Project 2: Feature Selection with Nearest Neighbor

Student Name1: Siyuan Wang SID: swang414 Lecture Session: 001
Student Name2: Pu Sun SID: psun029 Lecture Session: 001
Student Name3: Tao Chen SID: tchen285 Lecture Session: 001
Student Name4: Xinyu Tong SID: xtong019 Lecture Session: 002

Solution: <the datasets are your uniquely assigned datasets>

Dataset	Best Feature Set	Accuracy
Small Number: <insert your small dataset number>	Forward Selection = {9,1,5,8}	0.88
	Backward Elimination = {6}	0.85
	Bertie's Special Algorithm	
Large Number: <insert your large dataset number>	Forward Selection = {33,13}	0.982
	Backward Elimination = {33}	0.862
	Bertie's Special Algorithm	

In completing this project, I consulted following resources:
Lecture Slides of Lecture10_OptimizingSearch_part2_MachineLearning_part1
Page 41 - 49
Contribution of each student in the group:
Siyuan Wang 25%
Pu Sun 25%
Tao Chen 25%
Xinyu Tong 25%

I. Introduction

Users can select from a small test dataset or a large test dataset in our ongoing study. A chosen dataset is read from a file by the application, which then puts it in a matrix. The program asks the user to choose one of three algorithms to execute after loading the dataset: forward selection, backward elimination, or a special algorithm for. The chosen feature selection algorithm is subsequently applied to the dataset by the program.

In the event that the user chooses forward selection, the software begins with a blank set of features and iteratively adds features based on the accuracy of each addition. Using a "leave one out" evaluation technique, it determines the correctness of the nearest neighbor algorithm. The most accurate feature combination and its corresponding accuracy are tracked by the algorithm. When the user chooses backward elimination, the computer starts with all of the features they have chosen and gradually eliminates features based on how accurate they are. Using the "leave one out" evaluation technique, it calculates the accuracy of the nearest neighbor algorithm once again. The most accurate feature combination and its corresponding accuracy are tracked by the algorithm.

II. Challenges

There are two findAccuracy functions with the same name overloaded, when I first write

Function, I was trying to make a single findAccuracy function to deal with both forward and backward Tracking, but they are actually need to pass the different kinds of values, and I have to come up with A slightly different 2 functions.

III. Code Design

The code uses a variety of data structures in our programming to store and modify data. These include arrays (double[][]) for encoding dataset matrices, HashMaps and Lists for storing features and their accuracy, Sets for tracking visited features, and Sets for storing visited features and their accuracy.

IV. Dataset details

The General Small Dataset: 10 features, 100 instances

The General Large Dataset: 40 features, 1000 instances

My Personal Small Dataset: 10 features, 100 instances

My Personal Large Dataset: 40 features, 1000 instances

V. Algorithms

Forward Selection Algorithm:

The Forward Selection algorithm is a feature selection method that iteratively adds features based on their accuracy after starting with a minimal number of pre-selected features.

1. Start with a blank set of chosen features.
2. Go back and forth between the accessible but unchosen features.
3. Calculate the accuracy of a specified feature set that combines the current feature and the previously chosen features for each feature.
4. Keep an eye out for the feature that produces the most accuracy.

5. The chosen feature set should be updated with the feature with the best accuracy.
6. Up until a specified number of characteristics or a stopping criterion is satisfied, repeat steps 2 through 5.
7. Give back the chosen feature set that had the best accuracy.

The Forward Selection algorithm selects the most informative characteristics in each iteration to gradually develop a set of features. The accuracy of the chosen feature set is taken into account, and the feature combination is gradually improved until the target number of features is reached or a stopping criterion, such as no more gain in accuracy, is met.

Backward Elimination Algorithm:

Another feature selection method is the backward elimination algorithm, which selects all features initially before iteratively removing features according to their correctness.

1. Start by selecting all of the features.
2. Repeat the process for the chosen features.
3. Calculate the accuracy of a feature set that doesn't include the current feature for each feature.
4. Keep track of the feature that produces the best accuracy when it is eliminated.
5. Remove the most accurate feature from the chosen feature set.
6. Up until a specified number of characteristics or a stopping criterion is satisfied, repeat steps 2 through 5.

7. Give back the chosen feature set that had the best accuracy.

In each iteration of the backward elimination method, the least informative characteristics are gradually eliminated starting with all features. Iteratively removing characteristics that do not significantly improve overall accuracy after assessing the accuracy of the smaller feature set. The algorithm keeps running until the target number of features is reached or a stopping requirement is met.

