

Job Change Prediction for Data Scientist



Group 4:

Maria Alvi
Ugochi Madumere
Aparna Rachoori
Sonakshi Sharma
Promodini Venkatakrishnan

4/29/2021

❖ **Table of Contents**

- ❖ Executive summary
- ❖ Introduction
- ❖ Metadata
- ❖ Data import staging, cleaning, and normalization
- ❖ Exploratory data models and visualization
- ❖ Model sampling, analysis, and comparison
- ❖ Business implications and recommendations
- ❖ Conclusion
- ❖ References

Executive Summary

HR Analytics plays a major role in business improvement in the world of industry especially in larger firms with a huge number of employees situated at multiple locations. HR Analytics helps the HR team understand employee characteristics and help in deducing the reasons for employees moving on to a new job leaving the current company. Employees i.e., workforce is a pillar of strength for any company. There is a risk for the growth for the company and a huge loss if the strong and innovative employees leave the company. HR Analytics gives insights about the most possible reasons and situations causing this.

For instance, if an employee has a better and satisfying job task in another company than the current company, he may choose to move on to the new company. Our project aims at this very fact. Our goal is to predict how likely an employee changes his job i.e., move on to a next job. As we are trying to classify the employees in two different categories of who is likely to change the job and who is not, Logistic Regression would be a baseline model for the current scenario. The target variable is predicted based on the independent variables i.e., gender, education, training hours etc.

The dataset is analyzed and explored for patterns within the data, prepared and modelled for a classification algorithm i.e., Logistic Regression in this case. The accuracy of the baseline Logistic Regression model for this dataset is 68. Neural networks are implemented, and accuracy is increased by almost 2% i.e., 69.7% and education level, enrolled university and experience affect the decision of the employee whether to retain in the same job or move on to another. Some code snippets and visualization graphs are added in the document to support in the explanation. A jupyter notebook (. ipynb file) with the code for data exploration, visualization and modelling will be attached with this document.

Introduction

Hiring is a time-consuming process for both the employer and employee. Generally, hiring is done based on a savvy criterion because a good employee is an asset to the growing organization. Although companies are willing to hire year-round, the number of an employee with the desired qualifications are comparatively less. Although evaluation helps in choosing the best candidate from professional websites, not everyone is willing to change jobs. Our project discusses the above-mentioned criteria. The dataset reveals about the candidates, looking for a job change, and this depends on a few criteria like experience, educational level,

demographics, and more. Identifying the predominant candidates will save both the cost and time for the employer.

Data Description

The dataset is a second-hand data obtained from the Kaggle ^[1]. By Analyzing this dataset, few clarifications can be achieved by the employer, on how to hire the best suitable candidate, who is willing to undergo the job change. The dataset contains 13 independent variables such as city, training hours, major discipline, experience, educational level, and more to determine which candidate is looking for a job change. The dependent variable is named as the ‘target’ variable in the dataset. The data dictionary is given below:

Variable Name	Definition	Datatype
enrollee_id	Provides Unique ID for each candidate.	int
city	Provides the City code	object
city_development_index	Scales tells about the city development index	float
gender	Gender of the candidate	object
relevent_experience	Whether the candidate has relevant experience or not	object
enrolled_university	Type of university enrolled for	object
education_level	Type of degree the candidate holds	object

major_discipline	Type of major at the education.	object
experience	Years of experience of a candidate	object
company_size	Company size of the current employer	object
company_type	Type of the company	object
last new job	Years between the job change	object
training_hours	Trained number of hours	int
target	0 – Not willing for job change, 1 – willing for a job change	int

Figure 1: Data Dictionary.

Data Exploration and Visualization

Data exploration is a preliminary step in data analysis, which is usually done by creating the visuals of the features/variables in the dataset. The data visualization helps in determining the basic characteristics of the data such as the size, shape, and accuracy, which can be used to understand the data better.

Summary statistics

Figure 2 shows the summary statistics of the interval variables. There are a total of 19158 number of observations. The summary statistics gives the information of basic statistical measures such as mean, count, standard deviation, minimum, maximum and the interquartile range.

	enrollee_id	city_development_index	training_hours	target
count	19158.000000	19158.000000	19158.000000	19158.000000
mean	16875.358179	0.828848	65.366896	0.249348
std	9616.292592	0.123362	60.058462	0.432647
min	1.000000	0.448000	1.000000	0.000000
25%	8554.250000	0.740000	23.000000	0.000000
50%	16982.500000	0.903000	47.000000	0.000000
75%	25169.750000	0.920000	88.000000	0.000000
max	33380.000000	0.949000	336.000000	1.000000

Figure 2: Summary Statistics of Interval Variables.

Size and Shape of the Data

The size of the shape is (19158,15) where 19158 is the number of rows and 14 is the number of columns. The shape of the data refers to the total number of observations which is 268212.

Missing Values

Figure 3 shows the summary of variables with missing values. The variables with missing values are:

1. Gender.
2. Enrolled_University.
3. Education_level.
4. Major_discipline.
5. Experience.
6. Company_size.
7. Company_type.
8. Last_new_job.

enrollee_id	0
city	0
city_development_index	0
gender	4508
relevent_experience	0
enrolled_university	386
education_level	460
major_discipline	2813
experience	65
company_size	5938
company_type	6140
last_new_job	423
training_hours	0
target	0
dtype: int64	

Figure 3: Missing Value Summary.

Correlation

Figure 4 shows the correlation matrix between the target variable and interval variables. It is clear from figure 3 that there is no strong correlation between the target and independent interval variables.

	enrollee_id	city_development_index	training_hours	target
enrollee_id	1.000000	-0.040455	0.000998	0.049475
city_development_index	-0.040455	1.000000	0.001920	-0.341665
training_hours	0.000998	0.001920	1.000000	-0.021577
target	0.049475	-0.341665	-0.021577	1.000000

Figure 4: Correlation Matrix.

Analyzing Dependent Variable

The dependent variable is denoted as “Target”, which is a categorical variable with binary output that is 0 and 1. The categorical variable is generally analyzed by calculating the frequency count. Figure 4 represents the visualization of frequency distribution of dependent variables. The frequency count of the target variable is as follows:

- 0 = 14381
- 1 = 4777

The above distribution illustrates that there is a high imbalance in the dependent variable, which needs some transformation to make it more balanced.

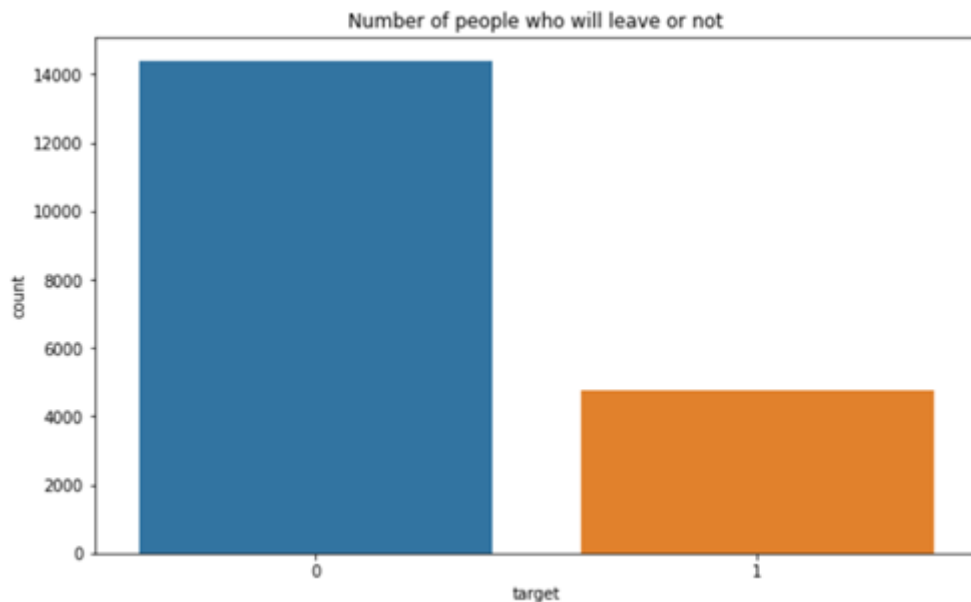


Figure 5: Frequency distribution of Dependent variable.

Analyzing Independent Variables

Categorical Variables

1. City

Variable city represents the city codes. It is a nominal variable. there are 123 unique values in this variable.

