TITLE:

Machine Learning Model Development and Evaluation using Python

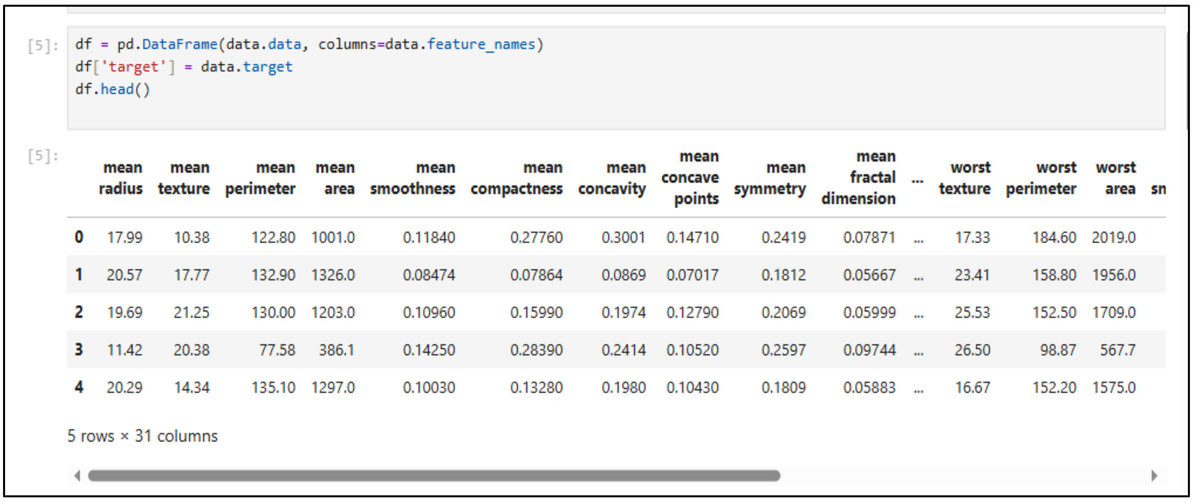
Tools Used: Python, Pandas, Scikit-learn, Matplotlib, Seaborn

1. Introduction

This project focuses on building and evaluating a basic machine learning model using Python. The objective is to understand the complete machine learning workflow, including data preparation, model selection, training, and evaluation. A supervised learning approach was used to predict outcomes based on labelled data, and model performance was assessed using standard evaluation metrics and visualizations.

2. Dataset Description

The dataset used in this project is the Breast Cancer Wisconsin dataset, provided by the scikit-learn library. The dataset contains numerical features extracted from medical images and a target variable indicating whether a tumour is benign or malignant. This dataset is widely used for introductory machine learning and classification tasks.



```
[5]: df = pd.DataFrame(data.data, columns=data.feature_names)
     df['target'] = data.target
     df.head()
```

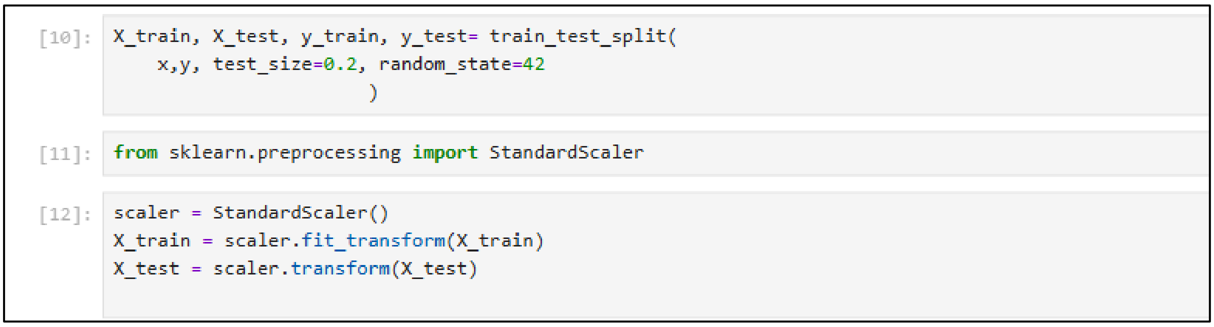
[5]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	sn
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	

5 rows × 31 columns

3. Data Preparation

The dataset was converted into a pandas DataFrame for analysis. The input features and target variable were separated. The data was split into training and testing sets to evaluate the model on unseen data. Feature scaling was applied to ensure that all variables contributed equally to the model.



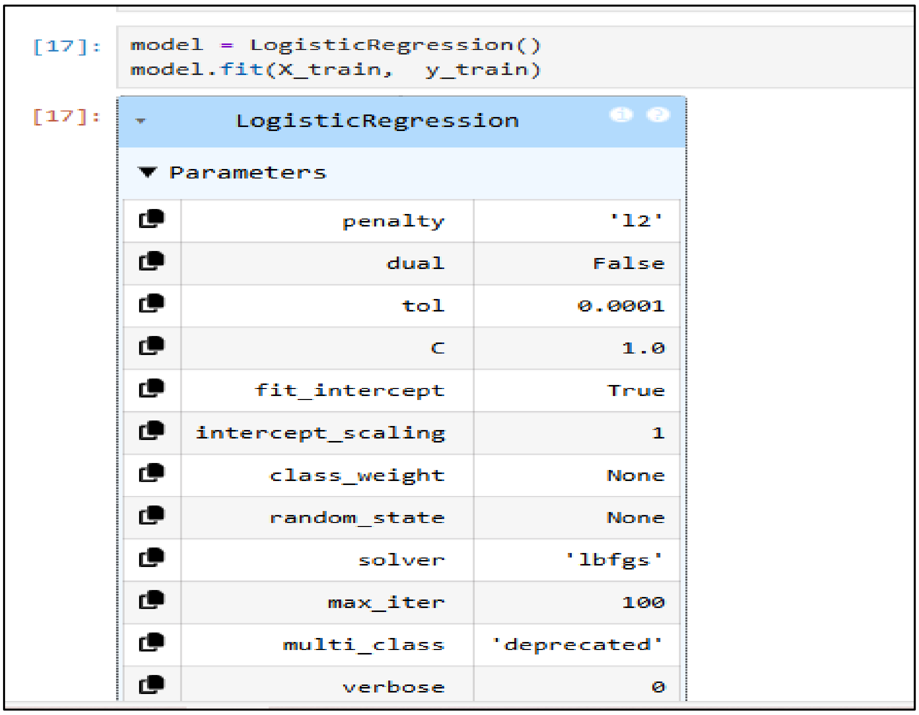
```
[10]: X_train, X_test, y_train, y_test= train_test_split(
          x,y, test_size=0.2, random_state=42
                               )

[11]: from sklearn.preprocessing import StandardScaler

[12]: scaler = StandardScaler()
      X_train = scaler.fit_transform(X_train)
      X_test = scaler.transform(X_test)
```

4. Model Selection and Training

Logistic Regression was selected as the machine learning algorithm for this task. It is a simple and interpretable classification model suitable for binary outcomes. The model was trained using the training dataset to learn patterns between input features and the target variable.



```
[17]: model = LogisticRegression()
      model.fit(X_train,  y_train)

[17]:  LogisticRegression
       ▼ Parameters
                  penalty          '12'
                     dual         False
                      tol        0.0001
                        C           1.0
            fit_intercept          True
        intercept_scaling             1
             class_weight          None
             random_state          None
                   solver       'lbfgs'
                 max_iter           100
              multi_class  'deprecated'
                  verbose             0
```

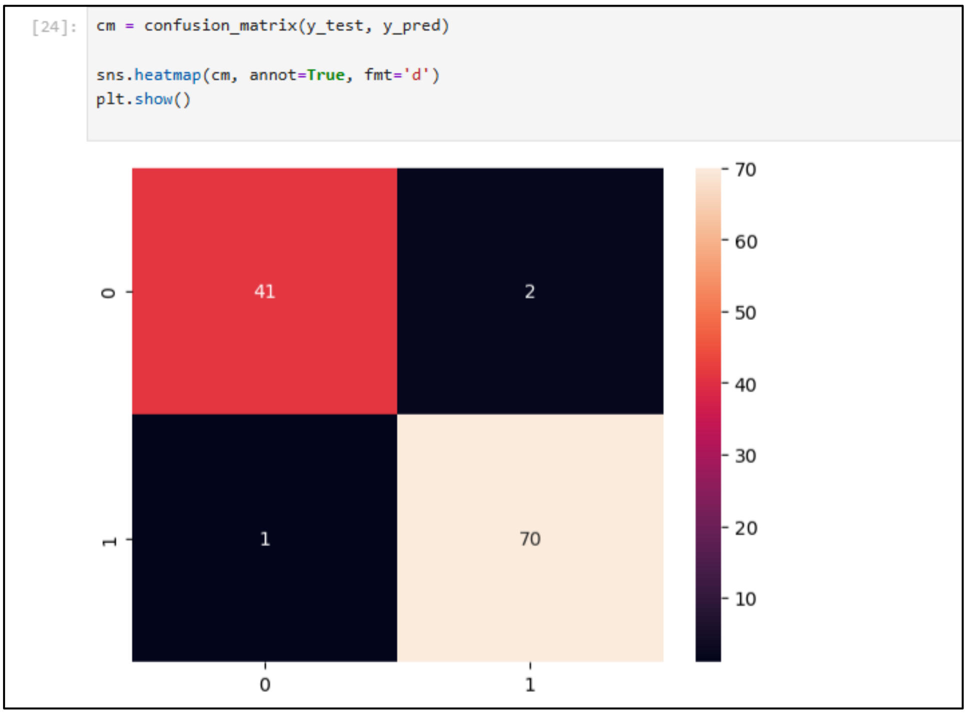
5. Model Evaluation

The trained model was evaluated using accuracy and classification metrics. Predictions were made on the test dataset, and performance was measured to assess how well the model generalizes to new data.

6. Performance Visualizations

6.1 Confusion Matrix

A confusion matrix was used to visualize correct and incorrect predictions made by the model.

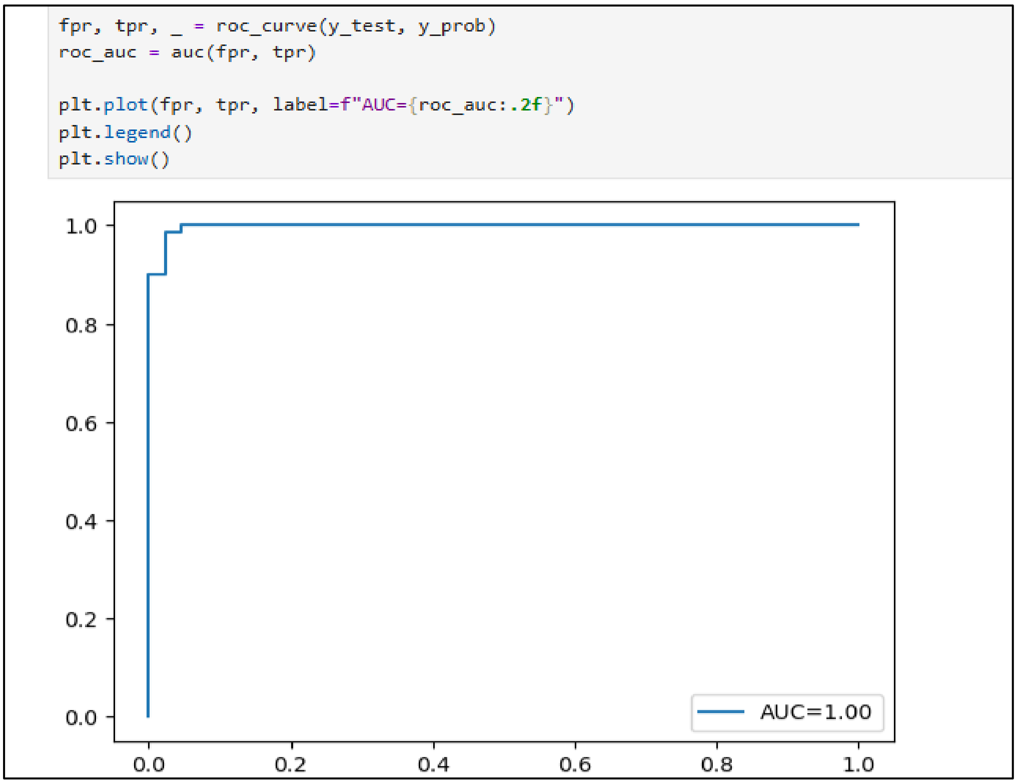


```
[24]: cm = confusion_matrix(y_test, y_pred)

      sns.heatmap(cm, annot=True, fmt='d')
      plt.show()
```

6.2 ROC Curve

A Receiver Operating Characteristic (ROC) curve was plotted to evaluate the model's performance across different classification thresholds.



```
fpr, tpr, _ = roc_curve(y_test, y_prob)
roc_auc = auc(fpr, tpr)

plt.plot(fpr, tpr, label=f"AUC={roc_auc:.2f}")
plt.legend()
plt.show()
```

7. Error Analysis and Model Improvements

Potential sources of error include overfitting, feature correlation, and model assumptions. Logistic Regression assumes linear relationships between features and the target variable. Model performance could be improved by using advanced algorithms such as Decision Trees or Random Forests, applying cross-validation, and tuning hyperparameters.

---

8. Conclusion

This project demonstrates the fundamental steps involved in developing and evaluating a machine learning model. By following a structured workflow and applying appropriate evaluation techniques, meaningful insights were obtained. This exercise provides a strong foundation for understanding supervised machine learning and model evaluation.