Supervised Learning : Algerian Forest Fires

Ana Carneiro – up202008569

Hugo Almeida - up202006814

Inês Silva – up202008076

Inteligência Artificial - 2022/2023

Especificação do trabalho

- O tema selecionado foi “Fogos Florestais na Argélia”.
- Foi-nos disponibilizado um conjunto de dados que contém vários detalhes tais como a data de captura de dados, a temperatura máxima desse dia, a humidade relativa, velocidade do vento, precipitação(mm), índice de humidade dos combustíveis finos(FFMC), índice de húmus(DMC), índice de seca(DC), índice de propagação inicial(ISI), índice de combustível disponível(BUI), índice meteorológico de incêndio(FWI).
- Por fim, uma indicação se nessas condições ocorreu ou não um incêndio florestal.
- O objetivo deste projeto é treinar vários modelos de modo a possibilitar o reconhecimento da possível ocorrência de um incêndio baseado nas condições do ambiente e de ocorrências passadas.

Trabalhos relacionados

- Previsão de Fogos Florestais na Algéria utilizando técnicas de Data Mining: Caso de Estudo do Algoritmo Árvore de Decisão
- https://www.researchgate.net/profile/Faroudja-Abid/publication/339062373_Predicting_Forest_Fire_in_Algeria_Using_Data_Mining_Techniques_Case_Study_of_the_Decision_Tree_Algorithm
- Machine Learning
- https://www.ibm.com/topics/machine-learning
- GitHub com projeto similar
- https://github.com/geeky-bit/SVR--Decision-Trees--Random-Forests--DeepNeuralNets--to-PREDICT-FOREST-FIRES

Ferramentas e algoritmos a utilizar

- A linguagem utilizada para desenvolvimento foi Python e o ambiente adotado foi Jupyter Notebook.
- Ferramentas utilizadas:
  - Pandas – análise e preparação de dados
  - Sklearn – criação, treino e teste da performance dos modelos
  - Matplotlib – criação de charts
- Algoritmos implementados:
  - Decision Tree: Constrói um modelo de árvore para tomar decisões baseadas nas características dos dados.
  - Neural Networks: Emita a estrutura e o funcionamento do cérebro de um humano de modo a possibilitar a aprendizagem e classificação de dados.
  - K-NN: Atribui uma instância de teste a uma classe baseada nas K instâncias de teste mais próximas no espaço de características.
  - SVM: Constrói Hiper planos de modo a separar as classes e maximizar a margem entre elas.

Pré-processamento de Dados

- Antes de tudo, temos de efetuar o pré-processamento dos dados fornecidos, de modo a aumentar a precisão da análise dos mesmos, reduzir o tempo e os recursos necessários para treinar o modelo, etc.
- Começamos por verificar que a linha 122 separava os dados em 2 regiões diferentes, então decidimos adicionar uma nova coluna que definia em qual região cada conjunto de dados pertencia.
- Em seguida, removemos a linha 122 e todos os valores nulos e verificamos se existiam dados duplicados.

Pré-processamento de Dados

- Depois, removemos os espaços extra presentes no título de algumas colunas e dos atributos "fire" e "not fire" da coluna "Classes".
- Por fim, convertemos todos os dados do tipo Object para o tipo int ou float conforme o que fazia mais sentido, convertendo também os dados da coluna "Classes" para 1 no caso de ocorrência de fogo e 0 no caso contrário.
- Em seguida iniciamos a fase de análise de dados.

Pré-processamento de Dados

Análise de dados

Depois de verificar que os dados estavam em ordem, procedemos para uma análise mais profunda dos mesmos, começando por criar uma matriz de correlação para verificar a influência de cada característica do ambiente na ocorrência de um fogo.

Para além disso, criamos também vários gráficos que permitem comparar a quantidade de dados que geraram fogos ou não, a quantidade de fogos que ocorreram em cada região e em cada mês.

