ML Project: Breast Cancer Detection Using Machine Learning Classifier

Follow the **“Breast Cancer Detection Using Machine Learning Classifier End to End Project”** step by step to get 3 Bonus.  
1. Raw Dataset  
2. Ready to use Clean Dataset for ML project  
3. Full Project in Jupyter Notebook File

Breast Cancer Detection Machine Learning End to End Project

Goal of the ML project

We have extracted features of breast cancer patient cells and normal person cells. As a Machine learning engineer / Data Scientist has to create an ML model to classify **malignant**and **benign** tumor. To complete this ML project we are using the supervised machine learning classifier algorithm.

Import essential libraries

|  |  |
| --- | --- |
| 1  2  3  4  5 | # import libraries  import pandas as pd # for data manupulation or analysis  import numpy as np # for numeric calculation  import matplotlib.pyplot as plt # for data visualization  import seaborn as sns # for data visualization |

Load breast cancer dataset & explore

|  |  |
| --- | --- |
| 1  2  3 | #Load breast cancer dataset  from sklearn.datasets import load\_breast\_cancer  cancer\_dataset = load\_breast\_cancer() |
| 1 | type(cancer\_dataset) |

**Output >>>** sklearn.utils.Bunch

The scikit-learn store data in an object bunch like a dictionary.

|  |  |
| --- | --- |
| 1  2 | # keys in dataset  cancer\_dataset.keys() |

**Output >>>**dict\_keys([‘data’, ‘target’, ‘target\_names’, ‘DESCR’, ‘feature\_names’, ‘filename’])

|  |  |
| --- | --- |
| 1  2 | # featurs of each cells in numeric format  cancer\_dataset['data'] |

**Output >>>**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | 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]]) |

These numeric values are extracted features of each cell.

|  |  |
| --- | --- |
| 1  2 | # malignant or benign value  cancer\_dataset['target'] |

The **target**stores the values of malignant or benign tumors.

|  |  |
| --- | --- |
| 1  2 | # target value name malignant or benign tumor  cancer\_dataset['target\_names'] |

**Output >>>**array([‘malignant’, ‘benign’], dtype='<U9′)

**0** means malignant tumor  
**1** mean benign tumor

The **cancer\_dataset[‘DESCR’]** store the description of breast cancer dataset.

|  |  |
| --- | --- |
| 1  2 | # description of data  print(cancer\_dataset['DESCR']) |

**Output >>>**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110 | .. \_breast\_cancer\_dataset:    Breast cancer wisconsin (diagnostic) dataset  --------------------------------------------    \*\*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/    .. topic:: References       - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction       for breast tumor diagnosis. IS&amp;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. |

Features name of malignant & benign tumor.

|  |  |
| --- | --- |
| 1  2 | # name of features  print(cancer\_dataset['feature\_names']) |

**Output >>>**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | ['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'] |

When we call load\_breast\_cancer() class it downloads breast\_cancer.csv file and you can see file location.

|  |  |
| --- | --- |
| 1  2 | # location/path of data file  print(cancer\_dataset['filename']) |

**Output >>>** C:\ProgramData\Anaconda3\lib\site-packages\sklearn\datasets\data\breast\_cancer.csv

Create DataFrame

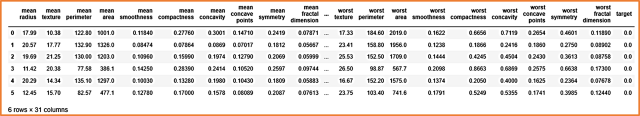
Now, we are creating DataFrame by concate ‘data’ and ‘target’ together and give columns name.

|  |  |
| --- | --- |
| 1  2  3 | # create datafrmae  cancer\_df = pd.DataFrame(np.c\_[cancer\_dataset['data'],cancer\_dataset['target']],               columns = np.append(cancer\_dataset['feature\_names'], ['target'])) |

Head of cancer DataFrame

|  |  |
| --- | --- |
| 1  2 | # Head of cancer DataFrame  cancer\_df.head(6) |

**Output >>>**



The tail of cancer DataFrame

|  |  |
| --- | --- |
| 1  2 | # Tail of cancer DataFrame  cancer\_df.tail(6) |

**Output >>>**



Getting information of cancer DataFrame using ‘**.info()**‘ method.

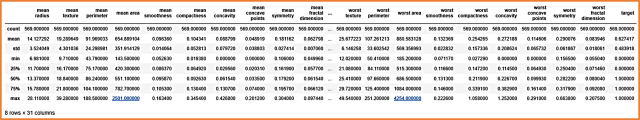
|  |  |
| --- | --- |
| 1  2 | # Information of cancer Dataframe  cancer\_df.info() |

**Output >>>**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | <class 'pandas.core.frame.DataFrame'>  RangeIndex: 569 entries, 0 to 568  Data columns (total 31 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  target                     569 non-null float64  dtypes: float64(31)  memory usage: 137.9 KB |

|  |  |
| --- | --- |
| 1  2 | # Numerical distribution of data  cancer\_df.describe() |

**Output >>>**



We have clean and well formated DataFrame, so DtaFrame is ready to visualize.

