Aplikasi Prediksi Banckrupty Berbasis Jaringan Syaraf Tiruan

oleh:
Alamsyah
Budi Prasetiyo
UNIVERSITAS NEGERI SEMARANG

2019

Source Code

1. **Halaman Administrator**
   Terdapat 181 *line code*

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1.0, user-scalable=no">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <meta name="msapplication-tap-highlight" content="no">
 <meta name="description" content="">
 <meta name="keywords" content="">
 <title>{% block title %}{% endblock %}</title>
 <!-- Favicons-->
 <link rel="icon" href="{% static "storage/images/logo/favicon.ico" %}" sizes="32x32">
 <!-- Favicons-->
 <link rel="apple-touch-icon-precomposed" href="{% static "storage/images/logo/favicon.ico" %}">
 <!-- For iPhone -->
 <meta name="msapplication-TileColor" content="#00bcd4">
 <meta name="msapplication-TileImage" content="{% static "storage/images/logo/favicon.ico" %}">
 <!-- For Windows Phone -->
 <!-- CORE CSS-->
 <link href="{% static "css/materialize.min.css" %}" type="text/css" rel="stylesheet" media="screen,projection">
 <link href="{% static "css/style.min.css" %}" type="text/css" rel="stylesheet" media="screen,projection">
 <!-- Custome CSS-->
 <link href="{% static "css/custom/custom.min.css" %}" type="text/css" rel="stylesheet" media="screen,projection">
 <!-- INCLUDED PLUGIN CSS ON THIS PAGE -->
 <link href="{% static "js/plugins/prism/prism.css" %}" type="text/css" rel="stylesheet" media="screen,projection">
 <link href="{% static "js/plugins/perfect-scrollbar/perfect-scrollbar.css" %}" type="text/css" rel="stylesheet" media="screen,projection">
 <link href="{% static "js/plugins/data-tables/css/jquery.dataTables.min.css" %}" type="text/css" rel="stylesheet" media="screen,projection">
 <!-- <link href="{% static "js/plugins/chartist-js/chartist.min.css" %}" type="text/css" rel="stylesheet" media="screen,projection"> -->
 <link href="{% static "js/plugins/dropify/css/dropify.css" %}" type="text/css" rel="stylesheet" media="screen,projection">
 {% block css %}{% endblock %}
</head>
<body>
 <!-- ////////////////////////////////////////////////////////////////////////////// //// -->
{% block preloader %}{% endblock %}
 <!-- START HEADER -->
 <header id="header" class="page-topbar">
 <!-- start header nav-->
 <div class="navbar-fixed">
 <nav class="navbar-color">
 <div class="nav-wrapper">
 <ul class="left">
 <!-- <li><h1 class="logo-wrapper"><a href="{% url 'home:index' %}" class="brand-logo darken-1" style="margin:3px 0"><img src="{% static
```

```
    "images/logo/header.png" %}" alt="materialize logo"></a> <span
    class="logo-text"></span></h1></li> -->
44.  <a href="#" data-activates="slide-out" class="button-collapse"><i
    class="mdi-navigation-menu"></i></a>
45.  </ul>
46.  <ul class="right hide-on-med-and-down">
47.
48.  <li><a href="javascript:void(0);" class="waves-effect waves-block
    waves-light toggle-fullscreen"><i class="mdi-action-settings-
    overscan"></i></a>
49.  </li>
50.  </ul>
51.  <!-- translation-button -->
52.  </div>
53.  </nav>
54.  </div>
55.  <!-- end header nav-->
56.  </header>
57.  <!-- END HEADER -->
58.
59.  <!-- START MAIN -->
60.  <div id="main">
61.  <!-- START WRAPPER -->
62.  <div class="wrapper">
63.
64.  <!-- START LEFT SIDEBAR NAV-->
65.  <aside id="left-sidebar-nav">
66.  <ul id="slide-out" class="side-nav fixed leftside-navigation">
67.  <li class="user-details cyan darken-2">
68.  <div class="row">
69.  <div class="col col s4 m4 l4">
70.  <img src="{% static "images/avatar/avatar.png" %}" alt="" class="circle
    responsive-img valign profile-image">
71.  </div>
72.  <div class="col col s8 m8 l8">
73.  <ul id="profile-dropdown" class="dropdown-content">
74.  <li><a href="#"><i class="mdi-action-face-unlock"></i> Profile</a>
75.  </li>
76.  <li class="divider"></li>
77.  <li>
78.  <a href="{% url 'logout' %}"
79.  ><i class="mdi-hardware-keyboard-tab"></i>Logout</a>
80.  </li>
81.  </ul>
82.  <a class="btn-flat dropdown-button waves-effect waves-light white-text
    profile-btn" href="#" data-activates="profile-dropdown">Fadli Dony<i
    class="mdi-navigation-arrow-drop-down right"></i></a>
83.  <p class="user-roal">Administrator</p>
84.  </div>
85.  </div>
86.  </li>
87.  <li class="li-hover"><div class="divider"></div></li>
88.  <li class="li-hover"><p class="ultra-small margin more-
    text">Administrator</p></li>
89.  <li class="bold {% block dashboard %}{% endblock %}"><a
    href="/administrator/" class="waves-effect waves-cyan"><i class="mdi-
    action-dashboard"></i> Dashboard</a>
90.  </li>
91.  <li class="no-padding">
92.  <ul class="collapsible collapsible-accordion">
93.  <li class="bold {% block jst %}{% endblock %}"><a class="collapsible-
    header waves-effect waves-cyan bold {% block jst2 %}{% endblock %}"><i
    class="mdi-device-now-widgets"></i> RNN</a>
94.  <div class="collapsible-body" style="{% block collapsjst %}{% endblock
    %}">
95.  <ul>
96.  <li class="{% block data %}{% endblock %}"><a
    href="/administrator/dataset/">Data</a>
```

```
97.  </li>
98.  <li class="{% block latih %}{% endblock %}"><a href="{% url
     'administrator:latih' %}">Latih</a>
99.  </li>
100.    </ul>
101.    </div>
102.    </li>
103.    <!-- <li class="bold {% block akun %}{% endblock %}"><a href="#"
     class="waves-effect waves-cyan"><i class="mdi-action-dashboard"></i>
     Akun</a>
104.    </li> -->
105.    <!-- <li class="bold {% block riwayat %}{% endblock %}"><a href="#"
     class="waves-effect waves-cyan"><i class="mdi-action-dashboard"></i>
     Riwayat Perhitungan</a>
106.    </li> -->
107.    <li class="li-hover"><div class="divider"></div></li>
108.    <li class="li-hover"><p class="ultra-small margin more-
     text">SETTING</p></li>
109.    <!-- <li class="bold {% block umum %}{% endblock %}"><a href="#"
     class="waves-effect waves-cyan"><i class="mdi-action-dashboard"></i>
     Umum</a>
110.    </li> -->
111.    <li class="no-padding">
112.    <ul class="collapsible collapsible-accordion">
113.    <li class="bold {% block konten %}{% endblock %}"><a
     class="collapsible-header waves-effect waves-cyan bold {% block konten2
     %}{% endblock %}"><i class="mdi-action-assignment"></i> Konten</a>
114.    <div class="collapsible-body" style="{% block collapskonten %}{%
     endblock %}">
115.    <ul>
116.     <li class="{% block home %}{% endblock %}"><a href="{% url
     'administrator:home' %}">Home</a>
117.     </li>
118.     <li class="{% block petunjuk %}{% endblock %}"><a href="{% url
     'administrator:petunjuk' %}">Petunjuk</a>
119.     </li>
120.     <li class="{% block kontak %}{% endblock %}"><a href="{% url
     'administrator:kontak' %}">Kontak</a>
121.     </li>
122.    </ul>
123.    </div>
124.    </li>
125.    </ul>
126.    </li>
127.    <li class="bold {% block slider %}{% endblock %}"><a href="{% url
     'administrator:slider' %}" class="waves-effect waves-cyan"><i
     class="mdi-action-dashboard"></i> Slider</a>
128.    </li>
129.    <br>
130.    <br>
131.    <br>
132.    </aside>
133.    <!-- END LEFT SIDEBAR NAV-->
134.    <!--
     ////////////////////////////////////////////////////////////////////////
     //// -->
135.   {% block content %}{% endblock %}
136.    <!--
     ////////////////////////////////////////////////////////////////////////
     //// -->
137.    </div>
138.    <!-- END WRAPPER -->
139.    </div>
140.    <!-- END MAIN -->
141.    <!--
     ////////////////////////////////////////////////////////////////////////
     //// -->
142.    <!-- START FOOTER -->
```

```
143.    <footer class="page-footer">
144.    <div class="footer-copyright">
145.    <div class="container">
146.    <span>Copyright © <?php echo date("Y"); ?> <a class="grey-text text-
   lighten-4">DIAGNOSIS BANKRUPTCY All rights reserved.</span>
147.    <span class="right"> Developed by Fadli Dony Pradana</span>
148.    </div>
149.    </div>
150.    </footer>
151.    <!-- END FOOTER -->
152.
153.    <!-- ================================================
154.    Scripts
155.    ================================================ -->
156.    <!-- jQuery Library -->
157.    <script type="text/javascript" src="{% static "js/plugins/jquery-
   1.11.2.min.js" %}"></script>
158.    <!-- <script type="text/javascript" src="{% static "js/jquery.js"
   %}"></script> -->
159.    <!--materialize js-->
160.    <script type="text/javascript" src="{% static
   "js/materialize.min.js" %}"></script>
161.    <!-- END CONTENT -->
162.    <script type="text/javascript" src="{% static
   "js/plugins/dropify/js/dropify.js" %}"></script>
163.    <!--prism-->
164.    <script type="text/javascript" src="{% static
   "js/plugins/prism/prism.js" %}"></script>
165.    <!--scrollbar-->
166.    <script type="text/javascript" src="{% static "js/plugins/perfect-
   scrollbar/perfect-scrollbar.min.js" %}"></script>
167.    <script type="text/javascript" src="{% static "js/plugins/data-
   tables/js/jquery.dataTables.min.js" %}"></script>
168.    <script type="text/javascript" src="{% static "js/plugins/data-
   tables/data-tables-script.js" %}"></script>
169.    <!-- chartist -->
170.    <script type="text/javascript" src="{% static
   "js/plugins/chartjs/chart-script.js" %}"></script>
171.    <script type="text/javascript" src="{% static
   "js/plugins/chartjs/util.js" %}"></script>
172.    <!--plugins.js - Some Specific JS codes for Plugin Settings-->
173.    <script type="text/javascript" src="{% static "js/plugins.min.js"
   %}"></script>
174.    <!--custom-script.js - Add your own theme custom JS-->
175.   {% block js %}{% endblock %}
176.    <script type="text/javascript" src="{% static "js/custom-script.js"
   %}"></script>
177.    <script type="text/javascript">
178.    $(".button-collapse").sideNav();
179.    </script>
180.   </body>
181.   </html>
```

2. Proses Pembentukan Jaringan Saraf Tiruan

Terdapat 218 *line code*

```
1. def rnn(dataset_latih, dataset_uji, alfa_masuk, hidden_masuk,
   epoch_masuk, mse_masuk, momentum_masuk):
2.
3.     # Fungsi aktivasi Sigmoid Biner
4.     def sigmoid_biner(x):
5.         output = 1/(1+np.exp(-x))
```

```
6.          return output
7.
8.      # Fungsi aktivasi Sigmoid Biner
9.      def sigmoid_bipolar(x):
10.         output = 2/(1+np.exp(-x)) -1
11.         return output
12.
13.     def
    momentum(momentum,sinapsis_2,sinapsis_delta,sinapsis_1,sinapsis_0):
14.           beda = sinapsis_1-sinapsis_0
15.           output = sinapsis_2+sinapsis_delta+(momentum*beda)
16.           return output
17.
18.     # Turunan fungsi aktivasi Sigmoid Biner
19.     def turunan_sigmoid_biner(output):
20.         return output*(1-output)
21.
22.     def turunan_sigmoid_bipolar(output):
23.         return (1+output)*(1-output)/2
24.
25.     def perubahan_w(alfa, omega,x):
26.           return x*alfa*omega
27.
28.     def faktor_kesalahan_unit(omega):
29.           kesalahan = np.zeros(hidden_dim)
30.           for x in range(hidden_dim):
31.                 kesalahan[x] = sinapsis_1[x]*omega
32.           return kesalahan
33.
34.     def net_faktor_kesalahan_unit(Zfnet_total, faktor_kesalahan):
35.           kesalahan = np.zeros(hidden_dim)
36.           for x in range(hidden_dim):
37.                 kesalahan[x] =
    faktor_kesalahan[x]*turunan_sigmoid_biner(Zfnet_total[x])
38.           return kesalahan
39.
40.     dataset = dataset_latih
41.     alfa = float(alfa_masuk)
42.     input_dim = len(dataset[0])-1
43.     hidden_dim = hidden_masuk
44.     output_dim = 1
45.     epoch = epoch_masuk
46.     mse = float(mse_masuk)
47.     aktivasi = 1
48.     log_mse = []
49.     log_ynet = []
50.     momentum_dim = float(momentum_masuk)
51.
52.
53.     # membuat neuron
54.     sinapsis_0 = np.random.uniform(low=-
    2,high=2,size=(input_dim+1,hidden_dim))
55.     sinapsis_1 = np.random.uniform(low=-
    2,high=2,size=(hidden_dim+1,output_dim))
56.     sinapsis_h = np.random.uniform(low=-
    2,high=2,size=(hidden_dim,hidden_dim))
57.
58.     sinapsis_0_update = np.zeros_like(sinapsis_0)
59.     sinapsis_1_update = np.zeros_like(sinapsis_1)
60.     sinapsis_h_update = np.zeros_like(sinapsis_h)
61.
62.     sinapsis_0_0 = np.zeros_like(sinapsis_0)
63.     sinapsis_1_0 = np.zeros_like(sinapsis_1)
64.     sinapsis_h_0 = np.zeros_like(sinapsis_h)
65.
66.     sinapsis_0_1 = np.zeros_like(sinapsis_0)
67.     sinapsis_1_1 = np.zeros_like(sinapsis_1)
68.     sinapsis_h_1 = np.zeros_like(sinapsis_h)
```

```
69.
70.     i =0
71.     ulang=1
72.     while ulang==1:
73.           n=0
74.           error = 0.00
75.           for x in range(len(dataset)):
76.                 masukkan = np.zeros(input_dim+1)
77.                 for data in range(input_dim):
78.                       masukkan[data] = dataset[x][data]
79.                 masukkan[input_dim] = 1
80.                 target = dataset[x][input_dim]
81.                 Znet = np.dot(masukkan,sinapsis_0)
82.                 if aktivasi ==1:
83.                       Zfnet = sigmoid_biner(Znet)
84.                 else:
85.                       Zfnet = sigmoid_bipolar(Znet)
86.                 Unet = np.dot(Zfnet,sinapsis_h)
87.                 Zfnet_total = np.zeros_like(sinapsis_1)
88.                 if aktivasi ==1:
89.                       Zfnet_total = sigmoid_biner(Znet+Unet)
90.                 else:
91.                       Zfnet_total = sigmoid_bipolar(Znet+Unet)
92.                 temp_Zfnet_total = Zfnet_total
93.                 Zfnet_total = np.zeros(hidden_dim+1)
94.                 for temp in range(hidden_dim):
95.                       Zfnet_total[temp] = temp_Zfnet_total[temp]
96.                 Y_net = np.dot(Zfnet_total,sinapsis_1)
97.                 if aktivasi ==1:
98.                       Y_fnet = sigmoid_biner(Y_net)
99.                       omega = (target-
    Y_fnet)*turunan_sigmoid_biner(Y_fnet)
100.                else:
101.                      Y_fnet = sigmoid_bipolar(Y_net)
102.                      omega = (target-
    Y_fnet)*turunan_sigmoid_bipolar(Y_fnet)
103.
104.
105.                sinapsis_0_0 = sinapsis_0_1
106.                sinapsis_1_0 = sinapsis_1_1
107.                sinapsis_h_0 = sinapsis_h_1
108.
109.                sinapsis_0_1 = sinapsis_0
110.                sinapsis_1_1 = sinapsis_1
111.                sinapsis_h_1 = sinapsis_h
112.
113.                sinapsis_1_update = perubahan_w(alfa,omega,Zfnet_total)
114.                sinapsis_1_update_temp = sinapsis_1_update
115.                sinapsis_1_update = np.zeros_like(sinapsis_1)
116.
117.                for temp in range(hidden_dim+1):
118.                      sinapsis_1_update[temp] =
    sinapsis_1_update_temp[temp]
119.                      sinapsis_1[temp] =
    momentum(momentum_dim,sinapsis_1[temp],sinapsis_1_update[temp],sinapsis_
    1_1[temp],sinapsis_1_0[temp])
120.                kesalahan_unit =
    net_faktor_kesalahan_unit(Zfnet_total,faktor_kesalahan_unit(omega))
121.                for temp in range(input_dim+1):
122.                      sinapsis_0_update[temp] =
    perubahan_w(alfa,masukkan[temp],kesalahan_unit)
123.                      sinapsis_0[temp] =
    momentum(momentum_dim,sinapsis_0[temp],sinapsis_0_update[temp],sinapsis_
    0_1[temp],sinapsis_0_0[temp])
124.                for temp in range(hidden_dim):
125.                      sinapsis_h_update[temp] =
    perubahan_w(alfa,Zfnet[temp],kesalahan_unit)
```

```
                        sinapsis_h[temp] = momentum(momentum_dim,sinapsis_h[temp],sinapsis_h_update[temp],sinapsis_h_1[temp],sinapsis_h_0[temp])

                error += (target-Y_fnet)**2
                n+=1

            # print ("error mse: "+str(error))
            error = error/n
            log_mse.append(error)

            i+=1
            if error<mse:
                  ulang=0
            if i>epoch-1:
                  ulang=0
      temp = ""
      for x in range(len(sinapsis_0)):
          for y in range(len(sinapsis_0[x])):
              if y == hidden_dim-1:
                  temp += str(sinapsis_0[x][y])+"\n"
              else:
                  temp += str(sinapsis_0[x][y])+","

      if default_storage.exists('setting/sinapsis_0_temp.txt'):
          default_storage.delete('setting/sinapsis_0_temp.txt')
      path = default_storage.save('setting/sinapsis_0_temp.txt', ContentFile(temp))

      temp = ""
      for x in range(len(sinapsis_h)):
          for y in range(len(sinapsis_h[x])):
              if y == hidden_dim-1:
                  temp += str(sinapsis_h[x][y])+"\n"
              else:
                  temp += str(sinapsis_h[x][y])+","

      if default_storage.exists('setting/sinapsis_h_temp.txt'):
          default_storage.delete('setting/sinapsis_h_temp.txt')
      path = default_storage.save('setting/sinapsis_h_temp.txt', ContentFile(temp))

      temp = ""
      for x in range(len(sinapsis_1)):
          for y in range(len(sinapsis_1[x])):
                  temp += str(sinapsis_1[x][y])+"\n"
      if default_storage.exists('setting/sinapsis_1_temp.txt'):
          default_storage.delete('setting/sinapsis_1_temp.txt')
      path = default_storage.save('setting/sinapsis_1_temp.txt', ContentFile(temp))

      dataset = dataset_uji

      total_benar=0
      for x in range(len(dataset)):
                masukkan = np.zeros(input_dim+1)
               # print ("masukkan : "+str(masukkan))

                for data in range(input_dim):
                      masukkan[data] = dataset[x][data]

                masukkan[input_dim] = 1;
                target_data = dataset[x][input_dim]
                Znet = np.dot(masukkan,sinapsis_0)
                if aktivasi ==1:
                      Zfnet = sigmoid_biner(Znet)
                else:
                      Zfnet = sigmoid_bipolar(Znet)
```

```
189.                Unet = np.dot(Zfnet,sinapsis_h)
190.                Zfnet_total = np.zeros_like(sinapsis_1)
191.                if aktivasi== 1:
192.                      Zfnet_total = sigmoid_biner(Znet+Unet)
193.                else:
194.                      Zfnet_total = sigmoid_bipolar(Znet+Unet)
195.                temp_Zfnet_total = Zfnet_total
196.                Zfnet_total = np.zeros(hidden_dim+1)
197.                for temp in range(hidden_dim):
198.                      Zfnet_total[temp] = temp_Zfnet_total[temp]
199.                Zfnet_total[hidden_dim] = 1
200.                Y_net = np.dot(Zfnet_total,sinapsis_1)
201.                if aktivasi ==1:
202.                      Y_fnet = sigmoid_biner(Y_net)
203.                      omega = (target-
    Y_fnet)*turunan_sigmoid_biner(Y_fnet)
204.                else:
205.                      Y_fnet = sigmoid_bipolar(Y_net)
206.                      omega = (target-
    Y_fnet)*turunan_sigmoid_bipolar(Y_fnet)
207.                # print ("y("+str(x)+"): "+str(Y_fnet)+"-
    >Target:"+str(target_data))
208.                log_ynet.append("y("+str(x)+"): "+str(Y_fnet)+"-
    >Target:"+str(target_data))
209.                if Y_fnet>0.5:
210.                      Y_fnet=1.0
211.                else:
212.                      Y_fnet=0.0
213.
214.                if Y_fnet==target_data:
215.                      total_benar+=1
216.
217.          data = {'mse':log_mse,'ynet':log_ynet, 'benar':total_benar,
    'akurasi':str(total_benar/len(dataset)*100)+"%"}
218.          return data
```

3. Kode program utama python core.py

Terdapat 598 *line code*

```
1.  import numpy as np
2.  import statistics
3.  from django.core.files.storage import default_storage
4.  from django.core.files.base import ContentFile
5.
6.  np.random.seed(0)
7.
8.  def deletenull(temp):
9.      value = temp
10.     for x in range(len(temp)):
11.         value[x] = list(filter(("?").__ne__, temp[x]))
12.     return value
13.
14. def changenulltomode(dataset,temp):
15.     value = dataset
16.     n = len(temp)-1
17.     m = len(dataset[0])-1
18.     if default_storage.exists('data/modus.txt'):
19.         default_storage.delete('data/modus.txt')
20.     modus =""
21.     for x in range(n):
22.         modus+=str(max(set(temp[x]), key=temp[x].count))+"\n"
23.
24.     path = default_storage.save('data/modus.txt', ContentFile(modus))
25.
```

```
    for x in range(len(dataset)):
        for y in range(len(dataset[x])):
            if value[x][y]=="?":
                value[x][y]=max(set(temp[y]), key=temp[y].count)
    return value


def normalizedAttribute(dataset, temp):
    value = dataset
    n = len(temp)-1
    m = len(dataset[0])-1
    attrange = [0 for x in range(n)]
    attbase = [0 for x in range(n)]

    if default_storage.exists('data/attrange.txt'):
        default_storage.delete('data/attrange.txt')
    if default_storage.exists('data/attbase.txt'):
        default_storage.delete('data/attbase.txt')

    attrange_temp =""
    attbase_temp = ""

    for x in range(n):
        attrange[x] = float((max(temp[x]) - min(temp[x]))/2)
        attrange_temp += str(attrange[x])+"\n"
        attbase[x] = float((max(temp[x]) + min(temp[x]))/2)
        attbase_temp += str(attbase[x])+"\n"

    path = default_storage.save('data/attrange.txt',
ContentFile(attrange_temp))
    path = default_storage.save('data/attbase.txt',
ContentFile(attbase_temp))


    for x in range(len(dataset)):
        for y in range(m):
            if value[x][y]!="?":
                if attrange[y]!=float(0):
                    value[x][y] = (value[x][y]-attbase[y])/attrange[y]
                else:
                    value[x][y] = (value[x][y]-attbase[y])
            else:
                value[x][y]="?"
    return value

def changenulltozero(dataset):
    value = dataset
    for x in range(len(dataset)):
        for y in range(len(dataset[x])):
            if value[x][y]=="?":
                value[x][y]=0
    return value

def readdata(data):
    with open(data) as f:
        content = f.readlines()
        for x in range(len(content)):
            content[x] = content[x].replace("\n","")
            content[x] = content[x].replace("\t","")

        for x in range(len(content)):
            string2 = content[x].split(",")

        dataset = [[0 for x in range(19)] for y in range(len(content))]
        temp = [[0 for x in range(len(content))] for y in range(19)]
        for x in range(len(content)):
            string2 = content[x].split(",")
            for y in range(19):
```

```
                if y==0 and string2[0]=="P":
                    dataset[x][y] = 1
                    temp[y][x] = 1
                elif y==1 and string2[0]=="A":
                    dataset[x][y] = 1
                    temp[y][x] = 1
                elif y==2 and string2[0]=="N":
                    dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y>=0 and y<=2 :
                        dataset[x][y] = 0
                        temp[y][x] = 0

                    if y==3 and string2[1]=="P":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==4 and string2[1]=="A":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==5 and string2[1]=="N":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y>=3 and y<=5 :
                        dataset[x][y] = 0
                        temp[y][x] = 0

                    if y==6 and string2[2]=="P":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==7 and string2[2]=="A":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==8 and string2[2]=="N":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y>=6 and y<=8 :
                        dataset[x][y] = 0
                        temp[y][x] = 0

                    if y==9 and string2[3]=="P":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==10 and string2[3]=="A":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==11 and string2[3]=="N":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y>=9 and y<=11 :
                        dataset[x][y] = 0
                        temp[y][x] = 0

                    if y==12 and string2[4]=="P":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==13 and string2[4]=="A":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==14 and string2[4]=="N":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y>=12 and y<=14 :
                        dataset[x][y] = 0
                        temp[y][x] = 0

                    if y==15 and string2[5]=="P":
                        dataset[x][y] = 1
                        temp[y][x] = 1
```

```
                    elif y==16 and string2[5]=="A":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==17 and string2[5]=="N":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y>=15 and y<=17 :
                        dataset[x][y] = 0
                        temp[y][x] = 0

                    if y==18 and string2[6]=="B":
                        dataset[x][y] = 1
                        temp[y][x] = 1
                    elif y==18 and string2[6]=="NB":
                        dataset[x][y] = 0
                        temp[y][x] = 0

        if default_storage.exists('data/nominaltobinary.txt'):
            default_storage.delete('data/nominaltobinary.txt')
        data_temp = ""
        for x in range(len(dataset)):
            for y in range(len(dataset[x])):
                if y == 17:
                    data_temp += str(dataset[x][y])+"\n"
                else:
                    data_temp += str(dataset[x][y])+","
        path = default_storage.save('data/nominaltobinary.txt',
    ContentFile(data_temp))
        temp = deletenull(temp)
        dataset = changenulltomode(dataset,temp)
        dataset = normalizedAttribute(dataset, temp)
        data_temp = ""
        for x in range(len(dataset)):
            for y in range(len(dataset[x])):
                if y == 17:
                    data_temp += str(dataset[x][y])+"\n"
                else:
                    data_temp += str(dataset[x][y])+","
        if default_storage.exists('data/datanormalisasi.txt'):
            default_storage.delete('data/datanormalisasi.txt')
        path = default_storage.save('data/datanormalisasi.txt',
    ContentFile(data_temp))
        # dataset = changenulltozero(dataset)
        return dataset



    def rnn(dataset_latih, dataset_uji, alfa_masuk, hidden_masuk,
    epoch_masuk, mse_masuk, momentum_masuk):

        # Fungsi aktivasi Sigmoid Biner
        def sigmoid_biner(x):
            output = 1/(1+np.exp(-x))
            return output

        # Fungsi aktivasi Sigmoid Biner
        def sigmoid_bipolar(x):
            output = 2/(1+np.exp(-x)) -1
            return output

        def
    momentum(momentum,sinapsis_2,sinapsis_delta,sinapsis_1,sinapsis_0):
            beda = sinapsis_1-sinapsis_0
            output = sinapsis_2+sinapsis_delta+(momentum*beda)
            return output

        # Turunan fungsi aktivasi Sigmoid Biner
        def turunan_sigmoid_biner(output):
```

```
224.            return output*(1-output)
225.
226.        def turunan_sigmoid_bipolar(output):
227.            return (1+output)*(1-output)/2
228.
229.        def perubahan_w(alfa, omega,x):
230.           return x*alfa*omega
231.
232.        def faktor_kesalahan_unit(omega):
233.           kesalahan = np.zeros(hidden_dim)
234.           for x in range(hidden_dim):
235.                  kesalahan[x] = sinapsis_1[x]*omega
236.           return kesalahan
237.
238.        def net_faktor_kesalahan_unit(Zfnet_total, faktor_kesalahan):
239.           kesalahan = np.zeros(hidden_dim)
240.           for x in range(hidden_dim):
241.                  kesalahan[x] =
   faktor_kesalahan[x]*turunan_sigmoid_biner(Zfnet_total[x])
242.           return kesalahan
243.
244.        dataset = dataset_latih
245.        alfa = float(alfa_masuk)
246.        input_dim = len(dataset[0])-1
247.        hidden_dim = hidden_masuk
248.        output_dim = 1
249.        epoch = epoch_masuk
250.        mse = float(mse_masuk)
251.        aktivasi = 1
252.        log_mse = []
253.        log_ynet = []
254.        momentum_dim = float(momentum_masuk)
255.
256.
257.        # membuat neuron
258.        sinapsis_0 = np.random.uniform(low=-
   2,high=2,size=(input_dim+1,hidden_dim))
259.        sinapsis_1 = np.random.uniform(low=-
   2,high=2,size=(hidden_dim+1,output_dim))
260.        sinapsis_h = np.random.uniform(low=-
   2,high=2,size=(hidden_dim,hidden_dim))
261.
262.        temp_awal = ""
263.        for x in range(len(sinapsis_0)):
264.            for y in range(len(sinapsis_0[x])):
265.                if y == hidden_dim-1:
266.                    temp_awal += str(sinapsis_0[x][y])+"\n"
267.                else:
268.                    temp_awal += str(sinapsis_0[x][y])+","
269.
270.        if default_storage.exists('setting/sinapsis_0_awal.txt'):
271.            default_storage.delete('setting/sinapsis_0_awal.txt')
272.        path = default_storage.save('setting/sinapsis_0_awal.txt',
   ContentFile(temp_awal))
273.
274.        temp_awal = ""
275.        for x in range(len(sinapsis_h)):
276.            for y in range(len(sinapsis_h[x])):
277.                if y == hidden_dim-1:
278.                    temp_awal += str(sinapsis_h[x][y])+"\n"
279.                else:
280.                    temp_awal += str(sinapsis_h[x][y])+","
281.
282.        if default_storage.exists('setting/sinapsis_h_awal.txt'):
283.            default_storage.delete('setting/sinapsis_h_awal.txt')
284.        path = default_storage.save('setting/sinapsis_h_awal.txt',
   ContentFile(temp_awal))
285.
```

```
        temp_awal = ""
        for x in range(len(sinapsis_1)):
            for y in range(len(sinapsis_1[x])):
                    temp_awal += str(sinapsis_1[x][y])+"\n"
        if default_storage.exists('setting/sinapsis_1_awal.txt'):
            default_storage.delete('setting/sinapsis_1_awal.txt')
        path = default_storage.save('setting/sinapsis_1_awal.txt',
    ContentFile(temp_awal))


        sinapsis_0_update = np.zeros_like(sinapsis_0)
        sinapsis_1_update = np.zeros_like(sinapsis_1)
        sinapsis_h_update = np.zeros_like(sinapsis_h)

        sinapsis_0_0 = np.zeros_like(sinapsis_0)
        sinapsis_1_0 = np.zeros_like(sinapsis_1)
        sinapsis_h_0 = np.zeros_like(sinapsis_h)

        sinapsis_0_1 = np.zeros_like(sinapsis_0)
        sinapsis_1_1 = np.zeros_like(sinapsis_1)
        sinapsis_h_1 = np.zeros_like(sinapsis_h)

        i =0
        ulang=1
        while ulang==1:
           n=0
           error = 0.00
           for x in range(len(dataset)):
                  masukkan = np.zeros(input_dim+1)
                  for data in range(input_dim):
                         masukkan[data] = dataset[x][data]
                  masukkan[input_dim] = 1
                  target = dataset[x][input_dim]
                  Znet = np.dot(masukkan,sinapsis_0)
                  if aktivasi ==1:
                         Zfnet = sigmoid_biner(Znet)
                  else:
                         Zfnet = sigmoid_bipolar(Znet)
                  Unet = np.dot(Zfnet,sinapsis_h)
                  Zfnet_total = np.zeros_like(sinapsis_1)
                  if aktivasi ==1:
                         Zfnet_total = sigmoid_biner(Znet+Unet)
                  else:
                         Zfnet_total = sigmoid_bipolar(Znet+Unet)
                  temp_Zfnet_total = Zfnet_total
                  Zfnet_total = np.zeros(hidden_dim+1)
                  for temp in range(hidden_dim):
                         Zfnet_total[temp] = temp_Zfnet_total[temp]
                  Y_net = np.dot(Zfnet_total,sinapsis_1)
                  if aktivasi ==1:
                         Y_fnet = sigmoid_biner(Y_net)
                         omega = (target-
    Y_fnet)*turunan_sigmoid_biner(Y_fnet)
                  else:
                         Y_fnet = sigmoid_bipolar(Y_net)
                         omega = (target-
    Y_fnet)*turunan_sigmoid_bipolar(Y_fnet)


                  sinapsis_0_0 = sinapsis_0_1
                  sinapsis_1_0 = sinapsis_1_1
                  sinapsis_h_0 = sinapsis_h_1

                  sinapsis_0_1 = sinapsis_0
                  sinapsis_1_1 = sinapsis_1
                  sinapsis_h_1 = sinapsis_h

                  sinapsis_1_update = perubahan_w(alfa,omega,Zfnet_total)
```

```
                sinapsis_1_update_temp = sinapsis_1_update
                sinapsis_1_update = np.zeros_like(sinapsis_1)

                for temp in range(hidden_dim+1):
                      sinapsis_1_update[temp] =
   sinapsis_1_update_temp[temp]
                      sinapsis_1[temp] =
   momentum(momentum_dim,sinapsis_1[temp],sinapsis_1_update[temp],sinapsis_
   1_1[temp],sinapsis_1_0[temp])
                kesalahan_unit =
   net_faktor_kesalahan_unit(Zfnet_total,faktor_kesalahan_unit(omega))
                for temp in range(input_dim+1):
                      sinapsis_0_update[temp] =
   perubahan_w(alfa,masukkan[temp],kesalahan_unit)
                      sinapsis_0[temp] =
   momentum(momentum_dim,sinapsis_0[temp],sinapsis_0_update[temp],sinapsis_
   0_1[temp],sinapsis_0_0[temp])
                for temp in range(hidden_dim):
                      sinapsis_h_update[temp] =
   perubahan_w(alfa,Zfnet[temp],kesalahan_unit)
                      sinapsis_h[temp] =
   momentum(momentum_dim,sinapsis_h[temp],sinapsis_h_update[temp],sinapsis_
   h_1[temp],sinapsis_h_0[temp])


                error += (target-Y_fnet)**2
                n+=1

          # print ("error mse: "+str(error))
          error = error/n
          log_mse.append(error)

          i+=1
          if error<mse:
                ulang=0
          if i>epoch-1:
                ulang=0
       temp = ""
       for x in range(len(sinapsis_0)):
           for y in range(len(sinapsis_0[x])):
               if y == hidden_dim-1:
                   temp += str(sinapsis_0[x][y])+"\n"
               else:
                   temp += str(sinapsis_0[x][y])+","

       if default_storage.exists('setting/sinapsis_0_temp.txt'):
           default_storage.delete('setting/sinapsis_0_temp.txt')
       path = default_storage.save('setting/sinapsis_0_temp.txt',
   ContentFile(temp))

       temp = ""
       for x in range(len(sinapsis_h)):
           for y in range(len(sinapsis_h[x])):
               if y == hidden_dim-1:
                   temp += str(sinapsis_h[x][y])+"\n"
               else:
                   temp += str(sinapsis_h[x][y])+","

       if default_storage.exists('setting/sinapsis_h_temp.txt'):
           default_storage.delete('setting/sinapsis_h_temp.txt')
       path = default_storage.save('setting/sinapsis_h_temp.txt',
   ContentFile(temp))

       temp = ""
       for x in range(len(sinapsis_1)):
           for y in range(len(sinapsis_1[x])):
                   temp += str(sinapsis_1[x][y])+"\n"
       if default_storage.exists('setting/sinapsis_1_temp.txt'):
```

```
            default_storage.delete('setting/sinapsis_1_temp.txt')
        path = default_storage.save('setting/sinapsis_1_temp.txt', ContentFile(temp))

        dataset = dataset_uji

        total_benar=0
        for x in range(len(dataset)):
                masukkan = np.zeros(input_dim+1)
               # print ("masukkan : "+str(masukkan))

                for data in range(input_dim):
                        masukkan[data] = dataset[x][data]

                masukkan[input_dim] = 1;
                target_data = dataset[x][input_dim]
                Znet = np.dot(masukkan,sinapsis_0)
                if aktivasi ==1:
                        Zfnet = sigmoid_biner(Znet)
                else:
                        Zfnet = sigmoid_bipolar(Znet)
                Unet = np.dot(Zfnet,sinapsis_h)
                Zfnet_total = np.zeros_like(sinapsis_1)
                if aktivasi== 1:
                        Zfnet_total = sigmoid_biner(Znet+Unet)
                else:
                        Zfnet_total = sigmoid_bipolar(Znet+Unet)
                temp_Zfnet_total = Zfnet_total
                Zfnet_total = np.zeros(hidden_dim+1)
                for temp in range(hidden_dim):
                        Zfnet_total[temp] = temp_Zfnet_total[temp]
                Zfnet_total[hidden_dim] = 1
                Y_net = np.dot(Zfnet_total,sinapsis_1)
                if aktivasi ==1:
                        Y_fnet = sigmoid_biner(Y_net)
                        omega = (target-Y_fnet)*turunan_sigmoid_biner(Y_fnet)
                else:
                        Y_fnet = sigmoid_bipolar(Y_net)
                        omega = (target-Y_fnet)*turunan_sigmoid_bipolar(Y_fnet)
                # print ("y("+str(x)+"): "+str(Y_fnet)+"->Target:"+str(target_data))
                log_ynet.append("y("+str(x)+"): "+str(Y_fnet)+"->Target:"+str(target_data))
                if Y_fnet>0.5:
                        Y_fnet=1.0
                else:
                        Y_fnet=0.0

                if Y_fnet==target_data:
                        total_benar+=1

        # print ("Total benar: " + str(total_benar))
        # print ("Jumlah Epoch: " + str(i-1))
        # print ("Akurasi: " + str(total_benar/len(dataset)*100)+"%")

        data = {'mse':log_mse,'ynet':log_ynet, 'benar':total_benar, 'akurasi':str(total_benar/len(dataset)*100)+"%"}
        return data


   def uji_rnn(dataset_uji, alfa_masuk, hidden_masuk, epoch_masuk, mse_masuk, momentum_masuk):

        # Fungsi aktivasi Sigmoid Biner
        def sigmoid_biner(x):
```

```
            output = 1/(1+np.exp(-x))
            return output

        # Fungsi aktivasi Sigmoid Biner
        def sigmoid_bipolar(x):
            output = 2/(1+np.exp(-x)) -1
            return output

        def
    momentum(momentum,sinapsis_2,sinapsis_delta,sinapsis_1,sinapsis_0):
           beda = sinapsis_1-sinapsis_0
           output = sinapsis_2+sinapsis_delta+(momentum*beda)
           return output

        # Turunan fungsi aktivasi Sigmoid Biner
        def turunan_sigmoid_biner(output):
            return output*(1-output)

        def turunan_sigmoid_bipolar(output):
            return (1+output)*(1-output)/2

        def perubahan_w(alfa, omega,x):
           return x*alfa*omega

        def faktor_kesalahan_unit(omega):
           kesalahan = np.zeros(hidden_dim)
           for x in range(hidden_dim):
                  kesalahan[x] = sinapsis_1[x]*omega
           return kesalahan

        def net_faktor_kesalahan_unit(Zfnet_total, faktor_kesalahan):
           kesalahan = np.zeros(hidden_dim)
           for x in range(hidden_dim):
                  kesalahan[x] =
    faktor_kesalahan[x]*turunan_sigmoid_biner(Zfnet_total[x])
           return kesalahan

        dataset = dataset_uji
        alfa = float(alfa_masuk)
        input_dim = len(dataset[0])-1
        hidden_dim = hidden_masuk
        output_dim = 1
        epoch = epoch_masuk
        mse = float(mse_masuk)
        aktivasi = 1
        log_mse = []
        log_ynet = []
        momentum_dim = float(momentum_masuk)


        # membuat neuron
        sinapsis_0 = 2*np.random.random((input_dim+1,hidden_dim)) -1
        sinapsis_1 = 2*np.random.random((hidden_dim+1,output_dim)) - 1
        sinapsis_h = 2*np.random.random((hidden_dim,hidden_dim)) - 1


        with open(default_storage.path('setting/sinapsis_1.txt')) as f:
            content = f.readlines()
            for x in range(len(content)):
                content[x] = content[x].replace("\n","")
                content[x] = content[x].replace("\t","")
            for y in range(len(sinapsis_1)):
                    sinapsis_1[y][0] = float(content[y])

        with open(default_storage.path('setting/sinapsis_0.txt')) as f:
            content = f.readlines()
            for x in range(len(content)):
                content[x] = content[x].replace("\n","")
```

```
534.                content[x] = content[x].replace("\t","")
535.            for x in range(len(content)):
536.                string2 = content[x].split(",")
537.                for y in range(len(string2)):
538.                    sinapsis_0[x][y]= float(string2[y])
539.
540.        with open(default_storage.path('setting/sinapsis_h.txt')) as f:
541.            content = f.readlines()
542.            for x in range(len(content)):
543.                content[x] = content[x].replace("\n","")
544.                content[x] = content[x].replace("\t","")
545.            for x in range(len(content)):
546.                string2 = content[x].split(",")
547.                for y in range(len(string2)):
548.                    sinapsis_h[x][y]= float(string2[y])
549.
550.        total_benar=0
551.        for x in range(len(dataset)):
552.                 masukkan = np.zeros(input_dim+1)
553.                # print ("masukkan : "+str(masukkan))
554.
555.                 for data in range(input_dim):
556.                        masukkan[data] = dataset[x][data]
557.
558.                 masukkan[input_dim] = 1;
559.                 target = dataset[x][input_dim]
560.                 Znet = np.dot(masukkan,sinapsis_0)
561.                 if aktivasi ==1:
562.                        Zfnet = sigmoid_biner(Znet)
563.                 else:
564.                        Zfnet = sigmoid_bipolar(Znet)
565.                 Unet = np.dot(Zfnet,sinapsis_h)
566.                 Zfnet_total = np.zeros_like(sinapsis_1)
567.                 if aktivasi== 1:
568.                        Zfnet_total = sigmoid_biner(Znet+Unet)
569.                 else:
570.                        Zfnet_total = sigmoid_bipolar(Znet+Unet)
571.                 temp_Zfnet_total = Zfnet_total
572.                 Zfnet_total = np.zeros(hidden_dim+1)
573.                 for temp in range(hidden_dim):
574.                        Zfnet_total[temp] = temp_Zfnet_total[temp]
575.                 Zfnet_total[hidden_dim] = 1
576.                 Y_net = np.dot(Zfnet_total,sinapsis_1)
577.                 if aktivasi ==1:
578.                        Y_fnet = sigmoid_biner(Y_net)
579.                        omega = (target-
    Y_fnet)*turunan_sigmoid_biner(Y_fnet)
580.                 else:
581.                        Y_fnet = sigmoid_bipolar(Y_net)
582.                        omega = (target-
    Y_fnet)*turunan_sigmoid_bipolar(Y_fnet)
583.                 # print ("y("+str(x)+"): "+str(Y_fnet)+"-
    >Target:"+str(target_data))
584.                 log_ynet.append("y("+str(x)+"): "+str(Y_fnet)+"-
    >Target:"+str(target))
585.                 if Y_fnet>0.5:
586.                        Y_fnet=1.0
587.                 else:
588.                        Y_fnet=0.0
589.
590.                 if Y_fnet==target:
591.                        total_benar+=1
592.
593.        # print ("Total benar: " + str(total_benar))
594.        # print ("Jumlah Epoch: " + str(i-1))
595.        # print ("Akurasi: " + str(total_benar/len(dataset)*100)+"%")
596.
```

```
597.       data = {'mse':"",'ynet':log_ynet, 'benar':total_benar,
   'akurasi':str(total_benar/len(dataset)*100)+"%"}
598.       return data
```

Panduan Penggunaan Aplikasi Prediksi Banckrupty Berbasis Jaringan Syaraf Tiruan

oleh:

Alamsyah
Budi Prasetiyo
UNIVERSITAS NEGERI SEMARANG

2019

I. Pendahuluan

Buku Pedoman Penggunaan Aplikasi Prediksi Banckrupty merupakan panduan yang berisi petunjuk operasional penggunaan aplikasi prediksi banckrupty untuk melakukan prediksi kebangkrutan perusahaan. Mulai dari login, hingga pengisian tanda-tanda sesuai gejala yang terjadi. Aplikasi ini terdiri dari beberapa menu untuk operator, diantaranya login, logout, dashboard, input gejala, halaman pengujian, dan halaman RNN.

Tahapan pembuatan Aplikasi Prediksi Banckrupty dijelaskan sebagai berikut ini.

(1) Tahap Analisis Kebutuhan Sistem

Fokus utama pada tahap ini yaitu mendeskripsikan secara lengkap mengenai perangkat lunak yang akan dikembangkan. Tahapan ini bertujuan memperoleh data mengenai rancangan dari sistem yang akan dikembangkan sesuai dengan harapan. Berikut adalah tahapannya.

a) Analisis kebutuhan sistem

Kebutuhan sistem adalah kebutuhan yang harus disediakan oleh sistem sehingga dapat dimanfaatkan oleh pengguna. Kebutuhan sistem yang harus terpenuhi yaitu: (1) sistem mampu menampilkan Menu Halaman Awal yang berisi tentang informasi login dan pendaftaran akun; (2) sistem mampu menampilkan Menu Halaman Pengisian Data pada dashboard; (3) sistem mampu menampilkan halaman Prediksi.

b) Analisis pengguna

Analisis pengguna dilakukan untuk menentukan sasaran dari pengguna.

(2) Tahap Desain

Desain dibuat dengan mengacu pada analisis kebutuhan agar perangkat lunak dapat berfungsi sesuai dengan yang diperlukan. Desain tersebut diantaranya adalah desain antarmuka dan desain struktur pangkalan data. Berikut adalah tahapannya.

a) Desain antarmuka

Dalam desain antarmuka, terdapat beberapa antarmuka pada perangkat lunak, yaitu (1) antarmuka halaman awal; (2) antarmuka halaman dashboard; (3) antarmuka halaman user; (4) antarmuka halaman prediksi.

b) Desain logo

Desain logo dibangun dengan tujuan agar dapat merepresentasikan sebuah perangkat lunak. Logo akan menjadi brand yang sangat penting agar pengguna dapat mengenali perangkat lunak dengan mudah.

c) Desain pangkalan data

Perancangan pangkalan data bertujuan supaya tersedianya fasilitas penyimpanan data yang dapat mendukung perangkat lunak yang akan dikembangkan. Pangkalan data yang digunakan adalah *database MySQL*. Pangkalan data digunakan untuk menyimpan data pengguna, pasien/dataset, data gejala, dan untuk menyimpan hasil prediksi/klasifikasi.

(3) Tahap *Coding* (implemetasi)

Tahap ini merupakan realisasi tahap analisis dan tahap desain dengan cara memulai *coding* dan memasang secara *online*. Kode secara nyata ditulis dan disusun menjadi sebuah aplikasi operasional, pangkalan data dan kebutuhan lainnya sudah dipelajari pada tahap sebelumnya. Kode ditulis dengan bahasa PHP dan *database MySQL*.

(4) Tahap Pengujian

Pada tahap ini dilakukan pengujian oleh pakar (ahli) yaitu pakar kesehatan, serta ahli sistem (IT). Pengujian dilakukan setelah perangkat lunak sudah menjadi suatu perangkat lunak yang siap pakai. Pengujian dilakukan untuk memastikan bahwa perangkat lunak yang dibuat telah sesuai dengan harapan awal atau tidak.

(5) Tahap Revisi

Setelah tahap pengujian dilakukan, tahap revisi mulai dijalankan.Tahap revisi merupakan tahap perbaikan berdasarkan masukkan dari ahli. Masukkan dari ahli akan ditinjau ulang dan direalisasikan. Tahap ini memungkinkan akan membentuk *cycle* atau kembali ke tahap *coding* untuk perbaikan.

(6) Tahap Penggunaan

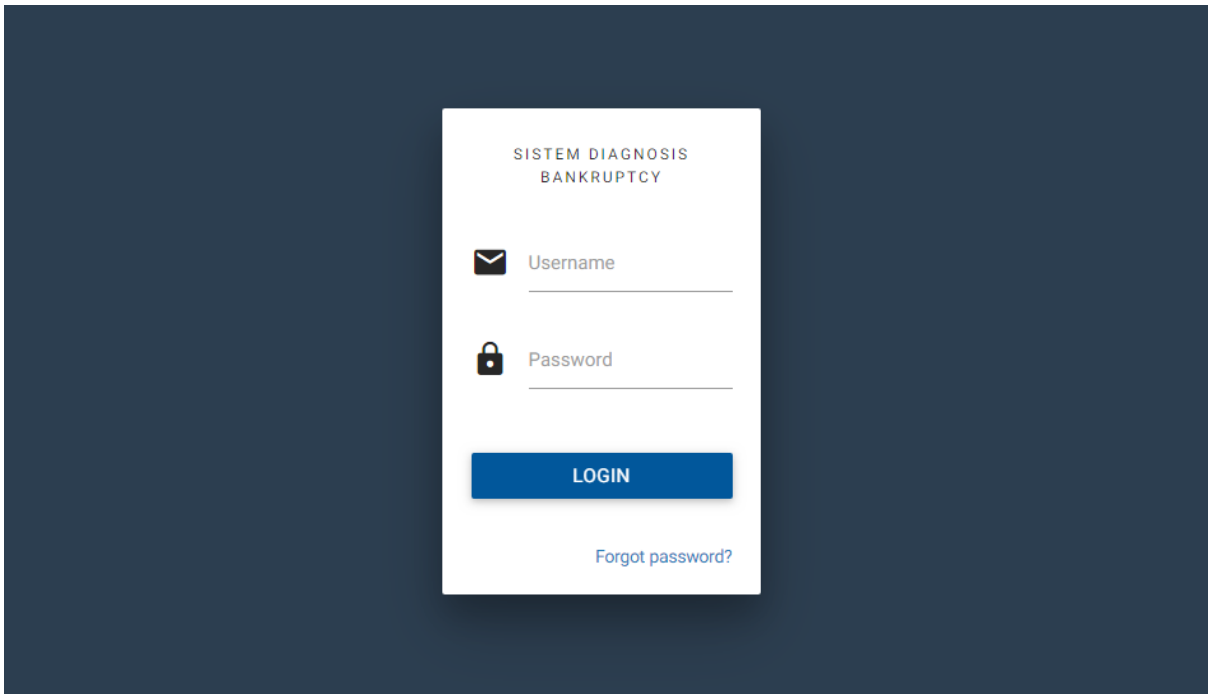
Tahap penggunaan dilakukan saat laman (perangkat lunak) benar-benar siap dioperasionalkan. Tahap ini diberikan kepada pengguna akhir secara langsung. Pengguna akhir melakukan simulasi prediksi.

II. Petuntuk Penggunaan Sistem

A. Login Sistem

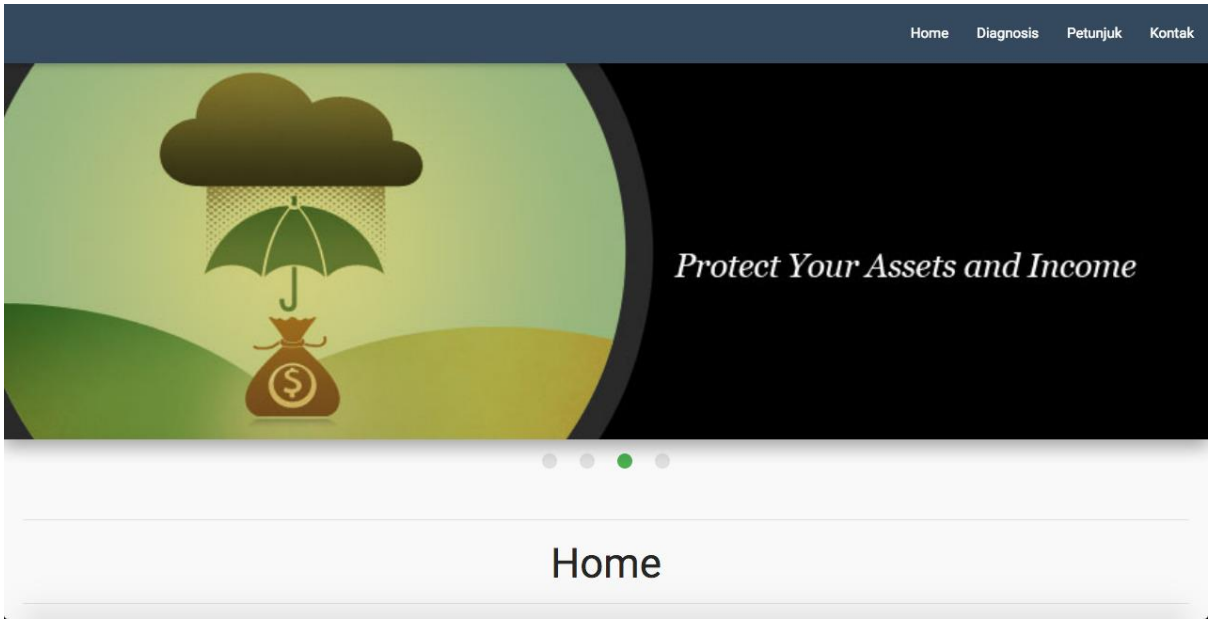
Sistem ini berbasis aplikasi laman (web). Untuk dapat mengakses aplikasi ini diperlukan koneksi internet.

1. Silakan klik laman **http://103.23.100.138:8080**
2. Memasukkan username dan password untuk login yang telah dibagikan. Atau untuk demo bisa menggunakan user: admin, password:12345. Pada halaman awal pengguna perlu memasukan username dan password seperti pada Gambar 1.



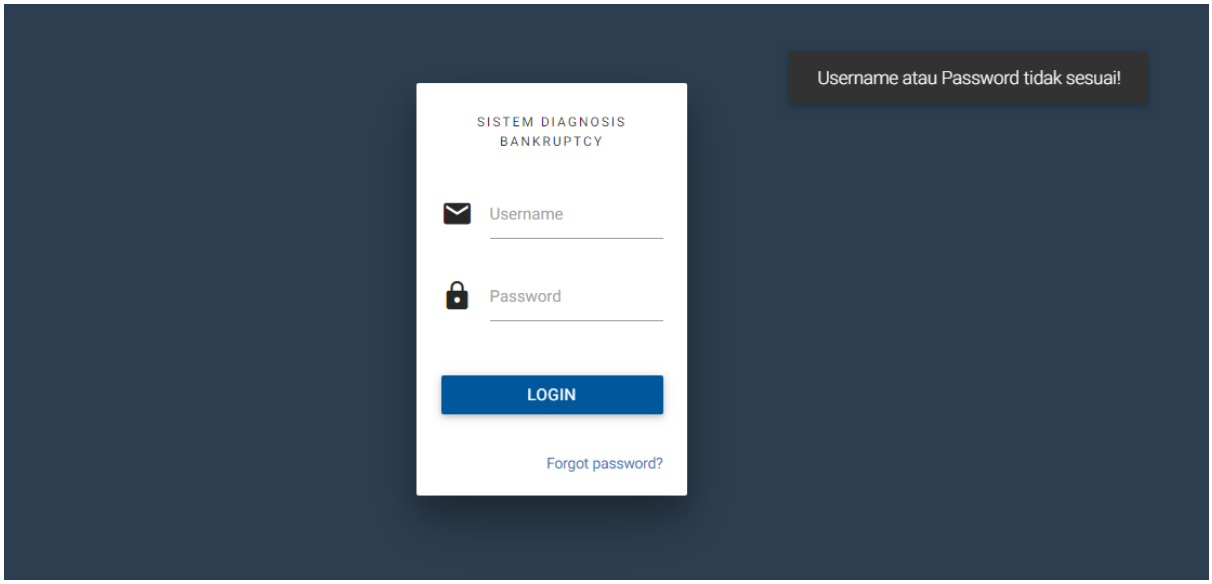
Gambar 1. Menu login

3. Setelah login sukses, pengguna akan masuk kehalaman dashboard, lihat Gambar 2.



Gambar 2. Halaman dashboard

4. Apabila login gagal, periksa kembali username & password. Status login gagal ditunjukan pada Gambar 3.

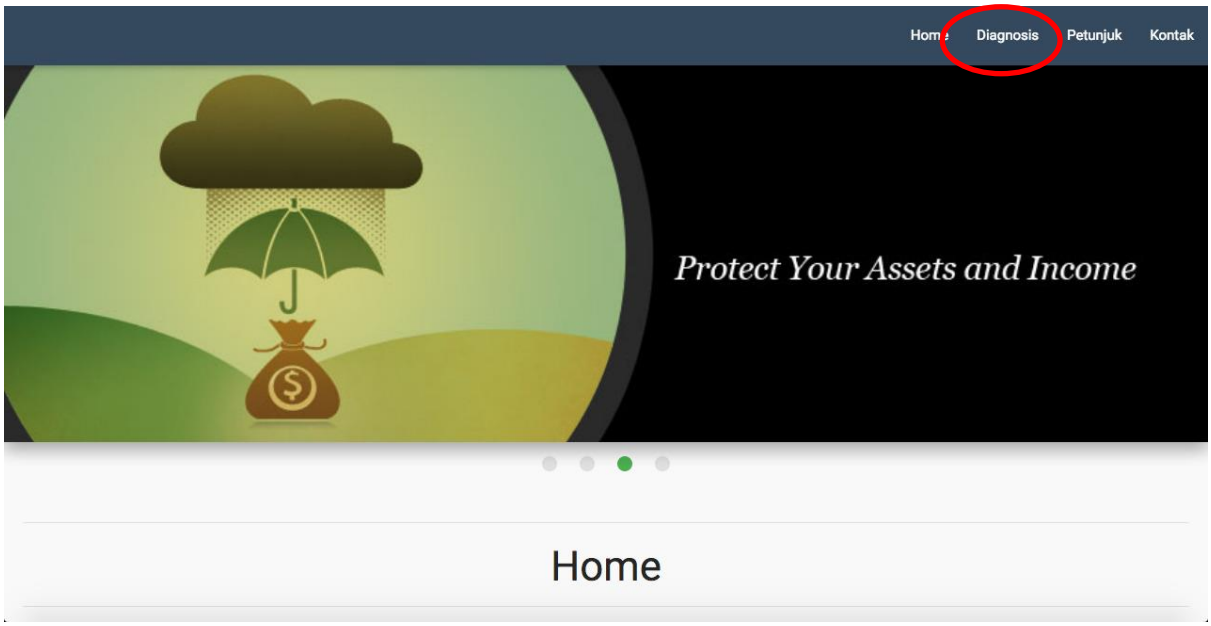


Gambar 3. Login gagal

B. Melakukan Prediksi

Prediksi kebangrutan/banckrupty berdasarkan dataset yang diperoleh dari UCI Machine Learning repository. Untuk melakukan prediksi, klik menu Diagnosis 

seperti pada Gambar 4.

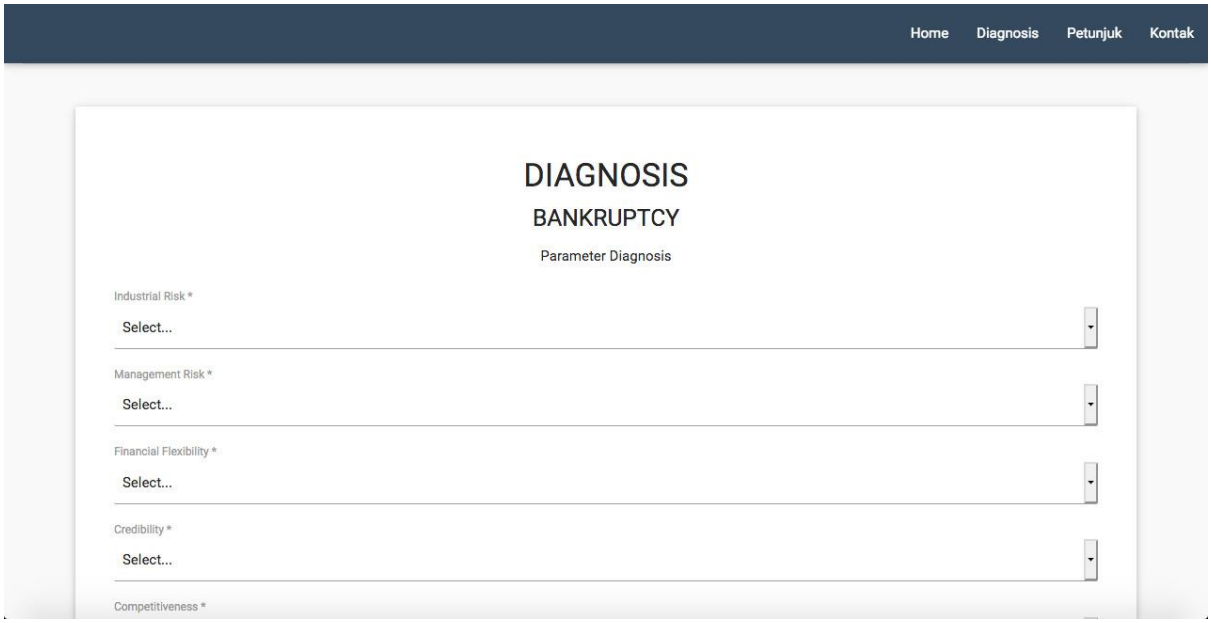


Gambar 4. Menu untuk masuk masuk ke halaman diagnosis

Selanjutnya akan tampil halaman diagnosisi seperti pada Gamabar 5. Pada halaman ini pengguna akan diminta untuk menginputkan gejala-gejala/tanda-tanda diantaranya:

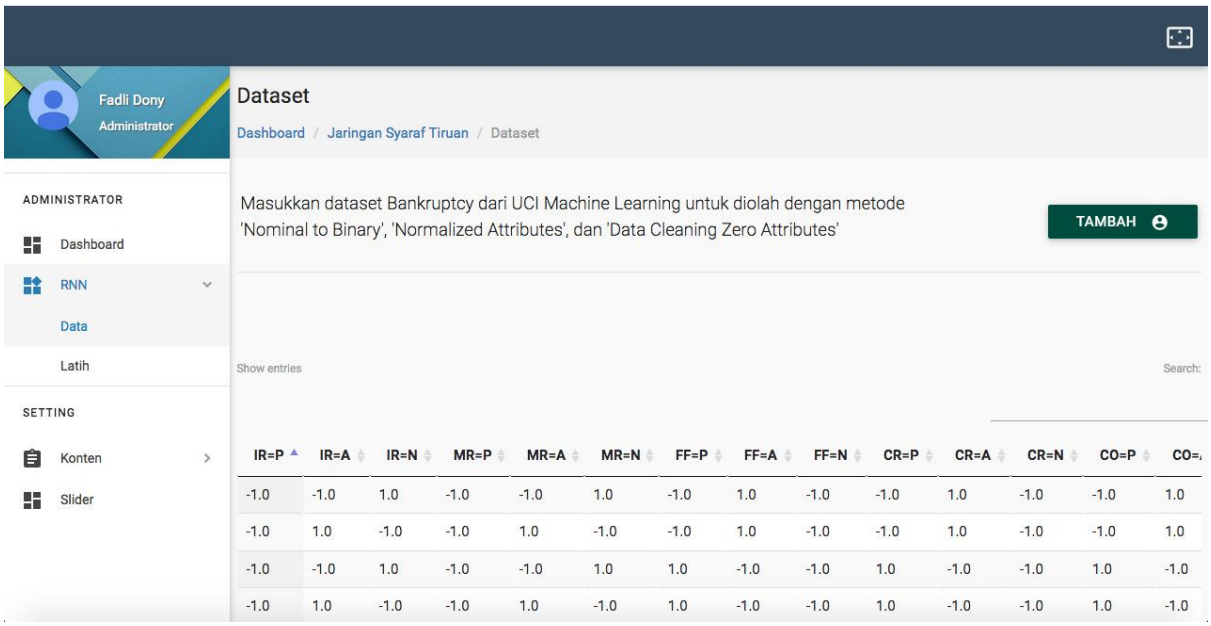
1) *Industrial Risk*: {P,A,N}
2) *Management Risk*: {P,A,N}
3) *Financial Flexibility*: {P,A,N}
4) *Credibility*: {P,A,N}
5) *Competitiveness*: {P,A,N}
6) *Operating Risk*: {P,A,N}

Informasi atribut: (P=*Positive*, A-*Average*, N-*negative*)
Seperti yang ditunjukkan pada Gambar 5.



Gambar 5. Input gejala/tanda-tanda

Kemudian klik simpan, selanjutnya akan diarahkan pada halaman dataset seperti pada Gambar 6.

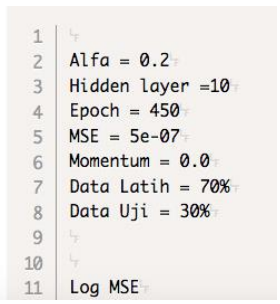


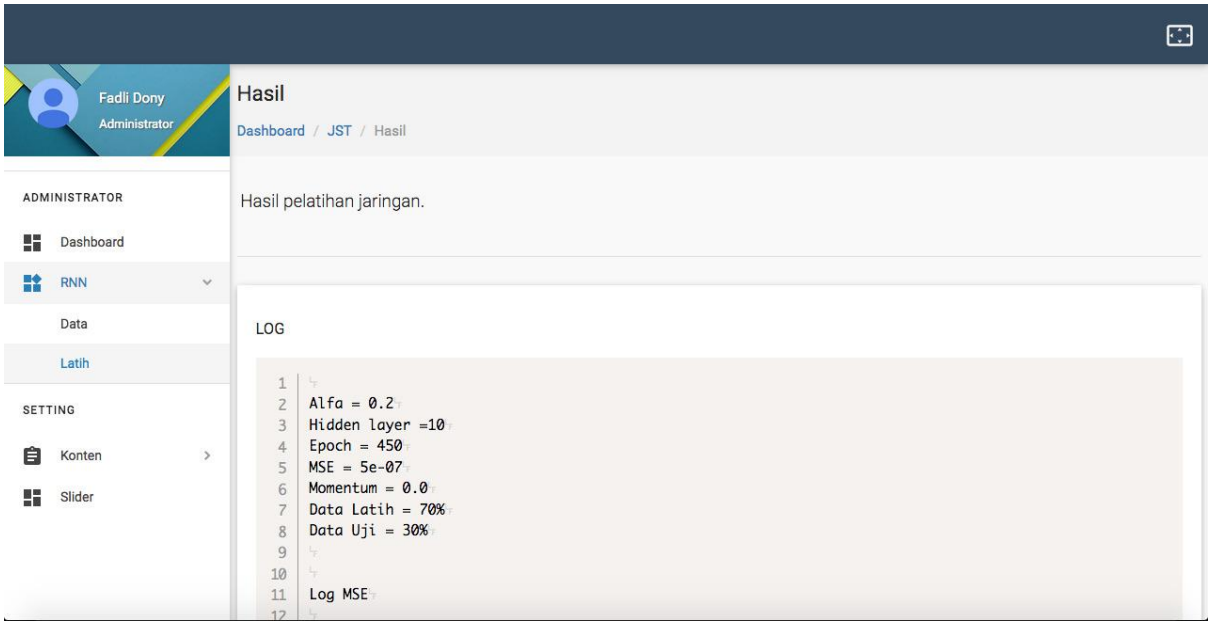
Contoh dataset yang digunakan seperti pada Tabel 1.

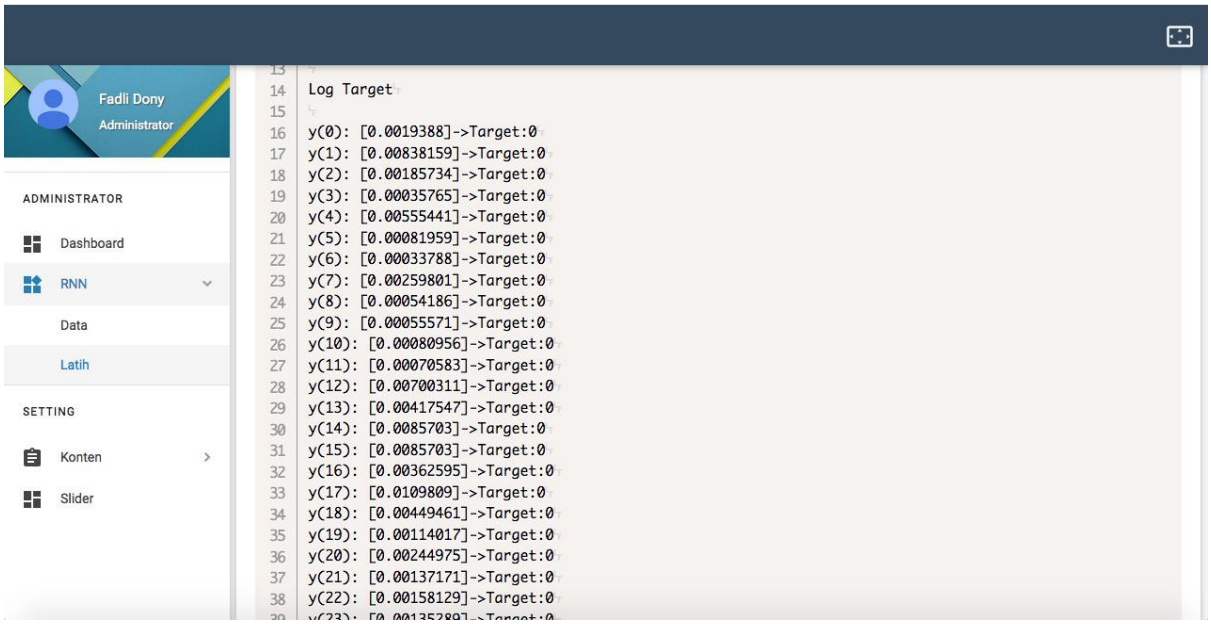
Tabel 1.Contoh Dataset sebagai data training.

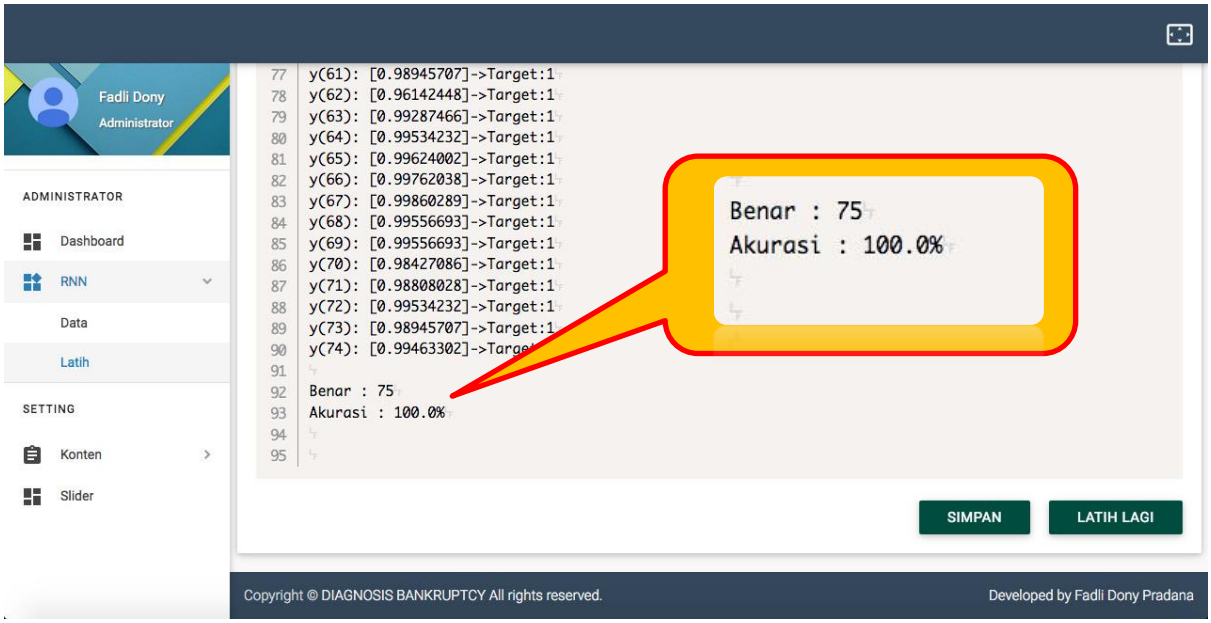
IR	MR	FF	CR	CO	OP	Class
P	P	A	A	A	P	NB
N	N	A	A	A	N	NB
A	A	A	A	A	A	NB
P	P	P	P	P	P	NB
N	N	P	P	P	N	NB
A	A	P	P	P	A	NB
P	P	A	P	P	P	NB
P	P	P	A	A	P	NB
P	P	A	P	A	P	NB
P	P	A	A	P	P	NB
P	P	P	P	A	P	NB
P	P	P	A	P	P	NB
N	N	A	P	P	N	NB
N	N	P	A	A	N	NB
N	N	A	P	A	N	NB
N	N	A	P	A	N	NB
N	N	A	A	P	N	NB
N	N	P	P	A	N	NB
N	N	P	A	P	N	NB
A	A	A	P	P	A	NB
A	A	P	A	A	A	NB
A	A	A	P	A	A	NB
A	A	A	A	P	A	NB
A	A	P	P	A	A	NB
A	A	P	A	P	A	NB
P	N	A	A	A	P	NB
N	P	A	A	A	N	NB
P	N	A	A	A	N	NB
P	N	P	P	P	P	NB
N	P	P	P	P	N	NB
P	N	P	P	P	N	NB
N	N	A	P	P	P	NB
P	N	P	A	A	P	NB
N	P	A	P	A	P	NB
N	P	A	A	P	N	NB
P	N	P	P	A	N	NB
N	P	P	A	P	A	NB
…	…	…	…	…	…	…
P	N	N	N	A	A	B

Selanjutnya operator dapat konfigurasi pelatihan (training) Jaringan Syaraf Tiruan untuk membentuk pola model, sehingganti dapat digunakan untukmelakukan prediksi *Banckrupty*. Pengaturan konfigurasi ditunjukkan pada Gambar berikut.




Fadli Dony
Administrator
ADMINISTRATOR
Dashboard
RNN
Data
Latih
SETTING
Konten
Slider
Hasil
Dashboard / JST / Hasil
Hasil pelatihan jaringan.
LOG
Alfa = 0.2
Hidden layer =10
Epoch = 450
MSE = 5e-07
Momentum = 0.0
Data Latih = 70%
Data Uji = 30%
Log MSE


Fadli Dony
Administrator
ADMINISTRATOR
Dashboard
RNN
Data
Latih
SETTING
Konten
Slider
Log Target
y(0): [0.0019388]->Target:0
y(1): [0.00838159]->Target:0
y(2): [0.00185734]->Target:0
y(3): [0.00035765]->Target:0
y(4): [0.00555441]->Target:0
y(5): [0.00081959]->Target:0
y(6): [0.00033788]->Target:0
y(7): [0.00259801]->Target:0
y(8): [0.00054186]->Target:0
y(9): [0.00055571]->Target:0
y(10): [0.00080956]->Target:0
y(11): [0.00070583]->Target:0
y(12): [0.00700311]->Target:0
y(13): [0.00417547]->Target:0
y(14): [0.0085703]->Target:0
y(15): [0.0085703]->Target:0
y(16): [0.00362595]->Target:0
y(17): [0.0109809]->Target:0
y(18): [0.00449461]->Target:0
y(19): [0.00114017]->Target:0
y(20): [0.00244975]->Target:0
y(21): [0.00137171]->Target:0
y(22): [0.00158129]->Target:0


Fadli Dony
Administrator
ADMINISTRATOR
Dashboard
RNN
Data
Latih
SETTING
Konten
Slider
y(61): [0.98945707]->Target:1
y(62): [0.96142448]->Target:1
y(63): [0.99287466]->Target:1
y(64): [0.99534232]->Target:1
y(65): [0.99624002]->Target:1
y(66): [0.99762038]->Target:1
y(67): [0.99860289]->Target:1
y(68): [0.99556693]->Target:1
y(69): [0.99556693]->Target:1
y(70): [0.98427086]->Target:1
y(71): [0.98808028]->Target:1
y(72): [0.99534232]->Target:1
y(73): [0.98945707]->Target:1
y(74): [0.99463302]->Target:1
Benar : 75
Akurasi : 100.0%
Benar : 75
Akurasi : 100.0%
SIMPAN
LATIH LAGI
Copyright © DIAGNOSIS BANKRUPTCY All rights reserved.
Developed by Fadli Dony Pradana

C. Keluar Aplikasi (Logout)

Demi kemanan apabila telah selesai melakukan aktivitas, pengguna disarankan untuk logout dengan klik "User" 

, kemudian "Logout" 

.