TUGAS UJIAN AKHIR SEMESTER

MATA KULIAH PEMBELAJARAN SECARA STASTIK DAN OPTIMASI

"DIMENSIONALITY REDUCTION"

Oleh :

Aditya Rifky Ramadhan – 2101201036

Marita Fauziah – 2101201046

Queen Hesti Ramadhamy – 2101201037

PROGRAM STUDI MAGISTER TEKNIK ELEKTRO

UNIVERSITAS TELKOM

BANDUNG

2020

Daftar Isi

BAB 1

PENDAHULUAN

1.1. Latar Belakang

Seiring perkemabangan teknologi ketersedian data melimpah dan sangat disayangkan banyak sekali data yang berukuran besar namun memiliki sifat kurang akurat, tidak konsisten bahkan tidak lengkap. Hal tersebut disebabkan oleh kesalahan yang dilakukan oleh manusia *(human error)* maupun perangkat (*computer error*) pada proses penginputan data. Kesalahan lain dapat disebabkan ketidak konsitenan dalam format ketika proses peingisian data. Data yang tidak sesuai berpengaruh kepada data yang sudah relevan sehingga memungkiknan data menjadi sulit untuk dimengerti. Oleh karena itu dibutuhkan teknik *preprocessing* pada data yang digunakan. *Preprocessing* merupakan suatu teknik dari data mining untuk mengolah suatu data mentah sehingga menjadi data yang berkualitas dan relavan. Salah satu proses dari teknik *preprocessing* adalah *Data Reduction* atau *Dimensionality Reduction*. Dengan adanya proses tersebut kesalahan dalam data akan berhasil dikurangin , mengungi komplesitas watu serta ruang dan dapat membuat model yang lebih sederhana sehingga menghasilkan data yang relevan dan mudah dimengerti.

Dalam tugas ini akan dilakukan proses *Dimensionality Reduction* yang terdiri menjadi bagian yaitu *Principal Component Analysis*, *Linear Discriminant* dan *Factor Analysis*. Data yang digunakan adalah data kanker dan metode *Neural Network* digunakan untuk memproses dan mengklasifikasi data yang sudah dilakukan *Dimenstionality Reduction.* Hasil akurasi yang didapatkan dari 3 model tersebut akan dibandingkan untuk mengetahui model *dimensionality reduction* yang menghasilkan akurasi yang optimal. Tugas ini dilakukan menggunakan *google collab and phyton.*

BAB 2

DASAR TEORI

2.1 Preprocessing

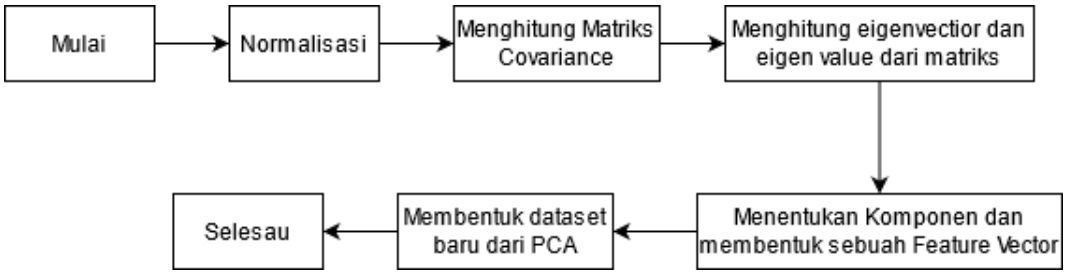
Preprocesing merupakan tahap yang dilakukan untuk menyiapkan data sebelum dilakukan tahapan *processing.* Pada tahap ini, data yang diolah merupakan data mentah yang berukuran besar yang mempunyai beberapa kesalahan seperti data kurang akurat, data kurang konsisten dan data sulit untuk dimengerti. Dari tahapan *preprocessing* menghasilkan data yang akurat sehingga data tersebut dapat menghasilkan nilai akurasi yang tinggi. Pada tahapan *preprocessing* terdiri beberapa proses *Data Cleaning, Data Integration, Data Reduction* dan *Data Transformation Discrerization.* Tujuan dari *preprocessing* adalah menghasilkan dataset baru yang akan menjadi inputan data pada tahapan *processing.*

2.2 Dimentionality Reduction

Dimentionality Reduction adalah transformasi data dari ruang berdimensi tinggi menjadi ruang berdimensi rendah sehingga representasi berdimensi rendah mempertahankan beberapa komponen yang berguna dari dari dataset asli. Kegunaan dari *Dimentionality Reduction* adalah untuk mengurangi data yang tidak sesuai sehingga menghasilkan dataset yang sudah sesuai. Ada beberapa jenis dari *Dimentionality Reduction* seperti *Principal Component Analysis, Liniear Discriminant Analysis* dan *Factor Analysis*.

2.3 Principal Component Analysis

Principal Component Analysis (PCA) merupakan suatu metode bagian dari *Dimensionality Reduction* yang berguna untuk mereduksi dimensi variable pada banyak komponen, kompresi data dan Analisa static. *Principal Analysis* memiliki fungsi mereduksi dimensi dari data set input menjadi komponen utama yang mempunyai dimensi lebih kecil dengan mempertahankan *variability* dalam data dimana komponen utama yang terbentuk tidak mempunyai korelasi dengan komponen lainnya serta meminimalisir kehilangan informasi yang relevan.



Skema diatas adalah proses bagaimana PCA dapat menghasilkan dataset baru. Dalam PCA, komponen pertama adalah kombinasi linear yang terbobot dari variabel asal yang dapat menerangkan keragaman terbesar, komponen utama yang kedua adalah kombinasi liniear terboboti dari variabel asal yang tidak berkorelasi dengan komponen utama pertama dan seterusnya. Penentuan banyak komponen utama yang diekstrak ditentukan dengan beberapa kriteria, sebagai berikut :

- Kriteria *Eigen Value,* dimana komponen utama ditentukan dengan melihat komponen yang memiliki nilai eigen lebih besar atau sama dengan satu. Nilai eigen dari komponen utama yang kurang dari satu dikeluarkan dari analisis
- Kriteria *Apriori,* dimana peneliti sudang menentukan jumlah komponen utama yang akan diekstrak
- Kriteria *Presentase varians,* dimana jumlah komponen utama yang diekstrak ditentuka oleh presentasi kumulatof varians
- Scree test dimana dengan membuat plot *eigen value* terhadap komponen utama berdasarkan urutan yang diperoleh.

Proses yang dilakukan oleh PCA untuk medapatkan dataset baru dijelaskan dalam langkah berikut :

1. Menyiapkan data set yang ingin diolah.
2. Merepresentasikan setiap variable menjadi variabel vector
3. Mencari nilai rata rata dari variable vector
4. Mengurangi nilai rata rata dari masing masing vector untuk mendapatkan nilai *Zero mean* (Yi) dengan tujuan untuk mengambil hasil yang dibedakan dari masing masing variabel dan membuang variabel yang sama.
5. Mencari nilai matriks kovarian
6. Mencari nilai eigen vector dan eigen value dari matriks kovarian
7. Nilai eigen vektror dengan nilai eigen tertinggi ke terendah
8. Mencari nilai PCA dengan menggunakan persamaan Yi.v*

2.4 Linear Discriminant Analysis

Liniear Discriminant Analysis (LDA) adalah metode ekstasi fitur yang memiliki perpaduan dari perhitungan statistika dan matematika yang memberlakukan property statistic terpisah untuk tiap objek. LDA juga dapat diartikan model *Dimentionality Reduction* yang mencari proyeksi terbaik yang dapat memisahkan data dengan pendekatan *least-squares.* LD memberikan perlakukan statistic yang terpisah untuk tiap komponen dengan menemukan kombinasi linier dari fitur yang menjadi ciri khas objek setiap komponen. Tujuan

metode LDA untuk mencari proyeksi linear untuk memaksimumkan pemisahan antar kelas dan juga meminimumkan jarak didalam komponen yang sama.

2.5 Factor Analysis

Factor Analysis (FA) adalah salah satu teknik statistika yang digunakna untuk memberikan deskripsi relatif sederhana melalui reduksi jumlah pengubah yang disebut faktor. Analisis faktor adalah prosedur mengidentifikasi komponen berdasarkan kesamaan data. Kesamaan data dianalisa ditunjukan dengan kolerasi data yang tinggi. Sederhananya, FA digunakan untuk mengurangi dimensi dari komponen. Ada beberapa jenis FA seperti *Exploratory Factor Analysis* dan *Confirmatory Factor Analysis*. Proses yang dilakukan FA dalam melakukan pengurangan dimensi, sebagai berikut :

1. Melakukan uji korelasi antar komponen asal dengan tujuan untuk pengurangan komponen FA menjadi lebih sederhana
2. Uji kelayakan data menggunakna basis faktor
3. Mencari nilai akar ciri dari matriks R
4. Mengurutkan akar ciri yang terbentuk dari terbesar sampai terkecil
5. Mencari proporsi keragaman untuk mengetahui beberapa faktor yang terbentuk
6. Mengalokasikan setiap komponen kedalam faktor sesuai dengan nilai *loading* dimana jika nilai *loading* identic maka dilakukan rotasi secara *orthogonal* maupun *non orthogonal*.
7. Dataset baru sudah terbentuk.

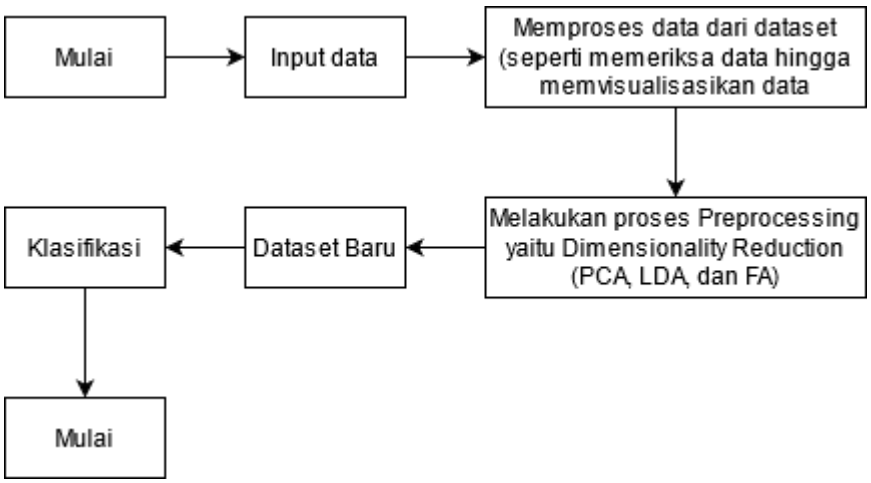
2.5 Neural Network

Neural Network adalah metode yang juga dikenal sebagai jaringan saraf tiruan yang berupaya untuk mensimulasikan struktur biologis otak manusia serta sistem saraf. Jaringan saraf tiruan ini memiliki konsep membuat kecerdasan buatan yang dapat mengklasifikan sebuah data berupa penilaian dari setiap parameter parameternya. Cara kerja dari ANN dengan memodelkan hubungan antara data input dan output untuk mengenali atau menemukan pola pola data. Dalam ANNs memiliki pemodelan jaringan dalam pemodelan jaringan tersebut terdapat *input layer, hidden layer* dan *output layer. Input layer* bertindak seperti distributor dan *input* ke lapisan ini langsug ditransmisikan ke *Hidden Layer*. *Hidden Layer* bertindak sebagai pendeteksi fitur secara teori dan terdiri lebih dari satu *hidden layer*.

BAB 3

PERANCANGAN SISTEM

Bab ini menjelaskan tentang alur kerja dari Tugas UAS ini, penjelasan tentang skema jaringan yang disimulasilkan *preprocessing, new data set* yang dijadikan sebagai input bagian *classification.*



Dari skema diatas dijelaskan alur perancangan yang dilakukan dengan menggunakan *google collab* sebagai simulator dengan dataset yang digunakan mengenai kanker yang didownload melalui github. Proses yang dilakukan dapat dijelaskan detail sebagai berikut :

1. Melakukan *import* dataset dari github
2. Meimport library yang digunakan

```
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
import keras #memasukan library keras
from keras.models import Sequential
from keras.layers import Activation, Dropout, Flatten, Dense
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
import sklearn.metrics as metrics
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from time import time
from sklearn.decomposition import PCA
```

3. Membaca dataset yang d*import* kemudian dilanjutkan dengan menproses data tersebut dimulai dari pengecekan data hingga visualisasi data yang telah di*import.* Dengan menggunakan *code* dibawah ini. Detail penjelasan terlampir pada *google collab.*

```
data = pd.read_csv('https://raw.githubusercontent.com/syamsulrizal123/SLO/main/data_cancer.c
sv', sep = ',')
data.head()
data.shape
data.info()
sns.set()
data.hist(figsize=(10,10), color='Blue')
plt.show()
data.drop(["Unnamed: 32","id"],axis=1,inplace=True)
data.diagnosis.replace(('B', 'M'), (1, 0), inplace=True)
data.isnull().sum()
fig=plt.subplots(figsize=(20,20))
plt.title('Correlation of Features', y= 1.05, size =15)
corr=data.corr()
sns.heatmap(corr, annot=True)
plt.show()
data_m = data[data.diagnosis == 0]
data_b = data[data.diagnosis == 1]
labels = ["M","B"]
values = [len(data_m),len(data_b)]
trace = [go.Pie(labels=labels, values=values,
            marker=dict(colors=["blue","brown"]))]
layout = go.Layout(title="Percentage of M = malignant, B = benign ")
fig = go.Figure(data=trace,layout=layout)
pyo.iplot(fig)
x=data.iloc[:, 1:32].values
y=data.iloc[:, 0].values
x.shape
```

4. *Preprocessing* dimana melakukan proses *dimensionality reduction.*

PCA	scaler = StandardScaler() x_scaled = scaler.fit_transform(x) pca = PCA(n_components=2) pca.fit(x_scaled) X_reduced_pca = pca.transform(x_scaled) pca_data = pd.DataFrame(X_reduced_pca,columns=["principalcomponent1","principalcomponent2"]) pca_data["diagnosis"] = y pca_data.head() gambar = pca_data["diagnosis"] data = [go.Scatter(x = pca_data.principalcomponent1 y = pca_data.principalcomponent2,

	mode = 'markers', marker=dict(size=12, color=gambar, symbol="pentagon", line=dict(width=2)))] layout = go.Layout(title="PCA", xaxis=dict(title="Principal Component 1"), yaxis=dict(title="Principal Component 2"), hovermode="closest") fig = go.Figure(data=data,layout=layout) pyo.iplot(fig)
LDA	scaler = StandardScaler() x_scaled = scaler.fit_transform(x) lda = LinearDiscriminantAnalysis(n_components= 1) lda.fit(x_scaled,y) X_reduced_lda = lda.transform(x_scaled) lda_data = pd.DataFrame(X_reduced_lda,columns=["principalcomponent1"]) lda_data["diagnosis"] = y lda_data.head() gambar = lda_data["diagnosis"] data = [go.Scatter(x = lda_data.principalcomponent1, mode = 'markers', marker=dict(size=12, color=gambar, symbol="pentagon", line=dict(width=2)))] layout = go.Layout(title="LDA", xaxis=dict(title="Principal Component 1"), yaxis=dict(title="Principal Component 2"), hovermode="closest") fig = go.Figure(data=data,layout=layout) pyo.iplot(fig)

FA	scaler = StandardScaler() #digunakan untuk melakukan normalisasi data
	x_scaled = scaler.fit_transform(x)
	fa = FactorAnalysis(n_components= 2)
	fa.fit(x_scaled,y)
	X_reduced_fa = fa.transform(x_scaled)

```
x_reduced= fa.fit_transform(x_scaled,y)
fa_data = pd.DataFrame(X_reduced_fa,columns=["principalcomponent1","principalcomponent2"])
fa_data["diagnosis"] = y
fa_data.head()
gambar = fa_data["diagnosis"]
data = [go.Scatter(x = fa_data.principalcomponent1,
            y = fa_data.principalcomponent2,
            mode = 'markers',
            marker=dict(
                size=12,
                color=gambar,
                symbol="pentagon",
                line=dict(width=2) #çevre çizgileri
                ))]

layout = go.Layout(title="Factor Analysis",
            xaxis=dict(title="Principal Component 1"),
            yaxis=dict(title="Principal Component 2"),
            hovermode="closest")
fig = go.Figure(data=data,layout=layout)
pyo.iplot(fig)
```

5. Membagi data menjadi dua data yaitu data train dan data test

```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state =0)
```

6. Membuat dan mengelola data train dan data test

PCA	X_train = pca.fit_transform(X_train) X_test = pca.transform(X_test)
LDA	X_train = lda.fit_transform(X_train,y_train) X_test = lda.transform(X_test)
FA	X_train = fa.fit_transform(X_train,y_train) X_test = fa.transform(X_test)

7. Melakukan tahapan ANN

```
model = Sequential()
model.add(Dense(10, input_dim=2,activation='softmax'))
model.add(Dense(20, activation='softmax'))
model.compile(loss='sparse_categorical_crossentropy', optimizer='sgd', metrics=['accuracy'])
history=model.fit(X_train, y_train,epochs=100, validation_data=(X_test,y_test))
```

8. Menghitung Akurasi dan memvisualisasikan dalam bentuk grafik model loss dan model akurasi

```
y_pred_ann = model.predict(X_test)
ac=model.evaluate(X_test,y_test)
print("PCA - Neural Network Accuracy = {0:.2f}%".format(ac*100))
plt.plot(history.history['loss'], label='train loss')
plt.plot(history.history['val_loss'], label='test loss')
plt.title('Model Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'])
plt.show()
plt.savefig('Model_Loss.png')
plt.title('Model Accuracy')
plt.plot(history.history['val_accuracy'], label='test accuracy')
plt.plot(history.history['accuracy'], label='train accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epochs')
plt.legend()
plt.show()
```

BAB 4

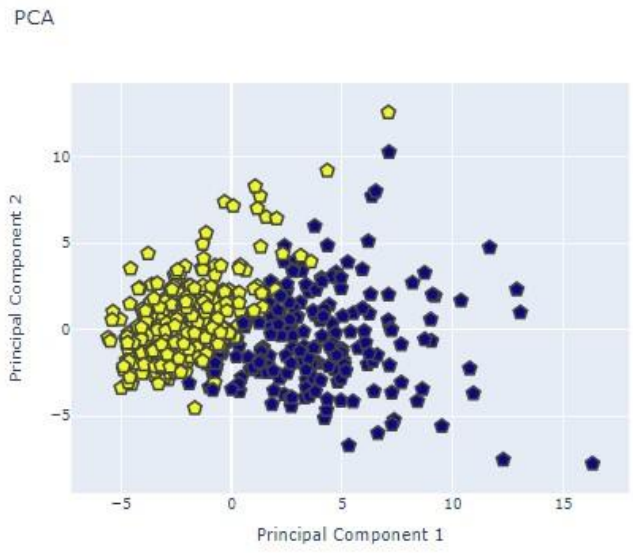
HASIL SIMULASI

4.1 Hasil Proses *Preprocessing*

Dalam skema sebelumnya dilakukan 3 tahapan *dimensionality reduction* yang merupakan bagian dari tahapan *preprocessing*. Setelah melakukan proses didapatkan hasil sebagai berikut :

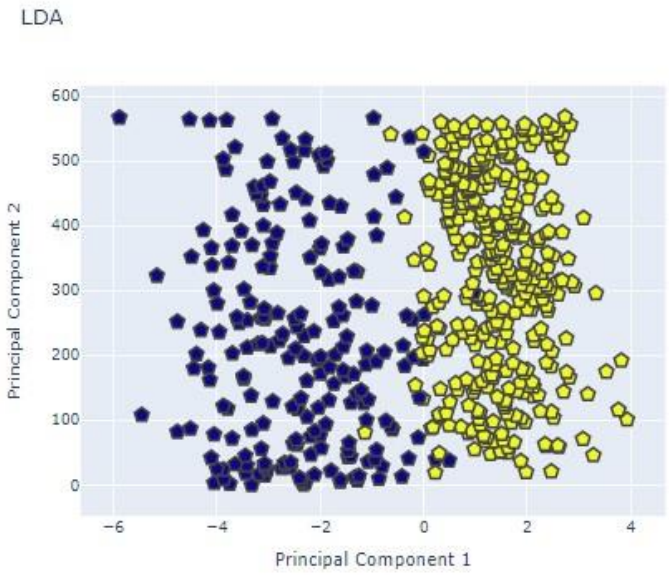
- *Principal Component Analysis*
  Dataset baru yang dihasilkan

	principalcomponent1	principalcomponent2	diagnosis
0	9.192837	1.948583	0
1	2.387802	-3.768172	0
2	5.733896	-1.075174	0
3	7.122953	10.275589	0
4	3.935302	-1.948072	0



- *Liniear Discriminant Analysis*
  Dataset baru yang dihasilkan

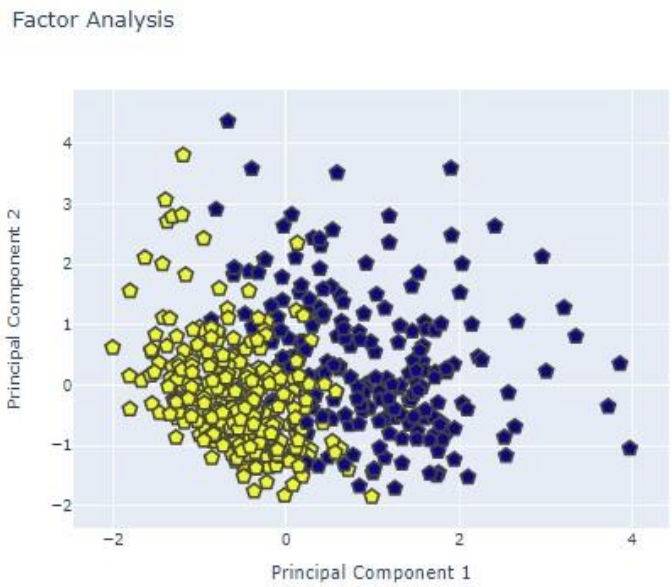
	principalcomponent1	diagnosis
0	-3.323927	0
1	-2.319108	0
2	-3.747425	0
3	-4.048549	0
4	-2.281158	0



- *Factor Analysis*

  Dataset baru yang dihasilkan

	principalcomponent1	principalcomponent2	diagnosis
0	1.193106	2.804874	0
1	1.762309	-1.451941	0
2	1.577860	0.390700	0
3	-0.670756	4.372901	0
4	1.756956	-0.291862	0



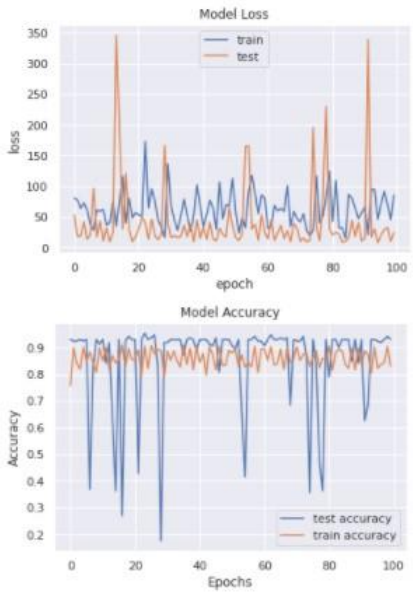
4.2 Hasil Akurasi

Setelah melakukan proses *preprocessing* dilakukan proses klasifikasi menggunakan metode *Neural Network* . Dari proses tersebut didapatkan hasil akurasi yang berbeda dari setiap model yang digunakan. Hasil Akurasi yang didapatkan ditunjukan pada tabel dibawah :

Model	Hasil Akurasi
PCA	94.15 %
LDA	93.57%
FA	63.16%
Tanpa *Dimensionality Reduction*	71.35%

- PCA

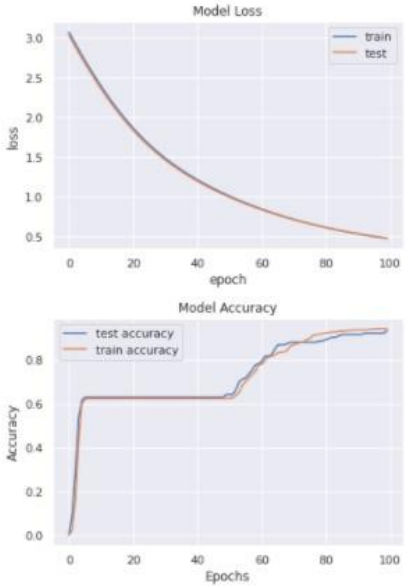
Dari hasil grafik yang didapatkan dari model loss dan model akurasi dapat dilihat bahwa terdapat perbedaan yang cukup signifikan dari data train dan data test. Dari model loss dapat dilihat bahwa data test mengalami kenaikan loss sedangkan data train mendapatkan nilai data loss yang tidak telalu tinggi dan dapat dilihat dari model akurasi nilai akurasi yang didapatkan dari data test cukup stabil dibandingkan dengan data train yang mengalami penaikan dan penurunan nilai akurasi. Namun, hasil yang didapatkan masih sesuai dengan yang diharapkan.



- LDA

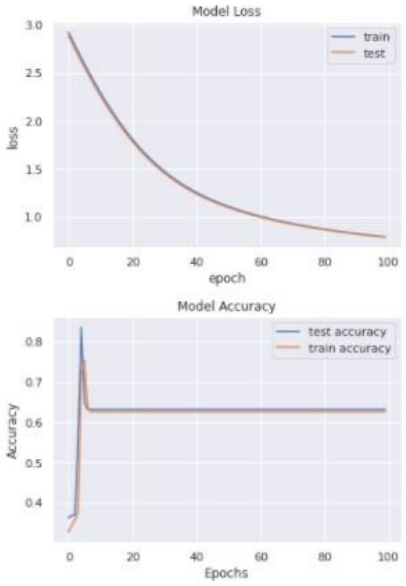
Dari hasil grafik yang didapatkan dari model loss dan model akurasi dapat dilihat bahwa tidak terdapat perbedaan yang cukup signifikan atau dapat dikatan memiliki hasil yang sama dari data train dan data test. Dari grafik tersebut dapat dilihat lebih *smooth.* Dari model loss dapat dilihat bahwa data test dan data train mengalami penuruan loss disesuaikan

dengan banyak epoch dan jika dilihat dari model akurasi, nilai akurasi yang didapatkan dari data test dan data train naik secara konstan.



- FA

Dari hasil grafik yang didapatkan dari model loss dan model akurasi dapat dilihat bahwa tidak terdapat perbedaan yang cukup signifikan atau dapat dikatan memiliki hasil yang sama dari data train dan data test. Dari kedua grafik tersebut dapat dilihat bahwa nilai loss semakin berkurang seiring dengan banyaknya epoch dan nilai akurasi naik seiring dengan banyaknya epoch.



BAB 5

KESIMPULAN

Sekarang ketersediaan data sangat melimpah dan data tersebut umumnya berukuran besar dengan banyak komponen didalamnya. Dengan ukuran data yang besar belum tentu data tersebut menghasilkan data yang relevan , dimana data tersebut bisa saja kurang konsisten, kurang akurat dan kurang lengkap. Tahapan *preprocessing* perlu dilakukan untuk mendapatkan data yang sudah dioptimalkan sehingga data yang didapatkan lebih relevan dan dapat diproses untuk menghasilkan nilai akurasi yang tinggi. Metode *dimentionally reduction* yang merupakan bagian tahapan *preprocessing* untuk mengurangi data data yang kurang relevan. Metode *dimentionally reduction* terdiri dari 3 jenis yaitu *Principal Component Analysis* (PCA), *Liniear Discriminant Analysis* (LDA) dan *Factor Analysis* (FA). Pada tugas ini disimulasikan ketiga model tersebut didapatkan nilai akurasi yang berbeda-beda. Model PCA mendapatkan nilai akurasi 94.15 % sedangkan Model LDA dan FA mendapatkan nilai akurasi masing masing 93.57% dan 63.16 %. Dari hasil tersebut dapat disimpulkan bahwa model PCA memiliki nilai akurasi yang paling tinggi. Dapat dilihat juga bahwa penggunaan *Dimensionality Reduction* meningkatkan akurasi yang didapatkan dimana menggunakan salah satu model *Dimensionality Reduction* akurasi yang didapatkan 93.57 % sedangkan tanpa menggunakan *Dimensionality Reduction* akurasi yang dapatkan 71.35%.