**5.Domaći zadatak – Asocijativna analiza**

Cilj asocijativne analize i problem koji se njom rešava je da se za dati skup podataka nađu na neki način pravila koja predviđaju pojavjivanje objekata na osnovu pojavljivanja ostalih objekata u posmatranim podacima. Jedan od algoritama za kreiranje asocijativnih pravila je algoritam FP rasta, tj. „Frequent Pattern Growth” algoritam, koji upotreljava kompresovanu reprezentaciju baze podataka pomoću FP stabla. Svaki podatak iz baze se ovde pedstavlja kao jedan put u FP stalu. Jednom kada je napravljeno stablo, ono čuva sve osnovne informacije iz baze podataka, a dalji paterni se mogu pridobiti korišćenjem rekurzivnog pristupa „podeli i osvoji“ (divide and conquer) za Pre kreiranja FP stabla, izračuna se potpora svakog entiteta i u svakom skupu entiteta, entiteti se uređuju u opadajućem redosledu po potpori. Uz čvor se pamti i broj observacija čije reprezentacije sadrže taj čvor. Takođe, u stablu se ulančavaju i čvorovi koji odgovaraju istom entitetu.

**Dataset**:

Za potrebe domaćeg zadatka korišćen je „ groceries.csv“ skup podataka (u .csv formatu), preuzet sa [https://www.kaggle.com/irfanasrullah/groceries](https://www.kaggle.com/irfanasrullah/groceries%20%20)  („Groceries Market Basket Dataset“ dataset).

Na sledećoj slici prikazan je deo dataseta:

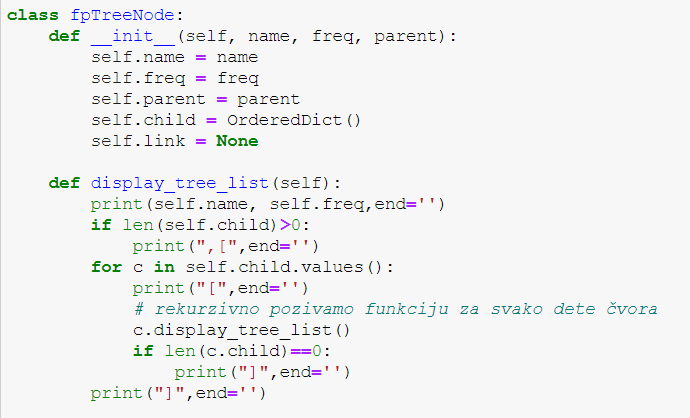


Radi se o datasetu koji sadrži podatke o tome koje sve pehrambene prozvode sadrži „potrošačka korpa“ jednog kupca. Skup podataka sadrži 9835 sadržaja potrošačkih korpi kupaca sa različitim spiskom namirnica. Pribavljene observacije sadrže ukupno 169 jedinstvenih stavki.

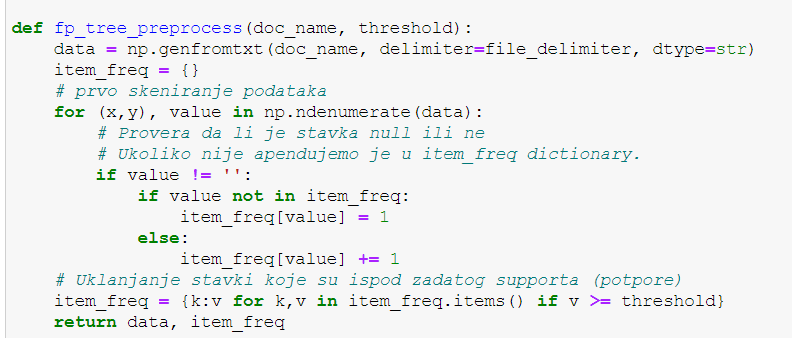
**Obašnjenje zadatka:**

Ova implementacija koristi rekurziju za kreiranje FP stabla. Takođe korišćen je zasebni dictionary za linkovanje svih čvorova sa istim imenom. Ovaj dictionary se kasnije koristi za majnovanje stabla. Sam dataset podataka se skenira 2 puta kako bi se kreiralo FP stablo. Dictionary koji sadrži veze sličnih čvorova kontinuirano se ažurira pri svakom insertu u FP stablo. Zatim se stablo majnuje da bi se pronšle stavke koje se često pojavljuju za inicijalno odabranu stavku.

Klasa **fpTreeNode** () definiše strukturu koja se koristi za kreiranje FP-stabla, kao i metodu za štampanje FP stabla ili uslovnog FP stabla u nested list formatu. Svaki čvor ima ime, frekvenciju - koja se stalno ažurira, OrderedDict za child čvorove, link ka parent čvoru i pokazivač za povezivanje sa drugim čvorovima sa istim imenom. Pokazivač pomaže u stvaranju i održavanju dictionary-ja koji sadrži veze do sličnih čvorova.



Što se tiče same obrade podataka - Funkcija **fp\_tree\_preprocess()** uzima ulazni skup podataka i skenira ga jednom kako bi kreirala dictionary frekventnih stavki. Nakon uklanjanja bilo kakvih null vrednosti i praznih redova, ona kreira dictionary frekventnih stavki. Jednom kada se on kreira, dalje se briše bilo koja stavka iz dictionary koja je ispod zadatog supporta (vrednosti potpore).

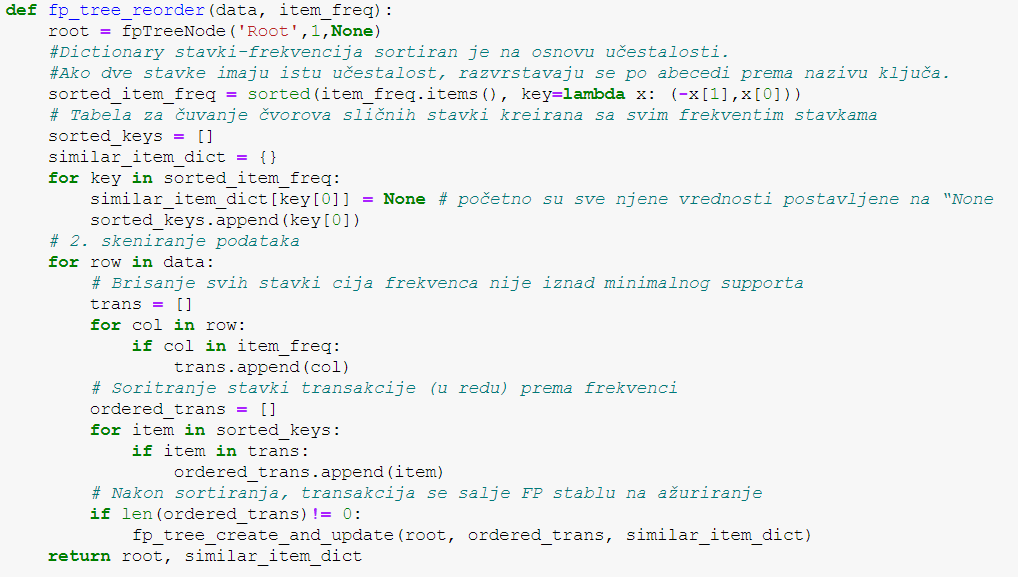


Funkcija **fp\_tree\_reorder()** po drugi put skenira skup podataka. Nekoliko stvari se istovremeno dešava u ovoj funkciji. Nakon brisanja stavki sa manjom vrednosti support(potpore), sortira stavke u datoj transakciji(redu). Sortirana transakcija se zatim šalje funkciji za kreiranje FP stable.

Funkcija prvo kreira Root čvor za FP-stablo. Sam dictionary stavki-frekvencija sortiran je na osnovu učestalosti. Ako dve stavke imaju istu učestalost, razvrstavaju se po abecedi prema nazivu ključa.Tabela za čuvanje čvorova sličnih stavki je konstruisana i početno su sve njene vrednosti postavljene na “None”.

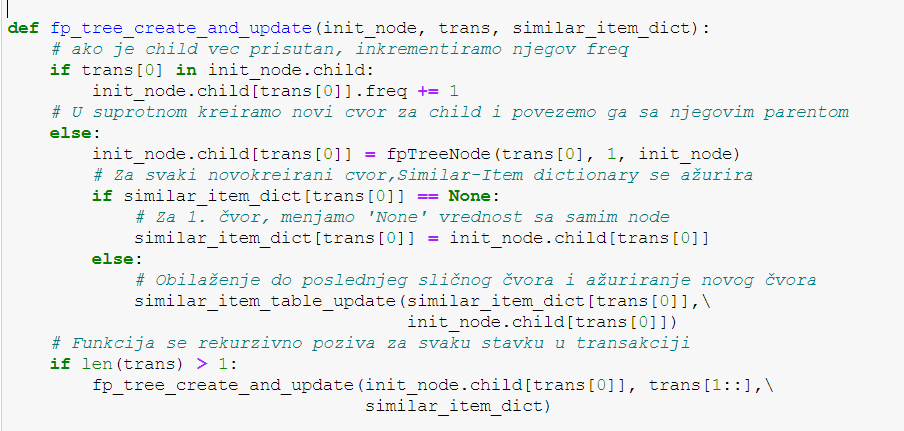
Na osnovu ključeva sortiranog dictionary-ja frekvencija stavki, stavke u svakoj kupovini se preuređuju. Odbacuje se svaka stavka koja nije prisutna u ključevima dictionary-ja.

Sortirana transakcija(sama kupovina u ovom slučaju. tj. jedan red u datasetu ) sada se šalje FP-Tree-u za izgradnju stabla.

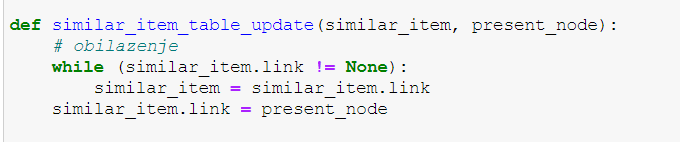


Dve rekurzivne funkcije **fp\_tree\_create\_and\_update()** i **similar\_item\_table\_update()** rade istovremeno.

U funkciji fp\_tree\_create\_and\_update (), svaka stavka u transakciji (transakcija = red skupa podataka) dodaje se FP-stablu u pravilnom redosledu. Ova funkcija se rekurzivno poziva za svaku stavku u transakcij. Ako stavka nije prisutna kao dete za svoj roditeljski čvor, kreiraće se novi čvor. Ako je već dodata, njena učestalost se u tom slučaju uvećava. Ovaj postupak se nastavlja za svaku stavku u datasetu. Kada se transakcija završi, funkcija prima novu transakciju (red) od prethodne funkcije fp\_tree\_reorder ().



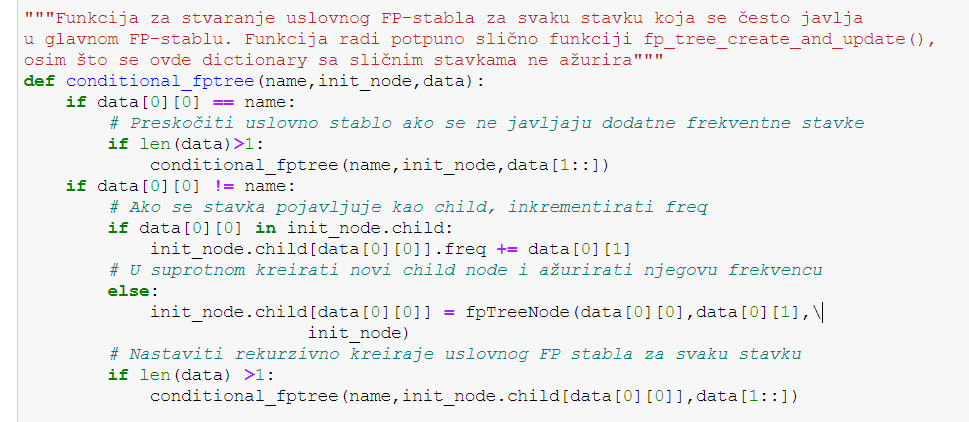
Svaki put kada se kreira novi čvor, rekurzivna funkcija **similar\_item\_table\_update ()** se poziva u okviru funkcije fp\_tree\_create\_and\_update (). Kao što je ranije pomenuto, dictionary similar\_item pomaže u povezivanju svih čvorova sa istim imenom. Dakle, svaki put kada se kreira novi čvor, funkcija similar\_item\_table\_update () prolazi kroz dictionary kako bi pronašla poslednji čvor sa istim imenom i povezuje novi čvor sa prethodnim čvorom.



Stvaranje Conditional Leaf Base-a:

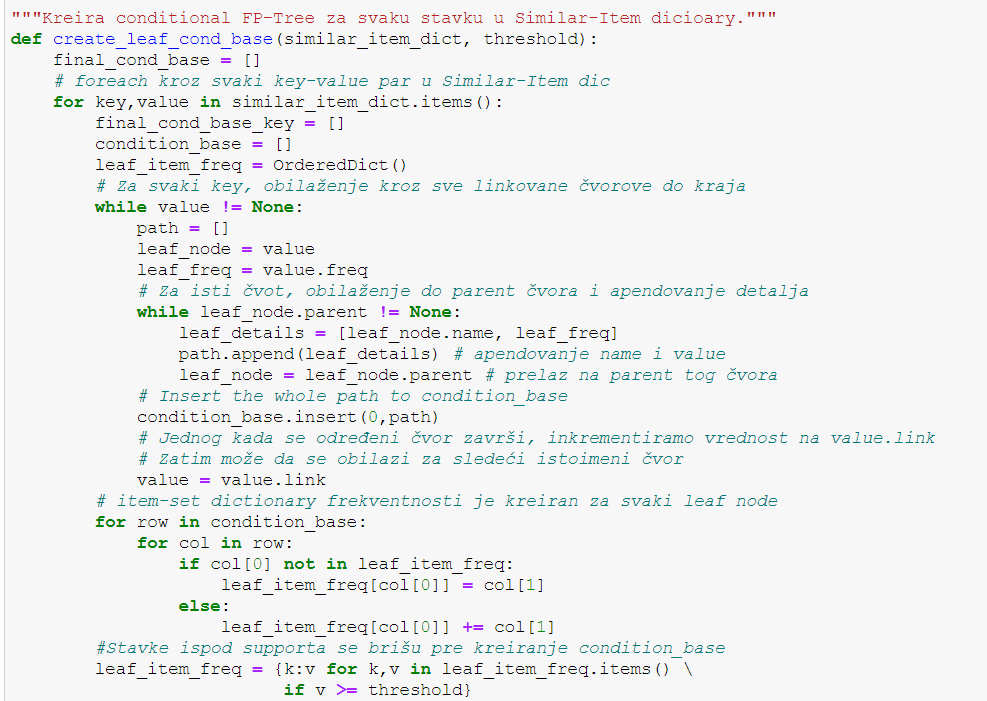
Kako bi se pronašli paterni frekventnih stavki korišćenjem FP Growth agoritma, uslovno FP stablo mora da se generiše za svaku stavku koja se često pojavljuje. To se radi uz pomoć prethodno kreiranog dictionary-ja sličnih stavki. Za svaku stavku koja se pojavljuje u toj tabeli kreira se uslovno FP-stablo i skupovi stavki ispod vrednosti za support(potpore) se ponovo uklanjaju.

**conditional\_fptree()** - Funkcija za stvaranje uslovnog FP-stabla za svaku stavku koja se često javlja u glavnom FP-stablu. Funkcija radi potpuno slično funkciji fp\_tree\_create\_and\_update(), osim što se ovde dictionary sa sličnim stavkama ne ažurira



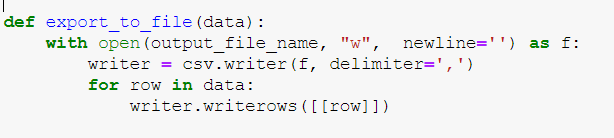
Funkcija c**reate\_leaf\_cond\_base()** radi sledeće: prolazi kroz svaki čvor za određeni ključ u dictionaty-ju sličnih stavki. Unutar svakog čvora odradi traverse do korena FP-stabla i održava logove kao evidenciju svakog imena čvora kojim je obišla da bi došala do korena.

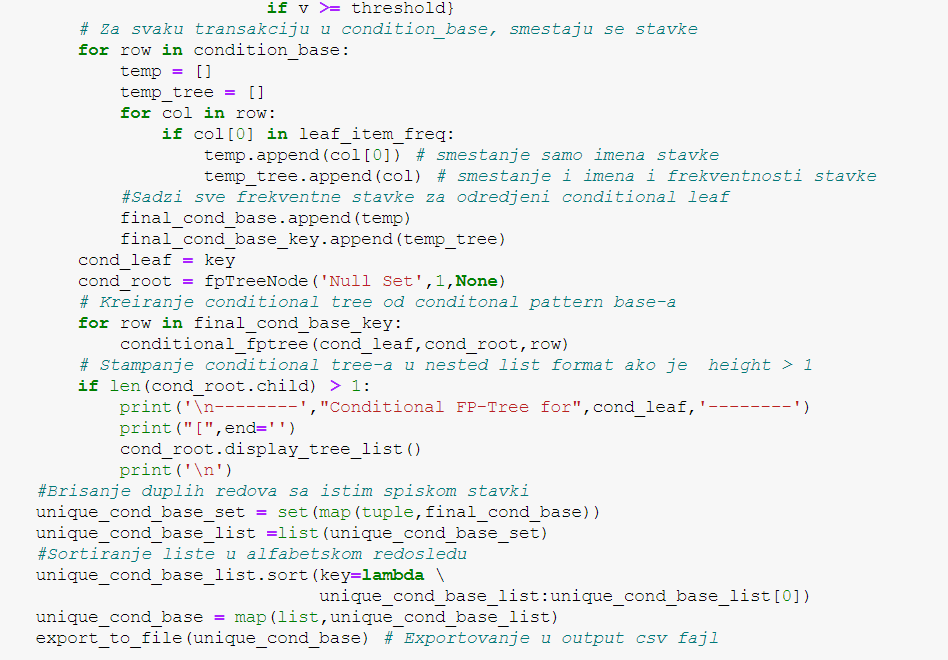
Jednom kada bazu uslova formiraju svi čvorovi određenog „ključa“, generiše se tabela sa čestim stavkama za conditional leaf.



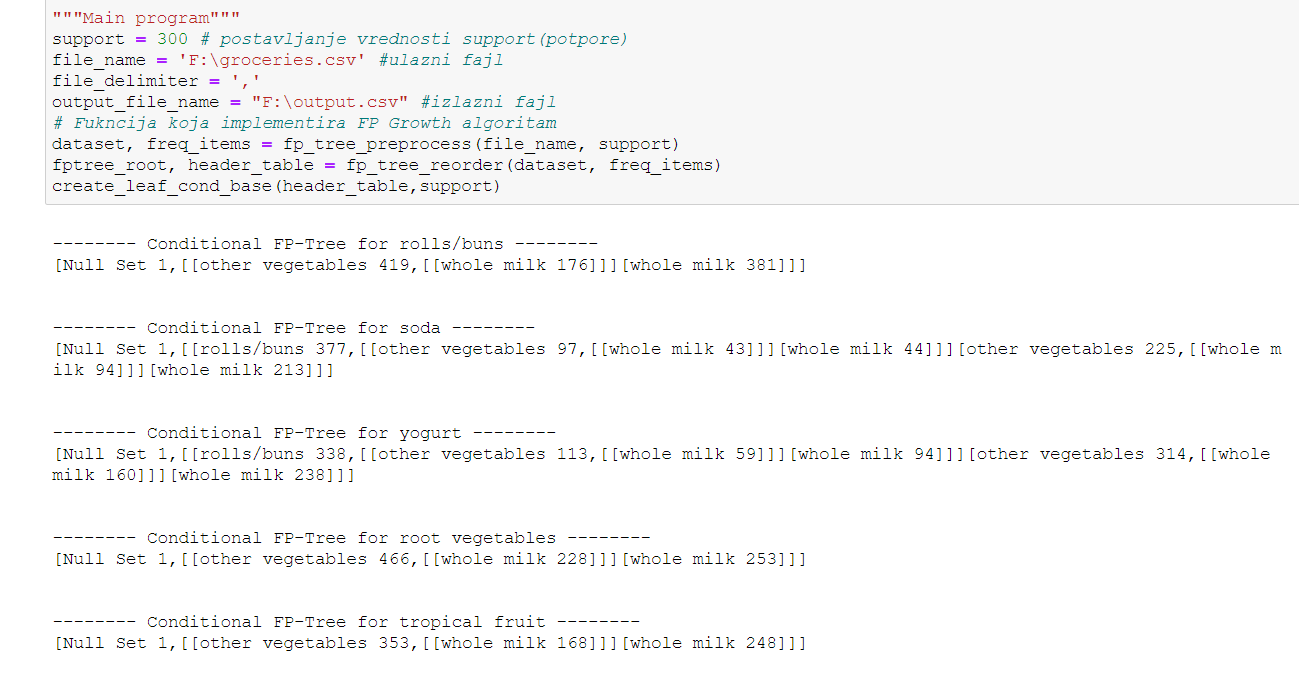
Stavke ispod zadate potpore(supporta) uklanjaju se iz tabele, a takođe i iz uslovne baze.

Sve ostale stavke dodaju se u CSV datoteku kao česti skupovi stavki. (Za ovu potrebu korišćen je fajl “output.csv” priložen sa ostalim fajlovima u zadatku). Za tu potrebu se koristi funkcija **export\_to\_file()**





Glavni program, definisanje ulaznog i izlaznog csv fajla, pozivanje fukncije koja implementira FP Growth algoritam:



output.csv:

