Introduction To Machine Learning

פרויקט סופי -דו"ח מסכם:

מגישות:

לילך כהן, ת"ז: 209492487

נועה לוי, ת"ז: 209226687

מספר קבוצה: **24**

חלק ראשון: Data understanding & Exploration הפנייה לנספחים נעשית כך: (נ.1) = תמונה 1 בנספחים

כדי להתחיל את התהליך, טענו את קובץ ה- train ובדקנו איך הדאטה נראה, הסתכלנו על נתונים סטטיסטיים וקיבלנו תמונת מצב ראשונית, ובדקנו איזה סוגי פיצ'רים הדאטה מכיל. זיהינו 3 סוגי פינצ'רים:

- פיצ'רים קטגוריאלים- בתוכם פיצ'רים עם מספר רב של ערכים מסוג 'file_type_trid' ('object' ו- 'c').
- פיצ'רים בינאריים- קטגוריאליים המכילים ערכים 0 או 1 (למשל 'has_debug','has_relocations')
- פיצ'רים נומריים- פיצ'רים עם ערכים רציפים. (למשל 'size', 'vsize')

לכל קבוצת פיצ'רים התאמנו את השלבים בניתוח הראשוני של הדאטה ובעיבוד המקדים. ולמען הנוחות נאזרכת כל קבוצה בשמות שהוגדרו כאן.

בנוסף הסרנו את העמודה '256sha' שמכילה מזהה ייחודי לכל קובץ, כלומר ערך אחר שונה בכל שורה. בשלב האימון היא לא רלוונטית.

<u>זיהוי התנהגות קורלטיבית בין פיצ'רים-</u> יצרנו 3 מטריצות קורלציה

1. מטריצת קורלציה כוללת על ה- train סט - שמנו לב שיש קורלציה גבוהה בין המשתנים 'size', 'numstrings' ובין 'size','MZ' ובין 'numstrings' ל 'MZ'. מכאן הסקנו שאולי בהמשך נוכל להוריד חלק מהמשתנים כדי להפחית מימדיות מבלי לפגוע בתחזית. (נ.1).
2. שתי מטריצות קורלציה נוספות כאשר הנתונים מופרדים לפי הlabel: אחת על ה- train כאשר ה- label = 1 , ואחת נוספת בה הlabel = 0 (נ. 3 ואז 2 בהתאמה).

ראינו שכאשר label = 0 , הקורלציות בין הפיצ'רים גבוהות משמעותית מאשר ב- label =1 . כלומר יש קורלציה גבוהה יותר בין הפיצ'רים כאשר הקובץ הוא לא זדוני.

<u>התפלגות של הפיצ'רים הבינאריים בכל label-</u>בדקנו בכל משתנה בינארי איך הוא מתפלג (נ.8-4) לכל label ומצאנו שלכל המשתנים הבינארים, חוץ מ 'has_relocations' (נ.5), יש הבדל משמעותי בין ה- labels. המשמעות של ההבדל הגדול יכולה להיות שאלו פיצ'רים איכותיים עבור התחזית. זיהינו מספר פיצ'רים כאלה-ביניהם:.has_resources, has_tls

<u>התפלגות של הפיצ'רים הנומריים בכל - label</u> (נ.19-9) השתמשנו ב- boxplot כדי לבחון את הפיצ'רים ובכך הנחנו שהם מתפלגים נורמלית, אף שאנחנו לא יודעות זאת בשלב זה. בחינה של הפלוטים מאפשרת לנו להסיק מספר מסקנות. ניתן לראות כי יש הבדל בהתפלגות בין כל label בפיצ'רים avleangth , MZ, לעומת זאת בפיצ'ר A ניתן לראות כי אין הבדל משמעותי. בנוסף ניתן לראות שרובם מכילים כמות גדולה של outliers.

חלק שני- עיבוד מקדים – preprocessing

<u>עבור כל קבוצה של משתנים ביצענו את התהליכים הבאים:</u>

<u>טיפול בערכים חסרים: None</u> בפיצ'רים הקגטוריאלייmfile_type_trid ו-C החלפנו None במחרוזת 'NONE_VAL'. בפיצרים הקטגוריאליים הבינאריים החלפנו את הערכים החסרים ב- 0. בפיצ'רים הנומריים החלפנו ערכים חסרים בחציון.

<u>התמודדות עם outliers:</u> עדיף להסיר ערכים חריגים, ובכך אומנם נעלה מעט את ההטייה (bias) אך ה- variance ירד בצורה משמעותית ונקבל דאטה שניתן לעבוד איתו בצורה יותר נכונה.

בפיצ'רים קטגוריאליים- ב- file_type_trid - זיהינו שיש ערכים שחוזרים מספר רב של פעמים, ויש נתונים המופיעים מספר נמוך של פעמים או פעם אחת. במקרה זה החלטנו שנשמור את הערכים השכיחים ואילו ערכים 'נדירים' נאחד לערך 'other'. בחירת מספר הערכים שנשמור נעשתה מהתבוננות ב- barplot שמציג בכמה שורות כל ערך מופיע (נ.20). בפיצ'ר זה בחרנו לשמור את 30 הערכים השכיחים ביותר, וב) 'C' -נ.21) שמרנו על 6 הערכים השכיחים ביותר.

בפיצ'רים נומריים- מהתבוננות בהתפלגות של הפיצ'רים הנומריים (נ.22) זיהינו כי ההתפלגות שלהם היא skewed distribution, התפלגות הדומה לנורמלית, אך אחד הזנבות ארוך משמעותית מהאחר. לאחר שזיהינו מאפיין זה בנינו

פונקציה להתמודדות עם outliers המיועדת להתפלגות מסוג זה באמצעות(capping & Inter-Quartile Range (IQR).
(קביעת גבול עליון ותחתון במבוסס על ההתפלגות של הפיצ'ר (אחוז 75 ואחוז 25) ושינוי הערך בחריגים החורגים מהגבול לערך הגבול.)

לאחר שהשתמשנו בשיטה זו על כל הפיצ'רים ראינו שהיא הייתה אגרסיבית מדי עבור חלק מהם, והם איבדו הרבה מידע. לכן השתמשנו בשיטה נוספת להתמודדות עם outliers - precentile method. כך חילקנו את הפיצ'רים הנומריים לשלוש קבוצות ובכל אחת עשינו הסרת outliers מותאמת:

עבור תת-קבוצה 1numeric אנו משתמשים ב-capping & Inter-Quartile Range - IQR המתאים להתפלגות המוטה.

עבור תתי קבוצות 2numeric ו- 3numeric אנו משתמשים בשיטת percentile, כאשר לכל קבוצה הזנו אחוז שונה. (שיטת האחוזון היא קביעת אחוזון שווה בשני הצדדים כדי לזהות חריגים, והחלפתם בערך הגבול.)

בנספחים ניתן לראות boxplot ו- kde של כל אחד מהפיצ'רים הנומריים לפני ואחרי הטיפול בערכים חריגים. לפני (נ.37-26) אחרי (נ.53-33) ב(נ.38) ניתן לראות את ההיסטוגרמות של כולם אחרי.

<u>הערכת איכות פיצ'רים ויצירת פיצ'רים חדשים והקטנת מימדים:</u>

פיצ'רים קטגוריאלים- בפיצ'רים קטגוריאלים המכילים ערכים שאינם מספריים (מילים) היינו צריכות להחליט על שיטה לקידוד (encoding) המשתנים-

השיטה הראשונה שניסינו היא יצירת עמודות dummies, בפועל שיטה שיוצרת עמודות כמספר הערכים השונים בפיצ'ר. כפי שציינו קודם בעמודות האלה (file_type_trid, C) יש מספר רב של ערכים שונים, גם לאחר ההתמודדות עם ה-outliers.לכן שימוש בה מוסיף כ- 35 עמודות ומגדיל משמעותית את מימדיות הדאטה ופוגע בתחזית. לכן עברנו לשימוש בשתי השיטות האחרות.

השיטה השנייה לקידוד הקטגוריאלים היא פשוט להיפטר מהם! ניסינו גם שיטה זו, (הדאטה לאחר הסרת הקטגוריאלים נקרא X_train_drop_cat , X_test_drop_cat), בהמשך נתייחס לתוצאות המודלים בהרצות עם דאטהסט שלא מכיל את המשתנים הקטגוריאליים.

השיטה השלישית שניסינו היא ordinal encoding. בשיטה זו מחליפים את הערכים המילוליים בערכים מספריים. מתוך הבנה שבהחלפה של הערכים המספריים אנחנו יוצרות יחס סדר בין הערכים השונים, בחרנו להחליף כל ערך במספר הפעמים שהוא מופיע בסט ה- train. כך ערכים שכיחים יותר יקבלו מספרים גדולים יותר. ויחס הסדר שנוצר מבוסס על שכיחות הערך ומספר משהו על הדאטה המקורי.

בנוסף בנינו פיצ'ר חדש 'interactions_file_type_trid_C' המכיל את השילובים השונים בין ערכי שני הפיצ'רים. גם אותו קודדנו. בהמשך השתמשנו במבחן chi square על המשתנים הקטגוריאלים כדי להבין את האיכות של כל אחד מהם לתחזית, וראינו כי C_count_encoded ו- interactions_file_type_trid_C_count_encoded משמעותית פחות טובים מהאחרים ולכן ויתרנו עליו. בנוסף, יצרנו פונקציה הבוחרת את 10 המשתנים הנומריים הטובים ביותר בהתבסס על מבחן ANOVA F-value. בעקבות השימוש בפונקציה הוסרו ['A', 'imports', 'printables' , 'numstrings'] באמצעות הפונקציה selectkbest של sklearn עם מבחן anova f-value.

פיצ'רים בינאריים - לאחר מספר ניסיונות הגענו למסקנה שהסרה של פיצ'ר מסוג זה פוגעת בתחזית.

עשינו סטנדרטיזציה על שני הסטים באמצעות StandardScaler של sklearn . החשיבות של שלב זה היא עבור הרגרסיה הלוגיסטית והPCA בהם לא נרצה שהבדל בסדרי הגודל של הערכים ישפיעו על התוצאות.

בסופו של דבר הגענו לשלב הרצת המודלים עם 4 סטים של דאטה:

- דאטה שמכיל רק את הפיצ'רים הבינאריים והנומריים, שגם עברו סינון ע"י selectkbest עם מבחן anova f-value
- דאטה שמכיל את כל סוגי הפיצ'רים לאחר שהקטגוריאלים והנומריים עברו סינון לפי השיטות שתיארנו.

- דאטה שמכיל את כל הפיצ'רים ללא סינון.
- וסט נוסף שמכיל את כל הנתונים ללא סינון ועם pca.

מימדיות גדולה בעייתית מכיוון שהיא מובילה לשונות גדולה(variance) וזה יגרום ל over fitting שיפגע בתחזית על דאטה חדש (test!). בנוסף כאשר יש הרבה מימדים, התצפיות יותר קשות לסיווג כי כל התצפיות נהיות במרחק שווה מהאחרות ואז לא ניתן לסווג בצורה שמציגה את המציאות. יש מספר סימנים מעידים למימדיות גבוהה: זמן ריצה גבוה(אך יכול להיגרם גם בגלל אלגוריתמים מורכבים או כי יש הרבה דאטה), אם קשה למצוא דפוסים ויחסים משמעותיים בנתונים(כמו שהסברנו לעיל שהמרחק בין הנקודות נהיה גדול וזהה ולכן יותר קשה לסווג למחלקות), אם התחזית לא טובה(מימדיות גדולה עלולה להביא לover fitting והמודל שהתאמן על סט הדאטה ילמד גם את הרעש ולכן יביא ביצועים לא טובים בסט הבדיקה). ההשפעה על המודל מפורטת בהמשך.

<u>חלק שלישי: בחירת היפר פרמטרים והרצת מודלים</u>.(בנספח 78 יש הסבר על bayes שניסינו ובסוף לא השתמשנו בו.)

בכל המודלים למעט knn השתמשנו בGridsearchCV, הסבר על השימוש הספציפי בכל מודל נמצא בנספחים *77*. נתאר את ההיפר פרמטרים שבחרנו עבור כל מודל:

<u>מודל רגרסיה לוגיסטית:</u>

<u>:'penalty'</u> הוספנו למודל כפרמטר את 'penalty' כדי לבצע רגולריזציה על מנת למנוע\ לצמצם overfitting. בדרך זו <u>הוספנו מעט bias (הטייה)</u>למודל על מנת לצמצם משמעותית את השונות. בחרנו להשתמש בl2 הידוע גם כ'ridge' וbl1 הידוע גם כ 'lasso' . בחרנו לא להוסיף את '***elasticnet***' מכיוון שהוא ממילא מוסיף גם את l1 וגם את l2, אך יש solvers שלא עובדים איתו.

<u>:'c'</u> הוספנו למודל את הC הבאים: [10,100,200,300,400] כאשר כבל ש C גדל, הרגולריזציה פחות משמעותית. כלומר כבל שC גדל, השונות גדלה, וההטייה קטנה. הC חשוב לרגולריזציה כי הוא מגביל את המשקולות של הרגולריזציה.

<u>:'solver'</u>השתמשנו ב 'liblinear' וב'lbfgs'מכיוון ש'liblinear' תומך בl1 וב(l2lasso&ridge),אך רצינו גם לשמור את ברירת שמחדל שהואlbfgs' 'אבל ה'liblinear' .

לרגרסיה לוגיסטית יש עוד המון היפר פרמטרים שניתן להוסיף, אך משיקולי זמן ריצה בחרנו רק את אלה.

הגיע לתוצאה הטובה ביותר עם דאטה סט ללא משתנים קטגוריאלים ועם בחירה של המשתנים הטובים ביותר מבין הבינאריים.

<u>מודל **KNN:**</u>

'n_neighbors' -מספר השכנים הקרובים ביותר. בחנו השפעה של מס' k שונים ובחרנו את הטוב ביותר.בשאר ההיפר פרמטרים האפשריים במודל זה לקחנו את ה- defult שעבדה היטב. כבל שk- זה גדל, כך גדלה ההטייה. עבור k גדול מדי, קשה לקבל "תמונה חדה" על הסיווג האמיתי. לעומת זאת, עם הקטנת הK השונות גדלה, ובK קטן מדי קיים סיכוי ל overfitting(ראה דוגמא k=1)

הגיע לתוצאה הטובה ביותר עם דאטה סט בו בחרנו את המשתנים הטובים ביותר.

<u>מודל **multi layer perception:**</u> משיקולי זמן ריצה ויעילות בבחירת ההיפרפרמטרים, בחרנו בבאים:

'activation' מייצג את האקטיבציה לשכבות הנסתרות(לדוגמה relu, logistic וכו').כאשר נבחר באקטיבציה לא לינארית כמו סיגמואיד נוכל לייצג את המציאות בצורה יותר מדויקת ובכך להפחית את ההטייה, ומנגד צריך להיזהר שהמודל לא מתאמן עם אקטיבציות אלו יותר מדי זמן ומגדיל את השונות ויוצר overfitting (במידה וכן, נשנה אקטיבציה\ נעניש בעזרת ההיפר פרמטר .('alpha'

''solver : בדקנו את שלושת הsolvers: 'sgd','adam','lbfgs' ובחרנו לוותר על 'lbfgs', הוא לא תרם לטיב התוצאות אך הכפיל ביותר מפי 2 את זמן הריצה.

verbose'': ברירת המחדל היא False ושינינו ל True מכיוון שרצינו לצפות בהתקדמות.

max_iter'': ברירת המחדל היא 200 ושינינו ל 1000 מכיוון שראינו הבדלים משמעותיים בתוצאות כאשר הרשנו לאלגוריתם לרוץ מספר רב יותר של איטרציות עד ההתכנסות, ועם זאת זמן האלגוריתם היה סביר. בכל איטרציה המודל מתאים את הפרמטרים כדי להוריד את הtraining loss ולשפר את הביצוע. ככל ש'max_iter' גדל, כך המודל מותאם יותר לסט האימון, והטיית המודל קטנה, אך חשוב להיזהר כי לפעמים מאימון יתר יכול להיגרם overfitting (ולכן השונות גדלה מכל איטרציה).

