CS 6923 Machine Learning
Spring 2019
Final Project Report

Name: Haris Khan

NetID: hrk269

PART I: Preprocessing (No more than two pages for this part)

1. How does your program handle missing value? And why?
    - There are 48 attributes in the dataset; 3 which are not useful to us are id_num, player_id, and quidditch_league_player. Id_num and player_id are not useful attributes for supervised learning because all values associated with this feature will be distinct. quidditch_league_player is the label and cannot be used. Therefore, there are only 45 features that could be used for supervised learning.
    - Of the usable 45 features, the data entries only show missing data for the same 4 features: house (a categorical feature), weight (a categorical feature of quantitative data), player code (a categorical feature), and move specialty (a categorical feature with many possible values).
    - Since these are all categorical features, data that is missing features will be given their own "missing" category. For the sake of ease, we will leave the values as "?" and interpret the "?" as representing the "missing" category.
        - This is the ideal method for dealing with the missing values because interpolating is not possible, and removing data entries with missing data will lead to inaccuracies in the data which will reflect poorly on the supervised learning models and their performance accuracies.
2. If your program converts numeric features to categorical features, or categorical features to numeric features. Describe how it does it.
    - In the model one – the K Nearest Neighbors model, categorical features are converted to numeric features using encoding. These features were assigned a numerical value to represent the labels for their categorical feature
    - In model two – the Support Vector Machine/Support Vector Classifier, categorical features would need to be converted to numerical features using hot one encoding. However, this was not implemented because there are enough quantitative features to use for the model and converting categorical features for use in this model would incur significant additional dimensions and greatly increase runtime
    - In model three – the Neural Network/Multi-Layer Perceptron, hot one encoding was initially performed on the categorical variables. However, it was found that the neural network had a much lower accuracy when taking these categorical variables into

consideration in the input layer. Therefore, only quantitative features were selected for this model.

3. Describe any feature selection, combination or creation, and any feature values combination performed by your program and the reasons for doing so.
    - All feature selection and combinations are done after the program has read the csv dataset using the Python Pandas library
    - Features are selected by columns using the Pandas library according to their usefulness/need in the model that is being run.
    - For model one – the K-Nearest Neighbor model, a combination of quantitative and categorical variables was selected. Only categorical features whose values in the dataset had greater than 2 labels were selected in the combination.
        - Features that had only 1 or 2 different labels in the dataset were not considered because they would only create additional noise
    - For model two – the SVM/SVC, only quantitative features were selected. Hot-one encoded categorical features were not combined because there are enough quantitative features to use for the model and converting categorical features for use in this model would incur significant additional dimensions and greatly increase runtime
    - For model three – the Neural Network/ the neural network, hot one encoding was initially performed on the categorical variables. However, it was found that the neural network had a much lower accuracy when taking these categorical variables into consideration in the input layer. Therefore, only quantitative features were selected for this model.
4. Describe other preprocessing used in your program (e.g. centralizing, normalization)
    - Numerical data is normalized in model three – the Neural Network, so that data could converge more quickly

PART II: Classification (No more than two pages for each model in this part)

Model One:

1. Supervised learning method used in this model is:

   K – Nearest Neighbors

2. Why you choose this supervised learning method?
   - There are many features in this dataset and there may be a pattern to the how combinations of some feature values cluster around the label
   - KNN's decision boundary can take on any form and is not restricted to linear, polynomial, or radially separable decision boundaries
   - The data is a form of sports statistics and it is possible athletic features are common with players who go on to become professional quidditch players
3. Describe the method you used to evaluate this method.
   - The training data was divided into training and test data – 60% of the data entries would serve as the training data, while 40% would serve as the test data.
     - The data were stratified by the quidditch_league_player labels to ensure the model is properly trained with 'Yes' and 'No' examples
   - Categorical features were encoded so that they may be used in the calculation of distance from the nearest neighbor
     - Encoding is acceptable here because these features act as labels
   - The model's accuracy and True Positive Rate, taken from the confusion matrix, were used to evaluate this model
     - True Positive Rate is a good evaluator because the data labels are heavily skewed in favor of 'No'.
4. Describe process of experimenting different parameter settings or associated techniques.
   - Parameter name: K – number of nearest neighbors
     - Parameter values: K = {5, 10, 25, 50, 100, 500}
     - Performance of different values:
       - Note: these values are found with uniform weights

k	Accuracy, [%]
5	75.58083375699034
10	87.35383833248602
25	88.73029994916116
50	93.92857142857143
100	96.00152516522623
250	96.90010167768175
500	Memory Error

     - Analysis:

        - There seems to be overfitting for k ≥ 100; therefore, we can tune this parameter to have an optimal value of k = 100
        - We see that as k gets larger, accuracy increases for the model
- Parameter name: weights - weight function used in prediction
    - Parameter values: Weights = {Uniform, Distance}
        - Note: Results from varying k values in previous parameter analysis were performed using uniform weights
    - Performance of different values:
        - Weights = 'Distance'
            - Weigh points by the inverse of their distance
            - Closer neighbors of a query point will have a greater influence than neighbors which are further away

k	Accuracy, [%]
5	75.72064056939502
10	80.32536858159634
25	88.20665988815455
50	93.09481443823081
100	95.55287239450941

    - Analysis:
        - Accuracy increases as K increases, but accuracy when weights are uniform are consistently higher
        - It seems the most suitable parameters are uniform weights with k=100 neighbors

5. Accuracy and Confusion matrix with most suitable parameters

		Predicted	
		Yes	No
Correct	Yes	100	2108
	No	1351	75121

Accuracy: 95.60371123538384%

True Positive Rate: 4.53%

Model Two:

1. Supervised learning method used in this model is:

   SVM (Support Vector Machine) – SCV (Support Vector Classifier)

2. Why you choose this supervised learning method?
   - This method defines a definitive decision boundary within the data that is used to classify and label
   - The kernel trick – the decision boundary does not need to be a straight line, and this model can encompass much more complex relationships between datapoints without complex transformations
   - An SVM can handle a very large number of features well
3. Describe the method you used to evaluate this method.
   - The training data was divided into training and test data – 60% of the data entries would serve as the training data, while 40% would serve as the test data.
     - The data were stratified by the quidditch_league_player labels to ensure the model is properly trained with 'Yes' and 'No' examples
   - Quantitative features were selected; hot one encoding could have been performed on the many numerous provided categorical features but was not done to avoid incurring significant additional dimensions
   - The model's accuracy and True Positive Rate, taken from the confusion matrix, were used to evaluate this model
     - True Positive Rate is a good evaluator because the data labels are heavily skewed in favor of 'No'.
4. Describe process of experimenting different parameter settings or associated techniques.
   - Parameter name: kernel – the type of decision boundary
     - Parameter values: kernel = {linear, rbf, sigmoid}
     - Performance of different values:
       - Without changing other model parameters:

Kernel	Accuracy, [%]
Linear	94.37086934417896
rbf	87.41992882562278

sigmoid	97.19369598373157

- Analysis:
  - The Sigmoid kernel provides the highest accuracy

- Parameter name: gamma – the Kernel coefficient
  - Parameter values: gamma = {10, 100}
  - Performance of different values:
    - Note: gamma parameter was auto set in kernel analysis to a value of 1 / n_features
    - This parameter is only applicable to kernel = {rbf, sigmoid}

Kernel	gamma	Accuracy, [%]
Rbf	10	97.13396034570411
	100	97.144128113879
Sigmoid	10	97.19369598373157
	100	97.19369598373157

  - Analysis:
    - The Sigmoid kernel consistently performs better regardless of the value that gamma has been given
      - Although it provides the best accuracy, this kernel fails to predict any example with a 'Yes' label. Therefore, its True Positive Rate is extremely poor.
    - The rbf kernel with gamma 10 also has a very low True Positive Rate at 0.045%
    - The model's most suitable parameters are with a linear kernel

5. Accuracy and Confusion matrix with most suitable parameters

		Predicted	
		Yes	No
Correct	Yes	224	1984
	No	2107	74365

Accuracy: 94.80045754956788%
True Positive Rate: 10.15%

Model Three:

1. Supervised learning method used in this model is:

   Neural Network (Multi-Layer Perceptron)

2. Why you choose this supervised learning method?
   - Neural networks can handle large amounts of data and "learn" how to interpret it
   - A perceptron can detect complex nonlinear relationships which is good for pattern recognition
   - Starting constants can be randomly initialized and corrected through back propagation
3. Describe the method you used to evaluate this method.
   - The training data was divided into training and test data – 60% of the data entries would serve as the training data, while 40% would serve as the test data.
     - The data were stratified by the quidditch_league_player labels to ensure the model is properly trained with 'Yes' and 'No' examples
   - Initially, hot one encoding was performed on the numerous categorical features used in model one, but it was found that using these features made the model too complex and negatively impacted the accuracy
   - Therefore, only quantitative features were selected for this model
4. Describe process of experimenting different parameter settings or associated techniques.
   - Parameter name: hidden layer size
     - Parameter values: hidden layer size = {(10,10,10), (5,12,5), (12), (2), (7,3)}
     - Performance of different values:
       - Note: this parameter was tested with the following model parameters:
         - Activation = 'logistic' (this is the sigmoid activation function)
         - Max_iter = 3000
         - Learning rate init = 0.001 (Default)
         - Alpha = 0.0001 (lambda for regularization)