2. Gender

The variable 'Gender' is a nominal variable which is defined by three categories that are:

- Male: 17729
- Female: 1238
- Other: 191

Figure 6 illustrates the gender distribution with respect to the target variable. as it can be observed from figure 6 that more than 12000 are not going to leave the job while almost 4500

are most likely to change their job. Whereas in the categories 'Female' and 'Other' do not show much difference in changing their job or not. So, it can be concluded that the number of 'Male' who are not going to leave their job is going to have the most impact in classifying whether a data scientist would change their job or not.

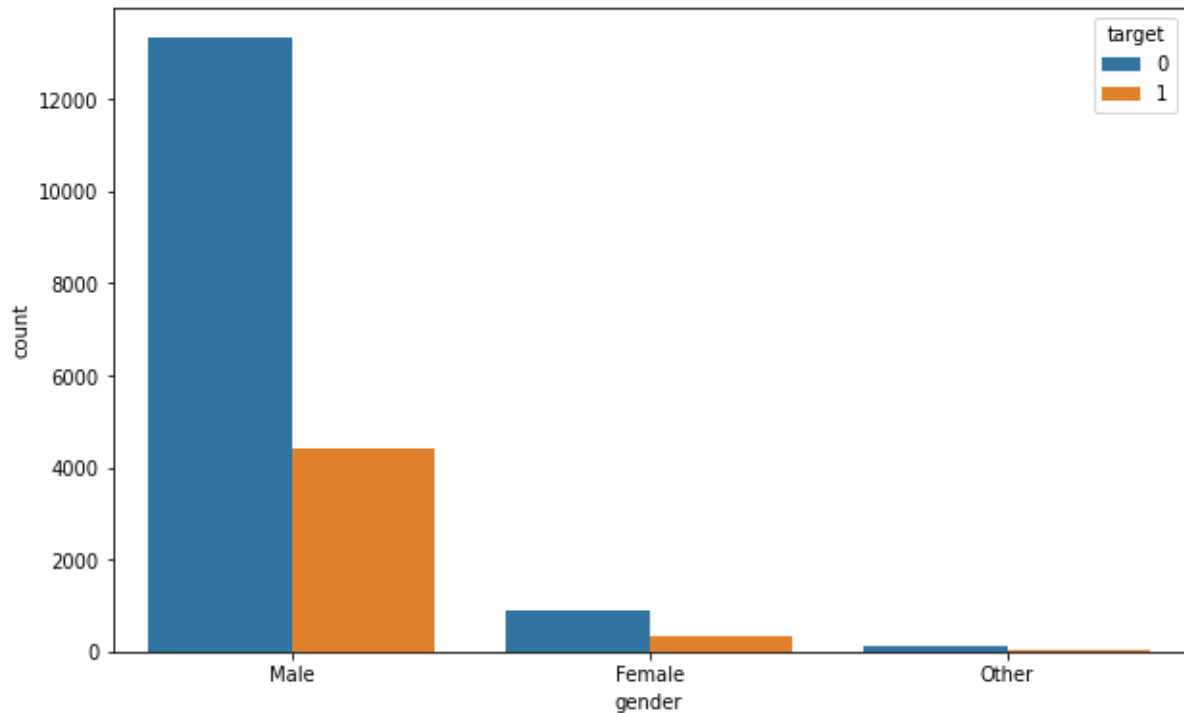


Figure 6: Gender Distribution.

3. Relevant experience

The variable relevant experience is a nominal variable with two categories that are:

- Has experience: 13792.
- No relevant experience: 5366

Figure 7 shows that people who have relevant experience are more likely to stay on their current job than people who have experience.

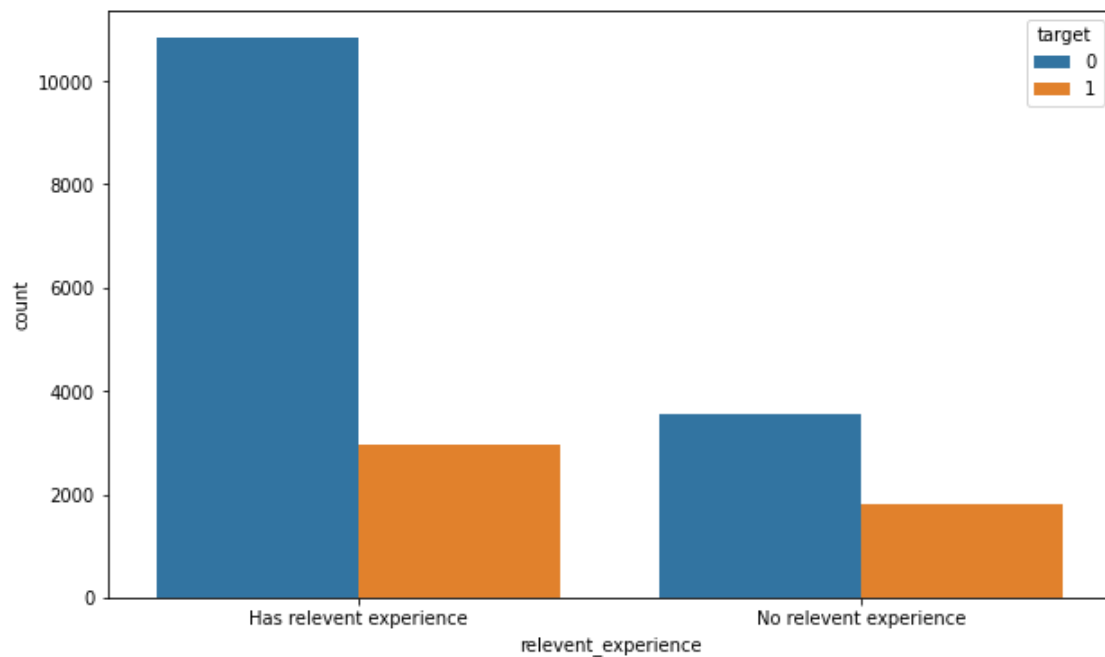


Figure 7: Relevant Experience Distribution.

4. Enrolled University

Enrolled University variable is a categorical variable which is described by three categories, they are:

- No Enrollment: 13817
- Full Time Course: 3757
- Part Time Course: 1198

Figure 8 illustrates the distribution of people who are either enrolled in a university for full time course or part time course or who are not enrolled in any university. As described by Figure 8, people who are not enrolled in any course are not going to change their job.

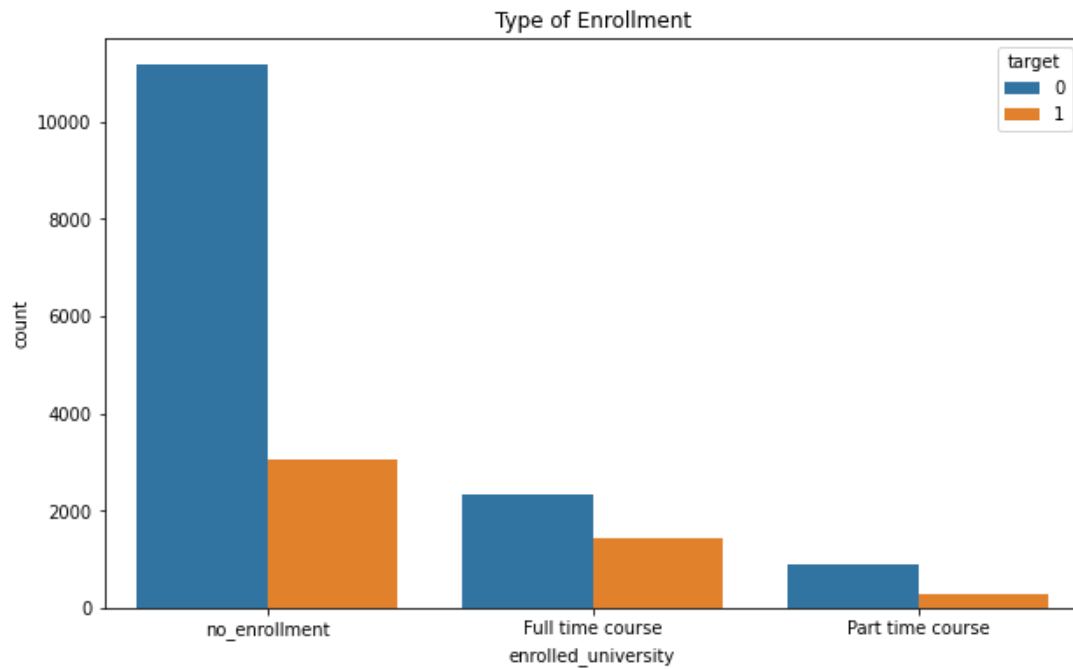


Figure 8: Enrolled in a University Distribution.

