Predicting Fake Job Postings

Dev Divyendh Dhinakaran G01450299
Tejaswi Samineni G01460925

Assignment 2
CS 657 Mining Massive Datasets

Abstract

This report details the processes undertaken to prepare the dataset, train, validate, and test various machine learning models for detecting fraudulent job postings. The dataset was pre-processed and cleaned, and imbalanced classes were handled using PySpark's sampling techniques. Four models were evaluated: Logistic Regression, Linear SVC, Random Forest Classifier, and Multilayer Perceptron Classifier. The performance of each model was assessed using accuracy and F1 scores.

1 Introduction

Fraudulent job postings can lead to significant financial loss and reputational damage for both job seekers and legitimate companies. Therefore, identifying fraudulent job postings is a crucial task. In this project, we applied various machine learning techniques using PySpark to classify job postings as fraudulent or non-fraudulent.

2 Data Preparation

The initial dataset contained various features related to job postings. The data preparation steps included:

2.1 Encoding the Label Column

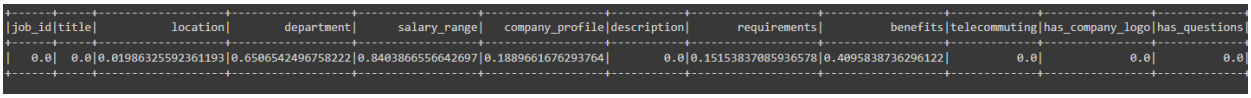
The label column `fraudulent` was encoded, and any invalid entries (values other than 0 or 1) were removed.

2.2 Handling Missing Values

- A count of missing values per attribute was performed to determine the percentage of missing data.
- Attributes with more than 1% missing values were removed from the dataset.

Attribute	Percentage
job_id	0.0%
title	0.0%
location	0.01986%
department	0.65065%
salary_range	0.84038%
company_profile	0.18896%
description	0.0%
requirements	0.15153%
benefits	0.40958%
telecommuting	0.0%
has_company_logo	0.0%
has_questions	0.0%
employment_type	0.19291%
required_experience	0.39343%
required_education	0.45155%
industry	0.27507%
function	0.36296%
fraudulent	0.0%

Table 1: Percentage of Missing Values per Attribute



job_id	title	location	department	salary_range	company_profile	description	requirements	benefits	telecommuting	has_company_logo	has_questions
0.0	0.0	0.01986325592361193	0.6506542496758222	0.8403866556642697	0.1889661676293764	0.0	0.1515383708593657 8	0.4095838736296122	0.0	0.0	0.0

Figure 1: Percentage of missing values per attribute (part 1)

has_questions	employment_type	required_experience	required_education	industry	function	fraudulent
0.0	0.19291524224920428	0.3934339266768832	0.451550159141813	0.2750795709065189	0.36296121655074853	0.0

Figure 2: Percentage of missing values per attribute (part 2)

2.3 Data Cleaning

The following cleaning operations were performed on the dataset:

- Removal of non-letter characters.
- Conversion of text to lower case.
- Removal of multiple spaces.

job_id	title	description	telecommuting	has_company_logo	has_questions	fraudulent
1	food a fastgrowin...	food a fastgrowin...	0	1	0	0
2	organised focused...	organised focused...	0	1	0	0
3	our client locate...	our client locate...	0	1	0	0
4	the company esri ...	the company esri ...	0	1	0	0
5	job title itemiza...	job title itemiza...	0	1	1	0

only showing top 5 rows

Figure 3: cleaned data (after basic text cleaning)

3 Handling Imbalanced Data

The dataset exhibited a significant class imbalance, with a predominance of non-fraudulent postings. To address this:

- The `sampleBy` function in PySpark was used to undersample the majority class, thereby balancing the dataset.

