Universidade do Minho
Escola de Engenharia

Conceção de modelos de aprendizagem e decisão

Aprendizagem e Decisão Inteligentes

A96890, André Pimentel Filipe

A81971, Marcelo Araújo de Sousa

A100827, Tiago Granja Rodrigues

A100546, Tomás Monteiro Sousa

2023/24

1 Índice

2 Introdução

Este relatório surge no âmbito da Unidade Curricular de Aprendizagem e Decisão Inteligentes, em que foi proposto a conceção de modelos de aprendizagem.

Neste trabalho nos foram propostas duas tarefas:

- **Tarefa Dataset Atribuído (Grupo Ímpar):** em que foi fornecido pela equipa docente, um dataset com informações sobre pacientes, como análises médias, em que se pretendia prever a Categoria de cada paciente. Este problema foi caracterizado como sendo um problema de classificação.
- **Tarefa Dataset Grupo:** Nesta tarefa foi solicitada a escolha de um dataset, para um problema de regressão. O dataset que o grupo escolheu, contém informações sobre vendas de carros nos EUA, sendo o objetivo principal prever o preço de cada veículo.

Para cada tarefa iremos estudar o modelo de negócio do problema, analisar, explorar e preparar os dados, com o intuito de criar modelos de *machine learning* para prever o resultado de cada problema.

3 Metodologia

Para exploração dos dados de forma metódica, optou-se por utilizar a metodologia CRISP-DM. Esta metodologia oferece uma estrutura sólida e bem definida para conduzir projetos de análise de dados, dividindo o processo em seis fases distintas:

- **Estudo do negócio:** Nesta fase inicial, analisamos detalhadamente o contexto do negócio para identificar os seus objetivos e requisitos.
- **Exploração dos dados:** Aqui iremos analisar e explorar os dados fornecidos pelo dataset e perceber de que forma influenciam o atributo alvo, que é o que pretendemos prever.
- **Preparação dos dados:** Nesta fase, realizamos tarefas como limpeza, transformação, seleção e integração dos dados. Esta fase é fundamental para garantir a qualidade e consistência dos dados utilizados nos modelos subsequentes.
- **Modelação:** Nesta fase, iremos utilizar diversos algoritmos e técnicas de modelação, de forma a encontrar o modelo mais adequado para prever o atributo alvo.
- **Avaliação:** Embora não seja abordada neste projeto específico, a fase de implementação envolve a integração dos resultados da análise de dados no ambiente de negócios.
- **Desenvolvimento:** Embora não seja abordada neste projeto, esta fase envolve aspetos como o planeamento da implementação, produção do relatório final e revisão do projeto.

Esta metodologia será aplicada a ambos os datasets, garantindo uma abordagem consistente e abrangente para a análise e modelação dos dados.

4 Dataset HCV

4.1 Estudo do Negócio

O objetivo deste projeto é desenvolver modelos para prever a categoria médica de pacientes com base em análises médicas e dados dos mesmos. Para isso, dispomos de um dataset que possui informações de diversos pacientes. Para alcançar este objetivo, estabeleceu-se uma série de metas que iremos seguir ao longo deste projeto:

- Analisar e explorar todos os atributos do dataset, utilizando gráficos, de modo a compreender melhor as características e possíveis padrões presentes nos dados.
- Preparar os dados para que possam ser utilizados pelos modelos de previsão. Isto inclui tarefas como limpeza, tratamento de *missing values* e transformação de variáveis, garantindo a qualidade e consistência dos dados.
- Desenvolver diferentes modelos de previsão para determinar a categoria médica dos pacientes. No final, será feita uma análise do desempenho destes modelos para identificar o mais adequado às nossas necessidades.

Este problema é considerado um problema de classificação, uma vez que o atributo que queremos prever (***Category***) é um valor discreto.

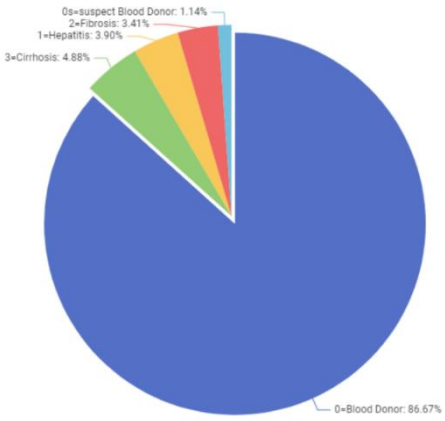
4.2 Exploração dos dados

Este dataset é composto por 18 colunas (atributos) e 615 linhas. Os atributos são os seguintes:

- **ID:** Identificador único do paciente.
- **Age:** Idade do paciente.
- **year_of_birth:** Ano de nascimento do paciente.
- **month_of_birth:** Mês de nascimento do paciente.
- **day_of_birth:** Dia de nascimento do paciente.
- **Sex:** Sexo do paciente.
- **birth_location:** Local de nascimento do paciente.
- **ALB:** Albumina no sangue.
- **ALP:** Fosfatase alcalina no sangue.
- **ALT:** Alanina aminotransferase no sangue.
- **AST:** Aspartato aminotransferase no sangue.
- **BIL:** Bilirrubina no sangue.
- **CHE:** Enzima colinesterase no sangue.
- **CHOL:** Colesterol no sangue.
- **CREA:** Creatina no sangue.
- **GGT:** Gama-glutamiltransferase no sangue.
- **PROT:** Proteína no sangue.
- **CATEGORY:** Categoria do paciente. Este atributo classifica os pacientes em diferentes grupos com base na sua condição médica e é o atributo que queremos prever.

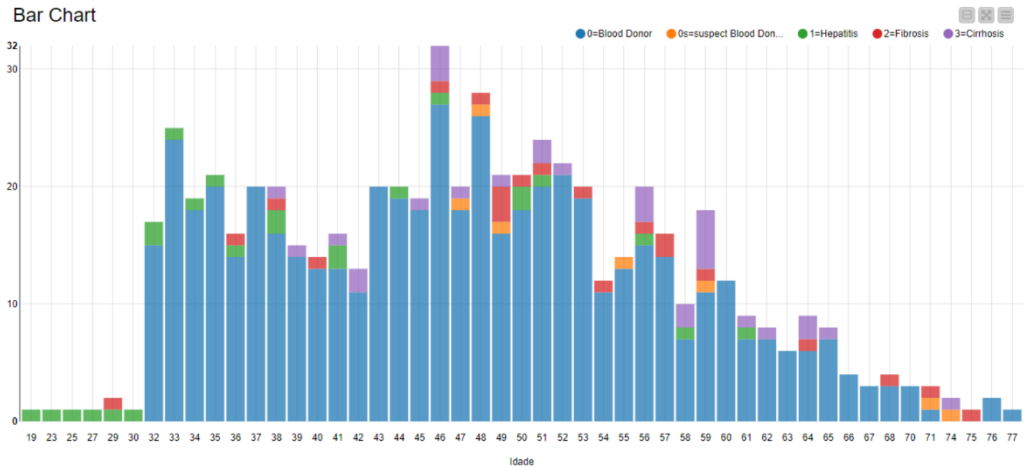
Em seguida, realizou-se uma exploração mais detalhada dos atributos para compreender melhor e identificar quais dos atributos influencia o atributo alvo.

4.2.1 Categoria (Category)



Ao analisar o atributo *Category*, verificou-se que possui os seguintes valores: **0=Blood Donor**, **0s=Suspect Blood Donor**, **1=Hepatitis**, **2=Fibrosis**, **3=Cirrosis**. Observou-se ainda, que a maioria dos pacientes são classificados como "Blood Donor" (86,67%), enquanto as outas categorias têm distribuições semelhantes, sendo o "Suspect Blood Donor" o menos comum (1,14%).

4.2.2 Idade (Age)



Ao analisar este atributo, verificou-se que as idades dos pacientes variam entre 19 e 77 anos, com a maioria dos pacientes concentrados na faixa etária entre 32 e 59 anos. Verificou-se que todos os pacientes com idade inferior a 32 anos foram diagnosticados com Hepatite, com a exceção de um único caso, que foi diagnosticado com Fibrose. Além disso, observou-se que a presença de Cirrose é identificada apenas em pacientes com idade superior a 38 anos, enquanto os casos de suspeitos de doadores de sangue surgem a partir dos 47 anos.

4.2.3 Ano, Mês e Dia de nascimento (Year, Month, Day_of_Birth)

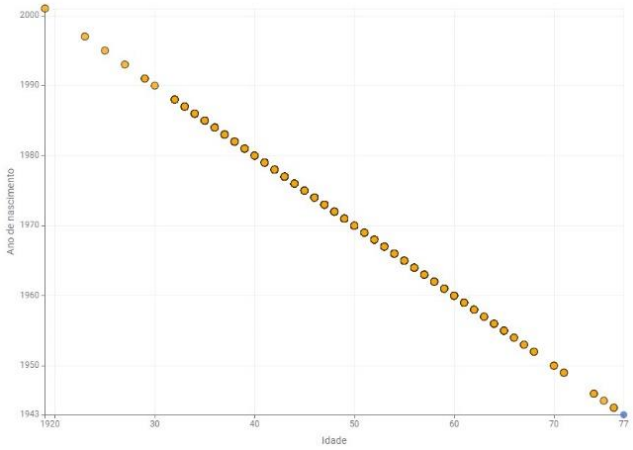


Figura 1 Relação entre idade e ano de nascimento

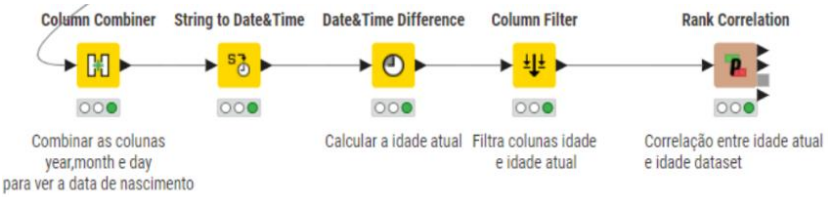


Figura 3 Calcular a idade atual e comparar com a idade do dataset

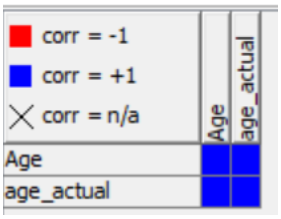


Figura 2 Correlação entre idade atual e idade do dataset

A idade e o ano de nascimento apresentam uma relação inversamente proporcional, tornando o atributo do ano de nascimento redundante neste contexto.

