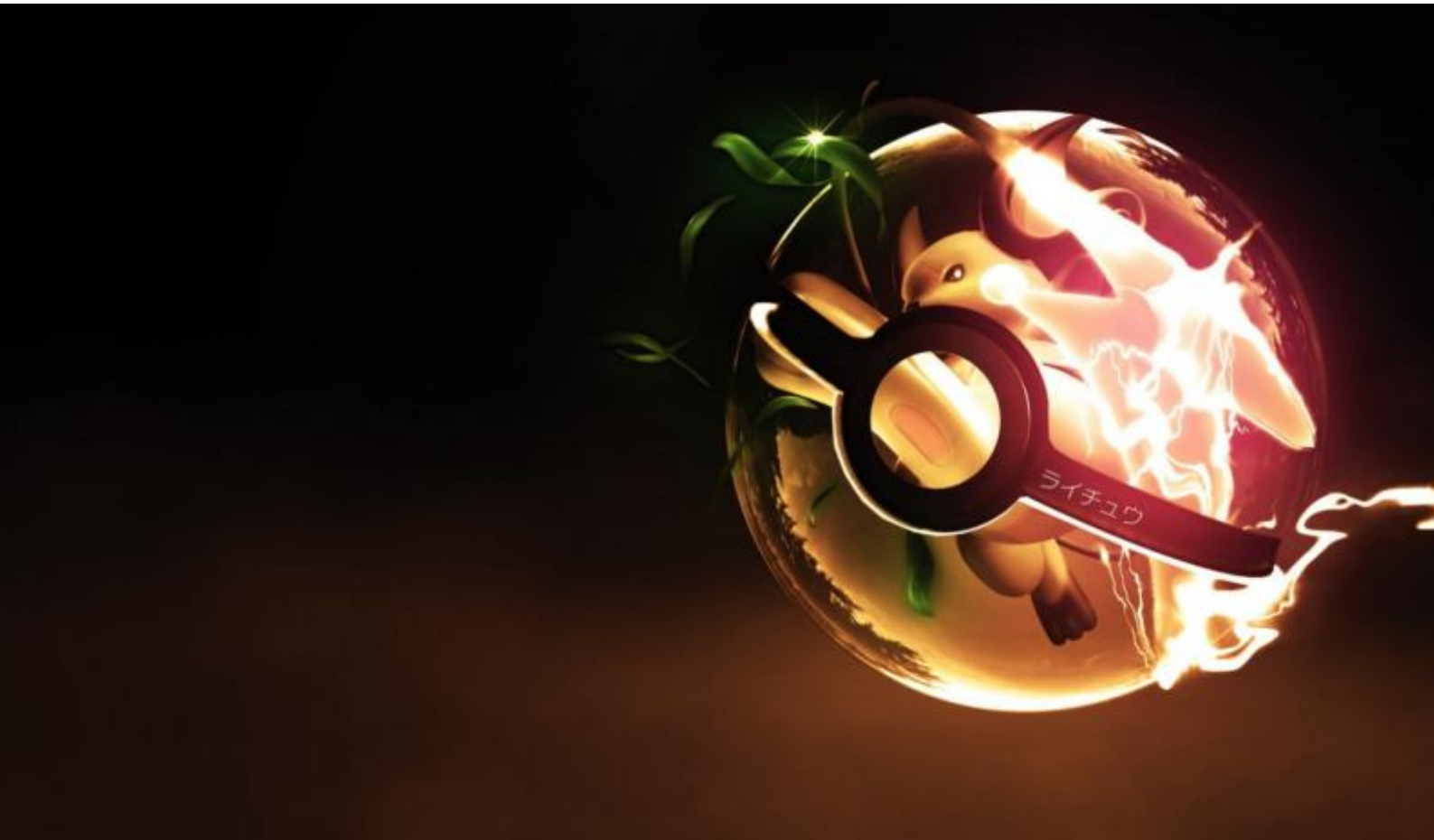
Pokémon Classification Project

Charles Christopher Hyland 450411920

Rowena Kok 440256380

Table of Content



Modeling

Model Technique

Due to our response variable of Pokémon type being a categorical variable, we can classify this problem as a classification problem. This means that given our predictors, we want to be able to classify which type a Pokémon belongs to. However, due to the curse of dimensionality, it will lead to much less accurate predictions from our problems due to the wide range of potential types we can classify a Pokémon to be in. Therefore, to improve our predictions and for ease of simplicity, we can convert the problem from a multi-class classification problem into *n* binomial classification problems, where *n* is the number of primary Pokémon types. For an example, instead of classifying whether a Pokémon X is a fire, water, or bug type, we can instead run 3 binary classification problems and ask 3 separate questions on whether is X a fire type or not, whether it is a water type or not, and whether is it a bug type or not. From this, the multi-binary classification problem will say yes to one of these questions, and therefore give us our solution.

This is in line with our data mining goal as we wish to produce a model that is able to classify a Pokémon given its feature. We wish to formulate the problem in such a way that will allow us to generate the best classifier without much regards to interpretation of the models. Our multi-binary classification formulation is therefore in line with our data-mining goal whereby we simply require a single output predicting to which class each Pokémon belongs.

For a multi-binary classification, need to have a sufficient variety of data and sufficient observations for each class. Ideally, we require a size of at least 20 Pokémon for each Pokémon type in our dataset to assist our task. This will ensure there is sufficient data about each Pokémon type in order for our models to be able to find patterns and associations for each Pokémon type. Except for the fairy and flying types, we have sufficient observations for the Pokémon types.

We currently have over 800 observations of Pokémon in our dataset. This is a relatively small sample but will still allow for us to split the data into a training and test set. Considering there aren't that many Pokémon, this is in fact the current Pokémon population for 6 of the 7 largest different regions in the Pokémon world and therefore it is not actually feasible to obtain many more observations. Despite this, the data we have gathered is of high quality as it is complete and quite accurate with no missing features. However, there were a few features that we need to convert from a categorical variable into a numerical dummy variable for us to work with the data better and to allow for Python to work with data. In particular, these features included the 2nd type of a Pokémon and its status on whether or not it was a legendary Pokémon.

