Foundation of Data Science: Project Report

Rana Al-rubaye

Data balance

- **Up-sample:** is the process of randomly duplicating observations from the minority class in order to reinforce its signal.
- **Down-Sample:** randomly removing observations from the majority class to prevent its signal from dominating the learning algorithm.
- **Note:** Up and Down sampling give same accuracy while down sampling faster since we have less samples. We offer both options in our solution but we set default to down-sample.

Example of balancing the dataset using Down-sample

0	**6787**
1	**365**

We random remove samples from majority that match the number of samples in minority samples.

1	**365**
0	**365**

```
df_majority = data[data.fraudulent == 0]
df_minority = data[data.fraudulent == 1]
  # Downsample majority class
  df_majority_downsampled = resample(df_majority,
                eplace=False,  # sample without replacement
                n_samples=df_minority.shape[0],  # to match minority class
                random_state=123) # reproducible results
  # Combine minority class with downsampled majority class
  df_sampled = pd.concat([df_majority_downsampled, df_minority])
```

Preprocessing- TF-IDF (top 45 words)

- We have text data in columns (title, location, description, and requirements), we need to make all dataset as numeric data.
- Apply TF-IDF on the text data and pick top 45 words. I tried different top from top10 - top >200 when I go over 45 noise increase.

	title	location	description	requirements	telecommuting	has_company_logo	has_questions
3951	Sr Network Engineer	US, PA, Mechanicsburg	This position is prim...	Bachelor of Science degree (EE, CS, etc.) (both a degr...	0.00000	0.00000	0.00000
5089	Finance Analyst	GB, , London	About the Company...	Your profileYou are a motivated, optimistic and analyti...	.00000	1.00000	0.00000
3837	English Teacher Abroad	US, WI, Milwaukee	Play with kids, get p...	University degree required.\xa0TEFL / TESOL / CELTA ...	.00000	1.00000	1.00000
7456	Sales Representative	US, VA, Hampton	Westview Financial S...	The qualifications for this position are:• A minimum of ...	.00000	0.00000	0.00000
6819	Product Analyst	DE, BE, Berlin	We are looking for a ...	Experience writing SQL queries in a professional settin...	.00000	1.00000	1.00000
7772	PPC Analyst	GB, CMD, London	Has your paid searc...	Someone who constantly strives to learn new skills an...	.00000	1.00000	0.00000
5681	Oracle Apps Techno-Functional Consultant	US, CT, Bloomfield	Technical/Functional...	Skill \xa0\xa0 \xa0FSCM,RICE ,PL/SQL,OBIEE,XML.Locat...	.00000	1.00000	0.00000
8146	Trainee German Content Analyst	GB, CMD, London	Locaria is a full servi...	KEY SKILLS:\xa0- Must be fluent in German and Englis...	.00000	1.00000	0.00000
8375	Superstar (Store Manager) - Oakbrook Center	US, IL, Oak Brook	Ready to let your sta...	Our Superstars possess ... 3-5 years management exp...	.00000	1.00000	1.00000
7995	Placement Coordinator	CA, AB, Edmonton	Discuss camp place...	There is a 1 week on, 1 week off on-call component to ...	.00000	1.00000	1.00000
7465	Senior Sales Executive - West	US, CA, San Francisco	Apcera is revolutioni...	Strong competitive spirit and attention to detailStrong ...	.00000	1.00000	1.00000
4638	Data Intern - Retail & Apparel Analysis	GB, LND, London	About EDITDEDITD'...		.00000	1.00000	1.00000
5272	Visual Designer	US, CA, Palo Alto	The Senior Visual De...	Someone who wants to influence your own developme...	.00000	1.00000	0.00000
511	Senior Account Manager	NZ, , Wellington	We're looking for an...	Well networked with senior executives in New Zealand...	.00000	1.00000	1.00000
8334	Account Executive	US, VA, Herndon	We are a global net...	. RequirementsB.A. or B.S. Degree.3 years/above exp...	.00000	1.00000	1.00000
4478	Senior Designer	US, ,	#URL_6123c7dc036f...	We're looking for individuals who can:Designing fun, t...	.00000	1.00000	1.00000
7921	Title Closer	US, AL, Tuscaloosa	Looking for a chang...	Are you a top Closer in your market? Are you a go-get...	.00000	1.00000	1.00000
1517	Senior PHP Developer	NZ, N, Auckland	Vend is growing - bi...	We're particularly looking for people who:Have a mini...	.00000	1.00000	1.00000
4921	Data Scientist	MT, ,	What is Casumo?\xa...	You areMotivated, driven, loyal and analytical.Educate...	.00000	1.00000	0.00000
710	Mobile Product Manager		Adcash is looking for...	Deep knowledge of the mobile landscape, devices/ope...	0.00000	1.00000	0.00000
926	Beauty & Fragrance consultants needed	GB, , Nottingham	Luxury beauty &...		0.00000	1.00000	0.00000