Além disso, o dia e o mês de nascimento não são relevantes para a resolução deste problema. Com o objetivo de validar a precisão das idades presentes no dataset, procedeu-se ao cálculo das idades atuais dos pacientes com base nas suas datas de nascimento.

Conforme esperado, detetaram-se discrepâncias entre as idades registadas no dataset e as idades calculadas atualmente, o que é explicável pelo tratamento dos dados ter ocorrido em um momento anterior. Apesar de se observar uma correlação perfeita (correlação de 1) entre a idade apresentada no dataset e a idade atual calculada, é de salientar que esta correlação não assegura a exatidão absoluta das idades fornecidas.

4.2.4 Sexo (Sex)

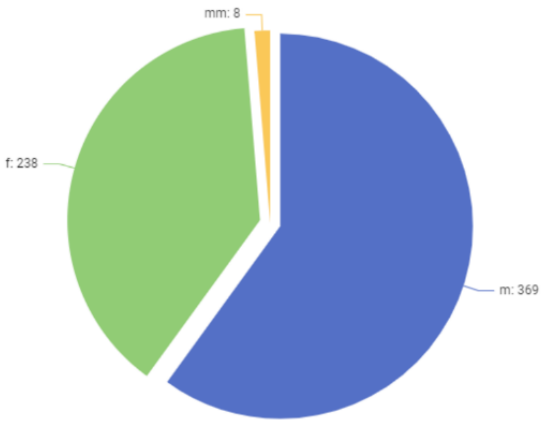


Figura 4 Distribuição dos sexos

Analisando a relação entre o sexo do paciente e a categoria, observou-se que existe uma correlação de 0,0828, o que indica que o sexo não tem uma influência significativa sobre a categoria.

Dos pacientes presentes no dataset, 238 são do sexo feminino ('f') e 369 são do sexo masculino ('m'). Além disso, foi identificado 8 registos com a designação de sexo 'mm', o que provavelmente foi um erro na inserção dos dados.

4.2.5 Local de nascimento (Birth_Location)

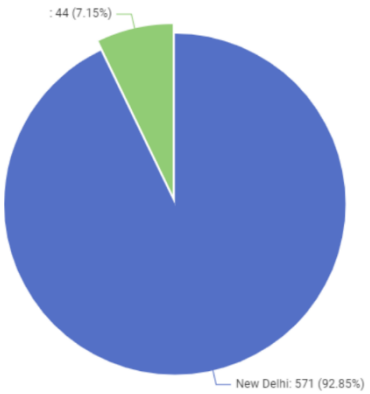


Figura 5 Distribuição do local de nascimento

Ao analisar o local de nascimento, é possível verificar que 571 dos pacientes nasceram em New Delhi, representando 92,85% do total de pacientes. Também se verifica 44 *missing values*, que correspondem a 7,15% dos registos.

Como a maior parte dos pacientes nasceu em New Delhi, não é possível tirar conclusões de como o local de nascimento influencia na categoria do paciente.

4.2.6 ALB

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
ALB	14,9	41,6202	?	82,2	5,7806	-0,1768	5,9833	1	0	0

Figura 6 Estatísticas ALB

Os níveis de albumina no sangue variam entre 14,9 e 82,2, com uma média de 41,602 e possui 1 valor em falta.

4.2.7 ALP

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
ALP	11,3	68,2839	?	416,6	26,0283	4,6549	54,9729	18	0	0

Figura 7 Estatísticas ALP

Os níveis de fosfatase alcalina no sangue variam entre 11,3 e 416,6, com uma média de 68,2839.

4.2.8 ALT

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
ALT	0,9	28,4508	?	325,3	25,4697	5,5061	47,1293	1	0	0

Figura 8 Estatísticas ALT

Os níveis de alanina aminotransferase variam entre 0,9 e 325,3, com uma média de 28,4508. Este atributo possui apenas 1 missing value.

4.2.9 AST

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
AST	10,6	34,7863	?	324	33,0907	4,9403	30,8366	0	0	0

Figura 9 Estatísticas AST

Os níveis de Aspartato Aminotransferaseno variam entre 10,6 e 324, com uma média de 34,7863.

Figura 10 Scatter Plot dos níveis de AST em relação à Categoria

Ao analisar o nodo *Rank Correlation* observa-se uma correlação de 0.4484 entre os valores de AST e a Categoria do paciente. Isto sugere uma influência moderada de AST nos valores da Categoria. Decidiu-se explorar mais esta relação e, utilizando o nodo *Scatter Plot*, pode-se observar que, de forma geral, os doadores de sangue tendem a ter os valores mais baixos de AST, enquanto os pacientes com Cirrose possuem os valores mais altos. As demais categorias apresentam níveis de AST mais equilibrados.

4.2.10 BIL

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
BIL	0,8	11,3967	?	254	19,6731	8,3854	83,1867	0	0	0

Figura 11 Estatísticas BIL

Os níveis de Bilirrubina variam entre 0,8 e 254, com uma média de 11,3967.

4.2.11 CHE

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
CHE	1,42	8,1966	?	16,41	2,2057	-0,1102	1,3147	0	0	0

Figura 12 Estatísticas CHE

Os níveis de Colinesterase variam entre 1,42 e 16,41, com uma média de 8,1966.

4.2.12 CHOL

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
CHOL	1,43	5,3681	?	9,67	1,1327	0,3758	0,694	10	0	0

Figura 13 Estatísticas CHOL

Os níveis de Colesterol variam entre 1,43 e 9,67, com uma média de 5,3681.

4.2.13 CREA

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
CREA	8	81,2878	?	1 079,1	49,7562	15,1693	280,1002	0	0	0

Figura 14 Estatísticas CREA

Os níveis de Creatina variam entre 8 e 1079,1, com uma média de 81,2878.

4.2.14 GGT

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
GGT	4,5	39,5332	?	650,9	54,6611	5,6327	43,7126	0	0	0

Figura 15 Estatísticas GGT

Os níveis de Gama-glutamiltransferase variam entre 4,5 e 650,9, com uma média de 39,5332.

4.2.15 PROT

Column	Min	Mean	Median	Max	Std. Dev.	Skewness	Kurtosis	No. Missing	No. +∞	No. -∞
PROT	44,8	72,0441	?	90	5,4026	-0,9637	3,5445	1	0	0

Figura 16 Estatísticas PROT

Os níveis de Proteína variam entre 44,8 e 90, com uma média de 72,0441. Este atributo possui apenas 1 missing value.

4.3 Pré Processamento

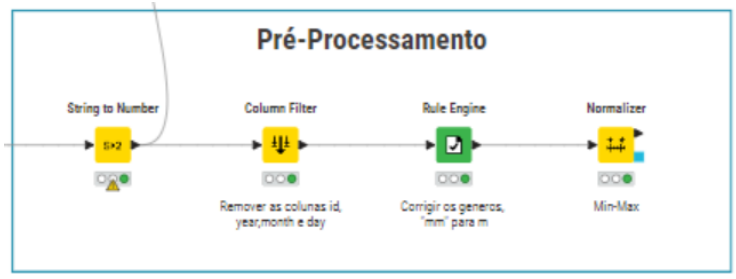


Figura 17 Pré-processamento dos dados

Para a preparação dos dados para os modelos, realizou-se um pré-processamento comum a todos os modelos. Neste pré-processamento, inicialmente, converteu-se os valores das análises médicas (ALB, ALP, ALT, AST, BIL, CHE, CHOL, CREA, GGT, PROT) em valores numéricos, pois ao ler o CSV, os atributos vinham representados como *strings*.

Em seguida, removeu-se a coluna 'id', uma vez que se tratava de um identificador único de cada paciente e não continha informações relevantes para a previsão. O dia e o mês de nascimento também foram removidos, pois não eram atributos relevantes para prever o resultado. Além disso, decidiu-se remover a coluna do ano de nascimento, pois já tínhamos o atributo idade e não havia a necessidade de mantê-lo, uma vez que seria redundante.

Para corrigir o atributo sexo, que originalmente possuía três valores distintos ('m', 'f' e 'mm'), utilizou-se o nodo *Rule Engine*, e corrigiu-se o género 'mm' para 'm', pois é provável que tenha sido um erro de digitação. Esta correção foi realizada com base na premissa de que 'mm' provavelmente foi inserido incorretamente e deveria ser interpretado como 'm'.

Por fim normalizou-se os dados utilizando o algoritmo *min-max*, que ajusta os valores de forma a estarem representados no intervalo de 0 a 1. Desta forma garante-se que todas a variáveis estão na mesma escala, o que facilita a análise e tratamento dos modelos.

4.4 Modelação

Após a exploração de dados e o pré-processamento terem sido realizados, resta-nos então criar modelos de Machine Learning de modo a prever a nossa Categoria. Para isso, além do pré-processamento realizado anteriormente, fizemos uma preparação de dados diferente para cada algoritmo, esta preparação traduziu-se em tratar os *missing values* de várias formas (remover as linhas, substituir pela média, mediana e interpolação linear), de modo a perceber como estes influenciavam na previsão do modelo. Para cada algoritmo utilizamos "*hold-out validation*" e "*cross validation*". Além disso, decidimos também explorar os hiperparâmetros de cada nó, de modo a observar a sua influência no desempenho dos modelos. Para isso utilizamos o nodo *"Parameter Optimization Loop"* que nos permite testar vários hiperparâmetros de cada nodo, por exemplo para o "Decision Tree Leaner" podemos fazer vários testes utilizando o Gini Index ou o Gain Ratio, No Pruning ou Pruning MDL, e podemos alterar também o tamanho das partições dos nodos "Partititoning" e "X-Partitioner". Testamos algoritmos de classificação, regressão e ainda clustering, fazendo tabelas, de forma a poder comparar e avaliar os resultados. Devido a estas tabelas serem bastante extensas, vamos colocar apenas pequenos exemplos do que foi feito, e em anexo enviamos um excel com as tabelas completas.

