In [25]:

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

In [26]:

```
divorce_data = pd.read_csv('divorce_data.csv')
```

In [27]:

```
divorce_data.head()
```

Out[27]:

	Atr1	Atr2	Atr3	Atr4	Atr5	Atr6	Atr7	Atr8	Atr9	Atr10	...	Atr46	Atr47	Atr48	Atr49	Atr
0	2	2	4	1	0	0	0	0	0	0	...	2	1	3	3	
1	4	4	4	4	4	0	0	4	4	4	...	2	2	3	4	
2	2	2	2	2	1	3	2	1	1	2	...	3	2	3	1	
3	3	2	3	2	3	3	3	3	3	3	...	2	2	3	3	
4	2	2	1	1	1	1	0	0	0	0	...	2	1	2	3	

5 rows × 55 columns

In [28]:

```
divorce_data.tail()
```

Out[28]:

	Atr1	Atr2	Atr3	Atr4	Atr5	Atr6	Atr7	Atr8	Atr9	Atr10	...	Atr46	Atr47	Atr48	Atr49	/
165	0	0	0	0	0	0	0	0	0	0	...	1	0	4	1	
166	0	0	0	0	0	0	0	0	0	0	...	4	1	2	2	
167	1	1	0	0	0	0	0	0	0	1	...	3	0	2	0	
168	0	0	0	0	0	0	0	0	0	0	...	3	3	2	2	
169	0	0	0	0	0	0	0	1	0	0	...	3	4	4	0	

5 rows × 55 columns

In [29]:

```
divorce_data.shape
```

Out[29]:

```
(170, 55)
```

In [30]:

```
divorce_data.columns
```

Out[30]:

```
Index(['Atr1', 'Atr2', 'Atr3', 'Atr4', 'Atr5', 'Atr6', 'Atr7', 'Atr8', 'At
r9',
       'Atr10', 'Atr11', 'Atr12', 'Atr13', 'Atr14', 'Atr15', 'Atr16', 'Atr
17',
       'Atr18', 'Atr19', 'Atr20', 'Atr21', 'Atr22', 'Atr23', 'Atr24', 'Atr
25',
       'Atr26', 'Atr27', 'Atr28', 'Atr29', 'Atr30', 'Atr31', 'Atr32', 'Atr
33',
       'Atr34', 'Atr35', 'Atr36', 'Atr37', 'Atr38', 'Atr39', 'Atr40', 'Atr
41',
       'Atr42', 'Atr43', 'Atr44', 'Atr45', 'Atr46', 'Atr47', 'Atr48', 'Atr
49',
       'Atr50', 'Atr51', 'Atr52', 'Atr53', 'Atr54', 'Class'],
      dtype='object')
```

In [31]:

```
with open('divorce.txt') as f:
    contents = f.read()
    print(contents)
```

```
1. If one of us apologizes when our discussion deteriorates, the discussio
n ends.
2. I know we can ignore our differences, even if things get hard sometime
s.
3. When we need it, we can take our discussions with my spouse from the be
ginning and correct it.
4. When I discuss with my spouse, to contact him will eventually work.
5. The time I spent with my wife is special for us.
6. We don't have time at home as partners.
7. We are like two strangers who share the same environment at home rather
than family.
8. I enjoy our holidays with my wife.
9. I enjoy traveling with my wife.
10. Most of our goals are common to my spouse.
11. I think that one day in the future, when I look back, I see that my sp
ouse and I have been in harmony with each other.
12. My spouse and I have similar values in terms of personal freedom.
13. My spouse and I have similar sense of entertainment.
14. Most of our goals for people (children, friends, etc.) are the same.
15. Our dreams with my spouse are similar and harmonious.
16. We're compatible with my spouse about what love should be.
17. We share the same views about being happy in our life with my spouse
18. My spouse and I have similar ideas about how marriage should be
19. My spouse and I have similar ideas about how roles should be in marria
ge
20. My spouse and I have similar values in trust.
21. I know exactly what my wife likes.
22. I know how my spouse wants to be taken care of when she/he sick.
23. I know my spouse's favorite food.
24. I can tell you what kind of stress my spouse is facing in her/his lif
e.
25. I have knowledge of my spouse's inner world.
26. I know my spouse's basic anxieties.
27. I know what my spouse's current sources of stress are.
28. I know my spouse's hopes and wishes.
29. I know my spouse very well.
30. I know my spouse's friends and their social relationships.
31. I feel aggressive when I argue with my spouse.
32. When discussing with my spouse, I usually use expressions such as â€˜y
ou alwaysâ€™ or â€˜you neverâ€™ .
33. I can use negative statements about my spouse's personality during our
discussions.
34. I can use offensive expressions during our discussions.
35. I can insult my spouse during our discussions.
36. I can be humiliating when we discussions.
37. My discussion with my spouse is not calm.
38. I hate my spouse's way of open a subject.
39. Our discussions often occur suddenly.
40. We're just starting a discussion before I know what's going on.
41. When I talk to my spouse about something, my calm suddenly breaks.
42. When I argue with my spouse, Ä± only go out and I don't say a word.
43. I mostly stay silent to calm the environment a little bit.
44. Sometimes I think it's good for me to leave home for a while.
45. I'd rather stay silent than discuss with my spouse.
```

```
46. Even if I'm right in the discussion, I stay silent to hurt my spouse.
47. When I discuss with my spouse, I stay silent because I am afraid of no
t being able to control my anger.
48. I feel right in our discussions.
49. I have nothing to do with what I've been accused of.
50. I'm not actually the one who's guilty about what I'm accused of.
51. I'm not the one who's wrong about problems at home.
52. I wouldn't hesitate to tell my spouse about her/his inadequacy.
53. When I discuss, I remind my spouse of her/his inadequacy.
54. I'm not afraid to tell my spouse about her/his incompetence.
```

