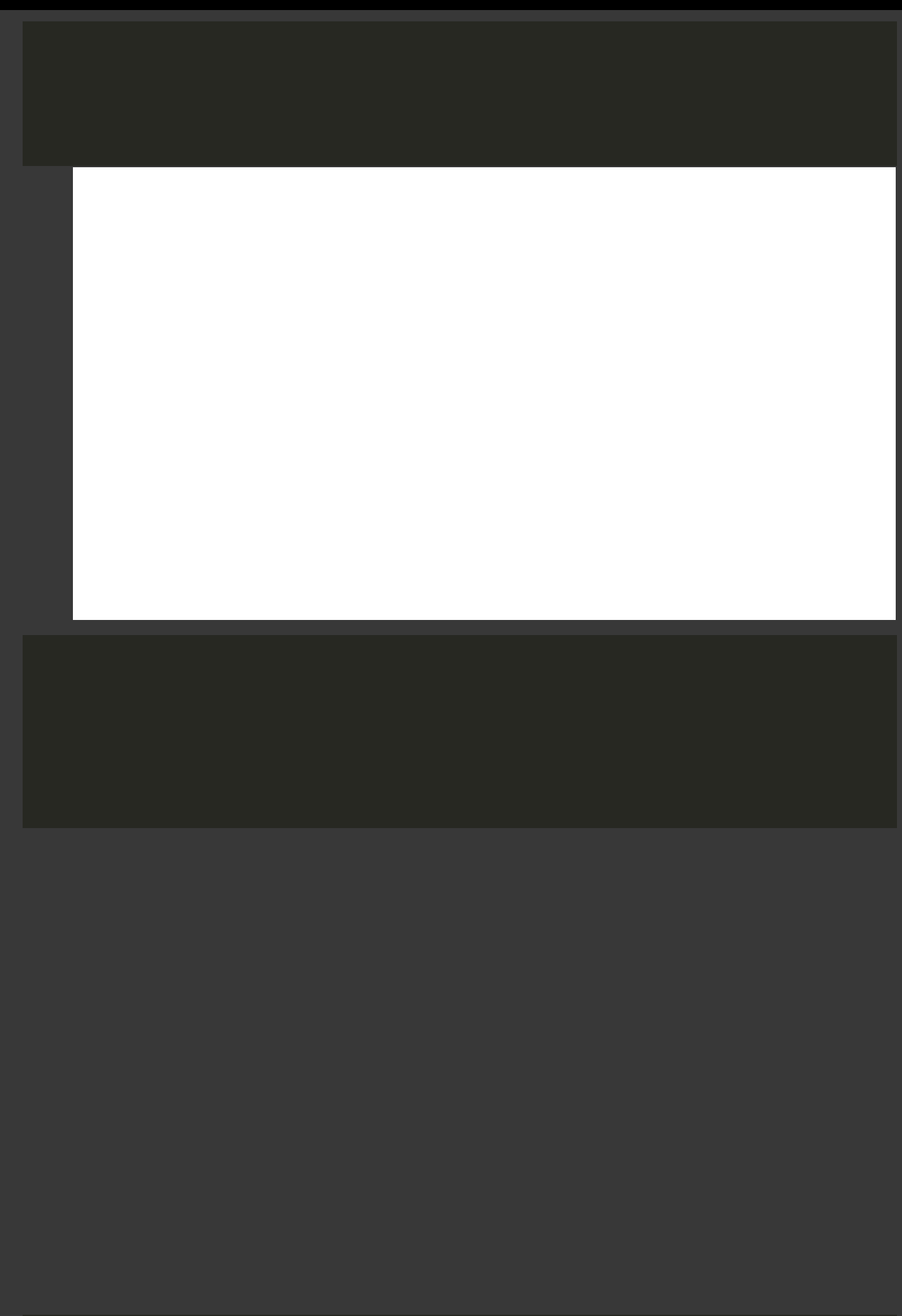
9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



* import pandas as pd

2 import numpy as np

3

4 dataset = pd.read\_csv('/content/mushroom edibility classification dataset.csv') 5 dataset.head()

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  | **stalk-** |  |
|  | **Unnamed:** | **class** | **cap-** | **cap-** | **cap-** | **bruises** | **odor** | **stalk- stalk- surface-** | | |  |
|  | **0** |  | **shape** | **surface** | **color** |  |  | **shape** | **root** | **above-** |  |
|  |  |  |  |  |  |  |  |  |  | **ring** |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | 0 | poisonous | 5.0 | 2 | 4.0 | bruises | 6 | 0 | 3 | 2 |  |
| is there |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **1** | 1 | edible | 5.0 | 2 | 9.0 | bruises | 0 | 0 | 2 | 2 |  |
| is there |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **2** | 2 | edible | 0.0 | 2 | 8.0 | bruises | 3 | 0 | 2 | 2 |  |
| is there |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **3** | 3 | poisonous | 5.0 | 3 | 8.0 | bruises | 6 | 0 | 3 | 2 |  |
| is there |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **4** | 4 | edible | 5.0 | 2 | 3.0 | no | 5 | 1 | 3 | 2 |  |
| bruises |  |
|  |  |  |  |  |  |  |  |  |  |  |

* from sklearn.impute import SimpleImputer
* impute = SimpleImputer(missing\_values=np.nan, strategy='mean')

3 impute.fit(dataset[['cap-shape']])

4 dataset[['cap-shape']] = impute.transform(dataset[['cap-shape']])

5 impute.fit(dataset[['cap-color']])

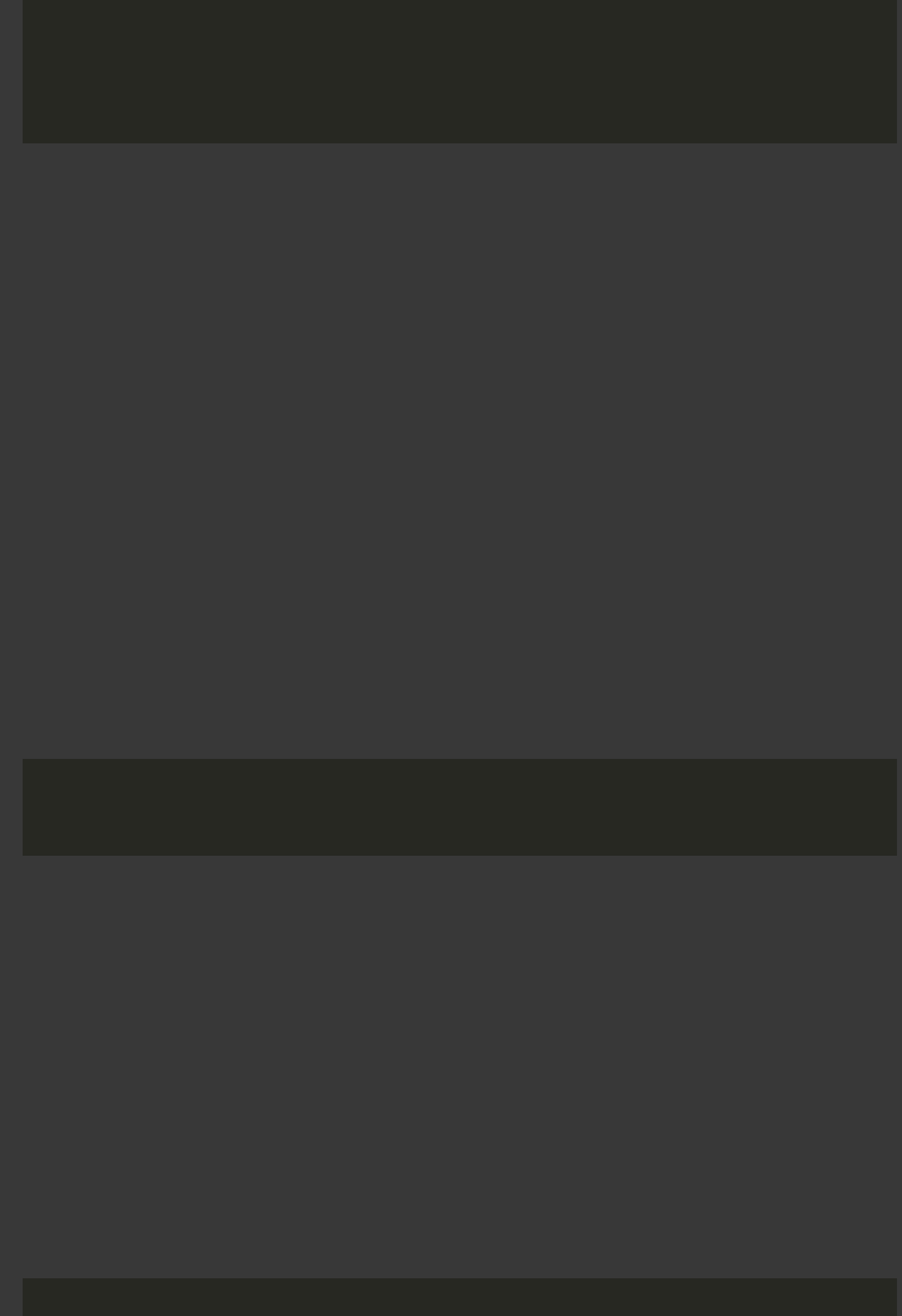
6 dataset[['cap-color']] = impute.transform(dataset[['cap-color']])

7 dataset.isnull().sum()

|  |  |
| --- | --- |
| Unnamed: 0 | 0 |
| class | 0 |
| cap-shape | 0 |
| cap-surface | 0 |
| cap-color | 0 |
| bruises | 0 |
| odor | 0 |
| stalk-shape | 0 |
| stalk-root | 0 |
| stalk-surface-above-ring | 0 |
| stalk-surface-below-ring | 0 |
| stalk-color-above-ring | 0 |
| stalk-color-below-ring | 0 |
| veil-type | 0 |
| veil-color | 0 |
| ring-number | 0 |
| ring-type | 0 |
| spore-print-color | 0 |
| population | 0 |
| habitat | 0 |
| dtype: int64 |  |

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 1/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



* from sklearn.preprocessing import LabelEncoder

2 encoder = LabelEncoder()

3 dataset['class'] = encoder.fit\_transform(dataset['class'])

4 dataset['bruises'] = encoder.fit\_transform(dataset['bruises'])

5 dataset.info()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| <class 'pandas.core.frame.DataFrame'> | | | |  |
| RangeIndex: 3124 entries, 0 to 3123 | | |  |  |
| Data | columns (total 20 columns): | |  |  |
| # | Column | Non-Null Count | | Dtype |
| --- | ------ | -------------- | | ----- |
| 0 | Unnamed: 0 | 3124 | non-null | int64 |
| 1 | class | 3124 | non-null | int64 |
| 2 | cap-shape | 3124 | non-null | float64 |
| 3 | cap-surface | 3124 | non-null | int64 |
| 4 | cap-color | 3124 | non-null | float64 |
| 5 | bruises | 3124 | non-null | int64 |
| 6 | odor | 3124 | non-null | int64 |
| 7 | stalk-shape | 3124 | non-null | int64 |
| 8 | stalk-root | 3124 | non-null | int64 |

* stalk-surface-above-ring 3124 non-null int64

10 stalk-surface-below-ring 3124 non-null int64

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 11 | stalk-color-above-ring | 3124 | non-null | int64 |
| 12 | stalk-color-below-ring | 3124 | non-null | int64 |
| 13 | veil-type | 3124 | non-null | int64 |
| 14 | veil-color | 3124 | non-null | int64 |
| 15 | ring-number | 3124 | non-null | int64 |
| 16 | ring-type | 3124 | non-null | int64 |
| 17 | spore-print-color | 3124 | non-null | int64 |
| 18 | population | 3124 | non-null | int64 |
| 19 | habitat | 3124 | non-null | int64 |

dtypes: float64(2), int64(18)

memory usage: 488.2 KB

* features\_list = dataset.columns.to\_list()

2 features\_list = features\_list[2:]

3 features\_list

['cap-shape',

'cap-surface',

'cap-color',

'bruises',

'odor',

'stalk-shape',

'stalk-root',

'stalk-surface-above-ring',

'stalk-surface-below-ring',

'stalk-color-above-ring',

'stalk-color-below-ring',

'veil-type',

'veil-color',

'ring-number',

'ring-type',

'spore-print-color',

'population',

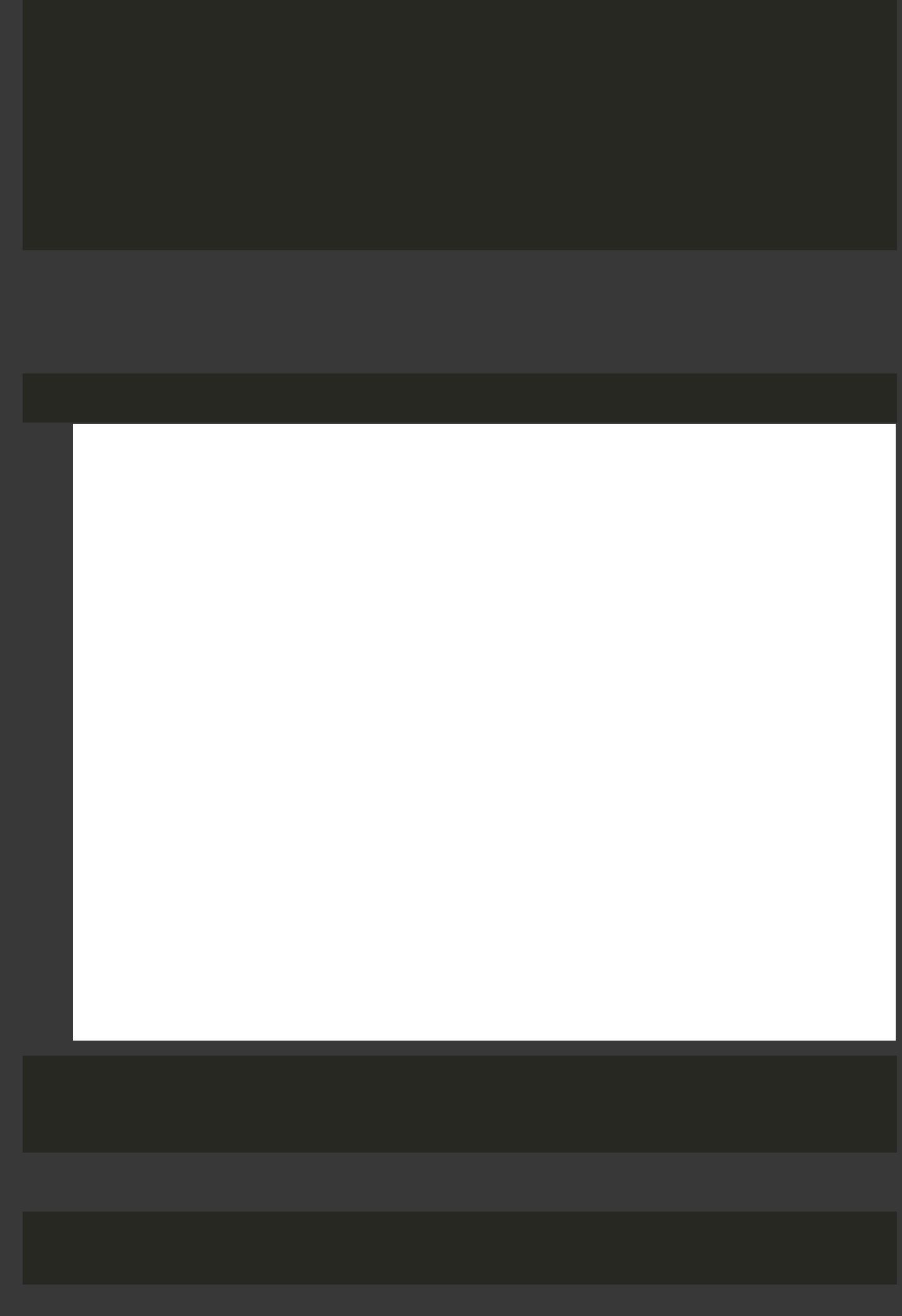
'habitat']

* label = ['class']

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 2/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory

2



* x = dataset[features\_list]

4 y = dataset[label]

5

6 from sklearn.model\_selection import train\_test\_split

7 x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, random\_state 8 print(x\_train.shape)

9 print(x\_test.shape)

1. print(y\_train.shape)
2. print(y\_test.shape)

(2499, 18) (625, 18) (2499, 1) (625, 1)

* dataset

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  | **stal** |
|  | **Unnamed:** | **class** | **cap-** | **cap-** | **cap-** | **bruises** | **odor** | **stalk- stalk- surfac** | | |
|  | **0** |  | **shape** | **surface** | **color** |  |  | **shape** | **root** | **abov** |
|  |  |  |  |  |  |  |  |  |  | **ri** |
|  |  |  |  |  |  |  |  |  |  |  |
| **0** | 0 | 1 | 5.0 | 2 | 4.00000 | 0 | 6 | 0 | 3 |  |
| **1** | 1 | 0 | 5.0 | 2 | 9.00000 | 0 | 0 | 0 | 2 |  |
| **2** | 2 | 0 | 0.0 | 2 | 8.00000 | 0 | 3 | 0 | 2 |  |
| **3** | 3 | 1 | 5.0 | 3 | 8.00000 | 0 | 6 | 0 | 3 |  |
| **4** | 4 | 0 | 5.0 | 2 | 3.00000 | 1 | 5 | 1 | 3 |  |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |  |
| **3119** | 3119 | 1 | 5.0 | 0 | 5.10077 | 1 | 1 | 0 | 1 |  |
| **3120** | 3120 | 0 | 2.0 | 0 | 3.00000 | 0 | 5 | 1 | 1 |  |
| **3121** | 3121 | 0 | 2.0 | 3 | 3.00000 | 0 | 5 | 1 | 1 |  |
| **3122** | 3122 | 1 | 5.0 | 0 | 3.00000 | 1 | 2 | 0 | 1 |  |
| **3123** | 3123 | 0 | 5.0 | 3 | 2.00000 | 0 | 5 | 1 | 1 |  |

3124 rows × 20 columns

* from sklearn.naive\_bayes import GaussianNB

2 gnb = GaussianNB()

3 gnb.fit(x\_train, y\_train.values.ravel())

GaussianNB(priors=None, var\_smoothing=1e-09)

* print("Training accuracy of the model is {:.2f}".format(gnb.score(x\_train, y\_train)))

2 print("Testing accuracy of the model is {:.2f}".format(gnb.score(x\_test, y\_test)))

Training accuracy of the model is 0.75

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 3/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory

Testing accuracy of the model is 0.73



* predictions = gnb.predict(x\_test)

2 print(predictions)

[0101100011101100111110000000000001001 0000101011000101010010010010010001100 1000100100001000010011111010000110000 0000101000000010000100000010000010000 1100100011110111100100101000011010001 0100001010011100100000011100111100100 0001100111100100101101000000000000000 1000110000011011100011001001100110001 1000001001001001110100000001101001010 0110100110100101101110001100100000000 0000011000001000010010110111100000100 0100010011000010010111000000010011000 0110101110000101111000000110001100000 0100001110010010011110010010000000100 1100100001010010101011111010110011001 0000000100110010001001101100000001010 011011100110110011000011101011000]

* from sklearn.metrics import confusion\_matrix

2 mat = confusion\_matrix(predictions, y\_test)

3 print(mat)

[[391 0]

[170 64]]

* from seaborn import heatmap
* heatmap(mat, cmap="Pastel1\_r", xticklabels=['class\_0' ,'class\_1' ,'class\_2'], ytickla <matplotlib.axes.\_subplots.AxesSubplot at 0x7f8b14525d50>
* from sklearn.ensemble import RandomForestClassifier

2 rfc = RandomForestClassifier(n\_estimators = 50)

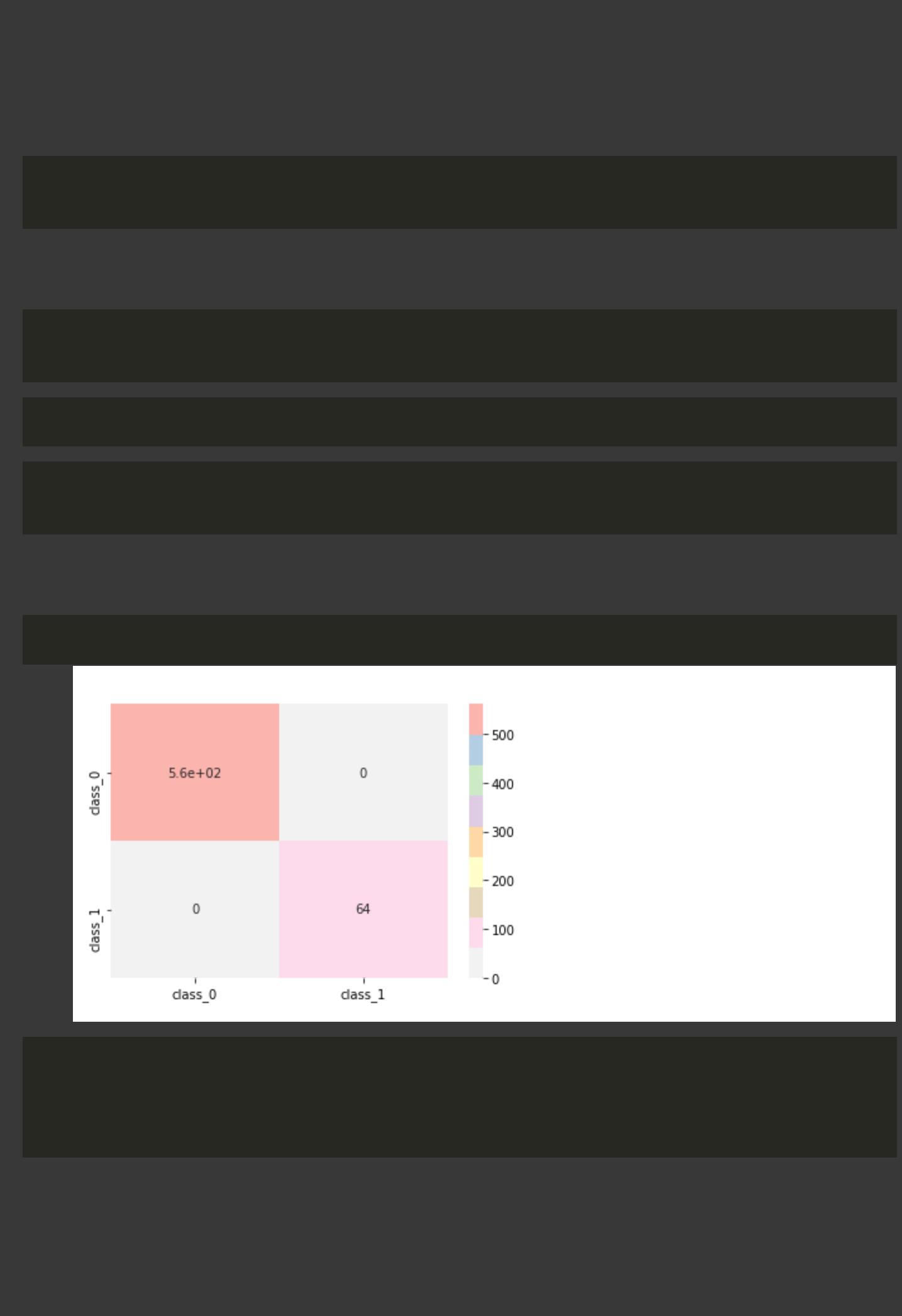
3 rfc.fit(x\_train, y\_train.values.ravel())

RandomForestClassifier(bootstrap=True, ccp\_alpha=0.0, class\_weight=None, criterion='gini', max\_depth=None, max\_features='auto',

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 4/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory

max\_leaf\_nodes=None, max\_samples=None,



min\_impurity\_decrease=0.0, min\_impurity\_split=None,

min\_samples\_leaf=1, min\_samples\_split=2,

min\_weight\_fraction\_leaf=0.0, n\_estimators=50,

n\_jobs=None, oob\_score=False, random\_state=None,

verbose=0, warm\_start=False)

* print("Training accuracy {:.2f}".format(rfc.score(x\_train, y\_train)))

2 print("Testing accuracy {:.2f}".format(rfc.score(x\_test, y\_test)))

Training accuracy 1.00

Testing accuracy 1.00

* k = rfc.score(x\_train, y\_train)

2 l = rfc.score(x\_test, y\_test)

* predictions = rfc.predict(x\_test)
* mat = confusion\_matrix(predictions, y\_test)

2 print(mat)

[[561 0]

* + 0 64]]

1 heatmap(mat, cmap="Pastel1\_r", xticklabels=['class\_0' ,'class\_1'], yticklabels=['clas <matplotlib.axes.\_subplots.AxesSubplot at 0x7f8b1309d490>

* from sklearn.neural\_network import MLPClassifier
* nnc = MLPClassifier(hidden\_layer\_sizes=(7), activation="relu", max\_iter=10000)

3

4 nnc.fit(x\_train, y\_train.values.ravel())

MLPClassifier(activation='relu', alpha=0.0001, batch\_size='auto', beta\_1=0.9,

beta\_2=0.999, early\_stopping=False, epsilon=1e-08,

hidden\_layer\_sizes=7, learning\_rate='constant',

learning\_rate\_init=0.001, max\_fun=15000, max\_iter=10000,

momentum=0.9, n\_iter\_no\_change=10, nesterovs\_momentum=True,

power\_t=0.5, random\_state=None, shuffle=True, solver='adam',

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 5/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory

tol=0.0001, validation\_fraction=0.1, verbose=False, warm\_start=False)



* print("Training accuracy {:.2f}".format(nnc.score(x\_train, y\_train)))

2 print("Testing accuracy {:.2f}".format(nnc.score(x\_test, y\_test)))

Training accuracy 1.00

Testing accuracy 1.00

* g = nnc.score(x\_train, y\_train)

2 h = nnc.score(x\_test, y\_test)

* predictions = nnc.predict(x\_test)

2 print(predictions)

[0000000010100100000100000000000000000 0000100000000001000000000000000001000 0000000000000000000000000000000100000 0000001000000000000000000010000000000 0000000000000001100000100000010000000 0100001000010000100000001000000000000 0000000000000000100000000000000000000 0000010000000000000000000000000100000 1000001000000001110000000000000000010 0010100000000001000000000100100000000 0000000000000000000010000010000000000 0100000000000000010010000000000000000 0100000000000000101000000000000000000 0100001010010010000000000000000000000 1000100000000000000000000000110010000 0000000000000010001000101100000000000 001011100100000000000000100010000]

* mat = confusion\_matrix(predictions, y\_test)

2 print(mat)

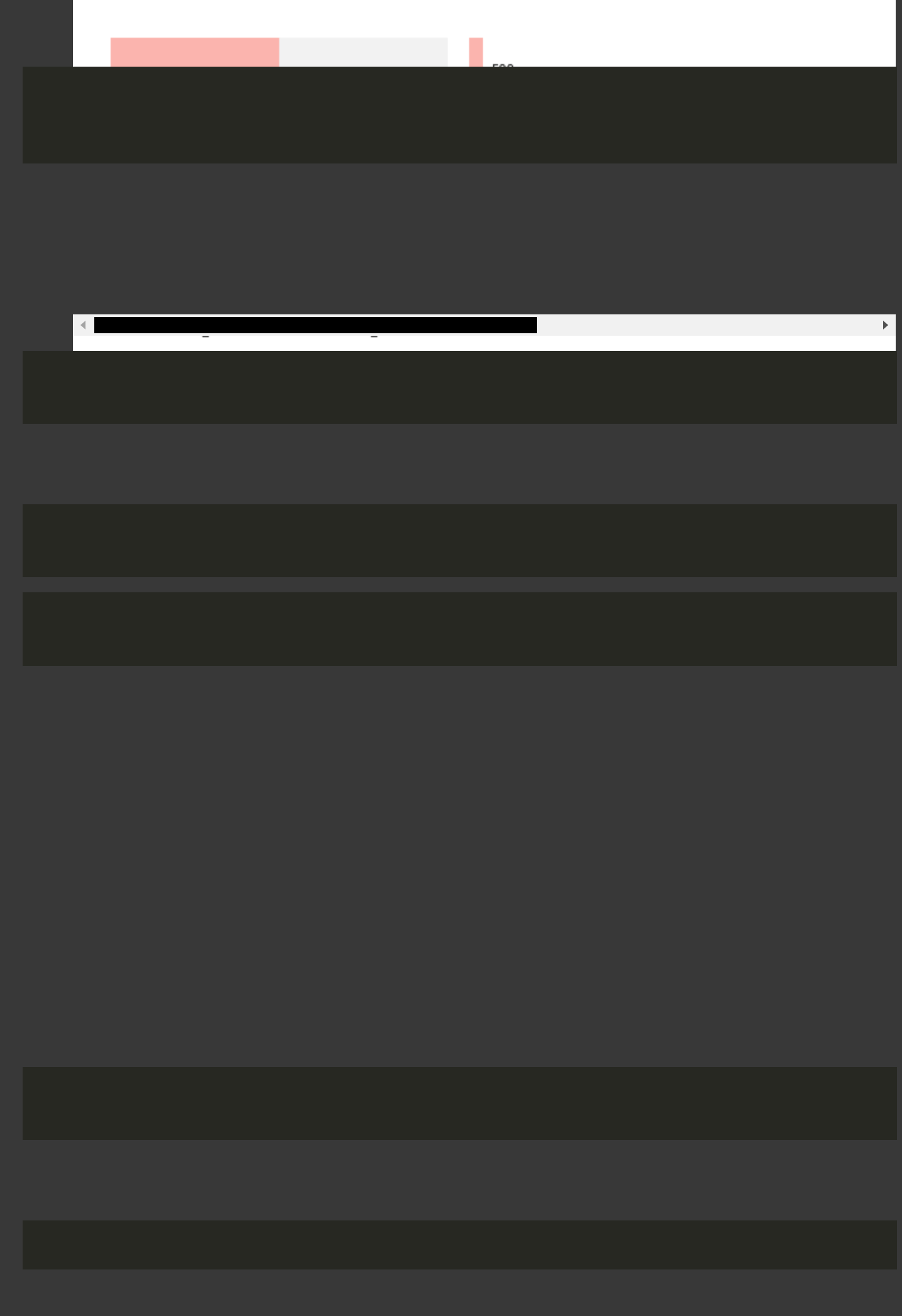
[[561 1]

* 0 63]]

1 heatmap(mat, cmap="Pastel1\_r", xticklabels=['class\_0' ,'class\_1'], yticklabels=['clas

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 6/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



<matplotlib.axes.\_subplots.AxesSubplot at 0x7f8b12fa4090>

* from sklearn.svm import LinearSVC
* svc = LinearSVC(random\_state=0, tol=1e-5)

3 svc.fit(x\_train, y\_train.values.ravel())

/usr/local/lib/python3.7/dist-packages/sklearn/svm/\_base.py:947: ConvergenceWarning:

"the number of iterations.", ConvergenceWarning)

LinearSVC(C=1.0, class\_weight=None, dual=True, fit\_intercept=True, intercept\_scaling=1, loss='squared\_hinge', max\_iter=1000, multi\_class='ovr', penalty='l2', random\_state=0, tol=1e-05, verbose=0)

* print("Training accuracy {:.2f}".format(svc.score(x\_train, y\_train)))

2 print("Testing accuracy {:.2f}".format(svc.score(x\_test, y\_test)))

Training accuracy 1.00

Testing accuracy 1.00

* c = svc.score(x\_train, y\_train)

2 d = svc.score(x\_test, y\_test)

* predictions = svc.predict(x\_test)

2 print(predictions)

[0000000010100100000100000000000000000 0000100000000001000000000000000001000 0000000000000000000000000000000100000 0000001000000000000000000010000000000 0000000000000001100000100000010000000 0100001000010000100000001000001000000 0000000000000000100000000000000000000 0000010000000000000000000000000100000 1000001000000001110000000000000000010 0010100000000001000000000100100000000 0000000000000000000010000010000000000 0100000000000000010010000000000000000 0100000000000000101000000000000000000 0100001010010010000000000000000000000 1000100000000000000000000000110010000 0000000000000010001000101100000000000 001011100100000000000000100010000]

* mat = confusion\_matrix(predictions, y\_test)

2 print(mat)

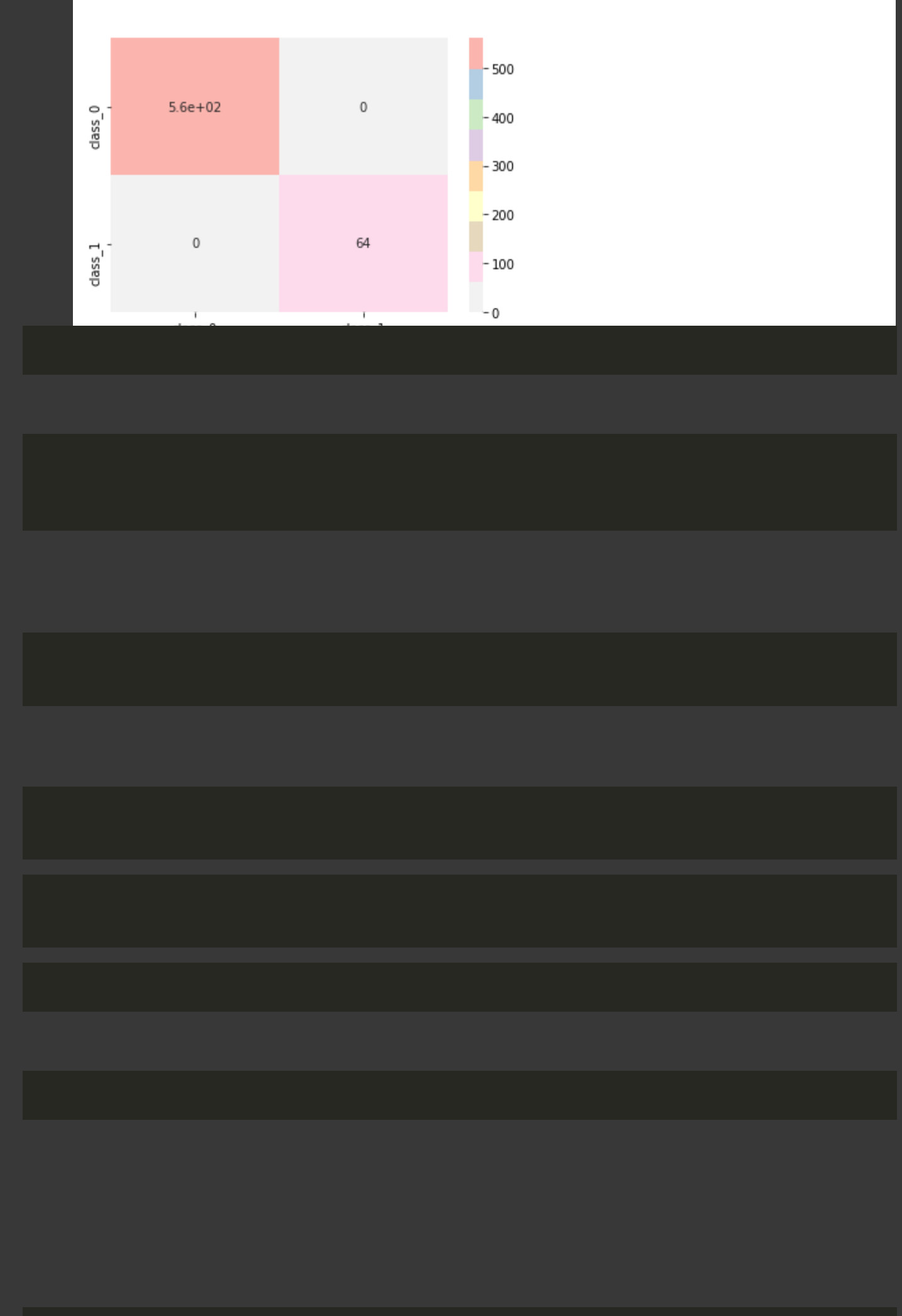
[[561 0]

* 0 64]]

1 heatmap(mat, cmap="Pastel1\_r", xticklabels=['class\_0' ,'class\_1'], yticklabels=['clas

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 7/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



<matplotlib.axes.\_subplots.AxesSubplot at 0x7f8b0ae80150>

* dataset.shape (3124, 20)
* from sklearn.neighbors import KNeighborsClassifier

2 knn = KNeighborsClassifier(n\_neighbors=4)

3 knn.fit(x\_train, y\_train.values.ravel())

KNeighborsClassifier(algorithm='auto', leaf\_size=30, metric='minkowski', metric\_params=None, n\_jobs=None, n\_neighbors=4, p=2, weights='uniform')

* print("Training accuracy is {:.2f}".format(knn.score(x\_train, y\_train)))

2 print("Testing accuracy is {:.2f} ".format(knn.score(x\_test, y\_test)))

Training accuracy is 1.00

Testing accuracy is 1.00

* from sklearn.preprocessing import StandardScaler

2 scaler = StandardScaler()

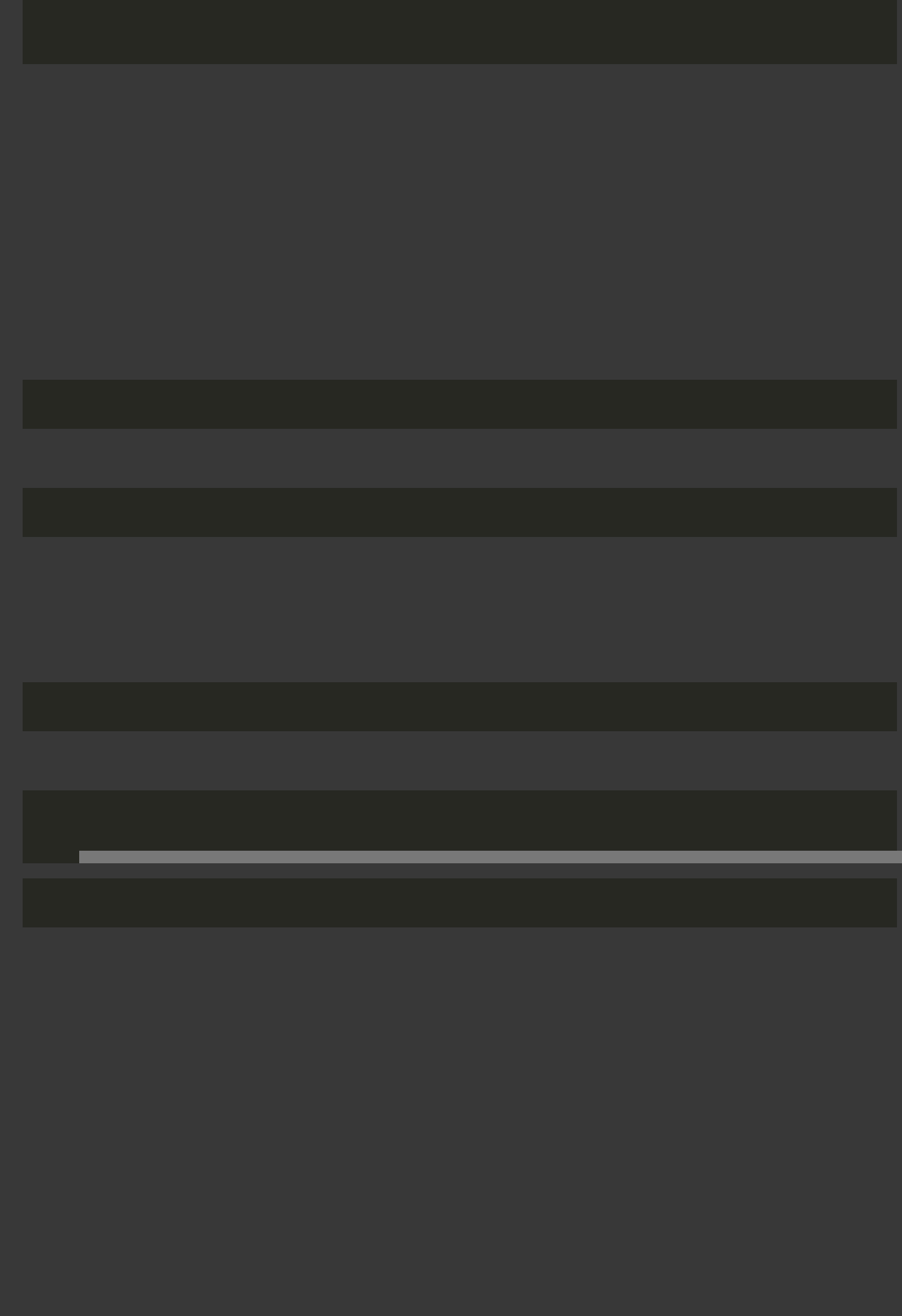
* from sklearn.decomposition import PCA
* pca = PCA(n\_components = 20)
* dataset.shape (3124, 20)
* dataset.keys()

Index(['Unnamed: 0', 'class', 'cap-shape', 'cap-surface', 'cap-color', 'bruises', 'odor', 'stalk-shape', 'stalk-root', 'stalk-surface-above-ring', 'stalk-surface-below-ring', 'stalk-color-above-ring', 'stalk-color-below-ring', 'veil-type', 'veil-color', 'ring-number', 'ring-type', 'spore-print-color', 'population', 'habitat'],

dtype='object')

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 8/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



* principal\_components = pca.fit\_transform(dataset)

2 print(principal\_components)

[[-1.56149644e+03 1.75644912e+00 -4.69402624e+00 ... -6.36860385e-17 -8.67392349e-17 -4.45828644e-19]

|  |  |  |  |
| --- | --- | --- | --- |
| [-1.56050630e+03 | -2.53584919e+00 | 3.38009823e-01 ... | 1.22025085e-16 |
| 1.94885401e-17 | -1.58396936e-19] |  |  |
| [-1.55950559e+03 | -1.27472557e+00 | 6.04136973e-01 ... | 2.11140764e-17 |

4.21433226e-17 1.39273540e-19]

...

* 1.55950308e+03 -1.89496063e+00 5.23388261e-01 ... -2.69003194e-19 5.39515930e-18 8.71659307e-21]
* 1.56050655e+03 -1.37099425e+00 -3.49445642e-01 ... 2.78907261e-18 3.11425279e-18 1.30037121e-20]
  + 1.56150530e+03 -9.26327991e-01 -7.58207404e-01 ... 2.04400584e-18 2.20304899e-18 1.68392133e-20]]

1 principal\_components.shape (3124, 20)

* pca.explained\_variance\_ratio\_

array([9.99971075e-01, 7.42186181e-06, 6.33020968e-06, 3.30342007e-06, 2.65744282e-06, 1.99296105e-06, 1.71644894e-06, 1.61560938e-06,

1.44765222e-06, 8.66060988e-07, 4.99185310e-07, 4.25596854e-07,

2.91286644e-07, 2.09925344e-07, 9.69637532e-08, 3.58417965e-08,

1.47435763e-08, 2.56717982e-39, 1.43737303e-39, 1.00444054e-43])

* sum(pca.explained\_variance\_ratio\_)

0.9999999999999997

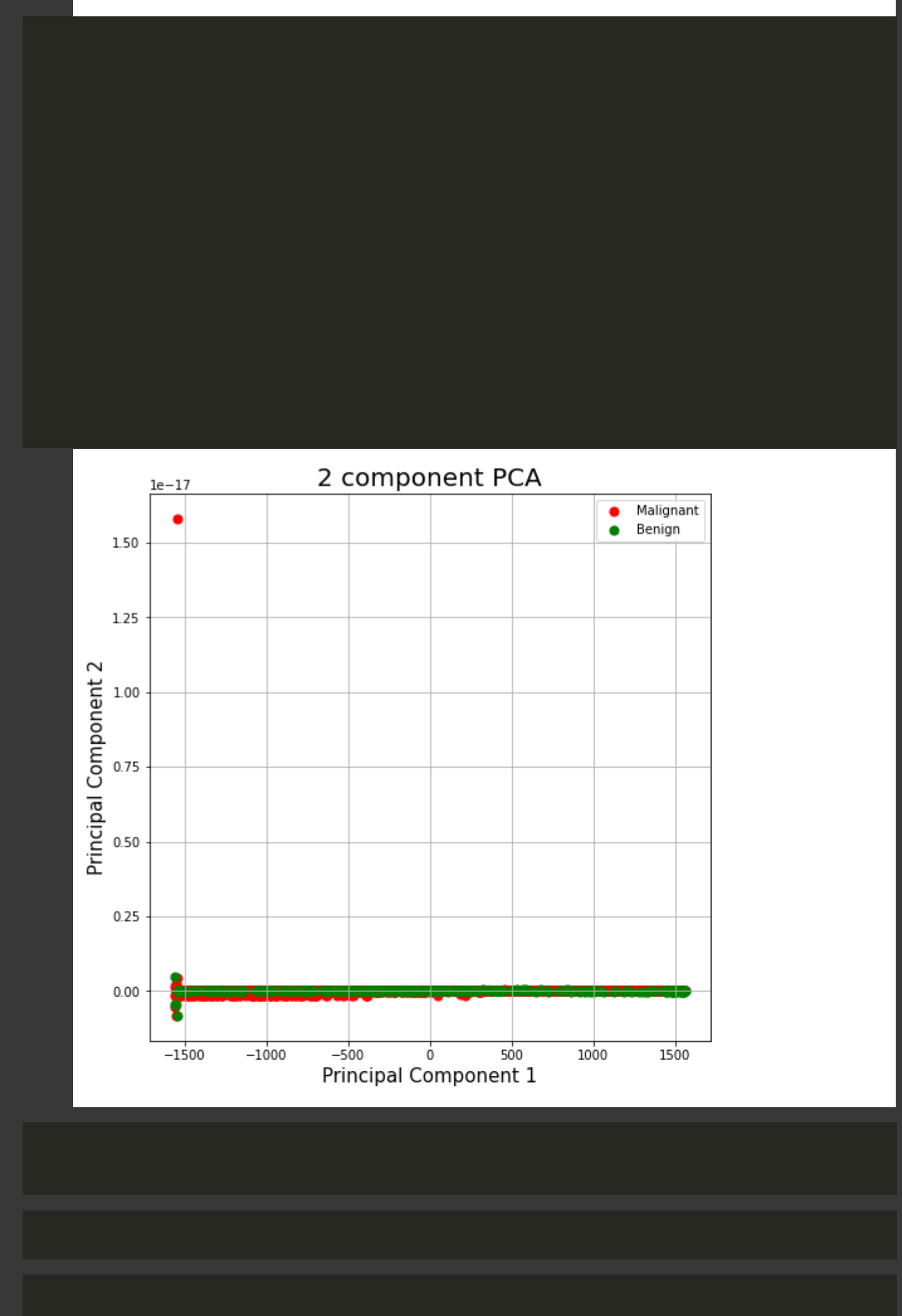
* principal\_df = pd.DataFrame(data=principal\_components, columns=["a", "b", "c", "d", "

2 main\_df=pd.concat([principal\_df, dataset[["class"]]], axis=1)

* main\_df.head()

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 9/17

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 9/9/21, 9:57 PM | | |  |  |  | Lab 08 - CSE422.ipynb - Colaboratory | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1 | |  | **a** | **b** |  | **c** | **d** | **e** | **f** | **g** |  |  |
| import matplotlib.pyplot as plt | | | |  |  |  |  |  |  |  |
| 2 | | **0** | -1561.496439 | 1.756449 | -4.694026 | | 0.159432 | 3.021411 | -0.525178 | 0.762823 | 0.68184 |  |
| 3 | |  |
| fig = plt.figure(figsize = (8,8)) | | | |  |  |  |  |  |  |  |
| 4 | | ax = fig.add\_subplot(1,1, 1) | | | |  |  |  |  |  |  |  |
| 5 | | ax.set xlabel('Principal Component 1', fontsize = 15) | | | | | | | 0.748402 | -0.029880 | -0.27399 |  |
|  |  | **1** | -1560.506304 -2.535849 | | 0.338010 | | -2.653949 | -2.025029 |  |
| 6 | | ax.set\_ylabel('Principal Component 2', fontsize = 15) | | | | | | |  |  |  |  |
| 7 | | ax.set\_title('2 component PCA', fontsize = 20) | | | | | | | -0.699287 | -0.078174 | 0.49937 |  |
| 8 | | **2** | -1559.505594 | -1.274726 | 0.604137 | | 2.928215 | -0.357481 |  |
| targets = [0, 1] | |  |  |  |  |  |  |  |  |  |



* colors = ['r', 'g']

10 for**3** -target,1558.503231color 0in.004963zip(targets,colors):-1.812748-1.008242 3.580844 -2.267762 0.384989 1.38629

1. indicesToKeep = main\_df['class'] == target
2. ax.scatter(main\_df.loc[indicesToKeep, 'a']

|  |  |  |  |
| --- | --- | --- | --- |
| 13 | **4** | -1557.494233 5.440754 -2.884493 -0.955288 -1.240247 1.166657 -1.384592 1.15083 |  |
|  | , main\_df.loc[indicesToKeep, 't'] |  |
| 14 |  | , c = color |  |
| 15 |  | , s = 50) |  |

1. ax.legend(["Malignant", "Benign"])
2. ax.grid()

* X = main\_df.drop("class" , axis=1)

2 y = main\_df["class"]

* x\_train, x\_test, y\_train, y\_test = train\_test\_split(X , y , test\_size=0.2, random\_sta
* knn\_2=KNeighborsClassifier(n\_neighbors=4)

( )

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 10/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory

* knn\_2.fit(x\_train, y\_train)

KNeighborsClassifier(algorithm='auto', leaf\_size=30, metric='minkowski',



metric\_params=None, n\_jobs=None, n\_neighbors=4, p=2, weights='uniform')

* print("Training accuracy {:.2f}".format(knn\_2.score(x\_train, y\_train)) )
* print("Testing accuracy {:.2f} ".format(knn\_2.score(x\_test, y\_test)) )

Training accuracy 0.92 Testing accuracy 0.91

* rfc = RandomForestClassifier(n\_estimators=50)

2 rfc.fit(x\_train, y\_train)

RandomForestClassifier(bootstrap=True, ccp\_alpha=0.0, class\_weight=None,

criterion='gini', max\_depth=None, max\_features='auto',

max\_leaf\_nodes=None, max\_samples=None,

min\_impurity\_decrease=0.0, min\_impurity\_split=None,

min\_samples\_leaf=1, min\_samples\_split=2,

min\_weight\_fraction\_leaf=0.0, n\_estimators=50,

n\_jobs=None, oob\_score=False, random\_state=None,

verbose=0, warm\_start=False)

* print("Training accuracy {:.2f}".format(rfc.score(x\_train, y\_train)))

2 print("Testing accuracy {:.2f}".format(rfc.score(x\_test, y\_test)))

Training accuracy 1.00

Testing accuracy 1.00

* i = rfc.score(x\_train, y\_train)

2 j = rfc.score(x\_test, y\_test)

* predictions = rfc.predict(x\_test)
* mat=confusion\_matrix(predictions, y\_test)

2 print(mat)

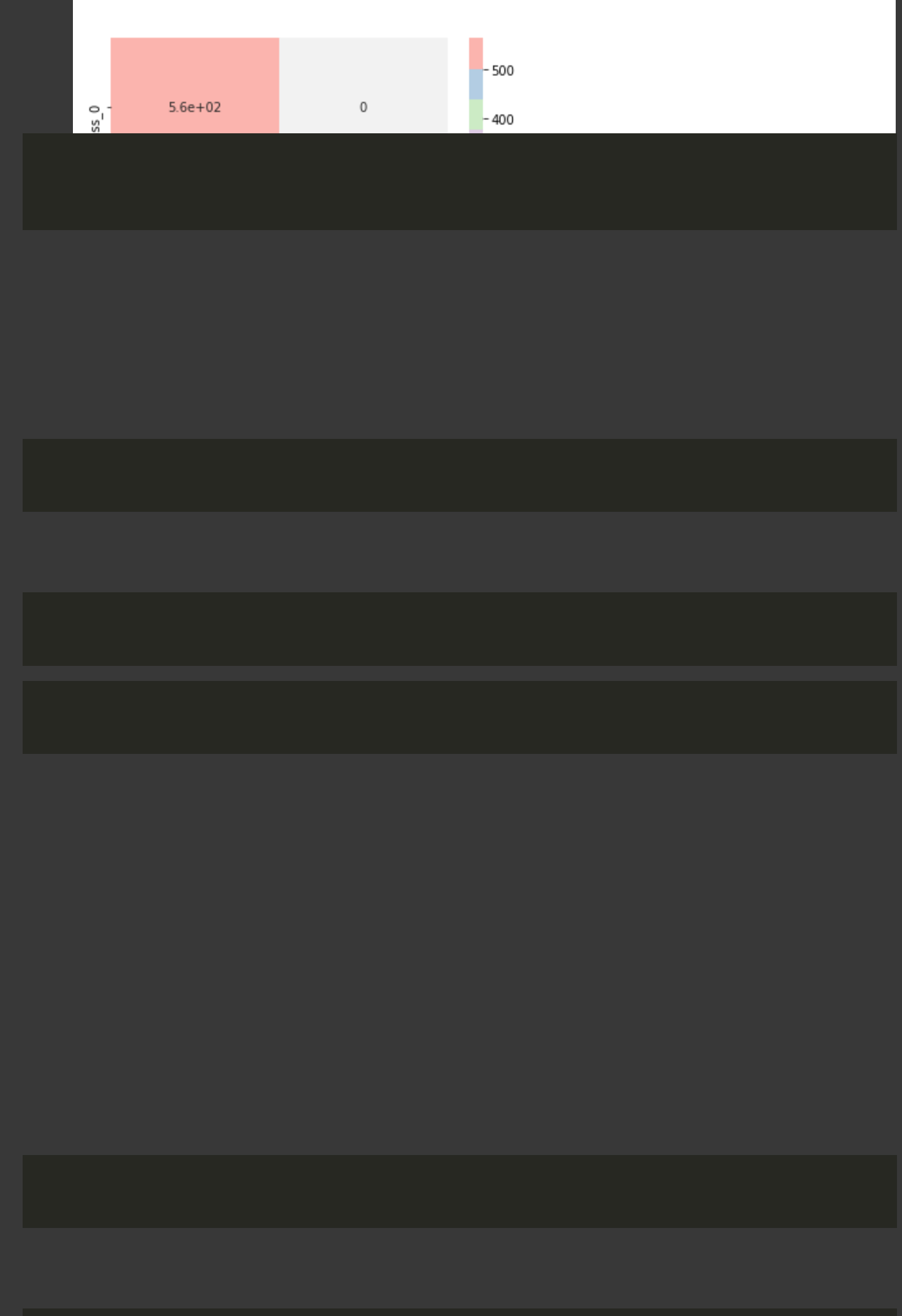
[[565 0]

* 0 60]]

1 heatmap(mat , cmap="Pastel1\_r", xticklabels=['class\_0' ,'class\_1'], yticklabels=['cla

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 11/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



<matplotlib.axes.\_subplots.AxesSubplot at 0x7f8b079f25d0>

* nnc=MLPClassifier(hidden\_layer\_sizes=(7), activation="relu", max\_iter=10000)

2

3 nnc.fit(x\_train, y\_train)

MLPClassifier(activation='relu', alpha=0.0001, batch\_size='auto', beta\_1=0.9,

beta\_2=0.999, early\_stopping=False, epsilon=1e-08,

hidden\_layer\_sizes=7, learning\_rate='constant',

learning\_rate\_init=0.001, max\_fun=15000, max\_iter=10000,

momentum=0.9, n\_iter\_no\_change=10, nesterovs\_momentum=True,

power\_t=0.5, random\_state=None, shuffle=True, solver='adam',

tol=0.0001, validation\_fraction=0.1, verbose=False,

warm\_start=False)

* print("Training accuracy {:.2f}".format(nnc.score(x\_train, y\_train)))

2 print("Testing accuracy {:.2f}".format(nnc.score(x\_test, y\_test)))

Training accuracy 1.00

Testing accuracy 1.00

* e = nnc.score(x\_train, y\_train)

2 f = nnc.score(x\_test, y\_test)

* predictions = nnc.predict(x\_test)

2 print(predictions)

[0000000000000001000000010000000000001 0000000010101000000000000000000000000 0000000000000100000000000100000000001 0000000000000000000001000100001000010 0001000100000000001000000001000000100 0000000001000000000000000000110000000 0001000000001000000100100100000000000 0000000000001100000000100000010001010 0010000001000010100100000100000000000 0000000000100000000000000000000000000 0010001000000000000000100000000000000 0000000000000000000000000000001000000 0000010000010000000000000000000000000 0000000000100000000001000010000000000 1000000000000000000000000001001000000 0100000000001000000001000000000000000 100100000000000000000100000101001]

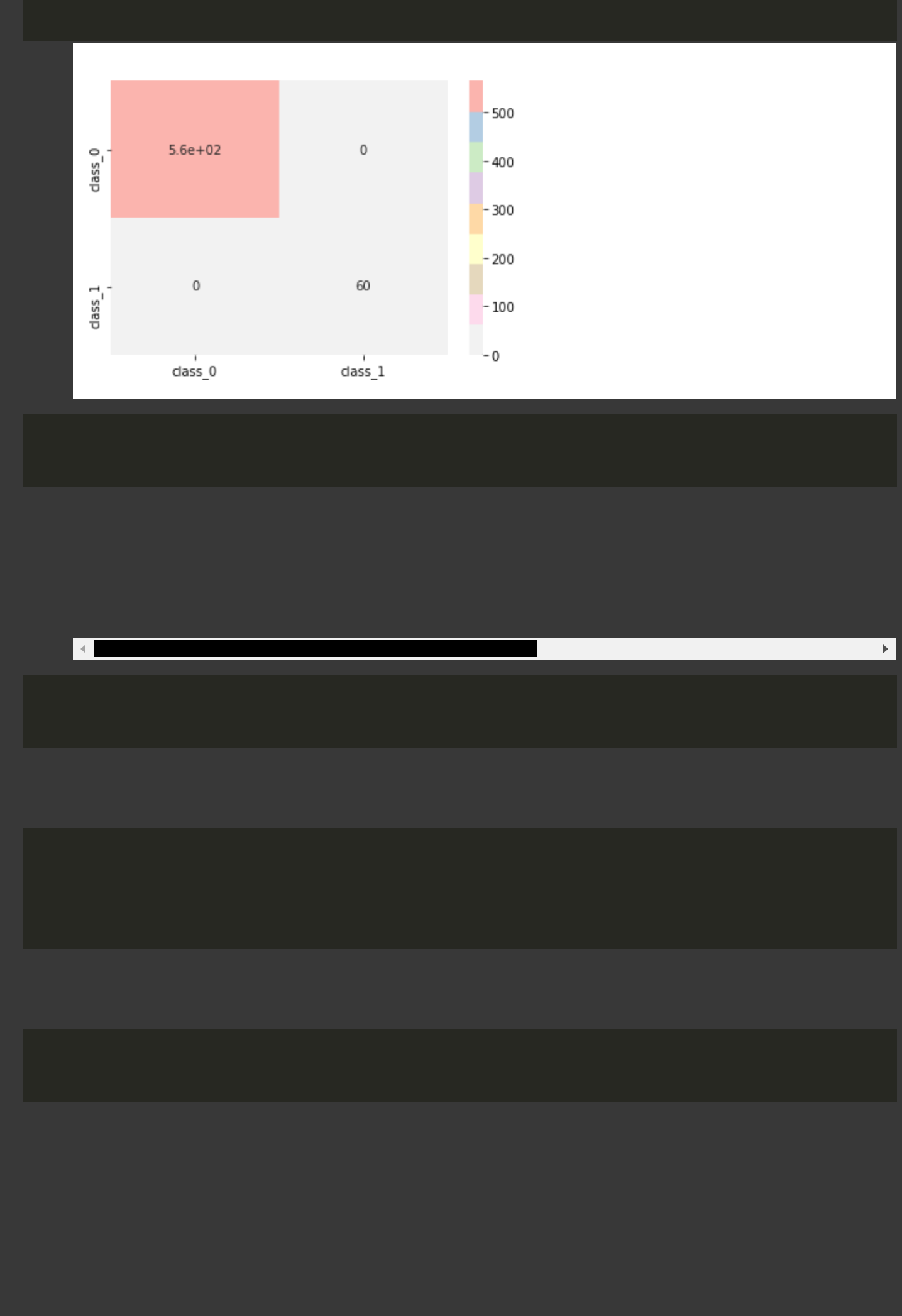
* mat=confusion\_matrix(predictions, y\_test)

2 print(mat)

|  |  |
| --- | --- |
| [[565 | 0] |
| [ 0 | 60]] |

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 12/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



* heatmap(mat , cmap="Pastel1\_r", xticklabels=['class\_0' ,'class\_1'], yticklabels=['cla <matplotlib.axes.\_subplots.AxesSubplot at 0x7f8b0790f290>
* svc = LinearSVC(random\_state=0, tol=1e-5)

2 svc.fit(x\_train, y\_train)

/usr/local/lib/python3.7/dist-packages/sklearn/svm/\_base.py:947: ConvergenceWarning:

"the number of iterations.", ConvergenceWarning)

LinearSVC(C=1.0, class\_weight=None, dual=True, fit\_intercept=True, intercept\_scaling=1, loss='squared\_hinge', max\_iter=1000, multi\_class='ovr', penalty='l2', random\_state=0, tol=1e-05, verbose=0)

* print("Training accuracy {:.2f}".format(svc.score(x\_train, y\_train)))

2 print("Testing accuracy {:.2f}".format(svc.score(x\_test, y\_test)))

Training accuracy 0.97

Testing accuracy 0.98

* a = svc.score(x\_train, y\_train)

2 b = svc.score(x\_test, y\_test)

3 print ("{:.2f}".format(a))

4 print ("{:.2f}".format(b))

0.97

0.98

* predictions = svc.predict(x\_test)

2 print(predictions)

[0000000000000001000000010000000000000 0000000000101000000000000000000000000 0000000000000100000000000100000000000 0000000000000000000001000100000000010 0001000100000000001000000001000000100 0000000001000000000000000000110000000 0001000000001000000100000100000000000 0000000000001100000000000000010001010 0010000001000000100100000000000000000

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 13/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory

0000000000100000000000000000000000000



0010001000000000000000100000000000000

0000000000000000000000000000000000000

0000010000010000000000000000000000000

0000000000100000000000000010000000000

1000000000000000000000000001001000000

0100000000001000000001000000000000000

100100000000000000000000000101001]

* mat=confusion\_matrix(predictions, y\_test)

2 print(mat)

[[565 11]

* + 0 49]]

1 heatmap(mat , cmap="Pastel1\_r", xticklabels=['class\_0' ,'class\_1' ], yticklabels=['cl <matplotlib.axes.\_subplots.AxesSubplot at 0x7f8b08d4aad0>

* n\_groups = 2
* means\_frank = (c,a)

3 means\_guido = (d,b)

4

5 fig, ax = plt.subplots()

6 index = np.arange(n\_groups)

7 bar\_width = 0.3

8 opacity = 1

9

1. rects1 = plt.bar(index, means\_frank, bar\_width,
2. alpha=opacity,
3. color='r')

13

1. rects2 = plt.bar(index + bar\_width, means\_guido, bar\_width,
2. alpha=opacity,
3. color='b')

17

1. plt.title(' Support Vector Machine')
2. plt.xticks(index + bar\_width, ('Pre\_PCA', 'Post\_PCA'))
3. plt.legend()

21

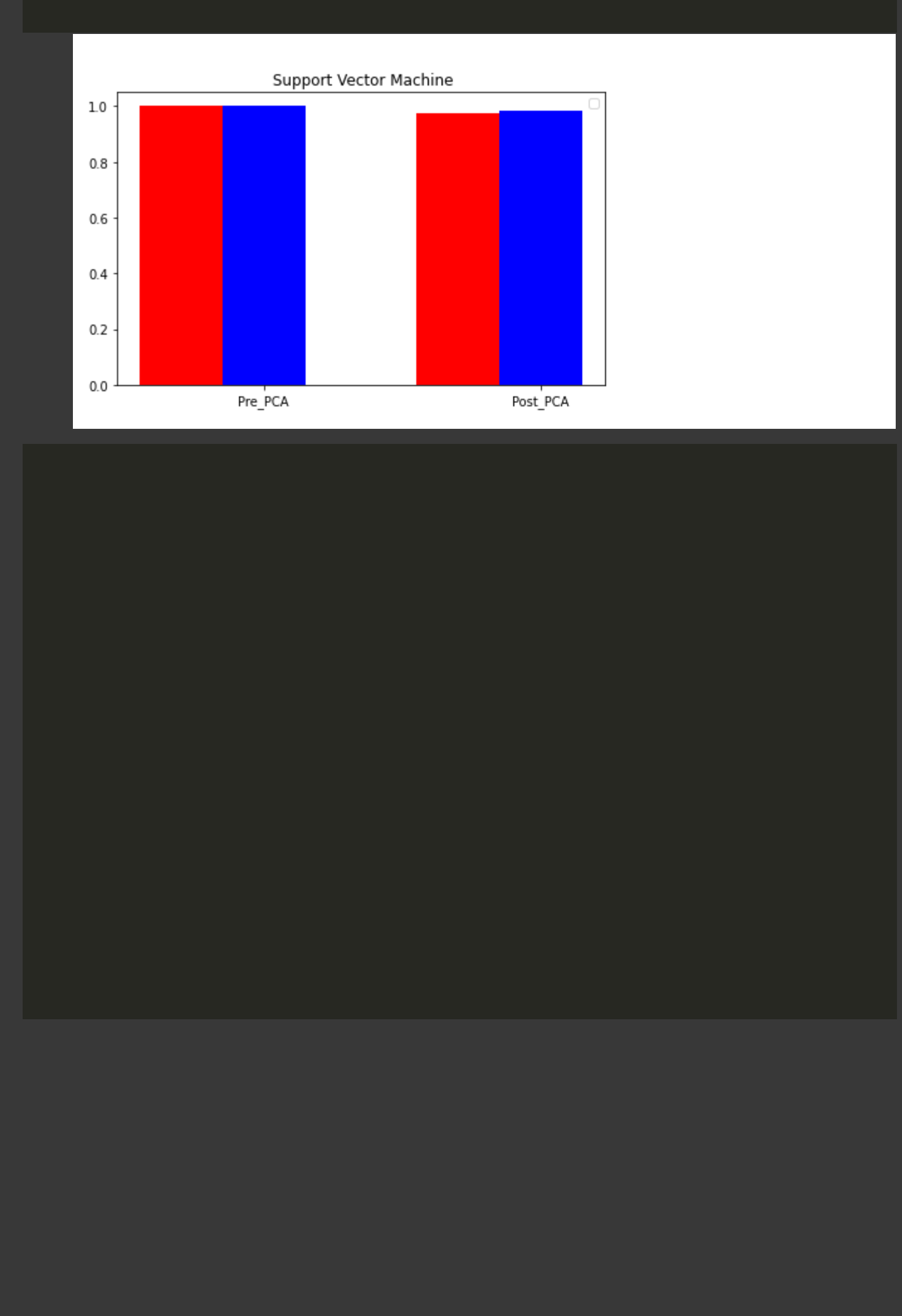
1. plt.tight\_layout()

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 14/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory

1. plt.show()

No handles with labels found to put in legend.



* n\_groups = 2
* means\_frank = (g,e)

3 means\_guido = (h,f)

4

5 fig, ax = plt.subplots()

6 index = np.arange(n\_groups)

7 bar\_width = 0.3

8 opacity = 1

9

1. rects1 = plt.bar(index, means\_frank, bar\_width,
2. alpha=opacity,
3. color='r')

13

1. rects2 = plt.bar(index + bar\_width, means\_guido, bar\_width,
2. alpha=opacity,
3. color='b')

17

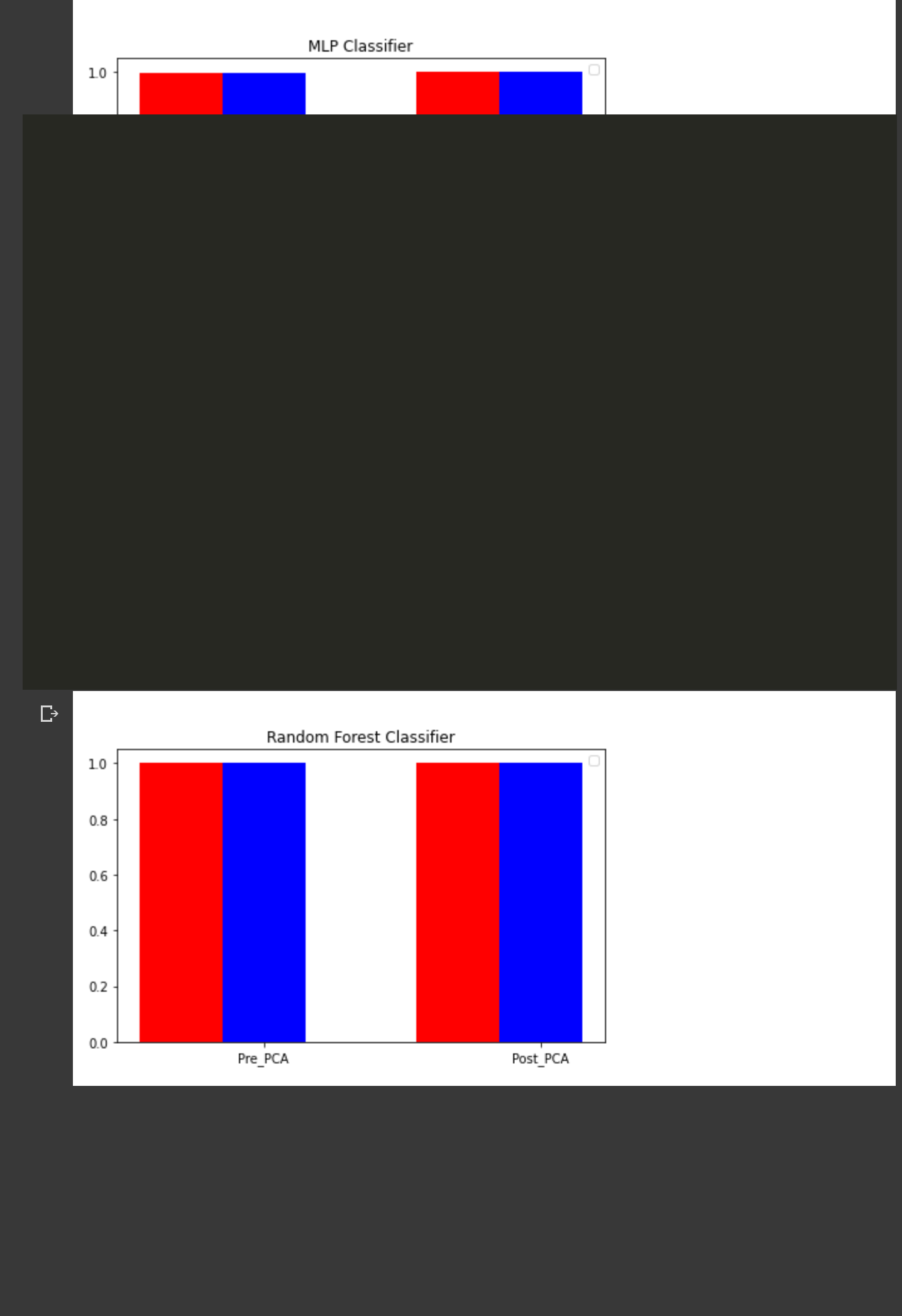
1. plt.title('MLP Classifier')
2. plt.xticks(index + bar\_width, ('Pre\_PCA', 'Post\_PCA'))
3. plt.legend()

21

1. plt.tight\_layout()
2. plt.show()

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 15/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



No handles with labels found to put in legend.

* n\_groups = 2
* means\_frank = (k,i)

3 means\_guido = (l,j)

4

5 fig, ax = plt.subplots()

6 index = np.arange(n\_groups)

7 bar\_width = 0.3

8 opacity = 1

9

1. rects1 = plt.bar(index, means\_frank, bar\_width,
2. alpha=opacity,
3. color='r')

13

1. rects2 = plt.bar(index + bar\_width, means\_guido, bar\_width,
2. alpha=opacity,
3. color='b')

17

1. plt.title('Random Forest Classifier')
2. plt.xticks(index + bar\_width, ('Pre\_PCA', 'Post\_PCA'))
3. plt.legend()

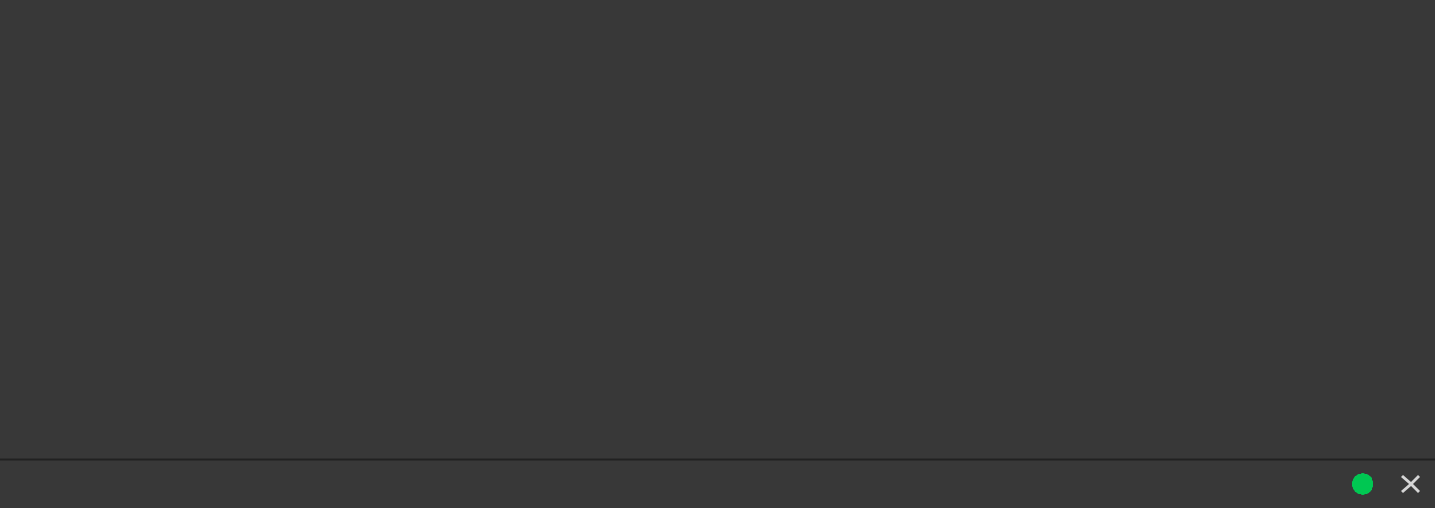
21

1. plt.tight\_layout()
2. plt.show()

No handles with labels found to put in legend.

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 16/17

9/9/21, 9:57 PM Lab 08 - CSE422.ipynb - Colaboratory



* 0s completed at 9:57 PM

https://colab.research.google.com/drive/1yXndEKZHi9fQ\_fkUa-5vTKjujxG\_vCdB#scrollTo=-XR3fQGBRtEK&printMode=true 17/17