Legendary Pokémon Classification and Prediction Using Machine Learning

İlhami Yılmaz
Institute of Informatics
Hacettepe University
Ankara, Türkiye
ilhamiyilmaz.iy@gmail.com

Kemal Öztürk
Institute of Informatics
Hacettepe University
Ankara, Türkiye
kemalozturk00@gmail.com

***Abstract*—The "Legendary Pokémon Classification and Prediction Using Machine Learning" project leverages advanced machine learning algorithms to identify and predict legendary Pokémon from a dataset obtained on Kaggle.com. Pokémon, fictional creatures with diverse characteristics and abilities, have captivated enthusiasts for decades. This project focuses on employing state-of-the-art classification algorithms to automatically distinguish legendary Pokémon from the dataset and make predictions on their legendary status.**

The dataset encompasses various attributes such as abilities, type, attack, defense, and more, providing a rich source for training and evaluating machine learning models. The primary objective is to explore, implement, and compare multiple classification algorithms, selecting the most suitable model for accurately identifying legendary Pokémon. Additionally, the project delves into predicting whether a Pokémon is legendary based on its attributes.

Key steps in the project include data preprocessing, handling missing values, exploring feature correlations, and employing classification algorithms such as logistic regression, decision trees, and random forests. Evaluation metrics, including accuracy, precision, recall, and F1-score, will be utilized to assess the performance of the models. The project aims to not only create an efficient classification system but also gain insights into the characteristics that define legendary Pokémon.

The outcomes of this project hold significance for Pokémon enthusiasts, researchers, and anyone interested in the application of machine learning to character recognition within diverse datasets. Through this exploration, we anticipate contributing valuable insights to the intersection of machine learning and the captivating world of Pokémon.

Keywords—machine learning, Pokémon classification, predictive modelling, data science, Kaggle dataset, classification algorithms, model training, feature selection, data preprocessing, decision trees, random forest

I. Introduction

The integration of machine learning techniques into diverse domains has revolutionized the way we analyze and interpret complex datasets. In this context, our project, titled "Legendary Pokémon Classification and Prediction Using Machine Learning," applies state-of-the-art classification algorithms to a dataset sourced from Kaggle.com. The objective is to employ these algorithms to identify and predict legendary Pokémon, fascinating fictional creatures with distinct attributes and capabilities.

The project leverages a comprehensive dataset containing information on various Pokémon attributes, such as abilities, types, attack and defense stats, and more. By employing machine learning algorithms, we aim to discern patterns within the dataset that differentiate legendary Pokémon from others. This endeavor is not only an exploration of machine learning techniques but also a captivating journey into the realm of Pokémon classification.

The primary goals of our project include:

1. Classification: Utilize machine learning algorithms to accurately classify Pokémon as legendary or non-legendary based on their attributes.
2. Prediction: Develop models capable of predicting whether a Pokémon is legendary, enhancing the understanding of the factors contributing to a Pokémon's legendary status.
3. Algorithm Comparison: Evaluate and compare the performance of various classification algorithms to identify the most effective method for this specific dataset.

As we embark on this project, we anticipate contributing insights into the application of machine learning in the context of character recognition within a unique and beloved dataset. The subsequent sections of this paper will delve into the dataset, methodology, results, and discussions, providing a comprehensive overview of our approach and findings.

II. Data Description And Pre-Processing

A. Dataset

The dataset is retrieved from Kaggle.com and it contains the details of the Pokemon Characters. This data contains 801 characters and 41 variables. The data has 522 missing points. Dealing with missing points are going to be covered in the following part of the analysis. The main variable description is given below.

- *name*: The English name of the Pokemon

- *japanase_name*: The original Japanese name of the Pokemon
- *pokedex_number*: Shows the id of the Pokemon in the Pokedex
- *percentage_male*: The percentage of the species that are male. Blank if the Pokemon is genderless.
- *type1*: The Primary Type of the Pokemon
- *type2*: The Secondary Type of the Pokemon
- *classification*: The Classification of the Pokemon as described by the Sun and Moon Pokedex
- *height_m*: Height of the Pokemon in meters
- *weight_kg*: The Weight of the Pokemon in kilograms
- *capture_rate*: Capture Rate of the Pokemon
- *base_egg_steps*: The number of steps required to hatch an egg of the Pokemon
- *abilities*: Abilities that the Pokemon can have.
- *experience_growth*: The Experience Growth of the Pokemon
- *base_happiness*: Base Happiness of the Pokemon
- *against_?:* Eighteen features that denote the amount of damage taken against an attack of a particular Pokemon type
- *hp*: The Base HP of the Pokemon
- *attack*: The Base Attack of the Pokemon
- *defense*: The Base Defense of the Pokemon
- *sp_attack*: The Base Special Attack of the Pokemon
- *sp_defense*: The Base Special Defense of the Pokemon
- *speed*: The Base Speed of the Pokemon
- *generation*: The numbered generation which the Pokemon was first introduced.
- *is_legendary*: Shows if the Pokemon is legendary.

