In [23]:

```
%matplotlib inline

import numpy as np
import pandas as pd
import scipy.stats as stats
import matplotlib.pyplot as plt
import sklearn
import statsmodels.api as sm

import seaborn as sns
sns.set_style("whitegrid")
sns.set_context("poster")

# special matplotlib argument for improved plots
from matplotlib import rcParams
```

In [24]:

```
cd = pd.read_csv("//home//yeshua//Documents//study//excel//kid.csv")
```

In [25]:

```
print(cd.keys())
```

```
Index(['id', 'age', 'bp', 'sg', 'al', 'su', 'rbc', 'pc', 'pcc', 'ba', 'bgr',
       'bu', 'sc', 'sod', 'pot', 'hemo', 'pcv', 'wc', 'rc', 'htn', 'dm', 'cad',
       'appet', 'pe', 'ane', 'classification'],
      dtype='object')
```

In [15]:

```
df = cd.dropna()
```

In [16]:

```
cd.head()
```

Out[16]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	NaN	0.0	1.0	1.0	...	44	7800	5.2	1.0	1.0	0.0	1.0	0.0	0.0	0
1	1	7.0	50.0	1.020	4.0	0.0	NaN	0.0	1.0	1.0	...	38	6000	NaN	0.0	0.0	0.0	1.0	0.0	0.0	0
2	2	62.0	80.0	1.010	2.0	3.0	0.0	0.0	1.0	1.0	...	31	7500	NaN	0.0	1.0	0.0	0.0	0.0	1.0	0
3	3	48.0	70.0	1.005	4.0	0.0	0.0	1.0	0.0	1.0	...	32	6700	3.9	1.0	0.0	0.0	0.0	1.0	1.0	0
4	4	51.0	80.0	1.010	2.0	0.0	0.0	0.0	1.0	1.0	...	35	7300	4.6	0.0	0.0	0.0	1.0	0.0	0.0	0

5 rows × 26 columns

In [17]:

```
cd['classification'].unique()
```

Out[17]:

```
array([0, 1])
```

In [18]:

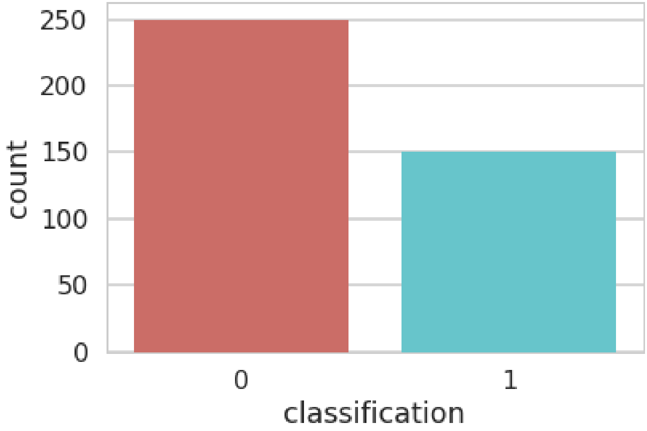
```
cd['classification'].value_counts()
```

Out[18]:

```
0    250
1    150
Name: classification, dtype: int64
```

In [19]:

```
sns.countplot(x='classification', data=cd, palette='hls')
plt.show()
```



In [20]:

```
cd.fillna(0)
pd.DataFrame(cd).fillna(0).head()
```

Out[20]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
0	0	48.0	80.0	1.020	1.0	0.0	0.0	0.0	1.0	1.0	...	44	7800	5.2	1.0	1.0	0.0	1.0	0.0	0.0	0
1	1	7.0	50.0	1.020	4.0	0.0	0.0	0.0	1.0	1.0	...	38	6000	0	0.0	0.0	0.0	1.0	0.0	0.0	0
2	2	62.0	80.0	1.010	2.0	3.0	0.0	0.0	1.0	1.0	...	31	7500	0	0.0	1.0	0.0	0.0	0.0	1.0	0
3	3	48.0	70.0	1.005	4.0	0.0	0.0	1.0	0.0	1.0	...	32	6700	3.9	1.0	0.0	0.0	0.0	1.0	1.0	0
4	4	51.0	80.0	1.010	2.0	0.0	0.0	0.0	1.0	1.0	...	35	7300	4.6	0.0	0.0	0.0	1.0	0.0	0.0	0