data

Format: %.5f

	telecommuting	has_company_logo	has_questions	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4
4307	0.00000	1.00000	1.00000	.55923	0.43496	0.31068	0.62137	0.12427
3921	0.00000	1.00000	1.00000	-0.15713	0.94281	0.07857	-0.15713	0.23570
224	0.00000	1.00000	0.00000	0.40973	0.79669	0.09105	-0.31868	-0.29591
6940	0.00000	1.00000	1.00000	-0.91258	0.35905	0.07480	-0.07480	0.16456
773	0.00000	1.00000	1.00000	-0.20294	0.77119	0.24353	-0.44648	0.32471
2168	0.00000	1.00000	0.00000	-0.70493	0.54828	0.07833	-0.31330	0.31330
581	0.00000	1.00000	1.00000	-0.72060	0.60801	0.02252	-0.29275	-0.15763
1831	0.00000	1.00000	0.00000	-0.60268	0.40179	0.26786	-0.60268	-0.20089
8838	0.00000	1.00000	1.00000	0.28868	0.00000	0.28868	-0.28868	0.86603
328	0.00000	1.00000	1.00000	0.14780	0.93606	0.19707	-0.24633	-0.04927
3919	0.00000	1.00000	0.00000	-0.24905	0.42694	0.28463	-0.24905	0.78272
3728	0.00000	0.00000	0.00000	-0.67568	0.70711	-0.06285	-0.18856	0.06285
5463	1.00000	1.00000	0.00000	-0.49344	0.77540	-0.21147	-0.28196	0.17623
302	0.00000	1.00000	1.00000	-0.55182	0.80651	0.21224	0.00000	0.00000
8377	0.00000	1.00000	1.00000	0.00000	-0.28868	0.28868	-0.28868	0.86603
5068	0.00000	1.00000	1.00000	-0.54477	0.41506	0.72635	-0.05188	0.02594
5075	0.00000	1.00000	0.00000	-0.71189	0.67375	0.19068	0.03814	0.03814
7396	0.00000	1.00000	0.00000	-0.46756	0.84746	-0.08767	0.02922	0.23378
1769	0.00000	1.00000	1.00000	-0.74763	0.60745	0.23363	0.09345	-0.09345
8063	0.00000	1.00000	0.00000	-.29374	0.91058	-0.26436	0.11749	0.02937

Preprocessing (1-2)

- Split the columns that has text from columns that has number

```
# Take only columns that has number
X_clean = data.drop(self._pp_colums, axis=1)
self._shiftCols = X_clean.shape[1]
# Take only columns that has text
X_pre = data[self._pp_colums]
textData = []
for index, row in X_pre.iterrows():
    text = "{} {} {} {}".format(row['title'],
                                row['location'],
                                row['description'],
                                row['requirements'])
    textData.append(text)
documents = pd.DataFrame(textData, columns=['headline_text'])
```

Preprocessing (2-2)

- Apply TF-IDF on columns that has text. Then combine generated numeric columns with existing numeric columns in dataset

```
# create the TD-IDF transform
if self.vectorizer == None:
   self.vectorizer = TfidfVectorizer(stop_words='english', norm='l2', use_idf=False, smooth_idf=False)
   # apply TF-IDF on training
   vectors = self.vectorizer.fit_transform(documents["headline_text"])
else:
   # apply TF-IDF on testing
   vectors = self.vectorizer.transform(documents["headline_text"])
# get the numberic data from TD-IDF as 2D array
data_pro = pd.DataFrame(vectors.toarray(), columns=self.vectorizer.get_feature_names())
# create new N columns in datase, where _numberOfTopics is number of  words that we want to pick
default set to 45
for col in range(data_pro.shape[1]):
   X_clean["Topic_{}".format(col)] = np.nan
   if col > self._numberOfTopics:
       break
# merge the data of N words of TD-IDF into dataset
for row in range(data_pro.shape[0]):
   for col in range(data_pro.shape[1]):
       X_clean.iloc[row, self._shiftCols + col] =  data_pro.iloc[row, col]
       if col > self._numberOfTopics:
          Break
```

