Project 2

CS 205. Artificial Intelligence

Instructor: - Dr. Eamonn Keogh

Pratheek Chindodi Rajashekar
SID 862002345
pchin006@ucr.edu
22-March-2018

In completing this project, I consulted. . .

- **Dr. Eamonn Keogh**: Several inputs on implementing the project and for getting an idea of an original search algorithm
- Machine Learning 001 and Machine Learning 002 slides
- Python 3.6 documentation. https://docs.python.org/3/
- Idea for Original Search Algorithm. https://github.com/ader003/cs170-FSwNN
- UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/index.php Wine Dataset for the Extra Credit Question
- Wikipedia K-Nearest Neighbors Algorithm https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
- K-Nearest Neighbors Algorithm http://dataaspirant.com/2016/12/23/k-nearest-neighbor-classifier-intro/
- Feature Selection with Nearest Neighbor https://github.com/MichaelUy/Feature-with-Nearest-Neighbor
- Wrapper Methods https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/
- Functions from **Scipy** such as stats to normalize the data.

CS205: Project2
Feature Selection with k-Nearest Neighbor Algorithm

Pratheek Chindodi Rajashekar, SID 862002345

1. Introduction

This is the second programming assignment for CS 205 Artificial Intelligence. This project uses three search algorithms to find the best feature subset and its corresponding accuracy. The first two algorithms are Forward Selection and Backward Elimination. The third algorithm designed is an extension to Backward Elimination, which stops the search when the accuracy of the feature subset degrades when compared to the previous level. It is Backward Elimination with Pruning. The language that I used to develop the Feature Selection with k-Nearest Neighbors is Python. It took 226 lines of code to finish the project. In addition to the datasets assigned for this project, I took the Wine Data Set from the UCI Machine Learning Repository to test the algorithms and obtain the best attribute subset from the Wine Data Set.

2. Steps for Implementing the Project

1) Load the Dataset into the Code and perform Z- Score Normalization on the continuous features of the dataset such that the features of the dataset exist between 1 and -1.
2) The total number of instances and the features present in the dataset are computed.
3) Choose a Feature Selection algorithm to find the best attributes and the best accuracy.
4) At the first, the Feature Selection algorithm uses all the features present in the dataset to find the accuracy. This accuracy is found by finding the nearest neighbors of the dataset to the test instance. The test instance is obtained by leave-one out cross validation, where, one instance is considered as test instance and the rest of the instances in the dataset are considered as training set. The distance metric used for k-Nearest Neighbors is Euclidean Distance.
5) A Wrapper method such as Forward Selection or Backward Elimination is used to find the best subset of features which gives the highest accuracy.
6) For the forward selection algorithm, the search for the best attribute set begins by considering only a single feature of the normalized dataset.
7) For the backward elimination, the search of the best feature subset begins by considering all the features in the dataset.
8) For the backward elimination with pruning, the search of the best feature subset begins by considering all the subsets and then removing the number of features until a point is reached where there is a decrease in the accuracy of the algorithm.

3. Algorithms

3.1 Datasets :-

The datasets used for this project are **CS205_SMALLtestdata_9.txt** and **CS205_BIGtestdata_58.txt** . In addition to this, I have used the **Wine Dataset** of UCI Machine Learning Repository.

3.2 Leave-one-out cross validation

This is the cross-validation technique that is used in this project. With Leave- one-out cross validation technique, one instance of the dataset is taken as the test instance upon which a prediction is done. The rest of the instances in the dataset is considered as the training set for which the model is trained.

3.3 K-Nearest Neighbor Algorithm

Leave-one-out cross validation technique considers one instance as the test instance. A distance metric, that is, Euclidean distance is used to measure the similarity between this test instance and each of the other instances in the dataset. The distance is calculated for each of the instance in the training set. These distances are sorted in an increasing order. After the distances are sorted, the first k distances are considered and points corresponding to these k distances are found. In our dataset, we have only two classes. Therefore, if more number of points are closer to the first class, then the test instance is given the label of first class. Similarly, if more number of points are closer to 2^{nd} class, then the test instance is labelled as 2^{nd} class. The K-Nearest Neighbor with leave-one-out-cross validation is an iterative method, where, one of the instances becomes a test instance and the rest of the dataset becomes the training set.