In [32]:

```
divorce_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 170 entries, 0 to 169
Data columns (total 55 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Atr1    170 non-null    int64
 1   Atr2    170 non-null    int64
 2   Atr3    170 non-null    int64
 3   Atr4    170 non-null    int64
 4   Atr5    170 non-null    int64
 5   Atr6    170 non-null    int64
 6   Atr7    170 non-null    int64
 7   Atr8    170 non-null    int64
 8   Atr9    170 non-null    int64
 9   Atr10   170 non-null    int64
 10  Atr11   170 non-null    int64
 11  Atr12   170 non-null    int64
 12  Atr13   170 non-null    int64
 13  Atr14   170 non-null    int64
 14  Atr15   170 non-null    int64
 15  Atr16   170 non-null    int64
 16  Atr17   170 non-null    int64
 17  Atr18   170 non-null    int64
 18  Atr19   170 non-null    int64
 19  Atr20   170 non-null    int64
 20  Atr21   170 non-null    int64
 21  Atr22   170 non-null    int64
 22  Atr23   170 non-null    int64
 23  Atr24   170 non-null    int64
 24  Atr25   170 non-null    int64
 25  Atr26   170 non-null    int64
 26  Atr27   170 non-null    int64
 27  Atr28   170 non-null    int64
 28  Atr29   170 non-null    int64
 29  Atr30   170 non-null    int64
 30  Atr31   170 non-null    int64
 31  Atr32   170 non-null    int64
 32  Atr33   170 non-null    int64
 33  Atr34   170 non-null    int64
 34  Atr35   170 non-null    int64
 35  Atr36   170 non-null    int64
 36  Atr37   170 non-null    int64
 37  Atr38   170 non-null    int64
 38  Atr39   170 non-null    int64
 39  Atr40   170 non-null    int64
 40  Atr41   170 non-null    int64
 41  Atr42   170 non-null    int64
 42  Atr43   170 non-null    int64
 43  Atr44   170 non-null    int64
 44  Atr45   170 non-null    int64
 45  Atr46   170 non-null    int64
 46  Atr47   170 non-null    int64
 47  Atr48   170 non-null    int64
 48  Atr49   170 non-null    int64
 49  Atr50   170 non-null    int64
 50  Atr51   170 non-null    int64
```

```
 51  Atr52   170 non-null    int64
 52  Atr53   170 non-null    int64
 53  Atr54   170 non-null    int64
 54  Class   170 non-null    int64
dtypes: int64(55)
memory usage: 73.2 KB
```

In [33]:

```
divorce_data.describe()
```

Out[33]:

	Atr1	Atr2	Atr3	Atr4	Atr5	Atr6	Atr7	
count	170.000000	170.000000	170.000000	170.000000	170.000000	170.000000	170.000000	1
mean	1.776471	1.652941	1.764706	1.482353	1.541176	0.747059	0.494118	
std	1.627257	1.468654	1.415444	1.504327	1.632169	0.904046	0.898698	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
50%	2.000000	2.000000	2.000000	1.000000	1.000000	0.000000	0.000000	
75%	3.000000	3.000000	3.000000	3.000000	3.000000	1.000000	1.000000	
max	4.000000	4.000000	4.000000	4.000000	4.000000	4.000000	4.000000	