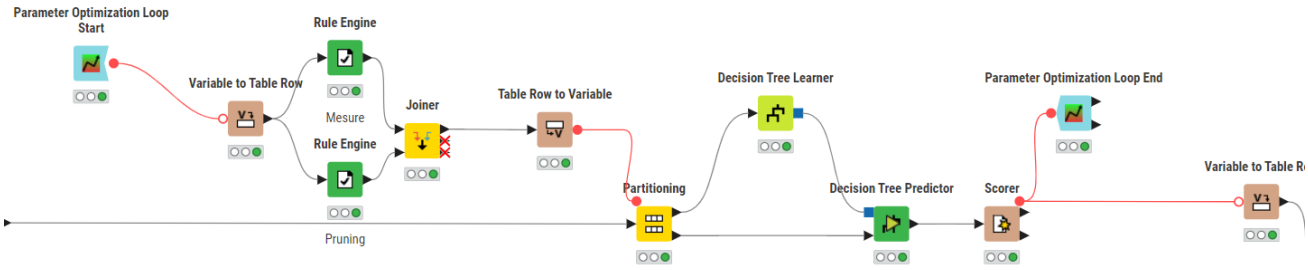


Figura 18 Exemplo de um modelo utilizando o Decision Tree e o Parameter Optimization Loop

4.4.1 Algoritmos de Classificação

Visto este problema ser classificado como um problema de classificação, começamos por explorar algoritmos para este tipo de problemas, entre eles Decision Tree, Logistic Regression, Random Forest Tree e Gradient Boosted Tree. Como dito anteriormente fizemos vários testes para o tratamento de missing values e testamos vários hiperparâmetros dos nodos. Com base nos vários testes que fizemos construímos esta tabela (que está incompleta, devido a ser bastante extensa).

Algoritmo	Missing Values	HiperParâmetros	Hold-Out Validation		Cross Validation	
			Accuracy	k	Accuracy	k
Decision Tree	Remover Linhas	**Partitioning:** 80/20 Stratified **| X-Partitioner:** 20 validations Stratified **|Decision Tree:** No prunning | Gain Ratio	94.07	0.629	93.04	0.637
		Partitioning: 80/20 Stratified **| X-Partitioner:** 20 validations Stratified **|Decision Tree:** No prunning | Gini Index	93.22	0.643	92.53	0.607
		Partitioning: 80/20 Stratified **| X-Partitioner:** 20 validations Stratified **|Decision Tree:** Prunning MDL | Gain Ratio	93.22	0.577	92.87	0.576
		Partitioning: 80/20 Stratified **| X-Partitioner:** 20 validations Stratified **|Decision Tree:** Prunning MDL |Gini Index	91.53	0.496	91.68	0.518
	Média	**Partitioning:** 80/20 Stratified **| X-Partitioner:** 20 validations Stratified **|Decision Tree:** No prunning | Gain Ratio	93.5	0.719	90.24	0.586
		...				
	Mediana	**Partitioning:** 80/20 Stratified **| X-Partitioner:** 20 validations Stratified **|Decision Tree:** No prunning | Gain Ratio	93.5	0.719	90.24	0.586
		...				
	Interpolação Linear	**Partitioning:** 80/20 Stratified **| X-Partitioner:** 20 validations Stratified **|Decision Tree:** No prunning | Gain Ratio	91.06	0.614	91.22	0.628
		...				
Logistic Regression	Remover Linhas	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 100 epots	94.59	0.685	93.55	0.599
		Partitioning: 75/25 Stratified | **X-Partitioner:** 20 validations Stratified | **Logistic Regression:** 400 epots	95.95	0.779	95.42	0.743
		Partitioning: 80/20 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 100 epots	94.07	0.66	93.72	0.613
		Partitioning: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Logistic Regression:** 400 epots	96.61	0.821	95.08	0.724
	Média	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 100 epots	92.21	0.631	92.52	0.627
		...				
	Mediana	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 100 epots	92.21	0.631	92.52	0.627
		...				
	Interpolação Linear	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 100 epots	92.86	0.67	92.68	0.635
		...				
Random Forest Tree	Remover Linhas	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 70 levels	97.46	0.866	95.08	0.729
		Partitioning: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 50 levels	96.61	0.828	95.08	0.729
	Média	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 70 levels	92.68	0.664	93.5	0.7
		Partitioning: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 50 levels	90.24	0.469	93.5	0.7
	Mediana	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 70 levels	91.87	0.658	93.82	0.715
		Partitioning: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 50 levels	91.06	0.563	93.82	0.715
	Interpolação Linear	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 70 levels	91.87	0.573	93.66	0.713
		Partitioning: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Random Forest:** 30 models | 50 levels	95.12	0.769	93.66	0.713
Gradient Boosted Tree	Remover Linhas	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Gradient Boosted:** 4.1 learning rate | 70 models | 10 levels	98.31	0.917	93.89	0.643
		...				
	Média	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Gradient Boosted:** 4.1 learning rate | 70 models | 10 levels	91.87	0.588	91.06	0.568
		...				
	Mediana	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Gradient Boosted:** 4.1 learning rate | 70 models | 10 levels	94.31	0.739	91.06	0.568
		...				
	Interpolação Linear	**Partitioning**: 80/20 Stratified | **X-Partitioner:** 20 validations Stratified | **Gradient Boosted:** 4.1 learning rate | 70 models | 10 levels	93.5	0.692	90.89	0.563
		...				

De uma forma geral, podemos verificar que para o tratamento dos *missing values*, retirar as linhas com os mesmos, acaba por dar melhores resultados. Além disso verificamos que com Hold-Out Validation tivemos melhor resultados do que com Cross-Validation.

O algoritmo Gradiente Boosted Tree, destaca-se como tendo o melhor resultado com uma accuracy de 98.31% e consequentemente, obteve o maior coeficiente de Cohen's Kappa, com um valor de 0.917. Este resultado obtido foi com um learning rate de 4.1, 70 models e 10 levels.

4.4.2 Algoritmos de Regressão

Além dos algoritmos de classificação, testamos também modelos de regressão, para isso tivemos de transformar a o atributo Category em valores numéricos. Para isso atribuímos um valor a cada Categoria utilizando o Rule Engine.

Algoritmo	Missing Values	HiperParâmetros	Hold-Out Validation				Cross Validation			
			R^2	MAE	MSE	RMSE	R^2	MAE	MSE	RMSE
Linear Regression	Remover Linhas	**Partitioning:** 75/25 Draw Randomly | **X-Partitioner:** 10 validations Stratified	0.759	0.066	0.015	0.123	0.639	0.074	0.02	0.142
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified	0.759	0.074	0.018	0.136	0.636	0.074	0.02	0.142
	Média	**Partitioning:** 75/25 Draw Randomly | **X-Partitioner:** 10 validations Stratified	0.285	0.099	0.034	0.184	0.552	0.102	0.031	0.176
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified	0.171	0.098	0.035	0.188	0.555	0.102	0.031	0.176
	Mediana	**Partitioning:** 75/25 Draw Randomly | **X-Partitioner:** 10 validations Stratified	0.285	0.099	0.034	0.184	0.553	0.102	0.031	0.176
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified	0.171	0.098	0.035	0.188	0.555	0.102	0.031	0.175
	Interpolação Linear	**Partitioning:** 75/25 Draw Randomly | **X-Partitioner:** 10 validations Stratified	0.32	0.098	0.032	0.18	0.55	0.103	0.031	0.176
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified	0.193	0.097	0.035	0.186	0.557	0.012	0.031	0.175
Polynomial Regression	Remover Linhas	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 2	0.703	0.082	0.02	0.141	0.163	0.081	0.047	0.216
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 3	-2.313	0.133	0.222	0.471	-3.123	0.095	0.23	0.479
	Média	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 2	0.539	0.144	0.04	0.201	0.11	0.104	0.061	0.248
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 3	0.25	0.144	0.066	0.256	-3.961	0.117	0.343	0.585
	Mediana	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 2	0.54	0.144	0.04	0.201	0.11	0.104	0.062	0.248
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 3	0.261	0.113	0.065	0.254	-4.286	0.118	0.365	0.604
	Interpolação Linear	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 2	0.496	0.117	0.044	0.21	0.036	0.106	0.067	0.258
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **Polynomial Learner:** degree 3	0.508	0.104	0.043	0.208	0.273	0.093	0.05	0.0224
Redes neuronais (RProp)	Remover Linhas	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **RProp:** 50 iterations | 5 hidden layers | 10 neurons	0.957	0.018	0.003	0.053	0.886	0.025	0.006	0.077
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **RProp**: 50 iterations | 9 hidden layers | 10 neurons	0.829	0.08	0.011	0.105	0.654	0.049	0.018	0.135
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **RProp:** 50 iterations | 5 hidden layers | 6 neurons	0.926	0.035	0.005	0.069	0.795	0.042	0.011	0.104
		Partitioning: 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **RProp**: 200 iterations | 5 hidden layers | 10 neurons	0.912	0.02	0.006	0.076	0.644	0.055	0.025	0.157
	Média	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **RProp:** 50 iterations | 5 hidden layers | 10 neurons	0.545	0.062	0.027	0.165	0.886	0.025	0.006	0.077
		...								
	Mediana	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **RProp:** 50 iterations | 5 hidden layers | 10 neurons	0.477	0.072	0.031	0.177	0.691	0.062	0.021	0.146
		...								
	Interpolação Linear	**Partitioning:** 80/20 Draw Randomly | **X-Partitioner:** 20 validations Stratified | **RProp:** 50 iterations | 5 hidden layers | 10 neurons	0.565	0.061	0.026	0.162	0.658	0.066	0.024	0.154
		...								

Aqui voltamos a observar que remover as linhas com missing values, e utilizar hold-out validation dá melhores resultados.

As análises de regressão polinomial e linear revelaram-se inadequadas para prever os resultados devido à baixa precisão observada. Os valores de MAE, MSE e RMSE também possuem valores bastantes altos, com alguns valores de RMSE superiores a 0.5 (normalizados entre 0 e 1), indicando a presença de muitos erros grandes.

O destaque, no entanto, vai para as redes neurais, que apresentaram os R^2 mais elevados e os valores mais baixos de MAE, MSE e RMSE, o que indica que houve uma boa adaptação aos dados. Adicionalmente, descobrimos que aumentar o número de iterações nem sempre resulta em melhores resultados, assim como aumentar o número de camadas ocultas e neurónios. O melhor resultado foi obtido com 50 iterações, 5 camadas ocultas e 10 neurónios.

4.4.3 Clustering

Utilizou-se ainda algoritmos de aprendizagem não supervisionada neste caso, o K-means, atribuindo um cluster a cada uma das categorias, neste caso foram utilizados 5 clusters. De forma a balancear os dados para os clusters terem tamanhos iguais, utilizou-se o nodo SMOTE.

Algoritmo	Missing Values	HiperParâmetros	Hold-Out Validation		Cross Validation	
			Accuracy (%)	k	Accuracy (%)	k
K-means	Remover Linhas	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 1000 iterations	60.49	0.506	58.17	0.475
		Partitioning: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 2000 iterations	56.53	0.457	63.88	0.549
	Média	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 1000 iterations	46.03	53.97	54.68	0.426
		Partitioning: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 2000 iterations	54.27	0.429	58.60	0.477
	Mediana	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 1000 iterations	56.07	0.451	61.05	0.507
		Partitioning: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 2000 iterations	64.32	0.554	54.31	0.430
	Interpolação Linear	**Partitioning**: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 1000 iterations	58.02	0.475	56.18	0.454
		Partitioning: 75/25 Stratified | **X-Partitioner:** 10 validations Stratified | **Logistic Regression:** 2000 iterations	61.47	0.518	62.55	0.533

Olhando por exemplo, para o número de iterações, não conseguimos retirar boas conclusões em como isso influencia o modelo, pois em alguns casos dá melhor resultado e noutros pior, a mesma dúvida fica entre utilizar Hold-Out ou Cross Validation, uma vez que os resultados são semelhantes.

De forma geral, podemos verificar que este algoritmo não é adequado para o nosso problema.

4.5 Avaliação dos Resultados

Olhando para os vários resultados obtidos, podemos dizer que os algoritmos de classificação revelaram-se adequar melhor a este problema, o que era o expectável, visto este ser um problema de classificação. Os algoritmos de regressão não se mostraram serem adequados, com a exceção, das redes neuronais que conseguiram resultados medianos, mas não tão bons como os modelos de classificação. E os algoritmos com clustering, neste caso apenas foi utilizado o K-means, também não mostraram ser adequados.

Os algoritmos que mais se destacaram foram os que são baseados em árvores de decisão, tendo o Gradient Boosted Tree sido o algoritmo com o melhor resultado entre todos os algoritmos realizados. Além disso, a utilização de hold-out validation acabou por dar melhores resultados em relação ao cross-validation.

Observou-se ainda, como a preparação de dados, possui uma grande influência em cada modelo, neste caso verificou-se que remover as linhas para os missing values, acaba sempre por dar melhores resultados, e também se pode observar como é que os hiperparâmetros têm bastante influencia nos resultados obtidos de cada algoritmo.

5 Carros Usados

5.1 Estudo do Negócio

O objetivo deste projeto é desenvolver modelos para prever o preço de carros usados com base em características dos mesmos. As metas neste dataset passam por:

1- Analisar e Exploração de Dados

- Realizar uma análise exploratória para entender as características do dataset de carros, utilizando gráficos para identificar padrões, tendências e anomalias nos dados.
- Explorar relações dos diferentes atributos como preço, cilindrada, fabricante, entre outros.

2- Preparar os Dados

- Realizar um tratamento de dados.
- Normalizar as variáveis numéricas para melhorar a eficácia dos modelos a prever.

3- Desenvolver modelos de aprendizagem automática

- Utilização de técnicas de *machine learning* para prever o preço dos veículos

4- Validar e comparar os modelos

- Comparar os modelos desenvolvidos em termos de precisão dos resultados obtidos.

Este projeto pode ser classificado como um problema de regressão, dado que o atributo que pretendemos prever, o preço dos carros, é um valor contínuo.

5.2 Exploração dos Dados

Este dataset possui 426880 linhas e 26 colunas. As linhas representam informações de vendas de carros usados no mercado dos Estados Unidos, e as colunas são representadas pelas seguintes categorias:

1. **id:** Identificador único para cada entrada no dataset.
2. **url:** URL associado ao anúncio.
3. **region:** Região onde o veículo está localizado.
4. **region_url:** URL da região onde o veículo está listado.
5. **price:** Preço do veículo.
6. **year:** Ano de fabricação do veículo.
7. **manufacturer:** Fabricante ou marca do veículo.
8. **model:** Modelo do veículo.
9. **condition:** Condição do veículo (usado, novo, etc.).
10. **cylinders:** Número de cilindros do motor.
11. **fuel:** Tipo de combustível utilizado pelo veículo.
12. **odometer:** Leitura do odómetro, indicando a quilometragem do veículo.
13. **title_status:** O status legal do veículo.
14. **transmission:** Tipo de transmissão do veículo (automático, manual, etc.).
15. **VIN:** Número de identificação do veículo (*Vehicle Identification Number*).
16. **drive:** Tipo de tração do veículo (dianteira, traseira, integral, etc.).

17. **size:** Tamanho do veículo (compacto, médio, grande, etc.).
18. **type:** Tipo de carroceria do veículo.
19. **paint_color:** Cor da pintura do veículo.
20. **image_url:** URL da imagem associada ao anúncio do veículo.
21. **description:** Descrição do veículo fornecida no anúncio.
22. **county:** País onde o veículo está localizado.
23. **state:** Estado onde o veículo está localizado.
24. **lat:** Latitude da localização do veículo.
25. **long:** Longitude da localização do veículo.
26. **posting_date:** Data em que o anúncio foi publicado.

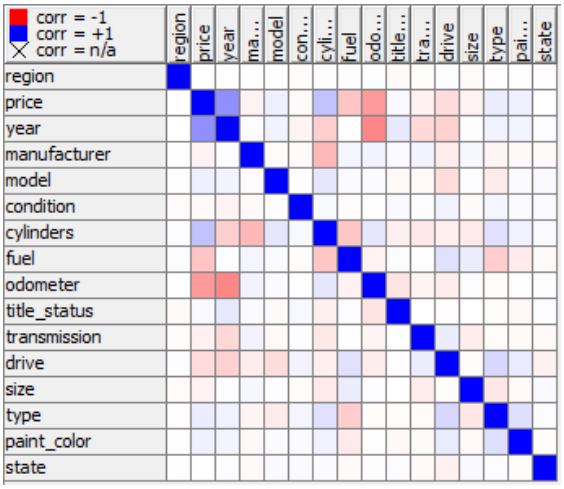


Figura 19 - Correlação dos Dados

Antes de passar para a exploração de cada um dos atributos, analisou-se a matriz de correlação, para perceber quais atributos influenciam o preço, e verificou-se que o ano, cilindro, odómetro e combustível, têm uma influência significativa no preço do veículo.

5.2.1 Manufacturer

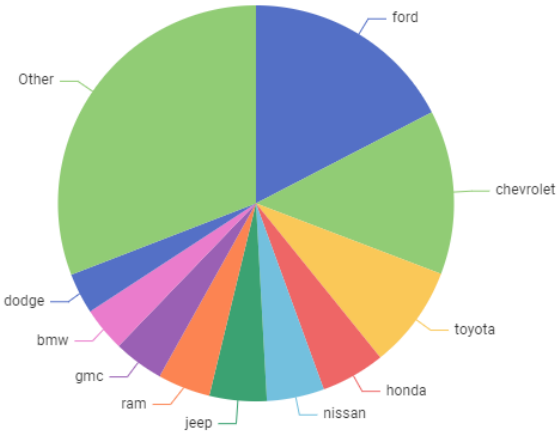


Figura 20 - Distribuição Marcas

No que diz respeito às marcas presentes no dataset, observa-se a presença de 42 marcas distintas, sendo que as mais proeminentes em termos de vendas são a Ford, Chevrolet e Toyota.

Relativamente à qualidade dos dados, identificou-se a presença de aproximadamente 17 mil missing values, correspondendo a cerca de 4% do dataset. Em virtude da relevância destes dados e visando preservar a integridade da análise, optou-se por excluir as linhas correspondentes. Contudo, nos demais atributos, não foram observados problemas de integridade ou consistência nos dados, não sendo necessária qualquer intervenção para o tratamento nessa categoria.

5.2.2 Model

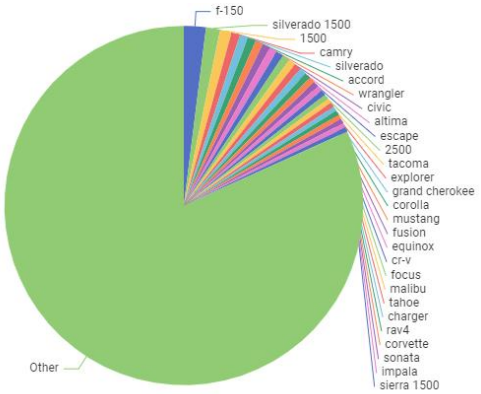


Figura 21 - Distribuição Modelos

No atributo dedicado aos modelos presentes dataset, verificou-se que existe uma vasta diversidade de dados, totalizando aproximadamente 23 mil valores únicos, provavelmente sendo resultado da inclusão de modelos por parte dos utilizadores.
De forma validar e consolidar estes dados, desenvolveu-se um script em *Python* para extrair informações de um site de vendas de veículos usados nos Estados Unidos e que permitiu identificar cerca de 3400 modelos distintos. No capítulo do pré processamento (5.3.) irá ser abordado em mais detalhe como foram obtidos melhores resultados, esperando-se remover casos como o exemplo abaixo, do modelo Tacoma, em que o dataset em questão apresenta uma grande diversidade de nomes para o mesmo modelo.

20569	Row...	tacoma	2582
20635	Row...	tacoma access cab pickup	474
20663	Row...	tacoma double cab pickup	253
20637	Row...	tacoma access cab sr5	243
20652	Row...	tacoma double cab	216
20636	Row...	tacoma access cab sr	149
20618	Row...	tacoma 4x4	128
20668	Row...	tacoma double cab trd	126
20700	Row...	tacoma prerunner	107
20630	Row...	tacoma access cab	95
20638	Row...	tacoma access cab trd	88
20732	Row...	tacoma sr5	83
20598	Row...	tacoma 4wd	64
20665	Row...	tacoma double cab sr5	64
20782	Row...	tacoma trd sport	61
20767	Row...	tacoma trd off road 4x4	45
20785	Row...	tacoma trd sport 4x4 gas	44

Figura 22 - Diferentes nomes usados para o mesmo modelo

5.2.3 Ano

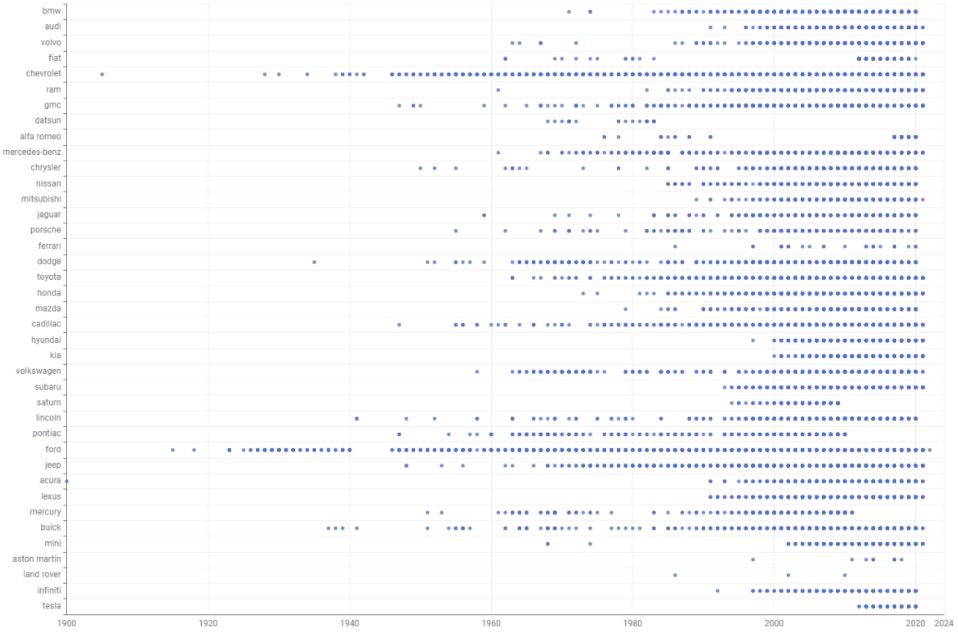


Figura 23 - Dispersão dos anos pelas Marcas

Ao analisar o gráfico, percebe-se que certas marcas ainda estão em produção e outras já não, como é o caso da Saturn.

Por outro lado, ao examinar o gráfico que relaciona o ano do veículo com o preço, observou-se uma relação inversamente proporcional entre o preço e a idade do veículo. No entanto, chega-se a um ponto em que os preços dos carros tendem a estagnar ou até mesmo a valorizar, devido ao mercado de carros clássicos.

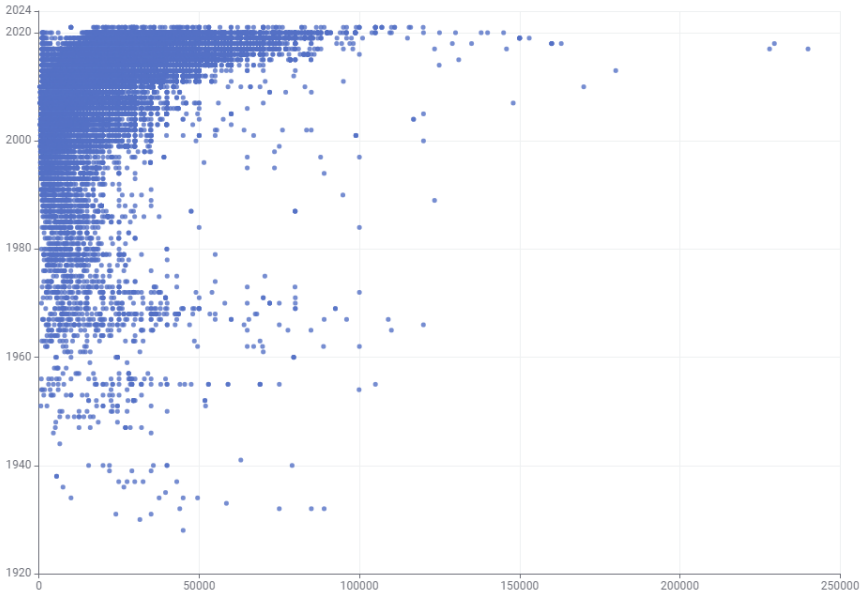


Figura 24 - Dispersão do preço em relação ao Ano

5.2.4 Preço

Figura 25 - Dispersão do preço pelas marcas

Ao analisar o gráfico, podemos observar uma grande variação de preços no mercado de carros usados. Essa variação pode ser atribuída ao facto de carros, mesmo sendo caros, como a Porsche, perderem o seu valor se forem antigos ou tiverem acumulado um grande número de quilômetros. Além disso, carros que servem apenas para peças também podem fazer com que o preço desça significativamente.

Além disso, é possível identificar que determinadas marcas estão mais relacionadas a um mercado de luxo, o que é evidente pela concentração de pontos mais à direita do gráfico.

Por fim, é possível observar que a Ford tem uma grande variedade de modelos, o que resulta numa presença em todas as faixas de preço, refletindo-se no gráfico.

Ao examinar o dataset, também se detetaram valores que foram introduzidos de forma aleatória. Esta irregularidade tornou-se aparente quando se compararam modelos idênticos, revelando diferenças consideráveis nos preços.

Por outro lado, verificou-se a ocorrência inversa, com valores estabelecidos como zero. Dado que o objetivo principal consiste em calcular o preço de veículos usados, tais valores não contribuem para o treino do modelo e, portanto, foram removidas as linhas do dataset.

5.2.5 Odómetro

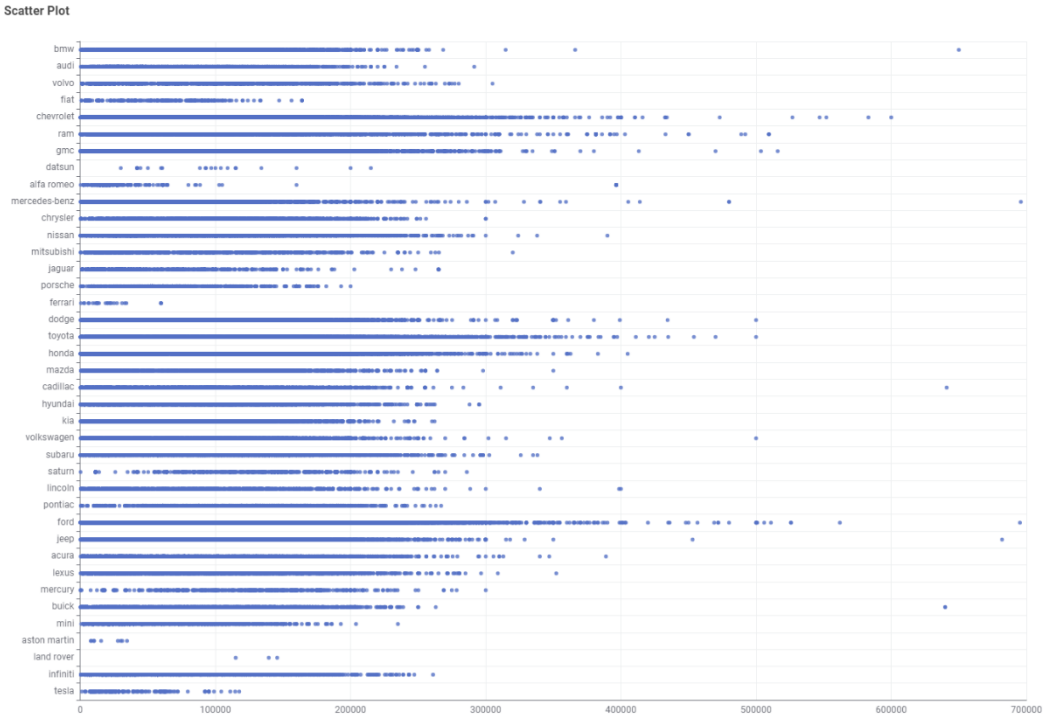


Figura 26 - Dispersão das milhas pelas Marcas

A análise dos valores de milhas revelou a existência de 4.400 valores em falta. Além disso, observou-se uma distribuição irregular, com um valor mínimo de 0 milhas e um valor máximo de 10.000.000 milhas.

Estes valores extremos, poderão aparecer devido aos utilizadores inserirem valores aleatórios. É importante notar que um automóvel, mesmo que seja novo, não tem 0 milhas devido ao controle de qualidade por parte da marca. Além disso, 10 milhões de milhas são praticamente impossíveis, considerando o tempo de vida útil dos automóveis. Após a correção dos valores, os mesmos situam-se num mínimo de 200 milhas e num máximo de 900.000 milhas, com uma média de aproximadamente 94 mil milhas.

Ao analisar o gráfico observou-se que as marcas mais vendidas estão associadas a automóveis com mais milhas, também comparando com o capítulo anterior, verificou-se que as marcas de automóveis mais caras estão, por norma, associadas a menos milhas no odómetro.

5.2.6 Fuel

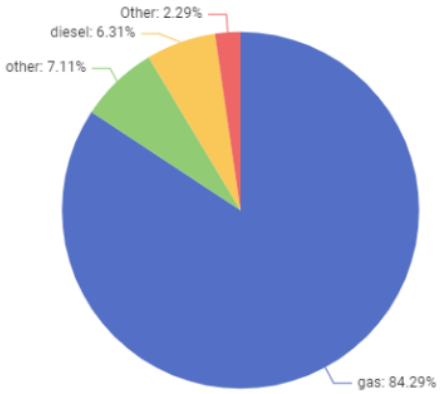


Figura 27 - Distribuição do tipo de Combustível

No que toca ao tipo de combustível, constatou-se que existem 3013 valores em falta. Além disso, foram identificadas 5 categorias diferentes de combustível: gasolina, diesel, elétrico, híbrido e outro.

Com o objetivo de salvaguardar a integridade da análise, foi decidido implementar uma estratégia para lidar com *os missing values*, reconhecendo a sua importância para a previsão dos preços.

Analisando os dados sabemos que cerca de 84% dos carros são a gasolina, e depois de realizar a correção conseguimos manter as proporções semelhantes, sendo isto importante pois desta forma mantemos a distribuição da classe.

5.2.7 Type

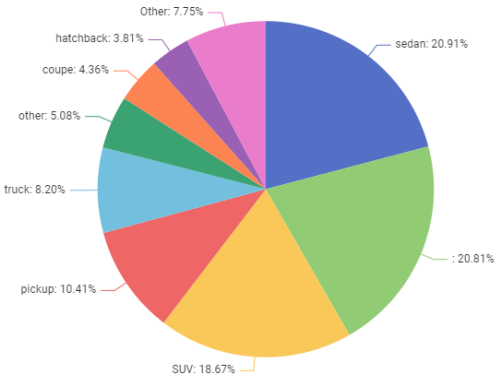


Figura 28 - Distribuição do tipo de carro

No que diz respeito aos dados referentes ao tipo de veículo presentes no conjunto de dados, identificou-se a ausência de 92.858 valores. Além disso, foram observadas 13 categorias distintas de tipos de veículos: *SUV, bus, convertible, coupe, hatchback, mini-van, offroad, other, pickup, sedan, truck, van, Wagon*.

