

LEAD SCORING CASE STUDY

INSIGHTS AND RECOMMENDATIONS

UTTAM KUMAR

AGENDA

Introduction

Data Overview

Key Insights

Visualizations

Recommendations

Action Plan

Conclusion



INTRODUCTION

OBJECTIVE:

1.IDENTIFY FACTORS INFLUENCING LEAD CONVERSION AND RECOMMEND ACTIONABLE STEPS TO IMPROVE CONVERSION RATES.

SCOPE:

1. ANALYZED LEAD DATA USING VARIOUS METRICS.
- 2.FOCUSED ON KEY PERFORMANCE INDICATORS (KPIS) LIKE CONVERSION RATES, SOURCE QUALITY, AND TIME SPENT.

DATA OVERVIEW

- **Dataset Details:**

- Total leads analyzed.
- Key variables: Source, Time Spent, Converted.

- **Assumptions (if any):**

- Data completeness and reliability were assumed.
- Outliers and missing values were addressed.

```
[17]: import pandas as pd
# Loading the dataset and data dictionary using the correct paths
leads_data = pd.read_csv('C:/Users/ashish/Desktop/LeadScoringCaseStudy/leads.csv')
data_dictionary = pd.read_excel('C:/Users/ashish/Desktop/LeadScoringCaseStudy/leads_data_dictionary.xlsx')

# Displaying first few rows of the leads data to inspect the structure
leads_head = leads_data.head()

# Checking for missing values and data types in the dataset
missing_values = leads_data.isnull().sum()
data_types = leads_data.dtypes

# Inspecting the data dictionary
data_dict_head = data_dictionary.head()

leads_head, missing_values, data_types, data_dict_head

[17]:
```