We have used a plethora of models in our modelling.
Since we are using supervised methods, we require the assumption that all data types are known before execution which we have already fulfilled. We also assumed a binomial distribution for our multi-binary classifications. Finally, we know that the response variable is a binary discrete variable.

The models we have used are listed in the table alongside their assumptions:

Model	Assumptions
Naïve Bayes	Independence of features between each other given the class
Linear Discriminant Analysis	Conditional probability density functions are normally distributed Homoscedasticity such that the class covariances are identical Means of classes are the same
Quadratic Discriminant Analysis	Conditional probability density functions are normally distributed Heteroscedasticity such that the class covariances are not identical
Logistic Regression	Independent error terms Linearity of independent variables and log odds Large sample size Model is correctly classified No perfect multicollinearity in features
Decision Tree	Data can be described by the features Class label can be predicted using logical set of decisions summarised by decision tree Independence
Random Forest/Extremely Random Forest	At each step of building individual trees, we find the best split of the data We build the tree by using bootstrap sample rather than whole dataset Aggregate tree outputs by averaging
Adaptive Boosting/Grad	Weak learning assumption whereby we assume our weak learning algorithm can find weak classifiers which can classify the data better than a 50% guess
Gradient Boosting	Independence of observations Assumptions related to interaction depth

Test Design

We will use **accuracy** to help measure the goodness of a model. Accuracy measures the percentage of correct classifications we make, i.e. $accuracy = \frac{true\ positives + true\ negatives}{total}$. Our data mining goal aims for this to be as high as possible.

In order to compute the accuracy, we have split up the data into a training and test set. We train our model on our training set. Furthermore, we tune the hyperparameters required for some of the models via cross-validation on the training set. We then estimate the test dataset with our models and compare whether were the model's classifications were accurate or not. Therefore, the accuracy criteria will be constructed from the test data. We predict the test

dataset only once as the hyperparameters we have selected from our cross-validation approach should be the optimal hyperparameters.