3.4 Forward Selection

This is a type of Wrapper Method, which is an iterative method in which the algorithm starts with having no feature in the model. In each iteration, only a single feature is added that improves our model.

In Table 1, for the Small dataset, the time taken by the algorithm, the accuracy, best feature subset for the dataset is tabulated. It can be seen that all the algorithms have the same accuracy. But Forward Selection algorithm is faster compared to the other algorithms. This is because the forward selection algorithm starts with less number of features and then it keeps on adding features until it completes all the combinations.

In Table 2, that is for the Big dataset with higher number of features, the forward feature selection has higher accuracy as compared to the other feature selection algorithms again works better than the backward elimination by taking less time to find the feature subset. So, these results match the theoretical concepts that the forward feature selection algorithm is faster than the backward feature elimination.

3.5 Backward Elimination

This is a type of Wrapper method, which is an iterative method that begins with all the features in the Feature Set. And then it removes a single feature at each level of the tree, as opposed to the Forward Selection where a single feature is added at each level of the tree.
In Table 1, for the Small Dataset it can be observed that the Backward elimination has an accuracy that is equal to the Forward selection. However, since the backward elimination starts with all the combinations of the features, it takes more computation time to find the best feature subset and the accuracy as compared to forward selection.
Figure 2 shows the accuracy of the algorithm with respect to the number of features. For datasets with fewer number of features, the accuracy and the best feature subset are almost equal for forward selection and backward elimination.
In Table 2, for the Large Dataset, the Backward elimination, in addition to lower accuracy, considers more number of features for the best feature subset.

3.6 Backward Elimination with Pruning

This is the custom search that has been done for this project. It is an extension of Backward Elimination. In this algorithm, initially, all the features are considered to find the best feature subset. This algorithm starts eliminating the features that are irrelevant or the features that hamper the accuracy of backward elimination. This iterative method of removing a feature continues till a level i in the tree where the accuracy increases. When the accuracy decreases at any level, the search stops and gives the accuracy and the best feature subset of the last iteration.

To understand this better, let us consider a dataset with 10 features. The search begins by eliminating one of the 10 features in the subset. If the accuracy found at level 9 is greater than level 10, then the search continues to the next level of the tree. If the accuracy at level 9 is found to be lesser than the accuracy of level 10, the search stops and gives the accuracy of level 10, that is the level with all the features as the best subset. Since this algorithm does not search the entire space as opposed to backward elimination, it takes less number of computations to find the accuracy and the best feature subset.

In Table 1, for smaller dataset, the accuracy of Backward elimination with Pruning is equal to that of both the other algorithms. The time taken to find the accuracy and the best feature subset is somewhat equal to that of backward elimination.

In Table 2, for bigger dataset, although, the accuracy is a little lower when compared to backward elimination, the computation time to find the best feature subset and accuracy is comparatively smaller than forward selection and lesser than half of the time taken for backward elimination. This algorithm can be useful in datasets where there are an enormous number of features, since it stops at a level where the accuracy decreases.

4. RESULTS

Table 1: CS170_SmallData9.txt

	Time(seconds)	Best Accuracy(%)	Best Feature Subset
Forward Selection	3.9828	93.0	{8,10}
Backward Elimination	5.18631	93.0	{8,10}
Backward Elimination with Pruning	5.1956	93.0	{8,10}

Table 2: CS170_BigData58.txt

	Time (seconds)	Best Accuracy(%)	Best Feature Subset
Forward Selection	309.79	98.0	{5,41}
Backward Elimination	488.16	90.0	{3,4,5,10,20,21,24,25,27,28,32,33, 36,39,40,41,42,43,44,46,48,49,50}
Backward Elimination with Pruning	241.927	86.0	{1,3,4.5.9,10,11,12,13,14,15,16,17, 18,19,20,21,24,25,26,27,28,29,32, 33,34,35,36,37,38,39,40,41,42,43 44,46,47,48,49,50}

Table 3: Accuracy of Forward Selection and Backward elimination for different number of features