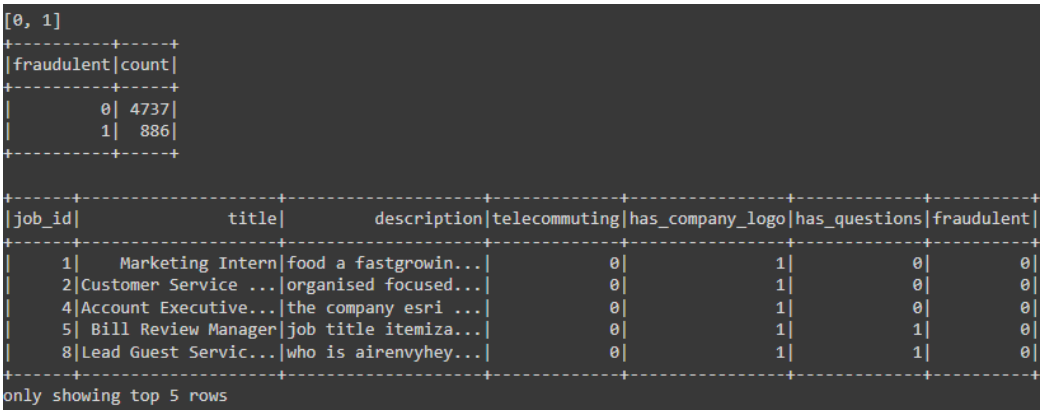


Figure 4: Down Sampling the Data for a ratio of {0: 0.3, 1: 1.0}

4 Feature Extraction

Text features were extracted from the `title` and `description` columns using the following steps:

- Tokenization of text into words.
- Removal of stopwords.
- Conversion of text into TF-IDF vectors.

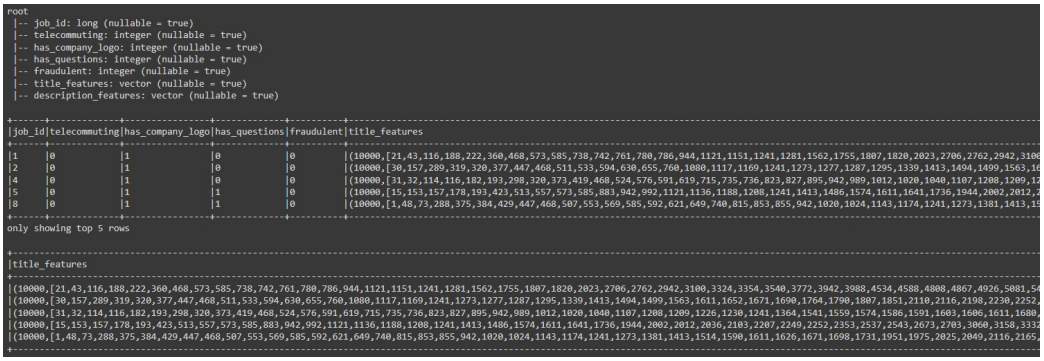


Figure 5: Down Sampling the Data for a ratio of {0: 0.3, 1: 1.0}

5 Random Split

The dataset was randomly split into training (70%) and test (30%) sets.

6 Model Training and Evaluation

Four machine learning models were trained, validated, and tested:

- Logistic Regression
- Linear SVC
- Random Forest Classifier
- Multilayer Perceptron Classifier

6.1 Cross-Validation

For each model, a 10-fold cross-validation was performed to optimize hyperparameters and prevent overfitting. The following metrics were used to evaluate model performance:

- Accuracy
- F1 Score

6.2 Logistic Regression

Metric	Value
Accuracy	0.9468
F1 Score	0.9444
Best Parameters	regParam: 0.01, maxIter: 20, elasticNetParam: 0.0, family: auto

Table 2: Logistic Regression Performance

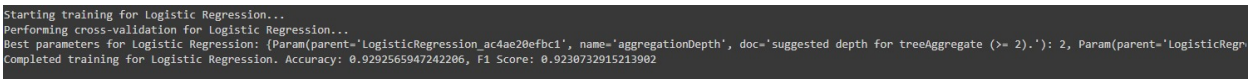


Figure 6: Random Forest Output

6.3 Linear SVC

Metric	Value
Accuracy	0.9375
F1 Score	0.9352
Best Parameters	regParam: 1.0, maxIter: 100, standardization: True, aggregationDepth: 2

