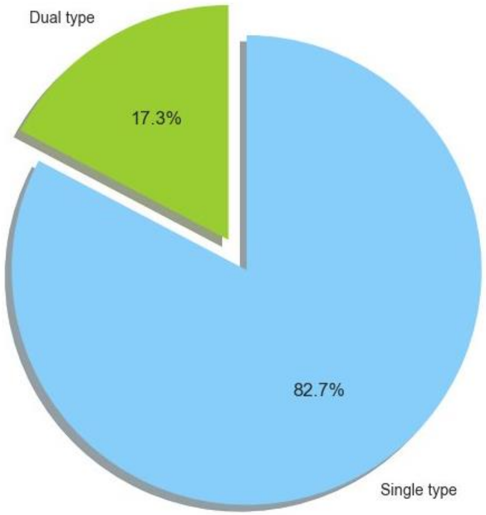
MACHINE LEARNING AND DATA ANALYSIS

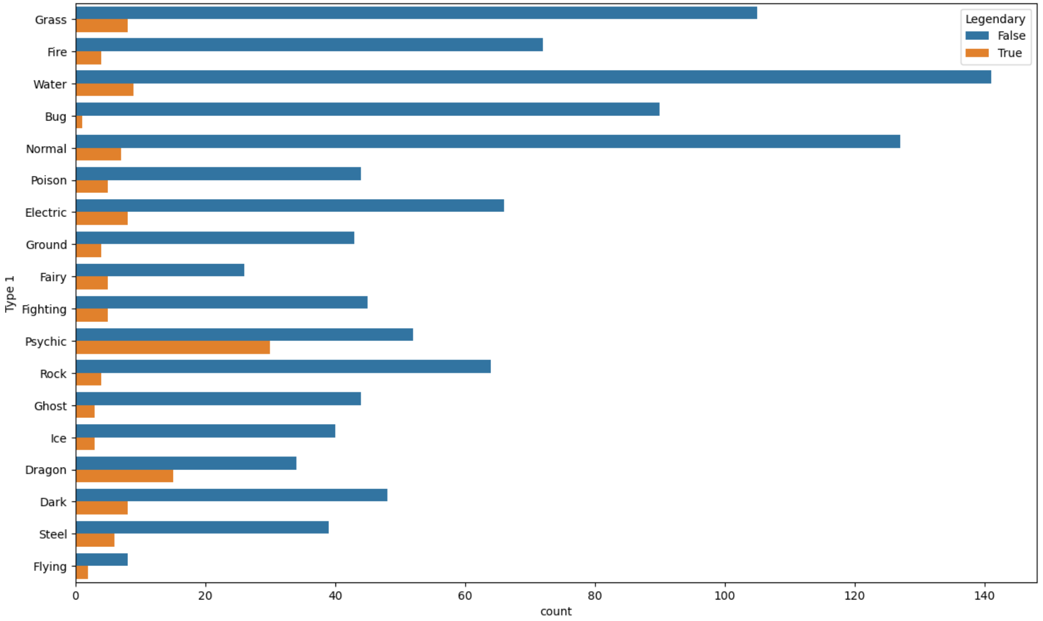
MAGISTRELLO CAMILLA (4512554), ENRICO PEZZANO (4825087)

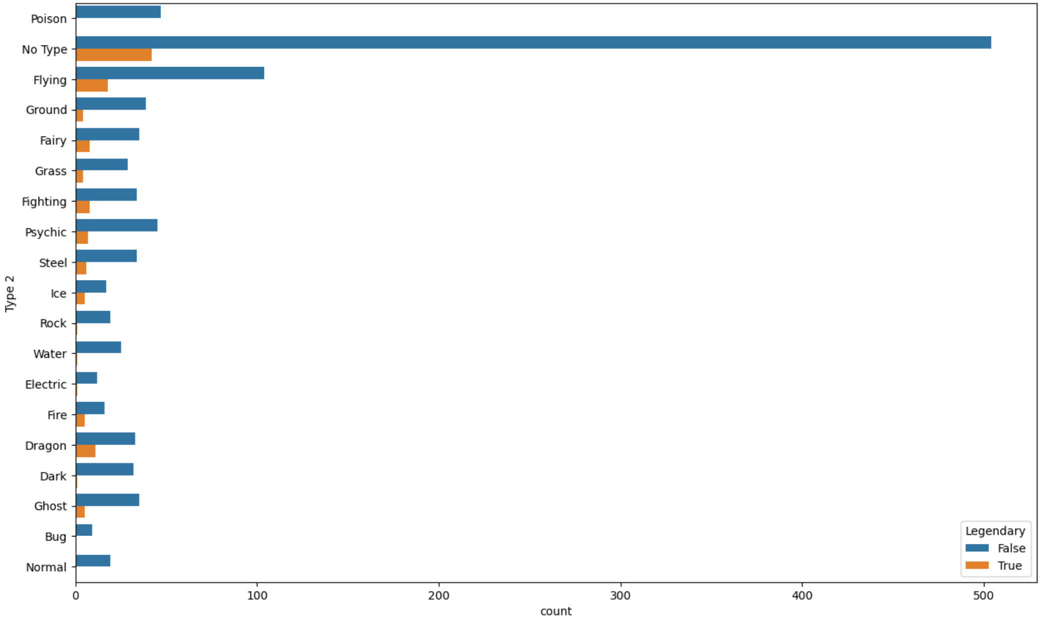
THE DATASET

CSV file with 1200+ lines and 16 columns

Number
Name
Form
Type 1
Type 2
Total
HP
Attack
Defense
Sp. Atk
Sp. Def
Speed
Generation
Legendary
Evolution Number
Final Form

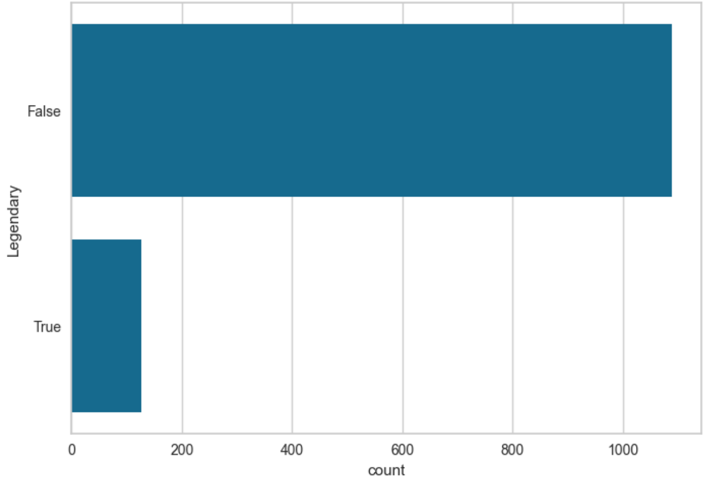


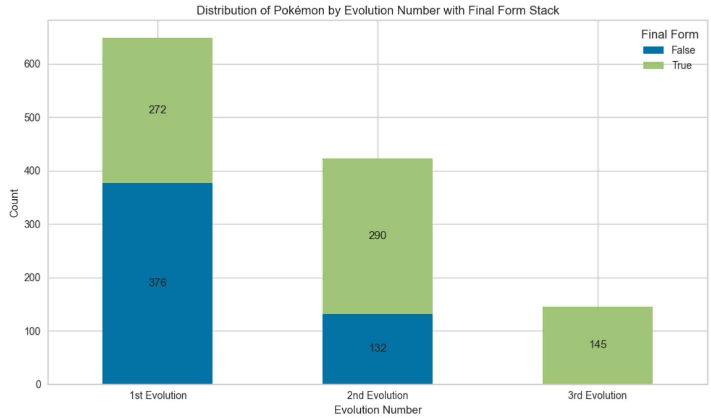


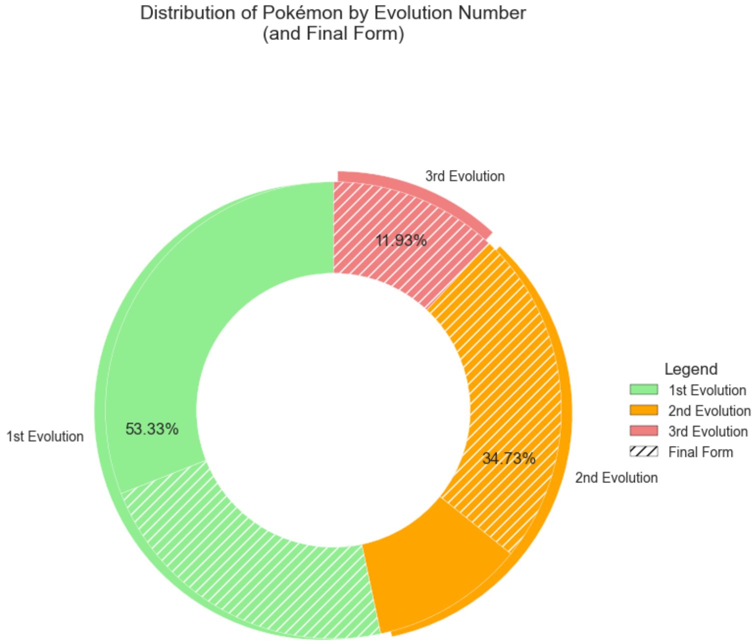


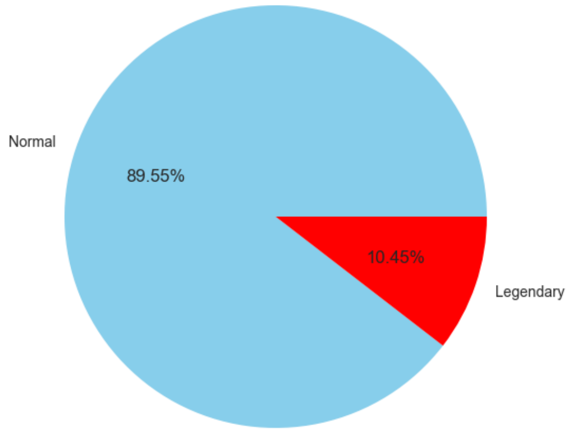
HOW MANY POKEMON?

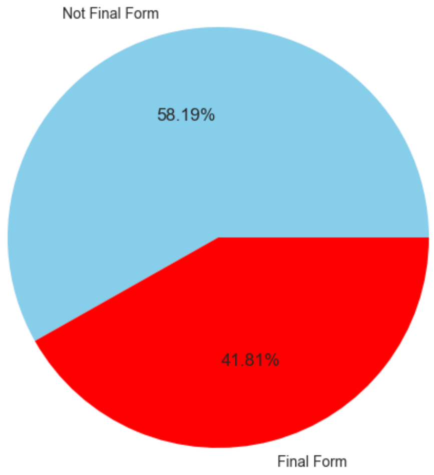
- **1088** legendary
- **793** 1st or 3rd evolution
- **127** non-legendary
- **422** 2nd evolution



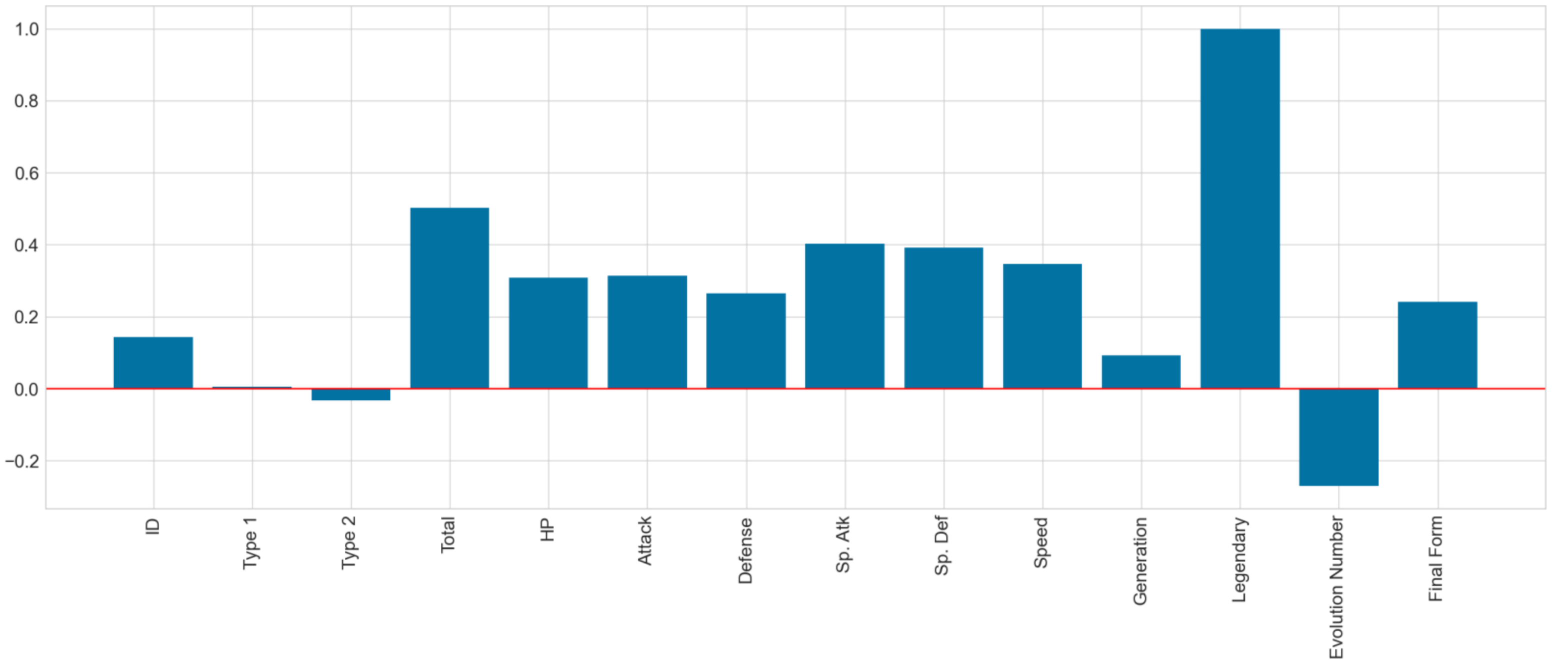


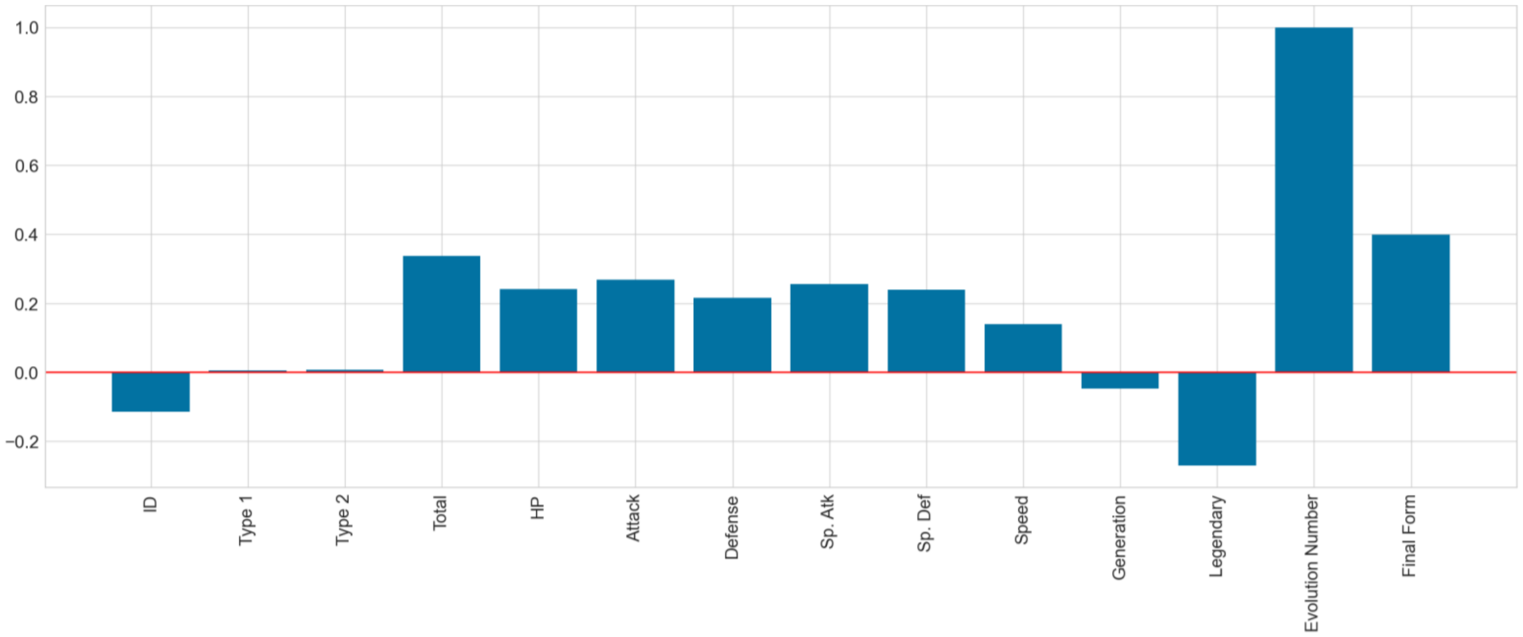






CORRELATIONS





PROBLEMS SELECTION

Based on the features, determine whether a Pokémon... :

- ...is **legendary** or not
- ...is in its **second evolution**

GridSearchCsv

A fundamental tool in optimizing **hyper-parameters** of machine learning models.

