lab10

September 16, 2024

[135]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
```

[136]:
```
dataset = pd.read_csv("general_data.csv")
df = pd.DataFrame(dataset)
df.head()
```

[136]:

	Age	Attrition	BusinessTravel	Department	DistanceFromHome	\
0	51	No	Travel_Rarely	Sales	6	
1	31	Yes	Travel_Frequently	Research & Development	10	
2	32	No	Travel_Frequently	Research & Development	17	
3	38	No	Non-Travel	Research & Development	2	
4	32	No	Travel_Rarely	Research & Development	10	

	Education	EducationField	EmployeeCount	EmployeeID	Gender	…	\
0	2	Life Sciences	1	1	Female	…	
1	1	Life Sciences	1	2	Female	…	
2	4	Other	1	3	Male	…	
3	5	Life Sciences	1	4	Male	…	
4	1	Medical	1	5	Male	…	

	NumCompaniesWorked	Over18	PercentSalaryHike	StandardHours	\
0	1.0	Y	11	8	
1	0.0	Y	23	8	
2	1.0	Y	15	8	
3	3.0	Y	11	8	
4	4.0	Y	12	8	

	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	YearsAtCompany	\
0	0	1.0	6	1	
1	1	6.0	3	5	
2	3	5.0	2	5	
3	3	13.0	5	8	
4	2	9.0	2	6	

	YearsSinceLastPromotion	YearsWithCurrManager

```
0                       0                       0
1                       1                       4
2                       0                       3
3                       7                       5
4                       0                       4

[5 rows x 24 columns]
```

[137]:
```
df.isnull().sum()
```

[137]:
```
Age                       0
Attrition                 0
BusinessTravel            0
Department                0
DistanceFromHome          0
Education                 0
EducationField            0
EmployeeCount             0
EmployeeID                0
Gender                    0
JobLevel                  0
JobRole                   0
MaritalStatus             0
MonthlyIncome             0
NumCompaniesWorked       19
Over18                    0
PercentSalaryHike         0
StandardHours             0
StockOptionLevel          0
TotalWorkingYears         9
TrainingTimesLastYear     0
YearsAtCompany            0
YearsSinceLastPromotion   0
YearsWithCurrManager      0
dtype: int64
```

[138]:
```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='mean')

num_companies_reshaped = df[['NumCompaniesWorked']].values

df['NumCompaniesWorked'] = imputer.fit_transform(num_companies_reshaped)

df['NumCompaniesWorked'] = df['NumCompaniesWorked'].squeeze()
```

[139]:
```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy='mean')

num_companies_reshaped = df[['TotalWorkingYears']].values

df['TotalWorkingYears'] = imputer.fit_transform(num_companies_reshaped)

