Airline Passenger Satisfaction

Progetto ML & AI

Emanuele Carta

Sommario

1 Introduzione

I dati che andremo ad utilizzare riguardano i risultatati di una survey per rilevare il livello di soddisfazione dei passeggeri di una compagnia aerea a noi non specificata.

Gli obbiettivi del nostro progetto sono:
- Identificare i fattori più rilevanti per la soddisfazione;
- Predire la soddisfazione in base ai fattori considerati.

Al fine di raggiungerli, ci avvarremo della Regressione Logistica.

1.2 *La Regressione Logistica nel Machine Learning*

La **Regressione Logistica** è uno degli algoritmi di Machine Learning più popolari, che rientra nella tecnica di **Supervised Learning**. È il metodo di riferimento per i problemi di **classificazione binaria**. Viene utilizzato per prevedere la variabile categorica dipendente utilizzando un determinato insieme di variabili indipendenti.
Nella **Regressione Logistica**, invece di adattare una retta di regressione, inseriamo una funzione logistica a forma di "S", che prevede due valori massimi (0 o 1).
La **Regressione Logistica** può essere classificata in tre tipi:
- **Binomiale**: nella regressione logistica binomiale, possono esserci solo due possibili tipi di variabili dipendenti;
- **Multinomiale**: nella regressione logistica multinomiale, possono esserci tre o più, non ordinate, possibili tipi di variabili dipendenti**;**
- **Ordinale**: nella regressione logistica multinomiale, possono esserci tre o più, ordinate , possibili tipi di variabili dipendenti**.**

Per i nostri fini, la scelta della **Regressione Logistica** è perfetta perché permette facilmente di eseguire la classificazione e di effettuare previsioni.
Per lo svolgimento del nostro progetto useremo la **Regressione Logistica Binomiale**.

2 Dataset

I nostri dati sono contenuti in due dataset diversi, uno che andremo ad utilizzare per il training (***airline-training***) e uno per il testing (***airline-test***).

Il dataset ***airline-training*** è composto da **103.904** righe e da **25** colonne, invece ***airline-test*** è composto da **25.976** righe e **25** colonne.

Volendo scoprire la dimensione del dataset "originario" sommiamo le righe dei due dataset ottenendo **129.880** istanze. Grazie a questo calcolo troviamo facilmente che il Test set è il **20%** del dataset "originario" e per il Training set è l' **80%**.

Andiamo ora a descrivere le colonne presenti in entrambi i nostri dataset.

Nome	Descrizione
Gender	*Sesso del passeggero* (**Male**, **Female**)
Customer Type	*Tipo di passeggero* (**Loyal custumer**, **Disloyal costumer**)
Age	*Anni del passeggero*
Type of Travel	*Causa del viaggio del passeggero* (**Business**, **Personal**)

Class	*Tipo di classe scelta dal passeggero* (**Eco**, **Eco Plus**, **Business**)
Flight distance	*Distanza del volo*
Inflight wifi service	*Livello di soddisfazione del wifi* (**0 : non selezionabile** ;**1-5**)
Departure/Arrival time convenient	*Livello di soddisfazione di convenienza orario di partenza/arrivo*
Ease of Online booking	*Livello di soddisfazione della facilità di prenotazione online* (**0-5**)
Gate location	*Livello di soddisfazione della posizione del Gate* (**0-5**)
Food and drink	*Livello di soddisfazione del cibo e delle bevande* (**0-5**)
Online boarding	*Livello di soddisfazione dell'imbarco* (**0-5**)
Seat comfort	*Livello di soddisfazione del confort del sedile* (**0-5**)
Inflight entertainment	*Livello di soddisfazione dell'intrattenimento in volo* (**0-5**)
On-board service	*Livello di soddisfazione del servizio abbordo* (**0-5**)
Leg room service	*Livello di soddisfazione dello spazio delle gambe* (**0-5**)
Baggage handling	*Livello di soddisfazione della gestione dei bagagli* (**0-5**)
Check-in service	*Livello di soddisfazione del servizio di check-in* (**0-5**)
Inflight service	*Livello di soddisfazione del servizio a terra* (**0-5**)
Cleanliness	*Livello di soddisfazione riguardo alla pulizia* (**0-5**)
Departure Delay in Minutes	*Ritardo nella partenza* (**in minuti**)
Arrival Delay in Minutes	*Ritardo nell'arrivo* (**in minuti**)
Satisfaction	*Soddisfazione riguardo la compagnia aerea* (***Satisfied, neutral or dissatisfied***)

Svolgiamo ora una veloce categorizzazione delle colonne a nostra disposizione, dividendole in: *categoriche*, *numeriche*, *binarie* e *non binarie*.

categoriche	Gender, Customer Type, Type of Travel, Class, Inflight wifi service, Departure/Arrival time convenient, Ease of Online booking, Food and drink, Online boarding, Seat comfort, Inflight entertainment, On-board service, Leg room service, Baggage handling, Checkin service, Inflight service, Cleanliness
numeriche	Age, Flight Distance, Departure Delay in Minutes, Arrival Delay in Minutes
binarie	Gender, Customer Type , Type of Travel
non binarie	Class, Inflight wifi service, Departure/Arrival time convenient, Ease of Online booking, Food and drink, Online boarding, Seat comfort, Inflight entertainment, On-board service, Leg room service, Baggage handling, Checkin service, Inflight service, Cleanliness, Age, Flight Distance, Departure Delay in Minutes, Arrival Delay in Minutes, Gate Location

Sono presenti **4** colonne *numeriche* e **18** colonne *categoriche* di cui **3** *binarie* e **15** *non binarie*.

2.1 *Modifica della Tipologia di Dati*

Prima di andare a svolgere l'analisi preliminare e le varie operazioni di pulizia, modificheremo il tipo dei dati a nostra disposizione.
Le colonne che saranno modificate sono:
Gender, *Customer Type*, *Type of Travel*, *Class*, *Inflight wifi service*, *Departure/Arrival time convenient*, *Ease of Online booking*, *Gate location*, *Food and drink*, *Online boarding*, *Seat comfort*, *Inflight entertainment*, *On-board service*, *Leg room service*, *Baggage handling*, *Check-in service*, *Inflight service*, *Cleanliness*.

Come visto prima queste colonne contengono feature *binarie* o *non binarie*.
La modifica renderà le feature variabili categoriche statistiche, cioè variabili che assumono un numero limitato, e generalmente fisso, di valori possibili. Nell'ambiente di programmazione Python ci avvarremo della libreria *Pandas* che ci permette facilmente di svolgere questa azione. Invece in R andremo a convertire le nostre colonne d'interesse in *Factor*, senza l'uso di alcuna libreria specifica, permettendoci quindi memorizzare i nostri dati come vettori di valori interi.

2.2 *I nostri Dataset sono Bilanciati?*

Controlliamo se i nostri dataset sono bilanciati, cioè se contengono un ugual numero di istanze per ogni classe.
In caso di forte sbilanciamento dovremo effettuare un operazione di ribilanciamento. Tale fenomeno può provocare notevoli problemi nella classificazione, poiché tende ad assegnare più istanze alla classe target maggioritaria.

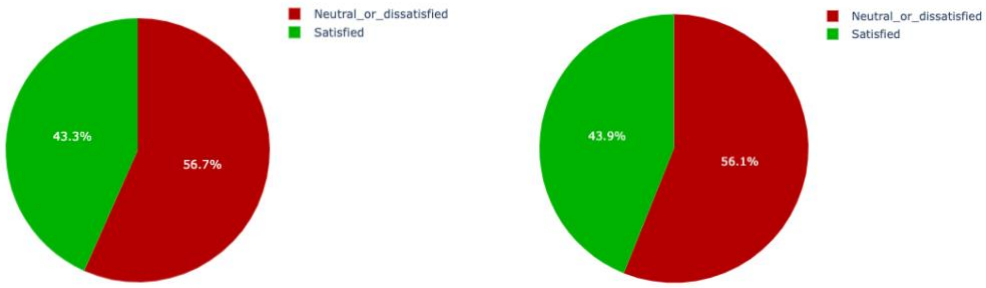

Figura 1: Grafici a torta Train set (sinistra) e Test set (destra)

Come si nota nella **Figura 1** i nostri dati, in entrambi i dataset, sono leggermente sbilanciati verso la categoria "*Neutral or dissatisfied*". Poiché lo sbilanciamento non supera il **60%** decidiamo però di non effettuare nessun ribilanciamento.

3 Analisi dei Dati

Iniziamo quindi a svolgere la nostra analisi descrittiva sui dati. Tutte le analisi verranno effettuate solo sul dataset di training.

Il nostro obbiettivo per queste analisi sarà quello di portare a galla le variabili poco significative, per poi rimuoverle, dandoci la possibilità di avvantaggiarci sia dal punto di vista temporale, che dal punto di vista computazionale.
Iniziamo la nostra analisi descrivendo le nostre variabili numeriche.

	Age	**Flight Distance**	**Departure Delay in Minutes**	**Arrival Delay in Minutes**
mean	39.37	1189.44	14.81	15.17
std	15.11	997.14	38.23	38.69
min	7	31	0	0
25%	27	414	0	0
50%	40	843	0	0
75%	51	1743	12	13
max	85	4983	1592	1584

Partendo dalla colonna *Age,* notiamo subito che i clienti della compagnia aerea hanno un'età media di **39.37** anni; possiamo ipotizzare così che la compagnia sia usata più spesso da persone adulte, che quindi potrebbero svolgere viaggi principalmente di lavoro.
Passando alla colonna *Flight Distance*, salta all'occhio che il chilometraggio medio dei voli è **1189.44** km, una distanza molto elevata per riguardare viaggi intra Europei, quindi potremmo pensare alla possibilità di avere tra le mani dati di una compagnia aerea extra Europea. Essendo la distanza media molto lunga, immaginiamo che il confort nel viaggio, ma soprattutto i ritardi nell'arrivo, possano influenzare molto la soddisfazione del cliente.
Nelle colonne *Departure Delay in Minutes* e *Arrival Delay in Minutes* la media è di circa **15** minuti, che come ritardo, non particolarmente significativo.

Passiamo ora alla visualizzazione delle variabili categoriche.

	unique	**top**	**freq**
Gender	2	*Female*	52727
Customer Type	2	*Loyal Customer*	84923
Type of Travel	2	*Business Travel*	71655
Class	3	*Business*	49665
Inflight wifi service	6	3	25868
Departure/Arrival time convenient	6	4	25546
Ease of Online booking	6	3	24449
Food and drink	6	4	24359
Online boarding	6	4	30762
Seat comfort	6	4	31765
Inflight entertainment	6	4	29423
On-board service	6	4	30867
Leg room service	6	4	28789
Baggage handling	6	4	37383
Checkin service	6	4	29055
Inflight service	6	4	37945
Cleanliness	6	4	27179

Durante l'analisi delle variabili numeriche abbiamo ipotizzato, data l'età media di circa **40** anni, la possibilità che la maggior parte dei viaggi fatti da questa compagnia avvengano principalmente per ragioni lavorative. Questa ipotesi viene confermata dai dati presenti nella colonna *Type of Travel* dove la feature più presente è "*Business Travel*", cioè i viaggi di lavoro.
Notiamo inoltre che la maggior parte dei dati riguardano "*Loyal Customers*", cioè viaggiatori che usufruiscono abitualmente della compagnia aerea, e clienti che viaggiano nella Business Class, quindi persone che pagano di più per poter viaggiare con maggiori vantaggi e miglior comfort.
Per visualizzare al meglio i dati e riuscire ad estrarre più informazioni possibili, utilizzeremo i grafici di seguito riportati.
Sia in Python che in R gli output dei plot sono identici; di conseguenza, per una questione puramente estetica e di gestione di spazio, andremo a visualizzare solo i grafici prodotti con Python.

Partiamo con l'osservare la soddisfazione dei clienti in base al sesso, tipo di cliente e tipo di viaggio.

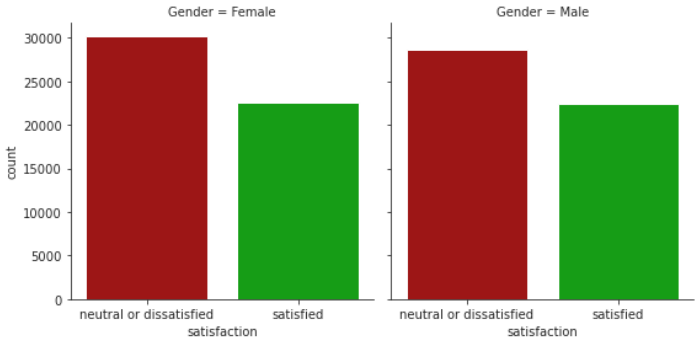

Figura 2: Grafico a barre di Gender in base alla satistaction

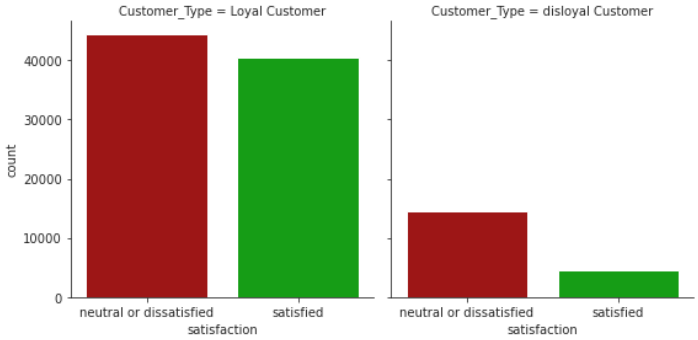

Figura 3: Grafico a barre del Customer Type in base alla satistaction

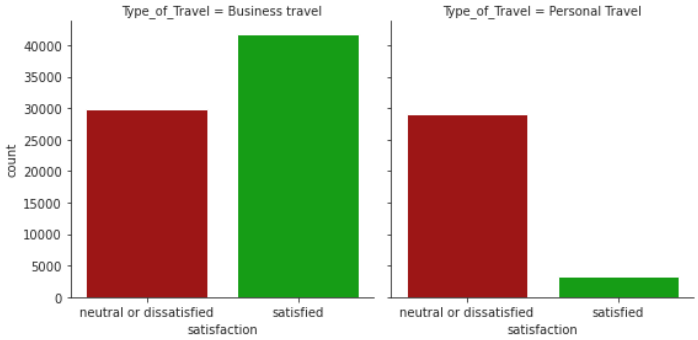

Figura 4: Grafico a barre del Type of Travel in base alla satistaction

La **Figura 2** fa emergere in modo chiaro la non importanza del sesso del cliente nella sua soddisfazione. Quindi la prima colonna che andremo a eliminare durante le operazioni sarà quella del *Gender*.
Dalla **Figura 3**, si nota come la compagnia aerea abbia molti clienti fedeli, ma allo stesso tempo non soddisfatti o neutri. La possibilità che questo leggero dislivello sia correlato allo sbilanciamento del dataset non è da escludere. Anche guardando però i clienti non abituali, cioè "*disloyal Custumers*", vediamo che la situazione non cambia.
Nella **Figura 4**, dove confrontiamo *Type of Travel* con la soddisfazione del cliente, notiamo grandi differenze: infatti i clienti che effettuano viaggi lavorativi sono in maggioranza soddisfatti dai vari servizi offerti dalla compagnia, al contrario, tra gli usuari che viaggiano per ragioni personali, gli insoddisfatti arrivano ad essere sei volte tanto gli utenti contenti del servizio.

Passiamo ora a vedere se l'età, che, come abbiamo detto prima, varia principalmente tra i 20 e i 60 anni, è significativa nella soddisfazione.

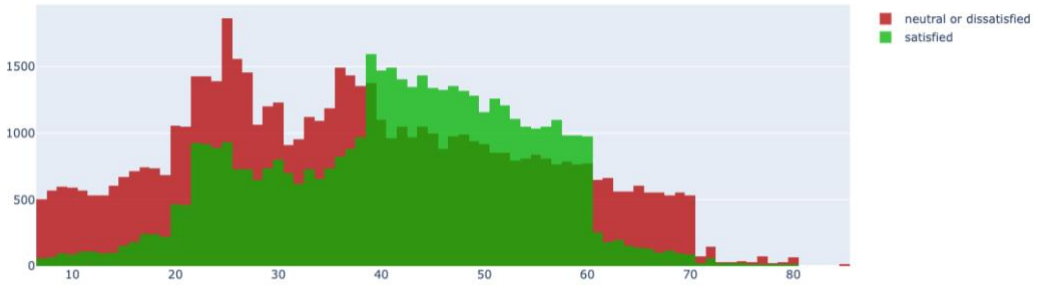

Figura 5: Grafico delle Distribuzioni della variabile Age in base alla satisfaction

Si nota facilmente che i clienti con età compresa tra i **20** e i **38** anni sono meno soddisfatti rispetto a quelli della fascia **41-60** anni.

Vediamo ora in modo più veloce le altre variabili alla ricerca di colonne poco significative.

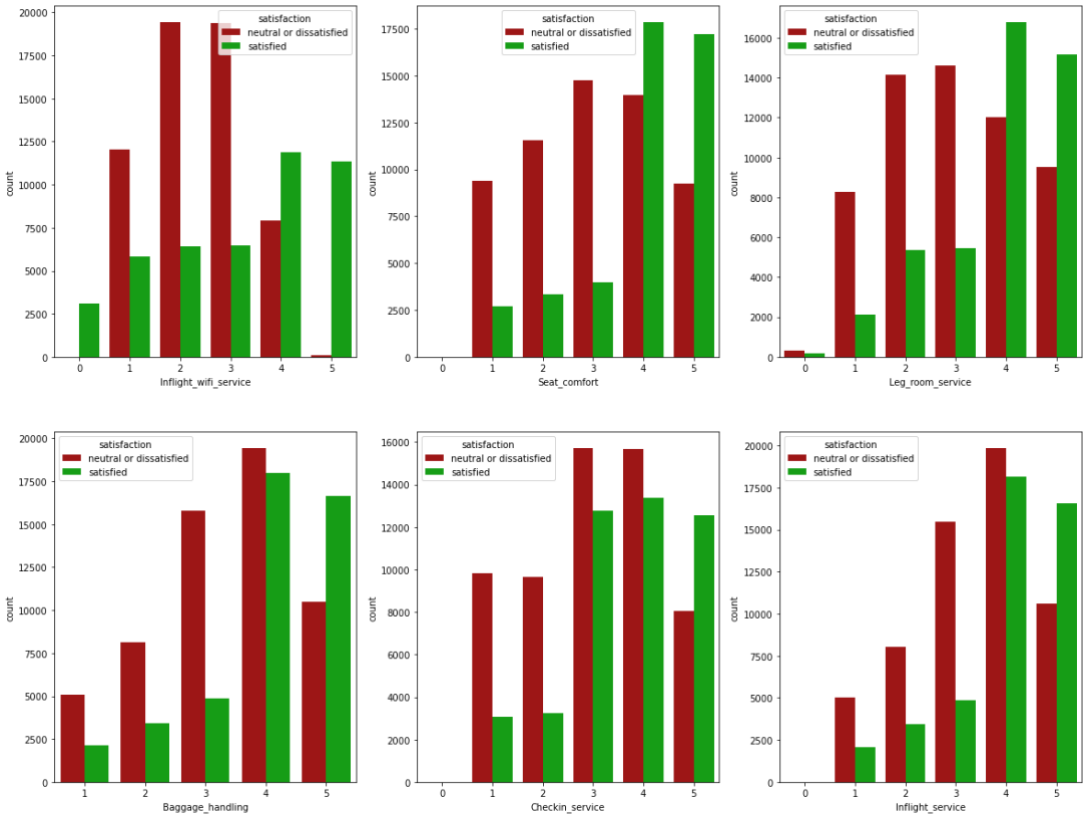

Figura 6.1: Grafici a Barre di tutte colonne in base alla satisfaction

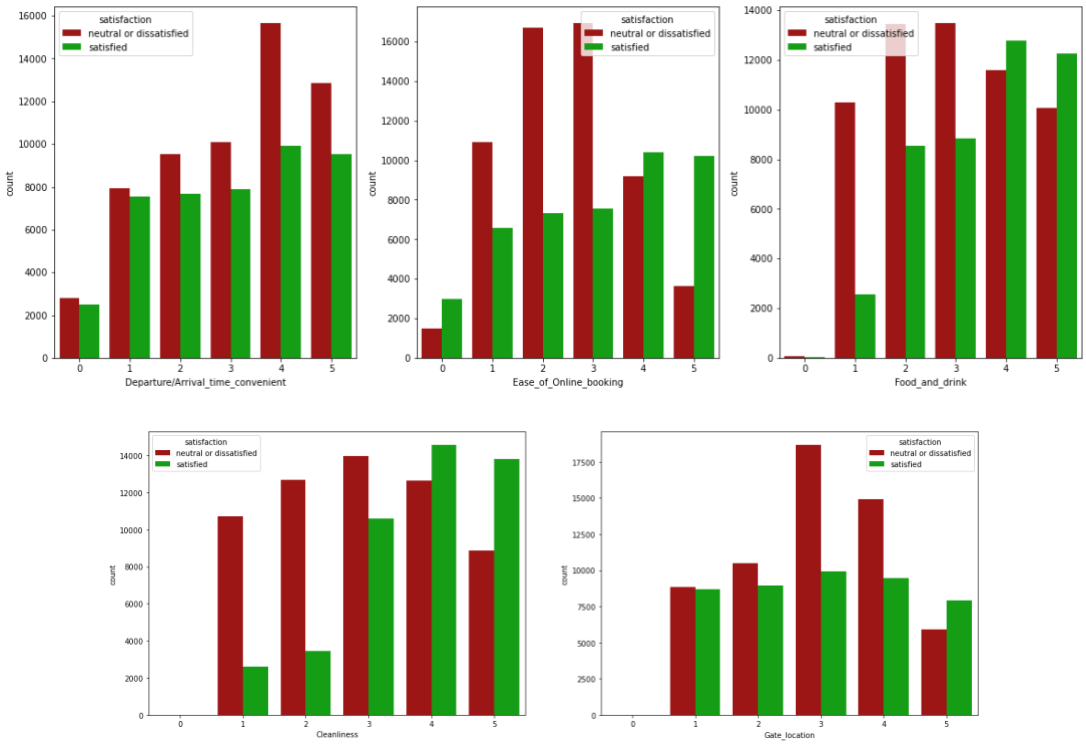

Figura 6.2: Grafici a Barre di tutte colonne in base alla satisfaction

La **Figura 6.1** e la **Figura 6.2**, composte entrambe da grafici a barre delle colonne, evidenziano, in particolare, due variabili che potrebbero essere superflue per la nostra classificazione e predizione, ovvero: *Gate location* e *Departure/Arrive time Convenient*.

L'ultima analisi che andremo a svolgere sul nostro set di dati riguarda la variabile *Class*.

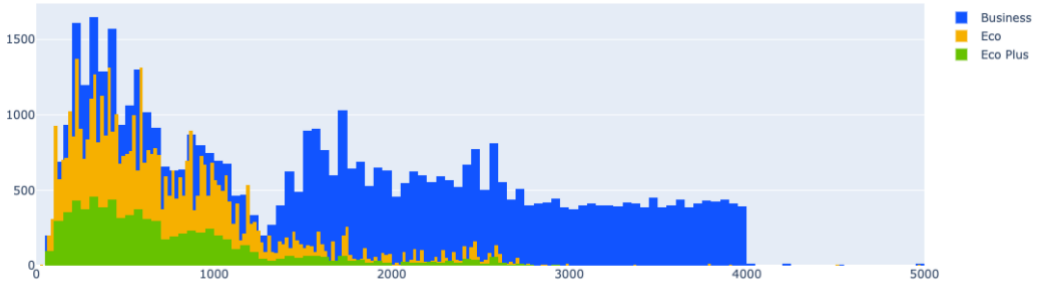

Figura 6: Grafico delle Distribuzioni della colonna Class in base al Fight Distance

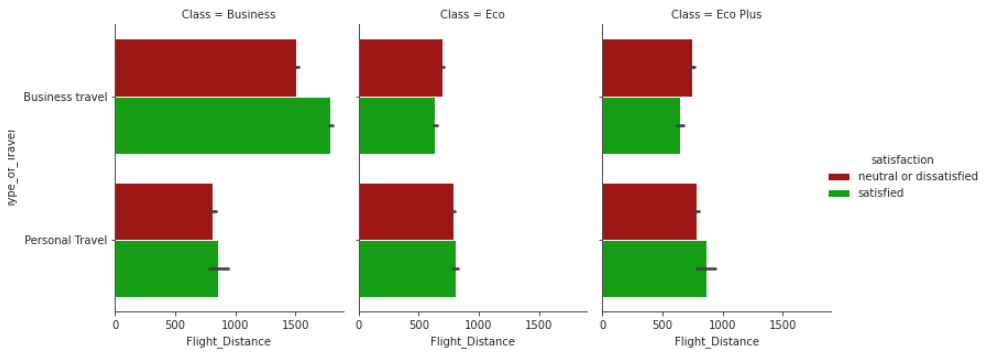

Figura 7: Grafico a Barre diviso per ogni variabile categorica di Class, per Type of Travel, Fight Distance e colorato dalla satisfaction

Come vediamo dalla **Figura 6**, più la distanza aumenta, più i clienti scelgono la business class. La **Figura 8** ci dà un'altra panoramica sulla situazione vista nella **Figura 4**, da ciò intuiamo che la distanza nel volo potrebbe avere una forte incidenza sulle nostre variabili.

4 Operazioni sui Dataset

Per le operazioni divideremo il paragrafo in due parti. Nella prima illustreremo le operazioni fatte su Python, dopodiché quelle effettuate su R.

Operazioni in Python

La prima operazione che siamo andati a svolgere sui nostri dati è quella di rimozione delle colonne *Gender*, *Gate Location* e *Departure/Arrive time Convenient*, che precedentemente dalle nostre analisi abbiamo notato essere poco significative.
In seguito abbiamo proceduto con la ricerca dei valori mancanti. Troviamo che sono presenti **310** *NA* nel dataset di training e **83** *NA* in quello di testing; tutti questi valori presenti nella colonna *Arrival Delay in Minutes* di ambi i dataset. Decidiamo quindi di sostituirli con la media delle loro colonne, cioè **15.17** minuti nel primo dataset e **14.74** minuti nel secondo.

Finite le operazioni di pulizia ci concentriamo su quelle di modifica. Su entrambi i dataset abbiamo applicato gli stessi metodi: quindi, per comodità, ci rivolgeremo all'oggetto delle modifiche al singolare.
Il primo cambio che è stato svolto è quello di rendere binaria la colonna *satisfaction*, cioè la nostra variabile target, trasformando le istanze "*neutral or dissatisfied*" in **0** e "*satisfied*" in **1**. In seguito, tramite l'utilizzo della libreria *sklearn* (che verrà usata anche per la costruzione dei modelli), applichiamo la funzione `LabelEncoder()` che permette di codificare le etichette di destinazione con un valore compreso tra **0** e $\mathbf{n_{classi}}$ **-1**.

Terminato il processo di modifica, abbiamo poi eseguito la divisione dei dataset per la costruzione dei modelli:
nelle variabili `y_test` e `y_train` abbiamo inserito la nostra variabile target, rendendolo un array tramite la funzione di *Pandas* `to_numpy()`;
nelle variabili `X_test` e `X_train` abbiamo inserito tutte le feature.
Divisi in dataset, abbiamo normalizzato `X_test` e `X_train`, sempre con l'aiuto della libreria *sklearn*, e trasformati in array tramite la funzione `fit_trasform()`**.**

Operazioni in R

Come nella sezione Python, i primi procedimenti svolti sono stati l'eliminazione delle colonne e la ricerca degli *NA* con la loro sostituzione.

Anche qui si è andati a trasformare le istanze della colonna *satisfaction* in **0** e **1**.
L'operazione di normalizzazione è stata svolta tramite l'utilizzo della funzione `scale()` presente di base dei pacchetti di RStudio.

5 Creazione dei Modelli

Come detto nell'Introduzione, per raggiungere il nostro obbiettivo, useremo la Regressione Logistica.

5.1 Modelli in Python

Il nostro modello di regressione verrà creato inserendo tutte le variabili che abbiamo deciso di mantenere.

```
Optimization terminated successfully.
Current function value: 0.348218
Iterations 7
Logit Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:               103904
Model:                          Logit   Df Residuals:                   103885
Method:                           MLE   Df Model:                           18
Pseudo R-squ.:                 0.4911   Log-Likelihood:                -36181.
converged:                       True   LL-Null:                       -71094.
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
x1            -0.8020      0.011    -73.772      0.000      -0.823      -0.781
x2            -0.1189      0.010    -11.765      0.000      -0.139      -0.099
x3            -1.3104      0.013   -102.506      0.000      -1.336      -1.285
x4            -0.3286      0.011    -28.973      0.000      -0.351      -0.306
x5             0.0219      0.011      1.912      0.056      -0.001       0.044
x6             0.4686      0.015     30.579      0.000       0.439       0.499
x7            -0.3271      0.014    -23.994      0.000      -0.354      -0.300
x8            -0.0770      0.014     -5.374      0.000      -0.105      -0.049
x9             0.8494      0.013     63.653      0.000       0.823       0.876
x10            0.0997      0.014      6.983      0.000       0.072       0.128
x11            0.1311      0.018      7.130      0.000       0.095       0.167
x12            0.3715      0.012     29.909      0.000       0.347       0.396
x13            0.3711      0.011     35.083      0.000       0.350       0.392
x14            0.1747      0.013     13.548      0.000       0.149       0.200
x15            0.3957      0.011     37.592      0.000       0.375       0.416
x16            0.1478      0.014     10.941      0.000       0.121       0.174
x17            0.2852      0.016     17.792      0.000       0.254       0.317
x18            0.1581      0.034      4.663      0.000       0.092       0.225
x19           -0.3421      0.034    -10.074      0.000      -0.409      -0.276
==============================================================================
```

Come vediamo delle **19** features che abbiamo deciso di usare per eseguire la nostra regressione, **18** hanno un **P < 0.05**. La feature che ha **0.056** è *Flight Distance*, che rimuoviamo poiché poco significativa.

Realizziamo quindi il nostro secondo modello eliminando la colonna *Flight Distance*. I risultati che si ottengono sono pressoché uguali, non vi sono grandi variazioni: di conseguenza, decidiamo di non esporre l'output della descrizione del modello.

5.2 Modelli in R

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.6038  -0.2397  -0.0525   0.1336   4.2273

                                 Estimate Std. Error z value Pr(>|z|)
(Intercept)                      -4.93880    0.10198 -48.430  < 2e-16 ***
Customer_Typedisloyal Customer   -3.30344    0.04767 -69.304  < 2e-16 ***
Age                              -0.05741    0.01510  -3.802 0.000144 ***
Type_of_TravelBusiness travel     4.52116    0.05246  86.186  < 2e-16 ***
ClassBusiness                     0.91821    0.06000  15.304  < 2e-16 ***
ClassEco                          0.24179    0.05792   4.174 2.99e-05 ***
Flight_Distance                   0.01451    0.01505   0.964 0.335248
Inflight_wifi_service2            0.08118    0.05240   1.549 0.121337
Inflight_wifi_service4            1.50041    0.04776  31.417  < 2e-16 ***
Inflight_wifi_service1            0.34705    0.06098   5.691 1.26e-08 ***
Inflight_wifi_service5            7.58062    0.13419  56.492  < 2e-16 ***
Inflight_wifi_service0           24.53002   88.81948   0.276 0.782411
Ease_of_Online_booking2          -0.24506    0.04924  -4.977 6.47e-07 ***
Ease_of_Online_booking5          -0.70574    0.05598 -12.606  < 2e-16 ***
Ease_of_Online_booking4           0.20167    0.04659   4.328 1.50e-05 ***
Ease_of_Online_booking1          -0.27593    0.05416  -5.095 3.49e-07 ***
Ease_of_Online_booking0          -3.68467    0.96487  -3.819 0.000134 ***
Food_and_drink1                  -0.06666    0.06748  -0.988 0.323253
Food_and_drink2                   0.20289    0.05583   3.634 0.000279 ***
Food_and_drink4                   0.13921    0.05386   2.585 0.009746 **
Food_and_drink3                   0.10451    0.05493   1.903 0.057085 .
Food_and_drink0                   0.11934    1.82054   0.066 0.947737
Online_boarding5                  2.92531    0.05698  51.343  < 2e-16 ***
Online_boarding2                  0.15112    0.05417   2.790 0.005278 **
Online_boarding1                  0.06259    0.06484   0.965 0.334421
Online_boarding4                  1.74673    0.04113  42.468  < 2e-16 ***
Online_boarding0                  3.57216    0.96968   3.684 0.000230 ***
Seat_comfort1                     0.14318    0.07003   2.045 0.040892 *
Seat_comfort2                    -0.37830    0.06405  -5.906 3.51e-09 ***
Seat_comfort3                    -1.47286    0.05576 -26.415  < 2e-16 ***
Seat_comfort4                    -0.80427    0.04778 -16.832  < 2e-16 ***
Seat_comfort0                   -20.16008 6522.63860  -0.003 0.997534
Inflight_entertainment1          -0.62536    0.09442  -6.623 3.51e-11 ***
Inflight_entertainment2           0.13885    0.08396   1.654 0.098154 .
Inflight_entertainment3           0.97465    0.07466  13.055  < 2e-16 ***
Inflight_entertainment4           0.66103    0.06813   9.703  < 2e-16 ***
Inflight_entertainment0         -40.16728 1534.52821  -0.026 0.979117
On_board_service1                -0.73544    0.05929 -12.404  < 2e-16 ***
On_board_service2                -0.62112    0.05653 -10.987  < 2e-16 ***
On_board_service3                -0.09405    0.04430  -2.123 0.033761 *
On_board_service5                 0.49092    0.04836  10.152  < 2e-16 ***
On_board_service0                22.03932 3983.22496   0.006 0.995585
Leg_room_service5                 0.92542    0.04544  20.367  < 2e-16 ***
Leg_room_service4                 0.78163    0.04163  18.774  < 2e-16 ***
Leg_room_service2                 0.08722    0.04609   1.892 0.058433 .
Leg_room_service1                -0.17930    0.05962  -3.007 0.002638 **
Leg_room_service0                 2.14373    1.00781   2.127 0.033411 *
Baggage_handling3                -0.55831    0.04524 -12.340  < 2e-16 ***
Baggage_handling5                 0.70463    0.04222  16.688  < 2e-16 ***
Baggage_handling1                 0.23354    0.06865   3.402 0.000669 ***
Baggage_handling2                 0.05081    0.05974   0.851 0.395022
Checkin_service1                 -0.65105    0.04670 -13.942  < 2e-16 ***
Checkin_service3                  0.02186    0.03586   0.610 0.542146
Checkin_service5                  0.69549    0.04189  16.603  < 2e-16 ***
Checkin_service2                 -0.46469    0.04621 -10.055  < 2e-16 ***
Inflight_service4                -0.64482    0.04395 -14.672  < 2e-16 ***
Inflight_service3                -1.31311    0.05668 -23.169  < 2e-16 ***
Inflight_service1                -0.40356    0.07613  -5.301 1.15e-07 ***
Inflight_service2                -0.60540    0.06900  -8.774  < 2e-16 ***
Cleanliness1                     -0.92283    0.07327 -12.595  < 2e-16 ***
Cleanliness2                     -0.87430    0.07122 -12.276  < 2e-16 ***
Cleanliness3                     -0.42842    0.05897  -7.265 3.73e-13 ***
Cleanliness4                     -0.57471    0.05789  -9.927  < 2e-16 ***
Departure_Delay_in_Minutes        0.17262    0.04756   3.629 0.000284 ***
Arrival_Delay_in_Minutes         -0.33437    0.04754  -7.033 2.03e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

    Null deviance: 142189  on 103903  degrees of freedom
Residual deviance:  38211  on 103839  degrees of freedom
AIC: 38341

Number of Fisher Scoring iterations: 17
```

Come notiamo, anche in questo caso la feature *Flight Distance* risulta non essere significativa. Esaminando le altre colonne notiamo che, complessivamente, tutte le altre features sono relativamente significative, sia pure in misura più o meno rilevante.

Procediamo quindi a ricreare il nostro modello eliminando la colonna *Flight Distance*. Come nel modello eseguito in Python, le differenze nelle features rimanenti sono pressoché nulle, quindi, anche utilizzando questo linguaggio, decidiamo di non esibire l'output del modello.

5.3 *Denominazione dei Modelli*

Avendo creato i modelli nello stesso modo sia in Python che in R, andiamo a svolgere la classificazione in entrambi, per studiare l'impatto sulle modifiche.
Il modello creato con la feature *Flight Distance* presente verrà chiamato **Modello 1**, mentre il **Modello 2** sarà quello senza.

6 Classificazione e Predizione

6.1 *Modello 1*

Python

```
Accuracy = 0.8713427779488759
ROC Area under Curve = 0.8670132343303003
Time taken = 2.128680944442749
              precision    recall  f1-score   support

           0    0.87255   0.90249   0.88727     14573
           1    0.86967   0.83154   0.85017     11403

    accuracy                        0.87134     25976
   macro avg    0.87111   0.86701   0.86872     25976
weighted avg    0.87129   0.87134   0.87099     25976
```

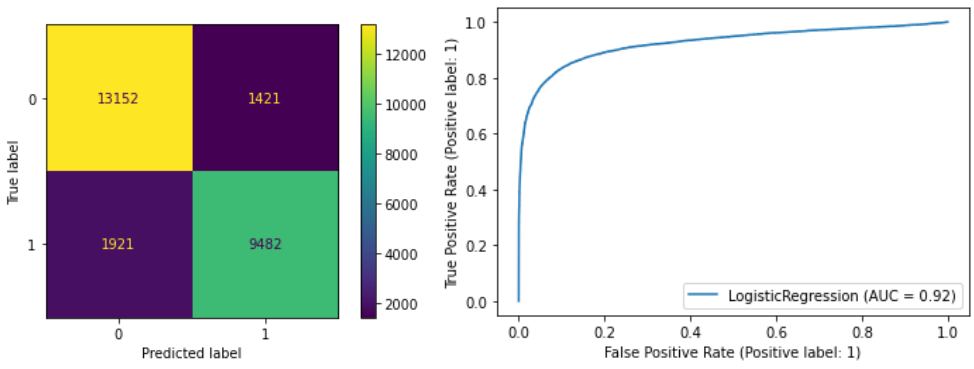

Figura 8: Matrice Confusione e ROC curve del Modello 1, Python

R

```
Confusion Matrix and Statistics

          Reference
Prediction     1     0
         1 10402   753
         0  1001 13820

               Accuracy : 0.9325
                 95% CI : (0.9294,
0.9355)
    No Information Rate : 0.561
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8626

 Mcnemar's Test P-Value : 3.686e-09

            Sensitivity : 0.9122
            Specificity : 0.9483
         Pos Pred Value : 0.9325
         Neg Pred Value : 0.9325
             Prevalence : 0.4390
         Detection Rate : 0.4004
   Detection Prevalence : 0.4294
      Balanced Accuracy : 0.9303

       'Positive' Class : 1
```

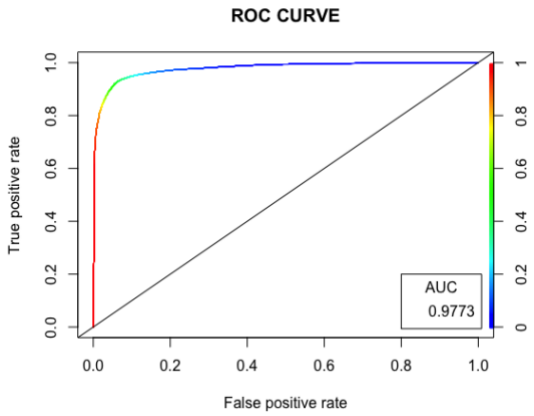

Figura 9: ROC curve Modello 1, R

Analisi Modello 1 e Comparazione

Mettiamo a confronto le matrici di confusione del Modello 1 in entrambi i linguaggi:

	Python		**R**	
satisfied	**9482**	**1921**	**10402**	**753**
neutral or dissatisfied	**1421**	**13152**	**1001**	**13820**
		Predicted		

Come possiamo notare dalla matrice di confusione, le predizioni non sono eccessivamente diverse. Notiamo però che nella predizione della variabile target "*satisfied* ", il nostro classificatore in R si comporta meglio.

Calcoliamo ora delle metriche per valutare la nostra classificazione in Python e in R.
Le metriche di cui ci avvarremo saranno: *accuracy* (percentuale di dati classificati correttamente), *sensitivity* (tasso di veri positivi), *specificity* (taso di veri negativi), *precision* (percentuale di veri positivi su tutti quelli dichiarati positivi) e *recall* (percentuale di positivi che vengono individuati).

	Python	**R**
Accuracy	**0.87**	**0.93**
Sensitivity	**0.83**	**0.93**

Specificity	**0.90**	**0.93**
Precision	**0.87**	**0.91**
Recall	**0.83**	**0.93**

Possiamo così notare una leggera differenza, forse dovuta ai linguaggi utilizzati. Infatti, avendo usato un modello di regressione ed essendo R un linguaggio prettamente a scopo statistico, questo leggero divario potrebbe essere attribuito all'implementazione nei due linguaggi per lo svolgimento della regressione.
Come possiamo notare, il **Modello 1** in entrambi i linguaggi raggiunge dei buoni punteggi in tutte le metriche, arrivando ad avere un *accuracy* di **0.87** e **0.93**.

6.2 Modello 2

Python

```
Accuracy = 0.8711887896519864
ROC Area under Curve = 0.8668759942287425
Time taken = 2.3060359954833984
              precision    recall  f1-score   support

           0    0.87252   0.90222   0.88712     14573
           1    0.86935   0.83154   0.85002     11403

    accuracy                        0.87119     25976
   macro avg    0.87093   0.86688   0.86857     25976
weighted avg    0.87113   0.87119   0.87083     25976
```

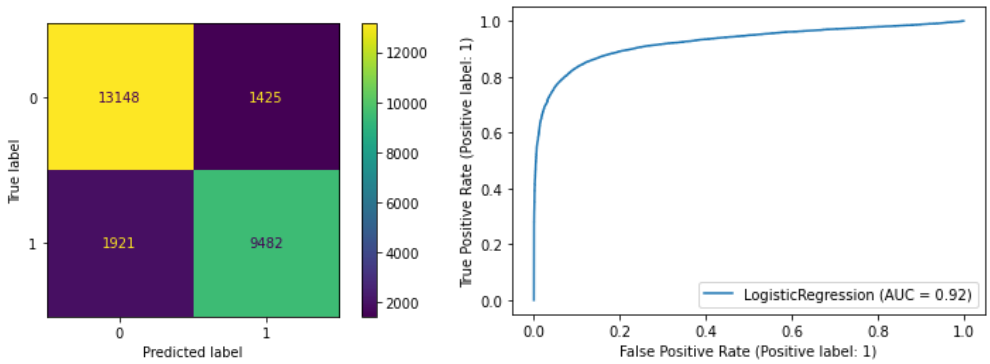

Figura 11: Matrice Confusione e ROC curve del Modello 2, Python

R

```
Confusion Matrix and Statistics

          Reference
Prediction     1     0
         1 10404   755
         0   999 13818

               Accuracy : 0.9325
                 95% CI : (0.9294,
0.9355)
    No Information Rate : 0.561
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8626

 Mcnemar's Test P-Value : 6.546e-09

            Sensitivity : 0.9124
            Specificity : 0.9482
         Pos Pred Value : 0.9323
         Neg Pred Value : 0.9326
             Prevalence : 0.4390
         Detection Rate : 0.4005
   Detection Prevalence : 0.4296
      Balanced Accuracy : 0.9303

       'Positive' Class : 1
```

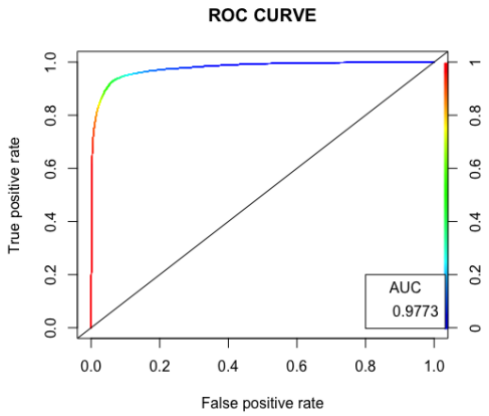

Figura 12: ROC curve Modello 2, R

Analisi Modello 2 e Comparazione

Confrontiamo le matrici di confusione in entrambi i linguaggi:

	Python		**R**	
satisfied	**9482**	**1921**	**10404**	**755**
neutral or dissatisfied	**1425**	**13148**	**999**	**13818**
		Predicted		

Le differenze con il **Modello 1** sono veramente minime. Vediamo che il punteggio di *accuracy* cambia di pochi decimali, restando praticamente di **0.87** (Python) e **0.93** (R). Come nel precedente modello, il nostro scopo è stato raggiunto.

7 Conclusioni

I nostri obbiettivi per questo progetto erano:

- Identificare i fattori più rilevanti per la soddisfazione;
- Predire la soddisfazione in base ai fattori considerati.

Il primo lo abbiamo ottenuto sia tramite l'analisi che per mezzo della creazione dei nostri modelli, scoprendo che le features: *Gender*, *Gate Location* e *Departure/Arrive time Convenient* e *Flight Distance* non erano significative.

Il secondo obbiettivo è stato acquisito tramite l'utilizzo della regressione logistica binomiale, non riscontrando problemi eccessivi.

Riguardo l'uso dei due linguaggi di programmazione, Python e R, la differenza è molta. Per quanto riguarda Python, vi è un maggiore quantità di fonti da cui attingere nel Web a differenza di R. D'altro canto, in R lo sviluppo del codice è stato molto più facile e veloce, soprattutto nella gestione dei dati.

Bibliografia

https://pandas.pydata.org/docs/user_guide/categorical.html#object-creation

https://www.guru99.com/r-factor-categorical-continuous.html

https://www.statista.com/statistics/742763/regional-carriers-average-passenger-trip-length/

https://machinelearningmastery.com/logistic-regression-for-machine-learning/

https://www.javatpoint.com/logistic-regression-in-machine-learning

Tutti consultati durante i mesi di Aprile e Maggio 2022.

Ambiente di lavoro

- Python versione 9.7.1
- R versione 4.1.2
- Visual Studio Code versione 1.67.1
- R Studio versione RStudio 2022.02.1