It involves defining a set of values for each **hyper-parameter** of a model and testing all possible combinations of these values.

✓

- Simplicity and clarity
- Comprehensive exploration
- Systematic evaluation

✗

- Computational cost
- Complex models
- Choice of values

Feature Importance

They indicate how much each feature contributes to the result of a machine learning model (mainly used in models like Decision Trees and Random Forest).

KNeighborsClassifier is based on distances between points and not on coefficients or feature importance, so they cannot be calculated.

DATASET PREPARATION

Transforming the values in columns "Type1" and "Type2" from strings to numeric representations corresponding to the type.

RandomOverSampler to over-sample the legendary class by picking samples at random with replacement.

Based on the features, determine whether a Pokémon is **Legendary** or not

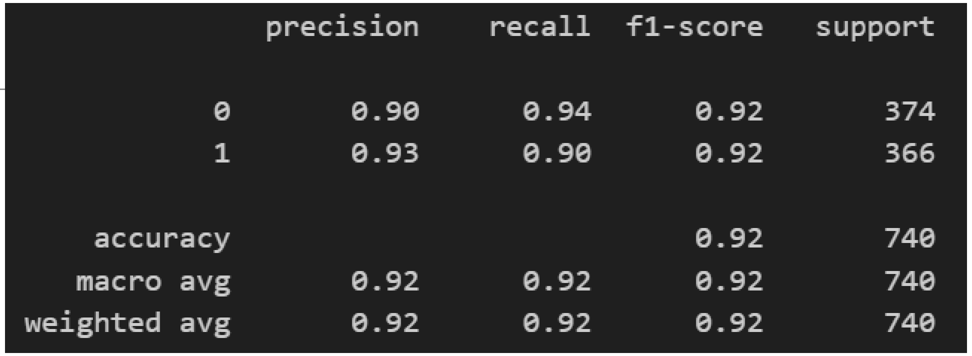
DATASET PREPARATION

Drop the columns:

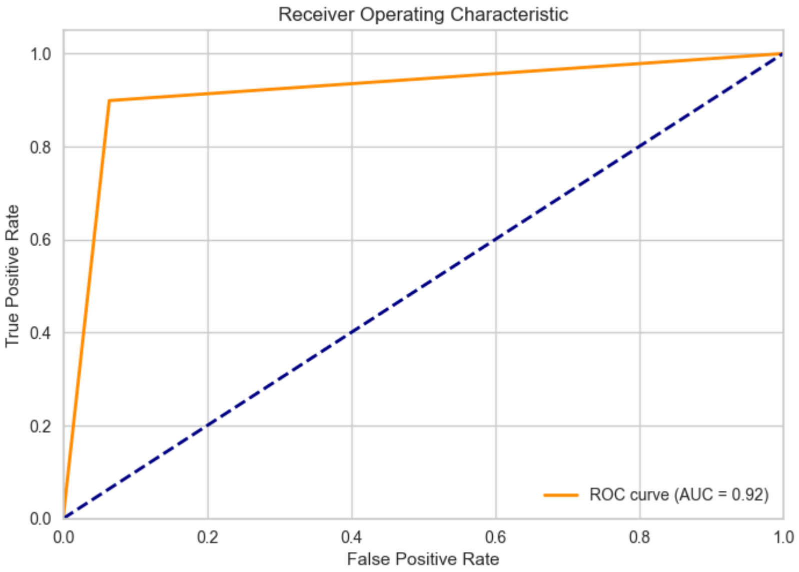
- ❑ ID
- ❑ Name
- ❑ Form
- ❑ **Legendary**

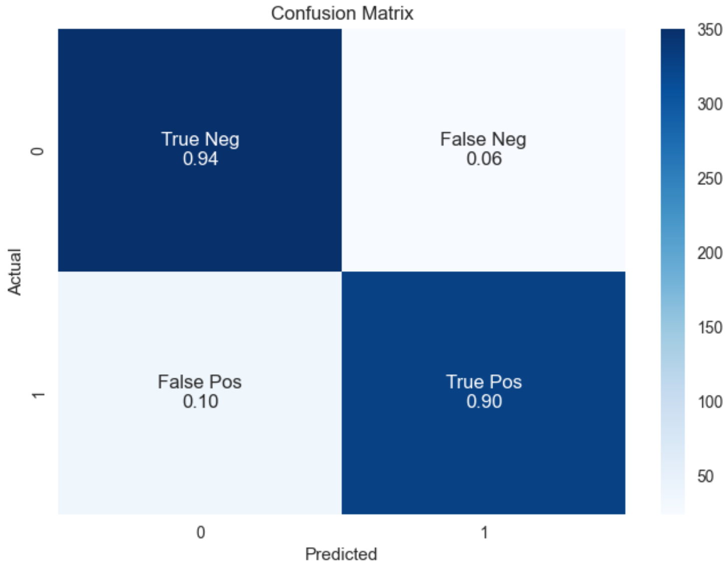
DECISION TREE CLASSIFIER **WITHOUT** HYPERPARAMETERS

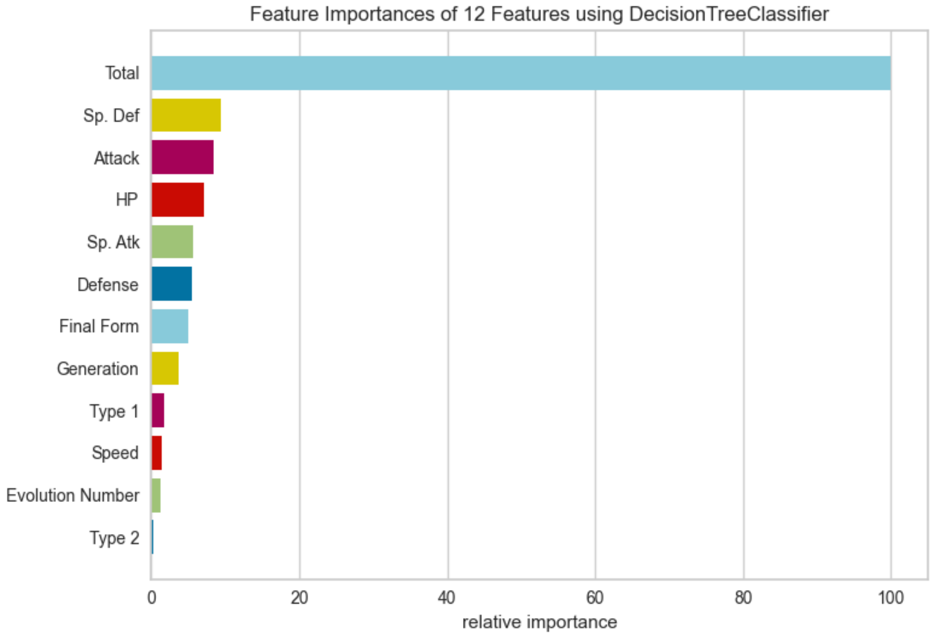
Testing accuracy: 0.9175675675675675
Training accuracy: 1.0



```
              precision    recall  f1-score   support

           0       0.90      0.94      0.92       374
           1       0.93      0.90      0.92       366

    accuracy                           0.92       740
   macro avg       0.92      0.92      0.92       740
weighted avg       0.92      0.92      0.92       740
```



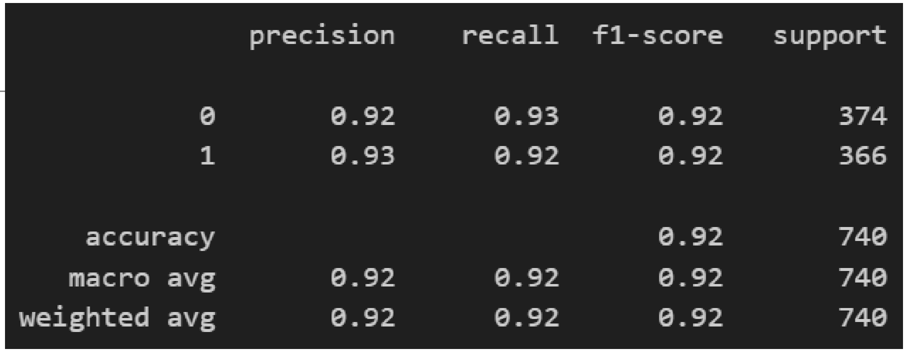




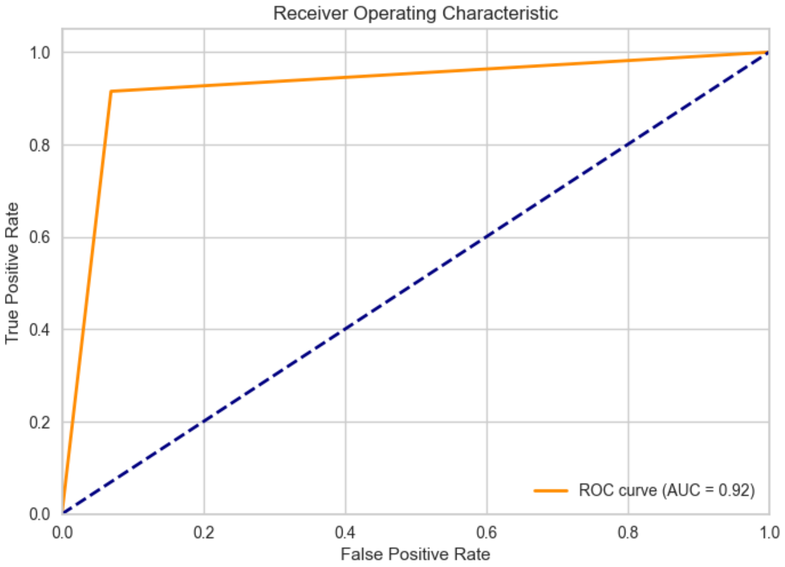
DECISION TREE CLASSIFIER WITH HYPERPARAMETERS

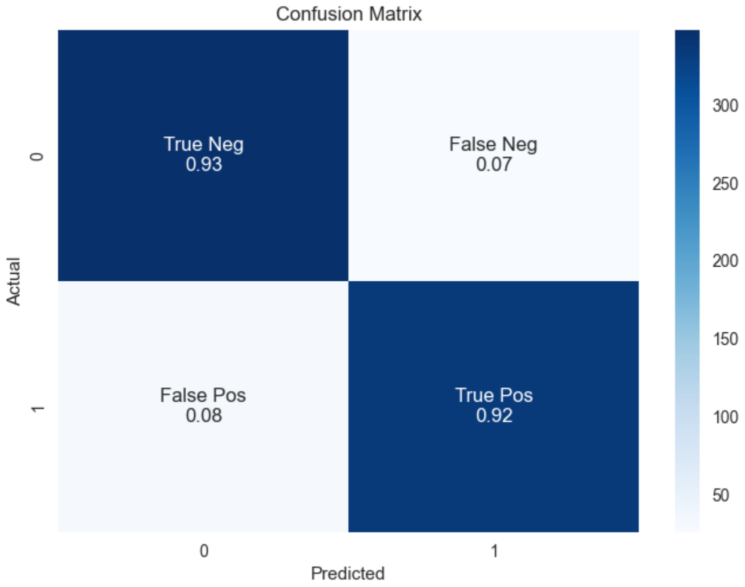
'criterion': 'entropy'
'max_depth': None
'min_samples_leaf': 1
'min_samples_split': 2

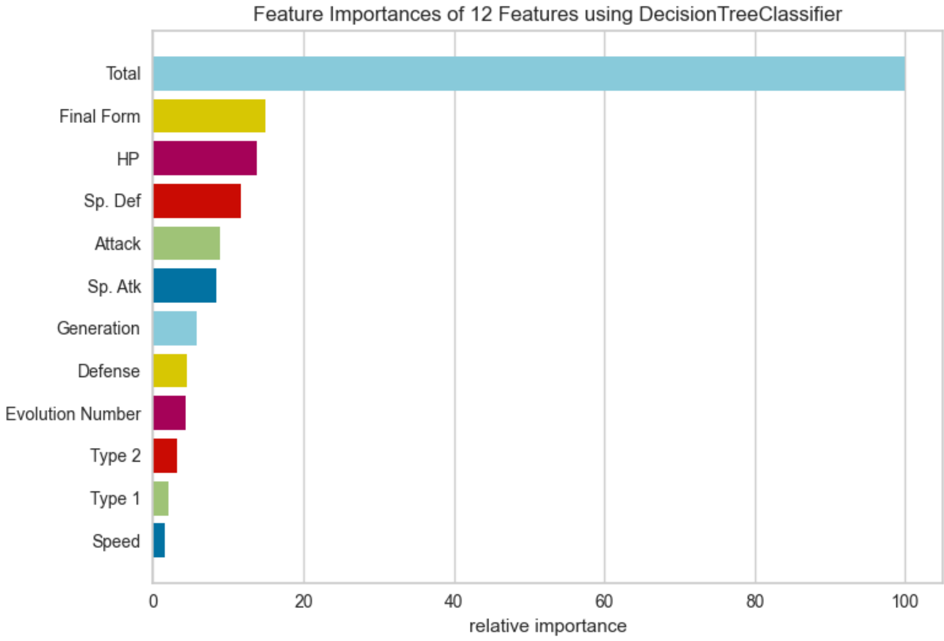
Testing accuracy: 0.9229729729729729730
Training accuracy: 1.0



```
              precision    recall  f1-score   support

           0       0.92      0.93      0.92       374
           1       0.93      0.92      0.92       366

    accuracy                           0.92       740
   macro avg       0.92      0.92      0.92       740
weighted avg       0.92      0.92      0.92       740
```

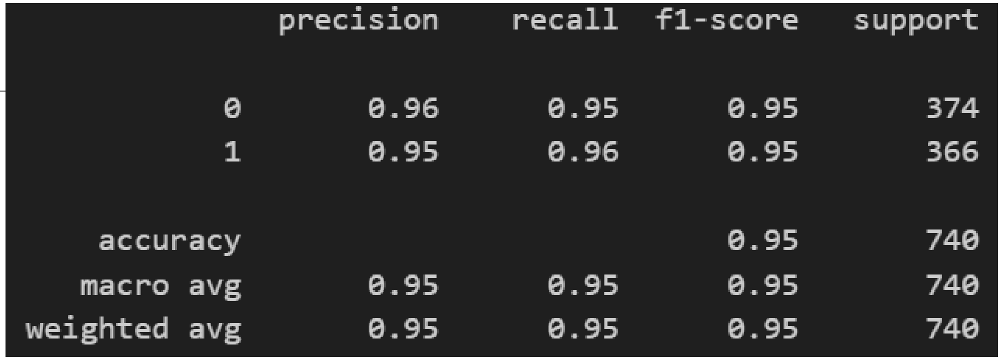




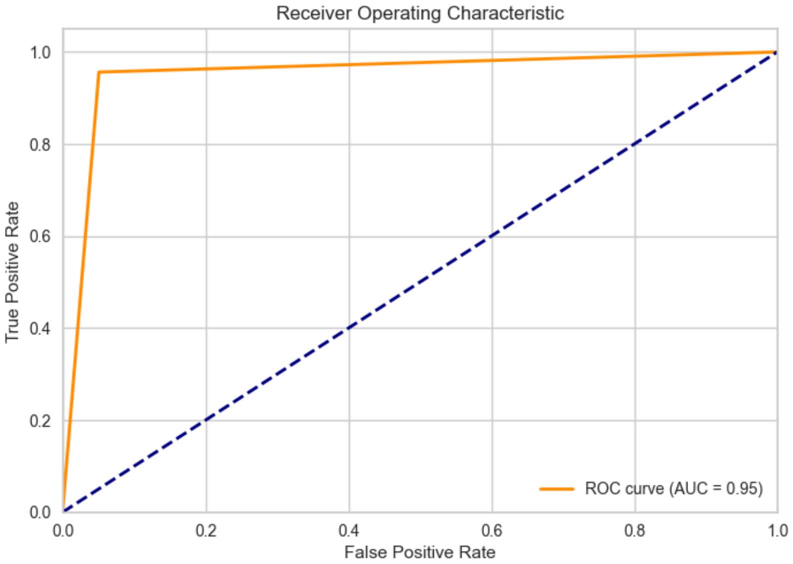


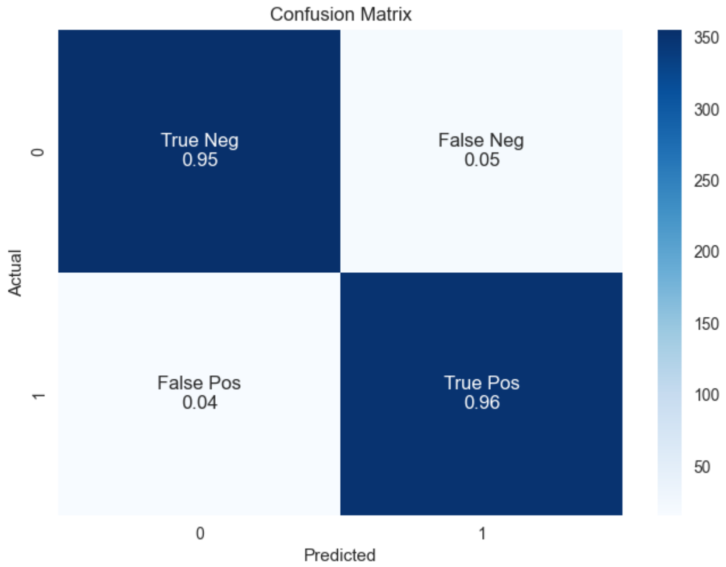
RANDOM FOREST CLASSIFIER **WITHOUT** HYPERPARAMETERS

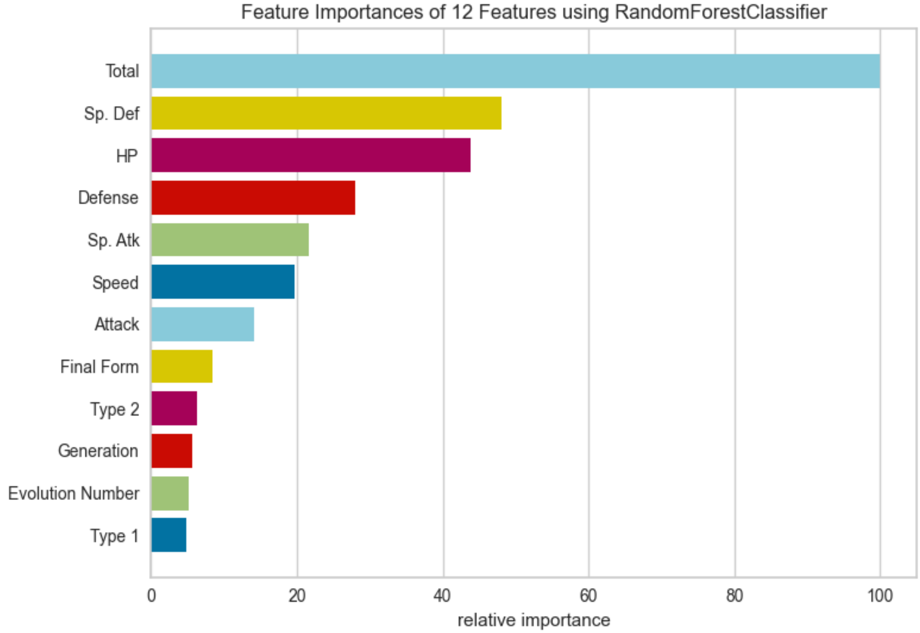
Testing accuracy: 0.9527027027027027
Training accuracy: 1.0



```
              precision    recall  f1-score   support

           0       0.96      0.95      0.95       374
           1       0.95      0.96      0.95       366

    accuracy                           0.95       740
   macro avg       0.95      0.95      0.95       740
weighted avg       0.95      0.95      0.95       740
```

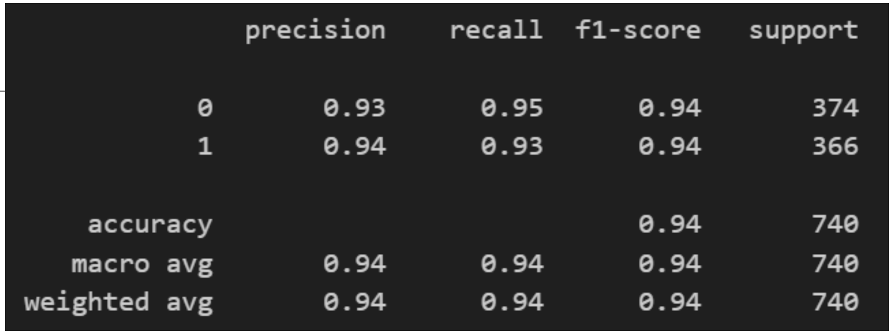






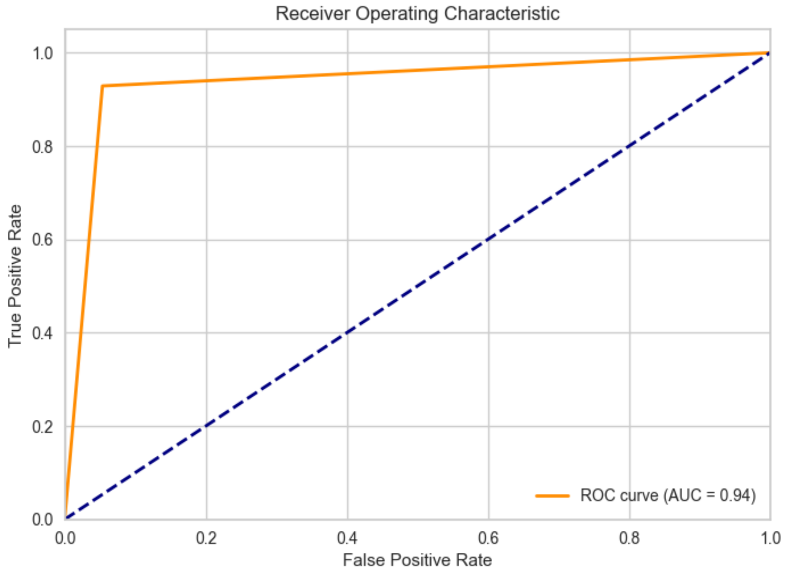
RANDOM FOREST CLASSIFIER WITH HYPERPARAMETERS

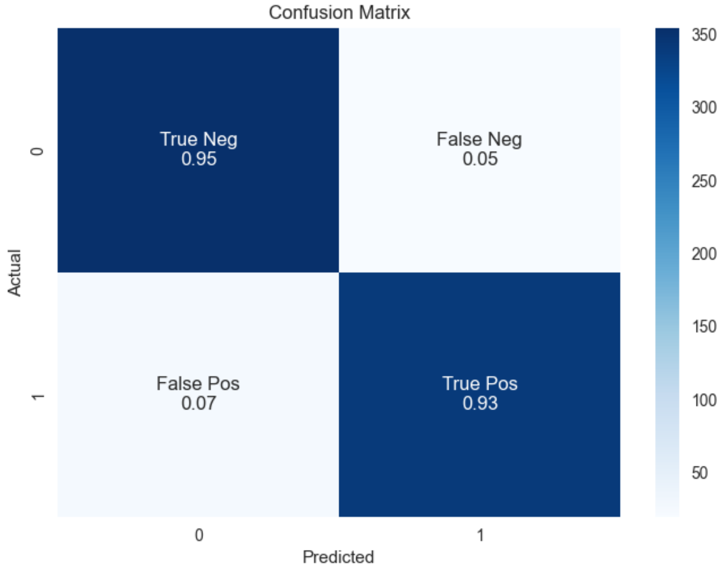
Testing accuracy: 0.9378378378378378
Training accuracy: 1.0

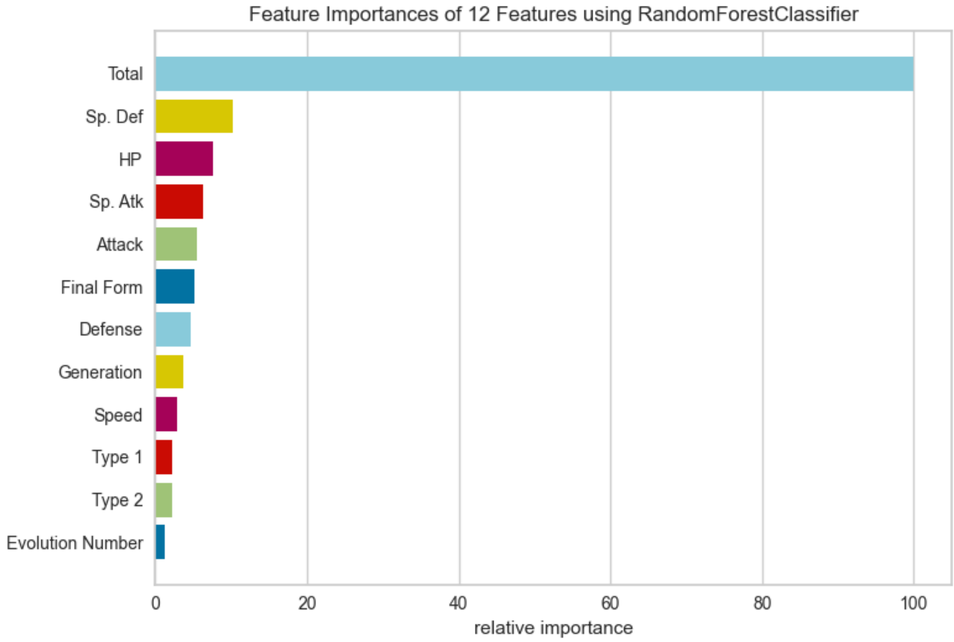


```
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       374
           1       0.94      0.93      0.94       366

    accuracy                           0.94       740
   macro avg       0.94      0.94      0.94       740
weighted avg       0.94      0.94      0.94       740
```

'max_depth’: None
'max_features’: None
'min_samples_leaf’: 1
'min_samples_split’: 2
'n_estimators’: 70

