Breast Cancer Prediction

Group 7

Group Members

Vrushabh Ajaybhai Desai (VrushabhAjaybhaiDesai@my.unt.edu) Document, Code

Varsha Salil (VarshaSalil@my.unt.edu) Document, Code

Kareem Baba Shaik (kareembabashaik@my.unt.edu) Code, Testing

Manasa Kilaru (manasakilaru@my.unt.edu) Dataset, Code

Alekhya Sree Jetti (AlekhyaSreeJetti@my.unt.edu) Document, Testing

Motivation

- In the present world, Breast cancer is one of the major causes of death in women. Over 2 million women get diagnosed with breast cancer every year. The condition is gradually progressing and will only exhibit most of the distinguishing symptoms during more advanced stages. However, the severity of the condition can be curbed by timely prediction and proper treatment.
- The existing diagnosis approaches include physically invasive tests and chemical tests on blood samples. Body cells are then classified into Benign (or non-cancerous) cells or Malignant(cancerous) cells. Cancerous cells can also be detected by analyzing several body parameters like clump thickness, cell size, cell shape etc.
- In this project, we propose a non-invasive study and classification of a patient's body parameters of interest to classify breast cells into two categories i.e, Benign or Malignant, depending on the patient's diagnosis result. This method provides a faster, more economic approach to breast cancer diagnosis.

Objective

- The main objective of the project is to generate an accurate system which predicts Breast Cancer by analyzing several body parameters such as clump thickness, cell size, cell shape etc.

Milestones:

- Project proposal: The project proposal is the initial stage where the project idea is presented, and we started outlining the goals, objectives, and requirements of the project.
- Algorithm generation: Once the proposal has been accepted, we started working on the algorithm where we worked on a step-by-step procedure for solving a specific problem.
- Code generation: After the algorithm is created, the next milestone is the implementation of the algorithm in a programming language.
- Testing the code for efficiency: The code generated needs to be tested for efficiency and accuracy to ensure it performs as intended. This milestone involves running the code through different tests to check for errors and make sure it meets the requirements of the project.
- Project report and presentation: summarizing the entire project, including the goals, objectives, requirements, algorithm, and code. The report and presentation also include the testing results and the overall efficiency of the project.