' 'early_stopping: היפר פרמטר זה מאפשר למודל להפסיק לרוץ לפני כמות האיטרציות שנקבעו, במידה וציון ה validation לא משתפר ברמה משמעותית. בזכות בחירה זו זמני הריצה התקצרו משמעותית, ונמנע מצב של overfitting כי המודל לא ממשיך לרוץ על סט האימון ללא צורך. אמנם קיים סיכון שההטייה תגדל כי המודל עלול לא להגיע לתוצאה המיטבית, אך מבדיקה שביצענו על ריצת המודל עם ההיפר פרמטר ובלעדיו, ההבדל זניח למדי.

hidden_layer_sizes', 'alpha', 'learning_rate_init'' : השתמשנו בברירת המחדל לאחר שניסינו מספרים אחרים שהפיקו תוצאות פחות טובות והגדילו משמעותית את זמן הריצה.
'hidden_layer_sizes' מייצג את ארכיטקטורת של רשת הנוירונים, 'alpha' מייצג את כמות הרגולריזציה(כדי למנוע overfitting), 'learning_rate_init' מייצג את המספר הראשוני לקצב הלמידה.

הגיע לתוצאה הטובה ביותר עם דאטה סט בו יש את המשתנים, ועבר pca.

מודל--**Random Forest**

n_estimators' ' - הוא מספר עצי ההחלטה. ככל שיגדל, כך השונות תקטן אך ההטייה לא תשתנה ולכן נרצה שהיפר פרמטר זה יהיה כמה שיותר גדול, במגבלות הזמן ריצה האפשרי.

max_depth''- מייצג את העומק המקסימלי של העצים ב- randomforest. עומק גדול מדי עלול ליצור overfitting מכיוון שעץ עמוק מדי עלול ללמוד גם דברים שלא צריכים להיכלל כמו רעש \ outliers. עם זאת לא נרצה עומק קטן מדי, אז הסיווג יסתמך על פחות מידע ותגדל ההטייה.

critertion''- ה 'log_loss' לקחה זמן רב מדי ולא הביאה תוצאות טובות, ולכן בדקנו את האופציות האחרות. אין השפעה משמעותית על ההטייה והשונות- אלו בעיקר דרכים שונות ליצירת כל חלוקה בעץ.

הגיע לתוצאה הטובה ביותר עם דאטה סט בו יש את כל המשתנים.

חלק רביעי – הערכת מודלים:

confusion matrix : עשינו לכל אחת מהדאטה סטים שיצרנו confusion matrix *75-71*,*79שמסביר לעומק ולא רק מצרף פלט*. באופן כללי אצלנו הTP מייצג שחזינו שהקובץ זדוני והוא באמת זדוני, FP מייצג שחזינו שהקובץ זדוני אבל הוא למעשה לא זדוני, FN מייצג שחזינו שהקובץ לא זדוני אך הוא זדוני, TN מייצג שחזינו שהקובץ לא זדוני והוא באמת לא זדוני. ניתן לראות שבכל האופציות קיבלנו מספרים יחסית דומים: הTP וה TN מהווים את רוב המסה כלומר לרוב אם חזינו שמשהו הוא זדוני הוא באמת זדוני, ואם חזינו שהוא לא זדוני אז הוא בד"כ לא זדוני. ולכן בכל המדדים שבדקנו(recall, precision, accuracy) התוצאה תמיד גדולה מ0.88(הסבר על כל מדד נמצא בנספחים), ומעידה על כך שביצועי המודל טובים.

ROC AUC K FOLD-(נ.54- 76) כל מודל הערכנו בעזרת K-Fold cross validation. ובנינו פלט ROC על כל K-Fold וקיבלנו בצורה ויזואלית אישור לטיב המודל שלנו, וגם תוצאות מספריות שהראו שכל המודלים הביאו תחזית טובה, בכולם למעט הרגרסיה הלינארית עם תוצאה מעל 0.9 כלומר המודלים מסווגים בצורה טובה את מרבית הדוגמאות. המודל היחיד שמסווג בצורה מעט פחות טובה הוא מודל הרגרסיה הלוגיסטית שאת ביצועיו לא הצלחנו לשפר תחת זמן ריצה סביר, יתכן שהוא לא מתאים לדאטה שלנו.

ניתן לראות כי לכל מודל יש תוצאה טובה ביותר עבור סידור שונה של הדאטה. כך למשל knn מגיע לשיא (0.945 auc) עם דאטה סט בו בחרנו את המשתנים הקטגוריאלים והנומריים בטובים ביותר ולעומתו random forest הגיע לשיא (0.976 auc) עם דאטה סט בוא לא ביצענו בחירה של משתנים.

ניתן לראות כי בשלושת המודלים: רגרסיה לוגיסטית, knn, mlp הורדת המימדיות שיפרה את התוצאות והם הגיעו לתוצאה הטובה ביותר עם דאטה סט שעבר הורדת מימדיות. לעומת זאת random forest הגיע לתוצאה הטובה ביותר שלו כאשר לא הורדנו מימדים, שהיא גם התוצאה הכי טובה בהשוואה לכל המודלים (עם ובלי הורדת מימדים). מכאן אנחנו מסיקות שלכל מודל מתאים דרך אחרת מבחינת הורדת מימדיות ובחירת משתנים.

לא היה פערי ביצועים משמעותיים בין הtrain לvalidation - ניתן לראות זאת בפלטים של הroc curve שהפעלנו על הולידיישן, ובגלל חוסר הפערי ביצועים ניתן להניח שהמודל לא בoverfitted. הכחול מבוסס על הvalidation set, הצבעוני לוקח מהtrain מהחלק של הk-fold. (דוגמא בנספחים 81 82)

בחירת המודל הסופי- לאחר שקלול התוצאות בחרנו להשתמש ב- random forest ולסדר את הדאטה כך שהוא מכיל גם את הפיצ'רים הקטוגריאלים שקודדנו באמצעות ordinal encoding וגם את הפיצ'רים הנומריים והבינאריים ללא שום סינון של פיצ'רים. זה המודל שהביא לנו את ה- auc הגבוה ביותר וגם את הroc הטוב ביותר ולכן בחרנו בו.

ביצירת המודל השתמשנו בכלים להתמודדות עם משתנים קטגוריאלים וכלים להתמודדות עם **outliers** אותם לא למדנו בכיתה ועליהם הרחבנו בתיאור העיבוד המקדים. תודה מעומק הלב לאתר **kaggle**, התנ"ך החדש שלנו, שבלעדיו כל זה לא היה אפשרי. קישורים למאמרים ספציפיים באתר נמצאים בנספחים.

<u>התרומה של כל אחת:</u>

את כל הפרויקט עשינו בשיתוף פעולה וייעוץ תמידי אחת עם השנייה כאשר לכל אחת היה תחומי אחריות עליהן היא שמה יותר דגש. את החלק הראשון והרביעי עשינו ביחד, על החלק השני והחמישי לילך היתה אחראית, על החלק השלישי וכתיבת הדו"ח נועה היתה אחראית.

נספחים:

מקורות מידע

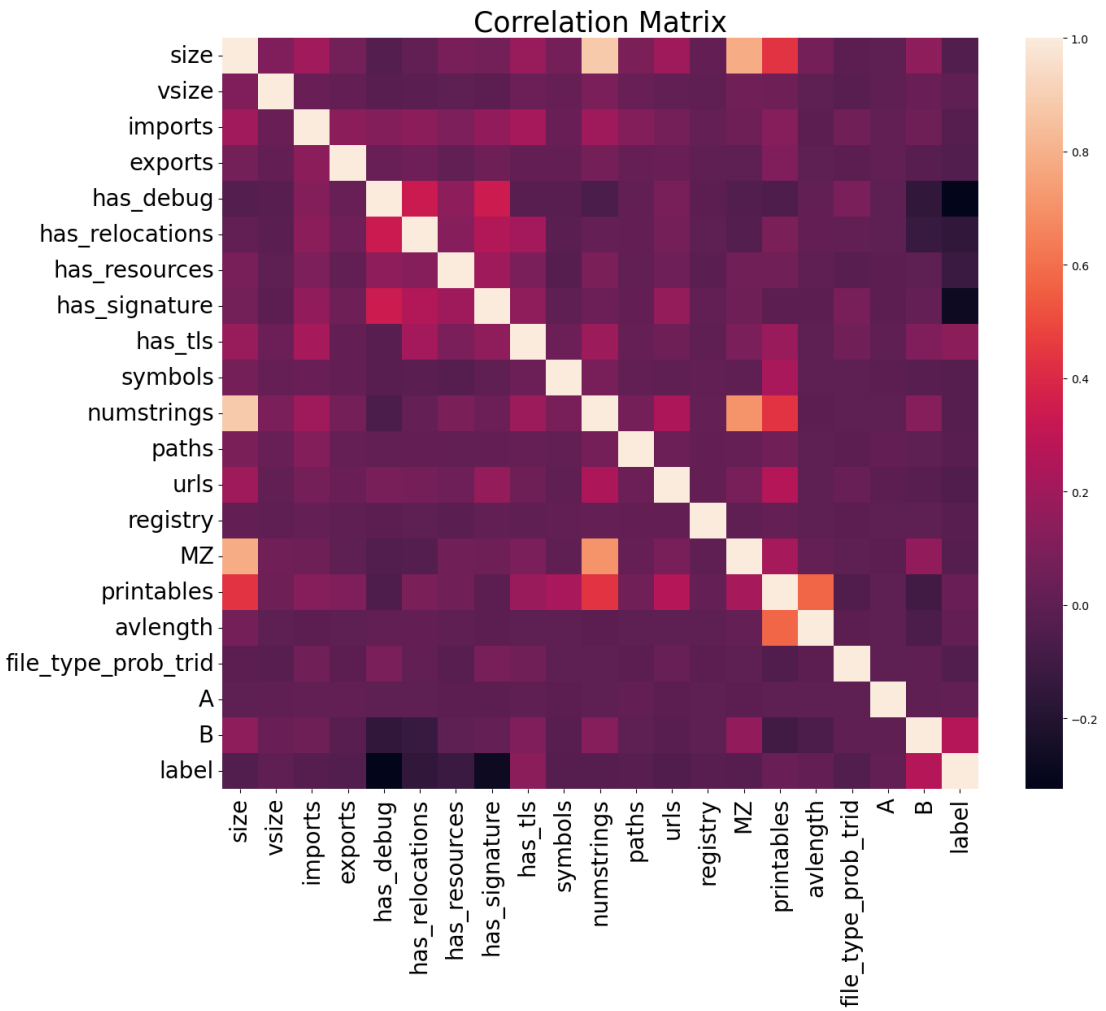
https://www.kaggle.com/code/aimack/how-to-handle-outliers

ROC and AUC | Kaggle

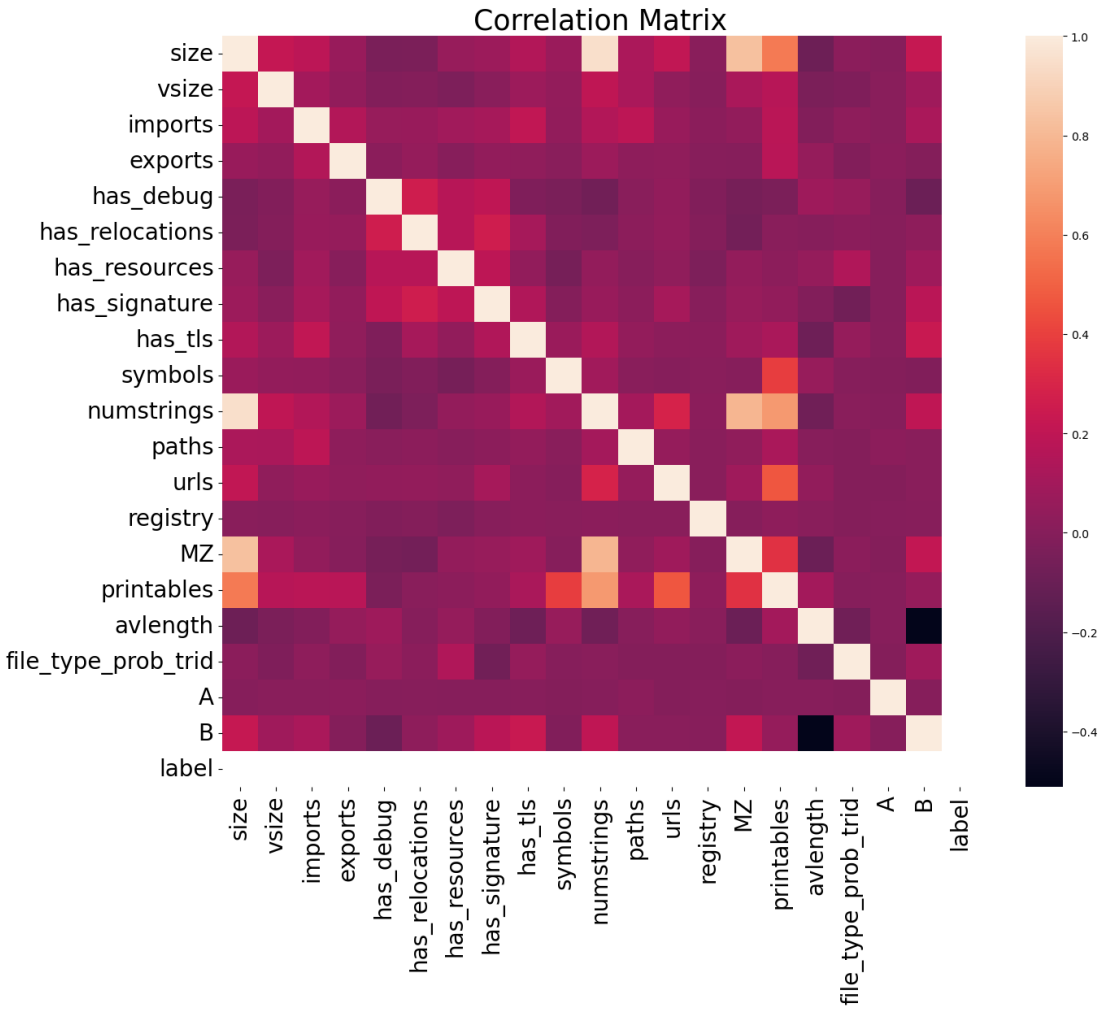
Comprehensive Guide on Feature Selection | Kaggle

כל הפלטים של המחברת ממוספרים:

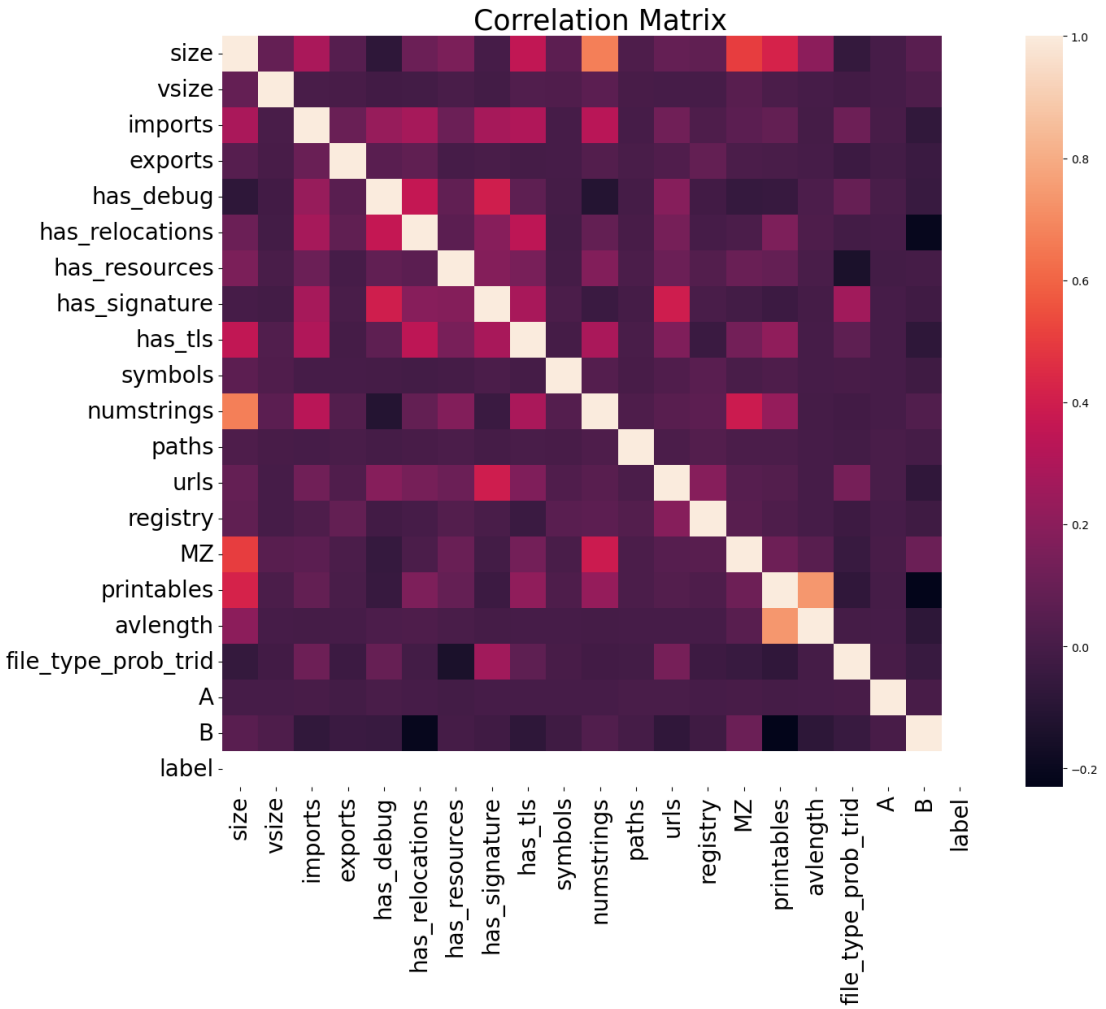
1.



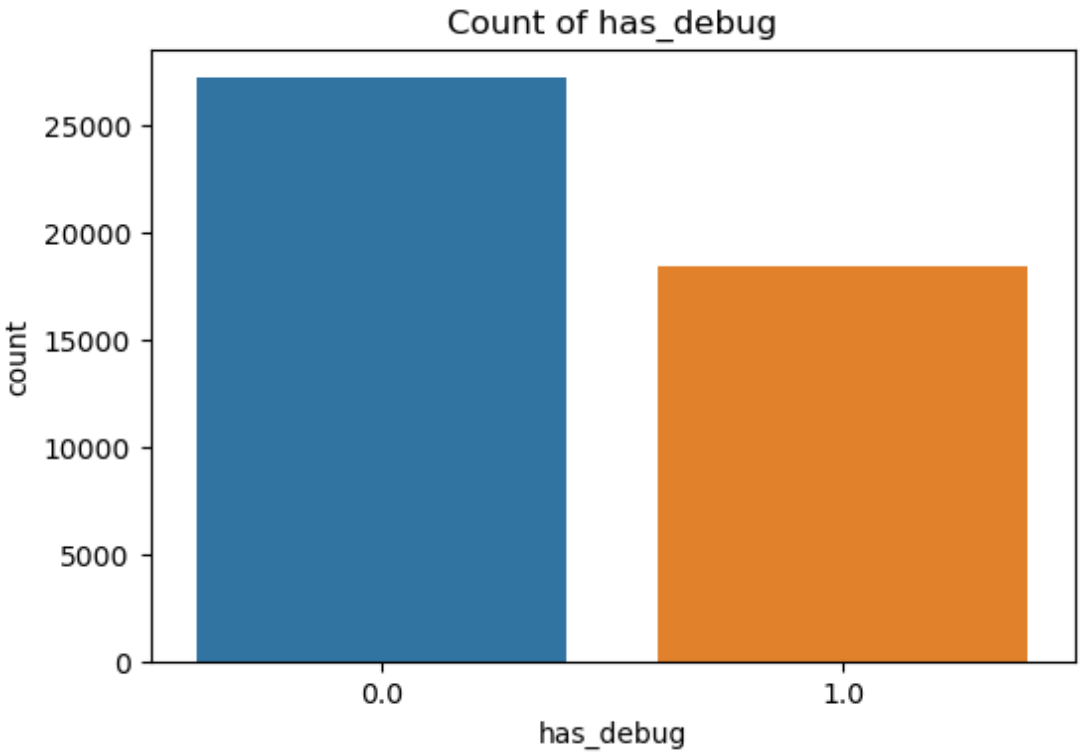
2.


Correlation Matrix
size
vsize
imports
exports
has_debug
has_relocations
has_resources
has_signature
has_tls
symbols
numstrings
paths
urls
registry
MZ
printables
avlength
file_type_prob_trid
A
B
label
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

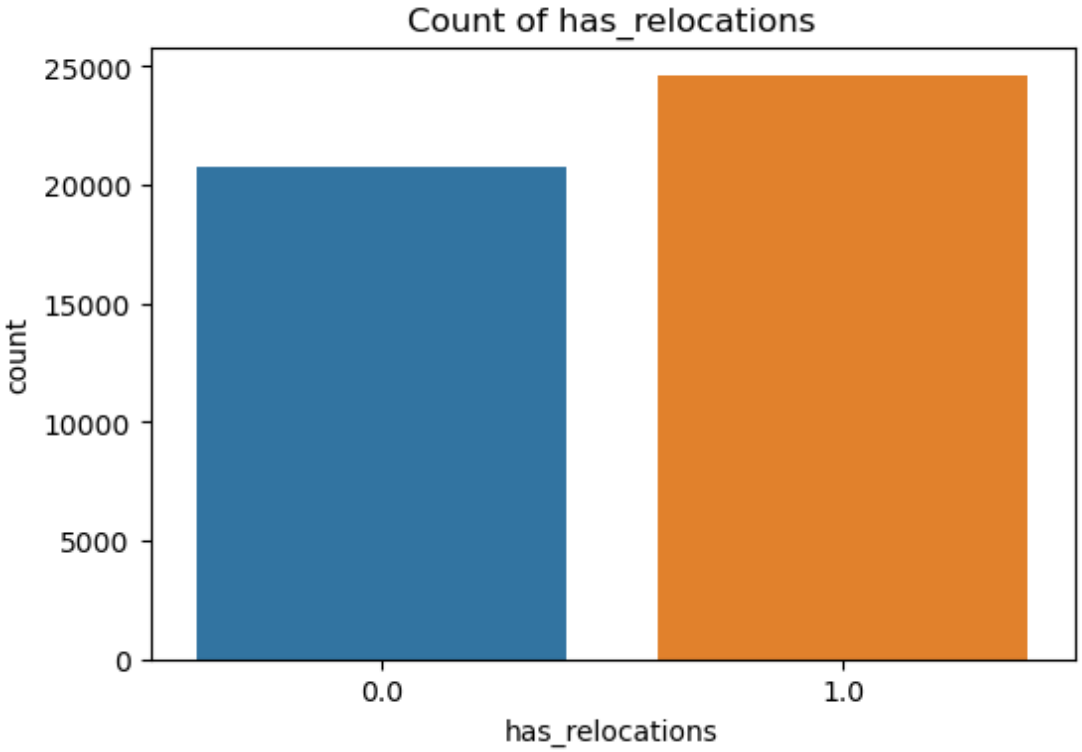
3.


Correlation Matrix
size
vsize
imports
exports
has_debug
has_relocations
has_resources
has_signature
has_tls
symbols
numstrings
paths
urls
registry
MZ
printables
avlength
file_type_prob_trid
A
B
label
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

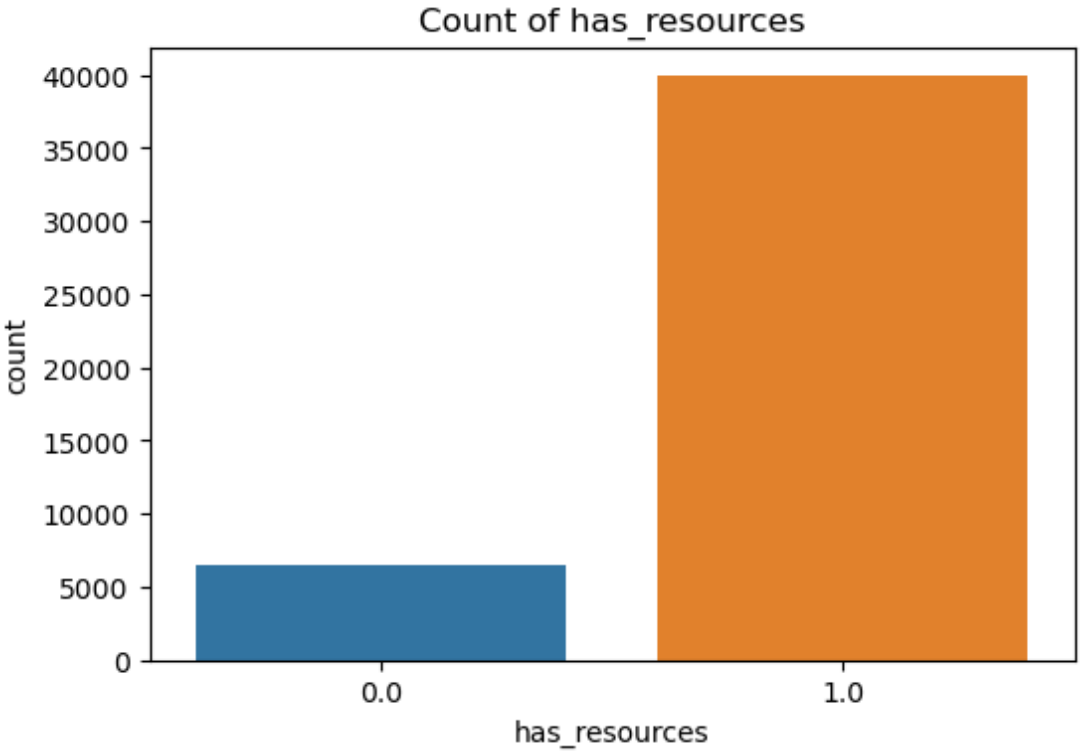
4.


Count of has_debug
count
25000
20000
15000
10000
5000
0
0.0
1.0
has_debug

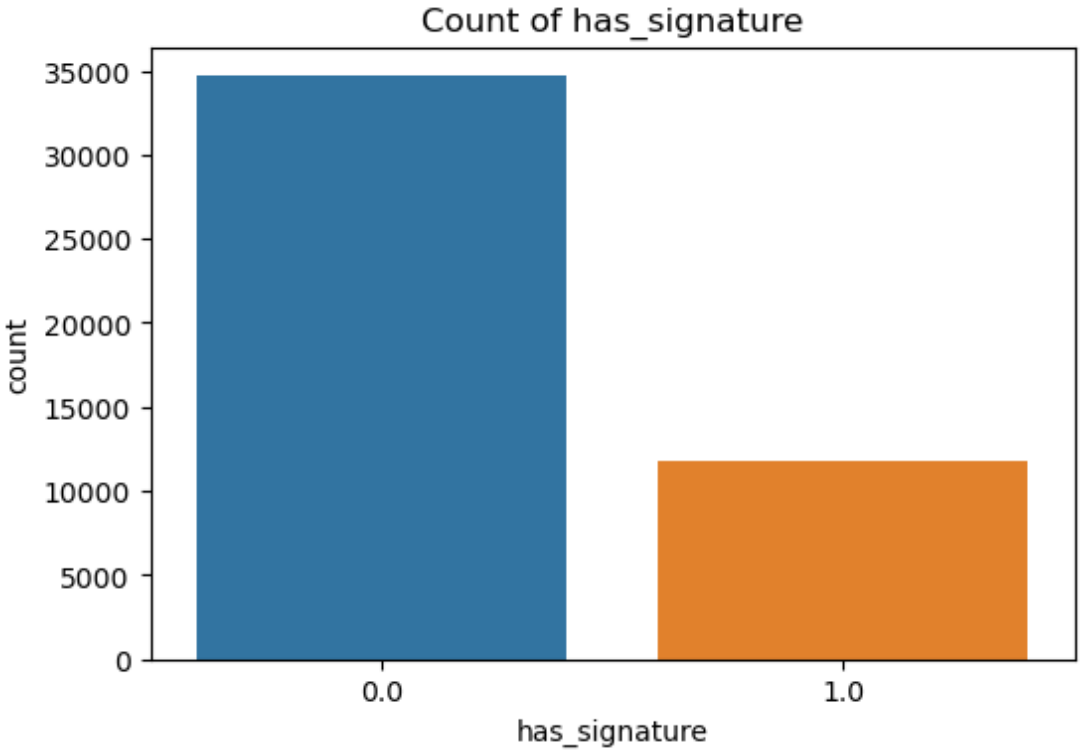
5.


Count of has_relocations
25000
20000
15000
10000
5000
0
count
0.0
1.0
has_relocations

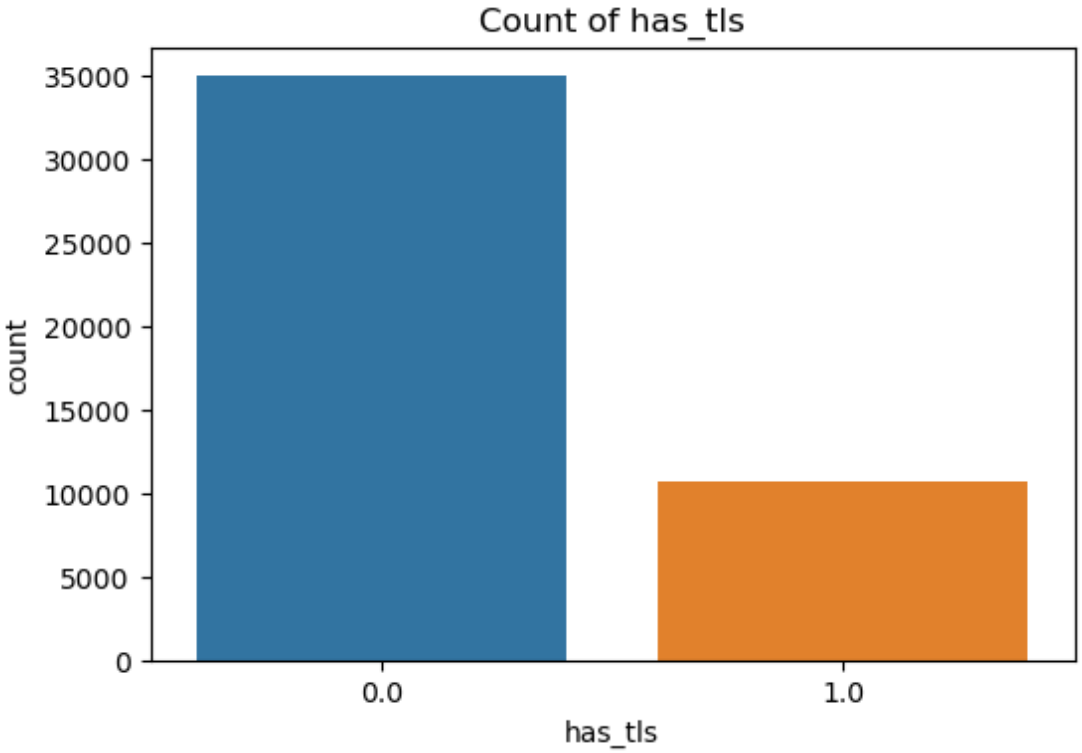
6.


Count of has_resources
40000
35000
30000
25000
20000
15000
10000
5000
0
count
0.0
1.0
has_resources

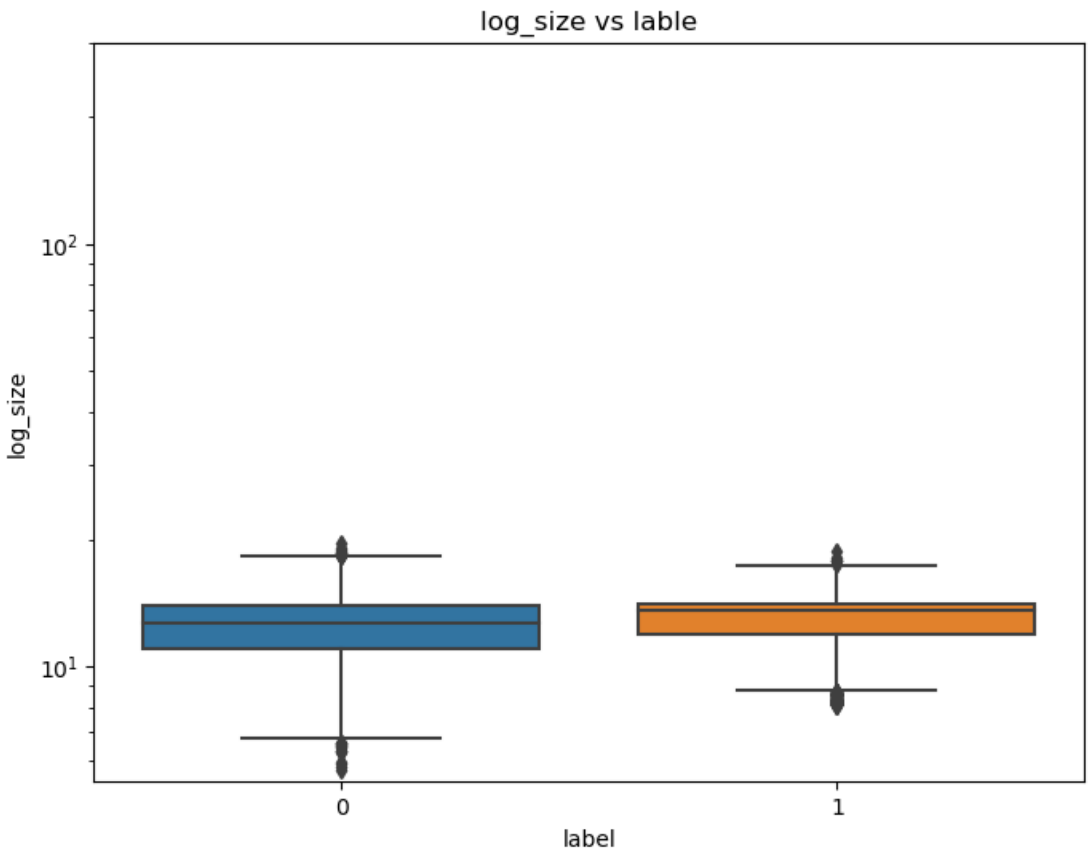
7.


Count of has_signature
35000
30000
25000
20000
15000
10000
5000
0
count
0.0
1.0
has_signature

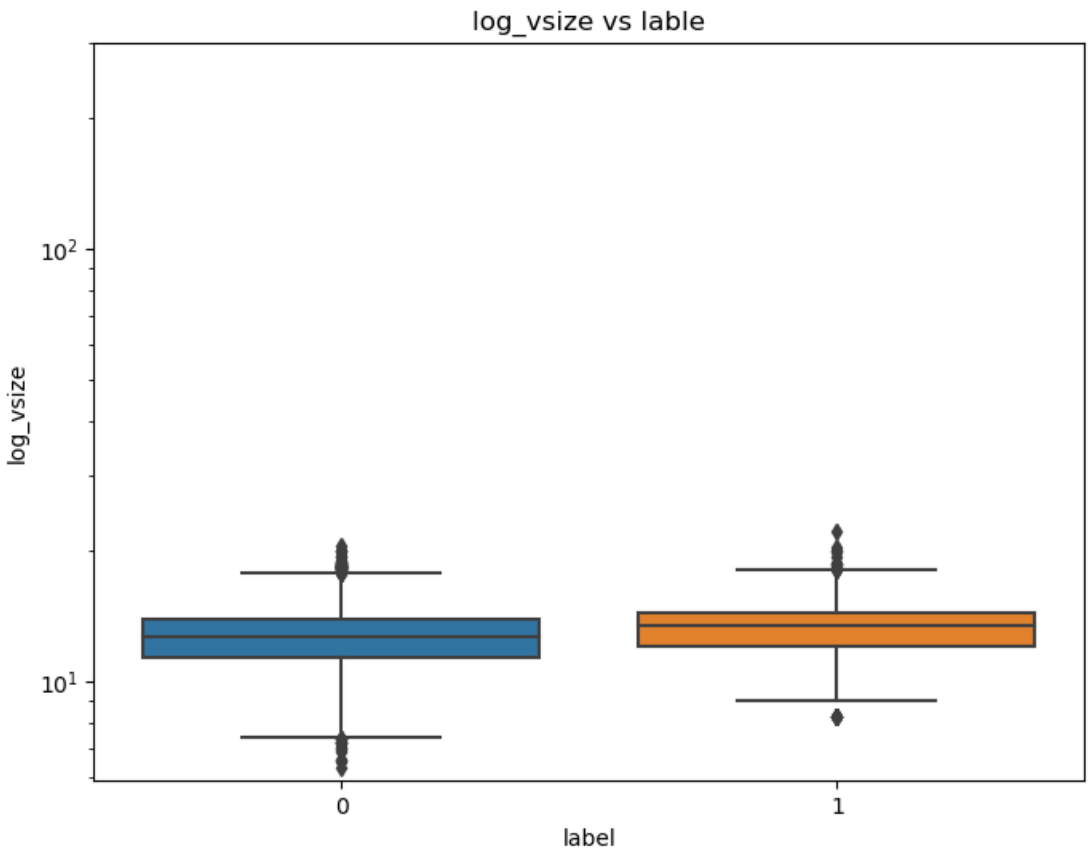
8.


Count of has_tls
35000
30000
25000
20000
15000
10000
5000
0
count
0.0
1.0
has_tls

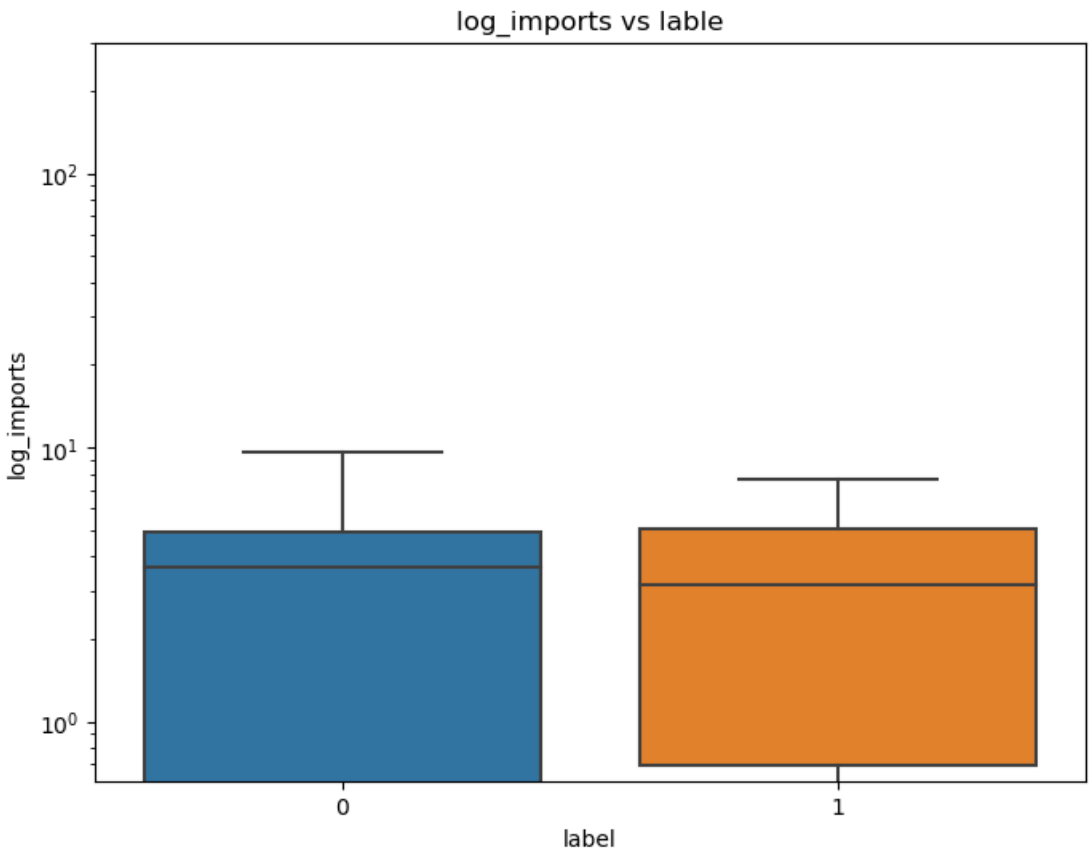
9.


log_size vs lable
log_size
10^2
10^1
0
1
label

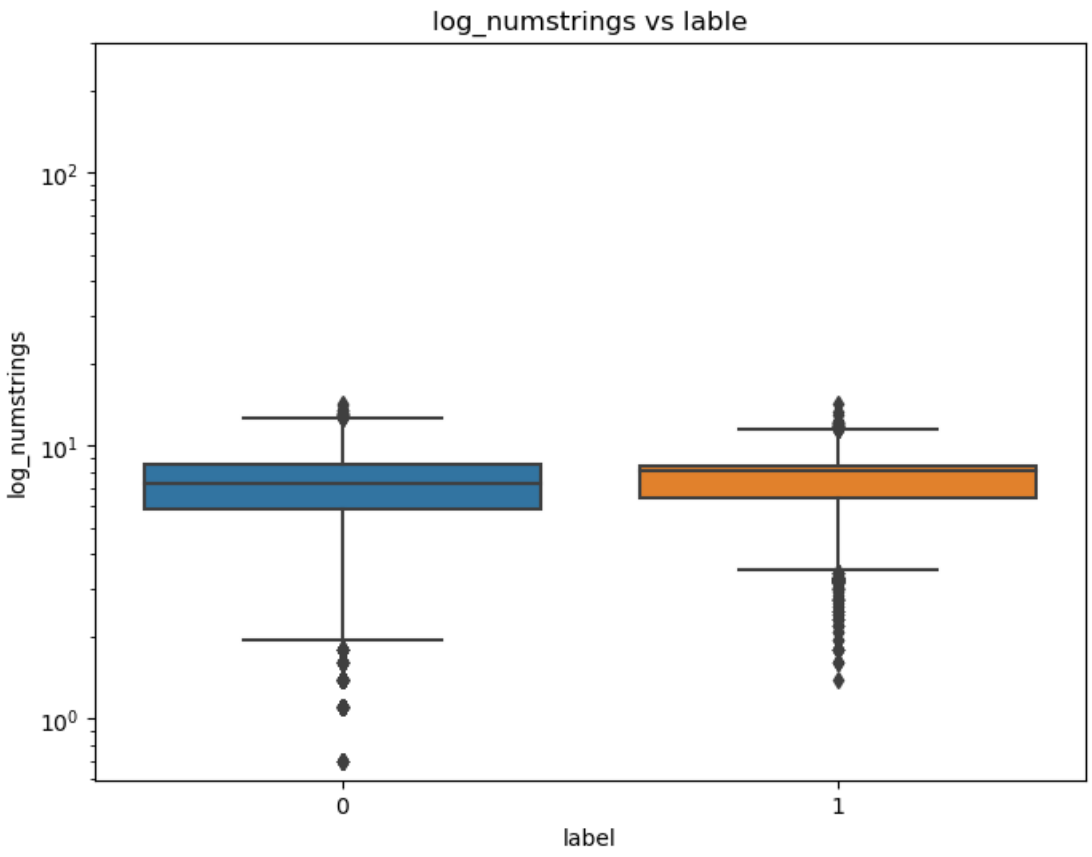
10.


log_vsize vs lable
log_vsize
10^2
10^1
0
1
label

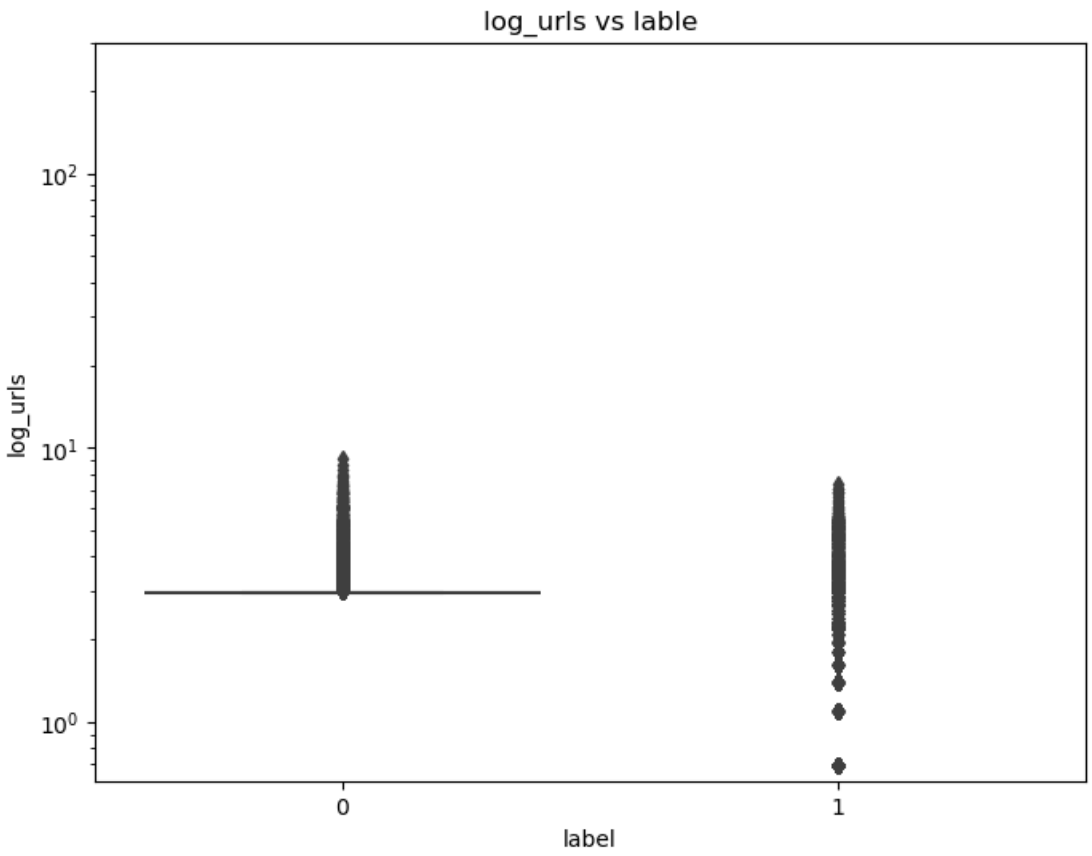
11.


