UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Ofertas de celulares do Buscapé
Extração, análise e visualização

Alunos

Arthur Lubambo Peixoto Accioly (alpa)

Marlon Reghert Alves dos Santos (mras)

Victor Duarte Diniz Monteiro (vddm)

5 de Julho de 2017

Índice

1 - Proposta

O objetivo desse projeto é desenvolver um sistema capaz de inferir o preço de telefones móveis baseado nos seus requisitos. A ideia é extrair dados como preço e características dos produtos no buscapé e enriquecer as características com fontes diferentes. A partir dos dados será feito uma regressão linear para explanação da relação entre características e preço do produto. Com esse regressor seremos capaz de identificar as principais *features* que devem ser levadas em consideração na hora de inferir o valor do telefone e então criar um modelo com significância estatística de forma a estimar o custo do telefone para o cliente baseado nas especificações oferecidas na entrada.

O projeto é dividido entre os módulos de extração de dados (crawler), processamento em um servidor e a visualização das estimativas em uma página web.

2 - Módulos

Para a melhor organização do projeto, o dividimos em 3 módulos que serão explicados no capítulo seguinte, sendo eles:

- Crawler
    - Módulo responsável por extrair os dados da página do Buscapé (ver cap 3) e dar como saída um csv, sendo cada linha um aparelho celular e as colunas as configurações para uma dada feature (memória, processador, etc). Os dados são formatados e pré-processados para que facilite o processamento.
- Processamento
    - Dado o csv resultante da extração de dados, este módulo é responsável por processá-lo, em uma instância virtual da Google Cloud, e expor os resultados através de uma API acessível via HTTP.

      Para o processamento utilizamos o scikit-learn para realizar a regressão linear dos dados a partir de uma feature de entrada e usando o vetor de preço como a base do treinamento.
- Visualização
    - Este módulo é responsável por fornecer, através de uma página web, as entradas para que o usuário interaja, e a partir deste input, se comunicar com o servidor, descrito acima, e exibir de forma intuitiva os resultados das estimativas.

3 - Base de dados e extração

Como explicado anteriormente, a fonte de dados deste trabalho foi a listagem de celulares e smartphones que a empresa Buscapé disponibiliza em seu website. Para conseguir realizar a extração, foi necessário criar um crawler da página de forma a obter as características de interesse e então salvá-las em um arquivo formato *.csv*, para permitir que os outros módulos do projeto pudessem utilizar os dados sem grandes barreiras e não depender de conexões à Internet.

O primeiro desafio na criação do Crawler foi identificar quais seriam as features extraídas e onde podíamos encontrá-las. As características extraídas foram: *marca, linha, modelo, chips, câmera traseira, câmera, frontal, tamanho da tela, resolução, velocidade do processador, memória interna* e *memória RAM*. Como o Buscapé, naturalmente, oferece uma lista das especificações técnicas dos aparelhos móveis na página detalhada da aparelho, encontrar estas características não foi desafiante. No entanto, foi necessário que, para cada aparelho encontrado na listagem, o crawler também obtivesse a página detalhada do aparelho e extraísse as informações desta página.

Entretanto, apesar do Buscapé oferecer informações técnicas, estas poderiam ser exibidas em dois formatos tabulares diferentes, utilizando a tag de HTML *div* ou a tag *table*. Para não haver perda de dados, criamos funções para possibilitar a extração das duas formas.

Outro desafio encontrado na etapa de extração de valores da página web foi o fato da página exibida no browser ser diferente da extraída por uma requisição HTTP através da máquina. Isto porque o site utilizava de *cookies* para alterar a lista de aparelhos. Só foi percebido este comportamento validar a exibição dos dados da página em modo anônimo/privado, oferecido pelo browser. A maior consequência disto foi que alguns aparelhos obtidos pelo crawler não estavam disponíveis na loja e, logo, apesar de especificação, não haviam preços. Para estes, não deixamos de extrair as especificações, mas o preço foi reportado como não existente.

O último desafio da fase de extração era a normalização de valores. Isto porque era natural que na página web valores, como exemplo para *câmera frontal*, fosse exibido como "2.0 megapixels" ou "2 megapixels". O tratamento foi, para alguns casos, extrair o valor através do uso de expressões regulares e transformá-los para tipos nativos de Python, como **float**, **int** ou **str**. Algumas das características que sofreram tal transformação foram as variáveis *memória interna*, *memória RAM* e *velocidade de processador*.

As ferramentas usadas para realizar tal tarefa foram *Python* com o suporte da biblioteca de parser de HTML *BeautifulSoup*. Uma análise mais aprofundada sobre os dados pode ser encontrada no *python notebook* **analysis**, no diretório *analysis/*.

4 - Processamento

O módulo de processamento se encontra na pasta **server** do repositório, escrito todo em python.
O nome é server pois nós levantamos um servidor web, dockerizado, que exibe uma API acessível via HTTP para o qual podemos requisitar análise de dados dada uma feature específica.

Dentro da pasta **core** está todo o código responsável por pré-processar os dados, instanciar o servidor e executar o processamento.
Neste diretório temos 3 arquivos principais:

- data_set.py
    - Aqui nós lemos o CSV de entrada e abstraímos a representação da matriz, pré processando os dados. Fizemos a abstração da seguinte forma:
        - Cada categoria (processador, memória, etc) é mapeado em um index inteiro.
        - Cada subgrupo de uma categoria é mapeado em um index, ex: Processador possui versões 1.0, 2.0, 3.0. A detecção dos subgrupos é realizada agrupando os valores presentes na coluna.
        - Para cada celular e cada categoria, sabemos a que subcategoria ele pertence.
        - Mantemos os dados na memória para otimizar o processamento evitando IO em cada request. Por isso os dados são pré processados durante a inicialização do servidor
        - A classe DataSet fornecesse acesso aos dados abstraídos em memória.
- analyser.py
    - Aqui encontraremos o script responsável por realizar a regressão linear, utilizando a biblioteca scikit-learn. Dado uma categoria, nós montamos uma matriz NxM, onde N é o número de celulares e M o número de sub-grupos

dessa categoria. Essa é uma matriz binária, e a posição (i,j) terá 1 caso o i-ésimo celular pertence ao subgrupo j dessa feature, e 0 caso contrário.

- Dada essa matriz binária, nós removemos as colunas nulas, e em seguida aplicamos o método *fit* do scikit-learn, veja o código abaixo:

```
@staticmethod
def analyse(filters_dict, category, dataset):
    matrix, train_vector = Analyser.gen_matrix(filters_dict, category, dataset)

    coefs_indexes, null_columns = Analyser.del_null_columns(matrix)

    regr = linear_model.LinearRegression(fit_intercept=True)
    regr.fit(matrix, train_vector)

    return  (regr.coef_, regr.intercept_, coefs_indexes, null_columns)
```

  - retornamos os coeficientes, o valor de interceptação, um mapeamento dos índices do vetor de coficientes para os subgrupos e quais colunas estavam nulas e foram removidas.

- server.py
  - Utilizando o framework Flask, levantamos um servidor expondo duas rotas:
    - GET /categories
      - Retorna, para cada categoria, o seu nome (ex: Memória), seu id e um mapa de seus subgrupos. Esse mapa contém o nome do subgrupo e um id, (ex: 16GB -> 0 , 8GB -> 1)
      - Exemplo:

```
[
    {
        "name": "marca",
        "id": 1,
        "groups": {
                "samsung": 0,
                "apple": 1,
                "motorola": 2
            }
    },
    {
        "name": "processador",
        "id": 0,
        "groups": {
                "1.0": 0,
                "2.0G+": 2,
                "3.0": 1
            }

    }
]
```


- POST /analyse
  - O body dessa rota deve conter um JSON do seguinte formato:

    {

    “category”: 0,

    “filters”: {

    1: 2

    }

    }

    Nesse caso acima, a feature a ser base da regressão será a indexada pelo número 0 (baseando-se no retorno de /categories), e “filters” contém uma whitelist (opcional) de filtros de categorias. Acima o filters apenas levará em conta no processamento, celulares cuja categoria indexada pelo índice 1 seja do subgrupo 2.

    Como resultado teremos o vetor com os coeficientes e um mapa dos indexes desse vetor, assim sabemos que o index 0 do vetor, por exemplo, representa a categoria 1.

A web gui (próximo capítulo) faz uso dessa interface para comunicar a escolha do usuário, e através da resposta, exibir a estimativa.

Este servidor está dockerizado, dessa forma, para rodar localmente, basta ter o docker instalado e buildar a imagem que está na raiz do diretório **server**.

Com docker ganhamos a vantagem de testar o servidor independentemente de máquina, e com isso, realizar o deploy em máquinas virtuais em provedores de cloud. Por isso, o servidor está rodando na Google Cloud, e é acessível pelo seguinte endereço: **23.251.151.44:5000**

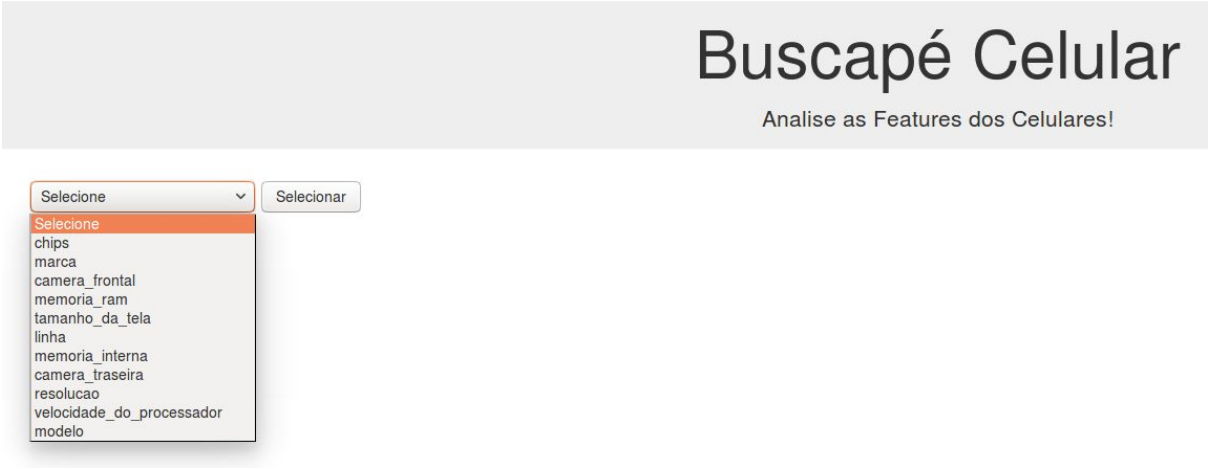
5 - Visualizações

Para realizar a interação com o usuário e a visualização dos resultados, nós elaboramos uma página web, estática, que se comunica com o servidor para executar a estimativa do preço e através dos resultados, exibir gráficos para o usuário.

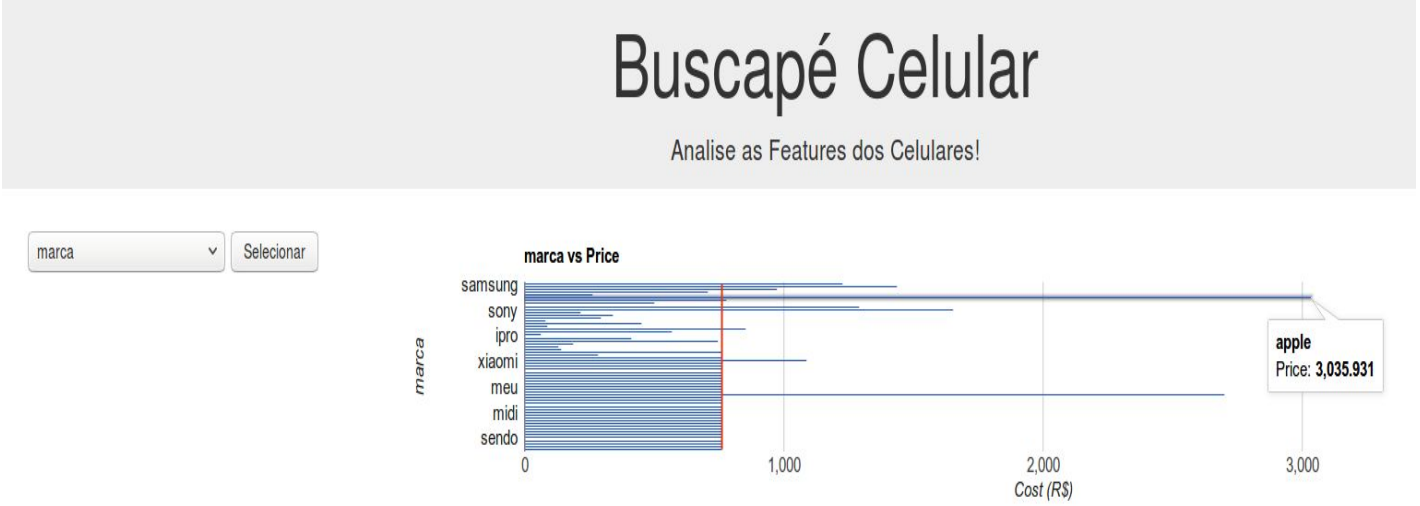
Toda a configuração da página pode ser encontrada no diretório **webgui**.

Abrindo o arquivo **buscape.html** você terá acesso a interface,.

A seguir uma sequência de fotos apresentando-a.



Com as categorias carregadas do servidor, o usuário pode escolher uma feature a ser analisada



Após a escolha da feature, um gráfico é exibido. No eixo x temos os preços, crescendo para a direita e no eixo y temos todos os subgrupos possíveis para a feature escolhida. Passando o mouse em cima da barra, nós conseguimos ver o valor exato

para aquele subgrupo. Perceba que neste caso, a Apple se apresenta como a merca com maior preço (R$3.035,931)

6 - Conclusão

Neste projeto aprendemos as dificuldades da extração, processamento e análise de dados. Extrair informações de uma página web traz um conjunto de problemas, em especial podemos falar da dinamîcidade do markup de uma página pertencente a uma entidade que não quer ter seus dados extraídos, porém, encontrando os padrões certos e utilizando técnicas como regex, conseguimos realizar a extração.

Do lado do processamento, conseguimos ver a gama de bibliotecas que a comunidade python oferece para que consigamos executar as tasks em tempo hábil.

E aprendemos que a visualização dos dados é uma forma de agregar valor aos resultados obtidos.

Nessa tarefa aprendemos como nos dividir, arquitetar e implementar, em grupo, um projeto prático de ciência dos dados.

Poderíamos ter melhorado a visualização adicionando mais gráficos e insights, além de cruzar dados com outros sites, estes insights ficaram para trabalhos futuros.