5. Education Level

Education Level is a nominal variable which describes the education background of each employee. The education level is divided into 5 categories, that are.

- Graduate: 11598
- Masters: 4361
- High School: 2017
- Phd: 414
- Primary School: 308

As shown by figure 9, more than 11,000 employees have a graduate degree and among them almost 9000 people are not going to change their job. There are very few employees who have only primary education or a PhD.

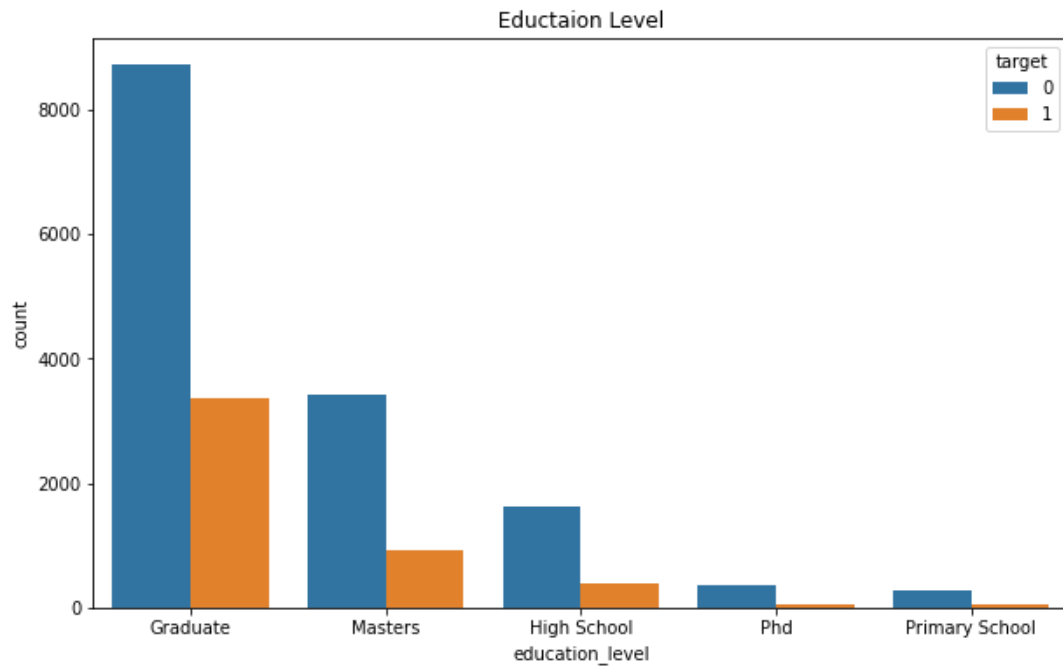


Figure 9: Education level Distribution.

6. Major Discipline

Figure 10 shows the frequency distribution plot for major discipline variable which is a nominal variable with six categories that are as follows:

- STEM: 14492
- Humanities: 669
- Other: 381
- Business Degree: 327
- Arts: 253
- No Major: 223

Majority of the employees have a STEM degree and almost 11,000 employees will not change their job while 3,000 employees are more likely to switch their job. There are very few people with no major.

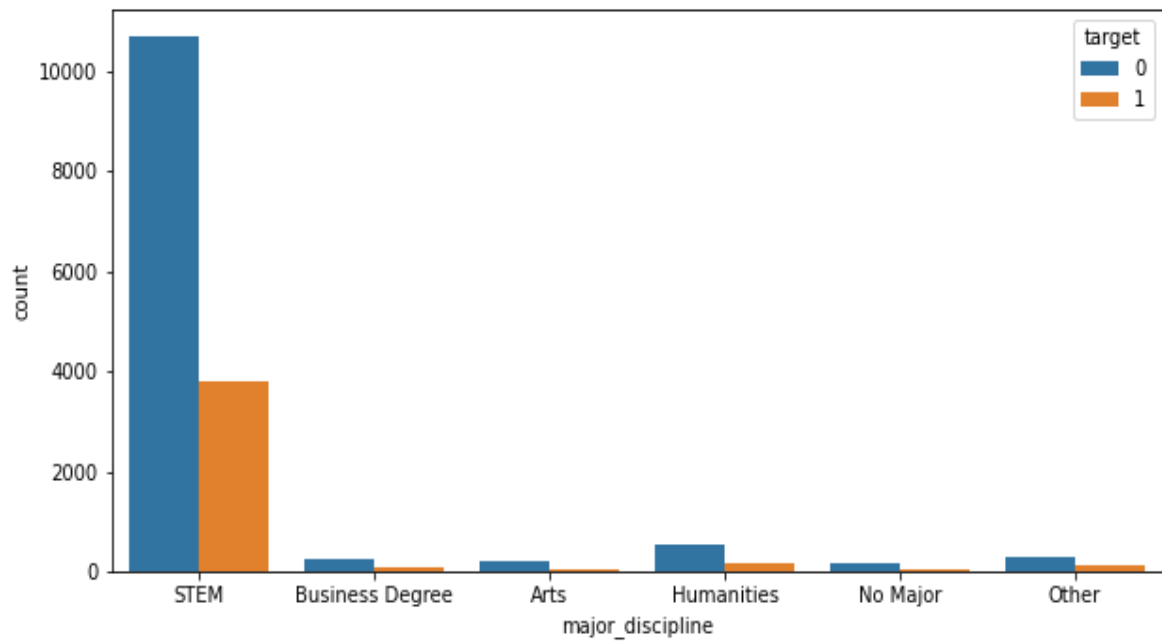


Figure 10: Major Discipline Distribution.

7. Experience

Experience is an ordinal variable which defines the years of experience of an employee. The categories range from less than 1 to greater than 20, where less than 1 means no or less than one year experience and greater than 20 is defined as more than 20 years of experience. Figure 11 describes the distribution of experience. As seen from the figure that people with more than 20 years of experience are going to stay at their current job while people with 3 to 4 years of experience are more likely to leave their current job.

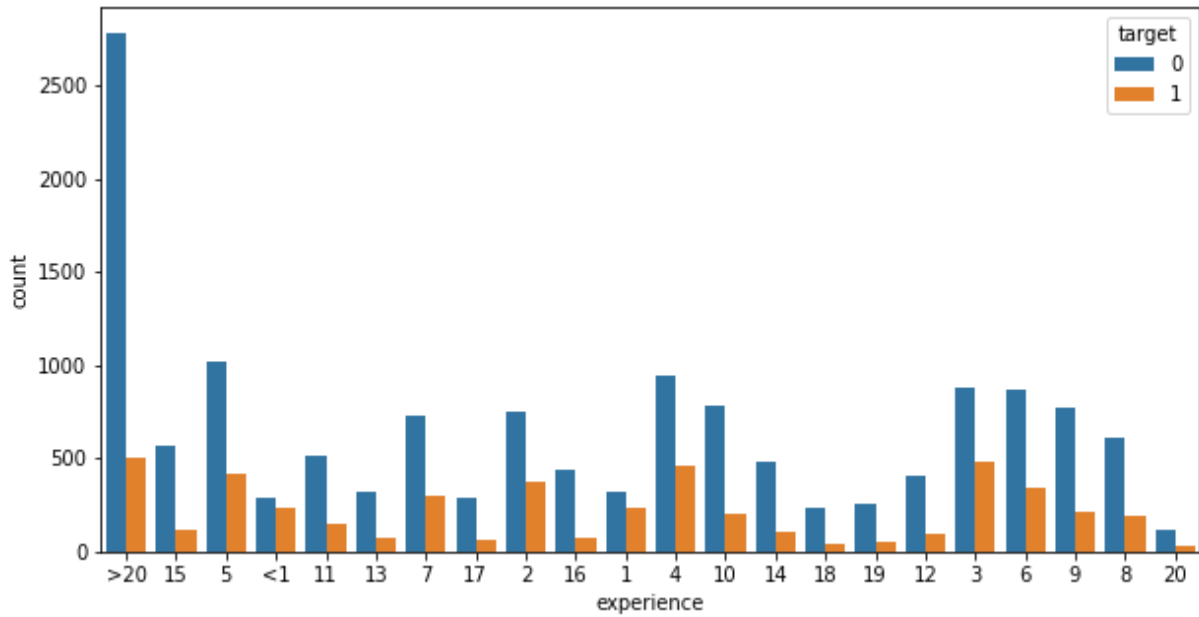


Figure 11: Experience Distribution.

8. Company Size

Company size is an ordinal variable which defines the size of the company based on the number of people. More companies have a number of employees from 50 to 99 where people are not going to change their current job. The second highest company size has a number of employees from 100 to 500.

