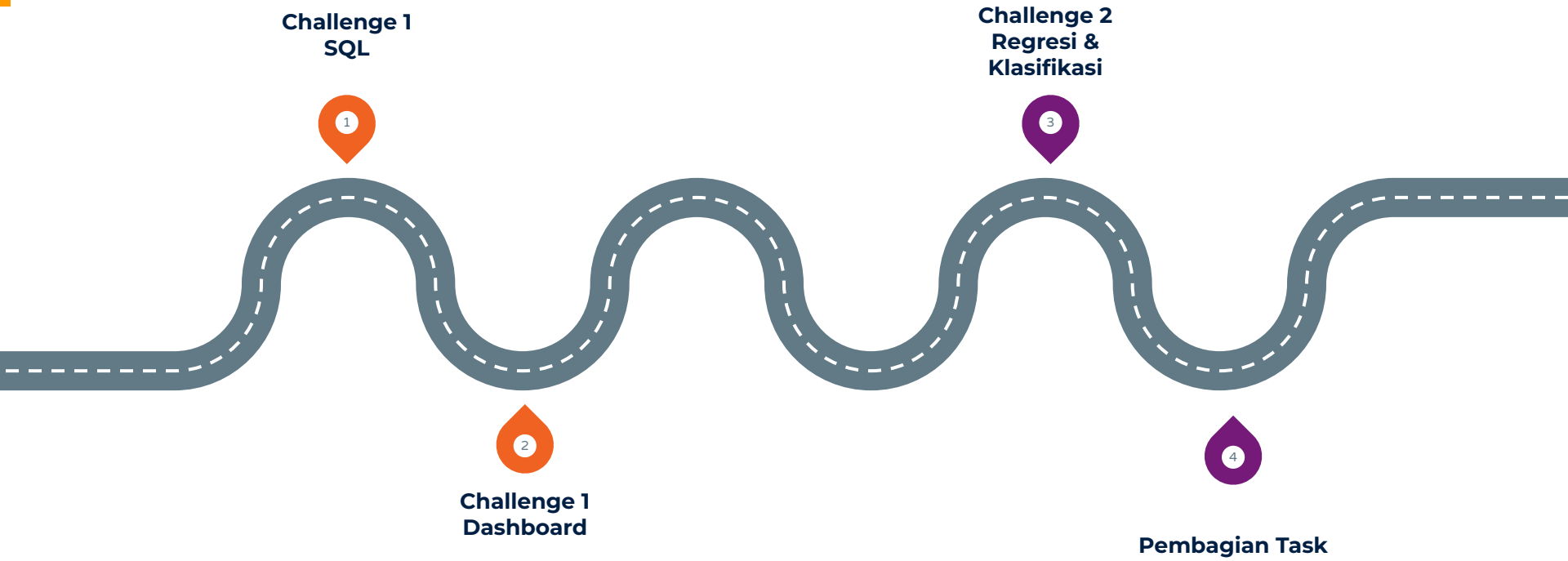
Final Presentation

DS1-12 - Data Science

Anggota Tim

TIM	NAMA
DS1-13	Taufiq Qurohman Ruki
	Aleisya Zahari Salam

Outline


Challenge 1
SQL
1
Challenge 1
Dashboard
2
Challenge 2
Regresi &
Klasifikasi
3
4
Pembagian Task

Challenge Chapter 01

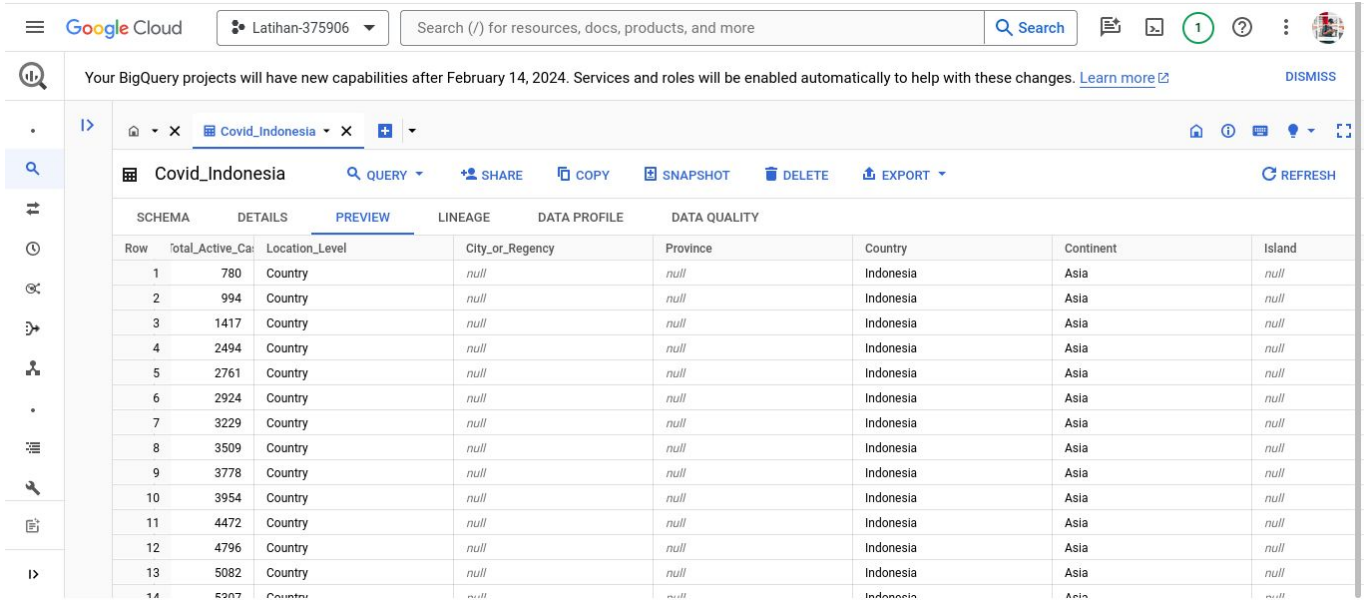
QUERY SQL

Kami baru saja direkrut jadi junior data scientist di tim data dan teknologi kesehatan. User membutuhkan insight dari data yang sangat banyak tentang kasus Covid-19 di Indonesia

QUERY SQL

Misi Pertama
Jumlah total kasus Covid-19 aktif yang baru di setiap provinsi lalu diurutkan berdasarkan jumlah kasus yang paling besar
Mengambil 2 (dua) location iso code yang memiliki jumlah total kematian karena Covid-19 paling sedikit
Data tentang tanggal-tanggal ketika rate kasus recovered di Indonesia paling tinggi beserta jumlah ratenya
Total case fatality rate dan case recovered rate dari masing-masing location iso code yang diurutkan dari data yang paling rendah
Data tentang tanggal-tanggal saat total kasus Covid-19 mulai menyentuh angka 30.000-an
Jumlah data yang tercatat ketika kasus Covid-19 lebih dari atau sama dengan 30.000

Bahasan



Row	Total_Active_Ca	Location_Level	City_or_Regency	Province	Country	Continent	Island
1	780	Country	null	null	Indonesia	Asia	null
2	994	Country	null	null	Indonesia	Asia	null
3	1417	Country	null	null	Indonesia	Asia	null
4	2494	Country	null	null	Indonesia	Asia	null
5	2761	Country	null	null	Indonesia	Asia	null
6	2924	Country	null	null	Indonesia	Asia	null
7	3229	Country	null	null	Indonesia	Asia	null
8	3509	Country	null	null	Indonesia	Asia	null
9	3778	Country	null	null	Indonesia	Asia	null
10	3954	Country	null	null	Indonesia	Asia	null
11	4472	Country	null	null	Indonesia	Asia	null
12	4796	Country	null	null	Indonesia	Asia	null
13	5082	Country	null	null	Indonesia	Asia	null

Dataset Covid Berhasil Di Import ke BigQuery

Bahasan

1. Jumlah total kasus Covid-19 baru setiap provinsi dari yang terbanyak

Provinsi Jawa Barat memiliki jumlah kasus baru terbanyak yaitu 13.496 kasus, dan yang terendah Provinsi Sulawesi Barat dengan jumlah 6 kasus

```
-- soal 1
SELECT Province, SUM(New_Active_Cases) jumlah_kasus_baru
FROM challenge01.kasus_covid
WHERE Province IS NOT NULL
GROUP BY Province
ORDER BY jumlah_kasus_baru DESC
```

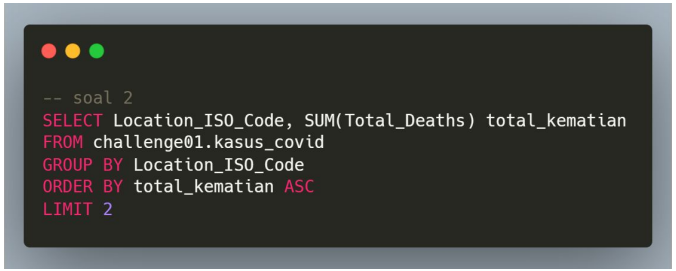
Row	Province	jumlah_kasus_baru
1	Jawa Barat	13496
2	DKI Jakarta	10922
3	Banten	2558
4	Jawa Tengah	1423
5	Jawa Timur	1136
6	Daerah Istimewa Yogyakarta	669
7	Sumatera Utara	664
8	Sulawesi Utara	565
9	Bali	474
10	Sumatera Selatan	313
11	Kalimantan Timur	272
12	Papua	237

Row	Province	jumlah_kasus_baru
23	Nusa Tenggara Timur	102
24	Sulawesi Tengah	90
25	Jambi	77
26	Kepulauan Bangka Belitung	73
27	Maluku	49
28	Bengkulu	34
29	Sulawesi Tenggara	34
30	Gorontalo	31
31	Kalimantan Utara	27
32	Nusa Tenggara Barat	15
33	Maluku Utara	14
34	Sulawesi Barat	6

Bahasan

2. Dua *Location iso code* dengan jumlah total kematian karena Covid-19 paling sedikit

Dua *location iso code* yang memiliki jumlah total kematian karena Covid-19 paling sedikit yaitu **ID-MA** dengan total kematian 147.196 dan **ID-MU** dengan total kematian 167.511



```
-- soal 2
SELECT Location_ISO_Code, SUM(Total_Deaths) total_kematian
FROM challenge01.kasus_covid
GROUP BY Location_ISO_Code
ORDER BY total_kematian ASC
LIMIT 2
```

Row	Location_ISO_Code	total_kematian
1	ID-MA	147196
2	ID-MU	167511

Bahasan

3. Tanggal-tanggal dengan rate kasus recovered paling tinggi dengan jumlah ratenya di Indonesia

Rate kasus recovered di Indonesia paling tinggi dengan tingkat rate 97.37%

```sql
-- soal 3
SELECT Date,
    ROUND(Total_Recovered/Total_Cases*100) Case_Recovered_Rate
FROM challenge01.kasus_covid
WHERE Location= 'Indonesia'
ORDER BY Case_Recovered_Rate DESC
```

Row	Date	Case_Recovered_Rate
1	2022-05-29	97.37
2	2022-05-27	97.37
3	2022-05-23	97.37
4	2022-05-30	97.37
5	2022-05-24	97.37
6	2022-05-26	97.36
7	2022-06-01	97.36
8	2022-06-03	97.36
9	2022-05-25	97.36
10	2022-05-31	97.36
11	2022-06-05	97.36
12	2022-06-02	97.36

Bahasan

4. Total *case fatality rate* dan *case recovered rate berdasarkan location* dari yang paling rendah

Total *case fatality rate* dan *case recovered rate* berdasarkan *location* yang diurutkan dari data yang paling rendah

```sql
-- soal
WITH fatality_rate_od AS(
  SELECT Location AS fatality_rate_location,
    SUM(Case_Fatality_Rate) AS Total_Case_Fatality_Rate,
    ROW_NUMBER( ) OVER (ORDER BY SUM(Case_Fatality_Rate) ASC) AS order_fatality_rate
  FROM challenge01.kasus_covid
  GROUP BY Location
  ORDER BY Total_Case_Fatality_Rate ASC
),
recovered_rate_od AS(
  SELECT Location AS recovered_rate_location,
    SUM(Case_Recovered_Rate) AS Total_Case_Recovered_Rate,
    ROW_NUMBER( ) OVER (ORDER BY SUM(Case_Recovered_Rate) ASC) AS order_recovered_rate
  FROM challenge01.kasus_covid
  GROUP BY Location
  ORDER BY Total_Case_Recovered_Rate ASC
)

SELECT
  fatality_rate_location,Total_Case_Fatality_Rate,
  recovered_rate_location,Total_Case_Recovered_Rate
FROM fatality_rate_od
JOIN recovered_rate_od ON fatality_rate_od.order_fatality_rate = recovered_rate_od.order_recovered_rate
```

Bahasan

Hasil SQL 4

Total *case fatality rate* dan *case recovered rate* berdasarkan *location* yang diurutkan dari data yang paling rendah

Row	fatality_rate_location	Total_Case_Fatality_Rate	recovered_rate_location	Total_Case_Recovered_Rate
1	Kalimantan Utara	14.285000000000021	Papua	608.23260000000084
2	Nusa Tenggara Timur	15.934500000000002	Nusa Tenggara Timur	700.82079999999894
3	Papua	16.895300000000013	Lampung	703.80569999999989
4	Jambi	17.326799999999977	Aceh	709.14220000000114
5	Sulawesi Tenggara	19.668699999999919	Maluku Utara	718.42810000000179
6	Kalimantan Barat	20.560999999999932	Sumatera Selatan	722.4222999999995
7	Sulawesi Barat	21.755600000000072	Bengkulu	722.54159999999877
8	Sulawesi Selatan	22.457400000000142	Kalimantan Selatan	728.18399999999906
9	Sumatera Barat	24.010300000000047	Nusa Tenggara Barat	730.55370000000062
10	Papua Barat	24.334100000000088	Indonesia	730.55679999999961
11	Maluku Utara	24.628000000000114	Sulawesi Tengah	732.01280000000008
12	Kepulauan Bangka Belitung	24.991900000000012	Sulawesi Barat	732.87229999999954
13	Kalimantan Timur	25.197899999999986	Kalimantan Utara	733.72659999999894
14	Gorontalo	30.654200000000088	Sulawesi Tenggara	741.66440000000159

Bahasan

5. Tanggal-tanggal total kasus Covid-19 menyentuh angka 30.000-an

Total kasus Covid-19 pertama yang mencapai 30.000-an kasus ada pada tanggal 2020-06-06

```sql
-- soal 5
SELECT Date,Total_Cases
FROM challenge01.kasus_covid
WHERE Total_Cases >= 30000 and Location =
'Indonesia'
LIMIT 10
```

Row	Date	Total_Cases
1	2020-06-06	30514
2	2020-06-07	31186
3	2020-06-08	32033
4	2020-06-09	33075
5	2020-06-10	34316
6	2020-06-11	35295
7	2020-06-12	36406
8	2020-06-13	37420
9	2020-06-14	38277
10	2020-06-15	39294

Bahasan

6. Jumlah data dengan kasus covid lebih dari atau sama dengan 30.000

Jumlah data yang tercatat ketika kasus covid lebih dari atau sama dengan 30.000 yaitu **833 data.**

```sql
-- soal 6
SELECT
    COUNT(*) AS Jumlah_Data
FROM
    challenge01.kasus_covid
WHERE
    Total_Cases >= 30000 and Location = 'Indonesia'
```

Row	Jumlah_Data
1	833

DASHBOARD

Challenge 1

Latar Belakang

COVID-19 telah menjadi pandemi global yang mempengaruhi kehidupan manusia di seluruh dunia. Penyebarannya yang cepat dan dampaknya yang luas telah menuntut tanggapan yang cepat dan efektif dari berbagai pihak, termasuk pemerintah, lembaga kesehatan, dan masyarakat umum.

Masalah & Solusi

Masalah

Sejak kasus pertama COVID-19 dilaporkan pada awal tahun 2020, penanganan pandemi telah mengalami berbagai tahap, mulai dari mitigasi awal hingga upaya vaksinasi massal. Dalam upaya untuk memahami dan mengelola penyebaran COVID-19 dengan lebih baik, banyak negara dan lembaga telah mengembangkan berbagai alat, termasuk dashboard interaktif.

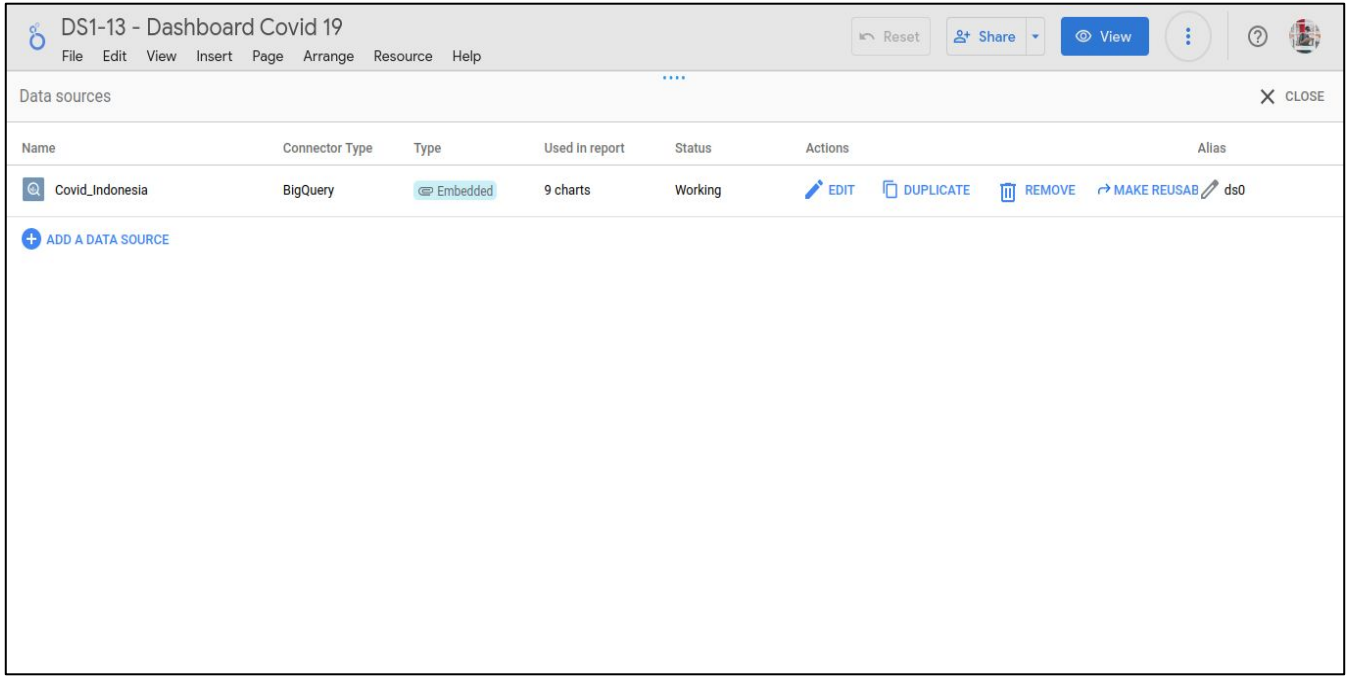
Solusi

Dashboard adalah platform visual yang menyajikan informasi melalui grafik dan tabel untuk memudahkan pemahaman situasi terkini, seperti penyebaran COVID-19. Pengguna dapat melakukan analisis data dan mengambil keputusan berdasarkan informasi yang disajikan. Sebagai alat interaktif, dashboard membantu berbagai pihak, mulai dari pemerintah hingga individu, dalam mengelola informasi dengan lebih efisien.

Tujuan

Tujuan utama dari pembangunan dashboard COVID-19 ini adalah untuk memantau penyebaran COVID-19 dan sebagai alat bagi pembuat kebijakan lembaga kesehatan maupun masyarakat umum untuk mengambil keputusan yang tepat berbasis data.

Bahasan

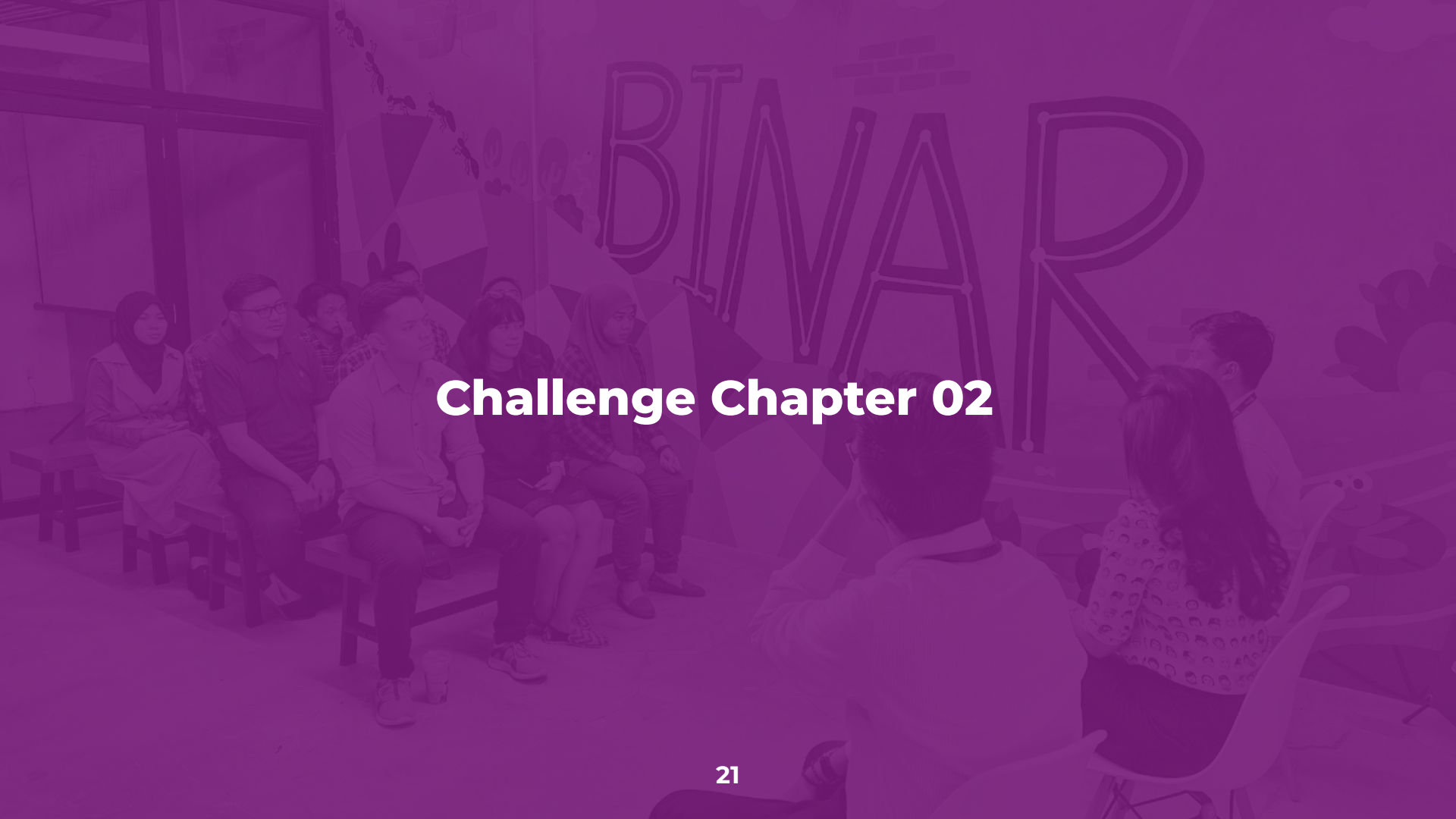


Data Berhasil di Import Ke Looker dari BigQuery

Bahasan

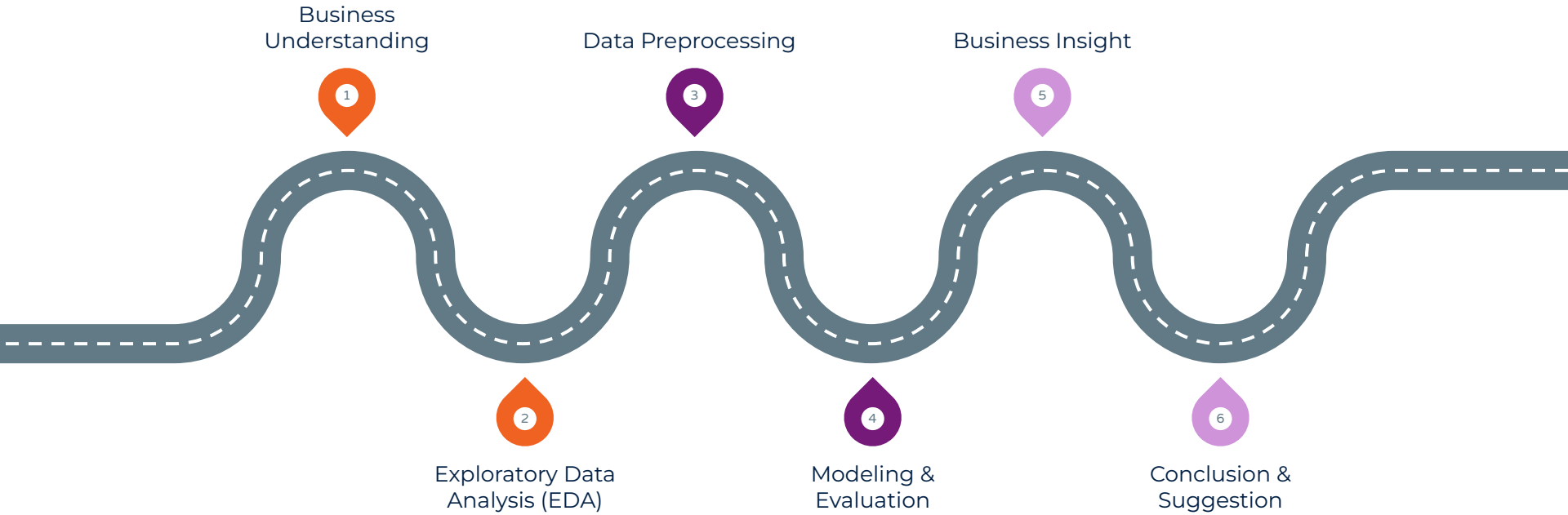
Tekan Tombol

DASHBOARD



Challenge Chapter 02

Outline


Business Understanding
1
Exploratory Data Analysis (EDA)
2
Data Preprocessing
3
Modeling & Evaluation
4
Business Insight
5
Conclusion & Suggestion
6

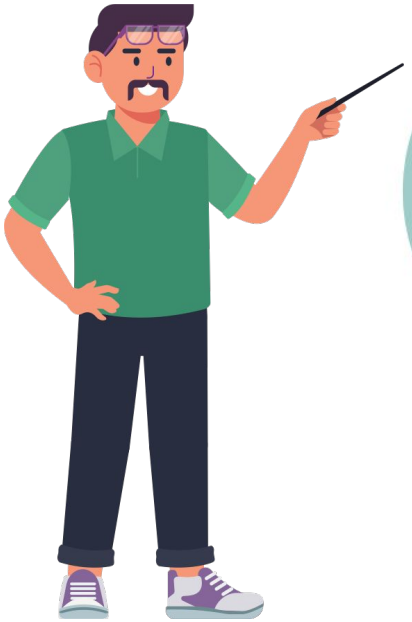
Latar Belakang

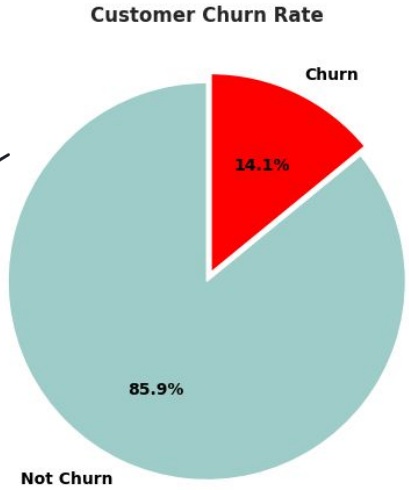
Telekomunikasi menjadi salah satu pilar utama dalam menghubungkan jutaan orang di seluruh dunia melalui teknologi informasi yang terus berkembang.

Peningkatan permintaan akan koneksi internet yang cepat dan andal, didorong oleh inovasi teknologi seperti 5G dan infrastruktur serat optik, telah memicu persaingan yang semakin ketat antara perusahaan telekomunikasi dan penyedia layanan internet (ISP)

Perubahan pola konsumen telekomunikasi yang terus berubah

Business Understanding





14,1% dari **4250** pelanggan yang meninggalkan layanan. Hal ini perlu diperhatikan karena tingkat churn rate **yang dapat diterima sekitar 5% - 7%**

Dampak?

1. Revenue dan profit menurun.
2. Citra Perusahaan buruk.

Tujuan

Tujuan dari masalah ini adalah untuk mengidentifikasi pelanggan yang kemungkinan akan beralih menggunakan layanan komunikasi melalui klasifikasi dan prediksi.

Exploratory Data Analysis (EDA)

Info Dataset

```
baris, kolom = df.shape
print('Dataset ini terdiri dari: ')
print(f'{baris} baris')
print(f'{kolom} kolom')
```

```
Dataset ini terdiri dari:
4250 baris
20 kolom
```

```
# cek data duplicate
print('Jumlah data duplicate: ', df.duplicated().sum())
```

✓ 0.0s

```
Jumlah data duplicate:  0
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4250 entries, 0 to 4249
Data columns (total 20 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   state                         4250 non-null   object
 1   account_length                4250 non-null   int64
 2   area_code                     4250 non-null   object
 3   international_plan            4250 non-null   object
 4   voice_mail_plan               4250 non-null   object
 5   number_vmail_messages         4250 non-null   int64
 6   total_day_minutes             4250 non-null   float64
 7   total_day_calls               4250 non-null   int64
 8   total_day_charge              4250 non-null   float64
 9   total_eve_minutes             4250 non-null   float64
 10  total_eve_calls               4250 non-null   int64
 11  total_eve_charge              4250 non-null   float64
 12  total_night_minutes           4250 non-null   float64
 13  total_night_calls             4250 non-null   int64
 14  total_night_charge            4250 non-null   float64
 15  total_intl_minutes            4250 non-null   float64
 16  total_intl_calls              4250 non-null   int64
 17  total_intl_charge             4250 non-null   float64
 18  number_customer_service_calls 4250 non-null   int64
 19  churn                         4250 non-null   object
dtypes: float64(8), int64(7), object(5)
memory usage: 664.2+ KB
```

Dataset terdiri dari **4250 baris** dan **20 kolom.**

Tidak ditemukan nilai null ataupun data **duplikat.**

Exploratory Data Analysis (EDA)

Descriptive Statistic

Dilakukan pada keduanya, kolom numerik dan kategorik

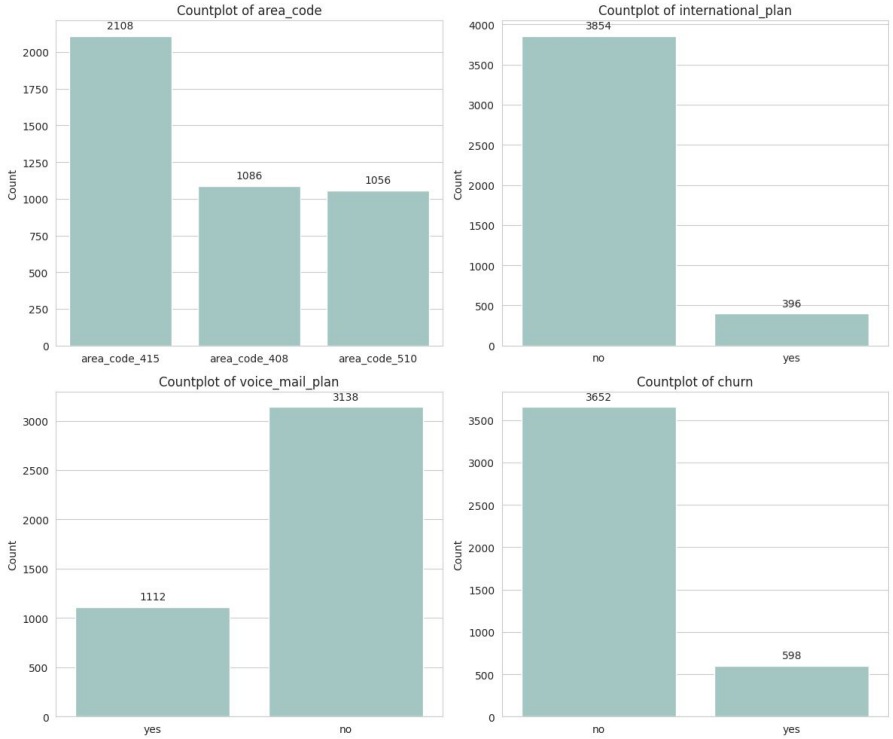
	count	mean	std	min	25%	50%	75%	max
account_length	4250.0	100.236235	39.698401	1.0	73.0000	100.00	127.0000	243.00
number_vmail_messages	4250.0	7.631765	13.439882	0.0	0.0000	0.00	16.0000	52.00
total_day_minutes	4250.0	180.259600	54.012373	0.0	143.3250	180.45	216.2000	351.50
total_day_calls	4250.0	99.907294	19.850817	0.0	87.0000	100.00	113.0000	165.00
total_day_charge	4250.0	30.644682	9.182096	0.0	24.3650	30.68	36.7500	59.76
total_eve_minutes	4250.0	200.173906	50.249518	0.0	165.9250	200.70	233.7750	359.30
total_eve_calls	4250.0	100.176471	19.908591	0.0	87.0000	100.00	114.0000	170.00
total_eve_charge	4250.0	17.015012	4.271212	0.0	14.1025	17.06	19.8675	30.54
total_night_minutes	4250.0	200.527882	50.353548	0.0	167.2250	200.45	234.7000	395.00
total_night_calls	4250.0	99.839529	20.093220	0.0	86.0000	100.00	113.0000	175.00
total_night_charge	4250.0	9.023892	2.265922	0.0	7.5225	9.02	10.5600	17.77
total_intl_minutes	4250.0	10.256071	2.760102	0.0	8.5000	10.30	12.0000	20.00
total_intl_calls	4250.0	4.426353	2.463069	0.0	3.0000	4.00	6.0000	20.00
total_intl_charge	4250.0	2.769654	0.745204	0.0	2.3000	2.78	3.2400	5.40
number_customer_service_calls	4250.0	1.559059	1.311434	0.0	1.0000	1.00	2.0000	9.00

	count	unique	top	freq
state	4250	51	WV	139
area_code	4250	3	area_code_415	2108
international_plan	4250	2	no	3854
voice_mail_plan	4250	2	no	3138
churn	4250	2	no	3652

Exploratory Data Analysis (EDA)

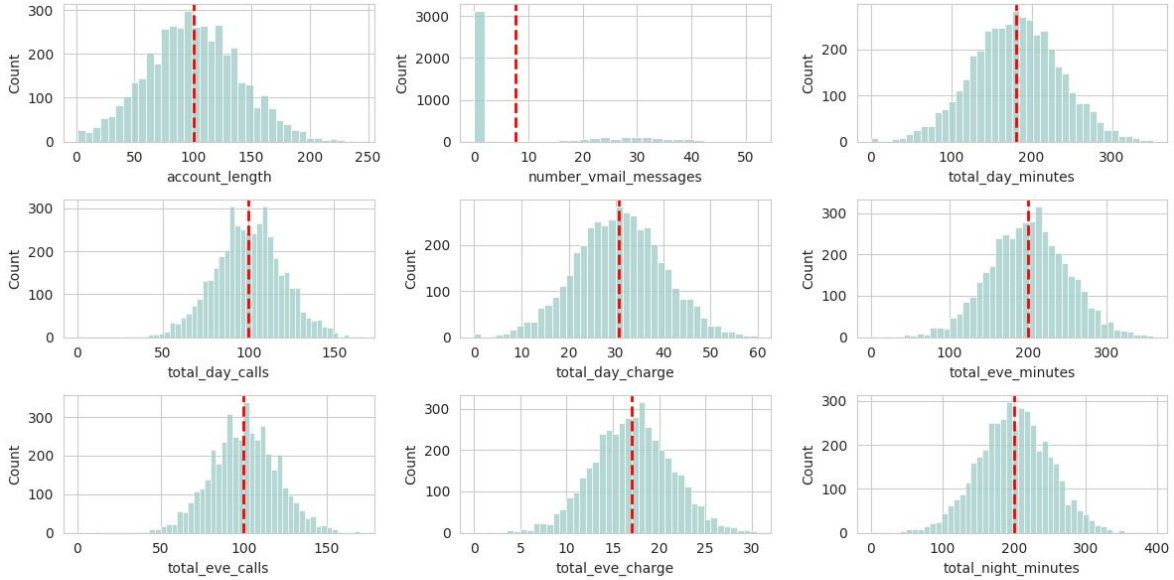
Univariate Analysis

- **Area_code_415** memiliki jumlah kostumer terbanyak
- **Sebagian besar** kostumer memilih untuk **tidak melakukan international plan**
- **3138 kostumer** memilih untuk tidak menggunakan paket voice mail.



Exploratory Data Analysis (EDA)

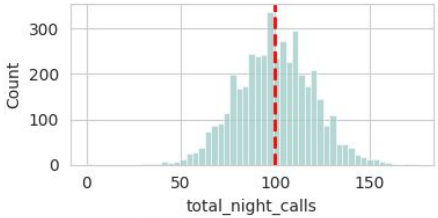
Univariate Analysis

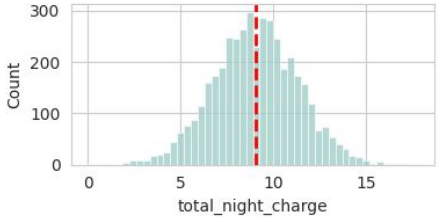


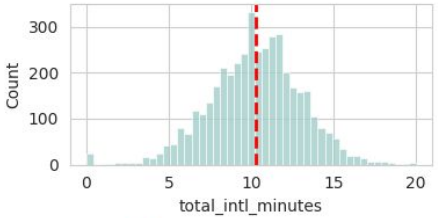
Data sebagian besar terlihat memiliki distribusi normal, tetapi untuk dipastikan akan dilakukan test apakah memiliki distribusi normal atau tidak.

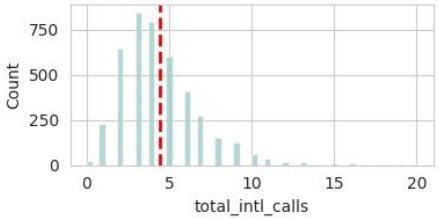
Exploratory Data Analysis (EDA)

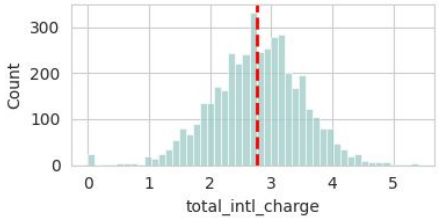
Univariate Analysis











Data sebagian besar terlihat memiliki distribusi normal, tetapi untuk dipastikan akan dilakukan test apakah memiliki distribusi normal atau tidak.

Exploratory Data Analysis (EDA)

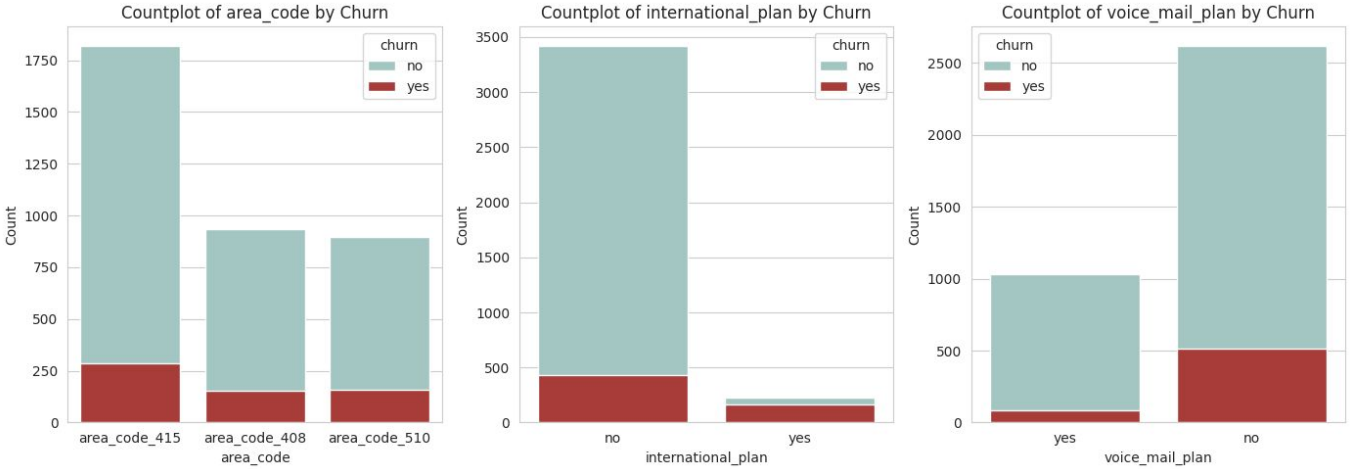
Outliers check

	Column Name	is Outlier	Lower Limit	Upper Limit	Outlier	No Outlier
0	account_length	True	-8.00000	208.00000	20	4230
1	number_vmail_messages	True	-24.00000	40.00000	86	4164
2	total_day_minutes	True	34.01250	325.51250	25	4225
3	total_day_calls	True	48.00000	152.00000	28	4222
4	total_day_charge	True	5.78750	55.32750	26	4224
5	total_eve_minutes	True	64.15000	335.55000	34	4216
6	total_eve_calls	True	46.50000	154.50000	24	4226
7	total_eve_charge	True	5.45500	28.51500	34	4216
8	total_night_minutes	True	66.01250	335.91250	37	4213
9	total_night_calls	True	45.50000	153.50000	33	4217
10	total_night_charge	True	2.96625	15.11625	37	4213
11	total_intl_minutes	True	3.25000	17.25000	62	4188
12	total_intl_calls	True	-1.50000	10.50000	100	4150
13	total_intl_charge	True	0.89000	4.65000	62	4188
14	number_customer_service_calls	True	-0.50000	3.50000	335	3915

Outlier yang dimiliki masing-masing kolom **tidak lebih dari 10%,** serta setiap data yang ada **dianggap penting**. Untuk itu diputuskan untuk tidak menghilangkan outlier

Exploratory Data Analysis (EDA)

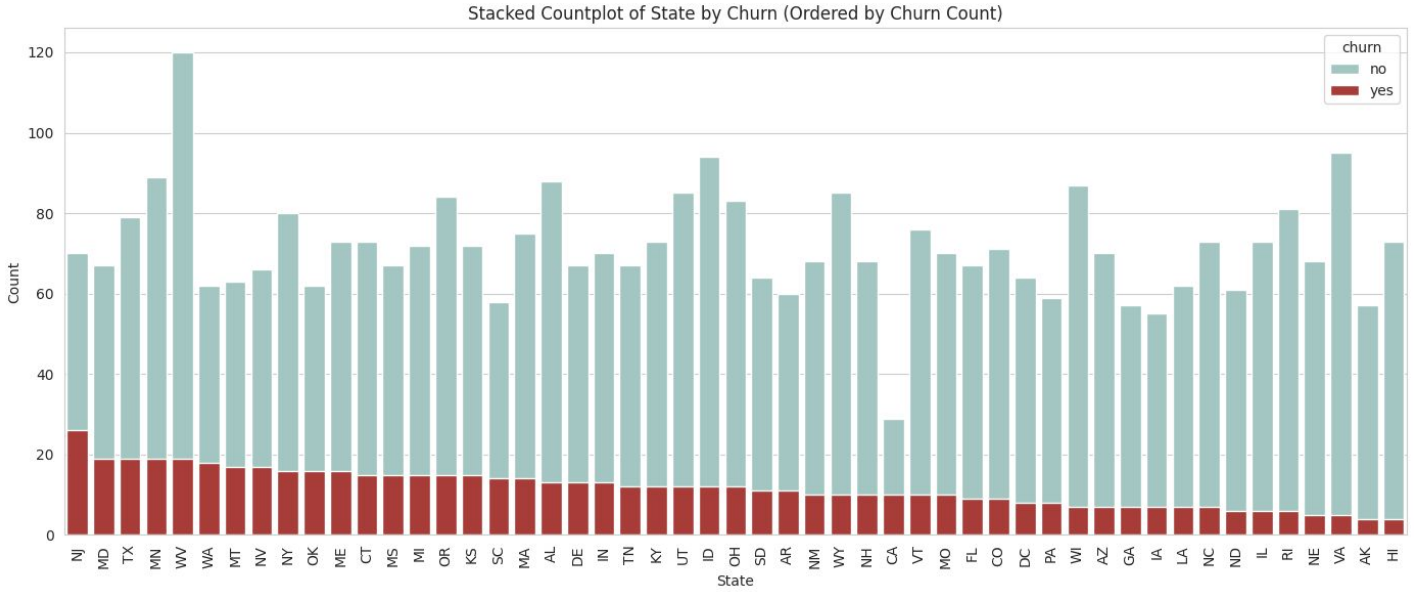
Bivariate Analysis



- **Area_code_415** memiliki **tingkat churn yang lebih tinggi** dibanding area yang lain.
- Kostumer yang **tidak memilih international plan** **lebih banyak yang churn** dibanding yang tidak. Tetapi perlu dipertimbangkan karena jumlah kostumer yang memilih international plan sedikit.
-

Exploratory Data Analysis (EDA)

Bivariate Analysis

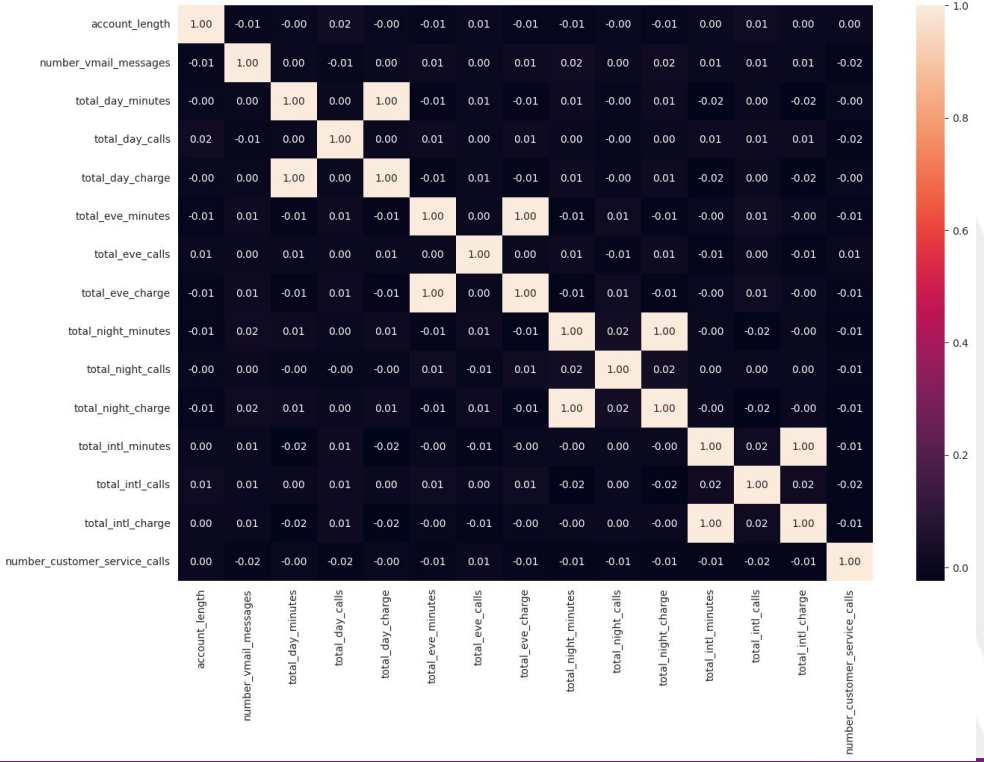


Kode negara **NJ** memiliki tingkat churn yang **paling tinggi**, dan yang **paling rendah** ada pada kode negara **HI.**

Exploratory Data Analysis (EDA)

Korelasi data

Total charge dan **total minutes** memiliki **korelasi yang sangat tinggi.** Untuk itu akan **dihapus** salah satunya, yaitu **total minutes.**



Exploratory Data Analysis (EDA)

Korelasi data

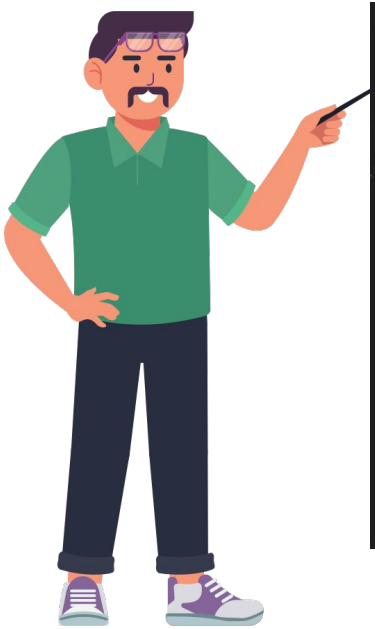
- Untuk data kategorical menggunakan **chi-square.**
- Diputuskan **menghapus area_code,** karena tidak memiliki hubungan yang baik terhadap fitur target.
- Pertimbangan lainnya untuk **menghapus kolom state** karena memiliki 51 nilai unik, dan dianggap tidak memberikan informasi yang cukup penting.

```
# categorical
from scipy.stats import chi2_contingency
cats2 = ['state', 'area_code', 'international_plan', 'voice_mail_plan']
chi2_array, p_array = [], []
for column in cats2:

    crosstab = pd.crosstab(df[column], df['churn'])
    chi2, p, dof, expected = chi2_contingency(crosstab)
    chi2_array.append(chi2)
    p_array.append(p)

df_chi = pd.DataFrame({
    'Variable': cats2,
    'Chi-square': chi2_array,
    'p-value': p_array
})
df_chi.sort_values(by='Chi-square', ascending=False)
```

	Variable	Chi-square	p-value
2	international_plan	282.653490	1.983190e-63
0	state	85.993673	1.169028e-03
3	voice_mail_plan	55.109814	1.139804e-13
1	area_code	1.216654	5.442606e-01

Data Preprocessing

Pengelompokkan data



```
# drop kolom yang tidak dipakai
df_fe = df_fe.drop(columns=['state', 'total_day_minutes', 'total_eve_minutes', 'total_night_minutes', 'total_intl_minutes'])
```

✓ 0.0s

```
nums2 = ['account_length', 'number_vmail_messages',
         'total_day_calls', 'total_day_charge',
         'total_eve_calls', 'total_eve_charge',
         'total_night_calls', 'total_night_charge',
         'total_intl_calls', 'total_intl_charge',
         'number_customer_service_calls']

normal1 = ['total_day_charge', 'total_eve_charge',
           'total_night_calls', 'total_night_charge']
non_normal1 = ['account_length', 'number_vmail_messages',
               'total_day_calls', 'total_eve_calls',
               'total_intl_calls',
               'total_intl_charge', 'number_customer_service_calls']

cats2 = ['area_code', 'international_plan', 'voice_mail_plan']
```

✓ 0.0s

Dilakukan pengelompokkan untuk mempermudah proses data preprocessing

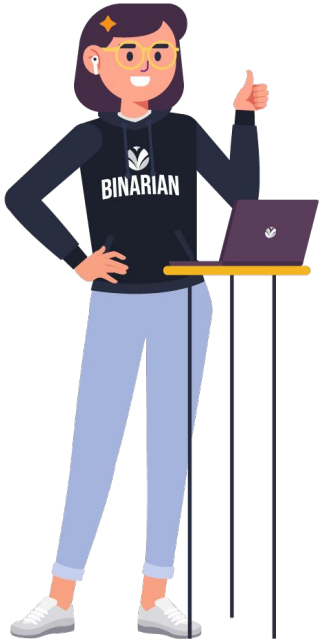
Data Preprocessing

```
# standar scaler
scaler = StandardScaler()

# transform data non normal
df_fe[non_normal1] = scaler.fit_transform(df_fe[non_normal1])

# minmax scaler
minmax = MinMaxScaler()

# transform data normal
df_fe[normal1] = minmax.fit_transform(df_fe[normal1])

# label encoder
label = LabelEncoder()

# label encoder untuk kolom churn
df_fe['churn'] = label.fit_transform(df_fe['churn'])
```

✓ 0.0s

- **Standard scaler** dilakukan pada data dengan **distribusi tidak normal.**
- **Minmax scaller** dilakukan pada **data distribusi normal.**
- **Label encoder** untuk **kolom target ‘churn’.**

Data Preprocessing




```
# one hot encoder
ohe = OneHotEncoder()

# transform data kategorikal
ohe.fit(df_fe[cats2])

# transform data kategorikal
df_fe_ohe = ohe.transform(df_fe[cats2]).toarray()
df_fe_ohe = pd.DataFrame(df_fe_ohe, columns=ohe.get_feature_names_out(cats2))

# drop kolom kategorikal
df_fe = df_fe.drop(columns=cats2)
```

- **Kolom categorical** akan dilakukan **One hot Encoding.**
- **Menghapus kolom yang tidak diperlukan.**

Modeling & Evaluation

Pemilihan Model

Logistic Regression

Decision Tree

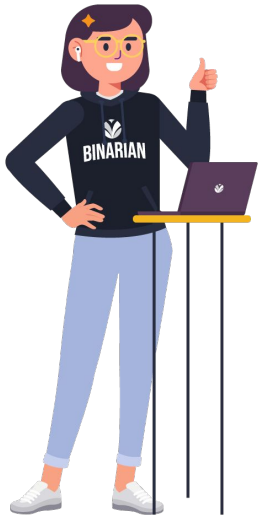
KNN

SVM

Modeling & Evaluation

Pemilihan Model

Logistic Regression



Decision Tree

KNN

SVM

Modeling & Evaluation

Evaluasi model

```
# akurasi
print('Akurasi Logistic Regression: ', accuracy_score(y_test, y_pred_logreg))
print('Akurasi KNN: ', accuracy_score(y_test, y_pred_knn))
print('Akurasi Decision Tree: ', accuracy_score(y_test, y_pred_tree))
print('Akurasi SVM: ', accuracy_score(y_test, y_pred_svm))
```

✓ 0.0s

```
Akurasi Logistic Regression:  0.7994524298425736
Akurasi KNN:  0.8459958932238193
Akurasi Decision Tree:  0.9041752224503764
Akurasi SVM:  0.8829568788501027
```

Masing-masing model akan dilatih, kemudian dievaluasi masing-masing kinerja model.

Modeling & Evaluation

Evaluasi model

```
Classification Report KNN:
              precision    recall  f1-score   support

           0       0.96      0.73      0.83       758
           1       0.77      0.97      0.86       703

    accuracy                           0.85      1461
   macro avg       0.87      0.85      0.84      1461
weighted avg       0.87      0.85      0.84      1461

Classification Report Decision Tree:
              precision    recall  f1-score   support

           0       0.92      0.89      0.91       758
           1       0.89      0.92      0.90       703

    accuracy                           0.90      1461
   macro avg       0.90      0.90      0.90      1461
weighted avg       0.90      0.90      0.90      1461

Classification Report SVM:
              precision    recall  f1-score   support

           0       0.91      0.86      0.88       758
           1       0.86      0.91      0.88       703

    accuracy                           0.88      1461
   macro avg       0.88      0.88      0.88      1461
weighted avg       0.88      0.88      0.88      1461
```

```python
# Evaluasi logistic regression
from sklearn.metrics import mean_absolute_error, mean_squared_error

print('Mean Absolute Error Logistic Regression: {:.2f}'.format(mean_absolute_error(y_test, y_pred_logreg)))
print('Mean Squared Error Logistic Regression: {:.2f}'.format(mean_squared_error(y_test, y_pred_logreg)))
print('Root Mean Squared Error Logistic Regression: {:.2f}'.format(np.sqrt(mean_squared_error(y_test, y_pred_logreg))))
```

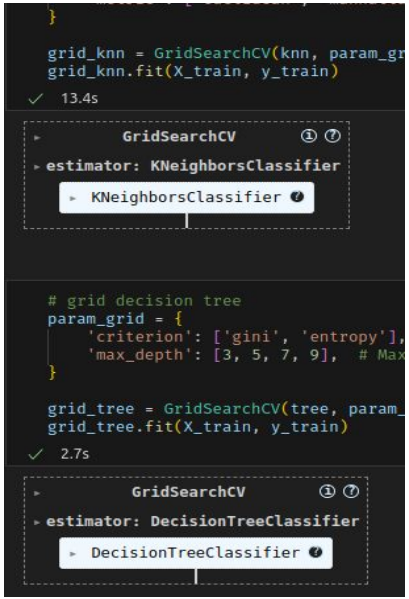
✓ 0.0s

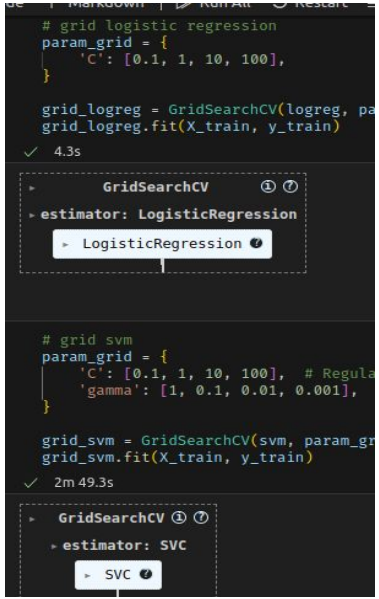
```
Mean Absolute Error Logistic Regression: 0.20
Mean Squared Error Logistic Regression: 0.20
Root Mean Squared Error Logistic Regression: 0.45
```

Hasil report dari masing-masing model.

Modeling & Evaluation

Tuning Model





Keempat model akan dilakukan tuning model menggunakan **GridSearch**.

digunakan untuk menemukan kombinasi hyperparameter terbaik yang menghasilkan performa model optimal.

Modeling & Evaluation

Tuning Model

```
Akurasi KNN Tunning:  0.8863791923340179
Classification Report KNN Tunning:
              precision    recall  f1-score   support

           0       0.98      0.80      0.88       758
           1       0.82      0.98      0.89       703

    accuracy                           0.89      1461
   macro avg       0.90      0.89      0.89      1461
weighted avg       0.90      0.89      0.89      1461
```

```
Akurasi Decision Tree Tunning:  0.9000684462696783
Classification Report Decision Tree Tunning:
              precision    recall  f1-score   support

           0       0.89      0.92      0.91       758
           1       0.91      0.88      0.89       703

    accuracy                           0.90      1461
   macro avg       0.90      0.90      0.90      1461
weighted avg       0.90      0.90      0.90      1461
```

Terdapat peningkatan berdasarkan hasil tuning. **Model dipilih** berdasarkan **hasil report yang paling bagus,** yaitu **SVM**.

```
Akurasi SVM Tunning:  0.9671457905544147
Classification Report SVM Tunning:
              precision    recall  f1-score   support

           0       0.97      0.96      0.97       758
           1       0.96      0.97      0.97       703

    accuracy                           0.97      1461
   macro avg       0.97      0.97      0.97      1461
weighted avg       0.97      0.97      0.97      1461
```

```
Akurasi Logistic Regression Tunning:  0.7994524298425736
Mean Absolute Error Logistic Regression Tunning:  0.2005475701574264
Mean Squared Error Logistic Regression Tunning:  0.2005475701574264
Root Mean Squared Error Logistic Regression Tunning:  0.44782537909036196
```

Testing data baru

Testing predict

```
df_test = pd.read_csv('Data_Test.csv')
df_test.head()
```

✓ 0.0s Python

	id	state	account_length	area_code	international_plan	voice_mail_plan	number_vmail_messages	total_day_minutes	total_day_calls	total_day_charge	total_eve_minutes	total_eve_calls
0	1	KS	128	area_code_415	no	yes	25	265.1	110	45.07	197.4	99
1	2	AL	118	area_code_510	yes	no	0	223.4	98	37.98	220.6	101
2	3	IA	62	area_code_415	no	no	0	120.7	70	20.52	307.2	76
3	4	VT	93	area_code_510	no	no	0	190.7	114	32.42	218.2	111
4	5	NE	174	area_code_415	no	no	0	124.3	76	21.13	277.1	112

```
# svm
y_pred_svm = best_svm.predict(df_test)

# simpan hasil prediksi
df_test['churn'] = y_pred_svm
```

✓ 0.3s Python

```
df_test.head(5)
```

✓ 0.0s Python

ber_customer_service_calls	area_code_area_code_408	area_code_area_code_415	area_code_area_code_510	international_plan_no	international_plan_yes	voice_mail_plan_no	voice_mail_plan_yes	churn
-0.497639	0.0	1.0	0.0	1.0	0.0	0.0	1.0	0
-1.281734	0.0	0.0	1.0	0.0	1.0	1.0	0.0	0
1.854646	0.0	1.0	0.0	1.0	0.0	1.0	0.0	0
1.070551	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0
1.070551	0.0	1.0	0.0	1.0	0.0	1.0	0.0	0

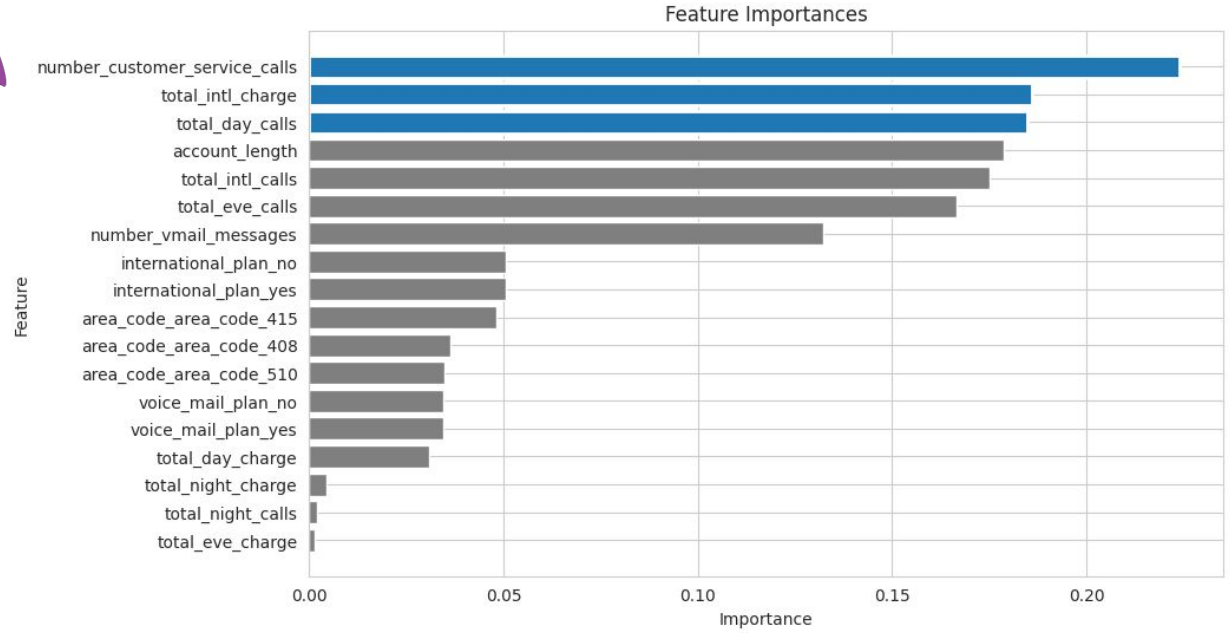
```
# save to csv dengan mengambil
df_test.to_csv('hasil_prediksi.csv', index=False)
```

✓ 0.0s Python

Testing dilakukan menggunakan Model SVM dikarenakan memiliki performa yang bagus dan hasilnya model ini bekerja dengan baik untuk prediksi dan klasifikasi.

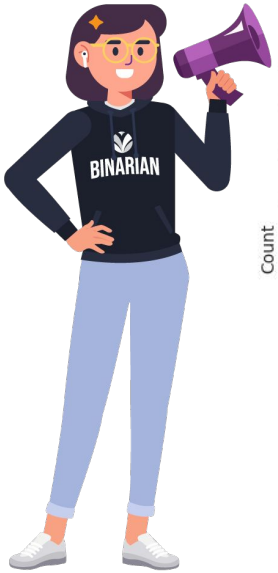
Business Insight

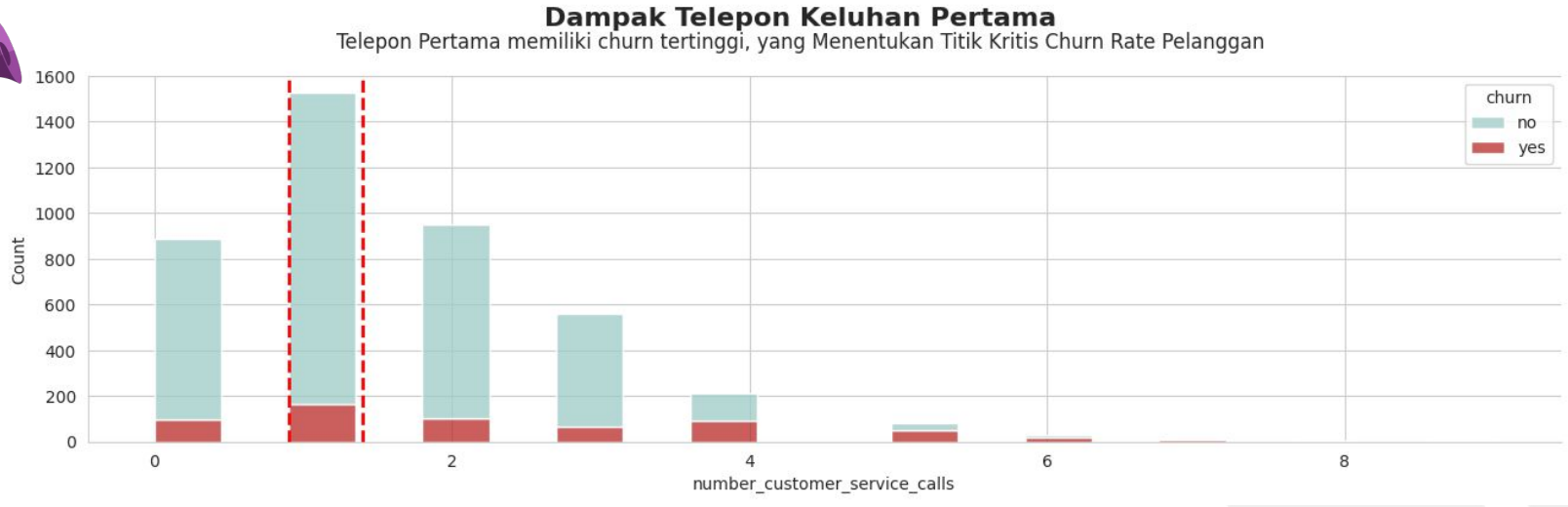
Feature Importance



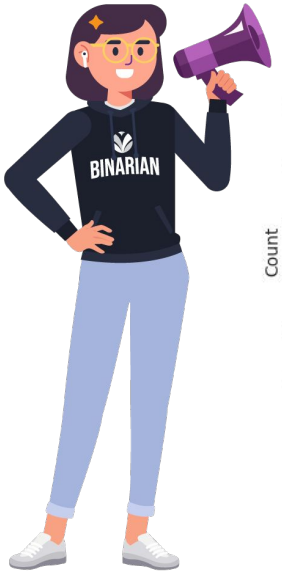
Berdasarkan model yang telah ditentukan dan dilatih, berikut adalah **3 fitur terpenting yang mempengaruhi pelanggan melakukan churn.**

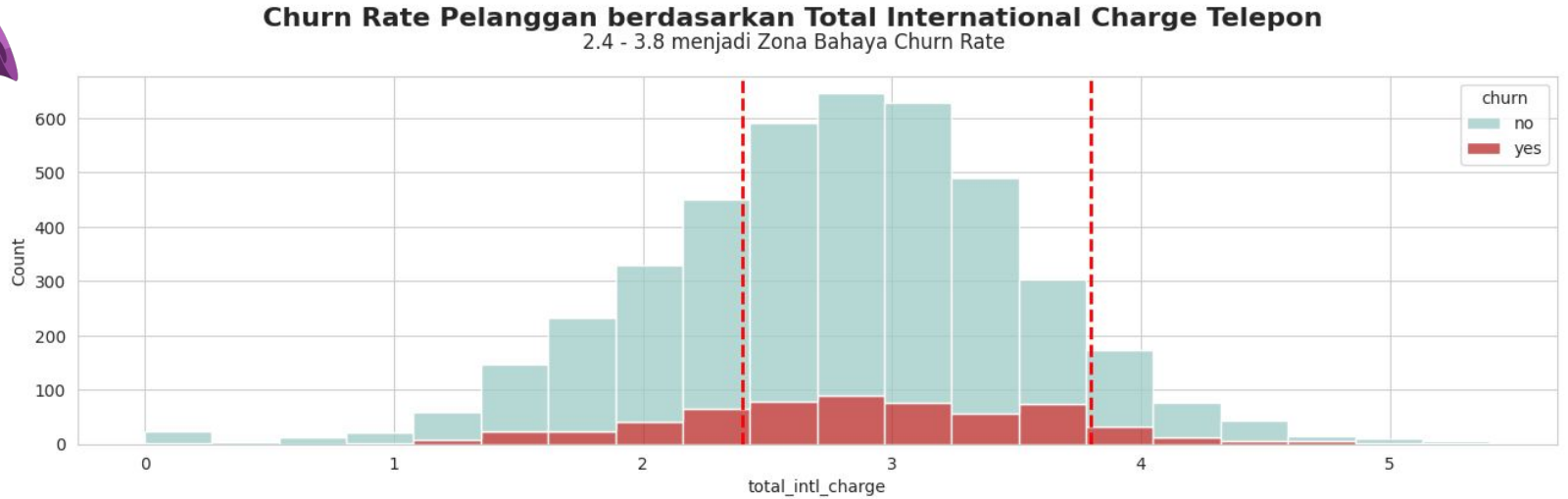
Business Insight



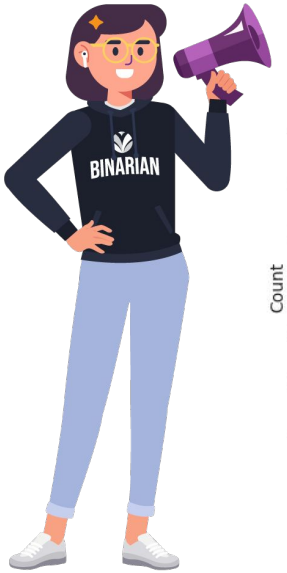


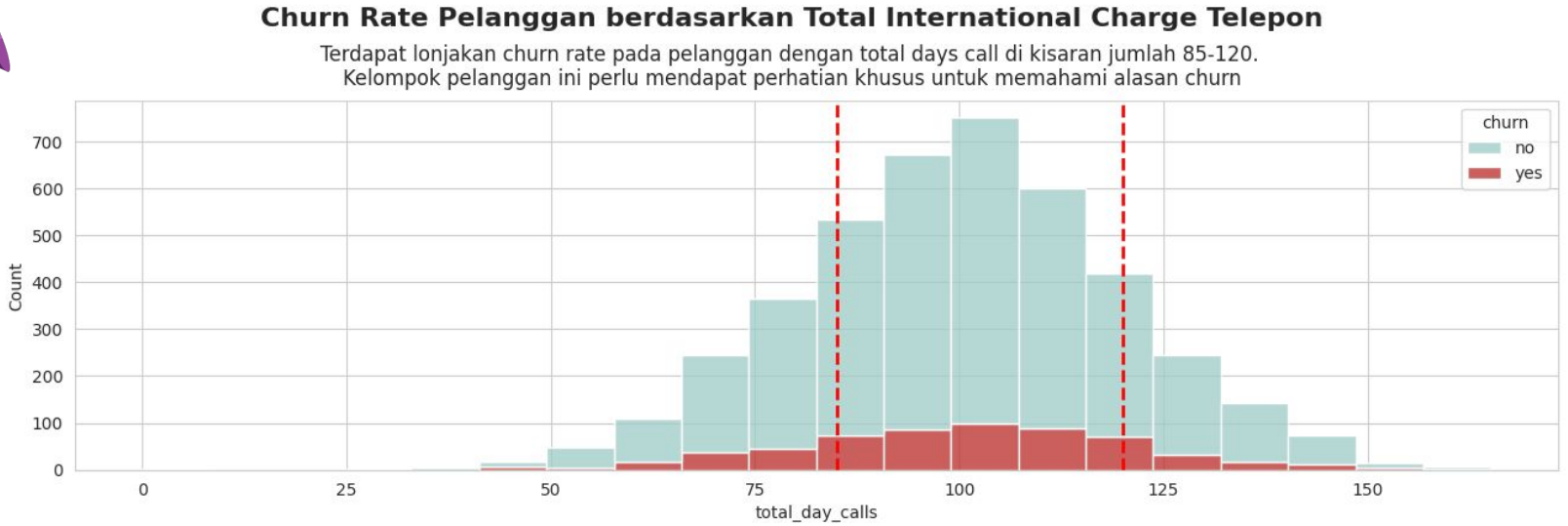
Business Insight





Business Insight





Conclusion & Suggestion

Kesimpulan

- Dari beberapa algoritma yang diuji pada kasus ini, **algoritma SVM** muncul sebagai yang **paling efektif untuk klasifikasi dan prediksi churn**.
- Tahapan preprocessing merupakan aspek yang paling vital dalam proses ini. Dengan memperhatikan pengelolaan data yang tepat sebelum masuk ke dalam model, ini dapat memastikan bahwa model yang dihasilkan memberikan prediksi yang akurat dan berguna.
- Oleh karena itu, kombinasi penggunaan algoritma SVM dengan tahapan preprocessing yang cermat dapat menjadi pendekatan yang efektif dalam menangani masalah churn dalam dataset yang diberikan.

Conclusion & Suggestion

Saran

- Berdasarkan hasil analisis, disarankan untuk melanjutkan **pengembangan** dengan fokus pada **peningkatan kualitas data,** termasuk proses preprocessing yang cermat untuk membersihkan data dan melakukan transformasi jika diperlukan.
- **Pertimbangkan untuk menggabungkan pendekatan yang berbeda** atau menggunakan ensemble learning guna meningkatkan kemampuan model dalam menangani kompleksitas data churn.
- Evaluasi yang cermat terhadap model yang dikembangkan sangat penting, termasuk pengujian performa terhadap data validasi atau data uji yang tidak pernah dilihat sebelumnya, serta perhatikan faktor-faktor bisnis yang mungkin memengaruhi churn.
- Terus memantau performa model secara berkala dan memperbarui model sesuai kebutuhan akan membantu memastikan relevansi dan efektivitasnya dalam mengatasi masalah churn.
- Pertimbangkan untuk menambah kolom baru berdasarkan data yang sudah ada, seperti total menit keseluruhan.
- Menambahkan data baru seperti umur dan pekerjaan, mungkin saja akan memberikan insight baru.

Report Pembagian Tugas Challenge 01

Nama	Tasklist/Deliverable
Taufiq Qurohman Ruki	SQL soal 5-6, dashboard, deck presentasi challenge 2
Aleisya Zahari Salam	SQL soal 1-4, script SQL (GoogleDocs), deck presentasi challenge 1

Report Pembagian Tugas Challenge 02

Nama	Tasklist/Deliverable
Taufiq Qurohman Ruki	Latar belakang,tujuan, preprocessing (Encode, normlisasi,dan smote), evaluasi model, testing dengan dataset baru, dan bikin ppt.
Aleisya Zahari Salam	Proses EDA ,Fitur selection, Preprocessing (Pengelompokkan data), modelling, tunning data, feature importance & business insight, ppt

Note: Pengerjaan task ini kami berdua setiap orangnya mencoba secara bersama-sama semua tahapan, kemudian diakhir menggabungkan atau saling melengkapi dalam tiap tahapannya.

Report Pembagian Tugas Final Presentation

Nama	Tasklist/Deliverable
Taufiq Qurohman Ruki	PPT Challenge chapter 01
Aleisya Zahari Salam	PPT Challenge chapter 02

BINAR