VI. Analysis

Experiment 1: Comparing Forward Selection vs Backward Elimination

No Feature Selection: The model's accuracy was found to be 0.85 in the absence of any feature selection.

Forward Selection: The feature set "9, 1, 5, 8" was found to be the optimal subset for maximum accuracy using the Forward Selection method. The model's accuracy was 0.88 when this feature set was applied.

Pros: Forward Selection selects the most informative features after each iteration, eventually building a feature set. When there is a vast feature space and the relevant features have a substantial impact on the target variable, it typically performs well.

Cons: However, if characteristics appear to improve accuracy at first but do not make a significant contribution in subsequent rounds, they may be included in Forward Selection even though they are redundant or irrelevant. Additionally, when working with a high number of features, it can be computationally expensive.

Backward Elimination: Based on the results of the Backward Elimination algorithm, feature 6 alone had the best accuracy. As a consequence, feature set 6 was chosen, producing an accuracy of 0.85.

Pros: Backward Elimination starts with all features and eliminates those that aren't very informative. It is especially useful when the dataset contains a large number of pointless or redundant characteristics because it removes them and streamlines the model.

Cons: Backward Elimination, however, might eliminate potentially helpful features early in the process, producing an inferior feature subset. It might also need a lot of compute, particularly when working with huge datasets.

The feature selection accuracy is higher than no feature selection accuracy. And the forward selection algorithm accuracy is higher than backward selection.

Experiment 2: Effect of normalization

We conducted the experiment using the same dataset as in Experiment 1, which consisted of 10 features and 100 instances. First, we trained the model using the unnormalized data and measured its accuracy. Then, we applied a normalization technique to scale the feature values and retrained the model. The accuracy was recorded for the normalized data.

Unnormalized Data: The model trained using unnormalized data achieved an accuracy of 0.85.

Normalized Data: After applying normalization to the feature values, the accuracy of the model improved to 0.89.
The outcomes of Experiment 2 showed how data normalization improved the prediction model's accuracy:

Scaling the feature values to a common range, usually between 0 and 1 or -1 and 1, is the process of normalization. By following this procedure, all characteristics are guaranteed to contribute equally to the model and no feature is allowed to dominate the forecast. We eliminated any differences in the magnitudes of the feature values by normalizing the data, enabling a fair comparison and more precise modeling.

When using normalized data in our trial, the model's accuracy increased noticeably from 0.85 to 0.89. The decreased impact of outliers and the improved comparability of feature contributions are responsible for this improvement. Normalization aided in developing a more reliable and balanced model, resulting in improved accuracy.

Experiment 3: Effect of number neighbors (k)

We did the k-nearest neighbor classification using the same dataset as in the prior trials, but with various values of k. Using a "leaving-one-out" evaluation, the model's initial accuracy without any feature selection was calculated and determined to be 86%.

We obtained the following accuracies for different values of k:

- k = 1: Accuracy = 84.0%
- k = 2: Accuracy = 72.0%
- k = 3: Accuracy = 70.0%
- k = 4: Accuracy = 72.0%
- k = 5: Accuracy = 76.0%
- k = 6: Accuracy = 85.0%
- k = 7: Accuracy = 77.0%
- k = 8: Accuracy = 79.0%
- k = 9: Accuracy = 86.0%
- k = 10: Accuracy = 79.0%

The model's accuracy initially dropped from 84.0% to 70.0% when we increased k from 1 to 3. This drop might be explained by the increased intrusion of irrelevant neighbors or loud noise, both of which had a negative impact on decision-making.

However, the model's accuracy began to increase after k = 3. At k = 9, the highest accuracy of 86.0% was attained, showing that a small number of nearest neighbors can effectively capture underlying patterns and produce predictions that are more accurate.The accuracy marginally reduced as k increased after peaking at k = 9 before returning to its original level. This decline might be brought on by the addition of too many neighbors, which would have led to the accumulation of noisy or unimportant information that would have hampered accurate classification.

