**LAPORAN UAS**

**Teknologi Data**

**UAS**



**Oleh:**

**Bagus Satria putra 1841720146 TI-3F**

**PROGRAM STUDI D-IV TEKNIK INFORMATIKA JURUSAN TEKNOLOGI INFORMASI POLITEKNIK NEGERI MALANG**

**26 JUNI 2021**

|  |  |
| --- | --- |
| **No.** | **Keterangan** |
|  | Membuat program naivebayes.py  import pandas as pd  from probaspek import ProbAspek  class NaiveBayes:      def \_\_init\_\_(self):          pass          # TODO: [LANGKAH-2] Buat property untuk menampung data dari file CSV          self.data\_training = None          # TODO: [Langkah-3] Buat variabel dictionary untuk menampung matriks Probabilitas untuk semua aspek          self.aspek\_umur = {'Tua': None, 'Muda': None}          self.aspek\_status = {'Belum Kawin': None, 'Kawin': None, 'Cerai': None}          self.aspek\_pendidikan = {'Tidak Sekolah': None, 'SD': None, 'SLTP': None, 'SLTA': None, 'Sarjana': None}          self.aspek\_tanggungan = {'0': None, '1': None, '2': None, '3': None, '4': None, '5': None}          self.aspek\_pekerjaan = {'Tiada': None, 'Buruh Lepas': None, 'Petani': None, 'Aparatur Negara': None}          self.aspek\_penghasilan = {'Tiada': None, 'Rendah': None, 'Sedang': None, 'Tinggi': None}          # TODO: [Langkah-4] Buat variabel untuk menampung Prior Probability          self.prior\_probability = {'Miskin': 0, 'Tidak Miskin': 0}      # TODO: [LANGKAH-5] Load data training dari file CSV      def load\_data\_training(self):          self.data\_training = pd.read\_csv('daftar\_masyarakat\_miskin.csv', sep=';')          print(self.data\_training)          print('-------------------------------------------------------------')      # TODO: [LANGKAH-6] Membuat object ProbAspek untuk semua nilai pada aspek, sekaligus menghitung jumlah miskin dan tidak miskin      def buat\_prob\_aspek(self, nama\_aspek: str, nilai\_aspek: str) -> ProbAspek:          #probilitas asek kosong          prob\_aspek = ProbAspek(nama\_aspek, nilai\_aspek)          #menghitung jumlah miskin dan tidak miskin          tua\_miskin = self.data\_training.loc[(self.data\_training[nama\_aspek] == nilai\_aspek) &                                               (self.data\_training['Actual Class'] == 'Miskin')]          tua\_tidak\_miskin = self.data\_training.loc[(self.data\_training[nama\_aspek] == nilai\_aspek) &                                               (self.data\_training['Actual Class'] == 'Tidak Miskin')]          prob\_aspek.jml\_miskin = len(tua\_miskin)          prob\_aspek.jml\_tidak\_miskin= len(tua\_tidak\_miskin)          return prob\_aspek      # TODO: [LANGKAH-7] Mengisi semua nilai pada matris probabilitas aspek      def mulai\_training(self):          # Aspek Umur          pu\_tua = self.buat\_prob\_aspek('Umur', 'Tua')          pu\_muda = self.buat\_prob\_aspek('Umur', 'Muda')          # Jadikan array          arr\_pu = [pu\_tua, pu\_muda]          # Hitung total masing-masing nilai aspek berapa kali muncul di miskin dan tidak miskin          total\_u = ProbAspek.hitung\_jml\_total\_aspek(arr\_pu)          # Hitung probabilitas aspek untuk masing-masing nilai aspek          pu\_tua.hitung\_p\_aspek\_miskin(total\_u['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_u['Tidak Miskin'])          pu\_muda.hitung\_p\_aspek\_miskin(total\_u['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_u['Tidak Miskin'])          # Print matrix probabilitas, tetapi bentuknya vertikal, bukan tabel          ProbAspek.print\_matrix\_probabilitas(arr\_pu)          self.aspek\_umur['Tua'] = pu\_tua          self.aspek\_umur['Muda'] = pu\_muda          # TODO :1.1 