BIG DATA MANAGEMENT

PROJECT ONE

Abstract

In this project I have familiarized myself with apache spark and also made use of pyspark (the python variant). Finally, I have used machine learning to build models for a classification task.

Ngomba Litombe

01/06/2020

Sn1199003

1. I downloaded and installed both spark and pyspark
2. I registered in Kaggle and downloaded the real/fake job data.
3. After ensuring that all the installations were properly done, I wrote scala code using notepad to create a file called Scala_stats.scala (which is part of the deliverables).
   Ran spark-shell in command line and then to run the file, the command
   : load Scala_stats.scala

```
Spark context Web UI available at http://DESKTOP-9KU8FJ9:4040
Spark context available as 'sc' (master = local[*], app id = local-1591187392357).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.4.5
      /_/

Using Scala version 2.11.12 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_241)
Type in expressions to have them evaluated.
Type :help for more information.

scala> :load Scala_stat.scala
```

The following results were obtained:

a) Number of lines in the csv file = 17880
b) Number of fake job postings = 866
c) Number of real job postings =17014
d) Top 10 most required education in fake job postings

required_education	count(required_education)
High School or eq...	170
Bachelor's Degree	100
Unspecified	61
Master's Degree	31
Some High School ...	20
Certification	19
Associate Degree	6
Professional	4
Some College Cour...	3
Doctorate	1

e) Top 10 most required education in real job postings

required_education	count(required_education)
Bachelor's Degree	5045
High School or eq...	1910
Unspecified	1336
Master's Degree	385
Associate Degree	268
Certification	151
Some College Cour...	99
Professional	70
Vocational	49
Doctorate	25

4. Analytics in Pyspark

In this section, I used pyspark with jupyter notebook.
Firstly, all rows with zero values are not considered in the following computations.

f) Average maximum salary in fake job postings = 159218.71
Standard deviation of maximum salary in fake job postings = 635164.98

g) median of the minimum salary in the range in real job postings = 36000.0

h) after sorting the data for g) above and taking the mid index as the median, I got a median of 35000 different from that above.
Looking at the data carefully, I discovered the following outliers:

- Dates are present in the data instead of actual values in some entries
- There exist unusually high values (>7 figures)
- There exist very low salaries which is not a normal occurrence (<4 figures)

After removing the outliers recomputing f) and g) again produces the following results:
f)
- mean = 54056.387
- standard deviation = 37538.975

g) computed median of minimum salary = 40000.0
median after sorting = 40000

i) I computed the most popular bigrams and trigrams in the **description** and **requirements** fields.
Most popular bigrams for real job postings

```
Out[158]: [('of:the', 13993),
           ('in:the', 12341),
           ('will:be', 10881),
           ('in:a', 10862),
           ('ability:to', 10653),
           ('you:will', 9466),
           ('we:are', 8426),
           ('to:work', 7898),
           ('looking:for', 7727),
           ('experience:in', 7716)]
```

Top 10 most popular trigrams for real job postings

```
[('you:will:be', 3663),
 ('are:looking:for', 3513),
 ('be:able:to', 3366),
 ('the:ability:to', 3295),
 ('we:are:looking', 3214),
 ('as:well:as', 3183),
 ('looking:for:a', 2902),
 ('be:responsible:for', 1865),
 ('ability:to:work', 1756),
 ('to:join:our', 1683)]
```

Top 10 most popular bigrams for fake job postings

```
[('of:the', 623),
 ('ability:to', 598),
 ('in:the', 540),
 ('to:work', 508),
 ('in:a', 443),
 ('we:are', 437),
 ('to:the', 379),
 ('able:to', 364),
 ('looking:for', 361),
 ('for:the', 319)]
```

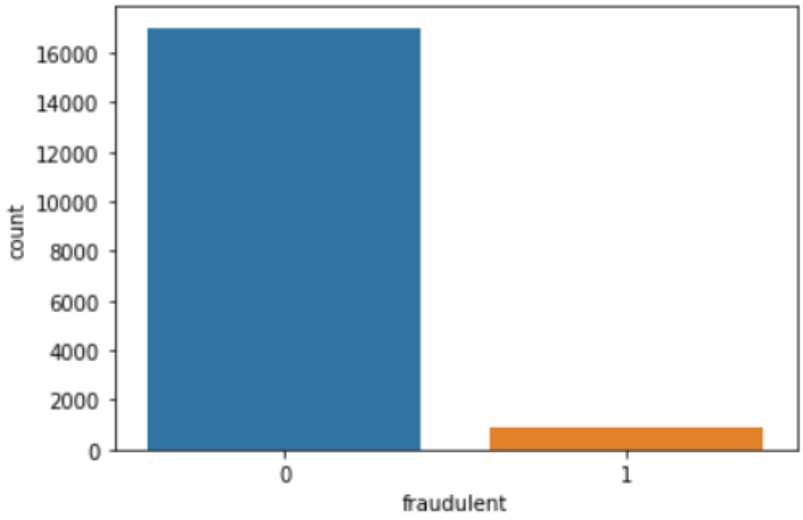
Top 10 most popular trigrams for fake job postings

```
[('are:looking:for', 202),
 ('oil:and:gas', 165),
 ('we:are:looking', 164),
 ('be:able:to', 161),
 ('ability:to:work', 132),
 ('looking:for:a', 122),
 ('and:gas:industry', 115),
 ('able:to:work', 112),
 ('the:ability:to', 99),
 ('we:are:seeking', 98)]
```

