Today's theme is about

Real/Fake Job Posting Prediction

It's a Story Time!

FCED Team 5:

Shane Lee

Guo Ke Xuan

Lau Kai Feng

Contents

Introduction

There is an increase in frauds and scams around the world, this includes fake jobs as well. When one is searching for a new job they may get tricked into applying for a fake job and then get scammed instead.

Fake job listings

This dataset contains 18K job descriptions out of which about 800 are fake. The data consists of both textual information and meta-information about the jobs.

Problem Definition

What factors determine if a job posting is real or fraudulent?

Why identify fraudulent job listings?

It is to prevent data harvesting and scams on various job posting websites to lure job seekers, which can cause both financial loss and identity theft of individuals.

What are the variables?

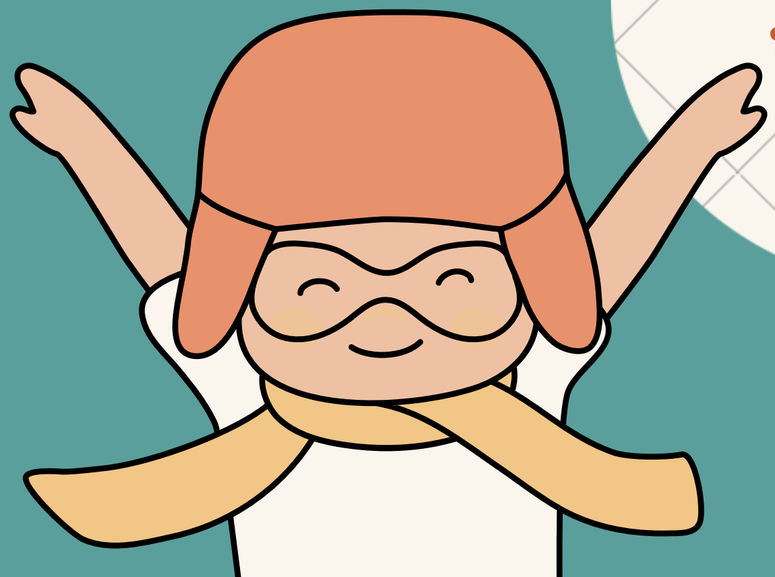
Presence of salary value, presence of questions, company logo, minimum and maximum salary, level of experience, industry

Our solution

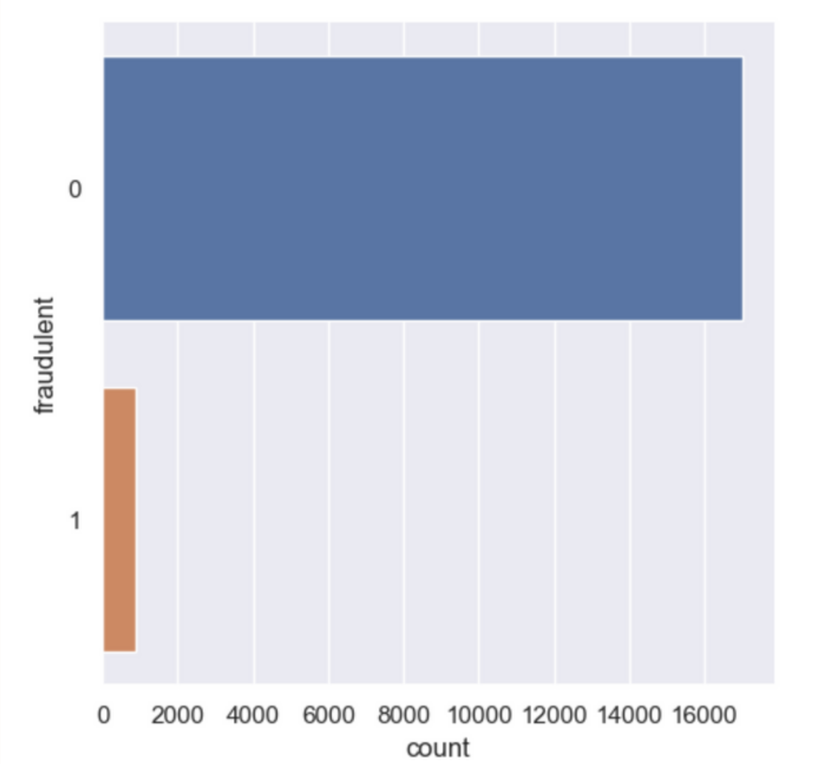
We will be using Natural Language Processing (NLP) techniques to classify job listings as fraudulent or not.
Making use of Decision Tree Classifier, Random forest classification and logistic regression to derive our findings.

