

# Artificial Intelligence

Samkeun Kim <skim@hknu.ac.kr>

<http://cyber.hankyong.ac.kr>

# What Is AI?

“인공지능”, “머신러닝” 및 “딥러닝”이라는 용어를 상호 교환적으로 사용

⇒ 그러나 엄격하게 따진다면 이들은 서로 다른 의미를 갖는다:

## **Artificial Intelligence (AI)**

기계에게 인간의 행동을 모방할 수 있는 능력 부여

IBM의 Deep Blue(초창기 체스 프로그램)가 AI의 한 예 => 알파고 제로, ...

## **Machine Learning (ML)**

기계가 통계 기법을 사용하여 이전 정보와 경험을 통해 학습하는 AI의 한 분야

목표는 기계가 과거 데이터 학습에 기초하여 미래 예측을 수행하는 것: IBM Watson, 스팸 메일 분류기 등

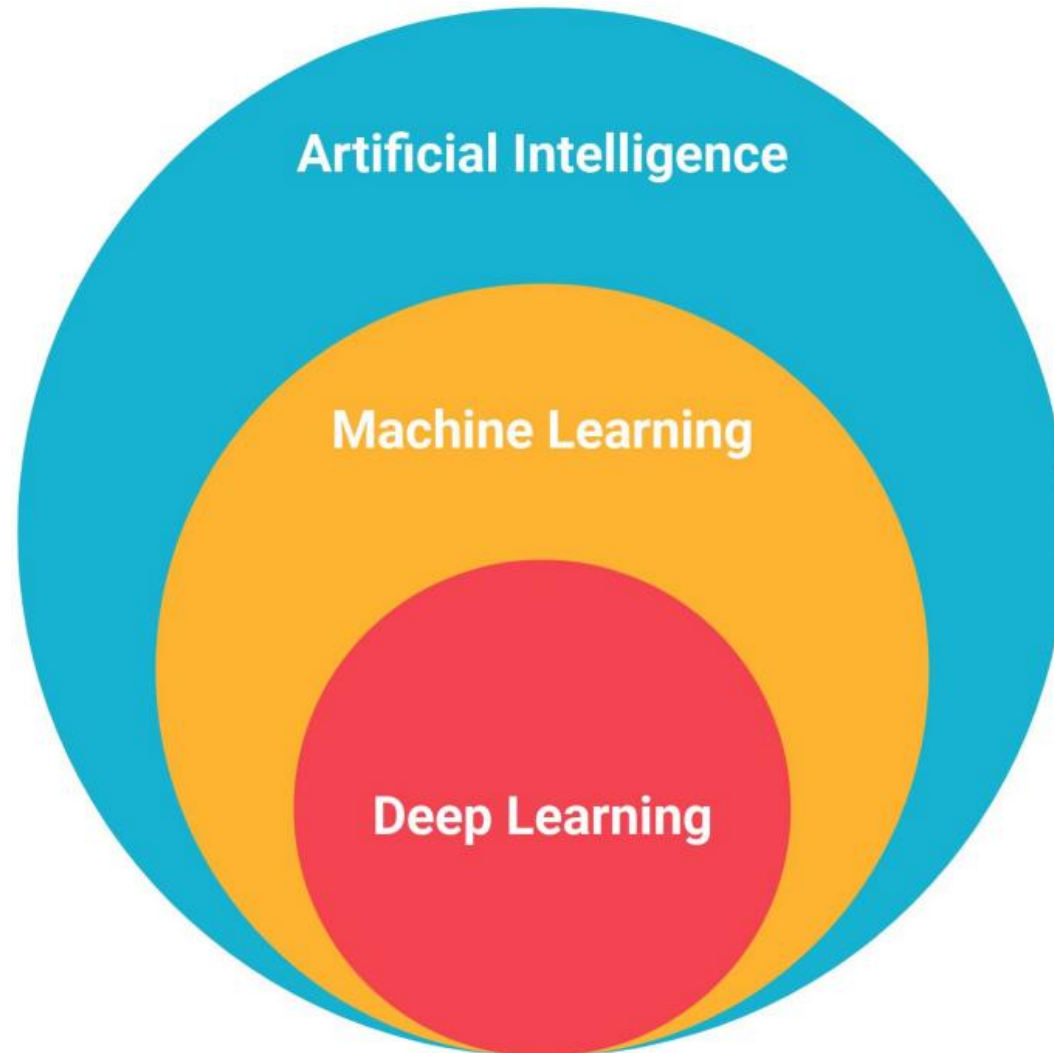
## **Deep Learning (DL)**

예측을 수행하는데 심층 신경망(Deep NN)을 이용하는 머신 러닝의 한 분야:

특히 컴퓨터 비전, 음성 인식, 자연어 이해 등에서 탁월한 예측을 수행

## *What Is AI?*

The relationship between AI, machine learning, and deep learning



# Machine Learning Tsunami

1958년, Rosenblatt's Perceptron

1969년, Minsky's Perceptrons

1970년대, 인공신경망 암흑기 (1차)

...

1986년, MLP

1990년대, DNN 학습 불가능! => 인공신경망 암흑기 (2차)

...

2006년, Geoffrey Hinton's Deep Learning (>98% accuracy)

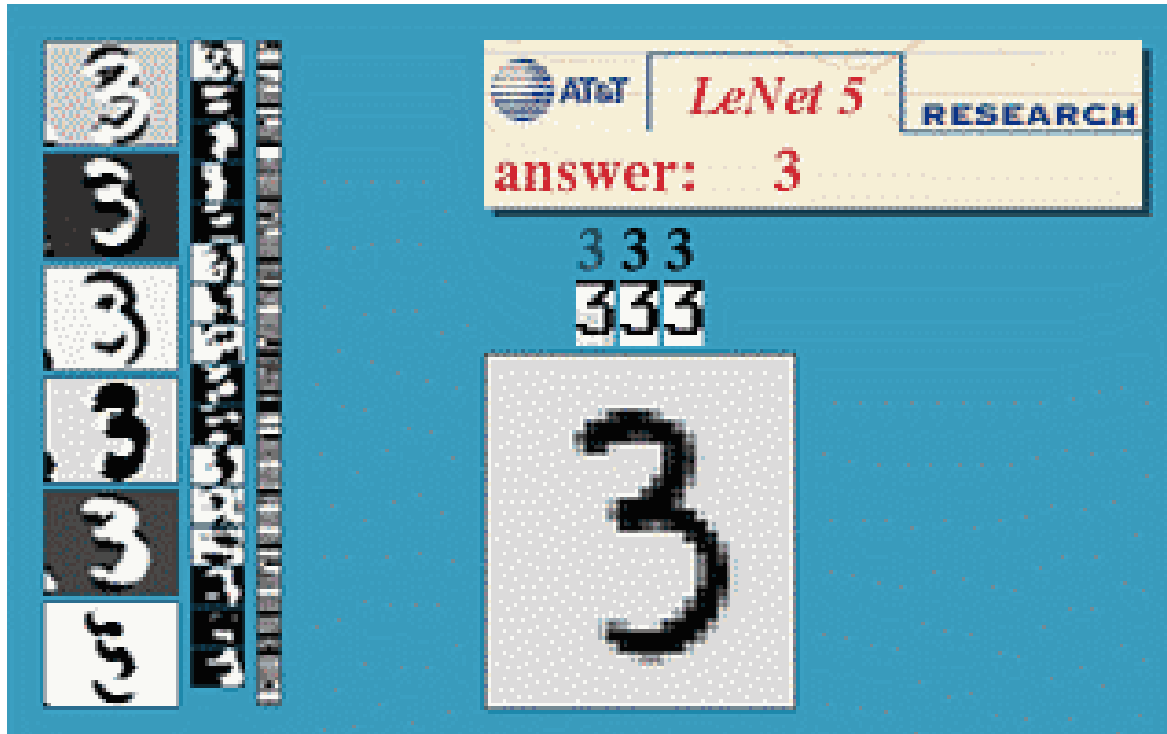
이후 10여년, 오늘날 하이테크 제품의 심장부에 놓여 있다...

