

Lampiran 1 Surat Permohonan Pengambilan Data



**UMKT**  
Program Studi  
**Teknik Informatika**  
Fakultas Sains dan Teknologi

Telp. 0541-748511 Fax. 0541-766832

Website <http://informatika.umkt.ac.id>

email: [informatika@umkt.ac.id](mailto:informatika@umkt.ac.id)



بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Nomor : 003-009/FST.1/D.3/C/2024

Lampiran : -

Perihal : **Permohonan Pengambilan Data**

**Kepada Yth.**  
**Kepala Dinas Kesehatan Kota Samarinda**  
di -

**Tempat**

*Assalamu'alaikum Warrahmatullahi Wabarrakatuh*

Puji Syukur kepada Allah Subhanahu wa ta'ala yang senantiasa melimpahkan Rahmat-Nya kepada kita sekalian. Aamiin.

Sehubungan untuk memenuhi Tugas Akhir/Skripsi Tahun Akademik 2023/2024, maka dengan ini kami bermaksud untuk melakukan pengambilan data di Dinas Kesehatan Kota Samarinda. Adapun data yang diminta yaitu data penyakit stunting di Kota Samarinda tahun 2023, dengan nama mahasiswa sebagai berikut:

No	Nama	NIM	Program Studi
1	Ari Ahmad Dhani	2011102441090	Teknik Informatika
2	Bima Satria	2011102441102	Teknik Informatika
3	Lidya Sari	2011102441121	Teknik Informatika
4	Mukminatul Munawaroh	2011102441064	Teknik Informatika
5	Siti Muawwanah	2011102441153	Teknik Informatika

Demikian surat permohonan ini dibuat. Atas perhatiannya dan kerjasamanya kami mengucapkan terima kasih.

*Wassalamu'alaikum Warrahmatullahi Wabarrakatuh*

Samarinda, 4 Ramadhan 1445 H  
15 Maret 2024 M

  
Program Studi S1 Teknik Informatika  
**Syahrudin, S.Kom., M.TI**  
NIDN. 1118019203













Lampiran 2 Tampilan Dataset Stunting Kota Samarinda

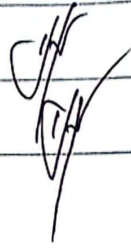
	Nama	J K	Bera t	Tingg i	LiL A	BB/U	ZS BB/U	Tanggal Pengukura n	ZS TB/U	BB/TB	ZS BB/ TB	Naik Berat Bada n	Jml Vit A	TB/U
1	DIMAS ADITYA	L	9.01		0	Kurang	-0.39	2023-01-02	-0.21	Gizi Baik	-0.39	O		Normal
2	SITI AISYAH	P	12	94	0		-2.25	2023-01-02	-2.09	Gizi Baik	-1.46	O		Pendek
3	M AL FATHI ALMAHIRA	L	8.01	69	0	Berat Badan Normal	-0.53	2023-01-02	-0.65	Gizi Baik	-0.14	O		Normal
4	AKIRA AKBAR GUINIA	P	6.03		0	Berat Badan Normal	-0.31	2023-01-02	0.42	Gizi Baik	-0.74	O		Normal
5	QAMELA	P	10.06		0	Risiko Lebih	0.090972222	2023-01-02	0.23	Gizi Lebih	0.09	O		Normal
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
150462	ADNAN IRAGUSTI	L	3	50		Berat Badan Normal	-0.73	2023-12-30	0.06	Gizi Baik	-1.19	-		Normal
150463	SIENA AL RAISHA	P	13		0	Berat Badan Normal	-1.22	2023-12-13	-1.11	Gizi Baik	-0.85	O		Normal
150464	AFIZAH KHAIRINA	P	2.05	45		Kurang	-2.03	2023-12-09	-2.63	Gizi Baik	-0.35	-		Normal
150465	MARSYA KHOLIF	P	3	49		Berat Badan Normal	-1.56	2023-12-19	-1.3	Gizi Baik	-1.04	-		Pendek
150466	MUHAMMAD IQBAL	L	2.09	49		Kurang	-2.97	2023-12-29	-2.48	Gizi Baik	-1.37	-		Normal

Lampiran 3 Kartu Bimbingan

**KARTU KENDALI BIMBINGAN LAPORAN KARYA ILMIAH**

Nama : Lidya Sari  
 NIM : 2011102441121  
 Nama Dosen Pembimbing : Taghfirul Azhima Yoga Siswa S.Kom., M.Kom.  
 Judul Penelitian : Model RFGS-CS Untuk Mengatasi *High Dimensional Data* Stunting Kota Samarinda

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	7 Feb 2024	Pembahasan perjanjian RTA, diskusi pemahaman tentang Data Science, Data Mining dan Machine Learning. Lalu diberi tugas untuk mengerjakan SLR Review Paper	
2	16 Feb 2024	Membahas hasil SLR Review dan diberi tugas untuk mengerjakan SLR Technical Paper	
3	22 Feb 2024	Membahas tentang SLR Technical dan diberi tugas untuk mencari paper/artikel rujukan sesuai topik riset	
4	5 Maret 2024	Membahas SLR Technical dan road maps penelitian	
5	6 Maret 2024	Penentuan judul penelitian	
6	8 Maret 2024	Membuat canvas penelitian untuk disubmit ke prodi	
7	13 Maret 2024	Membahas proposal penelitian dan diberi tugas untuk menulis bab 1 dan bab 2	
8	15 April 2024	Pengajuan surat permohonan data untuk penelitian	
9	18 April 2024	Revisi proposal bab 1, bab 2 dan perbaikan penulisan format skripsi	
10	24 Mei 2024	Revisi proposal bab 1, bab 2, untuk penyelesaian akhir sebelum submit	
11	10 Mei 2024	Menyusun pembahasan penelitian yang benar dan cara submit paper sinta	
12	16 Mei 2024	Konsultasi bab 3 dan bab 4, memperbaiki pembahasan dan kesimpulan	
13	13 Juni 2024	Mereview seluruh naskah skripsi serta publikasi dan memperbaiki naskah mulai dari abstrak hingga kesimpulan	

14	25 Juni 2024	Revisi abstrak dan bagian permodelan dalam naskah publikasi	
15	28 Juni 2024	Revisi naskah publikasi dan skripsi sebelum publish dan seminar hasil	

Dosen Pembimbing



Taghfirul Azhima Yoga Siswa  
S.Kom., M.Kom

Mengetahui



Ketua Program Studi

  
Atbansyah S.Kom., M.TI

## Lampiran 4 *Codingan Flow*

```
import pandas as pd

# Membaca data dari file CSV (ganti dengan lokasi file yang sesuai)
data = pd.read_csv('datasetfixstunting.csv')

# Menghitung jumlah data sebelum penghapusan duplikat
total_data_sebelum = len(data)

# Mengurutkan data berdasarkan kolom "Nama" dan "Tanggal Pengukuran" secara menurun
data = data.sort_values(by=['Nama', 'Tanggal Pengukuran'], ascending=[True, False])

# Menghapus duplikat berdasarkan kolom "Nama" dan mempertahankan yang pertama (yang memiliki tanggal pengukuran terbaru)
data = data.drop_duplicates(subset='Nama', keep='first')

# Menghitung jumlah data setelah penghapusan duplikat
total_data_sesudah = len(data)

# Menyimpan data yang telah dihapus duplikatnya ke file CSV baru (jika diperlukan)
data.to_csv('dataset_remove_duplikat.csv', index=False)

print(f"Total data sebelum penghapusan duplikat: {total_data_sebelum}")
print(f"Total data setelah penghapusan duplikat: {total_data_sesudah}")
```

```
dataset = pd.read_csv('dataset_remove_duplikat.csv')
dataset.info()
```

```
# Menyimpan hasilnya ke file CSV baru (jika diperlukan)
dataset.to_csv('dataset.csv', index=False)
```

```
dataset = pd.read_csv('dataset.csv')
dataset.info()
```

```
dataset['TB/U'].value_counts()
```

```

import pandas as pd

# Membaca data dari file CSV atau sumber data lainnya
data = pd.read_csv('dataset.csv')

# Menghitung jumlah data sebelum penghapusan
jumlah_data_sebelum = len(data)

# Menghapus baris yang memiliki setidaknya satu nilai yang hilang
data_tanpa_missing = data.dropna()

# Menghitung jumlah data setelah penghapusan
jumlah_data_sesudah = len(data_tanpa_missing)

# Menyimpan hasilnya ke file CSV baru (jika diperlukan)
data_tanpa_missing.to_csv('nama_file_tanpa_missing.csv', index=False)

# Mencetak jumlah data sebelum dan setelah penghapusan
print(f"Jumlah data sebelum penghapusan: {jumlah_data_sebelum}")
print(f"Jumlah data setelah penghapusan: {jumlah_data_sesudah}")

```

```

data_tanpa_missing['TB/U'].value_counts()

```

```

# Membuat kamus penggantian nilai
penggantian = {
    'Normal': 'Tidak Stunting',
    'Tinggi': 'Tidak Stunting',
    'Pendek': 'Stunting',
    'Sangat Pendek': 'Stunting'
}

# Mengganti nilai dalam kolom "TB/U" sesuai dengan kamus penggantian
data_tanpa_missing['TB/U'] = data_tanpa_missing['TB/U'].replace(penggantian)

# Menyimpan hasilnya ke file CSV baru (jika diperlukan)
data_tanpa_missing.to_csv('dataset_stunting_samarinda_modified.csv', index=False)

```



```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
def make_confusion_matrix(cf,
                          group_names=None,
                          categories='auto',
                          count=True,
                          percent=True,
                          cbar=True,
                          xyticks=True,
                          xyplotlabels=True,
                          sum_stats=True,
                          figsize=None,
                          cmap='Blues',
                          title=None):

    # CODE TO GENERATE TEXT INSIDE EACH SQUARE
    blanks = ['' for i in range(cf.size)]

    if group_names and len(group_names)==cf.size:
        group_labels = ["{}\n".format(value) for value in group_names]
    else:
        group_labels = blanks

    if count:
        group_counts = ["{0:0.0f}\n".format(value) for value in cf.flatten()]
    else:
        group_counts = blanks

    if percent:
        group_percentages = ["{0:.2%}".format(value) for value in cf.flatten()/np.sum(cf)]
    else:
        group_percentages = blanks
```

```

box_labels = [f"{v1}{v2}{v3}".strip() for v1, v2, v3 in zip(group_labels,group_counts,group_percentages)]
box_labels = np.asarray(box_labels).reshape(cf.shape[0],cf.shape[1])

# CODE TO GENERATE SUMMARY STATISTICS & TEXT FOR SUMMARY STATS
if sum_stats:
    #Accuracy is sum of diagonal divided by total observations
    accuracy = np.trace(cf) / float(np.sum(cf))

    #if it is a binary confusion matrix, show some more stats
    if len(cf)==2:
        #Metrics for Binary Confusion Matrices
        # precision = cf[1,1] / sum(cf[:,1])
        # recall = cf[1,1] / sum(cf[1,:])
        # f1_score = 2*precision*recall / (precision + recall)
        stats_text = "\n\nAccuracy={:0.3f}".format(accuracy)
    else:
        stats_text = "\n\nAccuracy={:0.3f}".format(accuracy)
else:
    stats_text = ""

# SET FIGURE PARAMETERS ACCORDING TO OTHER ARGUMENTS
if figsize==None:
    #Get default figure size if not set
    figsize = plt.rcParams.get('figure.figsize')

if xyticks==False:
    #Do not show categories if xyticks is False
    categories=False

```

```

# MAKE THE HEATMAP VISUALIZATION
plt.figure(figsize=figsize)
sns.heatmap(cf,annot=box_labels,fmt="", cmap=cmap,cbar=cbar,xticklabels=categories,yticklabels=categories)

if xyplotlabels:
    plt.ylabel('True label')
    plt.xlabel('Predicted label' + stats_text)
else:
    plt.xlabel(stats_text)

if title:
    plt.title(title)

```

```

import pandas as pd
import numpy as np
stunting = pd.read_csv('dataset_stunting_samarinda_modified.csv')
stunting = stunting.rename(columns={'TB/U': 'Kelas'})
stunting = stunting.drop('Nama', axis=1)
stunting = stunting.drop('Tanggal Pengukuran', axis=1)
stunting

```

```

stunting['Kelas'].value_counts()

```



```

#Transformasi Data
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OrdinalEncoder
ordinal = OrdinalEncoder()
labelencoder = LabelEncoder()

df_transform = stunting[['JK', 'Berat', 'Tinggi', 'LiLA', 'BB/U', 'ZS BB/U', 'ZS TB/U', 'Naik Berat Badan', 'BB/TB', 'ZS BB/TB', 'Kelas']]

stunting['JK'] = labelencoder.fit_transform(stunting['JK'])
stunting['BB/U'] = labelencoder.fit_transform(stunting['BB/U'])
stunting['BB/TB'] = labelencoder.fit_transform(stunting['BB/TB'])
stunting['Naik Berat Badan'] = labelencoder.fit_transform(stunting['Naik Berat Badan'])

df = stunting[['JK', 'Berat', 'Tinggi', 'LiLA', 'BB/U', 'ZS BB/U', 'ZS TB/U', 'Naik Berat Badan', 'BB/TB', 'ZS BB/TB', 'Kelas']]

```

```
df_transform
```

```

X = df.drop(['Kelas'],axis=1)
Y = df['Kelas']

```

```

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X)

```

```
X_normalized
```

```

from sklearn.feature_selection import chi2

X_normalized = df.drop(['Kelas'],axis=1)
y = df['Kelas']

X_abs = abs(X_normalized) # Mengambil nilai absolut dari data yang sudah dinormalisasi
chi_scores = chi2(X_abs, y)

nilai_chi = pd.Series(chi_scores[0],index = X_normalized.columns)
p_values = pd.Series(chi_scores[1],index = X_normalized.columns)
#p_values.sort_values(ascending = False , inplace = True)

p_values.plot.bar()

```

```
p_values.sort_values(ascending=False).plot.bar()
```

```
p_values.sort_values(ascending=False)
```

```
from sklearn.feature_selection import chi2

X_normalized = df.drop(['Kelas'],axis=1)
y = df['Kelas']

X_abs = abs(X_normalized) # Mengambil nilai absolut dari data yang sudah dinormalisasi
chi_scores = chi2(X_abs, y)

nilai_chi = pd.Series(chi_scores[0],index = X_normalized.columns)
p_values = pd.Series(chi_scores[1],index = X_normalized.columns)
#p_values.sort_values(ascending = False , inplace = True)

nilai_chi.plot.bar()
```

```
nilai_chi.sort_values(ascending=False).plot.bar()
```

```
nilai_chi.sort_values(ascending=False)
```

```
from sklearn.ensemble import RandomForestClassifier
X2 = df.drop(['Kelas', 'Naik Berat Badan', 'JK', 'BB/TB', ],axis=1)
clf = RandomForestClassifier(max_depth=2, random_state=42)
```

```

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_predict, KFold

clf = RandomForestClassifier(max_depth=2, random_state=42)
kf = KFold(n_splits=10, shuffle=True, random_state=42)

# Menyimpan akurasi setiap fold
fold accuracies = []

for train_index, test_index in kf.split(X2):
    X_train, X_test = X2.iloc[train_index], X2.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
    clf.fit(X_train, y_train)
    y_pred_fold = clf.predict(X_test)
    fold_accuracy = accuracy_score(y_test, y_pred_fold)
    fold accuracies.append(fold_accuracy)

# Rata-rata akurasi
mean_accuracy = sum(fold accuracies) / len(fold accuracies)

# Lakukan cross-validation untuk mendapatkan prediksi pada seluruh data
y_pred2 = cross_val_predict(clf, X2, y, cv=kf)

# Membuat confusion matrix
cf_matrix = confusion_matrix(y, y_pred2)

# Fungsi untuk menampilkan confusion matrix
def make_confusion_matrix(cf_matrix, figsize=(8,6), cbar=False, title="Confusion Matrix"):
    plt.figure(figsize=figsize)
    sns.heatmap(cf_matrix, annot=True, fmt='d', cmap='Blues', cbar=cbar)
    plt.title(title)
    plt.ylabel('Actual')
    plt.xlabel('Predicted')
    plt.show()

# Menampilkan confusion matrix
make_confusion_matrix(cf_matrix, title="Model RF + Chi Square (K-Fold = 10)")

# Menampilkan classification report
print(classification_report(y, y_pred2, labels=['Stunting', 'Tidak Stunting']))

# Menampilkan hasil akurasi dari fold 1 sampai 10
for i, accuracy in enumerate(fold accuracies):
    print(f"Fold {i+1} Accuracy: {accuracy:.4f}")

# Menampilkan rata-rata akurasi
print(f"Mean Accuracy: {mean_accuracy:.4f}")

```

```

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV, KFold, cross_val_predict

# Inisialisasi model Random Forest
clf = RandomForestClassifier(random_state=42)

# Tentukan kumpulan parameter yang ingin diuji
param_grid = {'max_depth': [2]}

# Inisialisasi KFold dengan K=10
kf = KFold(n_splits=10, shuffle=True, random_state=42)

# Inisialisasi Grid Search dengan model, parameter grid, dan jumlah lipatan validasi silang (CV)
grid_search = GridSearchCV(estimator=clf, param_grid=param_grid, cv=kf, n_jobs=-1)

# Lakukan pencarian grid pada data
grid_search.fit(X2, y)

# Model terbaik dari Grid Search
best_model = grid_search.best_estimator_

# Menyimpan akurasi setiap fold
fold accuracies = []

for train_index, test_index in kf.split(X2):
    X_train, X_test = X2.iloc[train_index], X2.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
    best_model.fit(X_train, y_train)
    y_pred_fold = best_model.predict(X_test)
    fold_accuracy = accuracy_score(y_test, y_pred_fold)
    fold accuracies.append(fold_accuracy)

# Rata-rata akurasi
mean_accuracy = sum(fold accuracies) / len(fold accuracies)

# Lakukan cross-validation untuk mendapatkan prediksi pada seluruh data
y_pred = cross_val_predict(best_model, X2, y, cv=kf)

```

```

# Membuat confusion matrix
cf_matrix = confusion_matrix(y, y_pred)

# Fungsi untuk menampilkan confusion matrix
def make_confusion_matrix(cf_matrix, figsize=(8,6), cbar=False, title="Confusion Matrix"):
    plt.figure(figsize=figsize)
    sns.heatmap(cf_matrix, annot=True, fmt='d', cmap='Blues', cbar=cbar)
    plt.title(title)
    plt.ylabel('Actual')
    plt.xlabel('Predicted')
    plt.show()

# Memanggil fungsi untuk menampilkan confusion matrix
make_confusion_matrix(cf_matrix, title="Model RF + Chi Square + Grid Search (K-Fold = 10)")

# Menampilkan classification report
print(classification_report(y, y_pred, labels=['Stunting', 'Tidak Stunting']))

# Menampilkan hasil akurasi dari fold 1 sampai 10
for i, accuracy in enumerate(fold accuracies):
    print(f"Fold {i+1} Accuracy: {accuracy:.4f}")

# Menampilkan rata-rata akurasi
print(f"Mean Accuracy: {mean_accuracy:.4f}")

```

## RIWAYAT HIDUP



Lidya Sari, atau biasa disapa dengan sebutan Dya, lahir di Samarinda tepatnya di daerah Loa Buah pada 09 September 2002, Penulis merupakan anak pertama dari Bapak Sumarmin(Alm) dan Ibu Harnanik. Menempuh pendidikan di SDN 009 Tenggarong Seberang tahun 2008 kemudian 2014, SMPN 1 Tenggarong Seberang tahun 2014 – 2017, SMKN 3 Tenggarong tahun 2017 – 2020. Kemudian penulis tercatat sebagai mahasiswa pada perguruan tinggi swasta Universitas Muhammadiyah Kalimantan Timur pada Fakultas Sains Dan Teknologi jurusan Teknik informatika pada tahun 2020. Saat menjadi mahasiswa penulis pernah melaksanakan perogram magang di Dinas Perdagangan Kota Samarinda selama 2 bulan yang dilaksanakan pada semester 7. Demikian deskripsi riwayat hidup

yang penulis sampaikan jika terdapat kesalahan atau kekurangan mohon dimaafkan karena kesempurnaan hanya milik Sang Maha pencipta, maka penulis mengharapkan kritik dan saran mengenai skripsi ini.