```
import pandas as pd

file_path = "/content/transfusion.csv"

df = pd.read_csv(file_path)
print(df.head())
```

```
   Recency (months)  Frequency (times)  Monetary (c.c. blood)  Time (months)  \
0                 2                 50                  12500             98
1                 0                 13                   3250             28
2                 1                 16                   4000             35
3                 2                 20                   5000             45
4                 1                 24                   6000             77

   whether he/she donated blood in March 2007
0                                           1
1                                           1
2                                           1
3                                           1
4                                           0
```

```
import pandas as pd

file_path = "/content/transfusion.csv"
df = pd.read_csv(file_path)

print(df.info())
print(df.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 748 entries, 0 to 747
Data columns (total 5 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Recency (months)                        748 non-null    int64
 1   Frequency (times)                       748 non-null    int64
 2   Monetary (c.c. blood)                   748 non-null    int64
 3   Time (months)                           748 non-null    int64
```

```
 4   whether he/she donated blood in March 2007  748 non-null    int64
dtypes: int64(5)
memory usage: 29.3 KB
None
   Recency (months)  Frequency (times)  Monetary (c.c. blood)  Time (months)  \
0                 2                 50                  12500             98
1                 0                 13                   3250             28
2                 1                 16                   4000             35
3                 2                 20                   5000             45
4                 1                 24                   6000             77

   whether he/she donated blood in March 2007
0                                           1
1                                           1
2                                           1
3                                           1
4                                           0
```

```
import pandas as pd

file_path = "/content/transfusion.csv"
df = pd.read_csv(file_path)

print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 748 entries, 0 to 747
Data columns (total 5 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Recency (months)                        748 non-null    int64
 1   Frequency (times)                       748 non-null    int64
 2   Monetary (c.c. blood)                   748 non-null    int64
 3   Time (months)                           748 non-null    int64
 4   whether he/she donated blood in March 2007  748 non-null    int64
dtypes: int64(5)
memory usage: 29.3 KB
None
```

```
import pandas as pd

file_path = "/content/transfusion.csv"
df = pd.read_csv(file_path)

df.rename(columns={"whether he/she donated blood in March 2007": "target"}, inplace=True)

print(df.head(2))
```

```
   Recency (months)  Frequency (times)  Monetary (c.c. blood)  Time (months)  \
0                 2                 50                  12500             98
1                 0                 13                   3250             28

   target
0       1
1       1
```

```
target_proportions = df["target"].value_counts(normalize=True).round(3)
print(target_proportions)
```

```
target
0    0.762
1    0.238
Name: proportion, dtype: float64
```

```
!pip install pandas
!pip install scikit-learn

import pandas as pd
from sklearn.model_selection import train_test_split

file_path = "/content/transfusion.csv"


df = pd.read_csv(file_path)
```

```
print(df.info())
print(df.head())


print(df.info())

df.rename(columns={"whether he/she donated blood in March 2007": "target"}, inplace=True)

print(df.head(2))


target_proportions = df["target"].value_counts(normalize=True).round(3)
print(target_proportions)

X = df.drop(columns=["target"])

y = df["target"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, stratify=y, random_state=42
)

print(X_train.head(2))
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.11/dist-packages (1.6.1)
Requirement already satisfied: numpy>=1.19.5 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.26.4)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn) (3.5.0
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 748 entries, 0 to 747
Data columns (total 5 columns):
 #   Column                                  Non-Null Count  Dtype
```

```
---  ------                                        --------------  -----
 0   Recency (months)                              748 non-null    int64
 1   Frequency (times)                             748 non-null    int64
 2   Monetary (c.c. blood)                         748 non-null    int64
 3   Time (months)                                 748 non-null    int64
 4   whether he/she donated blood in March 2007    748 non-null    int64
dtypes: int64(5)
memory usage: 29.3 KB
None
   Recency (months)  Frequency (times)  Monetary (c.c. blood)  Time (months)  \
0                 2                  50                  12500             98
1                 0                  13                   3250             28
2                 1                  16                   4000             35
3                 2                  20                   5000             45
4                 1                  24                   6000             77

   whether he/she donated blood in March 2007
0                                           1
1                                           1
2                                           1
3                                           1
4                                           0
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 748 entries, 0 to 747
Data columns (total 5 columns):
 #   Column                                        Non-Null Count  Dtype
---  ------                                        --------------  -----
 0   Recency (months)                              748 non-null    int64
 1   Frequency (times)                             748 non-null    int64
 2   Monetary (c.c. blood)                         748 non-null    int64
 3   Time (months)                                 748 non-null    int64
 4   whether he/she donated blood in March 2007    748 non-null    int64
dtypes: int64(5)
memory usage: 29.3 KB
None
   Recency (months)  Frequency (times)  Monetary (c.c. blood)  Time (months)  \
0                 2                  50                  12500             98
1                 0                  13                   3250             28

   target
0       1
1       1
target
```

```
!pip install tpot

import pandas as pd
from sklearn.model_selection import train_test_split
from tpot import TPOTClassifier
from sklearn.metrics import roc_auc_score

file_path = "/content/transfusion.csv"
df = pd.read_csv(file_path)

df.rename(columns={"whether he/she donated blood in March 2007": "target"}, inplace=True)

X = df.drop(columns=["target"])
y = df["target"]


X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, stratify=y, random_state=42
)

tpot = TPOTClassifier(generations=5, population_size=20, verbosity=2,
                      scoring='roc_auc', random_state=42, config_dict='TPOT light')


tpot.fit(X_train, y_train)


y_pred = tpot.predict_proba(X_test)[:, 1]

tpot_auc_score = roc_auc_score(y_test, y_pred)

print(f"TPOT AUC Score: {tpot_auc_score:.4f}")

for idx, transform in enumerate(tpot.fitted_pipeline_.steps):
    print(f"Step {idx}: {transform}")
```

```
Requirement already satisfied: tpot in /usr/local/lib/python3.11/dist-packages (0.12.2)
Requirement already satisfied: numpy>=1.16.3 in /usr/local/lib/python3.11/dist-packages (from tpot) (1.26.4)
Requirement already satisfied: scipy>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from tpot) (1.13.1)
Requirement already satisfied: scikit-learn>=1.4.1 in /usr/local/lib/python3.11/dist-packages (from tpot) (1.6.1)
Requirement already satisfied: deap>=1.2 in /usr/local/lib/python3.11/dist-packages (from tpot) (1.4.2)
Requirement already satisfied: update-checker>=0.16 in /usr/local/lib/python3.11/dist-packages (from tpot) (0.18.0)
Requirement already satisfied: tqdm>=4.36.1 in /usr/local/lib/python3.11/dist-packages (from tpot) (4.67.1)
Requirement already satisfied: stopit>=1.1.1 in /usr/local/lib/python3.11/dist-packages (from tpot) (1.1.2)
Requirement already satisfied: pandas>=0.24.2 in /usr/local/lib/python3.11/dist-packages (from tpot) (2.2.2)
Requirement already satisfied: joblib>=0.13.2 in /usr/local/lib/python3.11/dist-packages (from tpot) (1.4.2)
Requirement already satisfied: xgboost>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from tpot) (2.1.4)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.24.2->tp
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.24.2->tpot) (2025.
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.24.2->tpot) (202
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn>=1.4.1-
Requirement already satisfied: requests>=2.3.0 in /usr/local/lib/python3.11/dist-packages (from update-checker>=0.16->tpo
Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/python3.11/dist-packages (from xgboost>=1.1.0->tpot) (2
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas>=
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests>=2.3.0-
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests>=2.3.0->update-chec
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests>=2.3.0->updat
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests>=2.3.0->updat
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
/usr/local/lib/python3.11/dist-packages/sklearn/base.py:1230: FutureWarning: passing a class to None is deprecated and wi
  warnings.warn(
/usr/local/lib/python3.11/dist-packages/sklearn/base.py:1270: FutureWarning: passing a class to None is deprecated and wi
  warnings.warn(
is_regressor
is_classifier
is_regressor
is_classifier
is_classifier
is_classifier
is_classifier
is_classifier
```

```
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_regressor
is_classifier
is_classifier
is_regressor

Generation 1 - Current best internal CV score: 0.7411756911072915

Generation 2 - Current best internal CV score: 0.7411756911072915

Generation 3 - Current best internal CV score: 0.7423330644124078

Generation 4 - Current best internal CV score: 0.7423330644124078

Generation 5 - Current best internal CV score: 0.7423330644124078

Best pipeline: LogisticRegression(RobustScaler(input_matrix), C=25.0, dual=False, penalty=l2)
TPOT AUC Score: 0.7858
Step 0: ('robustscaler', RobustScaler())
Step 1: ('logisticregression', LogisticRegression(C=25.0, random_state=42))
```

```
import pandas as pd

variance = X_train.var().round(3)

print(variance)
```

```
Recency (months)              66.929
Frequency (times)             33.830
Monetary (c.c. blood)    2114363.700
Time (months)                611.147
dtype: float64
```

```
!pip install numpy
import pandas as pd
import numpy as np


X_train_normed = X_train.copy()
X_test_normed = X_test.copy()

col_to_normalize = X_train.var().idxmax()

X_train_normed[col_to_normalize] = np.log1p(X_train_normed[col_to_normalize])
X_test_normed[col_to_normalize] = np.log1p(X_test_normed[col_to_normalize])

print(X_train_normed.var().round(3))

assert X_train_normed.columns.equals(X_test_normed.columns), "Mismatch in columns!"
```

```
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages (1.26.4)
Recency (months)          66.929
Frequency (times)         33.830
Monetary (c.c. blood)      0.835
Time (months)            611.147
dtype: float64
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score

logreg = LogisticRegression()

logreg.fit(X_train, y_train)

y_pred_prob = logreg.predict_proba(X_test)[:, 1]

logreg_auc_score = roc_auc_score(y_test, y_pred_prob)

print("Logistic Regression AUC Score:", logreg_auc_score)
```

```
Logistic Regression AUC Score: 0.7857596948506039
```

```
from operator import itemgetter

model_scores = [
    ("Logistic Regression", 0.85),
    ("Random Forest", 0.92),
    ("Decision Tree", 0.78),
    ("SVM", 0.88)
]

sorted_models = sorted(model_scores, key=itemgetter(1), reverse=True)

print("Models sorted by AUC score:")
for model, score in sorted_models:
    print(f"{model}: {score}")
```

```
Models sorted by AUC score:
Random Forest: 0.92
SVM: 0.88
Logistic Regression: 0.85
Decision Tree: 0.78
```