Mineração de Dados

Take-Home Exam (Duplicates)

Teste realizado por: Marcelo Feliz nº50356

Duplicates - Consider a high dimensional labeled dataset with a large number of instances. This dataset has several duplicates and near duplicates. Assume you will apply some splitting procedure in order to perform a classification task. Discuss how important it is (or not) to remove the duplicates in the training set; the duplicates in the test set; and the duplicates between train and test sets. IF this is a relevant task, how to do it ?

Para começar vamos definir o que queremos dizer por duplicados, duplicados pode significar duas coisas ligeiramente parecidas: Mais do que uma instância exatamente igual, isto é chamado duplicação exata; por outro lado, mais do que uma instância associada à mesma observação mas os valores não são exatamente os mesmos, ou seja, uma duplicação parcial.

Um dos passos mais importantes ao lidar com "data" é analisar a informação que temos, o que não é nenhuma exceção quando se fala em duplicados já que esta faz parte da análise de dados. Devemos começar por ver quantas linhas estão duplicadas e qual a zona em que as mesmas se encontram, existem 2 razões para isto, nomeadamente: ver a quantidade de dados duplicados. Se existirem poucas linhas duplicadas ou até nenhuma podemos passar à frente sem nos preocuparmos com duplicação. Também é necessário ver se existem padrões, um padrão comum é a existência de blocos duplicados devido a data ter sido copiada para o fim da data existente, se este for o caso podemos simplesmente eliminar o bloco de data.

No caso de o "data set" ser um conjunto de imagens o mesmo se aplica, a utilização de "hashing" em imagens poder ser útil neste caso. Criptografia como MD5 e SHA1 serão suficientes para encontrar duplicados exatos.

Quando temos duplicados é provável que exista um impacto negativo no resultado já que a "training data" irá ter as mesmas instâncias múltiplas vezes de variáveis

dependentes ou independentes, logo quando o modelo aprende com esta data teremos uma grande precisão em testes "in-sample" mas em "out of sample" teremos muito menos, o que nos leva ao "over fitting". Isto deve-se ao facto de que manter dados duplicados afetará a precisão de "cross-validation" já que instâncias idênticas podem existir no "subset" de treino e no "subset" de teste.

No entanto, ter um dataset maior é melhor para o treino, e manter "imagens" (instâncias) iguais com pequenas variações não é necessariamente mau. Duplicados não significam over-fitting, eles simplesmente dão mais peso a essa imagem no treino, por exemplo, se todas as imagens estiverem repetidas 10 vezes então será o mesmo de ter 1 de cada, o maior problema é o potencial data set pequeno de exemplos únicos e também caso se use esse data set também para teste (split), já que provavelmente as imagens que estaremos a testar foram já treinadas, o que pode não ser uma representação desejada.

O que nos leva à última possibilidade do problema (apagar ou não apagar os dados duplicados), caso omitamos os duplicados podemos estar a "estragar" a relação base de cada objeto distinto. Se o treino é uma pequena representação do "mundo real" então provavelmente não iremos querer isso já que iremos estar a treinar para algo um pouco diferente.

Para clarificar o último ponto considere-se o seguinte cenário em que existem 2 objetos distintos. A data original contém 99 objetos A e 1 objeto B. Depois de mandar fora os duplicados teremos 1 objeto A e 1 objeto B. Um classificador treinado na data original será diferente do treinado depois da limpeza, o que pode influenciar a validade do classificador no ambiente real.

Falando agora sobre duplicados no test set, neste caso é bastante raro o benefício de o manter salvo em algumas exceções de "mundo real", o que acontece caso tenhamos duplicados deste modo é a potencialidade para ter resultados totalmente diferentes entre testes, já que se 50% do teste consistir em instâncias iguais entre si o modelo basicamente ao acertar uma das previsões dessa instância ganha automaticamente 50% de respostas corretas, mesmo que falhe todas as outras, o que na realidade sem duplicados daria um valor final de aproximadamente 2%.

Estes últimos pontos mostram que é mais vantajoso manter duplicados no train do que no teste, não só isso como devemos tentar manter os duplicados separados entre train e teste e desta forma não estejamos a dar respostas ao que queremos que o modelo teste.

“Data Augmentation” é algo importante quando se fala em duplicados, já que este método é designado com o objetivo de transformar um modelo mais robusto e não deve em teoria alterar as etiquetas correspondentes. Isto é conseguido dando ao modelo instâncias que foram modificadas ligeiramente sobre uma ou mais transformações (deixando a etiqueta igual). Data Augmentation é mais utilizado quando o data set é pequeno ou poderia ser um pouco mais representativo.

Em suma, existem muitas possibilidades, métodos, preferências e situações, e tendo tudo isso em conta não existe um caminho perfeito. Ao longo da minha resposta tentei responder com os métodos mais utilizados, mas sempre com noção de que o contrário mesmo que mais raro pode existir, ser seguido e ter sucesso. Como podemos ver, tratar de duplicados é uma tarefa com uma influência gigante que deve ser estudado para cada caso aplicacional diferente.

Para responder à última parte da pergunta, em caso de uma grande data set em que usaremos “split” em treino e teste para classificação provavelmente o melhor neste caso é a eliminação dos duplicados no teste e também a eliminação dos duplicados entre treino e teste, neste caso o modelo que for criado em teoria apresentará uma classificação consistente para todos os testes mesmo os “off-sample”. Depois de feito essa parte deve ser feita a análise do resultado e tentar perceber de que forma os suplicados no treino estão a influenciar o modelo (também vai depender da percentagem de duplicados presentes). Para “trabalhar” com duplicados podemos sempre removê-los manualmente, utilizar algum programa que identifique instâncias exatamente iguais (não é difícil de ser criado), no caso de instâncias parecidas podemos usar o WEKA, já que este possui essa funcionalidade, mas existem muitos outros programas.