**דקשתרגיל 2 – התנהגות לקוחות**

זמן הגשה – עד לסוף הסמסטר

בתרגיל זה עליכם לנתח את התנהגות הלקוחות של ספק מורצים מליסבון, פורטוגל. שימו לב שאת חלקו הראשון של התרגיל ביצענו בכיתה – ועליכם למעשה לנסות לשחזר אותו לבד על פי ההנחיות המופיעות מטה. בנוסף יש כאן כמובן חלקים שלא ביצענו בכיתה.

התרגיל בנוי באופן שעליכם לבצע שלב אחרי שלב. אין לדלג על שלבים. בסיום עליכם להעלות את קובץ המחברת (*בשם המכיל את הת.ז. שלכם , עם Ex1\_2, וסיומת ipynb*) לתיבה שהוגדרה לשם כל ב Moodle.

1. ראשית עליכם לפתוח את סביבת Jupyter.
2. נוריד את קובץ הנתונים מהאתר של אוניברסיטת קליפורניה בכתובת <https://archive.ics.uci.edu/ml/datasets/wholesale+customers> (או מהקובץ שנמצא במודל – זה אותו הקובץ)
   1. עליכם לחפש שם מאגר שנקרא "Wholesale custemers", להיכנס אליו ותחת Data Folder להוריד את קובץ ה csv.
   2. פתחו את קובץ ה csv ותראו שם שדות שונים, השתמשו בנתונים שנמצאים באתר על מנת **להסביר במחברת** מה המשמעות של כל אחד מהשדות
3. נוריד את הספריהvisuals מהמודל ונוודא שהקבצים שלה נמצאים באותה ספרייה של המחברת שלנו
4. עכשיו שאנחנו מכירים את הקובץ קצת יותר טוב, קדימה לעבודה:
   1. נתחיל בסדרה של הצהרות על כפי שעשינו בכיתה:

ב numpy שנשתמש בה עבור חישובים מתמטיים בסיסיים, בעיקר מעל ייצוג ווקטורי (אוסף של ערכים):

**import** **numpy** **as** **np**

נמשיך עם pandas שכפי שלמדנו בכיתה, מאפשרת עבודה עם טבלאות בפייתון – באופן שמזכיר את אקסל:

**import** **pandas** **as** **pd**

נרצה גם להשתמש בפונקציית display המציגה את הנתונים בצורה יותר יפה מעל המחברת:

**from** IPython.display **import** display

כעת נייבא את ספריית ה visuals שנועדה להציג מספר גרפים ספציפיים לתרגיל זה:

**import** visuals as vs

ולגבי Matplotlib נרצה שהתוצרים (גרפים) יודפסו ויישמרו במחברת עצמה:

%matplotlib inline

* 1. כעת נרצה לטעות את קובץ ה csv שהורדנו. לשם כך נשתמש בפקודה read\_csv של pandas. תוכלו לקרוא עליה כאן:

<http://pandas.pydata.org/pandas-docs/version/0.23/generated/pandas.read_csv.html>

או בדוגמא קצת יותר ספציפית כאן:

<https://chrisalbon.com/python/data_wrangling/pandas_dataframe_importing_csv/>

שמרו את התוכן של קובץ ה csv במשתנה בשם **data**

* 1. כעת, אחרי שוידאתם שהנתונים של קובץ ה csv נמצאים במשתנה data, נראה להיפטר מהשדות Region ו Channel שבקובץ המקורי. אין לשנות את הקובץ עצמו אלא לבצע זאת באמצעות הפקודה drop של pandas. דוגמא פשוטה לכך תוכלו למצוא כאן:

<https://chrisalbon.com/python/data_wrangling/pandas_dropping_column_and_rows/>

* 1. כעת יש בידינו את המשתנה (מסוג dataframe) בשם data המכיל את ה csv בלי העמודות Region, Channel. הדפיסו במחברת את 10 השורות הראשונות של המשתנה data
  2. כעת נרצה לקבל סטטיסטיקות על הנתונים במשתנה data באמצעות הפקודה describe. תוכלו לקרוא עליה כאן: <http://www.datasciencemadesimple.com/descriptive-summary-statistics-python-pandas/>
  3. כתוצאה מהסעיף האחרון - כתבו במחברת, מהו טווח הערכים (בין המינימום למקסימום) של השדה Milk? בנוסף, מהו האחוזון ה 75 של השדה Fresh? בנוסף – האם הנתונים של Frozen מתפלגים התפלגות נורמאלית? בנוסף – נניח (הגם שזה לא נכון...) שהנתונים של Milk מתפלגים נורמאלית, מה זה אומר לגבי הערך של סטיית התקן ? מה המשמעות שלו?
  4. כעת נבחר לנו 3 אינדקסים בטבלה אשר יייצגו בעבורנו לקוחות ספציפיים.

