Trabalho 1 - INF-0615

Kaleb Roncatti de Souza e Nelson Gomes Brasil Júnior

1 de outubro de 2022

1 Introdução

No trabalho vigente, iremos analisar um conjunto de dados relacionado a redes sociais. A ideia central do projeto é realizar a predição do número de compartilhamentos num post dadas algumas *features* com características de cada um dos posts. Ao longo de todo o trabalho, utilizaremos o modelo de regressão linear com a linguagem `R`, realizando-se combinações entre as diferentes *features* e também variações de graus de polinômios. É importante notar que os resultados foram apresentados neste relatório em formato de texto corrido.

2 Tarefas

Nesta seção, segmentaremos as respostas apresentando de maneira incremental os resultados obtidos via código.

2.1 Inspeção dos dados

Começamos o projeto através da leitura dos conjuntos de dados de treino, validação e teste. Analisando o número de exemplos, temos 9336 (64% do total) no conjunto de treino, 2334 (16%) no de validação e 2918 no de teste (20%). Das features apresentadas, apenas uma delas pode ser considerada como categórica: **weekday**, que diz qual dia da semana a publicação foi ou será feita. Para tratar este tipo de dado, podemos utilizar a técnica de `One Hot Encoding`, a qual transforma a coluna weekday em sete colunas (**sunday**, **monday**, ..., **saturday**) também discretizadas da seguinte forma: A feature **sunday** será 1 para os dados em que a feature **weekday** tem o valor *sunday* e 0 caso contrário, e o mesmo vale para todas as features que criaremos. Teremos então, 17 features numéricas e 7 categóricas.

Na base de dados que temos, não existe nenhum exemplo sem anotação, portanto não precisamos realizar nenhuma ação. Porém, se fosse o caso, poderíamos tomar algumas atitudes, como por exemplo preencher os dados com a média ou mediana dos dados daquela feature (ou um agrupamento de dados da feature). Dependendo da quantidade e da relevância dos dados faltantes, podemos removê-los sem prejuízo para o resultado final do modelo que será posteriormente treinado.

Ponto extremamente importante é, aplicamos os tratamentos necessários na base de teste mas a deixamos **intocável** até os momentos de obtenção de predições, tal conjunto não deve **em nenhuma hipótese** ser utilizado ao longo do processo de treinamento.

2.2 Normalização

Após a criação das features categóricas, vamos manipular as features numéricas fazendo uma normalização das mesmas, de modo a não termos valores muito discrepantes entre si os quais possivelmente enviesariam nossos modelos, e desta forma prontos para o treinamento. Para isto, escolhemos a técnica da `Z-Norma`, a qual calcula a média (μ) e o desvio padrão (σ) para cada feature do conjunto de treinamento e aplica a seguinte transformação:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

aos dados de treino, validação e teste. Note que, ao aplicar esta transformação aos dados de treino, todas as features terão média 0 e variância 1 e, se os dados seguirem uma distribuição normal padrão, nada ocorreria.

2.3 Regressão linear - Baseline

Para o baseline, vamos considerar que iremos usar todas as 24 sem nenhuma análise prévia ou combinação entre elas. Ou seja, iremos simplesmente procurar uma conjunto de parâmetros $\{\theta_1, \ldots, \theta_{24}\}$ que melhor ajusta a função $f(x) = \theta_1 x_1 + \ldots + \theta_{24} x_{24}$ no conjunto de dados de treino, onde aqui x_i, $i = 1, \ldots, 24$ representa cada uma das features. Os valores dos coeficientes podem ser conferidos no código anexo a este trabalho.

Para o baseline, o erro no conjunto de validação e treino foi: $MAE_{validation} = 718.93$ e $MAE_{train} = 716.38$. Já para o conjunto de teste, obtivemos uma $MAE_{test} = 727.63$

2.4 Regressão linear - Combinação de *features*

No que diz respeito à combinação de features, começamos explorando o conjunto a nível correlação entre features numéricas para entendermos melhor quais características se influenciam entre si. Pode-se observar na figura 1 que as features:

- `global_sentiment_polarity`
- `global_rate_positive_words`
- `global_rate_negative_words`
- `rate_positive_words`
- `rate_negative_words`

possuem um alto grau de correlação entre si, assim como as features:

- `log_self_reference_max_shares`
- `log_self_reference_avg_sharess`

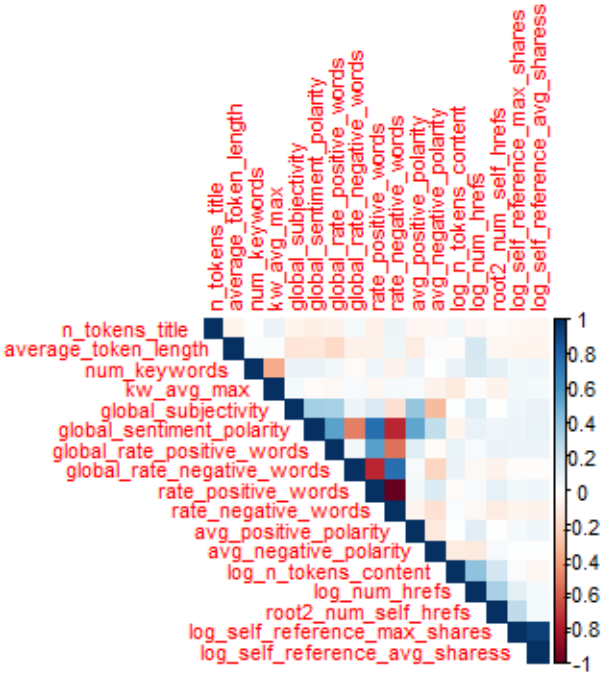

Figura 1: Gráfico de correlação entre as features numéricas

Desta maneira, para a criação dos modelos personalizados, começamos com um modelo que inclue todas as features (através de . em R) mais as combinações (grau 2) de praticamente todas as features selecionadas, apenas deixamos de lado a feature `rate_positive_words`, que possui grande correlação com a feature `rate_negative_words` (correlação igual a -1 neste caso). No entanto, percebemos que aumentando o grau de combinação deste modelo, o custo computacional começou a aumentar também dado que o modelo combina muitas features entre si (muitas combinações entre si), e optamos por testar outros tipos de modelos a nível experimental. O valor de MAE para tal modelo está representado no valor cuja abcissa `Model = 1` na Figura 2.

Para o outro tipo de modelo combinado que geramos, escolhemos um modelo genérico que combinou features que possuem menor correlação entre si, porque imaginamos que a combinação de features menos correlacionadas gerariam mais não-linearidades e a princípio poderiam melhorar o desempenho dos nossos modelos. Ademais, variou-se o grau de combinação (2 até 5), e os valores de MAE estão representados no valor cujas abcissas `Model = 2 -> 5` na Figura 2. Conseguimos observar o fenômeno de *overfitting* para `Model = 4 -> 5`, mas ainda assim, o `Model = 1` se apresentou com melhor MAE para o conjunto de validação.

Assim sendo, para o melhor modelo com as features combinadas (`Model = 1`) tivemos os seguintes valores de MAE para os conjuntos de treino, validação e teste respectivamente: $MAE_{train} = 708.37$, $MAE_{validation} = 713.63$ e $MAE_{test} = 727.97$.

