Relatório ML

Arthur Augusto Claro Sardella - 2212763
Luiza Marcondes Paes Leme - 2210275
Pedro Gonçalves Mannarino - 2210617

Introdução

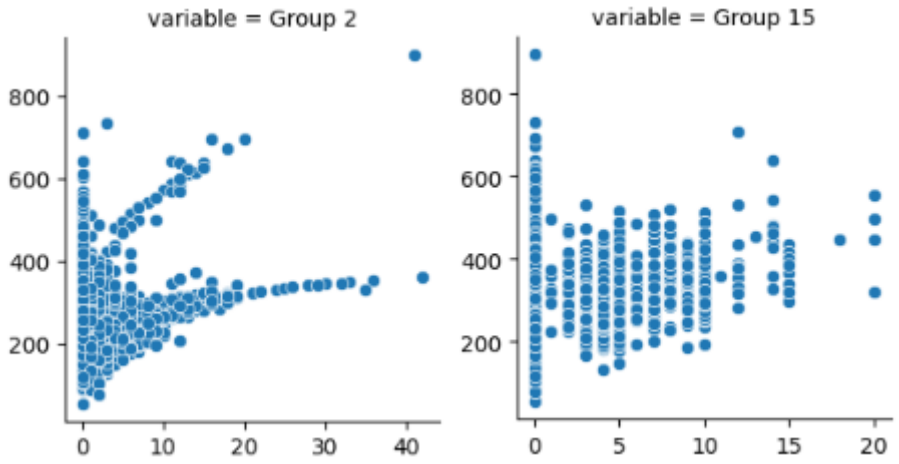
O dataset escolhido, retirado de uma competição ativa do Kaggle, apresenta uma seleção de moléculas, em que as amostras contêm uma string denotando sua representação química, sua temperatura de fusão em Kelvin, e um conjunto de diversas informações arbitrárias denominadas “Grupos”. O objetivo é conseguir estimar a temperatura de derretimento de uma nova molécula dadas as informações sobre ela.

Análise Exploratória

Análise estatística e multivariada

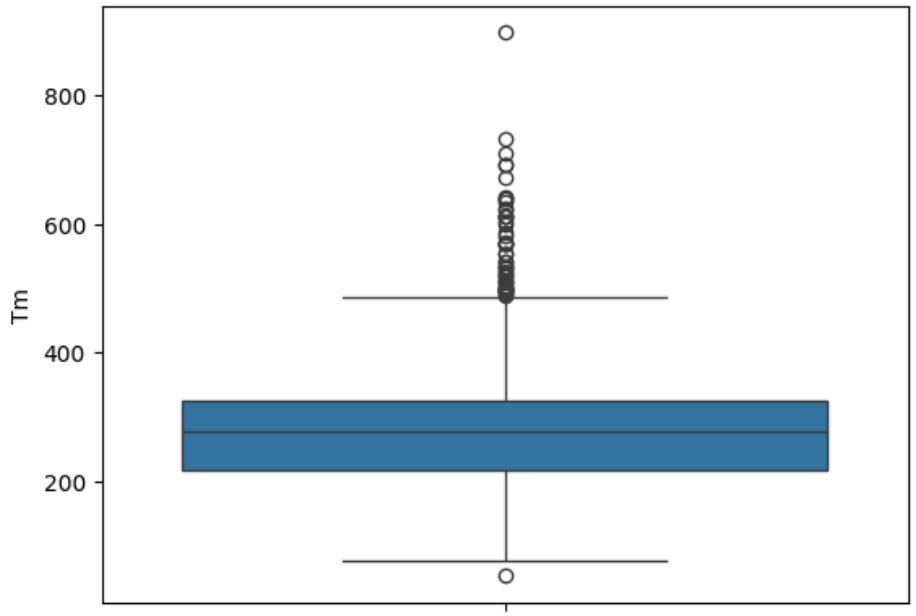
Para a análise exploratória, começamos com uma visualização de “count, mean, std, min, 25%, 50%, 75%, max” para cada uma das colunas. A partir dessa informação, conseguimos perceber que algumas colunas estavam completamente zeradas e seus dados não iriam agregar no aprendizado dos modelos futuros. Depois, utilizando um histograma de frequência de temperatura, conseguimos visualizar que a distribuição possui uma assimetria à direita, ou seja, tendendo a temperaturas mais baixas, com as mais altas sendo principalmente prováveis outliers.

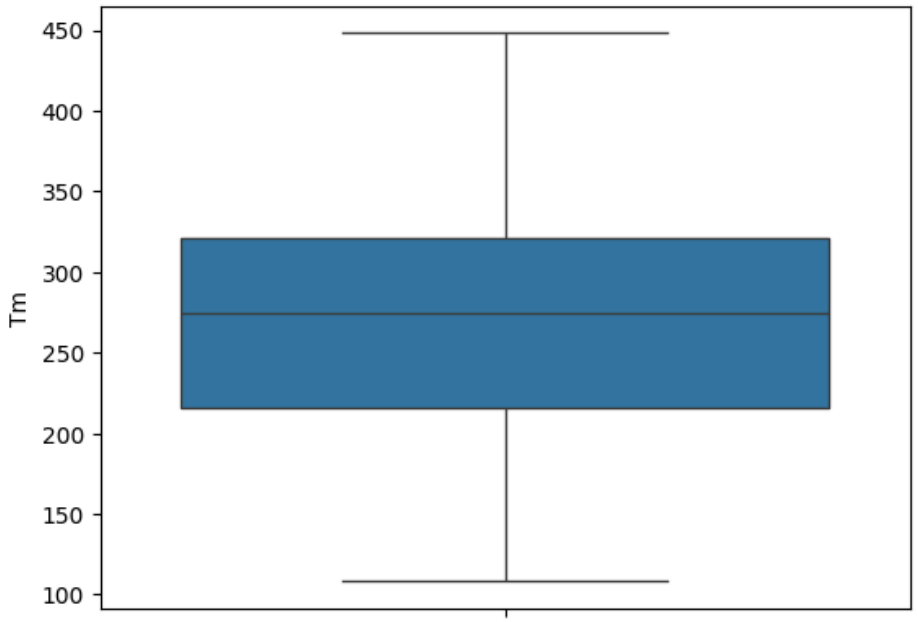
A quantidade de colunas iniciais, sendo 426, efetivamente impossibilitou certas formas de visualização, como pairplot, em especial pelo tempo de execução e pela incapacidade de visualizar os dados devido a densidade de informação. Entretanto, conseguimos construir uma relação da temperatura com todos os grupos individualmente através de um facet grid e visualizá-la, observando que, particularmente com grupo 2 e grupo 15, existe uma leve covariância entre eles e a temperatura, mas não determinamos que essa influência era significativa o suficiente para o aprofundamento dessa relação.



Limpeza de dados

Para a limpeza de dados, começamos com um boxplot da temperatura e identificamos alguns outliers, fazendo a remoção deles (saindo de 2662 para 2569 amostras) e guardando o resultado em um dataframe separado.



