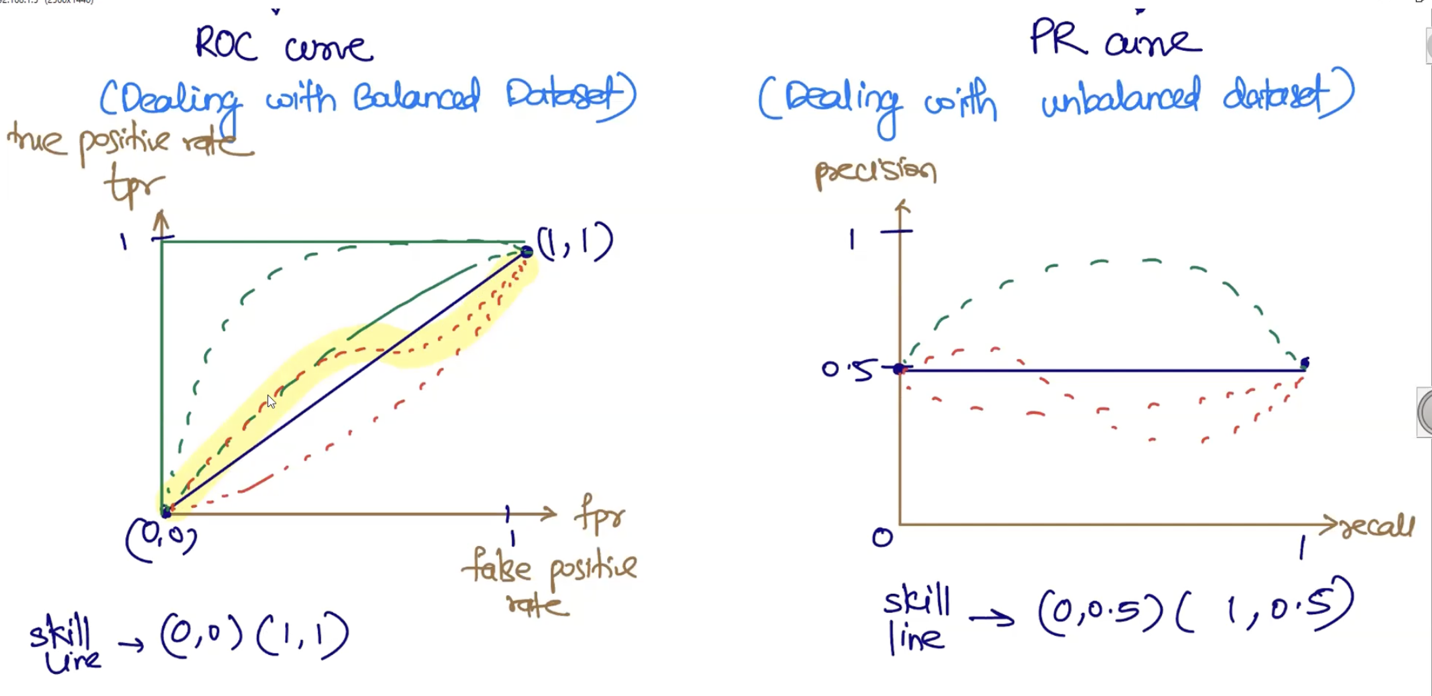
Visualization-Based for Classification metrics

We have two Visualization-Based for Classification metrics:

1. ROC Curve 🡺 Dealing with balanced dataset
2. PR Curve 🡺 Dealing with unbalanced dataset

We Reject the model if it is bellow skill line



Example:

|  |  |  |
| --- | --- | --- |
|  |  | data = pd.read\_csv('Social\_Network\_Ads.csv') |
|  |  | features = data.iloc[:,[2,3]].values  label = data.iloc[:,4].values |
|  |  | data.Purchased.value\_counts()  0 257  1 143  Name: Purchased, dtype: int64 |
|  |  | from sklearn.model\_selection import train\_test\_split  from sklearn.linear\_model import LogisticRegression  X\_train,X\_test,y\_train,y\_test = train\_test\_split(features,  label,  test\_size=0.2,  random\_state=199) |
|  |  | model = LogisticRegression()  model.fit(X\_train,y\_train) |
|  |  | model.score(X\_train,y\_train)  0.8375 |
|  |  | model.score(X\_test,y\_test)  0.8875 |
|  | **ROC Curve**  We know this dataset is unbalanced and we canot use ROC. We just show this for understand the topic |  |
|  | For ROC, you need to use probability for true label. For label one. | model.predict\_proba(X\_test) |
|  | We are extracting index 1 | model.predict\_proba(X\_test)[:,1]  array([0.30836264, 0.91079273, 0.13700251, 0.00401364, 0.88616071,  0.26557903, 0.55949243, 0.21612017, 0.99728588, 0.19815624,  0.42705761, 0.87926715, 0.18204413, 0.31550988, 0.1630387 ,  0.969827 , 0.8154532 , 0.93892347, 0.02560173, 0.5908719 ,  0.99426102, 0.92376635, 0.00518353, 0.02644595, 0.0780933 ,  0.13476784, 0.78985382, 0.00831599, 0.98412275, 0.05254734,  0.00132795, 0.95645756, 0.00688454, 0.0142874 , 0.73021371,  0.17713787, 0.01319186, 0.90149143, 0.99146485, 0.24832158,  0.92737277, 0.96220317, 0.35798254, 0.01415401, 0.25640291,  0.00131539, 0.00388266, 0.33006896, 0.0373644 , 0.04169682,  0.33854344, 0.00638312, 0.00600248, 0.00117359, 0.00283913,  0.05827129, 0.02258472, 0.1630387 , 0.14744885, 0.99654298,  0.59661342, 0.00170823, 0.33854344, 0.26557903, 0.79533085,  0.00940094, 0.06287203, 0.17783311, 0.01900801, 0.0475731 ,  0.17713787, 0.20895466, 0.36567181, 0.05596516, 0.05723516,  0.7283332 , 0.07843621, 0.50980936, 0.59546459, 0.8767174 ]) |
|  | Plot ROC | from sklearn.metrics import roc\_curve  from sklearn.metrics import roc\_auc\_score  import matplotlib.pyplot as plt  %matplotlib inline |
|  | Step1: Extract the prob of true value for your label (1)  #Step2: Calc AUC  #Step3: Plot the Skill line (0,0) (1,1)  #Step4: Plot ROC  y\_test is True value  probabilityValues is the predicted value | 1.  probabilityValues = model.predict\_proba(X\_test)[:,1]  2.  auc = roc\_auc\_score(y\_test, probabilityValues)  3.  plt.plot([0,1],[0,1] , linestyle='--')  4.  fpr,tpr,\_ = roc\_curve(y\_test , probabilityValues)  plt.plot(fpr,tpr)  print(auc)  0.9574036511156186 |
|  | #Note: For this example use of ROC is invalid since the dataset is an UNBALANCED dataset. |  |
|  | **PR Curve** |  |
|  | #Step1: Extract the prob values for label 1  #Step2: Calc PR and AUC  #Step3: Plot the Skill line (0,0.5) (1,0.5)  #Step4: Plot PR Curve | from sklearn.metrics import precision\_recall\_curve  from sklearn.metrics import auc  import matplotlib.pyplot as plt  %matplotlib inline  #Step1:  probabilityValues = model.predict\_proba(X\_test)[:,1]  #Step2:  precision,recall,\_ = precision\_recall\_curve(y\_test, probabilityValues)  aucPR = auc(recall,precision)  print(aucPR)  #Step3:  plt.plot([0,1],[0.5,0.5], linestyle='--')  #Step4  plt.plot(recall,precision)  0.9164619166014315 |
|  | We will accept the model !!! |  |

Example 2:

Use-Case: You need to create a model that can predict the species of the iris flower based on the biological factor of iris flower:

* Iris Setosa
* Iris Versicolor
* Iris Virginica

Use:

LogisticRegression

KNeighborClassifier

SL = 0.1