In [1]:
```
import numpy as np # arrays and data manipulation
import matplotlib.pyplot as plt # data visualization
import pandas as pd # read, clean, and rearrange data
import seaborn as sns # more styles and colors for data visualization on top of matplotlib

from sklearn.datasets import load_breast_cancer # import dataset from sklearn
from sklearn.model_selection import train_test_split # divide data into train and test parts
from sklearn.preprocessing import StandardScaler # feature scaling so that data works with models better
from sklearn.svm import SVC # support vector machine -> support vector classification: scales quadratically
from sklearn.metrics import classification_report, confusion_matrix
    # classification report gives metrics like accuracy, precision, f1-score, support
    # confusion matrix gives a confusion matrix (TP, TN, FP, FN) between y_true and y_predict

breast_cancer = load_breast_cancer()
```

In [2]:
```
breast_cancer
```

Out[2]:

```
{'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]]),
 'target': array([0, 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, 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, 1, 0, 0, 0, 0, 0, 0, 1]),
 'frame': None,
 'target_names': array(['malignant', 'benign'], dtype='<U9'),
 '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 Attrib
utes: 30 numeric, predictive attributes and the class\n\n:Attribute Information:\n    - radius (mean of dis
tances 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 (numb
er 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 Me
an Radius, field\n    10 is Radius SE, field 20 is Worst Radius.\n\n    - class:\n            - WDBC-Malign
ant\n            - WDBC-Benign\n\n:Summary Statistics:\n\n===================================== ====== ====
==\n                                        Min    Max\n===================================== ====== ======
\nradius (mean):                      6.981  28.11\ntexture (mean):                     9.71   39.28\np
erimeter (mean):                    43.79  188.5\narea (mean):                       143.5  2501.0\nsmo
othness (mean):                   0.053  0.163\ncompactness (mean):                 0.019  0.345\nconcav
ity (mean):                    0.0    0.427\nconcave points (mean):             0.0    0.201\nsymmetry
(mean):                     0.106  0.304\nfractal dimension (mean):           0.05   0.097\nradius (stan
dard error):             0.112  2.873\ntexture (standard error):           0.36   4.885\nperimeter (stan
dard error):          0.757  21.98\narea (standard error):              6.802  542.2\nsmoothness (standa
rd error):         0.002  0.031\ncompactness (standard error):       0.002  0.135\nconcavity (standard e
rror):          0.0    0.396\nconcave points (standard error):     0.0    0.053\nsymmetry (standard erro
r):           0.008  0.079\nfractal dimension (standard error):   0.001  0.03\nradius (worst):
7.93   36.04\ntexture (worst):                     12.02  49.54\nperimeter (worst):                   50.
41  251.2\narea (worst):                       185.2  4254.0\nsmoothness (worst):                 0.071
0.223\ncompactness (worst):                0.027  1.058\nconcavity (worst):                  0.0    1.2
52\nconcave points (worst):             0.0    0.291\nsymmetry (worst):                   0.156  0.664
\nfractal 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. W
illiam H. Wolberg, W. Nick Street, Olvi L. Mangasarian\n\n:Donor: Nick Street\n\n:Date: November, 1995\n\nT
```

```
his is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.\nhttps://goo.gl/U2Uwz2\n\nFeatures a
re computed from a digitized image of a fine needle\naspirate (FNA) of a breast mass.  They describe\nchara
cteristics 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." Pr
oceedings 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 Se
ts",\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.. dropdown:: Refere
nces\n\n  - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction\n    for breast tumo
r diagnosis. IS&T/SPIE 1993 International Symposium on\n    Electronic Imaging: Science and Technology, vol
ume 1905, pages 861-870,\n    San Jose, CA, 1993.\n  - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Brea
st 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.\n',
 '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',
 'data_module': 'sklearn.datasets.data'}
```

