Exercise 3

1.1 Problem Statement:

To train an Support Vector Machine (SVM) based classifier to predict whether the cancer is malignant or benign.

1.2 Description of Machine learning Algorithm:

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three

Advantages of SVM:

- Effective in high dimensional cases
- Its memory efficient as it uses a subset of training points in the decision function called support vectors
- Different kernel functions can be specified for the decision functions and its possible to specify custom kernels

1.3 Description of the Dataset:

Title of the dataset: Breast cancer Dataset

Data Set Characteristics: Multivariate

Area: Life

Number of samples (or instances) in the dataset: 569

Number of attributes (or features): 32

Missing values: No

Attribute Information:

1) ID number
2) Diagnosis (M = malignant, B = benign)
3-32)

Ten real-valued features are computed for each cell nucleus:

a) radius (mean of distances from center to points on the perimeter)
b) texture (standard deviation of gray-scale values)
c) perimeter
d) area
e) smoothness (local variation in radius lengths)
f) compactness (perimeter^2 / area - 1.0)
g) concavity (severity of concave portions of the contour)
h) concave points (number of concave portions of the contour)
i) symmetry
j) fractal dimension ("coastline approximation" - 1)

1.4 Data Preprocessing and Exploratory Data Analysis (EDA):

Data preprocessing is the process of transforming raw data into an understandable format. It is also an important step in data mining as we cannot work with raw data. The quality of the data should be checked before applying machine learning or data mining algorithms.

Major Tasks in Data Preprocessing:

1. Data cleaning
2. Data integration
3. Data reduction
4. Data transformation

Exploratory data analysis (EDA) is a technique that data professionals can use to understand a dataset before they start to model it. Some people refer to EDA as data exploration. The goal of conducting EDA is to determine the characteristics of the dataset. Conducting EDA can help data analysts make predictions and assumptions about data. Often, EDA involves data visualization, including creating graphs like histograms, scatter plots and box plots.

Major Tasks in EDA:

1. Observe your dataset
2. Find any missing values
3. Categorize your values
4. Find the shape of your dataset
5. Identify relationships in your dataset
6. Locate any outliers in your dataset

1.5 Machine Learning Package Used for Model building:

The scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It provides simple and efficient tools for predictive data analysis. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The sklearn.model_selection.train_test_split() method splits arrays or matrices into random train and test subsets. The parameters for the method are as follows:

sklearn.model_selection.train_test_split(*arrays, test_size=None, train_size=None, random_state=Non e, shuffle=True, stratify=None)

It returns lists containing train-test split of inputs.

1.6 Implementation:

```
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()

cancer
```

output:


```
{'DESCR': '.. _breast_cancer_dataset:\n\nBreast cancer wisconsin (diagnostic) dataset\n--------------------------------------\n\n**Data Set Characteristics:**\n\n    :Number of Instances: 569\n\n    :Number of Attributes: 30 numeric, predictive attributes and the class\n\n    :Attribute Information:\n        - radius (mean of distances from center to points on the perimeter)\n        - texture (standard deviation of gray-scale values)\n        - perimeter\n        - area\n        - smoothness (local variation in radius lengths)\n        - compactness (perimeter^2 / area - 1.0)\n        - concavity (severity of concave portions of the contour)\n        - concave points (number of concave portions of the contour)\n        - symmetry\n        - fractal dimension ("coastline approximation" - 1)\n\n        The mean, standard error, and "worst" or largest (mean of the three\n        worst/largest values) of these features were computed for each image,\n        resulting in 30 features.  For instance, field 0 is Mean Radius, field\n        10 is Radius SE, field 20 is Worst Radius.\n\n        - class:\n                - WDBC-Malignant\n                - WDBC-Benign\n\n    :Summary Statistics:\n\n    ===================================== ====== ======\n                                           Min    Max\n    ===================================== ====== ======\n    radius (mean):                         6.981  28.11\n    texture (mean):                        9.71   39.28\n    perimeter (mean):                      43.79  188.5\n    area (mean):                           143.5  2501.0\n    smoothness (mean):                     0.053  0.163\n    compactness (mean):                    0.019  0.345\n    concavity (mean):                      0.0    0.427\n    concave points (mean):                 0.0    0.201\n    symmetry (mean):                       0.106  0.304\n    fractal dimension (mean):              0.05   0.097\n    radius (standard error):               0.112  2.873\n    texture (standard error):              0.36   4.885\n    perimeter (standard error):            0.757  21.98\n    area (standard error):                 6.802  542.2\n    smoothness (standard error):           0.002  0.031\n    compactness (standard error):          0.002  0.135\n    concavity (standard error):            0.0    0.396\n    concave points (standard error):       0.0    0.053\n    symmetry (standard error):             0.008  0.079\n    fractal dimension (standard error):    0.001  0.03\n    radius (worst):                        7.93   36.04\n    texture (worst):                       12.02  49.54\n    perimeter (worst):                     50.41  251.2\n    area (worst):                          185.2  4254.0\n    smoothness (worst):                    0.071  0.223\n    compactness (worst):                   0.027  1.058\n    concavity (worst):                     0.0    1.252\n    concave points (worst):                0.0    0.291\n    symmetry (worst):                      0.156  0.664\n    fractal dimension (worst):             0.055  0.208\n    ===================================== ====== ======\n\n    :Missing Attribute Values: None\n\n    :Class Distribution: 212 - Malignant, 357 - Benign\n\n    :Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian\n\n    :Donor: Nick Street\n\n    :Date: November, 1995\n\nThis is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.\nhttps://goo.gl/U2Uwz2\n\nFeatures are computed from a digitized image of a fine needle\naspirate (FNA) of a breast mass.  They
```

describe\ncharacteristics of the cell nuclei present in the
image.\n\nSeparating plane described above was obtained using\nMultisurface
Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree\nConstruction Via Linear
Programming." Proceedings of the 4th\nMidwest Artificial Intelligence and
Cognitive Science Society,\npp. 97-101, 1992], a classification method which
uses linear\nprogramming to construct a decision tree. Relevant
features\nwere selected using an exhaustive search in the space of 1-
4\nfeatures and 1-3 separating planes.\n\nThe actual linear program used to
obtain the separating plane\nin the 3-dimensional space is that described
in:\n[K. P. Bennett and O. L. Mangasarian: "Robust Linear\nProgramming
Discrimination of Two Linearly Inseparable Sets",\nOptimization Methods and
Software 1, 1992, 23-34].\n\nThis database is also available through the UW CS
ftp server:\n\nftp ftp.cs.wisc.edu\ncd math-prog/cpo-dataset/machine-
learn/WDBC/\n\n.. topic:: References\n\n   - W.N. Street, W.H. Wolberg and
O.L. Mangasarian. Nuclear feature extraction \n   for breast tumor
diagnosis. IS&T/SPIE 1993 International Symposium on \n   Electronic
Imaging: Science and Technology, volume 1905, pages 861-870,\n   San Jose,
CA, 1993.\n   - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer
diagnosis and \n   prognosis via linear programming. Operations Research,
43(4), pages 570-577, \n   July-August 1995.\n   - W.H. Wolberg, W.N.
Street, and O.L. Mangasarian. Machine learning techniques\n   to diagnose
breast cancer from fine-needle aspirates. Cancer Letters 77 (1994) \n   163-
171.',
 'data': array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
         1.189e-01],
        [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
         8.902e-02],
        [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
         8.758e-02],
        ...,
        [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
         7.820e-02],
        [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
         1.240e-01],
        [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
         7.039e-02]]),
 'data_module': 'sklearn.datasets.data',
 'feature_names': array(['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'], dtype='<U23'),
 'filename': 'breast_cancer.csv',
 'frame': None,
 'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
        0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0,
        1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0,
        1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1,
        1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
        0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,

```
        1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
        1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0,
        0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
        1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1,
        1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0,
        0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0,
        0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0,
        1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1,
        1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1,
        1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
        1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1,
        1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1,
        1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 0, 1, 0, 0, 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, 1]),
 'target_names': array(['malignant', 'benign'], dtype='<U9')}
```

Let's view the data in a dataframe.

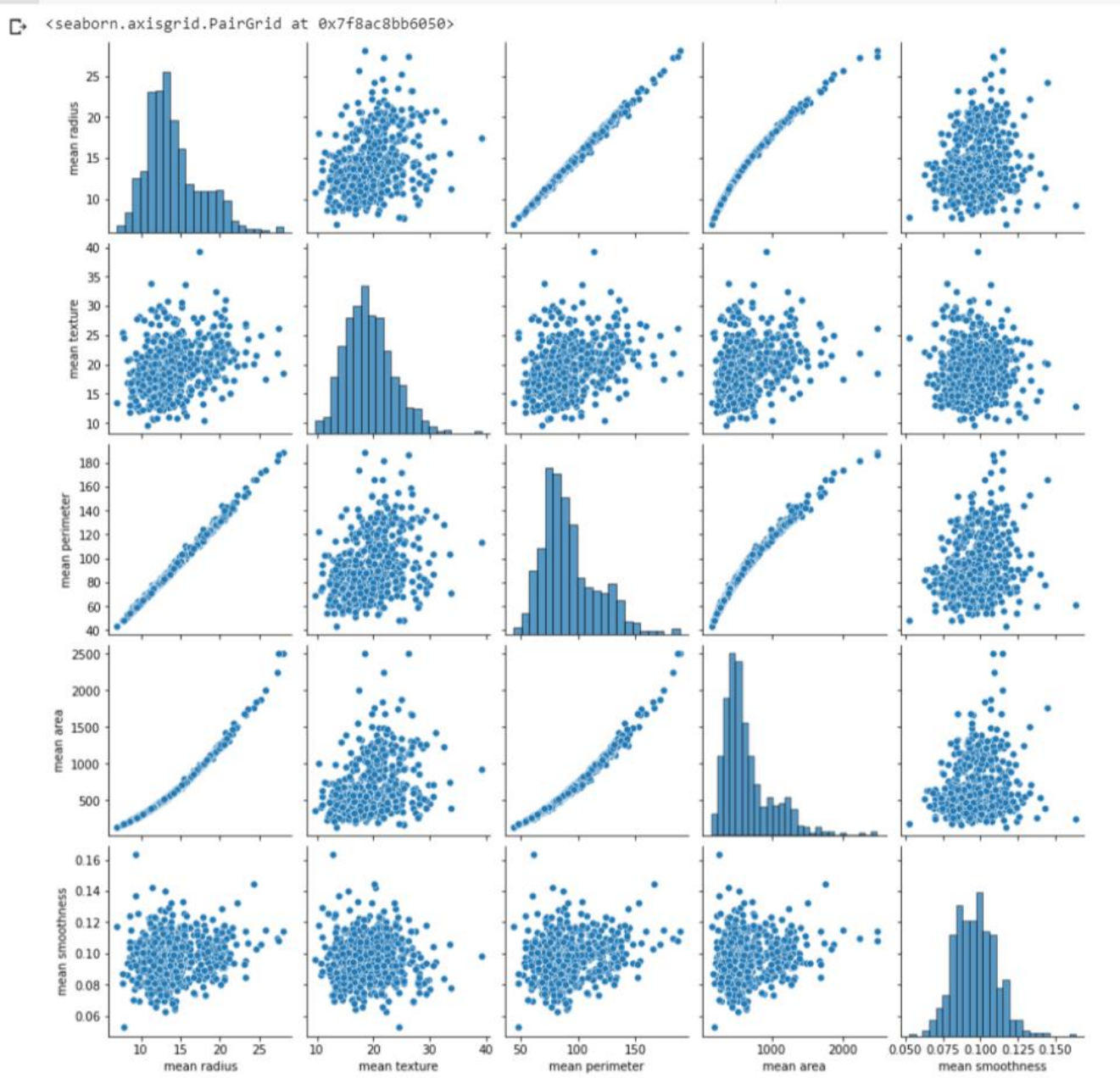
```
df_cancer = pd.DataFrame(np.c_[cancer['data'], cancer['target']], columns = np
.append(cancer['feature_names'], ['target']))

df_cancer.head()
```

