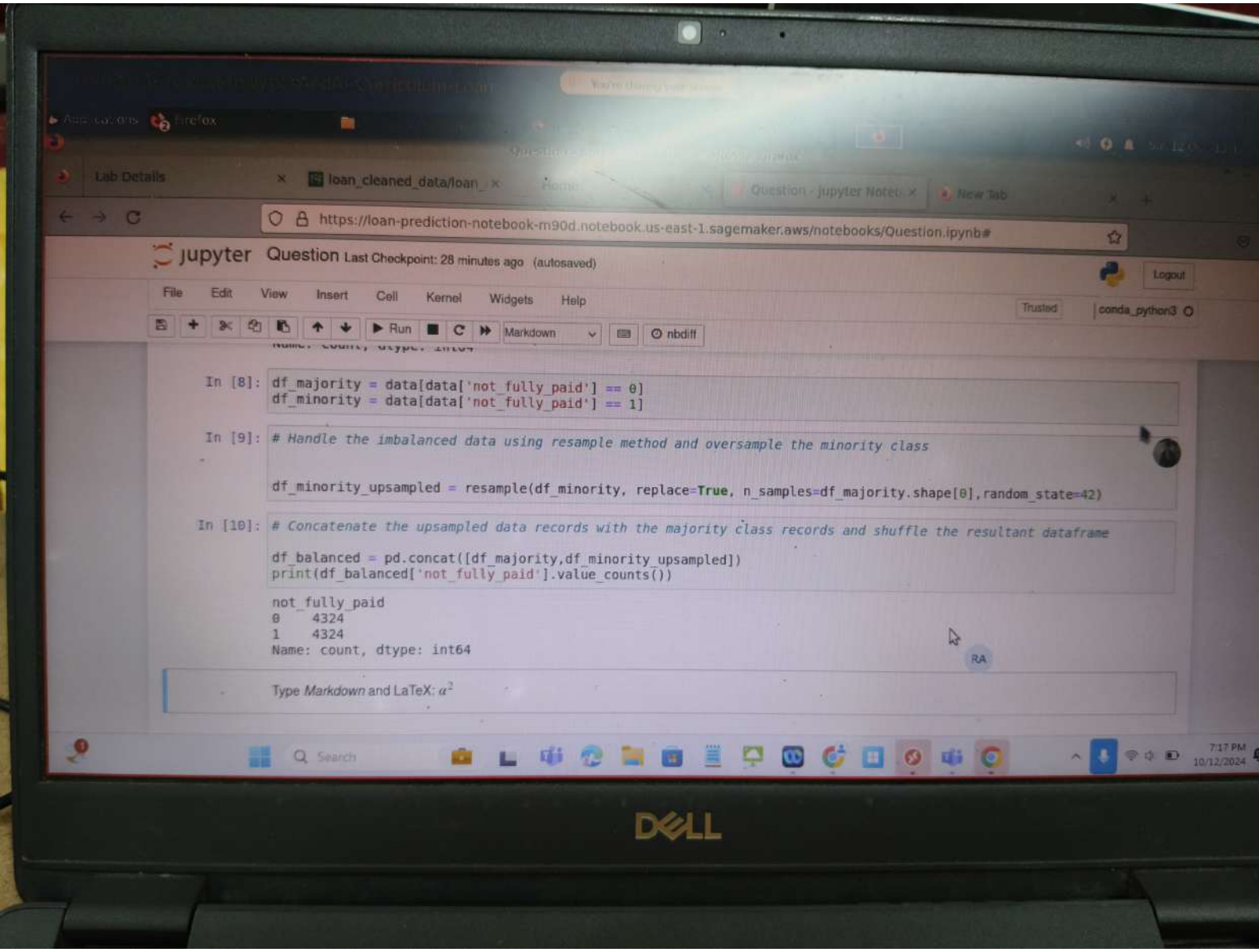
In [7]:
```
from sklearn.utils import resample
'not_fully_paid'
print(data['not_fully_paid'].value_counts())
```

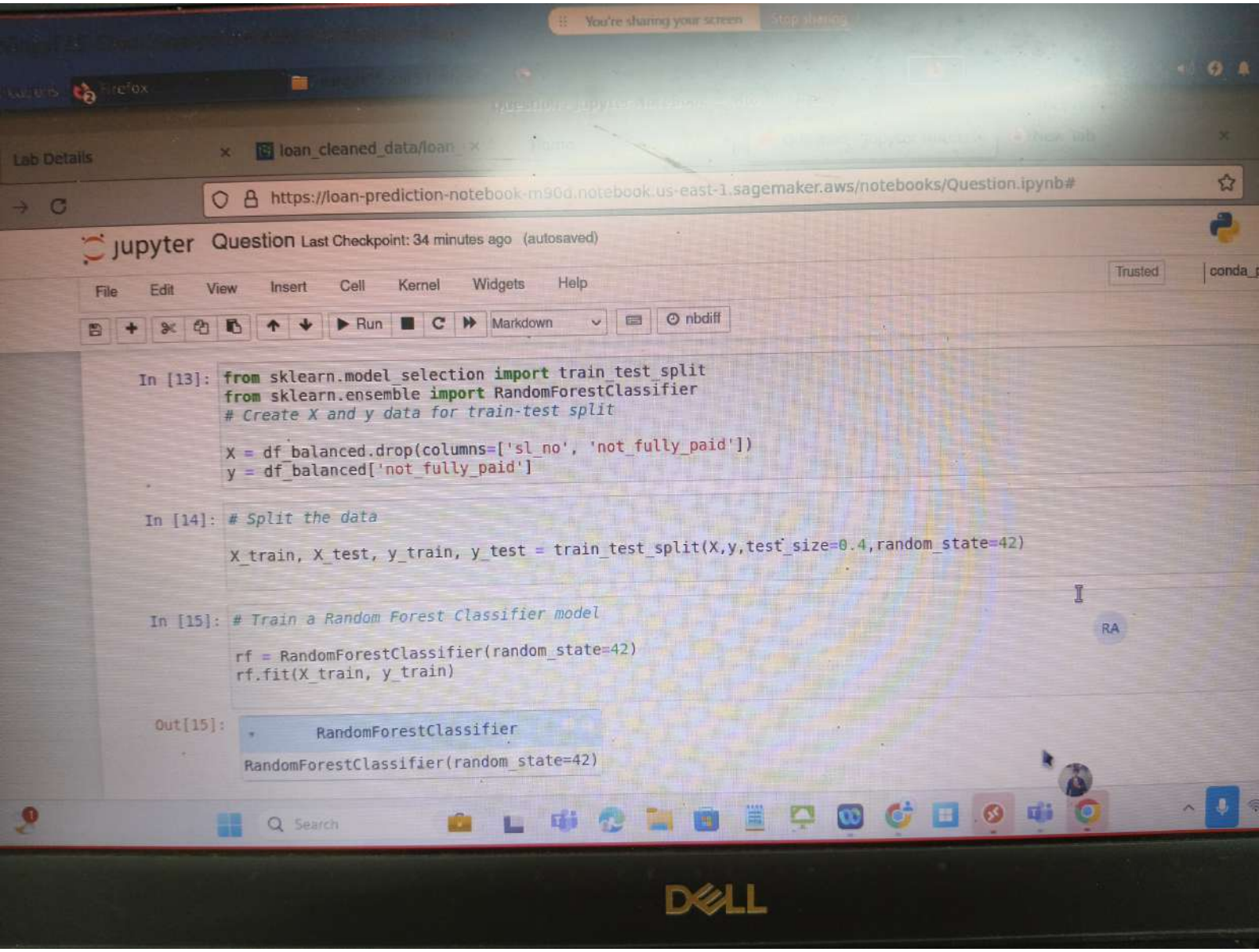
```
not_fully_paid
0    4324
1     683
Name: count, dtype: int64
```