8 rows × 55 columns

In [34]:

```
divorce_data.isnull().sum()
```

Out[34]:

```
Atr1     0
Atr2     0
Atr3     0
Atr4     0
Atr5     0
Atr6     0
Atr7     0
Atr8     0
Atr9     0
Atr10    0
Atr11    0
Atr12    0
Atr13    0
Atr14    0
Atr15    0
Atr16    0
Atr17    0
Atr18    0
Atr19    0
Atr20    0
Atr21    0
Atr22    0
Atr23    0
Atr24    0
Atr25    0
Atr26    0
Atr27    0
Atr28    0
Atr29    0
Atr30    0
Atr31    0
Atr32    0
Atr33    0
Atr34    0
Atr35    0
Atr36    0
Atr37    0
Atr38    0
Atr39    0
Atr40    0
Atr41    0
Atr42    0
Atr43    0
Atr44    0
Atr45    0
Atr46    0
Atr47    0
Atr48    0
Atr49    0
Atr50    0
Atr51    0
Atr52    0
Atr53    0
Atr54    0
```

```
Class    0
dtype: int64
```

In [35]:

```
divorce_data.nunique()
```

Out[35]:

```
Atr1     5
Atr2     5
Atr3     5
Atr4     5
Atr5     5
Atr6     5
Atr7     5
Atr8     5
Atr9     5
Atr10    5
Atr11    5
Atr12    5
Atr13    5
Atr14    5
Atr15    5
Atr16    5
Atr17    5
Atr18    5
Atr19    5
Atr20    5
Atr21    5
Atr22    5
Atr23    5
Atr24    5
Atr25    5
Atr26    5
Atr27    5
Atr28    5
Atr29    5
Atr30    5
Atr31    5
Atr32    5
Atr33    5
Atr34    5
Atr35    5
Atr36    5
Atr37    5
Atr38    5
Atr39    5
Atr40    5
Atr41    5
Atr42    5
Atr43    5
Atr44    5
Atr45    5
Atr46    5
Atr47    5
Atr48    5
Atr49    5
Atr50    5
Atr51    5
Atr52    5
Atr53    5
Atr54    5
```

```
Class    2
dtype: int64
```

In [36]:

```
divorce_data['Class'].values
```

Out[36]:

```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

In [37]:

```
divorce_data['Class'].unique()
```

Out[37]:

```
array([1, 0], dtype=int64)
```

In [38]:

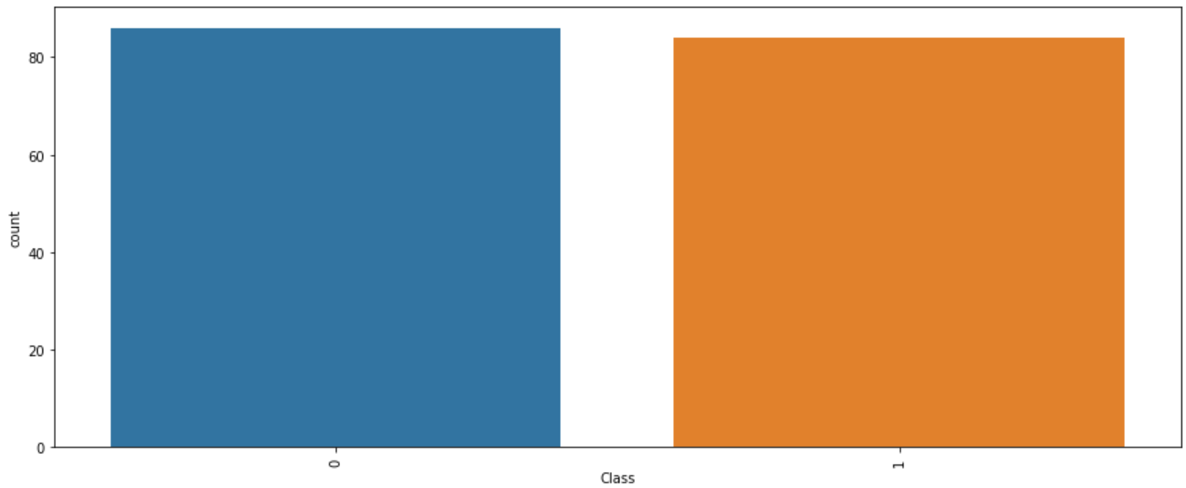
```
divorce_data['Class'].value_counts()
```

Out[38]:

```
0    86
1    84
Name: Class, dtype: int64
```

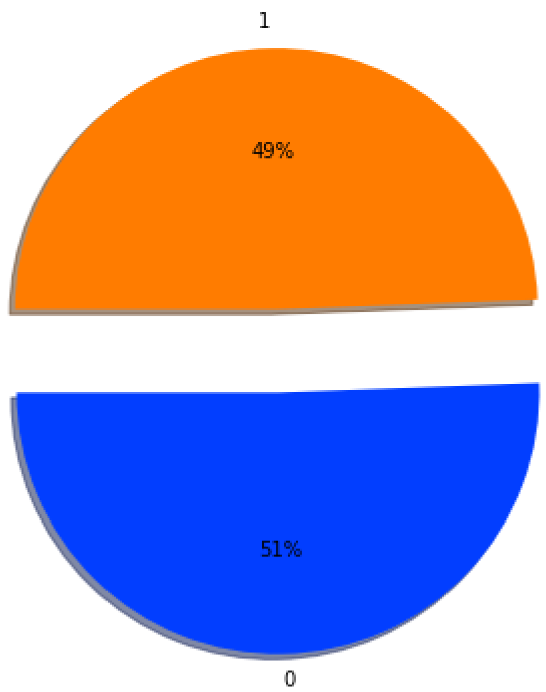
In [39]:

```
plt.figure(figsize=(15,6))
sns.countplot('Class', data = divorce_data)
plt.xticks(rotation = 90)
plt.show()
```



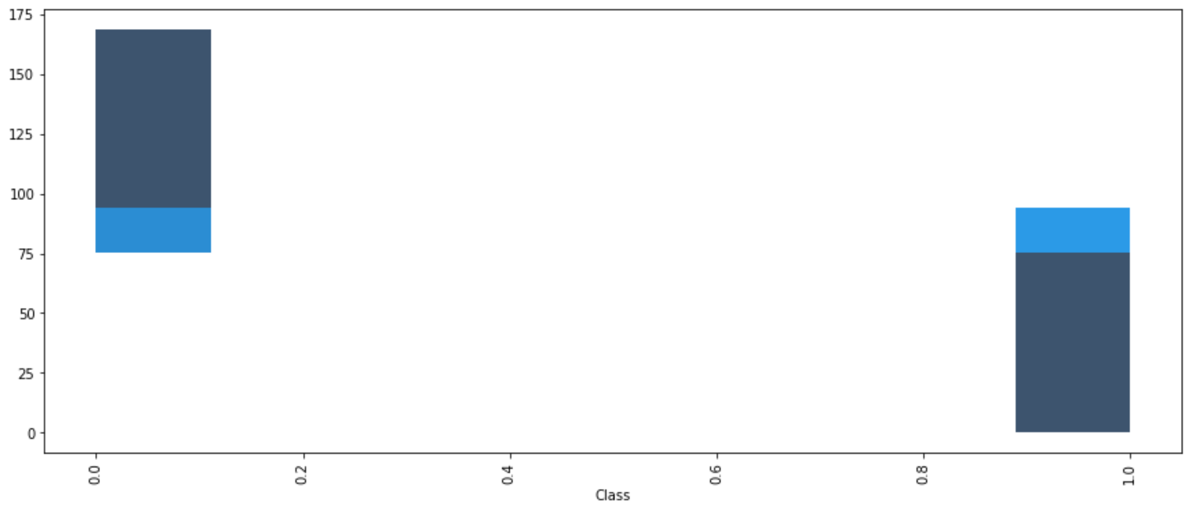
In [40]:

```
plt.figure(figsize=(15,6))
colors = sns.color_palette('bright')
explode = [0.3, 0.02]
plt.pie(divorce_data['Class'].value_counts(), colors = colors,
        labels = [0, 1], autopct = '%0.0f%%', shadow = 'True',
        explode = explode , startangle = 180)
plt.show()
```



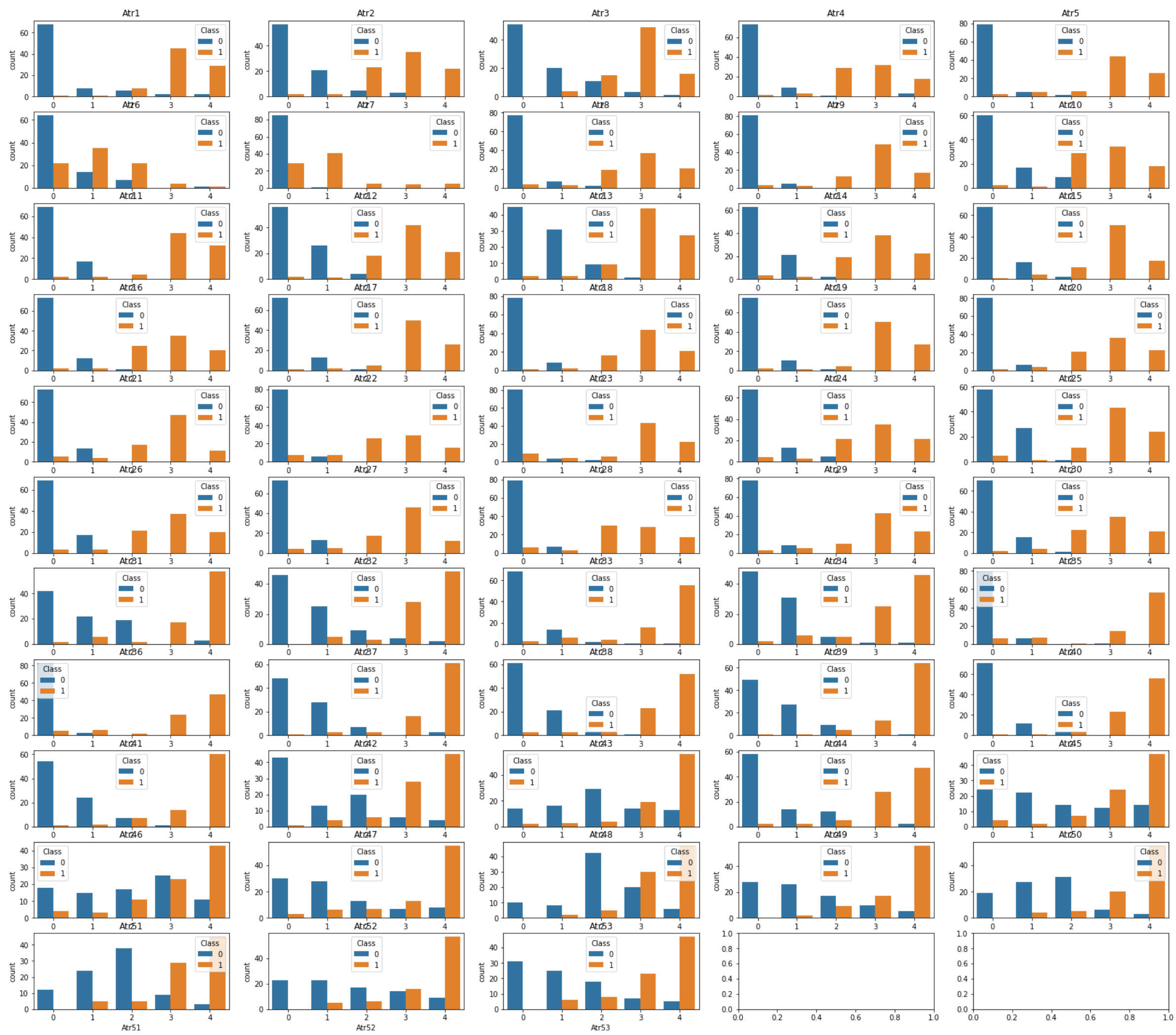
In [41]:

```
plt.figure(figsize=(15,6))
sns.histplot(x=divorce_data['Class'],y=divorce_data.index)
plt.xticks(rotation = 90)
plt.show()
```



In [43]:

```
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==54:
            break
        sns.countplot(ax = axes[i,j],x=f'Atr{s}',data=divorce_data,
                      hue='Class')
        axes[i,j].set_title(f'Atr{s}')
```



In [44]:

```
divorce_data.corr()
```

Out[44]:

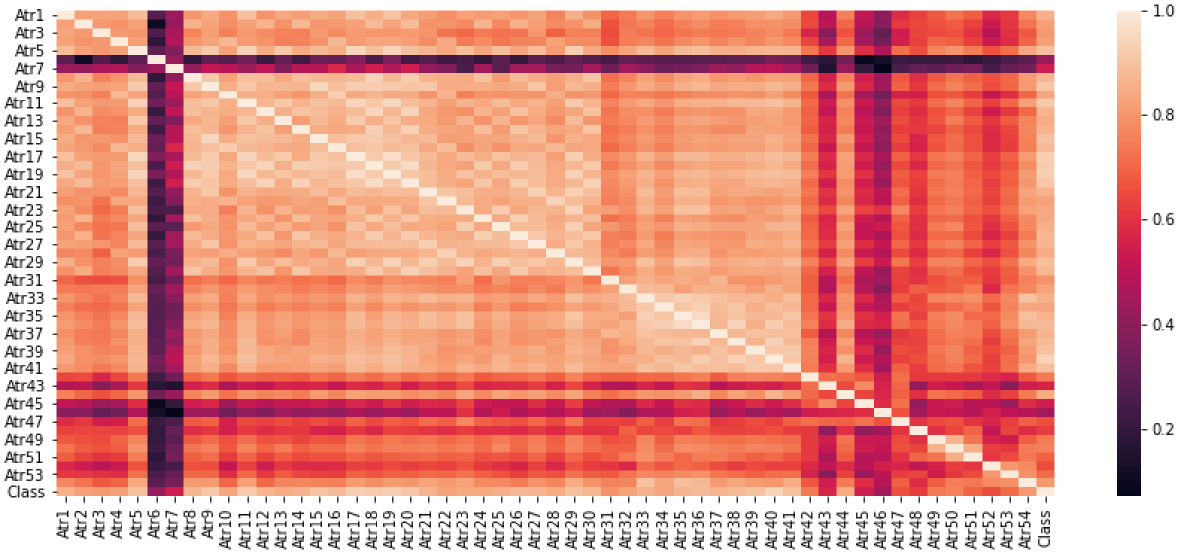
	Atr1	Atr2	Atr3	Atr4	Atr5	Atr6	Atr7	Atr8	At
Atr1	1.000000	0.819066	0.832508	0.825066	0.881272	0.287140	0.427989	0.802357	0.8459
Atr2	0.819066	1.000000	0.805876	0.791313	0.819360	0.102843	0.417616	0.864284	0.8277
Atr3	0.832508	0.805876	1.000000	0.806709	0.800774	0.263032	0.464071	0.757264	0.8166
Atr4	0.825066	0.791313	0.806709	1.000000	0.818472	0.185963	0.474806	0.798347	0.8290
Atr5	0.881272	0.819360	0.800774	0.818472	1.000000	0.297834	0.381378	0.877584	0.9163
Atr6	0.287140	0.102843	0.263032	0.185963	0.297834	1.000000	0.424212	0.184019	0.3013
Atr7	0.427989	0.417616	0.464071	0.474806	0.381378	0.424212	1.000000	0.412807	0.5175
Atr8	0.802357	0.864284	0.757264	0.798347	0.877584	0.184019	0.412807	1.000000	0.9153
Atr9	0.845916	0.827711	0.816653	0.829053	0.916327	0.301342	0.517522	0.915301	1.0000
Atr10	0.790183	0.782286	0.753017	0.873636	0.823659	0.266076	0.498266	0.828031	0.8523
Atr11	0.892253	0.823380	0.805915	0.808533	0.936955	0.340135	0.432479	0.889795	0.9115
Atr12	0.794307	0.862835	0.780258	0.793992	0.846513	0.209801	0.511761	0.890338	0.8690
Atr13	0.842996	0.791073	0.758969	0.751623	0.915033	0.305109	0.373361	0.840350	0.8730
Atr14	0.817099	0.875800	0.750602	0.757000	0.845576	0.224459	0.491021	0.888822	0.8681
Atr15	0.848754	0.801316	0.806909	0.794184	0.879461	0.323787	0.494110	0.873804	0.9490
Atr16	0.831822	0.806497	0.775528	0.878416	0.853561	0.311056	0.573290	0.865680	0.8933
Atr17	0.895970	0.822317	0.808161	0.809968	0.947429	0.377330	0.461450	0.881005	0.9223
Atr18	0.853739	0.883856	0.797395	0.835296	0.894474	0.251856	0.544550	0.941084	0.9255
Atr19	0.900446	0.829422	0.798999	0.832750	0.943349	0.365227	0.469995	0.873546	0.9164
Atr20	0.840966	0.884176	0.807892	0.815896	0.892909	0.230486	0.544207	0.922465	0.9022
Atr21	0.815708	0.790468	0.796069	0.775132	0.871994	0.273564	0.409827	0.861939	0.9094
Atr22	0.785280	0.795406	0.727933	0.839534	0.840265	0.220010	0.378915	0.857010	0.8499
Atr23	0.822534	0.773018	0.706585	0.744783	0.888584	0.246478	0.254912	0.845731	0.8502
Atr24	0.813233	0.868240	0.740476	0.776640	0.833608	0.191458	0.446469	0.896841	0.8510
Atr25	0.822084	0.769244	0.724506	0.736228	0.888740	0.291159	0.288867	0.809110	0.8387
Atr26	0.803507	0.861421	0.728653	0.762765	0.836194	0.200634	0.443149	0.883414	0.8502
Atr27	0.829037	0.817364	0.797595	0.767206	0.883768	0.283895	0.444643	0.848766	0.9039
