# Pricing Challenge

Felipe Formenti Ferreira

# *GitHub*

**Link**

https://github.com/fformenti/pricing_challenge

**Scripts**

Todos os códigos necessários para esta análise constam aqui. Desde a o arquivo para dar carga na base de dados até os códigos para geração de gráficos

**Analysis**

Apresentação final e uma pasta chamada “viz” com todos os gráficos (inclusive alguns que não entraram na apresentação final).

**Data**

Na pasta output estão os arquivos com os resultados dos modelos

**** Todos os gráficos estão em inglês, peço desculpas por isso.***

## Objetivo

Criar um model preditivo para a venda de cada produto dado um preço.

Como sabemos que o preço (apesar de ser importante) não é a única variável que influencia as vendas, irei explorar outras possíveis variáveis para poder incluir no modelo preditivo.

## Abordagem

Irei dividir os dados em um grupo de treinamento e outro de teste. Treinarei os modelos com um grupo e usarei o outro para testar minhas previsão. A meta é poder vencer o modelo de regressão linear simples (proposta no desafio) usando o preço como única variável explicativa. Esse sera o modelo baseline, e se não conseguir uma melhora significativa então o modelo não serve.

# DADOS

# *Armazenamento*

POSTGRES: LOCAL

POSTGRES: AWS RDS

sales.csv

| PROD_ID | DATE_ORDER | QTY_ORDER | REVENUE |
|---|---|---|---|
| P1 | 2015-07-01 | 1.0 | 23.45 |
| P2 | 2015-05-23 | 3.0 | 65.39 |

comp_prices.csv

| PROD_ID | DATE_EXTRACTION | COMPETITOR | COMPETITOR_PRICE | PAY_TYPE |
|---|---|---|---|---|
| P1 | 2015-07-01 08:10:38 | C1 | 25.99 | 1 |
| P1 | 2015-05-23 08:10:38 | C2 | 27.99 | 1 |

Os dois arquivos .csv foram colocados em uma base de dados Postgres.
Dentro da pasta scripts é possível encontrar as duas rotinas que são usadas para dar o upload das tabelas.
O arquivo data_loader_local.sql cria e preenche uma base de dados local
O arquivo data_loader_rds.py cria e preenche uma base de dados na AWS RDS

# *Inconsistências*

The sales.csv file contains **transactional information** where each line represents a sale. The comp_prices.csv file contains **monitoring data of competitors' prices**. We have data available for 6 competitors, C1 to C6, which are monitored twice per day. The information below describes the data in each column:

Olhando para a tabela abaixo podemos ver que alguns produtos foram monitorados 8 vezes no mesmo dia para um mesmo tipo de pagamento, ao invés de duas como sugere os documento.

O problema foi contornado escolhendo o mínimo dos valores (dentro de um mesmo tipo de pagamento) já que o menor preço é mais provável de afetar as vendas.

5 rows

| | prod_id | date_order (yyyy-MM-dd) | competitor | pay_type | cnt |
|---|---|---|---|---|---|
| 1 | P7 | 2015-01-05 | C4 | 1 | 8 |
| 2 | P6 | 2015-02-17 | C4 | 2 | 8 |
| 3 | P6 | 2015-02-17 | C2 | 2 | 8 |
| 4 | P7 | 2015-01-05 | C4 | 2 | 8 |
| 5 | P6 | 2015-02-17 | C4 | 1 | 8 |

# *Tratamento de Dados*

Agregar a venda de cada produto por dia faz com que a previsão da quantidade de vendas fique mais fácil, além disso podemos criar um preço médio diário a partir dessa tabela. Essa operação foi feita dentro da base de dados por ser conveniente e para termos as duas tabelas a nossa disposição em futuras análises.

