

Holiday Package Prediction

Kelompok : 7 Bola Naga

1. Fatimah Azzahra A
2. Fulka Ilmy Avicenna
3. Fenny Experence Rompis
4. Putri Kirey Eki Yogaswara
5. Siti Zahrotul Jannah
6. Bagus Ariobimo
7. Steve
8. Cholifatur Rohman Dimi Saputra



Stage 0

Latar Belakang

Latar Belakang

Problem

- Perusahaan "Trips & Travel.Com" ingin mengaktifkan dan membangun model bisnis yang layak untuk memperluas basis customer.
- Salah satu cara untuk memperluas basis pelanggan adalah dengan memperkenalkan penawaran paket baru.
- Dari data yang ada, hanya **18%** customer yang melakukan pembelian package pada tahun lalu.
- Hal tersebut dimungkinkan terjadi karena segmentasi yang tidak tepat sasaran, atau bahkan tidak ada **segmentasi customer** yang mengakibatkan **tingginya pengeluaran biaya marketing**.



Latar Belakang

Goal

- Effective customer targeting
- Lower marketing cost

Objective

- Membuat model machine learning yang bisa membantu **memprediksi customer** yang berpotensi membeli package
- Membuat **segmentasi customer** untuk meningkatkan akurasi dalam targeting customer

Business metric

- Conversion Rate

Stage 1

EDA

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4888 entries, 0 to 4887
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
 --- 
 0   CustomerID      4888 non-null    int64  
 1   ProdTaken       4888 non-null    int64  
 2   Age              4662 non-null    float64 
 3   TypeofContact   4863 non-null    object  
 4   CityTier         4888 non-null    int64  
 5   DurationOfPitch 4637 non-null    float64 
 6   Occupation       4888 non-null    object  
 7   Gender            4888 non-null    object  
 8   NumberOfPersonVisiting 4888 non-null  int64  
 9   NumberOfFollowups 4843 non-null    float64 
 10  ProductPitched   4888 non-null    object  
 11  PreferredPropertyStar 4862 non-null  float64 
 12  MaritalStatus     4888 non-null    object  
 13  NumberOfTrips     4748 non-null    float64 
 14  Passport          4888 non-null    int64  
 15  PitchSatisfactionScore 4888 non-null  int64  
 16  OwnCar            4888 non-null    int64  
 17  NumberOfChildrenVisiting 4822 non-null  float64 
 18  Designation        4888 non-null    object  
 19  MonthlyIncome      4655 non-null    float64 
dtypes: float64(7), int64(7), object(6)
memory usage: 763.9+ KB
```

EDA

Memiliki 4888 rows dan 20 column dengan 2 type data:

categorical (object):

- a. TypeofContact
- b. Occupation
- c. Gender
- d. ProductPitched
- e. MaritalStatus
- f. Designation

Numerical (Int64, Float64):

- a. CustomerID
- b. ProdTaken
- c. Age
- d. CityTier
- e. DurationOfPitch
- f. NumberOfPersonVisiting
- g. NumberOfFollowups
- h. PreferredPropertyStar
- i. NumberOfTrips
- j. Passport
- k. PitchSatisfactionScore
- l. OwnCar
- m. NumberOfChildrenVisiting
- n. MonthlyIncome

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4888 entries, 0 to 4887
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
 --- 
 0   CustomerID      4888 non-null    int64  
 1   ProdTaken       4888 non-null    int64  
 2   Age              4662 non-null    float64 
 3   TypeofContact   4863 non-null    object  
 4   CityTier         4888 non-null    int64  
 5   DurationOfPitch 4637 non-null    float64 
 6   Occupation       4888 non-null    object  
 7   Gender            4888 non-null    object  
 8   NumberOfPersonVisiting 4888 non-null  int64  
 9   NumberOfFollowups 4843 non-null    float64 
 10  ProductPitched   4888 non-null    object  
 11  PreferredPropertyStar 4862 non-null  float64 
 12  MaritalStatus     4888 non-null    object  
 13  NumberOfTrips     4748 non-null    float64 
 14  Passport          4888 non-null    int64  
 15  PitchSatisfactionScore 4888 non-null  int64  
 16  OwnCar             4888 non-null    int64  
 17  NumberOfChildrenVisiting 4822 non-null  float64 
 18  Designation        4888 non-null    object  
 19  MonthlyIncome      4655 non-null    float64 
dtypes: float64(7), int64(7), object(6)
memory usage: 763.9+ KB
```

EDA

pada dataset ini terdapat missing values pada beberapa column, yaitu:

- a. Age
- b. TypeofContact
- c. DurationOfPitch
- d. NumberOfFollowups
- e. PreferredPropertyStar
- f. NumberofTrips
- g. numberOfChildrenVisiting
- h. MonthlyIncome

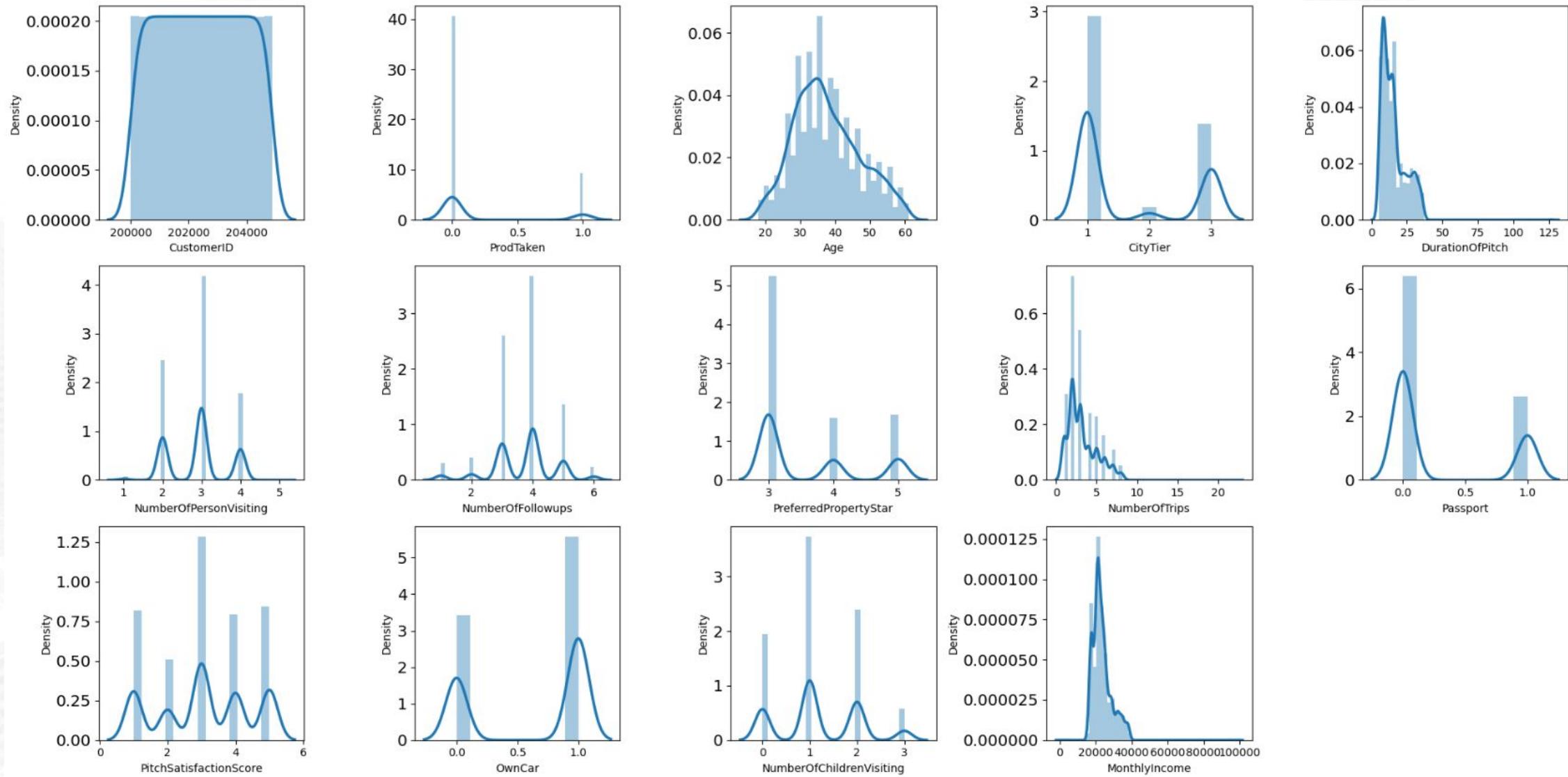
Tidak hanya itu, ketika melakukan drop pada column CustomerID terdapat **141 duplicate** pada data.

```
dupli_data = df.drop('CustomerID', axis=1).duplicated().sum()

print(f'after drop column CustomerID, this dataset has {dupli_data} duplicate data')

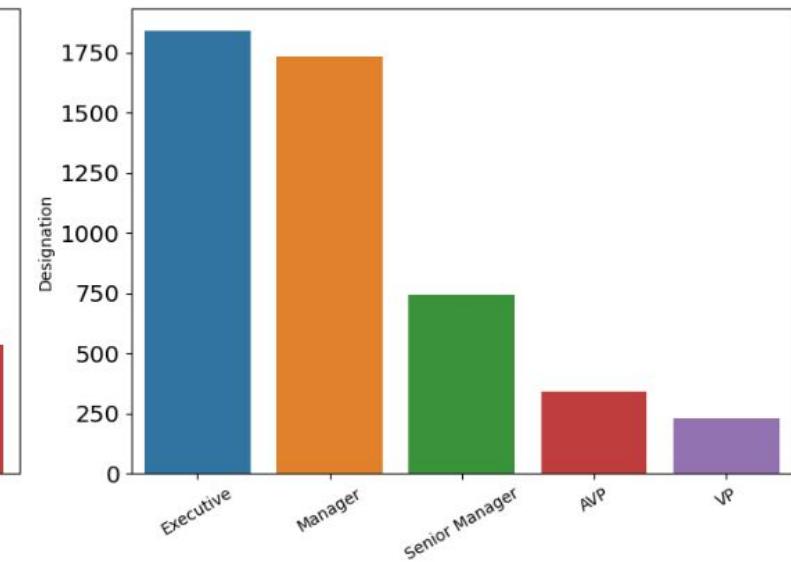
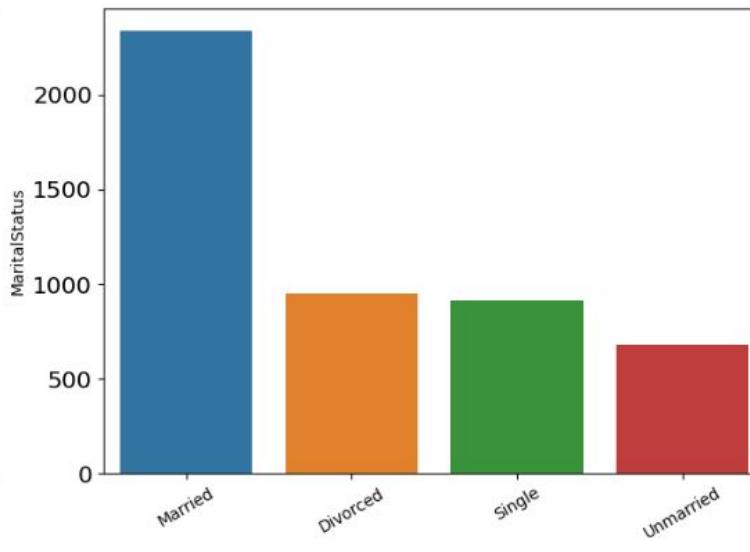
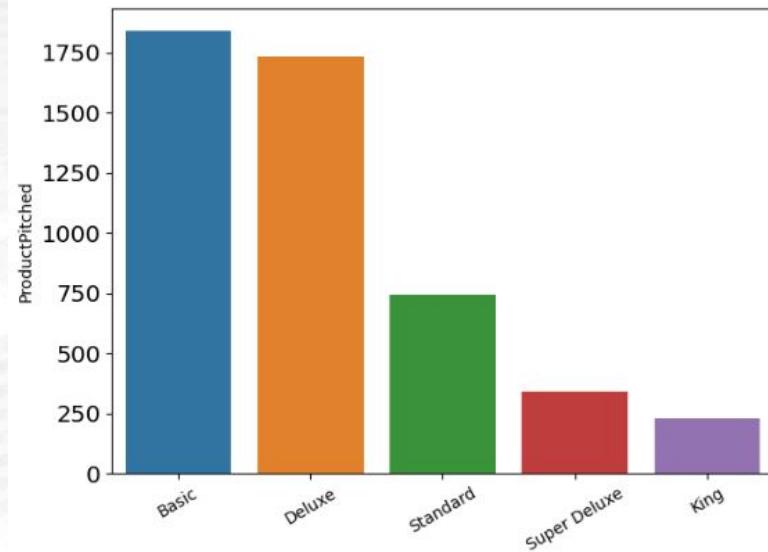
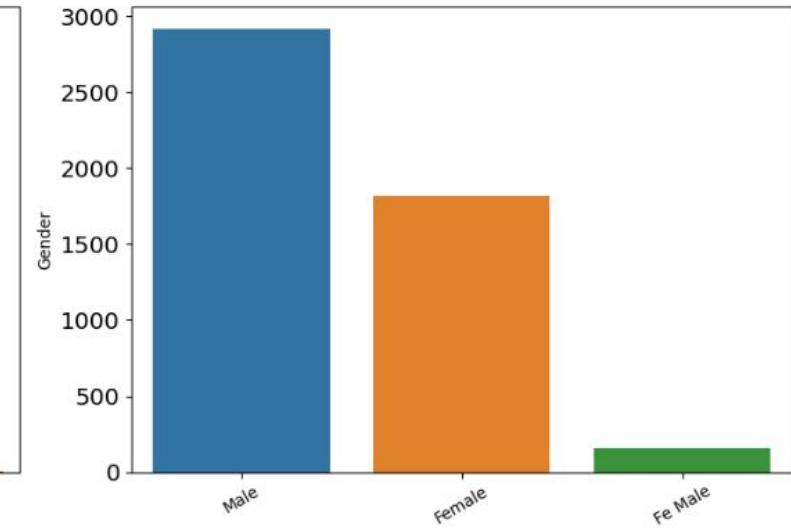
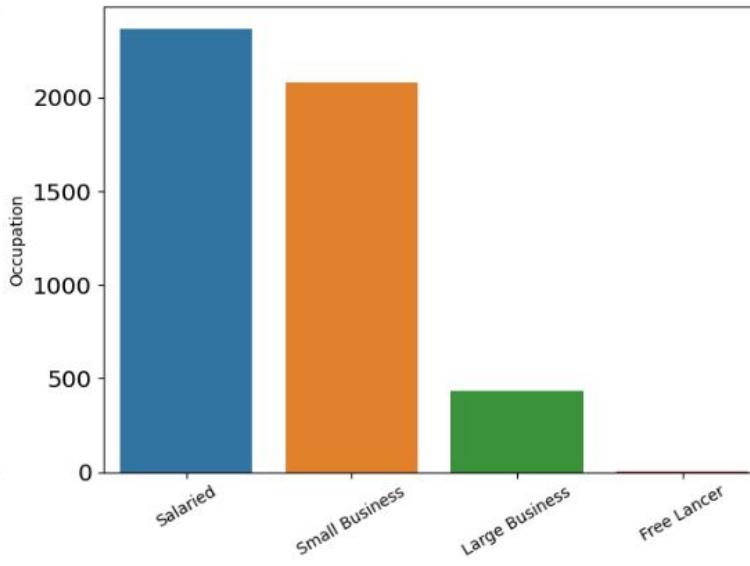
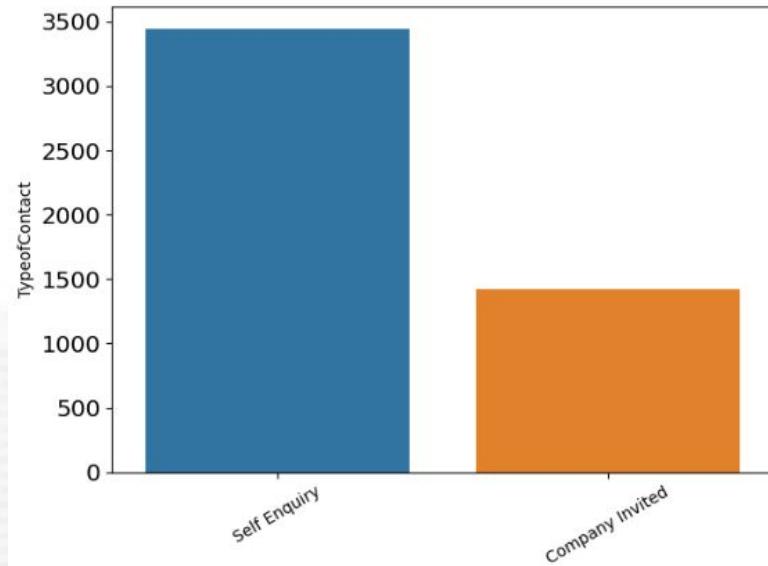
after drop column CustomerID, this dataset has 141 duplicate data
```

Numerical Columns



Pada sebaran `DurationOfPitch`, `NumberOfPitch`, dan `MonthlyIncome` mengalami **positive Skewness** (skew ke kanan) ini terjadi dikarenakan terdapat outlier pada kolom tersebut, sedangkan untuk `CustomerID` dan `Age` memiliki sebaran yang **normal**.

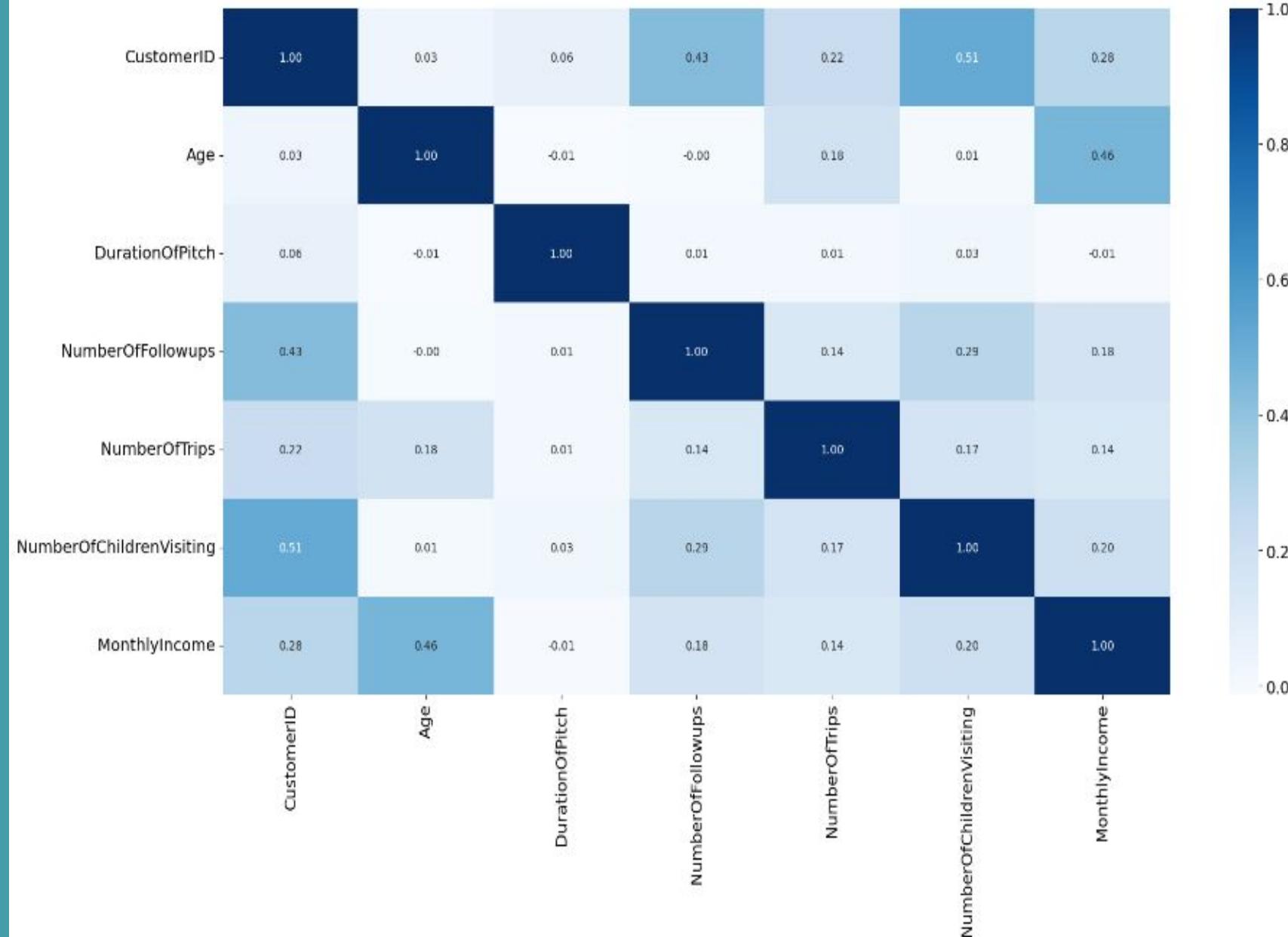
Categorical Columns



Pada chart Occupation, **Free lancer** dapat dihapus nantinya dikarenakan mempunyai **value yang sangat sedikit** jika dibandingkan dengan lainnya.

Terdapat kesalahan penulisan pada kolom Gender, "Fe male" yang menyebabkan value "Female" terbagi 2 pada MaritalStatus, dapat diasumsikan bahwa Single dan Unmarried merupakan status yang sama, maka dapat dijadi satu.

Heatmap Correlation



- Kolom 'Age' dengan 'DurationOfPitch' dan 'NumberOfFollowups' memiliki korelasi negatif lemah
- Kolom 'Age' memiliki nilai korelasi positif tinggi (mendekati 0.7), nantinya dapat digabung menjadi fitur baru
- Kolom 'MonthlyIncome' memiliki korelasi negatif lemah terhadap kolom 'DurationOfPitch'

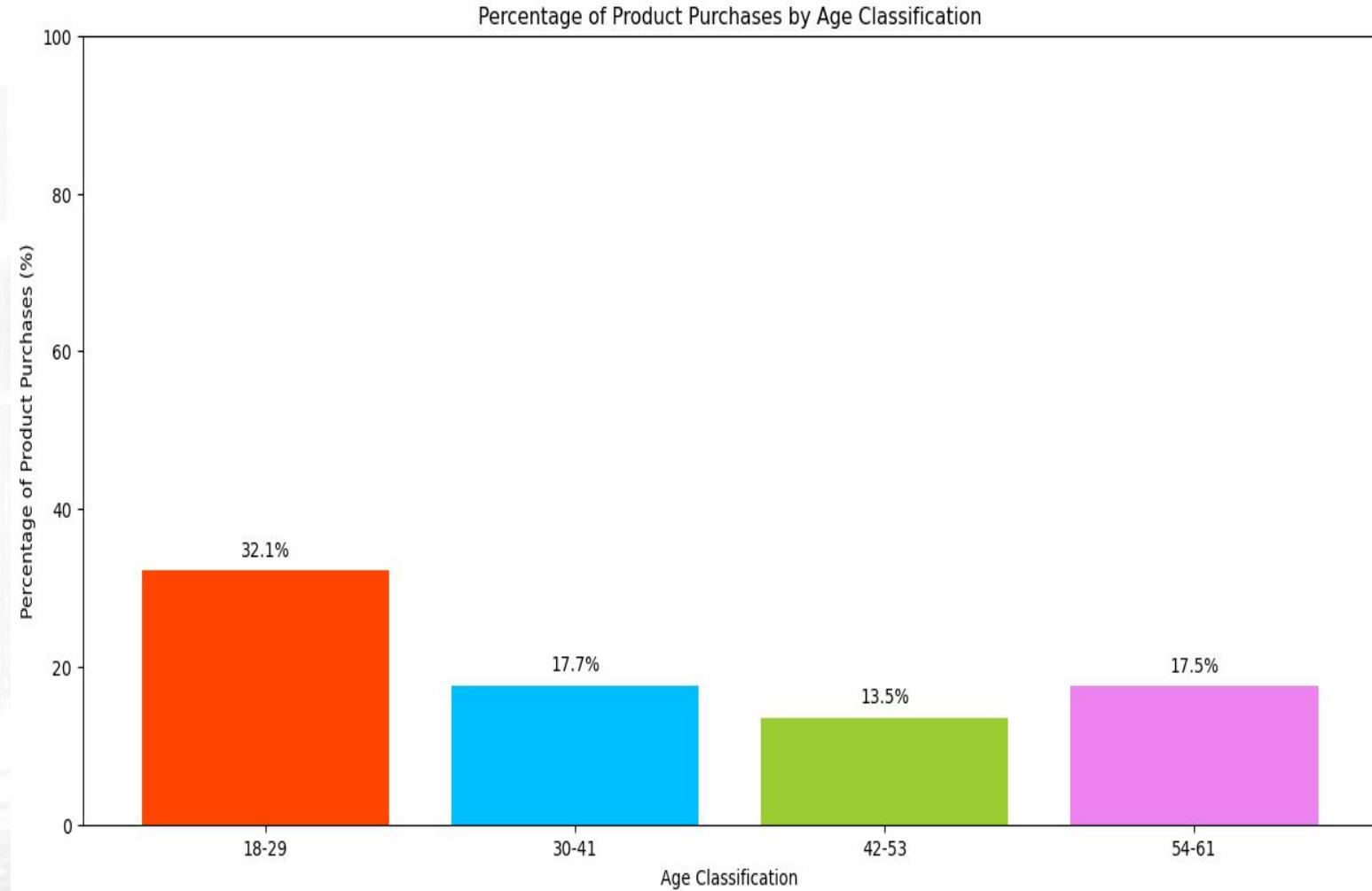
Stage 1

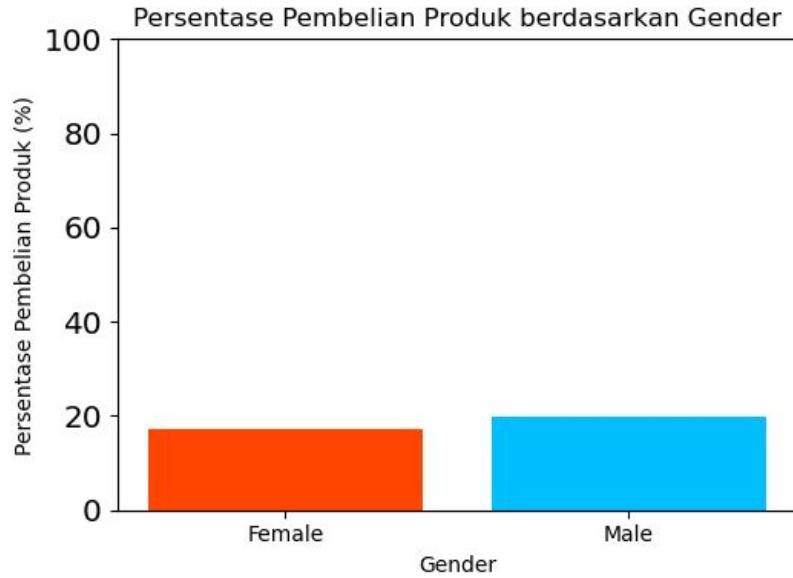
Business Insight

Business Insight

Bisa dilihat pada chart disamping, bahwa **kelompok Usia 18-29 tahun dapat diprioritaskan** karena memiliki presentase CVR paling besar (32.1%) customer membeli produk/package.

kesimpulan ini mungkin saja menjelaskan bahwa tren travelling bagi umur 18-29 cenderung memiliki gaya hidup yang aktif dan lebih banyak menghabiskan uang untuk liburan dan rekreasi.

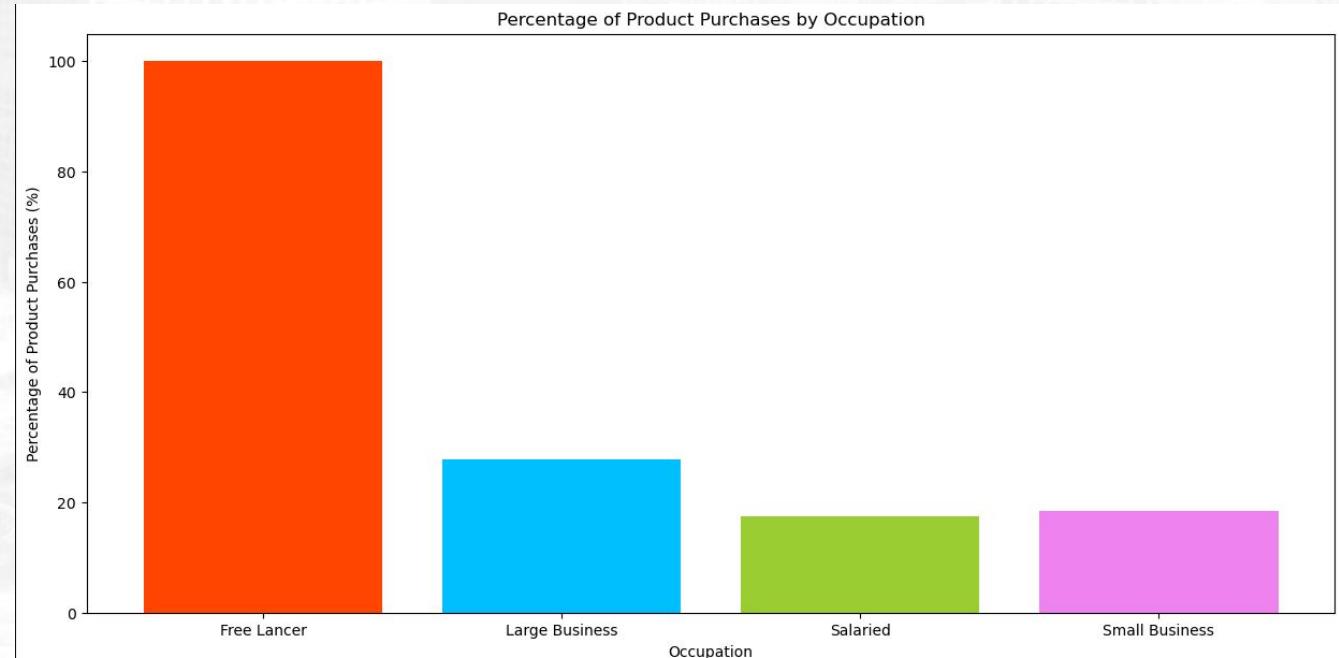




Dari visualisasi disamping dapat dilihat bahwa customer Pria lebih banyak yang melakukan pembelian dibandingkan customer Wanita. Walaupun begitu, selisih jumlah transaksi hanya berjumlah 6,6% berdasarkan jumlah data yang ada.

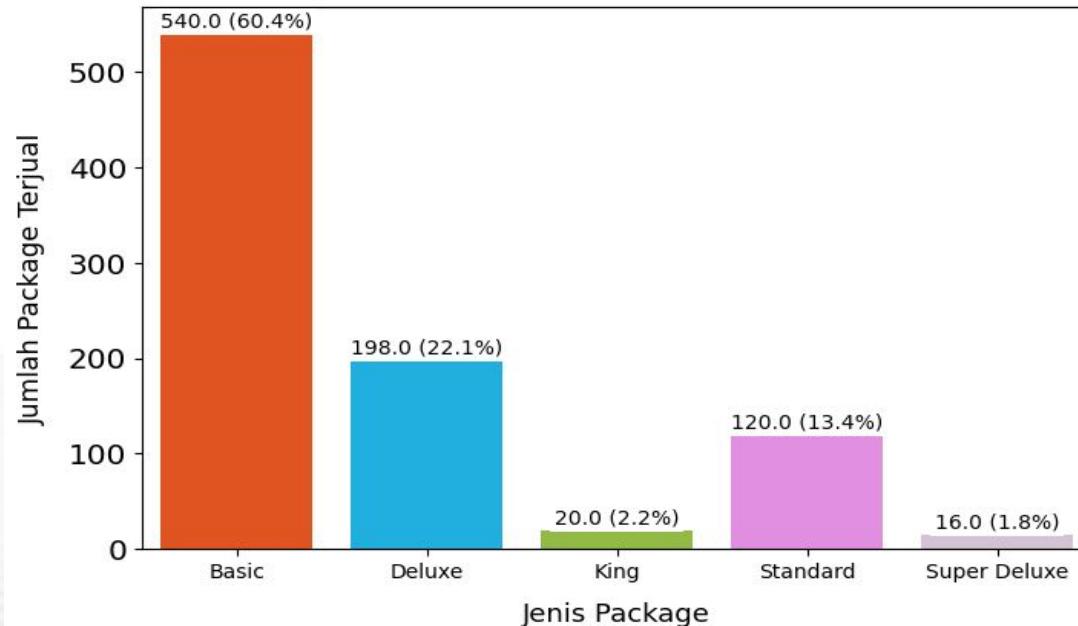
Jika melihat dari jenis pekerjaan, ternyata Free Lancer lebih unggul dalam pembelian produk sebesar 100% dibandingkan dengan Large Business, Salaried, ataupun Small Business.

Dari visualisasi yang ini dapat disimpulkan, bahwa **customer pria dan customer yang bekerja sebagai Free Lancer lebih cenderung melakukan pembelian produk package**. Disusul Large Business sebesar 17%, Small Business 11,3%, dan terakhir Salaried 10,7%.



Total Package Terjual Berdasarkan Jenisnya

Mayoritas package yang dibeli oleh customers adalah tipe Basic, Deluxe, dan Standard



Berdasarkan analisis tersebut, Dapat Disimpulkan customer dengan pembelian terbanyak sesuai jabatan adalah **Executive**, diikuti **Manager**, **Senior Manager**, dll.

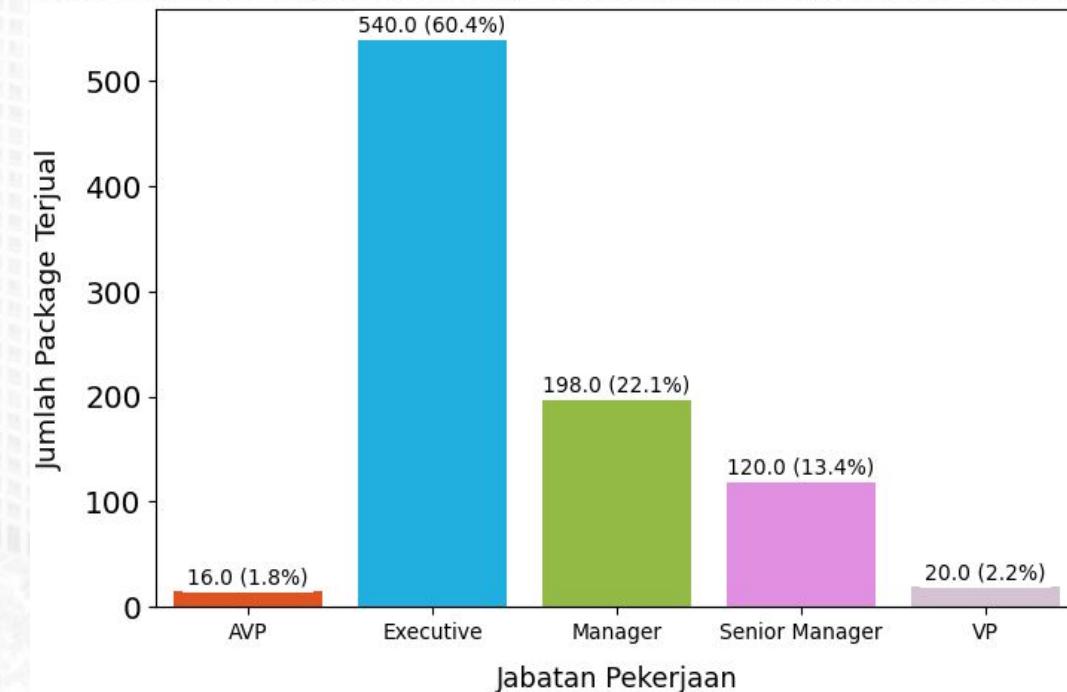
dapat dipahami jika di kalangan **Executive** cenderung memiliki kemampuan finansial yang lebih baik, kelompok ini juga menjadi faktor yang memengaruhi keputusan pembelian.

Customer lebih **tertarik** untuk purchase jenis **package Basic** jika dibandingkan package lainnya.

Untuk menarik minat customer terhadap paket lainnya, perusahaan dapat melakukan campaign atau promosi.

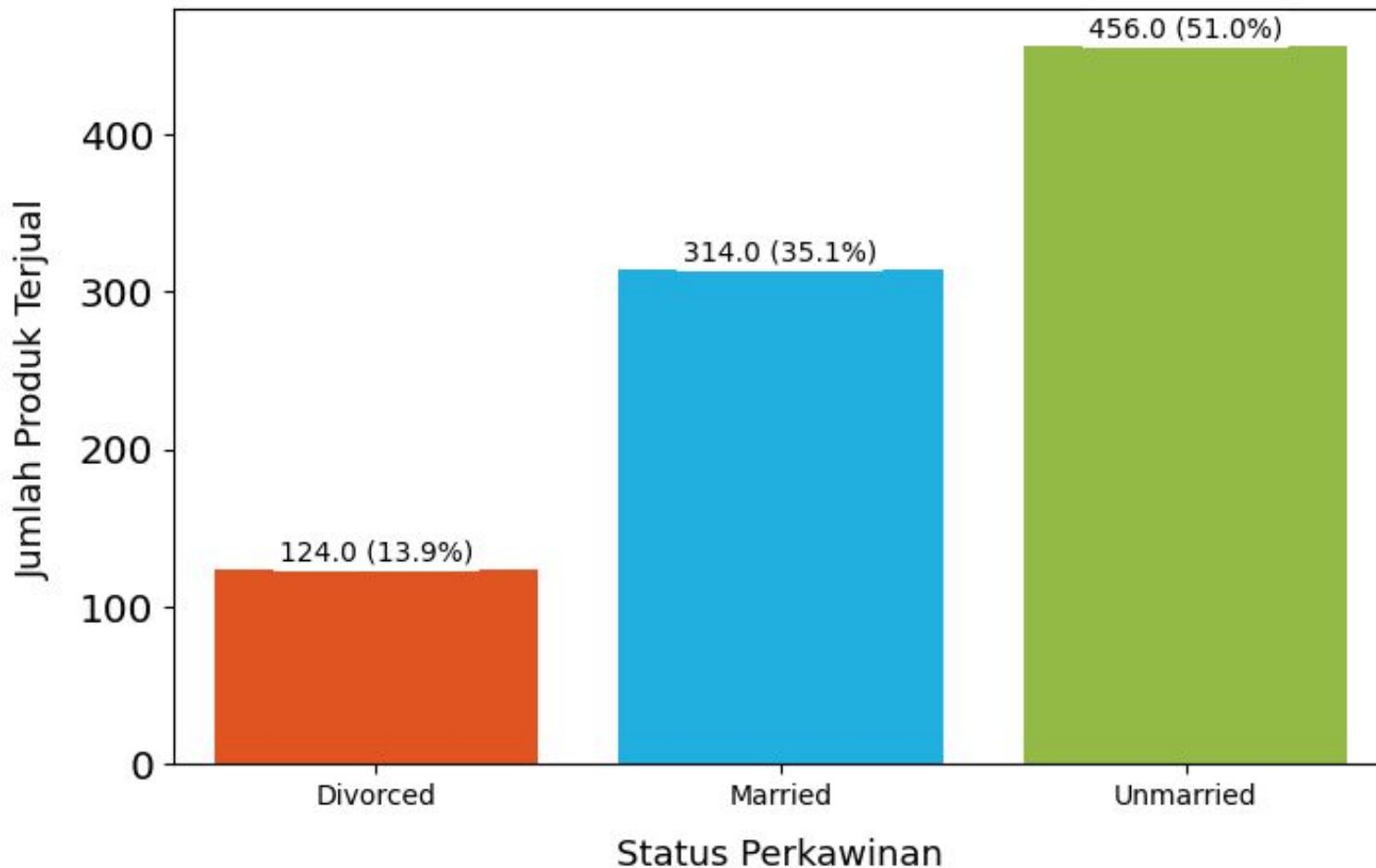
Total Package Terjual Berdasarkan Jenisnya

Mayoritas customers yang membeli package memiliki jabatan Executive, Manager dan Senior Manager



Total Penjualan Produk Berdasarkan Status Pernikahan

Mayoritas customer yang membeli package memiliki status Married dan Unmarried



Dari diagram di samping tersebut menunjukkan bahwa pelanggan yang **belum ada status menikah (unmarried)** cenderung lebih tertarik dan lebih memungkinkan untuk membeli package/produk dibandingkan mereka yang sudah **menikah (Married)** atau bahkan bagi mereka yang telah **ceraai (Divorced)**.

hal ini juga bisa diinformasikan bahwa mungkin ada beberapa faktor yang memengaruhi penjualan package dengan status perkawinan seperti kebebasan waktu dan fleksibilitas yang mereka miliki.

Stage 2

Data Cleanse

Handling Missing Values

- Age : Mean
- TypeofContact : Self Enquiry
- DurationOfPitch : Median
- NumberOfFollowups : Median
- PreferredPropertyStar : 4
- NumberOfTrips : Median
- NumberOfChildren : Modus
- MonthlyIncome : Median

```
print("Jumlah nilai kosong:")
print(df_simp.isnull().sum())
```

```
Jumlah nilai kosong:
CustomerID          0
ProdTaken           0
Age                 0
TypeofContact       0
CityTier            0
DurationOfPitch    0
Occupation          0
Gender              0
NumberOfPersonVisiting 0
NumberOfFollowups   0
ProductPitched     0
PreferredPropertyStar 0
MaritalStatus       0
NumberOfTrips       0
Passport            0
PitchSatisfactionScore 0
OwnCar              0
NumberOfChildrenVisiting 0
Designation          0
MonthlyIncome        0
dtype: int64
```

Handling Inconsistent data

- Gender : Fe Male → Female
- MaritalStatus : Single → Unmarried
- Age : dtype = Float → Int

```
# Handling Inconsistent data

## Fe Male --> Female
df_simp['Gender'] = df_simp['Gender'].apply(lambda x: 'Female' if x == 'Fe Male' else x)

## Single --> Unmarried
df_simp['MaritalStatus'] = df_simp['MaritalStatus'].apply(lambda x: 'Unmarried' if x == 'Single' else x)

## type Age --> integer (menghindari koma pada umur)
df_simp['Age'] = df_simp['Age'].astype('int')
```

Handling Duplicate Data

- Data tidak memiliki duplicate data, maka final data yang akan digunakan menjadi berukuran 4888 rows
- Kolom ‘Customer ID’ tidak diikutsertakan dalam pengolahan data

```
df_simp.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4888 entries, 0 to 4887
Data columns (total 19 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   ProdTaken        4888 non-null   int64  
 1   Age              4888 non-null   int32  
 2   TypeofContact    4888 non-null   object  
 3   CityTier          4888 non-null   int64  
 4   DurationOfPitch  4888 non-null   float64 
 5   Occupation        4888 non-null   object  
 6   Gender            4888 non-null   object  
 7   NumberOfPersonVisiting  4888 non-null   int64  
 8   NumberOffollowups 4888 non-null   float64 
 9   ProductPitched    4888 non-null   object  
 10  PreferredPropertyStar 4888 non-null   float64 
 11  MaritalStatus     4888 non-null   object  
 12  NumberOfTrips     4888 non-null   float64 
 13  Passport          4888 non-null   int64  
 14  PitchSatisfactionScore 4888 non-null   int64  
 15  OwnCar            4888 non-null   int64  
 16  NumberOfChildrenVisiting 4888 non-null   float64 
 17  Designation        4888 non-null   object  
 18  MonthlyIncome      4888 non-null   float64  
dtypes: float64(6), int32(1), int64(6), object(6)
memory usage: 706.6+ KB
```

Stage 2

Feature Engineer

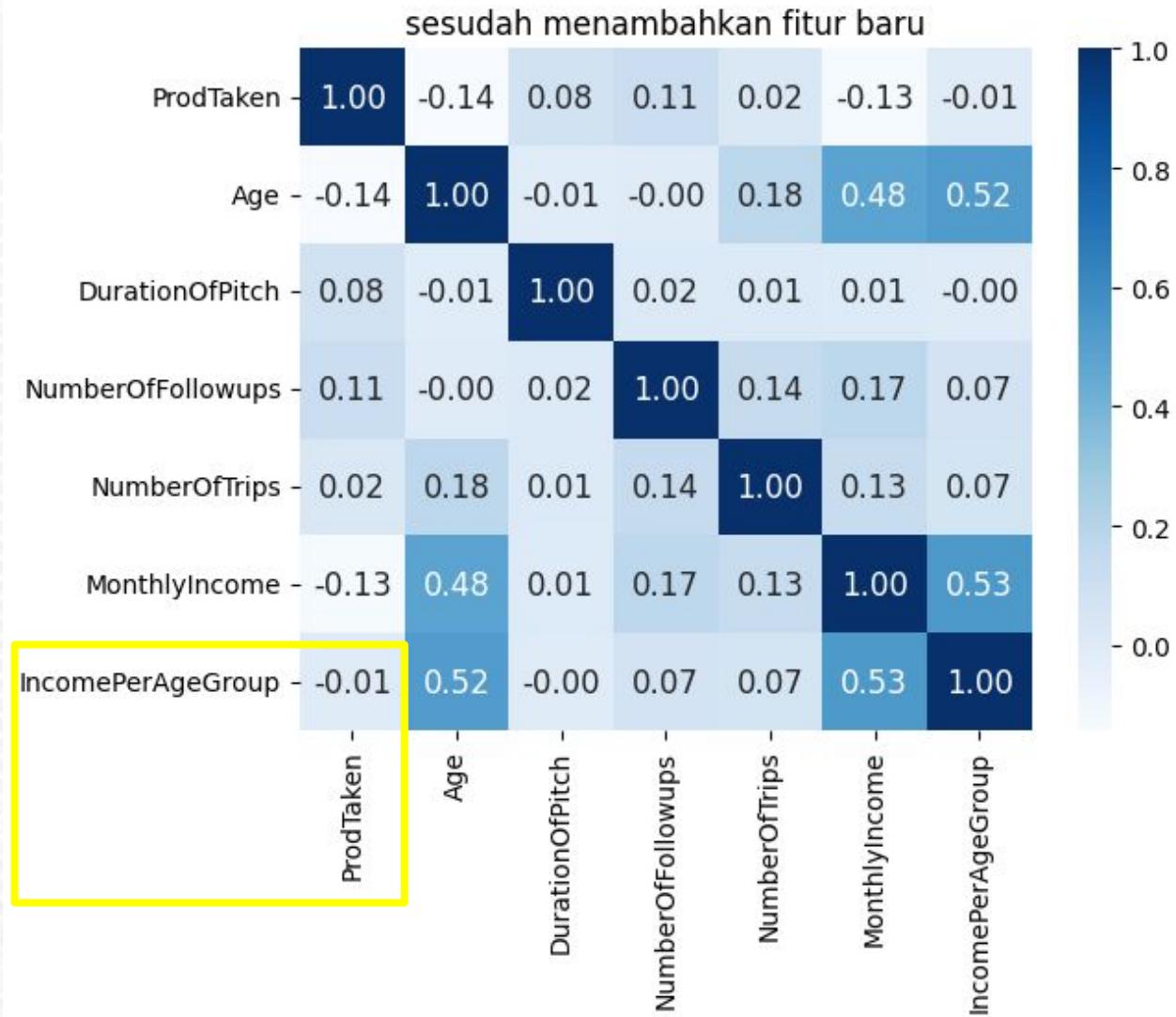
Feature Extraction (pisah)

- Nilai korelasi paling tinggi pada fitur numerical terdapat pada fitur Age - MonthlyIncome (0.48)
- tetapi memiliki nilai korelasi **lemah negatif** terhadap ProdTaken

Fitur Baru

- membuat fitur baru IncomePerAgeGroup, merupakan rata-rata income bulanan dari setiap group umur
- Fitur baru tetap memiliki **korelasi lemah negatif** terhadap ProdTaken.

memutuskan untuk **tidak menambahkan fitur baru**



Handling Outlier

```
print(f'Jumlah baris sebelum memfilter outlier: {len(df_simp)}')

filtered_entries = np.array([True] * len(df_simp))

for col in ['Age', 'DurationOfPitch', 'NumberOfTrips', 'MonthlyIncome']:
    zscore = abs(stats.zscore(df_simp[col])) # hitung absolute z-scorenya
    filtered_entries = (zscore < 3) & filtered_entries # keep yang kurang dari 3 absolute z-scorenya

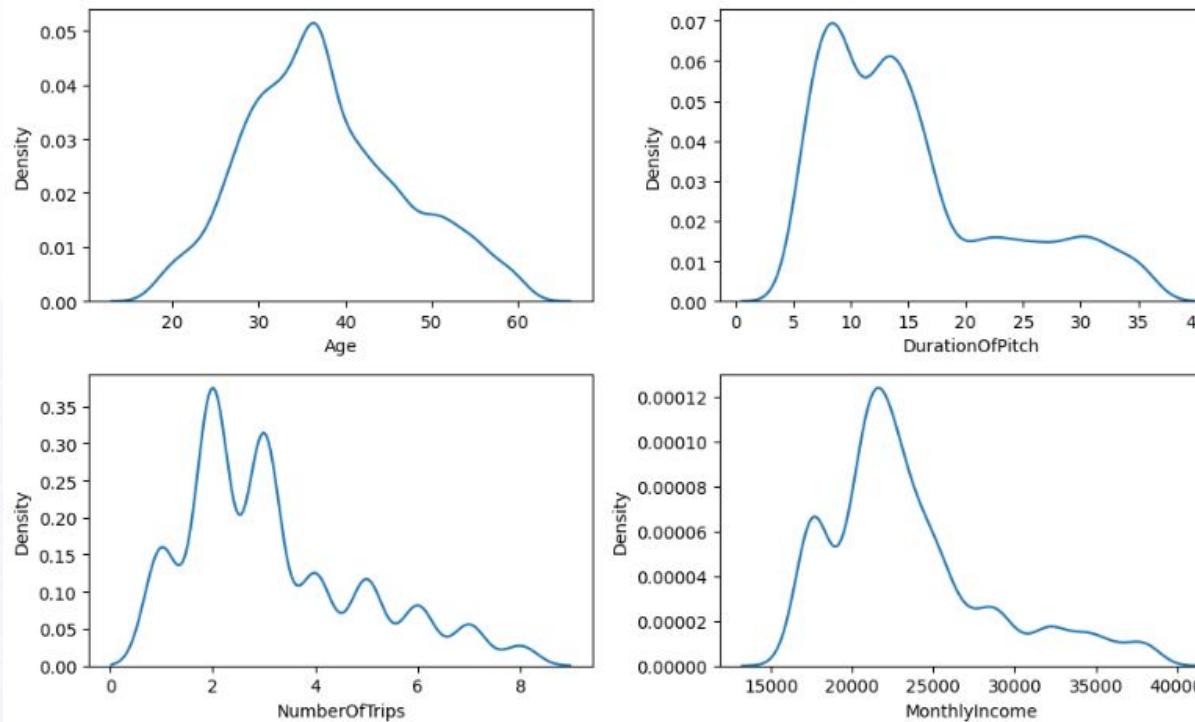
df_simp = df_simp[filtered_entries] # filter, cuma ambil yang z-scorenya dibawah 3

print(f'Jumlah baris setelah memfilter outlier: {len(df_simp)}')
```

Jumlah baris sebelum memfilter outlier: 4888

Jumlah baris setelah memfilter outlier: 4878

Feature Transformation



Metode Transformasi

- log: DOP_norm, MI_std, NumberOfTrips
- Normalisasi: Age
- Standardisasi: DOP_norm, MI_std

	Age_norm	DOP_norm	MI_std	NumberOfTrips
count	4737.000000	4.737000e+03	4.737000e+03	4737.000000
mean	0.454836	6.719931e-16	4.127958e-15	1.017576
std	0.212101	1.000106e+00	1.000106e+00	0.558580
min	0.000000	-1.988380e+00	-1.822228e+00	0.000000
25%	0.302326	-8.125703e-01	-5.890190e-01	0.693147
50%	0.441860	-7.697311e-02	-1.528550e-01	1.098612
75%	0.581395	6.821585e-01	4.857057e-01	1.386294
max	1.000000	1.960575e+00	2.592653e+00	2.079442

setelah dilakukannya transformasi:

- `Age` memiliki nilai Min: 0 dan Max: 1
- `DOP_norm` dan `MI_std` memiliki mean 0 dan standar deviasi sebesar 1
- `NumberOfTrips` memiliki nilai min: 0 dan max: 2

Feature Encoding

Encoding pada kolom:

- TypeofContact
- Gender
- ProductPitched
- Designation

ONEHOT Encoding:

- Occupation
- MaritalStatus

```
: df_simp3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4878 entries, 0 to 4887
Data columns (total 23 columns):
 #   Column           Non-Null Count Dtype  
--- 
 0   status_Divorced    4878 non-null  uint8  
 1   status_Married     4878 non-null  uint8  
 2   status_Unmarried   4878 non-null  uint8  
 3   work_Large Business 4878 non-null  uint8  
 4   work_Salaried      4878 non-null  uint8  
 5   work_Small Business 4878 non-null  uint8  
 6   ProdTaken          4878 non-null  int64  
 7   TypeofContact      4878 non-null  int64  
 8   CityTier            4878 non-null  int64  
 9   Gender              4878 non-null  int64  
 10  NumberOfPersonVisiting 4878 non-null  int64  
 11  NumberOfFollowups   4878 non-null  float64 
 12  ProductPitched     4878 non-null  int64  
 13  PreferredPropertyStar 4878 non-null  float64 
 14  NumberOfTrips       4878 non-null  float64 
 15  Passport            4878 non-null  int64  
 16  PitchSatisfactionScore 4878 non-null  int64  
 17  OwnCar              4878 non-null  int64  
 18  NumberOfChildrenVisiting 4878 non-null  float64 
 19  Designation          4878 non-null  int64  
 20  Age_norm             4878 non-null  float64 
 21  DOP_norm             4878 non-null  float64 
 22  MI_std               4878 non-null  float64 

dtypes: float64(7), int64(10), uint8(6)
memory usage: 843.6 KB
```

Feature Selection

Feature Selection dibagi menjadi 2:

- a. Metode **Chi Square** untuk **fitur numerik** pada fitur numerik terdapat 4 fitur yang terpilih yaitu: **DOP_norm, NumberOfFollowups, NumberOfTrips, MI_std**
- b. **Metode Anova** untuk **fitur categorical** terdapat 10 fitur untuk fitur kategorikal: **status_Divorced, status_Married, status_Unmarried, work_Large Business, CityTier, ProductPitched, PreferredPropertyStar, Passport, PitchSatisfactionScore, Designation'**

```
df_simp3_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4878 entries, 0 to 4877
Data columns (total 14 columns):
 #   Column           Non-Null Count
 ---  -- 
 0   DOP_norm        4878 non-null
 1   NumberOfFollowups 4878 non-null
 2   MI_std          4878 non-null
 3   NumberOfPersonVisiting 4878 non-null
 4   status_Divorced 4878 non-null
 5   status_Married   4878 non-null
 6   status_Unmarried 4878 non-null
 7   work_Large Business 4878 non-null
 8   TypeofContact    4878 non-null
 9   CityTier         4878 non-null
 10  ProductPitched   4878 non-null
 11  PreferredPropertyStar 4878 non-null
 12  Passport         4878 non-null
 13  Designation      4878 non-null
dtypes: float64(14)
memory usage: 533.7 KB
```

Class Imbalance

Sebelum melakukan class imbalance dilakukannya split pada dataset

```
Jumlah train dataset terdiri dari 3902 baris
```

```
Jumlah class 0 : 3174
```

```
Jumlah class 1 : 728
```

```
Jumlah test dataset terdiri dari 976 baris
```

```
Jumlah class 0 : 786
```

```
Jumlah class 1 : 190
```

dikarenakan terdapat ketimpangan pada class 1 maka dilakukan-nya SMOTE

```
Jumlah train dataset terdiri dari 4761 baris
```

```
Jumlah class 0 : 3174
```

```
Jumlah class 1 : 1587
```

train data setelah dilakukannya SMOTE

Feature Tambahan (Susulan)

Feature tambahan yang mungkin akan membantu membuat performansi model antara lain :

1. Frequent Package
2. Cicilan (Yes/No)
3. Monthly Spending
4. Work (WFO/WFH/Hybrid)

Stage 3

Modeling

modelling

Tim kami melakukan 4 modelling yang berbeda dan akan memilih performa model yang terbaik:

1. Decision Tree Classifier

Training Accuracy : 1.0

Testing Accuracy : 0.9016393442622951

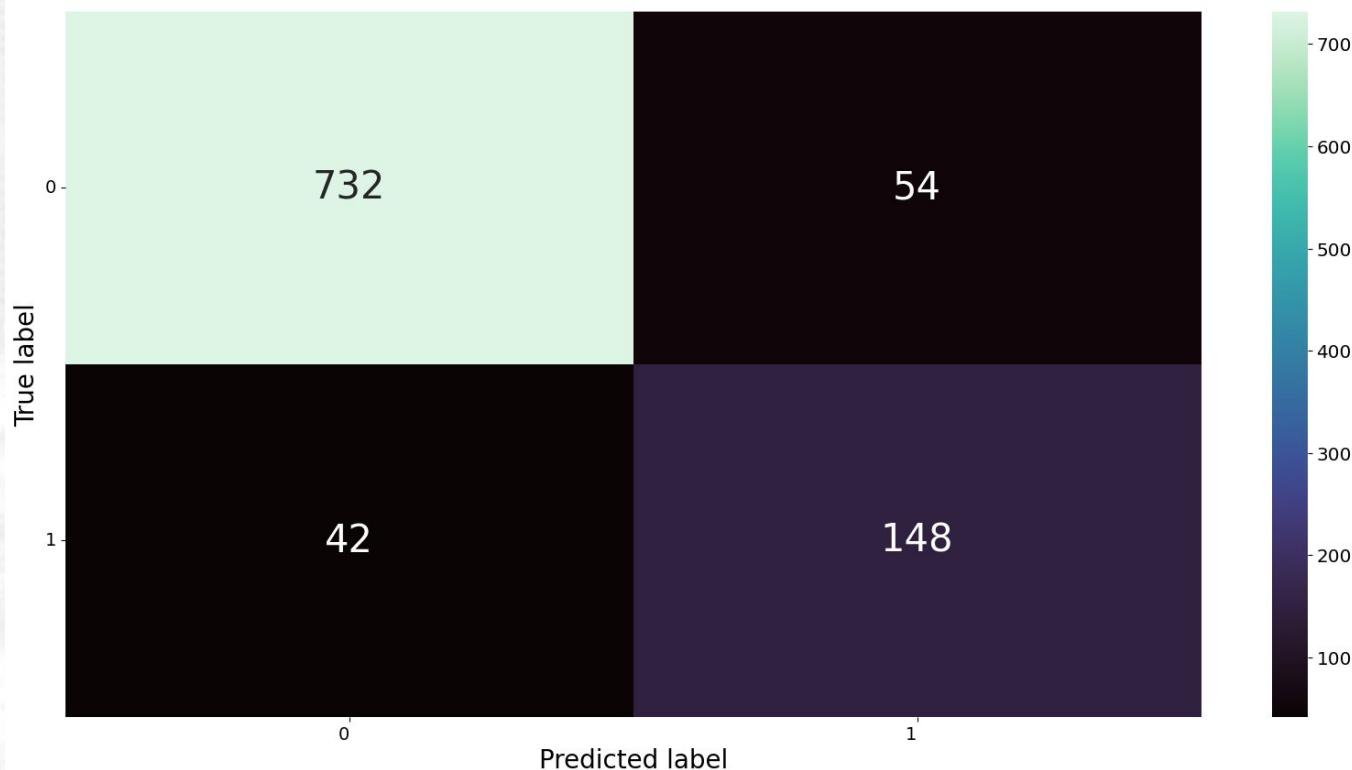
Confusion matrix

```
[[732 54]
 [ 42 148]]
```

Classification report:

	precision	recall	f1-score	support
0	0.95	0.93	0.94	786
1	0.73	0.78	0.76	190
accuracy			0.90	976
macro avg	0.84	0.86	0.85	976
weighted avg	0.90	0.90	0.90	976

Confusion Matrix for Testing Model
(Decision Tree Classifier)



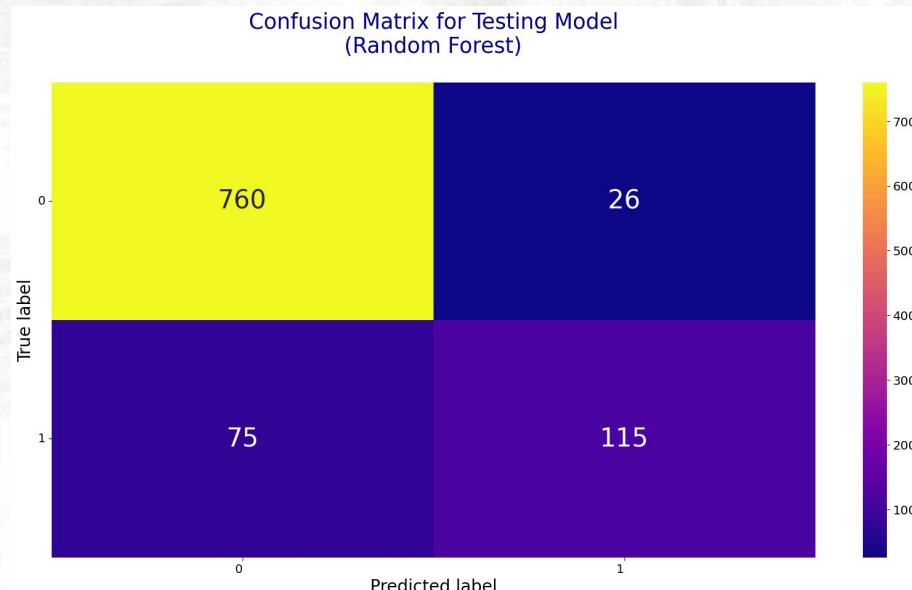
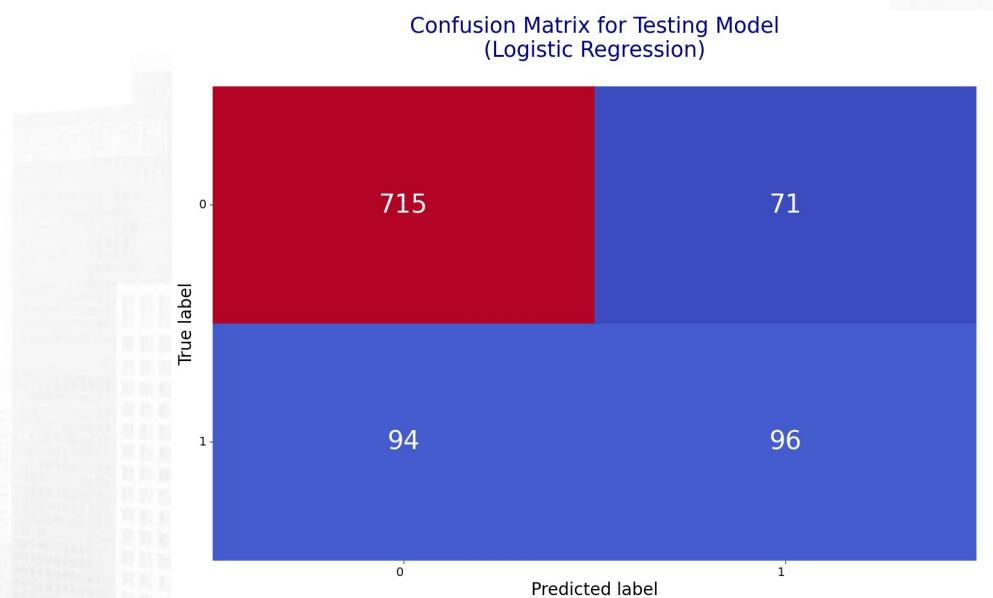
2. Linear Regression

Training Accuracy : 0.7781978575929427
 Testing Accuracy : 0.8309426229508197

Confusion matrix
 [[715 71]
 [94 96]]

Classification report

	precision	recall	f1-score	support
0	0.88	0.91	0.90	786
1	0.57	0.51	0.54	190
accuracy			0.83	976
macro avg	0.73	0.71	0.72	976
weighted avg	0.82	0.83	0.83	976



Training Accuracy : 1.0
 Testing Accuracy : 0.8965163934426229

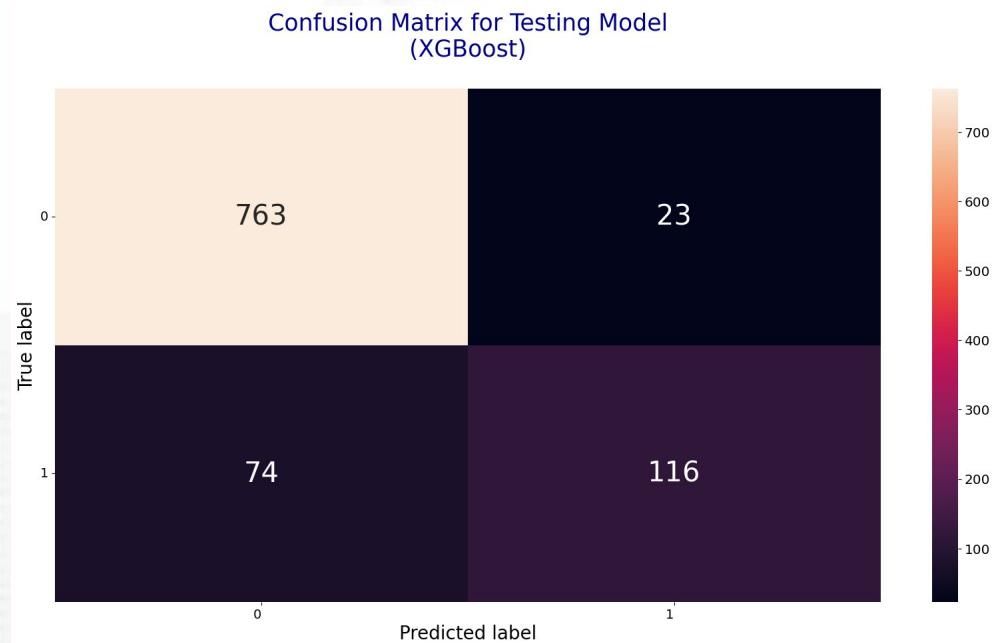
Confusion matrix
 [[760 26]
 [75 115]]

Classification report

	precision	recall	f1-score	support
0	0.91	0.97	0.94	786
1	0.82	0.61	0.69	190
accuracy			0.90	976
macro avg	0.86	0.79	0.82	976
weighted avg	0.89	0.90	0.89	976

3. Random Forest

4. XGBoost



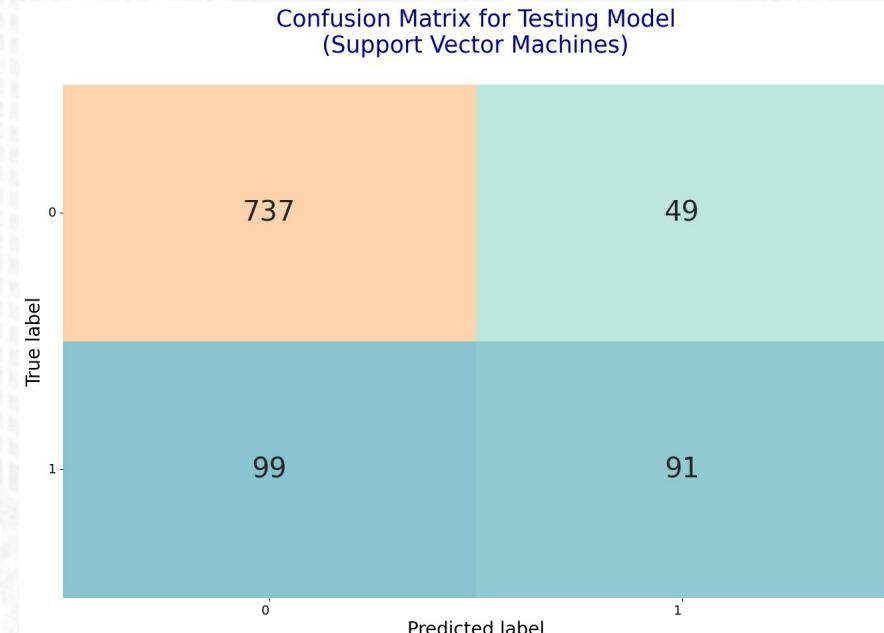
Training Accuracy : 0.9880277252678009
Testing Accuracy : 0.9006147540983607

Confusion matrix
[[763 23]
 [74 116]]

Classification report

	precision	recall	f1-score	support
0	0.91	0.97	0.94	786
1	0.83	0.61	0.71	190
accuracy	0.87	0.79	0.82	976
macro avg	0.90	0.90	0.89	976
weighted avg	0.90	0.90	0.89	976

5. Support Vector Classification (SVC)



Modeling

- Model yang digunakan yaitu **DecisionTree**
- » Training Accuracy : 1.0
Testing Accuracy : 0.8995901639344263

Confusion matrix:

```
[[732  54]
 [ 44 146]]
```

Classification report:

	precision	recall	f1-score	support
0	0.94	0.93	0.94	786
1	0.73	0.77	0.75	190
accuracy			0.90	976
macro avg	0.84	0.85	0.84	976
weighted avg	0.90	0.90	0.90	976

Terdapat 732 prediksi benar untuk kelas 0 (True Positive).

Terdapat 54 prediksi salah untuk kelas 0 (False Positive).

Terdapat 44 prediksi salah untuk kelas 1 (False Negative).

Terdapat 146 prediksi benar untuk kelas 1 (True Negative).

Kesimpulan

- Model memiliki tingkat akurasi yang relatif baik dengan Testing Accuracy sebesar 0.8995.
- Model cenderung lebih baik dalam memprediksi kelas 0 (nilai 0) dengan precision, recall, dan f1-score yang lebih tinggi dibandingkan dengan kelas 1 (nilai 1).
- Meskipun tingkat akurasi secara keseluruhan baik, terdapat kesalahan prediksi dalam memprediksi kelas 1 dengan jumlah False Negative yang cukup signifikan (96).
- Model perlu diperbaiki untuk meningkatkan performa dalam memprediksi kelas

Hyperparameter Tuning

Setelah menentukan model, lalu saatnya melakukan Hyperparameter tuning dengan menggunakan metode **GridSearchCV**. kita telah menentukan parameter sesuai kriteria grid pada umumnya, di model **DecisionTree**.

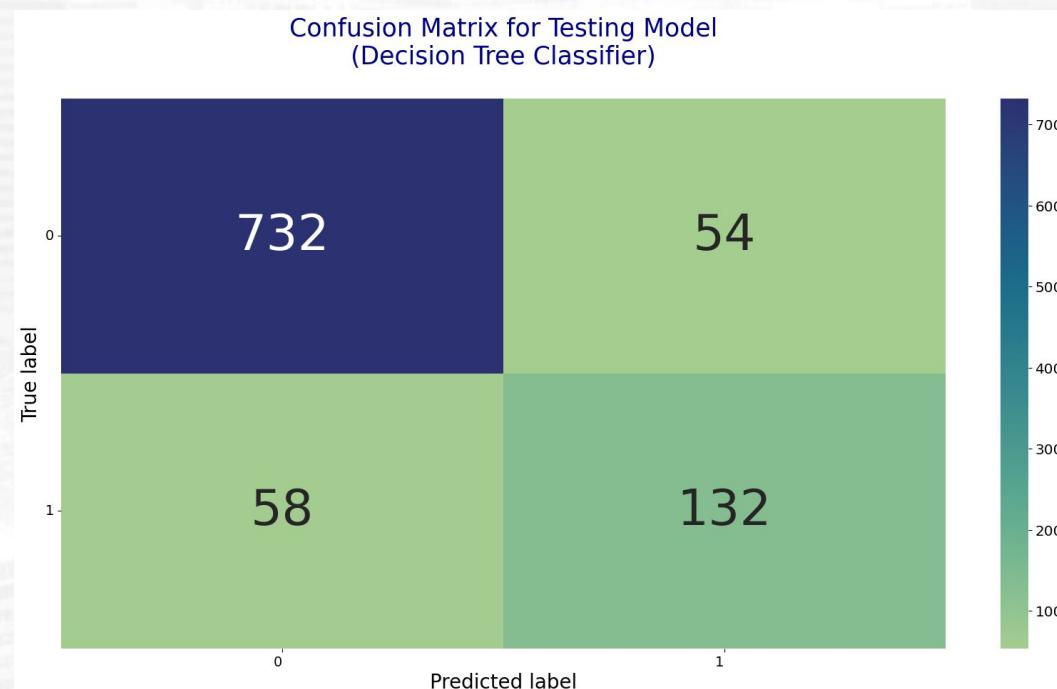
```
# mengatur parameter grid
param_grid = {
    'criterion': ['gini', 'entropy'],
    'max_depth': [1, 5, 10],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
```

hasil output **Hyperparameter**:

```
Best parameters: {'criterion': 'gini', 'max_depth': 5, 'min_samples_leaf': 1, 'min_samples_split': 2}
Best score: 0.9666666666666668
```

Hyperparameter Tuning

visualisasi model setelah Hyperparameter Tuning:



```
Training Accuracy: 0.9848771266540642
Testing Accuracy: 0.8852459016393442
Confusion matrix:
[[732 54]
 [ 58 132]]
Classification report:
precision    recall   f1-score   support
          0       0.93      0.93      0.93      786
          1       0.71      0.69      0.70      190
accuracy                           0.89      976
macro avg       0.82      0.81      0.82      976
weighted avg    0.88      0.89      0.88      976
```

Kesimpulan

Dalam kedua kasus, model memperoleh jumlah prediksi yang benar (True Positive dan True Negative) yang signifikan. Namun, setelah penggunaan hyperparameter yang dioptimalkan, terjadi peningkatan False Negative dan penurunan False Positive.

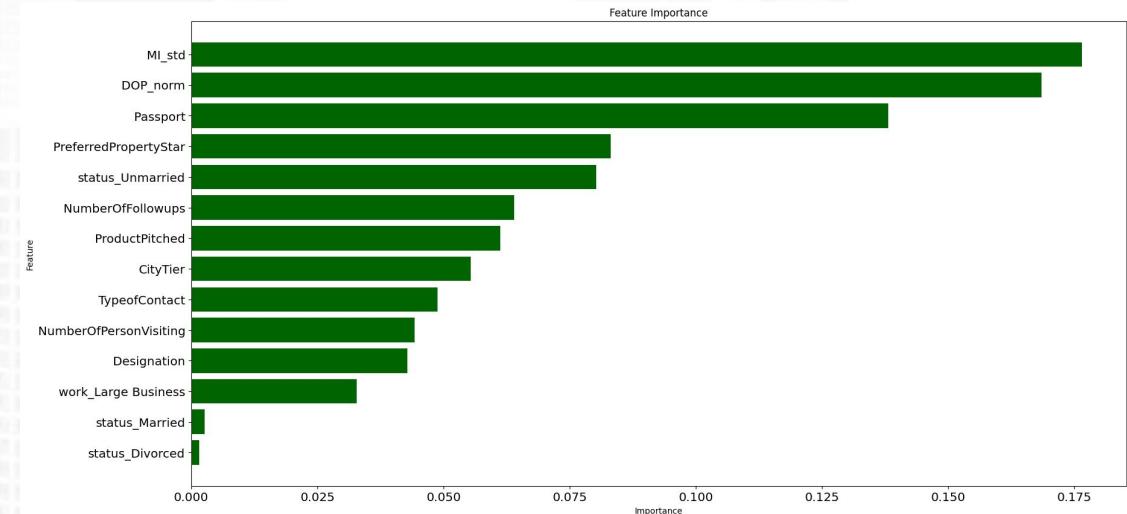
Hal ini menunjukkan bahwa setelah menggunakan hyperparameter yang dioptimalkan, model cenderung memiliki kinerja yang lebih baik dalam mengklasifikasikan kelas positif (nilai 1) dengan lebih akurat, tetapi mengorbankan beberapa klasifikasi kelas negatif (nilai 0) yang sebelumnya benar.

Stage 3

Feature Importance

Feature Importance

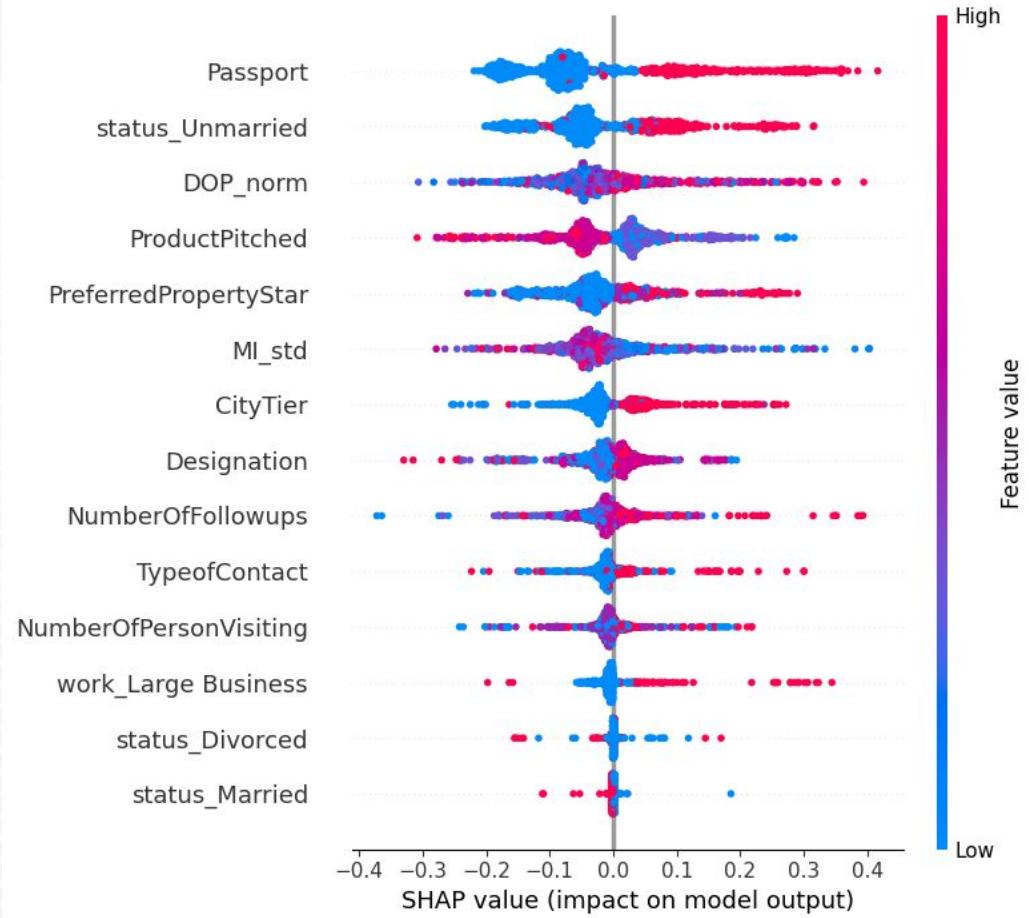
- Berdasarkan graph disamping, 3 feature teratas yang memiliki efek signifikan dalam memprediksi output (ProdTaken) antara lain adalah Monthly Income, Passport, dan Duration of Pitch
- Jumlah penghasilan berpengaruh terhadap keputusan customer untuk membeli package. Calon customer yang sudah memiliki paspor atau yang duration of pitch nya lebih lama memiliki probabilitasnya yg lebih tinggi untuk membeli package.
- Perusahaan perlu memperhatikan fitur2 penting seperti jumlah penghasilan, kepemilikan paspor, dan duration of pitch pada saat melakukan kegiatan marketing agar conversion rate dapat meningkat dan cost marketing menjadi lebih efisien.



	Feature	Importance
2	MI_std	0.175378
0	DOP_norm	0.165450
12	Passport	0.134719
11	PreferredPropertyStar	0.083867
6	status_Unmarried	0.076996
1	NumberOfFollowups	0.065324
10	ProductPitched	0.058107
9	CityTier	0.054101
3	NumberOfPersonVisiting	0.050977
8	TypeofContact	0.048505
13	Designation	0.046262
7	work_Large Business	0.032730
5	status_Married	0.004688
4	status_Divorced	0.002896

SHAP Values

Fitur Passport memiliki kontribusi yang tinggi dalam membuat prediksi. Hal ini menunjukkan bahwa status customer yang memiliki paspor dapat mempengaruhi hasil dari pitch penjualan. Mungkin ada faktor-faktor terkait perjalanan atau mobilitas yang mempengaruhi minat atau kemampuan seseorang untuk membeli produk.



Stage 4

Business Impact &

Recomendation

Type Customer Recommendation

- a. customer dengan pendapatan bulanan low middle dengan range \$11.000 - \$20.000
 - b. melakukan Durasi Penawaran diatas 22 menit
 - c. memilih property star dengan bintang 5.0
 - d. mempunyai passport
 - e. belum menikah

Marketing Costs

```
[38] df_finalized = X_test.copy()
    df_finalized.reset_index(drop=True, inplace=True)
    df_finalized['ProdTaken'] = y_test.reset_index(drop=True)
    df_finalized['ProdTaken_Pred'] = y_pred

[51] CostperCall = 4.15
    cust_before = df_finalized.shape[0]
    CostBefore = cust_before*CostperCall
    cust_after = df_finalized[df_finalized['ProdTaken_Pred']==1].shape[0]
    CostAfter = cust_after * CostperCall
    costDiff = CostBefore - CostAfter
    decreasePercent = (costDiff/CostBefore)*100

[52] print(f'''Without model we aimlessly contact {cust_before} customer, makes the Cost Before Using Model: ${CostBefore:.1f}''')
    print(f'''After Using model we just need to contact {cust_after} customer, which makes the Cost After Using Model: ${CostAfter:.1f}''')
    print(f'''Differences cost before and after using model : ${costDiff:.1f}, which decrease by {decreasePercent:.2f}%''')

Without model we aimlessly contact 976 customer, makes the Cost Before Using Model: $4050.4
After Using model we just need to contact 192 customer, which makes the Cost After Using Model: $796.8
Differences cost before and after using model : $3253.6, which decrease by 80.33%
```

Conversion Rate

Conversion Rate

```
[41] Before = df_finalized.shape[0]
    cv_before = df_finalized[df_finalized['ProdTaken'] == 1].shape[0]
    percentage_bfr = (cv_before/Before)*100

    After = df_finalized[df_finalized['ProdTaken_Pred']==1].shape[0]
    cv_after = df_finalized[(df_finalized['ProdTaken']==1)&(df_finalized['ProdTaken_Pred']==1)].shape[0]
    percentage_aftr = (cv_after/After)*100

    diff_percent = percentage_aftr - percentage_bfr
```

```
[43] print
    print(f'''conversion before using model : {cv_before}, with a conversion Rate of {percentage_bfr:.2f}%''')
    print(f'''conversion after using model : {cv_after}, with a conversion Rate of {percentage_aftr:.2f}%''')
    print(f'''Therefore the conversion rate has increased by {diff_percent:.2f}%'''')
```

```
conversion before using model : 190, with a conversion Rate of 19.47%
conversion after using model : 134, with a conversion Rate of 69.79%
Therefore the conversion rate has increased by 50.32%
```

Application of Conversion Rate on Gross Sales

```
[65] customer = 1000
     basic_pack = 100 #Dalam Dollar

     cust_convers_before = np.floor((percentage_bfr/100) * customer)
     gross_sales_bfr = cust_convers_before*basic_pack

     cust_convers_after = np.floor((percentage_aftr/100) * customer)
     gross_sales_aftr = cust_convers_after*basic_pack

     diff = gross_sales_aftr - gross_sales_bfr

     gross_sales_percent = (diff/gross_sales_bfr)*100

[89] print('Application of Conversion Rate on Gross Sales')
     print(f'''we assume that we got the same budget for telemarketing for 1000 customer and the same package price (${basic_pack})''')
     print('-----\n')
     print(f'''Without Modelling\n-----\nTotal Customer Conversion: {cust_convers_before}\nGross Sales: ${gross_sales_bfr}\n''')
     print(f'''With Modelling\n-----\nTotal Customer Conversion: {cust_convers_after}\nGross Sales: ${gross_sales_aftr}\n''')
     print(f'''The differences gross sales before and after modelling is ${diff} which has increase up to {gross_sales_percent:.0f}%''')

Application of Conversion Rate on Gross Sales
we assume that we got the same budget for telemarketing for 1000 customer and the same package price ($100)

Without Modelling
-----
Total Customer Conversion: 194.0
Gross Sales: $19400.0

With Modelling
-----
Total Customer Conversion: 697.0
Gross Sales: $69700.0

The differences gross sales before and after modelling is $50300.0 which has increase up to 259%
```