TCSS 555: Machine Learning
User Profiling in Social Media

# 1 Introduction

Social media platforms like Facebook, Instagram, TikTok, YouTube, and more, have made it easy for users to create their own content on the web. There is a lot of interest to mine this data for use in personalized information access services, recommender systems, tailored advertisements, and other applications that can benefit from personalization. Research in psychology has suggested that behavior and preferences of individuals can be explained to a great extent by age, gender and underlying psychological constructs (or so called "personality traits"). In addition to a myriad of applications in e-commerce, user profiling is used in digital text forensics as well, see e.g. http://pan.webis.de/

The goal of this project is to build a system for automatic recognition of the age, gender, and personality of Facebook users. When given as input the status updates, profile picture and "likes" of a Facebook user, this system should return as output the age, gender and personality trait scores of that user.

# 2 Supervised Learning Tasks

In this project you will use machine learning techniques to automatically generate a program that takes as **input** the following information from a Facebook user:

- **status updates**
- **profile picture**
- **page "likes"**

and infers the following information about the user as **output**:

- **gender**, as either "male" or "female"
- **age**, as either "xx-24", "25-34", "35-49", or "50-xx"
- **personality**, as a score between $[1, 5]$ for each of the five traits of the Big Five personality model, namely Openness to experience, Conscientiousness, Extroversion, Agreeableness, and Emotional Stability (reversely referred to as Neuroticism).

Table 1 contains a brief description of the personality traits of the Big Five Personality Model. Inferring gender is a *binary classification* or concept learning task.[1] Inferring age is a *multi-class classification task*. Inferring the personality scores corresponds to solving five *regression tasks*.

The folder Files/project/papers on the course Canvas website contains papers that describe how other people have approached the same or very similar problems. To get a better understanding of the problem domain, it is highly recommended that you read one or more of these papers. They are named according to the data source that they leverage, i.e., text, image, or relation. A good starting point for your literature survey might be text-Farnadi-2013.pdf [1], where personality prediction is treated as a classification problem rather than a regression problem.

You will work on the project in a team of maximum 3 students. Within a team, each student should work on making inferences based on a different source. That means that in your team, you are either responsible for making inferences based on **status updates** (text), or based on **profile pictures** (images), or based on **likes** (relational data). By the end of week 3, you should agree within your team who is going to work on which source, *and you should indicate this in the last column of the sign-up sheet in week 1 on*

[1] The treatment of gender as a binary concept in this project stems from the fact that the Facebook data used in the project only includes the genders "male" and "female". In 2014, Facebook introduced dozens of other options for users to identify their gender. The data used in the project was collected before that time.

Table 1: Overview of the Big Five Personality Model.

| Trait | Description |
|---|---|
| **Openness** | Openness is related to imagination, creativity, curiosity, tolerance, political liberalism, and appreciation for culture. People scoring high on Openness like change, appreciate new and unusual ideas, and have a good sense of aesthetics. |
| **Conscientiousness** | Conscientiousness measures preference for an organized approach to life in contrast to a spontaneous one. People scoring high on Conscientiousness are more likely to be well organized, reliable, and consistent. They enjoy planning, seek achievements, and pursue long-term goals. Non-conscientious individuals are generally more easy-going, spontaneous, and creative. They tend to be more tolerant and less bound by rules and plans. |
| **Extroversion** | Extroversion measures a tendency to seek stimulation in the external world, the company of others, and to express positive emotions. People scoring high on Extroversion tend to be more outgoing, friendly, and socially active. They are usually energetic and talkative; they do not mind being at the center of attention, and make new friends more easily. Introverts are more likely to be solitary or reserved and seek environments characterized by lower levels of external stimulation. |
| **Agreeableness** | Agreeableness relates to a focus on maintaining positive social relations, being friendly, compassionate, and cooperative. People scoring high on Agreeableness tend to trust others and adapt to their needs. Disagreeable people are more focused on themselves, less likely to compromise, and may be less gullible. They also tend to be less bound by social expectations and conventions, and more assertive. |
| **Emotional Stability** | Emotional Stability, reversely referred to as Neuroticism, measures the tendency to experience mood swings and emotions such as guilt, anger, anxiety, and depression. People scoring low on Emotional Stability (high Neuroticism) are more likely to experience stress and nervousness, while people scoring high on Emotional Stability (low Neuroticism) tend to be calmer and self-confident. |

*Canvas.* As an individual student, you should use your own source to make inferences about gender, age, and personality. Students usually start with making machine learning models for inference of gender, and then, time permitting, expand this to the other tasks (age and personality) later in the quarter.