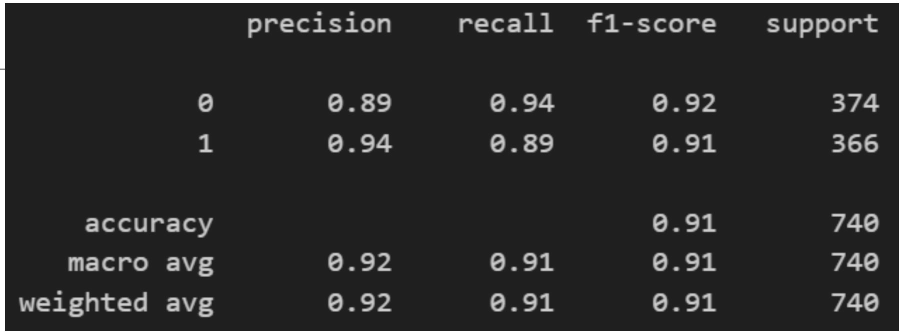




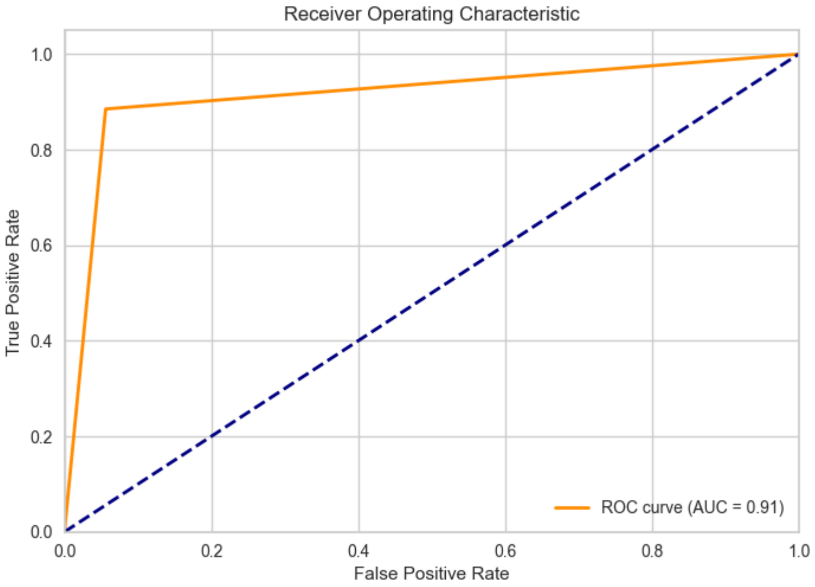


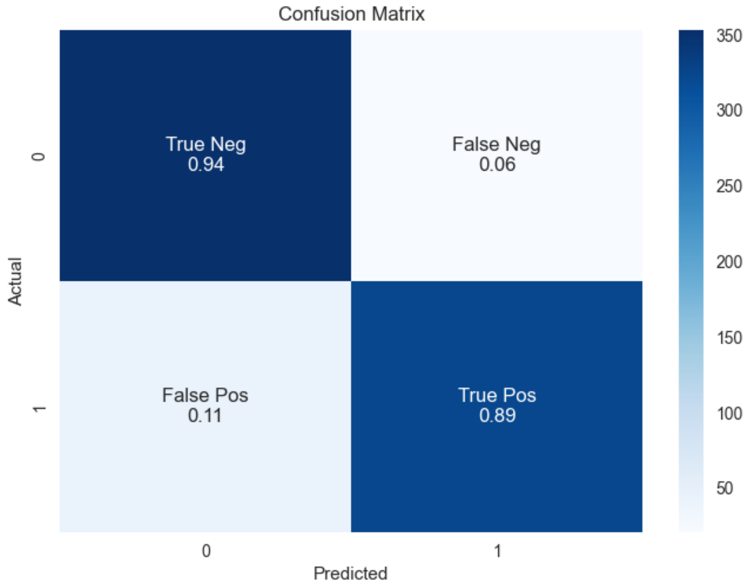
LOGISTIC REGRESSION **WITHOUT** HYPERPARAMETERS

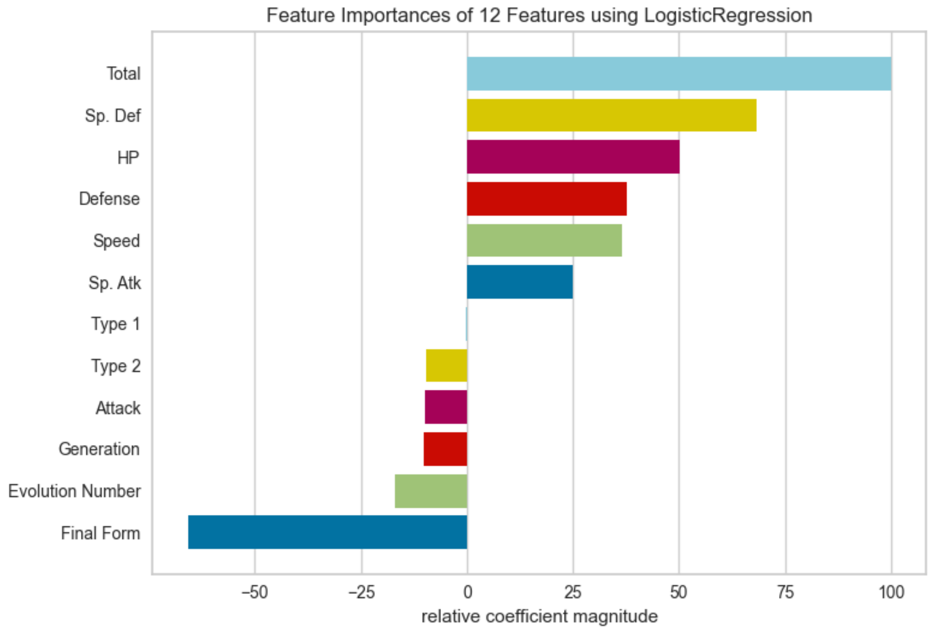
Testing accuracy: 0.9148648648648648
Training accuracy: 0.915041782729805



```
              precision    recall  f1-score   support

           0       0.89      0.94      0.92       374
           1       0.94      0.89      0.91       366

    accuracy                           0.91       740
   macro avg       0.92      0.91      0.91       740
weighted avg       0.92      0.91      0.91       740
```



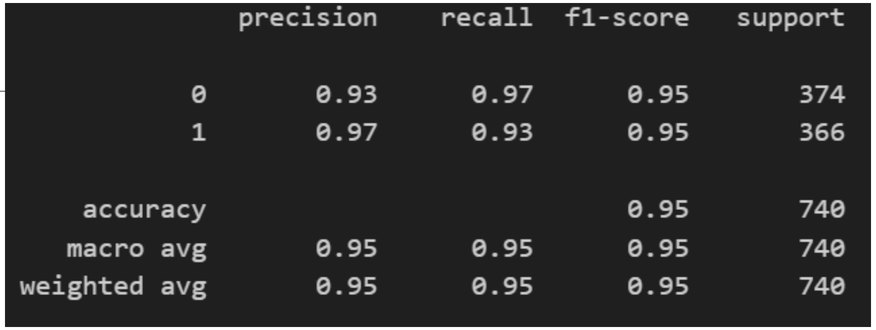




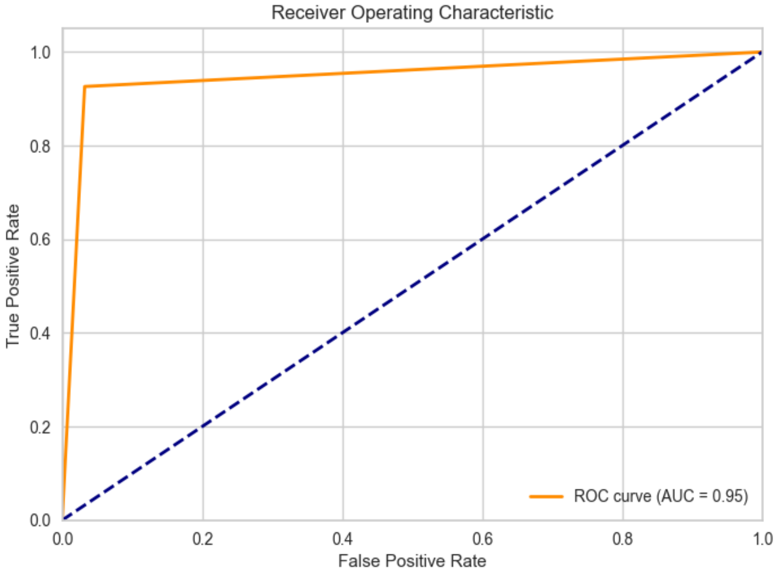
LOGISTIC REGRESSION WITH HYPERPARAMETERS

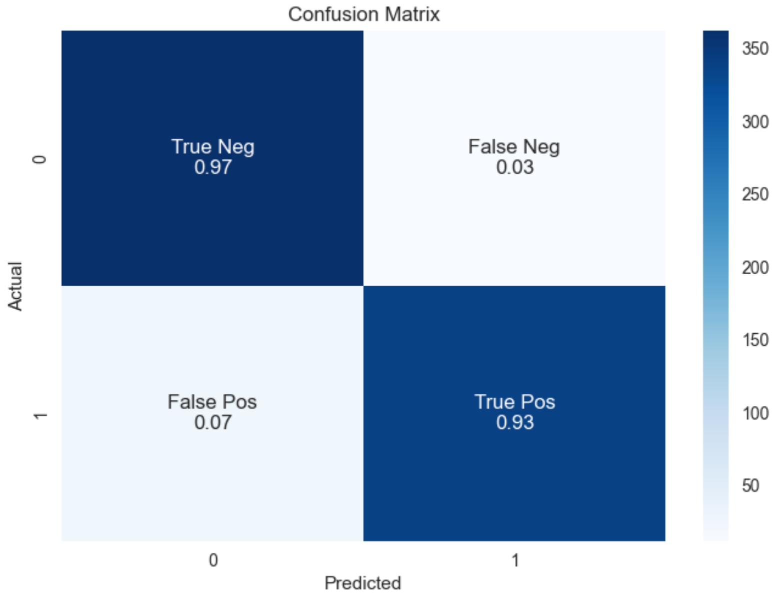
'C': 10
'penalty': 'l1'
'solver': 'liblinear'

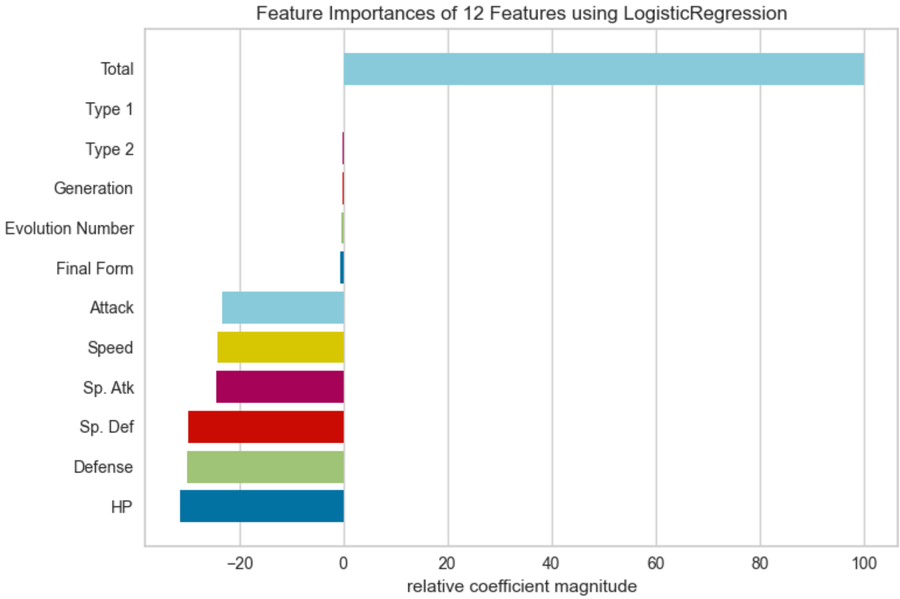
Testing accuracy: 0.9472972972972973
Training accuracy: 0.9533426183844012



```
              precision    recall  f1-score   support

           0       0.93      0.97      0.95       374
           1       0.97      0.93      0.95       366

    accuracy                           0.95       740
   macro avg       0.95      0.95      0.95       740
weighted avg       0.95      0.95      0.95       740
```

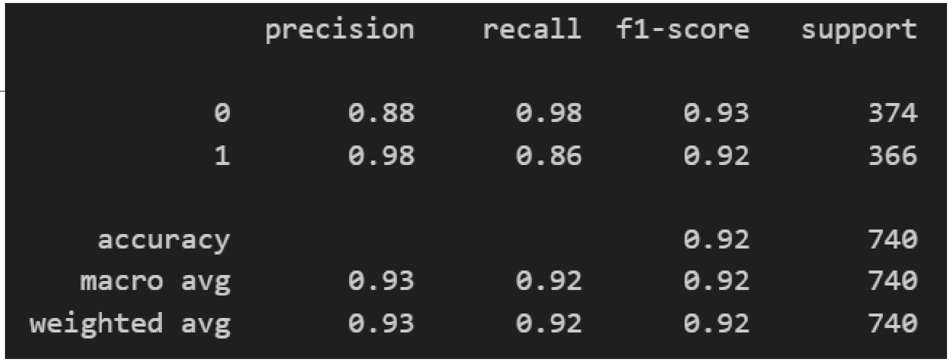




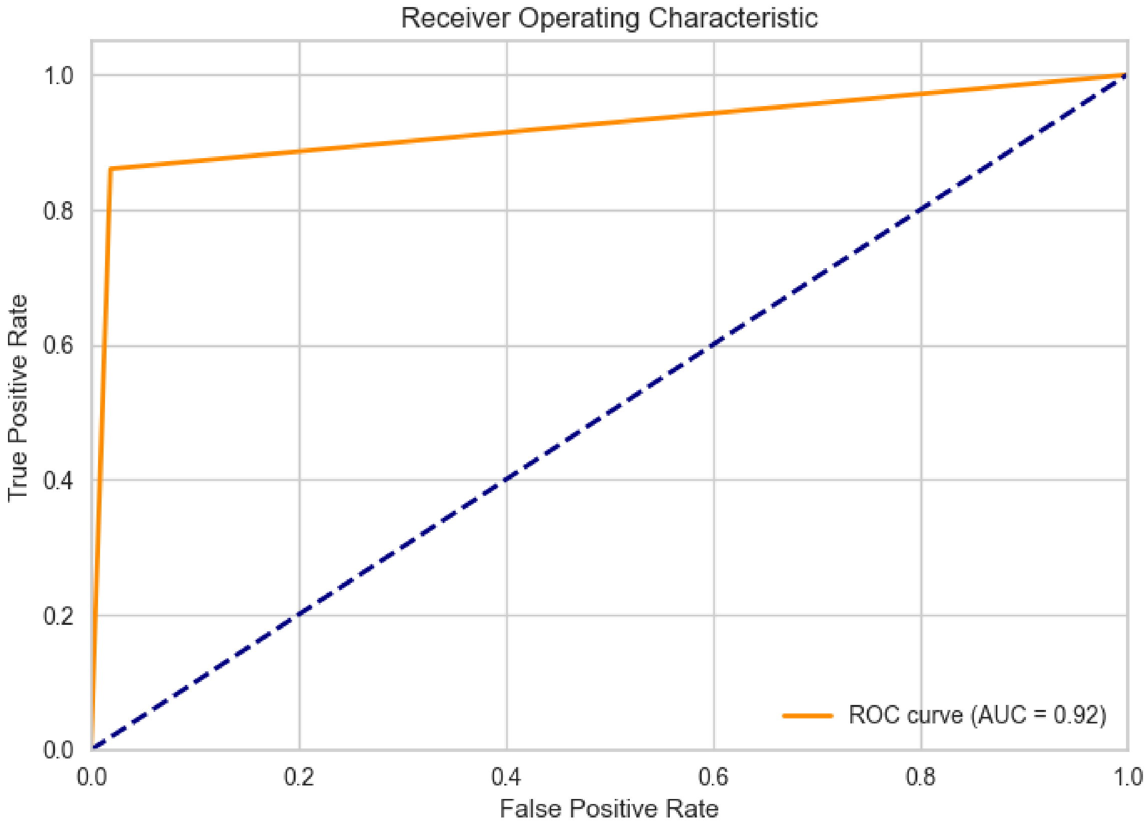


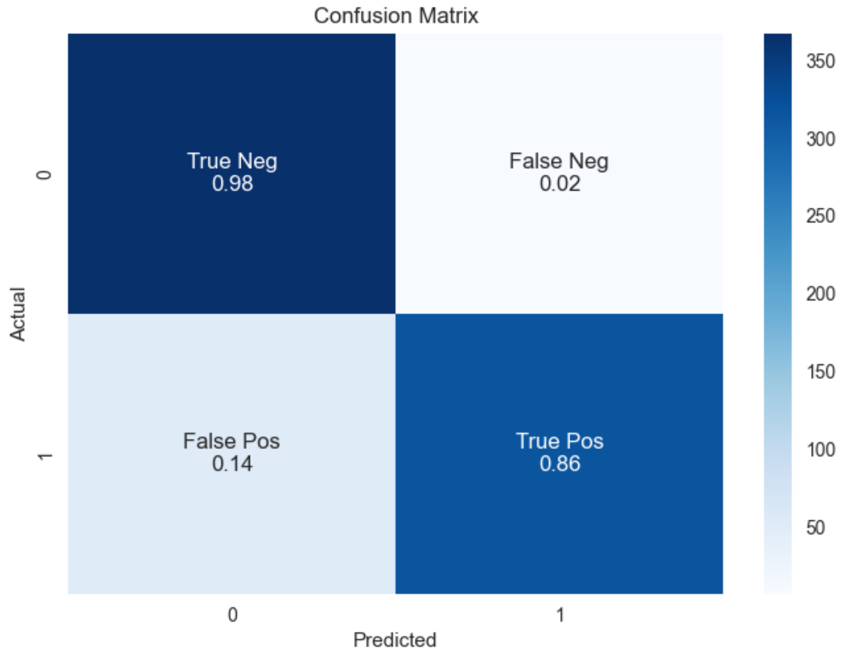
K-NEAREST NEIGHBORS **WITHOUT** HYPERPARAMETERS

Testing accuracy: 0.9216216216216216
Training accuracy: 0.9352367688022284



```
              precision    recall  f1-score   support

           0       0.88      0.98      0.93       374
           1       0.98      0.86      0.92       366

    accuracy                           0.92       740
   macro avg       0.93      0.92      0.92       740
weighted avg       0.93      0.92      0.92       740
```