Number of Features	Forward Selection(%)	Backward Elimination(%)
1	**81.0**	**81.0**
2	**93.0**	**93.0**
3	**92.0**	**92.0**
4	**84.0**	**84.0**
5	**81.0**	**81.0**
6	**77.0**	**77.0**
7	**77.0**	**77.0**
8	**71.0**	**71.0**
9	**66.0**	**66.0**
10	**60.0**	**60.0**

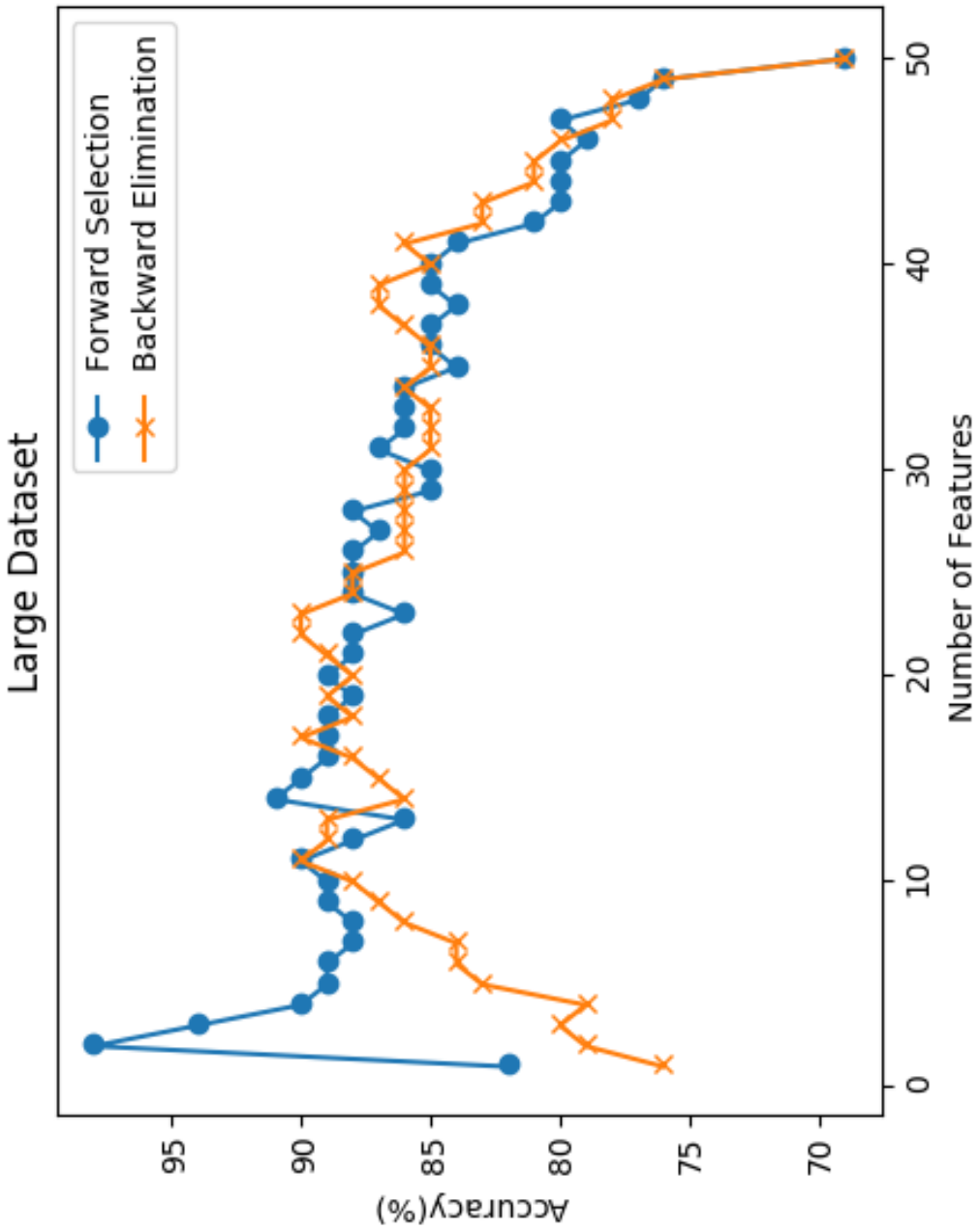

Figure 1 : Large Dataset

The above figure displays the Accuracy of the Forward Selection and Backward Elimination against the number of features

