MULTI COLLINEARITY

Multicollinearity

The Dummy Variable Trap leads to the problem known as **multicollinearity**.

Multicollinearity occurs where there is a dependency between the independent features.

Multicollinearity is a serious issue in machine learning models like Linear Regression and Logistic Regression.

So, in order to overcome the problem of multicollinearity, one of the dummy variables has to be dropped.

Here, I will practically demonstrate how the problem of multicollinearity is introduced after carrying out the one-hot encoding.

One of the common ways to check for multicollinearity is the Variance Inflation Factor (VIF):

VIF=1, Very Less Multicollinearity
VIF<5, Moderate Multicollinearity
VIF>5, Extreme Multicollinearity (This is what we have to avoid)

Compute the VIF scores:

```
def calculate_vif(data):
    vif_df = pd.DataFrame(columns = ['Var', 'Vif'])
    x_var_names = data.columns
    for i in range(0, x_var_names.shape[0]):
        y = data[x_var_names[i]]
        x = data[x_var_names.drop([x_var_names[i]])]
        r_squared = sm.OLS(y,x).fit().rsquared
        vif = round(1/(1-r_squared),2)
        vif_df.loc[i] = [x_var_names[i], vif]
    return vif_df.sort_values(by = 'Vif', axis = 0, ascending=False,
inplace=False)


X=df.drop(['Salary'],axis=1)

calculate_vif(X)
```

Output:

	Var	V
2	2.0	9.9
0	0.0	9.8

From the output, we can see that the dummy variables which are created using one-hot encoding have VIF above 5.

We have a multicollinearity problem.
Now, let us drop one of the dummy variables to solve the multicollinearity issue:

```
df = df.drop(df.columns[[0]],axis=1)
calculate_vif(df)
```

Output:

	Var	
2	3.0	2
1	2.0	1

VIF has decreased. We solved the problem of multicollinearity.

Now, the dataset is ready for building the model.