DAFTAR PUSTAKA

Abdan Syakuro 2017. “ANALISIS SENTIMEN MASYARAKAT TERHADAP E-COMMERCE PADA MEDIA SOSIAL MENGGUNAKAN METODE NAÏVE BAYES CLASSIFIER (NBC) DENGAN SELEKSI FITUR INFORMATION GAIN (IG)”

Defit, S. (2013). Penggunaan Algoritma Apriori dalam Menganalisa Perilaku Mahasiswa dalam Melilih Matakuliah (Studi Kasus : FKIP UPI "YPTK"). *Jurnal Media Processor*.

Dinkes 2010. Profil Kesehatan Kabupaten Polewali Mandar Dinas Kesehatan Kabupaten Polewali Mandar. Polewali.

Darman, Ridho. "Analisis Visualisasi dan Pemetaan Data Tanaman Padi di Indonesia Menggunakan Microsoft Power BI." *Jurnal Ilmiah Rekayasa dan Manajemen Sistem Informasi* 4.2 (2018): 156-162.

Fitriyanto, E. T. (2017). Penentuan Aturan Asosiasi pada Transaksi Penjualan Obat Menggunakan Algoritma Apriori Studi Kasus pada RSUD Dr. Soetrasno Rembang.

Herundika Cahyono Pratama, M. B. (2014). *Penerapan Algoritma Apriori Dalamm Menemukan Hubungan Data Awal Masuk Mahasiswa dengan Prestasi Akademik.* Tanjung Pinang: Universitas Maritim Raja Ali Haji (UMRAH)

Infodatin. 2019. “Peningkatan Gaya Hidup Sehat dengan Perilaku CERDIK.” *Pusat Data dan Informasi Kementerian Kesehatan RI*.

Kaufman L., and P.J. Rousseeuw.(1990). Finding Groups in Data. New York: John Wiley & Sons.

Listriani, D., Setyaningrum, A. H., & Eka, F. (2016). *PENERAPAN METODE ASOSIASI MENGGUNAKAN ALGORITMA APRIORI PADA APLIKASI ANALISA POLA BELANJA KONSUMEN (Studi Kasus Toko Buku Gramedia Bintaro)*. *9*(2).

Malik, R. A., Defit, S., & Yuhandri. (2018). PERBANDINGAN ALGORITMA K-MEANS CLUSTERING DENGAN FUZZY C-MEANS MENGUKUR TINGKAT KEPUASAN TERHADAP TELEVISI DAKWAH SURAU TV. *RABIT (Jurnal Teknologi Dan Sistem Informasi Univrab*, *3*(1), 10–21

Nasution, M. Z. (2019). PENERAPAN PRINCIPAL COMPONENT ANALYSIS (PCA) DALAM PENENTUAN FAKTOR DOMINAN YANG MEMPENGARUHI PRESTASI BELAJAR SISWA (Studi Kasus : SMK Raksana 2 Medan). *Jurnal Teknologi Informasi*, *3*(1).

Prasetyo, E. (2012). Data Mining Konsep dan Aplikasi Menggunakan Matlab. Andi Offset

Pei, J., Kamber, M., & Han, J. (2012). *Data Mining. Concepts and Techniques* (3rd ed.). Morgan Kaufmann..

Rustam, Z., Fitri, S. G., Selsi, R., & Pandelaki, J. (2019, December). The Global Kernel k-means Clustering Algorithm for Cerebral Infarction Classification. In *Journal of Physics: Conference Series* (Vol. 1417, No. 1, p. 012027). IOP Publishing.

Suryanto. (2019). Data Mining Untuk Klasifikasi dan Klasterisasi Data. Informatika

Saefuloh M, Wayunah. Analisis faktor yang berhubungan dengan kejadian *Stroke* di RSUD Indramayu.J Pendidik Keperawatan Indonesia. 2016:2(2):65–76

Simanjuntak, H. E., & Windarto. (2020). JURNAL MEDIA INFORMATIKA BUDIDARMA Analisa Data Mining Menggunakan Frequent Pattern Growth pada Data Transaksi Penjualan PT Mora Telematika Indonesia untuk Rekomendasi Strategi Pemasaran Produk Internet. *JURNAL MEDIA INFORMATIKA BUDIDARMA*, *4*(4). https://doi.org/10.30865/mib.v4i4.2300

Silitonga, P. (2016). Analisis Pola Penyebaran Penyakit Pasien Pengguna Badan Penyelenggara Jaminan Sosial (Bpjs) Kesehatan Dengan Menggunakan Metode Dbscan Clustering. *Jurnal Times*, *5*(1), 36-39.

scribd.com/doc/151484440/Kategori-Umur-Menurut-Depkes-RI

Wicaksono, D. D. (2015). Penerapan Algoritma Apriori untuk Informasi Saran Kueri Barang. 9-12.

Qolis, “Pemetaan dan analisa sebaran sekolah untuk peningkatan layanan pendidikan di kabupaten kediri dengan gis,” hal. 1–5, 2009
Inolabs, “Pengertian WebGIS,” 2016

.LAMPIRAN

Lampiran 1 Tabel Hasil dan perbedaan setiap klaster pada Pasien Penyakit *Stroke*

Cluster	0	1	2	3
Jumlah Pasien	114 Pasien	135 Pasien	109 Pasien	128 Pasien
Usia	Dewasa	Lansia	Lansia	Dewasa
Jenis Kelamin	L	P	L	P
Tekanan Darah	Normal (16) Tinggi (98)	Normal (5) Tinggi (130)	Normal (10) Tinggi (99)	Normal (14) Tinggi (114)
Kolestrol	Normal (1) Tinggi (113)	Normal (22) Tinggi (113)	Normal (10) Tinggi (99)	Normal (2) Tinggi (126)
Keluhan Utama	-Lemah Ekstremitas -Bicara Tidak Jelas -Kesadaran Menurun -Mual -Sakit Kepala -Sesak -Muntah -bibir tidak simetris -lemah anggota tubuh =9 Keluhan	-Lemah Ekstremitas -Bicara Tidak Jelas -Kesadaran Menurun -Tidak sadarkan diri -Mual -Sakit Kepala -Lemah Anggota Tubuh -sesak -Demam -Bibir tidak simetris -mulut mencong -muntah = 12 Keluhan	-Lemah ekstremitas -Sesak -Bicara Tidak Jelas -Kesadaran Menurun -Tidak sadarkan diri -Mual -Sakit Kepala -Lemah Anggota Tubuh -Mual -demam -Mulut mencong kekanan =11Keluhan	-Lemah Ekstremitas -Bicara Tidak Jelas -Kesadaran Menurun -Tidak sadarkan diri -Mual -Sakit Kepala -Lemah Anggota Tubuh -Demam -Sesak -Mulut Mencong ke kiri -Mulut Mencong kekanan = 12 Keluhan
Riwayat Penyakit	HT DM STROKE JANTUNG KOLESTROL	HT STROKE DM	HT STROKE DM VERTIGO KOLESTROL	KOLESTROL STROKE HT DM VERTIGO
Jenis Stroke	Hemoragik (25) Non Hemoragik (89)	Hemoragik (21) Non Hemoragik (114)	Hemoragik (19) Non Hemoragik (90)	Hemoragik (28) Non Hemoragik (100)

Lampiran 2 Listing Program Clustering

-Import Libraries

```
#modul yang digunakan
import numpy as np
import pandas as pd
import seaborn as sns

from matplotlib import pyplot as plt
import plotly.graph_objs as go
from datetime import datetime
import plotly.express as px
from sklearn_pandas import DataFrameMapper
from sklearn.preprocessing import LabelBinarizer
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn import metrics


from sklearn.metrics import silhouette_samples, silhouette_score
from mlxtend.frequent_patterns import apriori, association_rules
```

-Input Data

```
df = pd.read_csv("Data-pasien-7-Copy.csv")
pd.set_option('display.max_columns', None)
pd.set_option('display.max_row', None)

print(df.shape)
df.head()
```

-Mengubah data menjadi angka

```
# dataframe mulriple values in each column for permaslahan Umkm
# dataframe mulriple values in each column for permaslahan Umkm
from sklearn.preprocessing import MultiLabelBinarizer

mlb = {}
res = {}

for column in gejala.columns:
    mlb[column] = MultiLabelBinarizer()

    res[column] = pd.DataFrame(mlb[column].fit_transform(df[column].apply(lambda x: [j.strip() for j in x.split(",")])),
                               columns=mlb[column].classes_,
                               index=df[column].index)

arrays = [np.concatenate(([np.array([column]*len(mlb[column].classes_)) for column in df.columns])),
          np.concatenate(([mlb[column].classes_ for column in gejala.columns]))]

gejala_end = pd.DataFrame(columns = arrays, index = gejala.index)

for column in gejala.columns:
    gejala_end[column] = res[column]

gejala_end
```

-Menyederhanakan data dengan menggunakan PCA mentransformasikan linear sehingga terbentuk sistem koordinat baru dengan variasi maksimum.

```
pca = PCA(n_components=36)
pca.fit(result)
variance = pca.explained_variance_ratio_
print(variance)
var = np.cumsum(np.round(variance, 3)*100)
print(var)
plt.figure(figsize=(15,6))
plt.ylabel('% Variance Explained')
plt.xlabel('# of Features')
plt.title('PCA Analysis')
plt.ylim(0,100.5)
plt.plot(var)
```

```
pca = PCA(n_components=3).fit(result)
pca2 = pca.transform(result)
pd.set_option('display.max_row', None)

# print(pca2)
pca_df = pd.DataFrame(pca2)
print(pca_df.shape)
pca_df
```

```
def PCA_Scratch(X , num_components):

    #Step-1
    X_meaned = X - np.mean(X , axis = 0)

    #Step-2
    cov_mat = np.cov(X_meaned , rowvar = False)

    #Step-3
    eigen_values , eigen_vectors = np.linalg.eigh(cov_mat)

    #Step-4
    sorted_index = np.argsort(eigen_values)[::-1]
    sorted_eigenvalue = eigen_values[sorted_index]
    sorted_eigenvectors = eigen_vectors[:,sorted_index]

    #Step-5
    eigenvector_subset = sorted_eigenvectors[:,0:num_components]

    #Step-6
    X_reduced = np.dot(eigenvector_subset.transpose() , X_meaned.transpose() ).transpose()

    return X_reduced

#Applying it to PCA function
mat_reduced = PCA_Scratch(result , 3)

# Creating a Pandas DataFrame of reduced Dataset
pca_df = pd.DataFrame(mat_reduced , columns = [0,1,2])

#Concat it with target variable to create a complete Dataset
pca_df = pd.concat([pca_df] , axis = 1)
pca_df.head()
```

```
plt.figure(figsize = (12, 7))
sns.scatterplot(pca2[:,0], pca2[:,1],
                palette='Set1',
                s=100, alpha=0.2).set_title('Origin Dataset', fontsize=15)

plt.legend()
plt.ylabel('PCA2')
plt.xlabel('PCA1')
plt.show()
```

```
Scene = dict(xaxis = dict(title  = 'PC1'),yaxis = dict(title  = 'PC2'),zaxis = dict(title  = 'PC3'))
trace = go.Scatter3d(x=pca2[:,0], y=pca2[:,1], z=pca2[:,2], mode='markers',marker=dict(color = "black", colorscale='Viridis', si
layout = go.Layout(margin=dict(l=0,r=0),scene = Scene, height = 800,width = 800)

data = [trace]
fig = go.Figure(data = data, layout = layout)
fig.show()
```

```
from scipy import stats
import numpy as np

test_df = pca_df.copy()
z = np.abs(stats.zscore(test_df))
print(z)

threshold = 3
print(np.where(z > 3))
```

Lampiran 3 Algoritma Clustering K-Means

```
class KMeansClustering:

    '''Implementing Kmeans algorithm.'''

    def __init__(self, X, num_clusters):
        self.K = num_clusters
        self.plot_figure = True
        self.max_iterations = 100
        self.num_examples, self.num_features = X.shape

    def initialize_random_centroids(self,X):
        centroids = np.zeros((self.K, self.num_features))
        for k in range(self.K):
            centroid = X[np.random.choice(range(self.num_examples))]
            centroids[k] = centroid
#         print(centroids)
        return centroids


    def create_clusters(self, X, centroids):
        clusters = [[] for _ in range(self.K)]
        for point_idx, point in enumerate(X):
            closest_centroid = np.argmin(np.sqrt(np.sum((point-centroids)**2, axis=1)))
            clusters[closest_centroid].append(point_idx)
#         print(clusters)
        return clusters


    def calculate_new_centroids(self, clusters, X):
        centroids = np.zeros((self.K, self.num_features))
        for idx, cluster in enumerate(clusters):
            new_centroid = np.mean(X[cluster], axis=0)
            centroids[idx] = new_centroid
#         print(centroids)
        return centroids

    def predict_cluster(self, clusters, X):
        y_pred = np.zeros(self.num_examples)
        for cluster_idx, cluster in enumerate(clusters):
            for sample_idx in cluster:
                y_pred[sample_idx] = cluster_idx
#         print(cluster)
        return y_pred

    def fit(self, X):
        centroids = self.initialize_random_centroids(X)
        for it in range(self.max_iterations):
            clusters = self.create_clusters(X, centroids)
            prev_centroids = centroids
            centroids = self.calculate_new_centroids(clusters, X)
#             for idx, cluster in enumerate(clusters):
            diff = centroids - prev_centroids
            if not diff.any():
                break
        y_pred = self.predict_cluster(clusters, X)
#         print(centroids)
        return y_pred


    def predict(self, X):
        distance = self.compute_distance(X, self.centroids)
#         print(distance)
```

Lampiran 4 Listing Program Asosiasi

```
#memasukkan nilai min_support
freq_items2 = apriori(result_clus2, min_support = 0.70)
freq_items2.head()
```

```
D:\New folder (3)\lib\site-packages\mlxtend\frequent_patterns\fpcommon.py:111: DeprecationWarning:

DataFrames with non-bool types result in worse computationalperformance and their support might be discontinued in the future.P
lease use a DataFrame with bool type
```

```
# mengubah nilai integer menjadi nama item masing" menggunakan usecolname = true

freq_items2 = apriori(result_clus2, min_support=0.70, use_colnames=True, verbose=1)
freq_items2['length'] = freq_items2['itemsets'].apply(lambda x: len(x))
# freq_items.to_csv('large frequens.csv')
freq_items2
```

```
# antecedent (JIKA) and consequent (MAKA)

rules2 = association_rules(freq_items2, metric="confidence", min_threshold= 0.95)
# rules['length'] = rules['antecedents'].apply(lambda x: len(x))
pd.set_option('display.max_row', None)
rules2
```

Lampiran 5 Large –Itemset 1(L1)

(Kode Python pembentukan C1)

```
def createC1(data):
    C1 = []
    for transaction in data:
        for item in transaction:
            if not [item] in C1:
                C1.append([item])
    C1.sort()

    #membuat set setiap item di C1
    return [set(x) for x in C1]

C1 = createC1(trx)
print('C1 Items = (num={0})\n'.format(len(C1)))
C1
```

Lampiran 6 Kode Python pembentukan C1 dengan menampilkan nilai *support*

```
def scanD(data, Ck, minSupport):
    count = {}

    for transaction in data:
        tid = set(transaction)
        for candidate in Ck:
            if candidate.issubset(tid):
                can = frozenset(candidate)
                if can not in count:
                    count[can]=1
                else:
                    count[can] += 1
                    numItems = float(len(trx))
    L_List = []
    L_Support = {}
    C_Support = {}
    temp_Support = {}

    #menghitung support untuk setiap itemset

    for key in count:
        support = count[key]/numItems

        if support >= minSupport:
            L_List.insert(0, key)
            L_Support[key] = count[key], format(support, '.3f')
        C_Support[key] = count[key], format(support, '.3f')
        temp_Support[key] = support

    return L_List, L_Support, C_Support, temp_Support

minSupport = 0.70
L1, L1_support, C1_Support, temp_Support = scanD(trx, C1, minSupport)
print('C1 Items = (num={0})\n'.format(len(C1_Support)))
C1_Support
```

Lampiran 7 *Large-itemset* 2 (L2)

Kode *Python* pembentukan 2 atau lebih *items*

```
# Pembentukan kandidat 2-item C2
def createCk(Lk, k):
    L_List = []
    lenLk = len(Lk)
    for i in range(lenLk):
        for j in range(i+1, lenLk): 
            L1 = list(Lk[i])[:k-2]; L2 = list(Lk[j])[:k-2]
            L1.sort(); L2.sort()
            if L1==L2:
                L_List.append(Lk[i] | Lk[j])
    return L_List
C2 = createCk(L1, k=2)
print('C2 Items = (num={0})\n'.format(len(C2)))
C2
```

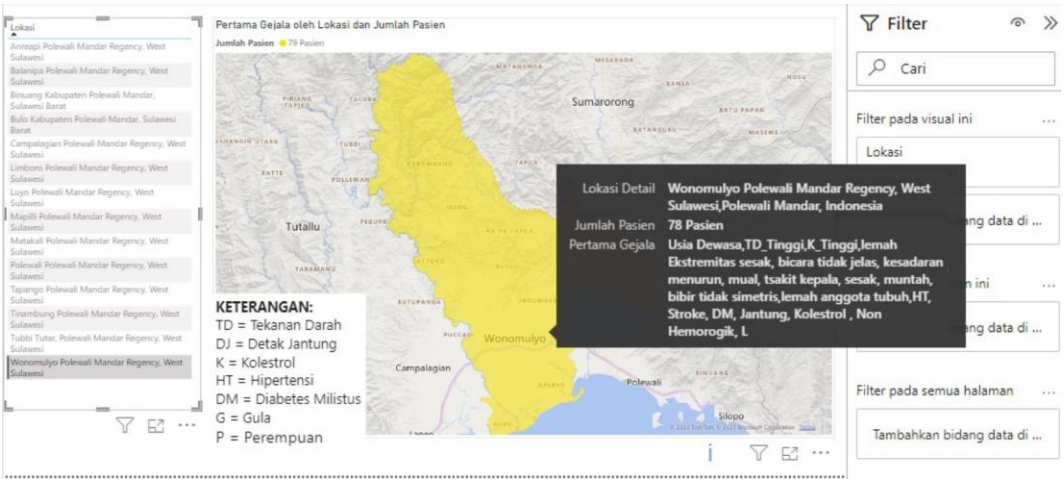
Lampiran 8 Code Python pembentukan *Frequent Itemset*

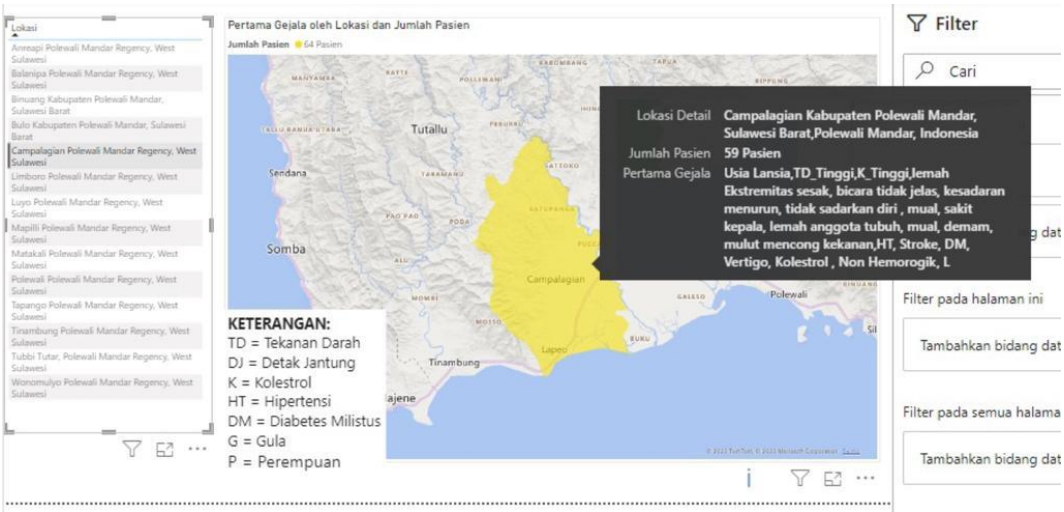
```
#pembentukan Frequent Itemset
def apriori(trx, minSupport):
    C1 = createC1(trx)
    D = list(map(set, trx))
    L1, L1_Support, C1_Support, temp_Support = scanD(D, C1, minSupport)
    L = [L1]

    k = 2
    while (len(L[k-2]) > 0):
        Ck = createCk(L[k-2], k)
        Lk, Lk_sup, Ck_sup, temp_sup = scanD(D, Ck, minSupport)
        temp_Support.update(temp_sup)
        L.append(Lk)
        k += 1
    return L, temp_Support
L, temp_Support = apriori(trx, minSupport)
```

Lampiran 9 Kode *Python* Pembentukan Aturan Asosiasi Dan Nilai *Confidence*

```python
# pembentukan aturan asosiasi dan penghitungan nilai confidence

minConf = 0.90

def generateRules(L, temp_Support, minConf):
    bigRuleList = []
    for i in range(1, len(L)):
        for freqSet in L[i]:
            H1 = [frozenset([item]) for item in freqSet]
            if (i > 1):
                rulesFromConseq(freqSet, H1, temp_Support, bigRuleList, minConf)
            else:
                calcConf(freqSet, H1, temp_Support, bigRuleList, minConf)
    return bigRuleList


def calcConf(freqSet, H, temp_Support, brl, minconf):
    prunedH = []
    for conseq in H:
        conf = temp_Support[freqSet]/temp_Support[freqSet-conseq]
        lift = temp_Support[freqSet]/(temp_Support[freqSet-conseq] * temp_Support[conseq])
        if (1-conf) !=0:
            conv = format((1-temp_Support[conseq])/(1-conf), '.3f')
        else:
            conv = 'inf'

        conf2 = temp_Support[freqSet]/temp_Support[conseq]
        lift2 = temp_Support[freqSet]/(temp_Support[freqSet-conseq] * temp_Support[conseq])
        if (1-conf2) !=0:
            conv2 = format((1-temp_Support[freqSet-conseq])/(1-conf2), '.3f')
        else:
            conv2 = 'inf'

        if conf >= minConf:
            print (freqSet-conseq, '-->', conseq, 'conf:', format(conf, '.3f'),
                   'supp:', format(temp_Support[freqSet], '.3f'), 'lift:',format(lift, '.3f'),
                   'conv:', conv)
            brl.append((freqSet-conseq, conseq, conf))
            prunedH.append(conseq)

            if conf2 >= minConf:
                if (len(conseq) > 1):
                    print (conseq, '-->', freqSet-conseq, 'conf:', format(conf2, '.3f'),
                           'Supp:', format(temp_Support[freqSet], '.3f'), 'Lift:',format(lift2, '.3f'),
                           'Conv:', conv2)
    return prunedH

def rulesFromConseq(freqSet, H, temp_Support, brl, minConf):
    m = len(H[0])
    if(len(freqSet) > (m + 1)):
        Hmp1 = createCk(H, m+1)
        Hmp1 = calcConf(freqSet, Hmp1, temp_Support, brl, minConf)
        if (len(Hmp1) > 1):
            rulesFromConseq(freqSet, Hmp1, temp_Support, brl, minConf)

print('Rules Items = {num=0}\n'.format(len(rules0)))
rules0= generateRules(L, temp_Support, minConf)
```

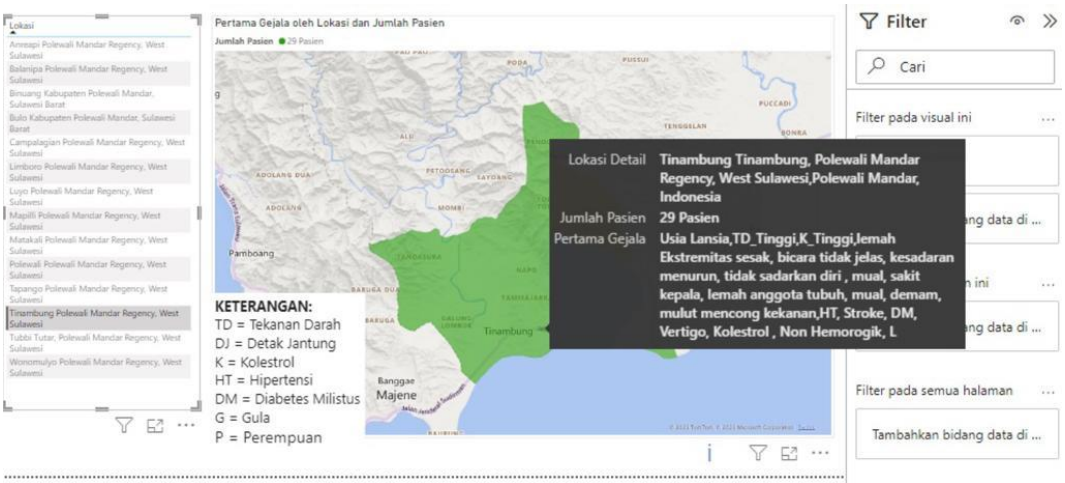
Lampiran 10 Hasil Visualisasi di beberapa kecamatan

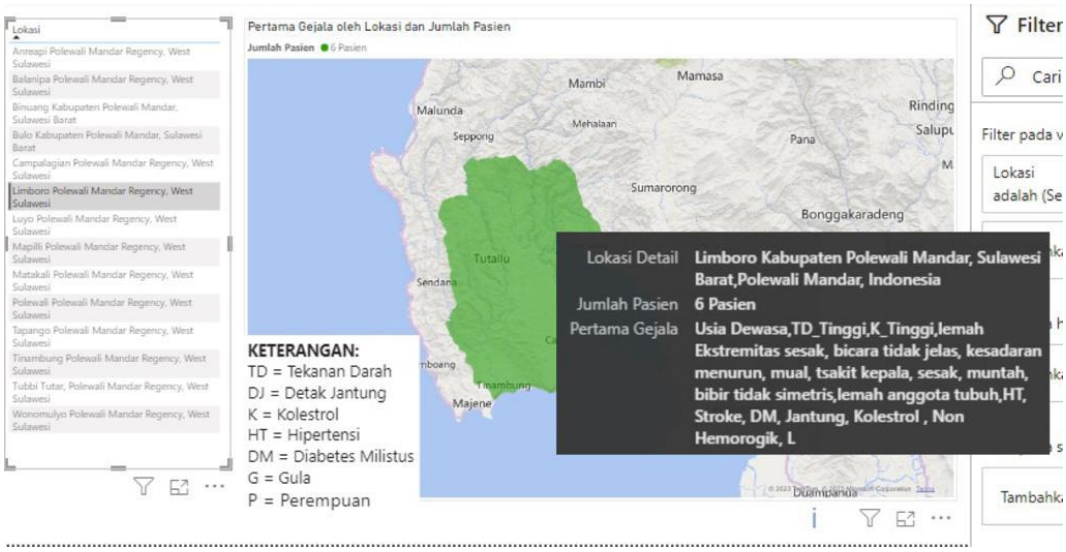





Lokasi
Pertama Gejala oleh Lokasi dan Jumlah Pasien
Jumlah Pasien 18 Pasien
Lokasi Detail Balanipa Kabupaten Polewali Mandar, Sulawesi Barat,Polewali Mandar, Indonesia
Jumlah Pasien 17 Pasien
Pertama Gejala Usia Lansia,TD_Tinggi,K_Tinggi,lemah Ekstremitas sesak, bicara tidak jelas, kesadaran menurun, tidak sadarkan diri , mual, sakit kepala, lemah anggota tubuh, mual, demam, mulut mencong kekanan,HT, Stroke, DM, Vertigo, Kolestrol , Non Hemorogik, L
KETERANGAN:
TD = Tekanan Darah
DJ = Detak Jantung
K = Kolestrol
HT = Hipertensi
DM = Diabetes Milistus
G = Gula
P = Perempuan
Filter
Cari
Filter pada halaman ini
Tambahkan bidang data di ...
Filter pada semua halaman
Tambahkan bidang data di ...


Lokasi
Pertama Gejala oleh Lokasi dan Jumlah Pasien
Jumlah Pasien 27 Pasien
Lokasi Detail Binuang Kabupaten Polewali Mandar, Sulawesi Barat,Polewali Mandar, Indonesia
Jumlah Pasien 26 Pasien
Pertama Gejala Usia Dewasa,TD_Tinggi,K_Tinggi,lemah Ekstremitas sesak, bicara tidak jelas, kesadaran menurun, mual, tsakit kepala, sesak, muntah, bibir tidak simetris,lemah anggota tubuh,HT, Stroke, DM, Jantung, Kolestrol , Non Hemorogik, L
KETERANGAN:
TD = Tekanan Darah
DJ = Detak Jantung
K = Kolestrol
HT = Hipertensi
DM = Diabetes Milistus
G = Gula
P = Perempuan
Filter
Cari


Lokasi
Pertama Gejala oleh Lokasi dan Jumlah Pasien
Jumlah Pasien 29 Pasien
Lokasi Detail Tinambung Tinambung, Polewali Mandar Regency, West Sulawesi,Polewali Mandar, Indonesia
Jumlah Pasien 29 Pasien
Pertama Gejala Usia Lansia,TD_Tinggi,K_Tinggi,lemah Ekstremitas sesak, bicara tidak jelas, kesadaran menurun, tidak sadarkan diri , mual, sakit kepala, lemah anggota tubuh, mual, demam, mulut mencong kekanan,HT, Stroke, DM, Vertigo, Kolestrol , Non Hemorogik, L
KETERANGAN:
TD = Tekanan Darah
DJ = Detak Jantung
K = Kolestrol
HT = Hipertensi
DM = Diabetes Milistus
G = Gula
P = Perempuan
Filter
Cari
Filter pada visual ini
Filter pada semua halaman
Tambahkan bidang data di ...


Lokasi
Anreapi Polewali Mandar Regency, West Sulawesi
Balanipa Polewali Mandar Regency, West Sulawesi
Binuang Kabupaten Polewali Mandar, Sulawesi Barat
Bulo Kabupaten Polewali Mandar, Sulawesi Barat
Campalagian Polewali Mandar Regency, West Sulawesi
Limboro Polewali Mandar Regency, West Sulawesi
Luyo Polewali Mandar Regency, West Sulawesi
Mapilli Polewali Mandar Regency, West Sulawesi
Matakali Polewali Mandar Regency, West Sulawesi
Polewali Polewali Mandar Regency, West Sulawesi
Tapango Polewali Mandar Regency, West Sulawesi
Tinambung Polewali Mandar Regency, West Sulawesi
Tubbi Tutar, Polewali Mandar Regency, West Sulawesi
Wonomulyo Polewali Mandar Regency, West Sulawesi
Pertama Gejala oleh Lokasi dan Jumlah Pasien
Jumlah Pasien 6 Pasien
Mambi
Mamasa
Malunda
Mehalaan
Seppong
Pana
Sumarorong
Bonggakaradeng
Tutallu
Sendana
Tinambung
Majene
Duampanua
Lokasi Detail
Limboro Kabupaten Polewali Mandar, Sulawesi Barat,Polewali Mandar, Indonesia
Jumlah Pasien
6 Pasien
Pertama Gejala
Usia Dewasa,TD_Tinggi,K_Tinggi,lemah Ekstremitas sesak, bicara tidak jelas, kesadaran menurun, mual, tsakit kepala, sesak, muntah, bibir tidak simetris,lemah anggota tubuh,HT, Stroke, DM, Jantung, Kolestrol , Non Hemorogik, L
KETERANGAN:
TD = Tekanan Darah
DJ = Detak Jantung
K = Kolestrol
HT = Hipertensi
DM = Diabetes Milistus
G = Gula
P = Perempuan
Filter
Cari
Filter pada v
Lokasi
adalah (Se
Tambahk