In [1]:
```
get_ipython().system('pip install naive-bayes')
```

```
Collecting naive-bayes
  Downloading naive-bayes-0.1.1.tar.gz (1.8 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Requirement already satisfied: scikit-learn>=0.15.2 in c:\users\rohit\anaconda3\lib\site-packages (from naive-b
ayes) (1.0.2)
Requirement already satisfied: numpy>=1.9.0 in c:\users\rohit\anaconda3\lib\site-packages (from naive-bayes) (1
.21.5)
Requirement already satisfied: joblib>=0.11 in c:\users\rohit\anaconda3\lib\site-packages (from scikit-learn>=0
.15.2->naive-bayes) (1.1.0)
Requirement already satisfied: scipy>=1.1.0 in c:\users\rohit\anaconda3\lib\site-packages (from scikit-learn>=0
.15.2->naive-bayes) (1.9.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\rohit\anaconda3\lib\site-packages (from scikit-
learn>=0.15.2->naive-bayes) (2.2.0)
Building wheels for collected packages: naive-bayes
  Building wheel for naive-bayes (setup.py): started
  Building wheel for naive-bayes (setup.py): finished with status 'done'
  Created wheel for naive-bayes: filename=naive_bayes-0.1.1-py3-none-any.whl size=2153 sha256=07da0b6789c8375d0
40d050dee6ed2ab44c50d25b0bc771d9ac7869576d9cc21
  Stored in directory: c:\users\rohit\appdata\local\pip\cache\wheels\29\fc\1f\44ea7f6ccb226e209a1fa1c4cfd6cddef
a75e1691504542ae5
Successfully built naive-bayes
Installing collected packages: naive-bayes
Successfully installed naive-bayes-0.1.1
```

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.naive_bayes import MultinomialNB as MB
from sklearn.naive_bayes import GaussianNB as GB
```

In [4]:
```
train=pd.read_csv('SalaryData_Train.csv')
train
```

Out[4]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek
0	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40
1	50	Self-emp-not-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13
2	38	Private	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40
3	53	Private	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40
4	28	Private	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40
...	...	...	...	...	...	...	...	...	...	...	...	...
30156	27	Private	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White	Female	0	0	38
30157	40	Private	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0	0	40
30158	58	Private	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Female	0	0	40
30159	22	Private	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Male	0	0	20
30160	52	Self-emp-inc	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Female	15024	0	40

30161 rows × 14 columns

In [5]:
```
test=pd.read_csv('SalaryData_Test.csv')
test
```

Out[5]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperwee
0	25	Private	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male	0	0	4
1	38	Private	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male	0	0	5
2	28	Local-gov	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male	0	0	4
3	44	Private	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male	7688	0	4
4	34	Private	10th	6	Never-married	Other-service	Not-in-family	White	Male	0	0	3
...	...	...	...	...	...	...	...	...	...	...	...	
15055	33	Private	Bachelors	13	Never-married	Prof-specialty	Own-child	White	Male	0	0	4
15056	39	Private	Bachelors	13	Divorced	Prof-specialty	Not-in-family	White	Female	0	0	3
15057	38	Private	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	White	Male	0	0	5
15058	44	Private	Bachelors	13	Divorced	Adm-clerical	Own-child	Asian-Pac-Islander	Male	5455	0	4
15059	35	Self-emp-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	6

15060 rows × 14 columns

In [6]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30161 entries, 0 to 30160
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   age            30161 non-null  int64 
 1   workclass      30161 non-null  object
 2   education      30161 non-null  object
 3   educationno    30161 non-null  int64 
 4   maritalstatus  30161 non-null  object
 5   occupation     30161 non-null  object
 6   relationship   30161 non-null  object
 7   race           30161 non-null  object
 8   sex            30161 non-null  object
 9   capitalgain    30161 non-null  int64 
 10  capitalloss    30161 non-null  int64 
 11  hoursperweek   30161 non-null  int64 
 12  native         30161 non-null  object
 13  Salary         30161 non-null  object
dtypes: int64(5), object(9)
memory usage: 3.2+ MB
```