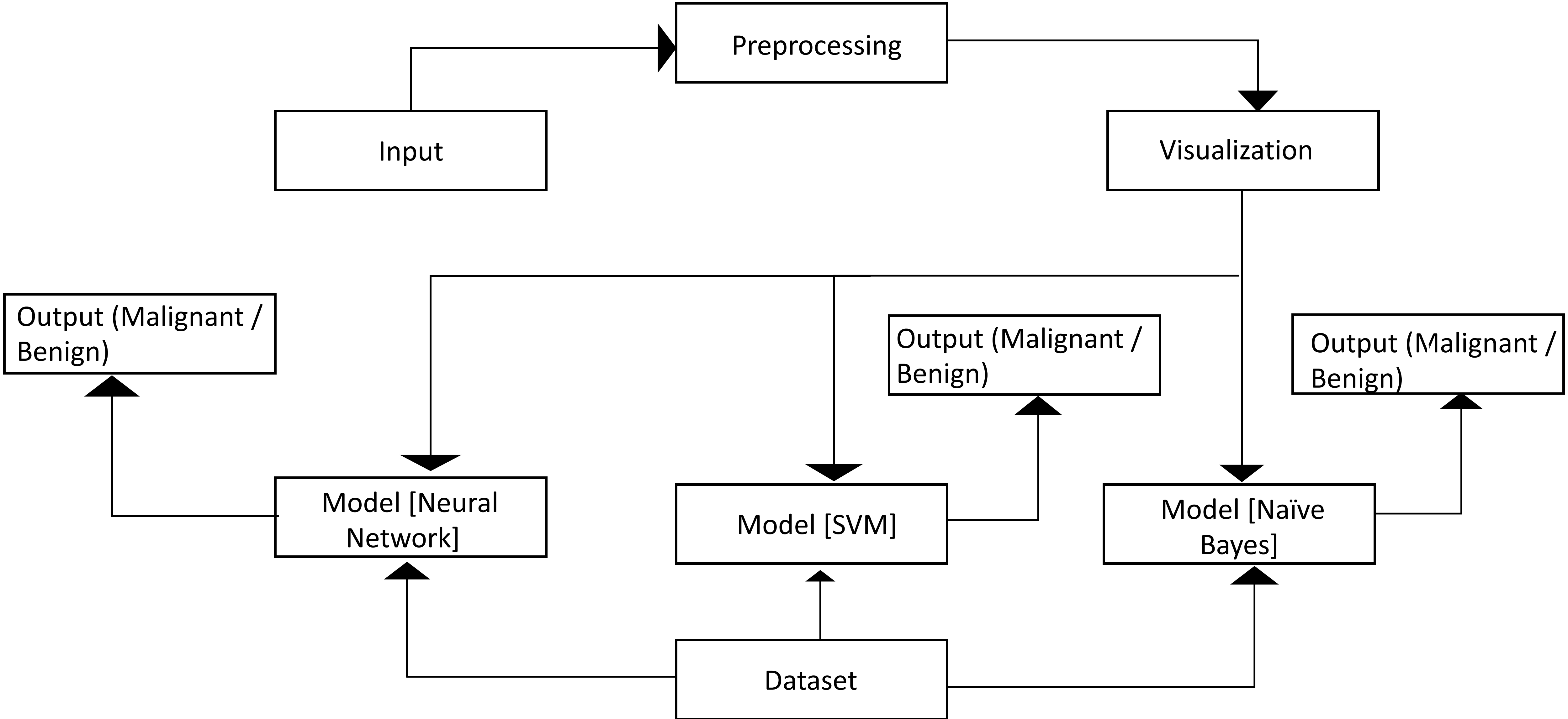
Project Design

- The first step includes obtaining the relevant data for the diagnosis and data preprocessing for classification, by removing null values, dropping unnecessary columns etc. The following dataset will be used for training and testing purposes: https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/.
- The next step is data visualization.
- A Gaussian Naïve Bayes model is generated next from sklearn.naive_bayes package and is fit with the data for training purpose.
- The model then classifies the data into Benign or malignant and returns the predication along with the accuracy of the prediction.
- The dataset was also classified using a Support Vector Machine (SVM) model and a Neural Network model to compare and improve the accuracy of predictions.

Data Specification

- This provided dataset is from wisconsin university. It has 31 columns. In which diagnosis column is the outpuut column which we trying to predict.
- This dataset has 357 Benign and 212 Malign data. And it has 569 rows and 33 columns in which two columns id and unnamed: 32 are not useful for the project.
- All provided columns are non-null data and every column are float values except diagnosis which accepts string values.
- We have observed that parameter_se and area_se are corelated with the feature radius_se, the same way as radius_worst with parameter_worst and area_worst and the third correlation is perimeter_mean, area_mean with radius_mean. We observed this using heatmap.
- After this, we can know that breast cancer is majorly depend on three parameters perimeter, area and radius.
- Link for the dataset: https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/

Workflow


Preprocessing
Input
Visualization
Output (Malignant / Benign)
Output (Malignant / Benign)
Output (Malignant / Benign)
Model [Neural Network]
Model [SVM]
Model [Naïve Bayes]
Dataset

Models used:

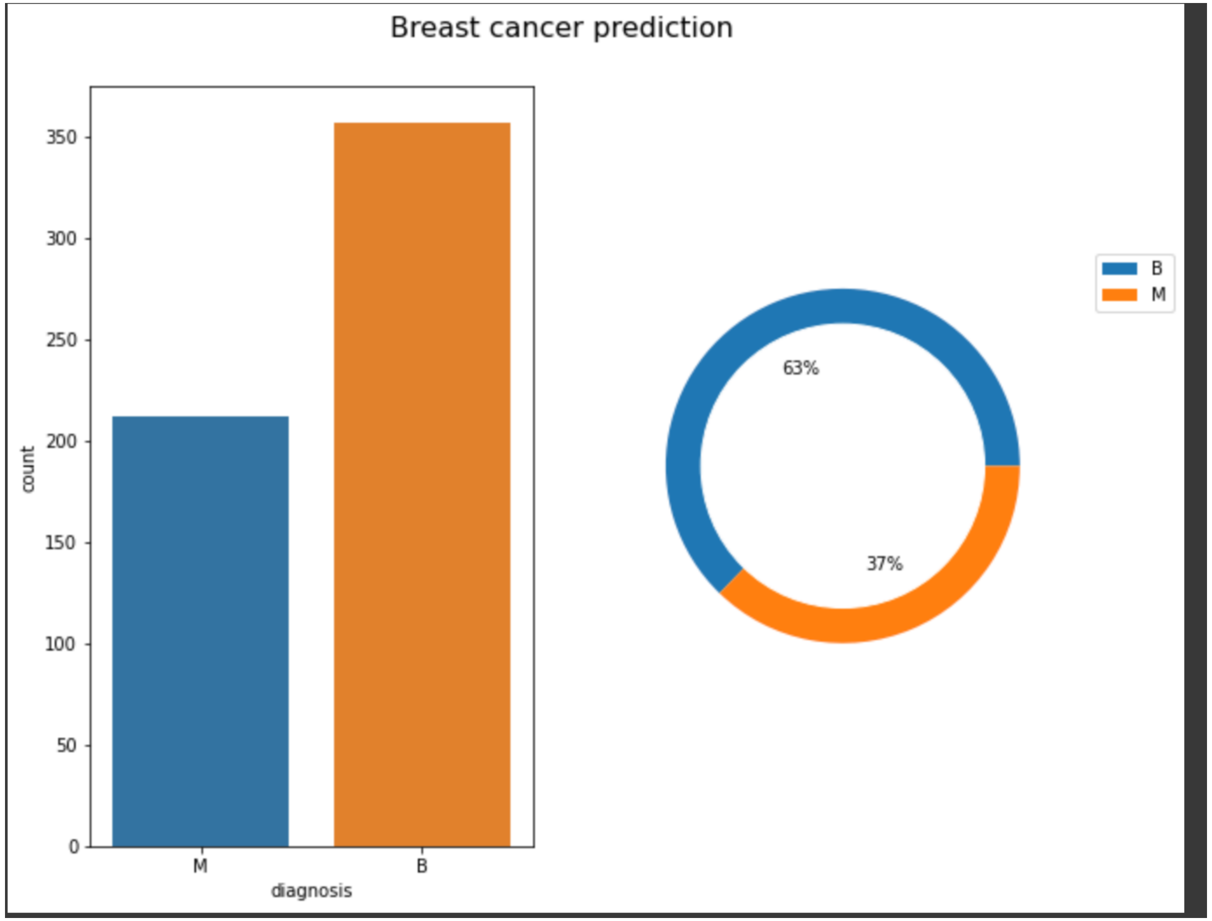
- Gaussian Naïve Bayes Model
- Support Vector Machine
- Neural Network

