Faculty of Engineering
Cairo University

Machine Learning

Project Report Team 9

Supervised by: Dr. Dina Elreedy

Name	Section	B.N
Ahmed Ihab	1	2
Mostafa Elgendy	2	28
Weam Bassem	2	36
Yousef Ahmed	2	38

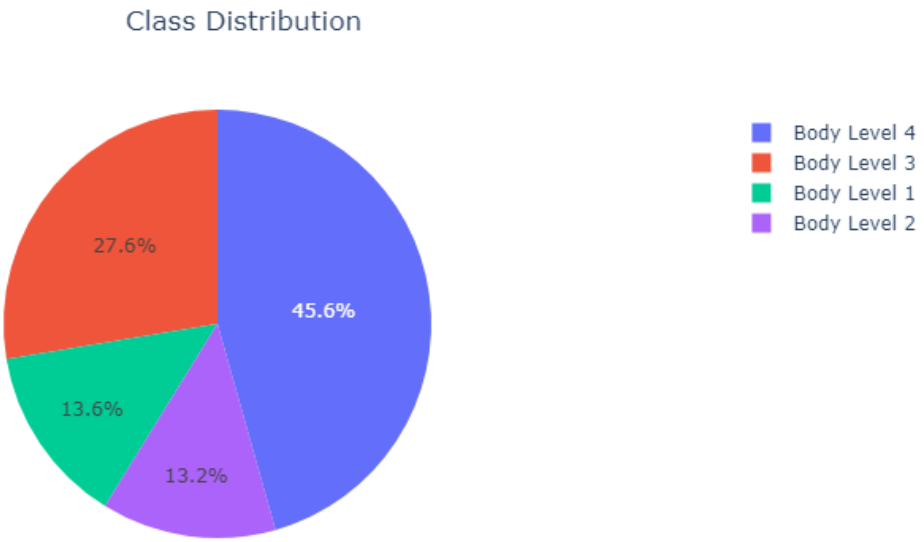
- Workload Division

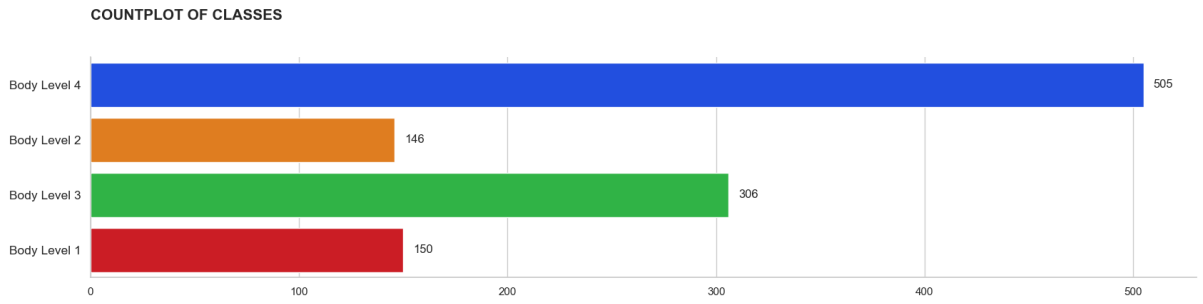
Name	Modules
Ahmed Ihab	- SVM - Data Analysis
Mostafa Elgendy	- Decision Tree - Preprocessing
Weam Bassem	- Logistic Regression - Preprocessing
Yousef Ahmed	- Random Forest - Baseline Models

1. Data Analysis & Preprocessing

The provided dataset is initially split into train and test with ratios of 0.75 & 0.25 respectively. This is done to ensure that the generalization ability of the implemented models is a good approximation to its generalization ability on the real test set then the following analysis is done on the training dataset.

The following plots show that there is a target class imbalance in the training dataset.





All the continuous attributes are standardized (normalized) using StandardScaler().

The categorical attributes' details are shown in the following table.

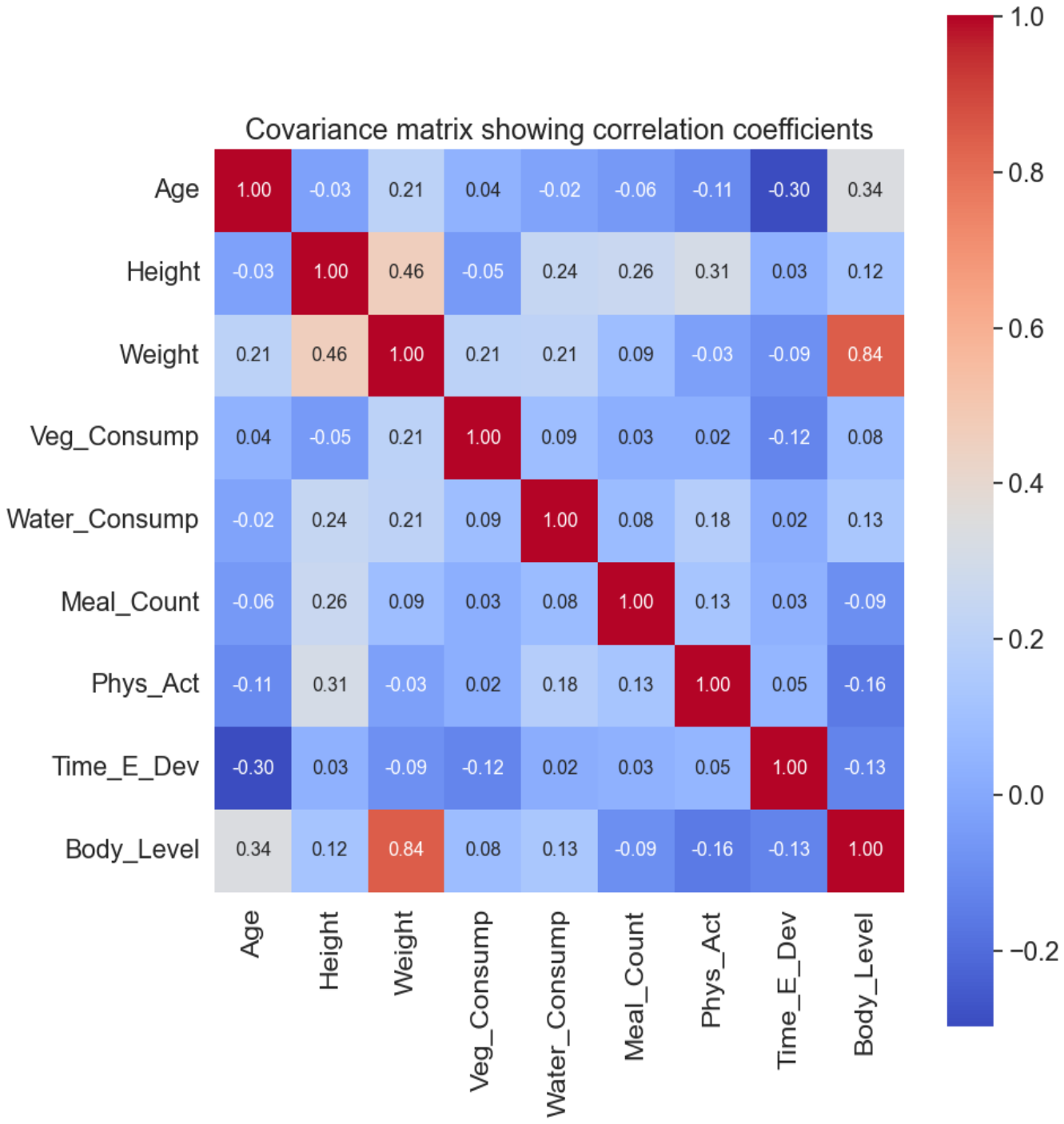
	Attribute	Unique Values
0	Gender	[Male, Female]
1	H_Cal_Consump	[yes, no]
2	Alcohol_Consump	[Sometimes, no, Frequently, Always]
3	Smoking	[no, yes]
4	Food_Between_Meals	[Sometimes, no, Frequently, Always]
5	Fam_Hist	[yes, no]
6	H_Cal_Burn	[no, yes]
7	Transport	[Public_Transportation, Automobile, Bike, Moto...

Those attributes can be classified and represented as follows according to the four levels of data.

Attributes	**Level of Data**	**Representation**
Gender, Transport	Nominal	One-hot encoded
Others	Ordinal	Label encoded

2. Feature Engineering

The following plot shows the covariance matrix between all continuous features (after standardization) and the target label (Body_Level).



Outlier analysis was performed based on the interquartile distance of each continuous variable. An attribute value (x) is considered to be an outlier if one of the following conditions is satisfied:

- $x > Q_3 + 3 * IQD$
- $x < Q_1 - 3 * IQD$

The analysis results are shown in the following table:

	Attribute	Number of Outliers	Percentage from Dataset
0	Age	8	0.722674
1	Height	0	0.000000
2	Weight	0	0.000000
3	Veg_Consump	0	0.000000
4	Water_Consump	0	0.000000
5	Meal_Count	177	15.989160
6	Phys_Act	0	0.000000
7	Time_E_Dev	0	0.000000

From the previous analysis results, we can conclude that the ‘Meal_Count’ attribute has many outlier values and it has a very low correlation with the target variable. So, this attribute was removed from the dataset to not affect the results of the classification task.

3. Models

1. baseline

Three baseline models are implemented which are:

1. Most frequent (ZeroR): It always returns the most frequent class label in the training data.
2. Uniform: It generates predictions uniformly at random from the list of unique classes observed in training data, i.e. each class has equal probability.
3. Constant: It always predicts a constant label that is provided by the user. This is useful for metrics that evaluate a non-majority class.

Strategy: most_frequent

	precision	recall	f1-score	support
1	0.00	0.00	0.00	40
2	0.00	0.00	0.00	55
3	0.00	0.00	0.00	100
4	0.47	1.00	0.64	175
accuracy			0.47	370
macro avg	0.12	0.25	0.16	370
weighted avg	0.22	0.47	0.30	370

Strategy: uniform

	precision	recall	f1-score	support
1	0.13	0.30	0.18	40
2	0.12	0.18	0.14	55
3	0.25	0.25	0.25	100
4	0.41	0.22	0.28	175
accuracy			0.23	370
macro avg	0.23	0.24	0.21	370
weighted avg	0.29	0.23	0.24	370

Strategy: constant

	precision	recall	f1-score	support
1	0.11	1.00	0.20	40
2	0.00	0.00	0.00	55
3	0.00	0.00	0.00	100
4	0.00	0.00	0.00	175
accuracy			0.11	370
macro avg	0.03	0.25	0.05	370
weighted avg	0.01	0.11	0.02	370

2. Logistic Regression

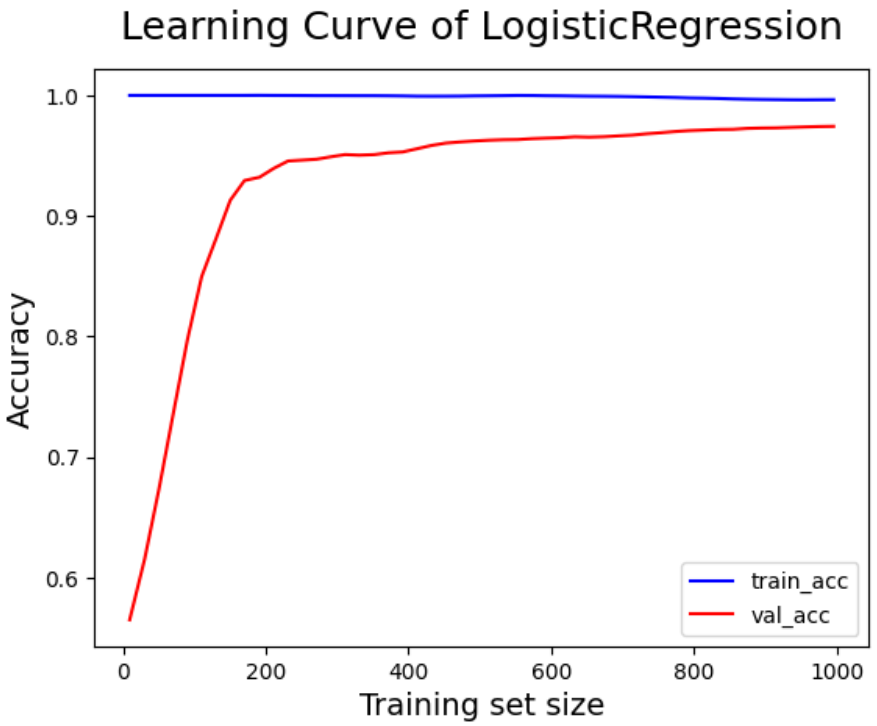
Parameter Tuning

In the Logistic Regression model, several hyperparameters can be tuned. we choose to tune **penalty, C, and solver.**

Penalty: determines the type of regularization to be used in the logistic regression model (L1 / L2)
C: controls the strength of the regularization in the model. A smaller value of C results in stronger regularization
Solver: determines the algorithm to be used for optimization in the logistic regression model

After using GridSearchCV after passing to it all possible values for parameters it told us that the best parameters are {'C': 100, 'penalty': 'l2', 'solver': 'lbfgs'} **Best score:** 0.9882563161716929

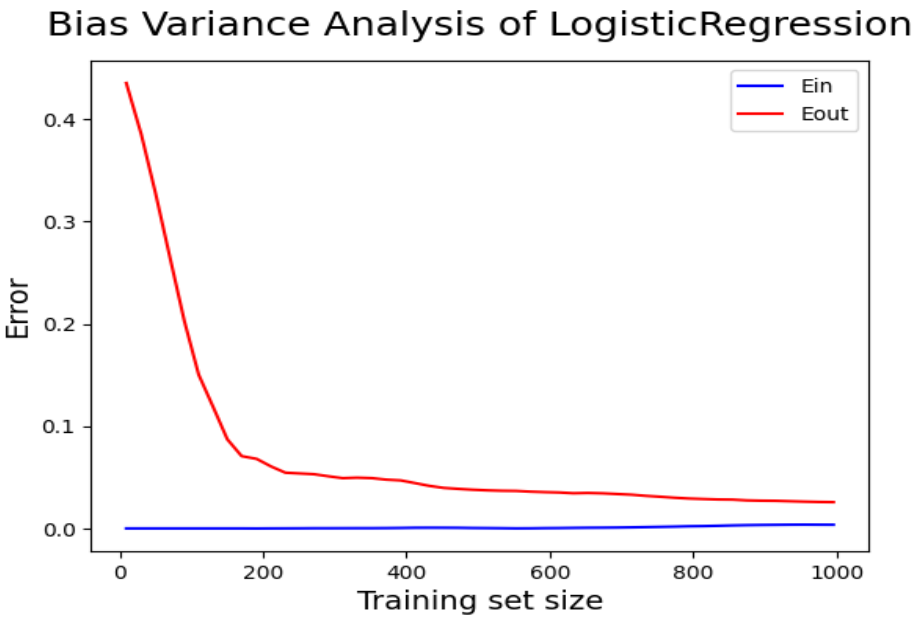
Learning Analysis



- This curve shows how the performance of a model improves with increasing amounts of training data.
- As we can see the training and validation scores are converging at high values as the number of training examples increases and that means the model neither

overfits nor underfitting the data, and it generalizes well to unseen data

Bias-Variance Analysis



Conclusions:

- It is a **complex model**, as Eout changes rapidly and the final **Eout is small**
- Eout decreases as the variance decreases and Ein increases as it becomes harder to fit the points
- Eout starts much higher for the complex model as the number of points initially to the small and thus large variance
- Ein and Eout become the same as N tends to infinity due to **Hoeffding's inequality**

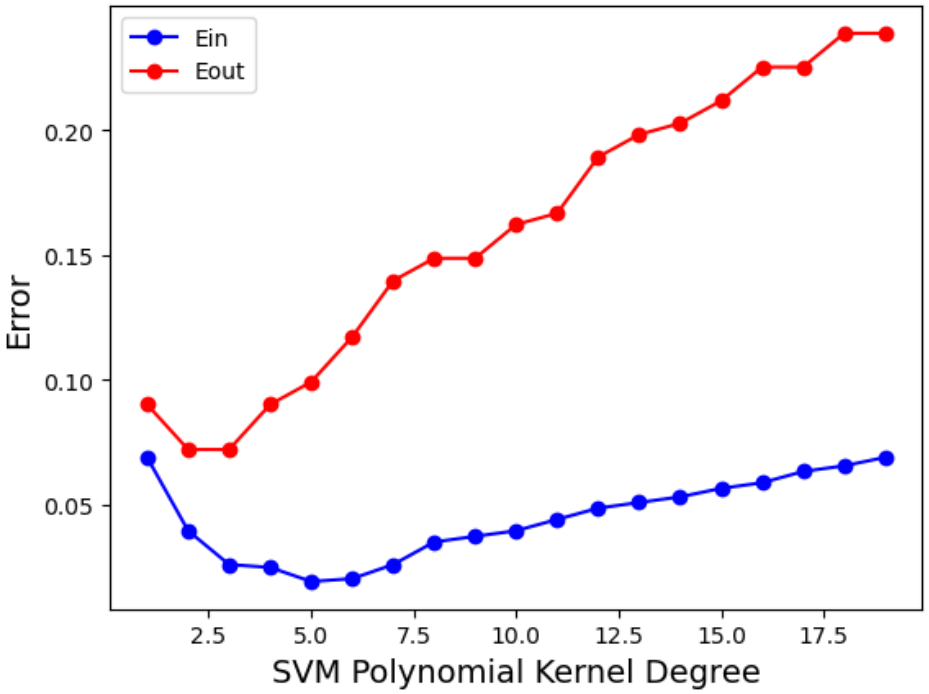
Model Evaluation

	precision	recall	f1-score	support
1	0.85	1.00	0.92	40
2	0.91	0.87	0.89	55
3	0.99	0.95	0.97	100
4	1.00	0.99	1.00	175
accuracy			0.96	370
macro avg	0.94	0.95	0.94	370
weighted avg	0.97	0.96	0.97	370

