

Benign vs Malignant Findings



This note outlines the clinical differences between **benign** and **malignant** findings, specifically in breast and lung evaluations.



Benign Findings

- Growth Pattern: Non-invasive, localized
- Borders: Well-defined, smooth margins
- Growth Rate: Slow-growing or stable over time
- **Symptoms:** Often asymptomatic
- Histology: Normal cellular architecture, no atypia
- Examples:
 - Breast: Fibroadenoma, cyst, fibrocystic changes
 - Lung: Granuloma, hamartoma, post-inflammatory scar



∏ Impression (Example):

No suspicious mass, distortion, or abnormal calcifications. Findings consistent with benign etiology. Routine follow-up recommended.



Malignant Findings

- Growth Pattern: Invasive, potential to spread (metastasis)
- Borders: Irregular, spiculated or ill-defined
- Growth Rate: Rapid progression
- Symptoms: May include pain, weight loss, cough, bleeding

- Histology: Atypical cells, mitotic activity, abnormal nuclei
- Examples:
 - Breast: Invasive ductal carcinoma, lobular carcinoma
 - Lung: Adenocarcinoma, squamous cell carcinoma, small cell carcinoma

Impression (Example):

Spiculated mass in the upper outer quadrant with associated lymphadenopathy. Findings suspicious for malignancy. Biopsy recommended. Breast Lump Classification: Benign vs Malignant



This project explores the classification of breast tumors into benign (non-cancerous) and malignant (cancerous) types using machine learning models. The goal is to identify which algorithm best predicts tumor type based on various diagnostic features.



Importing the libraries

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score, precision_score, roc_auc_score, ConfusionMatrixDisplay,recall_score, f
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
```

READING IN THE DATASET

```
In [16]: df = pd.read_csv(r'C:\Users\USER\Desktop\breasrcancer datazset.csv')
    df
```

Out	[16]]:
-----	------	----

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	(
0	842302	М	17.99	10.38	122.80	1001.0	0.11840	0.27760	_
1	842517	М	20.57	17.77	132.90	1326.0	0.08474	0.07864	
2	84300903	М	19.69	21.25	130.00	1203.0	0.10960	0.15990	
3	84348301	М	11.42	20.38	77.58	386.1	0.14250	0.28390	
4	84358402	М	20.29	14.34	135.10	1297.0	0.10030	0.13280	
•••									
564	926424	М	21.56	22.39	142.00	1479.0	0.11100	0.11590	
565	926682	М	20.13	28.25	131.20	1261.0	0.09780	0.10340	
566	926954	М	16.60	28.08	108.30	858.1	0.08455	0.10230	
567	927241	М	20.60	29.33	140.10	1265.0	0.11780	0.27700	
568	92751	В	7.76	24.54	47.92	181.0	0.05263	0.04362	

569 rows × 33 columns

In [17]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):

#	Column	Non-Null Count	Dtype
0	id	569 non-null	int64
1	diagnosis	569 non-null	object
2	radius_mean	569 non-null	float64
3	texture_mean	569 non-null	float64
4	perimeter_mean	569 non-null	float64
5	area_mean	569 non-null	float64
6	smoothness_mean	569 non-null	float64
7	compactness_mean	569 non-null	float64
8	concavity_mean	569 non-null	float64
9	concave points_mean	569 non-null	float64
10	symmetry_mean	569 non-null	float64
11	<pre>fractal_dimension_mean</pre>	569 non-null	float64
12	radius_se	569 non-null	float64
13	texture_se	569 non-null	float64
14	perimeter_se	569 non-null	float64
15	area_se	569 non-null	float64
16	smoothness_se	569 non-null	float64
17	compactness_se	569 non-null	float64
18	concavity_se	569 non-null	float64
19	concave points_se	569 non-null	float64
20	symmetry_se	569 non-null	float64
21	<pre>fractal_dimension_se</pre>	569 non-null	float64
22	radius_worst	569 non-null	float64
23	texture_worst	569 non-null	float64
24	perimeter_worst	569 non-null	float64
25	area_worst	569 non-null	float64
26	smoothness_worst	569 non-null	float64
27	compactness_worst	569 non-null	float64
28	concavity_worst	569 non-null	float64
29	concave points_worst	569 non-null	float64
30	symmetry_worst	569 non-null	float64
31	<pre>fractal_dimension_worst</pre>	569 non-null	float64
32	Unnamed: 32	0 non-null	float64
dtype	es: float64(31), int64(1)	, object(1)	

memory usage: 146.8+ KB

SHAPE OF DATASET

df.isna().sum()

```
Out[20]: id
                                     0
         diagnosis
                                     0
         radius_mean
                                     0
         texture_mean
         perimeter_mean
         area_mean
         smoothness_mean
          compactness_mean
         concavity_mean
         concave points_mean
         symmetry_mean
         fractal_dimension_mean
         radius_se
         texture_se
         perimeter_se
         area_se
         smoothness_se
         compactness_se
         concavity_se
         concave points_se
         symmetry_se
         fractal_dimension_se
         radius_worst
         texture_worst
         perimeter_worst
         area_worst
          smoothness_worst
          compactness_worst
          concavity_worst
         concave points_worst
         symmetry_worst
         fractal_dimension_worst
         dtype: int64
```

MAPPING TARGET COLUMN TO INT DATA TYPE

```
In [21]: df['diagnosis'] = df['diagnosis'].map({'B':0, 'M':1})
```

CORRELATION MATRIX

```
In [22]: dcorr = df.corr()
                 plt.figure(figsize=(18, 8))
                 sns.heatmap(dcorr, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.8)
Out[22]: <Axes: >
                                      id -1.00 0.04 0.07 0.10 0.07 0.10 0.01 0.00 0.05 0.04 0.02 0.05 0.14 0.01 0.14 0.18 0.10 0.03 0.06 0.08 0.02 0.03 0.08 0.06 0.08 0.11 0.01 0.00 0.02 0.04 0.03
                              diagnosis -0.04 1.00 0.73 0.42 0.74 0.71 0.36 0.60 0.70 0.78 0.33 0.01 0.57 0.01 0.56 0.55 0.07 0.29 0.25 0.41 0.01 0.08 0.78 0.78 0.78 0.78 0.79 0.42 0.59 0.66 0.79 0.42 0.32
                          radius_mean -0.07 0.73 1.00 0.32 1.00 0.99 0.17 0.51 0.68 0.82 0.15 0.31 0.68 0.10 0.67 0.74 0.22 0.21 0.19 0.38 0.10 0.09 0.30 0.97 0.30 0.97 0.94 0.12 0.41 0.53 0.74 0.16 0.01
                         texture_mean - 0.10 0.42 0.32 1.00 0.33 0.32 0.02 0.24 0.30 0.29 0.07 0.08 0.28 0.39 0.28 0.39 0.28 0.26 0.01 0.19 0.14 0.16 0.01 0.05 0.35 0.91 0.36 0.34 0.08 0.28 0.30 0.30 0.31 0.12
                      perimeter_mean - 0.07 0.74 1.00 0.33 1.00 0.99 0.21 0.56 0.72 0.85 0.18 0.26 0.69 0.09 0.69 0.74 0.20 0.25 0.23 0.41 0.08 0.01 0.97 0.30 0.97 0.94 0.15 0.46 0.56 0.77 0.19 0.05
                                                                                                                                                                                                                            - 0.8
                            area_mean -0.10 0.71 0.99 0.32 0.99 1.00 0.18 0.50 0.69 0.82 0.15 0.28 0.73 0.07 0.73 0.80 0.17 0.21 0.21 0.37 0.07 0.02 0.96 0.29 0.96 0.20 0.96 0.12 0.39 0.51 0.72 0.14 0.00
                    smoothness mean -0.01 0.36 0.17 0.02 0.21 0.18 1.00 0.66 0.52 0.55 0.56 0.58 0.30 0.07 0.30 0.25 0.33 0.32 0.25 0.38 0.20 0.28 0.21 0.04 0.24 0.21 0.81 0.47 0.43 0.50 0.39 0.50
                   compactness mean -0.00 0.60 0.51 0.24 0.56 0.50 0.66 1.00 0.88 0.83 0.60 0.57 0.50 0.05 0.55 0.46 0.14 0.74 0.57 0.64 0.23 0.51 0.54 0.25 0.59 0.51 0.57 0.87 0.82 0.82 0.51 0.69
                       concavity_mean - 0.05
                                                               0.72 | 0.69 | 0.52 | 0.88 | 1.00 | 0.92 | 0.50 | 0.34 | 0.63 | 0.08 | 0.66 | 0.62 | 0.10 | 0.67 | 0.69 | 0.68 | 0.18 | 0.45 | 0.69 | 0.30 | 0.73 | 0.68 | 0.45 | 0.75 | 0.88 | 0.86 | 0.41 | 0.51 |
