{  
 "cells": [  
 {  
 "cell\_type": "markdown",  
 "id": "44847216",  
 "metadata": {},  
 "source": [  
 "# Week 1 : "  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "0a1ddcb0",  
 "metadata": {},  
 "source": [  
 "###### Data Exploration : "  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "d33e58e3",  
 "metadata": {},  
 "source": [  
 "(1) Perform descriptive analysis. Understand the variables and their corresponding values."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 1,  
 "id": "32c5d9dd",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "import pandas as pd\n",  
 "import numpy as np\n",  
 "import matplotlib.pyplot as plt\n",  
 "import seaborn as sns"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 2,  
 "id": "368bf531",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<div>\n",  
 "<style scoped>\n",  
 " .dataframe tbody tr th:only-of-type {\n",  
 " vertical-align: middle;\n",  
 " }\n",  
 "\n",  
 " .dataframe tbody tr th {\n",  
 " vertical-align: top;\n",  
 " }\n",  
 "\n",  
 " .dataframe thead th {\n",  
 " text-align: right;\n",  
 " }\n",  
 "</style>\n",  
 "<table border=\"1\" class=\"dataframe\">\n",  
 " <thead>\n",  
 " <tr style=\"text-align: right;\">\n",  
 " <th></th>\n",  
 " <th>Pregnancies</th>\n",  
 " <th>Glucose</th>\n",  
 " <th>BloodPressure</th>\n",  
 " <th>SkinThickness</th>\n",  
 " <th>Insulin</th>\n",  
 " <th>BMI</th>\n",  
 " <th>DiabetesPedigreeFunction</th>\n",  
 " <th>Age</th>\n",  
 " <th>Outcome</th>\n",  
 " </tr>\n",  
 " </thead>\n",  
 " <tbody>\n",  
 " <tr>\n",  
 " <th>0</th>\n",  
 " <td>6</td>\n",  
 " <td>148</td>\n",  
 " <td>72</td>\n",  
 " <td>35</td>\n",  
 " <td>0</td>\n",  
 " <td>33.6</td>\n",  
 " <td>0.627</td>\n",  
 " <td>50</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>1</th>\n",  
 " <td>1</td>\n",  
 " <td>85</td>\n",  
 " <td>66</td>\n",  
 " <td>29</td>\n",  
 " <td>0</td>\n",  
 " <td>26.6</td>\n",  
 " <td>0.351</td>\n",  
 " <td>31</td>\n",  
 " <td>0</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>2</th>\n",  
 " <td>8</td>\n",  
 " <td>183</td>\n",  
 " <td>64</td>\n",  
 " <td>0</td>\n",  
 " <td>0</td>\n",  
 " <td>23.3</td>\n",  
 " <td>0.672</td>\n",  
 " <td>32</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>3</th>\n",  
 " <td>1</td>\n",  
 " <td>89</td>\n",  
 " <td>66</td>\n",  
 " <td>23</td>\n",  
 " <td>94</td>\n",  
 " <td>28.1</td>\n",  
 " <td>0.167</td>\n",  
 " <td>21</td>\n",  
 " <td>0</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>4</th>\n",  
 " <td>0</td>\n",  
 " <td>137</td>\n",  
 " <td>40</td>\n",  
 " <td>35</td>\n",  
 " <td>168</td>\n",  
 " <td>43.1</td>\n",  
 " <td>2.288</td>\n",  
 " <td>33</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " </tbody>\n",  
 "</table>\n",  
 "</div>"  
 ],  
 "text/plain": [  
 " Pregnancies Glucose BloodPressure SkinThickness Insulin BMI \\\n",  
 "0 6 148 72 35 0 33.6 \n",  
 "1 1 85 66 29 0 26.6 \n",  
 "2 8 183 64 0 0 23.3 \n",  
 "3 1 89 66 23 94 28.1 \n",  
 "4 0 137 40 35 168 43.1 \n",  
 "\n",  
 " DiabetesPedigreeFunction Age Outcome \n",  
 "0 0.627 50 1 \n",  
 "1 0.351 31 0 \n",  
 "2 0.672 32 1 \n",  
 "3 0.167 21 0 \n",  
 "4 2.288 33 1 "  
 ]  
 },  
 "execution\_count": 2,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "df = pd.read\_csv('hcd.csv')\n",  
 "df.head()"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "eea664b0",  
 "metadata": {},  
 "source": [  
 "Stated that, on the columns below, a value of zero does not make sense and thus indicates missing value:\n",  
 "\n",  
 "1. Glucose\n",  
 "\n",  
 "2. BloodPressure\n",  
 "\n",  
 "3. SkinThickness\n",  
 "\n",  
 "4. Insulin\n",  
 "\n",  
 "5. BMI"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 3,  
 "id": "320661a4",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "#Replacing 0 vales with nan\n",  
 "cols\_with\_null\_as\_zero = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']\n",  
 "df[cols\_with\_null\_as\_zero] = df[cols\_with\_null\_as\_zero].replace(0, np.NaN)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 4,  
 "id": "01f75e12",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "<class 'pandas.core.frame.DataFrame'>\n",  
 "RangeIndex: 768 entries, 0 to 767\n",  
 "Data columns (total 9 columns):\n",  
 " # Column Non-Null Count Dtype \n",  
 "--- ------ -------------- ----- \n",  
 " 0 Pregnancies 768 non-null int64 \n",  
 " 1 Glucose 763 non-null float64\n",  
 " 2 BloodPressure 733 non-null float64\n",  
 " 3 SkinThickness 541 non-null float64\n",  
 " 4 Insulin 394 non-null float64\n",  
 " 5 BMI 757 non-null float64\n",  
 " 6 DiabetesPedigreeFunction 768 non-null float64\n",  
 " 7 Age 768 non-null int64 \n",  
 " 8 Outcome 768 non-null int64 \n",  
 "dtypes: float64(6), int64(3)\n",  
 "memory usage: 54.1 KB\n"  
 ]  
 }  
 ],  
 "source": [  
 "df.info()"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 5,  
 "id": "cfc70f32",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "Pregnancies 0\n",  
 "Glucose 5\n",  
 "BloodPressure 35\n",  
 "SkinThickness 227\n",  
 "Insulin 374\n",  
 "BMI 11\n",  
 "DiabetesPedigreeFunction 0\n",  
 "Age 0\n",  
 "Outcome 0\n",  
 "dtype: int64"  
 ]  
 },  
 "execution\_count": 5,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "df.isnull().sum()"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 6,  
 "id": "13fff82b",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<div>\n",  
 "<style scoped>\n",  
 " .dataframe tbody tr th:only-of-type {\n",  
 " vertical-align: middle;\n",  
 " }\n",  
 "\n",  
 " .dataframe tbody tr th {\n",  
 " vertical-align: top;\n",  
 " }\n",  
 "\n",  
 " .dataframe thead th {\n",  
 " text-align: right;\n",  
 " }\n",  
 "</style>\n",  
 "<table border=\"1\" class=\"dataframe\">\n",  
 " <thead>\n",  
 " <tr style=\"text-align: right;\">\n",  
 " <th></th>\n",  
 " <th>Pregnancies</th>\n",  
 " <th>Glucose</th>\n",  
 " <th>BloodPressure</th>\n",  
 " <th>SkinThickness</th>\n",  
 " <th>Insulin</th>\n",  
 " <th>BMI</th>\n",  
 " <th>DiabetesPedigreeFunction</th>\n",  
 " <th>Age</th>\n",  
 " <th>Outcome</th>\n",  
 " </tr>\n",  
 " </thead>\n",  
 " <tbody>\n",  
 " <tr>\n",  
 " <th>count</th>\n",  
 " <td>768.000000</td>\n",  
 " <td>763.000000</td>\n",  
 " <td>733.000000</td>\n",  
 " <td>541.000000</td>\n",  
 " <td>394.000000</td>\n",  
 " <td>757.000000</td>\n",  
 " <td>768.000000</td>\n",  
 " <td>768.000000</td>\n",  
 " <td>768.000000</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>mean</th>\n",  
 " <td>3.845052</td>\n",  
 " <td>121.686763</td>\n",  
 " <td>72.405184</td>\n",  
 " <td>29.153420</td>\n",  
 " <td>155.548223</td>\n",  
 " <td>32.457464</td>\n",  
 " <td>0.471876</td>\n",  
 " <td>33.240885</td>\n",  
 " <td>0.348958</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>std</th>\n",  
 " <td>3.369578</td>\n",  
 " <td>30.535641</td>\n",  
 " <td>12.382158</td>\n",  
 " <td>10.476982</td>\n",  
 " <td>118.775855</td>\n",  
 " <td>6.924988</td>\n",  
 " <td>0.331329</td>\n",  
 " <td>11.760232</td>\n",  
 " <td>0.476951</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>min</th>\n",  
 " <td>0.000000</td>\n",  
 " <td>44.000000</td>\n",  
 " <td>24.000000</td>\n",  
 " <td>7.000000</td>\n",  
 " <td>14.000000</td>\n",  
 "

