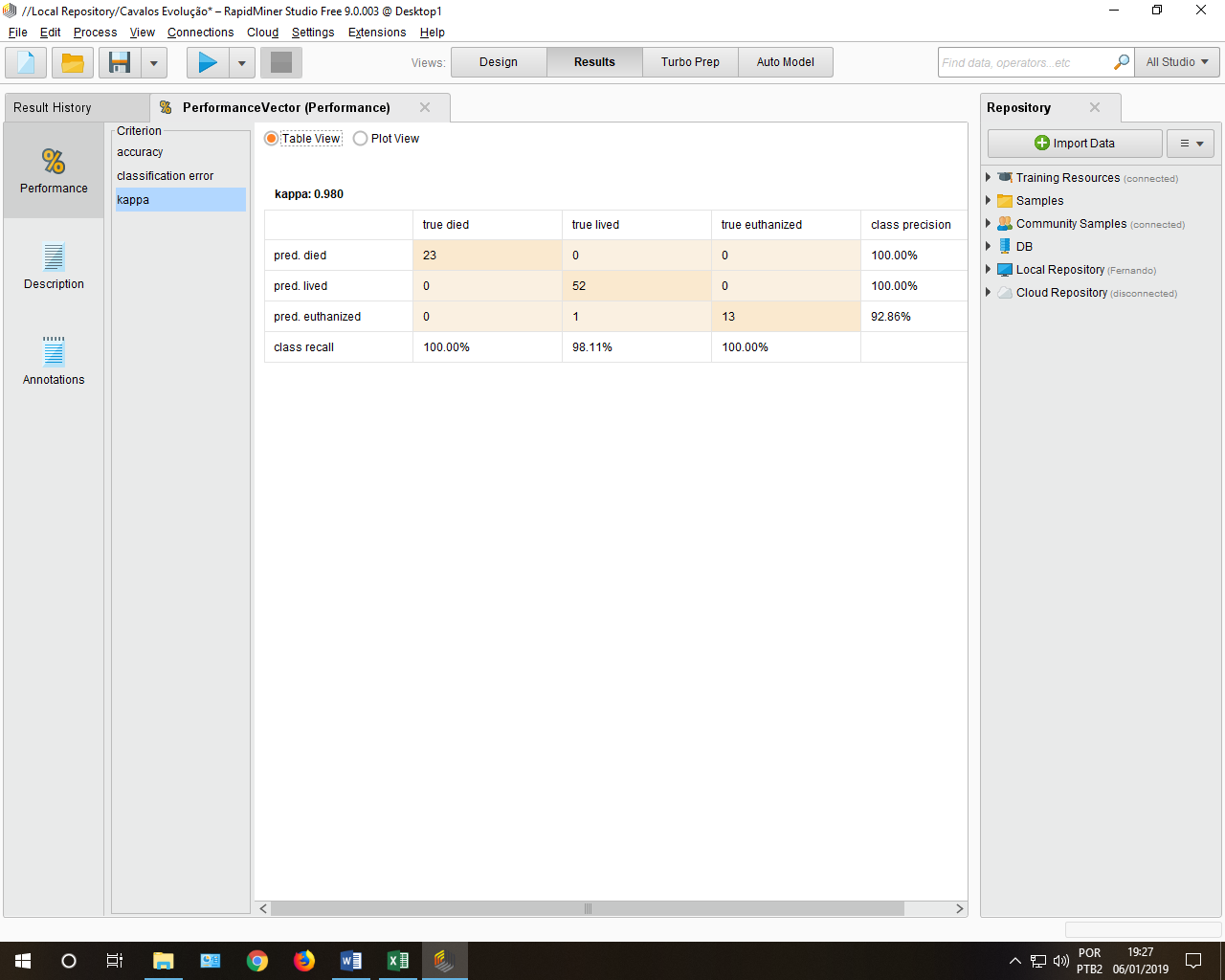


**ALUNO: FERNANDO MAURO BULEO BARBOSA**

**(buleo\_1021@esp.puc-rio.br)**

**RESUMO:**

* O trabalho foi realizado usando o **RapidMiner**
* Foi desenvolvida solução para aplicação de **Oversample** usando os operadores do **RapidMiner**
* Dentre as simulações realizadas a que proporcionou o melhor desempenho conta com a aplicação das seguintes técnicas:
* Tratamento de Missing Values
* Retirada de Atributos com muitos Missing Values (redução de dimensionalidade / análise exploratória dos dados)
* Balanceamento de Dados (OverSample)
* Seleção de atributos por peso (Weight by Information Gain / redução de dimensionalidade)
* Retirada de Outliers
* Aplicação do Modelo Gradient Boosted Trees
* O melhor desempenho obtido foi **Acurácia** de **98,88%** e **Kappa** de **0,98**
* Somente 1 registro da base de teste foi classificado com erro na simulação em que obtive o melhor desempenho



* Apresento nas próximas páginas o relatório detalhando cronologicamente os experimentos realizados
* Ao fim desse documento apresento os seguintes Anexos:
* Relação completa dos experimentos realizados e respectivos resultados
* Processos implementados no RapidMiner

1. **Análise Exploratória dos Dados**

Utilizei o operador “Read CSV” para ler as bases de treino e teste. Identifiquei os seguintes aspectos na análise exploratória resultante do processamento desse operador.

***Papéis (Role):***

Identifiquei a necessidade de alterar os papéis (role) de alguns campos:

* Hospital\_number 🡪 ID
* OutCome 🡪 Label

***Tipos de Dados (Type):***

Quanto aos tipos de dados (type) considerei os seguintes campos numéricos como Integer:

* **hospital\_number**
* **pulse**
* **respiratory\_rate**
* **lesion\_1**
* **lesion\_2**
* **lesion\_3**

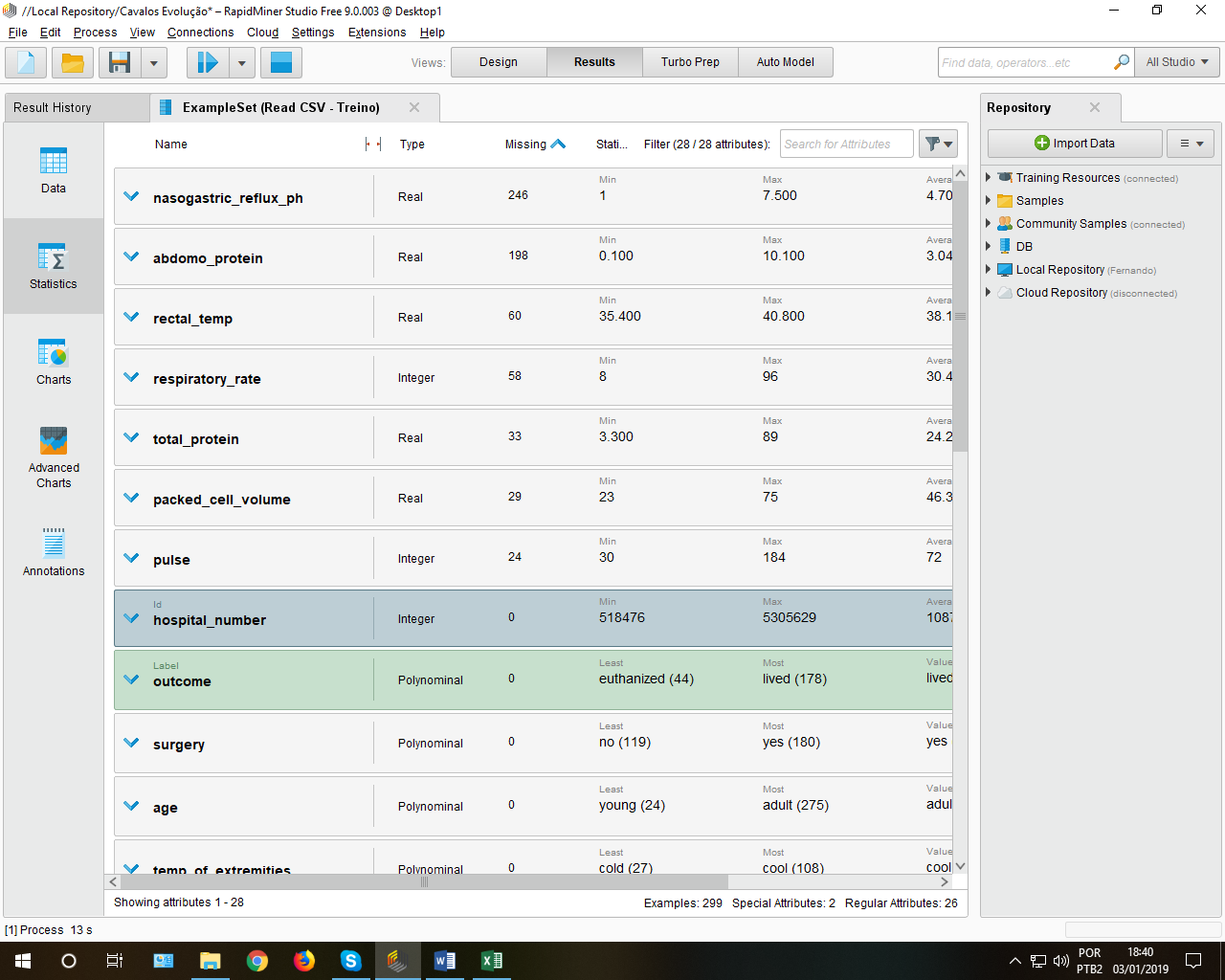
Os demais campos numéricos foram marcados com o tipo Real.

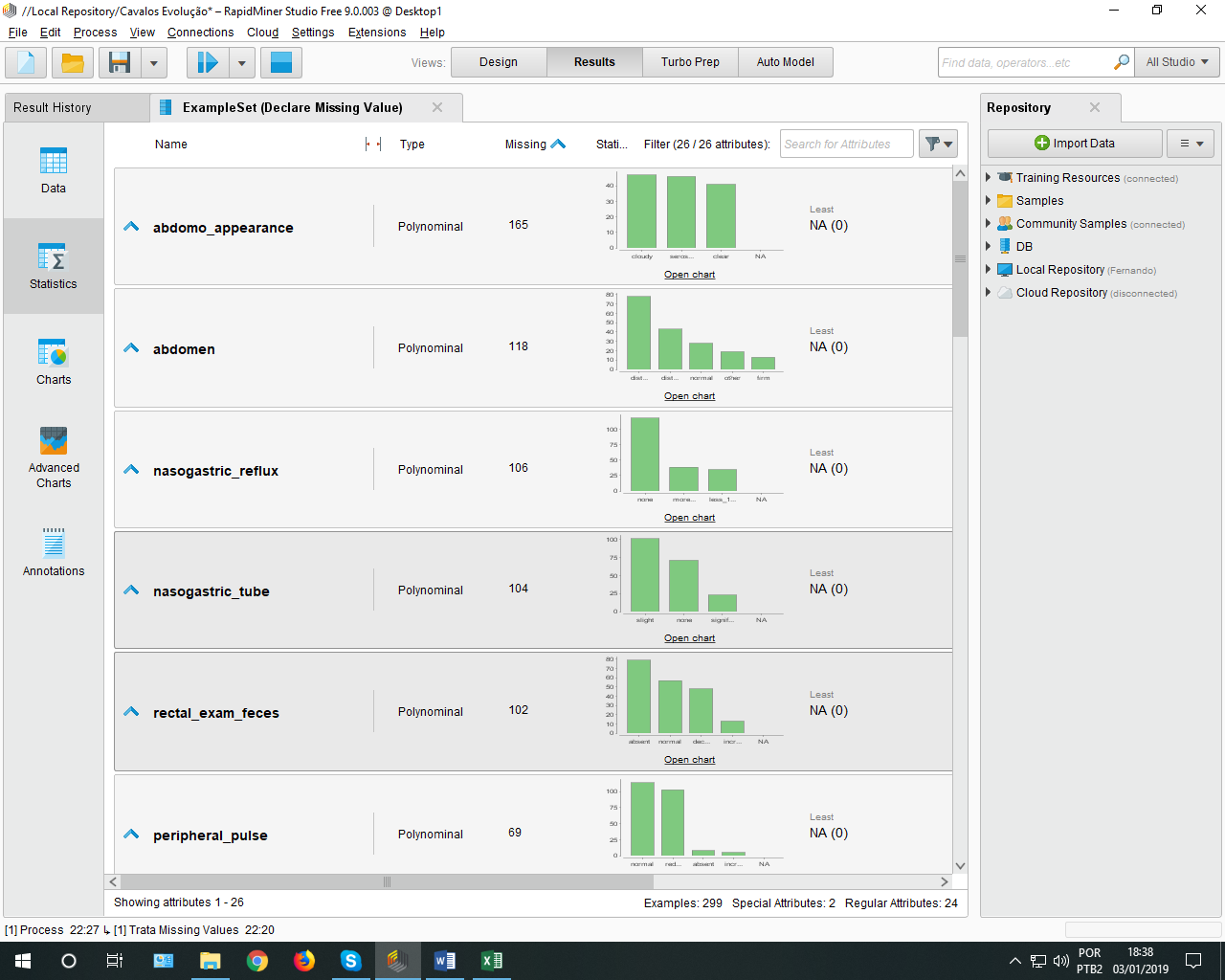
Os campos alfanuméricos foram marcados como Polynomial.

