SPIRNT-3

TEAM ID	PNT2022TMID17480
Project Title	Early Detection Of Chronic Kidney Disease using Machine learning

{

"cells": [

{

"cell_type": "markdown",

"metadata": {

"_cell_guid": "ef22745a-3680-441c-952a-7ce07e28ef3a",

"_uuid": "78003d4f6d7b9dc5c6b298422dfe6e0e7601a24f"

},

"source": [

"# Predicting Chronic Kidney Disease based on health records\n",

"Given 24 health related attributes taken in 2-month period of 400 patients, using the information of the 158 patients with complete records to predict the outcome (i.e. whether one has chronic kidney disease) of the remaining 242 patients (with missing values in their records).\n",

"\n"

]

```
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "_cell_guid": "b0692cea-7400-461d-a8af-c29231368f02",
   "_uuid":
"92c521a749de6c280b92828423b40db416fe1241"
  },
  "source": [
   "## Load Modules and helper functions"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 2,
  "metadata": {
   "_cell_guid": "1a35d647-79b7-40a8-86aa-adc4e5bb33cf",
   "_uuid": "e80657e1dba9c8011e3c89ff215e50b2793bf637"
  },
  "outputs": [],
  "source": [
   "import numpy as np\n",
```

```
    "import pandas as pd\n",
    "import seaborn as sns\n",
    "import matplotlib.pyplot as plt\n",
    "from sklearn.model_selection import train_test_split, GridSearchCV\n",
    "from sklearn.metrics import roc_curve, auc, confusion_matrix, classification_report,accuracy_score\n",
    "from sklearn.ensemble import RandomForestClassifier\n",
    "import warnings\n",
    "warnings.filterwarnings('ignore')\n",
    "\n",
    "\n",
    "# from subprocess import check_output\n",
    "# print(check_output([\"ls\", \"../input\"]).decode(\"utf8\"))"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "metadata": {},
   "outputs": [],
```

```
  "source": [
   "%matplotlib inline\n",
   "\n",
   "def auc_scorer(clf, X, y, model): # Helper function to plot the ROC curve\n",
   "    if model=='RF':\n",
   "        fpr, tpr, _ = roc_curve(y, clf.predict_proba(X)[:,1])\n",
   "    elif model=='SVM':\n",
   "        fpr, tpr, _ = roc_curve(y, clf.decision_function(X))\n",
   "    roc_auc = auc(fpr, tpr)\n",
   "\n",
   "    plt.figure()    # Plot the ROC curve\n",
   "    plt.plot(fpr, tpr, label='ROC curve from '+model+' model (area = %0.3f)' % roc_auc)\n",
   "    plt.plot([0, 1], [0, 1], 'k--')\n",
   "    plt.xlim([0.0, 1.0])\n",
   "    plt.ylim([0.0, 1.05])\n",
   "    plt.xlabel('False Positive Rate')\n",
   "    plt.ylabel('True Positive Rate')\n",
   "    plt.title('ROC Curve')\n",
   "    plt.legend(loc=\"lower right\")\n",
```

```
   "    plt.show()\n",
   "\n",
   "    return fpr,tpr,roc_auc"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "_cell_guid": "68be8886-4e23-4f5d-a35b-e9c4ab3df8de",
   "_uuid": "7c0340215ef5df788ab038b09acb0d4a0478ca9e"
  },
  "source": [
   "## Load files"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 4,
  "metadata": {
   "_cell_guid": "2795e5b4-a0e0-4ca6-9374-37b554a6115b",
   "_uuid":
"7411d96d0f393dd253f6dc6aa8d6a24a220647c4"
```

```
  },
  "outputs": [],
  "source": [
   "df = pd.read_csv(\"C:/Users/Sinegalatha/Desktop/2nd year online class/nalaiya thiran/dataset/kidney_disease.csv\")"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 5,
  "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>id</th>\n",
"      <th>age</th>\n",
"      <th>bp</th>\n",
"      <th>sg</th>\n",
"      <th>al</th>\n",
"      <th>su</th>\n",
"      <th>rbc</th>\n",
"      <th>pc</th>\n",
"      <th>pcc</th>\n",
"      <th>ba</th>\n",
```

```
"      <th>...</th>\n",
"      <th>pcv</th>\n",
"      <th>wc</th>\n",
"      <th>rc</th>\n",
"      <th>htn</th>\n",
"      <th>dm</th>\n",
"      <th>cad</th>\n",
"      <th>appet</th>\n",
"      <th>pe</th>\n",
"      <th>ane</th>\n",
"      <th>classification</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>0</td>\n",
"      <td>48.0</td>\n",
"      <td>80.0</td>\n",
"      <td>1.020</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
```

```
"      <td>NaN</td>\n",
"      <td>normal</td>\n",
"      <td>notpresent</td>\n",
"      <td>notpresent</td>\n",
"      <td>...</td>\n",
"      <td>44</td>\n",
"      <td>7800</td>\n",
"      <td>5.2</td>\n",
"      <td>yes</td>\n",
"      <td>yes</td>\n",
"      <td>no</td>\n",
"      <td>good</td>\n",
"      <td>no</td>\n",
"      <td>no</td>\n",
"      <td>ckd</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>1</td>\n",
"      <td>7.0</td>\n",
"      <td>50.0</td>\n",
"      <td>1.020</td>\n",
```

```
"      <td>4.0</td>\n",
"      <td>0.0</td>\n",
"      <td>NaN</td>\n",
"      <td>normal</td>\n",
"      <td>notpresent</td>\n",
"      <td>notpresent</td>\n",
"      <td>...</td>\n",
"      <td>38</td>\n",
"      <td>6000</td>\n",
"      <td>NaN</td>\n",
"      <td>no</td>\n",
"      <td>no</td>\n",
"      <td>no</td>\n",
"      <td>good</td>\n",
"      <td>no</td>\n",
"      <td>no</td>\n",
"      <td>ckd</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>2</td>\n",
"      <td>62.0</td>\n",
```

```
"      <td>80.0</td>\n",
"      <td>1.010</td>\n",
"      <td>2.0</td>\n",
"      <td>3.0</td>\n",
"      <td>normal</td>\n",
"      <td>normal</td>\n",
"      <td>notpresent</td>\n",
"      <td>notpresent</td>\n",
"      <td>...</td>\n",
"      <td>31</td>\n",
"      <td>7500</td>\n",
"      <td>NaN</td>\n",
"      <td>no</td>\n",
"      <td>yes</td>\n",
"      <td>no</td>\n",
"      <td>poor</td>\n",
"      <td>no</td>\n",
"      <td>yes</td>\n",
"      <td>ckd</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
```

```
"     <td>3</td>\n",
"     <td>48.0</td>\n",
"     <td>70.0</td>\n",
"     <td>1.005</td>\n",
"     <td>4.0</td>\n",
"     <td>0.0</td>\n",
"     <td>normal</td>\n",
"     <td>abnormal</td>\n",
"     <td>present</td>\n",
"     <td>notpresent</td>\n",
"     <td>...</td>\n",
"     <td>32</td>\n",
"     <td>6700</td>\n",
"     <td>3.9</td>\n",
"     <td>yes</td>\n",
"     <td>no</td>\n",
"     <td>no</td>\n",
"     <td>poor</td>\n",
"     <td>yes</td>\n",
"     <td>yes</td>\n",
"     <td>ckd</td>\n",
"   </tr>\n",
```

```
"    <tr>\n",
"      <th>4</th>\n",
"      <td>4</td>\n",
"      <td>51.0</td>\n",
"      <td>80.0</td>\n",
"      <td>1.010</td>\n",
"      <td>2.0</td>\n",
"      <td>0.0</td>\n",
"      <td>normal</td>\n",
"      <td>normal</td>\n",
"      <td>notpresent</td>\n",
"      <td>notpresent</td>\n",
"      <td>...</td>\n",
"      <td>35</td>\n",
"      <td>7300</td>\n",
"      <td>4.6</td>\n",
"      <td>no</td>\n",
"      <td>no</td>\n",
"      <td>no</td>\n",
"      <td>good</td>\n",
"      <td>no</td>\n",
"      <td>no</td>\n",
```

```
      "      <td>ckd</td>\n",
      "    </tr>\n",
      "  </tbody>\n",
      "</table>\n",
      "<p>5 rows × 26 columns</p>\n",
      "</div>"
     ],
     "text/plain": [
      "   id   age    bp    sg   al   su    rbc        pc         pcc\
ba  \\\n",
      "0   0  48.0  80.0  1.020  1.0  0.0     NaN    normal\
notpresent  notpresent   \n",
      "1   1   7.0  50.0  1.020  4.0  0.0     NaN    normal\
notpresent  notpresent   \n",
      "2   2  62.0  80.0  1.010  2.0  3.0  normal    normal\
notpresent  notpresent   \n",
      "3   3  48.0  70.0  1.005  4.0  0.0  normal  abnormal\
present  notpresent   \n",
      "4   4  51.0  80.0  1.010  2.0  0.0  normal    normal\
notpresent  notpresent   \n",
      "\n",
      "  ...  pcv    wc   rc  htn   dm  cad appet   pe  ane\
classification  \n",
```

```
      "0 ... 44 7800 5.2 yes yes no good no no ckd \n",
      "1 ... 38 6000 NaN no no no good no no ckd \n",
      "2 ... 31 7500 NaN no yes no poor no yes ckd \n",
      "3 ... 32 6700 3.9 yes no no poor yes yes ckd \n",
      "4 ... 35 7300 4.6 no no no good no no ckd \n",
      "\n",
      "[5 rows x 26 columns]"
     ]
    },
    "execution_count": 5,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "df.head()\n"
  ]
 },
```

```
{
 "cell_type": "code",
 "execution_count": 6,
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "0      7800\n",
     "1      6000\n",
     "2      7500\n",
     "3      6700\n",
     "4      7300\n",
     "       ... \n",
     "395    6700\n",
     "396    7800\n",
     "397    6600\n",
     "398    7200\n",
     "399    6800\n",
     "Name: wc, Length: 400, dtype: object"
    ]
   },
```

```
   "execution_count": 6,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df['wc']"
 ]
},
{
 "cell_type": "code",
 "execution_count": 7,
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "<class 'pandas.core.frame.DataFrame'>\n",
    "RangeIndex: 400 entries, 0 to 399\n",
    "Data columns (total 26 columns):\n",
    " #   Column          Non-Null Count  Dtype  \n",
```

"--- ------ -------------- ----- \n",

" 0 id 400 non-null int64 \n",

" 1 age 391 non-null float64\n",

" 2 bp 388 non-null float64\n",

" 3 sg 353 non-null float64\n",

" 4 al 354 non-null float64\n",

" 5 su 351 non-null float64\n",

" 6 rbc 248 non-null object \n",

" 7 pc 335 non-null object \n",

" 8 pcc 396 non-null object \n",

" 9 ba 396 non-null object \n",

" 10 bgr 356 non-null float64\n",

" 11 bu 381 non-null float64\n",

" 12 sc 383 non-null float64\n",

" 13 sod 313 non-null float64\n",

" 14 pot 312 non-null float64\n",

" 15 hemo 348 non-null float64\n",

" 16 pcv 330 non-null object \n",

" 17 wc 295 non-null object \n",

" 18 rc 270 non-null object \n",

" 19 htn 398 non-null object \n",

" 20 dm 398 non-null object \n",

```
    " 21  cad             398 non-null    object \n",
    " 22  appet           399 non-null    object \n",
    " 23  pe              399 non-null    object \n",
    " 24  ane             399 non-null    object \n",
    " 25  classification  400 non-null    object \n",
    "dtypes: float64(11), int64(1), object(14)\n",
    "memory usage: 81.4+ KB\n"
   ]
  }
 ],
 "source": [
  "df.info()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 8,
 "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>id</th>\n",
"      <th>age</th>\n",
"      <th>bp</th>\n",
"      <th>sg</th>\n",
```

```
"      <th>al</th>\n",
"      <th>su</th>\n",
"      <th>bgr</th>\n",
"      <th>bu</th>\n",
"      <th>sc</th>\n",
"      <th>sod</th>\n",
"      <th>pot</th>\n",
"      <th>hemo</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>count</th>\n",
"      <td>400.000000</td>\n",
"      <td>391.000000</td>\n",
"      <td>388.000000</td>\n",
"      <td>353.000000</td>\n",
"      <td>354.000000</td>\n",
"      <td>351.000000</td>\n",
"      <td>356.000000</td>\n",
"      <td>381.000000</td>\n",
"      <td>383.000000</td>\n",
```

```
"      <td>313.000000</td>\n",
"      <td>312.000000</td>\n",
"      <td>348.000000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>mean</th>\n",
"      <td>199.500000</td>\n",
"      <td>51.483376</td>\n",
"      <td>76.469072</td>\n",
"      <td>1.017408</td>\n",
"      <td>1.016949</td>\n",
"      <td>0.450142</td>\n",
"      <td>148.036517</td>\n",
"      <td>57.425722</td>\n",
"      <td>3.072454</td>\n",
"      <td>137.528754</td>\n",
"      <td>4.627244</td>\n",
"      <td>12.526437</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>std</th>\n",
"      <td>115.614301</td>\n",
```

```
"     <td>17.169714</td>\n",
"     <td>13.683637</td>\n",
"     <td>0.005717</td>\n",
"     <td>1.352679</td>\n",
"     <td>1.099191</td>\n",
"     <td>79.281714</td>\n",
"     <td>50.503006</td>\n",
"     <td>5.741126</td>\n",
"     <td>10.408752</td>\n",
"     <td>3.193904</td>\n",
"     <td>2.912587</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>min</th>\n",
"     <td>0.000000</td>\n",
"     <td>2.000000</td>\n",
"     <td>50.000000</td>\n",
"     <td>1.005000</td>\n",
"     <td>0.000000</td>\n",
"     <td>0.000000</td>\n",
"     <td>22.000000</td>\n",
"     <td>1.500000</td>\n",
```

```
"      <td>0.400000</td>\n",
"      <td>4.500000</td>\n",
"      <td>2.500000</td>\n",
"      <td>3.100000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>25%</th>\n",
"      <td>99.750000</td>\n",
"      <td>42.000000</td>\n",
"      <td>70.000000</td>\n",
"      <td>1.010000</td>\n",
"      <td>0.000000</td>\n",
"      <td>0.000000</td>\n",
"      <td>99.000000</td>\n",
"      <td>27.000000</td>\n",
"      <td>0.900000</td>\n",
"      <td>135.000000</td>\n",
"      <td>3.800000</td>\n",
"      <td>10.300000</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>50%</th>\n",
```

```
"     <td>199.500000</td>\n",
"     <td>55.000000</td>\n",
"     <td>80.000000</td>\n",
"     <td>1.020000</td>\n",
"     <td>0.000000</td>\n",
"     <td>0.000000</td>\n",
"     <td>121.000000</td>\n",
"     <td>42.000000</td>\n",
"     <td>1.300000</td>\n",
"     <td>138.000000</td>\n",
"     <td>4.400000</td>\n",
"     <td>12.650000</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>75%</th>\n",
"     <td>299.250000</td>\n",
"     <td>64.500000</td>\n",
"     <td>80.000000</td>\n",
"     <td>1.020000</td>\n",
"     <td>2.000000</td>\n",
"     <td>0.000000</td>\n",
"     <td>163.000000</td>\n",
```

```
"     <td>66.000000</td>\n",
"     <td>2.800000</td>\n",
"     <td>142.000000</td>\n",
"     <td>4.900000</td>\n",
"     <td>15.000000</td>\n",
"   </tr>\n",
"   <tr>\n",
"     <th>max</th>\n",
"     <td>399.000000</td>\n",
"     <td>90.000000</td>\n",
"     <td>180.000000</td>\n",
"     <td>1.025000</td>\n",
"     <td>5.000000</td>\n",
"     <td>5.000000</td>\n",
"     <td>490.000000</td>\n",
"     <td>391.000000</td>\n",
"     <td>76.000000</td>\n",
"     <td>163.000000</td>\n",
"     <td>47.000000</td>\n",
"     <td>17.800000</td>\n",
"   </tr>\n",
" </tbody>\n",
```

```
      "</table>\n",
      "</div>"
     ],
     "text/plain": [
      "              id         age          bp          sg          al          su  \\\n",
      "count  400.000000  391.000000  388.000000  353.000000  354.000000  351.000000   \n",
      "mean   199.500000   51.483376   76.469072    1.017408    1.016949    0.450142   \n",
      "std    115.614301   17.169714   13.683637    0.005717    1.352679    1.099191   \n",
      "min      0.000000    2.000000   50.000000    1.005000    0.000000    0.000000   \n",
      "25%     99.750000   42.000000   70.000000    1.010000    0.000000    0.000000   \n",
      "50%    199.500000   55.000000   80.000000    1.020000    0.000000    0.000000   \n",
      "75%    299.250000   64.500000   80.000000    1.020000    2.000000    0.000000   \n",
      "max    399.000000   90.000000  180.000000    1.025000    5.000000    5.000000   \n",
      "\n",
```

"   bgr   bu   sc   sod   pot   hemo \n",
"count 356.000000 381.000000 383.000000 313.000000 312.000000 348.000000 \n",
"mean   148.036517   57.425722   3.072454 137.528754   4.627244   12.526437 \n",
"std   79.281714   50.503006   5.741126   10.408752   3.193904   2.912587 \n",
"min   22.000000   1.500000   0.400000   4.500000   2.500000   3.100000 \n",
"25%   99.000000   27.000000   0.900000 135.000000   3.800000   10.300000 \n",
"50%   121.000000   42.000000   1.300000 138.000000   4.400000   12.650000 \n",
"75%   163.000000   66.000000   2.800000 142.000000   4.900000   15.000000 \n",
"max   490.000000 391.000000   76.000000 163.000000   47.000000   17.800000 "
]
},
"execution_count": 8,
"metadata": {},
"output_type": "execute_result"
}

```
 ],
 "source": [
  "df.describe()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 9,
 "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>id</th>\n",
"      <th>age</th>\n",
"      <th>bp</th>\n",
"      <th>sg</th>\n",
"      <th>al</th>\n",
"      <th>su</th>\n",
"      <th>rbc</th>\n",
"      <th>pc</th>\n",
"      <th>pcc</th>\n",
"      <th>ba</th>\n",
"      <th>...</th>\n",
"      <th>pcv</th>\n",
"      <th>wc</th>\n",
```

```
"      <th>rc</th>\n",
"      <th>htn</th>\n",
"      <th>dm</th>\n",
"      <th>cad</th>\n",
"      <th>appet</th>\n",
"      <th>pe</th>\n",
"      <th>ane</th>\n",
"      <th>classification</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"  </tbody>\n",
"</table>\n",
"<p>0 rows × 26 columns</p>\n",
"</div>"
],
"text/plain": [
 "Empty DataFrame\n",
 "Columns: [id, age, bp, sg, al, su, rbc, pc, pcc, ba, bgr, bu, sc, sod, pot, hemo, pcv, wc, rc, htn, dm, cad, appet, pe, ane, classification]\n",
 "Index: []\n",
```

```
    "\n",
    "[0 rows x 26 columns]"
   ]
  },
  "execution_count": 9,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "df[df.duplicated()]"
]
},
{
 "cell_type": "code",
 "execution_count": 10,
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "id                  0\n",
```

"age 9\n",

"bp 12\n",

"sg 47\n",

"al 46\n",

"su 49\n",

"rbc 152\n",

"pc 65\n",

"pcc 4\n",

"ba 4\n",

"bgr 44\n",

"bu 19\n",

"sc 17\n",

"sod 87\n",

"pot 88\n",

"hemo 52\n",

"pcv 70\n",

"wc 105\n",

"rc 130\n",

"htn 2\n",

"dm 2\n",

"cad 2\n",

"appet 1\n",

```
    "pe                1\n",
    "ane                1\n",
    "classification     0\n",
    "dtype: int64"
   ]
  },
  "execution_count": 10,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "df.isna().sum()"
]
},
{
 "cell_type": "code",
 "execution_count": 12,
 "metadata": {},
 "outputs": [],
 "source": [
  "df2 = df.dropna(axis=0)"
```

```
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {
    "_cell_guid": "7d1d9de5-6b08-4317-ac17-b3d6da8379cf",
    "_uuid": "5be7dfb54962791ae37d47cf089cc699b54fbae2"
   },
   "source": [
    "## Cleaning and preprocessing of data for training a classifier"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {
    "_cell_guid": "d26cdb91-8988-455a-9c16-cf98b901a82a",
    "_uuid": "d673c87769a249511ff4143f186d74482e201516"
   },
   "outputs": [],
   "source": [
```

```
    "# Map text to 1/0 and do some cleaning\n",
    "df[['htn','dm','cad','pe','ane']] = df[['htn','dm','cad','pe','ane']].replace(to_replace={'yes':1,'no':0})\n",
    "df[['rbc','pc']] = df[['rbc','pc']].replace(to_replace={'abnormal':1,'normal':0})\n",
    "df[['pcc','ba']] = df[['pcc','ba']].replace(to_replace={'present':1,'notpresent':0})\n",
    "df[['appet']] = df[['appet']].replace(to_replace={'good':1,'poor':0,'no':np.nan})\n",
    "df['classification'] = df['classification'].replace(to_replace={'ckd':1.0,'ckd\\t':1.0,'notckd':0.0,'no':0.0})\n",

"df.rename(columns={'classification':'class'},inplace=True)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 10,
   "metadata": {
    "_cell_guid": "ad80743d-6471-4871-a817-c6cf96f73f3b",
```

```
   "_uuid": "c40044cea7ae9b4ab210b54ccc2c3c8f37783287"
  },
  "outputs": [],
  "source": [
   "# Further cleaning\n",
   "df['pe'] = df['pe'].replace(to_replace='good',value=0) # Not having pedal edema is good\n",
   "df['appet'] = df['appet'].replace(to_replace='no',value=0)\n",
   "df['cad'] = df['cad'].replace(to_replace='\\tno',value=0)\n",
   "df['dm'] = df['dm'].replace(to_replace={'\\tno':0,'\\tyes':1,' yes':1, '':np.nan})\n",
   "df.drop('id',axis=1,inplace=True)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 11,
  "metadata": {
   "_cell_guid": "10731633-cd50-41ce-9c95-a60dc8ac8e8f",
   "_uuid": "775c0d2c2c49b847ab698964a033e5f124fa42d3"
```

```
},
"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>age</th>\n",
"     <th>bp</th>\n",
"     <th>sg</th>\n",
"     <th>al</th>\n",
"     <th>su</th>\n",
"     <th>rbc</th>\n",
"     <th>pc</th>\n",
"     <th>pcc</th>\n",
"     <th>ba</th>\n",
"     <th>bgr</th>\n",
"     <th>...</th>\n",
"     <th>pcv</th>\n",
"     <th>wc</th>\n",
"     <th>rc</th>\n",
"     <th>htn</th>\n",
"     <th>dm</th>\n",
"     <th>cad</th>\n",
"     <th>appet</th>\n",
"     <th>pe</th>\n",
"     <th>ane</th>\n",
"     <th>class</th>\n",
```

```
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>48.0</td>\n",
"      <td>80.0</td>\n",
"      <td>1.020</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>NaN</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>121.0</td>\n",
"      <td>...</td>\n",
"      <td>44</td>\n",
"      <td>7800</td>\n",
"      <td>5.2</td>\n",
"      <td>1.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
```

```
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>7.0</td>\n",
"      <td>50.0</td>\n",
"      <td>1.020</td>\n",
"      <td>4.0</td>\n",
"      <td>0.0</td>\n",
"      <td>NaN</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>NaN</td>\n",
"      <td>...</td>\n",
"      <td>38</td>\n",
"      <td>6000</td>\n",
"      <td>NaN</td>\n",
"      <td>0.0</td>\n",
```

```
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>62.0</td>\n",
"      <td>80.0</td>\n",
"      <td>1.010</td>\n",
"      <td>2.0</td>\n",
"      <td>3.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>423.0</td>\n",
"      <td>...</td>\n",
"      <td>31</td>\n",
"      <td>7500</td>\n",
```

```
"      <td>NaN</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>3</th>\n",
"      <td>48.0</td>\n",
"      <td>70.0</td>\n",
"      <td>1.005</td>\n",
"      <td>4.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>117.0</td>\n",
"      <td>...</td>\n",
```

```
"      <td>32</td>\n",
"      <td>6700</td>\n",
"      <td>3.9</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>1.0</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>4</th>\n",
"      <td>51.0</td>\n",
"      <td>80.0</td>\n",
"      <td>1.010</td>\n",
"      <td>2.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
```

```
      "      <td>106.0</td>\n",
      "      <td>...</td>\n",
      "      <td>35</td>\n",
      "      <td>7300</td>\n",
      "      <td>4.6</td>\n",
      "      <td>0.0</td>\n",
      "      <td>0.0</td>\n",
      "      <td>0.0</td>\n",
      "      <td>1.0</td>\n",
      "      <td>0.0</td>\n",
      "      <td>0.0</td>\n",
      "      <td>1.0</td>\n",
      "    </tr>\n",
      "  </tbody>\n",
      "</table>\n",
      "<p>5 rows × 25 columns</p>\n",
      "</div>"
     ],
     "text/plain": [
      "    age    bp    sg    al    su    rbc    pc    pcc    ba    bgr ...  pcv  wc  \\\n",
```

```
    "0  48.0  80.0  1.020  1.0  0.0  NaN  0.0  0.0  0.0  121.0  ...
44  7800   \n",
    "1   7.0  50.0  1.020  4.0  0.0  NaN  0.0  0.0  0.0   NaN  ...
38  6000   \n",
    "2  62.0  80.0  1.010  2.0  3.0  0.0  0.0  0.0  0.0  423.0  ...
31  7500   \n",
    "3  48.0  70.0  1.005  4.0  0.0  0.0  1.0  1.0  0.0  117.0  ...
32  6700   \n",
    "4  51.0  80.0  1.010  2.0  0.0  0.0  0.0  0.0  0.0  106.0  ...
35  7300   \n",
    "\n",
    "    rc  htn   dm  cad appet   pe  ane  class  \n",
    "0  5.2  1.0  1.0  0.0   1.0  0.0  0.0    1.0  \n",
    "1  NaN  0.0  0.0  0.0   1.0  0.0  0.0    1.0  \n",
    "2  NaN  0.0  1.0  0.0   0.0  0.0  1.0    1.0  \n",
    "3  3.9  1.0  0.0  0.0   0.0  1.0  1.0    1.0  \n",
    "4  4.6  0.0  0.0  0.0   1.0  0.0  0.0    1.0  \n",
    "\n",
    "[5 rows x 25 columns]"
   ]
  },
  "execution_count": 11,
  "metadata": {},
```

```
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df.head()"
 ]
},
{
 "cell_type": "markdown",
 "metadata": {
  "_cell_guid": "042af755-5ec1-4e4d-8923-c14e743195c7",
  "_uuid": "cd05059199eacedc1d7fdaa950c85189977a8914"
 },
 "source": [
  "## Check the portion of rows with NaN\n",
  "- Now the data is cleaned with improper values labelled NaN. Let's see how many NaNs are there.\n",
  "- Drop all the rows with NaN values, and build a model out of this dataset (i.e. df2)"
 ]
},
{
```

```
  "cell_type": "code",
  "execution_count": 11,
  "metadata": {
   "_cell_guid": "e4242e89-614a-4a6f-bd45-2f82351b5d04",
   "_uuid": "3d98f3adfcf97d96b3bc9ff37b4ab8c4c13d3dbb"
  },
  "outputs": [
   {
    "ename": "NameError",
    "evalue": "name 'df2' is not defined",
    "output_type": "error",
    "traceback": [
     "\u001b[1;31m---------------------------------------------------------------------\u001b[0m",
     "\u001b[1;31mNameError\u001b[0m Traceback (most recent call last)",
     "\u001b[1;32m<ipython-input-11-0875a9e648e4>\u001b[0m in \u001b[0;36m<module>\u001b[1;34m\u001b[0m\n\u001b[1;32m----> 1\u001b[1;33m \u001b[0mdf2\u001b[0m\u001b[1;33m[\u001b[0m\u001b[1;34m'class'\u001b[0m\u001b[1;33m]\u001b[0m\u001b[1;33m.\u001b[0m\u001b[0mvalue_counts\u001b[0m\u001b[1;33m(\
```

u001b[0m\u001b[1;33m)\u001b[0m\u001b[1;33m\u001b[0m\u001b[1;33m\u001b[0m\u001b[0m\n\u001b[0m",

"\u001b[1;31mNameError\u001b[0m: name 'df2' is not defined"

]

}

],

"source": [

"\n",

"df2['class'].value_counts()"

]

},

{

"cell_type": "markdown",

"metadata": {

"_cell_guid": "f6ad991c-eb2a-4910-9809-642635149e04",

"_uuid": "a0d03c254efeec4bcf94ad55c854a83ca85dfe27"

},

"source": [

"## Examine correlations between different features"

]

},

{
"cell_type": "code",
"execution_count": 13,
"metadata": {
"_cell_guid": "19b64021-bcdf-4f88-913b-962390628f8d",
"_uuid":
"4943b12c51e4561e285a6272ce2180d0601bb6ae"
},
"outputs": [
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAmUAAAIrCAYAAA
Cu3e47AAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAAL
EgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cG
xvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwbG
90bGliLm9yZy+WH4yJAAAgAElEQVR4nOzdeZxkVX3//9
cbUNmGibuOgOOCqODGouCKLAqIOkb9IjGKRp2gUYz
LL2rcUGPUxIjGlXZJiChREAmisogCgrI0CAMIgsIoCKK4
wLC4zuf3R93Wsu2e6Z65VX2r+/V8PO6jb9176nOnz
nn1LmpKiRJkjS3NpjrBCRJkmRRJkmS1AkWWZZIkSR1gU
SZJktQBFmWSJEkdYFEmSZLUARZlUp8kL0hyxno8/tJ
Dmwzp7Vcb7ck1wzrenMtydIklWiV5vGfvd9J/iXXJDUl+0jx
RpKrk9yc5BFzlzfcWNt7Jal7LMrUONUn+Jsl484f0uuaPy
nOq/JkhyS5Ij+Y1W1T1UdPlc5cZ8KOj63+8kWwGvAR5c
VfdomrvxlmbSlJXLiYV5zTWb62UyymeTbjHhkll6+Jmx
VfdomrwXeHlVbV5xlmbklOTfLiYV5zTWb62UyyMsme

w8hJ0p+zKFOnJHk18H7gX4G7A1sDHwGevg6xNprJMc0b
9wZ+XlU/nXTsknUJ1rXPStfymcoo5Ch1WlW5uXViAxYD
NwPPXkObO9Ar2q5ttvcDd2jO7QZcA7wO+AnwaeAQ4Gj
gCOAm4MXNdT4JXAf8GPgXYMMmxguAM/qu9wHg6ua
55wGPa47vDfwW+F2T84XN8VOBFzf7GwBvAn4I/BT4H2
Bxc24pUMCBwI+AG4A39l33kcB4c93rgfdN835MvOZ/bmK
sBJ476f16b3ON64GPAZsAmwG3Aaub/G8GljTH7tI8903A7
4Etmsf/Arx/TXH7rrsfcAHwK+BbwEP7zq0EXgusAG4EPgd
sPM3r27C5zg3AlcA/NO/bRv3vN7DnpNdzZPOzgFuAHzTtl
wBfAH4GXAUc3HetQ1iHz0qT3y+bePs0594J/AH4dZPHh6
Z4bROfgeX0PsvXAa9Zj3xm9F71xX8JcCmwCvgusAO9/2Z
WN+/lzcA/NW2fRq+4/VUT50GTfp+va36fvwE2ah7/uIn9P
WCPuf7/i5vbKGxznoCb28RGr9D5/cQfkWnavB04C7gbcFd
6f/Df0ZzbrXn+e+gVDZs0f9h+ByyjVyRtAhwLHEavMLkbc
A7w902MF/DnRdnfAndu/tC8hl6xt3Fz7hDgiEn5/fEPH/B3w
PeB+wKbA8cAn27OTfxB/niT08OaP2gPas5/G3hes785sMs0
78fEa35f85qfQK8I2bY5/37gOOBOwCLgS8C7+p57zaR4pw
PPbPZPAn7AnwqN04FnzCDuDvSK0EfRKxQObP5wTxTP
K5v3fEnz/EuBg6Z5fQcBlwFbNW2/wTSFxjSvp4D7N/sb0Cu
s3wLcvvm9XAk8ue/3OdvPyu/oFTcbAi+lV1xlcm7TvLaJz8
CRTfyH0CsW91zHfGbzXj2bXtG0MxDg/sC9+34/e/bl+QB6
n6m9gNsB/0Tvc337vvYXNNfdBNiW3j9klvS9zvvN9f9f3Nx
GYZvzBNzcJjbgucBP1tLmB8C+fY+fDKxs9nej13u1cd/5Q4
DT+x7fnV7x09+rcwDwjWb/BfQVZVNc/5fAw/pir6koOwV4
Wd+5bZs/shv1/UHesu/8OcBzmv3TgbfR9FqtIZ/d6BVlm/Ud
+zzw5uaP7S39fxCBXYGr+p47uYh5B/CfTY4/AV4JvJvJvBvYm
KYXbQZxP0pTKPed/x7whGZ/JfC3fef+DfjQV6bM
CTWPei7FHAjyadfwPwX+vxWfl+37lNm+vdY3Ju07y2ic/A
Aye9F59cx3xm816dCLxymrxW8udF2ZuBz/c93oBeQbdbX/
u/6zt/f3pF+Z7A7Wby376bm1tvc/xfXfJz4C5JNqqq30/TZgm

94cAJP2yOTfhZVf160nOu7tu/N71/7V+XZOLYBpPa/FGS1
9AbNlpC7w/cFvQKk5mYKteN6P1xnfCTvv1b6fWKAbyIX
q/gZUmuAt5WVcdPc51fVtUtk66zhF5P4qbAeX2vNfR6daZ
zGr1etx2Ai4CT6Q2X7UKvALkhyd3WEvfewIFJXtEX9/b8+
e9p8uvuP9dvCX/+u/nhNO1m4t7AkiS/6ju2IfDNvsez/az88X
VU1a1Nu82Zncmv7yHrmM9s3qut6P0DZyb+7HNcVauTX
A3ca6o8q+r7Sf6RXlG5XZITgVdX1bUzvJ60YFmUqUu+T
W8OzjJ6c2mmci1/Pnl76+bYhJriOf3HrqbX23CXNRR+AC
R5HL25MXsAlzR/jH5JrwCZ7lpT5Tpha3q9WtcDW67piVV
1BXBAkg2AvwaOTnLnScXXhDsm2azv3NbAxfTmFt0GbF
dVP57qMlMc+xa9Hr1nAKdV1XeTbA08hV7BxgziXg28s6re
uabXOEPX0SsgJmy9HrGuptebt80a2qzTZ2UGsdZkK3rDjr
Dmz/Pa8pnNe3U1cL9pzk3O+1r6CsX0KsKt6PWWTfmcqvo
s8NkkW9Abbn0P8Lw15CMJv32pDqmqG+nN9/lwkmVJNk
1yuyT7JPm3ptmRwJuS3DXJXZr2R0wXc4prXEdvrtR/JNk
iyQZJ7pfkCVM0X0SviPoZsFGSt9DrKZtwPbC0KZymciT
wqiT3SbI5vW+Ufm4mf+CT/G2Su1bVanqTq6E3cXw6b0ty
+6aQ3A84qnnux4FDm94tktwryZP78r9zksUTQarqVnrzrv6
BPxVh3wL+fuLxDOJ+HDgoyaPSs1mSpyRZtLbXPYXPAw
cn2TLJHYHXr0OMCecANyV5XZJNkmyYZPskO0/VeJafl
alcT2/e2tq8ufmsbwe8kN4XH9Yln9m8V58AXptkx+Z3dP8k
E/+AmJz354GnJNkjye3oza38Db3PxV9Ism2S3ZPcgd4/sm5j
zZ9dSQ2LMnVKVb0PeDW9b/79jN6/6F9Ob4Iz9L5tNk7vm
14XAec3x2bj+fSG075Lb47Y0cA9p2h3IvBV4HJ6wze/5s+Hh
45qfv48yflTPP9T9L7Ndjq9b+b9GnjFFO2msjdwSZKb6X0
D9DlTDMtO+EnzOq4FPkNvXtFEz8vr6E3KPivJTcDX6PW
E0bQ5Ergyya+STAwhnkZvmOycvseLmtfBDOKO05v8/qE
mr+/Tm3+1Lj5O7/dwIb3f9THrGIeq+gPwVODh9H4fN9Ar
Thav4Wkz/axM5QPAs5L8Msl/rqHdafTeo1OA91bVSeuYz4
zfq6o6it43RD9L7xuSx9L7cgDAu+j9w+dXSV5bVd+j94WX

