E-mail marketing with cloud services – an overview

Pavel Stoynov,
Slavi Slavchev
Sofia University "St. Kliment Ohridski",
Faculty of Economics and Business Administration,

Abstract.
E-mail marketing is developing very fast in last decade and recently broadly used for offering financial services. It shows low costs and is very powerful marketing tool at the same time. In this article, we consider the e-mail marketing process and some ways of using Python software language to fix some errors in the e-mail list.

Keywords: e-mail marketing; cloud services; SaaS

1. Introduction.

E-mail marketing is developing very fast in last decade. It shows low costs and is very powerful marketing tool at the same time. In this article, we consider the e-mail marketing process and some ways of using Python software language to fix some errors in the e-mail list.

2. Process of e-mail marketing.

The first step of the process is to prepare the sample of addressees. We could find them in Internet, from different address books etc.

After creating the list of addresses, we must ensure that consumer protection laws are followed (for example opt-out should be present in the e-mails sent by e-mail marketing tool).

The next step is to start sending messages periodically ensuring that enough time passes between two consequitive e-mails.

Then a software tool for marketing must be used to send the mails to the recipients automatically. Sendy, for example, is easy-to-use tool, low cost, software as a service (SaaS). The actual sendout of emails, including in the case of Sendy, is straightforward, with very good explanations and engaging user interface which helps out sending the emails in a very easy way

3. Using Python for improving e-mail list.

The programming language Python can be utlised for processing big lists of emails. Imagine a very long list of emails in the following, at first seeming inconvenient format:

```
emailBase = Pavel Stoynov Sofia University email: pavel.t.stoynov@gmail.com assistant
Slavi Slavchev Sofia University email: s_slavchev@feb.uni-sofia.bg PhD candidate
```

Then use some medium-complexity programming code to separate those emails.

With Python having an easy syntax, this task is relatively straightforward, with the help of Google if needed.

```
splitLines = emailBase.split('\n') # records each line
for k in splitLines: # goes through each new line
   if "@" in k: # if you find the @ sign...
      k = k.strip(" \t\n\r") # removes any tabs and surplus spaces
      splitByWhiteSpace = k.split(' ') # splits by space
      for m in splitByWhiteSpace: # considers the new output
         if "@" in m: # if you find the @ sign in this new output
            thisEmail = m # the email is now found
            break # no need to proceed with this line
      print (splitByWhiteSpace[0], "\t", splitByWhiteSpace[1], "\t", thisEmail) # prints the name, surname and email, split by tab
```

Emails often have typos like:

```
John.smith@gmail,com
John.smith@gmail
```

In such cases, again Python can be utilised, with some quick research on Google, to capture wrong emails and potentially not waste the sample.

```
import re  # we tell Python we want to use the so-called Regular Expressions
def validateEmail(email): # start writing a function
  if len(email) > 7:  # it is unlikely an email is less than 7 characters
    if re.match("^.+\\@(\\[?)[a-zA-Z0-9\\-\\.]+\\.([a-zA-Z]{2,3}|[0-9]{1,3})(\\]?)$", email) != None:  # use an email validator
      return 1  # if the email is valid then we say "1" (true)
  return 0  # otherwise we say "0" (false)
for k in emails:  # iterate through each email
  if validateEmail(k)==0:  # if the email is not valid
    print(k)  # then print it
```

This task can usually be done by the email sendout tool itself, usually with the bounce-back report (if an email fails to be sent, it marks it as invalid). If there is no built-in email validator beforehand, there are disadvantages:

- You only realise the emails are invalid once already sent, which means amending them and resending the email – and this is double work
- Usually a separate campaign needs to be done which means the report stats will be skewed

Another proofreading guideline which can be checked is the correct domains. For example this email address seems valid at first:

```
Hanns.gass@raiffaisen.de
```

but if we look closely we'll see that raiffeisen is misspelt. We can use a plain built-in console program (available in various operation systems) - in the Windows case, CMD:

```
ping raiffaisen.de
Ping request could not find host raiffaisen.de. Please check the name and try again.
```

Either with Python or with Excel (or other manner) we can find all distinct domain names and check them before uploading the sample file – again beneficial for better planning of the sample.

4. Conclusion.

E-mail marketing becomes more and more popular. With the available software tools for e-mail marketing it becomes easy to use and very effective way of dowing marketing in web.