Predicting Wine Quality Using KNN

Bibek Karki
Department of Computer Science,
bkarki3@huskers.unl.edu

Biquan Zhao
School of Natural Resources, University of Nebraska - Lincoln
bzhao10@huskers.unl.edu

Tyler Zinsmaster
Department of Computer Engineering, University of Nebraska
tzinsmaster@huskers.unl.edu

***Abstract*—We implemented and evaluated the KNN for the white-wine dataset to predict the quality of a white wine, given its physicochemical properties. Exploratory Data Analysis and extensive feature engineering methods are used to produce best features to feed into our model. Moreover, Model selection is used to find the best parameter for a KNN classifier we have engineered. Of particular note is that for s-fold cross-validation, different choices of neighbors, weights, distance and metrics were compared to find the optimal choices for our parameters.**

I. Introduction

KNN is one of the simplest instance based learning system that performs great for small datasets. It is a slow algorithm but has great applications such as Recommender System. It is simple to understand, implement and use for classification.

We are predicting the quality of white-wine based on its physicochemical properties. The target value is converted to binary by labeling quality greater than five as one and zero for the remaining. We start with a brief description of the data set used in the development of the system and the metric we used to measure performance, followed by algorithm and methods we implemented.

II. Data Summary

A. Description Of Data Set

The data set is white-wine quality pulled from UCI repository for Machine Learning. It contains 12 features and 4898 samples.The target variable is quality, renamed as y. Data set contains m(4898) examples, one example (x^i, y^i) per row. In particular, the i-th row contains columns $x^i \in \mathbb{R}^{11}$ and $y^{(i)} \in \{0, 1\}$ The size of training set is $\mathbb{R}^{4898\times12}$.The eleven features are physicochemical tests. These tests value is used to determine the quality of a wine.

B. Descriptive Statistics

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
count	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000
mean	6.854788	0.278241	0.334192	6.391415	0.045772	35.308085	138.360657	0.994027	3.188267	0.489847	10.514267
std	0.843868	0.100795	0.121020	5.072058	0.021848	17.007137	42.498065	0.002991	0.151001	0.114126	1.230621
min	3.800000	0.080000	0.000000	0.600000	0.009000	2.000000	9.000000	0.987110	2.720000	0.220000	8.000000
25%	6.300000	0.210000	0.270000	1.700000	0.036000	23.000000	108.000000	0.991723	3.090000	0.410000	9.500000
50%	6.800000	0.260000	0.320000	5.200000	0.043000	34.000000	134.000000	0.993740	3.180000	0.470000	10.400000
75%	7.300000	0.320000	0.390000	9.900000	0.050000	46.000000	167.000000	0.996100	3.280000	0.550000	11.400000
max	14.200000	1.100000	1.660000	65.800000	0.346000	289.000000	440.000000	1.038980	3.820000	1.080000	14.200000

Fig. 1. Summary of each of the variables in the dataset in terms of mean, standard deviation, and quartiles

C. Feature Selection and Scaling

The data is standardized by subtracting its mean and dividing by standard deviation. Data has zero mean and unit variance after standardization. This will avoid over representation of some of the features. Standardization makes all features to contribute equally to the dissimilarity measures. It always provide better result for classification.

We notice there are redundant features. Wine is acidic, So fixed acidity and ph provides similar information. Also, we could see pair plot to notice that these features have non zero Co-variance. Similarly for Sulphates, Total sulphur dioxide and free sulphur dioxide as all are just sulphates. Moreover, we notice same phenomenon for residual sugar and density. those features are dependent as adding sugar increases density.SO, more residual sugar means more density.

Using intution and pair plot we found some of redundant features. We then used Feature Selection using Backward search to find the optimal features. This is possible since features and data set is small enough. In backward search, we keep removing feature until and unless our accuracy stops increasing in all possible subsets of features. Complexity of backward search is O(n^2). There are 2^n possible KNN models to compare for feature selection. Our optimal features is 'volatile acidity', 'citric acid', 'residual sugar', 'pH', 'sulphates' and 'alcohol'.

