CLASSIFICATION

Get the data and performing data visualization

```
!wget -q -O -
iqdata.si/uploads/dataset/241/ccpp.zip|
jar -xvf ccpp.zip
mkdir -p ./data/ccpp/
mv ./*.csv ./data/ccpp/
ls -lhtr ./data/ccpp/
```

```
Time zone: Etc/UTC (UTC)
Eval: IO.popen("ln -s -f -T /home/jupyter-iq-data-ccpp/ccpp /home/jupyter-iq-data-ccpp/.ccpp", ...)
mapped: 1024 lines.
mapped: 9568 lines.
mapped: 9568 lines.
inflating: ./data/ccpp/ccpp.csv
total 1.8M
drwxr-xr-x 2 root root 4096 Dec  1 00:00 ./
-rw-r--r-- 1 root root 1.8M Dec  1 00:00 ccpp.csv
```

```
# Import and display the first 5 rows of the data
import pandas as pd, numpy, matplotlib, seaborn, sklearn, scipy
from matplotlib import pyplot as plt
from sklearn import preprocessing, model_selection, metrics, tree
from scipy.stats import zscore, chi2_contingency
df = pd.read_csv('./data/ccpp/ccpp.csv')
df.shape, df.dtypes, df.nunique(), df.isnull().sum(), df.sample(5)
df[df.duplicated()]
```

```
/usr/local/lib/python3.10/dist-packages/numpy/_distutils_gscript.py:1:
DeprecationWarning: distutils Version classes are deprecated. Use packaging.version instead.
((9568, 5),
 AT    float64
 V     float64
 AP    float64
 RH    float64
 PE    float64
 dtype: object,
 AT    2744
 V     2165
 AP    1013
 RH    4811
 PE    4836
 dtype: int64,
 AT    0
 V     0
 AP    0
 RH    0
 PE    0
 dtype: int64,
      AT     V       AP     RH     PE
 9    7.9  39.1  1012.16  81.11  488.62
 1   23.8  58.3  1011.90  96.20  471.77
 2   14.9  41.5  1017.60  75.10  444.37
 3   10.3  43.7  1022.86  74.20  488.56
 4    6.1  44.2  1029.87  69.45  499.26
 Empty DataFrame
 Columns: [AT, V, AP, RH, PE]
 Index: [])
```

```
# Describe the data
df.describe(include='all')
```

	AT	V	AP	RH	PE
count	9568.000000	9568.000000	9568.000000	9568.000000	9568.000000
mean	19.651231	54.305804	1013.259078	73.308978	454.365009
std	7.452162	12.707893	5.938784	14.600275	17.066995
min	1.810000	25.360000	992.890000	25.560000	420.260000
25%	13.510000	41.740000	1009.710000	63.327500	435.675000
50%	20.345000	52.080000	1014.810000	74.975000	451.557500
75%	25.720000	62.377500	1018.175000	84.830000	468.430000
max	37.640000	81.560000	1033.030000	100.160000	495.760000

CORRELATION

Measured Depth, Deviation, Surface Casing Weight, Production Casing Size, and Cumulative Drill Print are missing values.

```
# df.describe()
```

```
# Correlation
df.corr(numeric_only=True)
```

	AT	V	AP	RH	PE
AT	1.000000	0.844107	-0.507549	-0.542535	-0.948128
V	0.844107	1.000000	-0.413502	-0.312187	-0.869780
AP	-0.507549	-0.413502	1.000000	0.099574	0.518429
RH	-0.542535	-0.312187	0.099574	1.000000	0.389794
PE	-0.948128	-0.869780	0.518429	0.389794	1.000000

CORRELATION WITH WORK COLLER ON THE OPEN DATASET

```
corr_matrix=df.corr(numeric_only=True)
corr_matrix
```

