BDP

CREDIT CARD FRAUD DETECTION:

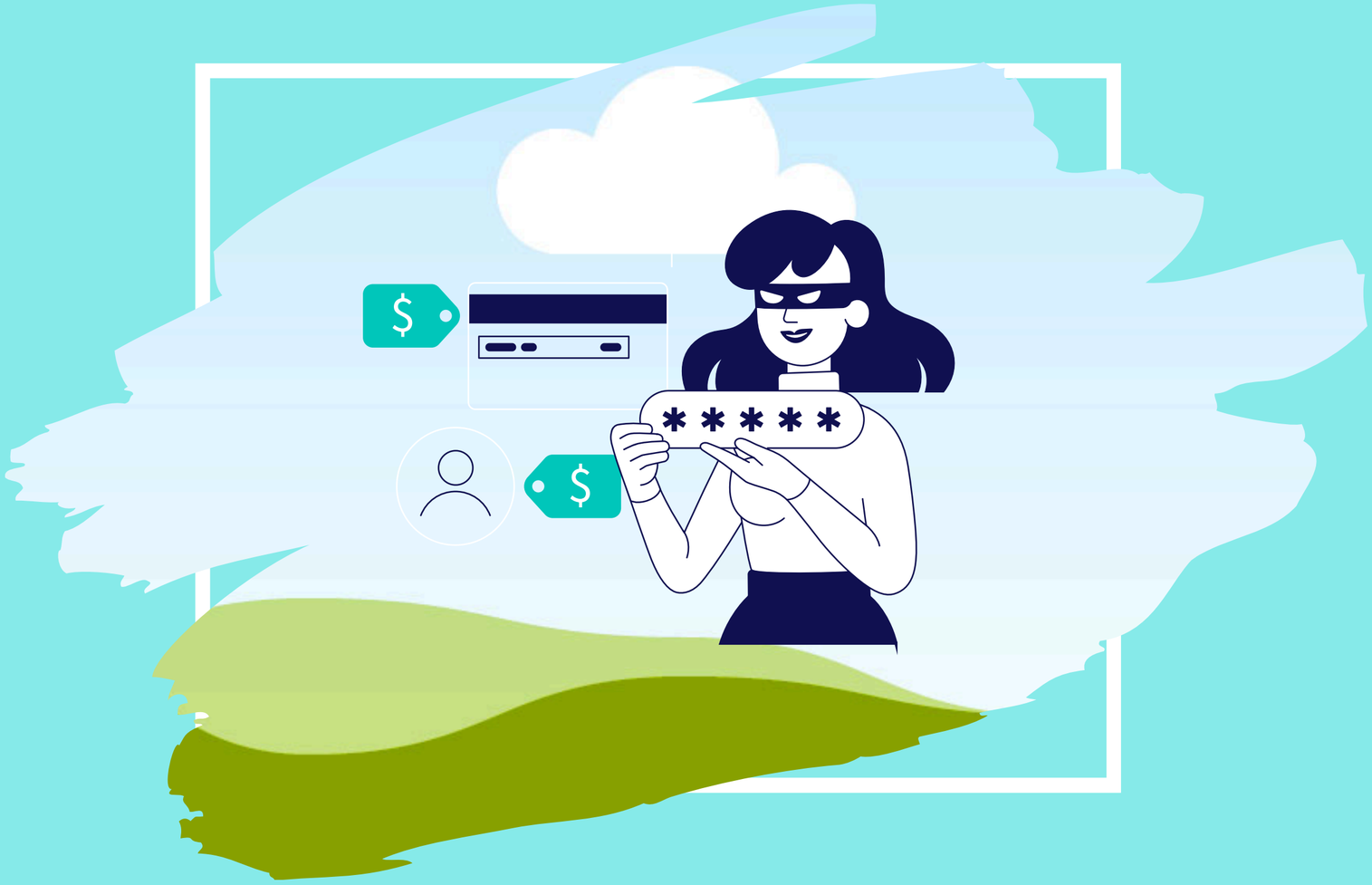
Using PySpark

BHAVIKMANWANI (202318009
MANTHAN SOLANKI (202318012)
ANMOLPOONIA (202318009)

PROBLEM DESCRIPTION:

In the realm of financial transactions, credit card fraud poses a significant challenge for both consumers and financial institutions. The rapid growth of digital transactions has made it easier for fraudsters to exploit vulnerabilities in payment systems. To combat this issue, advanced techniques such as machine learning can be employed to detect fraudulent activities in real-time.

