Milestone 3

Jin Young Kim – Bernoulli Naïve Bayes

Insoo Rhee – K-Nearest Neighbor

David Sun – Random Forest

1. What classifiers did you use for milestone 3? (5 points)

We used random forest, k-nearest neighbor, and Bernoulli Naïve Bayes classifiers.

2. Which features did you select for your classifiers? Please comment on the reason for your feature selection. If you choose to work on the bonus question, you can add your features extracted from external datasets at this step. (5 points)

The features selected are price, latitude, longitude, number of bedrooms, and number of bathrooms. These attributes were selected as they are likely important determinants of a listing's popularity. To improve classifier performance, feature selection may be performed using methods such as Fisher score or PCA before performing the classification to run the classifier on most relevant features only.

3. How did you perform cross-validation? Please describe the procedure. (10 points)

Cross-validation was performed using the *train_test_split* and *cross_val_score* methods from the sklearn library. Data was first split into a training dataset and a test dataset in a 60:40 or 70:30 using *train_test_split*. Cross validation was performed on the training dataset using *cross_val_score*. The method splits the training dataset into k-1 training subsets and 1 validation set and returns the average accuracy across the k folds.

4. What performance did the first version of your classifiers achieve on the validation dataset (in cross-validation) and on the test dataset? Please comment on the performance of the classifier. (15 points: 5 points for performance, and 10 points for comments).

1. K-Nearest Neighbor – The first version performed poorly with accuracy=0.6422 and log loss=5.746 on validation dataset and log loss=5.456 on the test dataset.

Log loss of the first version on the test dataset (K-Nearest Neighbor)

Name	Submitted	Wait time	Execution time	Score
Before_modification_submission.csv	just now	1 seconds	2 seconds	5.45648

Complete

2. Bernoulli Naïve Bayes – The first version performed poorly with accuracy=0.201589 and log loss=1.09832 on the validation dataset and log loss=1.10761 on the test dataset. The model performed more poorly than random guesses in terms of accuracy.

Log loss of the first version on the test dataset (Bernoulli Naïve Bayes)

Name	Submitted	Wait time	Execution time	Score
submission_bayes_first.csv	a few seconds to go	0 seconds	1 seconds	1.10761

Complete

3. Random Forest – The first version performed poorly with accuracy=0.64 on and log loss of 2.29 on validation dataset and 2.05 on the testset, requiring some tweaking to reduce the high log loss.

Log loss of the first version on the test dataset (Random Forest)

5. What actions did you take in order to improve your classifiers? You can modify your dataset or the parameters of your classifier. Please record your modifications in your report. (20 points)

K-Nearest Neighbor

1. N_neighbors (# of nearest neighbors) was increased from 3 to 100.
2. Leaf-size was increased from 40 to 100.
3. The classifier's algorithm was set to KDtree (Different algorithm parameters were set until the one with the best result (KDtree) was found).
4. Data was scaled using StandardScaler.

Following these modifiactions, the models' log loss improved from 5.45648 to 0.72688.

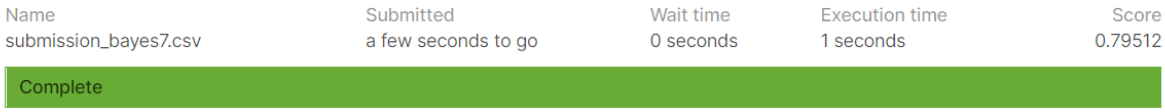
Log loss achieved after modifications (K-Nearest Neighbor)

Bernoulli Naïve Bayes

1. Additive smoothing – additive smoothing was used to smooth the data. Additive smoothing prevents assigning probability of 0 to categorical data not seen in the training dataset.
2. Class prior probabilities – the model was made to learn class prior probabilities instead of assuming the data to have a uniform prior distribution.

Following the two modifications above, the model's log loss improved from 1.10761 to 0.79512.