D9J7z54KPLWqfjtN+DvQm4d4A73P5t3ofTtY0lpMfEtIkjR
ESZbSKw5vt47Do5LmGXvKJEmSOsCiTJIkqQMcvpQkSe
oAe8okSZI6wKJMkiSpA0Zt8VjHWiVJ6qasvYnWxJ4ySZ
KkDrAokyRJ6gCLMkmSpA6wKJMkSeoAizJJkqQOsCiTJ
EnqAIsySZKkDrAokyRJ6oBWi7IkxyY5L8klSZY3x16U5PI
kpyb5eJIPNcfvmuQLSc5ttse0mYskSdIoafWG5EnuVFW/S
LIJcC7wZOBMYAdgFfB14MKqenmSzwIfqaozkmwNnFhV
D1rLJVzRX5KkbnJF//XU9vDlwUkuBM4CtgKeB5xWVb+o
qt8BR/W13RP4UJILgOOALZIsmhwwyfIk40nGx8bGWk5
XkiSpG1q792WS3egVWrtW1a1JTgW+B0zX+7VB0/a2NcW
tqjFgohqzp0ySJM1LbfaULQZ+2RRkDwR2ATYFnpDkjkk
2Ap7Z1/4k4OUTD5I8vMVcJEmSRkqbRdkJwEZJVgDvoD
eE+WPgX4Gzga8B3wVubNofDOyUZEWS7wIHtZiLJEnSS
Gl1ov+UF0g2r6qbm56yLwKfqqovrmM4hy8lSeomJ/qvp2Gs
U3ZIM5n/YuAq4NghXFOSJGmkDLynrGUjlawkSQuIPW
XryRX9JUmSOsCiTJIkqQMsyiRJkjrAokySJkKkDLMokSZ
I6oLXbLA3Lb394davxbn/vrVqNJ0mStC7sKZMkSeoAizJJ
kqQOsCiTJEnqAIsySZKkDrAokyRJ6gCLMkmSpA4YeFG
WZGmSiwd9HUmSpFFmT5kkSVIHDKso2yjJ4UlWJDk6y
aZJViZ5T5Jzmu3+Q8pFkiSpc4ZVlG0LjFXVQ4GbgJc1x2+
qqkcCHwLeP9UTkyxPMp5kfGxsbDjZSpIkDVmqarAXSJ
YCp1fV1s3j3YGDgYcDu1fVlUluB/ykqu68lnDlbZYkSeqkz
HUCo25YPWWTK7+a4vhgq0NJkqQOG1ZRtnWSXZv9A4
Azmv39+35+e0i5SJIkdc6wirJLgQOTrADuBHTrADuBHy
DrwReNaRcJEmSOmfgc8qmvXCyEtipqm6YxdOcUyZJUjc
5p2w9uU6ZJElSB2w0VxeuqqVzdW1JkqSusadMkSpA
JEmSOsCiTJIkqQPm7NuXX62ikkpUkaQHx25frac4m+q+rk
1Zc3mq8Jz30AQCce+U1rcYF2Pm+W7YeU5IkzU8OX0qSJ
HWARZkkSVIHWJRJkiR1gEWZJElSB1iUSZIkdYBFmSRJ
UgdYlEmSJHWARZkkSVIHWJRJkiR1gEWZJElSB1iUSZIk
dYBFmSRJUgdYlEmSJHWARZkkSVIHWJRJkiR1gEWZJElSB
1iUSZIkdYBFmSRJUgdYlEmSJHWARZkkSVIHWJRJkiR1gE
WZJElSB1iUSZIkdYBFmSRJUgdYlEmSJHWARZkkSVIHDG
Xx2CSbAZ8HtgQ2BN4

BrALeB9wAnA/ct6r2G0Y+kiRJXTOsFf33Bq6tqqcAJFkM
XAw8vqquSnLkkPKQJEnqpGENX14E7JnkPUkeB9wHuL
KqrmrOT1uUJVmeZDzJ+NjY2DBylSRJGrqh9JRV1eVJdg
T2Bd4FnDyL544BE9VYtX3vS0mSpC4Y1pyyJcAvquqIJDc
DLwXum2RpVa0E9h9GHpIkSV01rDllDwH+Pclq4Hf0irJ7
AickuQE4Z0h5SJIkddKwhi9PBE7sP5Zk86p6YJIAHwbGh
5GLJElSF83lOmUvSXIBcAmwGDhsDnORJEmaU8Mavvvw
LVXUocOhcXV+SJKlLXNFfkiSpAyzKJEmSOsCiTJIkqQ
NSVXOdw2yMVLKSJC0gmesERt2cTfRfV6tO+nqr8RY9a
XcAbjz2+FbjAixeth8HfODTrcc98pXPaz2mJEmaWw5fSpI
kdYBFmSRJUgdYlEmSJHWARZkkSVIHWJRJkiR1gEW
ZJElSB3SiKEuyMsld5joPSZKudKJokySJGmhG3pRluTY
JOcluSTJ8mFfX5IkqYvmYkX/v6uqXyTZBDg3yRfmIAdJk
qROmYvhy4OTXBCVDXb9gR2rapbk5wKbLym51TVGDBRjVXb9
76UJEnqgmH3lC0GftkUZA8Edhny9SVJkjpp2ExZCBGS
VYA76A3hClJkrTgDXX4sqp+A+wzxamlw8xDkiSpa1ynTJI
kqQMsyiRJkjrAokySJKkDLMokSZI6wKJMkiSpA1JVc53
DbIxUspIkLSCZ6wRGnT1lkiRJHTAXNyRfL7ec0e56s5s9t
ndTgVWnnNZqXIBFezyBD514RutxX/7kx3Ljsce3Hnfxsv1a
jylJkmbGnjJJkqQOsCiTJEnqAIsySZKkDrAokyRJ6gCLM
kmSpA6wKJMkSeoAizJJkqQOGEpRlmSzJF9OcmGSi5Ps
n2Rlkrs053dKcuowcpEkSeqiYfWU7QGsqqpFcW1UPq6rtgROGd
F1JkqSRMKyi7CJgzTvSfAzYGxAaY
oSZI0d4Zym6WqujzC+wLuSnAT8nj8VhRuv4bljwEQ1
Vm3fZkmSJKkLhjWnbAlwa1UAbwX2AFYCezNHnmM
PKQJEnqqmHdkPwhwL8nWQ38DngpsAnwsAnwyST/DJw9pD
wkSZI6aVjDlycCJ05x6x6gHDuL4kSVLXuU6ZJElSB1iUSZIk
dYBFmSRJUgdYlEmSJHWARZkkSVIHpKrmOofZGKlkJUlaQ
UlaQDLXCYw6e8okSZI6YFiLx7bmprMvaD3mHTZ9BMfkzWbg/ZDoAbjz2

+1bgAi5ftx3uO+3rrcV/3tN0Hli/AqpPazXnRk3ZvNZ4kSfO
RPWWSJEkdYFEmSZLUARZlkiRJHWBRJkmS1AEWZZ
IkSR1gUSZJktQBQynKktw8jOtIkiSNqoEXZUkyjOtIkiSNs
oEUS0mWJrk0yUeA84FNkvxHkvOTnJLkrk27+yf5WpILm
3P3G0Q+kiRJXTfIFf23BV5YVS9LUsD5VfWaJG8B3gq8H
PgM8O6q+mKSjZmiSEyyHFgOcNhhh/G8XR8zwJQlSdKo
ufHY42d1b+zFy/br5H06B1mU/bCqzmr2VwOfa/aPAI5Jsgi4
V1V9EaCqfj1VkKoaA8YmHrZ9myVJkqQuGGRRdssazhX
eTV6SJLVhg/kxdX1Yr2ID4FnN/t8AZ1TVTcA1SZYBJLlD
kk2HlI8kSZovssHsto4aVma3ANslOQ/YHXh7c/x5wMFJVg
DfAu4xpHwkSZI6ZSDDl1W1Eti+7/Hmze6bJ7W7gl6RJkm
StE56q2+NvkHOKZMkSRq8Dbs7JDkb8+NVSJIkjTh7yiRJ
0mjr8OT92bAokyRJIy0bzI85ZfOjtJQkSRpxqZrVnQnm2k
glK0nSAjJn3VWrTv7GrOqDRXs9sZNdayM3fHnrOee1Gm
/TR+7Yi3vu+a3GBdh05x142SePbj3uR170LG489vjW4y5eth
9A67EHFbc/tiRpAXNFf0mSJLVl5HrKJEmS+rl4rCRJUhc
4fClJkqS22FMmSZJG2zwZvrSnTJIkjbYNMrttBpLsneR7S
b6f5PVTnH9ukhXN9q0kD1vvl7G+ASRJkuaTJBsCHwb2A
R4MHJDkwZOaXQU8oaoeCrwDGFvf6zp8KUmSRlrav/flI4
HvV9WVvfj5X+DpwHcnGlTVt/ranwVsub4XHUpPWZKlS
S5LcnjTzXd0kk2T7Nx0+V2Y5Jwki4aRjyRJmkeS2W1rdy/g
6r7H1zTHpvMi4Kvr8QqA4Q5fbguMNd18NwEvBz4HvLK
qHgbsCdw2+UlJlicZTzI+NrbePYOSJGmB668tmm355CZT
PG3KWzkleSK9oux165vXMIcvr66qM5v9I4A3AtdV1bkAV
XXTVE+qqjH+NE5bbd9mSZIkjbgZTt6fMKm2mMo1wFZ
9j7cErp3cKMlDgU8A+1TVz2eVxBSG2VM2ucK8aYpjkiRJ
s5MNZret3bnANknuk+T2wHOA4/T2wHOA4/7sksnWwDHA86rq
xjCLsq2T7NrsH0BvUtySJDsDJFmUxC8eSJKkWckGmdW
2NlX1e3rTrE4ELgU+X1WXJDkoyUcANcY+kuSCJO
Pr+zqGWQRdChyY5DDgCuCDwNeBBDybZhN58sj2Bm4eY

kyRJ0l+oqq8AX5l07GN9+y8GXtzmNYdZlK2uqoMmHTsX
2GWIOUiSpPnGe19KkiSpLUPpKauqlcD2w7iWJElaWDJP
esqcWC9JkkbbPCnK5serkCRJGnH2lEmSpNE2s1sndV6q
Rmr91pFKVpKkBWTOKqPbLrpkVvXBJg/ZrpNV3Mj1lP
360u+1Gm/jB20LwC1nnt1qXIDNHvMoPvDVb7Ye95X7PI4
bjz2+9biLl+0HwK3nnt9q3E133qEXd/w7rcYF2HSnRwz0vZ
AkaVhGriiTJEnql5ndOqnzLMokSdJomydzyuZHaSlJkjTi7
CmTJEmjbQY3GR8FFmWSJGm0OXwpSZKkktthTJkmSR
prfvpQkSeqCeTKnrNXSMsnSJJclOTzJiiRHJ9k0yc5JvpXk
wiTnJFmUZMMk701yUdP2FW3mIkmSNEoG0VO2LfCiqj
ozyaeAlwMHAftX1blJtgBuA5YD9wEeUVW/T3KnAeQiSZ
Lmuw3mx/DlIF7F1VV1ZrN/BPBk4LqqOhegm6qqt8DewI
fa/apql9MFSzJ8iTjScbHxsYGkK4kSRplSWa1ddUgesom3x
T0JuAOU7TLFG3/MljVGDBRjVXb976UJEkjzp6yaW2dZ
Ndm/wDgLGBJkp0BmvlkGwKnAAdNDHU6fClJkjBFG
WXAgcmWQHcCfggsD/wwSQ/XAicDGwOf/80A
cpEkSfNdMrutowYxfLm6qg6adOxcYJcp2r662SRJktZNhw
ut2XCdMkmSNNIyT9Ypa7Uoq6qVwjebKi//x
4FZIkSSPO4UtJkjTanFMmSZLUAfNkTlmq1rp+a5eMVL
KSJC0gc1YZ/e4n18+qPrjdPe7eySrOnjJJkjTSMk8m+o9cUf
a7a37carzbbXkvAG4d/06rcQE23ekR/MsxJ7ce901/vRc3Hnt
863EXL9sPoPXYE3Fv+mr778UW+wzuvRjkeyxJatE8Gb6c
H6WlJEnSiBu5njJJkqR+t218h1m1XzWWSJEkd
YFEmSZI0SZK9k3wvyfeTvH6K80uSfDXJ90nkWN9rWpRJ
kiT1SbIh8GFgH+DBwAFJHjyp2d8BJ1bVI+A4gemqj+
qVJLh70dSRJklrySOD7VXVf0W/g6Zrp+o5C/i
rJPdcn4vaGJkvTn7gVc3ff4mubYbNvMyrCKso2SHN6M
uR6dZNMkb0lybbpKLk4wl8+TGVZIkqdOSLE8y8y3rctn9ckiq
dNvmvATNrMyrCKsm2Bsap6KHAT8DlcLgQ1W1c1VtD2w
CuKqmJEkaUoaq6qd+raxCSLO06tJmVYRl

V1fVmc3+EcBjgScmOTvJRcDuwHZTPbG/mh0bm/yeSZIk
te5cYJsk90lye+A5wHGT2hwHPL/5FuYuwI1Vdd36XHRYi
8dO7s4r4CPATlV1dZJDgI2nfGKvep2oxqrt2yxJkiT1q6rfJ3
k5cCKwIfCpqrokyUHN+Y8BXwH2Bb4P3Aq8cH2vO6yibO
sku1bVt4EDgDOARwM3JNkceBZw9JBykSRJWqOq+gq9
wqv/2Mf69gv4hzavOayi7FLgwCSHAVfQW8vjjsBFwEp63
YSSJEkL1sCLsqpaSW/htcne1GySJEkLnjcklyRJI+13G95u
rlNohYvHSpIkdYA9ZZIkaaTVei3Z2h0WZZIkaaStnidVmc
OXkiRJHZAarepypJKVJGkBmbN7WP901a2zqg/utmjTTt5
v2+FLSZI00kasg2laI1eU3Xjs8a3GW7ysdx/0W84abzUuwGa
77MRbjzqh9bhve/berb8P8Kf3YlDv8aByHlTcVSd9vfW4i56
0+0B/d5Kk0TVyRZkkSVK/+TLR36JMkiSNtHlSk/ntS0mSp
C6wp0ySJI00J/pLkiR1wOp5smJW68OXSZYmubjtuJIkSV
OpqlltXdWpOWVJ7LmTJEkL0qCKoI2SHA48ArgceD6wG
/A+4AbgfOC+VbVfkkOAJcDS5tzfDCgnSZI0D7kkxpptC7y
oqs5M8ing1cDfA4+vqquSHDmp/Y7AY6vqtgHlI0mS5qnVq
+dHUTao4curq+rMZv8IYCfgyqq6qjk2uSg7brqCLMnyJO
NJxsfGxgaUriRJ0twaVE/Z5JJ18Vra3zJtoKoxYKIaq0Hcok
aSJI2ueTJ6ObCesq2T7NrsHwB8DbhvkqXNsf0HdF1JkrTA
zJdvXw6qp+xS4MAkhwFXAK8EVgAnJLkBOGdA15UkS
QvMfFmnrPWirKpWAg+efDzJN6rqgUkCfBgYb9of0nYOk
iRp4ehy79dsDHOdspckuQC4hN4cs8OGeG1JkqROG9pirV
V1KHDosK4nSZIWhvnSU+YK+pIkaaTNk2XKunWbJUm
SpIXKnjJJkjTSHL6UJEnqgPlSlGXEXshIJStJ0gKSubrwxd
dcP6v6YPst7z5nua7JyPWU3XLWeKvxNttlJwAGcfumxcv2
461HndB63Lc9e++B5QvtvxeDijsRe9Ti3nLm2a3H3ewxjwI
G97uTpC5bPVodTNMauaJMkiSp33wpyvz2pSRJUgdYlEm
SpJE2zBuSJ7lTkpOTXNH8vOMUbbZK8o0klya5JMkrZxL
bokySJI201VWz2tbT64FTqmob4JTm8WS/B15TVQ8CdgH
+Iclf3Bd8MosySZI00qpmt62npwOHN/uHA8v+Mp+6rqrOb

/ZXAZcC91pbYIsySZKkmbt7VV0HveILuNuaGidZCjwCW
OtX7wf+7csmmeOravtBX0uSJC08s50nlmQ5sLzv0FhVjfW
d/xpwjyme+sZZXmdz4AvAP1bVTWtr75IYkiRppM12nlhT
gI2t4fye051Lcn2Se1bVdUnuCfx0mna3o1eQfaaqjplJXsMavt
woyeFJViQ5OsmmSVYmuQtAkp2SnDqkXCRJktbVccCBz
f6BwP9NbpAkwCeBS6vqfTMNPKyibFt6XYMPBW4CXjb
TJyZZnmQ8yfjY2LRFrSRJWqCGuSQG8G5gryRXAHs1j0
myJMlXmjaPAZ4H7J7kgmbbd22BhzV8eXVVndnsHwEcP
NMnTupirLZvsyRJkkbbMBf0r6qfA3tMcfxaYN9m/wzW4V
6gwyrKJr9dRW8Nj4meuo2HlIckSZpnvM3S7GydZNdm/w
DgDGAlsGNz7JlDykOSJM0zQx6+HJhhFWWXAgcmWQH
cCfgo8DbgA0m+CfxhSHlIkiR10sCHL6tqJTDVrQW+CTxg
0NeXJEnz23wZvnSdMkmSNNLmS1HmbZYkSZI6wJ4ySZ
I00ro8eX82LMokSdJImy9FmcOXkiRJHZARqy5HKllJkha
QWa9g35avXfz9WdUHe25//znLdU1Gbviy7dssbbbLTgDce
OzxrcYFWLxsP974v19Ze8NZeudz9h1YvtD+ezGouBOxRy3
uzaee0XrczXd7LDB6vztJasOIdTBNy+FLSZKkDhi5njJJkq
R+86WnzKJMkiSNtNXzZMq5RZkkSRpp86WnzDllkiRJH
WBPmSRJGmmr50dHmUWZJEkabavnSVXm8KUkSVIH
DK2nLMlmwOeBLYENgXcAVwIfADYDfgPsUVWrhpWT
JEkaffNlov8why/3Bq6tqqcAJFkMfAfYv6rOTbIFcNsQ85E
kSfPAfCnKhjl8eRGwZ5L3JHkcsDVwXVWdC1BVN1XV7y
c/KcnyJONJxsfGxoaYriRJ0vAMraesqi5PsiOwL/Au4CRmc
IPxqhoDJqqxavvel5IkabT9oVbPdQqtGFpPWZIlwK1VdQ
TwXmAXYEmSnZvzi5L4bVBJkrQgDbMIegjw70lWA78D
XgoE+GCSTVk3A9QAACAASURBVOjNJ9sTuHmIOUmS
pBE3T6aUDXX48kTgxClO7TKsHCRJ0vyzep5UZa5TJkmS
1AHO4ZIkSSNtviyJYVEmSZJG2nwpyhy+lCRJ6gB7yiRJ0
kibLxP9M2JdfiOVrCRJC0jm6sKfOfP8WdUHz33MDnOW
65rYUyZJkkbaiHUwTWvkirKVP/9Vq/GW3vmvADjz8h+2

GhfgMQ+4Nzcee3zrcRcv249X/8//tR73fc9/OgCnXnplq3F3e9
B9ATjq7BWtxgV49qMeyjuOObn1uG/+67049MuntR73VU9
5At++4ketx911m60B+OqFl7Uad5+HPRCA9xz39VbjArzua
bvz9i+c1HrctzzzSa3HlKRhGLmiTJIkqd/q+dFR5rcvJUnSa
KuqWW3rI8mdkpyc5Irm5x3X0HbDJN9JMqNhM4sySZK
kmXs9cEpVbQOc0jyeziuBS2ca2KJMkiSNtGH2lAFPBw5v
9g8Hlk3VKMmWwFOAT8w0sEWZJEkaaaurZrUlWZ5kv
G9bPovL3b2qrgNoft5tmnbvB/4JWD3TwJ2Y6J9kN+C1Vb
XfXOciSZLmt6oaA8amO5/ka8A9pjj1xpnET7If8NOqOq+p
cWakE0WZJEnSump7mbKq2nO6c0muT3LPqrouyT2Bn07
R7DHA05LsC2wMbJHkiKr62zVdt9XhyySbJflykguTXJxk/
yR7NN88uCjJp5LcoWm7d5LLkpwB/HWbeUiSpIVjyHPKj
gMObPYPBP5i4dCqekNVbVlVS4HnAF9fW0EG7c8p2xu4t
qoeVlXbAycA/w3sX1UPodcz99IkGwMfB54KPI6puwglSZK
65t3AXkmuAPZqHpNkSZKvrE/gtouyi4A9k7wnyeOApcBV
VXV5c/5w4PHAA5vjV1SvZD1iuoD9k/HGxqYd/pUkSQvU
H1avntW2Pqrq51W1R1Vt0/z8RXP82qrad4r2p850znyrc8qq
6vIkOwL7Au8C1nQPlRn1H06ajFdt32ZJkiSpC1otypIsAX5
RVUckuRk4CFia5P5V9X3gecBpwGXAfZLcr6p+ABzQZh6
SJGnhWO0Nyaf0EODfk6wGfge8FFgMHJVkI+Bc4GNV9Z
tmTZAvJ7kBOAPYvuVcJEnSAmBRNoWqOhE4cYpTj5ii7
Qn05pZJkiQteK5TJkmSRloLy1x0gkWZJEkaafOkJvPel5Ik
SV1gT5kkSRppTvSXJEnqgPkypywj9kJGKllJkhaQzNWF
D/3yabOqD171lCfMWa5r4pwySZKkDhi54ctVq1a1Gm/Rok
UA/PqSy1qNC7Dxdg/k2l/d3HrcJX+1ObeccVbrcTd77C4A3
PTVk1uNu8U+ewHt/+6g9/u79dzzW4+76c47DCzfQcUF+M0
VP2g17h22uR8Aq045rdW4AIv2eAK3XXRJ63E3ech2/Ozm
21qPe9fNN2k9pqR2OKdMkiSpA+ZHSebwpSRJUifYUyZJ
kkaaw5eSJEkdMGIrSUzL4UtJkqQO6ERRluQfk2w613lIkq
TRs3p1zWrrqk4UZcA/AhZlkiRp1qpqVltXDaQoS7I0yWVJ

Dk+yIsnRSTZNskeS7yS5KMmnktwhycHAEuAbSb4xiHwk
SZK6bpA9ZdsCY1X1UOAm4NXAfwP7V9VD6H3J4KVV9
Z/AtcATq+qJA8xHkiTNQ6urZrV11SCLsqur6sxm/whgD+
Cqqrq8OXY48Pi1BUmyPMl4kvGxsbEBpSpJkkZVzXLrqk
EuidHK666qMWCiGqtB3KJGkiSNri7PE5uNQfaUbZ1k12
b/AOBrwNIk92+OPQ+YuKHeKmDRAHORJEnzlMOXa3c
pcGCSFcCdgEOBFwJHJbkIWA18rGk7BnzVif6SJGmhGu
Tw5eqqOmjSsVOAR0xuWFUfBD44wFwkSdI8NV+GL73N
kiRJGmldHpKcjYEUZVW1Eth+ELElSZLmI3vKJEnSSJsn
HWUWZZIkabQ5p0ySJKkD5sucsq7ckFySJGlBy4h1+Y1Us
pIkLSCZqwu/7rPHz6o+eM/f7Ddnua7JyA1fnnrpla3G2+1B9
wXghBXfazUuwN4P3ZYbjz2+9biLl+3Hvu9q/z6gX3nDcgA
OP3281bgHPn4nAD54whmtxgV4xd6P5dmH/nfrcY961Qt4+
xdOaj3uW575JD7x9bNbj/vi3R8FwEdP/larcV+616MBOOA
Dn241LsCRr3we+7//8Nbjfu4fDxzYf3cA41f9uNW4O93nXq3
GkxaiEetgmpbDl5IkSTOU5E5JTk5yRfPzjtO0+6skRye5LM
mlfbeenJZFmSRJGmlVNattPb0eOKWqtqF3p6LXT9PuA8
AJVfVA4GH0bj+5RhZlkiRppK2u2W3r6enAxNyLw4Flkxs
k2QJ4PPBJgKr6bVX9am2BLcokSdKCkmR5kvG+bfksnn7
3qroOoPl5tyna3Bf4GfBfSb6T5BNJNltb4JGb6C9JktRvtkO
SVTUGTPuNuSRfA+4xxak3zvASGwE7AK+oqrOTfIDeM
Oeb1/YkSZKkkdX2ty+ras/pziW5Psk9q+q6JPcEfjpFs2uAa6
pq4uv2RzP93LM/mtHwZZKlSS6eSVtJkqR57DjgwGb/QO
D/Jjeoqp8AVyfZtjm0B/DdtQW2p0ySJI20Id9m6d3A55O8C
PgR8GyAJEuAT1TVvk27VwCfSXJ74ErghWsLPJuibMMk
HwceDfyY3rcPlgAfBu4K3Aq8pKouS/LfwG3AA4F7N4kcC
OwKnF1VL2hewAHAP9NbBfjLVfW6WeQjSZI01MVjq+r
n9Hq+Jh+/Fti37/EFwE6ziT2bb19uA3y4qrYDfgU8k94kuVd
U1Y7Aa4GP9LW/I7A78CrgS8ChwHbAQ5I8vKko39O0eTi
wc5Kpvlb6x29IjI21v4q9JEkabUNeEmNgZtNTdlVT9QGcB

yyl12t2VPLHW0jdoa/9l6qqklwEXF9VFwEkuaR57r2BU6vq
Z83xz9Bb0+PY/otO+oZEtX2bJUmSNNpW1+q5TqEVsynK
ftO3/wfg7sCvqurha2m/etJzVzfX/f0sri1JkjSvrc/isTcBVyWZ
mOCWJA+bxfPPBp6Q5C5JNgQOAE5bj3wkSdICVDW7ra
vWd0X/5wIvSnIhcAm9yf8z0qyC+wbgG8CFwPlV9RdfK5U
kSVqTId/7cmBmNHxZVSuB7fsev7fv9N5TtH/BGp7bf+6zw
Gdnnq4kSdKfG/KSGAPjvS8lSZI6wMVjJUnSSOvykORsW
JRJkqSRNl+KMocvJUmSOsCeMkmSNNK6vEr/bGTEuvx
GKllJkhaQrL3JYLzwI0fOqj74r5cdMGe5rsnI9ZTdeOzxrc
ZbvGy/gcSdiL187POtxx1b/v8Gli+M3ntsXH93k+O+/QsntR7
3Lc98EgC3nnNeq3E3feSOADzjvZ9qNS7AF1/7d63HlDQ4I1
eUSZIk9Vs9TwbSLMokSdJIG7GpWNPy25eSJEkdYE+ZJ
EkaaavnydcvLcokSdJIc/hSkiRJrWm9KEuyNMnFUxx/QZI
lbV9PkiQtbKtrdltXDbOn7AWARZkkSWpVVc1q66pBzSn
bMMnHgUcDPwY+DewEfCbJbcCuwKXA4cBTgdsBz66qy
waUjyRJmqdqnqxTNqiesm2AD1fVdsCv6N0eaRx4blU9vKp
ua9rdUFU7AB8FXjugXCRJ0jy2umpWW1cNqii7qqouaPbP
A5ZO0+6YtbVJsjzJeJLxsbGxVpOUJEnqikENX/6mb/8Pw
CZrafeH6XKpqjFgohqrQdwrT5Ikja4uzxObjWGuU7YKW
DTE60mSpAWgy9+onI1hFmX/DXysb6K/JEnSerOnbBpVt
RLYvu/xe/tOf6Fvf2lfm3Fgt7ZzkSRJGhXeZkmSJI00e8okS
ZI6oMvLXMyG976UJEnqAHvKJEnSSJsvPWUWZZIkaaT
NlzllGbEXMlLJSpK0gGSuLvyUd4/Nqj748uuXz1muazJyP
WVtr+i/eNl+ANz8zW+1Ghdg88c9mpd+4ujW4370xc9q/X2
AP70Xg3qPB5Wzcf3dTY576JdPaz3uq57yBABuPee8VuNu
+sgdAXjJYZ9rNS7Ax/9+/4F+3qSuGGb/UpI7AZ+jt7TXSu
D/VdUvp2j3KuDF9DqULgJeWFW/XlNsJ/pLkqSRNuQbkr
8eOKWqtgFOaR7/mST3Ag4Gdqqq7YENgeesLbBFmSRJ0s
w9HTi82T8cWDZNu42ATZJsBGwKXLu2wBZlkiRppFXV
rLb1dPequq657nXA3abI58fAe4EfAdcBN1bVSWsLPHJzyi

RJkvrNdkgyyXJged+hsaoa6zv/NeAeUzz1jTOMf0d6PWr3A
X4FHJXkb6vqiDU9z6JMkiQtKE0BNraG83tOdy7J9UnuW
VXXJbkn8NMpmu0JXFVVP2uecwzwaGCNRZnDl5IkaaQ
NefjyOODAZv9A4P+maPMjYJckmyYJsAdw6doCD7woS3
JIktcO+jqSJGlhqprdtp7eDeyV5Apgr+YxSZYk+UovnzobO
Bo4n95yGBuwhp65CQ5fSpKkkTbM2yxV1c/p9XxNPn4tsG/
f47cCb51N7IH0lCV5Y5LvNRPltm2OnZrk0CSnJ7k0yc5Jjk
lyRZJ/GUQekiRp/hvy8OXAtN5TlmRHegukPaKJfz4wsQT2
b6vq8UleSW8MdkfgF8APkhzaVJ+SJEkLziCGLx8HfLGqb
gVIclzfuYn9n9gRLJtb5SHIlsBXwF0VZ/9dWDzvsMPa/25IBp
CxJkkbVqYe8vJP3spytQc0pm65v8DfNz9V9+xOPp8xl0tdW
axD3cZMkSZprg5hTdjrwAD4IfDNtq8hSZI03wxk+LSZ6H8c
xU4te/xboPIQ5IkaVS4or8kSVIHWJRJkEZJElSB1i
USZIkdYBFmSRJUgeky/eAmsJIJStJ0gIyL1bVn0v2lEmSJ
HXAoG6zNDA3ffXkVuNtsc9eAKw66eutxgVY5VY9KTd+Z9vnr
f2hrP0/MftyCBuN7V42X4Arcf+Y9zjvtJqXIDFT9t3YO/Fq
MWFAf7uRuy9+PYVP2o97q7bbA3AqlWrWo27aNEigJH7f
8UgP8fSQmVPmSRJUgdYlEmSJHWARZkkSVIHWJRJki
R1gEWZJElSB3SiKEuyMsld5joPSZKkudKJokySJGmxm3VRluTYJGcmOS/JAc2x65Mc3hz7apLd
kiUWxJDnOYUVJ8SkVXmoXe96k4GZWHSL9Rg1Sup6Cf/tlCsrSdcHqJ1/GCxdQDvNTLvzl7x5IqjNO
HIKHGYJ9C4rNvCdu6FAyjR/SMKy2A0l96NLdu0/fNjbUuyk/LUFs3h+EJn43sBq5gPMxHYNhsOUy
WdsiTPB15Lb+X9FcDngTcBtwd+Djy3qq5Mch4A2rQn
4CrAkSVrgWuspS7Id8ZZg96p6BPcDWcPKqFEn4oPT/KQ3FAB
81zd8KnNEcPw7Yuq08JEmSRlGbw5ePBT5RVdcBJJmf3kyx6+uUQvYtHChARM9OS
EvgxCQXAf8fsF3T9vHAEU27y+lA/2/nIbaseRNMb42q9+sHEPiHZzpg5RpQ85yzgzGObHFqUpVtYAWkY
LVxYKD+uPWm7B8dS5Oh8h+rVDqvOA7RlyX6bKozDkYd3Al4yYxr+33Xf8zSuyvhMBh5NpKL8f8fbgzBoh7GXL9Ck
x0JSPrTlhB9+MK3/eGOdjJLJmmQQM2s5kCKZQXV9+cWmGBGr+eUKbxOVvHJq2jsJ3tWvMIu1VoECJ0ydx2eZH
bkinRijrkpUeJ+MQzyXoVkS0WZEW7QVYG2jd7XQMqw/yMdmr1MiRjG4eaZIaQ1kXs9dq3Z27T28Gr
PkGvSaYEr8UuoedcLq5ASvyHoTIe4MeCrpvNa8vjX1w6cArDaKyTSUrJK6n/lNjpdtISwqSFzJBwDsN
SyJmSVH3JJdIIl0SBZzgd6z2Mx4E0LW8EY4kEFr0AAe1VfERUJ8Y7d7JTsFBwOFNuz6FaMjZpUp
4N91wIIkVLw1l1gyIm4w3bFN4B0drKEzUxpFFEmSWLWfK4QX/h2uv8AkkSWXVgFBZWkMSmSlNuD
mS/FrSeBDZLpcuUoLT/ZHiIP8nSCOEjkjpaM8L3f0nkLWq/oaB8QgF/d8W+DEb7j7gnWGeT9uVB
HsUh/Ku5XTwvoDhF8+LHbpNiBGBh4wTQ1xWzkYiuu2Xla35T/Wgmm5ktg5qwzd/+hJJKS9oIEl0Tgjrx
pJktwyKJA6GP5Ghxh+oqh/ExWYCVPxG1MYGhtVHjRgRhXSVXQTi/vH3NESSlDHd8pSoHnwz0+Qu
ZZqwiRSFM0WkrqXoBCyNQo3zOzP6fHGCGHO+7MngGGaCuqYZGGlIqOCOLqVLv2ltxIqWl+QhoL

IB4H+v1ZE+ZJElSB7S6TtkwXPCj61qN9/Ct7wnAd6/9Wat
xAR685K7ceOzxrcddvGw//ukzX2o97r8996kAfPuKdtfy3X
Wb3v3mx045q9W4AMv32IW3f+Gk1uO+5ZlP4qMnf6v1uC
/d69GMX/XjtTecpZ3ucy8Ajjp7Ratxn/2ohwLwrmNPaTUuw
BuW7cG/fvFrrcf952fsObD/7gBWrVrVatxFixYBDCznNxz5
5dbjvuuAp/BvX/pG63H/6alPBODy63/eatwH3P3OrcaTBsW
eMkmSpA6wKJMkSeoAizJJkqQOsCiTJEnqAIsySZKkDr
AokyRJ6oA5L8qSPDzJvnOdhyRJ0lya86IMeAOw71wnIUmSNJe6UJRJkiQt
eMkmSpA6wKJMkSeoAizJJkqQOsCiTJEnqAIsySZKkDr
GRVlSY5Ncl6SS5Isb47dnOQ/kpyf5rk/Um+leTi
JI9sjm+W5O3A74O3A/kkuSLL/oF6oJElSl820
p+zvqmpHYCfg4CR3BjYDzq+qqHYDTgLf2td+sqh4NvAz4
VHPsjcDXq2pn4nAvwO3A94CfK6qHl5Vn5t84STLk4wnGV+hb1uHlyhJktR9M73N0sFJXtQs/xVw//oz0BrvsLjk2
R8bG1uHlyhJktR9M73N0sFJntHsbwVsA6zGF
CVBVpyfZIslfAU8rBFsprcoBJqq5xAuBSrFtiskkgPhhOzDPDJo
SV2w1qIsyW7z69hd5K2vrbkPPpFLOqvjvmnP1zL5Dxcp
qNmk7AkSdJ8PhyAGMbL1bMyoUqX0mSvHm0lPZnWCYKrWLXUHbq
R8gyWOBG6vqRuBE4Jb0Jy8i3qtLj7cA2k4DP9FGWvwHANuRHb47N1YH+7OTf5U4/uUh9CCKL
UXYCsFGSekZavI6r502UslKkrSAH0c47vkIs4gOIH2Jm4ZsH0f6vXz9Fq
eBKq6s3Am6d42heq6g2T4Ikdag2T4twG/P0U8X8B7DzjJCkUL
xTJkmStODN9NuXf6GqNp/m+ICIJoOUIZB0s05YEldrdZJ7PggkQTVXrGzY+XLOkYSp
USZIkdYBFmRZSRJUgekavI6r502UslKkrSAZK4TGHXrPNF
/rtx86hlrbzQLm+/2WAB+fdnlrcYF2PiBD+Blnzy69bgfedGz
uPHY41uPu3jZjZfgCtx/5j3C+d0GpcgMVP3Xtg78WoxYUB/u
5G7L049MuntR73VU95AgCrVq1qNe6iRb21s1/z6eNajQvw
H8972sj97gBu+fY5rcbdbdNdHAnDwfx2zlpaz958v/OvWY2r
hcvhSkiSpAyzKJEmSOsCiTJIkqQMsyiRJkjrAokySJKkDLL
MokSZI6wKJMkiSpAyzKJEmSOsCiTJIkqQMsyiRJkjrAokySJKkDLMokSZI6wKJMkiSpAyzKJEmSOsCiTJIkqQMs
Jjk6yaZIdk5yW5LwkLRJya6qph9BzkSRJkDr
m4B/AD4IPKuqdgr6sxm/wjKuqdgr6sxm/wjgycD