Status          # Aspek Status          ps\_belum\_kawin = self.buat\_prob\_aspek('Status', 'Belum Kawin')          ps\_kawin = self.buat\_prob\_aspek('Status', 'Kawin')          ps\_cerai = self.buat\_prob\_aspek('Status', 'Cerai')          # Jadikan array          arr\_ps = [ps\_belum\_kawin, ps\_kawin, ps\_cerai]          # Hitung total masing-masing nilai aspek berapa kali muncul di miskin dan tidak miskin          total\_s = ProbAspek.hitung\_jml\_total\_aspek(arr\_ps)          # Hitung probabilitas aspek untuk masing-masing nilai aspek          ps\_belum\_kawin.hitung\_p\_aspek\_miskin(total\_s['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_s['Tidak Miskin'])          ps\_kawin.hitung\_p\_aspek\_miskin(total\_s['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_s['Tidak Miskin'])          ps\_cerai.hitung\_p\_aspek\_miskin(total\_s['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_s['Tidak Miskin'])          # Print matrix probabilitas, tetapi bentuknya vertikal, bukan tabel          ProbAspek.print\_matrix\_probabilitas(arr\_ps)          self.aspek\_status['Belum Kawin'] = ps\_belum\_kawin          self.aspek\_status['Kawin'] = ps\_kawin          self.aspek\_status['Cerai'] = ps\_cerai          # TODO :1.2 Pendidikan          # Aspek Pendidikan          pn\_tidak\_sekolah = self.buat\_prob\_aspek('Pendidikan', 'Tidak Sekolah')          pn\_sd = self.buat\_prob\_aspek('Pendidikan', 'SD')          pn\_sltp = self.buat\_prob\_aspek('Pendidikan', 'SLTP')          pn\_slta = self.buat\_prob\_aspek('Pendidikan', 'SLTA')          pn\_sarjana = self.buat\_prob\_aspek('Pendidikan', 'Sarjana')          # Jadikan array          arr\_pn = [pn\_tidak\_sekolah, pn\_sd, pn\_sltp, pn\_slta, pn\_sarjana]          # Hitung total masing-masing nilai aspek berapa kali muncul di miskin dan tidak miskin          total\_pn = ProbAspek.hitung\_jml\_total\_aspek(arr\_pn)          # Hitung probabilitas aspek untuk masing-masing nilai aspek          pn\_tidak\_sekolah.hitung\_p\_aspek\_miskin(total\_pn['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pn['Tidak Miskin'])          pn\_sd.hitung\_p\_aspek\_miskin(total\_pn['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pn['Tidak Miskin'])          pn\_sltp.hitung\_p\_aspek\_miskin(total\_pn['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pn['Tidak Miskin'])          pn\_slta.hitung\_p\_aspek\_miskin(total\_pn['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pn['Tidak Miskin'])          pn\_sarjana.hitung\_p\_aspek\_miskin(total\_pn['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pn['Tidak Miskin'])          # Print matrix probabilitas, tetapi bentuknya vertikal, bukan tabel          ProbAspek.print\_matrix\_probabilitas(arr\_pn)          self.aspek\_pendidikan['Tidak Sekolah'] = pn\_tidak\_sekolah          self.aspek\_pendidikan['SD'] = pn\_sd          self.aspek\_pendidikan['SLTP'] = pn\_sltp          self.aspek\_pendidikan['SLTA'] = pn\_slta          self.aspek\_pendidikan['Sarjana'] = pn\_sarjana          # TODO :1.3 Tanggungan          # Aspek Tanggungan          pt\_nol = self.buat\_prob\_aspek('Tanggungan', 'Tidak Ada')          pt\_satu = self.buat\_prob\_aspek('Tanggungan', '1')          pt\_dua = self.buat\_prob\_aspek('Tanggungan', '2')          pt\_tiga = self.buat\_prob\_aspek('Tanggungan', '3')          pt\_empat = self.buat\_prob\_aspek('Tanggungan', '4')          pt\_lima = self.buat\_prob\_aspek('Tanggungan', '5')          # Jadikan array          arr\_pt = [pt\_nol, pt\_satu, pt\_dua, pt\_tiga, pt\_empat, pt\_lima]          # Hitung total