Challenge 1: Cancer Detection

Name: **María Paloma Prado Durán**

student code: **224807991**

Email: maria.prado0799@alumnos.udg.mx

Programación II

March 24, 2025

Dataset Extraction.

To start the work, the first step is to load the information with which we will work. To do this, the *load_data_frame* function was created, which allows us to read the CSV file and assign it under the name df. Once this is done, we proceed to use the *preprocessing* function, which eliminates unnecessary columns, defines the independent variables X and the dependent one y, finally, scales the data.

It's important to mention that the dataset to be loaded will be breast-cancer-wisconsin.data.csv, which contains information on patients with a positive or negative diagnosis of cancer. It contains 30 features derived from digital images of breast tumors. These features are grouped into three types of measures:

- *Mean measures (mean):* Average value of the feature in the image.
- *Standard errors (se):* Standard error of the feature.
- *Worst value measurements (worst):* Worst (highest) value observed in the image.

Each of these measures applies to the following 10 properties of the cell nucleus:

- Radius: Average radius of the cell nucleus.
- Texture: Variation in the texture of the cell nuclei (standard deviation of grayscale values).
- Perimeter: Perimeter of the cell nucleus.
- Area: Area occupied by the cell nucleus.
- Smoothness: Uniformity of the contours (local change in the radius length).
- Compactness: Ratio of perimeter to area (perimetro^2/area -1)
- Concavity: Depth of the concave parts of the nucleus contour.
- Concave points: Number of concave points in the nucleus contour.
- Symmetry: Symmetry of the cell nucleus.
- Fractal dimension: Measure of the complexity of the nucleus contour (ratio of changes in scale to contour detail).

Model Construction.

Once we know the information with which we will work, we can proceed to build a model that allows us to make predictions in the diagnosis of patients, for this through the *model_training* function that will take as arguments the independent variables X and the dependent y defined above, so as a first step the information will be divided into testing and training in a proportion of 20% - 80%, once this is done, a RandomForestClassifier model is initialized, which is a commonly used machine learning algorithm, registered by Leo Breiman and Adele Cutler, which combines the result of multiple decision trees to reach a single result[1]. This model has the following advantages for information management:

- *Lower risk of overfitting:* Decision trees are at risk of overfitting, as they tend to closely fit all samples within the training data. However, when there are a large number of decision trees in a random forest, the classifier will not overfit the model, as the averaging of uncorrelated trees reduces the overall variance and prediction error.
- *Provides flexibility:* Since the random forest can handle regression and classification tasks with a high degree of accuracy, it is a popular method among data scientists. Feature bagging also makes the random forest classifier an effective tool for estimating missing values, as it maintains accuracy when a portion of the data is missing.
- *Ease of determining feature importance:* Random forest makes it easier to assess the importance, or contribution, of variables to the model. There are a few ways to assess feature importance. Gini importance and mean decay accuracy (MDI) are commonly used to measure how much a model's accuracy decreases when a given variable is excluded. However, permutation importance, also known as mean decay accuracy (MDA), is

[1] IBM. (n.d.). What is Random Forest? Retrieved March 24, 2025, from https://www.ibm.com/mx-es/think/topics/random-forest

another importance measure. MDA identifies the average decrease in accuracy by randomly permuting feature values across OOB samples.

Once the model is initialized, it will be trained with the training information to subsequently generate predictions with the test information, so that through the *model_evaluate* function important metrics are calculated that will provide us with information about how well the model is classifying, the metrics to be calculated are:

- *accuracy:* Measures the proportion of correct predictions out of the total number of predictions made, i.e., whether the model has good overall performance or not.
- *precision:* This measure tells us how many times the model actually predicted malignancy. A high value indicates that the model makes few false positives (FPs), meaning that few benign cases were misclassified as malignant. This is useful when the cost of a false positive is high (for example, if an unnecessary biopsy is expensive or risky).
- *recall:* It measures how many of all the truly malignant cases the model correctly identified. A high value indicates that the model makes few false negatives (FNs), meaning that few malignant tumors were misclassified as benign. This is key in medical problems where failing to detect a malignant case can be dangerous (as is the case with our dataset).
- F1-score: This is the harmonic average of Precision and Recall, giving a single value that balances both aspects. A high value indicates that the model has a good balance between avoiding false positives and false negatives.

Likewise, the confusion matrix will be graphed, which, using a 2x2 table, shows how many of the model's predictions were correct and incorrect in each category, in our case detecting cancer or not.

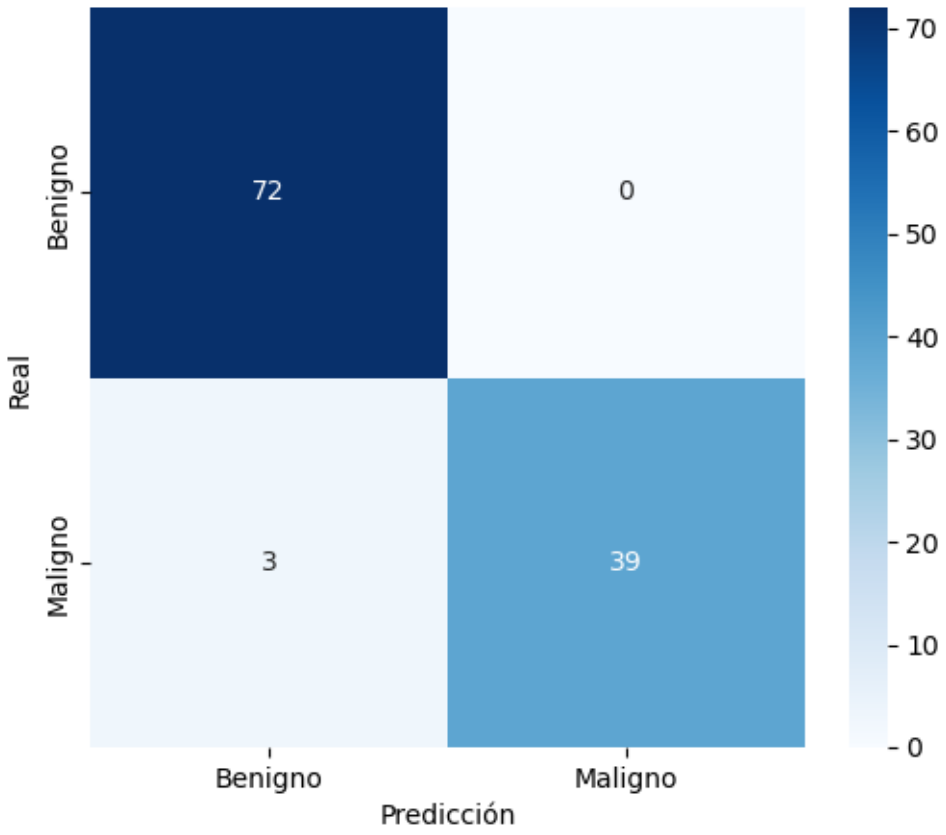

Image 1. Confusion matrix RandomForestClassifier.

The ROC Curve shows the model's performance for different classification thresholds, that is, how the ratio of true positives (TPR) to false positives (FPR) changes. An *auc_score* greater than 0.9 indicates a very good model.

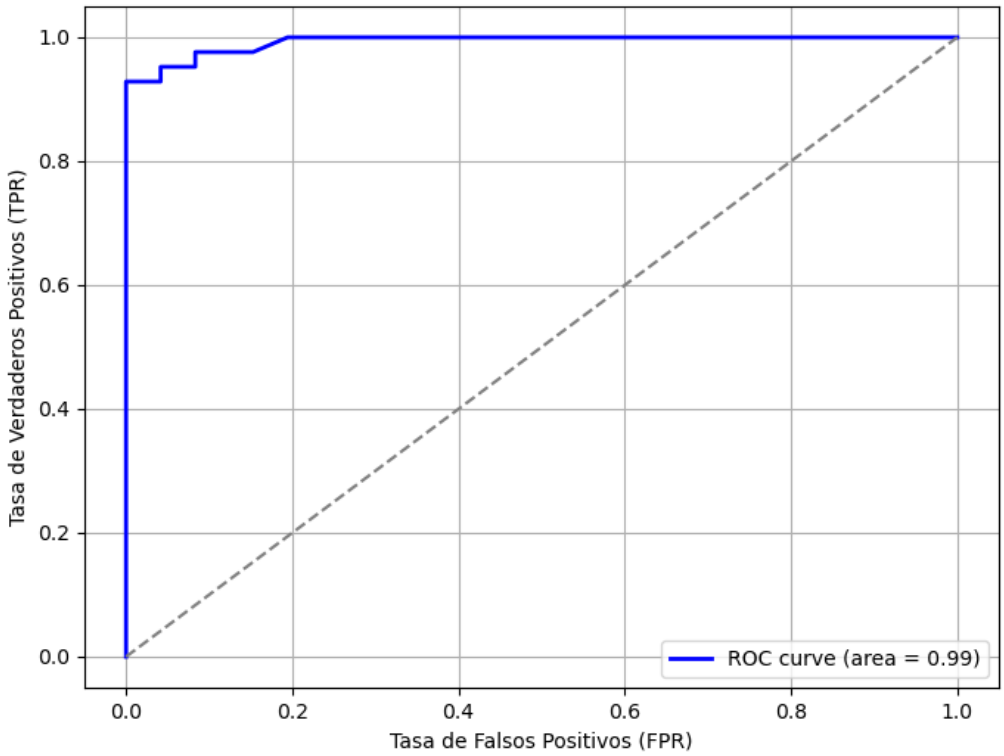

Image 2. ROC Curve, RandomForestClassifier

It is important to mention that these two plots will be saved in a folder called plots as a .png image.

Finally, it generates a classification report summarizing the performance of a classification model on a test dataset. The report includes several important evaluation metrics used to measure the effectiveness of a classification model, such as precision, recall, f1-score, etc. The report will be saved as a .txt file.

MLOps.

Once the metrics have been calculated, we want to save them. Therefore, in the ***mlops_pipline.py*** script, which contains the aforementioned functions, we will use the mlflow library. This is an open-source platform purpose-built to assist machine learning practitioners and teams in handling the complexities of the machine learning

process. MLflow[2] focuses on the full lifecycle for machine learning projects, ensuring that each phase is manageable, traceable, and reproducible. The following functions will be used in this challenge:

- *set_tracking:* Defines the location where MLflow will store experiments and models.
- *set_experiment:* Defines or creates an experiment in MLflow to organize runs.
- *log_metric:* Saves numerical model metrics, such as precision, recall, and F1 score.
- *log_text:* Saves a text file in the MLflow log.
- *log_artifact:* Saves files such as images, graphs, or reports in MLflow. In this case, the confusion matrix and the generated and saved ROC curve will be saved.
- *log_model:* Saves the trained model in MLflow for future predictions or deployments.

Execution Guide.

To run the script, the user is encouraged to follow these steps:

1. Clone the repository https://github.com/Paloma-PD/Challenges-Progra2.git using a git clone command.
2. Install the minimum dependencies for execution using the *requirements.txt* file.
3. Run the ***mlops_pipline.py*** script, which contains all the functions.
4. Run the *upload_results.py* script to update the results/changes directly on GitHub.

It is important to mention that to run the scripts, you must be in the directory where they are located, if you run them from the terminal or git bash.

** For more details on execution, consult the repository's README file.

[2] mlflow: https://pypi.org/project/mlflow/

Extra: Architecture diagram.

As support, the project architecture diagram is included

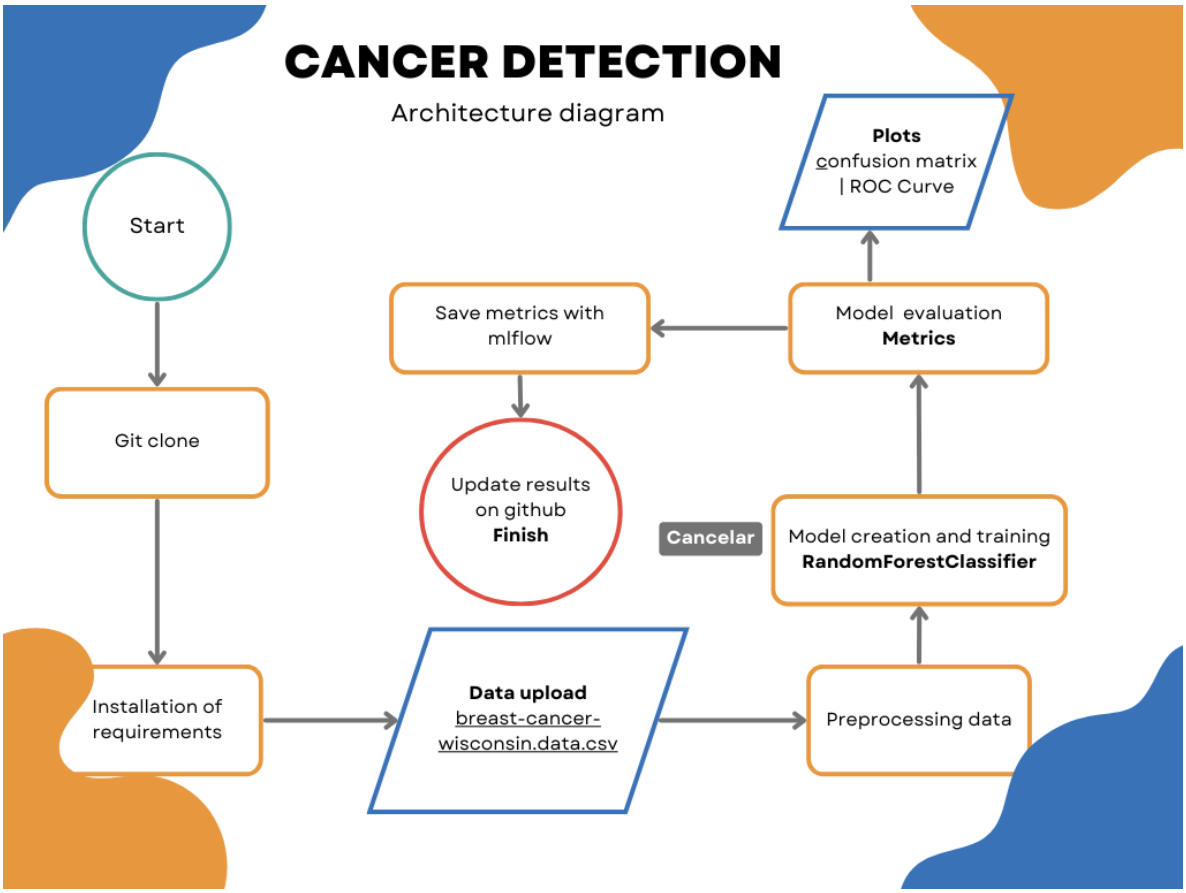

Image 3. Architecture diagram (Canva.com)