ירדן רותם 305284515, נועם מוסקוביץ' 308353341.

דוח עבודה 4 – מדעי הנתונים

מבנה העבודה: את העבודה חילקנו ל-2 מחלקות:

מחלקת model: מחלקה זו מבצעת את כלל החישובים, הכנות הנתונים, בניית המודל והרצתו וכלל, כל מה שלא עוסק בהצגת הGUI.

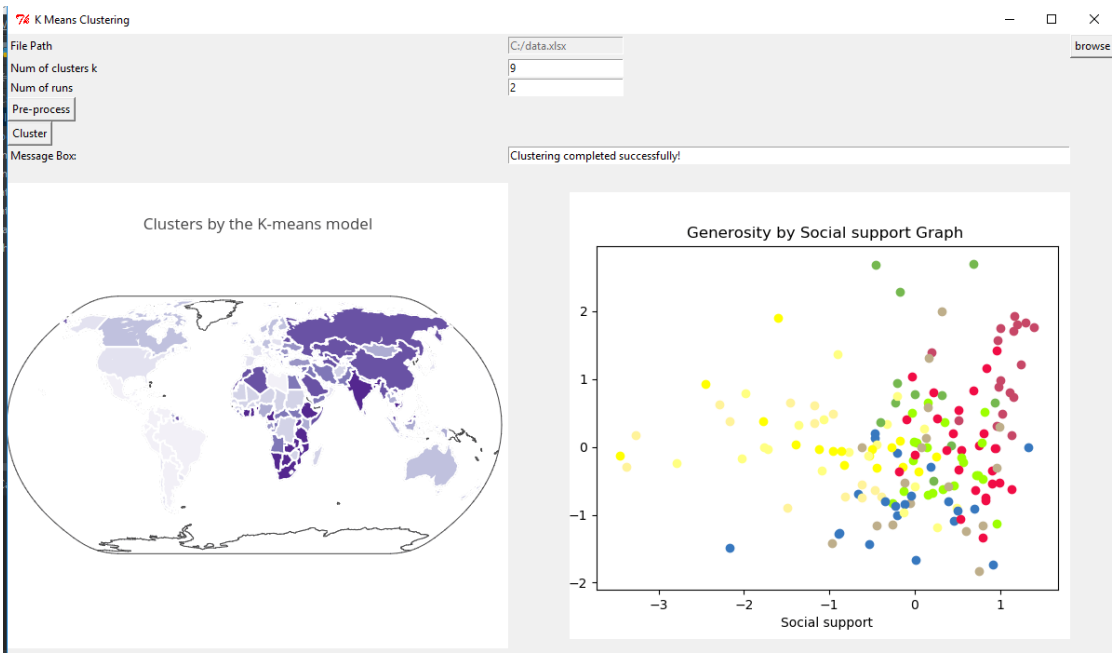
פרמטרים:

- Path: מכיל את הנתיב לקובץ הDATA.
- :Clusters מייצג את מספר הקלסטרים על פי דרישת המשתמש.
- Runs: מייצג את מספר הריצות על פי דרישת המשתמש.
- Df: שומר את הנתונים בצורת Dataframe מהחבילה pandas.

פונקציות:

- initBeforePrepare: מאתחל את כל המשתנים הלוקלים לפני הכנה הנתונים.
- Setים: מגדיר את הפרמטר בהתאם לשמו בכותרת.
- ReadCsv: קורא את קובץ הDATA על פי הנתיב אותו מקבלת בחתימה.
- PrepareData: מריץ את כלל הכנת המידע ומבצע את השלבים: קריאת המידע, מילוי ערכים חסרים, סטנדרטיזציה, קיבוץ הנתונים.
- Fill: ממלא את הנתונים החסרים בערך הממוצע של אותה עמודה.
- Standart: מבצע סטנרטיזציה לנתונים ע"פ: ערך – ממוצע / סטיית תקן.
- Grouping: מקבץ את כל הנתונים כך שכל רשומה תכיל את שם המדינה ותמצע את שאר הפרמטרים.
- KmeansModel: מריץ את מודל הKmeans תוך שימוש בספריה sklearn. את הערכים להרצה נתנו לפי מה שהמשתמש הזין בGUI ואיתחלנו דרך הפונקציה initBeforePrepare. לאחר מכן מדפיסים את תוצאות המודל ב-2 הצורות שנתבקשנו להדפיס.
- PrintByScatter: מדפיס את תוצאות המדגכם כאשר ציר הX: Social support וציר הY הוא Generosity. לאחר מכן במקום להדפיס את הגרף אנחנו שומרים אותו מקומית על המחשב כדי להציג אותו בGUI. השתמשנו כמובן בscatter של Plt.
- PrintMap: על מנת לבנות המפה של מדינות השתמשנו בchoropleth, לאחר מכן שמרתי גם כאן את התמונה על המחשב על מנת להציג אותה בGUI לאחר מכן. כמו כן, המפה נפתחת למשתמש באינטרנט אם ירצה להסתכל עליה בצורה אינטרקטיבית.

<u>מחלקת GUI:</u> מחלקה זו מקבלת את כל החישובים מהModel ומציגה אותם בחלון תוך שימוש בTKinter. כך נראית התוצאה הסופית:



<u>פרמטרים:</u>

- Path: מכיל את הנתיב לקובץ הDATA.
- Clusters: מייצג את מספר הקלסטרים על פי דרישת המשתמש.
- Runs: מייצג את מספר הריצות על פי דרישת המשתמש.
- isPrepareDone: בודק האם בוצעה הכנת נתונים לפני שמאפשר לעשות cluster.
- M – מכיל את המודל דרכו אנו מריצים את כלל ניתוח הנתונים.

<u>פונקציות:</u>

- init: הפונקציה שמגדירה את מבנה הממשק. כמו שניתן לראות בהערות של העבודה אנחנו עובדים בשיטת הgrid. השיטה בעצם מחלקת את חלון התצוגה לשורות ועמודות וכך מאפשרת לנו ליצור את התוכנה שלנו בצורה מסודרת ונוחה. אנחנו עוברים שורה אחר שורה ומגדירים מי יהיו הרכיבים אשר מרכיבים אותה, כמו שניתן לראות כאן:

```
#first row of the GUI
self.label_path = Label(master, text="File Path")
self.entry_path = Entry(master, state=DISABLED)
self.browse_button = Button(master, text="browse", command=lambda: self.browsefile(m))

#second row of the GUI
self.label_clusters = Label(master, text="Num of clusters k")
self.entry_clusters = Entry(master, validate="key", validatecommand=(vcmd, '%P'))
```

כמה הסברים לדוגמא: בentry_path הstate הוא DISABLED כדי שהמשתמש יבחר את הנתיב דרך הbrowse. כמו כן, ניתן לראות בentry_clusters יש validatecommand שמוודא שהמשתמש מכניס רק מספרים.
לאחר שסיימנו להגדיר את הפרמטרים עצמם שירכיבו את השורות, נבנה את מבנה ועיצוב התוכנה ע"י שימוש בgrid.

```
# LAYOUT
#Positions in the first row
self.label_path.grid(row=0, column=0,columnspan=2,sticky=W)
self.entry_path.grid(row=0, column=1, columnspan=10,sticky=W)
self.browse_button.grid(row=0, column=2,sticky=E)

#Positions in the second row
self.label_clusters.grid(row=1, column=0, columnspan=2,sticky=W)
self.entry_clusters.grid(row=1, column=1,sticky=W)
```

כמו שניתן לראות, נגדיר שורה ועמודה לכל אחד מהמאפיינים, כאשר אם נוסיף Sticky, אותו פרמטר יוצמד לכיוון אותו ציינו (W-west וכן הלאה).

- PrintToMessageBox: מתחת לכפתור cluster הוספתי מין תיבת טקסט כזאת שתדפיס הודעות למשתמש פרק לdialog. פונקציה זו מקבלת את הטקסט ומדפיסה בתיבת טקסט את השורה המתאימה.
- ShowDialog: כמו כן הוא, מציג את הדיאלוג.
- ShowErrorDialog: מציג דיאולוג שגיאה.
- Prepare_data: מקבל את המודל, ומבצע הכנת הנתונים. תחילה בודק האם בסיס הנתונים הוא לא ריק, לאחר מכן בודק כי ערך הקלאסטר הוא תקין (מה הוא ערך תקין? ערך גדול מ1, וערך שלא עולה על מספר הרשומות. לא יתכן שנרצה לקבל 6 קלאסטרים על 5 רשומות – כפי שכתוב בדוקומנטציה).
- Browsefile: מקבל את המודל, קורא את הנתיב כפי שהמשתמש בחר בשיטת הbrowse ולאחר מכן קורא את המידע.
- Cluster: תחילה בודק האם בוצעה הכנת נתונים (על אף שלא ניתן ללחוץ על הכפתור ללא שבוצעה הכנת נתונים) במידה ולא בוצעה, מדפיס דיאלוג שגיאה. לאחר מכן בודק שכל המשתנים אותחלו כמו שצריך, במידה ולא מדפיס הודעת שגיאה. במידה והכול הלך "חלק" מריץ את מודל הK means. בשלב זה התמונות שמורות בקובץ png. על מנת להציג אותן בדרך שאני מצאתי התמונות חייבות להיות בפורמט gif, לכן אני ממיר את הקבצים לgif ומציג אותם למסך. גם כאן, קודם מגדיר את הפרמטר עצמו ואחרי זה מוסיף אותו לgrid. בגמר מציג הודעות מתאימות עם דיאלוג ובתיבת טקסט שיצרתי.
- ConvertToGif: כמו שציינתי קודם יש צורך להציג את התמונות וניתן לעשות זאת כאשר פורמט התמונה הוא gif, לכן יצרתי פונקציה שממירה את התמונה הנוכחית מpng לgif.
- Validate: מוודא שמה שמוכנס לentry של cluster וrunsו הוא ערך נומרי בלבד.