Com o propósito de assegurar a integridade da análise, foi deliberadamente implementada uma estratégia para lidar com os valores ausentes, reconhecendo sua relevância para a previsão de preços, isto devido a correlação verificada na imagem.

5.2.8 Transmission

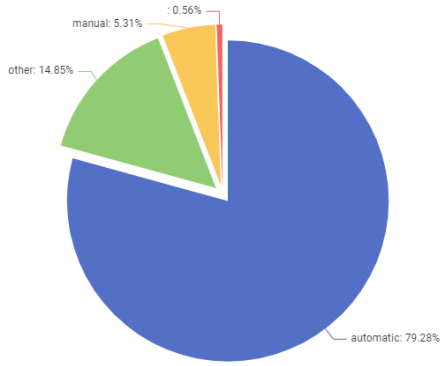


Figura 29 - Distribuição do tipo de Transmissão

Observamos a falta de 2.556 valores nos dados de transmissão dos veículos. Esses dados estão divididos em três categorias: automática, manual e outra.

Analisando os dados podemos concluir que o tipo de transmissão mais usada é automático, e ao fazermos o tratamento de dados conseguimos manter a proporção das classes semelhante.

5.2.9 Cylinder

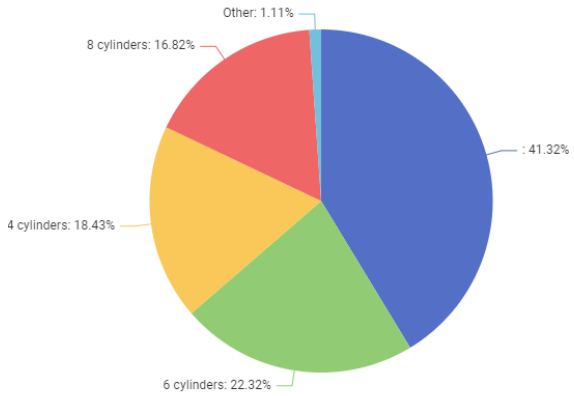


Figura 30 - Distribuição do número de Cilindros

Identificamos a ausência de 177.678 valores nos dados sobre o número de cilindros dos veículos. Esses dados estão agrupados em oito categorias distintas: 6 cilindros, 4 cilindros, 8 cilindros, 5 cilindros, 10 cilindros, outro, 3 cilindros e 12 cilindros.

5.2.10 Atributos extra: Size, Drive

Em relação aos dados de tamanho dos veículos no conjunto de dados, identificamos a falta de 306.361 valores. Esses dados estão categorizados em quatro grupos distintos: *full-size, mid-size, compact, sub-compact*. A análise dos dados revelou que o tamanho do veículo pode ser um fator significativo na determinação do preço. Portanto, é crucial lidar com os valores ausentes de forma adequada para preservar a integridade da análise.

Já em relação a tração dos veículos foram identificados 130.567 valores ausentes nos dados. Estes são classificados em três categorias: tração nas quatro rodas (4WD), tração nas rodas dianteiras (FWD) e tração nas rodas traseiras (RWD).

5.3 Pré-Processamento

Para o pré-processamento deste dataset, inicialmente, começou-se por remover colunas que não são relevantes para a previsão do preço do veículo. Estas colunas incluem ID, URL, region_url, VIN, image_url, description e posting_date, pois são informações relevantes para a gestão do site, mas não contribuem para a previsão do preço. Além disso, como só existe um país, essa informação também pode ser removida. Da mesma forma, as colunas de latitude e longitude podem ser eliminadas, visto que já temos a informação do "estado" e acabam por ser redundantes.

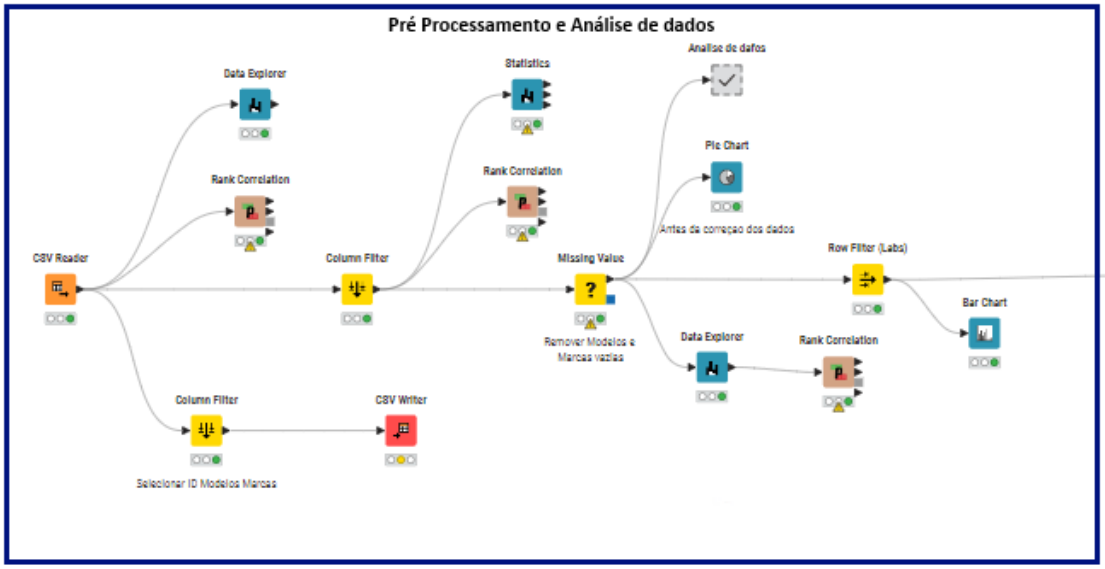


Figura 31 - Sequencia de nodos do Pré Processamento

Além disso, na análise inicial, identificámos valores que não faziam sentido, como um carro comum custar 12.345.678 $. Para corrigir isso, utilizámos *Row Filter*, evitando assim possíveis erros na análise do problema.

Além disso, devido à importância fundamental das marcas e modelos para a nossa análise e correção, decidimos remover as linhas com valores vazios nesses atributos. Dada a falta de dados disponíveis, seria praticamente impossível recuperar esses valores de forma fiável.

5.3.1 Correção dos modelos

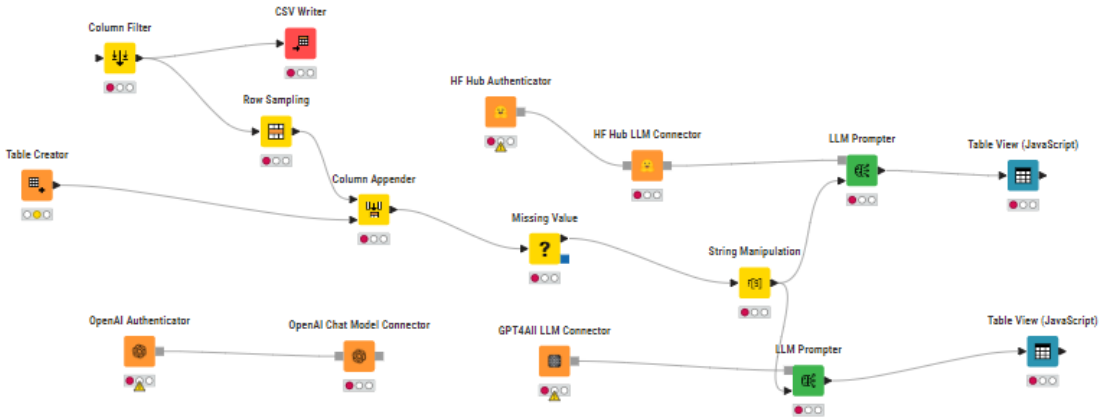


Figura 32 - Sequencia de nodos para o uso de LLM

No capítulo anterior, foi observada uma grande variedade de valores nos modelos. Na primeira tentativa, optou-se por utilizar os nós de LLM no *Knime*. Contudo, devido à necessidade de processamento do output e aos custos associados, visto que era necessário pagar pelo acesso à API após um certo número de pedidos, decidiu-se optar por uma abordagem mais convencional. Assim, recorreu-se a scripts em Python para corrigir os dados, conforme mencionado anteriormente.

```python
import pandas as pd
from difflib import get_close_matches
from fuzzywuzzy import process

with open('correct_models.csv', 'r') as file:
    reader = csv.reader(file)
    data1 = list(reader)
aligned_data = {}

def find_match(modelo_errado, modelos_corretos):
    modelo_errado = ' '.join(modelo_errado.split()[:3])
    match, score = process.extractOne(modelo_errado, modelos_corretos)
    return match if score > 80 else None

def find_match2(modelo_errado, modelos_corretos):
    modelo_errado = ' '.join(modelo_errado.split()[:3])
    #Correçao dos modelos "F's" da Ford
    match = get_close_matches(modelo_errado, modelos_corretos, n=3, cutoff=0.5)
    match = sorted(match, key=lambda x: process.extractOne(modelo_errado, x)[1], reverse=True)
    if not match:
        match = get_close_matches(modelo_errado, modelos_corretos, n=3, cutoff=0.3)
        match = sorted(match, key=lambda x: process.extractOne(modelo_errado, x)[1], reverse=True)
    return match[0] if match else None

with open('dados.csv', 'r') as file:
    reader = csv.reader(file)
    data = list(reader)
    for row in data:
        if isinstance(row[3], str) and row[3].lower() in aligned_data:
            if row[3] == "ford":
                correspondencia.append((row[0], row[3], find_match2(row[2], aligned_data[row[3]])))
            else:
                correspondencia.append((row[0], row[3], find_match(row[2], aligned_data[row[3]])))
        else:
            print(f"Marca {row[2]} não encontrada no dicionário de marcas corrigidas")
```

Figura 33 – Script da correção dos modelos

Apesar de se constatar uma correlação reduzida entre o preço e os modelos, é crucial realçar que os modelos são extremamente úteis para corrigir outras informações, como o número de cilindros, tipo de tração, entre outros atributos.

Com base nos dados obtidos anteriormente, a equipa optou por retificar a lista de modelos, recorrendo novamente ao *Python* e utilizando funcionalidades das bibliotecas Pandas, *difflib e fuzzywuzzy.*

Por meio de algoritmos de comparação de palavras, foi possível encontrar as correspondências mais adequadas, reduzindo o número de modelos únicos de 23 mil para 1167.