III. Methods

A. KNN Algorithm

For any $x_i^{(i)} \in \mathbb{R}^{11}$, we calculate l1 or l2 norm with each $x_i^{(i)} \in (x, y)$, the train set. We store the value in a vector d. Then arg $\min_x$ (d). We find k minimum arguments. We then use a threshold for conditional probability to assign class 'c' to the test point. The Probability is given by fig.2.

$$p(y = c|\mathbf{x}, \mathcal{D}, K) = \frac{1}{K} \sum_{i \in N_K(\mathbf{x}, \mathcal{D})} \mathbb{I}(y_i = c)$$

Fig. 2. Conditional Probability for y

B. KNN Arguments

We use Minkowski distance metric or L_p norm to measure the dissimilarity between two vectors. Mathematically L_p norm is given by,

$$L_p(\vec{x},\vec{z}) := d(\vec{x},\vec{z}) = \left[\sum_{i=1}^{d}|x_i - z_i|^p\right]^{1/p}$$

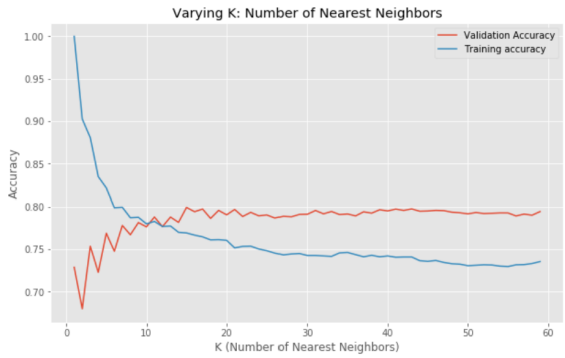
Fig. 3. L_p norm

We use inverse distance for weighted KNN. We compute the inverse of 'd'. We then sum all positive and negative classes of k neighbors and compare them to assign class for the particular test point. Using inverse distance our performace increased significantly. The classes nearer to test point are assigned more weight. Instead having to tweak the threshold for finding class probability that the test point lies, we can decay weight as distance increase to get better result.Inverse distance kernel is given by:

$$k(\vec{x},\vec{x_n}) = \frac{1}{d(\vec{x},\vec{x_n})+\epsilon}$$

C. Model Selection and Complexity

First, we tried to vary number of neighbors denoted by 'k' to find the best KNN model. When we use k equals 1 with l1 norm and uniform weights, our model seems to over fit. This produces great result in trainset but costly for testset as it fails to generalize. While, when k is large underfit and doesnot seem to produce any good result. We find that under this setting optimal value of k is 15.




```
Optimal K:  15
```

Fig. 4. Varing value of Neighbors

Also, we perform same test using all the arguments of KNN and found that optimal value of k is 11.

IV. Results

A. Cross-validation

We randomly split $S_{dataset}$ into S_{train} (say, 80% of the data) and S_test (the remaining 20%). Here, S_{test} is called

```
optimal k:  11
optimal distance:  Euclidean
optimal weights:  distance
optimal value 0.8472299944040291
```

Fig. 5. Optimal Arguments for KNN

the hold-out cross validation set.We then, train each model $M_i \in$ M(set of models) on S_{train} only, to get some score.We then Select the model with greatest score on the hold out cross validation set.

B. S-fold cross-validation

We randomly split $S_{dataset}$ into k disjoint subsets of m/k training examples each. Lets call these subsets $S_1, \ldots, S_k$. For each model M_i, we evaluate it as follows: For j = 1, . . . , k Train the model M_i on $S_1 \cup \cdots S_{k-1} \cup S_k$ to get some model.Test the model on S_j, to get a score. The estimated generalization error of model $M_i$is then calculated as the average of the scores of the model(averaged over j).We then pick the model M_i with the greatest average score, and retrain that model on the entire training set S. It is described visually on fig7 below.

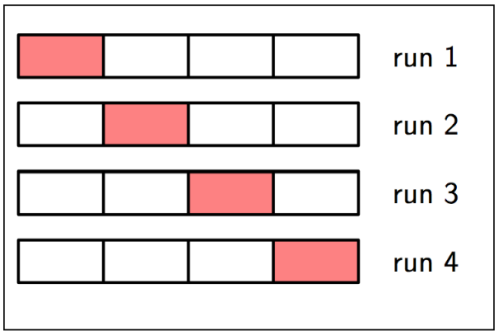


Fig. 6. S-fold cross-validation

C. Accuracy and Generalization error

We compare the true target y with the prediction('p') to find the accuracy. Accuracy simply computes the number of correct prediction divided by total prediction.

$$Accuracy = \frac{\sum_{i=1}^{m} 1\{y^{(i)} = p^{(i)}\}}{\sum_{i=1}^{m} 1}$$

Generalization Error is simply given by,

Generalization Error = 1 - Accuracy

Since the data set might be skewed. Accuracy and Generalization error will give us incorrect presentation of out accuracy. To better understand what were being misclassified, we use confusion matrix.