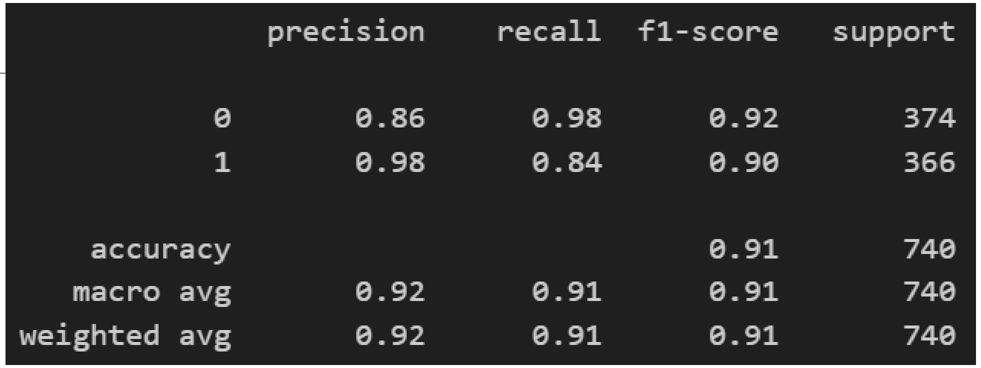




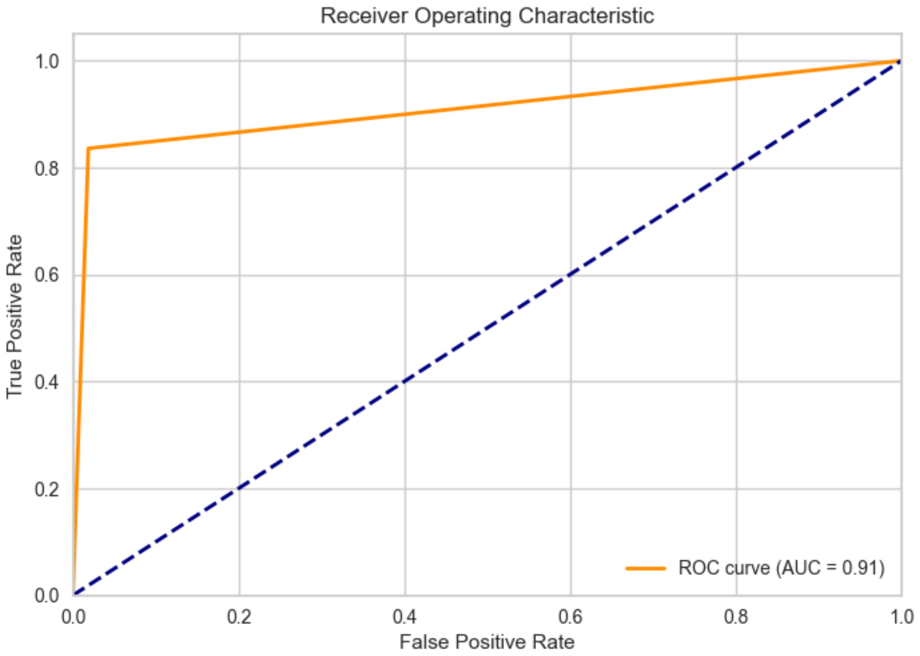
K-NEAREST NEIGHBORS WITH HYPERPARAMETERS

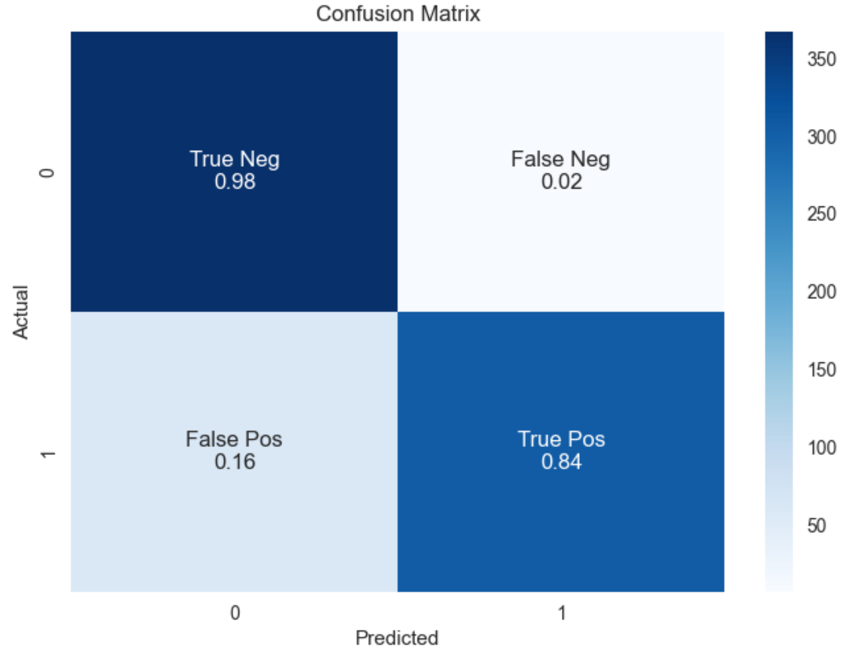
‘metric’: ‘manhattan’,
‘n_neighbors’: 5,
‘weights’: ‘distance’

Testing accuracy: 0.9094594594594595
Training accuracy: 1.0

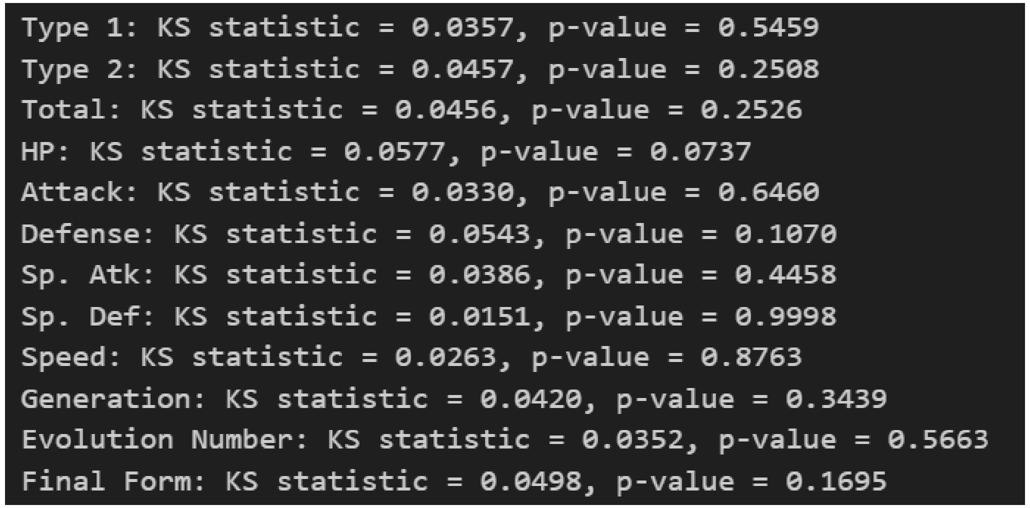


```
              precision    recall  f1-score   support

           0       0.86      0.98      0.92       374
           1       0.98      0.84      0.90       366

    accuracy                           0.91       740
   macro avg       0.92      0.91      0.91       740
weighted avg       0.92      0.91      0.91       740
```

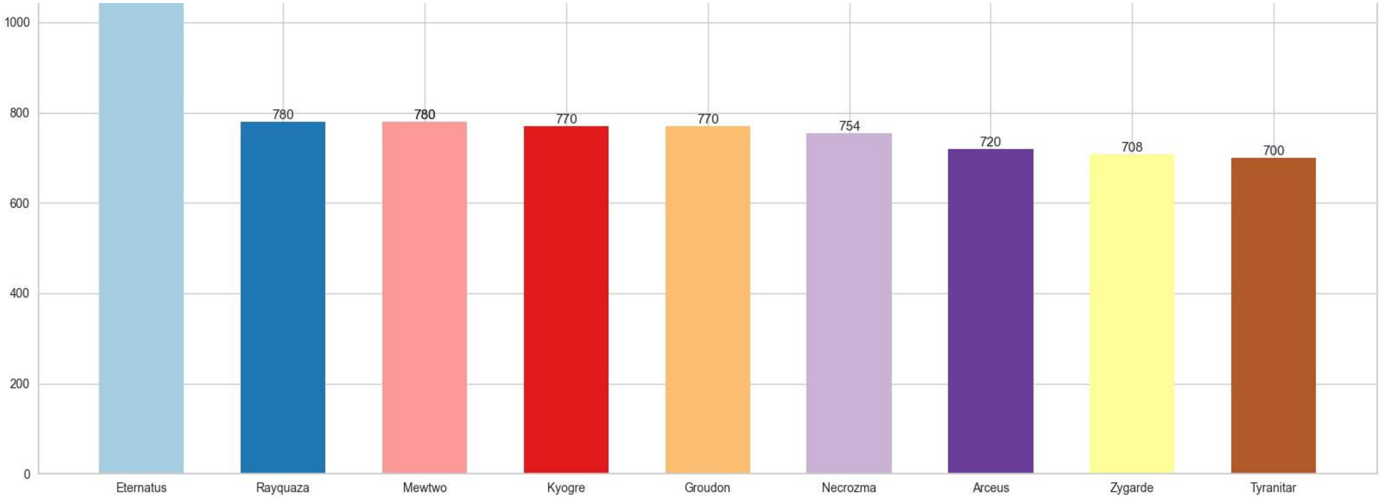




Checking the results...



```
Type 1: KS statistic = 0.0357, p-value = 0.5459
Type 2: KS statistic = 0.0457, p-value = 0.2508
Total: KS statistic = 0.0456, p-value = 0.2526
HP: KS statistic = 0.0577, p-value = 0.0737
Attack: KS statistic = 0.0330, p-value = 0.6460
Defense: KS statistic = 0.0543, p-value = 0.1070
Sp. Atk: KS statistic = 0.0386, p-value = 0.4458
Sp. Def: KS statistic = 0.0151, p-value = 0.9998
Speed: KS statistic = 0.0263, p-value = 0.8763
Generation: KS statistic = 0.0420, p-value = 0.3439
Evolution Number: KS statistic = 0.0352, p-value = 0.5663
Final Form: KS statistic = 0.0498, p-value = 0.1695
```



- Kolmogorov–Smirnov Test:
  - **p-value** < 0.05 implies a statistically significant difference.
  - It suggests that the distributions of the attributes are very similar between the two groups, except for the Performance Status
- Legendaries totals stats too high, as we suspected
- Hence why ...

Based on the features, determine whether a Pokémon is in its **second evolution**

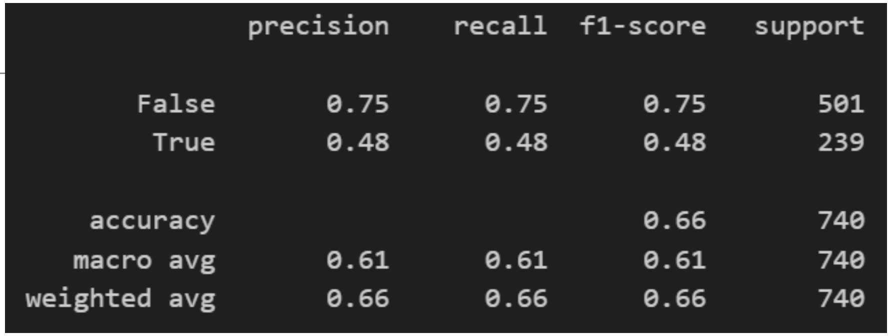
DATASET PREPARATION

Drop the columns:

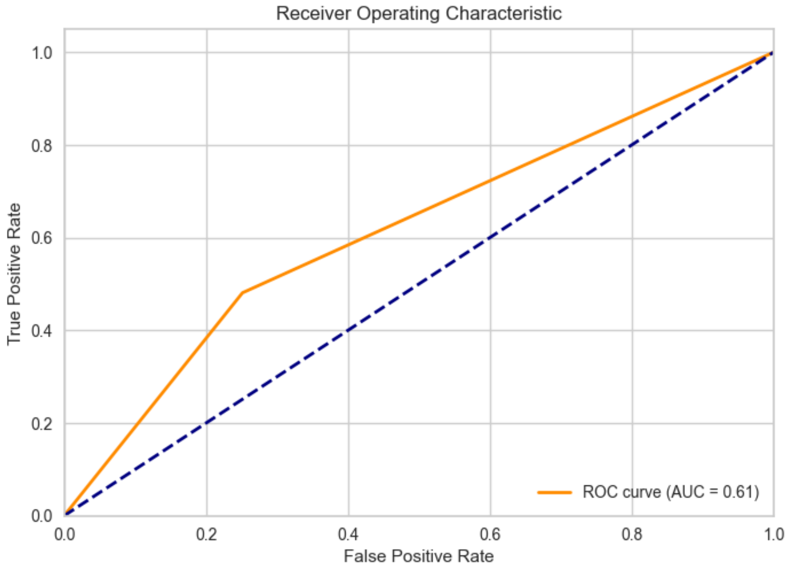
- ❑ ID
- ❑ Name
- ❑ Form
- ❑ **Evolution number**

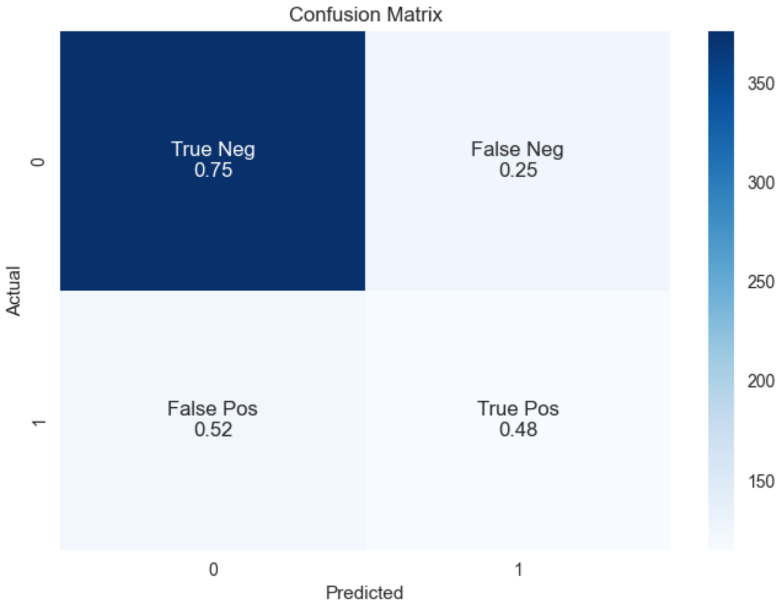
DECISION TREE CLASSIFIER **WITHOUT** HYPERPARAMETERS

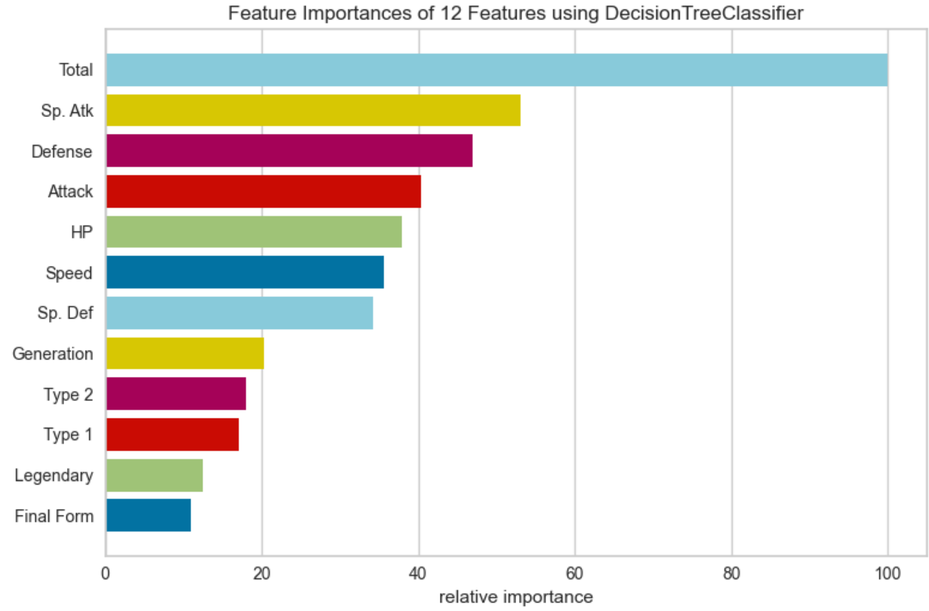
Testing accuracy: 0.6621621621621622
Training accuracy: 0.9993036211699164



```
              precision    recall  f1-score   support

       False       0.75      0.75      0.75       501
        True       0.48      0.48      0.48       239

    accuracy                           0.66       740
   macro avg       0.61      0.61      0.61       740
weighted avg       0.66      0.66      0.66       740
```



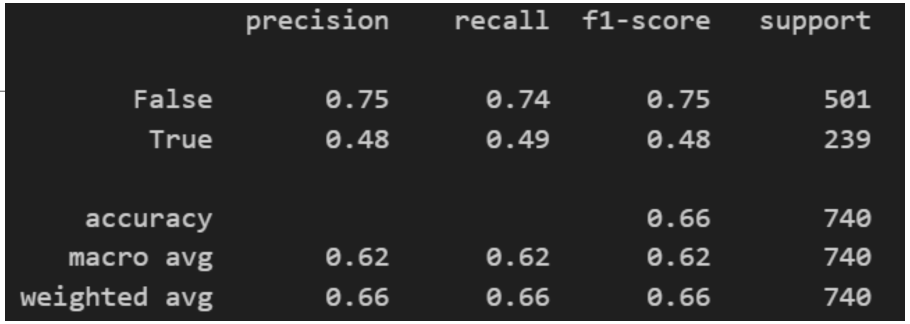




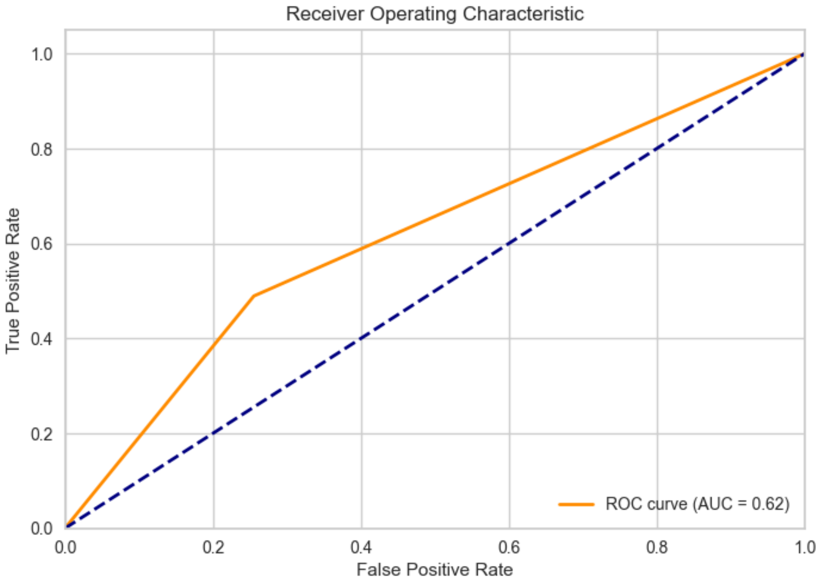
DECISION TREE CLASSIFIER WITH HYPERPARAMETERS

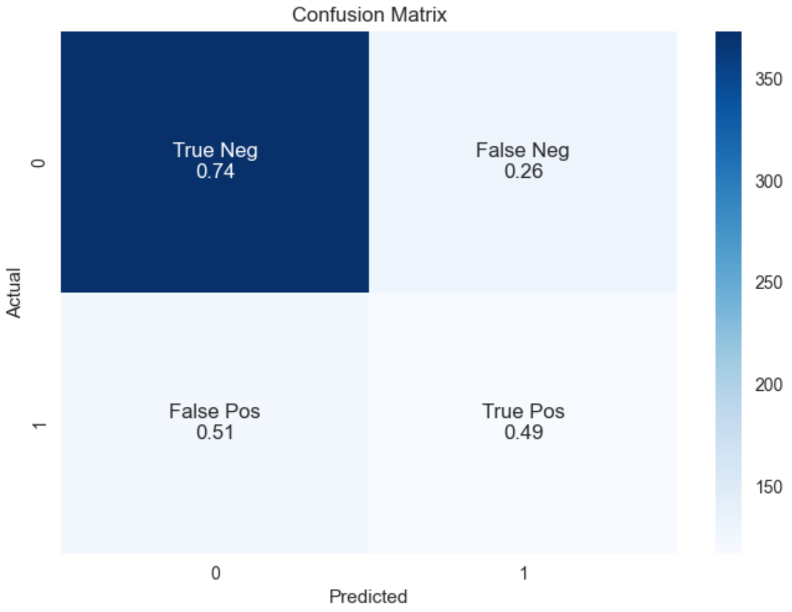
'criterion': ‘log_loss’
'max_depth': 10
'min_samples_leaf': 1
'min_samples_split': 2

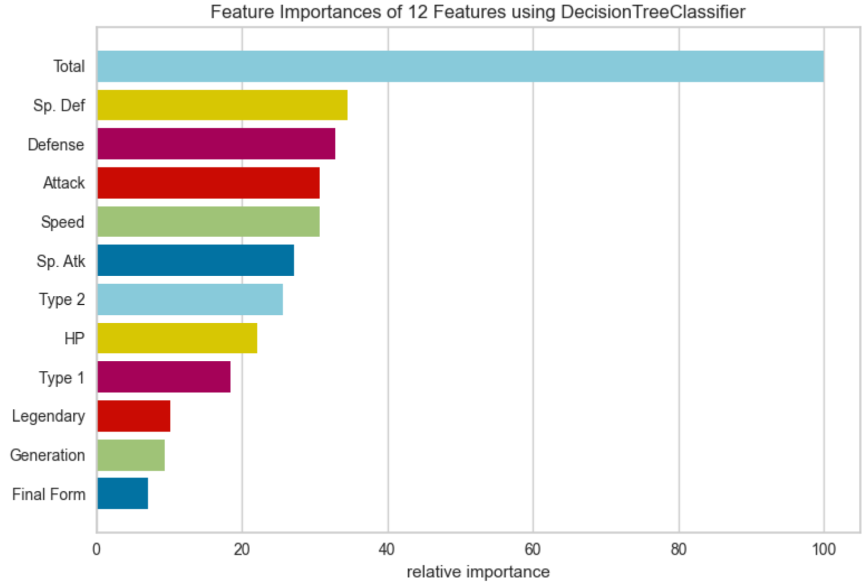
Testing accuracy: 0.6621621621621622
Training accuracy: 0.8760445682451253



```
              precision    recall  f1-score   support

       False       0.75      0.74      0.75       501
        True       0.48      0.49      0.48       239

    accuracy                           0.66       740
   macro avg       0.62      0.62      0.62       740
weighted avg       0.66      0.66      0.66       740
```

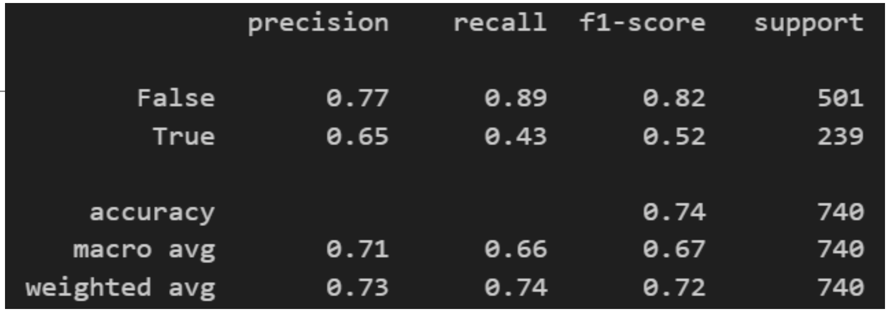




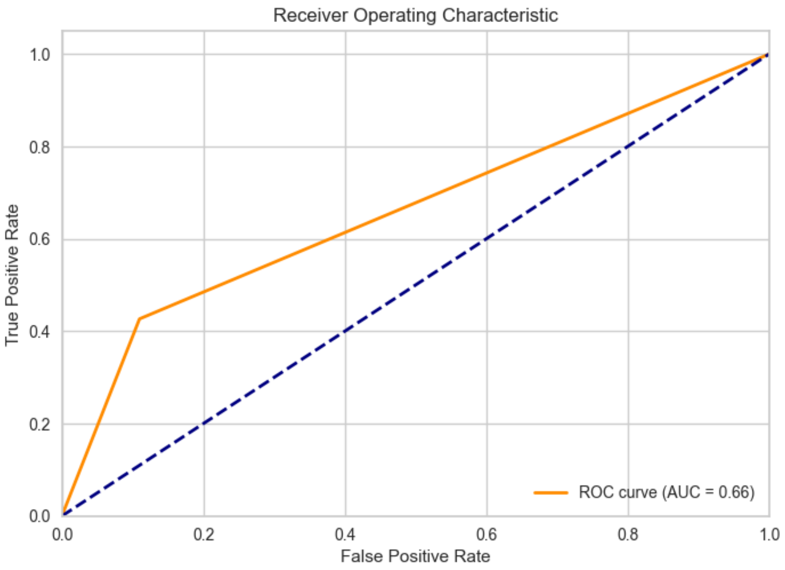


RANDOM FOREST CLASSIFIER **WITHOUT** HYPERPARAMETERS

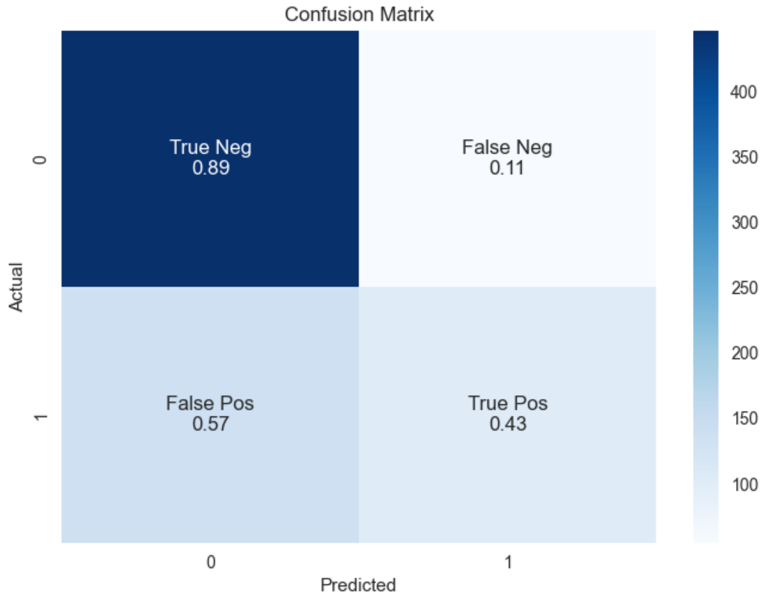
Testing accuracy: 0.7405405405405405
Training accuracy: 0.9993036211699164



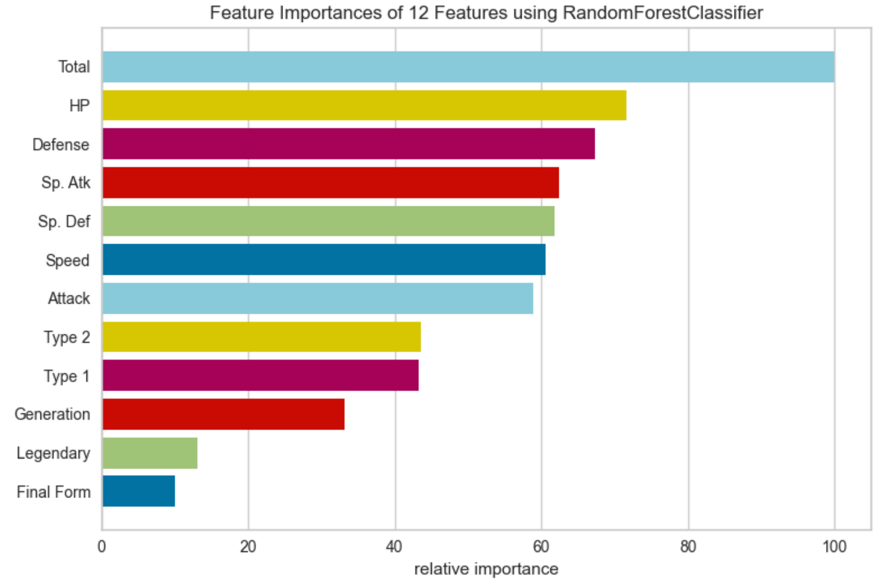
```
              precision    recall  f1-score   support

       False       0.77      0.89      0.82       501
        True       0.65      0.43      0.52       239

    accuracy                           0.74       740
   macro avg       0.71      0.66      0.67       740
weighted avg       0.73      0.74      0.72       740
```



