

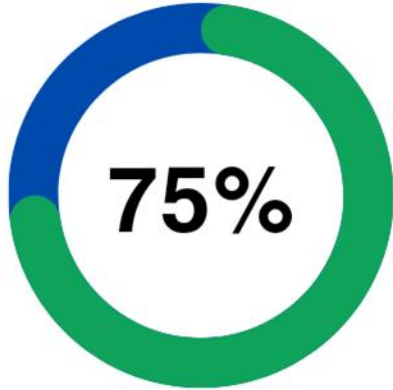
# Project CV 1:

# Face Recognition

## Kelompok CV A - Michio Kaku

- Amar Ma'ruf
- Bayuzen Ahmad
- Benedictus Dikha Arianda
- Dika Mahendra
- Elsa Nurul Hidayah
- Fariz Rachman Hadi
- Fatah Abdul Jalil
- Haris Raharjo Putro

# Background: Apakah wanita lebih banyak belanja daripada pria?

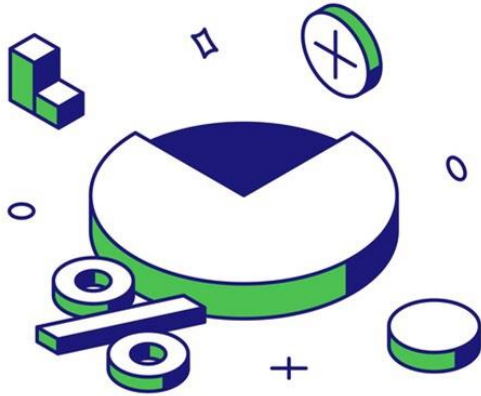


**Men spend 75% more on goods bought in-store than women.**

Meskipun stigma sosial cenderung menyimpulkan bahwa wanita lebih sering berbelanja, data statistik online mengungkapkan sebaliknya, dengan pria berbelanja 47% lebih banyak, dengan 75% nya di toko fisik.

Meskipun angka ini mengejutkan, data demografis seperti ini dapat menjadi alat yang sangat efektif dalam mengembangkan bisnis, terutama di sektor retail.

# Background: Apakah wanita lebih banyak belanja daripada pria?



Dengan memahami data demografis, bisnis dapat mengadaptasi strategi pemasaran yang lebih terfokus pada pria, meningkatkan pengalaman belanja pelanggan, dan secara keseluruhan meningkatkan efisiensi operasional, dampaknya dapat dirasakan pada pengeluaran keseluruhan bisnis.

Oleh karena itu, teknologi *computer vision* (CV) dan pendeteksian *gender* pengunjung menjadi kunci untuk mengoptimalkan strategi bisnis berdasarkan data demografis.

# Objective



Mengembangkan model klasifikasi gambar menggunakan arsitektur ResNet, VGGNet, dan GoogleNet untuk memahami pola pembelian berdasarkan jenis kelamin pengunjung. Penelitian ini bertujuan untuk mengembangkan strategy marketing dengan mendapatkan data gender pengunjung melalui classification image.

Goals : mendapatkan arsitektur dengan dengan score F1 yang terbaik.



# TIMELINE PROJECT









TASK TITLE	Complete	WEEK 1					WEEK 2				
		M	T	W	TH	F	M	T	W	TH	F
Project Planing											
Business Understanding 1	100%										
Create Model	100%										

## KETERANGAN

	Create VGGNET
	Create ResNet
	Create GoogleNet
	presentasi

## A decorative graphic on the right side of the page. It features a series of black lines of varying thicknesses that resemble circuit traces or a stylized 'E' shape. There are several small circles at the ends of these lines, some solid black and some white with black outlines. In the lower-left portion of this graphic, there are two small, square, black-and-white portrait photographs. The top portrait is of a man wearing a baseball cap with the letters 'MLB' on it. The bottom portrait is of an older man with glasses. To the left of the top portrait is the number '00166'.