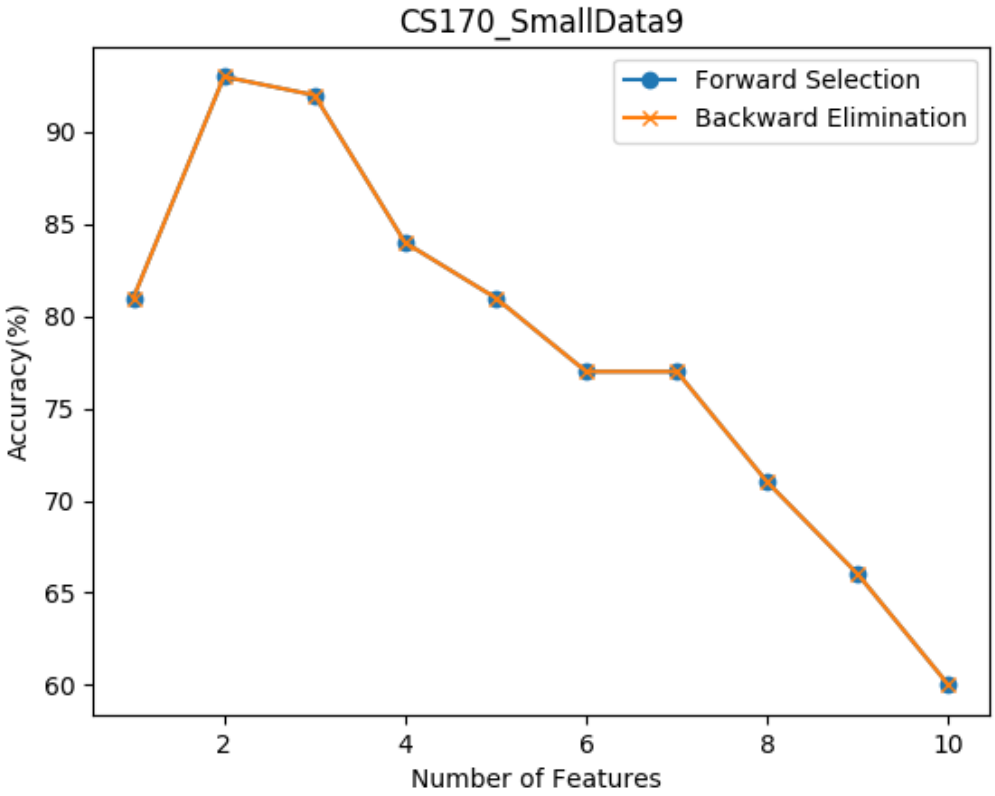

Figure 2: Small Dataset

5. Conclusion :-

- Forward Selection algorithm has the highest accuracy as compared to Backward elimination and Backward elimination with pruning since Forward selection begins with the least number of features. Forward selection algorithm can be smoothened further to get a higher accuracy and better feature subset.
- Backward Elimination has a very high computation time as compared to Forward elimination, especially in datasets with higher number of features.
- Backward Elimination with Pruning performs better on larger datasets where the accuracy decreases after a certain level in the search tree. This algorithm starts eliminating the features that hamper the accuracy of the algorithm. Since this algorithm does not search the entire space as opposed to backward elimination, it takes less number of computations to find the accuracy and the best feature subset.

6. Extra Credit Question

5.1 UCI Wine Dataset

The Wine dataset are the results of a chemical analysis of wines grown in three cultivars of Italy. The analysis has 13 features which indicate the constituents used in the three different cultivars in Italy. There are 178 number of instances with each instance having 13 features.

