In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
dating_data = pd.read_csv('d:\python programs\speed_dating.csv')
```

In [3]:

```
dating_data.head()
```

Out[3]:

	has_null	wave	gender	age	age_o	d_age	d_d_age	race	race_o	sam
0	0	1	female	21.0	27.0	6	[4-6]	asian/pacific islander/asian-american	european/caucasian-american	
1	0	1	female	21.0	22.0	1	[0-1]	asian/pacific islander/asian-american	european/caucasian-american	
2	1	1	female	21.0	22.0	1	[0-1]	asian/pacific islander/asian-american	asian/pacific islander/asian-american	
3	0	1	female	21.0	23.0	2	[2-3]	asian/pacific islander/asian-american	european/caucasian-american	
4	0	1	female	21.0	24.0	3	[2-3]	asian/pacific islander/asian-american	latino/hispanic american	

5 rows × 123 columns

In [4]:

```
dating_data.tail()
```

Out[4]:

	has_null	wave	gender	age	age_o	d_age	d_d_age	race	race_o	s
8373	1	21	male	25.0	26.0	1	[0-1]	european/caucasian-american	latino/hispanic american	
8374	1	21	male	25.0	24.0	1	[0-1]	european/caucasian-american	other	
8375	1	21	male	25.0	29.0	4	[4-6]	european/caucasian-american	latino/hispanic american	
8376	1	21	male	25.0	22.0	3	[2-3]	european/caucasian-american	asian/pacific islander/asian-american	
8377	1	21	male	25.0	22.0	3	[2-3]	european/caucasian-american	asian/pacific islander/asian-american	

5 rows × 123 columns

In [5]:

```
dating_data.shape
```

Out[5]:

```
(8378, 123)
```

In [6]:

```
dating_data.columns
```

Out[6]:

```
Index(['has_null', 'wave', 'gender', 'age', 'age_o', 'd_age', 'd_d_age',
       'race', 'race_o', 'samerace',
       ...
       'd_expected_num_interested_in_me', 'd_expected_num_matches', 'like',
       'guess_prob_liked', 'd_like', 'd_guess_prob_liked', 'met', 'decision',
       'decision_o', 'match'],
      dtype='object', length=123)
```

In [7]:

```
dating_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8378 entries, 0 to 8377
Columns: 123 entries, has_null to match
dtypes: float64(57), int64(7), object(59)
memory usage: 7.9+ MB
```

In [8]:

```
dating_data.describe()
```

Out[8]:

	has_null	wave	age	age_o	d_age	samerace	importance_
count	8378.00000	8378.000000	8283.000000	8274.000000	8378.000000	8378.000000	8
mean	0.87491	11.350919	26.358928	26.364999	4.185605	0.395799	
std	0.33084	5.995903	3.566763	3.563648	4.596171	0.489051	
min	0.00000	1.000000	18.000000	18.000000	0.000000	0.000000	
25%	1.00000	7.000000	24.000000	24.000000	1.000000	0.000000	
50%	1.00000	11.000000	26.000000	26.000000	3.000000	0.000000	
75%	1.00000	15.000000	28.000000	28.000000	5.000000	1.000000	
max	1.00000	21.000000	55.000000	55.000000	37.000000	1.000000	

8 rows × 64 columns

In [9]:

```
with open('d:\python programs\speed_dating.txt') as f:
    contents = f.read()
    print(contents)
```

