In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
np.random.seed(123)
```

In [2]:

```
data = pd.read_csv('C:/Users/admin/Downloads/data.csv')
```

In [3]:

```
label_encoder = LabelEncoder()
data.iloc[:,0] = label_encoder.fit_transform(data.iloc[:,0]).astype('float64')
```

In [4]:

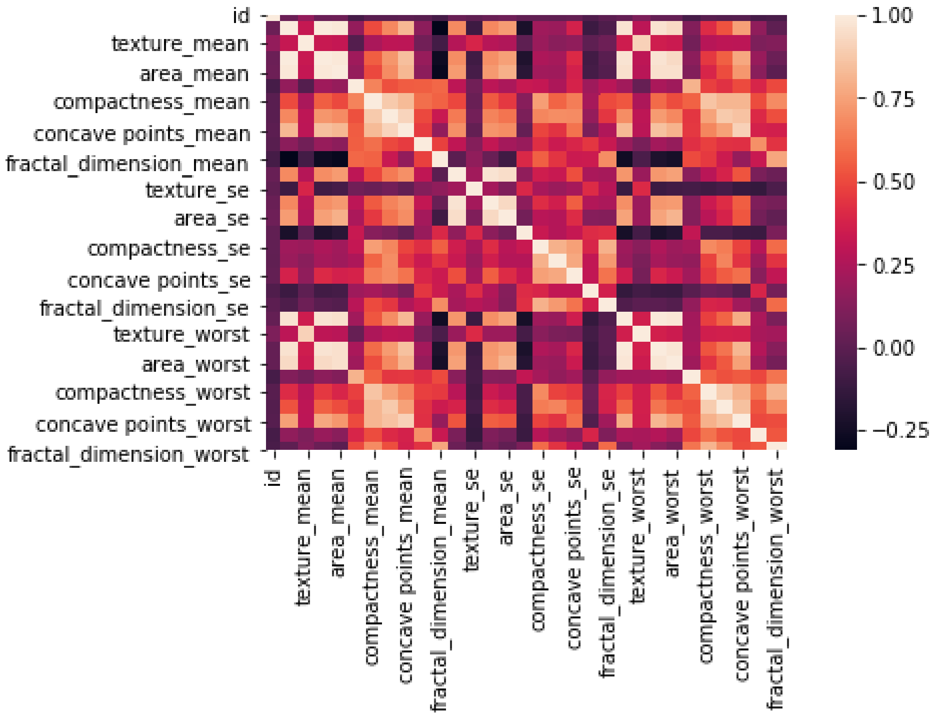
```
corr = data.corr()
```

In [5]:

```
sns.heatmap(corr)
```

Out[5]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x22028364e88>
```



In [21]:

```
columns = np.full((corr.shape[0],), True, dtype='int64')
for i in range(corr.shape[0]):
    for j in range(i+1, corr.shape[0]):
        if corr.iloc[i,j] >= 0.9:
            if columns[j]:
                columns[j] = False
selected_columns = data.columns[columns]
data = data[selected_columns]
```

In [ ]:

```

```

In [10]:

```
import numpy as np
import pandas as pd
import scipy.stats as stats
import matplotlib.pyplot as plt
import math
np.random.seed(6)
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=150000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=100000)
population_ages=np.concatenate((population_ages1,population_ages2))
gujarat_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
gujarat_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
gujarat_ages=np.concatenate((gujarat_ages1,gujarat_ages2))
population_ages.mean()
```

Out[10]:

```
43.000112
```

In [11]:

```
gujarat_ages.mean()
```

Out[11]:

```
39.26
```

In [12]:

```
stats.ttest_1samp(a=gujarat_ages,popmean=population_ages.mean())
```

Out[12]:

```
Ttest_1sampResult(statistic=-2.5742714883655027, pvalue=0.01311868542506167
8)
```

In [13]:

```
np.random.seed(12)
maharashtra_ages1=stats.poisson.rvs(loc=18,mu=33,size=30)
maharashtra_ages2=stats.poisson.rvs(loc=18,mu=13,size=20)
maharashtra_ages=np.concatenate((maharashtra_ages1,maharashtra_ages2))
maharashtra_ages.mean()
```

Out[13]:

```
42.8
```

In [14]:

```
stats.ttest_ind(a=gujarat_ages,b=maharashtra_ages,equal_var=False)
```

Out[14]:

```
Ttest_indResult(statistic=-1.7083870793286842, pvalue=0.09073104343957748)
```

In [15]:

```
np.random.seed(11)
before=stats.norm.rvs(scale=30,loc=250,size=100)
after=before+stats.norm.rvs(scale=5,loc=-1.25,size=100)
weight_df=pd.DataFrame({"weight_before":before,
                        "weight_after":after,
                        "weight_change":after-before})
weight_df.describe()
```