O processo iniciou-se com a exportação das colunas de identificação, úteis para posterior junção com as novas colunas corrigidas, bem como as marcas, que contribuíram para a obtenção de resultados mais precisos, e os modelos a corrigir. No que diz respeito ao script em si, que representa apenas um trecho do código simplificado, destacou-se o caso da Ford, onde a equipe identificou que o primeiro algoritmo não estava a produzir resultados ótimos e foi necessário recorrer a outro algoritmo, além disso, foi necessário corrigir a grafia de "f150" para "F-150", o que resultou em melhorias significativas nos resultados obtidos.

Embora este processo não tenha sido perfeito e tenha resultado em cerca de 10 mil valores vazios, é importante salientar a existência de valores redundantes, como por exemplo “150 4x4” e “150 4wd”, que possuem o mesmo significado, além disso, existe uma coluna dedicada ao tipo de tração, o que implica que o modelo deveria ser apenas “150”;

Desta forma obter-se valores mais uteis para a correção de atributos futuros sendo uma consequência que vale a pena considerando os benefícios.

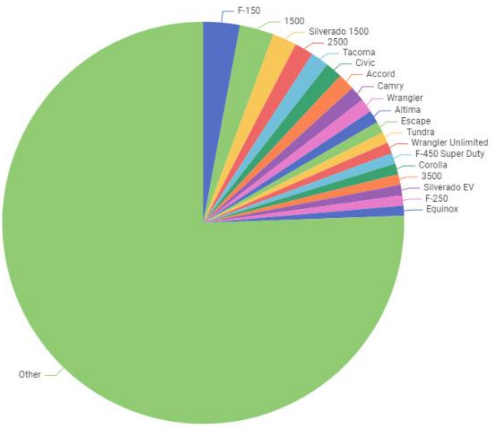


Figura 34 - Distribuição dos modelos depois da correção

5.3.2 Utilizando os modelos.

Após a correção dos modelos, revelou-se crucial para a retificação das demais informações. Dado que algumas colunas apresentam uma correlação significativa com os modelos, optámos por utilizar esta relação a nosso favor, permitindo-nos corrigir as informações sem alterar as proporções originais.

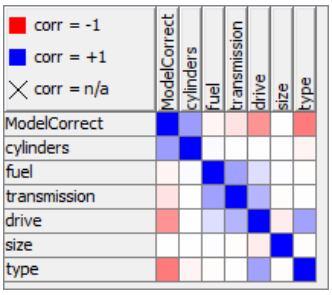


Figura 35 - Correlação dos modelos com os atributos a corrigir

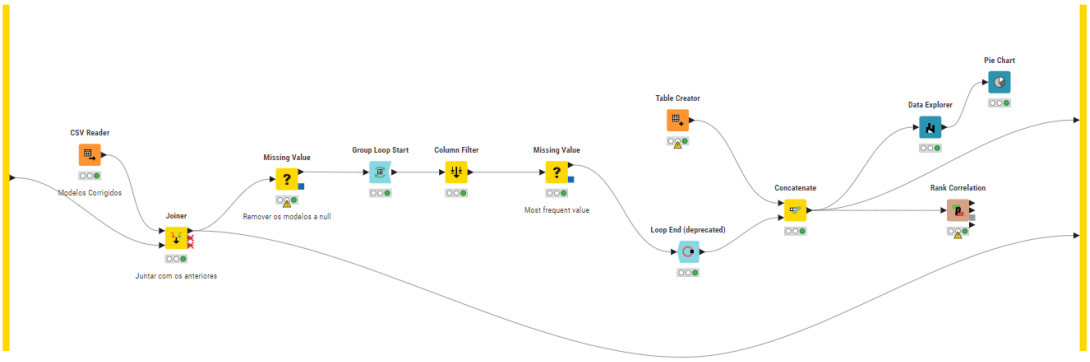


Figura 36 - Sequencia de nodos da correção dos modelos

Para abordar os dados em falta, implementámos um ciclo que percorre cada modelo individualmente. Durante este processo, procedemos à correção dos valores em falta. No caso de se tratar de um valor do tipo categoria, optámos por substituí-los pelo valor mais frequente, para esse atributo no modelo em questão. Por outro lado, para valores contínuos, utilizámos a média, garantindo assim uma abordagem mais realista à correção dos dados.

5.3.3 Utilizando a Marca

Após a correção utilizando os modelos, identificámos ainda a presença de alguns valores em falta. Neste sentido, após a correção, optámos por implementar um novo ciclo, semelhante ao anterior, mas agora focado nas marcas. Desta forma, conseguimos eliminar por completo os valores em falta.

Na fase de gerar os modelos eremos testar se foi a melhor abordagem, ou seria melhor eliminar as linhas com valores em falta.

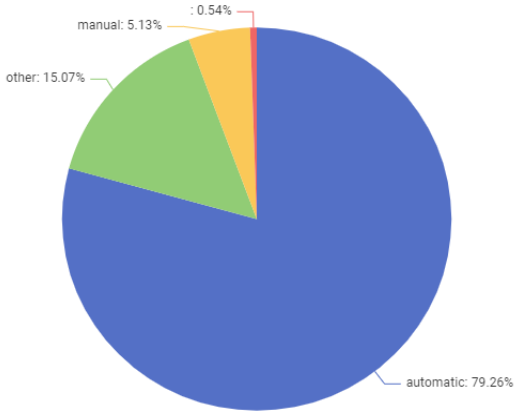


Figura 38 - Distribuição da transmissão apos tratamento de dados

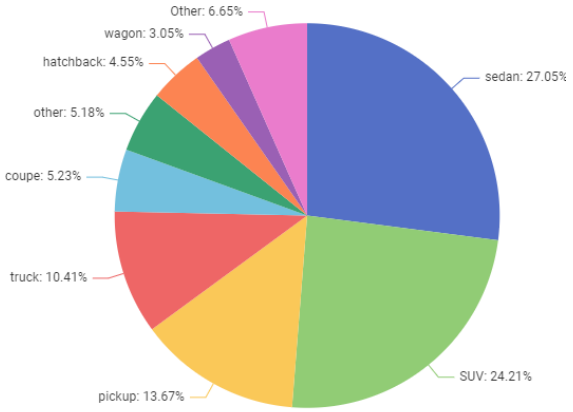


Figura 40 - Distribuição do tipo apos tratamento de dados

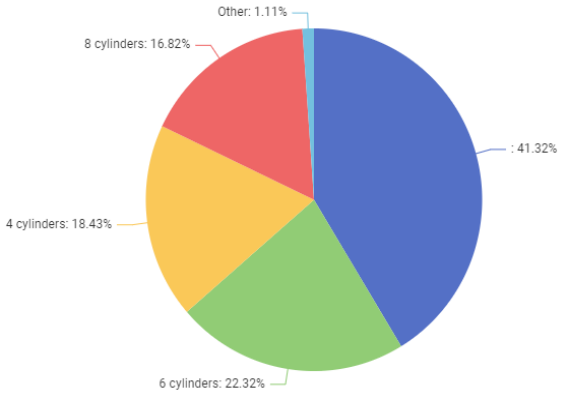


Figura 37 - Distribuição dos cilindros apos tratamento de dados

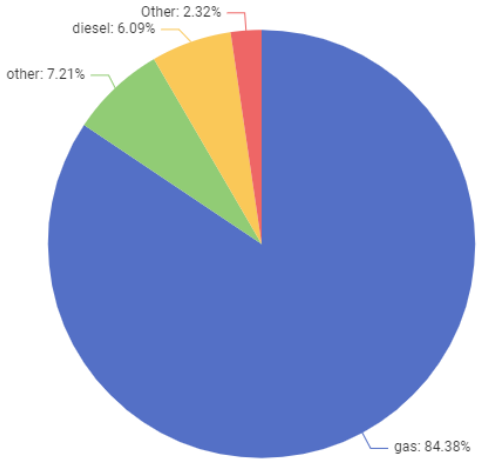


Figura 39 -Distribuição do combustível apos tratamento de dados

5.4 Modelação

Para determinar os melhores atributos, optamos por realizar uma sequência de nós que selecionariam os atributos mais relevantes com base nos dados fornecidos.

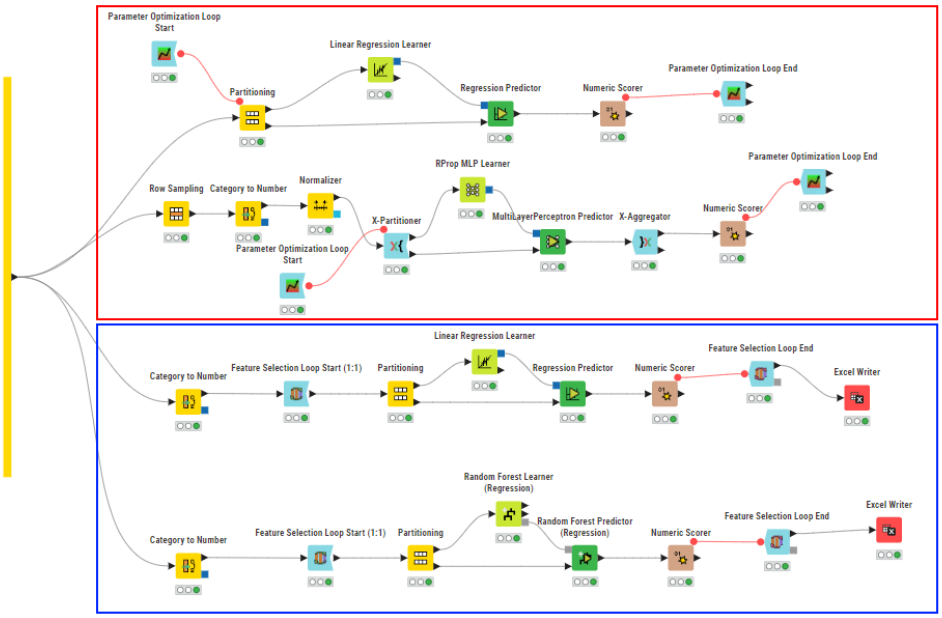


Figura 41 - Sequencia de nodos para selecionar os melhores atributos (Azul) e hipérmetros (Vermelho)

Após executar a sequência de nós, chegamos a um top 3 dos atributos, para os algoritmos de Regressão Linear e *Random Forest:*

Linear Regression		
Nr. of features	**R^2**	**Selected features**
8	0,603064	ManufactorCorret,year,cylinders,fuel,odometer,transmission,drive,type
6	0,600704	year,cylinders,fuel,odometer,drive,type
9	0,599429	ManufactorCorret,ModelCorrect,year,cylinders,fuel,odometer,transmission,drive,type
Random Forest		
Nr. of features	**R^2**	**Selected features**
9	0,921287	ManufactorCorret,ModelCorrect,year,cylinders,fuel,odometer,transmission,drive,type
7	0,898741	ManufactorCorret,ModelCorrect,year,cylinders,odometer,drive,type
7	0,896613	ManufactorCorret,ModelCorrect,year,cylinders,fuel,odometer,drive

Para simplificar a escolha, decidimos utilizar os nove atributos, pois obtivemos os melhores resultados e simplificamos a implementação.

Além disso, para o modelo mais simples, como a regressão linear, foram testados diferentes parâmetros para determinar os melhores. Também foram testados os melhores parâmetros para o *Rprop*, sendo que identificamos um bom custo-benefício com 630 interações, 3 camadas escondidas e 29 neurônios, uma vez que a partir desses valores o custo computacional se tornaria muito elevado para cada.

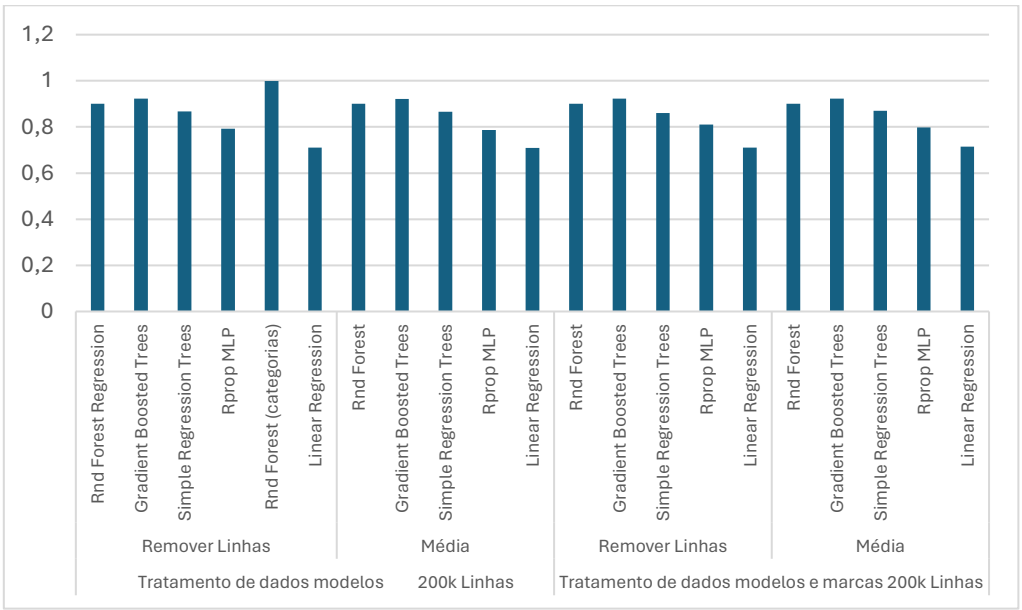
Apos chegarmos aos melhores atributos corremos diferentes algoritmos para tentarmos chegar aos melhores resultados.

Figura 42 - Sequencia de nodos para gerar os modelos para diferentes Algoritmos

No geral, os algoritmos que apresentaram os melhores resultados foram aqueles baseados em árvores de decisão. Destaca-se o algoritmo *Gradient Boosted Trees,* que se destaca por construir vários modelos fracos e refiná-los iterativamente, resultando em melhorias progressivas nos resultados. Isso é evidenciado pelos resultados superiores em comparação com o *Random Forest Regression,* que itera apenas sobre os atributos , sem considerar resultados anteriores.

Outra experiência realizada foi a criação de categorias para determinados intervalos de valores. Embora isto tenha produzido resultados promissores, com um R^2 em torno de 0,99, revelou-se enganador. Isto deve-se ao facto de que os intervalos de preço podem incluir o erro dos resultados obtidos nos modelos de regressão. Além disso, aumentar o número de categorias com intervalos de preços mais precisos tornaria computacionalmente inviável a execução dos testes nos nossos computadores. Por este motivo, esse método não foi mais adequado nos testes seguintes.

5.5 Avaliação dos resultados



Dados	Missing Values	Algoritmo	HiperParametros	Cross Validation / Hold-Out Validation				
				R^2 / Accuracy(%)	MAE	MSE	RMSE	k
Tratamento de dados modelos 200k Linhas	Remover Linhas	Rnd Forest Regression	**X-Partitioner:** 8 **Sampling:** Random	0,901	2 561,03	15 453 914.032	3931.147	
		Gradient Boosted Trees	**X-Partitioner:** 8 **Sampling:** Random	0,923	1 919.508	12 024085.963	3467.576	
		Simple Regression Trees	**X-Partitioner:** 8 **Sampling:** Random	0,867	2 152.586	20 840 919.861	4565.186	
		Rprop MLP	**X-Partitioner: 6 Sampling: Random Iter.: 663, hiddenlayer: 3 hidden neurons: 22**	0,792	0.061	0.008	0.087	
		Rnd Forest (categorias)	**Binner** 100 **Partitioning** 70/30 Draw Randomly	99.9				0.999
		Linear Regression	**Partitioning:** 70/30 Draw Randomly	0,71	4 852.248	44 456 007.102	6 667.534	
	Média	Rnd Forest	**X-Partitioner:** 8 **Sampling:** Random	0,9	2565.991	15 554 513.318	3943.921	
		Gradient Boosted Trees	**X-Partitioner:** 8 **Sampling:** Random	0,922	1925.751	12 053 417.922	3471.803	
		Simple Regression Trees	**X-Partitioner:** 8 **Sampling:** Random	0,866	2129.597	20 788 666.852	4559.459	
		Rprop MLP	**X-Partitioner: 6 Sampling: Random Iter.: 663, hiddenlayer: 3 hidden neurons: 22**	0,78612	0.06321	0.0082	0.0905	
		Linear Regression	**Partitioning:** 75/25 Draw Randomly	0,7093	4865.54	44 944 363.585	6704.0557	
Tratamento de dados modelos e marcas 200k Linhas	Remover Linhas	Rnd Forest	**X-Partitioner:** 8 **Sampling:** Random	0,90069	2556.45	15 432 123. 911	3928.37	
		Gradient Boosted Trees	**X-Partitioner:** 8 **Sampling:** Random	0,9231	1922.099	11 935 312. 117	3454.752	
		Simple Regression Trees	**X-Partitioner:** 8 **Sampling:** Random	0,86	2117.205	20 413 627.86	4518.144	
		Rprop MLP	**X-Partitioner: 6 Sampling: Random Iter.: 663, hiddenlayer: 3 hidden neurons: 22**	0,81075	0.0599	0.00723	0.08505	
		Linear Regression	**Partitioning: 66**/34 Draw Randomly	0,71	4871.77	45 144 478. 96	6718.964	
	Média	Rnd Forest	**X-Partitioner:** 8 **Sampling:** Random	0,9	2561.29	15 501 716.003	3937.221	
		Gradient Boosted Trees	**X-Partitioner:** 8 **Sampling:** Random	0,92322	1930.003	11 923 280.416	3453.01034	
		Simple Regression Trees	**X-Partitioner: 8 Sampling: Random**	0,869852	2142.690	20 347 751.846	4510.84	
		Rprop MLP	**X-Partitioner: 6 Sampling: Random Iter.: 663, hiddenlayer: 3 hidden neurons: 22**	0,79788	0.0605	0.00760	0.08723	
		Linear Regression	**Partitioning:** 75/25 Draw Randomly	0,7141	4834.813	44 048 172.344	6636.88	
Remover Dados Sem Tratamento 75 K Linhas	Remover Linhas	Rnd Forest	**X-Partitioner: 6 Sampling: Random Iter.: 663, hiddenlayer: 3 hidden neurons: 22**	0,872	1816.642	8 016 300,179	2831,397	
		Gradient Boosted Trees	**X-Partitioner:** 8 **Sampling:** Random	0,881	1731,926	7 439 497.727	2727,544	
		Simple Regression Trees	**X-Partitioner:** 8 **Sampling:** Random	0,645	2623,858	22 121 964,07	4703.399	
		Linear Regression	**Partitioning: 72**/28 Draw Randomly	0,476	4269,126	32 587 715,38	5708.565	

Ao analisar os resultados da tabela após a remoção das linhas com valores em falta, observa-se que os resultados não alcançaram um valor ideal de R^2 em comparação com os outros métodos. Embora tivesse ocorrido uma diminuição dos erros devido à redução da variedade de dados, e ao considerar todas as colunas, percebe-se que não era viável corrigi-los devido à impossibilidade de obter informações como o estado do veículo, a cor, entre outros. Portanto, além dos resultados não serem os mais satisfatórios, esta abordagem drástica não mostrou compensar.

Outra abordagem que adotamos foi o tratamento dos atributos com missing values mais interessantes para o problema, utilizando modelos e marcas. Observamos apenas que esta abordagem resultou melhorias no modelo de Machine Learning, enquanto nos demais casos, os resultados foram comparáveis à utilização exclusiva de modelos de carro para correção desses atributos.

Em relação às variáveis com maior impacto, destacam-se *o Manufactor*, o ano, o odómetro e o tipo, devido à forte correlação com o preço. Realizando um teste rápido com o algoritmo *Gradient Boosted Trees*, conseguimos um R^2 em torno de 0.82 e um erro absoluto de 3504.

Podemos concluir ainda que o melhor algoritmo para este Dataset é o *Gradient Boosted Trees*, pois, em várias configurações de atributos e correções de dados, obteve consistentemente os melhores resultados, e ainda que uma boa correção dos valores em falta pode levar a uma melhoria considerável dos resultados obtidos pelo mesmo algoritmo.

6 Conclusão

Com a conclusão deste trabalho, pudemos aprender vários conceitos de aprendizagem automática. Foi possível observar como é importante fazer uma boa exploração dos dados, assim como um consequente pré-processamento. Verificou-se que algoritmos diferentes, têm resultados melhores com preparações e tratamentos de dados diferentes.

Foi possível ainda, testar vários algoritmos que nunca tínhamos usados nas aulas, como o Random Forest e o Gradiente Boosted Tree, e testar nodos diferentes como o Feature Selection e o Parameter Optimization Loop que nos permitiram perceber quais eram os melhores atributos e hiperparâmetros a serem utilizados.

Por fim, acreditamos ter cumprido tudo que foi pedido, e aprendido mais sobre conceitos de machine learning.