

Machine Learning não é Data Mining

146098	Flavio Matheus Muniz Ribeiro da Silva
119555	Humberto Politi de Oliveira
155976	José Henrique Ferreira Pinto
157324	Tamara Martinelli de Campos

Thales Mateus Rodrigues Oliveira	148051
Tiago Lobato Gimenes	118827
Victor Cerqueira Leal	046873
Victor Rodrigues Matsuguma	118893

O Problema

Predição da Demanda de
Medicamentos por Região

Pessoas com acesso à internet a utilizam para procurar possíveis doenças à partir de seus sintomas.

Isso gera uma quantidade enorme de dados que gera um modelo de buscas por sintomas na internet (Google).

Pretendemos resolver de um problema da administração da saúde pública, que é a predição da demanda de medicamentos futura por região do país.

— — —

A Proposta

A proposta do projeto é realizar uma predição de estoque de medicamentos por estados do Brasil de acordo com os sintomas das doenças cadastradas no banco de dados.

Para isso, utilizamos um lista de doenças que possui seus sintomas cadastrados no banco e através dos sintomas podemos determinar quais medicamentos podem ser utilizados em seu tratamento.

Para a predição do estoque utilizamos os dados do Google Trends, que indicam as quantidades de busca por um sintoma na internet.

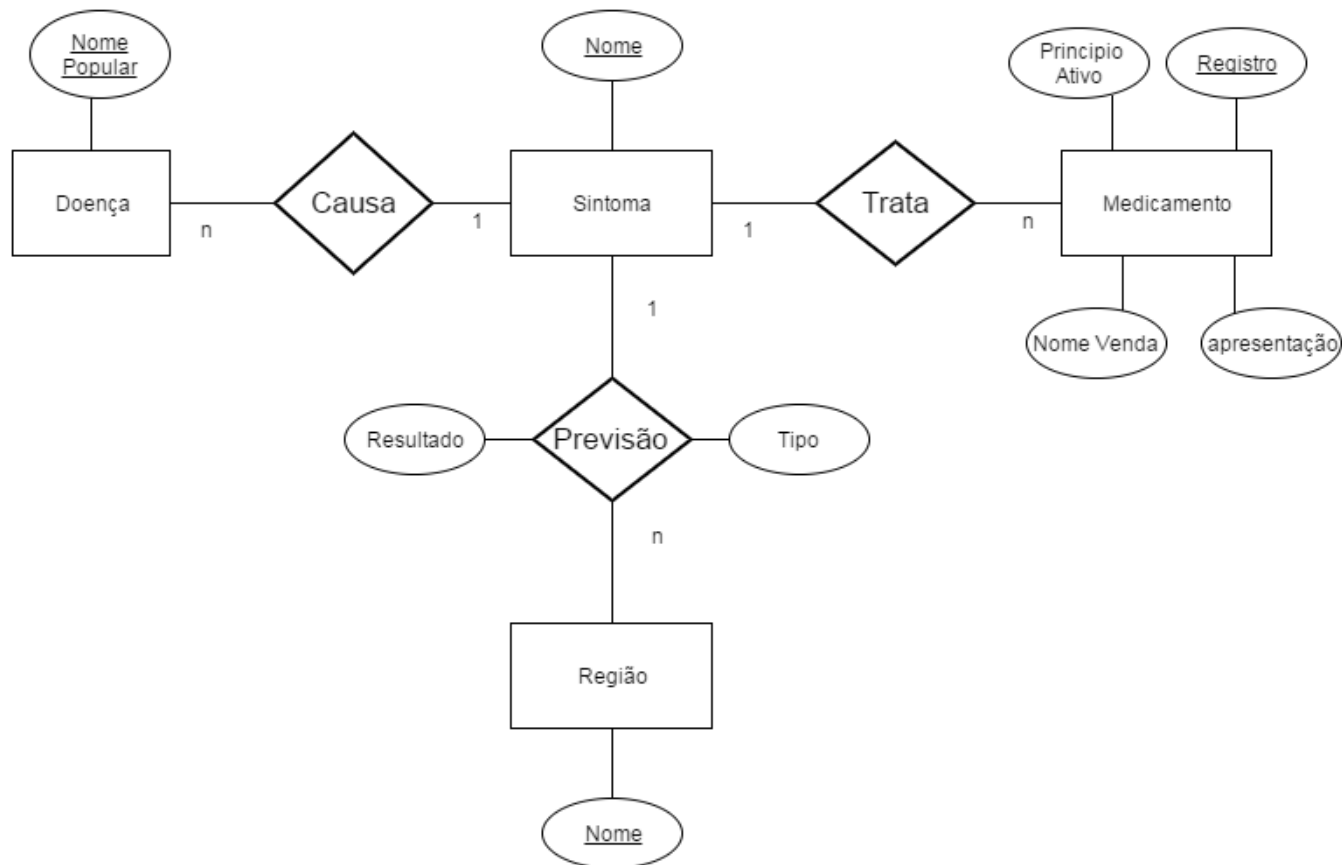
Dificuldades e Desafios

Inicialmente planejamos exibir as dosagens de medicamentos que seriam necessárias para o tratamento dos sintomas de uma doença.

Porém, apesar de todas as bulas de remédios cadastrados na ANVISA estarem disponíveis online, esses dados são fornecidos em formato PDF, e cada fabricante tem seu padrão de arquivo.

Dessa forma, não fomos capazes de baixar e manipular esses dados, o que acarretou numa mudança no nosso modelo conceitual, que atualmente não é capaz de prever as dosagens necessárias para tratar os sintomas de uma doença.

O Modelo Final do Banco de Dados



Extração de dados e Data cleaning

- ❖ Os dados de medicamentos foram extraídos do site da ANVISA em formato csv e continha mais de 25 mil tuplas. Para o uso no nosso projeto foi necessário que trocássemos a coluna de classe terapêutica para sintomas gerando uma relação 1:1 entre os sintomas e os medicamentos. Esse refinamento junto com a limpeza dos dados que não seriam usados levou um pouco mais de tempo e esforço do que imaginávamos. Ao final, ficamos com um arquivo com pouco mais de 5 mil dados.
- ❖ Seleção das doenças que estariam no banco e extração manual dos sintomas de cada doença.

Extração de dados: Google Trends

- ❖ **pytrends**: uma api não oficial do Google Trends
- ❖ Composição da requisição:
 - Termos de busca (até 5)
 - Período
 - Localização geográfica (ex: BR, BR-SP, BR-RJ)
- ❖ Nível de agregação diferente para períodos diferentes
 - Para resultados semanais, as requisições devem conter períodos de 36 meses ou menos
 - Período total: 2004 - 2016 → 5 requisições
- ❖ Um estado por requisição → 27 requisições
- ❖ 55 sintomas → 11 requisições
- ❖ Total: $5 \times 11 \times 27 = 1485$ requisições

Dificuldades

- ❖ Limite de requisições por tempo: existente, porém indeterminado. Quando ultrapassado, requisições eram negadas por 15 minutos ou até 4 horas
- ❖ Realizando uma requisição a cada 2 minutos, temos: 1485×2
= **49h30**
- ❖ Como consequência, diminuimos o número de sintomas

```
Exceeded Google's Rate Limit. Please use time.sleep() to space requests.  
Espera a partir de 2016-11-28 17:16:33.916047, por [ 2.13333333] minutos  
Exceeded Google's Rate Limit. Please use time.sleep() to space requests.  
Espera a partir de 2016-11-28 17:18:42.401726, por [ 4.26666667] minutos  
Exceeded Google's Rate Limit. Please use time.sleep() to space requests.  
Espera a partir de 2016-11-28 17:22:58.897528, por [ 8.53333333] minutos
```


Limitações do Google Trends

- ❖ Dados normalizados
 - Normalização pelo **período**
 - Normalização **entre os termos**
- ❖ Foco:
 - Na **evolução** do termo com o tempo
 - Na **relação** entre os termos
- ❖ A evolução natural do **acesso à internet** tem relação direta com a frequência das buscas
- ❖ Termos de busca são feitos por pessoas, não por bulas de remédio
 - Acentuação, espaços, gírias, etc
- ❖ Um termos não tem apenas uma conotação



Tópicos relacionados ?

Em ascensão ▼ ⋮

1	Frozen - Febre Congelante - Filme de 2015	Aumento repentino
2	Frozen - O Reino do Gelo - Filme de 2013	Aumento repentino
3	Zika Vírus - Tópico	Aumento repentino
4	Colossal - Site	Aumento repentino
5	Chicungunha - Doença	Aumento repentino

Consultas relacionadas ?

Em ascensão ▼ ⋮

1	febre congelante	Aumento repentino
2	frozen febre congelante	Aumento repentino
3	febre zika	Aumento repentino
4	zika	Aumento repentino
5	filme frozen	Aumento repentino

TEMOS OS DADOS



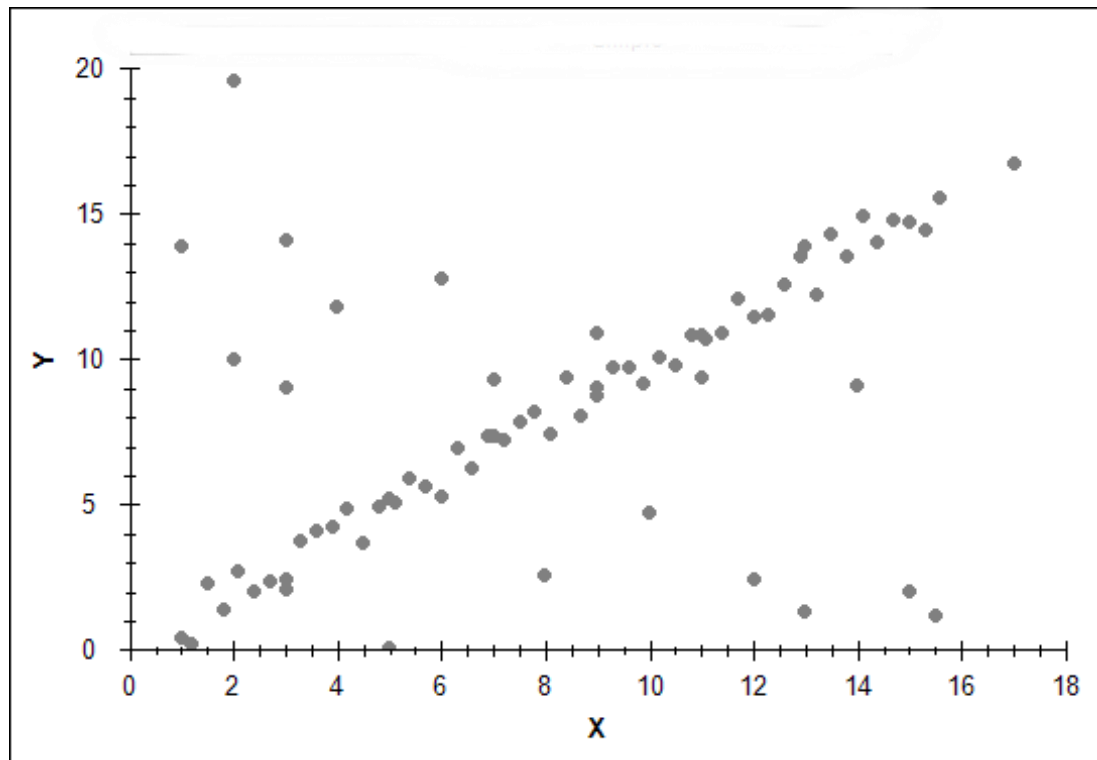
COMO FAZER PREDIÇÕES ?

Rede neural

**Regressão não paramétrica
baseada em kernels**

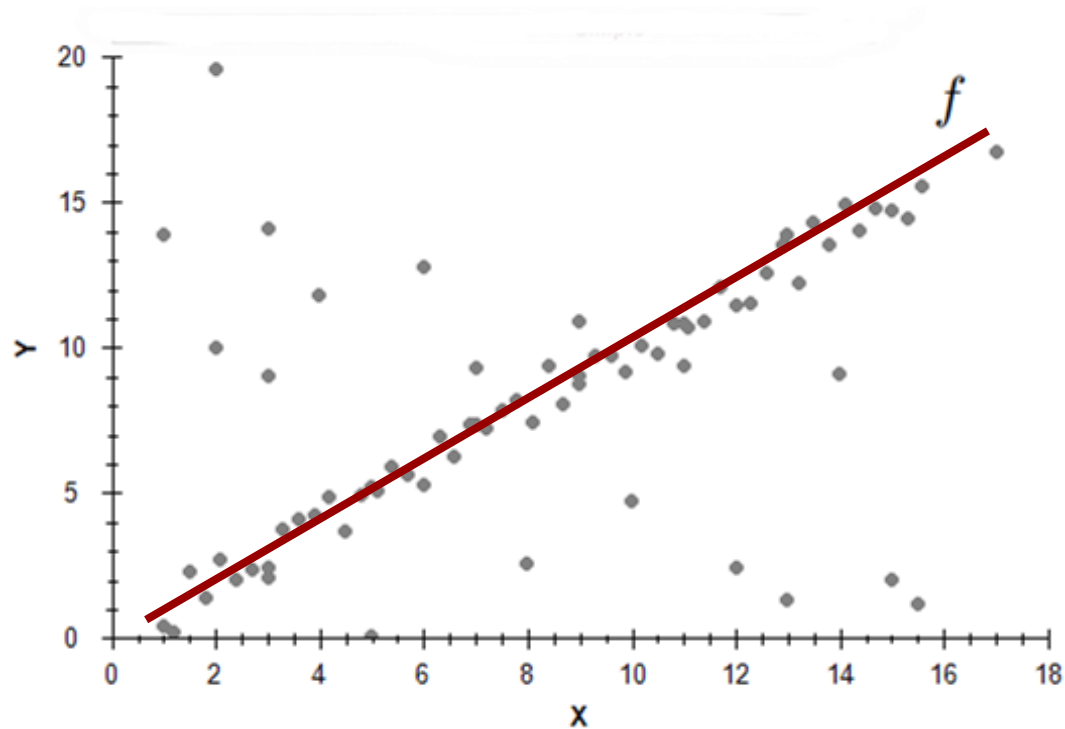
Regressão não Paramétrica baseada em Kernels

Problema



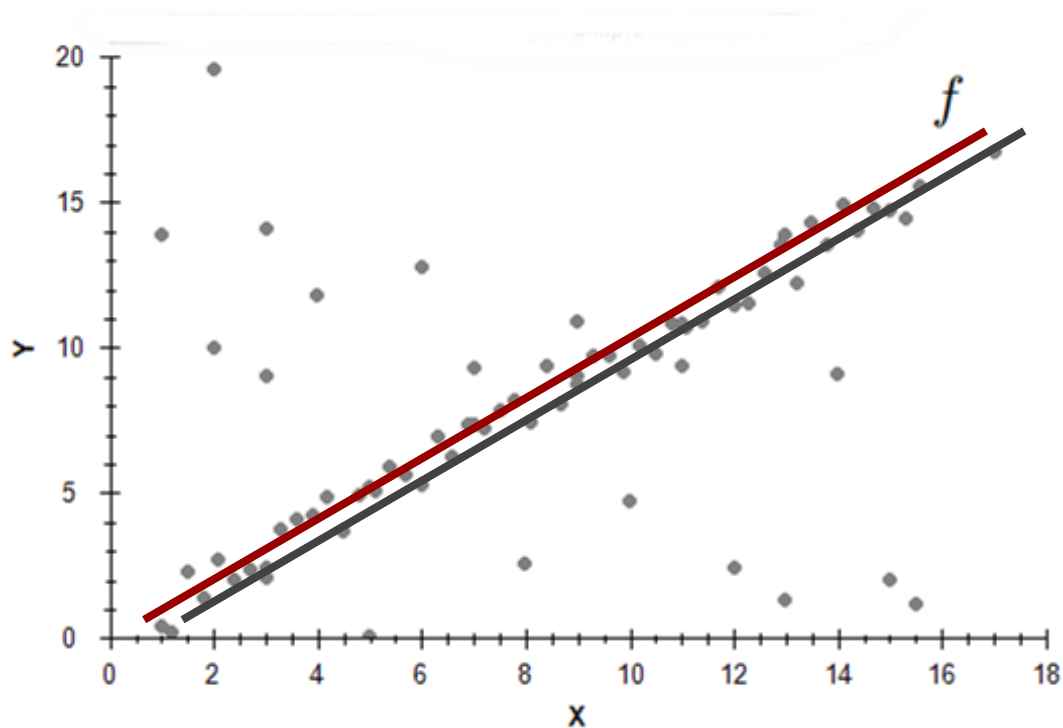
Regressão não Paramétrica baseada em Kernels

Problema



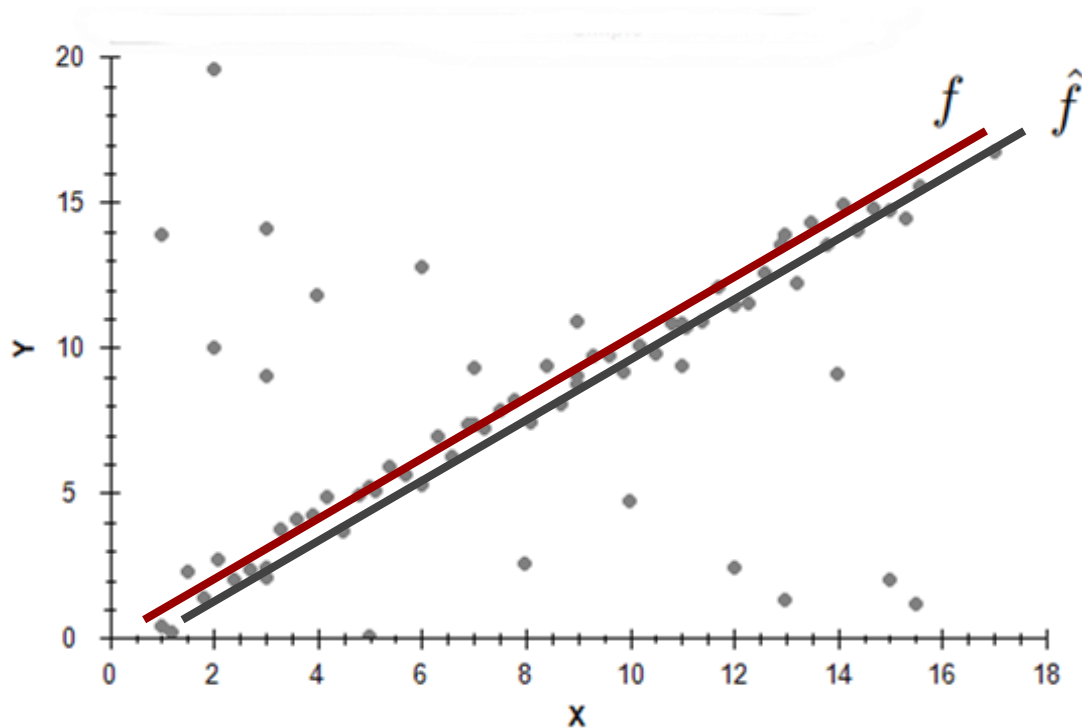
Regressão não Paramétrica baseada em Kernels

Problema



Regressão não Paramétrica baseada em Kernels

Problema



Regressão não Paramétrica baseada em Kernels


Solução

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K \left(\frac{x - X_i}{h} \right),$$

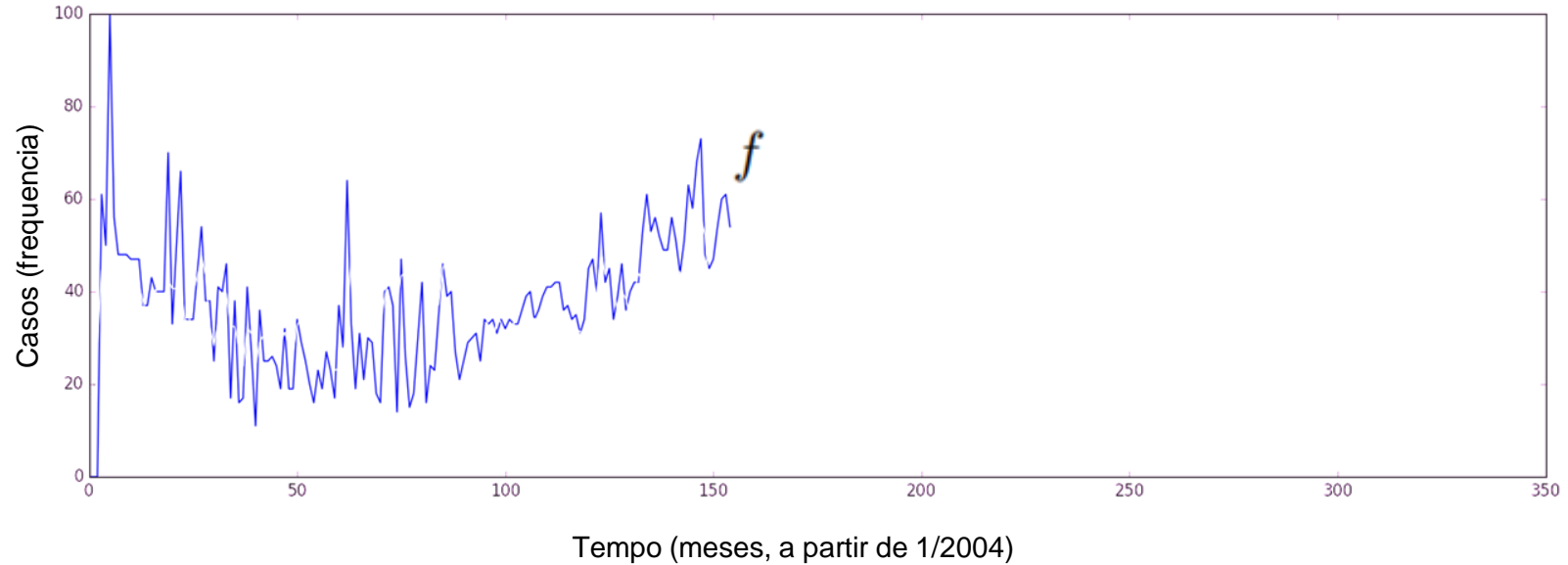
Regressão não Paramétrica baseada em Kernels

Solução

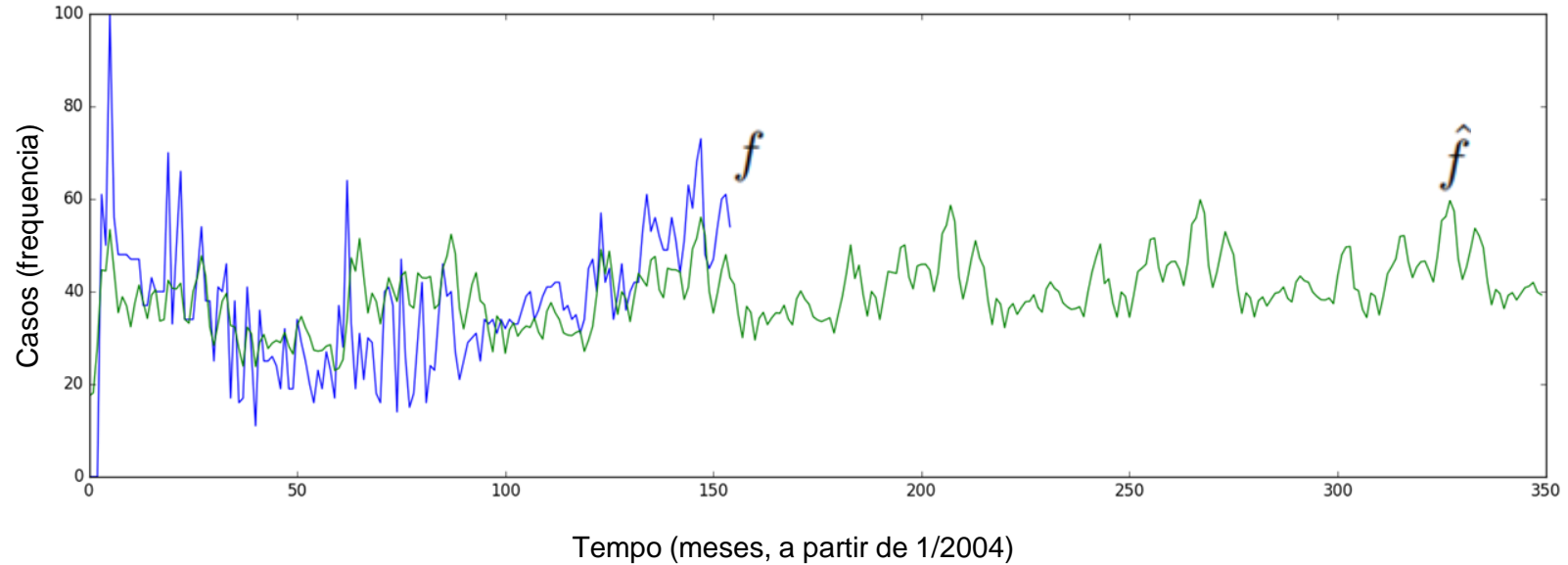
- ❖ Sinal composto de 4 partes
 - Tendência à longo prazo suave
 - Termo periodico de período longo
 - Termo periodico de período curto
 - Termo de ruído branco


$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K \left(\frac{x - X_i}{h} \right),$$

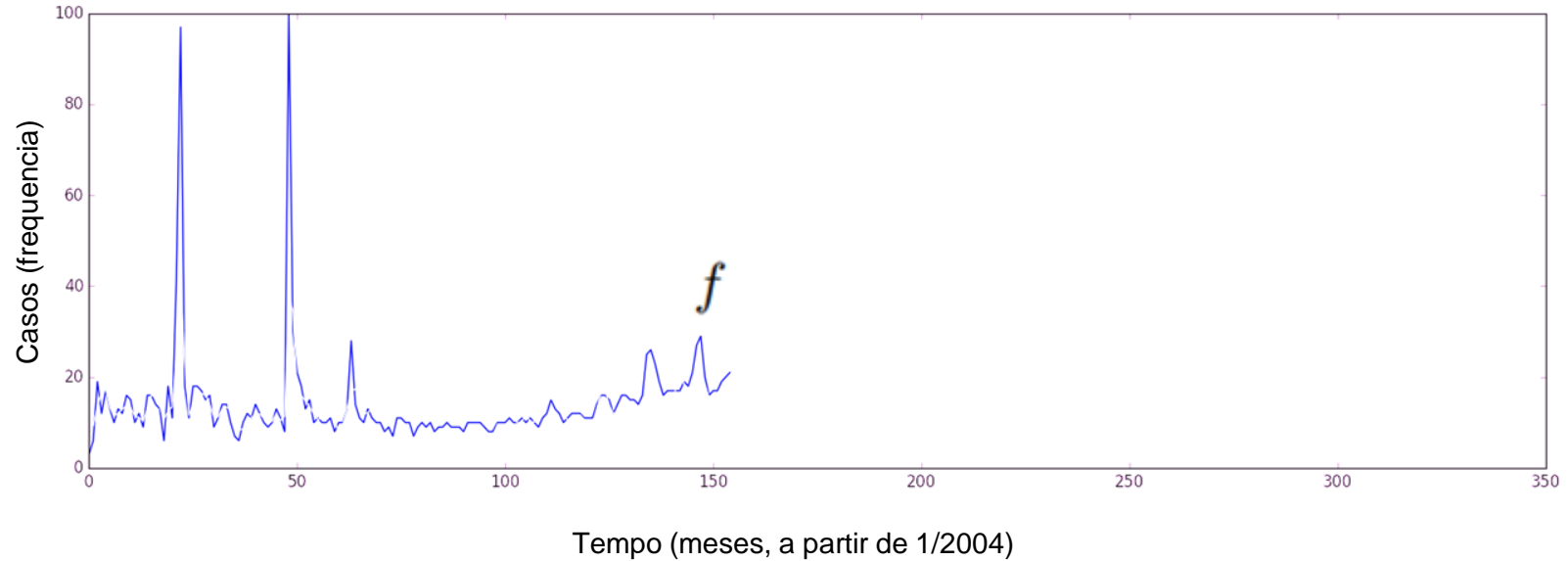
Regressão não paramétrica - Dores no corpo - SP



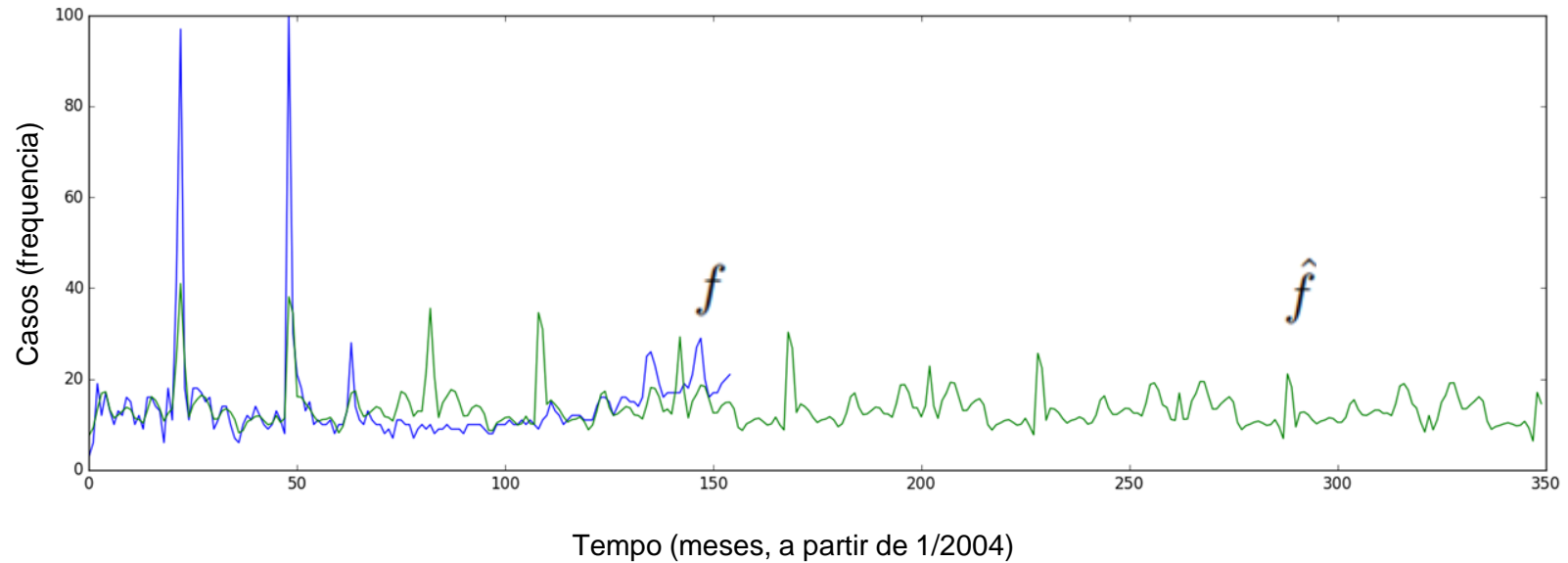
Regressão não paramétrica - Dores no corpo - SP



Regressão não paramétrica - Febre - SP



Regressão não paramétrica - Febre - SP



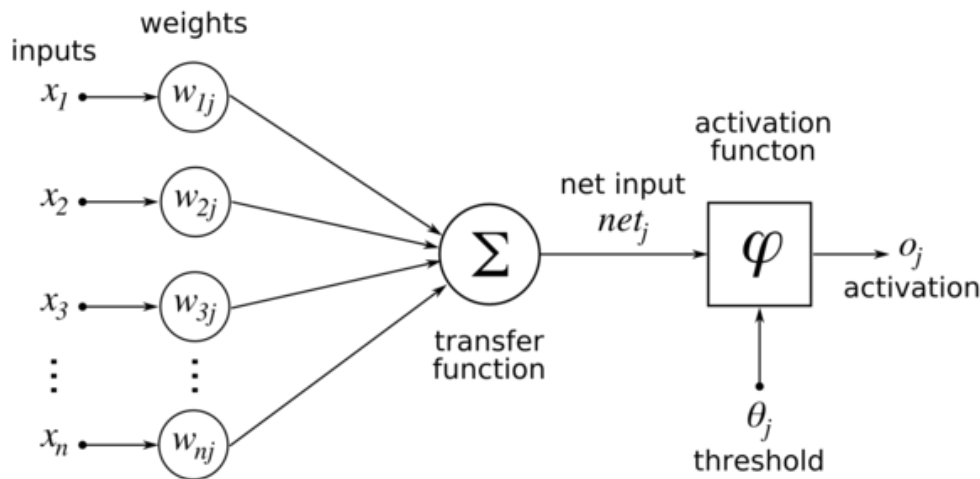
Redes Neurais Artificiais

Problema

❖ Aproximação de Funções

Como encontrar uma função que represente os dados desejados?

Como determinar uma saída, dada uma entrada?



Redes Neurais Artificiais

Solução

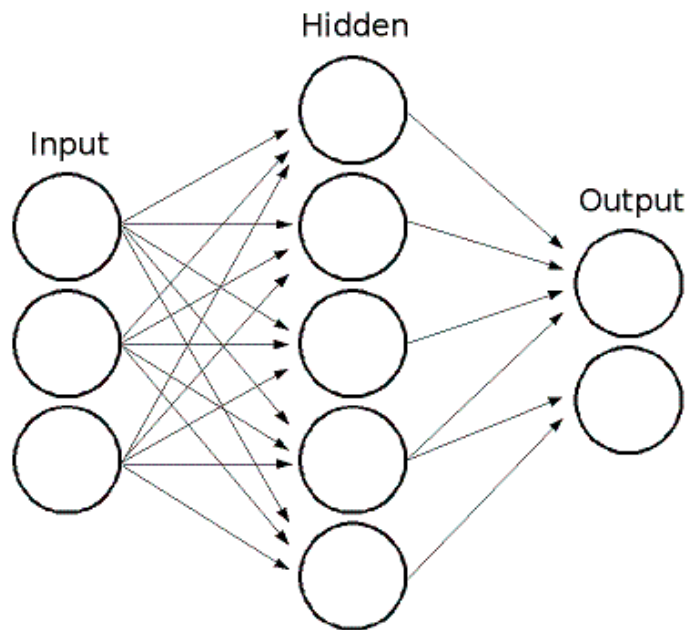
❖ Redes Neurais MLP

Rede Neural simples, sem realimentações e memória;

Uma única camada intermediária;

```
04  
65 -  
66
```

```
Syn(fold) = [tanh(input*w1') ones(1,1)]*w2';
```



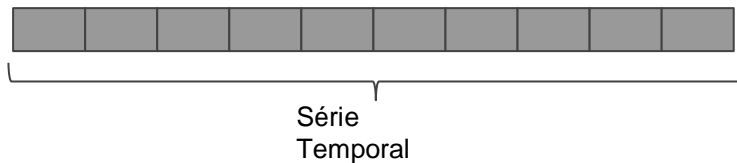
Redes Neurais Artificiais

Treinamento

- ❖ Precisamos prever uma série temporal

Utilizamos os dados do Google Trends para treinamento.

Os dados são divididos em 10 pastas.



Treinamos 10 redes neurais.

Para cada treinamento, uma pasta diferente é utilizada para validação;



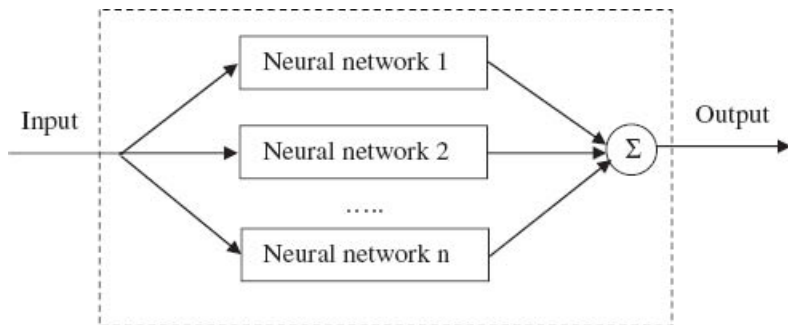
Redes Neurais Artificiais

Treinamento

❖ Precisamos prever uma série temporal

Para cada série teremos 10 redes neurais treinadas com conjuntos diferentes;

Fazer um ensemble de redes neurais garante que a média das previsões será melhor;



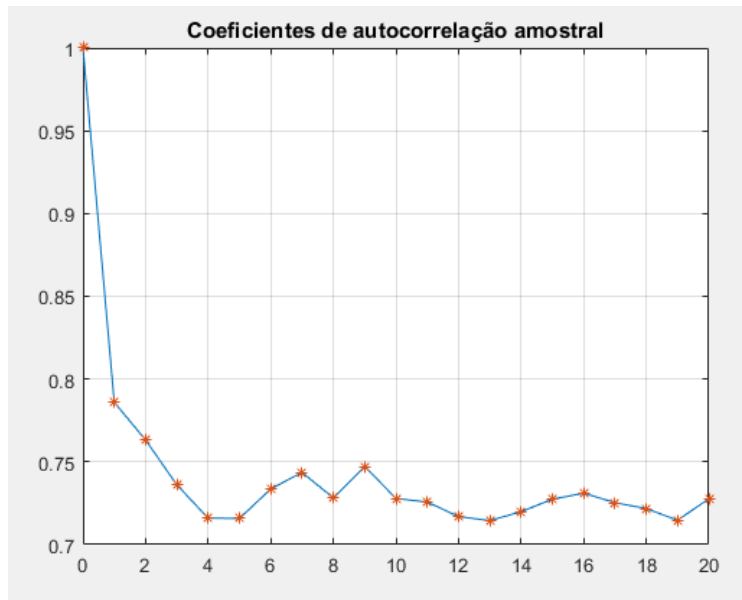
Escolhemos o valor previsto como o valor médio das 10 redes neurais

Redes Neurais Artificiais

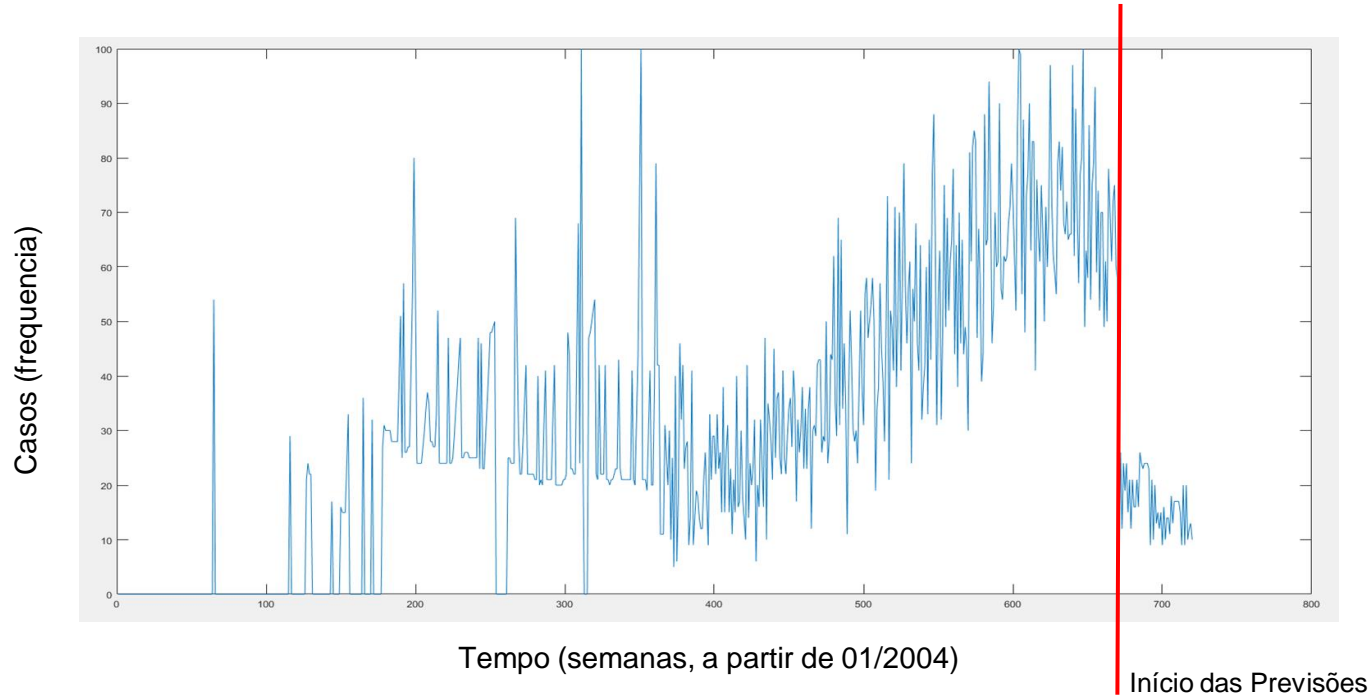
Testes

❖ 2350 Redes Neurais treinadas

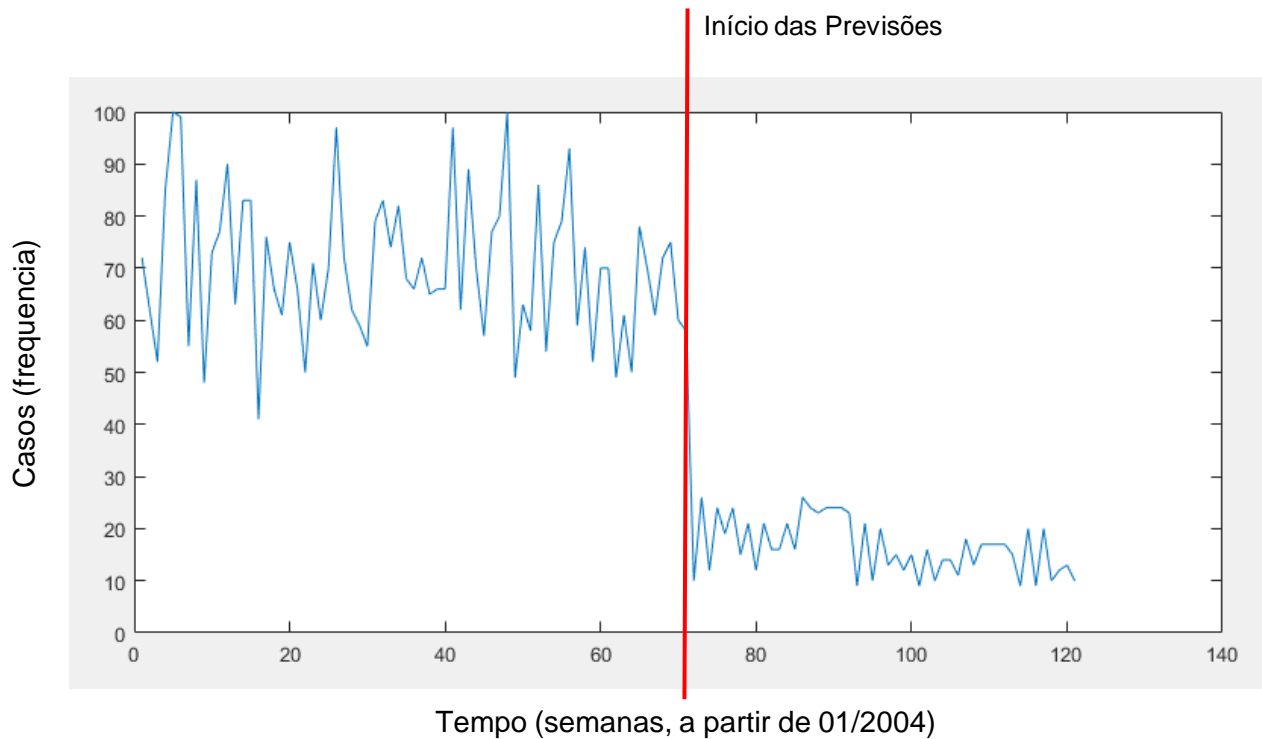
Cara rede neural com uma arquitetura diferente devido ao coeficiente de autocorrelação da série temporal;



Redes Neurais - Previsões



Redes Neurais - Previsões

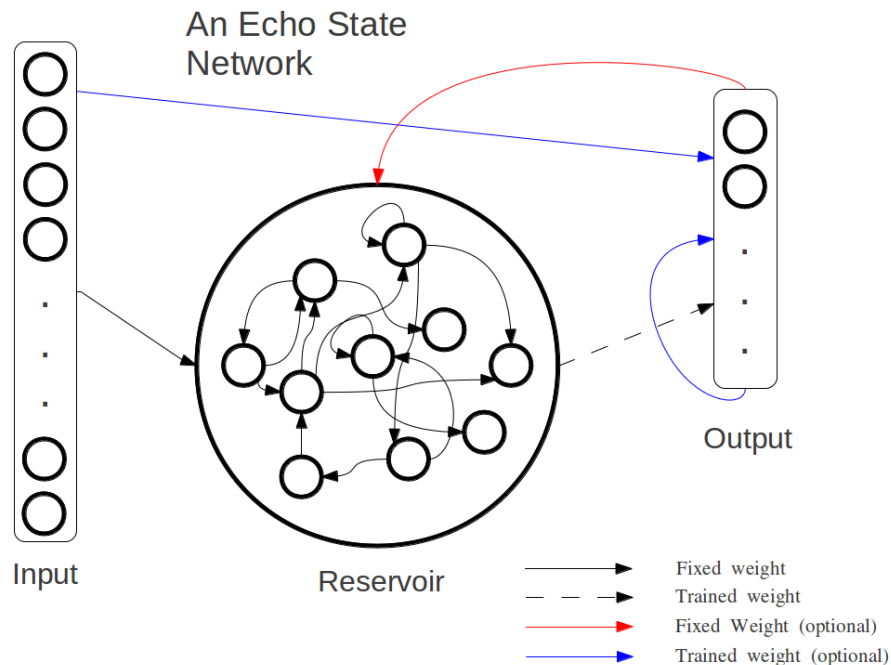


Redes Neurais Artificiais

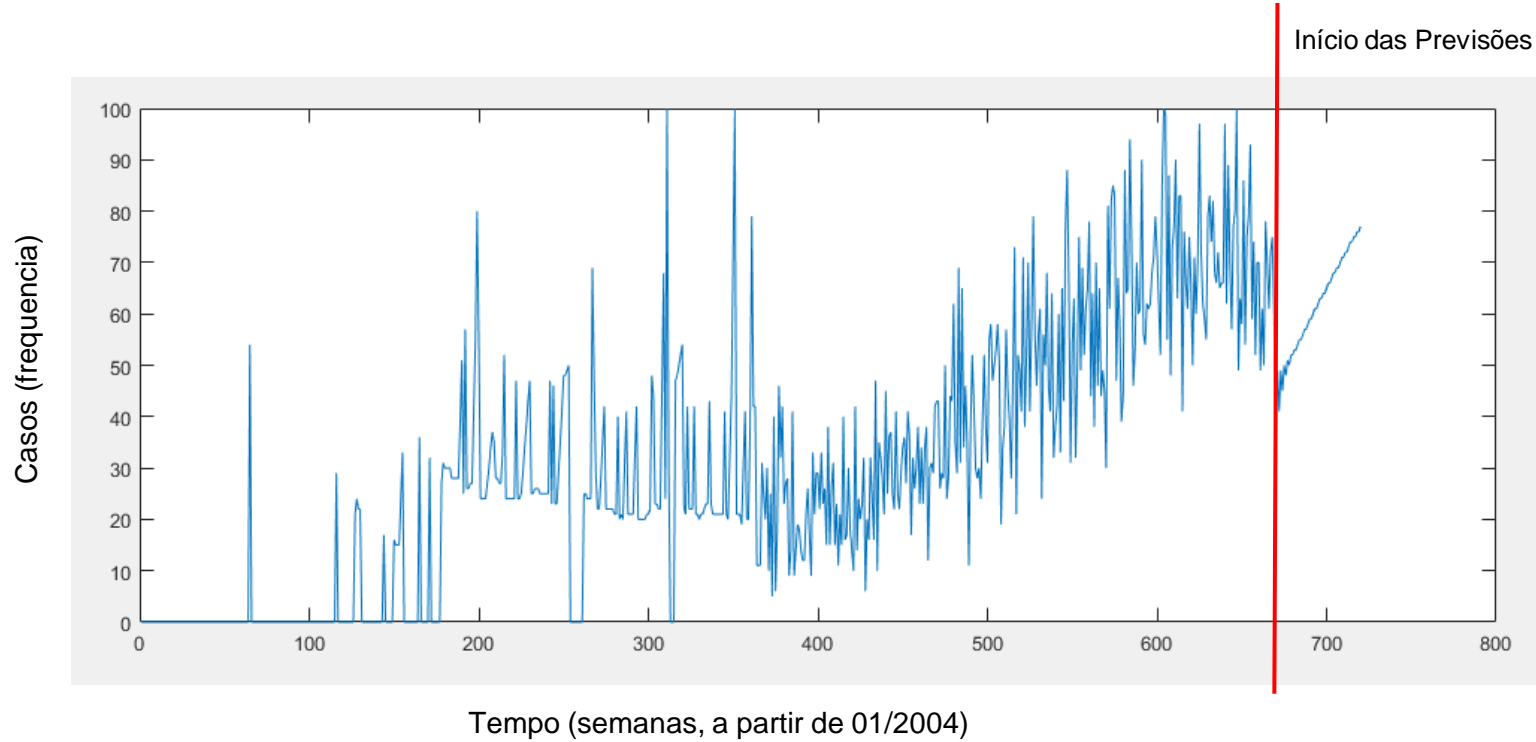
Outras Arquiteturas

❖ Redes Neurais Recorrentes

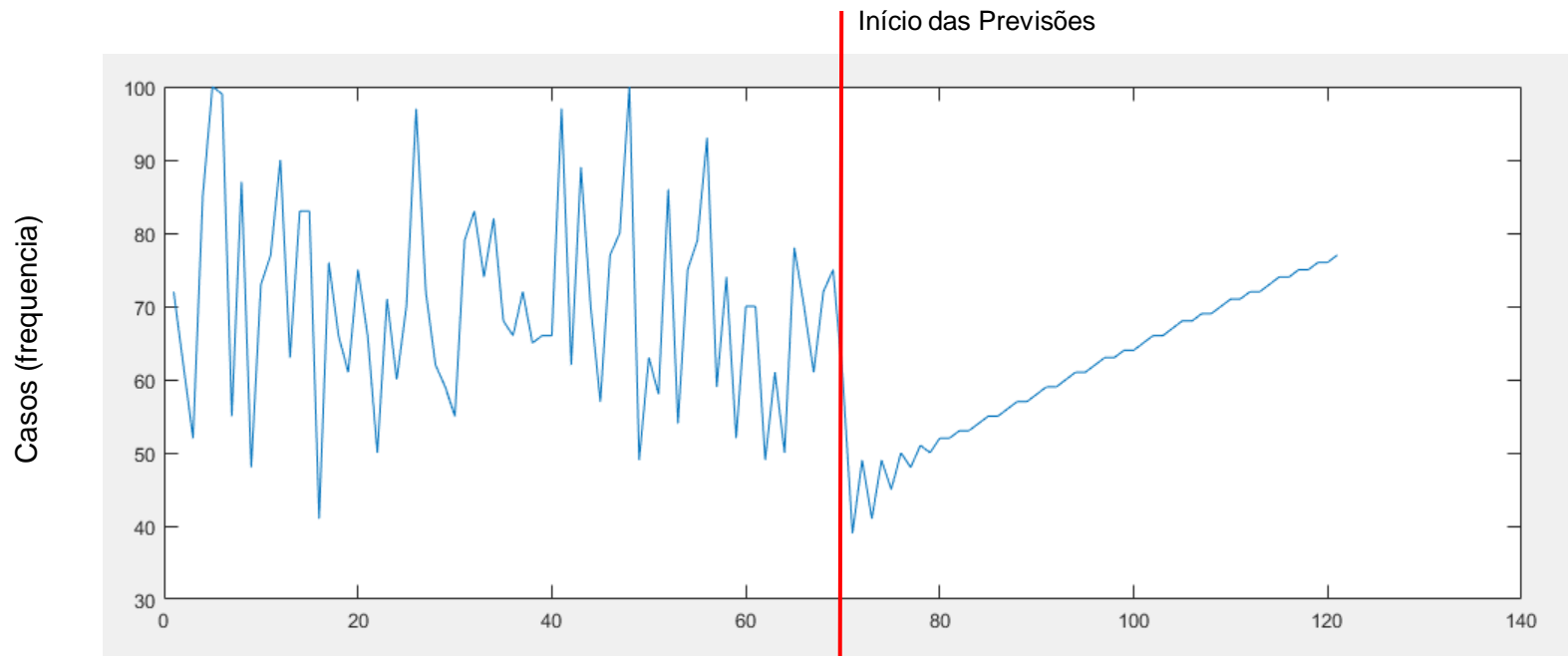
Teste de aplicação de redes neurais de estado eco



Redes Neurais - Previsões



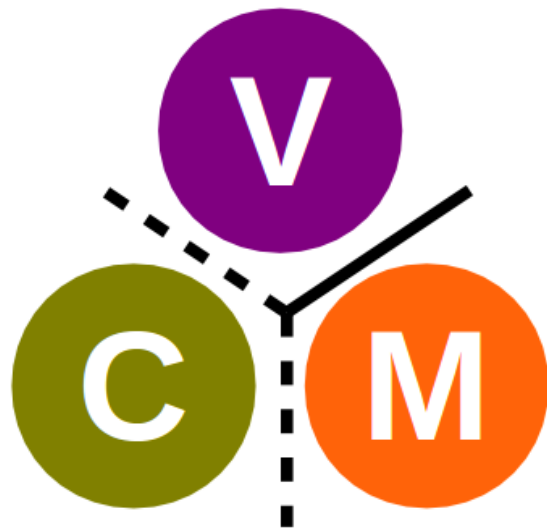
Redes Neurais - Previsões



Tempo (semanas, a partir de 01/2004)

Integração

django



Integração - Banco

```
1 from __future__ import unicode_literals
2
3 from django.db import models
4
5 # Create your models here.
6
7 class Medicamento(models.Model):
8     registro = models.CharField(max_length = 30, primary_key=True)
9     apresentacao = models.CharField(max_length = 300)
10    nome_venda = models.CharField(max_length = 20)
11    classe_terapeutica = models.CharField(max_length = 20)
12    principio_ativo = models.CharField(max_length = 20)
13    #laboratorio = models.CharField(max_length = 20)
14    #hospitalar = models.CharField(max_length = 20)
15
16    def __str__(self):
17        return self.nome_venda
18
19 class Doenca(models.Model):
20     #nome_cientifico = models.CharField(max_length = 20, primary_key=True)
21     nome_popular = models.CharField(max_length = 20, primary_key=True)
22
23     def __str__(self):
24         return self.nome_popular
25
26 class Sintoma(models.Model):
27     nome = models.CharField(max_length= 50, primary_key=True)
28
29     def __str__(self):
30         return self.nome
31
32 class Regiao(models.Model):
33     nome = models.CharField(max_length = 20, primary_key=True)
34
35     def __str__(self):
36         return self.nome
37
38 class Previsao(models.Model):
39     tipo = models.CharField(max_length=20)
40     resultado = models.TextField()
41     regiao = models.ForeignKey(Regiao, on_delete=models.CASCADE, default=None)
42     sintoma = models.ForeignKey(Sintoma, on_delete=models.CASCADE, default=None)
43
44     def __str__(self):
45         return "resultado do tipo: " + self.tipo + " para o sintoma: " + self.sintoma.nome + " na regiao: " + self.regiao.nome
46
47 class Trata(models.Model):
48     medicamento = models.ForeignKey(Medicamento, on_delete=models.CASCADE)
49     sintoma = models.ForeignKey(Sintoma, on_delete=models.CASCADE, default=None)
50
51     def __str__(self):
52         return self.medicamento.nome_venda + " trata " + self.sintoma.nome
```

- ❖ Abstração em cima do Sqlite3
- ❖ Todos os comandos são dados em python
- ❖ Modelo é representado como classes de python

Integração - Banco

```
1 from __future__ import unicode_literals
2
3 from django.db import models
4
5 # Create your models here.
6
7 class Medicamento(models.Model):
8     registro = models.CharField(max_length=30, primary_key=True)
9     apresentacao = models.CharField(max_length=300)
10    nome_venda = models.CharField(max_length=20)
11    classe_terapeutica = models.CharField(max_length=20)
12    principio_ativo = models.CharField(max_length=20)
13    #laboratorio = models.CharField(max_length=20)
14    #hospitalar = models.CharField(max_length=20)
15
16    def __str__(self):
17        return self.nome_venda
18
19 class Doenca(models.Model):
20     #nome_cientifico = models.CharField(max_length=20, primary_key=True)
21     nome_popular = models.CharField(max_length=20, primary_key=True)
22
23     def __str__(self):
24         return self.nome_popular
25
26 class Sintoma(models.Model):
27     nome = models.CharField(max_length=50, primary_key=True)
28
29     def __str__(self):
30         return self.nome
31
32 class Regiao(models.Model):
33     nome = models.CharField(max_length=20, primary_key=True)
34
35     def __str__(self):
36         return self.nome
37
38 class Previsao(models.Model):
39     tipo = models.CharField(max_length=20)
40     resultado = models.TextField()
41     regiao = models.ForeignKey(Regiao, on_delete=models.CASCADE, default=None)
42     sintoma = models.ForeignKey(Sintoma, on_delete=models.CASCADE, default=None)
43
44     def __str__(self):
45         return "resultado do tipo: " + self.tipo + " para o sintoma: " + self.sintoma.nome
46
47 class Trata(models.Model):
48     medicamento = models.ForeignKey(Medicamento, on_delete=models.CASCADE)
49     sintoma = models.ForeignKey(Sintoma, on_delete=models.CASCADE, default=None)
50
51     def __str__(self):
52         return self.medicamento.nome_venda + " trata " + self.sintoma.nome
```

- ❖ Abstração em cima do Sqlite3
- ❖ Todos os comandos são dados em python
- ❖ Modelo é representado como classes de python

```
thales@THALES:/mnt/c/Users/Thales/Documents/GitHub/lastTry/MC536---Data-Mining-Nao-Machine-Learning/PrevisaoEstoque$ python manage.py sqlmigrate estoque 0012
BEGIN;
--
-- Add field sintoma to previsao
--
ALTER TABLE "estoque_previsao" RENAME TO "estoque_previsao_old";
CREATE TABLE "estoque_previsao" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "sintoma_id" varchar(50) NOT NULL REFERENCES "estoque_sintoma" ("nome"), "tipo" varchar(20) NOT NULL, "resultado" text NOT NULL, "regiao_id" varchar(20) NOT NULL REFERENCES "estoque_regiao" ("nome"));
INSERT INTO "estoque_previsao" ("regiao_id", "resultado", "sintoma_id", "id", "tipo") SELECT "regiao_id", "resultado", NULL, "id", "tipo" FROM "estoque_previsao_old";
DROP TABLE "estoque_previsao_old";
CREATE INDEX "estoque_previsao_57934e61" ON "estoque_previsao" ("sintoma_id");
CREATE INDEX "estoque_previsao_4968fcf4" ON "estoque_previsao" ("regiao_id");
COMMIT;
```

Integração - Banco

```
[28/Nov/2016 13:31:58] "GET /admin/ HTTP/1.1" 200 6927
[28/Nov/2016 13:32:03] "GET /admin/estoque/sintoma/ HTTP/1.1" 200 18183
[28/Nov/2016 13:32:03] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 13:32:15] "GET /admin/estoque/sintoma/APARECIMENTO%20DE%20%20FERIDAS%20NA%20PELE/change/ HTTP/1.1" 200 4196
[28/Nov/2016 13:32:15] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 13:32:38] "POST /admin/estoque/sintoma/APARECIMENTO%20DE%20%20FERIDAS%20NA%20PELE/change/ HTTP/1.1" 302 0
[28/Nov/2016 13:32:38] "GET /admin/estoque/sintoma/ HTTP/1.1" 200 18750
[28/Nov/2016 13:32:38] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 13:32:39] "GET /static/admin/img/icon-yes.svg HTTP/1.1" 200 436
^[[A[28/Nov/2016 13:32:57] "GET /admin/estoque/sintoma/APARECIMENTO%20DE%20%20FERIDAS%20NA%20PELE/change/ HTTP/1.1" 200 4196
[28/Nov/2016 13:33:01] "GET /admin/estoque/ HTTP/1.1" 200 4657
[28/Nov/2016 13:33:05] "GET /admin/estoque/previsao/ HTTP/1.1" 200 4581
[28/Nov/2016 13:33:05] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 13:33:07] "GET /admin/estoque/ HTTP/1.1" 200 4657
[28/Nov/2016 13:35:55] "GET /admin/estoque/sintoma/ HTTP/1.1" 200 18499
[28/Nov/2016 13:35:56] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 13:36:19] "GET /admin/estoque/sintoma/ARRITIMIA/change/ HTTP/1.1" 200 4085
[28/Nov/2016 13:36:19] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 13:36:34] "POST /admin/estoque/sintoma/ARRITIMIA/change/ HTTP/1.1" 302 0
[28/Nov/2016 13:36:34] "GET /admin/estoque/sintoma/ HTTP/1.1" 200 18933
[28/Nov/2016 13:36:34] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 14:14:39] "GET /admin/ HTTP/1.1" 200 7581
[28/Nov/2016 14:14:39] "GET /admin/ HTTP/1.1" 200 7581
[28/Nov/2016 14:14:43] "GET /admin/estoque/sintoma/ HTTP/1.1" 200 18738
[28/Nov/2016 14:14:43] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 14:18:07] "GET /admin/estoque/sintoma/DOR%20NAS%20ARTICUACOES/change/ HTTP/1.1" 200 4133
[28/Nov/2016 14:18:08] "GET /admin/jsi18n/ HTTP/1.1" 200 3217
[28/Nov/2016 14:18:14] "POST /admin/estoque/sintoma/DOR%20NAS%20ARTICUACOES/change/ HTTP/1.1" 302 0
[28/Nov/2016 14:18:15] "GET /admin/estoque/sintoma/ HTTP/1.1" 200 19238
```

❖ Servidor de teste dado pelo Django

Integração - Banco

```
In [5]: import csv
...: from estoque.models import Sintoma
...: with open('sintomasSemAcentoInc.csv', 'rb') as f:
...:     reader = csv.reader(f)
...:     for row in reader:
...:         s = Sintoma(row[1])
...:         print s
...:         s.save()
...:
```

```
CANSACO
DOR ATRAS DOS OLHOS
DOR DE CABECA
DOR NAS ARTICUACOES
DOR MUSCULAR
DOR NOS OSSOS
FEBRE
MANCHAS VERMELHAS PELO CORPO
MOLEZA
NAUSEA
FALTA DE APETITE
TONTURA
VOMITO
AUMENTO DO TAMANHO DO BACO
AUMENTO DO TAMANHO DO FIGADO
DIARREIA
```

- ❖ Banco populado a partir do shell que o django prove
- ❖ Versão sql dos comandos em python para criação do banco

Integração - Banco

Django administration WELCOME, ADMIN [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

[Home](#) » [Estoque](#) » Sintomas

Select sintoma to change ADD SINTOMA +

Action: 0 of 57 selected

<input type="checkbox"/>	SINTOMA
<input type="checkbox"/>	VOMITO
<input type="checkbox"/>	TOSSE
<input type="checkbox"/>	TONTURA
<input type="checkbox"/>	SURGIMENTO DE NODULOS
<input type="checkbox"/>	SUDORESE
<input type="checkbox"/>	SONOLENCIA
<input type="checkbox"/>	SANGUE NAS FEZES
<input type="checkbox"/>	RUBOR FACIAL
<input type="checkbox"/>	RIGIDEZ NOS MUSCULOS DO ABDOMEN
<input type="checkbox"/>	RIGIDEZ NOS MUSCULOS DE PESCOCO E NUCA
<input type="checkbox"/>	RIGIDEZ NO MAXILAR
<input type="checkbox"/>	PESCOCO RIGIDO
<input type="checkbox"/>	PERDA DE PESO

- ❖ Site de administração
- ❖ Facilita a visualização do banco
- ❖ Pode se alterar os dados visualmente

Integração - Front End

```
def filtro(request):
    return render(request, 'estoque/filtro.html')

def equipe(request):
    return render(request, 'estoque/equipe.html')

def help(request):
    return render(request, 'estoque/help.html')

def mapa(request):
    return render(request, 'estoque/mapa.html')

def medicamentos(request, nomeDoenca):
    doenca = Doenca.objects.get(nome_popular=nomeDoenca)
    causas = Causa.objects.filter(doenca=doenca)
    medicamentos = []

    for causa in causas:
        med = Trata.objects.filter(sintoma=causa.sintoma)
        for m in med:
            aux = [
                m.medicamento.registro,
                m.medicamento.apresentacao,
                m.medicamento.principio_ativo,
                m.medicamento.nome_venda,
                m.medicamento.classe_terapeutica
            ]
            medicamentos.append(aux)

    json_data = json.dumps(medicamentos)

    context = {'medicamentos': json_data}

    return render(request, 'estoque/medicamentos.html', context)
```

❖ Endpoints

❖ Urls

Integração - Front End

```
def filtro(request):
    return render(request, 'estoque/filtro.html')

def equipe(request):
    return render(request, 'estoque/equipe.html')

def help(request):
    return render(request, 'estoque/help.html')

def mapa(request):
    return render(request, 'estoque/mapa.html')

def medicamentos(request, nomeDoenca):
    doenca = Doenca.objects.get(nome_popular=nomeDoenca)
    causas = Causa.objects.filter(doenca=doenca)
    medicamentos = []

    for causa in causas:
        med = Trata.objects.f
        for m in med:
            aux = [
                m.medicamento
                m.medicamento
                m.medicamento
                m.medicamento
                m.medicamento
            ]
            medicamentos.append(aux)

    json_data = json.dumps(medicamentos)
    context = {'medicamentos': json_data}

    return render(request, 'estoque/medicamentos.html', context)
```

❖ Endpoints

❖ Urls

```
urlpatterns = [
    # url(r'^updateDB/$', views.updateDB, name='updateDB'),
    # url(r'^fit/$', views.fit, name='fit'),
    url(r'^filtro/$', views.filtro, name='filtro'),
    url(r'^equipe/$', views.equipe, name='equipe'),
    url(r'^help/$', views.help, name='help'),
    url(r'^mapa/$', views.mapa, name='mapa'),
    url(r'^medicamentos/(?P<nomeDoenca>([A-Z])\w*)/$', views.medicamentos, name='medicamentos'),
    url(r'^previsao/(?P<registro>[0-9]+)/$', views.previsao, name='previsao'),
]
```


Integração - Front End

```
<!-- Bootstrap core CSS -->
<link href="{% static "estoque/bootstrap.min.css" %}" rel="stylesheet">

<!-- IE10 viewport hack for Surface/desktop Windows 8 bug -->
<link href="./files/ie10-viewport-bug-workaround.css" rel="stylesheet">

<!-- Custom styles for this template -->
<link href="{% static "estoque/dashboard.css" %}" rel="stylesheet">

<!-- Just for debugging purposes. Don't actually copy these 2 lines! -->
<!--[if lt IE 9]><script src="../../assets/js/ie8-responsive-file-warning.js"></script><![endif]-->
<script src="./files/ie-emulation-modes-warning.js.download"></script>
```

- ❖ Views Feitas a partir de templates em html

0 Front-End

- ❖ Framework MVC Django
- ❖ Desenvolvimento da View
- ❖ Criação das páginas através do Bootstrap
- ❖ Utilização de html, JavaScript, Json e da biblioteca d3.js

django



{JSON}



Bootstrap



JavaScript

</>
HTML

O Front-End - Página Inicial

- ❖ A busca pela predição do estoque de um medicamento se dá através da escolha de uma das doenças cadastradas no banco.
- ❖ A partir da escolha da doença o banco busca os sintomas relacionados e devolve a lista de medicamentos que tratam esses sintomas.
- ❖ A lista desses medicamentos foi implementada utilizando o framework DataTables do Bootstrap.
- ❖ Escolhendo um dos medicamentos, temos acesso à predição do seu estoque por estado do Brasil.



O Front-End - Predição

- ❖ A predição é exibida através de dois mapas do Brasil. Um para a predição a base de kernel e outro para a predição com Redes Neurais.
- ❖ Para a criação do mapa utilizamos a biblioteca d3.js que roda um topojson no JavaScript utilizando um arquivo Json com as coordenadas dos estados dos país.
- ❖ Com o mapa desenhado, separamos os estados pelo seu id no arquivo Json e fomos capazes de utilizar os valores provenientes das predições para colorir o mapa de acordo com a quantidade do medicamento selecionado por estado.
- ❖ Quanto maior a tonalidade, maior deverá ser o estoque.
- ❖ O intervalo de tempo desejado na previsão é selecionado acima dos mapas e a mudança no intervalo acarreta mudança na predição do mapa.

O Front-End - Predição

Slider

```
<script src="dados.js"></script>
```

```
<script>  
var time = 1;
```

```
$("#slider").slider({  
  min: 1,  
  max: 770,  
  value: 1,  
  tooltip: 'always'});
```

```
$("#slider").on("slide", function(slideEvt) {  
  $("#time").text(slideEvt.value);  
  time = slideEvt.value;  
  d3_queue.queue()  
    .defer(d3.json, "files/br.json")  
    // .defer(d3.tsv, "files/qtd.tsv")  
    .await(ready);  
});
```

```
var width = 600;  
var height = 500;
```

```
var svg1 = d3.select('div[id="svg1"]').append("svg")  
  .attr("id", "svg1")  
  .attr("width", "100%")  
  .attr("height", 500)  
  .style("position", "relative");
```

```
var svg2 = d3.select('div[id="svg2"]').append("svg")  
  .attr("id", "svg2")  
  .attr("width", "100%")  
  .attr("height", 500)  
  .style("position", "relative");
```

```
var g1 = svg1.append("g");  
var g2 = svg2.append("g");
```

```
var color1 = d3.scale.threshold()  
  .domain([20, 30, 40, 50, 60, 70, 80, 90, 100])  
  .range(["#ffe6e6", "#ffcccc", "#ffb3b3", "#ff9999", "#ff8080", "#ff6666", "#ff4d4d", "#ff3333", "#ff1a1a", "#ff0000"]);
```

```
var color2 = d3.scale.threshold()  
  .domain([20, 30, 40, 50, 60, 70, 80, 90, 100])  
  .range(["#e666cc", "#e6b3e6", "#df9fdf", "#d98cd9", "#d279d2", "#cc66cc", "#c653c6", "#bf40bf", "#ac39ac", "#993399"]);
```

Tonalidades

0 Front-End - Help

- ❖ Como complemento e para criar uma página mais real que pudesse ser utilizadas por profissionais da saúde, criamos uma aba de *help* que explica como utilizar a ferramenta disponível na página.

O Front-End - Equipe

- ❖ Ainda no intuito de deixar a aplicação mais real, criamos a aba *Equipe* que contém a foto, o nome e a função de cada membro do cluster.
- ❖ Os membros das duplas estão lado a lado, para melhor visualização e entendimento dos papéis

DEMONSTRAÇÃO DA APLICAÇÃO