Multi-Label Classification with k-Nearest Neighbor

Branley Mmasi ‘25, worked with David Liu ‘24
Swarthmore College, Department of Computer Science

Overview

Problem: Real-world data often has multiple labels (e.g., a movie can be "Action" & "Comedy"). Standard classification struggles with label independence.
Our Approach: We modified k-Nearest Neighbors (k-NN) to better handle multi-label data.
Key Idea: Weight neighbors based on predicted label similarity (Hamming Loss) to address label correlations for more accurate predictions.
Goal: We follow Chiang et al.'s approach to improve multi-label classification accuracy using a simplified weighted k-NN.

Motivation

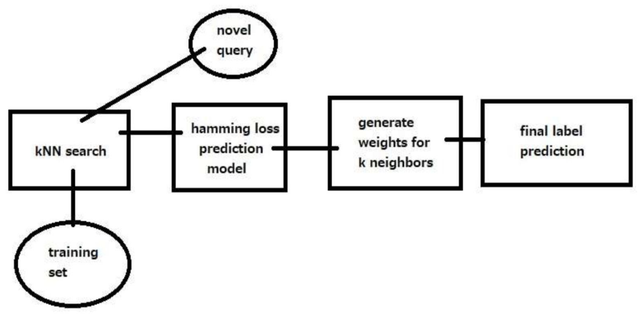
Labels are Correlated: Multi-label datasets exhibit inherent label correlations, i.e., "Action" and "Sci-Fi" movies are related. Ignoring this hurts performance.
Traditional k-NN is Naive: Treats all neighbors uniformly, neglecting varying neighbor relevance in multi-label contexts. Some neighbors are more "trustworthy" than others for multi-label prediction.
Our Solution: Weight neighbors by predicted Hamming Loss as a proxy for label similarity and neighbor "trustworthiness." Neighbors with similar labels get higher weight.
Dataset & Task: Enzyme Commission (EC) class prediction for substrate molecules (1039 instances, 196 features, 6 labels). Relevant to bioinformatics and drug discovery.

Methods

We used Hamming Loss as both our evaluation metric and to generate our weights, defined as the percentage of labels different between two labelsets

$$HammingLoss(L1, L2) = |L1 \delta L2| / |L|$$

Chiang et al.'s paper describes the training of a global ranking model based on neighbors' 'trustworthiness,' then using a weighted voting strategy to predict final label set of query point.
Due to time and knowledge constraints, we were unable to implement a ranking model, and
simplified their generalized pattern search weighting method to directly use Hamming Loss to generate weights.
Our overall approach for modified k-NN:



Training:

Find k nearest neighbors of each training point, z.
For each neighbor, create new point q containing various distances between z and neighbor.
Calculate Hamming Loss between z and neighbor.
Train a Linear Regression model where q are the inputs and the Hamming Losses are its predictions.

Predicting:

Find the novel points k nearest neighbors and generate a as above
Use Linear Regression Model to predict Hamming Loss of Neighbors, and assign weights to neighbors with

$$W = 1 - PredictedHammingLoss$$

Produce a weighted average score for each label:

$$Label_i(x) = \sum_{j=1}^{k} w_j * Y_k, j / \sum_{j=1}^{k} w_j$$

Results

Using k-NN applied independently to all labels as a baseline metric. Shown are the Hamming Losses:

K	Traditional	Modified
3	0.34	0.54
5	0.31	0.58
7	0.31	0.60
11	0.31	0.34
15	0.29	0.33
25	0.28	0.28

Traditional k-NN (Baseline): Consistent Hamming Loss around 0.3 across different k values.
Modified k-NN: *Worse* than traditional k-NN at low k values (k=3, 5, 7).
- *Similar* to traditional k-NN at higher k values (k=11, 15, 25). No significant improvement over baseline.

Discussion

Weighting Needs More Data at Low *k*: Complex weighting needs enough neighbors (higher k) to be effective.
Hamming Loss Prediction Not Optimal Weighting: Direct Hamming Loss prediction might not be the best way to weight neighbors.
Dataset Simplicity: Dataset might be too simple for complex model to show advantage.
Key Learning: Simplified weighting based on Hamming Loss alone isn't enough for significant improvement. Ranking neighbors (as in original paper) is likely crucial.

Acknowledgements

I would like to thank Dr. Ben Mitchell for his feedback and support.

References

Chiang, Tsung-Hsien, Lo, Hung-Yi, and Lin, Shou-De. A ranking-based knn approach for multi-label classification. 2012.

Srivastava, Gopal N. Multi-label classification of enzyme substrates.