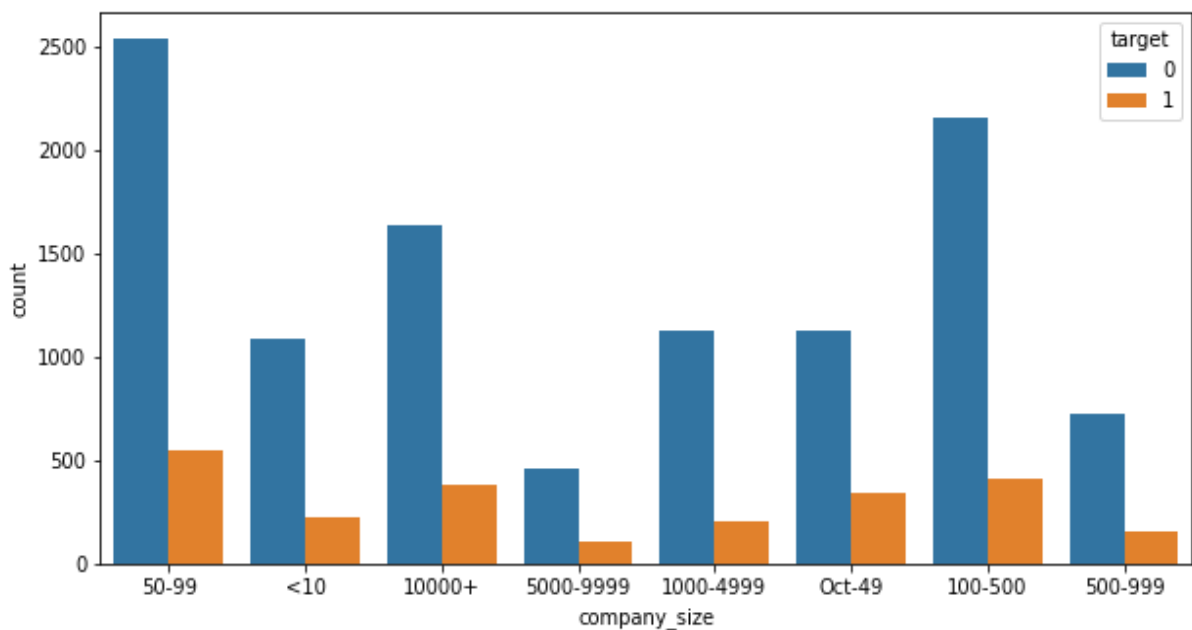


Figure 12: Company Size Distribution.

9. Company Type

Company type is a nominal variable with six categories, that are:

- Pvt Ltd: 9817
- Funded Startup: 1001
- Public Sector: 955
- Early-Stage Startup: 603
- NGO: 521
- Other: 121

Figure 13 shows a frequency distribution of company type which illustrates that there are more than 9000 private limited companies while there are very few company types which are other than the categories defined above. Also, it should be noticed that people who work in private limited companies are not going to change their job.

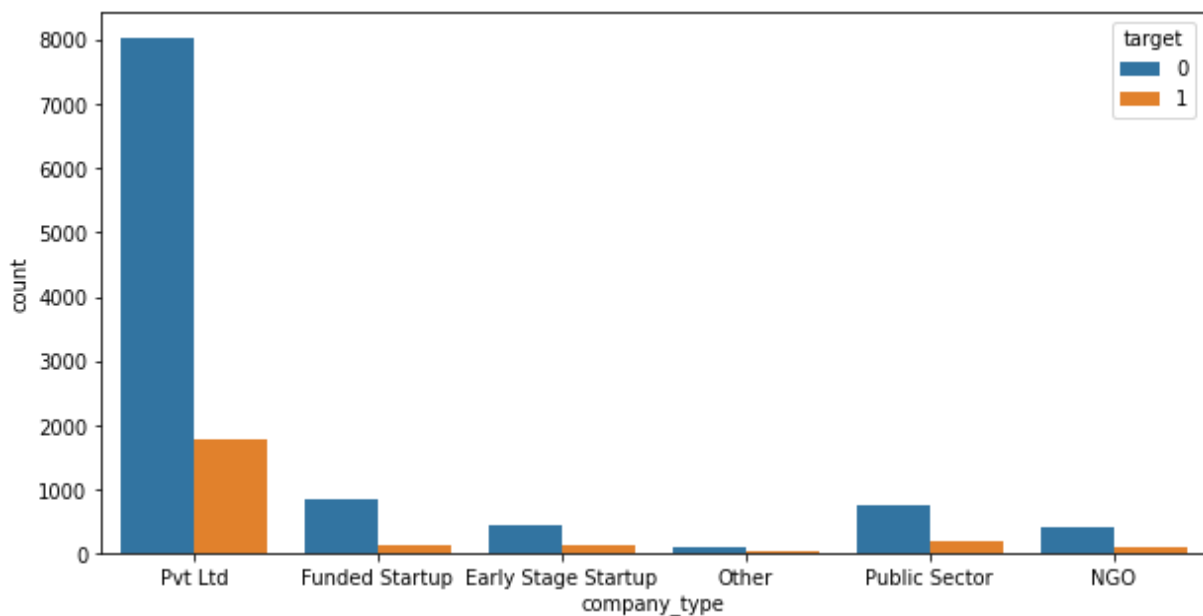


Figure 13: Company Type Distribution

10. Last New Job

Last new job is an ordinal variable which provides information about the years of the last job changed by an employee. There are six categories in this variable ranges from 0 to greater than 4 where 0 means that an employee has never changed the job and greater than 4 means an employee has changed his/her last job more than 4 years ago. As observed from figure 14 that people who had changed their last job in one year are more likely to stay at their current job.

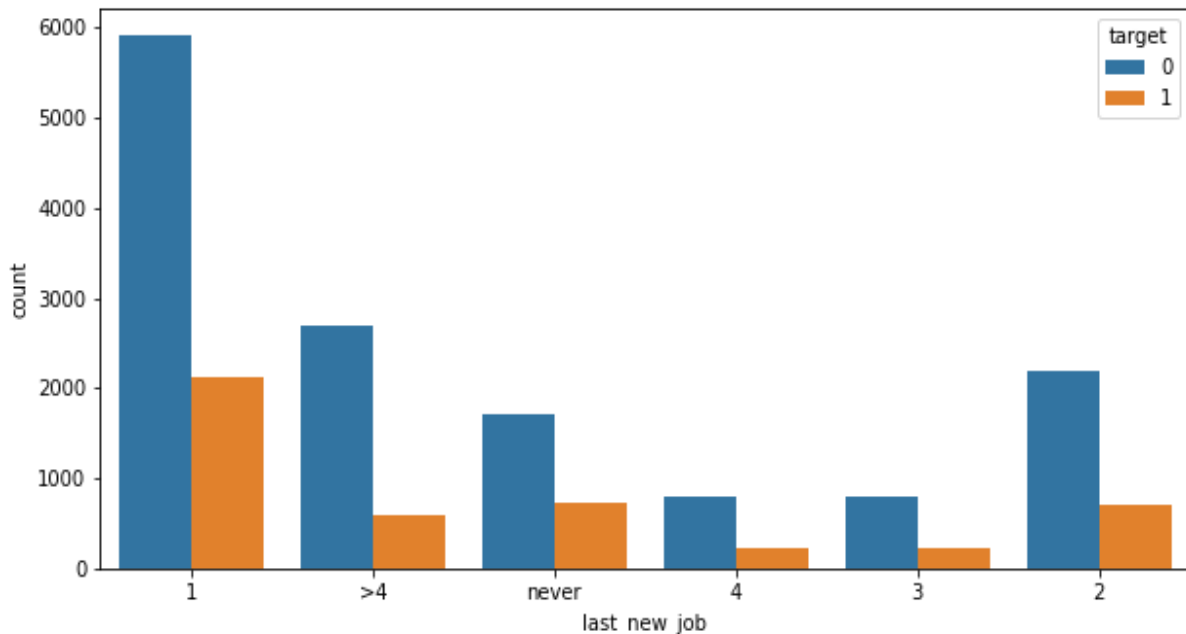


Figure 14: Last New Job Distribution.

11.City Development Index

City development index is an interval variable that has 93 unique values. Figure 15 shows a histogram of the city development index which is highly skewed towards the left. This exhibits that there are extreme outliers due to which there is no normal distribution in this variable. The best practice to observe the outliers is to examine the boxplot.