## sales

5 rows | Auto-commit

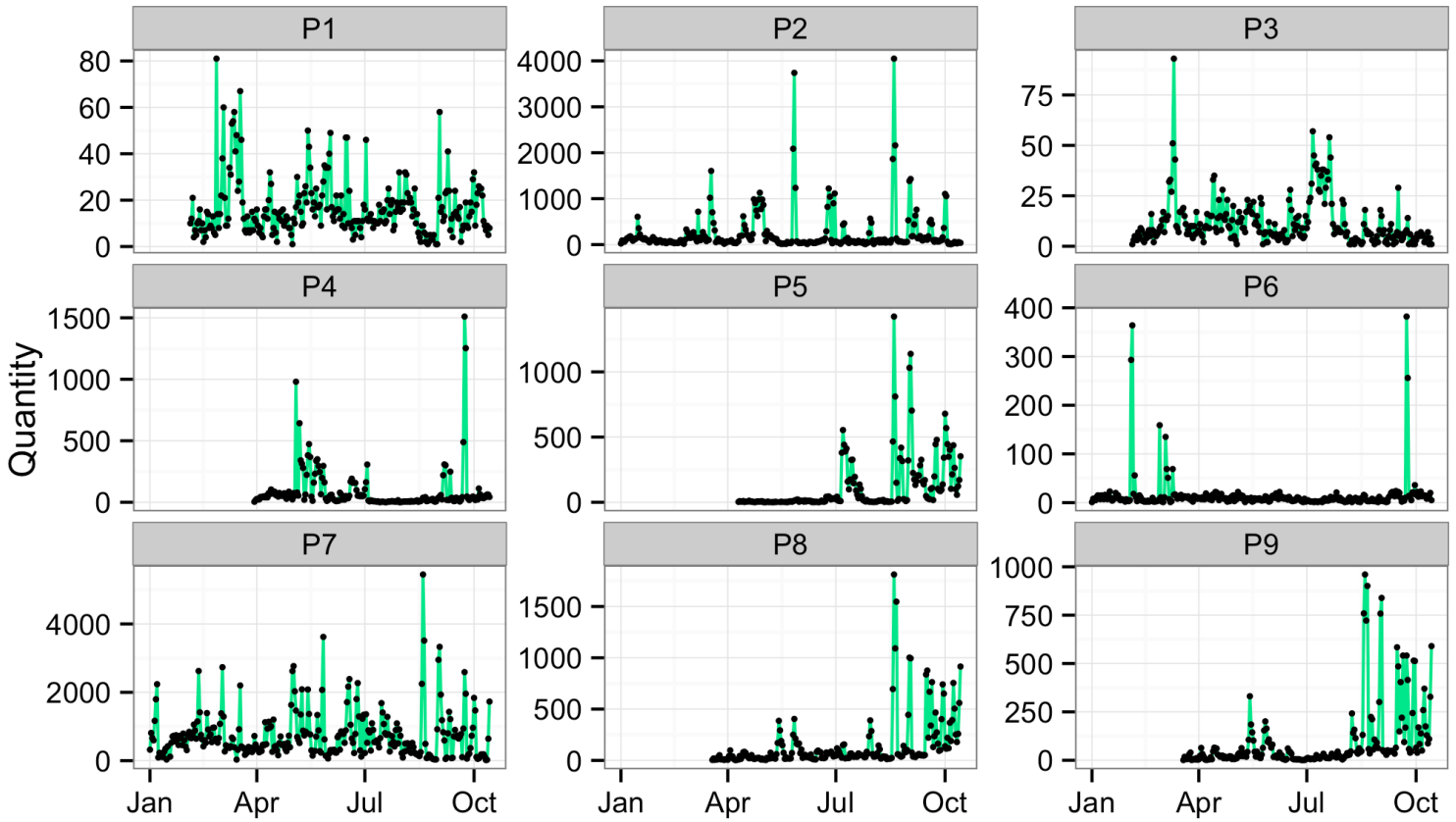
| | prod_id | date_order (yyyy-MM-dd) | qty_order | revenue |
|---|---|---|---|---|
| 1 | P1 | 2015-02-04 | 1 | 1499 |
| 2 | P1 | 2015-02-04 | 1 | 1499 |
| 3 | P1 | 2015-02-04 | 1 | 1499 |
| 4 | P1 | 2015-02-04 | 1 | 1499 |
| 5 | P1 | 2015-02-04 | 1 | 1499 |

## sales agregado

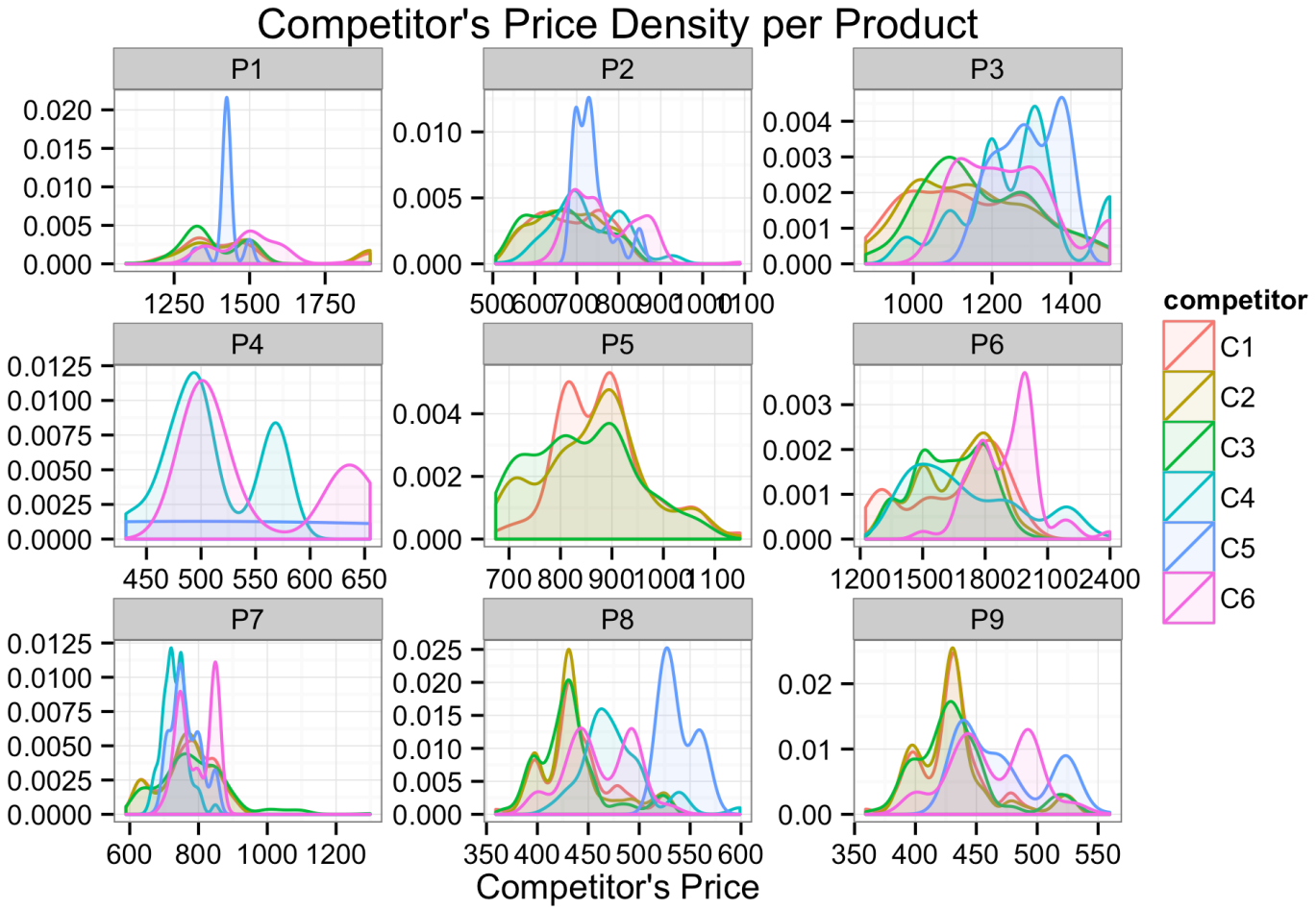
5 rows

| | prod_id | date_order (yyyy-MM-dd) | qty_order | revenue |
|---|---|---|---|---|
| 1 | P1 | 2015-02-04 | 10 | 14990 |
| 2 | P1 | 2015-02-05 | 12 | 17688.2 |
| 3 | P1 | 2015-02-06 | 21 | 31254.15 |
| 4 | P1 | 2015-02-07 | 4 | 5996 |
| 5 | P1 | 2015-02-08 | 7 | 10493 |

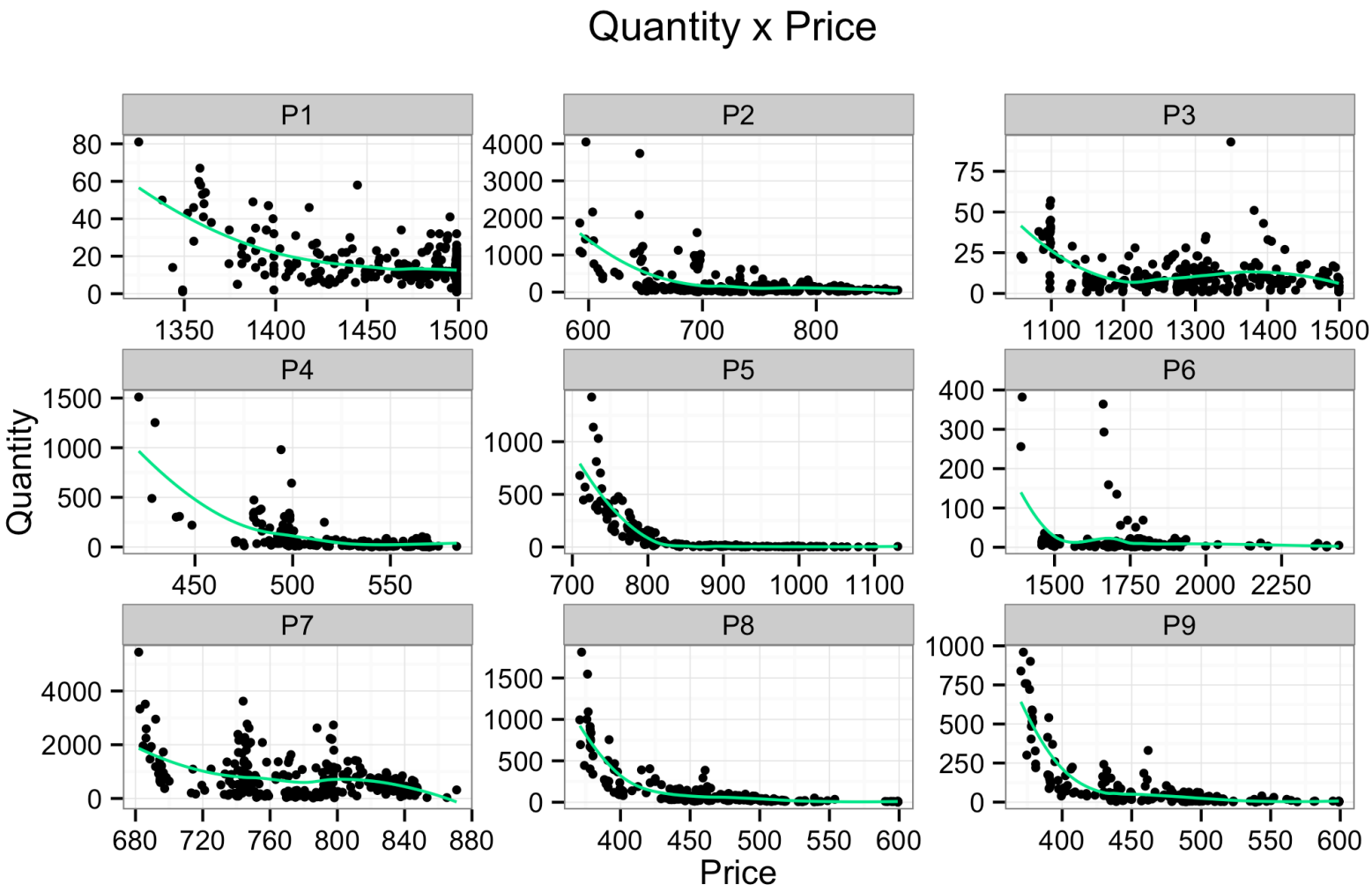
# Análise Exploratória



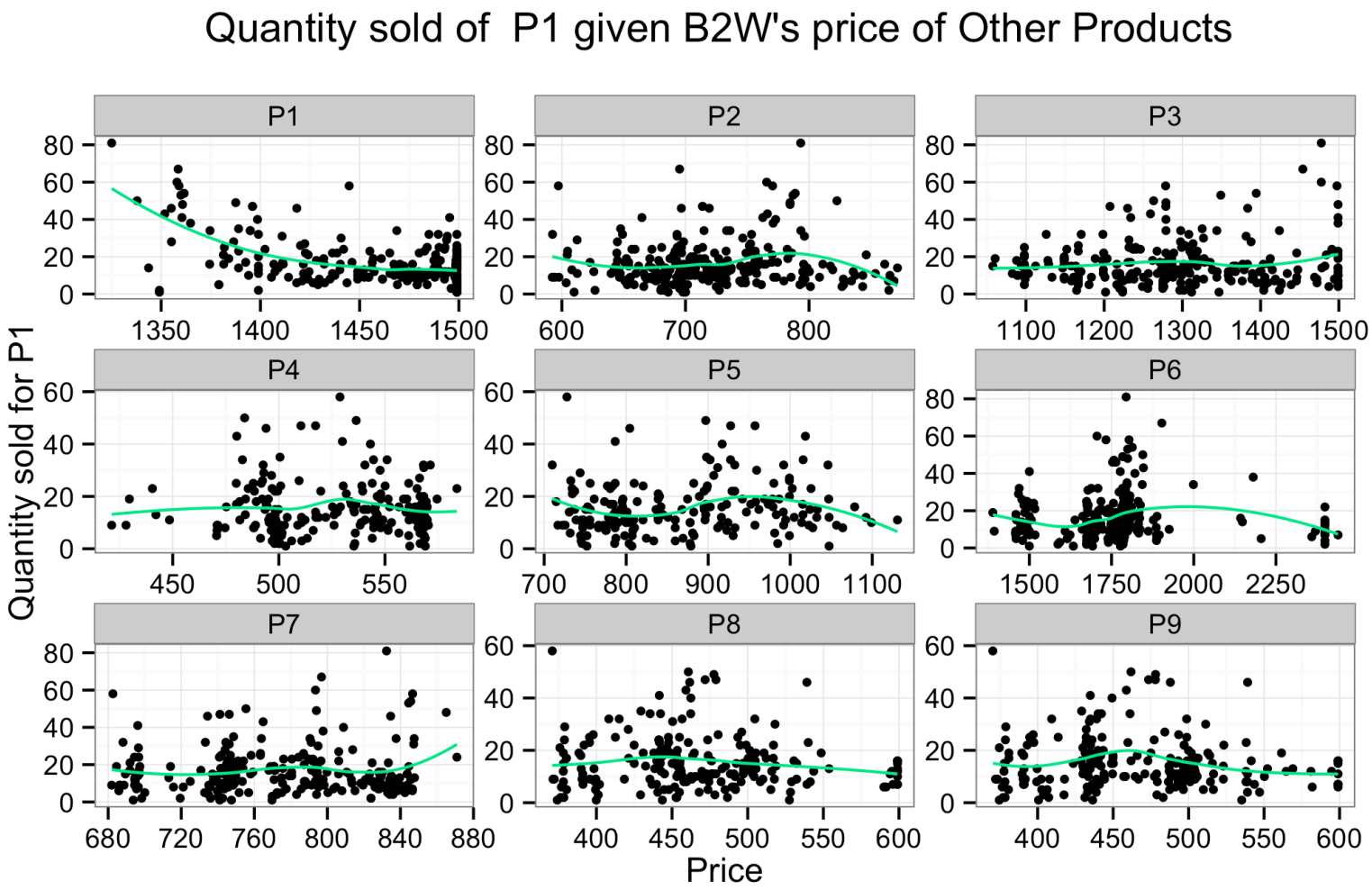

Olhando para a vendas dos produtos ao longo do ano de 2015 reparei que, apesar de errática, é bastante influenciada pelas vendas que a antecedem. Essa informação será bastante útil dentro de nosso modelo.
Vale ressaltar também que o eixo Y é diferente para cada gráfico. Imagino que prever P2 não será uma tarefa fácil.



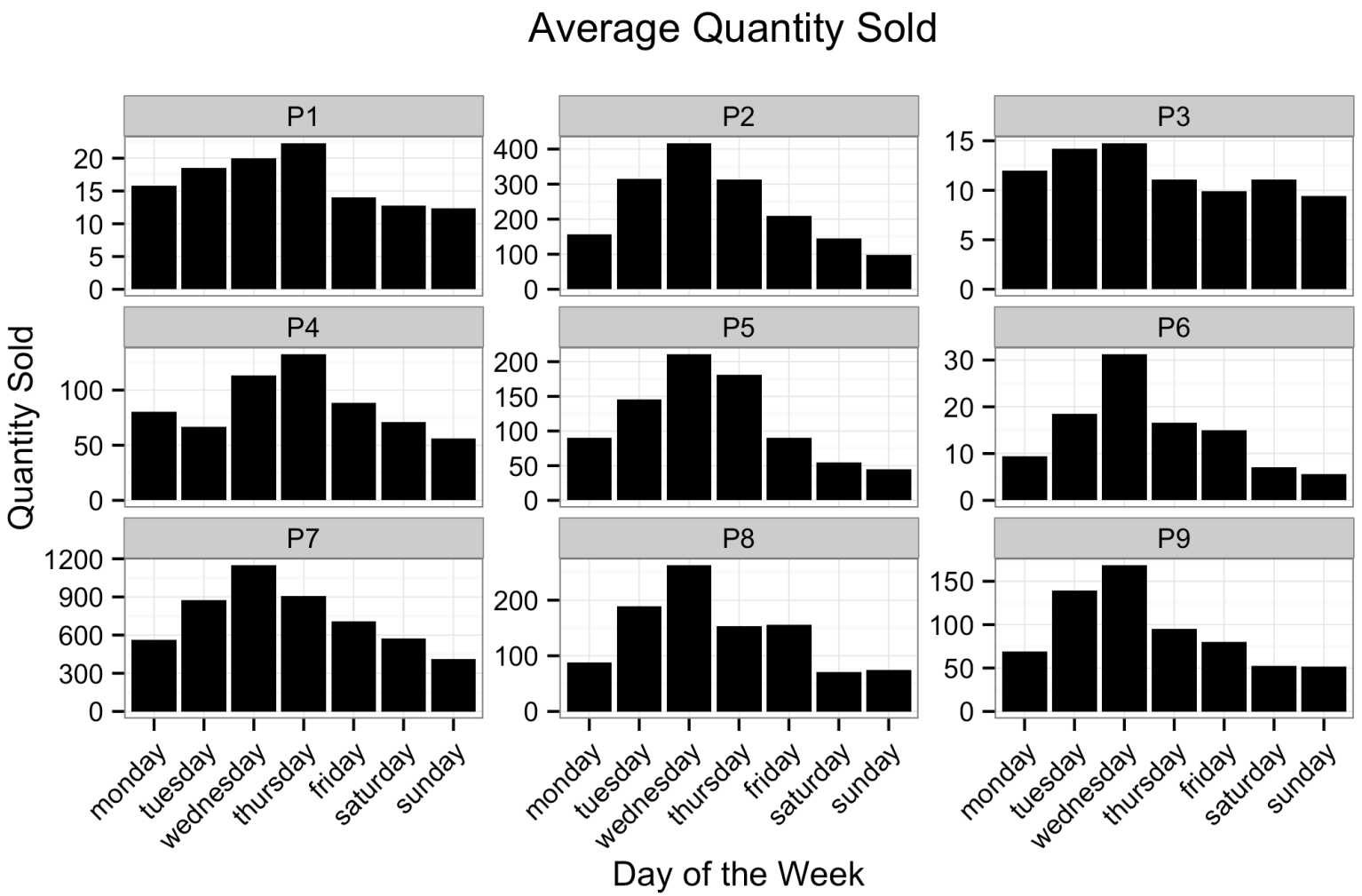

Através desses gráficos podemos ter uma noção da estratégia de preço dos concorrentes para cada produto. Apesar de não nos fornecer informações sobre como isso afeta a venda dos produtos da B2W é importante tê-los em um dashboard de monitoramento de preço dos concorrentes.




Como ja era de se esperar o preço afeta bastante a quantidade de vendas, especialmente se olharmos apenas para as faixas onde o preço é mais baixo. Percebe-se que existe um declino exponencial nessa faixa para a maioria dos produtos.




Imaginei que o preço de outros produtos do portfolio da B2W também pudessem ter efeito sobre as vendas. Neste caso escolhi o produto P1 para analisar. No entanto, pude constatar pelos gráficos acima, que o preço de outros produtos não afeta a venda de P1. O mesmo acontece com os outros produtos. Assim optei por não colocá-los dentro do modelo.




Como eu já esperava, o dia da semana é uma poderosa variável explicativa na hora de prever as vendas de um produto. Imagino que tenha um impacto forte em nosso modelo.

# Modelos

# *Variáveis Explicativas*

Depois de feita uma análise detalhada dos dados, separamos então as variáveis que nos mostraram ter relação com a quantidade de vendas de cada produto.

Preço B2W do produto
Preço da concorrência
Dia da Semana
Mês-Ano
Quantidade vendida no dia anterior
Quantidade total vendida nos três últimos dia *
Variação de vendas do penúltimo para o último dia *

* Não foi possível fazer a tempo

# *Dados Faltantes*

**Preço da concorrência**

Para preencher o preço do concorrentes eu usei a média dos preços dos outros concorrentes. Concorrentes que não tinham nenhum preço disponível para um determinado produto ficaram de fora do modelo.

Esse método não é o melhor, usar regressão Linear Múltipla seria uma solução mais elegante

# *Simplificações*

## Preço do Produto

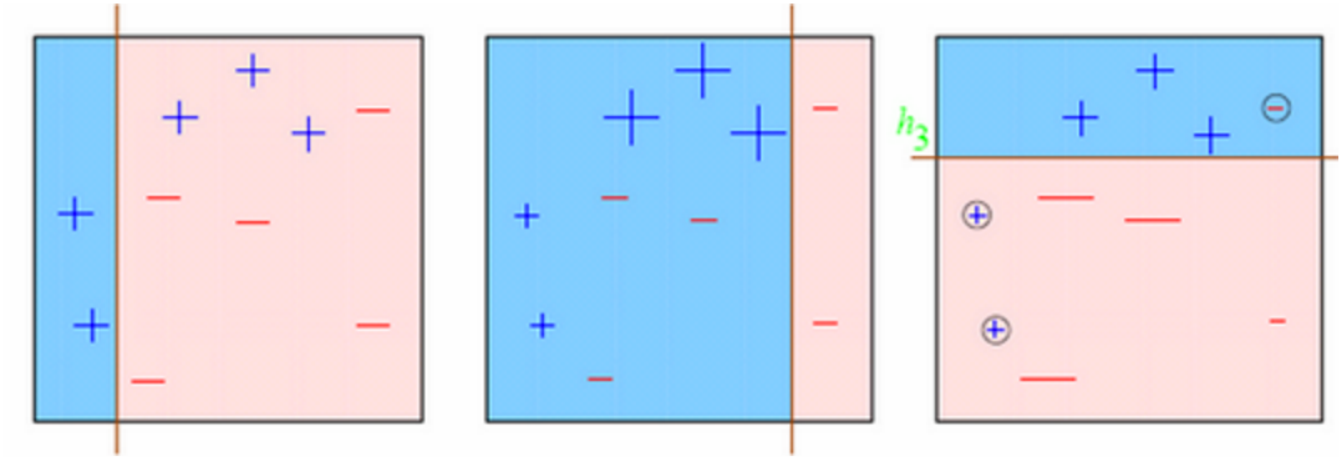
Considerei que todos os produtos que foram vendidos num mesmo dia, foram vendidos pelo mesmo preço.

## Preço da concorrência (Tipo de Pagamento)

Só utilizei o preço dos pagamentos imediatos (tipo 2).

# *Gradient Boosting*

## Como Funciona:




O modelo que escolhi foi o Gradient Boosting, dado seu alto poder de previsão. O desenho acima mostra o mecanismo por trás do algoritmo.

A ideia é combinar diversas árvores de decisão, assim como se faz no modelo Random Forest. Porém o gradient boosting coloca pesos diferentes a cada vez que um observação for classificada de forma errada.

No exemplo acima todos os pontos a direita do primeiro quadrado foram classificados como negativos. Como os três sinais positivos foram classificados de forma errada, no passo seguinte o peso dessas observações é alterado para que a próxima árvore de decisão tente classificá-los de forma correta.

# *Gradient Boosting*

**Escolha dos parâmetros:**

```
# Grid Search to find best parameters
params = {'n_estimators': [50, 100, 300, 500], 'max_depth': [2,3,4], 'max_features': ['sqrt']}
gb.grid_search(params)
```

```
[mean: -0.02141, std: 0.02599, params: {'max_features': 'sqrt', 'n_estimators': 50, 'max_depth': 2},
 mean: -0.02528, std: 0.03374, params: {'max_features': 'sqrt', 'n_estimators': 100, 'max_depth': 2},
 mean: -0.02324, std: 0.02946, params: {'max_features': 'sqrt', 'n_estimators': 300, 'max_depth': 2},
 mean: -0.02200, std: 0.02547, params: {'max_features': 'sqrt', 'n_estimators': 500, 'max_depth': 2},
 mean: -0.02141, std: 0.02599, params: {'max_features': 'sqrt', 'n_estimators': 50, 'max_depth': 3},
 mean: -0.02528, std: 0.03374, params: {'max_features': 'sqrt', 'n_estimators': 100, 'max_depth': 3},
 mean: -0.02324, std: 0.02946, params: {'max_features': 'sqrt', 'n_estimators': 300, 'max_depth': 3},
 mean: -0.02200, std: 0.02547, params: {'max_features': 'sqrt', 'n_estimators': 500, 'max_depth': 3},
 mean: -0.02141, std: 0.02599, params: {'max_features': 'sqrt', 'n_estimators': 50, 'max_depth': 4},
 mean: -0.02528, std: 0.03374, params: {'max_features': 'sqrt', 'n_estimators': 100, 'max_depth': 4},
 mean: -0.02324, std: 0.02946, params: {'max_features': 'sqrt', 'n_estimators': 300, 'max_depth': 4},
 mean: -0.02200, std: 0.02547, params: {'max_features': 'sqrt', 'n_estimators': 500, 'max_depth': 4}]
```

# *Gradient Boosting*

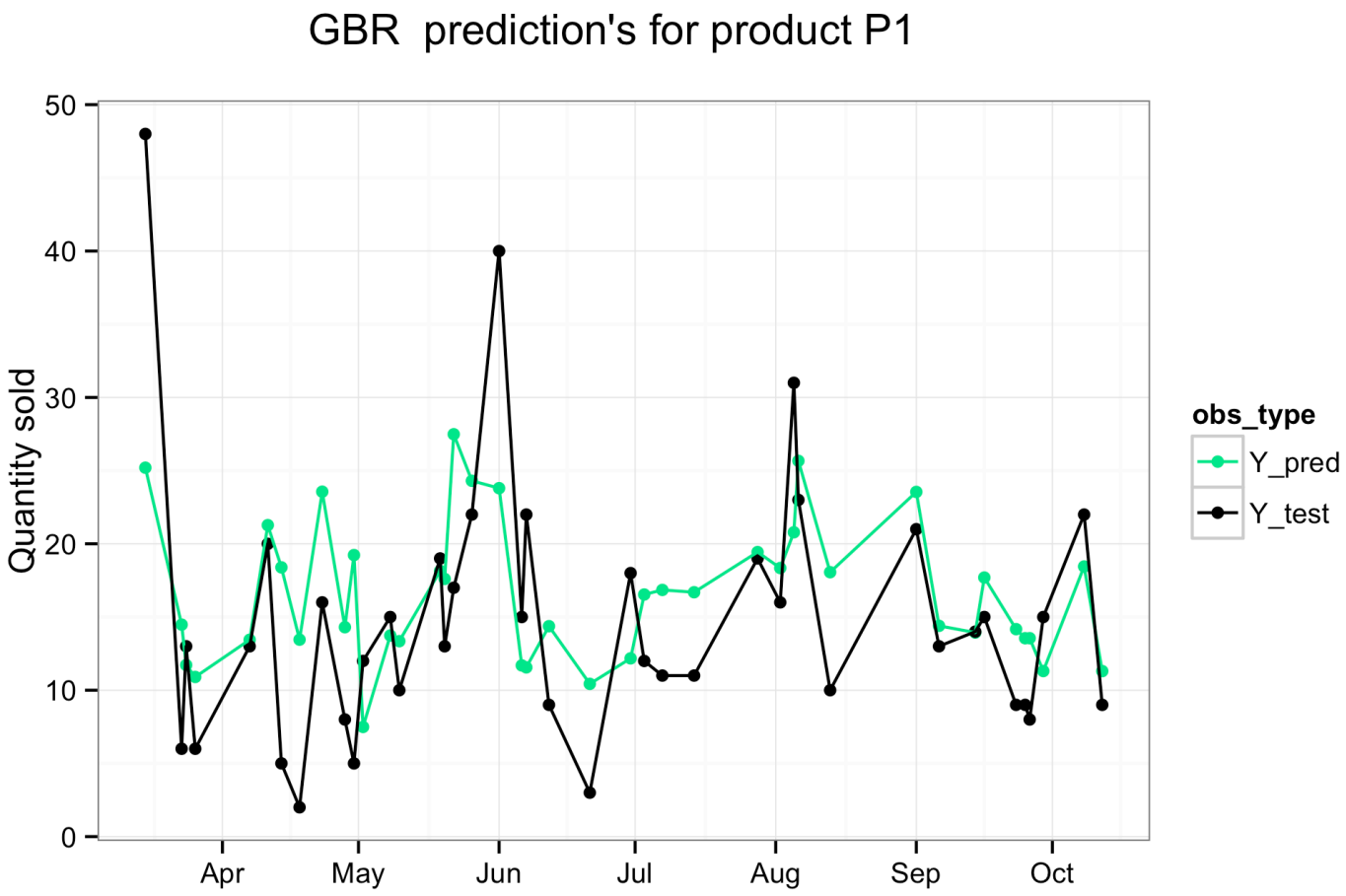
**Previsão (P1):**

```
MAE: 5.6427
MSE: 52.0945
```

| | prod_id | date_order | price | Y_test | Y_pred |
|---|---|---|---|---|---|
| **987** | P1 | 2015-03-16 | 1402.312500 | 24 | 26.999431 |
| **103** | P1 | 2015-03-22 | 1421.000000 | 7 | 16.499028 |
| **1896** | P1 | 2015-03-29 | 1499.000000 | 7 | 13.615307 |
| **927** | P1 | 2015-03-31 | 1483.776364 | 11 | 12.187607 |
| **1652** | P1 | 2015-04-04 | 1480.262500 | 8 | 7.095488 |
| **1031** | P1 | 2015-04-07 | 1487.461538 | 13 | 6.365761 |
| **910** | P1 | 2015-04-10 | 1457.333333 | 12 | 11.155337 |
| **291** | P1 | 2015-04-19 | 1389.007143 | 14 | 3.834154 |
| **246** | P1 | 2015-04-26 | 1399.000000 | 13 | 9.478059 |
| **716** | P1 | 2015-05-05 | 1440.670000 | 30 | 19.713369 |

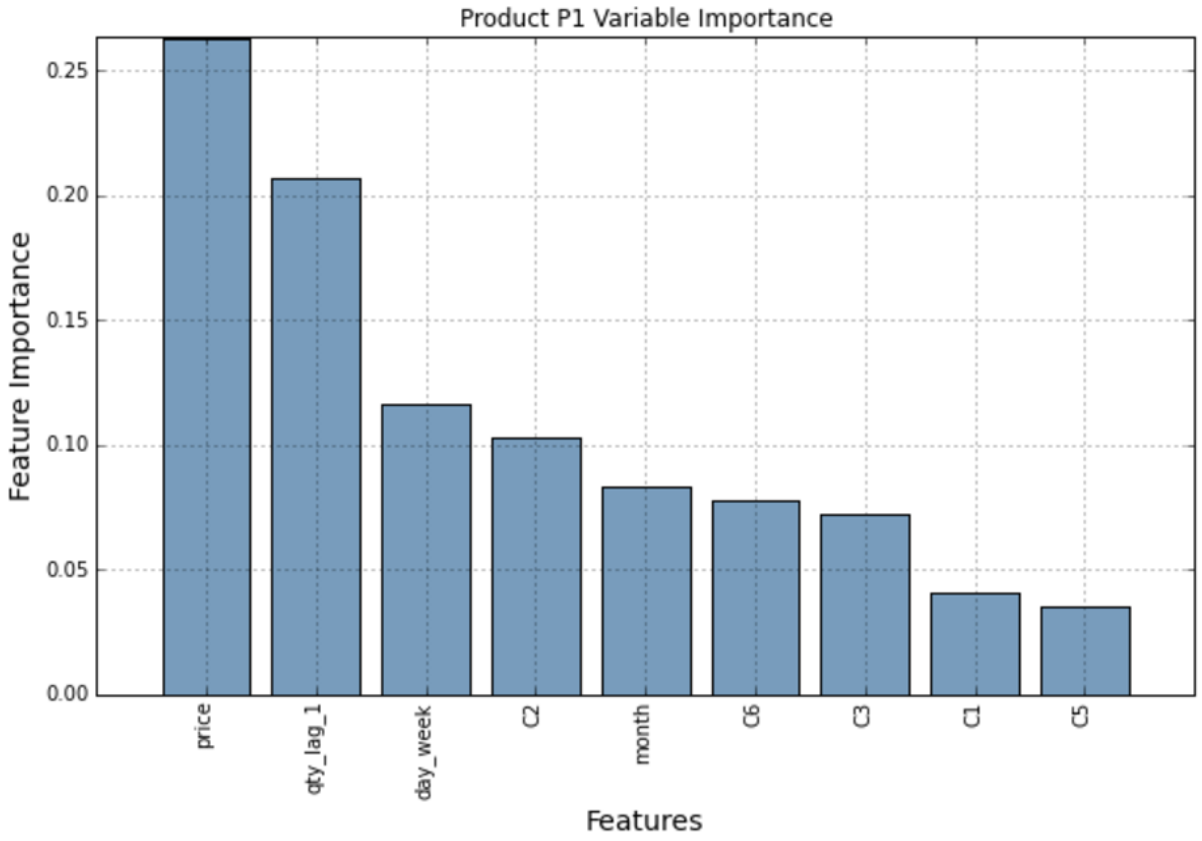
# *Gradient Boosting*

**Previsão (P1):**



# *Gradient Boosting*

**Variáveis que mais impactam as vendas (P1):**




Através deste gráfico podemos saber qual o maior concorrente da B2W para cada produto, assim como é possível ver quais variáveis influenciam mais a vendas de cada produto.

# *Linear Regression*

**Previsão (P1):**

```
MAE: 7.7900
MSE: 104.3331
```

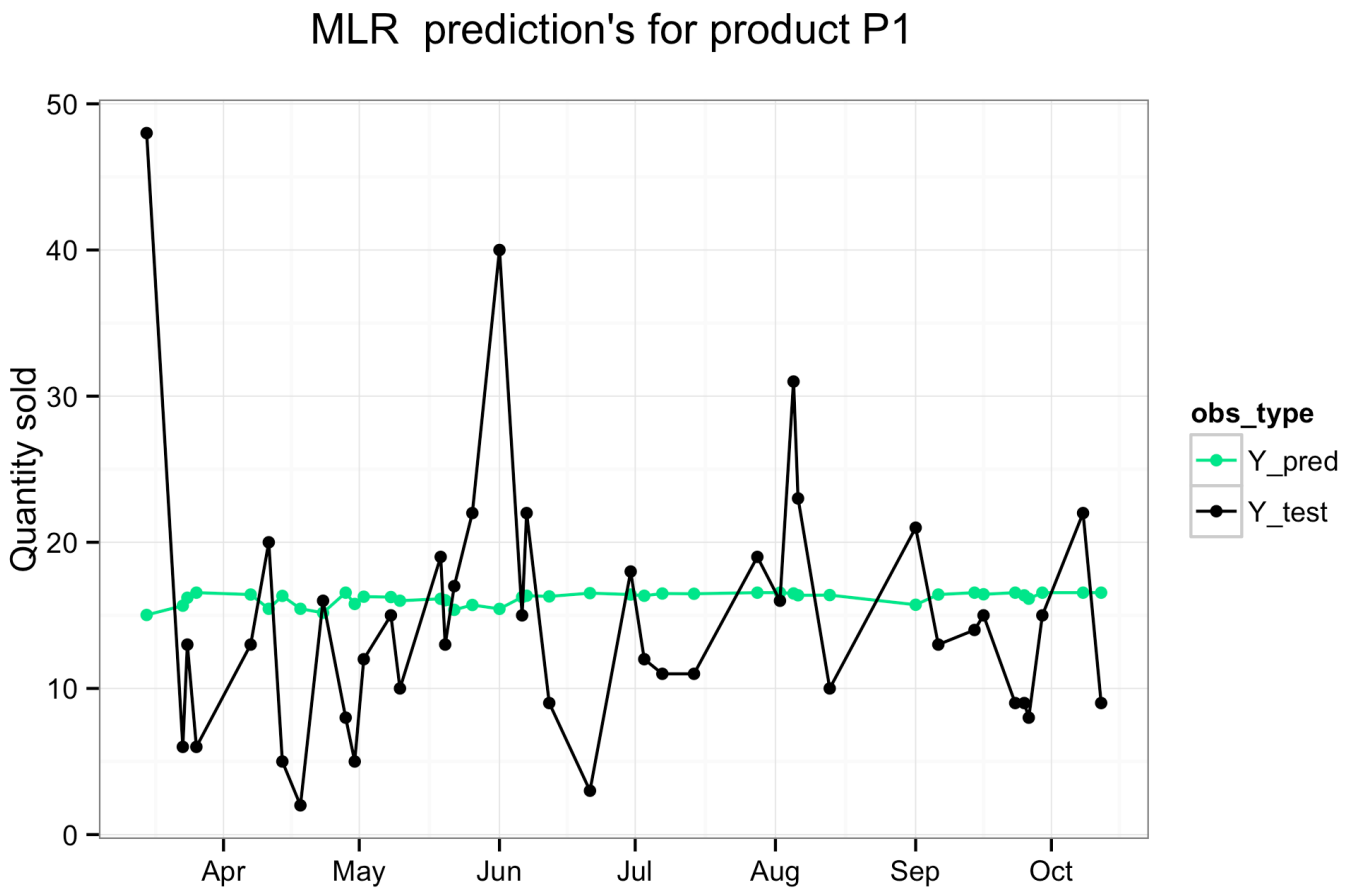
| | prod_id | date_order | price | Y_test | Y_pred |
|---|---|---|---|---|---|
| **987** | P1 | 2015-03-16 | 1402.312500 | 24 | 15.371705 |
| **103** | P1 | 2015-03-22 | 1421.000000 | 7 | 15.576552 |
| **1896** | P1 | 2015-03-29 | 1499.000000 | 7 | 16.431563 |
| **927** | P1 | 2015-03-31 | 1483.776364 | 11 | 16.264686 |
| **1652** | P1 | 2015-04-04 | 1480.262500 | 8 | 16.226168 |
| **1031** | P1 | 2015-04-07 | 1487.461538 | 13 | 16.305082 |
| **910** | P1 | 2015-04-10 | 1457.333333 | 12 | 15.974826 |
| **291** | P1 | 2015-04-19 | 1389.007143 | 14 | 15.225856 |
| **246** | P1 | 2015-04-26 | 1399.000000 | 13 | 15.335395 |
| **716** | P1 | 2015-05-05 | 1440.670000 | 30 | 15.792168 |

```
Price coef:  0.0109
```

Felizmente nosso modelo foi bem mais eficaz do que o modelo baseline. Podemos ver que as previsões giram em torno de 16 unidades uma vez que o preço se mantém estável. O próximo passo é usar todas as variáveis usados pelo Gradient Boosting e usar em outros modelos como Random Forest, Regressão Linear Múltipla entre outros, mas deixarei isso para uma próxima oportunidade

# *Linear Regression*

**Previsão (P1):**



# Obrigado

Felipe Formenti Ferreira