```
!pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.17.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensor
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from ten
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.10/dist-
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from t
Requirement already satisfied: h5py>=3.10.0 in /usr/local/lib/python3.10/dist-packages (from tensorfl
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tens
Requirement already satisfied: ml-dtypes<0.5.0,>=0.3.1 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from ten
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0de
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.10/dist-packages (from t
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflo
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tens
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (f
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorf
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from t
Requirement already satisfied: tensorboard<2.18,>=2.17 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: keras>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from tensorfl
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from as
Requirement already satisfied: rich in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->te
Requirement already satisfied: namex in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->t
Requirement already satisfied: optree in /usr/local/lib/python3.10/dist-packages (from keras>=3.2.0->
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (f
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from re
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from re
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tenso
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dis
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tenso
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from wer
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.10/dist-packages (from
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-i
```

```
from tensorflow.keras.datasets import imdb
(train_data, train_labels), (test_data, test_labels) = imdb.load_data(
    num_words=10000)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb.npz
17464789/17464789 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

```
print(train_data,train_data.shape)
```

```
[list([1, 14, 22, 16, 43, 530, 973, 1622, 1385, 65, 458, 4468, 66, 3941, 4, 173, 36, 256, 5, 25, 100,
 list([1, 194, 1153, 194, 8255, 78, 228, 5, 6, 1463, 4369, 5012, 134, 26, 4, 715, 8, 118, 1634, 14, 3
 list([1, 14, 47, 8, 30, 31, 7, 4, 249, 108, 7, 4, 5974, 54, 61, 369, 13, 71, 149, 14, 22, 112, 4, 24
 ...
 list([1, 11, 6, 230, 245, 6401, 9, 6, 1225, 446, 2, 45, 2174, 84, 8322, 4007, 21, 4, 912, 84, 2, 325
 list([1, 1446, 7079, 69, 72, 3305, 13, 610, 930, 8, 12, 582, 23, 5, 16, 484, 685, 54, 349, 11, 4120,
 list([1, 17, 6, 194, 337, 7, 4, 204, 22, 45, 254, 8, 106, 14, 123, 4, 2, 270, 2, 5, 2, 2, 732, 2098,
```

```
train_labels[0]
```

```
1
```

```
len(train_labels)
```

```
25000
```

```
len(train_labels)
```

```
25000
```

```
test_labels[0]
```

```
0
```

Release notes ✕ •••

Please follow our blog to see more information about new features, tips and tricks, and featured notebooks such as Analyzing a Bank Failure with Colab.

2024-09-23

- Improved code snippet search
- Updated Marketplace image and public local runtime container
- Improved the look-and-feel of interactive form dropdowns and checkboxes
- Fixed bugs
  - activating the skip link caused the notebook to scroll out of view
  - toggling a checkbox too much caused the page to crash
  - lightning fast drags could cause orphaned tabs
  - custom widgets snippet would show for local runtimes

Python package upgrades

- accelerate 0.32.1 -> 0.34.2
- arviz 0.18.0 -> 0.19
- autograd 1.6.2 -> 1.7.0
- bigframes 1.14.0 -> 1.18.0
- dask 2024.7.1 -> 2024.8.0
- distributed 2024.7.1 -> 2024.8.0
- duckdb 0.10.3 -> 1.1.0
- earthengine-api 0.1.416 -> 1.0.0
- flax 0.8.4 -> 0.8.5
- gdown 5.1.0 -> 5.2.0
- geemap 0.33.1 -> 0.34.3
- geopandas 0.14.4 -> 1.0.1
- google-cloud-aiplatform 1.59.0 -> 1.67.1
- google-cloud-bigquery-storage 2.25.0 -> 2.26.0
- holidays 0.54 -> 0.57
- huggingface-hub 0.23.5 -> 0.24.7
- ibis-framework 8.0.0 -> 9.2.0
- jax 0.4.26 -> 0.4.33
- jaxlib 0.4.26 -> 0.4.33
- kagglehub 0.2.9 -> 0.3.0
- lightgbm 4.4.0 -> 4.5.0
- matplotlib-venn 0.11.10 -> 1.1.1
- mizani 0.9.3 -> 0.11.4
- Pillow 9.4.0 -> 10.4.0
- plotly 5.15.0 -> 5.24.1
- plotnine 0.12.4 -> 0.13.6
- polars 0.20.2 -> 1.6.0
- progressbar2 4.2.0 -> 4.5.0
- PyDrive2 1.6.3 -> 1.20.0
- pymc 5.10.4 -> 5.16.2
- pytensor 2.18.6 -> 2.25.4
- scikit-image 0.23.2 -> 0.24.0
- scikit-learn 1.3.2 -> 1.5.2
- torch 2.3.1 -> 2.4.1
- torchaudio 2.3.1 -> 2.4.1
- torchvision 0.18.1 -> 0.19.1
- transformers 4.42.4 -> 4.44.2
- urllib3 2.0.7 -> 2.2.3
- xarray 2024.6.0 -> 2024.9.0

Python package inclusions

- bigquery-magics 0.2.0

2024-08-20

- TPU memory usage and utilization can now be checked with `!tpu-info`
- Gemini Chat responses are now grounded in relevant sources
- Added a new "Create Gemini API key" link in the user secrets

```
max([max(sequence) for sequence in test_data])
```

```
9999
```

```
word_index = imdb.get_word_index()
reverse_word_index = dict(
    [(value, key) for (key, value) in word_index.items()])
decoded_review = " ".join(
    [reverse_word_index.get(i - 3, "?") for i in train_data[0]])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb_word_index.jso
1641221/1641221 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

```
decoded_review
```

```
'? this film was just brilliant casting location scenery story direction everyone's really suited th
e part they played and you could just imagine being there robert ? is an amazing actor and now the s
ame being director ? father came from the same scottish island as myself so i loved the fact there w
as a real connection with this film the witty remarks throughout the film were great it was just bri
lliant so much that i bought the film as soon as it was released for ? and would recommend it to eve
ryone to watch and the fly fishing was amazing really cried at the end it was so sad and you know wh
at they say if you cry at a film it must have been good and this definitely was also ? to the two li
ttle boy's that played the ? of norman and paul they were just brilliant children are often left out
```

```
import numpy as np
def vectorize_sequences(sequences, dimension=10000):
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        for j in sequence:
            results[i, j] = 1.
    return results
```

```
#Data Vectorization
x_train = vectorize_sequences(train_data)
x_test = vectorize_sequences(test_data)
```

```
x_train[0]
```

```
array([0., 1., 1., ..., 0., 0., 0.])
```

```
x_test[0]
```

```
array([0., 1., 1., ..., 0., 0., 0.])
```

```
#label vectorization
y_train = np.asarray(train_labels).astype("float32")
y_test = np.asarray(test_labels).astype("float32")
```

```
#Building model using relu and compiling it
from tensorflow import keras
from tensorflow.keras import layers

model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

```
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
```

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

```
history = model.fit(partial_x_train,
                    partial_y_train,
```

panel

- Added a new "Gemini: Creating a prompt" snippet and touched up the existing "Gemini: Connecting to Gemini" snippet
- Added the ability to specify custom placeholder text for various interactive form params (see examples)
- Keyboard navigation a11y improvements to comments UI
- Various minor rendering improvements to interactive forms UI
- A11y improvements for the run button and header
- Updated tooltip styling
- A11y improvements for the file browser's disk usage bar
- On mobile, tooltips now trigger on long press
- On mobile, release notes updates will no longer display automatically
- Python package upgrades
    - astropy 5.3.4 -> 6.1.2
    - bigframes 1.11.1 -> 1.14.0
    - bokeh 3.3.4 -> 3.4.3
    - dask 2023.8.1 -> 2024.7.1
    - earthengine-api 0.1.412 -> 0.1.416
    - geopandas 0.13.2 -> 0.14.4
    - kagglehub 0.2.8 -> 0.2.9
    - keras 2.15.0 -> 3.4.1
    - lightgbm 4.1.0 -> 4.4.0
    - malloy 2023.1067 -> 2024.1067
    - numba 0.58.1 -> 0.60.0
    - numpy 1.25.2 -> 1.26.4
    - opencv-python 4.8.0.76 -> 4.10.0.84
    - pandas 2.0.3 -> 2.1.4
    - pandas-gbq 0.19.2 -> 0.23.1
    - panel 1.3.8 -> 1.4.5
    - requests 2.31.0 -> 2.32.3
    - scikit-learn 1.2.2. -> 1.3.2
    - scipy 1.11.4 -> 1.13.1
    - tensorboard 2.15.2 -> 2.17.0
    - tensorflow 2.15.0 -> 2.17.0
    - tf-keras 2.15.1 -> 2.17.0
    - xarray 2023.7.0 -> 2024.6.0
    - xgboost 2.0.3 -> 2.1.1
- Python package inclusions
    - einops 0.8.0

2024-07-22

- You can now embed Google sheets directly into Colab to streamline interactions with data with InteractiveSheet.

  Example:

  ```
  from google.colab import she
  sh = sheets.InteractiveSheet
  df = sh.as_df()
  ```

- Fixed multiple rendering bugs in cell editors with wide text content (i.e. text is no longer hidden or clipped)
- Fixed multiple accessibility issues in Colab's comments feature (e.g. proper keyboard focus management, added accessibility landmarks, etc)

```
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 60ms/step - accuracy: 0.6811 - loss: 0.6212 - val_accuracy: 0.8550 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.8861 - loss: 0.3631 - val_accuracy: 0.8832 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9214 - loss: 0.2557 - val_accuracy: 0.8882 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9367 - loss: 0.2001 - val_accuracy: 0.8828 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - accuracy: 0.9439 - loss: 0.1686 - val_accuracy: 0.8779 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 59ms/step - accuracy: 0.9570 - loss: 0.1365 - val_accuracy: 0.8850 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.9667 - loss: 0.1171 - val_accuracy: 0.8851 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9724 - loss: 0.1009 - val_accuracy: 0.8751 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9744 - loss: 0.0890 - val_accuracy: 0.8669 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9792 - loss: 0.0808 - val_accuracy: 0.8824 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9858 - loss: 0.0609 - val_accuracy: 0.8809 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9888 - loss: 0.0527 - val_accuracy: 0.8780 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9918 - loss: 0.0436 - val_accuracy: 0.8767 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9942 - loss: 0.0352 - val_accuracy: 0.8734 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 40ms/step - accuracy: 0.9953 - loss: 0.0303 - val_accuracy: 0.8721 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 46ms/step - accuracy: 0.9970 - loss: 0.0243 - val_accuracy: 0.8720 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 48ms/step - accuracy: 0.9975 - loss: 0.0205 - val_accuracy: 0.8713 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9975 - loss: 0.0187 - val_accuracy: 0.8731 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9987 - loss: 0.0144 - val_accuracy: 0.8715 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9994 - loss: 0.0113 - val_accuracy: 0.8716 - va
```

```
history_dict = history.history
history_dict.keys()
```

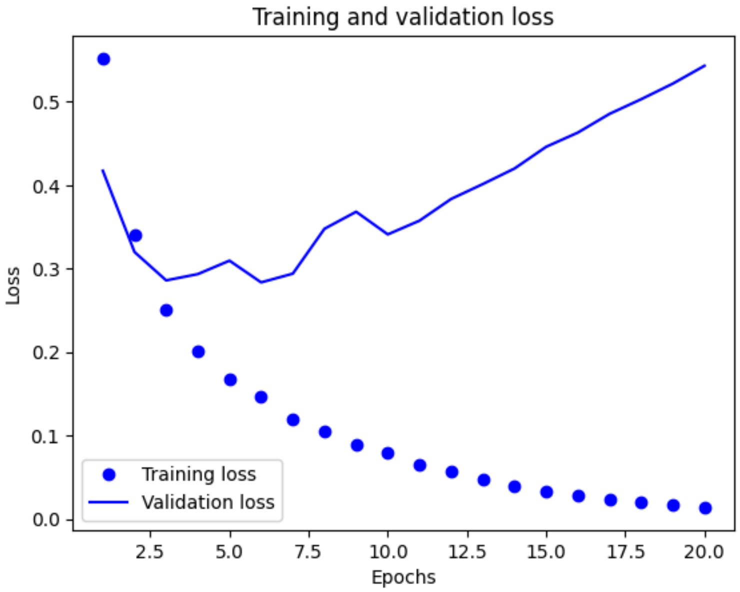
```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
#Plotting the training loss vs validation loss
import matplotlib.pyplot as plt
history_dict = history.history
loss_values = history_dict["loss"]
val_loss_values = history_dict["val_loss"]
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, "bo", label="Training loss")
plt.plot(epochs, val_loss_values, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()
```

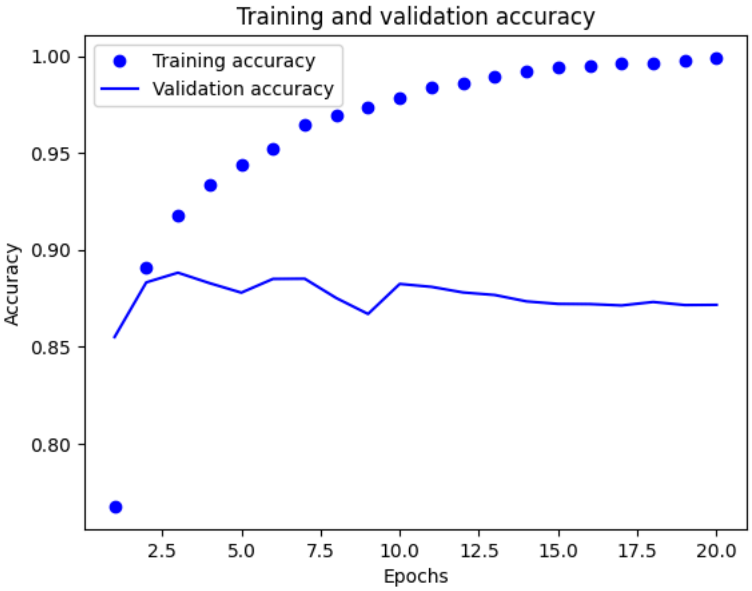
- Fixed bug where AI code generation would fail for extremely long broken code snippets
- Fixed multiple scrollbar bugs in the user secrets panel
- Added the ability for workspace admin to purchase Colab Pro and Pro+ Subscriptions for users
- Fixed bug where user secrets couldn't be moved to a tab
- Fixed several focus management accessibility issues in tabs, the table of contents, the left toolbar, and the run button
- Fixed bug where overflowing cells may be omitted when pasting from Google Sheets
- Fixed bug where the generate code button did not activate on touch
- Python package upgrades
    - bigframes 1.9.0 -> 1.11.1
    - cvxpy 1.3.4 -> 1.5.2
    - earthengine-api 0.1.408 -> 0.1.412
    - google-api-core 2.11.1 -> 2.19.1
    - google-api-python-client 2.84.0 -> 2.137.0
    - google-cloud-aiplatform 1.56.0 -> 1.59.0
    - google-cloud-bigquery 3.21.0 -> 3.25.0
    - google-cloud-core 2.3.3 -> 2.4.1
    - google-cloud-datastore 2.15.2 -> 2.19.0
    - google-cloud-firestore 2.11.1 -> 2.16.1
    - google-cloud-functions 1.13.3 -> 1.16.4
    - google-generativeai 0.5.4 -> 0.7.2
    - kagglehub 0.2.5 -> 0.2.8
    - pip 23.1.2 -> 24.1.2
    - setuptools 67.7.2 -> 71.0.4
    - sympy 1.12.1 -> 1.13.1
    - torch 2.3.0 -> 2.3.1
    - transformers 4.41.2 -> 4.42.4
- Python package inclusions
    - accelerate 0.32.1

2024-06-18

- Inline AI completions are now available to users on the free-of-charge tier
- Reduced latency for LSP and terminal connections
- Improved quality of inline completions
- Visual improvements to switch controls across Colab
- Various bug fixes, performance and a11y improvements to the user secrets panel
- Improved tooltip UX behavior
- Improved behavior when copying data from Google Sheets and pasting in Colab
- Scroll to cell fixes for single tabbed view and jump to cell command
- Improved tab header behavior



