Risk Scorecard Development:

> This notebook describes the procedure to develop risk score and risk scorecard.
>
> ### Risk Score:
>
> A risk score is a numerical score that is derived from the statistical analysis and represents transactions riskiness. Higher risk score corresponds to high default probability.
>
> This concept of scorecard development is popular in finance industry. Various types of scores like risk scores, credit scores, credit rating are being developed in financial domain to understand credit risk, default risk, product pricing etc. Risk scores are usually developed using big data analytics and efficient machine learning algorithms, including, but not limited to, logistic regression, ensemble, and gradient boosting decision trees.

Developing the Risk Scores from predicted probabilities of developed model:

> Once the machine learning model is assessed and finalized to be implemented in production, loan applicants risk scores and scorecard should be developed. These risk scores are further used to decide the score threshold for loan acceptance decisioning in real time.
>
> Defining threshold for loan applicants acceptance decision, based on the developed models prediction probabilities will be challenging as transactions predicted probabilities will be tightly placed between small scale. So, it necessitates the scaling methodology which can scale the predicted probabilities to a large range of scores.
>
> For developing risk scores, logistic regression is used as it assumes linearity between the dependent and the response variables. Lets first compute the risk score and then develop the risk scorecard.

1. Risk Score Computation

Explanatory example:

> Following simple example describes the steps along with the code for developing the risk score. This example is just for reference and you do not need to input your data in this code.

Following is a risk score development process flow:

1. Compute log of odds (log(good/bad)) of each transaction.
2. Train a logistic regression model on log of odds, target(default) labels and get intercept and slope of line which separates log of odds and log of target labels.
3. Develop risk scores by using log(odds), slope and intercept of logistic regression and by tuning PDO, score corresponding to 1% bad rate.

Definitions:

log of odds:

> Ratio of apllicants chances of being good to bad is odds and its log is log of odds. These good and bad probabilities are developed models prediction probabilities.
>
> #### PDO:
>
> Points to double the odds (PDO) define the score points by which odds double.
>
> #### Score corresponding to 1% bad rate:
>
> Score value for 1% bad rate.
>
> #### Scaling:

Scaling factors in the aligned score equation does not affect the predictive strength of the scorecard. scaling term 1 and scaling term 2 are used in the aligned score equation.

Intercept and Slope:

Slope and intercept of logistic regression line classifying log(odds) and fraud labels.

Aligned score equation:

Developed aligned score equation which is linear function of log of odds.

$$AlignedScore = \left[ScoreCorrepondingTo1PctBadRate + \frac{PDO}{ln(2)} * (ScalingTerm1 + intercept)\right] - \left[\frac{PDO}{ln(2)} * Slope * ScalingTerm2 * log(odds)\right]$$

In [1]:
```
# Import the required libraries
import warnings
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
import math

! pip install ipywidgets
! jupyter nbextension enable --py widgetsnbextension

import ipywidgets as widgets

%matplotlib inline
from scipy.integrate import ode
from ipywidgets import *
from IPython.display import clear_output, display, HTML
import random
from sklearn.linear_model import LogisticRegression

# model_pickle_file
import pickle
from pickle import load

# Disbale the future warning in python
warnings.simplefilter(action='ignore', category=FutureWarning)
```

```
Requirement already satisfied: ipywidgets in /usr/local/lib/python3.10/dist-packages (7.7.1)
Requirement already satisfied: ipykernel>=4.5.1 in /usr/local/lib/python3.10/dist-packages (from ipywidgets) (5.5.6)
Requirement already satisfied: ipython-genutils~=0.2.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets) (0.2.0)
Requirement already satisfied: traitlets>=4.3.1 in /usr/local/lib/python3.10/dist-packages (from ipywidgets) (5.7.1)
Requirement already satisfied: widgetsnbextension~=3.6.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets) (3.6.6)
Requirement already satisfied: ipython>=4.0.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets) (7.34.0)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets) (3.0.9)
Requirement already satisfied: jupyter-client in /usr/local/lib/python3.10/dist-packages (from ipykernel>=4.5.1->ipywidgets) (6.1.12)
Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.10/dist-packages (from ipykernel>=4.5.1->ipywidgets) (6.3.2)
Requirement already satisfied: setuptools>=18.5 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (67.7.2)
Collecting jedi>=0.16 (from ipython>=4.0.0->ipywidgets)
  Downloading jedi-0.19.1-py2.py3-none-any.whl (1.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 19.8 MB/s eta 0:00:00
Requirement already satisfied: decorator in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.7.5)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0
->ipywidgets) (3.0.39)
Requirement already satisfied: pygments in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (2.16.1)
Requirement already satisfied: backcall in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.2.0)
Requirement already satisfied: matplotlib-inline in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.1.6)
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (4.8.0)
Requirement already satisfied: notebook>=4.4.1 in /usr/local/lib/python3.10/dist-packages (from widgetsnbextension~=3.6.0->ipywidgets) (6.
5.5)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from jedi>=0.16->ipython>=4.0.0->ipywidgets)
(0.8.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidge
ts) (3.1.2)
Requirement already satisfied: pyzmq<25,>=17 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->i
pywidgets) (23.2.1)
Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipy
widgets) (23.1.0)
Requirement already satisfied: jupyter-core>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.
6.0->ipywidgets) (5.5.0)
Requirement already satisfied: nbformat in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywid
gets) (5.9.2)
Requirement already satisfied: nbconvert>=5 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ip
ywidgets) (6.5.4)
Requirement already satisfied: nest-asyncio>=1.5 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.
0->ipywidgets) (1.5.8)
Requirement already satisfied: Send2Trash>=1.8.0 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.
0->ipywidgets) (1.8.2)
Requirement already satisfied: terminado>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0
->ipywidgets) (0.17.1)
Requirement already satisfied: prometheus-client in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.
0->ipywidgets) (0.18.0)
Requirement already satisfied: nbclassic>=0.4.7 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0
->ipywidgets) (1.0.0)
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.10/dist-packages (from jupyter-client->ipykernel>=4.5.1->ipyw
idgets) (2.8.2)
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.10/dist-packages (from pexpect>4.3->ipython>=4.0.0->ipywidgets)
(0.7.0)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.10/dist-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipyth
on>=4.0.0->ipywidgets) (0.2.9)
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.10/dist-packages (from jupyter-core>=4.6.1->notebook>=4.4.1->wid
getsnbextension~=3.6.0->ipywidgets) (3.11.0)
Requirement already satisfied: jupyter-server>=1.8 in /usr/local/lib/python3.10/dist-packages (from nbclassic>=0.4.7->notebook>=4.4.1->widg
