```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

#Import Cancer data from the Sklearn library
# Dataset can also be found here (http://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+%28diagnostic%29)

from sklearn.datasets import load_breast_cancer
cancer = load_breast_cancer()


df_cancer = pd.DataFrame(np.c_[cancer['data'], cancer['target']], columns = np.append(cancer['feature_names'], ['target']))

df_cancer.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883

5 rows × 31 columns

```
df_cancer.shape
```

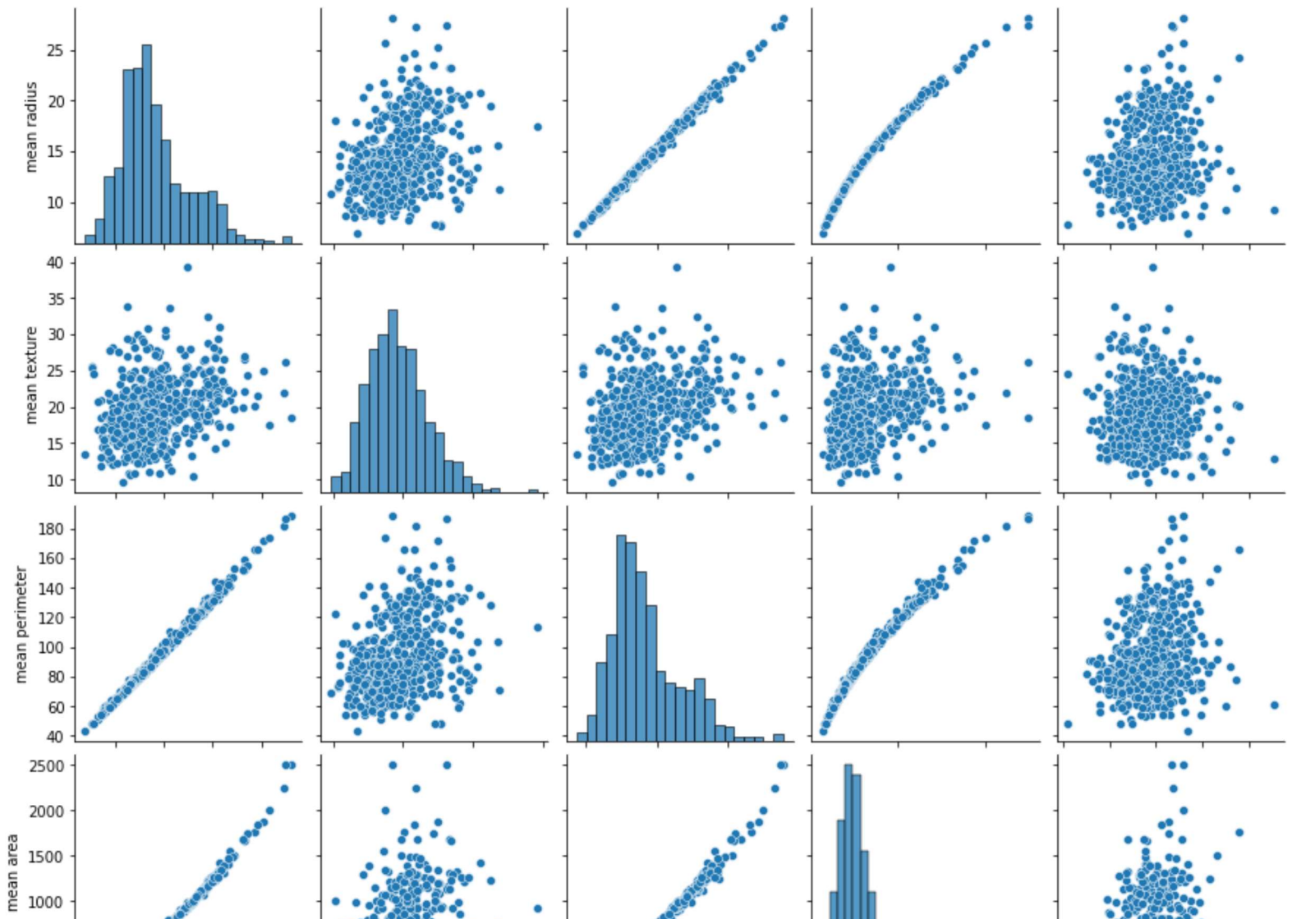
```
(569, 31)
```

