Feature Selection with Nearest Neighbor

Jasmine Kim

March 20, 2018

1 Introduction

This project was created for Dr. Eamonn Keogh's CS 205 Artificial Intelligence at the University of California, Riverside. For this project, I wrote a program that searches for features using:

- Forward Selection
- Backward Elimination
- Jasmine's Search Algorithm

All three searching algorithms utilize nearest neighbor for classification and leave-one-out cross validation for feature selection. There is a simple text line interface that allows the user to select a text file containing the data. It also allows the user to select the searching algorithm. I chose to use Python3 as my language of choice to implement this project.

2 Algorithms

In this section, I will go over the nearest neighbor classification algorithm. In addition, I will go into explaining the differences of the three search algorithms.

2.1 Nearest Neighbor Classification

The nearest neighbor algorithm searches for the data point closest in distance to the data point you are trying to classify. Once found, the test data point will be assigned the nearest neighbor's class.

2.1.1 Euclidean Distance

For this project, I used Euclidean distance as the distance measure for nearest neighbor. Each feature will has a 1-dimensional data point. So in the case when I am considering 5 features, we are working in a 5-D space. Euclidean distance can easily calculate the data in any dimensional space using the following equation

$$distance(a, b) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

Where i represents the feature. For example, to calculate the Euclidean distance between features 2,4,5, the following would be used to calculate the distance

$$distance(x1, x2) = \sqrt{(x1_2 - x2_2)^2 + (x1_4 - x2_4)^2 + (x1_5 - x2_5)^2}$$

2.1.2 Leave-one-out Cross Validation

In order to measure accuracy, I used a special case of k-fold cross validation called the leave-one-out cross validation. This is because there is not a lot of data per feature. In leave-one-out cross validation, I left one data point out for testing and the rest were used to train. This was done across the entire dataset in order to measure accuracy. Accuracy is calculated by taking the total number of correctly classified over the total number of instances:

$$accuracy = \frac{correct}{total}$$

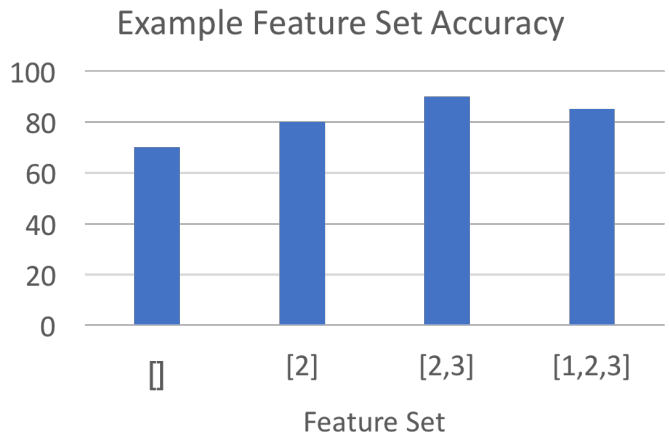

Figure 1: Example feature set versus accuracy.

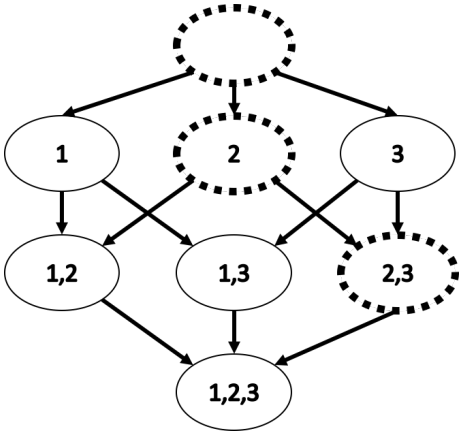

Figure 2: Forward selection example.

