In [353]:

```
import csv
import pandas as pd
df = pd.read_csv("amazon.tsv", delimiter='\t',encoding='utf-8')
df.head()
```

Out[353]:

	marketplace	customer_id	review_id	product_id	product_parent	product_title	p
0	US	24371595	R27ZP1F1CD0C3Y	B004LLIL5A	346014806	Amazon eGift Card - Celebrate	
1	US	42489718	RJ7RSBCHUDNNE	B004LLIKVU	473048287	Amazon.com eGift Cards	
2	US	861463	R1HVYBSKLQJI5S	B00IX1I3G6	926539283	Amazon.com Gift Card Balance Reload	
3	US	25283295	R2HAXF0IIYQBIR	B00IX1I3G6	926539283	Amazon.com Gift Card Balance Reload	
4	US	397970	RNYLPX611NB7Q	B005ESMGV4	379368939	Amazon.com Gift Cards, Pack of 3 (Various Desi...	

In [355]:

```
rating = df['star_rating'].value_counts()
rating
```

Out[355]:

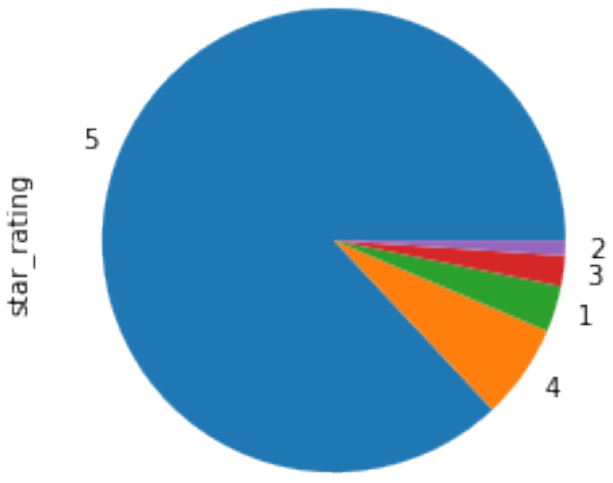
```
5    129029
4      9808
1      4766
3      3147
2      1560
Name: star_rating, dtype: int64
```

In [356]:

```
rating.plot(kind='pie')
```

Out[356]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x127c61390>
```



In [357]:

```
verified = df.groupby("verified_purchase")['star_rating'].value_counts()
verified
```

Out[357]:

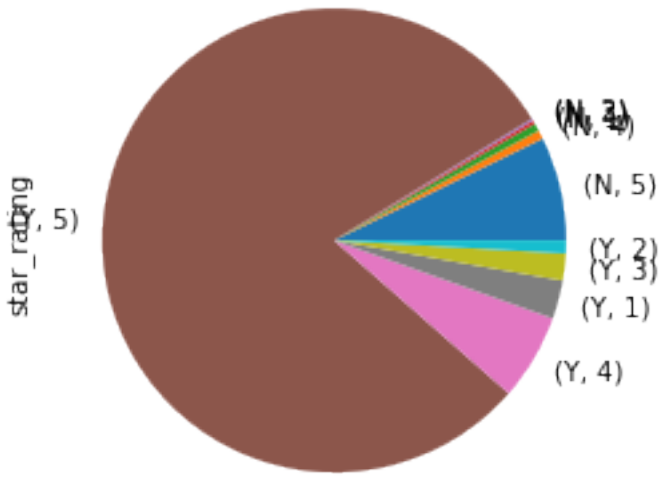
```
verified_purchase  star_rating
N                  5              10717
                   4                918
                   1                789
                   3                372
                   2                225
Y                  5             118312
                   4               8890
                   1               3977
                   3               2775
                   2               1335
Name: star_rating, dtype: int64
```

In [358]:

```
verified.plot(kind='pie')
```

Out[358]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x12891f908>
```



In [359]:

```
data=df
```

3.