2wMlJLgDeBGw51ROTLE8ynmR8bGxsONlKkiQN2bDuf
Tn5RuKrgEuqate1PrFqDJioxqrte19KkiR1wbB6yrZOMlG
AHQCcBdx14liS2yXZbki5SJIkdc6wirJLgQOTrADuRDOf
DHhPkguBC4BHDykXSZKkzhnW8OXqqjpo0rELgMcP6fq
SJEmd5jplkiRJHTDwnrKqWknvm5aSJEmahj1lkiRJHWB
RJkmS1AEWZZIkSR2QqsnrunbaSCUrSdICkrlOYNQNa0
mM1vzmih+0Gu8O29wPgBuPPb7VuACLl+3HJ75+dutxX7
z7owaWL8Ct557fatxNd94BgFvOOKvVuACbPXaXgb0Xg4
p720WXtB53k4f01l7+9cWXthp34+0fBAzuv49BxX3j/36l9bj
vfM6+AKw6+Rutxl201xMBeOknjm41LsBHX/yswf3uvnRC
+3GfujcAN3/99Fbjbr57b/WldxxzcqtxAd7813sN9P/HWlgcv
pQkSeoAizJJkqQOsCiTJEnqAIsySZKkDrAokyRJ6gCLMk
mSpA6wKJMkSeoAizJJkqQOaLUoS3JskvOSXJJkeXPs5iT
vTHJhkrOS3L05ftckX0hybrM9ps1cJEmSRknbPWV/V1U7
AjsBBye5M7AZcFZVPQw4HXhJ0/YDwKFVtTPwTOATLe
ciSZI0Mtouyg5OciFwFrAVsA3wW2DiHhTnAUub/T2BDy
W5ADgO2CLJoskBkyxPMp5kfGxsrOV0JUmSuqG1e18m2
Y1eobVrVd2a5FRgY+B39ae7nv+h75obNG1vW1PcqhoDJq
qxavvel5IkSV3QZk/ZYuCXTUH2QGCXtbQ/CXj5xIMkD2
8xF0mSpJHSZlF2ArBRkhXAO+gNYa7JwcBOSVYk+S5w
UIu5SJIkjZTWhi+r6jfAPlOc2ryvzdHA0c3+DcD+bV1fkiRp
lLlOmSRJUgdYlEmSJHWEmSJHWARZkkSVIHWJRJkiR1gEWZJ
ElSB+RP67qOhJFKVpKkBSRzncCos6dMkiSpA1pbp2xYbj
z2+LU3moXFy/YbSNyJ2Hv/62Gtxz3hn/9+YPnC6L3HxvV3
NznuSz9xdOtxP/riZwFw24UXtxp3k4dtD8Buh3yo1bgApx7y
8pH73cHgPsevOvzYVuMCHHrgspF8j9VN9pRJkiR1gEWZ
JElSB1iUSZIkdYBFmSRJUgdYlEmSJHWEmSJHWARZkkSVIHWJ
pEkSVqo7CmTJEnqAIsySZKkDrAokyRJ6gCLMkmSpA6wKJMk
4UkmzbHn53k4ub46c2x7ZKck+SCJt427bwkSZKk0TOroiz
JdsAbgd2r6mHAKyc1Oaaqdm7OXQq8qDn+FuDJzfGnNcc

OAj5QVQ8HdgKuWcfXIEmSNPJm21O2O3B0Vd0AUFW/
mHR++yTfTHIR8Fxgu+b4mcB/J3kJsGFz7NvAPyd5HXDv
qrptqgsmWZ5kPMn42NjYLNOVJOn/b+/co+yq6jv++YYgS
YAIRJYUJOShwEKIwCISsxSBahF5FYEA5WF5U+TdUms
L8pbaChZYNkAFTRFsgVVLopVAaQjyFghPwbp4CBRpp
Q0Y0PD89o+9T+bOMISZu8/J3Hvz+6x118w5c+c7vzzl3n7N/
e/9++7eDoDsYrlMmlr8p+HeBY21vAZwJjAGwfTRwKrAh8
ICkCbavJs2a/Q6YL2nHwQRtX2Z7G9vbHHnkkcM0NwiCI
AiCoDsYrlN2MzBL0gQASesM+PmawK8krUgUqaKSO/b6rtu
21/FXgR2FDSFOBJ2xcBc4Fp7f4TQRAEQRAE3c6wNiS3/a
ikc4GFt4zlNOBu4JfAwyQnDeBvcyK/SI7WcqD/SI7dg8BfA
AdKegN4ATir4P8IgiAIgiDoaoblllAHYngPMeZefzQZmD3L
+C4O8/bz8CoIgCIIgWOmJOmVBEARBEAQdQDhlQRAE
QRAEHUA4ZUEQBEEQBB1AOGVBEARBEAQdQDhlQ
RAEQRAEHYDs5dWC7Ti6ytggCIIgWInQSBvQ7XTbTJm
G+pJ01HDe36u63Whzt+l2o83dptuNNnebbjfa3G263Whzt+l2o83dptuNNnebbjfaHLodZ3NQS
Lc5ZcOhqT2Zuk23Se3QbV47dJvXDt3mtUO3Wd0mtWN/
wxVILztlQRAEQRAEXUM4ZUEQBEEQBB1AALztll4Vu49
qh27x26DavHbrRBEARB0JP0
8kxZEARBEAQdQDhlQRAEQRAEXUM4ZUEQBEEQBBE
EH0JNOmaTVR9qGkUDSOst7jbR9y0PSZEljWo7HSppUg
+7qkka1HI+SNK5Ut9HcMhPrvuRtLakqZJ2rp
6FWg1/myTtM9QznUadkr23kd3kqj4GHGX7mEdLd
nkr0lzQT+Dawhu2GJGX7mELdKcCFwkCeAt4E7gZNs
P1moO493bh31MnAvcKntLWkK20Bt2fc0ZZWK20Btj2lwFwk
LWCQra5s71iim7XvBWbafj0JIvPoZ20bZ2OtOtP
caHtmoe6Xg9cHHtmoe6XgKtsv5SP1wb2t/33Jbot
A/sVl3jumiqXdT92b3L/bYM27u3ZWj/v3Gz7d9/r3Ntau8Ob
JcPF9qeV4PmxsApwEbA6Op8DZ/d2cAfA0/Qd83tu6t9t
EYHH+fi3gGduTS+zNf+NDP8Y3/bOkjYDP

mH78kLdpvq7B4BtgEnAfGAusIntz5foBkNj9Hu/pav4JrAT
qRFh+0FJ2y3/V4bE1cC3gD3z8X7A94FtC3WfBNbNWgD7
Av8NbAz8A3DQcMRaH0x59PgRWjr1Gvizlu/HAHsBb9ak
PbrVWbD9enbMShlTdepZ95WaZluOsP2tFt3Fko4Aip0ySZ
cA44AdSA/dvYF7SnWBp4HbJc0FXq1O2r6gULepdlH3Z/e
N/PULwHpANXO4P+natE2e5R0HfCA76NXAaDywfol21j8
P+DhwVT51vKSZtr9SKH0tcAnpefNWoVYrs4CpdQ0Aqm
dbvjfm2v63fLwz8JkS7azxeWADSRe1/Gg89bTj7wLfAf4qH/
8n8M9AkVNGc/3d27bflLQn8He2L5a0qAbdYAj0mlOG7W
elfltw1fGgke0rW46/J+nYGnS3st16E82TdKvt7SQ92q6opM
OBE4APAQ8AM4A7gKLRuu37Bpy6XdLCEs0Wfi1pd9tzA
STtAbxYg+6rkra2fX/W3Qb4XQ26oyTJeapZ0ipAHU4kpBn
DaZIesn2mpPOBf6lB9/n8GgWsWYMe0Gi7qPWzs70w65w
92H1XZipHASeSHLD7W87/hjSgK2UXYEvbbwNImgMsAk
qdsjdtzy41bhAeIc1i/U/NutNtH10d2P5xnpUr4XlSdGJ3oLUt
LwFOKtQG+IDtayR9BSA7PLU4wA31d29I2h/4IrBbPrdqD
brBEOg1p+zZPKXrPMtyPPBYDboL8g31fdIU+r7Aj6pcBtv
/16buupIm2n4GQNJGpJkzgJIR5gnAdOAu2ztI2hQ4s0CPb
F9r7sYo0hT3eqW6maOBqyRVHdizwME16J4IXCvpedJntz
7p8ytlPnBNHrmbZP8NNehCn+PxW0nrA/8LFIdnbBe3gcF
osF2cQDOf3bqSplTpB5Im03fftYXtC4ELJR1n++IabByMt
YDqWfP+mjTnSToG+AHwWnWy4JlWcR6wSNIjA3RLQ8
QvSjqVNMtp4EDS/dE2th8EHpR0NalPnGj754V2tvKqpAn
kMK6kGaQ0lVKa6u8OIT3PzrX9VL4/uioftZvpNafsaFLu1
wbAc8CNwJdq0K06giPy12pocijpRms3V+tk4DZJT+TjKcA
xOXFzTpuaAEttL5WEpNVsPy5pkwK9ivvoy+t4gxTyOawG
XWw/AczIeUOyvaQOXeBhUnhmJ9KsxTyg7VnIFr5M2qj3
T0jX40ZSqLEOfpgTa/+GvpF7sXbdOYwtVO0CUrjnaeppF
5OBrUg5RHuSZnzrSII9CbhFUpUTOon6Nl2+dl06EgAAC
XxJREFUVNLx9OV+3UK6tm8U6lZOzgJSe9uO8lkySLMh

kPLKKkqeaRVzgK+T7r+3C7Va2R84neREAtyaz9XB50gh7
vcBkyVtCZxVgyN5MinEOEXS7aQBwN6FmtBQf2f7ZyQH
r8qVXdP2X5fqBkOjpxL9m0LSWOAY4JOkB9ZPgNkFnVi
luw9pxmUysAcwE/jLKlxToPsD0mjnRGBHUlLsqqWJmpJ
mATfY/o2k04CtgbNL7c3aE0gP2+oa30Z6IBaNgiVdQ3LGql
yc/YG1bRetqsqO81Lbb+XjVYDVbP+2RDdrjSU5e5+i3vZ2I
e/MYXwBGAuMtz2sHMYB9jZxfzyUw7ifJCVKn0+6P0pzO
VFaXbZpPnzc9mvLe/8wdL9NCvVUg6qDgLdsH16D9u+RZ
sAF3G37hRo0xwz8nAY714buQtufLrNuxSLpPtLz8hbbW+
VzD9meVqg7BjiWNDBcQloodnHpNW4KSbeQQrmjSekvv
yYtLDl5JO1aWegpp2xAkmbFy8C9tq8v0B2sY1/L9qx2NbN
uY51Oy9/4NCnUcUNp0m3DneRNpFFvNU1+ALC97dIk3gd
tf+y9zrWh28iqzqx1Denh3ZqIXkd7u3VALtWyc5Ietf3RAnu
bcHwX2d4qJ7k/bPvq6lyh7qokp7fu2azG2lvWmUaa1WtdJV
mUa6jmVhxeQApbzqV/+LJ0wLku8Oe8c2VyHSvA77a9bW
sbq8kpa+r+WJcUvZlE/zZxaKFudd8dDmxo+/Q6rkMwNHo
tfDmGNPq9Nh/vRQpVHSZpB9sntqm7yYCH6gJJDxbYW
VElZe4CXGL7ekln1KC7jCq5uSaatHcd260Ju+dI+sMadBd
JmmH7LgBJ2wK316Db1KpOaK69NZXD2JS9/yXpUtLquq
/n2a06aivOJs1mVStlD8rnimezgLckTc3h+KqcTnHytaQrgG
mk51kVDjRtLgCRtB4p7DVW/euHjSetIi2lcpyrAZtI9pY6T
1eRVi7uSgrffZE0k1MHj0j6I2AVSR8hhfDuqEG3qfvjetKs9
L9T78rZ0XlWdhZ9K0aDFUSvOWUfBna0/SaApNmkOPtn
SbkN7dJUx95Up9MUTdq7QNJ+wDX5eG/gRzXobgscLOm
ZfDwReEzSw6S6Se2O/ppa1QnNtbc/pZkcxqbsnUXO87H9U
u4oTnmP3xkK0wd0kv9RUycJyb4FA/LVDqlBd4btzWrQqd
iJVEfsQ/SVCoE0Q9t2rpqkKsT1Q/rXTIR68gEn2L5c0gl5w
LlQ9a0AP47khLxGCvHPB0pXdkJz98c421+uQWcgZ5H+9
9ts/zQPLH7RwN8JBqHXwpc/Bz5u++V8/H5S7sWm7YQ9q
o6bNKreBHgmH28E/Mz25oX2jiN1Og/b/kXudLawfWOJbl

M0Ya+kJaQH99j8tRrxrQK8Ynt8oc0bLe/ntn/Zpu504J9Iy+
mXrQz0O8tDDEez6fY2huSYbUMKad8EfLPd3Jam7W0KS
fcD+wyYzbquNGSXtaprXJWfKbrGLbqXA+fnJOzakHQg6
TObRN8g3W6zULGk0/O3m5Dy364n3de7AbeW5tZJusv2D
KVCyBeR7r/rbE8t0R3wN8aTrkEti40kPUbf/QF5YEia8Wx7
YCjpHOAO55ptQW/Qa07ZYcCppByRaoXS10ijnjNsD2uU
3VSHHvRHkoD76ugUVxS58z2OvlWdxcm7Tbe3unNbuvX+
kLQjqaBnv9ks2wtq0G4qf2g70srhF0gzOdUuHaX5TvNJC4
HupyUEZvv8Qt0bgb0qx0bSmsC1tj9XqLsrKWS3IXAxKdx
6huvZ3WA6cAV9NfxeBg4tGWhl3aYGhkuA1Unt4Q362kT
pQHYMafX0wLy9oly1YGj0VPgyT2v/mJQj8jgpdPmc7Vdp
I+zRqZ1Kr2Hbku6UNN32T0faniHyj6TO99x8vD9wJdB257
sC2lutuS1dfH9MADYnOWPVquc66kZBc/lDV5Cea3WXm
NjA9k416lVMpH+e4uuk613KPqSw2iPADko18r5BclhLuRw
4xvZPAJQWNH2HlMvXNk3dJ7bXVDM7t1xJ6j93IoUyD6
Ce+mfBEOgpp0yDV7K/k/Lk0qB5dgCOlvQ0aQugWmYCG
qSpzrdJmspt6TZOs31tDlN9lrSKeDbl26ZBc9f4GefdLmrm
Dklb2C7JuR2MK4F7lMrzmFRnriRvsWKa836zkIrcSipajdv
Cksohy9q35dmojuRd+rvinVuAD9veR9IetucoFdWdX6gZD
JGecspoqJJ9sELYeaQNGCZd4+AMyP2qFj0sy/0aSdtGiCZ
XETe1sOTx3DnOo3+JiXZXX1ZtYjRwSF6YUFtY1Pa5OW
rxqXzqENt17J84StLathcD1W4SdfVj9+SFTK07t9xSrU51Df
UYa6ap/q4qDfOSpM1JIfNJNegGQ6DXnLKmKtkHDdOFo
bCmOt8m2HWkDegwmlxFXJQztRzGkpymP2g513ZJDFZ
Am8hOTN2OzPmk2b3rSP//LPpSCErZMn89fcD5mdRTzq
NumurvLlOq5H8aqc7cGsBXa9ANhkCvJfo3Usk+CAbSrUn
uQfeteg76I2kz0vNdwM11r0jtFqK/6016yilrRTVWsg+CIBhJ
JG1Mynv7oO3Nlar77277nBE2rWdQQ1u9rQjq6O9aaswNi
u0L2tENhkfPOmVBEAS9Qi6QegppO6hqC6BHOrUWXDe
ihrZ66xZaaswNLPoLBbXrguHRazllQRAEvcg42/ekkn7LeH

OkjOlRmtrqrSuwfSaApDnACdUq15xfVlS3Lhg6nbylTxAE
QZB4UdJU8lZFkvYGfjWyJvUcCyTtJ2lUfs2inq3euo2BZUc
W07eXadAwEb4MgiDocPJWUJeRVgIuBp4CDogFJfXRUi
G/dau3V/P3xZXyu4Vcb3H7AWVHFtreYmQtWzkIpywIgq
DDyWU79ibVi1qHtJtE5PnUzGAV8p02Pl9pkHQwaVP6fm
VHbF85ooatJIRTFgRB0OFIugF4iZr3qAz6eLcK+bZLK+R
3HVF2ZOQIpywIgqDDiZWWzZMLP1cV8resKuTb3neET
QtWIiLRPwiCoPO5Q1Lk9DTLUttLgWUV8oHYESZYoU
RJjCAIgg6l6T0qg348J2kt4F+BmyQtBp4fYZuClYwIXwZB
EHQosZ3XyBA7wgQjRThlQRAEQRAEHUDklAVBEARB
EHQA4ZQFQRAEQRB0AOGUBUEQBEEQdADhlAVBE
ARBEHQA4ZQFQRAEQRB0AP8PM3ezyIA0C6UAAAAA
SUVORK5CYII=\n",

```
      "text/plain": [
       "<Figure size 792x648 with 2 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "corr_df = df2.corr()\n",
```

```
    "\n",
    "# Generate a mask for the upper triangle\n",
    "mask = np.zeros_like(corr_df, dtype=np.bool)\n",
    "mask[np.triu_indices_from(mask)] = True\n",
    "\n",
    "# Set up the matplotlib figure\n",
    "f, ax = plt.subplots(figsize=(11, 9))\n",
    "\n",
    "# Generate a custom diverging colormap\n",
    "cmap = sns.diverging_palette(220, 10, as_cmap=True)\n",
    "\n",
    "# Draw the heatmap with the mask and correct aspect ratio\n",
    "sns.heatmap(corr_df, mask=mask, cmap=cmap, vmax=.3, center=0,\n",
    "            square=True, linewidths=.5, cbar_kws={\"shrink\": .5})\n",
    "plt.title('Correlations between different predictors')\n",
    "plt.show()"
   ]
  },
  {
```

```
   "cell_type": "markdown",
   "metadata": {
    "_cell_guid": "288204a8-f8d8-4ceb-9a7d-d17d36c484b0",
    "_uuid": "de87bc336ca22e901bd0ee2c58d1160a8e7332fa"
   },
   "source": [
    "## Split the set for training models further into a (sub-)training set and testing set."
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 14,
   "metadata": {
    "_cell_guid": "5eb0de02-55b0-447d-9a14-7ec9915d7ffc",
    "_uuid": "71f45a3d6c98d57ac8f1592388f5700316de502a"
   },
   "outputs": [],
   "source": [
    "X_train, X_test, y_train, y_test = train_test_split(df2.iloc[:,:-1], df2['class'], \n",
    "                                                    test_size = 0.33, random_state=44,\n",
```

```
  "                                        stratify= df2['class'] )\n"
 ]
},
{
 "cell_type": "code",
 "execution_count": 34,
 "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>age</th>\n",
"      <th>bp</th>\n",
"      <th>sg</th>\n",
"      <th>al</th>\n",
"      <th>su</th>\n",
"      <th>rbc</th>\n",
"      <th>pc</th>\n",
"      <th>pcc</th>\n",
"      <th>ba</th>\n",
"      <th>bgr</th>\n",
"      <th>...</th>\n",
"      <th>hemo</th>\n",
"      <th>pcv</th>\n",
"      <th>wc</th>\n",
"      <th>rc</th>\n",
```

```
"      <th>htn</th>\n",
"      <th>dm</th>\n",
"      <th>cad</th>\n",
"      <th>appet</th>\n",
"      <th>pe</th>\n",
"      <th>ane</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>317</th>\n",
"      <td>58.0</td>\n",
"      <td>70.0</td>\n",
"      <td>1.020</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>102.0</td>\n",
"      <td>...</td>\n",
```

```
"      <td>15.0</td>\n",
"      <td>40</td>\n",
"      <td>8100</td>\n",
"      <td>4.9</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>296</th>\n",
"      <td>41.0</td>\n",
"      <td>70.0</td>\n",
"      <td>1.020</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
```

```
"      <td>125.0</td>\n",
"      <td>...</td>\n",
"      <td>16.8</td>\n",
"      <td>41</td>\n",
"      <td>6300</td>\n",
"      <td>5.9</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>157</th>\n",
"      <td>62.0</td>\n",
"      <td>70.0</td>\n",
"      <td>1.025</td>\n",
"      <td>3.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
```

```
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>122.0</td>\n",
"      <td>...</td>\n",
"      <td>12.6</td>\n",
"      <td>39</td>\n",
"      <td>7900</td>\n",
"      <td>3.9</td>\n",
"      <td>1.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>258</th>\n",
"      <td>42.0</td>\n",
"      <td>80.0</td>\n",
"      <td>1.020</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
```

```
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>98.0</td>\n",
"      <td>...</td>\n",
"      <td>13.9</td>\n",
"      <td>44</td>\n",
"      <td>8400</td>\n",
"      <td>5.5</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>291</th>\n",
"      <td>47.0</td>\n",
"      <td>80.0</td>\n",
"      <td>1.025</td>\n",
```

```
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>124.0</td>\n",
"      <td>...</td>\n",
"      <td>14.9</td>\n",
"      <td>41</td>\n",
"      <td>7000</td>\n",
"      <td>5.7</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"      <td>1.0</td>\n",
"      <td>0.0</td>\n",
"      <td>0.0</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
"<p>5 rows × 24 columns</p>\n",
```

```
      "</div>"
     ],
     "text/plain": [
      "     age    bp    sg   al   su  rbc   pc  pcc   ba    bgr  ...
hemo  pcv  \\\n",
     "317  58.0  70.0  1.020  0.0  0.0  0.0  0.0  0.0  0.0  102.0  ...
15.0   40   \n",
     "296  41.0  70.0  1.020  0.0  0.0  0.0  0.0  0.0  0.0  125.0  ...
16.8   41   \n",
     "157  62.0  70.0  1.025  3.0  0.0  0.0  1.0  0.0  0.0  122.0  ...
12.6   39   \n",
     "258  42.0  80.0  1.020  0.0  0.0  0.0  0.0  0.0  0.0   98.0  ...
13.9   44   \n",
     "291  47.0  80.0  1.025  0.0  0.0  0.0  0.0  0.0  0.0  124.0  ...
14.9   41   \n",
     "\n",
     "      wc   rc  htn   dm  cad appet   pe  ane  \n",
     "317  8100  4.9  0.0  0.0  0.0   1.0  0.0  0.0  \n",
     "296  6300  5.9  0.0  0.0  0.0   1.0  0.0  0.0  \n",
     "157  7900  3.9  1.0  1.0  0.0   1.0  0.0  0.0  \n",
     "258  8400  5.5  0.0  0.0  0.0   1.0  0.0  0.0  \n",
     "291  7000  5.7  0.0  0.0  0.0   1.0  0.0  0.0  \n",
     "\n",
```

```
      "[5 rows x 24 columns]"
     ]
    },
    "execution_count": 34,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "X_train.head()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 15,
  "metadata": {
   "_cell_guid": "a44df0d8-f87e-4f17-b7fc-5dc74aeb9af0",
   "_uuid": "fe32d3246677ac2ea2edb679f8bdeb036a12f24f"
  },
  "outputs": [
   {
    "name": "stdout",
```

```
   "output_type": "stream",
   "text": [
    "(105, 24)\n",
    "(53, 24)\n"
   ]
  }
 ],
 "source": [
  "print(X_train.shape)\n",
  "print(X_test.shape)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 16,
 "metadata": {
  "_cell_guid": "a91caed3-dab3-4151-99c4-5d7d6853f745",
  "_uuid":
"8c2d9a6eacd3d3f27be1c3c4bc042d839c778b69"
 },
 "outputs": [
  {
```

```
   "data": {
    "text/plain": [
     "0.0    76\n",
     "1.0    29\n",
     "Name: class, dtype: int64"
    ]
   },
   "execution_count": 16,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "y_train.value_counts()"
 ]
},
{
 "cell_type": "markdown",
 "metadata": {
  "_cell_guid": "4f9fa8e7-cc41-4776-86ba-654d8f225d0a",
  "_uuid": "13430eedffd5ecf5a1d70ab2cc86271d91a00d47"
 },
```

```
  "source": [
   "## Choosing parameters with GridSearchCV with 10-fold cross validations.\n",
   "(Suggestion for next time: try using Bayesian model selection method)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 17,
  "metadata": {
   "_cell_guid": "53b832b7-b6ed-4efc-93fe-e1c8d312bdc1",
   "_uuid": "068ddf02dceed8fc7f18c6d7bba431bfbd7de056"
  },
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Detailed classification report:\n",
     "             precision    recall  f1-score   support\n",
     "\n",
```

"   0.0   1.00   1.00   1.00   39\n",
"   1.0   1.00   1.00   1.00   14\n",
"\n",
"   accuracy   1.00   53\n",
"   macro avg   1.00   1.00   1.00   53\n",
"weighted avg   1.00   1.00   1.00   53\n",
"\n",
"Confusion Matrix:\n",
"[[39 0]\n",
" [0 14]]\n"
]
},
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAYoAAAEWCAYAAAB42tAoAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nO3dd3gUVffA8e8hhB5CE0SiUkKHEKQKIKEh5KRZApdoQLBAJIlUFUSxoPwQMIAvLyoqCCgKqKAIKIKIUkMIHSFIAvLyoqCCgKqKAIKIIKUkMIHSFIFAVCDVIScn5/7CaGkGyWkM2mnM/z7JOdmTszZyfJnpl7Z+4VVcUUYY4xJSZxJSZ5vB2CMMSZ7s0RhjDEmVZYojDHGpMoShTHGmFRZojDGGJMqSxTGGGNSZYnCGGNMqixRGJOFRKSniKwWkTMi8qeIrBeR/iISJSI3E/l8NBJRLS3iKwVkZMicjqT4mXmEiGZTyRCRP4SkRMiMl5E8mQwjuEiMs27RxrjPZYojMkiIvIA8BLwAlAJKAP0A1oAcUBxESnkdpmvyXdPVb8BoC+QCzQE8VLQq8B1zAXm8e4lGlQpWpqnwRpgvsUSRW6QBtBxVD3N1XQNIA94pBLPn8WuGVJI8Z6kaUgbB9AcP6ouhTA72tWfqzXz99WW4vaxI9D5EwCWV2GS64rYvSfq4XxU2nbWiZTATaPkBNaDdK7rvtadZGKKmChkW1W8Slo42/FUvUnPJ/Si1dRpGpddjsW9nvfyhmPZU5udVDXN9g5CmgnsXeG2n//Qv6Scn+4ci5Nxx+4ZxVkHM0S8TzJcrbpJsVQ+mnsyxISyQ+xoeEncvqpsdJFEvJNdeUQfcXRehcvXpVKLRqwJVBSYJIBKRGQJsirfk+zyG0OkuYU==

VLFMYYY1yyRGGMMcYlSxTGGGNcskRhchURiRKRcyI
SKyJ/icgsESmWokwzEfleRM6IyCkR+UpEaqUoU1xEJonI7
85t7XNOl0ljvyIiA0UkUkTOiki0iHwmInU9+XmNyQqWKE
xu1FFViwHBQH1gROICEWkKfAcsBm4AKgFbgZ9FpLKz
TAFgJVAb6AAUB5oBMUDjNPY5GXgGGAiUAqoBi4C7rz
Z4Ecl/tesY40liT2ab3EREooAnVHWFc/pNoLaq3u2cXgNsU
9X+Kdb7Bjiqqr1E5AngVaCKqsa6sc+qwC6gqaquT6PMK
mC2qs50Tvd2xnm7c1qBAcAgID+wDIhV1WHJtrEY+FFVJ
4rIDcA7QAsgFnhbVae4cYiMuWp2RWFyLREJAO4E9jmn
i+C4MvgsleKfAm2d7/8DfOtOknBqA0SnlSSuwr3ArUAt4B
Ogh4gIgIiUBNoB80QkH/AVjiuhCs79DxKR9te4f2NSZYnC
5EaLROQMcAg4Aox2zi+F42/+cCrrHAYS2x9Kp1EmLVdb
Pi2vq+pxVT0HrAEUaO5c1hVYq6p/Ao2A61R1rKpeVNX9
wP+AnpkQgzFXsERhcqN7VdUPaAnU4N8EcAJIAMqnsk5
54JjzfUwaZdJyteXTcijxjTrqhOcBDzhnPQjMcb6/GbhBRE
4mvoCRQLlMiMGYK1iiMLmWqv4IzAImOKfPAmuBbqk
U746jARtgBdBeRIq6uauVQICINHRR5ixQJNn09amFnGJ
6LtBVRG7GUSX1uXP+IeCAqpZI9vJT1bvcjNeYq2KJwuR
2k4C2IhLsnH4eeNR5K6ufiJQUkXFAU2CMs8zHOL6MPxe
RGiKST0RKi8hIEbniy1hV9wLTgLki0lJECohIIRHpKSLP
O4uFA/eLSBERCQQeTy9wVd0CHAVmAstU9aRz0XrgtIg
8JyKFRcRHROqISKOMHCBj0mOJwuRqqnoU+Ah40Tn9
E9AeuB9Hu8JBHLfQ3u78wkdVL+Bo0N4FLAdO4/hyLgP8
msauBgJhwFTgJPAbcB+ORmeAt4GLwN/Ah/xbjZSeuc5Y
Pkn2mS4BHXHc/nsAR5XZTMDfzW0ac1Xs9lhjjDEu2RW
FMcYYlyxRGGOMcckShTHGGJcsURhjjHEpx3U+VqZM
Ga1YsaK3wzDGmBxl06ZNx1T1uoysm+MSRcWKFdm4ca
O3wzDGmBxFRA5mdF2rejLGGOOSJQpjjDEuWaIwxhjjk
iUKY4wxLlmiMMYY45IlCmOMMS55LFGIyPsickREItNY
LiIyRUT2iUiEiNziqViMMcZknCevKGYBHVwsvxOo6nz1B

aZ7MBZjjDEZ5LEH7lR1tYhUdFGkM/CRc8jHdSJSQkTK
q6rLsYf3Hz1Lj/+uzcRIjTEm9zqyZwt7Vs67pm1488nsCiQb
IxiIds67IlGISF8cVx0UvD4wS4Izxpic7PyZE0R8PpWodUsp
WvrahnT3ZqKQVOalOoqSqs4AZgCUurmmzu/X1JNxGW
NMjtelSxeiN37HiBEjGDVqFEWLujsE/JW8eddTNHBjsuk
A4E8vxWKMMTne9u3b+eOPPwAYP3484eHhvPbaaxQpU
uSatuvNRPEl0Mt591MT4FR67RPGGGOudPbsWZ5//nmC
g4N54YUXAAgMDKR27dqZsn2PVT2JyFygJVBKKB0
YAvgKq+CywF7gL2Af8AfTwVizHG5FZLliwhNDSUgwcP8
thjjzF+/PhM34cn73p6av/GGJPbTZs2jN3p6IJ3lCoR6av/GGJPbTZs2jdDQUGrV
qsXq1atp3ry5R/aT48unfvzrlz
53j66acpUKCAx/YpjhP7nKPUzTX1+MGd3g7DGGOy3Pr1
6+nXrx/58+dn3bp1+Pj4YYZ2a/19WSMMdncyZ
Mn6d+/P02aNOHIkSM8dEmG4qLi8PX15dq1a
bti1Hjx5l4MCBjB07luGDmRYojDGmGzj+++/JgoiNGjR3PLLbbd
w4cIFb4cEWKIwxhivO3NGqh2rQhPj6eb7/9Q+nnl9G6xdfjuH
dGlvhwZYojDGGK8rVKgQx44dY9SoUURGRtK+fXvh3
ZSxTGGOMFERERtG/fnujoaESEJUuWvW8m3Qvh3
YFSxTGGJOFzp49m3QH0+bNm7TG
GJPLfPnll9SqVYsJEybw1qq1auXtsNJlz1EYw0w
WWbRoEcWLF+enn37q8+nn37itd/tu83Y4brO
KrVq14pZbbuuHMmTMUKlQFLI/fLI/F+noyxphsZt26dTRs
2JBhw4bx6efAuDn5+eVJHtLFEYY0wmOmnOHp556
r1ozjx4+zcOF9dW+HdU0sURURhjTCaaa7d69m6NHj3L77
PDgwezYsYN7770/XEfF2WNfE2iiiMMeYa7d69m6NHj3L77
bdz4cIFdu/eTVBQkLfDuN0ExQURG
hoKKpKwYIFs12LDZJb1JwG8wKWKWKIwx5hIdOoPDhxgyZAhjBo1itKlS3s7nBxDFMa
YFKwgcxWvOPsOaGDt
gOWL19OyvgYIwx15E1a9awYcOGvbvTXXXdx5MgRHn30UQoWLOjt0HIMm8kl6vvdr

ZA8yq4ojDHGDZcuXWLbtm0ANG/enPfee4+IiIhcnyTAEo
UxxqRry5YtNGvWjNtuu42///4bEeGxxx6jUKFC3g4tS1iiM
MaYNJw5c4YhQ4bQsGFDoqKimD59OmXLlvV2WFnO2ii
MMSYVp06dom7duhw6dIh+/frx+uuvU7JkSW+H5RWWK
IwxJpnTp09TvHhx/P396du3L23atKFp06beDsurrOrJGGN
wdOD35ptvEhAQwObNmwEYNWpUnk8SYFcUxhjDzz//T
EhICJGRkdx7771cd9113g4pW7ErCmNMnvb0009z++23c+r
UKRYvXsZChQu58cYbvR1WtmKJwxhj5yTv4+76669n2LB
h7Nixg06dOnxquzLOgU0xuQpu3btIiQkhMGDB9O5c2dv
h5NlrFNAY4xJx7lz53jxxRcJCgpi69atZHH5tn9atsh5RgeTRQi0k
FEdovIPhF5PpXl/iLylYhsFZHtItLHk/EYY/KmlStX
caNG0fPnj3ZvXs29HZYOYbH7noSER9gKtAWiAWY2i
MiXqrojWbFJa1bt/Z2ODmOJ2+PbPbQzsU9X9ACIyD+gMJE8UCvi
Jo2/eYsBxIN6DMRlj8oBLly7x7rvvvUqBAAZ588kl69epFz54
9KViwoLdDy5E8WfVUAThY3gg2bJlAIiYlr4MlEkdoHHHjF37
lw+++wzb4eVK3gyUQDyZ9ZBcH+AHHBBq
eDAmY0wutXXrVsLCwRsLCwggJCWHXrl307Nkz1444S
AFVFpJKIFAB6Al+mKPYPMFcHozJGJOL
HDhwgPfffx9wDCa0b98+XkY1e7isUShShqvHAA
GAZsBP4VFW3i0iNIDNBbcBK4DlVPePpj
PFixd5/fXXVrFkOHDuXEiRJ72U72ZLYx
JkdZs2YNIsEh7Nixg/vvv39t1ta1XXbFm7yWi9t5zGg
G5BhHjx6lXbt2vz9j/bBLWzt7KMmtmTPGAurMWN1NNVli
9fDsB1113H119zZn1zfrr6v9/zZft2y1ZCTFMaYbGv
WrVgHQpk6bihYcfmkGhJYc+Mb75PQ1zhdHUtFMaYOeFHH+
7dmbOnEmLFimLi28HVaeZW69cmp59vFQ