Model Construction

Some of the models we used contain some hyperparameters we need to tune. The models that require tuning include:

Model	Hyperparameters
Decision Trees	Maximum depth, maximum features, minimum sample to split an internal node, and criteria used.
Random Forests/Extremely Random Forest	Maximum depth, maximum features, minimum sample to split an internal node, minimum sample required to be a leaf node, and number of estimators.
Adaptive Boosting	Number of estimators and the learning rate
Gradient Boosting	Loss function used, number of estimators, the learning rate, minimum sample to split an internal node, and minimum sample required to be a leaf node

We used cross-validation in order to train our hyperparameters. We specify a range of potential values for the numerous hyperparameters we wish to tune for each model. We then iterate through all possible combination of hyperparameter values. In particular, we split the training dataset into k folds, and with that, we train the model on k-1 folds and predict the last fold. We repeat this process such that each fold gets a chance at being predicted. We then record the accuracy of our predictions as a whole with those particular hyperparameters settings. The final hyperparameters used for the model are the hyperparameters which led to the best predictions for the k-folds. However, this is costly to run computationally wise, and therefore we employed randomised search. This does not iterate through every possible combination of hyperparameters but instead terminates on a particular combination if previous similar combinations were not promising. This saves computing time.
From this, we then combine the 3 best models in order to create a voting ensemble. This takes a vote from each model and depending on the votes as to which Pokémon type to classify a particular Pokémon, we assign the Pokémon to that class. We specified 3 models because we require an odd number of models in order to ensure that there is a majority vote for each classification we make. A further step we utilised is soft voting, whereby we give a higher vote to the models that are seen as better at classifying Pokémon based on their accuracies as single base models.

Naïve Bayes

Naïve Bayes is a simple probabilistic classification technique that was employed. It is optimal under 0-1 loss, applying Bayes' theorem in order to predict the class of a given observation. The Bayes' classifier assigns a certain Pokemon X to type k if the conditional probability of X being k given its particular features is greater than the conditional probability of X being any other type. This posterior probability can be calculated from the prior probability of being class k, the likelihood of the features given class k and the evidence, as in Bayesian probability, which we can do since we have assumed the conditional independence of all the features.

Linear Discriminant Analysis (LDA)
LDA models the conditional distribution of the features separately for each class, restricting the covariance of each of the classes to be identical as in homoscedasticity. Predictions are made similarly as in the Naïve Bayes model, where we select the class k that maximises the conditional probability determined by Bayes' rule.

Quadratic Discriminant Analysis (QDA)
QDA is similar to LDA but does not restrict results by assuming homoscedasticity. This results in a likelihood ratio that allows for a quadratic separating surface between classes.

Logistic Regression
This model was also utilised due to the categorical response variable (type) and its conditional probability following a Bernoulli distribution. Using this fact, we can compute the likelihood and thus log likelihood of the data, which we maximise in order to calculate the coefficients of the regression model. The regression equation fitted to the training set can then be used to classify the test set.

Decision Tree
A decision tree constructs a tree like structure whereby the nodes are the features of the dataset. The leaf nodes are the final classification prediction for that particular observation. There are many ways to construct decision trees based on different methods of splitting the features. Such methods can include looking at the Gini coefficient or entropy each time we split a tree.

Random Forest
Random forests was another technique employed into the creation of the model. Random forests are extensions of decision trees, whereby multiple decision trees are created through the process of taking a bootstrap sample from the training set. Unlike typical decision trees which allows for splitting along the entire feature space, random forests injects a stochastic element into the algorithm by forcing each decision tree to be constructed based on a randomly selected subspace of the features. This forces different decision trees to be constructed. This will construct more robust predictions as unlike the single decision tree case which can overfit due to some features, the fact that the random forest decision trees has less features to construct trees means that it is much more robust to overfitting.

Extremely Random Forest
Extremely random forests are identical to random forest except we inject a further layer of randomness in the process. Random forests will optimise splits on trees whilst extremely random forests will randomise the splits on the trees too.

Adaptive Boosting
The technique of boosting was employed. Boosting is a technique whereby 'weak learners' are combined to create a single 'strong learner'. The mechanism behind boosting is that you have multiple rules (weak learners) which must at least have a higher than 50% chance of being able to correctly predict values. Through this, boosting combines the numerous weak learners into a single strong learner through a plethora of methods such as taking the average or weighted average of the weak learners.

Gradient Boosting
Another boosting model employed was gradient boosting, whereby the algorithm trains multiple models sequentially. Each new model aims to minimise the 0-1 loss function through the process of gradient descent, an optimisation technique by traversing the negative gradient of a function in order to find a local minimum of the loss function. Gradient boosting construct learners such that they will be maximally correlated with the negative gradient of the loss function and therefore allow for it to be an easier process to locate the minimum of the loss function.

Voting Ensemble
The final ensemble includes a voting ensemble of an extremely random forest, logistic regression, and gradient boosting. We selected these 3 models as they showed extremely high accuracy as models by themselves. Furthermore, these models construct their predictions in different manners, and therefore the idea is that logistic regressions may be quite capable at predicting bug type Pokémon whilst being quite poor at predicting normal type Pokémon relative to extremely random forests. From this, each model's weaknesses will be balanced out by the strengths of the other models in the ensemble vote.

Model Assessment

In the table below we can see that our accuracy has improved for this final ensemble compared to the individual models, thereby suggesting that this is the superior model.

Model	Mean Accuracy
Naive Bayes	0.426
LDA	0.927
QDA	0.67
Logistic Regression	0.942
Decision Tree	0.941
Random Forest	0.944
Extremely Random Forest	0.944
Adaptive Boosting	0.923
Gradient Boosting	0.936
Final ensemble	0.944

The processing time did not take too long to construct the base model and final ensemble since we utilised techniques such as randomised grid search which dramatically cut down the model creating time.

This final ensemble is not quite ready to deploy in real time Pokémon battle scenario due to the processing time (5-10 minutes) in time-critical scenarios. However, our model would be quite effective in a post-Pokémon sighting scenario. Therefore, we propose the logistic regression model to be deployed in a real-time scenario as it has a relatively fast computation time whilst still having a high accuracy rate.
Our models meet the criteria of high accuracy.

There were no unusual patterns revealed by the models and no calculation inconsistencies. This was definitely a result of no particular data quality issues faced in our dataset.

Revised parameter setting

There is no particular need to revised parameters as the optimal hyperparameters were already selected at the beginning of the process. Furthermore, we have tried a plethora of different models and even combined them. One potential task for the future is to try further voting ensembles and see if any improvements can be made there.

Evaluation

Result Evaluation

This study produced overwhelmingly successful results in terms of accurately classifying Pokemon into their respective types – the accuracy measure is very intuitive, simple to communicate as well as interpret, and thus is attractive from a business perspective. However, understanding the details of how each of the models involved in the final ensemble were constructed, as well as the voting process itself may be more difficult.

Additionally, the accuracy measure may be misleading since it may not be comparable across different classifiers. Further, the binary variable we created for each type contains a large majority of negatives, and accuracy does not account for this. For example, 13 of the 240 test observations were bug type – this is equivalent to 94% of the test set. This means that even if the classifier predicted that all 240 observations were not of bug type, it would have 94% accuracy, although 100% of true bug types were incorrectly classified as not-bug. To analyse this further and gain a more holistic perspective, we consider the following additional metrics, with results for each model in the table below:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

$$F\text{-}measure = \frac{2 \times True\ Positives}{2 \times True\ Positives + False\ Negatives + False\ Positives}$$

Model	Mean Precision	Mean Recall	Mean F-measure
Naive Bayes	0.077	**0.804**	0.139
LDA	0.244	0.169	0.176
QDA	0.097*	0.434	0.091
Logistic Regression	0.386*	0.026	0.040
Decision Tree	0.353*	0.039	0.057
Random Forest	0.469*	0.026	0.045
Extremely Random Forest	0.500*	0.030	0.050
Adaptive Boosting	0.204*	0.129	0.148
Gradient Boosting	0.280	0.142	**0.183**
Final ensemble	**0.520***	0.066	0.105

* Not all 18 types were included in this calculation as no positives were predicted in the test set.

From this, we can see that the accuracy measure does indeed exaggerate the success of the models. The precision metric can be interpreted as the percentage of positive predictions that

were correct. The table above indicates that the final voting ensemble performed the best with respect to this metric, and naïve Bayes performed the worse – this is in line with the accuracy measure.

As previously stated, given the high proportion of negatives for each type's binary response variable, it is possible that no positives would be predicted, is indicated by * in the table above. This can be further analysed by observing the recall measure, which represents the percentage of true positives that were correctly predicted as being positive. Surprisingly, the naïve Bayes model greatly outperformed all other models according to this metric. From its low precision and high recall, we can infer that it is too optimistic in determining a positive value for each type. Conversely, from the other models' moderate precision and low recall, it appears that they are much more conservative in classifying Pokemon as a certain type.

The F-measure can be interpreted as representing a combination of both precision and robustness. While all models had relatively low F-measure scores, the gradient boosting model performed the best.

As such, it appears that our models are more useful in ruling out what type a given Pokemon is, rather than correctly predicting a specific type. In regards to the original business success criteria, the goal of being able to classify all 18 types of Pokémon has been met, although each type can be predicted with varying accuracy. Further testing is required in order to determine the performance of the models for observations with incomplete information, given that one of the criteria was to be able to classify Pokemon with a quarter of the features having incomplete information.

Given the models we have produced and taking into consideration the evaluation above, our final list of models that both satisfy the data mining criteria and are acceptable in terms of our business goals include the final voting ensemble (with the highest accuracy), as well as the individual models that comprise this, which would be more practical in a real-life situation: extremely random forest, logistic regression, and gradient boosting.

Although we have completed one of the original three main business objectives – being able to identify a Pokemon's primary type – testing different combinations for the final ensemble, and adjusting the criteria for selection would be useful. The remaining business objectives are discussed in the Next Steps section below.

Process Review

Reviewing the process of the initial data mining project as well as the modelling stage allows us to appreciate the quality and completeness of the data that was collected as it resulting in higher quality models, and enabled greater focus on the modelling phase and evaluation of the results. However, given that some of the types (fairy and flying) did not have an ideal number of occurrences in the dataset, the predictive power of the final model may have been affected. In the future, this may be avoided by employing different sampling techniques that would result in greater numbers of these types in the dataset.

The data exploration report highlighted the relationships and notable aspects of certain attributes, demonstrating the relevance of all attributes, which were eventually all included in the final model. In line with the cyclical nature of CRISP-DM, it may be worthwhile to return to the data selection phase to determine how different combinations of attributes perform in

various models. Additionally, transforming categorical variables into binary variables in the data preparation stage was essential for the modelling.

However, in future stages of this project, metrics in addition to accuracy will be considered in the modelling phase in order to provide a more informative perspective of the models and allow for greater alignment with the business objectives.

Next Steps

The next stage is the deployment phase, where the effectiveness of the logistic regression algorithm we have a developed will be assessed using real world, unseen observations. Our model is ready to be implemented in a number of specially selected Pokemon trainers' upgradable Pokedexes, though this will be monitored under beta testing conditions.

Although not as high as that of the final ensemble, we are still confident in the accuracy of the logistic regression model, however, will continue to train our model as new data is collected in order to further improve accuracy and predictive power. As mentioned before, processing time is of great importance, especially in a battling scenario wherein the ability to quickly identify the type of an unknown attacking Pokemon could be critical in determining strategy and protecting the trainer and the defensive Pokemon.

Future steps involve completing the remaining business objectives, that is, extending the model to being able to identify a Pokemon's secondary type and whether or not it is potentially a legendary Pokemon. These were not implemented due to time constraints – focusing on developing a model to identify a Pokemon's primary type was considered of greater importance in order to be able to readily deploy this as a deliverable. As the deployment phase continues, we will be able to continue to develop the latter models and incorporate additional data that will be collected as the initial model is deployed.