LET'S GET TO KNOW MORE ABOUT

Data Preparation & Cleaning



Real : Fake Jobs count



	count
fraudulent	
0	17014
1	866

Insights

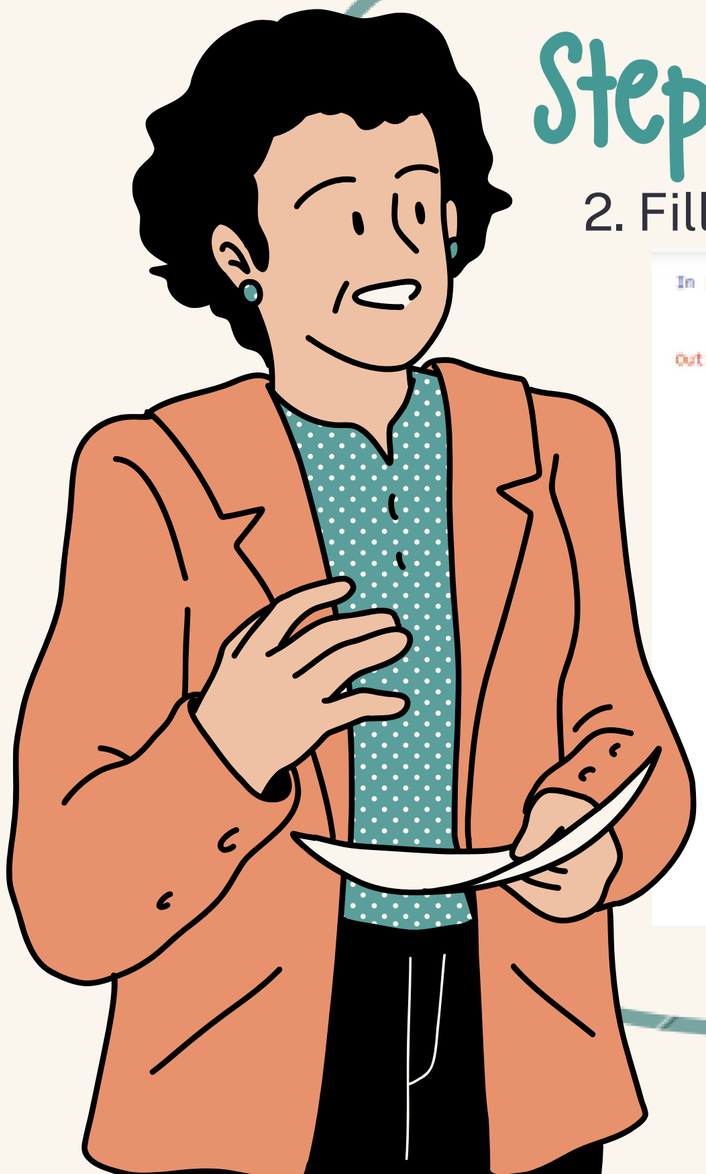
- data telecommuting, has_company_logo, has_questions. are not integer data type but a boolean data type

- Job_id, title,telecommuting, has_company_logo and has_questions do not have any missing details from the data set

Steps Taken:

1. Removal of columns: Job IDs, salary range, departments

```
[*]: #Removing undesired columns & nan
     FakejobData.function.fillna(FakejobData.department,inplace=True)
     FakejobData.drop(columns=['job_id','salary_range','department'],inplace=True)
```



Steps Taken:

2. Filling in values for cells with empty categorical data

In [14]:
```
#filling nan in categorical data
categ_cols=FakejobData[categ].fillna('None')
categ_cols
```

Out[14]:

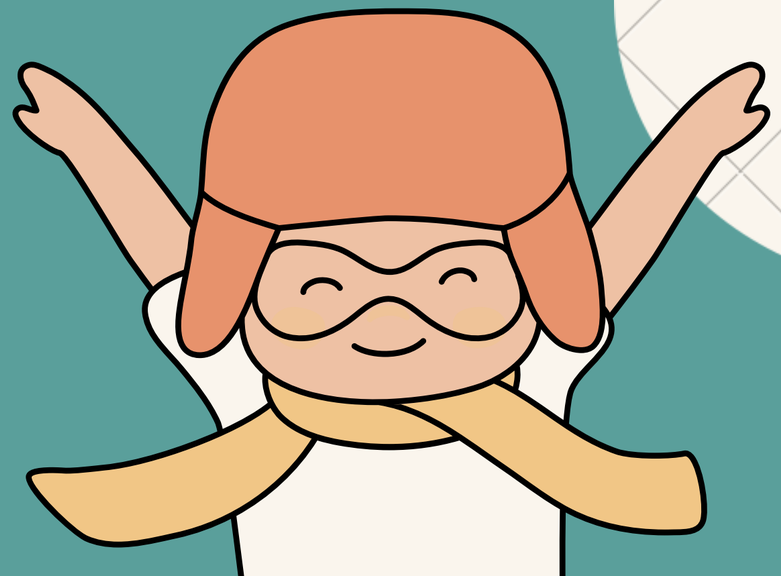
	employment_type	required_experience	required_education	industry	function	telecommuting	has_company_logo	has_questions	fraudulent
0	Other	Internship	None	None	Marketing	0	1	0	0
1	Full-time	Not Applicable	None	Marketing and Advertising	Customer Service	0	1	0	0
2	None	None	None	None	None	0	1	0	0
3	Full-time	Mid-Senior level	Bachelor's Degree	Computer Software	Sales	0	1	0	0
4	Full-time	Mid-Senior level	Bachelor's Degree	Hospital & Health Care	Health Care Provider	0	1	1	0
...	...	...	...	...	...	...	...	...	...
17875	Full-time	Mid-Senior level	None	Computer Software	Sales	0	1	1	0
17876	Full-time	Mid-Senior level	Bachelor's Degree	Internet	Accounting/Auditing	0	1	1	0
17877	Full-time	None	None	None	None	0	0	0	0
17878	Contract	Not Applicable	Professional	Graphic Design	Design	0	0	1	0
17879	Full-time	Mid-Senior level	None	Computer Software	Engineering	0	1	1	0