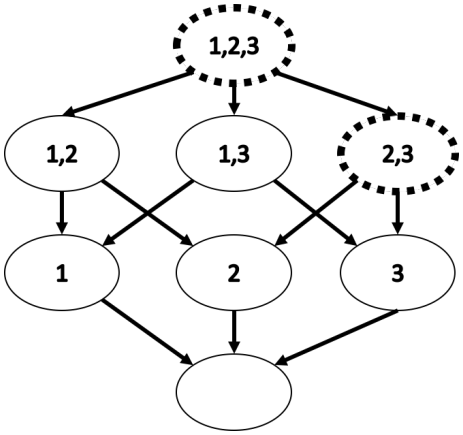

Figure 3: Backwards elimination example

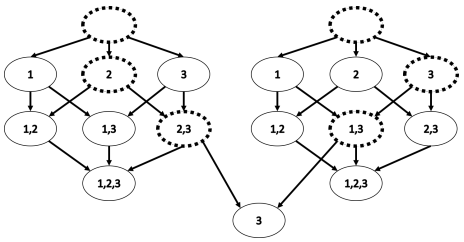

Figure 4: Jasmine's Search Algorithm example first feature

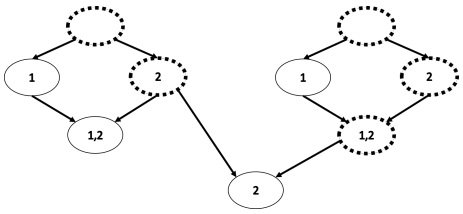

Figure 5: Jasmine's Search Algorithm example second feature

2.2 Search Algorithms

The searching algorithm was used in order to search through the features to find features that provide the highest accuracy when classifying using nearest neighbor and leave-one-out cross validation. In the end, all algorithms output a best feature subset which contains the list of features that provide the best accuracy. In order to reduce the amount of time, the searching algorithms only consider adding the feature to calculate the accuracy if it has not already been added.

In order to better demonstrate these algorithms, Figure 1 shows an example feature set with made-up accuracy results that will be referred to explain forward selection and backwards elimination. In this example, features [2,3] are considered the best features as they provide the highest accuracy.

2.2.1 Forward Selection

Forward selection is a form of greedy forward searching algorithm. The initial search space is an empty set with no features where the accuracy is the default rate and the operator is to add a single feature. The algorithm will add a feature and measure the accuracy using nearest neighbor and leave-one-out cross validation as mentioned above. It will then select the feature with the highest accuracy and add it to the best feature subset.

This is better explained through an example in Figure 2. Figure 2 shows using dotted lines the path of feature selection based on the accuracy in the example Figure 1. The search space starts off with no features and in this case will add feature 2 and feature 3. However, it will stop adding features to the best feature list as the accuracy decreases with the addition of 1.

2.2.2 Backward Elimination

Backward elimination is another form of greedy searching algorithm. However, unlike forward selection, the initial search space is the entire feature set where the accuracy is the default rate and the operator is to remove a single feature. The algorithm will remove a feature and measure the accuracy using nearest neighbor for classification and leave-one-out cross validation. It will then select the feature set with the highest accuracy and permanently remove the feature that provided the highest accuracy with its removal.

Again, this is better explained through an example in Figure 3. Figure 3 shows using dotted lines the path of feature selection based on the accuracy in the example Figure 1. The search space starts off with all features and in this case will remove feature 1. However, it will stop removing features from the best feature list as the accuracy decreases with the removal of feature 2 or 3.

2.2.3 Jasmine's Search Algorithm

I attempted to make a more accurate algorithm that does not find spurious features. Spurious features can be captured due to overfitting of the data. I wanted to reduce spurious features by doing the following procedure:

- Resample the dataset by creating 3 copies of the dataset and randomly deleting 20% or 40% of the data. I chose 20% for the small dataset and 40% for the large dataset in order to speed the algorithm up.
- Find the strongest feature from the 3 copies of the dataset using forward selection. Remove the strongest feature from the feature search space for the next run.
- Repeat for the next features until you have the best features.
- Re-calculate accuracy based on the final best features found list.

An example of the Jasmine Search Algorithm (JSA) is shown through Figure 4 and 5. In this example, Figure 4, JSA is attempting to find the first strong feature by resampling twice using forward selection. It finds that both runs found a mutual feature of 3, hence in this case the best and strongest feature is 3. As shown in Figure 5, JSA then removes the strongest feature (feature 3) and searches for the next strongest feature in the new search space. In this case, it finds that both runs found a mutual feature of 2. Therefore as a result, the best features found in this case are features [2,3].

