SVM_Classification_Breast_Cancer

December 2, 2018

```
In [2]: import pandas as pd
        import numpy as np

In [3]: import matplotlib.pyplot as plt
        import seaborn as sns

In [4]: %matplotlib inline

In [5]: from sklearn.datasets import load_breast_cancer

In [9]: #loading breast cancer
        cancer = load_breast_cancer()

In [12]: print (cancer['DESCR'])

Breast Cancer Wisconsin (Diagnostic) Database
=============================================

Notes
-----
Data Set Characteristics:
    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and the class

    :Attribute Information:
        - radius (mean of distances from center to points on the perimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
        - area
        - smoothness (local variation in radius lengths)
        - compactness (perimeter^2 / area - 1.0)
        - concavity (severity of concave portions of the contour)
        - concave points (number of concave portions of the contour)
        - symmetry
        - fractal dimension ("coastline approximation" - 1)

        The mean, standard error, and "worst" or largest (mean of the three
```

largest values) of these features were computed for each image,
resulting in 30 features. For instance, field 3 is Mean Radius, field
13 is Radius SE, field 23 is Worst Radius.

- class:
        - WDBC-Malignant
        - WDBC-Benign

:Summary Statistics:

	Min	Max
radius (mean):	6.981	28.11
texture (mean):	9.71	39.28
perimeter (mean):	43.79	188.5
area (mean):	143.5	2501.0
smoothness (mean):	0.053	0.163
compactness (mean):	0.019	0.345
concavity (mean):	0.0	0.427
concave points (mean):	0.0	0.201
symmetry (mean):	0.106	0.304
fractal dimension (mean):	0.05	0.097
radius (standard error):	0.112	2.873
texture (standard error):	0.36	4.885
perimeter (standard error):	0.757	21.98
area (standard error):	6.802	542.2
smoothness (standard error):	0.002	0.031
compactness (standard error):	0.002	0.135
concavity (standard error):	0.0	0.396
concave points (standard error):	0.0	0.053
symmetry (standard error):	0.008	0.079
fractal dimension (standard error):	0.001	0.03
radius (worst):	7.93	36.04
texture (worst):	12.02	49.54
perimeter (worst):	50.41	251.2
area (worst):	185.2	4254.0
smoothness (worst):	0.071	0.223
compactness (worst):	0.027	1.058
concavity (worst):	0.0	1.252
concave points (worst):	0.0	0.291
symmetry (worst):	0.156	0.664
fractal dimension (worst):	0.055	0.208

:Missing Attribute Values: None

:Class Distribution: 212 - Malignant, 357 - Benign

:Creator: Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian

:Donor: Nick Street

:Date: November, 1995

This is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.
https://goo.gl/U2Uwz2

Features are computed from a digitized image of a fine needle
aspirate (FNA) of a breast mass. They describe
characteristics of the cell nuclei present in the image.

Separating plane described above was obtained using
Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree
Construction Via Linear Programming." Proceedings of the 4th
Midwest Artificial Intelligence and Cognitive Science Society,
pp. 97-101, 1992], a classification method which uses linear
programming to construct a decision tree. Relevant features
were selected using an exhaustive search in the space of 1-4
features and 1-3 separating planes.

The actual linear program used to obtain the separating plane
in the 3-dimensional space is that described in:
[K. P. Bennett and O. L. Mangasarian: "Robust Linear
Programming Discrimination of Two Linearly Inseparable Sets",
Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server:

ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

References

- W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology, volume 1905, pages 861-870, San Jose, CA, 1993.
- O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. Operations Research, 43(4), pages 570-577, July-August 1995.
- W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994) 163-171.

```
In [129]: # set up dataframe

In [49]: cancer.keys()

Out[49]: ['target_names', 'data', 'target', 'DESCR', 'feature_names']

In [40]: df_features = pd.DataFrame(cancer['data'], columns=cancer['feature_names'])

In [50]: #data frame info
         df_features.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 30 columns):
mean radius                569 non-null float64
mean texture               569 non-null float64
mean perimeter             569 non-null float64
mean area                  569 non-null float64
mean smoothness            569 non-null float64
mean compactness           569 non-null float64
mean concavity             569 non-null float64
mean concave points        569 non-null float64
mean symmetry              569 non-null float64
mean fractal dimension     569 non-null float64
radius error               569 non-null float64
texture error              569 non-null float64
perimeter error            569 non-null float64
area error                 569 non-null float64
smoothness error           569 non-null float64
compactness error          569 non-null float64
concavity error            569 non-null float64
concave points error       569 non-null float64
symmetry error             569 non-null float64
fractal dimension error    569 non-null float64
worst radius               569 non-null float64
worst texture              569 non-null float64
worst perimeter            569 non-null float64
worst area                 569 non-null float64
worst smoothness           569 non-null float64
worst compactness          569 non-null float64
worst concavity            569 non-null float64
worst concave points       569 non-null float64
worst symmetry             569 non-null float64
worst fractal dimension    569 non-null float64
dtypes: float64(30)
memory usage: 133.4 KB


In [38]: # print out the first 5 df for cancer
         df_features.head()
```

Out[38]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness
0	17.99	10.38	122.80	1001.0	0.11840
1	20.57	17.77	132.90	1326.0	0.08474
2	19.69	21.25	130.00	1203.0	0.10960
3	11.42	20.38	77.58	386.1	0.14250
4	20.29	14.34	135.10	1297.0	0.10030

	mean compactness	mean concavity	mean concave points	mean symmetry
0	0.27760	0.3001	0.14710	0.2419
1	0.07864	0.0869	0.07017	0.1812
2	0.15990	0.1974	0.12790	0.2069
3	0.28390	0.2414	0.10520	0.2597
4	0.13280	0.1980	0.10430	0.1809

	mean fractal dimension	...	worst radius
0	0.07871	...	25.38
1	0.05667	...	24.99
2	0.05999	...	23.57
3	0.09744	...	14.91
4	0.05883	...	22.54

	worst texture	worst perimeter	worst area	worst smoothness
0	17.33	184.60	2019.0	0.1622
1	23.41	158.80	1956.0	0.1238
2	25.53	152.50	1709.0	0.1444
3	26.50	98.87	567.7	0.2098
4	16.67	152.20	1575.0	0.1374

	worst compactness	worst concavity	worst concave points	worst symmetry
0	0.6656	0.7119	0.2654	0.4601
1	0.1866	0.2416	0.1860	0.2750
2	0.4245	0.4504	0.2430	0.3613
3	0.8663	0.6869	0.2575	0.6638
4	0.2050	0.4000	0.1625	0.2364

	worst fractal dimension
0	0.11890
1	0.08902
2	0.08758
3	0.17300
4	0.07678

[5 rows x 30 columns]

In [55]:
```
from sklearn.model_selection import train_test_split
```

In [56]:
```
# model data
df_features
```

Out[56]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	\
0	17.990	10.38	122.80	1001.0	0.11840	
1	20.570	17.77	132.90	1326.0	0.08474	
2	19.690	21.25	130.00	1203.0	0.10960	
3	11.420	20.38	77.58	386.1	0.14250	
4	20.290	14.34	135.10	1297.0	0.10030	
5	12.450	15.70	82.57	477.1	0.12780	
6	18.250	19.98	119.60	1040.0	0.09463	
7	13.710	20.83	90.20	577.9	0.11890	
8	13.000	21.82	87.50	519.8	0.12730	
9	12.460	24.04	83.97	475.9	0.11860	
10	16.020	23.24	102.70	797.8	0.08206	
11	15.780	17.89	103.60	781.0	0.09710	
12	19.170	24.80	132.40	1123.0	0.09740	
13	15.850	23.95	103.70	782.7	0.08401	
14	13.730	22.61	93.60	578.3	0.11310	
15	14.540	27.54	96.73	658.8	0.11390	
16	14.680	20.13	94.74	684.5	0.09867	
17	16.130	20.68	108.10	798.8	0.11700	
18	19.810	22.15	130.00	1260.0	0.09831	
19	13.540	14.36	87.46	566.3	0.09779	
20	13.080	15.71	85.63	520.0	0.10750	
21	9.504	12.44	60.34	273.9	0.10240	
22	15.340	14.26	102.50	704.4	0.10730	
23	21.160	23.04	137.20	1404.0	0.09428	
24	16.650	21.38	110.00	904.6	0.11210	
25	17.140	16.40	116.00	912.7	0.11860	
26	14.580	21.53	97.41	644.8	0.10540	
27	18.610	20.25	122.10	1094.0	0.09440	
28	15.300	25.27	102.40	732.4	0.10820	
29	17.570	15.05	115.00	955.1	0.09847	
..	...	...	...	...	...	
539	7.691	25.44	48.34	170.4	0.08668	
540	11.540	14.44	74.65	402.9	0.09984	
541	14.470	24.99	95.81	656.4	0.08837	
542	14.740	25.42	94.70	668.6	0.08275	
543	13.210	28.06	84.88	538.4	0.08671	
544	13.870	20.70	89.77	584.8	0.09578	
545	13.620	23.23	87.19	573.2	0.09246	
546	10.320	16.35	65.31	324.9	0.09434	
547	10.260	16.58	65.85	320.8	0.08877	
548	9.683	19.34	61.05	285.7	0.08491	
549	10.820	24.21	68.89	361.6	0.08192	
550	10.860	21.48	68.51	360.5	0.07431	
551	11.130	22.44	71.49	378.4	0.09566	
552	12.770	29.43	81.35	507.9	0.08276	
553	9.333	21.94	59.01	264.0	0.09240	
554	12.880	28.92	82.50	514.3	0.08123	

555	10.290	27.61	65.67	321.4	0.09030
556	10.160	19.59	64.73	311.7	0.10030
557	9.423	27.88	59.26	271.3	0.08123
558	14.590	22.68	96.39	657.1	0.08473
559	11.510	23.93	74.52	403.5	0.09261
560	14.050	27.15	91.38	600.4	0.09929
561	11.200	29.37	70.67	386.0	0.07449
562	15.220	30.62	103.40	716.9	0.10480
563	20.920	25.09	143.00	1347.0	0.10990
564	21.560	22.39	142.00	1479.0	0.11100
565	20.130	28.25	131.20	1261.0	0.09780
566	16.600	28.08	108.30	858.1	0.08455
567	20.600	29.33	140.10	1265.0	0.11780
568	7.760	24.54	47.92	181.0	0.05263

	mean compactness	mean concavity	mean concave points	mean symmetry	\
0	0.27760	0.300100	0.147100	0.2419	
1	0.07864	0.086900	0.070170	0.1812	
2	0.15990	0.197400	0.127900	0.2069	
3	0.28390	0.241400	0.105200	0.2597	
4	0.13280	0.198000	0.104300	0.1809	
5	0.17000	0.157800	0.080890	0.2087	
6	0.10900	0.112700	0.074000	0.1794	
7	0.16450	0.093660	0.059850	0.2196	
8	0.19320	0.185900	0.093530	0.2350	
9	0.23960	0.227300	0.085430	0.2030	
10	0.06669	0.032990	0.033230	0.1528	
11	0.12920	0.099540	0.066060	0.1842	
12	0.24580	0.206500	0.111800	0.2397	
13	0.10020	0.099380	0.053640	0.1847	
14	0.22930	0.212800	0.080250	0.2069	
15	0.15950	0.163900	0.073640	0.2303	
16	0.07200	0.073950	0.052590	0.1586	
17	0.20220	0.172200	0.102800	0.2164	
18	0.10270	0.147900	0.094980	0.1582	
19	0.08129	0.066640	0.047810	0.1885	
20	0.12700	0.045680	0.031100	0.1967	
21	0.06492	0.029560	0.020760	0.1815	
22	0.21350	0.207700	0.097560	0.2521	
23	0.10220	0.109700	0.086320	0.1769	
24	0.14570	0.152500	0.091700	0.1995	
25	0.22760	0.222900	0.140100	0.3040	
26	0.18680	0.142500	0.087830	0.2252	
27	0.10660	0.149000	0.077310	0.1697	
28	0.16970	0.168300	0.087510	0.1926	
29	0.11570	0.098750	0.079530	0.1739	
..	...	...	...	...	
539	0.11990	0.092520	0.013640	0.2037	

540	0.11200	0.067370	0.025940	0.1818
541	0.12300	0.100900	0.038900	0.1872
542	0.07214	0.041050	0.030270	0.1840
543	0.06877	0.029870	0.032750	0.1628
544	0.10180	0.036880	0.023690	0.1620
545	0.06747	0.029740	0.024430	0.1664
546	0.04994	0.010120	0.005495	0.1885
547	0.08066	0.043580	0.024380	0.1669
548	0.05030	0.023370	0.009615	0.1580
549	0.06602	0.015480	0.008160	0.1976
550	0.04227	0.000000	0.000000	0.1661
551	0.08194	0.048240	0.022570	0.2030
552	0.04234	0.019970	0.014990	0.1539
553	0.05605	0.039960	0.012820	0.1692
554	0.05824	0.061950	0.023430	0.1566
555	0.07658	0.059990	0.027380	0.1593
556	0.07504	0.005025	0.011160	0.1791
557	0.04971	0.000000	0.000000	0.1742
558	0.13300	0.102900	0.037360	0.1454
559	0.10210	0.111200	0.041050	0.1388
560	0.11260	0.044620	0.043040	0.1537
561	0.03558	0.000000	0.000000	0.1060
562	0.20870	0.255000	0.094290	0.2128
563	0.22360	0.317400	0.147400	0.2149
564	0.11590	0.243900	0.138900	0.1726
565	0.10340	0.144000	0.097910	0.1752
566	0.10230	0.092510	0.053020	0.1590
567	0.27700	0.351400	0.152000	0.2397
568	0.04362	0.000000	0.000000	0.1587

	mean fractal dimension	...	worst radius	\
0	0.07871	...	25.380	
1	0.05667	...	24.990	
2	0.05999	...	23.570	
3	0.09744	...	14.910	
4	0.05883	...	22.540	
5	0.07613	...	15.470	
6	0.05742	...	22.880	
7	0.07451	...	17.060	
8	0.07389	...	15.490	
9	0.08243	...	15.090	
10	0.05697	...	19.190	
11	0.06082	...	20.420	
12	0.07800	...	20.960	
13	0.05338	...	16.840	
14	0.07682	...	15.030	
15	0.07077	...	17.460	
16	0.05922	...	19.070	

```
17                0.07356          ...                20.960
18                0.05395          ...                27.320
19                0.05766          ...                15.110
20                0.06811          ...                14.500
21                0.06905          ...                10.230
22                0.07032          ...                18.070
23                0.05278          ...                29.170
24                0.06330          ...                26.460
25                0.07413          ...                22.250
26                0.06924          ...                17.620
27                0.05699          ...                21.310
28                0.06540          ...                20.270
29                0.06149          ...                20.010
..                    ...          ...                   ...
539               0.07751          ...                 8.678
540               0.06782          ...                12.260
541               0.06341          ...                16.220
542               0.05680          ...                16.510
543               0.05781          ...                14.370
544               0.06688          ...                15.050
545               0.05801          ...                15.350
546               0.06201          ...                11.250
547               0.06714          ...                10.830
548               0.06235          ...                10.930
549               0.06328          ...                13.030
550               0.05948          ...                11.660
551               0.06552          ...                12.020
552               0.05637          ...                13.870
553               0.06576          ...                 9.845
554               0.05708          ...                13.890
555               0.06127          ...                10.840
556               0.06331          ...                10.650
557               0.06059          ...                10.490
558               0.06147          ...                15.480
559               0.06570          ...                12.480
560               0.06171          ...                15.300
561               0.05502          ...                11.920
562               0.07152          ...                17.520
563               0.06879          ...                24.290
564               0.05623          ...                25.450
565               0.05533          ...                23.690
566               0.05648          ...                18.980
567               0.07016          ...                25.740
568               0.05884          ...                 9.456

     worst texture  worst perimeter  worst area  worst smoothness  \
0            17.33           184.60      2019.0           0.16220
1            23.41           158.80      1956.0           0.12380
```

2	25.53	152.50	1709.0	0.14440
3	26.50	98.87	567.7	0.20980
4	16.67	152.20	1575.0	0.13740
5	23.75	103.40	741.6	0.17910
6	27.66	153.20	1606.0	0.14420
7	28.14	110.60	897.0	0.16540
8	30.73	106.20	739.3	0.17030
9	40.68	97.65	711.4	0.18530
10	33.88	123.80	1150.0	0.11810
11	27.28	136.50	1299.0	0.13960
12	29.94	151.70	1332.0	0.10370
13	27.66	112.00	876.5	0.11310
14	32.01	108.80	697.7	0.16510
15	37.13	124.10	943.2	0.16780
16	30.88	123.40	1138.0	0.14640
17	31.48	136.80	1315.0	0.17890
18	30.88	186.80	2398.0	0.15120
19	19.26	99.70	711.2	0.14400
20	20.49	96.09	630.5	0.13120
21	15.66	65.13	314.9	0.13240
22	19.08	125.10	980.9	0.13900
23	35.59	188.00	2615.0	0.14010
24	31.56	177.00	2215.0	0.18050
25	21.40	152.40	1461.0	0.15450
26	33.21	122.40	896.9	0.15250
27	27.26	139.90	1403.0	0.13380
28	36.71	149.30	1269.0	0.16410
29	19.52	134.90	1227.0	0.12550
..	...	...	...	...
539	31.89	54.49	223.6	0.15960
540	19.68	78.78	457.8	0.13450
541	31.73	113.50	808.9	0.13400
542	32.29	107.40	826.4	0.10600
543	37.17	92.48	629.6	0.10720
544	24.75	99.17	688.6	0.12640
545	29.09	97.58	729.8	0.12160
546	21.77	71.12	384.9	0.12850
547	22.04	71.08	357.4	0.14610
548	25.59	69.10	364.2	0.11990
549	31.45	83.90	505.6	0.12040
550	24.77	74.08	412.3	0.10010
551	28.26	77.80	436.6	0.10870
552	36.00	88.10	594.7	0.12340
553	25.05	62.86	295.8	0.11030
554	35.74	88.84	595.7	0.12270
555	34.91	69.57	357.6	0.13840
556	22.88	67.88	347.3	0.12650
557	34.24	66.50	330.6	0.10730

```
558          27.27          105.90       733.5          0.10260
559          37.16           82.28       474.2          0.12980
560          33.17          100.20       706.7          0.12410
561          38.30           75.19       439.6          0.09267
562          42.79          128.70       915.0          0.14170
563          29.41          179.10      1819.0          0.14070
564          26.40          166.10      2027.0          0.14100
565          38.25          155.00      1731.0          0.11660
566          34.12          126.70      1124.0          0.11390
567          39.42          184.60      1821.0          0.16500
568          30.37           59.16       268.6          0.08996

     worst compactness  worst concavity  worst concave points  worst symmetry  \
0              0.66560          0.71190               0.26540          0.4601
1              0.18660          0.24160               0.18600          0.2750
2              0.42450          0.45040               0.24300          0.3613
3              0.86630          0.68690               0.25750          0.6638
4              0.20500          0.40000               0.16250          0.2364
5              0.52490          0.53550               0.17410          0.3985
6              0.25760          0.37840               0.19320          0.3063
7              0.36820          0.26780               0.15560          0.3196
8              0.54010          0.53900               0.20600          0.4378
9              1.05800          1.10500               0.22100          0.4366
10             0.15510          0.14590               0.09975          0.2948
11             0.56090          0.39650               0.18100          0.3792
12             0.39030          0.36390               0.17670          0.3176
13             0.19240          0.23220               0.11190          0.2809
14             0.77250          0.69430               0.22080          0.3596
15             0.65770          0.70260               0.17120          0.4218
16             0.18710          0.29140               0.16090          0.3029
17             0.42330          0.47840               0.20730          0.3706
18             0.31500          0.53720               0.23880          0.2768
19             0.17730          0.23900               0.12880          0.2977
20             0.27760          0.18900               0.07283          0.3184
21             0.11480          0.08867               0.06227          0.2450
22             0.59540          0.63050               0.23930          0.4667
23             0.26000          0.31550               0.20090          0.2822
24             0.35780          0.46950               0.20950          0.3613
25             0.39490          0.38530               0.25500          0.4066
26             0.66430          0.55390               0.27010          0.4264
27             0.21170          0.34460               0.14900          0.2341
28             0.61100          0.63350               0.20240          0.4027
29             0.28120          0.24890               0.14560          0.2756
..                 ...              ...                   ...             ...
539            0.30640          0.33930               0.05000          0.2790
540            0.21180          0.17970               0.06918          0.2329
541            0.42020          0.40400               0.12050          0.3187
542            0.13760          0.16110               0.10950          0.2722
```

543	0.13810	0.10620	0.07958	0.2473
544	0.20370	0.13770	0.06845	0.2249
545	0.15170	0.10490	0.07174	0.2642
546	0.08842	0.04384	0.02381	0.2681
547	0.22460	0.17830	0.08333	0.2691
548	0.09546	0.09350	0.03846	0.2552
549	0.16330	0.06194	0.03264	0.3059
550	0.07348	0.00000	0.00000	0.2458
551	0.17820	0.15640	0.06413	0.3169
552	0.10640	0.08653	0.06498	0.2407
553	0.08298	0.07993	0.02564	0.2435
554	0.16200	0.24390	0.06493	0.2372
555	0.17100	0.20000	0.09127	0.2226
556	0.12000	0.01005	0.02232	0.2262
557	0.07158	0.00000	0.00000	0.2475
558	0.31710	0.36620	0.11050	0.2258
559	0.25170	0.36300	0.09653	0.2112
560	0.22640	0.13260	0.10480	0.2250
561	0.05494	0.00000	0.00000	0.1566
562	0.79170	1.17000	0.23560	0.4089
563	0.41860	0.65990	0.25420	0.2929
564	0.21130	0.41070	0.22160	0.2060
565	0.19220	0.32150	0.16280	0.2572
566	0.30940	0.34030	0.14180	0.2218
567	0.86810	0.93870	0.26500	0.4087
568	0.06444	0.00000	0.00000	0.2871

	worst fractal dimension
0	0.11890
1	0.08902
2	0.08758
3	0.17300
4	0.07678
5	0.12440
6	0.08368
7	0.11510
8	0.10720
9	0.20750
10	0.08452
11	0.10480
12	0.10230
13	0.06287
14	0.14310
15	0.13410
16	0.08216
17	0.11420
18	0.07615
19	0.07259

```
20                  0.08183
21                  0.07773
22                  0.09946
23                  0.07526
24                  0.09564
25                  0.10590
26                  0.12750
27                  0.07421
28                  0.09876
29                  0.07919
..                      ...
539                 0.10660
540                 0.08134
541                 0.10230
542                 0.06956
543                 0.06443
544                 0.08492
545                 0.06953
546                 0.07399
547                 0.09479
548                 0.07920
549                 0.07626
550                 0.06592
551                 0.08032
552                 0.06484
553                 0.07393
554                 0.07242
555                 0.08283
556                 0.06742
557                 0.06969
558                 0.08004
559                 0.08732
560                 0.08321
561                 0.05905
562                 0.14090
563                 0.09873
564                 0.07115
565                 0.06637
566                 0.07820
567                 0.12400
568                 0.07039

[569 rows x 30 columns]
```

In [62]:
```
#splitting for train test split
X = df_features
y = cancer['target']
```

In [63]:
```
X_train, X_test, y_train, y_test = train_test_split(X, np.ravel(y), test_size=0.4, rand
```

```
In [75]: # training Support vector classifier SMV

         from sklearn.svm import SVC

In [76]: model = SVC()

In [88]: model.fit(X_train, y_train)

         # C controls the cost of misclassification on the training data
         # large C -> low bias, high variance
         # low C -> high bias, low variance
         # need to be in between such that bias and variance is both low (around .5)

         # gamma : gausian radio basis function

Out[88]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
           decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
           max_iter=-1, probability=False, random_state=None, shrinking=True,
           tol=0.001, verbose=False)

In [78]: predictions = model.predict(X_test)

In [79]: from sklearn.metrics import classification_report, confusion_matrix

In [83]: # default values, the model prediction no tumors were in 0 class
         print(confusion_matrix(y_test, predictions))
         print("\n")
         print(classification_report(y_test, predictions))

[[  0  84]
 [  0 144]]


             precision    recall  f1-score   support

          0       0.00      0.00      0.00        84
          1       0.63      1.00      0.77       144

avg / total       0.40      0.63      0.49       228



/anaconda2/lib/python2.7/site-packages/sklearn/metrics/classification.py:1135: UndefinedMetricWa
  'precision', 'predicted', average, warn_for)


In [130]: # the above model needs to have parameters adujsted + normalize data
          # for that, gridsearch will be used to search for best parameter(all possbile combinat
          from sklearn.model_selection import GridSearchCV
```

```
In [113]: # want to test the C / gamma parameter from above
          param_grid = {'C':[0.1, 1, 10, 100, 1000],
                        'gamma':[1, 0.1, 0.01, 0.001, 0.0001],
                       'kernel': ['rbf']}

In [114]: # Gridsearch takes estimator ((SVC) , param_grid, verbose = int)
          grid = GridSearchCV(SVC(),  param_grid, verbose = 10)

In [118]: # --- What grid.fit does ---

          # First, it runs the same loop with cross-validation, to find the best parameter combi

          # Once it has the best combination, it runs fit again on all data passed to fit (witho
          # to built a single new model using the best parameter setting.


          grid.fit(X_train,y_train)

Fitting 3 folds for each of 25 candidates, totalling 75 fits
[CV] kernel=rbf, C=0.1, gamma=1 ...
[CV] . kernel=rbf, C=0.1, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=1 ...
[CV] . kernel=rbf, C=0.1, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=1 ...
[CV] . kernel=rbf, C=0.1, gamma=1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.1 ...
[CV]  kernel=rbf, C=0.1, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.1 ...
[CV]  kernel=rbf, C=0.1, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.1 ...
[CV]  kernel=rbf, C=0.1, gamma=0.1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.01 ...
[CV]  kernel=rbf, C=0.1, gamma=0.01, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.01 ...
[CV]  kernel=rbf, C=0.1, gamma=0.01, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.01 ...
[CV]  kernel=rbf, C=0.1, gamma=0.01, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.001 ...
[CV]  kernel=rbf, C=0.1, gamma=0.001, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.001 ...
[CV]  kernel=rbf, C=0.1, gamma=0.001, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.001 ...
[CV]  kernel=rbf, C=0.1, gamma=0.001, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.0001 ...
[CV]  kernel=rbf, C=0.1, gamma=0.0001, score=0.912280701754, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.0001 ...
[CV]  kernel=rbf, C=0.1, gamma=0.0001, score=0.894736842105, total=   0.0s
[CV] kernel=rbf, C=0.1, gamma=0.0001 ...
```

```
[CV]  kernel=rbf, C=0.1, gamma=0.0001, score=0.840707964602, total=   0.0s
[CV] kernel=rbf, C=1, gamma=1 ...
[CV] ... kernel=rbf, C=1, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1, gamma=1 ...
[CV] ... kernel=rbf, C=1, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1, gamma=1 ...
[CV] ... kernel=rbf, C=1, gamma=1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.1 ...
[CV] . kernel=rbf, C=1, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.1 ...


[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    0.1s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   5 out of   5 | elapsed:    0.1s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   6 out of   6 | elapsed:    0.1s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   7 out of   7 | elapsed:    0.1s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   8 out of   8 | elapsed:    0.1s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   9 out of   9 | elapsed:    0.1s remaining:    0.0s


[CV] . kernel=rbf, C=1, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.1 ...
[CV] . kernel=rbf, C=1, gamma=0.1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.01 ...
[CV]  kernel=rbf, C=1, gamma=0.01, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.01 ...
[CV]  kernel=rbf, C=1, gamma=0.01, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.01 ...
[CV]  kernel=rbf, C=1, gamma=0.01, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.001 ...
[CV]  kernel=rbf, C=1, gamma=0.001, score=0.912280701754, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.001 ...
[CV]  kernel=rbf, C=1, gamma=0.001, score=0.885964912281, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.001 ...
[CV]  kernel=rbf, C=1, gamma=0.001, score=0.867256637168, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.0001 ...
[CV]  kernel=rbf, C=1, gamma=0.0001, score=0.947368421053, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.0001 ...
[CV]  kernel=rbf, C=1, gamma=0.0001, score=0.90350877193, total=   0.0s
[CV] kernel=rbf, C=1, gamma=0.0001 ...
[CV]  kernel=rbf, C=1, gamma=0.0001, score=0.87610619469, total=   0.0s
[CV] kernel=rbf, C=10, gamma=1 ...
[CV] .. kernel=rbf, C=10, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=10, gamma=1 ...
[CV] .. kernel=rbf, C=10, gamma=1, score=0.622807017544, total=   0.0s
```

```
[CV] kernel=rbf, C=10, gamma=1 ...
[CV] .. kernel=rbf, C=10, gamma=1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.1 ...
