Covariate:

1. Difference between etest_p and mba_p
   Covariance → 16.886973
   It is positive variance but very weak, it means the difference is very less.
2. Degree_p and etest_p
   Covariance → 22.078774
   It is also weak and the difference it less.
   If the variance is more than 50 then we can consider the difference make some significant variance.

Corelation:

1. Mba_p and salary

0.141417→ It is positive relation, but very less means when mba passmark is increased then the salary is also proportionally increased but the increasing percentage is very less, it is just 14%.

VIF(Variance Inflation Factor) calculation

VIF

```
#Finding VIF:
from statsmodels.stats.outliers_influence import variance_inflation_factor
def calc_vif(X):
    vif=pd.DataFrame()
    vif["variable"]=X.columns
    vif["VIF"]=[variance_inflation_factor(X.columns,i) for i in range(X.shape[1])]
    return(vif)
```

```
#Finding VIF:
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

Variance inflation factor is imported from statsmodels

```
def calc_vif(X):
```

Function name → calc_vif

Parameter → X

```
vif=pd.DataFrame()
```

→ Table creation by pandas in the name – vif

```
vif["variable"]=X.columns
```

Columns of the arguments will be taken as the variable in the vif table.

```
vif["VIF"]=[variance_inflation_factor(X.values,i) for i in range(X.shape[1])]
```

First for loop runs, Row values of X is stored in i and the I values are stored with arguments columns.

1. X.shape→ represent the columns that is been created in the above step pd.DataFrame()
2. Range(X.shape[1]) → return with the columns position values

3. For I in range→ the above values are stored in the i.
4. .values→ convert the row and columns with values that are returned by the for loop.
5. variance_inflation_factor→ it lists of VIF values for all the independent vlaues

<u>Multicollinearity</u>

What is multicollinearity:

Multicollinearity occurs when two or more independent variables are highly corelated, making it challenging to discern their separate effects on the target variable.

Problems of multucollineartiy:

Unstable coefficients (small changes in data can cause large changes in estimates).

Reduced interpretability (hard to determine which variable is truly influencing the outcome).

Higher standard errors, leading to less reliable statistical significance tests.

How to find it

VIF → Variance Influence Factor, if VIF > 5, multicollinearity is present. VIF is between 1 and 5, moderate multicollinearity is present. VIF is less than 1 then we can go with the model creation.

VIF formula → 1/1−*R***2 where *R**2* is from regressing a predictor

How to avoid:

Remove Highly Correlated Predictors: Drop one of the correlated variables.

Feature Engineering: Combine correlated variables (e.g., PCA, dimensionality reduction).

Collect More Data: More observations can help reduce the effect of multicollinearity.

Use Regularization Techniques: Ridge Regression (L2) or Lasso Regression (L1) can help shrink coefficients and handle multicollinearity.

Use Principal Component Analysis (PCA): Transform correlated variables into uncorrelated components.

Sample python code for finding VIF:

```
#Finding VIF:
import pandas as pd
import numpy as np
from statsmodels.stats.outliers_influence import variance_inflation_factor
def calc_vif(X):
    vif=pd.DataFrame()
    vif["variable"]=X.columns
    vif["VIF"]=[variance_inflation_factor(X.values,i) for i in range(X.shape[1])]
    return(vif)
```

```
calc_vif(dataset[['ssc_p', 'hsc_p', 'degree_p', 'etest_p', 'mba_p', 'salary']])
```

The first step will bring the list of VIF values.

When we run the calc_vif(dataset[['ssc_p', 'hsc_p', 'degree_p', 'etest_p', 'mba_p', 'salary']])

It returns with the variable(columns) with VIF values

: calc_vif(dataset[['ssc_p', 'hs

	variable	VIF
0	ssc_p	78.168671
1	hsc_p	61.882196
2	degree_p	114.820554
3	etest_p	32.720365
4	mba_p	116.034378
5	salary	4.171783

The VIF values should be lesser than 5 is ok to consider. But expect salary all the values are more than VIF -5.

We can remove the mba_p and degree_p, as it has more VIF values

```
calc_vif(dataset[['ssc_p', 'hsc_p', 'etest_p', 'salary']])
```

	variable	VIF
0	ssc_p	52.626622
1	hsc_p	47.783807
2	etest_p	27.957044
3	salary	3.507225

Still ssc_p and etest_p has more than the expected value of VIF, we can remove these variable.

9]:
```
calc_vif(dataset[[ 'etest_p', 'salary']])
```

9]:

	variable	VIF
0	etest_p	2.826904
1	salary	2.826904

Here we can use etest_p variable as an input for the target variable salary to create the ai model.