In [7]:
```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15060 entries, 0 to 15059
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   age            15060 non-null  int64 
 1   workclass      15060 non-null  object
 2   education      15060 non-null  object
 3   educationno    15060 non-null  int64 
 4   maritalstatus  15060 non-null  object
 5   occupation     15060 non-null  object
 6   relationship   15060 non-null  object
 7   race           15060 non-null  object
 8   sex            15060 non-null  object
 9   capitalgain    15060 non-null  int64 
 10  capitalloss    15060 non-null  int64 
 11  hoursperweek   15060 non-null  int64 
 12  native         15060 non-null  object
 13  Salary         15060 non-null  object
dtypes: int64(5), object(9)
memory usage: 1.6+ MB
```

In [8]:
```
test.describe().round(2).style.background_gradient(cmap = 'Reds')
```

Out[8]:

	age	educationno	capitalgain	capitalloss	hoursperweek
count	15060.000000	15060.000000	15060.000000	15060.000000	15060.000000
mean	38.770000	10.110000	1120.300000	89.040000	40.950000
std	13.380000	2.560000	7703.180000	406.280000	12.060000
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	48.000000	13.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	3770.000000	99.000000

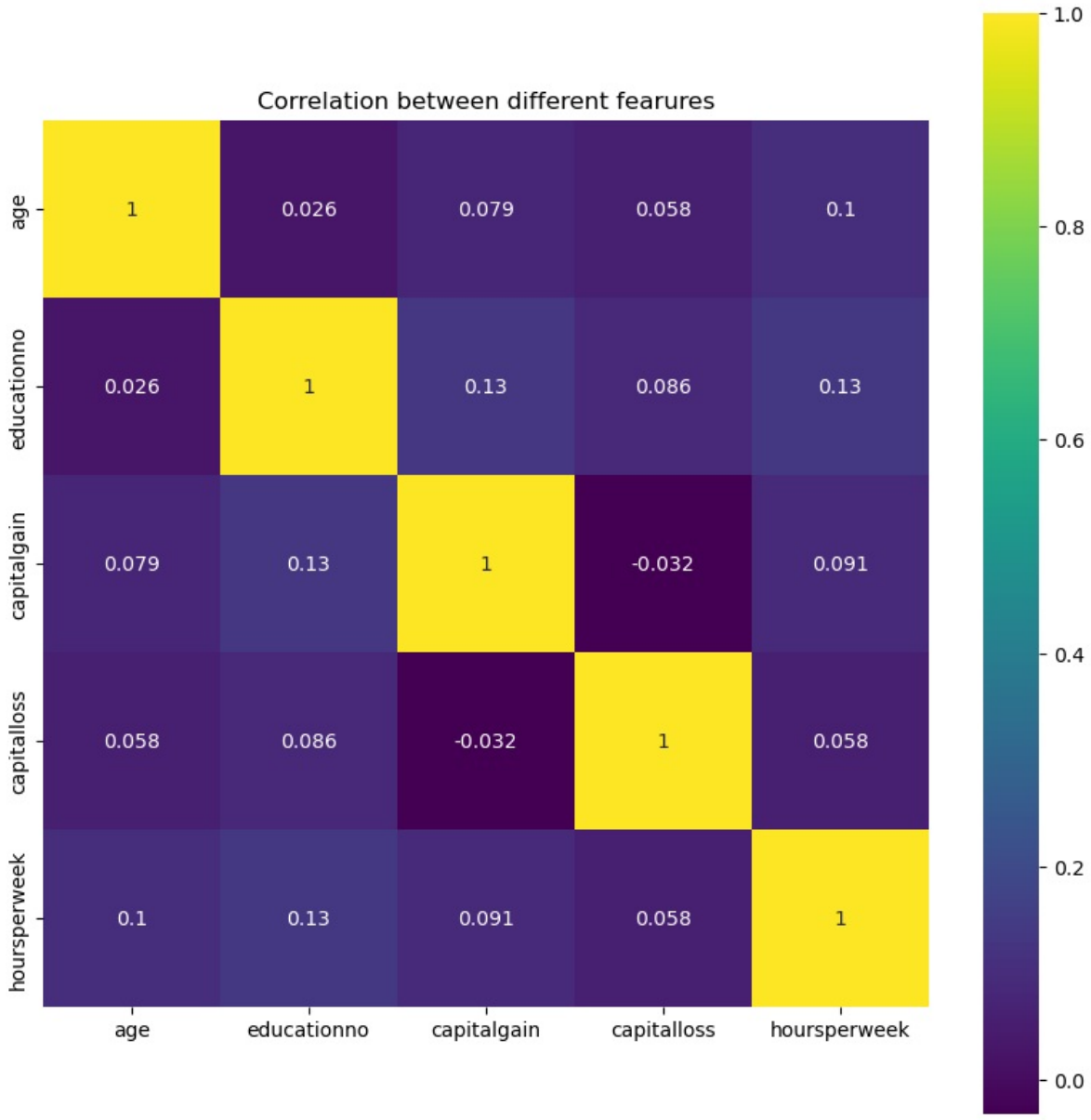
In [9]:
```
train.describe().round(2).style.background_gradient(cmap = 'Blues')
```