D. Confusion Matrix

Confusion matrix overcomes the issue of Accuracy. It shows the actual and prediction target for both positive and negative targets. Since we are performing binary classification, confusion matrix can be represented as:

		Predicted class: P	Predicted class: N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Fig. 7. Confusion matrix for Binary Classification

For our model with optimal parameters on Testset we get:

```
Confusion Matrix :

col_0    0    1
row_0
0      185  106
1      141  548
```

Fig. 8. Confusion matrix for wine quality classification

E. Precision, Recall and f1 score

Precision computes the accuracy of positive labels.

$$\text{precision} = \frac{TP}{TP + FP}$$

Fig. 9. Precision

Recall is the ratio of positive instances that are correctly detected by the classifier.

$$\text{recall} = \frac{TP}{TP + FN}$$

Fig. 10. Recall

The F1 score is the harmonic mean of precision and recall.

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{FN + FP}{2}}$$

Fig. 11. F1 score

The precision-recall plot was used to find the optimal threshold to increase f1 score. By observing the curve, 0.6 threshold is optimal for our model.

Another good metric to evaluate our classification and find optimal threshold is ROC curve.

Using the ROC curve we found the area under curve AUC to be 0.794195 where 1 represents perfect classifier and 0.5 represents totally random classifier.

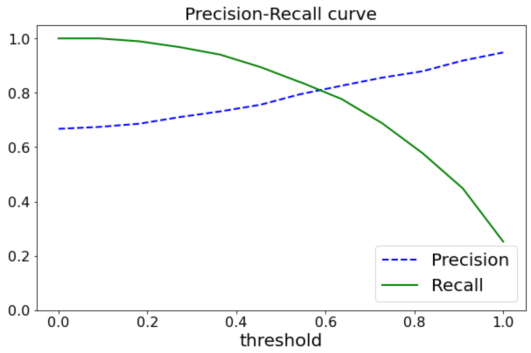


Fig. 12. Precision-Recall curve

F. Figures and Tables

Test performance metrics are roported in the Table I and Table II. Here k represents number of neighbor and p represents norms

ACKNOWLEDGMENT

Acknowledgment is given to the UNL CSCE478 professor, Dr. M. R. Hasan, and the class GA, Atharva Tendle, for their contributions in our learning of the subject matter, and in organizing the assignment and course materials.

REFERENCES

[1] Bibek Karki, Biquan Zhao, Tyler Zinsmaster, GitHub repository for the assignment, https://github.com/CSCE478-ML/Assignment1

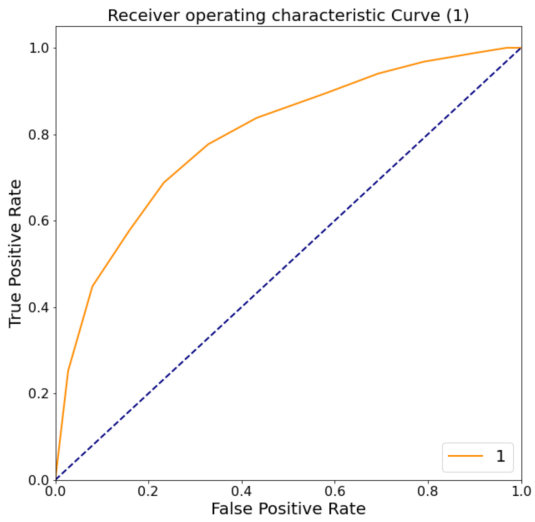


Fig. 13. ROC curve

TABLE I
TEST SET PERFORMANCE

KNN Parameters	**Metrics**	
	Accuracy	***F1 Score***
k=5, p=2, uniform weight	0.7449	0.8117
(Standardized) k=5, p=2, uniform weight	0.7548	0.8156
(Optimal)k=11, p=2, inverse distance weight	0.8296	0.8745
k=11, p=2, uniform weight	0.7673	0.8321

TABLE II
PERFORMANCE WITH DIFFERENT METRICS

KNN	***Precision***	***Recall***	***F1***	***Accuracy***	***Gen Error***
Optimal	0.8595	0.7992	0.8252	0.7592	0.2408

[2] Andrew Ng, "CS229 Lecture notes", https://see.stanford.edu/materials/aimlcs229/cs229-notes5.pdf

[3] Dr. M. R. Hasan slides regarding KNN can be found at $https://canvas.unl.edu/courses/97606/pages/lecture-slides-and-readingguide?module_item_id=2130394$.

[4] Dr. M. R. Hasan github repo on KNN $https://github.com/rhasanbd/K-Nearest-Neighbors-Learning-WithoutLearning$