웹 검색 결과에 순위를 매기고, 스마트폰 음성 인식을 개선시켜 주고, 비디오를 추천해 주고, Go 게임에서 세계 챔피언을 이기고, 자동차를 자율 주행하고 ...

# 성공 사례 1

## [Yann LeCun's LeNet-5 Demo](#)

- The most widely known CNN architecture - Created by Yann LeCun in 1998 and has been widely used for handwritten digit recognition (MNIST).



## 성공 사례 2

AlphaGo vs. Lee Sedol (March 2016) – Reinforcement Learning



# Machine Learning in Your Projects

집에서 만든 로봇에게 스스로 생각할 수 있게 하는 두뇌를 심어줄 수 있을까?

얼굴을 인식하게 할 수 있을까?

주변을 걸어 다니게 할 수 있을까?

기업마다 수천 톤의 데이터가 있다: 로그 데이터, 재정 데이터, 제품 데이터, 센서 데이터 등

⇒ 머신 러닝 기법을 이용하여 데이터에 숨겨져 있는 보석을 발굴해 낼 수 있다.

예를 들어,

- 고객을 그룹으로 나누어 각 그룹에 대한 최상의 전략 탐색
- 유사한 고객들이 구매한 내용을 기반으로 각 고객에게 제품 추천
- 사기 거래 탐지
- 내년도 수익 예측
- And more <https://www.kaggle.com/competitions>

# Lecture Objective and Approach

강의 목표:

AI 개념, 통찰력, 'Learning from Data' 할 수 있는 프로그램 구현

알고리즘의 토이 버전을 구현하는 것이 아니라, 제품 수준의 Python 프레임워크 사용:

- Scikit-Learn
- TensorFlow

Hands-on Approach



# 강의 일정

## Part I, The Fundamentals of Machine Learning

제 00강 강의 개요

제 01강 Machine Learning의 용어와 개념

제 02강 End-to-End Machine Learning Project [1]

제 03강 End-to-End Machine Learning Project [2]

제 04강 Classification - MNIST

제 05강 Training Models [1] - Linear Regression

제 06강 Training Models [2] - Polynomial Regression & Learning Curves

제 07강 Training Models [3] - Logistic Regression & Softmax Regression

제 08강 Support Vector Machines - Linear & Nonlinear SVM Classification, SVM Regression

제 09강 Support Vector Machines - Linear & Nonlinear SVM Classification, SVM Regression [2]

제 10강 Decision Trees - CART Training Algorithm, Regression

제 11강 Ensemble Learning and Random Forests

제 12강 Term Project 공지 - 공공데이터를 활용한 AI 기반의 공공서비스 개발

중간고사

## Part II, Neural Networks and Deep Learning

제 13강 Introduction to Artificial Neural Networks [1] - Perceptrons & MLP

제 14강 Introduction to Artificial Neural Networks [2] - MLP with Keras

제 15강 Introduction to Artificial Neural Networks [3] - Functional API, Callbacks, TensorBoard

제 16강 Training and Deploying TensorFlow Models at Scale [1] - Docker

제 17강 Training and Deploying TensorFlow Models at Scale [2] - Google Cloud AI Platform

제 18강 Dynamic Model을 위한 Flask Web App 개발

제 19강 Cloud Deployment - The Heroku Platform

제 20강 Training Deep Neural Nets [1] - Vanishing & Exploding Gradients Problems

제 21강 Training Deep Neural Nets [2] - Transfer Learning, Faster Optimizers, LR Scheduling

제 22강 Deep Computer Vision Using Convolutional Neural Networks

제 23강 Processing Sequences Using RNNs and CNNs

(제 24주 Term Project 결과발표)

기말고사

# Using Code Examples

Download:

<https://github.com/ageron/handson-ml2>

# Tales of a machine learning startup

1. How to build and deploy a diabetes diagnostic app using Tensorflow, Google Cloud AI Platform and Flask," <https://medium.com/@vinoroy70/tales-of-a-machine-learning-startup-50b290e1c9cd>
2. Flask Web Development, Miguel Grinberg, O'reily.
3. (주교재) Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, Chapter 19.

## 개발 순서:

- TensorFlow 모델 구축
- Google Cloud AI Platform(cloud service)에 Deploy
- Flask web app으로 만들기
- Cloud Deployment on the Heroku Platform

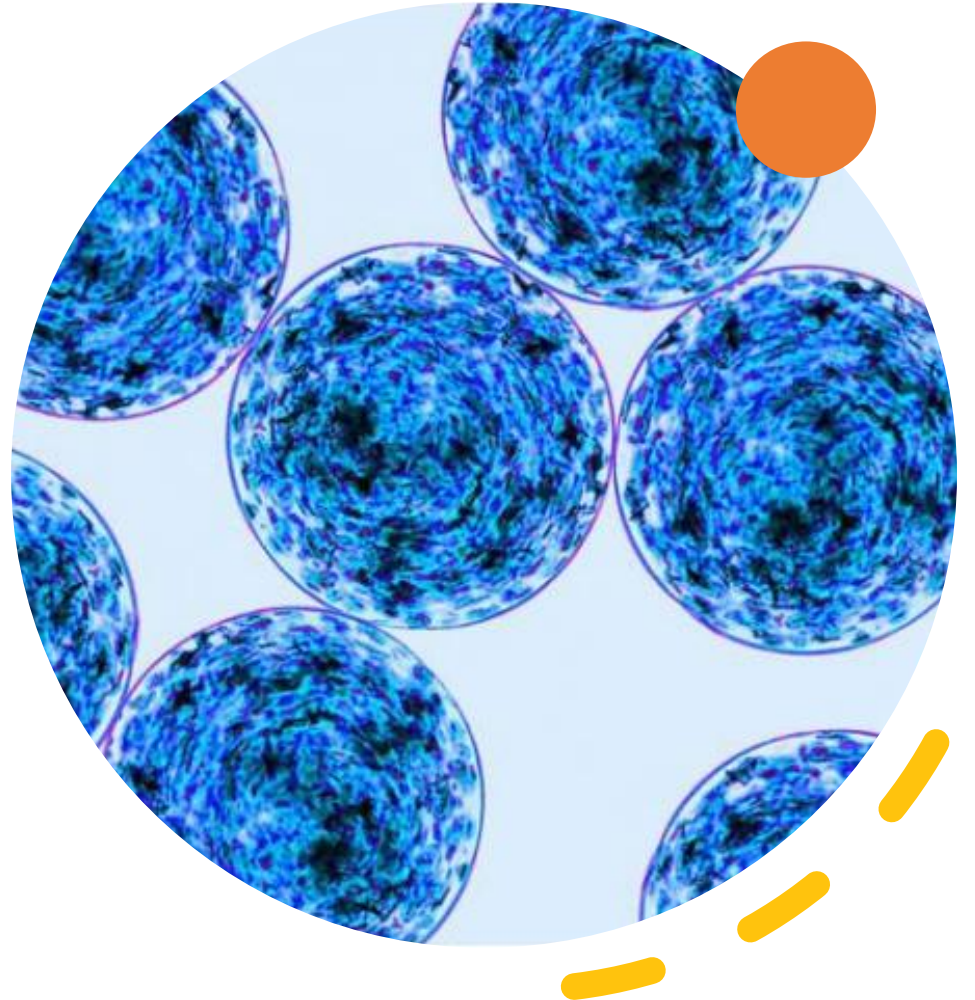
## *Tales of a machine learning startup*

ACME Learning Co.라는 새로운 기계 학습 스타트업 회사의 멤버라고 해보자.

이제 자신의 팀이 캐나다 당뇨병 협회를 위한 기계 학습 앱을 구축하기 위해 작업을 시작한다고 해보자.

팀원들은 '이번 크리스마스에는 뭔가 제대로 된 선물을 줄 수 있겠구나!'하는 기대감으로 마음이 벅차 오른다.

자, 이제 기대감을 현실로 만들어 보자!!



## *Tales of a machine learning startup*

자신이 스타트업의 CTO로서 긴급 회의를 소집하여 프로젝트 목표를 설명한다.

먼저 우리 팀은 의사가 실험실에서의 의료 데이터에 기반하여 여성 환자에 대한 당뇨병 진단을 예측하는 데 도움이 되는 웹 앱을 개발해야 한다고 취지를 설명한다.

데이터 과학 전문가 Mike가 '데이터가 있습니까?'라고 묻는다.

Mike에게 '우리는 운이 좋다! 계약 업체가 전체 연구 데이터 베이스에 대한 액세스를 제공해준다고 한다.'라고 말해준다.

Mike의 얼굴이 안심하는 듯 보인다: '품질 좋은 데이터를 통해 Tensorflow를 사용하여 Dense 신경망을 쉽게 구축할 수 있겠군!'



애플리케이션 엔지니어인 Kim이 '의사가 환자의 실험 결과 데이터를 입력하여 환자가 당뇨병인지 여부를 웹앱을 통해 즉시 진단 예측할 수 있다면 정말 멋진 일이 될 것입니다.'라고 제안한다.

아울러, Kim은 **Flask**를 사용하여 웹 앱을 쉽게 구축할 수 있는 방법을 자신 있게 설명한다.

이제 팀에서는 계획을 가진 듯하다: Dense 신경망 모델을 구축한 다음 Flask 웹앱을 통해 예측을 표시한다.

회의를 마치고 팀이 프로젝트 작업을 시작하려고 회의실을 빠져나가려고 하자 팀의 아키텍트인 Jhon이 '여러분, 모델을 어떻게 배포(deployment)할 것인가를 생각해 봤나요?'라고 말한다.

모델 결과 값을 배포하고 제공하는 것이 매우 중요하지만 재미없고 골치 아프다는 것을 모두가 알고 있기 때문에 팀원들이 여기저기서 한숨을 내쉬는 소리가 들린다.

## *Tales of a machine learning startup*

잠시 후 회의실 뒤쪽에서 인턴 Keisha가 'GCP를 사용하는 것이 어떤가요?'라고 더듬거리는 목소리로 말한다

그녀는 동료들이 모델을 배포하는데 사용하고 있는 **Google Cloud AI Platform** 배포 메커니즘 및 API에 대해 설명을 한다.

이 시점에서 팀원들은 '앱을 만들기 위한 퍼즐 조각들이 모두 갖춰졌구나'라고 깨닫는다.

그런데 한 가지가 더 남아있다.





애플리케이션을 개발할 때 가장 공통적인 두통거리는 애플리케이션을 **live production server**에 배포하는 것이다. 우리는 일찌감치 그 두통거리를 제거하고 진단 예측 애플리케이션을 라이브 URL로 푸시하기로 한다. 이후부터는 반복적으로 계속 빌드하면서 업데이트된 내용을 계속해서 배포할 수 있다.

이런 방법은 팀의 작업 진행 상황을 다른 사람에게 보여주기가 아주 쉽다.

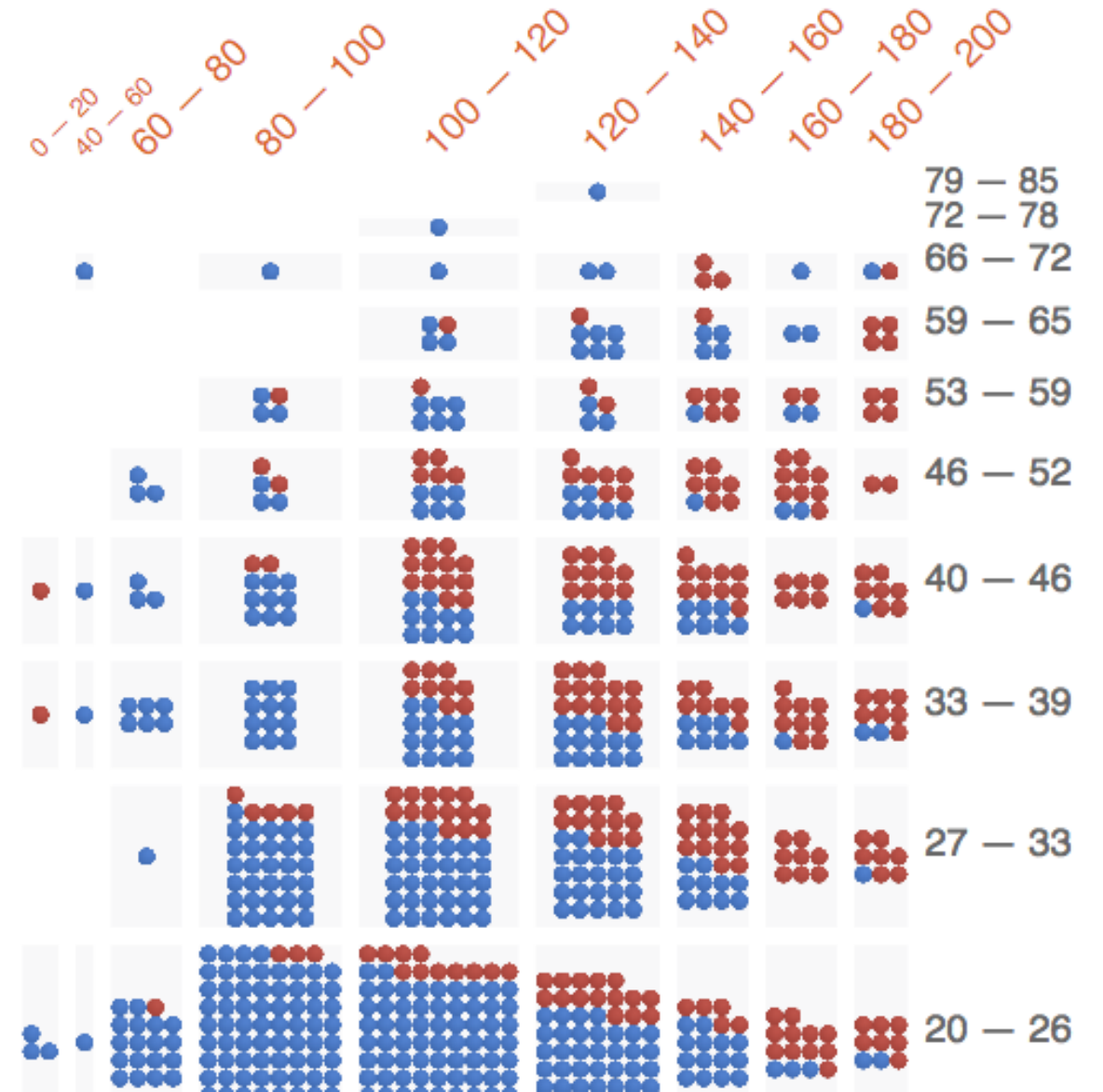
Google Cloud Platform, Nodejitsu, OpenShift 및 **Heroku**와 같은 서비스 제공 업체들이 제공하는 몇 가지 플랫폼이 있다.

여기서는 Heroku를 사용하지만 다른 옵션을 선택해도 아무런 문제가 되지 않는다.

## Step 1 : Building the Keras Tensorflow model

데이터 과학 전문가인 Mike는 현재 주어진 문제가 이진 분류 문제라고 설명한다.

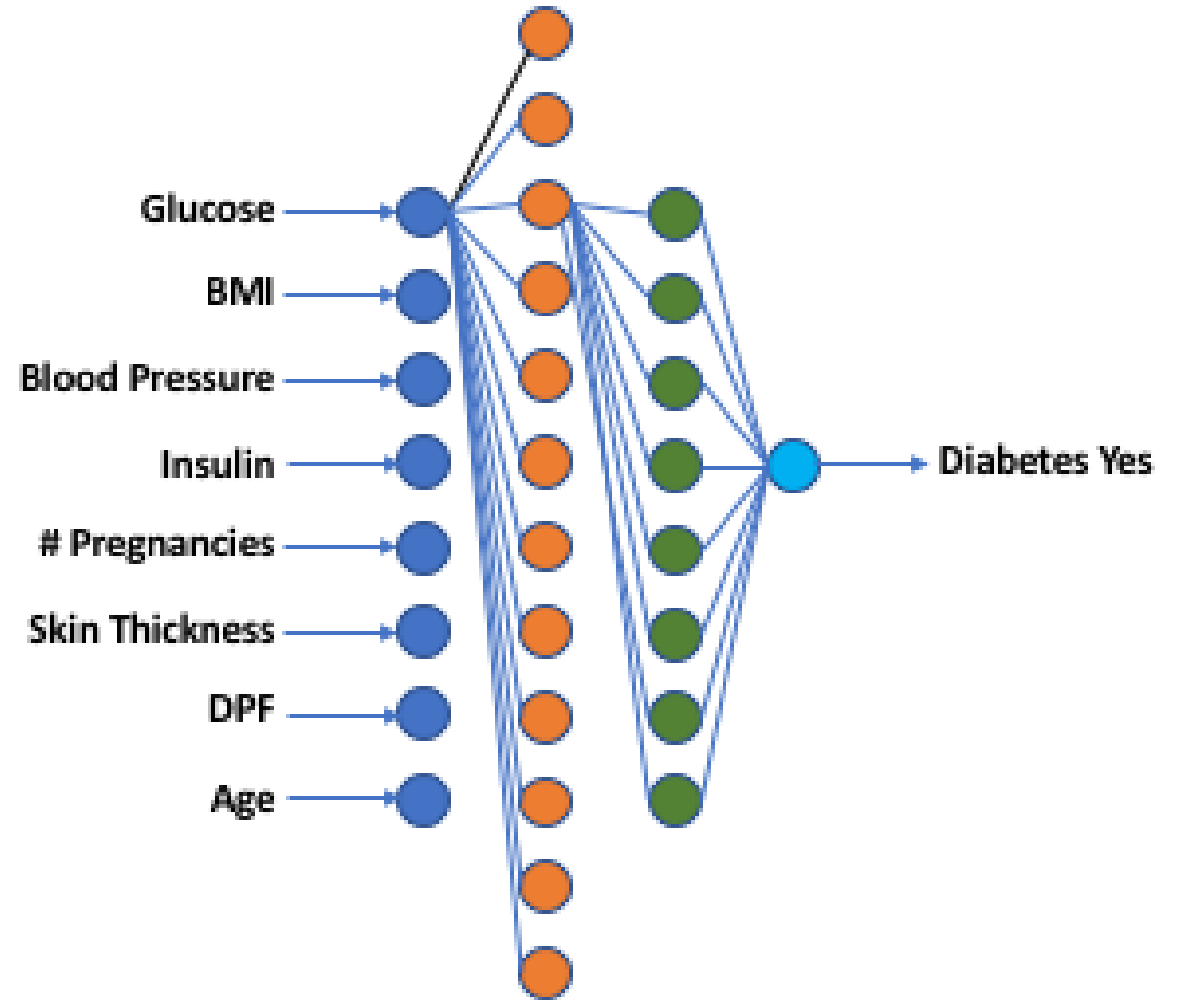
768명의 환자 데이터(포도당, BMI, 인슐린 등)와 환자가 당뇨병에 걸렸는지 여부에 대한 의료 전문가들의 평가가 주어진다.



Horizontal axis blood Glucose level, vertical axis age segments, and red points indicated diabetes result

예측 모델을 구축하기 위해 8개의 입력 노드, 12개의 첫 번째 히든 레이어 노드, 8개의 두 번째 히든 레이어 노드, 그리고 Sigmoid 활성화 함수를 갖는 1개의 출력 노드로 구성된 Dense 신경망을 사용한다.

보다 정교한 아키텍처로 구축하면 더 나은 성능의 모델을 얻을 수 있지만 여기서의 목적이 아니므로 그냥 넘어간다.



Architecture of the dense ANN

The following code highlights the model development using **Keras** and **Tensorflow**.

The complete code can be obtained on the project [github](#)

```
# create the model
inputs = keras.Input(shape=(8,))
hidden1 = Dense(12,activation='relu')(inputs)
hidden2 = Dense(8,activation='relu')(hidden1)
output = Dense(1,activation='sigmoid')(hidden2)
model = keras.Model(inputs,output)

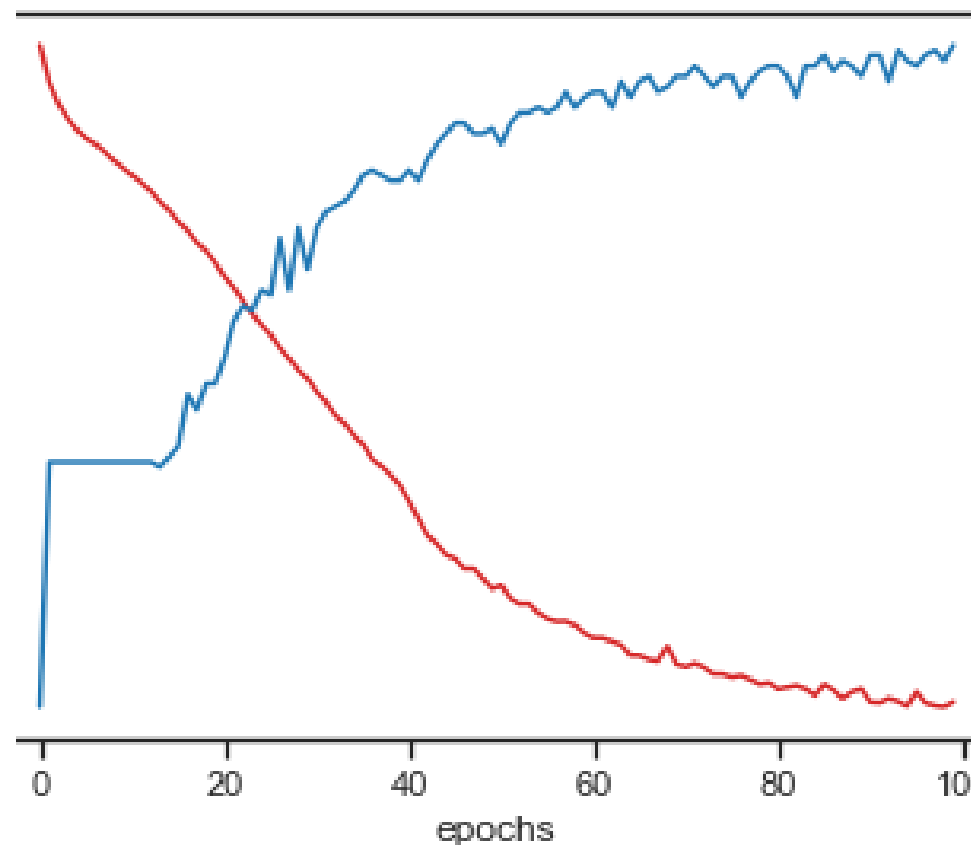
# summary to verify the structure
model.summary()

# compile the keras model
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=
['accuracy'])

# train the model on the training data
history =
model.fit(X_train,y_train,epochs=100,batch_size=16,verbose=0)
```

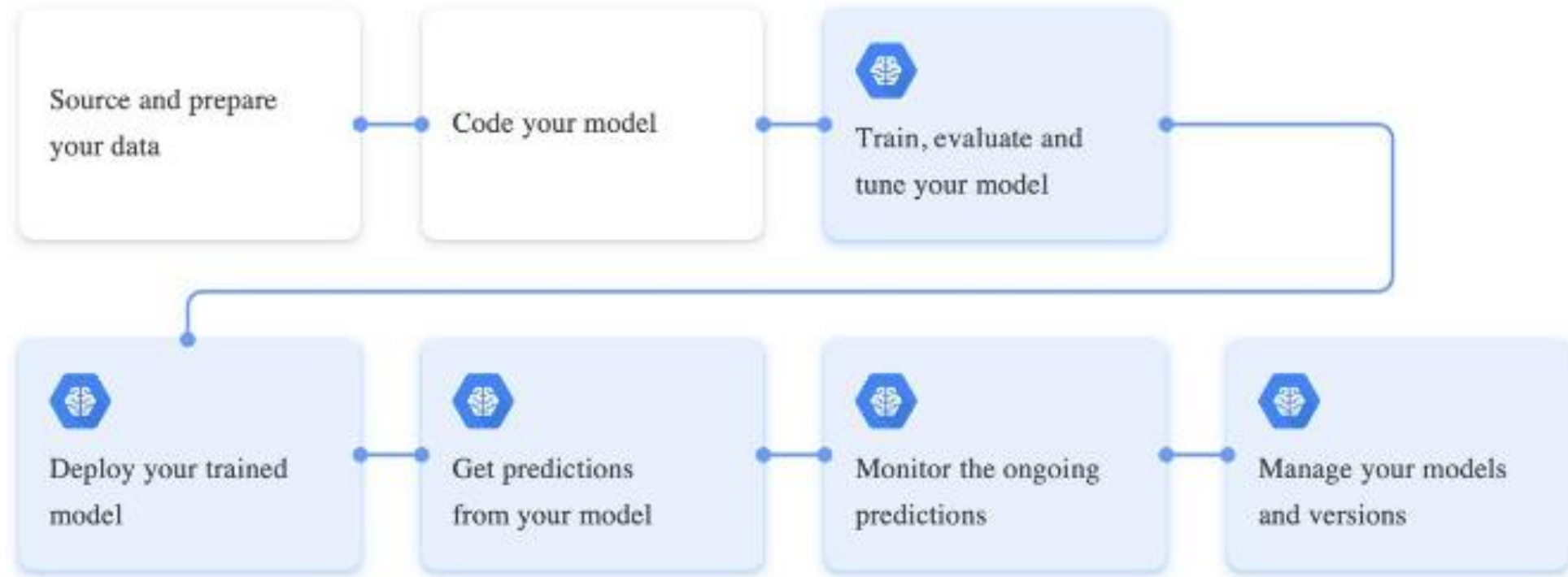
앞의 기본 ANN 아키텍처의 경우에도 결과를 받아 들일만 하다.

여기서의 목적은 모델을 세부적으로 조정하고 성능을 평가하기 위한 것이 아니므로 그대로 진행한다.



Loss and accuracy for 100 epochs of training

## Step 2 : Deploying the model on Google Cloud AI Platform



인턴 Keisha가 제안했듯이 **Google Cloud AI Platform**에 모델을 배포하면 보통 이슈가 되는 규모 확장, 모델 버전 관리 등에 대한 걱정없이 모델을 쉽게 예측 모델로 전환할 수 있다.

머신 러닝 모델 개발자는 Google Cloud AI Platform을 통해 모델 버전을 관리하는 일 뿐만 아니라 예측 모델을 쉽게 학습시키고 배포할 수 있다.

**REST API**를 통해 모델에 액세스 할 수 있으므로 예측 계산을 웹 애플리케이션에 쉽게 통합시킬 수 있다.

또한 Google Cloud AI Platform 인프라가 성능 문제도 관리해 주므로 성능에 관해서도 걱정할 필요가 없다.

Google Cloud AI Platform에 프로그래밍 방식으로 모델을 배포하려면 다음을 수행해야 한다:

- Google Cloud Platform (GCP)에서 계정 만들기
- GCP 소프트웨어 개발 키트 (SDK) 다운로드

다음 코드는 단계별로 python을 사용하여 프로그래밍 방식으로 모델을 배포하는 방법을 보여준다.



A — Create a **bucket** on Google Cloud AI Platform and transfer the model

```
# set the variables for the gcp ai platform
PROJECT_ID = 'your_project_name'

# name of the bucket the will contain the model as well as the
# region. This must absolutely be a GLOBALLY UNIQUE name
BUCKET_NAME = 'pima_bucket'
REGION = "us-central1"

# set the project
! gcloud config set project $PROJECT_ID
! echo $PROJECT_ID

# used if we need to create a new bucket
! gsutil mb -l $REGION gs://$BUCKET_NAME

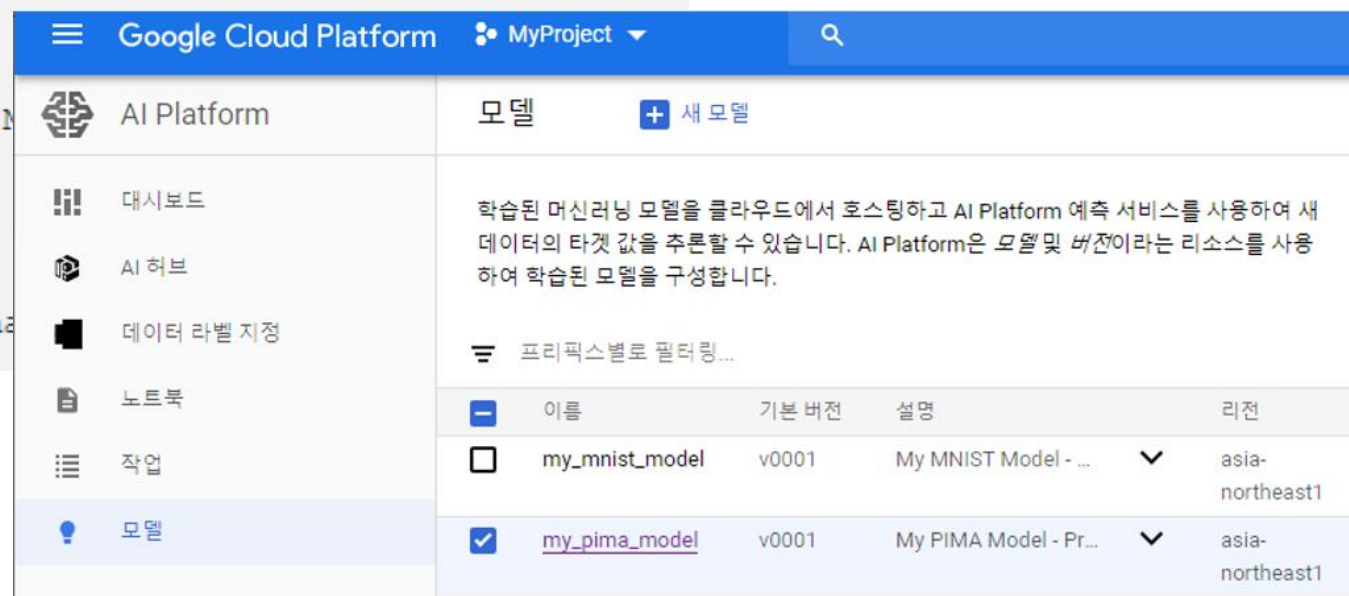
# used to copy the tensorflow model into the bucket
!gsutil cp -r {model_name} gs://{BUCKET_NAME}
```

## B — Create a model and a version in the Google Cloud AI Platform

```
# variable for the model creation
MODEL_NAME = "my_pima_model2"
MODEL_VERSION = "v0001"
MODEL_VERSION_NUM = '0001'

# create the model in the AI platform
! gcloud ai-platform models create $MODEL_NAME --regions $REGION

# create a version of the model
! gcloud ai-platform versions create {MODEL_NAME} \
  --model {MODEL_NAME} \
  --runtime-version 1.13 \
  --python-version 3.5 \
  --framework tensorflow \
  --origin gs://{BUCKET_NAME}/{model_name}
```



The screenshot shows the Google Cloud Platform AI Platform console. The left sidebar contains a navigation menu with options: 대시보드 (Dashboard), AI 허브 (AI Hub), 데이터 라벨 지정 (Data Labeling), 노트북 (Notebooks), 작업 (Jobs), and **모델 (Models)**. The main content area is titled '모델' (Models) and includes a '+ 새 모델' (New Model) button. Below this, there is a table listing existing models. The table has columns for '이름' (Name), '기본 버전' (Default Version), '설명' (Description), and '리전' (Region). Two models are listed: 'my\_mnist\_model' and 'my\_pima\_model'. The 'my\_pima\_model' row is selected, indicated by a checkmark in the first column.

이름	기본 버전	설명	리전
my_mnist_model	v0001	My MNIST Model - ...	asia-northeast1
<u>my_pima_model</u>	v0001	My PIMA Model - Pr...	asia-northeast1

### C — Make a prediction using the Google Cloud AI Platform REST API

```
# create the resource to the model web api
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "sapient-pen-260901-
dc0c62411b24.json"
project_id = 'sapient-pen-260901'
model_id = MODEL_NAME
model_path = "projects/{}/models/{}".format(project_id, model_id)
model_path += "/versions/v0001/" # if you want to run a specific
version
ml_resource = googleapiclient.discovery.build("ml", "v1").projects()

# function use to request a prediction from the web api of the model
# and get a reponse of the predctions
def predict(X):
    input_data_json = {"signature_name": "serving_default",
                       "instances": X.tolist()}
    request = ml_resource.predict(name=model_path,
body=input_data_json)
    response = request.execute()
    if "error" in response:
        raise RuntimeError(response["error"])
    return np.array([pred['dense_5'] for pred in
response["predictions"]])
```

### **Step 3 : Build the Flask web app**

팀이 구축해야 하는 프로젝트의 마지막 부분은 의료 전문가가 환자의 의료 데이터를 입력하고 진단 결과를 실시간으로(real time) 얻을 수 있는 **웹앱**이다.

우리 팀은 완벽한 웹앱을 개발하기 전에 Flask를 사용하여 MVP(Minimal Viable Product)를 구축하여 고객에게 보여 주기로 결정했다.

The complete code for the web app can be found on the project [github](#).

The following section highlights the important code sections.

## *Tales of a machine learning startup*

```
from flask import render_template
from app import app
from app.forms import LabForm

from flask import render_template, flash, redirect

import numpy as np
import pandas as pd

from sklearn.preprocessing import MinMaxScaler
import googleapiclient.discovery

import sys
import os

@app.route('/')
@app.route('/index')
def index():

    return render_template('index.html')

@app.route('/prediction', methods=['GET', 'POST'])
def lab():
    form = LabForm()

    if form.validate_on_submit():
```

```
# get the dorm data for the patient data and put into a form
for the
    X_test =
np.array([[float(form.preg.data), float(form.glucose.data), float(form.
blood.data), float(form.skin.data), float(form.insulin.data), float(form
.bmi.data), float(form.dpf.data), float(form.age.data)]])
    print(X_test.shape)
    print(X_test)

# in order to make a prediction we must scale the data using
the same scale as the one used to make
# model

# get the data for the diabetes data.
data = pd.read_csv('./diabetes.csv', sep=',')

# extract the X and y from the imported data
X = data.values[:, 0:8]
y = data.values[:, 8]

# use MinMaxScaler to fit a scaler object
scaler = MinMaxScaler()
scaler.fit(X)

# min max scale the data for the prediction
X_test = scaler.transform(X_test)

# create the resource to the model web api on GCP
MODEL_NAME = "my_pima_model2"
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "sapiant-pen-
260901-dc0c62411b24.json"
```

## *Tales of a machine learning startup*

```
project_id = 'sapien-pen-260901'
model_id = MODEL_NAME
model_path = "projects/{}/models/{}".format(project_id,
model_id)
model_path += "/versions/v0001/" # if you want to run a
specific version
ml_resource = googleapiclient.discovery.build("ml",
"v1").projects()

# format the data as a json to send to the web api
input_data_json = {"signature_name": "serving_default",
                    "instances": X_test.tolist()}
# make the prediction
request = ml_resource.predict(name=model_path,
body=input_data_json)
response = request.execute()
if "error" in response:
    raise RuntimeError(response["error"])

# extract the prediction from the response
predD = np.array([pred['dense_5'] for pred in
response["predictions"]])
print(predD[0][0])
res = predD[0][0]

return render_template('result.html', res=res)

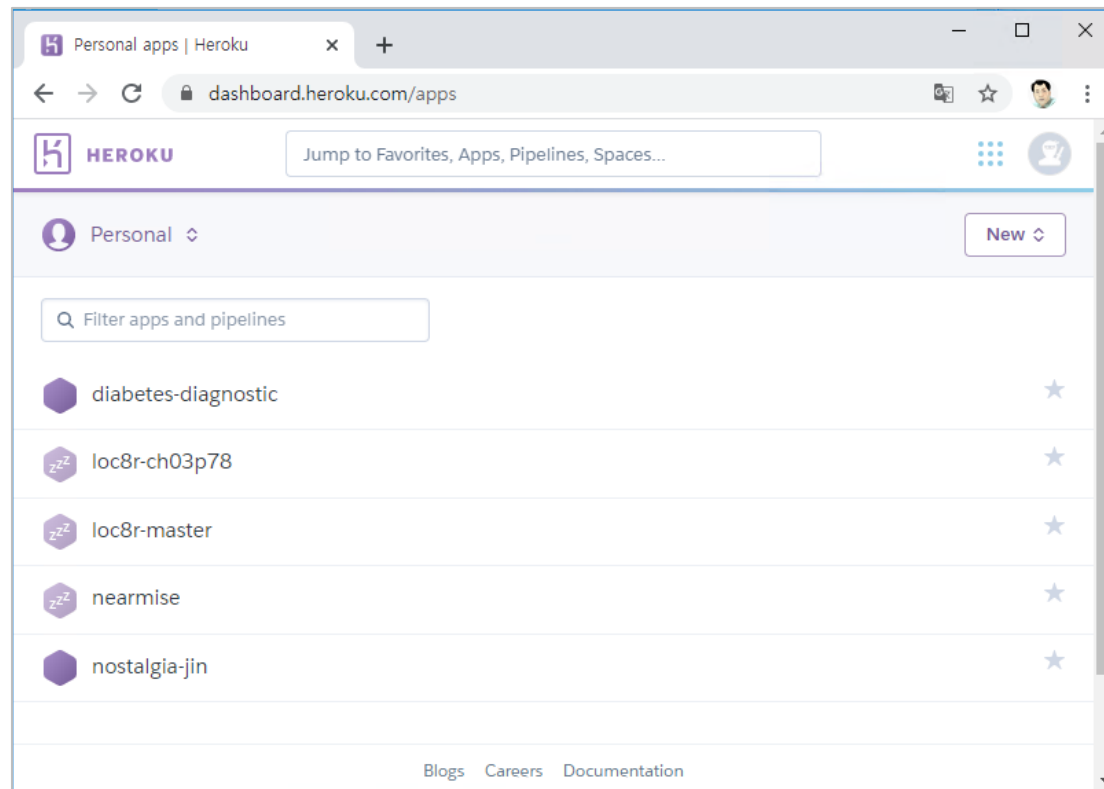
return render_template('prediction.html', form=form)
```

## Step 4 : Host in the cloud (Heroku platform)

Heroku는 2007년부터 사업을 시작한 최초의 PaaS 제공 업체 중 하나이다.

Heroku 플랫폼은 매우 유연하며 Python을 포함한 다양한 프로그래밍 언어들을 지원한다.

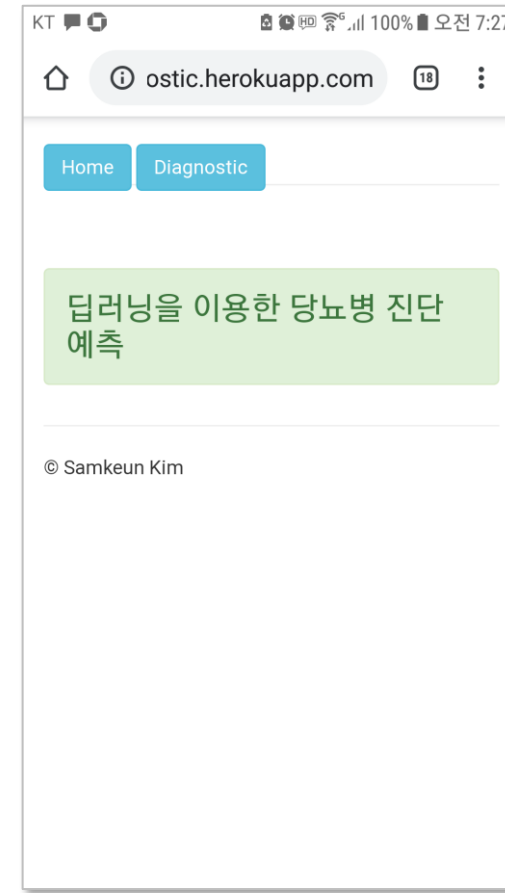
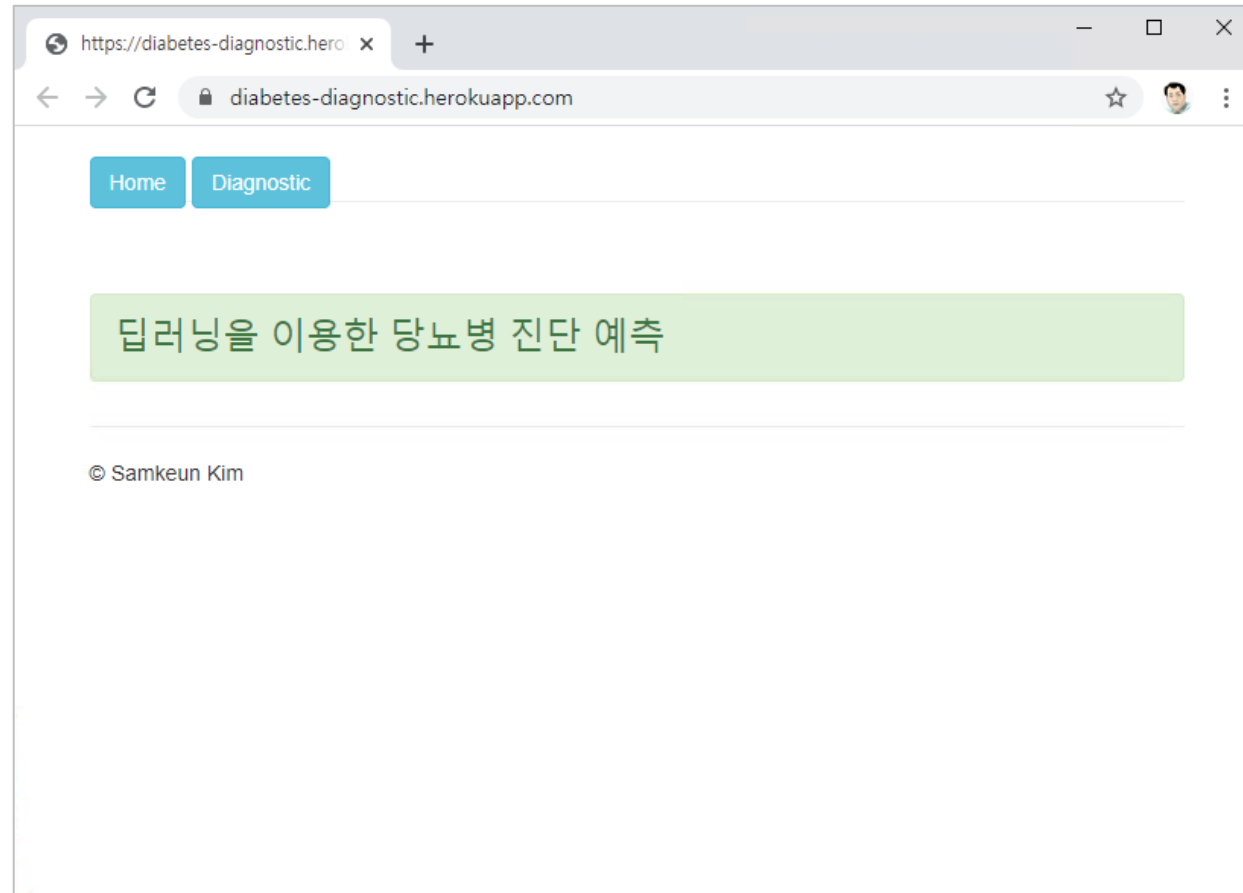
애플리케이션을 Heroku에 배포하기 위해 개발자는 Git을 사용하여 애플리케이션을 Heroku의 특수 Git 서버로 푸시 하면 애플리케이션의 설치, 업그레이드, 설정 및 배포가 자동으로 트리거 된다.





딥러닝을 이용한 당뇨병 진단 예측 웹앱:

## Diabetes Diagnostic Assistant



# Tales of a machine learning startup

https://diabetes-diagnostic.herokuapp.com/prediction

Home Diagnostic

## Fill in the medical data of the patient

# Pregnancies

0

Glucose

137

Blood pressure

40

Skin thickness

35

Insulin

168

BMI

43.1

DPF Score

2.38

Age

33

Submit

© Samkeun Kim

KT 100% 오전 7:28

ostic.herokuapp.com

Home Diagnostic

40

Skin thickness

35

Insulin

168

BMI

43.1

DPF Score

2.38

Age

33

Submit

© Samkeun Kim

