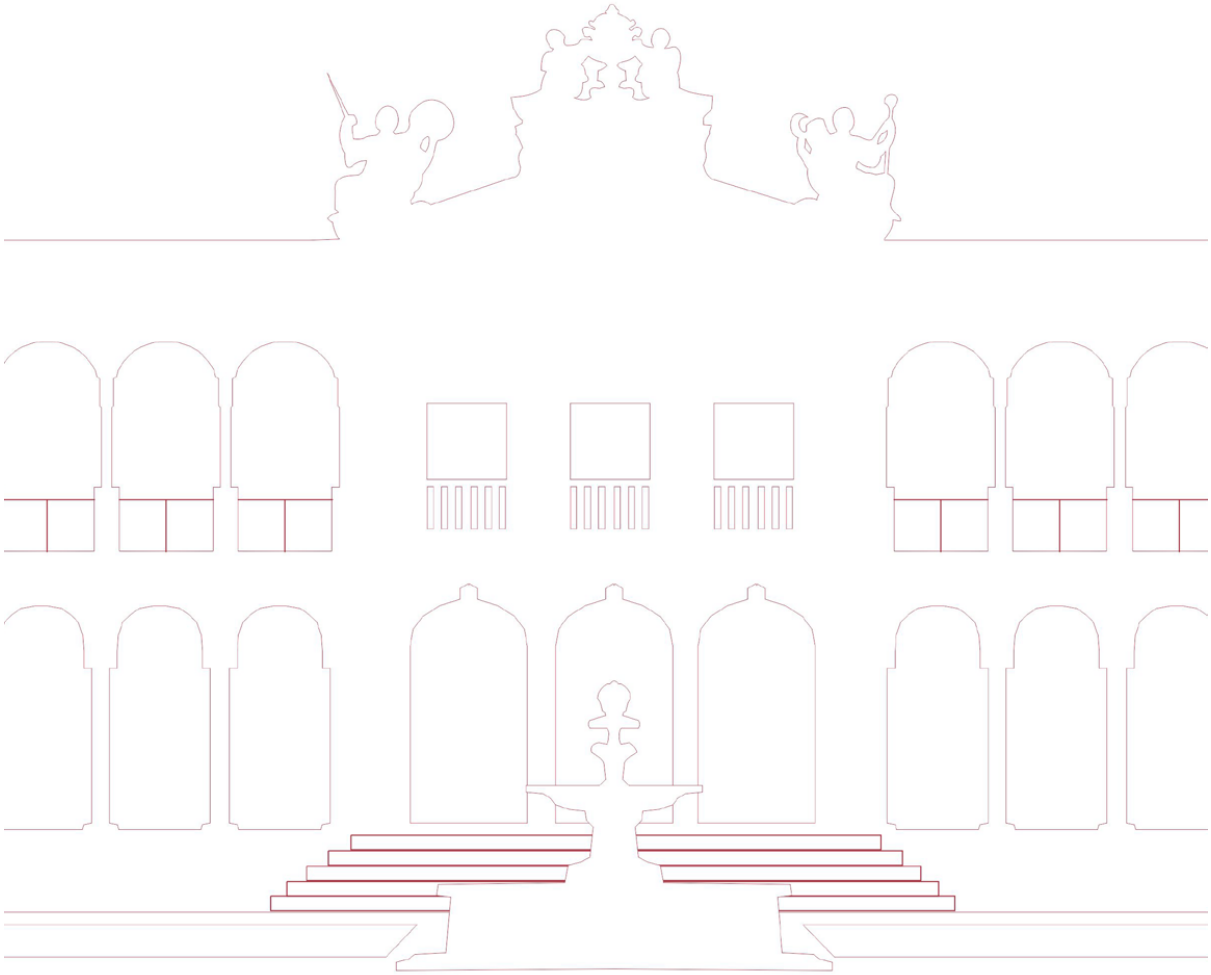
Dataset - Student Dropout

Licenciatura em Engenharia Informática
Aprendizagem Automática

UNIVERSIDADE DE ÉVORA
ESCOLA DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA



Ricardo Marques Oliveira (42647)

Docente: Luís Rato

Évora, 27 de julho de 2023

1 Introdução

Neste trabalho foi proposta a implementação de um modelo preditivo para verificar a quantidade de alunos em risco de abandonar os estudos. Através de um conjunto de dados previamente fornecido, o qual armazena o histórico académico de cada aluno, nomeadamente todas as etiquetas necessárias para o cálculo da precisão e da cobertura. Tendo assim, como objetivo, a utilização dos diferentes algoritmos de modo a analisar os resultados calculados da precisão e cobertura de cada um dele a fim de determinar qual o algoritmo que melhor desempenho tem perante o modelo.

Para tal, foram testados os algoritmos *KNN*, *Naïve Bayes*, *Bagging*, *Random Forest*, *Extra Tree* e *Gradient Boosting*, através da biblioteca de Python *scikit-learn* juntamente com as bibliotecas *NumPy* e *Pandas* para possibilitar operações em estruturas de dados (nomeadamente *arrays*), bem como a manipulação e análise de dados.

2 Algortimos

2.1 KNN

O algoritmo K vizinhos-mais-próximos escolhe uma instância de teste, dado um conjunto de dados e encontra os K vizinhos mais próximos a essa instância escolhida. Atribui a classe que ocorre com maior frequência entre os K vizinhos.

2.2 Naïve Bayes

O algortimo de Naïve Bayes baseia-se no teorema de Bayes, abordado também nas aulas teóricas. Inicialmente assume que uma ocorrência é independente das outras, aplica posteriormente o teorema de modo a determinar a probabilidade de cada uma, selecionando a ocorrência com maior probabilidade.

2.3 Random Forest

O algoritmo Random Forest gera, aleatoriamente, diversas árvores de decisão. Realiza a previsão individual de cada uma dessas árvores e, finalmente, combina o resultado delas, calculando a previsão final.

2.4 Extra Trees

O algortimo Extra Tress, à semelhança do anterior, gera diversas árvores de decisão, diferenciando apenas no método de aleatoriedade aplicado.

2.5 Gradient Boosting

O algoritmo Gradient Boosting constrói modelos de previsão fracos, gerando uma corrente de modelos, onde tenta ajustar cada um deles, de acordo com o anterior, de modo a minimizar o erro anterior. Ou seja, iteriza até alcançar um valor com a menor diferença possível entre o valor real e o valor de previsão.

2.6 Bagging

O algortimo Bagging ajusta classificadores em subconjuntos aleatórios do conjunto de dados e, em seguida, calcula as suas previsões individuais, de modo a formar uma previsão final.

3 Análise

Para ser selecionado o algortimo que melhor prevê os resultados do *dataset* fornecido, testou-se esse mesmo *dataset* com todos os algortimos apresentados anteriormente. Assim que extraídos todos os resultados dos diversos algoritmos, foram cuidadosamente analisados de modo a concluir que algortimo se destacava entre todos. Sendo que o objetivo era maximizar a cobertura, tendo em conta que a precisão era garantida a, pelo menos, 70%, comparou-se os resultados de ambas para cada um.

3.1 Resultados obtidos

3.1.1 KNN

Para K=1:

```
Conjunto de teste - cobertura para a classe positiva: 1.00
Conjunto de teste -  precisão para a classe positiva: 1.00
```

Para K=3:

```
Conjunto de teste - cobertura para a classe positiva: 0.81
Conjunto de teste -  precisão para a classe positiva: 0.96
```

Para K=5:

```
Conjunto de teste - cobertura para a classe positiva: 0.74
Conjunto de teste -  precisão para a classe positiva: 0.94
```

Para K=10:

```
Conjunto de teste - cobertura para a classe positiva: 0.54
Conjunto de teste -  precisão para a classe positiva: 0.91
```

3.1.2 Naïve Bayes

```
Conjunto de teste - cobertura para a classe positiva: 0.94
Conjunto de teste -  precisão para a classe positiva: 0.51
```

3.1.3 Random Forest

```
Conjunto de teste - cobertura para a classe positiva: 1.00
Conjunto de teste -  precisão para a classe positiva: 1.00
```

3.1.4 Extra Trees

```
Conjunto de teste - cobertura para a classe positiva: 1.00
Conjunto de teste -  precisão para a classe positiva: 1.00
```

3.1.5 Gradient Boosting

```
Conjunto de teste - cobertura para a classe positiva: 0.96
Conjunto de teste -  precisão para a classe positiva: 1.00
```

3.1.6 Bagging

```
Conjunto de teste - cobertura para a classe positiva: 0.99
Conjunto de teste -  precisão para a classe positiva: 1.00
```

Com estes resultados, conclui-se que, para os testes executados, somente o *Naïve Bayes* não consegue garantir uma precisão de 70%, logo excluímos esta opção. Tendo em conta que todos os restantes garantem a precisão mínima, maximizamos a cobertura. Logo, os algoritmos que classificam com maior precisão são *KNN* (com K=1), *Random Forest* e *Extra Tree*.

4 Conclusão

Dos três algoritmos anteriormente mencionados, uma vez que apresentam uma precisão e cobertura excelente, é quase indiferente a escolha tomada para este modelo de previsão, porém temos que ter em conta que os algortimos *Random Forest* e *Extra Tree* apresentam melhor efeciência para conjuntos de dados maiores, tendo em conta que este *dataset* contem 2110 instâncias, poderão apresentar benefícios de otimização de execução.