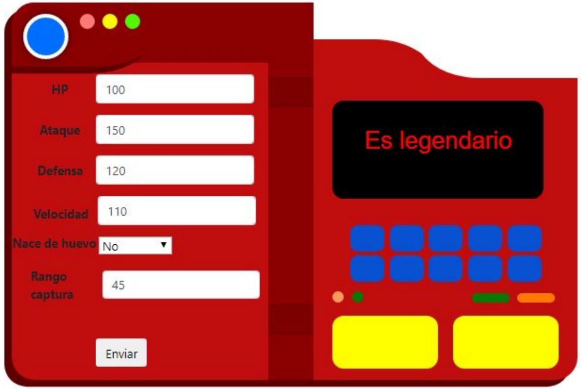
# Pokedex



HP: 100

Ataque: 150

Defensa: 120

Velocidad: 110

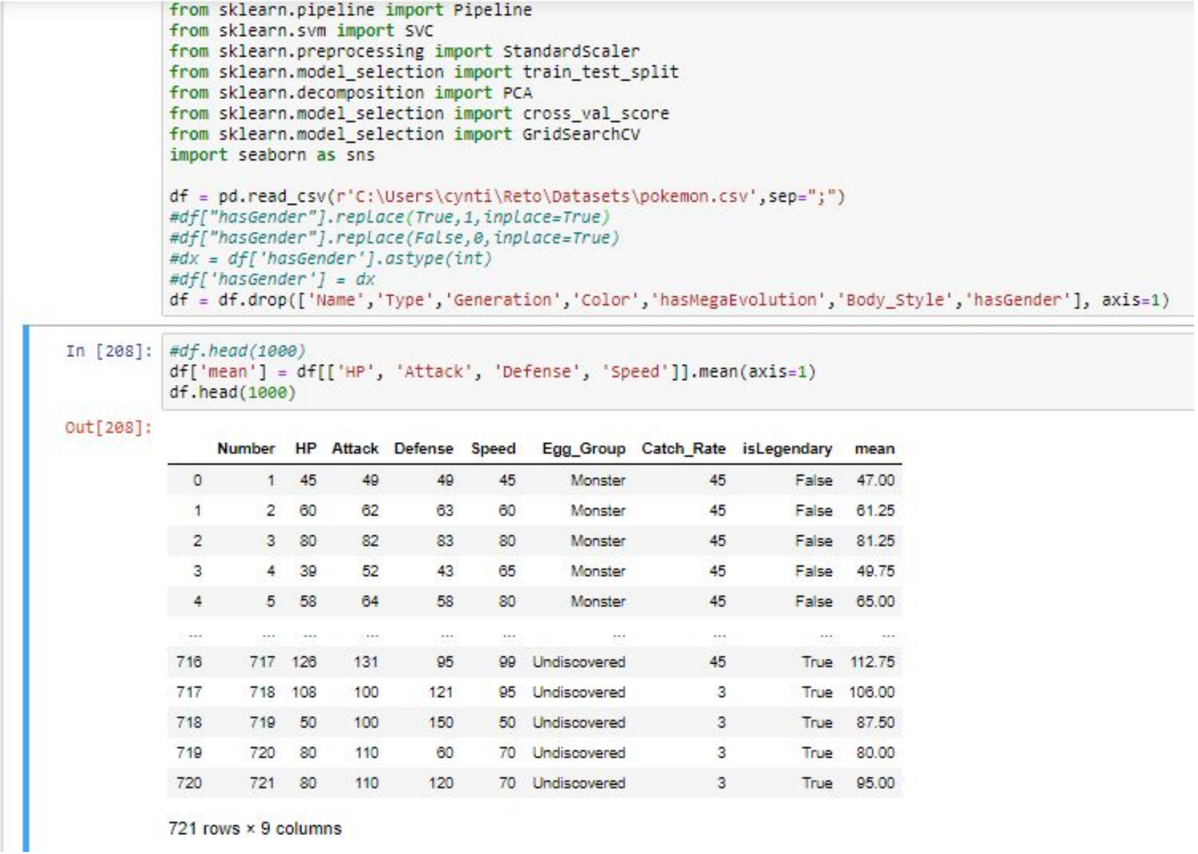
Nace de huevo: No

Rango captura: 45

Es legendario

Por: Cintia , Lorena , Christian y David
2º DAW 19/20

# Identificamos y eliminamos, los campos innecesarios



```
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
import seaborn as sns

df = pd.read_csv(r'C:\Users\cynti\Reto\Datasets\pokemon.csv',sep=";")