Out[9]:

	age	educationno	capitalgain	capitalloss	hoursperweek
count	30161.000000	30161.000000	30161.000000	30161.000000	30161.000000
mean	38.440000	10.120000	1092.040000	88.300000	40.930000
std	13.130000	2.550000	7406.470000	404.120000	11.980000
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	47.000000	13.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	4356.000000	99.000000

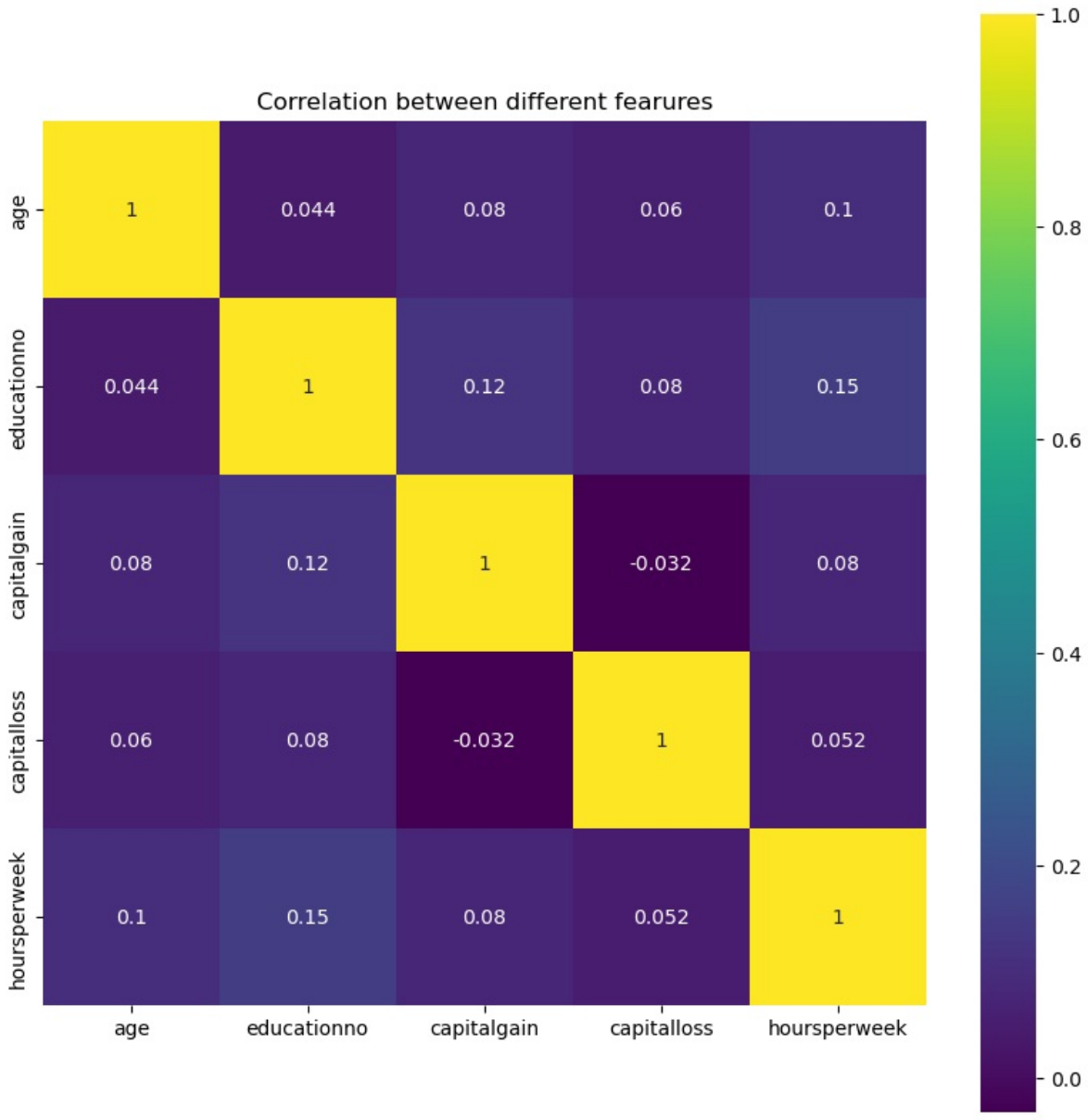
In [10]:
```
correlation = test.corr()
plt.figure(figsize=(10,10))
sns.heatmap(correlation, vmax=1, square=True,annot=True,cmap='viridis')
plt.title('Correlation between different fearures')
```