# 3 Training Data

A training dataset with 9500 labeled instances is provided. Before using this dataset, please sign the "Terms of use agreement" (available in week 1 on the course Canvas website). The dataset contains profile information of 9500 Facebook users, collected with the MyPersonality[2] application. The MyPersonality app was a popular Facebook app introduced in 2007. Users using the MyPersonality app took a standard Big Five Factor Model psychometric questionnaire and gave consent to record their responses and Facebook profile information. The training dataset includes the following information:

- **profile info**: a csv file with the userid, age, gender, and personality scores of all 9500 users. In the "gender" column, 0 denotes male and 1 denotes female.
- **text**: text files with the status updates of the users. All the status updates of a single user are combined into one file called "userid".txt. There are 9500 such text files.
- **image**: profile pictures of the users, with the naming convention "userid".jpg. There are 9500 such images.
- **relation**: a csv file with rows of the format "userid", "likeid", indicating that the user with "userid" liked the page with "likeid".

All 9500 users declared in their Facebook profile that their language is English. Still, a small fraction of the status updates is not in English.

[2] `http://mypersonality.org`

# 4 Required Submission Format

Prepare your software so that it can be executed via a command line call. The testing command `tcss555` (see section 5, item 7 for details on how to create) shall take as input (i) an absolute path to a test dataset with new instances (not containing the labels) and (ii) an absolute path to an empty output directory; the format looks like this:

```
tcss555 -i path/to/test/my-test-data/ -o path/to/output/directory/
```

Here is an example of what the script invocation might look like in a terminal window for Team 1's itadmin account:

```
itadmin@555team1:~$  tcss555 -i  /data/public-test-data/ -o ~/output/
```

The format of the test dataset is the same as the training dataset, with an identical internal directory structure. For each user of the test dataset, your software must output a corresponding XML file that looks like this:

```
<user
    id="8157f43c71fbf53f4580fd3fc808bd29"
age_group="xx-24"
gender="female"
extrovert="2.7"
neurotic="4.55"
agreeable="3"
conscientious="1.9"
open="2.1"
/>
```

This file should be saved in the specified output directory and have the user's id value as the base file name and "xml" as its extension. For example, the file name of the XML file whose contents are shown above would be:

```
8157f43c71fbf53f4580fd3fc808bd29.xml
```

A public test dataset with data of 334 Facebook users (no labels though!) is available in the folder `/data/public-test-data`. Check that your software runs properly as itadmin by executing the command:

```
tcss555 -i /data/public-test-data/ -o ~/results/week3/
```

You should also check that your software runs as another user; see section 5, item 8.

Using a similar command as the one above, the predictive capabilities of your software will be tested on a "hidden" test dataset of $n = 1334$ Facebook users. This set contains the 334 users from the public test dataset, as well as 1000 new users who are not in training dataset nor in the public test dataset. Your solutions for age and gender will be assessed based on **accuracy**, i.e. the number of correctly classified instances divided by the total number $n$ of instances. For personality identification, the average **Root Mean Squared Error (RMSE**) over all five personality traits will be used. RMSE is defined as

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

with $y_i$ the actual value and $\hat{y}_i$ the predicted value.

**Your software will be tested every week at some time point between Wed 9 am and Thu 9 am. Make sure that every Wed at 9 am, the command `tcss555` invokes a stable version of your**

**software on your VM, and that this stays available until at least Thu at 9 am. Testing will take place on:**

- **Wed Oct 9, 9 am (graded)**
- **Wed Oct 16, 9 am**
- **Wed Oct 23, 9 am (graded)**
- **Wed Oct 30, 9 am**
- **Wed Nov 06, 9 am**
- **Wed Nov 13, 9 am (graded)**
- **Wed Nov 20, 9 am**
- **Wed Nov 27, 9 am**
- **Mon Dec 02, 9 am (graded)**
- **Fri Dec 06, 9 am (graded)**

# 5 Virtual Machine User Guide

1. **Downloading tool to manage the VMs**

   Please read: `http://css.insttech.washington.edu/~lab/Support/HowtoUse/UsingVCLQS.html`

   Using VMs on a Windows or Mac computer is fairly easy because the manage_vc app takes care of a lot of details for you. Linux users have to do a lot more work, but it is still possible to do. One can always use a SET Lab computer to access the VMs.