3. SVM

Parameter Tuning

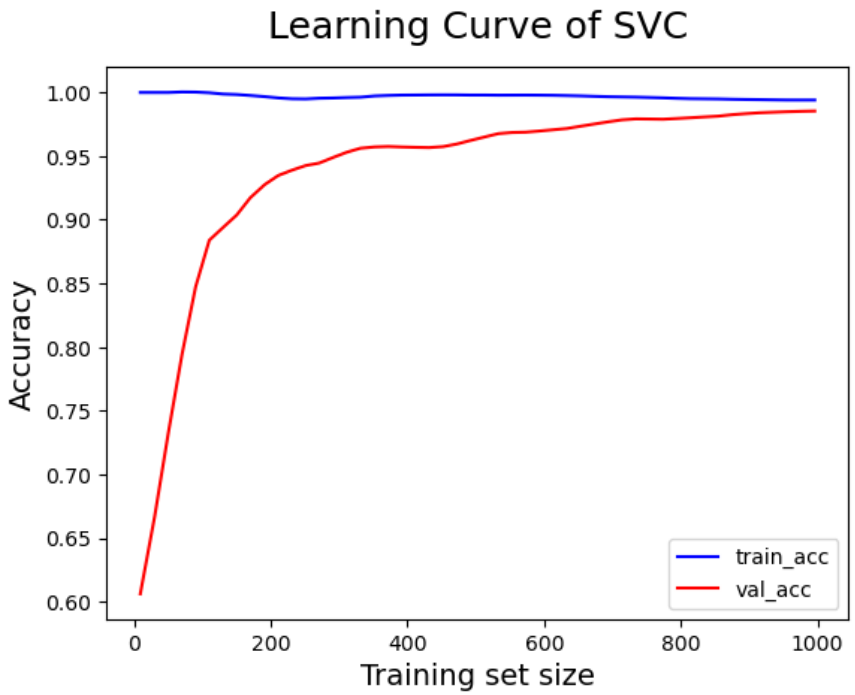
- We have made our parameter tuning by setting the kernel to **poly** and trying values for the degree from 1 to 20 and here are the results



- The degree of 3 was the best point as shown in the graph

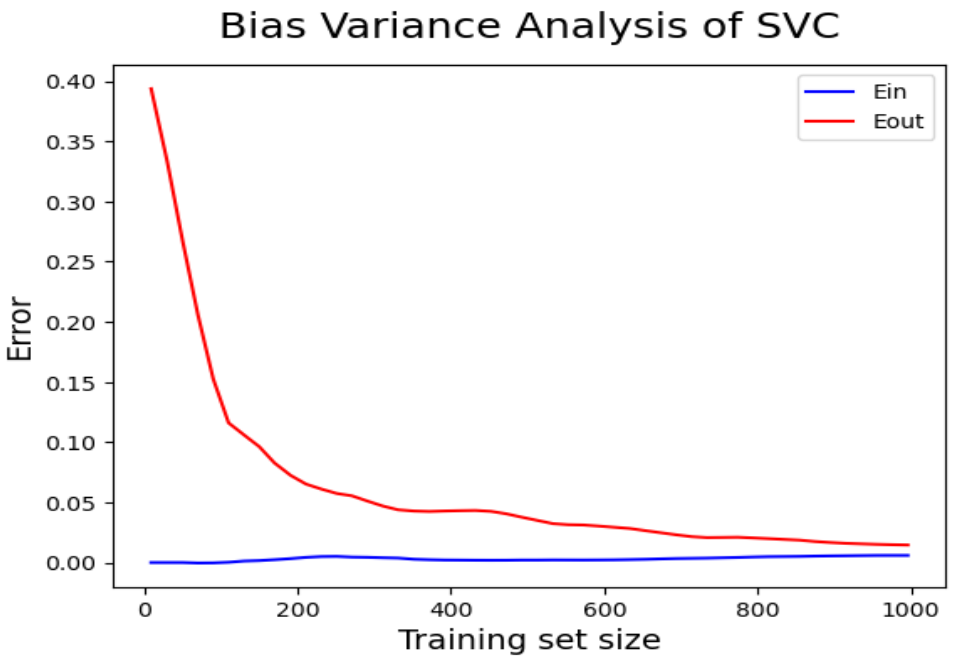
- Then we tried the **GridSearchCV** to see what are the best values for most of the parameters
  C: Controls the tradeoff between maximizing the margin and minimizing the classification error. A higher value of C means the classifier will prioritize classification accuracy over the margin width
  Kernel: Transform the input data into a higher-dimensional feature space where a linear boundary can be used to separate the data.
  Gamma: It controls the shape of the decision boundary and how tightly the algorithm fits the data. A small gamma value means the decision boundary is relatively smooth
  Degree: This parameter determines the degree of the polynomial function used in the polynomial kernel
  It gives us that the best parameters to use are **kernel='linear', degree=1, gamma=1, C=10.**
  However the degree parameter is useless in this case because the kernel is linear.

Learning Analysis



- This curve shows how the performance of a model improves with increasing amounts of training data.
- As we can see the training and validation scores are converging at high values as the number of training examples increases and that means the model neither overfits nor underfitting the data, and it generalizes well to unseen data

Bias-Variance Analysis



Conclusions:

- It is a **complex model**, as Eout changes rapidly and the final **Eout is small**
- Eout decreases as the variance decreases and Ein increases as it becomes harder to fit the points
- Eout starts much higher for the complex model as the number of points initially to the small and thus large variance
- Ein and Eout become the same as N tends to infinity due to **Hoeffding's inequality**

Model Evaluation

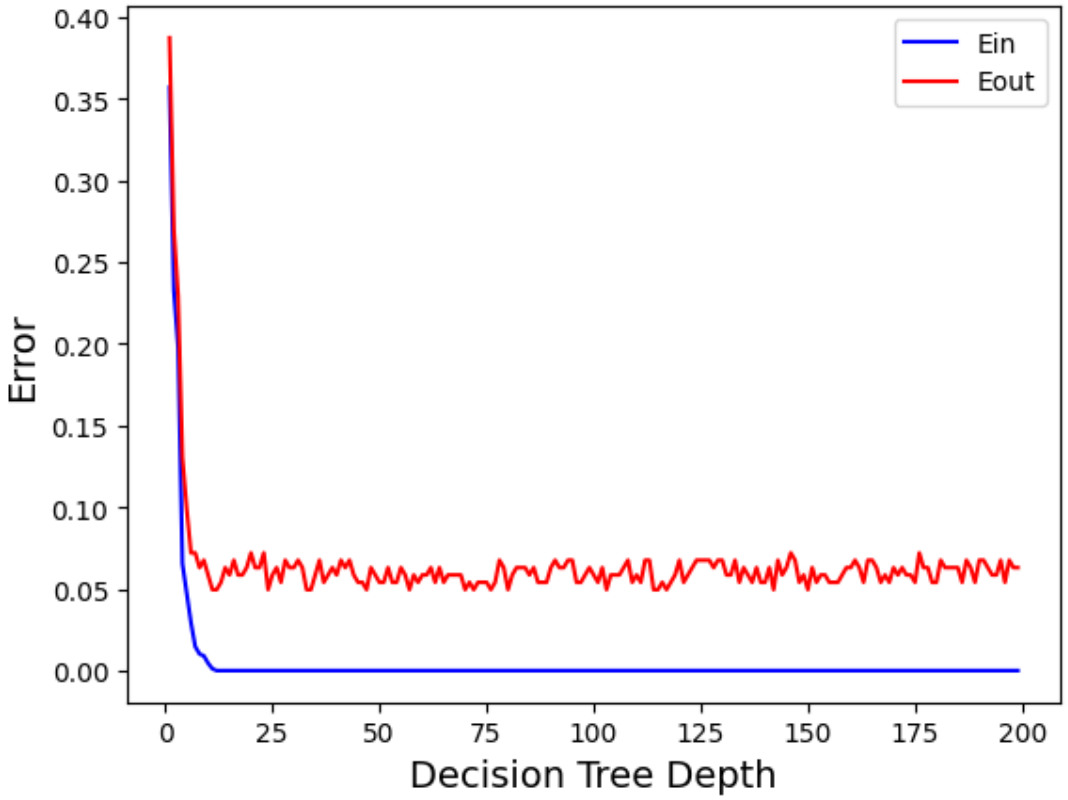
	precision	recall	f1-score	support
1	0.89	1.00	0.94	40
2	0.91	0.91	0.91	55
3	1.00	0.94	0.97	100
4	0.99	1.00	1.00	175
accuracy			0.97	370
macro avg	0.95	0.96	0.95	370
weighted avg	0.97	0.97	0.97	370

4. Decision Tree

Parameter Tuning

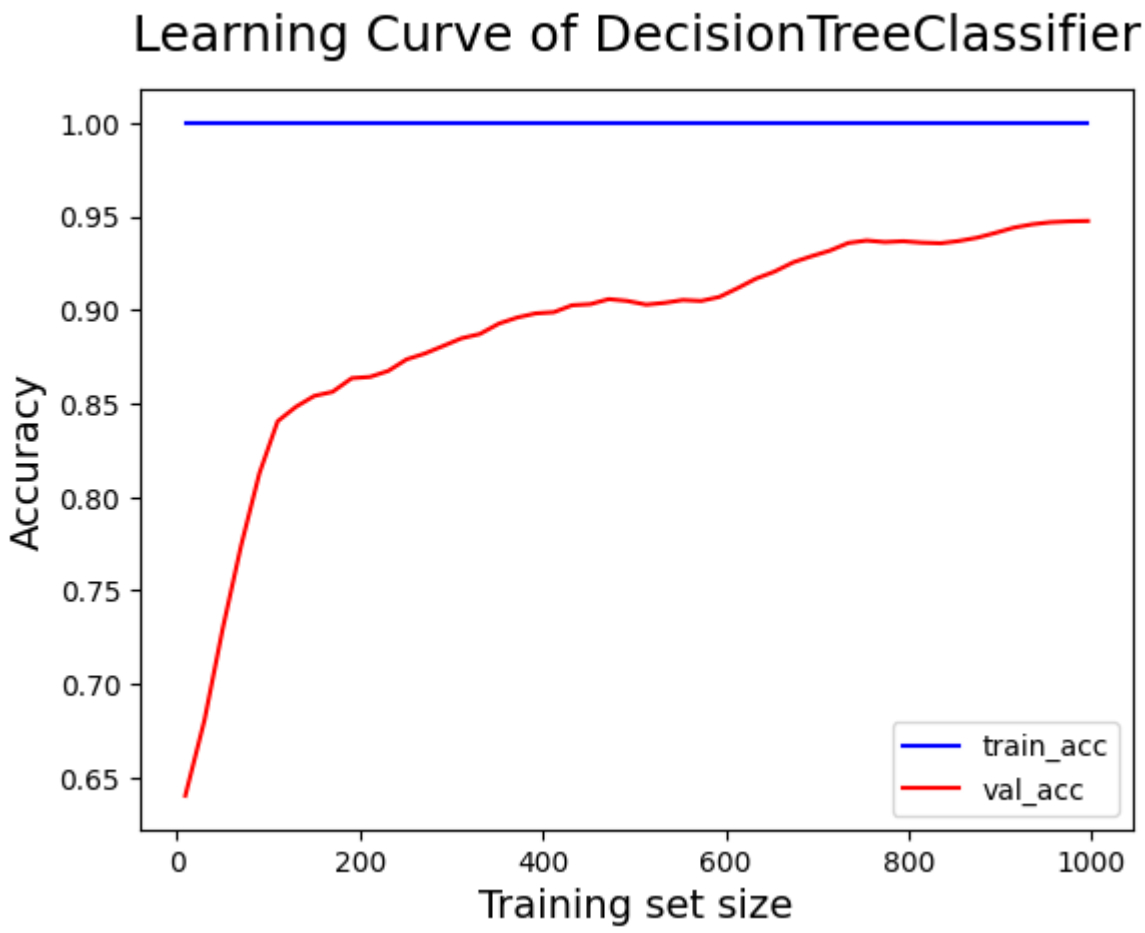
In decision tree algorithms, the depth parameter controls the maximum depth or level of the tree that is allowed during the learning process. The depth of a tree refers to the length of the longest path from the root node to any leaf node in the tree. Setting a maximum depth limit on a decision tree can be useful in preventing overfitting, but it is important to find the optimal balance between overfitting and underfitting through model selection techniques.

- Evaluating a decision tree for each depth in the range from 1 to 200:



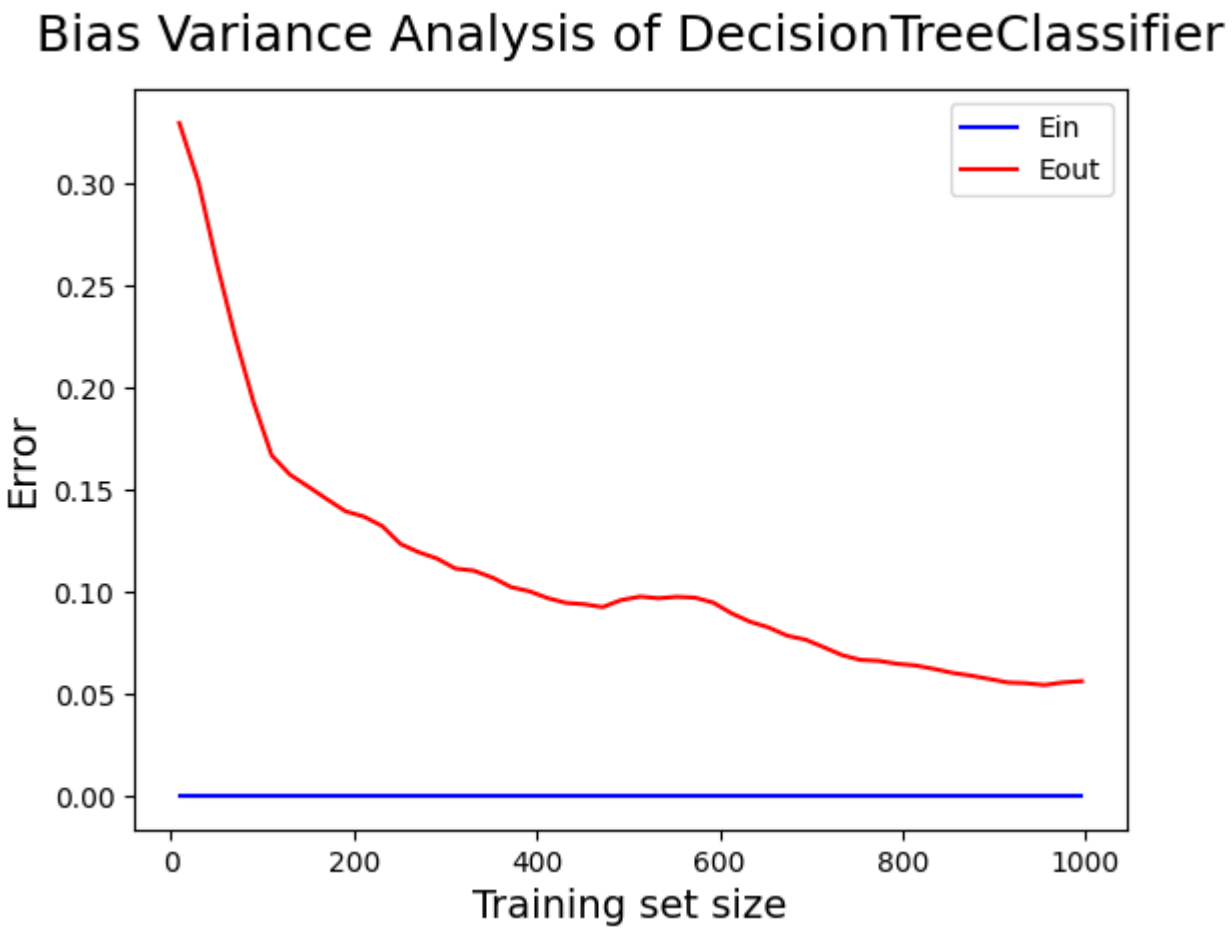
- As shown in the figure, by increasing the decision tree depth the in-sample and out-sample error decreased which indicates that the model can learn and generalize well to new, unseen data. However, it is important to strike a balance between overfitting and underfitting, as overly complex models can lead to overfitting and poor generalization performance.
- but we can observe that the in-sample and out-sample error saturated after depth almost equal to 25.
- Therefore, we will tune our depth parameter with max_depth = 25 proved by grid search.
- for max_depth, min_samples_split, min_samples_leaf, and max_features, GridSearchCV produced the default values for those parameters.

Learning Analysis



- As shown in the figure we are testing the model learning curve with the training set size
  - we can observe that by increasing the training set size the testing accuracy increases
  - then the model improves and generalizes well as more data is used to train it. A model that generalizes well to new, unseen data should be able to perform well on the testing set, even when it has not seen that data before. This is a desirable outcome that indicates the model is becoming more robust to different variations in the data and can make accurate predictions on new data.

Bias Variance Analysis



- From the above figure which represents a bias-variance analysis of the decision tree classifier, we can observe the following:
    - The model is complex as the model is complex.
    - Eout starts at the high point as the number of points initially is too small and thus the variance is too large.
    - Ein starts at a low point as the complex model can easily find a hypothesis that fits the small sample.
    - Eout and Ein become the same as N →∞ which is true by Hoeffding's inequality.
    - The final Eout is very low as N → ∞ and var → 0 and Eout includes only the bias.

Model Evaluation

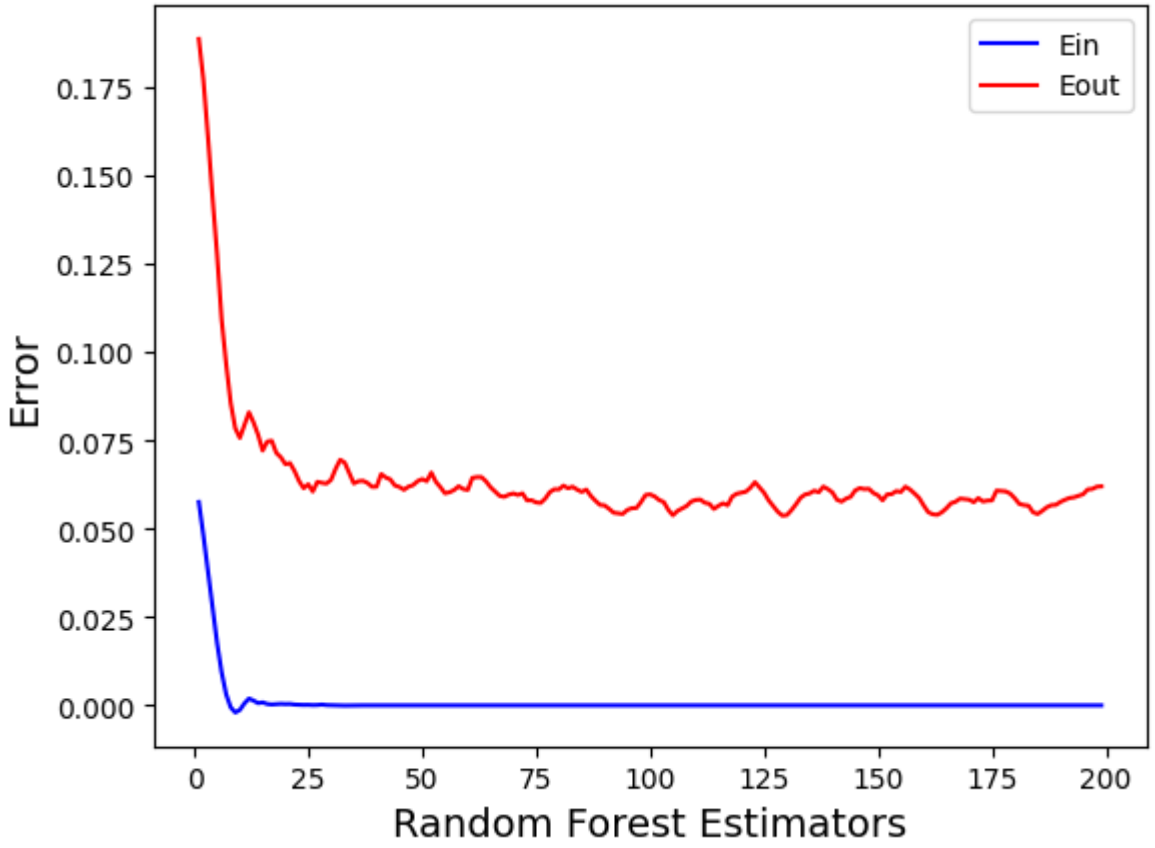
	precision	recall	f1-score	support
1	0.84	0.95	0.89	40
2	0.59	0.75	0.66	55
3	0.86	0.73	0.79	100
4	0.99	0.96	0.97	175
accuracy			0.86	370
macro avg	0.82	0.85	0.83	370
weighted avg	0.88	0.86	0.87	370

5. Random Forest

Parameter Tuning

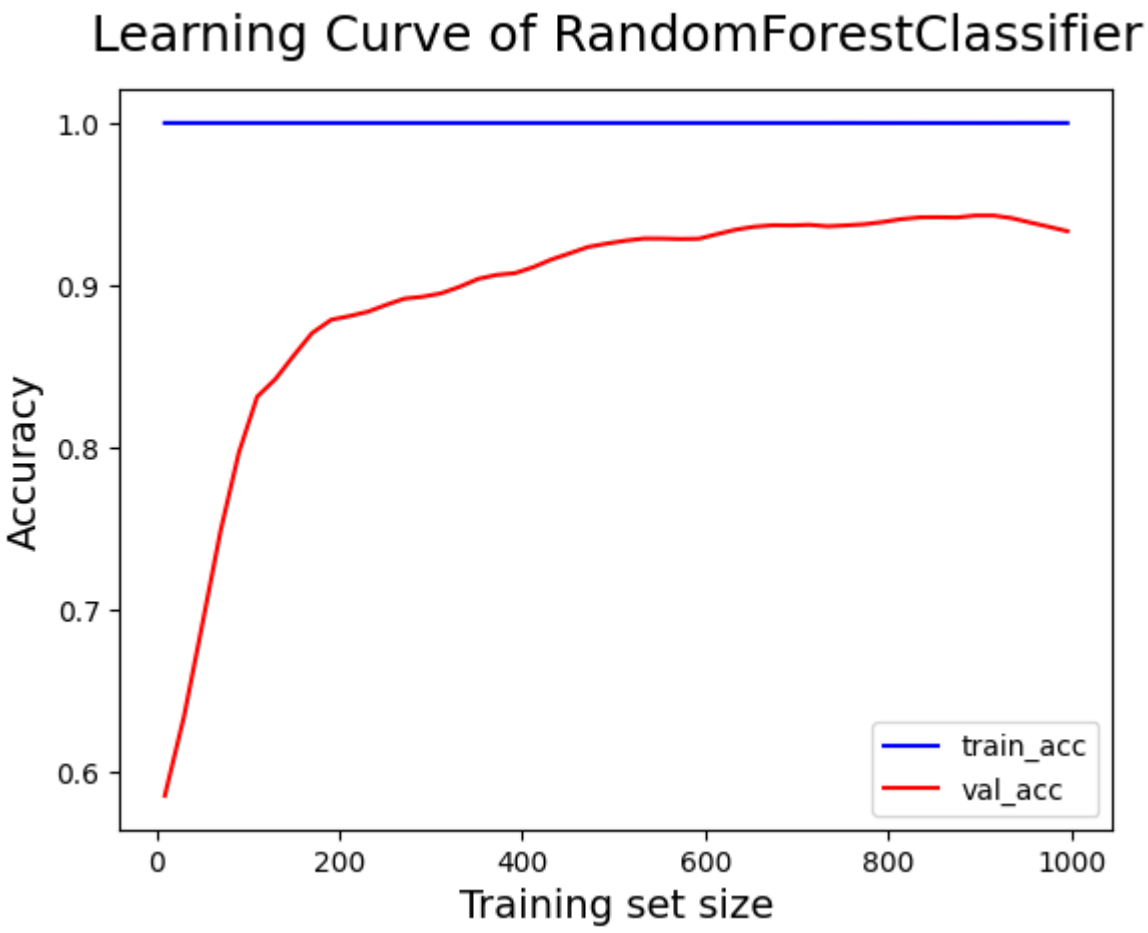
n_estimators is a hyperparameter in Random Forest that controls the number of decision trees built during training. Increasing n_estimators can improve the accuracy of the model up to a certain point, but setting it too high can lead to overfitting. The optimal value of n_estimators depends on the problem complexity, dataset size, and computational resources available. It is recommended to use a parallel implementation of the algorithm to speed up the training process.