Hidden Layer Size	Accuracy, [%]
(10,10,10)	88.13167259786478
(5,10,5)	88.09989832231825
(12)	88.37569903406202
(7)	88.64514489069649
(2)	87.31062531774275
(7,3)	85.84392475851551
(5,5)	87.02211489578038

     - Analysis:
       - A single hidden layer of size (7) provided the highest accuracy rather than multiple hidden layers of varying sizes
       - A layer with a length which is average length of the input and output layer seems appropriate
   - Parameter name: Activation
     - Parameter values: Activation = {sigmoid, 'tanh', 'relu'}
     - Performance of different values:
       - Note: A single hidden layer of size 7 is used

Activation	Accuracy, [%]
Logistic (Sigmoid)	88.64514489069649
Tanh	88.68708693441789
Relu	89.05439755973565

  - Analysis:
    - The Relu activation function yields a higher accuracy than the other activation functions

- Parameter name: solver - solver for weight optimization
  - Parameter values: Solver = {'Adam', 'lbfgs', 'sgd'}
    - 'Adam' - stochastic gradient-based optimizer
    - 'lbfgs' - optimizer in the family of quasi-Newton methods
    - 'sgd' - stochastic gradient descent
  - Performance of different values:
    - Note: this parameter was tested with the following model parameters:
      - A single hidden layer of size 7
      - Relu Activation function

Solver	Accuracy, [%]
Adam	89.05439755973565
Lbfgs	89.50432130147432
Sgd	90.82994407727504

  - Analysis:
    - Sgd – stochastic gradient descent has the highest accuracy

- Parameter name: learning_rate_init (alpha)
  - Parameter values: learning_rate_init = {0.001, 0.25, 0.5, 0.1, 0.05}
  - Performance of different values:
    - Note: this parameter was tested with the following model parameters:
      - A single hidden layer of size 7
      - Relu Activation function
      - sgd stochastic gradient descent for weight optimization

learning_rate_init (alpha)	Accuracy, [%]
0.001	90.82994407727504
0.25	90.34697508896798
0.5	90.52745297407219
0.1	89.35688866293849
0.05	89.27935943060498

  - Analysis:
    - learning_rate_init value of 0.001 has highest accuracy

- Parameter name: alpha (lambda) - L2 penalty (regularization term) parameter
  - Parameter values: alpha = {0, 0.0001, 0.01, 0.1, 0.25}
  - Performance of different values:
    - Note: this parameter was tested with the following model parameters:
      - A single hidden layer of size 7
      - Relu Activation function
      - sgd stochastic gradient descent for weight optimization

- learning_rate_init = 0.001

alpha (lambda)	Accuracy, [%]
0	87.27503812913065
0.0001	90.82994407727504
0.01	87.64234875444839
0.1	87.71733604473818
0.25	87.90671072699543

- Analysis:
  - Model performs best with lambda = 0.0001
  - The highest accuracy was yielded with A single hidden layer of size 7, Relu activation function, learning_rate_init = 0.001, sgd solver, and alpha = 0.0001
    - These will be used as the most suitable parameters

5. Accuracy and Confusion matrix with most suitable parameters

		Predicted	
		Yes	No
Correct	Yes	329	1879
	No	4183	72289

Accuracy: _____92.29537366548043%_____
True Positive Rate: _____14.9%_____

PART III: Best Hypothesis (No more than two pages for this part)

1. Which model do you choose as final method?

   Model number: ___three_____

   Supervised learning method used in this model: _Neural Network (Multi-Layer Perceptron)

2. Reasons for choosing this model.
   - All three models showed a high level of accuracy
   - However, the data is heavily skewed towards one label as opposed to the other - in other words, there are significantly more students who do not become professional quidditch players than there are those who do become professionals
   - Therefore, the True Positive Rate of each model is a better evaluation metric for the models than the model accuracy
     - If accuracy was to be used as the sole evaluation metric, the Zero-R classifier would be the best performing model yielding an accuracy of 97.194%
   - Since the Neural Network/Multi-Layer Perceptron yielded the highest true positive rate and all models had a high true negative rate, it is chosen as the final method for evaluating the testing dataset
3. What are the reasons do you think that make it has the best performance?
   - A Neural Network/Multi-Layer Perceptron can determine complex numerical relationships within the data and does not need to define a definitive boundary for classification

    - Although model one – the K Nearest Neighbors model also does not require a decision boundary, it does not implement an iterative weight optimization algorithm to optimize its classifications capability
- Back propagation ensures that regardless of our initialization, the model will balance itself out to yield the most precise outputs.
    - Back propagation also allows for the weights and biases used in the model to be optimized to best fit the training data
        - This process explores complex numerical relationships between different features in the dataset which allows the Multi-Layer Perceptron to consider each of the features in the input layer separately and output a label after analyzing all features individually
    - This is probably why the model classifies more students who become professional quidditch players correctly