[1]<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.696.724&rep=rep1&type=pdf>

[2]<http://www.jmlr.org/papers/volume5/dy04a/dy04a.pdf>

[3]<http://www.sciencedirect.com/science/article/pii/S000437029700043X>

[4]<https://pdfs.semanticscholar.org/1687/23227a0f337b4c4973d8bc796d4277131b7d.pdf>

[5]<https://arxiv.org/pdf/cs/0212038.pdf> (THIS ONE IS ENTIRELY THEORETICAL AND VERY MATHY)

The problem which featurization aims to solve is that of how to select from all possible features of a dataset those which are actually useful in classifying an object.[3] For a social media account, as an example, a feature set may include number of posts, number of comments, average score of posts, age of account, and frequency of posts. However, not all features necessarily lend themselves well (or are even beneficial at all) to classification. Therefor, a machine learning algorithm needs to be able to pick some subset of all the features to use for evaluation.

Traditionally featurization involves creating a 2 dimensional NxM matrix (**F**) with N samples of M features each, with **F**i,j = 1 if sample **n**i has feature **f**j , and 0 otherwise. However, social media data has additional levels of dimensionality that derive from the level of connection between both user account and posts, and between multiple user accounts. It is possible, using a different model of featurization, to take advantage of these connections.[1] Utilizing 5 separate matrixes to do feature selection allows the neural network to capture the connections between accounts in addition to simply the accounts themselves.

Given that there are two types of items, users and posts, there are 4 types of connections between items. First, there is a CoPost connection, whereby multiple posts are made by the same user. Second, there is CoFollowing, where one user if followed by two other users. Third, there is CoFollowed, where two users are followed by a third user. Finally, there is Following, where one user is followed by another. These relations are characterized by hypothesis named after them, essentially stating that posts made by one user are likely to be similar to each other, and users that are connected by one of these relations are likely to make posts similar to one another.

It is important to note that having too many features can be just as detrimental as having too few. To combat this, we can use the concept of redundancy to identify superfluous features and reduce the baseline feature set.[2] Two features X and Y are redundant if data can be clustered equally based on X independent of Y or Y independent of X. A feature X is irrelevant if use of that feature alone yields only a single cluster. Because picking different feature subsets can yield vastly different clusters and having irrelevant features can obscure the structure being looked at in the data, it is important that we identify these kinds of features. In addition, it is important that we pick the correct subset of features to ensure the clustering that exists is visible and is not obscured by irrelevant features. We want to find the smallest subset of features that shows useful clustering behavior in the data.

Feature searching is done via a greedy approach because a 2d search through the feature space is computationally intractable. The approach used is to add features to the final set if that feature improves on the results of the already selected feature, and to discard it if it does not. The search ends when adding new features no longer improves on the result. The advantage of this is that it is fast (O(d2) rather than O(2d)) and generally gives useable results. However, it is not optimal.

One way of achieving this is by using what is referred to as a wrapper, whereby the algorithm used to classify objects is itself used on a training set to evaluate the set of features, allowing the overall machine learning algorithm which set is most helpful in classifying the data. The other approach to this problem is by using a filter, in which case the final classification algorithm is distinct from the algorithm used to select features. The disadvantage to the filter approach is that it does not take into account the effect of selecting a feature on the performance of the final classification algorithm.

A possible issue with using wrapper-type feature selection is that it uses the training data itself to select the feature set.[4] This means that not only is the learning algorithm potentially vulnerable to overfitting, but the feature set itself may be. The source of this lies in the specific way these algorithms generally implement the feature selection. They start with either a full or empty set of features, and then add/remove one feature at a time until no further improvements are possible. The problem with this is that it ignores possible interactions between 2 or more features, where the addition of multiple features at the same time may yield an improvement even though single features don’t. Beyond that, these greedy algorithms cannot be improved without quickly making the problem computationally intractable.

A potential solution to this is to replace the greedy algorithm with a genetic one, whereby top-performing feature sets are selected and combined using crossover and mutation operators to produce new feature sets, which are then evaluated and combined in turn. This allows for potential variation far beyond the greedy algorithm without becoming a potentially O(2n) problem. In experiments with artificial data specifically designed to trap a greedy algorithm in a local maximum, with a feature space of 230 a genetic algorithm consistently outperformed a greedy one. While the genetic algorithm did not always find the global optimum, it was consistently better. Re-running these tests on a set of breast cancer data, the genetic algorithm again outperformed the greedy algorithm, confirming the initial results.

Overfitting is what happens when the classification algorithm gets data in the training set correct to the point that it begins to introduce error into the testing set classifications. For example, if an algorithm was asked to classify if a patient had heart disease based on the patient’s SSN, heart rate, and blood pressure, it would evaluate (perfectly) on the training set based on the patient’s SSN. However, this classification strategy would not hold whatsoever on any other set of data because SSNs are unique to each patient and therefor this evaluation is not generally applicable. According to this paper, however, this is mainly a problem when 1) the training dataset is small or 2) the feature selection algorithm used does not control for this. This can be done in a filter approach by testing the relevancy of a feature (by using that feature to classify a subset of the data not used in training), or if the wrapper approach is being used it is taken care of automatically. However, for small feature sets the wrapper approach is still vulnerable to overfitting in more subtle ways.