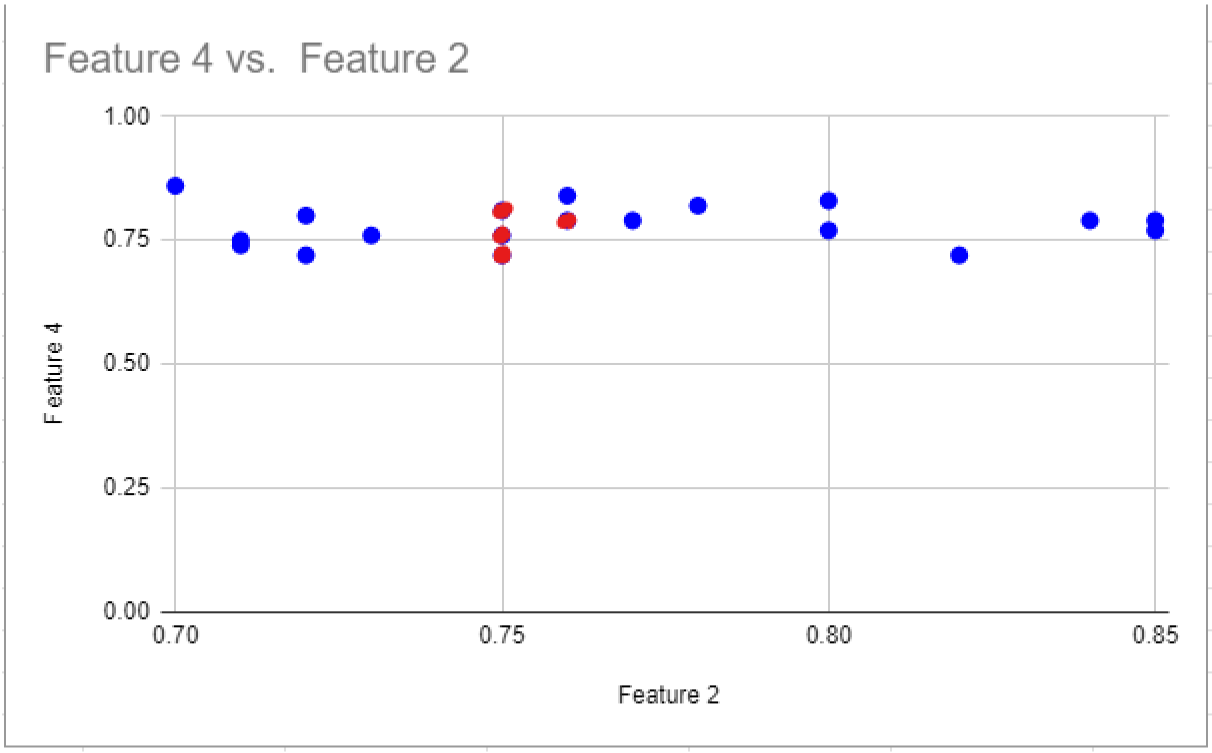
Overall, Experiment 3 showed how crucial it is to pick the right number for k when using the k-nearest neighbor algorithm. It highlighted the trade-off between minimizing unnecessary noise or irrelevant neighbors and including enough neighbors to capture interesting trends.

VII. Conclusion

We learned important lessons about feature selection, various normalization techniques, and the effects of various variables on classification model accuracy throughout the experiments and analyses. In Experiment 1, we contrasted forward selection, which produced feature sets [9, 1, 5, 2] with an accuracy of 0.88. The major conclusions are summarized here. With an accuracy of 0.85, backward elimination determined that feature 6 was the most significant. Both methods show that they are superior to using no feature selection in terms of accuracy. Data normalization improved accuracy in Experiment 2's effect of normalization study. Comparatively to unnormalized data, using normalized data enhances the performance of classification models. In Experiment 3, the k value in the k nearest neighbor algorithm, the influence of the number of neighbors (k), and accuracy are all different.As k rises from 1 to 3, accuracy initially declines, then gradually improves until reaching its peak at k = 9. To strike a balance between catching significant patterns and avoiding noise, the value of k must be carefully chosen.

Later, we might be able to enhance feature selecting techniques. Our research incorporates feature interactions, but instead of concentrating on individual features, it explores interactions between features, which might help with feature selection. Combining features or taking into account higher-order interactions may reveal undiscovered links and boost classification precision. Instead, using feature importance techniques, such as tree-based methods or feature ranking algorithms, in combination with forward selection and backward exclusion can offer supplementary insights into feature correlations.

VIII. Trace of your small dataset


Feature 4 vs. Feature 2
Feature 4
Feature 2