Out[15]:

	weight_before	weight_after	weight_change
count	100.000000	100.000000	100.000000
mean	250.345546	249.115171	-1.230375
std	28.132539	28.422183	4.783696
min	170.400443	165.913930	-11.495286
25%	230.421042	229.148236	-4.046211
50%	250.830805	251.134089	-1.413463
75%	270.637145	268.927258	1.738673
max	314.700233	316.720357	9.759282

In [16]:

```
stats.ttest_rel(a=before,b=after)
```

Out[16]:

```
Ttest_relResult(statistic=2.5720175998568284, pvalue=0.011596444318439859)
```

In [31]:

```
x = np.linspace(-15, 15, 9)
stats.kstest(x,'t',(10,))
```

Out[31]:

```
KstestResult(statistic=0.4425525362393591, pvalue=0.040125239990828926)
```

In [32]:

```
stats.kstest(x,'norm')
```

Out[32]:

```
KstestResult(statistic=0.4443560271592436, pvalue=0.03885014270517116)
```

In [33]:

```
stats.ks_2samp(gujarat_ages,maharashtra_ages)
```

Out[33]:

```
Ks_2sampResult(statistic=0.2, pvalue=0.2719135601522248)
```

In [37]:

```
df=pd.read_csv('C:/Users/admin/Downloads/data.csv')
df['radius_mean']=df.area_mean+df.area_mean*10
df.describe()
```

Out[37]:

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_me
count	5.690000e+02	569.000000	569.000000	569.000000	569.000000	569.000
mean	3.037183e+07	7203.780141	19.289649	91.969033	654.889104	0.096
std	1.250206e+08	3871.055421	4.301036	24.298981	351.914129	0.014
min	8.670000e+03	1578.500000	9.710000	43.790000	143.500000	0.052
25%	8.692180e+05	4623.300000	16.170000	75.170000	420.300000	0.086
50%	9.060240e+05	6062.100000	18.840000	86.240000	551.100000	0.095
75%	8.813129e+06	8609.700000	21.800000	104.100000	782.700000	0.105
max	9.113205e+08	27511.000000	39.280000	188.500000	2501.000000	0.163

8 rows × 32 columns

In [38]:

```
df.corr()
```

Out[38]:

```
                               id  radius_mean  texture_mean  perimeter_mean  area_mean  smo
                     id  1.000000     0.096893      0.099770        0.073159   0.096893
            radius_mean  0.096893     1.000000      0.321086        0.986507   1.000000
           texture_mean  0.099770     0.321086      1.000000        0.329533   0.321086
         perimeter_mean  0.073159     0.986507      0.329533        1.000000   0.986507
              area_mean  0.096893     1.000000      0.321086        0.986507   1.000000
        smoothness_mean -0.012968     0.177028     -0.023389        0.207278   0.177028
       compactness_mean  0.000096     0.498502      0.236702        0.556936   0.498502
         concavity_mean  0.050080     0.685983      0.302418        0.716136   0.685983
    concave points_mean  0.044158     0.823269      0.293464        0.850977   0.823269
          symmetry_mean -0.022114     0.151293      0.071401        0.183027   0.151293
 fractal_dimension_mean -0.052511    -0.283110     -0.076437       -0.261477  -0.283110
              radius_se  0.143048     0.732562      0.275869        0.691765   0.732562
             texture_se -0.007526    -0.066280      0.386358       -0.086761  -0.066280
           perimeter_se  0.137331     0.726628      0.281673        0.693135   0.726628
                area_se  0.177742     0.800086      0.259845        0.744983   0.800086
          smoothness_se  0.096781    -0.166777      0.006614       -0.202694  -0.166777
         compactness_se  0.033961     0.212583      0.191975        0.250744   0.212583
           concavity_se  0.055239     0.207660      0.143293        0.228082   0.207660
      concave points_se  0.078768     0.372320      0.163851        0.407217   0.372320
            symmetry_se -0.017306    -0.072497      0.009127       -0.081629  -0.072497
   fractal_dimension_se  0.025725    -0.019887      0.054458       -0.005523  -0.019887
           radius_worst  0.082405     0.962746      0.352573        0.969476   0.962746
          texture_worst  0.064720     0.287489      0.912045        0.303038   0.287489
        perimeter_worst  0.079986     0.959120      0.358040        0.970387   0.959120
             area_worst  0.107187     0.959213      0.343546        0.941550   0.959213
       smoothness_worst  0.010338     0.123523      0.077503        0.150549   0.123523
      compactness_worst -0.002968     0.390410      0.277830        0.455774   0.390410
        concavity_worst  0.023203     0.512606      0.301025        0.563879   0.512606
   concave points_worst  0.035174     0.722017      0.295316        0.771241   0.722017
         symmetry_worst -0.044224     0.143570      0.105008        0.189115   0.143570
fractal_dimension_worst -0.029866     0.003738      0.119205        0.051019   0.003738
            Unnamed: 32       NaN          NaN           NaN             NaN        NaN

32 rows × 32 columns
```