As shown in the following figure the tuning of random forest estimators values



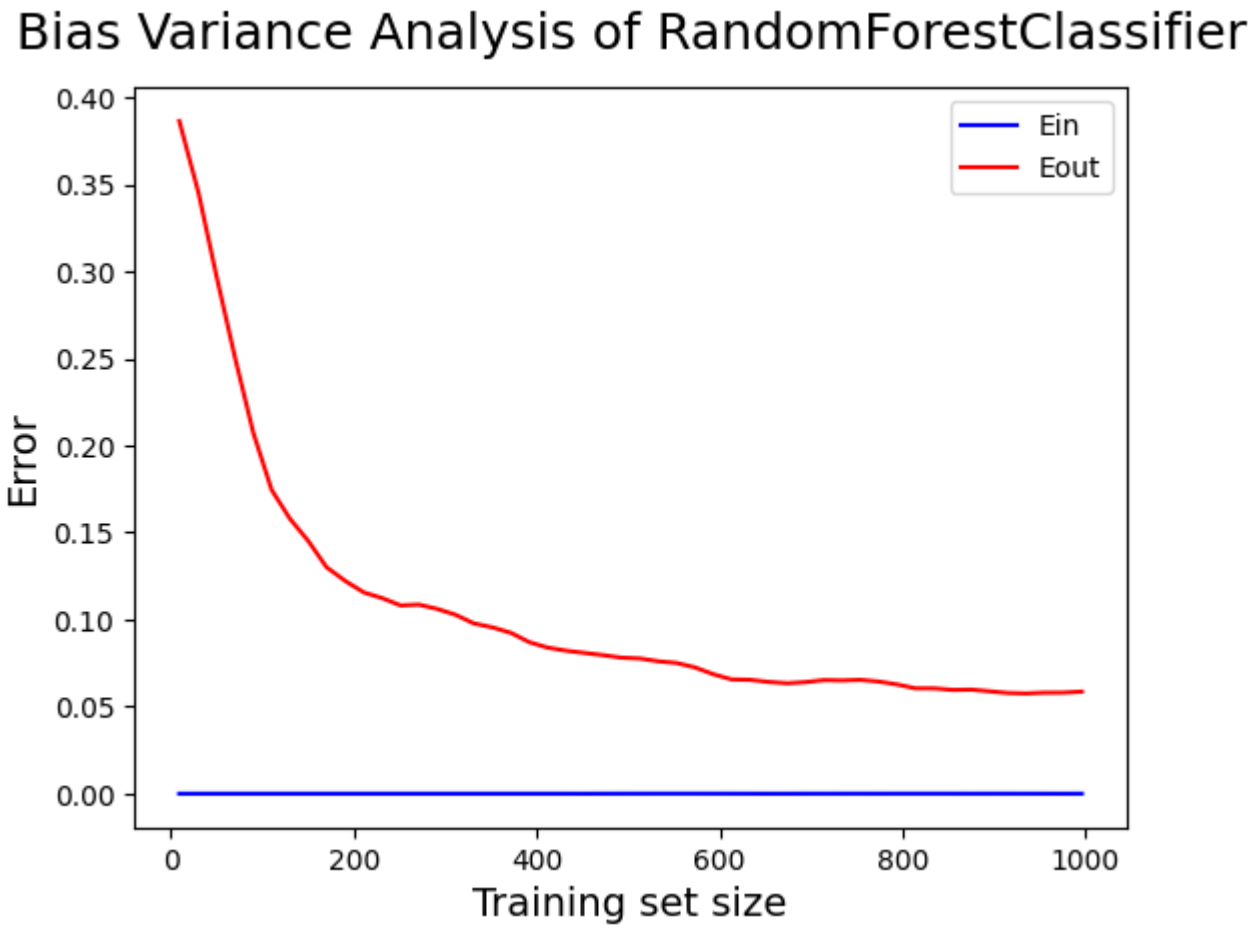
- by increasing the value we can show that decreasing the out-of-sample error but from the figure it almost saturates after value = 25
- we used GridSearchCV for the following parameters tuning:
- n_estimators, max_depth, min_samples_split, min_samples_leaf , max_features
- For all of those parameters, we go to the default value except n_estimators when using a value of 100.

Learning Analysis



- As shown in the figure we are testing the model learning curve with the training set size
- we can observe that by increasing the training set size the testing accuracy increases
- then the model improves and generalizes well as more data is used to train it. A model that generalizes well to new, unseen data should be able to perform well on the testing set, even when it has not seen that data before. This is a desirable outcome that indicates the model is becoming more robust to different variations in the data and can make accurate predictions on new data.

Bias Variance Analysis



- From the above figure which represents a bias-variance analysis of the decision tree classifier, we can observe the following:
- The model is complex.
- Eout starts at the high point as the number of points initially is too small and thus the variance is too large.
- Ein starts at a low point as the complex model can easily find a hypothesis that fits the small sample.
- Eout and Ein become the same as N →∞ which is true by Hoeffding's inequality.
- The final Eout is very low as N → ∞ and var → 0 and Eout includes only the bias.

Model Evaluation

	precision	recall	f1-score	support
1	0.85	1.00	0.92	40
2	0.85	0.75	0.80	55
3	0.84	0.91	0.87	100
4	0.98	0.94	0.96	175
accuracy			0.91	370
macro avg	0.88	0.90	0.89	370
weighted avg	0.91	0.91	0.91	370

4. Models Comparison

Model	Accuracy	Weighted Precision	Weighted Recall	Weighted F1
Logistic Regression	0.96	0.97	0.96	0.97
SVM	**0.97**	**0.97**	**0.97**	**0.97**
Decision Tree	0.86	0.88	0.86	0.87
Random Forest	0.92	0.92	0.92	0.92
ZeroR Baseline	0.47	0.22	0.47	0.3

Some notes for clarification:
Cross-validation: is a technique used in machine learning to assess the performance of a model and to tune its hyperparameters. It involves splitting the dataset into multiple subsets (or "folds") and using each subset in turn for testing the model while using the remaining subsets for training.

For example, if we use 10-fold cross-validation, we split the data into 10 equal-sized parts, and use 9 of them for training the model and 1 for testing, rotating which fold is used for testing until each fold has been used for testing once.

This way, each data point in the dataset is used for testing exactly once, and we get a more reliable estimate of the model's performance than if we only used a single train-test split. By averaging the performance across all the folds, we can get a more accurate estimate of the model's performance on unseen data.

Cross-validation is particularly useful when the dataset is small, noisy, or imbalanced, and can help to reduce overfitting and improve the generalization of the model. It can also help to identify the best hyperparameters for the model by testing different combinations of hyperparameters on the different folds and selecting the ones that perform best on average.