[CV]  kernel=rbf, C=10, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.1 ...
[CV]  kernel=rbf, C=10, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.1 ...
[CV]  kernel=rbf, C=10, gamma=0.1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.01 ...
[CV]  kernel=rbf, C=10, gamma=0.01, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.01 ...
[CV]  kernel=rbf, C=10, gamma=0.01, score=0.631578947368, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.01 ...
[CV]  kernel=rbf, C=10, gamma=0.01, score=0.637168141593, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.001 ...
[CV]  kernel=rbf, C=10, gamma=0.001, score=0.877192982456, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.001 ...
[CV]  kernel=rbf, C=10, gamma=0.001, score=0.868421052632, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.001 ...
[CV]  kernel=rbf, C=10, gamma=0.001, score=0.849557522124, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.0001 ...
[CV]  kernel=rbf, C=10, gamma=0.0001, score=0.964912280702, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.0001 ...
[CV]  kernel=rbf, C=10, gamma=0.0001, score=0.929824561404, total=   0.0s
[CV] kernel=rbf, C=10, gamma=0.0001 ...
[CV]  kernel=rbf, C=10, gamma=0.0001, score=0.884955752212, total=   0.0s
[CV] kernel=rbf, C=100, gamma=1 ...
[CV] . kernel=rbf, C=100, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=100, gamma=1 ...
[CV] . kernel=rbf, C=100, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=100, gamma=1 ...
[CV] . kernel=rbf, C=100, gamma=1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.1 ...
[CV]  kernel=rbf, C=100, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.1 ...
[CV]  kernel=rbf, C=100, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.1 ...
[CV]  kernel=rbf, C=100, gamma=0.1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.01 ...
[CV]  kernel=rbf, C=100, gamma=0.01, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.01 ...
[CV]  kernel=rbf, C=100, gamma=0.01, score=0.631578947368, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.01 ...
[CV]  kernel=rbf, C=100, gamma=0.01, score=0.637168141593, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.001 ...
[CV]  kernel=rbf, C=100, gamma=0.001, score=0.877192982456, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.001 ...
[CV]  kernel=rbf, C=100, gamma=0.001, score=0.868421052632, total=   0.0s
```

```
[CV] kernel=rbf, C=100, gamma=0.001 ...
[CV]  kernel=rbf, C=100, gamma=0.001, score=0.840707964602, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.0001 ...
[CV]  kernel=rbf, C=100, gamma=0.0001, score=0.947368421053, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.0001 ...
[CV]  kernel=rbf, C=100, gamma=0.0001, score=0.885964912281, total=   0.0s
[CV] kernel=rbf, C=100, gamma=0.0001 ...
[CV]  kernel=rbf, C=100, gamma=0.0001, score=0.893805309735, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=1 ...
[CV]  kernel=rbf, C=1000, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=1 ...
[CV]  kernel=rbf, C=1000, gamma=1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=1 ...
[CV]  kernel=rbf, C=1000, gamma=1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.1 ...
[CV]  kernel=rbf, C=1000, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.1 ...
[CV]  kernel=rbf, C=1000, gamma=0.1, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.1 ...
[CV]  kernel=rbf, C=1000, gamma=0.1, score=0.628318584071, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.01 ...
[CV]  kernel=rbf, C=1000, gamma=0.01, score=0.622807017544, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.01 ...
[CV]  kernel=rbf, C=1000, gamma=0.01, score=0.631578947368, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.01 ...
[CV]  kernel=rbf, C=1000, gamma=0.01, score=0.637168141593, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.001 ...
[CV]  kernel=rbf, C=1000, gamma=0.001, score=0.877192982456, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.001 ...
[CV]  kernel=rbf, C=1000, gamma=0.001, score=0.868421052632, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.001 ...
[CV]  kernel=rbf, C=1000, gamma=0.001, score=0.840707964602, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.0001 ...
[CV]  kernel=rbf, C=1000, gamma=0.0001, score=0.929824561404, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.0001 ...
[CV]  kernel=rbf, C=1000, gamma=0.0001, score=0.885964912281, total=   0.0s
[CV] kernel=rbf, C=1000, gamma=0.0001 ...
[CV]  kernel=rbf, C=1000, gamma=0.0001, score=0.858407079646, total=   0.0s


[Parallel(n_jobs=1)]: Done  75 out of  75 | elapsed:    0.8s finished
```

```
Out[118]: GridSearchCV(cv=None, error_score='raise',
                estimator=SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
            decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
            max_iter=-1, probability=False, random_state=None, shrinking=True,
            tol=0.001, verbose=False),
```

```
                fit_params=None, iid=True, n_jobs=1,
                param_grid={'kernel': ['rbf'], 'C': [0.1, 1, 10, 100, 1000], 'gamma': [1, 0.1,
                pre_dispatch='2*n_jobs', refit=True, return_train_score='warn',
                scoring=None, verbose=10)
```

```
In [119]: grid.best_params_
```

```
Out[119]: {'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}
```

```
In [122]: grid.best_estimator_
```

```
Out[122]: SVC(C=10, cache_size=200, class_weight=None, coef0=0.0,
            decision_function_shape='ovr', degree=3, gamma=0.0001, kernel='rbf',
            max_iter=-1, probability=False, random_state=None, shrinking=True,
            tol=0.001, verbose=False)
```

```
In [125]: # rerunning predictions on the model

          grid_predictions = grid.predict(X_test)
```

```
In [126]: print(confusion_matrix(y_test,grid_predictions))
```

```
[[ 76   8]
 [  1 143]]
```

```
In [127]: print(classification_report(y_test,grid_predictions))
```

```
             precision    recall  f1-score   support

          0       0.99      0.90      0.94        84
          1       0.95      0.99      0.97       144

avg / total       0.96      0.96      0.96       228
```

```
In [131]: # this model guesses much better at classifying the data in either the 0 or 1 class fo
          # support vector machine -> use gridsearch | to predict C and gamma values
```