<td>18.200000</td>\n",  
 " <td>0.078000</td>\n",  
 " <td>21.000000</td>\n",  
 " <td>0.000000</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>25%</th>\n",  
 " <td>1.000000</td>\n",  
 " <td>99.000000</td>\n",  
 " <td>64.000000</td>\n",  
 " <td>22.000000</td>\n",  
 " <td>76.250000</td>\n",  
 " <td>27.500000</td>\n",  
 " <td>0.243750</td>\n",  
 " <td>24.000000</td>\n",  
 " <td>0.000000</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>50%</th>\n",  
 " <td>3.000000</td>\n",  
 " <td>117.000000</td>\n",  
 " <td>72.000000</td>\n",  
 " <td>29.000000</td>\n",  
 " <td>125.000000</td>\n",  
 " <td>32.300000</td>\n",  
 " <td>0.372500</td>\n",  
 " <td>29.000000</td>\n",  
 " <td>0.000000</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>75%</th>\n",  
 " <td>6.000000</td>\n",  
 " <td>141.000000</td>\n",  
 " <td>80.000000</td>\n",  
 " <td>36.000000</td>\n",  
 " <td>190.000000</td>\n",  
 " <td>36.600000</td>\n",  
 " <td>0.626250</td>\n",  
 " <td>41.000000</td>\n",  
 " <td>1.000000</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>max</th>\n",  
 " <td>17.000000</td>\n",  
 " <td>199.000000</td>\n",  
 " <td>122.000000</td>\n",  
 " <td>99.000000</td>\n",  
 " <td>846.000000</td>\n",  
 " <td>67.100000</td>\n",  
 " <td>2.420000</td>\n",  
 " <td>81.000000</td>\n",  
 " <td>1.000000</td>\n",  
 " </tr>\n",  
 " </tbody>\n",  
 "</table>\n",  
 "</div>"  
 ],  
 "text/plain": [  
 " Pregnancies Glucose BloodPressure SkinThickness Insulin \\\n",  
 "count 768.000000 763.000000 733.000000 541.000000 394.000000 \n",  
 "mean 3.845052 121.686763 72.405184 29.153420 155.548223 \n",  
 "std 3.369578 30.535641 12.382158 10.476982 118.775855 \n",  
 "min 0.000000 44.000000 24.000000 7.000000 14.000000 \n",  
 "25% 1.000000 99.000000 64.000000 22.000000 76.250000 \n",  
 "50% 3.000000 117.000000 72.000000 29.000000 125.000000 \n",  
 "75% 6.000000 141.000000 80.000000 36.000000 190.000000 \n",  
 "max 17.000000 199.000000 122.000000 99.000000 846.000000 \n",  
 "\n",  
 " BMI DiabetesPedigreeFunction Age Outcome \n",  
 "count 757.000000 768.000000 768.000000 768.000000 \n",  
 "mean 32.457464 0.471876 33.240885 0.348958 \n",  
 "std 6.924988 0.331329 11.760232 0.476951 \n",  
 "min 18.200000 0.078000 21.000000 0.000000 \n",  
 "25% 27.500000 0.243750 24.000000 0.000000 \n",  
 "50% 32.300000 0.372500 29.000000 0.000000 \n",  
 "75% 36.600000 0.626250 41.000000 1.000000 \n",  
 "max 67.100000 2.420000 81.000000 1.000000 "  
 ]  
 },  
 "execution\_count": 6,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "df.describe()"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "92f530ae",  
 "metadata": {},  
 "source": [  
 "(2) Visually explore these variables using histograms. Treat the missing values accordingly."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 7,  
 "id": "c2197d3e",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "array([[<Axes: title={'center': 'Pregnancies'}>,\n",  
 " <Axes: title={'center': 'Glucose'}>,\n",  
 " <Axes: title={'center': 'BloodPressure'}>],\n",  
 " [<Axes: title={'center': 'SkinThickness'}>,\n",  
 " <Axes: title={'center': 'Insulin'}>,\n",  
 " <Axes: title={'center': 'BMI'}>],\n",  
 " [<Axes: title={'center': 'DiabetesPedigreeFunction'}>,\n",  
 " <Axes: title={'center': 'Age'}>,\n",  
 " <Axes: title={'center': 'Outcome'}>]], dtype=object)"  
 ]  
 },  
 "execution\_count": 7,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 },  
 {  
 "data": {  
 "image/png": "





















",  
 "text/plain": [  
 "<Figure size 1500x1500 with 9 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "df.hist(bins=9, figsize=(15,15))"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "a350ce54",  
 "metadata": {},  
 "source": [  
 "From above histograms, it is clear that Insulin has highly skewed data distribution and remaining 4 variables have relatively balanced data distribution therefore we will treat missing values in these 5 variables as below:-\n",  
 "\n",  
 "1) Glucose - replace missing values with mean of values.\n",  
 "2) BloodPressure - replace missing values with mean of values.\n",  
 "3) SkinThickness - replace missing values with mean of values.\n",  
 "4) Insulin - replace missing values with median of values.\n",  
 "5) BMI - replace missing values with mean of values."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 8,  
 "id": "06d1ec21",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "df['Insulin'] = df['Insulin'].fillna(df['Insulin'].median())"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 9,  
 "id": "7ac4feb0",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "cols\_mean\_for\_null = ['Glucose', 'BloodPressure', 'SkinThickness', 'BMI']\n",  
 "df[cols\_mean\_for\_null] = df[cols\_mean\_for\_null].fillna(df[cols\_mean\_for\_null].mean())"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "c0fae33d",  
 "metadata": {},  
 "source": [  
 "(3) Create a count (frequency) plot describing the data types and the count of variables:"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 10,  
 "id": "51b2f5c7",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "image/png": "

",  
 "text/plain": [  
 "<Figure size 640x480 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "df.dtypes.value\_counts().plot(kind='bar')\n",  
 "plt.grid()\n",  
 "plt.show()"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "b2bfc064",  
 "metadata": {},  
 "source": [  
 "###### Data Exploration"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "61a8caec",  
 "metadata": {},  
 "source": [  
 "(4) Check the balance of the data by plotting the count of outcomes by their value. Describe your findings and plan future course of action :"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 11,  
 "id": "8935aaff",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0 500\n",  
 "1 268\n",  
 "Name: Outcome, dtype: int64"  
 ]  
 },  
 "execution\_count": 11,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 },  
 {  
 "data": {  
 "image/png": "

",  
 "text/plain": [  
 "<Figure size 640x480 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "df['Outcome'].value\_counts().plot(kind='bar')\n",  
 "df['Outcome'].value\_counts()"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "ccd504fd",  
 "metadata": {},  
 "source": [  
 "Since classes in Outcome is little skewed so we will generate new samples using SMOTE (Synthetic Minority Oversampling Technique) for the class '1' which is under-represented in our data. We will use SMOTE out of many other techniques available since:\n",  
 "\n",  
 "1) It generates new samples by interpolation.\n",  
 "2) It doesn't duplicate data."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 12,  
 "id": "a50fc3b4",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "# Defining independent variables\n",  
 "X = df.drop('Outcome', axis=1)\n",  
 "y = df['Outcome']"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 13,  
 "id": "cf7d97af",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<div>\n",  
 "<style scoped>\n",  
 " .dataframe tbody tr th:only-of-type {\n",  
 " vertical-align: middle;\n",  
 " }\n",  
 "\n",  
 " .dataframe tbody tr th {\n",  
 " vertical-align: top;\n",  
 " }\n",  
 "\n",  
 " .dataframe thead th {\n",  
 " text-align: right;\n",  
 " }\n",  
 "</style>\n",  
 "<table border=\"1\" class=\"dataframe\">\n",  
 " <thead>\n",  
 " <tr style=\"text-align: right;\">\n",  
 " <th></th>\n",  
 " <th>Pregnancies</th>\n",  
 " <th>Glucose</th>\n",  
 " <th>BloodPressure</th>\n",  
 " <th>SkinThickness</th>\n",  
 " <th>Insulin</th>\n",  
 " <th>BMI</th>\n",  
 " <th>DiabetesPedigreeFunction</th>\n",  
 " <th>Age</th>\n",  
 " </tr>\n",  
 " </thead>\n",  
 " <tbody>\n",  
 " <tr>\n",  
 " <th>0</th>\n",  
 " <td>6</td>\n",  
 " <td>148.0</td>\n",  
 " <td>72.0</td>\n",  
 " <td>35.00000</td>\n",  
 " <td>125.0</td>\n",  
 " <td>33.6</td>\n",  
 " <td>0.627</td>\n",  
 " <td>50</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>1</th>\n",  
 " <td>1</td>\n",  
 " <td>85.0</td>\n",  
 " <td>66.0</td>\n",  
 " <td>29.00000</td>\n",  
 " <td>125.0</td>\n",  
 " <td>26.6</td>\n",  
 " <td>0.351</td>\n",  
 " <td>31</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>2</th>\n",  
 " <td>8</td>\n",  
 " <td>183.0</td>\n",  
 " <td>64.0</td>\n",  
 " <td>29.15342</td>\n",  
 " <td>125.0</td>\n",  
 " <td>23.3</td>\n",  
 " <td>0.672</td>\n",  
 " <td>32</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>3</th>\n",  
 " <td>1</td>\n",  
 " <td>89.0</td>\n",  
 " <td>66.0</td>\n",  
 " <td>23.00000</td>\n",  
 " <td>94.0</td>\n",  
 " <td>28.1</td>\n",  
 " <td>0.167</td>\n",  
 " <td>21</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>4</th>\n",  
 " <td>0</td>\n",  
 " <td>137.0</td>\n",  
 " <td>40.0</td>\n",  
 " <td>35.00000</td>\n",  
 " <td>168.0</td>\n",  
 " <td>43.1</td>\n",  
 " <td>2.288</td>\n",  
 " <td>33</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>...</th>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>763</th>\n",  
 " <td>10</td>\n",  
 " <td>101.0</td>\n",  
 " <td>76.0</td>\n",  
 " <td>48.00000</td>\n",  
 " <td>180.0</td>\n",  
 " <td>32.9</td>\n",  
 " <td>0.171</td>\n",  
 " <td>63</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>764</th>\n",  
 " <td>2</td>\n",  
 " <td>122.0</td>\n",  
 " <td>70.0</td>\n",  
 " <td>27.00000</td>\n",  
 " <td>125.0</td>\n",  
 " <td>36.8</td>\n",  
 " <td>0.340</td>\n",  
 " <td>27</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>765</th>\n",  
 " <td>5</td>\n",  
 " <td>121.0</td>\n",  
 " <td>72.0</td>\n",  
 " <td>23.00000</td>\n",  
 " <td>112.0</td>\n",  
 " <td>26.2</td>\n",  
 " <td>0.245</td>\n",  
 " <td>30</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>766</th>\n",  
 " <td>1</td>\n",  
 " <td>126.0</td>\n",  
 " <td>60.0</td>\n",  
 " <td>29.15342</td>\n",  
 " <td>125.0</td>\n",  
 " <td>30.1</td>\n",  
 " <td>0.349</td>\n",  
 " <td>47</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>767</th>\n",  
 " <td>1</td>\n",  
 " <td>93.0</td>\n",  
 " <td>70.0</td>\n",  
 " <td>31.00000</td>\n",  
 " <td>125.0</td>\n",  
 " <td>30.4</td>\n",  
 " <td>0.315</td>\n",  
 " <td>23</td>\n",  
 " </tr>\n",  
 " </tbody>\n",  
 "</table>\n",  
 "<p>768 rows × 8 columns</p>\n",  
 "</div>"  
 ],  
 "text/plain": [  
 " Pregnancies Glucose BloodPressure SkinThickness Insulin BMI \\\n",  
 "0 6 148.0 72.0 35.00000 125.0 33.6 \n",  
 "1 1 85.0 66.0 29.00000 125.0 26.6 \n",  
 "2 8 183.0 64.0 29.15342 125.0 23.3 \n",  
 "3 1 89.0 66.0 23.00000 94.0 28.1 \n",  
 "4 0 137.0 40.0 35.00000 168.0 43.1 \n",  
 ".. ... ... ... ... ... ... \n",  
 "763 10 101.0 76.0 48.00000 180.0 32.9 \n",  
 "764 2 122.0 70.0 27.00000 125.0 36.8 \n",  
 "765 5 121.0 72.0 23.00000 112.0 26.2 \n",  
 "766 1 126.0 60.0 29.15342 125.0 30.1 \n",  
 "767 1 93.0 70.0 31.00000 125.0 30.4 \n",  
 "\n",  
 " DiabetesPedigreeFunction Age \n",  
 "0 0.627 50 \n",  
 "1 0.351 31 \n",  
 "2 0.672 32 \n",  
 "3 0.167 21 \n",  
 "4 2.288 33 \n",  
 ".. ... ... \n",  
 "763 0.171 63 \n",  
 "764 0.340

27 \n",  
 "765 0.245 30 \n",  
 "766 0.349 47 \n",  
 "767 0.315 23 \n",  
 "\n",  
 "[768 rows x 8 columns]"  
 ]  
 },  
 "execution\_count": 13,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "X"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 14,  
 "id": "4cf673e9",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0 1\n",  
 "1 0\n",  
 "2 1\n",  
 "3 0\n",  
 "4 1\n",  
 " ..\n",  
 "763 0\n",  
 "764 0\n",  
 "765 0\n",  
 "766 1\n",  
 "767 0\n",  
 "Name: Outcome, Length: 768, dtype: int64"  
 ]  
 },  
 "execution\_count": 14,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "y"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 15,  
 "id": "667bb46d",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "((768, 8), (768,))"  
 ]  
 },  
 "execution\_count": 15,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "X.shape, y.shape"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 16,  
 "id": "2a59e2f2",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from imblearn.over\_sampling import SMOTE"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 17,  
 "id": "c63e2cff",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "X1, y1 = SMOTE(random\_state=108).fit\_resample(X, y)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 18,  
 "id": "c221ebf6",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "((1000, 8), (1000,))"  
 ]  
 },  
 "execution\_count": 18,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "X1.shape, y1.shape"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 19,  
 "id": "118752bb",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "image/png": "

",  
 "text/plain": [  
 "<Figure size 640x480 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 },  
 {  
 "data": {  
 "text/plain": [  
 "1 500\n",  
 "0 500\n",  
 "Name: Outcome, dtype: int64"  
 ]  
 },  
 "execution\_count": 19,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "y1.value\_counts().plot(kind='bar')\n",  
 "plt.grid()\n",  
 "plt.show()\n",  
 "y1.value\_counts()"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "80b92f62",  
 "metadata": {},  
 "source": [  
 "(5) Create scatter charts between the pair of variables to understand the relationships. Describe your findings:"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 20,  
 "id": "6c7d8449",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<div>\n",  
 "<style scoped>\n",  
 " .dataframe tbody tr th:only-of-type {\n",  
 " vertical-align: middle;\n",  
 " }\n",  
 "\n",  
 " .dataframe tbody tr th {\n",  
 " vertical-align: top;\n",  
 " }\n",  
 "\n",  
 " .dataframe thead th {\n",  
 " text-align: right;\n",  
 " }\n",  
 "</style>\n",  
 "<table border=\"1\" class=\"dataframe\">\n",  
 " <thead>\n",  
 " <tr style=\"text-align: right;\">\n",  
 " <th></th>\n",  
 " <th>Pregnancies</th>\n",  
 " <th>Glucose</th>\n",  
 " <th>BloodPressure</th>\n",  
 " <th>SkinThickness</th>\n",  
 " <th>Insulin</th>\n",  
 " <th>BMI</th>\n",  
 " <th>DiabetesPedigreeFunction</th>\n",  
 " <th>Age</th>\n",  
 " <th>Outcome</th>\n",  
 " </tr>\n",  
 " </thead>\n",  
 " <tbody>\n",  
 " <tr>\n",  
 " <th>0</th>\n",  
 " <td>6</td>\n",  
 " <td>148.000000</td>\n",  
 " <td>72.000000</td>\n",  
 " <td>35.000000</td>\n",  
 " <td>125.000000</td>\n",  
 " <td>33.600000</td>\n",  
 " <td>0.627000</td>\n",  
 " <td>50</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>1</th>\n",  
 " <td>1</td>\n",  
 " <td>85.000000</td>\n",  
 " <td>66.000000</td>\n",  
 " <td>29.000000</td>\n",  
 " <td>125.000000</td>\n",  
 " <td>26.600000</td>\n",  
 " <td>0.351000</td>\n",  
 " <td>31</td>\n",  
 " <td>0</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>2</th>\n",  
 " <td>8</td>\n",  
 " <td>183.000000</td>\n",  
 " <td>64.000000</td>\n",  
 " <td>29.153420</td>\n",  
 " <td>125.000000</td>\n",  
 " <td>23.300000</td>\n",  
 " <td>0.672000</td>\n",  
 " <td>32</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>3</th>\n",  
 " <td>1</td>\n",  
 " <td>89.000000</td>\n",  
 " <td>66.000000</td>\n",  
 " <td>23.000000</td>\n",  
 " <td>94.000000</td>\n",  
 " <td>28.100000</td>\n",  
 " <td>0.167000</td>\n",  
 " <td>21</td>\n",  
 " <td>0</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>4</th>\n",  
 " <td>0</td>\n",  
 " <td>137.000000</td>\n",  
 " <td>40.000000</td>\n",  
 " <td>35.000000</td>\n",  
 " <td>168.000000</td>\n",  
 " <td>43.100000</td>\n",  
 " <td>2.288000</td>\n",  
 " <td>33</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>...</th>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " <td>...</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>995</th>\n",  
 " <td>3</td>\n",  
 " <td>164.686765</td>\n",  
 " <td>74.249021</td>\n",  
 " <td>29.153420</td>\n",  
 " <td>125.000000</td>\n",  
 " <td>42.767110</td>\n",  
 " <td>0.726091</td>\n",  
 " <td>29</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>996</th>\n",  
 " <td>0</td>\n",  
 " <td>138.913540</td>\n",  
 " <td>69.022720</td>\n",  
 " <td>27.713033</td>\n",  
 " <td>127.283849</td>\n",  
 " <td>39.177649</td>\n",  
 " <td>0.703702</td>\n",  
 " <td>24</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>997</th>\n",  
 " <td>10</td>\n",  
 " <td>131.497740</td>\n",  
 " <td>66.331574</td>\n",  
 " <td>33.149837</td>\n",  
 " <td>125.000000</td>\n",  
 " <td>45.820819</td>\n",  
 " <td>0.498032</td>\n",  
 " <td>38</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>998</th>\n",  
 " <td>0</td>\n",  
 " <td>105.571347</td>\n",  
 " <td>83.238205</td>\n",  
 " <td>29.153420</td>\n",  
 " <td>125.000000</td>\n",  
 " <td>27.728596</td>\n",  
 " <td>0.649204</td>\n",  
 " <td>60</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>999</th>\n",  
 " <td>0</td>\n",  
 " <td>127.727025</td>\n",  
 " <td>108.908879</td>\n",  
 " <td>44.468195</td>\n",  
 " <td>129.545366</td>\n",  
 " <td>65.808840</td>\n",  
 " <td>0.308998</td>\n",  
 " <td>26</td>\n",  
 " <td>1</td>\n",  
 " </tr>\n",  
 " </tbody>\n",  
 "</table>\n",  
 "<p>1000 rows × 9 columns</p>\n",  
 "</div>"  
 ],  
 "text/plain": [  
 " Pregnancies Glucose BloodPressure SkinThickness Insulin \\\n",  
 "0 6 148.000000 72.000000 35.000000 125.000000 \n",  
 "1 1 85.000000 66.000000 29.000000 125.000000 \n",  
 "2 8 183.000000 64.000000 29.153420 125.000000 \n",  
 "3 1 89.000000 66.000000 23.000000 94.000000 \n",  
 "4 0 137.000000 40.000000 35.000000 168.000000 \n",  
 ".. ... ... ... ... ... \n",  
 "995 3 164.686765 74.249021 29.153420 125.000000 \n",  
 "996 0 138.913540 69.022720 27.713033 127.283849 \n",  
 "997 10 131.497740 66.331574 33.149837 125.000000 \n",  
 "998 0 105.571347 83.238205 29.153420 125.000000 \n",  
 "999 0 127.727025 108.908879 44.468195 129.545366 \n",  
 "\n",  
 " BMI DiabetesPedigreeFunction Age Outcome \n",  
 "0 33.600000

