UNIVERSIDADE FEDERAL DE VIÇOSA
CAMPUS VIÇOSA
CIÊNCIA DA COMPUTAÇÃO

ELCIO PEREIRA DE SOUZA JUNIOR

TRABALHO DE MINERAÇÃO DE DADOS: TÉCNICAS DE APRENDIZADO SUPERVISIONADO SOBRE UM DATASET

VIÇOSA
2018

Resumo

O objetivo deste trabalho é praticar os conceitos de aprendizado supervisionado que foram vistos em sala. Neste projeto, foram utilizadas as principais técnicas de classificação juntamente com a API sklearn (scikit-learn), a fim de comparar os resultados obtidos sobre um dataset inicialmente desconhecido. A linguagem de programação escolhida para este trabalho foi o Python3.

Palavras-chaves: Mineração, Dados, Aprendizado, Supervisionado, Dataset.

Lista de abreviaturas e siglas

UFV	Universidade Federal de Viçosa
SVM	Support Vector Machines
RF	Random Forest Classifier
NB	Naive Bayes (Bernoulli)
KNN	K-Nearest Neighbors
DT	Decision Trees

Sumário

1 Introdução

O que uma simples ida ao mercado e observações de corpos celestes têm em comum? A geração de enormes bases de dados, e não são apenas estas atividades que tanto se diferem em sua prática que tem esta habilidade. Atualmente vivemos em uma sociedade conectada, onde cada ação está ou pode ser armazenada, e foi sobre este crescente volume de dados, que se faz necessária a utilização de técnicas tanto computacionais como estatísticas para a extração de conhecimento. Em meio a um emaranhado de informações, pesquisas científicas e grandes conglomerados comerciais se beneficiam por meio da Mineração de Dados.

2 Objetivos

Este trabalho é parte do conjunto de avaliações da disciplina de Mineração de Dados da turma de 2018/2, e tem por objetivo uma abordagem prática dos conceitos abordados em sala de aula ao longo do semestre letivo.

2.1 Objetivos Específicos

O objetivo inicial deste trabalho é a extração de conhecimento (predição) sob um dataset inicialmente desconhecido (origem/informação), utilizando-se apenas dos métodos que foram discutidos na disciplina de Mineração de Dados - INF493, para os modelos que foram implementados, como não era disponível informações sob o domínio dos dados, fez-se necessária apenas a utilização de métricas como f1 score e acurácia para validação e comparação entre os modelos, estes serão apresentadas a frente com mais detalhes. Os códigos foram implementados na linguagem Python3 e foi utilizada a API Sklearn desenvolvida especificamente para análise e mineração de dados. O dataset juntamente com os códigos, que são mencionados, podem ser adquiridos na página <https://github.com/Necropsy/INF493-Homework2.git>.

3 Métodos

Neste capítulo, apresentaremos as estratégias empregadas e que demonstraram os melhores resultados junto ao dataset fornecido. As características básicas de cada método serão apenas mencionadas neste trabalho, visto que o detalhamento das mesmas não faz parte deste escopo. Para mais informações a respeito de cada uma destas, as referências serão fornecidas.

Observação: *Ao final deste trabalho será apresentada uma tabela com os melhores resultados obtidos por cada uma das técnicas juntamente com os parâmetros utilizados.*

3.1 Análise e Tratamento dos Dados

O dataset fornecido é composto por 4 arquivos onde 2 apresentam o conjunto de treinamento (*train_X.csv* e *train_y.csv*), um o conjunto de dados que deve ser predito (*test_X.csv*) e outro com um exemplo de como a resposta deve ser apresentada (*test_example_y.csv*). Antes de aplicar os métodos, foram necessários alguns ajustes em relação a representação dos dados dentro de nosso sistema, algumas *features* são do tipo numérico e outras categóricas, além destas questões, alguns dos dados numéricos divergiam de outros em ordem de grandeza, o que pode influenciar os classificadores. Para resolver estes problemas, os dados categóricos do nosso conjunto foram binarizados e após isto, todos os dados foram normalizados (os id's também foram removidos). Em seguida, após a binarização e a linearização, foram selecionadas em nosso conjunto de teste 75% das amostras para treinamento e os outros 25% para validação dos classificadores. A seguir, cada uma destas secções (sobre os métodos aplicados), discutirá um ponto de vista pessoal em relação ao desempenho do método e como este se portou em relação ao dataset.

3.2 SVM - Support Vector Machines

Um das primeiras técnicas utilizadas foi o SVM. Algumas de suas vantagens são (*dados removidos da página oficial da API*):

- Eficaz em espaços dimensionais elevados;
- Ainda é eficaz nos casos em que o número de dimensões é maior que o número de amostras;
- Usa um subconjunto de pontos de treinamento na função de decisão (chamados vetores de suporte), portanto, também é eficiente em termos de memória;

- Versátil: diferentes funções do Kernel podem ser especificadas para a função de decisão. Os kernels comuns são fornecidos, mas também é possível especificar kernels customizados.

Dentre as características citadas como positivas, podemos destacar alguns pontos negativos que foram observados durante a implementação do mesmo mediante ao dataset utilizado. Foi possível perceber que dentre todos os métodos, o SVM apresentou menor performance. Acreditamos que parte desta desvantagem do SVM em relação aos outros, se deve pelo fato da dimensão deste conjunto de dados não ser tão elevada, assim como o número de amostras bem elevado em relação ao número de dimensões.

3.3 DT - Decision Tree

Uma das técnicas mais interessantes que foram aplicadas neste trabalho, originando a idéia de se usar a técnica Random Forest. Uma idéia interessante que surgiu durante este experimento foi, a utilização do formato de visualização dos dados (disponíveis pela árvore de decisão), a fim de determinar quais *features* tinham maior relevância em comparação com as demais, de posse destas informações, seria possível determinar novos pesos para estes dados, ou até mesmo remover os outros com o intuito de simplificar o modelo.

Algumas vantagens das árvores de decisão são:

- Simples de entender e interpretar;
- Requer pouca preparação de dados;
- O custo de usar a árvore (ou seja, prever dados) é logarítmico no número de pontos de dados usados para treinar a árvore;
- Capaz de lidar com dados numéricos e categóricos;
- Capaz de lidar com problemas de várias saídas;
- Usa um modelo de caixa branca;
- Possível validar um modelo usando testes estatísticos. Isso torna possível explicar a confiabilidade do modelo.
- Funciona bem, mesmo que suas suposições sejam de algum modo violadas pelo verdadeiro modelo do qual os dados foram gerados.

3.4 RF - Random Forest

Motivado anteriormente pelo uso da técnica Decision Tree, foi implementado o método Random Forest, que superficialmente é uma generalização do DT. Uma ressalva deve ser feita em relação a esta estratégia, para florestas muito densas, o consumo de memória na máquina será consideravelmente elevado.

3.5 NB - Naive Bayes

Este é baseado na aplicação do teorema de Bayes com a suposição *"ingênua"* de independência entre cada par de recursos. Este se apresentou bem, seu tempo de execução aparenta ser baixo, porém, algumas das métricas ainda se demonstraram baixas em relação às já coletadas anteriormente.

4 Conclusão

De posse da *Tabela 1*, onde são apresentados os resultados, é perceptível a vantagem do uso da técnica de Random Forest nesta base de dados, essa foi a que melhor se adaptou ao dataset em nossos testes. Na Tabela 2 temos a definição dos parâmetros que foram utilizados para obtenção destes dados, assim como o notebook que acompanha este documento apresenta toda a implementação até aqui mencionada, como também alguns dados adicionais em relação o dataset e os classificadores.

Observação: Cada uma das funções aqui mencionadas, como a utilização de seu parâmetros, podem ser vistas em funcionamento junto ao notebook python que acompanha este trabalho.

Tabela 1 – Tabela de Resultados

Método	**Predição**	**Precision**	**F1_Score**	**Acurácia**
SVM	0.71990	0.47449	0.57199	0.82558
Decision Tree	0.61341	0.62955	0.62138	0.81156
Random Forest	***0.73783***	***0.63806***	***0.68432***	***0.85541***
Naive Bayes	0.53078	0.79231	0.63570	0.77695

Fonte: Próprio Autor

Tabela 2 – Tabela de Parâmetros

Método	**Parâmetros**
SVM	kernel="rbf", C=0.1, degree=2
Decision Tree	-
Random Forest	n_estimators=10000, criterion='entropy', warm_start=True
Naive Bayes	alpha=0.01, binarize=0.4, class_prior=None, fit_prior=True

Fonte: Próprio Autor

Tabela 3 – Matriz de Confusão

	Target	
Predict	True	False
True	22662	1
False	1	7502

Fonte: Próprio Autor

Referências

DEVELOPERS scikit-learn. **API Reference**. [S.l.], 2018. Disponível em: <http://scikit-learn.org/stable/modules/classes.html#api-reference>.