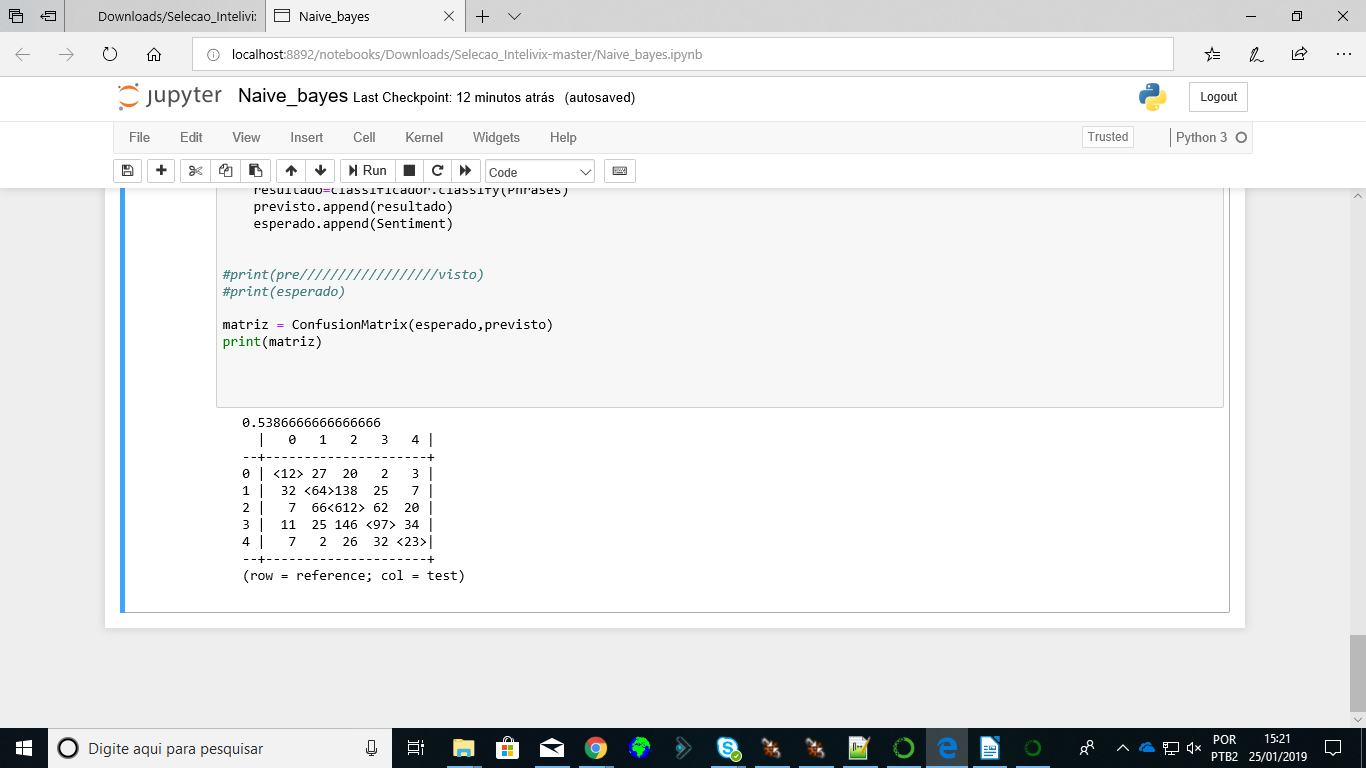
**Relatório**:

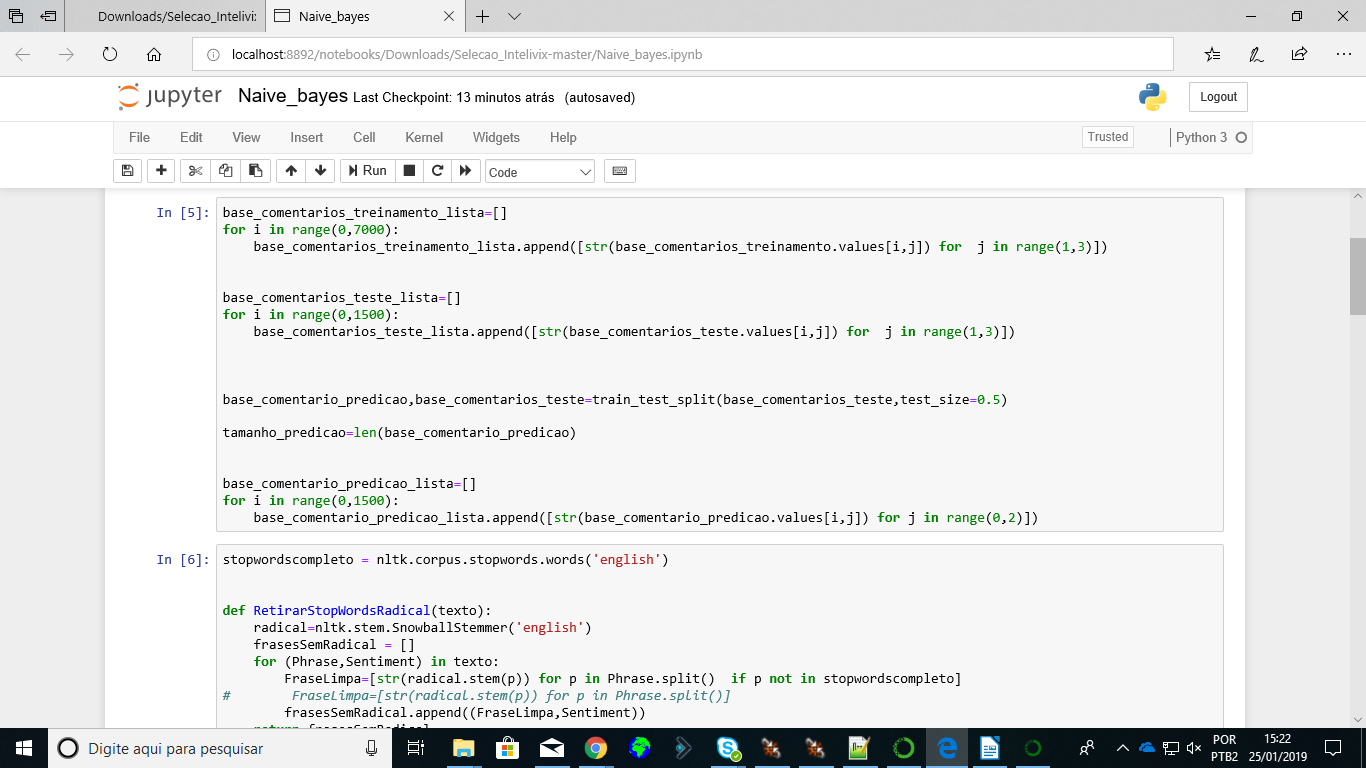
1° algorítimo - Naive Bayes

Acurácia e Matriz de confusão

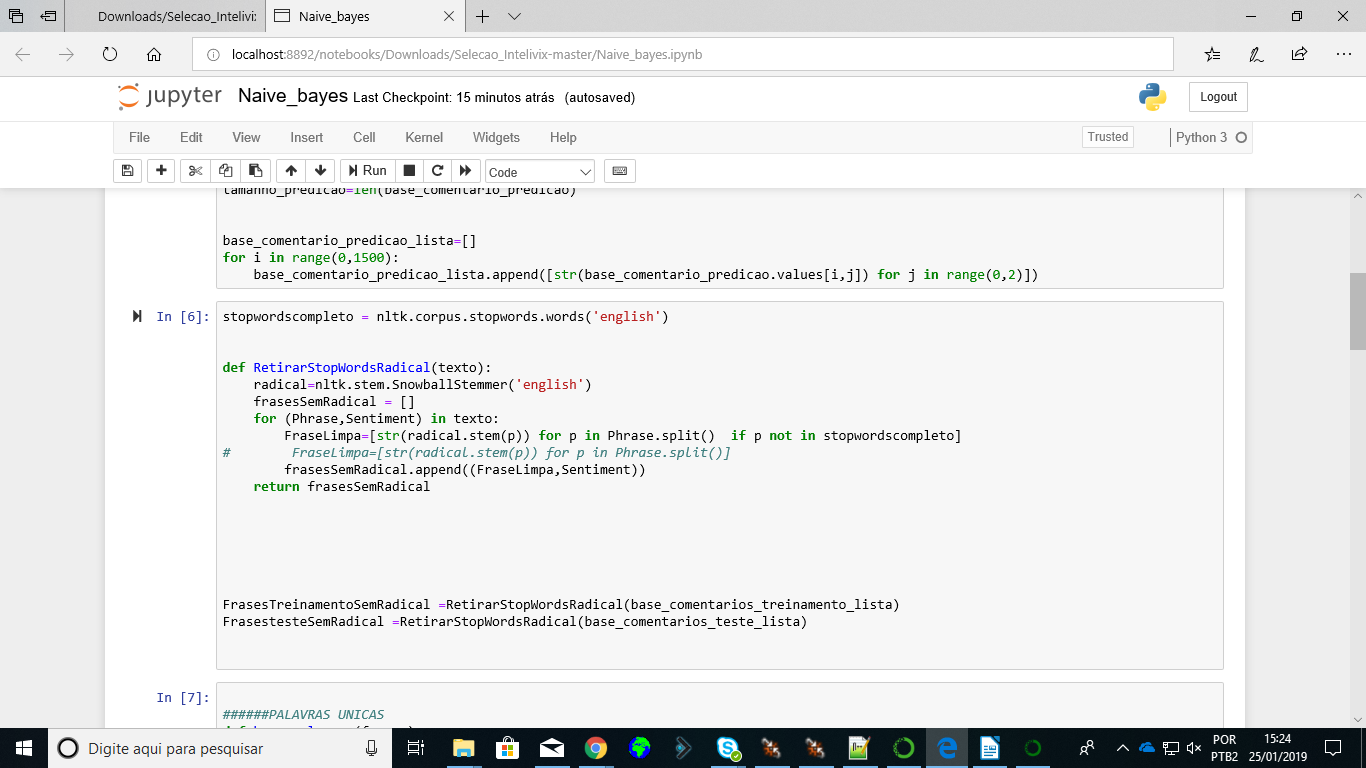


**Acurácia 53,86% (este obteve o melhor resultado) – é o algorítimo mais usado em análise de sentimentos.**

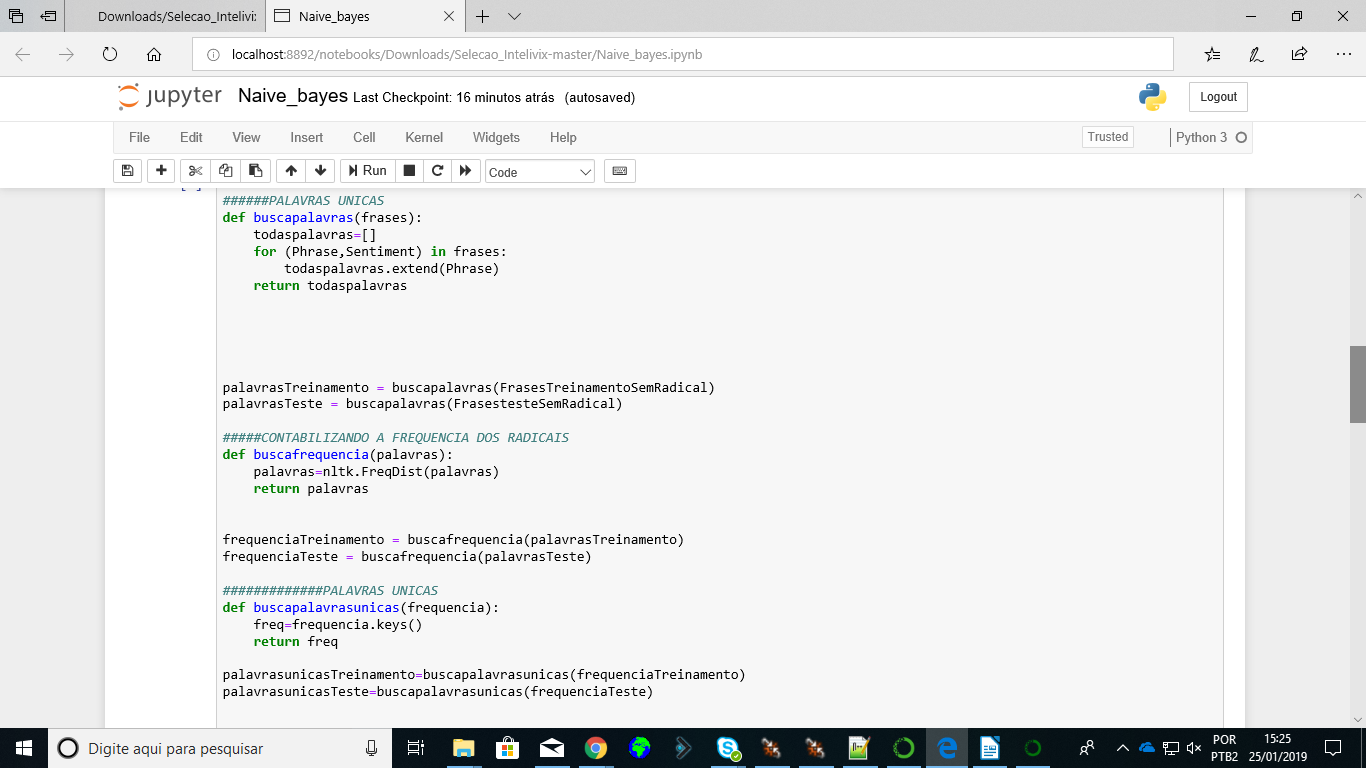
1 - Separamos 70% da base para treinamento e 15% para base de teste e 15 % para base de predição. (para poder conseguir executar (máquina com 4Gb de RAM), tiver que reduzir o tamanho da base.

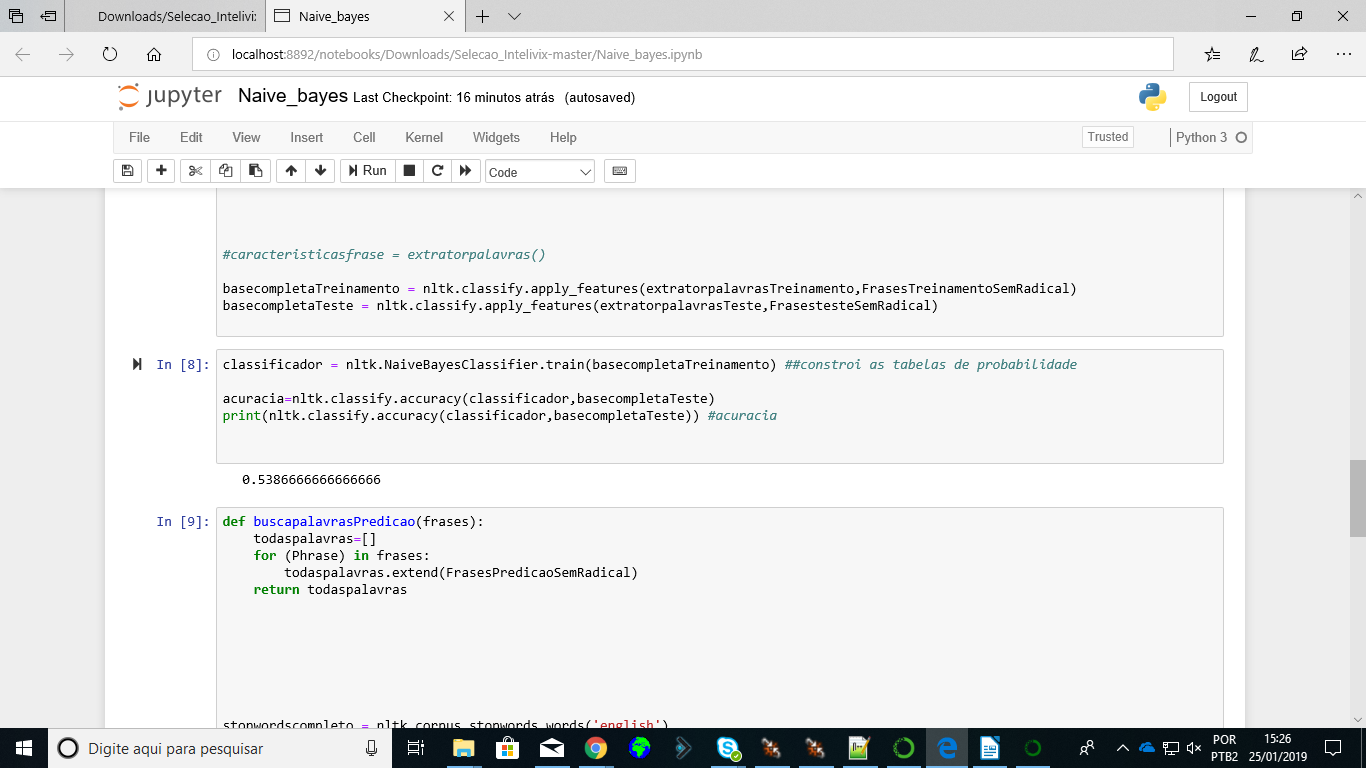


2 - Realizamos o tratamento das palavras que não vão influenciar na análise (stopwords)

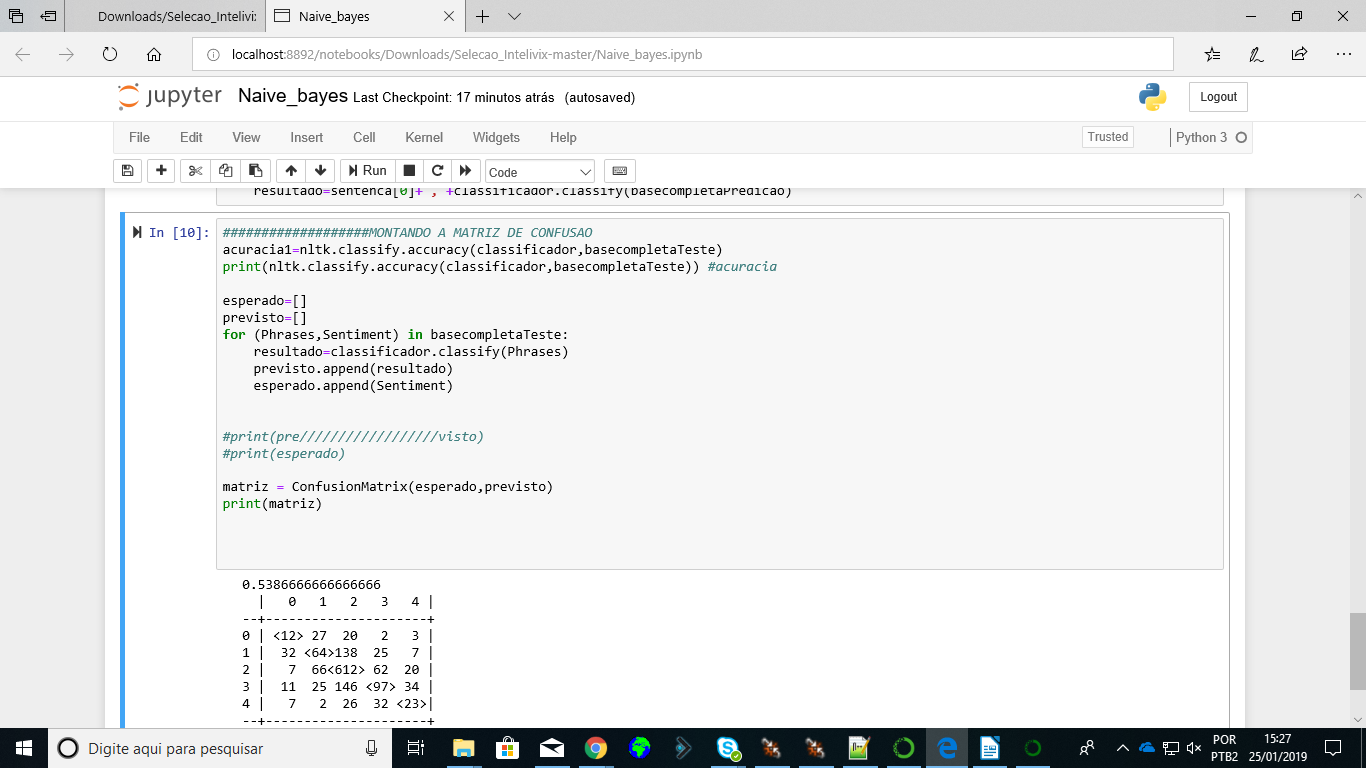
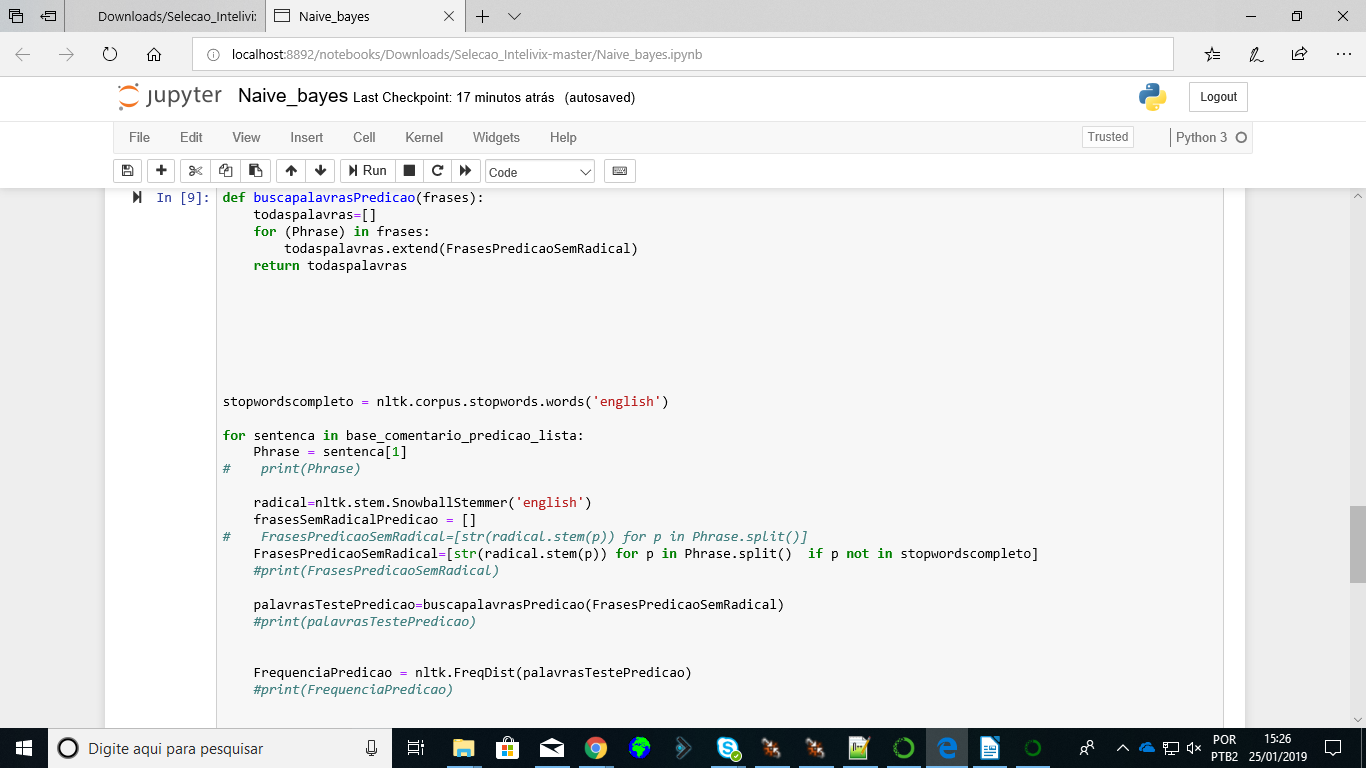


3 – Precisamos retirar as palavras repetidas dentro da mesa frase



4 – Depois da preparação do arquivo, realizamos o treinamento usando o algorítimo de Naive Bayes.

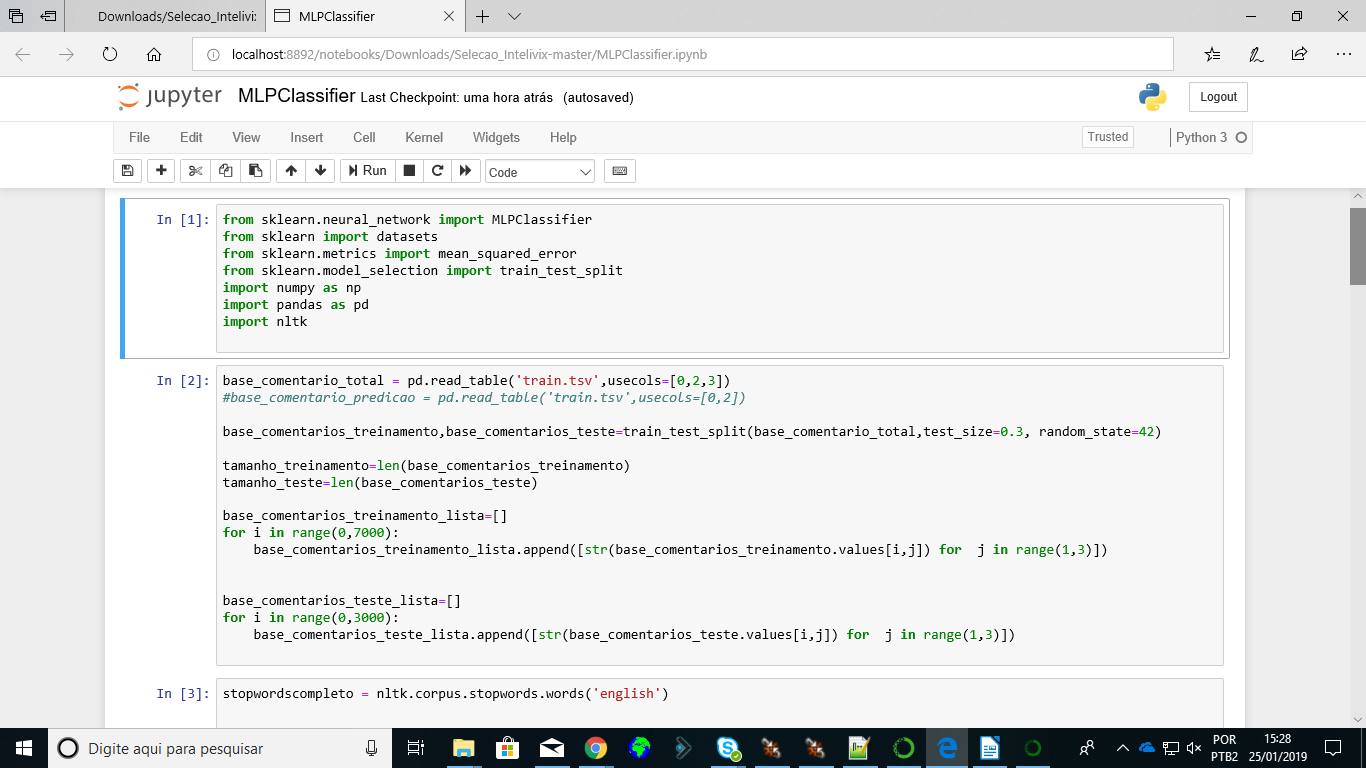
5 – Fizemos os tratamentos finais na base de avaliação, medimos a acurácia e a matriz de confusão



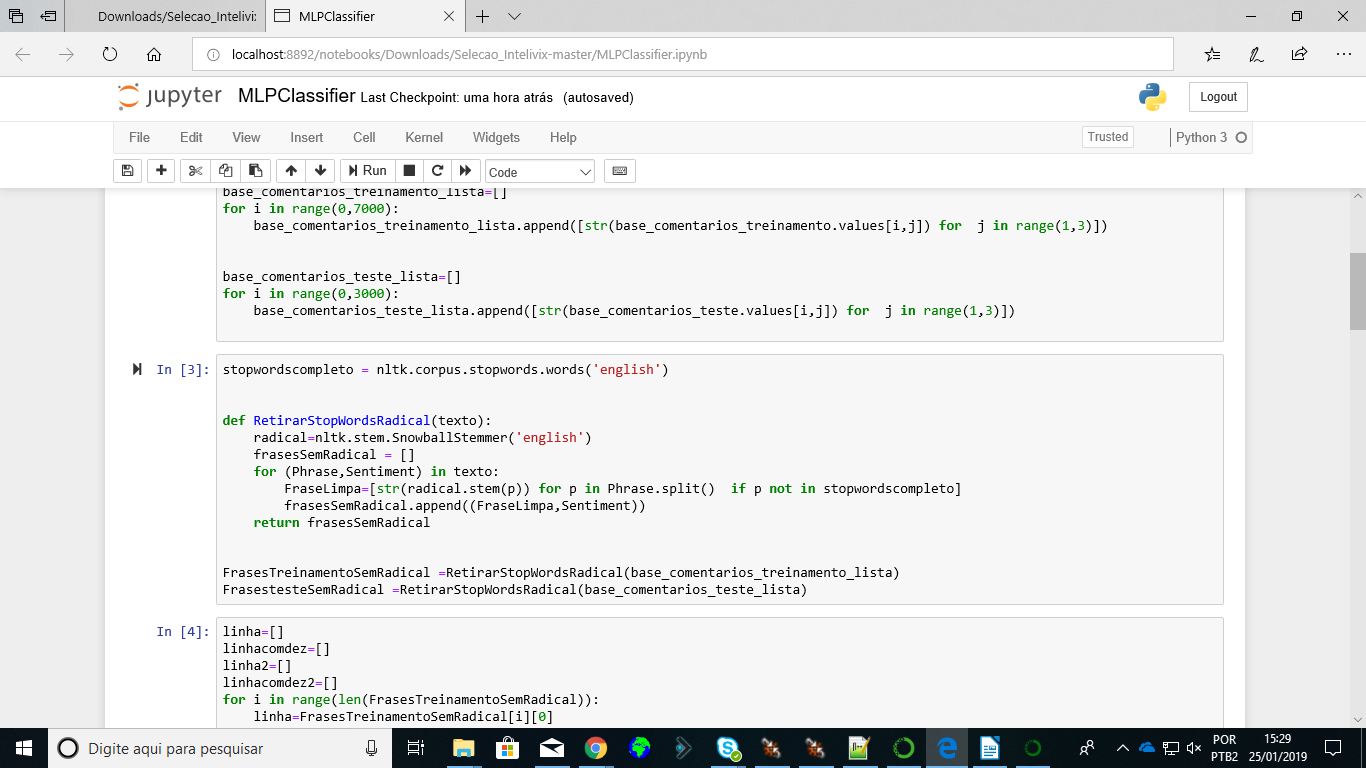
2 – MLPClassifier

**acurácia – 39,16%**

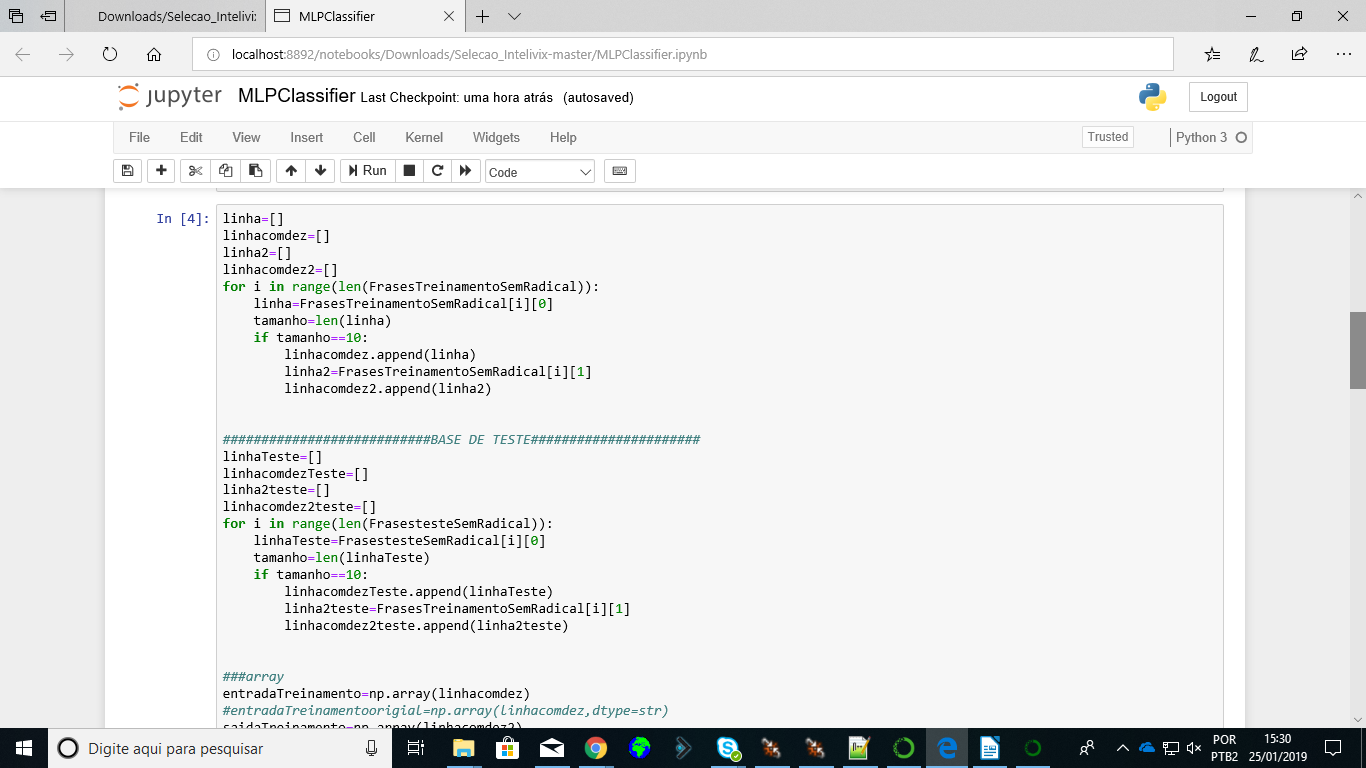
1 - Separamos 70% da base para treinamento e 30 % para base de predição. Para poder conseguir executar (máquina com 4Gb de RAM), tiver que reduzir o tamanho da base.

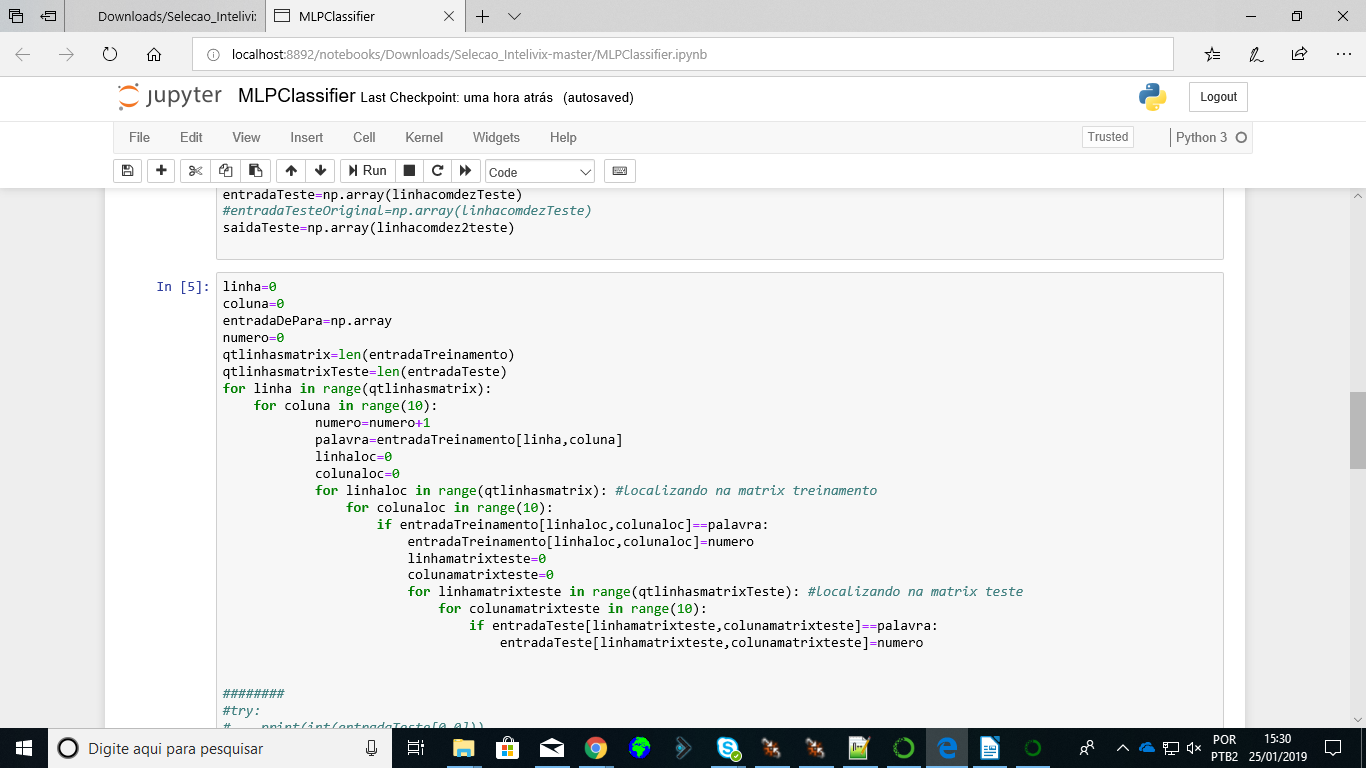


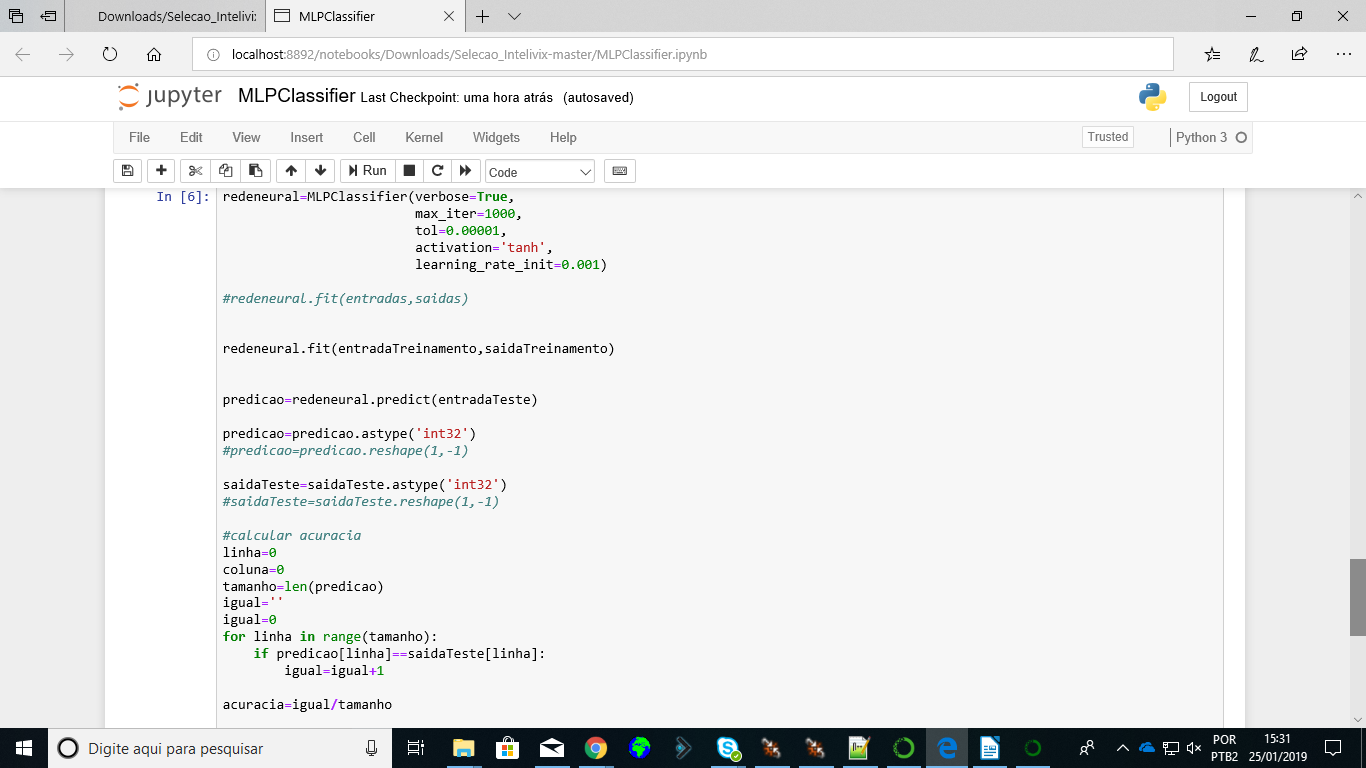
2 – Para utilizar o algorítimo foi necessário, além do tratamento das stopwords, transformar as tuplas em matrizes e para ajudar no tempo de processamento reduzir a dimensionalidade destas matrizes para 10 colunas X quantidade de linhas das bases (treinamento / teste)

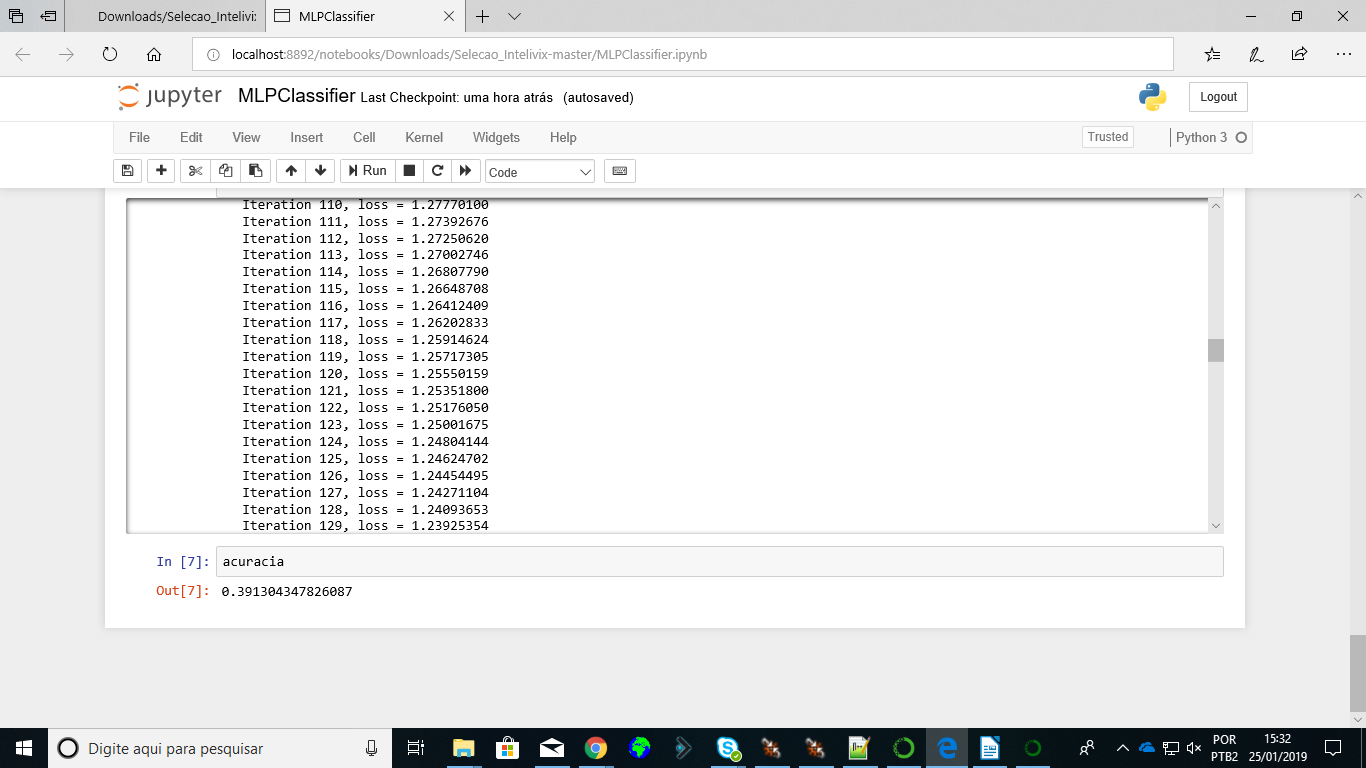


3 – As matrizes de entradas para execução dos algoritmos MLPClassifier tem que ser do tipo numérico, então montei um algorítimo auxiliar para converter cada palavra num número.





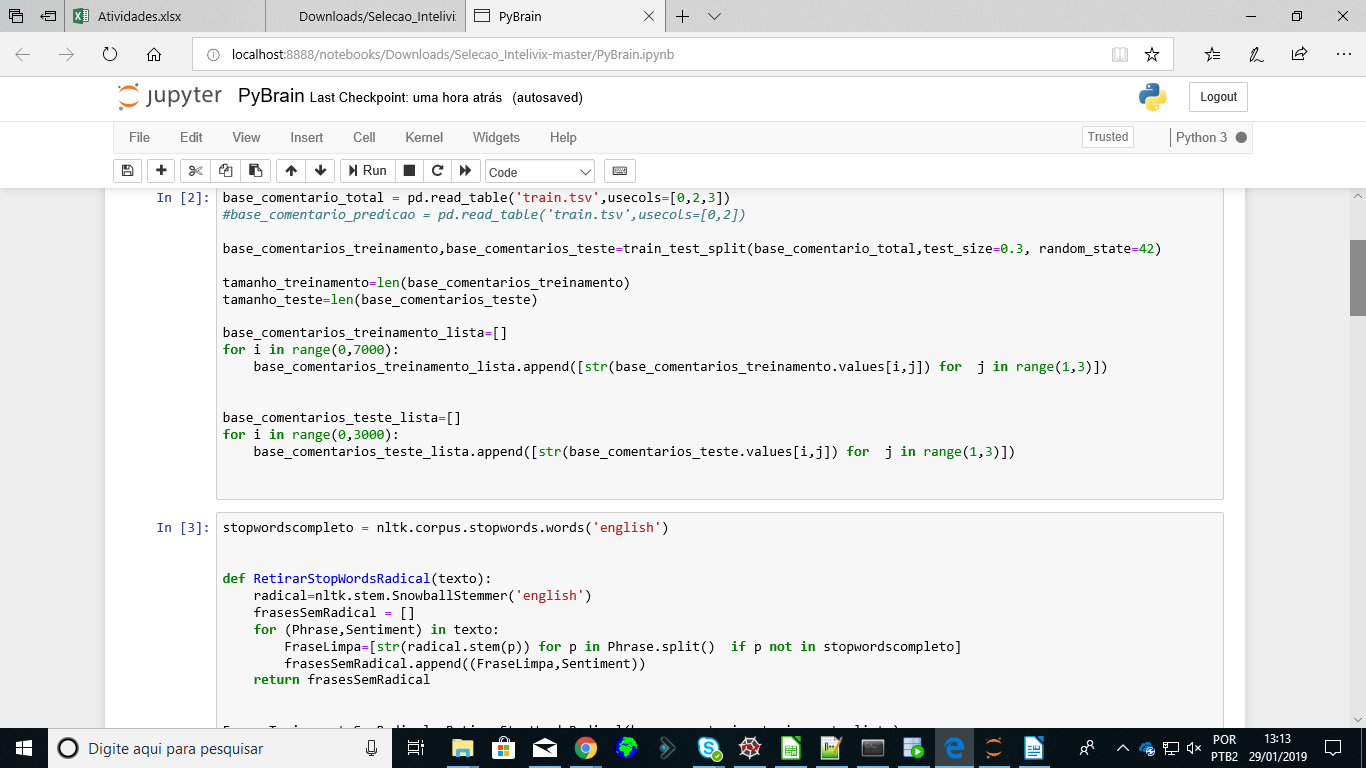
 4 – Executamos o algorítimo e foram testadas algumas configurações para a função e essa foi a escolhida.



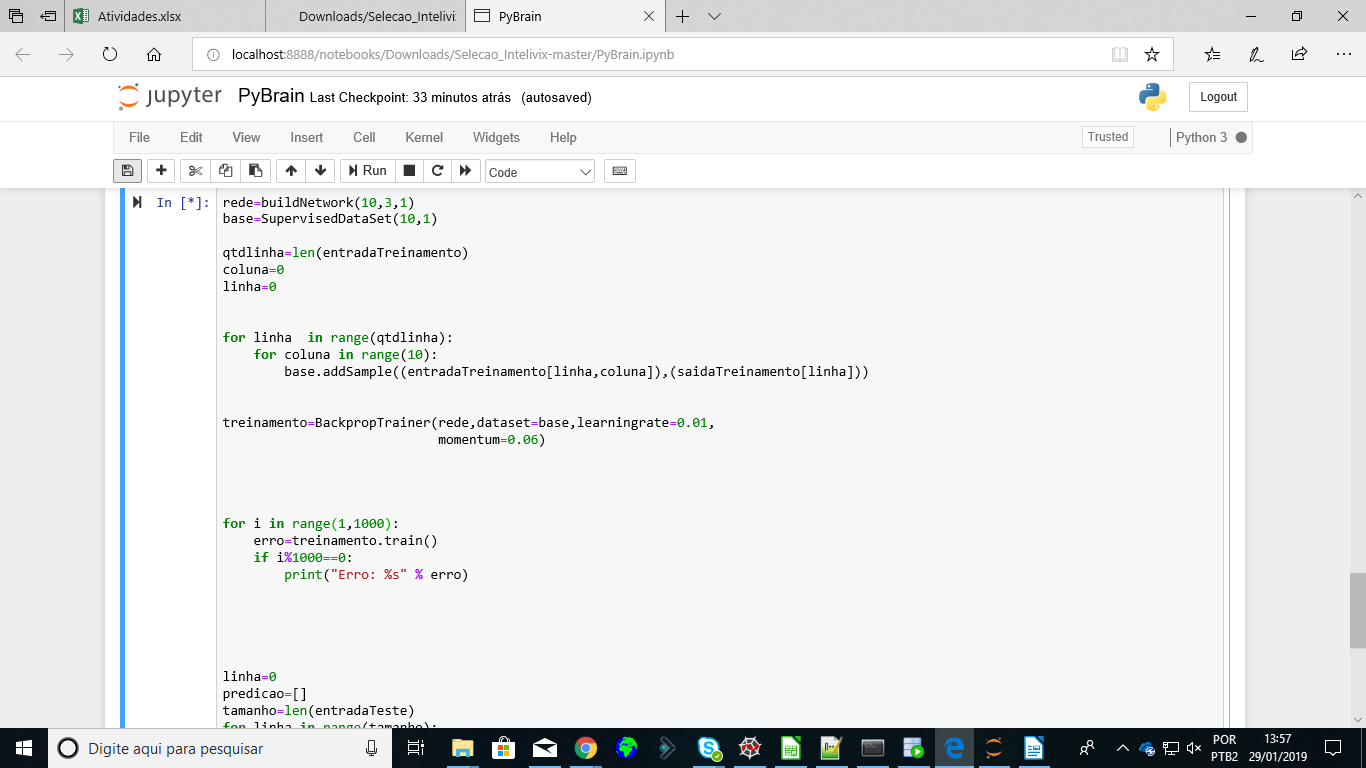
3 – Pybrain

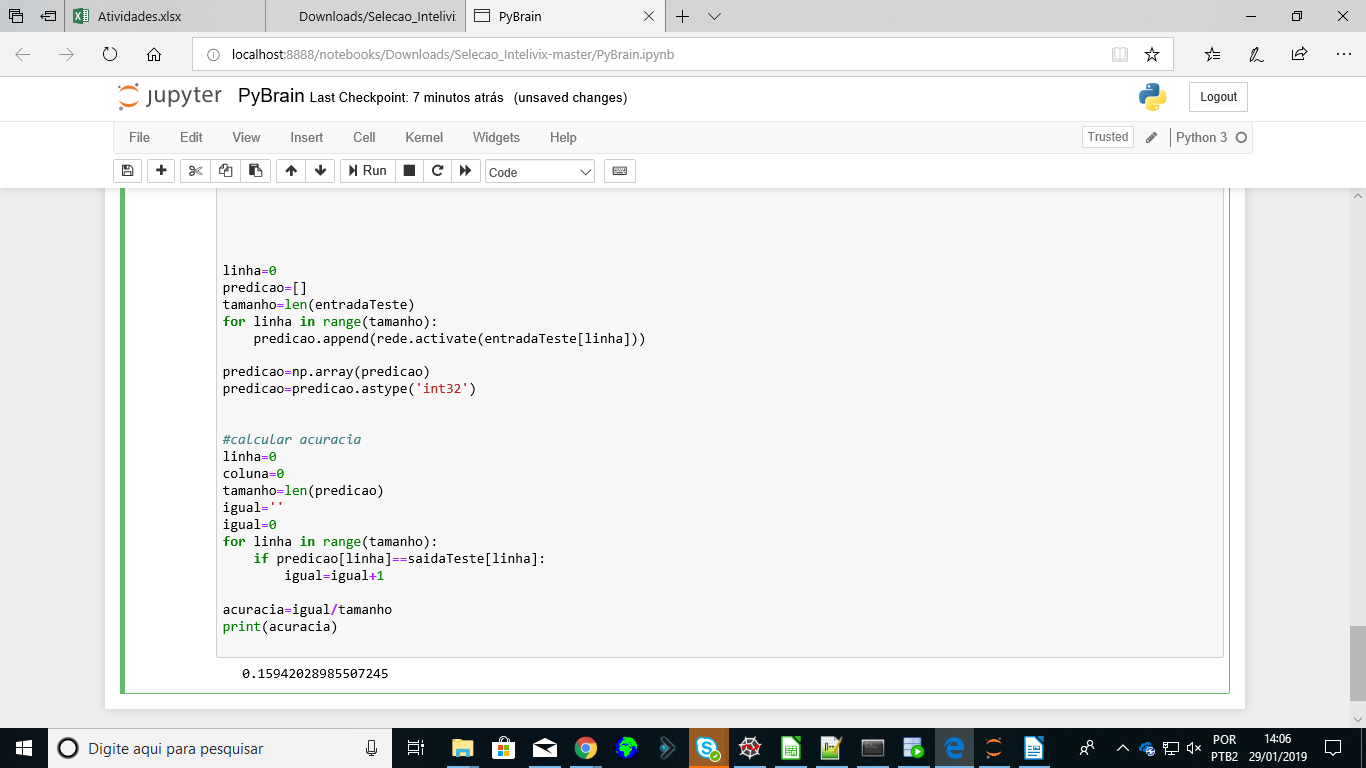
**Acuracia**=15.9 %

1 – Fizemos todo o tratamento da base realizado no algorítimo MLPClassifier.



2 – Usamos as configurações abaixo para treinamento e predição





Observações gerais:

1. Um linguista é necessário para melhorar o tratamento das bases com relação ao significado variado que uma mesma palavra pode ter.
2. Algumas etapas dos algoritmos foram implementadas para poder tratar as bases de tal modo, para que elas fossem usadas nos respectivos algorítimos. Isso deixou a execução lenta.
3. Podemos utilizar outras configurações para os algorítimos utilizados. As regras de negócio vão definir quais devem ser usadas para se obter o melhor resultado.
4. Nos algorítimos de redes neurais e redes densas, ainda falta a função que transforma os números nas palavras originais dentro das matrizes
5. Utilizando a base completa (train.tsv) consegui obter um acerto de 50% (**algoritmo Pybrain**), executando no spyder:

