

Análise de Alergia Alimentar

David Miguel Sousa Marques
Maria da Conceição Pereira Tavares

Noções Fundamentais de Machine Learning ⁱ

ⁱ Escola Superior de Tecnologia e Gestão – Politécnico do Porto

Grupo 7: {8190565, 8190709}@estg.ipp.pt

January 1, 2023

Índice

1	Introdução	2
2	Metodologia	3
2.1	Compreensão do problema	3
2.1.1	Sobre o dataset	4
2.2	Preparação dos dados	6
2.2.1	Carregar o dataset	6
2.2.2	Tratamento de dados	7
2.2.3	Outliers	7
2.3	Escolha do modelo, treino, otimização e validação	8
2.4	Desenvolvimento do Frontend	9
3	Aplicação	9
4	Resultados	16
5	Conclusão	20

1 Introdução

Machine Learning é uma área em constante evolução na computação e tem aplicações em uma ampla gama de setores, incluindo saúde, finanças, marketing e muito mais.

Este relatório visa explorar as possibilidades existentes na área aplicado a um determinado conjunto de dados, mais especificamente **Análise de Alergia Alimentar**, com o objetivo de identificar padrões e tendências.

Ao longo deste relatório, o conjunto de dados será analisado ao detalhe, serão aplicadas técnicas de *Machine Learning* e será avaliada a eficácia dos modelos gerados. Discutiremos também as limitações e desafios enfrentados durante o processo de exploração do conjunto de dados.

Esperamos que este relatório forneça uma visão clara e detalhada do poder do *Machine Learning* na análise de dados e possa inspirar futuras pesquisas e aplicações nesta área em constante evolução.

2 Metodologia

O trabalho realizado foi desenvolvido com o objetivo de explorar as capacidades e limitações do uso de algoritmos de aprendizado de máquina para resolver um problema específico.

Para alcançar esse objetivo, passamos por seis grandes fases, nomeadamente:

- Compreensão do problema;
- Preparação dos dados;
- Escolha do modelo;
- Treino e otimização;
- Validação;
- Frontend.

2.1 Compreensão do problema

Sendo esta a primeira fase, foi necessário realizar uma análise aprofundada do problema e dos dados que pretendemos estudar de forma a compreender as suas nuances e definir as métricas de avaliação adequadas. Esta fase servirá de base para o desenvolvimento consequente.

Primeiramente e atendendo aos requisitos do enunciado, o conjunto de dados foi escolhido: ‘**Childhood Allergies: Prevalence, Diagnosis, and Treatment Outcomes**’.

2.1.1 Sobre o dataset

Este dataset contém o poder de nos ajudar a entender melhor a prevalência e os resultados do tratamento de alergias infantis durante um longo período de tempo. Não apenas divulga o número de indivíduos que sofrem atualmente de asma, dermatite atópica, rinite alérgica e alergias alimentares por meio de dados retrospectivos relatados por profissionais de saúde mas também apresenta um conjunto de colunas que nos permitem obter informações valiosas sobre como esses resultados diferem em diferentes dados demográficos, como gênero, raça e etnia.

Ao examinar mais a fundo esses dados, podemos começar a reconhecer padrões nas tendências entre os casos diagnosticados abrindo caminho para novos tratamentos e estratégias de prevenção que podem prevenir reações alérgicas graves em muitas crianças em todo o mundo.

