

Teste de Software Escola de Artes Ciências e Humanidades

Professor Mestre: Adilson Lopes Khouri

30 de junho de 2019

Sumário

- 1 Contextualização
- 2 Lime
- 3 Manifold
- 4 Shap
- 5 Metamorphic
- 6 Agradecimentos
- 7 Contato

Contextualização

- 1 Classificadores definem uma superfície de decisão para classificar dados (em até n-classes)

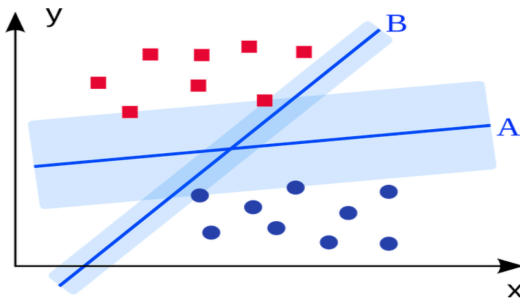


Figura: Exemplo utópico de fronteira de decisão

Contextualização

- 1 Os classificadores escolhem uma das retas azuis possíveis (há infinitas possibilidades) para ser a fronteira de decisão.
- 2 O problema acima é utópico, tipicamente as classes não ficam separadas perfeitamente.

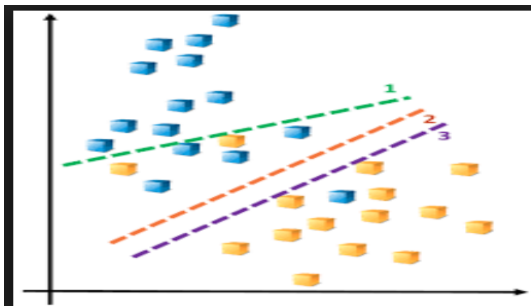


Figura: Exemplo real de fronteira de decisão

Contextualização

- 1 Para definir a fronteira é usado um algoritmo de otimização (que minimiza o erro de classificação)
- 2 **É considerado normal ter erros de classificação dado a natureza não determinística dos algoritmos de Machine Learning**

Contextualização

Quando ocorre um erro de classificação em algoritmos de machine learning podem ter várias razões (não mutuamente exclusivas):

- 1 Dados de treino insuficientes/viesados
- 2 Arquitetura do algoritmo mal planejada
- 3 O algoritmo aprendeu a função errada
- 4 Bug no código fonte
- 5 Variação na distribuição das principais variáveis usadas pelo modelo
- 6 Bug em outros sistemas que alimentam os dados do modelo

Contextualização

- 1 Além dos problemas citados há modelos caixa preta (não é possível entender como o modelo toma decisões[de onde foi criado o score?])
- 2 Regressão logística e CART são exemplos de algoritmos interpretáveis (é possível entender como foi tomada a decisão)

Contextualização

- 1 Dados todos esses problemas citados há linhas de pesquisa para cada um deles.
- 2 Os artigos LIME, SHAP e manifold tentam resolver o problema de quais variáveis são mais relevantes para modelos caixa preta.
- 3 O último artigo trata sobre identificação de bugs em código fonte de algoritmos de Machine Learning

Artigo 01

- 1 “Why Should I Trust You?” Explaining the Predictions of Any Classifier

Lime - overview

- 1 É agnóstico (não assume nada sobre o modelo, dessa forma, funciona para qualquer modelo)
- 2 Realiza uma interpretação Local do score gerado pelo modelo.
- 3 Um ponto falho é que a interpretação foi avaliada usando modelos pequenos (até 10 features) interpretáveis (explicar por que é falho)
- 4 Um ponto fraco da técnica é que a função de distância depende do tipo de dado de input (texto, imagem, numérico...)

Lime - funcionamento

- 1 Escolhe uma instância que deseja interpretar
- 2 Seleciona outras instâncias próximas a ela (usando uma métrica de distância D)
- 3 Usando a amostra executa uma logística para aprender, localmente, quais as variáveis mais importantes

Lime - fronteira de decisão

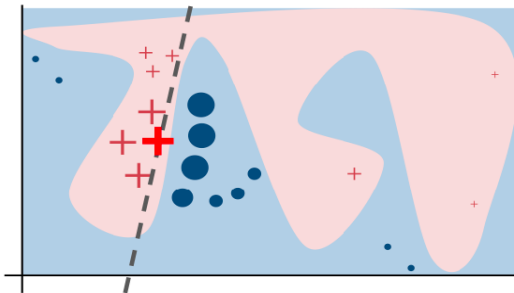


Figura: LIME fronteira de decisão [1]

Artigo 02

- 1 Manifold: A Model-Agnostic Framework for Interpretation and Diagnosis of Machine Learning Models

Manifold - overview

- 1 É um framework para ajudar o cientista a localizar falhas no modelo, para tal, os autores automatizam tarefas típicas para depurar modelos.
- 2 Para tal, os autores criaram uma “matriz de confusão” que relaciona modelos e classes, com essa matriz é possível avaliar visualmente onde os modelos não concordam.

Manifold - overview

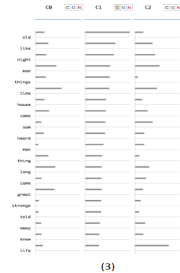
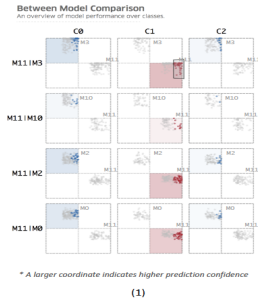


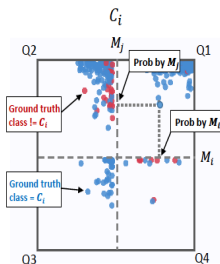
Figura: Visualização sugerida pelo manifold [2]

Manifold - overview

- 1 Após selecionar a célula (tipicamente será escolhida a células de diferença Q1, Q3) é exibida uma comparação da distribuição de variáveis das instâncias daquela célula.
- 2 No próximo gráfico é exibida uma sumarização dos valores (TF-IDF) das features, das instâncias selecionadas, e das instâncias pertencentes a cada classe.
- 3 Pode-se usar o gráfico do meio para avaliar quais features são mais relevantes para classificar uma instância em determinada classe

Manifold - funcionamento

- 1 A comparação de modelos é feita por meio de visualização das probabilidades e classes verdadeiras em uma “matriz de confusão”



	Instances in blue (GT class = C_i)	Instances in red (GT class $\neq C_i$)	Agreement
Q1	Class predicted by $M_i = C_i$ (M_i correct) Class predicted by $M_j = C_i$ (M_j correct) <u>TP for M_i and M_j</u>	Class predicted by $M_i = C_i$ (M_i wrong) Class predicted by $M_j = C_i$ (M_j wrong) <u>FP for M_i and M_j</u>	Agree
Q2	Class predicted by $M_i \neq C_i$ (M_i wrong) Class predicted by $M_j = C_i$ (M_j correct) <u>FN for M_i, TP for M_j</u>	Class predicted by $M_i \neq C_i$ (M_i correct) Class predicted by $M_j = C_i$ (M_j wrong) <u>TN for M_i, FP for M_j</u>	Disagree
Q3	Class predicted by $M_i \neq C_i$ (M_i wrong) Class predicted by $M_j \neq C_i$ (M_j wrong) <u>FN for M_i and M_j</u>	Class predicted by $M_i \neq C_i$ (M_i correct) Class predicted by $M_j \neq C_i$ (M_j correct) <u>TN for M_i and M_j</u>	Agree
Q4	Class predicted by $M_i = C_i$ (M_i correct) Class predicted by $M_j \neq C_i$ (M_j wrong) <u>TP for M_i, FN for M_j</u>	Class predicted by $M_i = C_i$ (M_i wrong) Class predicted by $M_j \neq C_i$ (M_j correct) <u>FP for M_i, TN for M_j</u>	Disagree

Figura: Matriz de confusão usada pelo Manifold [2]

Manifold - funcionamento

- 1 A avaliação de importância de features (segundo gráfico da esquerda para direita) é realizada usando uma métrica como TF-IDF para a classe e para o conjunto de dados classificados
- 2 Os gráficos acima, que não é a distribuição de variáveis mas a *Kullback-Leibler divergence*, indica o quanto a distribuição dos itens selecionados é parecida com a classe em questão
- 3 O último gráfico (da esquerda para direita) mostra a distribuição de features por classe, sumarizadas por TD-IDF

Artigo 03

1 Consistent Individualized Feature Attribution for Tree Ensembles

Shap - overview

- ❶ Não é agnóstico (funciona apenas para ensemble de árvore de decisão)
- ❷ Realiza uma interpretação Local do score gerado pelo modelo.
- ❸ Citam o problema de inconsistência (score alto para variáveis que não são importantes e o caso contrário)
- ❹ Um ponto falho é que o SHAP foi avaliado perguntando para algumas pessoas (os autores não citam esse número) quais features eram mais relevantes.

Shap - funcionamento

- 1 Realiza uma perturbação do dado de entrada, roda um modelo aditivo sobre o dado original e sobre o perturbado.
- 2 Interpreta o modelo aditivo para dizer o quanto a variável foi explicativa para aquela decisão
- 3 A diferença em relação ao LIME é que no modelo aditivo encontro o ϕ (equivalente ao beta da regressão) com probabilidade condicional, a logística usa uma técnica de otimização (e.g. Least square)

SHAP - output

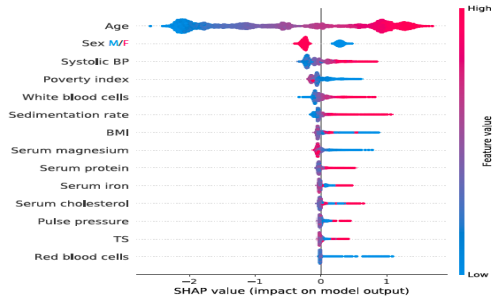


Figura: SHAP - output [3]

Artigo 04

- 1 Identifying Implementation Bugs in Machine Learning Based Image Classifiers using Metamorphic Testing

Metamorphic - overview

- 1 Uso do conceito de teste metamórfico para testar algoritmos de machine learning.
- 2 Solução não agnóstica dado que há necessidade de definir relação metamórfica para cada novo modelo
- 3 Os autores conseguem validar a solução criando bugs artificiais em algoritmos conhecidos de machine learning (como SVM) e aplicando a solução proposta.
- 4 Não há uso de bugs reais para avaliar o algoritmo

Metamorphic - funcionamento

- 1 São definidos relações metamórficas para cada um dos dois algoritmos de machine learning (SVM e Residual Network)
- 2 Os autores alteram o fonte dos algoritmos para chumbar uma semente aleatória fixa para todas as execuções
- 3 As relações metamórficas não devem causar alteração no output e função de perda no decorrer do treino
- 4 Com essas condições asseguradas, se ocorrer uma alteração na função de perda ou output há indício de bug

Metamorphic - exemplos de relações metamórficas

- 1 Alterar a ordem das instâncias de treino
- 2 Alterar a ordem das features de treino
- 3 Adicionar um valor constantes em determinadas features
- 4 Escalar as variáveis

Fim!

Agradeço a atenção

Contato

- E-mail: *adilson.khoury.usp@gmail.com*
- Phone: +55119444 – 26191
- Link LinkedIn
- Link Curriculum Lattes
- GitHub pessoal

Referências

- [1] M. T. Ribeiro, S. Singh, and C. Guestrin, "“why should i trust you?”: Explaining the predictions of any classifier," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 1135–1144. [Online]. Available: <http://doi.acm.org/10.1145/2939672.2939778>
- [2] J. Zhang, Y. Wang, P. Molino, L. Li, and D. S. Ebert, "Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models," *CoRR*, vol. abs/1808.00196, 2018. [Online]. Available: <http://arxiv.org/abs/1808.00196>
- [3] S. M. Lundberg, G. G. Erion, and S.-I. Lee, "Consistent individualized feature attribution for tree ensembles." *CoRR*, vol. abs/1802.03888, 2018. [Online]. Available: <http://dblp.uni-trier.de/db/journals/corr/corr1802.html#abs-1802-03888>
- [4] A. Dwarakanath, M. Ahuja, S. Sikand, R. M. Rao, R. P. J. C. Bose, N. Dubash, and S. Podder, "Identifying implementation bugs in machine learning based image classifiers using metamorphic testing," in *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ser. ISSTA 2018. New York, NY, USA: ACM, 2018, pp. 118–128. [Online]. Available: <http://doi.acm.org/10.1145/3213846.3213858>