```
df_cancer.columns
```

```
Index(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter error', 'area error',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error', 'fractal dimension error',
       'worst radius', 'worst texture', 'worst perimeter', 'worst area',
       'worst smoothness', 'worst compactness', 'worst concavity',
       'worst concave points', 'worst symmetry', 'worst fractal dimension',
       'target'],
      dtype='object')
```

```
# Let's plot out just the first 5 variables (features)
sns.pairplot(df_cancer, vars = ['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness'] )
```

```
<seaborn.axisgrid.PairGrid at 0x7fdf4e7e4cd0>
```




```
# Let's plot out just the first 5 variables (features)
sns.pairplot(df_cancer, hue = 'target', vars = ['mean radius', 'mean texture', 'mean perimeter','mean area','mean smoothness'] )
```

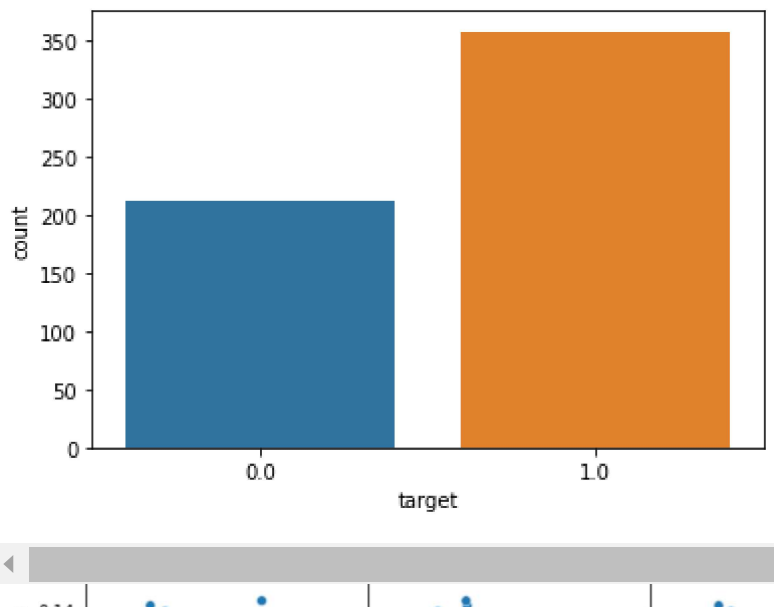
```
<seaborn.axisgrid.PairGrid at 0x7fdf34db1310>
```

```
df_cancer['target'].value_counts()
```

```
1.0    357
0.0    212
Name: target, dtype: int64
```

```
sns.countplot(df_cancer['target'], label = "Count")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg:
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf307ca1d0>
```



```
plt.figure(figsize=(20,12))
sns.heatmap(df_cancer.corr(), annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf30790150>
```

```
X = df_cancer.drop(['target'], axis = 1) # We drop our "target" feature and use all the remaining features in our dataframe to train
X.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883

5 rows × 30 columns


```
y = df_cancer['target']
y.head()
```

```
0    0.0
1    0.0
2    0.0
3    0.0
4    0.0
Name: target, dtype: float64
```

```
from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 20)


print ('The size of our training "X" (input features) is', X_train.shape)
print ('\n')
print ('The size of our testing "X" (input features) is', X_test.shape)
print ('\n')
print ('The size of our training "y" (output feature) is', y_train.shape)
print ('\n')
print ('The size of our testing "y" (output features) is', y_test.shape)
```

```
The size of our training "X" (input features) is (455, 30)


The size of our testing "X" (input features) is (114, 30)


The size of our training "y" (output feature) is (455,)


The size of our testing "y" (output features) is (114,)
```