***Missing Values:***

Quanto a Missing Values, identifiquei que estão identificados na base como “NA”. Marquei a opção “read not matching values as missings” na função Read CSV para que os campos preenchidos como “NA” em atributos Numéricos (Real e Integer) já sejam considerados como Missing Values.

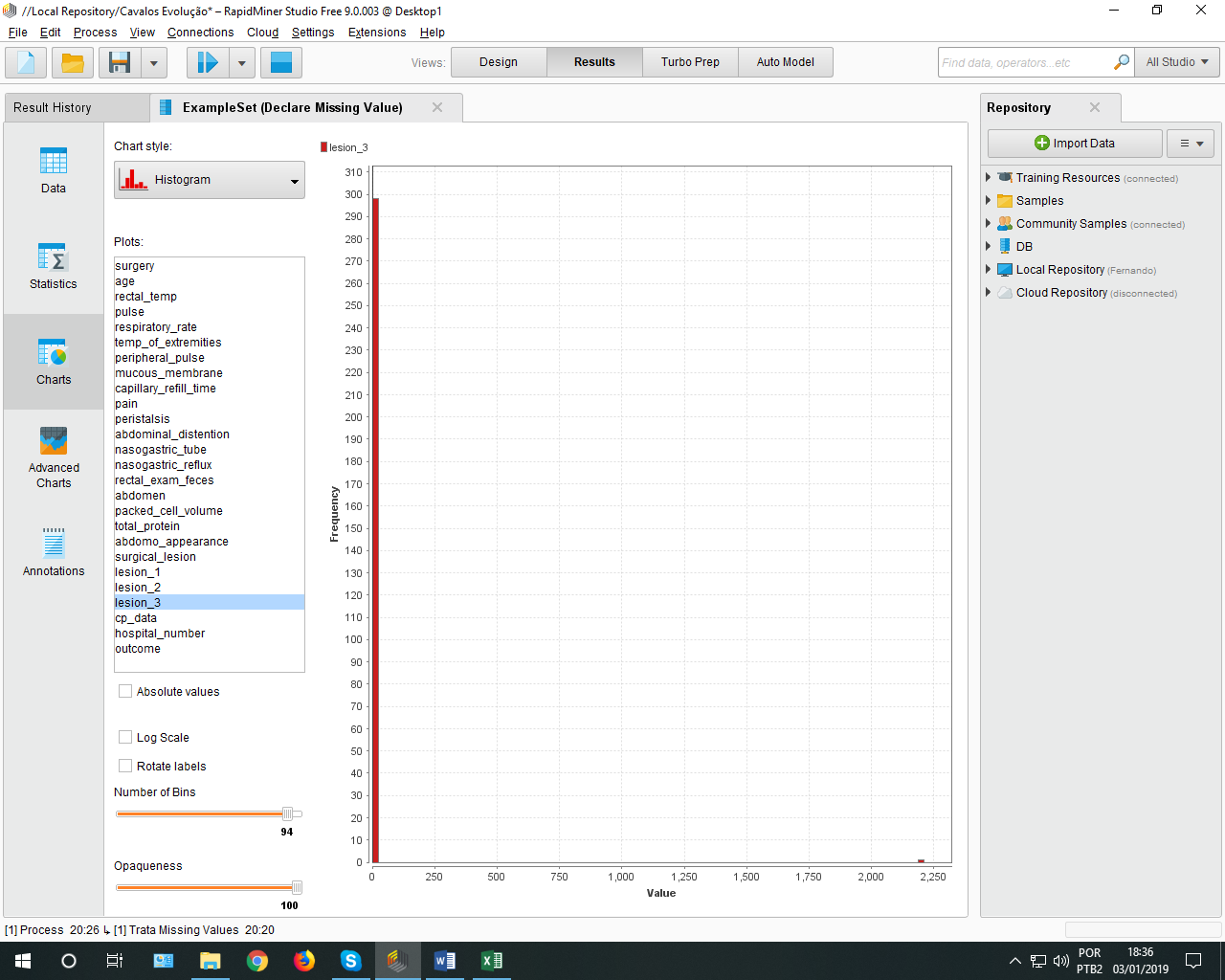
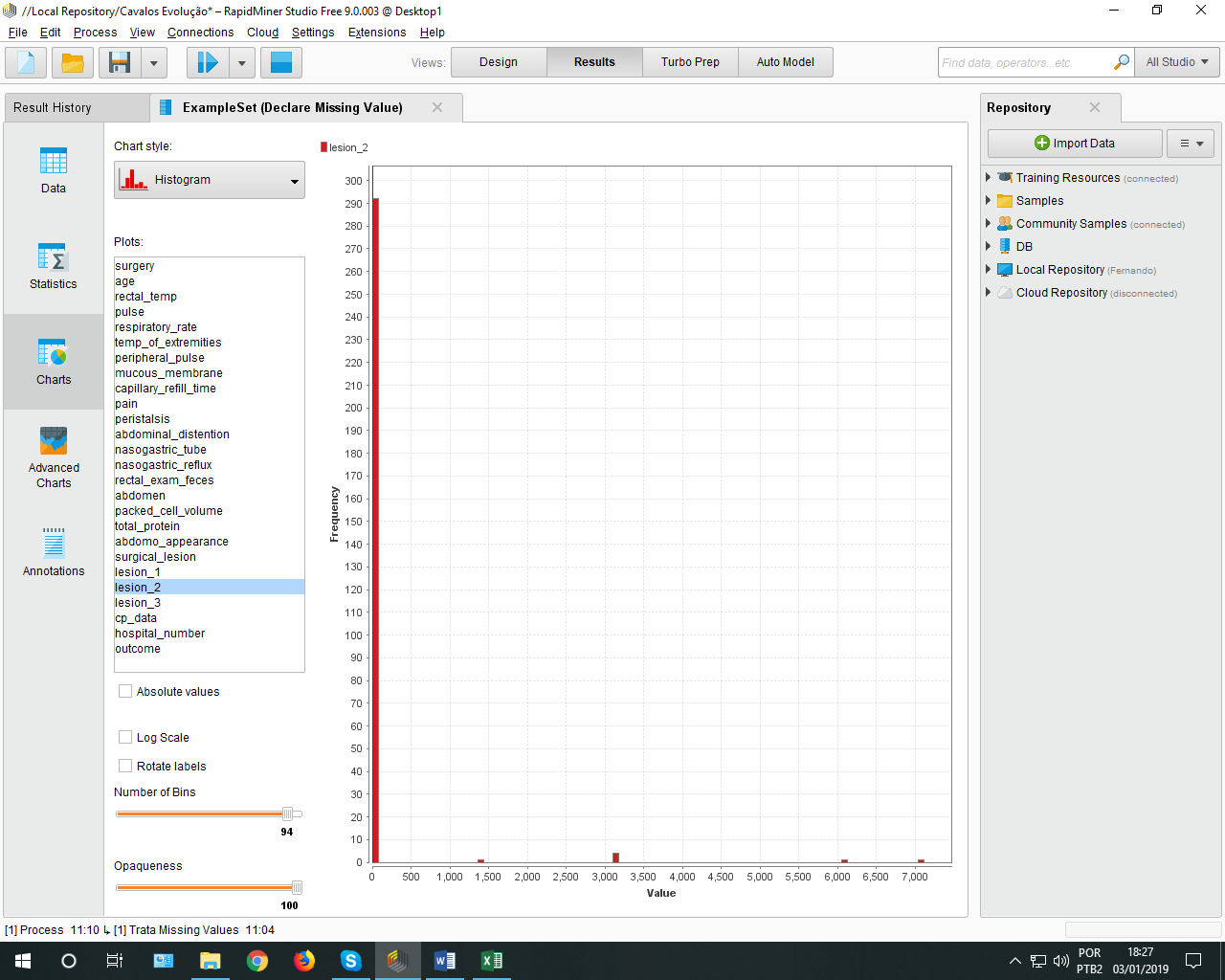
Buscando campos com grande quantidade de Missing Values, identifiquei que os campos numéricos “nasogastric\_reflux\_ph” e “abdomo\_protein” possuem mais de 50% dos valores preenchidos com Missing Values. O mesmo ocorre com o campo alfanumérico “abdomo\_appearance”. Já que mais da metade dos valores desses campos terão que ser estimados em uma etapa de Missing Values, eles são candidatos a serem expurgados da base em uma etapa posterior (Redução de Dimensionalidade) a fim de não impactar a performance e resultado do Modelo.





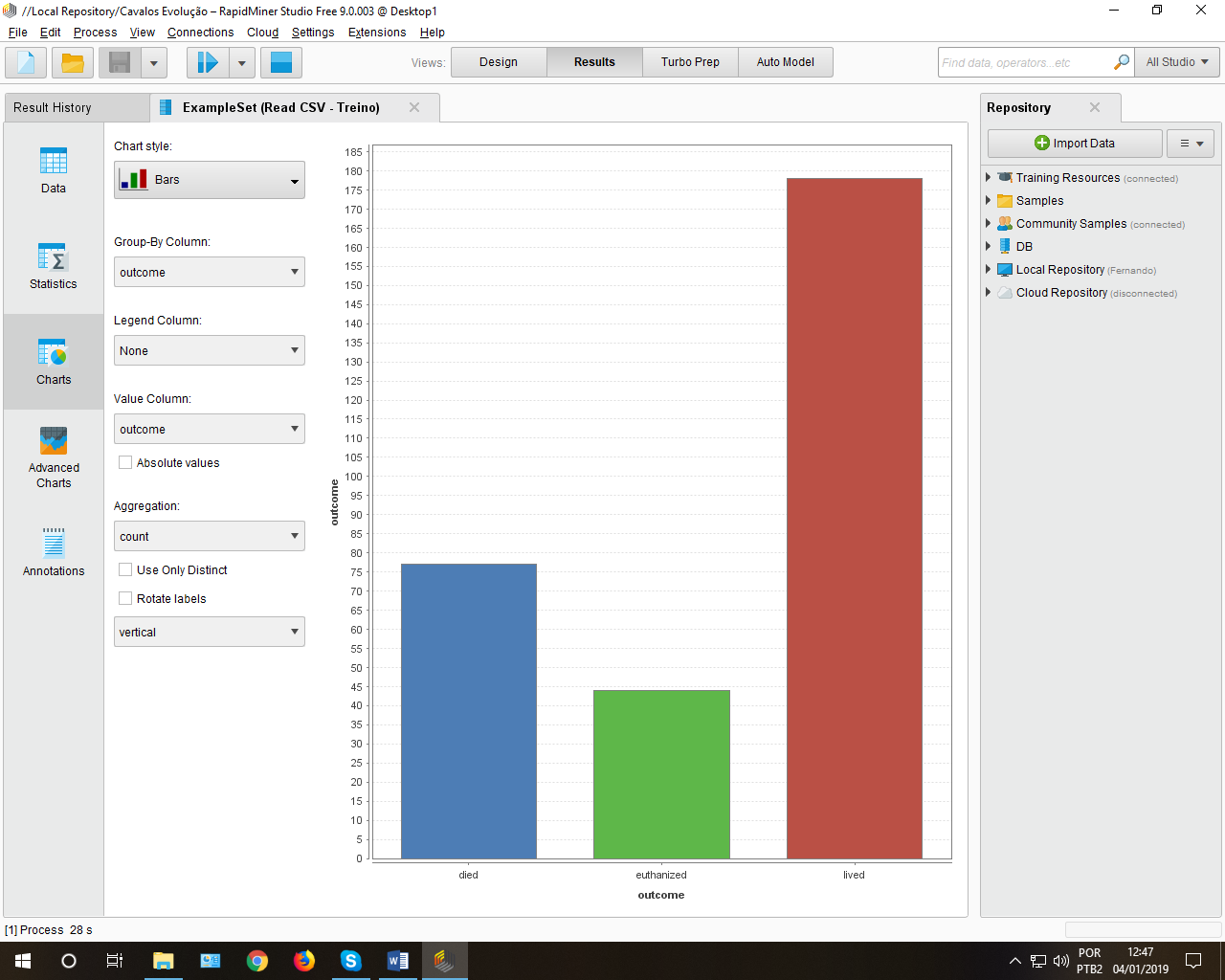
***Dispersão:***

Analisando, também, campos com pouca dispersão, identifiquei os campos lesion\_2 e lesion\_3, os quais estão preenchidos com 0 em mais de 290 registros (cada), em uma base de treino de 299 registros. Ou seja, quase a totalidade dos registros estão preenchidos com 0. São candidatos também a Redução de Dimensionalidade.



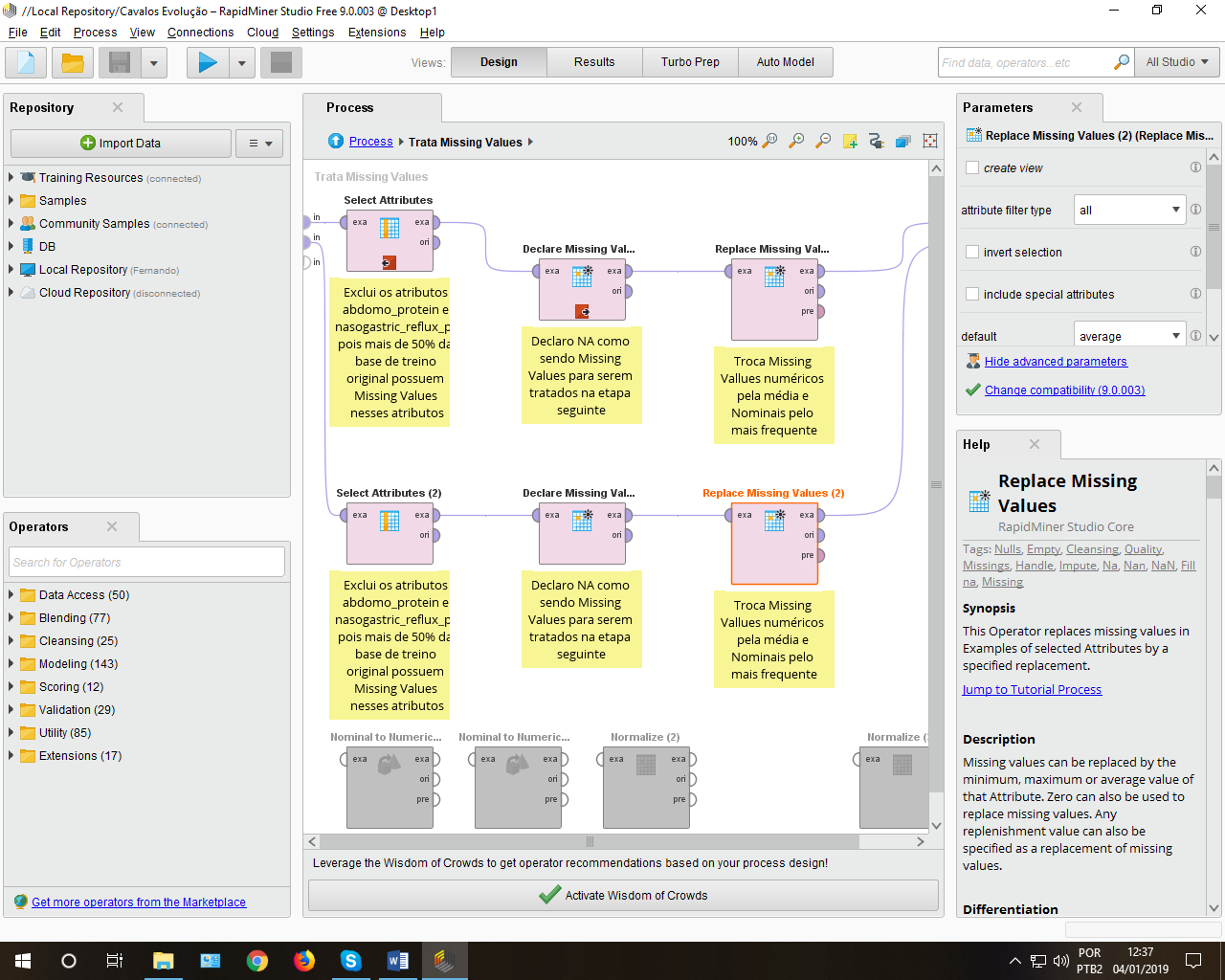
***Balanceamento:***

Analisando o campo Outcome (Label), verifiquei que os dados estão desbalanceados. O total de registros com o resultado Lived é superior à soma dos registros com os outros dois conjuntos de resultados possíveis (Euthanized e Died). Isso indica provável oportunidade de Balanceamento por Undersampling ou Oversampling da amostra.



1. **Tratamento de Missing Values:**

No tratamento de Missing Values utilizei inicialmente a função “Declare Missing Value” para indicar que os campos preenchidos com NA são Missing Values. Em seguida apliquei a função “Replace Missing Values” utilizando o parâmetro Average para que os Missing Values dos atributos numéricos fossem preenchidos com a média e os Missing Values dos atributos Nominais fossem preenchidos com o valor mais frequente.



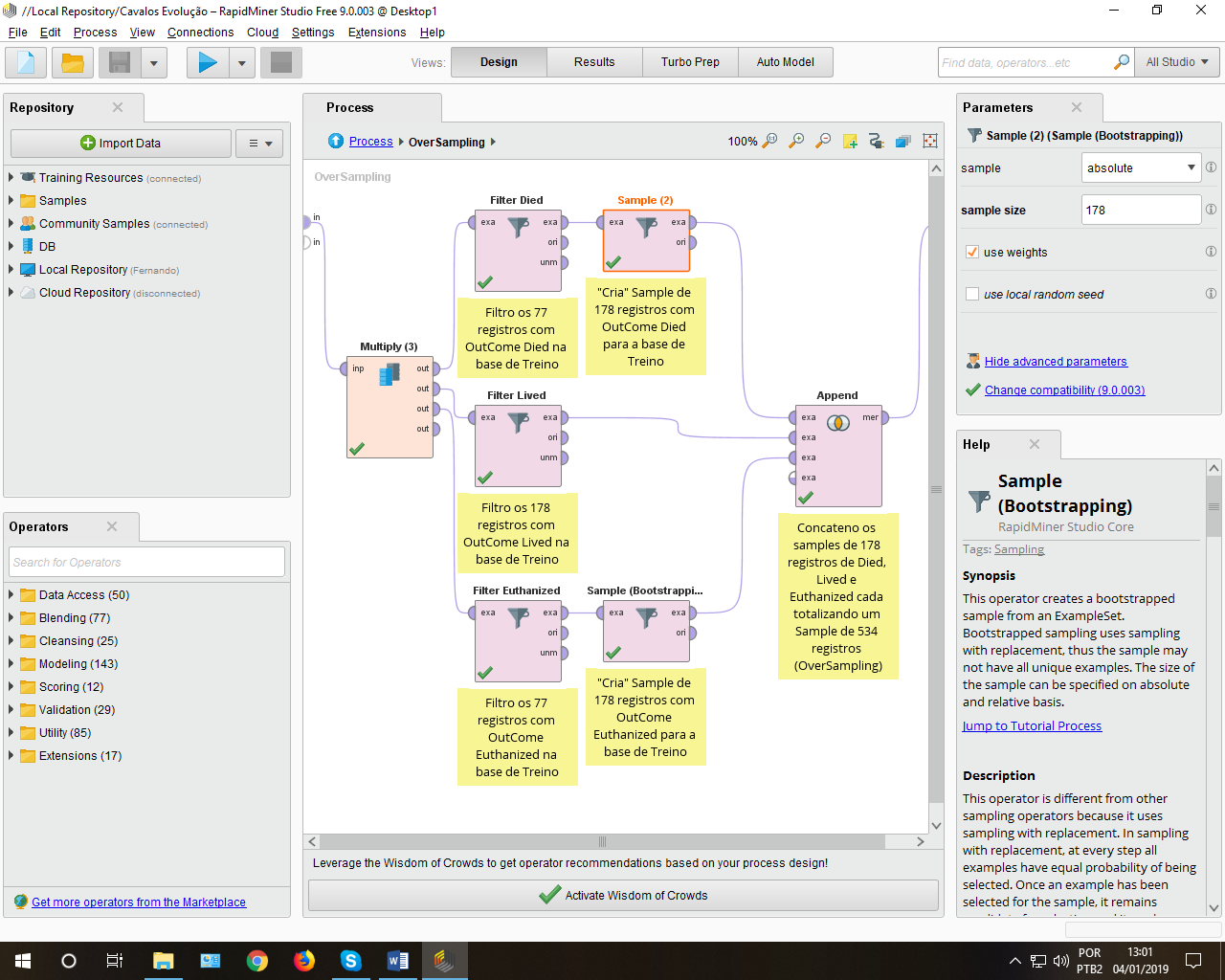
No mesmo subprocesso de Tratamento de Missing Values inclui a função de Select Attibutes posteriormente, para retirar atributos com muitos Missing Values (Redução de Dimensionalidade), como veremos mais adiante nesse documento.

1. **Balanceamento de Dados:**

Nas aulas de DataMining entendi que não haveria operador pronto no RapidMiner para realizar o OverSample. Ao mesmo tempo, acreditei que se aplicasse o UnderSample teria perda significativa de informações na classe predominante (Lived). Assim, me dispus a estudar e buscar desenvolver alternativa de OverSample dentro do RapidMiner.