df['TotalWorkingYears'] = df['TotalWorkingYears'].squeeze()
```

[140]:
```
df.isnull().sum()
```

[140]:
```
Age                       0
Attrition                 0
BusinessTravel            0
Department                0
DistanceFromHome          0
Education                 0
EducationField            0
EmployeeCount             0
EmployeeID                0
Gender                    0
JobLevel                  0
JobRole                   0
MaritalStatus             0
MonthlyIncome             0
NumCompaniesWorked        0
Over18                    0
PercentSalaryHike         0
StandardHours             0
StockOptionLevel          0
TotalWorkingYears         0
TrainingTimesLastYear     0
YearsAtCompany            0
YearsSinceLastPromotion   0
YearsWithCurrManager      0
dtype: int64
```

[141]:
```
df.columns
```

[141]:
```
Index(['Age', 'Attrition', 'BusinessTravel', 'Department', 'DistanceFromHome',
       'Education', 'EducationField', 'EmployeeCount', 'EmployeeID', 'Gender',
       'JobLevel', 'JobRole', 'MaritalStatus', 'MonthlyIncome',
       'NumCompaniesWorked', 'Over18', 'PercentSalaryHike', 'StandardHours',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'YearsAtCompany', 'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
```

[142]:
```
df.head()
```

[142]:

	Age	Attrition	BusinessTravel	Department	DistanceFromHome	\
0	51	No	Travel_Rarely	Sales	6	
1	31	Yes	Travel_Frequently	Research & Development	10	
2	32	No	Travel_Frequently	Research & Development	17	
3	38	No	Non-Travel	Research & Development	2	
4	32	No	Travel_Rarely	Research & Development	10	

	Education	EducationField	EmployeeCount	EmployeeID	Gender	…	\
0	2	Life Sciences	1	1	Female	…	
1	1	Life Sciences	1	2	Female	…	
2	4	Other	1	3	Male	…	
3	5	Life Sciences	1	4	Male	…	
4	1	Medical	1	5	Male	…	

	NumCompaniesWorked	Over18	PercentSalaryHike	StandardHours	\
0	1.0	Y	11	8	
1	0.0	Y	23	8	
2	1.0	Y	15	8	
3	3.0	Y	11	8	
4	4.0	Y	12	8	

	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear	YearsAtCompany	\
0	0	1.0	6	1	
1	1	6.0	3	5	
2	3	5.0	2	5	
3	3	13.0	5	8	
4	2	9.0	2	6	

	YearsSinceLastPromotion	YearsWithCurrManager
0	0	0
1	1	4
2	0	3
3	7	5
4	0	4

[5 rows x 24 columns]

[143]:
```
df_business = pd.get_dummies(df['BusinessTravel'], prefix='BusinessTravel').
↪astype(int)
df = pd.concat([df.drop('BusinessTravel', axis=1), df_business], axis=1)
```

[144]:
```
df_department = pd.get_dummies(df['Department'], prefix='Department').
↪astype(int)
df = pd.concat([df.drop('Department', axis=1), df_department], axis=1)
```

[145]:
```
df_education = pd.get_dummies(df['EducationField'], prefix='EducationField').
↪astype(int)
df = pd.concat([df.drop('EducationField', axis=1), df_education], axis=1)
```

[146]:
```
df_gender = pd.get_dummies(df['Gender'], prefix='Gender').astype(int)
df = pd.concat([df.drop('Gender', axis=1), df_gender], axis=1)
```

[147]:
```
df_job_level = pd.get_dummies(df['JobLevel'], prefix='JobLevel').astype(int)
df = pd.concat([df.drop('JobLevel', axis=1), df_job_level], axis=1)
```

[148]:
```
df_job_role = pd.get_dummies(df['JobRole'], prefix='JobRole').astype(int)
df = pd.concat([df.drop('JobRole', axis=1), df_job_role], axis=1)
```

[149]:
```
df_marital = pd.get_dummies(df['MaritalStatus'], prefix='MaritalStatus').
↪astype(int)
df = pd.concat([df.drop('MaritalStatus', axis=1), df_marital], axis=1)
```

[150]:
```
df['Over18'] = [1 if value == 'Y' else 0 for value in df['Over18']]
```

[151]:
```
df.head()
```

[151]:

	Age	Attrition	DistanceFromHome	Education	EmployeeCount	EmployeeID	\
0	51	No	6	2	1	1	
1	31	Yes	10	1	1	2	
2	32	No	17	4	1	3	
3	38	No	2	5	1	4	
4	32	No	10	1	1	5	

	MonthlyIncome	NumCompaniesWorked	Over18	PercentSalaryHike	…	\
0	131160	1.0	1	11	…	
1	41890	0.0	1	23	…	
2	193280	1.0	1	15	…	
3	83210	3.0	1	11	…	
4	23420	4.0	1	12	…	

	JobRole_Laboratory Technician	JobRole_Manager	\
0	0	0	
1	0	0	
2	0	0	
3	0	0	
4	0	0	

	JobRole_Manufacturing Director	JobRole_Research Director	\
0	0	0	
1	0	0	
2	0	0	
3	0	0	

```
4                                0                          0

   JobRole_Research Scientist  JobRole_Sales Executive  \
0                           0                        0
1                           1                        0
2                           0                        1
3                           0                        0
4                           0                        1

   JobRole_Sales Representative  MaritalStatus_Divorced  \
0                             0                       0
1                             0                       0
2                             0                       0
3                             0                       0
4                             0                       0

   MaritalStatus_Married  MaritalStatus_Single
0                      1                     0
1                      0                     1
2                      1                     0
3                      1                     0
4                      0                     1

[5 rows x 48 columns]
```

[152]:
```
X,y = df.drop('Attrition',axis=1),df['Attrition']
```

[153]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[154]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[155]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix

model = LogisticRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print(confusion_matrix(y_test, y_pred))
```

```
print(classification_report(y_test, y_pred))
print(model.score(X_test,y_test))
```

```
[[722  19]
 [120  21]]
              precision    recall  f1-score   support

          No       0.86      0.97      0.91       741
         Yes       0.53      0.15      0.23       141

    accuracy                           0.84       882
   macro avg       0.69      0.56      0.57       882
weighted avg       0.80      0.84      0.80       882

0.8424036281179138
```

[ ]: