In [30]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import rcParams
from matplotlib.cm import rainbow
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

In [31]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

In [32]:

```
df=pd.read_csv('dataset.csv')
```

In [33]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 3   trestbps  303 non-null    int64  
 4   chol      303 non-null    int64  
 5   fbs       303 non-null    int64  
 6   restecg   303 non-null    int64  
 7   thalach   303 non-null    int64  
 8   exang     303 non-null    int64  
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64  
 11  ca        303 non-null    int64  
 12  thal      303 non-null    int64  
 13  target    303 non-null    int64  
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

In [34]:

```
df.describe()
```

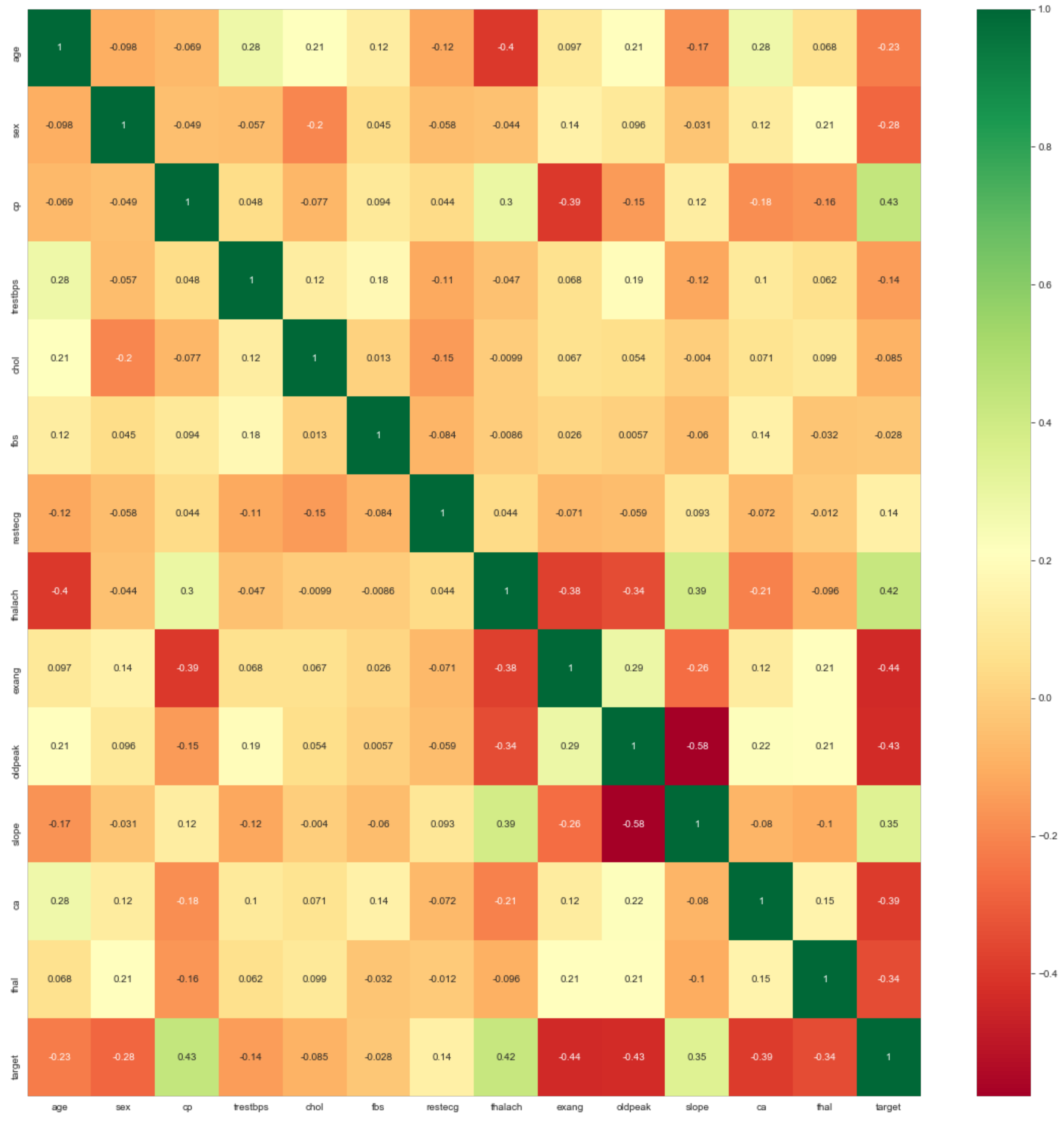
Out[34]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.3
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.6
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.0
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.0
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.0
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.0
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.0

Feature Selection

In [35]:

```
import seaborn as sns
corrmat=df.corr()
top_corr_features=corrmat.index
plt.figure(figsize=(20,20))
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```

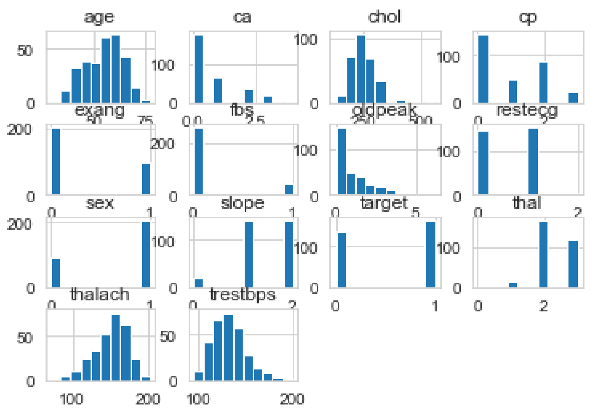


In [36]:

```
df.hist()
```

Out[36]:

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF9845FC8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF985BA48>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF98BC908>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF8FF6A48>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF902DB48>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF9063C48>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF909BD48>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF90D4E48>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF90E0A08>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF9116BC8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF9181088>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF91B9148>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF91F2288>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF922C388>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF9265488>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000018CF929C648>]],
      dtype=object)
```

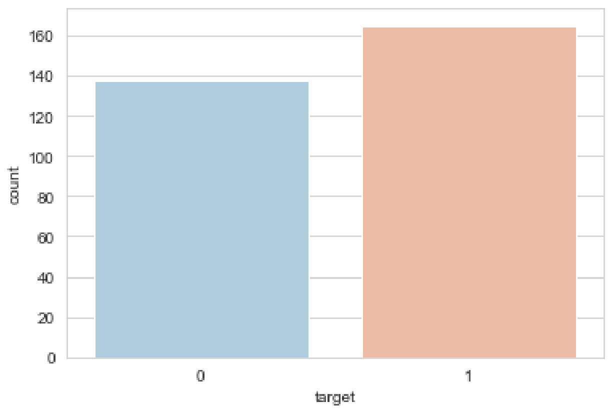


In [37]:

```
sns.set_style('whitegrid')
sns.countplot(x='target',data=df,palette='RdBu_r')
```

Out[37]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x18cf9549ac8>
```



In [38]:

```
dataset=pd.get_dummies(df,columns=['sex','cp','fbs','restecg','exang','slope','ca','thal'])
```

In [39]:

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
standardScaler=StandardScaler()
columns_to_scale=['age','trestbps','chol','thalach','oldpeak']
dataset[columns_to_scale]=standardScaler.fit_transform(dataset[columns_to_scale])
```

In [40]:

```
dataset.head()
```

Out[40]:

	age	trestbps	chol	thalach	oldpeak	target	sex_0	sex_1	cp_0	cp_1	...	slope_2	ca_0	ca_1	ca_2	ca_3	ca_4	th
0	0.952197	0.763956	-0.256334	0.015443	1.087338	1	0	1	0	0	...	0	1	0	0	0	0	
1	-1.915313	-0.092738	0.072199	1.633471	2.122573	1	0	1	0	0	...	0	1	0	0	0	0	
2	-1.474158	-0.092738	-0.816773	0.977514	0.310912	1	1	0	0	1	...	1	1	0	0	0	0	
3	0.180175	-0.663867	-0.198357	1.239897	-0.206705	1	0	1	0	1	...	1	1	0	0	0	0	
4	0.290464	-0.663867	2.082050	0.583939	-0.379244	1	1	0	1	0	...	1	1	0	0	0	0	

5 rows × 31 columns

In [41]:

```
y=dataset['target']
X=dataset.drop(['target'],axis=1)
```

In [44]:

```
from sklearn.model_selection import cross_val_score

knn_scores=[]

for k in range(1,21):
    knn_classifier=KNeighborsClassifier(n_neighbors =k)
    score=cross_val_score(knn_classifier,X,y,cv=10)
    knn_scores.append(score.mean())
```

In [50]:

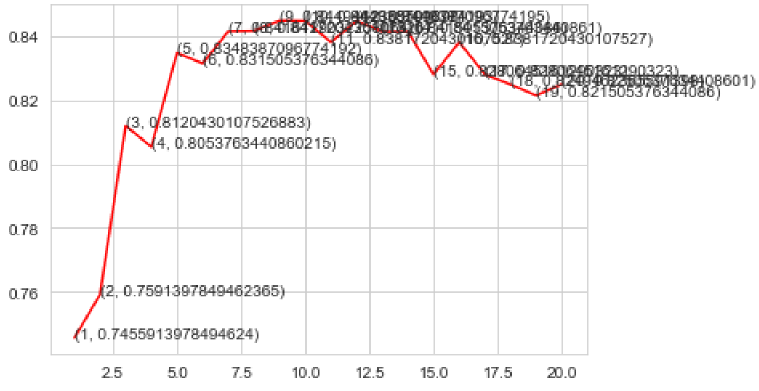
```
plt.plot([k for k in range(1,21)],knn_scores,color='red')

for i in range(1,21):
    plt.text(i,knn_scores[i-1], (i, knn_scores[i-1]))

plt.ticks([i for i in range(1,21)])
plt.Xlabel('Number of neighbors(k)')
plt.ylabel("Scores")
plt.title('K neighbors Classifier scores for different k values')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-50-ff7edf7d5a54> in <module>
      4     plt.text(i,knn_scores[i-1], (i, knn_scores[i-1]))
      5
----> 6 plt.ticks([i for i in range(1,21)])
      7 plt.Xlabel('Number of neighbors(k)')
      8 plt.ylabel("Scores")

AttributeError: module 'matplotlib.pyplot' has no attribute 'ticks'
```



In [48]:

```
knn_classifier = KNeighborsClassifier(n_neighbors = 12)
score=cross_val_score(knn_classifier, X,y,cv=10)

score.mean()
```

Out[48]:

```
0.8448387096774195
```

In [52]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [54]:

```
randomforest_classifier=RandomForestClassifier(n_estimators=10)
score=cross_val_score(randomforest_classifier,X,y,cv=10)
score.mean()
```

Out[54]:

```
0.821505376344086
```

In [ ]:

```

```