5.2 Results

- The given dataset has attributes which are continuous, similar to the datasets assigned for this project. These continuous values are discretized using Z-Score normalization so that the range of the attributes lie between 1 and -1. There are three classes in this dataset, with the first class having 59 instances, second class having 71 instances and third class having 48 instances.
- The above three algorithms were used to find the best feature subset and the accuracy of the dataset.
- In Table 4, the time taken by the algorithm, best accuracy of the algorithm, best feature subset found by the algorithm are tabulated.
- From Table 4, it can be observed that the custom search, that is, Backward Elimination with Pruning works faster when compared to other algorithms and also has a good accuracy . The custom search takes less number of computations when compared to the other algorithms.
- The forward selection algorithm has the highest accuracy for the Wine Dataset. But the number of features in the feature subset is equal to that of Backward Elimination and Backward Elimination with Pruning. The backward elimination algorithm is slow since it takes more time to find the best feature subset and the accuracy.
- Figure 2 indicates a plot of the accuracy of the forward selection and backward elimination algorithms with respect to the number of features. The values of the accuracy are tabulated in Table 5.
- For forward selection, the accuracy increases as the number of features increases until the number of features is equal to 10. For backward elimination, the accuracy decreases as the number of features in the best feature subset goes below 9.

Table 4: Wine Dataset(Source:- UCI Machine Learning Repository)

	Time(seconds)	Best Accuracy(%)	Best Feature Subset
Forward Selection	26.335	98.8764	{7,10,13,11,1,5,2,12}
Backward Elimination	34.1499	98.3146	{1,2,4,7,8,9,10,12,13}
Backward Elimination with Pruning	26.025	98.3146	{1,2,4,7,8,9,10,12,13}

5.3 Output Trace for Forward Selection Algorithm on Wine Dataset

```
Running nearest neighbor with all  13  features, using "leaving-one-out" evaluation, I get an accuracy of  95.5056179775 %

Beginning search.

Using feature(s) { 1 } accuracy is  59.5505617978 %
Using feature(s) { 2 } accuracy is  55.0561797753 %
Using feature(s) { 3 } accuracy is  38.202247191 %
Using feature(s) { 4 } accuracy is  39.3258426966 %


Feature set{ 7 } was best, accuracy is  70.2247191011 %

Using feature(s) { 7,1 } accuracy is  89.3258426966 %
Using feature(s) { 7,2 } accuracy is  74.1573033708 %
Using feature(s) { 7,3 } accuracy is  73.595505618 %


Feature set{ 7,10,13,11,1,5 } was best, accuracy is  97.7528089888 %

Using feature(s) { 7,10,13,11,1,5,2 } accuracy is  98.3146067416 %
Using feature(s) { 7,10,13,11,1,5,3 } accuracy is  96.0674157303 %
Using feature(s) { 7,10,13,11,1,5,4 } accuracy is  97.191011236 %


Feature set{ 7,10,13,11,1,5,2,12,6,9 } was best, accuracy is  98.8764044944 %

Using feature(s) { 7,10,13,11,1,5,2,12,6,9,3 } accuracy is  95.5056179775 %
Using feature(s) { 7,10,13,11,1,5,2,12,6,9,4 } accuracy is  97.191011236 %


(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 7,10,13,11,1,5,2,12,6,9,4,3 } was best, accuracy is  96.0674157303 %

Using feature(s) { 7,10,13,11,1,5,2,12,6,9,4,3,8 } accuracy is  95.5056179775 %


(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 7,10,13,11,1,5,2,12,6,9,4,3,8 } was best, accuracy is  95.5056179775 %

Finished search!! The best feature subset is { 7,10,13,11,1,5,2,12 },with Accuracy 98.8764044944 %

('total time elapsed', 26.33501410484314)

Process finished with exit code 0
```

5.4 Output Trace for Backward Elimination Algorithm on Wine Dataset

```
(Warning, Accuracy has decreased!
Feature set{ 10,13 } was best, removing 12  has the highest accuracy, 92.6966292135 %

Using feature(s) { 13 } accuracy is  66.2921348315 %
Using feature(s) { 10 } accuracy is  64.0449438202 %

(Warning, Accuracy has decreased!
Feature set{ 13 } was best, removing 10  has the highest accuracy, 66.2921348315 %

Finished search!! The best feature subset is { 1,2,4,7,8,9,10,12,13 },which has an accuracy of  98.3146067416 %
('total time elapsed', 34.14992809295654)

Process finished with exit code 0
```

