



'철새도 능력'
잡호핑 할 사람은
누구인가?

Team 1

이수진, 오무열, 정재하, 정희경, 신금하



INDEX

01 프로젝트 개요

- 주제선정
- 데이터 출처
- 데이터 구성

02 분석과정

- Data Visualization
- Data Processing

03 분석 및 모델링

- Keras를 활용한 Deep Learning
- Pycaret을 활용한 Machine Learning

04 분석결과 및 리뷰

- 분석결과
- 어려웠던 점
- 후속과제



01 프로젝트 개요

- 주제선정
- 데이터 출처
- 데이터 구성

■ 주제선정

빅데이터 전문회사의 교육 수료 후 데이터 사이언티스트로 전향할 사람 예측하기



■ 데이터 출처

HR Analytics: Job Change of Data Scientists

Predict who will move to a new job

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

kaggle

The screenshot shows the Kaggle interface. On the left is a sidebar with navigation links: Home, Compete, Data (which is selected), Code, Communities, Courses, and More. Below this is a 'Recently Viewed' section with links to other datasets. The main content area displays a dataset titled 'HR Analytics: Job Change of Data Scientists'. The title is in bold red text at the top. Below it is a subtitle 'Predict who will move to a new job'. A small profile picture of 'Möbius' and the text 'updated 5 months ago (Version 1)' are visible. The dataset has a 'Data' tab selected, followed by Tasks (2), Code (229), Discussion (16), Activity, and Metadata. To the right of the tabs are buttons for 'Download (2 MB)' and 'New Notebook'. Below the tabs, there are sections for 'Usability' (10.0), 'License' (CC0: Public Domain), and 'Tags' (business, education, tabular data, intermediate, binary classification). The 'Description' and 'Context and Content' sections provide detailed information about the dataset's purpose and the data it contains.

HR Analytics: Job Change of Data Scientists
Predict who will move to a new job

Möbius • updated 5 months ago (Version 1)

Data Tasks (2) Code (229) Discussion (16) Activity Metadata

Download (2 MB) New Notebook

Usability 10.0 License CC0: Public Domain Tags business, education, tabular data, intermediate, binary classification

Description

Context and Content

A company which is active in Big Data and Data Science wants to hire data scientists among people who successfully pass some courses which conduct by the company. Many people signup for their training. Company wants to know which of these candidates are really wants to work for the company after training or looking for a new employment because it helps to reduce the cost and time as well as the quality of training or planning the courses and categorization of candidates. Information related to demographics, education, experience are in hands from candidates signup and enrolment.

This dataset designed to understand the factors that lead a person to leave current job for HR researches too. By model(s) that uses the current credentials,demographics,experience data you will predict the probability of a candidate to look for a new job or will work for the company, as well as interpreting affected factors on employee decision.

The whole data divided to train and test . Target isn't included in test but the test target values data file is in hands for related tasks. A sample submission correspond to enrollee_id of test set provided too with columns : enrollee_id , target

■ 데이터 구성

Features

- enrollee_id : Unique ID for candidate
- city: City code
- city_development_index : Developement index of the city (scaled)
- gender: Gender of candidate
- relevent_experience: Relevant experience of candidate
- enrolled_university: Type of University course enrolled if any
- education_level: Education level of candidate
- major_discipline :Education major discipline of candidate
- experience: Candidate total experience in years
- company_size: No of employees in current employer's company
- company_type : Type of current employer
- lastnewjob: Difference in years between previous job and current job
- training_hours: training hours completed
- target: 0 – Not looking for job change, 1 – Looking for a job change

19158 rows × 14 columns

	enrollee_id	city	city_development_index	gender	relevent_experience	enrolled_university	education_level	major_discipline	experience	company_size	company_type	last_new_job	training_hours	target
0	8949	city_103	0.920	Male	Has relevent experience	no_enrollment	Graduate	STEM	>20	NaN	NaN	1	36	1.0
1	29725	city_40	0.776	Male	No relevent experience	no_enrollment	Graduate	STEM	15	50-99	Pvt Ltd	>4	47	0.0
2	11561	city_21	0.624	NaN	No relevent experience	Full time course	Graduate	STEM	5	NaN	NaN	never	83	0.0
3	33241	city_115	0.789	NaN	No relevent experience	NaN	Graduate	Business Degree	<1	NaN	Pvt Ltd	never	52	1.0
4	666	city_162	0.767	Male	Has relevent experience	no_enrollment	Masters	STEM	>20	50-99	Funded Startup	4	8	0.0

