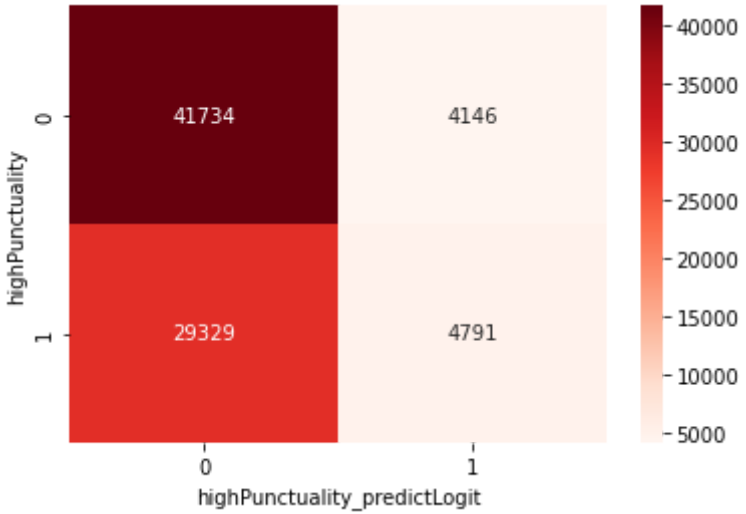
In [2]:
```
import pandas as pd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt
%matplotlib inline
import sklearn.metrics as metrics
import seaborn as sn
drRatings = pd.read_excel('./OBGYN_new_train_80000.xlsx')
# drRatings = pd.read_excel('./OBGYN_new_train_80000.xlsx',nrows=1000)
#shuffle the data so that they are in random sequence
drRatings = drRatings.sample(frac=1)
drRatings['highPunctuality'] = (drRatings['punctuality']>4).astype(int)
temp = pd.get_dummies(drRatings['state'])
drRatings = pd.concat([drRatings,temp],axis=1)
del temp
drRatings['postedTime']=pd.to_datetime(drRatings['postedTime'])
drRatings['year']=drRatings['postedTime'].dt.year
drRatings['hour']=drRatings['postedTime'].dt.hour
```

In [3]:
```
xcols = ['AK', 'AL', 'AR', 'AZ', 'CA','CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID', 'IL', 'IN',
       'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'NC', 'ND', 'NE',
       'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'PR', 'RI', 'SC', 'SD',
       'TN', 'TX', 'UT', 'VA', 'WA', 'WI', 'WV', 'WY', 'year', 'hour','numReviews']

ycol = 'highPunctuality'
x = drRatings[xcols]
x = sm.add_constant(x)
y = drRatings[ycol]

logit_model1 = sm.Logit(y, x)
logit_result = logit_model1.fit()

drRatings['highPunctuality_predictLogit'] = (logit_result.predict(x) >= 0.5).astype(int)
acc=metrics.accuracy_score(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality_predict
print('prediction accuracy is',acc)
confusion=metrics.confusion_matrix(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality
# print(confusion)
sn.heatmap(confusion, annot=True, cmap='Reds', fmt='d')
plt.xlabel("highPunctuality_predictLogit")
plt.ylabel("highPunctuality")
```

```
/Users/mac/opt/anaconda3/lib/python3.9/site-packages/statsmodels/tsa/tsatools.py:142: FutureWarning: I
n a future version of pandas all arguments of concat except for the argument 'objs' will be keyword-on
ly
  x = pd.concat(x[::order], 1)

Optimization terminated successfully.
         Current function value: 0.674984
         Iterations 5
prediction accuracy is 0.5815625
```

Out[3]:
```
Text(33.0, 0.5, 'highPunctuality')
```

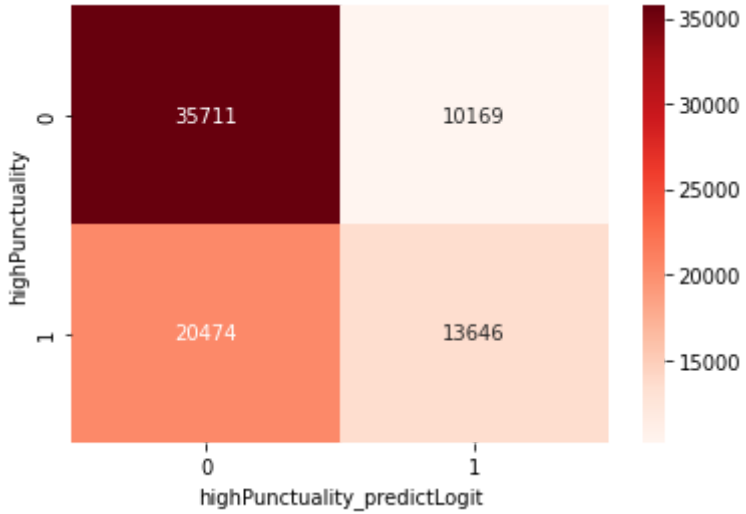


In [44]:
```
keywordlist = ['good','bad','time','wait','no']
for keyword in keywordlist:
  drRatings[keyword]=drRatings['review'].apply(lambda x: int(keyword in x))
```

