Trabalho 2

INF-0615 – Aprendizado Supervisionado I

Prof. Anderson Rocha

IDENTIFICAÇÃO

Membros do grupo:

- Daniele Montenegro da Silva Barros
- Rodrigo Dantas da Silva
- Thiago Bruschi Martins

QUESTÃO 1: INSPEÇÃO DOS DADOS

Através da função nrow() verificamos o número de observações em cada data set:

- Conjunto de treino: 9204 linhas
- Conjunto de validação: 2303 linhas
- Conjunto de teste: 2878 linhas

Nesta seção verificamos ainda se existia algum dado repetido ou faltante entre os conjuntos de dados, mas não foi encontrado nenhum. Caso houvesse dados faltantes poderíamos utilizar algumas estratégias como substituir os dados faltantes pela média ou moda de cada coluna, o que ajudaria a contornar o problema e não afetaria a normalização das features. Caso houvesse features categóricas poderíamos ainda utilizar a média ou moda dentro de cada categoria, pois cada categoria de dados pode ter uma certa faixa de valores para determinada feature.

QUESTÃO 2: VERIFICANDO O BALANCEAMENTO

Utilizando o comando *table(trainSet$target),* verificamos que o conjunto de treino não está balanceado pois os dados correspondentes à classe positiva representam apenas 27% dos dados. Para lidar com este desbalanceamento vamos utilizar a estratégia de dar pesos diferentes para cada classe na hora de treinar o modelo. Poderíamos utilizar outra estratégia como Oversampling ou Undersampling, ou ainda combiná-las com a estratégia de diferentes pesos para cada classe. No entanto, como o desbalanceamento não é exagerado e o modelo que vamos utilizar (*glmnet*) já possui um parâmetro específico para lidar com o desbalanceamento através dos pesos de cada observação, concordamos que a estratégia mais eficiente seria apenas dar pesos diferentes para cada observação. Estes pesos serão calculados proporcionalmente de acordo com a frequência de cada classe no conjunto de treino. Recentemente vimos em aula que essa pode não ser a melhor opção, e o mais correto seria testar valores menores dependendo da proporção de dados na vida real e também levar em conta custo relativo de Falsos Positivos ou Falsos Negativos, que variam de caso para caso. No entanto, como não temos informações mais precisas para este problema, os pesos serão escolhidos da forma mais simples possível, visando equilibrar a proporção dos dados.

QUESTÃO 3: NORMALIZAÇÃO

A normalização escolhida foi a Z-Norma por ser a mais recomendada, a princípio, nas aulas, junto com a MinMax. Como todas as features são contínuas e não há dados faltantes nos conjuntos, não houve muitos problemas nesta etapa. Os únicos cuidados especiais que foram tomados aqui foram o de não normalizar o target e o de utilizar a média e desvio padrão dos dados de treino para normalizar os dados de validação e teste.

QUESTÃO 4: BASELINE

O modelo de baseline foi criado utilizando um modelo regressão logística (*glmnet*) antes do balanceamento das classes e utilizando todas as features disponíveis. Apresentamos a seguir a Matriz de Confusão Relativa do Baseline para os dados de treino, validação e teste. Cada uma das tabelas contém também a acurácia balanceada, TPN e TPR de cada conjunto.

Tabela 1: Matriz de Confusão Relativa do Baseline no Treino

Modelo: Baseline Acurácia Balanceada: 0.515		Predição	
		0	1
Referência	0	TPN: 0.99	0.01
	1	0.94	TPR: 0.06

Tabela 2: Matriz de Confusão Relativa do Baseline na Validação

Modelo: Baseline Acurácia Balanceada: 0.515		Predição	
		0	1
Referência	0	TPN: 0.98	0.02
	1	0.95	TPR: 0.05

Tabela 3: Matriz de Confusão Relativa do Baseline no Teste

Modelo: Baseline Acurácia Balanceada: 0.515		Predição	
		0	1
Referência	0	TPN: 0.67	0.33
	1	0.64	TPR: 0.36

Comparando as 3 tabelas vemos que o resultado no conjunto de teste foi muito abaixo dos resultados no treino e validação, que foram parecidos entre si. Isso pode indicar que os dados de treino e validação não são bem representativos dos dados do conjunto de teste.

QUESTÃO 5: MELHORANDO O BASELINE

Para começar, balanceamos as classes de acordo com o que foi explicado na questão 2 e obtivemos o resultado apresentado na questão 4. Em seguida, exploramos a combinação de features e a criação de modelos exponenciais. Combinamos todas as features disponíveis de 2 a 2 até 10 a 10. Vale ressaltar aqui que demorou muito para que o RStudio conseguisse treinar todos esses modelos. Para os modelos exponenciais as features foram elevadas individualmente em expoentes de 1 até 10. O resultado da análise pode ser visto no gráfico da Figura 1.

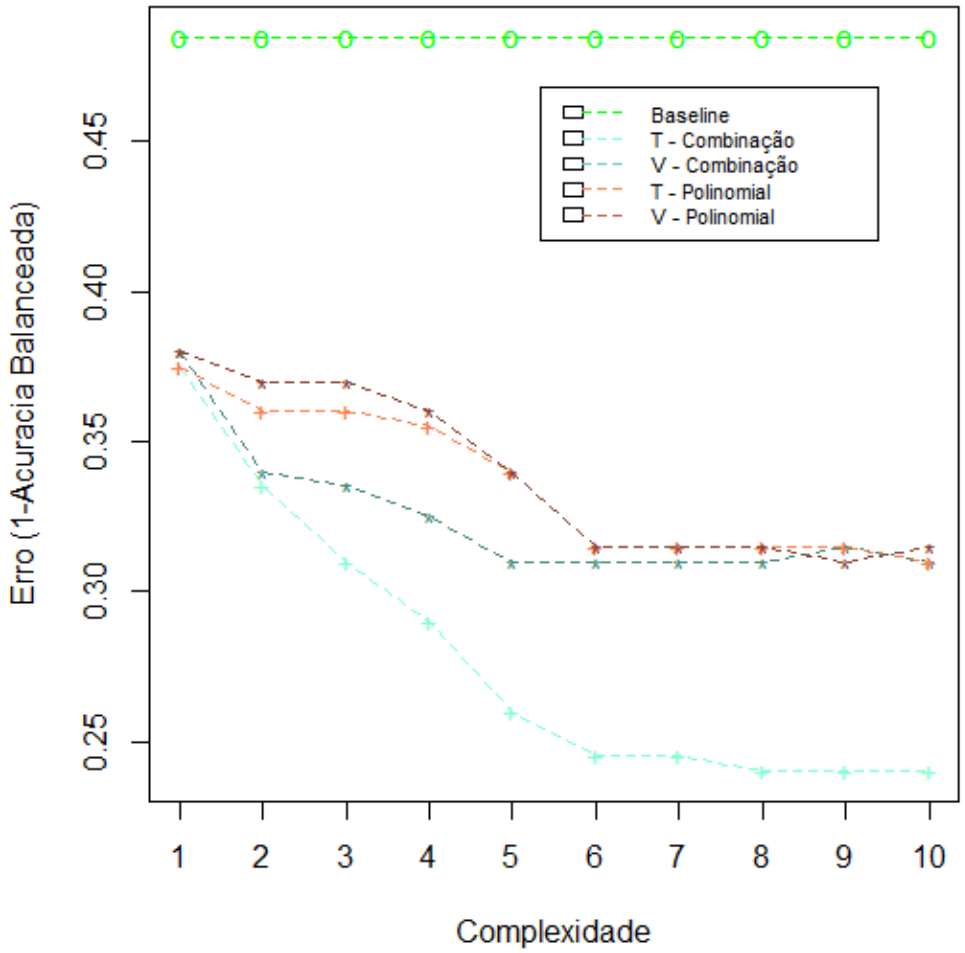

Figura 1. Gráfico de viés e variância dos modelos de combinação de features e polinomial

Junto com o gráfico, devemos olhar para a Tabela 3, que contém a acurácia balanceada calculada a partir da matriz de confusão relativa de cada modelo do gráfico anterior.

Tabela 3: Acurácia Relativa da Validação de cada um dos modelos testados

Modelos Polinomiais									
1	2	3	4	5	6	7	8	9	10
0.62	0.63	0.63	0.64	0.66	0.68	0.68	0.68	0.69	0.68
Modelo Combinatório									
1	2	3	4	5	6	7	8	9	10
0.62	0.66	0.66	0.67	0.69	0.69	0.69	0.69	0.68	0.69

Através da Tabela 3, observamos que os melhores modelos atingem uma acurácia de 0.69. No entanto, observando o gráfico da Figura 1 observamos que o modelo combinatório sofreu um overfitting muito grande a partir do grau de complexidade 3. Sendo assim, optamos por escolher o modelo polinomial de grau 9, que embora tenha um alto grau de complexidade, não obteve Overfitting e ainda assim atingiu o valor máximo de acurácia balanceada. Após balancearmos os dados, rodamos o modelo escolhido no conjunto de teste e obtivemos o seguinte resultado:

Tabela 4: Matriz de Confusão Relativa do Modelo polinomial de grau 9 no conjunto de teste

Modelo: Polinomial 9 Acurácia Balanceada: 0.65		Predição	
		0	1
Referência	0	TNR: 0.66	0.34
	1	0.35	TPR: 0.65

Podemos ver que desta vez o modelo obteve um resultado no teste bem mais parecido com o resultado do treino e validação. Isto pode indicar que o baseline, que apresentou um resultado bem diferente entre validação e teste, não foi capaz de generalizar os dados suficientemente para este conjunto de teste.

QUESTÃO 6: REGULARIZAÇÃO

O modelo escolhido para regularização foi o modelo de combinação de features de grau de complexidade 6. Isto porque ele apresentou um resultado no treino muito bom, porém um resultado na validação muito abaixo, indicando claramente *Overfitting*. Tendo em vista que a regularização é uma técnica para diminuir a complexidade do modelo e evitar Overfitting, acreditamos que este seja um bom modelo para se testar a regularização. No entanto, observando o gráfico de viés e variância que obtivemos aplicando a regularização, observamos que o erro (1 - acurácia balanceada) não melhorou. Para valores altos de lambda, ou seja, uma menor regularização, o erro foi alto tanto para o treino quanto para a validação, indicando Underfitting. Já para valores menores de lambda, o overfitting ocorreu muito rápido, ficando evidente já para lambda = 10e-3.

Na busca por um lambda melhor, afunilamos a nossa busca para valores em torno de 10e-2, que foi a região ótima no exemplo anterior. Obtivemos um resultado melhor, porém não expressivo. Este gráfico é mostrado na Figura 3. Mas podemos observar o intervalo entre o Underfitting e Overfitting de forma mais precisa.

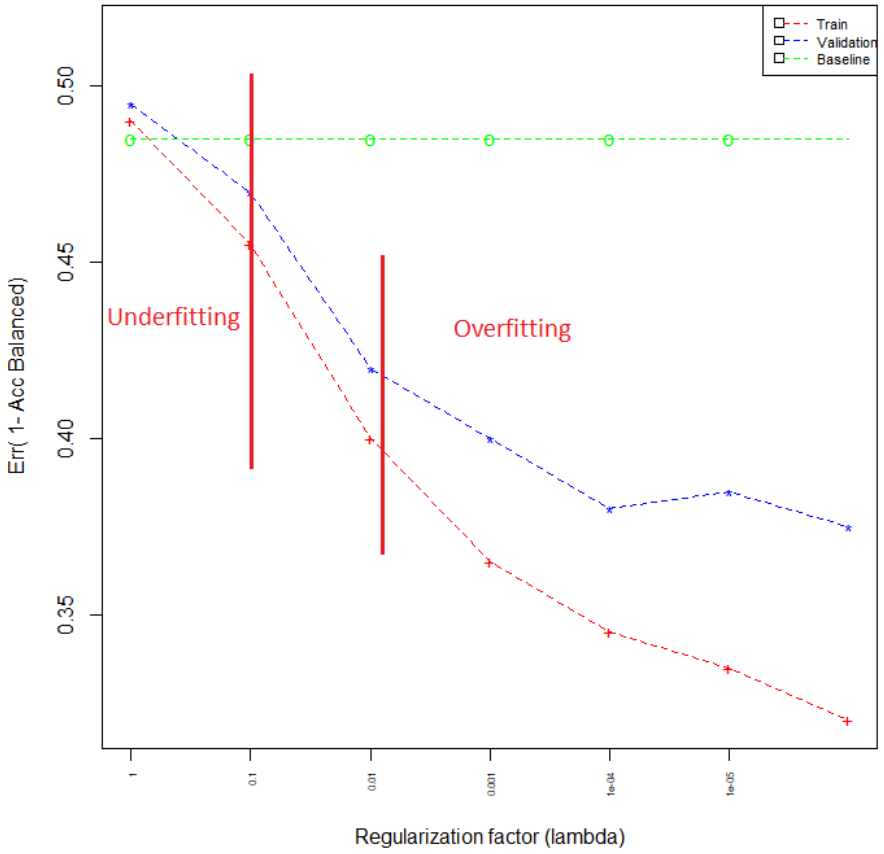

Figura 2. Gráfico de Viés e Variância variando o lambda pelo intervalo de potências de 10

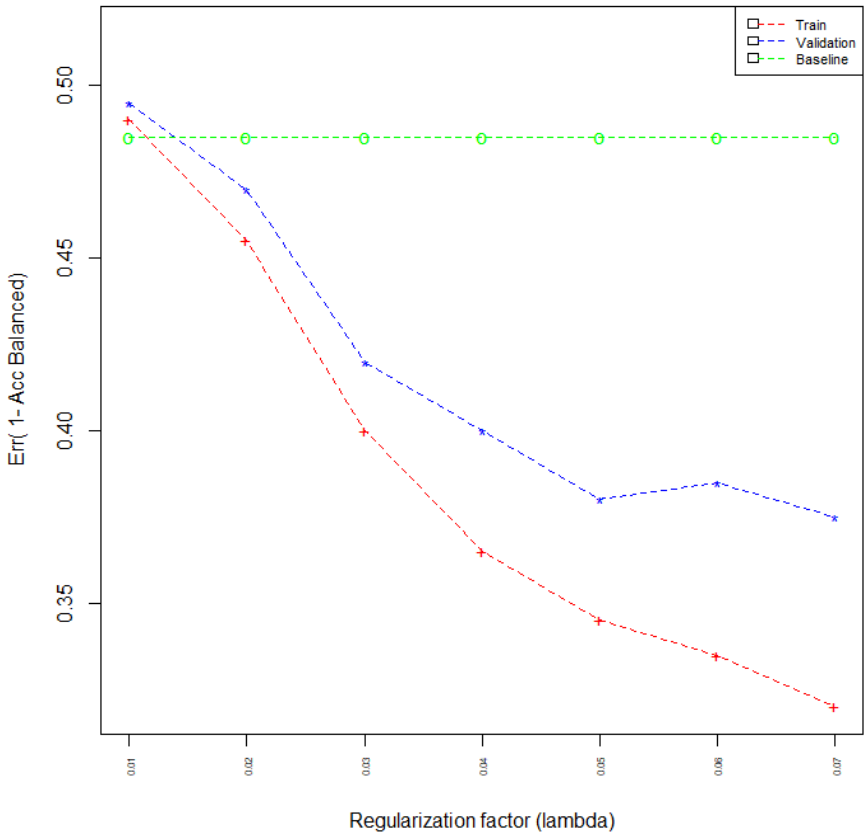

Figura 3. Gráfico de Viés e Variância variando o lambda em um intervalo próximo a 10e-2