17880 rows × 9 columns

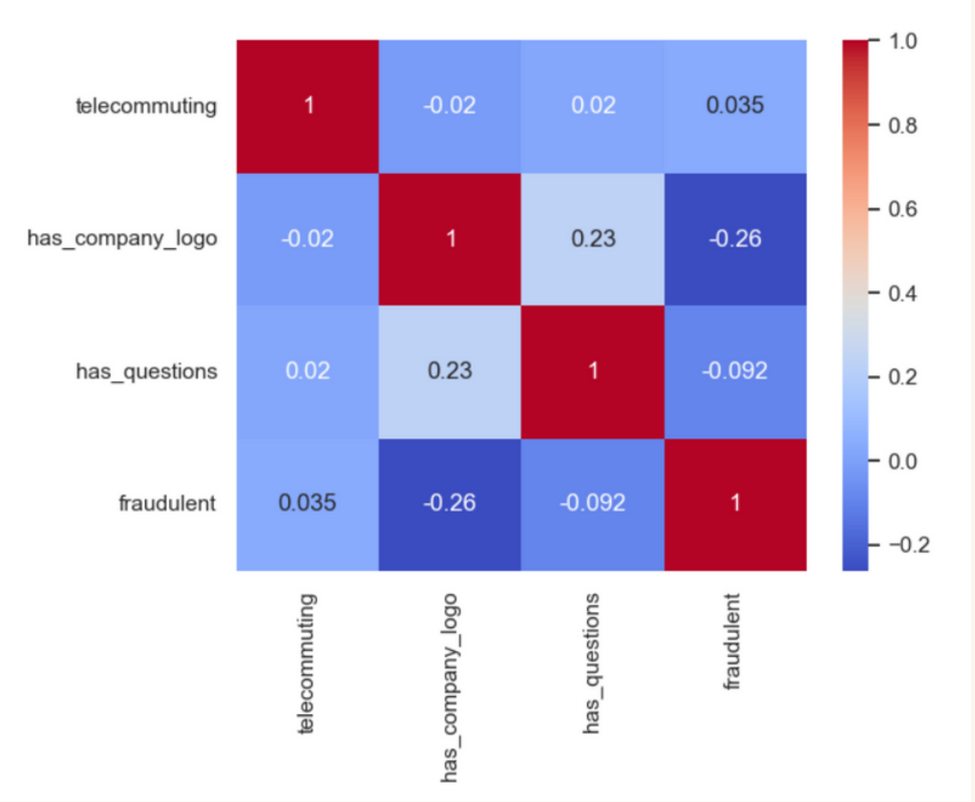
LET'S GET TO KNOW MORE ABOUT

Exploratory Data Analysis

(EDA)



EDA


telecommuting
has_company_logo
has_questions
fraudulent
1
-0.02
0.02
0.035
-0.02
1
0.23
-0.26
0.02
0.23
1
-0.092
0.035
-0.26
-0.092
1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
telecommuting
has_company_logo
has_questions
fraudulent

EDA

- break descriptions up into readable portions and to have a visual representation of the data
- using word stop analysis to filter out filler words
- standardised all the letters to be in lower-case

EDA

[23]:
```
#Cleaning Text & Removing Stopwords and Stemming

stemmer=PorterStemmer()
stop=set(stopwords.words('english'))
```

[24]:
```
def column_clean(text):
    text = text.lower()
    text = re.sub('[^a-zA-Z\s*]','',text)
    text=text.split()
    text=[stemmer.stem(word) for word in text if word not in set(stopwords.words('english'))]
    return (text)
```

[25]:
```
# Splitting Text Data to Fraud or Not Fraud
txt_fraud=txt_cols[txt_cols['fraudulent']==1]
txt_not_fraud=txt_cols[txt_cols['fraudulent']==0]
```

[26]:
```
#Apply cleaning to the title

txt_fraud['title']=txt_fraud['title'].apply(column_clean)
txt_not_fraud['title']=txt_not_fraud['title'].apply(column_clean)
```

EDA


Most Common Words in Real jobs
full time
commun skill
custom servic
project manag
year experi
work close
busi process
team member
social media
ideal candid
softwar develop
work environ
around world
high qualiti
well
new york
benefit packag
design develop
user experi
skill abil
includ limit
increas product
verbal written
attent detail
comput scienc
develop team
commun document
web develop
join team
account manag
track record
abl work
softwar engin
best practic
experi work
least year
competit salari
need relev job
bachelor degre
job descript
school diploma
degre requir
written commun


Most Common Words in Fake jobs
manag
develop
project
requir
commun skill
process
engin
account
respons
custom servic
oper
assist
report
technic
includ
work
experi
team
data
entri
year experi
abil work full time
equip
program
abl work
knowledg
document
project manag
product
plan
design
maintain
work home
understand
ensur
andor
perform
coordin
system

EDA


Most Common titles words in Fake jobs
engin
custom servic
data entri
entri clerk
assist
administr
clerk posit
manag
home base
base payrol
payrol data
director
senior
posit earn
staff want
work home
administr assist
servic repes
entri admincler
part time
want urgent
earn daili
develop
cruis staff
admincler posit
typistdata entri
payrol typistdata
project manag
call center
specialist
account
system
network market


Most Common titles words in Real jobs
manag
custom servic
english teacher
teacher abroad
softwar engin
develop
specialist
senior
servic associ
engin
director
assist
web develop
year old
abroad convers
apprenticeship na
product manag
intern
internship
market
part time
account manag
coordin
sale repres
graduat english
analyst
lead
sale
account execut
market manag
project manag
design
junior
busi develop

