COMP4211 Report

Name: TANG Wai Tin
Student ID: 20421717

Environment

Using python 3.7.4 and jupyter notebook (in Anaconda environment) to implement all tasks.
Please note that README.md have the information on how to run my codes.

Task 1: Calculate the Win Rate

As records in battles.csv do not contain same pokemons battle together, the win rate can be simply defined as (count of win column) / (count of first column + count of second column). The whole procedure is done at winrate.ipynb

Q1: When calculating the win rates of the Pokemons, you may notice that some of them have not participated in any battle. Explain how you deal with them.

When they are not participated in any battle, denominator, i.e. (count of first column + count of second column) of the win rate is 0, then, it would be NaN value after calculation.

I just directly stored the NaN values into the resulting csv file. When it comes to the case that using the win rate (e.g. linear regression on task 2), I will choose to drop the rows with NaN win rate as it should not be counted towards the regression procedure.

Here is the resulting df after win rate calculation. The new column is appended to the rightmost of the dataframe, then export and save to the csv.

```
In [9]: pkdf
```

Out[9]:

#	Name	Type 1	Type 2	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Has Gender	Legendary	WinRate
1	Bulbasaur	Grass	Poison	45	49	49	65	65	45	1	True	False	0.254902
2	Ivysaur	Grass	Poison	60	62	63	80	80	60	1	True	False	0.395833
3	Venusaur	Grass	Poison	80	82	83	100	100	80	1	True	False	0.631068
4	Mega Venusaur	Grass	Poison	80	100	123	122	120	80	1	True	False	0.578947
5	Charmander	Fire	NaN	39	52	43	60	50	65	1	True	False	0.445652
...	...	...	...	...	...	...	...	...	...	...	...	...	...
796	Diancie	Rock	Fairy	50	100	150	100	150	50	6	False	True	0.386364
797	Mega Diancie	Rock	Fairy	50	160	110	160	110	110	6	False	True	0.911765
798	Hoopa Confined	Psychic	Ghost	80	110	60	150	130	70	6	False	True	0.528090
799	Hoopa Unbound	Psychic	Dark	80	160	60	170	130	80	6	False	True	0.617886
800	Volcanion	Fire	Water	80	110	120	130	90	70	6	False	True	0.602151

