



Phishing

“Phishing” is where people are tricked into giving up secret information (like their passwords or credit card details) into a fake website that is disguised to look like a trustworthy website.

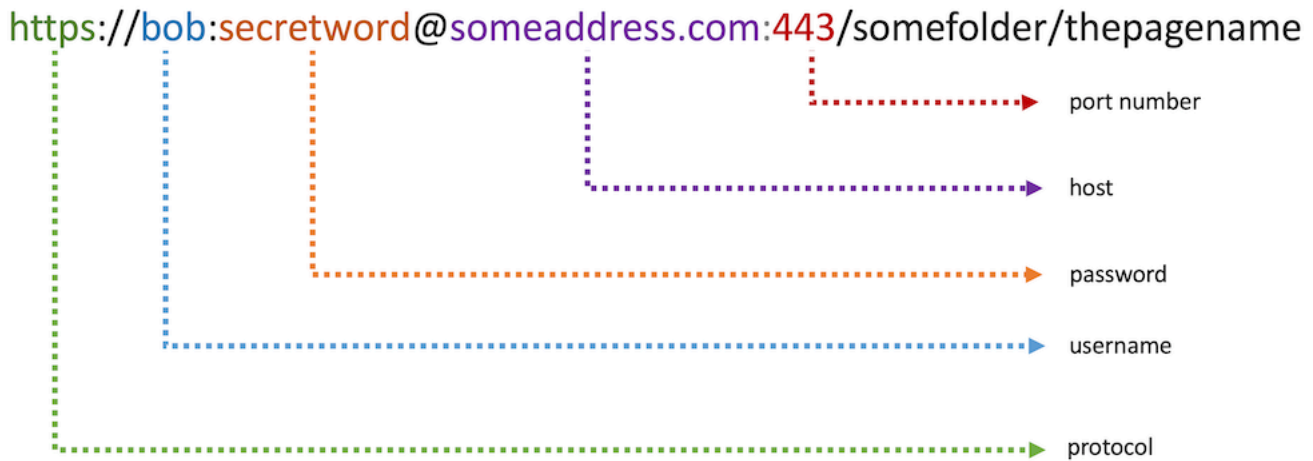
People are sent links to these fake phishing websites in emails or instant messages. How can they know if a link is safe to click on?

In this project, you will learn about the research that is being done to train machine learning systems to predict if a link is to a phishing website or a legitimate website.



This project worksheet is licensed under a Creative Commons Attribution Non-Commercial Share-Alike License
<http://creativecommons.org/licenses/by-nc-sa/4.0/>

Let's start by looking at some of the parts that make up a URL (web address):



port number	<p>Which port to connect to.</p> <p>If this isn't specified, the web browser will use 80 (if the protocol is "http") or 443 (if the protocol is "https").</p> <p>80 and 443 are the "standard" port numbers that are almost always used, but websites can use other numbers if they want.</p>
host	<p>Which web server to connect to.</p> <p>This can be a numeric address with dots in (called an "IP address") like 104.20.74.246</p> <p>Or it can be a text-based host name (called a "domain name") like <code>machinelearningforkids.co.uk</code> that the web browser will use to look up the IP address for (called a "DNS lookup").</p> <p>You can use <code>https://whois.net</code> to look up the IP address for a domain name.</p> <p>This will also let you find out when a domain name was registered, and how long it is before the domain name registration expires.</p>
password	<p>The password to use to identify the user</p> <p>If not specified, the web browser will not use a password.</p>
username	<p>The username to use to identify the user</p> <p>If not specified, the web browser will not use a username.</p>
protocol	<p>The communication protocol that the web browser should use.</p> <p>This will be "https" for secure connections, or "http" for insecure connections.</p>

Another thing to understand is "redirects". Sometimes a web address will send you to another web address. For example, if you visit `https://bit.ly/39XEEfP` you end up at `https://machinelearningforkids.co.uk` because the first address is a redirect.

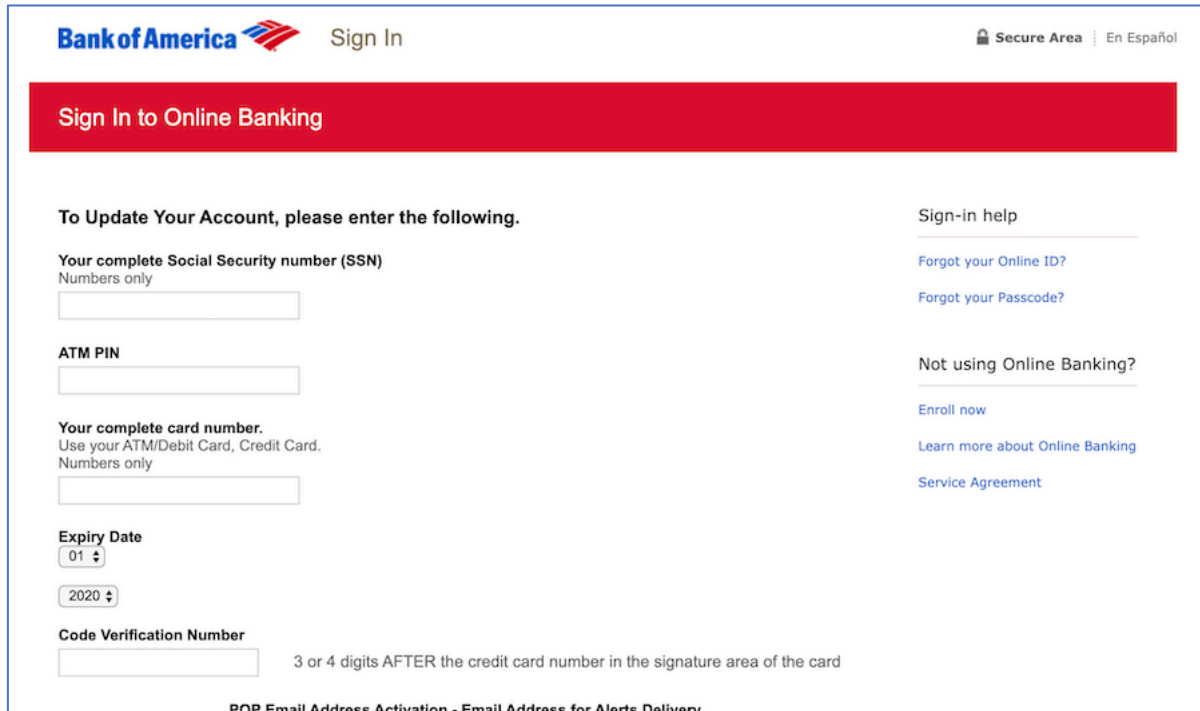
Redirects hide the real destination for a URL. You can join several together. For example, if you visit `https://bit.ly/35Jnlf8` you end up at `https://www.bbc.co.uk/news` after 4 redirects.

`https://bit.ly/35Jnlf8` ➡ `https://tinyurl.com/wrcfg53` ➡ `https://bbc.in/2FICKBL` ➡ `http://news.bbc.co.uk` ➡ `http://www.bbc.co.uk/news`

Now look at this URL:

<http://login.bankofamerica.com@www.josueizagirre.com/wp-content/cache/login.bankofamerica.com.uplogad/update.html>

The web page looks like this:



Bank of America Sign In Secure Area En Español

Sign In to Online Banking

To Update Your Account, please enter the following.

Your complete Social Security number (SSN)
Numbers only

ATM PIN

Your complete card number.
Use your ATM/Debit Card, Credit Card.
Numbers only

Expiry Date
01 2020

Code Verification Number
3 or 4 digits AFTER the credit card number in the signature area of the card

Sign-in help
[Forgot your Online ID?](#)
[Forgot your Passcode?](#)

Not using Online Banking?
[Enroll now](#)
[Learn more about Online Banking](#)
[Service Agreement](#)

POP Email Address Activation - Email Address for Alerts Delivery

This website has nothing to do with the real Bank of America.

This is a phishing URL.

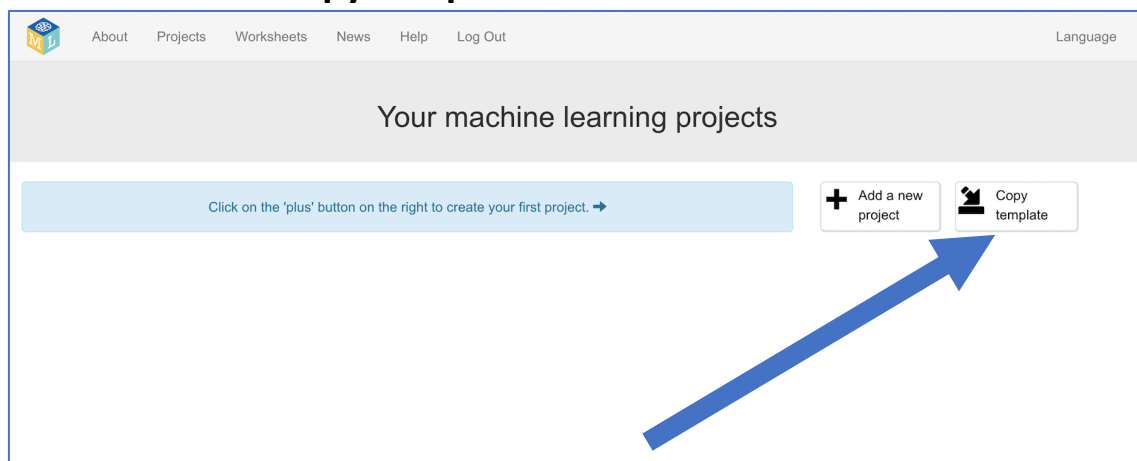
How could you tell?

port number	The page is using port 80 – a real banking website would be using the secure standard port 443.
host	The domain name for the website is www.josueizagirre.com which is not the real official domain for Bank of America.
username	The URL is giving the username “ login.bankofamerica.com ” although the web server ignores this. Putting a username like this here is a trick to make it look like the web address is “ login.bankofamerica.com ”.
protocol	The page is using “http” – a real banking website would be using the secure protocol “https”.

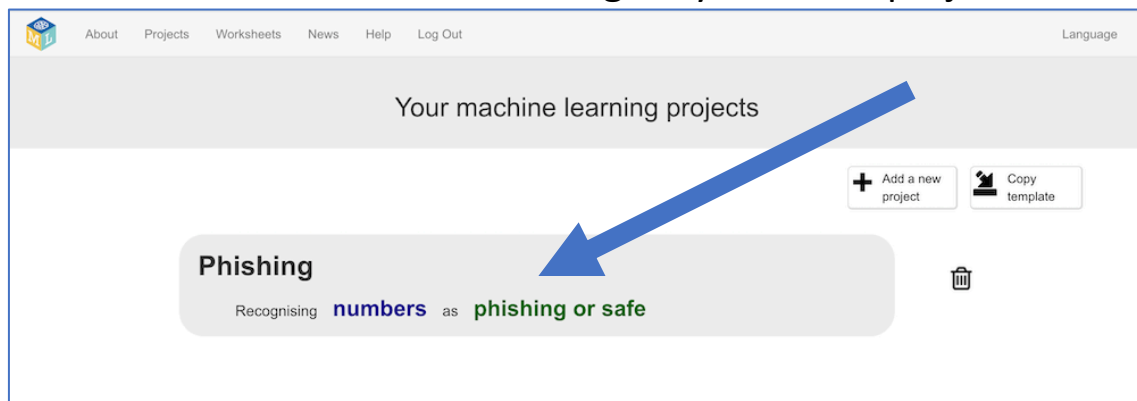
Do you think a computer could be trained to predict that this URL was for a phishing website?

1. Go to <https://machinelearningforkids.co.uk/> in a web browser
2. Click on **“Get started”**
3. Click on **“Log In”** and type in your username and password
If you don't have a username, ask your teacher or group leader to create one for you.
If you can't remember your username or password, ask your teacher or group leader to reset it for you.
4. Click on **“Projects”** on the top menu bar

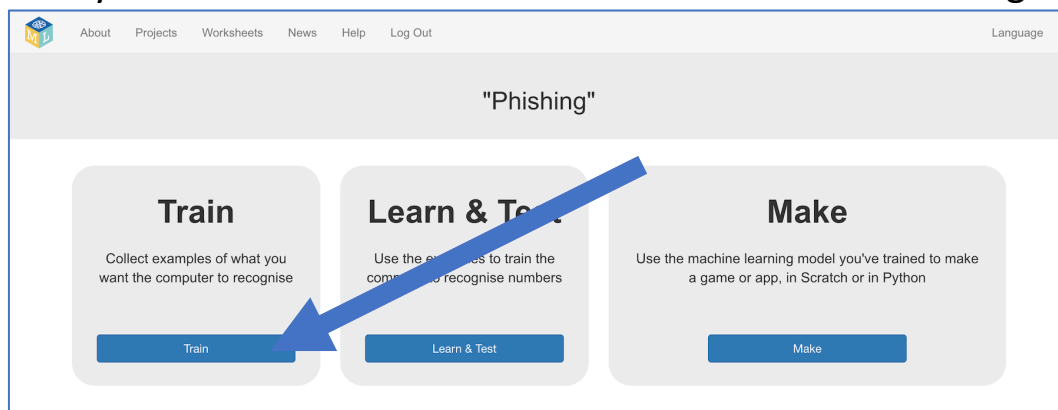
5. Click the **“Copy template”** button.



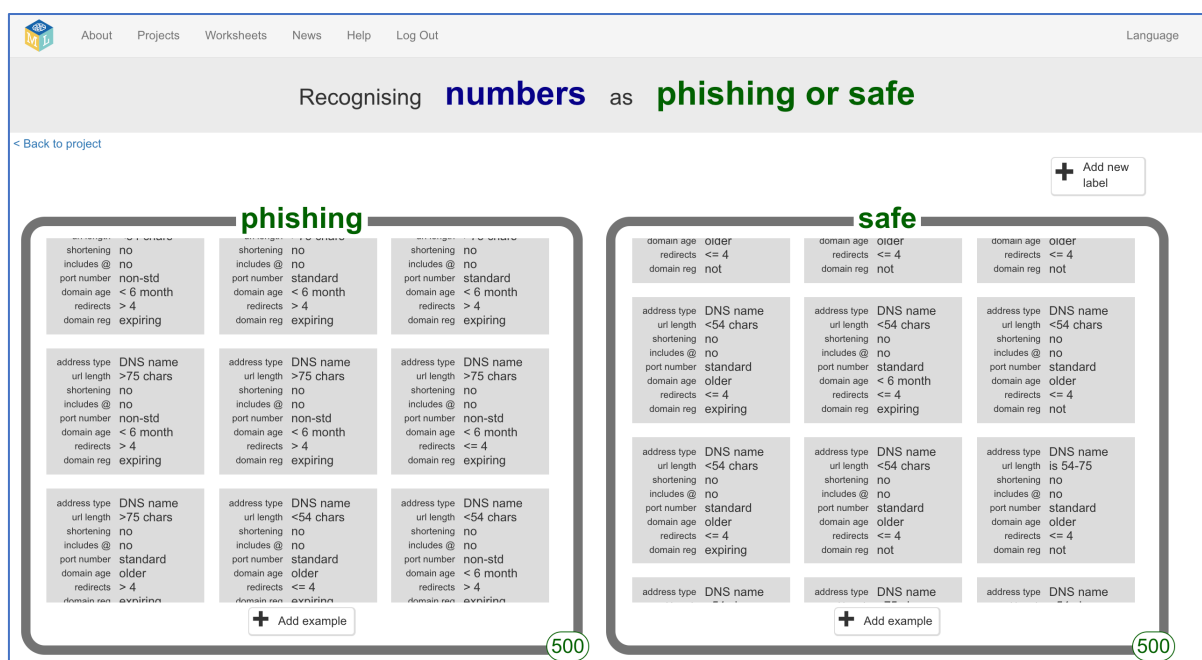
6. Click on the **“Import “Phishing””** button, and then click **“Import”**.
7. You should now see **“Phishing”** in your list of projects. Click on it.



8. This template project includes a sample set of training data to help save you time. Click the **Train** button to review the training data.



The sample training data includes details about 1000 URLs. 500 were identified as phishing websites, and 500 were identified as safe, legitimate websites.



Each URL has been described using the following attributes:

attribute	description	possible values	reason
address type	Has the host address in the URL been provided as a numeric IP address, or a text-based domain name?	DNS name if the URL had a text-based domain name (like "machinelearningforkids.co.uk") IP addr if the URL had a numeric host name (like "104.20.75.246")	Using IP addresses is unusual, and sometimes a way to hide the real destination for a web address.

attribute	description	possible values	reason
url length	How long is the URL?	<54 chars if the URL is shorter than 54 characters long is 54-75 if the URL is between 54 and 75 characters long >75 chars if the URL is longer than 75 characters long	Phishing URLs in emails tend to be longer, as it is easier to hide the suspicious part of the address if the URL is very long.
shortening	Does the URL use a shortening service, like bit.ly or tinyurl.com? Shortening services use redirects to make a short version of a long web address.	yes if the URL uses a recognized shortening service no if the URL does not use a recognized shortening service	Shortening services, like any redirects, make it hard to tell the true destination for a URL, so are useful to disguise a phishing link.
includes @	Does the URL include the @ symbol?	yes if the URL includes an @ no if the URL does not include @	Legitimate links with usernames/passwords are unusual. Phishing links take advantage of the fact that many users don't realize that anything before an @ is a username/password and put a pretend web address in there instead.
port number	Does the URL use the standard port numbers for web pages?	standard if the URL uses the standard port 80 or 443 non-std if the URL uses a custom, non-standard port number	Using non-standard port numbers is unusual.
domain age	How long ago was the domain name registered? For example, machinelearningforkids.co.uk was registered on 17 th April 2017, which you can check at whois.net	< 6 month if the domain name was registered less than 6 months ago older if the domain was registered longer than 6 months ago	Phishing websites are often not around for long. They are reported and discovered quickly, so attackers create new sites and register new domains. This means a link to a domain that was registered a long time ago may be more likely to be legitimate.
redirects	How many redirects does the web browser have to follow to reach the final page?	<= 4 if there are four or fewer redirects from the URL to the final webpage > 4 if there are more than four redirects to get to the final web page	One or two redirects is common to help web site developers manage their site, but more than that is possibly suspicious as a sign of trying to hide the true destination of a URL.

attribute	description	possible values	reason
domain reg	Has the domain been registered for many years, or is it due to expire soon?	expiring if the domain name is due to expire in the next year not if the domain name is not due to expire in the next year	Phishing websites are often not around for long. They are reported and discovered quickly, so attackers will only register domain names for a short time period to save their costs. Large reputable organizations are more likely to register domain names for several years.

For example:

<http://login.bankofamerica.com@www.josueizagirre.com/wp-content/cache/login.bankofamerica.com.uplogad/update.html> would be shown in the training data as:

```
address type  DNS name
url length   >75 chars
shortening   no
includes @    yes
port number   standard
domain age    older
redirects     <= 4
domain reg    expiring
```

Scroll through the rest of the training data that you've imported.

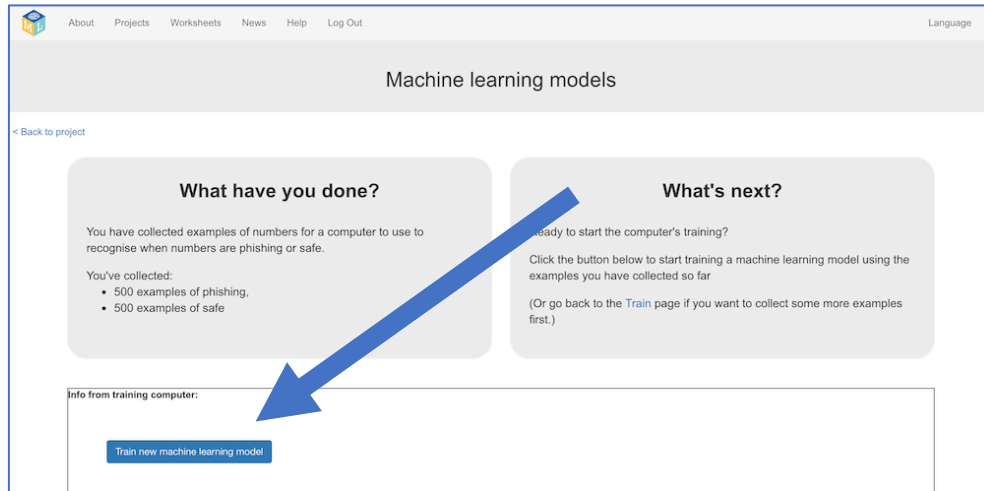
The phishing URLs from the sample project were reported by users around the world to the website <https://phishtank.com> as this is a reliable source for getting large numbers of phishing links.

The attributes chosen to describe the URLs are just a sample for your first starting project. After you've tried these, you'll look at some other attributes that can be used to train better machine learning models.

9. Click on the “< **Back to project**” link in the top-left.

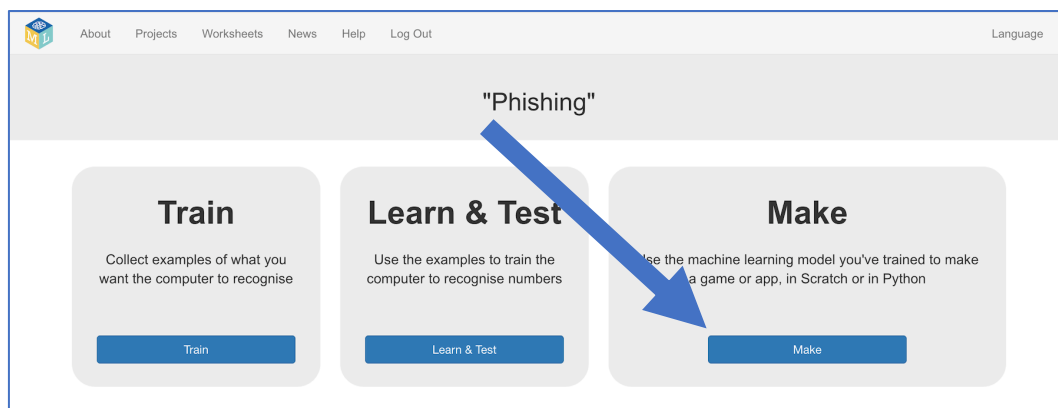
10. Click the “**Learn & Test**” button

11. Click the “Train new machine learning model” button

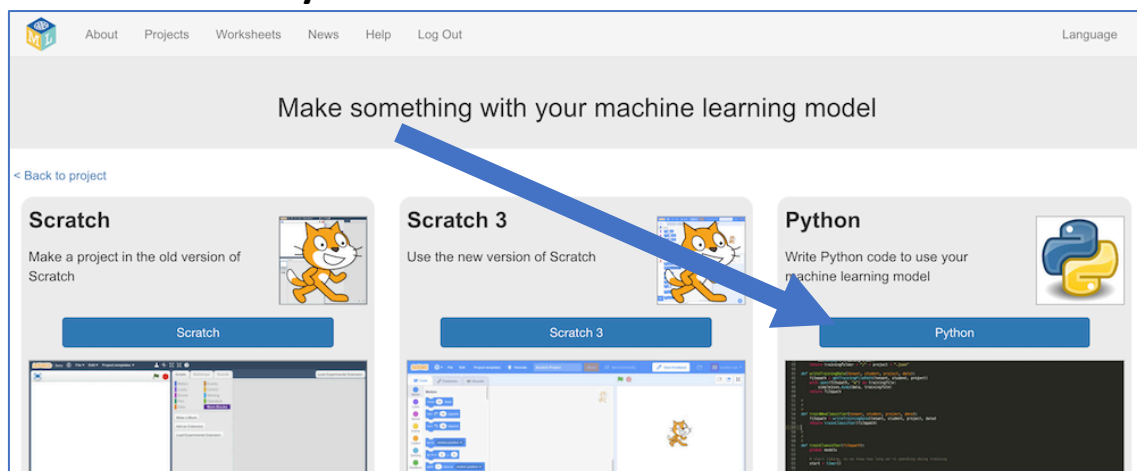


12. Click on the “< Back to project” link in the top-left.

13. Click the “Make” button



14. Click the “Python” button



15. Make a note of your “API key”

This is like a secret code that your Python program will be able to use to access your machine learning model.

The screenshot shows a web interface titled "Using machine learning in Python". It has a navigation bar with links: About, Projects, Worksheets, News, Help, Log Out, and a Language dropdown. Below the navigation bar, there's a "Back to project" link. The main content area is divided into three sections. The left section is a form for inputting data, with fields for address type (IP addr), url length (<54 chars), shortening (yes), includes @ (yes), port number (standard), domain age (< 6 month), redirects (<= 4), and domain reg (expiring). Below the form, it says "Running this code will print something like:" and shows a terminal output: "\$ python yourscrip.py" and "result: 'phishing' with 81% confidence". The middle section is a code editor with a Python script that uses the requests library to send a POST request to a machine learning model. The script includes a function to classify numbers and a demo function that uses the model to classify a list of data. The right section is a text input field for the API key, with a blue arrow pointing to it from the code editor. Below the input field, it says "Treat it like a password and make sure that you keep it secret!"

Note: The API key in the screenshot above is fake. API keys are like passwords. I wouldn't share my real API key, and you shouldn't share yours.

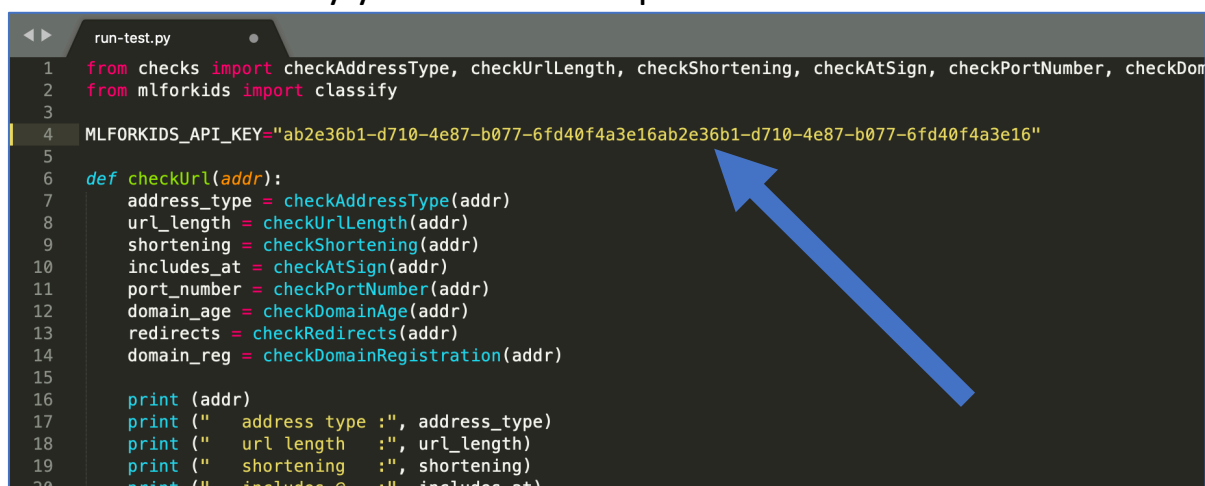
16. Go to <https://github.com/dalelane/mlforkids-phishing-sample> and download a zip with the sample Python project

The screenshot shows the GitHub repository page for "dalelane / mlforkids-phishing-sample". The repository has 1 commit, 1 branch, 0 packages, and 0 releases. The "Code" tab is selected, showing a list of files: .gitignore, README.md, checks.py, and constants.py. A blue arrow points to the "Clone or download" button. A modal is open, showing options to "Clone with SSH", "Use HTTPS", "Open in Desktop", and "Download ZIP".

17. Unzip the project zip file
18. Open a command prompt to the folder where you've unzipped your project code.
19. The **requirements.txt** file lists the third-party libraries that you will need to run the code.
- The simplest way to install all of these is to run the command:
- pip3 install -r requirements.txt**
- Ask your teacher or group leader to help if you're not sure how to do this.

```
D:\mlforkids-phishing-sample-master\dalelane$ pip3 install -r requirements.txt
Collecting certifi==2019.11.28 (from -r requirements.txt (line 1))
  Using cached https://files.pythonhosted.org/packages/b9/63/df50ac39e0d5b006c55a399c3bf1db9da7b5a24de7890bc9cf5d4d9e99/certifi-2019.11.28-py2.py3-none-any.whl
Requirement already satisfied: chardet==3.0.4 in /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages (from -r requirements.txt (line 2)) (3.0.4)
Requirement already satisfied: future==0.18.2 in /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages (from -r requirements.txt (line 3)) (0.18.2)
Requirement already satisfied: idna==2.8 in /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages (from -r requirements.txt (line 4)) (2.8)
Collecting python-dateutil==2.8.1 (from -r requirements.txt (line 5))
  Using cached https://files.pythonhosted.org/packages/d4/70/d60450c3dd48ef87586924207ae8907090de0b306af2bce5d134d78615cb/python_dateutil-2.8.1-py2.py3-none-any.whl
Requirement already satisfied: python-whois==0.7.2 in /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages (from -r requirements.txt (line 6)) (0.7.2)
Requirement already satisfied: requests==2.22.0 in /Library/Frameworks/Python.framework/Versions/3.7/lib/python3.7/site-packages (from -r requirements.txt (line 7)) (2.22.0)
Collecting six==1.13.0 (from -r requirements.txt (line 8))
  Using cached https://files.pythonhosted.org/packages/65/26/32b8464df2a97e6dd1b656ed26b2c194606c16fe163c695a992b36c11cdf/six-1.13.0-py2.py3-none-any.whl
Collecting urllib3==1.25.7 (from -r requirements.txt (line 9))
  Using cached https://files.pythonhosted.org/packages/b4/40/a9837291310ee1ccc242ceb6ebf9eb21539649f193a7c8c86ba15b98539/urllib3-1.25.7-py2.py3-none-any.whl
Installing collected packages: certifi, six, python-dateutil, urllib3
Found existing installation: certifi 2019.6.16
Uninstalling certifi-2019.6.16:
  Successfully uninstalled certifi-2019.6.16
Found existing installation: urllib3 1.25.3
Uninstalling urllib3-1.25.3:
  Successfully uninstalled urllib3-1.25.3
Successfully installed certifi-2019.11.28 python-dateutil-2.8.1 six-1.13.0 urllib3-1.25.7
```

20. Open the file “run-test.py” in your favourite code editor, and update MLFORKIDS_API_KEY on line 4.
- Set it to the API key you found in Step 15.



```
run-test.py
1 from checks import checkAddressType, checkUrlLength, checkShortening, checkAtSign, checkPortNumber, checkDomainAge, checkRedirects, checkDomainRegistration
2 from mlforkids import classify
3
4 MLFORKIDS_API_KEY="ab2e36b1-d710-4e87-b077-6fd40f4a3e16ab2e36b1-d710-4e87-b077-6fd40f4a3e16"
5
6 def checkUrl(addr):
7     address_type = checkAddressType(addr)
8     url_length = checkUrlLength(addr)
9     shortening = checkShortening(addr)
10    includes_at = checkAtSign(addr)
11    port_number = checkPortNumber(addr)
12    domain_age = checkDomainAge(addr)
13    redirects = checkRedirects(addr)
14    domain_reg = checkDomainRegistration(addr)
15
16    print(addr)
17    print("    address type :", address_type)
18    print("    url length    :", url_length)
19    print("    shortening    :", shortening)
20    print("    includes @    :", includes_at)
```

21. Run the program
- python3 run-test.py**

The program will use the machine learning model that you've trained to predict whether a variety of URLs are safe or phishing:

<https://machinelearningforkids.co.uk/help>

Hopefully it will predict that this is safe, as I send this link to people very often!

<https://www.bbc.co.uk>

This is a very old and reputable website, registered in August 1996. Hopefully it will think this is a safe link.

https://en.wikipedia.org/wiki/Machine_learning

The Wikipedia page for Machine Learning is another safe link.

https://91.198.174.192/wiki/Machine_learning

This is the same address as above – it's the address for the Wikipedia page about Machine Learning, but using the IP address to save your web browser needing to look up the address for the Wikipedia domain name. It's safe link to visit, but it's unusual to describe it using the IP address, so your machine learning model might (incorrectly?) think that it is suspicious.

<https://mickey@bit.ly/35Jnlf8>

This actually points to the BBC News website <https://www.bbc.co.uk/news> but it uses several redirects to hide that, so your machine learning model will likely predict that it is suspicious.

<https://login.bankofamerica.com@www.josueizagirre.com/wp-content/cache/login.bankofamerica.com.uplogad/update.html>

This is a phishing website pretending to be a Bank of America page. Hopefully your machine learning model will correctly predict that it is suspicious.

<http://flinxdeicdcc.com/2612aa892d962d6f8056b195ca6e550d/tnFBsV27sc3kIMJg0Z8snqenramMHIXz.php?country=US-United%20States&lang=en>

This is another phishing website that your machine learning model will hopefully correctly recognize.

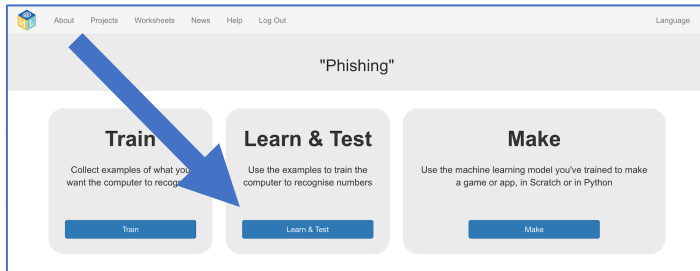
22. Edit the file `run-test.py` in your favourite code editor to test other URLs.

Look at how the example URLs above were tested to learn how to use the `checkUrl` function.

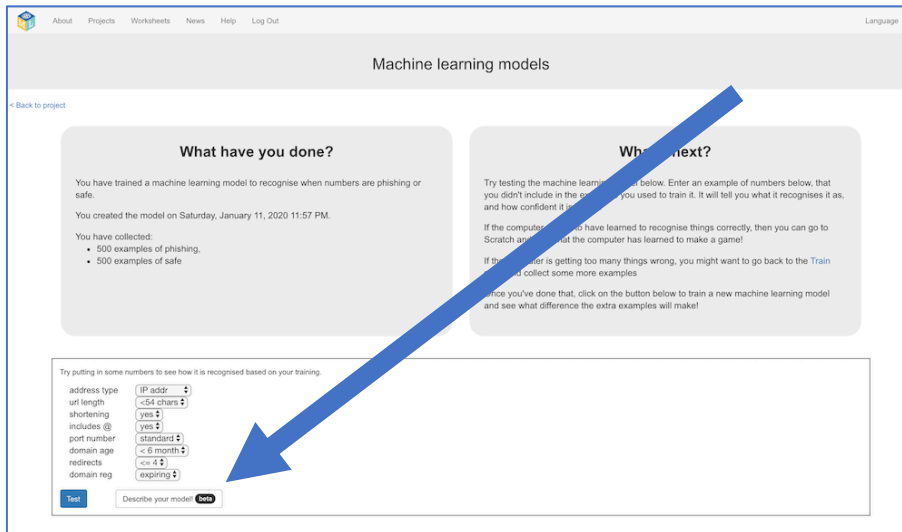
*Test the addresses for **legitimate websites** that you use and trust to see if your machine learning model predicts that they're safe.*

*Test the addresses for **phishing websites** to see if your machine learning model predicts that they can't be trusted. If you need to find URLs for phishing websites, <https://phishtank.com> is a good place to look. Find new URLs that have only just been reported as phishing sites to see how your ML model copes.*

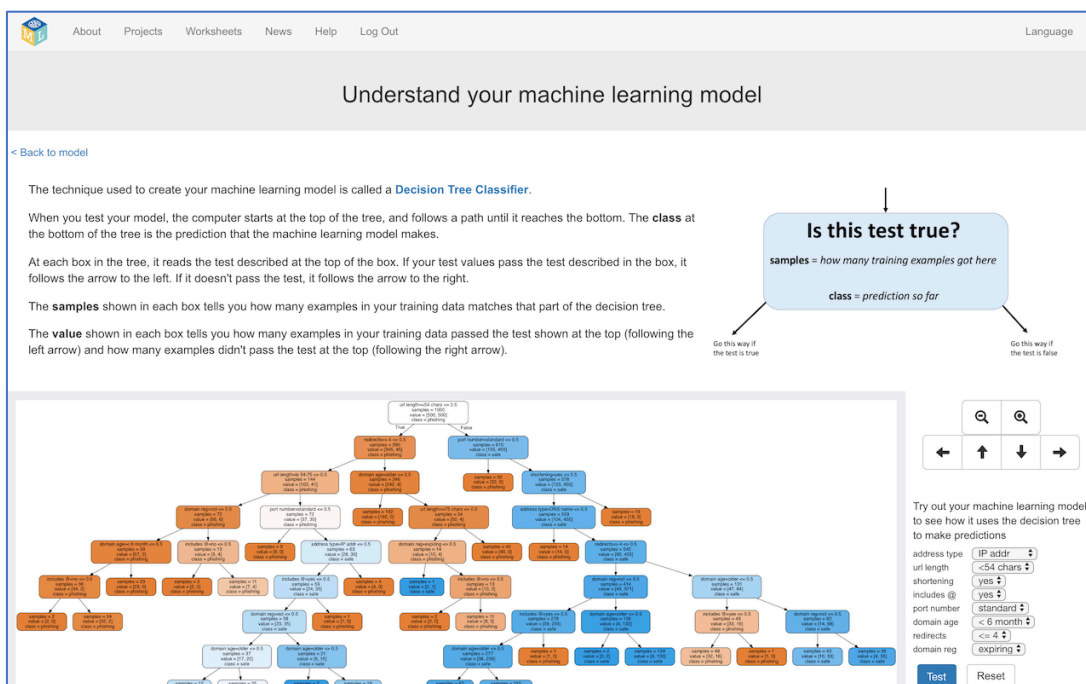
23. Back on Machine Learning for Kids, click the “Learn & Test” button



24. Click the “Describe your model” button



25. Examine the visualisation for the machine learning model *This shows how your model is making predictions. Use the **Test** button to see how it works.*



What have you done?

You've used a pre-prepared dataset to train a type of machine learning model called a "decision tree classifier" so that it can predict if a URL is going to be for a legitimate website or a phishing website.

You've used that machine learning model from a Python program, which you've modified so that it can make predictions about your own web addresses.

26. Plan your own project!

Importing the template project means you didn't have to decide which features the machine learning model should use.

Your project used eight features (described in Step 8 above).

Think about how you could improve your machine learning model by adding additional features.

What other attributes about web sites do you think would help your machine learning model to be better at recognizing phishing URLs?

The following pages include links to some research projects on this topic which might help to give you some ideas.

Machine Learning research about Phishing

Training machine learning models to recognize phishing URLs is an active topic in ML research.

Try looking at some of the papers listed below.

Even if you don't understand everything they say, running your own experiment should help you get the general idea.

Do any of these papers give you an idea for your own project?

<https://ibm.biz/phishing-hassan>

This paper describes Hassan's research into 30 different attributes of phishing links. Twelve of these are based on attributes of the URL, as you did in your project.

Some attributes described are based on the way the link was displayed. For example, if the phishing link was in an email, was the URL displayed when you hover a mouse pointer over the link?)

<https://ibm.biz/phishing-rajab>

This paper describes Rajab's research into how to tell which features are the most useful to recognize phishing URLs.

They identified the protocol (whether the page starts with http or https) and whether the SSL certificate for a website is signed by a trusted source as being two of the most significant attributes to use.

<https://ibm.biz/phishing-joshi>

Joshi & Pattanshetti describe 48 attributes that they found to be useful, including whether the page contains secure forms, whether forms on the page try to send emails, whether the icon on the address bar looks valid, and many more.

<http://ibm.biz/phishing-basnet>

Basnet, Sung & Lui describe how they reviewed 177 different attributes of URLs to identify the most useful to train a machine learning model.

Some of these looked at the contents of the web pages themselves, not just the URL address of them, such as counting the number of links on the page, whether the page contains iframes, whether it has password fields in it, and so on.

<https://ibm.biz/phishing-mohammad>

Mohammad, Thabtah & McCluskey describe a variety of features they identify as useful, some based on the URL like you have done, but some based on the use of JavaScript on webpages, and some using data from external sources like Google, PhishTank, Alexa, and others.

They describe the attributes they used very clearly, and you should recognize several of these from features in your project.

<https://ibm.biz/phishing-shirazi>

Shirazi, Haefner & Ray build on the Mohammad et al paper described above, and say how they identified the 10 most useful attributes they used to make “Fresh-Phish”.

One of these was the Google ranking for a URL, relying on Google to have a high rank for legitimate websites.

<https://ibm.biz/phishing-whittaker>

Whittaker, Ryner & Nazif describe how Google have evaluated URLs to identify phishing pages.

They describe features such as the presence of terms like “customerservice” in the URL, which they found to be more common in phishing sites.