In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:

```
ds = pd.read_csv('/home/yeshua/Downloads/employee_data.csv')
```

In [3]:

```
ds.describe()
```

Out[3]:

	Unnamed: 0	id	age	healthy_eating	active_lifestyle	salary
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000
mean	499.500000	499.500000	41.155000	4.944000	5.683000	2227.46100
std	288.819436	288.819436	13.462995	2.013186	2.048587	1080.20976
min	0.000000	0.000000	18.000000	0.000000	0.000000	553.00000
25%	249.750000	249.750000	30.000000	4.000000	4.000000	1360.00000
50%	499.500000	499.500000	41.000000	5.000000	6.000000	2174.00000
75%	749.250000	749.250000	53.000000	6.000000	7.000000	2993.75000
max	999.000000	999.000000	64.000000	10.000000	10.000000	5550.00000

In [4]:

```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
Unnamed: 0          1000 non-null int64
id                  1000 non-null int64
groups              1000 non-null object
age                 1000 non-null int64
healthy_eating      1000 non-null int64
active_lifestyle    1000 non-null int64
salary              1000 non-null int64
dtypes: int64(6), object(1)
memory usage: 54.8+ KB
```

In [5]:

```
ds.keys()
```

Out[5]:

```
Index(['Unnamed: 0', 'id', 'groups', 'age', 'healthy_eating',
       'active_lifestyle', 'salary'],
      dtype='object')
```

In [6]:

```
ds.head()
```

Out[6]:

	Unnamed: 0	id	groups	age	healthy_eating	active_lifestyle	salary

	Unnamed: 0	id	groups	age	healthy_eating	active_lifestyle	salary
0	0	0	A	36	5	5	2297
1	1	1	A	55	3	5	1134
2	2	2	A	61	8	1	4969
3	3	3	O	29	3	6	902
4	4	4	O	34	6	2	3574

In [7]:

```
ds.isnull().head()
```

Out[7]:

	Unnamed: 0	id	groups	age	healthy_eating	active_lifestyle	salary
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False

In [8]:

```
ds.shape
```

Out[8]:

```
(1000, 7)
```

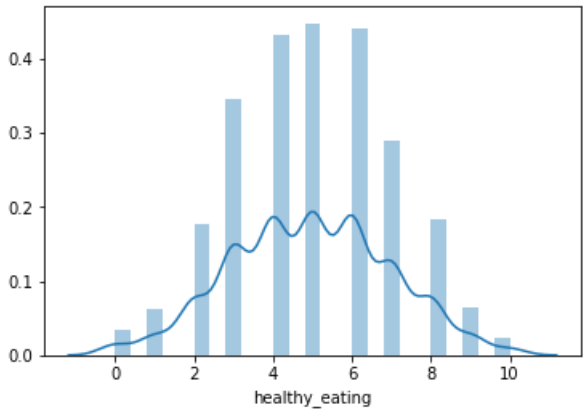
In [19]:

```
import seaborn as sns
sns.distplot(ds["healthy_eating"],hist=True)
```

```
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```

Out[19]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee8ef080>
```

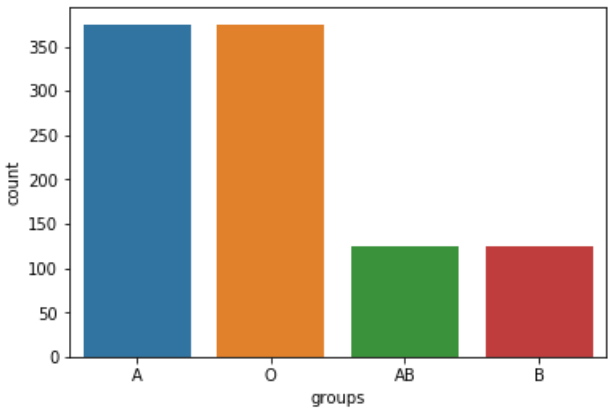


In [21]:

```
sns.countplot(ds["groups"])
```

Out[21]:

<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee589400>



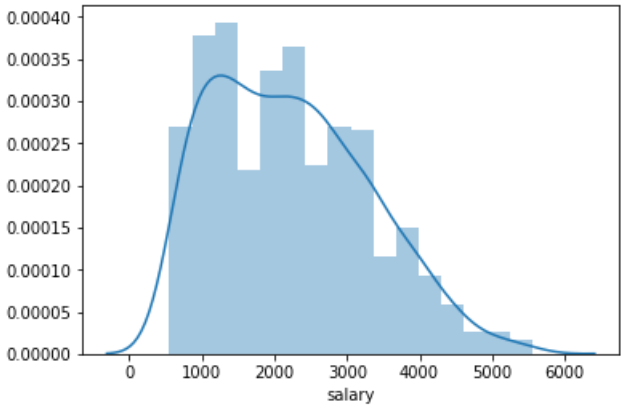
In [22]:

```
sns.distplot(ds["salary"])
```

```
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```

Out[22]:

<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee4ea780>

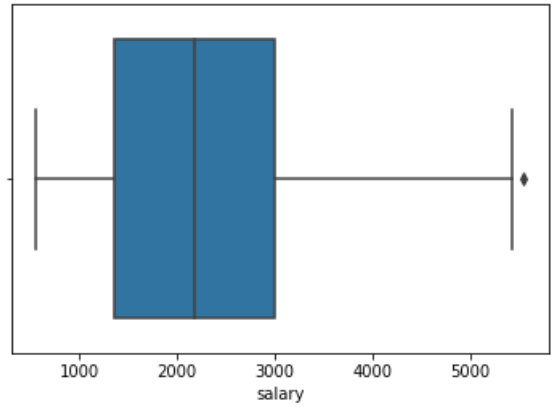


In [23]:

```
sns.boxplot(ds["salary"])
```

Out[23]:

<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee53a710>

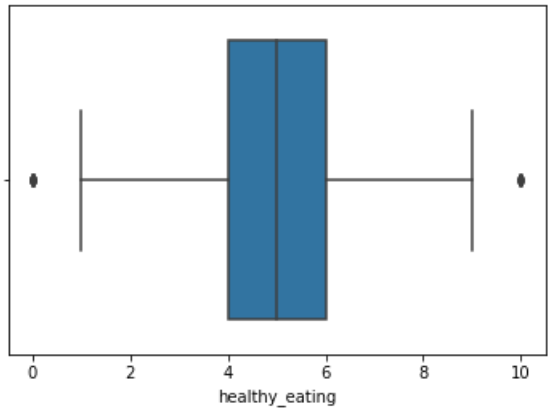


In [24]:

```
sns.boxplot(ds["healthy eating"])
```

Out[24]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee448c88>
```

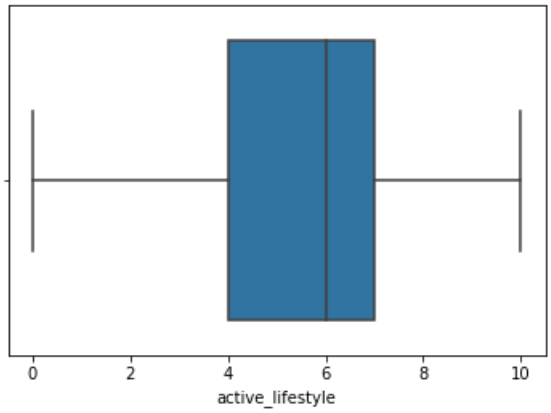


In [25]:

```
sns.boxplot(ds["active_lifestyle"])
```

Out[25]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee3ff4e0>
```

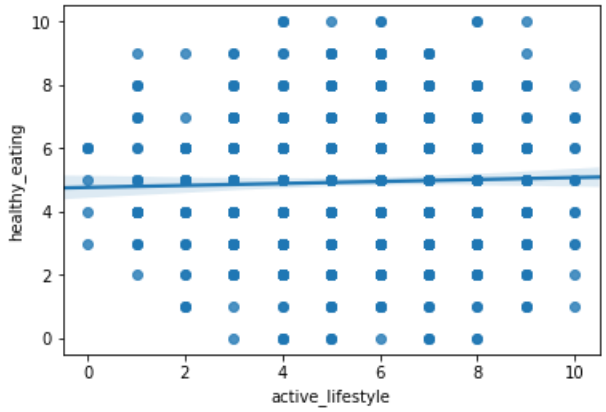


In [29]:

```
sns.regplot(x=ds["active_lifestyle"],y=ds["healthy_eating"]),
```

Out[29]:

```
(<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee4ea828>,)
```



In [27]:

```
corr = ds.corr()
corr
```

Out[27]:

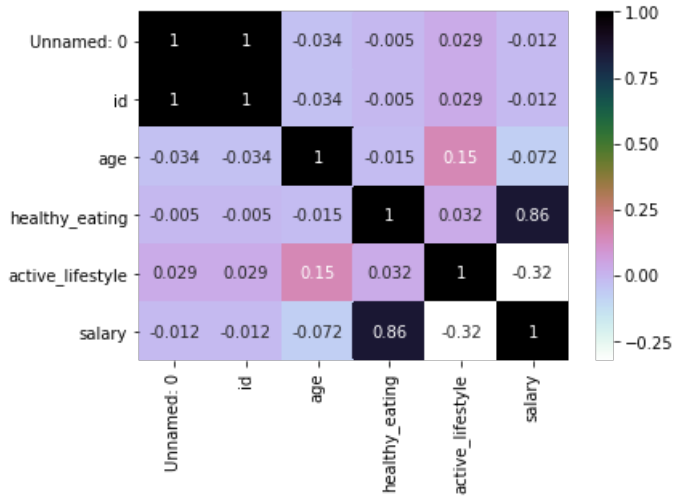
	Unnamed: 0	id	age	healthy_eating	active_lifestyle	salary
Unnamed: 0	1.000000	1.000000	-0.033595	-0.004993	0.028897	-0.012048
id	1.000000	1.000000	-0.033595	-0.004993	0.028897	-0.012048
age	-0.033595	-0.033595	1.000000	-0.014969	0.148267	-0.072231
healthy_eating	-0.004993	-0.004993	-0.014969	1.000000	0.031613	0.858405
active_lifestyle	0.028897	0.028897	0.148267	0.031613	1.000000	-0.323575
salary	-0.012048	-0.012048	-0.072231	0.858405	-0.323575	1.000000

In [28]:

```
sns.heatmap(corr,xticklabels=corr.columns.values,yticklabels=corr.columns.values,annot=True,cmap='cubehelix_r')
```

Out[28]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbdee0f22e8>
```



In [9]:

```
X = ds.iloc[:, :-2].values
y = ds.iloc[:, 6].values
```

In [10]:

```
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, random_state = 0)
```

```
/home/yeshua/anaconda3/lib/python3.6/site-packages/sklearn/cross_validation.py:41:
DeprecationWarning: This module was deprecated in version 0.18 in favor of the model_selection
module into which all the refactored classes and functions are moved. Also note that the interface
of the new CV iterators are different from that of this module. This module will be removed in 0.2
0.
  "This module will be removed in 0.20.", DeprecationWarning)
```

In [11]:

```
X = ds[['healthy_eating',
        'active_lifestyle']] # here we have 2 variables for multiple regression. If you just want t
o use one variable for simple linear regression, then use X = df['Interest_Rate'] for example.Alte
rnatively, you may add additional variables within the brackets
Y = ds['salary']
```

In [12]:

```
from sklearn import linear_model
regr = linear_model.LinearRegression()
regr.fit(X, Y)
```

Out[12]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [13]:

```
print('Intercept: \n', regr.intercept_)
print('Coefficients: \n', regr.coef_)
```

```
Intercept:
 972.852888018574
Coefficients:
 [ 466.54703352 -185.11357061]
```

In [14]:

```
New_healthy_eating = 5
New_active_lifestyle = 5
print ('Predicted salary: \n', regr.predict([[New_healthy_eating,New_active_lifestyle]]))
```

```
Predicted salary:
 [2380.0202026]
```

In [15]:

```
import statsmodels.api as sm
X = sm.add_constant(X)
```

In [16]:

```
model = sm.OLS(Y, X).fit()
```

In [17]:

```
predictions = model.predict(X)
```

In [18]:

```
print_model = model.summary()
print(print_model)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 salary   R-squared:                       0.860
Model:                            OLS   Adj. R-squared:                  0.860
Method:                 Least Squares   F-statistic:                     3062.
Date:                Tue, 30 Oct 2018   Prob (F-statistic):               0.00
Time:                        14:56:02   Log-Likelihood:                -7420.4
No. Observations:                1000   AIC:                         1.485e+04
Df Residuals:                     997   BIC:                         1.486e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                     coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const              972.8529     48.422     20.091      0.000     877.831    1067.875
healthy_eating     466.5470      6.362     73.334      0.000     454.063     479.031
active_lifestyle  -185.1136      6.252    -29.609      0.000    -197.382    -172.845
==============================================================================
Omnibus:                      759.813   Durbin-Watson:                   2.039
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            13252.958
Skew:                           3.402   Prob(JB):                         0.00
Kurtosis:                      19.486   Cond. No.                         30.0
==============================================================================
```

```
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```