2. **Determining team information**

   A team account was created for each team – if need be, see `http://css.insttech.washington.edu/~lab/Support/HowtoUse/RequestAndUseTeamAcct.html`

   The team accounts are called _555team1, _555team2, etc. The leading underscore is significant – it's the first character of the team account name.

   Students have been assigned as members of each team according to the UW Net ID information provided. User mdecock is a member of each team.

   You never login using the team account – the account exists as a shared place to store information. The information about the VM for the team is stored on the team account, and you use your credentials to access it (as described in the web page and below).

3. **Understanding the VM**

   There is one VM called ubuntu that resides on each team account. The initial amount of RAM is 8GB, and the disk is about 100GB. The amount of RAM can be changed if need be. The operating system installed is Ubuntu Workstation 22.04 LTS.

   The information about that VM is stored in the VCL information file. manage_vc will retrieve that information for you after you provide your credentials.

   When prompted by manage_vc for a username, you should provide the information in this form:

   ```
   uwnetid _555teamx
   ```

   where uwnetid is your UW Net ID (e.g., mdecock) and you must change the "x" in "_555teamx" to a number from 1 to the maximum number of teams, which corresponds to your team number. For example:

```
mdecock _555team4
```

Note the space between the end of the UW Net ID and the underscore – it is significant. The uwnetid must be a member of the team account used.

After clicking on OK, you will be prompted for **your** MyUW password. Once it is entered, the VCL information will be pulled from cssgate and placed in a local file.

4. **Managing the VM via manage_vc**

   Start the VM by pressing the "Start" button. If the VM is in the powered off state, then this is equivalent to turning on a computer. If it is already on, it will tell you, or you can check its state by clicking on the "Show State" button.

   Once a VM is started, you can connect to its display by clicking on the "Connect" button.

5. **Using Ubuntu**

   Connecting to the remote display should cause a window to open up. You are prompted for the itadmin password. You should attempt to paste the password using the "Paste itadmin password" button, but if it doesn't work, click on "Reveal itadmin password" and type it in manually.

   Once logged in, you can click on the dashboard icon in the upper left and a search box will appear. Enter "terminal" to open a terminal window. Alternatively, there may be a terminal icon already in the dock that you can click on to open the terminal window.

   Now you can enter unprivileged commands. If you need to elevate your privileges to that of the superuser:

   ```
   sudo command args
   ```

   is the best way to do so, since only the command you specify will run with superuser (root) privileges. sudo needs the itadmin password to execute the "command" that needs elevated privileges, such as:

   ```
   sudo apt install build-essential
   ```

   apt is the package management tool for Ubuntu. This example would install the "build-essential" package.

   Other commands or icons on the screen may not require special privileges, and can be used as is.

   To find out if something, for example "pandas", is available as a package in an Ubuntu repository, enter:

   ```
   sudo apt search pandas
   ```

   You may see a list of all package names that match, and then you can select which package you want to install. Often, this installs the package and all of its dependencies so that any user, not just itadmin, can use it.

   You should check the version of the package, as it may not be compatible with what you want to use. For example, "sklearn" is also known as "scikit-learn". You can see the ubuntu VM's `python3-sklearn` version by running:

   ```
   sudo apt show python3-sklearn
   ```

   However, some students want to use "model_selection" from sklearn, which is only available starting with version `0.19.1`, so you must use the pip3 method instead to install – see item 6.

6. **Using other package repositories**

   If you download packages from places other than Ubuntu's package repositories, you will need to be careful about how you install them. Look for ways to install the package or dataset so that `any` user can use them.

   If you are using the Python language, be aware that there are two versions of the Python interpreter installed. The default is called "python", and is version 2.7. The other is "python3", and is version 3.10. We recommend using python3, and all documentation reflects that.

   Python's module installer, commonly called "pip3", can install the latest version of packages from Python module repositories on the internet.

   pip3 installs modules for the current user by default, which means the default won't work for any other user, such as "ituser" or the user that runs the evaluation scripts. In order to install modules for any user, you must use the correct and version-specific "–target". Because the target is not usually writable by itadmin, you must use "sudo pip3" to install.

   For example, this is how to tell pip3 where to install the "pandas" module so that any user could import it:

   ```
   sudo pip3 install pandas --target=/usr/local/lib/python3.10/dist-packages
   ```

   Note that the version-specific folder `/usr/local/lib/python3.10` corresponds to the current version of python3.

7. **Creating the tcss555 script**

   The "tcss555" script must be executable and be located in your home directory (referred to by the tilde character "~"). An empty executable script file can be created as follows:

   ```
   cd ~
   touch tcss555
   chmod +x tcss555
   ```

   Then use a text editor to change tcss555 to however you will process the dataset as required by the weekly assignment. Make sure that tcss555 can accept the arguments: "-i" as an option for the input directory followed by an input directory path with a trailing slash "/" and "-o" as an option for the output directory followed by an output directory path with a trailing slash "/".

8. **Testing as another user**

   There is another user account called "ituser" which has the same password as itadmin. It can be used to test the tcss555 script as another user, similar to how a different user will run and evaluate your tcss555 script on the hidden test data. For a valid test, you should only run the instructions below – don't install anything as ituser or copy any files to its home directory.

   To test as another user, you should login to the ituser account, make an empty output directory, and try to run the tcss555 script:

   ```
   rm -rf ~/output
   mkdir output

   cd /home/itadmin
   tcss555 -i /data/public-test-data/ -o ~/output/
   ```

   If something doesn't work, it could be due to file permissions (e.g., only itadmin owns the file or can read it) or the data that the script references is only available from itadmin. Using the itadmin (NOT the ituser) account, you can then fix file permissions, dataset placement, or other problem such that ituser can run the script, increasing your confidence that the evaluation user account will not encounter any problems.

9. **Downloading tcss555.zip**

   The public-test-data and training folders are available from within the VM. First, you will need to know your VM's IP address (use the terminal window, or see step 13b):

   ```
   ip -br a
   ```

   It should be the IP address starting with "172.22.71.". For example, it could be "172.22.71.31".

   Secondly, you need to have some kind of secure copy program installed on your computer. If you use Windows, putty's pscp command can be used (as shown below). Mac and Linux users can use the scp command. Here is an example using Windows and the pscp command (which must be in your command path):

   ```
   mkdir c:\temp
   cd /d c:\temp
   pscp itadmin@172.22.71.31:/data/tcss555.zip .
   ```

   The final period indicates to securely copy the file to the current directory, which in this example is `c:\temp`.

   If you get a "permissions denied" or "access denied" error, it may be that the permissions of the `/data/tcss555.zip` file do not allow user accounts other than the owner (probably "root") to read it – ask SET lab staff to change that.

10. **Using command line access to VM**

    Command-line access to the Ubuntu VM via ssh (or putty on Windows) works fine, and provides a similar environment as the terminal window in ubuntu.

    Of course, without a GUI desktop, graphical tools such as "gedit", a web browser or a file system browser won't work. Instead of gedit, try using "nano" (or "vi" if you are brave). There is no good command-line substitute for a web browser, but if you are using it only for copying files, see the steps starting with step 13. As for browsing the file system, you should use the "cd" and "ls" commands.

    Here is a brief Linux command reference, categorized by task: `http://tinyurl.com/uwt-cs-linux-cmds`

    You can copy files from your computer using an "scp" ("pscp" on Windows) command, as documented below (starting in step 13). Since graphical SCP tools like WinSCP or FileZilla take a lot of time to document, we leave that exercise to you to figure out how to download, install and configure them to copy files to cssgate from your home network, or, from within the UW network or VPN (see step 12), directly on the Ubuntu VM.

11. **Changing the itadmin password for the VM**

    All teams are advised to change the itadmin password for their VM. This will prevent other teams from accidentally logging into and making changes on your VM. To change the itadmin password, in the terminal window, type the command:

    ```
    passwd
    ```

    and enter the old password then your desired password (twice, as prompted). Communicate that password to all team members.

12. **Determining IP address of the Ubuntu VM**

    Please recall that the IP addresses used by both the VM's host and the Ubuntu VM may be local to the UW, which means you can't directly get to it from off-campus. Since the off-campus method also works on-campus, we are providing instructions for just that way.

    The preferred off-campus method is to use the "Husky OnNet Remote Network Access (VPN)" to gain access to the UW-local network resources from home – see this web page: `https://itconnect.uw.edu/connect/uw-networks/about-husky-onnet/use-husky-onnet/`

VPN access requires the download and installation of a package, and then a connection step to the VPN. We leave it up to you to figure out and use, but after connecting you should be able to use ssh/putty or scp/pscp IP-based access to your Ubuntu VM directly from home. To get the IP address, see step 13.