O dataset original, é formado por cinquenta colunas com valores do tipo como podemos observar de seguida:

Coluna	Descrição	Tipo	Nulos
SUBJECT_ID	Identificador auto-incremental	Inteiro	0
BIRTH_YEAR	Ano de nascimento	String	0
GENDER_FACTOR	Género	String	0
RACE_FACTOR	Raça	String	0
ETHNICITY_FACTOR	Etnia	String	0
PAYER_FACTOR	Cobertura de seguro	String	0
ATOPIC_MARCH_COHORT	Progressão sequencial de condições alergénicas	String	0
AGE_START_YEARS	Idade inicial no estudo	String	0
AGE_END_YEARS	Idade final no estudo	String	0
SHELLFISH_ALG_START	Alergia a marisco no início	Float	327954

SHELLFISH_ALG_END	Alergia a marisco no final	Float	332149
FISH_ALG_START	Alergia a peixe no início	Float	331404
FISH_ALG_END	Alergia a peixe no final	Float	332673
MILK_ALG_START	Alergia a leite no início	String	325910
MILK_ALG_END	Alergia a leite no final	Float	328620
SOY_ALG_START	Alergia a soja no início	Float	330781
SOY_ALG_END	Alergia a soja no final	Float	331769
EGG_ALG_START	Alergia a ovo no início	Float	327135
EGG_ALG_END	Alergia a ovo no final	Float	329907
WHEAT_ALG_START	Alergia a trigo no início	Float	332054
WHEAT_ALG_END	Alergia a trigo no final	String	332511
PEANUT_ALG_START	Alergia a amendoim no início	Float	324547
PEANUT_ALG_END	Alergia a amendoim no final	String	331107
SESAME_ALG_START	Alergia a sésamo no início	Float	332434
SESAME_ALG_END	Alergia a sésamo no final	Float	333022
TREENUT_ALG_START	Alergia a nozes no início	Float	333199
TREENUT_ALG_END	Alergia a nozes no final	Float	333200
WALNUT_ALG_START	Alergia a nozes no início	Float	332496
WALNUT_ALG_END	Alergia a nozes no final	Float	333034
PECAN_ALG_START	Alergia a noz-pecã no início	Float	332915
PECAN_ALG_END	Alergia a noz-pecã no final	Float	333141
PISTACH_ALG_START	Alergia a pistáchio no início	Float	332831
PISTACH_ALG_END	Alergia a pistáchio no final	Float	333118
ALMOND_ALG_START	Alergia a amêndoas no início	Float	332814
ALMOND_ALG_END	Alergia a amêndoas no final	Float	333083
BRAZIL_ALG_START	Alergia a castanha brasileira no início	Float	333132
BRAZIL_ALG_END	Alergia a castanha brasileira no final	Float	333181
HAZELNUT_ALG_START	Alergia a avelã no início	Float	332947
HAZELNUT_ALG_END	Alergia a avelã no final	Float	333148
CASHEW_ALG_START	Alergia a caju no início	Float	332639
CASHEW_ALG_END	Alergia a caju no final	Float	333079

ATOPIC_DERM_START	Estado de dermatite atópica no início	Float	283685
ATOPIC_DERM.END	Estado de dermatite atópica no final	Float	291468
ALLERGIC_RHINITIS_START	Estado de rinite alérgica no início	Float	277633
ALLERGIC_RHINITIS.END	Estado de rinite alérgica no final	Float	307874
ASTHMA.START	Estado de asma no início	Float	269326
ASTHMA.END	Estado de asma no final	Float	307735
FIRST_ASTHMARX	Primeiro medicamento para asma prescrito	Float	215650
LAST_ASTHMARX	Último medicamento para asma prescrito	Float	215650
NUM_ASTHMARX	Número de medicamentos para asma prescritos	Float	21560

Tabela 1: Estrutura do dataset

2.2 Preparação dos dados

É necessário que os dados sejam limpos e preparados para serem utilizados na construção do modelo. Para isso, foram seguidos alguns passos nomeadamente:

2.2.1 Carregar o dataset

Nesta fase, o objetivo principal passa por carregar o dataset para memória para poder ser manipulado e utilizado nos passos seguintes.

Para isso, foi utilizado o pacote **Pandas**, que é capaz de ler um ficheiro csv e criar um *dataframe* com a informação existente no ficheiro

indicado.

Ao efetuar a leitura do ficheiro, é possível alterar os valores de forma a que quando o *dataframe* for criado, estes dados sejam armazenados com novos valores no seu lugar.

