In [21]:

```
import numpy as np
import pandas as pd
from sklearn import preprocessing
from sklearn import metrics
import seaborn as sns
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from matplotlib import pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder
from sklearn import preprocessing
from mlxtend.plotting import plot_decision_regions
```

executed in 262ms, finished 10:57:34 2021-11-27

In [22]:

```
test_tmp = pd.read_csv("SalaryData_Test(1).csv")
train_tmp = pd.read_csv("SalaryData_Train(1).csv")
```

executed in 194ms, finished 10:58:18 2021-11-27

In [23]:

```
df_tmp = test_tmp.append(train_tmp)
test = test_tmp.copy()
train = train_tmp.copy()
test.head()
```

executed in 78ms, finished 10:58:37 2021-11-27

Out[23]:

| | age | workclass | education | educationno | maritalstatus | occupation | relationship | race | sex |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 25 | Private | 11th | 7 | Never-married | Machine-op-inspct | Own-child | Black | Male |
| **1** | 38 | Private | HS-grad | 9 | Married-civ-spouse | Farming-fishing | Husband | White | Male |
| **2** | 28 | Local-gov | Assoc-acdm | 12 | Married-civ-spouse | Protective-serv | Husband | White | Male |
| **3** | 44 | Private | Some-college | 10 | Married-civ-spouse | Machine-op-inspct | Husband | Black | Male |
| **4** | 34 | Private | 10th | 6 | Never-married | Other-service | Not-in-family | White | Male |

In [24]:

```
train.head()
```

executed in 42ms, finished 10:58:54 2021-11-27

Out[24]:

| | age | workclass | education | educationno | maritalstatus | occupation | relationship | race | s |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 39 | State-gov | Bachelors | 13 | Never-married | Adm-clerical | Not-in-family | White | Ma |
| **1** | 50 | Self-emp-not-inc | Bachelors | 13 | Married-civ-spouse | Exec-managerial | Husband | White | Ma |
| **2** | 38 | Private | HS-grad | 9 | Divorced | Handlers-cleaners | Not-in-family | White | Ma |
| **3** | 53 | Private | 11th | 7 | Married-civ-spouse | Handlers-cleaners | Husband | Black | Ma |
| **4** | 28 | Private | Bachelors | 13 | Married-civ-spouse | Prof-specialty | Wife | Black | Fema |

In [25]:

```
str_c = ["workclass","education","maritalstatus","occupation","relationship","race","sex","
number = LabelEncoder()
for i in str_c:
    train[i]= number.fit_transform(train[i])
    test[i]=number.fit_transform(test[i])
test.head()
```

executed in 219ms, finished 10:59:13 2021-11-27

Out[25]:

| | age | workclass | education | educationno | maritalstatus | occupation | relationship | race | sex |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 25 | 2 | 1 | 7 | 4 | 6 | 3 | 2 | 1 |
| **1** | 38 | 2 | 11 | 9 | 2 | 4 | 0 | 4 | 1 |
| **2** | 28 | 1 | 7 | 12 | 2 | 10 | 0 | 4 | 1 |
| **3** | 44 | 2 | 15 | 10 | 2 | 6 | 0 | 2 | 1 |
| **4** | 34 | 2 | 0 | 6 | 4 | 7 | 1 | 4 | 1 |

In [26]:

```
train.head()
```

executed in 43ms, finished 10:59:32 2021-11-27

Out[26]:

| | age | workclass | education | educationno | maritalstatus | occupation | relationship | race | sex |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 39 | 5 | 9 | 13 | 4 | 0 | 1 | 4 | 1 |
| **1** | 50 | 4 | 9 | 13 | 2 | 3 | 0 | 4 | 1 |
| **2** | 38 | 2 | 11 | 9 | 0 | 5 | 1 | 4 | 1 |
| **3** | 53 | 2 | 1 | 7 | 2 | 5 | 0 | 2 | 1 |
| **4** | 28 | 2 | 9 | 13 | 2 | 9 | 5 | 2 | 0 |

In [27]:

```
mapping = {' >50K': 1, ' <=50K': 2}
train = train.replace({'Salary': mapping})
test = test.replace({'Salary': mapping})
df = train.append(test)
df1 = df.copy()
df1.head()
```

executed in 96ms, finished 10:59:51 2021-11-27

Out[27]:

| | age | workclass | education | educationno | maritalstatus | occupation | relationship | race | sex |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 39 | 5 | 9 | 13 | 4 | 0 | 1 | 4 | 1 |
| **1** | 50 | 4 | 9 | 13 | 2 | 3 | 0 | 4 | 1 |
| **2** | 38 | 2 | 11 | 9 | 0 | 5 | 1 | 4 | 1 |
| **3** | 53 | 2 | 1 | 7 | 2 | 5 | 0 | 2 | 1 |
| **4** | 28 | 2 | 9 | 13 | 2 | 9 | 5 | 2 | 0 |

In [28]:

```
df1.shape
```

executed in 19ms, finished 11:00:07 2021-11-27

Out[28]:

```
(45221, 14)
```

In [29]:

```
df1.describe().T
```

executed in 132ms, finished 11:00:18 2021-11-27

Out[29]:

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **age** | 45221.0 | 38.548086 | 13.217981 | 17.0 | 28.0 | 37.0 | 47.0 | 90.0 |
| **workclass** | 45221.0 | 2.204507 | 0.958132 | 0.0 | 2.0 | 2.0 | 2.0 | 6.0 |
| **education** | 45221.0 | 10.313217 | 3.816992 | 0.0 | 9.0 | 11.0 | 12.0 | 15.0 |
| **educationno** | 45221.0 | 10.118463 | 2.552909 | 1.0 | 9.0 | 10.0 | 13.0 | 16.0 |
| **maritalstatus** | 45221.0 | 2.585148 | 1.500460 | 0.0 | 2.0 | 2.0 | 4.0 | 6.0 |
| **occupation** | 45221.0 | 5.969572 | 4.026444 | 0.0 | 2.0 | 6.0 | 9.0 | 13.0 |
| **relationship** | 45221.0 | 1.412684 | 1.597242 | 0.0 | 0.0 | 1.0 | 3.0 | 5.0 |
| **race** | 45221.0 | 3.680281 | 0.832361 | 0.0 | 4.0 | 4.0 | 4.0 | 4.0 |
| **sex** | 45221.0 | 0.675062 | 0.468357 | 0.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| **capitalgain** | 45221.0 | 1101.454700 | 7506.511295 | 0.0 | 0.0 | 0.0 | 0.0 | 99999.0 |
| **capitalloss** | 45221.0 | 88.548617 | 404.838249 | 0.0 | 0.0 | 0.0 | 0.0 | 4356.0 |
| **hoursperweek** | 45221.0 | 40.938038 | 12.007640 | 1.0 | 40.0 | 40.0 | 45.0 | 99.0 |
| **native** | 45221.0 | 35.431503 | 5.931380 | 0.0 | 37.0 | 37.0 | 37.0 | 39.0 |
| **Salary** | 45221.0 | 1.752151 | 0.431769 | 1.0 | 2.0 | 2.0 | 2.0 | 2.0 |

In [30]:

```
df1.isnull().sum()
```

executed in 23ms, finished 11:00:30 2021-11-27

Out[30]:

```
age             0
workclass       0
education       0
educationno     0
maritalstatus   0
occupation      0
relationship    0
race            0
sex             0
capitalgain     0
capitalloss     0
hoursperweek    0
native          0
Salary          0
dtype: int64
```

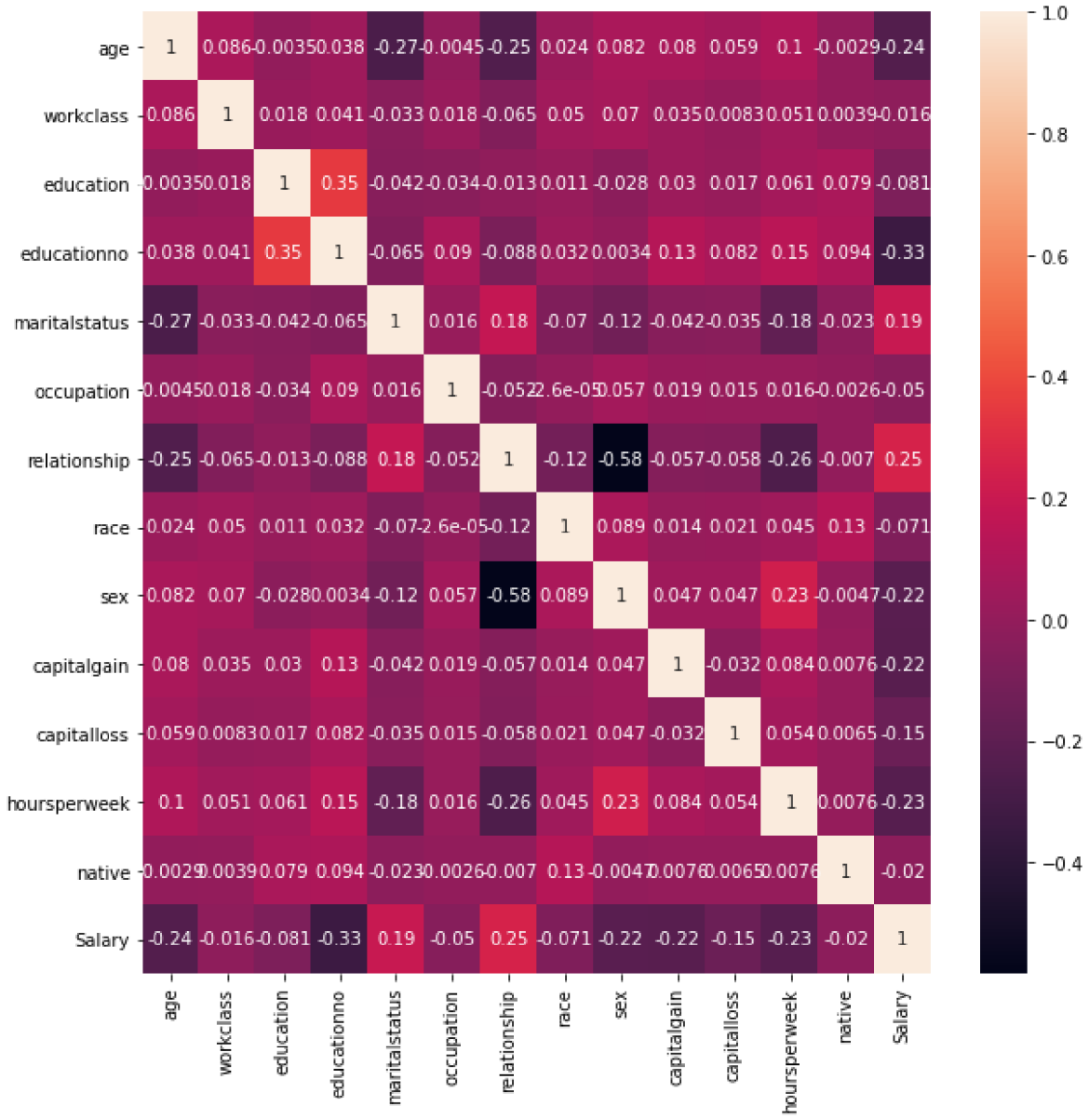
In [31]:

```
corr = df1.corr()
plt.figure(figsize=(10,10))
sns.heatmap(corr,annot=True)
```

executed in 4.08s, finished 11:00:49 2021-11-27

Out[31]:

<AxesSubplot:>

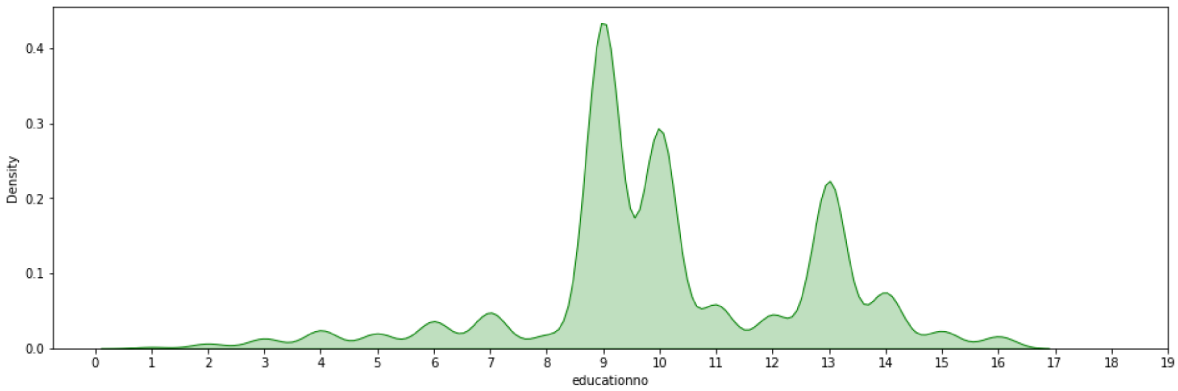


In [32]:

```
plt.rcParams["figure.figsize"] = 9,5
plt.figure(figsize=(16,5))
print("Skew: {}".format(df1['educationno'].skew()))
print("Kurtosis: {}".format(df1['educationno'].kurtosis()))
ax = sns.kdeplot(df1['educationno'],shade=True,color='g')
plt.xticks([i for i in range(0,20,1)])
plt.show()
```

executed in 929ms, finished 11:01:05 2021-11-27

```
Skew: -0.31062061074424
Kurtosis: 0.6350448194491634
```



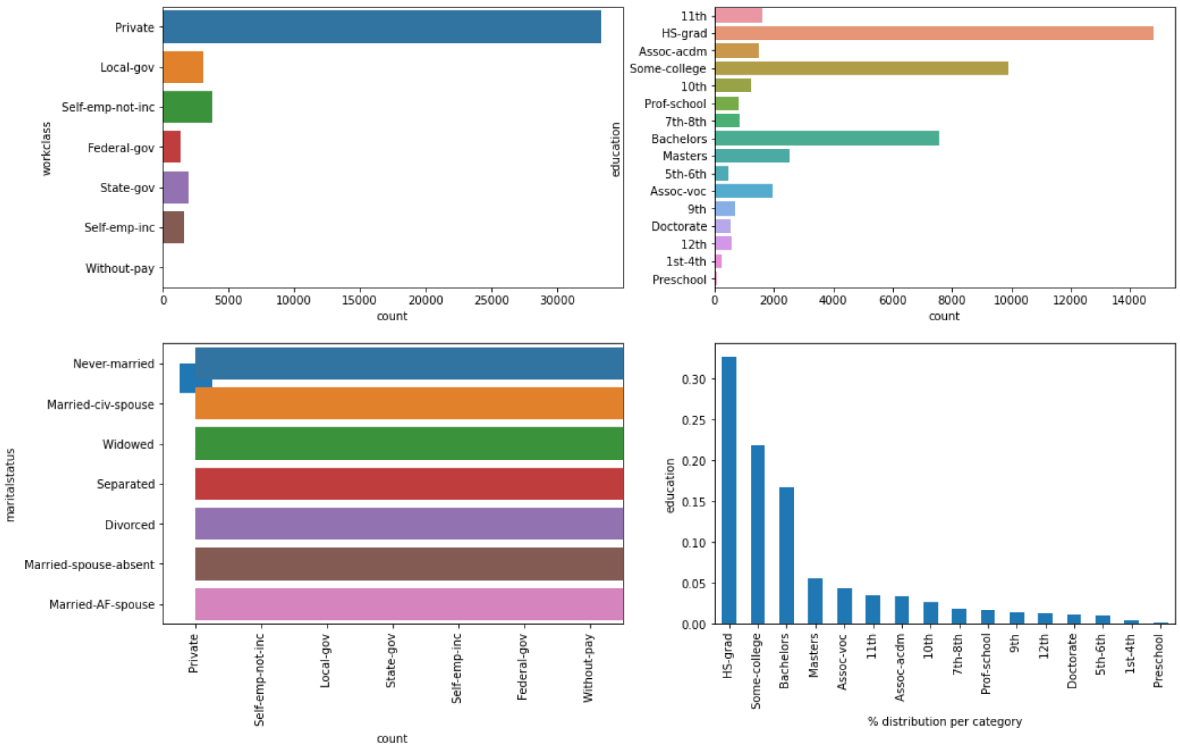
In [33]:

```
dfa = df_tmp[df_tmp.columns[0:13]]
obj_colum = dfa.select_dtypes(include='object').columns.tolist()
plt.figure(figsize=(16,10))
for i,col in enumerate(obj_colum,1):
    plt.subplot(2,2,i)
    sns.countplot(data=dfa,y=col)
    plt.subplot(2,2,i+2)
    df_tmp[col].value_counts(normalize=True).plot.bar()
    plt.ylabel(col)
    plt.xlabel('% distribution per category')
plt.tight_layout()
plt.show()
```

executed in 2.74s, finished 11:01:24 2021-11-27

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-33-e06cee192af2> in <module>
      5     plt.subplot(2,2,i)
      6     sns.countplot(data=dfa,y=col)
----> 7     plt.subplot(2,2,i+2)
      8     df_tmp[col].value_counts(normalize=True).plot.bar()
      9     plt.ylabel(col)