תארו מה ביצענו בפקודה הבאה:

1. indices = [47,138,359]
2. samples = pd.DataFrame(data.loc[indices], columns = data.keys()).reset\_index(drop = True)
3. display(samples)
   1. כעת נשתמש בספריית seaborn על מנת להמחיש ויזואלית (באמצעות מפת חום) את האחוזונים של ערכי כל אחד מהמשתנים של 3 הלקוחות שבחרנו
4. **import** seaborn as sns
5. percentiles = data.rank(pct=True) # rank by percentile
6. percentiles =
7. percentiles = percentiles.iloc[indices] # modify the index
8. sns.heatmap(percentiles, vmin=1, vmax=99, annot=True)
9. display(percentiles)

שימו לב שהקוד כאן חלקי. בשורה 7 עליכם להכפיל את ערכי האחוזונים ב 100 (במקור הם מגיעים בין 0 ל 1, ולוודא שיש רק ספרה אחת אחרי הנקודה. נסו למצוא לבד איך עושים את זה בפייתון באצמעות חיפוש בגוגל.

כעת כשרואים את מפת החום, ממבט על על הצבעים, מה ניתן לומר לגבי על אחד מהלקוחות שבחרנו?

i . כעת ננסה לבחון אם קיימות תכונות אשר ניתן לוותר עליהן במהלך הנסיון שלנו לבצע סגמנטציה ללקוחות. תכונות שניתן לוותר עליהם הן תכונות שלא מוסיפות לנו ידע לגבי ההתנהגות של הלקוח. למשל – נניח שבנוסף לתכונה milk היתה לנו תכונה non-milk, אז ברור שאחת מהן מיותרת כיוון שיש ביניהן קורלציה. אחת מהדרכים לבחון האם קיימת קורלציה בין מספר שדות נתונים לשדה מסוים, היא באמצעות הפעלת רגרסיה בין השדות האלו לשדה הספציפי שרוצים לבחון. נבצע זאת באמצעות הספריה DecisionTreeRegressor.

למשל אפשר להשתמש בקוד הבא, המוצא רגרסיה בין השדות של קובץ בשם boston לבין שדה ספציפי הנקרא medv. אצלנו כמובן מדובר בשדות שונים – אז עליכם להתאים את הקוד בהתאם:

boston\_df = pd.read\_csv('Boston.csv')

X = boston\_df.drop('medv', axis = 1)

y = boston\_df.medv

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, train\_size = 0.75, random\_state = 0)

regr\_tree\_boston = DecisionTreeRegressor(random\_state=0)

regr\_tree\_boston.fit(X\_train, y\_train)

score= regr\_tree\_boston.score(X\_test,t\_test)

לאחר שהתאמתם את הקוד לנתונים של הבעיה שלנו, עליכם להדפיס את score.

עליכם להריץ את הבדיקה הזאת בכל פעם על שדה אחר, כלומר בפעם הראשונה על Fresh, בשניה על Milk וכך הלאה – בכל פעם לבדוק את הרגרסיה בין יתר השדות לשדה הספציפי. אנא כתבו במחברת את התוצאות שיצאו לגבי כל הבדיקות שביצעתם.

j. בסעיף הקודם בדקנו רגרסיה בין כל התכונות (למעט התכונה שרצינו לבדוק) לבין התכונה שרצינו לבדוק. דרך נוספת – פשוטה יותר - לוודא קשר בין זוגות של תכונות היא באמצעות הצגת גרפים המראים בכל פעם זוג תכונות. למשל גרף של Milk בציר X ו Fresh בציר Y, וכן הלאה. נרצה לראות את כל האפשרויות של גרפים כאלו על בסיס התכונות של הנתונים שלנו – ונוכל לבצע את זה באמצעות הפקודה scatter\_matrix. תוכלו לראות דוגמא לשימוש בה כאן:

<https://pandas.pydata.org/pandas-docs/stable/visualization.html>

עליכם להציג scatter\_matrix של כל התכונות במשתנה data שלנו. בנוסף להצגת הגרפים, כתבו בין אילו תכונות נראה שיש קשר של רגרסיה.

k. שימו לב שבגרפים שקיבלתם בסעיף הקודם וכמו כן גם בעקומות שנמצאות על האלכסון הראשי, רואים בבירור שהנתונים נוטים לצד השמאלי – ובהחלט אינם מתפלגים התפלגות נורמאלית. בואו נבדוק למשל אם תכונה מסויימת מתפלגת נורמאלית על ידי השוואה שלה להתפלגות הנורמאלית (כפי שעשינו בכיתה – עם QQ PLOT – המבוססת על השוואת האחוזונים – ה Percentiles של הנתונים שלנו כנגד האחוזונים של ההתפלגות הנורמאלית).

הקוד הבא מראה איך לבצע QQ PLOT על התכונה Milk:

import pylab

import scipy.stats as stats

measurements = data['Milk']

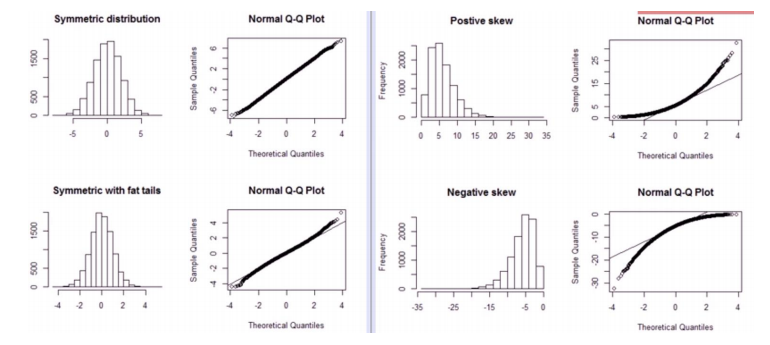
stats.probplot(measurements, dist="norm", plot=pylab)

# QQ PLOT of original data

pylab.show()

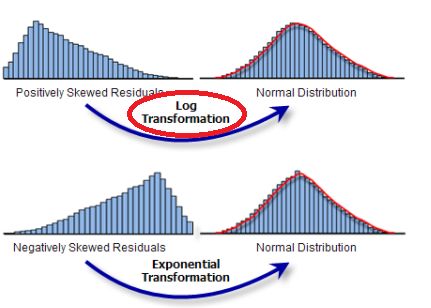
קיבלנו QQ\_PLOT של Milk. שימו לב לעקומה הכחולה המייצגת את Milk לעומת האדומה המייצגת את ההתפלגות הנורמאלית.

השוו את מה שקיבלתם לארבעת המצבים (התיאורטיים) הבאים:



המצב הכי דומה למצב שקיבלנו הוא של Positive Skew.

על מנת לבצע תיקון כך שהנתונים שלנו יתפלגו כמו בהתפלגות נורמאלית – עלינן לבצע טרנספורציה על הנתונים שלנו. אחת השיטות המקובלות נקראת Box Cox transformation. למקרה שלנו שבו יש Positive Skew אנחנו צריכים לבצע Log Transformation:



שזה פשוט אומר שצריכים להעביר את הנתונים שלנו דרך Log של הערכים המקוריים שלהם.

ניתן לבצע זאת כך (על הדוגמא שלנו שמתייחסת לשדה milk):

log\_measurements = np.log(measurements)

כעת השלימו כמו קודם – והציגו את ה QQPLOT של log\_measurements – וודאו שכעת אין לנו סטייה משמעותית מההתפלגות הנורמאלית.

l. כעת עליכם להפעיל את פונקצית ה log על כל הנתונים של data (קראו למשתנה החדש log\_data) והציגו את ה scatter\_matrix (כמו בסעיף j) של log\_data. וודאו שהפעם הנתונים מתפלגים יותר נורמאלי מקודם.

m. אם תחזרו רגע לסעיפים e,f – תוכלו ללמוד שבחלק מהתכונות שאנו מודדים, קיים הבדל מהותי בין הערך של התכונה באחוזון ה 75 לבין המקסימום. למעשה אפשר גם לבדוק את ההבדל בין האחוזון ה 90 ולראות שהוא רחוק מהמקסימום. על מנת שערכי הקיצון שאינם מייצגים את עיקר הנתונים שלנו לא ישפיעו עליהם – עלינו להסירם.

נעשה זאת האמצעות שיטת תוקי (Tukey) שתוכלו לקרוא עליה כאן: <http://datapigtechnologies.com/blog/index.php/highlighting-outliers-in-your-data-with-the-tukey-method/>

בגדול השיטה אומרת שעלינו לחשב את ה IQR – שהוא בעצם הפרש הערכים בין האחוזון ה 23 לאחוזון ה 75 בתכונה שאותה אנו מודדים (למשל Milk) – ואז כל נתון שגדול מהאחוזון ה 75 ועוד ה 1.5 IQR או קטן מהאחוזון ה 25 פחות 1.5 IQR הוא למעשה Outlier- כלומר ערך שאנחנו רוצים להיפטר ממנו.

ראשית נדפיס את כל ה Outliers:

for feature in log\_data.keys():

Q1 = np.percentile(log\_data

Q3 = np.percentile(log\_data[feature],75)

step = (Q3-Q1)\*1.5

print ("Data points considered outliers for the feature '{}':".format(feature))

display(log\_data[~((log\_data[feature] >= Q1 - step) & (log\_data[feature] <= Q3 + step))])

כעת מכיוון שאנחנו "שמרנים" ☺ אנחנו נזרוק אך ורק outliers שמופיעים ביותר מתכונה אחת.

בידקו מיהם ה outliers שמופיעים ביותר מתכונה אחת ושימו את האינקסים שלהם במערך בשם outliers – והדפיסו את המערך.

כעת ניצור data frame המכיל את הנתונים שלנו המקוריים לאחר טרנספורמציית log, פחות ה outliers:

good\_data = log\_data.drop(log\_data.index[outliers]).reset\_index(drop = True)

n. הגיע הזמן לבצע את הסגמנטציה עצמה. על מנת לבצע סגמנטציה (clustering) נשמתמש באלגוריתם בשם GMM. למתעניינים באיך הוא עובד, ניתן לקבל פרטים כאן: <http://www.statisticshowto.com/gaussian-mixture-model/> (אין חובה לצורך ביצוע התרגיל)

בגדול, האלגוריתם יחלק לנו את הנתונים למספר חלקים שנגדיר לו (נקרא כמו קומפוננטות) על פי הערכים של התכונות בנתונים שלנו – כלומר הוא יחפש קבוצות של לקוחות הדומים אחד לשני מבחינת הצריכה של סוגי המוצרים השונים. כיצד נדע לכמה חלקים עדיף לחלק? פשוט ננסה כמה אפשרויות... נתחיל ב 2 קומפוננטות, נעבור ל 3 וכך הלאה – לפחות עד 5. אבל איך נדע מי מהקומפוננטות הכי טובה? בשביל זה יש מבחן שנקרא Silhouette score – המתייחס למרחקים בין הקומפוננטות (בין הקלאסטרים) ולמרחק בין כל נתון (לקוח במקרה שלנו) לקלאסטר שלו.

נתחיל בייבוא של הספריות של אלגוריתם ה GMM ושל ה Silhouette:

from sklearn.mixture import GMM

from sklearn.metrics import silhouette\_score

כעת נפעיל את האלגוריתם על בסיס 2 קומפוננטות:

clusterer = GMM(n\_components=2).fit(good\_data)

נבצע פרדיקציה לאיזו קומפוננטה מתאימה לכל אחד מהלקוחות שלנו:

preds = clusterer.predict(good\_data)

למעשה המשתנה preds מכיל את המיפוי של כל לקוח לקלאסטר הכי מתאים עבורו.

ולבסוף נחשב את ה silhouette score:

score = silhouette\_score(good\_data,preds)

כעת עליכם לבצע זאת עבור מספר קומפוננטות בין 2 ל 5. להציג את התוצאות של כל אחד מהניסויים ולכתוב מה מספר הקומפוננטות הטוב ביותר עבור הבעיה שלנו (בעל ה score הגבוה ביותר).

* 1. אם הגעתם לכאן, מזל טוב! כל שעליכם לעשות הוא להעלות את המחברת לתיבה שנפתחה לשם כך במודל.

**בהצלחה!**