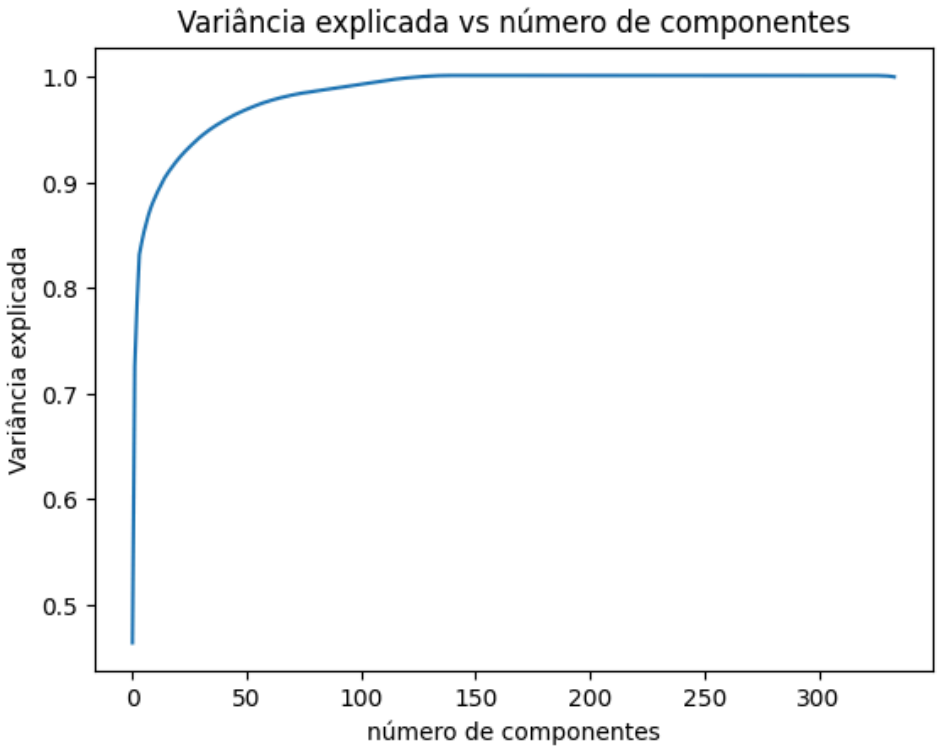


Depois, a partir desse dataframe, retiramos todas as colunas que estavam zeradas (saindo de 426 para 339 colunas), pois não apresentariam vantagem alguma durante o treinamento. Além dessas colunas, removemos mais duas, id e SMILES, que representavam, respectivamente, dados que se tratavam de identificadores das moléculas; e dados que não observamos uma forma prática de aproveitar para o treinamento dos modelos. Removemos também as poucas duplicatas (saindo de 2569 para 2565 amostras), limpando um pouco o dataframe. A esse ponto, visualizamos um novo histograma de frequências, verificando uma maior simetria nos dados. Por fim, fizemos o escalonamento desse novo dataframe, resultando no que usamos na maior parte do trabalho. O dataframe final ficou com 337 colunas e 2565 linhas ou amostras.

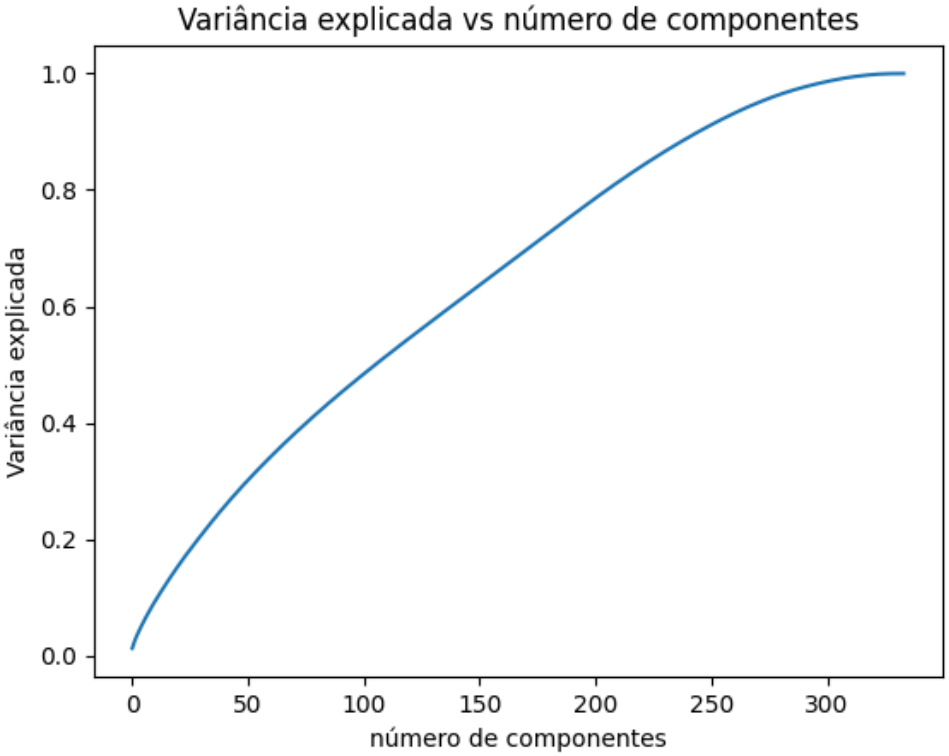
Vale destacar que testamos também a remoção de colunas consideradas ‘confusas’, com uma função auxiliar desenvolvida para detectar a porcentagem de ‘0’s presentes em cada coluna, e comparar com um dado valor para decidir se aquela coluna deveria fazer parte do retorno ou não. Nesse caso, a função foi usada para eliminar todas as colunas que eram pelo menos 99% compostas por ‘0’s (saindo de 337 para 78 colunas). Concluímos que essa remoção era muito agressiva e poderia prejudicar as análises seguintes, mas, mesmo assim, rodamos modelos de regressão e rede neural com o dataframe resultante dela para testar, confirmando um erro maior.

PCA

Posteriormente, tentamos reduzir a dimensionalidade através do PCA. Contudo, ao utilizá-lo, os modelos em que utilizamos esses dados tiveram resultados piores. Mesmo após visualizarmos o gráfico de variância explicada pela quantidade de colunas, não era óbvio o motivo desses resultados.



Após isso, tentamos rodar o PCA utilizando os dados escalonados e descobrimos que esse mesmo gráfico tomou uma forma diferente:



Isso revelou que a grandeza dos dados estava influenciando o resultado da variância explicada, e confirmou que todas (ou praticamente todas) as colunas são de fato importantes para o modelo.

Modelos

Utilizamos os métodos vistos em sala de aula para analisar os dados e aplicar variadas formas de regressão para minimizar o 'mean_absolute_error' das avaliações, tendo em vista que essa é a medida escolhida para avaliar os modelos no Kaggle. Também utilizamos o 'r2_score' para auxiliar na medição da qualidade do modelo.

Aplicamos os seguintes métodos:

- Regressão Linear
    - Linear Regression
    - Ridge
    - Lasso
    - ElasticNet
    - Ransac
    - Huber
- SVM
    - SVR Linear (dados originais, dados escalonados, PCA)
    - NuSVR (dados originais, dados escalonados)
    - SVR Não-Linear (dados originais, dados escalonados)
- Rede Neural
    - Keras

A seguir, segue o compilado de resultados dos modelos. Também é possível ver os gráficos dos resultados de cada um no repositório (as imagens são muito grandes para colocá-las aqui).

Regressores Lineares

Linear Regression

R2 score	0.6400415384178433
Mean Absolute error	33.22711496566913
Mean Absolute error with Cross Validation	31.894266131796748

Ridge

R2 score	0.6278697505216295
Mean Absolute error	33.72813850437482
Mean Absolute error with Cross Validation	32.08519825938677

Lasso

R2 score	0.6390379729600415
Mean Absolute error	33.13279636072692
Mean Absolute error with Cross Validation	31.633830376668566

ElasticNet

R2 score	0.6407796454066192
Mean Absolute error	33.21607789779686
Mean Absolute error with Cross Validation	31.861983408447536

RANSAC

R2 score	0.36647289008539985
Mean Absolute error	39.70640365232994
Mean Absolute error with Cross Validation	38.1282970062711

Huber

R2 score	0.5248172176877497
Mean Absolute error	34.401080605540194
Mean Absolute error with Cross Validation	37.478413605215444

SVM

SVR Linear (dados originais)

R2 score	0.7399927042616334
Mean Absolute error	28.119166645570772
Mean Absolute error with Cross Validation	31.778070453779076

NuSVR (dados originais)

R2 score	0.5974438833989077
Mean Absolute error	33.94200162359375

Mean Absolute error with Cross Validation	38.6031322520874

SVR Não-Linear (dados originais)

R2 score	0.45796224903527216
Mean Absolute error	40.18892143991224
Mean Absolute error with Cross Validation	42.26756609544757

SVR Linear (dados escalonados)

R2 score	0.7121478153106454
Mean Absolute error	28.374455975710646
Mean Absolute error with Cross Validation	31.83201905964802

NuSVR (dados escalonados)

R2 score	0.7384652039704711
Mean Absolute error	27.96503229892052
Mean Absolute error with Cross Validation	31.493820192925085

SVR Não-Linear (dados escalonados)

R2 score	0.36246061848354094
Mean Absolute error	44.44542815465159
Mean Absolute error with Cross Validation	47.0945425585164

SVR Linear (PCA)

R2 score	0.6367229668080696
Mean Absolute error	31.739251548606724
Mean Absolute error with Cross Validation	35.81537027532742

Rede-Neural

Keras

R2 score	0.6270595084511554
Mean Absolute error	31.085508935831857
Mean Absolute error with Cross Validation	29.769184223055838

Considerações finais

O maior desafio que enfrentamos durante todo o trabalho foi o grande número de colunas iniciais. Além disso, complementar a esse desafio foi o fato de que, individualmente, o peso de cada coluna que influenciava o resultado final era pequeno, mas o conjunto de todas as colunas era o que se apresentava essencial para o devido processamento dos dados. Isso restringiu nossas capacidades de remoção de colunas ou alteração da dimensionalidade de qualquer forma, como visto no PCA.

Comparando com algumas submissões na competição do Kaggle, percebemos que os melhores resultados foram obtidos utilizando informações da coluna SMILES, com decomposições da string em dados processáveis. Como mencionado, não encontramos uma forma prática de aproveitar essas informações dentro do escopo do trabalho, o que possivelmente limitou os modelos de atingirem resultados melhores.

Acreditamos que um próximo passo a ser tomado para melhorar a qualidade do atual melhor modelo de aprendizagem que temos até agora, ou seja, a rede neural, seria descobrir uma forma de incorporar a coluna SMILES diretamente na rede de maneira decomposta, com cada caractere da string influenciando-a diferentemente, através de label encoding.