```
{
"nbformat": 4,
"nbformat_minor": 0,
"metadata": {
"colab": {
"name": "iris_perceptron.ipynb",
"provenance": [],
"collapsed_sections": [],
"authorship_tag": "ABX9TyNm6LuOeKdUz6aJ6DW4AfJr",
"include_colab_link": true
},
"kernelspec": {
"name": "python3",
"display_name": "Python 3"
},
"language_info": {
"name": "python"
}
},
"cells": [
{
"cell_type": "markdown",
"metadata": {
"id": "view-in-github",
"colab_type": "text"
},
"source": [
"<a href=\"https://colab.research.google.com/github/irhallac/SisLab/blob/main/iris_perceptron.ipynb\" target=\"_parent\">
<img src=\"https://colab.research.google.com/assets/colab-badge.svg\" alt=\"Open In Colab\"/>
```

```
</a>"
]
},
{
"cell_type": "code",
"source": [
"# Lojik kapıların Perceptron öğrenme algoritmasıyla modellenmesi\n",
"\n",
"# credits for the Perceptron class: Aashir Javed\n",
"# Available: github.com/aashirjaved\n",
"# Perceptron-Machine-Learning-Using-Python-\n",
"# File: Perceptron.py"
],
"metadata": {
"id": "I7sz4vaJQpvM"
},
"execution_count": 1,
"outputs": []
},
{
"cell_type": "code",
"source": [
"import matplotlib.pyplot as plt\n",
"import numpy as np\n",
"import pandas"
],
"metadata": {
"id": "8MkfMisUP8f6"
},
"execution_count": 2,
"outputs": []
```

},

{

"cell_type": "code",

"source": [

"df = pandas.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data', header=None)\n"

],

"metadata": {

"id": "4_l_FEMbeOf7"

},

"execution_count": 3,

"outputs": []

},

{

"cell_type": "code",

"source": [

"df"

],

"metadata": {

"colab": {

"base_uri": "https://localhost:8080/",

"height": 424

},

"id": "fsuKdFYTX93Y",

"outputId": "bf94cf2c-39c1-4b26-c291-8deb85f971a8"

},

"execution_count": 4,

"outputs": [

{

"output_type": "execute_result",

"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>0</th>\n",
				"      <th>1</th>\n",
				"      <th>2</th>\n",
				"      <th>3</th>\n",
				"      <th>4</th>\n",
				"
```

```
        </tr>\n",
        "
</thead>\n",
"  <tbody>
        \n",
        "   <tr>
            \n",
            "     <th>0</th>\n",
            "     <td>-7.43</td>\n",
            "     <td>-4.17</td>\n",
            "     <td>8.68</td>\n",
            "     <td>0.66</td>\n",
            "     <td>-6.06</td>\n",
            "     <td>-2.18</td>\n",
            "     <td>Iris-setosa</td>\n",
            "
        </tr>\n",
        "   <tr>
            \n",
            "     <th>1</th>\n",
            "     <td>8.39</td>\n",
            "     <td>-9.33</td>\n",
            "     <td>-0.70</td>\n",
            "     <td>1.74</td>\n",
            "     <td>-3.41</td>\n",
            "     <td>-7.33</td>\n",
            "     <td>Iris-setosa</td>\n",
            "
        </tr>\n",
        "   <tr>
            \n",
```

```
            "    <th>2</th>\n",
            "    <td>-5.63</td>\n",
            "    <td>-3.56</td>\n",
            "    <td>9.66</td>\n",
            "    <td>2.73</td>\n",
            "    <td>-6.96</td>\n",
            "    <td>-0.87</td>\n",
            "    <td>Iris-setosa</td>\n",
            "
</tr>\n",
"   <tr>
          \n",
            "    <th>3</th>\n",
            "    <td>-6.33</td>\n",
            "    <td>-3.82</td>\n",
            "    <td>9.58</td>\n",
            "    <td>0.91</td>\n",
            "    <td>-6.95</td>\n",
            "    <td>-1.56</td>\n",
            "    <td>Iris-setosa</td>\n",
            "
</tr>\n",
"   <tr>
          \n",
            "    <th>4</th>\n",
            "    <td>-5.83</td>\n",
            "    <td>-4.26</td>\n",
            "    <td>11.13</td>\n",
            "    <td>0.62</td>\n",
            "    <td>-6.74</td>\n",
            "    <td>-1.74</td>\n",
```

```
                                    "      <td>Iris-setosa</td>\n",
                                    "
                                </tr>\n",



                            </tbody>\n",
                            "
                        </table>\n",
                        "<p>5 rows × 6 columns</p>\n",
                        "
</div>"
],
"text/plain": [
"      0     1    2    3     4     5          4\n",
"0   -7.43  -4.17  8.68  0.66  -6.06  -2.18       Iris-setosa\n",
"1    8.39   -9.33  -0.70  1.74  -3.41  -7.33     Iris-setosa\n",
"2    -5.63  -3.56  9.66  2.73    -6.96  -0.87    Iris-setosa\n",
"3    -6.33  -3.82  9.58   0.91    -6.95  -1.56   Iris-setosa\n",
"4    -5.83  -4.26  11.13  0.62    -6.74  -1.74    Iris-setosa\n",
"\n",
"[5 rows x 6 columns]"
]
},
"metadata": {},
"execution_count": 4
}
]
},
{
"cell_type": "code",
"source": [
```

"cikis = df.iloc[0:100, 4].values\n",

"cikis = np.where(cikis == 'Iris-setosa', -1, 1)\n",

"giris = df.iloc[0:100, [0, 2]].values"

],

"metadata": {

"id": "CWVfaaGYeYbV"

},

"execution_count": 5,

"outputs": []

},

{

"cell_type": "code",

"source": [

"plt.title('2D görünüm', fontsize=16)\n",

"\n",

"plt.scatter(giris[:50, 0], giris[:50, 1], color='black', marker='o', label='setosa')\n",

"plt.scatter(giris[50:100, 0], giris[50:100, -1], color='green', marker='x', label='versicolor')\n",

"plt.xlabel('sapel length')\n",

"plt.ylabel('petal length')\n",

"plt.legend(loc='upper left')\n",

"\n",

"plt.show()"

],

"metadata": {

"colab": {

"base_uri": "https://localhost:8080/",

"height": 297

},

"id": "0BmGhWo5etKL",

"outputId": "e64bfa46-6ccc-4742-d5f9-26e707fc4d2e"

},

