UNIVERSIDADE DE SÃO PAULO

Dérick William de Moraes Frias

MAC0460 - Exercício Programa 2: PokEP

São Paulo - SP

2023

1. Introdução

O exercício proposto era o de utilizar alguns modelos de machine learning para ajudar o Professor Carvalho a classificar pokémons (Quem diria)! Para isso, utilizei os algoritmos propostos, de SVM, Decision Tree, Random Forest e Regressão Linear.

2. Metodologia:

O dataset utilizado foi um arquivo .csv (pokedata.csv) contendo uma longa série de dados sobre pokemons. A partir deve arquivo bruto, removi alguns dos parâmetros que, de alguma forma, não poderiam ser utilizados e montei o dataframe (df) principal com o restante de dados. Então, após uma separação desse dataframe em dados de treino e dados de teste.

Utilizei uma seed fixa em todos os geradores aleatórios com o intuito de obter a repetibilidade dos experimentos na correção pelos monitores.

Os modelos, já supracitados, foram SVM, Decision Tree, Random Forest e Regressão Linear. Estes foram testados primeiramente num contexto com dados não-normalizados, e depois com dados normalizados. Os resultados dessas duas abordagens geraram um contraste de relevância interpretativa.

No treinamento de cada modelo, utilizei GridSearch para encontrar a melhor combinação de hiperparâmetros, juntamente com k Fold cross-validation.

Os hiperparâmetros e features escolhidos estão descritos no próprio código.

3. Resultados

SVM

Parâmetros escolhidos: {‘C’: 0.1, ‘gamma’: ‘auto’, ‘kernel’: ‘rbf’, ‘max_iter’: 30}
Acurácia: 0.5581395348

Decision Tree

Parâmetros escolhidos: {‘criterion’: ‘gini’, ‘max_depth’: 5, ‘min_samples_split’: 12, ‘splitter’: ‘best’}
Acurácia: 0.8604651162

Random Forest

Parâmetros escolhidos: {‘class_weight’: ‘balanced’, ‘criterion’: ‘gini’, ‘max_depth’: 5, ‘min_sample_split’: 2, ‘n_estimators’: 300}
Acurácia: 0.8372093023

Regressão Linear

Parâmetros escolhidos: {‘C’: 0.01, ‘max_iter’: 100, ‘penalty’: ‘12’, ‘solver’: ‘newton-cg’}
Acurácia: 0.6279069767

SVM (Normalizado)

Parâmetros escolhidos: {‘C’: 2, ‘gamma’: 0.5, ‘kernel’: ‘rbf’, ‘max_iter’: 200}
Acurácia: 0.7906976744

Decision Tree (Normalizado)

Parâmetros escolhidos: {‘criterion’: ‘gini’, ‘max_depth’: 5, ‘min_samples_split’: 12, ‘splitter’: ‘best’}
Acurácia: 0.8372093023

Random Forest (Normalizado)

Parâmetros escolhidos: {‘class_weight’: ‘balanced’, ‘criterion’: ‘gini’, ‘max_depth’: 5, ‘min_sample_split’: 2, ‘n_estimators’: 300}
Acurácia: 0.8604651162

Regressão Linear (Normalizado)

Parâmetros escolhidos: {‘C’: 1, ‘max_iter’: 10, ‘penalty’: ‘11’, ‘solver’: ‘liblinear’}
Acurácia: 0.7906976744

4. Discussão

Os resultados mostram que a normalização dos dados melhora a acurácia nos modelos de Regressão Linear, SVM e Random Forest. Na Decision Tree, a acurácia foi melhor com dados não normalizados. Isso se dá pela natureza da Decision Tree, que separa o hiperespaço de features por regiões de decisão, e essas regiões estarem mais acertadas quanto mais os dados forem fidedignos a amostrar do ‘ambiente real’. Nesse contexto então, a normalização pode acabar deslocando os features de foma que a acurácia sofre uma pequena perda.

Na Decision Tree e na Regressão Linear, o feature mais relevante é ‘against_electric’, o que faz sentido levando em conta que nosso ajudante pikachu consegue classificar uma boa parte dos pokémons somente usando do contraste desse atributo com os tipos ‘Water’ ou ‘Normal’.

Os valores de acurácia para quando os dados estão normalizados, ficam no entorno de 0.80. Nesse patamar, é um dado relevante para o uso classificatório de pokémons por parte do professor Carvalho.

5. Conclusão

A normalização de dados se mostrou um ferramenta poderosa para otimizar o uso dos modelos de learning. Além disso, o comportamento dos modelos foi bem equilibrado ao final das análises, estando todos aptos a gerarem uma resposta suficientemente precisa para ajudar na classificação dos pokémons.