Performance Measures

Question 1:

The Cohen's kappa coefficient is an alternative measurement for accuracy. Do more research on Kappa() (such as from vcd package). How do you calculate kappa? Show the formula, and command. Give an example. What is the common interpretation of kappa statistic value? What is weighted and un-weighted kappa statistics, and when to use them?

The Cohen's kappa coefficient is a statistic that measures the accuracy of classification algorithms adjusted for the possibility of correct predictions obtained by chance alone. It is especially important when working with datasets which contain severely imbalanced observations. For example, for a dataset containing 98 observation of class A and 2 observations of class B a model always predicting the dominant class A will still show an impressive precision rate. However, observations of class B will always be misclassified. The kappa statistic will reward a classifier only if it shows better results than those obtained by chance alone.

The values for the kappa coefficient fall in the interval between 0 and 1, where 0 indicates poor agreement and 1 indicates perfect agreement between the predictions produced by the model, and the true values. There is no universally accepted interpretation of the values between these two extremes. Below is one of the possible scales used to interpret the kappa statistics (Cohen's Kappa, n.d.):

- 0 - agreement equivalent to chance;
- 0.1 – 0.20 – slight agreement;
- 0.21 – 0.40 – fair agreement;

- 0.41 – 0.60 – moderate agreement;
- 0.61 – 0.80 – substantial agreement;
- 0.81 – 0.99 – near perfect agreement;
- 1 – perfect agreement.

While the names for these categories can be subjective and vary between authors, the numeric values allow comparing predictive capabilities of different models. In addition, depending on the application area, some models might be sufficiently precise having "substantial agreement" (e.g., predicting viewer's movie preferences), and in some cases (e.g., medical diagnostic applications), "near perfect agreement" might be required.

The Kappa statistic is calculated by comparing an observed accuracy with the expected accuracy (random chance) using classifier results displayed as a contingency matrix. Kappa measured the percentage of data values in the main diagonal (correct classifications) of the table and then adjusts these values for the amount of agreement that could be expected due to chance alone.

$$Kappa = \frac{Observed\ Accuracy - Expected\ Accuracy}{(1 - Expected\ Accuracy)}$$

$$Kappa = \frac{Pr(a) - Pr(e)}{1 - Pr(e)},$$

where

Pr(a) – proportion of the actual agreement between the classifier and the true values;

Pr(e) – proportion of the expected agreement between the classifier and the true values.

Using spam message classification example (please see, full code and step-by-step comments in MSDS664_W2_Weakly_Natalia_Assignment.r) we received the following results for classifying 1115 text messages into the spam and ham categories:

```
sms.predictions ham spam
           ham  947    9
           spam   9  150
```

In practice, it is convenient to use specialized R commands in order to calculate Kappa statistic, in particular, Kappa() command from the vcd package:

```
> #for calculating kappa statistic
> library(vcd)
> #Using vcd package to calculate Kappa statistic
> Kappa(table1)
            value     ASE     z Pr(>|z|)
Unweighted 0.934 0.01542 60.58        0
Weighted   0.934 0.01542 60.58        0
```

The above output shows that Cohen's Kappa for the naïve Bayes spam classifier used in the example is 0.934. It means that the classifier has near perfect agreement. It provides reliable results that cannot be explained just by chance.

Similar results can be obtained by using confusionMatrix() command from the caret package.

```
> #Confusion Matrix using caret package
> confusionMatrix(table1)
Confusion Matrix and Statistics


sms.predictions ham spam
           ham  947    9
           spam   9  150

               Accuracy : 0.9839
                 95% CI : (0.9746, 0.9904)
    No Information Rate : 0.8574
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.934
 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9906
```

```
           Specificity : 0.9434
        Pos Pred Value : 0.9906
        Neg Pred Value : 0.9434
            Prevalence : 0.8574
        Detection Rate : 0.8493
  Detection Prevalence : 0.8574
     Balanced Accuracy : 0.9670

      'Positive' Class : ham
```

The above output shows that confusionMatrix() command calculates several useful indicators of the model's performance, including Kappa statistic. The result, 0.934, is equivalent to the one previously received using Kappa() function from the vcd package.

