Heart Disease Prediction

Team ID: B02

—

Pattern Recognition

—

Team Members:

1) Peter Melad
2) Michael Maher
3) Nourhan Tarek
4) Beeshoy Elshayep

PREPROCESSING STEP

-MISSING VALUES:

We had to check first for the missing values in the dataset (Training, Testing) we found that there are no missing values. However, we made a preprocessing step that ensures that there will be no any missing values if any other dataset were put in our model so, we used the mean of each column to fill any missing values.

-STANDATIZATION & SCALING:

In this step we used (sklearn preprocessing) (**StandardScaler**) So each column in the Dataset (Training, Testing) was scaled.

-FEATURE SELECTION:

As an Important step in preprocessing, We had to specify the most important features in our Dataset we used (sklearn.feature_selection) (SelectKBest, chi2) to select the **MOST EFFECTIVE** features in the Dataset. Due So, we got 5 features that their value was above 30.

-DIMENSIONALITY REDUCTION:

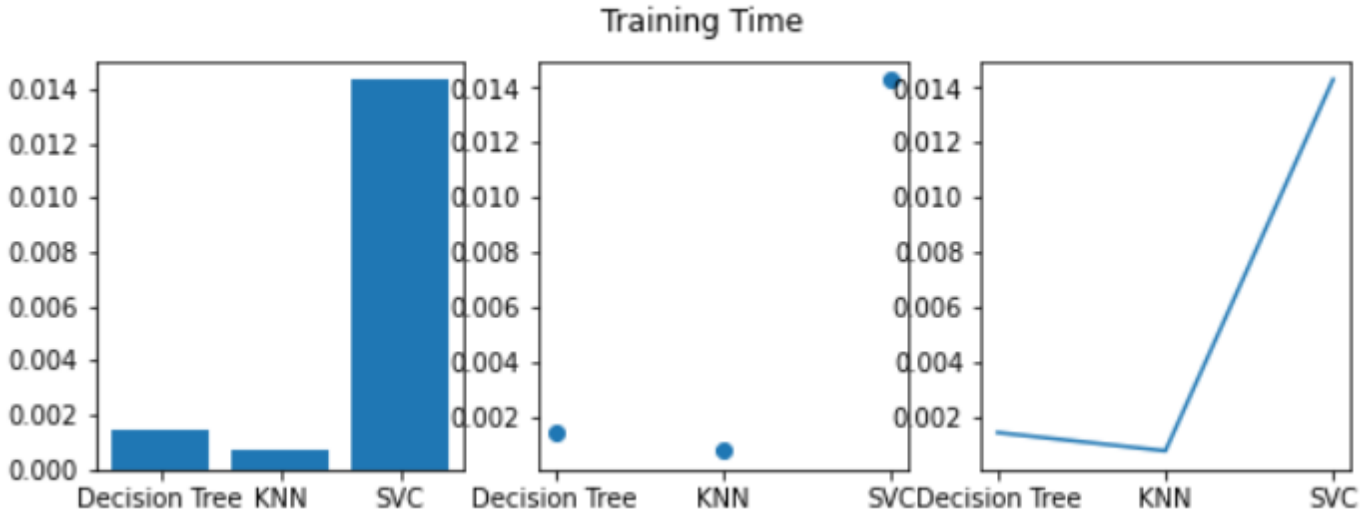
In this step we tried to maintain our data accuracy and to reduce the number of features (Dimensions) So, we used the PCA model to transform the data. At first, we didn't allocate any value for "n_components" to see the graph and after many tries, we dedicated 0.95 as a value for "n_components".