The theory as to why Jasmine's Search Algorithm should reduce spurious feature selections than both Forward Selection and Backward Elimination is because:

- By resampling the dataset, it reduces the likelihood of finding spurious features. This is because through the removal of random datapoints from the training set, the model is less likely to overfit to its data. As a result, it should find less spurious features.
- By removing the strongest feature, it is easier to determine if the other features found is a true feature. This is because the strongest features may overshadow the "goodness" of a weaker feature. Hence by removing a stronger feature, it is easier for the weaker feature to stand out a true feature instead of becoming overshadowed by the stronger feature. This also removes any chance of finding a spurious feature.

2.3 Datasets

The datasets used for this project were the following:
`CS205_SMALLtestdata__35.txt`
`CS205_BIGtestdata__2.txt`
Each of the datasets contain the class in the first column and features in all other columns.

2.3.1 Z-Score Normalization

A weakness in the nearest neighbor algorithm is that it is very sensitive to any measurement unit since it relies on a distance measurement. Hence, a normalization technique is necessary to make sure the data is standardized. I used z-score normalization which is the following:

$$zscore = \frac{value - mean}{\sigma}$$

The mean and standard deviations were calculated per feature or column.

3 Results

This section will go over the best features found and its accuracy from the three different searching algorithms on the two datasets. I also measured the time lapsed in running the three search algorithms.

3.1 Small Dataset

Table 1 shows the results from running the small dataset `CS205_SMALLtestdata__35.txt` using all three search algorithms. This dataset contained 10 features and 100 data points. From running the algorithms, it is clear that feature 6 is the strongest feature, as it appears in all three search algorithms. Feature 3 appears to be the next strongest feature as it appears twice.

Table 1: Feature Selection Results - Small Dataset 35

Feature Search on Dataset 35			
	Best Features	Accuracy	Time (Seconds)
Forward Selection	[6,3,1]	90%	12.59
Backwards Elimination	[6,8,10]	90%	16.27
Jasmine's Search Algorithm	[6,3]	84%	58.13

In terms of time, forward selection is the quickest and JSA is the slowest. JSA would naturally be the slowest as it requires running forward selection multiple times by resampling.

As for accuracy, forward selection appears to have performed the best as it had an accuracy of 90% and was able to pick up the two strongest features unlike backwards elimination which appears to have picked up a weaker or spurious feature 8 and 10.

Jasmine's Search Algorithm performed the worst in terms of accuracy even though its primary goal was to detect less spurious features and outperform the other algorithms. I believe this is because of two reasons. First, JSA uses a resampling method. Resampling requires finding the strongest feature and removing it from the feature space. This can reduce the accuracy because in feature selection, certain features may only perform accurately in combination with another feature. By removing the strongest feature, it removes the possibility of finding those stronger feature sets when used in conjunction with another feature. And secondly, the JSA attempts to find only one feature at a time under the assumption that if one feature performs well, in conjunction accuracy will only increase. However, this is not necessarily the case because the conjunction of the best features together does not represent the best features for the dataset. As shown in Figure 6, In this run, JSA found correctly that features 6 and 3 are the strongest features. However, upon removing features 6 and 3, it found feature 4 was another strong feature but in conjunction with features 6 and 3 decreased the accuracy.

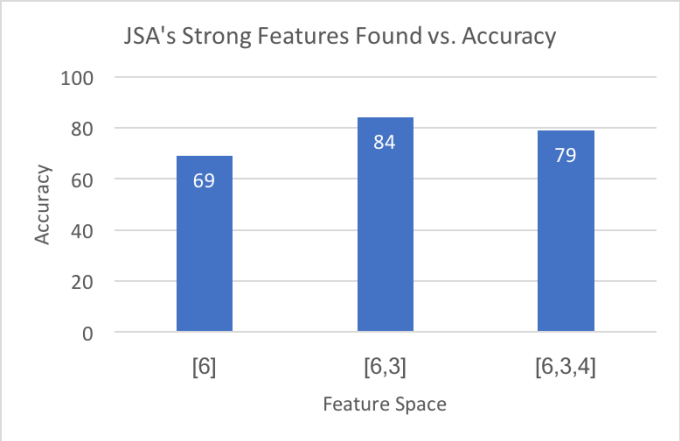

Figure 6: Example feature set versus accuracy.