etsnbextension~=3.6.0->ipywidgets) (1.24.0)
Requirement already satisfied: notebook-shim>=0.2.3 in /usr/local/lib/python3.10/dist-packages (from nbclassic>=0.4.7->notebook>=4.4.1->wid
getsnbextension~=3.6.0->ipywidgets) (0.2.3)
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.
6.0->ipywidgets) (4.9.3)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbexte
nsion~=3.6.0->ipywidgets) (4.11.2)
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=
3.6.0->ipywidgets) (6.1.0)
Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextensio
n~=3.6.0->ipywidgets) (0.7.1)
Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnb
extension~=3.6.0->ipywidgets) (0.4)
Requirement already satisfied: jupyterlab-pygments in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsn
bextension~=3.6.0->ipywidgets) (0.2.2)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbext
ension~=3.6.0->ipywidgets) (2.1.3)
Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbe
xtension~=3.6.0->ipywidgets) (0.8.4)
Requirement already satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbext
ension~=3.6.0->ipywidgets) (0.8.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension
~=3.6.0->ipywidgets) (23.2)
Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgets
nbextension~=3.6.0->ipywidgets) (1.5.0)
Requirement already satisfied: tinycss2 in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~
=3.6.0->ipywidgets) (1.2.1)
Requirement already satisfied: fastjsonschema in /usr/local/lib/python3.10/dist-packages (from nbformat->notebook>=4.4.1->widgetsnbextensio
n~=3.6.0->ipywidgets) (2.18.1)
Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.10/dist-packages (from nbformat->notebook>=4.4.1->widgetsnbextensi
on~=3.6.0->ipywidgets) (4.19.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.1->jupyter-client->ipykernel>=
4.5.1->ipywidgets) (1.16.0)
Requirement already satisfied: argon2-cffi-bindings in /usr/local/lib/python3.10/dist-packages (from argon2-cffi->notebook>=4.4.1->widgetsn
bextension~=3.6.0->ipywidgets) (21.2.0)
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat->notebook>=4.4.1->w
idgetsnbextension~=3.6.0->ipywidgets) (23.1.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbfor
mat->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (2023.7.1)
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat->notebook>=4.
4.1->widgetsnbextension~=3.6.0->ipywidgets) (0.30.2)
```

```
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat->notebook>=4.4.1->
widgetsnbextension~=3.6.0->ipywidgets) (0.11.0)
Requirement already satisfied: anyio<4,>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from jupyter-server>=1.8->nbclassic>=0.4.7->note
book>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (3.7.1)
Requirement already satisfied: websocket-client in /usr/local/lib/python3.10/dist-packages (from jupyter-server>=1.8->nbclassic>=0.4.7->not
ebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.6.4)
Requirement already satisfied: cffi>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from argon2-cffi-bindings->argon2-cffi->notebook>=4.
4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.16.0)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->nbconvert>=5->notebook>=4.4.1
->widgetsnbextension~=3.6.0->ipywidgets) (2.5)
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from bleach->nbconvert>=5->notebook>=4.4.1->widgets
nbextension~=3.6.0->ipywidgets) (0.5.1)
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.10/dist-packages (from anyio<4,>=3.1.0->jupyter-server>=1.8->nbclassic>=
0.4.7->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (3.4)
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.10/dist-packages (from anyio<4,>=3.1.0->jupyter-server>=1.8->nbclassi
c>=0.4.7->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.3.0)
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (from anyio<4,>=3.1.0->jupyter-server>=1.8->nbclas
sic>=0.4.7->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.1.3)
Requirement already satisfied: pycparser in /usr/local/lib/python3.10/dist-packages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->n
otebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (2.21)
Installing collected packages: jedi
Successfully installed jedi-0.19.1
Enabling notebook extension jupyter-js-widgets/extension...
Paths used for configuration of notebook:
        /root/.jupyter/nbconfig/notebook.json
Paths used for configuration of notebook:

      - Validating: OK
Paths used for configuration of notebook:
        /root/.jupyter/nbconfig/notebook.json
```

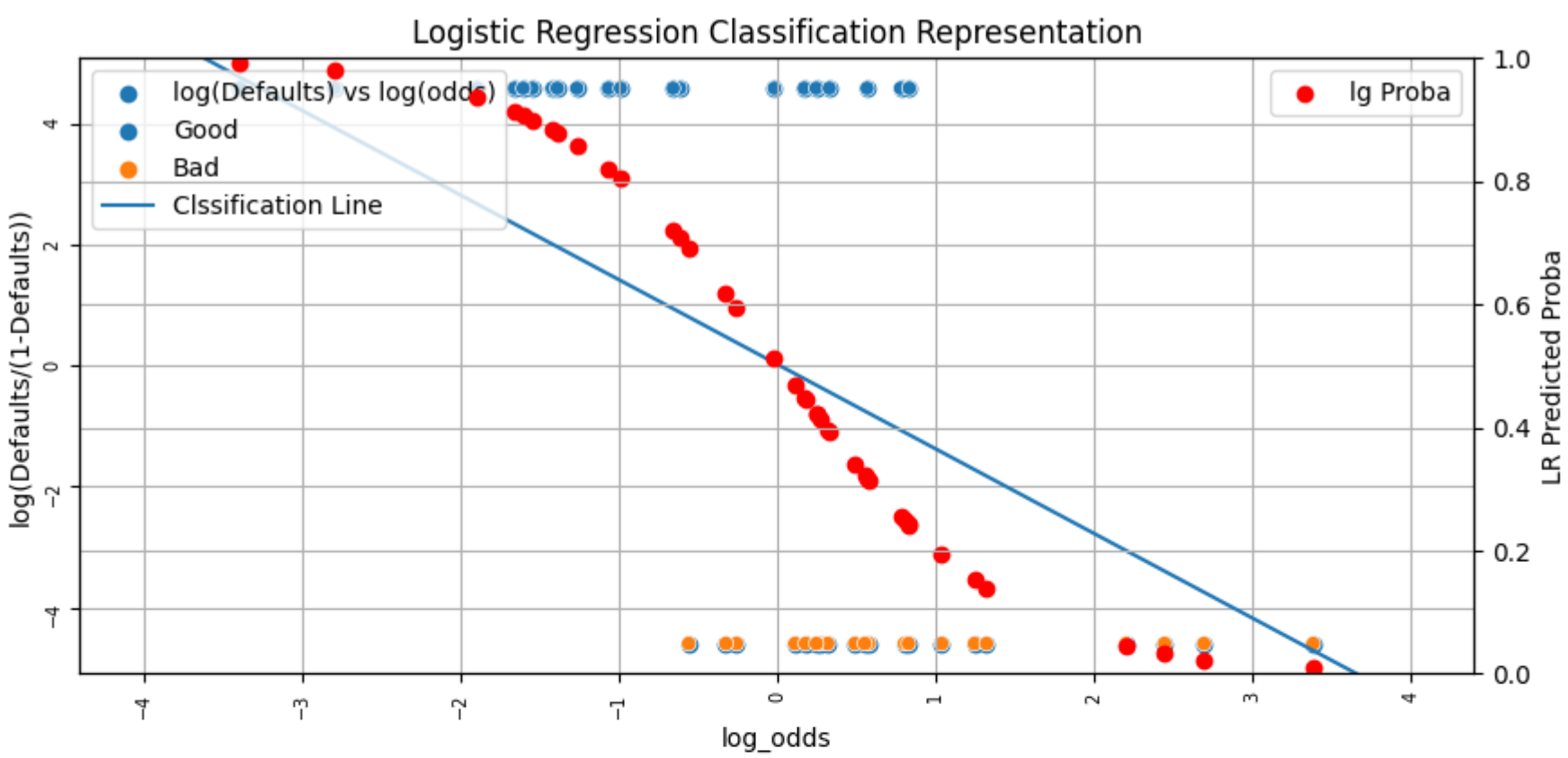
In [18]:

```
def risk_score_example(example_points):
    """
This function presents an example of risk score computation steps.
Input: example_points - number of example points that you would like to use for this demo. Example points are good and bad points each.
"""

# lets prepare the dataframe of these example points
np.random.seed(10)
bad_proba=np.random.uniform(0.3,1,example_points)  # example points which belongs to bad probability (Proba range = 0.3 to 1)
bad_proba=np.append(bad_proba,np.random.uniform(0,0.7,example_points)) # example points which belongs to good probability (Proba range
good_proba=1-bad_proba
fraud_1=[1]*example_points # 1 - good example points (fraud label)
fraud_0=[0]*example_points # 0 - bad example points (fraud label)
fraud=fraud_1+fraud_0
fraud_1_s=["Good"]*example_points # 1 - good example points (fraud label)
fraud_0_s=["Bad"]*example_points # 0 - bad example points (fraud label)
fraud_s=fraud_1_s+fraud_0_s


## --------------------      1.     Compute log of odds (log(good/bad)) of each transaction.   ---------------------- ##
odds=good_proba/bad_proba
log_odds=np.log(odds)

data={"good_proba":good_proba,
     "bad_proba":bad_proba,
     "fraud":fraud,
      "fraud_s":fraud_s,
      "odds":odds,
     "log_odds":log_odds
     }
example_score_df = pd.DataFrame(data)


## ------------------  2. Lets build a logistic regression model on log(odds), fraud labels  -------------------- ##
log_score = LogisticRegression(random_state=0, solver="lbfgs", max_iter = 1000).fit(pd.DataFrame(example_score_df['log_odds'])
                           , example_score_df['fraud'])

print("-"*75)
print("Logistic regressions Intercept and Slope are :")

 # Intercept value :
print('Intercept:', log_score.intercept_.ravel())

# Slope value :
print('Slope:', log_score.coef_.ravel())

# predicted probabilities
lg_predicted_proba=log_score.predict_proba(np.array([example_score_df['log_odds'].to_list()]).reshape(-1,1))[:,0]
lg_predicted_proba=1-lg_predicted_proba

## logistic regression classification line
x_lg_line=[i for i in range(-4,5)]
y_lg_line=log_score.coef_.ravel()*x_lg_line+log_score.intercept_.ravel()


## ----------------- 3.     Develop risk scores by using log(odds), slope and intercept of logistic regression and by tuning PDO, score
# Following parameters needs to be tuned in the score computation process. For this example these are already tuned:
Slope=log_score.coef_.ravel()  # logistic regression fit slope
Intercept=log_score.intercept_.ravel()  # logistic regression fit intercept
PDO=20  # points to double the odds
score_corresponding_to_1pct_bad_rate= 500  # score corresponding to 1% bad rate
Term_1_scaling_factor=math.log(89)  # Term 1 scaling factor
Term_2_scaling_factor=0.7 # Term 2 scaling factor

Aligned_score_term1=score_corresponding_to_1pct_bad_rate+((Term_1_scaling_factor+Intercept)*PDO/math.log(2))