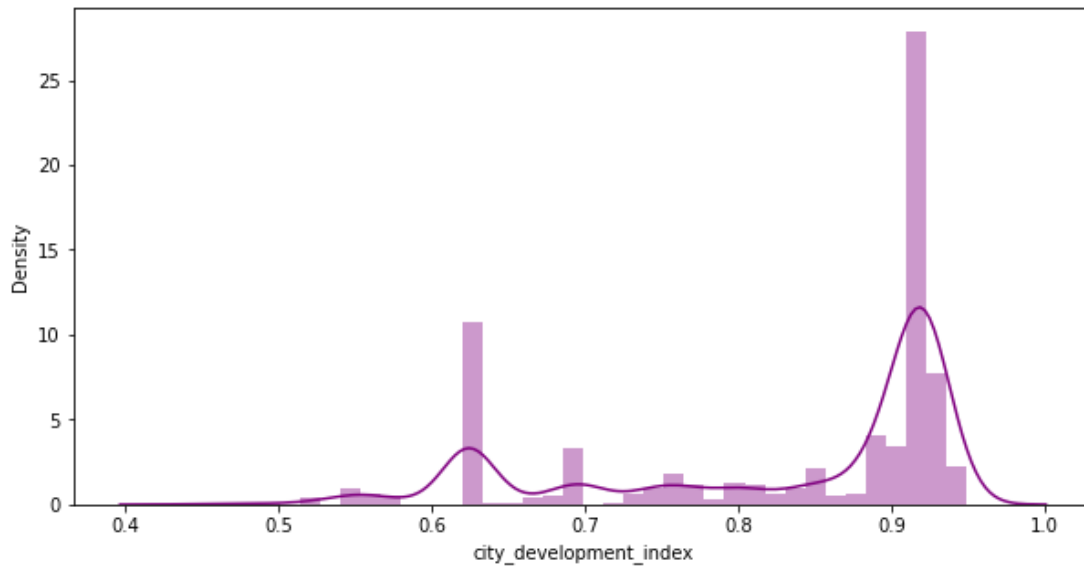


Figure 15: Histogram of City Development Index

Figure 16 shows a boxplot of city development index which shows that there is an outlier with a value less than 0.5 which is the minimum value of the boxplot.

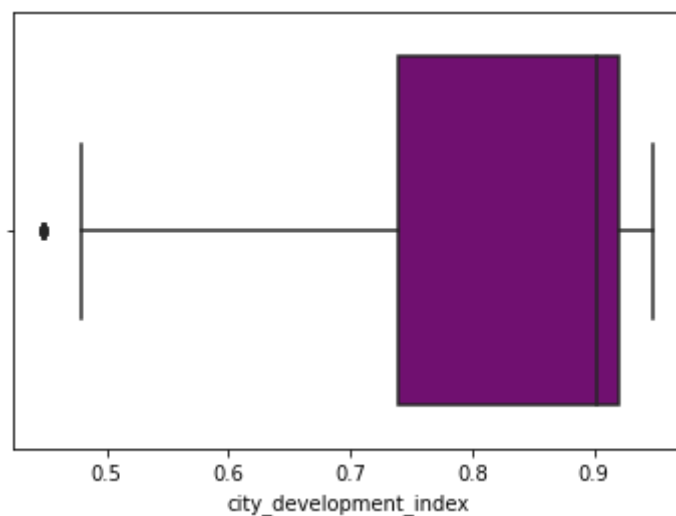


Figure 16: Boxplot of City Development Index.

12. Training Hours

Training hours is an interval variable which gives information about the number of trained hours that an employee has. There are 241 unique values in this variable ranging from 0 to 350. Figure 17 shows the histogram of training hours which is highly skewed towards the right that indicates that there are some outliers in this variable.

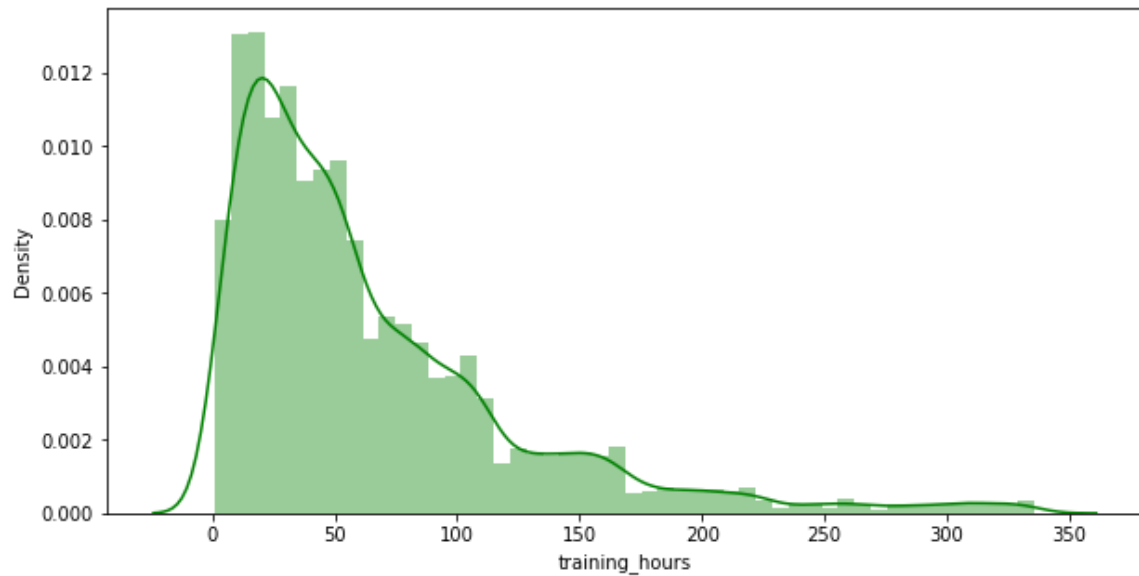


Figure 17: Histogram of Training Hours.

Figure 18 shows a boxplot of training hours that shows a tail of outliers towards the left of the boxplot.

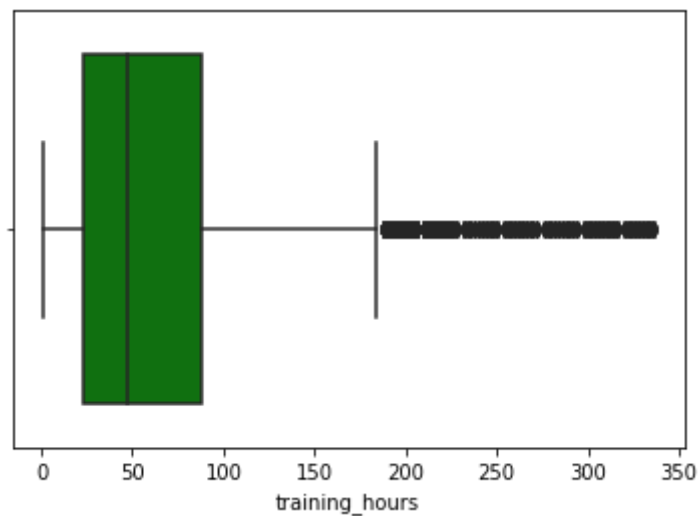


Figure 18: Boxplot of Training Hours.

There is some transformation required in the variable's city development index and training hours to diminish the effect of outliers.

Data Preprocessing

After exploring the data, we prepared it for a Machine learning algorithm.

Imputation:

We started by imputing missing values using the mode, as we had a lot of categorical data. (see fig. 19)

```
[67] hrdf.fillna(hrdf.select_dtypes(include='object').mode().iloc[0], inplace=True)

[68] hrdf.isna().sum()

enrollee_id      0
city             0
city_development_index  0
gender           0
relevent_experience  0
enrolled_university  0
education_level  0
major_discipline  0
experience        0
company_size      0
company_type      0
last_new_job      0
training_hours    0
target           0
dtype: int64
```

Figure 19: Data after imputation.

Categorical Feature Encoding:

We then used pandas replace () function to simplify, reduce/classify and transform the variable “company size” to an integer. This was done as a precursor to one-hot encode these variables. (see fig. 20)

```
[72] hrdf_t['company_size'] = hrdf_t['company_size'].replace(['<10','10/49','<50','50-99'],50) #company size is measured by the average number of employees

[73] hrdf_t['company_size'] = hrdf_t['company_size'].replace(['100-500','500-999'],550)

[74] hrdf_t['company_size'] = hrdf_t['company_size'].replace(['1000-4999'],3000)

[75] hrdf_t['company_size'] = hrdf_t['company_size'].replace(['5000-9999'],7000)

[76] hrdf_t['company_size'] = hrdf_t['company_size'].replace(['10000+'],10000)

[77] hrdf_t.company_size.value_counts()

50      11800
550     3448
10000   2019
3000     1328
7000       563
Name: company_size, dtype: int64
```

Figure 20: Transformation of Company Size variable.