```
"execution_count": 6,
"outputs": [
{
"text/plain": [
"<Figure size="" 432x288="" with="" 1="" Axes="">
            "
            ]
            },
            "metadata": {
            "needs_background": "light"
            }
            }
            ]
            },
            {
            "cell_type": "code",
            "source": [
            "class Perceptron(object): \n",
            "    def __init__(self, ogrenme_orani=0.1, iter_sayisi=10):\n",
            "        self.ogrenme_orani = ogrenme_orani\n",
            "        self.iter_sayisi = iter_sayisi\n",
            "\n",
            "    def ogren(self, X, y):\n",
            "        self.w = np.zeros(1 + X.shape[1])\n",
            "        #self.w = np.random.rand((1 + X.shape[1])) * 2\n",
            "        self.hatalar = []\n",
            "        for _ in range(self.iter_sayisi):\n",
            "            hata = 0\n",
            "            for xi, hedef in zip(X, y):\n",
            "                degisim = self.ogrenme_orani * (hedef - self.tahmin(xi))\n",
            "                self.w[1:] += degisim * xi\n",
```

```
"                self.w[0] += degisim\n",
"                hata += int(degisim != 0.0)\n",
"            self.hatalar.append(hata)\n",
"        return self\n",
"\n",
"    def net_input(self, X):\n",
"        return np.dot(X, self.w[1:]) + self.w[0]\n",
"\n",
"    def tahmin(self, x):\n",
"        return np.where(self.net_input(x) >= 0.0, 1, -1)"
],
"metadata": {
"id": "N8F1p5oLsYgw"
},
"execution_count": 7,
"outputs": []
},
{
"cell_type": "code",
"source": [
"siniflandirici = Perceptron(ogrenme_orani=0.1, iter_sayisi=10)"
],
"metadata": {
"id": "CZa2cksXUiUm"
},
"execution_count": 8,
"outputs": []
},
{
"cell_type": "code",
"source": [
```

```
"siniflandirici.ogren(giris, cikis)"
],
"metadata": {
"colab": {
"base_uri": "https://localhost:8080/"
},
"id": "-aa08KLVeMGS",
"outputId": "a78b1516-22d1-41f1-95e6-dd62673ea15f"
},
"execution_count": 9,
"outputs": [
{
"output_type": "execute_result",
"data": {
"text/plain": [
"<__main__.Perceptron at="" 0x7f4dcf91c290="">
            "
            ]
            },
            "metadata": {},
            "execution_count": 9
            }
            ]
            },
            {
            "cell_type": "code",
            "source": [
            "siniflandirici.w\n",
            "    "
            ],
            "metadata": {
```

```
"colab": {
"base_uri": "https://localhost:8080/"
},
"id": "meG_94siYJOT",
"outputId": "bbcdd369-5b14-43b8-fddf-380993355b35"
},
"execution_count": 10,
"outputs": [
{
"output_type": "execute_result",
"data": {
"text/plain": [
"array([-0.4 , -0.68,  1.82])"
]
},
"metadata": {},
"execution_count": 10
}
]
},
{
"cell_type": "code",
"source": [
"siniflandirici.hatalar"
],
"metadata": {
"colab": {
"base_uri": "https://localhost:8080/"
},
"id": "as2BEjs-YKx5",
"outputId": "32aa1cd7-fd51-46fb-d86d-fef12b44eb20"
```

},

"execution_count": 11,

"outputs": [

{

"output_type": "execute_result",

"data": {

"text/plain": [

"[2, 2, 3, 2, 1, 0, 0, 0, 0, 0]"

]

},

"metadata": {},

"execution_count": 11

}

]

},

{

"cell_type": "code",

"source": [

"plt.plot(range(1, len(siniflandirici.hatalar) + 1), siniflandirici.hatalar)\n",

"plt.xlabel('Deneme')\n",

"plt.ylabel('Hatalı tahmin sayısı')\n",

"plt.show()"

],

"metadata": {

"colab": {

"base_uri": "https://localhost:8080/",

"height": 279

},

"id": "R_4EIZEVXJpT",

"outputId": "cc18ce6d-5263-4b97-a836-4e74d20a0a66"

},

```
                    "execution_count": 12,
                    "outputs": [
                    {
                    "output_type": "display_data",
                    "data": {
                    "image/png": "iVBORw0KGgoAAA\n",
                    "text/plain": [
                    "<Figure size="" 432x288="" with="" 1="" Axes="">"
          ]
         },
         "metadata": {
           "needs_background": "light"
         }
        }
      ]
    },
    {
      "cell_type": "code",
      "source": [
        ""
      ],
      "metadata": {
        "id": "L4Dst_4mYGNH"
      },
      "execution_count": null,
      "outputs": []
    }
  ]
}
```