EP2 - Expedição Para Catalogar

Aprendizado de Maquina - MAC5832

Henrique Santos Apocalypse - NUSP: 14198290

9 de julho de 2023

1 Introdução

No EP2 dado o seguinte contexto que uma nova região pokemon, ainda não catalogada, vai ser explorada. Assim, você e o seu fiel Pikachu tem o dever de classificar alguns pokemons dessa região em *type1 == water* e *tyoe1 == normal*. Assim, a nossa missão é, com base nos dados da pokedex, criar um classificador binário para determinar se o *type1* do pokemon é *water* ou *normal*. Dentre os classificadores, utilizamos o Regressão Logística, Support Vector Machine, Decision Tree e Random Forest, para verificar a acuracia perante alguns parametros especificos de cada modelo de aprendizado.

2 Metodologia

2.1 Dataset

O conjunto de dados apresenta todos os dados relevantes no mundo pokemon. Nele temos informações completas sobre nome, fraquezas, vantagens, status, tipo, classificação e outros. Para o treinamento, filtrei apenas os itens que *type1 == "water"* e *type1 == "normal"*, em seguida, separei os parâmetros solicitados, como o "against_electric"e a feature "base_total". Logo em seguida separamos o dataset em um conjunto para treinamento e outro para testes, para o final. Utilizei um seed em random_state para conseguir reproduzir e avaliar os resultados, igual ao EP1. Por fim, os valores foram padronizados com *StandardScaler()* da biblioteca *sklearn.preprocessing* que deixa os dados na mesma escala, com média zero e desvio padrão unitário.

2.2 Modelos, Parâmetros, Hiperparâmetros e Features Utilizados

2.2.1 Modelos

Regressão Logística, Support Vector Machine, Decision Tree, Random Forest.

2.2.2 Parâmetros

Para o nosso caso, o único parâmetro que podemos ter é o da coluna "against_electric". Essa coluna informa o quanto de "dano"um pokemon recebe de um pokemon do *type1 == eletric*. Assim, o modelo aprende a relação que existe entre o dano recebido contra pokemons elétricos e a classificação dos pokemons em *Water* ou *Normal*.

2.2.3 Hiper-parâmetros

Para cada modelo foi utilizado a biblioteca utilizado a função *"GridSearchCV"* da biblioteca *sklearn*. Retornou os seguintes hiperparametors para os modelos:

- Decision Tree, retornou *max_depth:* 4 e *min_samples_leaf:* 4
- SVM, retornou *C:* 1*, degree:* 1 *e kernel: rbf*
- Logistic Regression, retornou *C:* 1
- Random Forest, retornou *max_depth:* 5*, min_samples_leaf:* 2*, n_estimators:* 200

2.2.4 Features

Como features, inicialmente, eu não identifiquei nenhuma propriedade que fizesse sentido, então não adicionei nenhuma. Dessa forma, todos os classificadores obtiveram quase a mesma precisão de $86,63\%$. Em alguns testes, resolvi adicionar as features, e coloquei a *base_total*, que aumentou significativamente a precisão dos meus classificadores.

2.3 Avaliação dos Modelos e Resultados

Para Avaliar cada modelo, separei o conjunto de dados em duas partes, uma com 80% para treinamento, e uma menor com 20% para validação. Dessa forma, ao final, comparei os resultados dado por cada learn algorithm e comparei com o resultado real. Para cada caso, foi obtido:

- Decision Tree: $90,9\%$
- Logistic Regression: $81,81\%$
- SVM: $86,36\%$
- Random Forest: $93,18\%$ (máximo)

3 Discussão

3.1 Features

Inicialmente, não senti necessidades de utilizar alguma feature. Primeiro por que não entendi a relação de nenhuma delas com a classificação do tipo do pokemon. Outro motivo foi que, pela ideia proposta, não imaginei que conseguiríamos essas informações na situação hipotética de encontrar um pokemon desconhecido pelo caminho. Assim, o resultado sempre dava por volta de 86%. Ao analisar os dados, notei que existia pokemons normais que levavam 2.0 de dano para o tipo elétrico. Esse fato se deve pelo *type2*, por exemplo, um pokemon chamado pidgey. Esse fato, levava a uma classificação errada do problema e essa precisão de 86%.

Para contornar isso, experimentei as features, e a que mais trouxe resultado das que eu testei, foi o *base_total*. Com isso, de alguma forma ele conseguia prever de forma melhor esses casos onde o *type2* influenciava no valor de "against_electric". Isso melhorou alguns dos classificadores, como o *Decision Tree* com $90,9\%$ e *Random Forest* com $93,18\%$. Esse resultado me deixou confuso por não entender diretamente essa relação entre os status e o tipo primário e secundário do pokemon, mas feliz, pois melhorou a precisão dos classificadores.

3.2 Utilidade dos Dados

Esses dados poderiam ajudar sim o professor Carvalho na sua aventura de classificação de novos pokemons utilizando apenas um pokemon do tipo elétrico. Para melhorar essa tarefa, ele poderia levar um pokemon de outro tipo para ajudar essa classificação, combinando os dois *against* para determinar com melhor precisão os pokemons, não deixando que o *type*2 levasse o classificador a erros. Dando uma olhada em um site sobre vantagens e desvantagens de pokemons, podemos ver que, pokemon elétrico fornece 2.0 de dano no tipo água e voador e um pokemon de terrestre fornece 0.0 em voador e 1.0 em normal, poderíamos ter:

- "against_electric"= 2.0 e "against_rock"= 1.0 pokemon é tipo1 água
- "against_electric"= 2.0 e "against_rock"= 0.0 pokemon é tipo1 normal e tipo2 voador
- "against_electric"= 1.0 e "against_rock"= 1.0 pokemon é tipo1 normal
- "against_electric"= 1.0 e "against_rock"= 0.0 pokemon nem normal, nem água.

Dessa forma, ajudando a catalogar melhor os pokemons que fossem aparecendo pelo caminho, descartando esse problema de dar o dobro de dano. Outra dica mais pratica, seria, ver se o pokemon tem asas quando aplicar um ataque elétrico e retornar 2.0, notaríamos de cara que é do tipo voador e normal!

4 Conclusão

Por fim, acho que esse é um modelo simples que vai ajudar de forma consistente o professor Carvalho a classificar pokemons de *type1* entre *water* e *normal*. Obviamente, teria erros com os casos onde o *type*2 influenciasse no "against_electric", como é o caso de pokemons voadores. Uma boa proposta seria o professor Carvalho permitir que fosse levado mais pokemons de tipos distintos, como o terrestre por exemplo, seria uma feature que permitiria avaliar esse cenário que gera problemas.