In [39]:

```
df.cov()
```

Out[39]:

	id	radius_mean	texture_mean	perimeter_mean	area
id	1.563015e+16	4.689241e+10	5.364807e+07	2.222490e+08	4.26294
radius_mean	4.689241e+10	1.498507e+07	5.345932e+03	9.279350e+04	1.36227
texture_mean	5.364807e+07	5.345932e+03	1.849891e+01	3.443976e+01	4.85993
perimeter_mean	2.222490e+08	9.279350e+04	3.443976e+01	5.904405e+02	8.43577
area_mean	4.262946e+09	1.362279e+06	4.859938e+02	8.435772e+03	1.23843
smoothness_mean	-2.280205e+04	9.637959e+00	-1.414779e-03	7.083607e-02	8.7617
compactness_mean	6.318839e+02	1.019142e+02	5.376681e-02	7.147141e-01	9.26493
concavity_mean	4.991277e+05	2.116942e+02	1.036923e-01	1.387234e+00	1.92449
concave points_mean	2.142177e+05	1.236615e+02	4.897693e-02	8.023604e-01	1.12419
symmetry_mean	-7.579262e+04	1.605555e+01	8.418876e-03	1.219216e-01	1.45959
fractal_dimension_mean	-4.635137e+04	-7.737690e+00	-2.321158e-03	-4.485888e-02	-7.0342
radius_se	4.959431e+06	7.864004e+02	3.290374e-01	4.661401e+00	7.14909
texture_se	-5.190618e+05	-1.415388e+02	9.166951e-01	-1.162988e+00	-1.28671
perimeter_se	3.471365e+07	5.687110e+03	2.449449e+00	3.405303e+01	5.17010
area_se	1.010874e+09	1.408937e+05	5.084087e+01	8.234928e+02	1.28085
smoothness_se	3.632916e+04	-1.938431e+00	8.540990e-05	-1.478818e-02	-1.7622
compactness_se	7.603492e+04	1.473698e+01	1.478660e-02	1.091112e-01	1.33972
concavity_se	2.084665e+05	2.426548e+01	1.860393e-02	1.672962e-01	2.20595
concave points_se	6.076269e+04	8.893062e+00	4.348380e-03	6.105470e-02	8.0846
symmetry_se	-1.788548e+04	-2.319861e+00	3.245070e-04	-1.639643e-02	-2.1089
fractal_dimension_se	8.510281e+03	-2.037039e-01	6.197726e-04	-3.551365e-04	-1.8518
radius_worst	4.979381e+07	1.801273e+04	7.329267e+00	1.138581e+02	1.63752
texture_worst	4.973106e+07	6.840074e+03	2.411015e+01	4.525811e+01	6.21824
perimeter_worst	3.360214e+08	1.247597e+05	5.174593e+01	7.923282e+02	1.13417
area_worst	7.629681e+09	2.114118e+06	8.412838e+02	1.302615e+04	1.92192
smoothness_worst	2.951016e+04	1.091765e+01	7.611070e-03	8.352553e-02	9.9251
compactness_worst	-5.838341e+04	2.377826e+02	1.880100e-01	1.742478e+00	2.16166
concavity_worst	6.051816e+05	4.139786e+02	2.701101e-01	2.858506e+00	3.76344
concave points_worst	2.890528e+05	1.837197e+02	8.349085e-02	1.231848e+00	1.67017
symmetry_worst	-3.420616e+05	3.438390e+01	2.794199e-02	2.842997e-01	3.12580
fractal_dimension_worst	-6.743751e+04	2.613185e-01	9.260129e-03	2.239052e-02	2.3756
Unnamed: 32	NaN	NaN	NaN	NaN	

32 rows × 32 columns

In [47]:

```
import seaborn as sn
df1=sn.load_dataset('iris')
sn.pairplot(df,kind='scatter')
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
<ipython-input-47-925fb4734010> in <module>
      1 import seaborn as sn
      2 df1=sn.load_dataset('iris')
----> 3 sn.pairplot(df,kind='scatter')

c:\python\python37\lib\site-packages\seaborn\axisgrid.py in pairplot(data,
hue, hue_order, palette, vars, x_vars, y_vars, kind, diag_kind, markers, h
eight, aspect, dropna, plot_kws, diag_kws, grid_kws, size)
   2105     if grid.square_grid:
   2106         if diag_kind == "hist":
-> 2107             grid.map_diag(plt.hist, **diag_kws)
   2108         elif diag_kind == "kde":
   2109             diag_kws.setdefault("shade", True)

c:\python\python37\lib\site-packages\seaborn\axisgrid.py in map_diag(self,
```

In [ ]:

```
plt.show()
```

In [ ]:

```

```