Normalize the data

- Put all data in same scale using min/max scale

```
# normalize the data
min_max_scaler = preprocessing.MinMaxScaler()
X_norm = min_max_scaler.fit_transform(X_train)
```

	telecommuting	has_company_logo	has_questions	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4
4307	0.00000	1.00000	1.00000	-0.55923	0.43496	0.31068	0.62137	0.12427
3921	0.00000	1.00000	1.00000	-0.15713	0.94281	0.07857	-0.15713	0.23570
224	0.00000	1.00000	0.00000	0.40973	0.79669	0.09105	-0.31868	-0.29591
6940	0.00000	1.00000	1.00000	-0.91258	0.35905	0.07480	-0.07480	0.16456
773	0.00000	1.00000	1.00000	-0.20294	0.77119	0.24353	-0.44648	0.32471
2168	0.00000	1.00000	0.00000	-0.70493	0.54828	0.07833	-0.31330	0.31330
581	0.00000	1.00000	1.00000	-0.72060	0.60801	0.02252	-0.29275	-0.15763
1831	0.00000	1.00000	0.00000	-0.60268	0.40179	0.26786	-0.60268	-0.20089
8838	0.00000	1.00000	1.00000	0.28868	0.00000	0.28868	-0.28868	0.86603
328	0.00000	1.00000	1.00000	0.14780	0.93606	0.19707	-0.24633	-0.04927
3919	0.00000	1.00000	0.00000	-0.24905	0.42694	0.28463	-0.24905	0.78272
3728	0.00000	0.00000	0.00000	-0.67568	0.70711	-0.06285	-0.18856	0.06285
5463	1.00000	1.00000	0.00000	-0.49344	0.77540	-0.21147	-0.28196	0.17623
302	0.00000	1.00000	1.00000	-0.55182	0.80651	0.21224	0.00000	0.00000
8377	0.00000	1.00000	1.00000	0.00000	-0.28868	0.28868	-0.28868	0.86603
5068	0.00000	1.00000	1.00000	-0.54477	0.41506	0.72635	-0.05188	0.02594
5075	0.00000	1.00000	0.00000	-0.71189	0.67375	0.19068	0.03814	0.03814
7396	0.00000	1.00000	0.00000	-0.46756	0.84746	-0.08767	0.02922	0.23378
1769	0.00000	1.00000	1.00000	-0.74763	0.60745	0.23363	0.09345	-0.09345
8063	0.00000	1.00000	0.00000	-0.29374	0.91058	-0.26436	0.11749	0.02937

	0	1	2	3	4	5	6	7
0	1.00000	1.00000	1.00000	0.26898	0.82039	0.73251	0.69429	0.34021
1	0.00000	0.00000	1.00000	0.22034	0.76369	0.68102	0.23734	0.38667
2	0.00000	1.00000	1.00000	0.08329	0.65961	0.62674	0.46323	0.70549
3	0.00000	0.00000	0.00000	0.08400	0.69361	0.70968	0.50979	0.58152
4	0.00000	0.00000	0.00000	0.29559	0.85744	0.46274	0.32426	0.72839
5	0.00000	1.00000	0.00000	0.26269	0.65138	0.66492	0.35549	0.87448
6	0.00000	1.00000	0.00000	0.19043	0.78301	0.34561	0.62907	0.28044
7	0.00000	1.00000	0.00000	0.12110	0.61465	0.74254	0.65439	0.66865
8	0.00000	1.00000	1.00000	0.16345	0.79894	0.66001	0.56316	0.66585
9	0.00000	1.00000	0.00000	0.12170	0.64194	0.21856	0.53867	0.60932
10	0.00000	0.00000	0.00000	0.22285	0.82581	0.27126	0.64491	0.52572
11	0.00000	1.00000	1.00000	0.25601	0.83406	0.32736	0.40435	0.27546
12	0.00000	0.00000	0.00000	0.15195	0.53206	0.82853	0.30826	0.49117
13	0.00000	1.00000	0.00000	0.16488	0.44221	0.70810	0.24691	0.28950
14	0.00000	1.00000	1.00000	0.12786	0.21029	0.37576	0.55403	0.58168
15	0.00000	1.00000	1.00000	0.05779	0.57878	0.27392	0.49421	0.53900
16	0.00000	1.00000	1.00000	0.88972	0.81858	0.50048	0.33131	0.60984
17	0.00000	1.00000	0.00000	0.15408	0.74056	0.50947	0.78892	0.42773
18	0.00000	1.00000	1.00000	0.21541	0.88622	0.48158	0.37140	0.42747
19	0.00000	1.00000	1.00000	0.19568	0.73540	0.33165	0.39948	0.76198