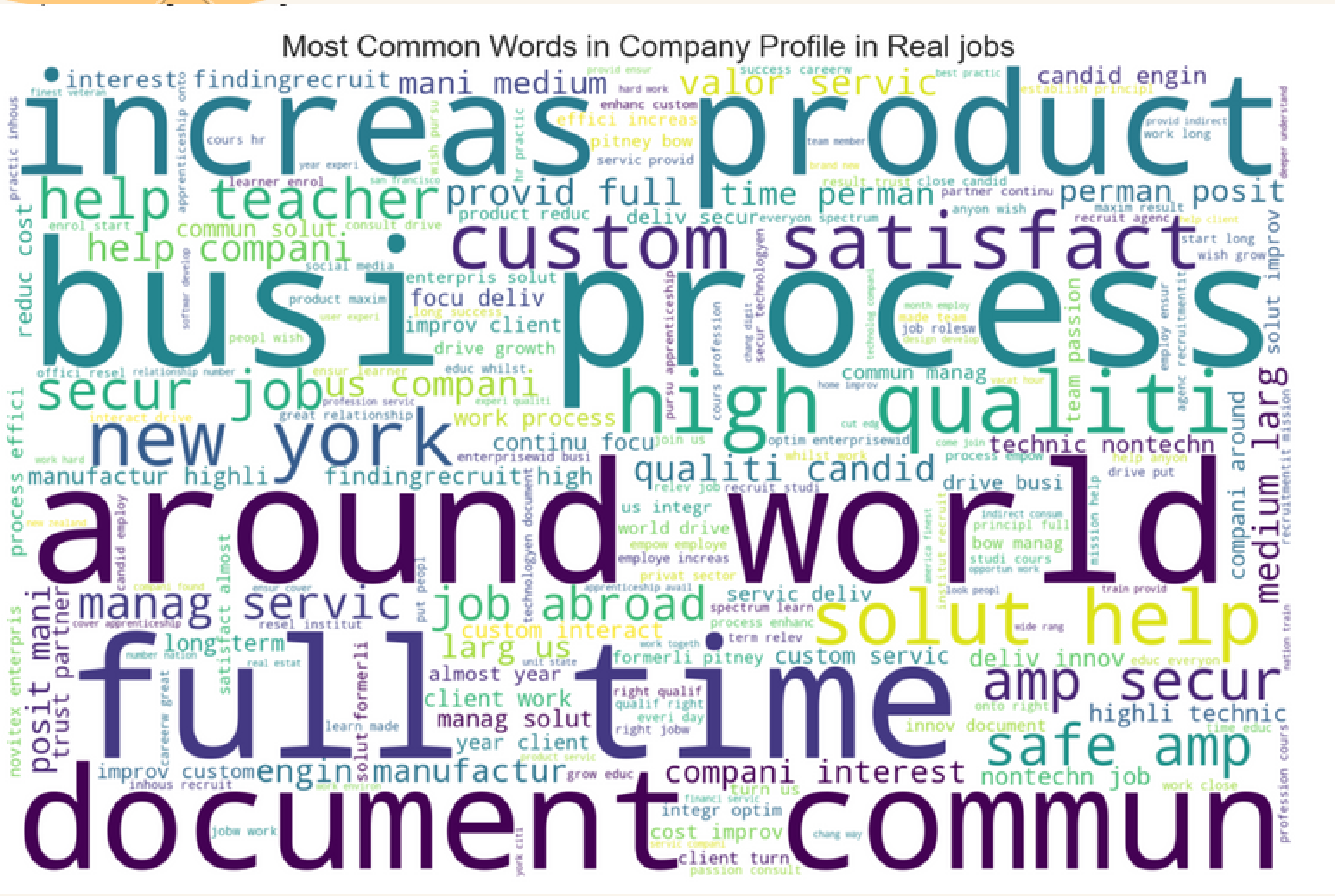
EDA


Most Common Words in Benefits in Fake jobs
onlin train
worklif balanc
benefit packag
work environ
train provid
offer full
posit offer
special skill
data entri
entri level
start work
level posit
complet train
entri complet
train need
start dont
provid start
full onlin
opportun pass
pass byposit
easi start
work easi
startedearn much
let opportun
special
need

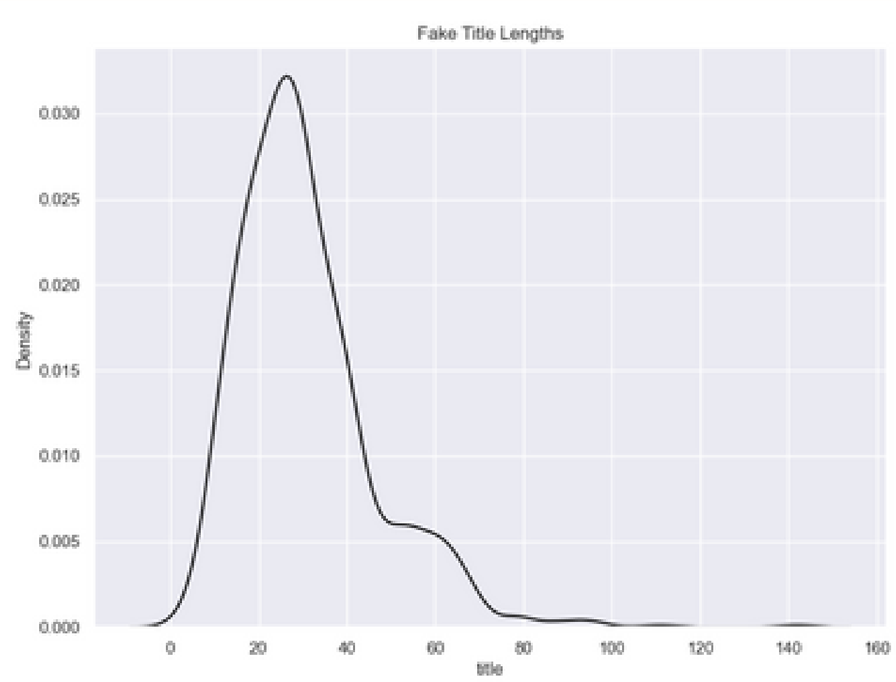

Most Common Words in Benefits in Real jobs
health dental
dental vision
equal opportun
see job
work environ
competit salari
medic dental
benefit packag
offer competit
job descript
stock option
paid holiday
month employ
paid time
comprehens benefit
health insur
packag fulltim
life insur
base salari
fulltim employe