~\anaconda3\lib\site-packages\matplotlib\pyplot.py in subplot(*args, **kwargs)
   1140
   1141     fig = gcf()
-> 1142     ax = fig.add_subplot(*args, **kwargs)
   1143     bbox = ax.bbox
   1144     axes_to_delete = []

~\anaconda3\lib\site-packages\matplotlib\figure.py in add_subplot(self, *args, **kwargs)
   1400                     # more similar to add_axes.
   1401                     self._axstack.remove(ax)
-> 1402             ax = subplot_class_factory(projection_class)(self, *args, **kwargs)
   1403
   1404         return self._add_axes_internal(key, ax)

~\anaconda3\lib\site-packages\matplotlib\axes\_subplots.py in __init__(self, fig, *args, **kwargs)
     37
     38         self.figure = fig
---> 39         self._subplotspec = SubplotSpec._from_subplot_args(fig, args)
     40         self.update_params()
     41         # _axes_class is set in the subplot_class_factory

~\anaconda3\lib\site-packages\matplotlib\gridspec.py in _from_subplot_args(figure, args)
    687                     num = int(num)
    688                 if num < 1 or num > rows*cols:
--> 689                     raise ValueError(
    690                         f"num must be 1 <= num <= {rows*cols}, not {num}")
    691                 return gs[num - 1]   # -1 due to MATLAB indexing.
```

```
ValueError: num must be 1 <= num <= 4, not 5
```




In [34]:

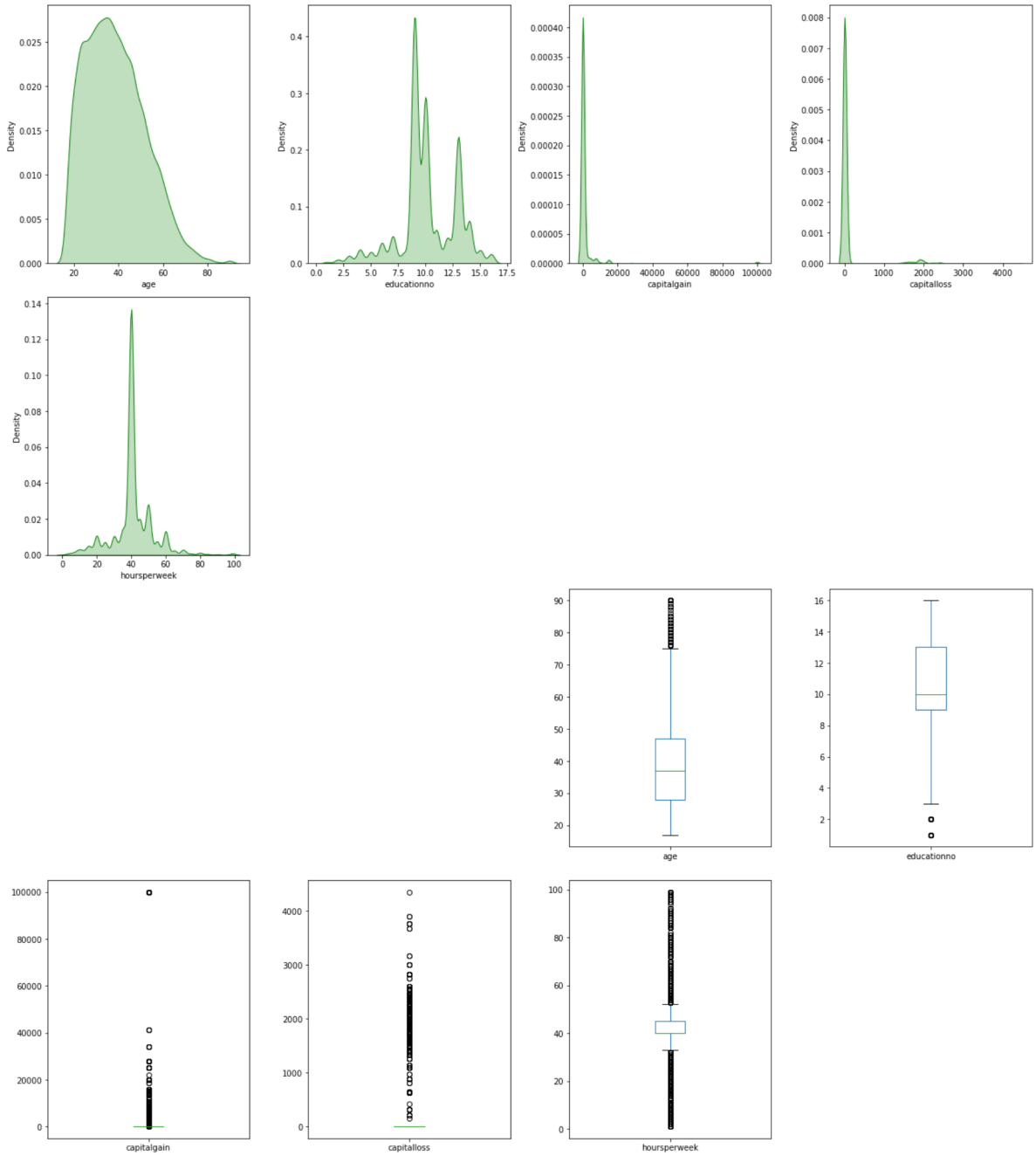
```
num_columns = dfa.select_dtypes(exclude='object').columns.tolist()
```

executed in 22ms, finished 11:01:58 2021-11-27

In [35]:

```
plt.figure(figsize=(18,40))
for i,col in enumerate(num_columns,1):
    plt.subplot(8,4,i)
    sns.kdeplot(df[col],color='g',shade=True)
    plt.subplot(8,4,i+10)
    df[col].plot.box()
plt.tight_layout()
plt.show()
num_data = df[num_columns]
pd.DataFrame(data=[num_data.skew(),num_data.kurtosis()],index=['skewness','kurtosis'])
```

executed in 7.08s, finished 11:02:19 2021-11-27



Out[35]:

| | age | educationno | capitalgain | capitalloss | hoursperweek |
|---|---|---|---|---|---|
| **skewness** | 0.532784 | -0.310621 | 11.788871 | 4.517536 | 0.340536 |
| **kurtosis** | -0.155931 | 0.635045 | 150.147899 | 19.376085 | 3.201287 |

In [36]:

```
col = df1.columns
x_train = train[col[0:13]]
y_train = train[col[13]]
x_test = test[col[0:13]]
y_test = test[col[13]]
def norm_func(i):
    x = (i-i.min())/(i.max()-i.min())
    return (x)
x_train = norm_func(x_train)
x_test =  norm_func(x_test)
```

executed in 82ms, finished 11:02:41 2021-11-27

In [39]:

```
model_linear = SVC(kernel = "linear")
model_linear.fit(x_train,y_train)
pred_test_linear = model_linear.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, pred_test_linear))
```

executed in 1m 1.23s, finished 11:06:00 2021-11-27

```
Accuracy: 0.8097609561752988
```

In [40]:

```
model_poly = SVC(kernel = "poly")
model_poly.fit(x_train,y_train)
pred_test_poly = model_poly.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, pred_test_poly))
```

executed in 1m 2.41s, finished 11:07:17 2021-11-27

```
Accuracy: 0.8435590969455511
```

In [41]:

```
model_rbf = SVC(kernel = "rbf")
model_rbf.fit(x_train,y_train)
pred_test_rbf = model_rbf.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, pred_test_rbf))
```

executed in 1m 54.8s, finished 11:09:41 2021-11-27

```
Accuracy: 0.8432934926958832
```

In [42]:

```
model_sigmoid = SVC(kernel = "sigmoid")
model_sigmoid.fit(x_train,y_train)
pred_test_sigmoid = model_sigmoid.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, pred_test_sigmoid))
```

executed in 2m 12s, finished 11:12:18 2021-11-27

```
Accuracy: 0.5768924302788845
```