-TRAINING AND VALIDATION DATA SIZE:

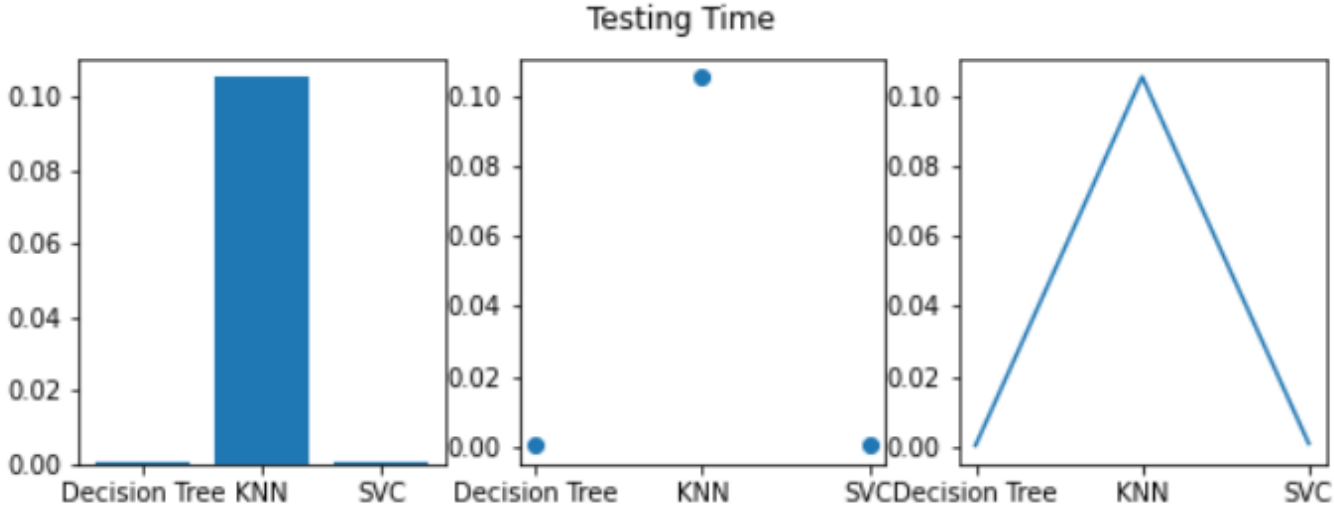
After many tries, we dedicated that Data would be spliced to 80% training and 20% for validation.

Models Summarize

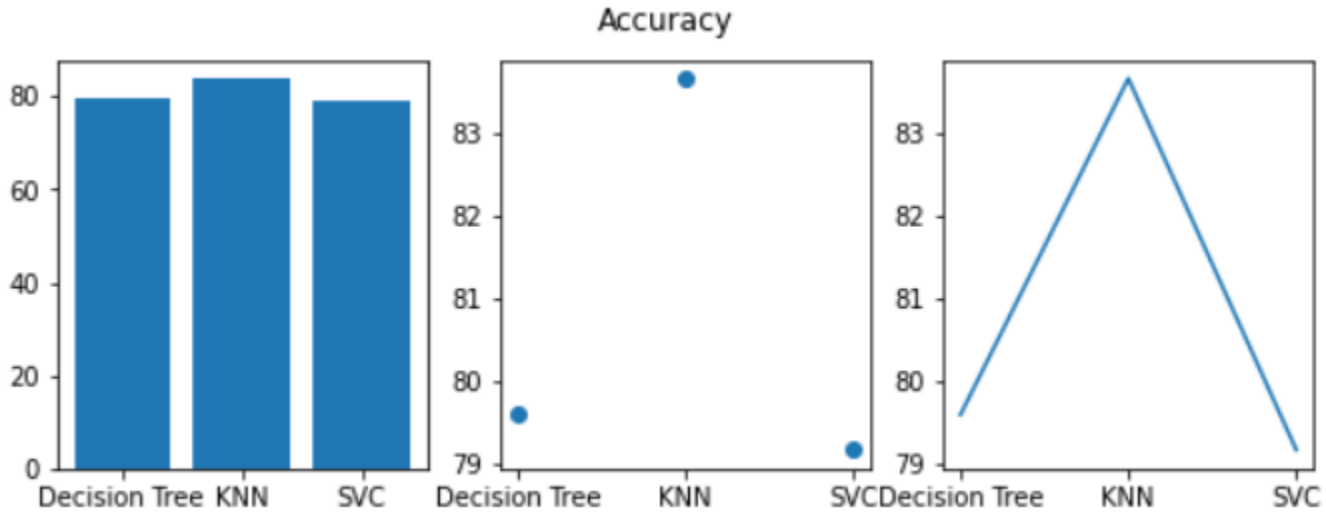
-TRAINING TIME



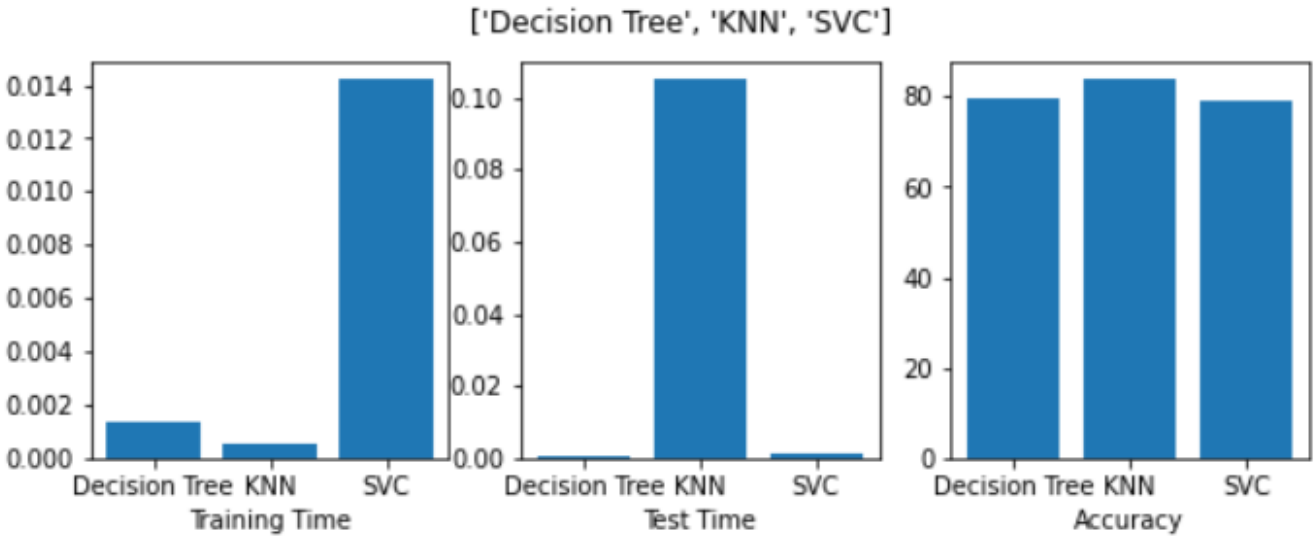
-TESTING TIME



-ACCURACY



----BAR GRAPH FOR THE 3 MODELS----



-The highest Training time is the SVC model
-The highest Test time is the KNN model
-The highest Accuracy is the KNN model

HYPERPRAMETERS

1)Decision Tree:

-max_depth:

(Knowing that max_depth leads to max decision made by the model)

After many tries (fixing other parameters), we found this any value for max_depth below 6 leaded to lower accuracy and any above value leads to the same accuracy.

-Splitter:

(Knowing that splitter is the strategy used to choose the split at each node)

After many tries (fixing other parameters), we found this:
-using the “best” strategy leaded to lower accuracy.
-using the “random” strategy leaded to higher accuracy.
-No other strategies were found for the splitter parameter.

- min_samples_leaf:

(Knowing that **min_samples_leaf** is the minimum number of samples required to be at a leaf node)

After many tries (fixing other parameters), we found this any value below 9 to 3 leaded to lower accuracy (75.5102% at min_samlpes_leaf=7), when giving value 9 or 10 the accuracy raise up to (83.6734%) any value above 10 leaded to lower accuracy (71.4285% at min_samlpes_leaf=13)

Values of 1 or 2 leaded to (81.632%)

2)KNN

-n_neighbors:

(Knowing that n_neighbors is the Number of neighbors to use to choose the class)

After many tries (fixing other parameters), we found this any value below 75 leaded to lower accuracy (85.71428 % at n_neighbors =51), when giving value 75 to 94 the accuracy raise up to (87.75510 %) any value above 94 leaded to lower accuracy (83.6734 % at min_samlpes_leaf=125)

- weights:

(Knowing that **weights** is weight function used in prediction)

After trying("uniform") (fixing other parameters), the accuracy was (79.59183%) when using ("distance") the accuracy was (87.75510 %)

-No other values were found for this parameter.

-algorithm:

(Knowing that **algorithm** Algorithm used to compute the nearest neighbors)

"auto": leaded to (87.755102%)
"ball_tree": leaded to (87.755102%)
"kd_tree": leaded to (87.755102%)
"brute": leaded to (87.755102%)

3)SVC

-USING K-FOLD MODEL WITH SVC:

Using K-Fold with SVC leaded to higher accuracy on Kaggle

-**kernel:**

(Knowing that **kernel** Specifies the kernel type to be used in the algorithm)

“sigmoid”: leaded to (66.666666%)
“linear”: leaded to (75.0%)
“poly”: leaded to (77.083333%)
“rbf”: leaded to (79.166666 %)

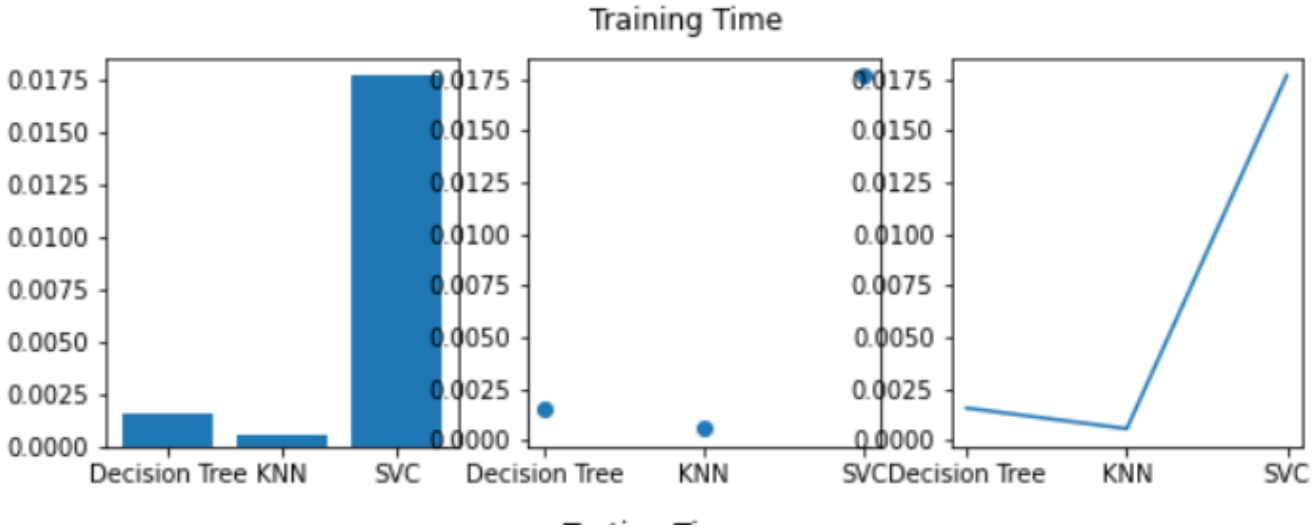
- C:

(Knowing that **c** is the Regularization parameter)