Out[10]:
```
Text(0.5, 1.0, 'Correlation between different fearures')
```



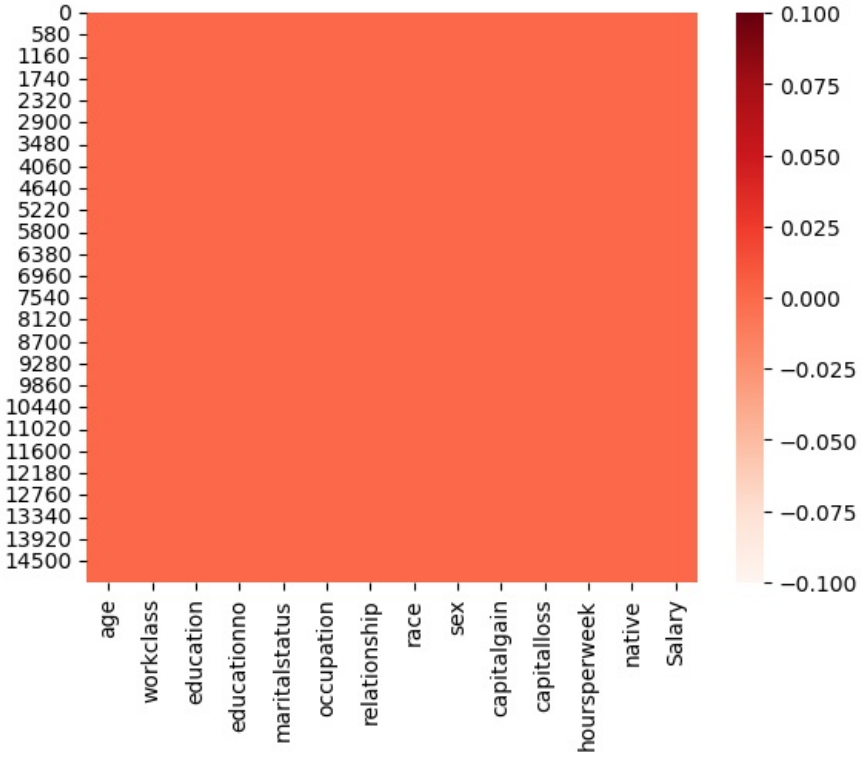
In [11]:
```
correlation = train.corr()
plt.figure(figsize=(10,10))
sns.heatmap(correlation, vmax=1, square=True,annot=True,cmap='viridis')
plt.title('Correlation between different fearures')
```

Out[11]:
```
Text(0.5, 1.0, 'Correlation between different fearures')
```



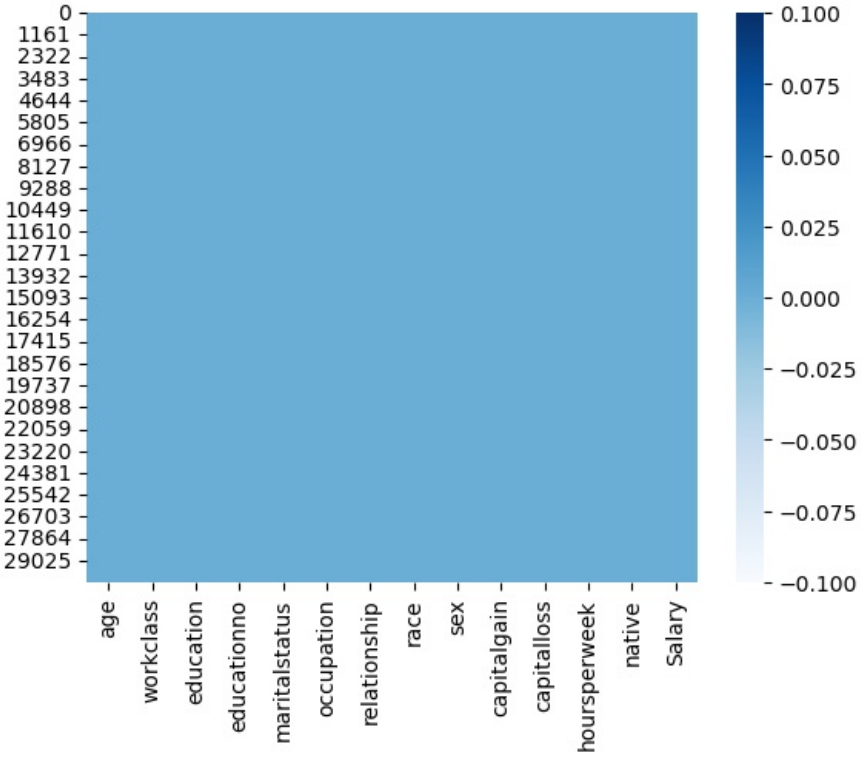
In [12]:
```
sns.heatmap(test.isnull(),cmap='Reds')
```