Let's Explore Our Dataset

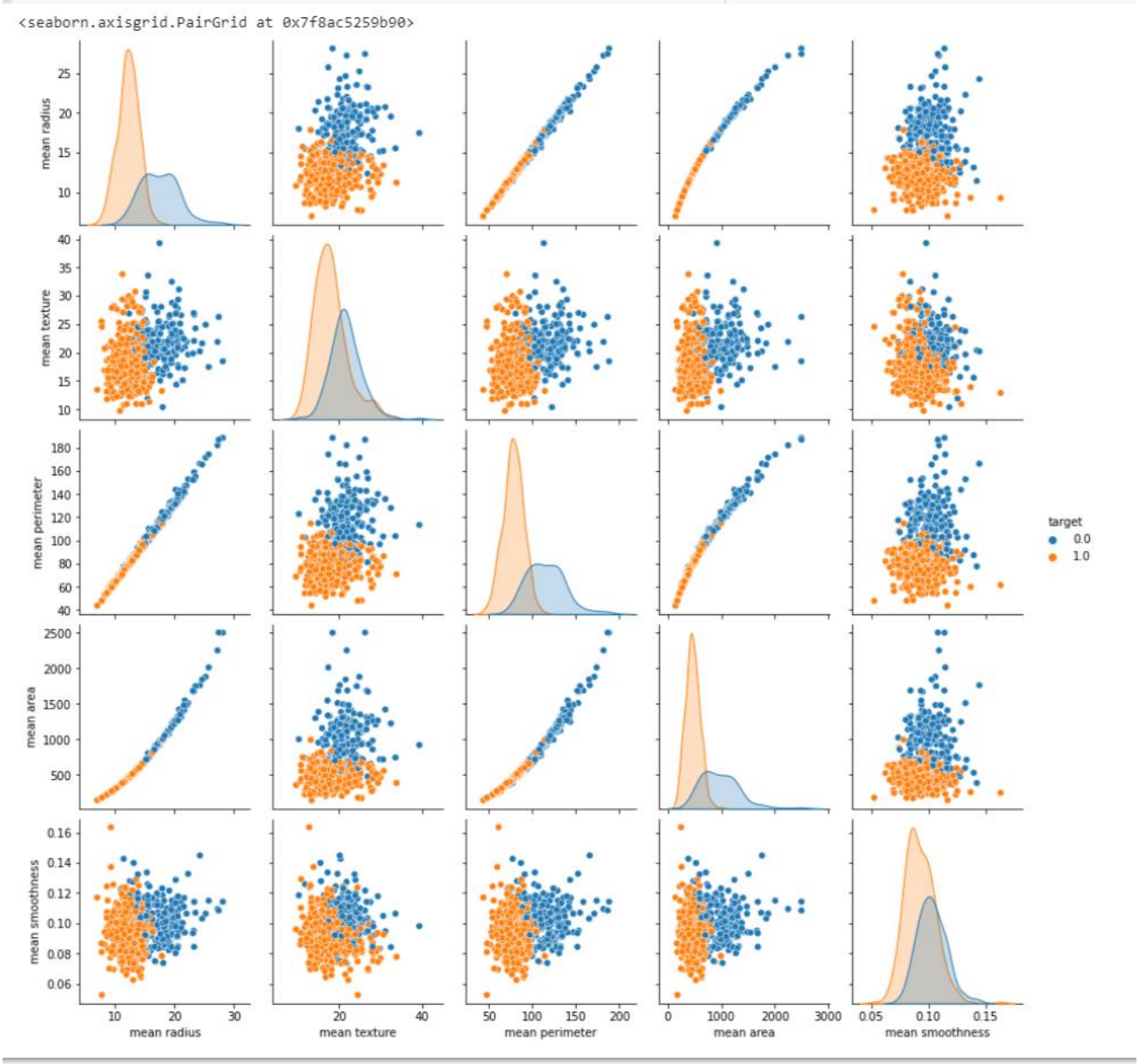
```
df_cancer.shape
(569, 31)
df_cancer.columns
Index(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter 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')
```