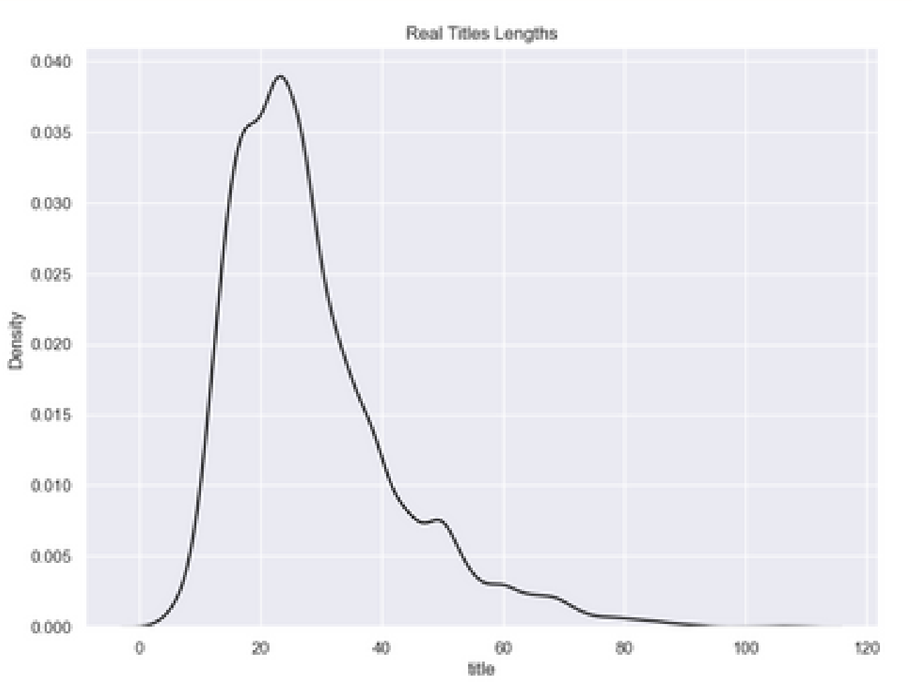
EDA


Most Common Words in Company Profile in Fake jobs
sign bonu
aptitud staf
refin resourc
candid enjoy
employe receiv
compens packag
year access
servic town
staf solut

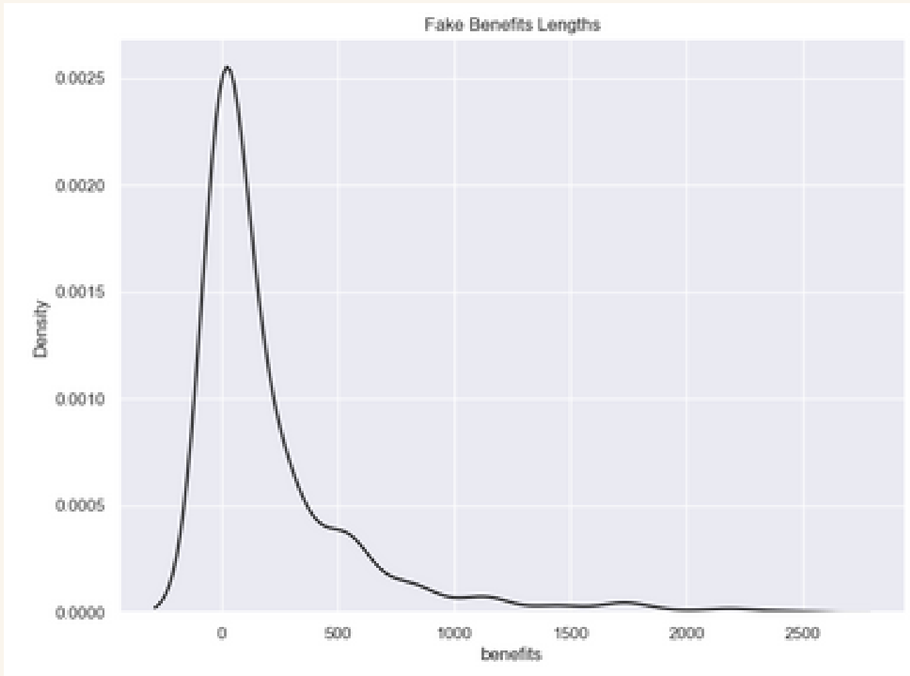

Most Common Words in Company Profile in Real jobs
increas product
busi process
around world
full time
document commun
custom satisfact
high qualiti
new york
solut help