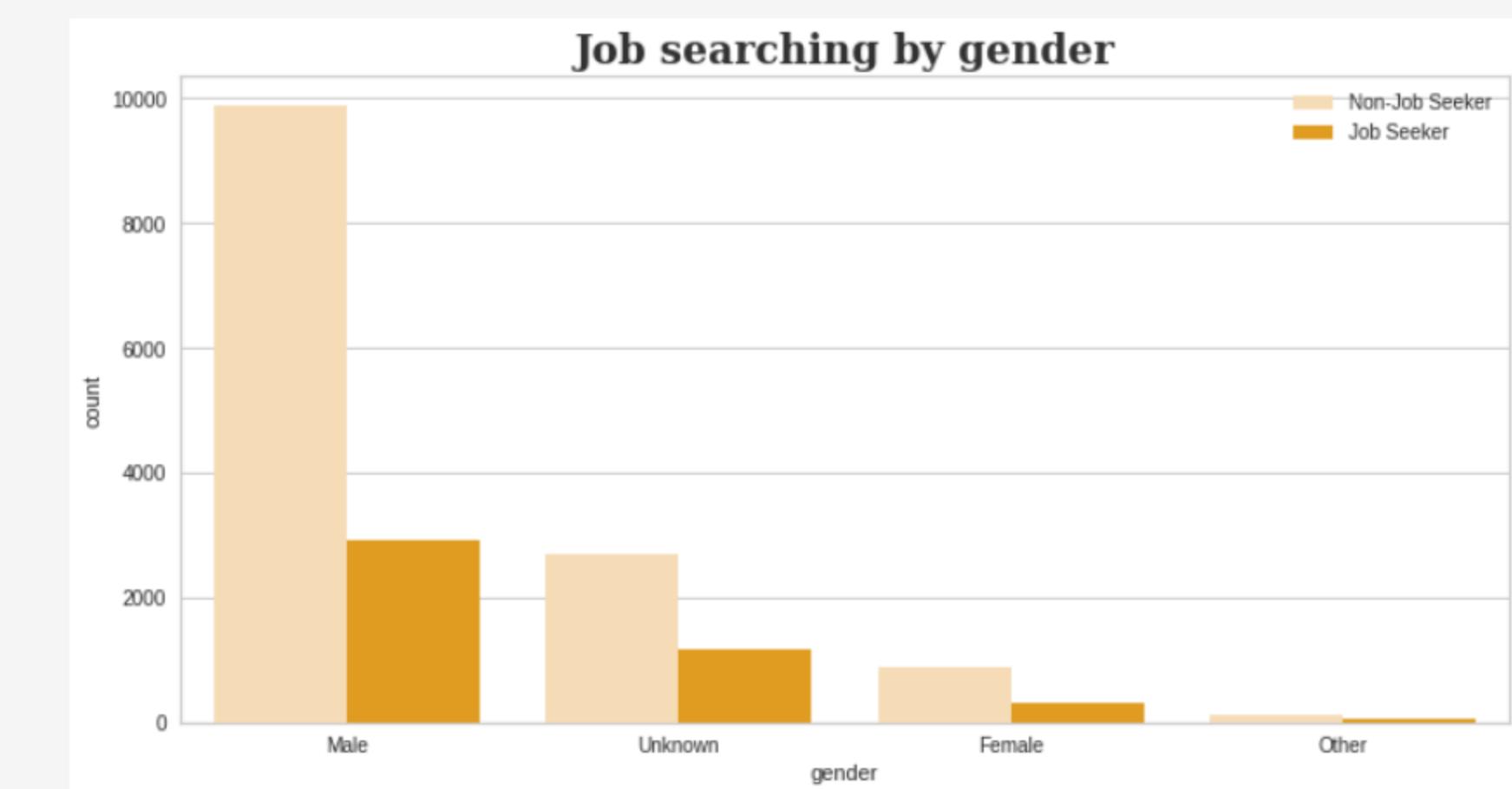
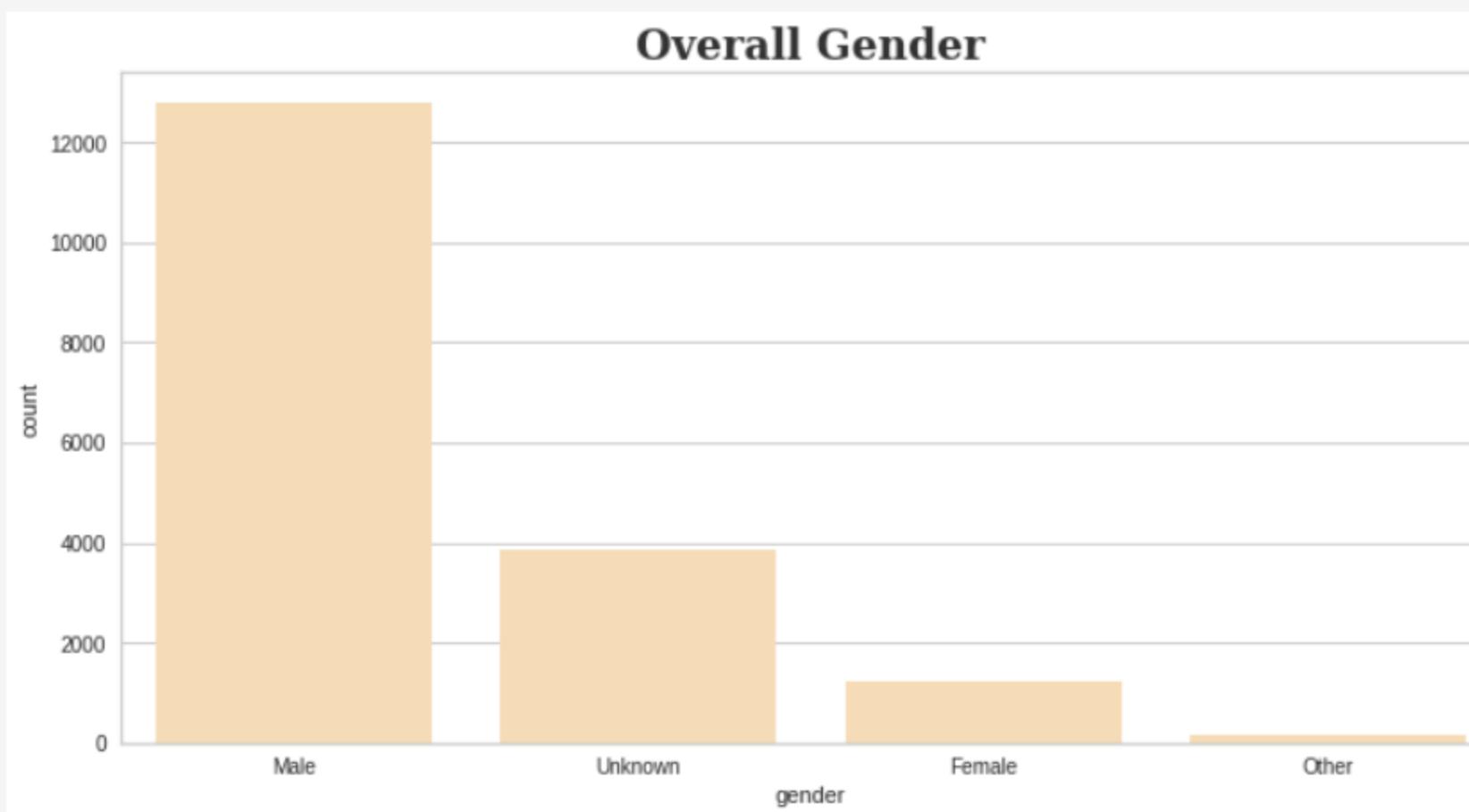