800 rows × 13 columns

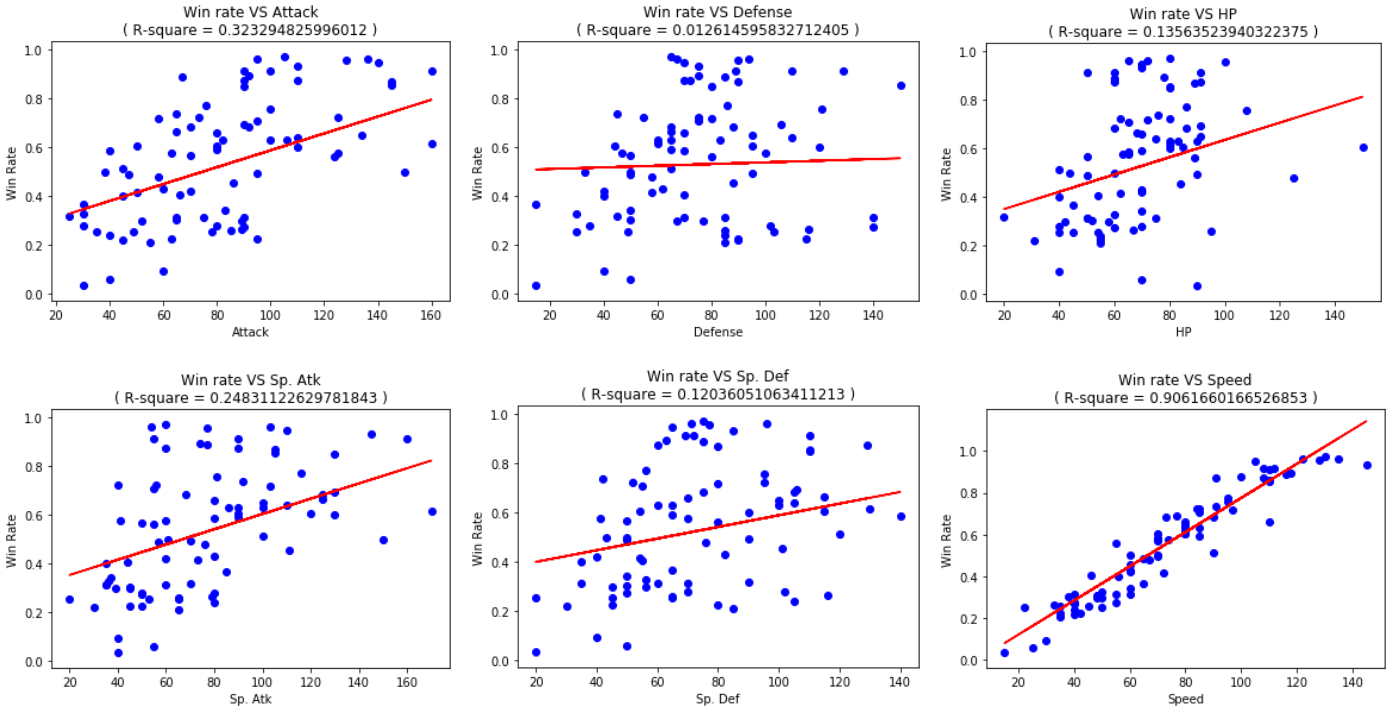
Task 2: Finding the most correlated feature using Linear Regression

I dropped the NaN win rate values, split the train data and test data in 8:2 and random_state = 4211. Then, I plot the linear regression model and calculate the R2 score. The whole procedure is done at linear_regression.ipynb.

Q2: Report the validation R2 score of each model to evaluate its prediction performance.

Q3: Plot the regression line and the data points of the validation set for each of six models.

Q4: Find the feature that is most correlated to the win rate. Explain how you find it.



R2 score and graphs plotted are shown above.

Speed is the most correlated feature. We can see R2 score is the highest among 6 features and the scattered point are not discrete with the regression line.

Task 3: Legendary Pokemon Classification

This task is done at logistic_neural.ipynb

Logistic Regression:

Q5: Report model setting, training time, and performance of the logistic regression model. Also report the mean and standard deviation of the training time, accuracy and F1 score for each setting

I used SGDClassifier to perform the logistic regression. The settings are:

Changing parameters:

- Random state: 0, 1, 4, 9, 16, 25, 36, 4211

Defined parameters (same at all changing cases):

- loss: ‘log’
- max_iter: 500

Others are set as default.

	Training time	Accuracy	F1-score
Mean	0.0055132	0.9468750	0.6940832
SD	0.0003803	0.0116926	0.1109332

Single-hidden-layer Neural Network:

Q6: Report model setting, training time, and performance of the neural networks for each H value. Report the mean and standard deviation of training time, accuracy and F1 score.

Used MLPClassifier to perform neural network training. The settings are:

Changing parameters:

- hidden_layer_sizes: 1,2,4,8,16,32,64

Defined parameters (same at all changing cases):