log_imports vs lable
log_imports
10^2
10^1
10^0
0
1
label

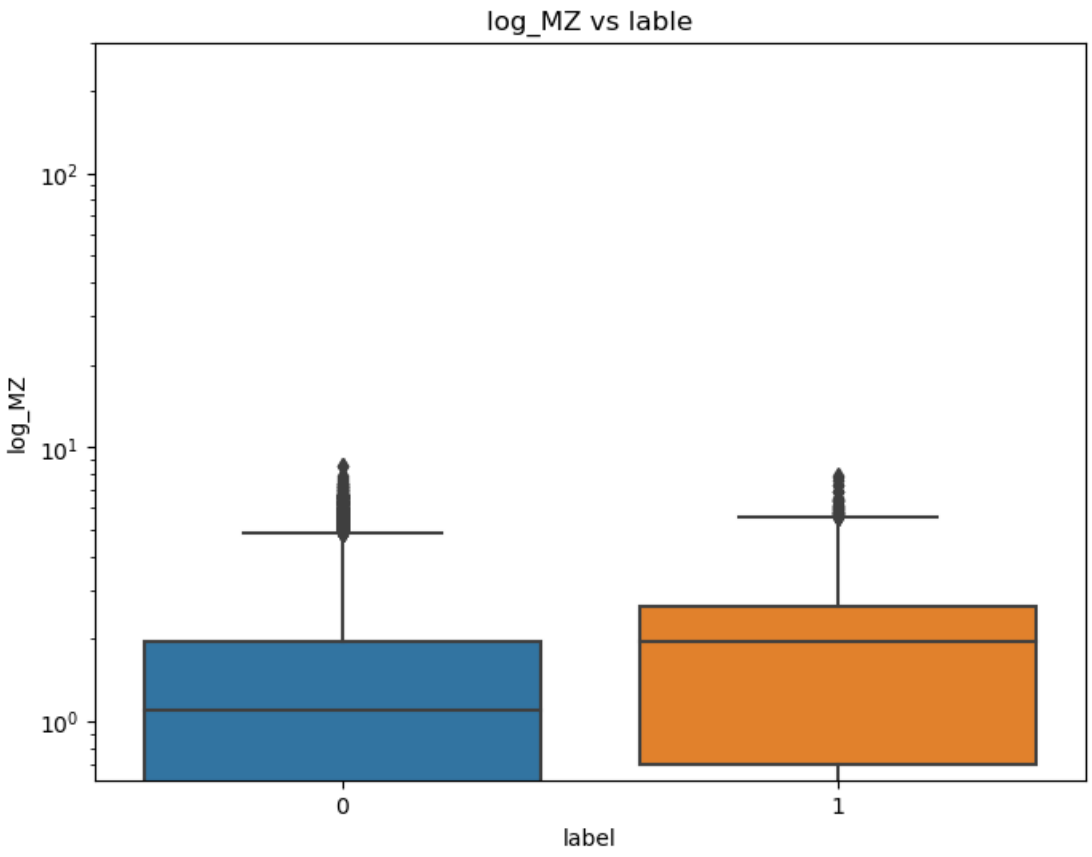
12.


log_numstrings vs lable
log_numstrings
10^2
10^1
10^0
0
1
label

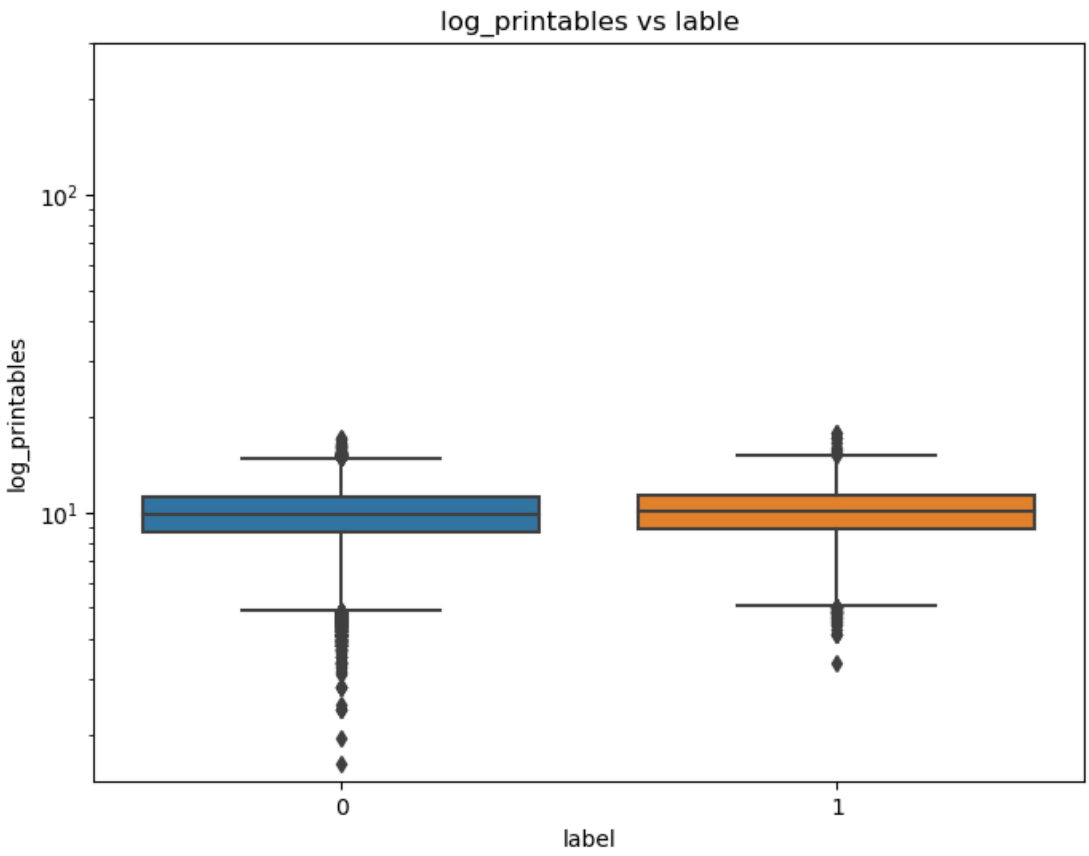
13.


log_urls vs lable
log_urls
10^2
10^1
10^0
0
1
label

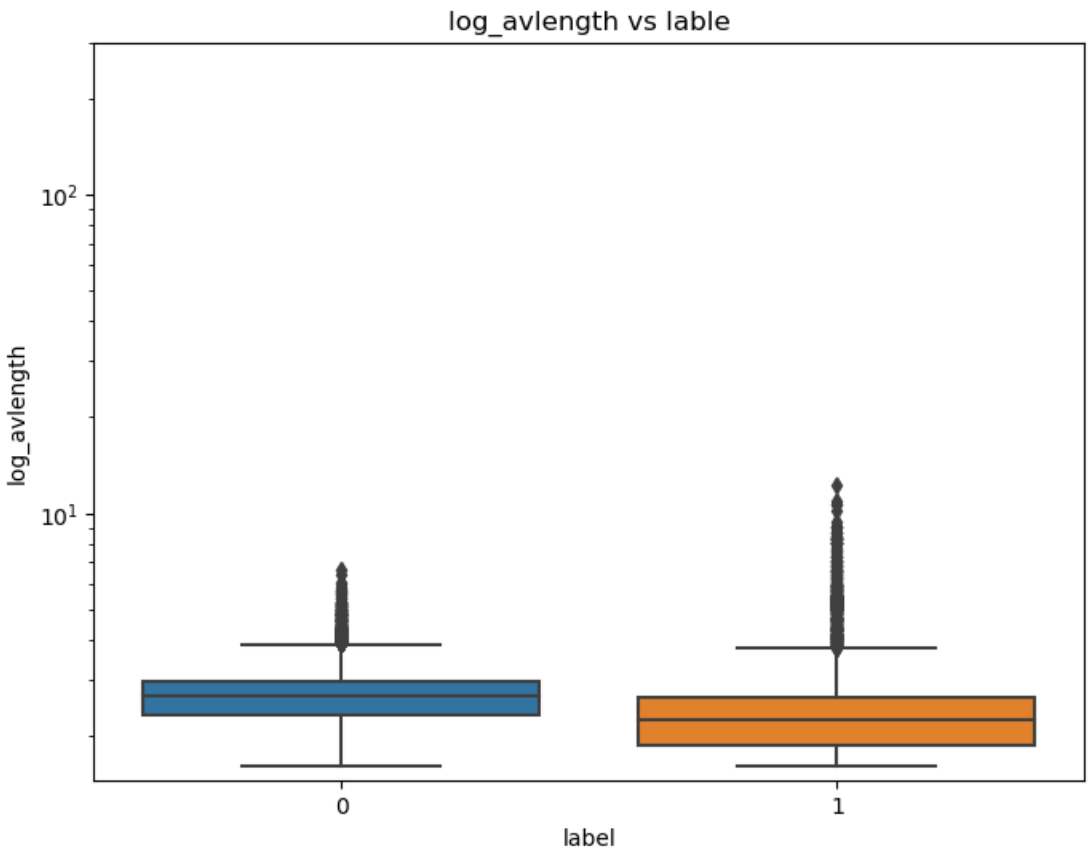
14.


log_MZ vs lable
log_MZ
10^2
10^1
10^0
0
1
label

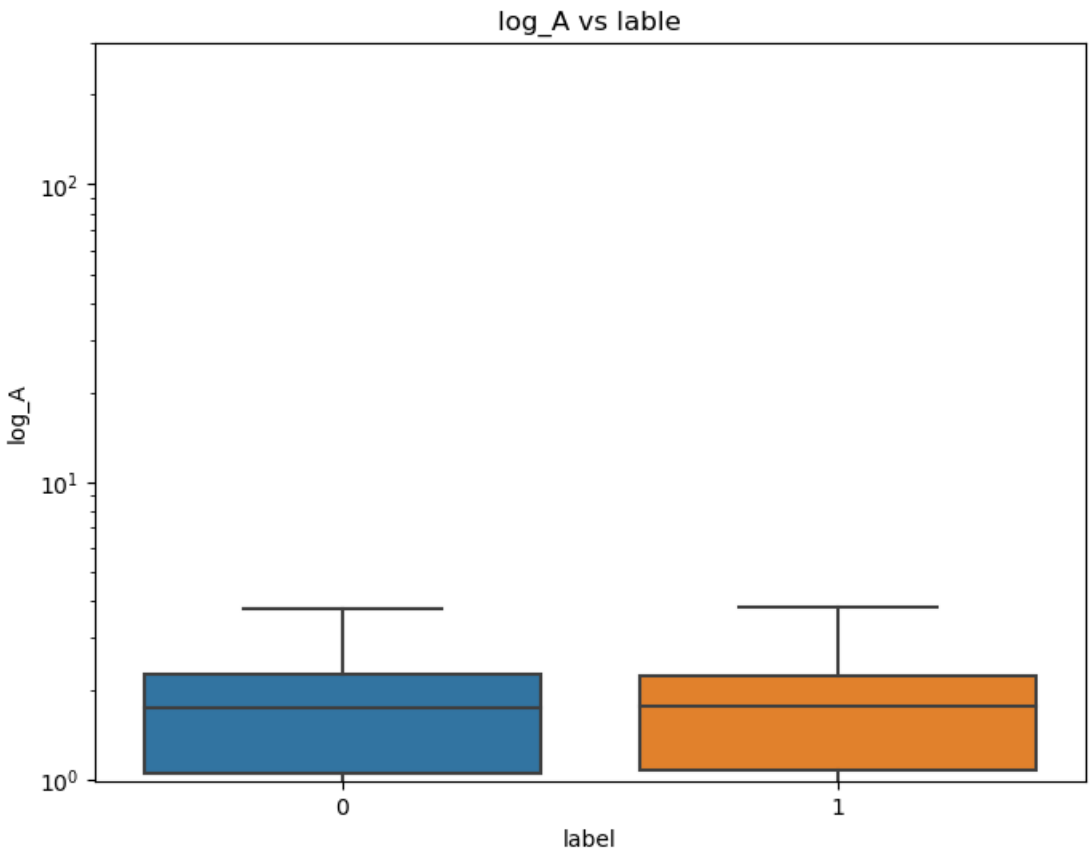
15.


log_printables vs lable
log_printables
10^2
10^1
0
1
label

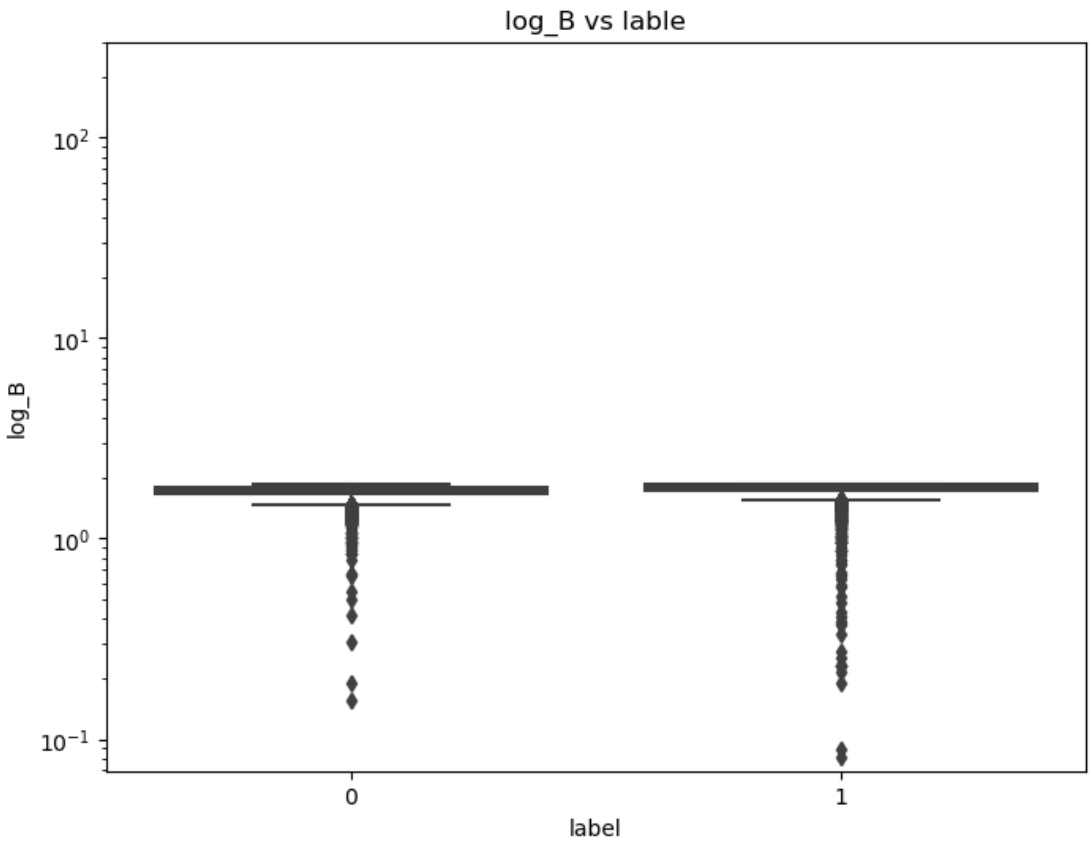
16.