0.627000 50 1 \n",  
 "1 26.600000 0.351000 31 0 \n",  
 "2 23.300000 0.672000 32 1 \n",  
 "3 28.100000 0.167000 21 0 \n",  
 "4 43.100000 2.288000 33 1 \n",  
 ".. ... ... ... ... \n",  
 "995 42.767110 0.726091 29 1 \n",  
 "996 39.177649 0.703702 24 1 \n",  
 "997 45.820819 0.498032 38 1 \n",  
 "998 27.728596 0.649204 60 1 \n",  
 "999 65.808840 0.308998 26 1 \n",  
 "\n",  
 "[1000 rows x 9 columns]"  
 ]  
 },  
 "execution\_count": 20,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "dfn = pd.concat([X1, y1], axis=1)\n",  
 "dfn"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 21,  
 "id": "65f4134a",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "image/png": "























",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "sns.set(rc={'figure.figsize':(5,5)})\n",  
 "sns.scatterplot(x=\"Pregnancies\", y=\"Glucose\", data=dfn, hue=\"Outcome\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 22,  
 "id": "41bba485",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "image/png": "











































































































































































































































































































",  
 "text/plain": [  
 "<Figure size 1800x1500 with 64 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "fig, axes = plt.subplots(8, 8, figsize=(18, 15))\n",  
 "fig.suptitle('Scatter Plot for Features in Training Data')\n",  
 "\n",  
 "for i, col\_y in enumerate(X1.columns):\n",  
 " for j, col\_x in enumerate(X1.columns): \n",  
 " sns.scatterplot(ax=axes[i, j], x=col\_x, y=col\_y, data=dfn, hue=\"Outcome\", legend = False)\n",  
 "\n",  
 "plt.tight\_layout()"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "ba605050",  
 "metadata": {},  
 "source": [  
 "From the above scatter plot we can see that : \n",  
 "\n",  
 "1. Impressively effective in differentiating between the Outcome groups is glucose alone.\n",  
 "2. A certain degree of class distinction may also be made based just on age.\n",  
 "3. It appears that no pairings in the dataset can clearly differentiate between the various outcome groups.\n",  
 "4. To create a model for the prediction of classes in outcomes, we must combine many factors."  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "0acc1056",  
 "metadata": {},  
 "source": [  
 "(6) Perform correlation analysis. Visually explore it using a heat map :"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 23,  
 "id": "9f6ac0f5",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "image/png": "























",  
 "text/plain": [  
 "<Figure size 1200x800 with 2 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "plt.figure(figsize=(12,8))\n",  
 "sns.heatmap(X1.corr(), annot=True);"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "8f524b23",  
 "metadata": {},  
 "source": [  
 "Significant Correlation Present between the pairs :\n",  
 "1. Age - Pregnancies\n",  
 "2. BMI - Skin Thickness\n",  
 "\n",  
 "No Negative Correlation present."  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "6bcd2aa9",  
 "metadata": {},  
 "source": [  
 "# Week 2 :"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "61cc87cc",  
 "metadata": {},  
 "source": [  
 "### Data Modelling :"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "adac2721",  
 "metadata": {},  
 "source": [  
 "(1) Devise strategies for model building. It is important to decide the right validation framework. Express your thought process. "  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "b03efaab",  
 "metadata": {},  
 "source": [  
 "Answer : From the problem based statement we can conclude that this is a classification based problem, so we will use multiple popular algorithms and compare the accuracy and other parameters.We will be using :\n",  
 "1. Logistic Regression\n",  
 "2. Decision Tree\n",  
 "3. RandomForest Classifier\n",  
 "4. K-Nearest Neighbour (KNN)\n",  
 "5. Support Vector Machine (SVM)\n",  
 "6. Naive Bayes\n",  
 "7. Adaptive Boosting"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 24,  
 "id": "778860d0",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "# Train-Test split\n",  
 "from sklearn.model\_selection import train\_test\_split, KFold"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 25,  
 "id": "3886c50a",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "X\_train, X\_test, y\_train, y\_test = train\_test\_split(X1, y1, test\_size=0.2, random\_state =6)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 26,  
 "id": "d993ded3",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "((800, 8), (200, 8))"  
 ]  
 },  
 "execution\_count": 26,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "X\_train.shape, X\_test.shape"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 27,  
 "id": "7cb69350",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "((800,), (200,))"  
 ]  
 },  
 "execution\_count": 27,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "y\_train.shape, y\_test.shape"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "4312d6b1",  
 "metadata": {},  
 "source": [  
 "(2) Apply an appropriate classification algorithm to build a model. Compare various models with the results from KNN algorithm."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 28,  
 "id": "e9fdee34",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models = []\n",  
 "model\_accuracy = []\n",  
 "model\_f1 = []\n",  
 "model\_auc = []"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "7b24a15b",  
 "metadata": {},  
 "source": [  
 "###### Knn"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 29,  
 "id": "5e61ebeb",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.neighbors import KNeighborsClassifier\n",  
 "knn = KNeighborsClassifier(n\_neighbors=3)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 30,  
 "id": "42fc70f9",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<style>#sk-container-id-1 {color: black;background-color: white;}#sk-container-id-1 pre{padding: 0;}#sk-container-id-1 div.sk-toggleable {background-color: white;}#sk-container-id-1 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-1 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-1 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-1 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-1 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-1 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-1 input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-1 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-1 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-1 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-1 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-1 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-1 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-1 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-1 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-1 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-1 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-1 div.sk-item {position: relative;z-index: 1;}#sk-container-id-1 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-1 div.sk-item::before, #sk-container-id-1 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-1 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-1 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-1 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-1 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-1 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-1 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-1 div.sk-label-container {text-align: center;}#sk-container-id-1 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-1 div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-1\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>KNeighborsClassifier(n\_neighbors=3)</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation

is unable to render, please try loading this page with nbviewer.org.</b></div><div class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-1\" type=\"checkbox\" checked><label for=\"sk-estimator-id-1\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">KNeighborsClassifier</label><div class=\"sk-toggleable\_\_content\"><pre>KNeighborsClassifier(n\_neighbors=3)</pre></div></div></div></div></div>"  
 ],  
 "text/plain": [  
 "KNeighborsClassifier(n\_neighbors=3)"  
 ]  
 },  
 "execution\_count": 30,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "knn.fit(X\_train, y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 31,  
 "id": "a9f1e8a2",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.88625"  
 ]  
 },  
 "execution\_count": 31,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Accuracy score for training dataset\n",  
 "knn.score(X\_train, y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 32,  
 "id": "72a3af20",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.8"  
 ]  
 },  
 "execution\_count": 32,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Accuracy score for testing dataset\n",  
 "knn.score(X\_test, y\_test)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 33,  
 "id": "735f0142",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "AUC: 0.843\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "







",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# ROC(Reciever Operating Characteristics curve)\n",  
 "from sklearn.metrics import roc\_auc\_score\n",  
 "from sklearn.metrics import roc\_curve\n",  
 "from sklearn.metrics import auc\n",  
 "probs = knn.predict\_proba(X\_test) \n",  
 "probs = probs[:, 1] \n",  
 "\n",  
 "auc\_knn = roc\_auc\_score(y\_test, probs) \n",  
 "print('AUC: %.3f' %auc\_knn)\n",  
 "fpr, tpr, thresholds = roc\_curve(y\_test, probs) \n",  
 "plt.plot([0, 1], [0, 1], linestyle='--') \n",  
 "plt.plot(fpr, tpr, marker='.') \n",  
 "plt.xlabel(\"False Positive Rate\")\n",  
 "plt.ylabel(\"True Positive Rate\")\n",  
 "plt.title(\"ROC (Receiver Operating Characteristics) Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 34,  
 "id": "32f7e397",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "f1=0.817 auc\_pr=0.849 ap=0.784\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "





",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# Precision-Recall Curve\n",  
 "from sklearn.metrics import precision\_recall\_curve\n",  
 "from sklearn.metrics import f1\_score\n",  
 "from sklearn.metrics import accuracy\_score\n",  
 "from sklearn.metrics import average\_precision\_score\n",  
 "pred\_test = knn.predict(X\_test) \n",  
 "precision, recall, thresholds = precision\_recall\_curve(y\_test, probs)\n",  
 "f1 = f1\_score(y\_test, pred\_test) \n",  
 "auc\_knn\_pr = auc(recall, precision) \n",  
 "ap = average\_precision\_score(y\_test, probs) \n",  
 "print('f1=%.3f auc\_pr=%.3f ap=%.3f' % (f1, auc\_knn\_pr, ap))\n",  
 "plt.plot([0, 1], [0.5, 0.5], linestyle='--') \n",  
 "plt.plot(recall, precision, marker='.') \n",  
 "plt.xlabel(\"Recall\")\n",  
 "plt.ylabel(\"Precision\")\n",  
 "plt.title(\"Precision-Recall Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 35,  
 "id": "df4a848f",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models.append('KNN')\n",  
 "model\_accuracy.append(accuracy\_score(y\_test, pred\_test))\n",  
 "model\_f1.append(f1)\n",  
 "model\_auc.append(auc\_knn)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "44e62f88",  
 "metadata": {},  
 "source": [  
 "###### Decision Tree Classifier"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 36,  
 "id": "9507ca9d",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.tree import DecisionTreeClassifier\n",  
 "dtc = DecisionTreeClassifier(random\_state=0)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 37,  
 "id": "ebf63a2f",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<style>#sk-container-id-2 {color: black;background-color: white;}#sk-container-id-2 pre{padding: 0;}#sk-container-id-2 div.sk-toggleable {background-color: white;}#sk-container-id-2 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-2 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-2 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-2 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-2 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-2 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-2

input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-2 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-2 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-2 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-2 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-2 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-2 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-2 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-2 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-2 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-2 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-2 div.sk-item {position: relative;z-index: 1;}#sk-container-id-2 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-2 div.sk-item::before, #sk-container-id-2 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-2 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-2 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-2 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-2 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-2 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-2 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-2 div.sk-label-container {text-align: center;}#sk-container-id-2 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-2 div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-2\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>DecisionTreeClassifier(random\_state=0)</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.</b></div><div class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-2\" type=\"checkbox\" checked><label for=\"sk-estimator-id-2\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">DecisionTreeClassifier</label><div class=\"sk-toggleable\_\_content\"><pre>DecisionTreeClassifier(random\_state=0)</pre></div></div></div></div></div>"  
 ],  
 "text/plain": [  
 "DecisionTreeClassifier(random\_state=0)"  
 ]  
 },  
 "execution\_count": 37,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "dtc.fit(X\_train,y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 38,  
 "id": "fba3e78f",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "1.0"  
 ]  
 },  
 "execution\_count": 38,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Training score\n",  
 "dtc.score(X\_train,y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 39,  
 "id": "a34b6497",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.805"  
 ]  
 },  
 "execution\_count": 39,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Testing score\n",  
 "dtc.score(X\_test, y\_test)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 40,  
 "id": "8c2938f4",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "AUC: 0.805\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "







",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# ROC(Reciever Operating Characteristics curve)\n",  
 "probs = dtc.predict\_proba(X\_test) \n",  
 "probs = probs[:, 1] \n",  
 "\n",  
 "auc\_dtc = roc\_auc\_score(y\_test, probs) \n",  
 "print('AUC: %.3f' %auc\_dtc)\n",  
 "fpr, tpr, thresholds = roc\_curve(y\_test, probs) \n",  
 "plt.plot([0, 1], [0, 1], linestyle='--') \n",  
 "plt.plot(fpr, tpr, marker='.') \n",  
 "plt.xlabel(\"False Positive Rate\")\n",  
 "plt.ylabel(\"True Positive Rate\")\n",  
 "plt.title(\"ROC (Receiver Operating Characteristics) Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 41,  
 "id": "f2a686c8",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type":

"stream",  
 "text": [  
 "f1=0.806 auc\_pr=0.856 ap=0.750\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "





",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# Precision-Recall Curve\n",  
 "pred\_test = dtc.predict(X\_test) \n",  
 "precision, recall, thresholds = precision\_recall\_curve(y\_test, probs)\n",  
 "f1 = f1\_score(y\_test, pred\_test) \n",  
 "auc\_dtc\_pr = auc(recall, precision) \n",  
 "ap = average\_precision\_score(y\_test, probs) \n",  
 "print('f1=%.3f auc\_pr=%.3f ap=%.3f' % (f1, auc\_dtc\_pr, ap))\n",  
 "plt.plot([0, 1], [0.5, 0.5], linestyle='--') \n",  
 "plt.plot(recall, precision, marker='.') \n",  
 "plt.xlabel(\"Recall\")\n",  
 "plt.ylabel(\"Precision\")\n",  
 "plt.title(\"Precision-Recall Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 42,  
 "id": "b6b94800",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models.append('DTC')\n",  
 "model\_accuracy.append(accuracy\_score(y\_test, pred\_test))\n",  
 "model\_f1.append(f1)\n",  
 "model\_auc.append(auc\_dtc)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "86259d93",  
 "metadata": {},  
 "source": [  
 "###### Logistic Regression :"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 43,  
 "id": "f3d441f6",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.linear\_model import LogisticRegression\n",  
 "lr = LogisticRegression(max\_iter=300)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 44,  
 "id": "0b2fd76e",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<style>#sk-container-id-3 {color: black;background-color: white;}#sk-container-id-3 pre{padding: 0;}#sk-container-id-3 div.sk-toggleable {background-color: white;}#sk-container-id-3 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-3 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-3 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-3 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-3 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-3 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-3 input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-3 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-3 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-3 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-3 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-3 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-3 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-3 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-3 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-3 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-3 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-3 div.sk-item {position: relative;z-index: 1;}#sk-container-id-3 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-3 div.sk-item::before, #sk-container-id-3 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-3 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-3 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-3 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-3 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-3 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-3 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-3 div.sk-label-container {text-align: center;}#sk-container-id-3 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-3 div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-3\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>LogisticRegression(max\_iter=300)</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.</b></div><div class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-3\" type=\"checkbox\" checked><label for=\"sk-estimator-id-3\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">LogisticRegression</label><div class=\"sk-toggleable\_\_content\"><pre>LogisticRegression(max\_iter=300)</pre></div></div></div></div></div>"  
 ],  
 "text/plain": [  
 "LogisticRegression(max\_iter=300)"  
 ]  
 },  
 "execution\_count": 44,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "lr.fit(X\_train,y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 45,  
 "id": "7d11b7e5",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.73375"  
 ]  
 },  
 "execution\_count": 45,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Train score\n",  
 "lr.score(X\_train,y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 46,  
 "id": "e190f023",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.75"  
 ]  
 },  
 "execution\_count": 46,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Test score\n",  
 "lr.score(X\_test, y\_test)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 47,  
 "id": "5437d6b7",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "AUC: 0.853\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "







",  
 "text/plain": [  
 "<Figure size 500x500

with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# ROC(Reciever Operating Characteristics curve)\n",  
 "probs = lr.predict\_proba(X\_test) \n",  
 "probs = probs[:, 1] \n",  
 "\n",  
 "auc\_lr = roc\_auc\_score(y\_test, probs) \n",  
 "print('AUC: %.3f' %auc\_lr)\n",  
 "fpr, tpr, thresholds = roc\_curve(y\_test, probs) \n",  
 "plt.plot([0, 1], [0, 1], linestyle='--') \n",  
 "plt.plot(fpr, tpr, marker='.') \n",  
 "plt.xlabel(\"False Positive Rate\")\n",  
 "plt.ylabel(\"True Positive Rate\")\n",  
 "plt.title(\"ROC (Receiver Operating Characteristics) Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 48,  
 "id": "6968544c",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "f1=0.752 auc\_pr=0.842 ap=0.843\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "



",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# Precision-Recall Curve\n",  
 "pred\_test

