11/11/2022

Relatório Final

Bootcamp - Precificação dinâmica

Ana Chaves / Paulo Angellotti/ Victor Mitsuo

RELATÓRIO FINAL

DICIONÁRIO DE DADOS

Name	Texto	O título da listagem. Obs: Observe que limpamos os dados para remover textos que parecem preços (por exemplo, US$ 20) para evitar vazamentos. Esses preços removidos são representados como [rm]
Item_condition_id	1 - 2 – 3 – 4 – 5	A condição dos itens fornecidos pelo vendedor. Categoria da condição do item, entre 1 que significa ruim até 5 ótimo.
Category_name	Texto	Categoria da listagem
Brand_name	Texto	Marca
Price	Números	O preço pelo qual o item foi vendido. Esta é a variável de destino que você irá prever.
Shipping	0 ou 1	Frete: 1 se a taxa de envio for paga pelo vendedor e 0 pelo comprador.
item_description	Texto	A descrição completa do item.
Date	Date	Criado para este desafio.
Stock	Números	Criado para este desafio, é o que tem em estoque disponível.
gen_cat		Categoria principal do item.
sub1_cat		Subcategoria nível 1
sub2_cat		Subcategoria nível 2

HIPOTESES

HIPÓTESE 1: VAMOS AVALIAR O IMPACTO DA *"BRAND"* PARA O NOSSO MODELO.

Medimos o impacto que a Brand teria no modelo, fizemos teste com brand e sem brand, utilizando o modelo de *ligthgbm*.

----------*Modelo com brand*----------
MAE: 13.470141
RMSE: 1266.732524

----------*Modelo sem brand*----------
MAE: 10.244517
RMSE: 624.229604

Podemos observar que com as *brands* o modelo perde em ambas as métricas, isso pode ser devido ao grande desbalanceamento entre as marcas.

Contudo, essas informações de *brand* não serão descartadas

HIPÓTESE 2: USAR PCA, IRÁ MELHORAR O PROCESSAMENTO?

Testamos a utilização dos dados com PCA, para melhorar a questão de memória. Foi utilizado uma regressão linear pela simplicidade.

Nesta configuração o modelo é capaz de convergir, tendo métricas aceitáveis considerando a distribuição dos valores que estamos tentando prever. Porém, ao comparar a aplicação do PCA com modelos desenvolvidos sem a sua utilização, notamos uma grande deterioração dos resultados, o que significa que não poderemos utilizá-lo nas etapas subsequentes deste trabalho.

HIPÓTESE 3 – ESCOLHA DO MODELO

Colocamos todos os modelos, que até então estavam avulsos em dois notebooks, em apenas um, avaliamos as métrica e juntamente com o tempo que leva para cada modelo executar.

Pelo modelo *MLP*, obtivemos os seguintes resultados:

```
MLP
MAE: 10.546949903299131
RMSE: 28.168716160371442
RMSLE: 0.2264574632599648
```

Pelo modelo *XGBoost*, obtivemos os resultados abaixo:

```
XGBoost
MAE: 11.842656
RMSE: 31.02
RMSLE: 0.279296
```

Pelo modelo *LigthGBM*, obtivemos os seguintes resultados:

```
LigthGBM:
MAE: $ 10.85
RMSE:  $ 21.20
RMSLE: 0.274586
```

Pelas métricas, temos uma leve vantagem da MLP sobre os outros modelos. Devido ao escopo apresentado no início do projeto, devemos prosseguir com modelos que apresentem métricas e que além disso se beneficiem do aumento da CPU/GPU. Sendo assim, optamos por prosseguir com a MLP.

HIPÓTESE 4 - O ESTOQUE PODE SER UM FATOR DE ALTERAÇÃO DO PREÇO DE VENDA, DEVIDO À OFERTA E DEMANDA?

Após a limpeza dos dados, avaliou-se se o estoque era relevante ao modelo ou não. Como o estoque não apresenta um inventário e não houve alteração durante o ano.

Temos muitos valores de 'stock' abaixo de 20 e poucos acima deste valor. Nesse sentido, podemos pensar que um produto com estoque elevado está encontrando dificuldades para ser vendido, enquanto um produto com baixo valor de estoque está em seu preço ideal ou muito barato.

Devido ao perfil apresentado, a variável 'stock' não mostra ter um grande impacto na variação do preço. Além do estoque ser praticamente constante durante todo o ano, no momento em que essa variável cai de forma brusca, o preço médio dos produtos permanece inalterado. Sendo assim, podemos dizer que é uma variável que não adicionada nada a um modelo preditivo e sua permanência para os modelos será discutida.

Para o modelo julgamos ser uma variável irrelevante e será excluída.

MODELO

Após toda a análise nos relatórios anteriores, compilamos os pontos principais neste documento.

Para chegar a um modelo que precifique com coerência qualquer produto que seja inserido, com uma margem de erro baixa e que tenha uma resposta praticamente instantânea, utilizamos uma rede neural, pois é um modelo robusto para o processamento de uma grande quantidade de dados e nos permite diversas alternativas de pré-processamento e arquitetura da rede para atingir os requisitos desejados.

Nosso público alvo neste modelo são valores entre $3 e $250, pois a análise exploratória indica que 99,51% dos pertencem a este grupo de consumo, isto automaticamente impõe uma limitação ao nosso modelo já não teremos boas predições para produtos com valor acima de $250.

Devido ao grande volume de dados, tivemos diversos problemas relacionados à capacidade de memória RAM nos sistemas utilizados, devido a essa condição, as redes exploradas foram razoavelmente rasas e com limitações em relação a quantidade de neurônios e técnicas de pré-processamento utilizadas.

Separamos os dados em treino, teste e validação. Os conjuntos de teste e validação foram completamente isolados do dataset de treino para que não ocorram vazamentos de dados mesmo considerando que, em teoria, os procedimentos adotados durante o pré-processamento trabalhassem linha a linha e não no conjunto como um todo.

A principal limpeza nos dados ocorreu nas colunas de texto onde aplicamos diversas expressões regulares para eliminar símbolos, siglas e padrões indesejáveis, também corrigimos expressões que poderiam gerar ambiguidade ao realizarmos a tokenização e stemming nas palavras. As colunas de texto foram concatenadas para uma única coluna que carregaria as informações de descrição, marca e nome. Utilizamos a biblioteca nltk para definir as *stopwords* e também para realizar o *stemming*.

Por fim, vetorizamos os textos com TF-IDF para estruturar os dados de texto, a técnica TF-IDF (*Term Frequency - Inverse Document Frequency*) mostra o quão importante é uma palavra dentro de um texto. Devido ao volume de dados, a saída desta função gera uma matriz esparsa.

As colunas categóricas foram tratadas com One Hot Encoder, ou seja, foram codificadas de forma distribuída onde cada possibilidade se torna uma nova coluna e o valor 1 mostra a presença desta característica e o valor 0 a ausência. Devido a complexidade presente nas colunas categóricas do conjunto de dados os resultados são matrizes esparsas.

A arquitetura da rede conta com uma sequência de 5 camadas densas, com camadas de *dropout* entre a segunda e terceira camada e entre a terceira e quarta camada. A última camada é onde a predição é realizada. Essa arquitetura mais simples foi escolhida devido aos problemas que enfrentamos devido aos limites impostos pelo hardware utilizado no projeto. A camada de *dropout* desabilita uma determinada fração dos neurônios de forma aleatória durante treinamento, dessa forma, os neurônios vizinhos aos neurônios desativados precisam se adaptar e realizar as representações dos neurônios desativados. Esse tipo de abordagem

evita uma especialização muito intensificada ao conjunto de treino e permite que o modelo seja capaz de generalizar com mais facilidade.

Utilizamos na maior parte da rede a função de ativação PReLU. Inicialmente, devido ao problema apresentado, a função de ativação ReLU parecia mais adequada devido a sua característica de ignorar valores negativos. Para este problema, através de um teste A/B, comprovamos que, para camadas ocultas, a função PReLU apresenta um desempenho superior em relação a função ReLU.

Este modelo foi treinado com epoch = 30 e batch_size = 2048. Tivemos um grande avanço no resultado, o MAE anterior era de $ 9.49 e agora é de $8.94.

LINKS

- Kaban: Trello
- GitHub: Repositório
- Canvas: Apresentação