5 rows × 26 columns

In [32]:

```
cd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 26 columns):
id                400 non-null int64
age               400 non-null float64
bp                400 non-null int64
sg                400 non-null int64
al                400 non-null int64
su                400 non-null int64
rbc               400 non-null int64
pc                400 non-null int64
pcc               400 non-null float64
ba                400 non-null float64
bgr               400 non-null float64
bu                400 non-null int64
sc                400 non-null float64
sod               400 non-null float64
pot               400 non-null float64
hemo              400 non-null float64
```

```
pcv               400 non-null object
wc                400 non-null object
rc                400 non-null object
htn               400 non-null float64
dm                400 non-null float64
cad               400 non-null float64
appet             400 non-null float64
pe                400 non-null float64
ane               400 non-null int64
classification    400 non-null int64
dtypes: float64(13), int64(10), object(3)
memory usage: 81.3+ KB
```

In [26]:

```
cd.isnull().values.any()
```

Out[26]:

```
True
```

In [27]:

```
cd.isnull().sum().sum()
```

Out[27]:

```
1009
```

In [28]:

```
cd.dropna().head()
```

Out[28]:

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	...	pcv	wc	rc	htn	dm	cad	appet	pe	ane	classification
3	3	48.0	70.0	1.005	4.0	0.0	0.0	1.0	0.0	1.0	...	32	6700	3.9	1.0	0.0	0.0	0.0	1.0	1.0	0
9	9	53.0	90.0	1.020	2.0	0.0	1.0	1.0	0.0	1.0	...	29	12100	3.7	1.0	1.0	0.0	0.0	0.0	1.0	0
11	11	63.0	70.0	1.010	3.0	0.0	1.0	1.0	0.0	1.0	...	32	4500	3.8	1.0	1.0	0.0	0.0	1.0	0.0	0
14	14	68.0	80.0	1.010	3.0	2.0	0.0	1.0	0.0	0.0	...	16	11000	2.6	1.0	1.0	1.0	0.0	1.0	0.0	0
20	20	61.0	80.0	1.015	2.0	0.0	1.0	1.0	1.0	1.0	...	24	9200	3.2	1.0	1.0	1.0	0.0	1.0	1.0	0

5 rows × 26 columns

In [30]:

```
cd = cd.fillna(0)
```

In [31]:

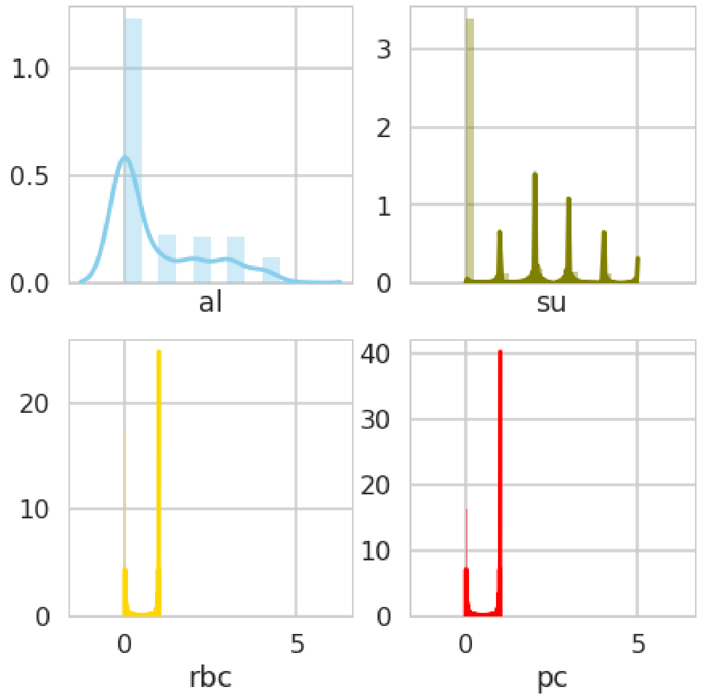
```
cd['ane'] = cd.ane.astype(int)
cd['bu'] = cd.bu.astype(int)
cd['bp'] = cd.bp.astype(int)
cd['sg'] = cd.sg.astype(int)
cd['al'] = cd.al.astype(int)
cd['su'] = cd.su.astype(int)
cd['rbc'] = cd.rbc.astype(int)
cd['pc'] = cd.pc.astype(int)
```

In [33]:

```
f, axes = plt.subplots(2, 2, figsize=(7, 7), sharex=True)
sns.distplot( cd["al"] , color="skyblue", ax=axes[0, 0])
sns.distplot( cd["su"] , color="olive", ax=axes[0, 1])
sns.distplot( cd["rbc"] , color="gold", ax=axes[1, 0])
sns.distplot( cd["pc"] , color="red", ax=axes[1, 1])
```

```
plt.show()
```