In [8]:
```
df_majority = data[data['not_fully_paid'] == 0]
df_minority = data[data['not_fully_paid'] == 1]
```

In [ ]:
```
# Handle the imbalanced data using resample method and oversample the min

df_minority_upsampled = None
```

In [ ]:
```
# Concatenate the upsampled data records with the majority class records a

df = None
```


Lab Details
loan_cleaned_data/loan_
https://loan-prediction-notebook-m90d.notebook.us-east-1.sagemaker.aws/notebooks/Question.ipynb#
jupyter Question Last Checkpoint: 28 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
conda_python3
Run
Markdown
nbdiff
In [8]:
df_majority = data[data['not_fully_paid'] == 0]
df_minority = data[data['not_fully_paid'] == 1]
In [9]:
# Handle the imbalanced data using resample method and oversample the minority class
df_minority_upsampled = resample(df_minority, replace=True, n_samples=df_majority.shape[0],random_state=42)
In [10]:
# Concatenate the upsampled data records with the majority class records and shuffle the resultant dataframe
df_balanced = pd.concat([df_majority,df_minority_upsampled])
print(df_balanced['not_fully_paid'].value_counts())
not_fully_paid
0 4324
1 4324
Name: count, dtype: int64
Type Markdown and LaTeX: α^2
RA
Search
7:17 PM
10/12/2024
DELL


You're sharing your screen
Stop sharing
Lab Details
loan_cleaned_data/loan
https://loan-prediction-notebook-m90d.notebook.us-east-1.sagemaker.aws/notebooks/Question.ipynb#
jupyter Question Last Checkpoint: 34 minutes ago (autosaved)
Trusted
File Edit View Insert Cell Kernel Widgets Help
Run
Markdown
nbdiff
In [13]:
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
# Create X and y data for train-test split
X = df_balanced.drop(columns=['sl_no', 'not_fully_paid'])
y = df_balanced['not_fully_paid']
In [14]:
Split the data
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.4,random_state=42)
In [15]:
# Train a Random Forest Classifier model
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
Out[15]:
RandomForestClassifier
RandomForestClassifier(random_state=42)
RA
Search

https://loan-prediction-notebook-m90d.notebook.us-east-1.sagemaker.aws/notebooks/Question.ip

jupyter Question Last Checkpoint: 37 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Run Markdown nbdiff

- Predict using the trained **Random Forest Classifier** model *rf* on the test data *X_test*.
- Evaluate the predictions by comparing it with the actual test data *y_test*.
- Print the classification report to determine the evaluation metric scores.

In [16]:
```
from sklearn.metrics import classification_report
# Predict using the trained Random Forest Classifier model

y_pred = rf.predict(X_test)
```

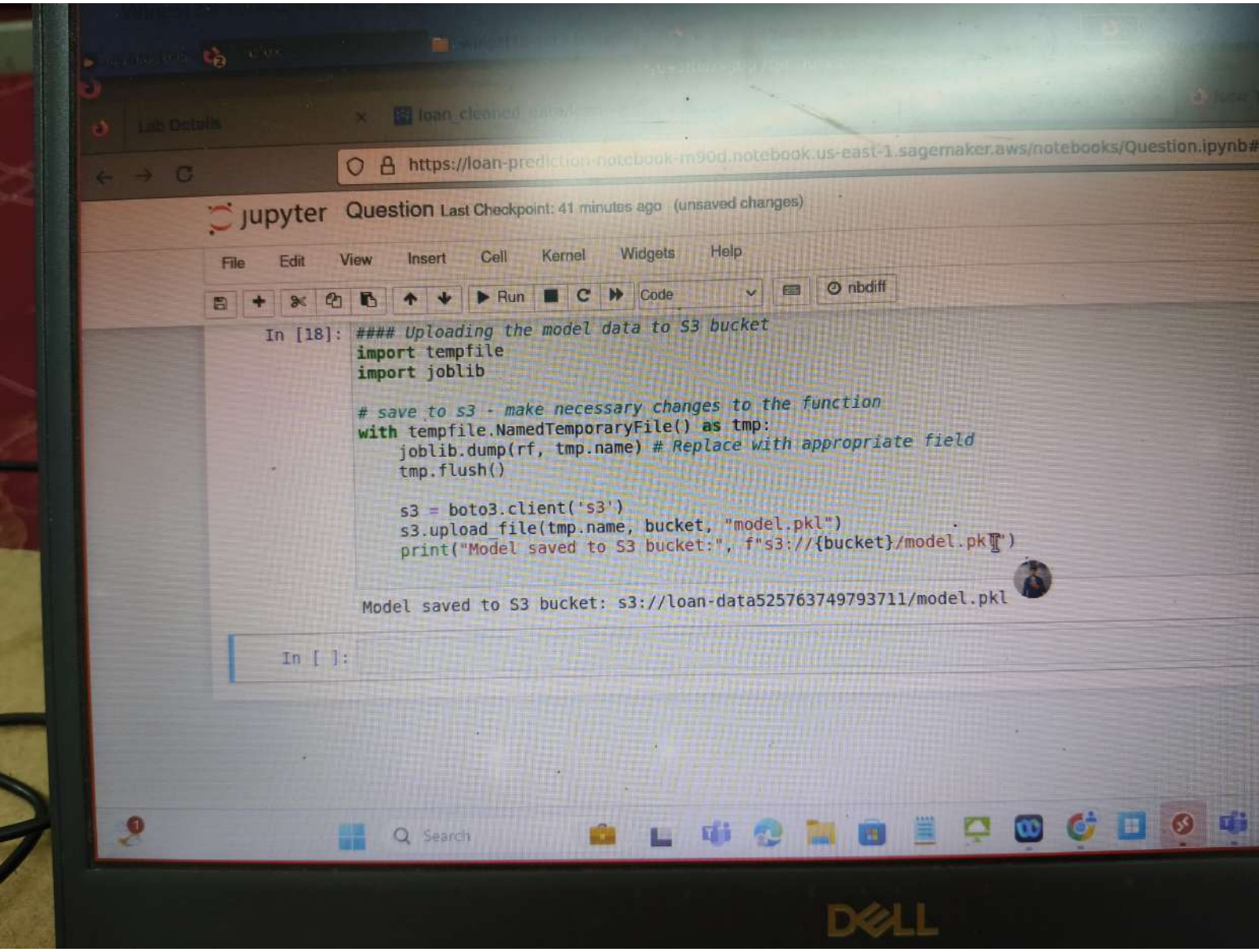
In [17]:
```
# Print the classification report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.96      0.97      1729
           1       0.97      0.98      0.97      1731

    accuracy                           0.97      3460
   macro avg       0.97      0.97      0.97      3460
weighted avg       0.97      0.97      0.97      3460
```

RA

Search

DELL


https://loan-prediction-notebook-m90d.notebook.us-east-1.sagemaker.aws/notebooks/Question.ipynb#
jupyter Question Last Checkpoint: 41 minutes ago (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help
Run Code nbdiff
In [18]:
Uploading the model data to S3 bucket
import tempfile
import joblib
# save to s3 - make necessary changes to the function
with tempfile.NamedTemporaryFile() as tmp:
joblib.dump(rf, tmp.name) # Replace with appropriate field
tmp.flush()
s3 = boto3.client('s3')
s3.upload_file(tmp.name, bucket, "model.pkl")
print("Model saved to S3 bucket:", f"s3://{bucket}/model.pkl")
Model saved to S3 bucket: s3://loan-data525763749793711/model.pkl
In [ ]:
Search
DELL