The next step is to Visualize our data

```
# 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'] )
```




```
# 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'] )
```

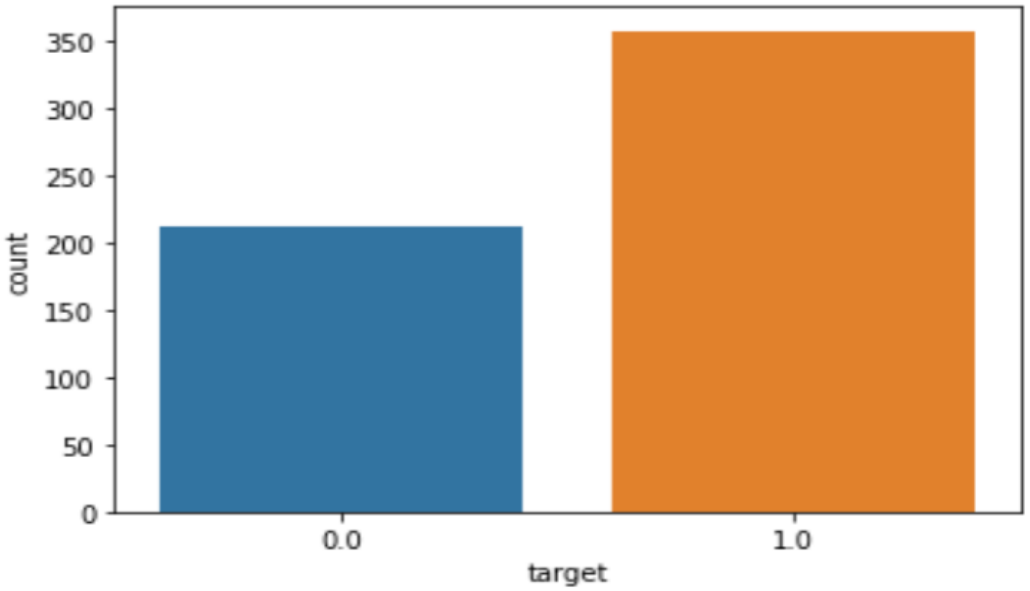


As we can see, we have 212 - Malignant, and 357 - Benign

Let's visulaize our counts

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

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f8ac0ea9ed0>
```



Let's check the correlation between our features

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

Model Training

From our dataset, let's create the target and predictor matrix

"y" = Is the feature we are trying to predict (Output). In this case we are trying to predict whether our "target" is Cancer (Malignant) or not (Benign). I.e. we are going to use the "target" feature here.

"X" = The predictors which are the remaining columns (mean radius, mean texture, mean perimeter, mean area, mean smoothness, etc

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

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23.57	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14.91	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22.54	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678

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
```

Create the training and testing data

Now that we've assigned values to our "X" and "y", the next step is to import the python library that will help us to split our dataset into training and testing data.

Training data = Is the subset of our data used to train our model.

Testing data = Is the subset of our data that the model hasn't seen before. This is used to test the performance of our model.

```
from sklearn.model_selection import train_test_split
```

Let's split our data using 80% for training and the remaining 20% for testing.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, ran
dom_state = 20)
```

Let now check the size our training and testing data.

```
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,)
```

Import Support Vector Machine (SVM) Model

```
from sklearn.svm import SVC
svc_model = SVC()
```

Now, let's train our SVM model with our "training" dataset.

```
svc_model.fit(X_train, y_train)
SVC()
```

Let's use our trained model to make a prediction using our testing data

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

Let us now check the accuracy

```
from sklearn.metrics import accuracy_score, confusion_matrix
score = accuracy_score(y_predict,y_test)
print("Accuracy: ",score)
Accuracy:  0.9298245614035088
```

1.7 Results and Discussion

Hence, to train Support Vector Machine algorithm (SVM) based classifier to predict whether the cancer is malignant or benign is implemented successfully.