In [45]:
```
xcols = ['AK', 'AL', 'AR', 'AZ', 'CA','CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID', 'IL', 'IN',
       'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'NC', 'ND', 'NE',
       'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'PR', 'RI', 'SC', 'SD',
       'TN', 'TX', 'UT', 'VA', 'WA', 'WI', 'WV', 'WY', 'year', 'hour','numReviews'] + keywordlist

ycol = 'highPunctuality'
x = drRatings[xcols]
x = sm.add_constant(x)
y = drRatings[ycol]

logit_model1 = sm.Logit(y, x)
logit_result = logit_model1.fit()

drRatings['highPunctuality_predictLogit'] = (logit_result.predict(x) >= 0.5).astype(int)
acc=metrics.accuracy_score(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality_predict
print('prediction accuracy is',acc)
confusion=metrics.confusion_matrix(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality
# print(confusion)
sn.heatmap(confusion, annot=True, cmap='Reds', fmt='d')
plt.xlabel("highPunctuality_predictLogit")
plt.ylabel("highPunctuality")
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tsa/tsatools.py:117: FutureWarning: In a future ver
sion of pandas all arguments of concat except for the argument 'objs' will be keyword-only
  x = pd.concat(x[::order], 1)

Optimization terminated successfully.
         Current function value: 0.646484
         Iterations 6
prediction accuracy is 0.6169625
```

Out[45]:
```
Text(33.0, 0.5, 'highPunctuality')
```



In [46]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import feature_extraction
stop_words = feature_extraction.text.ENGLISH_STOP_WORDS
stop_words = ["dr", "doctor"] + list(stop_words)

def calTFIDF(texts,max_features=None):
  vectorizer = TfidfVectorizer(max_features=max_features,lowercase=True,stop_words=stop_words,ngram_rang
  TFIDF = vectorizer.fit_transform(texts)
  TFIDF=pd.DataFrame(TFIDF.toarray(),columns=vectorizer.get_feature_names())
  return(TFIDF)
```

In [47]:
```
TFIDF=calTFIDF(drRatings['review'],max_features=6)
TFIDF
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function get_fe
ature_names is deprecated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please u
se get_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```

Out[47]:

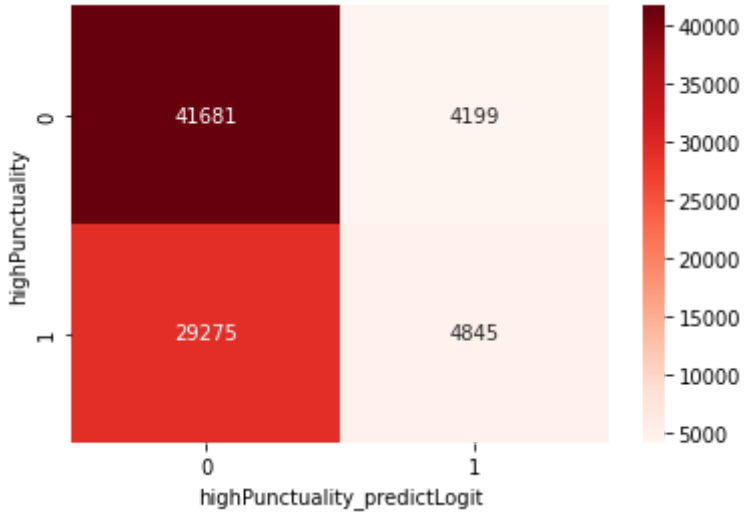
	great	like	office	recommend	staff	time
0	1.000000	0.0	0.000000	0.000000	0.000000	0.000000
1	0.000000	0.0	0.735079	0.000000	0.677981	0.000000
2	1.000000	0.0	0.000000	0.000000	0.000000	0.000000
3	0.000000	1.0	0.000000	0.000000	0.000000	0.000000
4	0.000000	0.0	0.735079	0.000000	0.677981	0.000000
...	...	...	...	...	...	...
79995	0.900705	0.0	0.000000	0.434432	0.000000	0.000000
79996	0.000000	0.0	0.000000	0.000000	0.000000	0.000000
79997	0.000000	0.0	0.000000	0.000000	0.000000	0.000000
79998	0.467924	0.0	0.000000	0.000000	0.416534	0.779453
79999	0.000000	1.0	0.000000	0.000000	0.000000	0.000000

