**Trabalho de Linguagens de Programação**

**Bike Sharing Demand**

**Victor S. Trambaioli, Jefferson G. Lima, Joelson M. Pereira**

Escola Politécnica – Universidade Federal do Rio de Janeiro (UFRJ)

Trambaioli2@poli.ufrj.br, jeffersongli@poli.ufrj.br,

***Resumo.*** *Neste trabalho, tivemos como objetivo desenvolver uma solução para um problema de Data Science da plataforma Kaggle. Dentre vários problemas propostos, escolhemos o “Bike Sharing Demand”, onde é proposto que, a partir de uma base de dados, a quantidade de bicicletas alugadas a cada hora de cada dia seja prevista.*

1. **O Problema:**

Os sistemas de alugueis de bicicletas se tornaram algo muito comum nos dias atuais. Os usuários, de maneira muito simples e prática, podem liberar o veículo em um determinado ponto de alugueis, na maioria dos casos com uso de seu próprio SmartPhone, e pedalar até algum outro local de armazenamento das bikes da mesma empresa. Por ser um sistema automatizado e integrado, é possível que dados, como hora, data, tempo de uso, perfil do cliente etc., sejam coletados e armazenados. É a partir desses dados que esse problema foi proposto, onde deveremos prever a quantidade de aluguéis futuros partindo dos dados do passado.

1. **Solução:**

- Dados:

Para o desenvolvimento da solução para o problema nos foi fornecido dois arquivos .csv, um de treino (train.csv) e um de teste (test.csv), onde o arquivo de treino possui dados completos de alugueis de bicicletas, que foram registrados durante um certo período de tempo, e o arquivo de teste é o arquivo em que não consta o número total de “bikes” alugadas em um dia, sendo nosso dever fazer essa previsão.

São os dados contidos em cada arquivo:

- datetime: Data e hora do registro.

- season: Estação do ano, tendo como legenda : 1- primavera 2- verão 3- outono 4- inverno.

- holiday: Se o dia é feriado, com 1 para sim e 0 para não.

- workingday: Se o dia não é nem fim de semana e nem feriado, com 1 para sim e 0 para não.

- weather: Diz como está o clima do tempo no dia, com legenda: 1- claro, poucas nuvens, parcialmente nublado. 2- Névoa. 3- Neve Leve, chuva leve. 4- Chuva pesada, neve pesada.

- temp: Temperatura em Celsius.

-atemp: Sensação térmica em Celsius.

- humidity: humidade relativa do ar.

- windspeed: velocidade dos ventos.

Esses itens são comuns entre os dois arquivos de dados, tendo o arquivo de treino três colunas a mais, que são elas:

- casual: Número de alugueis feitos por usuários sem cadastro.

- registered: Número de alugueis feitos por usuários com cadastro.

- count: Número de alugueis totais. Este item é o que será necessário prever para o arquivo de teste.

- Desenvolvimento:

A primeira etapa realizada para o desenvolvimento da solução para o problema foi a de análise do cenário e dos dados. Primeiro verificamos se havia dados nulos em alguma das colunas dos dois arquivos, obtendo uma resposta satisfatória de que todos os dados estão completos. Depois usamos a função “describe()” para analisar os números dos dados da tabela de treino, onde verificamos, por exemplo, que há dias registrados com humidade relativa 0 e também com humidade 100, fato que tivemos que pesquisar pois nem sabíamos que tais acontecimentos eram possíveis. Outra observação interessante é que foram coletados dados tanto em dias de inverno com bastante frio – temperatura mínima de 0.82 graus celsius – e dias de calor extremo – temperatura máxima de 41 graus celsius – o que nos mostra o quão completa é a base de dados fornecida.

Agora usamos a função “pivot\_table()” para analisar a relação do feriado e do dia de trabalho com as estações, tendo como resultado a média de alugueis em cada caso. Conseguimos observar que quando é feriado o número de alugueis tende a diminuir na primavera e no verão, e tendem a aumentar no outono e no inverno, apesar dessas diferenças não serem muito significativas. Em relação ao dia de trabalho, observamos que o número de alugueis tende a diminuir apenas no verão, sendo nos demais casos maior quando é dia útil.

Por fim, nessa primeira etapa, utilizamos a função “corr()” para observar a correlação entre os itens, principalmente entre o número de alugueis e os demais. As colunas “temp” e “atemp” são as que possuem maior correlação com o número de bicicletas alugadas por hora, com valor 0,46 cada uma. As colunas “windspeed” e “season” também possuem boa correlação com o número de alugueis, estando ambas na casa do 0,1. Uma observação que nos surpreendeu foi a de que o clima e a humidade possuem uma correlação negativa com a coluna “count”, o que para nós foi bastante contra-intuitivo já que é trivial pensar que em dias de muita chuva ou muita neve, por exemplo, o número de alugueis de bicicletas deveria reduzir bastante.

Tendo feito as primeiras análises os dados da tabela, começamos a realizar uma limpeza nos dados, a fim de conseguir um aumento entre as correlações desejadas. A primeira coluna a ser manipulada foi a de humidade, já que sua correlação é de -0.31 com o número alugueis. Através da função “loc()” dividimos os dados dessa coluna em 8 partes, fazendo isso para os dois arquivos de dados, onde não substituímos o valor e sim criamos uma nova coluna chamada “humidity2”, para caso o resultado encontrado não seja satisfatório. Posteriormente usamos o “pivot\_table()” para analisar a média de alugueis de bicicletas por cada uma das 8 partes de humidade. Assim, observamos que algumas partes possuem médias bastante parecidas e decidimos agrupá-las, reduzindo assim a humidade para quatro valores absolutos. Observamos novamente as correlações e verificamos que a coluna nova criada tem uma correlação -0.25 com a coluna “count”, ou seja, obtivemos um aumento de 0,6 nesse dado.

Posteriormente realizamos exatamente o mesmo procedimento com as colunas de temperatura, sensação térmica e velocidade dos ventos. A temperatura teve uma redução de menos de 0,01 em sua correlação, a sensação térmica teve sua correlação aumentada em menos de 0,01 e a velocidade dos ventos teve sua correlação aumentada em pouco mais de 0,01.

Outra observação que foi realizada logo quando batemos o olho nos dados é de que há uma coluna para indicar se é feriado, uma para indicar se é dia útil, mas não há uma coluna para indicar se é fim de semana. Então resolvemos criar a coluna “weekend”, onde terá resultado 1 quando as outras duas colunas citadas acima forem 0. Para isso utilizamos a função “loc()”, onde preenchemos os valores 1, e a função “fillna()” para preencher com o valor 0 os dados que ficaram como nulos. Observamos que essa nova coluna não possui uma correlação boa com a quantidade total de alugueis, entretanto possui boa correlação com a quantidade de usuários casuais, dando a entender que aos fins de semana aumenta o número de alugueis por pessoas sem registro.

Por fim, depois de já ter realizado as limpezas devidas nos dados e criado uma coluna para informar quando o dia é no fim-de-semana, resolvemos criar, a partir da coluna “datetime”, colunas de hora, dia e mês. Após realizado esse procedimento, observamos que todas elas possuem correlações positivas com a quantidade de alugueis, sendo os dados de “hora” o que possui a maior, resultado que foi bastante satisfatório já que provavelmente ajudará na previsão desejada.

Agora chegamos na última etapa do processo, que é a de realizar a previsão da coluna “count” para o arquivo “test”. Foram retiradas as colunas “datetime” – por ser valor em string-, “temp2” – por ter correlação menor que a coluna “temp” -, “windspeed” – por ter correlação menor que a coluna “windspeed2”- e “humidity” – por ter correlação menor que a coluna “humidity2”.

Após realizar alguns testes iniciais, percebemos que a função de decisão, dentre a que conhecemos, que retorna um valor mais satisfatório é a “DecisionTreeClassifler()”. Assim, utilizando a mesma, decidimos realizar a previsão da coluna “count” do arquivo de teste por dois caminhos, sendo o primeiro uma maneira direta, onde obtemos a coluna desejada a partir dos dados em comum entre os dois arquivos, e uma maneira indireta, onde realizamos a previsão das colunas “casual” e “registered” para o arquivo de teste e as somamos para obter a coluna “count”. Realizando essas duas etapas, e submetendo seus resultados na plataforma “Kaggle”, concluímos que o caminho direto obteve um maior resultado, com valor de 0,62938.

1. **Conclusão:**

Por fim, concluímos que o desafio foi satisfatório para a prática do aprendizado que obtivemos em aula, com a manipulação e observação dos dados visando sempre os simplificar o máximo possível.

Entretanto ficou a sensação de que o resultado obtido poderia ter sido melhor, talvez com alguma relação entre os dados que, com a falta de experiência, não conseguimos visualizar, ou também com algum método de resolução diferente que ainda não tivemos conhecimento.