

Data Traveller

Dokumen
Laporan Final
Project



Anggota Tim

Tony Hermawan Widjanarko

Esraminar Siregar

Rayhan Prawira Daksa

Rianita

Farhan Rizki

Ryan Anugrah

Latar Belakang Masalah

PT. Chikitrans adalah sebuah perusahaan travel yang menjual 5 paket travel yaitu Basic, Standard, Deluxe, Super Deluxe & King. Sebagai tim data internal PT. Chikitrans, kami bertanggung jawab untuk memberikan rekomendasi bisnis berdasarkan data yang tersedia guna meningkatkan performa penjualan dari perusahaan tersebut.

Berdasarkan data penjualan yang sudah dicatat selama ini, conversion rate yang didapat hanya sebesar 18%. Biaya marketing yang dikeluarkan untuk melakukan follow up kepada calon customer juga tinggi yaitu Rp. 10,000 per follow up. Dari latar belakang tersebut, tim data internal PT. Chikitrans berusaha mencari solusi untuk dua masalah yaitu :

- Conversion rate yang rendah (18%)
- Biaya marketing yang tinggi

Latar Belakang Masalah

Goal :

Goal yang akan dicapai oleh tim data internal PT. Chikitrans adalah untuk meningkatkan conversion rate yang secara tidak langsung akan mengurangi biaya marketing

Objective :

1. Membuat model untuk memprediksi customer mana yang potensial untuk membeli paket travel
2. Memberikan rekomendasi bisnis untuk meningkatkan conversion rate dan mengoptimalkan biaya marketing

Business Metrics :

- Conversion rate (primary)
- Customer acquisition cost (secondary)

Dataset

Dataset terdiri dari 4,888 baris data yang terdiri dari profil customer dan detail transaksi


Profil Customer :

- CustomerID
- Age
- CityTier
- Occupation
- Gender
- MaritalStatus
- Passport
- OwnCar
- MonthlyIncome
- Designation

Detail Transaksi :

- TypeofContact
- DurationOfPitch
- NumberOfPersonVisiting
- NumberOfFollowups
- ProductPitched
- PreferredPropertyStar
- NumberOfTrips
- PitchSatisfactionScore
- NumberOfChildrenVisiting
- ProdTaken

Descriptive Statistics



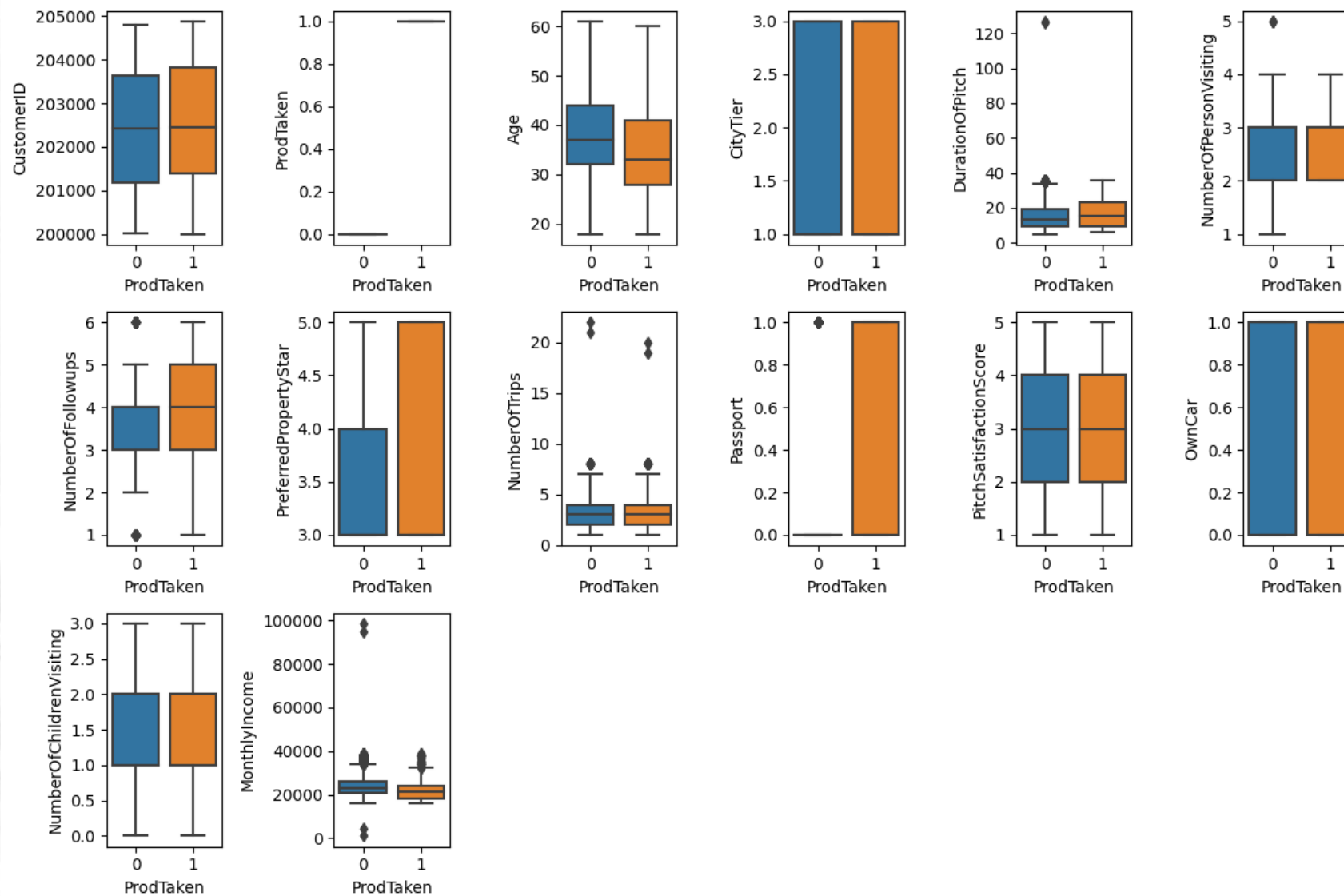
```
df[num_cols].describe().T
```

	count	mean	std	min	25%	50%	75%	max
CustomerID	4888.0	202443.500000	1411.188388	200000.0	201221.75	202443.5	203665.25	204887.0
ProdTaken	4888.0	0.188216	0.390925	0.0	0.00	0.0	0.00	1.0
Age	4662.0	37.622265	9.316387	18.0	31.00	36.0	44.00	61.0
CityTier	4888.0	1.654255	0.916583	1.0	1.00	1.0	3.00	3.0
DurationOfPitch	4637.0	15.490835	8.519643	5.0	9.00	13.0	20.00	127.0
NumberOfPersonVisiting	4888.0	2.905074	0.724891	1.0	2.00	3.0	3.00	5.0
NumberOfFollowups	4843.0	3.708445	1.002509	1.0	3.00	4.0	4.00	6.0
PreferredPropertyStar	4862.0	3.581037	0.798009	3.0	3.00	3.0	4.00	5.0
NumberOfTrips	4748.0	3.236521	1.849019	1.0	2.00	3.0	4.00	22.0
Passport	4888.0	0.290917	0.454232	0.0	0.00	0.0	1.00	1.0
PitchSatisfactionScore	4888.0	3.078151	1.365792	1.0	2.00	3.0	4.00	5.0
OwnCar	4888.0	0.620295	0.485363	0.0	0.00	1.0	1.00	1.0
NumberOfChildrenVisiting	4822.0	1.187267	0.857861	0.0	1.00	1.0	2.00	3.0
MonthlyIncome	4655.0	23619.853491	5380.698361	1000.0	20346.00	22347.0	25571.00	98678.0

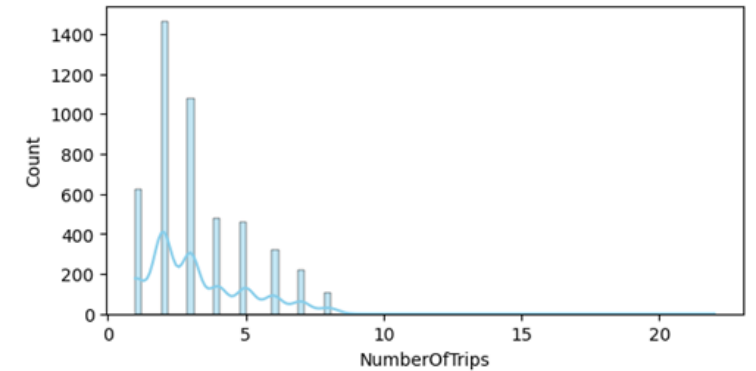
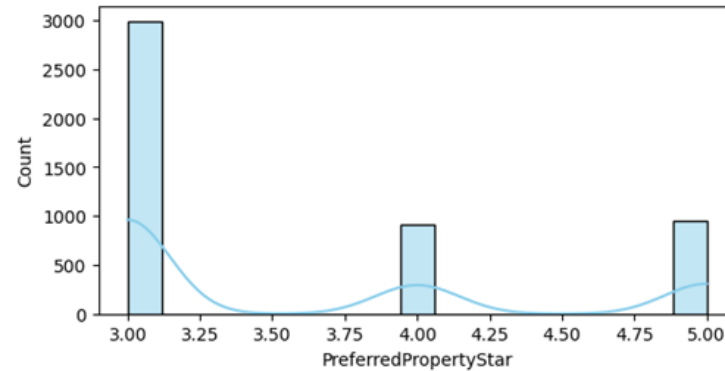
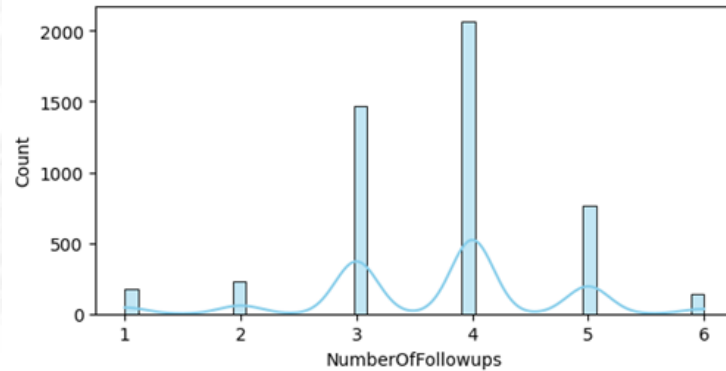
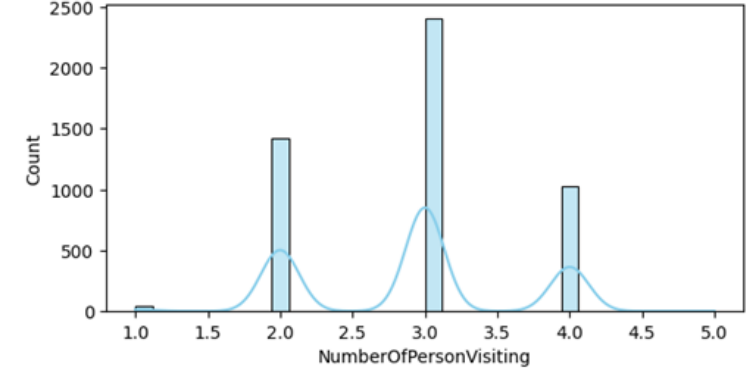
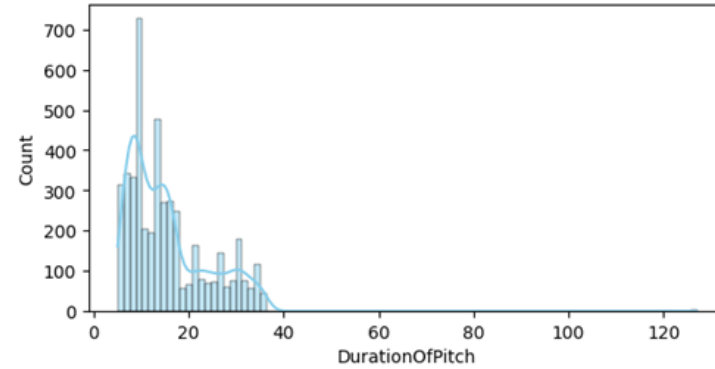
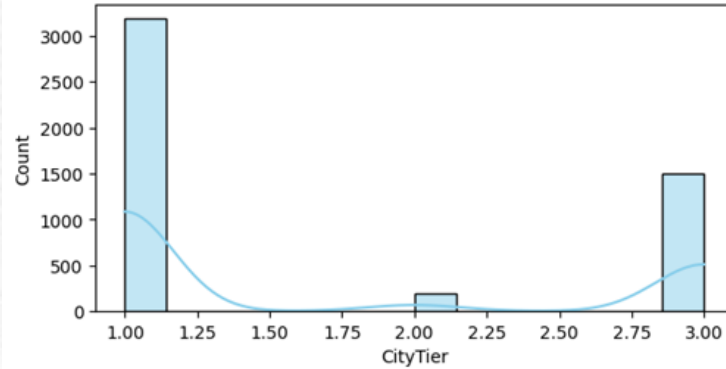
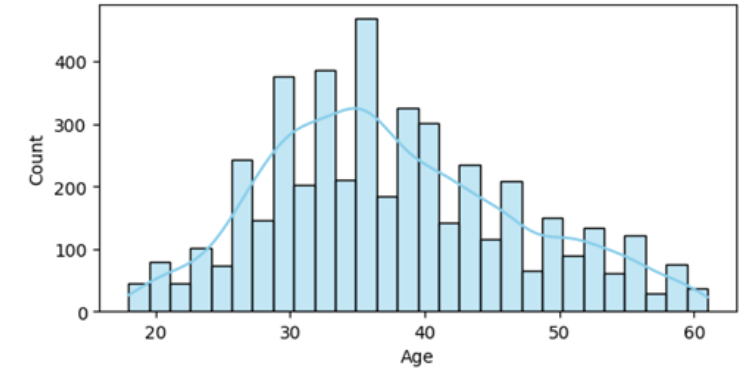
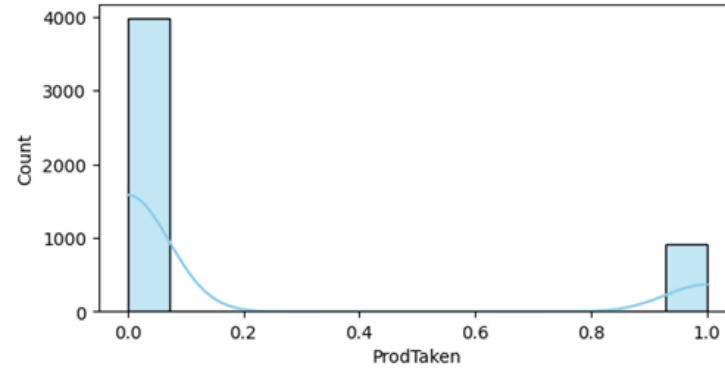
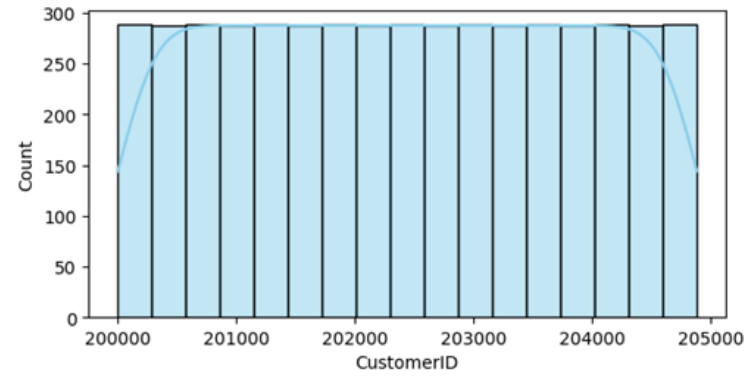
Descriptive Statistics

1. Jumlah baris dalam dataset adalah 4.888.
2. Terdapat beberapa kolom yang memiliki missing value, yaitu kolom Age, DurationOfPitch, NumberOfTrips, PreferredPropertyStar, dan MonthlyIncome. Kolom-kolom ini perlu ditangani dengan metode pengisian nilai yang sesuai sebelum dilakukan analisis lebih lanjut.
3. Rata-rata usia (Age) pelanggan dalam dataset adalah 37.62 tahun dengan standar deviasi sebesar 9.32. Distribusi usia cenderung mendekati distribusi normal.
4. Durasi presentasi (DurationOfPitch), jumlah perjalanan (NumberOfTrips), dan pendapatan bulanan (MonthlyIncome) memiliki skewness positif, yang menunjukkan adanya pencilan atau outlier dalam data.
5. Sebagian besar pelanggan (75%) berada di CityTier 1.
6. Mayoritas pelanggan tidak memiliki paspor (Passport) dan tidak memiliki mobil pribadi (OwnCar).
7. Nilai kepuasan terhadap presentasi (PitchSatisfactionScore) rata-rata adalah 3.08 dari skala 1 hingga 5.
8. Mayoritas pelanggan (75%) memiliki jumlah anak yang mengunjungi (NumberOfChildrenVisiting) sebanyak 1.
9. Pendapatan bulanan (MonthlyIncome) rata-rata pelanggan adalah sebesar 23,619.85 dengan standar deviasi sebesar 5,380.70.
10. Terdapat kolom-kolom diskrit atau ordinal lainnya yang tidak mencerminkan distribusi data tertentu.

Univariate Analysis



Univariate Analysis



Univariate Analysis

1. Pada kolom TypeofContact distribusi Self Enquiry lebih banyak daripada Company Visited
2. Pada kolom Occupation distribusi Salaried dan Small Business paling banyak dibanding yang lainnya, dan pada Free Lancer nantinya akan dihapus karena tidak ada nilainya
3. Pada kolom Gender jenis kelamin laki-laki lebih banyak dibandingkan perempuan. Dan ada kesalahan penulisan Fe Male yang dimana seharusnya Female.
4. Pada kolom ProductPitched basic dan Deluxe memiliki penjualan tertinggi
5. Pada kolom MaritalStatus Customer yang sudah menikah memiliki jumlah paling banyak
6. Pada kolom Designation customer dengan jabatan Executive dan Manager memiliki jumlah paling banyak

Handling Missing Values

```
[ ] #df_mv4 = df.copy()
    col_to_impute = ['Age', 'DurationOfPitch', 'NumberOfFollowups', 'PreferredPropertyStar', 'NumberOfTrips', 'NumberOfChildrenVisiting', 'MonthlyIncome']

[ ] from sklearn.experimental import enable_iterative_imputer
    from sklearn.impute import IterativeImputer

    imputer = IterativeImputer(random_state=100, max_iter=10)
    #df_mv4[col_to_impute] = imputer.fit_transform(df_mv4[col_to_impute])
    df_copy[col_to_impute] = imputer.fit_transform(df_copy[col_to_impute])

[ ] #df_mv4['TypeofContact'].fillna(df_mv4['TypeofContact'].mode()[0], inplace=True)
    df_copy['TypeofContact'].fillna(df_copy['TypeofContact'].mode()[0], inplace=True)
```

Opsi yang kami ambil untuk handle missing value adalah menggunakan metode MICE karena imputasi dilakukan menggunakan model regresi.

Handling Missing Values

```
#df_mv4.isnull().sum()
df_copy.isnull().sum()
```

```
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
```

Setelah dilakukan proses imputasi untuk missing values menggunakan metode MICE, dataset sudah tidak ada kolom yang kosong.

Handling Invalid Values

Berdasarkan hasil pengamatan EDA:

- Terdapat kesalahan pada penulisan kolom Gender dimana "Fe Male" bermakna sama dengan "Female". Maka "Female" akan direplace menjadi "Female"
- Terdapat penggunaan istilah yang berbeda pada kolom Marital status yaitu "Unmarried" dan "Single" dimana kedua status tersebut bermakna sama. Maka "Unmarried" akan direplace menjadi "Single"

```
#Terdapat kesalahan penulisan pada kolom Gender dimana 'Fe Male' seharusnya 'Female'  
df_copy['Gender'] = df_copy['Gender'].replace({'Fe Male': 'Female'})  
  
#Terdapat penggunaan istilah yang berbeda pada 'Unmarried' dan 'Single' dimana kedua status itu sama  
df_copy['MaritalStatus'] = df_copy['MaritalStatus'].replace({'Unmarried': 'Single'})
```

Handling Duplicates Data

Karena keterbatasan data, metode yang kami pakai untuk handling outlier adalah Z-Score.

```
[ ] # view total rows before filtered
print(f'Jumlah baris sebelum memfilter outlier adalah {df_copy.shape[0]}')

# handle outlier using z-score
filtered_entries = np.array([True] * len(df_copy))
skewed_cols = ['DurationOfPitch', 'NumberOfTrips', 'MonthlyIncome']
for col in skewed_cols:
    zscore = abs(stats.zscore(df_copy[col]))
    filtered_entries = (zscore < 3) & filtered_entries

# view total rows after filtered
df_copy = df_copy[filtered_entries]
print(f'Jumlah baris setelah memfilter outlier adalah {df_copy.shape[0]}')
```

```
Jumlah baris sebelum memfilter outlier adalah 4747
Jumlah baris setelah memfilter outlier adalah 4737
```

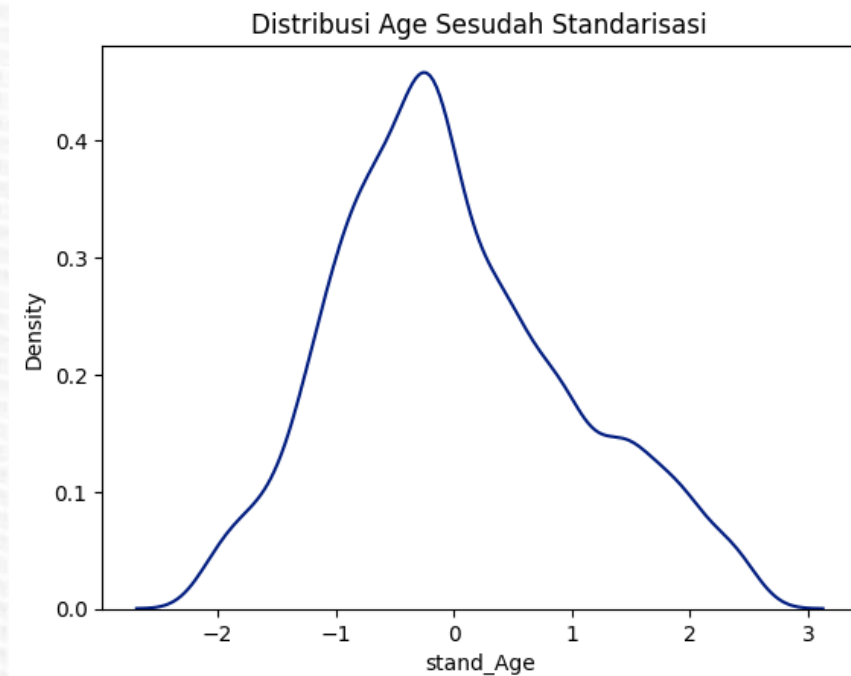
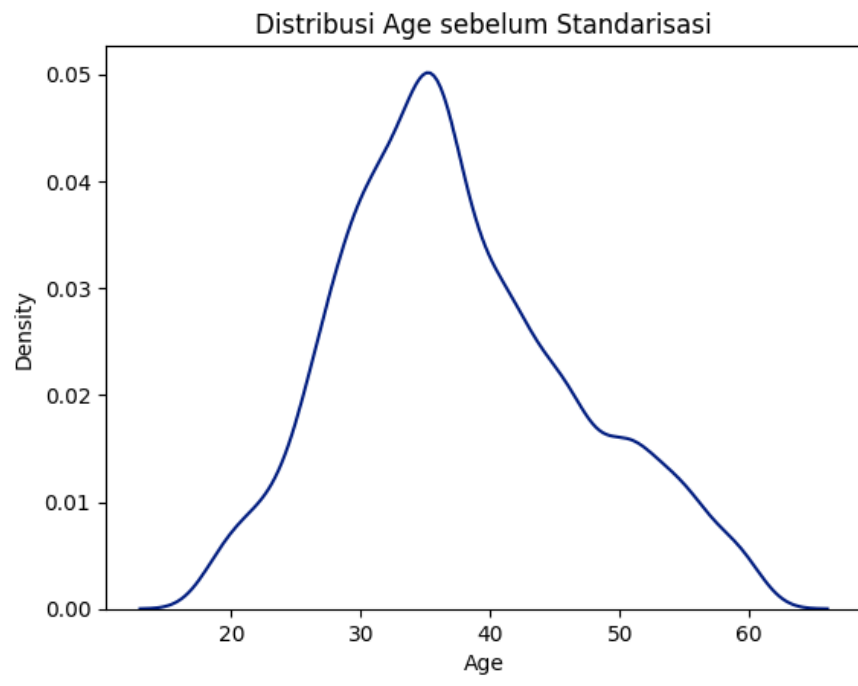
Feature Extraction

Pada feature extraction kita akan menggabungkan kolom NumberOfPersonVisiting dan NumberOfChildrenVisiting menjadi kolom baru yaitu TotalVisiting.

```
▶ # Merge column NumberOfPersonVisiting & NumberOfChildrenVisiting  
df_copy['TotalVisiting'] = df_copy['NumberOfPersonVisiting'] + df_copy['NumberOfChildrenVisiting']
```

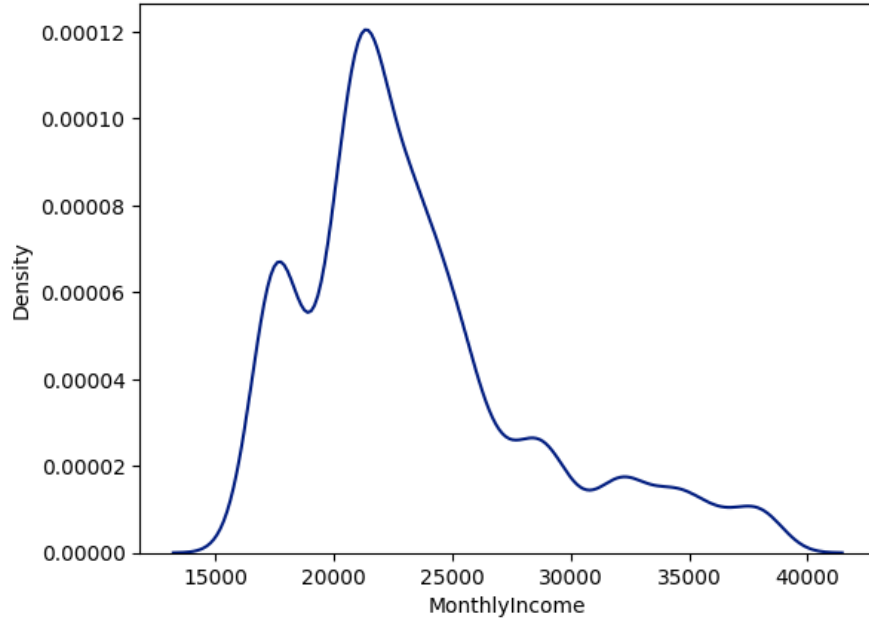
Feature Transformation

Berdasarkan insight yang kita dapat dari EDA, ada beberapa fitur yang berkorelasi dengan target tapi belum berdistribusi normal atau skew positif. Maka kita lakukan log Transformasi dan Standarisasi agar distribusinya normal/ mendekati normal.

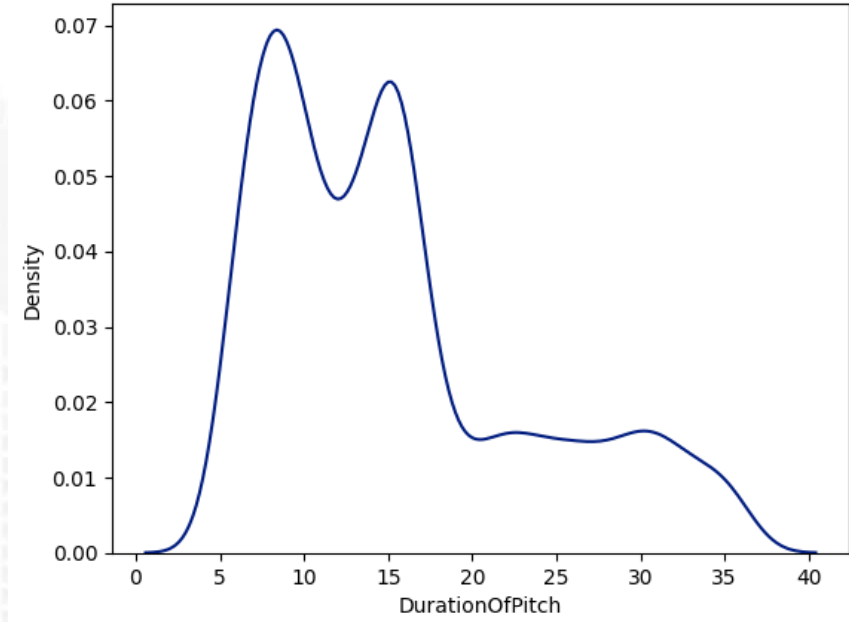


karena distribusi sudah normal maka dilakukan standarisasi

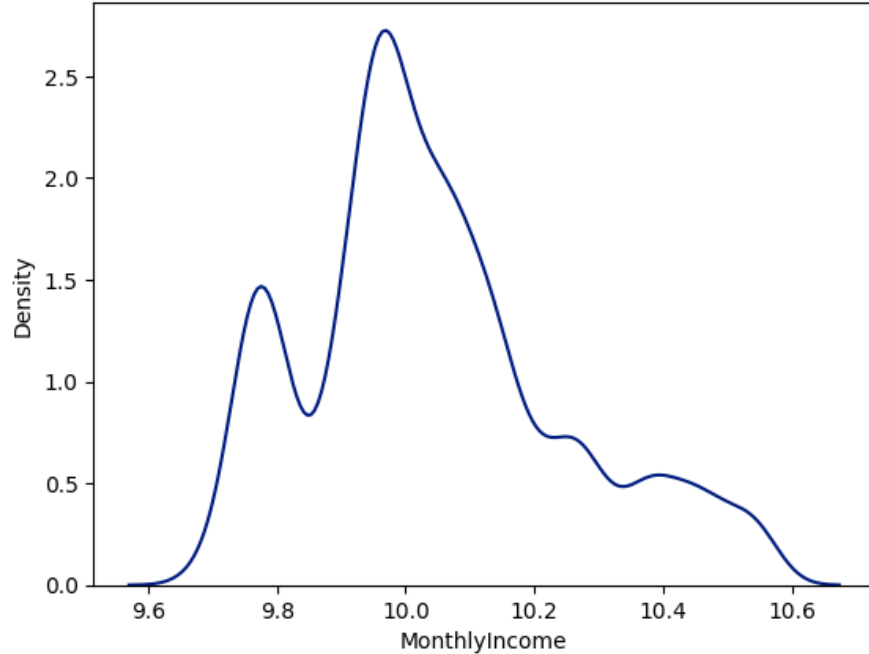
Distribusi MonthlyIncome Sebelum Log Transformation



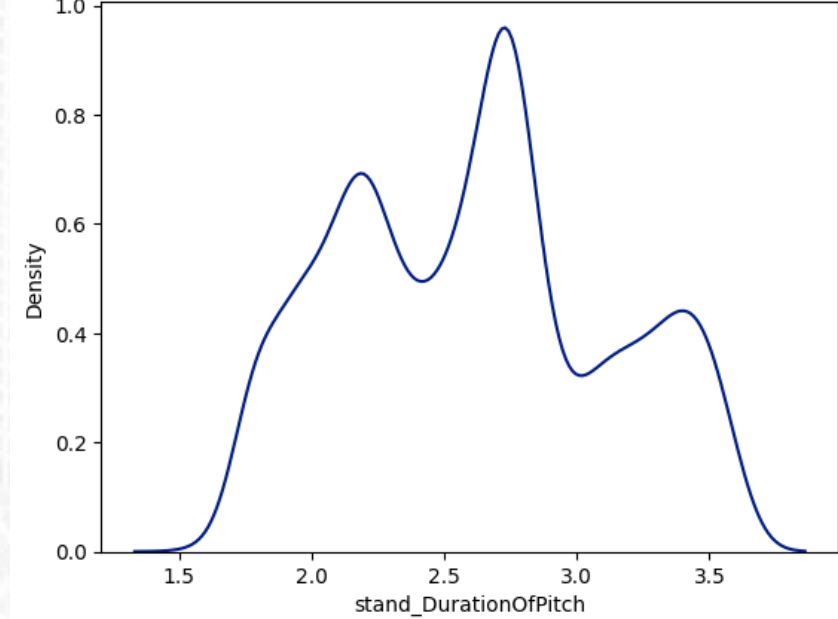
Distribusi DurationOfPitch Sebelum Log Transformation



Distribusi MonthlyIncome Setelah Log Transformation



Distribusi DurationOfPitch Setelah Log Transformation



Feature Encoding

Pada Feature Encoding categorical dibagi menjadi dua metode yaitu encoding yang bertipe data ordinal dan encoding yang bertipe data selain ordinal dan menggunakan librari **LabelEncoder** dan **OneHotEncoder** dari sklearn

```
[ ] # Encoding kolom kategorikal dengan tipe data ordinal
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
df_copy.Occupation=encoder.fit_transform(df_copy.Occupation)
df_copy.Designation=encoder.fit_transform(df_copy.Designation)
df_copy.ProductPitched=encoder.fit_transform(df_copy.ProductPitched)
df_copy.sample(5, random_state=50).T
```

Feature Encoding

```
# Encoding kolom kategorikal selain tipe ordinal
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder

# Creating instance of OneHotEncoder
encoder = OneHotEncoder()

# Define the columns to be one-hot encoded
columns_to_encode = ['Gender', 'TypeofContact', 'MaritalStatus']

# Create the ColumnTransformer
ct = ColumnTransformer([('encoder', encoder, columns_to_encode)], remainder='passthrough')

# Apply one-hot encoding to the selected columns
encoded_data = ct.fit_transform(df_copy)

# Create a new DataFrame with the encoded data
encoded_df = pd.DataFrame(encoded_data)

# Update column names for one-hot encoded columns
encoded_columns = ct.named_transformers_['encoder'].get_feature_names_out(columns_to_encode)
new_columns = list(encoded_columns) + list(df_copy.columns.drop(columns_to_encode))
encoded_df.columns = new_columns

# Update df_copy with the encoded DataFrame
df_copy = encoded_df
```

Sebelum Feature Encoding

ProdTaken	1	0	0	1
TypeofContact	Self Enquiry	Self Enquiry	Company Invited	Self Enquiry
CityTier	3	1	1	2
Occupation	2	3	2	2
Gender	Female	Male	Female	Male
NumberOfPersonVisiting	3	3	3	3
NumberOfFollowups	3.0	5.0	4.0	2.0
ProductPitched	1	0	0	0
PreferredPropertyStar	3.0	3.0	3.0	5.0
MaritalStatus	Single	Single	Single	Married
NumberOfTrips	1.0	2.0	2.0	2.0
Passport	1	0	0	0
PitchSatisfactionScore	2	2	5	1
OwnCar	1	0	0	1
NumberOfChildrenVisiting	0.0	1.0	1.0	2.0
Designation	2	1	1	1
TotalVisiting	3.0	4.0	4.0	5.0

Setelah Feature Encoding

Gender_Female	1.000000	0.000000	0.000000	1.000000	0.000000
Gender_Male	0.000000	1.000000	1.000000	0.000000	1.000000
TypeofContact_Company Invited	0.000000	1.000000	0.000000	1.000000	0.000000
TypeofContact_Self Enquiry	1.000000	0.000000	1.000000	0.000000	1.000000
MaritalStatus_Divorced	0.000000	1.000000	0.000000	1.000000	1.000000
MaritalStatus_Married	0.000000	0.000000	0.000000	0.000000	0.000000
MaritalStatus_Single	1.000000	0.000000	1.000000	0.000000	0.000000
ProdTaken	1.000000	0.000000	1.000000	0.000000	0.000000
CityTier	3.000000	1.000000	1.000000	1.000000	1.000000
Occupation	2.000000	2.000000	0.000000	2.000000	3.000000
NumberOfPersonVisiting	3.000000	3.000000	3.000000	2.000000	2.000000
NumberOfFollowups	3.000000	4.000000	4.000000	3.000000	3.000000
ProductPitched	1.000000	1.000000	0.000000	0.000000	0.000000
PreferredPropertyStar	3.000000	4.000000	3.000000	3.000000	4.000000
NumberOfTrips	1.000000	2.000000	7.000000	2.000000	1.000000
PitchSatisfactionScore	2.000000	3.000000	3.000000	5.000000	5.000000
OwnCar	1.000000	1.000000	0.000000	1.000000	1.000000
NumberOfChildrenVisiting	0.000000	2.000000	0.000000	1.000000	0.000000
Designation	2.000000	2.000000	1.000000	1.000000	1.000000
TotalVisiting	3.000000	5.000000	3.000000	3.000000	2.000000

Feature Selection

Pada feature selection kita memakai 2 metode yang digunakan untuk menentukan fitur-fitur apa saja yang digunakan pada pemodelan dan mempunyai korelasi terhadap target yaitu metode **ANOVA** dan metode **RandomForest**

Opsi 1: ANOVA

```
[ ] # separate feature and target
X = df_copy.drop(['ProdTaken'], axis=1, inplace=False)
y = df_copy['ProdTaken'].values

[ ] # import library
from sklearn.feature_selection import f_regression, SelectKBest

# Applying SelectKBest class to extract top 10 best features
fs = SelectKBest(score_func=f_regression,k=10)
# Applying feature selection
fit = fs.fit(X,y)

[ ]
features_score = pd.DataFrame(fit.scores_)
features = pd.DataFrame(X.columns)
feature_score = pd.concat([features,features_score],axis=1)
# Assigning column names
feature_score.columns = ["Input_Features","F_Score"]
print(feature_score.nlargest(15,columns="F_Score"))
```

Input_Features	F_Score
Passport	349.887262
MaritalStatus_Single	189.338584
ProductPitched	131.313478
stand_Age	99.316231
log_MonthlyIncome	91.005589
MaritalStatus_Married	73.987553
NumberOfFollowups	62.739475
PreferredPropertyStar	44.675740
Designation	43.418107
CityTier	36.756101
stand_DurationOfPitch	32.227653
MaritalStatus_Divorced	25.768527
PitchSatisfactionScore	12.767022
TypeofContact_Company Invited	8.149897
TypeofContact_Self Enquiry	8.149897

Hasil F-Score dari metode ANOVA menunjukkan bahwa tidak ada fitur yang terlalu dominan sehingga relatif terdistribusi.

Feature Selection

Opsi 2 : Random Forest Feature Performance

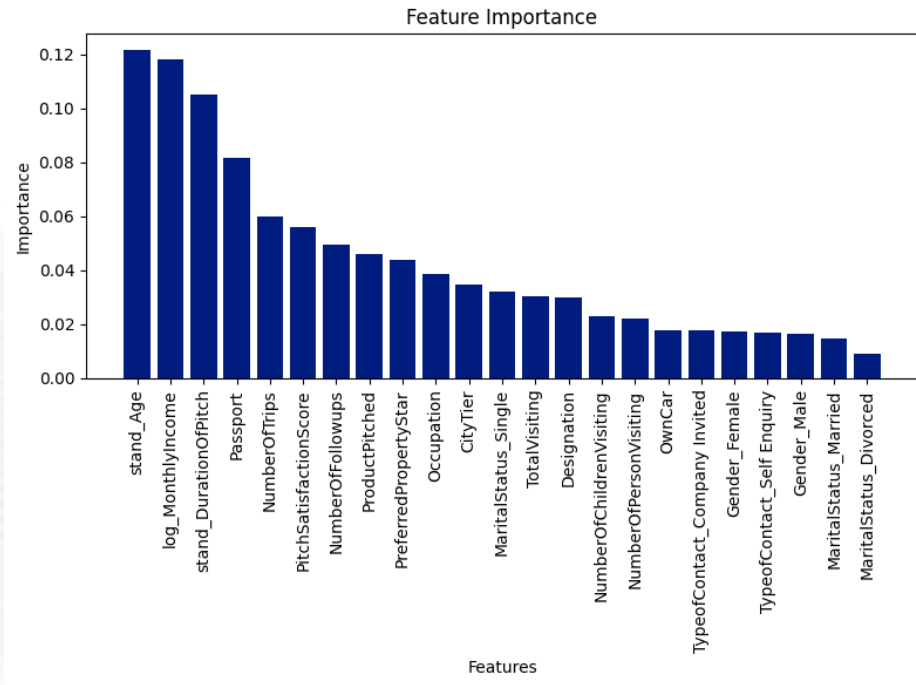
```
[ ] #Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
▶ # Train Random Forest classifier
rf = RandomForestClassifier()
rf.fit(X, y)

# Extract feature importances
importances = rf.feature_importances_
feature_names = X.columns

# Sort feature importances in descending order
indices = np.argsort(importances)[::-1]
sorted_importances = importances[indices]
sorted_feature_names = feature_names[indices]
```

Feature Selection



Hasil Feature Performance dari metode Random Forest menunjukkan bahwa tidak ada fitur yang terlalu dominan sehingga relatif terdistribusi

```
[ ] # separate feature and target
X = df_copy.drop(['ProdTaken', 'DurationOfPitch', 'NumberOfFollowups', 'ProductPitched', 'PitchSatisfactionScore'], axis=1, inplace=False)
y = df_copy['ProdTaken'].values
```

Kami melakukan separasi antara dataframe target dengan dataframe fitur. Kami juga melakukan drop beberapa kolom transaksi karena prediksi yang akan kita lakukan adalah untuk customer baru.

Split Train-Test Data

```
[ ] #Split the data
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=42, stratify=y)
```

```
[ ] # check the shape of X_train and X_test

print(f'Jumlah data latih terdiri dari {len(X_train)} baris')
print(f'Jumlah data uji terdiri dari {len(X_test)} baris')
```

```
Jumlah data latih terdiri dari 3315 baris
Jumlah data uji terdiri dari 1422 baris
```

Split data dilakukan menggunakan rasio 70% data training dan 30% data testing

Handling Data Imbalance

Karena adanya ketimpangan atau ketidakseimbangan dalam jumlah data pada kolom target maka perlu dilakukan oversampling. Oversampling akan dilakukan menggunakan metode **SMOTE**.

```
[ ] from imblearn import under_sampling, over_sampling
    X_over_SMOTE, y_over_SMOTE = over_sampling.SMOTE(random_state=40).fit_resample(X_train, y_train)
    print(X_over_SMOTE.shape)
    print(y_over_SMOTE.shape)
```

```
(5382, 18)
(5382,)
```

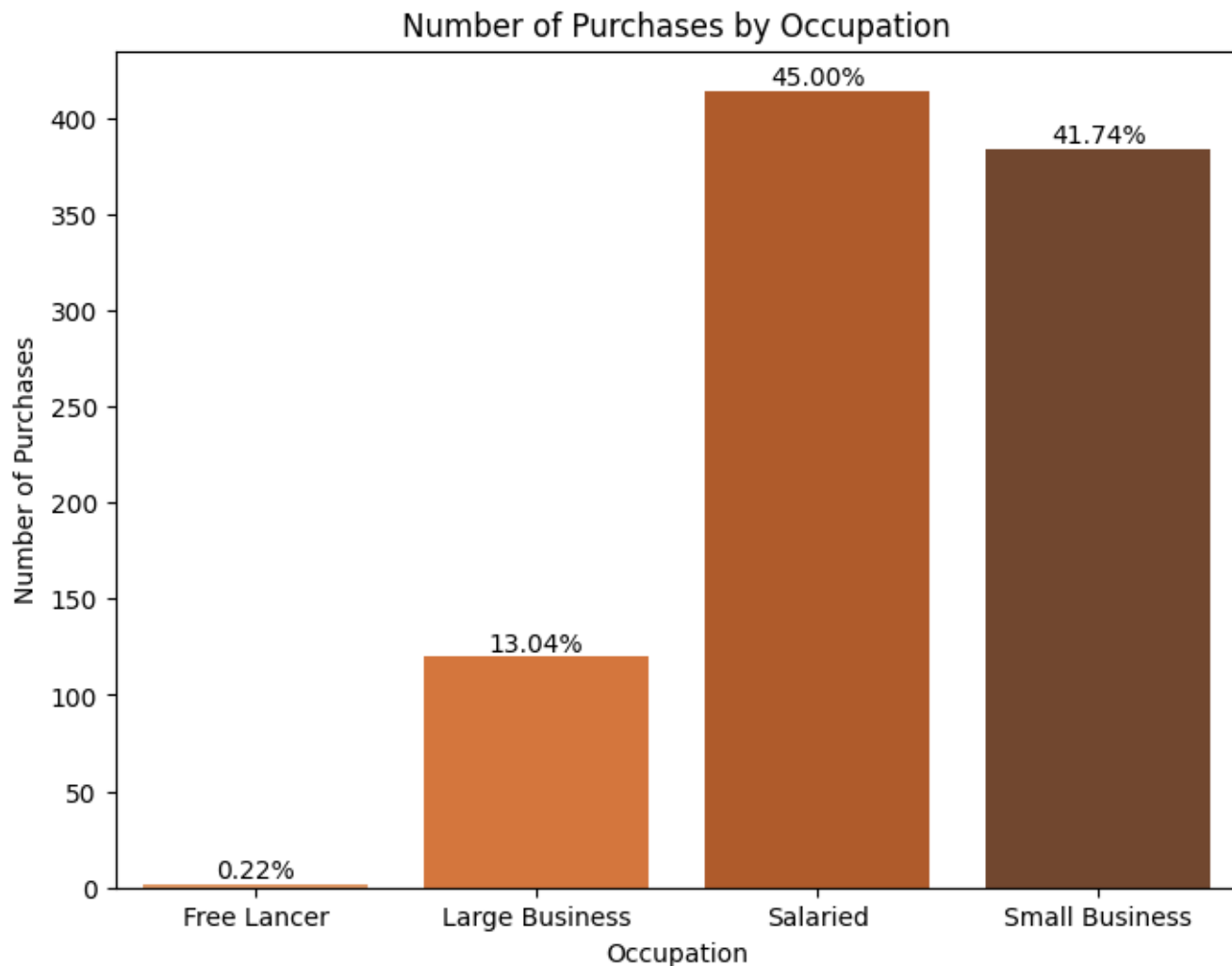
```
[ ] print('Original')
    print(pd.Series(y_train).value_counts())
    print('\n')
    print('SMOTE')
    print(pd.Series(y_over_SMOTE).value_counts())
```

```
Original
0.0    2691
1.0     624
dtype: int64
```

```
SMOTE
0.0    2691
1.0    2691
dtype: int64
```

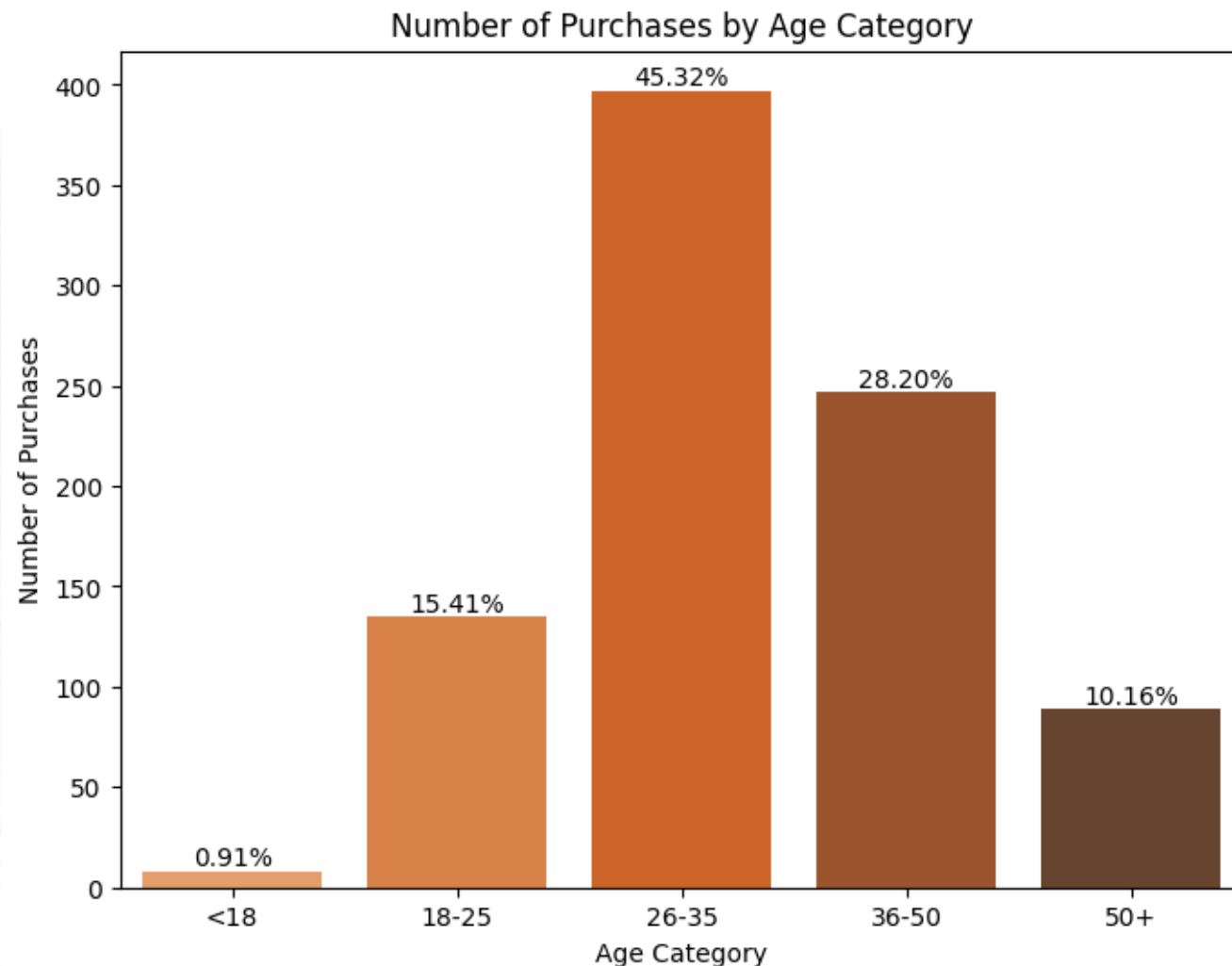
Setelah dilakukan oversampling menggunakan SMOTE maka dapat dilihat bahwa jumlah data pada target sudah terdistribusi secara merata, oleh karena itu siap untuk dilakukan pemodelan.

Insight



Mayoritas yang membeli paket adalah **karyawan/pengusaha**

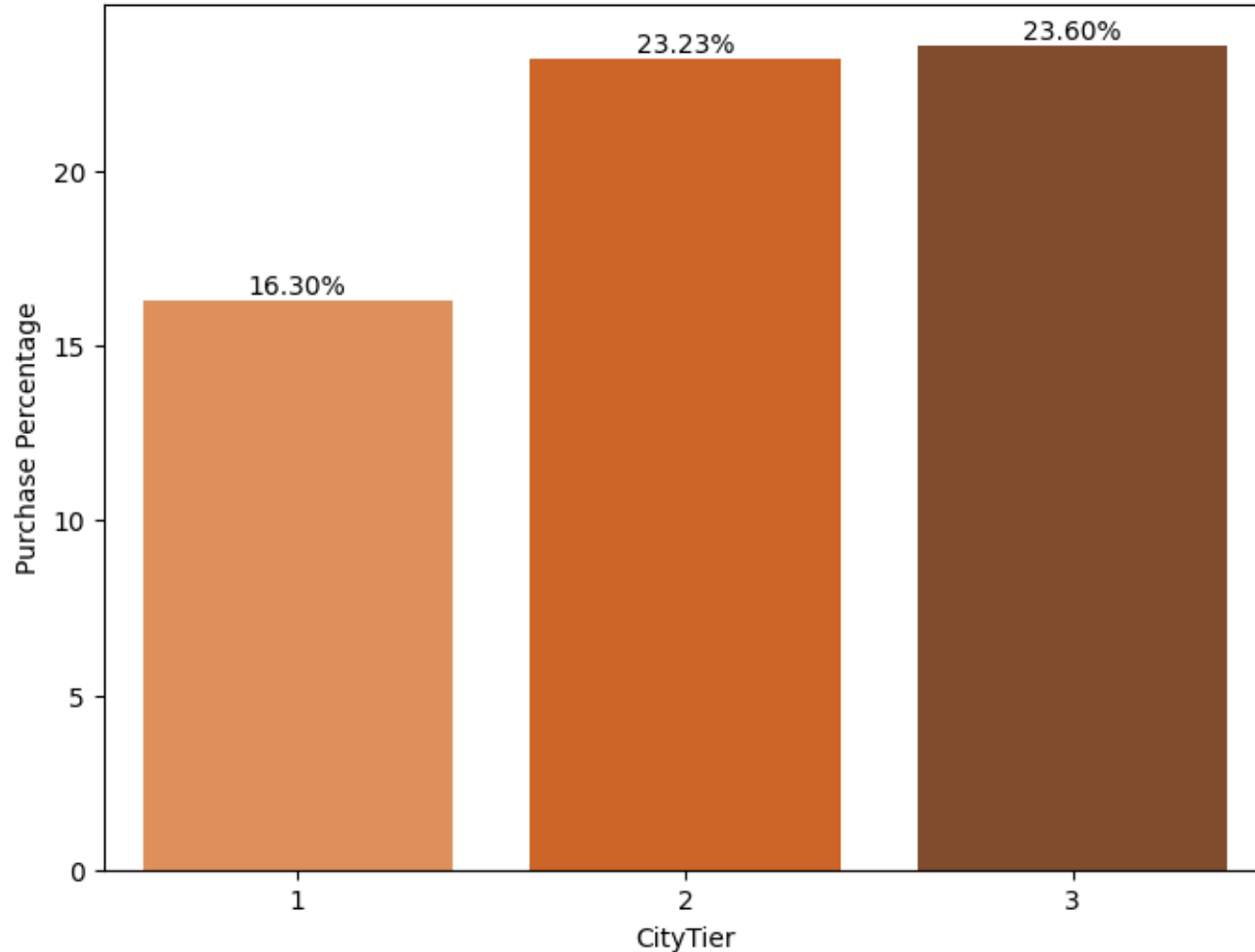
Insight



Mayoritas yang membeli paket adalah **individu produktif** dengan **rentang usia 26 hingga 35 tahun**

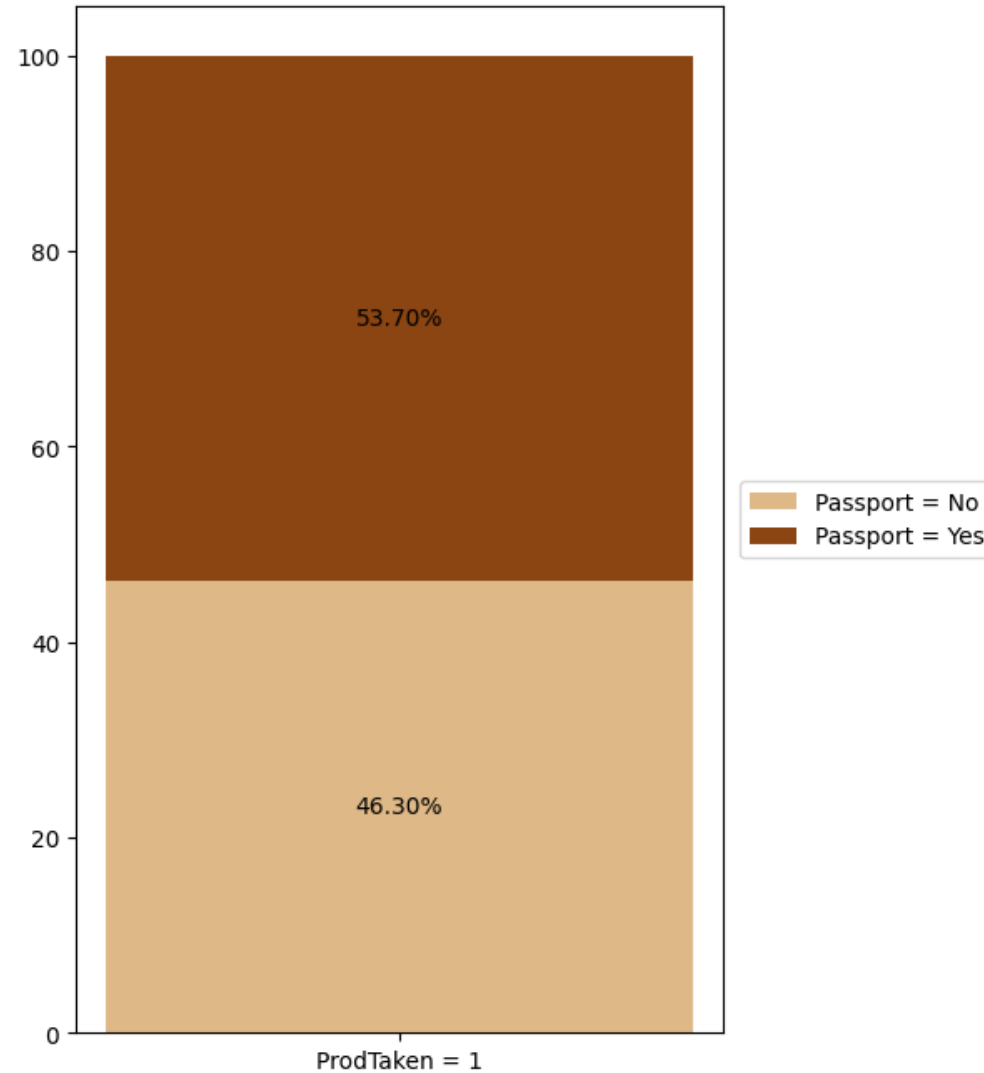
Insight

Purchase Percentage by CityTier



Orang yang tinggal di **Tier 2 dan Tier 3** **memiliki daya beli lebih besar** daripada populasi Tier 1

Insight



Orang yang **memiliki paspor** kemungkinan besar akan membeli passport

Modelling Experiments

	Accuracy	Precision	Recall	F1	AUC
XGBClassifier	86.50	68.63	52.24	59.32	87.73
RandomForestClassifier	85.58	68.86	42.91	52.87	86.58
ExtraTreesClassifier	86.15	69.83	46.64	55.93	85.85
BaggingClassifier	84.81	63.4	45.9	53.25	86.48
GradientBoostingClassifier	83.33	57.49	44.4	50.11	82.29

Dari perbandingan 5 model, **XGBClassifier** memiliki nilai keseluruhan yang paling baik. Dalam kasus yang dihadapi, kami memilih skor precision sebagai metrik utama karena kami bertujuan menaikkan conversion rate dengan mengurangi angka *false positive* (FP).

Modelling Experiments

```
[ ] params = {
    'max_depth' : [int(x) for x in np.linspace(10, 110, num = 11)],
    'min_child_weight' : [int(x) for x in np.linspace(1, 20, num = 11)],
    'gamma' : [float(x) for x in np.linspace(0, 1, num = 11)],
    'tree_method' : ['auto', 'exact', 'approx', 'hist'],

    'colsample_bytree' : [float(x) for x in np.linspace(0, 1, num = 11)],
    'eta' : [float(x) for x in np.linspace(0, 1, num = 100)],

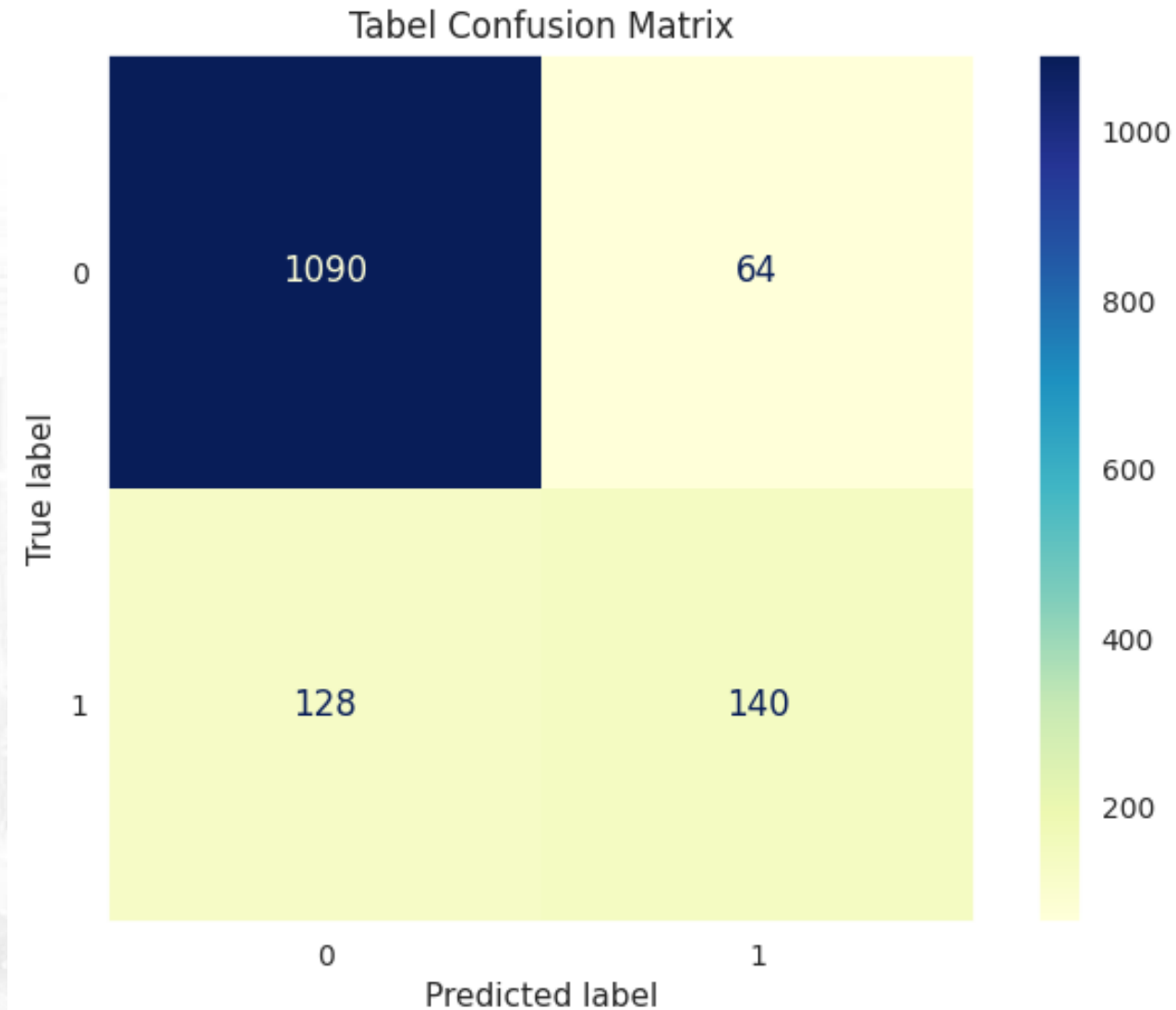
    'lambda' : [float(x) for x in np.linspace(0, 1, num = 11)],
    'alpha' : [float(x) for x in np.linspace(0, 1, num = 11)]
}

xgb_tuned = RandomizedSearchCV(
    estimator=XGBClassifier(),
    param_distributions=params,
    n_iter=10,
    scoring='recall',
    cv=5,
    n_jobs=-1)
```

Setelah dilakukan hyperparameter tuning, ternyata model mengalami penurunan nilai akurasi pada data test dan semakin memperbesar perbedaan nilai akurasi data training dan data test sehingga terindikasi overfitting. Karena secara keseluruhan model mengalami penurunan performa, maka kita akan menggunakan default parameter.

	train_acc	test_acc	precision	recall	f1-score	roc_auc (train-proba)	roc_auc (test-proba)
xgb	99.31	86.50	68.63	52.24	59.32	99.99	87.73
xgb_tuned	94.76	45.52	64.55	45.52	53.39	99.99	87.73

Modelling Result



Modelling Result



Modelling Result

Conversion Rate Simulation

All Customer	1,422
Conversion Rate	18%

Before Model

	Customer
True Positive	140
False Positive	64
True Negative	1,090
False Negative	128
Conversion Rate	68%

After Model

$$\begin{aligned}
 & \text{TP} / (\text{TP} + \text{FP}) \\
 &= 140 / (140 + 64) \\
 &= 68\%
 \end{aligned}$$

Modelling Result

Cost Efficiency Simulation

Cost per Follow Up = IDR 10,000

Num of Follow Up 4 = Average Num of Follow Up from historical data

Num of Follow Up 6 = Max Num of Follow Up from historical data

	Customer	Num of Follow Up	Cost (IDR)
All Customer	1,422	4	56,880,000
Total			56,880,000

Before Model

	Customer	Num of Follow Up	Cost (IDR)
True Positive	140	6	8,400,000
False Positive	64	6	3,840,000
True Negative	1,090	1	10,900,000
False Negative	128	1	1,280,000
Total			24,420,000

After Model

Modelling Result

Customer Acquisition Cost Simulation

	Total Follow Up	Total Customer	Marketing Budget (IDR)	Cost (IDR)	Marketing Budget Balance (IDR)	Customer Acquisition Cost	Additional Customer Pool
Before Model	5,688	1,422	56,880,000	56,880,000	0	40,000	0
After Model	2,422	1,422	56,880,000	24,420,000	32,460,000	17,172	1,890

Executive Summary & Recommendation

1	Menyediakan layanan pendaftaran paspor untuk membantu calon pelanggan memilih tujuan wisata yang beragam dalam paket travel yang ditawarkan
2	Memfokuskan saluran pemasaran di media sosial untuk menjangkau segmen pelanggan terbesar yaitu yang melek teknologi, berusia 26 hingga 35 tahun
3	Kampanye pemasaran untuk menargetkan profil pelanggan yang tepat, salah satu dari kondisi berikut : <ul style="list-style-type: none">• tinggal di city tier 2 & 3• karyawan• pengusaha
4	Mengirimkan notifikasi email untuk menginformasikan calon pelanggan dan pembeli tentang promo baru atau paket khusus
5	Memprioritaskan tindak lanjut kepada pelanggan potensial (hingga 6x tindak lanjut) daripada pelanggan non-potensial (1x tindak lanjut) untuk meningkatkan efisiensi anggaran pemasaran

Pembagian Tugas

Tony Hermawan Widjanarko : Project Leader, Modelling, Business Recommendation & Final Presentation

Esraminar Siregar : Exploratory Data Analysis, Data Preprocessing & Business Recommendation

Rayhan Prawira Daksa : Data Preprocessing, Modelling & Business Recommendation

Rianita : Exploratory Data Analysis, Data Preprocessing & Business Recommendation

Farhan Rizki : Exploratory Data Analysis, Data Preprocessing & Notulensi

Ryan Anugrah : Data Preprocessing, Modelling & Business Recommendation