13. **Logging into your Ubuntu VM from cssgate**

    Make sure you understand step 12 before continuing with this step.

    (a) **Logging into "cssgate.insttech.washington.edu" (the host name)**
    Use ssh or putty; for the Windows-based putty, see step 14a.
    You must provide your UW Net Id as the user name and your MyUW password as the password.

    (b) **Determining your Ubuntu VM's IP address from cssgate**

    ```
    grep 555teamn /etc/hosts
    ```

    which, if you change the last "n" in "555teamn" to your team number, will find your team's IP address/name pair in the `/etc/hosts` file, and show it to you. For example, for team 6:

    ```
    grep 555team6 /etc/hosts
    ```

    should show:

    ```
    172.22.71.36 555team6
    ```

    (c) **Logging into the Ubuntu VM**
    After successfully logging into cssgate, use the ssh command on cssgate to login to your Ubuntu VM; generically, that looks like:

    ```
    ssh itadmin@555teamn
    ```

    where the last "n" in "555teamn" is your team number.
    The name of the form "555teamn" only works from a cssgate login session.
    Of course, when you login via ssh with `itadmin@172.22.71.x` or `itadmin@555teamn`, you must supply the itadmin password. The ssh login works the same for any other user on your ubuntu VM, such as ituser or any user account you created.
    Once logged in, you can run whatever commands you want.

14. **Copying files from your computer to the Ubuntu VM**

    (a) ***From Windows*, copying the file(s) from your computer to cssgate**

    i. **Downloading pscp.exe (and/or putty.exe)**
    We suggest creating a folder called `C:\tools` and putting it in that folder using the command shell:
    `mkdir c:\tools`
    You can find pscp.exe and putty.exe from `https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html`
    By putting "pscp.exe" in `c:\tools`, you'll know exactly where it is. Note that this is a command-line tool, so you'll want to run it from a command shell in Windows.
    You only need to download and install once.

    ii. **Copying the file from your Windows computer to cssgate**
    Let's say my computer has a login account of "stephen" and I am running Windows 7, 8 or 10, my file ("week5.py") is on the Windows desktop (`C:\users\stephen\desktop`), and my UW Net Id is "srondeau". The following commands would copy the "week5.py" file to the home directory of my cssgate account if I provide my MyUW password:

```
cd /d c:\users\stephen\desktop
c:\tools\pscp week5.py srondeau@cssgate.insttech.washington.edu:
```

The trailing colon ":" is always significant – don't forget it! If you do, you will end up with a file called `srondeau@cssgate.insttech.washington.edu` whose contents are exactly the same as week5.py.

iii. **Proceed to step 14c**

(b) ***From a Mac*, copying the file(s) from your computer to cssgate**

Use scp from a terminal:

i. **Copying the file to cssgate**
Let's presume "week5.py" is on my home directory on the Mac, which is the default when a terminal window is opened:

```
scp week5.py srondeau@cssgate.insttech.washington.edu:
```

The trailing colon ":" is always significant – don't forget it!

ii. **Proceed to step 14c**

(c) **Copying the file(s) from cssgate to your Ubuntu VM**

Let's say you are in team6:

```
scp week5.py itadmin@555team6:
```

or you could use the IP address obtained in step 13b:

```
scp week5.py itadmin@172.22.71.36:
```

Again, you must supply the itadmin password.

(d) **Copying using a directory and the "-r" (recursive) option to pscp or scp**

```
scp -r mydir itadmin@172.22.71.36:
```

Consequently, if you have a lot of files you want to copy to the Ubuntu VM, it may be best to put them all in one directory and copy that directory, in one command.

# 6 Workflow Instructions

Here is an approach to tasks to complete to provide the best chance of a successful evaluation on the VM:

1. Make sure you can gain network access to the VM from off campus.
   (a) Install and use Husky OnNet (section 5, item 12),
   ** OR **
   (b) Use cssgate as an intermediate system (section 5, item 13).
2. Determine your VM's IP address (section 5, item 13).
3. Put your code on the VM.

   Remember that your code must accept two arguments, an input directory and an output directory (section 4). It must either concatenate the input directory to the desired input data files (.csv files), including any intermediate subdirectory (such as 'text' or 'profile'), or open the input directory and specify the relative path of each desired input file to open it. It must either concatenate the output directory to the name of each .xml file or open the output directory and write all files there.