Wtt96yEee8zBKFMSZbOHz4MGXLlsXHx4eRI0fi7+9Py5Y
tvR2WwaqejDFedunSJaZMmUL16tWZNm0aAJ07d7YkkY
1YojDGeM3GjRtp3LgxzzzzDM2aNeOuu+7ydkgmFW4nCh
Gx2wyMMZnmzTffpHHjxhw+fJj58+fzzTffUKVKFW+HZ
VKRbqIQkWYisgNHNxyISD0RmebxyIwxuY6qkjhUQOPG
jQkNDWXnzp10797dOvDLxtLtwkNEfgW6Al+qan3nvEhV
rZMF8V3BuvAwJmf67bff6N+/P3Xq1OH//u//vB1OnnMtXX
i4VfWkqodSzLqUkZ0ZY/KeCxuMG7cOOrUqcPatWuik
Hcuf22EMi0gxQZ3fhA3FWQxljjCubNm3i4YcfZteuXXTr1
o1JkyZxww03eDssc5XcSRQhwGDG5DfjZMy
yhWrBgiwtKlS7nzzju9HY7JIHcSRXVVfSj5DBjZMy
EZY3KqhIQEPvjgA9auXcvMmTOpXr06kZGR5Mtd+Ln
ZO789t5xc54xJg+LjIykRYsWPPHEE+zdu5ezZ88CWJLIB
dK8ohCRpkAz4DoRGZFsUVHAzj5rFOmbCmxDnhxwDtPfj
iTpyIv78/H3zwAY8++qjd7pqLuKsqPdbAUoQLf5p/SYDQ
dt5wanUb3DtCvOwPtOGPj7fnQTZA1
AgTFnxPnyaxwQMl7g3jnBrA8PjH24Qp/Cnxz7Jx7j+LVT
QS7uekTL9eh4F2H/UTvjJYQvL31PpK5nqPlF9ugV2pqF
lWrGNJv3iWyhnS7jUO8hcwE5fqW0cmHQOx8Dg2J0PX9oe
gLeBb5vh/Bcpy1r5rXnrBw6kHvAVpALUULe0QmjoDUvBG
wsHBmq7GSAPCwmHgY9eZXVTESxtTEH1J3/FJDLCBlD8DQWMz
+K/Hdmd5zlUomqwycmmx3t6b5Wt1+G48qSs7b5ZYm++9E
CeF0ERyM5s1Z+88/E48OQcz1Rz1h3EN/3DXKG8E1nC04FmMw
tHJfN6sgxvq/BTsy0gaRtSD

m3HVmH1RVRMAVPU8sMeShDG5Q0REBLfffjv9+vUjO
DiYMWPGeDskk425uqKoISIRzvcCVHFOC6CqGuTx6Iwx
mW7BggX07NmTkiVL8tFHH/Hwww/bMxHGJVeJomaWR
WGM8bjTp09TvHhxWrZsSWhoKKNHj7auN4xb0u0UML
uxxmxjrs7vv//O008/zZ9//sm6devw8bFRAvIiT3cKmGEi0kF
EdovIPhF5Po0yLUUkXES2W9cgxmSeuLg4JkyYQM2aNV
mxYgXdu3cnp50YmuzBnS48MsT5HMZUoC2OsbY3iMiXq
rojWZkSwDSgg6r+LiJlPRWPMXnJwYMH6dSpE/bit4
s2JF33nmHm2++2dthmRzKrUQhIVX1W7LaIhNtMpTTlnbTNaN8jH9vmFWNLLEddRQd1Pj15zZBXHoy8
X3O7cxD+gM7EhW5kHgC1X9HUBVj1zF9o0xKagqIsL111999vbAsXRrkWSWLlSVcTB3Lt9pvz2daM7R
+dGPbFYBDyaajnfOSqwaUFJFVIrJJRRjlzCWuZaSQFX
DgYpurJfaX2fKCtL8QAPgbqA98KKIVLtiQyWyGy
Mi4tLudiYPG337t20adOGRx55hPz58xMTE+PtkEwu406iF
fVUxnYdjSOLkASBeDoojxlmW9V9ayqHgNWA/VSbkhVZ
6hqQ1Vt6Ovrm4FQjMl94uPGT16NEFBQezfv9/6MVm5xGRPQWzzy
+/WFuEyXTuJIpIEXkQ8BGRqiLyKbv31Rg3+Pj4jbj4Grp
KQD0BFJWWS0GmotIfhEpgqNNzu54IwXs7EAbUFmvvcNNa4/LVm+ZcoQb4x8mS8x/i1zwX/7+W
27cru3bsJCQkhXz6P3sho8ih3BPcHQ3nu54F
KoaDwwAluH48v9UVbeLSIiIhDjL7MTR9PchCBHMHCpwZTeSC6xpWmhatBaSTF/+
W7gN3IlJfVbdkUTzpsgfuTF506dIlZCYsZ8+
mW7du3g7L5Clz9u20KMm4LVCefuBuoojsEJHhInKfs/+YMNXg4RHk
xZM/r370/Dhg3Ztm0bvXv3toHEjrZ8sFUGFxHUHYDKqXvRIp3aCQy00GVjh1
Rnl6cCMMQ5hYWFERUexcUZGRp06dZjcnV44tJbJVVlH+eLlg1SE8dL5n6tlRuhLVN5aI3aKyMyV
CTyJSF3gW6KGqBwPlQtW9Wl7kVI3Dh/ba63z
PixAkASpYs6eXITE0m0aonEakpDq0M2Ae2JawJHLN
0cQmoFoOIvuqeFaM9krxQSJhvjxAfaQuVzRSZmTf/3YBKF3

vh2UM4EaiUNUmWRGIMXnZf//7X55//nnuvfdeJk+ezE033
eTtkIxJkmaiEJFPVbW7iGzj8q43bIQ7YzJBTEwMUVFRN
GjQgCeffJLAwEA6dOjg7bCMuYKrK4rE6957siIQY/IKVe
Wjjz5i2LBh+Pn5sWfPHgoWLGhJwmRbaTZmq+ph59v+q
now+QvonzXhGZO77Ny5k1atWtG7d2+qVq3KokWLyJ/fY
8PCGJMp3Hngrm0q8+7M7ECMye22bt1KvXr1iIiIYMaM
Gfz0008EBVkNrsn+XLVRPIXjyqGyiEQkW+QH/OzpwIzJ
LaKjowkICCAoKIgxY8bw+OOPU7asDeZoco40H7gTEX+g
JPA6kHy86zOqejwLYkuVPXBncoo///yTgYMHs3TpUnbt2
kWFCinH7TIm63jVRvEUjyqgTtV1SggFDiT7IWIlMrIzojRR
YWBg1a9Zk8eLFPPvss5QpU8bbYRmTYa5a0T7BccfWz587
xyYfsU6Byh6My5gc6fz587RoYoDtm1bpk2bRmTYa5a0T7BccfTJhy3
fDMuaapJkoVPUe589KWReOMTlTXFwcvr6+FCpUiFatW
jFkyBB69Ohh41WbXMGdvp5uE5GSACLSTUQ+EpHSHRo3B8
UzEggULCAwMZPPmzQCMHz+enj17Wpl[IwuYY7t8dOB/4
RkXo4eo49CHzs0aiMyQH279/P3XffTbdu3ShdurQNQ2pyL
Xf+suPVcWtUZ2Cyqk7GcYusMXnWxIkTqV27Nmv[WrGH
SpEmsX7+e4OBgb4dljEe480joGREZATwCNBcRH8DXs2E
Zk73FxsZy1113MXnyZAICrNd9k7u5M2b29cCDwAZVXe
Nsn2ipqh9lRYAp2XMUxhuOHTvG8OHDue++++jUqRMJ
CQlW1WRyFI8OXKSqfwAH4760kYUxWS0hI4P
3336d69erMnj2bffv2v2AViSMHmKO3c9dQfWA92A7sCvItL
V04EZ4207duygZcuWPP7449SqVVYvw8HCGDBni7bCMyX
LutFG8ADRS1SMAInIdsAJY4MnAjPG2jRs3sn37dt577z16
9+5tVxEmz3KnjWJbqtZVnlcLiUmJQRqLSGSz9N2e+xJ4tbx
oZHHnkEVeXEiROUKmU91pm4FWAF23kV8ARL0CWNbwyvzF7O
PBfIAioB8xQ1ec8HZgxWSE+Pp7Pp7K0PJN9J6ZmuAWjAIiWwNahj9RLnRSJzedj93ZNpVZxkJudVsqga
NWvsqWpjknE1HkVVYAJQBdgGDgDQqEgdGDFPVP7IqMGOywzNqZN
mxg0aBAdOnRg6tSpVK5aP
k5KrK4r3ga+BLjh6kH0nSy

IyxsNOnTrFF198AcCtt97Kr7/+ytKlSy1JGJMGV3c9+anq/5zvd4vI5qwIyBhPUVU+/fRTBg0aRExMDFFRUdxwww00btzY26EZk625uqIoJCL1ReQWEbkFKJxi2pgc47fffuPOO++kZ8+eVKhQgV9++YUbbrjB2EZkyO4uqI4DExMNv1XsmkFWnsqKGMy05kzZ2jQoAEJCQlMmTKF/v374+Pj4+2wjMkxXA1c1CorAzEms0VERBAUFISfnx/vvfceTZo0sXGrjckAe9TU5DpHjx7l0UcfpV69eqSnp7Nx40ZmzZpFjRo1rNzSvX9/bYRmTZ7lT9XTZ2NjOnmP7eSwik+fFxcURERHB1KlT6devn3XgZ4yXXfWT2aq6WUQaeSIYk3etXLmSJUuWMHHHHiRKpVq8bBgwcpVKiQt8MyxuBeG8WQZJP5gFuAox6LyOQpf//9N0OHDmXOnDlUnDlUqVKFF154gdKlS1OqVKlCnYbceeBO79kr4I42ixS64rDGLclJCTw3//+lxo4gdKlS1uSMCYbceeBO79kr4I42ixS64rDGLclJCTw3//+lxo

1avDpp5/y4osvsm3bNkqXLu3t0IwxKbi8onA+aFdMVYdnU
Twmjzh16hSjRo0iODiY6dOnU6NGDW+HZIxJQ5pXFCKS
39m1hg17ajJFbGwsEydO5NKlS5QsWZJff/2V77//3pKEMd
mcqyuK9TiSRLiIfAl8BpxNXKiqX3g4NpOLLF68mKeffpp
Dhw4RHBxM69atqVy5srfDMsa4wZ02ilJADI4xsu8BOjp/G
pOugwcP0rlzZ+69915KlCjBzz//TOvWNty6MTmJqyuKss47
niL594G7RCl7gTXmCqpK165d2bFjB2++azaDBg3Cev81Ju
dxlSh8gGK41124MUnuZXU7duZXU7duXZ588kkAqlWrRkh
IiCUJY3Ip+882bvvrr7948MEHadeuHSLCgAEDvB2SMSY
LXHU34yZv+uGHH7jvvvs4d+4z18GpMHmG
JwrgUFxeHr68vQUFBtG3blldffZVq1ap5OyxjTBayqieTqjN
nzjB48GCaN2/OpUuXKF26NJ999pklCWPyIEsU5jKq
EHNmjWZPHky9evX58KFC94OyxjjRZYoTJJjx47x47Rs
unTpQpkyZfjll1+YPn06RYoxhgvskRhkvj5+fH3338z
ceJENm7cSJMmTbwdkjEmG7BEkcf99NNP3HnnnTGxlK
wYEF+/fVXBg8eTP78dp+BMyWTUM5NP8EHz5
s3ZsWMH+/fvB7CH5owxV7BvhTxGVZk1axbbZ1dn1qxZD
B8+nB07dhAUFO0Iwx2ZTVL+RBH330EdWrV+fdd9+lb
t263g7HGJPN2RVFHnDu3DlGjx5NdHQ0IsLnn3/OOmjjVrL
EkYY9xiiSKXW7ZsGXXq1GHs2LEsrwYgJIlS1pbhDHG
bfZtkUv9+eef9OjjRgw4dOuDr68vatWt59NFHvR2WMSY
qXHjxrF48WLGjh3L1q1badWrdDMsbkUKKq3o7hqpS6
uaYeP7jT22FkS5s2bUrqwC8mJoYTJ04QBjo7bCMMdm
AiGxS1YYZWdejVxQi0lZE9orIARF5Mc3yv8A4ljIhPP1i4jYG
JoZcPr0aQYOHEjjxo0ZOXIkAKVLl7ZkmRYrhdBcqPsqLRFG
cb0hjRxk8BijMqx9XRq4/amiLlfpuMvw12lZ6TYYLi

0wF7gRqAQ+ISK0UxQ4Ad6hqEPAKMMNT8eRGqspnn31
GjRo1CAsL46mnnmL27NneDssYk8t48jmKxsA+Vd0PICLz
gM7AjieqNqUAABiJSURBVMQCqvpLsvLrgAAPxpPrfPL
JJzz88MPUr1+fxYsX06hRI2+HZIzJhTyZKCoAh5JNRwO3
uij/OPBNagtEpC/QF6BY+SqZFV+OdPHiRfbv30+NGjXo2r
Ur586do3fv3tY3kzHGYzzZRiGpzEu15VxEWuFIFM+ltlxV
Z6hqQ1Vt6Ovrm4kh5iyrV68dOi3acf78eQoWLMgT
TzxhScIY41GeTBTRwI3JpgOAP1MWEpEKEpEgYCbQWVVjP
BhPjnXs2DH69Onfcwblz53j33Xdtvdn1fcwblz53j33XdtvGpjTJbx5KnoBqC
qiFQC/gB6Ag8mLyAiNwFfAI+o6h4PxpJj7d+/n0aNGnH69
Gmef/55Xnzx RRtIyBiTpTyWKFQ1Xk QGAMsAH+B9Vd0
uIiHO5e8CLwGlgWkiAhCf0ft8c5vTp09TvHhxKlWqRJ8+f
ejduzd16tTxdljGmDzIHrjLZv755x9GjxrWHX7oeuUVZsyYwdatWwkIs
BvBjDHX7loeuLNW0GxkYgLGKioPr06UPhwwoW9
HZIxxliiyA7i4+N54LBgATVr1uuTHH3+kRYsW3g7g7L
GGMA6xTQqxKr/fLnz0+5cuV47bXXRhjMlWLFF
4yYYNG7j11lvZvHkzAGFhYYwYMIcBQp4OTJjjLmcJ
YosdurUKQYMGMCtt95KdHQ0MTT95KdHQ0MTH26IgxJnuzRJGFEjv
wmz59OgMGDGDXrl20bdvW22EZY4xL1pidhXJbu3EmFC
hX46quvaNjQHhcxuQM9hyFB124cIG33rLevXq0bFjR+
Li4siXLx8+Pj7eDs0Yk8dk24GL8rIjR47QrEYGoYdqsGb/DweNOSsgLqyMaCKJyiVZF7cZv/B/T9CjvNEngbYklikpZuXp/wSHZ9vR+h58dmb2Zxa1vnzWJaMQ6E/JxNvA9RFtvc0SMc
ZcE2ujyATbtm1j0j7LPPUr9+fZ599lna/+fZ599lni4uIQEUsSxpxph

0/3+PLFFkQGRkJJ06daJS6daJ27dqsWbOGHHHkpQHO2
+/3dshZaro6Gj8/PyoXWLEiznFCjDE5gKoSEx0cffT19UVE0vz2vnbtGsP8+GzXr17VmvQaM9vH/xtYCMnhf4
yrTtWtWTm+Lj41m1ahUnzvvvv9gfLL2Lkka9Lka9ghv2bu1zD4
KlS1uSMCaHERHKlClDbGysq1ZGUdBlijc8Ouvv9KwYUPAtGnD3r1

**7Abj77rtzTVVTaixJGJMzeeJ/1xKFCydOnOCpp56iadOm
HDt2jM8++4zAwEBvh2WMMVnKEkUaLly4QP369ZkxYw
aDBg1i586d3H///XamnUV8fHwIDg6mTp06dOzYkZMnTy
Yt2759O61bt6ZatWpUrVqVV155heS3eX/zzTc0bNiQmjVr
UqNGDYYNG+aNj+DSAw88QFBQEG+//bZH95PWcYyKi
qJw4cIEBwcnvS5evOKOHD3PPPfd4IrR
MsWvXLpo2bUrBggWZMGFCmuUOHDjArbfeStWqVenR
o0fSMVdVBg4cSGBgIEFBQUnDDwcN54442k+cOGDeP777/33IdKTlVz1KvkTTXUk6Kjo5Pef/DB
B7p582aP7i872rFjh7dD0KJFiyZ9/+ijj+oLLy1ixcjAy8u7prWP3z4sN50000e2XZKaR3HA
cOaO3atTN1X664sz9XZYYNG6aLFi1ye3/+vvvvU6cuRIfeutt9Is161bN50+nTZumqqpLli
zRDh06aEJCgq5du1YbN26sqo7PUblylyoKG3btm2q+0ntfxjYqBn83rW7npzOnz/P+P
Hjee211/j000/p3LkzvXv39nZYXjfmq+3s+PN/GdZ2tSkR
2xttvlmzZtSkREBACffPIJt912G+3atQOgSJEihIWF0b
JDQ3nzzzTd54YUXqFGjBuAYPbB///5XbDM3nann36ajRs
3IiKMHj2aLl26UKxYMWJjYwFYsGABX3/9NbNmzaJ379
6UKlWKLVu2EBwczKMFCwkPD6dEiRIABAYG8vPPP5
MvXz5CQkL4/fffAZ0aRK33XbbZftu164dR44cIT
YcXX3yRZs2a8fPPP9OpUyeCg4MZNmwfHxNGrUiOnT
p1OwYEEqVqzIgw8+yA8//EEBcXBwzZsxgxIgR7Nu3j+HDh
xMSEuL2cXTrFmzWLRoEZcuXSIyMpKhQ4Qy8eJFv7
4YwoWLMjSpUspUaoU4eHHhhISE8M8//1Ch5UClShSy
aZNm3jsccoUqTIZTd9XLp0ieeff55Vq1ZZx4cIFQkND6devn
8tYPv/8c4rjNGc4rjweeeQRzp49CzgGBWrVuizJgx
lC9fnvDwcLZt25bqfqpUqUKDBg0iPG0G0blzZPG0blzZ7e
PS2rKli1L2bJlWbJVvC+wSVpLVJXv/+eTz75BIBH32ma

eeeorFixfTq1cvRIQmTZpw8uRJDh8+TFRUFIGBgUnPZv
Xs2ZPFixdTq1Ytbr75ZmJiYvjrr7+4/vrrryn+9FjVE7By5Uq
CgoJ4+eWX6dKlC7feequ3QzJOly5dYuXKlXTq1AlwVDs1
aNDgsjJVqlQhNjaW06dPExkZecXy1Lzyyiv4+/uzbds2IiIia
N26dbrr7NmzhxUrVvD222/TuXNnFi5cCDhudqhYsSLlypX
jmWeeYfDgwWzYsIHPP/+cJ5544ortfPll1SpUoXw8HCaN
28OwMmTJ/nxxx8JDQ2ld+/ezJ8/n23bthEfH8/06dOT1r3xxx
htZu3YtzZs3p3fv3ixYsIB169bx0ksvuYw95XEE+O2335Kqn
UJDQ1NdLzIykk8++YT169fzwgsvUKRIEbZs2ULTpk356K
OPAOjVqxfjx48nIiKCunXrMmbMGAD69OnDlClTWLt27
WXbfO+99qr/D392fDhg1s2LCB//3fxw4cCDN2A8cOEDJkiU
pWLAg4PhSXr58OZs3b2b2OZs3b2bFj
R5r7KVSoEAsXLmTz5s388MMPDB069LKqy0Q9evS4rGo
u8ZX4ua9WTEwMJUqH9+x/l5QEAAf/zxBwB//PEHN95
4Y1LZxGVpzU90yy238PPPP2conquR5680BBg0axOTJkwkM
DOS7776zgYRSuJoz/zbme463bp1S+r9t0ePHowdwdO5Y
+ffowb948evTokbTdHTt2JK/e/f/du
KlWqRLVq1QDHWefUqHWefUqVMZNGgQQNIXfd26dYmbn
NjcXP
zw8/Pz8KFSrEyZMnk65wDHWefUqVMZNGgQQNIXfd26dYmbn
x44dk/YfERHBqVOPi4iIiPPPMI333
wDOPpDi4iIYMGCBYj1Me9e/cmfeaUDh8+zHXXZc0c0H
RcXx4ABAwgPD8fHx4c9e/YkLS8vHS8wNp7ScgIICRI
0eyevVq8uXLxx9//MHkf/99e/xVn5/PnzXR6bq5Vko8r
mat1wJE0//zzz0yMMnV5MlEkJCSgqvj4+BAIG/wNCeeuklRo
wYQaFChbwdmnEp1invuuuYepU6ycycOBAa
teuzerVqy8ru3//fooXL46fnx/1atX1mpaZDy6vWWoqbiq
elvBe9aNGiSe+bNm3Kvn37+qn20nBrz7r7d
q1FC5c+Ko+b+K2U/ti+SC7xOn4+Pjry9+1lnR0lpvp
p9JN9/avtLi5CqhqyrvvPMO/PAfuLE/Fy582f/e/lf

bffply5cmzdupWEhITL/m+T/47S2s+sWbM4evQomzZtwtf
Xl4oVK6b63EGPHj3YvXv3FfOHDBlCr169Uo3VlTJlynDy5
Eni4+PJnz8/0dHR3HDDDYDjSuHQoUNJZROXXbx4MdX
5ic6fP3/Vf2sZkeeqnrZu3UqzZs2YOnUqAA8++CBjxoyxJJF
N+fv7M2XKFCZMmEBcXBwPPfQQP/30EytWrAAcZ8wD
Bw7k2WefBWD48OG89tprSWeZCQkJTw48YrttmvXjrC
wsKTpEydOAFCuXDl27txJQkJCUtVSakSE++67jyFDhlCz
Zk1Kly6d6nbTO2NPqUaNZN9NKFGTvnSFsHaNqRQcUo+X58u
8jhmFn9/f0qWLMmaNWuAf+MtUaIE/v7+PTTTwDMmT
MnaZ327dszffr0pDj27NmT1N6QmmrVql2WRE6dOkX58u
XJly8fH3/8MZcuXUp1vbT2c+rUKfqWLYuvry8//DDBw8
eTHX9+fPnEx4efsUrI0kCHH8zrVq16Jz96Fe2qu1
RHH32EqrJu3Tr8/f0pX748jRo1Yu/evRw4cICLFy8ybb968y6
oP9+zZk+7dZJkhzySK2NhYg4dSoMGDdi/f7/HG39M5qlf
vz716tVj3rx5FC5cmMWLFzNu3DiqV69OHMGG
DAAgKCiISZMm8cADD1CzZk3q1KnD4cOHr9jmqFGjOH
HiBHXq1KFevXr88MMPPALzxxhvcc889tG7dmvLly7uMq0e
PHsyePTup2vhgpQpbNy4kaCgIGrVqsW77757VZ+1UKF
CfPDB3Tr1o26desmNY5nhuTHMTN9+OGDB8+nKCgI
MLDw5PaSz744ANCQ0Np2rTpZWe9KWW2
6hTp069OvXz+XVSdGiRalSpUpS8uzfvz8ffvghTZo0Yc+eP
ZddRSSX1n4eeughcSMOGDZkzZ07SgpIgI
JEycybtw4AgICOH3acRPH78eCZOnEhgYCA
xMTE8/vjjjSWUqV65MYGAgTz75JNOmTQMcN2OEhYX
Rvn17atasSffu3ald21EdHBcXx759
OnD9HR0fTt25c33njDrTrpvGrnzp3UrFnT22EYS
2rQp6c4nQ1KD/CuvvHLHFstT+h6lm/E80UZoEABSpUqx
fz582nWrWrJm3wzHGXKX77ruPmJgYb4eRKX77ruPmJgYb4eRrcTHxzN06NAs2
VeuvKKIi4ZukMJCEhgXz58kxN2zWxKwpjcr
bMvqLIdd+cv/zyCw0aNMZZ59NapQELElcpZx2AmGMc
Vcpx0rMfJp0v0a9qzyatAJ2

fDE/26u+fY8fvw4ffv25bbbbuPkyZMsWrSIzz//3BJEBhQqV
IiYmBhLFsbkMOocjyKz7+LMNW0UMTExfPLJJwwbNoz
Ro0dTrFgxb4eUYwUEBBAdHc3Ro0e9HYox5ioljnCXmXJ0
oti9ezfz58/npZdeomrVqhw8eDDpfnaTcb6+vpk6OpYxJmfz
aL2MiHQQkd0isk9Enk9luYjIFOfyCBG5xZ3tnjt3jpdeeimp
m+bEJxctSRhjTObz2BWFiPgAU4G2QDSwQUS+VNUdyY
rdCVR1vm4Fpjt/pinufCx169blt99+46GHHuL//u//KFeunGc
+hDHGGI9WPTUG9qnqfgARmQd0BpInis7AR86+0teJSAk
RKa+qVz5K63T22GHy+VdmxYoVtGnTxoPhG2OMAc8mi
grAoWTT0Vx5tZBamQrAZYlCRPoCfZ2TF/bu3Rv5n//8J3
OjzZnKAMe8HUQ2YcfiX3Ys/mXH4l/VM7qiJxNFal1Hprzf
0p0yqOoMYAaAiGzM6EMjuY0di3/ZsfiXHYt/2bH4l4hszOi
6nmzMjgZuTDYdAKTsON2dMsYYY7zIk4liA1BVRCqJSA
GgJ/BlijJfAr2cdz81AU65ap8wxhiT9TxW9aSq8SIyAFgG+A
Dvq+p2EQlxLn8XWArcBewD/gH6uLHpGR4KOSeyY/EvO
xb/smPxLzsW/8rwschxnQIaY4zJWtYRkjHGGJcsURhjjHE
p2yYKT3X/kRO5cSwech6DCBH5RUTqeSPOrJDesUhWrp
GIXBKRrlkZX1Zy51iISEsRCReR7SLyY1bHmFXc+B/xF5
GvRGSr81i40x6a44jI+yJyREQi01iese9NVc12LxyN378BlYE
CwFagVooydwHf4HgWownwq7fj9uKxaAaUdL6/My8fi2Tlv
sdxs0RXb8ftxb+LEjh6QrjJOV3W23F78ViMBMY7318HH
AcKeDt2DxyLFsAtQGQayzP0vZldryiSuv9Q1YtAYvcfySV1
/6Gq64ASIlI+qwPNAukeC1X9RVVPOCfX4XgeJTdy5+8C4
Gngc+BIVgaXxdw5Fg8CX6jq7wCqmluPhzvHQgE/ERGgG
I5EEZ+1YXqeqq7G8dnSkqHvzeyaKNLq2uNqy+QGV/s5H
8dxxpAbpXssRKQCcB/wbhbG5Q3u/F1UA0qKyCoR2SQiv
bIsuqzlzrEIA2rieKB3G/CMqiZkTXjZSoa+N7PreBSZ1v1H
LuD25xSRVjgSxe0ejch73DkWk4DnVPWS4+Qx13LnWOQ
HGgBtgMLAWhFZp6p7PB1cFnPnWLQHwoHWQBVguYi
sUdXTng4um8nQ92Z2TRTW/ce/3PqcIhIEzATuVNWYLIo

tq7lzLBoC85xJogxwl4jEq+qirAkxy7j7P3JMVc8CZ0VkNV
APyG2Jwp1j0Qd4Qx0V9ftE5ABQA1ifNSFmGxn63syuVU/
W/ce/0j0WInIT8AXwSC48W0wu3WOhqpVUtaKqVgQWA
P1zYZIA9/5HFgPNRSS/iBTB0XvzziyOMyu4cyx+x3FlhYiU
w9GT6v4sjTJ7yND3Zra8olDPdf+R47h5LF4CSgPTnGfS8Z
oLe8x081jkCe4cC1XdKSLfAhFAAjBTVVO9bTInc/Pv4hVg
lohsw1H98pyq5rrux0VkLtASKCMi0cBowBeu7XvTuvAwx
hjjUnatejLGGJNNWKIwxhjjkiUKY4wxLlmiMMYY45IlC
mOMMS5ZojDZkrPn1/Bkr4ouysZmwv5micgB5742i0jTDG
xjpojUcr4fmWLZL9cao3M7iccl0tkbaol0ygeLyF2ZsW+Td9
ntsSZbEpFYVS2W2WVdbGMW8LWqLhCRdsAEVQ26hu
1dc0zpbVdEPgT2qOqrLsr3Bhqq6oDMjsXkHXZFYXIEES
kmIiudZ/vbROSKXmNFpLyIrE52xt3cOb+diKx1rvuZiKT3
Bb4aCHSuO8S5rUgRGeScV1REljjHNogUkR7O+atEpKGI
vAEUdsYxx7ks1vlzfvIzfOeVTBcR8RGRt0RkgzjGCejnxmF
Zi7NDNxFpLI6xSLY4f1Z3PqU8FujhjKWHM/b3nfvZktpx
NOYK3u4/3V72Su0FXMLRiVs4sBBHLwLFncvK4HiyNPG
KONb5cyjwgvO9D+DnLLsaKOqc/xzwUir7m4Vz7AqgG/Ar
jg71tgFFcXRNvR2oD3QB/pdsXX/nz1U4zt6TYkpWJjHG+4
APne8L4OjJszDQFxjlnF8Q2AhUSiXO2GSf7zOgg3O6OJDf
+f4/wOfO972BsGTrvwY87HxfAke/T0W9/fu2V/Z+ZcsuPIw
BzqlqcOKEiPgCr4lICxzdUVQAygF/JVtnA/C+s+wiVQ0Xk
TuAWsDPzu5NCuA4E0/NWyIyCjiKoxfeNsBCdXSqh4h8A
TQHvgUmiMh4HNVVa67ic30DTBGRgkAHYLWqnnNWd
wXJvyPy+QNVgQMp1i8sIuFARWATsDxZ+Q9FpCqO3kB
909h/O6CTiAxzThcCbiJ39gFlMoklCpNTPIRjZLIGqhonIl
E4vuSSqOpqZyK5G/hYRN4CTgDLVfUBN/YxXFUXJE6Iy
H9SK6Sqe0SkAY4+c14Xke9Udaw7H0JVz4vIKhzdXvcA5ib
uDnhaVZels4lzqhosIv7A10AoMAVHX0Y/qOp9zob/VWms
L0AXVd3tTrzGgLVRmJzDHzjiTBKtgJtTFhCRm51l/ge8h2

NIyHXAbSKS2OZQRESqubnP1cC9znWK4qg2WiMiNwD/qOpsYIJzPynFOa9sUjMPR2dszXF0ZIfz51OJ64hINec+U6Wqp4CBwDDnOv7AH87FvZMVPYOjCi7RMuBpcV5eiUj9tPZhTCJLFCanmAM0FJGNOK4udqVSpiUQLiJbcLQjTFbVozi+OOeKSASOxFHDnR2q6mYcbRfrcbRZzFTVLUBdYL2zCugFYFwqq88AIhIbs1P4DsfYxivUMXQnOMYS2QFsFpFI4L+kc8XvjGUrjm6138RxdfMzjvaLRD8AtRIbs3Fcefg6Y4t0Thvjkt0ea4wxxiW7ojDGGOOSJQpjjDEuWaIwxhjjkiUKIwxhjjkiUKFmOMMS5ZojDGGOPS/wMxy+Vkw4/agQAAAABJRU5ErkJggg==\n",

```
    "text/plain": [
     "<Figure size 432x288 with 1 Axes>"
    ]
   },
   "metadata": {
    "needs_background": "light"
   },
   "output_type": "display_data"
  },
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "Best parameters:\n",
```

```
    "{'class_weight': None, 'max_depth': 2, 'n_estimators': 8, 'random_state': 42}\n"
   ]
  }
 ],
 "source": [
  "tuned_parameters = [{'n_estimators':[7,8,9,10,11,12,13,14,15,16],'max_depth':[2, 3,4,5,6,None],\n",
  "                     'class_weight':[None,{0: 0.33,1:0.67},'balanced'],'random_state':[42]}]\n",
  "clf = GridSearchCV(RandomForestClassifier(), tuned_parameters, cv=10,scoring='f1')\n",
  "clf.fit(X_train, y_train)\n",
  "\n",
  "print(\"Detailed classification report:\")\n",
  "y_true, lr_pred = y_test, clf.predict(X_test)\n",
  "print(classification_report(y_true, lr_pred))\n",
  "\n",
  "confusion = confusion_matrix(y_test, lr_pred)\n",
  "print('Confusion Matrix:')\n",
  "print(confusion)\n",
  "\n",
```

```
   "# Determine the false positive and true positive rates\n",
   "fpr,tpr,roc_auc = auc_scorer(clf, X_test, y_test, 'RF')\n",
   "\n",
   "print('Best parameters:')\n",
   "print(clf.best_params_)\n",
   "clf_best = clf.best_estimator_\n"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "_cell_guid": "fcb927c2-e0d6-40bd-afd9-5ff2e340f1e6",
   "_uuid": "f6eaae9019e5a7838bd85b2f927687f41a5eec7c"
  },
  "source": [
   "## Examine feature importance\n",
   "Since I pruned the forest (*max_depth*=2) and decrease the number of trees (*n_estimators*=8), not all features are used."
  ]
 },
 {
```

```
  "cell_type": "code",
  "execution_count": 18,
  "metadata": {
   "_cell_guid": "c389c50a-3437-474b-a02f-7fdc7a5607fa",
   "_uuid": "78fa6089f36ed50c7888b6d32ae94e5e7e0f7bf1"
  },
  "outputs": [
   {
    "data": {
     "text/plain": [
      "Text(0.5, 1.0, 'Feature Importance')"
     ]
    },
    "execution_count": 18,
    "metadata": {},
    "output_type": "execute_result"
   },
   {
    "data": {
     "image/png":
"iVBORw0KGgoAAAANSUhEUgAAAskAAADlCAYAAAA
ChxGqJAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAAL
EgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cG
```

xvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwbG
90bGliLm9yZy+WH4yJAAAgAElEQVR4nO3debglVXnv8e
+PRgYRRKGdgLbRoEgMRNIMMU4YNSAqJiqBGIwoEoI
4JNFcYpyHqLkOV33QlquYiBJEr2gbUCARRQQijTIHTIu
ttC0igoCoIOG9f1Tt7k1xunt3s2ufob+f5znP2TW+q87Ztfdbq
1atlapCkiRJ0mqbTHcBJEmSpJnGJFmSJEnqMEmWJEmSOky
SJc1pSZYn+VWSXwz9PGwM+3zgMpOKtT
ZIXJzl3usshSX0zSZa0MXh2Vd1v6GfldBYmy6bTGVX9Dzd
ZyS9KGMEmWtFFKcv8kH0/y0JPPaZY9M8tUkP
0tyQ5JPJ9m2XXYisAD4Ulsr/XdJnpJkRWf/K2ub21rdzyX5
VJJbgBevLf4SZa8kRyf57yS3JnlbW/4Za8kRyf57yS3JnlbW+bzk9yS5JQkm7XrPiX
JiiSva49leZIXdv4On0zy0yQ/SPL6JJu0y16c5Jt
GLg99tj/3m73oFJvtPGvjbJm4f2v7At718k+WFbhn8YWj6v
Ldv32mO5KMlO7bJdk5yV5MYkc5eD3/zZK0wUySJW2
s/gW4E/gt4HHAM4Aj2mUB3gzat
rp/9pxHgHAZ8DtgU+vY74o9gf+Sxw6NC6DwG2BF7ZlfSxw6
NC6DwG2B3YA/gI4PsmjO2PChbfcBrg
EeBPw5cBRwfnvs27br3NZuty1wIPBXSZ7bKe1n
Mclj2vl/05b1mcA2wEuAXybmZCjgLOKmNfSjw4SS/vR5/I0n
aYCbJkjYGX0jy8/bnC0keAhwA/HX
Wyqjqrqm6vqp8C76NwNJIO+NJVdPbmm5qBrv3a4eZ9+4F0eDYVvA5Fn00
SVVcAlwNnVtUz8GWaxHvXvYG9rj
+TpwGnBwyHxG9TpcGnBwW3P9
p8DfV9WtVbUeUceC9w2NB2K6vqQ1V1Z1X9aqqCVNXXqu
qyqrqrqi4F/pV/pV7/r3eUlW/qqpLgEuAPdr5I
0MeBawvKo+0cb+NvD/MsXaUa1OcdsuxLeqsubwYHc9kp6W
jdwH+DHSQazNwGubZc/CPgg8ERg63bZTfeyDNcOvX7
uKP6CdDr381xfRDhqZvqqrbhqZ/QFNL
ZR7Skn2Ad5FU4O9GbA58NiJaVsYm4Hnta7SJEvaGF0t7cPj
OBfQZNOlqbAieuqzySNA7WJEvaGF0L3A5sX1tj/bVN

XgVv47gQJ2r6ptaJoZZGj76uzvNuC+g4m2hnZ+Z53hbdY
Vf9we0DZfGFgArARuAH5Dk5AOL/vRGso91TQ0TSKWA
DtV1f1p2i1nivWmci3wyDXM//rQ32fbtonHX424X0m6V0yS
JW10qurHwJnAe5Nsk2ST9sG3QROBrYFfAD9PsgPw2s4uf
kLThnfgu8AW7QNs9wFeT1ObuqHx+/CWJJsleSJNU4bPV
tX/AKcA70iydZKH07QRXlt3cz8Bdhw8GNjaGrixqn7d1tL/
2XqU62PA25LsksbuSbYD/g14VJLDktyn/dlrqC2zJPXKJFn
SxupFNE0DrqRpSvE54KHtsrcAewI307Tf/Xxn23cCr2/bOn
+mbQd8NE3C9yOamuUVrN3a4o/bdW2MlTQPDR5VVVe1
y15BU95rgHNpaoVPWMu++vgpcAO6Bs5aLD0kap
1RNdedMkjQXJHkK8Kmq2nG6yyJJs4s4xMk1yZIkSVKHSbIkS
ZLUYXMLSZIkqcOaZUYEmSJEnqcOaZEmSJKnDJFmSJEnqmJEj7m2//fa1
cOHC6S6GJEmS6S6GJEmS5rCLLrohqrhqDv4EzNAkeeHChSxdunS6iyFJ
kEmS5rCLLrohqrhqDv4EzNAkeeHChSxdunS6i
yFJkqQ5LMkP1rTM5haSJElSh0my
WYJEuSJEkdJsmSJElSh0myJEmS1DEju4CTpLls4bnd
B2y9914JhLIklaE2uSJUmSpA6TZEmSJKnDJFmSJEnqM
EmWJEmSOkySJUmSpA6TZEmSJKnDJFmSJEnqMEm
WJEmSOkySJUmSpA6TZEmSJKnDJFmSJEnqMEmWJE
mSOkySJUmSpI6RkuQk+ye5OsmyJMFSS5tf85LssfQ
suVJLktycZKl4yy8JEmS1IdN17VCknnAccDTgRXAhUm
WVNWVQ6t9H3hyVd2U5Nyy3JEmS1
JtRapL3BpZV1TVVdQdwMnDQ8ApVdV5V3dROXgDsO
N5iSpIkSZMzSpK8A3Dt0PSKdt6avBT48tB0AWcmuSjJket
fREmSJGmy1tncAsgU82rKFZP9aJLkJw3NfmJVrUxyf+CsJN+
ZSa6qqnOm2PZI4EiABQsWrEuxFAsSZIkqR+j1CSvAHJJ/AHY
RWNldKcnuwMeAg6rqxsH8qlrZ/nsdcBpN840Hqarjq2pRVS2a
P3/+aOMgfnZ4P5Vb/X09cCpN8K4qKrjq2p
RVS2aP3/+6EcgSZIkjdkoSfKFwG5Jdk2yKXAosGR4hSQLgC8Ah1fVd4fmb5lk68F74ADg8nEVXpIkSRqHdbZJrqo7kxwNnAHMA054VV1R5Kh2+WLgDcB9gQ8nAbizqhYBDwBOa+dtAnymqk4fy5FIkiRJYzJKm2Sq6kvAlybNWzz0/mXAy6bY7hrgUZPnS5IkSbOZI+5JkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUsdISXKS/ZNcnWRZkmOnWJ4kH2iXX5pkj1G3lSRJkmaadSbJSeYBxwEHAnYDzA/fHbnVfmJX1mpt1QSaQsL0gY3ZV5PYFD7BSMhbnN2UD5DSzxfHbBSwSyCVBvMoRwqeqmCFZDZ+mvKa4iV7zqqj/2pqUiZHGT4qBJJQnIyxeR/xKhG7yLnjB1S2eZBGQZaSUy8NYvbLQcmTdQxTZ2YYJ6OFsDVw4VUAi3ybuKL/YQJmk1xPPKkMvjK9z0g7ADM8cYVqMqqz1r8DAGK8pdbXiIUHYJwvlxDFX4uw/cI1X00uhIG

ODUdt6mwElV9ZVejkSSJEkak1HaJFNVpwOnd+YtHnp9
BHDEFNtdA+zRnS9JkiTNZI64J0mSJHWYJEuSJEkdJsm
SJElSh0myJEmS1GGSLEmSJHWYJEuSJEkdJsmSJElSh0
myJEmS1GGSLEmSJHWYJEuSJEkdJsmSJElSh0myJEm
S1GGSLEmSJHWYJEuSJEkdJsmSJElSh0myJEmS1GGSL
EmSJHWYJEuSJEkdJsmSJElSx0hJcpL9k1ydZFmSY6dY/
sIkl7Y/5yXZY9RtJUmSpJlmnUlyknnAccABwG7AoUl266z2
feDJVbU78Dbg+PXYVpIkSZpRRqlJ3htYVlXXVNUdwMn
AQcMrVNV5VXVTO3kBsOOo20qSJEkzzShJ8g7AtUPTK9
p5a/JS4MsbuK0kSZI07TYdYZ1MMa+mXDHZjyZJfsIGb
HskcCTAggULRiiWJEmS1I9RapJXADsNTe8IrOyulGR34
GPAQVX1s/XZFqCqjq+qRVW1aP78+aOUXZIkSerFKEny
hcAuSXZOshlwCLBkeIUkC4DPA4dV1XfXZ1tJkiRppllnc4
uqujPJMcAZwDzghKq6IslR7fLFwBuB7YAPJwG4s60Vnn
Lbno5FkiRJGotR2iRTVacDp3fmLR56fQRwxKjbSpIkSTO
ZI+5JkiRJHSbJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHS
bJkiRJUodJsiRJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsi
RJktRhkixJkiR1mCRLkiRJHSbJkiRJUodJsiRJktRhkixJki
R1mCRLkiRJHSbJkiRJUsdISXKS/ZNcnWRZkmOnWL5r
kvOT3J7kNZ1ly5NcluTiJEvHVXBJkiSpL5uua4Uk84DjgK
cDK4ALkyypqiuHVrsReCXw3DXsZr+quuHeFlaSJEmahF
FqkvcGllXVNVV1B3AycNDwClV1fVVdCPymhzJKkiRJEz
VKkrwDcO3Q9Ip23qgKODPJRUmOXJ/CSZIkSdNhnc0tg
Ewxr9Yjxh9U1cokDwLOSnJVVZ1zjyBNAn0kwIIFC9Zj95
IkSdJ4jVKTvALYaWh6R2DlqAGqamX7+3rgVJrmG1Otd
3xVLaqqRfPnzx9195IkSdLYjZIkXwjskmTnJJsBhwBLRtl5
kq2SbD14DTwDuHxDCytJkiRNwjqbW1TVnUmOAc4A5g
EnVNUVSY5qly9O8hBgKbANcFeSVwO7AdsDpyYZxDqp
qr7Sz6FIkiRJ4zFKm2Sq6nTg9M68xUOvr6NphtF1C7DHv
SmgJEmSNGkjJcmSJGnjsPDY0zZou+XvOnDMJZGml8N

SS5IkSR0myZIkSVKHSbIkSZLUYZIsSZIkdZgkS5IkSR0
myZIkSVKHSbIkSZLUYZIsSZIkdTiYyEbMDuMlSZKmZ
k2yJEmS1GGSLEmSJHWYJEuSJEkdJsmSJElSh0myJEm
S1GGSLEmSJHWYJEuSJEkd9pM8g9hvsSRJ0swwUk1ykv
2TXJ1kWZJjp1i+a5Lzk9ye5DXrs60kSZI006yzJjnJPOA44
OnACuDCJEuq6sqh1W4EXgk8dwO2lSRp1vCun7RxGKU
meW9gWVVdU1V3ACcDBw2vUFXXV9WFwG/Wd1tJkiR
pphklSd4BuHZoekU7bxT3ZltJkiRpWoySJGeKeTXi/kfeNs
mRSZYmWfrTn/50xN1LkiRJ4zdKkrwC2Gloekdg5Yj7H3n
bqjq+qhZV1aL58+ePuHtJkiRp/EZJki8Edkmyc5LNgEOAJ
SPu/95sK0mSJE2LdfZuUVV3JjkGOAOYB5xQVVckOapd
vjjJQ4ClwDbAXUleDexWVbdMtW1fByNJkiSNw0iDiVTV
6cDpnXmLh15fR9OUYqRtJUmSpJnMYaklSZKkDpNkSZI
kqcMkWZIkSeowSZYkSZI6TJIlSZKkDpNkSZIkqWOkLu
AkSZLmgoXHnrZB2y1/14FjLolmOmuSJUmSpA6TZEmSJ
KnDJFmSJEnqMEmWJEmSOkySJUmSpA6TZEmSJKnD
JFmSJEnqMEmWJEmSOkySJUmSpA5H3JM0Izkq1uzl/07
SXGBNsiRJktRhkixJkiR1jJQkJ9k/ydVJliU5dorlSfLBdvml
SfYcWrY8yWVJLk6ydJyFlyRJkvqwzjbJSeYBxwFPB
FyZZUlVXDq12ALBL+7MP8JH298B+VXXD2EotSZIk9Wi
UmuS9gWVVdU1V3QGcDBzUWecg4JPVuADYNslDx1xW
SZIkaSJGSZJ3AK4dml7Rzht1nQLOTHJRkiM3tKCSJEnS
pIzSBVymmFfrsc4fVNXKJA8CzkpyVVWdc48gTQJ9JMC
CBQtGKJYkSZLUj1FqklcAOw1N7wisHHWdqhr8vh44lab
5xj1U1fFVtaiqFs2fP3+00kuSJEk9GCVJvhDYJcnOSTYDD
gGWdNZZAryo7eViX+Dmqvpxkq2SbA2QZCvgGcDlYyy/J
EmSNHbrbG5RVXcmOQY4A5gHnFBVVyQ5ql2+GDgdeC
awDPglcHi7+YOBU5MMYp1UVV8Z+1FIkiRJYzTSsNRVd
TpNIjw8b/HQ6wJePsV21wB73MsySpIkSRPliHuSJElbiw
yJEmS1GGSLEmSJHWYJ

GGSLEmSJHWYJEuSJEkdJsmSJElSh0myJEmS1GGSLE
mSJHWYJEuSJEkdm053AbTxWHjsaRu03fJ3HTjmkkiSJ
K2dNcmSJElSh0myJEmS1GGSLEmSJHXYJlmSpBnM5z
mk6WFNsiRJktQxUpKcZP8kVydZluTYKZYnyQfb5Zcm2
XPUbSVJkqSZZp1JcpJ5wHHAAcBuwKFJduusdgCwS/tzJ
PCR9dhWkiRJmlFGqUneG1hWVddU1R3AycBBnXUOAj5
ZjQuAbZM8dMRtJUmSpBlllCR5B+DaoekV7bxR1hllW0m
SJGlGGaV3i0wxr0ZcZ5Rtmx0kR9I01QD4RZKrRyhb1/bA
DRuw3YaYZKy1xsu7jTfOeD0x3oTizYH3y1w+trXG8/iMtz
7xejKXj2/G/C1nWLyHr2nBKEnyCmCnoekdgZUjrrPZCNs
CUFXHA8ePUJ41SrK0qhbdm33MxFjGM57xNp54c/nYjGc
8401fvLl8bH3FG6W5xYXALkl2TrIZcAiwpLPOEuBFbS8X
+wI3V9WPR9xWkiRJmlHWWZNcVXcmOQY4A5gHnFB
VVyQ5ql2+GDgdeCawDPglcPjatu3lSCRJkqQxGWnEvao6
nSYRHp63eOh1AS8fddse3avmGjM4lvGMZ7yNJ95cPjbjGc
940xdvLh9bL/HS5LeSJEmSBhyWWpIkSeowSZYkSZI6TJI
lSZKkjjmRJCfZarrLMJsleeDafqa7fOPSdkW4xdD0lkkW9
hhvqySbDE1vkuS+fcWby5LMS/Kp6S5HX+b6eyXJf4wyb8
wxn5PkPe3Ps3uO9bg+969+JXlAkt2T7Dn46SHGtHzPJnnB
KPNmu77ywFn94F6SxwMfA+5XVQuS7AH8ZVUd3VO8R
wAfAH4fuAs4H/jrqrqmp3hf4p4jFN4MLAU+WlW/HlOc73
P3ERIHMUPTeckjxhFnirhnM8UIjFX11J7iLQUeX1V3tNO
bAd+sqr16incB8LSq+kU7fT/gzKp6fE/xXg58uqp+3k4/ADi0
qj7cR7w2xoHAbwOrLj6q6q09xToDePbg/9enaXhvTuS9sob
PlFWq6jljjrcFcF/gbOAprP6M2Qb4clU9ZpzxhuK+E9gb+H
Q761BgaVX9fU/xzgYeCnwWOLnPrk6TnAW8oHOen1xVf
9RjzH8E/qkT82+r6vU9xXsU8FqakdBW9cLVx/mX5G3Ai4
HvsfrcqHHH6nzPLgBual9vC/ywqnYeZ7yhuN+uqj3XNW+
M8R4M/CPwsKo6IMluwO9X1cd7itdrHjhSF3Az2PuBP6Ido
KSqLknypB7jnQQcB/xxO30I8K/APj3FuwaY38YA+FPgJ8

CjgP8LHDaOIMMnZ3tFuwtDSU+PXjP0egvgecCdPcbbdDj
Bqqo72kS5L1sMkp423i96rh18WVUdNxTvpiQvA3pJkpMs
pkmA9qP5kHo+8K0+YrWWA99MsgS4bTCzqt7XQ6xJvzc
n9V55T/v7T4CHAIPa+UNp/r7j9pfAq4GHAd8emn8LzWdp
Xw4Efreq7gJI8i/Ad4BekuSq2i/JQ4CDgeOTbAN8pqre3kO4
7QfJahv7piQP6iHOsAOq6nWdmM8EekmSaS42FtN8z/1PT
zEGDgYe2ffF9+B7tv3cXNJ2j0uSA4CnjTteu99nAjsk+eDQo
m3o97Psn4FPAP/QTn8X+AzQS5JMz3ngbE+SqaprkwzP6v
OESlWdODT9qXawlL48rqqG/9lfSnJOVT0pydhrKpIcAby
KZvjwi4F9gfOAPxx3LICquqgz65tJvt5HrNZPkzynqpYAJD
mIfseVvy3JnlX17TbeIuBXPcbbJEnafstJMo9maPi+PL6qdk
9yaVW9Jcl7gc/3GG9l+7MJsHWPcabjvTmR90pVfb3d/9um
+mzpId4HgA8keUVVfWjc+1+HbYEb29f37ztYVV0HfLCtV
f474I1AH0nyXUkWVNUPAZI8nLXcHRiTeUk2r6rb25hbA
pv3GO/OqvpIj/sfdjnNe+X6CcXbq6qOGkxU1Zfb2uxxW0lz
1/k5wPDn2a3AX/cQb2D7qjolyd/DqkHler3Q6TMPnO1J8rV
tVXu1NYKvBP6rx3hnt//4f6X5UPpT4LRBe6KqunFtG2+A+
VN8GM5vl/Vx1fsqYC/ggrZmZFfgLT3EAVbVWg9sAiyiqd
3qy1HAp5MMarCuBV7UY7xXA59NspLm/fIwmvdMX84A
TmlrKormeL/SY7xBEvfLJA8Dfgb0cssQoKp6ey92TcN781V
M9r0yP8kjBk3FkuzM6s+WPnw0ySuBQWL+NZomY7/pKd
47ge+0CWvauL3UIgMkeQzN/+v5NOfBycDf9hTudcA3hi5q
ngQc2VMs0mQfnwP+I8knaN6fLwH+pa+YNBdtRwOnArc
PZvbwHQur3yuXd2KNtenRkBuSvJ7mLk4Bf07znhmrqroE
uCTJSTS53oKqunrccaZwW5LtaC/ckuxL00y0L73mgbO9Tf
L2NG2En0bzQXgm8KqqGvsbro33/fblcJvdgbG33W1vl3yU
pq0UwCOAo2m+YF5WVf9nzPEurKq9klwM7FNVtye5uKp
+d5xxhuINt9H6Dc3t3rdW1bl9xBuKez+a9/6tPcfZAngFza2g
W2jasH9oXG3Jp4i3Cc2X5fD58LGq6uUqPskbgA8BT2X1rf
OPVdUbeoo3kTb6bazBexOaW5PL6fG9meZBmjNo2ir+Mc

1dnDcMapZ7iLc/zehUg+cpFgJHVtWZPcX7GHAfVidWhw
H/U1VH9BGvjflQmov+AP/Z1vT2FesCmsqTz1bVyr7itLFO
BC6juUhdDpxfVX3eESPJt2mS81WfLVV1Ro/xvj/F7F6ej2n
vyn6U5m9611CwXu4ctRfgb2L1BeM5wFt6ugAgzUOr7wE2
q6qdk/wuzWdZLxcBaR56/BDNsypX0Fx8P7+qLu0pXq954
KxOkietvcV0NPAEmi/QbwAf6THpGXxx7gwcBDweeF2PX
5ynAofT1IA+lebBgvtU1TN7incw8JWquqVNuPYE3tbj8W1
H8+E0+P+dS/Nh0ddF1Sk0yfHww0MPqKpenixO83TvrwdJ
cdvcYvOq+mVP8bYE/gp4IpM5Hz7APdvoXwdsCWxTVW
Npo9/GmvS5fmnbdOUJNA+9vJfmXO/reQeSbA7s2k5eNbi
V3lOsS6pqj3XNG3PM3WmS/+EHv3ppDtReeP9q6NzbhKa
d+djPvSRPpXlfPpGm4uRi4Jy2aUsv2rtv/1xVF/YVoxNvi+6
5NtW8McX6elU9edz7nSmSXETzff61qnpcO+/Sqtq9p3hbA
MfQVA7dSs+VQ32b1UlypzH6wM00TzF/sYd4UyU921bVw
eOO1cab+BfnUOwn07Tj+0pfDzRM+vjSPBV+DqsfVnoh8J
SqGvtDE228iSYGmXxvGqfQfAgOP/zV5/lwTqcd7ap5Sa6o
qt8eY6xJX+B8p6oel6ZXhsuq6qTBvJ7i3YfmAmcizR/amsgX
VNX32ulHAJ+r/p6wPwHYnaYma1A7WFX1kp7iTfrcm0dT
S74fTbOqX1XVrmvf6l7Fu5LmgfEfcPeHZvtKtCbWI0OS99
E0s1jC3Ztb9FVZM5+mzXq3V6C+es75z6raZ/jzpOckedKfn
fOBl3HPC+KxnOuzvU3yFjQ1IZ9tp59H86H40iT7VdWrxxz
v0Z0E5+wkl4w5xrDBbfIDgcVV9cUkb+4x3ip93WrqmPTxP
bCqhh+QeHuS5/YY7ztJ9q2qCwCS7AN8s8d4k+5NY9Lnwy
Tb6E/62H6U5KM0twzf3dby9tmP/Udomj8Mej45rJ3XV/OH
19L8DYebdxzeUyyAfatqtx733zWxcy9N/9Jb0dTQfYPmQb
C+Hzo7oOf9A5Cmh5AdgC1z976Kt6HpSacPgwvRQeVMa
O4e9ZK00iSPnwGeRXOB8xfAT3uKBXB5kj+jefhyF5o2u+f
1GG/Sn51fpDkP/p0eOm6Y7UnybwFPrao7AZJ8hKY9ytNp
2heN26STnkl/cU7apI/v7CSHAKe0088HTusx3j7Ai5L8sJ1eA
PxXkstoarXGfSU/6d40Jn0+/C1wbpK7tdFvm5mM+yGiSR/

bwcD+wHuq6udte9rX9hhvr84X2Vd7/iL7Jk27z0FPOR+lSf
L6cn6S3arqyh5jDJvkuXcp8HvAY2nunP48yflV1du5XlU/6G
vfHX9E02fxjqzurhCaO1ZjffAyyd+0L/+Nu48TAP32FrJdV
X08yavayqivp9+ec15B0x3b7TRN1c4A+uhNY2DSn533rar/1
dfOZ3tzi6uBvavq5nb6/jQPaOw6zluVg6SGpubl0cAP2+mH
A1dW1WPHEWeKuPel+eK8rKr+u/3i/J2+Hq6ZtEkdX5Jba
T4At2x/D6425wG/qKptxhlvKO7D17Z83F88Sfaiear+bj0k1D
27M7u3cabrfNiCJlFeRNMU6Czg/eNs6zZdxzZp09D8YdK3
YJ8EfImmzfrtsGpgpL5uMU/k3OvEvB9NbfxrgIdUVZ9dsk
1Ukj+n+TsuZHVlXtUYBypK8qb25aNpmq58keZ98myaNt6
93FVJckFV7ZtmcKQP0rxnPldVj+wj3lDcbWj+hn0/sP5frP7
shLZyiKbZ09jPwSRvB86rtt/pcZvtSfJLaToz/xqs6ubnH2mu
lt5cVWOpiZl0sqPxSxLgor6SgJkgE+pNY7rOh0kkWhvLuZ
7m4a9/5u69WxxeVWf3FG/S7fOXAX/DPXss6Ou9ObGebN
L0zf9EmtrkH9A8Z/GNqvrquGNNlzaBvIlmAJpVt9Cr6r09x
DoTeN4geUyyNU0vJfuPO1a7/2fRNA/YiaYXiG1o8pUv9RR
vL+AEVvctfzPwkr4u4KahcuhWmuZHt9P0kjW4IB5L5des
bm7R3rL4Mk17uqtomlqsqKrbGOOtyrnyxbgxq6pKcn6Sv
WpCT2hPg0/SfEG/o50+FDgRGGtt3TSeD723dduIzvXtaG7
XL2R1zzl99mU66VuwP6x20KAJmci519oSeB/NRX+fI6dNp
x2qx2G2OxZw92ca7qA5L/ryAuDcqroc2C9Nl3Dvobnz0YeP
A0dX1TcA0jwo/wmaB1vHbtKfoVW1dXocKXhWJ8mZeoS
48+mvwb1mt/2Ao5Isp3lCu9dbsNNg0g9MTNqkE6257A1V9
dn2FuzTaXqW+QirH14at0m3z78qzSAKX+LuPRb0NSLkx
M69qvrffex3hjkvye9UVR/PFnWdCHwrTReoRdNPeZ8Dpe
xedx9W/MYkvfRi07p1kCC38c5ta1/nhDXkgWMbKXhWJ8l
MeIQ4zXoTeUJ7Gs3JJLLTTniQaK1qJzydZZvFJt2zTC+3r
tdiS5rk+BlD84r+hk2fk+fepA2d65sCh7e9ofTapryq3tHekX5i
O+vwqvrOuOMM2STJA6rqJmAwuEifudi32gfkh0cK/tqg95
Dqqau7Ceo1D5ztbZInOkKcNJNN+oGJSdlY2glPUpJ/A35E

07PM79H0xPCtvtoIz3Vz9dybtI3hXE/yIpqeOj5Hk7QeDLyj
qk7sKd7anjOo6ql/5knpOw+c7TXJK5JsC3wBOCvJTTRPik
obo0nX1k3EXPhinIEm3eXcRCV5FE3zkQdX1WPTjL73nK
p6e08h5+S5N2kbw7leVZ9MspSmWWiAP+mzq8Kq2q+vfc8
QveaBs7omeVgmMEKcJGnma/udfS3NKIKDUcYunytd+E
mjSrId8CaaocwLOBd4a1X9bFoL1oM+8sDZXpO8SlemsVc
AAAFWSURBVE1mhDhJ0sx336r6VtPz4ypztScIaW1Opuk
m8Hnt9AtpRvx72rSVqCd95IFzJkmWJKl1Q5JH0o6cluT5
wI+nt0jStHhgVQ2PsPf2JM+dttLMMibJkqS55uXA8cCuSX
4EfJ+mBk3a2Jyd5BDglHb6+cBp01ieWWXOtEmWJAkgye
Y0ycBC4IE0A32MdVhjaTYYGpFu0O3jPJpxAmCMI9PN
VdYkS5Lmmi8CP6cZ1tgej7TRWtOIdD7HNRprkiVJc4o9
WUiNNY1IV1VjGZFurttkugsgSdKYnZfkd6a7ENIMMBiR
7gdtn8mPA26Y3iLNHja3kCTNCdMxrLE0w/26qn6dhCSb
V9VVSR493YWaLUySJUlzxbOmuwDSDOPIxPeCbZIlSZL
mOEcmXn8myZIkSVKHD+5JkiRJHSbJkiRJUodJsiRJktR
hkixJkiR1mCRLkiRJHf8fCMQ9OauT1YYAAAAASUVOR
K5CYII=\n",

```
   "text/plain": [
    "<Figure size 864x216 with 1 Axes>"
   ]
  },
  "metadata": {
   "needs_background": "light"
  },
  "output_type": "display_data"
```

```
   }
  ],
  "source": [
   "plt.figure(figsize=(12,3))\n",
   "features = X_test.columns.values.tolist()\n",
   "importance = clf_best.feature_importances_.tolist()\n",
   "feature_series =
pd.Series(data=importance,index=features)\n",
   "feature_series.plot.bar()\n",
   "plt.title('Feature Importance')"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 19,
  "metadata": {
   "_cell_guid": "6cb8a976-2759-4977-90a6-ff4620cdcad0",
   "_uuid":
"7adf7c33dee3385773492ecb9b3d837d1e018667"
  },
  "outputs": [
   {
```

```
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Index(['sg', 'al', 'su', 'bgr', 'sc', 'pot', 'pcv', 'wc', 'rc',
'dm'], dtype='object')\n"
    ]
   }
  ],
  "source": [
   "list_to_fill = X_test.columns[feature_series>0]\n",
   "print(list_to_fill)"
  ]
 },
 {
  "cell_type": "markdown",
  "metadata": {
   "_cell_guid": "81c680cf-3cfd-4a38-a0d5-04c57df2118d",
   "_uuid": "c139814126054 22f6120faba5721309fd52864d6"
  },
  "source": [
   "## Next, I examine the rest of the dataset (with missing
values across the rows)\n",
```

"Are there correlations between occurence of missing values in a row? The plot suggests, seems no."

]

},

{

"cell_type": "code",

"execution_count": 20,

"metadata": {

"_cell_guid": "f9075981-ffbc-4650-aa48-f88f2329af28",

"_uuid": "02c51e5a3b53c0812ec4402a29b4220c49193867"

},

"outputs": [

{

"data": {

"image/png": "iVBORw0KGgoAAAANSUhEUgAAAmsAAAIbCAYAAAC0btklAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nOzde7hdZXnv/e8vAeSUAMGAHBIDGuEjWjVooC1G91WEX0tbW1TtEjbvd2te9uDu27fle7qVIUp24qLfWsWGqRYK2owaIJFAIICg2RhCCnBBIOcpB7/7Fm2sVyhaxJxpxrjrm+n+saV8Yc4xxn3uCeZbiv5xnjeVJSJIkaTjNmu4EMEmStGUnuGWkS

UPMYk2SJGmIWaxJkiQNMYs1SZKkIWaxJkmSNMS2m
+4EeuQ8I5IkDadMdwKjyp41SZKkIWaxJkmSNMQs1iRJk
oaYxZokSdIQs1iTJEkaYhZrkiRJQ8xiTZIkaYg1Wqwl+XK
SK5Jcl2RJ99jbkvwwyaVJPp7kr7rH5yf5YpLl3e3FTeYiSZI
0ClLV3DyzSeZV1fokOwHLgeOAy4DnA5uAfwaurqrTknw
K+EhVLUuyEFhaVf/PVm7hpLiSJA0nJ8Xtk6aHQU9PcjV
wObAAeCvwzapaX1WPAJ8f1/ZY4K+SXAVcCMxNMmdi
wCRLkqxIsqLT6TScriRJ0nBrbLmpJC9jrAA7uqoeSHIp8
ANgS71ls7ptH3yiuFXVATZXafasSZKkGaXJnrXdgA3dQu
1g4EXAzsDPJ9kjyXbA68e1vwQ4bfOHJIc3mIskSdJIaLJY
uxjYLslK4H2MDYXeCvx/wHeBfwK+D9zbbX86cGSSlUm+
D5zaYC6SJEkjodEXDCa9QbJrVd3X7Vm7ADi3qi54kuEcB
pUkaTj5gkGfDGKetfd2XyK4FrgZ+PIA7ilJkjQS+t6z1rBW
JStJ0gxiz1qfuIKBJEnSELNYkyRJGmIWa5IkSUOssUlxB2
XdPfc1Emff3XdtJI4kSVI/2bMmSZI0xCzWJEmShpjFmiRJ
0hCzWJMkSRpiFmuSJElDrO/FWpJFSa7t930kSZJGkT1r
kiRJQ2xQxdp2Sc5LsjLJF5LsnGR1kjOSfK+7PXNAuUiSJL
XGoIq1g4BOVT0X2Ai8o3t8Y1W9APgr4C8nuzDJkiQrkqzo
dDqDyVaSJGlIpKr6e4NkEfCtqlrY/fwK4HTgcOAVVbUqyf
bAj6tqz62EK1cwkCRpKGW6ExhVg+pZm1gR1iTH+1s1Sp
IktdCgirWFSY7u7p8MLOvunzTuz38ZUC6SJEmtMahi7Xr
glCQrgXnAR7vHn5Lku8DvAL83oFwkSZJao+/PrG3xxslq4
MiququHy3xmTZKk4eQza33iPGuSJElDbLvpunFVLZque0
uSJLWFPWuSJElDbNqeWXuSWpWsJEkziM+s9Yk9a5IkS
UNs2p5Ze7I2fmVpI3HmvuY4Hr5lbSOxdli4fyNxJEmSJrJn
TZIkaYhZrEmSJA0xizVJkkaYhZrEmSZI0xCzWJEmShpjF
miRJ0hAbyNQdSXYBPgfsD8wG3gcv3cBdwJXBgVb1
mEPlIkiS1xaDmWTseWFdVvwiQZDfgWuClVXVzkk8PK
A9JkqRWGdQw6DXAsUnOSPIS4ABgVVXd3D2/xWItyZI
kK5Ks6HQ6g8hVkiRpaAykZ62qfpjkCODVwJ8BX+vh2g6

wuUqrplYwkCRJaoNBPbO2L7C+qs5Pch/wduDAJIuqajVw
0iDykCRJaptBPbP2HOADSR4DHmGsWNsHuDjJXcD3Bp
SHJElSqwxqGHQp8LjxyyS7VtXBSQKcDawYRC6SJEltM
p3zrP1mkquA64DdgI9NYy6SJElDaVDDoD+jqs4Ezpyu+0u
SJLWBKxhIkiQNMYs1SZKkIZaqmu4cetGqZCVJmkEy3
QmMqml7Zu3JuvcfLm4kzm6/dDwP37K2kVg7LNwfgBuP
Oa6ReIuXOfGvJEka4zCoJEnSELNYkyRJGmIWa5IkSUP
MYk2SJGmIWaxJkiQNsaEo1pKsTvLU6c5DkiRp2AxFsSZ
JkqTJDbxYS/LlJFckuS7JkkHfX5IkqU2mo2ft16vqCOBI4P
Qkez5R4yRLkqxIsqLT6QwmQ0mSpCExHSsYnJ7kdd39Bc
DiJ2pcVR1gc5VWTa1gIEmS1AYDLdaSvAw4Fji6qh5Icim
w4yBzkCRJapNBD4PuBmzoFmoHAy8a8P0lSZJaZdDF2sX
AdklWAu8DLh/w/SVJklploMOgVfUQcMIkpxYNMg9JkqS
2cJ41SZKkIWaxJkmSNMQs1iRJkoZYqmq6c+hFq5KVJG
kGyXQnMKrsWZMkSRpi07GCwTbZ+NWvNRJn7gmv5J
Fbb2sk1vb77QPAjccc10i8xcuWNhpLkiS1lz1rkiRJQ8xiTZI
kaYhZrEmSJA0xizVJkqQhZrEmSZI0xCzWJEmShthAirU
kuyT5xyRXJ7k2yUlJVid5avf8kUkuHUQukiRJbTKonrXjg
XVVdVhVHQpcPKD7SpIktdqgirVrgGOTnJHkJVV171Qv
TLIkyYokKzqdTh9TlCRJGj4DWcGgqn6Y5Ajg1cCfJbkEe
JT/KBZ3fIJrO8DmKq2aWsFAkiSpDQb1zNq+wANVdT7w
QeD5wGrgiG6T1w8iD0mSpLYZ1NqgzwE+kOQx4BHg7cB
OwP9J8j+A7w4oD0mSpFYZ1DDoUmCyFcWfNYj7S5IktZ
XzrEmSJA0xizVJkqQhZrEmSZI0xCzWJEmShliqarpz6E
WrkpUkaQbJdCcwquxZkyRJGmKDmmetMQ9efW0jcXY6
7FAevfOuRmJtN/+pANx4zHGNxFu8bGmjsaDZ3CRJ0uD
YsyZJkjTELNYkSZKGmMWaJEnSELNYkyRJGmIDKda
S3DeI+0iSJI2avhdrSTKI+0iSJI2ivhRRSRYluT7JR4ArgZ2
S/EWSK5N8Pcn8brtnJvmnJFd3zz2jH/lIkiS1VT97vA4C/qa
qntf9fGVVPR/4JvAn3WN/B5xdVYcBPwfc1sd8JEmSWqefx
qntf9fGVVPR/4JvAn3WN/B5xdVYcBPwfc1sd8JEmSWqefx

dqPqury7v5jwGe7++cDxySZA+xXVRcAVNVPquqBiUGSL
EmyIsmKTqfTx3QlSZKGTz9XMLj/Cc4VU1xDrKo6wOYq
rZpawUCSJKkNBvXg/yzgl7v7bwaWVdVGYG2S1wIkeUqS
nQeUjyRJUisMqli7HzgkyRXAK4A/7R5/K3B6kpXAd4CnD
SgfSZKkVujLMGhVrQYOHfd51+7uH01odyNjxZskSZIm4
fxnkiRJQ8xiTZIkaYhZrEmSJA0xizVJkqQhlqqa7hx60apkJ
UmaQaY0f6p6189Jcfvins//fSNxdn/Didzz2S81E+uk/wzAjccc
10i8xcuWNhoLhjs3SZK0ZQ6DSpIkTZDk+CQ/SHJTkndPc
v7EJCuTXNVdFvOYqV7bK4s1SZKkcZLMBs4GTgCeDZy
c5NkTmn0dOKyqDgd+HfhED9f2xGJNkiTp8V4A3FRVq6r
qYeAzwInjG1TVffUfD/7vwn88V7/Va3vVumfWJEmSxrvx
mON6egHxWZdd8lvAknGHOlXVGfd5P2DNuM9rgRdOjJ
PkdcCfAXsBv9jLtb2wWJMkSTNKtzDrPEGTyd5s/ZmCsK
ouAC5I8lLgfcCxU722Fw6DSpIkPd5aYMG4z/sD67bUuKq+
BTwjyVN7vXYqBlKsJVmU5IYk53XfnPhCkp2THJXkO0m
uTvK9JHMGkY8kSRohmdXbtnXLgcVJDkiyA/Am4MLH3
TJ5ZpJ0958P7ADcPZVrezXIYdCDgLdV1WVJzgVOA04F
Tqqq5UnmAg8OMB9JkqSfUVWPJjkNWArMBs6tquuSnN
o9fw7weuBXkjzCWP1yUveFg0mv3ZZ8Blmsramqy7r75wP
vAW6rquUAVbVxsouSLKH7EODHPvYx3rjH3oPIVZIktU
WaXzyhqi4CLppw7Jxx+2cAZ0z12m0xyGJt4sN1G4GnbPW
ixz8EWE2tYCBJkkZDZo32SleDfMFgYZKju/snA5cD+yY5
CiDJnCS+nSpJknrT/DNrQ2WQGV8PnJJkJTAPOAs4CTg
rydXA14AdB5iPJEkaBUlvW8sMsifrsao6dcKx5cCLBpiDJE
lSqzjsKEmS2m3En1kbSLFWVauBQwdxL0mSNLOkhUOb
vWjfU3aSJEkziMOgkiSp3WaNdt9TxibbbY1WJStJ0gwyb
WOR/3bcf+6pPnjG0i+1atzUnjVJktRuI/7MWuuKtbvv/0kjcf
bcZUfW/+1nG4k1760nAbD67nsaibdoz90bjQWw6s4NjcQ7c
P4erLvnvkZi7bv7rgA8cuttjcTbfr99GokjSWqZES/WRnuQ
V5IkqeVa17MmSZI0Xkb8BQOLNUmS1G4jXqyN9reTJEl

qOXvWJElSu434CwYWa5IkqdVcbqoHSRYluSHJeUlWJvl
Ckp2THJXkO0muTvK9JHOSzE7ywSTXdNu+s8lcJEmSRk
E/etYOAt5WVZclORc4DTgVOKmqlieZCzwILAEOAJ5XV
Y8mmdeHXCRJ0qibZc9ar9ZU1WXd/fOB44Dbqmo5QFVt
rKpHgWOBc7r7VNX6yYIlWZJkRZIVnU6nD+lKkiQNr37
0rE1cn2sj8JRJ2mWStj8brKoDbK7SqqkVDCRJ0ojIaE9u0
Y9vtzDJ0d39k4HLgX2THAXQfV5tO+AS4NTuPg6DSpKkJ
2VWettaph/F2vXAKUlWAvOAs4CTgLOSXA18DdgR+AR
wC7Cye/zNfchFkiSNuCQ9bW3Tj2HQx6rq1AnHlgMvqTt
f+lukiRJT04fhkGTHA98CJgNfKKq/nzC+bcAf/nzc2ZUF
6untuNbAJ+CnwaFUduS25OM+aJEnSOElmA2cDrwTWA
suTXFhV3x/X7Gbg56tQ5ITGHu+/oXjzr+8qu5qIp9Gi7Wq
Wg0c2mRMSZKkJ9T8c2gvAG6qqlUAST4DnAj8e7FWVd8
Z1/5yYP+mk9hstF+fkCRJIy+zZvW2jZsmRByP2DN
uM9ru8e25G3AV8d9LuCSJFdMErtnDoNKkqQZZcK0YJ
OZrKtu0unGkrycsWLmHX1xV65LsBXwtyQ1V9a0nm
2/rirU9d9mxsVjz3npSY7EAu25+1DGAjhw/h6Nxdp3910bi
wWw/X77NBpPkjTDNP+G51pwFFEKe+6yI+6yI+v/9rONxNpc
V3bz5eVeu6f96R5ALGhlVnTrHW1KS4e+6yI+v/9rONxNpc
9K2++55G4i3ac/dGYwGsunNDI/EOnL8H6+65r5FYm4u+R
269rZF4m4u+G485rpF4i5ctbSSOJKl1lgOLSXTj
CVZCHwJeGtV/XDkrbiR
Jj9Nwz1p3zfLTgKWMTd1xblVdl+TU7b27b
fMUHXsDF3SPbQd8qqou3pZ8Wr+4zWr+fcmqugi4i4a
MKxc8bt/wbwG5Nctwo4rMlcfBtUkiRpiNmzWq2NS0j
1wmJNkiS1WwsoZCS1WvsXKZ+9F34dBkyxmTJEn5GiT1
t1oeF3IfJoL7ddknOS7IKnMmS7IyzmTJEn5GfcBZkiT1
R9Lb1jjKKDtYOAjpV9VxgIe5IcB
OX7+r03milSEkSZJGz6CGQdU1WXd/fOB04Gbk/w+sD
MwD7gO+IeJ05YvKnkmYLn5aw

EeAI6tqTZL3As0t+ilJkmaOFg5t9mJQw6ALkxzd3T8ZWN
bdvyvJrsAvDygPSZKkVhlUz9r1wClJPgbcCHwU2AO4Blj
N2IKpkiRJvevDclPDpO/FWlWtBp49yak/7G6SJElPWizWJ
EmShpjPrEmSJGm62LMmSZLabcR71lI1cVaNodaqZCVJ
mkGmrWJae/of9FQf7P/hM1pV3TkMKkmSNMRaNwy6ad
OmRuLMmTOHDZ/6fCOx9njzGwC4dUMzue23xxzW3XNf
I7H23X1XAB69/c5G4m2393zuv3xFI7F2edGRADzy49sbibf9
0/YGYPXd9zQSb9GeuzcaS5LUH6O+vHjrijVJkqTHGfFiz
WFQSZKkIWbPmiRJarcRX8jdnjVJkqQh1nixlmRRkmubj
itJkjSpzOpta5mhyjiJw7KSJKknmZWetinFTI5P8oMkNyV
59yTn35JkZXf7TpLDpnptr/pVHG2X5DzgecAPgV8BXgb8
b+Au4ErgwKp6TZL3AvsCi7rn3tynnCRJkrYqyWzgbOCV
wFpgeZILq+r745rdDPx8VW1IcgLQAV44xWt70q+etYOA
TlU9F9gI/BfgY8AJVXUMMH9C+yOAE6vKQk2SJPVm1q
zetq17AXBTVa2qqoeBzwAnjm9QVd+pqg3dj5cD+0/12p6/3
rZc/ATWVNVl3f3zgSOBVVV1c/fYpye0v7CqHpwsUJIlSV
YkWdHpdPqUriRJaq2kp218bdHdlkyIuB+wZtzntd1jW/I24
KtP8tqt6tcw6MQ1unbbSvv7txioqsNY1yJANbWCgSRJmpk
m1BaTmezBtknXH03ycsaKtWN6vXaq+tWztjDJ0d39k4F/A
g5Msqh77KQ+3VeSJM0wGestm/I2BWuBBeM+7w+sm+S+
zwU+wdijXHf3cm0v+lWsXQ+ckmQlMA84E3gHcHGSZcDt
wL19urckSZpJmn9mbTmwOMkBSXYA3gRcOL5BkoXAl4
C3VtUPe7m2V40Pg1bVauDZE48n+UZVHZyxkvZsYEW3/
XubzkGSJOnJqqpHk5wGLAVmA+dW1XVJTu2ePwf4Y2B
P4CPd3rpHq+rILV27LfkMcl6z30xyCrAD8K+MvR0qSZK0
bfqwkHtVXQRcNOHYOeP2fwP4jaleuy0GVqxV1ZmMDY
dKkiQ1pw/F2jBxxQBJktRqmdpzaK012t9OkiSp5VK1TVN/
DFqrkpUkaQaZtrHIH7/3z3uqD5723ne3aty0dcOgt6xvZsaP
hfN2Y+NXv9ZIrLknvBKAy374o0bivfhZT+em29c3EuuZe8
8D4Ac/vquReAc97ams2bCxkVgL9pgLwMM/WrOVllOzw9

