Faculty of computers
and artificial intelligence

Cover sheet

<u>AI330 Machine Learning Project</u>

Team no. :

Name	ID
عمار سعد محمد احمد جودة	20210588
سلمى محمود فوزي	20210417
سلمى ايمن عبدالفتاح	20210411
على عبد ربه	20210578
ادم محمد عطيه	20210133
شروق محمد السيد	20210449
فادي هاني عبدالملاك	20210656

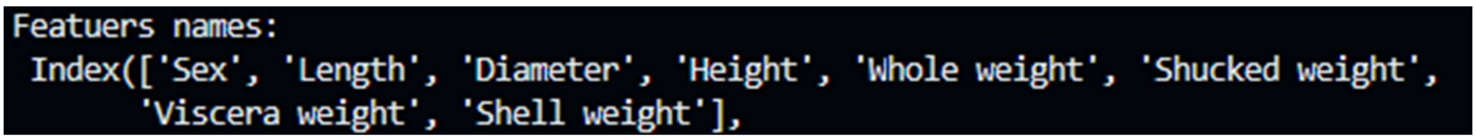
Numerical dataset

General information about dataset

Name	abalone.csv
type of the problem	regression
Total no. of samples	4000 row
No. of samples in training\validation	3341
No. of samples in testing	836

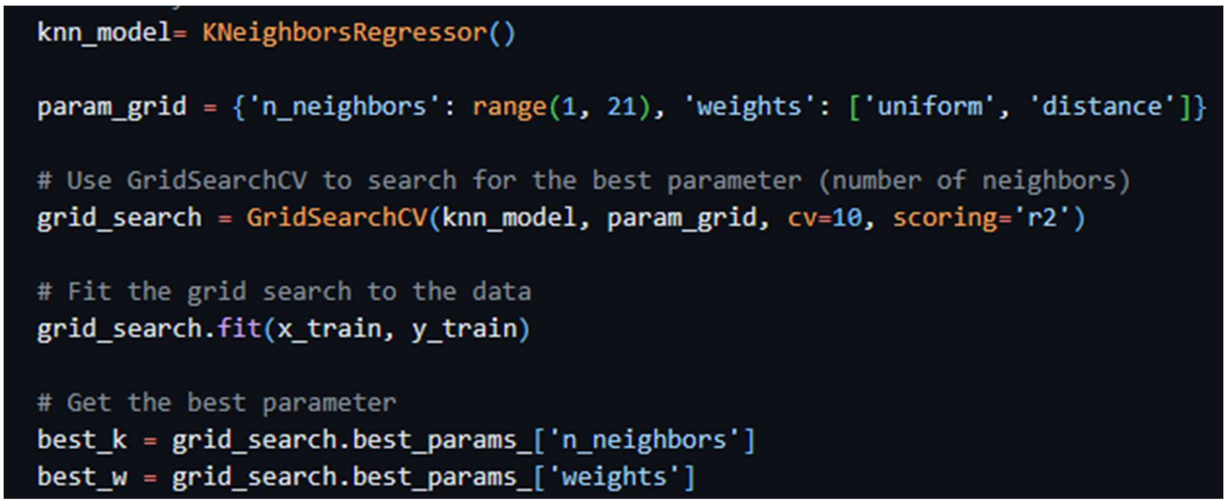
b)Implementation details

>>



```
Featuers names:
 Index(['Sex', 'Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight'],
```

>>



```
knn_model= KNeighborsRegressor()

param_grid = {'n_neighbors': range(1, 21), 'weights': ['uniform', 'distance']}

# Use GridSearchCV to search for the best parameter (number of neighbors)
grid_search = GridSearchCV(knn_model, param_grid, cv=10, scoring='r2')

# Fit the grid search to the data
grid_search.fit(x_train, y_train)

# Get the best parameter
best_k = grid_search.best_params_['n_neighbors']
best_w = grid_search.best_params_['weights']
```

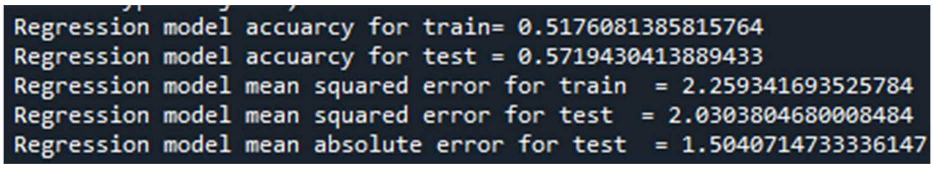
>>the hyperparameters that we used :

-randome_state in train_test_split: is responsible for shuffling the data

-n_neighbors : The number of neighbors to use for predictions

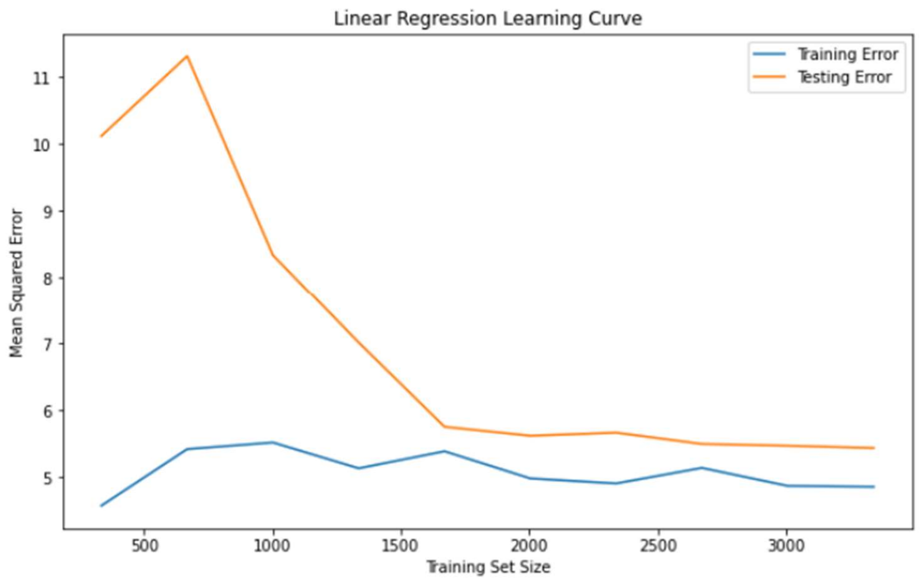
-weights: Weight function used in prediction. It specifies the weight given to each neighbor. and we used weights = distance which says Closer neighbors have a greater influence on the prediction than farther neighbors.

>> linear Regression model results details:

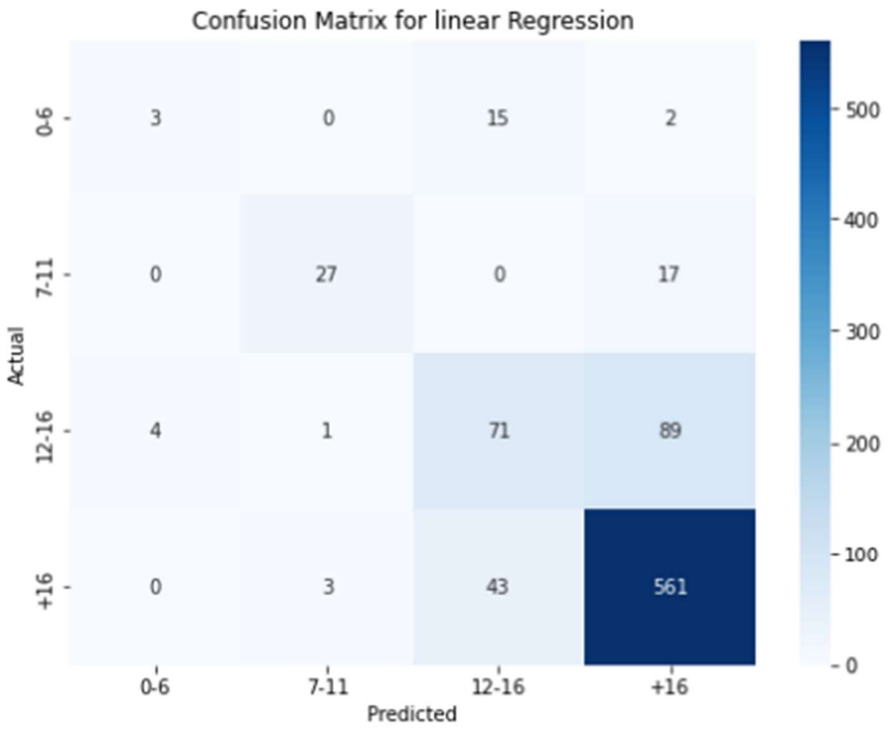


```
Regression model accuarcy for train= 0.5176081385815764
Regression model accuarcy for test = 0.5719430413889433
Regression model mean squared error for train  = 2.259341693525784
Regression model mean squared error for test  = 2.0303804680008484
Regression model mean absolute error for test  = 1.5040714733336147
```

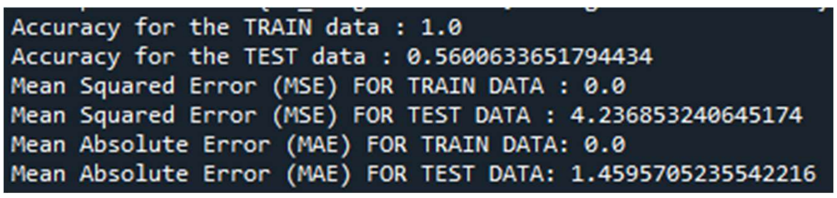
-loss curve:



-confusion matrix for actual vs predictions

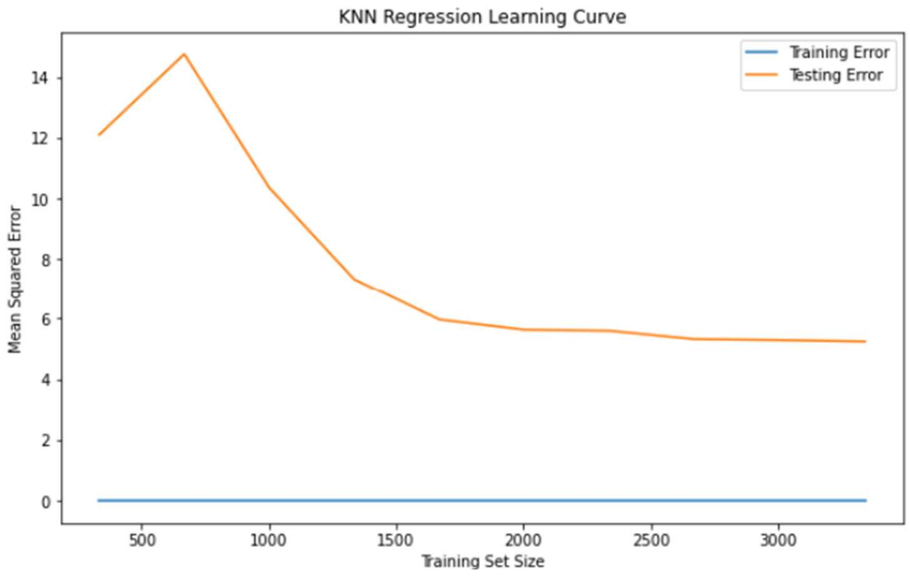


>>KNeighborsRegressor model results details:

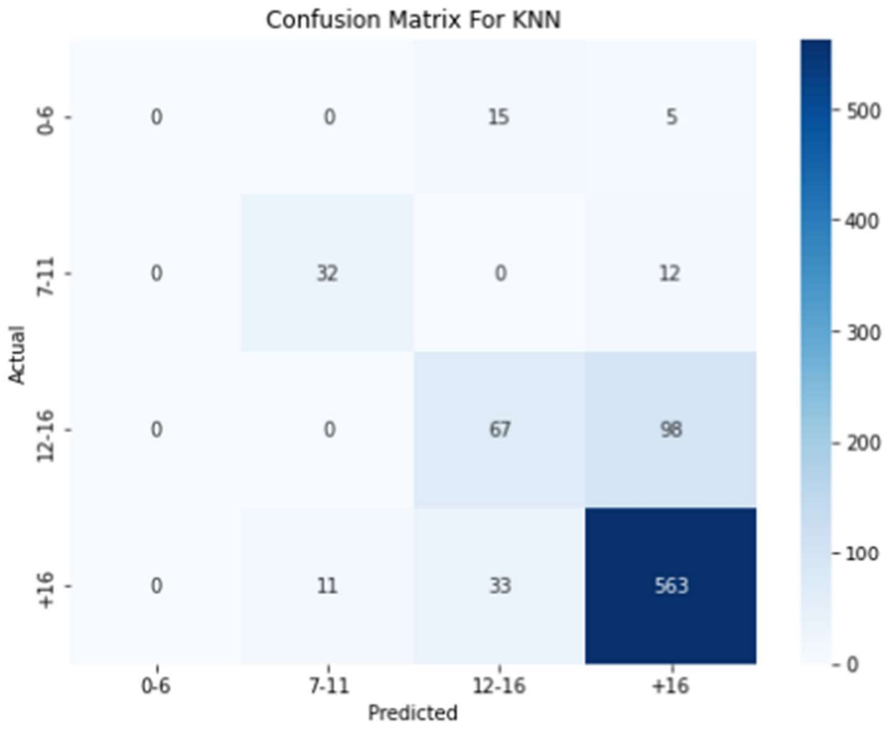


```
Accuracy for the TRAIN data : 1.0
Accuracy for the TEST data : 0.5600633651794434
Mean Squared Error (MSE) FOR TRAIN DATA : 0.0
Mean Squared Error (MSE) FOR TEST DATA : 4.236853240645174
Mean Absolute Error (MAE) FOR TRAIN DATA: 0.0
Mean Absolute Error (MAE) FOR TEST DATA: 1.4595705235542216
```

-loss curve:



-confusion matrix actual vs predictions:



	0-6	7-11	12-16	+16
0-6	0	0	15	5
7-11	0	32	0	12
12-16	0	0	67	98
+16	0	11	33	563

Image dataset

General information about dataset

Name	Malaria
No. of classes	2 classes
Total no. of samples	1000 per class
Size of image	average 15KB
No. of samples in training\validation	1455
No. of samples in testing	624

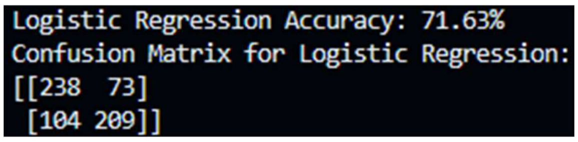
>>the hyperparameters that we used :

n_clusters in Kmeans: The number of clusters to form.

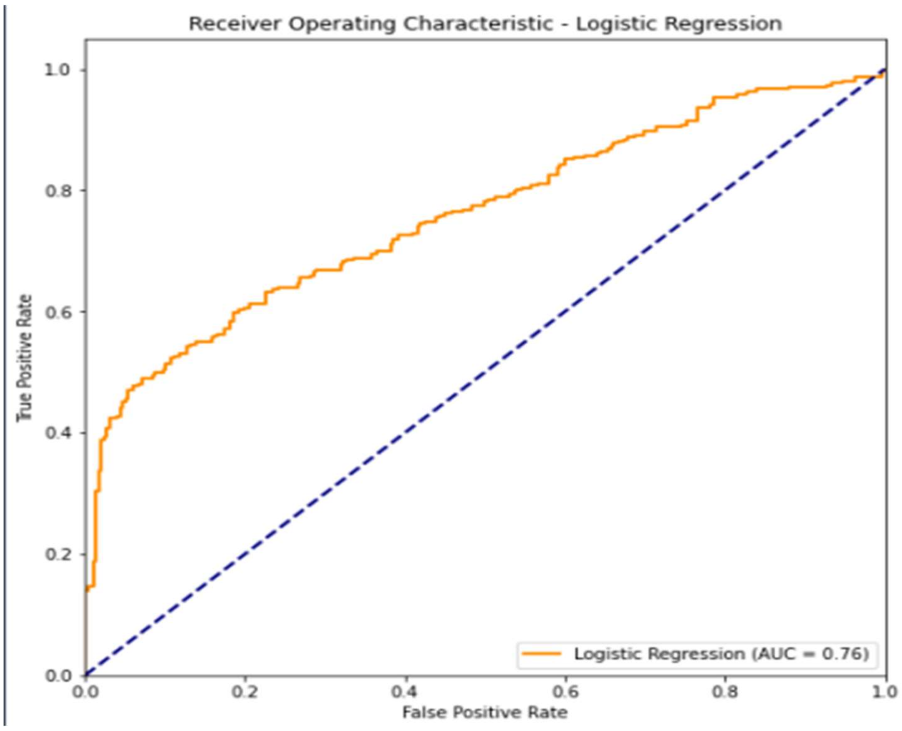
n_init in Kmeans: Number of times the k-means algorithm will be run with different centroid seeds. The final results will be the best output of n_init consecutive runs in terms of inertia.

n_components in PCA: Number of components to keep. If not specified, all components are kept.

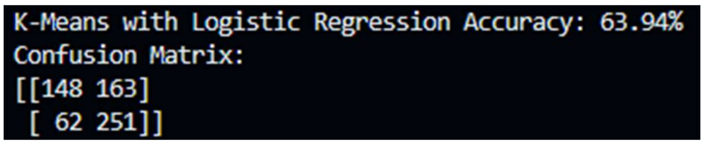
>>Logistic regression



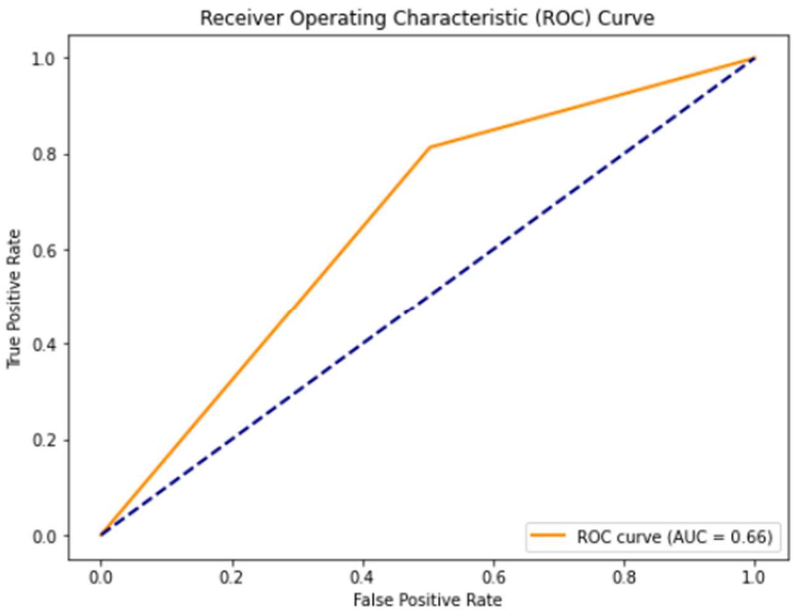
```
Logistic Regression Accuracy: 71.63%
Confusion Matrix for Logistic Regression:
[[238  73]
 [104 209]]
```

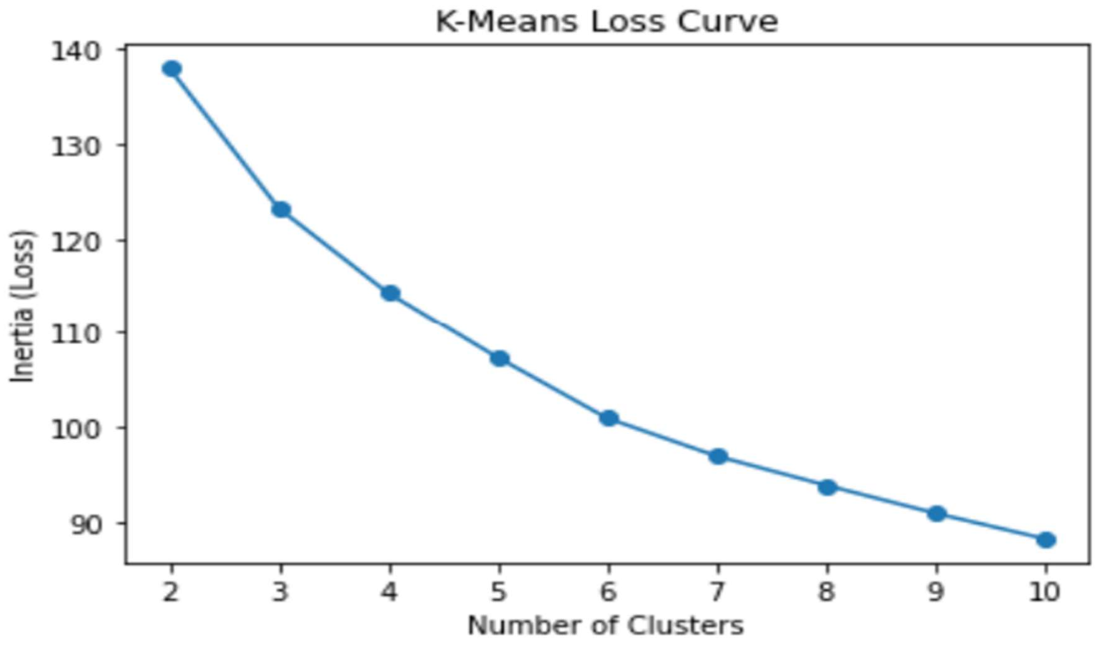


>> KMeans



```
K-Means with Logistic Regression Accuracy: 63.94%
Confusion Matrix:
[[148 163]
 [ 62 251]]
```


Receiver Operating Characteristic (ROC) Curve
True Positive Rate
False Positive Rate
ROC curve (AUC = 0.66)


K-Means Loss Curve
Inertia (Loss)
Number of Clusters