log_avlength vs lable
log_avlength
10^2
10^1
0
1
label

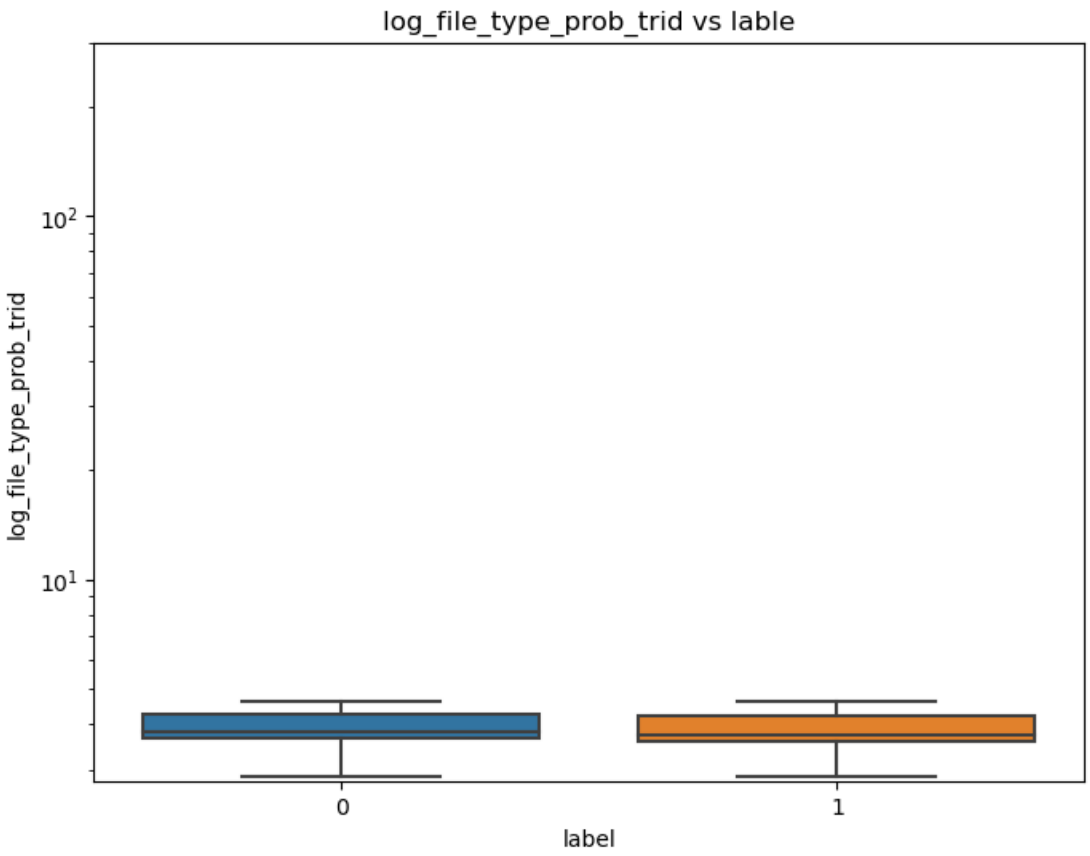
17.


log_A vs lable
log_A
10^2
10^1
10^0
0
1
label

18.


log_B vs lable
log_B
10^2
10^1
10^0
10^{-1}
0
1
label

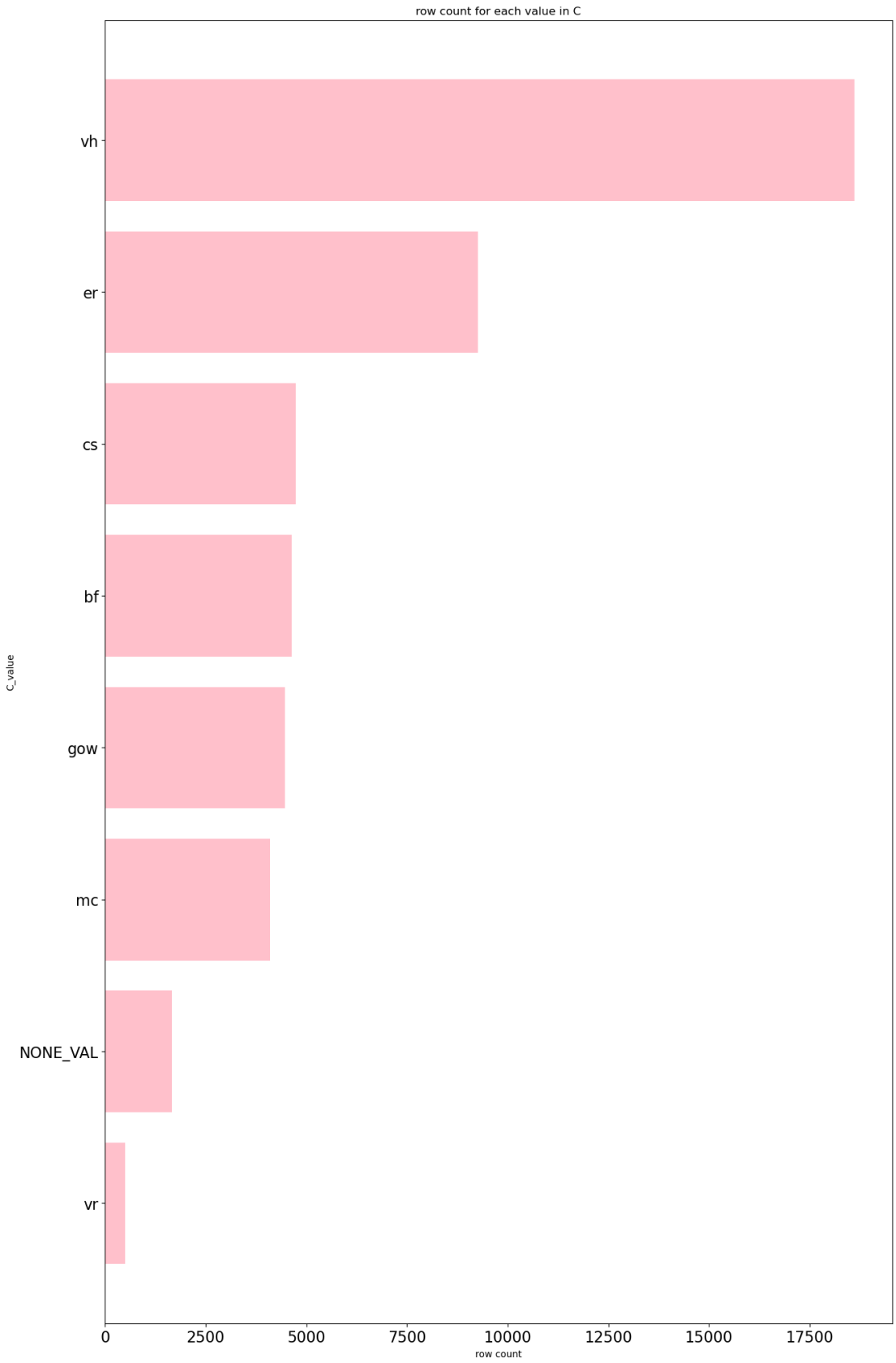
19.


log_file_type_prob_trid vs lable
log_file_type_prob_trid
10^2
10^1
0
1
label

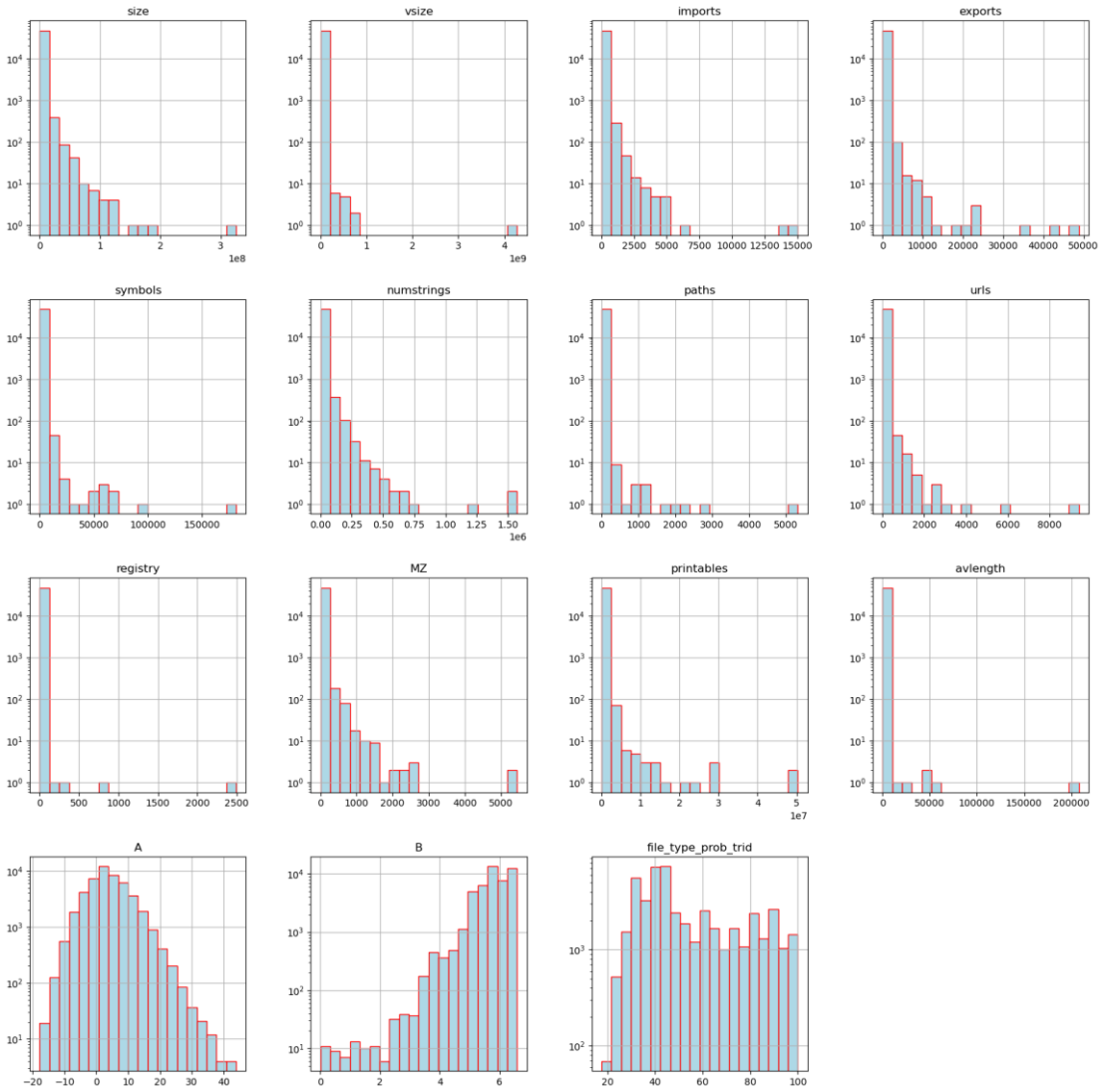
20.


row count for each value in file_type_trid
file_type_trid_value
Win64 Executable (generic)
Win32 Executable MS Visual C++ (generic)
Win32 Executable (generic)
Win32 Dynamic Link Library (generic)
Generic CIL Executable (.NET, Mono, etc.)
Windows Control Panel Item (generic)
InstallShield setup
UPX compressed Win32 Executable
DOS Executable Generic
Win32 Executable Microsoft Visual Basic 6
Win32 EXE PECompact compressed (generic)
Generic .NET DLL/Assembly
OS/2 Executable (generic)
WinRAR Self Extracting archive (4.x-5.x)
NSIS - Nullsoft Scriptable Install System
Microsoft Visual C++ compiled executable (generic)
Win32 Executable Delphi generic
Inno Setup installer
Win16 NE executable (generic)
Windows ActiveX control
DOS Borland compiled Executable (generic)
DOS Executable Borland Pascal 7.0x
Windows screen saver
DirectShow filter
Generic Win/DOS Executable
Win32 EXE PECompact compressed (v2.x)
Win32 Executable Borland Delphi 7
Win32 Executable PureBasic (generic)
Win32 Executable MS Visual C++ 4.x
ASPack compressed Win32 Executable (generic)
Microsoft Resource Library (x86)
Win32 EXE Yoda's Crypter
AutoIt3 compiled script executable
Win32 Dynamic Link Library - Borland C/C++
Win32 Executable Borland Delphi 5
WinRAR Self Extracting archive
Obfuscated subsetted Font
Win32 MS Cabinet Self-Extractor (WExtract stub)
Wise Installer executable
Windows Runtime Metadata
Win32 Executable Borland Delphi 6
MS generic-sfx Cabinet File Unpacker (32/64bit MSCFU)
MinGW32 C/C++ Executable
Win32 Executable PowerBASIC/Win 9.x
Win16/32 Executable Delphi generic
Windows Icons Library (generic)
Microsoft Update - Self Extracting Cabinet
Petite compressed Win32 executable
DOS Executable Borland C++
Win32 Executable MS Visual C++ 5.0
Win32 Executable Microsoft Visual Basic 5
UPX compressed Win64 Executable
Win32 Executable Microsoft Visual Basic 4
Clipper DOS Executable
Win32 Executable Borland Delphi 3
Python Dynamic module
NeoLite compressed Win32 executable
CExe compressed Win32 executable
Microsoft Windows Defender Virus Definition Module
Mew compressed Win32 Executable
Borland Package Library
UPX - NRV compressed Win32 Executable
Windows system resource library
GRASP animation
DOS Borland Executable (generic)
Total Commander Lister extension (plugin)
16-bit Microsoft C compiled executable (generic)
Winzip Win32 self-extracting archive (generic)
Microsoft Resource Library
Macromedia Projector/Flash executable
Win 9x/ME Control Panel applet
Thinapp Packaged Portable Application Launcher executable
Microsoft Input Method Editor
Win32 Executable Watcom C++ (generic)
Extensible Firmware Interface x86-64 application
Win32 EXE RosAsm
VirtualDub Filter Plug-in
WinArchiver Mountable compressed Archive
Win32 Executable kkrunchy compressed
MS Flight Simulator Gauge
foobar 2000 generic component
Total Commander Content plugin
Photoshop filter plug-in
GIMP Plugin (Win)
PhotoShop plug-in (generic)
OllyDbg plugin
Microsoft CLR native image executable
tElock compressed/encrypted Win32 executable
0
1000
2000
3000
4000
5000
6000
7000
8000
row count

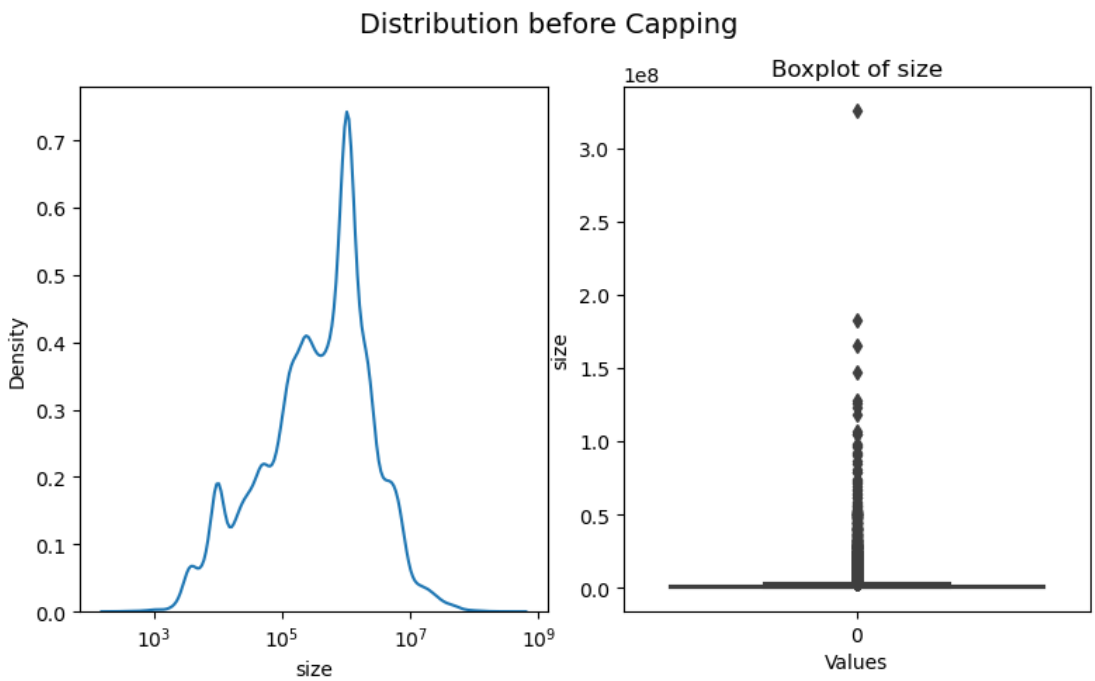
21)


row count for each value in C
vh
er
cs
bf
gow
mc
NONE_VAL
vr
c_value
0
2500
5000
7500
10000
12500
15000
17500
row count

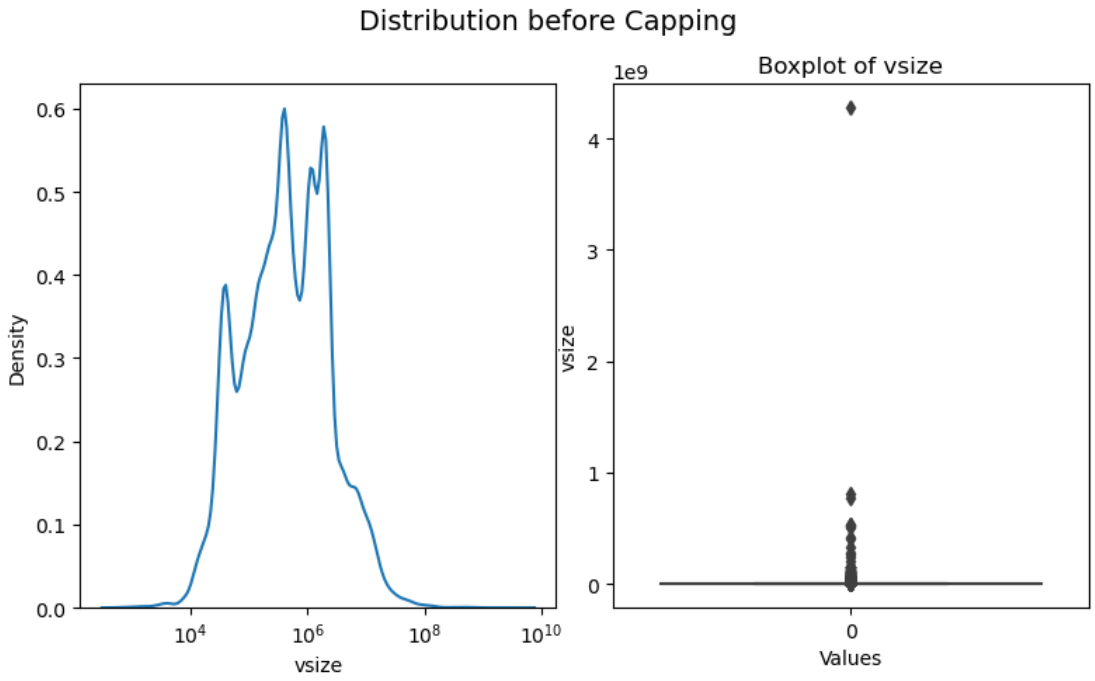
22)


size
vsize
imports
exports
symbols
numstrings
paths
urls
registry
MZ
printables
avlength
A
B
file_type_prob_trid

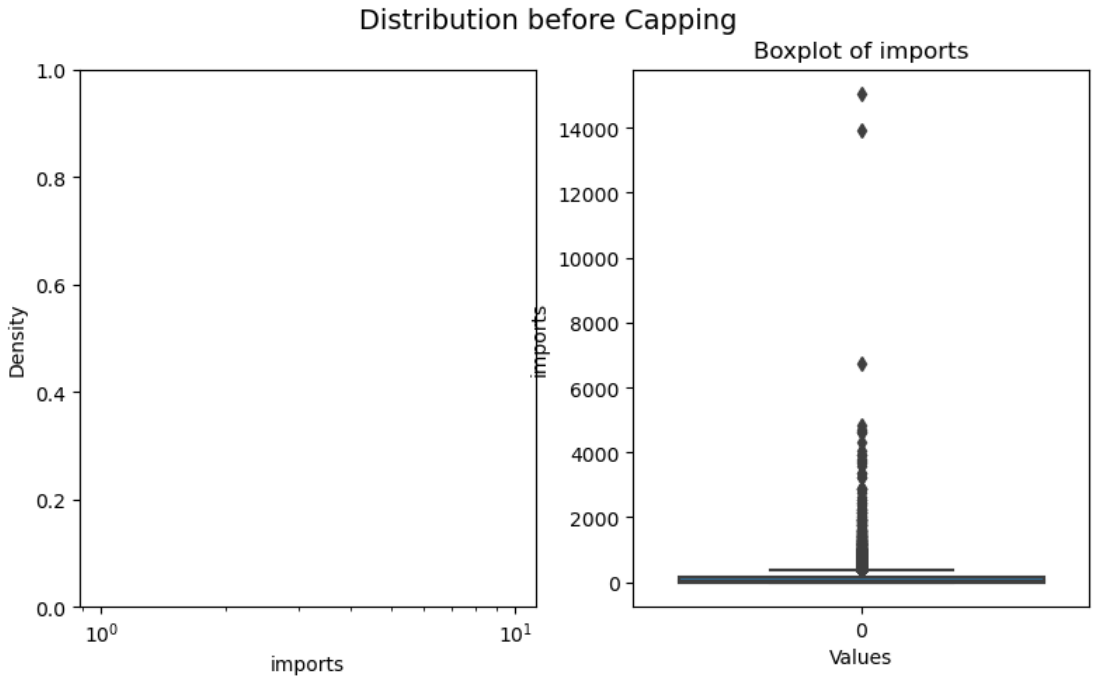
23)


Distribution before Capping
Density
size
Boxplot of size
1e8
size
Values

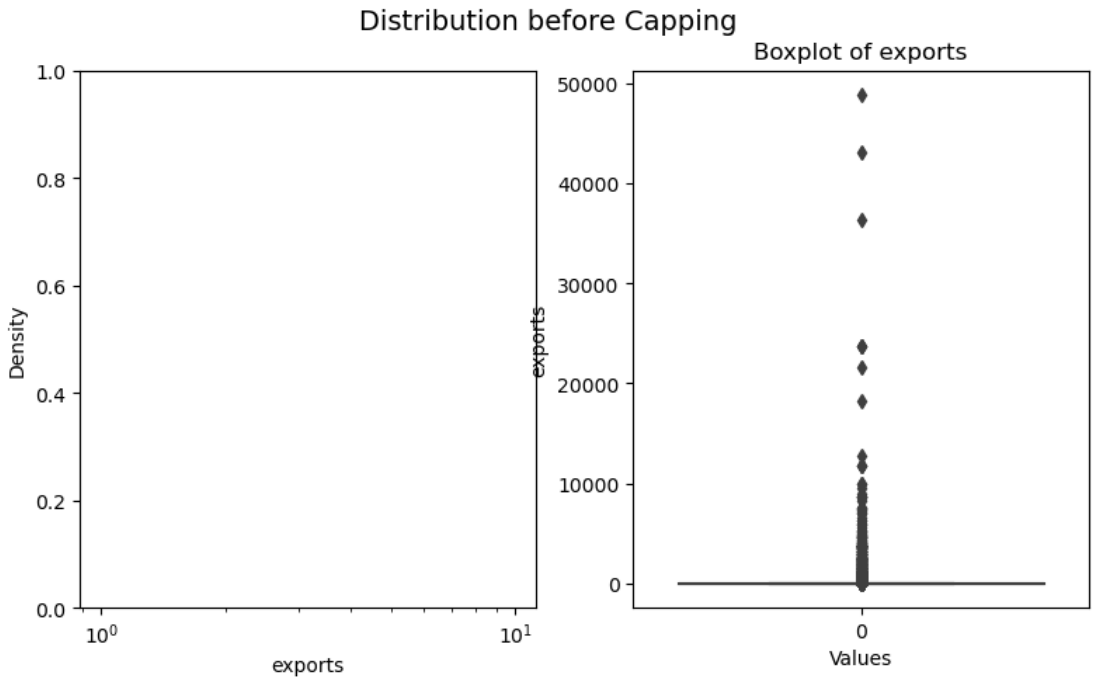
24)


Distribution before Capping
Density
vsize
Boxplot of vsize
1e9
vsize
Values

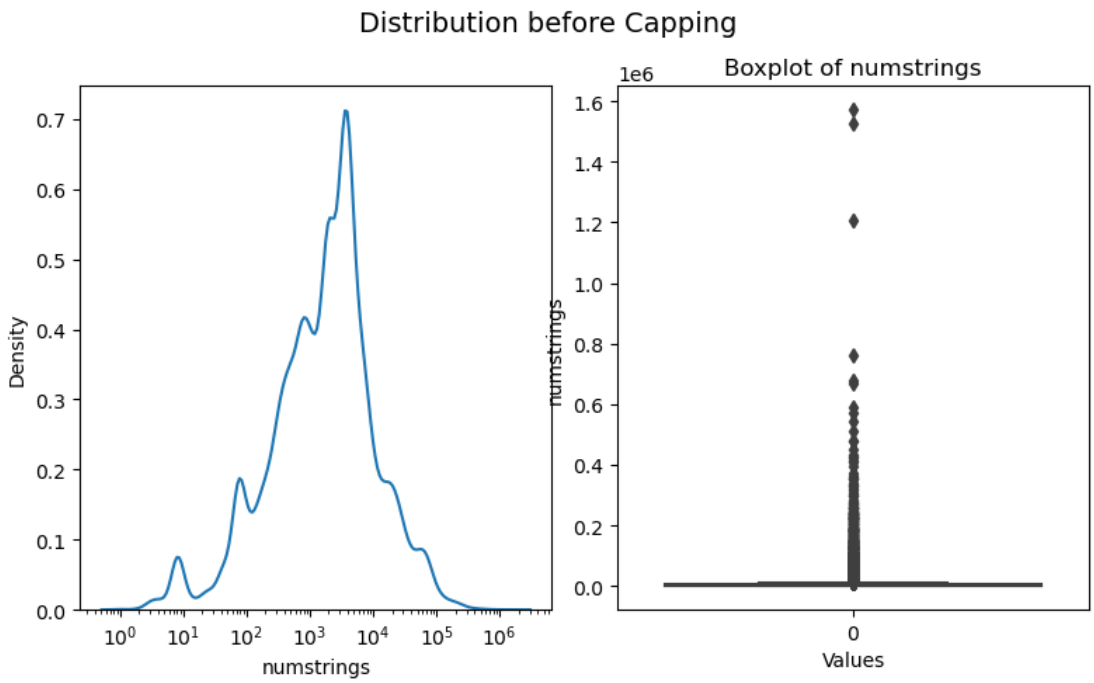
25)


Distribution before Capping
Density
imports
Boxplot of imports
imports
Values

26)

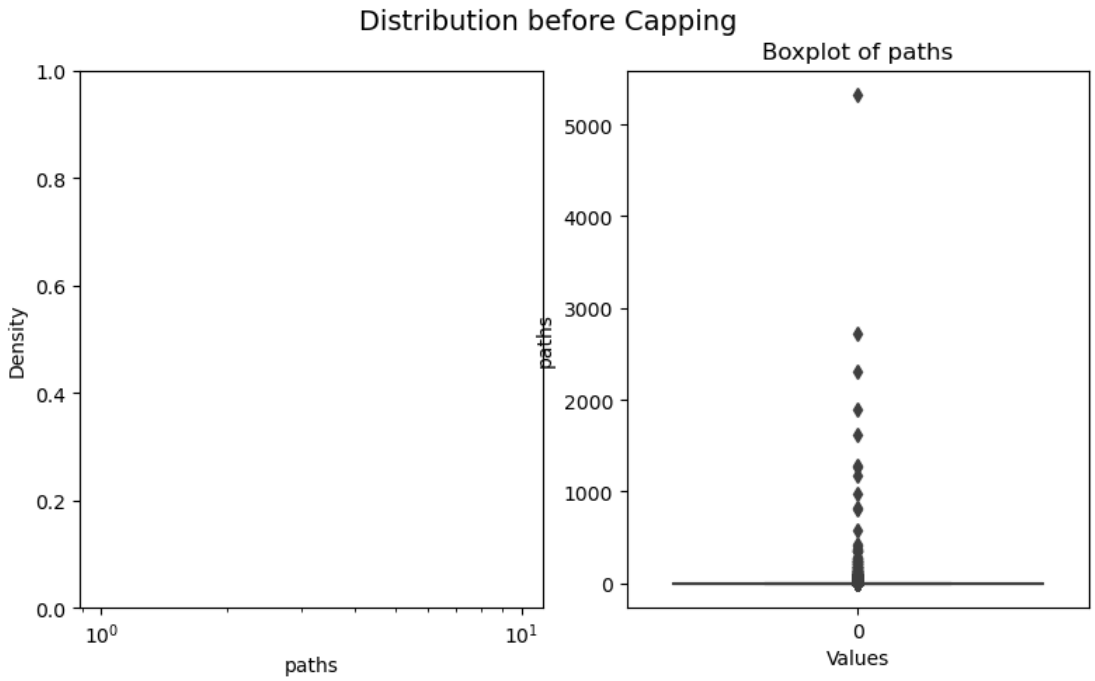

Distribution before Capping
Density
exports
Boxplot of exports
exports
Values

27)

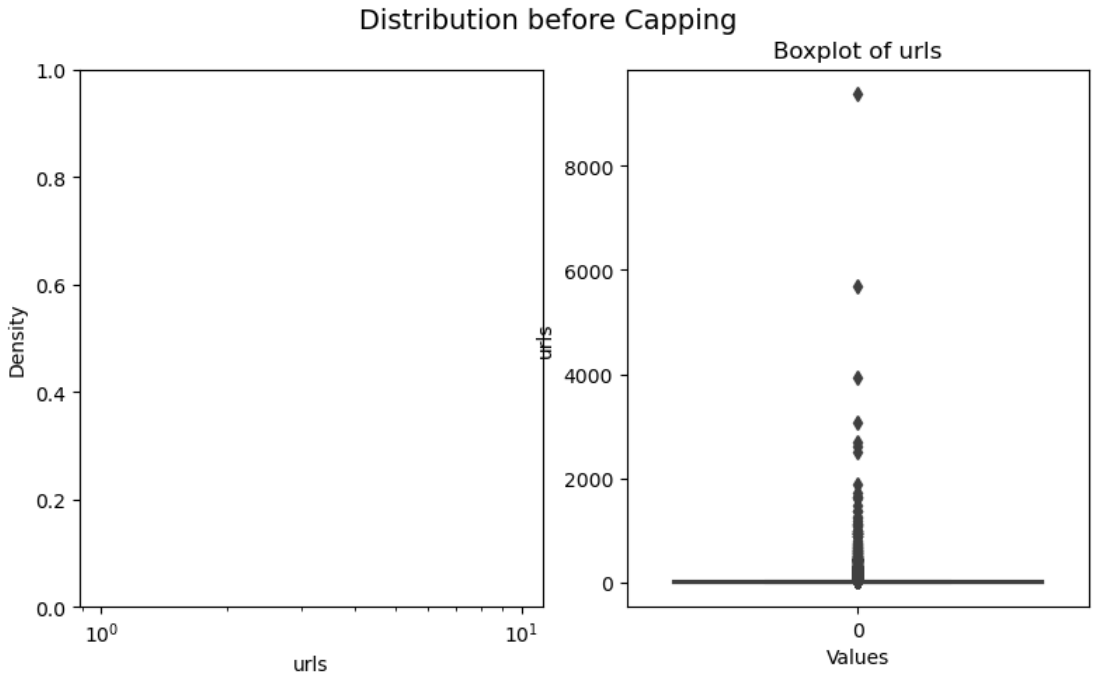

Distribution before Capping
Density
numstrings
Boxplot of numstrings
1e6
numstrings
Values