02 분석과정

- Data Visualization
- Data Proprocessing

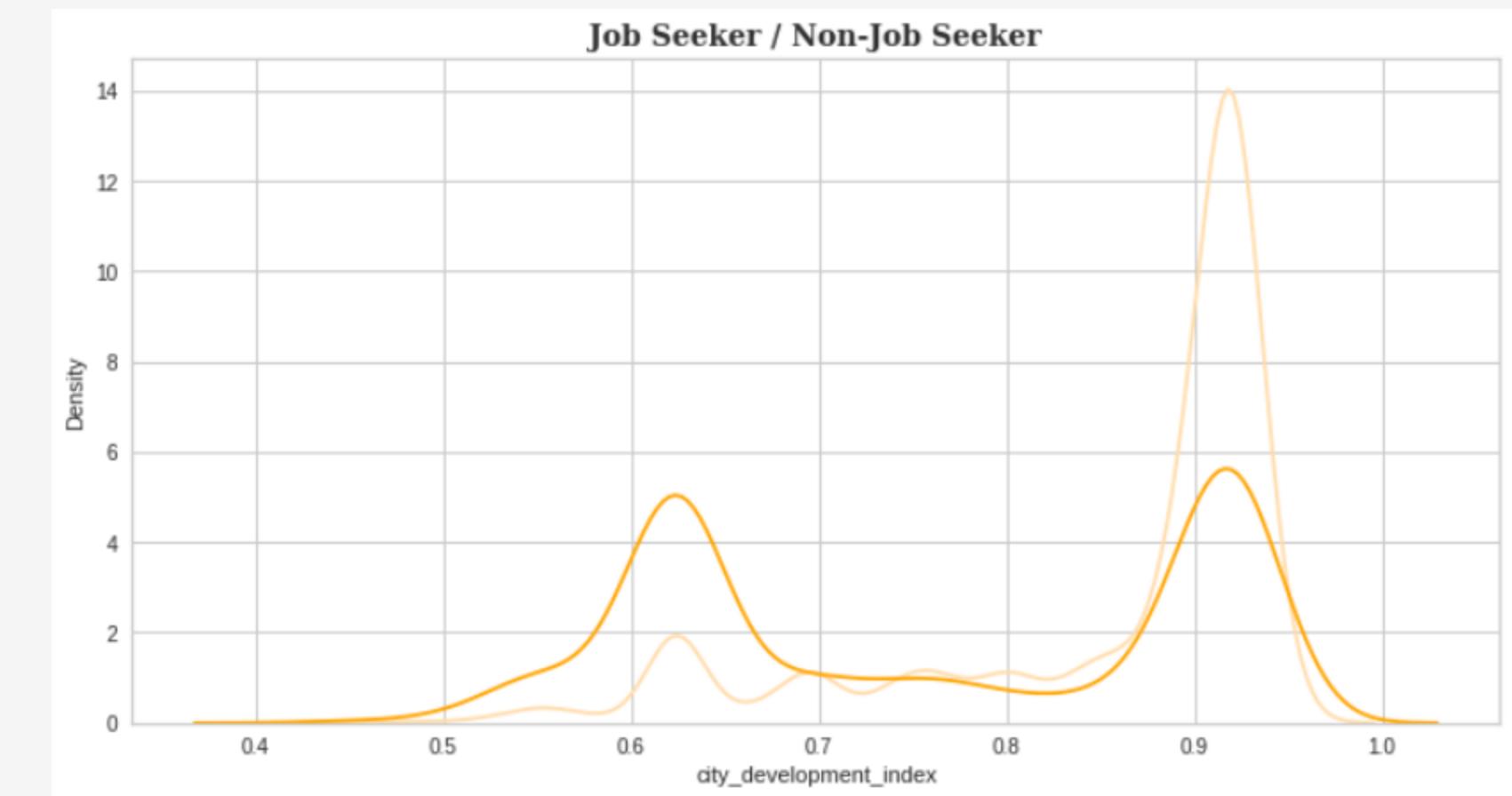
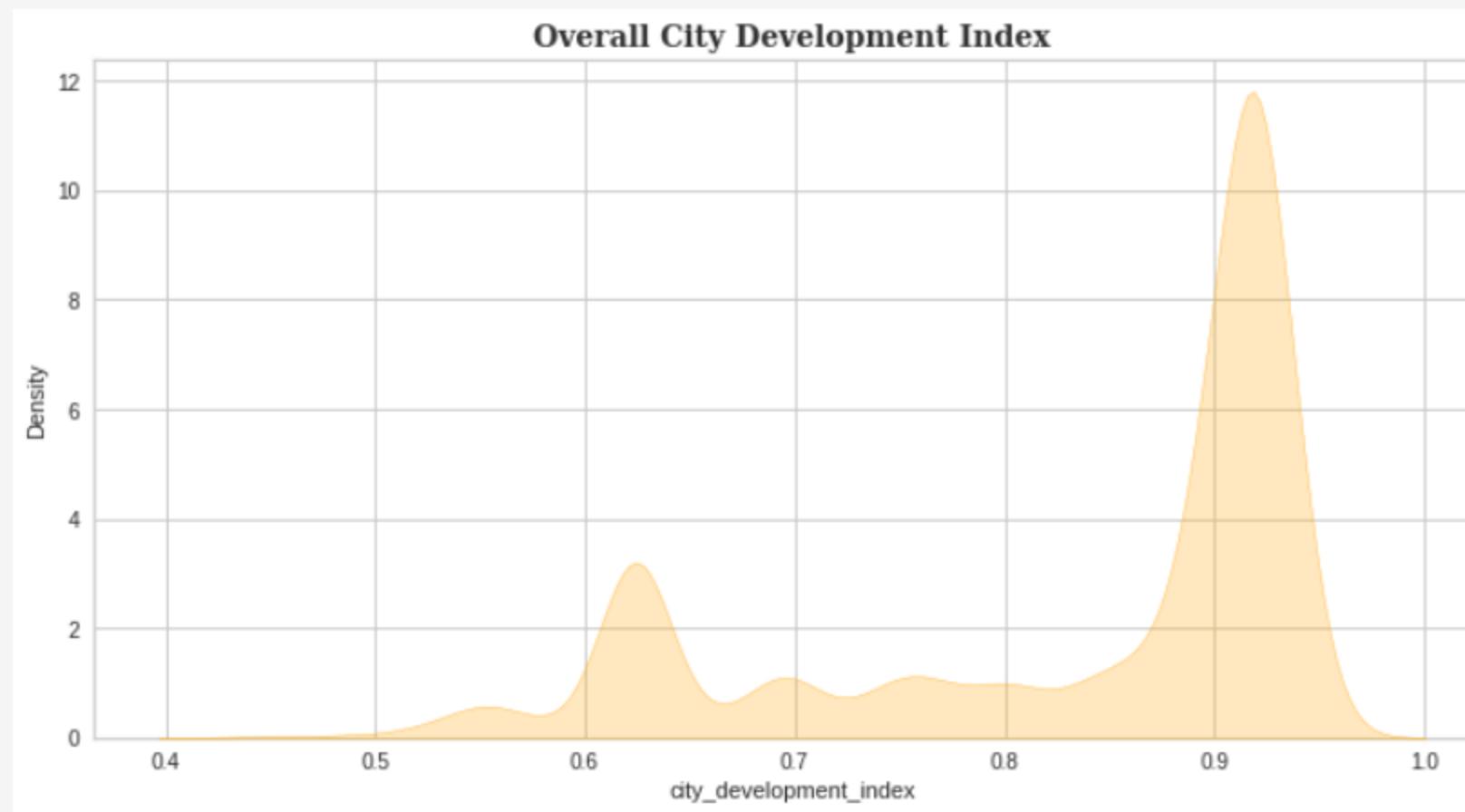
■ Data Visualization