   (a) Write your code on your team's itadmin account on the VM directly.
   ** OR **

(b) Download tcss555.zip once (section 5, item 9) onto your home computer or laptop, write your code and test it there, then upload everything you need to itadmin on the VM using scp/pscp or WinSCP (section 5, item 14).

4. Test your code, logged in as itadmin on the VM

   (a) Make sure the tcss555 script works to start your program (section 5, item 7)

   (b) Correct any syntax errors in your code.

   (c) For packages/modules not installed, install them for everyone (section 5, items 5 and 6).

   (d) Continue to test and fix tcss555 (and your code) until it can run without errors.

5. Test your code as ituser on the VM (section 5, item 8)

   Make sure you **do not modify** or install packages to the ituser account.

   Testing as ituser as instructed will check to make sure any packages/modules you have are accessible by other users, such as the user account used to evaluate your code.

6. Verify that your output files are correct:

   (a) The number of .xml files created must be the same as the number of user ids from the input directory's profile.csv file.

   (b) The content of each .xml file is as defined in section 4, including valid values as specified in section 2.

# 7 Description of the Data

The data provided is in a folder called "/data/training" (9500 users) and a folder called "/data/public-test-data" (334 users). Both folders are internally structured in the same way. They contain:

- A subfolder "profile". This folder contains a csv file with one row per user. This csv file has columns for user id, age, gender, and the five personality scores. For the training dataset, all values are available to you in the csv file. For the public test dataset, only the the user id is listed, and the other columns are left blank.

- A subfolder "text". This folder contains one text file per user, named "userid".txt, combining all the status updates of the user. Most status updates are in English, although you might encounter some text in other languages (e.g. Spanish) as well.

- A subfolder called "image". This folder contains one picture per user, named "userid".jpg.

- A subfolder called "relation". This folder contains a csv file with rows of the format "userid", "likeid", indicating that the user with "userid" liked the page with "likeid".

- A subfolder called "LIWC". This folder contains a csv file with one row per user. The columns correspond to 82 LIWC features, extracted from the status updates. In Files/project/papers/text-Tausczik-2010.pdf on Canvas, you can find an overview of what each of the LIWC features means [2]. The paper text-Farnadi-2013.pdf in Files/project/papers on Canvas mentions only 81 LIWC features [1]. The reason is that in our data the first feature (Seg) has the same value for all users. Hence it can be dropped.

Each week, your system will be tested on data in a different folder. The directory structure for the hidden test directory is identical to the directory structure of the public test directory.

# 8 Programming Languages and Packages

You can choose freely among the available programming languages. We recommend to use **Python**, **R** or **Java**. Many good machine learning packages are freely available and can be installed on your VM.

**Languages:**

- Java: `https://www.java.com/en/download/help/linux_x64_install.xml`
- Python: `https://docs.python-guide.org/starting/install3/linux/#install3-linux`
- R: `https://cran.r-project.org/doc/manuals/r-release/R-admin.html#`

**Machine learning packages:**

- Python packages
  - **sklearn**: the scikit learn library provides machine learing and data mining tools. For more info, please check http://scikit-learn.org/
  - **TensorFlow**: an open source library for machine learning and deep neural networks developed by Google. For more info, please check https://www.tensorflow.org. **Keras** is a high-level neural networks API that runs on top of TensorFlow, see https://keras.io/. If you want to train and test your own deep neural networks and you have not worked with TensorFlow before, you will want to install Keras in addition to TensorFlow, and design and run your models using the Keras API.
  - The **Keras** webpage contains an illustration of transfer learning, with code examples, that will be useful for students working on deep learning for image classification: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html
  - **PyTorch**: an open source tensor based library optimized for machine learning and deep learning, primarily developed by Meta's AI Research Lab. For more info, please visit https://pytorch.org/. You can install PyTorch following the instructions on https://pytorch.org/get-started/locally/.
  - Students working on deep learning and using PyTorch for image classification may have to install **Torchvision**, see https://pytorch.org/vision/stable/index.html and refer the examples there within.
- R packages
  - CRAN: provides a wide variety of statistical and graphical machine learning techniques. For more info, please check https://cran.r-project.org/
- Java packages
  - Weka: has a collection of machine learning algorithms for data mining tasks. For more info, please check http://www.cs.waikato.ac.nz/ml/weka/

OpenCV is a library for image processing `http://opencv.org/`. It has C++, C, Python and Java interfaces. The image data in the project is somewhat noisy, and you may want to remove images that do not contain faces in them, or that contain multiple faces. You can use OpenCV Haar Feature-based Cascade Classifiers to detect faces: https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html

# 9 Grading

The project counts for 35% of your final grade. The project will be graded out of a total of 35 points.