Table 3: Linear SVC Performance

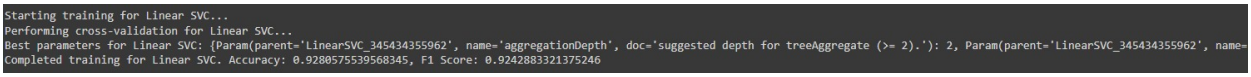


Figure 7: Linear SVC Output

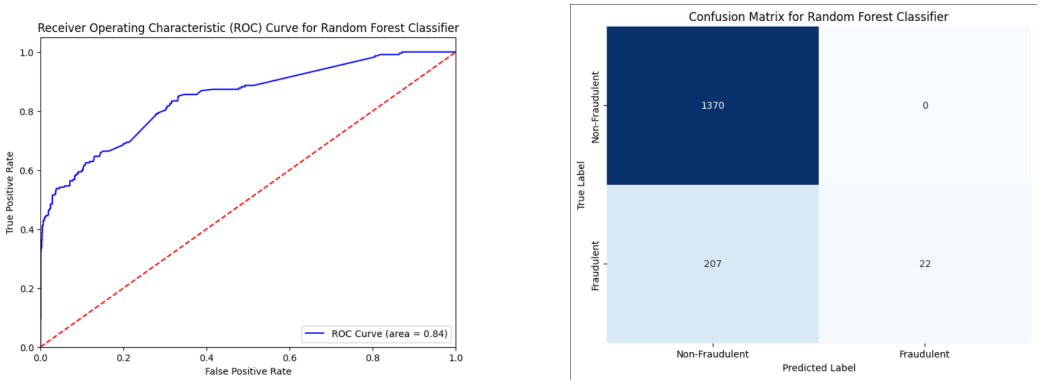
6.4 Random Forest Classifier

Metric	Value
Accuracy	0.8730
F1 Score	0.8269
Best Parameters	Bootstrap: True Impurity: gini Max Depth: 10 Num Trees: 10

Table 4: Random Forest Classifier Performance

```
Starting training for Random Forest Classifier...
Completed training for Random Forest Classifier.
Best parameters for Random Forest Classifier: {Param(parent='RandomForestClassifier_3da1e832b17a', name='bootstrap', doc='Whether bootstrap samp
Accuracy: 0.8705440900562852
F1 Score: 0.8217094195262513
```

Figure 8: Random Forest Output



(a) ROC Curve (b) Confusion Matrix

Figure 9: Roc and Comfusion matix for Random Forest

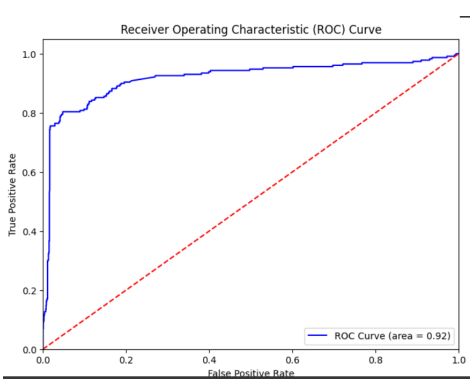
6.5 Multilayer Perceptron Classifier

Metric	Value
Accuracy	0.9425
F1 Score	0.9414
Best Parameters	Block Size: 128, Max Iter: 100, Step Size: 0.03, Solver: l-bfgs, Layers: [20003, 5, 4, 3]

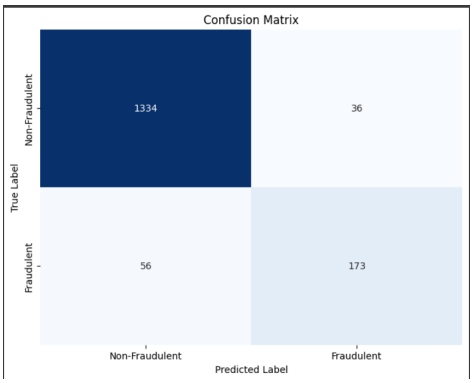
Table 5: Multilayer Perceptron Classifier Performance

```
Starting training for Multilayer Perceptron Classifier...
Number of features in the dataset: 20003
Starting training for Multilayer Perceptron Classifier...
Completed training for Multilayer Perceptron Classifier.
Best parameters for Multilayer Perceptron Classifier: {Param(parent='MultilayerPerceptronClassifier_b1ee631137fd', name='blockSize', doc='block s
Accuracy: 0.9424640400250156
F1 Score: 0.9413588972559905
```

Figure 10: Multilayer Perceptron Output



(a) ROC Curve



(b) Confusion Matrix

Figure 11: Roc and Comfusion matix for Multi-Layer Perceptron

7 Conclusion

This project demonstrated the efficacy of various machine learning models in detecting fraudulent job postings. The data preparation, handling of missing values, and techniques used to manage class imbalance significantly contributed to the performance of the models. Future work may involve exploring additional feature extraction techniques and tuning hyperparameters further to enhance model accuracy.