	Prospect ID	Lead Number	Lead Origin	
0	76273697	8846	8846	Website
1	24273696	1212	4212	Referral
2	8149321	4212	4212	Referral
3	8149321	4212	4212	Referral
4	12045678	4212	4212	Referral

```
Lead Source Do Not Call No Call Converted TotalVisits
0 Organic Search No No 0 0.0
1 Organic Search No No 0 1.0
2 Direct Traffic No No 1 2.0
3 Direct Traffic No No 1 1.0
4 Google No No 1 2.0
```

```
[18]: # Handling missing values
# Drop columns with a large percentage of missing values (Threshold: 40% or more)
missing_threshold = 0.4
columns_to_drop = leads_data.columns[leads_data.isnull().sum() > missing_threshold]
leads_data = leads_data.drop(columns=columns_to_drop)

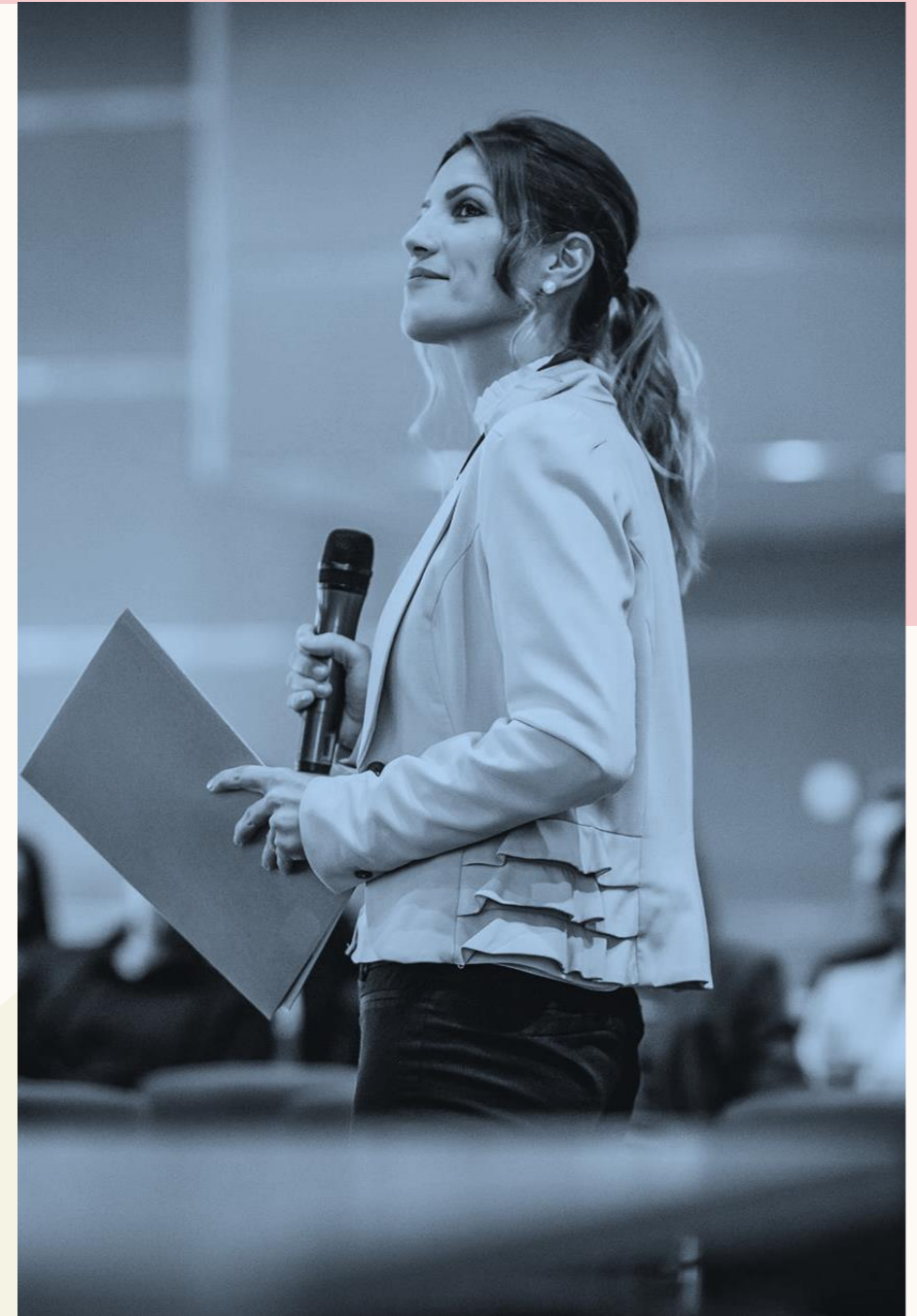
# Impute missing values in remaining columns
# For numerical columns, use median imputation
numerical_cols = leads_data.select_dtypes(include=["float64", "int64"]).columns
leads_data[numerical_cols] = leads_data[numerical_cols].fillna(leads_data[numerical_cols].median())

# For categorical columns, use mode imputation
categorical_cols = leads_data.select_dtypes(include=["object"]).columns
leads_data[categorical_cols] = leads_data[categorical_cols].fillna(leads_data[categorical_cols].mode()[0])

# Encoding categorical variables using one-hot encoding
leads_data_encoded = pd.get_dummies(leads_data, drop_first=True)

# Rescaling numerical variables
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
leads_data_encoded[numerical_cols] = scaler.fit_transform(leads_data_encoded[numerical_cols])

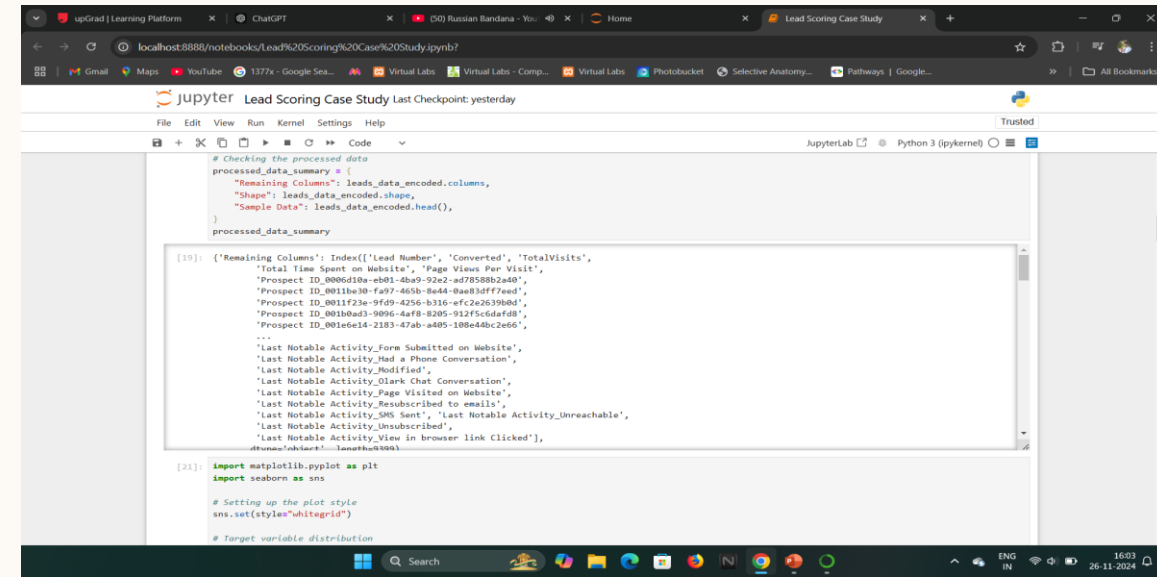
# Checking the processed data
processed_data_summary = {
    "Remaining Columns": leads_data_encoded.columns,
    "Shape": leads_data_encoded.shape,
    "Sample Data": leads_data_encoded.head(),
}
```