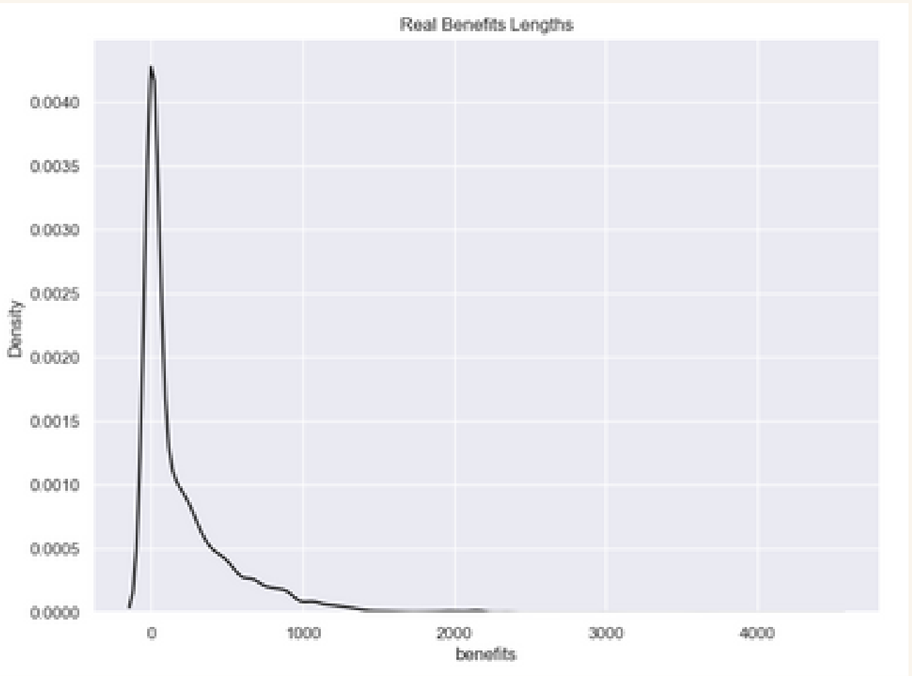
EDA


Fake Title Lengths
Density
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
100
120
140
160
title

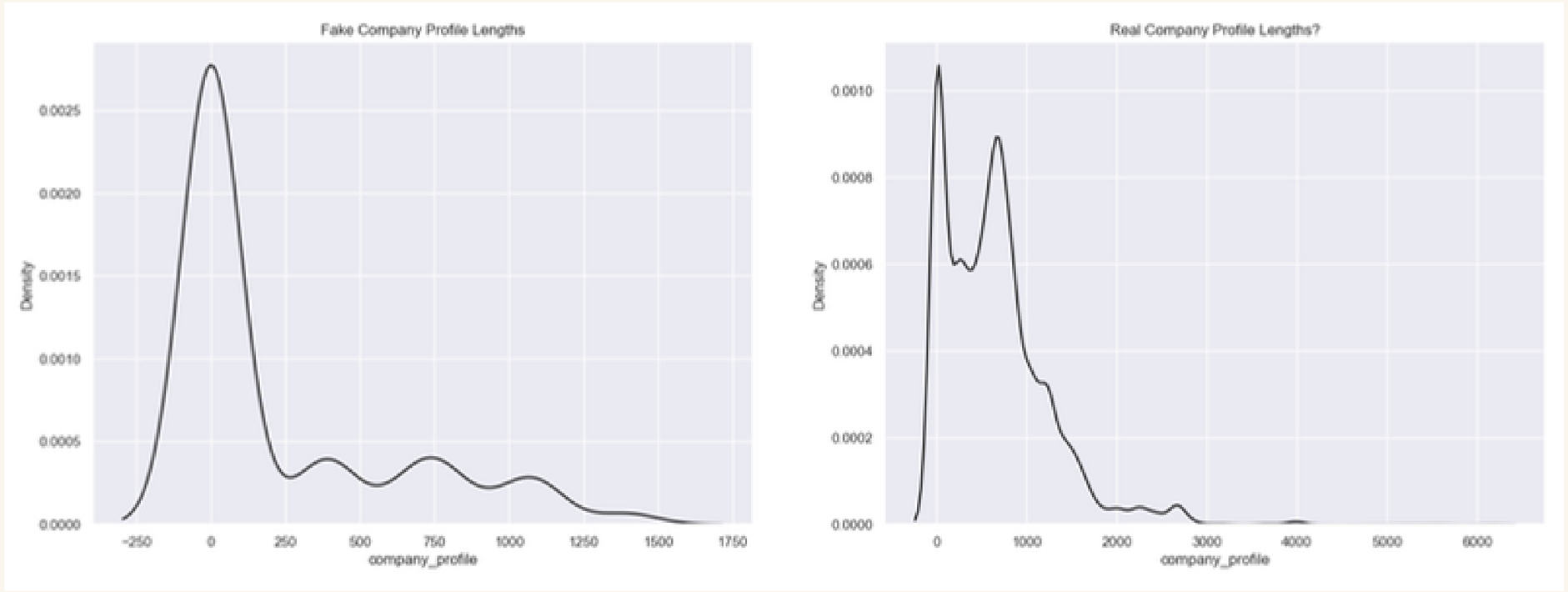

Real Titles Lengths
Density
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
100
120
title

EDA


Fake Benefits Lengths
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
500
1000
1500
2000
2500
benefits

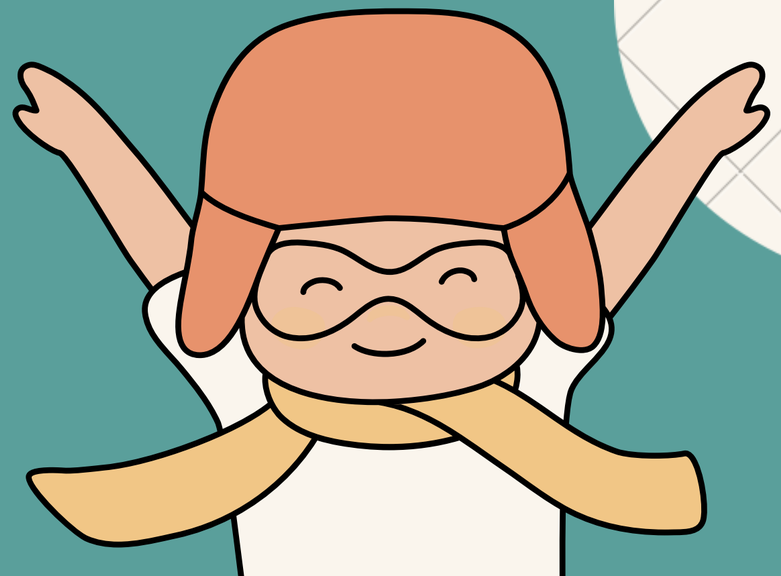

Real Benefits Lengths
Density
0.0040
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
1000
2000
3000
4000
benefits