#df["hasGender"].replace(True,1,inplace=True)
#df["hasGender"].replace(False,0,inplace=True)
#dx = df['hasGender'].astype(int)
#df['hasGender'] = dx
df = df.drop(['Name','Type','Generation','Color','hasMegaEvolution','Body_Style','hasGender'], axis=1)
```

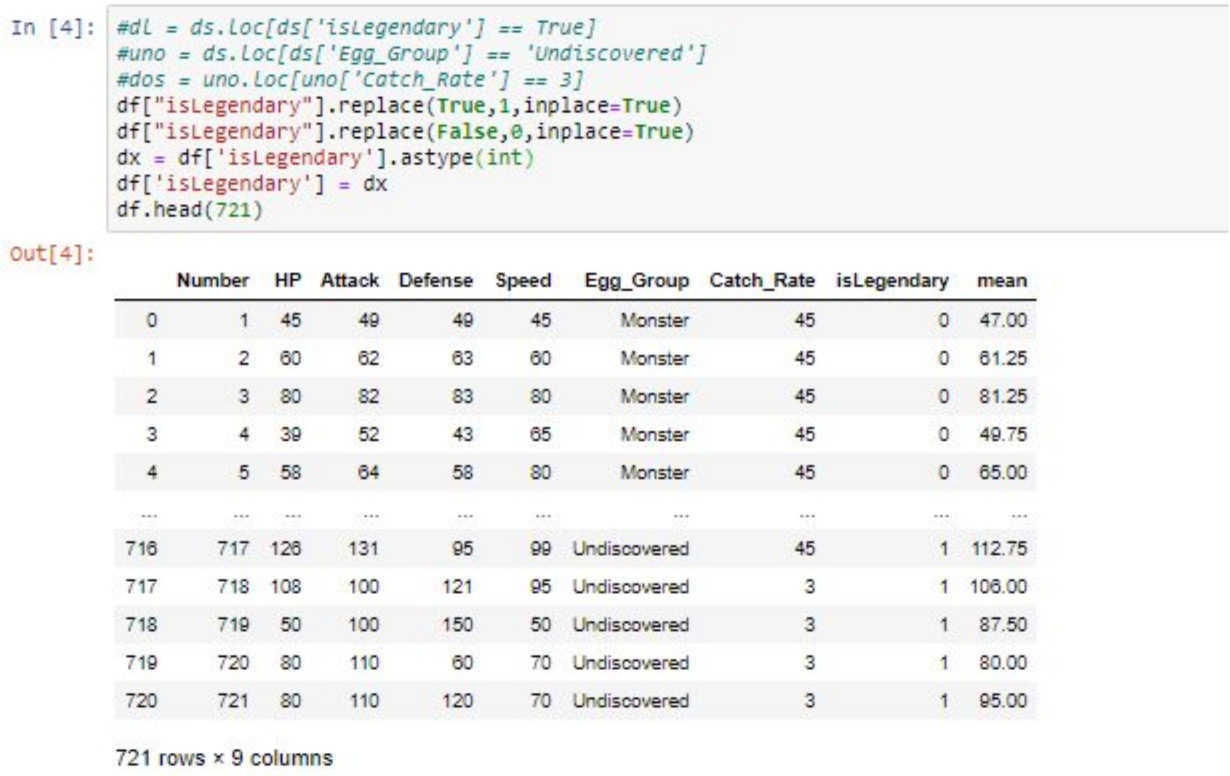
```
In [208]: #df.head(1000)
          df['mean'] = df[['HP', 'Attack', 'Defense', 'Speed']].mean(axis=1)
          df.head(1000)
```