```
* gender: Gender of self
* age: Age of self
* age_o: Age of partner
* d_age: Difference in age
* race: Race of self
* race_o: Race of partner
* samerace: Whether the two persons have the same race or not.
* importance_same_race: How important is it that partner is of same race?
* importance_same_religion: How important is it that partner has same religio
n?
* field: Field of study
* pref_o_attractive: How important does partner rate attractiveness
* pref_o_sinsere: How important does partner rate sincerity
* pref_o_intelligence: How important does partner rate intelligence
* pref_o_funny: How important does partner rate being funny
* pref_o_ambitious: How important does partner rate ambition
* pref_o_shared_interests: How important does partner rate having shared inter
ests
* attractive_o: Rating by partner (about me) at night of event on attractivene
ss
* sincere_o: Rating by partner (about me) at night of event on sincerity
* intelligence_o: Rating by partner (about me) at night of event on intelligen
ce
* funny_o: Rating by partner (about me) at night of event on being funny
* ambitous_o: Rating by partner (about me) at night of event on being ambitiou
s
* shared_interests_o: Rating by partner (about me) at night of event on shared
interest
* attractive_important: What do you look for in a partner - attractiveness
* sincere_important: What do you look for in a partner - sincerity
* intellicence_important: What do you look for in a partner - intelligence
* funny_important: What do you look for in a partner - being funny
* ambtition_important: What do you look for in a partner - ambition
* shared_interests_important: What do you look for in a partner - shared inter
ests
* attractive: Rate yourself - attractiveness
* sincere: Rate yourself - sincerity
* intelligence: Rate yourself - intelligence
* funny: Rate yourself - being funny
* ambition: Rate yourself - ambition
* attractive_partner: Rate your partner - attractiveness
* sincere_partner: Rate your partner - sincerity
* intelligence_partner: Rate your partner - intelligence
* funny_partner: Rate your partner - being funny
* ambition_partner: Rate your partner - ambition
* shared_interests_partner: Rate your partner - shared interests
* sports: Your own interests [1-10]
* tvsports
* exercise
* dining
* museums
* art
* hiking
* gaming
* clubbing
* reading
* tv
* theater
* movies
* concerts
* music
* shopping
```

* yoga
* interests_correlate: Correlation between participantâ€™s and partnerâ€™s ratings of interests.
* expected_happy_with_sd_people: How happy do you expect to be with the people you meet during the speed-dating event?
* expected_num_interested_in_me: Out of the 20 people you will meet, how many do you expect will be interested in dating you?
* expected_num_matches: How many matches do you expect to get?
* like: Did you like your partner?
* guess_prob_liked: How likely do you think it is that your partner likes you?
* met: Have you met your partner before?
* decision: Decision at night of event.
* decision_o: Decision of partner at night of event.
* match: Match (yes/no)

In [10]:

```
dating_data.duplicated().sum()
```

Out[10]:

```
0
```

In [11]:

```
dating_data.isnull().sum()
```

Out[11]:

```
has_null              0
wave                  0
gender                0
age                  95
age_o               104
                   ... 
d_guess_prob_liked    0
met                 375
decision              0
decision_o            0
match                 0
Length: 123, dtype: int64
```

In [12]:

```
dating_data.nunique()
```

Out[12]:

```
has_null             2
wave                21
gender               2
age                 24
age_o               24
                    ..
d_guess_prob_liked   3
met                  7
decision             2
decision_o           2
match                2
Length: 123, dtype: int64
```

In [13]:

```
dating_categorical = ['gender', 'race', 'race_o', 'field']
dating_numerical = ['has_null', 'wave', 'age', 'age_o', 'd_age', 'samerace', 'importance_sam
 'importance_same_religion', 'pref_o_attractive', 'pref_o_sincere', 'pref_o_intelligence', '
 'pref_o_ambitious', 'pref_o_shared_interests', 'attractive_o', 'sinsere_o', 'intelligence_o
 'ambitous_o', 'shared_interests_o', 'attractive_important', 'sincere_important', 'intellice
 'funny_important', 'ambtition_important', 'shared_interests_important', 'attractive', 'sinc
 'funny', 'ambition', 'attractive_partner', 'sincere_partner', 'intelligence_partner', 'funn
 'shared_interests_partner', 'sports', 'tvsports', 'exercise', 'dining', 'museums', 'art', '
 'reading', 'tv', 'theater', 'movies', 'concerts', 'music', 'shopping', 'yoga', 'interests_c
 'expected_happy_with_sd_people', 'expected_num_interested_in_me', 'expected_num_matches', '
```

In [14]:

```
dating_data[dating_categorical].nunique()
```

Out[14]:

```
gender      2
race        5
race_o      5
field     219
dtype: int64
```

In [15]:

```
dating_data[dating_categorical].value_counts()
```

Out[15]:

```
gender  race                        race_o                      field
male    european/caucasian-american european/caucasian-american business
224
female  european/caucasian-american european/caucasian-american social work
158
male    european/caucasian-american european/caucasian-american mba
135
                                                                law
97
female  european/caucasian-american european/caucasian-american law
90

...
male    european/caucasian-american black/african american      chemistry
1
female  european/caucasian-american other                       climate chan
ge       1
male    european/caucasian-american black/african american      business sch
ool      1
                                                                business [mb
a]       1
        other                       other                       theater
1
Length: 1386, dtype: int64
```

In [16]:

```
dating_data[dating_categorical].isnull().sum()
```

Out[16]:

```
gender     0
race      63
race_o    73
field     63
dtype: int64
```

In [17]:

```
dating_data[dating_numerical].nunique()
```

Out[17]:

```
has_null                          2
wave                             21
age                              24
age_o                            24
d_age                            35
                                 ..
expected_num_interested_in_me    18
expected_num_matches             17
like                             18
guess_prob_liked                 19
met                               7
Length: 61, dtype: int64
```

In [18]:

```
dating_data[dating_numerical].isnull().sum()
```

Out[18]:

```
has_null                            0
wave                                0
age                                95
age_o                             104
d_age                               0
                                 ...
expected_num_interested_in_me    6578
expected_num_matches             1173
like                              240
guess_prob_liked                  309
met                               375
Length: 61, dtype: int64
```

In [19]:

```
dating_data['field'].unique()
```

Out[19]:

```
array(['law', 'economics', 'masters in public administration',
       'masters of social work&education', 'finance', 'business',
       'political science', 'money', 'operations research',
       'tc [health ed]', 'psychology', 'social work',
       'speech language pathology', 'speech languahe pathology',
       'educational psychology', 'applied maths/econs', 'mathematics',
       'statistics', 'organizational psychology',
       'mechanical engineering', 'finanace', 'finance&economics',
       'undergrad - gs', 'mathematical finance', 'medicine', 'mba', nan,
       'german literature', 'business & international affairs',
       'mfa creative writing', 'engineering', 'electrical engineering',
       'classics', 'operations research [seas]', 'chemistry',
       'journalism', 'elementary/childhood education [ma]',
       'microbiology', 'masters of social work', 'communications',
       'marketing', 'international educational development',
       'education administration', 'business [mba]', 'computer science',
       'climate-earth and environ. science', 'financial math',
       'business- mba', 'religion', 'film', 'sociology',
       'economics; english', 'economics; sociology', 'polish', 'english',
       'psychology and english', 'biomedical engineering',
       'economics and political science', 'art history/medicine',
       'philosophy', 'marine geophysics', 'theory', 'nutrition/genetics',
       'neuroscience', 'comparative literature',
       'international relations', 'history of religion',
       'international affairs - economic development',
       'modern chinese literature', 'business; marketing',
       'physics [astrophysics]', 'physics',
       'business/ finance/ real estate', 'biochemistry', 'art education',
       'american studies [masters]', 'biology', 'cell biology', 'math',
       'international affairs/finance', 'international affairs',
       'international affairs/international finance', 'health policy',
       'english and comp lit', 'international finance and business',
       'sociomedical sciences- school of public health', 'epidemiology',
       'international business', 'medical informatics',
       'international finance; economic policy', 'law and social work',
       'international development', 'business/law', 'clinical psychology',
       'religion; gsas', 'international affairs and public health',
       'history',
       'business and international affairs [mba/mia dual degree]', 'qmss',
       'climate change', 'public administration', 'ma biotechnology',
       'international affairs/business', 'ecology',
       'master in public administration', 'computational biochemsistry',
       'neurobiology', 'mathematics; phd', 'history [gsas - phd]',
       'biomedicine', 'master of international affairs',
       'sociology and education', 'elementary education',
       'american studies', 'arts administration', 'conservation biology',
       'japanese literature', 'biotechnology',
       'earth and environmental science', 'philosophy [ph.d.]',
       'philosophy and physics', 'nutrition', 'ma science education',
       'genetics', 'law and english literature [j.d./ph.d.]', 'french',
       'nutritiron', 'gs postbacc premed', 'art history',
       'molecular biology', 'genetics & development', 'electrical engg.',
       'business school', 'international politics',
       'mba / master of international affairs [sipa]',
       'medicine and biochemistry', 'social studies education',
       'ma teaching social studies', 'education policy',
       'education- literacy specialist', 'anthropology/education',
       'bilingual education', 'speech pathology', 'education',
       'math education', 'tesol', 'cognitive studies in education',
       'finance/economics', 'museum anthropology',
       'environmental engineering', 'business administration',
       'curriculum and teaching/giftedness', 'anthropology',
```

In [20]:

```
dating_data['field'].value_counts()
```

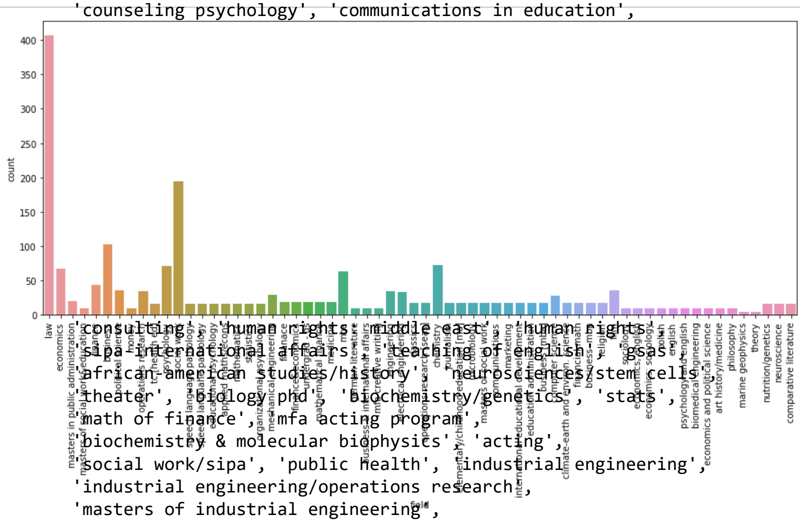
Out[20]:

```
business                          631
law                               604
mba                               468
social work                       414
international affairs             287
                                 ...
mfa  poetry                         6
fundraising management              6
business [finance & marketing]      6
marine geophysics                   5
theory                              5
Name: field, Length: 219, dtype: int64
```

```
       'instructional tech & media', 'school psychology',
       'instructional media and technology', 'sipa / mia',
       'english education', 'ma in quantitative methods',
       'early childhood education', 'architecture', 'urban planning',
       'ed.d. in higher education policy at tc',
       'international security policy - sipa',
       'applied physiology & nutrition', 'music education',
       'counseling psychology', 'communications in education',
       'consulting', 'human rights: middle east', 'human rights',
       'sipa-international affairs', 'teaching of english', 'gsas',
       'african-american studies/history', 'neurosciences/stem cells',
       'theater', 'biology phd', 'biochemistry/genetics', 'stats',
       'math of finance', 'mfa acting program',
       'biochemistry & molecular biophysics', 'acting',
       'social work/sipa', 'public health', 'industrial engineering',
       'industrial engineering/operations research',
       'masters of industrial engineering',
       'mba - private equity / real estate', 'general management/finance',
       'climate dynamics'], dtype=object)
```

In [21]:

```python
plt.figure(figsize=(15,6))
sns.countplot('field', data = dating_data.head(2000))
plt.xticks(rotation = 90)
plt.show()
```



In [22]:

```python
import string
import re
```

In [23]:

```python
dating_data['race'] = dating_data['race'].str.lower()
dating_data['race'] = dating_data['race'].str.replace("'", "", regex=False)
dating_data['race'] = dating_data['race'].str.replace(" ", "_", regex=False)
dating_data['race_o'] = dating_data['race_o'].str.lower()
dating_data['race_o'] = dating_data['race_o'].str.replace("'", "", regex=False)
dating_data['race_o'] = dating_data['race_o'].str.replace(" ", "_", regex=False)
```

In [24]:

```python
dating_data.race = dating_data.race.fillna('Not Available')
dating_data.race_o = dating_data.race_o.fillna('Not Available')
dating_data.field = dating_data.field.fillna('Not Available')
```

In [25]:

```
dating_data[dating_categorical].isnull().sum()
```

Out[25]:

```
gender    0
race      0
race_o    0
field     0
dtype: int64
```

In [26]:

```
dating_data.drop(columns=['expected_num_interested_in_me'],inplace=True)
```

In [27]:

```
dating_numerical.remove('expected_num_interested_in_me')
```

In [28]:

```
for i in dating_numerical:
    dating_data[i] = dating_data[i].fillna(dating_data[i].mean())
```

In [29]:

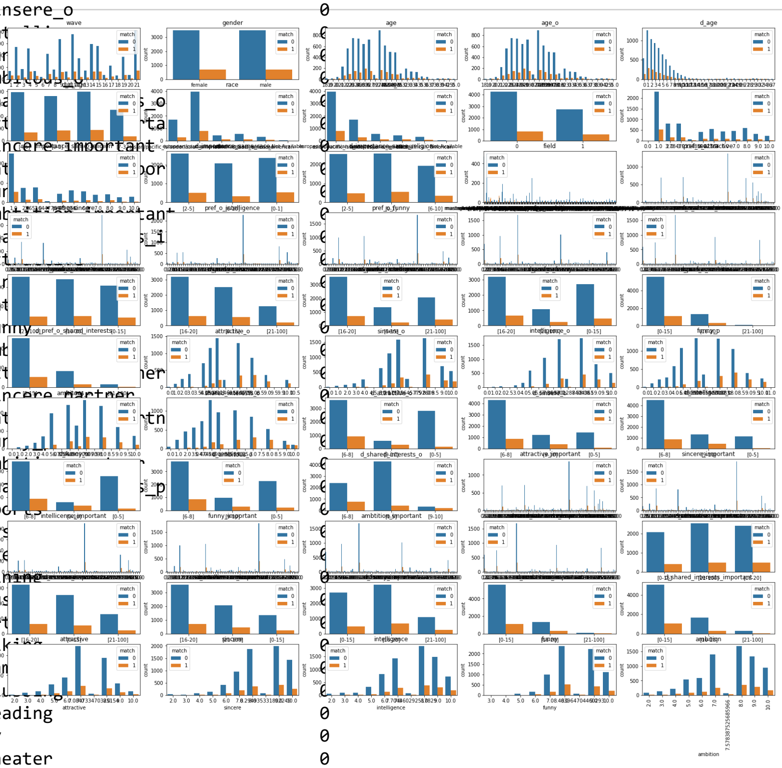
```
dating_data[dating_numerical].isnull().sum()
```

Out[29]:

```
has_null                          0
wave                              0
age                               0
age_o                             0
d_age                             0
samerace                          0
importance_same_race              0
importance_same_religion          0
pref_o_attractive                 0
pref_o_sincere                    0
pref_o_intelligence               0
pref_o_funny                      0
pref_o_ambitious                  0
pref_o_shared_interests           0
attractive_o                      0
sinsere_o                         0
```

In [30]:

```
fig, axes = plt.subplots(11,5,figsize=(28,25))
s=0
for i in range(0,11):
    for j in range(0,5):
        s+=1
        if s==123:
            break
        sns.countplot(ax = axes[i,j],x=dating_data.columns[s],
                     data=dating_data,
                     hue='match')
        plt.xticks(rotation = 90)
        axes[i,j].set_title(dating_data.columns[s])
```



```
funny                             0
sincere_partner                   0
dining                            0
reading                           0
tv                                0
theater                           0
movies                            0
concerts                          0
music                             0
shopping                          0
yoga                              0
interests_correlate               0
expected_happy_with_sd_people     0
expected_num_matches              0
like                              0
guess_prob_liked                  0
met                               0
dtype: int64
```

In [31]:

```
dating_data.match.value_counts()
```

Out[31]:

```
0    6998
1    1380
Name: match, dtype: int64
```

In [32]:

```
match = dating_data[dating_data['match']==1]
not_match = dating_data[dating_data['match']==0]
```

In [33]:

```
match.groupby('gender')['match'].count()
```

Out[33]:

```
gender
female    690
male      690
Name: match, dtype: int64
```

In [34]:

```
not_match.groupby('gender')['match'].count()
```

Out[34]:

```
gender
female    3494
male      3504
Name: match, dtype: int64
```

In [35]:

```
dating_data.corr()
```

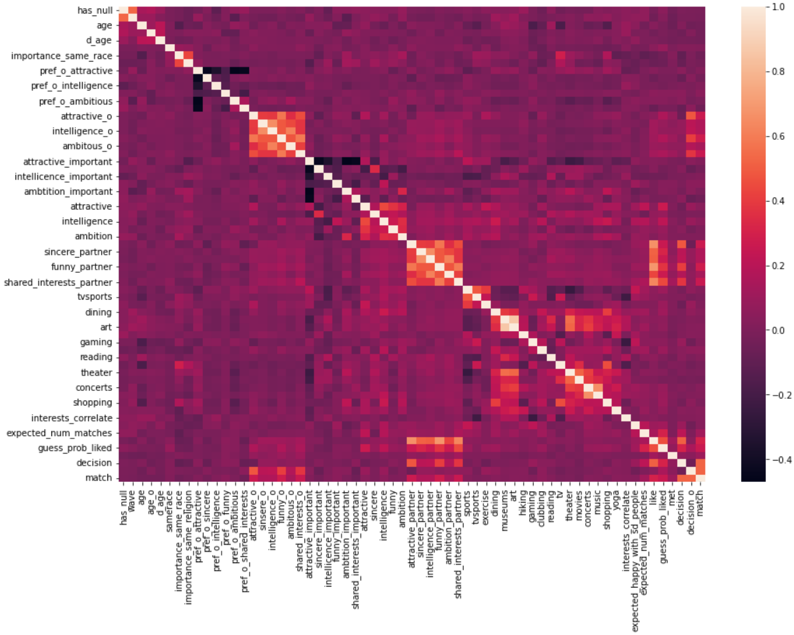
Out[35]:

	has_null	wave	age	age_o	d_age	samerace	importance_sar
has_null	1.000000	0.529313	0.144285	0.165107	0.094874	-0.016382	-0
wave	0.529313	1.000000	0.094523	0.092863	0.022024	-0.014967	-0
age	0.144285	0.094523	1.000000	0.099012	0.202476	0.007107	-0
age_o	0.165107	0.092863	0.099012	1.000000	0.208846	0.005737	-0
d_age	0.094874	0.022024	0.202476	0.208846	1.000000	-0.006238	-0
...	...	...	...	...	...	...	...
guess_prob_liked	0.041519	0.021093	-0.012547	-0.009376	-0.019391	0.082328	-0
met	-0.035000	-0.054883	-0.059553	-0.028931	-0.036715	-0.002383	0
decision	-0.002146	-0.011598	0.015801	-0.049065	-0.026940	0.023036	-0
decision_o	-0.009000	-0.010831	-0.047566	0.015043	-0.028545	0.023626	-0
match	-0.013011	-0.017404	-0.034832	-0.035632	-0.038239	0.013028	-0

63 rows × 63 columns

In [36]:

```
plt.figure(figsize=(15,10))
sns.heatmap(dating_data.corr())
plt.show()
```



In [37]:

```
from sklearn.preprocessing import StandardScaler
```

In [38]:

```
x = dating_data[dating_numerical]
y = dating_data['match']
```

In [39]:

```
x = pd.DataFrame(StandardScaler().fit_transform(x))
```

In [40]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
                                                    test_size=0.15,
                                                    random_state=42)
```

In [41]:

```
from sklearn.tree import DecisionTreeClassifier
classifier= DecisionTreeClassifier(criterion='entropy', random_state=0)
classifier.fit(x_train, y_train)
```

Out[41]:

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

In [42]:

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

In [43]:

```
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(y_test, y_pred)
```

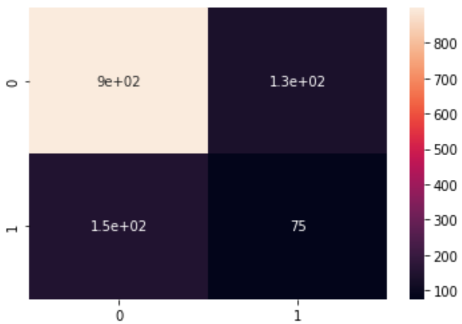
In [44]:

```
print('Confusion matrix : \n',cm)
```

```
Confusion matrix :
 [[899 134]
 [149  75]]
```

In [45]:

```
sns.heatmap(cm, annot = True)
plt.show()
```



In [46]:

```
from sklearn import metrics
from sklearn.metrics import accuracy_score
```

In [47]:

```
:("\n Classification report for classifier %s:\n%s\n" % (classifier,
                                                        metrics.classification_report(y_test
```

```
 Classification report for classifier DecisionTreeClassifier(criterion='entrop
y', random_state=0):
              precision    recall  f1-score   support

           0       0.86      0.87      0.86      1033
           1       0.36      0.33      0.35       224

    accuracy                           0.77      1257
   macro avg       0.61      0.60      0.61      1257
weighted avg       0.77      0.77      0.77      1257
```

In [48]:

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

In [49]:

```
rfc = RandomForestClassifier(n_estimators=100, random_state=42)
```

In [50]:

```
rfc.fit(x_train, y_train)
```

Out[50]:

```
▾         RandomForestClassifier
RandomForestClassifier(random_state=42)
```

In [51]:

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

In [52]:

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: {:.2f}%".format(accuracy * 100))
```

```
Accuracy: 85.20%
```

In [53]:

```
cm= confusion_matrix(y_test, y_pred)
```

In [54]:

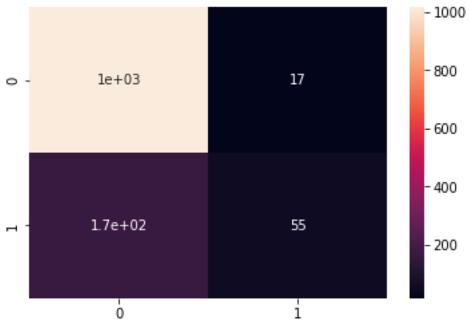
```
cm
```

Out[54]:

```
array([[1016,   17],
       [ 169,   55]], dtype=int64)
```

In [55]:

```
sns.heatmap(cm, annot = True)
plt.show()
```



In [56]:

```
print("\n Classification report for classifier %s:\n%s\n" % (rfc,
                                                               metrics.classification_report(y
```

```
 Classification report for classifier RandomForestClassifier(random_state=42):
              precision    recall  f1-score   support

           0       0.86      0.98      0.92      1033
           1       0.76      0.25      0.37       224

    accuracy                           0.85      1257
   macro avg       0.81      0.61      0.64      1257
weighted avg       0.84      0.85      0.82      1257
```