



Engenharia Elétrica
PET ENGENHARIA ELÉTRICA

**FERRAMENTA DE ESTATÍSTICA PESQUEIRA PARA ANÁLISE DE DADOS DE
PESCA**

**Palmas,
2018**

SUMÁRIO

1. INFORMAÇÕES GERAIS	3
1.1. USO DA FERRAMENTA	3
1.2. OBJETIVO E APLICAÇÕES	3
2. USANDO A FERRAMENTA	3
2.1. ALIMENTANDO A BASE DE DADOS	3
2.2. FILTROS	4
2.3. PROCESSAMENTO	5
2.4. SAIDAS	6
2.5. EXEMPLO	7

1. INFORMAÇÕES GERAIS

1.1. USO DA FERRAMENTA

Por estar em estágio inicial de desenvolvimento, a ferramenta atua como um framework, ou seja, traz um conjunto de bibliotecas desenvolvido pela própria equipe para executar cálculos de estatística pesqueira. Nessa primeira fase, portanto, o arquivo que usará as bibliotecas deve estar no mesmo diretório das mesmas (mais informações sobre o uso serão dadas nas próximas seções).

1.2. OBJETIVO E APLICAÇÕES

O foco da ferramenta aqui documentada é facilitar a análise de dados oriundos de diversos tipos de pescaria. Em particular, atualmente são gerados gráficos e planilhas contendo a produção total e média categorizada por diversas variáveis, incluindo tipo de pesca, localidade, potência do motor, ano, mês, pescador e espécie. O programa ainda é capaz de realizar análise de correlação e regressão (exponencial e linear) da produção em função do ano, potência do motor, mês, comprimento do barco e algumas outras variáveis numéricas que serão apresentadas posteriormente.

2. USANDO A FERRAMENTA

Para usar a ferramenta, é necessário importar os módulos *processamento*, *saída* e *filtros*. Além disso, devem ser instalados do repositório padrão os módulos: *mysql-connector-python*, *numpy*, *pandas* e *matplotlib*. Além disso, salienta-se que, para realizar os testes do evento testes, deve-se executar o script *mysql* fornecido junto com a ferramenta.

2.1. ALIMENTANDO A BASE DE DADOS

Para alimentar a base de dados, basta importar os módulos mencionados e fazer o seguinte:

```
processo = processamento.Processamento()
processo.inserir_producao_csv("base")
```

2.2. FILTROS

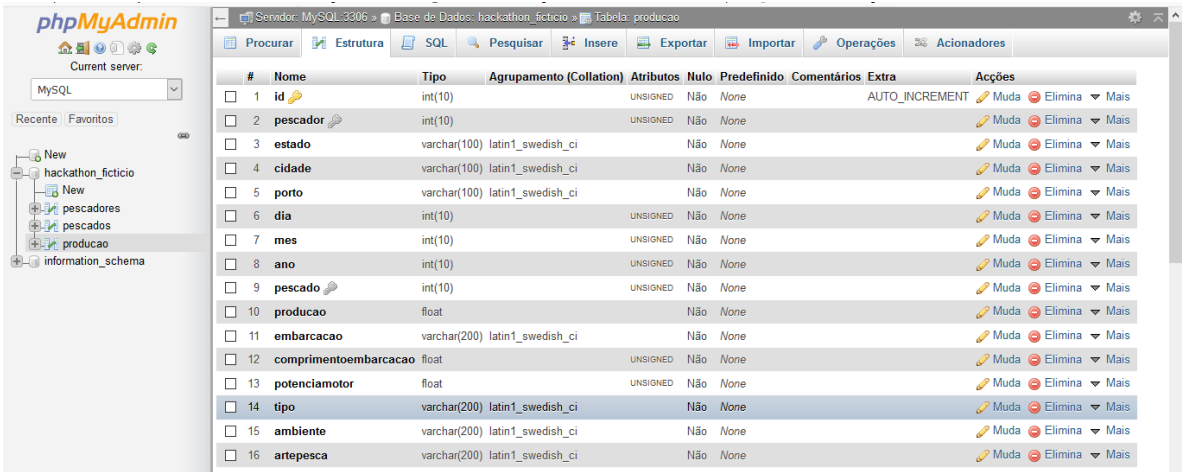
Os filtros são usados para determinar quais dados serão analisados e quais não serão. A sua sintaxe é a seguinte:

```
import filtros

var = filtros.Filtros(campo, relação, valor)
```

O valor de campo pode ser qualquer uma as colunas da tabela *producao*, conforme imagem abaixo:

Figura 1 - Tabela Producao



Os campos relação e valor podem ser entendidos conforme tabela abaixo:

Relação	Valor	Significado
igual	a	== a
diferente	a	!= a
menor	a	< a
maior	a	> a
menorigual	a	<= a

maior igual	a	$\geq a$
intervalo fechado	(a,b)	$\geq a \ \&\& \leq b$
intervalo aberto	(a,b)	$> a \ \&\& < b$

Os filtros ainda possuem sobrecarga dos operadores de multiplicação, que realiza uma operação and com os filtros e o operador da adição, que realiza uma operação or. Por exemplo, deseja-se criar um filtro que selecione o ano de 1978 e o estado da Paraíba:

```
filtro1 = filtros.Filtro("ano", "igual", 1978)
filtro2 = filtros.Filtro("estado", "igual", "'Paraíba'")
filtro3 = filtro1 * filtro2
```

Observe as aspas simples que envolvem o estado da Paraíba, para indicar que é um campo de texto na base de dados.

A classe filtro possui um método de classe para criar um filtro customizado, no qual você simplesmente insere o código MySQL como parâmetro:

```
filtro1 = filtros.Filtro.filtro_customizado(condicao)
```

2.3. PROCESSAMENTO

Essa classe é responsável por processar os dados. Aqui, serão abordados os dois métodos mais usados. O Primeiro deles deixa a função produção pesqueira, média ou total, em função de qualquer outra grandeza da base dados. A sua sintaxe é:

```
processo = processamento.Processamento()
processo.produto_em_funcao_de(variavel, grandeza, filtros)
```

O parâmetro *variavel* representa o campo da tabela que ficará como variável independente. O valor de *grandeza* é "media" ou "total", indicando se a produção será, respectivamente, média ou total. O valor de *filtros* é um objeto da classe explicada anteriormente.

A segunda função é responsável para fazer análise de regressão da produção, também média ou total, em função de algum outro campo numérico da tabela, a ser especificado. O método analisa se os pontos se ajustam melhor ao modelo linear ou exponencial, e então faz o ajuste que melhor se encaixa. Sua sintaxe é:

```
processo.ajustar_modelo(var_independente, grandeza, filtros)
```

Onde *var_independente* é o campo que assumirá o posto de variável independente.

2.4. SAIDAS

Antes de gerar a saída, é obrigatório configurar os textos do gráfico, logo, a seguinte função deve ser chamada:

```
exportar = saidas.Saidas()  
exportar.configurar_textos(eixo_x, eixo_y, titulo)
```

O parâmetro *eixo_x* serve para informar o título que terá o eixo x. De forma análoga, o *eixo_y* rotula as ordenadas e, por fim, *titulo* é o título do gráfico.

Após esse processo, podem-se usar três funções: *carregar_curvas*, *carregar_pontos* e *ajuste_pontos*. As duas primeiras servem para plotar uma saída do método *producao_em_funcao_de*, sendo a primeira usada para marcar apenas pontos, e a segunda é responsável por traçar linhas contínuas. A terceira função é usada para plotar os resultados de uma análise de regressão. As sintaxes são as seguintes:

```
exportar.carregar_pontos(dados, nome, eixo_x, eixo_y)  
exportar.carregar_curva(dados, nome, eixo_x, eixo_y)  
exportar.ajuste_pontos(modelo)
```

O parâmetro *dados* é o retorno da função *producao_em_funcao_de*. Já o *modelo* é o retorno da função *ajustar_modelo*. Em ambos os casos, o campo *nome* representa a identificação

daquela forma plotada na legenda. Por fim, para ver os gráficos, é necessário usar a função `exportar.plotar()`.

Outro ponto que vale a pena ressaltar é a possibilidade de salvar planilhas de Excel na pasta Tabelas, que o acompanha o diretório da ferramenta. Para tal, têm-se as seguintes funções:

`exportar.gerar_planilha(nome,dados)`

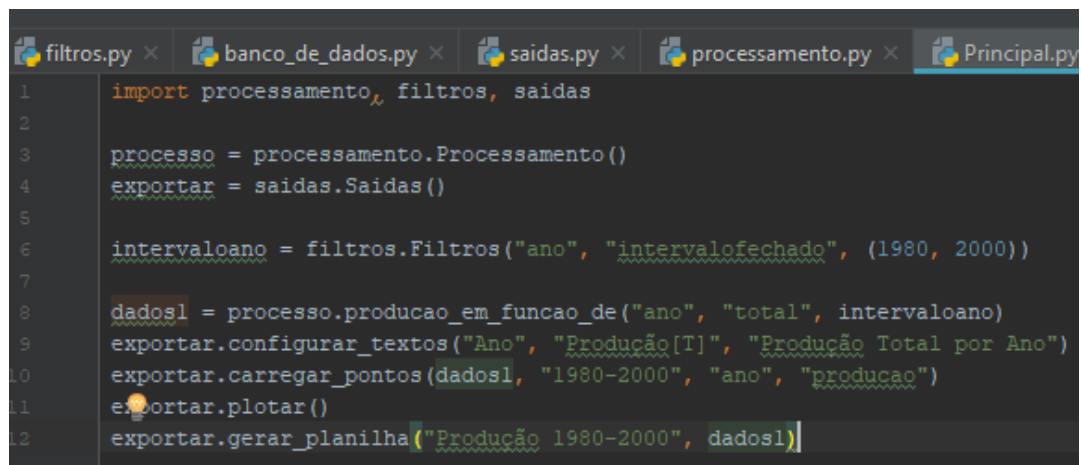
`exportar.planilha_ajuste(modelo)`

Onde *nome* é o nome do arquivo que será criado.

2.5. EXEMPLO

As linhas abaixo mostram um exemplo onde se plota a produção total por ano de 1980 a 2000, gerando gráfico e uma planilha.

Figura 2 - Exemplo



```
filtros.py x banco_de_dados.py x saidas.py x processamento.py x Principal.py
1 import processamento, filtros, saidas
2
3 processo = processamento.Processamento()
4 exportar = saidas.Saidas()
5
6 intervaloano = filtros.Filtros("ano", "intervalofechado", (1980, 2000))
7
8 dados1 = processo.producao_em_funcao_de("ano", "total", intervaloano)
9 exportar.configurar_textos("Ano", "Produção[T]", "Produção Total por Ano")
10 exportar.carregar_pontos(dados1, "1980-2000", "ano", "producao")
11 exportar.plotar()
12 exportar.gerar_planilha("Produção 1980-2000", dados1)
```