                 concave points_mean - 0.04 0.78 0.82 0.29 0.85 0.82 0.29 0.85 0.82 0.55 0.83 0.92 1.00 0.46 0.17 0.70 0.02 0.71 0.69 0.03 0.49 0.44 0.62 0.10 0.26 0.83 0.29 0.86 0.81 0.45 0.67 0.75 0.91 0.38 0.37
                                                                                                                                                                                                                            - 0.6
                      symmetry_mean - 0.002 0.33 0.15 0.07 0.18 0.15 0.56 0.60 0.50 0.46 1.00 0.48 0.30 0.13 0.31 0.22 0.19 0.42 0.34 0.39 0.45 0.33 0.19 0.09 0.22 0.18 0.43 0.47 0.43 0.43 0.43
               fractal_dimension_mean -0.05 0.01 0.31 0.08 0.26 0.28 0.58 0.57 0.34 0.17 0.48 1.00 0.00 0.16 0.04 0.09 0.40 0.56 0.45 0.34 0.35 0.69 0.25 0.05 0.21 0.23 0.50 0.46 0.35 0.18 0.33
                              radius_se - 0.14 0.57 0.68 0.28 0.69 0.73 0.30 0.50 0.63 0.70 0.30 0.00 1.00 0.21 0.97 0.95 0.16 0.36 0.33 0.51 0.24 0.23
                                                                                                                                                           0.72 0.19
                             texture_se -0.01 0.01 0.10 0.39 0.09 0.07 0.07 0.05 0.08 0.02 0.13 0.16 0.21 1.00 0.22 0.11 0.40 0.23 0.19 0.23 0.41 0.28 0.11 0.41 0.10 0.08 0.07 0.08
                                                               0.69 0.73 0.30 0.55 0.66 0.71 0.31 0.04 0.97 0.22 1.00 0.94 0.15 0.42 0.36 0.56 0.27 0.24 0.70 0.20 0.72 0.73 0.13 0.34 0.42 0.55 0.11 0.09
                                                                                                                                                                                                                            - 0.4
                                area_se -0.18 0.55 0.74 0.26 0.74 0.80 0.25 0.46 0.62 0.69 0.22 0.09 0.95 0.11 0.94 1.00 0.08 0.28 0.27 0.42 0.13 0.13 0.76 0.20 0.76 0.81 0.13 0.28 0.39 0.54 0.07 0.02
                        smoothness_se - 0.10 0.07 0.22 0.01 0.20 0.17 0.33 0.14 0.10 0.03 0.19 0.40 0.16 0.40 0.15 0.08 1.00 0.34 0.27 0.33 0.41 0.43 0.23 0.07 0.22 0.18 0.31 0.06 0.06 0.10
                       compactness_se - 0.03 0.29 0.21 0.19 0.25 0.21 0.32 0.74 0.67 0.49 0.42 0.56 0.36 0.23 0.42 0.28 0.34 1.00 0.80 0.74 0.39 0.80 0.20 0.14 0.26 0.20 0.23 0.68 0.64 0.48 0.28 0.59
                          concavity_se - 0.06 0.25 0.19 0.14 0.23 0.21 0.25 0.57 0.69 0.44 0.34 0.45 0.33 0.19 0.36 0.27 0.27 0.80 1.00 0.77 0.31 0.73 0.19 0.10 0.23 0.19 0.10 0.23 0.19 0.17 0.48 0.66 0.44 0.20 0.44
                     concave points_se - 0.08 0.41 0.38 0.16 0.41 0.37 0.38 0.64 0.68 0.62 0.39 0.34 0.51 0.23 0.56 0.42 0.33 0.74 0.77 1.00 0.31 0.61 0.36 0.09 0.39 0.34 0.22 0.45 0.55 0.60 0.14 0.31
                                                                                                                                                                                                                             - 0.2
                          symmetry se -0.02 0.01 0.10 0.01 0.02 0.07 0.20 0.23 0.18 0.10 0.45 0.35 0.24 0.41 0.27 0.13 0.41 0.39 0.31 0.31 1.00 0.37 0.13 0.08 0.10 0.11 0.01 0.06 0.04 0.03 0.39 0.08
                  fractal_dimension_se -0.03 0.08 0.04 0.05 0.01 0.02 0.28 0.51 0.45 0.26 0.33 0.69 0.23 0.28 0.24 0.13 0.43 0.80 0.76 0.61 0.37 1.00 0.04 0.00 0.02 0.17 0.39 0.38 0.22 0.11 0.59