Receiver Operating Characteristic



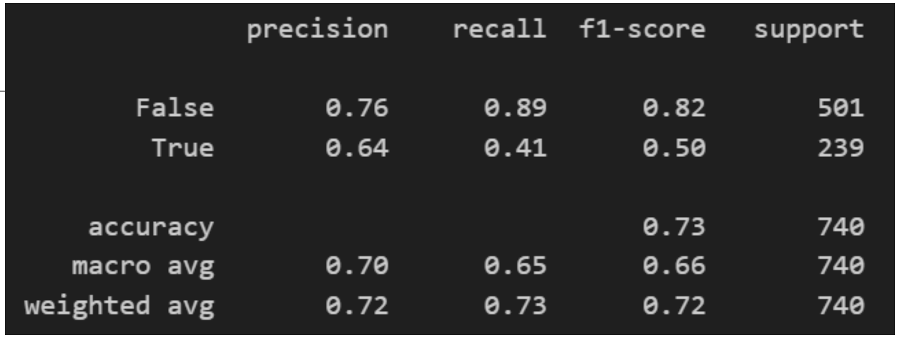
Confusion Matrix



Feature Importances of 12 Features using RandomForestClassifier

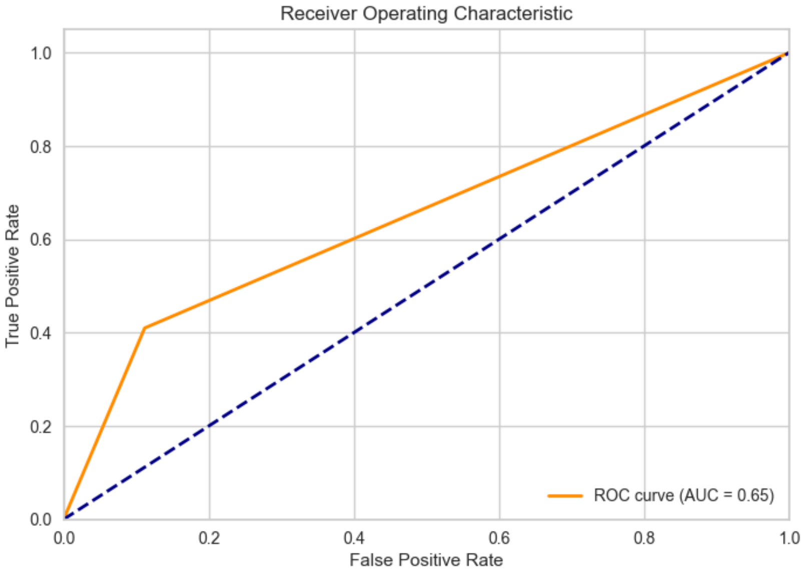
RANDOM FOREST CLASSIFIER WITH HYPERPARAMETERS

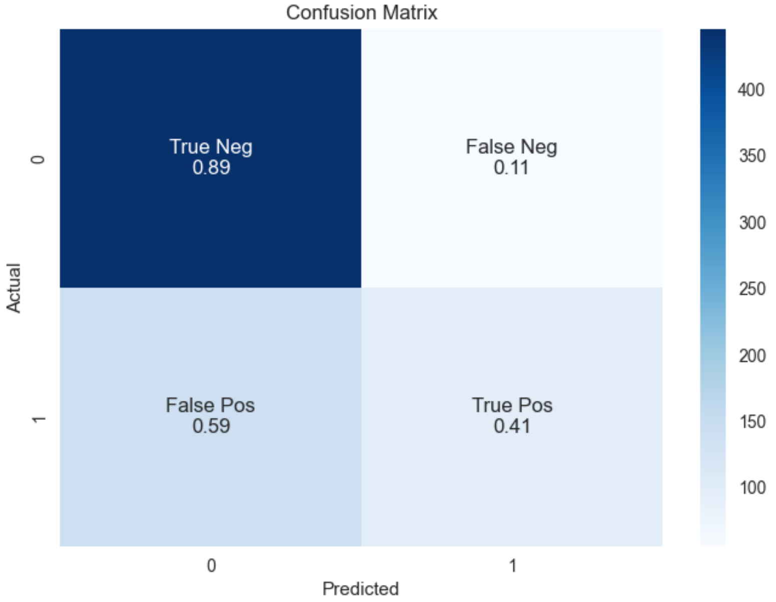
Testing accuracy: 0.7337837837837838
Training accuracy: 0.8600278551532033

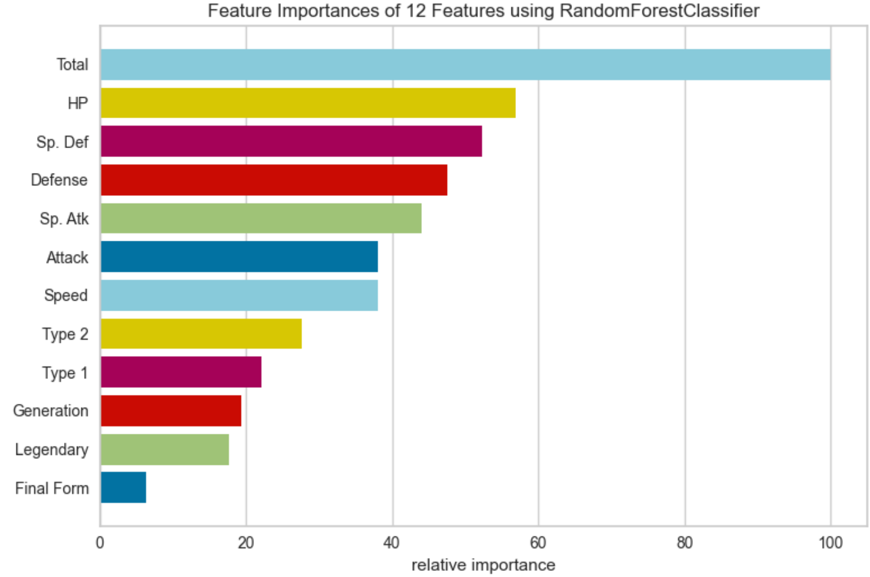


```
              precision    recall  f1-score   support

       False       0.76      0.89      0.82       501
        True       0.64      0.41      0.50       239

    accuracy                           0.73       740
   macro avg       0.70      0.65      0.66       740
weighted avg       0.72      0.73      0.72       740
```

'max_depth': 10
'max_features': ‘sqrt’
'min_samples_leaf': 5
'min_samples_split': 4
'n_estimators': 80

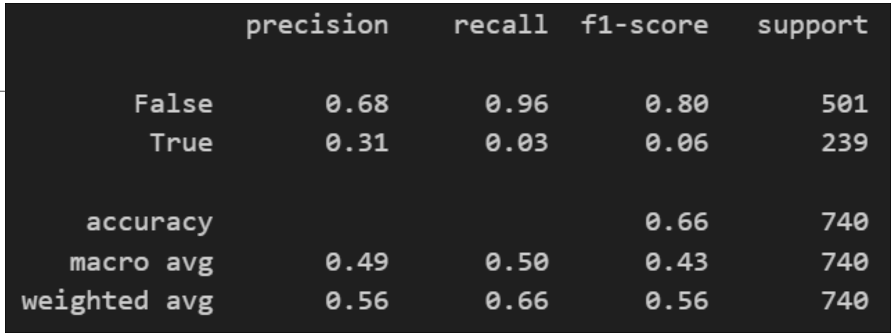




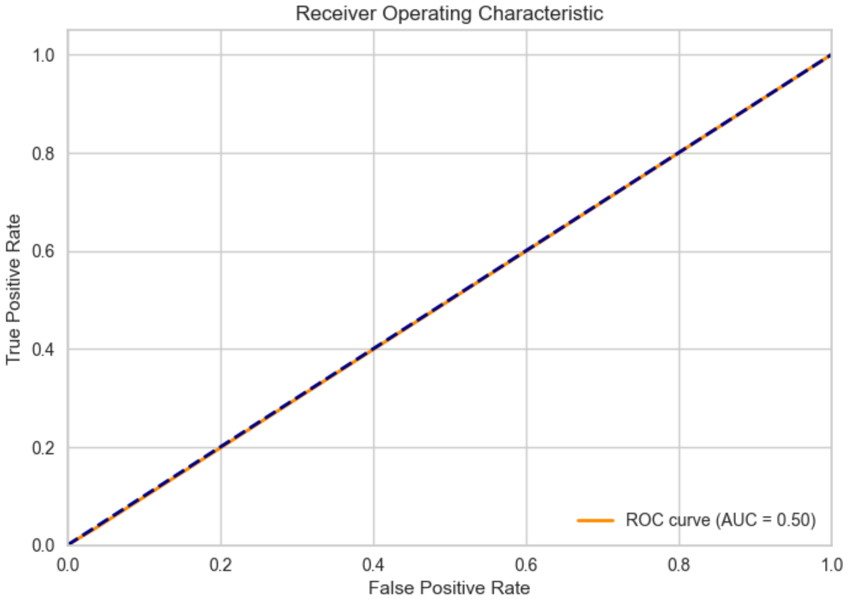


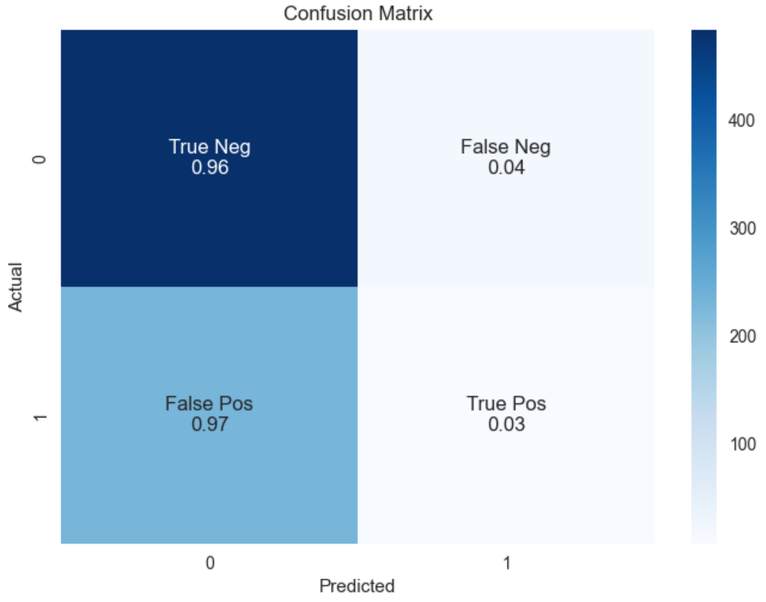
LOGISTIC REGRESSION **WITHOUT** HYPERPARAMETERS

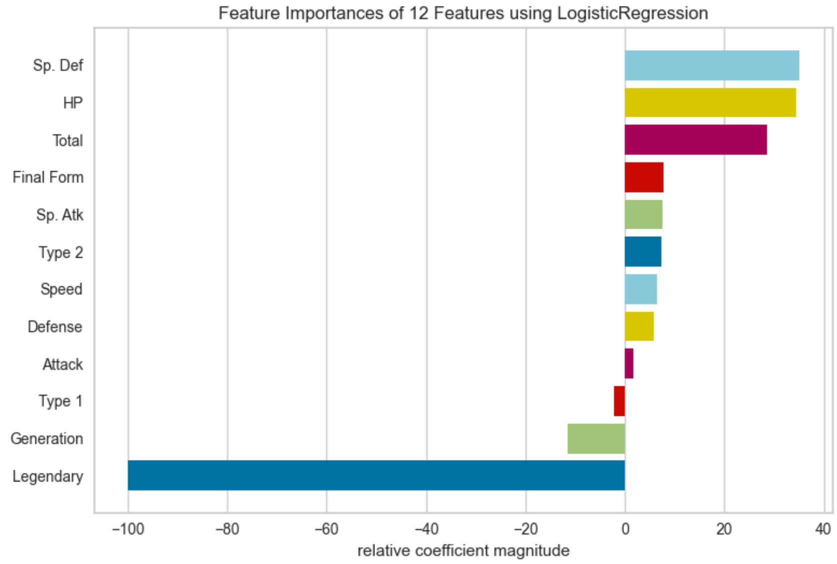
Testing accuracy: 0.6635135135135135
Training accuracy: 0.6385793871866295



```
              precision    recall  f1-score   support

       False       0.68      0.96      0.80       501
        True       0.31      0.03      0.06       239

    accuracy                           0.66       740
   macro avg       0.49      0.50      0.43       740
weighted avg       0.56      0.66      0.56       740
```



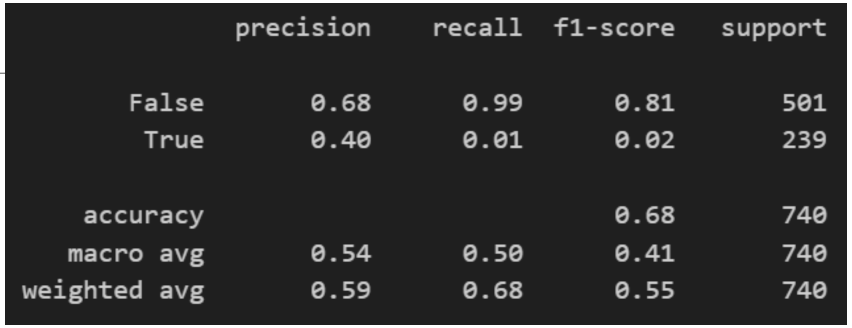




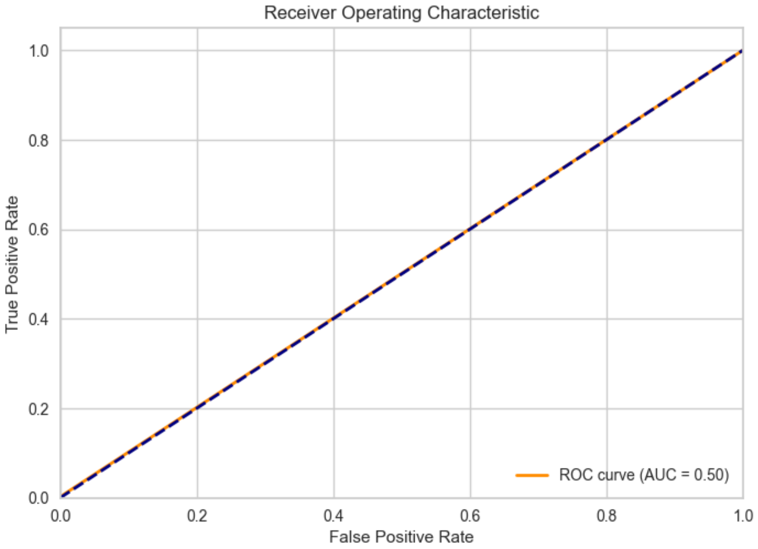
LOGISTIC REGRESSION WITH HYPERPARAMETERS