PHprW5feMW51juyd5zd2k0FsCNxxzXSLzFy5Y2EkeSRsI
UF2dvK4dBJUmShljretYkSZIex7dBJUmShpdvg0qSJGna2
LMmSZLaLaPd99T3b5dkUZJr+30fSZKkUWTPmiRJajen7
mjEdknOS7IyyReS7JxkdZKnAiQ5MsmlA8pFkiSNkCQ9b
W0zqGLtIKBTVc8FNgLvmOqFSZYkWZFkRafT6VuCkiR
Jw2hQw6Brquqy7v75wOlTvbCqOsDmKq2aWsFAkiSNiBF
/wWBQxdrENbsKeJT/6NnbcUB5SJKkUeMza41YmOTo7v
7JwDJgNXBE99jrB5SHJElSqwyqWLseOCXJSmAe8FHgf
wIfSvJt4KcDykOSJI2apLetZfo+DFpVq4FnT3Lq28Cz+n1/S
ZI02uIwqCRJkqaLk+JKkqR2821QSZKkIdbC59B6kaqJs2o
MtVYlK0nSDDJtFdOdZ32sp/pg/jt/q1XV3Wj3G0qSJD0JS
Y5P8oMkNyV59yTnD07yL0keSvKuCedWJ7kmyVVJVmxr
Lq0bBl2zYWMjcRbsMZd7v/yVRmLt9trXAHD5Tbc0Eu9F
z1zYaCyAy374o0bivfhZT+fRO+9qJNZ2858KwLp77msk3r6
77wrA7RvvbyTe3nN3aTQWwI3HHNdIvMXLljYaS5LaLL
Oa7XtKMhs4G3glsBZYnuTCqvr+uGbrGVuR6bVbCPPyq
mrkf5j2rEmSpHbLrN62rXsBcFNVraqqh4HPACeOb1BVd
1TVcuCR5r/Q41msSZKkGSXJkiQrxm1LJjTZD1gz7vPa7r
GpKuCSJFdMErtnrRsGlSRJepweJ8Wtqg7QeYImkwXs5S
WGF1fVuiR7AV9LckNVfaunJMexZ02SJLVakp62KVgLL
Bj3eX9g3VTzqap13T/vAC5gbFj1SbNYkyRJerzlwOIkByTZ
AXgTcOFULkyyS5I5m/eBVwHXbksyDoNKkqR2a3hS3Kp
6NMlpwFJgNnBuVV2X5NTu+XOSPA1YAcwFHkvyu4ythf
5U4IJuD952wKeq6uJtyWdgxVq3uvwcY12Js4H3AauADwG
7AA8Bv1BVmwaVkyRJGgENT90BUFUXARdNOHbOuP0f
M1bTTLQROKzJXAbZs3Y8sK6qfhEgyW7AvwInVdXyJH
OBBweYjyRJ0tAb5DNr1wDHJjkjyUuAhcBt3TlKqKqNVf
XoxIvGv17b6TzRixuSJGlGSnrbWmZgPWtV9cMkRwCvB
v4MuIQpvAY74fXaamoFA0mSpDYY5DNr+wLrq+r8JPcB
S4B9kxzVHQadAzw4We+aJEnSlkxxOo7WGuQza88BPpD

kMcaWZng7Y5POnZVkJ8aeVzsWaGahSEmSNDP04QWD
YTLIYdCljL0CO9GLBpWDJElS2zjPmiRJajeHQSVJkobY
iA+Djva3kyRJajl71iRJUqtl1mgPg6Zqq1OdDZNWJStJ0gw
ybRXThk99vqf6YI83v6FV1Z09a5Ikqd0y2k91ta5Y+8kNP2
wkzo4HP4ufXHdDM7EOORiATZuaWYN+zpw53LHpgUZi
7TVnZ6DZ3G5+3VsaiXXABX8HwKO339lIvO32ng/A99c1E
+/Z+85vNBbAjccc10i8xcuWsvqNv9ZIrEWf+2ug2dwkSc1p
XbEmSZI03qg/s2axJkmS2m3E51kb7UFeSZKklrNnTZIktd
uIv2AwFN8uycuSfGW685AkSS00K71tLTMUxZokSZIm1+
gwaJJdgM8B+wOzgfcBdwEf7N5rOfD2qnooyfHAX3bPX9l
kHpIkaeaILxj05HhgXVUdVlWHAhcDnwROqqrnMFawvT
3JjsDHgV8CXgI8reE8JEnSTOEwaE+uAY5NckaSlwCLgJu
ravNMtucBLwUO7h6/scbWuzp/SwGTLEmyIsmKTqfTcLq
SJEnDrdFh0Kr6YZIjgFcDfwZc8kTNpxizA2yu0qqpFQwkS
dKImDXaj+A3+u2S7As8UFXnM/ac2s8Bi5I8s9vkrcA3gRuA
A5I8o3v85CbzkCRJ2hZJjk/ygyQ3JXn3JOcPTvIvSR5K8q5
eru1V0/OsPQf4QJLHgEeAtwO7AZ9PsvkFg3O6LxgsAf4xy
V3AMuDQhnORJEkzQcPzrCWZDZwNvBJYCyxPcmFVf
X9cs/XA6cBrn8S1PWl6GHQpMNkqzs+bpO3FjD27JkmS9
KT14W3QFwA3VdWqbvzPACcC/15wVdUdwB1JfrHXa3s1
2oO8kiRJE4x/ebG7LZnQZD9gzbjPa7vHpmJbrp2Uy01Jkq
R263E6jgkvL05msoBTejFyG6+dlMWaJElqt+aHQdcCC8Z
93h9YN4BrJ+UwqCRJ0uMtBxYnOSDJDsCbgAsHcO2kMj
YnbWu0KllJkmaQaVsaYOPFX++pPph7/C9sNdckr2ZsWcz
ZwLlV9f4kpwJU1TlJngasAOYCjwH3Ac+uqo2TXdvTF5qY
S9uKtYd+cFMjgZ5y0DP5yXU3NBJrx0PGXmrdtGlTI/Hmz
JnTaCyAR269rZF42++3D6v+UzPT4h144acBePiWtY3E22H
h/gBcd+sdjcQ7ZL+9Go0FcOMxxzUSb/Gypfzozb/RSKynf+o
TQLO5NRlLUmtMW7G26ZJ/7qmYmfOqV7RqzSmHQSV
JkoaYLxhIkqR2mz17ujPoK3vWJEmShpg9a5IkqdX6sILB

UBmKnrUkv5tk5+nOQ5IkadgMRbEG/C5gsSZJkno3a1Zv
W8v0JeMki5LckOS8JCuTfCHJzkl+Icm/JrkmyblJnpLkdG
Bf4BtJvtGPfCRJ0ghLettapp/l5UFAp6qeC2wE/gvwSeCkqn
oOY8/Lvb2qPszYMgwvr6qX9zEfSZKk1ulnsbamqi7r7p8P/
AJwc1X9sHvsPOClWwuSZEmSFUlWdDpPtOaqJEmakWa
lt61l+vk2aCNLI1RVB9hcpTW2goEkSRoNSfueQ+tFP7/dwi
RHd/dPBv4JWJTkmd1jbwW+2d3fBMzpYy6SJEmt1M9i7
XrglCQrgXnAmcCvAZ9Pcg1ji56e023bAb7qCwaSJKlnI/6C
QT+HQR+rqlMnHPs68LyJDavqLOCsPuYiSZJGVQufQ+v
FaA/ySpIktVxfetaqajVwaD9iS5IkPY4vGEiSJGm6uJC7JEl
qtYz4M2upamQ6tEFpVbKSJM0g01YxPbD8yp7qg52Pen6r
qjuHQSVJkoZY64ZBm1rB4CkHPZNHfnx7I7G2f9reAKz/5
KcaiTfvV9/MPZ+7oJFYu7/xdQBs+LvPNRJvj7e8kTvP+lgjs
ea/87cAWH/epxuJN++UkwF46N9ubiTeU55xQKOxAB68+t
pG4u102KFs+PQXG4m1x8mvB+Du//O3jcTb821v5fY/P7O
RWHu/+/cAuOOMv2wk3l5/8LuNxJE0ZFo4d1ovWlesSZIkj
ZdZoz1QONrfTpIkqeXsWZMkSe02e/Z0Z9BX9qxJkiRNkO
T4JD9IclOSd09yPkk+3D2/Msnzx51bneSaJFclWbGtuUypZ
y3JIuArVeWqBJIkabg0PM9aktnA2cArgbXA8iQXVtX3xz
U7AVjc3V4IfLT752Yvr6q7msjHnjVJkqTHewFwU1Wtqqq
Hgc8AJ05ocyLwNzXmcmD3JPv0I5leirXZST6e5LoklyTZK
ckzklyc5Iok305yMECSTyb5aJJvJFmV5OeTnJvk+iSf3Bww
ycndbsJrk5zR9JeTJEmjL5nV0zYF+wFrxn1e2z021TYFXN
Ktj5Y8ya/173op1hYDZ1fVIcA9wOuBDvDOqjoCeBfwkXHt
9wBeAfwe8A/AmcAhwHOSHJ5kX+CMbpvDgaOSvHYbv4
8kSZppkp62JEuSrBi3TSyoJhtXnbhKwhO1eXFVPZ+xodLf
TvLSbfl6vRRrN1fVVd39K4BFwM8Bn09yFfAxYHz33z/U2
FpW1wC3V9U1VfUYcF332qOAS6vqzqp6FPg74Ge+zPj/oJ
1Op7dvJ0mSNEFVdarqyHHbxAJjLbBg3Of9gXVTbVNVm
/+8A7iAsWHVJ62XqTseGrf/U2Bv4J6qOnwr7R+bcO1j3fs+

OpWbdv8Dbv6PWE2tYCBJkkZE8wu5LwcWJzDPgzoAAC
AASURBVDkAuBV4E/DmCW0uBE5L8hnGXiy4t6puS7IL
MKuqNnX3XwX86bYksy0vGGwEbk7yBvj3V1gP6+H67wI/
n+Sp3bcuTga+uQ35SJKkmajHYdCt6Y74nQYsBa4HPldV1
yU5Ncmp3WYXAauAm4CPA+/oHt8bWJbkauB7wD9W1c
Xb8vW2dVLctwAfTfKHwPaMvS1x9VQu7Faf/x34BmPjvh
dV1d9vYz6SJEnbrKouYqwgG3/snHH7Bfz2JNetqq
KRVrVbUaOHTc5w+OO338JO1/9QmuHX/uU0Azq59Lkq
QZaYpveLaWy01JkqR2a/6ZtaEy2qWoJElSy9mzJkmS2m3
WaPc9jfa3kyRJarmMvczQGq1KVpKkGWTaHhx7eNXqnu
qDHQ5c1KqH3Fo3DPrA8isbibPzUc/n4VWrG4m1w4GLAF
j72+9qJN7+Z3+QNaf+XiOx/6Z/6BDe97
BcbifXMS/8RgFv/6x82Em+/v/hfAPzgx3c1Eu+gpz210VgAt7
3nfY3E2+f9f8TdnU82EmvPJb8KwJ1nnt1IvPm/99uN/lsAW
Hv6HzQT78NjSxDfeMxxjcRbvGxpI3EkbSOHQSVJkjRdW
tezJkmS9DhTWJEJJElSu414seZ02qwZbLkoSaZ4NKmix
LVaRnwFA4s1SZLUbq4NKkmS1SZLUbq4NKkmSNMR8Zm3qkixKckOS85K
sTPKFJDsnOSrJd5JcneR7Seh+W6Q8Zde2mSI5rMR5Ik
zQCz0tvWMv3oNzwI6FTVc4GNwHnAZ4HAB3fqarDgGOBB4
HPAG8ESLIPsGSqWqWJIlSVYkWdlEvkJ0vDq
xzDomqq6rLt/PvAWg5Wg5QVRsBknvBrwB+PbvBnwBf+f
LFhVdYDNVVo1tYKBJEkaDRnxG1R8WJxZkDQqWmK4rjMlsVVNM
twN1JngucxFhPmyRJGmmTgvcxFhPmyRJGkYg/Xx4r9MJpM0EYNUyqtY
ivYQmv9Arw9ZtxD2T8fIIBgnArwerGSqmNRRZDLFyFHR
9Xm1zj95ngN8Hdquqa/qXOe55N3cnvSFsyZLJGGkVGG6M2f9t4U
bgLOCfyjvy8SEWLsd9k2E+ltxZFmGZBv7QMxCPmxc5zbXlv
T21n9OnPPqlH8XaY1V16oRjy4FFVUy4EXqRSR
MNpP5EmSJLVco71aVbUOTstZPLP7Enmk/nAOt
Dk+yQ+S3J3ZOcT5IPSDnYH
KSJM0g0/aa5aZNm3qD+bMmfOEuSaZDfwQeCWwFn7
0+uqu+Pa/Nq4J3Aq4EXAh+qqm2b8n0iHo2dZJCmShNMhOLb0eC8Ab

qqqVVX1MGNTjZ04oc2JwN/UmMuB3bsrMk3l2p607k3Mj
V9Z2kicua85jodvWdtIrB0W7g/Avx3/+kbiPePiL3LziW9uJ
NYBf/8pAG58yQmNxFv87a9y4zHHNRNr2djf5ZrfPL2ReA
s+/mEAVq75cSPxnrvgaY3GArjncxc0Em/3N76OB6+5rpFY
Oz1nbInejRdd0ki8ua9+Fbf82jsaibXwrz8CwC1ve2cz8f7PW
QCN/oab/vcgaSjsB6wZ93ktY71nW2uz3xSv7Yk9a5IkaUYZ
v+54d1sysckkl00cat1Sm6lc25PW9axJkiRtiwnrjk9mLbBg3
Of9gXVTbLPDFK7tiT1rkiRJj7ccWJzkgCQ7AG8CLpzQ5k
LgV7pvhb4IuLeqbpvitT2xZ02SJGmcqno0yWnAUmA2cG5
VXZfk1O75c4CLGHsT9CbgAeDXnujabcnHYk2SJGmCqr
qIsYJs/LFzxu0X8NtTvXZbOAwqSZI0xPparCX5/SSnd/fPT
PLP3f1fSHJ+d4bfK5NcneTr/cxFkiSpjfo9DPot4L8CHwaOB
J6SZHvgGOAa4OPAS6vq5iTz+pyLJEkaQY/M3n66U+irfg
+DXgEckWQO8BDwL4wVbS8BfgJ8q6puBqiq9ZMFGD8X
SqfzRG/ZSpKkmaiqt61t+tqzVlWPJFnN2BsS3wFWAi8Hng
GsBo6aQozxc6FUUysYSJIktcEgXjD4FvCu7p/fBk4FrmKse
Pv5JAcAOAwqSZKejMeqetraZhDF2reBfYB/qarbGRv+/H
ZV3QksAb6U5GrgswPIRZIkqVX6Ps9aVX0d2H7c52eN2/8
q8NV+5yBJkkZXtbC3rBdOiitJklpt1Is1J8WVJEkaYvasSZ
KkVmvjSwO9sFiTJEmtNuK1GmnZOG+rkpUkaQbJdN14
3T339VQf7Lv7rtOW65PRup61e7/8lUbi7Pba1/DA8isbibXz
Uc8HYM2S32kk3oLOh1h72n9rJNb+f/UBAH70lt9sJN7T/+7
j3HjMcY3EWrxsbILjO874y0bi7fUHvwvA8lVrG4l31IH7Nx
oLYO3pf9BIvP0/fAYbv/q1RmLNPeGVANzd+WQj8fZc8qu
N/0aGOd4w5ybNFC3reOpZ64o1SZKk8R4b8YE33waVJEk
aYvasSZKkVhv1YVB71iRJkoaYPWuSJGDd1NkiRpiD3
22GgXa9M2DJoxDsNKkiQ9gYH2rCVZBHwV+AZwNPDlJ
L8MPAZ8tarePch8JElS+434KOi0DIMeBPwa8A/AHwEvrK
oHksybrHGSJcASgI997GOctNe+A0tUkiQNP98Gbd6Pqupy
4Fjgr6vqAYCqWj9Z46rqVNWRVXXkkiVLBpmnJElqgceo

**nrZtkWRekq8lubH75x5baHd8kh8kuSnJu8cdf2+SW5Nc1d1
evbV7Tkexdn/3z+Ban5IkaRtVVU/bNno38PWqWgx8vfv5cZ
LMBs4GTgCeDZyc5NnjmpxZVYd3t4u2dsPpfMD/EuDXk+
wMY5XqNOYiSZI0FScC53X3zwNeO0mbFwA3VdWqqno
Y+Ez3uidl2oq1qroYuBBYkeQq4F3TlYskSWqvAfes7V1Vt3
Xvexuw1yRt9gPWjPu8tntss9OSrExy7paGUccb6AsGVbUa
OHTc5z8H/nyQOUiSpNHS6zRr419e7OpUVWfc+X8CnjbJ
pe+Z6i0mObY5y48C7+t+fh/wF8CvP1EwJ8WVJEkzSrcw6z
zB+WO3dC7J7Un2qarbkuwD3DFJs7XAgnGf9wfWdWPfPi
7Wx4GvbC1fJ6WVJEmtNuBh0AuBU7r7pwB/P0mb5cDiJA
ck2QF4U/c6ugXeZq8Drt3aDe1ZkyRJmro/Bz6X5G3ALcAb
AJLsC3yiql5dVY8mOQ1YCswGzq2q67rX//9JDmdsGHQ18
Ftbu2FaNpFcq5KVJGkGmew5rYG4du3tPdUHh+6/97Tl+m
TYsyZJklrtsXZ1PPWsdcXapk2bGokzZ84c1j/wk0Zizdt5Rw
DW3XNfI/H23X3XRmNBs7ldd+tkz1L27pD9xt52XrNhYyP
xFuwxF2j2N9JkLGj27+Hu+5v5/e65y9jvt8l4q+7c0EisA+ePv
dHedLwm/x5uWX9vI7EWztsNgJtun3Qxl549c+953HDbnY3
EOnif+Y3EkfTktK5YkyRJGs+eNUmSpCHWsufve+bUHZI
kSUPMnjVJktRqoz4M2njPWpJFSX5mgrckv9qdg0SSJKkx
Vb1tbTPIYdBfBSzWJEmSetCvYdDZ3fWufg64Ffhb4Ejg75
I8CBwNXA+cB/wSsD3whqq6oU/5SJKkEeULBk/OYuDsqjo
EuIexlQdWAG+pqsOr6sFuu7uq6vmMrUD/rj7lIkmS1Fr9K
tZurqqruvtXAIu20O5LW2uTZEmSFUlWdDqdRpOUJEnt
91hVT1vb9GsY9KFx+z8FdtpKu59uKZeq6gCbq7RqakZ5S
ZI0GkZ9GHSQU3dsAuYM8H6SJGkGGPFabaDF2ieBc8a9
YCBJkrTN2ji02YvGi7WqWg0cOu7zB8ed/uK4/UXj2qwAX
tZ0LpIkSW3nCgaSJKnVXvysp2e6c+gn1waVJEkaYhZrkiR
JQ8xiTZIkaYilZXOTtCpZSZJmkJF+bmw62bMmSZI0xFr
3NmhTKxjMmTOH9Q/8pJFY83beEYB199zXSLx9d9+10Vj
QbG7X3XpHI7EO2W8vANZs2NhIvAV7zAWa/Y00GQua/

Xu4+/5mfr977jL2+20y3qo7NzQS68D5ewA0Hq/Jv4db1t/bS
KyF83YD4Kbb1zcS75l7z+OG2+5sJNbB+8wH4MZjjmsk3u
JlSxuJI80U9qxJkiQNMYs1SZKkIWaxJkmSNMQs1iRJkoZ
Y34u1JO9N8q5+30eSJGkU2bMmSZI0xPpSrCV5T5IfJPkn
4KDusUuTnJnkW0muT3JUki8luTHJ/+pHHpIkSW3X+Dxr
SY4A3gQ8rxv/SuCK7umHq+qlSX4H+HvgCGA98G9Jzqyq
u5vOR5Ikqc360bP2EuCCqnqgqjYCF447t3n/GuC6qrqtqh4
CVgELJguWZEmSFUlWdDqdPqQrSZI0vPq1gsGW1vB8q
PvnY+P2N3+eNJeq6gCbq7RqakZ5SZKkNuhHz9q3gNcl2S
nJHOCX+nAPSZKkGaHxnrWqujLJZ4GrgB8B3276HpIkS
TNFX4ZBq+r9wPsnHP7guPOXApeO+/yyfuQhSZLUds6zJ
kmSNMQs1iRJkoaYxZokSdIQs1iTJEkaYqna0pRoQ6lVyU
qSNINkuhMYVf2aFLdvmpoUd86cOax/4CeNxJq3844ArLv
nvkbi7bv7ro3GgmZzu+7WOxqJdch+ewGwZsPGRuIt2GM
u0OxvpMlY0Ozfw933N/P73XOXsd9vk/FW3bmhkVgHzt8D
oPF4Tf493LL+3kZiLZy3GwA33b6+kXjP3HseN9x2ZyOxDt
5nPgA3HnNcI/EWL1vaaCxp1DkMKkmSNMQs1iRJkoaYx
ZokSdIQs1iTJEkaYkNRrCVZneSp052HJEnSsBmKYk2SJ
EmTa3TqjiS/AryLsfnQVgKfA/4Q2AG4G3hLVd2eZE/g08B
84Hs4N4skSdKkGutZS3II8B7gFVV1GPA7wDLgRVX1PO
AzwO93m/8JsKx7/EJgYVN5SJIkjZImh0FfAXyhqu4CqKr1
wP7A0iTXAP8NOKTb9qXA+d12/whsccbLJEuSrEiyotPpN
JiuJEnS8GtyGDT87HJQZwH/u6ouTPIy4L3jzk1p6aiq6gCb
q7RqakZ5SZKkNmiyZ+3rwBu7z6ORZB6wG3Br9/wp49p+
C3hLt90JwB4N5iFJkjQyGutZq6rrkrwf+GaSnwL/ylhP2ue
T3ApcDhzQbf4/gU8nuRL4JnBLU3lIkiSNkkbfBq2q84DzJh
z++0na3Q28atyh32syD0mSpFHhPGuSJElDzGJNkiRpCyAO
SZIkDTGLNUmSpCGWqilNdzYsWpWsJEkziEtH9ok9a5I
kSUOs0ak7BqGpFQzmzJnTaCyAO+97sJF483fdqdFY0Gxu
q+7c4upgPTlw/thcyLdvvL+ReHvP3QWYOb+RpnNrMt66e

+5rJNa+u+8K0Hi8OzY90Ei8vebs3Pjvd82GjY3EW7DHXG
5Zf28jsRbO2w0Y7n9bTceThok9a5IkSUPMYk2SJGmIWax
JkiQNMYs1SZKkITbtxVqSw5O8errzkCRJGkbTXqwBhw
MWa5IkSZOYUrGW5MtJrkhyXZIl3WP3JfmLJFcm+XqS
+d3jlyb5yyTfSXJtkhd0j++S5Nwky5P8a5ITk+wA/ClwUpKr
kpzUry8qSZLURlPtWfv1qjoCOBI4PcmewC7AlVX1fOCbw
J+Ma79LVf0c8A7g3O6x9wD/XFVHAS8HPgBsD/wx8Nmq
OryqPrvN30iSJGmETLVYOz3J1cDlwAJgMfAYsLm4Oh84
Zlz7TwNU1beAuUl2B14FvDvJVcClwI7Awq3dOMmSJCuS
rOh0OlNMV5IkaTRsdQWDJC8DjgWOrqoHklzKWKE1U
W1hf/PnAK+vqh9MiP/CJ7p/VXWAzVVaNTVLtSRJUhtM
pWdtN2BDt1A7GHjRuGt/ubv/ZmDZuGtOAkhyDHBvVd0
LLAXemSTdc8/rtt0EuL6HJEnSJKZSrF0MbJdkJfA+xoZ
Ae4HDklyBfAKxl4U2GxDku8A5wBv6x/bPqK1Mcm33
M8A3gGf7goEkSdLP2uowaFU9BJww8XgSquqPgD+a5LIv
VtV/nxDnQeC3Jom/HjhqyhlLkiTNIMMwz5okSZK2YKs9a
1tSVbtu4fjLnnQ2kiRJehx71iRJkoaYxZokSdIQS9XEKdGG
WquSlSRpBsl0JzCqnvQza9OlqUlx58yZ02gsgDvve7CRePN
33anRWNBsbqvu3NBIrAPn7wHA7RvvbyTe3nN3AWbOb6
Tp3JqMt+6e+xqJte/uY4/GNh3vjk0PNBJvrzk7N/77XbNhYy
PxFuwxl1vW39tIrIXzdgOG+99W0/FuPOa4RuItXra0kTia2
RwGlSRJGmIWa5IkSUPMYk2SJGmIWaxJkiQNMYs1SZ
KkIWaxJkmSNMQs1iRJkoZY34u1JIuS3JDkvCQrk3whyc
5JjkjyzSRXJFmaZJ9+5yJJktQ2g+pZOwjoVNVzgY3AbwN
nAb9cVUcA5wLvn+zCJEuSrEiyotPpDhdSZKk4TCoFQz
WVNVl3f3zgf8BHAp8LQnAbOC2yS6sqg6wuUqrpmapliRJ
aoNBFWsT1/TcBFxXVUcP+SJEmtNKhh0IVJNhdmJwO
XA/M3H0uyfZJDBpSLJElSawyqWLseOCXJSmAe3efVgD
OSXA1cBfzcgHKRJElqjUENgz5WVadNOHYV8NIB3V+SJ
KmVnGdNkiRpiPW9WKuqtVX1nMcbL8l+Sb7U76wkSZLaw
K+GXG++E5MqP5XNVWyaWGVdGWaiCn6fPr7cDPG4rdxx
ceR9bm/CR3WB1RfXdfTVRIVbhEiGtRqFWdVXDeDYKw
t19v1ldYKGSdlx+dc2K23z1iQxlf9RgnzTH7M/XU7txqb
1RYRgMhAbwbcvtOZ5+nd8R/qUmCmqQ8NtTbqNwNLh
ERVMjyzb9ODzJO/F74iuhNdBd6mM5bAqPSKMhQRGgDB
12qn0wsTWuMhQeXyiX9FX7j94VnNdWnc6+Y+ZshMyMg
dUwjoJB4ynQHEDd5uXnN2tXEVs3d3P2nKdrfeRWW
zyQH8G2zV5N+27LNdhVtjUVhYV0bLQrO95Q4+BgCbKZ
qaDzWtGn1Or5gBkK1qn2YnSVRs+5Mf0vZ3WZ/ocxlrv
MVY9zJS3zyScIhLptrhb9WtQqWuNXgQQKr1l92E/UcK
fk6S+lQzzK9vj0qd3SLa9QJkA4QkNZ3kzYpPWPgsB4w
i4BnyBxU0j7smXnNhBb9pJZd1Jki8UsRTlJKxnLPNGy
Rq4cPktGE4FJvL8ZpNtwX7dpRzQm9mVrNgUQTWd2
WRJ57zVRKhX4fh5BqCgK7VYrE5oyLtM4ZvNYhdNAo
BEJjZWNR2XaYuadMMg2P2ksT97XVX2lPnHqH2pVk9
VsY9RtfKvPbq4g8PhW4Ko3WE9xkMk5hQrK8PGdWkjE
TRrbDYcK+5xY7JvO3tX5J8eR2FIaM5ZpFJ5vF5Z3T3Iu
K7bqHnD3J5rjz7dBYD/S0yl8nMXVA5Tw0GXqV8tDsD
5c3BXQyqeFhQ2uVd0Y8l+mFR9FfbKb3SbVn4RzGn

xFk1hlqrkpUkaQbJdCcwquxZkyRJGmKDehu0MU2tYDB
nzpxGYwHced+DjcSbv+tOjcaCZnNbdeeGRmIdOH8PAG7
feH8j8faeuwswc34jTefWZLx199zXSKx9d98VoPF4d2x6oJF
4e83ZufHf75oNGxuJt2CPudyy/t5GYi2ctxsw3P+2mo534zH
HNRJv8bKljcbSzGTPmiRJ0hCzWJMkSRpiFmuSJElDzGJ
NkiRpiFmsSZIkDTGLNUmSpCHWaLGW5MtJrkhyXZIl3
WP3JXl/kquTXJ5k7+7x+Um+mGR5d3txk7lIkiSNgqZ71n6
9qo4AjgROT7InsAtweVUdBnwL+M1u2w8BZ1bVUcDrgU8
0nIskSVLrNV2snZ7kauByYAGwGHgY+Er3/BXAou7+scBf
JbkKuBCYm2TOxIBJliRZkWRFp9NpOF1JkqTh1tgKBkle
xlgBdnRVPZDkUmBH4JH6jwVIfzrunrO6bZ9wSveq6gCbq
7RqapZqSZKkNmiyZ203YEO3UDsYeNFW2l8CnLb5Q5L
DG8xFkiRpJDRZrF0MbJdkJfA+xoZCn8jpwJFJVib5PnBq
g7lIkiSNhMaGQavqIeCESU7tOq7NF4AvdPfvAk5q6v6SJE
mjyHnWJEmShpjFmiRJ0hCzWJMkSRpiFmuSJElDLP8xB
VortCpZSZJmkEx3AqPKnjVJkqQhZrEmSZI0xCzWJEmS
hpjFmiRJ0hCzWJMkSRpiFmuSJElDbJuLtSTvTfKuJpKR
JEnS49mzJkmSNMR6LtaS/EqSlUmuTvK3E879ZpLl3XNf
TLJz9/gbklzbPf6t7rFDknwvyVXdeIub+UqSJEmjo6diLckh
wHuAV1TVYcDvTGjypao6qnvueuBt3eN/DBzXPf6fusdOB
T5UVYcDRwJrt3DPJUlWJFnR6XR6SVeSJKn1tuux/SuAL
1TVXQBVtT553OoShyb5X8DuwK7A0u7xy4BPJvkc8KXus
X8B3pNkf8aKvBsnu2FVdYDNVZrLTUmSpBml12HQ8MQ
F0yeB06rqOcD/BHYEqKpTgT8EFgBXJdmzqj7FWC/bg8D
SJK/oMRdJkqSR12ux9nXgjUn2BEgyb8L5OcBtSbYH3rL5
YJJnVNV3q+qPgbuABUkOBFZV1YeBC4HumSNKptkvIUmSNK
p6GgatquuSvB/4ZpKfAv8KrB7+ED3+Aq5hrHgD+ED3
BYIwVvBdDbwb+H+TPAL8GPjTbfgekiRJIylvVrXoMrFXJ
SpI0g2TrTfRkOM+aJEnSELNYkyRpiFmuSJIkDTGLNUmS
2SJGmIta1Yy1S2JL811baDjGVuoxnP3KY/1rDHM7fpjzXs

8UYkN/VJ24q1qVoypLGajmduwxHP3KY/1rDHM7fpjzXs8
WZSburRqBZrkiRJI8FiTZIkaYiNarHW2XqTaYnVdDxzG
4545jb9sYY9nrlNf6xhjzeTclOP2raCgSRJ0owyqj1rkiRJI8Fi
TZIk6f+2d+bRlhXVGf99DUg3Q8sYERS6QYEgtIC0tL0Eg
aDIHAI0EAXCtCTIrMQFkYBMRiMYYBGGyBRkZsUAG
hmCDc0oMoOAcSEIhJBIbBCQmS9/7Dr97rs23X3r1PPdvl
2/td5675zmfOx7T51Tu6r23tXHVGetUqlUKpVKpY8ZKGd
N0uKjbcNIIWmZOf2Mtn0AkiZKGttxPE7ShBZ6i0sa03E8
RtJi7azsPyQtJOn7o23H7FhQ7sFIIGm90bZhfkXS0pImSVq
/+cnQKP7OlLTzvJwbTUr1g5JWk7Ro+nsTSQdLWqqEdq
V3BiLBQNJU4HvAErZXlvRx4Eu2D8jUWxU4FfgU8C5wJ
3CY7V9laF0LdH/JLwH3AGfbfn0edZ5MOk2V6EZTgG2v
mmHb9NnYhu3NetVKevcAU22/mY7fB9xue3Km3l3A5rZfS
cdLADfYnpqh9WXgYtsvpuOlgd1s/1OObUlja+BjwCwH1fZ
xmVrXA9s2310bSt7XUvfgPZ6DTtu269W2pHuT7T+b27ke9
LYDNk6Ht9i+NkcnaU0HPghcCVxm++cttG4Edu5qv5fZ3iJ
T7yTg2116X7H99Uy91YEjgFWAhZvzmW3ueOCvgCcYajP
uVavrnbkyMDP9vRTwtO2JGbbdZ3v9uZ3rQe8DwEnAira
3lLQW8Cnb52Zole4HHwA2ACYA1wPXAGvY3ipHr9KOh
ef+n8wXfBfYgmhM2H5Q0sZzvmSOXAKcAeyQjncFLgU2z
ND6FbB8uh5gF+B/gNWBfwZ2nxeRzhdLGhV+lA4nIZOvd
vw9FtgReLuF3sKdzobtN5PDlsvYxklIeq+0mNXZz/YZHVoz
Je0HZDlrks4CFgM2JV6QOwF3Z9oG8BRwu6RrgFc77Dwl
Q6vkfS11D76Tfv8FsALQzCTuRnz2nkgzuIsByyVHoxnEjA
dWzLAPSd8EPglcnE4dLGmq7SNz9GxvKmkFYBpwjqTxw
OW2T8iQW65xrJL2TEl/kmNXYkvbR3XpbQVkOWuEQ3
oW8U57p4VdEN/Xam0HLs07Mz2r19j+93S8JbB5L1rpmq2
AlSSd1vFP42n3zrwAOB/423T8n8DlQM/OGuX7wXdtvy1p
B+AfbZ8u6f4WepUWDIqzhu1npGFbk7V5Ycj2RR3H35d0
YKbWerY7H5hrJc2wvbGknkfakvYFDgE+BDwATAHuAH

qeSbB9b9ep2yXd0qtOB7+RtJ3ta5Kt2wMvtNB7VdL6tu9Le
hsAr2VqjZEkp6lkSQsBbRzJqbYnSXrI9jcknQz8awu959LP
GGDJFjql72uRe2D7lnT98bN7HjLs+hJwKOGY3ddx/nfEQ
CuHrYF1bb+bbL0QuB/IctYAbD8PnJZm2f4G+Dsgx1l7V9
LKtp9Otq3CHGYq54GFJC1q+42kNw5YtIXe27bPbHF9J4
8Qs1//W0hvsu39mwPbP06zd73wHLEash3Q+Xy9DBzWwrb
lbF8h6chk29uSsvuuwv3gW5J2A/YEtk3nFmmhV2nBoDhrz
6QpYKeZnIOBx1roTU8Pz6XEC3EX4EdNnIPt3/agtfxsXrL
Lp3/LGTkeAkwG7koj9zWBb2ToNDN0DWOIKe8VcrQS+
wMXS2o6y2eAPVroHQpcKek54j6sSNyLHK4HrkijbCdbr2t
hW+Ow/F7SisD/AT0vqzTYzrqHs6PwfT2EcvcA4nlYtQkpk
DSRoedhnrF9KnCqpINsn97Cnm6WAprn+/1thCT9KfFd7U
S0j8uAr2TKHQXc2uHYbkzmfo2K3vwq4CZJ5xP3dW/gwk
zbIJzuA4AfAG80J3t8VzZ8E7hf0iNdWllL5cALkr5OzOYa+
CJxP+YZ2w8CD0q6hOg3V7b9i0x7OnlV0rLJLiRNIcJkcijd
D+5FvCdPtP1kelb7MrZ2QWBQnLX9iRizlYBngRuAL7fQ
azqj/dLvZqiyN/FQ9RIfdjhwm6Qn0vGqwAGKINCcl+Prtl+
XRBoZPy5pjQwdiBFiE9PxFrEctU+mFrafAKakuCbZfjlXK
/EwsbSyBTFjci2QG/fzNaJz+2vi895ALF/m8kNFsO23GRpp
Z+uVim1MNPcVYonmKfLv60RgPSLmZwdiJrfNjM5hwM
2SmvjPCbTbJPpsSQczFGd2M/F9vZWh1TgJ04k2sjEtZtWI
5a1Lgc/Zfq6FDoSDcQYxSHiKiKHNmrW2bUl/TjiAmxOf9
Xjb17ewb8/0+4jO/xW9vSsbLgS+RTz/77awqWE34BjCkQS
Ykc7l8HliSf99wERJ6wLHtXAkDyeWLVeVdDsxcNkpU6to
P2j7UcLha2Ial7T997l6lXYMRIJBadKSwAHAp4kXzq3Am
T12mI3WzsSszkRge2AqcFSzrJSh9wNixHMosBkRNLtITtC
npGnAdbZ/J+loYH3ipZ1r27LES7H53m4jXmQ9jWI79K4g
nLQmhmg3YGnbPWdfJef4ddvvpOOFgEVt/z7TtnGE47cR
LdtI0juVP4xtfB4YB4y3PU+xjR22lWq/D6Xl3k8TgdAnE+03
J36z0VwUWDMdPt4sxWVqfY9YmmkGPrsD79jeN1Pvg8T