```
#Plotting training accuracy vs validatition accuracy
plt.clf()
acc = history_dict["accuracy"]
val_acc = history_dict["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training accuracy")
plt.plot(epochs, val_acc, "b", label="Validation accuracy")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```



```
model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
model.fit(x_train, y_train, epochs=4, batch_size=512)
results = model.evaluate(x_test, y_test)
```

```
Epoch 1/4
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 22ms/step - accuracy: 0.7282 - loss: 0.5605
Epoch 2/4
```

- A11y improvements for notebook-focused cells
- Python package upgrades
  - torch 2.2.1 -> 2.3.0
  - torchaudio 2.2.1 -> 2.3.0
  - torchvision 0.17.1 -> 0.18.0
  - torchtext 0.17.1 -> 0.18.0
  - google-cloud-aiplatform 1.51.0 -> 1.56.0
  - bigframes 1.5.0 -> 1.8.0
  - regex 2023.12.25 -> 2024.5.15

2024-05-13

- Code actions are now supported to automatically improve and refactor code. Code actions can be triggered by the keyboard shortcut "Ctrl/⌘ + ."
- Python package upgrades
  - bigframes 1.0.0 -> 1.5.0
  - google-cloud-aiplatform 1.47.0 -> 1.51.0
  - jax[tpu] 0.4.23 -> 0.4.26
- Python package inclusions
  - cudf 24.4.1

2024-04-15

- TPU v2 runtime is now available
- L4 runtime is now available for paid users
- New distributed fine-tuning Gemma tutorial on TPUs (GitHub)
- Symbol rename is now supported with keyboard shortcut F2
- Fixed bug causing inability to re-upload deleted files
- Fixed breaking bug in colabtools %upload_files_async
- Added syntax highlighting to %%writefile cells
- Cuda dependencies that come with Torch are cached for faster downloads for packages that require Torch and its dependencies (GitHub issue)
- Python package upgrades
  - bigframes 0.24.0 -> 1.0.0
  - duckdb 0.9.2 -> 0.10.1
  - google-cloud-aiplatform 1.43.0 -> 1.47.0
  - jax 0.4.23 -> 0.4.26

2024-03-13

- Fixed bug that sometimes caused UserSecrets to move / disappear
- Improved messaging for mounting drive in an unsupported environment (GitHub issue)
- Python package upgrades
  - torch 2.1.0 -> 2.2.1
  - torchaudio 2.1.0 -> 2.2.1
  - torchvision 0.16.0 -> 0.17.1
  - torchtext 0.16.0 -> 0.17.1
  - PyMC 5.7.2 -> 5.10.4
  - BigFrames 0.21.0 -> 0.24.0
  - google-cloud-aiplatform 1.42.1 -> 1.43.0
  - tornado 6.3.2 -> 6.3.3

```
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9015 - loss: 0.2864
Epoch 3/4
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - accuracy: 0.9254 - loss: 0.2141
Epoch 4/4
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.9361 - loss: 0.1811
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8839 - loss: 0.2897
```

```
results
```

```
[0.2888040542602539, 0.8860399723052979]
```

```
model.predict(x_test)
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
array([[0.20673797],
       [0.99810493],
       [0.8371271 ],
       ...,
       [0.07179032],
       [0.06511712],
       [0.4686889 ]], dtype=float32)
```

```
model_1_layer = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])

model_1_layer.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])

x_val1 = x_train[:10000]
partial_x_train = x_train[10000:]

y_val1 = y_train[:10000]
partial_y_train = y_train[10000:]


history1_layer = model_1_layer.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val1, y_val1))
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 89ms/step - accuracy: 0.7164 - loss: 0.5984 - val_accuracy: 0.8593 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.8915 - loss: 0.3697 - val_accuracy: 0.8733 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9137 - loss: 0.2878 - val_accuracy: 0.8847 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9247 - loss: 0.2383 - val_accuracy: 0.8872 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9379 - loss: 0.2080 - val_accuracy: 0.8858 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9452 - loss: 0.1808 - val_accuracy: 0.8853 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9525 - loss: 0.1629 - val_accuracy: 0.8831 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9516 - loss: 0.1532 - val_accuracy: 0.8862 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9597 - loss: 0.1368 - val_accuracy: 0.8806 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 35ms/step - accuracy: 0.9603 - loss: 0.1324 - val_accuracy: 0.8856 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 56ms/step - accuracy: 0.9696 - loss: 0.1116 - val_accuracy: 0.8856 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 61ms/step - accuracy: 0.9708 - loss: 0.1094 - val_accuracy: 0.8848 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 54ms/step - accuracy: 0.9732 - loss: 0.0991 - val_accuracy: 0.8840 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 50ms/step - accuracy: 0.9781 - loss: 0.0900 - val_accuracy: 0.8816 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 30ms/step - accuracy: 0.9805 - loss: 0.0843 - val_accuracy: 0.8813 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9829 - loss: 0.0793 - val_accuracy: 0.8808 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9825 - loss: 0.0757 - val_accuracy: 0.8794 - va
Epoch 18/20
```

2024-02-21

- Try out Gemma on Colab!
- Allow unicode in form text inputs
- Display documentation and link to source when displaying functions
- Display image-like ndarrays as images
- Improved UX around quick charts and execution error suggestions
- Released Marketplace image for the month of February (GitHub issue)
- Python package upgrades
  - bigframes 0.19.2 -> 0.21.0
  - regex 2023.6.3 -> 2023.12.25
  - spacy 3.6.1 -> 3.7.4
  - beautifulsoup4 4.11.2 -> 4.12.3
  - tensorflow-probability 0.22.0 -> 0.23.0
  - google-cloud-language 2.9.1 -> 2.13.1
  - google-cloud-aiplatform 1.39.0 -> 1.42.1
  - transformers 4.35.2 -> 4..37.2
  - pyarrow 10.0.1 -> 14.0.2

2024-01-29

- New Kaggle Notebooks <> Colab updates! Now you can:
  - Import directly from Colab without having to download/re-upload
  - Upload via link, by pasting Google Drive or Colab URLs
  - Export & run Kaggle Notebooks on Colab with 1 click
- Try these notebooks that talk to Gemini:
  - Gemini and Stable Diffusion
  - Learning with Gemini and ChatGPT
  - Talk to Gemini with Google's Speech to Text API
  - Sell lemonade with Gemini and Sheets
  - Generate images with Gemini and Vertex
- Python package upgrades
  - google-cloud-aiplatform 1.38.1 -> 1.39.0
  - bigframes 0.18.0 -> 0.19.2
  - polars 0.17.3 -> 0.20.2
  - gdown 4.6.6 -> 4.7.3 (GitHub issue)
  - tensorflow-hub 0.15.0 -> 0.16.0
  - flax 0.7.5 -> 0.8.0
- Python package inclusions
  - sentencepiece 0.1.99

2024-01-08

- Avoid nested scrollbars for large outputs by using `google.colab.output.no_verti` Example notebook
- Fix bug where downloading models from Hugging Face could freeze

```
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9868 - loss: 0.0680 - val_accuracy: 0.8783 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 42ms/step - accuracy: 0.9888 - loss: 0.0622 - val_accuracy: 0.8792 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 46ms/step - accuracy: 0.9909 - loss: 0.0582 - val_accuracy: 0.8729 - va
```

```
history_dict1 = history1_layer.history
history_dict1.keys()
```

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
import matplotlib.pyplot as plt
history_dict1 = history1_layer.history
loss_value1 = history_dict1["loss"]
val_loss_value1 = history_dict1["val_loss"]
epochs1 = range(1, len(loss_value1) + 1)

#Plotting graph of Training and Validation loss
plt.plot(epochs1, loss_value1, "ro", label="Training loss")
plt.plot(epochs1, val_loss_value1, "r", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()

#Plotting graph of Training and Validation Accuracy
plt.clf()
accuracy1 = history_dict1["accuracy"]
val_accuracy1 = history_dict1["val_accuracy"]
plt.plot(epochs1, accuracy1, "ro", label="Training accuracy")
plt.plot(epochs1, val_accuracy1, "r", label="Validation accuracy")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```

- Python package upgrades
  - huggingface-hub 0.19.4 -> 0.20.2
  - bigframes 0.17.0 -> 0.18.0

2023-12-18

- Expanded access to AI coding has arrived in Colab across 175 locales for all tiers of Colab users
- Improvements to display of ML-based inline completions (for eligible Pro/Pro+ users)
- Started a series of notebooks highlighting Gemini API capabilities
- Enable ⌘/Ctrl+L to select the full line in an editor
- Fixed bug where we weren't correctly formatting output from multiple execution results
- Python package upgrades
  - CUDA 11.8 to CUDA 12.2
  - tensorflow 2.14.0 -> 2.15.0
  - tensorboard 2.14.0 -> 2.15.0
  - keras 2.14.0 -> 2.15.0
  - Nvidia drivers 525.105.17 -> 535.104.05
  - tensorflow-gcs-config 2.14.0 -> 2.15.0
  - bigframes 0.13.0 -> 0.17.0
  - geemap 0.28.2 -> 0.29.6
  - pyarrow 9.0.0 -> 10.0.1
  - google-generativeai 0.2.2 -> 0.3.1
  - jax 0.4.20 —> 0.4.23
  - jaxlib 0.4.20 —> 0.4.23
- Python package inclusions
  - kagglehub 0.1.4
  - google-cloud-aiplatform 1.38.1

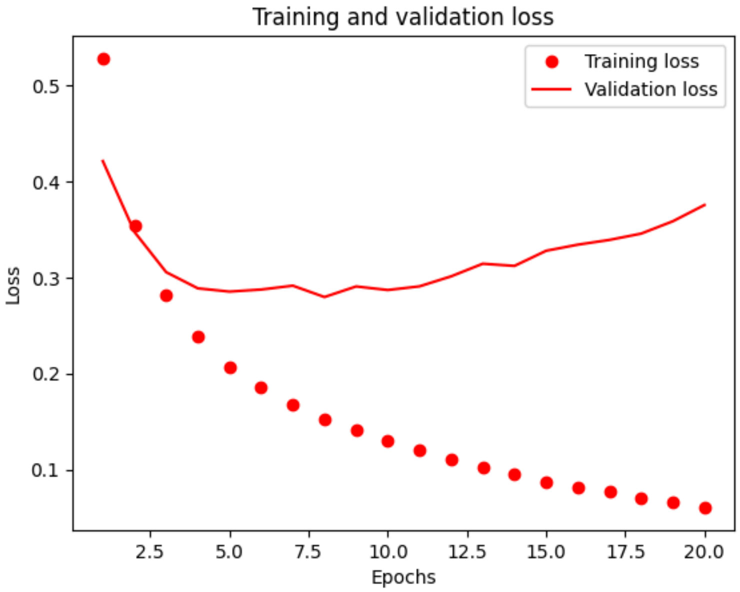
2023-11-27

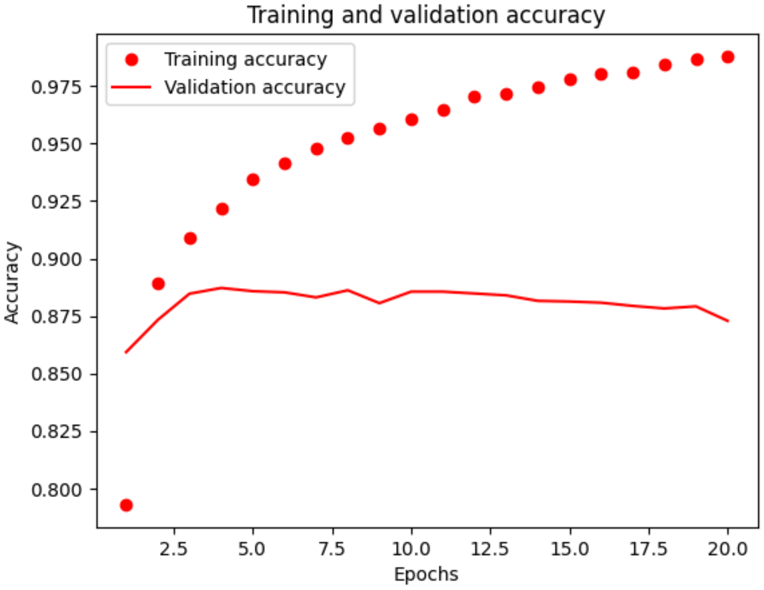
- Removed warning when calling `await` to make it render as code
- Added "Run selection" to the cell context menu
- Added highlighting for the `%%python` cell magic
- Launched AI coding features for Pro/Pro+ users in more locales
- Python package upgrades
  - bigframes 0.12.0 -> 0.13.0
- Python package inclusions
  - transformers 4.35.2
  - google-generativeai 0.2.2

2023-11-08

- Launched Secrets, for safe storage of private keys on Colab (tweet)
- Fixed issue where TensorBoard would not load (#3990)
- Python package upgrades
  - lightgbm 4.0.0 -> 4.1.0
  - bigframes 0.10.0 -> 0.12.0
  - bokeh 3.2.2 -> 3.3.0
  - duckdb 0.8.1 -> 0.9.1
  - numba 0.56.4 -> 0.58.1
  - tweepy 4.13.0 -> 4.14.0
  - jax 0.4.16 -> 0.4.20
  - jaxlib 0.4.16 -> 0.4.20

2023-10-23





```
model_1_layer = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])

model_1_layer.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
model_1_layer.fit(x_train, y_train, epochs=5, batch_size=512)
result_1_layer = model_1_layer.evaluate(x_test, y_test)
```

```
Epoch 1/5
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 23ms/step - accuracy: 0.7415 - loss: 0.5424
Epoch 2/5
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.9048 - loss: 0.2927
Epoch 3/5
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 22ms/step - accuracy: 0.9202 - loss: 0.2357
Epoch 4/5
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9265 - loss: 0.2061
Epoch 5/5
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 31ms/step - accuracy: 0.9382 - loss: 0.1814
782/782 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8858 - loss: 0.2839
```

```
print(result_1_layer)

#The loss on the test set is 0.2778, and the accuracy is 88.85%.
```

- Updated the **Open notebook** dialog for better usability and support for smaller screen sizes
- Added smart paste support for data from Google Sheets for R notebooks
- Enabled showing release notes in a tab
- Launched AI coding features for Pro/Pro+ users in Australia AU Canada CA India IN and Japan JP (tweet)
- Python package upgrades
  - earthengine-api 0.1.357 -> 0.1.375
  - flax 0.7.2 -> 0.7.4
  - geemap 0.27.4 -> 0.28.2
  - jax 0.4.14 -> 0.4.16
  - jaxlib 0.4.14 -> 0.4.16
  - keras 2.13.1 -> 2.14.0
  - tensorboard 2.13.0 -> 2.14.1
  - tensorflow 2.13.0 -> 2.14.0
  - tensorflow-gcs-config 2.13.0 -> 2.14.0
  - tensorflow-hub 0.14.0 -> 0.15.0
  - tensorflow-probability 0.20.1 -> 0.22.0
  - torch 2.0.1 -> 2.1.0
  - torchaudio 2.0.2 -> 2.1.0
  - torchtext 0.15.2 -> 0.16.0
  - torchvision 0.15.2 -> 0.16.0
  - xgboost 1.7.6 -> 2.0.0
- Python package inclusions
  - bigframes 0.10.0
  - malloy 2023.1056

2023-09-22

- Added the ability to scope an AI generated suggestion to a specific Pandas dataframe (tweet)
- Added Colab link previews to Docs (tweet)
- Added smart paste support for data from Google Sheets
- Increased font size of dropdowns in interactive forms
- Improved rendering of the notebook when printing
- Python package upgrades
  - tensorflow 2.12.0 -> 2.13.0
  - tensorboard 2.12.3 -> 2.13.0
  - keras 2.12.0 -> 2.13.1
  - tensorflow-gcs-config 2.12.0 -> 2.13.
  - scipy 1.10.1-> 1.11.2
  - cython 0.29.6 -> 3.0.2
- Python package inclusions
  - geemap 0.26.0

2023-08-18

- Added "Change runtime type" to the menu in the connection button
- Improved auto-reconnection to an already running notebook (#3764)
- Increased the specs of our highmem machines for Pro users
- Fixed `add-apt-repository` command on Ubuntu 22.04 runtime (#3867)
- Python package upgrades

```
[0.28184446692466736, 0.8876399993896484]
```

```
model_1_layer.predict(x_test)
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
array([[0.27810204],
       [0.9997605 ],
       [0.7863883 ],
       ...,
       [0.13692543],
       [0.09721168],
       [0.61300015]], dtype=float32)
```

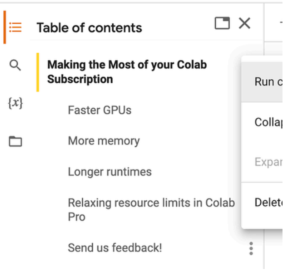
```
model_3_layers = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model_3_layers.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
x_val3 = x_train[:10000]
partial_x_train = x_train[10000:]

y_val3 = y_train[:10000]
partial_y_train = y_train[10000:]

history_3_layers = model_3_layers.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val3, y_val3))
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 56ms/step - accuracy: 0.6635 - loss: 0.6302 - val_accuracy: 0.8623 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 60ms/step - accuracy: 0.8881 - loss: 0.3648 - val_accuracy: 0.8836 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 31ms/step - accuracy: 0.9227 - loss: 0.2466 - val_accuracy: 0.8682 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9332 - loss: 0.1965 - val_accuracy: 0.8816 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9473 - loss: 0.1594 - val_accuracy: 0.8882 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9631 - loss: 0.1238 - val_accuracy: 0.8870 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9687 - loss: 0.1040 - val_accuracy: 0.8755 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9744 - loss: 0.0873 - val_accuracy: 0.8753 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9769 - loss: 0.0779 - val_accuracy: 0.8625 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9830 - loss: 0.0648 - val_accuracy: 0.8803 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 44ms/step - accuracy: 0.9885 - loss: 0.0506 - val_accuracy: 0.8706 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 47ms/step - accuracy: 0.9904 - loss: 0.0392 - val_accuracy: 0.8672 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 31ms/step - accuracy: 0.9916 - loss: 0.0352 - val_accuracy: 0.8750 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9941 - loss: 0.0268 - val_accuracy: 0.8742 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9969 - loss: 0.0204 - val_accuracy: 0.8753 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9985 - loss: 0.0143 - val_accuracy: 0.8701 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9994 - loss: 0.0106 - val_accuracy: 0.8506 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9891 - loss: 0.0311 - val_accuracy: 0.8715 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9960 - loss: 0.0177 - val_accuracy: 0.8697 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9998 - loss: 0.0053 - val_accuracy: 0.8695 - va
```

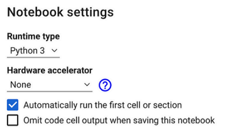
  - bokeh 2.4.3 -> 3.2.2
  - cmake 3.25.2 -> 3.27.2
  - cryptography 3.4.8 -> 41.0.3
  - dask 2022.12.1 -> 2023.8.0
  - distributed 2022.12.1 -> 2023.8.0
  - earthengine-api 0.1.358 -> 0.1.364
  - flax 0.7.0 -> 0.7.2
  - ipython-sql 0.4.0 -> 0.5.0
  - jax 0.4.13 -> 0.4.14
  - jaxlib 0.4.13 -> 0.4.14
  - lightgbm 3.3.5 -> 4.0.0
  - mkl 2019.0 -> 2023.2.0
  - notebook 6.4.8 -> 6.5.5
  - numpy 1.22.4 -> 1.23.5
  - opencv-python 4.7.0.72 -> 4.8.0.76
  - pillow 8.4.0 -> 9.4.0
  - plotly 5.13.1 -> 5.15.0
  - prettytable 0.7.2 -> 3.8.0
  - pytensor 2.10.1 -> 2.14.2
  - spacy 3.5.4 -> 3.6.1
  - statsmodels 0.13.5 -> 0.14.0
  - xarray 2022.12.0 -> 2023.7.0
- Python package inclusions
  - PyDrive2 1.6.3

2023-07-21

- Launched auto-plotting for dataframes, available using the chart button that shows up alongside datatables (post)
- Added a menu to the table of contents to support running a section or collapsing/expanding sections (post)



- Added an option to automatically run the first cell or section, available under Edit -> Notebook settings (post)


Notebook settings

Runtime type
Python 3

Hardware accelerator
None

☑ Automatically run the first cell or section
□ Omit code cell output when saving this notebook

- Launched Pro/Pro+ to Algeria, Argentina, Chile, Ecuador, Egypt, Ghana, Kenya, Malaysia, Nepal, Nigeria, Peru, Rwanda, Saudi Arabia, South Africa, Sri Lanka, Tunisia, and Ukraine (tweet)
- Added a command, "Toggle tab moves focus" for toggling tab trapping in the editor (Tools -> Command palette, "Toggle tab moves focus")
- Fixed issue where `files.upload()` was

```
history_dict_3 = history_3_layers.history
history_dict_3.keys()
```

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
loss_val3 = history_dict_3["loss"]
val_loss_val3 = history_dict_3["val_loss"]
epochs3 = range(1, len(loss_val3) + 1)
plt.plot(epochs3, loss_val3, "go", label="Training loss")
plt.plot(epochs3, val_loss_val3, "g", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()
```

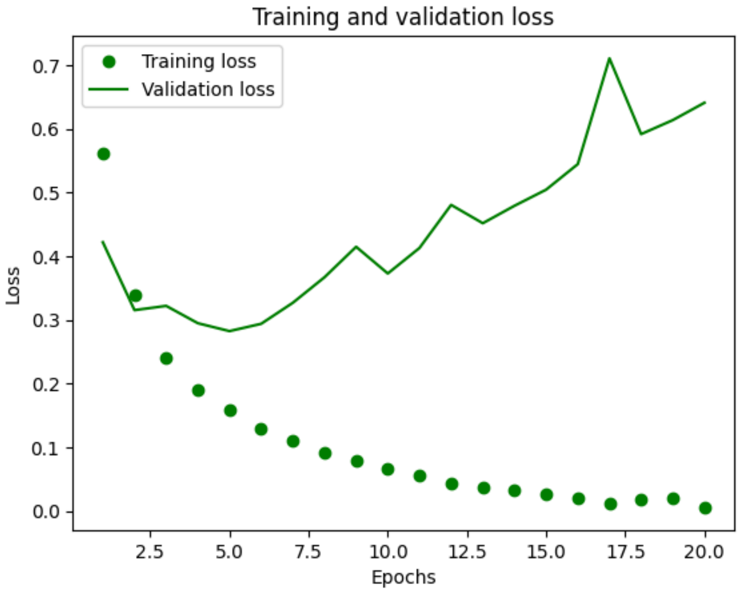
```
plt.clf()
accuracy3 = history_dict_3["accuracy"]
val_accuracy3 = history_dict_3["val_accuracy"]
plt.plot(epochs3, accuracy3, "go", label="Training acc")
plt.plot(epochs3, val_accuracy3, "g", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```

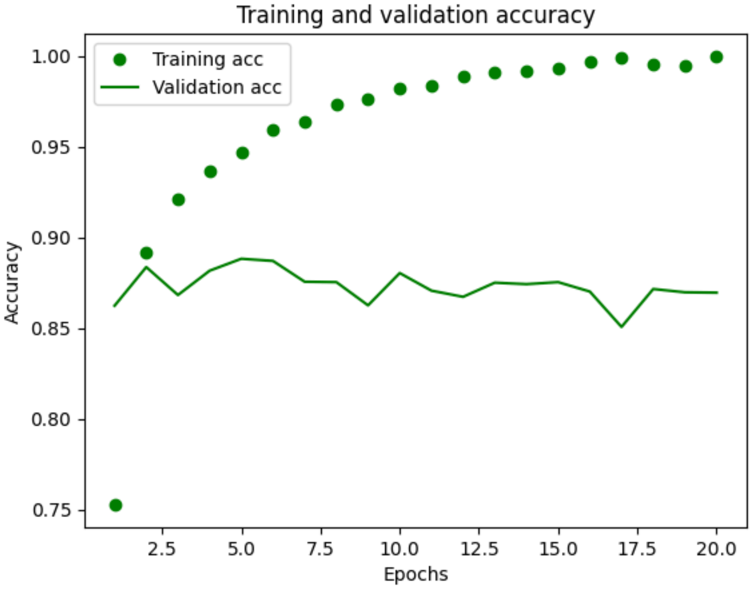
sometimes returning an incorrect filename (#1550)

- Fixed f-string syntax highlighting bug (#3802)
- Disabled ambiguous characters highlighting for commonly used LaTeX characters (#3648)
- Upgraded Ubuntu from 20.04 LTS to 22.04 LTS
- Updated the Colab Marketplace VM image
- Python package upgrades:
  - autograd 1.6.1 -> 1.6.2
  - drivefs 76.0 -> 77.0
  - flax 0.6.11 -> 0.7.0
  - earthengine-api 0.1.357 -> 0.1.358
  - GDAL 3.3.2->3.4.3
  - google-cloud-bigquery-storage 2.20.0 -> 2.22.2
  - gspread-dataframe 3.0.8 -> 3.3.1
  - holidays 0.27.1 -> 0.29
  - jax 0.4.10 -> jax 0.4.13
  - jaxlib 0.4.10 -> jax 0.4.13
  - jupyterlab-widgets 3.0.7 -> 3.0.8
  - nbformat 5.9.0 -> 5.9.1
  - opencv-python-headless 4.7.0.72 -> 4.8.0.74
  - pygame 2.4.0 -> 2.5.0
  - spacy 3.5.3 -> 3.5.4
  - SQLAlchemy 2.0.16 -> 2.0.19
  - tabulate 0.8.10 -> 0.9.0
  - tensorflow-hub 0.13.0 -> 0.14.0

2023-06-23

- Launched AI coding features to subscribed users starting with Pro+ users in the US (tweet, post)
- Added the Kernel Selector in the Notebook Settings (tweet)
- Fixed double space trimming issue in markdown #3766
- Fixed run button indicator not always centered #3609
- Fixed inconsistencies for automatic indentation on multi-line #3697
- Upgraded Python from 3.10.11 to 3.10.12
- Python package updates:
  - duckdb 0.7.1 -> 0.8.1
  - earthengine-api 0.1.350 -> 0.1.357
  - flax 0.6.9 -> 0.6.11
  - google-cloud-bigquery 3.9.0 -> 3.10.0
  - google-cloud-bigquery-storage 2.19.1 -> 2.20.0
  - grpcio 1.54.0 -> 1.56.0
  - holidays 0.25 -> 0.27.1
  - nbformat 5.8.0 -> 5.9.0
  - prophet 1.1.3 -> 1.1.4
  - pydata-google-auth 1.7.0 -> 1.8.0
  - spacy 3.5.2 -> 3.5.3
  - tensorboard 2.12.2 -> 2.12.3
  - xgboost 1.7.5 -> 1.7.6
- Python package inclusions:
  - gcsfs 2023.6.0
  - geopandas 0.13.2
  - google-cloud-bigquery-connection 1.12.0
  - google-cloud-functions 1.13.0
  - grpc-google-iam-v1 0.12.6





```
model_3_layers = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

```
model_3_layers.compile(optimizer='rmsprop',
              loss='binary_crossentropy',
              metrics=['accuracy'])

model_3_layers.fit(x_train, y_train, epochs=3, batch_size=512)
results_3_layers = model_3_layers.evaluate(x_test, y_test)
```

```
Epoch 1/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 22ms/step - accuracy: 0.6770 - loss: 0.6332
Epoch 2/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 21ms/step - accuracy: 0.8907 - loss: 0.3257
Epoch 3/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9195 - loss: 0.2245
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8656 - loss: 0.3314
```

```
print(results_3_layers)
```

```
[0.3260330259799957, 0.8672800064086914]
```

    - multidict 6.0.4
    - tensorboard-data-server 0.7.1

2023-06-02

- Released the new site colab.google
- Published Colab's Docker runtime image to us-docker.pkg.dev/colab-images/public/runtime (tweet, instructions)
- Launched support for Google children accounts (tweet)
- Launched DagsHub integration (tweet, post)
- Upgraded to Monaco Editor Version 0.37.1
- Fixed various Vim keybinding bugs
- Fixed issue where the N and P letters sometimes couldn't be typed (#3664)
- Fixed rendering support for compositional inputs (#3660, #3679)
- Fixed lag in notebooks with lots of cells (#3676)
- Improved support for R by adding a Runtime type notebook setting (Edit -> Notebook settings)
- Improved documentation for connecting to a local runtime (Connect -> Connect to a local runtime)
- Python package updates:
    - holidays 0.23 -> 0.25
    - jax 0.4.8 -> 0.4.10
    - jaxlib 0.4.8 -> 0.4.10
    - pip 23.0.1 -> 23.1.2
    - tensorflow-probability 0.19.0 -> 0.20.1
    - torch 2.0.0 -> 2.0.1
    - torchaudio 2.0.1 -> 2.0.2
    - torchdata 0.6.0 -> 0.6.1
    - torchtext 0.15.1 -> 0.15.2
    - torchvision 0.15.1 -> 0.15.2
    - tornado 6.2 -> 6.3.1

2023-05-05

- Released GPU type selection for paid users, allowing them to choose a preferred NVidia GPU
- Upgraded R from 4.2.3 to 4.3.0
- Upgraded Python from 3.9.16 to 3.10.11
- Python package updates:
    - attrs 22.2.0 -> attrs 23.1.0
    - earthengine-api 0.1.349 -> earthengine-api 0.1.350
    - flax 0.6.8 -> 0.6.9
    - grpcio 1.53.0 -> 1.54.0
    - nbclient 0.7.3 -> 0.7.4
    - tensorflow-datasets 4.8.3 -> 4.9.2
    - termcolor 2.2.0 -> 2.3.0
    - zict 2.2.0 -> 3.0.0

2023-04-14

- Python package updates:
    - google-api-python-client 2.70.0 -> 2.84.0
    - google-auth-oauthlib 0.4.6 -> 1.0.0
    - google-cloud-bigquery 3.4.2 -> 3.9.0
    - google-cloud-datastore 2.11.1 -> 2.15.1
    - google-cloud-firestore 2.7.3 -> 2.11.0

```
model_3_layers.predict(x_test)
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step
array([[0.37547025],
       [0.99891275],
       [0.9890443 ],
       ...,
       [0.19173409],
       [0.1914591 ],
       [0.7925559 ]], dtype=float32)
```

```
model_32_units = keras.Sequential([
    layers.Dense(32, activation="relu"),
    layers.Dense(32, activation="relu"),
    layers.Dense(32, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
#model compilation
model_32_units.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
#model validation
x_val_32 = x_train[:10000]
partial_x_train = x_train[10000:]

y_val_32 = y_train[:10000]
partial_y_train = y_train[10000:]


history_32_units = model_32_units.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val_32, y_val_32))
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 63ms/step - accuracy: 0.6712 - loss: 0.6093 - val_accuracy: 0.8322 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 38ms/step - accuracy: 0.8812 - loss: 0.3260 - val_accuracy: 0.8621 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 36ms/step - accuracy: 0.9092 - loss: 0.2387 - val_accuracy: 0.8729 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 39ms/step - accuracy: 0.9339 - loss: 0.1825 - val_accuracy: 0.8751 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.9545 - loss: 0.1349 - val_accuracy: 0.8592 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 56ms/step - accuracy: 0.9534 - loss: 0.1270 - val_accuracy: 0.8846 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 43ms/step - accuracy: 0.9723 - loss: 0.0865 - val_accuracy: 0.8816 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.9785 - loss: 0.0737 - val_accuracy: 0.8806 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.9865 - loss: 0.0542 - val_accuracy: 0.8777 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.9872 - loss: 0.0474 - val_accuracy: 0.8781 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.9905 - loss: 0.0386 - val_accuracy: 0.8759 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.9947 - loss: 0.0259 - val_accuracy: 0.8754 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.9989 - loss: 0.0133 - val_accuracy: 0.8721 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 36ms/step - accuracy: 0.9932 - loss: 0.0299 - val_accuracy: 0.8737 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 49ms/step - accuracy: 0.9996 - loss: 0.0063 - val_accuracy: 0.8731 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 58ms/step - accuracy: 0.9935 - loss: 0.0250 - val_accuracy: 0.8726 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 50ms/step - accuracy: 0.9997 - loss: 0.0039 - val_accuracy: 0.8701 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.9954 - loss: 0.0169 - val_accuracy: 0.8716 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 1.0000 - loss: 0.0020 - val_accuracy: 0.8719 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.9998 - loss: 0.0021 - val_accuracy: 0.8426 - va
```

```
history_dict_32 = history_32_units.history
history_dict_32.keys()
```

- google-cloud-language 2.6.1 -> 2.9.1
- google-cloud-storage 2.7.0 -> 2.8.0
- google-cloud-translate 3.8.4 -> 3.11.1
- networkx 3.0 -> 3.1
- notebook 6.3.0 -> 6.4.8
- jax 0.4.7 -> 0.4.8
- pandas 1.4.4 -> 1.5.3
- spacy 3.5.1 -> 3.5.2
- SQLAlchemy 1.4.47 -> 2.0.9
- xgboost 1.7.4 -> 1.7.5

2023-03-31

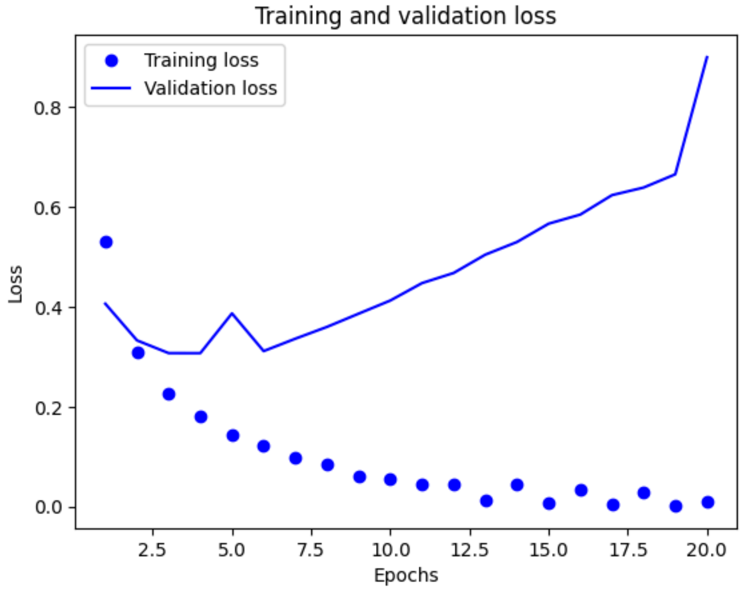
- Improve bash ! syntax highlighting (GitHub issue)
- Fix bug where VIM keybindings weren't working in the file editor
- Upgraded R from 4.2.2 to 4.2.3
- Python package updates:
  - arviz 0.12.1 --> 0.15.1
  - astropy 4.3.1 --> 5.2.2
  - dopamine-rl 1.0.5 --> 4.0.6
  - gensim 3.6.0 --> 4.3.1
  - ipykernel 5.3.4 -> 5.5.6
  - ipython 7.9.0 -> 7.34.0
  - jax 0.4.4 -> 0.4.7
  - jaxlib 0.4.4 -> 0.4.7
  - jupyter_core 5.2.0 -> 5.3.0
  - keras 2.11.0 -> 2.12.0
  - lightgbm 2.2.3 -> 3.3.5
  - matplotlib 3.5.3 -> 3.7.1
  - nltk 3.7 -> 3.8.1
  - opencv-python 4.6.0.66 -> 4.7.0.72
  - plotly 5.5.0 -> 5.13.1
  - pymc 4.1.4 -> 5.1.2
  - seaborn 0.11.2 -> 0.12.2
  - spacy 3.4.4 -> 3.5.1
  - sympy 1.7.1 -> 1.11.1
  - tensorboard 2.11.2 -> 2.12.0
  - tensorflow 2.11.0 -> 2.12.0
  - tensorflow-estimator 2.11.0 -> 2.12.0
  - tensorflow-hub 0.12.0 -> 0.13.0
  - torch 1.13.1 -> 2.0.0
  - torchaudio 0.13.1 -> 2.0.1
  - torchtext 0.14.1 -> 0.15.1
  - torchvision 0.14.1 -> 0.15.1

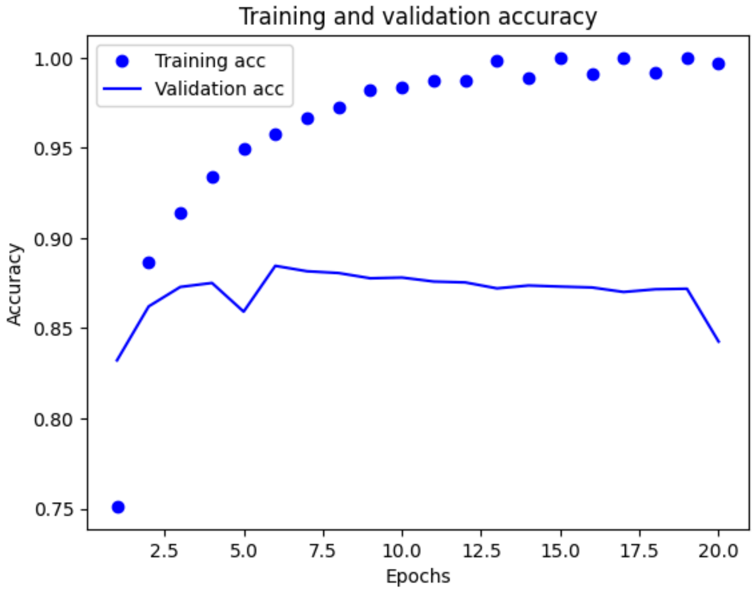
2023-03-10

- Added the Colab editor shortcuts example notebook
- Fixed triggering of @-mention and email autocomplete for large comments (GitHub issue)
- Added View Resources to the Runtime menu
- Made file viewer images fit the view by default, resizing to original size on click
- When in VIM mode, enable copy as well as allowing propagation to monaco-vim to escape visual mode (GitHub issue)
- Upgraded CUDA 11.6.2 -> 11.8.0 and cuDNN 8.4.0.27 -> 8.7.0.84
- Upgraded Nvidia drivers 525.78.01 -> 530.30.02
- Upgraded Python 3.8.10 -> 3.9.16
- Python package updates:
  - beautifulsoup4 4.6.3 -> 4.9.3
  - bokeh 2.3.3 -> 2.4.3
  - debugpy 1.0.0 -> 1.6.6
  - Flask 1.1.4 -> 2.2.3

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
loss_value_32 = history_dict_32["loss"]
val_loss_value_32 = history_dict_32["val_loss"]
epochs_32 = range(1, len(loss_value_32) + 1)
plt.plot(epochs_32, loss_value_32, "bo", label="Training loss")
plt.plot(epochs_32, val_loss_value_32, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()


plt.clf()
accuracy_32 = history_dict_32["accuracy"]
val_accuracy_32 = history_dict_32["val_accuracy"]
plt.plot(epochs_32, accuracy_32, "bo", label="Training acc")
plt.plot(epochs_32, val_accuracy_32, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```





  - jax 0.3.25 -> 0.4.4
  - jaxlib 0.3.25 -> 0.4.4
  - Jinja2 2.11.3 -> 3.1.2
  - matplotlib 3.2.2 -> 3.5.3
  - nbconvert 5.6.1 -> 6.5.4
  - pandas 1.3.5 -> 1.4.4
  - pandas-datareader 0.9.0 -> 0.10.0
  - pandas-profiling 1.4.1 -> 3.2.0
  - Pillow 7.1.2 -> 8.4.0
  - plotnine 0.8.0 -> 0.10.1
  - scikit-image 0.18.3 -> 0.19.3
  - scikit-learn 1.0.2 -> 1.2.2
  - scipy 1.7.3 -> 1.10.1
  - setuptools 57.4.0 -> 63.4.3
  - sklearn-pandas 1.8.0 -> 2.2.0
  - statsmodels 0.12.2 -> 0.13.5
  - urllib3 1.24.3 -> 1.26.14
  - Werkzeug 1.0.1 -> 2.2.3
  - wrapt 1.14.1 -> 1.15.0
  - xgboost 0.90 -> 1.7.4
  - xlrd 1.2.0 -> 2.0.1

2023-02-17

- Show graphs of RAM and disk usage in notebook toolbar
- Copy cell links directly to the clipboard instead of showing a dialog when clicking on the link icon in the cell toolbar
- Updated the Colab Marketplace VM image
- Upgraded CUDA to 11.6.2 and cuDNN to 8.4.0.27
- Python package updates:
  - tensorflow 2.9.2 -> 2.11.0
  - tensorboard 2.9.1 -> 2.11.2
  - keras 2.9.0 -> 2.11.0
  - tensorflow-estimator 2.9.0 -> 2.11.0
  - tensorflow-probability 0.17.0 -> 0.19.0
  - tensorflow-gcs-config 2.9.0 -> 2.11.0
  - earthengine-api 0.1.339 -> 0.1.341
  - flatbuffers 1.12 -> 23.1.21
  - platformdirs 2.6.2 -> 3.0.0
  - pydata-google-auth 1.6.0 -> 1.7.0
  - python-utils 3.4.5 -> 3.5.2
  - tenacity 8.1.0 -> 8.2.1
  - tifffile 2023.1.23.1 -> 2023.2.3
  - notebook 5.7.16 -> 6.3.0
  - tornado 6.0.4 -> 6.2
  - aiohttp 3.8.3 -> 3.8.4
  - charset-normalizer 2.1.1 -> 3.0.1
  - fastai 2.7.0 -> 2.7.1
  - soundfile 0.11.0 -> 0.12.1
  - typing-extensions 4.4.0 -> 4.5.0
  - widgetsnbextension 3.6.1 -> 3.6.2
  - pydantic 1.10.4 -> 1.10.5
  - zipp 3.12.0 -> 3.13.0
  - numpy 1.21.6 -> 1.22.4
  - drivefs 66.0 -> 69.0
  - gdal 3.0.4 -> 3.3.2 GitHub issue
- Added libudunits2-dev for smoother R package installs GitHub issue

2023-02-03

- Improved tooltips for pandas series to show common

```
history_32_units = model_32_units.fit(x_train, y_train, epochs=3, batch_size=512)
results_32_units = model_32_units.evaluate(x_test, y_test)
results_32_units
```

```
Epoch 1/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.9473 - loss: 0.2463
Epoch 2/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 3s 32ms/step - accuracy: 0.9639 - loss: 0.1133
Epoch 3/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 3s 42ms/step - accuracy: 0.9771 - loss: 0.0760
782/782 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8652 - loss: 0.4043
[0.4011080861091614, 0.8665599822998047]
```

```
model_32_units.predict(x_test)
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
array([[0.02573252],
       [0.99998534],
       [0.5480723 ],
       ...,
       [0.00979932],
       [0.02333886],
       [0.68163997]], dtype=float32)
```

```
model_64_units = keras.Sequential([
    layers.Dense(64, activation="relu"),
    layers.Dense(64, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model_64_units.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
# validation
x_val_64 = x_train[:10000]
partial_x_train = x_train[10000:]

y_val_64 = y_train[:10000]
partial_y_train = y_train[10000:]

history_64 = model_64_units.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val_64, y_val_64))
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 5s 121ms/step - accuracy: 0.6512 - loss: 0.6017 - val_accuracy: 0.8685 - v
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 53ms/step - accuracy: 0.8874 - loss: 0.3157 - val_accuracy: 0.8884 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 55ms/step - accuracy: 0.9193 - loss: 0.2267 - val_accuracy: 0.8650 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 53ms/step - accuracy: 0.9382 - loss: 0.1771 - val_accuracy: 0.8455 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 76ms/step - accuracy: 0.9398 - loss: 0.1637 - val_accuracy: 0.8801 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 79ms/step - accuracy: 0.9568 - loss: 0.1256 - val_accuracy: 0.8778 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 54ms/step - accuracy: 0.9679 - loss: 0.0998 - val_accuracy: 0.8829 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 54ms/step - accuracy: 0.9718 - loss: 0.0860 - val_accuracy: 0.8332 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 54ms/step - accuracy: 0.9683 - loss: 0.0869 - val_accuracy: 0.8808 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 53ms/step - accuracy: 0.9873 - loss: 0.0471 - val_accuracy: 0.8799 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 56ms/step - accuracy: 0.9907 - loss: 0.0384 - val_accuracy: 0.8801 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 94ms/step - accuracy: 0.9880 - loss: 0.0403 - val_accuracy: 0.8782 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 68ms/step - accuracy: 0.9956 - loss: 0.0230 - val_accuracy: 0.8772 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 55ms/step - accuracy: 0.9994 - loss: 0.0117 - val_accuracy: 0.8770 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 52ms/step - accuracy: 0.9876 - loss: 0.0367 - val_accuracy: 0.8756 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 53ms/step - accuracy: 0.9999 - loss: 0.0063 - val_accuracy: 0.8757 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 55ms/step - accuracy: 0.9946 - loss: 0.0197 - val_accuracy: 0.8751 - va
Epoch 18/20
```

statistics about the series object

- Made the forms dropdown behave like an autocomplete box when it allows input
- Updated the nvidia driver from 460.32.03 to 510.47.03
- Python package updates:
    - absl-py 1.3.0 -> 1.4.0
    - bleach 5.0.1 -> 6.0.0
    - cachetools 5.2.1 -> 5.3.0
    - cmdstanpy 1.0.8 -> 1.1.0
    - dnspython 2.2.1 -> 2.3.0
    - fsspec 2022.11.0 -> 2023.1.0
    - google-cloud-bigquery-storage 2.17.0 -> 2.18.1
    - holidays 0.18 -> 0.19
    - jupyter-core 5.1.3 -> 5.2.0
    - packaging 21.3 -> 23.0
    - prometheus-client 0.15.0 -> 0.16.0
    - pyct 0.4.8 -> 0.5.0
    - pydata-google-auth 1.5.0 -> 1.6.0
    - python-slugify 7.0.0 -> 8.0.0
    - sqlalchemy 1.4.46 -> 2.0.0
    - tensorflow-io-gcs-filesystem 0.29.0 -> 0.30.0
    - tifffile 2022.10.10 -> 2023.1.23.1
    - zipp 3.11.0 -> 3.12.0
    - Pinned sqlalchemy to version 1.4.46

2023-01-12

- Added support for @-mention and email autocomplete in comments
- Improved errors when GitHub notebooks can't be loaded
- Increased color contrast for colors used for syntax highlighting in the code editor
- Added terminal access for custom GCE VM runtimes
- Upgraded Ubuntu from 18.04 LTS to 20.04 LTS (GitHub issue)
- Python package updates:
    - GDAL 2.2.2 -> 2.2.3.
    - NumPy from 1.21.5 to 1.21.6.
    - attrs 22.1.0 -> 22.2.0
    - chardet 3.0.4 -> 4.0.0
    - cloudpickle 1.6.0 -> 2.2.0
    - filelock 3.8.2 -> 3.9.0
    - google-api-core 2.8.2 -> 2.11.0
    - google-api-python-client 1.12.11 -> 2.70.0
    - google-auth-httplib2 0.0.3 -> 0.1.0
    - google-cloud-bigquery 3.3.5 -> 3.4.1
    - google-cloud-datastore 2.9.0 -> 2.11.0
    - google-cloud-firestore 2.7.2 -> 2.7.3
    - google-cloud-storage 2.5.0 -> 2.7.0
    - holidays 0.17.2 -> holidays 0.18
    - importlib-metadata 5.2.0 -> 6.0.0
    - networkx 2.8.8 -> 3.0
    - opencv-python-headless 4.6.0.66 -> 4.7.0.68
    - pip 21.1.3 -> 22.04
    - pip-tools 6.2.0 -> 6.6.2
    - prettytable 3.5.0 -> 3.6.0
    - requests 2.23.0 -> 2.25.1
    - termcolor 2.1.1 -> 2.2.0
    - torch 1.13.0 -> 1.13.1

```
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 76ms/step - accuracy: 0.9999 - loss: 0.0039 - val_accuracy: 0.8747 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 84ms/step - accuracy: 0.9999 - loss: 0.0027 - val_accuracy: 0.8221 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 53ms/step - accuracy: 0.9713 - loss: 0.0878 - val_accuracy: 0.8737 - va
```

```
history_dict_64 = history_64.history
history_dict_64.keys()
```

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
loss_value64 = history_dict_64["loss"]
val_loss_value64 = history_dict_64["val_loss"]
epochs_64 = range(1, len(loss_value64) + 1)
plt.plot(epochs_64, loss_value64, "bo", label="Training loss")
plt.plot(epochs_64, val_loss_value64, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()

plt.clf()
accuracy_64 = history_dict_64["accuracy"]
val_accuracy_64 = history_dict_64["val_accuracy"]
plt.plot(epochs_64, accuracy_64, "bo", label="Training acc")
plt.plot(epochs_64, val_accuracy_64, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```

  - torchaudio 0.13.0 -> 0.13.1
  - torchtext 0.14.0-> 0.14.1
  - torchvision 0.14.0 -> 0.14.1

2022-12-06

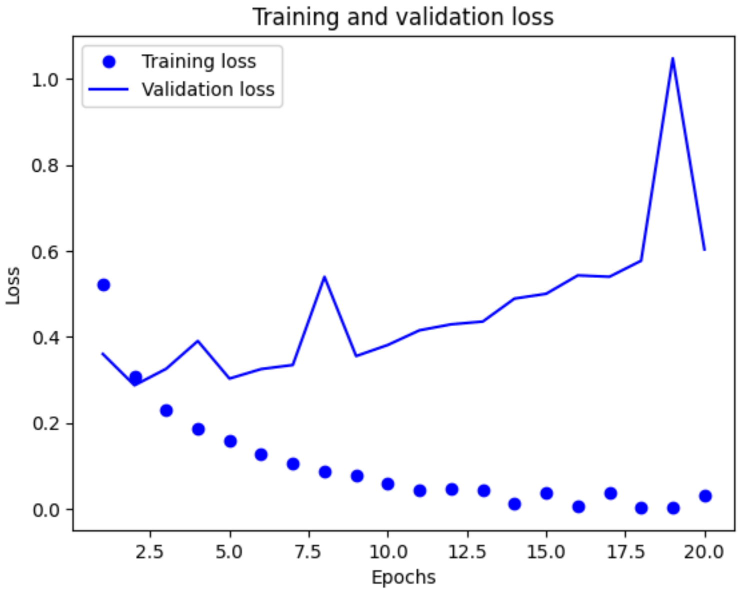
- Made fallback runtime version available until mid-December (GitHub issue)
- Upgraded to Python 3.8 (GitHub issue)
- Python package updates:
  - jax from 0.3.23 to 0.3.25, jaxlib from 0.3.22 to 0.3.25
  - pyarrow from 6.0.1 to 9.0.0
  - torch from 1.12.1 to 1.13.0
  - torchaudio from 0.12.1 to 0.13.0
  - torchvision from 0.13.1 to 0.14.0
  - torchtext from 0.13.1 to 0.14.0
  - xlrd from 1.1.0 to 1.2.0
  - DriveFS from 62.0.1 to 66.0.3
- Made styling of markdown tables in outputs match markdown tables in text cells
- Improved formatting for empty interactive table rows
- Fixed syntax highlighting for variables with names that contain Python keywords (GitHub issue)

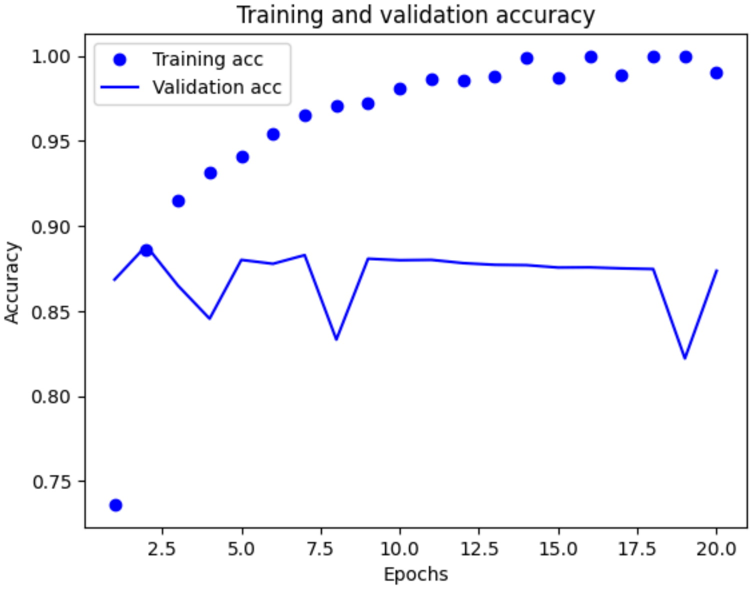
2022-11-11

- Added more dark editor themes for Monaco (when in dark mode, "Editor colorization" appears as an option in the Editor tab of the Tools → Settings dialog)
- Fixed bug where collapsed forms were deleted on mobile GitHub issue
- Python package updates:
  - rpy2 from 3.4.0 to 3.5.5 (GitHub issue)
  - notebook from 5.5.0 to 5.7.16
  - tornado from 5.1.1 to 6.0.4
  - tensorflow_probability from 0.16.0 to 0.17.0
  - pandas-gbq from 0.13.3 to 0.17.9
  - protobuf from 3.17.3 to 3.19.6
  - google-api-core[grpc] from 1.31.5 to 2.8.2
  - google-cloud-bigquery from 1.21.0 to 3.3.5
  - google-cloud-core from 1.0.1 to 2.3.2
  - google-cloud-datastore from 1.8.0 to 2.9.0
  - google-cloud-firestore from 1.7.0 to 2.7.2
  - google-cloud-language from 1.2.0 to 2.6.1
  - google-cloud-storage from 1.18.0 to 2.5.0
  - google-cloud-translate from 1.5.0 to 3.8.4

2022-10-21

- Launched a single-click way to get from BigQuery to Colab to





```
history_64 = model_64_units.fit(x_train, y_train, epochs=3, batch_size=512)
results_64_units = model_64_units.evaluate(x_test, y_test)
results_64_units
```

```
Epoch 1/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 41ms/step - accuracy: 0.9454 - loss: 0.2108
Epoch 2/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 40ms/step - accuracy: 0.9698 - loss: 0.0991
Epoch 3/3
49/49 ━━━━━━━━━━━━━━━━━━━━ 4s 62ms/step - accuracy: 0.9820 - loss: 0.0620
782/782 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.8641 - loss: 0.4049
[0.39651817083358765, 0.8678399920463562]
```

```
model_64_units.predict(x_test)
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step
array([[0.0480094 ],
       [0.9999997 ],
       [0.4658258 ],
       ...,
       [0.05174194],
       [0.06017439],
       [0.86283445]], dtype=float32)
```

```
model_128units = keras.Sequential([
    layers.Dense(128, activation="relu"),
    layers.Dense(128, activation="relu"),
```

further explore query results (announcement)
- Launched Pro, Pro+, and Pay As You Go to 19 additional countries: Austria, Belgium, Bulgaria, Croatia, Cyprus, Czechia, Denmark, Estonia, Finland, Greece, Hungary, Latvia, Lithuania, Norway, Portugal, Romania, Slovakia, Slovenia, and Sweden (tweet)
- Updated jax from 0.3.17 to 0.3.23, jaxlib from 0.3.15 to 0.3.22, TensorFlow from 2.8.2 to 2.9.2, CUDA from 11.1 to 11.2, and cuDNN from 8.0 to 8.1 (backend-info)
- Added a `readonly` option to `drive.mount`
- Fixed bug where Xarray was not working (GitHub issue)
- Modified Markdown parsing to ignore block quote symbol within MathJax (GitHub issue)

2022-09-30

- Launched Pay As You Go, allowing premium GPU access without requiring a subscription
- Added vim and tcllib to our runtime image
- Fixed bug where open files were closed on kernel disconnect (GitHub issue)
- Fixed bug where the play button/execution indicator was not clickable when scrolled into the cell output (GitHub issue)
- Updated the styling for form titles so that they avoid obscuring the code editor
- Created a GitHub repo, backend-info, with the latest apt-list.txt and pip-freeze.txt files for the Colab runtime (GitHub issue)
- Added `files.upload_file(filename)` to upload a file from the browser to the runtime with a specified filename

2022-09-16

- Upgraded pymc from 3.11.0 to 4.1.4, jax from 0.3.14 to 0.3.17, jaxlib from 0.3.14 to 0.3.15, fsspec from 2022.8.1 to 2022.8.2
- Modified our save flow to avoid persisting Drive filenames as titles in notebook JSON
- Updated our Terms of Service
- Modified the `Jump to Cell` command to locate the cursor at the end of the command palette input (`Jump to cell` in Tools → Command palette in a notebook with section headings)
- Updated the styling of the Drive notebook comment UI
- Added support for terminating your runtime from code: `python from google.colab import runtime runtime.unassign()`
- Added regex filter support to the Recent notebooks dialog
- Inline google.colab.files.upload JS to fix `files.upload()` not working (GitHub issue)

2022-08-26

- Upgraded PyYAML from 3.13 to 6.0 (GitHub issue), drivefs from 61.0.3 to 62.0.1

```
    layers.Dense(128, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model_128units.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
# validation
x_val_128 = x_train[:10000]
partial_x_train = x_train[10000:]

y_val_128 = y_train[:10000]
partial_y_train = y_train[10000:]

history_128 = model_128units.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val_128, y_val_128))
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 6s 152ms/step - accuracy: 0.6314 - loss: 0.6301 - val_accuracy: 0.8621 - v
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 85ms/step - accuracy: 0.8768 - loss: 0.3200 - val_accuracy: 0.8870 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 88ms/step - accuracy: 0.9171 - loss: 0.2188 - val_accuracy: 0.8842 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 87ms/step - accuracy: 0.9251 - loss: 0.1939 - val_accuracy: 0.8860 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 5s 86ms/step - accuracy: 0.9400 - loss: 0.1559 - val_accuracy: 0.8872 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 85ms/step - accuracy: 0.9753 - loss: 0.0813 - val_accuracy: 0.8845 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 87ms/step - accuracy: 0.9856 - loss: 0.0530 - val_accuracy: 0.8845 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 6s 121ms/step - accuracy: 0.9898 - loss: 0.0415 - val_accuracy: 0.8822 - v
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 86ms/step - accuracy: 0.9983 - loss: 0.0134 - val_accuracy: 0.8810 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 5s 86ms/step - accuracy: 0.9869 - loss: 0.0612 - val_accuracy: 0.8807 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 117ms/step - accuracy: 0.9997 - loss: 0.0041 - val_accuracy: 0.8832 - v
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 104ms/step - accuracy: 0.9999 - loss: 0.0015 - val_accuracy: 0.8818 - v
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 5s 86ms/step - accuracy: 0.9675 - loss: 0.1886 - val_accuracy: 0.8804 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 6s 119ms/step - accuracy: 0.9999 - loss: 0.0015 - val_accuracy: 0.8820 - v
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 98ms/step - accuracy: 1.0000 - loss: 5.9014e-04 - val_accuracy: 0.8831
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 85ms/step - accuracy: 1.0000 - loss: 2.7505e-04 - val_accuracy: 0.8793
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 5s 86ms/step - accuracy: 1.0000 - loss: 1.6025e-04 - val_accuracy: 0.8805
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 5s 87ms/step - accuracy: 1.0000 - loss: 1.1837e-04 - val_accuracy: 0.8823
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 5s 86ms/step - accuracy: 1.0000 - loss: 7.5542e-05 - val_accuracy: 0.8821
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 6s 120ms/step - accuracy: 1.0000 - loss: 5.9381e-05 - val_accuracy: 0.8827
```

```
history_dict_128 = history_128.history
history_dict_128.keys()
```

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
loss_value128 = history_dict_128["loss"]
val_loss_value128 = history_dict_128["val_loss"]
epochs_128 = range(1, len(loss_value128) + 1)
plt.plot(epochs_128, loss_value128, "bo", label="Training loss")
plt.plot(epochs_128, val_loss_value128, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()
```

```
plt.clf()
```

- Upgraded TensorFlow from 2.8.2 to 2.9.1 and ipywidgets from 7.7.1 to 8.0.1 but rolled both back due to a number of user reports (GitHub issue, GitHub issue)
- Stop persisting inferred titles in notebook JSON (GitHub issue)
- Fix bug in background execution which affected some Pro+ users (GitHub issue)
- Fix bug where `Download as .py` incorrectly handled text cells ending in a double quote
- Fix bug for Pro and Pro+ users where we weren't honoring the preference (Tools → Settings) to use a temporary scratch notebook as the default landing page
- Provide undo/redo for scratch cells
- When writing ipynb files, serialize empty multiline strings as [] for better consistency with JupyterLab

2022-08-11

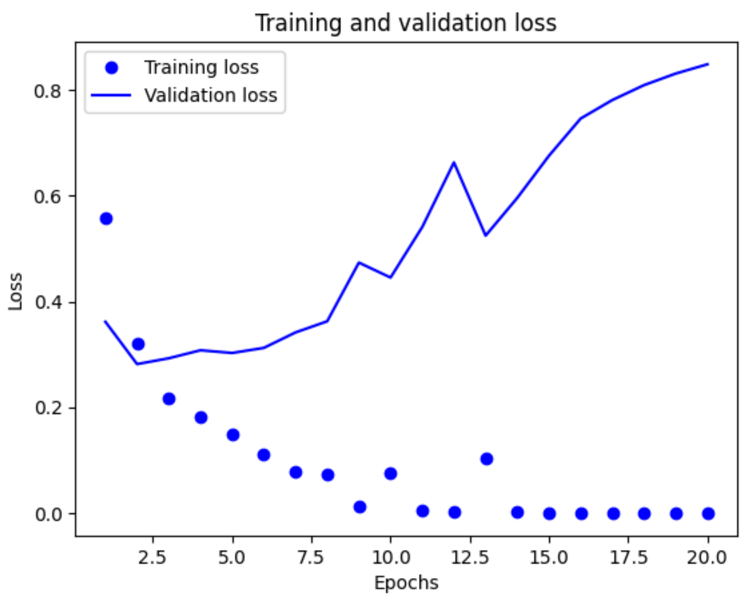
- Upgraded ipython from 5.5.0 to 7.9.0, fbprophet 0.7 to prophet 1.1, tensorflow-datasets from 4.0.1 to 4.6.0, drivefs from 60.0.2 to 61.0.3, pytorch from 1.12.0 to 1.12.1, numba from 0.51 to 0.56, and lxml from 4.2.0 to 4.9.1
- Loosened our `requests` version requirement (GitHub issue)
- Removed support for TensorFlow 1
- Added Help → Report Drive abuse for Drive notebooks
- Fixed indentation for Python lines ending in [
- Modified styling of tables in Markdown to left-align them rather than centering them
- Fixed special character replacement when copying interactive tables as Markdown
- Fixed ansi 8-bit color parsing (GitHub issue)
- Configured logging to preempt transitive imports and other loading from implicitly configuring the root logger
- Modified forms to use a value of `None` instead of causing a parse error when clearing raw and numeric-typed form fields

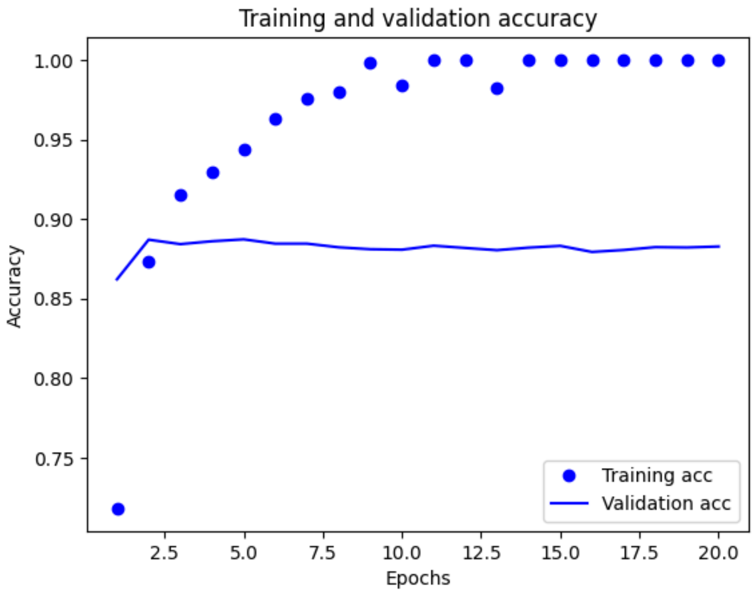
2022-07-22

- Update scipy from 1.4.1 to 1.7.3, drivefs from 59.0.3 to 60.0.2, pytorch from 1.11 to 1.12, jax & jaxlib from 0.3.8 to 0.3.14, opencv-python from 4.1.2.30 to 4.6.0.66, spaCy from 3.3.1 to 3.4.0, and dlib from 19.18.0 to 19.24.0
- Fix `Open in tab` doc link which was rendering incorrectly (GitHub issue)
- Add a preference for the default tab orientation to the Site section of the settings menu under Tools → Settings
- Show a warning for `USE_AUTH_EPHEM` usage when running authenticate_user on a TPU runtime (code)

2022-07-01

```
accuracy_128 = history_dict_128["accuracy"]
val_accuracy_128 = history_dict_128["val_accuracy"]
plt.plot(epochs_128, accuracy_128, "bo", label="Training acc")
plt.plot(epochs_128, val_accuracy_128, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```





```
history_128 = model_128units.fit(x_train, y_train, epochs=2, batch_size=512)
results_128_units = model_128units.evaluate(x_test, y_test)
results_128_units
```

```
Epoch 1/2
49/49 ━━━━━━━━━━━━━━━━━━━━ 3s 65ms/step - accuracy: 0.9303 - loss: 0.2993
Epoch 2/2
49/49 ━━━━━━━━━━━━━━━━━━━━ 6s 92ms/step - accuracy: 0.9704 - loss: 0.0899
782/782 ━━━━━━━━━━━━━━━━━━━━ 3s 3ms/step - accuracy: 0.8564 - loss: 0.4396
[0.4302583634853363, 0.8579999804496765]
```

```
model_128units.predict(x_test)
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 3s 4ms/step
array([[0.03385768],
       [1.        ],
       [0.9768075 ],
       ...,
```

- Add a preference for code font to the settings menu under Tools → Settings
- Update drivefs from 58.0.3 to 59.0.3 and spacy from 2.2.4 to 3.3.1
- Allow display_data and execute_result text outputs to wrap, matching behavior of JupyterLab (does not affect stream outputs/print statements).
- Improve LSP handling of some magics, esp. %%writefile (GitHub issue).
- Add a FAQ entry about the mount Drive button behavior and include link buttons for each FAQ entry.
- Fix bug where the notebook was sometimes hidden behind other tabs on load when in single pane view.
- Fix issue with inconsistent scrolling when an editor is in multi-select mode.
- Fix bug where clicking on a link in a form would navigate away from the notebook
- Show a confirmation dialog before performing Replace all from the Find and replace pane.

2022-06-10

- Update drivefs from 57.0.5 to 58.0.3 and tensorflow from 2.8.0 to 2.8.2
- Support more than 100 repos in the GitHub repo selector shown in the open dialog and the clone to GitHub dialog
- Show full notebook names on hover in the open dialog
- Improve the color contrast for links, buttons, and the `ipywidgets.Accordion` widget in dark mode

2022-05-20

- Support URL params for linking to some common pref settings: force_theme=dark, force_corgi_mode=1, force_font_size=14. Params forced by URL are not persisted unless saved using Tools → Settings.
- Add a class `markdown-google-sans` to allow Markdown to render in Google Sans
- Update monaco-vim from 0.1.19 to 0.3.4
- Update drivefs from 55.0.3 to 57.0.5, jax from 0.3.4 to 0.3.8, and jaxlib from 0.3.2 to 0.3.7

2022-04-29

- Added 🦀 mode (under Miscellaneous in Tools → Settings)
- Added "Disconnect and delete runtime" option to the menu next to the Connect button
- Improved rendering of filter options in an interactive table
- Added git-lfs to the base image
- Updated torch from 1.10.0 to 1.11.0, jupyter-core from 4.9.2 to 4.10.0, and cmake from 3.12.0 to 3.22.3
- Added more details to our FAQ about unsupported uses (using

```
           [0.31125492],
           [0.06801374],
           [0.9265763 ]], dtype=float32)
```

```
MSE_model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
# compilation of model
MSE_model.compile(optimizer="rmsprop",
              loss="mse",
              metrics=["accuracy"])
# validation of model
x_val_MSE = x_train[:10000]
partial_x_train = x_train[10000:]

y_val_MSE = y_train[:10000]
partial_y_train = y_train[10000:]
# Model Fit

history_MSE = MSE_model.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val_MSE, y_val_MSE))
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 55ms/step - accuracy: 0.6759 - loss: 0.2200 - val_accuracy: 0.8057 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 31ms/step - accuracy: 0.8765 - loss: 0.1171 - val_accuracy: 0.8809 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9147 - loss: 0.0780 - val_accuracy: 0.8856 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9344 - loss: 0.0592 - val_accuracy: 0.8874 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - accuracy: 0.9470 - loss: 0.0496 - val_accuracy: 0.8853 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 46ms/step - accuracy: 0.9560 - loss: 0.0408 - val_accuracy: 0.8832 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 32ms/step - accuracy: 0.9667 - loss: 0.0343 - val_accuracy: 0.8817 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9684 - loss: 0.0323 - val_accuracy: 0.8675 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9669 - loss: 0.0314 - val_accuracy: 0.8726 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.9679 - loss: 0.0298 - val_accuracy: 0.8462 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9780 - loss: 0.0222 - val_accuracy: 0.8768 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9874 - loss: 0.0157 - val_accuracy: 0.8703 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9871 - loss: 0.0162 - val_accuracy: 0.8762 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9894 - loss: 0.0124 - val_accuracy: 0.8756 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 45ms/step - accuracy: 0.9894 - loss: 0.0124 - val_accuracy: 0.8747 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 46ms/step - accuracy: 0.9911 - loss: 0.0106 - val_accuracy: 0.8763 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.9905 - loss: 0.0110 - val_accuracy: 0.8744 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9928 - loss: 0.0084 - val_accuracy: 0.8671 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9792 - loss: 0.0172 - val_accuracy: 0.8728 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9908 - loss: 0.0095 - val_accuracy: 0.8744 - va
```

```
historydict_MSE = history_MSE.history
historydict_MSE.keys()
```

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
import matplotlib.pyplot as plt
loss_value_MSE = historydict_MSE["loss"]
val_loss_value_MSE = historydict_MSE["val_loss"]
epochs_MSE = range(1, len(loss_value_MSE) + 1)
```

proxies, downloading torrents, etc.)
- Fixed issue with apt-get dependencies

2022-04-15

- Add an option in the file browser to show hidden files.
- Upgrade gdown from 4.2.0 to 4.4.0, google-api-core[grpc] from 1.26.0 to 1.31.5, and pytz from 2018.4 to 2022.1

2022-03-25

- Launched Pro/Pro+ to 12 additional countries: Australia, Bangladesh, Colombia, Hong Kong, Indonesia, Mexico, New Zealand, Pakistan, Philippines, Singapore, Taiwan, and Vietnam
- Added `google.colab.auth.authentica` to support using Service Account keys
- Update jax from 0.3.1 to 0.3.4 & jaxlib from 0.3.0 to 0.3.2
- Fixed an issue with Twitter previews of notebooks shared as GitHub Gists

2022-03-10

- Launched Pro/Pro+ to 10 new countries: Ireland, Israel, Italy, Morocco, the Netherlands, Poland, Spain, Switzerland, Turkey, and the United Arab Emirates
- Launched support for scheduling notebooks for Pro+ users
- Fixed bug in interactive datatables where filtering by number did not work
- Finished removing the python2 kernelspec

2022-02-25

- Made various accessibility improvements to the header
- Fix bug with forms run:auto where a form field change would trigger multiple runs
- Minor updates to the bigquery example notebook and snippet
- Include background execution setting in the sessions dialog for Pro+ users
- Update tensorflow-probability from 0.15 to 0.16
- Update jax from 0.2.25 to 0.3.1 & jaxlib from 0.1.71 to 0.3.0

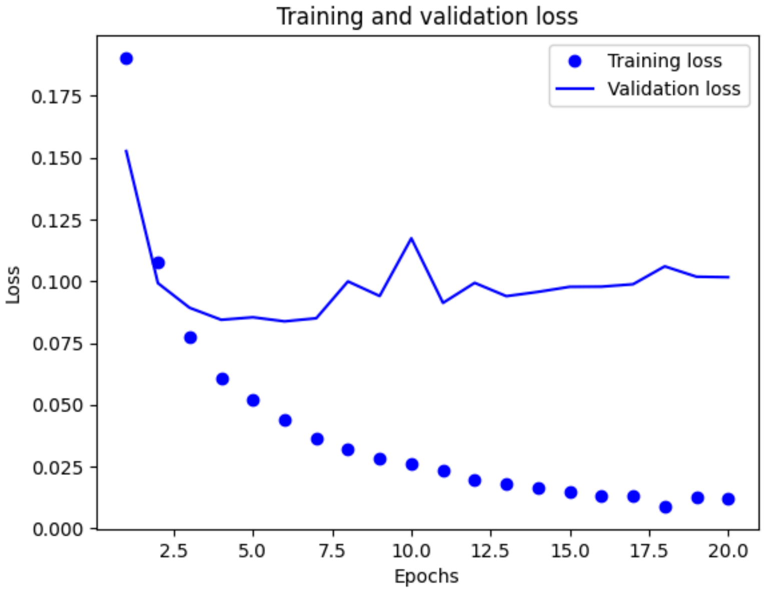
2022-02-11

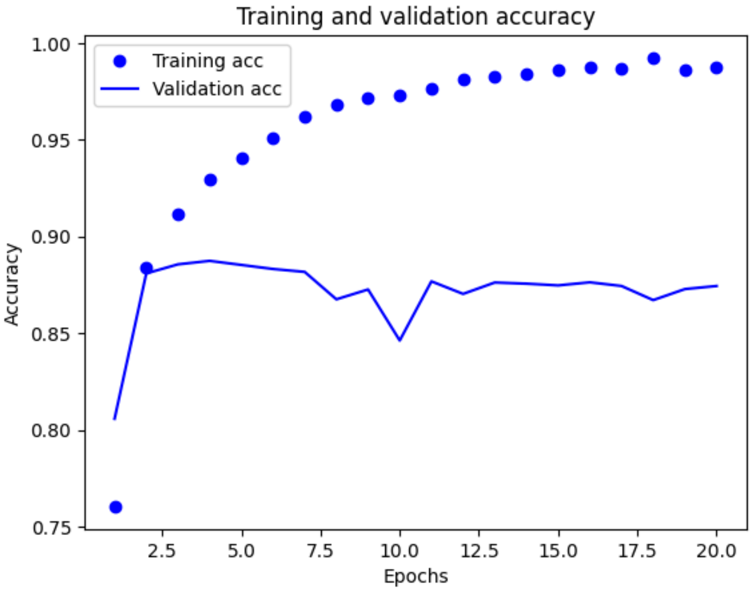
- Improve keyboard navigation for the open dialog
- Fix issue where nvidia-smi stopped reporting resource utilization for some users who were modifying the version of nvidia used
- Update tensorflow from 2.7 to 2.8, keras from 2.7 to 2.8, numpy from 1.19.5 to 1.21.5, tables from 3.4.4 to 3.7.0