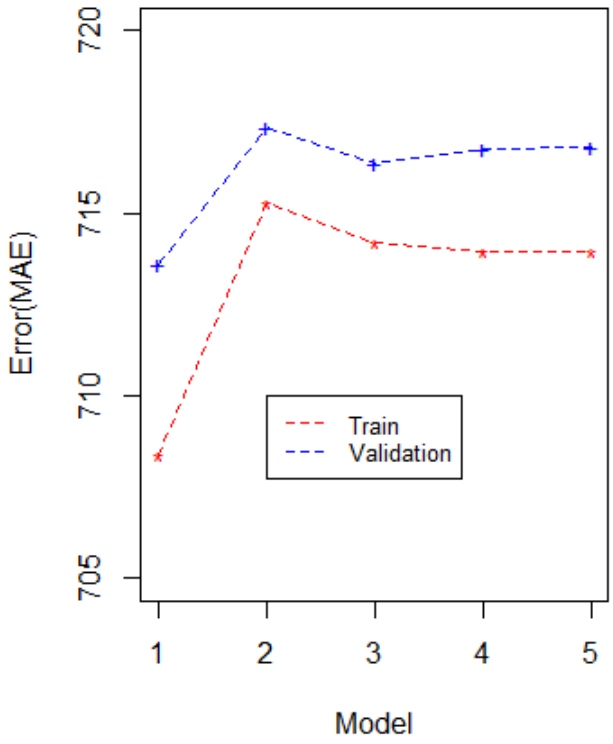

Figura 2: Gráfico do MAE pela aproximação por várias combinações de features

2.5 Regressão linear - Polinomial

Segundo o que vimos nas aulas e através de pesquisas na literatura, podemos perceber que a região de *underfitting* se mostra em $gr(p) \leq 3$ pois o erro no conjunto de treino e validação estão mais altos, levando ao entendimento de que a função escolhida é "simples demais" para generalizar os dados. O ponto ótimo se mostra em torno de $4 \leq gr(p) \leq 5$ e regiões de *overfitting* é a região onde $6 \leq gr(p)$, pois as funções escolhidas são tais que se ajustam bem ao conjunto de treino (e seu erro vai diminuindo), porém quando aplicado ao conjunto de validação leva a um erro maior, ou seja, o modelo não performa bem neste conjunto.

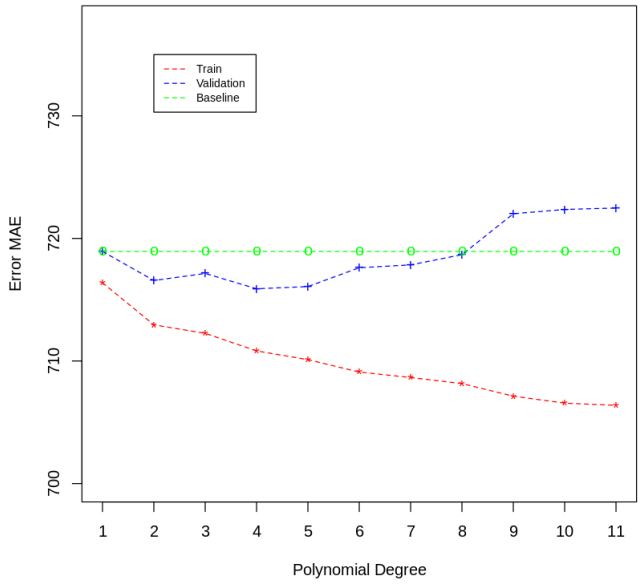

Figura 3: Gráfico do MAE pela aproximação polinomial conforme aumentamos o grau do polinômio

O polinômio de grau 4 foi tal que o erro no conjunto de validação foi o menor e portanto usaremos ele para calcular o erro no conjunto de teste, que foi: $MAE_{test} = 726.88$.

2.6 Conclusão

Através deste trabalho prático, fomos capazes de colocar em prática alguns elementos básicos de regressão linear estudados ao longo do curso de aprendizado de máquina supervisionado. Visualizamos na prática o efeito de *overfitting* de um modelo, região de erro na qual o conjunto de treinamento nos dá a falsa impressão de que continuamos melhorando o desempenho do modelo, mas quando observamos o conjunto de validação vemos que o que acontece na prática é que o modelo começa a realizar um *fit* muito específico dos dados de treino, não desempenhando tão bem em outros conjuntos genéricos.

No que diz respeito à combinação de features, realizamos uma experimentação inicial interessante baseando-se em valores de correlação, mas uma análise muito mais rigorosa poderia ser realizada em torno de tais modelos, combinando-se, por exemplo, features muito correlacionadas entre si e verificando se algumas não-linearidades são geradas de forma a melhorar nosso modelo.