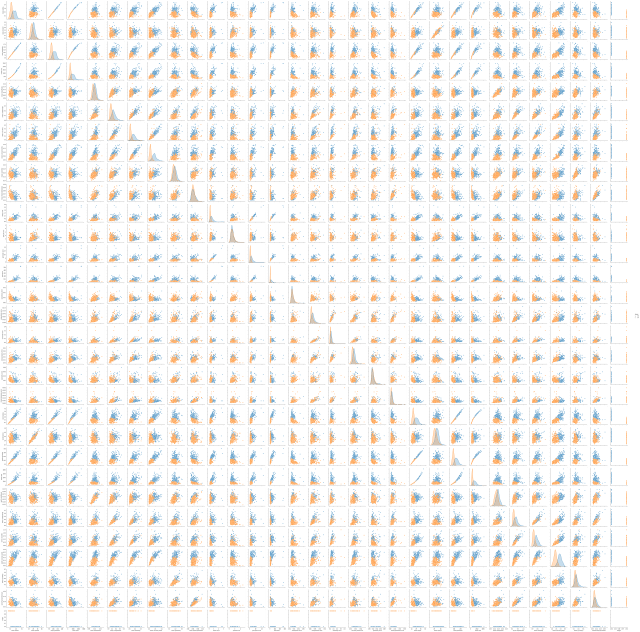
Data Visualization

Pair plot of breast cancer data

Basically, the pair plot is used to show the numeric distribution in the scatter plot.

|  |  |
| --- | --- |
| 1  2 | # Paiplot of cancer dataframe  sns.pairplot(cancer\_df, hue = 'target') |

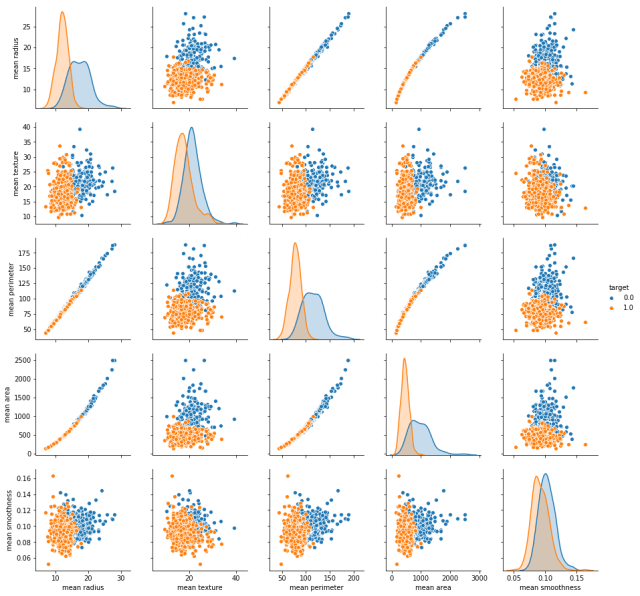
**Output >>>**



Pair plot of sample feature of DataFrame

|  |  |
| --- | --- |
| 1  2  3 | # pair plot of sample feature  sns.pairplot(cancer\_df, hue = 'target',               vars = ['mean radius', 'mean texture', 'mean perimeter', 'mean area', 'mean smoothness'] ) |

**Output >>>**



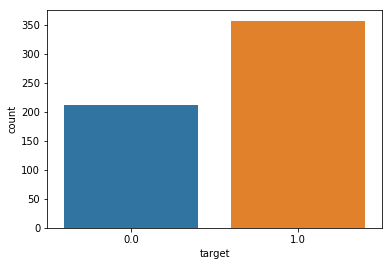
The pair plot showing malignant and benign tumor data distributed in two classes. It is easy to differentiate in the pair plot.

Counterplot

Showing the total count of malignant and benign tumor patients in counterplot.

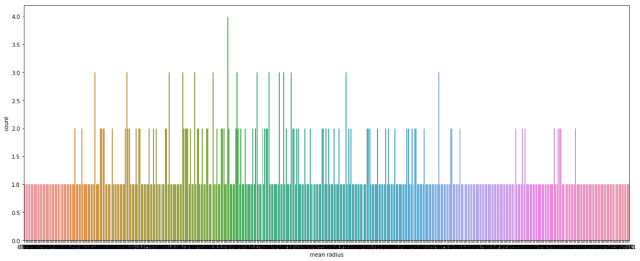
|  |  |
| --- | --- |
| 1  2 | # Count the target class  sns.countplot(cancer\_df['target']) |

**Output >>>**



In the below counterplot max samples mean radius is equal to 1.

|  |  |
| --- | --- |
| 1  2  3 | # counter plot of feature mean radius  plt.figure(figsize = (20,8))  sns.countplot(cancer\_df['mean radius']) |



