UNIVERSIDADE FEDERAL DE VIÇOSA
DEPARTAMENTO DE INFORMÁTICA
INF 493 - MINERAÇÃO DE DADOS

Relatório
Trabalho Prático 2

Autores: Bruno Conceição do Nascimento - 81830
Lucas Bissaro - 78567

Introdução

Com o avanço no processamento de máquinas nos últimos anos há uma grande quantidade de informação na internet e sob posse de muitas empresas, informações que vão desde imagens de diferentes tipos de animais a dados sigilosos de clientes. Em termos comerciais, a importância dos dados nas empresas aumentou significativamente, pois, através deles têm-se conseguido obter informações relevantes relativas ao seu negócio, clientes, estratégias de mercado, entre outras informações. Esta mudança se deve ao avanço das tecnologias de armazenamento de dados, que são capazes de armazenar uma grande quantidade dos mesmos por um preço acessível se comparadas às tecnologias utilizadas no passado.

Através da necessidade de extrair informações desses dados são utilizadas diversas técnicas de Aprendizado de Máquina onde os algoritmos podem aprender com erros e acertos, assim fazendo posteriormente previsões sobre estes dados.

Uma das várias técnicas é a Classificação de Dados que é uma subcategoria da Aprendizagem Supervisionada (utiliza dados de entrada e suas saídas esperadas para aprender uma regra geral). É amplamente utilizada em diversos problemas reais, tais como: reconhecer padrões em imagens, diferenciar espécies de plantas, classificar tumores benignos e malignos, dentre outros. Este problema é um dos tópicos mais ativos na área de Aprendizado de Máquina. Basicamente, o problema de classificação consiste em determinar o rótulo de algum objeto, baseado em um conjunto de atributos extraídos do mesmo. Para que isso ocorra é necessário um conjunto de treinamento com instâncias na qual os rótulos os objetos são conhecidos. Formalmente, a classificação de dados pode ser definida da seguinte forma:

Dado um conjunto de entradas $X_i = \{x_1, ..., x_a\}$ com N instâncias, sendo $i = \{1, ..., N\}$ e a o número de atributos de X, e um conjunto de rótulos de classificação $R = \{r_1, ..., r_b\}$, com $b \geq 2$, é montado um conjunto de dados de treinamento $T = \left\{ \left(X_1, c_1 \right), ..., (X_i, c_i) \right\}$ no qual cada entrada X_i é vinculada a um rótulo $c_i \in R$. O objetivo de um algoritmo de classificação é aprender, a partir de T, uma correlação entre os atributos de entrada tal que:

Dado uma nova entrada não rotulada $X' = \{x'_1, ...x'_a\}$, o classificador seja capaz de determinar o rótulo vinculado a ela.

O objetivo do trabalho realizado é a utilização de algoritmos para classificação de dados com intuito de treinar modelos capazes de classificar satisfatoriamente um conjunto de dados fornecido.

Metodologia

Utilizamos o Jupyter Notebooks que é uma aplicação Web voltada para a visualização de dados e resultados de análise, juntamente com as bibliotecas Pandas (biblioteca para análise de dados) e Sklearn (biblioteca de aprendizado de máquina). O trabalho foi implementado na linguagem Python 2.7.

Realizamos o treinamento de modelos capazes de classificar o conjunto de dados fornecidos para o trabalho, estes que estão separados em conjuntos de treinamento e teste, ambos possuem 14 campos rotulados de a até n sendo que os campos b, d, f, g, h, i, j e n são atributos categóricos e os demais são numéricos. Cada conjunto está vinculado a um outro conjunto que possui as suas respectivas classificações esperadas.

Inicialmente dividimos o conjunto de dados fornecido para treinamento em duas partes, sendo que 85% dos dados foram usados para treinamento e 15% para teste. Os dados foram selecionados aleatoriamente de forma a evitar sempre escolher um mesmo conjunto de dados, o que poderia atrapalhar a generalização do modelo criado pelo classificador.

Depois da divisão dos dados utilizamos o algoritmo Floresta Aleatória para realizar a classificação, ele possui todos os hiperparâmetros de uma árvore de decisão e também todos os hiperparâmetros de um classificador de *bagging* (o algoritmo utiliza bagging com Árvores de Decisão), para controlar a combinação de árvores. O algoritmo adiciona aleatoriedade extra ao modelo, quando está criando as árvores, ao invés de procurar pela melhor característica ao fazer a partição dos nós, ele busca a melhor característica em um subconjunto aleatório, o que cria uma grande diversidade e resulta em modelos melhores.

Juntamente com o algoritmo Floresta Aleatória, utilizamos o k-fold para obter os melhores parâmetros para o algoritmo. O k-fold é um método de validação cruzada que divide o conjunto de teste em k partes e faz o treinamento nas k-1 restantes, depois o modelo gerado é testado na parte k que não foi utilizado para o treinamento, esse processo se repete k vezes e ele retorna todos os valores de acurácia obtidos (k valores) e a média desses valores, essa média indica a acurácia obtida pelo método. Na implementação foi utilizado o GridSearch da biblioteca Pandas, ele utiliza o k-fold e retorna o modelo com os melhores parâmetros. Utilizamos k = 4 e foram feitas comparações com vários valores para os parâmetros, como o custo computacional aumenta à medida que aumentamos o número de parâmetros para ser testado, deixamos no código os melhores parâmetros encontrados em execuções anteriores para que a execução do GridSearch não demore muito.

Testamos o melhor modelo obtido com o conjunto de dados de teste e conseguimos os resultados que serão exibidos na próxima seção.

Resultados

Para uma melhor análise da qualidade do modelo, criamos uma matriz de confusão para apresentação dos resultados.

A matriz de confusão é uma tabela que permite a visualização do desempenho de um algoritmo. Cada linha da matriz representa as instâncias em uma classe prevista, enquanto cada coluna representa as instâncias em uma classe real (ou vice-versa).

Os dados de uma matriz de confusão representam o valor de cada uma das seguintes nomenclaturas:

- **Verdadeiros Positivos (VP):** Amostras preditas corretamente para o positivo.
- **Falsos Positivos (FP):** Amostras preditas erroneamente para o positivo.
- **Falsos Negativos (FN):** Amostras preditas erroneamente para o negativo.
- **Verdadeiros Negativo (VN):** Amostras preditas corretamente para o negativo.

Predição / Classe	**P**	**N**
P	VP	FP
N	FN	VN

Abaixo temos a matriz de confusão gerada com a comparação dos resultados obtidos pelo modelo em relação aos resultados esperados para o conjunto de dados de teste.

Predição / Classe	**P**	**N**
P	876	377
N	637	4144

Através da matriz gerada obtivemos as seguintes métricas para medir a qualidade do modelo:

Acuracia	0.8382499171362281
Precision	0.707680250783699
Recall	0.599601593625498

F1 Score	0.6491732566498922
Quantidade de verdadeiros positivos em relação a quantidade total:	0.8214709371293001

Acuracia: Mede a proporção de predições corretas, sem levar em consideração o que é positivo e o que é negativo. Esta medida é altamente suscetível a desbalanceamentos do conjunto de dados e pode facilmente induzir a uma conclusão errada sobre o desempenho do sistema.

ACURÁCIA = TOTAL DE ACERTOS / TOTAL DE DADOS NO CONJUNTO

ACURÁCIA = (VP + VN) / (P + N)

Precision: Identifica quantas amostras foram classificadas positivamente. É uma medida do quão exato é a classificação para as amostras positivas.

PRECISION = ACERTOS POSITIVOS / TOTAL DE PREDIÇÕES POSITIVAS

PRECISION = VP / (VP + FP)

Recall: Tem a mesma ideia da *precision,* porém para as amostras falsas negativas.

RECALL = ACERTOS POSITIVOS / TOTAL DE VALORES REALMENTE POSITIVOS.

RECALL = VP / (VP + FN)

F1 Score: É definido como duas vezes a média harmônica entre *R e P*.

F1 = 2 * (PRECISION * RECALL) / (PRECISION + RECALL)

Conclusão

Com o objetivo de treinar um modelo capaz de fazer boas classificações para um dado conjunto de dados, utilizamos o algoritmo Floresta Aleatória juntamente com a classificação cruzada para esse fim.

Utilizamos a Floresta Aleatória por ser um dos melhores algoritmos para a classificação de um conjunto de dados, o que é confirmado com as boas métricas obtidas para o modelo gerado por ele. A validação cruzada além de evitar overfitting também fornece as melhores escolhas de parâmetros para o modelo, a junção desta com o algoritmo agrega de forma a aumentar a capacidade de construir bons modelos.

Com base nos dados apresentados podemos concluir que conseguimos resultados satisfatórios com o modelo obtido, que se mostra está bem generalizado e sem overfitting, capaz de gerar boas classificações com qualquer conjunto de dados para serem rotulados.

Referências

https://medium.com/machina-sapiens/o-algoritmo-da-floresta-aleat%C3%B3ria-3545f6babdf8

http://scikit-learn.org/stable/supervised_learning.html