Log loss achieved after the modifications (Bernoulli Naïve Bayes)

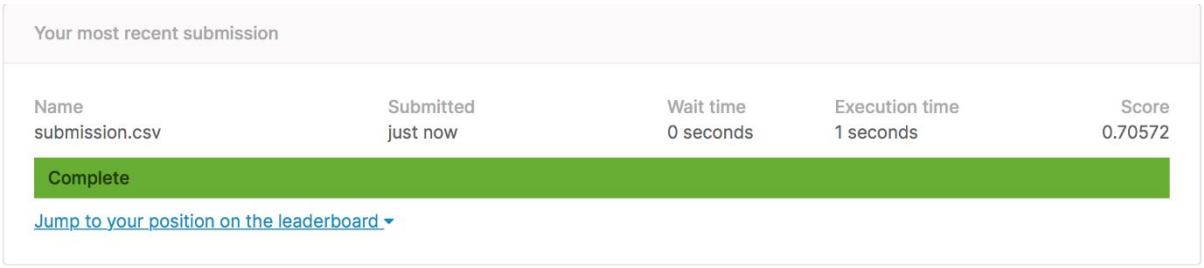


Random Forest

1. Tuning the “max- n_estimators” parameter to 30 - the default value was 10, which is not enough for our relatively large dataset.
2. Tuning the “max_depth” to 20 to reduce overfitting
3. Changing the splitting strategy – ‘entroy’ strategy produced better outcome than ‘gini’.

Following the modifications above, the model’s log loss improved from 2.05832 to 0.70572.

Log loss achieved after the modifications (Random Forest)



6. How did you check whether any overfitting occurred during your training? Did you observe overfitting? What did you do to avoid overfitting? (10 points)

Overfitting was checked by comparing between the training accuracy and test accuracy measured in terms of log loss. If the model achieved a higher accuracy on the training dataset than the test dataset, the model was deemed overfitting.

K-Nearest Neighbor – The initial model did not overfit because the log loss on cross-validation dataset was larger than the log loss on test dataset, 5.746 and 5.456 respectively. The modified model performed significantly better without overfitting (log loss on training dataset = 0.731, and log loss on test dataset = 0.72688). Tuning the parameters such as the number of nearest neighbors considered to predict the class of the unobserved data and the type of algorithm used as well as scaling the dataset helped reduce overfitting.

Bernoulli Naïve Bayes – Slight overfitting was observed for the initial Bayes classifier with log loss=1.09832 on the training dataset and log loss=1.10761 on the test dataset. The model performed slightly better on the training dataset as it produced smaller log loss. On the other hand, no overfitting was observed for the modified classifier. The modified version produced log loss of 0.79953 on the training dataset and 0.79512 on the test dataset. Selecting features relevant to a listing’s popularity and applying addictive correction to avoid assigning 0 probabilities to observations in the test dataset that are not observed in the training dataset helped reduce overfitting.

Random Forest – No overfitting was observed during training since the model performed better on the test dataset (log loss = 2.05) than the training dataset (log loss = 2.29). To avoid overfitting, the maximum depth of the classifier was reduced while the n_estimator parameter was increased. Splitting strategy was also adjusted. These parameters were carefully adjusted until improvements in classifier were observed.

7. What performance did you achieve on the validation dataset (in crossvalidation) and on the test dataset after your modifications? Please, try to explain the gains. (15 points: 5 points for performance, and 10 points for explanation)

Random forest – The modified version achieved log loss of 0.705 log loss, which is a 1.4 reduction in log loss compared to the initial model. Increasing the n_estimator parameters from default 10 to 30, set a limit to the max depth parameter to avoid any potential overfitting, and changing the splitting strategy from the standard 'gini' to 'entropy' all helped improve the model's performance and reduce log loss.

K-Nearest Neighbor – The modified version achieved the log loss = 0.731 on the validation dataset and log loss = 0.72688 on the test dataset. Tuning the model and scaling the dataset improved the model significantly.