Out[12]:
```
<AxesSubplot:>
```



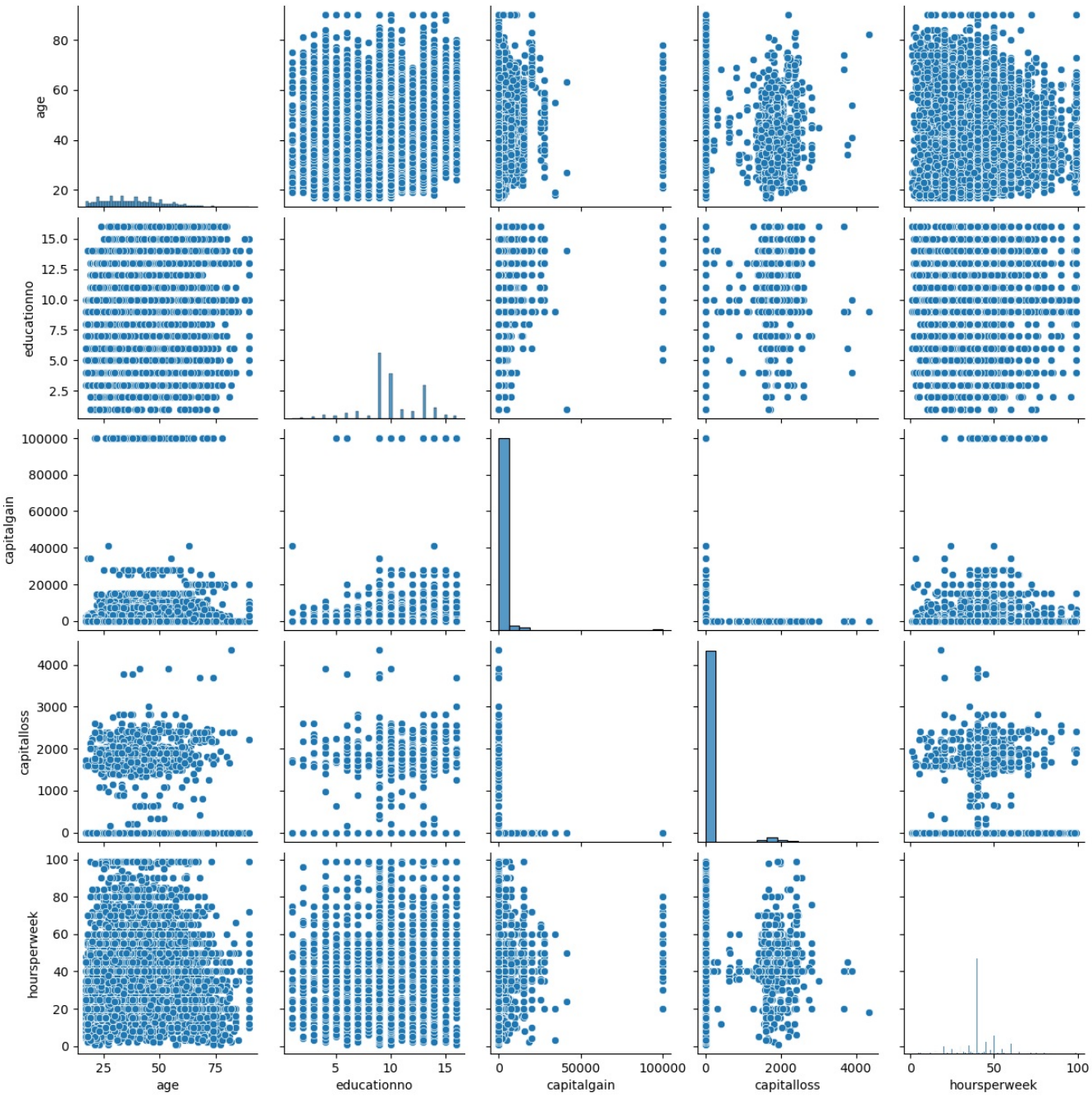
In [13]:
```
sns.heatmap(train.isnull(),cmap='Blues')
```

Out[13]:
```
<AxesSubplot:>
```



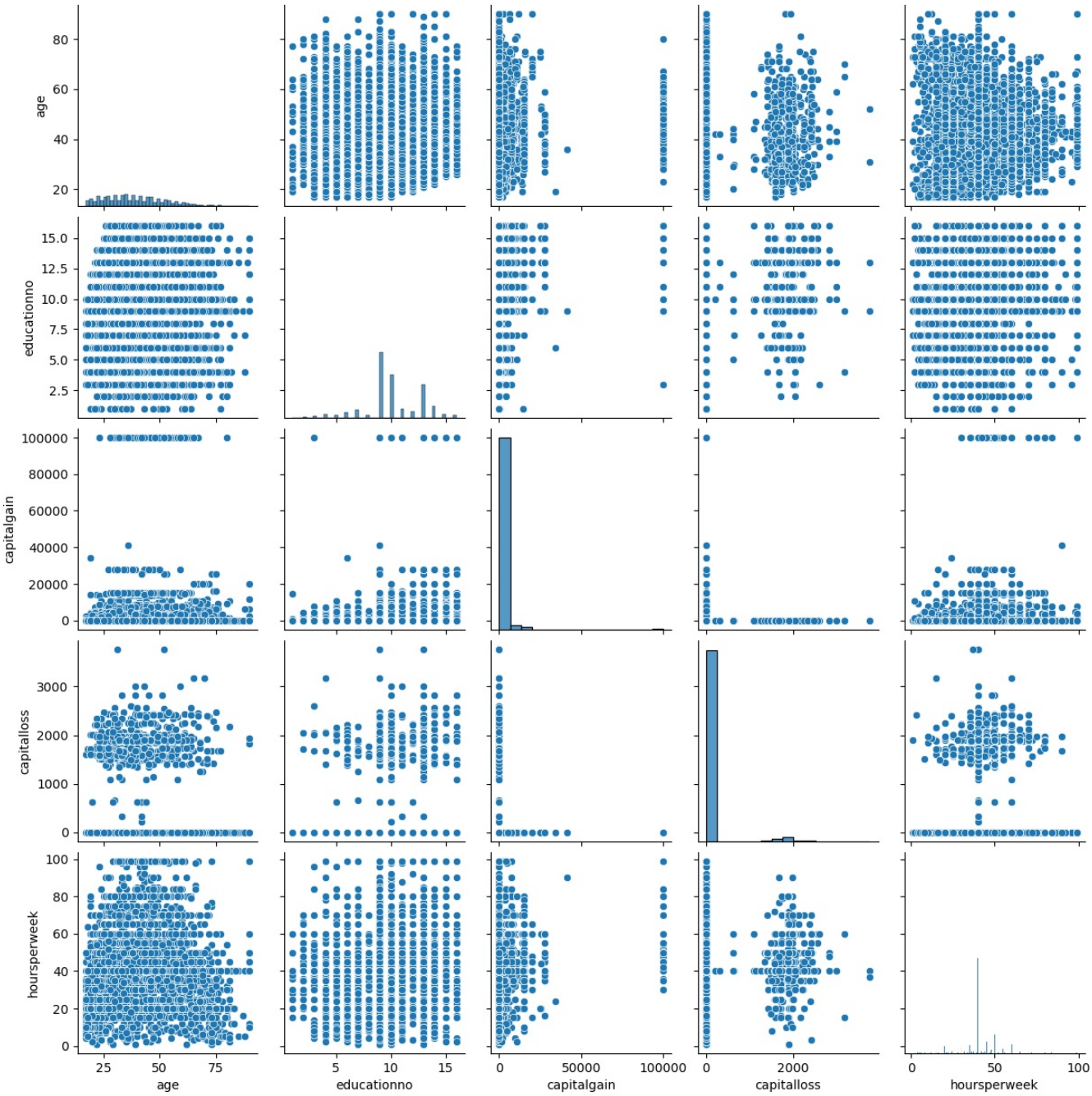
In [14]:
```
sns.pairplot(train)
```

Out[14]:
```
<seaborn.axisgrid.PairGrid at 0x2368658bd60>
```



In [15]:
```
sns.pairplot(test)
```

Out[15]:
```
<seaborn.axisgrid.PairGrid at 0x236896540a0>
```



In [16]:
```
train[['workclass','education','maritalstatus','occupation','relationship','race','sex','native','Salary']] = t
train
```

Out[16]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek	nat
0	39	0	0	13	0	0	0	0	0	2174	0	40	
1	50	1	0	13	1	1	1	0	0	0	0	13	
2	38	2	1	9	2	2	0	0	0	0	0	40	
3	53	2	2	7	1	2	1	1	0	0	0	40	
4	28	2	0	13	1	3	2	1	1	0	0	40	
...	...	...	...	...	...	...	...	...	...	...	...	...	
30156	27	2	6	12	1	9	2	0	1	0	0	38	
30157	40	2	1	9	1	8	1	0	0	0	0	40	
30158	58	2	1	9	6	0	4	0	1	0	0	40	
30159	22	2	1	9	0	0	3	0	0	0	0	20	
30160	52	5	1	9	1	1	2	0	1	15024	0	40	

30161 rows × 14 columns

In [17]:
```
test[['workclass','education','maritalstatus','occupation','relationship','race','sex','native','Salary']] = te
test
```

Out[17]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek	nat
0	25	0	0	7	0	0	0	0	0	0	0	40	
1	38	0	1	9	1	1	1	1	0	0	0	50	
2	28	1	2	12	1	2	1	1	0	0	0	40	
3	44	0	3	10	1	0	1	0	0	7688	0	40	
4	34	0	4	6	0	3	2	1	0	0	0	30	
...	...	...	...	...	...	...	...	...	...	...	...	...	
15055	33	0	7	13	0	4	0	1	0	0	0	40	
15056	39	0	7	13	4	4	2	1	1	0	0	36	
15057	38	0	7	13	1	4	1	1	0	0	0	50	
15058	44	0	7	13	4	6	0	4	0	5455	0	40	
15059	35	5	7	13	1	7	1	1	0	0	0	60	