5.5 Output Trace for Backward Elimination with Pruning Algorithm on Wine Dataset

```
Using feature(s) { 1,2,4,7,8,9,10,13 } accuracy is  95.50561797753 %
Using feature(s) { 1,2,4,7,8,9,10,12 } accuracy is  92.6966292135 %

(Warning, Accuracy has decreased!
Feature set{ 1,2,4,8,9,10,12,13 } was best, removing 7  has the highest accuracy, 97.7528089888 %

('total time elapsed', 26.025707006454468)
Finished search!! The best feature subset is { 1,2,4,7,8,9,10,12,13 },which has an accuracy of  98.3146067416 %

Process finished with exit code 0
```

Table 5: Accuracy of Forward Selection and Backward elimination for different number of features

Number of features	Forward Selection	Backward Elimination
1	70.2247	66.2921
2	92.6966	92.6966
3	96.6292	92.6966
4	96.6292	94.9438
5	97.1910	95.5056
6	97.7528	97.1910
7	98.3146	96.6292
8	98.8764	97.752
9	98.3146	98.3146
10	98.8764	97.7528
11	97.1910	97.1910
12	96.0674	97.1910
13	95.5056	95.5056

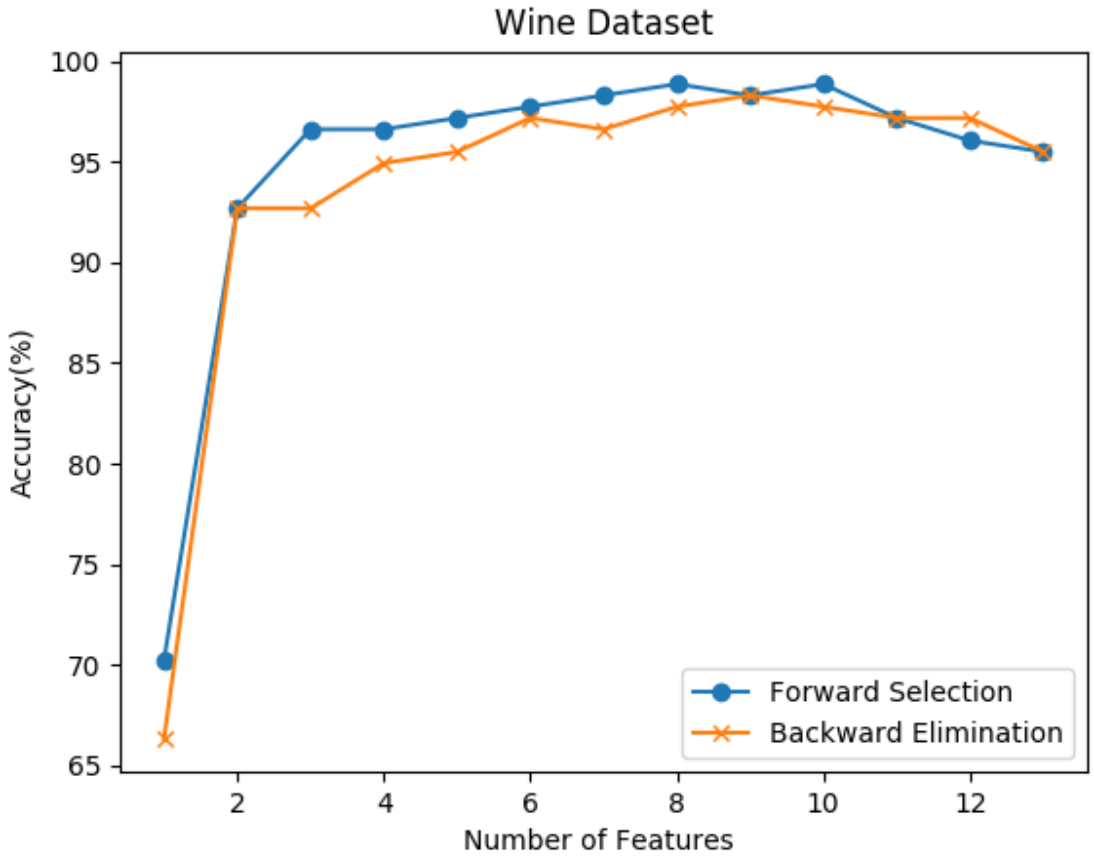

Figure 3:Wine Dataset

6. Acknowledgements

I would like to thank, Prof. Eamonn Keogh, for making us understand the Search algorithms in a much better way, making the concepts of Machine learning more clear and also giving me an opportunity to do this project on Feature Selection. I learnt a lot from the two programming assignments that were given with this course. I feel that I have learnt a lot on different algorithms used in Artificial Intelligence. Also, this course has made my coding skills much better to what it was before to this quarter.