In [4]:
```
df_data = np.c_[breast_cancer['data'], breast_cancer['target']] # stack data array next to target array alon
df_cols = np.append(breast_cancer['feature_names'], ['target']) # concatenate feature names with a column ca
breast_cancer_df = pd.DataFrame(df_data, columns = df_cols)
```

In [6]:
```
breast_cancer_df.head()
```

Out[6]:

| | mean radius | mean texture | mean perimeter | mean area | mean smoothness | mean compactness | mean concavity | mean concave points | mean symmetry | mean fractal dimension | ... | wor textu |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 17.99 | 10.38 | 122.80 | 1001.0 | 0.11840 | 0.27760 | 0.3001 | 0.14710 | 0.2419 | 0.07871 | ... | 17.3 |
| **1** | 20.57 | 17.77 | 132.90 | 1326.0 | 0.08474 | 0.07864 | 0.0869 | 0.07017 | 0.1812 | 0.05667 | ... | 23.4 |
| **2** | 19.69 | 21.25 | 130.00 | 1203.0 | 0.10960 | 0.15990 | 0.1974 | 0.12790 | 0.2069 | 0.05999 | ... | 25.5 |
| **3** | 11.42 | 20.38 | 77.58 | 386.1 | 0.14250 | 0.28390 | 0.2414 | 0.10520 | 0.2597 | 0.09744 | ... | 26.5 |
| **4** | 20.29 | 14.34 | 135.10 | 1297.0 | 0.10030 | 0.13280 | 0.1980 | 0.10430 | 0.1809 | 0.05883 | ... | 16.6 |