	Cumulative Gas	Cumulative Oil	Measured Depth	Deviation	Surface Casing Weight	Production Casing Size	Cumulative Drill Print	Average Mud Weight
Cumulative Gas	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Cumulative Oil	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Measured Depth	0.000000	0.000000	1.000000	0.497207	0.000000	0.000000	0.000000	0.000000
Deviation	0.000000	0.000000	0.497207	1.000000	0.000000	0.000000	0.000000	0.000000
Surface Casing Weight	0.000000	0.000000	0.000000	0.000000	1.000000	0.750000	0.000000	0.000000
Production Casing Size	0.000000	0.000000	0.000000	0.000000	0.750000	1.000000	0.000000	0.000000
Cumulative Drill Print	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000
Average Mud Weight	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
Target	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

```
# Correlation with Target
corr = df.corr(numeric_only=True).round(2)
print('Correlation with Target:\n', corr['Target'].sort_values(ascending=False), '\n\n')
print('Correlation with Target (absolute value):\n', abs(corr['Target'].sort_values(ascending=False)), '\n\n')
print('Correlation with Target (absolute value, rounded to 2 decimals):\n', round(abs(corr['Target'].sort_values(ascending=False)),2), '\n\n')
print('Correlation with Target (absolute value, rounded to 2 decimals, sorted by Target):\n', round(abs(corr['Target'].sort_values(ascending=False)),2).sort_values(ascending=False), '\n\n')
print('Correlation with Target (absolute value, rounded to 2 decimals, sorted by Target, excluding Target):\n', round(abs(corr['Target'].sort_values(ascending=False)),2).sort_values(ascending=False)[1:], '\n\n')
corr[corr.Target>0].Target
```

```
Correlation with Target:
 Measured Depth    1.00
Deviation         0.50
Target            0.00
Cumulative Oil    0.00
Cumulative Gas    0.00
Surface Casing Weight    0.00
Production Casing Size    0.00
Cumulative Drill Print    0.00
Average Mud Weight    0.00
Name: Target, dtype: float64


Correlation with Target (absolute value):
 Measured Depth    1.00
Deviation         0.50
Target            0.00
Cumulative Oil    0.00
Cumulative Gas    0.00
Surface Casing Weight    0.00
Production Casing Size    0.00
Cumulative Drill Print    0.00
Average Mud Weight    0.00
Name: Target, dtype: float64


Correlation with Target (absolute value, rounded to 2 decimals):
 Measured Depth    1.00
Deviation         0.50
Target            0.00
Cumulative Oil    0.00
Cumulative Gas    0.00
Surface Casing Weight    0.00
Production Casing Size    0.00
Cumulative Drill Print    0.00
Average Mud Weight    0.00
Name: Target, dtype: float64


Correlation with Target (absolute value, rounded to 2 decimals, sorted by Target):
 Measured Depth    1.00
Deviation         0.50
Target            0.00
Cumulative Oil    0.00
Cumulative Gas    0.00
Surface Casing Weight    0.00
Production Casing Size    0.00
Cumulative Drill Print    0.00
Average Mud Weight    0.00
Name: Target, dtype: float64


Correlation with Target (absolute value, rounded to 2 decimals, sorted by Target, excluding Target):
 Deviation         0.50
Cumulative Oil    0.00
Cumulative Gas    0.00
Surface Casing Weight    0.00
Production Casing Size    0.00
Cumulative Drill Print    0.00
Average Mud Weight    0.00
Name: Target, dtype: float64
```

```
# Correlation with Target (excluding Target)
corr[corr.Target>0].Target[1:]
```

```
Deviation    0.50
Name: Target, dtype: float64
```

```
# Correlation with Target (excluding Target) (Measured Depth, Deviation, Surface Casing Weight, Production Casing Size, Cumulative Drill Print, Average Mud Weight)
corr[corr.Target>0].Target[1:].drop(['Measured Depth', 'Deviation', 'Surface Casing Weight', 'Production Casing Size', 'Cumulative Drill Print', 'Average Mud Weight'])
```

```
KeyNotFound: [nil] "Measured Depth" not found: key_error_1
```

```
Surface Gravity Weight Kgms    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
Surface Gravity Weight Lbfs    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000    0.000000
```

TOP 5 ROW + 5 COLUMNS

```
Surface Gravity Weight Lbfs    0.000000    0.000000    0.000000    0.000000    0.000000
```

CORRELATION MATRIX

```
Surface Gravity Weight Lbfs    0.000000    0.000000    0.000000    0.000000    0.000000
```

	Equatorial Radius Kms	Equatorial Radius Miles	Equatorial Circumference Kms	Equatorial Circumference Miles	Polar Radius Kms	Polar Radius Miles	Polar Circumference Kms	Polar Circumference Miles
Equatorial Radius Kms	1.000000	1.000000	1.000000	1.000000	0.937000	0.937000	0.937000	0.937000
Equatorial Radius Miles	1.000000	1.000000	1.000000	1.000000	0.937000	0.937000	0.937000	0.937000
Equatorial Circumference Kms	1.000000	1.000000	1.000000	1.000000	0.937000	0.937000	0.937000	0.937000
Equatorial Circumference Miles	1.000000	1.000000	1.000000	1.000000	0.937000	0.937000	0.937000	0.937000
Polar Radius Kms	0.937000	0.937000	0.937000	0.937000	1.000000	1.000000	1.000000	1.000000
Polar Radius Miles	0.937000	0.937000	0.937000	0.937000	1.000000	1.000000	1.000000	1.000000
Polar Circumference Kms	0.937000	0.937000	0.937000	0.937000	1.000000	1.000000	1.000000	1.000000
Polar Circumference Miles	0.937000	0.937000	0.937000	0.937000	1.000000	1.000000	1.000000	1.000000
Surface Gravity Weight Kgms	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Surface Gravity Weight Lbfs	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Here we observe that most correlation between the following features related to the order of highest to lowest:

- Polar Circumference Kms
- Polar Circumference Miles
- Polar Radius Kms
- Polar Radius Miles
- Equatorial Radius Kms
- Equatorial Radius Miles
- Equatorial Circumference Kms
- Equatorial Circumference Miles

Here we observe strong negative linear correlation between the following features related to the order of highest to lowest:

- 1 Coordinate Around 2 Coordinate Kms

```
Surface Gravity Weight Lbfs    0.000000    0.000000    0.000000    0.000000    0.000000
```

**Measured Depth Feet, Measured Depth Meters, Horizontal Displacement Feet, Horizontal Displacement Meters, Vertical Depth Feet, Vertical Depth Meters, Dogleg Severity Degrees 100 Feet, Dogleg Severity Degrees 30 Meters, Dogleg Severity Degrees 100 Meters, Dogleg Severity Degrees 30 Feet, Dogleg Severity Degrees Total, Dogleg Severity Degrees Total Feet, Dogleg Severity Degrees Total Meters, Dogleg Severity Degrees Total 30 Feet, Dogleg Severity Degrees Total 30 Meters, Dogleg Severity Degrees Total 100 Feet, Dogleg Severity Degrees Total 100 Meters, Dogleg Severity Degrees Total 1 Feet, Dogleg Severity Degrees Total 1 Meters, Dogleg Severity Degrees Total 1 Kms, Dogleg Severity Degrees Total 1 Miles, Dogleg Severity Degrees Total 1 Nms, Dogleg Severity Degrees Total 1 Secs, Dogleg Severity Degrees Total 1 Mins, Dogleg Severity Degrees Total 1 Hrs, Dogleg Severity Degrees Total 1 Days, Dogleg Severity Degrees Total 1 Wks, Dogleg Severity Degrees Total 1 Mns, Dogleg Severity Degrees Total 1 Yrs, Dogleg Severity Degrees Total 1 Decs, Dogleg Severity Degrees Total 1 Cns, Dogleg Severity Degrees Total 1 Mls, Dogleg Severity Degrees Total 1 Kms Per Sec, Dogleg Severity Degrees Total 1 Mls Per Sec, Dogleg Severity Degrees Total 1 Kms Per Min, Dogleg Severity Degrees Total 1 Mls Per Min, Dogleg Severity Degrees Total 1 Kms Per Hr, Dogleg Severity Degrees Total 1 Mls Per Hr, Dogleg Severity Degrees Total 1 Kms Per Day, Dogleg Severity Degrees Total 1 Mls Per Day, Dogleg Severity Degrees Total 1 Kms Per Wk, Dogleg Severity Degrees Total 1 Mls Per Wk, Dogleg Severity Degrees Total 1 Kms Per Mn, Dogleg Severity Degrees Total 1 Mls Per Mn, Dogleg Severity Degrees Total 1 Kms Per Yr, Dogleg Severity Degrees Total 1 Mls Per Yr, Dogleg Severity Degrees Total 1 Kms Per Dec, Dogleg Severity Degrees Total 1 Mls Per Dec, Dogleg Severity Degrees Total 1 Kms Per Cn, Dogleg Severity Degrees Total 1 Mls Per Cn, Dogleg Severity Degrees Total 1 Kms Per Ml, Dogleg Severity Degrees Total 1 Mls Per Ml, Dogleg Severity Degrees Total 1 Kms Per Km, Dogleg Severity Degrees Total 1 Mls Per Km, Dogleg Severity Degrees Total 1 Kms Per Nm, Dogleg Severity Degrees Total 1 Mls Per Nm

```
import numpy as np, pandas as pd, matplotlib.pyplot as plt, seaborn as sns, os
%matplotlib inline
from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import (accuracy_score, precision_score, recall_score, f1_score,
                             roc_auc_score, roc_curve, confusion_matrix,
                             classification_report)
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import (RandomForestClassifier, GradientBoostingClassifier,
                              AdaBoostClassifier, VotingClassifier)
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.pipeline import Pipeline
from sklearn.base import BaseEstimator, TransformerMixin
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.manifold import TSNE
from scipy.stats import ttest_ind, chi2_contingency
import warnings, joblib, pickle, time, gc, os, pathlib, copy, collections, itertools, math
warnings.filterwarnings('ignore')
np.random.seed(42)
pd.pd.set_option('display.max_columns', 200)
pd.pd.set_option('display.max_rows', 200)
pd.pd.set_option('display.width', 160)
pd.pd.set_option('display.precision', 4)

SEED = 42
DATA_PATH = 'data/cardiovascular_disease_prime.csv'
OUT_DIR = 'outputs'
os.makedirs(OUT_DIR, exist_ok=True)
sns.set_theme(style='whitegrid', palette='deep', font_scale=1.0, rc={'figure.dpi': 110, 'savefig.dpi': 150, 'savefig.bbox': 'tight'})
plt.rcParams['figure.figure.max_open_warning'] = 0

df = pd.read_csv(DATA_PATH, sep=';')
print(f'Loaded {df.shape[0]:,} rows × {df.shape[1]} columns')
df.head(3)
```

```
Loaded 70,000 rows × 13 columns
```

	age_years	height_cm	weight_kg	gender	ap_hi	ap_hi_cat	ap_lo	ap_lo_cat	cholesterol	glucose	smoking	alcohol	cardio
0	50.4	168	62.0	2	110	Stage 1	80	Stage 1	1	1	0	0	0
1	55.4	156	85.0	1	140	Stage 2	90	Stage 2	3	1	0	0	1
2	51.3	165	64.0	1	130	Stage 1	70	Stage 1	3	1	0	0	1
3	48.3	169	82.0	2	150	Stage 2	100	Stage 2	1	1	0	0	1
4	47.6	151	67.0	1	100	Stage 1	60	Stage 1	1	1	0	0	0
5	60.1	157	93.0	2	120	Stage 1	80	Stage 1	2	2	0	0	1
6	60.9	158	71.0	1	120	Stage 1	80	Stage 1	1	1	0	0	1
7	61.9	164	70.0	2	130	Stage 1	70	Stage 1	3	1	0	0	1
8	48.2	158	78.0	2	100	Stage 1	60	Stage 1	1	1	0	0	0
9	54.3	161	62.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
10	48.9	157	56.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
11	48.7	170	67.0	2	120	Stage 1	80	Stage 1	1	1	0	0	0
12	63.1	154	70.0	1	150	Stage 2	90	Stage 2	3	3	0	0	1
13	52.2	157	67.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
14	61.4	162	70.0	2	140	Stage 2	90	Stage 2	1	1	0	0	1
15	48.3	157	60.0	1	110	Stage 1	70	Stage 1	1	1	0	0	0
16	54.3	165	60.0	2	120	Stage 1	80	Stage 1	1	1	0	0	0
17	48.3	158	55.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
18	48.3	161	65.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
19	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
20	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
21	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
22	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
23	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
24	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
25	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
26	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
27	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
28	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
29	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
30	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
31	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
32	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
33	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
34	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
35	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
36	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
37	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
38	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
39	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
40	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
41	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
42	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
43	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
44	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
45	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
46	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
47	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
48	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
49	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
50	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
51	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
52	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
53	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
54	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
55	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
56	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
57	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
58	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
59	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
60	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
61	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
62	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
63	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
64	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
65	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
66	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
67	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
68	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
69	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
70	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
71	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
72	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
73	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
74	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
75	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
76	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
77	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
78	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
79	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
80	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
81	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
82	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
83	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
84	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
85	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
86	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
87	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
88	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
89	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
90	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
91	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
92	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
93	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
94	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
95	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
96	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
97	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
98	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
99	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
100	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
101	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
102	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
103	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
104	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
105	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
106	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
107	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
108	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
109	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
110	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
111	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
112	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
113	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
114	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
115	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
116	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
117	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
118	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
119	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
120	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
121	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
122	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
123	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
124	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
125	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
126	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
127	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
128	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
129	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
130	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
131	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
132	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
133	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
134	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
135	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
136	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
137	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
138	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
139	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
140	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
141	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
142	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
143	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
144	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
145	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
146	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
147	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
148	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
149	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
150	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
151	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
152	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
153	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
154	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
155	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
156	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
157	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
158	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
159	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
160	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
161	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
162	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
163	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
164	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
165	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
166	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
167	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
168	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
169	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
170	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
171	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
172	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
173	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
174	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
175	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
176	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
177	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
178	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
179	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
180	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
181	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
182	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
183	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
184	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
185	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
186	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
187	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
188	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
189	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
190	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
191	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
192	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
193	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
194	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
195	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
196	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
197	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
198	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
199	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
200	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
201	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
202	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
203	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
204	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
205	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
206	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
207	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
208	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
209	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
210	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
211	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
212	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
213	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
214	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
215	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
216	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
217	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
218	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
219	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1	0	0	0
220	48.3	157	54.0	1	120	Stage 1	80	Stage 1	1	1

```
[1] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[2] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[3] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[4] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[5] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[6] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[7] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[8] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[9] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[10] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[11] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[12] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[13] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[14] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[15] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[16] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[17] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[18] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[19] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[20] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[21] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[22] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[23] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[24] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[25] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[26] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[27] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[28] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[29] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[30] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[31] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[32] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[33] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[34] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[35] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[36] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[37] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[38] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[39] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[40] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[41] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[42] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[43] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[44] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[45] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[46] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[47] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[48] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[49] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[50] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[51] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[52] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[53] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[54] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[55] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[56] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[57] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[58] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[59] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[60] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[61] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[62] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[63] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[64] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[65] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[66] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[67] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[68] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[69] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[70] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[71] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[72] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[73] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[74] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[75] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[76] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[77] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[78] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[79] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[80] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[81] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[82] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[83] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[84] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[85] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[86] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[87] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[88] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[89] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[90] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[91] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[92] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[93] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[94] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[95] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[96] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[97] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[98] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[99] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
[100] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
```

```
> cat(paste(sprintf("%s", gsub(pattern=" ", replacement="", x=.)), collapse=" "))
[1] 0.9999999999999999 'penalty2l1' 'solver' 'liblinear' 'tol' 0.0001
```

```
Predicted Value: 0 Real Value: 0 Sentence: 'Sometimes'
```

1	2	2	1	Female	Overweight
2	2	2	1	Male	Overweight
3	2	2	1	Male	Overweight
4	2	2	1	Male	Overweight

```
Predicted:[0.0] True: [0.0] Sepsis:[0.0]
Predicted:[0.0] True: [0.0] Sepsis:[0.0]
Predicted:[0.0] True: [0.0] Sepsis:[0.0]
```

```
[CV] max_features: 'log2', max_depth: None, n_estimators: 1000 ..... [
```

```
[0.0000000e+00 7.0000000e+00 1.0000000e+00 1.0000000e+00 1.0000000e+00
 1.0000000e+00 1
```

```
epoch 1 batch 1000 loss: 0.1 acc: 0.9 lr: 0.0000100
epoch 1 batch 1100 loss: 0.1 acc: 0.9 lr: 0.0000100
epoch 1 batch 1200 loss: 0.1 acc: 0.9 lr: 0.0000100
epoch 1 batch 1300 loss: 0.1 acc: 0.9 lr: 0.0000100
epoch 1 batch
```

```
0.0 0.0 0.0 
```

| | | |

```
[0.00000000]] [-0.00000000] [-0.00000000] [-0.00000000] [-0.00000000]
[-0.00000000
```

FINDING THE BEST ALGORITHM ON TRAINING DATASET

Training algorithm on training data and calculate scores

```
# train_dec_error
```

TESTING

```
# test_dec_error
```

Removing target from test set

```
# DROP THE TARGET COLUMN
X_test_no_target = X_test.drop('Transformed_Target', axis=1)
# PRINT THE SHAPE OF THE NEW DATAFRAME
print('X_test_no_target.shape:', X_test_no_target.shape)
```

```
X_test_no_target.shape: (2000, 11)
```

```
# head
X_test_no_target.head()
```

	Customer ID	Credit Score	Geography	Gender	Age	Tenure	Balance	Number of Products	Credit Card	Active Member	Estimated Salary
7	15677954	792	France	Female	28	4	130142.79	1	1	0	75192.12
8	15773898	765	Spain	Male	74	2	96707.07	1	0	0	92888.52
9	15651990	733	Spain	Male	43	8	62772.41	1	1	1	75799.31
10	15772798	651	France	Male	39	5	0.00	2	0	0	181404.33
11	15792365	772	Germany	Male	50	2	136857.06	1	1	1	62358.49

```
# head
y_test_labels[:5]
```

```
7    0
8    0
9    0
10   1
11   0
Name: Transformed_Target, dtype: int64
```

```
# head
y_test_pred[:5]
```

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

```
# head
y_test_pred_prob[:5]
```

```
array([[0.9, 0.1],
       [0.9, 0.1],
       [0.9, 0.1],
       [0.1, 0.9],
       [0.9, 0.1]])
```

```
# CONFUSION MATRIX
cm = confusion_matrix(y_test_labels, y_test_pred)
print('Confusion matrix\n\n', cm)
print('True Positives(TP) = ', cm[0,0])
print('False Positives(FP) = ', cm[0,1])
print('True Negatives(TN) = ', cm[1,0])
print('False Negatives(FN) = ', cm[1,1])
```

```
Confusion matrix

 [[1577  416]
 [ 207  800]]
True Positives(TP) =  1577
False Positives(FP) =  416
True Negatives(TN) =  207
False Negatives(FN) =  800
```

ACCURACY

DEALING WITH TEST COLUMN

```
# DROP THE TARGET COLUMN
X_test_no_target = X_test.drop('Transformed_Target', axis=1)
```