7. Output of Feature Selection :-

```
/Users/pratheek/venv/untitled51/bin/python /Users/pratheek/PycharmProjects/untitled5/final.py
Welcome to Pratheek's Feature Selection Algorithm
Select the Dataset file:cs9.txt
Please wait while I normalize the data!
Data Normalized.
This dataset has  10  features and  100  instances.

1) Forward Selection
2) Backward Elimination
3) Pratheek's Algorithm
Select the Algorithm:1
Running nearest neighbor with all  10  features, using "leaving-one-out" evaluation, I get an accuracy of  60.0 %

Beginning search.

Using feature(s) { 1 } accuracy is  66.0 %
Using feature(s) { 2 } accuracy is  71.0 %
Using feature(s) { 3 } accuracy is  73.0 %
Using feature(s) { 4 } accuracy is  67.0 %
Using feature(s) { 5 } accuracy is  70.0 %
Using feature(s) { 6 } accuracy is  73.0 %
Using feature(s) { 7 } accuracy is  69.0 %
Using feature(s) { 8 } accuracy is  81.0 %
Using feature(s) { 9 } accuracy is  64.0 %
Using feature(s) { 10 } accuracy is  72.0 %


Feature set{ 8 } was best, accuracy is  81.0 %

Using feature(s) { 8,1 } accuracy is  87.0 %
Using feature(s) { 8,2 } accuracy is  79.0 %
Using feature(s) { 8,3 } accuracy is  82.0 %
Using feature(s) { 8,4 } accuracy is  88.0 %
Using feature(s) { 8,5 } accuracy is  77.0 %
Using feature(s) { 8,6 } accuracy is  78.0 %
Using feature(s) { 8,7 } accuracy is  89.0 %
Using feature(s) { 8,9 } accuracy is  82.0 %
Using feature(s) { 8,10 } accuracy is  93.0 %


Feature set{ 8,10 } was best, accuracy is  93.0 %
(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 8,10,7,2,6 } was best, accuracy is  81.0 %

Using feature(s) { 8,10,7,2,6,1 } accuracy is  74.0 %
Using feature(s) { 8,10,7,2,6,3 } accuracy is  72.0 %
Using feature(s) { 8,10,7,2,6,4 } accuracy is  68.0 %
Using feature(s) { 8,10,7,2,6,5 } accuracy is  77.0 %
Using feature(s) { 8,10,7,2,6,9 } accuracy is  73.0 %


(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 8,10,7,2,6,5 } was best, accuracy is  77.0 %

Using feature(s) { 8,10,7,2,6,5,1 } accuracy is  71.0 %
Using feature(s) { 8,10,7,2,6,5,3 } accuracy is  69.0 %
Using feature(s) { 8,10,7,2,6,5,4 } accuracy is  69.0 %
Using feature(s) { 8,10,7,2,6,5,9 } accuracy is  77.0 %


(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 8,10,7,2,6,5,9 } was best, accuracy is  77.0 %

Using feature(s) { 8,10,7,2,6,5,9,1 } accuracy is  71.0 %
Using feature(s) { 8,10,7,2,6,5,9,3 } accuracy is  70.0 %
Using feature(s) { 8,10,7,2,6,5,9,4 } accuracy is  71.0 %


(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 8,10,7,2,6,5,9,1 } was best, accuracy is  71.0 %

Using feature(s) { 8,10,7,2,6,5,9,1,3 } accuracy is  66.0 %
Using feature(s) { 8,10,7,2,6,5,9,1,4 } accuracy is  64.0 %


(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 8,10,7,2,6,5,9,1,3 } was best, accuracy is  66.0 %

Using feature(s) { 8,10,7,2,6,5,9,1,3,4 } accuracy is  60.0 %


(Warning, Accuracy has decreased!  Continuing search in case of local maxima) Feature set{ 8,10,7,2,6,5,9,1,3,4 } was best, accuracy is  60.0 %

Finished search!! The best feature subset is { 8,10 },with  93.0 %Accuracy
('total time elapsed', 4.09577202796936)

Process finished with exit code 0
```

8. Source Code: -

First Page.

```
from scipy import stats
import math
import operator
import time
import sys

# Import Dataset
def getDataset(dataset_file):
   with open(dataset_file) as file:
      data = file.readlines()
      dataset = []
      for line in data:
         try:
            numbers = line.lstrip(" ")
            numbers = [float(number) for number in numbers.split()]
            numbers[0] = int(numbers[0])
            dataset.append(numbers)
         except ValueError:
            print("Error occured on line"+line)
   return dataset

# Normalize the Instances between 1 and -1
def normalize(activeDataSet):
   new_data = stats.zscore(activeDataSet)
   return new_data

# Find the Eucledian Distance
def eucledianDistance(active,test,train):
   distance= 0
   for i in range (len(active)):
      if active[i]:
         distance += pow((test[i]-train[i]),2)
   return math.sqrt(distance)

# Leave-one Out Cross Validation
def leaveOneOutValidation(active, trainingSet, testInstance):
   knn=1
   dis = []
   for x in range(len(trainingSet)):
         dist = eucledianDistance(active, testInstance, trainingSet[x])
         dis.append((trainingSet[x], dist))
   dis.sort(key=operator.itemgetter(1))
   neighbors = []
   for x in range(knn):
      neighbors.append(dis[x][0])
   return neighbors

# Calculate accuracy
def knnAccuracy(data,activation):
   accuracy = 0.00
   for i in range(len(data)):
      instancesRemain = list(data)
      leaveoneInstance = instancesRemain.pop(i)
      neighbors = leaveOneOutValidation(activation, instancesRemain,leaveoneInstance)
      if (len(neighbors) == 1):
         for x in range(len(neighbors)):
```

Last Page.

```
def pratheek(data,features): #Backward Elimination with pruning
   active = [1] * len(data[0])
   active[0] = 0
   accuracy = knnAccuracy(data, active)
   print "Running nearest neighbor with all " , features, " features, using \"leaving-
one-out\" evaluation, I get an accuracy of " , accuracy, "%\n"
   print "Beginning search.\n"
   start = time.time()
   posFlags = [i for i in range(1, features + 1)]
   featureSet = [i for i in range(1, features + 1)]
   bestFeatureSet = [i for i in range(1, features + 1)]
   globalAccuracy = 0.0
   for i in range(features - 1):
      retValue = backwardSubsets(data, posFlags, featureSet, globalAccuracy)
      featureSet = retValue[0]
      localAccuracy = retValue[1]
      if (localAccuracy < globalAccuracy):
         end = time.time()
         total = end - start
         print("total time elapsed", total)
         print"Finished search!! The best feature subset is {", ','.join(str(i) for i
in bestFeatureSet), "},which has an accuracy of ", globalAccuracy, "%"
         sys.exit()
      if (localAccuracy > globalAccuracy):
         globalAccuracy = localAccuracy
         bestFeatureSet = list(featureSet)

   print"Finished search!! The best feature subset is {", ','.join(str(i) for i in
bestFeatureSet), "},which has an accuracy of ", globalAccuracy, "%"

# Main
def main():
   print ("Welcome to Pratheek's Feature Selection Algorithm")
   testset = raw_input('Select the Dataset file:')
   dataset=getDataset(testset)
   print("Please wait while I normalize the data!")
   new_data=normalize(dataset)
   print("Data Normalized.")
   instances = len(new_data)
   features = len(new_data[0]) - 1
   print "This dataset has ", features, " features and " \
      , instances, " instances.\n"
   print "1) Forward Selection"
   print "2) Backward Elimination"
   print "3) Pratheek's Algorithm"
   choice = int(input("Select the Algorithm:"))
   if (choice == 1):
      forwardSelection(new_data,features)
   elif (choice == 2):
      backwardElimination(new_data,features)
   elif (choice == 3):
      pratheek(new_data,features)
   else:
      print("The choice is not correct. Please select a valid choice")

if __name__ =='__main__':
   main()
```