MtYCfpmXMXNuWAF7raHNjiBjAntucpM2I+7kR4QA9A
MxIM4w5tp0BXGD7ZznXz0ZvbHf7mt25edS6xfZnSthVGk
n3Eu/dm22vl849ZHtSpt5Y4EBiQPoykcx2eu57pCSSbiaWfR
cm2ttviKSbw0fTrgWVgXDWugI+G14C7rF9dYbe7JyEpW
xPy9Aq3tl1aH+GWKq5LicYt7Rtioy1GQxNlX8B2MR2T8G
8HXoP2v74fuWdlAAACGZJREFU3M7No1axzNJ0/RXEy7
UzUD6rjSS9GV2xXLPOSfq57Y/1aFspJ/d+2+spgu8ftn1Jc65
XraS3COHklpgJK9pG0rWTiNm+zozGrFjEEWhzCxGO5K
bELMprttec81XvqfUokeT0a4YntOQ6HcWyJCWdQix/XsP
wZdDcQeTyRLxgd+Z2TqbqT21v2PkMtHTWSj6ryxOrQR
MY3n73zrStefb3BT5s+5g2n7XSjkFZBh1LjNSvTMc7Estl+0
ja1PahPeqt0fWyny7pwUzbmgDPrYGzbF8t6dhMrWE0Qds
tKG3bMrY7A3dPSMstudwvaYrtuwAkbQjcnqlVMrMUyrY
RKBvbWNK2/5J0NrFc9q00K9amPuOZxExYk4W7ezqXN
RMGvCNptbQE35TdyQqqlnQeMIl4dzTLbyY/caRYm5N0
E7A4MfNyKxE03yYAf8sW13batQKx7DZOw2uhjSeydXNo
BgLNoFHEfcgqKUQ4QpcD2xBO7p7ELFEOj0j6SyJB46PE
MuEdmVpQ9lm9mmgb/0H7jFyAhdNM8zSGslUro8SgOGsf
ATaz/TaApDOJ9frPEnEPvVLSSSjd2ZWktG3TJe0KXJGO
dwJ+1EJvQ2APSU+n45WBxyQ9TITe9DLCK5lZCmXbCE
TgeanYxpK2TSPF6dh+Mb28j5jLNXNiclfn9JOWTu4RRLv
rjIHbK1Nriu21WtjSTck29xDwCWBtYtXgRUl32s7Ss/3rTD
u62YKoifYhhsqzQMw69xTvJ6lZXvshw2tKQrs4yWVtnyvp
kDTAvUX52dEHEY7LG0TIwvVAr5mlnZR8Vhez/bUWtn
RzHPH5brP9szQQ+mVB/UoPDMoy6C+AT9p+KR2/n4g3
WbOXJZvGCSBG/msAT6fjVYBHba+dYdtiRGf3sO1fps5u
Hds39KpVmlK2SXqZeLGOS7+bUd1CwCu2x2fat8qc/r2XD
kfSZCIbb1hWo/+wzMXcdIq3kaQ7lnDYNiCWtm8EvttL7M
pI2VYSSfcRxV07Z8KuylkuS9c331tTuqbn761D61zg5BRY3
ZpSba5LcwnCGf0qsILtNuU2iiHpi8RnnMDQJIDdQ5FoSce

kP9cglnuvJt4n2xLxeblxiHfZnqIoPH0acT+usr1ajl7SHE98vl
ZJVJIeY+hZhTQgJWZ2exqQSjoBuMOpnlxlsBgUZ20fopjj
zQxlcZ1EjHyOtT1PMwElnYMFDUkC7s3tdEea1KkfxFBm
aVYg70i1kRKxK/ND+1UEyl9AFIuGNBNme3qmXsmYn42
JjOPniZmTZneQNsHjrdtc0jqQSC74BBFnNgO41fZPcmwr
TXKEZhI172Ytwdk+OUPrBmDHxhGStCRwpe3PZ9q2Db
E8+GHgdGKJ9lhn7E6RHPDzGKqF+BKwd64DXnhA+jKx
VP4Gkd3ftN/cwfJYIou8O9YvKwau0o6BWAZNU9w/JuJfH
ieWQJ+1/So9LNn0Q2c2v2Lbku6UNNmFMswK8y9Eh3liOt
4NuAjoqVMfwTbSOnZlPmm/yxJLeRMYyo7OrSsFZWN+z
iPeIaVKRhRpc4lxwCnEgKhNxfyRYiVnbn01G1ZmeJzmm
0R7yWVnYinvEWBTRR3C7xCOea+cCxxg+1YARXLW+U
SsY8+UfGZtL6lyu9tAtNXHicHGcUTCWJu6bZUWDISzptl
X9b+T/IDUSh6bAvtLeorIMGs1M1GY0gkBpSkdA9evHG3
7yrSM9FkiA/lM8rZyg7Lf29NOu28Uolibs/0PhWwaKe6QtI7
tnBjhbi4C7laUKTJR369Nwd5JHr5V128lZWUzAy83jlrSui3
NaI0679EPZu1uk/iI7Z0lbW/7QkVB4Da1+CotGAhnjYJV/S
utKJJhNkL0pTPUFWfWJFPMijMbTdtGiNIZyCWTUB5P
HdK1DC8ZkZsN2pdtriQd7XdhYK+U6NFqCdn2iWmlZKN
0ai/bbfakHCNpadszk83LkN/33Z2Ssjp3t7lZKRPWmeVFCl
G6H2zK6bwoaW0iPGBCOxMruQyKs1ayqn8lkz5fhivZqZd
km1H6/44WpTOQs+KY3oNxhKPxuY5zbUp39GubK8mItN
/k9JRyfE4mZv6uIu7nNIaWpntl3fT7mK7zU2lXXqQEpfvBc
xQ7FxxN1LxbgtjbtjIKDEqCQbGq/pXBZH4Ivl8Q6Ofs6NL
UNtc/SFqL6BsE3FQq47efqP3gYDMQzlonalnVv1KpLJhIW
p2In/uA7bUVuxlsZ/uEUTat0ieo8JZ6I0WbflBD9e5mi+1T2t
hWyWPgnLVKpVLJIRVKPYLY/qrZSuiRfqhPV+kPVHhL
vX6ko95dd2Fi6LF2XqUcgxKzVqlUKm1ZzPbdUTJwFv1YJ
qMyepTeUq/vsP0NAEkXAoc0mbQpfq3nunmVMvTLtkeVS
qUy2rwgaTXS1kaSdgL+e3RNqvQZ0yXtKmlM+plGuy31+p

nukiczGdq3tfJHpi6DViqVCrO2vjqHyOybCTwJfKEmAlQa
OnYJ6NxS79X0d/ZuAf1Iqgm4SVfJk1tsrzO6li2YVGetUqlU
gFRGZCeiltQyxO4DNUanMozZ7RLg2CB+oJC0B3AkMKz
kie2LRtWwBZTqrFUqlQog6TrgRQrsb1kZTN5rlwDbubsE
9DULQsmT+YXqrFUqlQo187Myd1JB42aXgHWbXQJs7zL
KplUGnJpgUKlUKsEdkmo8TmVOvG77dWDWLgFA3S2n
MuLU0h2VSmWBZiT2t6wMLM9KWgr4N+BGSTOB50bZ
psoCQF0GrVQqCzR1W6hKDnW3nMofk+qsVSqVSqVSqf
QxNWatUqlUKpVKpY+pzlqlUqlUKpVKH1OdtUqlUqlUK
pU+pjprlUqlUqlUKn1MddYqlUqlUqlU+pj/BzAMy4Ov8fa2
AAAAAElFTkSuQmCC\n",

```
      "text/plain": [
       "<Figure size 792x648 with 2 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "# Are there correlation in missing values?\n",
    "corr_df = pd.isnull(df).corr()\n",
```

```
    "\n",
    "# Generate a mask for the upper triangle\n",
    "mask = np.zeros_like(corr_df, dtype=np.bool)\n",
    "mask[np.triu_indices_from(mask)] = True\n",
    "\n",
    "# Set up the matplotlib figure\n",
    "f, ax = plt.subplots(figsize=(11, 9))\n",
    "\n",
    "# Generate a custom diverging colormap\n",
    "cmap = sns.diverging_palette(220, 10, as_cmap=True)\n",
    "\n",
    "# Draw the heatmap with the mask and correct aspect ratio\n",
    "sns.heatmap(corr_df, mask=mask, cmap=cmap, vmax=.3, center=0,\n",
    "            square=True, linewidths=.5, cbar_kws={\"shrink\": .5})\n",
    "plt.show()"
   ]
  },
  {
   "cell_type": "markdown",
```

```
  "metadata": {
   "_cell_guid": "51c717c3-2605-4d24-a284-c700a4ed6b81",
   "_uuid": "a37311774ca129c95271d09cecf7797f7b69e6f7"
  },
  "source": [
   "## Make predictions with the best model selected above\n",
   "I filled in all NaN with 0 and pass it to the trained classifier. The results are as follows:\n",
   "- True positive = 180\n",
   "- True negative = 35\n",
   "- False positive = 0\n",
   "- False negative = 27\n",
   "----\n",
   "- Accuracy = 88.8%\n",
   "- ROC AUC = 99.2%"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 21,
  "metadata": {
```

```
 "_cell_guid": "6d5371a7-4a8f-47fc-8151-d659cf3e3494",
 "_uuid": "2cec0c3b125d51d54162e02ecf0a50173250d16e"
},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "              precision    recall  f1-score   support\n",
   "\n",
   "         0.0       0.56      1.00      0.72        35\n",
   "         1.0       1.00      1.00      0.87      0.93       207\n",
   "\n",
   "    accuracy                           0.89       242\n",
   "   macro avg       0.78      0.93      0.83       242\n",
   "weighted avg       0.94      0.89      0.90       242\n",
   "\n",
   "Confusion Matrix:\n",
   "[[ 35   0]\n",
   " [ 27 180]]\n",
   "Accuracy: 0.888430\n"
  ]
```

},
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAYoAAAEWCAYAA
AB42tAoAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAA
LEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0c
GxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwb
G90bGliLm9yZy+WH4yJAAAgAElEQVR4nO3deZxN9f/A8
dfb2BljizCVnbGMkaUoQpEWJIo2SwvKkrUiJdKi+gpZ+s5
PURFKocJX1khHWPsYjBRmOwGs7x/f9w70xgzd64xd+4s
7+fjcR9zzz2fc877npk573M+n3M+H1FVjDHGmJTk8nYAx
hhjMjdLFMYYYYY1yyRGGMMcYlSxTGGGNcskRhjDHGJ
UsUxhhjXLJEYYwxxxiVLFCZbEZFwEYkSkfMi8peITBeR
wknKNBaRFSJyTkTOiMiPIlIjSZkTOiMiPIlIjSZkiIjJORA4717XfOV0yhe
2KiPQTkTARuSAiESLyrYjU9uT3NSYjWKIw2VEbVS0M
BAF1gaHxMk0SkEfAzsAAoC1QAtgFrRaSis0xeYDlQE2gNF
AEaA5FAwxS2OR54GegHxg4oapdmm+xERMKB51V4l5Istxg
4oapdROR54B2gkqqed2ObvBVCmVXADFW
d6pzu5ozzbue0An2A/kBuYAlwXlUHJ1rHImAmW
BT4CmwHngY1Wd4MYuMua62RWFybZZExDNjd4ANjvC6I
48rg22SKfwO0dL6L6/D/ifO0nC6V4eR4eAe4AAfwP51
ERABEpBjQCBEpBjQCBEpBjQCpgtIrmAH3FcCZVzbzbr+/iNx/g9E2cZVzbr+/iNx/g9E2cx2
NF9EzgFHgOPACOfyk5Ygw3ou4qX+PUsm5PN/Pn4
5D1V/UdVo4A1gAJNMPM6AutgAJNnPM6AutU9SjQALhJVpQ
D8H9A5HWIw5hqWKEx29x29aWKEx29Iiq+gLNgNgPc/
SWKQOcdL6PTKFMSAJj0Ol3y4+p9gPzdI1AJ50dMPmJ1Pajf
d728DyorI6fgXMAwonQ4xgxGHMNSxTG

QvAOuCxZIo/jqMBG2AZcL+IFHJzU8sBfxGp76LMBaBgo
umbkws5yfQsoKOI3IajSuo75+dHgIOqWjTRy1dVH3QzX
mOuiyUKk92NA1qKSJBz+jWgq/NWVl8RKSYio4FGwEhn
ma9wHIy/E5HqIpJLREqIyDARueZgrKr7gMnALBFpJiJ5
RSS/iHQWkdecxUKAR0WkoIhUBp5LLXBV3QqcAKYCS
1T1tHPWBuCsiLwqIgVExEdEaolIg7TsIGNSY4nCZGuqeg
L4EnjDOf0rcD/wKI52hUM4bqG923nAR1Uv42jQ3g0sBc7i
ODiXBH5PYVP9gInAJOA08AfQHkejM8DHwBXgb+AL/q
1GSs0sZyxfJ/pOsUAbHLf/HsRRZTYV8HNzncZcF7s91hhjj
Et2RWGMMcYlSxTGGGNcskRhjDHGJUsUxhhjXMpynY
+VLFlSy5cv7+0wjDEmS9m8efNJVb0pLctmuURRvnx5Nm
3a5O0wjDEmSxGRQ2ld1qqejDHGuGSJwhhjjEuWKIwxxr
hkicIYY4xLliiMMca4ZInCGGOMSx5LFCLyuYgcF5GwF
OaLiEwQkf0iEioit3sqFmOMMWnnyecopuPodvnLFOY/AF
Rxvu4Apjh/mhwgJjaOmDjrudiYrMBjiUJVV4tIeRdF2gFfO
od8XC8iRUWkjKqmx9jDJp2oKpei47hwJYaLl2MdP6/EcO
FyLBevxHDxSiwXrsRy8XLM1T+TlEk8feFKLFdi4rz91YzJ
ES4d3s7ZTQtuaB3efDK7HInGCAYinJ9dkyhEpAfQA+DW
W2/NkOCyoujYuGsPyokOzokP5hejkz+oJy4fdcWRGK5ny
JKCeX0omDc3hfI5f+b1oUiBPJTxy3/V5wXz+pDHx5rIjPG
Uc6cjmffpGNYv+Z4SN/sTdQPr8maikGQ+S/aQpKrBQDB
A/fr1s3x9haoSFR171cE5KjqVg3v82XuSeRedB/OLl2O5Euv
+WXpen1wUzOdDoby5KZDXh0LOA3zZonmuOdAXzOf4
WSDJ9FXl8vmQP7cPuXIl92s1xmS0Dh1eYdPyHxg6dCjDh
w+nUCF3h4C/ljcTRQRwS6Jpf+Col2JJUVyccu5SzFVVLvE
H5qvOypM7O090ML9qOjr2us7Skx6cC+b1wa9AHsomOUu
/qly+3BTM45OQDP4t40gMeXPb2bwx2c2OHTsoWrQo5cq
VY8yYMYwaNYqaNWve8Hq9mSh+APqIyGwcjdhnMmP7
RPfpG/ll7wm3yiY+Sy+Y6KBdtGBeCub1Sf6AnuSsvEAeO0
s3xlyfCxcu8Pbbb/Of//yHp556iunTp556iunTp1O5cuV0W7/HEoWIzA

KaASVFJAIYAeQBUNVPgUXAg8B+4CLQ3VOxpIWqsmj
7X2wM/wcReK997WQP7gXz+thZujHGaxYuXEjv3r05dOg
Qzz77LGPGjEn3bXjyrqcnUpmvQG9Pbf9G7Pv7HCN+2M
Fvf0QSUKYIEzoHUaW0r7fDMsaYq0yePJnevXtTo0YNVq
9eTZMmTTyynSw3HoUnnb0Uzfhl+/jit3AK5vXh7XY1eaLh
reS2u3OMMZlETEwMJ06coEyZMjz++ONERUXRt29f8ub
N67FtWqLA0WA9b+ufvLd4N5EXLtO5wS0MblWNEoXze
Ts0Y4xJsGHDBnr27Enu3LlZv349JUuWZNCgQR7fbo5PF
GF/nuHNBWFsOXyaoFuK8nm3+gT6F/V2WMYYk+D06d
MMGzaMTz/9lDJlyjB+4k4cmyhOXbjCRz/v4esN
hyleMC8fdAyk4+3+doeRMSZT2b59Oy1btuTEiRP069epUa
NGUaRIkQyNIcclitg4ZdaGw3z08x4uXkmrZWW+Py9L+vKn4
F8ng7NGOMSRAdHU2ePHmoWrUqzZs3Z8iQIdx+u3f6f6Ts
1RiWLzoX94c8EOdhw9yx0VijOyXU2q35yxmdkYY1y5fPk
yY8aMYcaMGWzZsoXChQsza9Ysr8aUIxLF8XOXeH/xbr7
f8ic3F8nPJ0/U5eHMAohYNZMxJvNYsWIF/xbr7
p05cvnyZwoULezus7J0oomPj+3s1WlmFhebFaJPs0
rUyhftv7axpgsJioqih49Bz2x4x
1+4/yVs/7GDf8fPcU/UmRrSpQcWbvJ+ZjTEmqfz583Py5E
mGDx/OsGHDKFCggvLdDukq2SxR/no7i3YW7Lj9GLcU
L8D/danPfQGlrJrJGGJOphIaDGMmTa1Yh
t7yej2yTaK4FB3L1HmLTyD+JUGdiyKj2aViR/Hh9vh2
aMMQkuXLjAW2+9xc8XfhF2/0/yxYsXt7xsf28YKeDssSYY67yww8/0LduEs
2/83IH3dyKPIrWvezOsPBL6/83LfpdCjWZAUT
w4fPswLL7zAzz+NsRxdhk/UBUUzyiLdWmqwTGPdQIZ5DgGUuvv1P
6qRBfPdeQJlVu8nZYxhiTrPnz3s7Rq+BQhCb9jgNFTnWcOLr8xgYuTvnDPduzDvzz8b3Y1n
7YsmSguXolh8so/CF59gDw+wrAHq9OtcQXr5tsYk6lER
YcIEmjdvzu2u2338748748ePJnz8/efJkrQd8s1yiiYp
Zy7Rvm45XnugOqWL5Pd2WMYYk6jw6PD69WrVoCBsVr
XVV7n99tvx9c2KkuURx+C6YYl3Gd69KwQuav2zP

G5CynTp1i6NChBAcHU65cOebNm0e7du28HdYNyXKJIpc
IP/a5y8aIMMZkSsHBwUydOpUBAwbw1ltvZdmriMTEMd
Bc1lGgbFWNOrrX22EYY0yCPXv2cOLECe6++24uX77Mn
j17CAwM9HZYVxGRzapaPy3L2mm5Mcak0aVLlxgxYgSB
gYH07t0bVSVfvnyZLkncKEsUxhiTBkuXLqV27dqMGjW
Kjh07smTJkmzbA0SWa6MwxhhvW716Na1ataJKlSosXbq
U++67z9sheZRdURhjjBtiY2PZvn07AE2aNOGzzz4jNDQ02
ycJsERhjDGp2rp1K+auu+ZWtkW0Myak7Vl/Ri/e0A/Ei77Ov9UkJTkWSQZR6Bf6Qog/Ta2j6DVDJ2
YnCGGNScO7cOQYOHEj9+vUJDw9nypQplCpVytthZTIsQbnSiOXOG2rVrc+TIEXr27Ml77771HsWLFvB2WV1i
iMMaYRM6ePUuRIkXw8/OjR48e3HvvvTRq1MjbYXmVV
T0ZYwyODvw++OAD/P392BkCGN
Yu3YtvXr1IiwsjEceYSbbbrLhChKzKwpjTI7Wt29f7r77bs6
cOcOCBQuYN28et9xyi7fDylQsURhjcpzEnf1x4efOiFxsReUE5dr0nDgSt0fMB4
07d9K2bVsvRpV5WaeAxpgZgcZffu3fTq1YsOHToA0LdvX17dOOxH
CmiMMamIiorijTfeIDAwkG3bthEfH+/tkDI9SxTGmBxj3bp11K9fn8GDB9O6dWt27NhB165dvR1WpmdVT8aYHGPfvn0cOXKEWbNm0Sb+4/XNRS2SFe0JjzBtyEvcGYbS4jAYytEzL3nzRhOcxDacoaG0HEpf
4+mc+fO7Nmzh3c9GzPUAG0Xk1XdmahYb2CnqrYRKuAPSIyU1WveCouUUAGSqqQd9VdGsDutl5duBKh2aIFKlzBOd2xB2kM4Ks0lFyWrSw4tB70bGiXHfX0h56R2sxKlmAs8c4I5M4FERvcEI77eRxgcBB9C12qSWWLdzBByWfnfBhEiGYSoJnQr2XSMNB2oBmDtbzZgXPAFmdXJjTTGUsPZAS7hDrfnqwUgYz2tGwpAdNTs1YHwoCwq1ZXGK
7cuVm+fDktWrrTwdjhZjidvj20I7FfVAwAiMhtoByROFAr4
iqNv3sLAP0CMB2MyxuQAsbGxfPrp+TNm5cXXniBLl26
0LlzZ/Lly+ft0LIkQKT1Y9lQOOJJqOcH6W2EQgADgKbAdeV
tW4pCsSkR4isklENmW1xndjTMbasmULd955J48//vhVjhw5AoiIEDlnBMFA3zN36kN4bQHGHnIK1Yvp0A9yqTsa
uAi73q1Id2gEJnf5I3MLerf2YoRXfNf69BfVhMpaVo3jDOrMFJlP4UfmS2vrB7rsaH/6uh8iLsADvz3hnPyJCq9D5P/ZguzyqszsTt3BgEVnVeWlAPjG5xWr2+dQtXnDD/Sz0mVG+fO2XbEuYzmyUSxEagiIhV
EJC/QGfghSZnDwL0AIlIhvhZ0ODBg3z++eeeA
YzCh/fv3M2nSJIoWLerlyLIXjyUKvY0B+gBF2xHjCv
0R6iUgvZ7G3gcYish1ah1YDryqic9FZMxcuK7733HjVq1CA4OJgjR44wb948Pv3ruRlRPmHp4kiR2j57U6t40Oz3mJcAvGKaYWfx0l7fBsPS+UwPwqGO+c/e9H4gzO0X8NrF8VcXPw

q1GDQoEGcOnUKgAoVKng5suzJnsw2xmQpa9asoVevXuz
cuZNHH32U8ePH4+/v7+2wMr0beTLbeo81xmQZJ06coFW
rVpQuXZoff/yRhx9+2Nsh5QjWhYcxJlNTVZYuXQrATTfd
xE8//cSOHTssSWQgSxTGmExrx44d3HPPPbRq1YpVq1YB
cO+991KoUCHvBpbDWKIwxmQ6Fy9eZNiwYQQFBbFjx
w6mTp1K06ZNvR1WjmVtFMaYTEVVad68ORs2bKBr165
8+OGHNuKcl1miMMZkCseOaNUqVL4+PgwbNgw/Pz8a
NasmbfDMljVkzHGy2JjY5kwYQLVqlVj8uTJALRr186SR
CZiicIY4zWbNm2iYcOGvPGZyyzRu3JgHH3zQ2yGZZLidK
ETEbjMwxqSbDz74gIYNG3Ls2DHmzJnD4sWLqVSpkrfD
MslINVGISGMR2YmjGw5EpI6ITPZ4ZMaYbEdViY6OBq
Bhw4b07t2bXbt28fjjj1sHflYql14iMjvQEfgB1Wt6/wsTFVr
ZUB817AuPIzJmv744w9eeuklatWqxX/+8x9vh5Pj3EHm
5VPanqkSQfxaZlY8aYnOfy5cuMHj2aWrVqsW7dOqteyoL
cuT32iIg0BtTZXXg/nNVQxhjjyubNmNm3n66afEZvXs3jz32GO
PGjaNs2bLeDstcJ3cSRS9gBM4oZ7K
Fy4MCLCokWLeOCBB7wdjkkjdxJFNVV9KvEHInIXsNY
zIRljsqq4uDimTZvGunXrmDp1K1KtWqVSMsLIxcuexO/Kz
Mnd/eJ25+ZozJwcLCwmjjatCnPP/88P/30ExcuXPB2WCZ
FK8oRKQR0Bi4SUQGJppVBPDxdGDGmKzhwoUL
Fjx+Ln58e0adPo2rWr3e6ajbiqesoLFHaW8U30+Vkw
yXLl1i2rRpdOnShQ8++IASztx5juI2VT2UQfGk
yp6jMMb7IiIimDDhU1555RXCy8s5cuQIJUuW9HZYJ
RRH5UEQWiciK+FdaNmaydpjm3EkxuQbF4Dh
YSEhABYksjm3EkUM4HdQAgTgIPAJg/GZIwxKqtwTOMab7
+nfv36DBw4w8kKZNm7Jjxw+f+UL7y4iRKFmqEY3c
qr8Av4jIL54OzBiTecTFxdG+z9e+3kFnOnDnD3+L+LLRGmcW
xJ1FEO38eE5JZ5pfV+dCMGKKcjNiLGvVrACz
/feUK1cOX1/f1Bc22mMlbirmIY6B4VNNA5jbyEaFF
q/bt28f999/P448/ZmBwMADVq1X3+XmAN4wmpU5ma+kD24NVMXtUVntOJQdcp/eAKQwmNNLeAZp

**DwpPZxphs5vLly4wZM4Z3332XfPnyMXHiRHr16uXtsIyX
uXrgzgd4HEcfT/9T1TAReRgYBhQA6mZMiMaYjNK7d28
+++wzOnfuzNixYylTpoy3QzKZQIrPUYjIdOAWYANwB3
AIaAS8pqrzMyrApOw5CmPS1/Hjx4mLi+Pmm29m3759H
DhwgPvvv9/bYZl0diPPUbiqeqoPBKpqnIjkB04ClVX1r7Rsy
BiTucTFxTF16lRReffVVWrVqxZw5c6hSpQpVqlTxdmgmk3
HVmH1FVeMAVPUSsNeShDHZQ2hoKHHffffTc9e/YkKCiIk
SNHejskk4m5uqKoLiKhzvcCVHJOC6CqWGujx6Iwx6W7u3
Ll07tyZYsWK8eWXX/L000/bMxHGJVeJIiDDojDGZs2
cpUqQIzZo1o3fv3owYMcK63BuSbSbVTwMzGGrONuT6HD
x+mb9++HD16lPXr1+PjY6ME5ESe7hQwzUSktYjsEZH9Iv
JaCmWaiUiIiOwywrkGMST/R0dF89NFHBAQesGzZZMh5//
HGy2omhyRzc6cIjTZzPYUwCWuIYa3ujiPygqjsTlSkTA
Zaq+phESnlqXiMyUkOHTpE27ZtCQ0NZU2bNsycOZMy
3g7LZFFuJQoQoRKQDcqcqqp7rmPdDYH9qnrAuJ7ZQDtgZ6I
yTwLfU9fh3rx5tGvx5tGvXzh
qrzQ1JtepJRNoAIcD/nNNBIvKDG+suBxxJNB3h/CyxqkAx
EVklIptFpIt7t7YRtjElNVZsyYQMGDTh//4pF
HHrEkYW6O20O20Ub+G4OjgNoKohiOlZjrrrZnzjrRZnzx7uvfdenunnmUXSE4ugCJ54+ji/Kk
Zf6nqhdU9SWqGqyqEGqyq9VW1vp0dGeMQExPDSPVVEKEKZbhIXqAzkLTKagHQRyi0hBH
EjCAwMZLkLTKagHQRERyi0hBH
H1K7bqO+I3JsXx8fFizZg0d0dO3Zkz5499Or7OVvWvy7g5PHojo8mh
3Pmr6gvUBC4DX+PobjzV8ShUD/RlV3iEgvEe
nlLLMLR9tHKI7OB6eqalhavogxOcFfff/3Fs88+y6BBgxC
1axMyZMyldurS3QzPZhCRC3Qz37wrkOAasbePlr6Lz2p61
UWxsLMHBwQdOpSoqChmzJJ2u3BY4895u2wTBbi6Qxuxo
IbhF5W0RqpmUjDLjvTLb4buSHypn5ShGN++KAE+13lbwIzlpkEZEZV

**CpJgpVbQ40A04AwSKyXUSGezowY4zDxIkTCQ8PZ+bM
mSxdupSqVa+5MdAYj7quvp5EpDbwCtBJVfN6LCoXrOrJ
ZHeqyvz58ylfvjx169bl1KlTABQrVszLkZmszKNVTyISICJ
viUgYMBHHHU/+admYMca18PBw2rZty6OPPsq4ceMAR4
KwJGG8yZ0uPKYBs4BWqpr0OQhjTDqIjo5m7NixjBw5kl
y5cvHRRx/x8ssvezssYwA3EoWq3pkRgRiTk/33v//ltdde45F
HHmH8+PHceuut3g7JmAQpJgoR+R+gKSdcGdKKGg3TmcDGSoR+UZVHxeR7Vzd9YaNc
GdMOoiMjCQ8PJx69erxwgsvULlyZVq3buu3tsIy5RoqN2SJ
SRlWPiUiy/QGo6iGPRpYCa8w2WZ2q8uWXXzJ48GB8fX
3Zu3cvuXN7rMd9VDiF/BSWjZmT
E63a9cumjdvTrdu3ahSpQrz58+3ZGH81KJ/vI0uFoMcKtQk5M9BddS5
gxmR327Zto06dOoSGhhIcHMyvv/5KYKDV4JrMz1BbxYs
4rhwqikhoolm+wFpPB2ZMdhEREYG/vz/vvz+BgYYG/vz+BgYGMHDmS5
557jlKlbDBHk3W4aqPwPwBdVHd16lAEDBrBo0SDBrBo0SJ2795NuXJXJx+0yJuN46oE7VdVwoDd
wLtELESmelo0ZkxPExsYyceJEAgICIWLBgAa+88golS5Z4Zk3oHV0vejHGzSNJBhqXp+9Ul+4Pk2jjYRzxiLOsHGrZOHgh+Qtsp87t6JnfaJwR9xV3Lh59lfR9y
vRWvLLbZlPBbbnaM+jwaKzNhHzL1qmjkHrNVfNlkZkXwVqHrRxP78WhMQPYdxHKN8eqOnvKVNDzv4O

**FMaTFi1aRGRkJM888wyqyqlTpyhe3HqsMVmfR9sogP+J
yBIR6SYi3YCFwKK0bMyYzCoiIoKOHTvy0EMPMXHiRF
QVEbEkYQzuNWYPAf4LBAJ1gGBVfdXTgRmTEWJiYh
g/fjwBAQEsXLiQd955hzVr1thT1cYk4mo8iirAR0AlYDsw
WFX/zKjAjMkImzdvpn///rRu3ZpJkyZRsaL1dWlMUq6uK
D4HfgI64OhB9pMMicgYDztz5gzff/89AHfccQe///47ixYtsiRh
TApc3fXkq6r/53y/R0S2ZERAxniKqvLNN9/Qv39/IiMjCQ8
Pp2zZsjRs2NDboRmTqbm6osgvVF5HYRuR0okGTamCz
jjz/+4IEHHqBz586UK1eO3377jbJly3o7LGOyBFdXFMeAs
Ymm/0o0rUALTwVlTHo6d+4c9erVIy4ujgkTJvDSy/h4+Pj
7bCMyTJcDVzUPCMDMSa9hYaGEhgYiK+vL5999pVm
njVttTBrYo6Ym2zlx4gRdu3alTp06LFrkeOSnQ4cOliSMS
OPJgoRaS0ie0Rkv4i85qJcAxGjtT6kzI2Ii4tj6tSpVKtWWjVm
zZjFs2DCaNWvm7bCMyfLc6espTZjVkwCWgIRwEYR+
UFVdyZTbgywxFOxmJyhQ4cOzJ8/n6ZNmzfI3wRmsq4z3wXKG
CMyRbc6T1WnGNlv+mcvlVE3LmfsCGwX1UPqOoVYDa
OUfKS6gt8Bxy/jriNAeDChQvExcUB8Vz6AkpiBVyrF7gB2kt6LYrD
MpmXh9euBRNK97r2NAAAeNA5fqQp8/0AD5NzBZQ6+xnXlPcC
3w/oeKu3VWG8axyPaXOvjDlWm6YqBkLSqAi0RFfpzfPxRAGSU1BDh8ES
zALz5Q2C6tRAYrd0FKU0W8jCe1Hh8Y3sTGtsgp/H9yD6wW
pdCF0fNY4OM49UyyxVVh6SxH8Lp9P2Fmd1mbNmK/oDmAzSZDt8
mKCEMPgSeSXzcJI3V9q44t+zbwNDRxu1IwNMG9eH0HPgC8DQ0y+Snkz8X
DGH2+HNsmIBiy07W0bGNdT9Dga5EJmSy5jkAE0Ax3Ez/tAiJVQKSnJGwu
BHEtTObVzBj9VKZaRe7h4KGH7WNqHR5Rv2yybKjNxM+V4lxTJWLKW
5DxnPdtqrvMTjl9xQGhZo01e5bxSehBJnGcUqWvBXxQr0L6Dg0Nwu/P
bmgzKuVuHzuYzl04uyKVqX9h3hzDnBihNn7EFk6+DxWrKyKrqAMcKTs
t1q4Sk/GxO+jyOmRGTeVNDYexGUdHGUv2xdzOUxbIWVmV0txYjAFFKSs
KaO2r2XKt3SC7wHZa9yb9Ug0Hh1n9NQgYqYiG+ijpVgQXawcqkZUFUcl
ETJW7JJRUM2ykAWqBw7LXN6WEqvEJUO9EmQhQ8qMQBL3MSdMUUPO9CLv
iSMJXhJ0FfS5pUFjnjZHnH7QOfA8dq13nBg4Ka4fbwgEhLb+c61X3
A5iv/Da5Odbv9q/dz/KI2LYaCBg3dMhFH3pEkZ9ssk/UAgsqScwpHsy3
E/Q9/s7k4hZMJyzpEz3FZc+8MNqvY9f9pINhkTv/sD9GFRZdTKPQRbT
P8IQ7+QzkGo0ZBGHKbiG/Qfn3/WaJ3bJaQ08cXfXHQUMOf5G6Fuw3TcIxE
Nr3CmvDGsZ8L57xrPZXUWW84lpwo2XBMJUWRCUSuYDq3VcLTFtdtJ7lvm9
J95/bv/ZpD/l4FFNPN2eLu6dg+1ggfwOgHWXHOBB7jn8EVE2u8aFP2bB1S
AU6DrzjRvttj8+9yAVy1wNC2G1AvGa4iXCRSDyCT9cs4uQgW5OEBYUs+
QW2UPN+kg6jU0x2/hYTv6PWSIwbvJ69pcnHt4mUvq5/Txf/ujODh8X
WPCp76pXQMnLVxaMRbUszI0pY7nkR3HKEx0rGgsQ6EKzrRnf7/V0+
lVUPqfHIh19a9fuNVLCzXTdUY3LVlbpOyT/I4IJzXqKjBYhn9k1YS0n
WpPhq8sd1rQFhjeqkm/W3c0U0KR5oCtvU1qv2pA9BzA/wLKyY2aGqCDl
pMSyEdI9pSxbqn6mD3fi6PEqCsCg8Ko9Kg7gTRYfuWMdp8cGz6k7dXvfr
4TkOIuCpGcQOt0YajCs9yQ6JFxyJyUOi3TkIQNVF15p1Gbou+4BLHOzDk
vz6PBG1/NwE8HsRfmIK8H1oOOe5VHKWPQDCaTPfT1uk5oUlGjFaFLp
G2U8eAv2Vu7GQlK5X0d4gNSGbVFOuAPSiTv0mv8KBTk5wtAMJHX5KTs
GYx2wjI9cHK3ZBxkVAL/J/nyx2nJ6DZ/R5r4UQMIn6dbtoWKjuSiwNXbZ
5LHYXXwGXbfxzt+K4hoHgoOX9Mo/zKtIA4ZVX8wAgfe/MQ5vUgMDhDbuy
OHOzgwYP06**

NGDZcuW0aBBAxYvXkzdunW9HZYxOZY7VU9XjY3t7D
m2p8ciMjledHQ0oaGhTJo0iZ49e1oHfsZ42XU/ma2qW0Sk
gSeCMTnX8uXLWbhwIWPHjqVq1aocOnSI/PnzezssYwzut
VEMTDSZC7gdOOGxiEyO8vfffzNo0CBmzpxJpUqVeP311
ylRooQlCWMyEXceuPNN9MqHo80iua44jHFbXFwc//3vf6l
evTrffPMNb7zxBtu3b6dEiRLeDs0Yk4TLKwrng3aFVXVI
BsVjcogzZ84wfPhwgoKCmDJlCtWrV/d2SMaYFKR4RSEi
uZ1da9iwpyZdnD9/nrFjxI5/nrFjxxIbG0uxYsX4/fffWbFihSUJYzI5
V1cUG3AkiRAR+QQH4FrgQP1NVv/dwbCYbWbBgAX379u
XIkSMEBQXRokULas6O2wjDFucKeNojgQiWOM7IeB
Ns6fxqTq0KFDtGvXjkceeYSiRYuydu1aWrSw4daNyUpcX
VGUct7xFMa/D9zFs76+TapUlY4dO7Jz504+++ ++OAD+vfvT54
8ebwdljHmOrlKFD5O7Ar5yYQB4z2zdzmzErBrYBOvoSHBx
M8eLFue2227wdljEmjVwlimOEjWbXK0Ez4wdvNTmQ0
M8eLFue2227wdljEmjVwlimOqOirDIjFZ3j///MMPQoUMJD
g7mzTffZOTIkdb1hjHZGKtQoEH88
88/DBo0iCFD7I5qY7ILV4Avm/4ni3uUckNwAABvCSURBVAyLw
mRpw4YN4/333+fOO+9k6dKKl1lTx9shGWQWPQRJfvm/yc
hATNZy6dIlzp8/T8Fx1eJevTu33XZbPXr0IH1/MvJdr1jKt
JfZfba7b0qVLqV27Ni+Ni+88AIAVtWpVm7dm2bNVm7Dm7D/bu
O2vv/7iySefpFWrVFWrVvz000+MGpUFjJmYJbAQMmcVpbdrImbjT
6oqCjeeustXn31VVyXevHmzDh0/6uqcVxlXxOIyHd3Yq8
edd6hatq3wzLLzGCGZ2CCrejLJOnfnuHAMMGDKBJkyExsZok
QJvv32W0dsdfooRAlijjMVVSV77//noDoJQPHVCdsnoXQw
myt8MyxniRJQqT4OTJk7tRp04RJfvvtN6Z
mULBggW9HZoxPr6+/P9334334wdO5ZNmzZzx+61WVqZaE
jskY0wmYIkih/v555+pVt9ywowePQoVK6sNoPPfRHaXs+jYh++PRzFv35MQl5pXRETqfMnmlR8
9zkYYxwsUeRwhtj/zU+PSLG0Q0qmJ6ZG4mV8NzbwpaskzgkN
DTsq5DCqyvTp02natCmnTp3Kp06dDmvA6Vg8VpRBOZhS0nF
SSRRPRiTSSvzhtlM1K/F+1d2o0ZgOsr6vJG0Tj+Zw6YXm0
bqvTeq2SuVS3W02bYDqpdTv10yAOP3+zG0KJJNmowmQ
b+UiyQtTuY3mV2y5xsWQBPKkSt9g0b9kH5Tkdqqo+RzSe
Sfjv3RZjHKZQa0N2v4y5M5Fmd7mcG2tKwcKBFkmxxpmjM
lUXKmJSV5HVqlJMUA/V0tIzLN8sNCK8LNj9Is1kMC3+N
QJvv32W0sSxuRAlijjMVVSV77//noCAAMaPH+fP
myt8MyxniRJQqT4OTJk7tRp04RJfvvtN6Z
mULBggW9HZoxPr6+/P9334334wdO5ZNmzZzx+61WVqZaE
jskY0wmYIkih/v555+pVt9ywowePQoVK6sNoPPfRHaXs+jYh
9zkYYxwsUeRQkZGRPP/9888/z00038c0337B69Wo8KjE
DTsq5DCqyvTp02natCmnTp3Kp06dDmvA6Vg8VpRBOZhS0nF
pKx+IQf68ssvVatGp9++im33HIL2zZbq
MGEFERAQiwnfffceaNWssRhj3GKJptbsmQJtWrVYS

USxYsACAYsWKWVuEMcZtdrTIpo4ePUqnTp1o3bo1efL
kYcWKFfTu3dvbYRljsiBLFNnU6NGjWbBgAaNGjWLbtm
00b97c2yEZY7IoUVVvx3BdCpStqlFH93o7jExp8+bNCR34
RUZGcurUKSpXruztsIwxmYCIbFbV+mlZ1qNXFCLSWk
T2iMh+EXktmflPiUio8/WbiNgYmmlw9uxZ+vXrR8OGDR
k2bBgAJUqUsCRhjEkXHksUIuIDTAIeAGoAT4hIjSTFDg
L3qGog8DYQ7Kl4siNV5dtvv6V69epMnDiRF198kRkzZng7
LGNMNuPJ5ygaAvtV9QCAiMwG2gE74wuo6vJE5dcBT3rwemMOgbkZOjKDqc6Tyhl2g8F8v5m1jgUAj7qqXgqJ3m+8Ae/T8eUY+FRZhWPUbkozmF+SOg4v/hVNWxVZgMc7YMzk4a9eMz7i1KDFpVJ9yOw4o5DZm2W0NVBEcH2VAfB3OX6C7oNIjPiBEt8Cp0yANm2M3gLUxs3sNVf3EPmXJIg4UkFJCX2EA5Z1OM9Fdtb8CPD7oJ8pHiA3YzVNDl1Jz4+wHbScvp25eHKLJ3c4YcJ7n9ZkdEXgDHn6+lBT9bB5+5y+0s5PBUI+J4fBzYFpXVpL6MbYUkn3JX6gt3dK+X5lMmvkxYt
YT7bz9ddf8/TTT1O3bl0WLFhAgwYNvB2SMSYb8mSiKA
ccSTQdAdzhovxzwOLkZohID6AHQL6bc3Z1ypUrVzhw4A
DVq1enY8eOREVF0a1bN+ubyRjzMZ5so5BkPku2fkhkPku25VxEmu
NIFK8mN19Vg1W1vqrWF0lutTnD6tWrCQoKolWrVly6xN
8+fLx/PPPW5IwxniUJxNFBHBLoml/4GjSQiNHIB8ieAKFwF2qlqp
AfjybJOnjxJ9+7dueeee4iKiuLTTz+18aqNMNRnGk6eiG4EqI
lIB+BPoDDyZuICI3Ap8DzyjqnbPazIOHDhAgwYN2L
K+99hpvvPGGDSRkjMlQHksUqhojIn2ApTg8l+yWKnz+xBNzuWoQUIFunfvTr
Xs75nwJvAiWAyc4qpZi03ueb3Zw9e5YiRYpQoUIFunfvTr
du3ahVq5a3wzLG5ED2wF0mc/HiResGR5VxYVOdyUE4tl+JT5t2tcXBCGsIVgjDt3Tq
I5gx5sbdyAN31gqaiSxcuJCbb76Z9+/fd1g3+cFCgF1xF2sprDqRSEr4yEaGddVc6ctjM5T+7JRuiRmyCvyGvUmzuqYmVdx43G+e6r7chlEw3RbzzwXGq6Vdc+SiH8Dxj1bMrAj3cs9WxiN2xhyqFOEvRFRBLeaHSbwoVx7gVA8F6nlyFA18bGJhYfQ/jd5VTyq06qnS8mcHPTQpMqVxAT
YYSxSZQUxMDE888QRz584lKJMrHWqQeX8rt1f6m5y3FtYDg5wQW0qOLiLi
MAaxTQK+Kr/bLnTs3pUuX5t133yUkJMShEmU7FE4S
UbN27kjjvuYMuWLQBMnDiRoUOHkjdvXi9HZYx7yivVbT7BxCHZzt9zNE
kcHOnDlDnz59OD6nh4qDcFnx17cGOBf6F8W2rvKyVGrjo5
CP169vjIsaYrMGeo/Cgy5cvIz7M98qe3jc/lbh4Ot9YnP/M+AqJeOPXBX8AgZGKAKv8ovmLvbWjFHGB4N
LlwsfHx9uhGWNynETw7cFFOtnLlSurUqcMbb6aMbJq3TLWWHTw0+WbgyDOOjtMkYIlx7SC9XIgNvdmjq9D8VKpSdiXYuRRVrgr27M1hTMAzSh8+/ADPJCRDcBlQn+DvZOxmOO2cYi2mgWjgTtYWl2yhR6E0i6TQ9v6OuKWBkK+/g6PbBa+Igkn9eBDV9+V2Bg0eoxsYV6b8x4ltaZkiy
ljGGJNmlijS2c8//8yTjz7Jv/7vVnjq+oazakNnsWx5K/ZuVQJ8U/kRWOZ0BomtR8eK2mjsRhjboi1Ua
SD7du3s2fPHjp27PVdaMvZgvoVncWRfiOzjoDSyxNrPCvfFzH3nLXIdNc9unMVYnM2Wh4hivJRS+ISg
PWVV6hbty6vQqqwB5YwsoUIiOTfKpKeTjtrQtFu9ZnixBL4VLSg3nX3rX1G5YRgvHLNwOHoWuTTDBJGfYAXlG7v2BkdZcSIbxYIwC0ZPBSvixoRpYxrbPU9ZjF9TR1V5KaRhbbOsdR6uq9KCbx6p9Ztu1T0yKMPqcmKNX
OWVV6hbty6vvPIKK0dHRiIglCWNMtmK3x6bRjz/+SJ8+fT

h8+DDPPfccY8aMIU+ePN4OK11ER0cTERHBpUuXvB2K
MeY65c+fH39//3Q9HlmiSIOwsDDatm1LzZo1WbNmDXfff
be3Q0pXERER+Pr6Ur58eXLy0LPGZDWqSmRkJBEREV
SoUCHd1mtVT26KiYlh1apVANSqVYuffvqJrVu3ZrskAX
Dp0iVKlChhScKYLEZEKFGiRLrXBliicMPvv/9O/fr1uffee
9m3bx8ADz30ULapakqOJQljsiZP/O9aonDh1KlTvPjiizRq1
IiTJ0/y7bffUrlyZW+HZYwxxGcoSRQouX75M3bp1CQ4Opn
///uzatYtHH33UrQziI+PD0FBQdSqVYs2bdpw+vTw+vTphHk7d
uygRYsWVK1alSpVqvD222+T+DbxYsXU79+fQICAqhev
TqDBw/2xldw6YknniAiAwMJCPP/7Yo9tJaT+Gh4dToEABgo
KCEl5XrlzxWBzh4eGpjvnuqsyY8d4+OGHPRFaulBV+vX
rR+XKlQkMDEwYPiCpFStWTcWcPvtt1OrVi26du1KTEwM4D
gpbd++PYGBgTRs2JCwsDAjhw5QvPmzQkICKBmzZqM
Hz8+YV2DBw9mxYoVnv9y4PiCWemVv0NHGhOm
TdMtW7Z4dHuZZ0c6dO70dghYqVCjhfZucXXXT06NGqnrx
4kWtWLGiLlmyRFVVL1y4y4oK1bt9aJEyeq27du1Ysp0Kn
umvXLlVVjY6O1kmTJqVrbNHR0R0Te0/LFjx/TWW2/1yLq
TSmk/Hjx4UGvUrGmu23LFne25KjN48GCdPP3+29uLiYm
5rvhu1MKFC7V169Z1Yn169KFPeY10RNGuMGGfIE0Zz1xJR0XeGe
nj2qqvrGG2/o1KlVd21Va5e2aNFCVVWWPH
j2qmzdvVlXVs2fPauhqoheHq4tW7ZMNp7k/r/fl+f+2xf
eBTZrG467d9eR06dIlxowZvvss333333xDu3bt6Natn
qRP+5g59Gz6brOGmWLMKJNTbfLN2rUiNDQUC+/vpr
7rrrLlq1agVAgVAwYIFmThxyQ2axfIO8QJnSHHvP7661SvXh
1wjB740ksbPO8+fP07dtdvXzXz1t2oSIMGLECDp06PCP07dhwIDHdjwoL
5f/48AHPnzuWnn35i+vTpdOvWjeLFi7N8mUuCQN9d161OaKOnt
SEhFC1aFIDKzuaXzuXpXsXzdvCj2bqKRa9evTh/e8evTh8+DAA
q261ateL48eMEBQXXf8MYbb9f9C4YbbbAd9C4cWPwMVV
KYvDgwcTExNCNTglCvnz5KGcPKF++PE8+PEE++SQrV64
jqa4OBghg4dyv79+v79+xkZyAi9evVyez+P4ed5/xkyZAiYkCwcsI

YNGgQV65c4auvviJfvnwsWrSI4sWLExISQq9evbh48SKV
KlXi888/p1ixYmzevJlnn32WggULXnXTR2xsLK+99hqrVq
3i8uXL9O7dm549e7qM5bvvvmP06NGA48rjmWee4cKFC4
Bj0K/GjRuzatUqRo4cSZkyZQgJCWH79u3Jbuf8+fO0a9eO
U6dOER0dzejRo2nXrp3b+yU5CxYsoEuXLogId955J6dPn+
bYsWOUKVMmoUxkZCT58mzP5buWjatWqALRs2ZL33nuP555
7jp07dzJ06FAAqlevTnh4OH///dlypRJWIevry8BAQH8+ee
f1KhRg9tuu43IyEj++usvbr755huKPzVW9QQsX76cwMBA
3nrrLTp06MAdd9zh7ZCMU2xxsLMuXL6dt27aAo9qpXr16
V5WpVKkS58+f5+zZs4SFhV0zfn++bN++ndDQUF
q0aJHqMnv37BWIsmz8/PHHtGvjXjnnnzf5Cq0bo0XAj0G2Ky1UbqP7aNqXj
KlefnllxkwYAAbN27ku+++4/nnn79mPT3+mz4zY4/+QpDGyEmd3P6BeHzhEXbdKlAhGFdbEFpAcIL0mGbHUB9AHGx6hSfm9pOIcWlgGRDU3c9r1fVBcijcwMnonVjNIDvoLBHCgbdMQ0ITadOiDHR9JP8pl9qW7cqb+SeT+9G4ExzaNJqU+AnS77lVt3ZR1oyS/wKn8D31k1YDKqCJF1rO
NMVrmHdHUqPl5TEajDDtHqJQbt+u3p94vVNJyOTdHqRNvUaIWujSGxlc+K6+hVHvpNQdfaHe7+MebLZfdHqpPLQeEa8kbyw+UqZa+0LfKQcpoyg8s+bXXsmoXqFUXDdDa8djS+7kzPfjz8FnTcRj6ZaJKwX7r8kQZ2N8v7LBNmLSGrwb0FK5PyFKmPXzc1j4QQjbRXmGFw47oBVt1kw3lRx60BhtFGwn5V2f+/NrqQu+xXOMVj2d1K2kHoCUugwfDDXuUvmVvXK8uiL0HSTHX6kzxsyGb2tr3aWzH5UbVVVZpuZHFdqEqfeb27OoqQ3lkz0kk99gFQxsFZlCe6gS6Dky6qhJOWc2hx6ELN5XKr5Eml0e5hV1hvBr6bfaHv4VRPvDBhS0K5RyegYIkCk4nzYfNhb6ftNXE7Y5LxFFMQNOHHiM4RwKVKiIb3lLcNr3okPeLMQkb1YU5Li4oQthY6b2whzi6pYN4nRfwnWqFMjE1SJZVmLYF0zU3xLMqVaQvhYBHkwm6WU2bKEEbgSJeiUhiESckTv2bCN6E88vFevzu1Zv3V/Sjv8XDIKd2qFRQAIH/h91MHr09a6q16xvPbNd6ffx6T3+TNqz4PH1b6fJ30c9xXHRjKd1qvKwoWbnHX8Fp5zR9FtVZwaLZrIVbIlJlhnKUhqlnI9KOepS2Z7jkH3XZBWNJ/VQ8E2DcS1J0tLu8+DBzbeWJ3j3Zp+w2fMOCrl7fm1VShX3XRSsFwVi7vTgKEXqY/wBkrsyO/Uvx6a/nfNhPc/VX5qbZbLNQhvlllF23+NbItZxTq2H6pcEs4bLzHOOf5nnkS8RzUk/x1E/5buuYYQGQxK8ZPb6M1+0lT0pUxrgQakpydz8IZI2uSKOCbCRD9C5Sn/dOlGQkGn3YIo8uqZ1VOm9Op/dijtMTsxaNW7+MFnXbYgPGtyMhfg7OlS7j/ug6aS3xBOkVYAiOIJlG3k9F6wcBQM+03+kPAWyj3xgR3fdIi5ldgVILNVqHQS1bMg4UROxOX1gHtX45qmVBNhsD8ec7zq7EUjQy9wkfgnxB5yEHbFkP6I+/IjOkNPU+uCnCN4nhSoMf7Wgt9ZyBV0/KQrOGnuPfOb4MpWXfJ5rc+A//Y/lvpiIJLf6GtBA6UY70lxzHbtWPz6JR5GoZXnNxPSIwhQ6mgQyU1fUnuEIc0yDj+VjOPbCTnBa+ON7ocevVT+Nf5N7s1ilOtWSNt66Ljxz/i5ghCVvA/pKCdlpbBSWg9lXH3xaMXAEcyQGuFUKngNl2wSJHV+Q1hhUHXtDc3xdPK09etDaGtS3NClRHiPDybMVZPf2j8gPz+hZaEs7yhdAKWTN3EJjBodnFGuXwjF2pc+pyvQQBWK2yD4dY5cW3CNcyX1l6o73JrWdUqQxohQNTGM1kzSLcMhTZ3eC7fPuw8l/D3kLvmlEL7Xm/u9sGu+cGd/lzIwHYIIA5fQXJTnYejWvzNE8Dps5VN+q6lgFIJOn/U9+r1tc2whVdsr6qDc8GP7ltd/m83yTNe1ZCldQjPK4TdlvYGhy9z2eCPzYnGnpXBx+NKxVtmhvK96RoUmE9WlrEFqOcDpjq7ZlbOyjDEA1Dde7b28XBjf5X8L3fRsyT+Tvh5bVpb0kJVMnSmu17X5CmH2qDa63a8UfFHxkU/Qj+/wzV6VcQJoV9F/jlVehhtoIjSUHmoJm/yM/tGMw3m1oD2XVkaQ2VcsvBx6tfBEWHP/A/uhZUXsAIujEmRr6u2yDnuFvBpYj3yJvf8XkCiU+7Sy5djd6FnOsuOo17Y7VSU5ZbVLbsrCJJr+nmk/v/C6nrqzGEdFlXSaB1lYoUNXYQCfiRoEFmhM6tL6yi5+BLjDMn1emsbiPqKNFy27vDUYdqZ8KDZ4kJyp8gj8qFqTuVtJVzSsgLAoRpCNaJh1uSW5WSUomZFWOl0a4VJ6BmYpgTdwXBvoAtD7ZQzIp6EefkgwqF53YnS3Cm3y6QCmDRs/AGsVEfB8zvJM2aNxr8Ls4zEhgb1OxyqgSWoSYRD2GU0JfNDzMqfq+H6Xy4RGzSmrTFcPR1YBP+mBpdXJ4+Pj3XRRgVNaRMrLEuJLAWUXT+V/aL6WcmbzM6dA/gQvtHy7fAz48y6DS/H6jFxbnrEMVhnmJi5lxZRzZi4kDGmjH2fBEHwHwdQx4RiTJFzgTQkWVxwRE1UyTWrcQV+FdOk5T8m5g/Uv8gM0tiGqqdjH36p8TDLVKgTmIJTCHYAwK1nCy5kP0M3YSXD+oxKIk+sOnmfglAumZ2hX/Zl3WxaDaZRRQO0Uw9/2ckeIZ4qwr5MJhaJOB4bJpKj6HDqVt3OHXUZgzXkMu4WnFm1Ei+fu8gOG2CPeT7hUfCEXMqR1AHWOeH5bG7jrY/ZkAHA==

Z43LfJJZcokn69ygizJ49mwEDBnD58mVatWpF7tyOQ/Brr
73Gyy+/TFBQELVr16Zu3boJ88BxBdyhQwfGjRtHkSJFEj
4vVaoUR48edTvOtMqRiSIuLg5VxcfHh4YNG/Lmm28ydO
hQ8ufP7+3QjFOBAgUICQnhzJkzPPzww0yaNIl+/fpRs2ZN
Vq9efVXZAwcOULhwYXx9falZsyabN2+mTp06LtefUsJJ/F
nSe9ELFSqU8L5Ro0bs37+fEydOMH/+fIYPHw44/rbWrVt
HgQIFruv7xq87uQNYvFn1bly5Up4Hz8d3zCaWEr70V1J
t5F4+8ltL56rhK6qfPLJJ9x///1XfR4eHp5s+QIFClz1u/j4
pXbo027ZtIy4u7qr/28S/o5S2M336dE6cOMHmzZvJkycP5c
uXT/a5g06dOrFnz55rPh84h84cCBdunS56jN/f/+jRat
mzZa5Zt1KgRa9asARyJLD7JFSlShGnTpiXEXaFChYSEF
x0dTYcOHXjqqad49NFHr1rfpUuXrvvtvLvLS1yXNXTtm3baN
y4MZMmTQLgySefZOTIkZYkMik/Pz8mTJjARx99RHR0
NE899RS//vory5AxxnzP369eOVV14BYMiQIbz77rsJ/4Bx
cXGMHTv2mvW2/alRU1rKVLKGgNKlS7+qmT7EnnNF10/v378fg
K+++irhrPxGJN2P6cXPzz49+cuvvA
MycOTNhmfvvv58E9obkVK1a9aokVJcXQZGBcd4tNcXa5lLZLZz5swZUqRZwZSpR
WHDh1Kdvk5c+YQUYF3xPyu8QhHJcreFG5iMDgNUVjxhrqlLvjGkaIx/Xr8/Pyu
ap+Id/z6yR0DVtOPXM6OZFw3kN8sGxotVKBlQkmmzUoGrcHWoUaL
KFEFVee655wgICG3r2p3k2WHnHDWHnJMojh//jD
Bg2iXr16HDhwwwOONPyb91K1blzp16jB79wwKFCjAggU
GD16NNWqaVaN27do0aNPn36mPAgeSHmzl+dCEhmr6b
gIoFatWhw7duyadQ4fRq1Yt6tSpw1pw8qVKwF4//33
fjhh2nRokWy/+iRkzZiERGYS2ERgymDCBTZs2ERgYS
I0aNfj000+v67vmz5z+fadOm8djhjj1G7du2ExvH0kHg/pqcvvv
iCIUOGEBgYSEhISEJ7ybdbRp0+jduzdfffcSvV/llm0aWjE8iq17bN6NMMgjQfhjf8g
Ggm3ifbs2dPl1UmhQoVKlSQvJ86vaWX+OKLLjzzjv
u3fvVVcRiaW0naeeoNmNzTRv359Zs6cM6cmXDjw4148MEHq

VixIpUrV+aFF15g8uTJV82Lrx768MMPCQgIIDAwkDZt2i
S0je3atYuaNWtSvXp1Fi9enHAb7Nq1a/nqq69YsWJFQhvJ
okWLAMeVxv79+zNkWOUc0c34smXL6N69OxEREfTo0Y
P333/frTrpnGrXrl0EBAR4OwxjEsybN4/Nmzcn3PlkSGiQf/
vtt6+Zl9z/8I10M54j2ijy5s1L8eLFmTNnDo0bN/Z2OMaY69
S+fXsiIyO9HUamEhMTw6BBgzJkW9nyiiI6Oppx48Zx5syZ
hDOQuLg4cuXKMTVtN8SuKIzJ2tL7iiLbHTl/++036tW1rxxy
uvvJLQKAlYkrhOWe0Ewhjj4In/3Wxz9Pzn3/o0aMHd911
F6dPn2b+/Pl89913liDSIH/+/ERGRlqyMCaLUed4FOl9F2e2
aaOIjIzk66+/ZvDgwYwYMLChQt7O6Qsy9/fn4iICE6cOO
HtUIwx1yl+hLv0v0KbPbs2cOcOXMSbseKyMuJ/dGGP
MvzJtG4WItBaRPSKyX0ReS2a+iMgE5/xVGRUX
x5ptvJnTTHP9EpCUJY4xJfx6rehKAKyOA+aKyA+q
ujNRsQeAKs7XHcAU946qmn+M9//k
Pp0qU98yWMMcZ4tI2iuMZ4UH5D7V2O7P9XmMbFY7VRRTZcsBViUJEegA9nJOX9+3bF3bff
yaKcsCRRNMRXHu1kFyZcsBViUJEegA9nJOX9+3bF3bfff
elb7RZU0ngpLeDyCRsX/zL9sW/bF/8q1paF/Rkokiu68ikLef
ulEFVg4FgABHZlNYGmezG9sW/bF/8y/bFv2xf/EtENqV1
WU82ZkcAtySa9geSdpzuThljjDFe5MlEsRGoIIVRCQv0B
n4IUmZH4Auzruf7gTOuGqfMMYk/E8VvWkqjEi0gdYA
vgAn6vqDhHp5Zz/KbAS/0b74l
+2Lf9m++Jfti3+leV9kuQfujDHGZCzrCMkYY4YxLliiMMca
4lGkThae6/8iK3NgXTzn3QaiI/CYidZAR
GJFpGNxpeR3NkXItJMREJEZI/JLRMWYUN/5H/ET
kRxHZ5twX7rSHBeZqaqZ7oaiq2aoaJ4RbDjcdkOmp7iVdkH3YNSigZNfRPrdOpeEjjtTFNwbbxzXdxRAEUwvdnQZdBXTkUD87D4eGY2EWHgYnjh3sRDvrF9
kRxHZ5twX7rSHZjki8rmIHBeRsBTmp+24qaqZ7oaiq2aoaJ4RbDjcdkOmp7iVdkH3YNSigZNfRPrdOpeEjjtTFNwbbxzXdxRAEUwvdnQZdBXTkUD87D4eGY2EWHgYnjh3sRDvrF9

/4uAPoC3wHHMzK4DObOvngS+F5VDwOoanbdH+7sCwV
8RUSAwjgSRUzGhul5qroax3dLSZqOm5k1UaTUtcf1lskOr
vd7PofjjCE7SnVfiEg5oD3waQbG5Q3u/F1UBYqJyCoR2Sw
iXTIsuozlzr6YCATgeKB3O/CyqsZlTHiZSpqOm5l14KJ06/
4jG3D7e4pIcxyJ4m6PRuQ97uyLccCrqhrrOHnMttzZF7mB
esC9QAFgnYisV1XXg85nPe7si/uBEKAFUAlYKiJrVPWsp
4PLZNJ03MysicK6//iXW99TRAKBqcADqhqZQbFlNHf2R
X1gtjNJlAQeFJEYVZ2fMSFmGHf/R06q6gXggoisBuoA2S1
RuLMvugPvq6Oifr+IHASqAxsyJsRMI03Hzcxa9WTdf/wr1
X0hIrcC3wPPZMOzxcRS3ReqWkFVy6tqeWAu8FI2TBLg3
v/IAqCJiOQWkYI4em/elcFxZgR39sVhHFdWiEhpHD2pHs
jQKDOHNB03M+UVhXqu+48sx8198SZQApjsPJOO0WzY
Y6ab+yJHcGdfqOouEfkfEArEAVNVNdnbJrMyN/8u3gami
8h2HNUvr6pqtut+XERmAc2AkiISAYwA8sCNHTetCw9jj
DEuZdaqJ2OMMZmEJQpjjDEuWaIwxhjjkiUKY4wxLlmi
MMYY45IlCpMpOXt+DUn0Ku+i7Pl02N50ETno3NYWEW
mUhnVMFZEazvfDksz77UZjdK4nfr+EOXtDLZpK+SARe
TA9tm1yLrs91mRKInJeVQund1kX65gO/KSqc0WkFfCRq
gbewPpuOKbU1isiXwB7VfUdF+W7AfVVtU96x2JyDruiMF
mCiBQWkeXOs/3tInJNr7EiUkZEVic6427i/LyViKxzLvutiK
R2AF8NVHYuO9C5rjAR6e/8rJCILHSObRAmIp2cn68Skf
oi8j5QwBnHTOe8886fcxKf4TuvZDqIiI+IfCgiG8UxTkBPN
3bLOpwduolIQ3GMRbLV+bOa8ynlUUAnZyydnLF/7tzO1
uT2ozHX8Hb/6fayV3IvIBZHJ24hwDwcvQgUcc4riePJ0vgr4
vPOn4OA153vfQBfZ9nVQCHn568Cbyazvek4x64AHgN+x9
Gh3nagEI6uqXcAdYEOwP8lWtbP+XMVjrP3hJgSlYmPsT
3whfN9Xhw9eRYAegDDnZ/nAzYBFZKJ83yi7/ct0No5XQT
I7Xx/H/Cd8303YGKi5d8Fnna+L4qj36dC3v592ytzvzJlFx7G
AFGqGhQ/ISJ5gHdFpCmO7ijKAaWBvxItsxH43Fl2vqqGi
Mg9QA1grbN7k7w4zsST86GIDAdO4OiF915gnjo61UNEvg

eaAP8DPhKRMTiqq9Zcx/daDEwQkXxAa2C1qkY5q7sC5d
8R+fyAKsDBJMsXEJEQoDywGViaqPwXIlIFR2+geVLYfi
ugrYgMdk7nB24le/YBZdKJJQqTVTyFY2SyeqoaLSLhOA
5yCVR1tTORPAR8JSIfAqeApar6hBvbGKKqc+MnROS+5
Aqp6l4RqYejz5z3RORnVR3lzpdQ1UsisgpHt9edgFnxmwP6
quqSVFYRpapBIuIH/AT0Bibg6Mtopaq2dzb8r0pheQE6qO
oed+I1BqyNwmQdfsBxZ5JoDtyWtICI3OYs83/AZziGhFwP
3CUi8W0OBUWkqpvbXA084lymEI5qozUiUha4qKozgI+c2
0kq2nllk5zZODpja4KjIzucP1+MX0ZEqjq3mSxVPQP0Aw
Y7l/ED/nTO7pao6DkcVXDxlgB9xXl5JSJ1U9qGMfEsUZis
YiZQX0Q24bi62J1MmWZAiIhsxdGOMF5VT+A4cM4SkV
AciaO6OxtU1S042i424GizmKqqW4HawAZnFdDrwOhkFg
8GQuMbs5P4GcfYxsvUMXQnOMYS2QlsEZEw4L+kcsXvj
GUbjm61P8BxdbMWR/tFvJVAjfjGbBxXHnmcsYU5p41xy
W6PNcYY45JdURhjjHHJEoUxxhiXLFEYY4xxyRKFMcY
YlyxRGGOMcckShTHGGJcsURhjjHHp/wE9YM4U7w4yK
wAAAABJRU5ErkJggg==\n",

```
   "text/plain": [
    "<Figure size 432x288 with 1 Axes>"
   ]
  },
  "metadata": {
   "needs_background": "light"
  },
  "output_type": "display_data"
 }
```

```
   ],
   "source": [
    "df2 = df.dropna(axis=0)\n",
    "no_na = df2.index.tolist()\n",
    "some_na = df.drop(no_na).apply(lambda x: pd.to_numeric(x,errors='coerce'))\n",
    "some_na = some_na.fillna(0) # Fill up all Nan by zero.\n",
    "\n",
    "X_test = some_na.iloc[:,:-1]\n",
    "y_test = some_na['class']\n",
    "y_true = y_test\n",
    "lr_pred = clf_best.predict(X_test)\n",
    "print(classification_report(y_true, lr_pred))\n",
    "\n",
    "confusion = confusion_matrix(y_test, lr_pred)\n",
    "print('Confusion Matrix:')\n",
    "print(confusion)\n",
    "\n",
    "print('Accuracy: %3f' % accuracy_score(y_true, lr_pred))\n",
    "# Determine the false positive and true positive rates\n",
```

```
   "fpr,tpr,roc_auc = auc_scorer(clf_best, X_test, y_test, 'RF')\n",
    " \n"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 23,
   "metadata": {},
   "outputs": [],
   "source": [
    "import pickle\n",
    "pickle. dump(clf_best, open('randomclass_chronic', 'wb'))"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "# RandomForest Regressor"
   ]
```

```
 },
 {
  "cell_type": "code",
  "execution_count": 24,
  "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>RandomForestRegressor()</pre>In a

**Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org. <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\">RandomForestRegressor</label><div class=\"sk-toggleable__content\"><pre>RandomForestRegressor()</pre> "**

```
     ],
     "text/plain": [
      "RandomForestRegressor()"
     ]
    },
    "execution_count": 24,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
```

```
   "from sklearn.ensemble import RandomForestRegressor\n",
   "reg=RandomForestRegressor()\n",
   "reg.fit(X_train,y_train)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 25,
  "metadata": {},
  "outputs": [],
  "source": [
   "y_pred=reg.predict(X_test)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 26,
  "metadata": {},
  "outputs": [],
  "source": [
   "pickle. dump(reg, open('randomreg_chronic', 'wb'))"
```

```
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 27,
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "array([0.6 , 0.8 , 1.  , 0.78, 0.79, 0.29, 0.74, 1.  , 1.  , 1.  ,
0.29,\n",
       "       1.  , 1.  , 0.29, 0.09, 0.88, 0.29, 0.09, 0.77, 0.29, 0.29,
0.69,\n",
       "       0.66, 0.29, 0.99, 0.88, 1.  , 0.89, 0.98, 0.89, 0.29, 1.  ,
0.94,\n",
       "       1.  , 0.09, 0.07, 0.89, 1.  , 1.  , 0.07, 0.73, 1.  , 0.29,
0.99,\n",
       "       0.29, 0.01, 0.94, 0.8 , 1.  , 0.29, 0.29, 0.89, 0.89, 0.8 ,
0.89,\n",
       "       0.09, 0.62, 0.99, 0.8 , 1.  , 0.  , 0.  , 0.8 , 1.  , 0.69,
0.89,\n",
       "       0.29, 0.81, 0.29, 0.29, 0.29, 0.69, 0.99, 0.29, 0.87,
0.98, 0.09,\n",
```

" 0.28, 0.69, 0.89, 0.89, 0.29, 0.09, 0.8 , 1. , 0.22, 1. , 0.29,\n",

" 0.2 , 0.29, 0.89, 0.09, 0.29, 0.27, 1. , 0.8 , 0.09, 1. , 0.05,\n",

" 0.8 , 0.09, 0.8 , 0.09, 0.89, 0.73, 0.29, 0.73, 0.29, 0.29, 0.27,\n",

" 0.69, 0.09, 0.29, 0.29, 0.05, 0.29, 1. , 0.88, 1. , 0.09, 0.89,\n",

" 0.29, 0.89, 1. , 0.69, 0.29, 0.69, 0.76, 0.28, 0.09, 1. , 1. ,\n",

" 1. , 0.99, 0.29, 0.71, 0.28, 0.29, 0.01, 0.09, 0.29, 0.05, 0.98,\n",

" 0.29, 0.99, 0.87, 0.82, 0.27, 0.29, 0.93, 1. , 0.99, 0.89, 0.09,\n",

" 0.29, 1. , 0.69, 0.8 , 0.8 , 0.29, 1. , 0.09, 0.89, 0.8 , 1. ,\n",

" 0.29, 0.28, 0.89, 0.29, 0.29, 0.29, 1. , 0.25, 0.82, 0.23, 0.22,\n",

" 0.09, 0.09, 0. , 0.8 , 0.09, 0.89, 0.09, 0.29, 0.09, 0.88, 0.29,\n",

" 0.04, 0.29, 0.28, 0.29, 0.99, 0.29, 0.69, 0.99, 0. , 1. , 0.29,\n",

" 0.97, 0.29, 0.98, 0.69, 0.88, 0.91, 0.95, 0.29, 0.69, 0. , 0.09,\n",

```
       "       0.02, 0.  , 0.  , 0.  , 0.02, 0.  , 0.  , 0.  , 0.01, 0.  , 0.
,\n",
       "       0.  , 0.02, 0.  , 0.01, 0.  , 0.02, 0.  , 0.09, 0.  , 0.09,
0.03,\n",
       "       0.05, 0.  , 0.  , 0.01, 0.  , 0.  , 0.  , 0.  , 0.07, 0.  ,
0.01])"
      ]
     },
     "execution_count": 27,
     "metadata": {},
     "output_type": "execute_result"
    }
   ],
   "source": [
    "y_pred"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 28,
   "metadata": {},
   "outputs": [],
   "source": [
```

```
  "l_pred=list(y_pred)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 29,
 "metadata": {},
 "outputs": [],
 "source": [
  "l_test=list(y_test)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 30,
 "metadata": {},
 "outputs": [],
 "source": [
  "d={'prob':l_pred,'out':y_test}"
 ]
},
{
```

```
 "cell_type": "code",
 "execution_count": 31,
 "metadata": {},
 "outputs": [],
 "source": [
  "df_i=pd.DataFrame(d)\n"
 ]
},
{
 "cell_type": "code",
 "execution_count": 32,
 "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>prob</th>\n",
"      <th>out</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>0.60</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
```

```
"    <tr>\n",
"      <th>1</th>\n",
"      <td>0.80</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>1.00</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>4</th>\n",
"      <td>0.78</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>5</th>\n",
"      <td>0.79</td>\n",
"      <td>1.0</td>\n",
"    </tr>\n",
"  </tbody>\n",
"</table>\n",
```

```
    "</div>"
   ],
   "text/plain": [
    "   prob  out\n",
    "0  0.60  1.0\n",
    "1  0.80  1.0\n",
    "2  1.00  1.0\n",
    "4  0.78  1.0\n",
    "5  0.79  1.0"
   ]
  },
  "execution_count": 32,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "df_i.head()"
]
},
{
 "cell_type": "code",
```

"execution_count": 33,

"metadata": {},

"outputs": [],

"source": [

"df_i.to_csv('C:/Users/Sinegalatha/Desktop/2nd year online class/nalaiya thiran/output/file1.csv')"

]

},

{

"cell_type": "markdown",

"metadata": {},

"source": [

"## Summary of Results\n",

"With proper tuning of parameters using cross-validation in the training set, the Random Forest Classfier achieves an accuracy of 88.8% and an ROC AUC of 99.2%. Lesson learnt: It happens that some pruning helps improve the performance of RF a lot.[](http://)"

]

},

{

"cell_type": "code",

"execution_count": null,

```
   "metadata": {},
   "outputs": [],
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  },
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
```

```
   "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": "3.8.3"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 1
}
```