Preparação de dados

Em seguida, de modo a minimizar os dados removemos algumas colunas que concluímos não serem tão relevantes para a classificação e também as que possuíam correlação superior ao threshold definido.

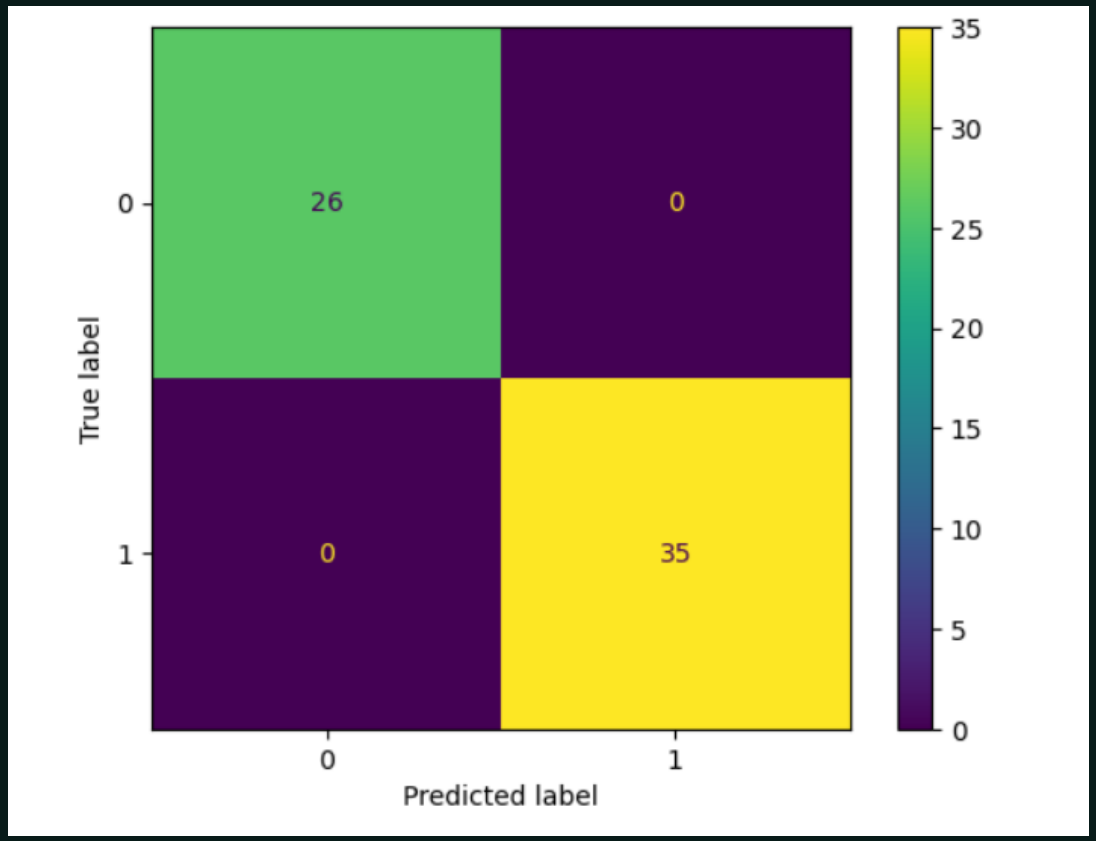
Depois, dividimos a base de dados em treino e teste (75%-25%) e efetuamos *scaling* da base de dados, de modo a normalizar os dados e atribuir igual importância às diferentes features durante o processo de treino.