Regular (unweighted) Kappa statistic is ideally suited for nominal (non-ordinal) categories. Weighted Kappa can be calculated for tables with ordinal categories as it calculates disagreements differently. Weighted Kappa considers not only diagonal elements in the confusion matrix but off-diagonal cells as well. This method involves three matrices, the matrix of observed scores, the matrix of expected scores based solely on chance, and the weight matrix. The weight matrix represents the seriousness of disagreement between real and predicted results. Thus, diagonal elements in this matrix are equal to zeros (meaning no disagreements). Off-diagonal cells contain non-zero weight, for example, 1,2 etc., depending on the seriousness of disagreement, or distance between the misclassified categories. For example, if we have three ordered categories, misclassifying an observation belonging to class 1 into class 2 will mean distance of 1, and mistakenly classifying the same observation into class 3 will result in distance 2.

The formula for weighted Kappa (Zaiontz, n.d.):

$$\text{Weighted } Kappa \;=\; 1 - \frac{\sum_{i,j}^{k} W_{i,j}\;\; p_{i,j}}{\sum_{i,j}^{k} w_{i,j}\, e_{i,j}}$$

Where k – number of categories, w, p, e – elements in the weight, observed and expected matrices.

In case if non-diagonal cells contain weight=1, then unweighted Kappa and weighted Kappa calculations result in the same values. In the above example with the spam classifier, Kappa() command from the vcd package yielded the same results for both weighted and unweighted Kappa coefficients.

Question 2.

Precision and recall are mainly used in an information retrieval context (e.g. return results from a search engine). What is the command/function in R and in what package to compute these two measures? Give an example. Explain the meaning of precision and recall. Do you prefer low or high values? Interpret the results.

Precision and recall are the two additional indicators allowing to evaluate the performance of predictive models. While they both originated in the information retrieval field (IR) which is still reflected in terminology, they can be effectively used to evaluate the performance of the classification models.

Precision, also known as positive predictive value, reflects the probability that a positive classification is correct. It is calculated as the number of observations correctly labeled as belonging to a positive class divided by the total number of elements in the positive class. In other words, precision is equal to the number of true positive results divided by the sum of true positives and false positives.

If the precision coefficient is equal to 1, then it means that the classifier correctly identified all items as belonging to a particular class. The closer precision gets to 1, the better the model is in properly identifying class elements. In the information retrieval terms, precision indicates how many of the retrieved documents are relevant.

In turn, recall, also known as sensitivity or true positive rate, is the number of observations correctly classified as positive class divided by the actual total number of observations in that class. Recall reflects the probability of correct classifications of elements of the target class. As in the case of precision, the desired level of recall is equal to 1, which means that every element of the target class was correctly identified as belonging to this class.

Mathematically, recall is equal to the number of true positives divided by the sum of true positives and false negatives. In terms of information retrieval, recall represents the fraction of relevant documents that were retrieved.

Caret package in R contains functions allowing to compute both precision and recall. Continuing with the spam classification example above:

```
> #Precision using caret package
> precision(table1) #Precision, positive predictive value
[1] 0.9905858
> #Sensitivity using caret package
> sensitivity(table1) #Recall (Sensitivity, true positive rate)
[1] 0.9905858
```

The above results demonstrate that the Naïve Bayes model used predict spam and ham messages demonstrated really good precision and recall, as both numbers are very close to 1. It means that it was able to correctly detect 99% of all ham messages and in 99% messages labeled as ham were actually ham.

The indicators for precision and recall also included as a part of the output of the confusionMatrix() command in the caret package (rounded to four decimals) :

```
> #Confusion Matrix using caret package
> confusionMatrix(table1)
Confusion Matrix and Statistics


sms.predictions ham spam
           ham  947    9
           spam   9  150

               Accuracy : 0.9839
                 95% CI : (0.9746, 0.9904)
    No Information Rate : 0.8574
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.934
 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9906
            Specificity : 0.9434
         Pos Pred Value : 0.9906
         Neg Pred Value : 0.9434
             Prevalence : 0.8574
         Detection Rate : 0.8493
   Detection Prevalence : 0.8574
      Balanced Accuracy : 0.9670

       'Positive' Class : ham
```

Precision is listed as positive predictive value – 0.9906, and recall is listed as sensitivity – 0.9906.