2.2.2 Tratamento de dados

Esta fase consiste em trabalhar os dados obtidos da fonte de forma a transforma-los em dados utilizaveis pelos modelos que serão treinados futuramente. A transformação de variáveis categóricas em variáveis numéricas, é um dos exemplos de casos de uso.

Além disso, o *dataset* é analisado linha a linha de forma a encontrar descrepâncias nos dados, bem como algumas correções nos valores, como por exemplo a alteração do tipo de dados ou o tratamento de valores em falta.

Por fim será analisada a correlação entre os dados recorrendo a multicolinearidade que é uma condição em que duas ou mais variáveis independentes são altamente correlacionadas entre si em uma análise estatística. Isso pode levar a problemas de interpretação dos coeficientes de regressão, uma vez que as mudanças nas variáveis altamente correlacionadas podem afetar mutuamente o resultado final.

2.2.3 Outliers

Ao explorar dados, os outliers são os valores extremos dentro de um conjunto de dados. Isso significa que os pontos de dados discrepantes

variam muito em relação aos valores esperados, sendo estes maiores ou significativamente menores.

Os outliers podem ser o resultado de problemas envolvendo o erro humano, equipamentos defeituosos ou uma amostra inadequada. Independentemente de como entram nos dados, podem ter um grande impacto na análise estatística e na aprendizagem da máquina porque irão afetar cálculos como a média e desvio padrão dos dados.

A visualização de dados como um gráfico de caixa ‘boxplot’ facilita a identificação de valores discrepantes. Esse gráfico mostrará a ‘caixa’ que identifica o intervalo interquartil cujo meio representa a média dos dados. Quaisquer outliers são mostrados fora dos valores mínimos e máximos do conjunto de dados respetivamente.

2.3 Escolha do modelo, treino, otimização e validação

Serão avaliados vários modelos de *machine learning* e escolhido o mais adequado para o problema em questão, baseado na sua performance em dados de treino e validação.

O modelo escolhido será então treinado com os dados de treino e otimizado, ajustando seus hiperparâmetros para melhorar sua performance.

Por fim, o modelo será avaliado com os dados de validação e posteriormente com as métricas de avaliação visando para obter a melhor eficiência e precisão possível.

2.4 Desenvolvimento do Frontend

Será então desenvolvido uma aplicação web, capaz de consumir o modelo obtido de forma a que previsões futuras possam ser feitas com novos valores mais facilmente e de forma mais iterativa possível sendo esta desenvolvida com recurso a *framework Flask*.

3 Aplicação

Nesta secção será abordada a aplicação e as escolhas das técnicas de acordo a metodologia abordada na secção anterior.

Tratamento de dados e outliers

Para o tratamento de dados, começamos por corrigir alguns erros presentes no *dataset*, utilizando o método 'replace' presente no objeto do tipo 'dataset' que foi criado ao fazer o *import* do documento.

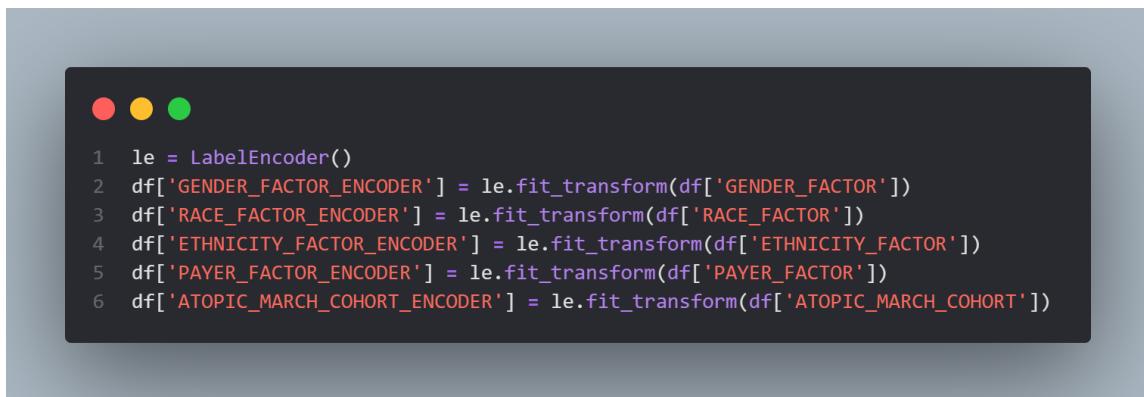
```
● ● ●  
1 df = pd.read_csv('../food-allergy-analysis-Zenodo.csv', low_memory=False).replace('FALSE', 'False').replace(np.NaN, 0) \  
2 .replace("NA", 0).replace('na', 0).replace('\\', '')  
3 df = df.drop(df.columns[0], axis=1).drop_duplicates(ignore_index=True)
```

