CS 6923 Machine Learning
Spring 2019
Final Project Report

Name: Satish Agrawal
NetID: sa5183

Name: Aral Hekimoglu
NetID: ahh335

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

1. **How does your program handle missing value? And why?**

- *Replace all missing values marked with '?' by null using numpy.Nan in given dataset to make the dataset more easily transformable.*
- *Drop columns having about 40% or more missing values as imputing or dropping those many rows would create a bias and affect the classification model's accuracy.*
- *Replace the missing values in feature "house" by setting them to "Other". We could've imputed them by using mode function but since we had the "Other" as an option it made more sense to use this.*
- *Impute meaningless values such as "Unknown/Invalid". Since, there were only 3 such values we can use Mode Imputation to replace these values. These values are replaced by "Female" which is the mode of this feature.*
- *Drop features which represents the Id-related attribute of the player. As these features are mostly unique they wouldn't affect the classification.*

2. **If your program converts numeric features to categorical features, or categorical features to numeric features. Describe how it does it.**

- *Replace labels "NO" with 0 and "YES" with 1.*
- *Convert "gender" feature which is a categorical feature to numeric feature by binary encoding the feature.*
- *Convert features having more than two categorical values. Since, we cannot use binary encoding we will use One-hot encoding to convert these features into numeric.*
  To achieve One-hot encoding we will use pandas' get_dummies() method to generate features from each of the values in the feature and then drop the initial feature. Features which were One-hot encoded : house and player_type.
- *Re-assign values in snitchnip and stooging features by their weights.*
  'None' can be 0, 'Norm' can be 1 and anything higher can be assigned 2.
- *For simpler classification model, re-assign 23 tactical features using binary encoding.*
  If the player is not using the tactic it can be encoded as 0 and 1 otherwise. Hence 'No' will be 0 and 'Steady', 'Up' and 'Down' will be 1.

```
attbs = ['body_blow','checking','dopplebeater_defence','no_hands_tackle','sloth_grip_roll',
       'twirl','spiral_dive','wronski_feint','zig-zag','porskoff_ploy','transylvanian_tackle' ,'woollongong_shimmy',
       'power_play','starfish_and_stick','bludger_backbeat','hawkshead_attacking_formation','chelmondiston_charge',
       'dionysus_dive','double_eight_loop','finbourgh_flick','reverse_pass','parkins_pincer','plumpton_pass']

data[attbs] = data[attbs].replace({'No': 0, 'Steady': 1, 'Up': 1, 'Down': 1})
```

- *Binary encode 'change' feature by re-assigning 'No' to 0 and 'Ch' to 1 and 'snitch_caught' feature by re-assigning 'No' to 0 and 'Yes' to 1.*

3. **Describe any feature selection, combination or creation, and any feature values combination performed by your program and the reasons for doing so.**

- *Drop features having more than 99% 'No' as they wouldn't affect the prediction of the classifier.*

- *Drop the features which have same value for all records and hence wouldn't provide any valuable information to improve the classifier.*

- *Combining features 'num_games_notpartof', 'num_games_injured' and 'num_games_satout' to create another feature 'num_games_notperformed' which represents the count of all the games that the player didn't contribute.*

- *Reduce 23 tactical features by creating two new features. First feature 'num_tac_changes' which represents the count of the tactical changes made by the player and second 'num_tac_used' which represents the count of all the tactics used by the player.*

- *Reduce feature by using the correlation matrix and dropping features with more than 80% correlation with another feature*

4. **Describe other preprocessing used in your program(e.g. centralizing, normalization)**

- *Log transform numerical features(mentioned below) which have high skewness and kurtosis by using numpy's log1p method. We are using log1p transformation as features have small values. Since features with high skew and kurtosis can impact the standardization.*

```
num_col = ['age', 'game_duration', 'num_game_moves', 'num_game_losses', 'num_practice_sessions', 'num_games_won',\
           'num_games_notpartof', 'num_games_injured', 'num_games_satout', 'num_games_notperformed', 'num_tac_changes',\
           'num_tac_used']
```

- Standardize numeric columns by using numpy's mean() method for finding mean and std() method for finding the standard deviation. This is done since we are using models like Logistic Regression which assume the data to be Gaussian distributed.

- Remove outliers which are 3 standard deviations away from either side of the mean for each numeric feature to help improve the prediction capability of the classification models.

- We have used over-sampling or under-sampling techniques such as SMOTE to convert imbalanced dataset into balanced dataset. In case of our imbalanced dataset having more than 90% values of 'No' class in the label feature, the model will be biased to predict 'No' for all the rows in the test set. This would mean that none of the players will become a league player which defeats our purpose. Hence, we want to make our model as unbiased as possible which is why we use over-sampling or under-sampling.

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**

   Logistic Regression

2. **Why you choose this supervised learning method?**

- *Simple and fast, so experimenting with it is simple.*
- *Good for creating a baseline and further improvements can be observed using this algorithm as the minimum performance.*
- *Few hyperparameters to control*, so easy to regularize effectively and get the best result from this algorithm.

3. **Describe the method you used to evaluate this method.**

- Cross validation
- Keeping out 0.2 of the data as test set
- Using rest in sklearn's RandomizedSearchCV to test
- Use f1 scoring to evaluate hyperparameters.