5 rows × 31 columns

In [7]:
```
breast_cancer_df.shape
```

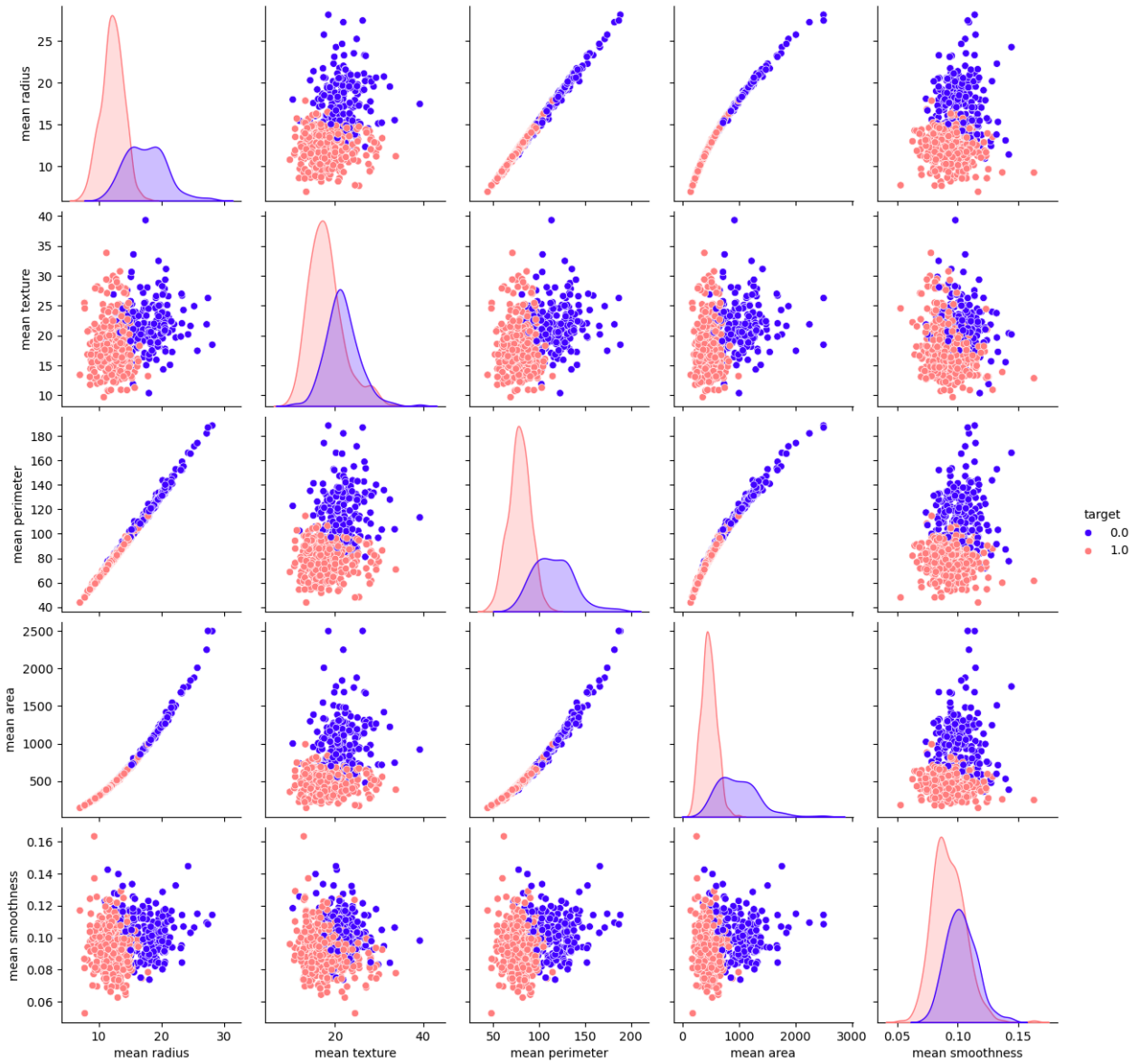
Out[7]:
```
(569, 31)
```

In [8]:
```
breast_cancer_df.columns
```

Out[8]:
```
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')
```

In [9]:
```
sns.pairplot(breast_cancer_df, hue = 'target', palette = 'gnuplot2', vars = ['mean radius', 'mean texture',
# 5x5 grid of plots of the different vars plotted against each other
# diagonal plots are KDE of each var (kernal density), non-diagonal plots are scatter plots of one var again
# colors are by target value so red = benign, blue = malignant
```

Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x22f54bed9d0>
```

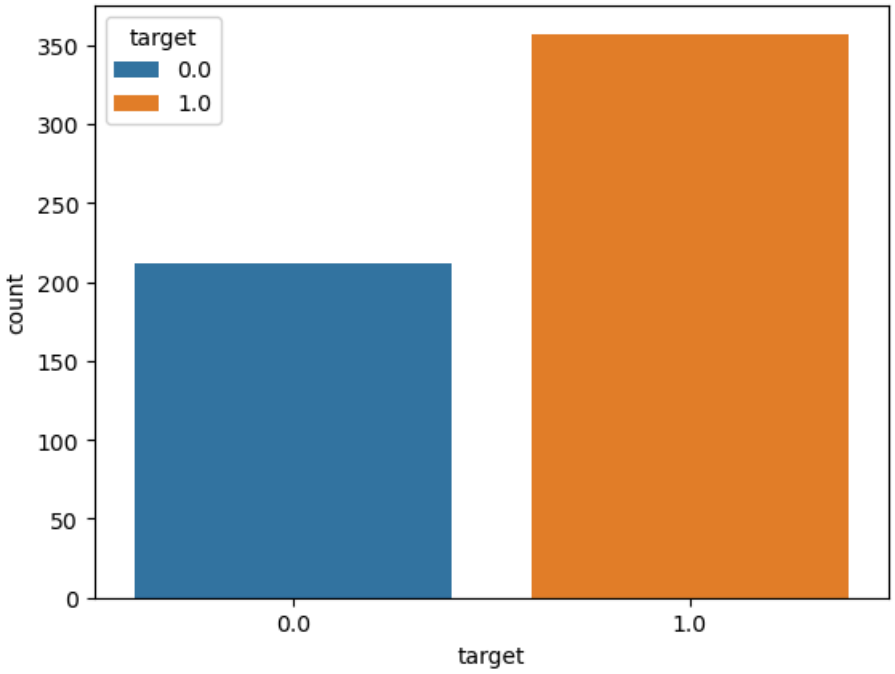



In [10]:
```
breast_cancer_df['target'].value_counts() # number of occurences of each value within targets
```

Out[10]:
```
target
1.0    357
0.0    212
Name: count, dtype: int64
```

In [11]:
```
sns.countplot(data = breast_cancer_df, x = 'target', hue = 'target') # plot histogram of number of occurence
```

Out[11]:
```
<Axes: xlabel='target', ylabel='count'>
```




In [12]:
```
train_data = breast_cancer_df.drop(['target'], axis = 1) # training data removes target values
train_data.head()
```

Out[12]:

| | mean radius | mean texture | mean perimeter | mean area | mean smoothness | mean compactness | mean concavity | mean concave points | mean symmetry | mean fractal dimension | ... | worst radius |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 17.99 | 10.38 | 122.80 | 1001.0 | 0.11840 | 0.27760 | 0.3001 | 0.14710 | 0.2419 | 0.07871 | ... | 25.38 |
| **1** | 20.57 | 17.77 | 132.90 | 1326.0 | 0.08474 | 0.07864 | 0.0869 | 0.07017 | 0.1812 | 0.05667 | ... | 24.99 |
| **2** | 19.69 | 21.25 | 130.00 | 1203.0 | 0.10960 | 0.15990 | 0.1974 | 0.12790 | 0.2069 | 0.05999 | ... | 23.57 |
| **3** | 11.42 | 20.38 | 77.58 | 386.1 | 0.14250 | 0.28390 | 0.2414 | 0.10520 | 0.2597 | 0.09744 | ... | 14.91 |
| **4** | 20.29 | 14.34 | 135.10 | 1297.0 | 0.10030 | 0.13280 | 0.1980 | 0.10430 | 0.1809 | 0.05883 | ... | 22.54 |

5 rows × 30 columns

In [13]:
```
target_data = breast_cancer_df['target'] # target data is only target values
target_data.head()
```

Out[13]:
```
0    0.0
1    0.0
2    0.0
3    0.0
4    0.0
Name: target, dtype: float64
```

In [14]:
```
X_train, X_test, y_train, y_test = train_test_split(train_data, target_data, test_size = 0.2, random_state =
# split train and target data into training and test datasets, 80% data for training, 20% data for testing,
```

In [15]:
```
print("X_train:", X_train.shape)
print("X_test:", X_test.shape)
print("y_train:", y_train.shape)
print("y_test:", y_test.shape)
```

```
X_train: (455, 30)
X_test: (114, 30)
y_train: (455,)
y_test: (114,)
```

In [16]:
```
sc = StandardScaler()
X_train = sc.fit_transform(X_train) # fit: compute mean and stdev of X_train, tranform: scale with fit data
X_test = sc.transform(X_test) # transform: scale with fit data of X_train
```

In [17]:
```
svc_model = SVC()
svc_model.fit(X_train, y_train) # fit the model on training data
```

Out[17]:

▾ SVC

```
SVC()
```

In [19]:
```
prediction = svc_model.predict(X_test) # predict test data based on fitted model
confusion_mat = np.array(confusion_matrix(y_test, prediction, labels=[1,0])) # create a confusion matrix wit
confusion_mat = pd.DataFrame(confusion_mat, index = ["is_malignant", "is_benign"], columns = ["pred_malignan
confusion_mat
```

Out[19]:

| | pred_malignant | pred_benign |
|---|---|---|
| **is_malignant** | 66 | 0 |
| **is_benign** | 1 | 47 |

In [20]:
```
print(classification_report(y_test, prediction)) # print classification report of pred vs actual data
```

```
              precision    recall  f1-score   support

         0.0       1.00      0.98      0.99        48
         1.0       0.99      1.00      0.99        66

    accuracy                           0.99       114
   macro avg       0.99      0.99      0.99       114
weighted avg       0.99      0.99      0.99       114
```

In [ ]:
```

```