                          radius worst -0.08 0.78 0.97 0.35 0.97 0.96 0.21 0.54 0.69 0.83 0.19 0.25 0.72 0.11 0.70 0.76 0.23 0.20 0.19 0.36 0.13 0.04 1.00 0.36 0.99 0.98 0.22 0.48 0.57 0.79 0.24 0.09
                         texture worst -0.06 0.46 0.30 0.91 0.30 0.29 0.04 0.25 0.30 0.29 0.09 0.05 0.19 0.41 0.20 0.20 0.07 0.14 0.10 0.09 0.05 0.00 0.36 1.00 0.37 0.35 0.23 0.36 0.37 0.36 0.23 0.22
                      perimeter_worst -0.08 0.78 0.97 0.36 0.97 0.96 0.24 0.59 0.73 0.86 0.22 0.21 0.72 0.10 0.72 0.76 0.22 0.20 0.23 0.39 0.10 0.00 0.99 0.37 1.00 0.98 0.24 0.53 0.62 0.82 0.27 0.14
                                                                                                                                                                                                                            - 0.0
                            area_worst -0.11 0.73 0.94 0.34 0.94 0.96 0.21 0.51 0.68 0.81 0.18 0.23 0.75 0.08 0.73 0.81 0.18 0.23 0.75 0.08 0.73 0.81 0.18 0.20 0.19 0.34 0.11 0.02 0.98 0.35 0.98 1.00 0.21 0.44 0.54 0.75 0.21 0.08
                    smoothness worst -0.01 0.42 0.12 0.08 0.15 0.12 0.81 0.57 0.45 0.45 0.45 0.45 0.45 0.45 0.45 0.30 0.14 0.07 0.13 0.13 0.31 0.23 0.17 0.22 0.01 0.17 0.22 0.23 0.24 0.21 1.00 0.57 0.52 0.55 0.49 0.62
                   compactness worst -0.00 0.59 0.41 0.28 0.46 0.39 0.47 0.87 0.75 0.67 0.47 0.46 0.29 0.02 0.34 0.28 0.06 0.68 0.48 0.45 0.06 0.39 0.48 0.36 0.53 0.44 0.57 1.00 0.89
                       concavity_worst -0.02 0.66 0.53 0.30 0.56 0.51 0.43 0.82 0.88 0.75 0.43 0.35 0.38 0.07 0.42 0.39 0.06 0.64 0.66 0.55 0.04 0.38 0.57 0.37 0.62 0.54 0.52 0.89 1.00
                                                                                                                                                                                                                              -0.2
                                                                    0.72 0.50 0.82 0.86 0.91 0.43 0.18 0.53 0.12 0.55 0.54 0.10 0.48 0.44 0.60 0.03 0.22 0.79 0.36 0.82
               fractal_dimension_worst -0.03
                                                                                              symmetry_mean
                                                                                                                                            concave points_se
                                                                                                                                                                                                         fractal_dimension_worst
                                                                                   concavity_mean
                                                                                        concave points_mear
                                                                                                                                 compactness_se
                                                                                                                                                      fractal_dimension_se
                                                                                                                                                                texture_worst
                                                                                                                                                                                smoothness_worst
```

SPLITTING THE DATASET