Figura 1: Excerto de código responsável pelo import dos dados, correção de valores e remoção de linhas duplicadas

Inicialmente existiam 333200 linhas para 50 colunas, sendo que depois do processo terminar sobram apenas 328494 linhas para 49 colunas.

De seguida, passou-se para o tratamento das variáveis categóricas. Recorrendo a técnica '*LabelEncoder*', criaram-se novas colunas sem apagar as originais da fonte, em que:

- GENDER_FACTOR- GENDER_FACTOR_ENCODER
0 = **Male**, 1 = **Female**.
- RACE_FACTOR- RACE_FACTOR_ENCODER
0 = **White**, 1 = **Black**, 2 = **Asian or Pacific Islander**, 3 = **Other**, 4 = **Unknown**.
- ETHNICITY_FACTOR- ETHNICITY_FACTOR_ENCODER
0 = **Non-Hispanic**, 1 = **Hispanic**.
- PAYER_FACTOR- PAYER_FACTOR_ENCODER
0 = **Non-Medicaid**, 1 = **Medicaid**.
- ATOPIC_MARCH_COHORT- ATOPIC_MARCH_COHORT_ENCODER
0 = **False**, 1 = **True**.



The screenshot shows a Jupyter Notebook cell with a dark theme. At the top left are three colored dots (red, yellow, green). The cell contains the following Python code:

```
1 le = LabelEncoder()
2 df['GENDER_FACTOR_ENCODER'] = le.fit_transform(df['GENDER_FACTOR'])
3 df['RACE_FACTOR_ENCODER'] = le.fit_transform(df['RACE_FACTOR'])
4 df['ETHNICITY_FACTOR_ENCODER'] = le.fit_transform(df['ETHNICITY_FACTOR'])
5 df['PAYER_FACTOR_ENCODER'] = le.fit_transform(df['PAYER_FACTOR'])
6 df['ATOPIC_MARCH_COHORT_ENCODER'] = le.fit_transform(df['ATOPIC_MARCH_COHORT'])
```

Figura 2: Excerto de código responsável pelo LabelEncoding

Consequentemente foi possível passar para a análise linha a linha, substituindo valores resultantes de algumas variáveis do *LabelEncoding*, pelos valores por defeito.