EDA


Fake Company Profile Lengths
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
-250
0
250
500
750
1000
1250
1500
1750
company_profile
Real Company Profile Lengths?
Density
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
0
1000
2000
3000
4000
5000
6000
company_profile

LET'S GET TO KNOW MORE ABOUT

Models Used

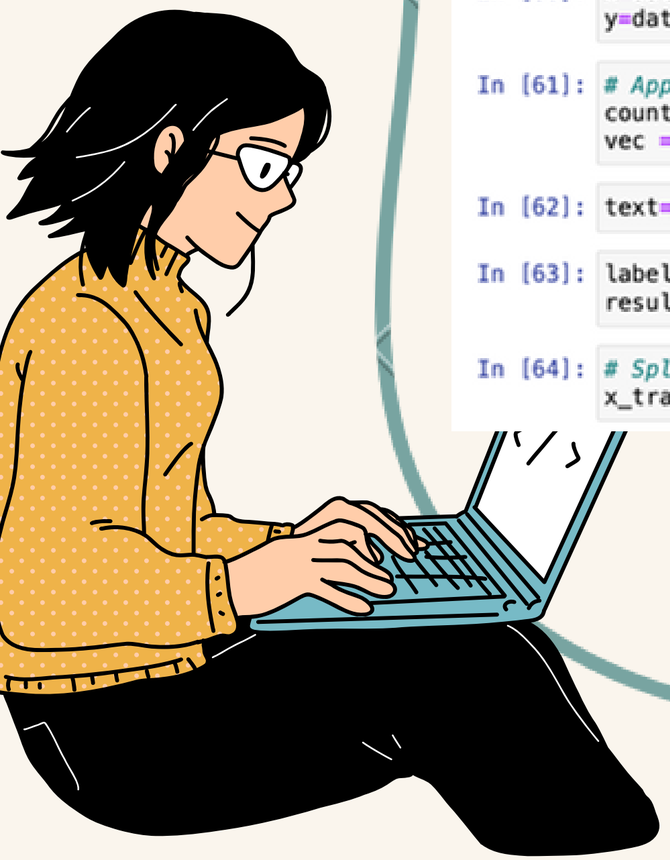


CountVectorizer technique:

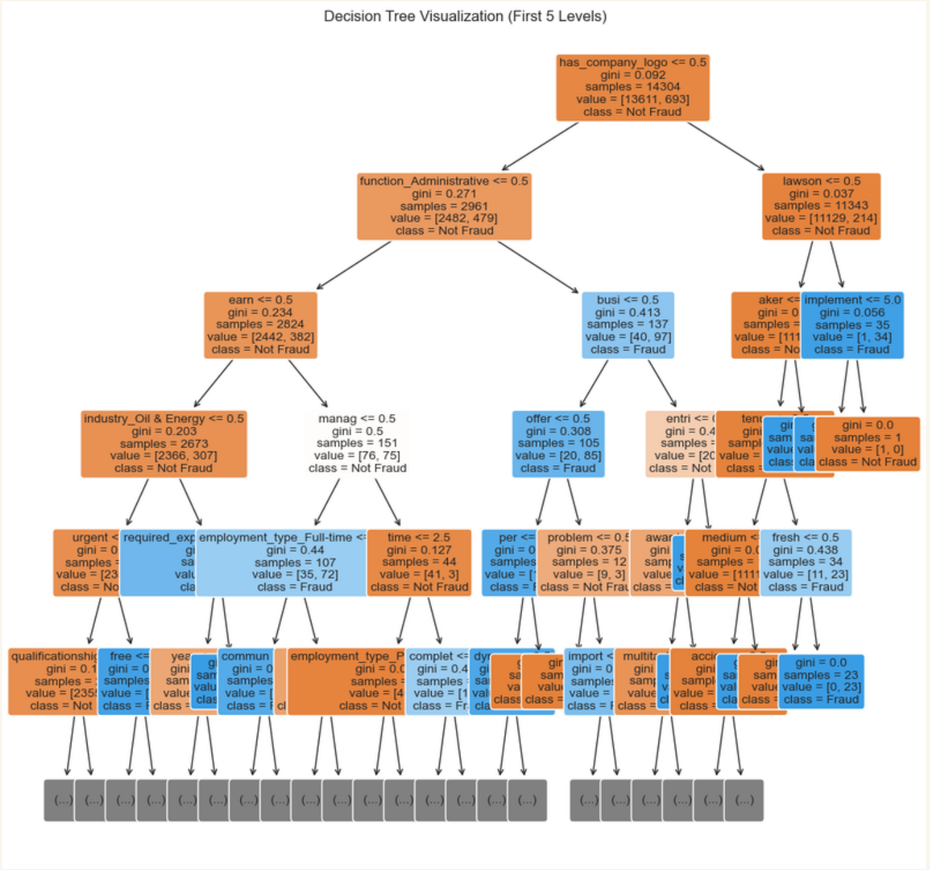
1. Tokenization
2. Vocabulary construction
3. Word counting