Support Vector Machine (SVM) Model

```
from sklearn.svm import SVC


svc_model = SVC()


svc_model.fit(X_train, y_train)
```

```
SVC()
```

```
y_predict = svc_model.predict(X_test)
```

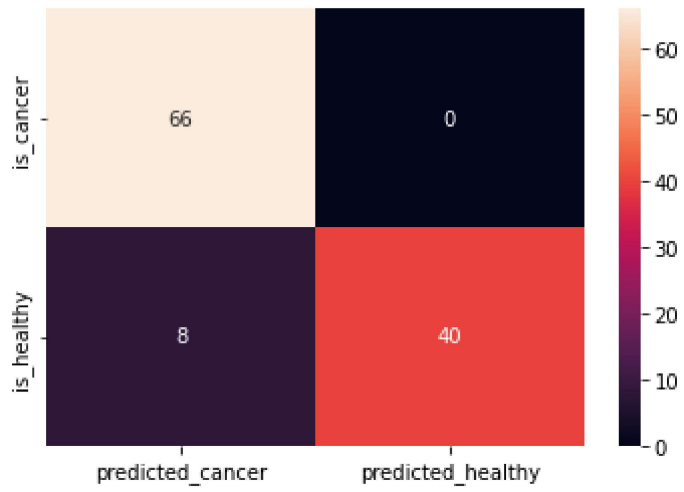
```
from sklearn.metrics import classification_report, confusion_matrix
```

```
cm = np.array(confusion_matrix(y_test, y_predict, labels=[1,0]))
confusion = pd.DataFrame(cm, index=['is_cancer', 'is_healthy'],
                         columns=['predicted_cancer','predicted_healthy'])
confusion
```

	predicted_cancer	predicted_healthy
is_cancer	66	0
is_healthy	8	40

```
sns.heatmap(confusion, annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf2f99cad0>
```




```
print(classification_report(y_test, y_predict))
```

```
                  precision    recall  f1-score   support

             0.0       1.00      0.83      0.91        48
             1.0       0.89      1.00      0.94        66

        accuracy                           0.93       114
       macro avg       0.95      0.92      0.93       114
    weighted avg       0.94      0.93      0.93       114
```

```
X_train_min = X_train.min()
X_train_min
```

```
mean radius                 6.981000
mean texture               10.380000
mean perimeter             43.790000
mean area                 143.500000
mean smoothness             0.052630
mean compactness            0.019380
mean concavity              0.000000
mean concave points         0.000000
mean symmetry               0.106000
mean fractal dimension      0.049960
radius error                0.111500
texture error               0.360200
perimeter error             0.757000
area error                  6.802000
smoothness error            0.001713
compactness error           0.002252
concavity error             0.000000
concave points error        0.000000
symmetry error              0.007882
fractal dimension error     0.000895
worst radius                7.930000
worst texture              12.490000
worst perimeter            50.410000
worst area                185.200000
worst smoothness            0.071170
worst compactness           0.027290
worst concavity             0.000000
worst concave points        0.000000
worst symmetry              0.156500
worst fractal dimension     0.055040
dtype: float64
```

```
X_train_max = X_train.max()
X_train_max
```

```
mean radius                 28.11000
mean texture                39.28000
mean perimeter             188.50000
mean area                 2501.00000
mean smoothness              0.14470
mean compactness             0.34540
mean concavity               0.42680
mean concave points          0.20120
mean symmetry                0.30400
mean fractal dimension       0.09296
radius error                 2.87300
texture error                4.88500
perimeter error             21.98000
area error                 542.20000
smoothness error             0.03113
compactness error            0.13540
concavity error              0.39600
concave points error         0.05279
symmetry error               0.07895
fractal dimension error      0.02984
worst radius                36.04000
worst texture               49.54000
worst perimeter            251.20000
worst area                4254.00000
worst smoothness             0.22260
worst compactness            1.05800
worst concavity              1.25200
worst concave points         0.29100
worst symmetry               0.57740
worst fractal dimension      0.20750
dtype: float64
```

```
X_train_range = (X_train_max- X_train_min)
X_train_range
```

```
mean radius                 21.129000
mean texture                28.900000
mean perimeter             144.710000
mean area                 2357.500000
mean smoothness              0.092070
mean compactness             0.326020
mean concavity               0.426800
mean concave points          0.201200
mean symmetry                0.198000
mean fractal dimension       0.043000
radius error                 2.761500
texture error                4.524800
perimeter error             21.223000
area error                 535.398000
smoothness error             0.029417
compactness error            0.133148
concavity error              0.396000
concave points error         0.052790
symmetry error               0.071068
fractal dimension error      0.028945
worst radius                28.110000
worst texture               37.050000
worst perimeter            200.790000
worst area                4068.800000
worst smoothness             0.151430
worst compactness            1.030710
worst concavity              1.252000
worst concave points         0.291000
worst symmetry               0.420900
worst fractal dimension      0.152460
dtype: float64
```

```
X_train_scaled = (X_train - X_train_min)/(X_train_range)
X_train_scaled.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	di
412	0.114345	0.391003	0.110290	0.053150	0.293907	0.126219	0.087512	0.025487	0.108081	C
461	0.967343	0.549827	0.988943	1.000000	0.605735	0.550334	0.851687	0.839463	0.505556	C
532	0.317052	0.205882	0.303849	0.183245	0.435973	0.163088	0.041050	0.093439	0.288384	C
495	0.373373	0.340138	0.361620	0.227953	0.469643	0.196522	0.159888	0.246074	0.215657	C

```
X_test_min = X_test.min()
X_test_range = (X_test - X_test_min).max()
X_test_scaled = (X_test - X_test_min)/X_test_range


svc_model = SVC()
svc_model.fit(X_train_scaled, y_train)
```

```
SVC()
```

```
y_predict = svc_model.predict(X_test_scaled)
cm = confusion_matrix(y_test, y_predict)
```

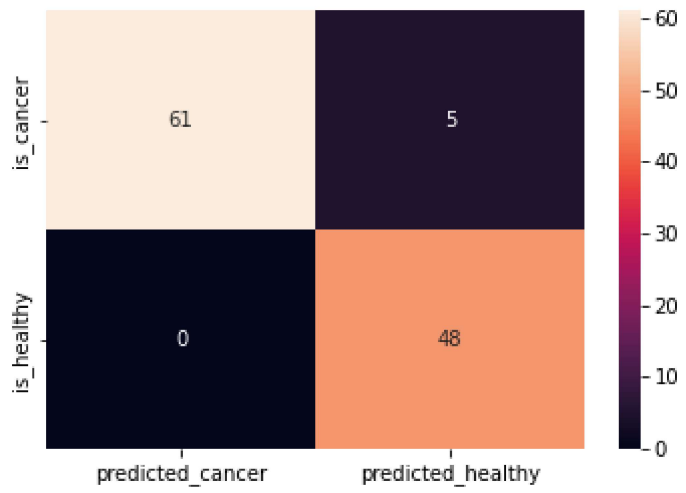
SVM on Normalized data

```
cm = np.array(confusion_matrix(y_test, y_predict, labels=[1,0]))
confusion = pd.DataFrame(cm, index=['is_cancer', 'is_healthy'],
                         columns=['predicted_cancer','predicted_healthy'])
confusion
```

	predicted_cancer	predicted_healthy
is_cancer	61	5
is_healthy	0	48

```
sns.heatmap(confusion,annot=True,fmt="d")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdf2f876b50>
```




```
print(classification_report(y_test,y_predict))
```

```
                precision    recall  f1-score   support

           0.0       0.91      1.00      0.95        48
           1.0       1.00      0.92      0.96        66

      accuracy                           0.96       114
     macro avg       0.95      0.96      0.96       114
  weighted avg       0.96      0.96      0.96       114
```

✓ 0s completed at 1:44 AM