Bernoulli Naïve Bayes – The modified version produced log loss of 0.79953 on the validation dataset and 0.79512 on the test dataset. The two modifications, additive smoothing (assigning non-zero probabilities to observations in the test dataset that were not observed during training) and readjusting of probabilities based on prior probabilities, helped improve classifier performance.

8. Evaluate one additional evaluation metrics mentioned in class on the validation dataset. Which metric did you use? What were the results? How do these results compare to the results for multi-class logarithmic loss? (10 points)

F1 score was used as an additional evaluation metrics. F1 scor emeasures the relation between precision and recall. Large F1 scores indicate better classifier performance while large log loss indicates the opposite.

K-Nearest Neighbor – The initial model had a F1 measure of 0.629 and log loss of 5.746, and the modified version gave F1 measure of 0.6031 and log loss of 0.731. Although F1 decreased slightly, log loss was decreased by over 80%; therefore, we can conclude the modification improved the model.

Bernoulli Naïve Bayes – The initial model had a F1 measure of 0.09443 and log loss of 1.09832, indicative of very poor performance. Following the modifications, the model had nearly a six-fold increase in the F1 measure (0.56548) and a 0.5328 reduction in log loss (0.56044). Both the gain in F1 and reduction in log loss are indicative of improvement in the model's performance.

Random Forest – The initial model had a F1 measure of 0.66 and log loss of 2.2. Following the modifications, F1 decreased to 0.56, and log loss decreased to 0.7. Despite the 10% reduction in F1, the 70% increase in log loss indicates that our model improved.

9. Compare your new classifier with the classifiers used in milestone 2. Try to explain the difference between the performance of the classifiers and the gains of the milestone 3 classifiers. (10 points)

<u>K-nearest neighbor vs SVM</u>

The initial k-nearest neighbor has significantly larger log-loss (≈5.30) than SVM (≈0.70) in Milestone 2. Increasing leaf-size from 40 to 100, changing the classifier's algorithm to KDtree, setting the number of nearest neighbors to 100, and data scaling improved the log loss score of K-nearest neighbor to 100 reduced log loss of k-nearest neighbor to 0.72688, which is almost on part with that of SVM. K-nearest neighbor's poorer performance compared to SVM can be explained by its sensitivity to a large feature space and outliers. Our model was trained in a feature space of five features, and the testing dataset of nearly 75,000 rows likely contains several outliers. SVM handles multi-dimensional problems and outliers better as it uses kernel functions to transform non-linear problems to linear ones and uses only the relevant datapoints (support vectors) to find a linear separation.

<u>Logistic regression vs Bernoulli Naïve Bayes</u>

Logistic regression performed better than Bayes on the testing data (log loss: 0.6945 vs 0.79512, respectively). One possible explanation in the performance reduction is that logistic regression generally performs better with Bayes given enough training data [1]. Another limitation of Bayes is that it assumes that all features are conditionally independent of one another. It is likely that the features included in our dataset are correlated (e.g. certain prices associated with certain locations (lat, long)), violating this assumption. Logistic regression handles correlated features better than Bayes as it assigns lower weights to correlated features.

Decision Tree vs Random Forest

Decision Tree from milestone 2 had higher log loss of around 1.24 and slightly lower accuracy of 68% compared to the accuracy of 69.5% of the random forest model on the training dataset. This gain in performance is expected since random forest is an ensemble of decision trees that reduces bias and overfitting in decision trees. Additionally, it is not surprising that multiple trees handle large datasets better than a single decision tree. Tuning of parameters such as the number of trees and splitting strategy further improve the model's performance.

Github URLs

K-nearest neighbor - git@github.com:irhee/Milestone3_cmpt459.git

Naïve Bayes - https://github.com/KatelynKimSFU/NaiveBayes

Decision tree - https://csil-git1.cs.surrey.sfu.ca/damiens/cmpt459.git

References

[1] Ng, 'On Discriminative vs. Generative classifiers: A comparison of logistic regression and naïve Bayes,' Neural Information Processing Systems 2001.