Question 5

The aim here is to build a machine learning model that distinguishes between real and fake job postings.

j) I carried out a classification task of predicting whether a job posting is fake or real based only on the telecommuting feature using naïve bayes.
At the start I noticed the class data is high imbalance.



Simply throwing this into a naive bayes classifier and using 5-fold cross validation. In one of the folds the following results was obtained.

	precision	recall	f1-score	support
0	0.97	0.96	0.96	3460
1	0.06	0.09	0.07	116
accuracy			0.93	3576
macro avg	0.52	0.52	0.52	3576
weighted avg	0.94	0.93	0.93	3576

It can be seen that due to the class imbalance, the recall and precision of the minority class is very low. This means that the classifier is bias towards the majority class.

Using two more attributes ("has_company_logo" and "has_questions") for predictions with cross validation using two different classifiers (k nearest neighbors and random forest). The following results were obtained in one of the iterations for the two difference classifiers.

K nearest neighbors

	precision	recall	f1-score	support
0	0.98	1.00	0.99	3514
1	0.00	0.00	0.00	62
accuracy			0.98	3576
macro avg	0.49	0.50	0.50	3576
weighted avg	0.97	0.98	0.97	3576

Random Forest

	precision	recall	f1-score	support
0	0.97	1.00	0.98	3460
1	0.00	0.00	0.00	116
accuracy			0.97	3576
macro avg	0.48	0.50	0.49	3576
weighted avg	0.94	0.97	0.95	3576

	precision	recall	f1-score	support

Observation

Still like I mentioned above, there is bias towards the minority class by the classifiers as can be seen form the precision and recall values.

k) Making predictions using the description attribute
Since the description parameter is text, I began with some processing;

-remove punctuations
-remove numbers
- reduce to lower case
- lemmatization (converting words to their corresponding lemmas in preparation for vectorization)
- vectorization using TF-IDF and an n-gram range of 2-3 in order to retain some meaning.

Using the SVM (support vector machines) classifier linearSVC, with 5-fold cross validation. In the best fold, the following result was obtained.

	precision	recall	f1-score	support
0	0.99	1.00	0.99	3455
1	1.00	0.66	0.80	121
accuracy			0.99	3576
macro avg	0.99	0.83	0.90	3576
weighted avg	0.99	0.99	0.99	3576

	precision	recall	f1-score	support

Using XGboost classifier

	precision	recall	f1-score	support
0	0.99	1.00	0.99	3514
1	0.91	0.16	0.27	62
accuracy			0.99	3576
macro avg	0.95	0.58	0.63	3576
weighted avg	0.98	0.99	0.98	3576

Decision Tree classifier

	precision	recall	f1-score	support
0	0.99	0.99	0.99	3455
1	0.79	0.78	0.78	121
accuracy			0.99	3576
macro avg	0.89	0.88	0.89	3576
weighted avg	0.99	0.99	0.99	3576

My observation on this subsection is that despite the different model used, they all performed quite well as the minority class was recognized. My conclusion is, in natural language processing (NLP), text preprocessing is the most important activity.

Experimenting further with other features (code for this section is file with name question5k2)
Features in use

- Description attribute (vectorized with 200 n-grams)
- Benefits (present or absent)
- Telecommuting
- Has_company_logo
- Has_questions
- Employment_type (one hot encoded)
- Required_education (one hot encoded)

All the above features were placed in one big matrix in suitable format and fed into classifiers and tested using 5-fold cross validation.

Performance of one of the folds with linearSVC

	precision	recall	f1-score	support
0	0.98	1.00	0.99	3514
1	0.31	0.08	0.13	62
accuracy			0.98	3576
macro avg	0.65	0.54	0.56	3576
weighted avg	0.97	0.98	0.98	3576

Performance using XGboost classifier

	precision	recall	f1-score	support
0	0.98	1.00	0.99	3514
1	0.70	0.11	0.19	62
accuracy			0.98	3576
macro avg	0.84	0.56	0.59	3576
weighted avg	0.98	0.98	0.98	3576

Observation and conclusion
Despite the fact that I invested more effort into fixing up more complex features for the training of the models, I registered no reasonable improvements in performance. This tells me that sometimes the simplex solution works best (**Occam's razor**).