In [360]:

```
import numpy           # linear algebra
import urllib          # load data from the web
import scipy.optimize # optimization routines
import random          # random number generation
import ast
```

In [371]:

```
N=df.shape[0]
```

In [375]:

```
verify = df['verified_purchase'].apply(lambda x: 1 if x=='Y' else 0)
length = df['review_body'].str.len().fillna(0)
star_rating = [int(i) for i in df['star_rating']]
def feat(d):
    return [1, verify[d], length[d]]
```

In [376]:

```
X = [feat(d) for d in range(N)]
Y=[star_rating[d] for d in range(N)]
```

In [377]:

```
theta, residuals, rank, s=numpy.linalg.lstsq(X, Y)
```

```
//anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: Fut
ureWarning: `rcond` parameter will change to the default of machine
precision times ``max(M, N)`` where M and N are the input matrix dim
ensions.
To use the future default and silence this warning we advise to pass
`rcond=None`, to keep using the old, explicitly pass `rcond=-1`.
  """Entry point for launching an IPython kernel.
```

In [378]:

```
theta
```

Out[378]:

```
array([ 4.84503503e+00,  4.98580265e-02, -1.24545522e-03])
```

θ0=4.845 : θ0 represent that if review is not verifies and the length of review is 0, the star rating we predict is about 4.844

θ1=0.04985 : θ1 represents that if the length of review remains the same and the review is verified, the star rating we predict will increase by 0.054 compared with the situation that review is not verified.

θ2=-0.001245 : θ2 represents that if the status of reviews remain the same, the star rating we predict will decrease 0.0012 when the length of review increase by 1

4.

In [380]:

```
def feat(d):
    return [1, verify[d]]
```

In [381]:

```
X = [feat(d) for d in range(N)]
Y=[star_rating[d] for d in range(N)]
```

In [382]:

```
theta_two, residuals, rank, s=numpy.linalg.lstsq(X, Y)
```

```
//anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: Fut
ureWarning: `rcond` parameter will change to the default of machine
precision times ``max(M, N)`` where M and N are the input matrix dim
ensions.
To use the future default and silence this warning we advise to pass
`rcond=None`, to keep using the old, explicitly pass `rcond=-1`.
  """Entry point for launching an IPython kernel.
```

In [383]:

```
theta_two
```

Out[383]:

```
array([4.578143  , 0.16793392])
```

θ0=4.5781 θ1=0.1679 These two coefficients vary so significantly because the number of parameters changes from 2 to 1. This linear function set only variable on whether the review is verified, while the previous function put emphasis on both verieid review and length of review. Therefore, the weight of parameter(s) changes.

5.

In [384]:

```
interN = round(len(X)*0.9)
X_train = X[:interN]
Y_train=Y[:interN]
X_test = X[interN:]
Y_test=Y[interN:]
```

In [386]:

```
theta, residuals, rank, s=numpy.linalg.lstsq(X_train, Y_train)
```

```
//anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:1: Fut
ureWarning: `rcond` parameter will change to the default of machine
precision times ``max(M, N)`` where M and N are the input matrix dim
ensions.
To use the future default and silence this warning we advise to pass
`rcond=None`, to keep using the old, explicitly pass `rcond=-1`.
  """Entry point for launching an IPython kernel.
```

In [389]:

```
X_matrix = np.matrix(X_test)
theta_matrix = np.matrix(theta)
prediction = X_matrix*theta_matrix.T
```

MSE on training data

In [391]:

```
from sklearn.metrics import mean_squared_error
MSE_train = mean_squared_error(Y_test, prediction)
print(MSE_train)
```

```
0.9723851990303892
```

MSE on training data

In [397]:

```
residuals/len(X_train)
```

Out[397]:

```
array([0.65548422])
```

6.

MAE on test dataset

In [398]:

```
from sklearn.metrics import mean_absolute_error
```

In [399]:

```
MAE_test = mean_absolute_error(Y_test, prediction)
print(MAE_test)
```

```
0.6221007247297486
```

R2 coefficient

In [400]:

```
from sklearn.metrics import r2_score
r2_score(Y_test, prediction)
```

Out[400]:

```
-0.048115873593577 83
```

8.