- Results on the scoreboard on Oct 10 (week 2): 2 points. If your results are at least as good as the baseline for all prediction tasks, your team gets full credit for this part.
- Progress update during class time on Oct 24 (week 4): 3 points. This is a group presentation, with slides. All team members should present. Students who do not present can not get credit for this part.

- Progress update through a video presentation posted on Canvas on Nov 14 (week 7): 5 points. This is a group presentation, with slides. All team members should present. Students who do not present can not get credit for this part.
- Final presentation during class time in week 10: 7 points. This is a group presentation, with slides. All team members should present. Students who do not present can not get credit for this part.
- Code and documentation: 8 points
- Report: 10 points (= 8 points group report + 2 points individual report)

Your grade for the progress updates is based on a 5 minute presentation (approx. 2-3 slides) and the results of your software so far.

# 10 Deliverables

**Slides for progress update 1 (1 upload per team; Oct 24)** Upload your slides on Canvas.

**Video and slides for progress update 2 (1 upload of each per team; Nov 14)** Upload your slides and post a link to your video on Canvas.

**Slides for final presentation (1 upload per team; Dec 3)** Upload your slides on Canvas.

**Code (1 upload per team; Dec 6, midnight)** To be submitted on the course Canvas. Submit a `zip` file containing your code and an additional documentation file in PDF, called `readme.pdf`. The documentation should contain enough details to allow a student who will take the machine learning course next year to understand the general structure of your code, compile and run it.

**Group report (1 upload per team; Dec 6, midnight)** Provide a write-up of your research in the form of an academic paper that could be submitted to a conference on data mining/machine learning. Your paper should be self-contained. Everyone who has read the assigned reading materials from the course should be able to read and understand your paper. That means that in your paper you can be brief about machine learning methods that are described in the assigned readings, but that you need to provide sufficient details about the problem domain, the dataset, as well as about any other machine learning methods that you used that were not covered in class. The reasons for this are: (1) a description of the problem domain and the dataset will allow to share your paper with interested parties who have not taken TCSS555 but who have general knowledge of machine learning; (2) a description of machine learning methods not covered in class will allow to evaluate whether you truly understood those methods instead of treating them as a black box. Your paper can for instance be divided into sections as follows (but if another structure works better for you, don't feel restricted to the one below):

1. Introduction: a description of the problem (profiling of Facebook users), what the goals of the study are, and a very brief description of the results.
2. Methodology: a brief description of the machine learning methods used.
3. Dataset and metrics: a description of the datasets and the evaluation measures used.
4. Results: an overview of the results you obtained by applying the methods from section 2 to the dataset from section 3 using the metrics from section 3. In addition to reporting numbers, your analysis of the results should also contain your insights into the results, i.e. why did a particular method work well/did not work well?
5. Conclusion and future work: briefly summarize your results and list opportunities for future research that seem promising to you but for which you did not find the time within this quarter.

Formatting guidelines: up to 8 pages, double column, ACM Proceedings format.[3] In case you need more than 8 pages, consider splitting your material in a main paper and an appendix. Submit a soft copy on the course canvas.

[3] `https://www.acm.org/publications/proceedings-template`

**Individual report (1 upload per student; Dec 6, midnight)** You will also submit a brief individual report (at most one page), which will:

- Describe the parts of the project you worked on (which machine learning methods you applied, which preprocessing steps you performed on the data, which parts of the term paper you wrote, who you worked with on what parts, etc.) and what parts of the project your teammates worked on.
- What you learned from the project.

The purpose of the individual report is to facilitate fair grading and to allow the instructor to understand well what you learned from the project. Submit a soft copy on the course canvas.

# References


[1] Golnoosh Farnadi, Susana Zoghbi, Marie-Francine Moens, and Martine De Cock. Recognising personality traits using Facebook status updates. In *Proceedings of WCPR13 (Workshop on Computational Personality Recognition) at ICWSM13 (7th International AAAI Conference on Weblogs and Social Media)*, pages 14–18, 2013.

[2] Y. R. Tausczik and J.W. Pennebaker. The psychological meaning of words: LIWC and computerized text analysis methods. *Journal of Language and Social Psychology*, 29(2454), 2010.