Out[208]:

| | Number | HP | Attack | Defense | Speed | Egg_Group | Catch_Rate | isLegendary | mean |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 45 | 49 | 49 | 45 | Monster | 45 | False | 47.00 |
| 1 | 2 | 60 | 62 | 63 | 60 | Monster | 45 | False | 61.25 |
| 2 | 3 | 80 | 82 | 83 | 80 | Monster | 45 | False | 81.25 |
| 3 | 4 | 39 | 52 | 43 | 65 | Monster | 45 | False | 49.75 |
| 4 | 5 | 58 | 64 | 58 | 80 | Monster | 45 | False | 65.00 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 716 | 717 | 126 | 131 | 95 | 99 | Undiscovered | 45 | True | 112.75 |
| 717 | 718 | 108 | 100 | 121 | 95 | Undiscovered | 3 | True | 106.00 |
| 718 | 719 | 50 | 100 | 150 | 50 | Undiscovered | 3 | True | 87.50 |
| 719 | 720 | 80 | 110 | 60 | 70 | Undiscovered | 3 | True | 80.00 |
| 720 | 721 | 80 | 110 | 120 | 70 | Undiscovered | 3 | True | 95.00 |

721 rows × 9 columns

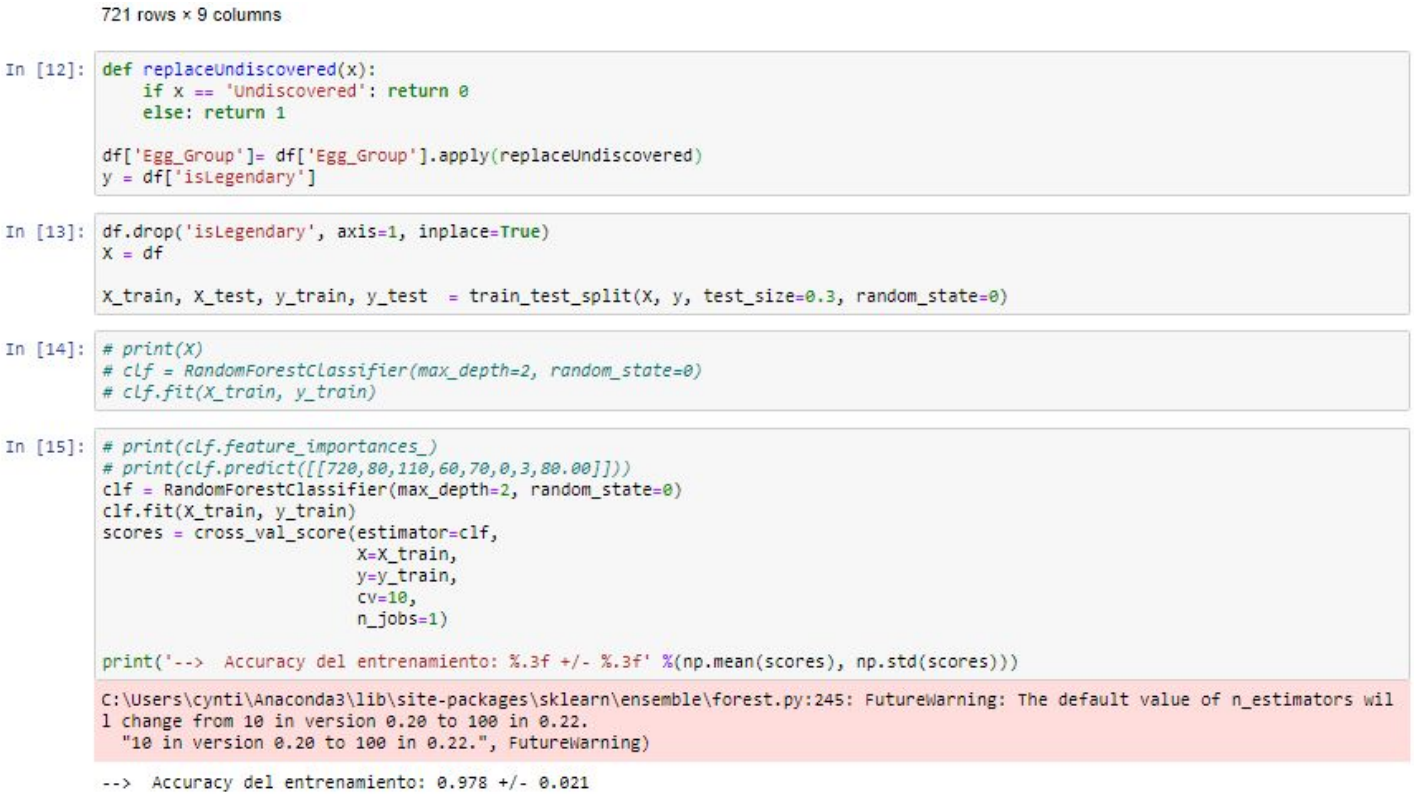
# Cambiamos el campo isLegendary por 0 y 1



```
In [4]: #dL = ds.loc[ds['isLegendary'] == True]
        #uno = ds.loc[ds['Egg_Group'] == 'Undiscovered']
        #dos = uno.loc[uno['Catch_Rate'] == 3]
        df["isLegendary"].replace(True,1,inplace=True)
        df["isLegendary"].replace(False,0,inplace=True)
        dx = df['isLegendary'].astype(int)
        df['isLegendary'] = dx
        df.head(721)
```

```
Out[4]:
      Number   HP  Attack  Defense  Speed     Egg_Group  Catch_Rate  isLegendary    mean
  0        1   45      49       49     45       Monster          45            0   47.00
  1        2   60      62       63     60       Monster          45            0   61.25
  2        3   80      82       83     80       Monster          45            0   81.25
  3        4   39      52       43     65       Monster          45            0   49.75
  4        5   58      64       58     80       Monster          45            0   65.00
...      ...  ...     ...      ...    ...           ...         ...          ...     ...
716      717  126     131       95     99  Undiscovered          45            1  112.75
717      718  108     100      121     95  Undiscovered           3            1  106.00
718      719   50     100      150     50  Undiscovered           3            1   87.50
719      720   80     110       60     70  Undiscovered           3            1   80.00
720      721   80     110      120     70  Undiscovered           3            1   95.00

721 rows × 9 columns
```