In [411]:

```
import csv
import pandas as pd
df = pd.read_csv("amazon.tsv", delimiter='\t',encoding='utf-8')
df.head()
dataset = []
# Read the header:
header = f[0].strip().split('\t')
for line in f[1:]:
    # Separate by tabs
    line = line.split('\t')
    # Convert to key-value pairs
    d = dict(zip(header, line))
    # Convert strings to integers for some fields:
    d['star_rating'] = int(d['star_rating'])
    d['helpful_votes'] = int(d['helpful_votes'])
    d['total_votes'] = int(d['total_votes'])
    dataset.append(d)
```

In [412]:

```
def feat(data):
    feat = [1, data['star_rating'], len('review_body')]
    return feat
```

In [413]:

```
X_train = [feat(d) for d in train]
Y_train = [1 if d['verified_purchase']=='Y' else 0 for d in train]
X_test = [feat(d) for d in test]
Y_test = [1 if d['verified_purchase']=='Y' else 0 for d in test]
```

In [414]:

```
lr = linear_model.LogisticRegression().fit(X_train, Y_train)
```

```
//anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logisti
c.py:432: FutureWarning: Default solver will be changed to 'lbfgs' i
n 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

In [415]:

```
prediction_test = lr.predict(X_test)
prediction_test
```

Out[415]:

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

accuracy of this predictor

In [416]:

```
lr.score(X_test, Y_test)
```

Out[416]:

```
0.5586558454624724
```

proportion of labels that are positive (i.e., the proportion of reviews that are verified) and the proportion of predictions that are positive

In [417]:

```
np.sum(Y_test)/len(Y_test)
```

Out[417]:

```
0.5586558454624724
```

In [418]:

```
np.sum(prediction_test)/len(prediction_test)
```

Out[418]:

```
1.0
```

9.

In [419]:

```
{'marketplace': 'US',
 'customer_id': '52088463',
 'review_id': 'R32AZRLQA1MH3E',
 'product_id': 'B004LLIKVU',
 'product_parent': '473048287',
 'product_title': 'Amazon.com eGift Cards',
 'product_category': 'Gift Card',
 'star_rating': 5,
 'helpful_votes': 0,
 'total_votes': 0,
 'vine': 'N',
 'verified_purchase': 'Y',
 'review_headline': 'easy, last-minute gift',
 'review_body': 'Fast, easy, last-minute gift!',
 'review_date': '2015-08-31\n'}
```

Out[419]:

```
{'marketplace': 'US',
 'customer_id': '52088463',
 'review_id': 'R32AZRLQA1MH3E',
 'product_id': 'B004LLIKVU',
 'product_parent': '473048287',
 'product_title': 'Amazon.com eGift Cards',
 'product_category': 'Gift Card',
 'star_rating': 5,
 'helpful_votes': 0,
 'total_votes': 0,
 'vine': 'N',
 'verified_purchase': 'Y',
 'review_headline': 'easy, last-minute gift',
 'review_body': 'Fast, easy, last-minute gift!',
 'review_date': '2015-08-31\n'}
```

using features of star_rating, helpful_votes/total_votes(helpful votes proportion)

In [420]:

```
def feat2(data):
    if data['total_votes']==0:
        return [1, data['star_rating'], 0]
    else:
        feat = [1, data['star_rating'], data['helpful_votes']/data['total_votes'
]]
        return feat
```

In [ ]:

In [421]:

```
X_train = [feat2(d) for d in train]
Y_train = [1 if d['verified_purchase']=='Y' else 0 for d in train]
X_test = [feat2(d) for d in test]
Y_test = [1 if d['verified_purchase']=='Y' else 0 for d in test]
```

In [422]:

```
lr2 = linear_model.LogisticRegression().fit(X_train, Y_train)
```

```
//anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logisti
c.py:432: FutureWarning: Default solver will be changed to 'lbfgs' i
n 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

In [423]:

```
pred=lr2.predict(X_test)
lr2.score(X_test, Y_test)
```

Out[423]:

```
0.5586558454624724
```

In [ ]:

In [ ]:

In [ ]: