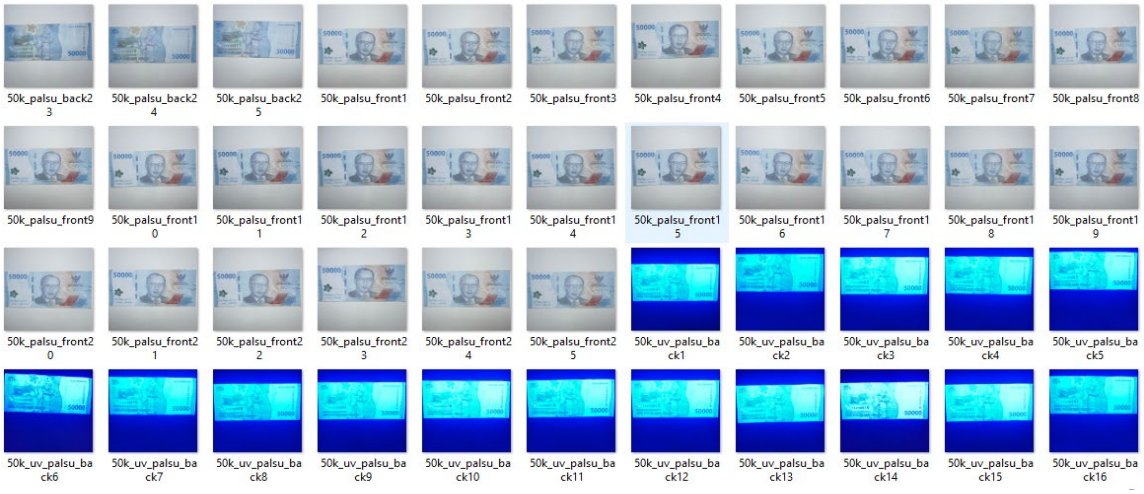
LAMPIRAN

Lampiran 1.
Data Citra Uang Kertas Rupiah 20.000 Asli

Lampiran 2.
Data Citra Uang Kertas Rupiah 20.000 Palsu

Lampiran 3.
Data Citra Uang Kertas Rupiah 50.000 Asli

Lampiran 4.
Data Citra Uang Kertas Rupiah 50.000 Palsu



Lampiran 5.
Data Citra Uang Kertas Rupiah 100.000 Asli

Lampiran 6.
Data Citra Uang Kertas Rupiah 100.000 Palsu

Lampiran 7.
Kode Pemrograman algoritma CNN menggunakan *python* pada *google colaboratory*

CNN (Final).ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect

```
[ ] # CONNECT GOOGLE DRIVE
    from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[ ] import os
```

```
[ ] set_1 = os.path.join('/content/drive/My Drive/Uang/train/20.000_asli')
    set_2 = os.path.join('/content/drive/My Drive/Uang/train/20.000_palsu')
    set_3 = os.path.join('/content/drive/My Drive/Uang/train/50.000-asli')
    set_4 = os.path.join('/content/drive/My Drive/Uang/train/50.000_palsu')
    set_5 = os.path.join('/content/drive/My Drive/Uang/train/100.000_asli')
    set_6 = os.path.join('/content/drive/My Drive/Uang/train/100.000_palsu')
```

```
[ ] set_7 = os.path.join('/content/drive/My Drive/Uang/test/20.000_asli')
    set_8 = os.path.join('/content/drive/My Drive/Uang/test/20.000_asli')
    set_9 = os.path.join('/content/drive/My Drive/Uang/test/50.000_asli')
    set_10 = os.path.join('/content/drive/My Drive/Uang/test/50.000_palsu')
    set_11 = os.path.join('/content/drive/My Drive/Uang/test/100.000_asli')
    set_12 = os.path.join('/content/drive/My Drive/Uang/test/100.000_asli')
```

```
[ ] print ('total data train uang 20.000 asli:', len(os.listdir(set_1)))
    print ('total data train uang 20.000 palsu:', len(os.listdir(set_2)))
    print ('total data train uang 50.000 asli:', len(os.listdir(set_3)))
    print ('total data train uang 50.000 palsu:', len(os.listdir(set_4)))
    print ('total data train uang 100.000 asli:', len(os.listdir(set_5)))
    print ('total data train uang 100.000 palsu:', len(os.listdir(set_6)))
    #train test)
    print('\n')
    print ('total data test uang 20.000 asli:', len(os.listdir(set_7)))
    print ('total data test uang 20.000 palsu:', len(os.listdir(set_8)))
    print ('total data test uang 50.000 asli:', len(os.listdir(set_9)))
```

CNN (Final).ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect

```
[ ] total data test uang 20.000 asli: 20
    total data test uang 20.000 palsu: 20
    total data test uang 50.000 asli: 20
    total data test uang 50.000 palsu: 20
    total data test uang 100.000 asli: 20
    total data test uang 100.000 palsu: 20
```

```
[ ] from keras.preprocessing.image import ImageDataGenerator
```

```
    train_datagen = ImageDataGenerator(rescale=1./255
                                       )
    valid_datagen = ImageDataGenerator(rescale=1./255
                                       )
```

```
[ ] train_generator = train_datagen.flow_from_directory('/content/drive/My Drive/Uang/train',
                                                        batch_size = 32,
                                                        target_size=(299,299),
                                                        class_mode="categorical",
                                                        seed=46)
    validation_generator = valid_datagen.flow_from_directory('/content/drive/My Drive/Uang/test',
                                                             batch_size = 32,
                                                             target_size=(299,299),
                                                             class_mode="categorical",
                                                             seed=46)
```

```
Found 480 images belonging to 6 classes.
Found 120 images belonging to 6 classes.
```

CNN (Final).ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect

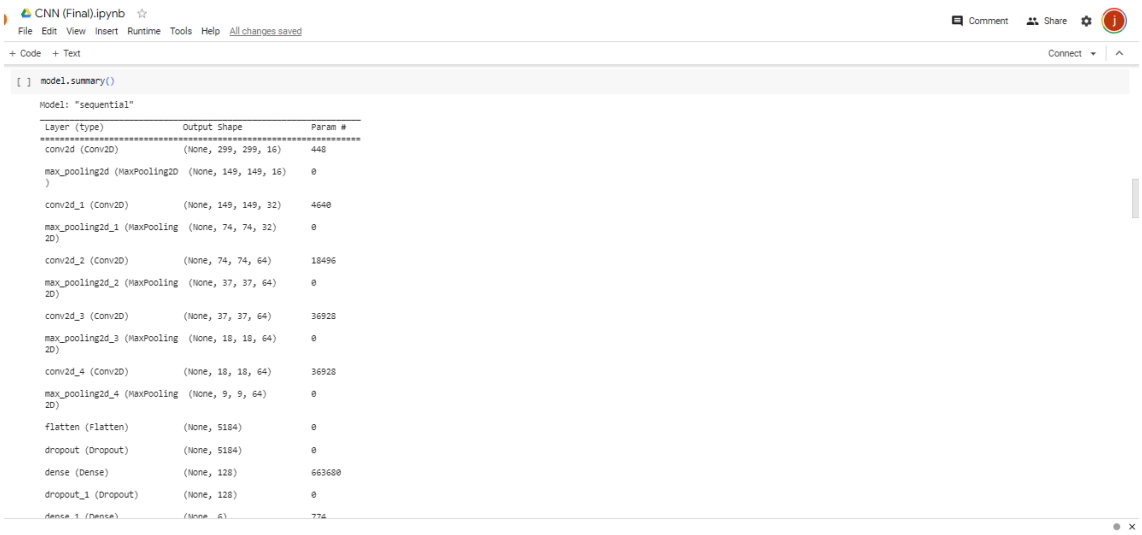
```
Found 480 images belonging to 6 classes.
Found 120 images belonging to 6 classes.
```

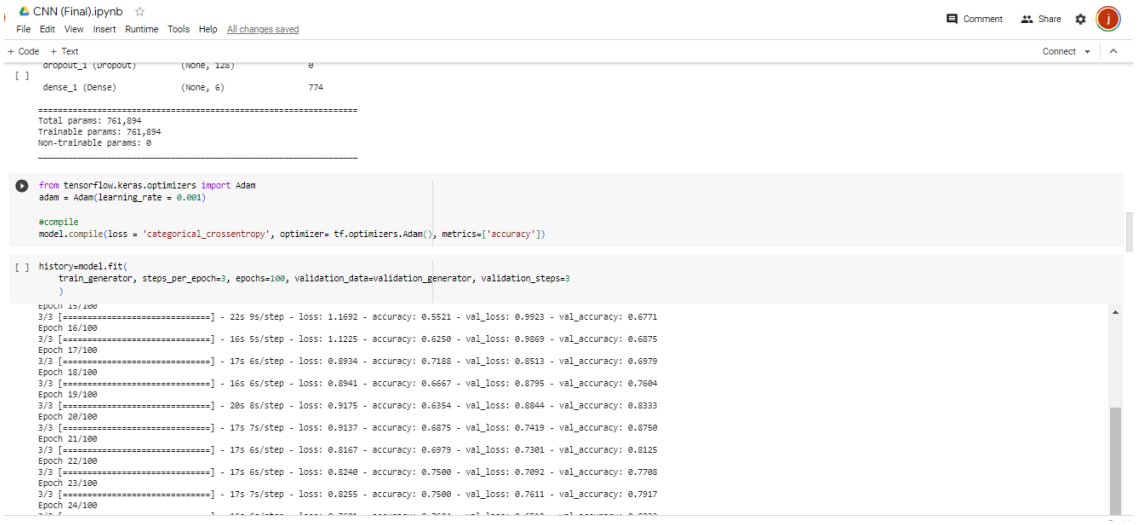
```
[ ] label_map=(train_generator.class_indices)
    print(label_map)
```

```
{'100.000_asli': 0, '100.000_palsu': 1, '20.000_asli': 2, '20.000_palsu': 3, '50.000-asli': 4, '50.000_palsu': 5}
```

```
[ ] import tensorflow as tf
    from tensorflow.keras import layers
    from tensorflow.keras import regularizers
```

```
    model = tf.keras.models.Sequential([
        #konvolusi pertama
        tf.keras.layers.Conv2D(16, (3,3), activation='relu', padding='same', kernel_initializer='he_normal', input_shape=(299, 299, 3)),
        tf.keras.layers.MaxPooling2D(2, 2),
        #konvolusi kedua
        tf.keras.layers.Conv2D(32, (3,3), activation='relu', padding='same', kernel_initializer='he_normal'), tf.keras.layers.MaxPooling2D(2,2),
        #konvolusi ketiga
        tf.keras.layers.Conv2D(64, (3,3), activation='relu', padding='same', kernel_initializer='he_normal'), tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(64, (3,3), activation='relu', padding='same', kernel_initializer='he_normal'), tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(64, (3,3), activation='relu', padding='same', kernel_initializer='he_normal'), tf.keras.layers.MaxPooling2D(2,2),
        #flatten dropout
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dropout(0.3),
        #fully connected layers
        tf.keras.layers.Dense(128, activity_regularizer=regularizers.l2(0.001), activation='relu'),
        tf.keras.layers.Dropout(0.6),
        tf.keras.layers.Dense(6, activation='softmax')
    ])
```


CNN (Final).ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Connect
[ ] model.summary()
Model: "sequential"
Layer (type) Output Shape Param #
conv2d (Conv2D) (None, 299, 299, 16) 448
max_pooling2d (MaxPooling2D) (None, 149, 149, 16) 0
conv2d_1 (Conv2D) (None, 149, 149, 32) 4640
max_pooling2d_1 (MaxPooling2D) (None, 74, 74, 32) 0
conv2d_2 (Conv2D) (None, 74, 74, 64) 18496
max_pooling2d_2 (MaxPooling2D) (None, 37, 37, 64) 0
conv2d_3 (Conv2D) (None, 37, 37, 64) 36928
max_pooling2d_3 (MaxPooling2D) (None, 18, 18, 64) 0
conv2d_4 (Conv2D) (None, 18, 18, 64) 36928
max_pooling2d_4 (MaxPooling2D) (None, 9, 9, 64) 0
flatten (Flatten) (None, 5184) 0
dropout (Dropout) (None, 5184) 0
dense (Dense) (None, 128) 663680
dropout_1 (Dropout) (None, 128) 0


CNN (Final).ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Connect
[ ]
dense_1 (Dense) (None, 6) 774
Total params: 761,894
Trainable params: 761,894
Non-trainable params: 0
from tensorflow.keras.optimizers import Adam
adam = Adam(learning_rate = 0.001)
#compile
model.compile(loss = 'categorical_crossentropy', optimizer= tf.optimizers.Adam(), metrics=['accuracy'])
[ ] history=model.fit(
train_generator, steps_per_epoch=3, epochs=100, validation_data=validation_generator, validation_steps=3
)
3/3 [==========] - 22s 9s/step - loss: 1.1692 - accuracy: 0.5521 - val_loss: 0.9923 - val_accuracy: 0.6771
Epoch 16/100
3/3 [==========] - 16s 5s/step - loss: 1.1225 - accuracy: 0.6250 - val_loss: 0.9869 - val_accuracy: 0.6875
Epoch 17/100
3/3 [==========] - 17s 6s/step - loss: 0.8934 - accuracy: 0.7188 - val_loss: 0.8513 - val_accuracy: 0.6979
Epoch 18/100
3/3 [==========] - 16s 6s/step - loss: 0.8941 - accuracy: 0.6667 - val_loss: 0.8795 - val_accuracy: 0.7604
Epoch 19/100
3/3 [==========] - 20s 8s/step - loss: 0.9175 - accuracy: 0.6354 - val_loss: 0.8844 - val_accuracy: 0.8333
Epoch 20/100
3/3 [==========] - 17s 7s/step - loss: 0.9137 - accuracy: 0.6875 - val_loss: 0.7419 - val_accuracy: 0.8750
Epoch 21/100
3/3 [==========] - 17s 6s/step - loss: 0.8167 - accuracy: 0.6979 - val_loss: 0.7301 - val_accuracy: 0.8125
Epoch 22/100
3/3 [==========] - 17s 6s/step - loss: 0.8240 - accuracy: 0.7500 - val_loss: 0.7092 - val_accuracy: 0.7708
Epoch 23/100
3/3 [==========] - 17s 7s/step - loss: 0.8255 - accuracy: 0.7500 - val_loss: 0.7611 - val_accuracy: 0.7917
Epoch 24/100

(alternative) CNN Final.ipynb

File Edit View Insert Runtime Tools Help Last saved at 6:12 AM

Comment Share

+ Code + Text

Connect

```
import tensorflow as tf
tf.keras.models.save_model(model, 'model_uang10.hdf5')
```

```
import matplotlib.pyplot as plt
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
```

```
plt.figure(figsize=(8, 8))
plt.subplot(2, 1, 1)
plt.plot(acc, label='training accuracy')
plt.plot(val_acc, label='validation accuracy')
plt.legend(loc='lower right')
plt.ylabel('accuracy')
plt.title('training and validation accuracy')
plt.show()
```

CNN (Final).ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect

```
model.evaluate(validation_generator)
```

```
4/4 [==============================] - 5s 1s/step - loss: 0.1371 - accuracy: 0.9833
[0.1370515376329422, 0.9833333492279053]
```

```
from sklearn.metrics import classification_report
```

```
test_generator = ImageDataGenerator(rescale=1./255)
test_data_generator=test_generator.flow_from_directory('/content/drive/My Drive/Uang/test',
                                                       target_size = (299,299), batch_size=30, shuffle=False)
```

```
Found 120 images belonging to 6 classes.
```

```
import sklearn
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns

for x in label_map:
      print(label_map[x],":",x)

prediksi=model.predict_generator(test_data_generator)
y_pred=np.argmax(prediksi,axis=1)

plt.figure(figsize=(8,8))
sns.heatmap(confusion_matrix(test_data_generator.classes,y_pred),annot=True)
plt.title("Confusion Matrix")
plt.ylabel("Aktual")
plt.xlabel("Prediksi")
```

CNN (Final).ipynb

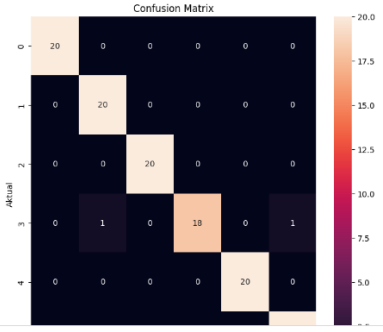
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

Connect

```
0 : 100.000_asli
1 : 100.000_palsu
2 : 20.000_asli
3 : 20.000_palsu
4 : 50.000-asli
5 : 50.000_palsu
<ipython-input-36-154944531f81>:10: UserWarning: `Model.predict_generator` is deprecated and will be removed in a future version. Please use `Model.predict`, which supports generators.
  prediksi=model.predict_generator(test_data_generator)
Text(0.5, 58.7222222222222, 'Prediksi')
```



Lampiran 8.
Kode Pemrograman algoritma SVM
menggunakan *python* pada *google colaboratory*

Final SVM Classification .ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Code + Text

Connect

Load Modules

```
[ ] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import accuracy_score
```

Prepare/collect data

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[ ] import os

    path = os.listdir('/content/drive/MyDrive/Data_uang/ready')
    classes = {'100.000 Asli':0, '100.000 Palsu':1,'20.000 Asli':2,'20.000 Palsu':3,'50.000 Asli':4,'50.000 Palsu':5,}
```

```
[ ] import pandas as pd
    from sklearn import svm
    from sklearn.model_selection import GridSearchCV
    import os
    import matplotlib.pyplot as plt
    from skimage.transform import resize
    from skimage.io import imread
    import numpy as np
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import classification_report,accuracy_score,confusion_matrix
```

Final SVM Classification Ver 1.ipynb

File Edit View Insert Runtime Tools Help

Comment Share

+ Code + Text

Connect

Feature Scaling = minmax scaler

```
[ ] x_train = x_train/255
    x_test = x_test/255
    print(x_train.max(), x_train.min())
    print(x_test.max(), x_test.min())
```

Dimensional Reduction: PCA

```
[ ] from sklearn.decomposition import PCA
```

```
[ ] print(x_train.shape, x_test.shape)

    pca = PCA(.98)

    pca_train = x_train
    pca_test = x_test
```

```
    (480, 268203) (120, 268203)
```

```
[ ] # print(pca_train.shape, pca_test.shape)
    # print(pca.n_components_)
    # print(pca.n_features_)
```

Train Model

```
from sklearn.svm import SVC
```

```
[ ] sv = SVC()
    sv.fit(x_train, y_train)
```

```
SVC
SVC()
```

Final SVM Classification Ver 1.ipynb

File Edit View Insert Runtime Tools Help

Code + Text

Comment Share Connect

```
flat_data_arr=[]
target_arr=[]
#please use datadir='/content' if the files are upload on to google collab
#else mount the drive and give path of the parent-folder containing all category images folders.
datadir='/content/drive/MyDrive/Data_uang/ready'
for i in classes:
  print(f'loading... category : {i}')
  path=os.path.join(datadir,i)
  for img in os.listdir(path):
    img_array=imread(os.path.join(path,img))
    img_resized=resize(img_array,(299,299,3))
    flat_data_arr.append(img_resized.flatten())
    target_arr.append(classes[i])
  print(f'loaded category:{i} successfully')
```

```
loading... category : 100.000 Asli
loaded category:100.000 Asli successfully
loading... category : 100.000 Palsu
loaded category:100.000 Palsu successfully
loading... category : 20.000 Asli
loaded category:20.000 Asli successfully
loading... category : 20.000 Palsu
loaded category:20.000 Palsu successfully
loading... category : 50.000 Asli
loaded category:50.000 Asli successfully
loading... category : 50.000 Palsu
loaded category:50.000 Palsu successfully
```

Split Data

```
flat_data=np.array(flat_data_arr)
target=np.array(target_arr)
df=pd.DataFrame(flat_data)
df['Target']=target
x=df.iloc[:,:-1]
y=df.iloc[:,-1]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=42,stratify=y)
print('Splitted Successfully')
```

```
Splitted Successfully
```

Final SVM Classification Ver 1.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Code + Text

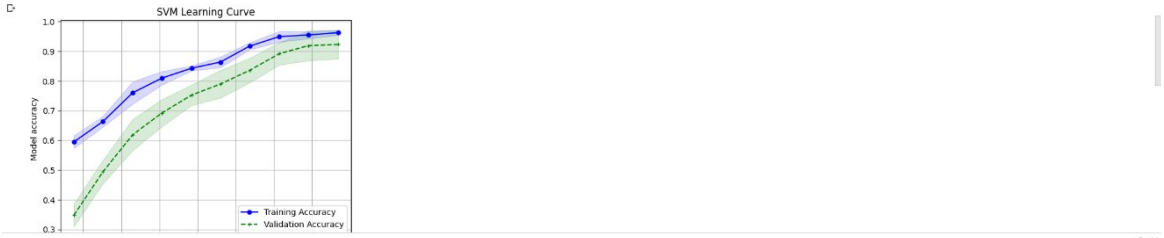
Comment Share Connect

```
print(train_sizes)
```

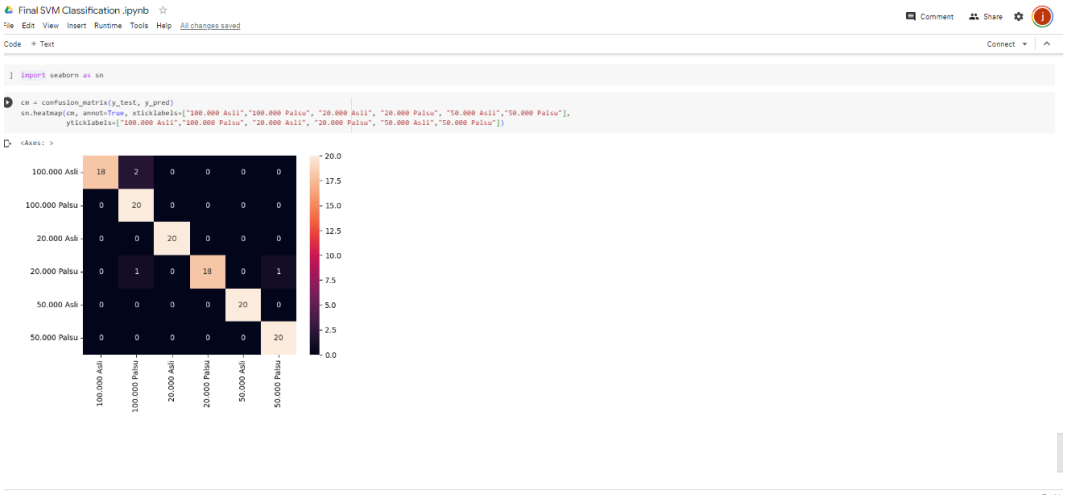
```
[ 38  76 115 153 192 230 268 307 345 384]
```

```
train_mean = np.mean(train_scores, axis=1)
train_std = np.std(train_scores, axis=1)
test_mean = np.mean(test_scores, axis=1)
test_std = np.std(test_scores, axis=1)
```

```
plt.plot(train_sizes, train_mean, color='blue', marker='o', markersize=5, label='Training Accuracy')
plt.fill_between(train_sizes, train_mean + train_std, train_mean - train_std, alpha=0.15, color='blue')
plt.plot(train_sizes, test_mean, color='green', marker='+', markersize=5, linestyle='--', label='Validation Accuracy')
plt.fill_between(train_sizes, test_mean + test_std, test_mean - test_std, alpha=0.15, color='green')
plt.title('SVM Learning Curve')
plt.xlabel('Training Data Size')
plt.ylabel('Model accuracy')
plt.grid()
plt.legend(loc='lower right')
plt.show()
```




Final SVM Classification Ver 1.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
Code + Text
Connect
sv = SVC()
sv.fit(x_train, y_train)
SVC()
Evaluation
print("Training Score:", sv.score(x_train, y_train))
print("Testing Score:", sv.score(x_test, y_test))
Training Score: 0.9729166666666667
Testing Score: 0.9666666666666667
import pickle
filename = 'SVM.h5'
pickle.dump(sv, open(filename, 'wb'))
y_pred = sv.predict(x_test)
import numpy as np
import pandas as pd
from pandas import Series,DataFrame
import matplotlib
from matplotlib import pyplot as plt
%matplotlib inline
train_sizes, train_scores, test_scores = learning_curve(estimator=sv, X=x_train, y=y_train,
cv=5, train_sizes=np.linspace(0.1, 1.0, 10),
n_jobs=-1)
print(train_sizes)


Final SVM Classification .ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
Code + Text
Connect
import seaborn as sn
cm = confusion_matrix(y_test, y_pred)
sn.heatmap(cm, annot=True, xticklabels=["100.000 Asli","100.000 Palsu", "20.000 Asli", "20.000 Palsu", "50.000 Asli","50.000 Palsu"],
yticklabels=["100.000 Asli","100.000 Palsu", "20.000 Asli", "20.000 Palsu", "50.000 Asli","50.000 Palsu"])
<Axes: >
100.000 Asli
100.000 Palsu
20.000 Asli
20.000 Palsu
50.000 Asli
50.000 Palsu
18 2 0 0 0 0
0 20 0 0 0 0
0 0 20 0 0 0
0 1 0 18 0 1
0 0 0 0 20 0
0 0 0 0 0 20
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0

Lampiran 9.
Lembar Konsultasi Dosen Pembimbing

UPN "VETERAN JAKARTA

LEMBAR KONSULTASI PEMBIMBING TUGAS AKHIR
MAHASISWA FAKULTAS TEKNIK

Nama : Jonathan Andrew Pandapotan Simarmata

NIM : 1910314017

PROGRAM STUDI : S1 Teknik Elektro

JUDUL TUGAS AKHIR : Analisis Performansi metode Convolutional Neural Networks (CNN) dan Support Vector Machine (SVM) dalam mendeteksi uang kertas rupiah emisi tahun 2022

PEMBIMBING : Ibu Fajar Rahayu, S.T., M.T.

No	HARI	TANGGAL	MATERI KONSULTASI	PARAF PEMBIMBING
1	Rabu	13 April 2022	Diskusi & Pengajuan Bab 1	
2	Selasa	7 Juni 2022	Diskusi format & Pengajuan Bab 2	
3	Jumat	10 Juni 2022	Diskusi & Pengajuan Bab 3	
4	Senin	13 Juni 2022	Diskusi & Pengajuan Revisi Bab 3	
5	Selasa	21 Juni 2022	finalisasi Proposal	
6	Senin	19 Juni 2023	Pembahasan Sistematika	
7	Selasa	20 Juni 2023	Diskusi penulisan bab 4	
8	Jumat	25 Juni 2023	Diskusi penulisan bab 5	
9	Senin	26 Juni 2023	Finalisasi draft keseluruhan	
10	Selasa	27 Juni 2023	finalisasi Skripsi	

Keterangan :
1. Setiap konsultasi harus diparaf Dosen Pembimbing
2. Minimal jumlah konsultasi sebanyak 8 kali pertemuan
3. Penyelesaian TA/Skripsi paling cepat 3 (tiga) bulan dan paling lambat 6 (enam) bulan sejak surat pengantar dikeluarkan oleh sekretariat Fakultas

Jakarta, 3 Juli 2023
Kepala Program Studi

(Achmad Z.P.)

Formulir. 26

FAKULTAS TEKNIK
UPN"VETERAN"JAKARTA

LEMBAR KONSULTASI PEMBIMBING TUGAS AKHIR
MAHASISWA FAKULTAS TEKNIK

Nama : Jonathan Andrew Pandapotan Simarmata

NIM : 1910314017

PROGRAM STUDI : S1 Teknik Elektro

JUDUL TUGAS AKHIR : Analisis Performansi metode CNN dan SVM dalam Mendeteksi keaslian dan anomali Citra uang kertas rupiah emisi tahun 2022

PEMBIMBING : Bapak Achmad Zuchriadi, ST, MT

No	HARI	TANGGAL	MATERI KONSULTASI	PARAF PEMBIMBING
1	Rabu	13 April 2022	Diskusi & Pengajuan Bab 1	
2	Selasa	7 Juni 2022	Diskusi format & Pengajuan Bab 2	
3	Jumat	10 Juni 2022	Diskusi & Pengajuan Bab 3	
4	Senin	13 Juni 2022	Diskusi & Pengajuan Revisi Bab 3	
5	Selasa	21 Juni 2022	finalisasi Proposal	
6	Senin	12 Juni 2023	Diskusi bab 4	
7	Kamis	15 Juni 2023	Revisi bab 4 & Diskusi bab 5	
8	Jumat	23 Juni 2023	Revisi bab 4 & 5	
9	Senin	26 Juni 2023	Finalisasi Draft keseluruhan	
10	Selasa	27 Juni 2023	Finalisasi skripsi	

Keterangan :
1. Setiap konsultasi harus diparaf Dosen Pembimbing
2. Minimal jumlah konsultasi sebanyak 8 kali pertemuan
3. Penyelesaian TA/Skripsi paling cepat 3 (tiga) bulan dan paling lambat 6 (enam) bulan sejak surat pengantar dikeluarkan oleh sekretariat Fakultas

Jakarta, 3 Juli 2023
Kepala Program Studi

(Achmad Z., S.T., M.T.)