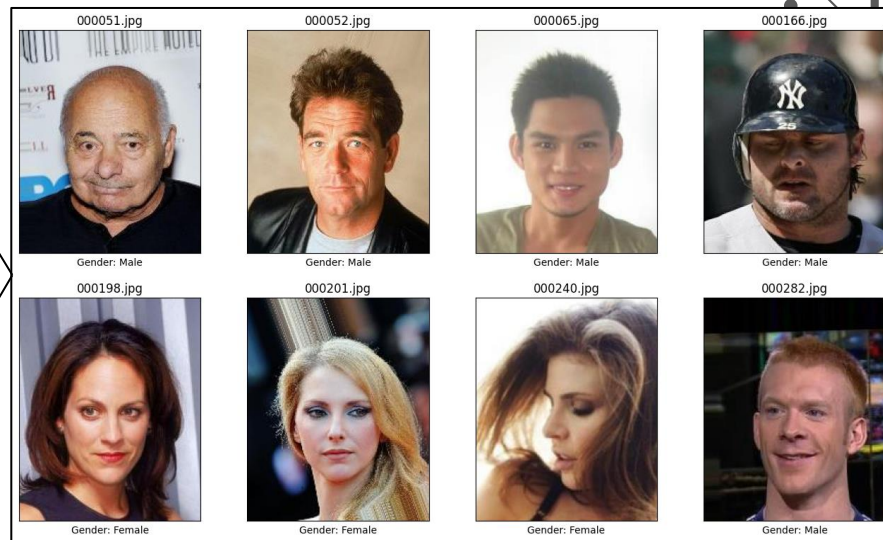
			
000051	000052	000065	000166
			
000545	000559	000572	000608

	Image_Id	5_o_Clock_Shadow	Arched_Eyebrows	Attractive	Bags_Under_Eyes	Bald	Bangs	Big_Lips	Big_Nose	Black_Hair	...
0	000001.jpg	0	1	1	0	0	0	0	0	0	...
1	000002.jpg	0	0	0	1	0	0	0	1	0	...
2	000003.jpg	0	0	0	0	0	0	1	0	0	...
3	000004.jpg	0	0	1	0	0	0	0	0	0	...
4	000005.jpg	0	1	1	0	0	0	1	0	0	...
...	...	...	...	...	...	...	...	...	...	...	...
202594	202595.jpg	0	0	1	0	0	0	1	0	0	...
202595	202596.jpg	0	0	0	0	0	1	1	0	0	...
202596	202597.jpg	0	0	0	0	0	0	0	0	1	...
202597	202598.jpg	0	1	1	0	0	0	1	0	1	...
202598	202599.jpg	0	1	1	0	0	0	0	0	0	...
202599 rows x 41 columns											

# Datasets



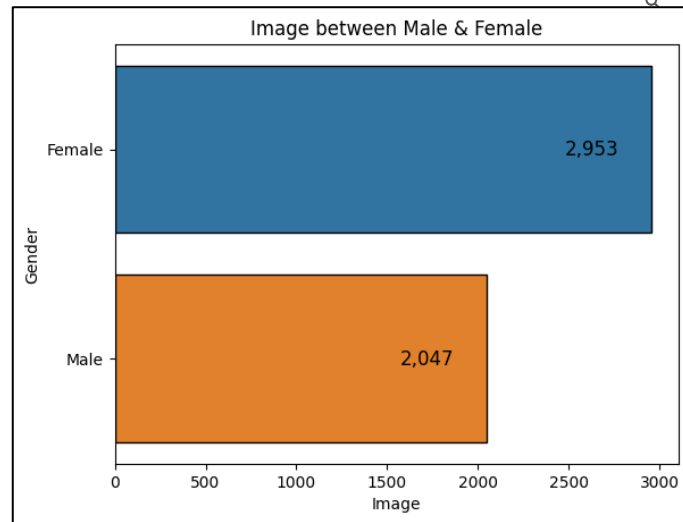
	img_number	5_o_Clock_Shadow	Arched_Eyebrows	Attractive
0	000001.jpg	0	1	1
1	000002.jpg	0	0	0
2	000003.jpg	0	0	0
3	000004.jpg	0	0	1
4	000005.jpg	0	1	1
5 rows × 41 columns				
(202599, 41)				



# Datasets

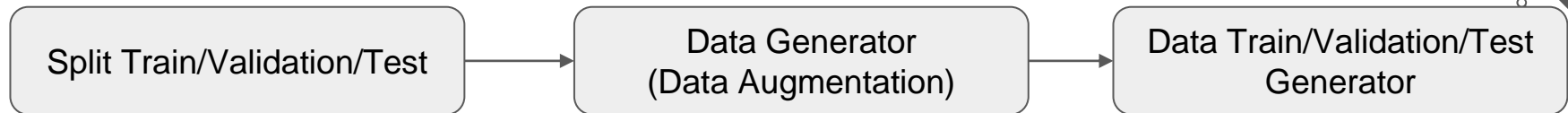
```
1 final_data = attr_list[attr_list['img_number'].isin(os.listdir(data_path +  
2                                                    '/Images'))]  
3 print(f'Full Data: {attr_list.shape[0]} rows')  
4 print(f'Filtered Data: {final_data.shape[0]} rows')
```

Full Data: 202599 rows  
Filtered Data: 5000 rows





# Preprocessing Images Steps



```
1 print('Data Training: ', len(train_data))
2 print('Data Validation: ', len(val_data))
3 print('Data Testing: ', len(testing_data))
```

```
Data Training: 4000
Data Validation: 500
Data Testing: 500
```

```
datagen = ImageDataGenerator(
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

```
1 train_gen = datagen_train.flow_from_dataframe(
2     dataframe=train_data,
3     directory='/content/drive/MyDrive/Dataset/project_cv_1_FaceRecognition/Images/',
4     x_col='Image_Id',
5     y_col='Male',
6     batch_size=32,
7     class_mode='raw',
8     target_size=(178, 218),
9     validate_filenames=False
10 )
11
12 # prepare data validation using data generator which is taken from image directory
13 val_gen = datagen_val.flow_from_dataframe(
14     dataframe=val_data,
15     directory='/content/drive/MyDrive/Dataset/project_cv_1_FaceRecognition/Images/',
16     x_col='Image_Id',
17     y_col='Male',
18     batch_size=32,
19     class_mode='raw',
20     target_size=(178, 218),
21     validate_filenames=False
22 )
```

**Note:**  
Validation and Test set only Rescaling

# Arsitektur Model

## VGG - VGG16

1. Framework: PyTorch
2. Kustomisasi Top Layer:
  - a. Input: 4096 node
  - b. Output: 2 node
3. Optimizer:
  - a. SGD
  - b. Learning Rate: 0.003
  - c. Momentum: 0.9
4. Loss Func.: Cross Entropy Loss

```
# Architecture
model = models.vgg16(pretrained=True)
model.classifier[6] = nn.Linear(4096, 2)

optimizer = optim.SGD(model.parameters(),
                        lr=0.003, momentum=0.9)
criterion = nn.CrossEntropyLoss()
```

## GoogLeNet - InceptionV3

1. Framework: TensorFlow
2. Weights: Imagenet
3. Layer Trainable: False
4. Kustomisasi Top Layer: Yes
5. Optimizer:
  - a. Adam
  - b. Learning Rate: 0.001
6. Loss Func.: Binary Cross Entropy

```
# adding custom layer, global average pooling layer,
# and prediction layer
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation="relu")(x)
x = Dense(512, activation="relu")(x)
x = Dense(256, activation="relu")(x)
x = Dense(128, activation="relu")(x)
x = Dense(64, activation="relu")(x)
x = Dense(32, activation="relu")(x)
x = Dense(16, activation="relu")(x)
x = Dense(8, activation="relu")(x)
x = Flatten()(x)
predictions = Dense(1, activation="sigmoid")(x)

# merge model
model = Model(inputs=base_model.input,
              outputs=predictions)
```

## ResNet - ResNet50

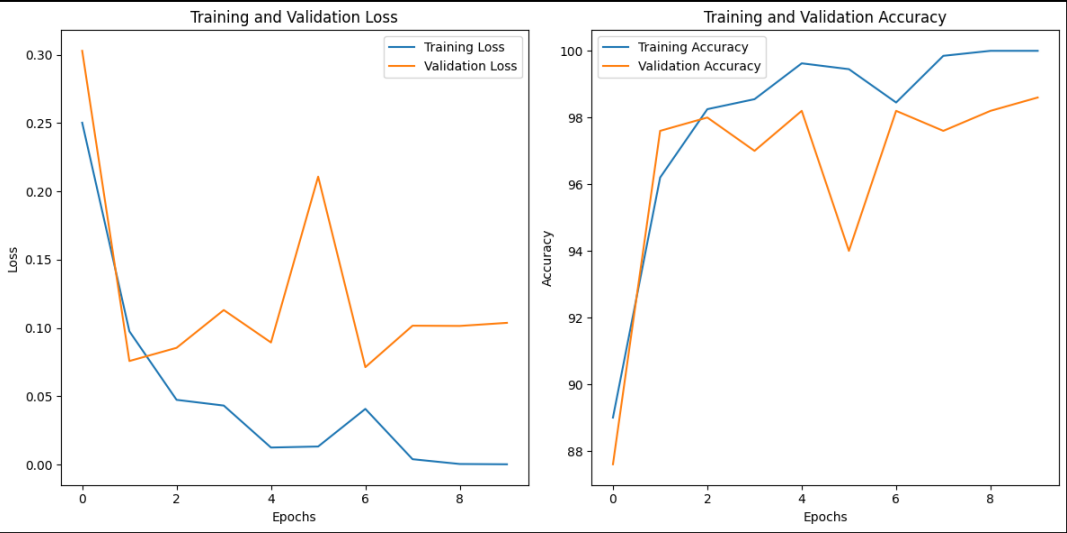
1. Framework: PyTorch
2. Kustomisasi Top Layer:
  - a. Input: sejumlah unit pada model tersebut.
  - b. Output: 2 node
3. Optimizer:
  - a. Adam
  - b. Learning Rate: 0.001
4. Loss Func.: BCEWithLogitsLoss

```
# Load the saved model
model = models.resnet50(pretrained=True)
num_features = model.fc.in_features
model.fc = nn.Linear(num_features, 2)
```

# Training Performance

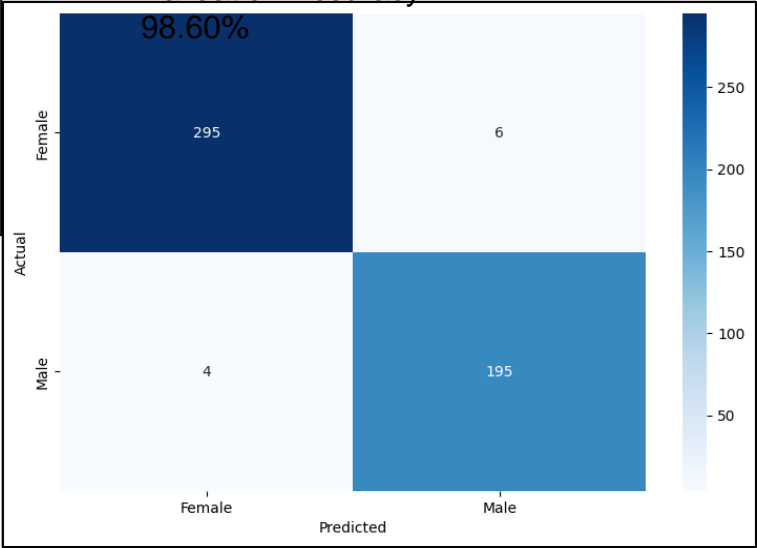
Model: VGG16

Epoch: 10 | Batch Size: 32  
Total Runtime: 20 menit



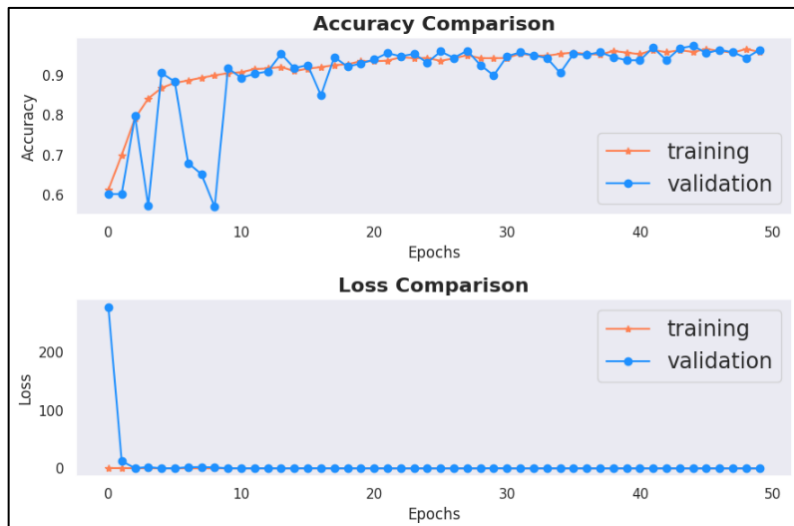
Training Loss : 0.0002  
Validation Loss : 0.1037  
Training Accuracy : 100%  
Validation Accuracy :

Classification Report:				
	precision	recall	f1-score	support
0	0.99	0.98	0.98	301
1	0.97	0.98	0.97	199
accuracy			0.98	500
macro avg	0.98	0.98	0.98	500
weighted avg	0.98	0.98	0.98	500



# Training Performance

Model: Googlenet - inceptionv3



```
Classification Reports:
precision    recall  f1-score   support

   Female    0.92    0.99    0.96     295
    Male    0.99    0.88    0.93     205

 accuracy          0.95     500
  macro avg       0.96    0.94    0.95     500
 weighted avg     0.95    0.95    0.95     500
```

Epoch: 50 | Batch Size: 32

Total Runtime: 1 Jam 30 menit

Training Loss :

0.1101

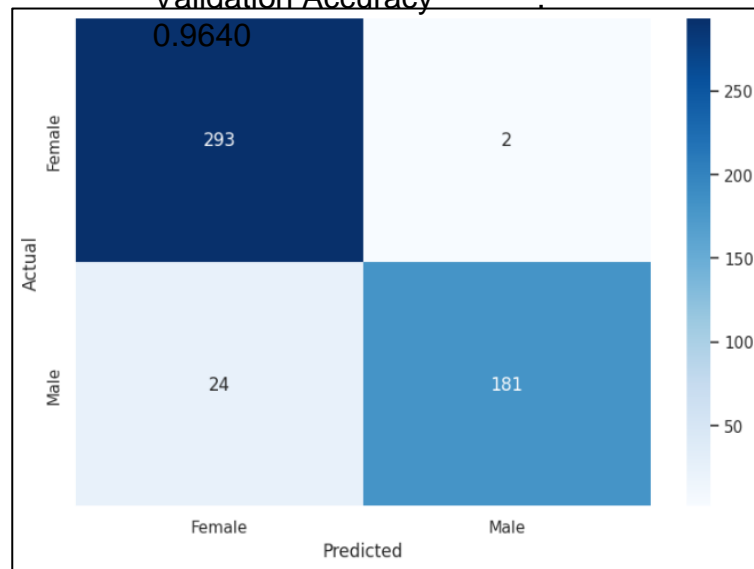
Validation Loss :

0.1183

Training Accuracy :

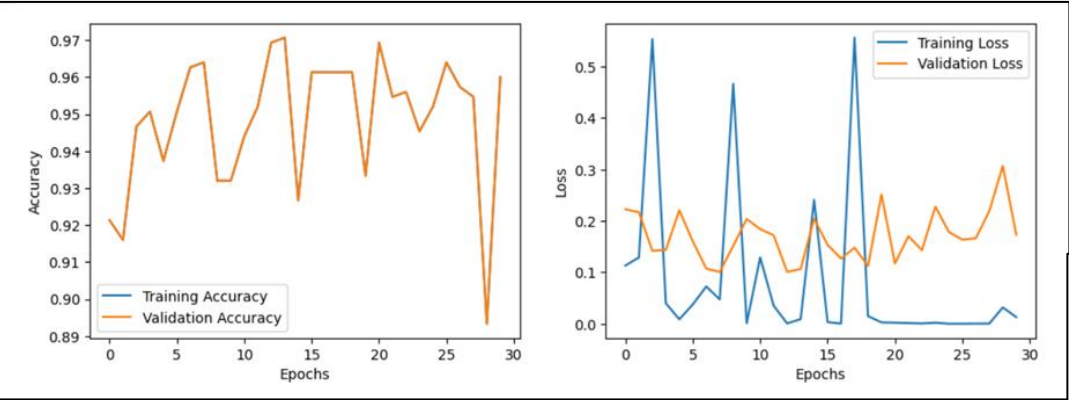
0.9588

Validation Accuracy :



# Training Performance

Model: ResNet50



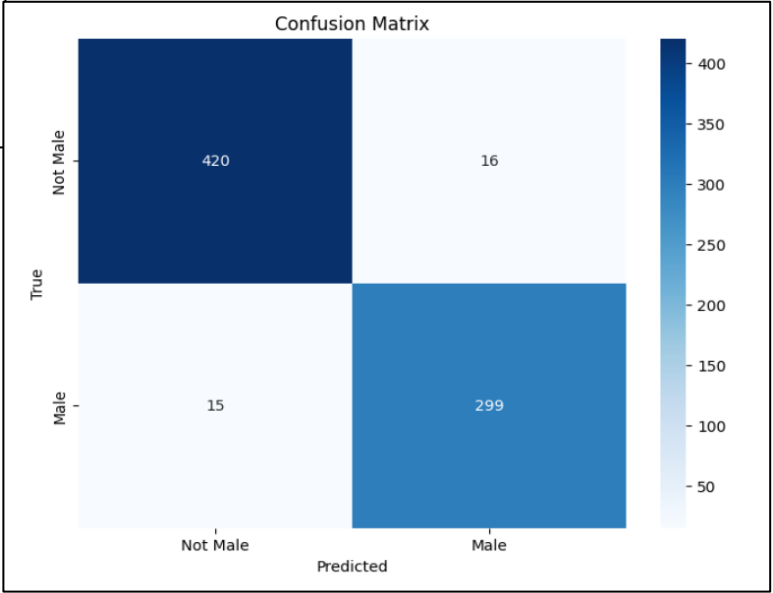
Epoch: 30 | Batch Size: 32  
Total Runtime: 3 Jam

Training Loss : 0.0129  
Validation Accuracy : 96%

Test Accuracy: 95.87%

Classification Report:

	precision	recall	f1-score	support
0	0.97	0.96	0.96	436
1	0.95	0.95	0.95	314
accuracy			0.96	750
macro avg	0.96	0.96	0.96	750
weighted avg	0.96	0.96	0.96	750



# Kesimpulan

Dari ke-tiga pretrained model, yaitu VGG, ResNet, dan GoogleNet, hasil menunjukkan bahwa model VGG mencapai kinerja terbaik. Dengan jumlah epochs **sebanyak 10**, model VGG mampu mencapai **akurasi sebesar 98.6%** dengan runtime < 30 menit.

Penerapan model pengenalan gender pada pelanggan supermarket menggunakan pretrained model VGG dengan kinerja unggul memiliki beberapa dampak positif pada nilai bisnis:

- Efisiensi Cost Campaign
- Pengalaman Pelanggan yang Ditingkatkan
- Analisis Data Lebih Mendalam
- Peningkatan Targeting Promosi

Secara keseluruhan, penerapan model VGG untuk pengenalan gender dapat memberikan nilai tambah yang signifikan dalam konteks supermarket, meningkatkan efisiensi operasional dan meningkatkan pengalaman pelanggan dengan pendekatan yang lebih terarah.