

```
from google.colab import files
uploaded = files.upload()
```

יש לבחור קבצים TravelInsur...ediction.csv

**TravelInsurancePrediction.csv**(text/csv) - 115409 bytes, last modified: 29.12.2025 - 100% done  
Saving TravelInsurancePrediction.csv to TravelInsurancePrediction (1).csv

```
import pandas as pd
df = pd.read_csv('TravelInsurancePrediction.csv')
df.head()
```

	Unnamed: 0	Age	Employment Type	GraduateOrNot	AnnualIncome	FamilyMembers	ChronicDiseases	FrequentFlyer	EverTravelledAbroad
0	0	31	Government Sector	Yes	400000	6	1	No	N
1	1	31	Private Sector/Self Employed	Yes	1250000	7	0	No	N
			Private						

Next steps: [Generate code with df](#) [New interactive sheet](#)

## ✓ חלק 1 – בחירת דאטה וטעינה

בדאטה זה כל שורה מייצגת אדם פוטנציאלי שנבדק האם רכש ביטוח נסיעות. הנתונים כוללים מאפיינים דמוגרפיים והתנהגותיים כגון גיל, סוג תעסוקה, הכנסה שנתית, מספר בני משפחה, מצב בריאותי והרגלי טיסה.

**TravelInsurance:** הוא העמודה (Target) המשתנה המנובא:

- ערך 1 מציין שהאדם רכש ביטוח נסיעות
- ערך 0 מציין שהאדם לא רכש ביטוח נסיעות

ומשמשות לחיזוי ההחלטה האם לרכוש ביטוח Features-שאר העמודות משמשות כ

## ✓ שאלות תיאורטיות – חלק 1

### מהי למידה מונחית (Supervised Learning)?

המטרה (Target) וגם תווית ידועה מראש (Features) למידה מונחית היא סוג של למידת מכונה שבה המודל מאומן על דאטה שמכיל גם נתוני קלט. היא ללמוד קשר בין הנתונים לבין התווית כדי לבצע חיזוי על נתונים חדשים.

### מדוע נדרש דאטה עם תוויות?

התוויות מאפשרות למודל לדעת מהי התשובה הנכונה עבור כל דוגמה בדאטה. ללא תוויות, המודל לא יכול למדוד שגיאה או ללמוד כיצד לשפר את עצמו.

### Feature ל-Target מה ההבדל בין

הוא המשתנה שאותו מנסים לחזות, Target הוא משתנה המספק מידע שעשוי להשפיע על החיזוי (כגון גיל או הכנסה שנתית), ואילו Feature במקרה זה האם האדם רכש ביטוח נסיעות.

### מדוע חשוב להבין את ההקשר של הדאטה לפני אימון מודל

הבנת ההקשר מאפשרת לבחור עיבוד נתונים מתאים, לזהות משתנים לא רלוונטיים או בעייתיים, ולפרש את תוצאות המודל בצורה נכונה.

Double-click (or enter) to edit

(EDA) חלק 2 – ניתוח נתונים חקרני

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1987 entries, 0 to 1986
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Unnamed: 0            1987 non-null   int64
1   Age                   1987 non-null   int64
2   Employment Type       1987 non-null   object
3   GraduateOrNot         1987 non-null   object
```

```

4 AnnualIncome      1987 non-null int64
5 FamilyMembers     1987 non-null int64
6 ChronicDiseases    1987 non-null int64
7 FrequentFlyer      1987 non-null object
8 EverTravelledAbroad 1987 non-null object
9 TravelInsurance    1987 non-null int64
dtypes: int64(6), object(4)
memory usage: 155.4+ KB

```

## ניתוח מבנה הדאטה

בדקתי את מבנה הדאטה, סוגי המשתנים והאם קיימים ערכים חסרים. ניתן לראות כי הדאטה כולל שילוב של משתנים `df.info()` באמצעות הפקודה נומריים ומשתנים קטגוריאליים, דבר שידרוש טיפול שונה בכל סוג.

```
df.describe(include='all')
```

	Unnamed: 0	Age	Employment Type	GraduateOrNot	AnnualIncome	FamilyMembers	ChronicDiseases	FrequentFlyer	EverTravelledAbroad
<b>count</b>	1987.000000	1987.000000	1987	1987	1.987000e+03	1987.000000	1987.000000	1987	1987
<b>unique</b>	NaN	NaN	2	2	NaN	NaN	NaN	2	2
<b>top</b>	NaN	NaN	Private Sector/Self Employed	Yes	NaN	NaN	NaN	No	No
<b>freq</b>	NaN	NaN	1417	1692	NaN	NaN	NaN	1570	1570
<b>mean</b>	993.000000	29.650226	NaN	NaN	9.327630e+05	4.752894	0.277806	NaN	NaN
<b>std</b>	573.741812	2.913308	NaN	NaN	3.768557e+05	1.609650	0.448030	NaN	NaN
<b>min</b>	0.000000	25.000000	NaN	NaN	3.000000e+05	2.000000	0.000000	NaN	NaN
<b>25%</b>	496.500000	28.000000	NaN	NaN	6.000000e+05	4.000000	0.000000	NaN	NaN
<b>50%</b>	993.000000	29.000000	NaN	NaN	9.000000e+05	5.000000	0.000000	NaN	NaN
<b>75%</b>	1489.500000	32.000000	NaN	NaN	1.250000e+06	6.000000	1.000000	NaN	NaN

## ניתוח סטטיסטי ראשוני

מאפשרת לקבל תמונה כללית על הנתונים, כולל ערכים ממוצעים, טווחים וערכים שכיחים. שלב זה מסייע בזיהוי חריגות `df.describe()` הפקודה והבנת התפלגות המשתנים בדאטה.

```
df['TravelInsurance'].value_counts()
```

	count
TravelInsurance	
0	1277
1	710

**dtype:** int64

```
df['TravelInsurance'].value_counts(normalize=True)
```

	proportion
TravelInsurance	
0	0.642677
1	0.357323

**dtype:** float64

## בדיקת איזון המחלקות

בדיקה זו חשובה כדי להבין האם קיימת בעיית חוסר איזון בין מחלקות, שעלולה להשפיע על `TravelInsurance` בדקתי את התפלגות המשתנה המנובא על ביצועי המודל והערכתו.

## EDA – שאלות תיאורטיות

### הוא שלב קריטי לפני אימון מודל EDA מדוע

מאפשר להבין את מבנה הדאטה, לזהות בעיות כמו ערכים חסרים, חוסר איזון בין מחלקות או משתנים חריגים, וכך לבחור תהליך עיבוד מתאים EDA לפני אימון המודל.

### EDA אילו בעיות עלולות להיווצר אם מדלגים על

או פרשנות לא נכונה של הביצועים Overfitting, עלול לגרום לאימון מודל על דאטה בעייתי, מה שיוביל לתוצאות שגויות EDA דלג על

### ?מהי בעיית חוסר איזון בין מחלקות ולמה היא חשובה

חוסר איזון מתרחש כאשר מחלקה אחת מופיעה הרבה יותר מאחרות. במצב כזה המודל עלול להעדיף את המחלקה הדומיננטית ולספק תחזיות לא אמינות.

(Missing Values) חלק 3 – ניתוח ערכים חסרים

```
df.isnull().sum()
```

	0
Unnamed: 0	0
Age	0
Employment Type	0
GraduateOrNot	0
AnnualIncome	0
FamilyMembers	0
ChronicDiseases	0
FrequentFlyer	0
EverTravelledAbroad	0
TravellInsurance	0

dtype: int64

## בדיקת ערכים חסרים בדאטה

בשלב זה בדקתי האם קיימים ערכים חסרים בדאטה ובאילו עמודות הם מופיעים. זיהוי ערכים חסרים הוא שלב חשוב לפני אימון מודל, מאחר ורוב מודלי למידת המכונה אינם יודעים להתמודד עם ערכים חסרים.

## אסטרטגיית טיפול בערכים חסרים

במשתנים נומריים אעדיף להשלים ערכים חסרים באמצעות החציון, מאחר והוא פחות מושפע מערכים חריגים.

כדי לשמר את התפלגות הדאטה, (Most Frequent) במשתנים קטגוריאליים אעדיף להשלים באמצעות הערך השכיח.

בחירה זו מאפשרת לשמור על כמות הנתונים ולמנוע איבוד מידע שעלול להיגרם ממחיקת שורות.

## שאלות תיאורטיות – ערכים חסרים

### מדוע דאטה אמיתי מכיל ערכים חסרים

ערכים חסרים יכולים להיווצר עקב טעויות הזנה, נתונים שלא נאספו, או חוסר מענה של משתמשים. בדאטה מהעולם האמיתי זו תופעה נפוצה.

### מהי Imputation?

היא טכניקה להשלמת ערכים חסרים באמצעות ערכים מחושבים כגון ממוצע, חציון או ערך שכיח Imputation.

### מדוע מחיקת שורות היא לרוב רעיון גרוע

מחיקת שורות גורמת לאיבוד מידע ועלולה להקטין משמעותית את גודל הדאטה או לפגוע בהתפלגות הנתונים.

### ?מתי נוריד עמודה ומתי נשלים ערכים חסרים

אם עמודה מכילה אחוז גבוה מאוד של ערכים חסרים או שאינה רלוונטית לבעיה, ניתן להסיר אותה. במקרים אחרים עדיף להשלים את הערכים מתאימות Imputation באמצעות שיטות.

מה הרעיון בחלק הזה? (בשפה פשוטה) Features I-Target חלק 4 – הפרדת

אנחנו אומרים למודל

⌘ (Target) זה מה שאתה צריך לנחש

✔ (Features) וזה המידע שמותר לך להשתמש בו

```
df = df.drop(columns=['Unnamed: 0'])
```

```
X = df.drop('TravelInsurance', axis=1)
y = df['TravelInsurance']
```

## Features I-Target חלק 4 – הפרדת

בשלב זה הפרדתי את הדאטה לשני רכיבים מרכזיים:

- Features (X) – כל המשתנים המשמשים כקלט למודל
- Target (y) – המשתנה שאותו המודל צריך לחזות

ואילו שאר העמודות משמשות כמידע מסביר לצורך החיזוי. הפרדה זו היא שלב קריטי לפני TravelInsurance בפרויקט זה המשתנה המנובא הוא אימון מודל למידת מכונה.

## שאלות תיאורטיות – חלק 4

### Features I-Target מדוע חובה להפריד בין

במהלך האימון (Data Leakage) הפרדה זו מונעת מצב שבו המודל נחשף לערך שאותו הוא אמור לחזות, וכך נמנעת זליגת מידע.

### Machine Learning מה המושגים של "למידה" במודל

בצורה מיטבית Target-לבין ה Features-למידה היא תהליך שבו המודל מתאים את הפרמטרים הפנימיים שלו כך שיוכל למפות בין ה

### מדוע אילו עמודות הוסרו ואילו הושארו?

מאחר והן מכילות מידע רלוונטי Features-מאחר והיא המשתנה המנובא. שאר העמודות נשארו כ Target-הופרדה כ TravelInsurance העמודה לחיזוי.

### Supervised Learning ל-Unsupervised Learning מה ההבדל בין

אין תוויות והמטרה היא לזהות מבנים Unsupervised Learning-המודל מאומן על דאטה עם תוויות ידועות מראש, ואילו ב Supervised Learning-ב ודפוסים בדאטה.

🔗 שלב 5.1 – חלוקת הדאטה Train / Test חלק 5 – חלוקת

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.2,
    random_state=42,
    stratify=y
)
```

## Train I-Test חלוקת הדאטה ל

המשמש להערכת ביצועי המודל על נתונים (Test) המשמש לאימון המודל, וסט בדיקה (Train) בשלב זה חילקתי את הדאטה לשני חלקים: סט אימון חדשים.

החלוקה בוצעה בצורה סטטיסטית לפי המשתנה המנובא, כדי לשמור על יחס דומה בין המחלקות בשני הסטים.

## Train / Test Split – שאלות תיאורטיות

### מדוע אסור להעריך מודל על דאטה שאיתו הוא אומן

הערכה על דאטה שאיתו המודל אומן תיתן תוצאות אופטימיות מדי ולא תשקף את יכולת ההכללה של המודל על נתונים חדשים.

### מהו Overfitting?

הוא מצב שבו המודל מתאים את עצמו יתר על המידה לדאטה, כולל רעש וחרגות, ולכן מבצע חיזוי גרוע על נתונים חדשים Overfitting.

### מהי Data Leakage?

מתרחש כאשר מידע מסט הבדיקה דלף לתהליך האימון, וגורם להערכת ביצועים לא אמינה Data Leakage.

### מדוע חלוקה סטטיסטית חשובה בבעיות סיווג

חלוקה סטטיסטית שומרת על יחס דומה בין המחלקות ומונעת מצב שבו אחת המחלקות מיוצגת בצורה לא הוגנת בסט הבדיקה.

## 🔗 בניית Pipeline I-ColumnTransformer 6 - חלק 6

```
numeric_features = X.select_dtypes(include=['int64', 'float64']).columns
categorical_features = X.select_dtypes(include=['object']).columns

print("Numeric features:", numeric_features)
print("Categorical features:", categorical_features)
```

```
Numeric features: Index(['Age', 'AnnualIncome', 'FamilyMembers', 'ChronicDiseases'], dtype='object')
Categorical features: Index(['Employment Type', 'GraduateOrNot', 'FrequentFlyer',
                             'EverTravelledAbroad'],
                             dtype='object')
```

```
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler, OneHotEncoder
```

```
numeric_pipeline = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())
])

categorical_pipeline = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('encoder', OneHotEncoder(handle_unknown='ignore'))
])
```

```
from sklearn.compose import ColumnTransformer

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_pipeline, numeric_features),
        ('cat', categorical_pipeline, categorical_features)
    ]
)
```

```
X_train_processed = preprocessor.fit_transform(X_train)
X_test_processed = preprocessor.transform(X_test)
```

## לעיבוד נתונים Pipeline בניית

Scaling לעיבוד הנתונים לפני אימון המודלים. המשתנים הנומריים עברו השלמת ערכים חסרים באמצעות חציון ולאחר מכן Pipeline בשלב זה נבנה One-Hot Encoding המשתנים הקטגוריאליים עברו השלמת ערכים חסרים באמצעות הערך השכיח וקידוד באמצעות. בוצע רק על סט האימון כדי למנוע זליגת מידע fit-כאשר תהליך ה, ColumnTransformer כל שלבי העיבוד בוצעו באמצעות.

## ❖ Pipelines I-Preprocessing – שאלות תיאורטיות

### מדוע מודלים דורשים קלט נומרי בלבד?

מרבית מודלי למידת המכונה מבצעים חישובים מתמטיים, ולכן אינם יודעים לעבוד ישירות עם טקסט או קטגוריות.

### מדוע Scaling מהמודלים רק לחלק מהמודלים?

מודלים המבוססים על מרחקים או גרדיאנטים רגישים לסקאלת הנתונים, בעוד שמודלים מבוססי עצים פחות מושפעים מכך.

### רק על נתוני האימון fit מדוע מבצעים?

על כל הדאטה עלול לגרום לזליגת מידע מסט הבדיקה לתהליך האימון ולפגוע באמינות הערכת המודל fit ביצוע.

🔗 1 Decision Tree 2 KNN 3 Logistic Regression חלק 7 – אימון מודלים והשוואה אנחנו נאמן: 1

```
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
```

```
log_reg = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('model', LogisticRegression(max_iter=1000))
])
```

```

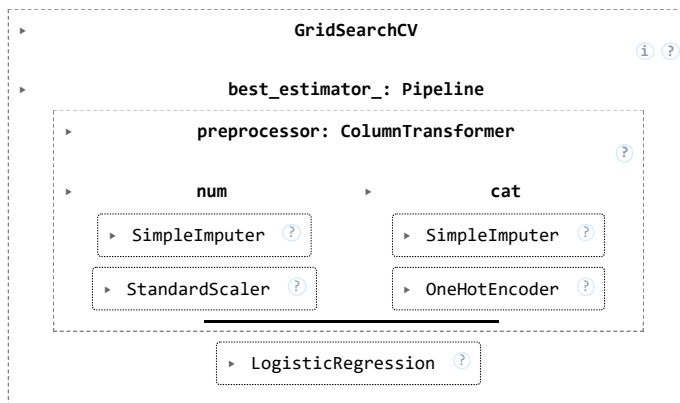
])

log_reg_params = {
    'model__C': [0.01, 0.1, 1, 10]
}

log_reg_grid = GridSearchCV(
    log_reg,
    log_reg_params,
    cv=5,
    scoring='accuracy'
)

log_reg_grid.fit(X_train, y_train)

```



```

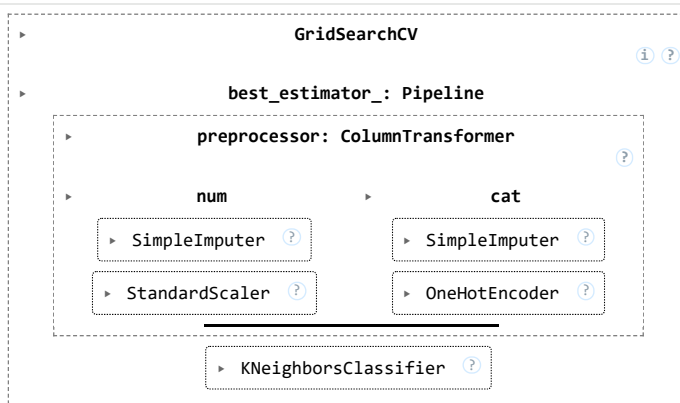
knn = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('model', KNeighborsClassifier())
])

knn_params = {
    'model__n_neighbors': [3, 5, 7, 9]
}

knn_grid = GridSearchCV(
    knn,
    knn_params,
    cv=5,
    scoring='accuracy'
)

knn_grid.fit(X_train, y_train)

```



```

tree = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('model', DecisionTreeClassifier(random_state=42))
])

tree_params = {
    'model__max_depth': [None, 5, 10, 20],
    'model__min_samples_split': [2, 5, 10]
}

tree_grid = GridSearchCV(
    tree,

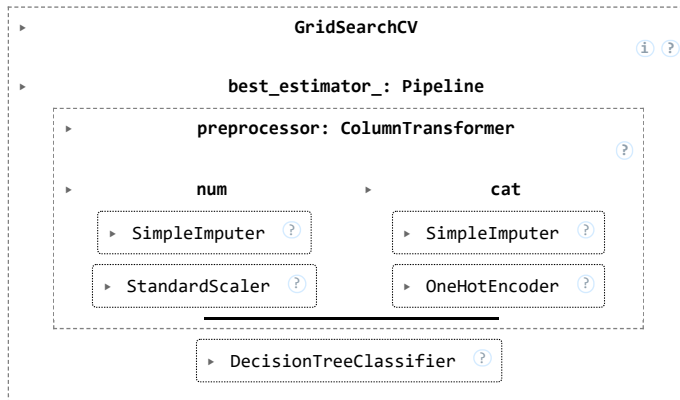
```

```

tree_params,
cv=5,
scoring='accuracy'
)

tree_grid.fit(X_train, y_train)

```



```

import pandas as pd

results = pd.DataFrame({
    'Model': ['Logistic Regression', 'KNN', 'Decision Tree'],
    'Best CV Accuracy': [
        log_reg_grid.best_score_,
        knn_grid.best_score_,
        tree_grid.best_score_
    ]
})

results

```

	Model	Best CV Accuracy	
0	Logistic Regression	0.782874	
1	KNN	0.802387	
2	Decision Tree	0.835740	

Next steps: [Generate code with results](#) [New interactive sheet](#)

## אימון מודלים והשוואתם

לכל מודל נבחרו היפר-פרמטרים GridSearchCV: Logistic Regression, KNN ו-Decision Tree. בשלב זה אומנו שלושה מודלים שונים באמצעות cv=5 עם Cross-Validation מתאימים, והאימון בוצע באמצעות

כדי לבחור את המודל בעל הביצועים הטובים ביותר, Cross-Validation-השוואת המודלים בוצעה על סמך דיוק ממוצע ב

### שאלות תיאורטיות – אימון מודלים

#### מה פירוש "לאמן מודל"?

אימון מודל הוא תהליך שבו המודל לומד להתאים את הפרמטרים שלו כך שיבצע חיזוי מדויק ככל האפשר על סמך הדאטה

#### מהו Hyperparameter?



KNN-הוא פרמטר שנקבע מראש ואינו נלמד ישירות מהדאטה, כגון עומק עץ או מספר שכנים ב Hyperparameter

#### מדוע מודלים שונים נותנים תוצאות שונות על אותו דאטה?

כל מודל מבוסס על הנחות ומתמטיקה שונות, ולכן כל אחד לומד דפוסים אחרים בדאטה ומציג ביצועים שונים

חלק 8 – בחירת המודל הטוב ביותר 🏆

results

	Model	Best CV Accuracy	
0	Logistic Regression	0.782874	
1	KNN	0.802387	
2	Decision Tree	0.835740	

Next steps: [Generate code with results](#) [New interactive sheet](#)

```
best_model_name = results.loc[results['Best CV Accuracy'].idxmax(), 'Model']
best_model_name
```

```
'Decision Tree'
```

```
best_model = max(
    [log_reg_grid, knn_grid, tree_grid],
    key=lambda x: x.best_score_
)
```

## ❏ בחירת המודל הטוב ביותר

נבחר המודל שהציג את הדיוק הממוצע הגבוה ביותר. בחירה זו מבוססת על ביצועים, Cross-Validation, לאחר השוואת ביצועי המודלים באמצעות על סט אימון בלבד, ללא שימוש בסט הבדיקה, כדי למנוע הטיה בתהליך הבחירה.

 Test Data חלק 9 – הערכת המודל על

```
y_pred = best_model.predict(X_test)
```

```
from sklearn.metrics import accuracy_score

accuracy = accuracy_score(y_test, y_pred)
accuracy
```

```
0.8140703517587939
```

```
from sklearn.metrics import confusion_matrix

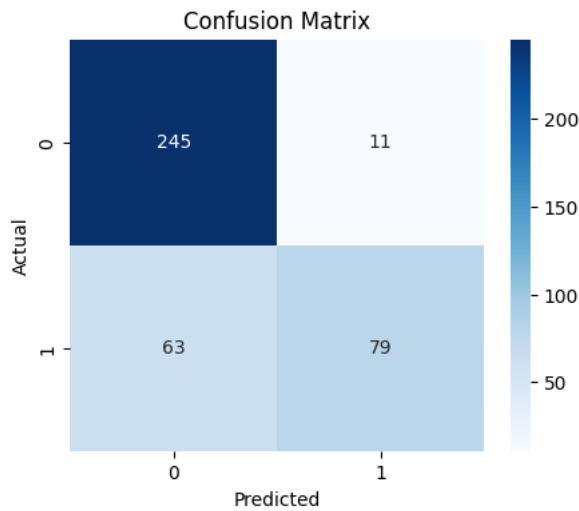
cm = confusion_matrix(y_test, y_pred)
cm
```

```
array([[245, 11],
       [ 63, 79]])
```

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(5, 4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```





```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.80	0.96	0.87	256
1	0.88	0.56	0.68	142
accuracy			0.81	398
macro avg	0.84	0.76	0.77	398
weighted avg	0.82	0.81	0.80	398

## הערכת ביצועי המודל על סט הבדיקה

Accuracy, Confusion Matrix בשלב זה הערכתי את המודל הנבחר על סט הבדיקה, אשר לא שימש בתהליך האימון. בוצעה הערכה באמצעות מדדי Precision, Recall ומדדי F1.

שילוב מדדים אלו מאפשר להבין לא רק את רמת הדיוק הכללית, אלא גם את סוגי השגיאות שהמודל מבצע.

### שאלות תיאורטיות – הערכת מודל

#### בלבד אינו מספיק Accuracy מדוע

אינו מבדיל בין סוגי שגיאות ועלול להטעות במיוחד כאשר קיימת בעיית חוסר איזון בין מחלקות Accuracy.

#### False Positive ו-False Negative מה הם

הוא מצב שבו המודל חזה שלילי אך הערך האמיתי False Negative הוא מצב שבו המודל חזה חיובי אך הערך האמיתי שלילי, ואילו False Positive חיובי.

#### Precision-חשוב יותר מ Recall מתי

Recall חשוב יותר כאשר החמצת מקרים חיוביים מהווה בעיה חמורה יותר מאשר זיהוי שגוי, לדוגמה במצבים רפואיים או זיהוי סיכונים.

חלק 10 – חיזוי על נתונים חדשים

```
new_data = pd.DataFrame([
    {
        'Age': 35,
        'Employment Type': 'Private Sector/Self Employed',
        'GraduateOrNot': 'Yes',
        'AnnualIncome': 60000,
        'FamilyMembers': 3,
        'ChronicDiseases': 0,
        'FrequentFlyer': 'Yes',
        'EverTravelledAbroad': 'Yes'
    },
    {
        'Age': 50,
        'Employment Type': 'Government Sector',
        'GraduateOrNot': 'No',
        'AnnualIncome': 40000,
        'FamilyMembers': 4,
        'ChronicDiseases': 1,
        'FrequentFlyer': 'No',
    }
])
```

```

    'EverTravelledAbroad': 'No'
},
{
    'Age': 28,
    'Employment Type': 'Private Sector/Self Employed',
    'GraduateOrNot': 'Yes',
    'AnnualIncome': 800000,
    'FamilyMembers': 1,
    'ChronicDiseases': 0,
    'FrequentFlyer': 'Yes',
    'EverTravelledAbroad': 'Yes'
}
])

```

```

new_predictions = best_model.predict(new_data)
new_predictions

```

```
array([1, 0, 0])
```

```

new_probabilities = best_model.predict_proba(new_data)
new_probabilities

```

```

array([[0.42857143, 0.57142857],
       [0.8       , 0.2       ],
       [0.87121212, 0.12878788]])

```

## חיזוי על נתונים חדשים

בשלב זה יצרתי דוגמאות חדשות המדמות לקוחות פוטנציאליים, וביצעתי חיזוי באמצעות המודל הנבחר. המודל חזה הן את המחלקה (רכישה / אי רכישה) והן את ההסתברות לכל תחזית.

שימוש בהסתברויות מאפשר קבלת החלטות מושכלת יותר, ולא רק חיזוי בינארי.

## ✓ שאלות תיאורטיות – חיזוי על נתונים חדשים

### Preprocessor מדוע חייבים להשתמש באותו

כדי להבטיח שהנתונים החדשים יעברו את אותו תהליך עיבוד כמו נתוני האימון, וכך לשמור על עקביות ודיוק בחיזוי.

### מחדש על נתונים חדשים fit מדוע אסור לבצע

מחדש ישנה את פרמטרי העיבוד והמודל, ויפגע באמינות החיזוי וביכולת ההכללה של fit ביצוע.

### מה מייצגת הסתברות חיזוי

הסתברות חיזוי מייצגת את רמת הביטחון של המודל בהשתייכות הדוגמה למחלקה מסוימת.

### כיצד משתמשים בביטחון החיזוי בקבלת החלטות

ניתן לקבוע סף החלטה, או להשתמש בהסתברויות לצורך תעדוף לקוחות, ניהול סיכונים וקבלת החלטות עסקיות.

יאללה, סיום 🎉 חלק 11 – רפלקציה מסכמת

## רפלקציה מסכמת

### מה היה החלק המאתגר ביותר בפרויקט

מלא שמטפל גם בערכים חסרים, גם בקידוד משתנים קטגוריאליים וגם במניעת זליגת מידע בין סט Pipeline החלק המאתגר ביותר היה בניית האימון לסט הבדיקה.

### Machine Learning אילו טעויות נפוצות קיימות בפרויקטי

והערכת המודל על נתונים שבהם, Train/Test-על כל הדאטה לפני החלוקה ל preprocessing ביצוע, EDA-טעויות נפוצות כוללות דילוג על שלב ה. הוא כבר אומן.

### מדוע איכות הדאטה חשובה יותר מבחירת מודל

דאטה איכותי מאפשר לכל מודל ללמוד דפוסים אמיתיים, בעוד שדאטה בעייתי יוביל לתוצאות גרועות גם אם נשתמש במודל מתקדם.