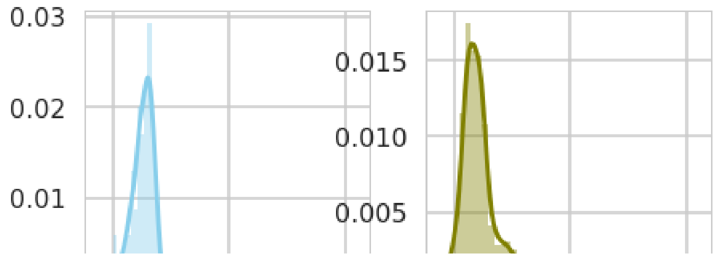
```
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: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

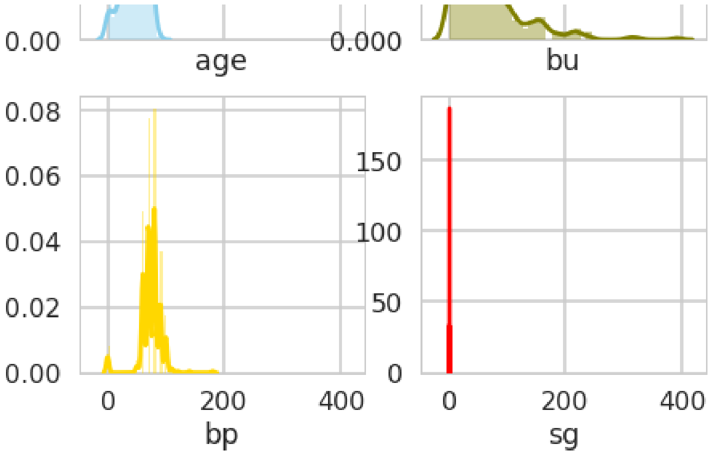


In [34]:

```
f, axes = plt.subplots(2, 2, figsize=(7, 7), sharex=True)
sns.distplot( cd["age"] , color="skyblue", ax=axes[0, 0])
sns.distplot( cd["bu"] , color="olive", ax=axes[0, 1])
sns.distplot( cd["bp"] , color="gold", ax=axes[1, 0])
sns.distplot( cd["sg"] , color="red", ax=axes[1, 1])
plt.show()
```

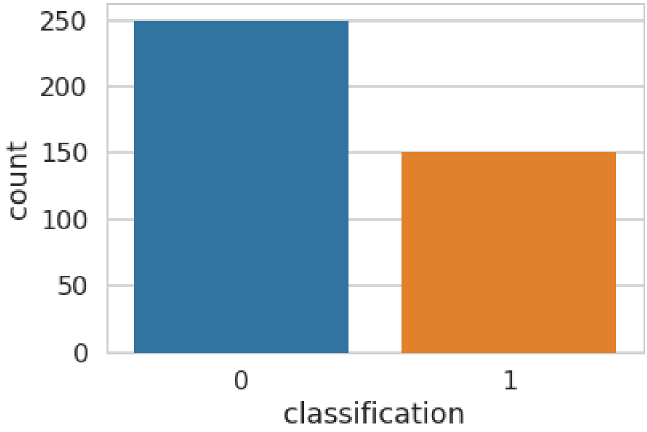
```
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: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





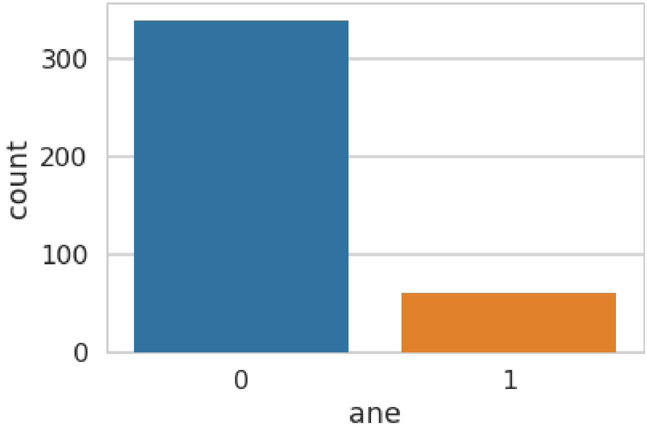
In [36]:

```
sns.countplot(x=cd["classification"])
plt.show()
```



In [38]:

```
sns.countplot(x=cd["ane"])
plt.show()
```



In [39]:

```
X = cd.drop('classification', axis = 1)
Y = cd['classification']
```

