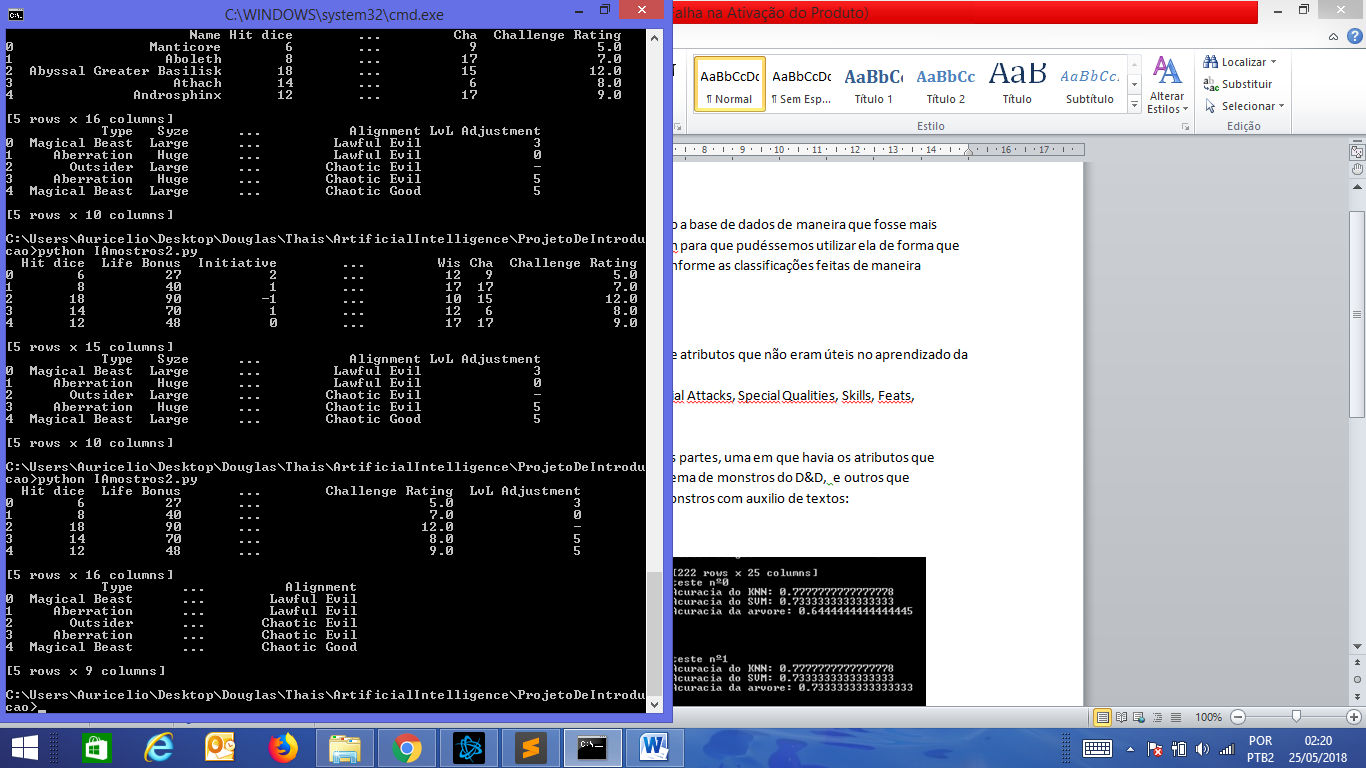
Nos últimos dois dias eu vim pré-processando a base de dados de maneira que fosse mais eficiente que o método depreciado do python para que pudéssemos utilizar ela de forma que as nossas acurácias não ficassem variando conforme as classificações feitas de maneira arbitrária.

Pre-Processamento:

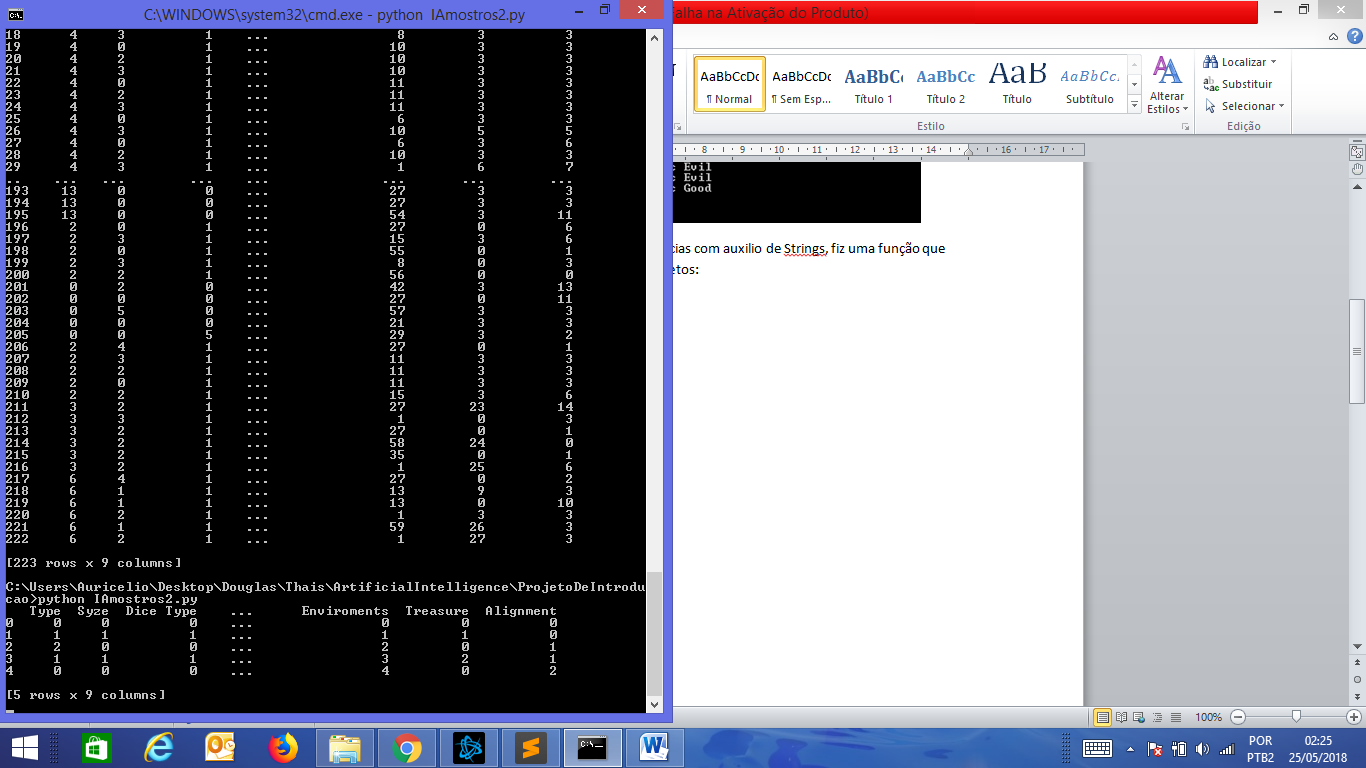
A primeira coisa que foi feita foi a remoção de atributos que não eram úteis no aprendizado da Inteligência Artificial:

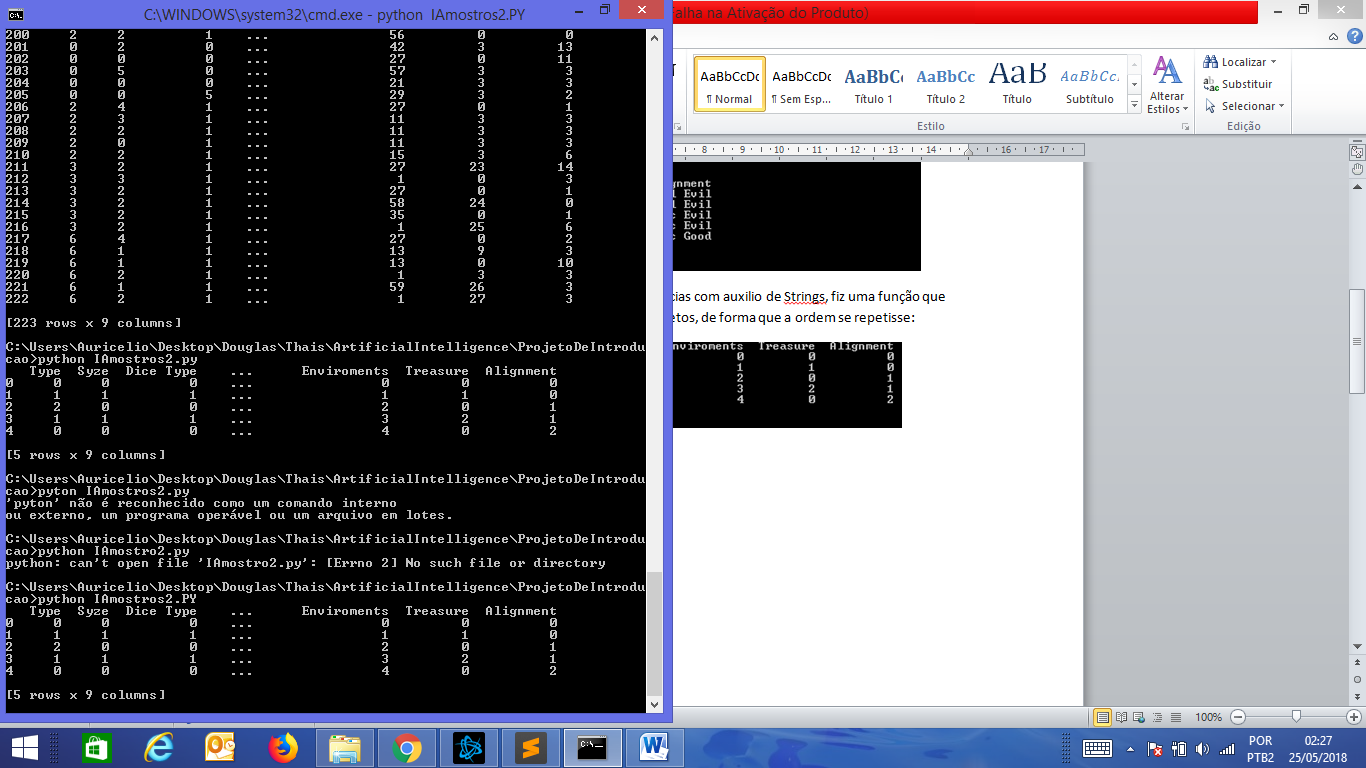
Name, ID,'Sub-Type, Attack, Full Attack, Special Attacks, Special Qualities, Skills, Feats, Organization eAdvancement

Depois disso separei o nosso Dataset em duas partes, uma em que havia os atributos que variavam naturalmente com números no sistema de monstros do D&D, e outros que rotulavam certas características de nossos monstros com auxilio de textos:

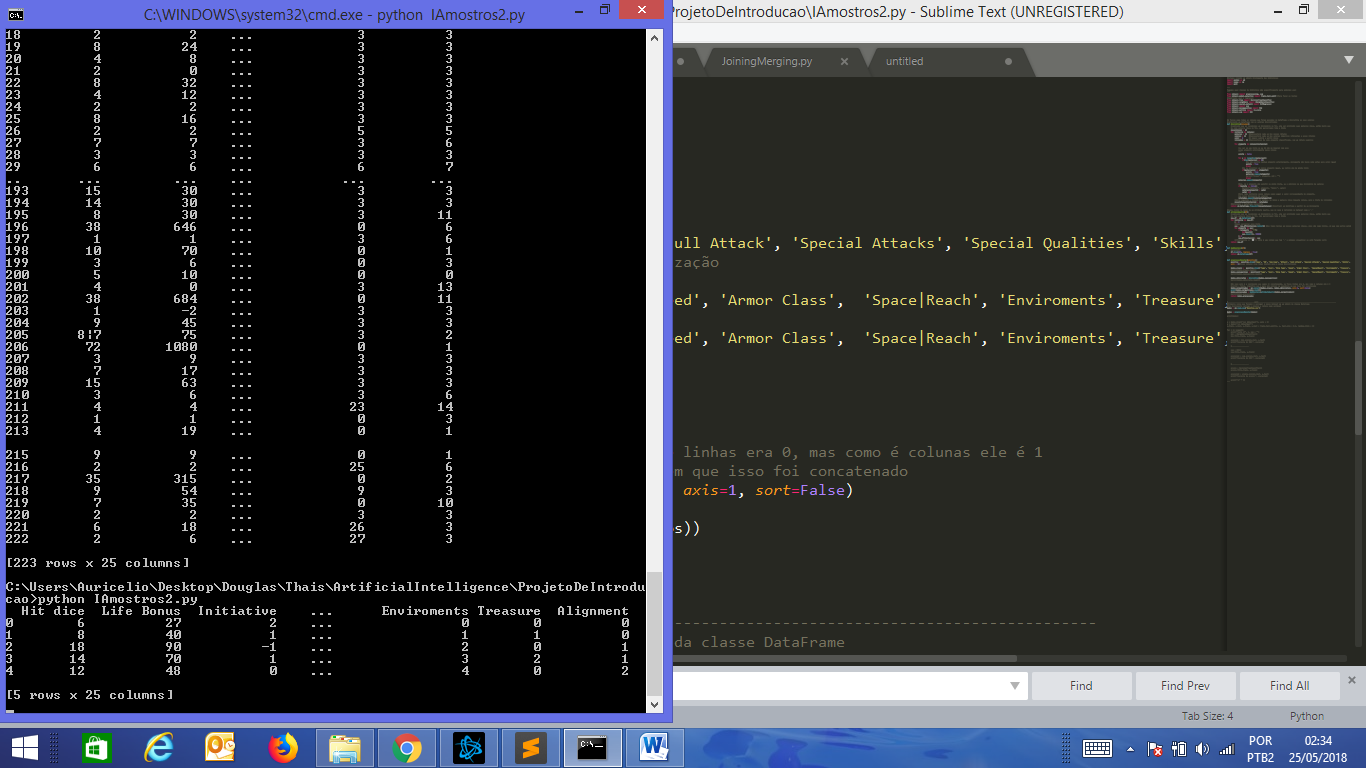


Em relação ao dataset que rotulava as instâncias com auxilio de Strings, fiz uma função que transformava elas em dados numéricos discretos, de forma que a ordem se repetisse todas as vezes que fizéssemos a normalização desses textos:



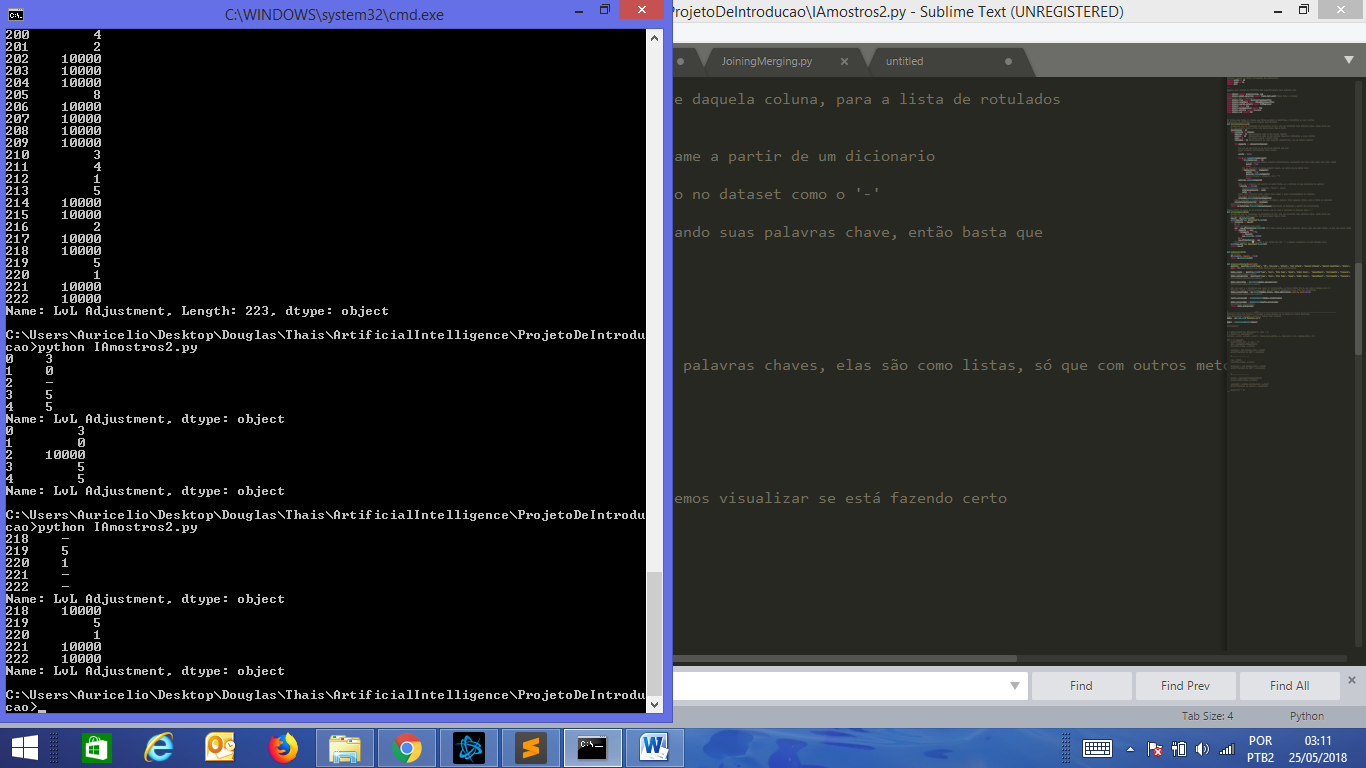


Com isso, justamos novamente, utilizando dos índices numerados pelo próprio pandas foi possível fazer a concatenação das colunas do DataFrame com os dados naturalmente numéricos, com aqueles que eu acabei de normalizar:



Algumas instâncias apresentavam o valor neutro, expresso pelo caractere ‘-‘, ele era uma outlier dentro da base de dados e como sua status de neutralidade era diferente do valor 0 ou 1, o mesmo não poderia ser configurado como elemento neutro multiplicativo ou aditivo.

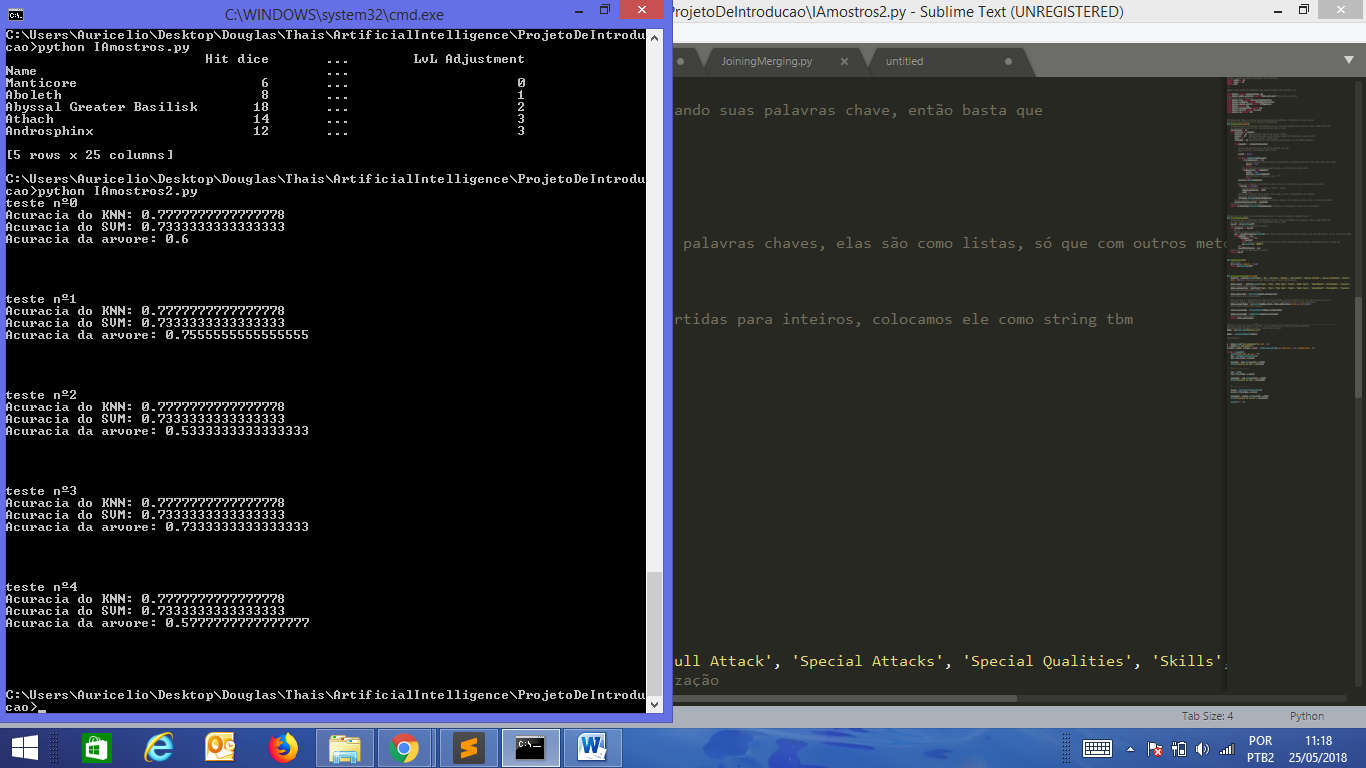
A solução encontrada foi expressa-lo como numericamente como um OutLier, explodindo ele para o valor 10000, local esse que nenhuma numeração chegaria, com isso, podemos fazer a nossa base de dados enxergar esse outlier e prever o mesmo quando vir a aparecer em algum elemento:



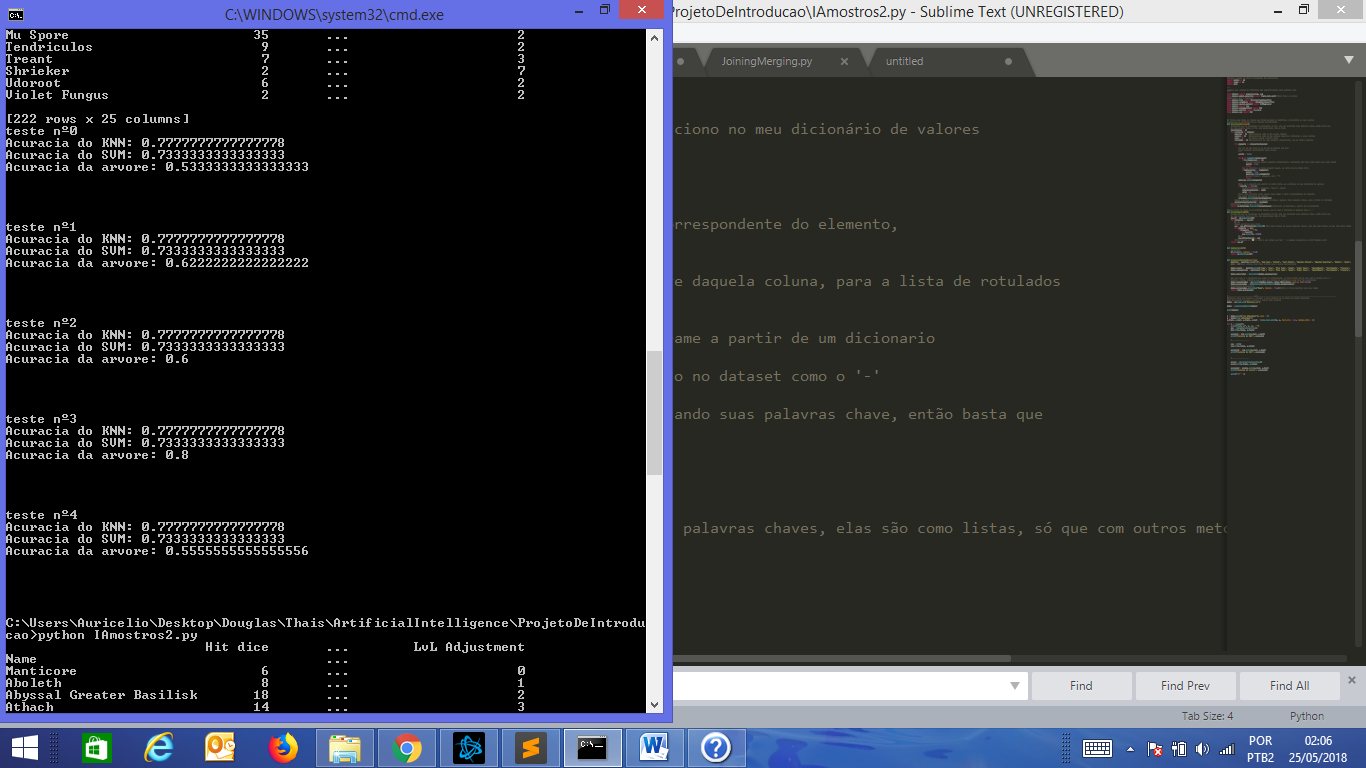
Dentre as instâncias, o elemento 205 apresentava no atributo “Hit Dice” expressa por uma String e era a única diferente dentro de todas as outras onde tínhamos apenas um numero inteiro, chegou-se a conclusão que o melhor seria retira-lo da base de dados e com isso nossos dados.

Processamento e analise da IA:

Mesmo com os dados plenamente processados, nossa IA continua dando valores estranhos no método da árvore de decisão:



A primeira coisa que eu pensei, foi que o Split da nossa base de dados poderia estar acontecendo dinamicamente para cada vez que treinamos as bases de dados, dando então valores diferentes, fiz então respectivamente o teste com a base de dados sendo dividida uma única vez para todos os testes, mas os resultados da arvore continuavam aleatórios:



Metas:

* Procurar entender o método da árvore e ver se assim descubro o motivo da variação
* Em caso de não achar o motivo, plotar os meus dados e procurar um motivo racionalmente os analisando.
* Determinar valores de elementos especificados para o programa manualmente e olhar seus retornos
* Retornar os valores numéricos em Strings que o correspondam a partir da base de dados
* Conversar com um programa Java