Question 3.

Include 3 more performance measures of your choice (explain the objective, show an example and interpret the result).

Continuing with the case of classification models, it should be noted that in some situations, it is important to compare algorithms not only based on their true positive value, but their true negative value, or specificity, as well. **Specificity** measures the proportion of negative observations that were correctly classified as negative (e.g., healthy people with negative test results to a virus in a sample). In the case of the spam filter example, if ham is considered a

positive class (as in the example above), specificity reflects the proportion of spam messages correctly classified as ham.

Specificity is equal to the number of true negative classifications divided by the sum of true negatives and false positives. It can be computed using specificity() function form the caret package (sample output shown below):

```
> #Specificity using caret package
> specificity(table1)
[1] 0.9433962
```

In our example specificity of about 0.9434 means that more than 94% of spam messages were correctly identified as spam. This indicator is also included in the output of the confusionMatrix() command.

In our continuing example of the spam filter, the true negative rate (specificity) of about 94% is slightly lower than the true positive rate (sensitivity) of 99%. It means that the model is our example is slightly better at correctly identifying ham messages (considered the positive class in the example) than spam messages (negative class). In case of a spam filter, it might be beneficial that the majority of the genuine messages get correctly identified and delivered even at the expanse of an occasional spam message getting through. In other circumstances, for example, in the antivirus and intrusion detection systems, the opposite might be true.

The **F-measure,** also called F-score, evaluates the performance of a model by combining precision and recall into one indicator. It uses harmonic mean as both precision and recall are expressed as coefficients between 0 and 1 that can be interpreted as rates. The value for F-measure is calculated as two times the product of precision and recall divided by the sum of precision and recall. By simplifying precision and recall we can find that Fa measure is equal to

two times true positives divided by the sum of false negatives, false positives and two times true positives.

F-measure can be easily computed using F_meas() function from the caret package.

```
> #F-measure using caret
> F_meas(table1)
[1] 0.9905858
```

As the output above shows, in the case of the spam classifier it was about 0.9906. F-measure provides a convenient way of comparing different models, but it assumes that both precision and recall are equally important.

In case of both sensitivity (true positive rate) and specificity (true negative rate) are equally important in comparing the performance of different models, it might be convenient to use another indicator that combines the first two – balanced accuracy. **Balanced accuracy** is simply an average of sensitivity and specificity and it is included in the output of the confusionMatrix() command from the caret package.

In our example, (sensitivity + specificity)/2 = (0.9906+0.9434)/2 = 0.9670.

```
      Sensitivity : 0.9906
      Specificity : 0.9434
Balanced Accuracy : 0.9670
```

Since balanced accuracy reflects how well classification worked for each class separately, it might be more informative than the overall accuracy indicator (ratio of correctly classified observations to all observations). In our example, the balanced accuracy of 0.9670 was slightly lower than overall accuracy 0.9839 as is reflected misclassifications for both ham and spam respectively. Balanced accuracy indicator might especially useful when working with datasets that are unevenly distributed between the two classes.

Overall, there are several indicators that can be used to evaluate the performance of predictive models. Simple accuracy might not always be the best, as it does not reflect the relative importance of misclassifications in two classes and doesn't take into consideration class composition of the test dataset, and as a result a possibility of obtaining correct classifications purely by chance. The best approach when comparing predictive models is to consider several indicators depending on what factors are more important for the application.

References

Cohen's Kappa Statistic. (n.d.) Statistics How To. Retrieved from https://www.statisticshowto.datasciencecentral.com/cohens-kappa-statistic/

Zaiontz, C. (n.d.) Weighed Cohen's Kappa. Real Statistics Using Excel. Retrieved from http://www.real-statistics.com/reliability/weighted-cohens-kappa/