CMPS142 Homework 2

Jacob Katzeff
Student ID 1426015

Christopher Hsiao
Student ID 1398305

1. Logistic Regression

$p(1|x,w) = \frac{\exp \mathbf{w}\cdot\mathbf{x}}{1+\exp \mathbf{w}\cdot\mathbf{x}}$ and $p(1|x,w) = 1 - p(0|x,w)$
$\iff p(0|x,w) = 1 - \frac{\exp \mathbf{w}\cdot\mathbf{x}}{1+\exp \mathbf{w}\cdot\mathbf{x}} = \frac{1}{1+\exp \mathbf{w}\cdot\mathbf{x}}$
$\iff \frac{p(1|x,w)}{p(0|x,w)} = \frac{\frac{\exp \mathbf{w}\cdot\mathbf{x}}{1+\exp \mathbf{w}\cdot\mathbf{x}}}{\frac{1}{1+\exp \mathbf{w}\cdot\mathbf{x}}} = \exp \mathbf{w}\cdot\mathbf{x}$
$\iff \ln\frac{p(1|x,w)}{p(0|x,w)} = \ln\exp \mathbf{w}\cdot\mathbf{x} = \mathbf{w}\cdot\mathbf{x}$
Thus, the definitions for $p(0|x,w)$ and $p(1|x,w)$ are equivalent to defining $\mathbf{w}\cdot\mathbf{x}$ as the log-odds, $\ln\frac{p(1|x,w)}{p(0|x,w)}$.

2. Bayesian Probabilities

a. The outcome space is $\{\{B,B\},\{G,B\},\{B,G\},\{G,G\}\}$, where the first index in each set is the younger. The atomic events are the individual sets inside the outcome space: $\{B,B\},\{B,G\},\{G,B\},\{G,G\}$.
b. There are three atomic events in which the neighbor has a girl. In only two of these is the other child a boy. Therefore, the probability is $\frac{2}{3}$.
c. There are two atomic events in which the older child is a girl. In only one of these is the younger a boy. Therefore, the probability is $\frac{1}{2}$.
d. This is the same question as part b. We are given that one child is a girl, and we don't know if they are younger or older. There are 3 atomic events in which a child is a girl, and only two of those events has a boy as the other child. Therefore, the probability is $\frac{2}{3}$.

3. Independence of Random Variables

$\mathrm{E}[VW] = \sum\limits_{i=1}^{\infty}\sum\limits_{j=1}^{\infty} v_i w_j p_{i,j}$. But since V and W are independent, $p_{i,j} = p_i p_j$.
So $\sum\limits_{i=1}^{\infty}\sum\limits_{j=1}^{\infty} v_i w_j p_{i,j} = \sum\limits_{i=1}^{\infty}\sum\limits_{j=1}^{\infty} v_i w_j p_i p_j = \sum\limits_{i=1}^{\infty} v_i p_i \sum\limits_{j=1}^{\infty} w_j p_j = \mathrm{E}[V]\mathrm{E}[W]$.

4. Weka Experiments

a. The accuracies and explanations of running `Nearest Neighbor`, `Naive Bayes` and `Logistic Regression` on the diabetes dataset are as follows:

1. Nearest Neighbor: 100%
   Nearest Neighbor wins out, because there isn't actually a model it is attempting to generalize. Since

we are using the training set as the test set, it simply finds the example with the set of features it sees, for every set of features.

2. Logistic Regression: 78.2552%
   Logistic Regression attempts to generalize by finding a point of max confusion and seeing where the examples fall around the sigmoid function. Such an approach could easily lead to false positives/negatives, as just because certain people may have feature values correlated to diabetes doesn't mean they are guaranteed to have the disease.

3. Naive Bayes: 76.3021%
   Naive Bayes assumes that parameters are independent of each other (hence, the "naive"). Since the features collected are largely biometric data, it is fairly likely that there is some relationship between the features, and that values for certain values of an example may have an impact on values of other features of the same example.

b. The linear function $\sum_i w_i \cdot x_i + bias$ has the weight vector:

$$w = [-0.1232, -0.0352, 0.0133, -0.0006, 0.0012, -0.0897, -0.9452, -0.0149]^T$$

c. After running the experiments in part $(a.)$ with 10-fold Cross Validation for our test option, the accuracies and explanations are as follows.

1. Logistic Regression: 77.2135%
   I think the slight decrease in accuracy from part $(4a)$ is because in 10-fold CV, each iteration of the model sees $\frac{1}{10}^{th}$ the data, thus making the model less accurate (theoretically). As a result, the average of 10 very slightly worse models yields a slightly worse model overall.