The objective of this project is to develop a credit card fraud detection system using PySpark, a powerful tool for big data processing and machine learning in the Python ecosystem. The dataset consists of a large number of credit card transactions, each characterized by various features such as transaction amount, merchant category, time of transaction, etc. The challenge lies in identifying patterns and anomalies that indicate potential fraudulent behavior.



METHODOLOGY:

01

Data Collection

02

Data Preprocessing

03

EDA

04

Feature Engineering

05

Model Building

06

Model Evaluation

TOOLS & TECH USED:

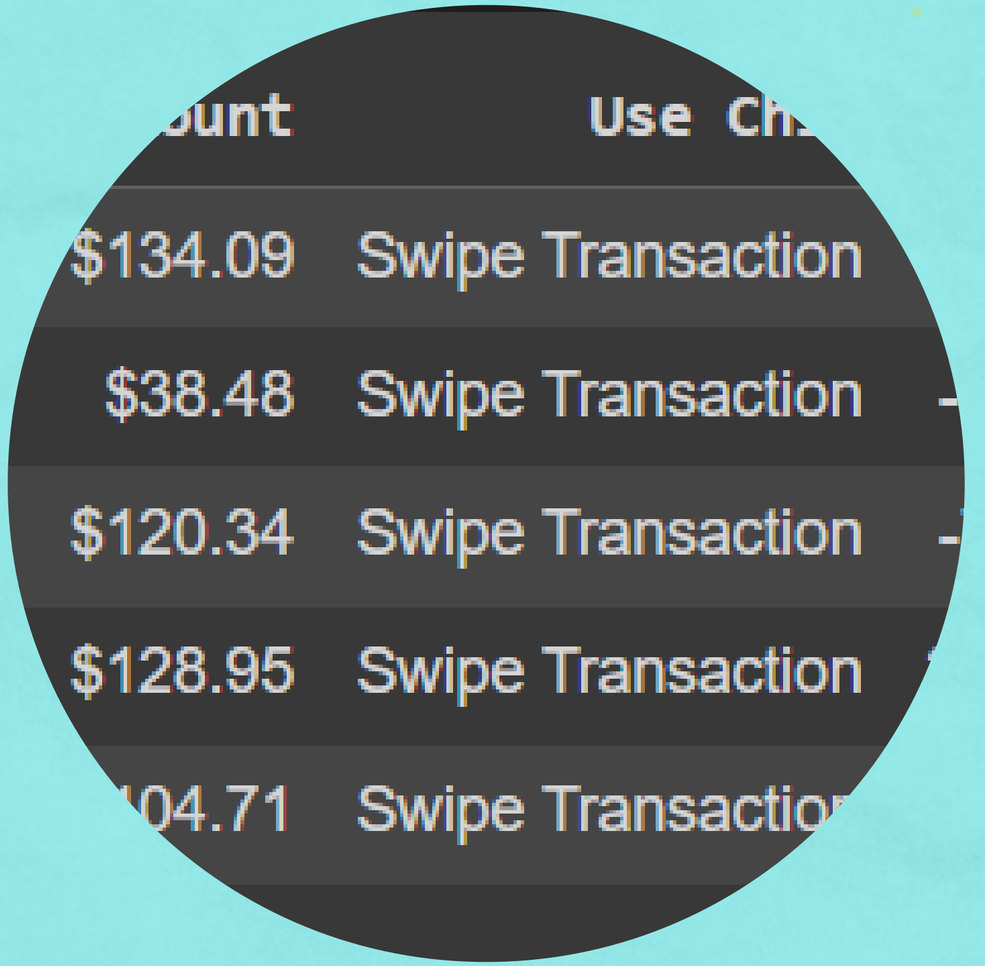
DATABRICKS

PYSPARK

Dataset Source:
https://www.kaggle.com/datasets/ealtman2019/credit-card-transactions?select=credit_card_transactions-ibm_v2.csv