4. **Describe process of experimenting different parameter settings or associated techniques.**

- Parameter name: C, inverse of regularization parameter.
  - Parameter values:
  - Performance of different values:
    - *As C gets smaller training f1 score decreases but validation f1 score increases.*
    - *As C gets larger, f1 score in training set increases but validation f1 decreases.*
  - Analysis:
    - *As C gets smaller regularization effect increases so chances of overfitting decreases, in trade of getting higher training performance.*
    - *As C gets larger, less regularization effect is used so high training performance is achieved, losing the generalization ability of the model*

5. **Accuracy and Confusion matrix with most suitable parameters**

		Predicted	
		YES	NO
Correct	YES	1245	1016
	NO	5624	12268

Accuracy: _____0.671_____

F1 : _____0.273_____

Model Two:

1. **Supervised learning method used in this model is**

   MLP Classifier

2. **Why you choose this supervised learning method?**

- *Neural networks are powerful to capture nonlinear information and complex relationships between features.*
- *There are a lot of data, so neural networks can learn without overfitting, so generalizing to unseen data is easier.*
- *At the simplest, MLP with no hidden neuron is same as logistic regression model.*

3. **Describe the method you used to evaluate this method.**

- Cross validation
- Keeping out 0.2 of the data as test set
- Using rest in sklearn's RandomizedSearchCV to test
- Use f1 scoring to evaluate hyperparameters.

4. **Describe process of experimenting different parameter settings or associated techniques.**

- Parameter name: Hidden layer size
  - Parameter values: (3,3),(5,5),(7,7),(9,9),(13,13),(20,20)
  - Performance of different values:
    - *Best is 7,7.*
    - *Hidden layers smaller than (7,7) gives better training performance but validation performance is not good.*
    - *Hidden layers bigger than (7,7) gives a close validation and training performance but both is not good.*
  - Analysis:
    - *Due to the power of neural networks, MLP classifier is easy to overfit. Even a 9,9 hidden layer results in overfitting.*

- Parameter name: Alpha
  - Parameter values: 1e-4,1e-3,1e-2,1e-1,1,10
  - Performance of different values:
    - *Increasing the alpha to 1 yields in faster convergence, however alpha of 10 results in oscillations in training performance and doesn't converge.*
  - Analysis:
    - *Using a low alpha slows down the learning process, however using a large alpha may result in no learning at all, therefore some value in between is the ideal value for that hyperparameter.*

5. Accuracy and Confusion matrix with most suitable parameters

		Predicted	
		YES	NO
Correct	YES	1286	975
	NO	5809	12083

Accuracy: _____0.663_____

F1 : _____ 0.275_____

Model Three:

1. **Supervised learning method used in this model is**

   Random Forests

2. **Why you choose this supervised learning method?**

- *Decision trees are useful when there are a lot of categorical variables and one hot encodings.*
- *Random forests are ensembles of decision trees and yields in better performance if tuned properly.*
- *Hyperparameters are intuitive and easy to optimize.*
- *Fast learning and fast prediction.*

3. **Describe the method you used to evaluate this method.**

- Cross validation
- Keeping out 0.2 of the data as test set
- Using rest in sklearn's RandomizedSearchCV to test
- Use f1 scoring to evaluate hyperparameters.

4. **Describe process of experimenting different parameter settings or associated techniques.**

- Parameter name: Maximum depth
  - Parameter values: 1,3,5,…,39
  - Performance of different values:
    - *Small values results in poor performance in both training and validation*
    - *As value increases training performance also increases but validation performance does not increase as quickly.*
    - *At large values train performance is almost perfect but validation performance is not at the same level.*
  - Analysis:
    - *At small values like 1 and 3, model cannot distinguish a lot of features therefore has a poor performance.*
    - *At high values it can fit training data perfectly but not able to generalize because validation performance is low.*
    - *Best value is somewhere in between with maximum depth of 23.*

- Parameter name: Number of estimators
  - Parameter values: 200,400,600,800
  - Performance of different values:
    - *As number of estimator trees increases, performance of both cases increases.*
  - Analysis:
    - *As number of estimator increases, training and testing time complexity also increases, so 400 is the suitable option for our case.*

- Parameter name: Criterion
    - Parameter values: gini,entropy
    - Performance of different values:
        - *Both have similar performance*
    - Analysis:
        - *Choice of decision tree criterion does not affect the performance for this model*

5. Accuracy and Confusion matrix with most suitable parameters

		Predicted	
		YES	NO
Correct	YES	1354	907
	NO	6157	11735

Accuracy: _____0.649_____

F1 : _____ 0.277_____

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

1. **Which model do you choose as final method?**

   Model number: 3

   Supervised learning method used in this model: Random Forests

2. **Reasons for choosing this model.**

- *Has the highest f1 score.*
- *Because there are a lot of categorical features, Random forests seems like a good option*
- *We also looked at the training and validation f1 score curves, for different number of samples. As number of samples increases training f1 dropped validation f1 increased. This is the learning curve. In that learning curve random forest model has the best looking curve. For logistic regression, training and validation curves did not have any gap, meaning that model was underfitting for the training set. For neural networks, gap was wide, meaning that model was overfitting. For random forests, gap was somewhere in between, there was some gap but not as much as the neural networks, which is close to the desired gap.*

3. **What are the reasons do you think that make it has the best performance?**

● *Logistic regression is not as powerful as random forests especially in categorical data like this with bunch of one hot encoded variables.*

● Neural networks are quite powerful and even hidden layers are chosen to be small, neural network model overfit and resulted in poor validation performance.

● Random forests are both fast and effective for data with a lot of one hot encoded variables. Since feature engineering was applied and most of the data given to the model had a meaning on its own, using comparisons on those features were effective enough to have a good model and that is what random forests do in essence.