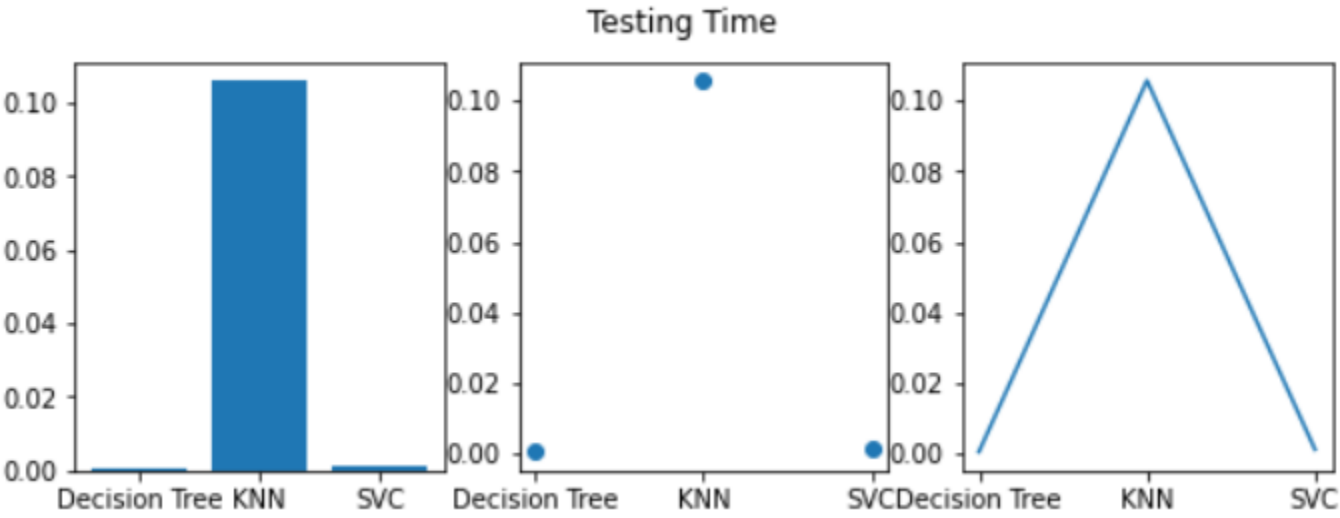
After many tries (fixing other parameters), we found this using 4 as a value leaded to lower accuracy (77.08333%), when giving value 2 the accuracy raise up to (81.25%) any value above 4 leaded to accuracy (79.1666% at c=8)