B. Descriptive Statistics

Since the data have 41 variables, showing the descriptive statistics for all variables is a bit unnecessary. Descriptive statistics helps us to understand the behavior of the data. To have an insight of the data, Table 1 is shown for some of the main variables.

	hp	attack	defense	sp_attack	sp_defense	speed	height_m	weight_kg
count	801	801	801	801	801	801	801	801
mean	68.96	77.86	73.01	71.31	70.91	66.33	1.16	61.38
std	26.58	32.16	30.77	32.35	27.94	28.91	1.07	107.98
min	1	5	5	10	20	5	0.1	0.1
25%	50	55	50	45	50	45	0.6	9.3
50%	65	75	70	65	66	65	1	28.5
75%	80	100	90	91	90	85	1.5	61.5
max	255	185	230	194	230	180	14.5	999.9

Table 1 Descriptive Statistics of Some Features

C. Exploratory and Confirmatory Data Analysis

In this part of the study, the data will be examined and necessary data transformations will be made. In addition, the features required for classification and prediction will be determined using the correlation matrix.

C.1 Data Exploration and Preprocessing: Feature Engineering and Scaling in Pokémon Dataset

Firstly, the data in the "type1" and "type2" columns, which are crucial in the dataset, have been examined. It has been observed that there are 18 different Pokemon types in the content of the data. It is predicted that these types will contribute to the classification and prediction processes. Therefore, a one-hot encoding transformation has been applied to both columns. Through this transformation, each type has been converted into a column in the data, with each column having values in the range of 0-1 as dummy variables. After the transformation, these columns have been added to the original data, and the "type1" and "type2" columns in the original data have been removed. Additionally, columns "classification" and "abilities" in the original dataset, which are predicted to have minimal contribution to the classification and prediction processes, have also been removed. This is due to both columns containing, respectively, "588" and "482" different categories.

As the second step, a "MinMaxScaler" transformation has been applied to all the data in our dataset. The purpose of this transformation is to prepare for classification and prediction. It is predicted that normalizing all values in the dataset will help machine learning models in achieving better performance. As a result of the transformation, all values in the dataset have been scaled to a range of 0-1.

Following the completion of transformations, dependent and independent variables in our dataset have been determined. The column determined as the dependent variable in our study is the "is_legendary" column.

C.2 Correlation Matrix

Correlation matrix is a widely used statistical metric that displays the correlation coefficients between variables. This process is highly beneficial for understanding how variables affect each other, and interpreting the matrix is quite straightforward. However, our dataset contains 71 variables, and creating this matrix with all variables would adversely impact interpretation. Hence, the matrix was generated with some column names hidden. The purpose of creating this matrix is due to the often-undesirable nature of high correlation in feature selection. Features with correlation among themselves can cause problems during model creation. High correlation between one or more features can lead to overfitting, loss of generalization ability, and decreased accuracy of predictions. In such cases, the model may not distinguish the variation of one feature from that of others. This can cause decrease model's reliability.

Therefore, in feature selection for classification problems, features with high correlation should be examined, and when necessary, these features should be excluded from the model. Removing features with high correlation can enhance model performance.