15060 rows × 14 columns

In [21]:
```
X=train.iloc[:,0:13]
Y=train.iloc[:,13]
x=test.iloc[:,0:13]
y=test.iloc[:,13]
```

In [22]:
```
X
```

Out[22]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek	nat
0	39	0	0	13	0	0	0	0	0	2174	0	40	
1	50	1	0	13	1	1	1	0	0	0	0	13	
2	38	2	1	9	2	2	0	0	0	0	0	40	
3	53	2	2	7	1	2	1	1	0	0	0	40	
4	28	2	0	13	1	3	2	1	1	0	0	40	
...	...	...	...	...	...	...	...	...	...	...	...	...	
30156	27	2	6	12	1	9	2	0	1	0	0	38	
30157	40	2	1	9	1	8	1	0	0	0	0	40	
30158	58	2	1	9	6	0	4	0	1	0	0	40	
30159	22	2	1	9	0	0	3	0	0	0	0	20	
30160	52	5	1	9	1	1	2	0	1	15024	0	40	

30161 rows × 13 columns

In [23]:
```
Y
```

Out[23]:
```
0        0
1        0
2        0
3        0
4        0
        ..
30156    0
30157    1
30158    0
30159    0
30160    1
Name: Salary, Length: 30161, dtype: int64
```

In [24]:
```
x
```

Out[24]:

	age	workclass	education	educationno	maritalstatus	occupation	relationship	race	sex	capitalgain	capitalloss	hoursperweek	nat
0	25	0	0	7	0	0	0	0	0	0	0	40	
1	38	0	1	9	1	1	1	1	0	0	0	50	
2	28	1	2	12	1	2	1	1	0	0	0	40	
3	44	0	3	10	1	0	1	0	0	7688	0	40	
4	34	0	4	6	0	3	2	1	0	0	0	30	
...	...	...	...	...	...	...	...	...	...	...	...	...	
15055	33	0	7	13	0	4	0	1	0	0	0	40	
15056	39	0	7	13	4	4	2	1	1	0	0	36	
15057	38	0	7	13	1	4	1	1	0	0	0	50	
15058	44	0	7	13	4	6	0	4	0	5455	0	40	
15059	35	5	7	13	1	7	1	1	0	0	0	60	

15060 rows × 13 columns

In [26]:
```
y
```

Out[26]:
```
0        0
1        0
2        1
3        1
4        0
        ..
15055    0
15056    0
15057    0
15058    0
15059    1
Name: Salary, Length: 15060, dtype: int64
```

In [27]:
```
classifier_mb = MB()
classifier_mb.fit(X,Y)
```

Out[27]:
```
MultinomialNB()
```

In [28]:
```
train_pred_m = classifier_mb.predict(X)
accuracy_train_m = np.mean(train_pred_m==Y)
```

In [29]:
```
test_pred_m = classifier_mb.predict(x)
accuracy_test_m = np.mean(test_pred_m==y)
```

In [30]:
```
print('Training accuracy is:',accuracy_train_m,'\n','Testing accuracy is:',accuracy_test_m)
```

```
Training accuracy is: 0.7729186698053778
 Testing accuracy is: 0.7749667994687915
```

In [31]:
```
classifier_gb = GB()
classifier_gb.fit(X,Y)
```

Out[31]:
```
GaussianNB()
```

In [32]:
```
train_pred_g = classifier_gb.predict(X)
accuracy_train_g = np.mean(train_pred_g==Y)
```

In [33]:
```
test_pred_g = classifier_gb.predict(X)
accuracy_test_g = np.mean(test_pred_g==Y)
```

In [34]:
```
print('Training accuracy is:',accuracy_train_g,'\n','Testing accuracy is:',accuracy_test_g)
```

```
Training accuracy is: 0.8000066310798714
 Testing accuracy is: 0.8000066310798714
```

In [ ]:
```

```