Models Summarize using PCA

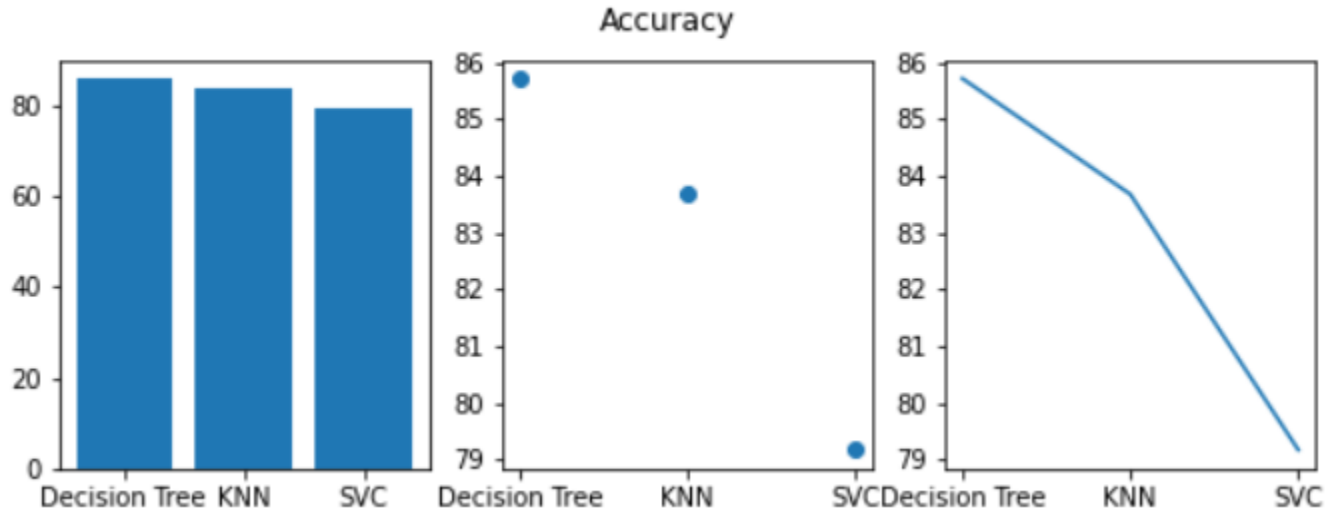
-Training Time



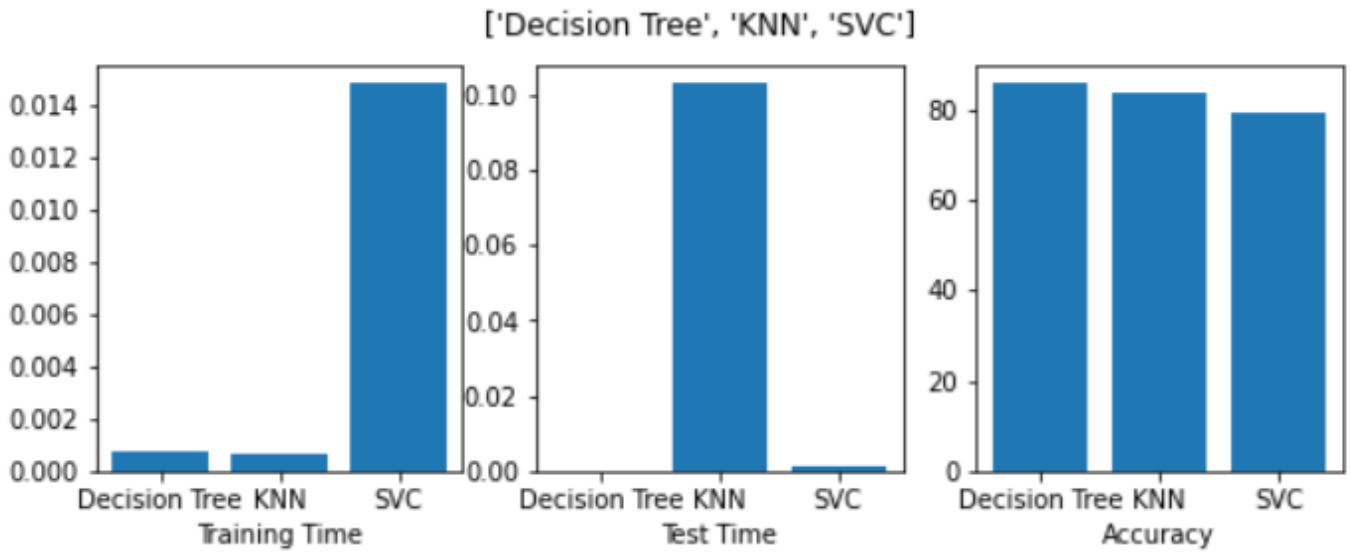
-Testing Time



-Accuracy



----BAR GRAPH FOR THE 3 MODELS----



CONCLUSION

Through the phase of the project we found out that accepting a single acceptable accuracy is not always right as there are always better… whether using the same model with different preprocessing techniques or different hyperparameter or different values for these hyperparameters or even using a very different model .However we used many models such as(Naïve Bayes ,Logistic Regression, Random Forest) So finally in our problem here we think that the more we try the more we know the better we can get.