Model Training

- I use stochastic gradient descent (SGD) learning. Also I notice similar results using different algorithms such as decision tree.
- SGD is the a good way to optimization in Machine Learning

```
def fit(self, X, y):
   print( "==Training the model ( takes up to 5 minutes )=======")
   [X_train_balance, y_balance] = textNLP.balanceData(X,y)   # balance data
   X_train = textNLP.pre_pro_Cols(X_train_balance) # pre-processing
   X_norm = min_max_scaler.fit_transform(X_train)# normalize data
   # Create stochastic gradient descent
   self.clf = SGDClassifier()
   self.clf.fit(X_norm, y_balance)
```

Feature selection

- Base on filter-base feature selection results:
  - I notice that starting Top_45 feature column and next features that generated by TF-IDF are not contributing to the model accuracy rather than they lead to noise.
  - I pick only top 45 features and ignore rest.

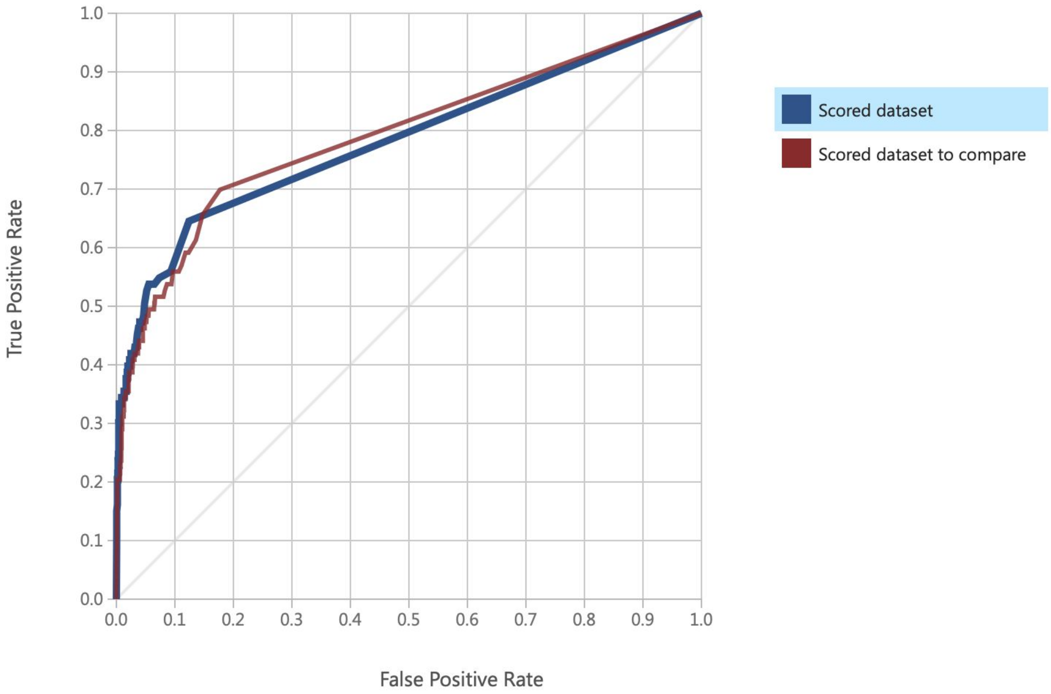
Rana Project › Filter Based Feature Selection › Features

rows	columns
1	9

	fraudulent	has_company_logo	has_questions	telecommuting	Topic_0	Topic_3	Topic_1	Topic_2	Topic_4
view as									
	1	0.256308	0.099393	0.043413	0.042841	0.029965	0.022064	0.00722	0.005666

Model Selection and Tuning

Not noticeable impact



Prediction

```
def predict(self, X):
   print("====  Testing the model =====")
   # remember to apply the same preprocessing in fit() on test data before making predictions
   X_test = textNLP.pre_pro_Cols(X)
   X_norm = min_max_scaler.fit_transform(X_test)
   return self.clf.predict(X_norm)
```

Model Evaluation

- To evaluate model, I split the dataset into 80% training and 20% testing, I find these results.

```
=============== Testing the model =====================
(Step 2 of 2) Pre-proces coloums that has text (token,  remove stop
              precision    recall  f1-score   support

           0       0.98      0.86      0.92      1685
           1       0.24      0.75      0.37       103

    accuracy                           0.85      1788
   macro avg       0.61      0.80      0.64      1788
weighted avg       0.94      0.85      0.88      1788
```

Questions?