2. Naive Bayes: 76.3021%
   Note that the accuracy of Naive Bayes did not change at all with respect to its accuracy in part $(4a)$. This is because Naive Bayes locates (is calculates a better word) the Probability Distribution of each feature of each example to the outcome stays consistent, regardless of which order you see the examples in. Since the model inevitably sees every example, the probability distribution assigned to each feature in the model is the same as in part $(4a)$.

3. Nearest Neighbor: 70.1823%
   Clearly, Nearest Neighbor suffered the most, with an accuracy loss of 29.8177%. This makes perfect sense, since in 10-fold CV, part of the examples are omitted. Thus, a fairly often number of times, the model can't locate the *best* example to associate the seen features with.

d. After normalizing the dataset, all of the feature values fell between 0 and 1.

The accuracy of the three algorithms stayed exactly the same. This is because the value of the features of the examples are distributed the same way in the normalized range $[0, 1]$ compared to their distributions in each features respective original range.

There is a very significant change in the weight values for all features. This is because when the dataset was normalized, their values were all forced within the range $[0, 1]$. As a result, there is a significant reduction in the disparity between feature values. Thus, this disparity does not need to be represented in the weights, and the weights can thus more closely represent the relationship between features, and their influence over the prediction.

e. Below details the impact each ridge parameter had on the CV Accuracies and Weights of models.

- 0: Accuracy and Weights stayed the same as in the $(4c)$ model - 77.2135%.
- 1: While Accuracy stayed the same, the weights on each feature for the most part were just slightly smaller across the board.
- 10: While Accuracy stayed the same, the weights on each feature were dramatically smaller this time around.
- 100: Now, not only is the trend of the weights becoming much smaller, but the accuracy is lowered substantially to 74.6094%
- 1000: Now, not only are the weights significantly reduced (not a single weight value ¿ 1), but the accuracy has tanked to 65.1042%.

Clearly, when ridge parameter values get very large, significant accuracy in the model is lost. This is because we severely penalize the weights, to the point where the weights are unable to adequately describe the relationship of the features, and thus are unable to generalize to new examples.

f. 3NN and 5NN have accuracies 72.6563% and 73.1771% respectively, which is a decent improvement over 1NN, which has accuracy 70.1823%. The highest accuracy is found at 7NN ($k = 7$), which boasts an accuracy of 74.7396%. 8NN ($k = 8$) is where the accuracy begins to decrease.

g. We would expect the accuracy to decrease for 1NN and Naive Bayes, because their accuracies can be affected by noise or unnecessary data. 1NN calculations would be thrown off because the ”nearest neighbor” isn’t actually the nearest, especially since the dimensionality is so drastically increased.

Naive Bayes would significantly over-represent the repeated feature, since it doesn’t consider features in conjunction with one another, and treats them as independent of other features.

Logistic Regression, we think, would assign every instance of the repeated feature the same weight. Perhaps this means that the influence over the outcome is the same as if there were only once instance of the feature?

Here are the accuracies of each algorithm over the modified dataset.

- 1NN: 66.1458%
- NB: 68.8802%
- LG: 77.2135%

As we had predicted, the accuracies for 1NN and NB suffered rather dramatically, while accuracy for LG stayed the same.

h. Here are the accuracies of each algorithm over the modified dataset.

- 1NN: 60.8073%
- NB: 74.349%

- LG: 75.651%

We observe that 1NN decreased pretty substantially, even compared to its accuracy in part $(4g)$. Decrease in accuracy is definitely expected, because not only do we inject noise, we also greatly increase the dimensionality of the problem (even more so than in part $4g$).

Naive Bayes does slightly worse, compared to its performance without the added noise and dimensionality of 76.3021%. This is due to the fact that the value of these 20 extra attributes is Normally distributed between $(0, 1)$. Thus, it is reasonable to expect that for every extra feature value, it's complement (or something relatively close to it) shows up later to balance things out, and render the attribute insignificant. Since it's unreasonable for this to happen to every single extra weight as this is a finite dataset, we do suffer a relatively small penalty of $\approx 2\%$.

Logistic Regression also suffers slightly, compared to its performance without the added attributes of 77.2135%. Our reasoning is that, similar to the way feature values mostly "balance out" to insignificance in NB, the same occurs in LR. In this case, it implies that for every penalty made to the weight of a feature, it's reasonable to expect that a value for that feature in a different example will reward the weight of the feature by the same amount. Again, since this is a finite dataset, it's unreasonable to expect this to occur often enough to yield actually nullify the negative impact of the extra features.