Future Scope:

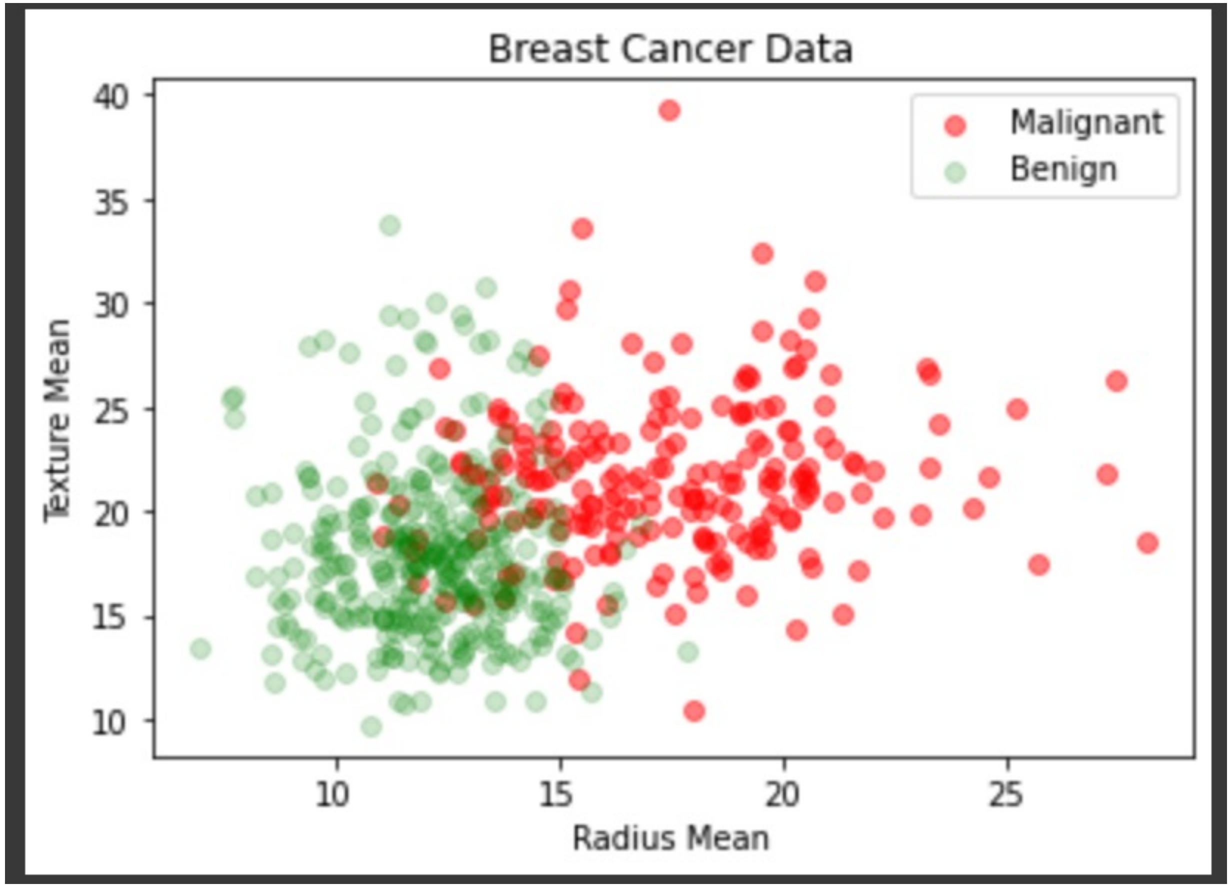
- We have implemented three different models, but we haven’t used validation set. So, if for the phase 2, someone can create validation set and tweak the hyper parameters, we can get more accuracy.
- After implementing the models, we want to implement the transfer learning and see how the models work with other datasets.

Demonstration

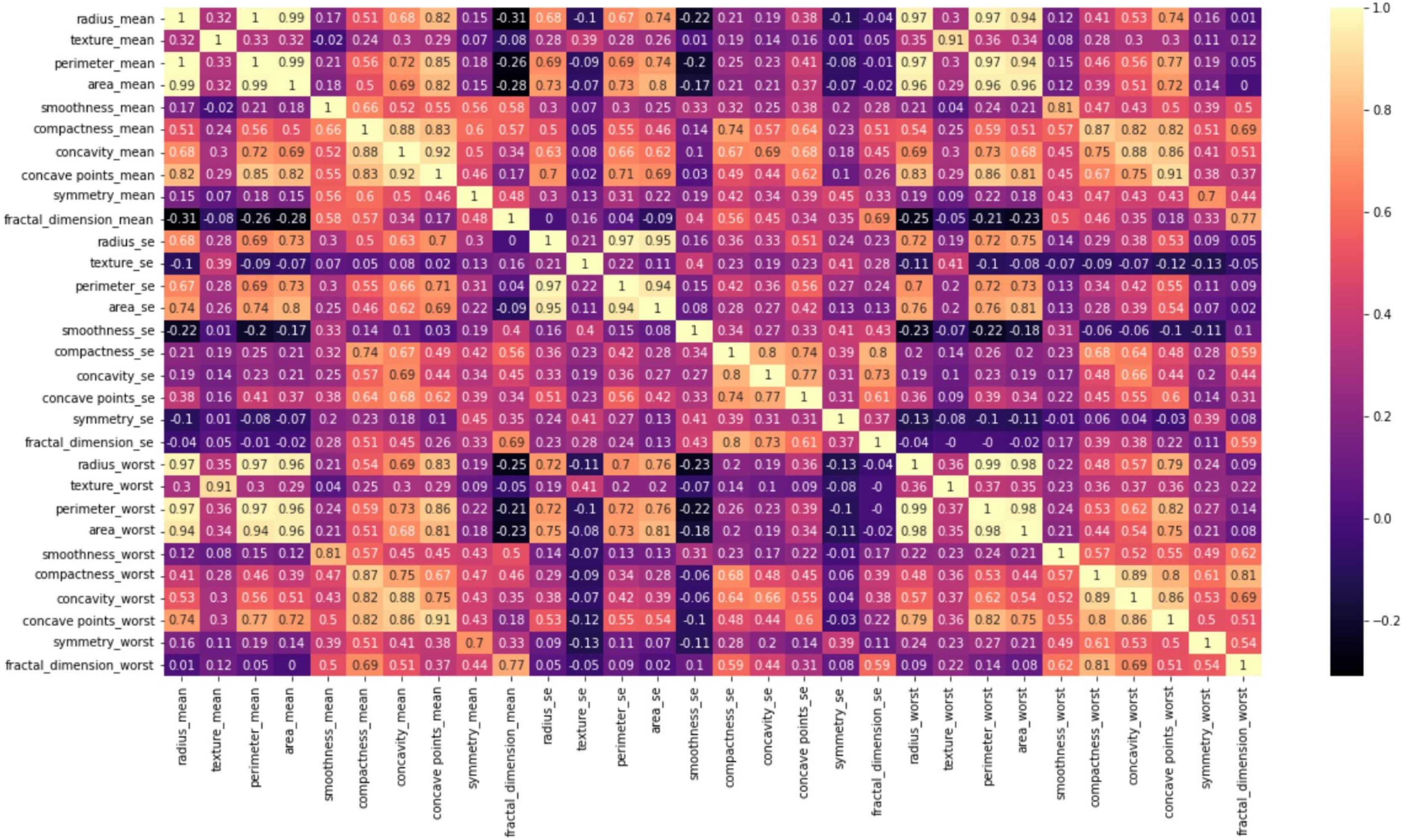
Data visualization:



Scatter plot:

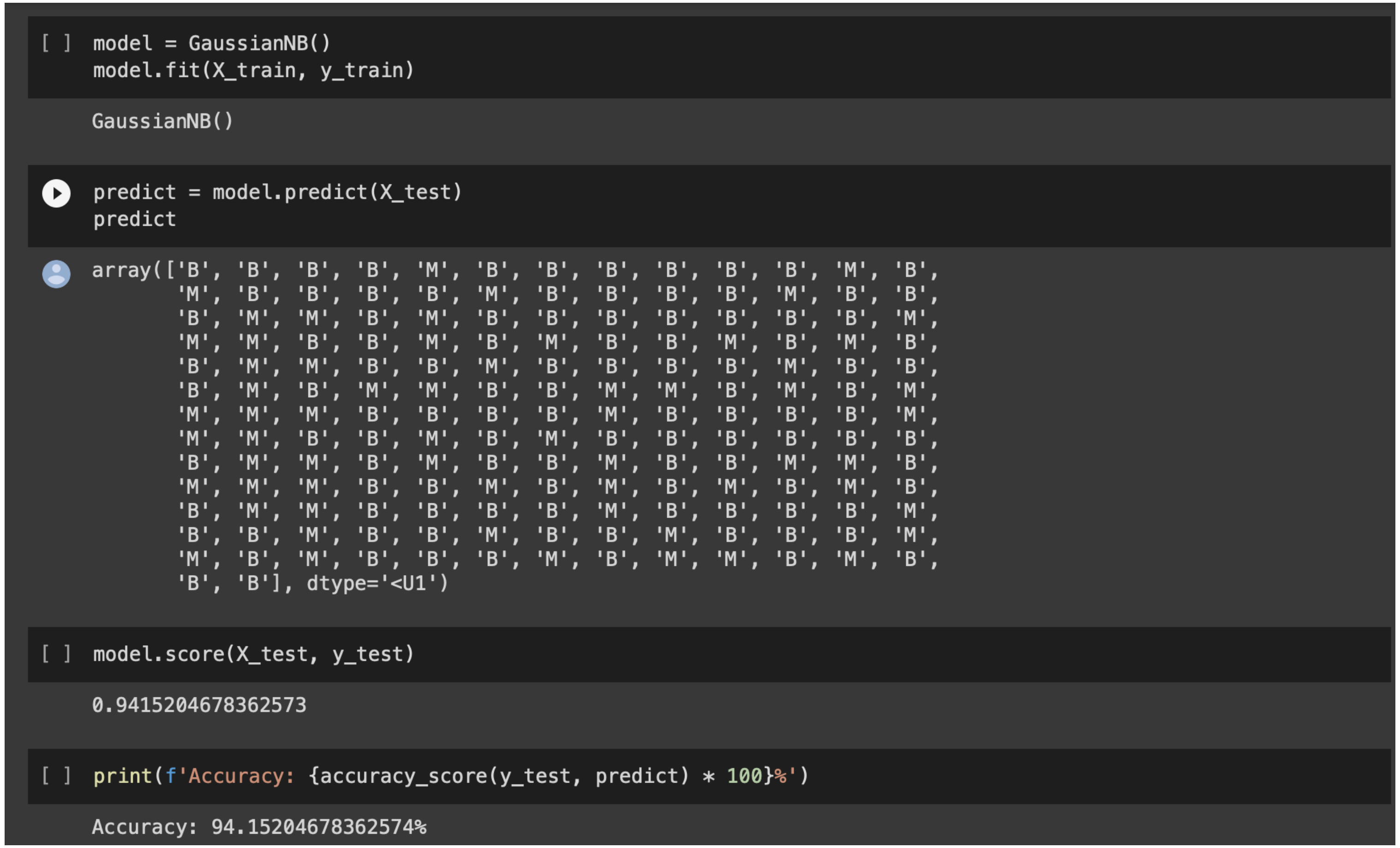


Heat Map:



	radius_mean	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
radius_mean	1	0.32	1	0.99	0.17	0.51	0.68	0.82	0.15	-0.31	0.68	-0.1	0.67	0.74	-0.22	0.21	0.19	0.38	-0.1	-0.04	0.97	0.3	0.97	0.94	0.12	0.41	0.53	0.74	0.16	0.01
texture_mean	0.32	1	0.33	0.32	-0.02	0.24	0.3	0.29	0.07	-0.08	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	0.3	0.3	0.11	0.12
perimeter_mean	1	0.33	1	0.99	0.21	0.56	0.72	0.85	0.18	-0.26	0.69	-0.09	0.69	0.74	-0.2	0.25	0.23	0.41	-0.08	-0.01	0.97	0.3	0.97	0.94	0.15	0.46	0.56	0.77	0.19	0.05
area_mean	0.99	0.32	0.99	1	0.18	0.5	0.69	0.82	0.15	-0.28	0.73	-0.07	0.73	0.8	-0.17	0.21	0.21	0.37	-0.07	-0.02	0.96	0.29	0.96	0.96	0.12	0.39	0.51	0.72	0.14	0
smoothness_mean	0.17	-0.02	0.21	0.18	1	0.66	0.52	0.55	0.56	0.58	0.3	0.07	0.3	0.25	0.33	0.32	0.25	0.38	0.2	0.28	0.21	0.04	0.24	0.21	0.81	0.47	0.43	0.5	0.39	0.5
compactness_mean	0.51	0.24	0.56	0.5	0.66	1	0.88	0.83	0.6	0.57	0.5	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	0.68	0.3	0.72	0.69	0.52	0.88	1	0.92	0.5	0.34	0.63	0.08	0.66	0.62	0.1	0.67	0.69	0.68	0.18	0.45	0.69	0.3	0.73	0.68	0.45	0.75	0.88	0.86	0.41	0.51
concave points_mean	0.82	0.29	0.85	0.82	0.55	0.83	0.92	1	0.46	0.17	0.7	0.02	0.71	0.69	0.03	0.49	0.44	0.62	0.1	0.26	0.83	0.29	0.86	0.81	0.45	0.67	0.75	0.91	0.38	0.37
symmetry_mean	0.15	0.07	0.18	0.15	0.56	0.6	0.5	0.46	1	0.48	0.3	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	0.7	0.44
fractal_dimension_mean	-0.31	-0.08	-0.26	-0.28	0.58	0.57	0.34	0.17	0.48	1	0	0.16	0.04	-0.09	0.4	0.56	0.45	0.34	0.35	0.69	-0.25	-0.05	-0.21	-0.23	0.5	0.46	0.35	0.18	0.33	0.77
radius_se	0.68	0.28	0.69	0.73	0.3	0.5	0.63	0.7	0.3	0	1	0.21	0.97	0.95	0.16	0.36	0.33	0.51	0.24	0.23	0.72	0.19	0.72	0.75	0.14	0.29	0.38	0.53	0.09	0.05
texture_se	-0.1	0.39	-0.09	-0.07	0.07	0.05	0.08	0.02	0.13	0.16	0.21	1	0.22	0.11	0.4	0.23	0.19	0.23	0.41	0.28	-0.11	0.41	-0.1	-0.08	-0.07	-0.09	-0.07	-0.12	-0.13	-0.05
perimeter_se	0.67	0.28	0.69	0.73	0.3	0.55	0.66	0.71	0.31	0.04	0.97	0.22	1	0.94	0.15	0.42	0.36	0.56	0.27	0.24	0.7	0.2	0.72	0.73	0.13	0.34	0.42	0.55	0.11	0.09
area_se	0.74	0.26	0.74	0.8	0.25	0.46	0.62	0.69	0.22	-0.09	0.95	0.11	0.94	1	0.08	0.28	0.27	0.42	0.13	0.13	0.76	0.2	0.76	0.81	0.13	0.28	0.39	0.54	0.07	0.02
smoothness_se	-0.22	0.01	-0.2	-0.17	0.33	0.14	0.1	0.03	0.19	0.4	0.16	0.4	0.15	0.08	1	0.34	0.27	0.33	0.41	0.43	-0.23	-0.07	-0.22	-0.18	0.31	-0.06	-0.06	-0.1	-0.11	0.1
compactness_se	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	0.8	0.74	0.39	0.8	0.2	0.14	0.26	0.2	0.23	0.68	0.64	0.48	0.28	0.59
concavity_se	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	0.8	1	0.77	0.31	0.73	0.19	0.1	0.23	0.19	0.17	0.48	0.66	0.44	0.2	0.44
concave points_se	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	0.31	0.61	0.36	0.09	0.39	0.34	0.22	0.45	0.55	0.6	0.14	0.31
symmetry_se	-0.1	0.01	-0.08	-0.07	0.2	0.23	0.18	0.1	0.45	0.35	0.24	0.41	0.27	0.13	0.41	0.39	0.31	0.31	1	0.37	-0.13	-0.08	-0.1	-0.11	-0.01	0.06	0.04	-0.03	0.39	0.08
fractal_dimension_se	-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	0.8	0.73	0.61	0.37	1	-0.04	-0	-0	-0.02	0.17	0.39	0.38	0.22	0.11	0.59
radius_worst	0.97	0.35	0.97	0.96	0.21	0.54	0.69	0.83	0.19	-0.25	0.72	-0.11	0.7	0.76	-0.23	0.2	0.19	0.36	-0.13	-0.04	1	0.36	0.99	0.98	0.22	0.48	0.57	0.79	0.24	0.09
texture_worst	0.3	0.91	0.3	0.29	0.04	0.25	0.3	0.29	0.09	-0.05	0.19	0.41	0.2	0.2	-0.07	0.14	0.1	0.09	-0.08	-0	0.36	1	0.37	0.35	0.23	0.36	0.37	0.36	0.23	0.22
perimeter_worst	0.97	0.36	0.97	0.96	0.24	0.59	0.73	0.86	0.22	-0.21	0.72	-0.1	0.72	0.76	-0.22	0.26	0.23	0.39	-0.1	-0	0.99	0.37	1	0.98	0.24	0.53	0.62	0.82	0.27	0.14
area_worst	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	0.2	0.19	0.34	-0.11	-0.02	0.98	0.35	0.98	1	0.21	0.44	0.54	0.75	0.21	0.08
smoothness_worst	0.12	0.08	0.15	0.12	0.81	0.57	0.45	0.45	0.43	0.5	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	0.57	0.52	0.55	0.49	0.62
compactness_worst	0.41	0.28	0.46	0.39	0.47	0.87	0.75	0.67	0.47	0.46	0.29	-0.09	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	0.89	0.8	0.61	0.81
concavity_worst	0.53	0.3	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	0.86	0.53	0.69
concave points_worst	0.74	0.3	0.77	0.72	0.5	0.82	0.86	0.91	0.43	0.18	0.53	-0.12	0.55	0.54	-0.1	0.48	0.44	0.6	-0.03	0.22	0.79	0.36	0.82	0.75	0.55	0.8	0.86	1	0.5	0.51
symmetry_worst	0.16	0.11	0.19	0.14	0.39	0.51	0.41	0.38	0.7	0.33	0.09	-0.13	0.11	0.07	-0.11	0.28	0.2	0.14	0.39	0.11	0.24	0.23	0.27	0.21	0.49	0.61	0.53	0.5	1	0.54
fractal_dimension_worst	0.01	0.12	0.05	0	0.5	0.69	0.51	0.37	0.44	0.77	0.05	-0.05	0.09	0.02	0.1	0.59	0.44	0.31	0.08	0.59	0.09	0.22	0.14	0.08	0.62	0.81	0.69	0.51	0.54	1

Gaussian Naïve Bayes:



```
model = GaussianNB()
model.fit(X_train, y_train)
```

```
GaussianNB()
```

```
predict = model.predict(X_test)
predict
```

```
array(['B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'B',
       'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'M',
       'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B',
       'B', 'M', 'B', 'M', 'M', 'B', 'B', 'M', 'M', 'B', 'M', 'B', 'M',
       'M', 'M', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M',
       'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'M', 'M', 'B',
       'M', 'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M',
       'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'M',
       'M', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'M', 'M', 'B', 'M', 'B',
       'B', 'B'], dtype='<U1')
```

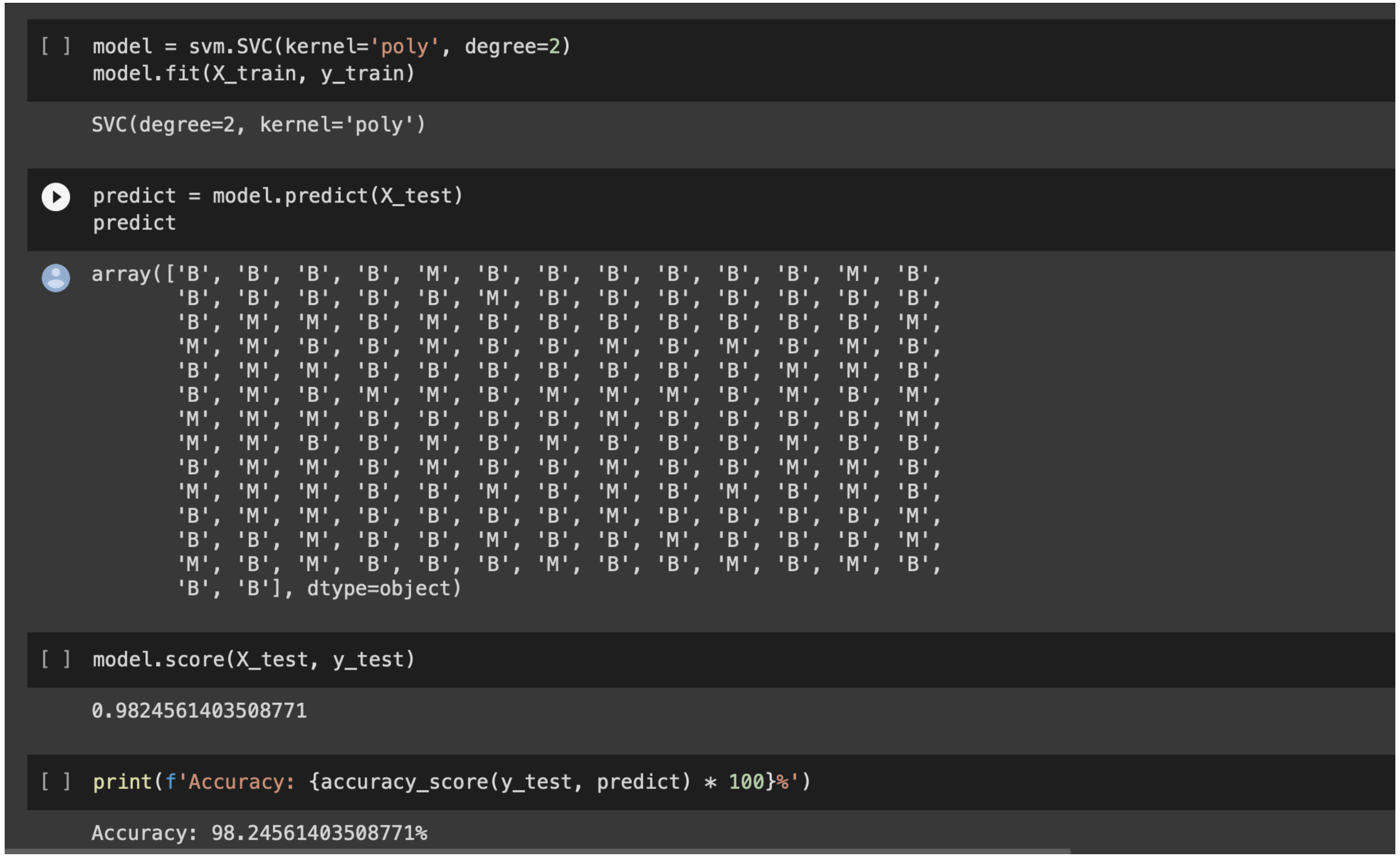
```
model.score(X_test, y_test)
```

```
0.9415204678362573
```

```
print(f'Accuracy: {accuracy_score(y_test, predict) * 100}%')
```

```
Accuracy: 94.15204678362574%
```

Support Vector Machine(SVM):



```
model = svm.SVC(kernel='poly', degree=2)
model.fit(X_train, y_train)
```

```
SVC(degree=2, kernel='poly')
```

```
predict = model.predict(X_test)
predict
```

```
array(['B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'B',
       'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'M',
       'M', 'M', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'M', 'B',
       'B', 'M', 'B', 'M', 'M', 'B', 'M', 'M', 'M', 'B', 'M', 'B', 'M',
       'M', 'M', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M',
       'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'M', 'M', 'B',
       'M', 'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M',
       'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'M',
       'M', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'M', 'B',
       'B', 'B'], dtype=object)
```

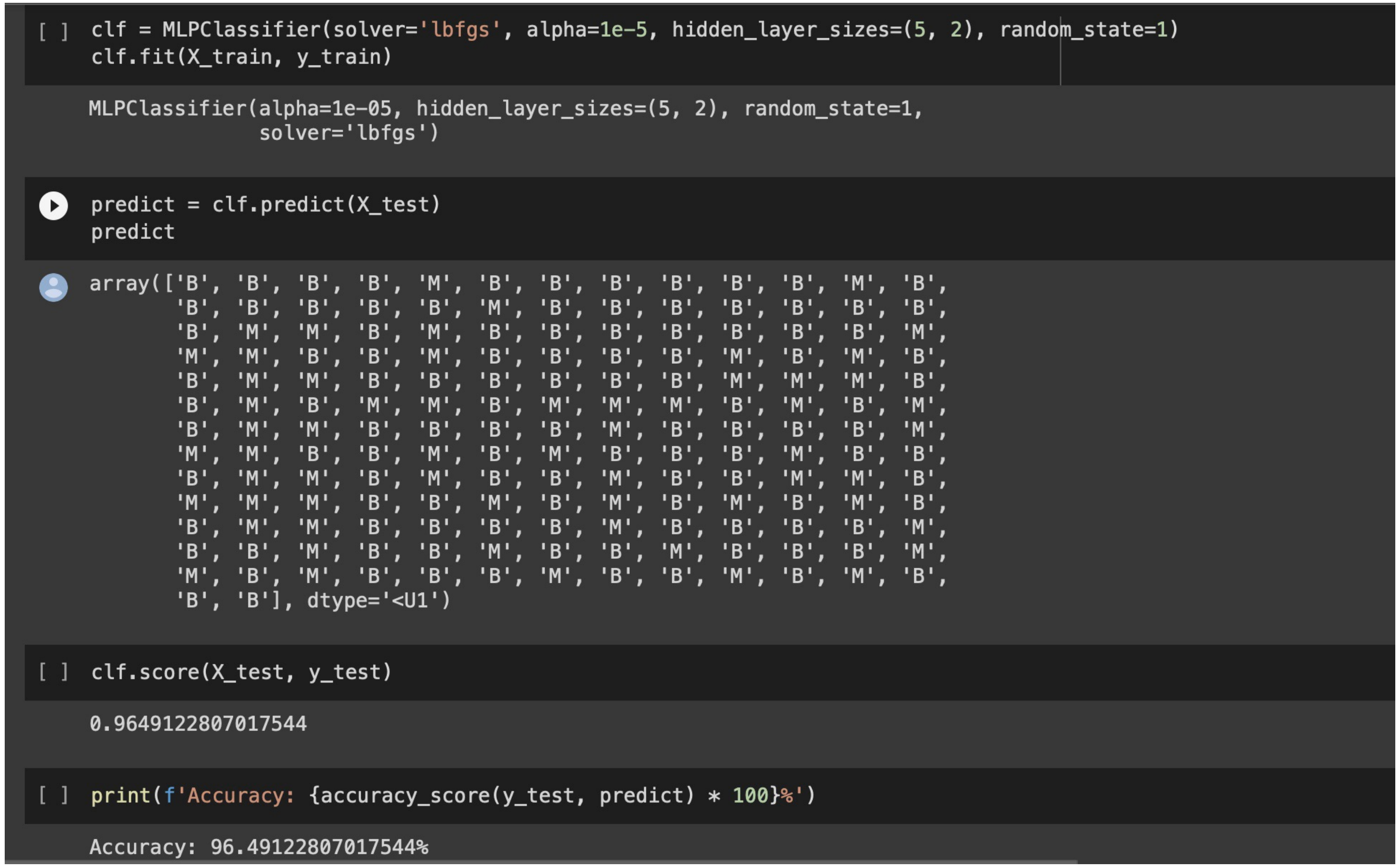
```
model.score(X_test, y_test)
```

```
0.9824561403508771
```

```
print(f'Accuracy: {accuracy_score(y_test, predict) * 100}%')
```

```
Accuracy: 98.24561403508771%
```

Neural Network:



```
[ ] clf = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2), random_state=1)
    clf.fit(X_train, y_train)

    MLPClassifier(alpha=1e-05, hidden_layer_sizes=(5, 2), random_state=1,
                  solver='lbfgs')
```

```
predict = clf.predict(X_test)
predict

array(['B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'B',
       'B', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'B', 'M',
       'M', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'B', 'B', 'B', 'B', 'B', 'M', 'M', 'M', 'B',
       'B', 'M', 'B', 'M', 'M', 'B', 'M', 'M', 'M', 'B', 'M', 'B', 'M',
       'B', 'M', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M',
       'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B',
       'B', 'M', 'M', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'M', 'M', 'B',
       'M', 'M', 'M', 'B', 'B', 'M', 'B', 'M', 'B', 'M', 'B', 'M', 'B',
       'B', 'M', 'M', 'B', 'B', 'B', 'B', 'M', 'B', 'B', 'B', 'B', 'M',
       'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'B', 'B', 'M',
       'M', 'B', 'M', 'B', 'B', 'B', 'M', 'B', 'B', 'M', 'B', 'M', 'B',
       'B', 'B'], dtype='<U1')
```

```
[ ] clf.score(X_test, y_test)

    0.9649122807017544
```

```
[ ] print(f'Accuracy: {accuracy_score(y_test, predict) * 100}%')

    Accuracy: 96.49122807017544%
```

Results

- Using the Naïve Bayes algorithm, a prediction accuracy of 94% was obtained.
- Using SVM, the prediction accuracy was 98%.
- Using Neural Networks, 96% accuracy was obtained.
- By comparison, SVM yields the best prediction accuracy.

Advantages over existing methods:

- Economic and faster diagnosis of Breast cancer.
- Non-invasive diagnosis using Gaussian Naive Bayes model.

Incremental Features:

- This project can be further expanded to diagnose other health conditions when the relevant data is available.
- Multiple Machine learning models can be used to compare and improve the diagnosis.
- Implement validation set to tweak the hyper parameters and check correct value for the model.

References:

- Base paper for the project: https://ieeexplore-ieee-org.libproxy.library.unt.edu/document/9754059
- Breast Cancer Wisconsin (Diagnostic) Data Set - UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
- Breast Cancer Prediction Using Machine Learning - Towards Data Science: https://towardsdatascience.com/breast-cancer-prediction-using-machine-learning-7e83c0b37216
- Breast Cancer Detection Using Machine Learning Techniques: A Review - International Journal of Intelligent Systems and Applications: https://www.researchgate.net/publication/321709047_Breast_Cancer_Detection_Using_Machine_Learning_Techniques_A_Review
- Breast Cancer Diagnosis Using Machine Learning Techniques: A Systematic Review - Applied Sciences: https://www.mdpi.com/2076-3417/10/10/3464
- Breast Cancer Detection Using Machine Learning Algorithms: A Systematic Review - Journal of Medical Systems: https://link.springer.com/article/10.1007/s10916-021-01778-7