Aligned_score_term2=PDO*Slope/math.log(2)*Term_2_scaling_factor
```

```
    print("-"*75)
    print("Aligned Score Equation is= %5.3f - %5.3f*score" %(Aligned_score_term1,Aligned_score_term2))
#     print("-"*75)

#     print("-"*75)
    print("Score corresponding to 1% bad rate is " ,Aligned_score_term1-(Aligned_score_term2*4.59511985013459))
#     print("-"*75)

#     print("-"*75)
    print("Score corresponding to 2% bad rate is " ,Aligned_score_term1-(Aligned_score_term2*3.89182029811063))
    print("-"*75)

    example_score_df["Alligned_Score"]=Aligned_score_term1-(Aligned_score_term2*example_score_df["log_odds"])

    ## Plot the logistic regression fit and alligned score graphs
    fig = plt.figure(figsize=(10, 10))

    x = np.linspace(0., 5., 100)
    y = np.sin(x)

    plt.subplots_adjust(wspace= 0.25, hspace= 0.25)

    # Create second axes, the top-left plot with orange plot
    colors = {0:'tab:blue', 1:'tab:orange'}

    sub2 = fig.add_subplot(2,1,1) # two rows, two columns, first cell
    log_frauds_1=[math.log(0.99/(1-0.99))]*example_points
    log_frauds_0=[math.log(0.01/(1-0.01))]*example_points
    log_frauds=log_frauds_1+log_frauds_0
    sub2.scatter(example_score_df["log_odds"],log_frauds,label='log(Defaults) vs log(odds)')
    sns.scatterplot(x=example_score_df["log_odds"],y=log_frauds,hue=example_score_df["fraud_s"])
    sub2.plot(x_lg_line,y_lg_line,label='Clssification Line')
    sub2.set_ylabel("log(Defaults/(1-Defaults))")
    sub2.set_xlabel("log_odds")
    sub2.set_title("Logistic Regression Classification Representation")
    sub2.tick_params(labelrotation=90,labelsize = 7)
    sub2.legend(loc='upper left')
    sub2.set_ylim([log_frauds_0[0]-0.5, log_frauds_1[0]+0.5])
    sub2.grid()
    sub3 = sub2.twinx()
    sub3.scatter(example_score_df["log_odds"],lg_predicted_proba,label='lg Proba',color='r')
    sub3.set_ylabel("LR Predicted Proba")
    sub3.set_ylim([0,1])
    sub3.legend(loc='upper right')
    sub3.grid()

    sub1 = fig.add_subplot(2,1,2) # two rows, two columns, fist cell
    sns.regplot(x=example_score_df["log_odds"],y=example_score_df["Alligned_Score"],
                line_kws=dict(alpha=0.2, color='red', linewidth=2))
    sns.scatterplot(x=example_score_df["log_odds"],y=example_score_df["Alligned_Score"],hue=example_score_df["fraud_s"])
    sub1.set_xlabel(" log_odds")
    sub1.set_ylabel("Alligned_Score")
    sub1.set_title("Alligned Score vs log(odds)")
    sub1.tick_params(labelrotation=90,labelsize = 7)
    sub1.set_ylim([example_score_df["Alligned_Score"].min()-20, example_score_df["Alligned_Score"].max()+20])
    sub1.grid()

    plt.show()

    print("-"*75)
    print("Example Dataframe is as follows :")

    example_score_df.rename(columns={'fraud':'Good_Applicant_Flag','fraud_s':'Applicants'},inplace=True)

    return(example_score_df)
```

In [19]:
```
risk_score_example(20)
```

```
---------------------------------------------------------------------------
Logistic regressions Intercept and Slope are :
Intercept: [0.02765529]
Slope: [-1.40055982]
---------------------------------------------------------------------------
Aligned Score Equation is= 630.313 - -28.288*score
Score corresponding to 1% bad rate is  [760.29997884]
Score corresponding to 2% bad rate is  [740.40494974]
---------------------------------------------------------------------------
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LogisticRegression was f
itted with feature names
  warnings.warn(
```



Example Dataframe is as follows :

Out[19]:

	good_proba	bad_proba	Good_Applicant_Flag	Applicants	odds	log_odds	Alligned_Score
0	0.160076	0.839924	1	Good	0.190583	-1.657666	583.420359
1	0.685474	0.314526	1	Good	2.179384	0.779042	652.350277
2	0.256446	0.743554	1	Good	0.344893	-1.064522	600.199294
3	0.175837	0.824163	1	Good	0.213353	-1.544809	586.612875
4	0.351045	0.648955	1	Good	0.540939	-0.614449	612.931031
5	0.542642	0.457358	1	Good	1.186473	0.170985	635.149470
6	0.561356	0.438644	1	Good	1.279753	0.246667	637.290383
7	0.167629	0.832371	1	Good	0.201387	-1.602529	584.980094
8	0.581622	0.418378	1	Good	1.390185	0.329437	639.631792
9	0.638162	0.361838	1	Good	1.763669	0.567396	646.363207
10	0.220248	0.779752	1	Good	0.282459	-1.264221	594.550184
11	0.032625	0.967375	1	Good	0.033725	-3.389518	534.429500
12	0.697236	0.302764	1	Good	2.302905	0.834171	653.909779
13	0.341465	0.658535	1	Good	0.518523	-0.656771	611.733818
14	0.131165	0.868835	1	Good	0.150967	-1.890694	576.828426
15	0.271232	0.728768	1	Good	0.372178	-0.988382	602.353149
16	0.194771	0.805229	1	Good	0.241883	-1.419300	590.163277
17	0.495687	0.504313	1	Good	0.982895	-0.017253	629.824565
18	0.057558	0.942442	1	Good	0.061073	-2.795679	551.228097
19	0.199797	0.800203	1	Good	0.249683	-1.387564	591.061044
20	0.620219	0.379781	0	Bad	1.633096	0.490478	644.187326
21	0.900481	0.099519	0	Bad	9.048329	2.202580	692.619504
22	0.738661	0.261339	0	Bad	2.826454	1.039023	659.704652
23	0.528106	0.471894	0	Bad	1.119122	0.112545	633.496306
24	0.690717	0.309283	0	Bad	2.233282	0.803472	653.041364
25	0.696190	0.303810	0	Bad	2.291533	0.829221	653.769745
26	0.567563	0.432437	0	Bad	1.312476	0.271916	638.004615
27	0.640803	0.359197	0	Bad	1.783989	0.578852	646.687275
28	0.544722	0.455278	0	Bad	1.196460	0.179367	635.386595
29	0.579273	0.420727	0	Bad	1.376837	0.319789	639.358853
30	0.436344	0.563656	0	Bad	0.774131	-0.256014	623.070469
31	0.634847	0.365153	0	Bad	1.738578	0.553068	645.957878
32	0.363946	0.636054	0	Bad	0.572193	-0.558279	614.519966
33	0.776535	0.223465	0	Bad	3.474968	1.245585	665.547913
34	0.936678	0.063322	0	Bad	14.792413	2.694114	706.524091
35	0.789510	0.210490	0	Bad	3.750819	1.321974	667.708807
36	0.920211	0.079789	0	Bad	11.533048	2.445217	699.483237
37	0.419923	0.580077	0	Bad	0.723909	-0.323089	621.173042
38	0.967173	0.032827	0	Bad	29.462336	3.383113	726.014563
39	0.561599	0.438401	0	Bad	1.281017	0.247654	637.318301

Observations :

1. Logistic regression line classifies good and bad appilcants.
2. Aligned score is linearly mapped for the log(odds).
3. Good applicants has lower risk scores than the bad/defaulter applicants.
4. Score corresponding to 1 pct bad rate is 760. Odds are doubling for 20 risk points which is PDO value. eg. Odds 29.46 (row no. 38) corresoponds to risk score of 726 and its halved odds 14.79 (row no. 34) corresponds to risk score of 706.

> Note: In this example we have adjusted scaling term 1 and scaling term 2 in such a way that score corresponding to 1% bad rate and PDO observations are correctly followed. In the actual scorecard development our objective is to get a required distribution and stability of risk scorecard by tuning the PDO, score corresponding to 1% bad rate, scaling term 1 and scaling term 2 which may not result in following the PDO and score corresponding to 1% bad rate as observed here.