Informe Pràctica 2 IAA

Gerard Gómez Izquierdo

Quadrimestre 3

01/01/2023

FIB-GIA

IAA

Índex

1. Anàlisis i preprocessat de dades: Profiling

La base de dades amb la que treballarem a continuació ve d'una empresa de proveïdora de telefonia i internet. Aquest data set conté informació sobre els clients de la companyia, incloent informació personal, així com detalls específics sobre els seus plans de telefonia, d'internet i ús. A més a més d'això, la base de dades inclou informació sobre si el client ha marxat o no de la companyia. Aquest és un factor molt important per a la companyia, ja que els hi pot ajudar a entendre els factors que contribueixen a que un client marxi i com canviar la manera de treballar per retenir més clients. Amb aquest anàlisi, pretenem guanyar coneixement dels comportament característics dels clients de la companyia per predir quins clients marxaran i recomanar que ha de fer l'empresa per retenir més clients.

1.1. Anàlisi de correlació de les variables

Variables Categòriques:

Abans d'estudiar la correlació que hi ha entre les variables numèriques que es troben en la nostra base de dades anem a estudiar que fer amb certes variables que poden ser redundants. A més es faran algunes propostes de fer merge entre diferents variables.

Customer ID: no ens aporta informació útil per al model

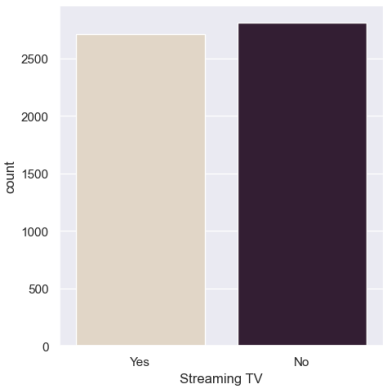
Gender: aquesta variable podria donar lloc a resultats on es diferencia les probabilitats de deixar l'empresa pel fet de ser home o dona. Aquest fet podria fer que les empreses tinguin preferència a l'hora d'escollir homes o dones i no volem que això succeeixi ja que èticament considero que no és correcte, per aquesta raó eliminarem aquesta variable.

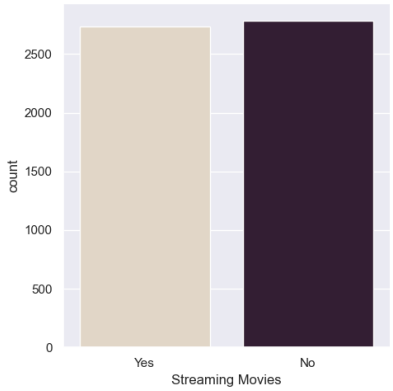
City: Ens aporta informació igual que a zip code, a més zip code ens aporta també una component de distancia entre ciutats, veurem que fer amb aquesta variable més endavant quan analitzem en profunditat la variable zip code, latitud i longitud.

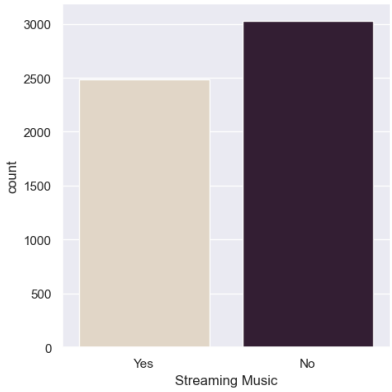
Internet Service: Ens proporciona informació que ja inclou la variable Internet Type ja que aquesta ja ens diu si té internet o no a més de quin tipus en té.

Streaming TV, Streaming Movies, Streaming Music: Aquests serveis d'Streaming ens plantegem agrupar-los tots en una variable que et digui el nombre de serveis que té cada client, de manera que reduirem la complexitat i mantindrem la informació de quants serveis de streaming té cada client. Però abans d'eliminar aquestes variables anem a veure quina part dels clients que marxen tenen cada servei per veure si el fet de tenir un Streaming en concret ens aporta molta informació sobre si el client marxa o no. Gràfics a la base de dades total:

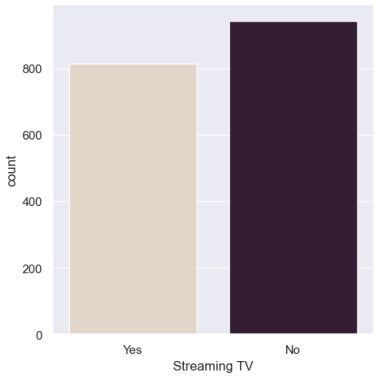
Gràfics a la base de dades total:

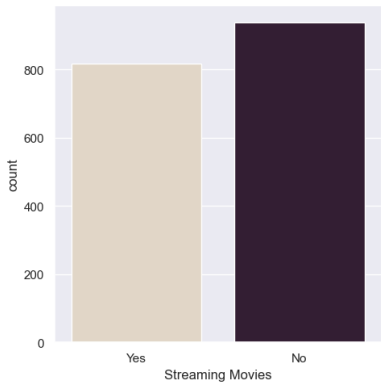


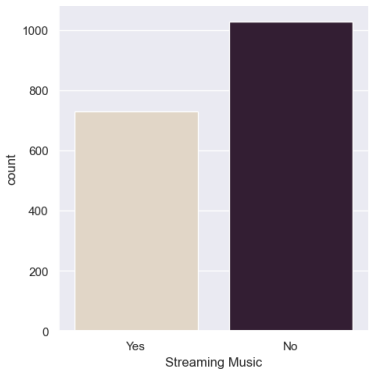




Gràfics a la base de dades amb només clients que han marxat:



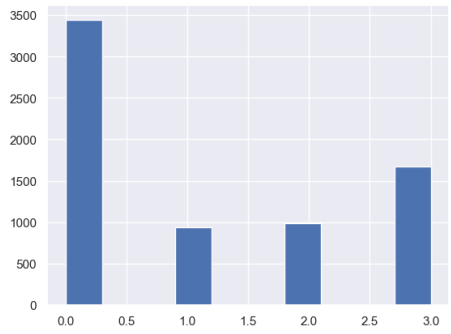




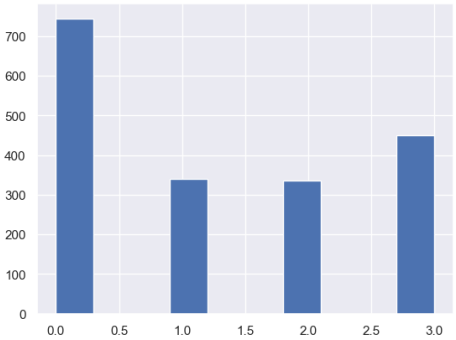
Aquests gràfics s'han fet respecte una base de dades que està formada únicament de clients que han marxat (churned) i la base de dades original. D'aquesta manera ara podem veure com els 3 streamings gairebé no influeixen a la variable objectiu, ja que la proporció dels clients que tenen cada servei de Streaming és molt semblant tant a la base de dades original com a la de clients que han marxat. Si per exemple haguéssim tingut un augment molt gran de la proporció de gent que té el servei de Streaming TV a la base de dades amb només clients que han marxat podríem assumir que tenir aquell servei en concret és determinant per decidir si un client es queda o no però no és el nostre cas. De manera que si ajuntem totes les variables en una a priori no perdrem informació rellevant.

Analitzem la distribució de la nova variable Numero Streams:

Distribució al data set total:



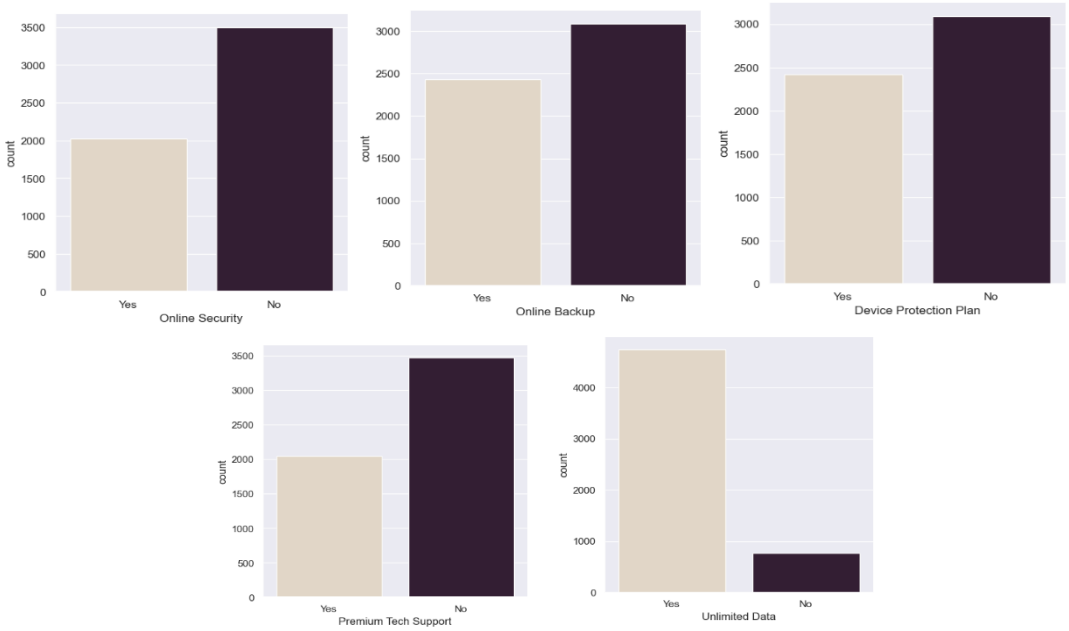
Distribució al data set de client s que han marxat:



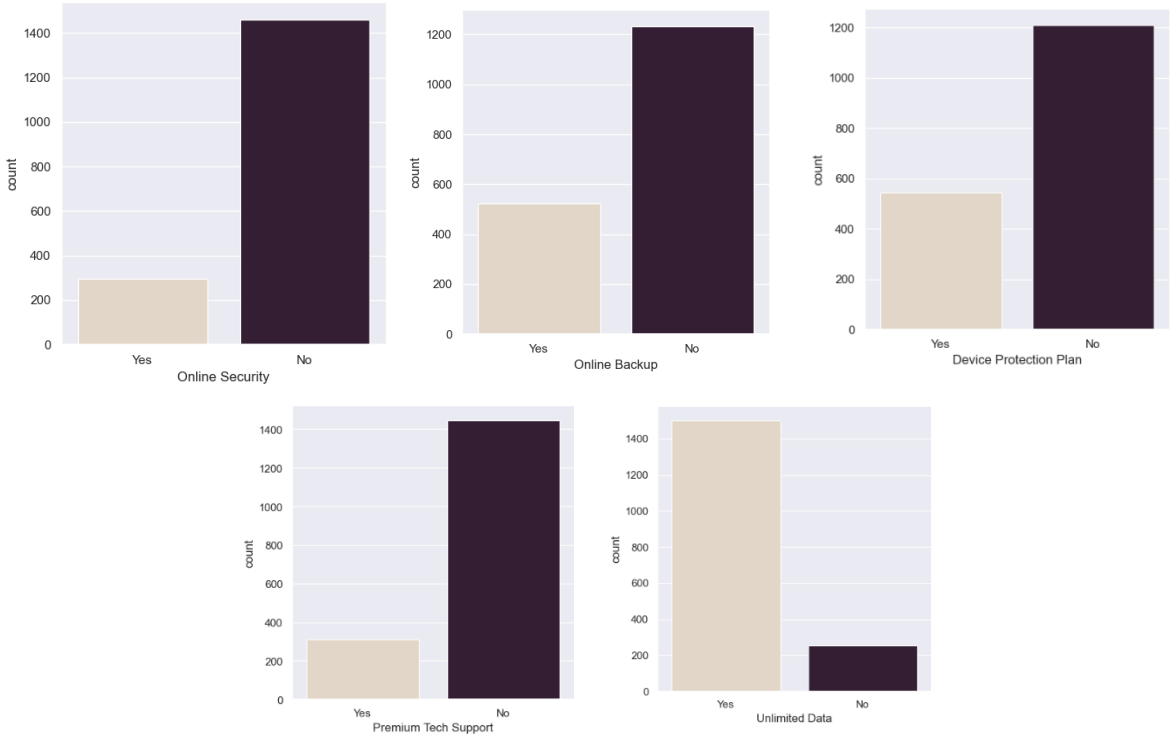
En els gràfics podem veure la distribució que segueix la variable Numero Streams en la base de dades amb tots els clients (esquerra) i en la que només té clients que marxen. Podem veure com la proporció de clients amb 1, 2 o 3 serveis augmenta una mica en la distribució dels clients que han marxat. D'aquest augment de proporció podríem intuir que els clients amb algun servei d'stream tenen més tendència a marxar.

Online security, Online backup, Device Protection Plan, Premium Tech Support i Unlimited Data: També podem agrupar-ho com una nova variable de serveis Premium per reduir la complexitat. Aquesta prendria els valors 0,1,2,3,4 depenent del nombre de serveis dels que despongui el client. Però per fer-ho haurem de mirar com afecten els diferents serveis Premium en concret a la variable objectiu.

Gràfics a la base de dades total:



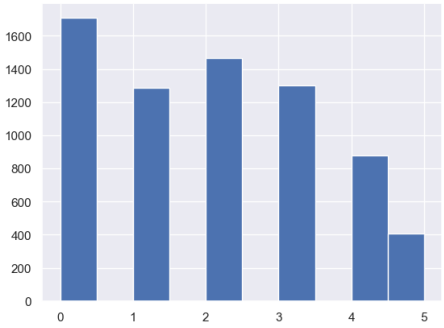
Gràfics a la base de dades amb només clients que han marxat:



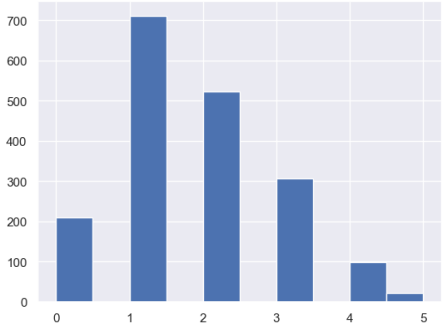
Podem veure que per norma general en tots els serveis premium la proporció de clients que sí tenen algun servei premium disminueix i la que no augmenta, això implica que per norma general els clients que tenen algun servei premium que proporciona l'empresa tenen més tendencia a quedar-se. A més també podem veure que no hi ha cap servei en concret el qual ens aporti molta informació sobre si el client es queda o marxa així que a priori al juntar les variables en una no perdrem informació rellevant.

Analitzem la distribució de la nova variable Premium services:

Distribució al data set total:



Distribució al data set de clients que han marxat



En els gràfics podem veure la distribució que segueix la variable Premium Services tant a la base de dades amb tots els clients com en la que només està formada per clients que han marxat. Quan analitzem els gràfics podem veure que la proporció de clients amb 0 serveis Premium disminueix molt notablement, això ens indica que els clients amb 0 serveis premium no tenen tendència a marxar de la empresa. També podem veure com és redueix notòriament la proporció dels clients

amb 3, 4 i 5 serveis premium, això ens pot indicar que els clients que tenen més de 3 serveis Premium tenen menys tendència a marxar. Pel que fa els clients amb 2 serveis també redueix una mica la proporció. En canvi la proporció de clients amb 1 servei augmenta i per tant podem intuir que aquests clients tindran més tendència a marxar que els altres.

Churn Category : Analitzarem que fer amb ella més endavant.

Conclusions:

Com a conclusió d'aquest primer anàlisi de la relació i la redundancia de les variables ctegòriques farem el següent:

Crear dues noves variables que recopilin el nombre de streams que té cada client i el nombre de serveis premium, d'aquesta manera aconseguim reduir la complexitat de la base de dades i això ens anirà molt bé en algoritmes com KNN. No eliminarem les altres columnes perquè en algoritmes que utilitzen arbres de decisió ens podrien funcionar bé, més endavant provarem els algoritmes amb diferents variables per veure el seu funcionament. Les variables redundants que eliminarem són: **Customer ID, Gender i Internet service**. I les variables que vigiliarem són: **City i Churn Category**.

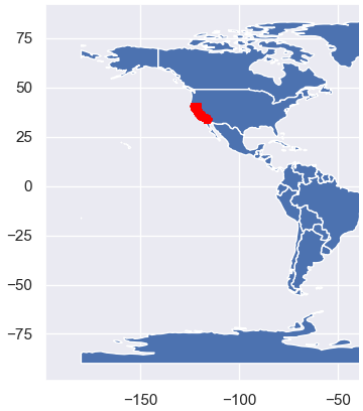
Variables Numèriques:

Un cop hem marcat que volem fer amb les variables categòriques esmentades anteriorment anem a estudiar que fer amb les variables de Latitud, Longitud i zip code i estudiarem les correlacions que existeixen entre totes les variables numèriques.

Latitud i longitud: La variable de Latitud i Longitud ens aporta informació sobre la localització de el client i ens podria aportar saber si els clients que viuen en alguna zona en concret tenen tendència a marxar de l'empresa. L'inconvenient que se'ns presenta amb aquesta variable és que en principi els models tractaran aquestes dues variables com a independents i no per parelles que és com representen informació útil. Com el model no tractarà correctament aquesta variable és millor eliminar-les i utilitzar la informació geogràfica del zip code per tenir en compte la localització de el client.

Abans d'eliminar la variable anem a veure quines són les localitzacions dels clients a Estats Units per veure on es troben aquests.

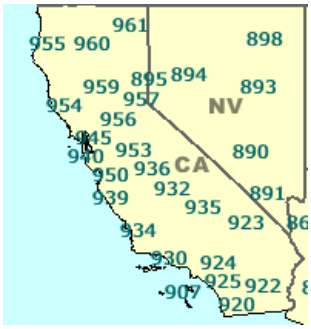
Grafic de les coordenades del dataset:



Imatge del estat de California:

Podem veure com tots els clients viuen a California.

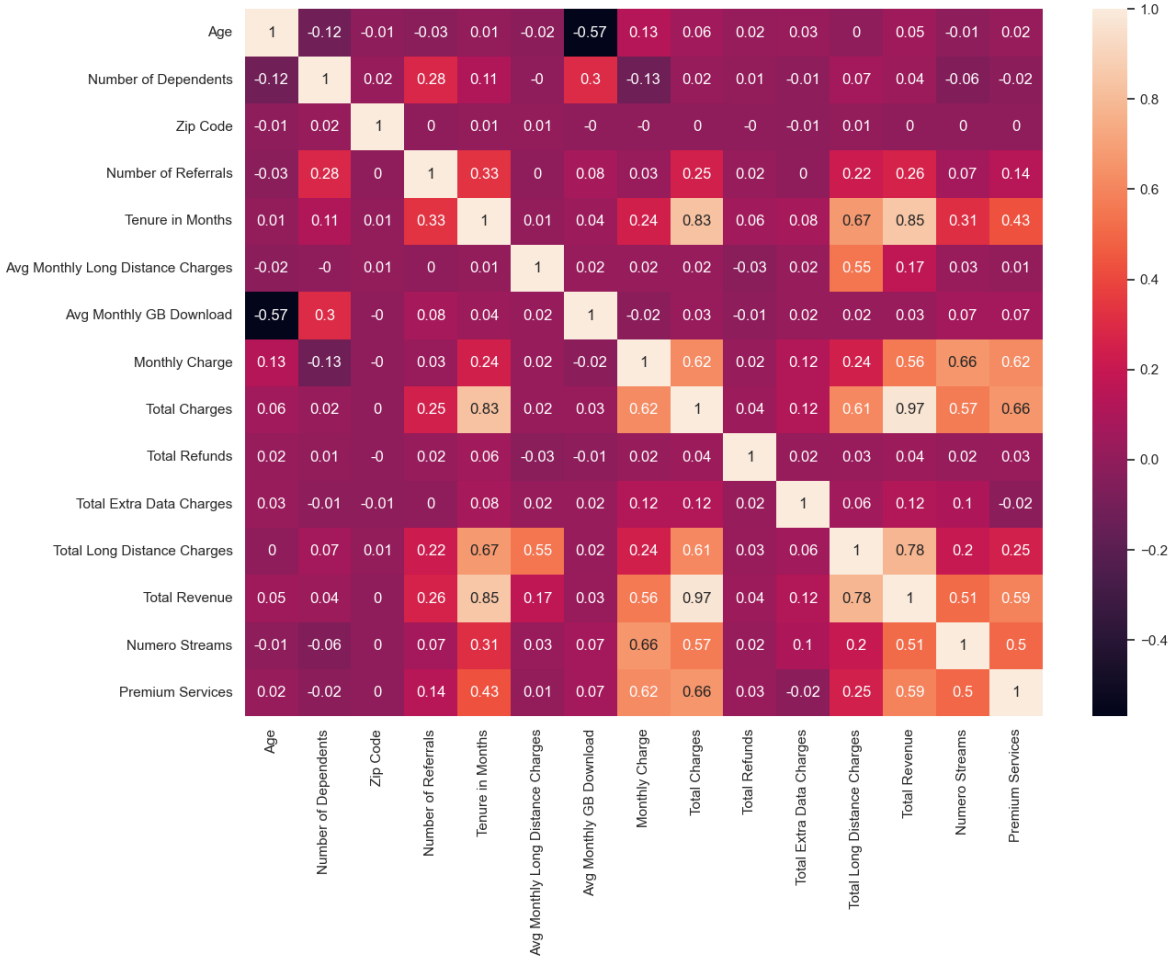
Zip Code: Amb la variable Zip code el model pot saber si els clients que viuen en una ciutat en concret són més o menys propensos a marxar. A més, aquesta variable també ens aporta informació amb una component geogràfica com podem veure en la següent imatge:



Com es pot veure Califòrnia en molts casos encarar que això no succeeix sempre, les ciutats properes tenen valors propers i ciutats llunyanes valors separats. Aquesta informació pot ser útil per al model.

Un cop decidit que utilitzarem Zip Code ja podem eliminar Latitude Longitude i City.

Analitzem Correlacions:



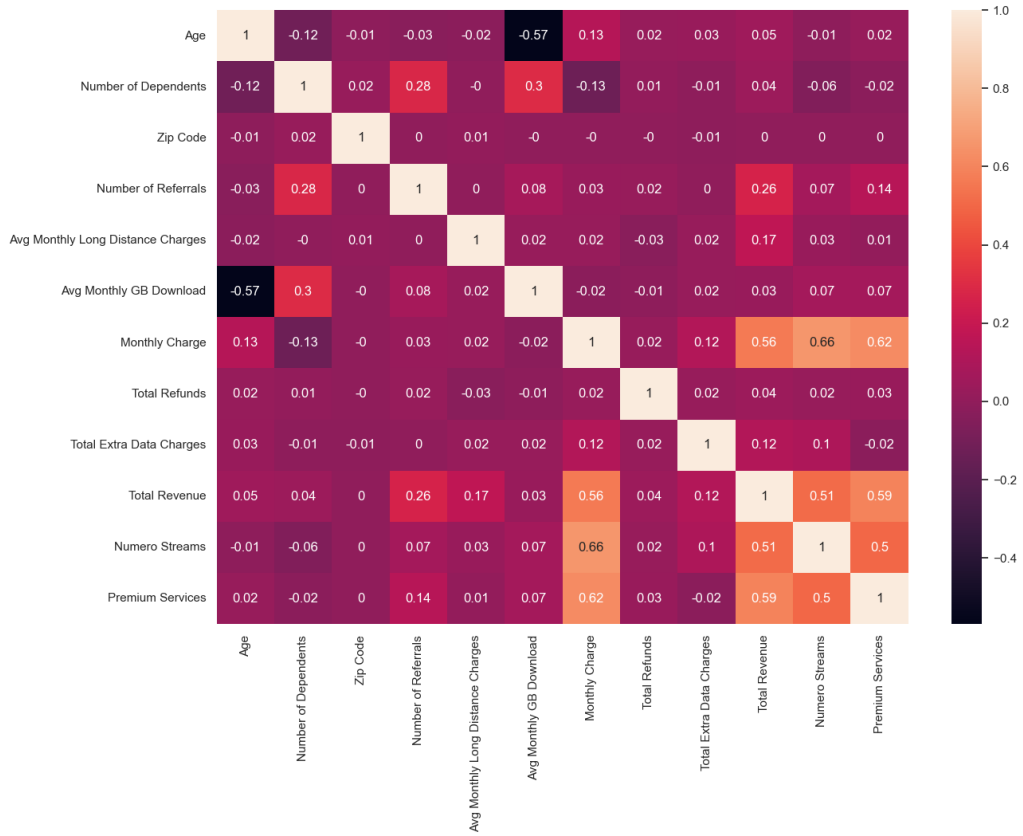
Variables amb correlacions altes:

La primera variable que veiem que té correlacions altes és 'Tenure in months', aquesta variable indica quant de temps porta el client a l'empresa, per aquesta raó està correlacionada amb Total Revenue i Total Charges i Total distance charges, perquè són variables que al ser total són més altes en gent que porta molt de temps a l'empresa.

Si passem a mirar la correlació entre Total Revenue i Total Charges veiem que és de gairebé 1 així que haurem d'eliminar una, així que com ja tenim la variable Monthly Charges he decidit **eliminar Total charges**.

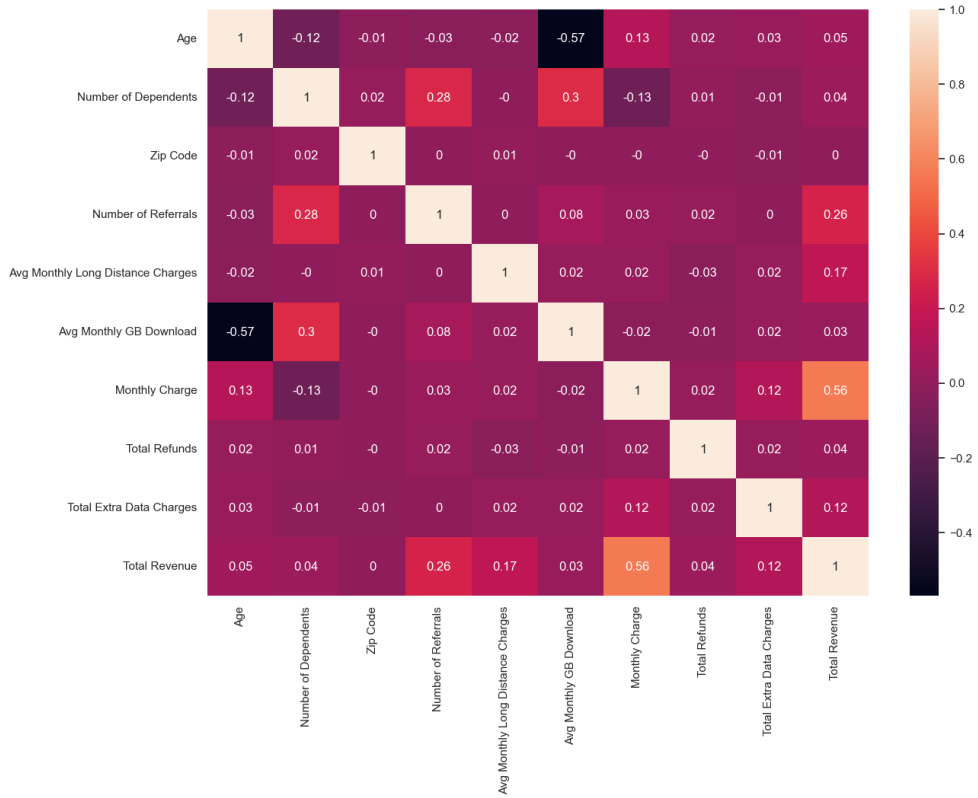
Veiem que Total Revenue segueix estant correlacionada amb Tenuer in months i Total long distance charges , amb les quals té coeficients de correlació de 0.85 i 0.78 respectivament. Així que com Total Revenue ens està aportant informació molt semblant a aquestes dues variables **eliminarem Tenure in months i Total long distance charges** i ens quedarem amb Total Revenue.

Un cop eliminades les variables esmentades obtenim el següent correlograma:



Ara podem apreciar com les variables que tenen més correlació són les de Numero Streams i Premium services que estan correlacionades amb Monthly Charge amb un coeficient de 0.66 i 0.62 respectivament. Això té sentit perquè la variable Monthly charge ens està dient quant es gasta cada més cada client en serveis de l'empresa. Donat que existeix una forta correlació amb les variables creades crec que és millor eliminar-les perquè ens aporten una informació que ja ens aporta la variable Monthly charge.

Després d'haver eliminat les 5 variables obtenim el següent correlograma:

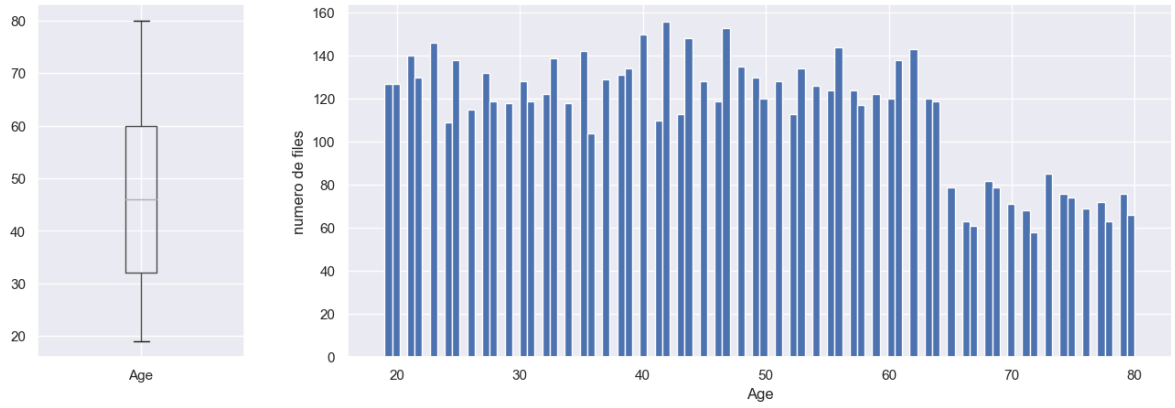


Veiem com obtenim un correlograma sense coeficients de correlació alts.

1.2.Anàlisi estadístic de les variables de manera independent.

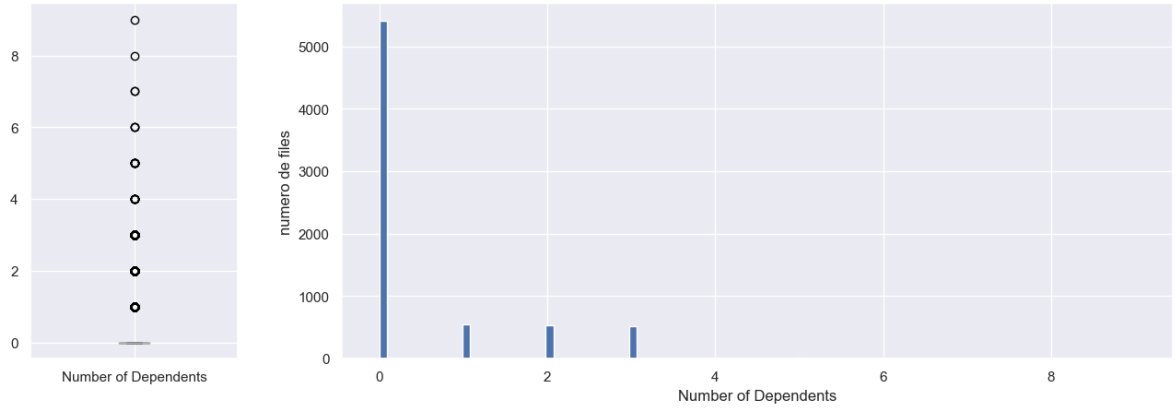
Variables numèriques:

Age:



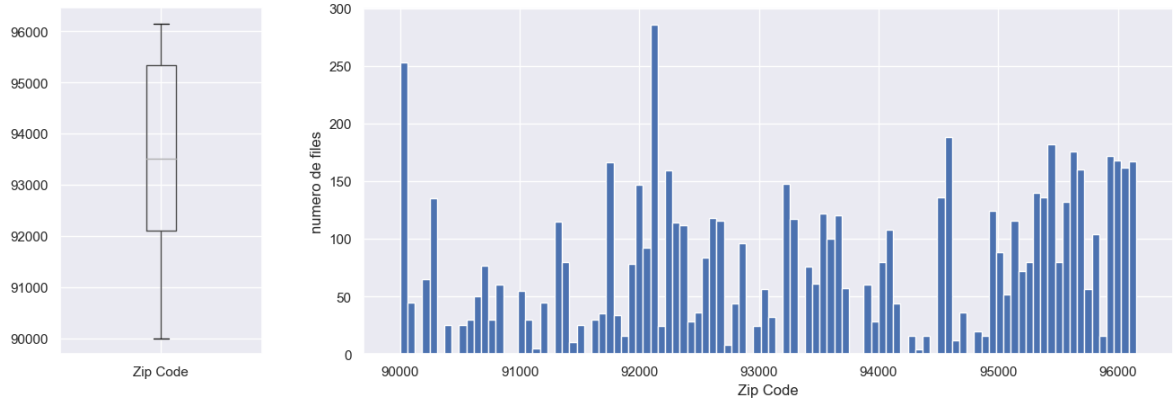
Aquesta variable segueix una distribució bastant uniforme encara que podem veure com a partir dels 65 disminueix notablement el nombre de clients. No trobem valors mancants ni outliers que a priori puguin empitjorar el comportament dels nostres futurs models. A més el rang d'aquesta variable està contingut entre 19 i 80 anys.

Number of Dependents:



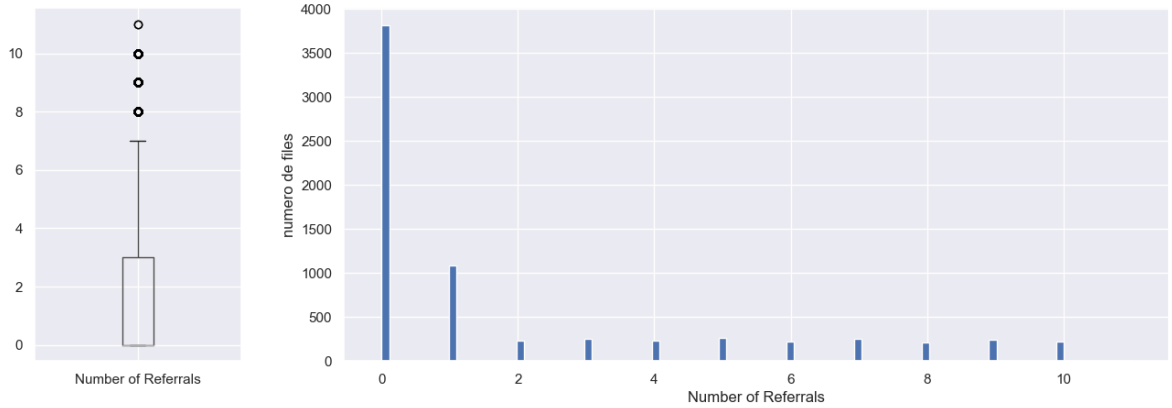
Aquesta variable segueix una distribució aparentment logarítmica on la majoria de mostres les trobem en el valor 0. No trobem presencia de valors mancants, tots els valors estan continguts entre el 0 i el 9 així que a priori no trobem valors que sobresurtin molt de la normalitat i que puguin afectar negativament el nostre model ni outliers que puguin representar errors.

Zip Code:



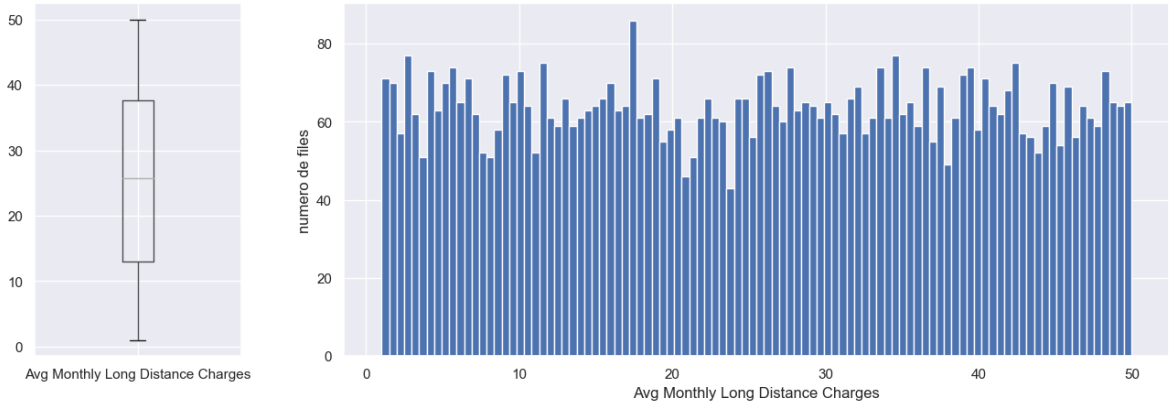
Podem veure com la distribució d'aquesta variable és molt irregular, els diferents pics que podem trobar representen zones amb molta població. No trobem valors mancants en aquesta variable ni outliers ja que tots els valors semblen estar correctes ja que estan contingut entre el 90001 i el 96150 que són Zip Codes de l'estat de California.

Number of referrals:



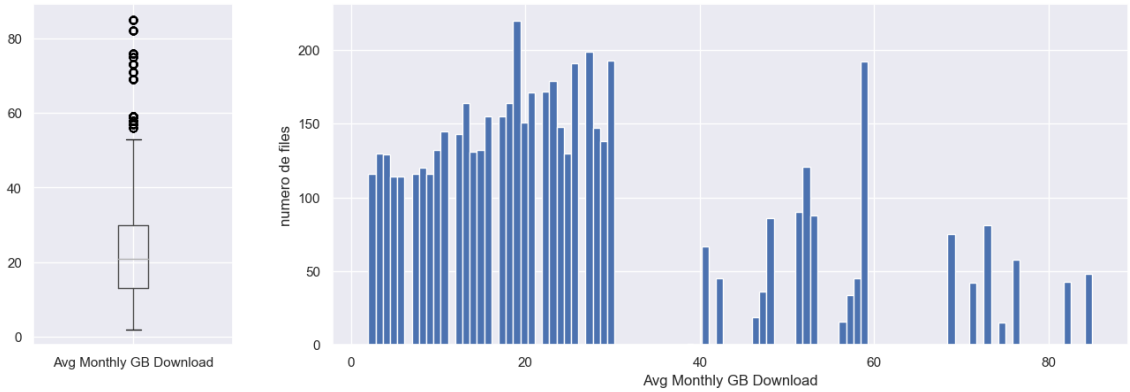
En aquesta variable tornem a veure una distribució logarítmica on la majoria de mostres s'acumulen en el valor 0 i l'1. No trobem valors mancants en aquesta variable i el rang es troba entre el 0 i l'11 així que a priori no trobem outliers que representin valors erronis.

Avg Monthly Long Distance Charges:



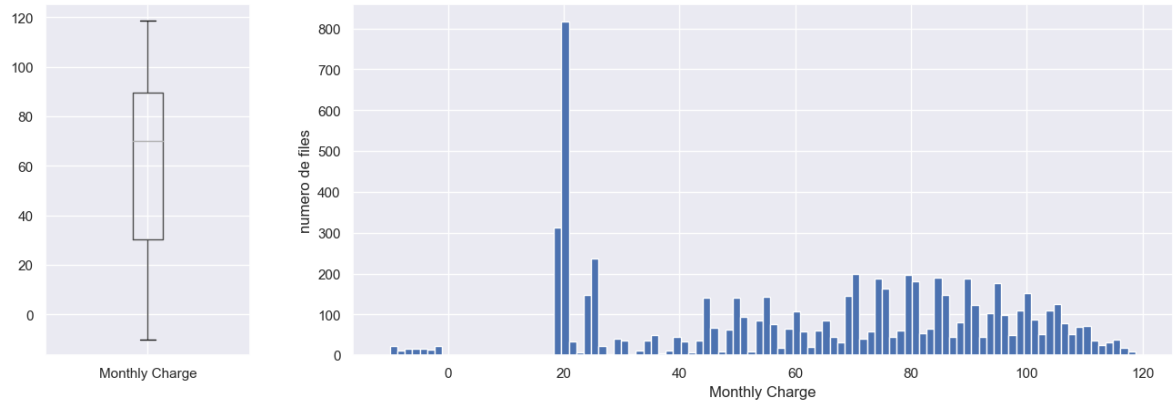
Veiem com aquesta variable segueix una distribució bastant uniforme. Pel que fa els valors mancants, aquesta variable té un total de 682 NAs. Més endavant quan acabem d'analitzar totes les variables decidirem que fer amb ells. Aquesta variable es troba entre els valors 1.01 i 49.99.

Avg Monthly GB Download:



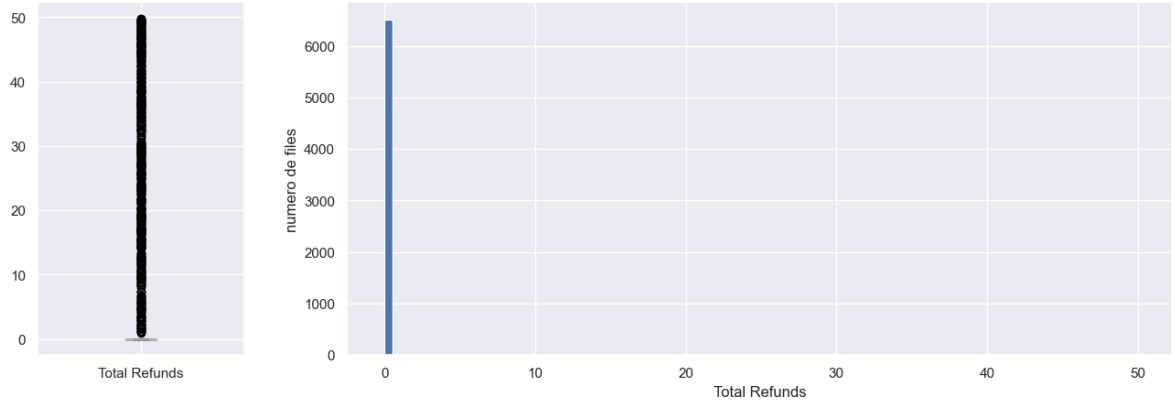
Aquesta variable té un tota de 1526 valors mancants, els quals tractarem quan acabem d'analitzar totes les variables. Pel que fa la distribució d'aquesta variable podem veure com no segueix cap distribució en concret. El rang d'aquesta variable es troba entre 2 i 85 així que en principi no trobem outliers que siguin valors erronis.

Monthly Charge:

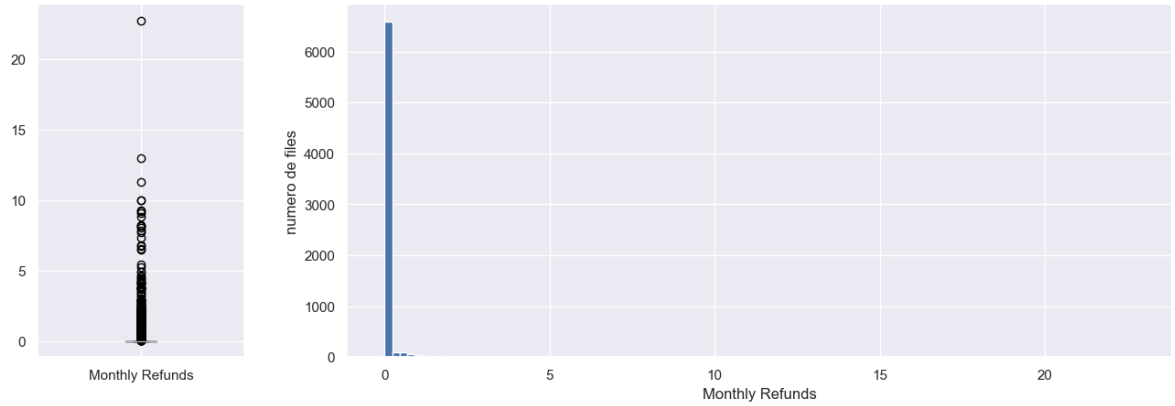


Podem veure com aquesta variable en principi no segueix cap distribució en concret. Podem veure com existeix un pic en els clients amb un monthly charge de 20 i com després fa una petita paràbola des del 40 fins el 118.75. A més, podem apreciar com existeixen una sèrie de valors negatius que podrien ser errors o casos en els que l'empresa dona diners als clients, al final d'aquest anàlisi veurem que fer amb aquests valors. El rang d'aquesta variable va des de -10 fins a 118.75. No existeixen valors mancants en aquesta variable.

Total Refunds

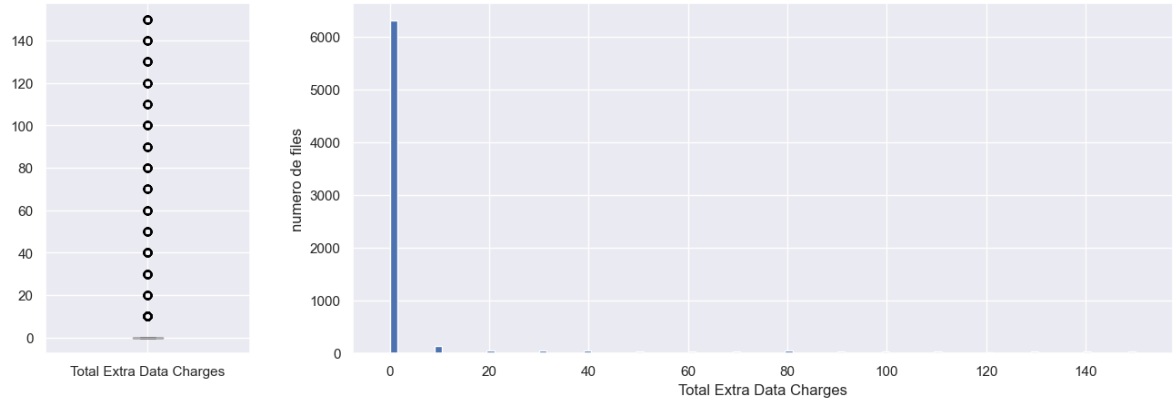


A simple vista podem veure com l'empresa no retorna gaires diners als clients ja que la majoria de valors es troben en el 0. Encara així existeixen algunes excepcions ja que encara que no s'apreciï en el gràfic el rang d'aquesta variable va de 0 a 49.79. Aquests valors que sobresurten molt de la mitja els haurem de vigilar perquè poden influir negativament en les futures prediccions del nostre model. No existeixen valors mancants en aquesta variable. Per obtenir més informació sobre aquesta variable la recodificarem dividint els valors d'aquesta per la variable 'Tenure in months' de manera que tindrem els Refunds per Month. Aquest canvi el fem perquè no es el mateix un client que li han retornat 10 euros euros en 10 mesos que un que ho ha fet en 1 mes. Resultat del canvi:

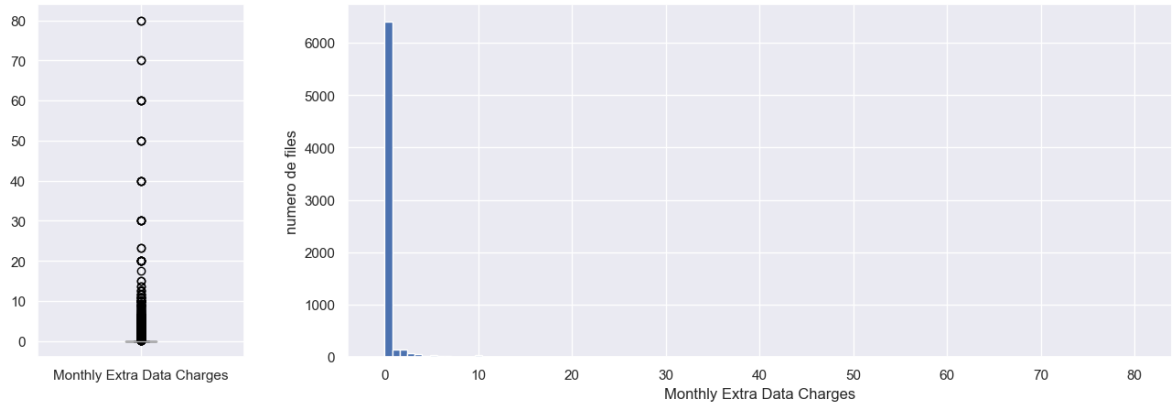


Veiem que la majoria de valors segueixen sent 0.

<u>Total Extra Data Charges:</u>

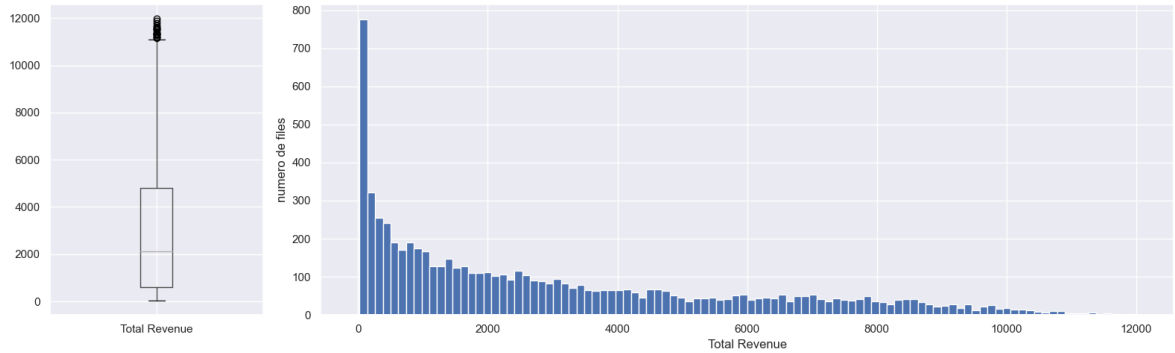


A primera vista veiem com la majoria de Extra Charges que es fan als clients sol ser 0. Encara així, el rang d'aquesta variable va de 0 a 150. Aquesta variable no conté valors mancants. Igual que passa en la variable anterior, al ser la majoria de valors propers o iguals a 0 els valors que s'allunyen molt del 0 els haurem de vigilar perquè aquests no distorsionin les prediccions del model. Per obtenir més informació sobre aquesta variable la recodificarem dividint els valors d'aquesta per la variable 'Tenure in months' de manera que tindrem els Extra Data Charges per Month. Aquest canvi el fem perquè no es el mateix un client que ha gastat 100 euros en extra charges en 10 mesos que un que ho ha fet en 1 mes. Resultat del canvi:

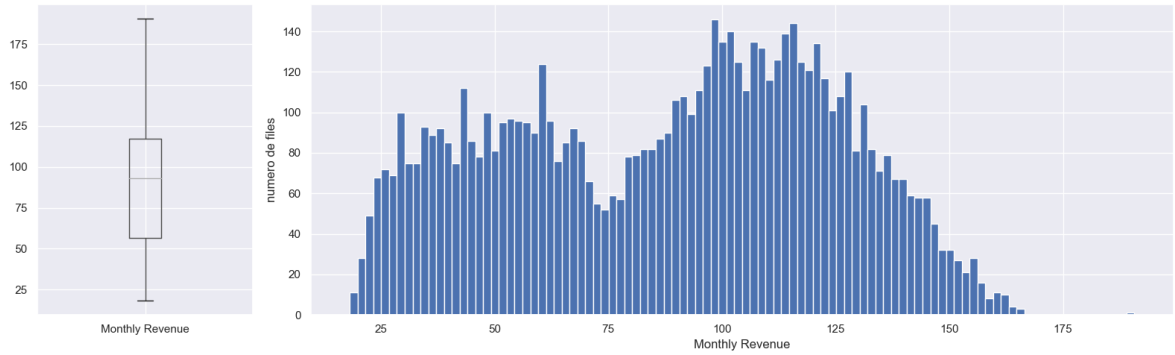


Com era d'esperar la gran majoria de valors segueixen sent 0.

Total Revenue:

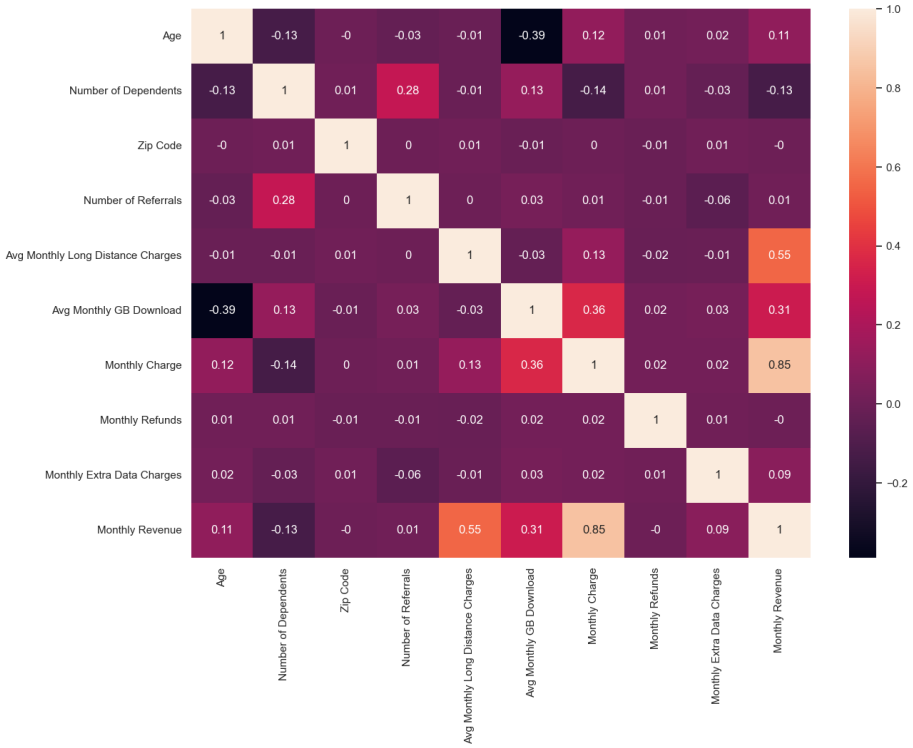


En aquesta variable podem veure com es segueix una distribució logarítmica ja que la majoria de valors són 0 i va disminuint progressivament fins arribar a 11979.34. Aquesta variable no conté valors mancants. En principi no trobem outliers que puguin tractar-se de valors erronis. Per obtenir més informació sobre aquesta variable la recodificarem dividint els valors d'aquesta per la variable 'Tenure in months' de manera que tindrem el Revenue per Month. Aquest canvi el fem perquè no es el mateix un client que ha cobrat 10000 euros en 10 mesos que un que ho ha fet en 1 mes. Resultat del canvi:

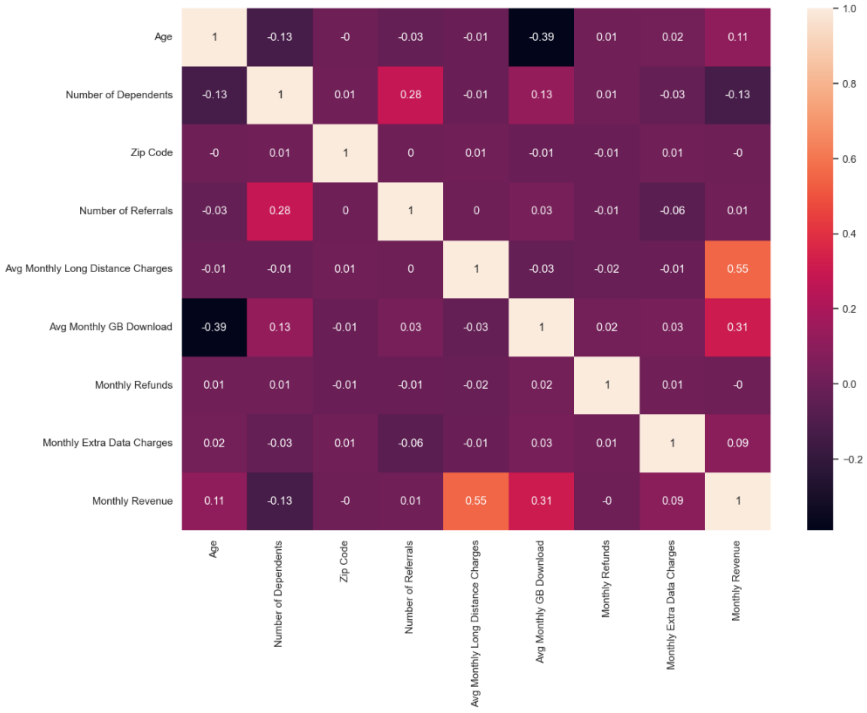


En aquesta variable podem veure un gran canvi en la distribució de la variable. Aquest canvi es deu a que ara estem obtenim

Abans d'analitzar les variables categòriques anem a veure quines són les correlacions que hi ha entre variables numèriques amb les noves variables monthly que hem creat:



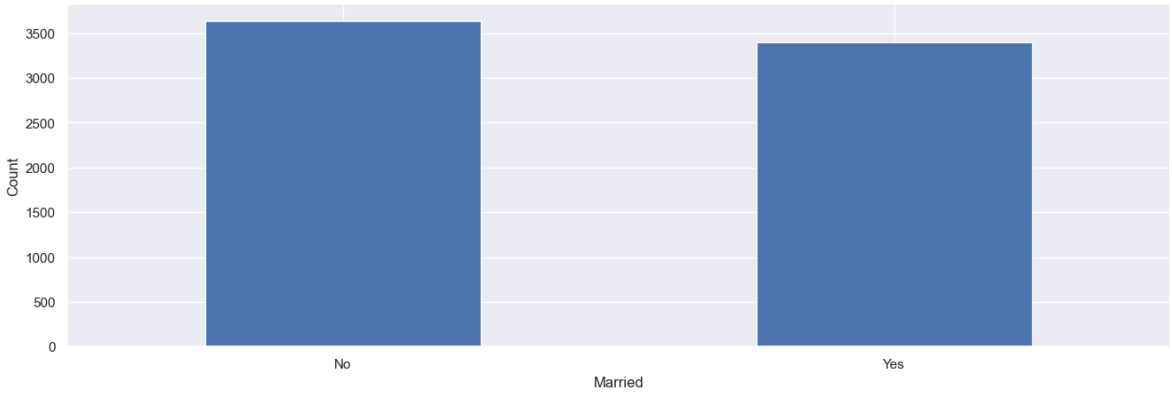
Veiem com la correlació entre Monthly Charge i Monthly Revenue és molt alta així que procedim a eliminar Monthly Charge perquè Monthly Revenue esta formada per la suma de moltes variables diferents que ens aporten més informació. El resultat d'eliminar la variable és el següent:



Podem veure que ja no existeixen correlacions molt altes. Recordem que ja no cal tractar els valors negatius que tenia Monthly Charge perquè hem eliminat la variable.

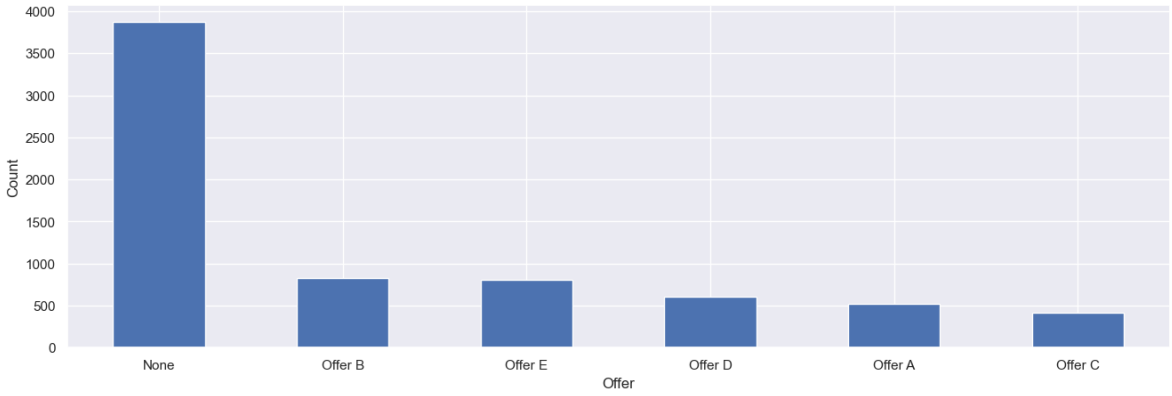
Variables categòriques:

Married:



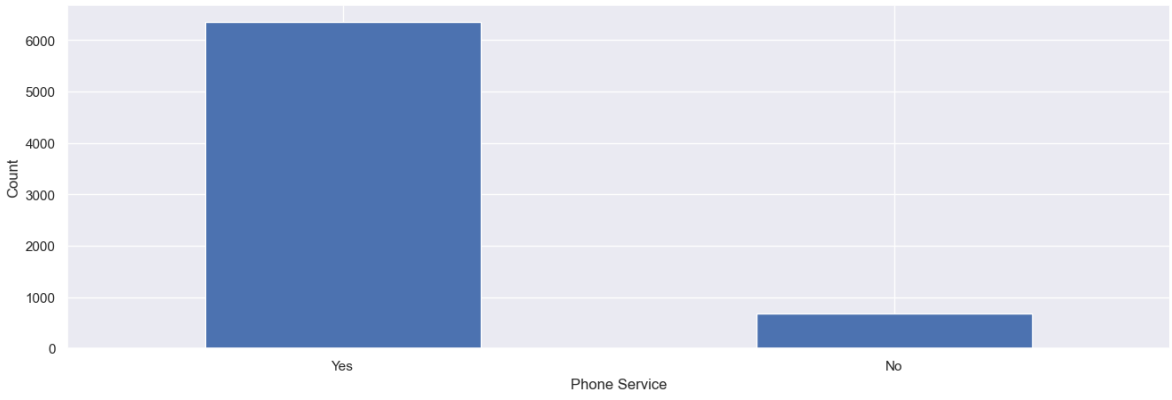
Variable dicotòmica que pren valor de si o no, no conté valors mancants.

Offer:



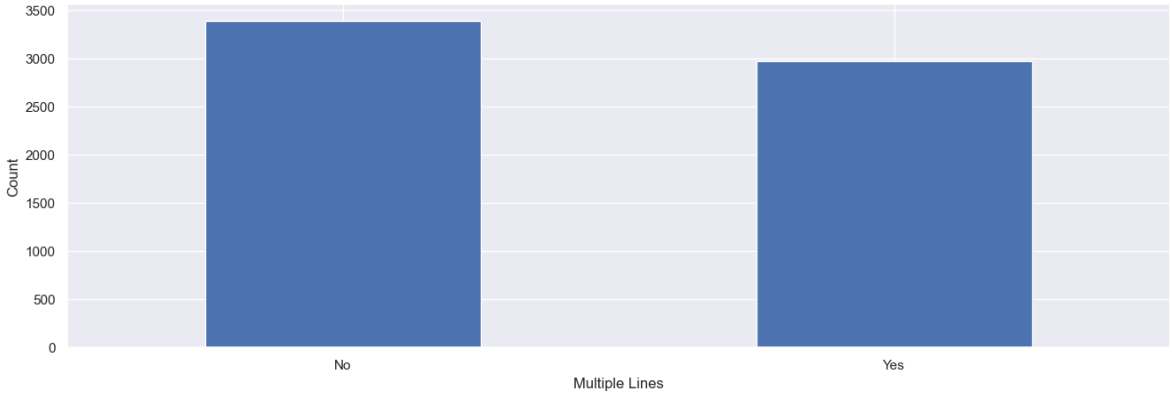
Variable que pot prendre el valor de None, Offer A, B, C, D o E. No conté valors mancants.

Phone Service:

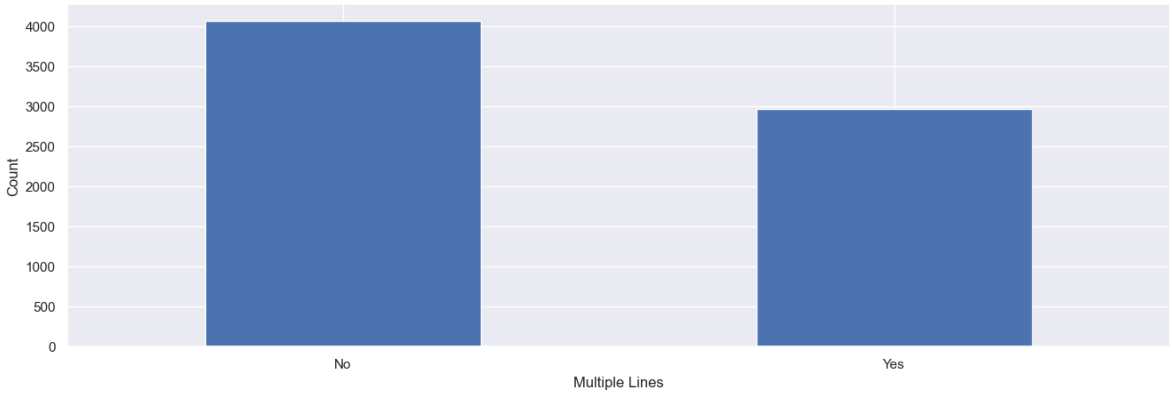


Variable dicotòmica que pot prendre valor de si o no, no conté valors mancants.

Multiple Lines:

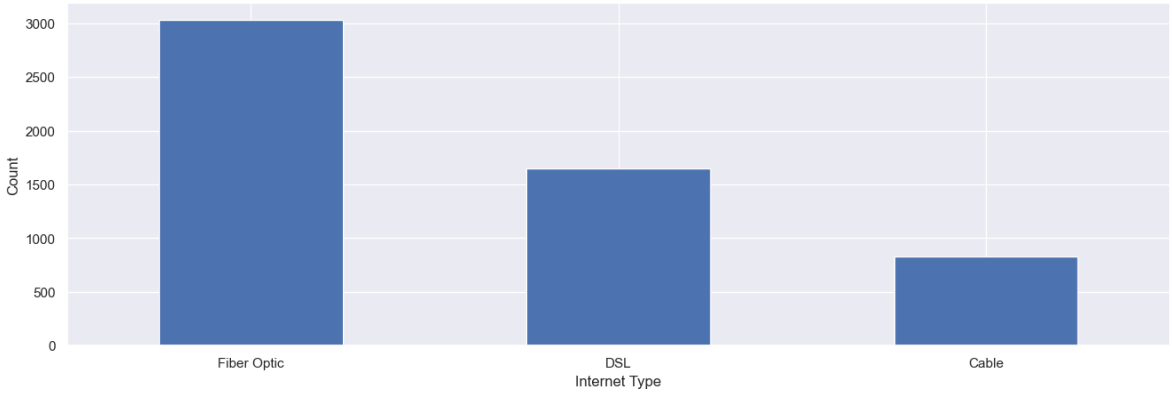


Aquesta variable pot prendre valor de si o no i conté un total de 682 valors mancants que si analitzem corresponen tots a aquells clients que no tenen Phone Service contractat. Per solucionar-ho substituirem els 682 valors mancants per la categoria NO ja que aquests no disposen de múltiples línies. Com a resultat obtenim el següent:

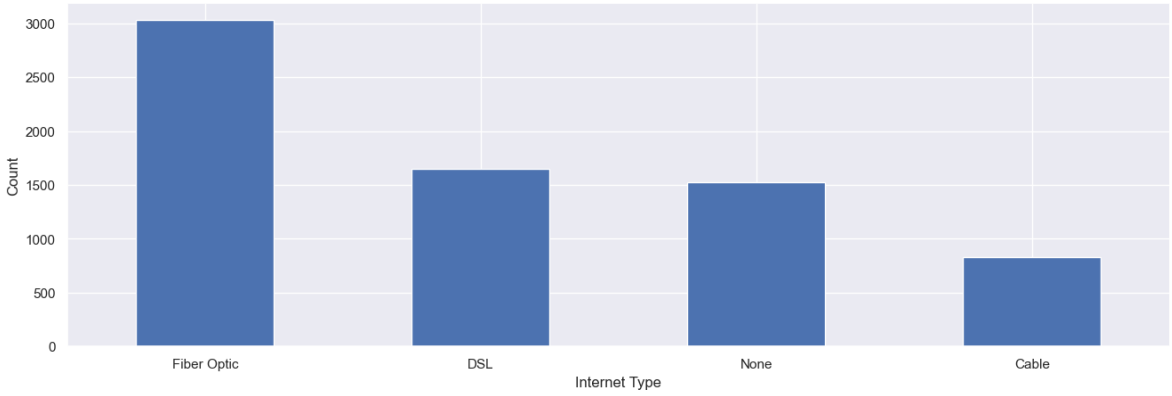


Podem veure com ha augmentat el nombre de No, ja no tenim valors mancants.

Internet Type:

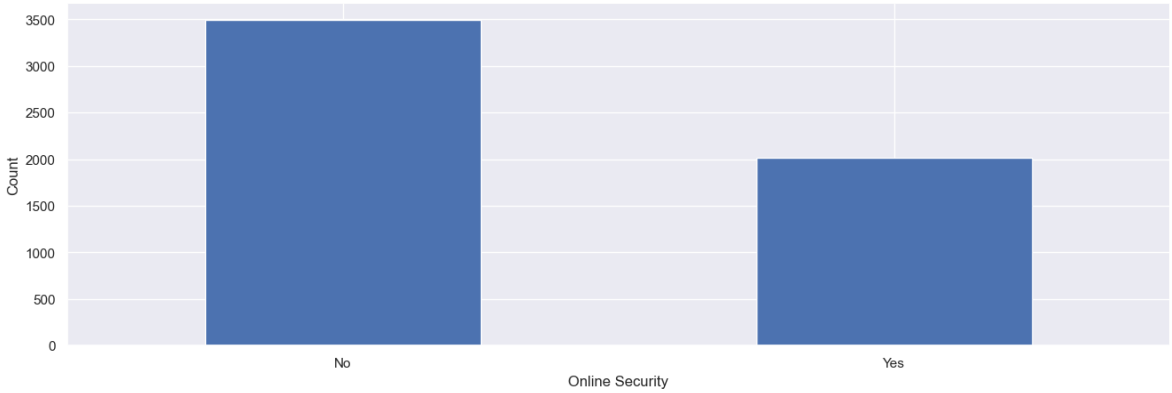


Aquesta variable pot prendre el valor de Fibra òptica, DSL o Cable. Conté un total de 1526 valors mancants. Si passem a analitzar quins són aquests valors mancants, ens trobarem amb que aquests NAs corresponen a aquells clients que no tenen internet contractat així que per solucionar aquest problema on trobem un NA posarem la categoria 'None'. Quedarà de la següent manera:

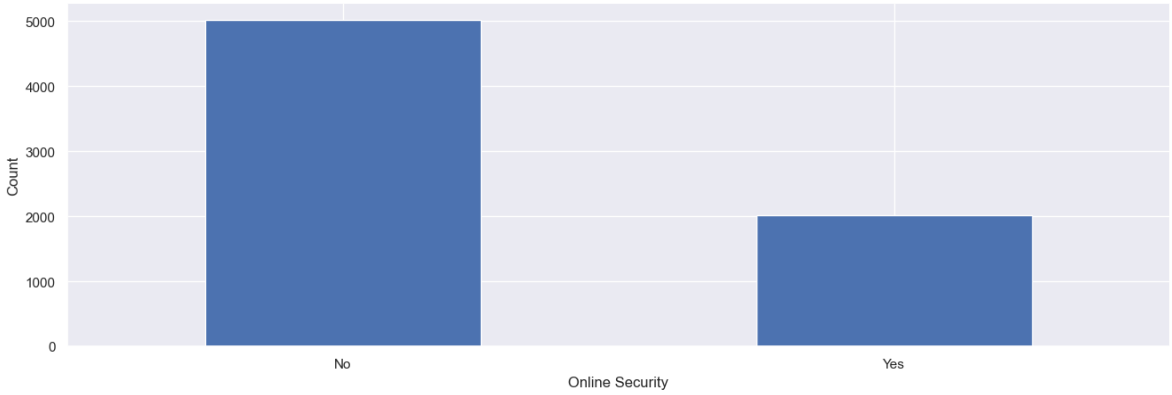


Veiem com hem creat una nova categoria que es None i que representa aquells clients que no tenen ''internet service contractat.

<u>Online Security:</u>

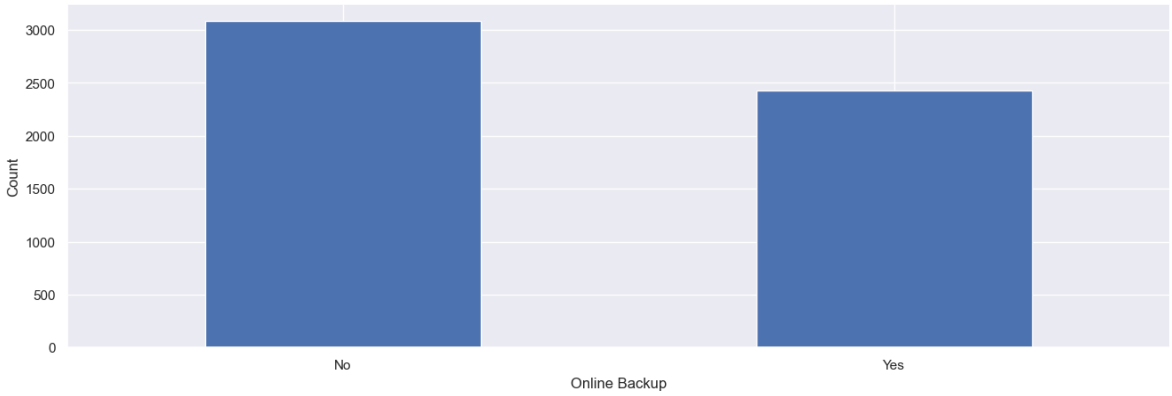


Aquesta variable es tracta d'una variable dicotòmica que pren valor de si o no i que té un total de 1526 valors mancants. Si analitzem els valors mancants tornem a veure que hi ha NA quan el client no disposa del servei d'internet, així que substituirem els NA per 'No'. El resultat és el següent:

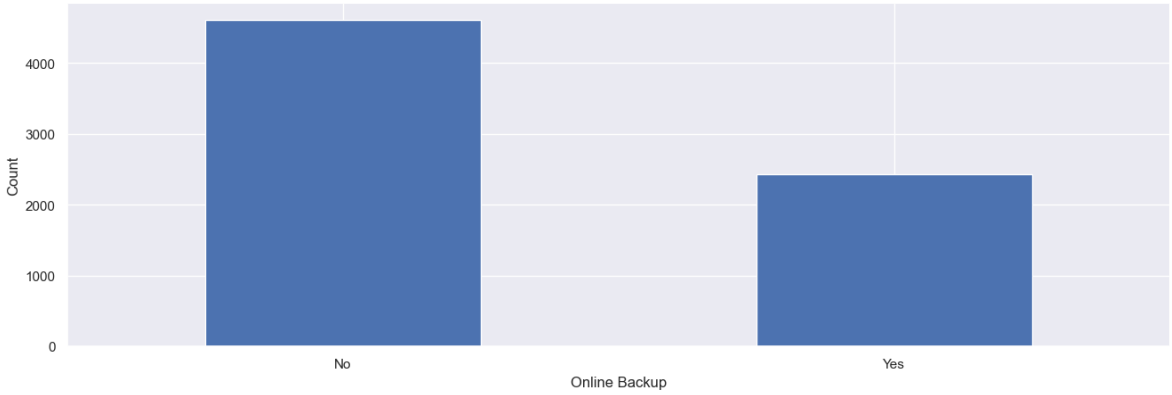


Podem veure com ha augmentat la proporció de No.

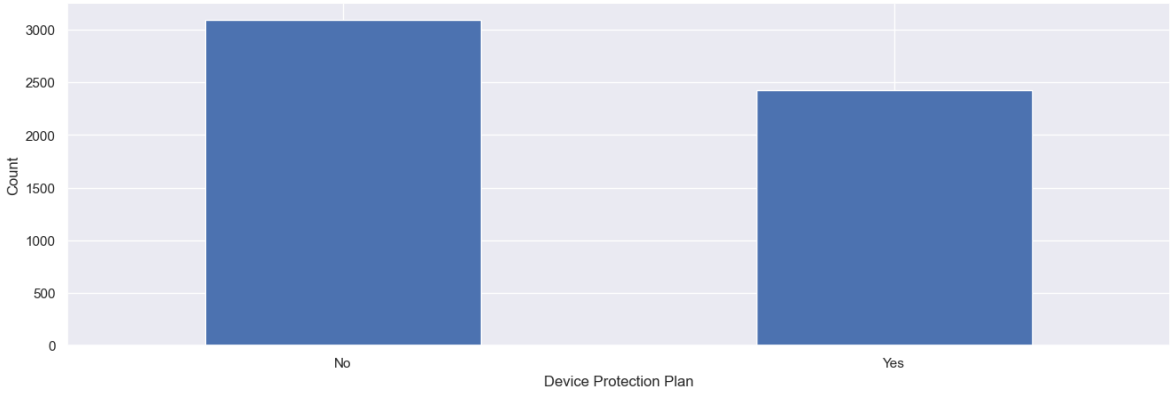
Online Backup:



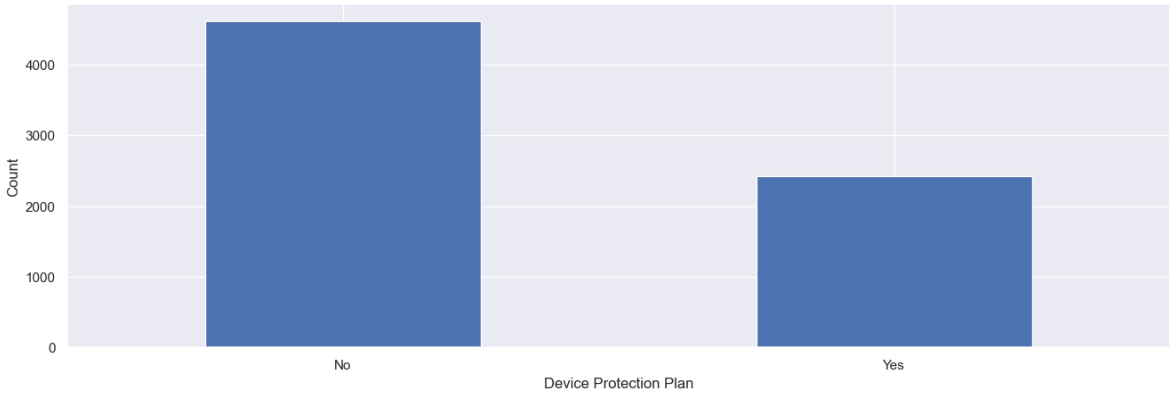
Tornem a veure una variable dicotòmica de si o no amb 1526 Na als que els hi passa el mateix, hi ha NA en els clients que no tenen contractat el Internet Service així que substituïm per 'No' i obtenim el següent:



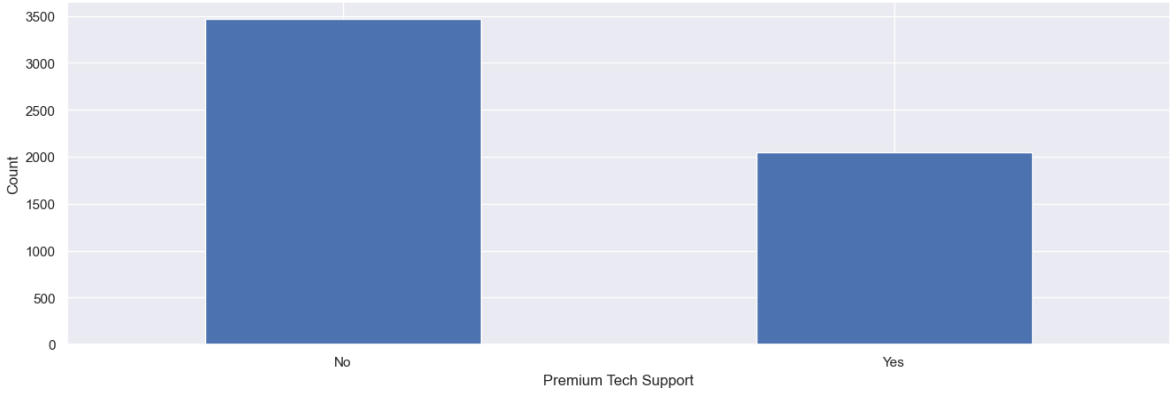
Device Protection Plan:



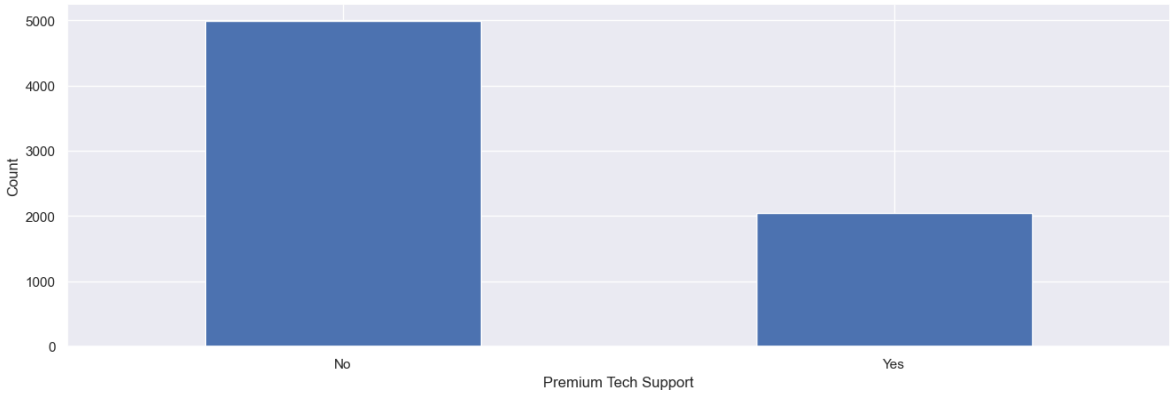
Es torna a donar el cas anterior, tenim una variable dicotòmica de si o no i 1526 NA els quals coincideixen amb aquells clients que no tenen contractat el servei d'internet. Si reemplacem els NA per la categoria 'No' obtenim el següent:



Premiem Tech Support:

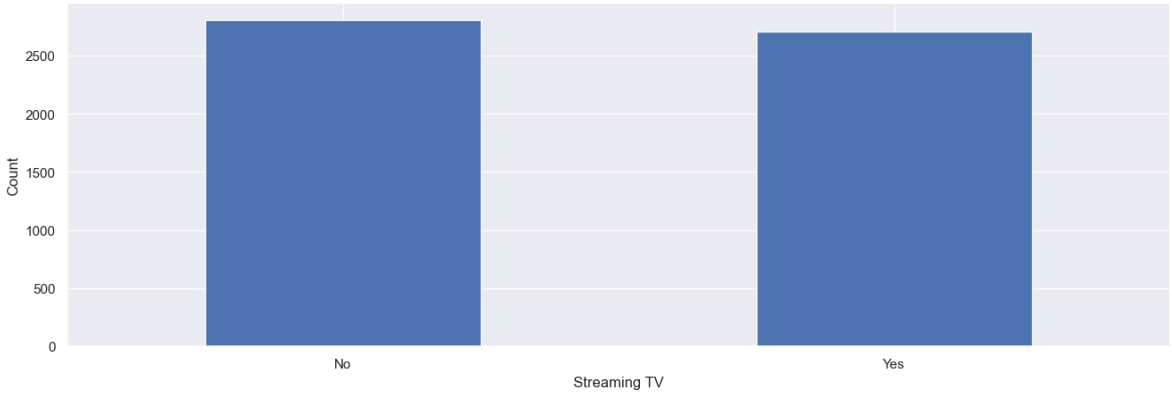


Ens tornem a trobar amb una variable dicotòmica de si o no on hi ha 1526 Na que coincideixen amb aquells clients que no tenen internet service així que els substituirem per la categoria 'No' i obtindrem el següent:

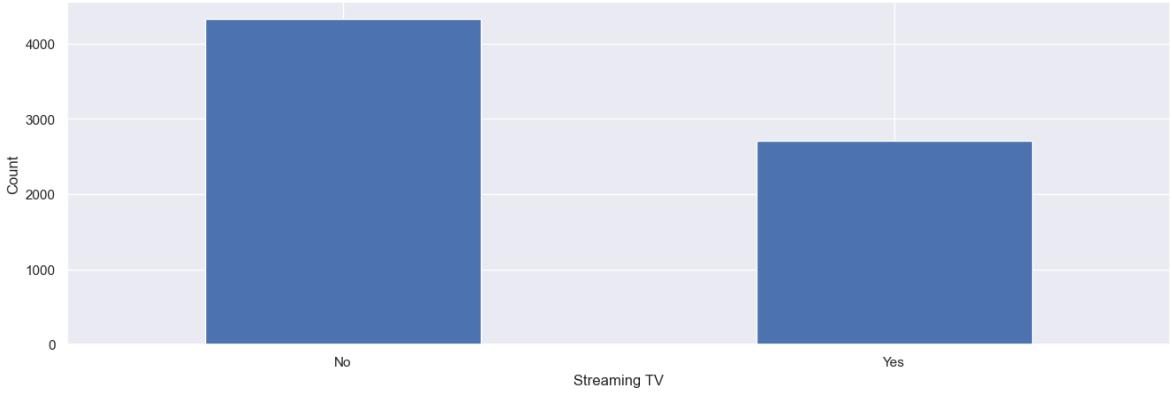


Podem veure com augmenta la proporció de No.

Streaming TV:

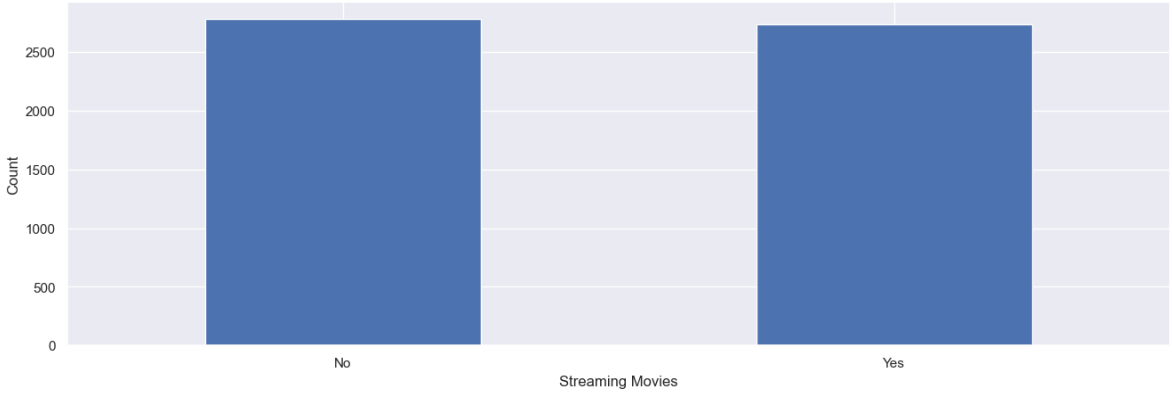


Torna a succeir el mateix que amb la variable anterior, el resultat de substituir els NA per 'No' és el següent:

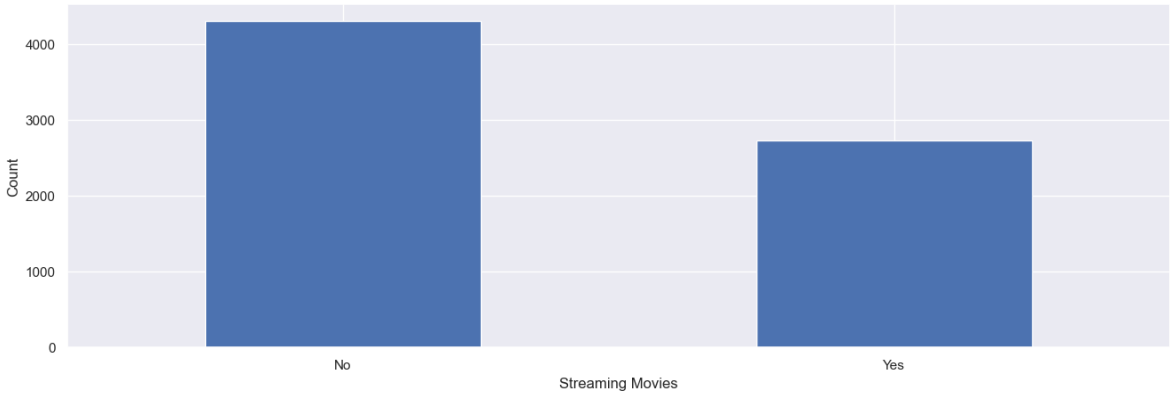


Podem veure com augmenta la proporció de No.

Streaming Movies:

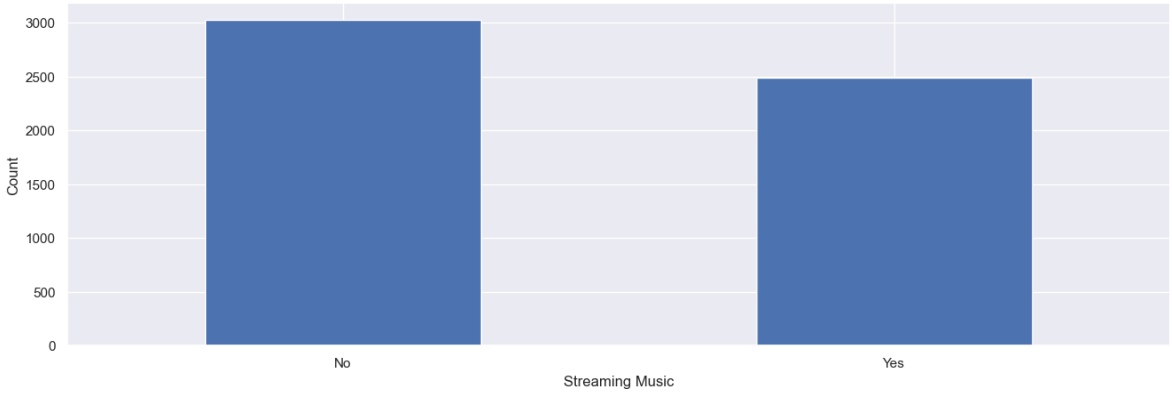


Torna a succeir el mateix que amb la variable anterior, el resultat de substituir els NA per 'No' és el següent:

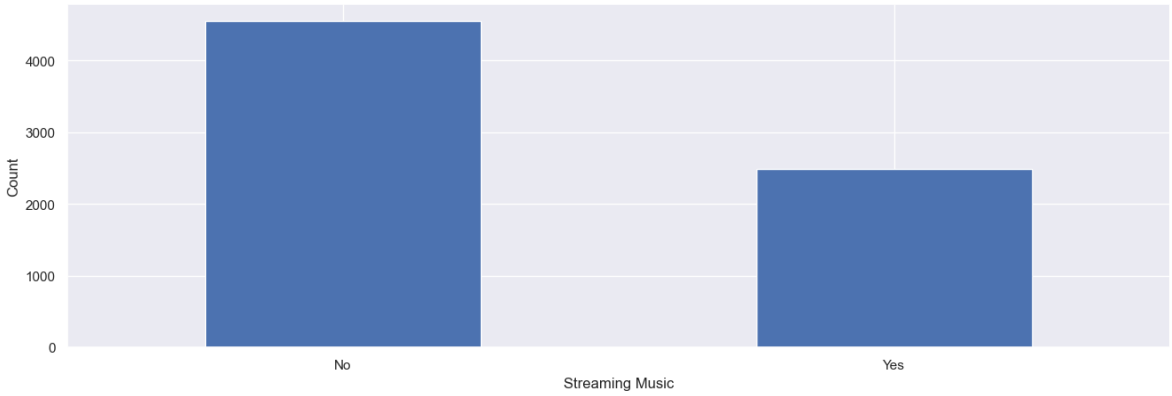


Podem veure com augmenta la proporció de No.

Streaming Music:

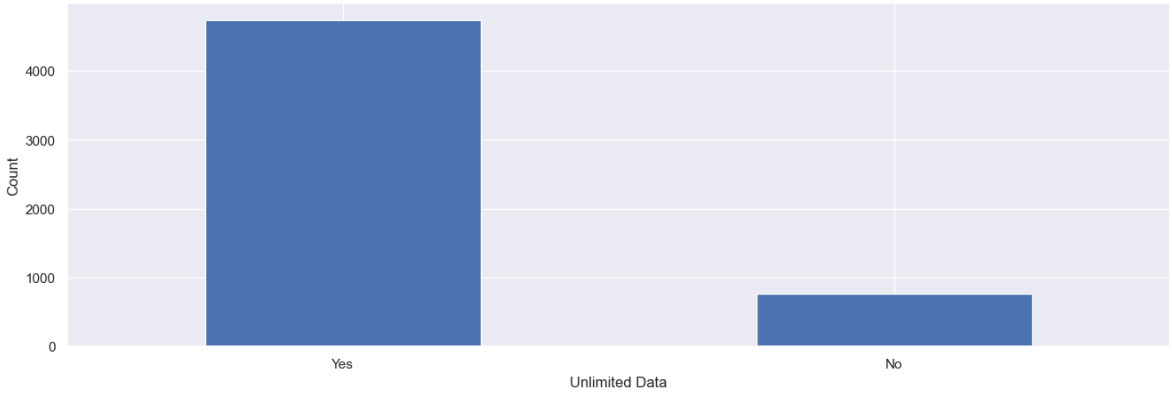


Torna a succeir el mateix que amb la variable anterior, el resultat de substituir els NA per 'No' és el següent:

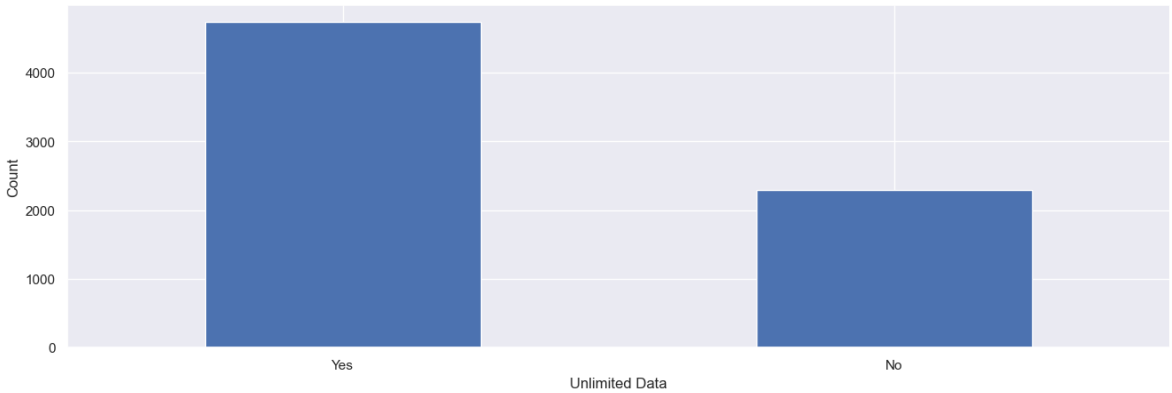


Podem veure com augmenta la proporció de No.

Unlimited Data:

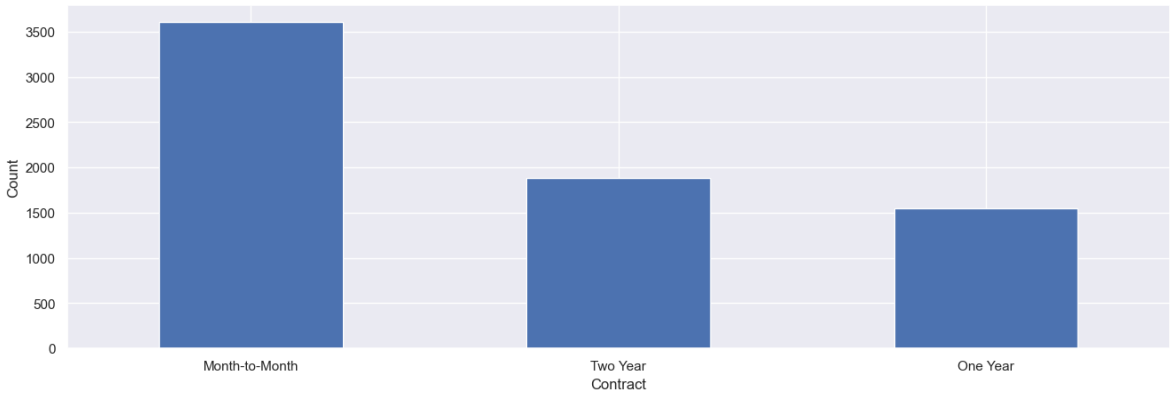


Torna a succeir el mateix que amb la variable anterior, el resultat de substituir els NA per 'No' és el següent:



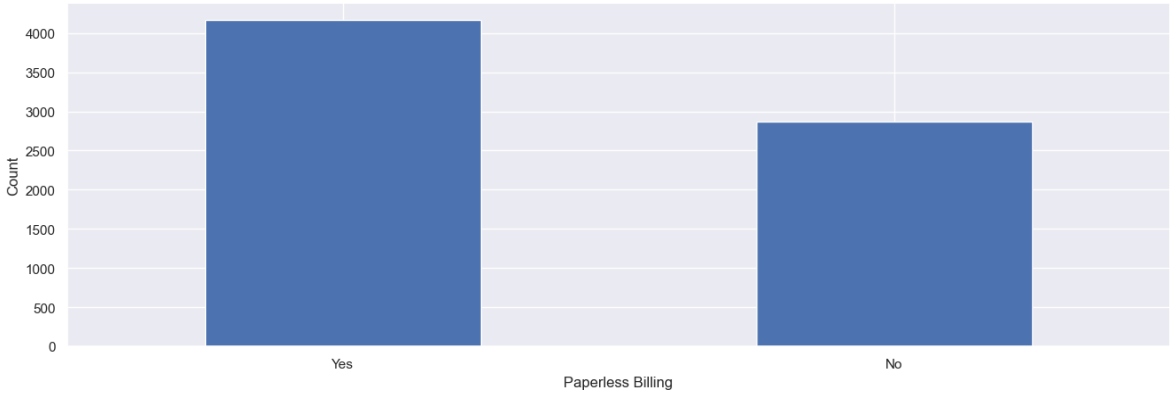
Podem veure com augmenta la proporció de No.

Contract:



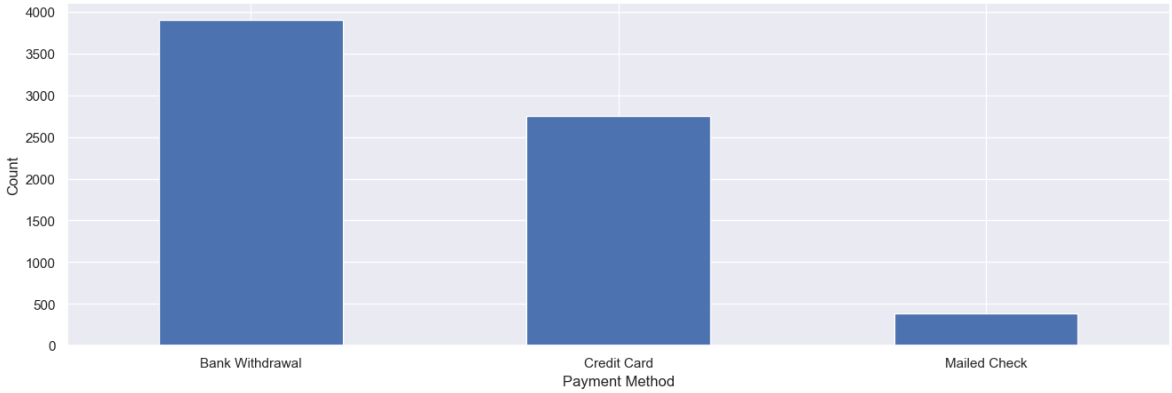
Aquesta variable té 3 modalitats de contracte: Month-to-Month, Two Year i One Year. La modalitat que més clients tenen és la month-to-month. Pel que fa els valors mancants, aquesta variable no en té cap.

Paperless billing:



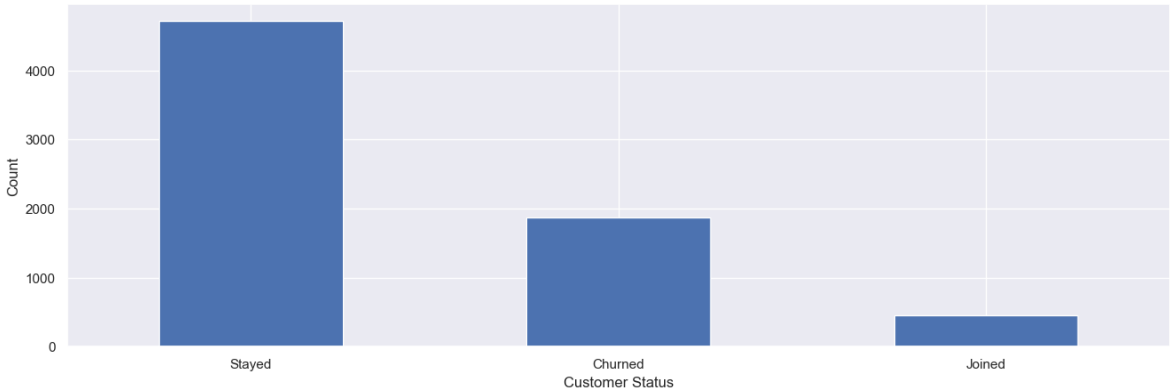
Es tracta d'una variable dicotòmica on predomina el Si respecte el No. Aquesta variable no conté dades mancants.

Payment Method:



Aquesta variable té 3 categories diferents pel que fa la forma de pagar la factura el client: Bank Bank Withdrawal (predominant), Credit Card i Mailed Check. Aquesta variable no té valors mancants.

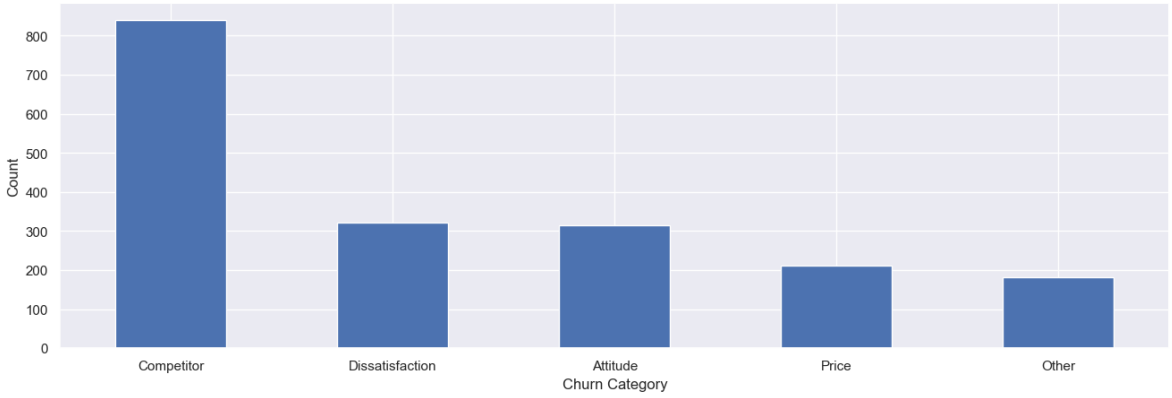
Customer Status:



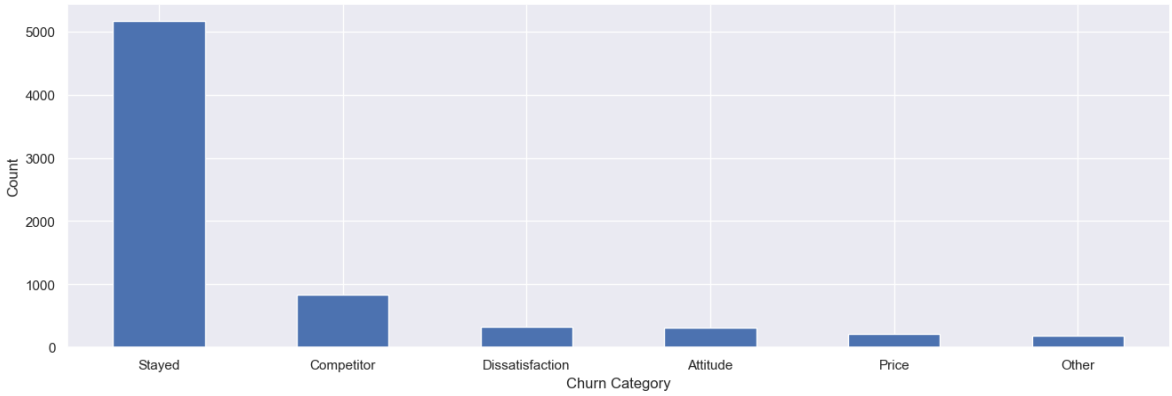
Aquesta variable és la variable objectiu, més endavant l'estudiarem amb més profunditat però de moment podem dir que té 3 categories: Stayed, Churned i Joined. En el nostre problema només ens

interessa saber si el client es queda o marxa per tant amb la categoria Joined el que farem és eliminar-la perquè aquells clients que s'acaben d'unir no ens aportaran informació útil als models.

Churn Category:

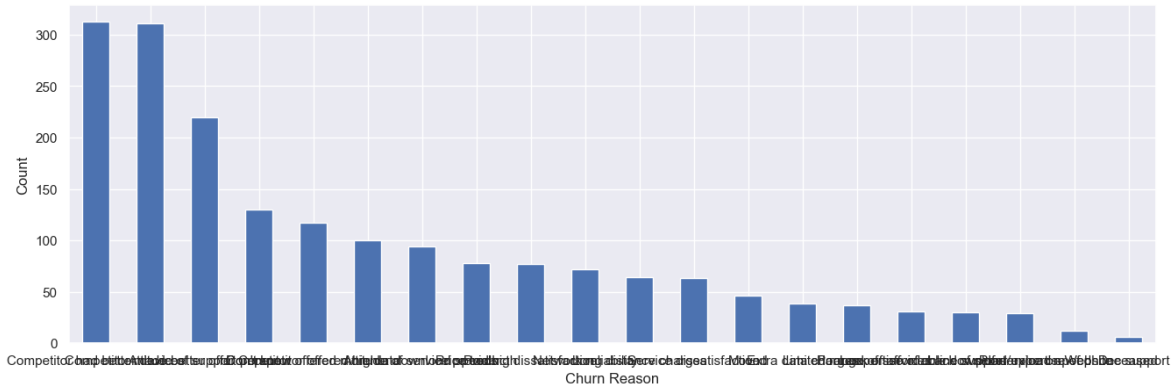


Aquesta variable té 5174 valors mancants perquè tots aquells clients de la categoria Stayed o Joined no tenen una categoria assignada, per això crearem una nova categoria en aquesta variable que es digui 'stayed'. El resultat és el següent:

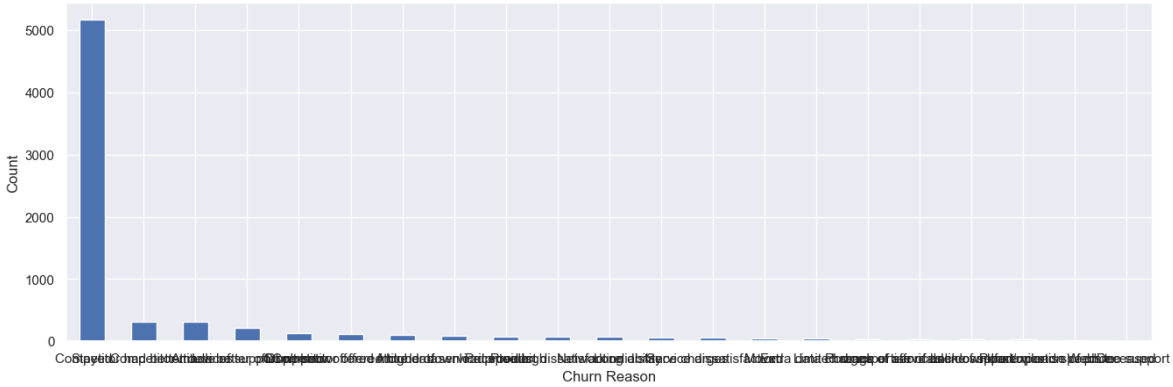


Veiem com la majoria de valors son Stayed. Cal tenir en compte que aquesta variable no la podrem tenir en compte a l'hora d'entrenar el model perquè ja dius si un client ha marxat o no.

Churn Reason:



Veiem com aquesta variable categòrica té moltes categories diferents i això farà que sigui més difícil d'utilitzar. A més aquesta variable ens està dient si un client a marxat o no per tant no la podrem utilitzar a l'hora d'entrenar models. Pel que fa els valors mancants aquesta en té els mateixos que churn category per tant els tornarem a substituir per 'Stayed'. Resultat:

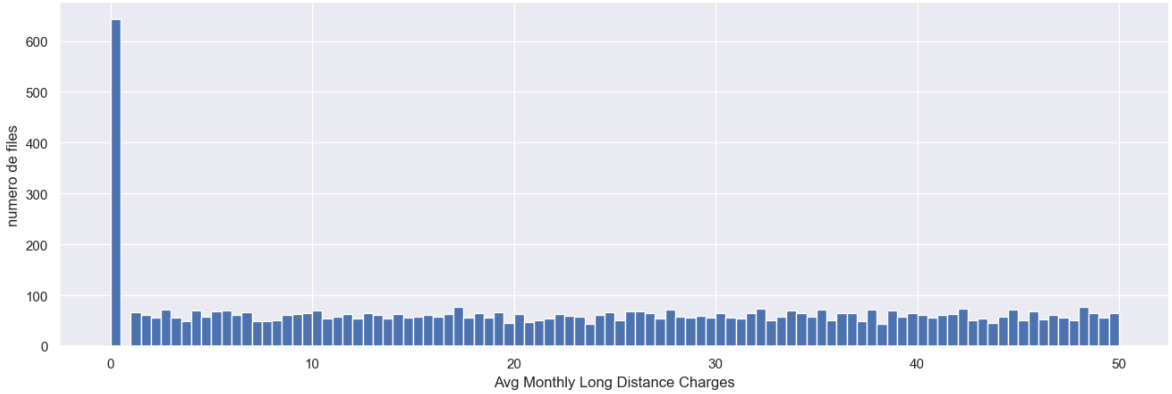


Veiem el que esperàvem, la majoria de clients és queden.

1.3. Tractem Missings a les Variables Numèriques

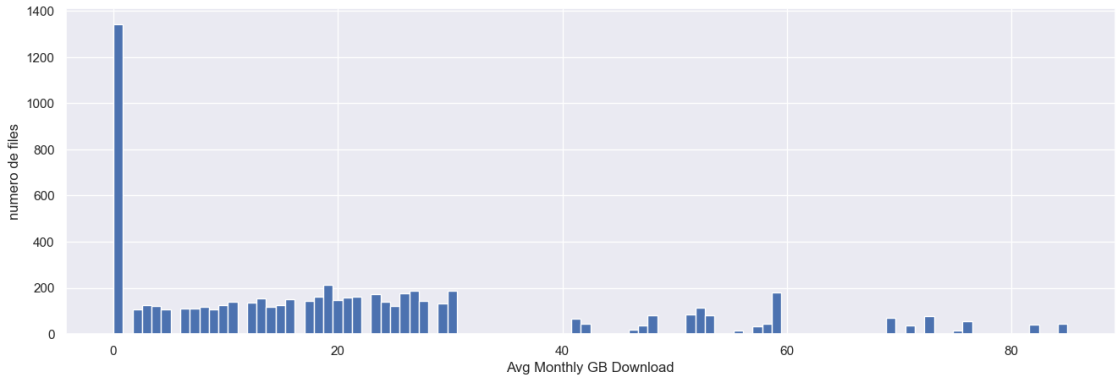
Durant l'anàlisi individual de les variables categòriques ja hem reemplaçat els valors mancants per els seus valors corresponents. Ara anem a reemplaçar els valors mancants d'aquelles variables numèriques que hem identificat a l'anàlisi individual d'aquestes. Recordem que les úniques variables numèriques on hem trobat valors mancants són: Avg Monthly Long Distance Charges i Avg Monthly GB Download.

Primer analitzarem **Avg Monthly Long Distance Charges**, si mirem el document de metadata podem veure com aquest mateix ens diu que aquesta variable tindrà 0 en tots aquells clients que no tinguin contractat el Phone Service. Si mirem la base de dades podrem veure com Tots aquells clients que no tenen phone service tenen missings a la variable Avg Monthly Long Distance Charges, per tant substituirem tots aquests missings per 0 tal i com ens diu la metadata. El resultat és el següent:



Podem veure com sobresurten tots aquells missings que hem substituït per 0.

Ara passarem a analitzar **Avg Monthly GB Download,** si mirem el document de metadata com hem fet anteriorment veiem que en aquest cas els valors mancants corresponen a aquells clients que no tenen el servei d'internet contractat i que per tant han descarregat 0 GB. Tal i com diu al metadata aquests clients han de tenir 0 en aquesta columna, per tant substituirem tots aquests valors mancants per 0.



Veiem com ara sobresurten tots aquells clients amb 0 GB descarregats.

1.4.Anàlisi de riscos i biaixos de les variables

Durant l'anàlisi que hem anat fent hem eliminat diferents variables. En aquest apartat farem una última reflexió sobre si les variables que hem decidit utilitzar poden presentar riscos o biaixos. Anem a veure quines variables tenim ara mateix:

```
0   Age
1   Married
2   Number of Dependents
3   Zip Code
4   Number of Referrals
5   Offer
6   Phone Service
7   Avg Monthly Long Distance Charges
8   Multiple Lines
9   Internet Type
10  Avg Monthly GB Download
11  Online Security
12  Online Backup
13  Device Protection Plan
14  Premium Tech Support
15  Streaming TV
16  Streaming Movies
17  Streaming Music
18  Unlimited Data
19  Contract
20  Paperless Billing
21  Payment Method
22  Customer Status
23  Churn Category
24  Churn Reason
25  Monthly Refunds
26  Monthly Extra Data Charges
27  Monthly Revenue
```

Crec que de les variables que utilitzem pot presentar risc utilitzar zip code, ja que aquesta variable no ens indica amb claredat la proximitat entre els clients i per tant no sabem si el model treballarà bé amb aquesta variable, a més zip code ens aporta una informació molt específica de cada client.

Per aquesta raó vigilarem la variable i decidirem que fer amb ella quan estiguem creant els models. Pel que fa les variables Churn Category i Churn Reason les haurem de treure a l'hora de fer els models perquè ens estan dient si un client ha marxat o no.

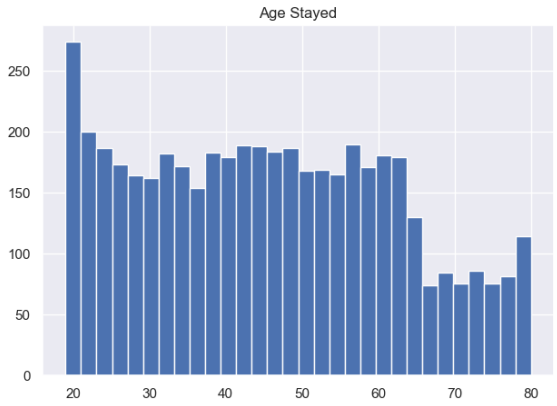
2. User profiling

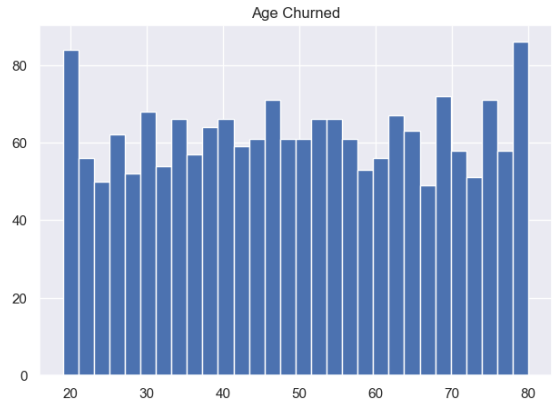
Per realitzar aquesta part del treball crearem dues bases de dades derivades de la general. Un d'aquests nous dataframes estarà composat per tots aquells clients que han marxat de l'empresa, en canvi l'altre estarà format per tots aquells clients que s'han quedat a l'empresa. El nostre objectiu és entendre quins clients són més propensos a marxar de l'empresa i quins tenen més tendència a quedar-se.

En la següent secció, us presentarem un anàlisis detallat de les característiques dels clients en ambdues bases de dades i identificarem les característiques que siguin més rellevants alhora de determinar si un client marxa o es queda.

Variables numèriques:

Age





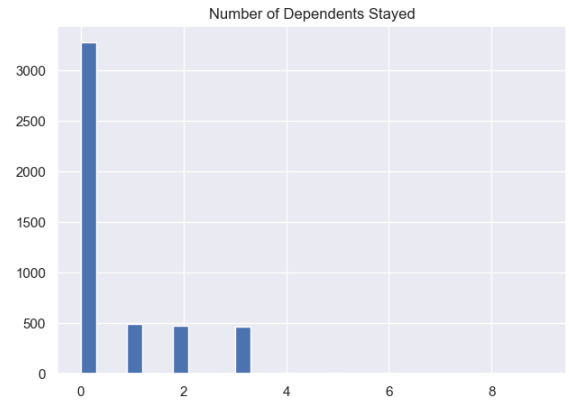

```
Stayed Age :

count    4720.000
mean       45.582
std        16.383
min        19.000
25%        32.000
50%        45.000
75%        58.000
max        80.000
```

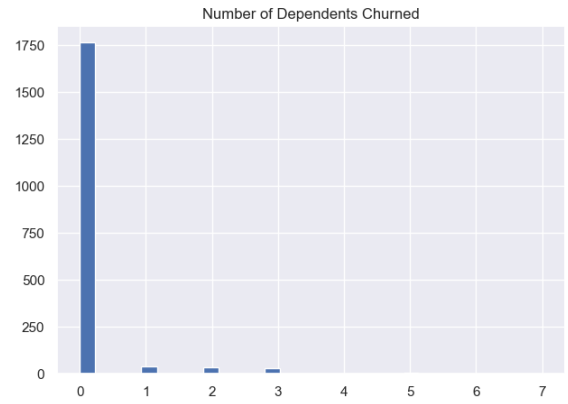
```
Churned Age :

count    1869.000
mean       49.736
std        17.605
min        19.000
25%        35.000
50%        50.000
75%        65.000
max        80.000
```

En els gràfics i estadístiques podem veure el comportament tant de la distribució com d'alguns aspectes estadístics de la variable edat. Si ens fixem en les distribucions podem veure que per norma general tant el nombre de clients que es mantenen a l'empresa com el nombre dels que marxen

tenen una distribució bastant uniforme de l'edat. L'únic aspecte a destacar és que la proporció dels clients majors de 65 anys que es mantenen a l'empresa és menor respecte la proporció de joves. En canvi en els clients que marxen podem veure com els clients amb més de 65 mantenen i inclús augmenten una mica la proporció respecte els més joves que marxen. Això ens dona a entendre que **aquells clients majors de 65 anys tenen més tendència a marxar** de l'empresa que a mantenir-se. Aquest fet també es pot veure reflexat en les estadístiques on es pot veure que la mitjana d'edat és major en clients que marxen. Les causes segurament són jubilacions dels clients ja que 65 anys és l'edat marcada per la jubilació.

Number of dependents



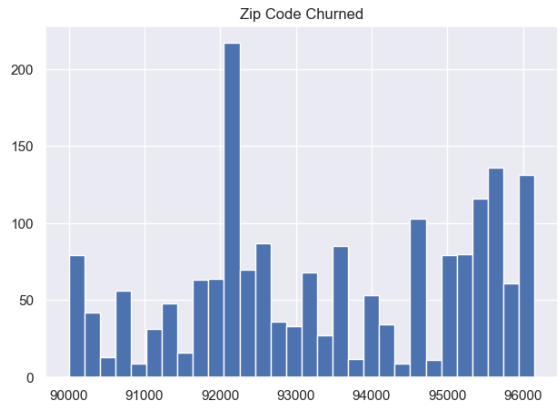



```
Stayed Number of Dependents :

count    4720.000
mean        0.618
std         1.058
min         0.000
25%         0.000
50%         0.000
75%         1.000
max         9.000
```

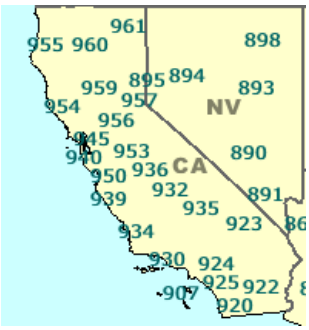
```
Churned Number of Dependents :

count    1869.000
mean        0.118
std         0.550
min         0.000
25%         0.000
50%         0.000
75%         0.000
max         7.000
```

En els gràfics i estadístiques podem veure la diferència entre el nombre de persones que viuen amb els clients que s'han quedat a l'empresa i els que han marxat. A primera vista podem veure clarament com la proporció de clients que es queden que viuen amb 1, 2 o 3 'dependents' és bastant major que la dels clients que marxen que viuen amb 1, 2 o 3 'dependents', això també és pot veure en les estadístiques on la mitjana de nombre de 'dependents' en clients que és queden és aproximadament 5 cops major que la mitjana dels que marxen. Aquest fet pot venir donat per que **els clients que viuen amb menys familiars tenen menys responsabilitats econòmiques i tenen més llibertat a l'hora de marxar**.

Zip Code

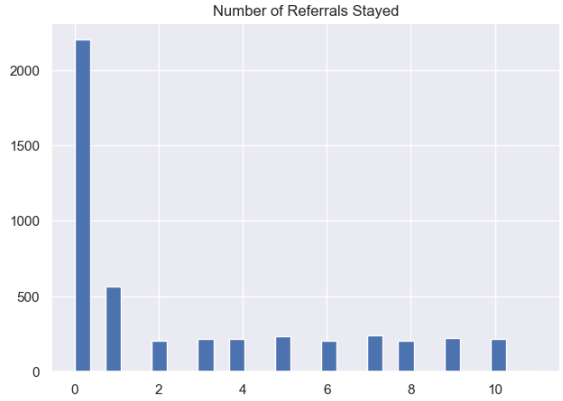


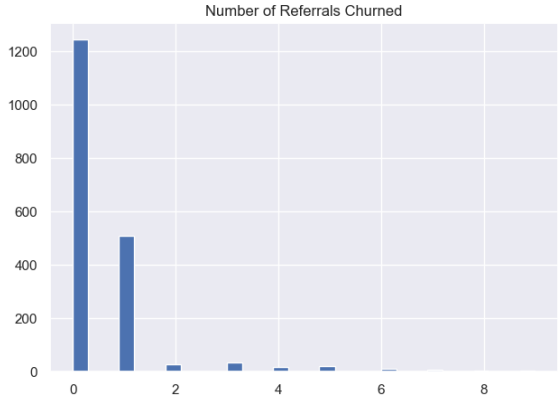
En les distribucions podem veure la diferencia dels Zip codes dels clients que es queden i els que marxen. Si analitzem aquestes distribucions i les comparem podem veure 2 aspectes. En primer lloc, la proporció de clients que és queden amb zip codes entre 95000 i 96000 és major que els que marxen. En segon lloc, podem veure com en els zip code propers a 92000 hi ha un gran augment de proporció dels clients que marxen respecte els que es queden. Per tant, podem apreciar com amb zip codes entre 95000 i 96000 els clients tenen més tendència a quedar-se i amb zip codes propers a 92000 els clients tenen més tendència a marxar. Anem a veure el mapa de zip codes anterior a veure si podem esbrinar la causa d'aquests canvis en la distribució.



En aquest mapa podem veure coses molt interessants, els Zip Codes entre 95000 i 96000 es troben tots al nord de Califòrnia, per tant **els clients que viuen al nord de Califòrnia per norma general tenen és tendència a quedar-se en l'empresa**. En canvi, els zip codes propers a 92000 són tots propers al sud de Califòrnia, per tant podem intuir que **aquells clients que viuen al sud de Califòrnia tenen més tendència a marxar de l'empresa.** Aquests fenòmens poden estar relacionats amb que el tracte per part de les empreses als clients varia, depenent de la zona de l'estat. Per exemple, la ciutat de San Diego a Califòrnia té Zip Codes molt propers a 92000, San Diego és colindant amb Tijuana (Ciutat de Mèxic), una ciutat on la qualitat de mida no és gaire bona, per tant, pot ser que aquesta marxa de clients vingui donada per pitjor qualitat de vida. En canvi zip codes entre 95000 i 96000 inclouen la ciutat de San Francisco, on la qualitat de vida és molt alta.

Number of Referrals





Stayed Number of Referrals :

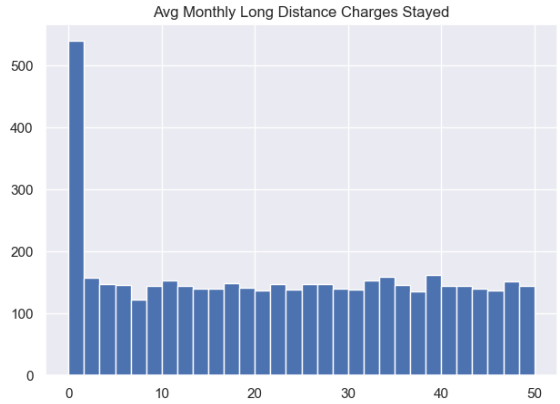
count	4720.000
mean	2.615
std	3.324
min	0.000
25%	0.000
50%	1.000
75%	5.000
max	11.000

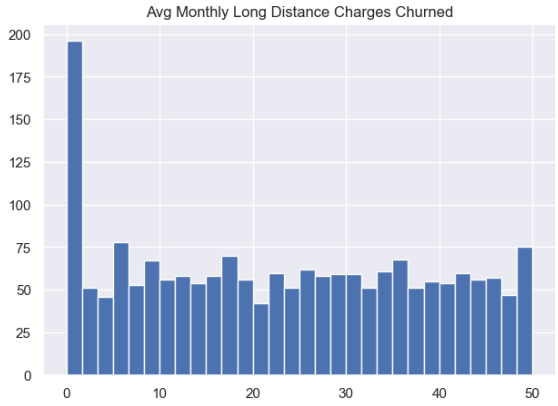
Churned Number of Referrals :

count	1869.000
mean	0.521
std	1.095
min	0.000
25%	0.000
50%	0.000
75%	1.000
max	9.000

En els gràfics i estadístiques es mostren les diferencies entre els clients que es queden i els que marxen de l'empresa pel que fa el nombre de vegades que fan referencia a l'empresa en el seu dia a dia. Clarament en les distribucions podem veure com aquells clients que fan més referencies a l'empresa tenen més tendència a quedar-se. Aquest fenomen també el podem veure reflexat a les estadístiques, ja que la mitjana de persones a les que fan referencia els clients que es queden és 5 cops major que la mitjana dels que marxen.

Avg Monthly Long Distance Charges





Stayed Avg Monthly Long Distance Charges :

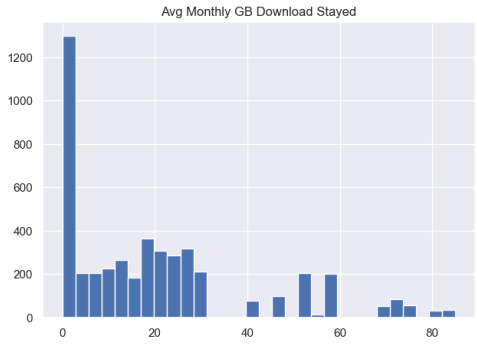
count	4720.000
mean	22.940
std	15.491
min	0.000
25%	9.190
50%	22.950
75%	36.472
max	49.990

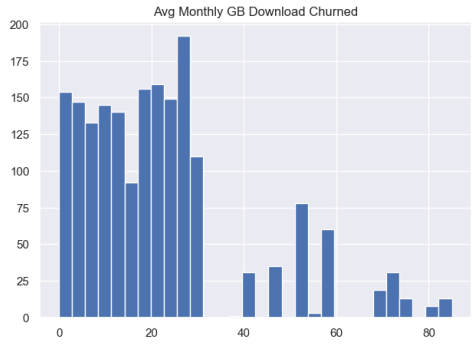
Churned Avg Monthly Long Distance Charges :

count	1869.000
mean	23.168
std	15.409
min	0.000
25%	9.500
50%	22.880
75%	36.360
max	49.980

En els gràfic i estadístiques que es mostren, es pot veure com la variable Avg Monthly Long Distance Charges no influeix en veure si un client es queda o marxa.

Avg Monthly GB Download





Stayed Avg Monthly GB Download :

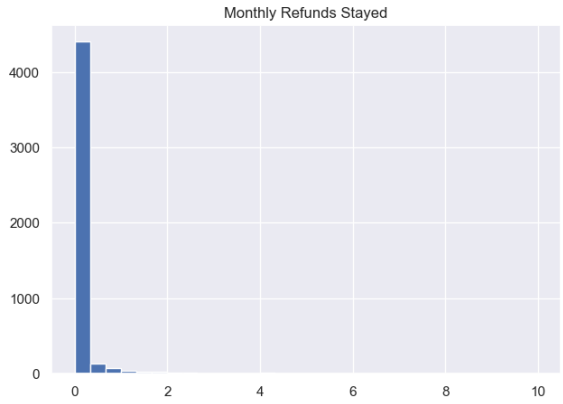
count	4720.000
mean	20.362
std	21.159
min	0.000
25%	0.000
50%	16.000
75%	27.000
max	85.000

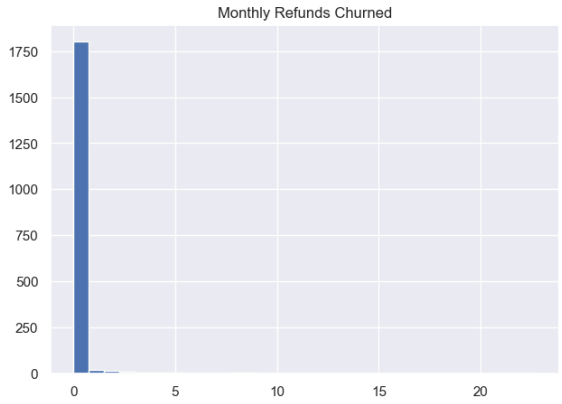
Churned Avg Monthly GB Download :

count	1869.000
mean	22.175
std	18.309
min	0.000
25%	9.000
50%	19.000
75%	27.000
max	85.000

En els gràfics i les estadístiques es mostren els GB descarregats cada més per part dels clients que es queden i els que marxen. Podem apreciar com la proporció de GB descarregats entre el 4 i els 30 augmenta molt en els cients que han marxat respecte els que es queden. Pel que fa valors superiors a 30 la proporció sembla augmentar molt poc en els clients que marxen. Per tant, podem intuir que **aquells clients que descarreguin entre 4 i 30 GB tindran més tendència a marxar**.

Monthly Refunds



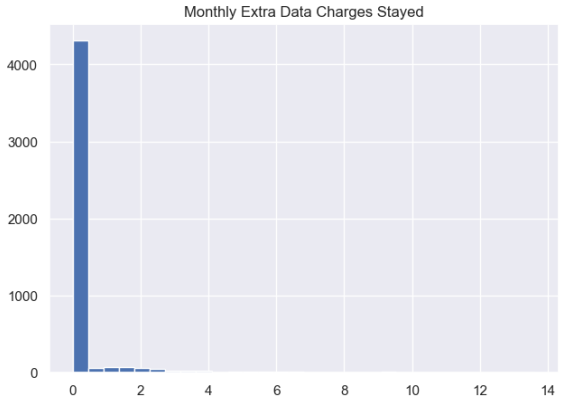


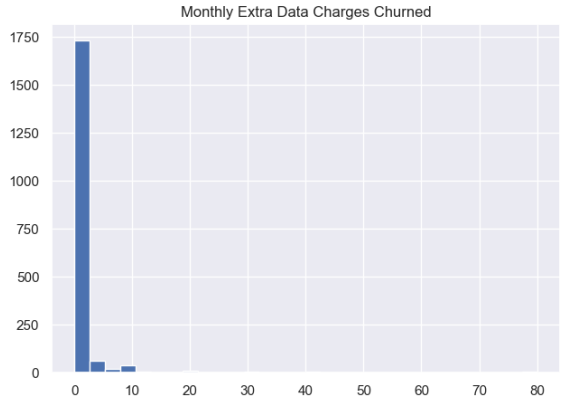

```
Stayed Monthly Refunds :

count    4720.000
mean        0.087
std         0.477
min         0.000
25%         0.000
50%         0.000
75%         0.000
max         9.982
```

```
Churned Monthly Refunds :

count    1869.000
mean        0.131
std         0.884
min         0.000
25%         0.000
50%         0.000
75%         0.000
max        22.720
```

En els gràfics i les estadístiques podem veure la diferencia que hi ha en els 'Refunds' entre les clients que es queden i aquells que marxen. Al ser els valors d'aquesta variable gairebé tot zeros no podem veure grans diferencies entre les distribucions. SI passem a mirar les estadístiques podem veure que el nombre de 'Refunds' en clients que han marxat és 1.5 vegades més gran que en els que es queden. Aquesta diferència al ser tant petita extraurem la conclusió de que aquesta variable gairebé no influeix en que un client es quedi o no.

Monthly Extra Data Charges





Stayed Monthly Extra Data Charges :

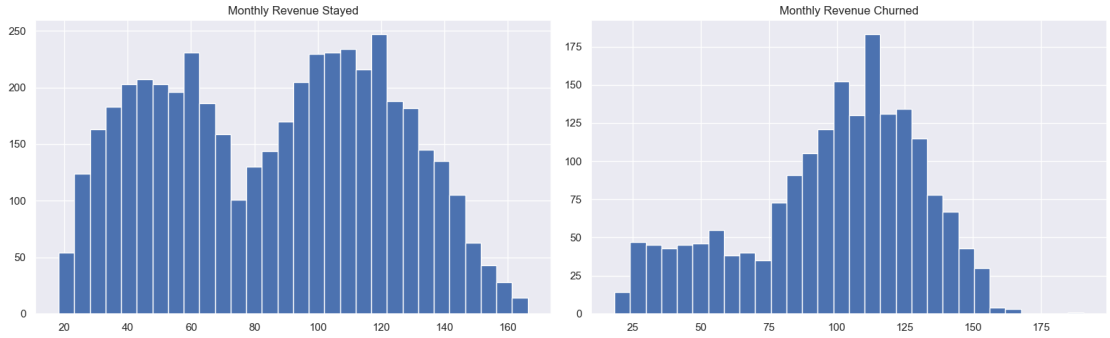
count	4720.000
mean	0.233
std	1.020
min	0.000
25%	0.000
50%	0.000
75%	0.000
max	13.636

Churned Monthly Extra Data Charges :

count	1869.000
mean	0.861
std	4.553
min	0.000
25%	0.000
50%	0.000
75%	0.000
max	80.000

En els gràfics i les estadístiques es mostren els 'Extra Data Charges' dels clients que han marxat de l'empresa i dels que s'han quedat, a simple vista no podem veure gaire canvi perquè la presencia de tants zeros fa que no és pugui interpretar bé. Si ens fixem en les estadístiques podem veure que la mitjana de extra de dades cobrades en els clients que marxen és gairebé 4 cops major que en els clients que es queden. Podem intuir que aquells clients que marxen solen tenir més carrecs per extra de dades utilitzades.

Monthly Revenue



Stayed Monthly Revenue :

count	4720.000
mean	86.069
std	36.413
min	18.166
25%	53.728
50%	89.609
75%	116.260
max	166.380

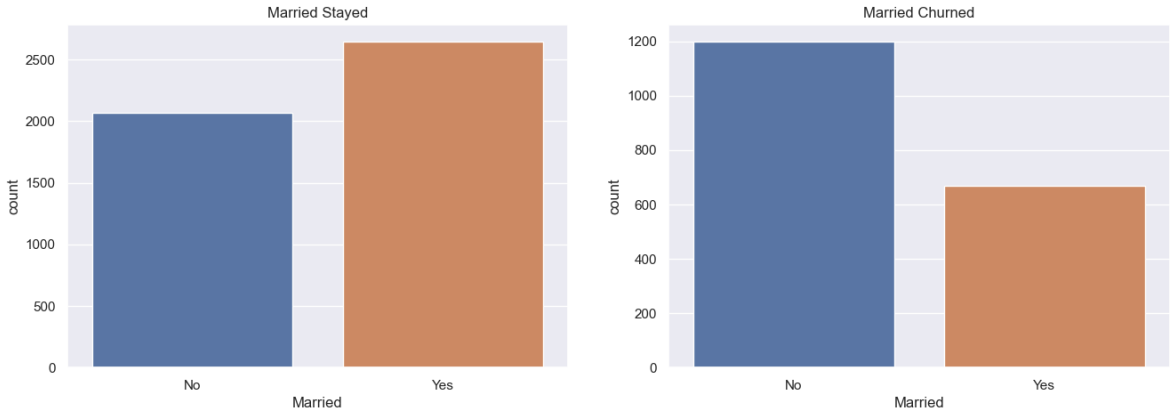
Churned Monthly Revenue :

count	1869.000
mean	98.331
std	32.499
min	18.275
25%	80.475
50%	103.749
75%	121.902
max	190.700

En aquests gràfics i estadístiques és mostra la diferència en el profit que treu l'empresa dels clients que s'han quedat i els clients que han marxat. Si ens fixem en les distribucions podem veure com aquestes són diferents. La diferencia està en els clients que tenen un 'Monthly Revenue' d'entre 20 i 75, en el cas dels clients que es queden, entre aquests dos valors la proporció respecte el total és gran, en canvi en el cas dels clients que han marxat, la proporció entre aquests dos valors és menor. El que ens està indicant això és que la majoria de clients que generen un revenue mensual d'entre 20 i 75 a l'empresa tenen més tendència a quedar-se. Pel que fa la resta de la distribució podem veure que entre els valors 75 i 170 la distribució en ambdós tipus de clients segueix una normal, en el cas dels clients que han marxat és pot veure com els clients que generen un 'Revenue' entre 75 i 140 tenen més tendència a marxar que els que generen menys.

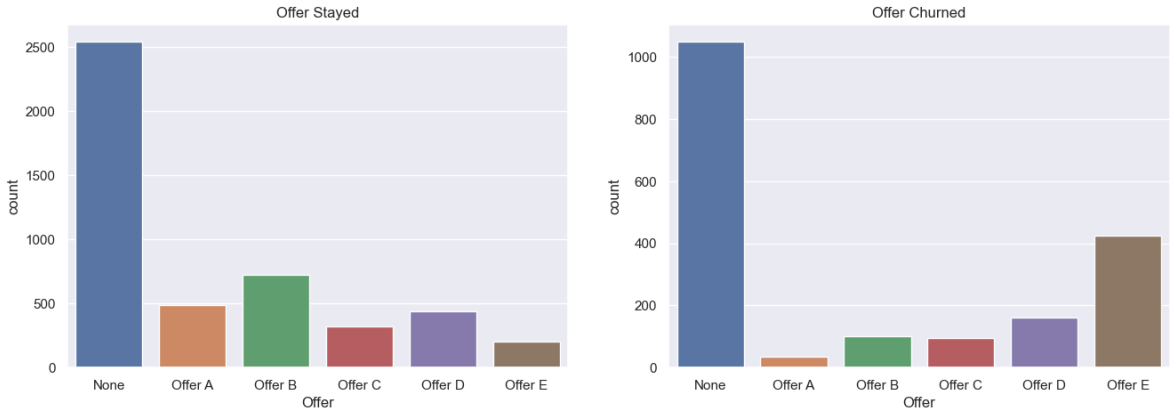
Variables categòriques:

Married:



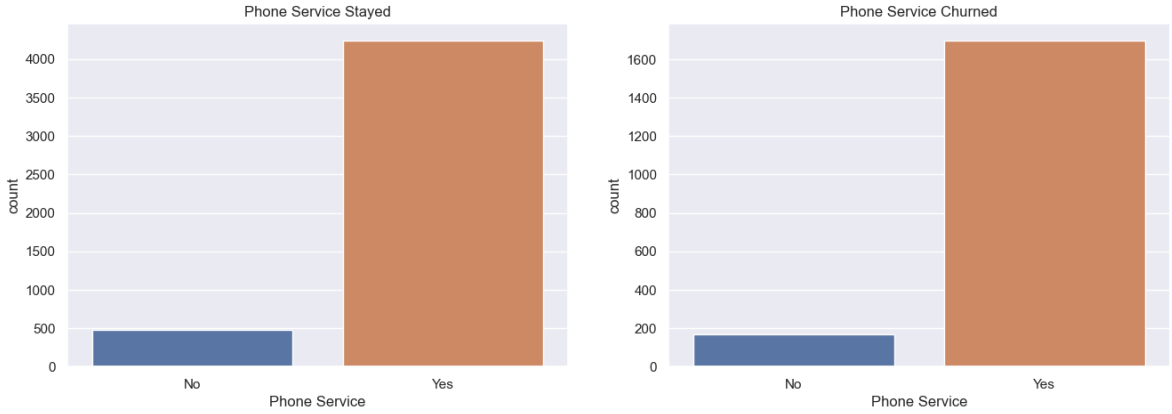
En aquests 2 gràfics es mostra la distribució de la variable 'Married' en clients que han marxat i en clients que s'han quedat a l'empresa. Podem veure com les proporcions canvien d'un gràfic a l'altre, en concret es pot apreciar que la proporció de No augmenta i la de Si disminueix en clients que marxen, per tant es pot extreure la conclusió de que **els clients no casats són més propensos a marxar de l'empresa que els que ho estan.**

Offer:



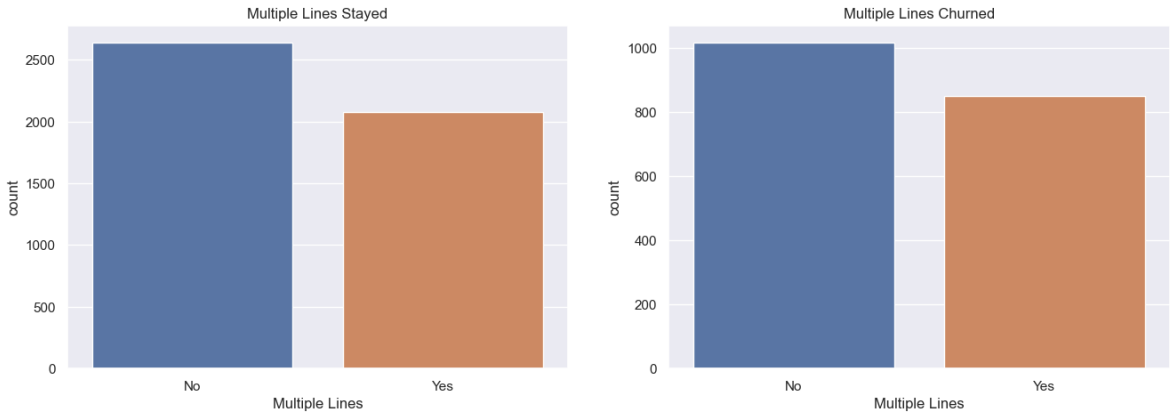
En aquests 2 gràfics es mostra la distribució de la variable 'Offer' en clients que han marxat i en clients que s'han quedat a l'empresa. A simple vista és pot veure com les proporcions de les ofertes varien. Podem veure com la proporció d'oferta A i B disminueix molt en els clients que han marxat, en canvi l'oferta E augmenta molt. Amb aquests canvis podem concloure que **l'oferta A i B tenen tendència a mantenir els clients**, en canvi **la E és tot lo contrari, els clients que la reben tenen tendència a marxar**.

Phone Service:



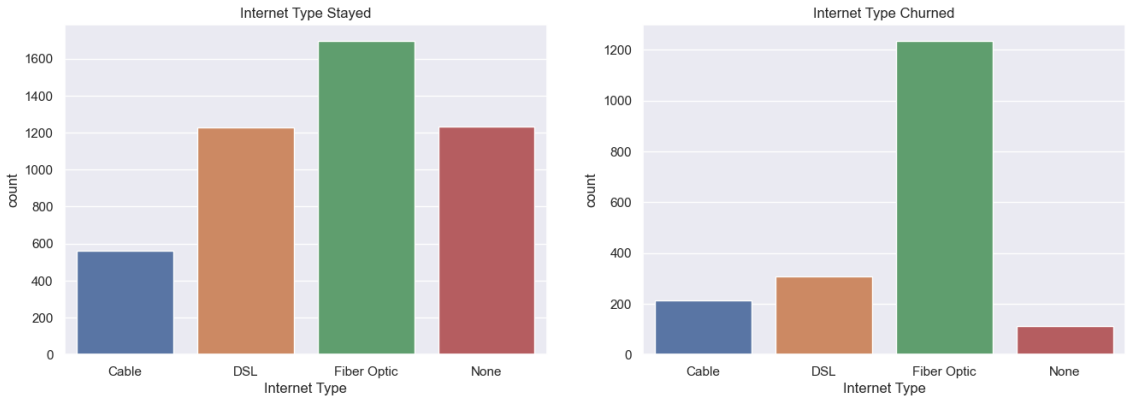
En aquests 2 gràfics es mostra la distribució de la variable 'Phone Service' en clients que han marxat i en clients que s'han quedat a l'empresa. Podem veure que la distribució no varia, per tant el fet de que un client tinguin o no el **servei de telèfon contractat no influeix** en si aquest marxarà o no.

Multiple Lines:



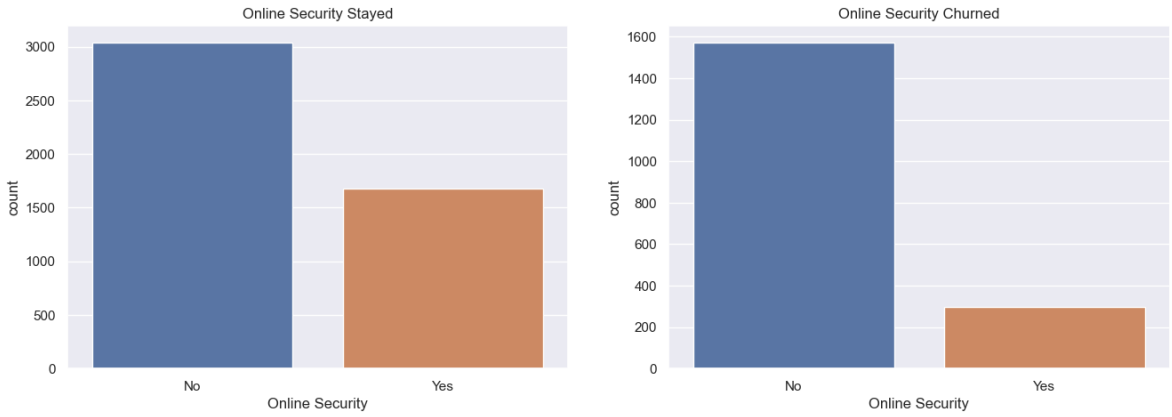
En aquests 2 gràfics es mostra la distribució de la variable 'Multiple Lines' en clients que han marxat i en clients que s'han quedat a l'empresa. Com es pot veure la distribució varia molt poc i per tant no podem afirmar que el fet de tenir contractades múltiples línies de telèfon influeixi en el comportament dels clients.

Internet Type:



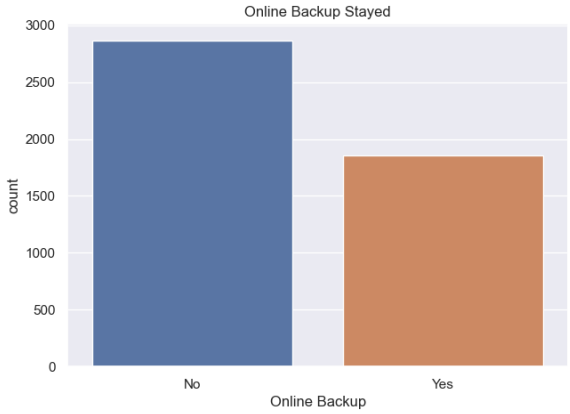
En aquests 2 gràfics es mostra la distribució de la variable 'Internet Type ' en clients que han marxat i en clients que s'han quedat a l'empresa. Es pot apreciar com les distribucions dels tipus de internet varien. El canvi de proporció es veu més clarament és quan un client té o no internet. La proporció de la categoria None disminueix molt en els clients que han marxat respecte els que s'han quedat. Això ens està indicant que **els clients que no tenen cap tipus d'internet contractat tenen més tendència a quedar-se en l'empresa**. Les altres proporcions que també varien són Cable i DSL, ambdues disminueixen en els clients que han marxat, això ens fa pensar que **els clients que tenen Cable i DSL contractats tenen més tendència de quedar-se a l'empresa.**

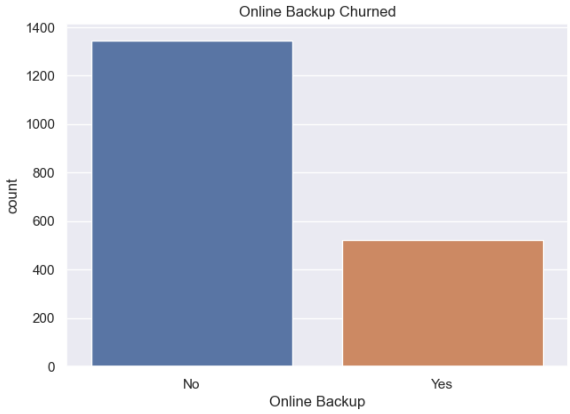
Online Security:



En aquests 2 gràfics es mostra la distribució de la variable 'Online Security 'en clients que han marxat i en clients que s'han quedat a l'empresa. Com podem veure la proporció de Si disminueix clarament en els clients que han marxat respecte els clients que s'han quedat. Per tant, podem concloure que **els clients que tenen El servei d'Online Security contractat tenen més tendència a quedar-se a l'empresa.**

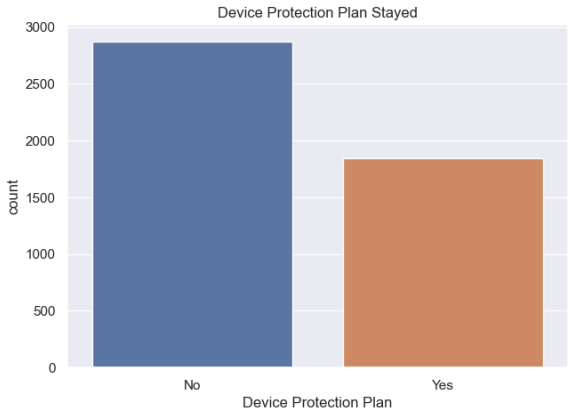
Online Backup:

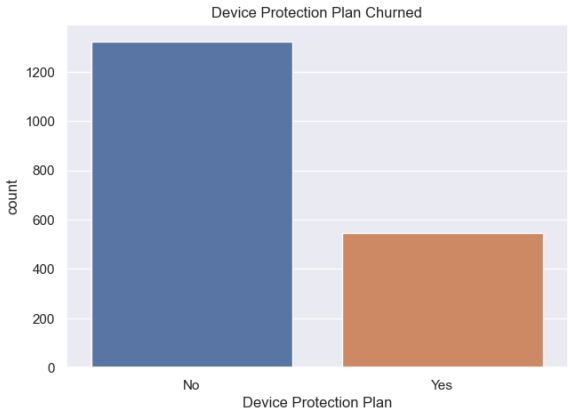




En aquests 2 gràfics es mostra la distribució de la variable 'Online Backup 'en clients que han marxat i en clients que s'han quedat a l'empresa. Podem apreciar com la proporció de Yes en els clients que han marxat disminueix una mica respecte la proporció de Yes en els clients que s'han quedat, per tant, aquells que tenen Online Backup contractat tenen una mica més de tendència a quedar-se a l'empresa.

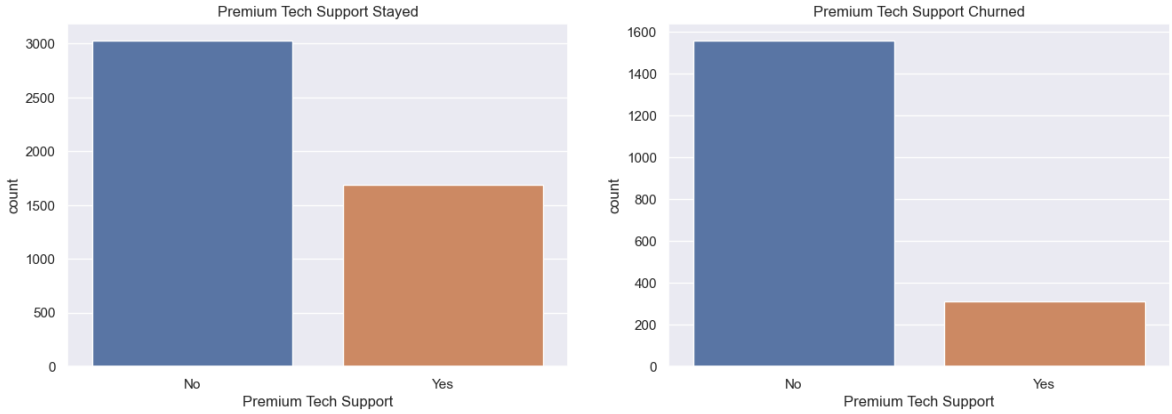
Device Protection Plan:





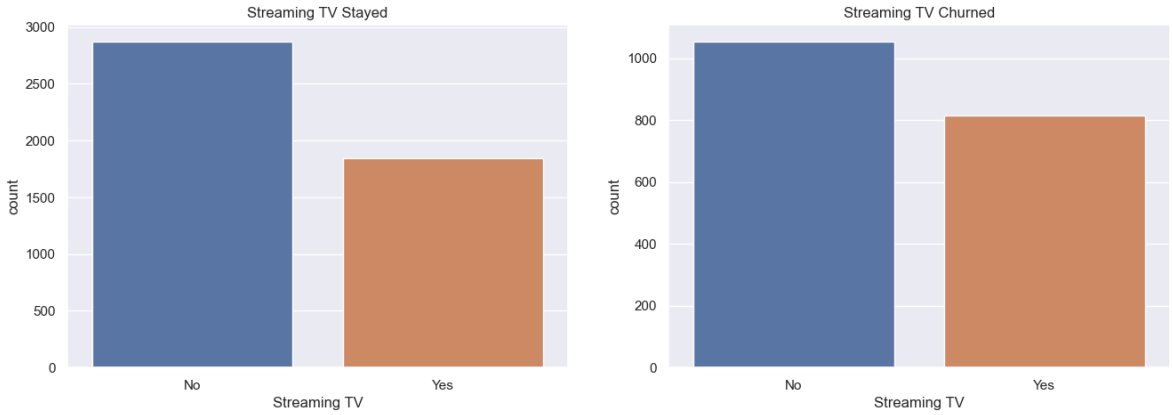
En aquests 2 gràfics es mostra la distribució de la variable 'Device Protection Plan' en clients que han marxat i en clients que s'han quedat a l'empresa. Igual que en la variable anterior 'Online Backup' podem apreciar com la proporció de Yes en els clients que han marxat disminueix una mica respecte la proporció de Yes en els clients que s'han quedat, per tant, aquells que tenen Device Protection Plan contractat tenen una mica més de tendència a quedar-se a l'empresa.

Premium Tech Support:



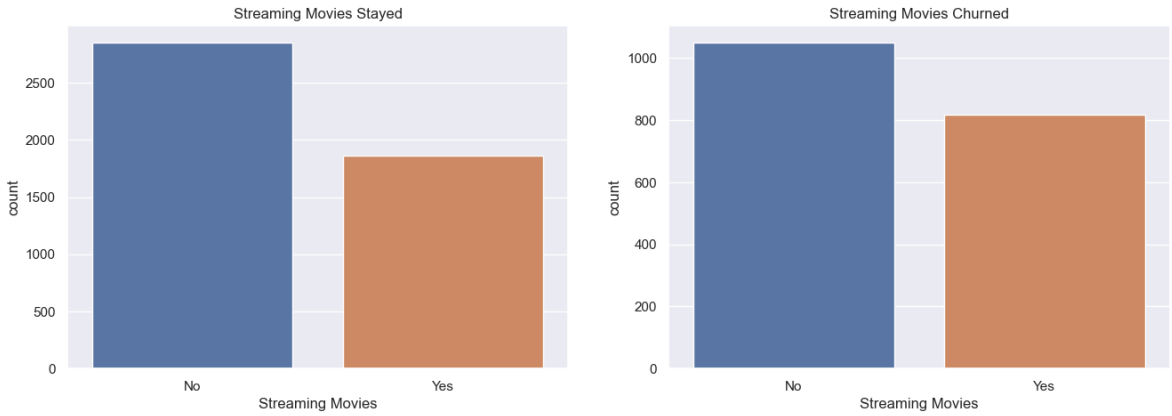
En aquests 2 gràfics es mostra la distribució de la variable 'Premium Tech Support' en clients que han marxat i en clients que s'han quedat a l'empresa. Tornem a veure com la categoria Yes disminueix la seva proporció en els clients que han marxat respecte els que s'han quedat, concloem per tant que aquells clients que tinguin contractat el servei de 'Premium Tech Support' tenen més tendència a quedar-se en l'empresa.

Streaming TV:



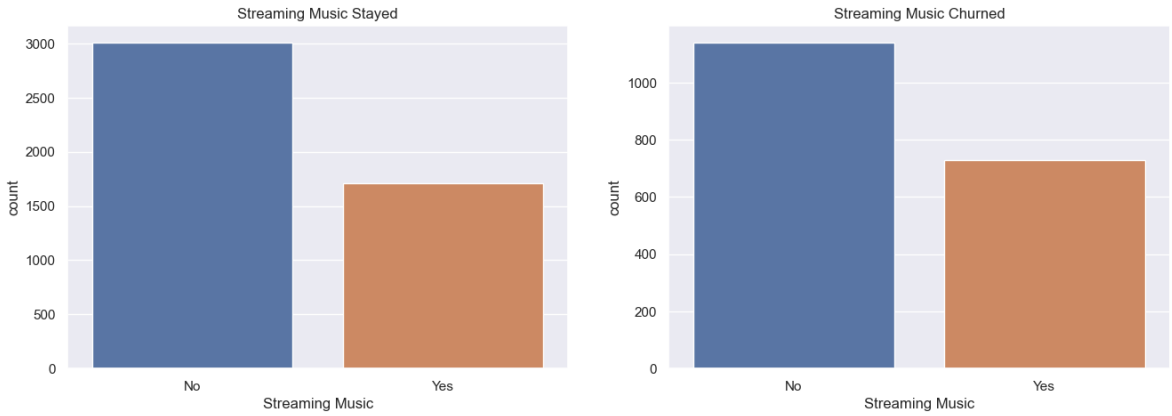
En aquests 2 gràfics es mostra la distribució de la variable 'Streaming TV' en clients que han marxat i en clients que s'han quedat a l'empresa. Podem veure com la proporció de Yes en els clients que han marxat augmenta una mica respecte els que s'han quedat. Malgrat això, el canvi en la proporció es tant petit que podria venir donat per una casualitat, per tant no extraurem cap conclusió sobre l'estat del client basant-nos en aquesta variable.

Streaming Movies:



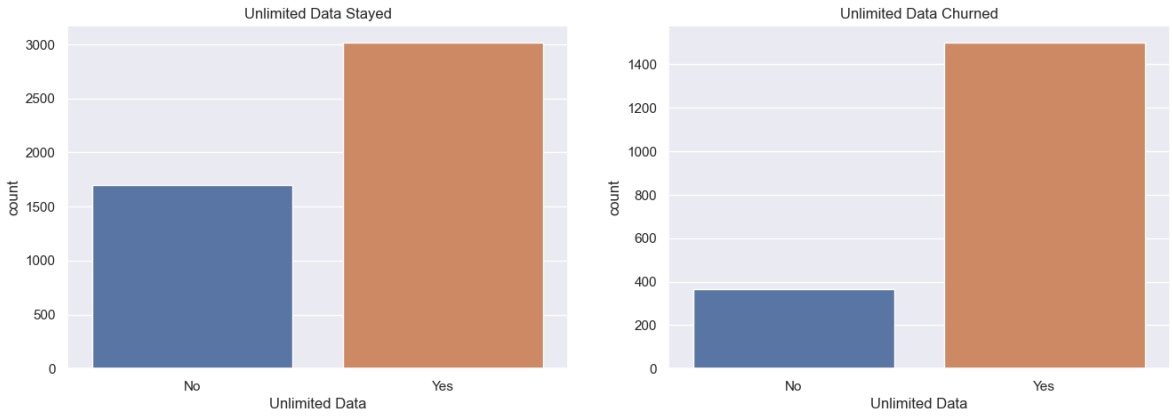
En aquests 2 gràfics es mostra la distribució de la variable 'Streaming Movies' en clients que han marxat i en clients que s'han quedat a l'empresa. Tornem a veure com hi ha un petit augment en la proporció de Yes en els clients que han marxat respecte els que no ho han fet. Encara haver-hi diferencies no extraurem conclusions perquè els canvis són molt petits.

Streaming Music:



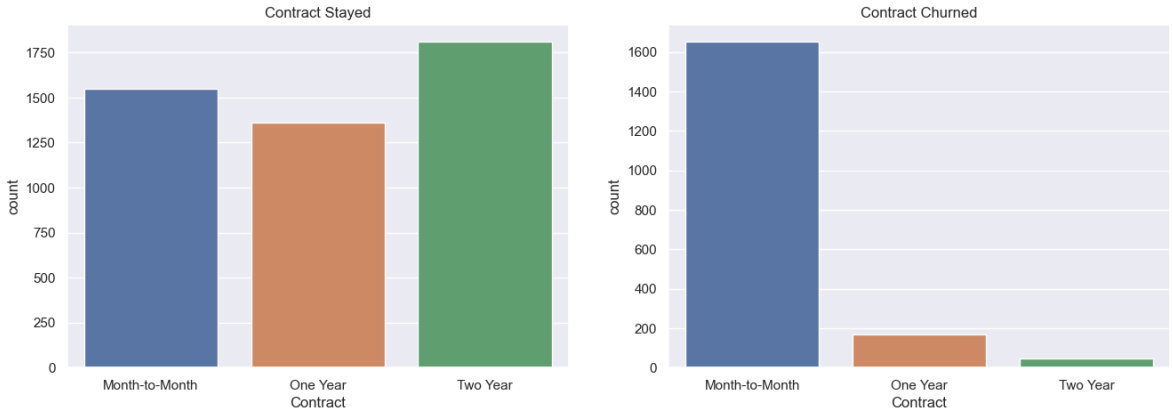
En aquests 2 gràfics es mostra la distribució de la variable 'Streaming Music' en clients que han marxat i en clients que s'han quedat a l'empresa. Podem veure com la proporció de Yes en els clients que han marxat augmenta una mica respecte els que s'han quedat. Encara així, el canvi en la proporció es tant petit que podria venir donat per una casualitat, per tant no extraurem cap conclusió sobre aquesta variable.

Unlimited Data:



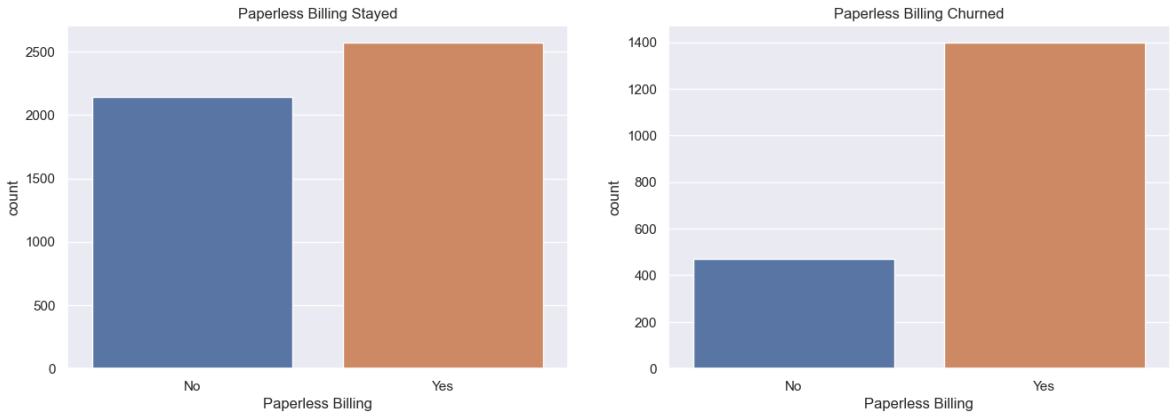
En aquests 2 gràfics es mostra la distribució de la variable 'Unlimited Data' en clients que han marxat i en clients que s'han quedat a l'empresa. Podem veure un augment en la proporció de No en clients que s'han quedat a l'empresa respecte la proporció de No dels clients que han marxat. Això ens indica que **els clients que no tenen el servei de 'Unlimited Data' tenen més tendència a quedar-se a l'empresa.**

Contract:



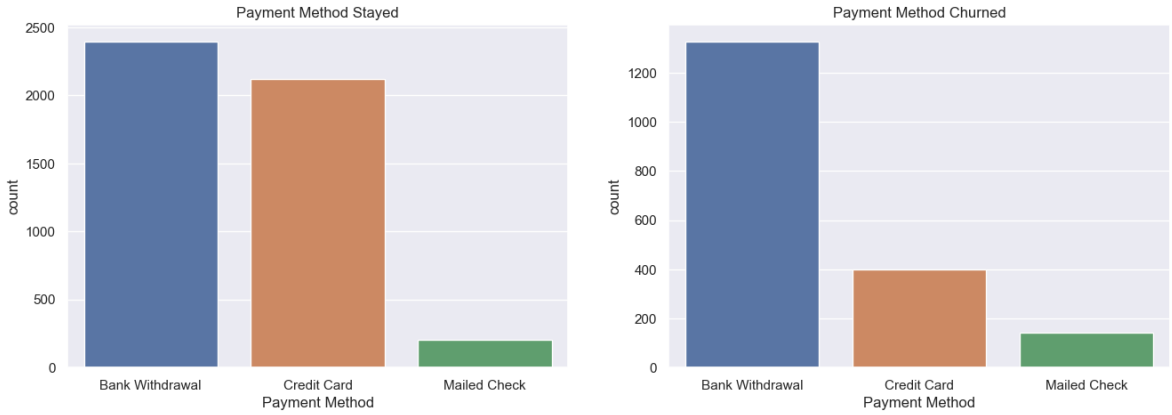
En aquests 2 gràfics es mostra la distribució de la variable 'Contract' en clients que han marxat i en clients que s'han quedat a l'empresa. En aquestes distribucions podem veure un gran canvi en les proporcions de les categories de les variables. Podem apreciar com la proporció de clients que han marxat que tenien el contracte d'1 o 2 anys es redueixen dràsticament en comparació amb aquestes mateixes proporcions en clients que s'han quedat. D'això podem concloure que **els clients que tenen un contracte d'1 o 2 anys tenen molta més tendència a quedar-se a l'empresa que a marxar.** Pel que fa la categoria Month-to-Month, podem veure com gairebé tots els clients que han marxat tenen aquest tipus de contracte, en canvi la proporció de clients que es queden i tenen contracte Month-to-Month és més baixa, per tant si nosaltres afegíssim un altre client amb contracte Month-to-Month aquest tindria més tendència a marxar que a quedar-se en l'empresa.

Paperless Billing:



En aquests 2 gràfics es mostra la distribució de la variable 'Paperless Billing' en clients que han marxat i en clients que s'han quedat a l'empresa. Podem veure com la proporció de No en clients que han marxat disminueix notòriament respecte aquesta mateixa proporció en clients que s'han quedat a l'empresa. D'aquest fet podem concloure que **aquells clients que no hagin escollit Paperless Billing tenen més tendència a quedar-se a l'empresa.**

Payment Method:



En aquests 2 gràfics es mostra la distribució de la variable 'Payment Method' en clients que han marxat i en clients que s'han quedat a l'empresa. Podem veure clarament com els clients amb un mètode de pagament de targeta de crèdit tenen una proporció més petita en els clients que han marxat que en els que s'han quedat, per tant podem concloure que **si un client paga amb targeta de crèdit és més propens a quedar-se en l'empresa.** A més pel que fa la proporció de **pagament amb transferències bancaries augmenta en els clients que han marxat per tant intuïm que aquests tindran més tendència a marxar**.

Conclusions:

Com hem pogut veure hi ha diferents variables de les que hem analitzat que no influeixen directament en que un client marxi de l'empresa. Encara així hem decidit mantenir aquestes variables perquè pot ser que la interacció d'aquesta variable amb una altre si influeixi en determinar si un client marxa o no.

- El perfil d'un client que és propens a marxar té les següents característiques:
    - Majors de 65 anys.
    - Viuen amb cap o pocs familiars.
    - Viuen al sud de Califòrnia (molt influent).
    - No fan gairebé cap referencia a l'empresa.
    - Solen descarreguen entre 4 i 30 GB.
    - Solen tenir més carrecs extra en dades.
    - Tenen un revenue entre 75 i 140.
    - No solen estar casats.
    - Solen rebre l'offerta E (molt influent).
    - Tenen servei d'internet contractat (poc influent).
    - No solen tenir el servei d'Online Security contractat (poc influent).
    - No solen tenir el servei de Premium tech Support (poc influent).
    - Solen tenir el servei d'unlimited data (poc influent).
    - Gairebé mai tenen contracte de One Year o Two Year, sempre tenen el contracte Month-to-Month (molt influent).
    - Solen escollir 'Paperless Billing'.
    - El principal mètode de pagament és extracció bancaria.

- El perfil d'un client que és propens a no marxar té les següents característiques:
    - Solen tenir entre 20 i 65 anys.
    - Solen viure amb més familiars.
    - Una gran part viu al nord de Califòrnia.
    - Fan més d'una referencia a l'empresa.
    - Solen tenir menys extra data charges.
    - La majoria de clients tenen un revenue d'entre 20 i 80.
    - Solen estar casats.
    - Reben oferta A i B (molt influent).
    - La gran majoria de clients que no tenen servei d'internet contractat no marxen (molt influent).
    - Solen tenir el servei d'Online Security contractat (poc influent).
    - Solen tenir el servei de Premium tech Support (poc influent).
    - No solen tenir el servei d'unlimited data (poc influent).
    - Gairebé tots tenen contracte de one o Two Years (molt influent).
    - No solen escollir 'Paperless Billing'.
    - La majoria dels clients que paguen amb tarjeta de crèdit es solen quedar.

Recomanacions a l'empresa:

A continuació proposarem un seguit d'accions que hauria de realitzar l'empresa que estem analitzant si volen mantenir un tipus de client que es propens a marxar d'aquesta.

1. Durant l'anàlisi de la variable Zip Code, hem vist que hi ha un pic de clients que marxen molt gran en el sud de Califòrnia. Per ajudar a l'empresa a mantenir aquests clients procuraria analitzar quins locals hi ha per aquestes localitats i intentar esbrinar la causa de que tots aquests clients estiguin marxant. Pot ser en algunes botigues d'aquesta zona el tracte als clients no es gaire bo i per aquesta raó més clients marxen.

2. Quan hem analitzat les ultimes ofertes que s'han fet als clients hem trobat una clara diferencia entre 2 tipus d'ofertes. Els clients que reben l'oferta E tenen molta tendència a marxar de l'empresa, en canvi aquells que reben la A gairebé mai marxen. Recomanaríem a l'empresa que intentessin canviar les condicions de l'oferta E o inclús eliminar-la i intentar fomentar l'oferta A perquè el canvi entre aquestes dues és molt significatiu. Sobretot, seria bona idea fomentar l'oferta A entre aquells clients que tenen les característiques que hem esmentat anteriorment.

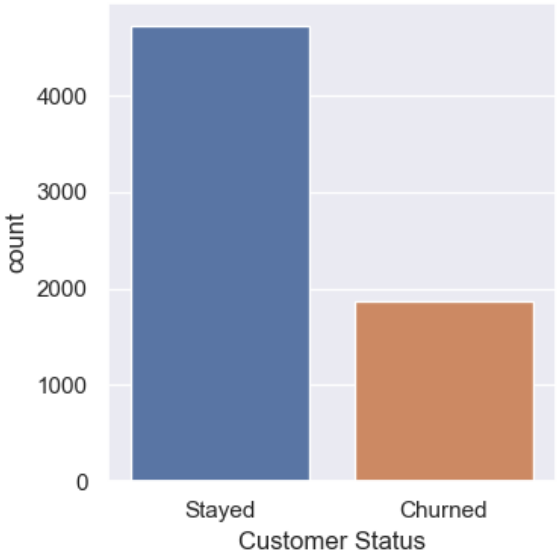
3. A l'hora d'analitzar el tipus de contracte que tenen els clients hem trobat que gairebé tots els clients que marxen tenen contracte Month-to-Month, en canvi els que tenen contracte One Year marxen molt poc i els que tenen Two Years gairebé mai marxen. Si l'empresa vol mantenir més clients els hi recomanaria fomentar que els clients tinguin contractes de més durada, ja sigui fent preus especials o utilitzant altres formes de Màrqueting.

4. Quan analitzem els tipus d'internet contractats veiem com la gran majoria de clients que no tenen contractat el servei d'internet es mantenen a l'empresa. Recomanaria fer un estudi sobre les diferencies en els plans que tenen contractats els clients que tenen internet i els que no, d'aquesta manera es podran comparar i trobar les virtuts dels plans que no proporcionen internet i les mancances d'aquells plans que si que en proporcionen.

3. Preprocessat de dades: Classifier

3.1. Estudi de balanceig de respecte a la variable objectiu.

En aquest apartat estudiarem la distribució de la variable objectiu 'Customer Status' abans de fer el particionat de dades en train-val/test.




```
Stayed      4720
Churned     1869
```

Com podem veure en el gràfic tenim un 71% dels clients que s'han quedat a l'empresa i un 29% que han marxat. Aquest fet significa que tenim un desbalanceig en la nostra base de dades pel que fa la variable objectiu 'Customer Status'. Tenir una base de dades desbalancejada pot ser un problema per als nostres models ja que pot donar lloc a que aquests estiguin esbiaxats per la classe majoritaria. Això resulta en una performance molt pobre a l'hora de predir la classe minoritaria, la qual cosa representa un gran problema ja que nosaltres volem predir amb presició quins clients marxaran.

3.2. Definició del particionat (train-val/test).

El particionat de les dades és molt important perquè permet evitar l'overfitting del model, és a dir, permet evitar que el model s'ajusti massa als dades d'entrenament i no sigui capaç de generalitzar bé a dades noves. Si utilitzéssim les mateixes dades per entrenar i validar el model, seria més probable que el model tingués una precisió aparentment elevada en les dades d'entrenament, però no fos capaç de generalitzar bé a dades noves. Això es deu a que el model s'ha ajustat massa a les característiques de les dades d'entrenament i no és capaç de generalitzar a dades noves.

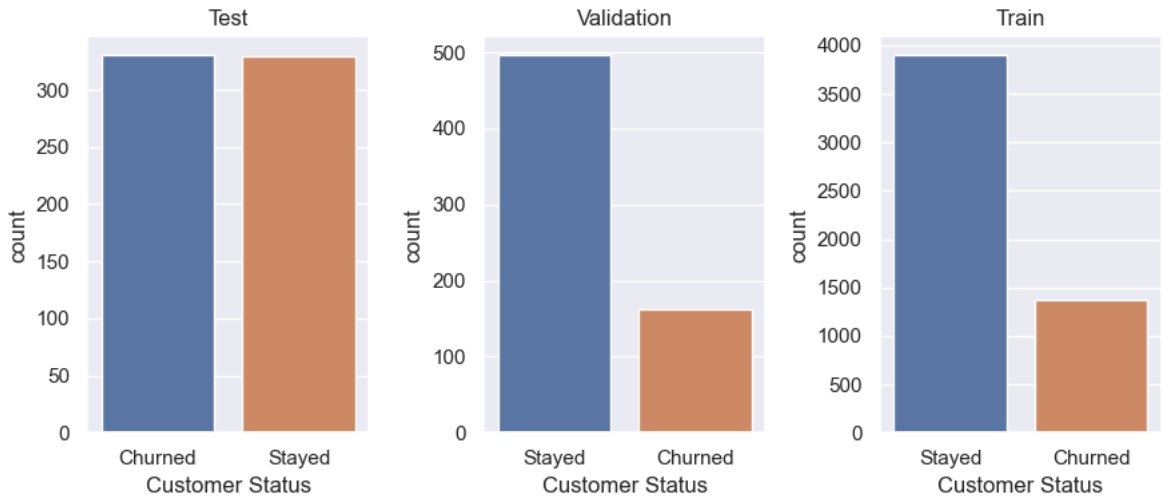
En canvi, si utilitzem un conjunt de validació diferent per avaluar el model, podem obtenir una mesura més precisa de la seva precisió en dades noves i estar més segurs que el model serà capaç de generalitzar bé a dades noves en el futur.

Abans de continuar amb aquest apartat cal recalcar que no utilitzarem cross validation perquè aquesta implica agafar mostres aleatòriament, per tant si la base de dades està desbalancejada es molt probable que els subconjunts de mostres que agafi cross-validation també ho estiguin i que, per conseqüència, l'entrenament dels models provoqui que aquests tinguin una capacitat de generalització molt pobre per a la classe minoritària.

A continuació, farem un particionat de les dades de manera que el test estigui totalment balancejat. Cal destacar que prenem mostres aleatòries quan fem aquesta partició perquè cal assegurar que les dades del conjunt d'entrenament, validació i test han de ser representatives del conjunt total de

dades. Si no es prenen mostres aleatòries, és possible que hi hagi una distorsió en la distribució de les dades entre els conjunts d'entrenament i validació, el que pot afectar la precisió del model.

Tenir el test totalment balancejat farà que les particions de train i val estiguin encara més desbalancejades. Per comprovar que el que estem esmentant es compleix anem a fer la partició i anem a mostrar la distribució de la variable objectiu 'Customer Status' en les 3 particions.



Podem veure la distribució de la variable objectiu a les tres particions fetes i veiem com efectivament al balancejar perfectament el test hem generat encara més desbalanceig en les altres particions, en concret en la partició validation ara tenim un 75% de clients stayed i un 25% churned i en la partició train tenim un 74% de clients stayed i un 26% churned. A continuació explicarem quina estratègia seguir per balancejar aquestes dues particions

3.3. Definició de l'estratègia per mitigar el desbalanceig en train i val.

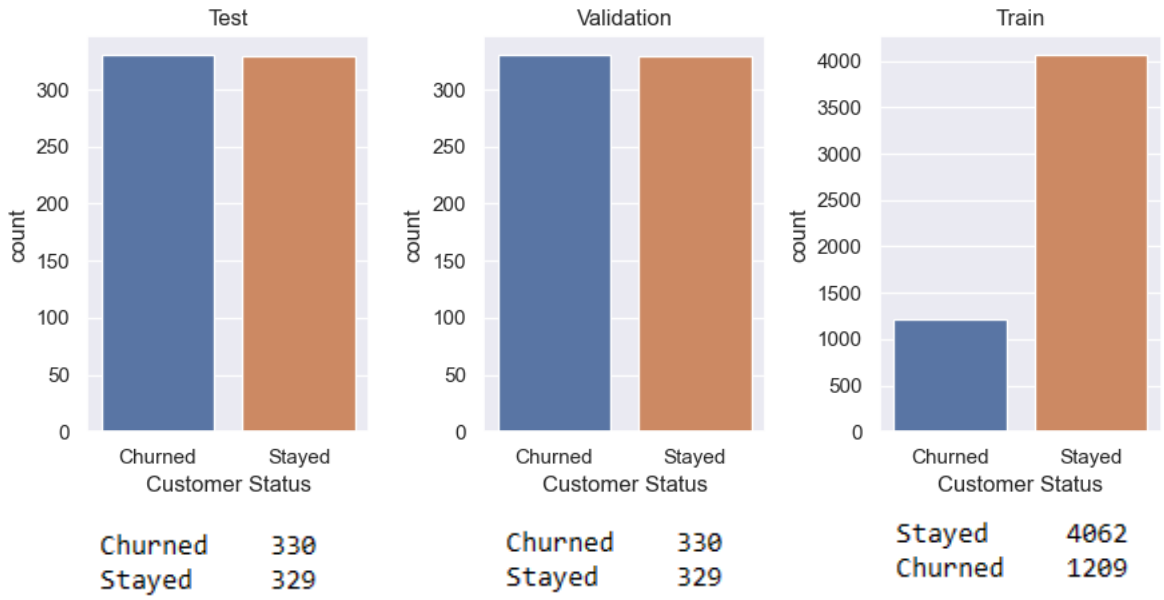
Tal i com hem vist en l'apartat anterior la partició de validation i la partició de train han quedat desbalancejades. Aquest fet s'ha de solucionar per les dues particions perquè la part de validació ha de representar al test, ja que si no és així quan entrenem el model i per exemple ajustem hiperparàmetres els resultats que ens doni la validació seran molt diferents als que ens donaria el test i per tant estaríem entrenant un model que obtindria mals resultats en el test. Pel que fa la partició train podem balancejar-la o no, no balancejar-la implicaria haver d'utilitzar pesos a l'hora d'entrenar els models. A continuació explicarem les diferents estratègies que podem utilitzar per balancejar les nostres particions:

- Undersampling: consisteix en eliminar mostres aleatòriament de la classe stayed de la base de dades.
- Oversampling: consisteix en duplicar mostres aleatòriament de la classe churned de la base de dades.

- SMOTE (Synthetic Minority Oversampling Technique): aquesta és una tècnica més sofisticada que crea mostres sintètiques de la classe churned enlloc de simplement duplicar les ja existents.
- Weighted sampling: consisteix en assignar pesos a les observacions en la base de dades de manera que el model doni més o menys importància a certes mostres durant l'entrenament.

Un cop hem analitzat la situació en la que ens trobem farem el següent. En primer lloc tornarem a fer la partició de la nostra base de dades de manera que Val també estigui balancejada, d'aquesta manera ens estalviarem utilitzar cap mètode de balanceig en aquesta partició. La part negativa de fer això és que la partició train perdrà 166 mostres on els clients han marxat. Això representa un 12% del total de mostres que tenia la partició train dels clients que havien marxat, així que tampoc suposa una gran pèrdua de dades i ens assegurem que la part de validation sigui lo més semblant possible al test.

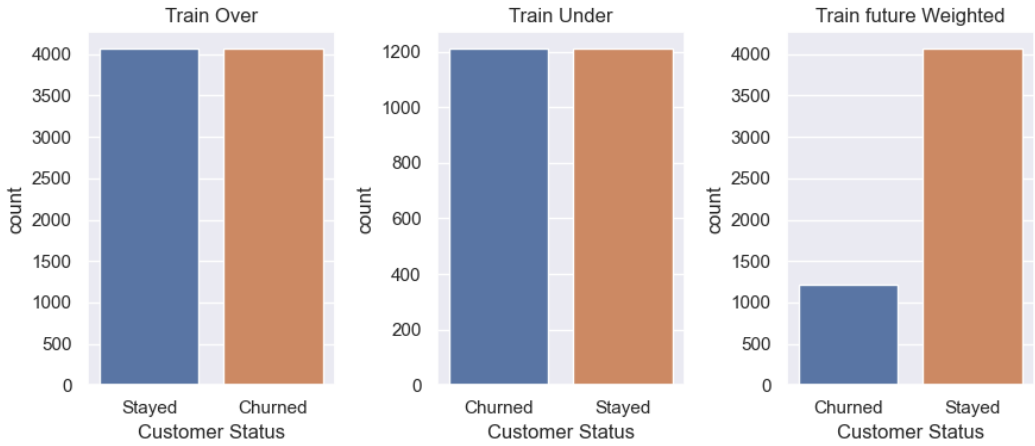
En segon lloc decidirem quins mètodes de balanceig utilitzem per a la partició train, però primer anem a tornar a fer el particionat amb els canvis esmentats.



Com podem veure obtenim lo esperat, test i val perfectament balancejats i train amb un desbalanceig de 77% de clients que s'han quedat i un 23% de clients que han marxat.

Per balancejar el train utilitzarem tres formes diferents: Undersampling, Oversampling i Weighted sampling a l'hora d'entrenar els models. D'aquesta manera més endavant podrem comparar els diferents resultats depenent del mètode emprat.

Resultats de balancejar el train amb diferents mètodes:



Podem veure com amb oversampling tenim 4062 mostres balancejades, amb undersampling tenim 1209 mostres balancejades i en el future weighted sampling tenim la partició de train original amb la qual provarem d'utilitzar pesos quan entrenem els models.

Cal tenir en compte que al fer under i oversampling al escollir aleatòriament quines mostres traiem i quines dupliquem és possible que la distribució de les variables variï una mica.

Síntesi:

Test: 659 mostres perfectament balancejades amb les dades originals (x_test i y_test)

Validate: 659 mostres perfectament balancejades amb les dades originals (x_val i y_val)

Train:

- Original per weighted sampling: 4062 Stayed i 1209 Churned (x_train i y_train)
- Oversampling: 8124 mostres perfectament balancejades amb el mètode oversampling (x_train_over i y_train_over)
- Undersampling: 2418 mostres perfectament balancejades amb el mètode undersampling (x_train_under i y_train_under)

3.4. Normalització de dades, basada en train-val.

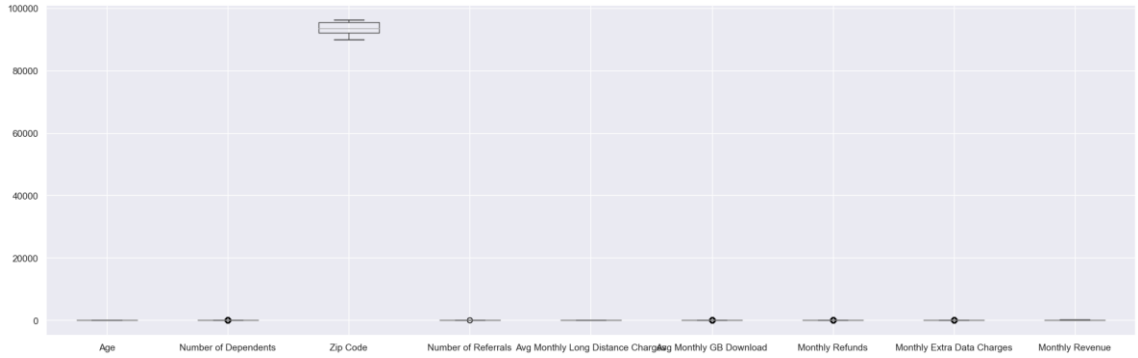
En aquest apartat normalitzarem totes les dades de la partició dels diferents train que tenim i també ho farem de la partició validation i test. Escalarem les nostres variables perquè tots aquells algoritmes que utilitzen distàncies per treballar necessiten que totes les variables numèriques tinguin el mateix rang.

Contemplarem 3 mètodes per normalitzar les nostres dades:

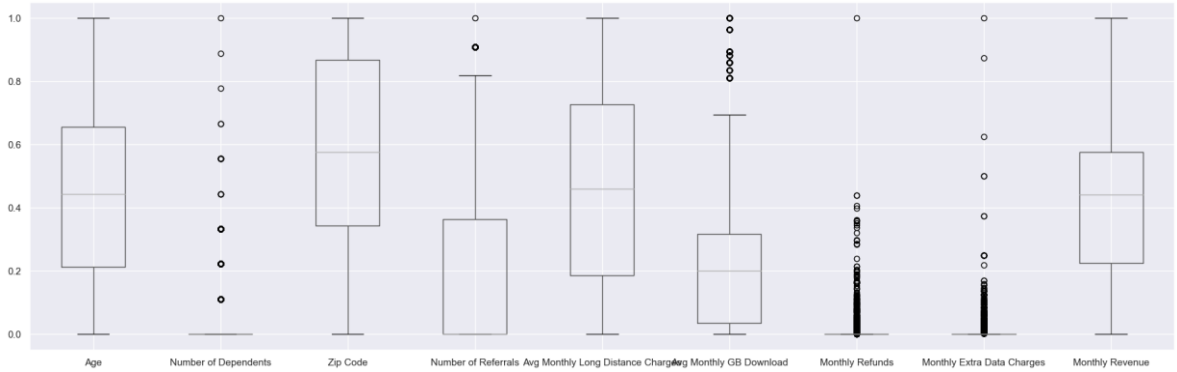
- Standarization: $\frac{X-\mu}{\sigma}$ transformarà les dades perquè tingui mitjana 0 i std 1
- Min-max scaling: $\frac{X-X_{min}}{X_{max}-X_{min}}$ fara que les dades estiguin dins del rang [0,1].
- Boxcox transformation: $\frac{X^{\lambda}-1}{\lambda}$ if $\lambda \neq 0$ or $ln(X)$ if $\lambda = 0$ transforma les dades perquè aquestes segueixin una distribució normal.

D’aquests 3 mètodes utilitzarem Min-max perquè redueix totes les variables al mateix rang [0,1] sense influir en la seva distribució. També podríem utilitzar Boxcox i Min-max per primer normalitzar la distribució de les variables i després escalar el rang de totes entre 0 i 1, però com els algoritmes que utilitzarem no assumeixen cap distribució de les dades, normalitzar la distribució d’aquestes no ens ajudaria.

A continuació anem a veure boxplots de totes les variables de les nostres particions de train:



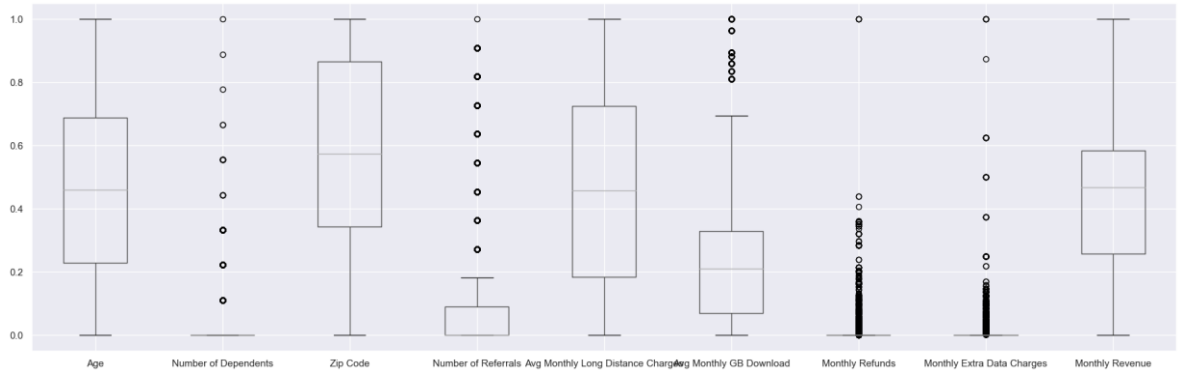
Com podem veure e la imatge es compleix el que havíem esmentat, les variables tenen rangs diferents. Ara utilitzarem el mètode minmax per escalar aquestes variables i visualitzarem els resultats.



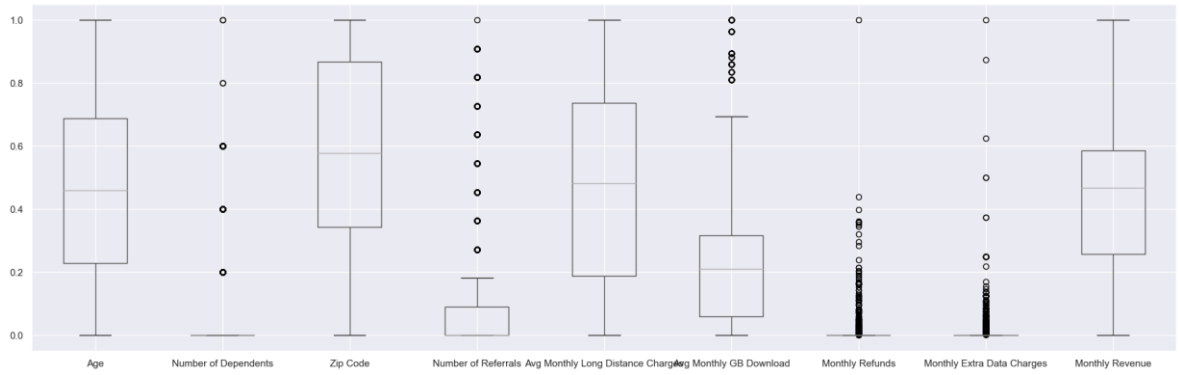
Es pot apreciar com ara totes les variables tenen el mateix rang [0,1], d’aquesta manera l’algoritme no assignarà més importància a una variable pel fet de tenir valors més grans.

Un cop hem vist els resultats amb la partició desbalancejada de train normalitzem també les particions de train amb oversampling i undersampling.

Resultats d'oversampling després de normalització:



Resultats d'oversampling després de normalització:



En aquests 2 resultats de la normalització de les particions del train balancejades podem veure com obtenim resultats molt semblants als de la partició train original. Tal i com hem comentat al final de l'apartat 3.3 al fer over i undersampling és possible que la distribució d'alguna variable variï. Aquí podem veure com la distribució de la variable number of referrals varia respecte la que té en la partició de train original (sense balancejar).

Utilitzarem la partició train normalitzada amb minmax en tots els models perquè encara que aquells com Decision Tree, XGBoost o Random Forest no siguin gaire sensibles a l'escalat de les dades, tenir les variables normalitzades pot ajudar a que el model entengui més fàcilment quines són les features més importants.

Un cop normalitzats els train també normalitzem la partició de validació i test amb minmax perquè així més endavant l'algoritme predigui millor els resultats.

4. Entrenament de Models (Classificador)

4.1. Definició de mètriques

En aquest apartat escollirem quina mètrica utilitzarem per entrenar i comparar els resultats dels diferents models. Abans d'escollir la mètrica cal pensar quin és l'objectiu, que volem maximitzar o minimitzar. En el nostre cas el que volem és que el model acerti totes les prediccions que fa i que no es deixi cap client churned sense identificar.

En l'enunciat de la pràctica se'ns demana dir a l'empresa amb quina precisió podem predir la gent que marxa. Malgrat això, d'acord amb l'objectiu marcat, considero que aquesta mètrica no es prou completa. Per exemple, si a l'empresa han marxat 200 persones però el nostre algorisme només prediu que han marxat 2 i aquests 2 efectivament han marxat la precisió de churned serà 100% però el nostre model realment estarà deixant passar molts clients que marxaran de l'empresa. Com hem vist abans aquest no és el nostre objectiu.

Per solucionar aquest inconvenient necessitem tenir en compte també la mètrica de recall, aquesta tindrà en compte quants clients dels que marxen en la realitat està predint el model, en el cas anterior de 200 clients el recall seria 2/200, per tant seria molt baix i mostraria que el nostre model no és bo. Per combinar ambdues mètriques utilitzarem la mètrica de F1 score dels clients churned, aquesta es calcula com la mitjana harmònica entre la precisió i el recall.

4.2. Fem les variables dummy per els algoritmes que les requereixen

Abans de començar a entrenar qualsevol model, és important preparar les dades de manera que el model pugui aprendre de manera efectiva. Per aconseguir això hem de crear variables dummies de les variables categòriques, ja que tots els models podran treballar amb aquestes, d'altra manera al entrenar els models ens trobaríem amb alguns errors.

Les variables categòriques són variables que poden prendre un nombre limitat de valors, com ara "Si" o "No" per qualsevol variable o "vermell" o "verd" per al color d'un objecte. Aquestes variables no es poden utilitzar directament com a entrada a alguns models perquè no són numèriques. En lloc d'això, cal convertir-les a forma numèrica creant variables dummies.

Per crear variables dummies, podem fer servir la codificació one-hot. Això implica crear una nova columna binària (0 o 1) per a cada categoria única de la variable categòrica original. Per exemple, si tenim una variable categòrica amb tres categories: "vermell", "verd" i "blau", crearíem tres noves columnes binàries: "vermell_Yes", "verd_Yes " i "blau_Yes ". Si una observació pertany a la categoria "vermell", el valor de la columna " vermell_Yes " seria 1 i els valors de les columnes " verd_Yes " i " blau_Yes " serien 0. Pel que fa les variables que són binaries, després de generar les dummies borrarem una de les dues columnes generades perquè ens aportaran la mateixa informació i així restem complexitat a les dades.

Després de crear variables dummies per a totes les variables categòriques del nostre conjunt de dades, podem utilitzar les dades resultants com a entrada dels nostres models.

4.3.KNN

Hem entrenat tres algorismes KNN amb dades desbalancejades, oversampled i undersampled.

El primer algorisme KNN es va entrenar amb un conjunt de dades desbalancejat, on la classe 'Stayed' tenia significativament més exemples que l'altra. Aquesta situació pot fer que l'algorisme tingui dificultats per identificar la classe menys comuna. Per mitigar aquest inconvenient, durant l'entrenament de l'algoritme hem utilitzat l'hiperparàmetre weight. Aquest paràmetre de pes en un algorisme KNN és utilitzat per determinar com es ponderaran els diferents veïns en la predicció final, i hi ha diferents maneres de fer-ho depenent de les nostres necessitats.

- **Pes uniforme:** tots els veïns tenen el mateix pes en la predicció final. Això és útil si no tenim cap raó especial per donar més pes a uns veïns en comparació amb d'altres.
- **Pes basat en la distància:** els veïns més propers tenen més pes en la predicció final. Això és útil si considerem que els veïns més propers són més rellevants que els veïns més llunyans.
- **Pes personalitzat:** es pot establir un pes personalitzat per a cada veí en funció de les nostres necessitats específiques. Això pot ser útil si tenim una raó específica per donar més pes a uns veïns en comparació amb d'altres.

Nosaltres en el cas d'aquest KNN utilitzarem **distància** perquè ens ajudarà a evitar prediccions desequilibrades.

Pel que fa els altres **hiperparàmetres** utilitzarem la funció GridSearchCV.

Aquesta funció pren com a entrada el model que haguem definit (KNN en aquest cas), un conjunt de paràmetres a buscar prèviament definits a un diccionari i un conjunt de dades d'entrenament. Posteriorment realitza una cerca exhaustiva a través dels diferents valors dels paràmetres per trobar els millors valors per a aquell model i conjunt de dades.

Per fer això, GridSearchCV divideix el conjunt de dades d'entrenament en conjunts d'entrenament i de validació, i entrena el model amb diferents combinacions de paràmetres utilitzant el conjunt d'entrenament. Després valida el model amb el conjunt de validació i utilitza una mètrica de rendiment, aquesta mètrica de rendiment la podem escollir nosaltres, per tant com volem maximitzar el valor de la mètrica f1 en els clients churned crearem el nostre propi score que la funció GridSearchCV maximitzarà al buscar els hiperparàmetres, aquest score consisitirà en el valor de f1 en els clients churned i el definirem així:

```
def scorer(true,pred):
    return f1_score(pred,true, pos_label = 'Churned')

score = make_scorer(scorer, greater_is_better=True)
```

Utilitzarem aquesta mètrica sempre que utilitzem GridSearchCV.

Finalment, GridSearchCV selecciona la combinació de paràmetres que produeix el millor rendiment i retorna el model entrenat amb aquesta combinació de paràmetres. Així, podem estar segurs que estem utilitzant els millors paràmetres per a aquest model i aquest conjunt de dades. Utilitzarem aquesta funció per tots els models.

N_neighbors especifica el nombre de veïns més properes que s'utilitzaran per fer prediccions. Això vol dir que, quan es fa una predicció per a una mostra de dades nova, es mira el nombre especificat de veïns més properes a aquesta mostra i es fa la predicció en funció del majoritari de les etiquetes d'aquestes mostres veïnes.

Els paràmetres que li passarem a la funció en aquest cas són els següents:

n_neighbors	[5, 7, 9, 15,20,25]
weights	['distance']

Després de passar aquests paràmetres la funció escolleix que n_neighbors ha de ser 20.

Un cop seleccionats els paràmetres, entrenem el model, fem la predicció de les dades de validació i obtenim un f1_score dels clients que han marxat de 0.73 i una precisió de 0.91.

Aquests resultats no són gaire bons, veiem com tenim una precisió bona però un f1 score baix. Això pot venir donat per que l'assignació de pesos per equilibrar el comportament de l'algoritme no ha tingut gaire èxit i per aquesta raó el recall (nombre de cops que l'algoritme prediu churned) és molt baix i per conseqüència el f1 score també. Ara veurem els resultats amb dades d'entrenament prèviament balancejades.

El segon algorisme KNN es va entrenar amb un conjunt de dades balancejat amb oversampling, on es van afegir mostres addicionals de la classe menys comuna (Churned) per tal de millorar les prediccions de l'algorisme.

Per escollir els hiperparàmetres vam utilitzar la funció GridSearchCV amb els següents valors de n_neighbors:

n_neighbors	[5, 7, 9, 15,20,25]

Després de passar aquests paràmetres, la funció escolleix que n_neighbors ha de ser 5.

Un cop seleccionats els paràmetres, entrenem el model, fem la predicció de les dades de validació i obtenim un f1_score dels clients que han marxat de 0.78 i una precisió de 0.79.

En aquests resultats es pot veure una millora en el f1 score, balancejant la base de dades hem aconseguit augmentar el recall, i ara deixem passar menys clients que marxaran de l'empresa encara que ha disminuït força la precisió.

Finalment, el tercer algorisme KNN es va entrenar amb un conjunt de dades balancejat amb undersampling, on es van eliminar aleatòriament algunes mostres de la classe més comuna (Stayed).

Els paràmetres proporcionats a la funció GridSearchCV han estat:

n_neighbors	[5, 7, 9, 15,20,25]

Després de passar aquests paràmetres, la funció escolleix que n_neighbors ha de ser 9.

Un cop seleccionats els paràmetres, entrenem el model, fem la predicció de les dades de validació i obtenim un f1_score dels clients que han marxat de 0.81 i una precisió de 0.76.

Podem veure com en aquest últim entrenament obtenim un f1 score més alt. Veiem com disminueix una mica la precisió però a canvi aconseguim millorar el recall i per això aconseguim aquest f1 score tan alt.

En resum, hem entrenat tres algorismes KNN amb conjunts de dades balancejats de diferents formes per tal de veure amb quin balanceig obtenim millors resultats. Hem pogut veure com KNN amb undersampling és l'algoritme que millor mètrica F1 Score ens ha donat malgrat que la precisió fos la pitjor de les 3.

4.4. Decision Tree

En aquest estudi, hem entrenat tres models d'arbre de decisió utilitzant tres versions diferents d'un conjunt de dades: un conjunt de dades desequilibrat, un conjunt de dades balancejat amb oversampling i un conjunt de dades balancejat amb undersampling. El conjunt de dades desequilibrat consistia en les dades originals d'entrenament tal com es van recopilar, sense fer cap modificació per a abordar qualsevol desequilibri en la distribució de classes i compensar-la amb pesos. El conjunt de dades amb oversampling es va crear generant artificialment mostres addicionals per a la classe minoritària amb l'objectiu de balancejar la distribució de classes. El conjunt de dades amb undersampling es va crear eliminant mostres de la classe majoritària per a balancejar la distribució de classes.

Recordem que els arbres de decisió comença amb un node arrel, que representa tot el conjunt de dades. A partir d'aquest node, es fan dividir el conjunt de dades en subconjunts més petits basant-se en la valoració d'alguna característica. Així, cada subconjunt es converteix en un node fill del node arrel. Aquest procés es repeteix recursivament pels nodes fills fins que s'arriba a un node on tots els exemples pertanyen a la mateixa classe. Per guiar l'arbre i que aconsegueixo obtenir els millors resultats possibles ajustarem els seus hiperparàmetres.

Explicació dels hiperparàmetres que estudiarem:

Criterion especifica la mesura que s'utilitzarà per determinar la qualitat de les divisions d'un node en l'arbre. Hi ha dues opcions principals per a aquest paràmetre: la impuresa de Gini o la entropia.

Max_depth especifica la profunditat màxima que es pot arribar a tenir l'arbre. Això vol dir que, un cop s'arriba a aquesta profunditat, es deixa de dividir el node en nodes fills i es considera que l'arbre ha acabat de creixer.

Min_samples_leaf especifica el nombre mínim de mostres que s'han de tenir en un node per tal que aquest es pugui dividir en dos nodes fills. Aquest paràmetre es fa servir per evitar que l'arbre es creï massa gran i, per tant, es sobreajusti a les dades d'entrenament.

Min_samples_split especifica el nombre mínim de mostres que s'han de tenir en un node per tal que aquest es pugui dividir en dos nodes fills. Això vol dir que, si un node té menys de les mostres especificades per aquest paràmetre, no es dividirà més enllà d'aquest nivell.

Base de dades desbalancejada:

En aquest apartat entrenarem un Arbre de Decisió utilitzant pesos per evitar que l'algoritme aprengui el desequilibri que trobem a les dades d'entrenament.

Per aconseguir això utilitzarem el parametre **class_weight** en l'entrenament d'aquest.

El paràmetre class_weight a l'arbre de decisió de scikit-learn és una opció que es pot utilitzar per indicar al model que doni més pes a una classe en concret durant el procés d'entrenament.

Per exemple, en el nostre conjunt de dades d'entrenament amb desbalanceig, si no especifiquem class_weight, el model donarà molt més pes a les instàncies de la classe amb més dades i, per tant, pot tenir un rendiment pobre en la classe amb menys dades. Si especifiquem class_weight, el model farà un esforç addicional per equilibrar el pes de cada classe, cosa que pot millorar la seva capacitat per generalitzar a les dades de test.

El paràmetre class_weight pot ser un diccionari, on cada clau és el nom d'una classe i el valor associat és el pes que s'ha de donar a aquesta classe. Però nosaltres utilitzarem l'opció "balanced", que assigna automàticament pesos a cada classe en funció de la seva freqüència.

Hiperparàmetres proporcionats a la funció GridSearchCV:

criterion	['gini','entropy']
max_depth	[5,7,9,10]
min_samples_leaf	[7,9,10,15]
min_samples_split	[1,3,5]

Hiperparàmetres triats:

criterion	entropy
max_depth	7
min_samples_leaf	9
min_samples_split	1

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.84 i una precisió de 0.82. Podem veure com obtenim bons resultats i per tant l'assignació de pesos a funcionat correctament. També podem apreciar una millora respecte els resultats trobats al entrenar els algoritmes KNN.

Base de dades balancejada amb oversampling:

Per entrenar aquest algoritme no cal el paràmetre class_weight perquè la base de dades ha estat balancejada prèviament.

Hiperparàmetres proporcionats a la funció GridSearchCV:

criterion	['entropy']
max_depth	[9,10,12]
min_samples_leaf	[9,10,12]
min_samples_split	[1,3,5]

Utilitzem només entropy perquè després de diferents proves obté millors resultats i així restem complexitat a la recerca dels paràmetre òptims.

Hiperparàmetres triats:

criterion	'entropy'
max_depth	12
min_samples_leaf	10
min_samples_split	1

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.82 i una precisió de 0.84. Podem veure com obtenim resultats bons, molt semblants als anteriors.

Base de dades balancejada amb undersampling:

Per entrenar aquest algoritme tampoc cal el paràmetre class_weight perquè la base de dades ha estat balancejada prèviament.

Hiperparàmetres proporcionats a la funció GridSearchCV:

criterion	['entropy']
max_depth	[5,7,9,10]
min_samples_leaf	[5,10,15]
min_samples_split	[1,3,5]

Hiperparàmetres triats:

criterion	'entropy'
max_depth	7
min_samples_leaf	15
min_samples_split	1

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.83 i una precisió de 0.80. Tornem a obtenir resultats molt semblants, millorem una mica els resultats amb oversampling, però obtenim uns de pitjors que utilitzant pesos. La causa per la que la precisió redueix pot ser que la quantitat de dades d'entrenament és menor perquè hem fet undersampling.

4.5. Random forest

Hem entrenat tres arbres de decisió aleatoris amb la llibreria scikit-learn utilitzant tres conjunts de dades diferents: un amb desbalanceig , un altre amb oversampling i un altre amb undersampling. Això ens permetrà comparar el rendiment dels models amb cada conjunt de dades i veure quina és la millor estratègia per a les nostres dades.

Recordem que els RF es basen en l'ús de molts arbres de decisió per fer prediccions. Cada subarbre és entrenat amb un subconjunt diferent de les dades d'entrenament, i cada arbre fa les seves pròpies prediccions. Per fer prediccions es fan servir totes les prediccions dels arbres i es pren la majoria dels vots com a resultat final.

Base de dades desbalancejada:

En aquest apartat entrenarem un RF amb la base de dades desbalancejada pel que fa la variable Customer Status. Per evitar que el model aprengui d'aquest desbalanceig utilitzarem pesos a l'hora d'entrenar l'algoritme. Per fer això tornarem a utilitzar el paràmetre class_weight, però aquest cop amb el valor 'balanced_subsample', quan utilitzem aquest valor el model ajustarà els pesos de cada classe de manera que estiguin representats de manera igual a cada submostra de les dades d'entrenament que generarà l'algoritme de random forest. Balanced utilitzaria les freqüències generals per assignar pesos i no la de les submostres.

Hiperparàmetres proporcionats a la funció GridSearchCV:

criterion	['gini','entropy']
n_estimators	[50,100]
max_depth	[15,20,30]
min_samples_split	[10,15]

Hiperparàmetres triats:

criterion	'entropy'
n_estimators	100

max_depth	20
min_samples_split	15

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.82 i una precisió de 0.90. Podem veure com obtenim un f1_score semblant als que hem obtingut en els anteriors arbres de decisió entrenats, en canvi en aquest cas obtenim una precisió de 0.90 bastant millor que les que hem vist fins ara.

Base de dades balancejada amb oversampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

criterion	[‘gini’,'entropy']
n_estimators	[50,100]
max_depth	[10,15,20]
min_samples_split	[1,3,5]

Hiperparàmetres triats:

criterion	‘gini’
n_estimators	100
max_depth	20
min_samples_split	1

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.78 i una precisió de 0.92. Obtenim un f1 score pitjor que en el cas anterior però una presició millor, això implica que el fet de haver fet oversampling empitjora la puntuació de recall del nostre model.

Base de dades balancejada amb undersampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

criterion	[‘gini’,'entropy']
n_estimators	[50,100]
max_depth	[5,7,10]
min_samples_split	[1,3,5]

Hiperparàmetres triats:

criterion	‘entropy’

n_estimators	100
max_depth	7
min_samples_split	3

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.85 i una precisió de 0.82. Com podem veure hem obtingut el resultat més gran en f1_score fins ara. Per tant podem veure com fer undersampling per equilibrar les dades d'entrenament pot ser una bona opció.

4.6. XGBoost

Per utilitzar XGBoost hem hagut de fer dummies també les bases de dades de y_train i y_val ja que aquest fa prediccions de 1 i 0, no de Churned o Stayed.

Explicació dels hiperparàmetres que estudiarem:

Learning_rate: En els algorismes de boosting, es construeixen una sèrie de models d'arbre de decisió de manera seqüencial, on cada model intenta corregir els errors del model anterior. Això es fa a través d'un procés anomenat "gradient descent", on es busca el mínim d'una funció d'error. learning rate controla la mida del pas que es fa en aquest procés en cada iteració.

Base de dades desbalancejada:

Per entrenar l'algoritme amb pesos hem hagut d'utilitzar la funció class_weight.compute_sample_weight(). Aquesta funció ens permet assignar el pes necessari a cada mostra perquè aquesta sigui tractada amb menys o més importància per part de l'algoritme. Aquests pesos calculats li passarem al paràmetre sample_weights a l'hora d'entrenar l'algoritme XGBoost.

Hiperparàmetres proporcionats a la funció GridSearchCV:

learning_rate	[0.1, 0.5, 1]
max_depth	[4,5]
n_estimators	[100,150]

Hiperparàmetres triats:

learning_rate	0.1
max_depth	4
n_estimators	100

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.86 i una precisió de 0.87. Fins al moment han estat els millors resultats que hem obtingut.

Base de dades balancejada amb oversampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

learning_rate	[0.1, 0.5, 1]
max_depth	[4,5]
n_estimators	[100,150]

Hiperparàmetres triats:

learning_rate	0.1
max_depth	4
n_estimators	100

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.85 i una precisió de 0.86. Podem veure com el fet de fer oversampling ha fet que perdem una mica de precisió.

Base de dades balancejada amb undersampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

learning_rate	[0.1, 0.5, 1]
max_depth	[4,5]
n_estimators	[200,250]

Hiperparàmetres triats:

learning_rate	0.1
max_depth	3
n_estimators	200

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.87 i una precisió de 0.85. El valor de f1_score es el major que hem obtingut fins al moment, com és la mètrica que estem avaluant, podem dir que aquest es el model que millors resultats ens ha donat.

Després d'haver entrenat els models XGBoost podem veure com els resultats són millors que tots els que hem entrenat anteriorment.

Conclusió de l'us d'arbres:

Hem pogut apreciar que en els 3 models diferents d'arbres entrenats obtenim bons resultats. Hi ha diversos factors que han influït en la precisió i el f1 score, com per exemple, la selecció de les features més importants i l'ajustament dels hiperparàmetres per evitar l'overfitting i l'underfitting.

4.7.SVM Lineal

Per entrenar un model SVM lineal el que hem de fer es quan creem el model utilitzar el paràmetre kernel i assigar-li el valor 'linear'.

Recordem que els SVM es basen en la idea de trobar una línia (o hiperplà en SVM Radial) que separi les mostres de les diferents classes el millor possible.

Per trobar aquesta línia, les SVM busquen la línia que tingui el màxim marge entre les mostres més properes de cada classe, anomenats vectors de suport. Una vegada trobada aquesta línia, es pot fer servir per fer prediccions sobre noves dades.

Explicació dels hiperparàmetres que estudiarem:

C és un paràmetre de regularització que controla la força amb què s'evita l'error en l'aprenentatge. Això vol dir que, quant més gran és C, més es vol minimitzar l'error en l'aprenentatge, cosa que pot portar a un model més precís, però també pot augmentar la possibilitat d'overfitting a les dades d'entrenament. Si C és molt gran, es pot acabar penalitzant massa les mostres que no són correctament classificades, cosa que pot portar a un model massa complex i poc generalitzable a les dades de test. Si C és molt petit, es pot acabar donant massa marge d'error a les mostres que no són correctament classificades, cosa que pot portar a un model massa simple i poc precís.

Gamma és un paràmetre que controla la complexitat del model. Això es fa a través de l'amplada dels kernels utilitzats en l'aprenentatge. Si gamma és molt gran, es fan servir kernels molt estrets i, per tant, el model es pot acabar sobreajustant a les dades d'entrenament. Si gamma és molt petit, es fan servir kernels molt amplis i, per tant, el model es pot acabar simplificant massa i no ser prou precís.

Base de dades desbalancejada:

Abans d'entrenar el model per evitar que l'algoritme aprengui del desbalanceig de la base de dades utilitzarem el paràmetre class_weight de l'algoritme i li assignarem el valor 'balanced'

Hiperparàmetres proporcionats a la funció GridSearchCV:

C	[0.5, 1, 2]
gamma	['auto', 'scale']

Hiperparàmetres triats:

C	1
gamma	'auto'

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.83 i una precisió de 0.82. Veiem com els resultats no milloren els de XGboost.

Base de dades balancejada amb oversampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

C	[0.5, 1, 2]
gamma	['auto', 'scale']

Hiperparàmetres triats:

C	1
gamma	'auto'

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.85 i una precisió de 0.83. Veiem com els resultats milloren respecte els d'aquest mateix algorisme entrenat amb pesos. Els resultats són bons comparats amb la resta d'algoritmes.

Base de dades balancejada amb undersampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

C	[0.5, 1, 2]
gamma	['auto', 'scale']

Hiperparàmetres triats:

C	1
gamma	'auto'

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.82 i una precisió de 0.80. Veiem com hem perdut precisió respecte els altres dos SVM,

això pot venir donat perquè aquestes dades d'entrenament al tenir menys mostres aporten menys informació al model.

4.8.SVM Radial

Per entrenar un model SVM radial el que hem de fer es quan creem el model utilitzar el paràmetre kernel i assigar-li el valor 'rbf' ja que serà amb la tècnica que treballarem.

Recordem que en el cas d'aquest SVM, enlloc de buscar la línia que separa les dues classes com fèiem al lineal, buscarem l'hiperplà, ja que ens trobem en un espai de tres dimensions gràcies a l'ús de 'rbf'.

Base de dades desbalancejada:

Tornem a utilitzar class_weight = 'balanced'

Hiperparàmetres proporcionats a la funció GridSearchCV:

C	[1, 2,3]
gamma	['auto', 'scale']

Hiperparàmetres triats:

C	2
gamma	'scale'

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.83 i una precisió de 0.84.

Base de dades balancejada amb oversampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

C	[1, 2, 3]
gamma	['auto', 'scale']

Hiperparàmetres triats:

C	3
gamma	'scale'

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.82 i una precisió de 0.85. Obtenim resultats pitjors que utilitzant pesos.

Base de dades balancejada amb undersampling:

Hiperparàmetres proporcionats a la funció GridSearchCV:

C	[0.5, 1, 2]
gamma	['auto', 'scale']

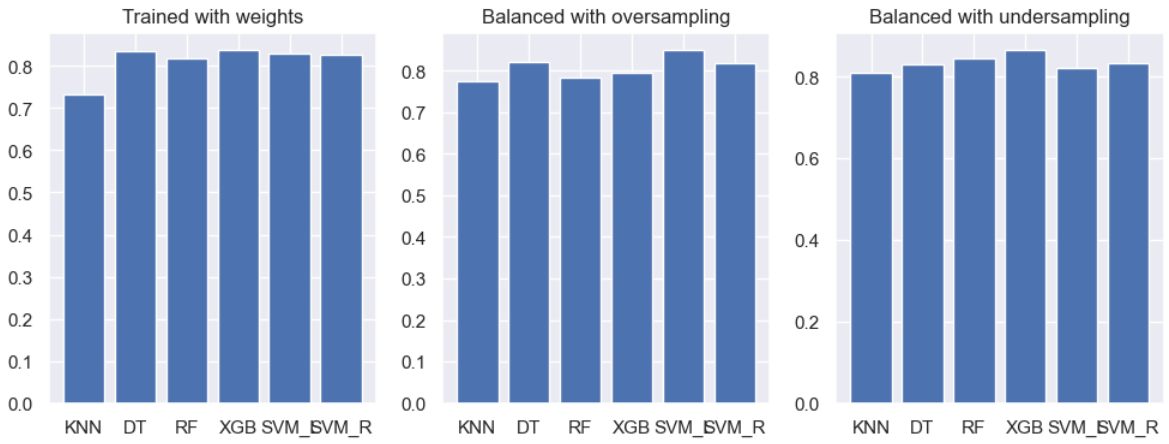
Hiperparàmetres triats:

C	2
gamma	'auto'

Resultats:

Com a resultat després de fer les prediccions de validació hem obtingut un f1 score dels clients Churned de 0.83 i una precisió de 0.81. Obtenim resultats semblants a si utilitzem pesos però amb menor precisió per el fet de treballar amb unes dades d'entrenament amb menys informació.

4.9. Decisió de model i resultats amb test



	KNN	DT	RF	XGB	SVM_L	SVM_R
Weighted	0.733	0.837	0.819	0.837	0.830	0.827
Oversampled	0.776	0.822	0.785	0.794	0.849	0.819
Undersampled	0.809	0.830	0.845	0.865	0.823	0.832

En aquest gràfic i taula podem veure un resum dels resultats que hem tingut en els 18 entrenaments que hem fet. A continuació extraurem les conclusions corresponents:

En primer lloc podem veure que els algoritmes de XGboost i DT són els que millors resultats han donat amb dades d'entrenament desbalancejades. Els dos arriben a un f1_score de 0.837 dels

clients que han marxat. D'aquest fet podem concloure també que assignar pesos a les mostres a l'hora d'entrenar un model és un procediment que funciona, ja que en general tots els algoritmes obtenen resultats semblants o inclús millors amb pesos que amb altres mètodes de balanceig com l'oversampling.

En segon lloc veiem com amb el mètode d'oversampling l'algoritme que millor ha treballat amb diferencia ha estat el SVM Linear el qual ha aconseguit un f1_score de la categoria Churned de 0.849. Pel que fa els resultats generals que hem obtingut en els models entrenats amb oversampling podem apreciar que han sigut els pitjors de les tres maneres amb les quals hem intentat mitigar el desbalanceig.

En tercer lloc es pot apreciar com l'algoritme que millors resultats obté amb la base de dades balancejada amb undersampling és XGBoost, aquest ha aconseguit arribar a un f1_score dels clients churned de 0.865, aquest resultat també és el millor entre tots els altres resultats obtinguts. També podem apreciar com el mètode d'undersampling ha sigut el que ha donat millors resultats entre els 3 que hem utilitzat.

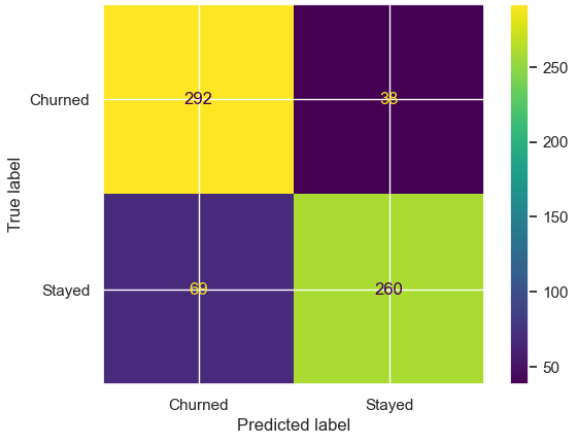
Finalment, un cop hem identificat l'algoritme i el mètode de balanceig que dona millors resultats en la partició de validació (XGBoost amb undersampling) anem a ajuntar les particions de train_undersampling amb val i a entrenar un XGBoost amb aquestes dades juntes. Posteriorment farem una predicció de les dades de test i analitzarem el comportament del nostre model aquestes dades completament noves.

A continuació es mostraran els resultats d'haver entrenat un XGBoost amb la partició de train balancejada amb undersampling ajuntada amb la partició de validation. Els hiperparàmetres utilitzats han estat els mateixos que hem obtingut anteriorment en l'apartat d'entrenament de XGBoost amb undersampling.

learning_rate	0.1
max_depth	3
n_estimators	200

Resultats finals:

Confusion matrix:



	precision	recall	f1-score	support
0	0.81	0.88	0.85	330
1	0.87	0.79	0.83	329

Tenint en compte que 0 identifica l'etiqueta Churned podem veure com hem obtingut un f1 score de 0.85 i una precisió de 0.81. Com podem veure els resultats obtinguts són bastant acceptables. Les raons per les quals aquests resultats són positius són les següents:

- El model de classificació utilitzat és robust i capaç de generalitzar bé a dades noves.
- Les dades utilitzades per entrenar el model són suficientment representatives de les dades que es trobaran a la partició de test, de manera que el model no es sobreajusti.
- S'ha utilitzat bé la partició de validació durant l'entrenament del model per assegurar que es generalitzi bé a dades noves.
- S'han utilitzat tècniques de preprocessament (eliminació de variables redundants, balanceig, normalització, imputació de missings,...) de dades adequades per millorar la qualitat de les dades d'entrenament i evitar problemes com la baixa variabilitat, el desbalanceig de les classes o la diferencia de rangs en les variables.
- S'han optimitzat adequadament els hiperparàmetres del model durant l'entrenament per millorar el rendiment.

5. Model Card

5.1. Detalls del Model

- Model creat per Gerard Gómez Izquierdo
- Data de creació del Model: 24/12/2022
- Primera versió del model
- Model XGBoost Classificador
- El model XGBoost Classifier ens ha permès classificar quins clients de l'empresa telefònica marxaran i quins no.
- Per qualsevol dubte del model enviar un correu a Gerard.gomez.izquierdo@estudiantat.upc.edu

5.2. Us per al que està destinat

- Aquest model ha estat creat principalment per fins acadèmics.
- Els usuaris que haurien de tractar amb aquest model haurien de ser alumnes i professors de la FIB
- Cal vigilar amb les prediccions d'aquest model perquè no acerta sempre, cal tenir en compte que no deixa de ser un programa i que si aquest model s'apliques a la vida real caldria algu que supervises les prediccions fetes.

5.3. Factors

- Les dades amb les que s'ha entrenat el model han estat balancejades amb undersampling.
- Els factors evaluatius han estat si la predicció l'estat del client era correcte o no.

5.4. Mètriques

- Per reflectir el potencial real del model utilitza la mètrica f1_score dels clients que han marxat.
- Els outliers no s'han tret ja que tots eren dades reals que a priori no empitjoren les decisions del model.
- Les dades s'han normalitzat amb minmax per aconseguir que les dades estiguin dins del rang [0,1]. No ha calgut fer gaussiana la distribució de les variables perquè els models entrenats no assumeixen distribucions de les dades.

5.5. Evaluation Data

- Abans de començar el preprocessament de les dades es va fer una partició del data set aleatori on el 80% l'ocupaven dades d'entrenament, el 10% dades de validació i l'altre 10% dades de test. Les dades de validació i test al fer la partició queden balancejades perfectament.
- Per analitzar els resultats s'ha normalitzat amb minmax les 3 particions del data set per separat.
- Aquestes particions s'han realitzat abans per evitar data leakage durant el preprocessament.

5.6. Training Data

- En l'apartat 1.2 del treball es pot veure un anàlisi detallat de les distribucions de les diferents variables que participen a la partició train.
- Les particions de train s'han balancejat amb dos mètodes diferents, oversampling i undersampling.

5.7. Ethical Considerations

- Per evitar biaixos relacionats amb el sexe vam eliminar la variable genere que contenia si la persona era home o dona.

5.8. Advertències i recomanacions

- Supervisar sempre per un humà les prediccions fetes per el model ja que poden estar equivocades.

6. Clustering

En aquesta apartat del treball ens endinsarem en el clustering, una tècnica de l'anàlisi de dades que ens permet agrupar diferents elements d'acord a les seves similituds. Utilitzarem dos mètodes de clustering: l'agrupament jeràrquic i el DBSCAN. A més, per poder visualitzar millor els resultats, reduirem la dimensionalitat dels nostres dades utilitzant l'algorisme PCA.

6.1. Hierarchical Clustering

L'agrupament jeràrquic és un tipus de clustering que es basa en la creació d'un arbre de clusters. Aquest arbre es construeix a partir de les dades que tenim i es va formant cap amunt a partir de aparellar els clusters més semblants.

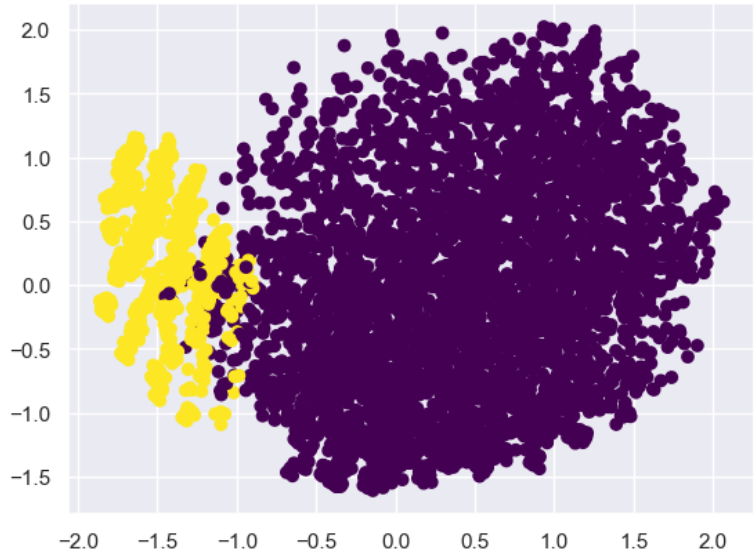
El procés comença considerant cada element de les dades com un cluster individual. A continuació, es busquen les parelles de clusters més semblants i es fusionen en un únic cluster. Això es fa iterativament fins que tots els elements queden agrupats en un únic cluster final.

Això es pot fer de diverses maneres, depenent de la mesura de distància que es triï entre els clusters i de com es decideixi fusionar-los. Algunes de les opcions més comunes són el mètode de l'enllaç més curt (o "single link"), el mètode de l'enllaç més llarg (o "complete link") i el mètode de enllaç de ward, el qual utilitzarem nosltres.

Una vegada tenim l'arbre de clusters, podem escollir en quin nivell de l'arbre volem tallar-lo per obtenir els diferents grups que ens interessin.

Procedim a entrenar l'algoritme amb la base de dades de train sense balancejar i posteriorment reduïm la dimensionalitat de les dades a 2 de manera que puguem veure les dades, si fem això obtenim les següents resultats:

Amb n_clusters = 2:



Veiem con diferencia un grup groc que trobem a l'esquerra del núvol de punts. Anem a veure si aquest núvol de punts groc representa algun grup de clients que puguem identificar.

Si analitzem quantes de les mostres que l'algoritme classifica com a grogues són clients que han marxat ens trobem el següent:

```
Stayed      1069
Churned       74
Name: Customer Status, dtype: int64
```

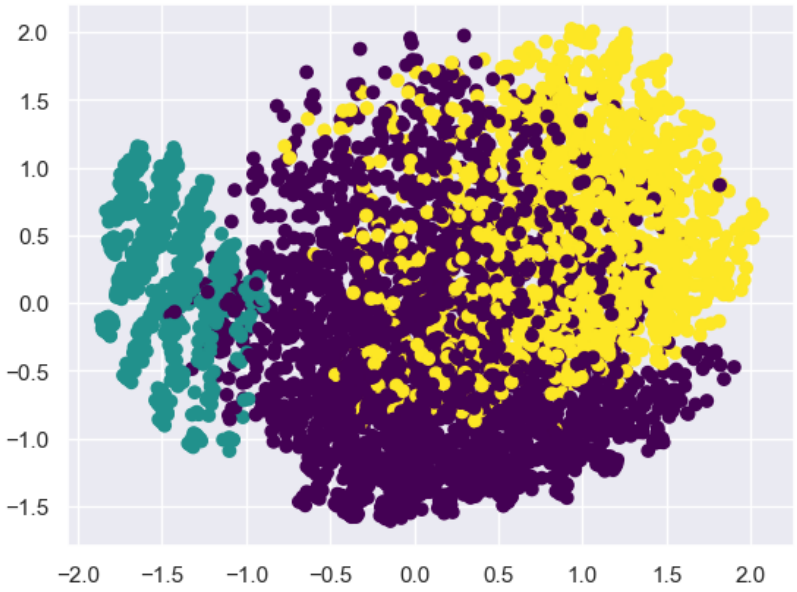
Podem veure com el 94% de les mostres grogues són clients que és queden a l'empresa.

Si analitzem les mostres liles trobem el següent:

```
Stayed      2993
Churned     1135
Name: Customer Status, dtype: int64
```

Veiem com en aquest núvol de punts trobem gairebé tots els clients que marxen però també una gran part dels que es queden

Amb n_clusters = 3:



Podem veure com s'ha creat un nou clúster de punts grocs, anem a veure el nombre de clients que es queden i marxen en aquest nou clúster.

```
Stayed      1242
Churned       82
Name: Customer Status, dtype: int64
```

Veiem com en aquest nou clúster el 94% dels clients tornen a ser clients que es queden a l'empresa.

Si analitzem el clúster lila veiem el següent:

```
Stayed      1751
Churned     1053
```

Seguim tenint un nombre molt elevat d'ambdós tipus de clients, però poc a poc sembla que a mesura que afegim clústers aquestes mostres liles comencen a representar més el tipus de client que marxa de l'empresa.

6.2. DBSCAN

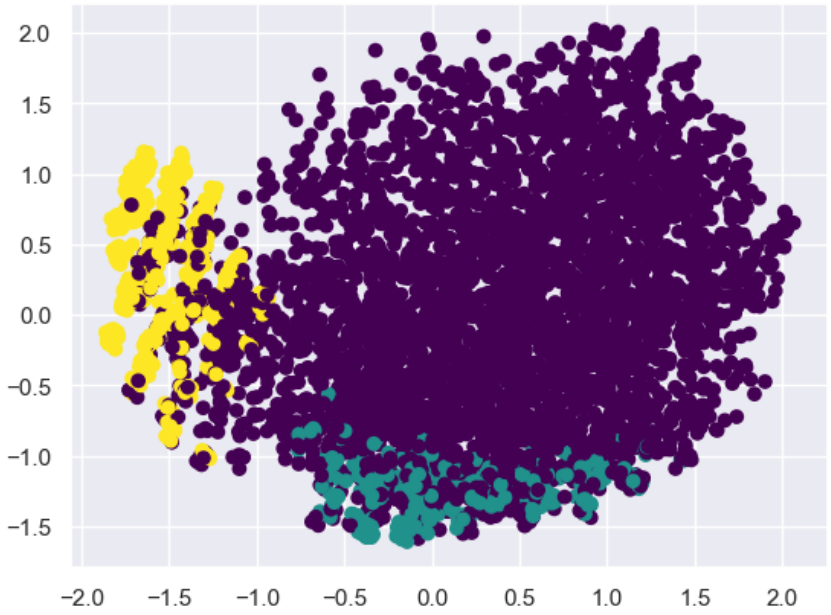
DBSCAN és un algorisme de clustering basat en densitat. Això vol dir que busca agrupar els punts dels nostres dades que es troben "prou a prop" els uns dels altres, deixant fora dels clusters els punts que es troben "massa allunyats" dels altres.

Per fer això, DBSCAN necessita dos paràmetres: una distància màxima (eps), que indica quina distància ha de tenir un punt amb altres punts per poder considerar-los "prou a prop", i un nombre mínim de punts (min_samples), que indica quants punts han de estar "prou a prop" per poder considerar que formen un cluster.

El funcionament de l'algorisme és el següent:

1. Es tria un punt de les nostres dades com a llavor i es mira quants punts més estan dins del radi de distància màxima (). Si n'hi ha prou, es considera que formen un cluster.
2. Si no s'ha arribat al nombre mínim de punts, el punt es considera que està en una zona que no pertany a cap cluster.
3. Si s'ha arribat al nombre mínim de punts, es consideren tots aquests punts com a part del cluster i es segueix explorant els punts veïns del cluster per veure si hi ha més punts que poden ser afegits.
4. Això es fa iterativament fins que ja no es troben més punts que puguin ser afegits al cluster.
5. A continuació, es tria un altre punt dels nostres dades que encara no s'hagi visitat i es torna a començar el procés des de l'etapa 1, fins que s'han visitat tots els punts dels nostres dades.

Procedim a executar el DBSCAN amb els paràmetres eps=1.5 i min_samples=75 i obtenim els següents resultats:



Veiem com torna a aparèixer el clúster de l'esquerra de mostres grogues, en canvi podem veure com ara ha aparegut un tercer clúster sota de les mostres liles. Anem a analitzar els valors de la variable objectiu a cada clúster:

Clúster groc:

```
Stayed     859
Churned     53
Name: Customer Status, dtype: int64
```

Tornem a veure com la gran majoria dels clients del clúster groc són clients que han marxat.

Cluster Blau:

```
Churned    296
Stayed     134
Name: Customer Status, dtype: int64
```

Veiem com trobem tant clients que han marxat com que s'han quedat. Encara així podem veure com augmenta la proporció de clients que marxen respecte la original de les dades d'entrenament.

Cluster Lila:

```
Stayed     3069
Churned     860
Name: Customer Status, dtype: int64
```

Veiem com trobem tant clients que han marxat com que s'han quedat. La proporció de clients que marxen es gairebé la mateixa que en la partició train original així que no sembla que aquest clúster contingui informació sobre la variable objectiu.