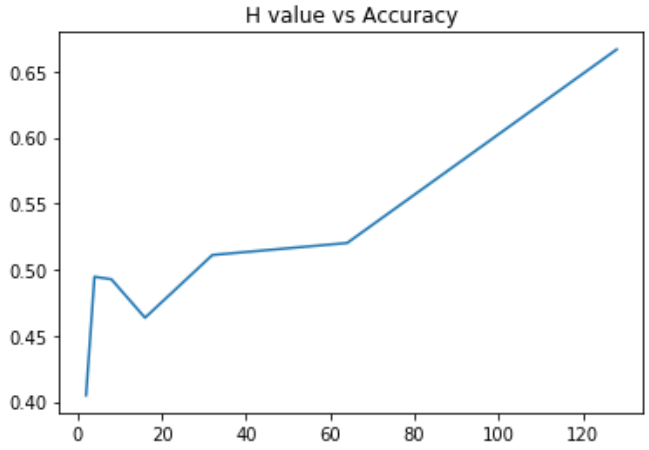
- max_iter: 500

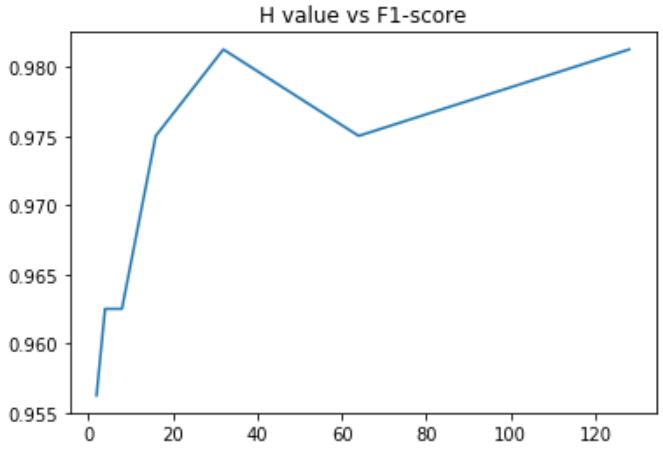
Others are set as default

	Training time	Accuracy	F1-score
Mean	0.5077272	0.9705357	0.9138067
SD	0.0740454	0.0092788	0.0310263

Q7: Compare training time, accuracy and F1 score of logistic regression and best neural network

Q8, 9, 10: Plot accuracy and F1 score for different values of H. Notice some trend and comment the gap between accuracy and F1 score.





We notice that F1-score and accuracy is increasing when H values increase. However, accuracy is far lower than F1-score. This may due to the model is overfitting so that score based on precision and recall is high.

H value with 64 is the best. Therefore, we collect the data from H = 64

	Training time	Accuracy	F1-score
H = 64	0.6667740	0.9812500	0.9464225

Compare to logistic regression, the three values are higher in neural network best model.

Task 4: Predicting the Winners in the Pokemon Battles

This task is done at Winner.ipynb

I choose GridSearchCV to have hyperparameter search.

Q11: Report 10 combinations of the hyperparameter setting.

My dynamic hyperparameter is set like this:

- hidden_layer_sizes: 2, 4, 8
- activation: relu, tanh
- solver: adam, sgd

As there are 3 x 2 x 2 = 12 combinations available, we report the 10 of it.

Combination	Hidden layer size	Activation	Solver
1	2	relu	adam
2	2	relu	sgd
3	2	tanh	adam
4	2	tanh	sgd
5	4	relu	adam
6	4	relu	sgd
7	4	tanh	adam
8	4	tanh	sgd
9	8	relu	adam
10	8	relu	sgd

Q12: Report three best hyperparameter settings as well as the mean and standard deviation of the validation accuracy of the five random data splits for each hyperparameter setting.

Rank 1 model:
Mean : 0.90246875 , SD : 0.0034536552194740036
With parameters: {'activation': 'relu', 'hidden_layer_sizes': (4,), 'solver': 'adam'}

Rank 2 model:
Mean : 0.8990625 , SD : 0.003947902924338457
With parameters: {'activation': 'relu', 'hidden_layer_sizes': (2,), 'solver': 'adam'}

Rank 3 model:
Mean : 0.8984687499999999 , SD : 0.0025221673467476355
With parameters: {'activation': 'tanh', 'hidden_layer_sizes': (2,), 'solver': 'adam'}

Reference from Winner.ipynb.

Q13: Report the accuracy of best model prediction.

0.90 from classification report

Q14: Print confusion matrix.