A feature dependent Naive Bayes approach and its application to the software defect prediction problem 阅读笔记

这篇文章中文名称叫做基于特征的朴素贝叶斯方法及其在软件缺陷预测中的应用。这是一篇软件缺陷预测方面的文章，利用的是机器学习的相关知识来对软件中的缺陷进行预测；这里用的是改进以后的朴素贝叶斯算法。

软件开发生命周期表示软件产品的各个阶段；包括规划、分析、设计、开发和测试阶段，可以是线性（瀑布）模型，也可以是周期性（增量、迭代、敏捷）模型。测试阶段是最重要的阶段之一。在此阶段，将进行软件验证和确认检查，一旦保证产品无缺陷，则批准在实时环境中设置。纠正软件开发过程中发现的缺陷的成本要比客户在实际环境中发现的成本低得多。如果产品是关键任务，在现场环境中发现的缺陷可能会停止操作程序，甚至可能导致致命后果。因此软件缺陷预测是很重要的领域。

机器学习算法辅助软件测试的方法现在很火热。通过这些方法，软件模块，不管它们是否容易出现缺陷，都会被自动分类。因此，开发人员和测试人员团队可以在成本和时间上有效地使用资源。这些分类方法通常被称为软件缺陷预测器。

迄今为止的研究表明，软件质量和缺陷预测之间存在相关性。软件质量由各种提议的度量集来评估，比如LOC,LOCQ,CBO。在软件缺陷预测中，这些度量和软件的缺陷倾向性（缺陷倾向性（dp）或非缺陷倾向性（ndp））分别被视为独立的输入和二进制依赖的输出。

朴素贝叶斯（NB）算法以其简单有效的特点，成为分类问题中应用最广泛的算法之一。它适用于许多学习场景，如图像分类、欺诈检测、web挖掘和文本分类。NB是一种基于特征相互独立且其权重同等重要的假设的概率方法。然而，在实践中，特征可能是相互关联的。在这种情况下，这种假设可能会导致性能的急剧下降。此外，学习器在独立地考虑特征时可能会暴露出一个隐藏的模式。在很多研究中，建议改进NB算法的性能可以分为两类。第一类是半朴素贝叶斯方法，其目的是通过减少条件独立性假设来获得更好的分类性能。另一方面则使用特征加权技术，其中高预测性特征的影响增大，低预测性特征的影响减小。

在这篇文章中，提出一种特征相关的朴素贝叶斯分类方法。特征作为一对包含在计算中，以创建彼此之间的依赖关系。将该方法应用于软件缺陷预测问题并且使用NASA PROMISE数据集进行实验。很明显这篇文章的思路是从第一类的角度—减少条件独立性假设方面来改进算法的。

文章从一开始提出两个问题，以解决这两个问题形成本文的写作思路。

问题1:FDNB的分类性能与标准NB和其他处理NB基本假设的修改NB相比如何？

问题2:不正确的分类基础结构（即在数据预处理中使用整个数据集）如何影响分类性能？

FDNB就是作者提出的基于特征相关的朴素贝叶斯方法。

从本文中先学习到了软件缺陷预测模型的构建方法，之后学习到朴素贝叶斯如何在软件缺陷预测中进行使用。

在创建软件缺陷预测模型时，使用的是另一个人提出的框架。这个框架最重要的原则是学习模型是建立在训练数据集的基础上的，没有包含任何测试数据。这是用新的数据集评价分类器性能的必要前提。预测模型的简要信息如下：将数据预处理步骤应用于训练集，然后将从这些过程中获得的参数用于测试集的预处理。分类器公式中使用的变量是根据预处理的训练数据确定的。根据分类器公式对测试数据中的每个实例进行类估计，最终生成性能报告。根据MxN交叉验证方法重复这些步骤。该方法将整个数据集划分为N个子集，其中一个子集作为测试数据，其余（N-1）个子集作为训练数据。此过程重复N次，以允许每个子集用作测试数据。因此，由样本选择（抽样偏差）引起的误导性结果被最小化。另一方面，为了减少样本顺序的影响，该过程被重复M次以获得统计上可靠的结果，并且在每次迭代中随机选择数据集的排序。本次实验中，使用了200×10交叉验证（M=20和N=10）。

作者提出的基于特征相关的朴素贝叶斯方法是每个特征的出现概率与其他特征一起被考虑在内。由于特征和区间是成对处理的，因此生成四维矩阵。该矩阵的每个维度包括：

第一维度：特征（用作主要特征）；

第二维度：特征的所有间隔值；

第三维度：特征（用作次要特征）；

第四维度：特征的所有间隔值；

文章之后的内容就是实验的过程和实验结果的比较。实验过程也就是编码过程，然后将改进的朴素贝叶斯算法用于特定数据集进行测试，得到实验结果以后将实验结果与其他人的进行比较。最终得到如下结果：

该方法在特征子集选择预处理和其他NB变化加权特征的情况下，优于标准NB方法。此外，FDNB方法与PCA预处理的NB相比表现出了很强的竞争性。FDNB方法是一种新的算法。因此，随着其他研究者的改进，预期会取得更好的结果。

在这项研究中，还表明，学习者模型的不当设计会产生误导性的结果。一个理想的学习者模型应该建立在训练数据的基础上，这些数据是现成的，并且是基于现实的。在预处理步骤中使用所有数据可能会导致高估性能。作者通过改变预处理步骤中使用的数据集的范围来重复第一个实验。在最初的（第一次）实验中，只通过训练数据建立模型，并通过测试数据进行验证。然而，在重复（第二次）实验中，在模型建立（数据预处理）期间使用完整的数据集，并在测试数据集上进行验证。后者的实验结果优于前者，说明了建立学习模型的重要性。

读完这篇文章收获很多，首先是知道了如何建立软件缺陷预测模型，其次是学会了朴素贝叶斯算法的原理和朴素贝叶斯算法在软件缺陷预测中的应用，最后是知道了朴素贝叶斯的改进方案和作者提出的基于特征的朴素贝叶斯在软件缺陷预测中的实验效果。

读完文章感觉还有可以改进的地方：首先可以针对不同的数据预处理步骤，分析这些步骤对FDNB分类器性能的影响。其次在这项研究中，一对特征的作用就好像它们具有相等的权重一样；事实上FDNB方法的性能可以在不同特征权重的情况下进行分析。最后FDNB分类器可利用其他数据库（如UCI机器学习库）应用于不同的问题领域，看看作者针对特定问题提出的改进算法能否在其他问题领域起到好的效果。