One hot encoding was applied to “company size to allow the representation of categorical data to be more expressive. The integer encoded variable is then removed and a new binary variable is added for each unique integer value. (see fig 21)

```
[78] from sklearn.preprocessing import OneHotEncoder
      encoder=OneHotEncoder(sparse=False)
      company_size_encoded = pd.DataFrame (encoder.fit_transform(hrdf_t[['company_size']]))

[79] company_size_encoded.columns = encoder.get_feature_names(['company_size'])
      hrdf_t.drop(['company_size'] ,axis=1, inplace=True)
      hrencdf= pd.concat([hrdf_t, company_size_encoded ], axis=1)
```

Figure 21: Company Size after One hot encoding.

We also applied the process of one-hot encoding to the categorical variables “company type”, “major_discipline”, and “gender” (see fig 22 & 23)

```
Company type column transformation

[80] hrdf_t['company_type'] = hrdf_t['company_type'].replace(['NGO','Other'], 'NGO & Other')

[81] from sklearn.preprocessing import OneHotEncoder
      encoder=OneHotEncoder(sparse=False)
      company_type_encoded = pd.DataFrame (encoder.fit_transform(hrdf_t[['company_type']]))

[82] company_type_encoded.columns = encoder.get_feature_names(['company_type'])
      hrencdf.drop(['company_type'] ,axis=1, inplace=True)
      hrencdf= pd.concat([hrencdf, company_type_encoded ], axis=1)

[83] major_discipline_encoded = pd.DataFrame (encoder.fit_transform(hrdf_t[['major_discipline']]))

[84] major_discipline_encoded.columns = encoder.get_feature_names(['major_discipline'])
      hrencdf.drop(['major_discipline'] ,axis=1, inplace=True)
      hrencdf= pd.concat([hrencdf, major_discipline_encoded ], axis=1)
```

Figure 22: One hot encoding of Company Type and Major Discipline.

Pandas replace () function was then used to transform “Education level”, “Enrolled University”, “Last new job” and “Experience” to numerical in an organized and neat fashion.

Making it easy for interpretation and averting the curse of dimensionality. (see fig 23,24 and 25)

```
Gender column transformation

[85] gender_encoded = pd.DataFrame (encoder.fit_transform(hrd_f_t[['gender']]))

[86] gender_encoded.columns = encoder.get_feature_names(['gender'])
     hrencdf.drop(['gender'],axis=1, inplace=True)
     hrencdf= pd.concat([hrencdf, gender_encoded ], axis=1)

Education level - column transformation

[87] hrencdf = hrencdf.replace({'Graduate':1})

[88] hrencdf = hrencdf.replace({'Masters':2})

[89] hrencdf = hrencdf.replace({'Phd':3})

[90] hrencdf = hrencdf.replace({'Primary School':0})

[91] hrencdf = hrencdf.replace({'High School':0})
```

Figure 23: One hot encoding Major Discipline and Gender.

enrolled university column transformation

```
[92] hrencdf = hrencdf.replace({'no_enrollment':0})
```

```
[93] hrencdf = hrencdf.replace({'Full time course':1})
```

```
[94] hrencdf = hrencdf.replace({'Part time course':1})
```

last new job column transformation

```
[95] hrencdf['last_new_job'] = hrencdf['last_new_job'].replace(['never'],0)
```

```
[96] hrencdf = hrencdf.replace({'never':0})
```

```
[97] hrencdf['last_new_job'] = hrencdf['last_new_job'].replace(['>4'],5)
```

```
[98] hrencdf = hrencdf.replace({'>4':5})
```

```
[99] hrencdf["last_new_job"] = pd.to_numeric(hrencdf["last_new_job"])
```

```
[100] hrencdf['experience'] = hrencdf['experience'].replace(['>20'],21)
```

```
[101] hrencdf['experience'] = hrencdf['experience'].replace(['<1'],0)
```

```
[102] hrencdf["experience"] = hrencdf["experience"].apply(pd.to_numeric)
```

```
[103] hrencdf = hrencdf.replace({'Has relevent experience':1})
```

Figure 24: One hot encoding of Experience and Last New Job.

From figure 25, it can be examined that our data is now clean and ready for Machine Learning algorithms

```
[104] hrencdf = hrencdf.replace({'No relevent experience':0})

[105] citylist = hrencdf.city.to_list()

[106] hrencdf['city'] = hrencdf['city'].str.replace(r'\D+', '').astype('int')

[107] hrencdf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19158 entries, 0 to 19157
Data columns (total 29 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   enrollee_id                             19158 non-null  int64
1   city                                     19158 non-null  int64
2   city_development_index                  19158 non-null  float64
3   relevent_experience                     19158 non-null  int64
4   enrolled_university                   19158 non-null  int64
5   education_level                        19158 non-null  int64
6   experience                             19158 non-null  int64
7   last_new_job                           19158 non-null  int64
8   training_hours                         19158 non-null  int64
9   target                                 19158 non-null  float64
10  company_size_50                        19158 non-null  float64
11  company_size_550                       19158 non-null  float64
12  company_size_3000                      19158 non-null  float64
13  company_size_7000                      19158 non-null  float64
14  company_size_10000                     19158 non-null  float64
15  company_type_Early Stage Startup        19158 non-null  float64
16  company_type_Funded Startup             19158 non-null  float64
17  company_type_NGO & Other                19158 non-null  float64
18  company_type_Public Sector              19158 non-null  float64
19  company_type_Pvt Ltd                    19158 non-null  float64
20  major_discipline_Arts                   19158 non-null  float64
21  major_discipline_Business Degree        19158 non-null  float64
22  major_discipline_Humanities             19158 non-null  float64
23  major_discipline_No Major               19158 non-null  float64
24  major_discipline_Other                  19158 non-null  float64
25  major_discipline_STEM                   19158 non-null  float64
26  gender_Female                           19158 non-null  float64
27  gender_Male                             19158 non-null  float64
28  gender_Other                            19158 non-null  float64
dtypes: float64(21), int64(8)
memory usage: 4.2 MB
```

Figure 25: Data after all the pre-processing.

Feature Scaling:

We started by balancing the target variable, as an imbalanced target will lead to biased results. (see fig 26)

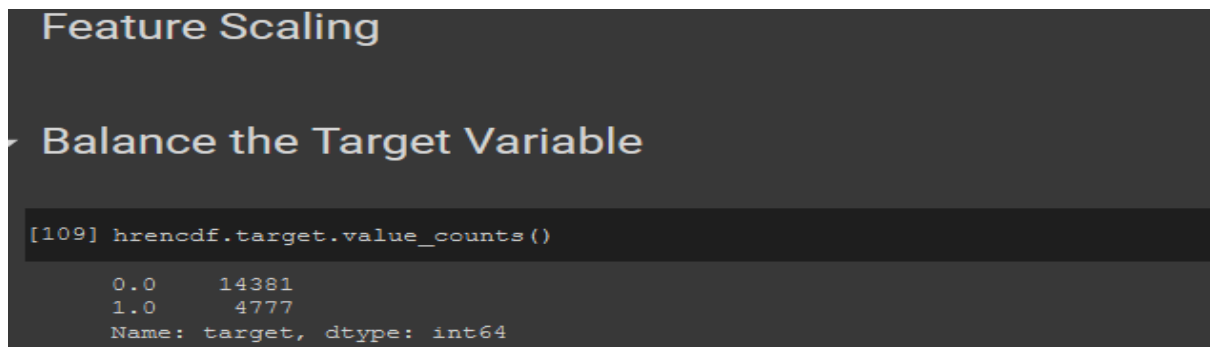


Figure 26: Imbalanced Target

We also dropped 'enrollee_id' as it adds no value to our modelling, and separated the zeros and the ones from the target variable. (See fig 27)