KEY INSIGHTS

5

- **Conversion Rate:**
 - Percentage of leads that converted.
 - Comparison by source and time spent.
- **Source Quality:**
 - Highlight top-performing lead sources.
 - E.g., Organic vs Paid traffic performance.
- **Impact of Engagement:**
 - Relationship between time spent and conversion likelihood.



```
# Checking the processed data
processed_data_summary = {
    "Remaining Columns": leads_data_encoded.columns,
    "Shape": leads_data_encoded.shape,
    "Sample Data": leads_data_encoded.head(),
}
processed_data_summary

[19]: {'Remaining Columns': Index(['Lead Number', 'Converted', 'TotalVisits',
                                'Total Time Spent on Website', 'Page Views Per Visit',
                                'Prospect ID_0006d10a-e801-4ba9-92e2-ad78588b2a40',
                                'Prospect ID_0011ba30-fa97-465b-8a44-8a83dfff7ead',
                                'Prospect ID_0011f23a-9fd9-4256-b316-efc2a26399dd',
                                'Prospect ID_001b0ad3-9096-4af8-8205-912f5c5dafd8',
                                'Prospect ID_001e6e14-2183-47ab-a405-108e44bc2e66',
                                ...],
                                dtype='object'),
       'Total Time Spent on Website',
       'Page Views Per Visit',
       'Prospect ID_0006d10a-e801-4ba9-92e2-ad78588b2a40',
       'Prospect ID_0011ba30-fa97-465b-8a44-8a83dfff7ead',
       'Prospect ID_0011f23a-9fd9-4256-b316-efc2a26399dd',
       'Prospect ID_001b0ad3-9096-4af8-8205-912f5c5dafd8',
       'Prospect ID_001e6e14-2183-47ab-a405-108e44bc2e66',
       ...],
       dtype='object')

[20]: {'Remaining Columns': Index(['Lead Number', 'Converted', 'TotalVisits',
                                'Total Time Spent on Website', 'Page Views Per Visit',
                                'Prospect ID_0006d10a-e801-4ba9-92e2-ad78588b2a40',
                                'Prospect ID_0011ba30-fa97-465b-8a44-8a83dfff7ead',
                                'Prospect ID_0011f23a-9fd9-4256-b316-efc2a26399dd',
                                'Prospect ID_001b0ad3-9096-4af8-8205-912f5c5dafd8',
                                'Prospect ID_001e6e14-2183-47ab-a405-108e44bc2e66',
                                ...],
                                dtype='object'),
       'Total Time Spent on Website',
       'Page Views Per Visit',
       'Prospect ID_0006d10a-e801-4ba9-92e2-ad78588b2a40',
       'Prospect ID_0011ba30-fa97-465b-8a44-8a83dfff7ead',
       'Prospect ID_0011f23a-9fd9-4256-b316-efc2a26399dd',
       'Prospect ID_001b0ad3-9096-4af8-8205-912f5c5dafd8',
       'Prospect ID_001e6e14-2183-47ab-a405-108e44bc2e66',
       ...],
       dtype='object')

[21]: import matplotlib.pyplot as plt
import seaborn as sns

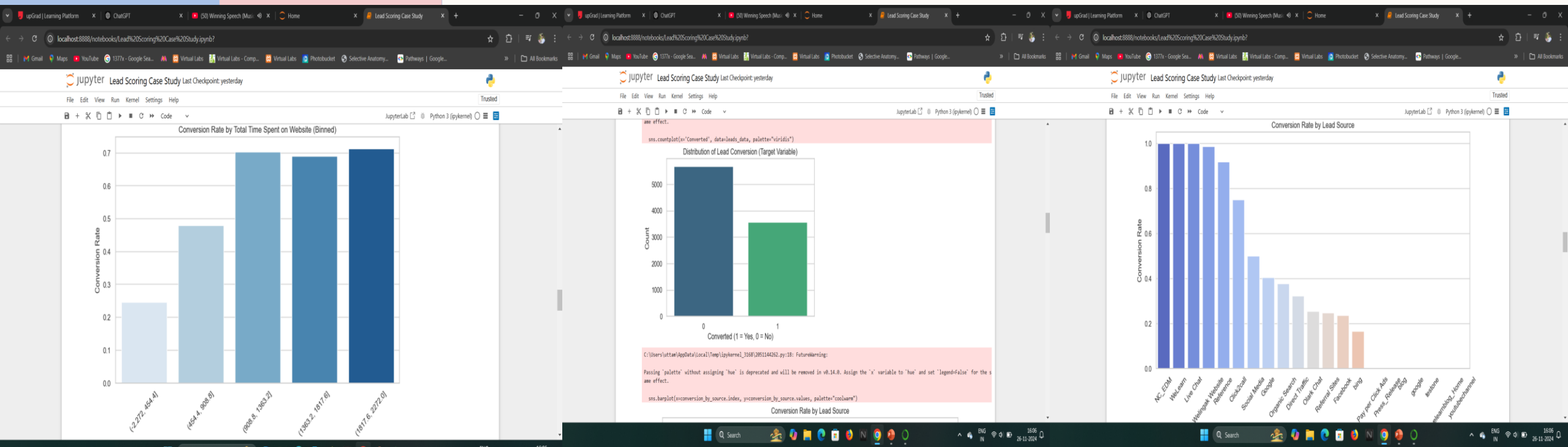
# Setting up the plot style
sns.set(style="whitegrid")

# Target variable distribution
```

VISUALIZATIONS

6

- Include relevant plots such as:
 - **Bar Chart:** Conversion rates by source.
 - **Histogram:** Distribution of time spent by converted leads.
 - **Heatmap** (if applicable): Correlation between variables.



RECOMMENDATIONS

7

•Improve High-Performing Sources:

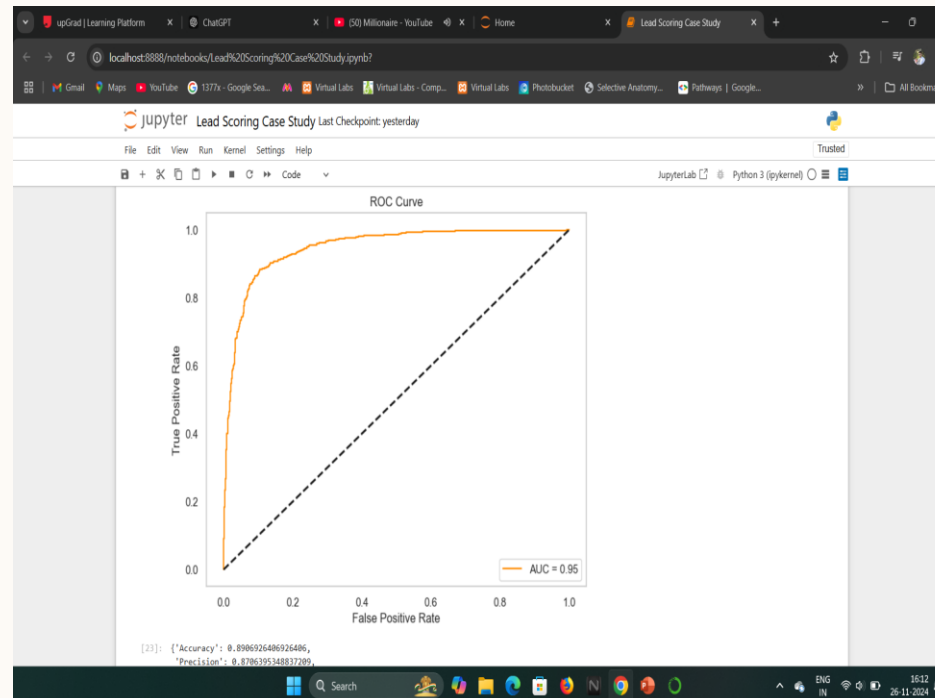
- Allocate more resources to top sources.
- Optimize underperforming channels.

•Engagement Strategies:

- Encourage more time spent on the platform.
- Personalized follow-ups for high-interest leads.

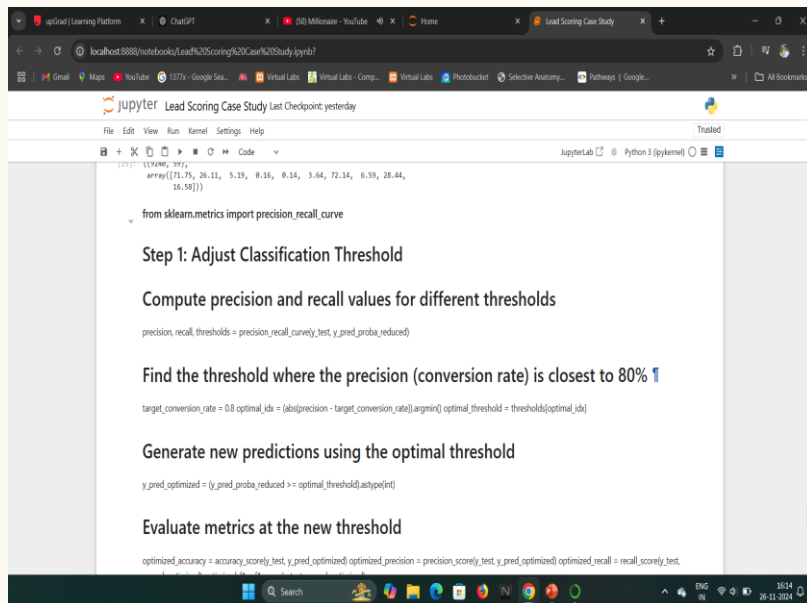
•Monitor and Iterate:

- Regularly analyze conversion trends.
- Adjust strategies based on data.



ACTION PLAN

- **Short Term:**
 - Identify quick wins from top sources.
 - Improve lead capture processes.
- **Long Term:**
 - Invest in predictive modeling for lead scoring.
 - Develop targeted campaigns based on insights.



```
from sklearn.metrics import precision_recall_curve

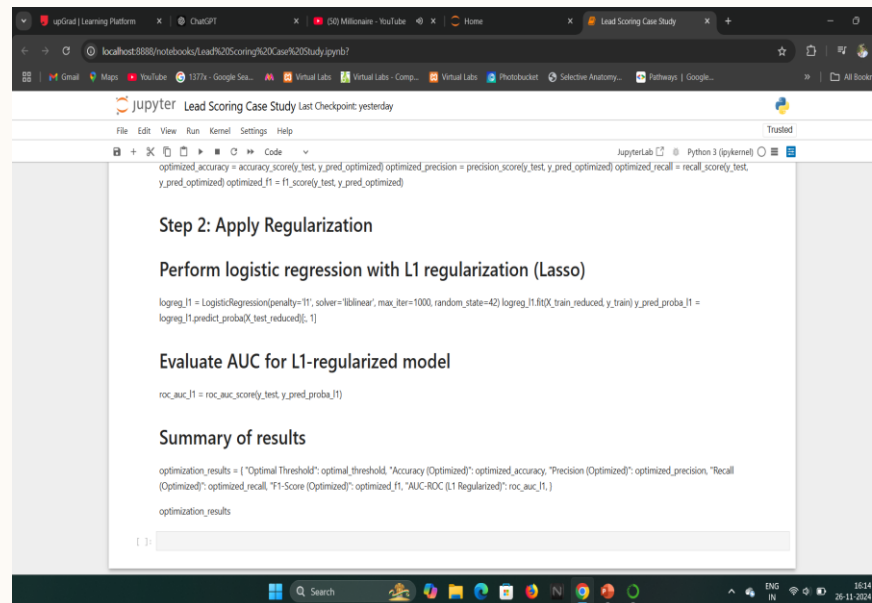
# Step 1: Adjust Classification Threshold

# Compute precision and recall values for different thresholds
precision, recall, thresholds = precision_recall_curve(y_test, y_pred_proba_reduced)

# Find the threshold where the precision (conversion rate) is closest to 80%
target_conversion_rate = 0.8
optimal_idx = (abs(precision - target_conversion_rate).argmin())
optimal_threshold = thresholds[optimal_idx]

# Generate new predictions using the optimal threshold
y_pred_optimized = (y_pred_proba_reduced >= optimal_threshold).astype(int)

# Evaluate metrics at the new threshold
optimized_accuracy = accuracy_score(y_test, y_pred_optimized)
optimized_precision = precision_score(y_test, y_pred_optimized)
optimized_recall = recall_score(y_test, y_pred_optimized)
```



```
from sklearn.metrics import precision_recall_curve

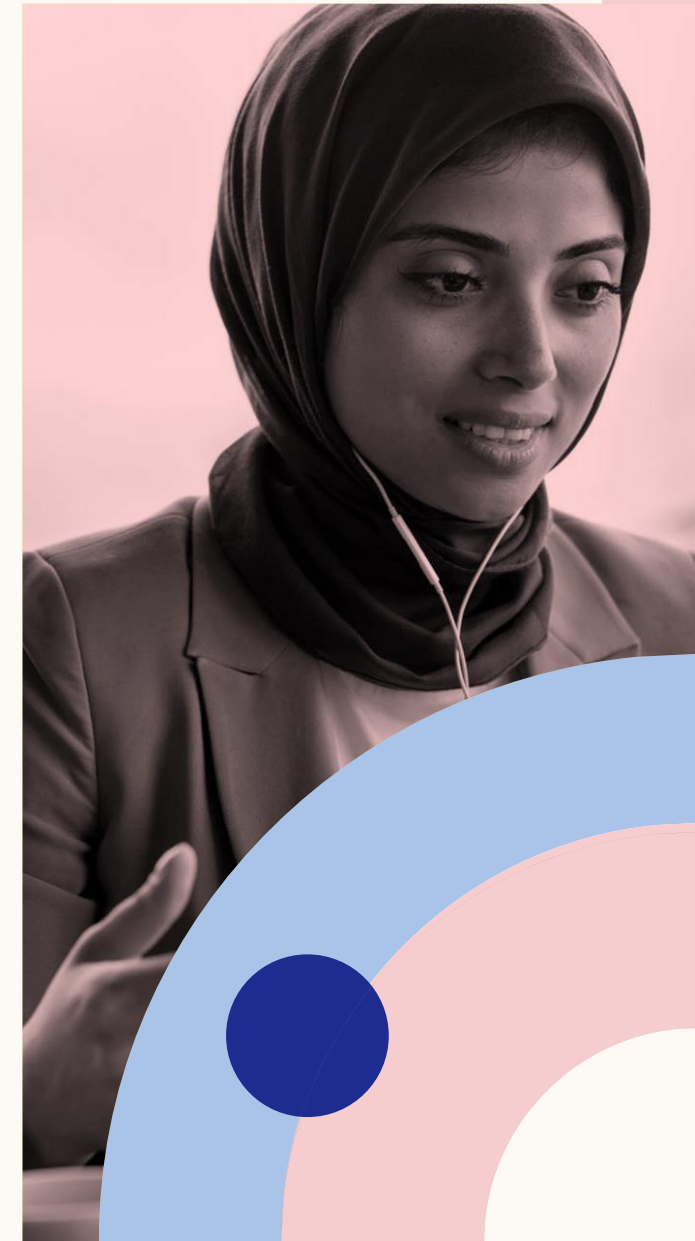
# Step 2: Apply Regularization

# Perform logistic regression with L1 regularization (Lasso)
logreg_l1 = LogisticRegression(penalty='l1', solver='liblinear', max_iter=1000, random_state=42)
logreg_l1.fit(X_train_reduced, y_train)
y_pred_proba_l1 = logreg_l1.predict_proba(X_test_reduced)[0, 1]

# Evaluate AUC for L1-regularized model
roc_auc_l1 = roc_auc_score(y_test, y_pred_proba_l1)

# Summary of results
optimization_results = {
    "Optimal Threshold": optimal_threshold,
    "Accuracy (Optimized)": optimized_accuracy,
    "Precision (Optimized)": optimized_precision,
    "Recall (Optimized)": optimized_recall,
    "F1-Score (Optimized)": optimized_f1,
    "AUC-ROC (L1 Regularized)": roc_auc_l1,
}

print(optimization_results)
```



CONCLUSION

- Recap major findings and their significance.
- Emphasize actionable recommendations and next steps.





THANK YOU

Uttam Kumar

26/04/2024

uttamkumar20142005@gmail.com