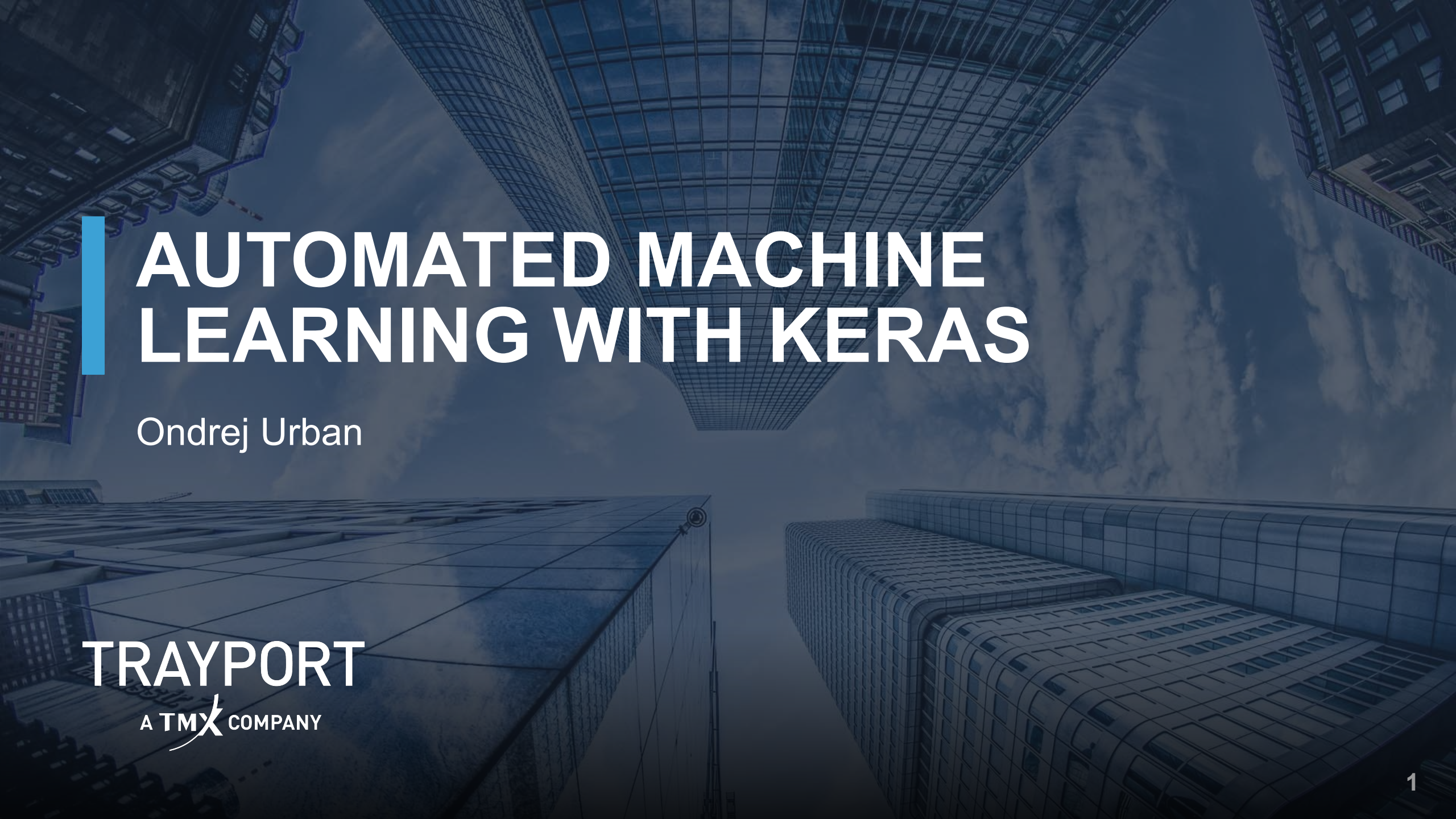


AUTOMATED MACHINE LEARNING WITH KERAS

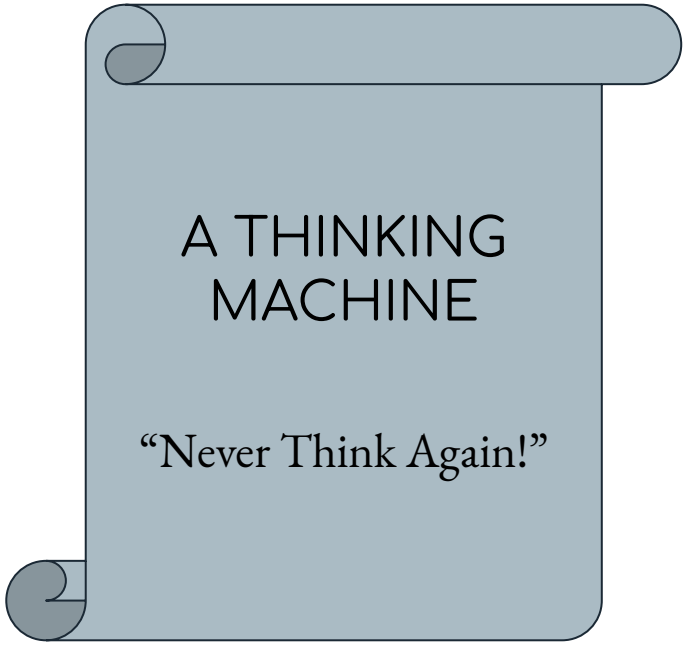
Ondrej Urban

TRAYPORT
A TMX COMPANY

STEPS TO PROGRESS TO A CYNIC

Computers

Laziness



Chasing Away Disappointment

Binary programming

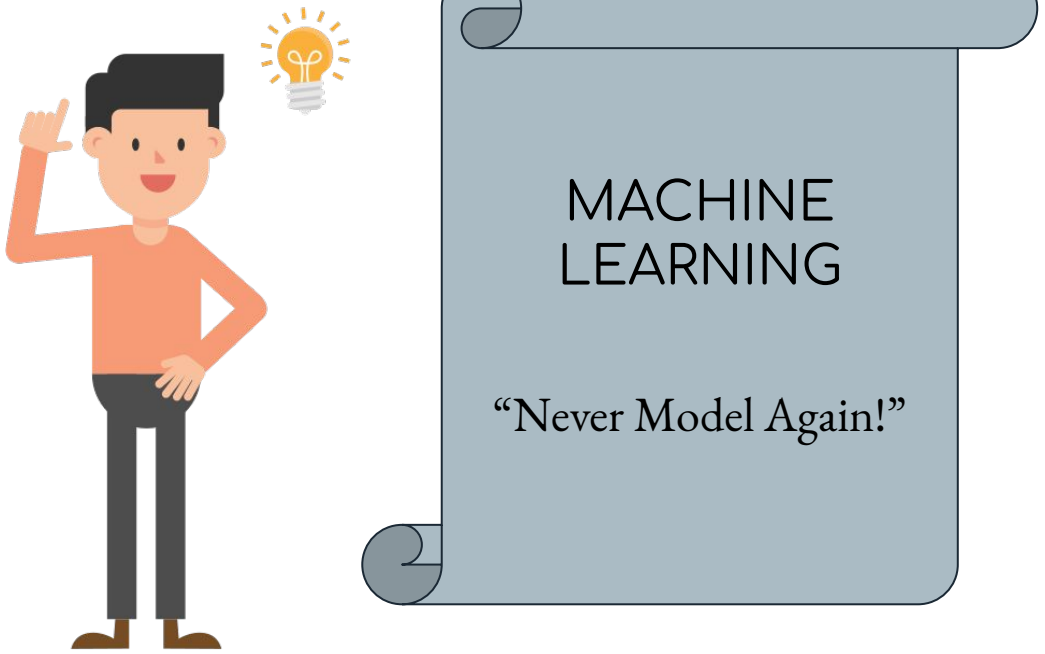
decades of development

Object-oriented programming

STEPS TO PROGRESS TO A CYNIC

Machine Learning

Laziness



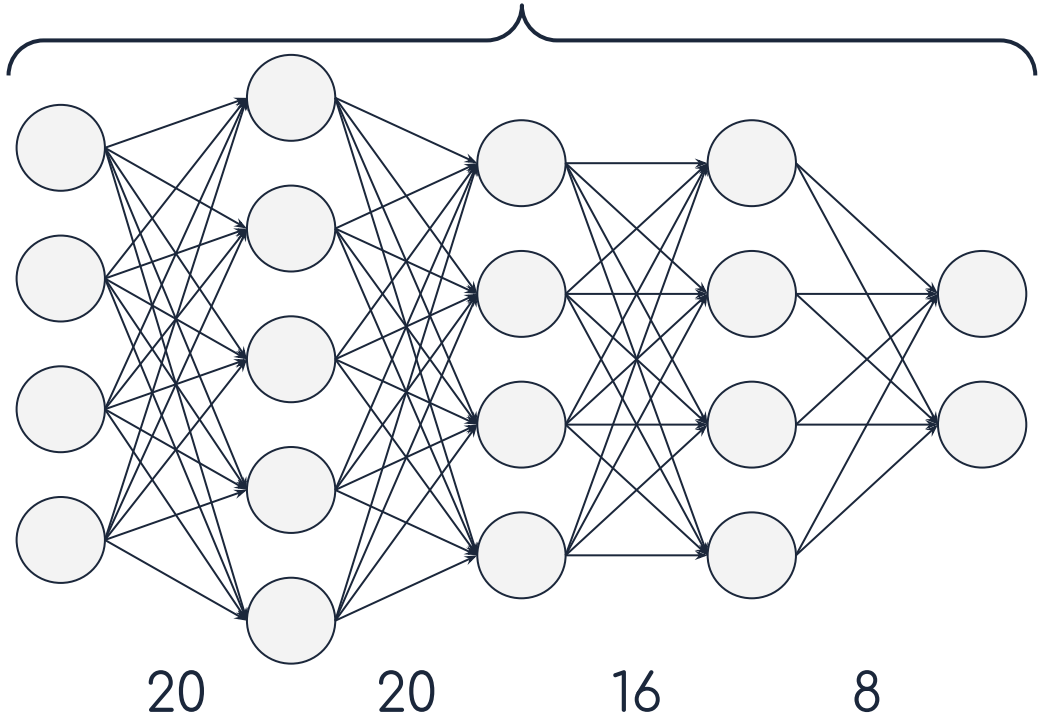
Chasing Away Disappointment

Individual algorithms

dedicated libraries, beginnings of automated machine learning

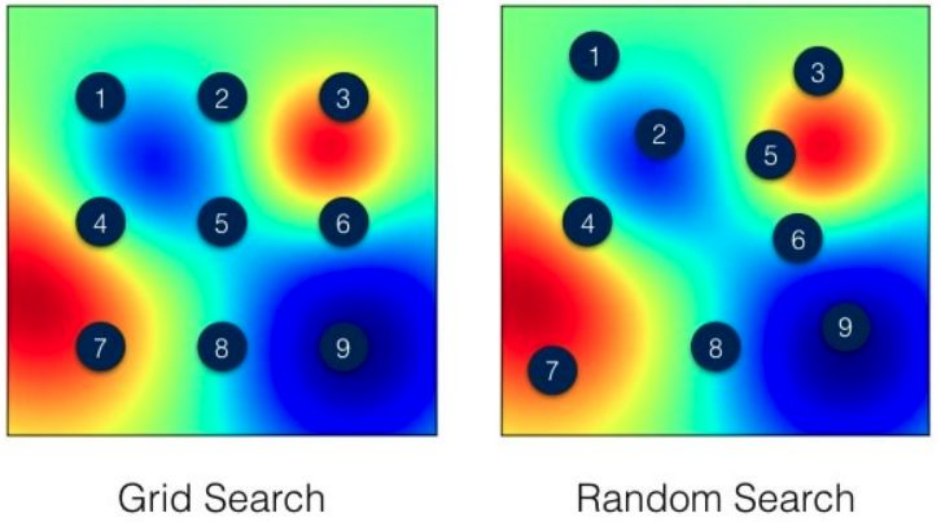
THIS TALK: HOW TO AUTOMATE HYPERPARAMETER SEARCH?

64 parameters, 3(-ish) hyperparameters



Example algorithms for hyperparameter search:

- Grid search
- Random search
- Bayesian optimization
- HYPERBAND algorithm

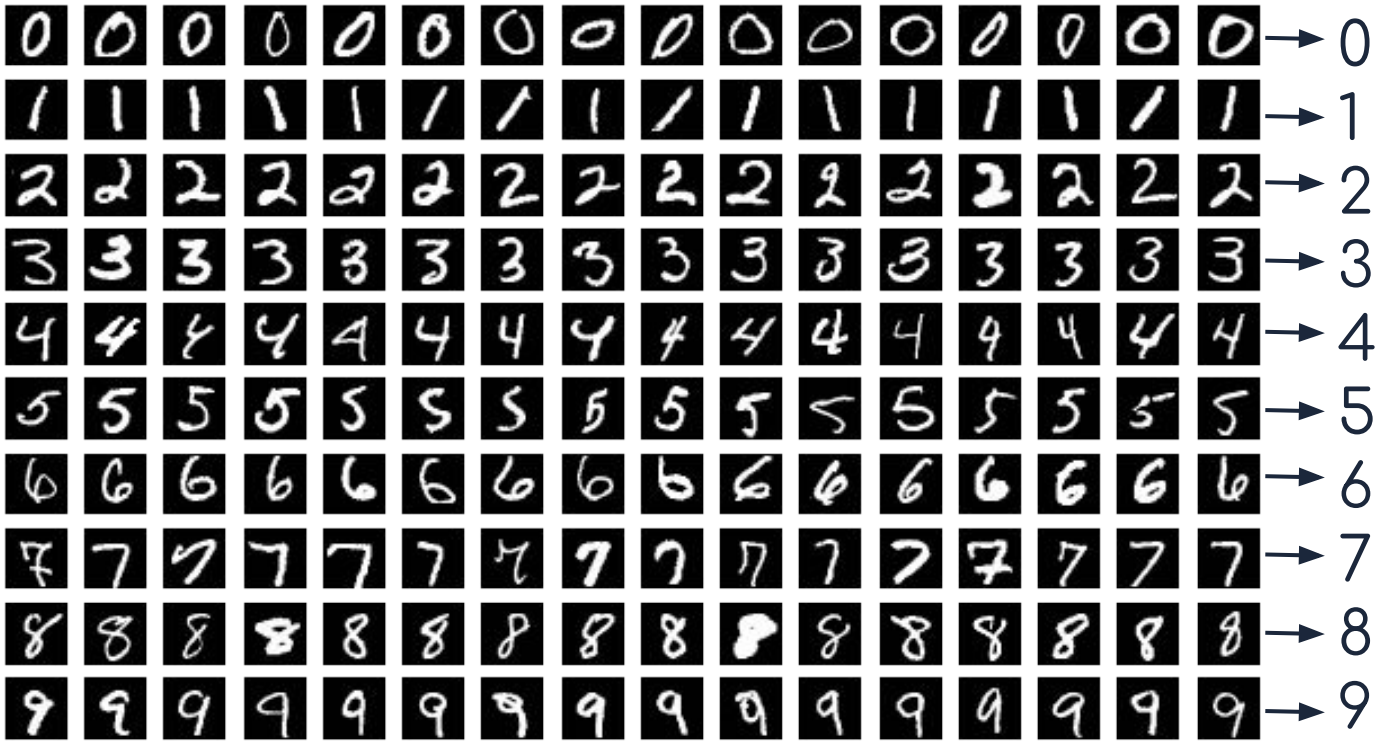


HOW TO AUTOMATE HYPERPARAMETER SEARCH *IN KERAS*?

- KerasTuner and AutoKeras
- How would three people with different levels of ML knowledge use them?
- machine learning for the masses

EXAMPLE SETTING

- MNIST digit classification
- example notebook available on git: https://git.visotech.at/ondrej.urban/automl-with-keras-europython-2021



1. DATA SCIENTIST

- (good) machine learning domain knowledge
- retains a lot of control
  - efficient dealing with possible issues
  - gain insights

1. DATA SCIENTIST

```
def build_model():
    # define layers
    # prepare model
    # compile model

    return model
```

1. DATA SCIENTIST

```
def build_model():
    # define layers
    # prepare model
    # compile model

    return model
```

```
conv_layer = Conv2D(
    filters=32,
    kernel_size=3,
)(input_layer)
```

```
dropout_layer = Dropout(
    rate=0.3,
)(previous_layer)
```

```
output_layer = Dense(
    units=10,
    activation="softmax",
)(dropout_layer)
```

1. DATA SCIENTIST

```
def build_model():
    # define layers
    # prepare model
    # compile model

    return model
```

```
conv_layer = Conv2D(
    filters=32,
    kernel_size=3,
)(input_layer)
```

```
dropout_layer = Dropout(
    rate=0.3,
)(previous_layer)
```

```
output_layer = Dense(
    units=10,
    activation="softmax",
)(dropout_layer)
```

```
model = Model(
    inputs=input_layer,
    outputs=output_layer,
)
```

1. DATA SCIENTIST

```
def build_model():
    # define layers
    # prepare model
    # compile model

    return model
```

```
conv_layer = Conv2D(
    filters=32,
    kernel_size=3,
)(input_layer)
```

```
dropout_layer = Dropout(
    rate=0.3,
)(previous_layer)
```

```
output_layer = Dense(
    units=10,
    activation="softmax",
)(dropout_layer)
```

```
model.compile(
    optimizer=Adam(
        learning_rate=1e-3
    ),
    # ...
)
```

```
model = Model(
    inputs=input_layer,
    outputs=output_layer,
)
```

1. DATA SCIENTIST

```
def build_model():
    # define layers
    # prepare model
    # compile model

    return model
```

```
conv_layer = Conv2D(
    filters=32,
    kernel_size=3,
)(input_layer)
```

```
dropout_layer = Dropout(
    rate=0.3,
)(previous_layer)
```

```
output_layer = Dense(
    units=10,
    activation="softmax",
)(dropout_layer)
```

```
model.compile(
    optimizer=Adam(
        learning_rate=1e-3
    ),
    # ...
)
```

```
model = Model(
    inputs=input_layer,
    outputs=output_layer,
)
```

1. DATA SCIENTIST

```
def build_model():
    # ...
    return model
```

```
def build_model(hp):
    # ...
    return model
```

```
conv_layer = Conv2D(
    filters=32,
    kernel_size=3,
)(input_layer)
```

```
conv_layer = Conv2D(
    filters=hp.Int("conv_filters", min_value=16,
                   max_value=128, steps=16),
    kernel_size=hp.Choice("kernel_size", [1, 3, 5]),
)(input_layer)
```

```
model.compile(
    optimizer=Adam(
        learning_rate=1e-3
    ),
    # ...
)
```

```
model.compile(
    optimizer=Adam(
        learning_rate=hp.Choice("lr",
                                [1e-2, 1e-3, 1e-4])
    ),
)
```

1. DATA SCIENTIST

```
from kerastuner.tuners import BayesianOptimization

def build_model(hp):
    # ...
    return model

tuner = BayesianOptimization(
   hypermodel=build_model,
   objective="val_accuracy",
)
```

1. DATA SCIENTIST

```
# (x, y), (val_x, val_y) = keras.datasets.mnist.load_data()
# ...plus normalization

tuner.search(
    x=x,
    y=y,
    validation_data=(
)
```

```
Trial 3 Complete [00h 00m 23s]
val_accuracy: 0.8956999778747559

Best val_accuracy So Far: 0.9441999793052673
Total elapsed time: 00h 01m 28s

Search: Running Trial #4

Hyperparameter    |Value             |Best Value So Far
num_conv_layers   |2                 |3
conv_units_1      |16                |16
kernel_size_1     |3                 |5
conv_units_2      |48                |32
kernel_size_2     |5                 |1
dropout           |0.3               |0.2
learning_rate     |0.0001            |0.01
conv_units_3      |48                |16
kernel_size_3     |5                 |1

 77/313 [======>.....................] - ETA: 16s - loss: 2.2823 - accuracy: 0.1367
```

2. TECHNICAL MANAGER

- High-level knowledge of domain knowledge machine learning
- concepts directly translated to code

2. TECHNICAL MANAGER

```
import autokeras as ak

input_node = ak.ImageInput()
normalization = ak.Normalization()(input_node)
convolution = ak.ConvBlock()(normalization)
output_node = ak.ClassificationHead()(convolution)

auto_model = ak.AutoModel(
    inputs=input_node,
    outputs=output_node,
    objective='val_accuracy',
)
```

2. TECHNICAL MANAGER

```
(x, y), (val_x, va

auto_model.fit(
    x=x,
    y=y,
    validation_dat
)
```

```
Trial 2 Complete [00h 00m 41s]
val_accuracy: 0.9695000052452087

Best val_accuracy So Far: 0.9715999960899353
Total elapsed time: 00h 01m 25s

Search: Running Trial #3

Hyperparameter    |Value             |Best Value So Far
conv_block_1/ke...|3                 |3
conv_block_1/se...|False             |False
conv_block_1/ma...|True              |True
conv_block_1/dr...|0                 |0
conv_block_1/nu...|2                 |2
conv_block_1/nu...|2                 |2
conv_block_1/fi...|32                |32
conv_block_1/fi...|32                |32
conv_block_1/fi...|32                |32
conv_block_1/fi...|512               |512
classification_...|flatten           |flatten
classification_...|0.25              |0
optimizer         |adam              |adam
learning_rate     |0.001             |0.001

 87/313 [=======>......................] - ETA: 22s - loss: 1.1560 - accuracy: 0.6430
```

3. DATA SCIENTIST OF THE FUTURE

- first steps towards what ML has been advertised as (admittedly, the toy problem lends itself well here)

3. DATA SCIENTIST OF THE FUTURE

```
import autokeras as ak

auto_model = ak.AutoModel(
    inputs=ak.ImageInput(),
    outputs=ak.ClassificationHead(),
)

(x, y), (val_x, val_y) = keras.datasets.mnist.load_data()

auto_model.fit(
    x=x,
    y=y,
    validation_data=(val_x, val_y),
)
```

3. DATA SCIENTIST OF THE FUTURE

```
import autoke

auto_model =

(x, y), (val_

auto_model.fi
    x=x,
    y=y,
    validatio
)
```

```
Trial 1 Complete [00h 04m 39s]
val_loss: 0.3742155134677887

Best val_loss So Far: 0.3742155134677887
Total elapsed time: 00h 04m 39s

Search: Running Trial #2

Hyperparameter    |Value             |Best Value So Far
image_block_1/n...|False             |False
image_block_1/a...|False             |False
image_block_1/b...|resnet            |resnet
image_block_1/r...|False             |False
image_block_1/r...|resnet50_v2       |resnet50
image_block_1/r...|False             |False
classification_...|flatten           |flatten
classification_...|0                 |0
optimizer         |adam              |adam
learning_rate     |0.001             |0.001

152/313 [=============>.............] - ETA: 1:33 - loss: 1.1156 - accuracy: 0.6705
```

SUMMARY

Automated machine learning can:

- make life easier for data scientists
- allow non-experts an easier access

Automated machine learning with Keras:

- KerasTuner and AutoKeras worth exploring

THANK YOU

TRAYPORT
A TMX COMPANY

info@trayport.com www.trayport.com

United Kingdom (Head Office)	Trayport Limited, 9 Appold Street, London EC2A 2AP, United Kingdom	+44 (0)20 7960 5500
Austria	Trayport Austria GmbH, Lemböckgasse 49/1A/5.OG, 1230 Vienna, Austria	+43 1 609 2290
Germany	Trayport Germany GmbH, Linzer Straße 11, 28359 Bremen, Germany	+49 (0)421 20109-0
Singapore (Asia Pacific)	Trayport Pte Ltd, One Raffles Place, Office Tower 1, #31-02, Singapore, 048616	+65 6411 4700