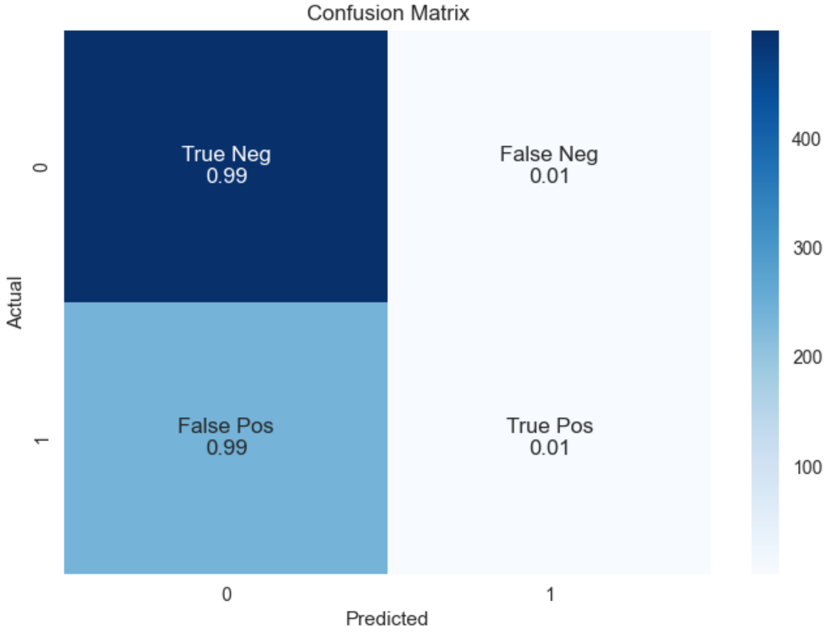
'C': 0.1
'l1_ratio': 0.1,
'penalty': ‘l1’
'solver': 'liblinear'

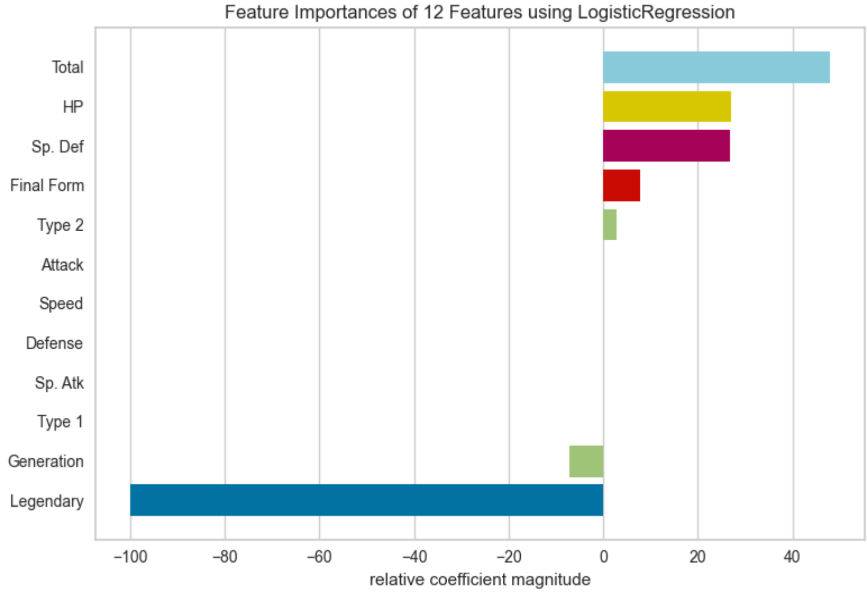
Testing accuracy: 0.6756756756756757
Training accuracy: 0.6448467966573816



```
              precision    recall  f1-score   support

       False       0.68      0.99      0.81       501
        True       0.40      0.01      0.02       239

    accuracy                           0.68       740
   macro avg       0.54      0.50      0.41       740
weighted avg       0.59      0.68      0.55       740
```

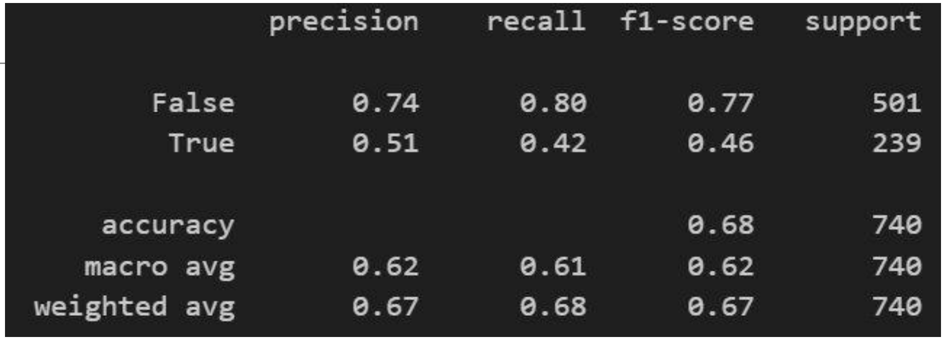




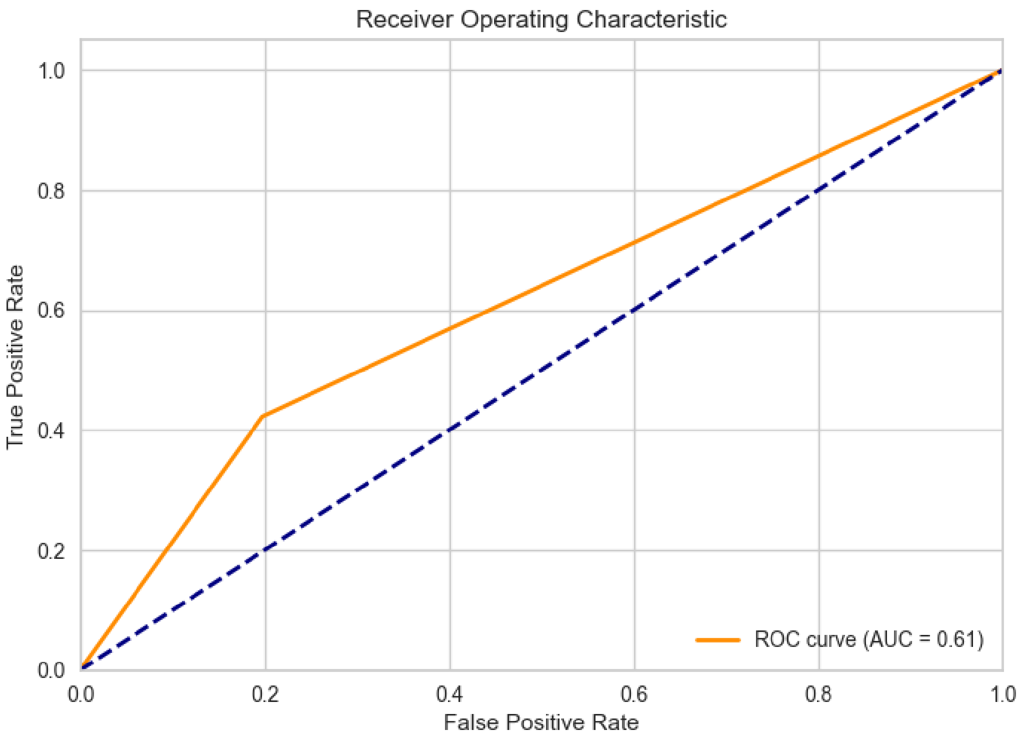


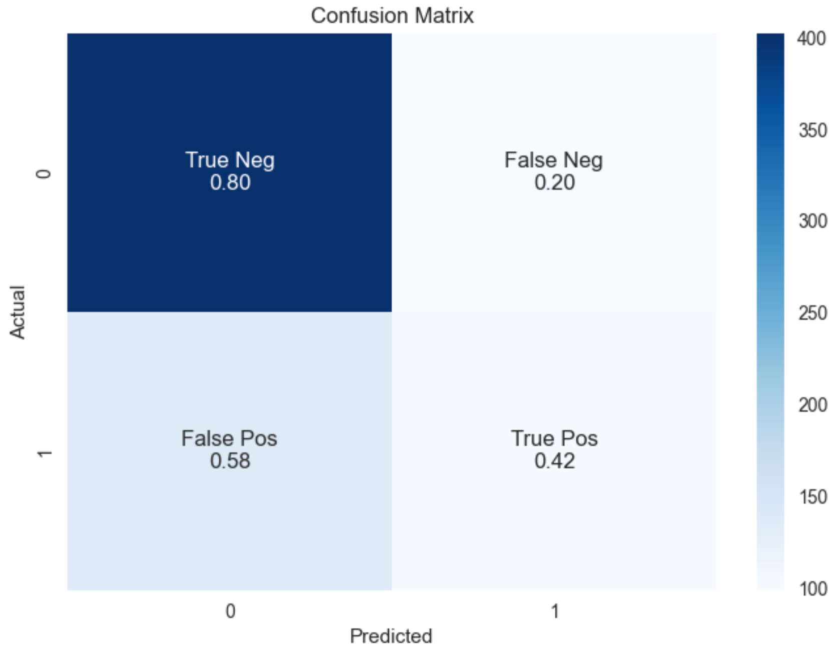
K-NEAREST NEIGHBORS **WITHOUT** HYPERPARAMETERS

Testing accuracy: 0.6797297297297298
Training accuracy: 0.7583565459610028



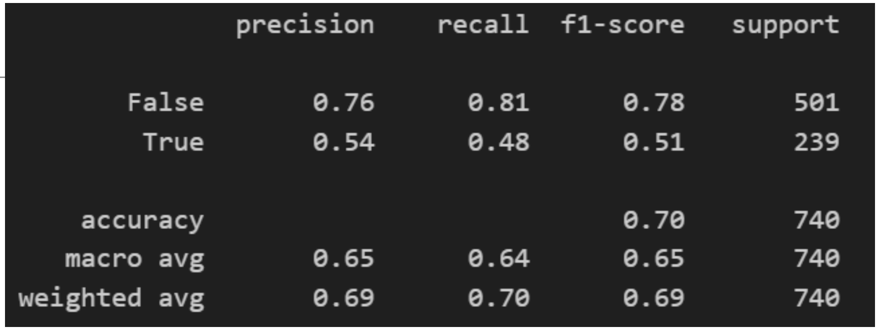
```
              precision    recall  f1-score   support

       False       0.74      0.80      0.77       501
        True       0.51      0.42      0.46       239

    accuracy                           0.68       740
   macro avg       0.62      0.61      0.62       740
weighted avg       0.67      0.68      0.67       740
```





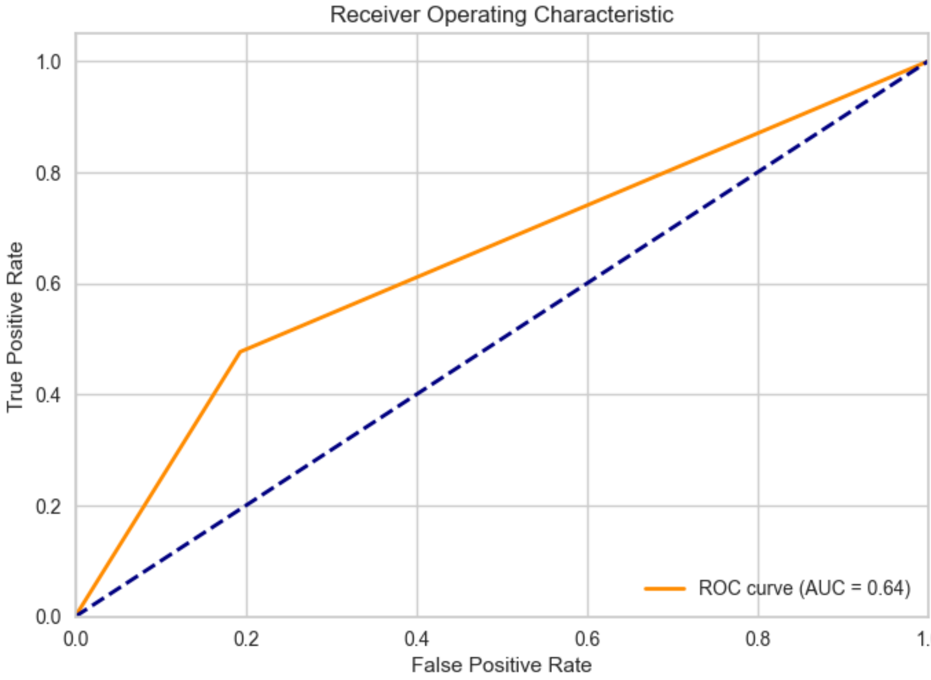
K-NEAREST NEIGHBORS WITH HYPERPARAMETERS

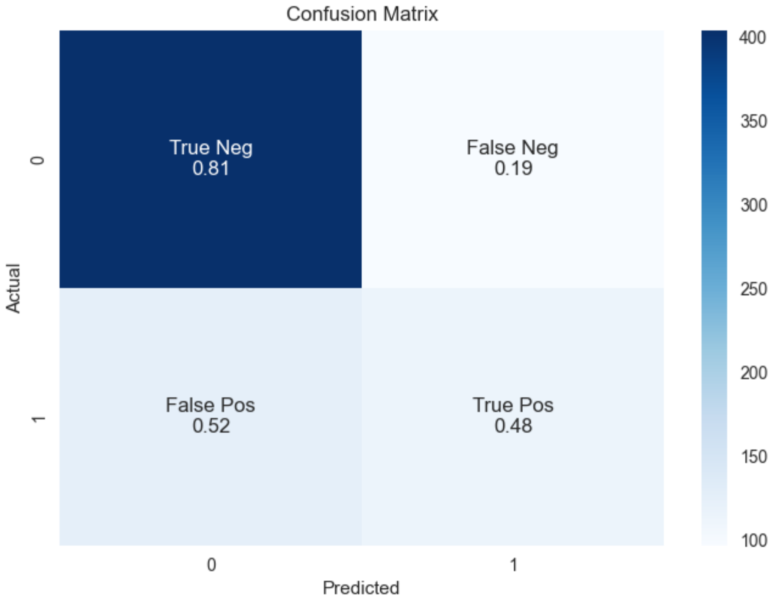
Testing accuracy: 0.7
Training accuracy: 0.9993036211699164



```
              precision    recall  f1-score   support

       False       0.76      0.81      0.78       501
        True       0.54      0.48      0.51       239

    accuracy                           0.70       740
   macro avg       0.65      0.64      0.65       740
weighted avg       0.69      0.70      0.69       740
```

'metric': 'manhattan',
'n_neighbors': 7,
'weights': 'distance'





CONCLUSIONS

IS LEGENDARY

- **The best model is LR with hyperparameters**, achieving a testing accuracy of 0.9473 and a training accuracy of 0.95, successfully avoiding overfitting.
- **The worst model is KNN with hyperparameters**, which suffers from overfitting and has the lowest accuracy among all models.

IS IN ITS SECOND EVOLUTION

- **The best model is the RF without hyperparameter tuning**, it achieves the best balance between testing and training performance.
- **The worst model is LR** (with or without hyperparameter tuning), which has a ROC AUC of 0.50—meaning it cannot distinguish between the positive and negative classes, equivalent to a coin toss.

CONCLUSIONS

Is Legendary? Is an easier variable to predict because Legendary Pokémon typically have much higher stats, and just a few key features are often enough to identify them.

Is Second Evolution? Is harder to predict, as it depends on the Pokémon's evolutionary structure, which is not always linear or uniform. Additionally, the data is less distinctive — for example, a second-stage Pokémon might have stats similar to those in the first or final stage.

THANK YOU