Heatmap

Heatmap of breast cancer DataFrame

In the below heatmap we can see the variety of different feature’s value. The value of feature **‘mean area’**and **‘worst area’**are greater than other and ‘mean perimeter’, ‘area error’, and ‘worst perimeter’ value slightly less but greater than remaining features.

|  |  |
| --- | --- |
| 1  2  3 | # heatmap of DataFrame  plt.figure(figsize=(16,9))  sns.heatmap(cancer\_df) |

**Output >>>**

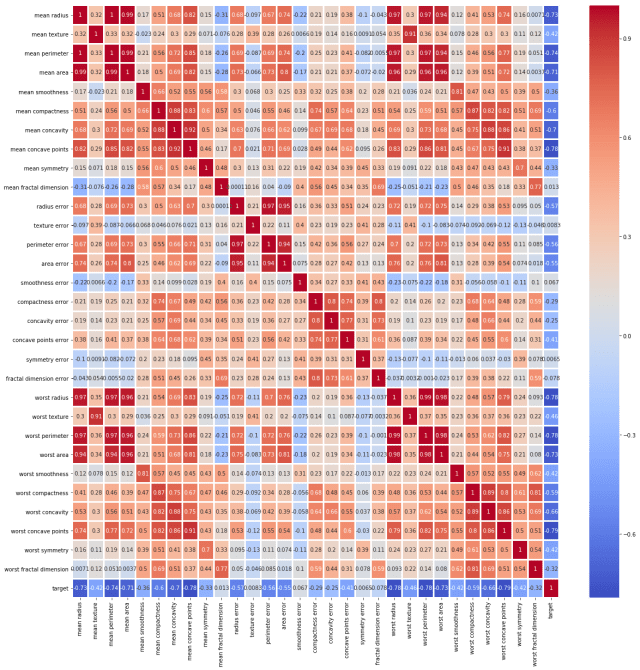
heatmap of breast cancer dataframe ml projects


Heatmap of a correlation matrix

To find a correlation between each feature and target we visualize heatmap using the correlation matrix.

|  |  |
| --- | --- |
| 1  2  3 | # Heatmap of Correlation matrix of breast cancer DataFrame  plt.figure(figsize=(20,20))  sns.heatmap(cancer\_df.corr(), annot = True, cmap ='coolwarm', linewidths=2) |

**Output >>>**



Correlation barplot

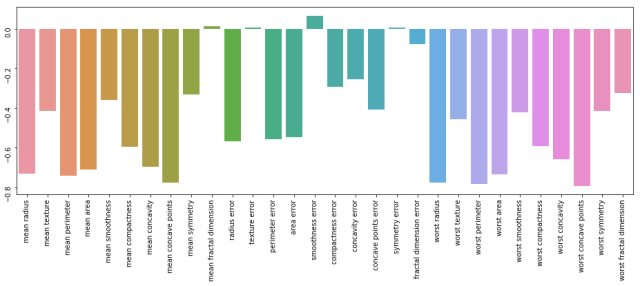
Taking the correlation of each feature with the target and the visualize barplot.

|  |  |
| --- | --- |
| 1  2  3 | # create second DataFrame by droping target  cancer\_df2 = cancer\_df.drop(['target'], axis = 1)  print("The shape of 'cancer\_df2' is : ", cancer\_df2.shape) |

**Output >>>**The shape of ‘cancer\_df2’ is : (569, 30)

|  |  |
| --- | --- |
| 1  2  3  4 | # visualize correlation barplot  plt.figure(figsize = (16,5))  ax = sns.barplot(cancer\_df2.corrwith(cancer\_df.target).index, cancer\_df2.corrwith(cancer\_df.target))  ax.tick\_params(labelrotation = 90) |

**Output >>>**



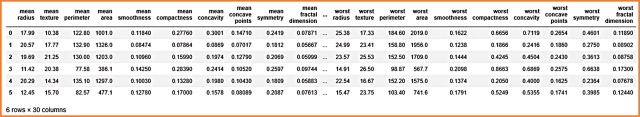
In the above correlation barplot only feature ‘smoothness error’ is strongly positively correlated with the target than others. The features ‘mean factor dimension’, ‘texture error’, and ‘symmetry error’ are very less positive correlated and others remaining are strongly negatively correlated.

Data Preprocessing

Split DataFrame in train and test

|  |  |
| --- | --- |
| 1  2  3 | # input variable  X = cancer\_df.drop(['target'], axis = 1)  X.head(6) |

**Output >>>**



|  |  |
| --- | --- |
| 1  2  3 | # output variable  y = cancer\_df['target']  y.head(6) |

**Output >>>**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | 0    0.0  1    0.0  2    0.0  3    0.0  4    0.0  5    0.0  Name: target, dtype: float64 |
| 1  2  3 | # split dataset into train and test  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= 5) | |