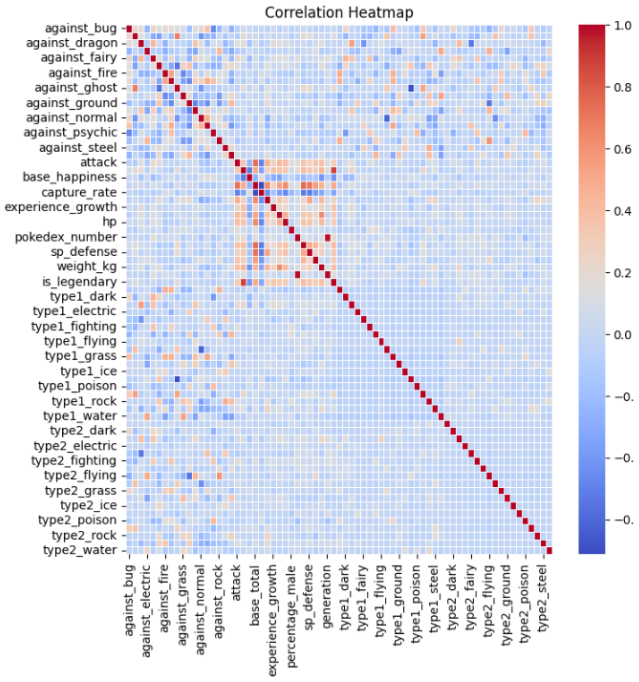

Figure 1 Correlation Matrix

As seen in the figure above, correlation coefficients between all variables have been examined. As a result of this examination, variables with correlation coefficients above 0.95 have been excluded from the dataset. One such variable with correlation coefficient above 0.95. This variable which is "generation" has been observed and removed from the dataset.

III. MODELLING

The dataset is now ready for modeling as data exploration and preprocessing have been completed, and necessary variables have been selected. The dataset are split into two parts as training and test data. The training dataset contains 80% of the observations, which equals to 640 observations. Similarly, the test dataset contains the other 20% of the observations, equals to 161 observations. The model will be conducted to the training dataset, and the test dataset will be used to check the errors of the model.

A. Decision Tree

A Decision Tree is a model structured as a tree used to classify a dataset and make class predictions for new incoming data. During the training phase, the model creates a set of decision rules based on the features present in the dataset, and it classifies or makes predictions using this rule set.

The tree structure starts with a root node and consists of a series of internal nodes and leaf nodes. Internal nodes contain a test representing a specific feature. Leaf nodes represent a class or a numerical prediction value. Each internal node tests a feature and determines a decision rule based on the result of the test. Depending on the test outcome, the tree follows a specific subtree. This decision process iteratively continues, with each internal node adding a new feature test. Decision Trees evaluate features to select the best splits. The splitting criterion is typically determined by measures like information gain or Gini index.

Decision Trees have a flexible structure that can be controlled with regularization parameters such as pruning and settings like maximum depth. This flexibility prevents the model from overfitting to the training data.

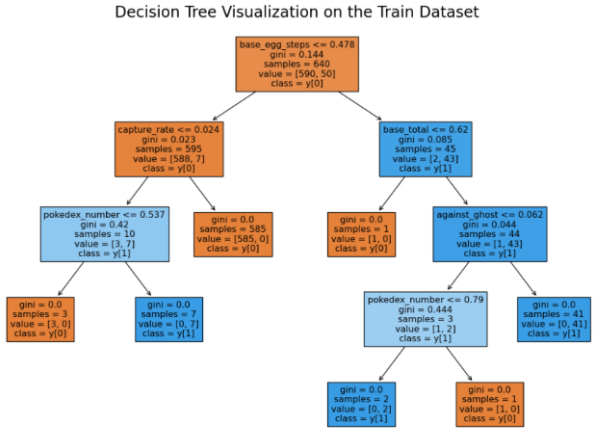

Figure 2 Decision Tree

The Decision Tree provided above is the model created specifically for our project. As mentioned earlier, 80% of the data is used as the training dataset, and 20% is used as the test dataset. After making predictions, it was observed that the model performed with an accuracy of 98.14%. As shown in the table below, only 3 values belonging to the legendary Pokemon class were incorrectly predicted. This indicates that the Decision Trees model is well-suited for the dataset used in the project.

	Actual Value Not Legendary	***Actual Value Legendary***
Predicted Value Not Legendary	141	0
Predicted Value Legendary	3	17

Table 2 Confusion Matrix

B. Gaussian Naïve Bayes

Gaussian Naive Bayes is a type of Naive Bayes classification algorithm. It is fundamentally based on Bayes' theorem and gets its "naive" name from the assumption that features are independent of each other. It assumes independence among each feature, meaning the value of one feature in an example provides no information about the values of other features. Under this assumption, it performs classification and prediction. The "Gaussian" prefix in this algorithm signifies that the features have a normal (Gaussian) distribution, indicating that feature values are typically symmetrically distributed around a center.

Like other Naive Bayes algorithms, Gaussian Naive Bayes uses Bayes' theorem. This involves using prior information (training data) and new feature values to update class probabilities. It is a fast and simple classification algorithm that is easy to implement. It generally performs well if the assumptions are true. However, the assumption of independence often is not true. If there are categorical features in the dataset or strong dependencies between features, the performance may decrease.

In the project, the accuracy value of the applied Gaussian Naive Bayes method is 59.006%. This indicates that the dataset used in the project is not very suitable for this model. This implies that the assumptions mentioned above do not hold for the dataset used. The model misclassified 63 non-legendary Pokemon values as legendary and 3 legendary

Pokemon values as non-legendary. You can see the relevant values in the table below.

	Actual Value Not Legendary	***Actual Value Legendary***
Predicted Value Not Legendary	78	63
Predicted Value Legendary	3	17

Table 3 Confusion Matrix

C. Logistic Regression

Logistic Regression is a statistical model used to solve classification problems. Despite its name containing "regression," it is, in fact, a classification algorithm, commonly used for binary classification problems.

Logistic regression multiplies input features by a set of weights and adds a term that includes the sum of these weights. The result goes through a sigmoid function. The sigmoid function compresses the output, interpreting it as a probability value between 0 and 1. The model learns by updating weights on training data, usually accomplished through an optimization algorithm (often gradient descent). Logistic regression creates a decision boundary that separates the feature space into two classes. This boundary is used to classify points in the feature space. It is a fast training algorithm, and prediction processes are efficient. It directly provides class probabilities. However, it creates a linear decision boundary, which may be insufficient for some problems especially for direct application to multi-class classification problems.

The logistic regression model was applied to the Legendary Pokemon classification problem, achieving a prediction accuracy of 96.9%. Since the problem involves binary classification, the model performed well with a high accuracy percentage. The point of error in predictions was the misclassification of 5 values that are Legendary Pokemon but predicted as non-legendary. You can see this in Table 4.

	Actual Value Not Legendary	***Actual Value Legendary***
Predicted Value Not Legendary	141	0
Predicted Value Legendary	5	15

Table 4 Confusion Matrix

D. Support Vector Machine

Support Vector Machine (SVM) is a powerful machine learning algorithm used for classification and regression problems. Its fundamental goal is to create a hyperplane to classify data points. This algorithm, which is effective on medium and small size data sets, especially in cases where there is a clear boundary between classes, is a robust algorithm that can perform efficient classification.

SVM classifies data by creating a hyperplane between two classes. This hyperplane is positioned to best separate data points into two classes. Support vectors form the foundation of SVM, representing the data points that are closest to the hyperplane and contribute to class separation. For the nonlinear relationships part, SVM can use the kernel trick, which involves mapping data points into a higher dimensional space into which a linear hyperplane can effectively classify. SVM can work effectively in multi-dimensional spaces and use the kernel trick for nonlinear classifications. Training SVM on large datasets can be time-consuming, and hyperparameter tuning may be necessary to manage model complexity.

Before using the SVM model, the data in the project was examined, and some improvements were made with the aim of obtaining the best performance from the SVM model. The hyperparameters in the SVM model and the selection of the kernel function were optimized for the most suitable parameters based on the data. Cross-validation was performed for this purpose, using the Grid Search algorithm. This algorithm essentially tests all possibilities in the given parameter pool according to the project data and selects the parameters that give the highest accuracy results. The selected parameters for prediction in the model are 'C': 0.01, 'gamma': 1, 'kernel': 'poly'. Using these parameters, the model predicted only 5 legendary Pokémon as non-legendary. The accuracy rate was 96.9%. The prediction table for the respective model is provided below.

	Actual Value Not Legendary	***Actual Value Legendary***
Predicted Value Not Legendary	141	0
Predicted Value Legendary	5	15

Table 5 Confusion Matrix

E. XGBoost Model

XGBoost is a machine learning algorithm used for classification, regression, and ranking problems. It is a derivative of the Gradient Boosting method and is particularly known for its high performance, speed, and generalization ability.

XGBoost builds a robust model by combining often decision trees or another classification models. The model focuses on correcting the errors of previous trees. Each tree is trained to minimize the remaining errors compared to the previous one. Learning iteratively occurs through the error term, leading to the development of an overall model. This characteristic ensures the speed and scalability of the XGBoost model. An another important feature is its ability to handle missing data, providing flexibility when dealing with incomplete datasets. The model is highly flexible for optimization, with numerous hyperparameters.

In the project for the classification and prediction of Legendary Pokémon, the XGBoost model performed consistently well. Similar to SVM, cross-validation was used during the model preparation. The Grid Search algorithm, as in SVM, was used to optimize hyperparameters for the model. The parameters 'colsample_bytree': 0.8, 'learning_rate': 0.2, 'max_depth': 3, 'subsample': 1.0 were selected using the Grid Search algorithm. This ensured maximizing the accuracy of the model. The accuracy of the model reached 98.76% in the prediction results with using

these specified parameters. The performance details of the model's predictions are shown in Table 6.

	Actual Value Not Legendary	***Actual Value Legendary***
Predicted Value Not Legendary	141	0
Predicted Value Legendary	2	18

Table 6 Confusion Matrix

F. K-Nearest Neighbors(KNN) Model

KNN is a simple and effective machine learning algorithm used for classification and regression problems. It looks at the labels of the k nearest neighboring data points to classify a data point or make a prediction. The most frequently occurring class or the average value is used as the prediction. During the training phase, the model stores examples from the dataset in its memory. When making predictions for a new data point, it identifies the k nearest neighbors of that point and uses the majority class or average value as the prediction. The value of k is a hyperparameter determined by the user, representing the number of neighbors.

KNN is an algorithm particularly successful in small and simple datasets. However, its performance may decrease in cases of large datasets or high-dimensional feature spaces.

As stated, the KNN model provides successful results in small datasets and can be considered as a suitable model for the dataset used in the project. Like SVM and XGBoost models, the K parameter was selected using cross-validation during model usage. Grid Search algorithm, as in other models, was employed, and the algorithm output identified 'K': 5 as the optimal value. Using this value, the accuracy rate in predictions was found to be 91.3%. The table regarding the errors of the model can be found below.

	Actual Value Not Legendary	***Actual Value Legendary***
Predicted Value Not Legendary	141	0
Predicted Value Legendary	14	6

Table 7 Confusion Matrix

ACCURACY COMPARISON TABLE OF MODELS

Models	***Accuracy Value***	***Misclassified Values***
Decision Tree	98.14%	3
Gaussian Naïve Bayes	59.01%	66
Logistic Regression	96.89%	5
Support Vector Machine	96.89%	5
XGBoost	98.76%	2
K-Nearest Neighbors	91.30%	14

IV. CONCLUSION

In conclusion, our project aimed to leverage machine learning algorithms for the classification and prediction of legendary Pokémon using a comprehensive dataset obtained from Kaggle.com. We applied various classification algorithms, including Decision Tree, Gaussian Naive Bayes, Logistic Regression, Support Vector Machine (SVM), XGBoost, and K-Nearest Neighbors (KNN). Each algorithm underwent a thorough exploration, preprocessing, and model optimization process.

Dataset Exploration and Preprocessing:

- The dataset, consisting of 801 Pokémon characters with 41 variables, was explored and pre-processed.
- Missing data points were addressed, and feature engineering was performed, including one-hot encoding for Pokémon types.
- Descriptive statistics and correlation matrices were employed for a better understanding of the dataset.

Overall Reflection:

- Each algorithm demonstrated its strengths and weaknesses based on the dataset characteristics.
- The Decision Tree and XGBoost models achieved the highest accuracies, demonstrating their suitability for this classification task.
- Gaussian Naive Bayes had lower accuracy, possibly due to the dataset's deviation from its independence assumptions.
- SVM and Logistic Regression provided robust results but with slight misclassifications.

Recommendations:

- The selection of the appropriate algorithm depends on the dataset characteristics and problem requirements.
- Feature engineering and preprocessing significantly impact model performance.
- Cross-validation and hyperparameter tuning enhance model accuracy.

Our project is a study that showcases how machine learning methods yield results for researchers dealing with smaller datasets, highlighting the techniques employed in various models. It serves as an engaging exploration of what can be accomplished with this fun dataset..

REFERENCES

[1] Banik, Rounak. "The Complete Pokemon Dataset". Kaggle. Accessed on November 29, 2023, https://www.kaggle.com/datasets/rounakbanik/pokemon

[2] C. M. Bishop, "Pattern Recognition and Machine Learning". Springer, 2006.

[3] A. Geron, "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow", O'Reilly Media, 2019.

[4] Türkoğlu, M.Fatih. "Support Vector Machine (SVM) Algoritması-Makine Öğrenmesi". Medium. Accessed on December 15, 2023, https://mfatihto.medium.com/support-vector-machine-algoritmas%C4%B1-makine-%C3%B6%C4%9Frenmesi-8020176898d8

[5] Güler, Efecan. "XGBoost Algoritması". Medium. Accessed on December 20, 2023, https://efecanxrd.medium.com/xgboost-algoritmas%C4%B1-6703b14efd5