In [40]:

```
import sklearn.cross_validation
X_train, X_test, Y_train, Y_test = sklearn.cross_validation.train_test_split(X, Y, test_size = 0.33
, random_state = 5)
print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(268, 25)
(132, 25)
(268,)
(132,)
```

```
/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 [45]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.cross_validation import train_test_split
x=np.array(cd.iloc[:,11].values).reshape(-1,1)
y=np.array(cd.iloc[:,25].values).reshape(-1,1)
```

In [46]:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 0)
```

In [47]:

```
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
```

```
/home/yeshua/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:578:
DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[47]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [48]:

```
y_pred = logreg.predict(x_test)
```

In [49]:

```
from sklearn.metrics import confusion_matrix,roc_auc_score,roc_curve
confusion_matrix(y_test,y_pred)
```

Out[49]:

```
array([[57, 15],
       [30, 18]])
```

In [50]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[50]:

```
0.625
```

In [54]:

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
             precision    recall  f1-score   support

          0       0.66      0.79      0.72        72
          1       0.55      0.38      0.44        48

avg / total       0.61      0.62      0.61       120
```

In [55]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.cross_validation import train_test_split
import seaborn as sns
from sklearn.metrics import classification_report

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 0)
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
```

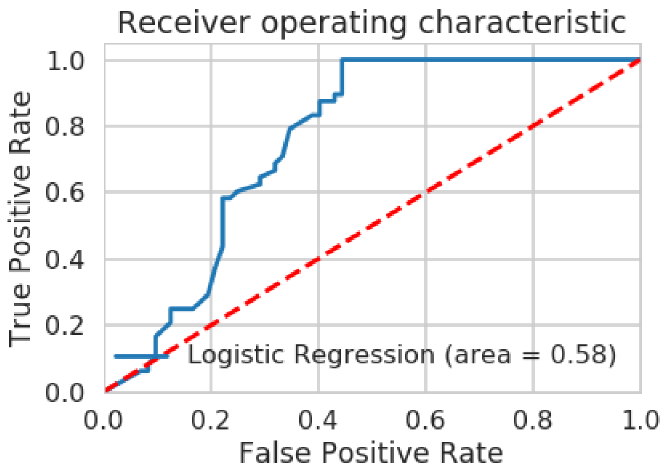
```
/home/yeshua/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:578:
DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[55]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

In [56]:

```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(y_test, logreg.predict(x_test))
fpr, tpr, thresholds = roc_curve(y_test, logreg.predict_proba(x_test)[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.savefig('Log_ROC')
plt.show()
```



In [73]:

```
x=np.array(cd.iloc[:,2:7].values).reshape(-1,1)
y=np.array(cd.iloc[:,25].values).reshape(-1,1)
```

In [75]:

```
y_pred = logreg.predict(x_test)
```

In [76]:

```
from sklearn.metrics import confusion_matrix,roc_auc_score,roc_curve
confusion_matrix(y_test,y_pred)
```

Out[76]:

```
array([[57, 15],
       [30, 18]])
```

In [77]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

Out[77]:

```
0.625
```

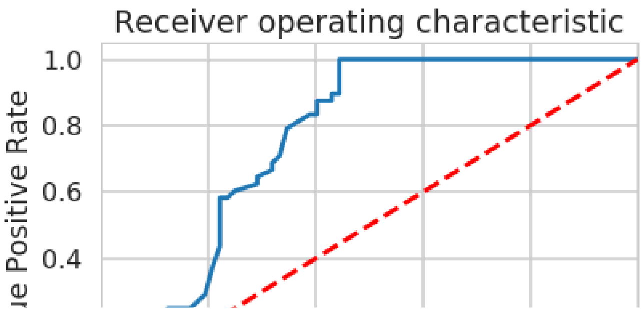
In [78]:

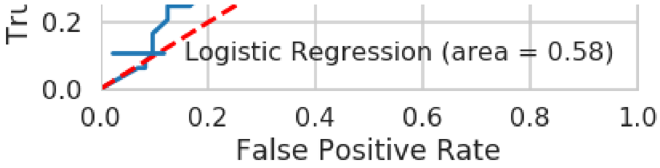
```
print(classification_report(y_test, y_pred))
```

```
             precision    recall  f1-score   support

          0       0.66      0.79      0.72        72
          1       0.55      0.38      0.44        48

avg / total       0.61      0.62      0.61       120
```

In [74]:

```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(y_test, logreg.predict(x_test))
fpr, tpr, thresholds = roc_curve(y_test, logreg.predict_proba(x_test)[:,1])
plt.figure()
plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.savefig('Log_ROC')
plt.show()
```




Tru
0.2
0.0
Logistic Regression (area = 0.58)
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate