

第 5 组: 沈亦韬 王鹏潇 张笑康 喻剑锟 车浩杰

BREAST CANCER DIAGNOSIS

DATA ANALYSIS

DATA ANALYSIS

IN:

- ▶ df = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/wdbc.data', names=column_names)
- ▶ df.info()

OUT:

- ▶ <class 'pandas.core.frame.DataFrame'>
- ▶ RangeIndex: 569 entries, 0 to 568
- ▶ Data columns (total 32 columns):
 - ▶ id 569 non-null int64
 - ▶ diagnosis 569 non-null object
 - ▶ radius_mean 569 non-null float64
 - ▶ texture_mean 569 non-null float64
 - ▶ perimeter_mean 569 non-null float64
 - ▶ area_mean 569 non-null float64
 - ▶ smoothness_mean 569 non-null float64
 - ▶ compactness_mean 569 non-null float64
 - ▶ concavity_mean 569 non-null float64
 - ▶ concave_points_mean 569 non-null float64
 - ▶ symmetry_mean 569 non-null float64
 - ▶ fractal_dimension_mean 569 non-null float64
 - ▶ radius_se 569 non-null float64

... (TOTAL 32 ATTRIBUTE)

DTYPES: FLOAT64(30), INT64(1), OBJECT(1)

DATA ANALYSIS

- ▶ 属性信息:
- ▶ 1 身份证号码
- ▶ 2 诊断 (M =恶性, B =良性)
- ▶ 3-32 为每个细胞核计算十个实值特征:
 - ▶ a) 半径 (从中心到周边点的距离的平均值)
 - ▶ b) 纹理 (灰度值的标准偏差)
 - ▶ c) 周界
 - ▶ d) 区域
 - ▶ e) 光滑度 (半径长度的局部变化)
 - ▶ f) 紧凑性 (周长² / 面积 - 1.0)
 - ▶ g) 凹度 (轮廓凹部的严重程度)
 - ▶ h) 凹点 (轮廓的凹入部分的数量)
 - ▶ i) 对称
 - ▶ j) 分形维数 ("海岸线近似" - 1)
- ▶ 对每个数据分别求平均值, 标准误差, “最差”或最大, 产生30个特征
- ▶ 所有功能值都用四位有效数字重新编码

DATA ANALYSIS

IN:

▶ df.head(10)

OUT:

id	diagnosis	radius_mean	texture_mean
842302	M	17.99	10.38
842517	M	20.57	17.77
84300903	M	19.69	21.25
84348301	M	11.42	20.38

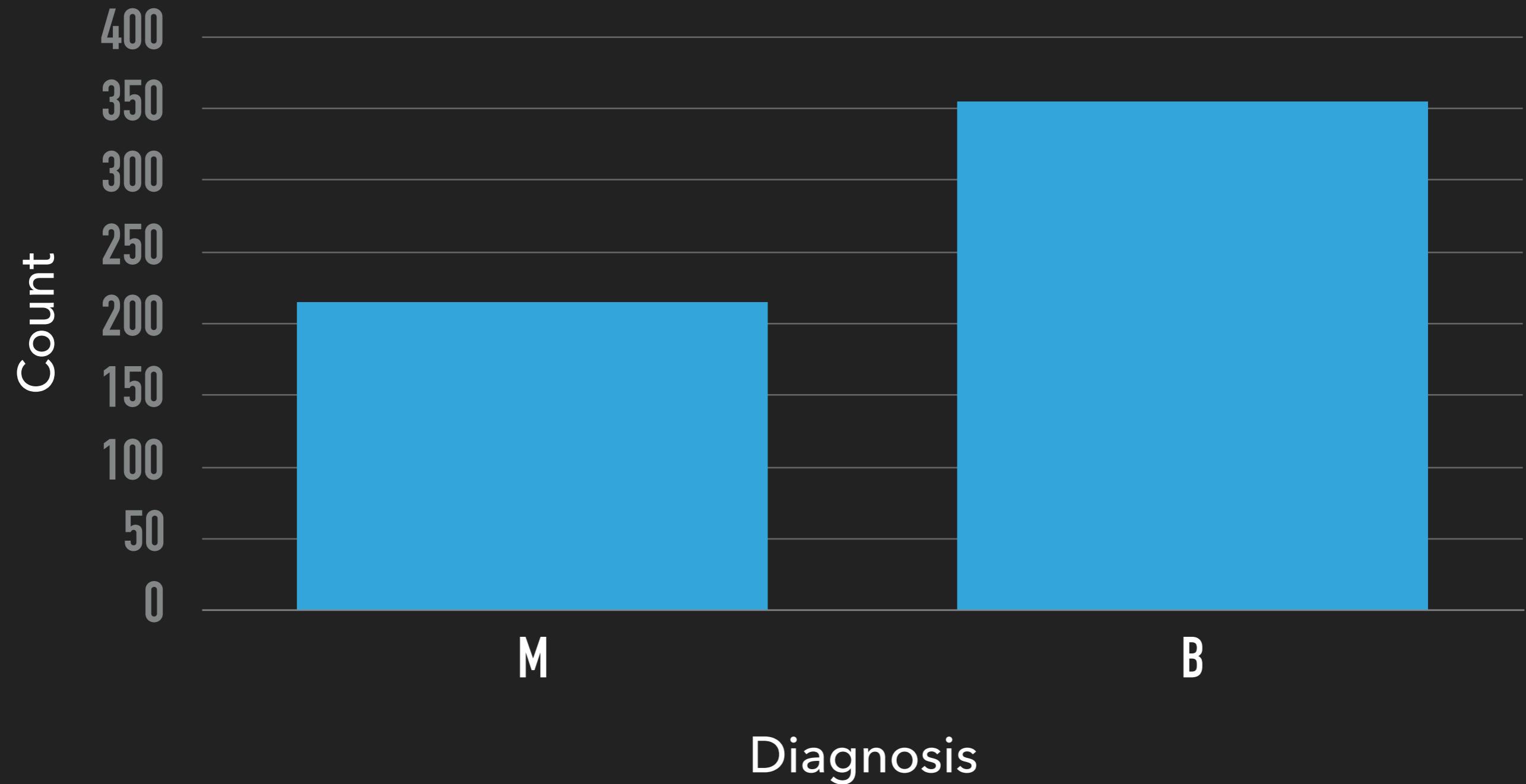
... (TOTAL 10 ROWS * 32 COLUMNS)

DATA ANALYSIS

IN:

▶ `sn.countplot(df["diagnosis"],label='count')`

OUT: ▶ `<matplotlib.axes._subplots.AxesSubplot at 0x18697281cc0>`



IN:

- ▶ dia = df['diagnosis']
- ▶ df.drop('id', axis=1,inplace=True)
- ▶ df.drop('diagnosis',axis=1,inplace=True)

id 不用于classification, 删去, diagnosis为分类标准

feature 的名字并不需要了解

DATA VISUALIZATION

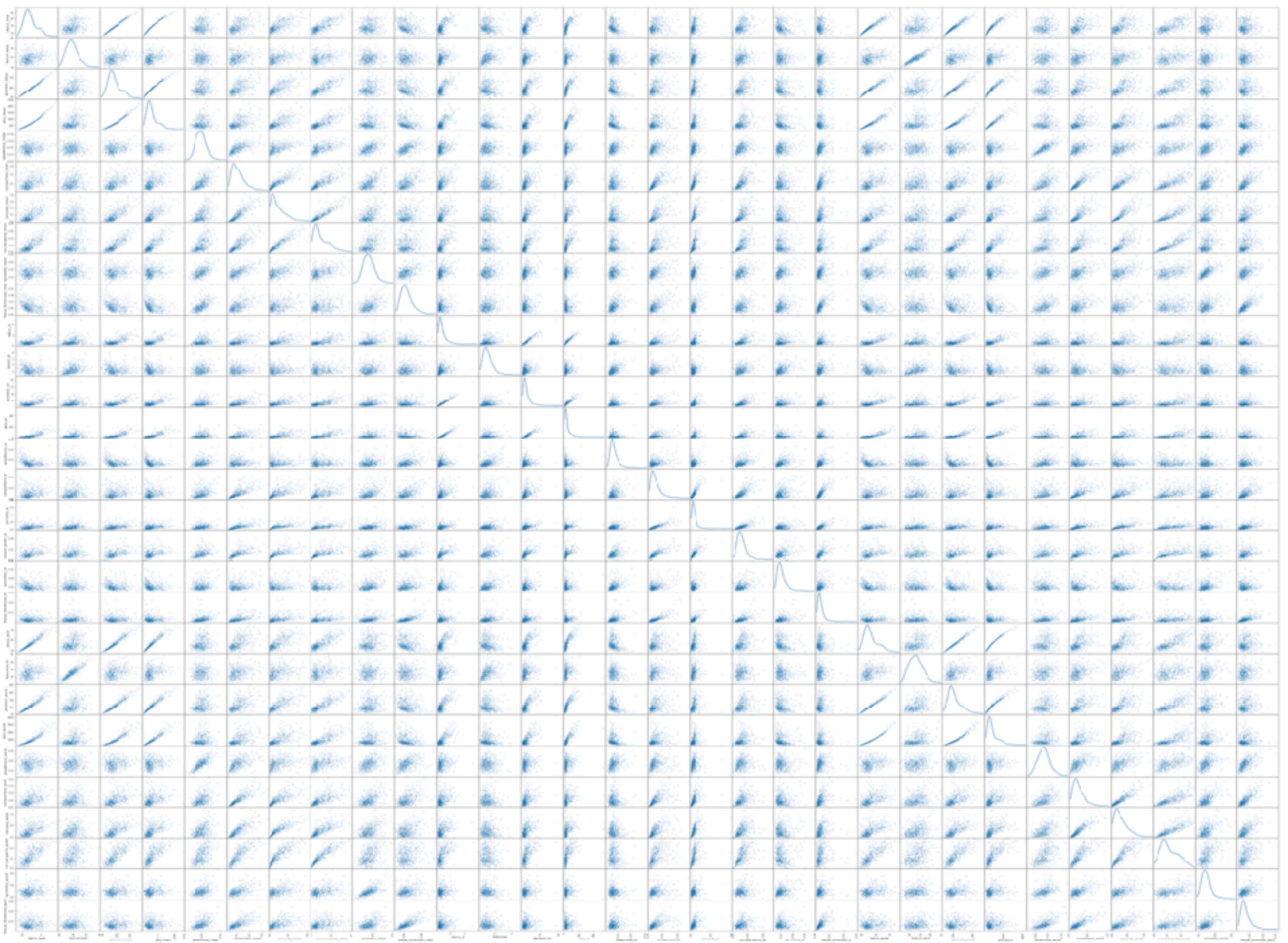
```
# 相似相关性热力图
```

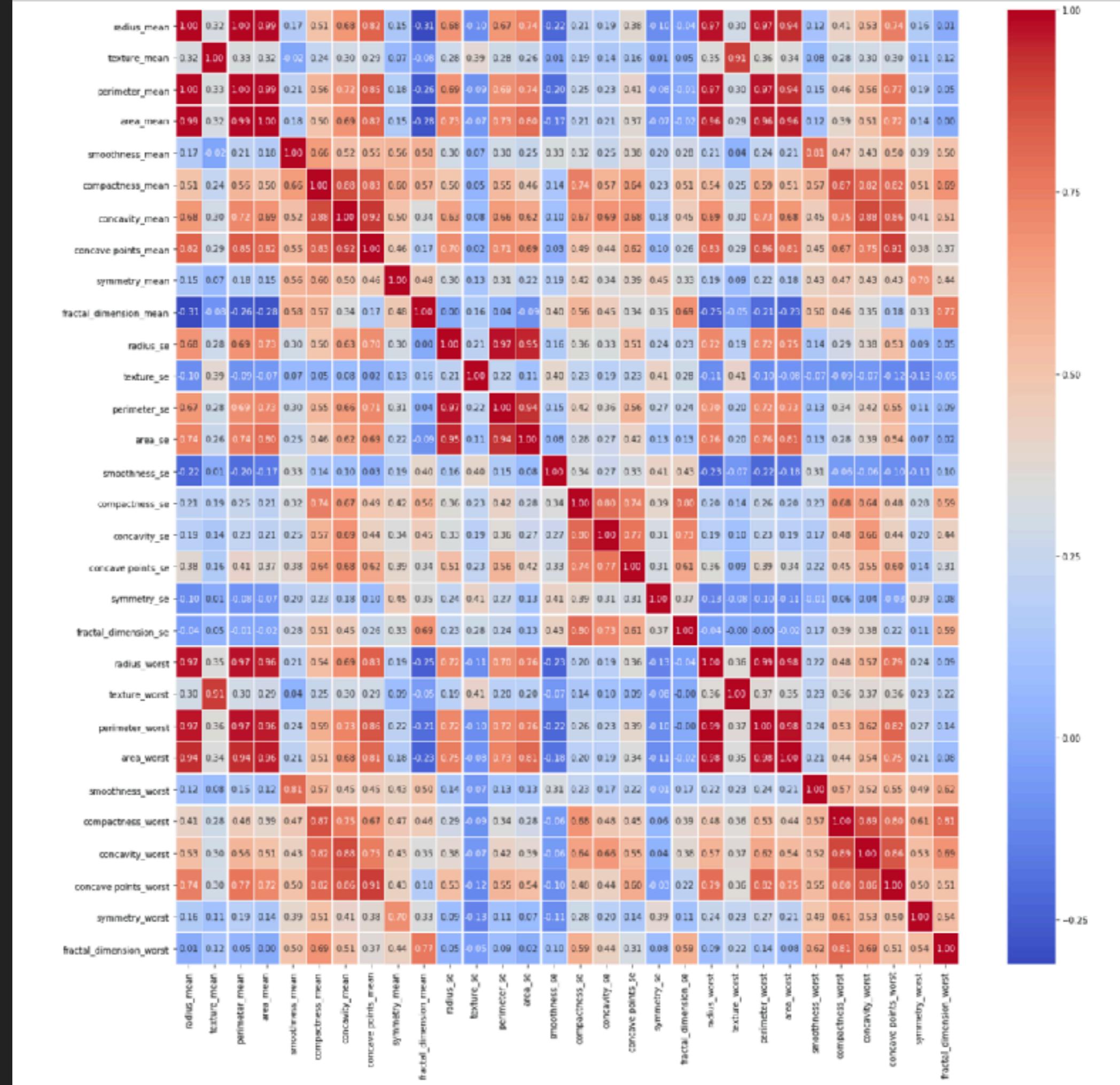
IN:

- ▶ `plt.subplots(figsize=(20, 20))`
- ▶ `sn.heatmap(data.corr(), annot=True, linewidths=.5, fmt= '.2f', cmap='coolwarm')`

OUT:

- ▶ `<matplotlib.axes._subplots.AxesSubplot at 0x1d3a9ce3a58>`



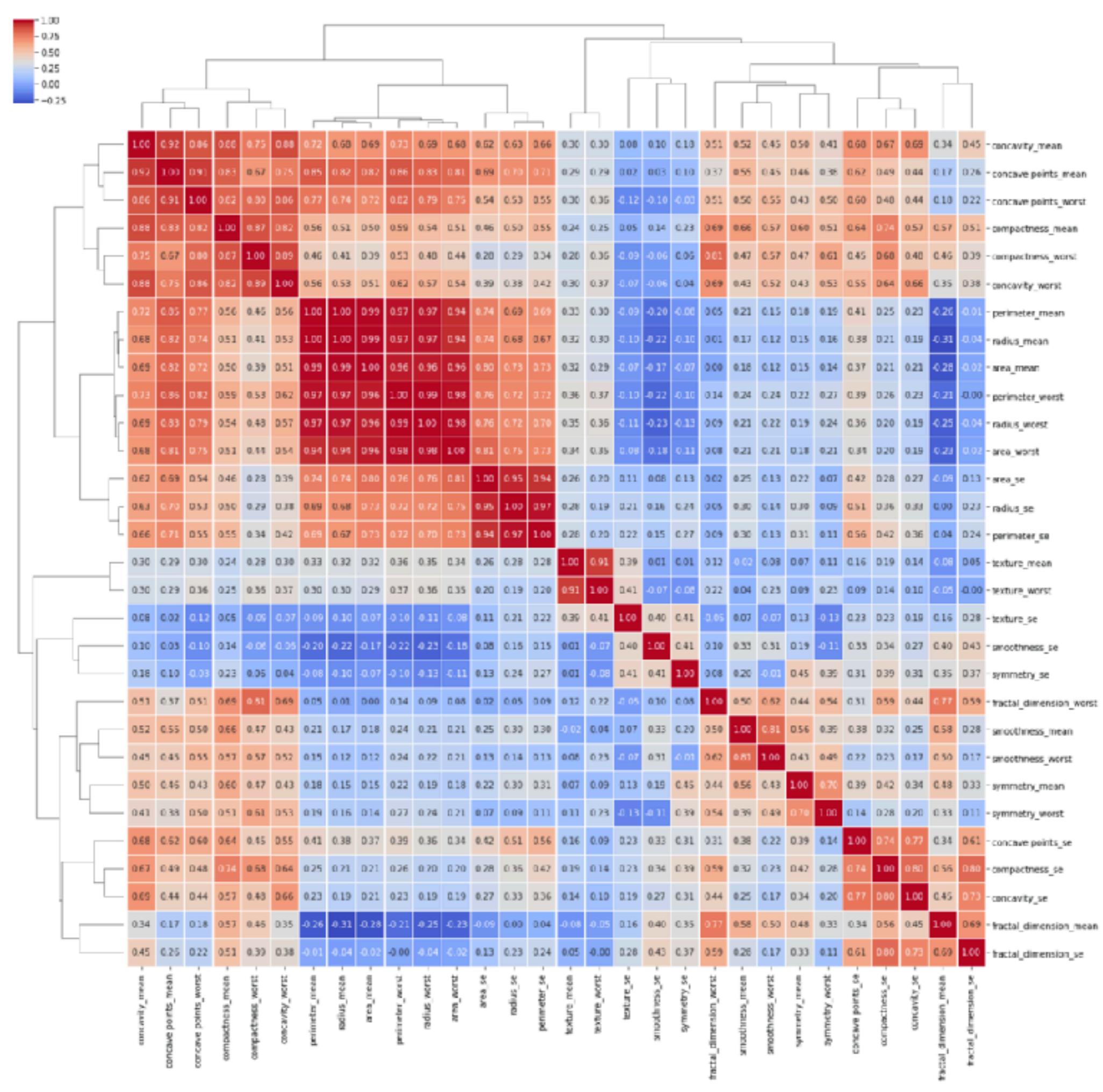


IN: # 加入分类的相关性热力图

```
▶ sn.clustermap(data.corr(), annot=True, linewidths=.5, fmt='.
2f', cmap='coolwarm', figsize=(20,20))
```

OUT:

```
▶ <matplotlib.axes._subplots.AxesSubplot at 0x1d3a9ce3a58>
  ▶ new correlation features
  ▶ 32 to 16 dimension
  ▶ concavity_mean, concave points_mean,
    concave points_worst
  ▶ compactness_mean, compactness_worst,
    concavity_worst
  ▶ perimeter_mean, radius_mean, area_mean,
    perimeter_worst, radius_worst, area_worst
  ▶ area_se, radius_se, perimeter_se
  ▶ texture_mean, texture_worst
  ▶ texture_se, smoothness_se, symmetry_se
  ▶ fractal_dimesion_worst
  ▶ smoothness_mean
  ▶ smoothness_worst
  ▶ symmetry_mean
  ▶ symmetry_worst
  ▶ concave points_se
  ▶ compactness_se
  ▶ concavity_se
  ▶ fractal_dimesion_mean
```



CLASSIFICATION ALGORITHM

**IN 32
FEATURES**

LOGISTIC REGRESSION

IN:

- ▶ LR = LogisticRegression()
- ▶ LR.fit(X_train, y_train)
- ▶ LR_pred = LR.predict(X_test)
- ▶ print(metrics.classification_report(y_test,LR_pred, digits = 5))
- ▶ LR_cm=metrics.confusion_matrix(y_test,LR_pred)

OUT:

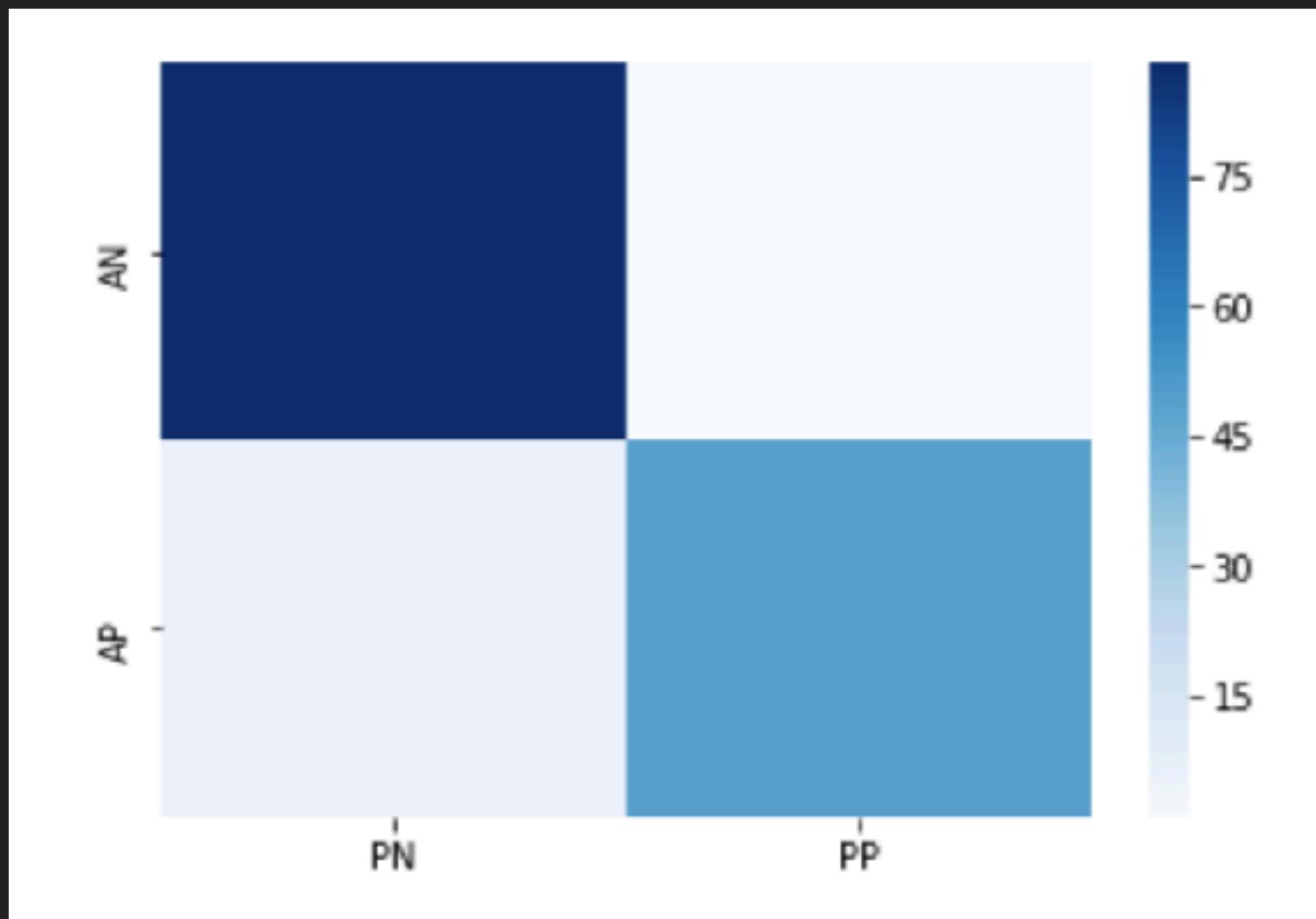
	Precision	Recall	f1-Score	Support
B	0.94624	0.98876	0.96703	89
M	0.98000	0.90741	0.94231	54
Avg/Total	0.95899	0.95804	0.94231	143

CLASSIFICATION ALGORITHM - LOGISTIC REGRESSION

IN: # 计算混淆矩阵

- ▶ `print(LR_cm)`
- ▶ `df_cm = pd.DataFrame(LR_cm, index = ['AN', 'AP'], columns=['PN', 'PP'])`
- ▶ `sn.heatmap(df_cm, cbar = True, cmap = 'Blues')`

OUT: ▶ <matplotlib.axes._subplots.AxesSubplot at 0x186b5dde198>



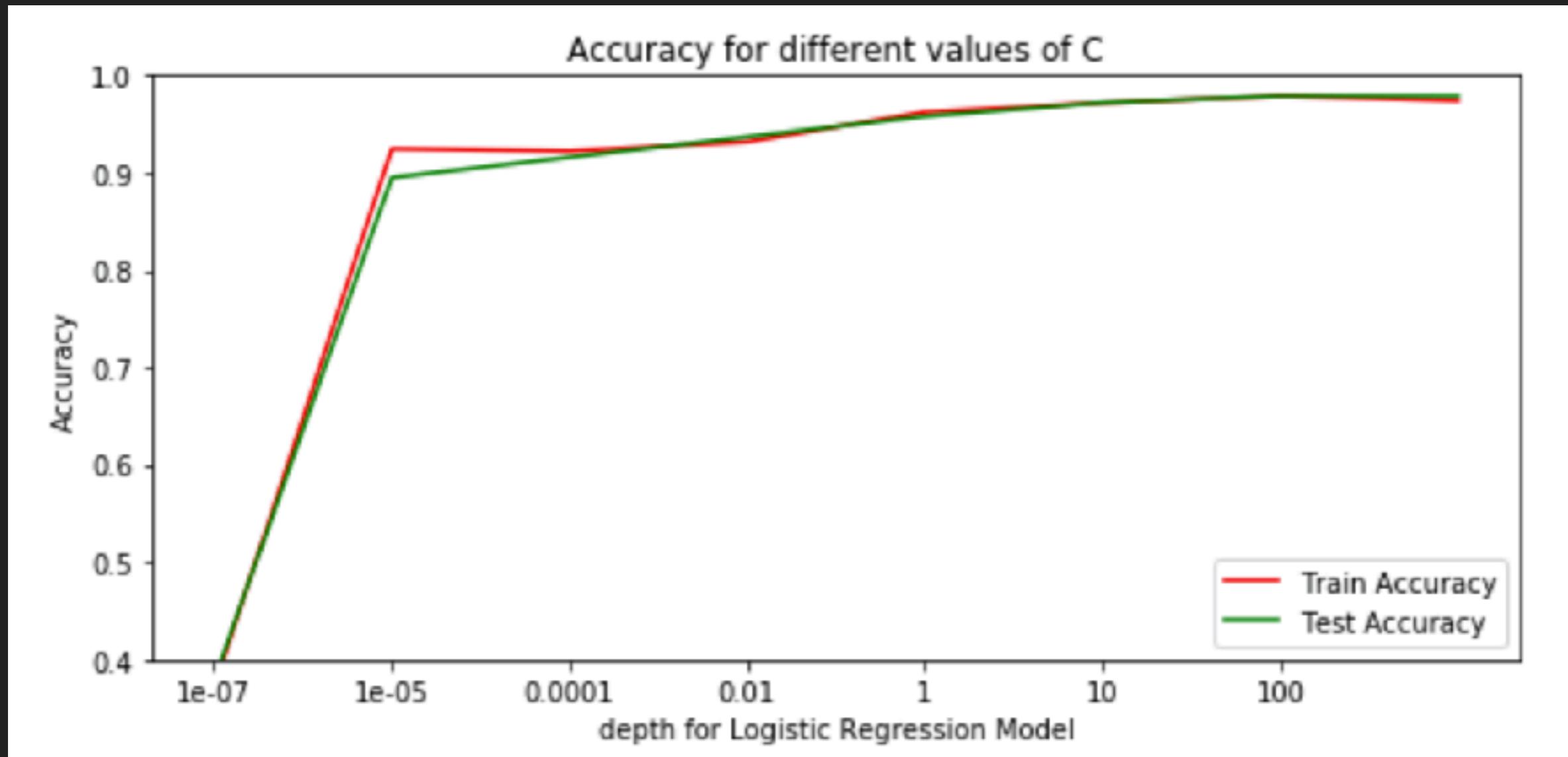
CLASSIFICATION ALGORITHM - LOGISTIC REGRESSION

IN: # Change Depth

▶ depth = [0.000001, 0.0001, 0.001, 0.01, 1, 10, 100, 1000]

OUT:

▶ Accuracy is maximum at depth 100 and accuracy 0.979



LOGISTIC REGRESSION+

IN:

- ▶ LR_pro = LogisticRegression(C=100)
- ▶ LR_pro.fit(X_train, y_train)
- ▶ LR_pred_pro = LR_pro.predict(X_test)
- ▶ print(metrics.classification_report(y_test,LR_pred_pro,digits = 5))
- ▶ LR_cm_pro=metrics.confusion_matrix(y_test,LR_pred_pro)

OUT:

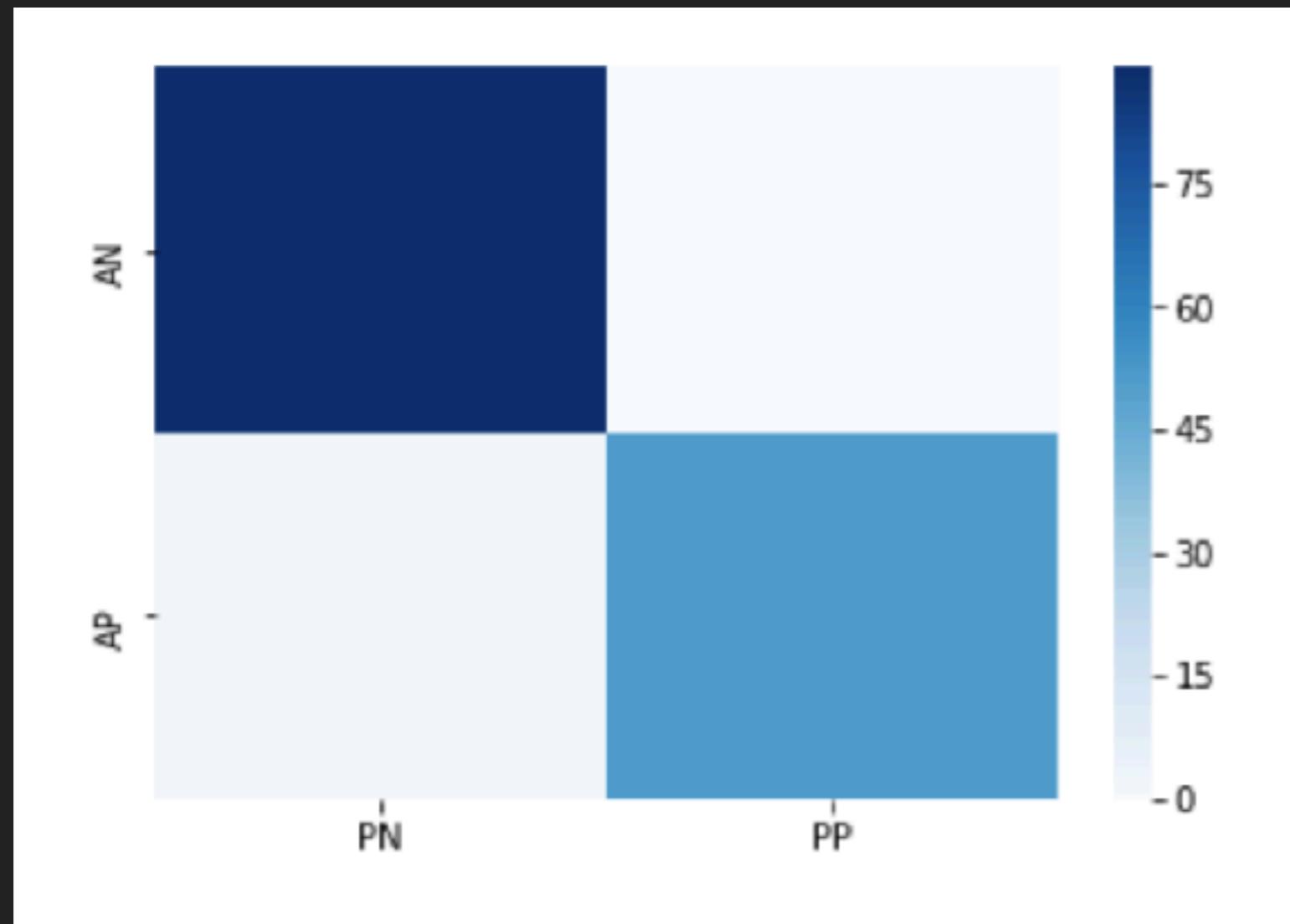
	Precision	Recall	f1-Score	Support
B	0.96739	1.00000	0.98343	89
M	1.00000	0.94444	0.97143	54
Avg/Total	0.97971	0.97902	0.97890	143

CLASSIFICATION ALGORITHM - LOGISTIC REGRESSION

IN: # 计算混淆矩阵

- ▶ `print(LR_cm_pro)`
- ▶ `df_cm_pro = pd.DataFrame(LR_cm_pro, index = ['AN', 'AP'], columns=['PN', 'PP'])`
- ▶ `sn.heatmap(df_cm_pro, cbar = True, cmap = 'Blues')`

OUT: ▶ <matplotlib.axes._subplots.AxesSubplot at 0x186ad715860>



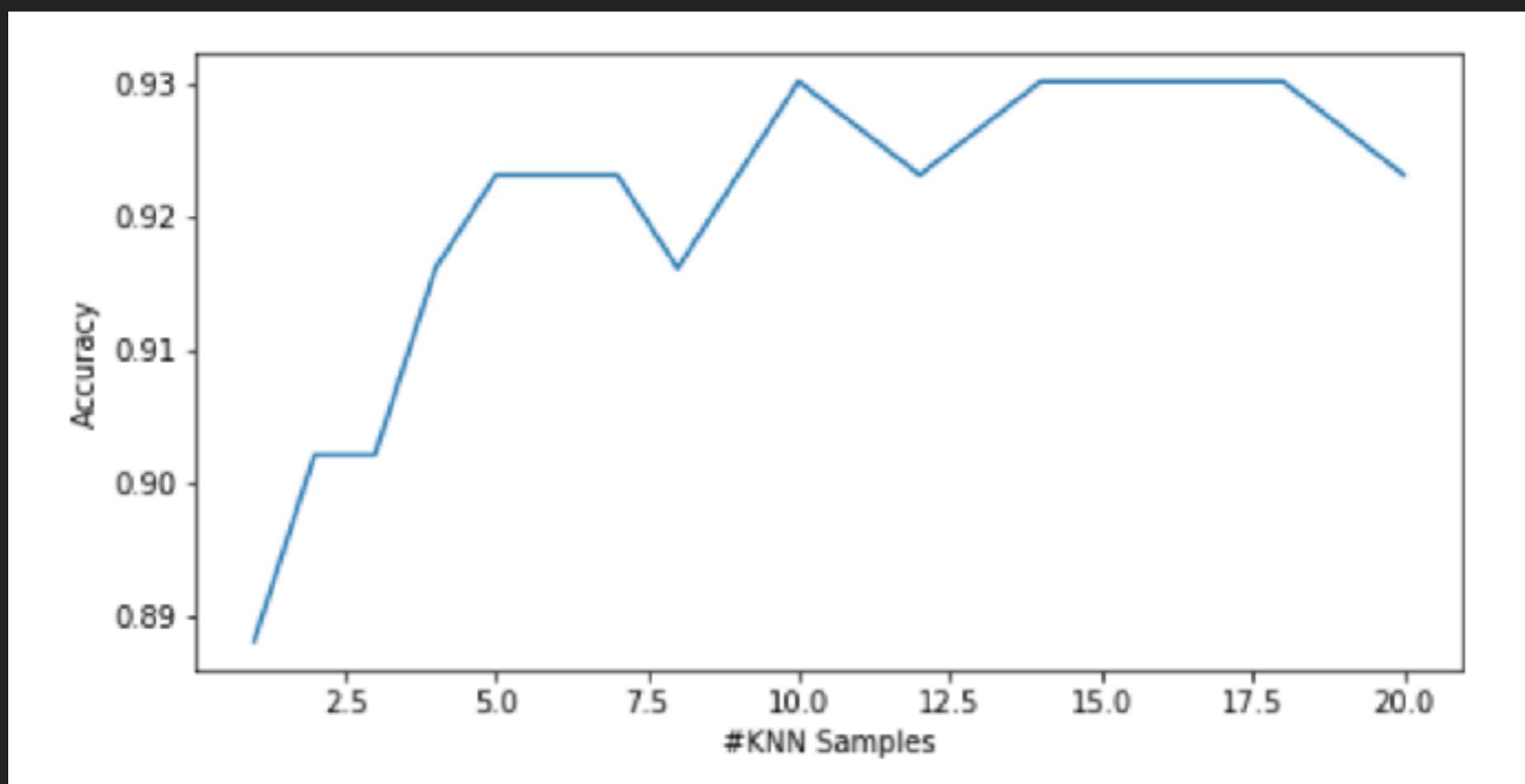
KNN

IN: # Change Number N of Nearest-Neighbor(s)

- ▶ $N = [1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 18, 20]$

OUT:

- ▶ Accuracy is max in Sample 10 and accuracy 0.9301



CLASSIFICATION ALGORITHM - KNN

IN:

- ▶ `KNN = KNeighborsClassifier(n_neighbors=10)`
- ▶ `print(metrics.classification_report(y_test,KNN_pred, digits = 5))`

OUT:

	Precision	Recall	f1-Score	Support
B	0.92473	0.96629	0.94505	89
M	0.94000	0.87037	0.90385	54
Avg/Total	0.93050	0.93007	0.92949	143

GAUSSIAN NAIVE BAYES

IN:

- ▶ GNB = GaussianNB(priors=None)
- ▶ GNB.fit(X_train,y_train)
- ▶ GNB_pred = GNB.predict(X_test)
- ▶ GNB_cm=metrics.confusion_matrix(y_test,GNB_pred)
- ▶ print(metrics.classification_report(y_test,GNB_pred, digits = 5))

OUT:

	Precision	Recall	f1-Score	Support
B	0.92632	0.98876	0.95652	89
M	0.97917	0.87037	0.92157	54
Avg/Total	0.94627	0.94406	0.94332	143

**IN 16
FEATURES**

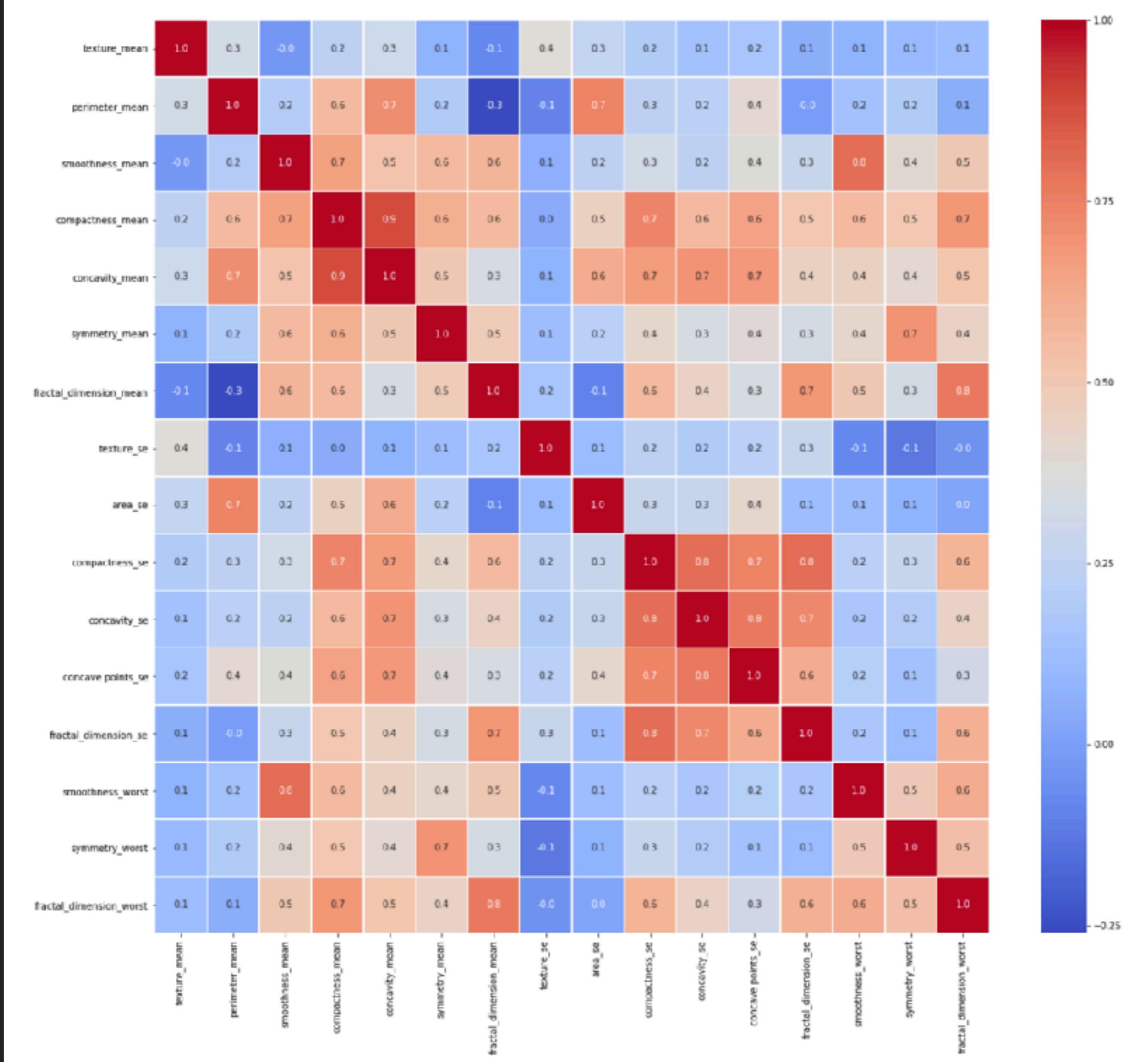
DROP ATTRIBUTES

IN:

- ▶ `drop_list = ['concave points_mean', 'concave points_worst', 'compactness_worst', 'concavity_worst', 'radius_mean', 'area_mean', 'perimeter_worst', 'radius_worst', 'area_worst', 'radius_se', 'perimeter_se', 'texture_worst', 'smoothness_se', 'symmetry_se']`
- ▶ `data_d = data.drop(drop_list, axis=1)`
- ▶ `data_d.shape`

OUT:

- ▶ `<matplotlib.axes._subplots.AxesSubplot at 0x186b4e82ba8>`



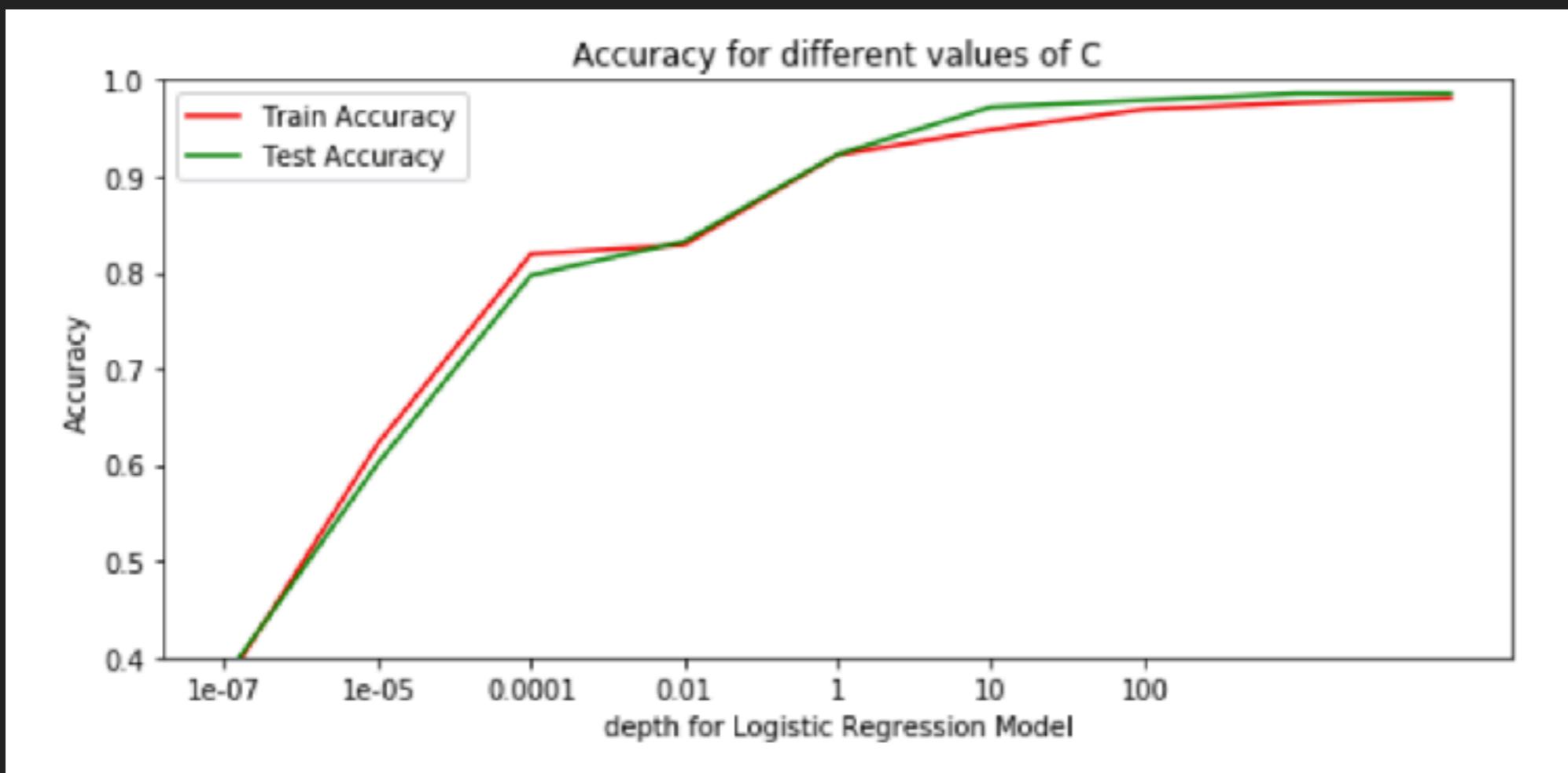
LOGISTIC REGRESSION

IN:

- ▶ depth = [0.0000001, 0.00001, 0.0001, 0.01, 1, 10, 100, 1000, 10000]

OUT:

- ▶ Accuracy is maximum at depth 1000 and accuracy 0.986



CLASSIFICATION ALGORITHM - LOGISTIC REGRESSION

IN:

- ▶ LR_pro_d = LogisticRegression(C=1000)

OUT:

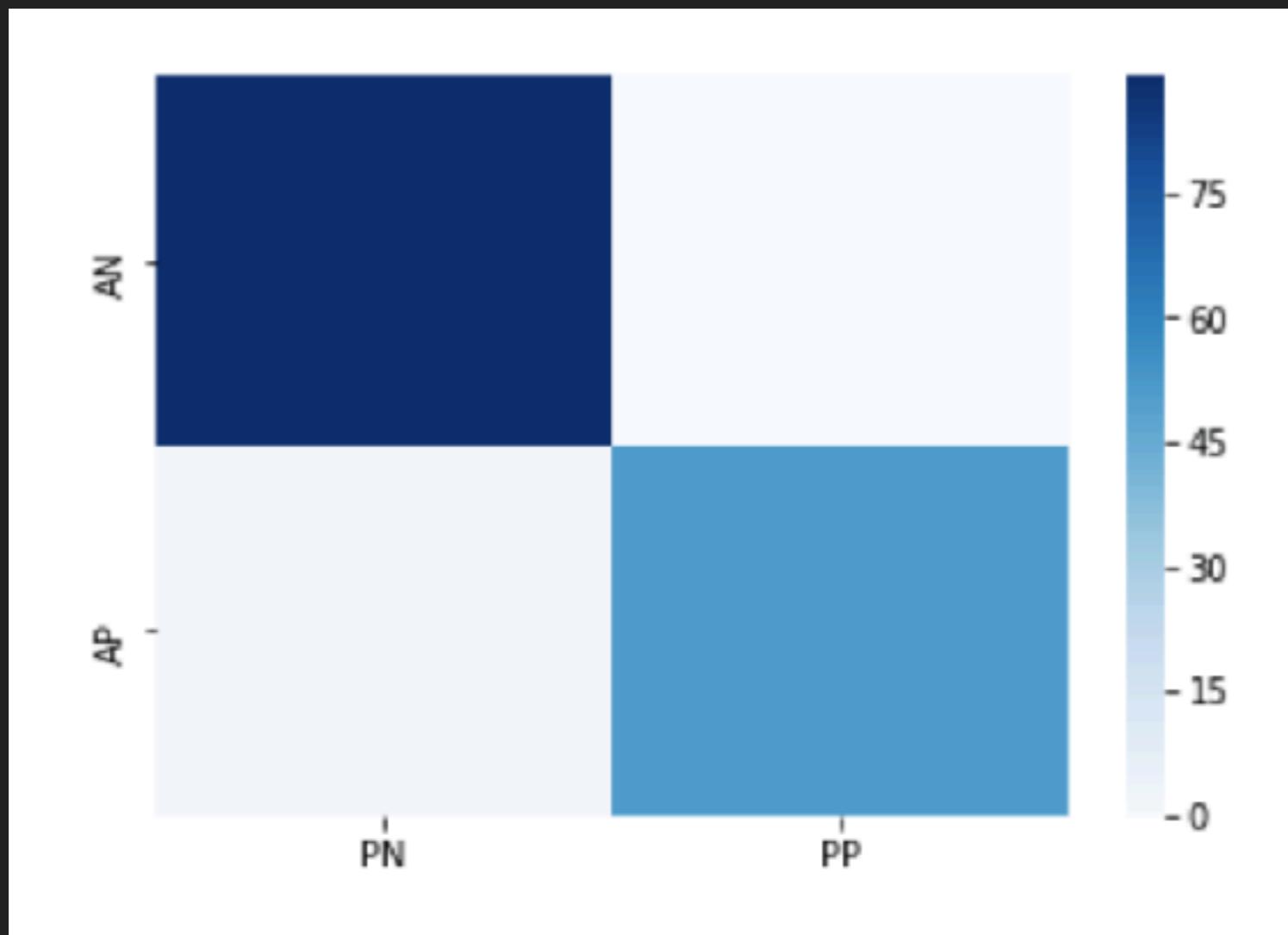
	Precision	Recall	f1-Score	Support
B	0.97802	1.00000	0.98889	89
M	1.00000	0.96296	0.98113	54
Avg/Total	0.98632	0.98601	0.98596	143

CLASSIFICATION ALGORITHM - LOGISTIC REGRESSION

IN:

- ▶ `print(LR_cm_pro_d)`
- ▶ `df_cm_pro_d = pd.DataFrame(LR_cm_pro_d, index = ['AN','AP'], columns=['PN','PP'])`
- ▶ `sn.heatmap(df_cm_pro_d, cbar = True, cmap = 'Blues')`

OUT: ▶ <matplotlib.axes._subplots.AxesSubplot at 0x186b5ec8b70>



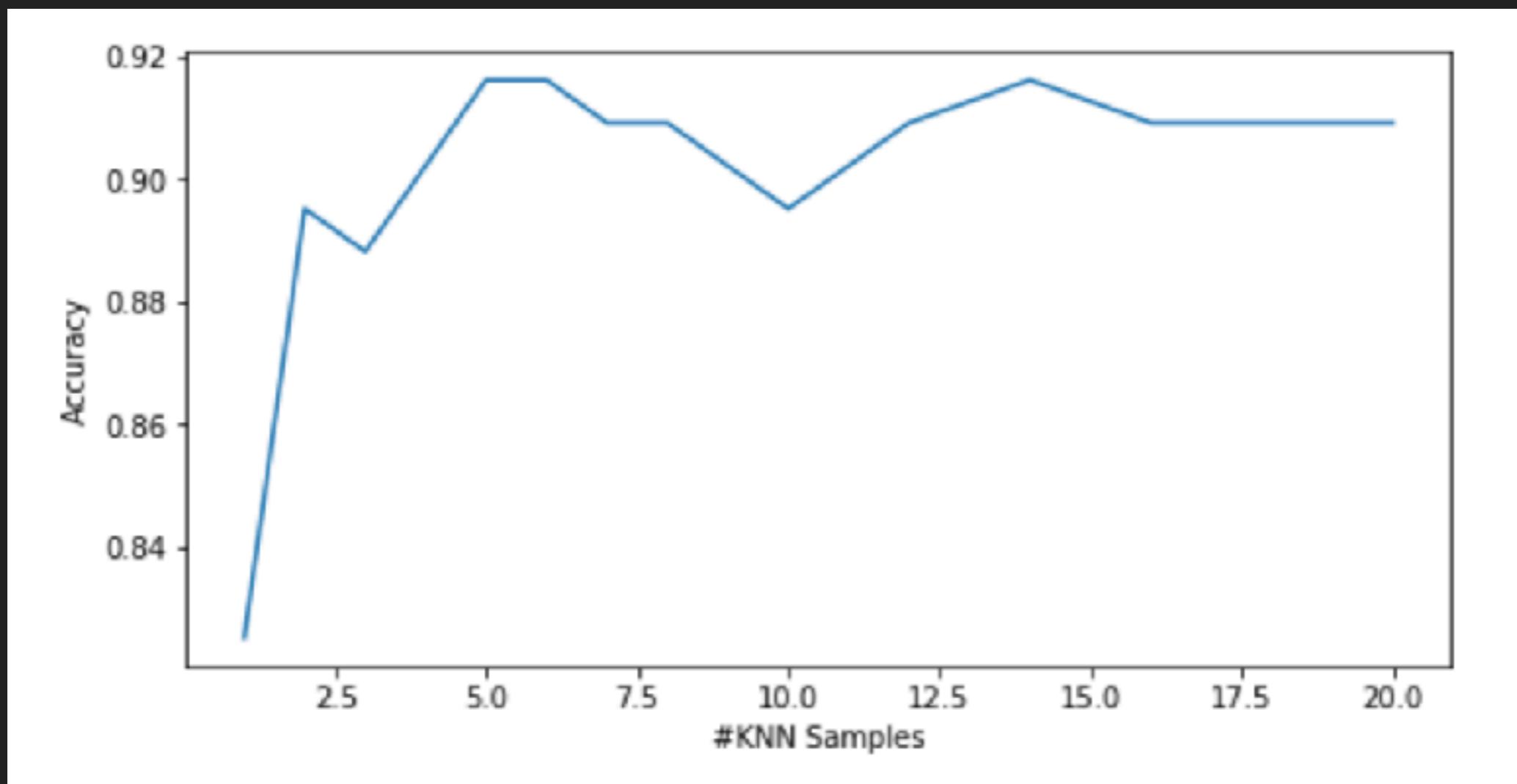
KNN

IN:

- ▶ $N = [1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 18, 20]$

OUT:

- ▶ Accuracy is max in Sample 5 and accuracy 0.9161



CLASSIFICATION ALGORITHM - KNN

IN:

- ▶ KNN_pro_d = KNeighborsClassifier(n_neighbors=5)

OUT:

	Precision	Recall	f1-Score	Support
B	0.91398	0.95506	0.93407	89
M	0.92000	0.85185	0.88462	54
Avg/Total	0.91625	0.91608	0.91539	143

RANDOM FOREST

IN:

- ▶ RF_d = RandomForestClassifier()

OUT:

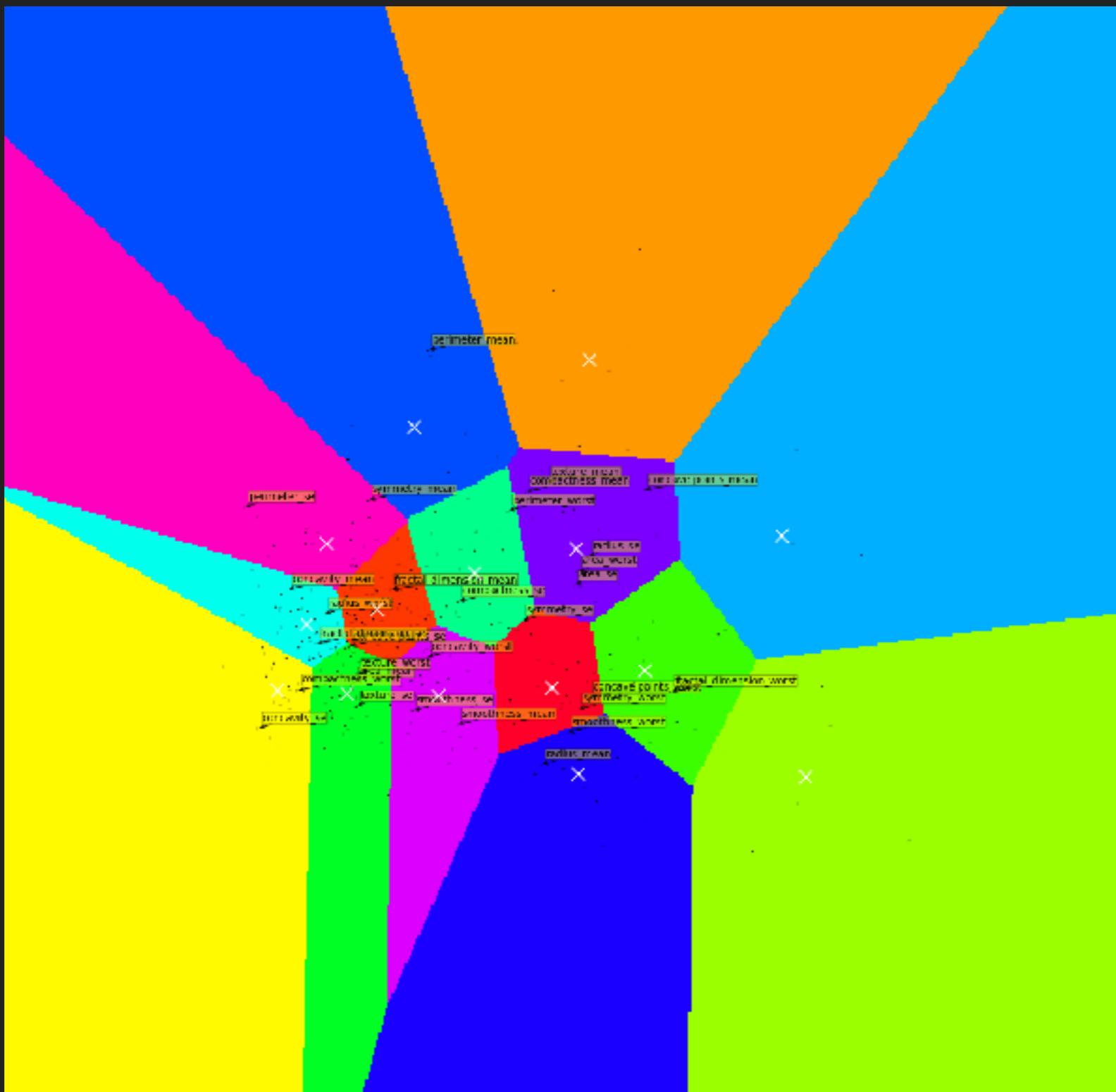
	Precision	Recall	f1-Score	Support
B	0.94624	0.98876	0.96703	89
M	0.98000	0.90741	0.94231	54
Avg/Total	0.95899	0.95804	0.95770	143

DIMENSION REDUCTION*

DIMENSION REDUCTION*

K-means clustering on the digits dataset (PCA-reduced)

PCA*



END