Atr28	0.762102	0.776943	0.689914	0.827847	0.809789	0.254858	0.351262	0.822361	0.8180
Atr29	0.858139	0.789827	0.755491	0.781792	0.925601	0.309302	0.349379	0.860194	0.8788
Atr30	0.792257	0.844007	0.752391	0.772562	0.837501	0.266464	0.448569	0.902820	0.8544
Atr31	0.699223	0.661210	0.652188	0.661251	0.785038	0.247634	0.334308	0.716731	0.7456
Atr32	0.739679	0.735763	0.747669	0.746677	0.832032	0.316605	0.442306	0.762425	0.8033

	Atr1	Atr2	Atr3	Atr4	Atr5	Atr6	Atr7	Atr8	At
Atr33	0.799735	0.757286	0.726481	0.764381	0.879037	0.292037	0.395764	0.818682	0.8449
Atr34	0.749774	0.714360	0.702500	0.729022	0.827560	0.279789	0.328700	0.780778	0.8101
Atr35	0.796413	0.753566	0.730290	0.770813	0.878289	0.276539	0.349076	0.827441	0.8549
Atr36	0.812867	0.781295	0.744390	0.794636	0.887498	0.287708	0.370158	0.845435	0.8716
Atr37	0.786890	0.747088	0.736984	0.760451	0.859581	0.281458	0.431979	0.800964	0.8390
Atr38	0.804129	0.751705	0.740642	0.790350	0.852601	0.297791	0.401769	0.815830	0.8494
Atr39	0.817035	0.787768	0.759820	0.763502	0.866293	0.296121	0.477063	0.797134	0.8506
Atr40	0.838355	0.788200	0.781657	0.798520	0.871809	0.351433	0.501758	0.822302	0.8756
Atr41	0.804182	0.780757	0.739967	0.768706	0.864434	0.329765	0.445483	0.821081	0.8524
Atr42	0.642307	0.648539	0.569293	0.639671	0.737922	0.227993	0.333211	0.699571	0.7374
Atr43	0.482223	0.503894	0.385152	0.452479	0.613142	0.171599	0.149930	0.555187	0.5856
Atr44	0.752972	0.699765	0.661830	0.707212	0.799453	0.339918	0.425874	0.760016	0.8086
Atr45	0.510160	0.489062	0.427409	0.446798	0.591656	0.094820	0.199548	0.542547	0.5753
Atr46	0.400296	0.389519	0.308149	0.340240	0.470758	0.127759	0.069850	0.433541	0.4343
Atr47	0.582693	0.616884	0.544863	0.552301	0.719899	0.212979	0.254225	0.675584	0.6938
Atr48	0.633564	0.643762	0.638256	0.630205	0.659220	0.200673	0.311110	0.588531	0.6117
Atr49	0.674843	0.659841	0.647961	0.699069	0.762257	0.201091	0.291325	0.674776	0.7115
Atr50	0.725443	0.680538	0.663995	0.685263	0.795960	0.221100	0.332370	0.729668	0.7555
Atr51	0.684143	0.636558	0.600603	0.624015	0.742664	0.179119	0.349920	0.690190	0.7137
Atr52	0.575463	0.536294	0.491803	0.534264	0.663855	0.205056	0.243104	0.658613	0.6523
Atr53	0.611422	0.610726	0.598749	0.588390	0.719493	0.258092	0.313725	0.705071	0.6992
Atr54	0.768522	0.728897	0.673012	0.698264	0.836799	0.292428	0.347493	0.807911	0.8109
Class	0.861324	0.820774	0.806709	0.819583	0.893180	0.420913	0.544835	0.869569	0.9123

55 rows × 55 columns

In [47]:

```
plt.figure(figsize=(15,6))
sns.heatmap(divorce_data.corr())
plt.show()
```



In [48]:

```
x = divorce_data.drop('Class',axis =1)
y = divorce_data['Class']
```

In [49]:

```
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 [50]:

```
# importing module
from sklearn.linear_model import LinearRegression
# creating an object of LinearRegression class
LR = LinearRegression()
# fitting the training data
LR.fit(x_train,y_train)
```

Out[50]:

```
LinearRegression()
```

In [51]:

```
y_prediction =  LR.predict(x_test)
y_prediction
```

Out[51]:

```
array([-2.98242255e-01,  1.04218550e+00, -1.91789945e-02,  1.03862867e+00,
        7.03564733e-02, -8.35928052e-04,  2.07739887e-02,  1.00384295e+00,
        9.70405323e-02,  9.92217945e-01,  1.06361244e+00,  1.53710970e-01,
        8.89485246e-02,  1.01424057e+00,  9.14619813e-01, -1.58940064e-01,
        1.04075060e+00, -4.42718464e-02,  9.76082286e-01,  2.16106731e-01,
        1.00445617e+00,  1.06361244e+00,  9.79472390e-01,  1.06082144e+00,
        1.06929121e+00,  3.62517357e-01])
```

In [53]:

```
print("Training Accuracy :", LR.score(x_train, y_train))
print("Testing Accuracy :", LR.score(x_test, y_test))
```

```
Training Accuracy : 0.9714138394151279
Testing Accuracy : 0.8981950750068269
```

In [57]:

```
from tensorflow.keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import cross_val_score
from tensorflow.keras.models import Sequential # initialize neural network library
from tensorflow.keras.layers import Dense # build our layers library
```

In [58]:

```
def build_classifier():
    classifier = Sequential() # initialize neural network
    classifier.add(Dense(units = 8, kernel_initializer = 'uniform', activation = 'relu',
    classifier.add(Dense(units = 4, kernel_initializer = 'uniform', activation = 'relu')
    classifier.add(Dense(units = 1, kernel_initializer = 'uniform', activation = 'sigmoi
    classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['acc
    return classifier
```

In [59]:

```
classifier = KerasClassifier(build_fn = build_classifier, epochs = 50)
accuracies = cross_val_score(estimator = classifier, X = x_train, y = y_train, cv = 2)
mean = accuracies.mean()
variance = accuracies.std()
```

```
Epoch 1/50
3/3 [==============================] - 0s 33ms/step - loss: 0.6931 - acc
uracy: 0.5417
Epoch 2/50
3/3 [==============================] - 0s 4ms/step - loss: 0.6929 - accu
racy: 0.5417
Epoch 3/50
3/3 [==============================] - 0s 2ms/step - loss: 0.6928 - accu
racy: 0.5417
Epoch 4/50
3/3 [==============================] - 0s 4ms/step - loss: 0.6926 - accu
racy: 0.5417
Epoch 5/50
3/3 [==============================] - 0s 4ms/step - loss: 0.6923 - accu
racy: 0.5417
Epoch 6/50
3/3 [==============================] - 0s 4ms/step - loss: 0.6919 - accu
racy: 0.5417
Epoch 7/50
```

In [61]:

```
print("Accuracy mean: "+ str(mean))
```

```
Accuracy mean: 0.9722222089767456
```

In [62]:

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

In [68]:

```
clf = RandomForestClassifier()
```

In [69]:

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

Out[69]:

```
RandomForestClassifier()
```

In [70]:

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

In [71]:

```
print("Training Accuracy :", clf.score(x_train, y_train))
print("Testing Accuracy :", clf.score(x_test, y_test))
```

```
Training Accuracy : 1.0
Testing Accuracy : 0.9615384615384616
```

In [72]:

```
from sklearn import metrics
print()

# using metrics module for accuracy calculation
print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))
```

```

ACCURACY OF THE MODEL:  0.9615384615384616
```