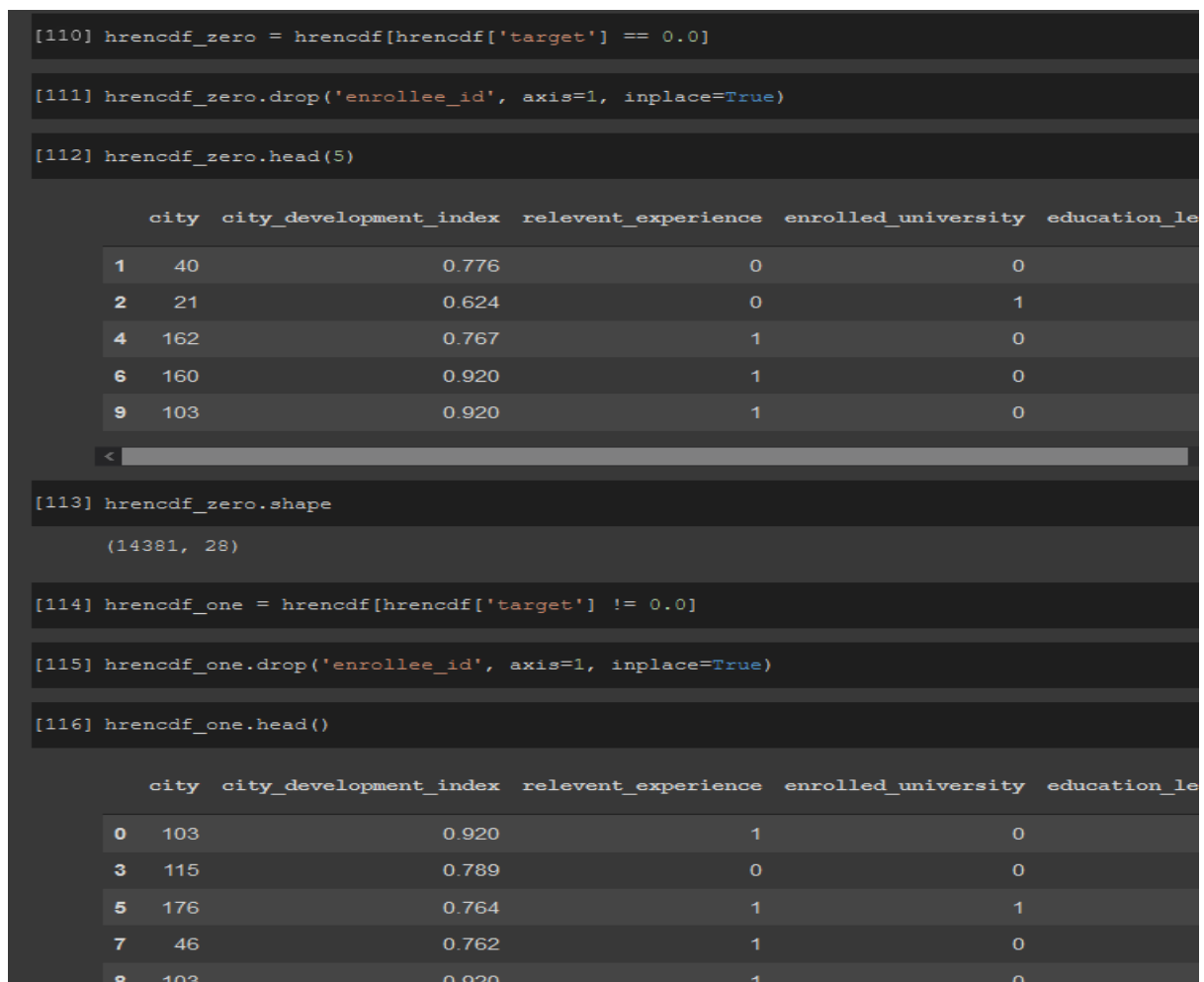


Figure 27: Preparation for sampling

Finally we downsampled the 0s by using the `resample()` function to reduce the rows to 5,000 and concatenated records of 1's..

We now have an even sample of (4777, 28) ones and (5000, 28) zeros, all set for Modelling. (See fig 28)

```
[117] hrencdf_one.shape
      (4777, 28)

[118]
      hrencdf_zero_downsampled = resample(hrencdf_zero ,replace=False,n_samples=5000,random_state=123)
      hrencdf_zero_downsampled.shape
      (5000, 28)

[119] hrencdf_sampled = pd.concat([hrencdf_one,hrencdf_zero_downsampled],ignore_index=True)

[120] hrencdf_sampled.shape
      (9777, 28)

[121] hrencdf_sampled.head()
```

	city	city_development_index	relevent_experience	enrolled_university	education_level	experience	las
0	103	0.920	1	0	1	21	
1	115	0.789	0	0	1	0	
2	176	0.764	1	1	1	11	
3	46	0.762	1	0	1	13	
4	103	0.920	1	0	1	7	

Figure 28: Sampling application

Data Modelling

Logistic Regression

There are several general steps followed when we prepared our classification model:

1. **Import** packages, functions, and classes.
2. **Get** the transformed preprocessed data.
3. **Create** a classification model and train (or fit) it with your existing data.
4. **Evaluate** the model to see if its performance is satisfactory.
5. **Summary** of the model
6. **Interpretation** of Results

Step 1: Import Packages, Functions, and Classes

Logistic_Regression, classification_report (), and confusion_matrix () from scikit-learn were imported.

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix.
hr_logreg = LogisticRegression ()
hr_logreg
```

Step 2: Get Data

Transformation of features was done by scaling each feature to a given range.

This estimator scales and translates each feature individually such that it is in the given range on the training set, e.g., between zero and one.

```
# create a scalar object
scaler = MinMaxScaler()
# fit and transform the data
Xtrain_norm = pd.DataFrame(scaler.fit_transform(X_train), columns=X_train.columns)
Xtest_norm = pd.DataFrame(scaler.fit_transform(X_test), columns=X_test.columns)
Xtrain_norm.head()
Xtest_norm.head()
```

	enrollee_id	city	city_development_index	relevent_experience	enrolled_university	education_level	experience	last_new_job	training_hours	company_size_50	company_size_550	company_size_3000
0	0.303940	0.547486	0.932136	1.0	0.0	0.333333	0.190476	0.4	0.134328	1.0	0.0	0.0
1	0.281109	0.335196	0.928144	1.0	0.0	0.666667	1.000000	1.0	0.423881	0.0	1.0	0.0
2	0.246712	0.553073	0.876248	0.0	1.0	0.333333	0.238095	0.0	0.038806	1.0	0.0	0.0
3	0.419805	0.569832	0.942116	0.0	1.0	0.666667	0.238095	0.6	0.056716	0.0	1.0	0.0
4	0.068824	0.111732	0.351297	1.0	0.0	0.333333	1.000000	0.2	0.540299	1.0	0.0	0.0
	enrollee_id	city	city_development_index	relevent_experience	enrolled_university	education_level	experience	last_new_job	training_hours	company_size_50	company_size_550	company_size_3000
0	0.313249	0.804469	0.213573	1.0	0.0	0.333333	0.095238	0.2	0.254491	1.0	0.0	0.0
1	0.884382	0.368715	0.812375	0.0	1.0	0.000000	0.142857	0.0	0.035928	1.0	0.0	0.0
2	0.390917	0.927374	0.944112	1.0	1.0	0.333333	0.238095	0.2	0.916168	1.0	0.0	0.0
3	0.403177	0.754190	0.896208	0.0	1.0	0.333333	0.285714	0.2	0.044910	1.0	0.0	0.0
4	0.890767	0.111732	0.351297	1.0	1.0	0.333333	0.285714	0.2	0.092814	0.0	1.0	0.0

Figure 29: X train and X Test first 5 rows.

Step 3: Create a Model and Train It

After preparing the input and output prepared, we created and defined the classification model by creating an instance of the class `LogisticRegression`:

```
hr_logreg = LogisticRegression()  
hr_logreg
```

The above statement created an instance of `LogisticRegression` and bound its references to the variable `hr_logreg`.

After the model was created, we need to fit (or train) the model. Model fitting is the process of determining the coefficients b_0, b_1, \dots, b_r that correspond to the best value of the cost function.

```
hr_logreg.fit(Xtrain_norm, y_train)
```

Step 4: Evaluate the Model

To check the performance of the model we used the following code:

```
pred = hr_logreg.predict(Xtest_norm)  
pred
```

This function returns the predicted output values as a one-dimensional array.

Output:

```
pred = hr_logreg.predict(Xtest_norm)  
pred  
  
array([1., 1., 0., ..., 0., 0., 1.])
```

Figure 30: Logistic Regression Prediction.

The accuracy of the model on the test data set after training is 68.8%.

```
pred_score = hr_logreg.score(Xtest_norm, y_test)  
pred_score
```

Output:

```
pred_score = hr_logreg.score(Xtest_norm, y_test)  
pred_score  
  
0.6881390593047034
```

Figure 29: Accuracy of the model.

To evaluate the accuracy of the model we used the score function. This function takes the input and output as arguments and returns the ratio of the number of correct predictions to the number of observations. To get more information on the accuracy of the model we used a confusion **matrix**. This function takes actual and predicted outputs as the arguments.

```
print ("Confusion Matrix:\n", confusion_matrix(y_test,pred))
```

Output:

```

Classification report:
              precision    recall  f1-score   support

     0.0         0.67       0.75       0.71         997
     1.0         0.71       0.62       0.66         959

 accuracy          0.69
 macro avg         0.69       0.69       0.69         1956
 weighted avg      0.69       0.69       0.69         1956

Confusion Matrix:
[[750 247]
 [363 596]]

```

Figure 31: Classification Report and Confusion Matrix.

The dimension of this matrix is 2*2 because this model is binary classification. There are two classes 0 and 1. Diagonal values represent accurate predictions, while non-diagonal elements are inaccurate predictions. In the output, 1346 are the correct predictions and 610 are incorrect predictions.

Step 5 Summary of the model

The summary of results displayed after training the model is below:

Logit Regression Results						
Dep. Variable:	target	No. Observations: 9777				
Model:	Logit	Df Residuals: 9752				
Method:	MLE	Df Model: 24				
Date:	Tue, 27 Apr 2021	Pseudo R-squ.: 0.1369				
Time:	04:06:41	Log-Likelihood: -5846.7				
converged:	False	LL-Null: -6774.4				
Covariance Type:	nonrobust	LLR p-value: 0.000				
	coef	std err	z	P> z	[0.025	0.975]
enrollee_id	8.456e-06	2.35e-06	3.594	0.000	3.84e-06	1.31e-05
city	-0.0002	0.001	-0.345	0.730	-0.001	0.001
city_development_index	-5.9031	0.212	-27.894	0.000	-6.318	-5.488
relevent_experience	-0.3463	0.055	-6.276	0.000	-0.454	-0.238
enrolled_university	0.2531	0.054	4.655	0.000	0.147	0.360
education_level	0.1711	0.039	4.427	0.000	0.095	0.247
experience	-0.0131	0.004	-3.058	0.002	-0.021	-0.005
last_new_job	0.0675	0.016	4.210	0.000	0.036	0.099
training_hours	-0.0008	0.000	-2.146	0.032	-0.002	-7.02e-05
company_size_50	1.3662	3.65e+06	3.75e-07	1.000	-7.15e+06	7.15e+06
company_size_550	0.6551	3.57e+06	1.83e-07	1.000	-7e+06	7e+06
company_size_3000	0.5434	3.57e+06	1.52e-07	1.000	-7e+06	7e+06
company_size_7000	0.7962	3.63e+06	2.19e-07	1.000	-7.12e+06	7.12e+06
company_size_10000	0.7466	3.64e+06	2.05e-07	1.000	-7.13e+06	7.13e+06
company_type_Early Stage Startup	0.7567	2.14e+06	3.53e-07	1.000	-4.2e+06	4.2e+06
company_type_Funded Startup	0.6945	2.07e+06	3.36e-07	1.000	-4.05e+06	4.05e+06
company_type_NGO & Other	1.1664	2.08e+06	5.6e-07	1.000	-4.08e+06	4.08e+06
company_type_Public Sector	1.4480	2.11e+06	6.87e-07	1.000	-4.13e+06	4.13e+06
company_type_Pvt Ltd	1.3745	2.14e+06	6.42e-07	1.000	-4.2e+06	4.2e+06
major_discipline_Arts	0.9768	5.3e+06	1.84e-07	1.000	-1.04e+07	1.04e+07
major_discipline_Business Degree	0.9318	5.3e+06	1.76e-07	1.000	-1.04e+07	1.04e+07
major_discipline_Humanities	0.8542	5.3e+06	1.61e-07	1.000	-1.04e+07	1.04e+07
major_discipline_No Major	0.7297	5.3e+06	1.38e-07	1.000	-1.04e+07	1.04e+07
major_discipline_Other	0.5716	5.3e+06	1.08e-07	1.000	-1.04e+07	1.04e+07
major_discipline_STEM	0.7098	5.3e+06	1.34e-07	1.000	-1.04e+07	1.04e+07
gender_Female	1.6445	4.62e+05	3.56e-06	1.000	-9.06e+05	9.06e+05
gender_Male	1.4236	4.62e+05	3.08e-06	1.000	-9.06e+05	9.06e+05
gender_Other	1.7058	4.62e+05	3.69e-06	1.000	-9.06e+05	9.06e+05

Figure 32: Summary of the Model.

Step 6 Interpretation of Results

1. For every unit increase in the scale of the city development there is a decrease in the log odds of an employee leaving the job of -5.9031.
2. For every unit increase in the relevant experience of an employee there is a decrease in the log odds of an employee leaving the job of -0.3463.
3. For every unit increase in the type of university enrolled there is an increase in the log odds of an employee leaving the job of 0.2531.
4. For every unit increase in the education level there is an increase in the log odds of an employee leaving the job of 0.1711.

5. For every unit increase in experience there is a decrease in the log odds of an employee leaving the job of -0.0131.
6. For every unit increase in the number of years between job changes there is an increase in the log odds of an employee leaving the job of 0.0675.
7. For every unit increase in the number of training hours there is a decrease in the log odds of an employee leaving the job of -0.0008.

Neural Networks


The Neural Networks is performed on the train dataset. The following steps are involved are:

1. Import the Library files
2. Defining the model
3. Predicting the values

1. Importing the library files:

Tensor flow provides better computation assistance on large numerical data. It also comprises machine learning and deep learning algorithms used to perform the analysis.

▾ Neural Networks

```
 import tensorflow as tf  
from keras.models import Sequential  
from keras.layers import Dense
```

2. Defining the model

In order to have a better performing model, there are about six layers of weighted sum added. Relu , activation function which provides the linear function between the output and input. It is the most commonly used activation function because it provides better performance. At the last layer we added sigmoid activation function because it helps to provide the probability of

the output ranging (0,1). The Adam optimizer is used in order to provide the best optimizations. It basically uses the gradient-based optimization of stochastic objective functions to implement the model.

```
model = Sequential()

model.add(Dense(8, activation='relu', input_shape=(28,)))

model.add(Dense(8, activation='relu'))

model.add(Dense(8, activation='relu'))

model.add(Dense(8, activation='relu'))

model.add(Dense(8, activation='relu'))

model.add(Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

model.fit(Xtrain_norm, y_train, epochs=50, batch_size=32, verbose=1) #validation_data=(X_val, Y_val))
```

The batch size 32 is picked for 50 epochs. Having 50 iterations provided the better model accuracy. The model is run in 50 epochs. Accuracy kept increasing with every epoch execution and the last epoch ended with the accuracy is 72.5%. But the average accuracy still stays 69% and the loss is 0.59. The loss parameter is similar to the MSE. The accuracy is improved compared to the Logistic Regression. The model is then tested on the test data set and the target values are predicted.

Model Comparison:

Based on model comparison, neural networks (0.68) performed slightly better than logistic regression(0.696). Addition of a few more layers in the neural network algorithm could enhance the accuracy of neural networks further. In addition a higher sample size could also add up the accuracy rate of neural networks.

Model	Accuracy
Logistics Regression	0.68
Neural Networks	0.696



Business Recommendations:

These are some of the real-time business recommendations obtained after performing the analysis in the dataset. These recommendations can help the employer to determine which category of people can be hired or willing to be hired.

1. People who are working in the private, public sector are willing to change jobs. While people at startups are not willing to change the job.
2. All the major-discipline studied graduates are willing to change the job. Business degree, Arts, Stem discipline are more aggressive in the job changing process.
3. Relevant experience has a negative effect on the dataset, which elucidates that relevant experience is not playing a vital role in employees switching their job.
4. Both the gender are equally aggressive and will be looking to switch the jobs.
5. The positive effect on the company size variable elucidates that whatever the size of the company, candidates are willing to change the job.
6. Candidates are not tied to the city. Henceforth, Importance is not given to which city the job is posted.
7. The negative effect on the independent variable, experience proves that a candidate's job change won't be based on the number of years of experience they gained.

Conclusion

We explored the distributions and statistical properties of every attribute in the dataset and the correlation among variables. We implemented two predictive models: Logistic Regression and Neural Networks to classify the employees into categories of who is likely to move on to another job. The accuracies of both the models are similar and based on the coefficients and p-value of the variables, Education level and experience stand out as the most significant factors considered while choosing the next job.

References:

1. <https://www.kaggle.com/arashnic/hr-analytics-job-change-of-data-scientists>.