CountVectorizer technique:

```
In [60]: x=data.drop(columns='fraudulent')
         y=data['fraudulent']

In [61]: # Applying Count Vectorizer
         count_vec = CountVectorizer(max_features=5000)
         vec = count_vec.fit_transform(data['text'])

In [62]: text=pd.DataFrame(vec.toarray(),columns=count_vec.get_feature_names_out())

In [63]: labels=pd.get_dummies(x.drop(columns='text'))
         result = pd.concat([labels, text], axis=1)

In [64]: # Splitting data to train and test
         x_train,x_test,y_train,y_test=train_test_split(result,y,test_size=0.2,random_state=42,stratify=y)
```



Decision tree classifier



Decision tree classifier

Our findings:

Training Classification report for Decision Tree Classifier

	precision	recall	f1-score	support
0	1.00	1.00	1.00	13611
1	1.00	1.00	1.00	693
accuracy			1.00	14304
macro avg	1.00	1.00	1.00	14304
weighted avg	1.00	1.00	1.00	14304

Testing Classification Report for Decision Tree Classifier

	precision	recall	f1-score	support
0	0.99	0.99	0.99	3403
1	0.83	0.77	0.80	173
accuracy			0.98	3576
macro avg	0.91	0.88	0.89	3576
weighted avg	0.98	0.98	0.98	3576

Random forest classification

- Combines the output of multiple decision trees to reach a single result
- For classification tasks, the output of the random forest is the class selected by most trees
- Benefit in its reduced risk of overfitting the data, as feature bagging allows estimating missing values when a portion of data is missing

Random forest classification

Our findings:

Training Classification report for Random Forest Classifier

	precision	recall	f1-score	support
0	1.00	1.00	1.00	13611
1	1.00	1.00	1.00	693
accuracy			1.00	14304
macro avg	1.00	1.00	1.00	14304
weighted avg	1.00	1.00	1.00	14304

Testing Classification Report for Random Forest Classifier

	precision	recall	f1-score	support
0	0.98	1.00	0.99	3403
1	1.00	0.59	0.74	173
accuracy			0.98	3576
macro avg	0.99	0.79	0.87	3576
weighted avg	0.98	0.98	0.98	3576

Logistic regression

- widely used for binary classification
- uses the logistic function to transform a linear combination of input features into a probability value ranging between 0 and 1, which corresponds to the two categories

Logistic regression

Our findings:

Training Classification report for Logistic Regression

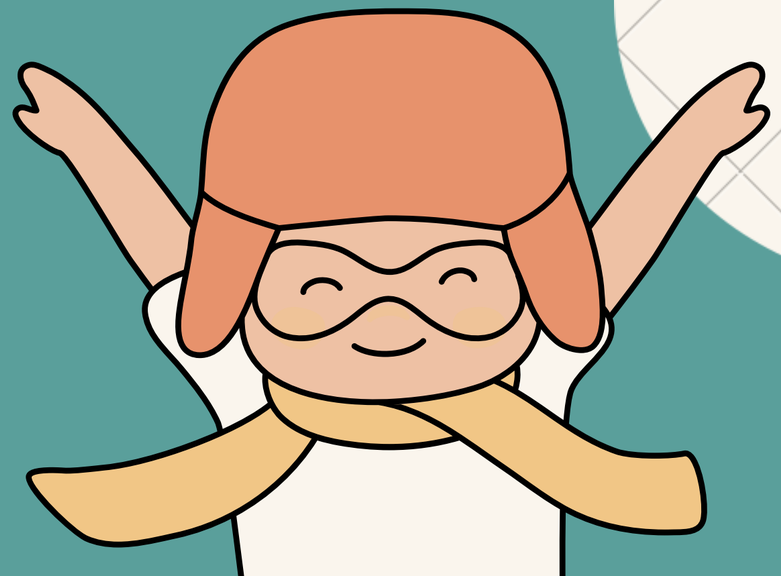
	precision	recall	f1-score	support
0	1.00	1.00	1.00	13611
1	1.00	0.99	1.00	693
accuracy			1.00	14304
macro avg	1.00	1.00	1.00	14304
weighted avg	1.00	1.00	1.00	14304

Testing Classification Report for Logistion Regression

	precision	recall	f1-score	support
0	0.99	1.00	0.99	3403
1	0.90	0.80	0.85	173
accuracy			0.99	3576
macro avg	0.95	0.90	0.92	3576
weighted avg	0.99	0.99	0.99	3576

LET'S GET TO KNOW MORE ABOUT

Results & Conclusion



Model comparisons:

Decision tree

	f1-score
accuracy	0.98
macro avg	0.89
weighted avg	0.98

Random forest

	f1-score
accuracy	0.98
macro avg	0.87
weighted avg	0.98

Logistic regression

	f1-score
accuracy	0.99
macro avg	0.92
weighted avg	0.99

Model comparisons:

- All the models used are of similar accuracy
- Best among the 3 different ones:

  Logistic Regression Model
- By comparing the different f-score values, LR model has the highest accuracy compared to the others

Our data-driven insights

The variables of the job posting has no strong correlation as to whether the job is fraudulent or not

Conclusion:

- By incorporating our ML findings, job-posting websites can more efficiently crack down on the prevalence of fake job postings
- Information and identity of individuals can be better protected from scammers and parties with ill intentions

Thank you!

Athlete

An athlete is a person trained in or experienced in exercises, sports, or games that require strength, agility, or endurance.

Who wanna be a professional athlete?

Wow! They're amazing!

There are many more professions to be your future dream job!

STUDY HARD AND...

Reach your Dream Job!