# Creamos el test para el entrenamiento y la efectividad



```
721 rows × 9 columns
```

```
In [12]: def replaceUndiscovered(x):
             if x == 'Undiscovered': return 0
             else: return 1

         df['Egg_Group']= df['Egg_Group'].apply(replaceUndiscovered)
         y = df['isLegendary']
```

```
In [13]: df.drop('isLegendary', axis=1, inplace=True)
         X = df

         X_train, X_test, y_train, y_test  = train_test_split(X, y, test_size=0.3, random_state=0)
```

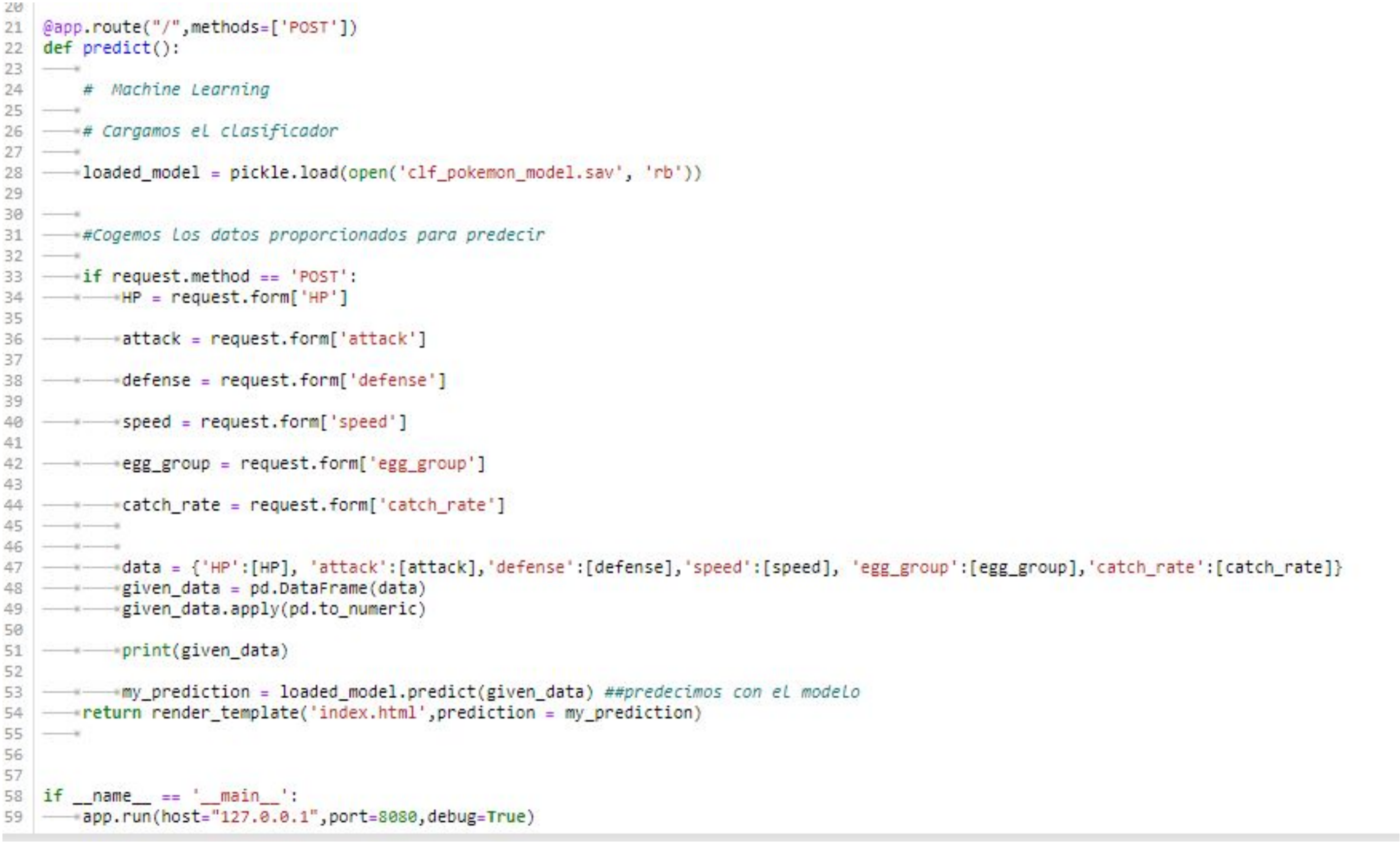
```
In [14]: # print(X)
         # clf = RandomForestClassifier(max_depth=2, random_state=0)
         # clf.fit(X_train, y_train)
```

```
In [15]: # print(clf.feature_importances_)
         # print(clf.predict([[720,80,110,60,70,0,3,80.00]]))
         clf = RandomForestClassifier(max_depth=2, random_state=0)
         clf.fit(X_train, y_train)
         scores = cross_val_score(estimator=clf,