```
In [ ]: X = df.drop(columns=['id', 'diagnosis',], axis=1) # predictor variables
y = df['diagnosis'] # target variables

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, stratify=y, random_state=42) # splitting
```

```
print(X.shape)
print(y.shape)
X_train.head()

(569, 30)
(569,)

Out[]:

radius_mean texture_mean perimeter_mean area_mean smoothness_mean compactness_mean concavity_mean point
```

•		radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	point
	78	20.18	23.97	143.70	1245.0	0.12860	0.34540	0.37540	(
3	30	16.03	15.51	105.80	793.2	0.09491	0.13710	0.12040	(
3	78	13.66	15.15	88.27	580.6	0.08268	0.07548	0.04249	(
2	13	17.42	25.56	114.50	948.0	0.10060	0.11460	0.16820	(
	89	14.64	15.24	95.77	651.9	0.11320	0.13390	0.09966	1

5 rows × 30 columns

DUE TO SOMEW OF THE MODEL ARE SENSITIVE I DECIDED TO SCALE THE DATASET USING STANDARD SCALER

```
In [24]: scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

IMPLEMENTED 4 DIFFERENT MODELS TO CHECK WHICH MODEL IS BEST

```
for class_name, classes in classifier :
    # cross validation score
    scores = cross_val_score(classes, X_scaled, y_train, cv=5, scoring='accuracy')
    print("Cross-validated scores:", scores)
    print("Mean Accuracy: {:.2f}".format(scores.mean()))

# fitting all the classes
    classes.fit(X_scaled, y_train)

# predicting the test set
    y_pred_tree = classes.predict(X_test_scaled)

    print('{:s} : {:.2f}'.format(class_name,accuracy_score(y_test, y_pred_tree)))

Cross-validated scores: [0.9875    0.9875    0.98734177   0.91139241]
```

Cross-validated scores: [0.9875 Mean Accuracy: 0.97 K nearest neighbor : 0.96 Cross-validated scores: [0.9625 1. 0.9875 0.97468354 0.93670886] Mean Accuracy: 0.97 logistic Regression : 0.97 Cross-validated scores: [0.975 0.9875 0.95 0.96202532 0.88607595] Mean Accuracy: 0.95 Random Forest: 0.96 Cross-validated scores: [0.9625 0.9625 0.925 0.86075949 0.87341772] Mean Accuracy: 0.92

KNEIGHBORS HAPPENS TO HAVE THE BEST ACCURACY AND CROSS VAL SCORE

Decision Tree : 0.90

```
print("Cross-validated scores:", scores)
print('----')
# Evaluation metrics
accuracy_tree = accuracy_score(y_test, y_pred)
roc_auc_tree = roc_auc_score(y_test, y_pred)
precision_tree = precision_score(y_test, y_pred)
recall_tree = recall_score(y_test, y_pred)
f1_tree = f1_score(y_test, y_pred)
print(f"Knearest Neighbor - Accuracy: {accuracy_tree:.2f}")
print('----')
print(f"Knearest Neighbor - ROC AUC: {roc_auc_tree:.2f}")
print('----')
print(f"Knearest Neighbor - Precision: {precision_tree:.2f}")
print('----')
print(f"Knearest Neighbor - Recall: {recall_tree:.2f}")
print('----')
print(f"Knearest Neighbor - F1 Score: {f1_tree:.2f}")
print('----')
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
ConfusionMatrixDisplay.from_estimator(knn, X_test_scaled, y_test, cmap='Blues')
```

Knearest Neighbor - Train Accuracy: 0.97 Knearest Neighbor - Test Accuracy: 0.96

Cross-validated scores: [0.95 0.975 0.95 0.975 0.875 0.925

0.9 0.925 0.84615385 0.87179487]

Knearest Neighbor - Accuracy: 0.96

Knearest Neighbor - ROC AUC: 0.95

Knearest Neighbor - Precision: 1.00

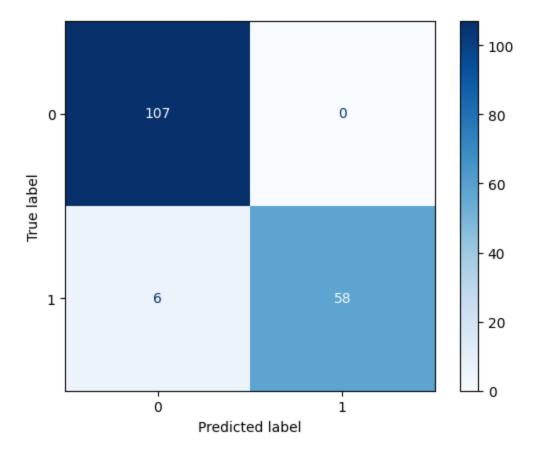
Knearest Neighbor - Recall: 0.91

Knearest Neighbor - F1 Score: 0.95

Classification Report:

support	f1-score	recall	precision	
107	0.97	1.00	0.95	0
64	0.95	0.91	1.00	1
171	0.96			accuracy
171	0.96	0.95	0.97	macro avg
171	0.96	0.96	0.97	weighted avg

Out[39]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x27795b91310>



SAVING THE MODEL

```
In [43]: import pickle
with open('breast_model.pkl', 'wb') as file:
    pickle.dump(knn, file)
```