It is possible to further pair down feature sets using the concepts of relevancy and by considering how contextually connected a feature may be to any other feature.[5]

A feature x is considered strongly relevant if and only if there exists some pair of features x, s such that x /= s and the probability of this instance being classified as y given x and s is not the same as the probability of the instance being classified as y given s and the probability of both x and s occurring is greater than 0. A feature x is weakly relevant if and only if there exists some subset of features S’ /= {} such that the probability of the instance being classified as y given x and s’ is not the same as the probability of the instance being y given s’. This second definition allows for there to exist two features x1 = x2. A feature is irrelevant if it is neither strongly nor weakly relevant. A learner should use all strongly relevant features and discard all irrelevant features. Some weakly relevant features may be used.

A feature is primary when the smallest subset of features for which it is relevant is of cardinality 0. That is to say, the probability of an instance being classified as y is not the same as the probability of it being classified as y given x. Otherwise that feature is contextual. A feature may be either contextual or primary if it is weakly relevant. It must be primary if it is strongly relevant.

A feature x1 is weakly context sensitive to x2 if there exists some subset of (S - {x1,x2}) = S’ such that the probability of an instance being classified as y given x1, x2, s’ is not the same as the probability of an instance being classified as y given x2, s’, and the probability of an instance being classified as y given x1, x2, s’ is not the same as the probability of an instance being classified as y given x1, s’. This implies that x1 is weakly context sensitive to x2 iff x2 is weakly context sensitive to x1. A feature x1 is strongly context sensitive to x2 if x1 iff x1 is a primary feature and x2 is a contextual feature and x1 is weakly context sensitive to x2.

These concepts can be used to identify which features are useful, which are redundant, and even which features may only be useful when considered in relation with another set of features. At the moment there is not a lot of work in the ways of practical application for context sensitivity, but it is a strong idea that may potentially make featurization a far faster and easier problem to solve.

Conclusion

Featurization, specifically the selection of a subset of a full feature set is a tremendously difficult problem. The naive solution (which is also the best solution) to simply search through the feature space is computationally intractable, which means the focus of research is and will remain on ways to as closely approximate the naive solution as is possible while staying within the practicality bounds of modern hardware. There are a number of common ways to approach this, but they all generally rely on using one of a small number of algorithms. Two such methods are with a greedy algorithm, one which starts with the empty set of features and adds to it, and one which starts with the full set and removes features. The advantage to the first is that it is much faster to learn on a small set of features, so each iteration of the process can be done that much quicker. However, in general having more features leads to a more accurate classifier and so the second can lead to a better result if it can be reasonably applied to the problem.

Another possible approach disregards the iterative process entirely, instead using an unrelated heuristic to determine the best set of features. The advantage to this is that testing a feature set against the classification algorithm itself is time-intensive, however it is generally worth it as disregarding that step means that there is no way to be certain that the feature set selected will be in any way good for the classification task.

A third approach, and one I personally favor, is to use a genetic algorithm to select the features, using the classification algorithm as the measure of fitness. This sidesteps the issues related to greedy algorithms when two features together may improve the results but neither on their own do so, which would cause the algorithm to disregard both. Genetic algorithms do not suffer from this. Genetic algorithms are not optimal, however they are generally applicable and tend to perform better on both artificial and real datasets. There is a small downside, which is that genetic algorithms are less efficient on small feature sets, but this is a negligible downside when compared to the increased performance and smaller risk of falling into local optimums.

In addition to the multitude of general approaches to feature selection, there are a number of aspects that can be applied to more specific problem domains such as social media. Social media contains contextual relevance both inside of a single datum and between multiple datum, which both increases the difficulty of processing and classifying the feature sets and increases the accuracy of a correctly processed feature set to be classified. Not much research has been done towards applying the concepts of relevancy and contextual sensitivity, as it is a relatively recently defined concept and is not generally applicable, but I believe if applied correctly it could result in both faster and more accurate feature selection. Similarly, research into the interconnectedness of items done here at ASU has been done, but it is very recent and has not been deeply examined or applied. These together have the potential to significantly increase the performance and speed of learning when applied to social media.

<https://arxiv.org/pdf/1111.4297.pdf> (Manual feature selection, little advanced math)

Based on statistical analysis of paid posters’ activity, it is determined that 84.3% have less than 50% of posts being replies to other users. This is opposed to 73% of legitimate users who have the majority of their activity being replies to other users. Furthermore, 78.57% of paid users post with a frequency below 5 minutes between posts, while only 56.85% of legitimate users match this frequency. By another measure, the duration with which a user maintains a single alias it was found that the activity of a paid and legitimate user are extremely similar, with 98.57% of paid users abandoning a given alias after 3 days and 90.19% of legitimate users doing the same. Finally it was found that the activity as measured by the number of distinct articles posted on was also similar, with paid and legitimate users commenting on 5 or fewer articles with 97.14% and 97.35% probability, respectively.

By far the most distinctive feature of a paid poster was done using a simple lexical analysis of posts. First, break a comment into words. Second, count the number of posts with an 80% or greater similarity. Using this method, it was found that fully 78.57% of paid commenters have 6 or more similar posts, while only 4.20% of legitimate users have the same. Conversely, it was found that 79.56% of legitimate posters have 0 similar posts compared to 5.71% of paid posters.

Using only 2 of the original measures, it is possible to achieve accuracy of 64.12%. However, by these measures only 2 of the 82 paid posters where identified (it did correctly label all legitimate users). Using all 4 of the statistical measures, an accuracy of 62.78% was achieved, with the algorithm correctly identifying 32 paid posters but mislabeling 33 legitimate posters. Finally, by adding the lexical analysis, an accuracy of 88.79% was achieved. Using all features resulted in 3 mislabeled legitimate posters and 60 of the 82 paid posters being correctly labeled.