masing-masing nilai aspek berapa kali muncul di miskin dan tidak miskin          total\_pt = ProbAspek.hitung\_jml\_total\_aspek(arr\_pt)          # Hitung probabilitas aspek untuk masing-masing nilai aspek          pt\_nol.hitung\_p\_aspek\_miskin(total\_pt['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pt['Tidak Miskin'])          pt\_satu.hitung\_p\_aspek\_miskin(total\_pt['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pt['Tidak Miskin'])          pt\_dua.hitung\_p\_aspek\_miskin(total\_pt['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pt['Tidak Miskin'])          pt\_tiga.hitung\_p\_aspek\_miskin(total\_pt['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pt['Tidak Miskin'])          pt\_empat.hitung\_p\_aspek\_miskin(total\_pt['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pt['Tidak Miskin'])          pt\_lima.hitung\_p\_aspek\_miskin(total\_pt['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pt['Tidak Miskin'])          # Print matrix probabilitas, tetapi bentuknya vertikal, bukan tabel          ProbAspek.print\_matrix\_probabilitas(arr\_pt)          self.aspek\_tanggungan['Tidak Ada'] = pt\_nol          self.aspek\_tanggungan['1'] = pt\_satu          self.aspek\_tanggungan['2'] = pt\_dua          self.aspek\_tanggungan['3'] = pt\_tiga          self.aspek\_tanggungan['4'] = pt\_empat          self.aspek\_tanggungan['5'] = pt\_lima          # TODO :1.4 Pekerjaan          # Aspek Pekerjaan          pk\_tiada = self.buat\_prob\_aspek('Pekerjaan', 'Tiada')          pk\_buruh = self.buat\_prob\_aspek('Pekerjaan', 'Buruh Lepas')          pk\_petani = self.buat\_prob\_aspek('Pekerjaan', 'Petani')          pk\_aparatur = self.buat\_prob\_aspek('Pekerjaan', 'Aparatur Negara')          # Jadikan array          arr\_pk = [pk\_tiada, pk\_buruh, pk\_petani, pk\_aparatur]          # Hitung total masing-masing nilai aspek berapa kali muncul di miskin dan tidak miskin          total\_pk = ProbAspek.hitung\_jml\_total\_aspek(arr\_pk)          # Hitung probabilitas aspek untuk masing-masing nilai aspek          pk\_tiada.hitung\_p\_aspek\_miskin(total\_pk['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pk['Tidak Miskin'])          pk\_buruh.hitung\_p\_aspek\_miskin(total\_pk['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pk['Tidak Miskin'])          pk\_petani.hitung\_p\_aspek\_miskin(total\_pk['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pk['Tidak Miskin'])          pk\_aparatur.hitung\_p\_aspek\_miskin(total\_pk['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_pk['Tidak Miskin'])          # Print matrix probabilitas, tetapi bentuknya vertikal, bukan tabel          ProbAspek.print\_matrix\_probabilitas(arr\_pk)          self.aspek\_pekerjaan['Tiada'] = pk\_tiada          self.aspek\_pekerjaan['Buruh Lepas'] = pk\_buruh          self.aspek\_pekerjaan['Petani'] = pk\_petani          self.aspek\_pekerjaan['Aparatur Negara'] = pk\_aparatur          # TODO :1.5 Penghasilan          # Aspek Penghasilan          ph\_tiada = self.buat\_prob\_aspek('Penghasilan', 'Tiada')          ph\_rendah = self.buat\_prob\_aspek('Penghasilan', 'Rendah')          ph\_sedang = self.buat\_prob\_aspek('Penghasilan', 'Sedang')          ph\_tinggi = self.buat\_prob\_aspek('Penghasilan', 'Tinggi')          # Jadikan array          arr\_ph = [ph\_tiada, ph\_rendah, ph\_sedang, ph\_tinggi]          # Hitung total masing-masing nilai aspek berapa kali muncul di miskin dan tidak miskin          total\_ph = ProbAspek.hitung\_jml\_total\_aspek(arr\_ph)          # Hitung probabilitas aspek untuk masing-masing nilai aspek          ph\_tiada.hitung\_p\_aspek\_miskin(total\_ph['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_ph['Tidak Miskin'])          ph\_rendah.hitung\_p\_aspek\_miskin(total\_ph['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_ph['Tidak Miskin'])          ph\_sedang.hitung\_p\_aspek\_miskin(total\_ph['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_ph['Tidak Miskin'])          ph\_tinggi.hitung\_p\_aspek\_miskin(total\_ph['Miskin']).hitung\_p\_aspek\_tidak\_miskin(total\_ph['Tidak Miskin'])          # Print matrix probabilitas, tetapi bentuknya vertikal, bukan tabel          ProbAspek.print\_matrix\_probabilitas(arr\_ph)          self.aspek\_penghasilan['Tiada'] = ph\_tiada          self.aspek\_penghasilan['Rendah'] = ph\_rendah          self.aspek\_penghasilan['Sedang'] = ph\_sedang          self.aspek\_penghasilan['Tinggi'] = ph\_tinggi      # TODO: [LANGKAH-8] Menghitung prior probability      def hitung\_prior\_probability(self):          pp\_miskin = self.buat\_prob\_aspek('Actual Class', 'Miskin')          pp\_tidak\_miskin = self.buat\_prob\_aspek('Actual Class', 'Tidak Miskin')          arr\_pp = (pp\_miskin, pp\_tidak\_miskin)          total\_pp = ProbAspek.hitung\_jml\_total\_aspek(arr\_pp)          self.prior\_probability['Miskin'] = total\_pp['Miskin'] / (total\_pp['Miskin'] + total\_pp['Tidak Miskin'])          self.prior\_probability['Tidak Miskin'] = total\_pp['Tidak Miskin'] / (total\_pp['Miskin'] + total\_pp['Tidak Miskin'])          # TODO: [SOAL-2] Prior Probability-nya masih 0, hitunglah prior probability yang sebenarnya!      # TODO: [LANGKAH-9] Membuat method untuk memprediksi hasil akhir berdasarkan nilai aspek      def prediksi(self, nilai\_umur: str, nilai\_status: str, nilai\_pendidikan: str, nilai\_tanggungan: str, nilai\_pekerjaan: str, nilai\_penghasilan: str):          self.hitung\_prior\_probability()          predict\_miskin = self.prior\_probability['Miskin'] \* \                          self.aspek\_umur[nilai\_umur].p\_aspek\_miskin \* \                          self.aspek\_status[nilai\_status].p\_aspek\_miskin \* \                          self.aspek\_pendidikan[nilai\_pendidikan].p\_aspek\_miskin \* \                          self.aspek\_tanggungan[nilai\_tanggungan].p\_aspek\_miskin \* \                          self.aspek\_pekerjaan[nilai\_pekerjaan].p\_aspek\_miskin \* \                          self.aspek\_penghasilan[nilai\_penghasilan].p\_aspek\_miskin          print('Peluang Miskin: {}'.format(predict\_miskin))          predict\_tidak\_miskin = self.prior\_probability['Tidak Miskin'] \* \                          self.aspek\_umur[nilai\_umur].p\_aspek\_tidak\_miskin \* \                          self.aspek\_status[nilai\_status].p\_aspek\_tidak\_miskin \* \                          self.aspek\_pendidikan[nilai\_pendidikan].p\_aspek\_tidak\_miskin \* \                          self.aspek\_tanggungan[nilai\_tanggungan].p\_aspek\_tidak\_miskin \* \                          self.aspek\_pekerjaan[nilai\_pekerjaan].p\_aspek\_tidak\_miskin \* \                          self.aspek\_penghasilan[nilai\_penghasilan].p\_aspek\_tidak\_miskin          print('Peluang Tidak Miskin: {}'.format(predict\_tidak\_miskin))          if predict\_tidak\_miskin > predict\_miskin:              hasil = "Tidak Miskin"              peluang = predict\_tidak\_miskin          else:              hasil = "Miskin"              peluang = predict\_miskin          return {'hasil': hasil, 'peluang': peluang} |
|  | Membuat Probaspek.py  class ProbAspek:      def \_\_init\_\_(self, nama\_aspek: str, nilai\_aspek: str):          pass      # TODO: [LANGKAH-1] Buat class untuk menampung nilai matriks probabilitas          self.nama\_aspek = nama\_aspek          self.nilai\_aspek = nilai\_aspek          self.jml\_miskin = 0          self.jml\_tidak\_miskin = 0          self.p\_aspek\_miskin = 0          self.p\_aspek\_tidak\_miskin = 0      #      def hitung\_p\_aspek\_miskin(self, jml\_total\_miskin\_aspek):          # try:          self.p\_aspek\_miskin = self.jml\_miskin / jml\_total\_miskin\_aspek          # except ZeroDivisionError:          #     self.p\_aspek\_miskin = 0          return self      #      def hitung\_p\_aspek\_tidak\_miskin(self, jml\_total\_tidak\_miskin\_aspek):          # try:          self.p\_aspek\_tidak\_miskin = self.jml\_tidak\_miskin / jml\_total\_tidak\_miskin\_aspek          # except ZeroDivisionError:          #     self.p\_aspek\_tidak\_miskin = 0          return self      #      def print(self):          print('Aspek    : {}'.format(self.nama\_aspek))          print('Nilai    : {}'.format(self.nilai\_aspek))          print('Jml Miskin: {}'.format(self.jml\_miskin))          print('Jml Tidak Miskin: {}'.format(self.jml\_tidak\_miskin))          print('P({}|Miskin): {}'.format(self.nilai\_aspek, self.p\_aspek\_miskin))          print('P({}|Tidak Miskin): {}'.format(self.nilai\_aspek, self.p\_aspek\_tidak\_miskin))          print('------------------------------------------')      #      @staticmethod      def hitung\_jml\_total\_aspek(pa\_list: list) -> dict:          jumlah = {'Miskin': 0, 'Tidak Miskin': 0}          for pa in pa\_list:              jumlah['Miskin'] += pa.jml\_miskin              jumlah['Tidak Miskin'] += pa.jml\_tidak\_miskin          return jumlah      #      @staticmethod      def print\_matrix\_probabilitas(pa\_list: list):          for pa in pa\_list:              pa.print() |
|  | Main.py  from naivebayes import NaiveBayes  class Main:      @staticmethod      def main():          nb = NaiveBayes()          nb.load\_data\_training()          nb.mulai\_training()          nilaiUmur = input("Masukkan nilai Umur[Tua/Muda] : ")          nilaiStatus = input("Masukkan nilai Status [Belum Kawin/Kawin/Cerai]: ")          nilaiPendidikan = input("Masukkan nilai Pendidikan [Tidak Sekolah/SD/SLTP/SLTA/Sarjana] : ")          nilaiTanggungan = input("Masukkan nilai Tanggungan [Tidak Ada/1/2/3/4/5]: ")          nilaiPekerjaan = input("Masukkan nilai Pekerjaan [Tiada/Buruh Lepas/Petani/Aparatur Negara] : ")          nilaiPenghasilan = input("Masukkan nilai Penghasilan [Tiada/Rendah/Sedang/Tinggi] : ")          hasil\_prediksi = nb.prediksi(nilai\_umur=nilaiUmur,                                       nilai\_status=nilaiStatus,                                       nilai\_pendidikan=nilaiPendidikan,                                       nilai\_tanggungan=nilaiTanggungan,                                       nilai\_pekerjaan=nilaiPekerjaan,                                       nilai\_penghasilan=nilaiPenghasilan)          print('=====================================')          print('Hasil akhir prediksi = {}, dengan peluang sebesar {}%'.format(hasil\_prediksi['hasil'],                                                                               hasil\_prediksi['peluang']))  Main.main() |
|  | Bikin data daftar masyarakat.csv |
|  | Buat data daftar\_masyarakat\_miskin.xlsx |
|  | Kita run main.py  Muncul data |
|  | Muncul data yang diurutkan secara vertical |
|  |  |
|  |  |
|  |  |
|  | Kita input ke pemrograman tersebut |
|  | Muncul Hasil akhir sebagai beriikut |

***--Alhamdulillah--***