***OverSample:***

Desenvolvi solução de OverSample no RapidMiner usando as funções ilustradas na figura do processo abaixo:

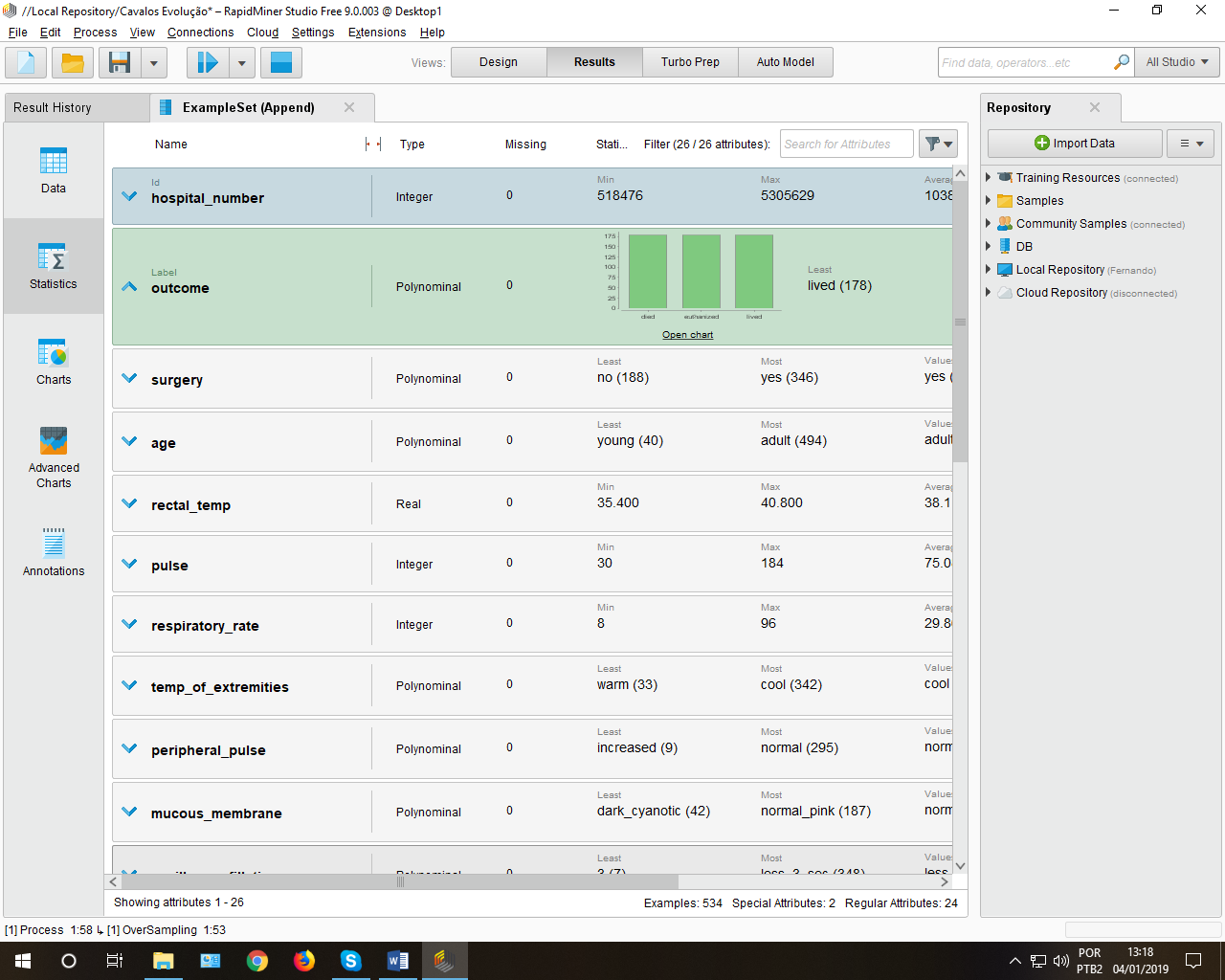


Usei o “Multiply” para replicar a base original de Treino em 3.

Usei a função “Filter” para filtrar separadamente os registros com Outcome Died, Lived e Euthanized a partir da saída da função Multiply.

Usei a função “Sample Bootstrapping” a qual é usada para gerar, a partir de um conjunto exemplo, uma amostra maior em tamanho que o conjunto exemplo original. Fixei o parâmetro de SampleSize em 178 para gerar Samples de 178 registros de Died e 178 registros de Euthanized.

Ao fim do processo, usei a função “Append” para concatenar os 3 Samples gerados, resultando em uma base de Teste de 534 registros (178 x 3).



1. **Teste com Random Forest e variações de Oversample e UnderSample (Balanceamento de Dados):**

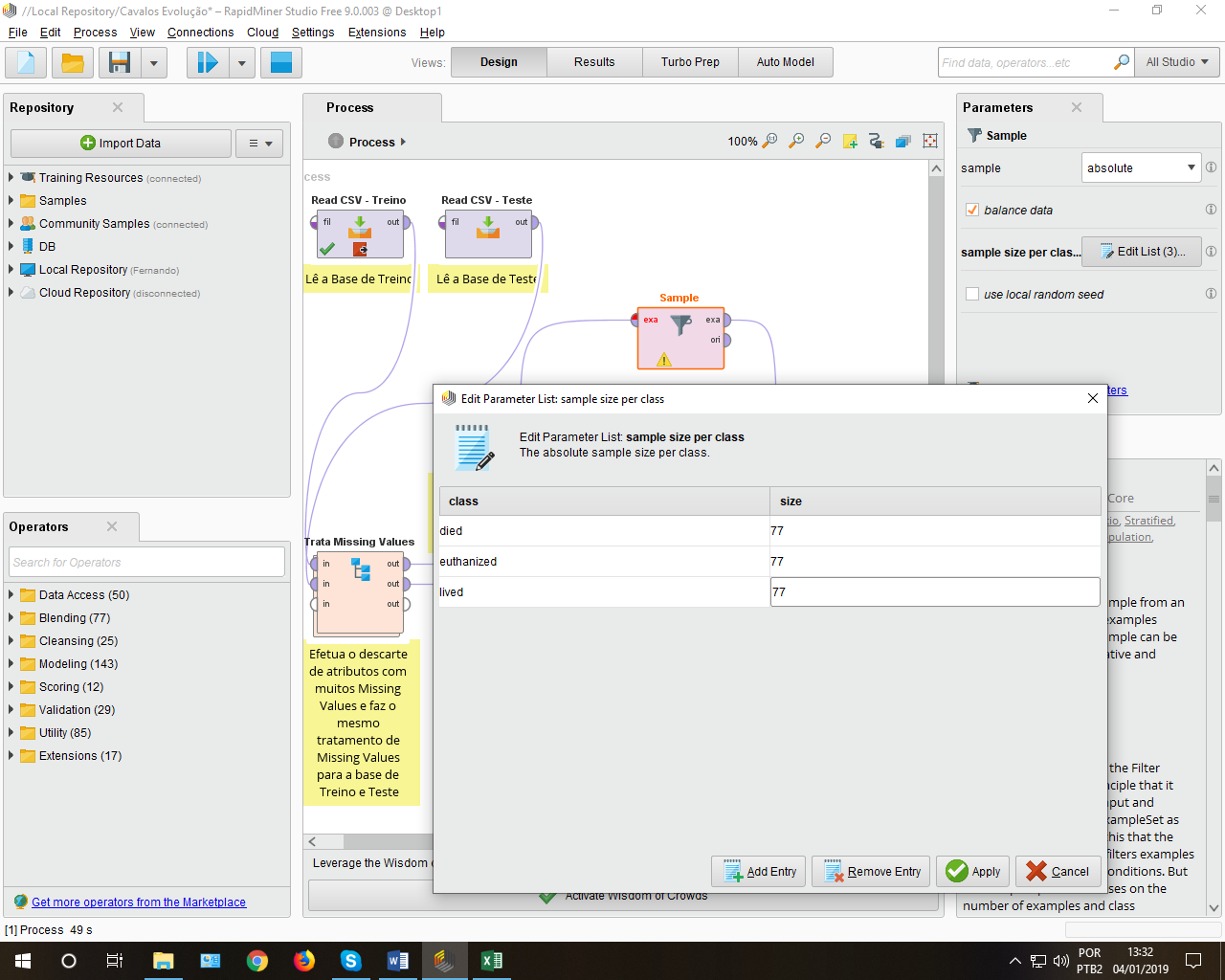
A fim de realizar o primeiro teste com um método e estabelecer a primeira referência de performance, executei o processo aplicando **Tratamento de Missing Values**, **Oversample** e o **Método Random Forest** com parâmetros Default. O resultado encontra-se abaixo:



Optei, em seguida, por fazer avaliações com o mesmo método aplicando o **UnderSample**. Minha intenção era confirmar a visão de que a alternativa de **Oversample** proporcionaria melhores resultados por evitar a perda de informações. Assim, testei modelos com **Undersample** limitando a 44, 77, 100, 178, 120, 130, 140 e 135 registros para cada possível resultado da base de treino (label Outcome). Ou seja:

* Undersample 44 🡪 Geração de Sample com 44 registros Lived, 44 registros Died e 44 registros Euthanized
* Undersample 77 🡪 Geração de Sample com 77 registros Lived, 77 registros Died e 44 registros Euthanized (dado que a base original de treino possui somente 44 registros Euthanized)
* Undersample 100 🡪 Geração de Sample com 100 registros Lived, 77 registros Died e 44 registros Euthanized (dado que a base original de treino possui somente 77 registros Died e 44 registros Euthanized)
* E assim por diante...

Para o UnderSample foi usado o operador **“Sample”** conforme parametrização ilustrada abaixo para o caso de UnderSample – 77:



Os resultados obtidos com essas simulações encontram-se abaixo:

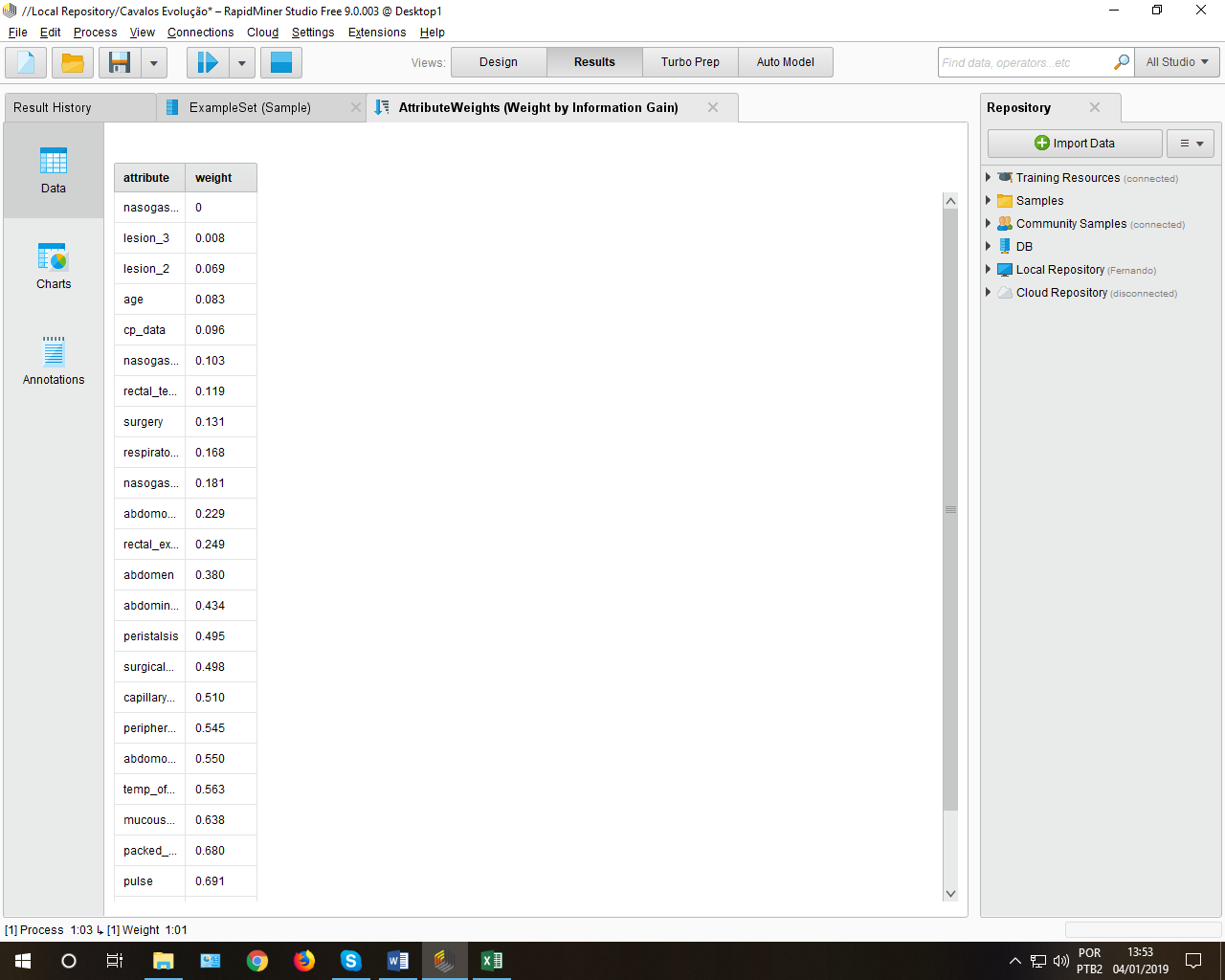


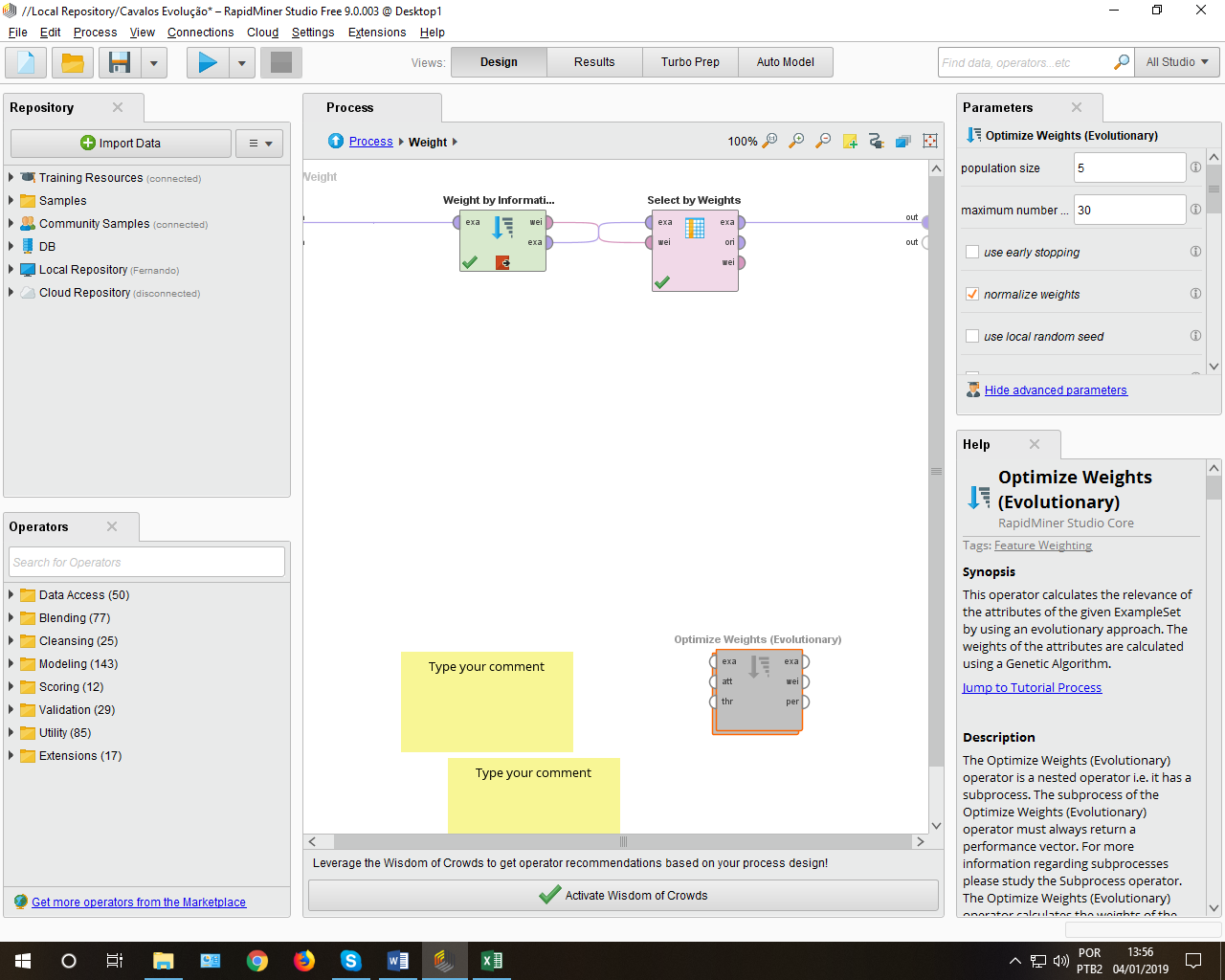
Como podemos observar os resultados foram interessantes pois consegui elevar a Acurácia de 86,52% para 96,63% e o Kappa de 0,772 para 0,939. Por outro lado, foi decepcionante e revelador observar que consigo resultados melhores com um Sample que mantém a quantidade original de registros de Died e Euthanized ao mesmo tempo que limita a 135 a quantidade de registros de Lived (**Undersample - 135**).

1. **Avaliações da Redução de Dimensionalidade:**

Tomando como base o método com melhor resultado até aquele momento (Missing Values / UnderSample (135) / Random Forest), apliquei estratégias de redução de dimensionalidade a fim de avaliar se haveria melhora nos resultados.

Apliquei, inicialmente, operadores para realizar a seleção por **Weight by Information Gain**.



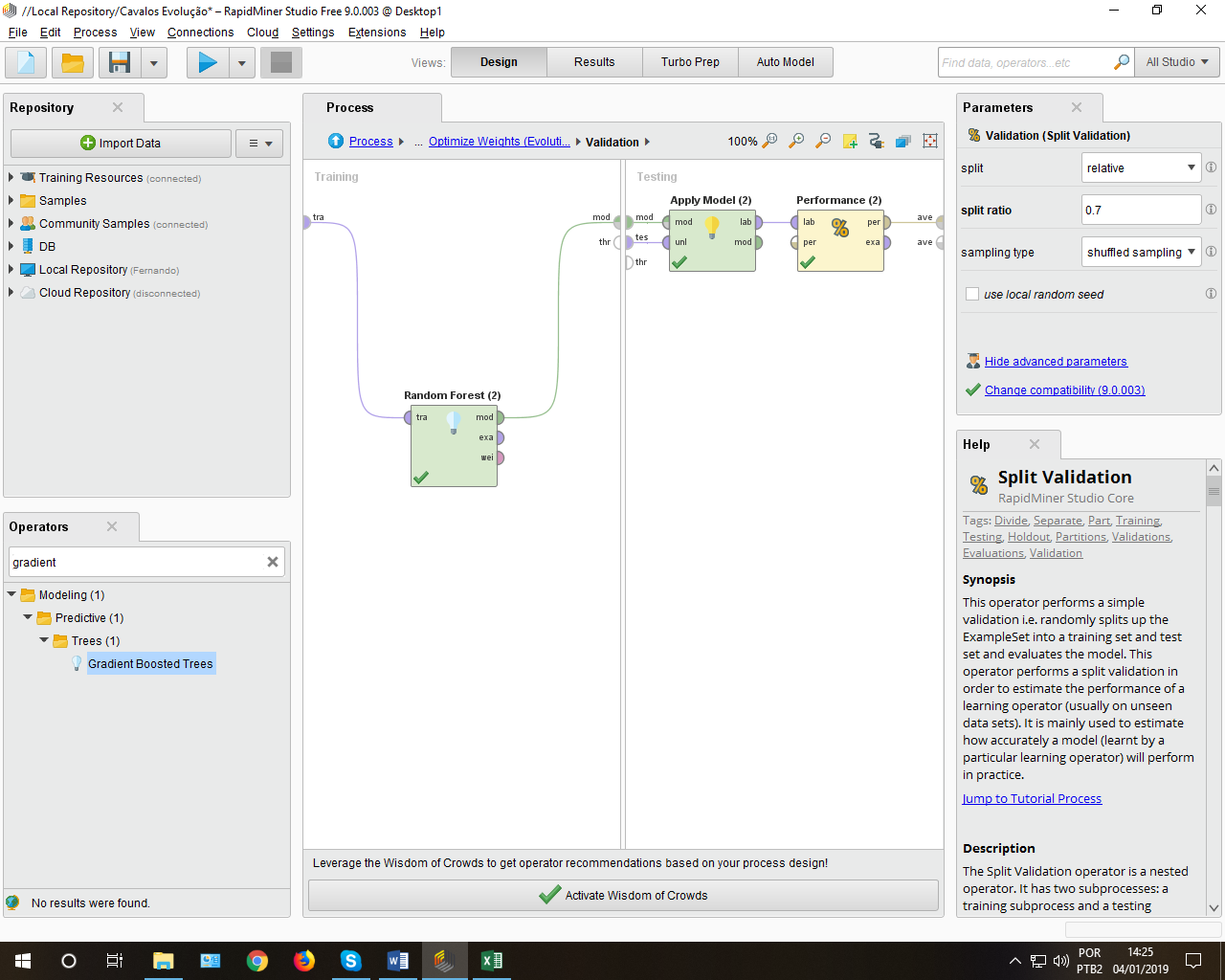


Variando o valor de corte no operador **“Select by Weights”** agreguei mais os seguintes resultados à análise:



Como podemos observar, a operação de **“Weight by Information Gain”** proporcionou piora no resultado do método.

Dando continuidade a avaliação de redução de dimensionalidade, substituí o operador **“Weight by Information Gain”** pelo operador **“Optimize Weights (Evolutionary)”** aplicando o método de Random Forest, conforme ilustração abaixo.

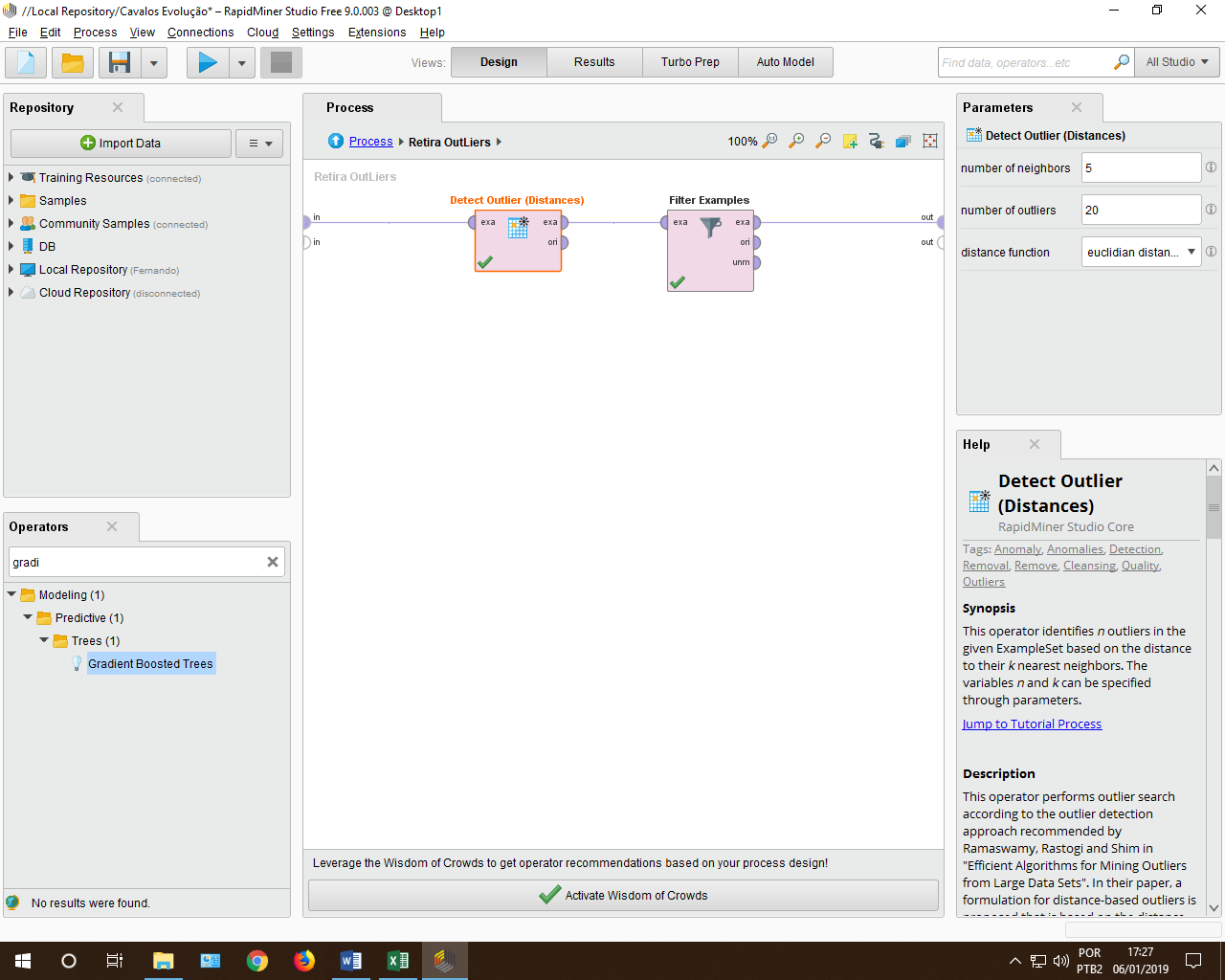


O resultado obtido, agregado abaixo à tabela de experimentos, foi muito ruim:



1. **Retirada de Outliers:**

Partindo, mais uma vez, do método com melhor resultado até o momento (Missing Values / UnderSample (135) / Random Forest), apliquei Operador de “Detect Outlier”, variando os parâmetros desse operador, a fim de avaliar se haveria melhora nos resultados.



Os resultados obtidos com a aplicação da retirada de Outliers, em algumas parametrizações, se igualaram aos melhores resultados obtidos anteriormente, mas não superaram os resultados anteriores, conforme podemos ver abaixo:



1. **Experimentando a Retirada de Outliers e Redução de Dimensionalidade no mesmo modelo:**

Optei por voltar a experimentar o operador **“Weight by Information Gain”,** mas dessa vez em conjunto com a retirada de Outliers a fim de avaliar se essa composição poderia trazer ganhos aos resultados do Modelo. Mais uma vez, obtive resultados que igualaram, mas não superaram os melhores resultados obtidos anteriormente, conforme pode ser verificado abaixo:



1. **Experimentando outras estratégias de Redução de Dimensionalidade:**

* **Atributos com muitos Missing Values**

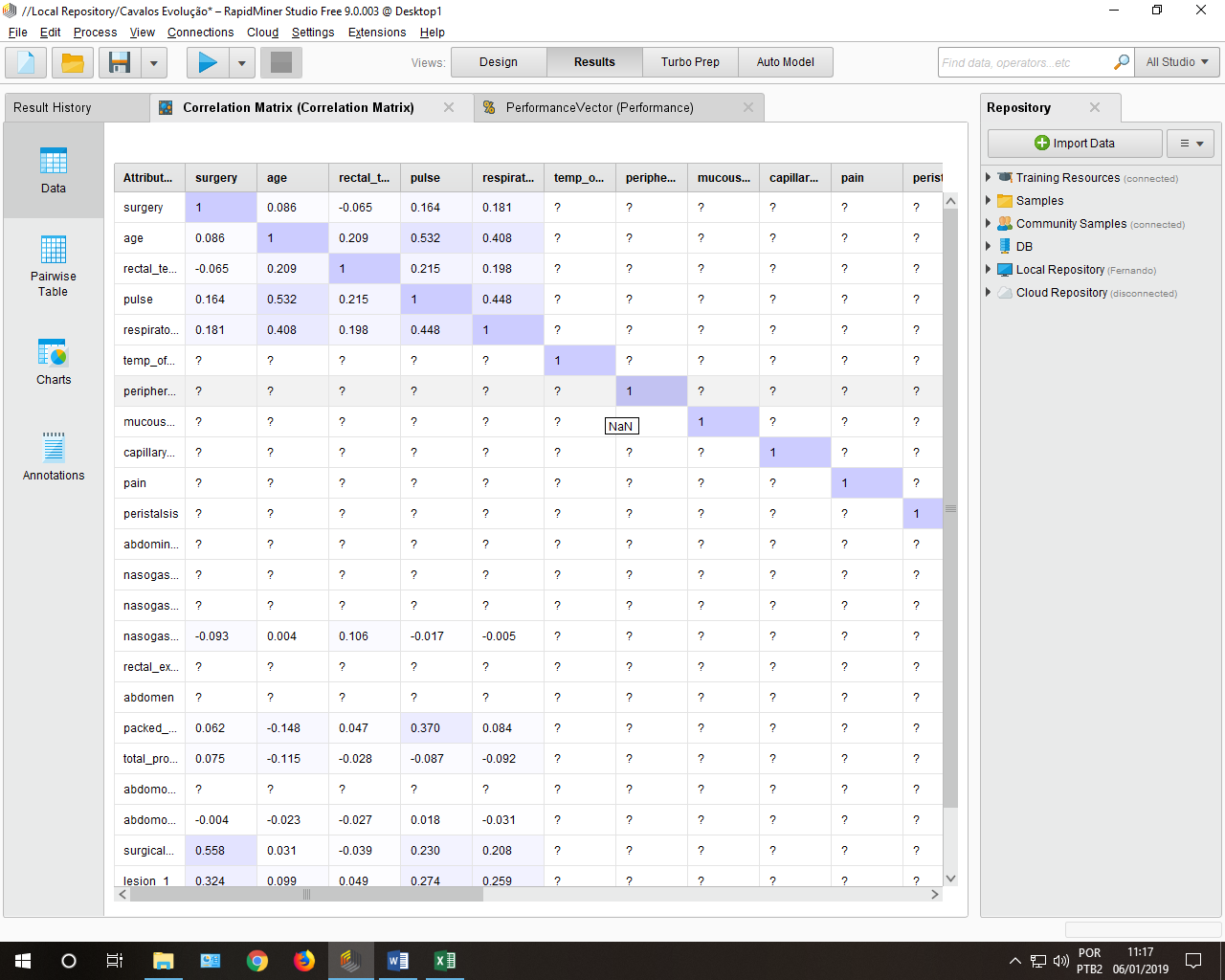
Conforme identificado na análise exploratória de dados, identifiquei que os campos numéricos “nasogastric\_reflux\_ph” e “abdomo\_protein” tem mais de 50% dos valores preenchidos com Missing Values. Já que mais da metade dos valores desses campos estão sendo estimados na etapa de Missing Values, eles são candidatos a serem expurgados (Redução de Dimensionalidade) a fim de não impactar a performance e resultado do Modelo.

* **Remove Useless Atributes**

Trata-se da aplicação do operador **Remove Useless Atributes** do RapidMiner apresentado nas aulas

* **Remove Correlated Atributes**

O objetivo é excluir atributos com forte correlação. Após montagem da matriz de correlação, verifiquei que não há atributos com correlação elevada. Conforme figura abaixo, a maior correlação existente entre atributos é 0,558. Ainda assim, experimentei remover os Correlated Atributes com correlação superior a 0,5.



Os resultados desses experimentos encontram-se abaixo. A aplicação do Operador **Remove Useless Atributes** foi o único que proporcionou desempenho equivalente aos melhores resultados anteriores. Os demais operadores proporcionaram piora.



1. **Experimentando outros Métodos (Não cabe ter um “método favorito”):**

Considerei como esgotadas as possibilidades de melhoria do resultado aplicando o método de Random Forest e passei a testar outros métodos. Foram testados os seguintes métodos além do Random Forest:

* + KNN
  + Árvore de Decisão
  + Gradient Boosted Tree

Esses métodos foram testados em cima da combinação de processos com os melhores resultados obtidos até aquele momento, ou seja, foram testados em conjunto com a aplicação de:

* + Tratamento de Missing Values
  + UnderSample (135)
  + Weight by Information Gain (corte 0,2)
  + Retira Outliers (Vizinhos = 10, Outliers = 15, Euclidiana).

Os resultados obtidos a partir da aplicação desses métodos, de acordo com a parametrização efetuada, encontram-se abaixo:



Como podemos observar, o KNN proporcionou melhor resultado quando K= 1 ou 2. Ainda assim, o resultado (Acurácia 94,38%) foi abaixo do obtido com o Random Forest.

Já o resultado obtido com a Árvore de Decisão (Acurácia 79,78%), na primeira execução com valores Default, foi muito abaixo do melhor resultado obtido com o Random Forest.

Por último, a aplicação do método Gradient Boosted Trees, apresentou vários resultados que igualaram o melhor resultado obtido com o Random Forest. Por isso, optei por me aprofundar nas simulações e ajustes finos em cima desse modelo.

1. **Método Gradient Boosted Tree – Ajuste fino:**

Partindo do modelo em que apliquei o método **Gradient Boosted Tree** com o melhor resultado, busquei experimentar operadores que não proporcionaram melhoria no cenário com **Random Forest** a fim de avaliar se trazem ganho ao método **Gradient Boosted Tree**.

* **Atributos com muitos Missing Values**

Conforme identificado na análise exploratória de dados, identifiquei que os campos numéricos “nasogastric\_reflux\_ph” e “abdomo\_protein” tem mais de 50% dos valores preenchidos com Missing Values. Experimentei a retirada desses atributos. A performance do resultado se manteve inalterada.



* **Oversample**

Experimentei usar a estratégia de Oversample, ou seja, um Sample de treino com 178 registros Died, 178 registros Euthanized e 178 registros Lived. Usei essa estratégia em conjunto com a retirada de atributos com muitos Missing Values. Como resultado obtive a melhor performance de todos os cenários até então (Acurácia de 97,75% e Kappa de 0,96).



Experimentei não aplicar a Retirada de Atributos com muitos Missing Values e verifiquei que a performance obtida não se mantém (Acurácia de 95,51% e Kappa de 0,921). Ou seja, para obtermos resultados satisfatórios na aplicação do Oversample nesse caso, é necessário aplicar também a Retirada de Atributos com muitos Missing Values.

* **Experimento com a parametrização da Retirada de Outliers**

Alterei o parâmetro Number of Outliers do operador Detect Outlier para 20. Não obtive melhora no resultado.

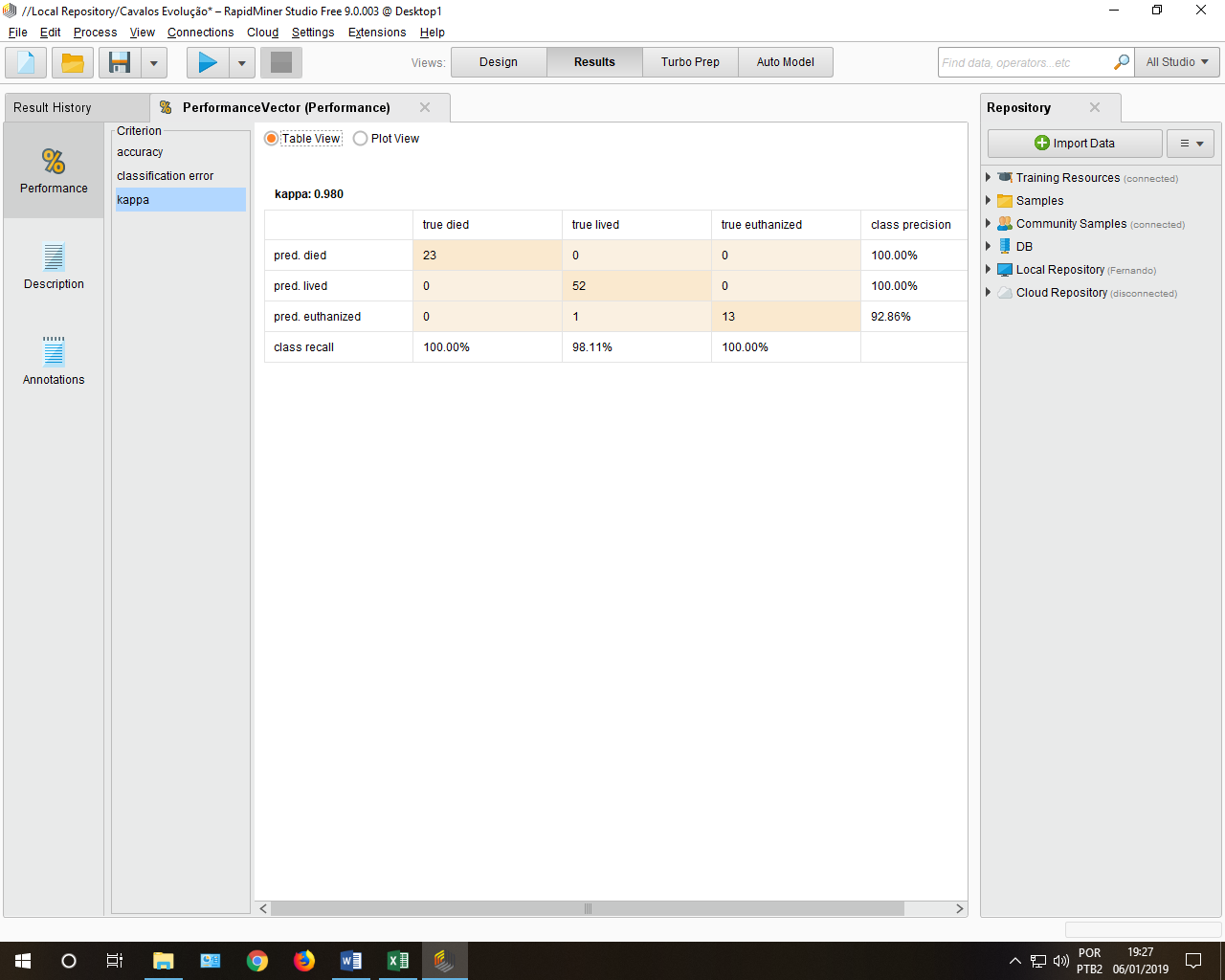


* **Experimento com a parametrização do Método Gradient Boosted Trees**

Alterei o parâmetro Number of Trees do operador Gradient Boosted Trees para 40. Obtive o melhor resultado de todos os experimentos realizados nesse trabalho (Acurácia de 98,88% e Kappa de 0,98).



Somente 1 registro da base de testes foi classificado de forma errada pelo método:



* **Experimento com a parametrização da Retirada de Outliers – Nova Tentativa**

Após obter o melhor resultado com o Operador Gradient Boosted Trees parametrizado com 40 em Number of Trees, fiz novas tentativas de ajuste fino no parâmetro Number of Outliers do Operador Detect Outlier. Os resultados encontram-se abaixo. Nenhum resultado superou o melhor obtido anteriormente.



* **Última chance para o operador Optimize Weight (Evolutionary)**

Busquei usar o “Optimize Weight (Evolutionary)” em lugar do “Weigth by Information Gain” no experimento com melhor performance obtida até o momento. Executei o teste usando o método Gradient Boosted Trees com parâmetros default dentro do “Optimize Weight (Evolutionary)”. Não obtive bons resultados.



* **Última chance para o Método KNN**

Experimentei usar o “KNN” em lugar do “Gradient Boosted Trees” no fluxo em que obtive a melhor performance de todos os experimentos, ou seja, “Trata Missing Values / Retira Atributos com muitos Missing Values / OverSample / Weight by Information Gain (corte 0,2) / Retira Outliers (Vizinhos = 10, Outliers = 20, Euclidiana)”. Não obtive bons resultados.



* **Última chance para o Método Árvore de Decisão**

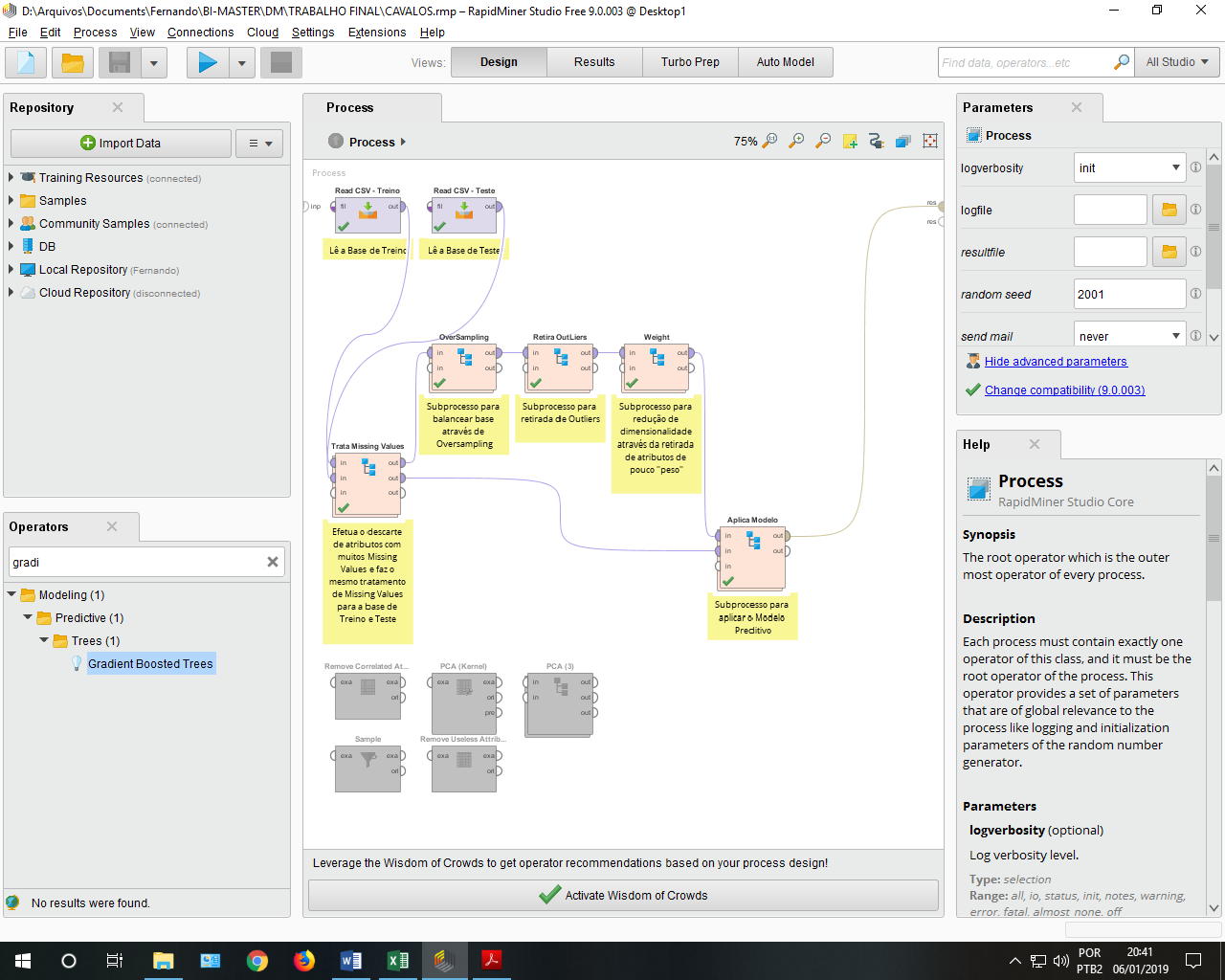
Experimentei usar a “Árvore de Decisão” em lugar do “Gradient Boosted Trees” no fluxo em que obtive a melhor performance de todos os experimentos, ou seja, “Trata Missing Values / Retira Atributos com muitos Missing Values / OverSample / Weight by Information Gain (corte 0,2) / Retira Outliers (Vizinhos = 10, Outliers = 20, Euclidiana)”. Não obtive bons resultados.



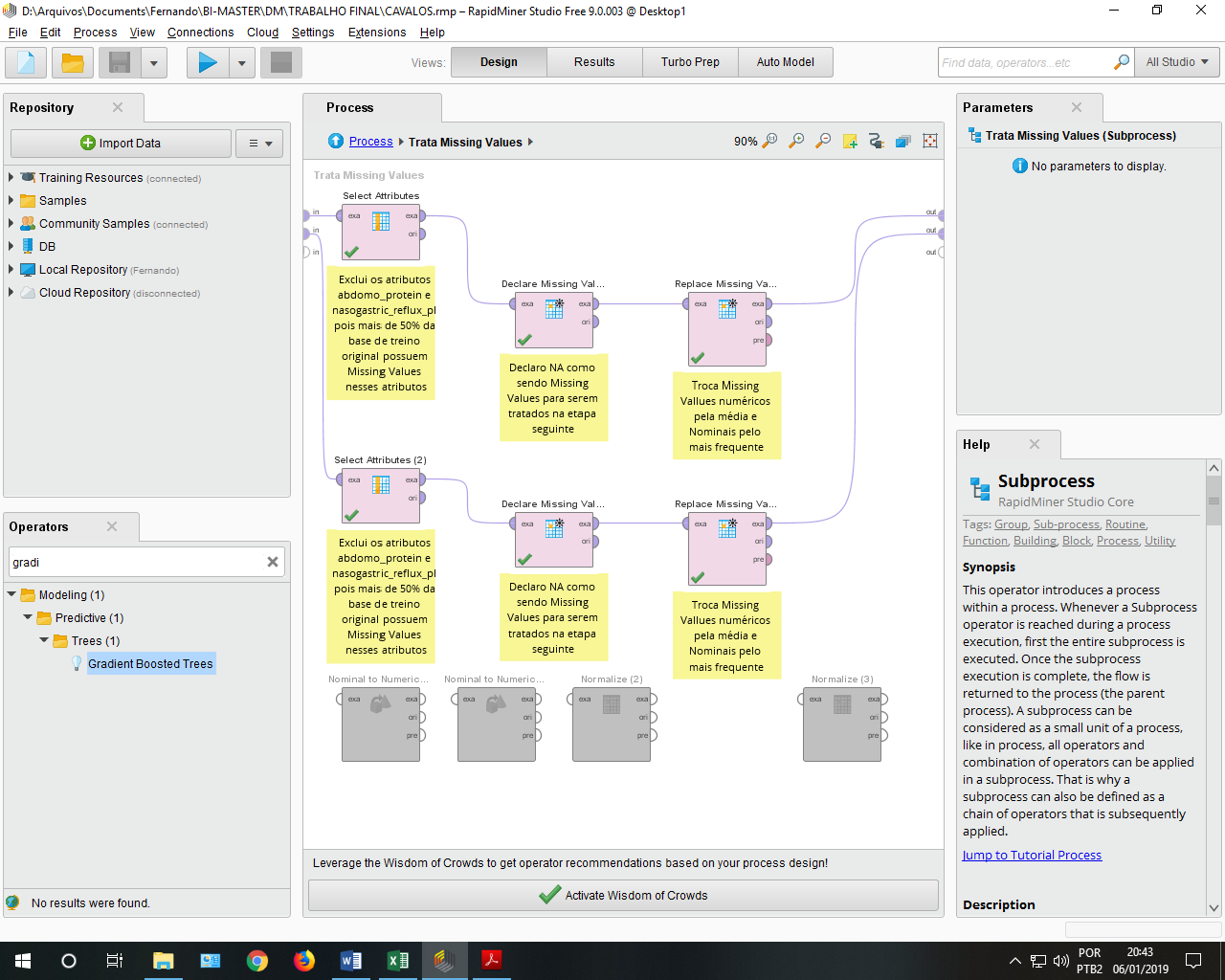
**ANEXO 1 - Relação completa dos experimentos realizados e respectivos resultados**

**ANEXO 2 - Processos implementados no RapidMiner**

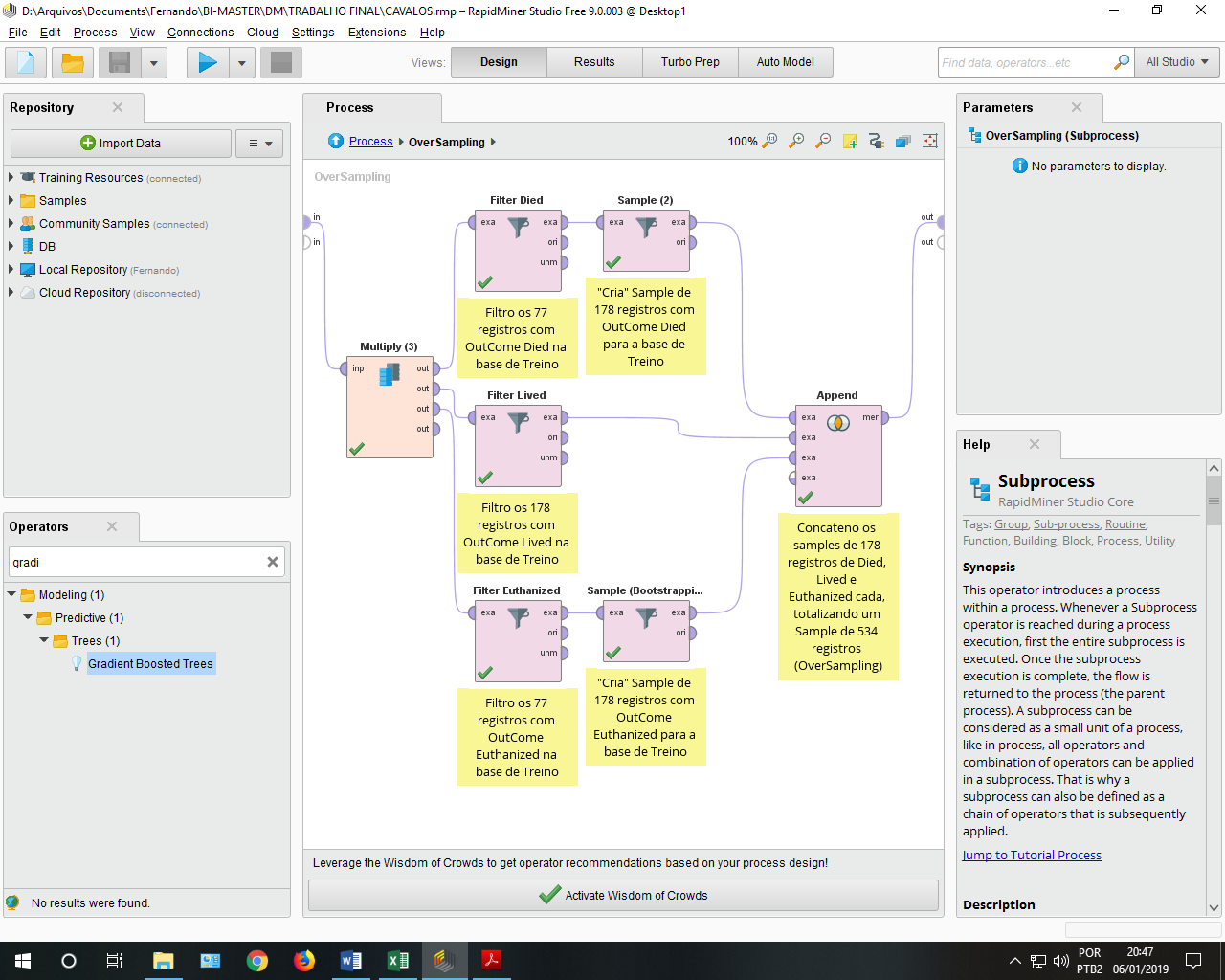
**Processo Principal:**



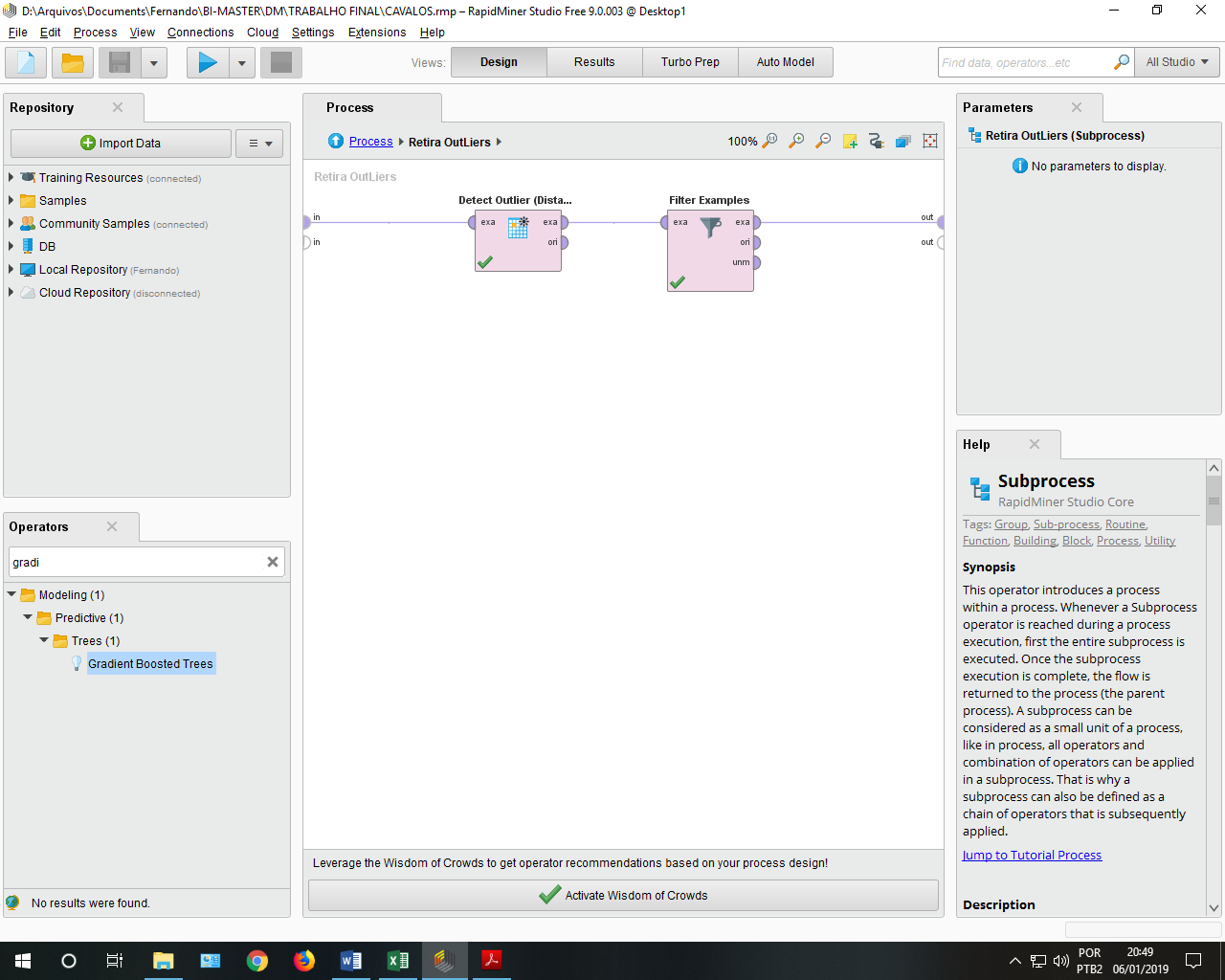
**Trata Missing Values:**



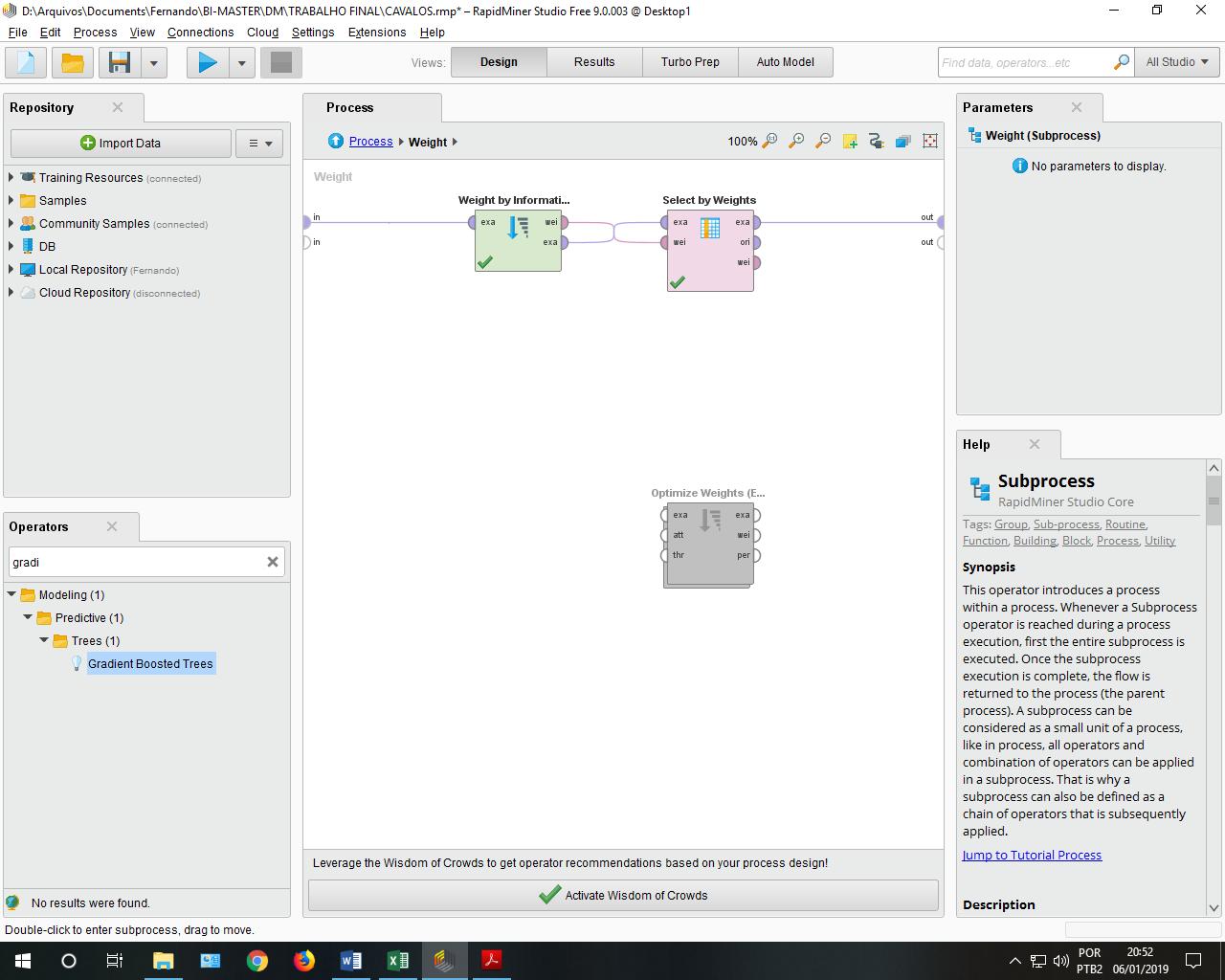
**Oversampling:**



**Retira Outliers:**



**Seleciona por Peso:**



**Aplica Modelo Preditivo:**