= lr.predict(X\_test) \n",  
 "precision, recall, thresholds = precision\_recall\_curve(y\_test, probs)\n",  
 "f1 = f1\_score(y\_test, pred\_test) \n",  
 "auc\_lr\_pr = auc(recall, precision) \n",  
 "ap = average\_precision\_score(y\_test, probs) \n",  
 "print('f1=%.3f auc\_pr=%.3f ap=%.3f' % (f1, auc\_lr\_pr, ap))\n",  
 "plt.plot([0, 1], [0.5, 0.5], linestyle='--') \n",  
 "plt.plot(recall, precision, marker='.') \n",  
 "plt.xlabel(\"Recall\")\n",  
 "plt.ylabel(\"Precision\")\n",  
 "plt.title(\"Precision-Recall Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 49,  
 "id": "888cd16e",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models.append('LR')\n",  
 "model\_accuracy.append(accuracy\_score(y\_test, pred\_test))\n",  
 "model\_f1.append(f1)\n",  
 "model\_auc.append(auc\_lr)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "6a3a2932",  
 "metadata": {},  
 "source": [  
 "###### Random Forest Classifier"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 50,  
 "id": "3a7907c9",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.ensemble import RandomForestClassifier\n",  
 "rf = RandomForestClassifier()"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 51,  
 "id": "912e4929",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<style>#sk-container-id-4 {color: black;background-color: white;}#sk-container-id-4 pre{padding: 0;}#sk-container-id-4 div.sk-toggleable {background-color: white;}#sk-container-id-4 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-4 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-4 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-4 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-4 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-4 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-4 input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-4 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-4 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-4 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-4 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-4 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-4 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-4 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-4 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-4 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-4 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-4 div.sk-item {position: relative;z-index: 1;}#sk-container-id-4 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-4 div.sk-item::before, #sk-container-id-4 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-4 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-4 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-4 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-4 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-4 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-4 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-4 div.sk-label-container {text-align: center;}#sk-container-id-4 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-4 div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-4\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>RandomForestClassifier()</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.</b></div><div class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-4\" type=\"checkbox\" checked><label for=\"sk-estimator-id-4\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">RandomForestClassifier</label><div class=\"sk-toggleable\_\_content\"><pre>RandomForestClassifier()</pre></div></div></div></div></div>"  
 ],  
 "text/plain": [  
 "RandomForestClassifier()"  
 ]  
 },  
 "execution\_count": 51,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "rf.fit(X\_train, y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 52,  
 "id": "6797bf8a",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "1.0"  
 ]  
 },  
 "execution\_count": 52,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Train score\n",  
 "rf.score(X\_train, y\_train) "  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 53,  
 "id": "2aaa3395",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.83"  
 ]  
 },  
 "execution\_count": 53,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Test score\n",  
 "rf.score(X\_test, y\_test)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 54,  
 "id": "f48d1e13",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "AUC: 0.906\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "









",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# ROC(Reciever Operating Characteristics curve)\n",  
 "probs = rf.predict\_proba(X\_test) \n",  
 "probs = probs[:, 1] \n",  
 "\n",  
 "auc\_rf = roc\_auc\_score(y\_test, probs) \n",  
 "print('AUC: %.3f' %auc\_rf)\n",  
 "fpr, tpr, thresholds = roc\_curve(y\_test, probs) \n",  
 "plt.plot([0, 1], [0, 1], linestyle='--') \n",  
 "plt.plot(fpr, tpr, marker='.') \n",  
 "plt.xlabel(\"False Positive Rate\")\n",  
 "plt.ylabel(\"True Positive Rate\")\n",  
 "plt.title(\"ROC (Receiver Operating Characteristics) Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 55,  
 "id": "342983b2",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "f1=0.840 auc\_pr=0.898 ap=0.897\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "





",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# Precision-Recall Curve\n",  
 "pred\_test = rf.predict(X\_test) \n",  
 "precision, recall, thresholds = precision\_recall\_curve(y\_test, probs)\n",  
 "f1 = f1\_score(y\_test, pred\_test) \n",  
 "auc\_rf\_pr = auc(recall, precision) \n",  
 "ap = average\_precision\_score(y\_test, probs) \n",  
 "print('f1=%.3f auc\_pr=%.3f ap=%.3f' % (f1, auc\_rf\_pr, ap))\n",  
 "plt.plot([0, 1], [0.5, 0.5], linestyle='--') \n",  
 "plt.plot(recall, precision, marker='.') \n",  
 "plt.xlabel(\"Recall\")\n",  
 "plt.ylabel(\"Precision\")\n",  
 "plt.title(\"Precision-Recall Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 56,  
 "id": "4dcebac2",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models.append('RF')\n",  
 "model\_accuracy.append(accuracy\_score(y\_test, pred\_test))\n",  
 "model\_f1.append(f1)\n",  
 "model\_auc.append(auc\_rf)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "3360da06",  
 "metadata": {},  
 "source": [  
 "###### SVM :"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 57,  
 "id": "d70153d8",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.svm import SVC\n",  
 "svm = SVC(probability=True)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 58,  
 "id": "8a6a85dd",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<style>#sk-container-id-5 {color: black;background-color: white;}#sk-container-id-5 pre{padding: 0;}#sk-container-id-5 div.sk-toggleable {background-color: white;}#sk-container-id-5 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-5 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-5 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-5 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-5 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-5 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-5 input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-5 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-5 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-5 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-5 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-5 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-5 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-5 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-5 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-5 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-5 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-5 div.sk-item {position: relative;z-index: 1;}#sk-container-id-5 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-5 div.sk-item::before, #sk-container-id-5 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-5 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-5 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-5 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-5 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-5 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-5 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-5 div.sk-label-container {text-align: center;}#sk-container-id-5 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-5 div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-5\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>SVC(probability=True)</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.</b></div><div class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-5\" type=\"checkbox\" checked><label for=\"sk-estimator-id-5\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">SVC</label><div class=\"sk-toggleable\_\_content\"><pre>SVC(probability=True)</pre></div></div></div></div></div>"  
 ],  
 "text/plain": [  
 "SVC(probability=True)"  
 ]  
 },  
 "execution\_count": 58,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "svm.fit(X\_train, y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 59,  
 "id": "d17c4b57",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.74"  
 ]  
 },  
 "execution\_count": 59,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Train score\n",  
 "svm.score(X\_train, y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 60,  
 "id": "ffbb98b7",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.73"  
 ]  
 },  
 "execution\_count": 60,  
 "metadata":

{},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Test score\n",  
 "svm.score(X\_test, y\_test)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 62,  
 "id": "95a6caa2",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "AUC: 0.827\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "







",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# ROC(Reciever Operating Characteristics curve)\n",  
 "probs = svm.predict\_proba(X\_test) \n",  
 "probs = probs[:, 1] \n",  
 "\n",  
 "auc\_svm = roc\_auc\_score(y\_test, probs) \n",  
 "print('AUC: %.3f' %auc\_svm)\n",  
 "fpr, tpr, thresholds = roc\_curve(y\_test, probs) \n",  
 "plt.plot([0, 1], [0, 1], linestyle='--') \n",  
 "plt.plot(fpr, tpr, marker='.') \n",  
 "plt.xlabel(\"False Positive Rate\")\n",  
 "plt.ylabel(\"True Positive Rate\")\n",  
 "plt.title(\"ROC (Receiver Operating Characteristics) Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 63,  
 "id": "51d4f9b1",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "f1=0.730 auc\_pr=0.810 ap=0.811\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "





",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# Precision-Recall Curve\n",  
 "pred\_test = svm.predict(X\_test) \n",  
 "precision, recall, thresholds = precision\_recall\_curve(y\_test, probs)\n",  
 "f1 = f1\_score(y\_test, pred\_test) \n",  
 "auc\_svm\_pr = auc(recall, precision) \n",  
 "ap = average\_precision\_score(y\_test, probs) \n",  
 "print('f1=%.3f auc\_pr=%.3f ap=%.3f' % (f1, auc\_svm\_pr, ap))\n",  
 "plt.plot([0, 1], [0.5, 0.5], linestyle='--') \n",  
 "plt.plot(recall, precision, marker='.') \n",  
 "plt.xlabel(\"Recall\")\n",  
 "plt.ylabel(\"Precision\")\n",  
 "plt.title(\"Precision-Recall Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 64,  
 "id": "f7de43a6",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models.append('SVM')\n",  
 "model\_accuracy.append(accuracy\_score(y\_test, pred\_test))\n",  
 "model\_f1.append(f1)\n",  
 "model\_auc.append(auc\_svm)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "208923f6",  
 "metadata": {},  
 "source": [  
 "###### Naive Bayes Algorithm :"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 65,  
 "id": "4067a54b",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.naive\_bayes import GaussianNB, BernoulliNB, MultinomialNB\n",  
 "gnb = GaussianNB()"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 66,  
 "id": "a61941a7",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<style>#sk-container-id-6 {color: black;background-color: white;}#sk-container-id-6 pre{padding: 0;}#sk-container-id-6 div.sk-toggleable {background-color: white;}#sk-container-id-6 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-6 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-6 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-6 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-6 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-6 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-6 input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-6 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-6 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-6 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-6 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-6 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-6 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-6 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-6 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-6 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-6 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-6 div.sk-item {position: relative;z-index: 1;}#sk-container-id-6 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-6 div.sk-item::before, #sk-container-id-6 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-6 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-6 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-6 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-6 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-6 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-6 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-6 div.sk-label-container {text-align: center;}#sk-container-id-6 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-6

div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-6\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>GaussianNB()</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.</b></div><div class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-6\" type=\"checkbox\" checked><label for=\"sk-estimator-id-6\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">GaussianNB</label><div class=\"sk-toggleable\_\_content\"><pre>GaussianNB()</pre></div></div></div></div></div>"  
 ],  
 "text/plain": [  
 "GaussianNB()"  
 ]  
 },  
 "execution\_count": 66,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "gnb.fit(X\_train, y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 67,  
 "id": "b7ee4408",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.7275"  
 ]  
 },  
 "execution\_count": 67,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Train score\n",  
 "gnb.score(X\_train, y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 68,  
 "id": "9e133e8e",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.75"  
 ]  
 },  
 "execution\_count": 68,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Test score\n",  
 "gnb.score(X\_test, y\_test)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 69,  
 "id": "072c67d1",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "AUC: 0.857\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "







",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# ROC(Reciever Operating Characteristics curve)\n",  
 "probs = gnb.predict\_proba(X\_test) \n",  
 "probs = probs[:, 1] \n",  
 "\n",  
 "auc\_gnb = roc\_auc\_score(y\_test, probs) \n",  
 "print('AUC: %.3f' %auc\_gnb)\n",  
 "fpr, tpr, thresholds = roc\_curve(y\_test, probs) \n",  
 "plt.plot([0, 1], [0, 1], linestyle='--') \n",  
 "plt.plot(fpr, tpr, marker='.') \n",  
 "plt.xlabel(\"False Positive Rate\")\n",  
 "plt.ylabel(\"True Positive Rate\")\n",  
 "plt.title(\"ROC (Receiver Operating Characteristics) Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 70,  
 "id": "1299abd0",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "f1=0.747 auc\_pr=0.850 ap=0.851\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "





",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# Precision-Recall Curve\n",  
 "pred\_test = gnb.predict(X\_test) \n",  
 "precision, recall, thresholds = precision\_recall\_curve(y\_test, probs)\n",  
 "f1 = f1\_score(y\_test, pred\_test) \n",  
 "auc\_gnb\_pr = auc(recall, precision) \n",  
 "ap = average\_precision\_score(y\_test, probs) \n",  
 "print('f1=%.3f auc\_pr=%.3f ap=%.3f' % (f1, auc\_gnb\_pr, ap))\n",  
 "plt.plot([0, 1], [0.5, 0.5], linestyle='--') \n",  
 "plt.plot(recall, precision, marker='.') \n",  
 "plt.xlabel(\"Recall\")\n",  
 "plt.ylabel(\"Precision\")\n",  
 "plt.title(\"Precision-Recall Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 71,  
 "id": "8fdd0b28",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models.append('GNB')\n",  
 "model\_accuracy.append(accuracy\_score(y\_test, pred\_test))\n",  
 "model\_f1.append(f1)\n",  
 "model\_auc.append(auc\_gnb)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "6ce38816",  
 "metadata": {},  
 "source": [  
 "###### AdaBoost - Adaptive Boosting Classifier :"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 72,  
 "id": "77529f75",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.ensemble import AdaBoostClassifier\n",  
 "abc = AdaBoostClassifier(n\_estimators=100)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 73,  
 "id": "9b5d0a16",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<style>#sk-container-id-7 {color: black;background-color: white;}#sk-container-id-7 pre{padding: 0;}#sk-container-id-7 div.sk-toggleable {background-color: white;}#sk-container-id-7 label.sk-toggleable\_\_label {cursor: pointer;display: block;width: 100%;margin-bottom: 0;padding: 0.3em;box-sizing: border-box;text-align: center;}#sk-container-id-7 label.sk-toggleable\_\_label-arrow:before {content: \"▸\";float: left;margin-right: 0.25em;color: #696969;}#sk-container-id-7 label.sk-toggleable\_\_label-arrow:hover:before {color: black;}#sk-container-id-7 div.sk-estimator:hover label.sk-toggleable\_\_label-arrow:before {color: black;}#sk-container-id-7 div.sk-toggleable\_\_content {max-height: 0;max-width: 0;overflow: hidden;text-align: left;background-color: #f0f8ff;}#sk-container-id-7 div.sk-toggleable\_\_content pre {margin: 0.2em;color: black;border-radius: 0.25em;background-color: #f0f8ff;}#sk-container-id-7 input.sk-toggleable\_\_control:checked~div.sk-toggleable\_\_content {max-height: 200px;max-width: 100%;overflow: auto;}#sk-container-id-7 input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label-arrow:before {content: \"▾\";}#sk-container-id-7 div.sk-estimator input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-7 div.sk-label input.sk-toggleable\_\_control:checked~label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-7 input.sk-hidden--visually {border: 0;clip: rect(1px 1px 1px 1px);clip: rect(1px, 1px, 1px, 1px);height: 1px;margin: -1px;overflow: hidden;padding: 0;position: absolute;width: 1px;}#sk-container-id-7 div.sk-estimator {font-family: monospace;background-color: #f0f8ff;border: 1px dotted black;border-radius: 0.25em;box-sizing: border-box;margin-bottom: 0.5em;}#sk-container-id-7 div.sk-estimator:hover {background-color: #d4ebff;}#sk-container-id-7 div.sk-parallel-item::after {content: \"\";width: 100%;border-bottom: 1px solid gray;flex-grow: 1;}#sk-container-id-7 div.sk-label:hover label.sk-toggleable\_\_label {background-color: #d4ebff;}#sk-container-id-7 div.sk-serial::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: 0;}#sk-container-id-7 div.sk-serial {display: flex;flex-direction: column;align-items: center;background-color: white;padding-right: 0.2em;padding-left: 0.2em;position: relative;}#sk-container-id-7 div.sk-item {position: relative;z-index: 1;}#sk-container-id-7 div.sk-parallel {display: flex;align-items: stretch;justify-content: center;background-color: white;position: relative;}#sk-container-id-7 div.sk-item::before, #sk-container-id-7 div.sk-parallel-item::before {content: \"\";position: absolute;border-left: 1px solid gray;box-sizing: border-box;top: 0;bottom: 0;left: 50%;z-index: -1;}#sk-container-id-7 div.sk-parallel-item {display: flex;flex-direction: column;z-index: 1;position: relative;background-color: white;}#sk-container-id-7 div.sk-parallel-item:first-child::after {align-self: flex-end;width: 50%;}#sk-container-id-7 div.sk-parallel-item:last-child::after {align-self: flex-start;width: 50%;}#sk-container-id-7 div.sk-parallel-item:only-child::after {width: 0;}#sk-container-id-7 div.sk-dashed-wrapped {border: 1px dashed gray;margin: 0 0.4em 0.5em 0.4em;box-sizing: border-box;padding-bottom: 0.4em;background-color: white;}#sk-container-id-7 div.sk-label label {font-family: monospace;font-weight: bold;display: inline-block;line-height: 1.2em;}#sk-container-id-7 div.sk-label-container {text-align: center;}#sk-container-id-7 div.sk-container {/\* jupyter's `normalize.less` sets `[hidden] { display: none; }` but bootstrap.min.css set `[hidden] { display: none !important; }` so we also need the `!important` here to be able to override the default hidden behavior on the sphinx rendered scikit-learn.org. See: https://github.com/scikit-learn/scikit-learn/issues/21755 \*/display: inline-block !important;position: relative;}#sk-container-id-7 div.sk-text-repr-fallback {display: none;}</style><div id=\"sk-container-id-7\" class=\"sk-top-container\"><div class=\"sk-text-repr-fallback\"><pre>AdaBoostClassifier(n\_estimators=100)</pre><b>In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook. <br />On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.</b></div><div

class=\"sk-container\" hidden><div class=\"sk-item\"><div class=\"sk-estimator sk-toggleable\"><input class=\"sk-toggleable\_\_control sk-hidden--visually\" id=\"sk-estimator-id-7\" type=\"checkbox\" checked><label for=\"sk-estimator-id-7\" class=\"sk-toggleable\_\_label sk-toggleable\_\_label-arrow\">AdaBoostClassifier</label><div class=\"sk-toggleable\_\_content\"><pre>AdaBoostClassifier(n\_estimators=100)</pre></div></div></div></div></div>"  
 ],  
 "text/plain": [  
 "AdaBoostClassifier(n\_estimators=100)"  
 ]  
 },  
 "execution\_count": 73,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "abc.fit(X\_train,y\_train)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 74,  
 "id": "df3ce13f",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.8475"  
 ]  
 },  
 "execution\_count": 74,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Train score\n",  
 "abc.score(X\_train,y\_train) "  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 75,  
 "id": "40405fcc",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/plain": [  
 "0.79"  
 ]  
 },  
 "execution\_count": 75,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "# Test score\n",  
 "abc.score(X\_test, y\_test)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 76,  
 "id": "0256109c",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "AUC: 0.870\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "







",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# ROC(Reciever Operating Characteristics curve)\n",  
 "probs = abc.predict\_proba(X\_test) \n",  
 "probs = probs[:, 1] \n",  
 "\n",  
 "auc\_abc = roc\_auc\_score(y\_test, probs) \n",  
 "print('AUC: %.3f' %auc\_abc)\n",  
 "fpr, tpr, thresholds = roc\_curve(y\_test, probs) \n",  
 "plt.plot([0, 1], [0, 1], linestyle='--') \n",  
 "plt.plot(fpr, tpr, marker='.') \n",  
 "plt.xlabel(\"False Positive Rate\")\n",  
 "plt.ylabel(\"True Positive Rate\")\n",  
 "plt.title(\"ROC (Receiver Operating Characteristics) Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 77,  
 "id": "7c5a4cca",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "f1=0.790 auc\_pr=0.823 ap=0.829\n"  
 ]  
 },  
 {  
 "data": {  
 "image/png": "





",  
 "text/plain": [  
 "<Figure size 500x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "# Precision-Recall Curve\n",  
 "pred\_test = abc.predict(X\_test) \n",  
 "precision, recall, thresholds = precision\_recall\_curve(y\_test, probs)\n",  
 "f1 = f1\_score(y\_test, pred\_test) \n",  
 "auc\_abc\_pr = auc(recall, precision) \n",  
 "ap = average\_precision\_score(y\_test, probs) \n",  
 "print('f1=%.3f auc\_pr=%.3f ap=%.3f' % (f1, auc\_abc\_pr, ap))\n",  
 "plt.plot([0, 1], [0.5, 0.5], linestyle='--') \n",  
 "plt.plot(recall, precision, marker='.') \n",  
 "plt.xlabel(\"Recall\")\n",  
 "plt.ylabel(\"Precision\")\n",  
 "plt.title(\"Precision-Recall Curve\");"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 78,  
 "id": "6f6336ac",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "models.append('ABC')\n",  
 "model\_accuracy.append(accuracy\_score(y\_test, pred\_test))\n",  
 "model\_f1.append(f1)\n",  
 "model\_auc.append(auc\_abc)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "aad0d3c9",  
 "metadata": {},  
 "source": [  
 "###### Model comparision :"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "511cf6b9",  
 "metadata": {},  
 "source": [  
 "(3) Compare various models with the results from KNN algorithm."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 79,  
 "id": "4271a135",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "model\_summary = pd.DataFrame(zip(models,model\_accuracy,model\_f1,model\_auc), columns = ['model','accuracy','f1\_score','auc'])\n",  
 "model\_summary = model\_summary.set\_index('model')"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 80,  
 "id": "a1170f17",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "text/html": [  
 "<div>\n",  
 "<style scoped>\n",  
 " .dataframe tbody tr th:only-of-type {\n",  
 " vertical-align: middle;\n",  
 " }\n",  
 "\n",  
 " .dataframe tbody tr th {\n",  
 " vertical-align: top;\n",  
 " }\n",  
 "\n",  
 " .dataframe thead th {\n",  
 " text-align: right;\n",  
 " }\n",  
 "</style>\n",  
 "<table border=\"1\" class=\"dataframe\">\n",  
 " <thead>\n",  
 " <tr style=\"text-align: right;\">\n",  
 " <th></th>\n",  
 " <th>accuracy</th>\n",  
 " <th>f1\_score</th>\n",  
 " <th>auc</th>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>model</th>\n",  
 " <th></th>\n",  
 " <th></th>\n",  
 " <th></th>\n",  
 " </tr>\n",  
 " </thead>\n",  
 " <tbody>\n",  
 " <tr>\n",  
 " <th>KNN</th>\n",  
 " <td>0.800</td>\n",  
 " <td>0.816514</td>\n",  
 " <td>0.843234</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>DTC</th>\n",  
 " <td>0.805</td>\n",  
 " <td>0.805970</td>\n",  
 " <td>0.805031</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>LR</th>\n",  
 " <td>0.750</td>\n",  
 " <td>0.752475</td>\n",  
 " <td>0.852585</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>RF</th>\n",  
 " <td>0.830</td>\n",  
 " <td>0.839623</td>\n",  
 " <td>0.906241</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>SVM</th>\n",  
 " <td>0.730</td>\n",  
 " <td>0.730000</td>\n",  
 " <td>0.827483</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>GNB</th>\n",  
 " <td>0.750</td>\n",  
 " <td>0.747475</td>\n",  
 " <td>0.857486</td>\n",  
 " </tr>\n",  
 " <tr>\n",  
 " <th>ABC</th>\n",  
 " <td>0.790</td>\n",  
 " <td>0.790000</td>\n",  
 " <td>0.870387</td>\n",  
 " </tr>\n",  
 " </tbody>\n",  
 "</table>\n",  
 "</div>"  
 ],  
 "text/plain": [  
 " accuracy f1\_score auc\n",  
 "model \n",  
 "KNN 0.800 0.816514 0.843234\n",  
 "DTC 0.805 0.805970 0.805031\n",  
 "LR 0.750 0.752475 0.852585\n",  
 "RF 0.830 0.839623 0.906241\n",  
 "SVM 0.730 0.730000 0.827483\n",  
 "GNB 0.750 0.747475 0.857486\n",  
 "ABC 0.790 0.790000 0.870387"  
 ]  
 },  
 "execution\_count": 80,  
 "metadata": {},  
 "output\_type": "execute\_result"  
 }  
 ],  
 "source": [  
 "model\_summary"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 81,  
 "id": "31513a98",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "image/png": "





















AAAAcAvCRwAAAAAAAABuQfgIAAAAAAAAwC3+H85b+mWN8KU/AAAAAElFTkSuQmCC",  
 "text/plain": [  
 "<Figure size 1600x500 with 1 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "model\_summary.plot(figsize=(16,5))\n",  
 "plt.title(\"Comparison of Different Classification Algorithms\");"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "e4eb237e",  
 "metadata": {},  
 "source": [  
 "By comparing the values and by looking at the graph we can see that the RandomForest Classifier model performed the best out of all the models and as such we will make our final model based on that."  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "d1ddc5bc",  
 "metadata": {},  
 "source": [  
 "(4) Create a classification report by analyzing sensitivity, specificity, AUC (ROC curve), etc."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 82,  
 "id": "64c69345",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "fm = rf"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 84,  
 "id": "76e40cda",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "from sklearn.metrics import classification\_report\n",  
 "report = classification\_report(y\_test, fm.predict(X\_test))"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 85,  
 "id": "8c2722bb",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 " precision recall f1-score support\n",  
 "\n",  
 " 0 0.87 0.78 0.82 99\n",  
 " 1 0.80 0.88 0.84 101\n",  
 "\n",  
 " accuracy 0.83 200\n",  
 " macro avg 0.83 0.83 0.83 200\n",  
 "weighted avg 0.83 0.83 0.83 200\n",  
 "\n"  
 ]  
 }  
 ],  
 "source": [  
 "print(report)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 86,  
 "id": "16d46db2",  
 "metadata": {},  
 "outputs": [  
 {  
 "data": {  
 "image/png": "



",  
 "text/plain": [  
 "<Figure size 500x500 with 2 Axes>"  
 ]  
 },  
 "metadata": {},  
 "output\_type": "display\_data"  
 }  
 ],  
 "source": [  
 "from sklearn.metrics import confusion\_matrix, ConfusionMatrixDisplay\n",  
 "cm = confusion\_matrix(y\_test, fm.predict(X\_test))\n",  
 "ConfusionMatrixDisplay(cm).plot()\n",  
 "plt.show()"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 88,  
 "id": "3a2c5399",  
 "metadata": {},  
 "outputs": [],  
 "source": [  
 "TP = cm[1,1]\n",  
 "TN = cm[0,0]\n",  
 "FP = cm[0,1]\n",  
 "FN = cm[1,0]\n",  
 "\n",  
 "Accuracy = (TP+TN)/(TP+TN+FP+FN)\n",  
 "Precision = TP/(TP+FP)\n",  
 "Sensitivity = TP/(TP+FN)\n",  
 "Specificity = TN/(TN+FP)"  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": 89,  
 "id": "fb99a2a2",  
 "metadata": {},  
 "outputs": [  
 {  
 "name": "stdout",  
 "output\_type": "stream",  
 "text": [  
 "Accuracy: 0.830\n",  
 "Precision: 0.802\n",  
 "Sensitivity: 0.881\n",  
 "Specificity: 0.778\n",  
 "AUC: 0.906\n"  
 ]  
 }  
 ],  
 "source": [  
 "print(\"Accuracy: %.3f\"%Accuracy)\n",  
 "print(\"Precision: %.3f\"%Precision)\n",  
 "print(\"Sensitivity: %.3f\"%Sensitivity)\n",  
 "print(\"Specificity: %.3f\"%Specificity)\n",  
 "print(\"AUC: %.3f\"%auc\_rf)"  
 ]  
 },  
 {  
 "cell\_type": "markdown",  
 "id": "9aaf0bf8",  
 "metadata": {},  
 "source": [  
 "Description of the variables :\n",  
 "1. TP : True Positive\n",  
 "2. TN : True Negative\n",  
 "3. FP : False Positive\n",  
 "4. FN : False Negative\n",  
 "5. cm : Confusion Matrix\n",  
 "\n",  
 "Note : Sensitivity Score is also called Recall score."  
 ]  
 },  
 {  
 "cell\_type": "code",  
 "execution\_count": null,  
 "id": "72e46b60",  
 "metadata": {},  
 "outputs": [],  
 "source": []  
 }  
 ],  
 "metadata": {  
 "kernelspec": {  
 "display\_name": "Python 3 (ipykernel)",  
 "language": "python",  
 "name": "python3"  
 },  
 "language\_info": {  
 "codemirror\_mode": {  
 "name": "ipython",  
 "version": 3  
 },  
 "file\_extension": ".py",  
 "mimetype": "text/x-python",  
 "name": "python",  
 "nbconvert\_exporter": "python",  
 "pygments\_lexer": "ipython3",  
 "version": "3.11.3"  
 }  
 },  
 "nbformat": 4,  
 "nbformat\_minor": 5  
}