2022-02-04

- Improve UX for opening content alongside your notebook, such as files opened from the file browser. This includes a multi-pane view and drag-drop support

```
plt.plot(epochs_MSE, loss_value_MSE, "bo", label="Training loss")
plt.plot(epochs_MSE, val_loss_value_MSE, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()


plt.clf()
acc_MSE = historydict_MSE["accuracy"]
val_acc_MSE = historydict_MSE["val_accuracy"]
plt.plot(epochs_MSE, acc_MSE, "bo", label="Training acc")
plt.plot(epochs_MSE, val_acc_MSE, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```





```
MSE_model.fit(x_train, y_train, epochs=8, batch_size=512)
results_MSE = MSE_model.evaluate(x_test, y_test)
results_MSE
```

```
Epoch 1/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9465 - loss: 0.0454
Epoch 2/8
```

- Better Twitter previews when sharing example Colab notebooks and notebooks opened from GitHub Gists
- Update pandas from 1.1.5 to 1.3.5
- Update openpyxl from 2.5.9 to 3.0.0 and pyarrow from 3.0.0 to 6.0.0
- Link to the release notes from the Help menu

2022-01-28

- Add a copy button to data tables
- Python LSP support for better completions and code diagnostics. This can be configured in the Editor Settings (Tools → Settings)
- Update gspread examples in our documentation
- Update gdown from 3.6 to 4.2

2022-01-21

- New documentation for the `google.colab` package
- Show GPU RAM in the resource usage tab
- Improved security for mounting Google Drive which disallows mounting Drive from accounts other than the one currently executing the notebook

2022-01-14

- Add a preference (Tools → Settings) to use a temporary scratch notebook as the default landing page
- Fix bug where / and : weren't working in VIM mode
- Update gspread from 3.0 to 3.4
- Update the Colab Marketplace VM image

```
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9592 - loss: 0.0360
Epoch 3/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.9649 - loss: 0.0322
Epoch 4/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 21ms/step - accuracy: 0.9716 - loss: 0.0271
Epoch 5/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9712 - loss: 0.0264
Epoch 6/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9737 - loss: 0.0247
Epoch 7/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9809 - loss: 0.0188
Epoch 8/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9782 - loss: 0.0208
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8545 - loss: 0.1243
[0.12150118499994278, 0.856440007686615]
```

```
MSE_model.predict(x_test)
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step
array([[0.01586813],
       [0.9999909 ],
       [0.999445  ],
       ...,
       [0.75893724],
       [0.00893678],
       [0.9225745 ]], dtype=float32)
```

```
tanh = keras.Sequential([
    layers.Dense(16, activation="tanh"),
    layers.Dense(1, activation="sigmoid")
])

tanh.compile(optimizer='rmsprop',
              loss='mse',
              metrics=['accuracy'])

x_val_tanh = x_train[:10000]
partial_x_train = x_train[10000:]

y_val_tanh = y_train[:10000]
partial_y_train = y_train[10000:]


historytanh_model = tanh.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val_tanh, y_val_tanh))
```

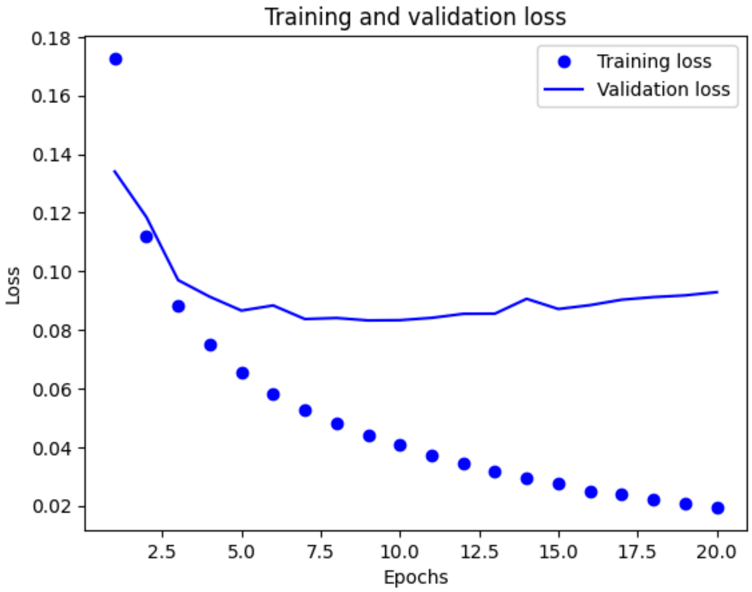
```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 55ms/step - accuracy: 0.7295 - loss: 0.1981 - val_accuracy: 0.8550 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 31ms/step - accuracy: 0.8861 - loss: 0.1164 - val_accuracy: 0.8531 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9048 - loss: 0.0912 - val_accuracy: 0.8843 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9178 - loss: 0.0775 - val_accuracy: 0.8838 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9287 - loss: 0.0671 - val_accuracy: 0.8894 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 34ms/step - accuracy: 0.9425 - loss: 0.0562 - val_accuracy: 0.8801 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 48ms/step - accuracy: 0.9440 - loss: 0.0535 - val_accuracy: 0.8881 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 54ms/step - accuracy: 0.9513 - loss: 0.0483 - val_accuracy: 0.8833 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 29ms/step - accuracy: 0.9578 - loss: 0.0437 - val_accuracy: 0.8861 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9612 - loss: 0.0396 - val_accuracy: 0.8847 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9650 - loss: 0.0372 - val_accuracy: 0.8839 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9709 - loss: 0.0330 - val_accuracy: 0.8810 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9743 - loss: 0.0296 - val_accuracy: 0.8820 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9738 - loss: 0.0289 - val_accuracy: 0.8738 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.9785 - loss: 0.0264 - val_accuracy: 0.8803 - va
Epoch 16/20
```

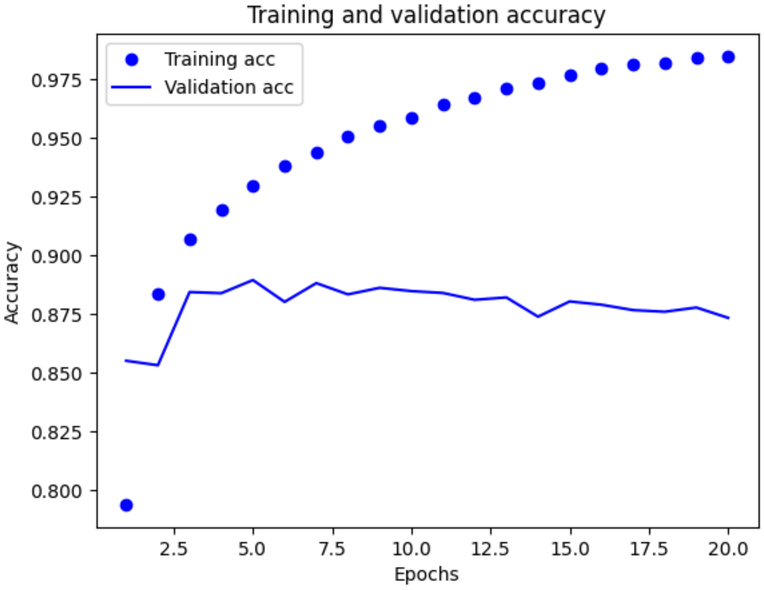
```
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9778 - loss: 0.0255 - val_accuracy: 0.8789 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 53ms/step - accuracy: 0.9834 - loss: 0.0224 - val_accuracy: 0.8766 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 47ms/step - accuracy: 0.9833 - loss: 0.0215 - val_accuracy: 0.8759 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 29ms/step - accuracy: 0.9864 - loss: 0.0193 - val_accuracy: 0.8777 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9855 - loss: 0.0188 - val_accuracy: 0.8733 - va
```

```python
historydict_tanh = historytanh_model.history
historydict_tanh.keys()
```

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```python
loss_value_tanh= historydict_tanh["loss"]
val_loss_value_tanh = historydict_tanh["val_loss"]
epochs_tanh = range(1, len(loss_value_tanh) + 1)
plt.plot(epochs_tanh, loss_value_tanh, "bo", label="Training loss")
plt.plot(epochs_tanh, val_loss_value_tanh, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()

plt.clf()
acc_tanh = historydict_tanh["accuracy"]
val_acc_tanh = historydict_tanh["val_accuracy"]
plt.plot(epochs_tanh, acc_tanh, "bo", label="Training acc")
plt.plot(epochs_tanh, val_acc_tanh, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```





```
tanh.fit(x_train, y_train, epochs=8, batch_size=512)
results_tanh = tanh.evaluate(x_test, y_test)
results_tanh
```

```
Epoch 1/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9456 - loss: 0.0460
Epoch 2/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9523 - loss: 0.0412
Epoch 3/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9628 - loss: 0.0349
Epoch 4/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9646 - loss: 0.0341
Epoch 5/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 35ms/step - accuracy: 0.9679 - loss: 0.0309
Epoch 6/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 22ms/step - accuracy: 0.9722 - loss: 0.0290
Epoch 7/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9729 - loss: 0.0277
Epoch 8/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9759 - loss: 0.0247
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8631 - loss: 0.1085
[0.1068594753742218, 0.8656399846076965]
```

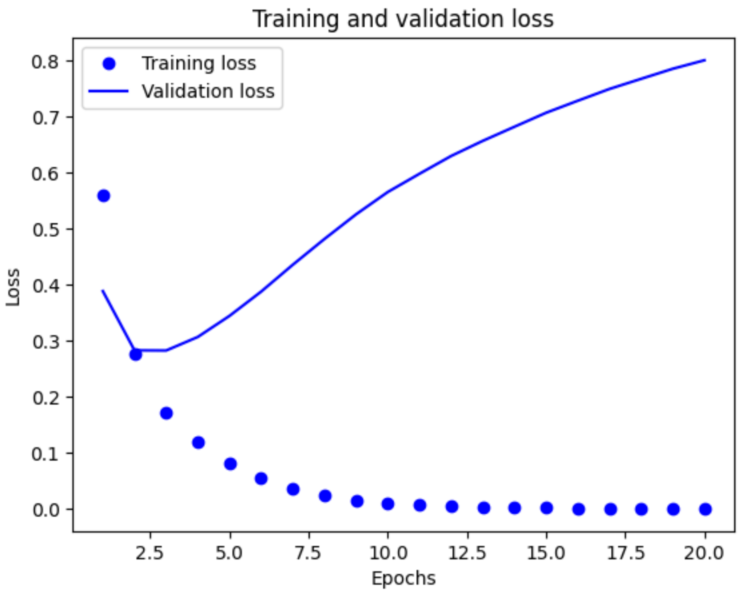
```
adam = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
     layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])

adam.compile(optimizer='adam',
             loss='binary_crossentropy',
             metrics=['accuracy'])

x_adam = x_train[:10000]
partial_x_train = x_train[10000:]

y_adam = y_train[:10000]
partial_y_train = y_train[10000:]


historyadam = adam.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_adam, y_adam))
```

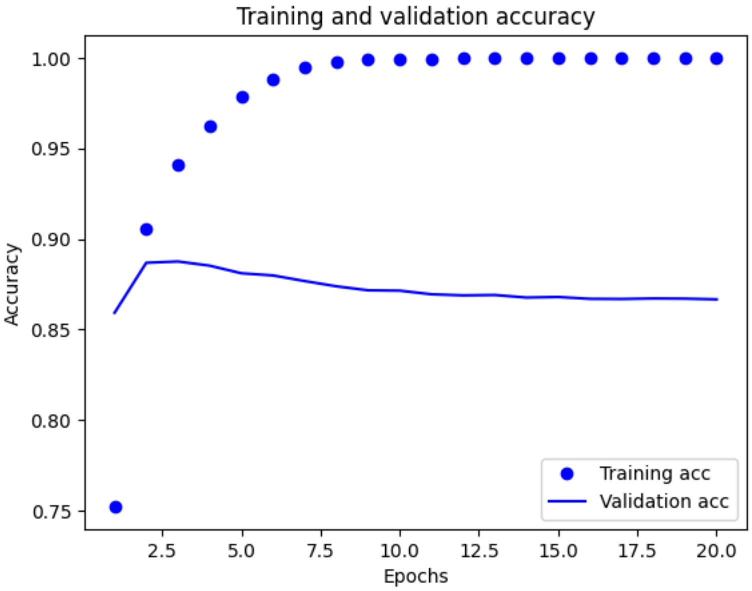
```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 4s 68ms/step - accuracy: 0.6596 - loss: 0.6326 - val_accuracy: 0.8593 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 47ms/step - accuracy: 0.9004 - loss: 0.3065 - val_accuracy: 0.8869 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 31ms/step - accuracy: 0.9411 - loss: 0.1784 - val_accuracy: 0.8876 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9620 - loss: 0.1229 - val_accuracy: 0.8853 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9794 - loss: 0.0831 - val_accuracy: 0.8811 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9880 - loss: 0.0552 - val_accuracy: 0.8799 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9943 - loss: 0.0376 - val_accuracy: 0.8768 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9979 - loss: 0.0230 - val_accuracy: 0.8739 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9993 - loss: 0.0146 - val_accuracy: 0.8717 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9993 - loss: 0.0109 - val_accuracy: 0.8715 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 45ms/step - accuracy: 0.9995 - loss: 0.0077 - val_accuracy: 0.8695 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 45ms/step - accuracy: 0.9999 - loss: 0.0048 - val_accuracy: 0.8689 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 30ms/step - accuracy: 0.9998 - loss: 0.0040 - val_accuracy: 0.8691 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9998 - loss: 0.0032 - val_accuracy: 0.8677 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 1.0000 - loss: 0.0023 - val_accuracy: 0.8680 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 1.0000 - loss: 0.0018 - val_accuracy: 0.8670 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 1.0000 - loss: 0.0015 - val_accuracy: 0.8669 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 1.0000 - loss: 0.0013 - val_accuracy: 0.8672 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 1.0000 - loss: 0.0011 - val_accuracy: 0.8671 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 1.0000 - loss: 9.2875e-04 - val_accuracy: 0.8667
```

```
historydict_adam = historyadam.history
historydict_adam.keys()
```

```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
loss_value_adam = historydict_adam["loss"]
val_loss_value_adam = historydict_adam["val_loss"]
epochs_adam = range(1, len(loss_value_adam) + 1)
plt.plot(epochs_adam, loss_value_adam, "bo", label="Training loss")
plt.plot(epochs_adam, val_loss_value_adam, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()


plt.clf()
acc_adam = historydict_adam["accuracy"]
acc_adam = historydict_adam["accuracy"]
val_acc_adam = historydict_adam["val_accuracy"]
plt.plot(epochs_adam, acc_adam, "bo", label="Training acc")
plt.plot(epochs_adam, val_acc_adam, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```





```
adam.fit(x_train, y_train, epochs=4, batch_size=512)
results_adam = adam.evaluate(x_test, y_test)
results_adam
```

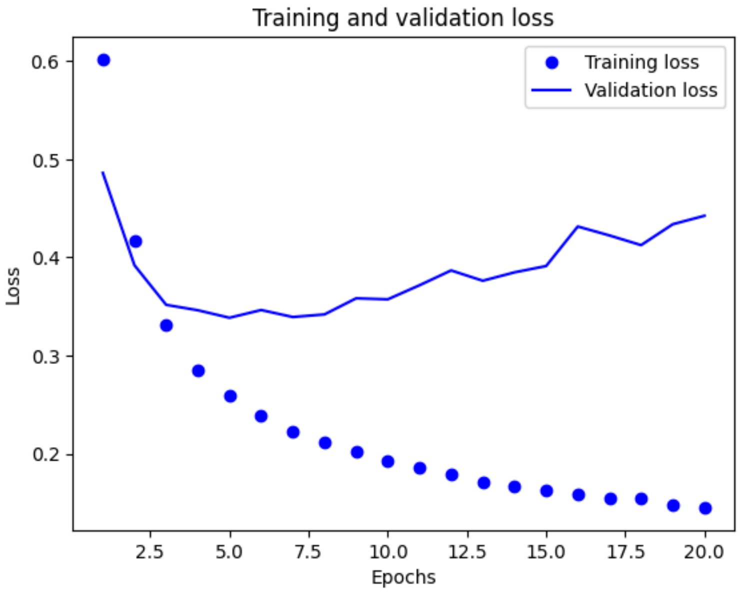
```
Epoch 1/4
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.9360 - loss: 0.2792
Epoch 2/4
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 22ms/step - accuracy: 0.9637 - loss: 0.1176
Epoch 3/4
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9808 - loss: 0.0672
Epoch 4/4
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9918 - loss: 0.0393
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8575 - loss: 0.5922
[0.5853615403175354, 0.8577200174331665]
```

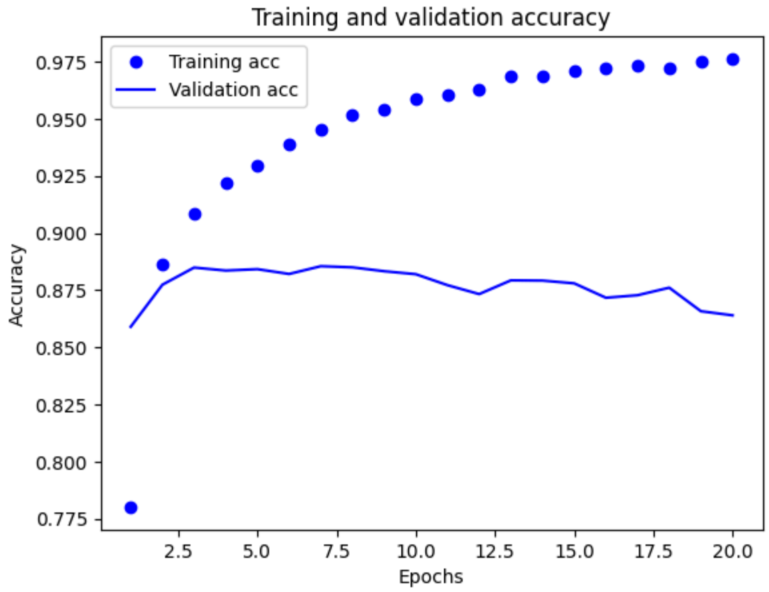
```
#Regularization
```

```
from tensorflow.keras import regularizers
regularization = keras.Sequential([
    layers.Dense(16, activation="relu",kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(16, activation="relu",kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(1, activation="sigmoid")
])
regularization.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])

history_regularization = regularization.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
historydict_regularization = history_regularization.history
historydict_regularization.keys()
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 4s 73ms/step - accuracy: 0.6956 - loss: 0.6662 - val_accuracy: 0.8590 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.8877 - loss: 0.4330 - val_accuracy: 0.8774 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9120 - loss: 0.3358 - val_accuracy: 0.8849 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.9218 - loss: 0.2886 - val_accuracy: 0.8836 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9302 - loss: 0.2598 - val_accuracy: 0.8842 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9407 - loss: 0.2376 - val_accuracy: 0.8821 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9476 - loss: 0.2190 - val_accuracy: 0.8855 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9535 - loss: 0.2113 - val_accuracy: 0.8850 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9582 - loss: 0.1943 - val_accuracy: 0.8833 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 46ms/step - accuracy: 0.9624 - loss: 0.1865 - val_accuracy: 0.8820 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 44ms/step - accuracy: 0.9690 - loss: 0.1747 - val_accuracy: 0.8772 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 32ms/step - accuracy: 0.9703 - loss: 0.1703 - val_accuracy: 0.8733 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9721 - loss: 0.1659 - val_accuracy: 0.8793 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9740 - loss: 0.1590 - val_accuracy: 0.8792 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9728 - loss: 0.1595 - val_accuracy: 0.8780 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9781 - loss: 0.1504 - val_accuracy: 0.8717 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 31ms/step - accuracy: 0.9740 - loss: 0.1523 - val_accuracy: 0.8728 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9771 - loss: 0.1470 - val_accuracy: 0.8761 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9819 - loss: 0.1389 - val_accuracy: 0.8658 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 41ms/step - accuracy: 0.9822 - loss: 0.1349 - val_accuracy: 0.8640 - va
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
loss_valu = historydict_regularization["loss"]
val_loss_value_r = historydict_regularization["val_loss"]
epochs_r = range(1, len(loss_valu) + 1)
plt.plot(epochs_r, loss_valu, "bo", label="Training loss")
plt.plot(epochs_r, val_loss_value_r, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()

plt.clf()
acc_r = historydict_regularization["accuracy"]
val_acc_r = historydict_regularization["val_accuracy"]
plt.plot(epochs_r, acc_r, "bo", label="Training acc")
plt.plot(epochs_r, val_acc_r, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```





```
regularization.fit(x_train, y_train, epochs=8, batch_size=512)
results_regularization = regularization.evaluate(x_test, y_test)
results_regularization
```

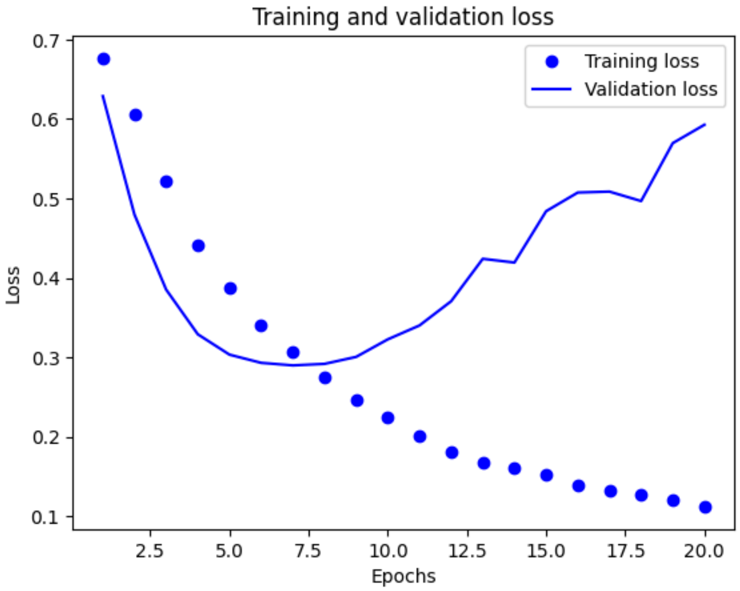
```
Epoch 1/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9350 - loss: 0.2588
Epoch 2/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9523 - loss: 0.2021
Epoch 3/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9542 - loss: 0.1945
Epoch 4/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9557 - loss: 0.1879
Epoch 5/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9541 - loss: 0.1888
Epoch 6/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9627 - loss: 0.1724
Epoch 7/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9633 - loss: 0.1728
Epoch 8/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 24ms/step - accuracy: 0.9616 - loss: 0.1719
782/782 ━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.8648 - loss: 0.4348
[0.42694592475891113, 0.8697999715805054]
```

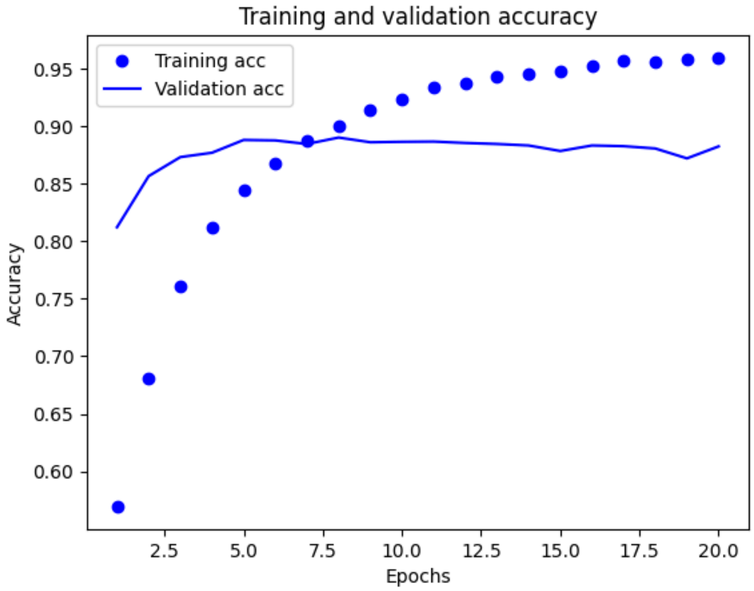
```
#Dropout
from tensorflow.keras import regularizers
Dropout = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dropout(0.5),
    layers.Dense(16, activation="relu"),
    layers.Dropout(0.5),
    layers.Dense(16, activation="relu"),
    layers.Dropout(0.5),
    layers.Dense(1, activation="sigmoid")
])
Dropout.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])

history_Dropout = Dropout.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
historydict_Dropout = history_Dropout.history
historydict_Dropout.keys()
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 3s 57ms/step - accuracy: 0.5385 - loss: 0.6875 - val_accuracy: 0.8120 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 30ms/step - accuracy: 0.6632 - loss: 0.6217 - val_accuracy: 0.8564 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.7480 - loss: 0.5409 - val_accuracy: 0.8729 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 34ms/step - accuracy: 0.8013 - loss: 0.4502 - val_accuracy: 0.8767 - va
Epoch 5/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 58ms/step - accuracy: 0.8409 - loss: 0.3942 - val_accuracy: 0.8878 - va
Epoch 6/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 31ms/step - accuracy: 0.8663 - loss: 0.3470 - val_accuracy: 0.8874 - va
Epoch 7/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8844 - loss: 0.3114 - val_accuracy: 0.8843 - va
Epoch 8/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.8968 - loss: 0.2785 - val_accuracy: 0.8899 - va
Epoch 9/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9122 - loss: 0.2494 - val_accuracy: 0.8858 - va
Epoch 10/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - accuracy: 0.9224 - loss: 0.2260 - val_accuracy: 0.8862 - va
Epoch 11/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9313 - loss: 0.2004 - val_accuracy: 0.8864 - va
Epoch 12/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9381 - loss: 0.1828 - val_accuracy: 0.8852 - va
Epoch 13/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 35ms/step - accuracy: 0.9436 - loss: 0.1622 - val_accuracy: 0.8843 - va
Epoch 14/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 47ms/step - accuracy: 0.9476 - loss: 0.1593 - val_accuracy: 0.8830 - va
Epoch 15/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 2s 59ms/step - accuracy: 0.9487 - loss: 0.1430 - val_accuracy: 0.8782 - va
Epoch 16/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - accuracy: 0.9512 - loss: 0.1402 - val_accuracy: 0.8830 - va
Epoch 17/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - accuracy: 0.9583 - loss: 0.1301 - val_accuracy: 0.8824 - va
Epoch 18/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 33ms/step - accuracy: 0.9580 - loss: 0.1236 - val_accuracy: 0.8804 - va
Epoch 19/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9571 - loss: 0.1219 - val_accuracy: 0.8718 - va
Epoch 20/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 29ms/step - accuracy: 0.9565 - loss: 0.1156 - val_accuracy: 0.8822 - va
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

```
loss_val = historydict_Dropout["loss"]
val_loss_val_d = historydict_Dropout["val_loss"]
epochs_d = range(1, len(loss_val) + 1)
plt.plot(epochs_d, loss_val, "bo", label="Training loss")
plt.plot(epochs_d, val_loss_val_d, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()


plt.clf()
acc_d = historydict_Dropout["accuracy"]
```

```
val_acc_d = historydict_Dropout["val_accuracy"]
plt.plot(epochs_d, acc_d, "bo", label="Training acc")
plt.plot(epochs_d, val_acc_d, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```





```
Dropout.fit(x_train, y_train, epochs=8, batch_size=512)
results_Dropout = Dropout.evaluate(x_test, y_test)
results_Dropout
```

```
Epoch 1/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9040 - loss: 0.3190
Epoch 2/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9113 - loss: 0.2660
Epoch 3/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 33ms/step - accuracy: 0.9215 - loss: 0.2114
Epoch 4/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 2s 24ms/step - accuracy: 0.9283 - loss: 0.2006
Epoch 5/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 24ms/step - accuracy: 0.9326 - loss: 0.1867
Epoch 6/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9367 - loss: 0.1714
Epoch 7/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9377 - loss: 0.1729
Epoch 8/8
49/49 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9408 - loss: 0.1581
```

```
782/782 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8682 - loss: 0.5006
[0.4915551543235779, 0.8683199882507324]
```

```
#Training model with hyper tuned parameters

from tensorflow.keras import regularizers
Hyper = keras.Sequential([
    layers.Dense(32, activation="relu",kernel_regularizer=regularizers.l2(0.0001)),
    layers.Dropout(0.5),
    layers.Dense(32, activation="relu",kernel_regularizer=regularizers.l2(0.0001)),
    layers.Dropout(0.5),
    layers.Dense(16, activation="relu",kernel_regularizer=regularizers.l2(0.0001)),
    layers.Dropout(0.5),
    layers.Dense(1, activation="sigmoid")
])
Hyper.compile(optimizer="rmsprop",
              loss="mse",
              metrics=["accuracy"])

history_Hyper = Hyper.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
history_dictHyper = history_Hyper.history
history_dictHyper.keys()
```

```
Epoch 1/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 4s 65ms/step - accuracy: 0.5296 - loss: 0.2594 - val_accuracy: 0.8087 - va
Epoch 2/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 39ms/step - accuracy: 0.6787 - loss: 0.2234 - val_accuracy: 0.8501 - va
Epoch 3/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.7759 - loss: 0.1768 - val_accuracy: 0.8649 - va
Epoch 4/20
30/30 ━━━━━━━━━━━━━━━━━━━━ 1s 39ms/step - accuracy: 0.8345 - loss: 0.1429 - val_accuracy: 0.8815 - va
```