Além disso, faz um cast das variáveis do tipo categórico para numérico, recorrendo ao **Pandas** e por fim, dependendo dos valores respetivos aos índices de alergénicos, cria uma nova coluna '**HAS_ALLERGIES**', que será a nossa variável dependente.

```

1  for line in range(0, len(df)):
2      if df['ATOPIC_MARCH_COHORT_ENCODER'][line] == 2:
3          df.loc[line, 'ATOPIC_MARCH_COHORT_ENCODER'] = 0
4      if df['RACE_FACTOR_ENCODER'][line] == 4:
5          df.loc[line, 'RACE_FACTOR_ENCODER'] = 3
6      if type(df['BIRTH_YEAR'][line]) == str:
7          df.loc[line, 'BIRTH_YEAR'] = pd.to_numeric(df['BIRTH_YEAR'][line].replace('\'', ''), downcast="integer")
8      if type(df['AGE_START_YEARS'][line]) == str:
9          df.loc[line, 'AGE_START_YEARS'] = pd.to_numeric(df['AGE_START_YEARS'][line].replace('\'', '')) 
10     if type(df['MILK_ALG_START'][line]) == str:
11         df.loc[line, 'MILK_ALG_START'] = pd.to_numeric(df['MILK_ALG_START'][line].replace('\'', ''), downcast="float")
12     if type(df['WHEAT_ALG_END'][line]) == str:
13         df.loc[line, 'WHEAT_ALG_END'] = pd.to_numeric(df['WHEAT_ALG_END'][line].replace('\'', ''), downcast="float")
14     if type(df['PEANUT_ALG_END'][line]) == str:
15         df.loc[line, 'PEANUT_ALG_END'] = pd.to_numeric(df['PEANUT_ALG_END'][line].replace('\'', ''), downcast="float")
16     if df['SHELLFISH_ALG_END'][line] > 0 or df['FISH_ALG_END'][line] > 0 or df['MILK_ALG_END'][line] > 0 or \
17         df['SOY_ALG_END'][line] > 0 or df['EGG_ALG_END'][line] > 0 or df['WHEAT_ALG_END'][line] > 0 or \
18         df['PEANUT_ALG_END'][line] > 0 or df['SESAME_ALG_END'][line] > 0 or df['TREENUT_ALG_END'][line] > 0 or \
19         df['WALNUT_ALG_END'][line] > 0 or df['PECAN_ALG_END'][line] > 0 or df['PISTACH_ALG_END'][line] > 0 or \
20         df['ALMOND_ALG_END'][line] > 0 or df['BRAZIL_ALG_END'][line] > 0 or df['HAZELNUT_ALG_END'][line] > 0 or \
21         df['CASHEW_ALG_END'][line] > 0 or df['ATOPIC_DERMAP_END'][line] > 0 or df['ALLERGIC_RHINITIS_END'][line] > 0 \
22         or df['ASTHMA_END'][line] > 0:
23         df.loc[line, 'HAS_ALLERGIES'] = 1
24     else:
25         df.loc[line, 'HAS_ALLERGIES'] = 0

```

Figura 3: Excerto de código responsável pelo LabelEncoding

Para resolver os outliers como se tratassem de valores ausentes, isto é, em todos os valores ausentes e valores discrepantes preencheremos o seu valor usando a média.

Identificamos três possíveis colunas que poderiam conter valores deste tipo, sendo estas:

- AGE-END-YEARS
- AGE-START-YEARS
- BIRTH-YEAR

```

● ● ●

1 df_original = df[ 'AGE_END_YEARS' ]
2 df[ 'AGE_END_YEARS' ] = resolve_outliers(df[ 'AGE_END_YEARS' ])
3 save_boxplot(df_original, df[ 'AGE_END_YEARS' ], 'AGE_END')
4
5 df_original = df[ 'AGE_START_YEARS' ]
6 df[ 'AGE_START_YEARS' ] = resolve_outliers(df[ 'AGE_START_YEARS' ])
7 save_boxplot(df_original, df[ 'AGE_START_YEARS' ], 'AGE_START')
8
9 df_original = df[ 'BIRTH_YEAR' ]
10 df[ 'BIRTH_YEAR' ] = resolve_outliers(df[ 'BIRTH_YEAR' ])
11 save_boxplot(df_original, df[ 'BIRTH_YEAR' ], 'BIRTH_YEAR')

```

Figura 4: Excerto de código responsável pela resolução dos outliers

Para calcular a multicolinearidade entre os dados, recorreu-se ao método ‘variance_inflation_factor’, proveniente do package ‘sklearn’.

O VIF é calculado para cada variável independente no modelo e mede o grau de correlação entre essa variável e as outras variáveis independentes no modelo. O VIF é definido como: **VIF = 1 / (1 - R²)**, onde R² é o coeficiente de determinação obtido a partir de uma regressão linear da variável independente em questão nas outras variáveis independentes do modelo.

```

1 threshold = 10
2 independent_variables = ['BIRTH_YEAR', 'GENDER_FACTOR_ENCODER', 'RACE_FACTOR_ENCODER', 'ETHNICITY_FACTOR_ENCODER',
3                           'PAYER_FACTOR_ENCODER', 'ATOPIC_MARCH_COHORT_ENCODER', 'AGE_START_YEARS', 'AGE_END_YEARS',
4                           'SHELLFISH_ALG_START', 'SHELLFISH_ALG_END', 'FISH_ALG_START', 'FISH_ALG_END', 'MILK_ALG_START',
5                           'MLK_ALG_END', 'SOY_ALG_START', 'SOY_ALG_END', 'EGG_ALG_START', 'EGG_ALG_END',
6                           'WHEAT_ALG_START', 'WHEAT_ALG_END', 'PEANUT_ALG_START', 'PEANUT_ALG_END', 'SESAME_ALG_START',
7                           'SESAME_ALG_END', 'WALNUT_ALG_START', 'WALNUT_ALG_END', 'ALMOND_ALG_START',
8                           'ALMOND_ALG_END', 'CASHEW_ALG_START', 'CASHEW_ALG_END', 'ATOPIC_DERM_START',
9                           'ATOPIC_DERM_END', 'ALLERGIC_RHINITIS_START', 'ALLERGIC_RHINITIS_END', 'ASTHMA_START',
10                          'ASTHMA_END']
11
12 x = df[independent_variables].astype('float')
13 for i in np.arange(0, len(independent_variables)):
14     vif = [variance_inflation_factor(x[independent_variables], ix) for ix in range(x[independent_variables].shape[1])]
15     maxloc = vif.index(max(vif))
16     if max(vif) > threshold:
17         print('dropping ', x[independent_variables].columns[maxloc], ' at index ', maxloc)
18         del independent_variables[maxloc]
19     else:
20         break

```

Figura 5: Excerto de código responsável pela verificação da correlação entre dados

Escolha dos modelos

O primeiro modelo escolhido foi o PCA, isto devido a quantidade de variáveis dependentes existentes no *dataset*. PCA é uma técnica de redução de dimensionalidade que permite representar um conjunto de dados multivariados em um conjunto de variáveis linearmente combinadas com menos dimensões, mantendo a maior parte da informação. É amplamente utilizado para análise exploratória de dados, visualização de dados e redução de dimensionalidade.



```
1  scaler = StandardScaler()
2  scaler.fit(x)
3  xpca = scaler.transform(x)
4
5  pca = PCA(.99)
6  pca.fit(xpca)
7
8  x = pca.transform(xpca)
9  y = df['HAS_ALLERGIES']
```

Figura 6: Excerto de código responsável pelo treino do modelo PCA

Após o treino do primeiro modelo, aplicou-se o modelo classificador: ‘**Random Forest**’. Este modelo é um conjunto de árvores de decisão que são construídas com amostras aleatórias dos dados de treinamento, utilizando uma técnica chamada *bagging*, que consiste em combinar várias árvores independentes para reduzir a variância e melhorar a generalização do modelo em que Cada árvore é construída a partir de uma amostra aleatória do conjunto de dados de treinamento e usa um subconjunto aleatório de recursos para a divisão de nós. Isso reduz a correlação entre as árvores e aumenta a diversidade do modelo.

Random Forest é conhecido por sua boa precisão e robustez em relação a dados ruidosos ou desbalanceados, e é amplamente utilizado em aplicações práticas devido à sua simplicidade e eficiência em grandes conjuntos de dados.

```
● ● ●  
1 x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.6)  
2  
3 model = RandomForestClassifier(n_estimators=100, max_depth=100)  
4 model.fit(x_train, y_train)  
5  
6 predictions = model.predict(x_test)  
7 accuracy = model.score(x_test, y_test)
```

Figura 7: Excerto de código responsável pelo treino do modelo classificatório: Random Forest

Para finalizar, este modelo treinado foi exportado para um ficheiro do tipo ‘.sav’, para ser consumido pelo frontend desenvolvido.

```
● ● ●  
1 joblib.dump(model, '../Frontend/model/random_forest_model.sav')
```

Figura 8: Excerto de código responsável pela exportação do modelo

4 Resultados

Nesta secção serão apresentados os resultados obtidos após a aplicação das técnicas abordadas no capítulo anterior. Primeiramente, após o tratamento de dados, foi possível resolver os outliers obtendo os seguintes resultados:

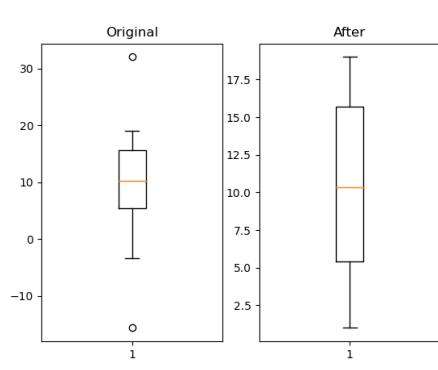


Figura 9: Outlier
'AGE-END_YEARS'

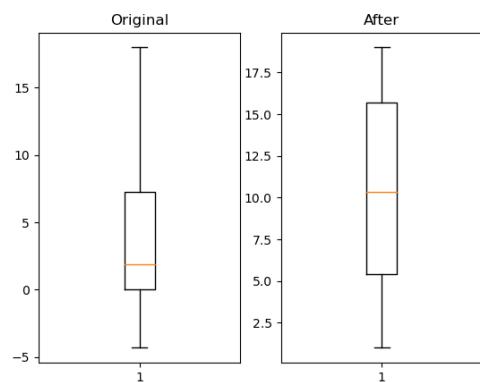


Figura 10: Outlier
'AGE-START_YEARS'

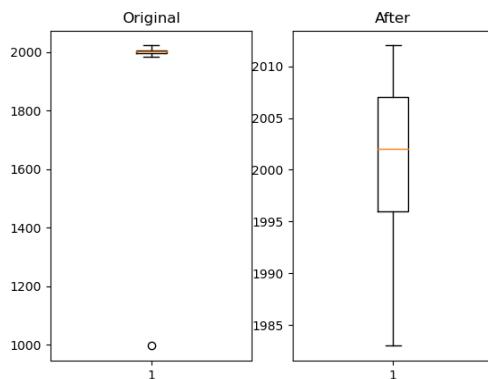


Figura 11: Outlier 'BIRTH_YEAR'

Ao treinar o modelo obtemos o seguinte resultado:

	precision	recall	f1-score	support	
1					
2					
3	0.0	0.98	0.99	0.99	98343
4	1.0	0.97	0.95	0.96	33037
5					
6	accuracy			0.98	131380
7	macro avg	0.97	0.97	0.97	131380
8	weighted avg	0.98	0.98	0.98	131380

Figura 12: Resumo de classificatório

Como podemos observar, o modelo treinado tem uma acurácia de 98% para acertar previsões. Em 33037 crianças alérgicas, o modelo consegue prever 95% de verdadeiros positivos, e em 98343 crianças não alérgicas, consegue prever 99% de verdadeiros negativos, concluindo assim que o modelo treinado tem uma acurácia de 98%.

Complementando os resultados obtidos ao treinar o modelo, calculou-se o mean absolute error de 2% e um mean square logarithmic error de 15%.

Tendo em conta uma amostra bastante reduzida do dataset real, foi gerado o gráfico de linhas capaz de comparar os valores reais com os valores previstos pelo modelo conforme podemos observar na figura abaixo.

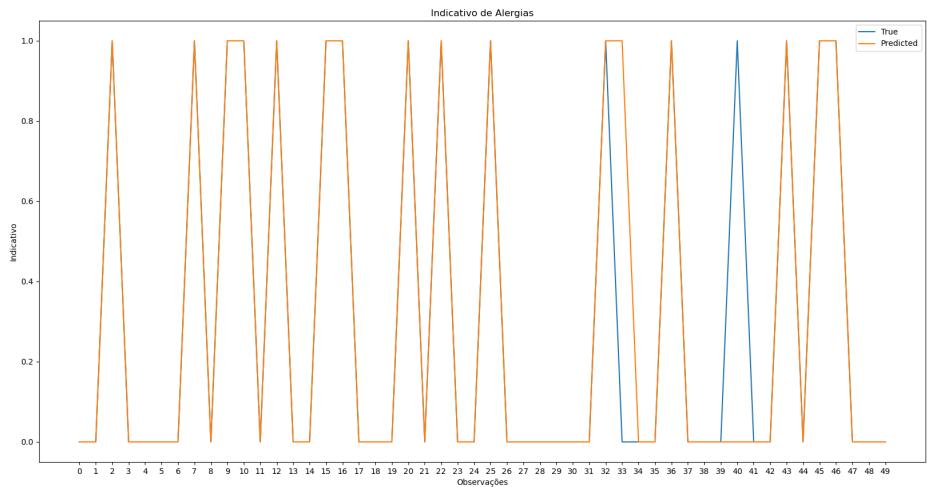


Figura 13: Gráfico de previsões VS valores reais

Por fim, utilizando o frontend desenvolvido é agora possível interagir com o modelo treinado de forma a fazer previsões para tentar perceber se um indivíduo é alérgico ou não.

Após preencher o formulário, a informação é enviada para o backend em que é feita a previsão utilizando o modelo previamente treinado obtendo uma resposta positiva caso a previsão seja igual a 1 ou negativa caso contrário.

Alergias na infância: prevalência, demografia

Gênero:
Masculino

Raça:
Outra

Etnia:
Não Hispânica

Histórico alérgico:
Não

Reações:

Marisco: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim	Sésamo: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim
Peixe: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim	Nozes: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim
Leite: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim	Amêndoas: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim
Soja: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim	Cajú: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim
Ovos: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim	Dermatite atópica: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim
Trigo: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim	Rinite: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim
Amendoim: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim	Asma: <input type="radio"/> Não <input type="radio"/> Talvez <input type="radio"/> Sim

Prever

Figura 14: Visualização do formulário presente na web app desenvolvida



Figura 15: Resposta positiva



Figura 16: Resposta negativa

5 Conclusão

Com base na análise do dataset escolhido e com recurso às técnicas de machine learning aprendidas durante o semestre, podemos concluir que a utilização de modelos preditivos pode ser uma ferramenta valiosa sendo que o trabalho desenvolvido visa a identificação e previsão de alergias em crianças.

Os resultados obtidos fornecem informações úteis para profissionais de saúde, e permitem uma abordagem mais personalizada e eficaz no tratamento e prevenção de alergias em crianças. No entanto, é importante lembrar que a utilização de modelos de machine learning não é infalível, e deve ser combinada com a avaliação clínica de um profissional de saúde qualificado.

Em resumo, este trabalho apresentou uma análise abrangente as técnicas de machine learning. Foi possível verificar que o grupo atingiu os objetivos propostos, com resultados significativos. As implicações práticas destes resultados são amplas, sugerindo novas possibilidades de aplicação e investigação. No entanto, é importante destacar que o estudo tem limitações, especialmente no que se refere ao tamanho da amostra. Recomenda-se, portanto, que pesquisas futuras ampliem a amostra e aprofundem a análise de algumas variáveis.