80000 rows × 6 columns

In [48]:
```
xcols = ['AK', 'AL', 'AR', 'AZ', 'CA','CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID', 'IL', 'IN',
       'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'NC', 'ND', 'NE',
       'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'PR', 'RI', 'SC', 'SD',
       'TN', 'TX', 'UT', 'VA', 'WA', 'WI', 'WV', 'WY', 'year', 'hour','numReviews']

drRatings = pd.concat([drRatings,TFIDF],axis=1)
ycol = 'highPunctuality'
x = drRatings[xcols].values
x = sm.add_constant(x)
x = np.concatenate([x,TFIDF.values],axis=1)

y = drRatings[ycol]

logit_model1 = sm.Logit(y, x)
logit_result = logit_model1.fit()

drRatings['highPunctuality_predictLogit'] = (logit_result.predict(x) >= 0.5).astype(int)
acc=metrics.accuracy_score(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality_predict
print('prediction accuracy is',acc)
confusion=metrics.confusion_matrix(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality
# print(confusion)
sn.heatmap(confusion, annot=True, cmap='Reds', fmt='d')
plt.xlabel("highPunctuality_predictLogit")
plt.ylabel("highPunctuality")
```

```
Optimization terminated successfully.
         Current function value: 0.674943
         Iterations 5
prediction accuracy is 0.581575
```

Out[48]:
```
Text(33.0, 0.5, 'highPunctuality')
```


41681
4199
29275
4845
0
1
highPunctuality
0
1
highPunctuality_predictLogit
40000
35000
30000
25000
20000
15000
10000
5000

In [49]:
```
TFIDF=calTFIDF(drRatings['review'],max_features=250)
TFIDF
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function get_fe
ature_names is deprecated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please u
se get_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```

Out[49]:

	10	20	30	able	absolutely	actually	ago	amazing	answer	answered	...	woman	women	wonderful	work	wor
0	0.157842	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.138267	0.0	...	0.000000	0.000000	0.000000	0.0	0.0000
1	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.000000	0.000000	0.000000	0.0	0.0000
2	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.267279	0.0	...	0.000000	0.000000	0.000000	0.0	0.0000
3	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.000000	0.000000	0.233562	0.0	0.0000
4	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.000000	0.000000	0.000000	0.0	0.0000
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
79995	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.000000	0.000000	0.000000	0.0	0.0000
79996	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.000000	0.000000	0.000000	0.0	0.0000
79997	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.210687	0.208501	0.000000	0.0	0.0000
79998	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.000000	0.000000	0.000000	0.0	0.0000
79999	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	...	0.000000	0.179078	0.000000	0.0	0.1795

80000 rows × 250 columns

In [ ]:
```
import numpy as np
xcols = ['AK', 'AL', 'AR', 'AZ', 'CA','CO', 'CT', 'DC', 'DE', 'FL', 'GA', 'HI', 'IA', 'ID', 'IL', 'IN',
       'KY', 'LA', 'MA', 'MD', 'ME', 'MI', 'MN', 'MO', 'MS', 'NC', 'ND', 'NE',
       'NJ', 'NM', 'NV', 'NY', 'OH', 'OK', 'OR', 'PA', 'PR', 'RI', 'SC', 'SD',
       'TN', 'TX', 'UT', 'VA', 'WA', 'WI', 'WV', 'WY', 'year', 'hour','numReviews']

drRatings = pd.concat([drRatings,TFIDF],axis=1)
ycol = 'highPunctuality'
x = drRatings[xcols].values
x = sm.add_constant(x)
x = np.concatenate([x,TFIDF.values],axis=1)

y = drRatings[ycol]

logit_model1 = sm.Logit(y, x)
logit_result = logit_model1.fit()

drRatings['highPunctuality_predictLogit'] = (logit_result.predict(x) >= 0.5).astype(int)
acc=metrics.accuracy_score(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality_predict
print('prediction accuracy is',acc)
confusion=metrics.confusion_matrix(y_true=drRatings['highPunctuality'],y_pred=drRatings['highPunctuality
# print(confusion)
sn.heatmap(confusion, annot=True, cmap='Reds', fmt='d')
plt.xlabel("highPunctuality_predictLogit")
plt.ylabel("highPunctuality")
```