Decision Tree

- Tempo para treinar Decision Tree classifier: 0.00294 segundos.
- Métricas de avaliação:

Decision Tree
Accuracy Obtained: 1.00000

	precision	recall	f1-score	support
0	1.00	1.00	1.00	26
1	1.00	1.00	1.00	35
accuracy			1.00	61
macro avg	1.00	1.00	1.00	61
weighted avg	1.00	1.00	1.00	61

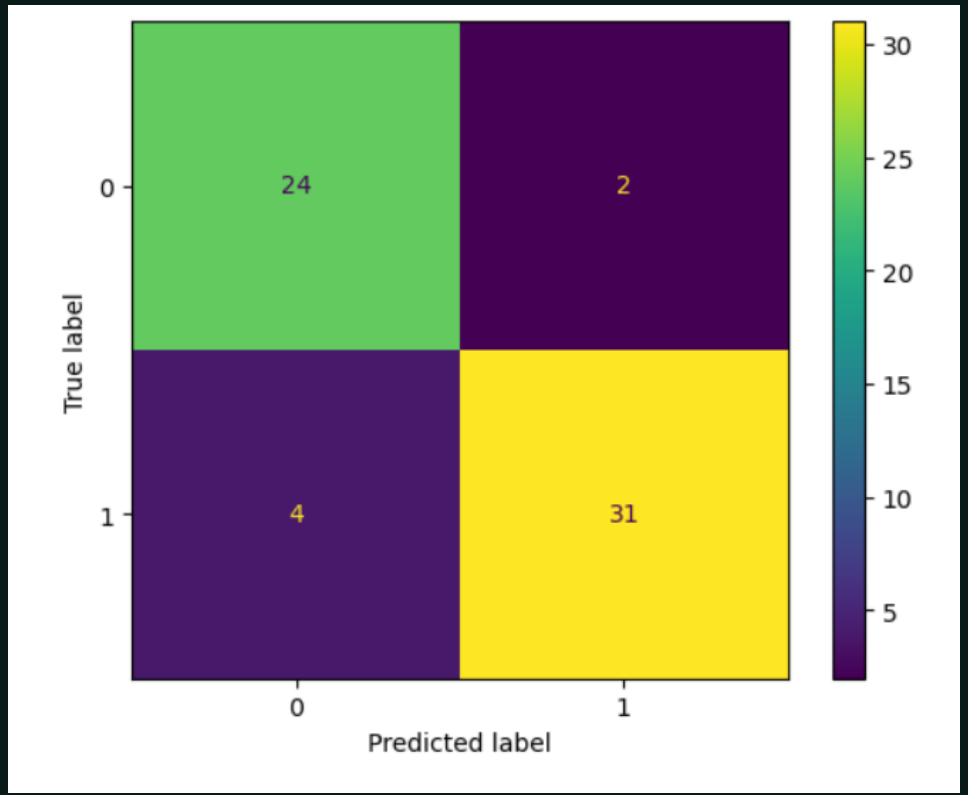


KNeighbors

- Tempo para treinar KNN classifier: 0.00207 segundos.
- Training Score: 0.9615384615384616
- Test Score: 0.9016393442622951
- Métricas de avaliação:

Accuracy Obtained: 0.90164

	precision	recall	f1-score	support
0	0.86	0.92	0.89	26
1	0.94	0.89	0.91	35
accuracy			0.90	61
macro avg	0.90	0.90	0.90	61
weighted avg	0.90	0.90	0.90	61

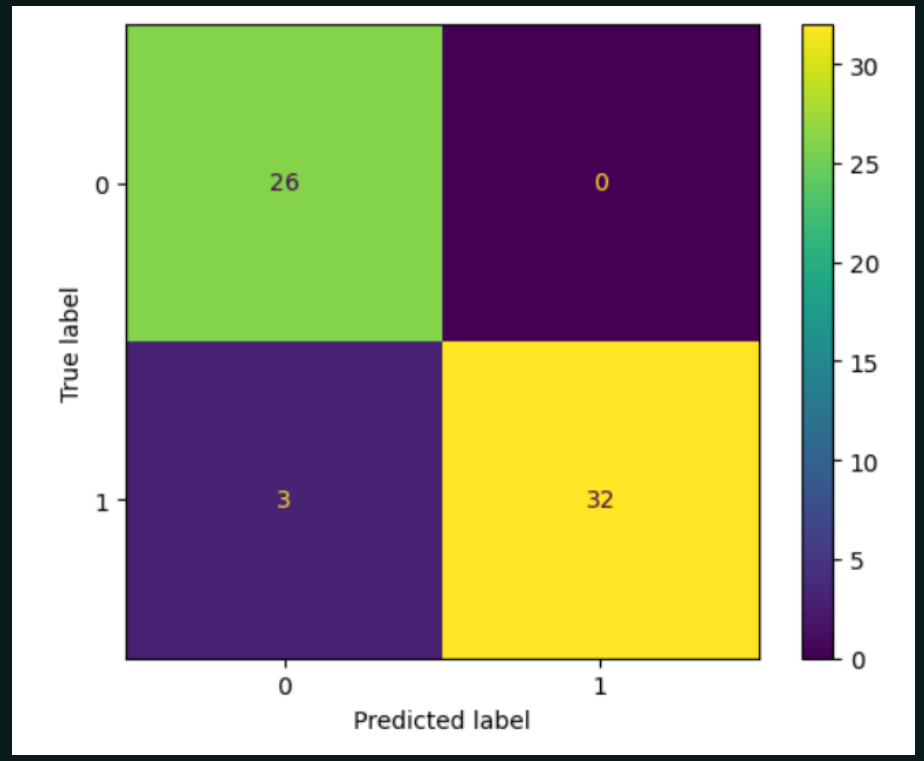


Random Forest

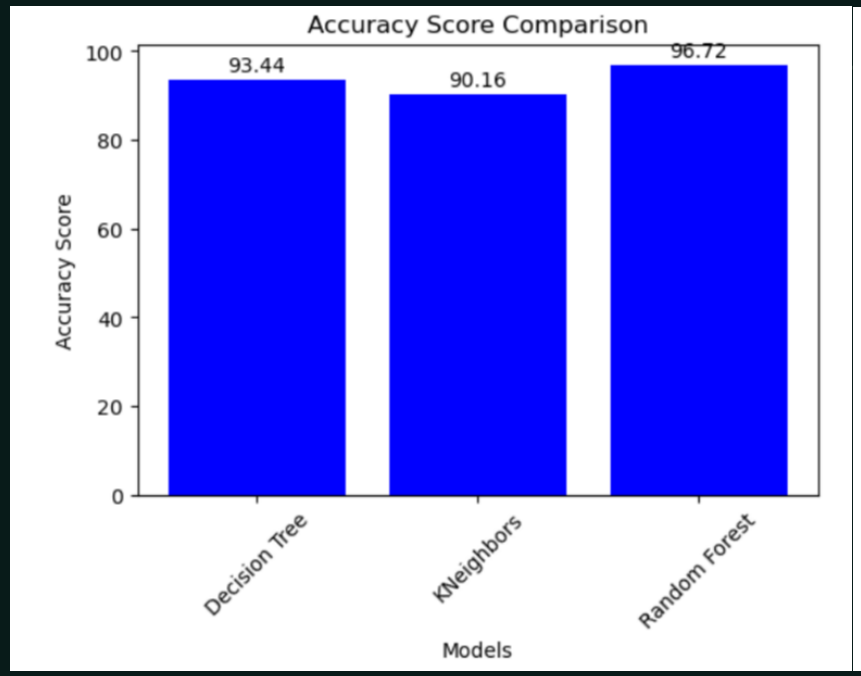
- Tempo para treinar Random Forest classifier: 0.10755 segundos.
- Training Score: 1.0
- Test Score: 0.9508196721311475
- Métricas de avaliação:

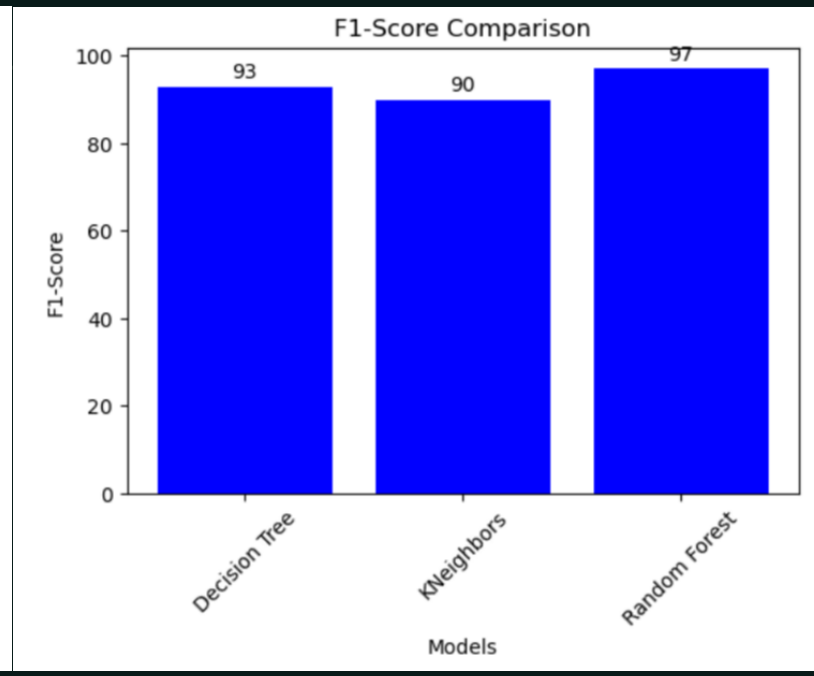
Accuracy Obtained: 0.95082

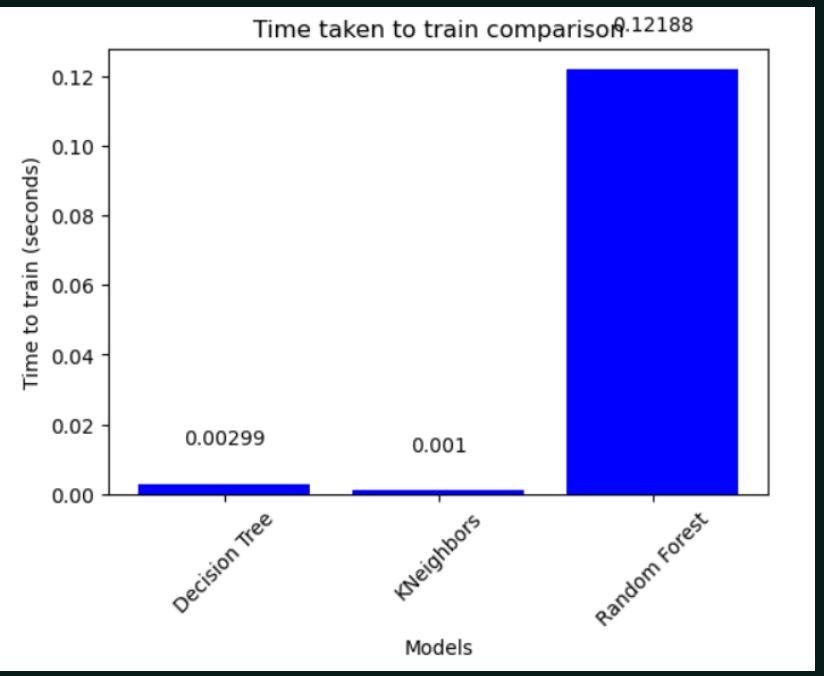
	precision	recall	f1-score	support
0	0.90	1.00	0.95	26
1	1.00	0.91	0.96	35
accuracy			0.95	61
macro avg	0.95	0.96	0.95	61
weighted avg	0.96	0.95	0.95	61



Avaliação e comparação







Através da análise concluímos que o algoritmo Random Forest possui melhor performance entre os 3, pois apesar de demorar o maior tempo a treinar, produz resultados com maior precisão e F1-score. Em segundo lugar vem o Decision Tree e em último o KNeighbors. Apesar disso, ambos continuam a possuir precisão, recall e F1-score elevados.