Todo o código desenvolvido encontra-se no ficheiro “code.py”.

A primeira função desenvolvida foi a “csvToAvro” , o objetivo desta função é a leitura da base de dados original em formato csv e a criação de uma base de dados em .avro com a mesma estrutura. O ficheiro “schema.avsc” contém a estrutura necessária para esta nova base de dados. Sempre que for necessário consultar informação ao longo deste exercício, estas consultas serão feitas na nova base de dados em .avro.

De seguida procedeu-se à aplicação de modelos ML de regressão (função “fitModels”) para encontrar o fator com maior impacto no cálculo global de cada cidade. A ideia será treinar os modelos com as nossas amostras e no final olhar para os coeficientes internos do modelo para ver qual das features tem maior influência. Inicialmente faz-se uma consulta à BD com o objetivo de obter os 3 valores correspondentes a cada fator que são as nossas features, o rank de cada cidade que será a nossa variável alvo, e o nome de cada cidade caso pretendemos testar o modelo com uma cidade em específico.

De seguida fez-se a divisão do conjunto de dados no qual 80% das amostras seriam para treinar os nossos modelos e 20% para os testar. Como neste problema todas as nossas features já se encontravam na mesma escala (0-100), não houve necessidade de as padronizar ou normalizar. Por fim procedeu-se ao treino dos algoritmos no qual apenas se utilizou modelos supervisionados lineares: LinearRegression, ElasticNet, LassoLars, Lasso e Ridge.

Para avaliar a performance dos modelos calculou-se a média dos erros quadráticos e os coeficientes de determinação. Os resultados obtidos são bastantes parecidos entre os algoritmos como se pode ver na fig.1. De seguida consultou-se os parâmetros internos e verificou-se que o fator social é o de maior impacto. É necessário referir que neste problema quanto menor o coeficiente associado a uma feature for, mais importante esta é. Isto acontece porque quanto menor a variável alvo é, melhor é a classificação da cidade (rank 1 = melhor classificação).

A função “indicePais” serve para calcular o indice de sustentabilidade, o fator com maior impacto no cálculo global do país e criar uma base de dados no qual é possível consultar estas métricas. A solução adotada para atribuir um rank a cada país foi: calcular as médias do fator social, fator económico e fator ambiental para cada país usando as suas cidades, e usar os modelos ML treinados para atribuir um “rank”. Este rank atribuído pelos algoritmos não é definitivo porque contém valores decimais, dois países diferentes podem ter o rank 9.40 e o rank 9.60. De forma a obter os ranks finais ordenou-se os países pelo valor atribuído pelos algoritmos e a sua posição nesta ordem são o seu rank (valor mais baixo = rank 1, 2º valor mais baixo = rank 2, etc.). Como as classificações atribuídas pelos algoritmos podem ser diferentes entre si, decidiu-se que na base de dados é possível consultar o rank de cada país de acordo com atribuição de cada algoritmo (5 colunas cada uma com a classificação correspondente a cada algoritmo). Para saber qual o fator mais importante em relação a cada país, multiplicou-se cada um dos seus fatores pelos seus coeficientes correspondente (medidos no modelo ML) e fez-se uma comparação entre os 3 valores. O menor valor corresponde ao fator mais importante no país em questão. Esta métrica também está presente na nova base de dados.

Para calcular as métricas necessária e criar a base de dados com o índice dos continentes utilizou-se a mesma lógica da função indicePais, só que em vez de se calcular para cada país calculou-se para cada continente. Isto é feito na função indiceContinente.

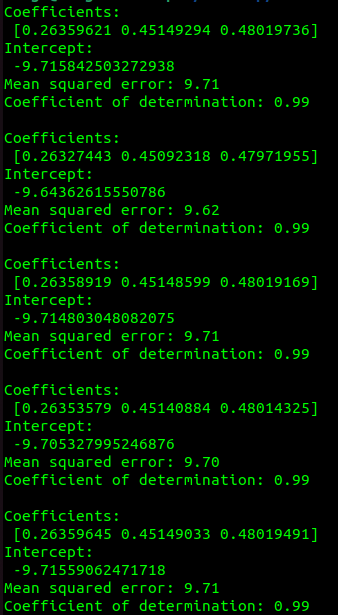


Figura 1 Resultados obtidos por cada algoritmo. Correspondendo de cima para baixo a LinearRegression, ElasticNet, LassoLars, Lasso e Ridge