                                  X=X_train,
                                  y=y_train,
                                  cv=10,
                                  n_jobs=1)

         print('-->  Accuracy del entrenamiento: %.3f +/- %.3f' %(np.mean(scores), np.std(scores)))
```

```
C:\Users\cynti\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n_estimators wil
l change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

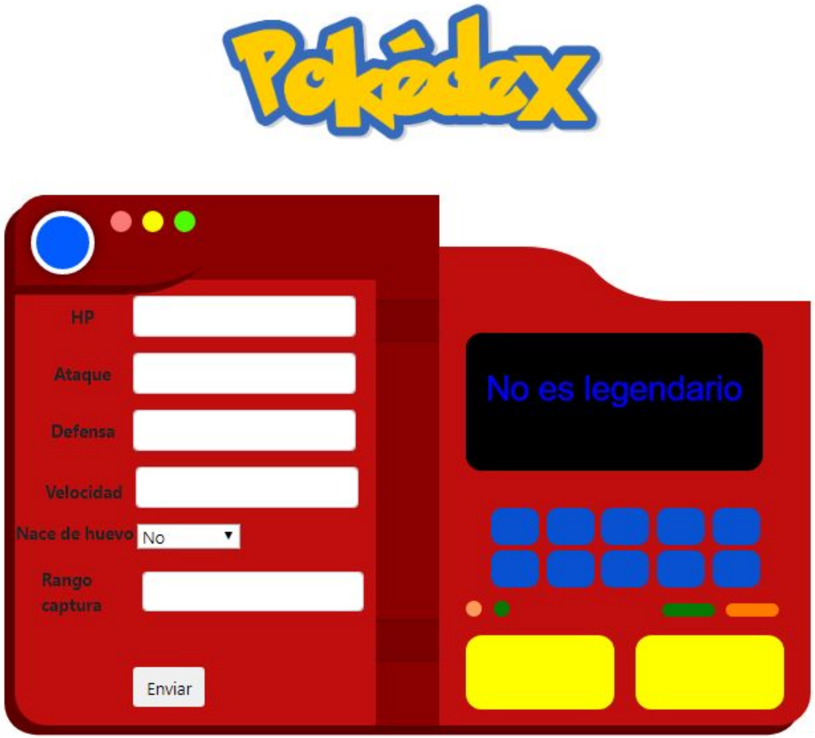
```
-->  Accuracy del entrenamiento: 0.978 +/- 0.021
```

# Información de los campos



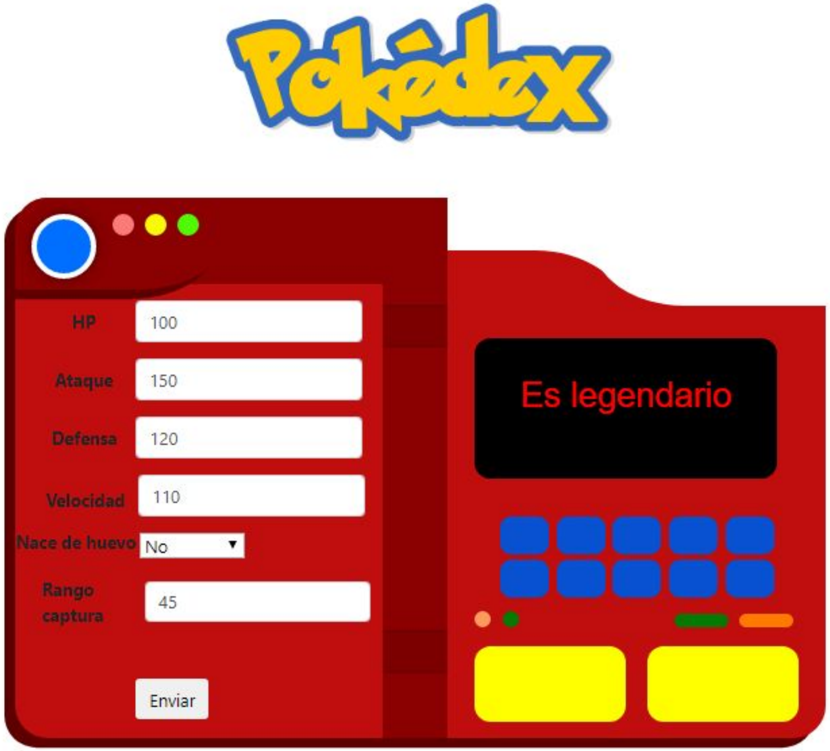
```
@app.route("/",methods=['POST'])
def predict():

    #  Machine Learning

    # Cargamos el clasificador

    loaded_model = pickle.load(open('clf_pokemon_model.sav', 'rb'))


    #Cogemos los datos proporcionados para predecir

    if request.method == 'POST':
        HP = request.form['HP']

        attack = request.form['attack']

        defense = request.form['defense']

        speed = request.form['speed']

        egg_group = request.form['egg_group']

        catch_rate = request.form['catch_rate']


        data = {'HP':[HP], 'attack':[attack],'defense':[defense],'speed':[speed], 'egg_group':[egg_group],'catch_rate':[catch_rate]}
        given_data = pd.DataFrame(data)
        given_data.apply(pd.to_numeric)

        print(given_data)

        my_prediction = loaded_model.predict(given_data) ##predecimos con el modelo
    return render_template('index.html',prediction = my_prediction)



if __name__ == '__main__':
    app.run(host="127.0.0.1",port=8080,debug=True)
```

# Ejemplo funcionamiento



HP:
Ataque:
Defensa:
Velocidad:
Nace de huevo: No
Rango captura:

No es legendario



HP: 100
Ataque: 150
Defensa: 120
Velocidad: 110
Nace de huevo: No
Rango captura: 45

Es legendario

# Gracias por la atención!!!