데이터 시각화



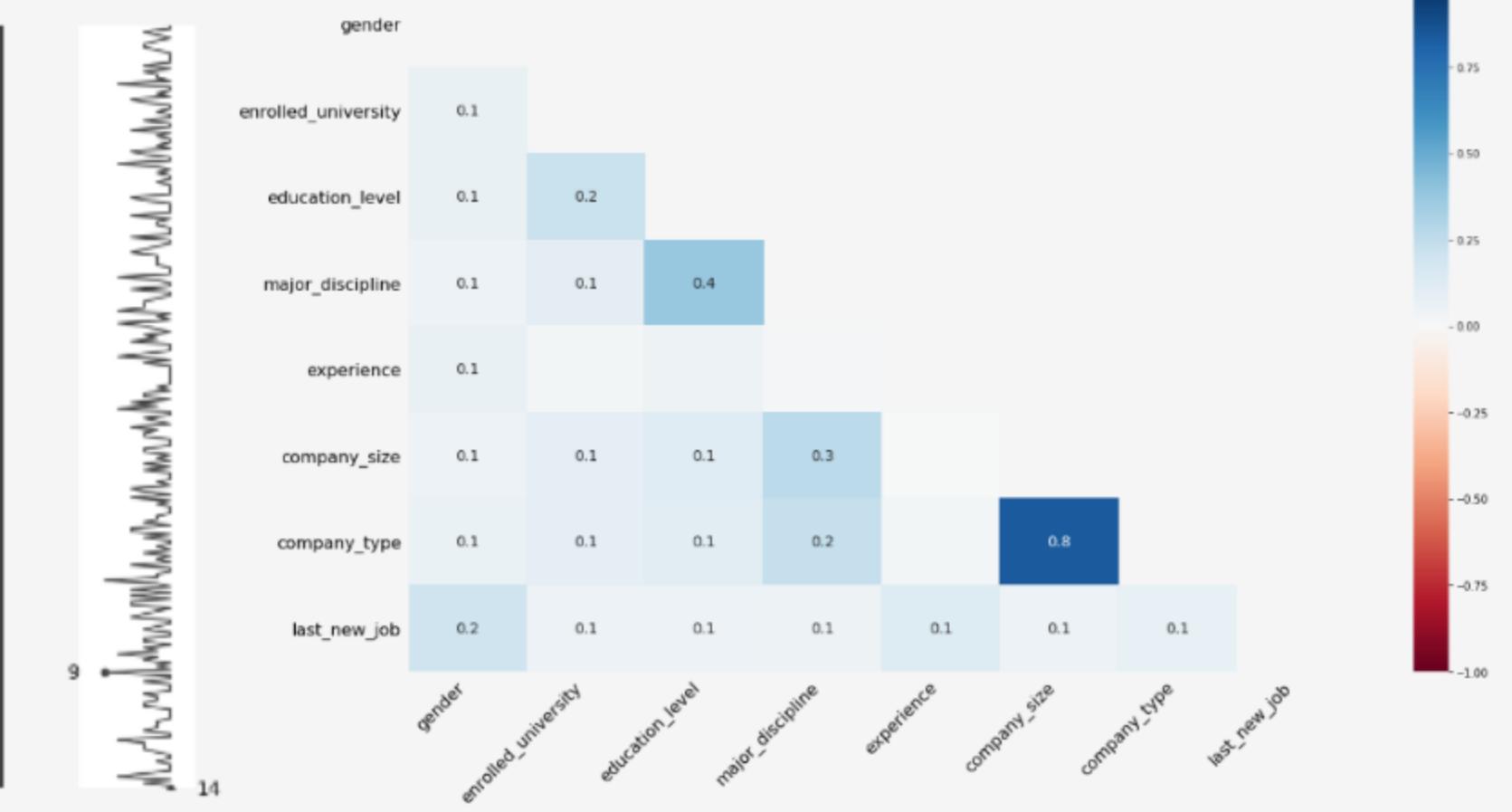
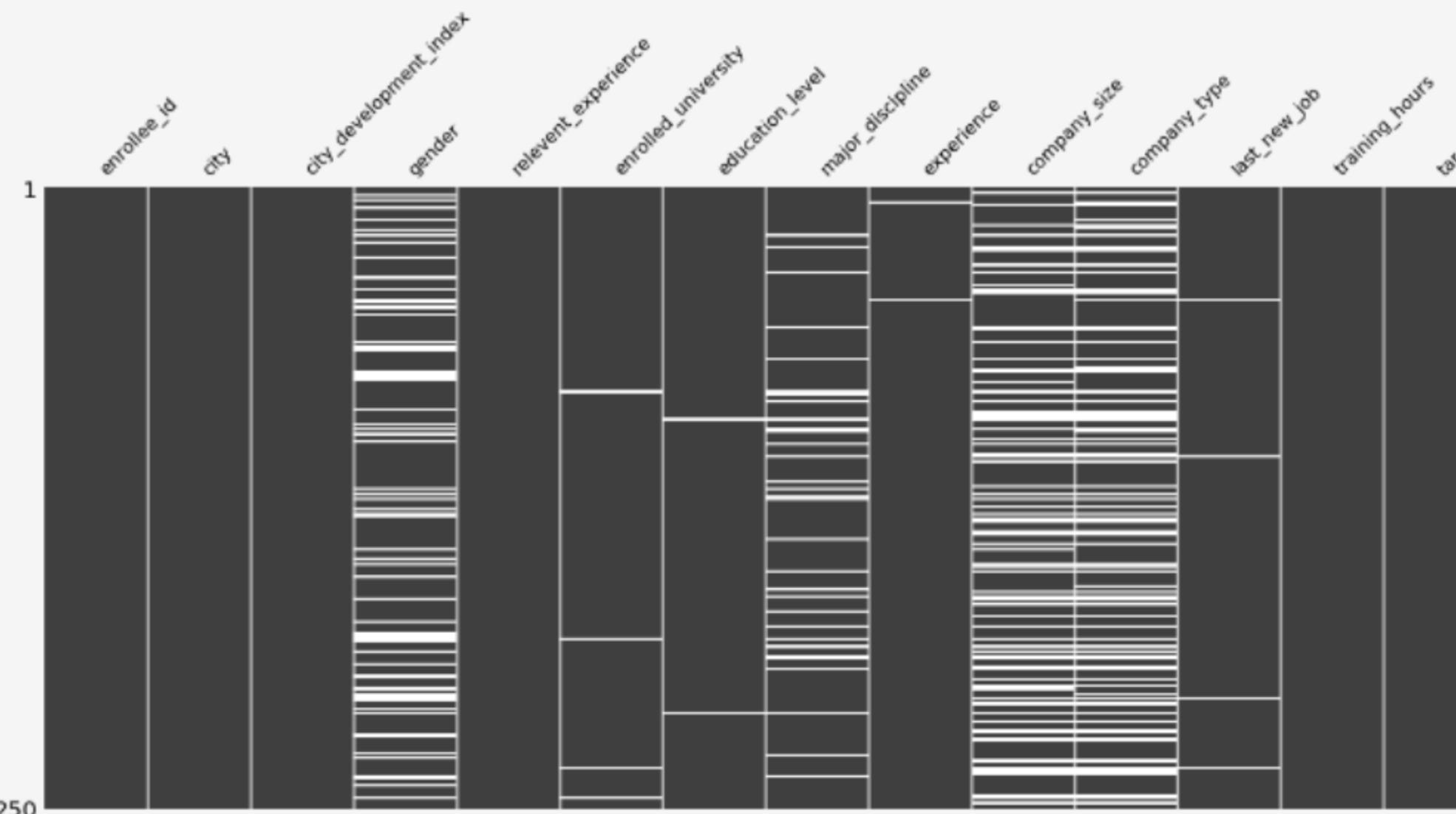
■ Data Visualization

데이터 시각화



Data Visualization

결측치 시각화



Imbalanced Data

How many are looking for a new role?

25%

Job-Seeker

75%

Non Job-Seeker

Data Preprocessing

데이터 전처리 방식

1. 결측치 처리

2. 데이터 불균형 처리

- 'enrollee_id' Column 삭제
- 'experience', 'enrolled_university', 'last_new_job', 'education_level' null 값 삭제
- str -> int 수정
- Label Encoding 후 KNN Imputer로 결측치 채우기

```
[9] # Delete  
  
df.drop(['enrollee_id'], axis=1, inplace=True)  
df.dropna(subset=["experience"], inplace=True)  
df.dropna(subset=["enrolled_university"], inplace=True)  
df.dropna(subset=["last_new_job"], inplace=True)  
df.dropna(subset=["education_level"], inplace=True)  
  
[10] # Replace  
  
df['company_type'].fillna('Unknown', inplace=True)  
df['major_discipline'].fillna('Unknown', inplace=True)  
df['gender'].fillna('Unknown', inplace=True)  
  
[11] # Change to int  
  
df['experience'] = df['experience'].apply(lambda x: '0' if x == '<1' else x)  
df['experience'] = df['experience'].apply(lambda x: '20' if x == '>20' else x)  
df['experience'] = df['experience'].astype(int)  
  
[12] # Change to int  
  
df['last_new_job'] = df['last_new_job'].apply(lambda x: '0' if x == 'never' else x)  
df['last_new_job'] = df['last_new_job'].apply(lambda x: '5' if x == '>4' else x)  
df['last_new_job'] = df['last_new_job'].astype(int)
```

```
[13] # Import LabelEncoder  
from sklearn.preprocessing import LabelEncoder  
  
[14] # LabelEncoder를 적용한 새로운 데이터프레임을 df_lb으로 생성  
df_lb = df.copy()  
  
[15] # LabelEncoder를 적용한 데이터는 object타입의 데이터이므로 해당 데이터들을 가지 않음  
transform_list = list(df_lb.select_dtypes(include=[object]).columns)  
transform_list  
  
['city',  
 'gender',  
 'relevant_experience',  
 'enrolled_university',  
 'education_level',  
 'major_discipline',  
 'company_size',  
 'company_type']  
  
[16] # LabelEncoder를 fit한 칼럼별 객체의 LabelEncoder 생성  
  
for i, value in enumerate(transform_list):  
    globals()['LabelEncoder_{}'.format(i)] = LabelEncoder()  
    print('LabelEncoder_{} : {}'.format(i+1, value))  
  
LabelEncoder_0 : city  
LabelEncoder_1 : gender  
LabelEncoder_2 : relevant_experience  
LabelEncoder_3 : enrolled_university
```

```
[17] # Import KNNImputer  
from sklearn.impute import KNNImputer  
  
[18] # imputer 정의  
imputer = KNNImputer(n_neighbors=5)  
  
[19] # KNNImputer의 fit_transform을 df_lb에 적용  
# df_im로 데이터프레임명 지정  
df_im = imputer.fit_transform(df_lb)  
  
[20] # df_im 확인  
df_im  
  
array([[ 5. ,  0.92 ,  1. , ...,  1. ,  36. ,  1. ,  
       [ 77. ,  0.776,  1. , ...,  5. ,  47. ,  0. ,  1. ,  
       [ 64. ,  0.624,  3. , ...,  0. ,  83. ,  0. ,  1. ,  
       ...  
       [ 5. ,  0.92 ,  1. , ...,  4. ,  44. ,  0. ,  
       [ 84. ,  0.802,  1. , ...,  2. ,  97. ,  0. ,  1. ,  
       [ 95. ,  0.855,  3. , ...,  1. , 127. ,  0. ,  1.]])
```

■ Data Preprocessing

데이터 전처리 방식

1. 결측치 처리

2. 데이터 불균형 처리

- SMOTE 활용하여 클래스 비율 동일하게 수정

```
# install imbalanced-learn  
# !pip install imbalanced-learn==0.7.0  
  
[ ] # import SMOTE  
from imblearn.over_sampling import SMOTE  
  
[ ] # X_train_copy와 y_train을 y_train(target)의 클래스 비율을 동일하게 하는 방식으로 SMOTE 실행  
smote = SMOTE(random_state=0)  
X_train_over, y_train_over = smote.fit_resample(X_train_copy, y_train)  
  
[ ] # y_train_over를 model의 label로 넣기 위해 array.reshape(-1, 1) 진행  
y_train_over = np.array(y_train_over)  
y_train_over = y_train_over.reshape(-1, 1)  
  
[ ] # y_test를 model의 label로 넣기 위해 array.reshape(-1, 1) 진행  
y_test = np.array(y_test)  
y_test = y_test.reshape(-1, 1)
```

■ Data Preprocessing

범주형 vs 수치형 데이터



- 범주형 : 몇 개의 범주로 나누어진 자료를 의미
 - 명목형 : 성별, 성공여부, 혈액형 등 단순히 분류된 자료
 - 순서형 : 개개의 값들이 이산적이며 그들 사이에 순서 관계가 존재하는 자료
- 수치형 : 이산형과 연속형으로 이루어진 자료를 의미
 - 이산형 : 이산적인 값을 갖는 데이터로 출산횟수 등을 의미
 - 연속형 : 연속적인 값을 갖는 데이터로 신장, 체중 등을 의미

```
[40] # import train_test_split
from sklearn.model_selection import train_test_split

[41] # train_test_split 실행
X = df_im.drop('target', axis=1)
y = df_im['target']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0, stratify=y)

[42] df_im.columns
Index(['city', 'city_development_index', 'gender', 'relevent_experience',
       'enrolled_university', 'education_level', 'major_discipline',
       'experience', 'company_size', 'company_type', 'last_new_job',
       'training_hours', 'target'],
      dtype='object')

[43] numeric_features = ['city_development_index', 'experience', 'last_new_job', 'training_hours']
numeric_transformer = MinMaxScaler()

categorical_features = ['city', 'gender', 'relevent_experience', 'enrolled_university',
                       'education_level', 'major_discipline', 'company_size', 'company_type']
categorical_transformer = OneHotEncoder(categories='auto') # categories='auto' : just for ignoring warning messages

preprocessor = ColumnTransformer(
    transformers=[ # List of (name, transformer, column(s))
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)])
preprocessor_pipe = Pipeline(steps=[('ColumnTransform', preprocessor)])
```

■ Data Preprocessing

데이터 전처리 전후 비교

Before

	enrollee_id	city	city_development_index	gender	relevent_experience	enrolled_university	education_level	major_discipline	experience	company_size	company_type
0	8949	city_103	0.920	Male	Has relevent experience	no_enrollment	Graduate	STEM	>20	NaN	NaN
1	29725	city_40	0.776	Male	No relevent experience	no_enrollment	Graduate	STEM	15	50-99	Pvt Ltd
2	11561	city_21	0.624	NaN	No relevant experience	Full time course	Graduate	STEM	5	NaN	NaN
3	33241	city_115	0.789	NaN	No relevent experience	NaN	Graduate	Business Degree	<1	NaN	Pvt Ltd
4	666	city_162	0.767	Male	Has relevent experience	no_enrollment	Masters	STEM	>20	50-99	Funded Startup

After

	city	city_development_index	gender	relevent_experience	enrolled_university	education_level	major_discipline	experience	company_size	company_type	last_new_job
0	5	0.920	1	0	2	0	5	20	4.6	6	1
1	77	0.776	1	1	2	0	5	15	5.0	5	5
2	64	0.624	3	1	0	0	5	5	4.6	6	0
3	50	0.767	1	0	2	2	5	20	5.0	1	4
4	57	0.764	3	0	1	0	5	11	2.8	6	1



03 분석 및 모델

- Keras를 활용한 Deep Learning
- Pycaret을 활용한 Machine Learning

■ Keras를 활용한 Deep Learning

```
[116] # Import tensorflow

from tensorflow.keras import models, layers, activations, initializers, losses, optimizers, metrics

[117] # model 생성

model = models.Sequential()

model.add(layers.Dense(input_dim=168, units=256, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.Activation('elu'))

model.add(layers.Dense(units=512, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.Activation('elu'))

model.add(layers.Dense(units=512, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.Activation('elu'))

model.add(layers.Dense(units=256, activation=None, kernel_initializer=initializers.he_uniform()))
model.add(layers.Activation('elu'))

model.add(layers.Dropout(rate=0.5))

model.add(layers.Dense(units=2, activation='softmax'))

[118] # model compile

model.compile(optimizer=optimizers.Adam(),
              loss=losses.categorical_crossentropy,
              metrics=[metrics.categorical_accuracy,
                      metrics.Precision(),
                      metrics.Recall(),
                      metrics.FalsePositives(),
                      metrics.FalseNegatives()])
```

■ Pycaret을 활용한 Machine Learning

```
[154] data = df_im.sample(frac=0.7, random_state=786).reset_index(drop=True)
      data_unseen = df_im.drop(data.index).reset_index(drop=True)

      print('Data for Modeling: ' + str(data.shape))
      print('Unseen Data For Predictions: ' + str(data_unseen.shape))
```

```
Data for Modeling: (12610, 13)
Unseen Data For Predictions: (5404, 13)
```

```
[156] model = setup(data=data,
                    target='target',
                    categorical_features=['city', 'gender', 'relevent_experience', 'enrolled_university',
                                         'education_level', 'major_discipline', 'company_size', 'company_type'],
                    numeric_features=['city_development_index', 'experience', 'last_new_job', 'training_hours'],
                    fix_imbalance=True,
                    session_id=123)
```

```
[154] data = df_im.sample(frac=0.7, random_state=786).reset_index(drop=True)
      data_unseen = df_im.drop(data.index).reset_index(drop=True)

      print('Data for Modeling: ' + str(data.shape))
      print('Unseen Data For Predictions: ' + str(data_unseen.shape))

      Data for Modeling: (12610, 13)
      Unseen Data For Predictions: (5404, 13)

[156] model = setup(data=data,
                    target='target',
                    categorical_features=['city', 'gender', 'relevent_experience', 'enrolled_university',
                                         'education_level', 'major_discipline', 'company_size', 'company_type'],
                    numeric_features=['city_development_index', 'experience', 'last_new_job', 'training_hours'],
                    fix_imbalance=True,
                    session_id=123)
```

Setup Successfully Completed!		
	Description	Value
0	session_id	123
1	Target Type	Binary
2	Label Encoded	0: 0, 1: 1
3	Original Data	(12610, 13)
4	Missing Values	False
5	Numeric Features	4
6	Categorical Features	8
7	Ordinal Features	False
8	High Cardinality Features	False
9	High Cardinality Method	None
10	Sampled Data	(12610, 13)
11	Transformed Train Set	(8826, 161)
12	Transformed Test Set	(3784, 161)
13	Numeric Imputer	mean
14	Categorical Imputer	constant
15	Normalize	False
16	Normalize Method	None
17	Transformation	False
18	Transformation Method	None
19	PCA	False
20	PCA Method	None
21	PCA Components	None
22	Ignore Low Variance	False
23	Combine Rare Levels	False
24	Rare Level Threshold	None
25	Numeric Binning	False
26	Remove Outliers	False
27	Outliers Threshold	None
28	Remove Multicollinearity	False
29	Multicollinearity Threshold	None
30	Clustering	False
31	Clustering Iteration	None
32	Polynomial Features	False
33	Polynomial Degree	None
34	Trigonometry Features	False
35	Polynomial Threshold	None
36	Group Features	False
37	Feature Selection	False
38	Features Selection Threshold	None
39	Feature Interaction	False
40	Feature Ratio	False
41	Interaction Threshold	None
42	Fix Imbalance	
43	Fix Imbalance Method	SMOTE



04

분석결과 및 리뷰

- 분석결과
- 어려웠던 점
- 개선사항

■ 분석결과

Top Model 비교

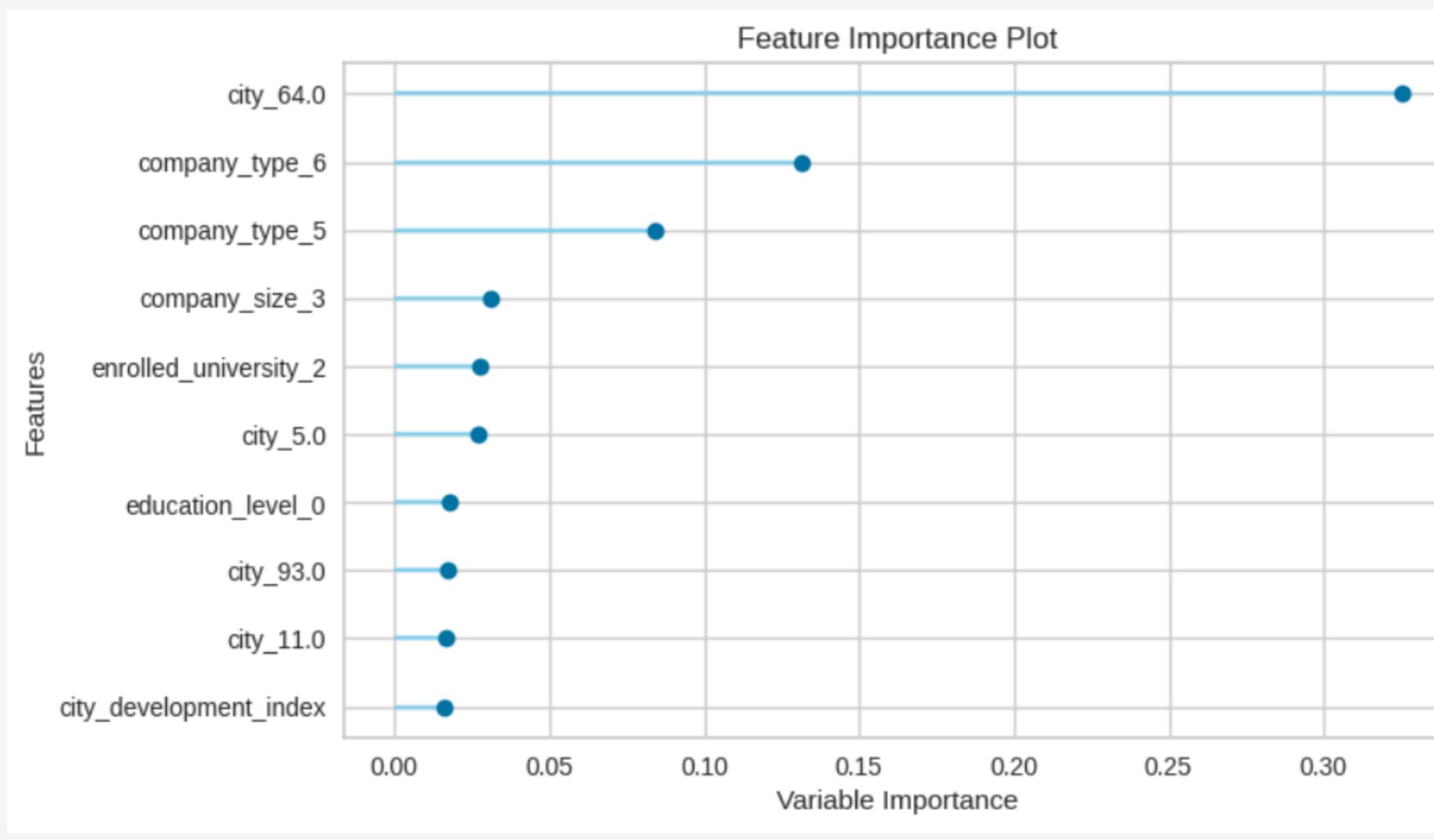


▶ top_3_models = compare_models(sort='Accuracy',
n_select = 3)

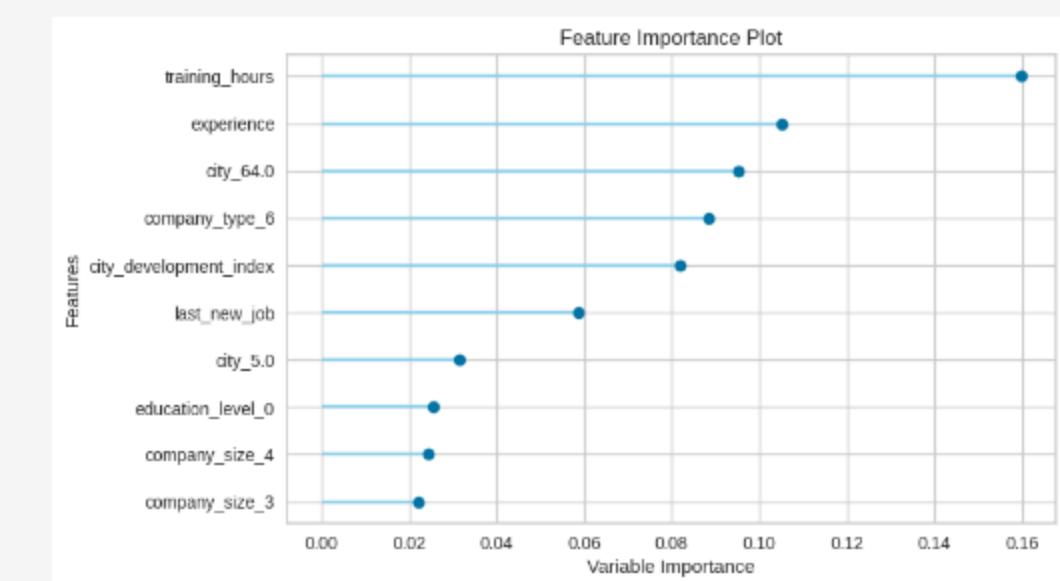
	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
0	Extreme Gradient Boosting	0.7879	0.7865	0.5123	0.5748	0.5415	0.4042	0.4055	
1	Gradient Boosting Classifier	0.7871	0.7859	0.5215	0.5714	0.5450	0.4065	0.4074	
2	Ada Boost Classifier	0.7844	0.7784	0.4697	0.5729	0.5157	0.3789	0.3823	
3	CatBoost Classifier	0.7843	0.7831	0.4974	0.5675	0.5299	0.3908	0.3924	
4	Light Gradient Boosting Machine	0.7811	0.7808	0.4743	0.5628	0.5146	0.3747	0.3770	
5	Extra Trees Classifier	0.7659	0.7415	0.4137	0.5277	0.4634	0.3167	0.3207	
6	Random Forest Classifier	0.7655	0.7359	0.3521	0.5326	0.4235	0.2840	0.2938	
7	Ridge Classifier	0.7642	0.0000	0.6973	0.5134	0.5911	0.4308	0.4411	
8	Linear Discriminant Analysis	0.7639	0.7829	0.6964	0.5129	0.5905	0.4299	0.4401	
9	Logistic Regression	0.7593	0.7826	0.7084	0.5063	0.5900	0.4264	0.4390	
10	Decision Tree Classifier	0.7038	0.6075	0.4188	0.4000	0.4088	0.2116	0.2118	
11	SVM - Linear Kernel	0.6502	0.0000	0.6361	0.4553	0.4577	0.2519	0.2968	
12	K Neighbors Classifier	0.5434	0.6045	0.6330	0.2971	0.4043	0.1067	0.1269	
13	Naive Bayes	0.4182	0.6254	0.8570	0.2776	0.4192	0.0782	0.1328	
14	Quadratic Discriminant Analysis	0.2664	0.5048	0.9727	0.2468	0.3937	0.0051	0.0226	

■ 분석결과

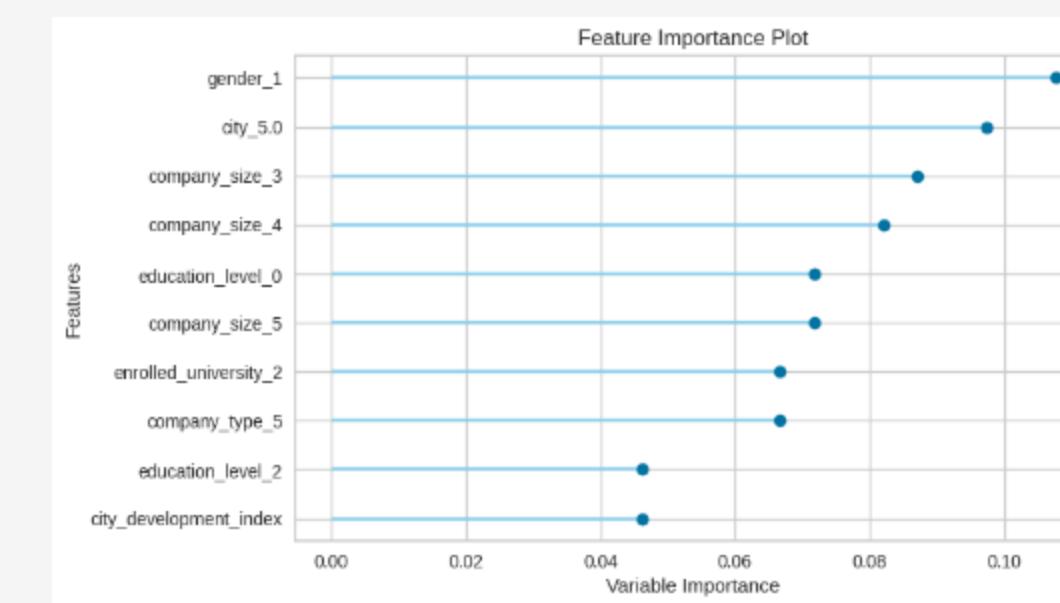
Top 3 Model - Feature Importance



Top 1 - XGB



Top 2 - GB



Top 3 - adaboost



End of Document