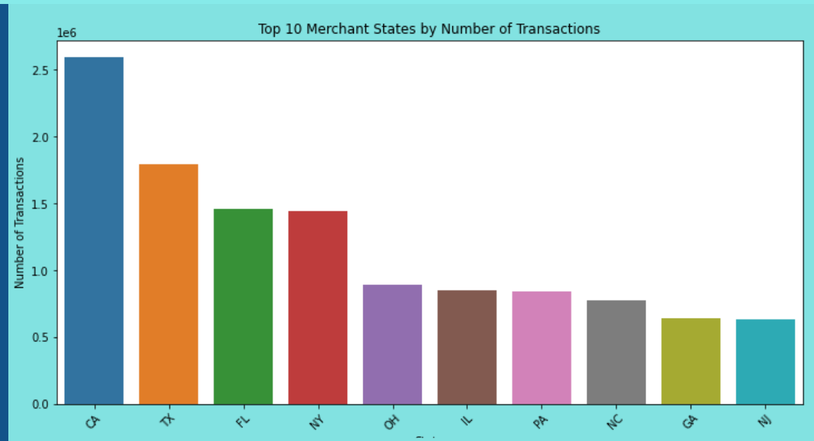
Dataset Description:
Size- 2.35 GB

Index	Data Field Description
0	User: Unique identifier for the user.
1	Card: Unique identifier for the card used in the transaction.
2	Year: Year of the transaction.
3	Month: Month of the transaction.
4	Day: Day of the transaction.
5	Time: Time of the transaction.
6	Amount: Transaction amount.
7	Use Chip: Indicator if the chip was used in the transaction.
8	Merchant Name: Name of the merchant where the transaction occurred.
9	Merchant City: City where the merchant is located.
10	Merchant State: State where the merchant is located.
11	Zip: Zip code of the merchant location.
12	MCC: Merchant Category Code.
13	Errors?: Indicator if there were any errors in the transaction.
14	Is Fraud?: Indicator if the transaction is flagged as fraud.



EXPLORATORY DATA ANALYSIS:

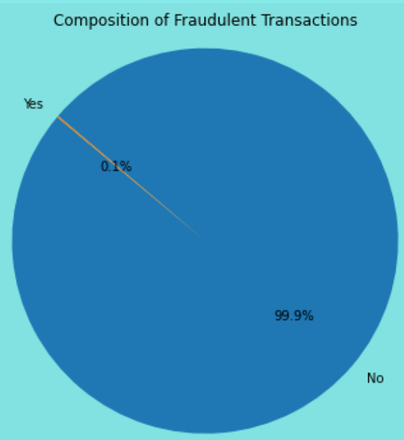
CALIFORNIA LEADING IN TERMS OF VOLUME



Bar Chart showing comparison of States by volume of Transactions

0.1%

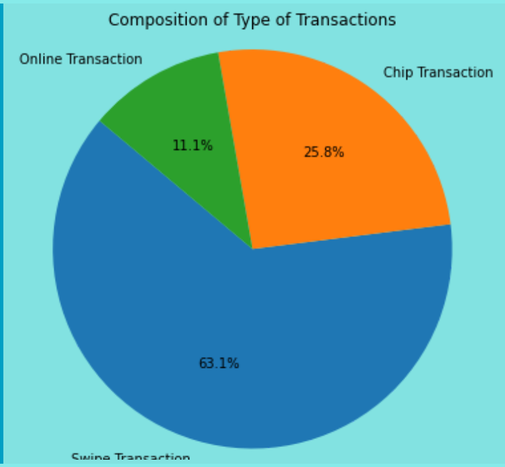
OF TRANSACTION IS BEING DETECTED AS FRAUDELENT



Pie Char Showing percentage of fraud transactions

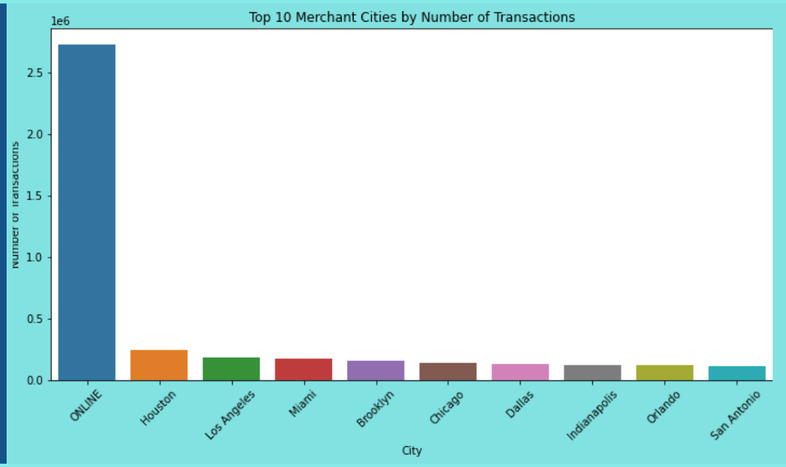
63.1%

OF TRANSACTION IS BEING IDENTIFIED AS SWIPE BASED TRANSACTIONS

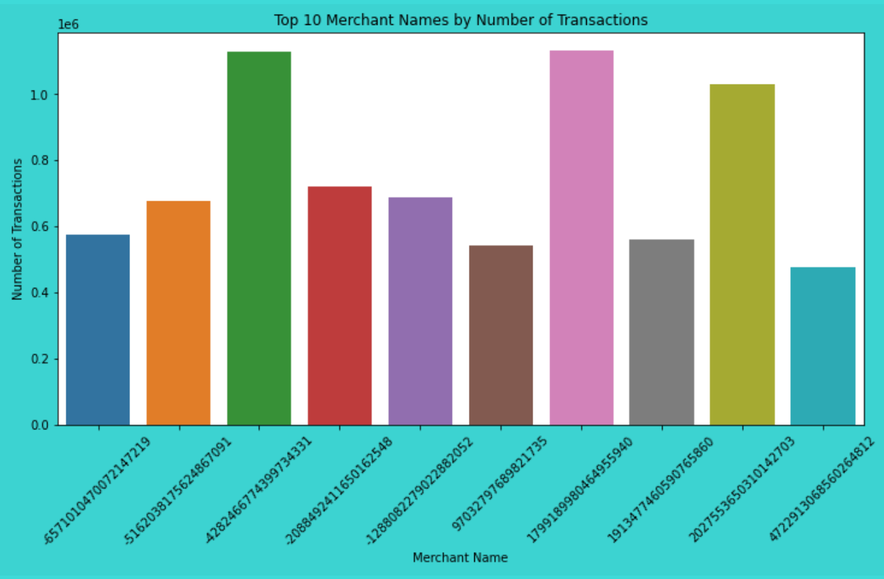


pie chart showing comparison of types of transactions

MAJORITY FRAUD TRANSACTION DETECTED ON ONLINE MODE RATHER THAN MERCHANT LEVEL

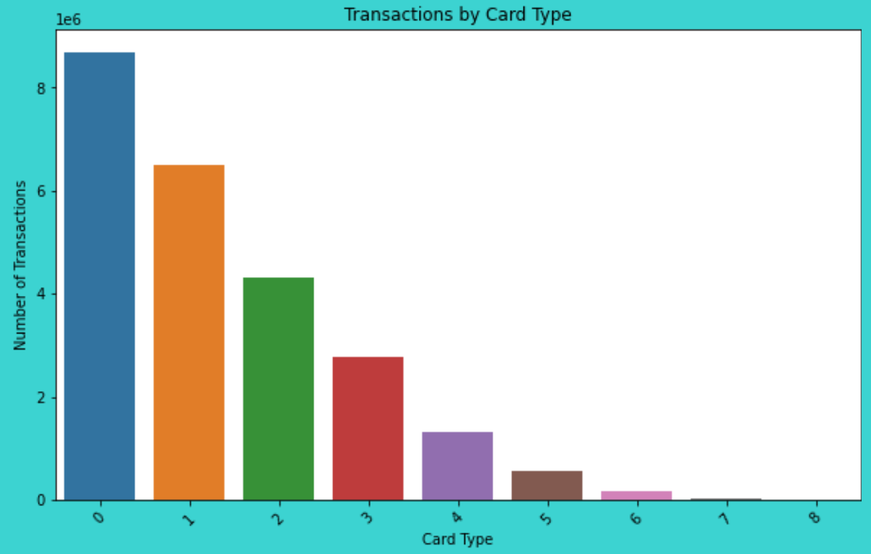


Bar Chart showing comparison of cities

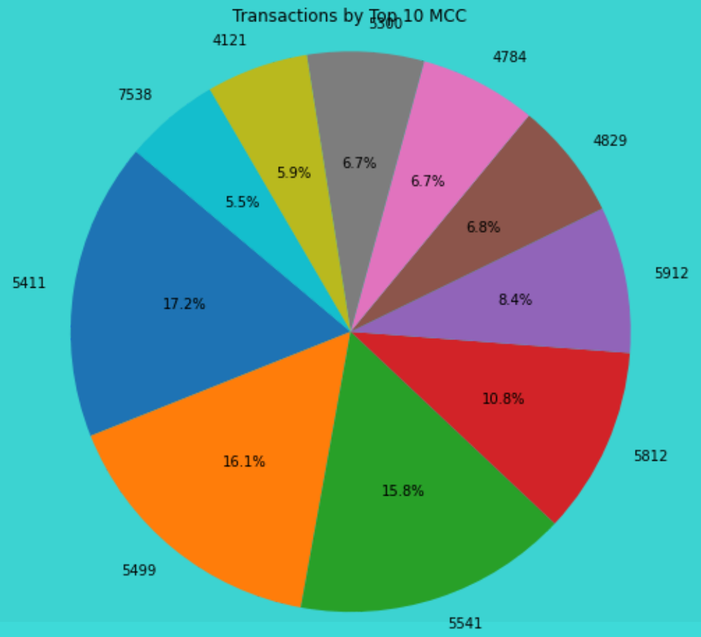


BAR CHAR SHOWING COMPARISON OF MERCHANTS

BAR CHART SHOWING COMPARISON OF CARD TYPES



pie chart showing comparison of MCC



WITH MCC 5411 LEADING BY 17.2%

Feature Engineering:

1

String Indexing:

String indexing is used to convert categorical string columns into numerical indices. Three columns are indexed: "Merchant City", "Use Chip", and "MCC".

2

One-Hot Encoding:

One-hot encoding is used to convert the indexed categorical columns into one-hot encoded vectors. This transformation prepares the categorical data for machine learning algorithms.

3

Vector Assembling:

Vector assembly is used to assemble multiple columns, including the one-hot encoded vectors and numerical columns, into a single feature vector column named "features".

4

Pipeline Creation:

A pipeline is created to sequentially apply the transformations, including indexing, one-hot encoding, and vector assembly, to the data.

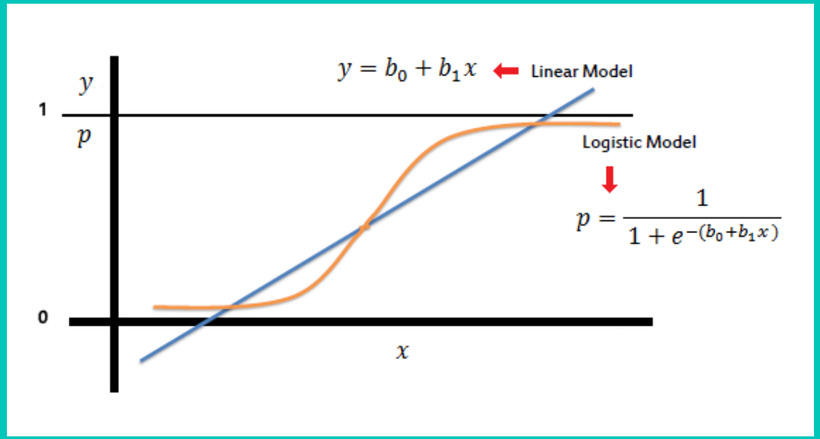
5

Fit-Tranform:

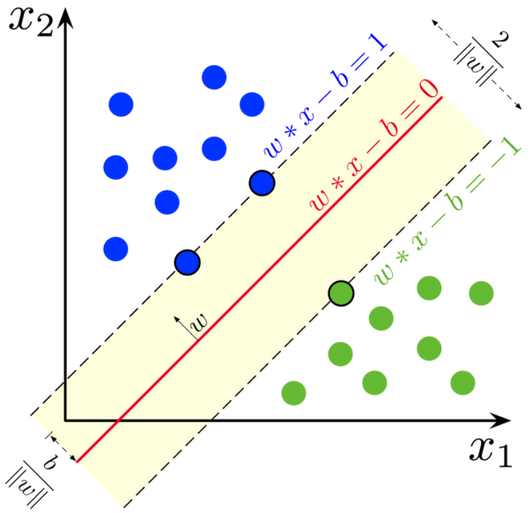
The pipeline is fitted to the data, determining the transformation parameters (e.g., category-to-index mappings) from the data.

ML MODELS:

Logistic Regression:



Support Vector Classifier:



Logistic Regression:

1 Model Training

We trained a Logistic Regression model with the following parameters:

```
logReg = LogisticRegression(labelCol='Is Fraud?', featuresCol='scaled_features',
    maxIter=40)
logreg = logReg.fit(train)
```

2 Model Evaluation

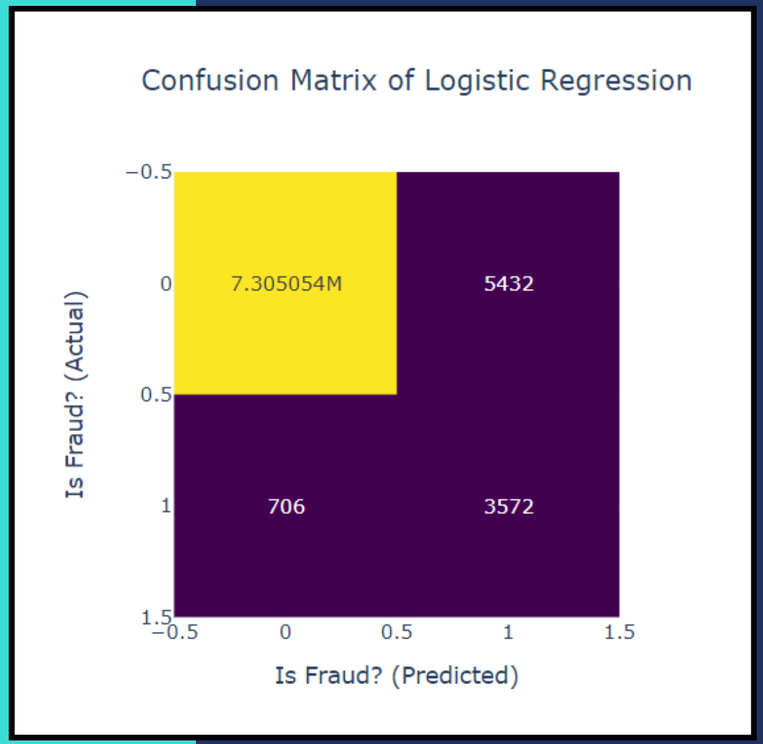
After training the model, we made predictions on the test dataset and evaluated the model's performance. The following table shows the first five rows of the actual fraud labels and the predicted labels:

```
y_pred_log = logreg.transform(test)
y_pred_log.select('Is Fraud?', 'prediction').show(5)
```

```
+---------+----------+
|Is Fraud?|prediction|
+---------+----------+
|        0|       0.0|
|        0|       0.0|
|        0|       0.0|
|        0|       0.0|
|        0|       0.0|
+---------+----------+
only showing top 5 rows
```

Logistic Regression Evaluation Results

Metric	Value
Accuracy	0.9992
Weighted Precision	0.9991
Weighted Recall	0.9992
F1 Score	0.9990



Spport Vector Classifier:

1 Model Training

The Support Vector Classifier (SVC) model is trained using the Linear Support Vector Classification algorithm. The following parameters are used for training:

- **labelCol**: 'Is Fraud?'
- **featuresCol**: 'scaled_features'
- **maxIter**: 50
- **regParam**: 0.5
- **tol**: 1×10^{-5}

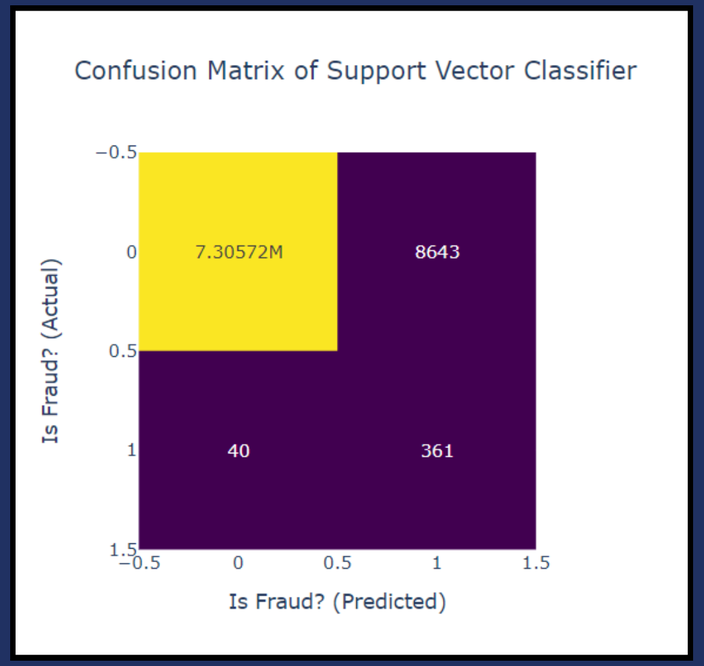
```
from pyspark.ml.classification import LinearSVC

svc = LinearSVC(labelCol='Is Fraud?', featuresCol='scaled_features',
                maxIter=50, regParam=0.5, tol=1e-5)
svc_model = svc.fit(train)
```

2 Model Evaluation

The trained SVC model is used to make predictions on the test dataset. Here are the predictions for the first 5 samples:

```
y_pred_svc = svc_model.transform(test)
y_pred_svc.select('Is Fraud?', 'prediction').show(5)
```

```
+---------+----------+
|Is Fraud?|prediction|
+---------+----------+
|        0|       0.0|
|        0|       0.0|
|        0|       0.0|
|        0|       0.0|
|        0|       0.0|
+---------+----------+
only showing top 5 rows
```



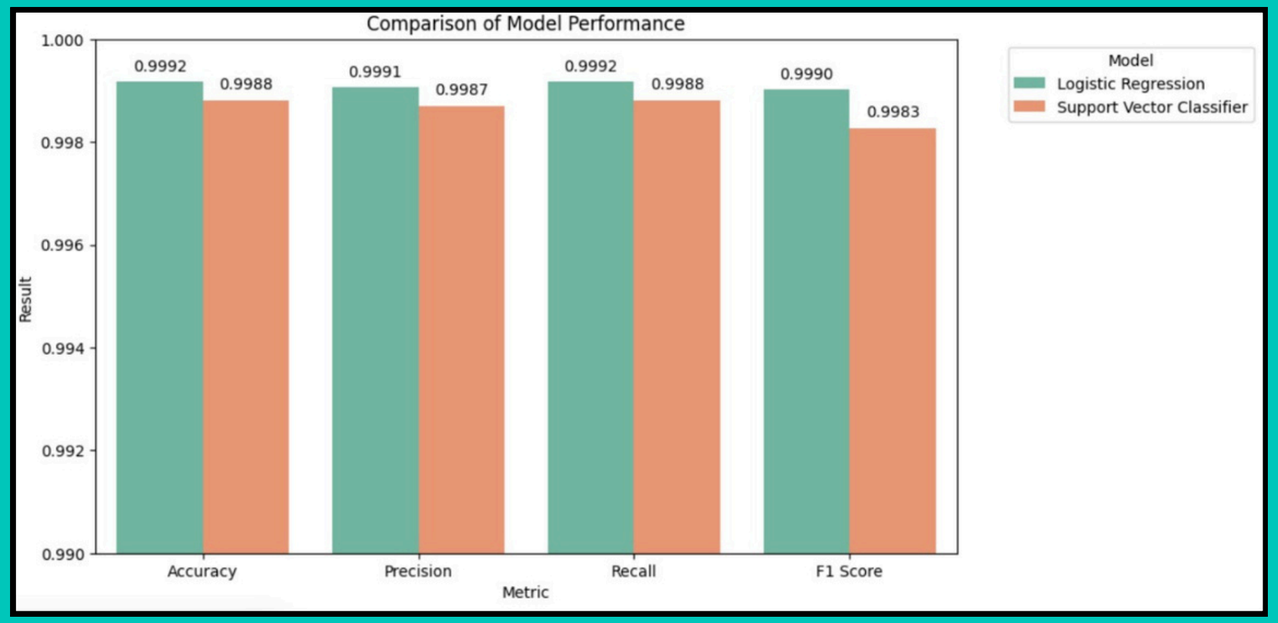
Support Vector Classifier Evaluation Results

Metric	Value
Accuracy	0.9988
Weighted Precision	0.9987
Weighted Recall	0.9988
F1 Score	0.9983

COMPARISON ANALYSIS OF LR AND SVM:

Model Evaluation Results

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.999161	0.999055	0.999161	0.999012
Support Vector Classifier	0.998813	0.998697	0.998813	0.998270



ACCURACY:

LR > SVM

PRECISION:

LR > SVM

RECALL:

LR > SVM

F1 SCORE:

LR > SVM

REFERENCES

• NUTAN BHOGENDRA SHARMA. ”FEATURE TRANSFORMER: VECTORASSEMBLER IN PYSPARK ML FEATURE (PART 3)”. MEDIUM. [ONLINE]. AVAILABLE: HTTPS://MEDIUM.COM/@NUTANBHOGENDRASHARMA/FEATURE-TRANSFORMER-VECTORASSEMBLER-IN-PYSPARK-ML-FEATURE-PART-3-B3C2C3C93EE9#:~:TEXT=VECTORASSEMBLER%20IS%20A%20TRANSFORMER%20THAT,LOGISTIC%20REGRESSION%20AND%20DECISION%20TREES.

• NUTAN BHOGENDRA SHARMA. ”ROLE OF STRINGINDEXER AND PIPELINES IN PYSPARK ML FEATURE”. MEDIUM. [ONLINE]. AVAILABLE: HTTPS://MEDIUM.COM/@NUTANBHOGENDRASHARMA/ROLE-OF-STRINGINDEXER-AND-PIP

• NUTAN BHOGENDRA SHARMA. ”ROLE OF ONEHOTENCODER AND PIPELINES IN PYSPARK ML FEATURE (PART 2)”. MEDIUM. [ONLINE]. AVAILABLE: HTTPS://MEDIUM.COM/@NUTANBHOGENDRASHARMA/ROLE-OF-ONEHOTENCODER-AND-PIPELINES-IN-PYSPARK-ML-FEATURE-PART-2-3275767E74F0.

• QIANG YAN, YONGLI LIU, HAO SHEN, ZHIWEN YU, XIANHAI YU. ”A NEW ARTIFICIAL BEE COLONY ALGORITHM FOR SELECTING OPTIMAL FEATURES IN MACHINE LEARNING-BASED SOFTWARE DEFECT PREDICTION”. SCIENCEDIRECT. [ONLINE]. AVAILABLE: HTTPS://WWW.SCIENCEDIRECT.COM/SCIENCE/ARTICLE/ PII/S187705092030065X.

• KAGGLE. ”CREDIT CARD TRANSACTIONS”. KAGGLE. [ONLINE]. AVAILABLE: HTTPS://WWW.KAGGLE.COM/DATASETS/EALTMAN2019/CREDIT-CARD-TRANSACTIONS?SELECT=CREDIT_CARD_TRANSACTIONS-IBM_V2.CSV

• VIDYASAGAR MACHUPALLI. ”BUILDING A CREDIT CARD FRAUD DETECTION ONLINE TRAINING PIPELINE WITH RIVER ML AND APACHE FLINK”. TOWARDS DATA SCIENCE. [ONLINE]. AVAILABLE: HTTPS://TOWARDSDATASCIENCE.COM/BUILDING-A-CREDIT-CARD-FRAUD-DETECTION-ONLINE-TRAINING-PIPELINE-WITH-RI

• ANALYTICS VIDHYA. ”SUPPORT VECTOR MACHINES(SVM) – A COMPLETE GUIDE FOR BEGINNERS”. ANALYTICS VIDHYA. [ONLINE]. AVAILABLE: HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2021/ 10/SUPPORT-VECTOR-MACHINESSVM-A-COMPLETE-GUIDE-FOR-BEGINNERS/.

• BUILTIN. ”LOGISTIC REGRESSION: UNDERSTANDING BUILDING SIMPLE MODELS IN PYTHON”. BUILTIN. [ONLINE]. AVAILABLE: HTTPS://BUILTIN.COM/ARTICLES/LOGISTIC-CLASSIFIER.