However, it's worth noting that JSA appears to have outperformed backpropagation in finding the strong features since it was able to find both strong features 3 and 6. This seems to suggest that the accuracy of JSA is lower because it is not overfitted to the data.

3.2 Large Dataset

Table 2 shows the results from running the large dataset `CS205_BIGtestdata__2.txt` using all three search algorithms. This dataset contained 50 features and 100 data points. From running the algorithms, it is clear that feature 1 is the strongest feature. Unlike the smaller dataset, it was more difficult to distinguish what the next strongest feature was in the larger dataset. Since the accuracy of the two algorithms, Forward Selection and Backwards Elimination, were so high, I believe that feature 7 is a spurious feature and not a true feature.

In terms of time, again forward selection is the quickest and JSA is the slowest. This is because the dataset itself will not impact the different in performance in algorithms.

Again, Jasmine's Search Algorithm performed the worst in terms of accuracy. As mentioned in the previous section, I believe it's because of the algorithm's resampling method and because the algorithm attempts to find only one feature at a time.

Table 2: Feature Selection Results - Large Dataset 2

Feature Search on Dataset 2			
	Best Features	Accuracy	Time (Minutes)
Forward Selection	[1,7,48]	97%	14.94
Backwards Elimination	[1,7]	94%	26.85
Jasmine's Search Algorithm	[1,2,21]	77%	49.90

3.3 Analysis

From the analysis mentioned in the previous two sections, I have determined the final results as shown in Table 3. For the smaller dataset, the strong features are features 3 and 6. For the larger dataset, the strong feature is 1.

4 Conclusion

It's clear that in terms of time complexity and literal accuracy, the search algorithms perform in the following from best to worst:

- Forward Selection

Table 3: Final Results

Feature Search on Dataset 35		
	Strong Features	Weaker Features
Dataset 35	6,3	1,8,7,10
Dataset 2	1	7,2,21,48

- Backwards Elimination
- Jasmine's Search Algorithm
  Even though in theory, Jasmine's Search Algorithm should have performed the best in terms of accuracy since its goal was to reduce the likelihood of finding spurious features. However, JSA appears to have outperformed backward elimination in finding the strong features. This suggests that the accuracy is lower for JSA because it is not overfitted to the data. Hence, accuracy by itself is not a good measure for how good feature selection is.

5 Republican vs Democrat - 1984 Congressional Voting Records (Extra Credit)

I retrieved a 1984 Congressional Voting Record dataset from the UCI Machine Learning database. This dataset contains 16 features and class labels that determine whether the member is a Republican or Democrat. I ran both Forwards Selection and Backward Elimination on the dataset to see what mutual features appear in both search algorithms. Table 4 shows the results.

Table 4: Republican vs Democrat Feature Results

Feature Search on 1984 Congressional Voting Records		
	Strong Features	Accuracy
Forwards Selection	[4,1,2,7,6,3,14,9]	96.55%
Backwards Elimination	[4,6,9,11]	96.09%
Mutual Strong Selection	[4,6,9]	94.25%

The following are the features in the dataset and the items in bold are the features that provided the highest accuracy:

1. Handicapped infants
2. Water project cost sharing
3. Adoption of the budget resolution
4. **Physician fee freeze**
5. El Salvador aid
6. **Religious Groups in Schools**
7. Anti-satellite test ban
8. Aid to Nicaraguan contras
9. **MX missile**
10. Immigration
11. Synfuels corporation cutback
12. Education spending
13. Superfund right to sue
14. Crime
15. Duty-free exports
16. Export administration act South Africa

It appears that the three features that are most likely to determine the party affiliation are related to health benefits, religion, and military defense. This makes sense as all three of these areas are areas of big debate between the two parties. Democrats lean towards universal health care, separation of Church and State, and lowering spending on national defense. And Republicans lean towards more availability of health care options, endorsement of faith-based organizations, and strong national defense.