28)

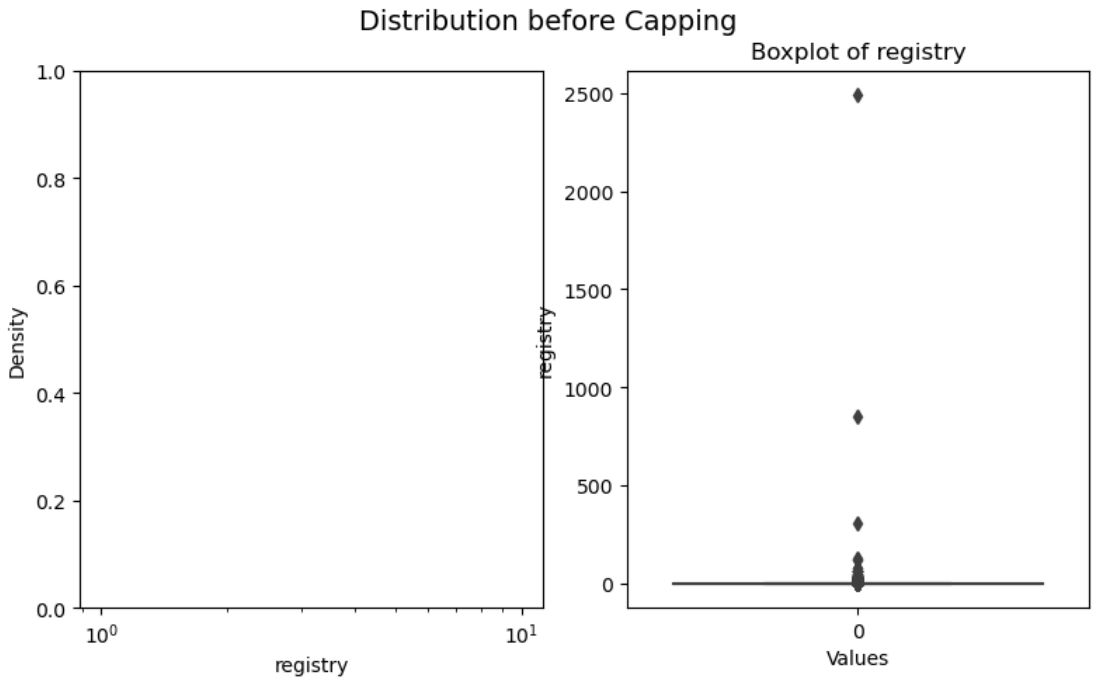
29)


Distribution before Capping
Boxplot of paths
Density
paths
Values
1.0
0.8
0.6
0.4
0.2
0.0
10^0
10^1
5000
4000
3000
2000
1000
0

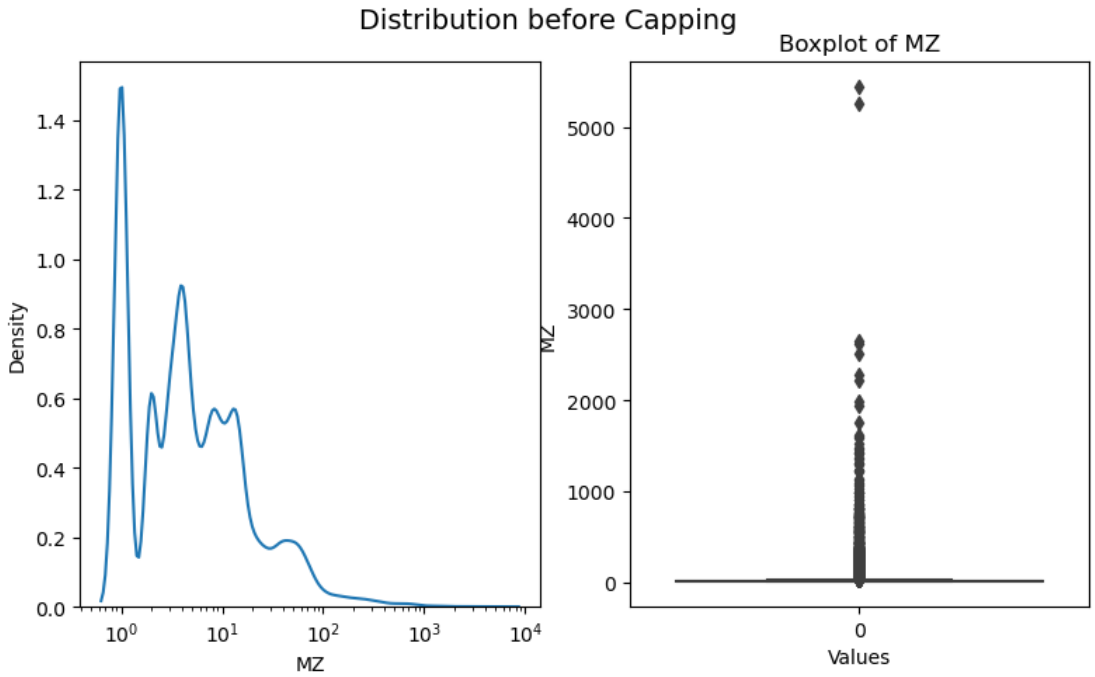
30)


Distribution before Capping
Boxplot of urls
Density
urls
Values
1.0
0.8
0.6
0.4
0.2
0.0
10^0
10^1
8000
6000
4000
2000
0

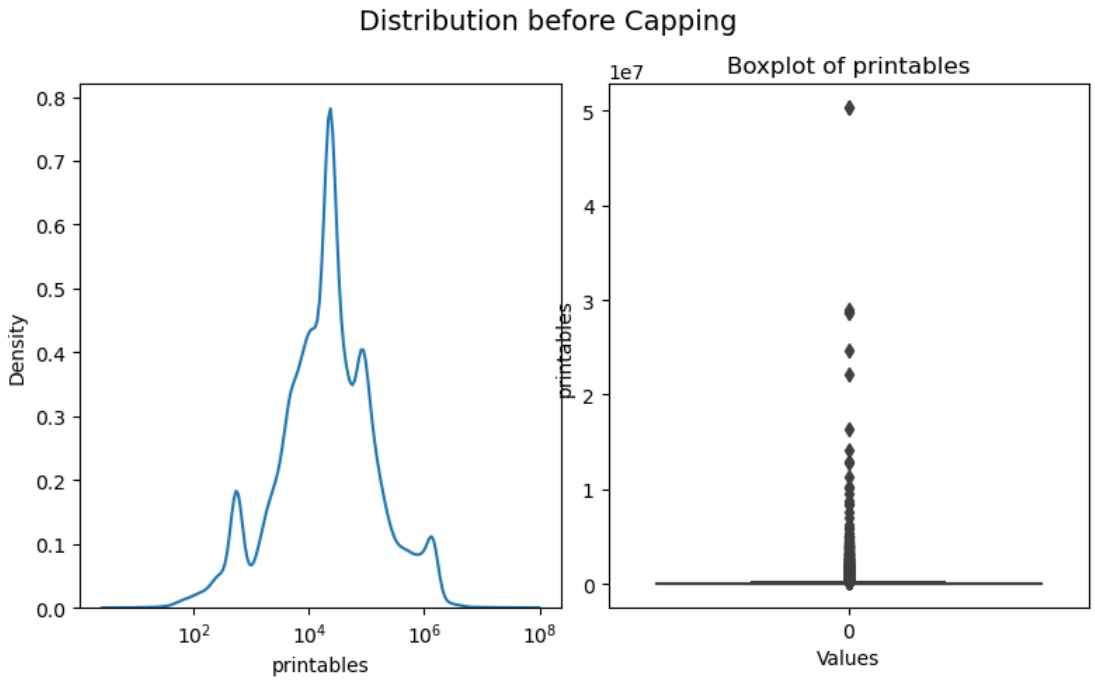
31)


Distribution before Capping
Boxplot of registry
Density
registry
registry
Values

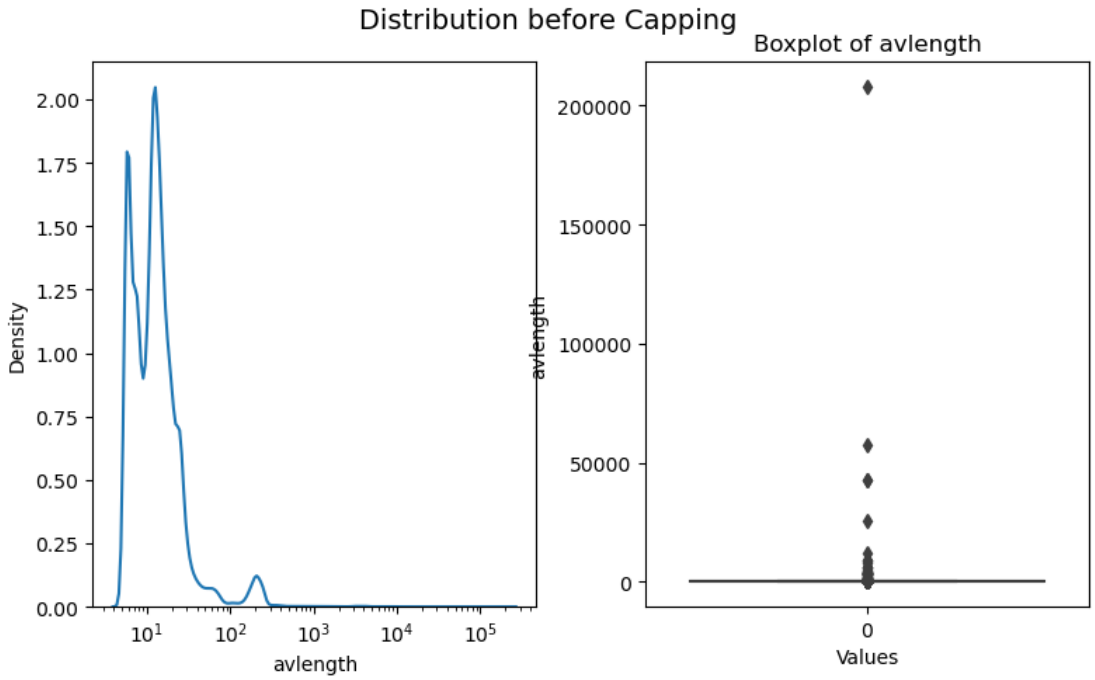
32)


Distribution before Capping
Boxplot of MZ
Density
MZ
MZ
Values

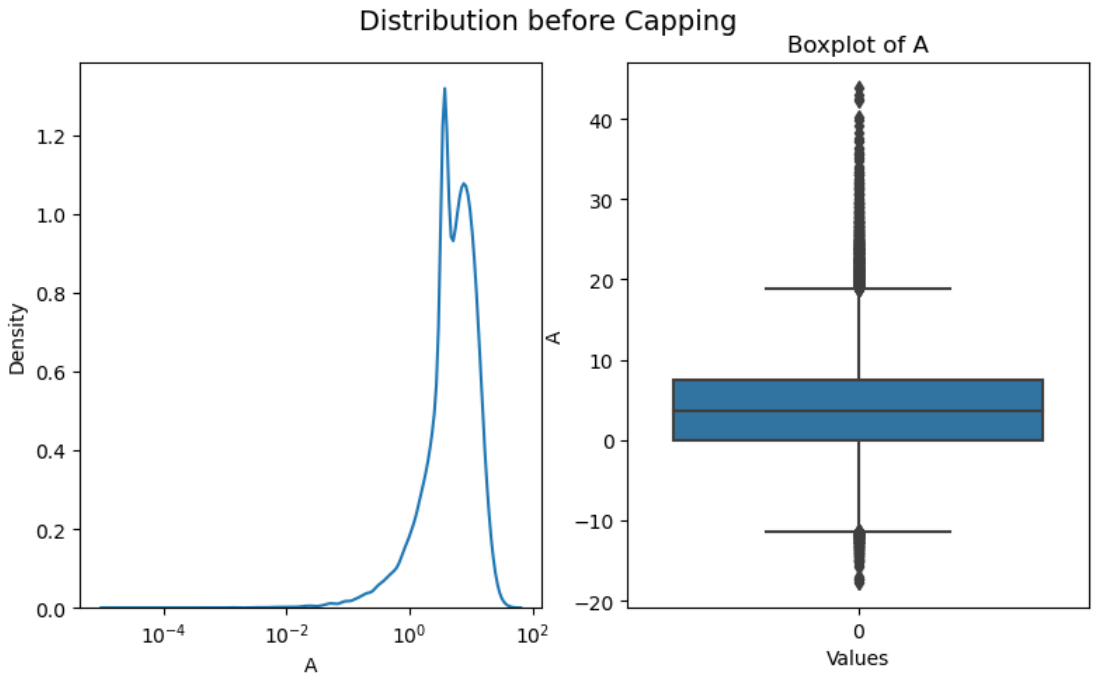
33)


Distribution before Capping
Density
printables
Boxplot of printables
1e7
printables
Values

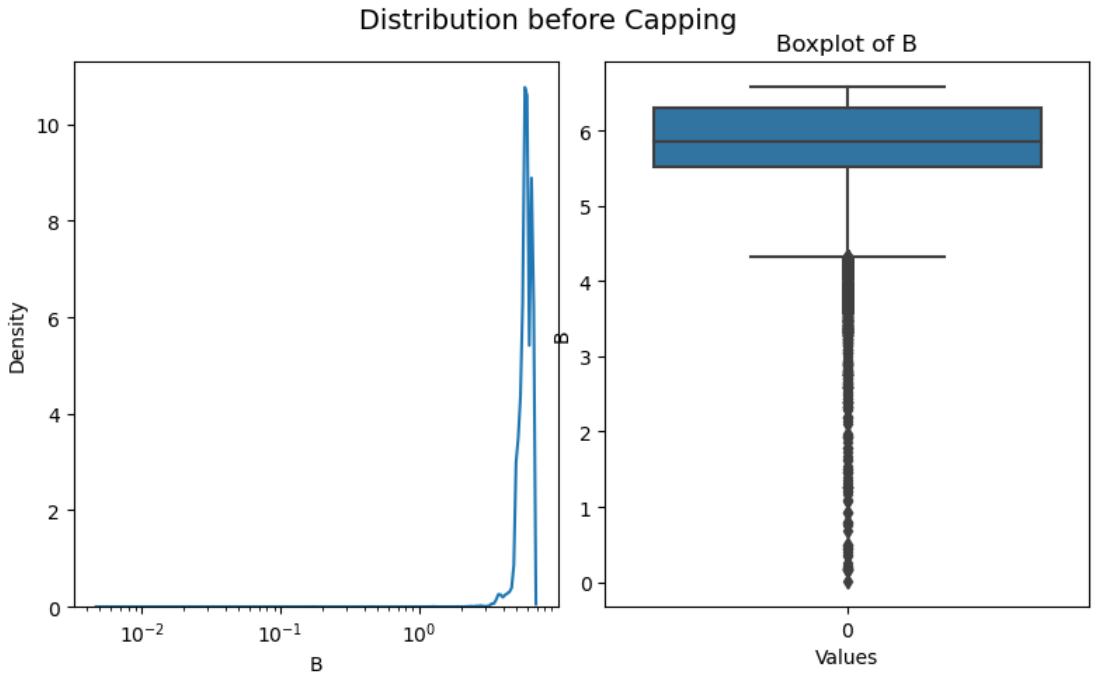
34)


Distribution before Capping
Density
avlength
Boxplot of avlength
avlength
Values

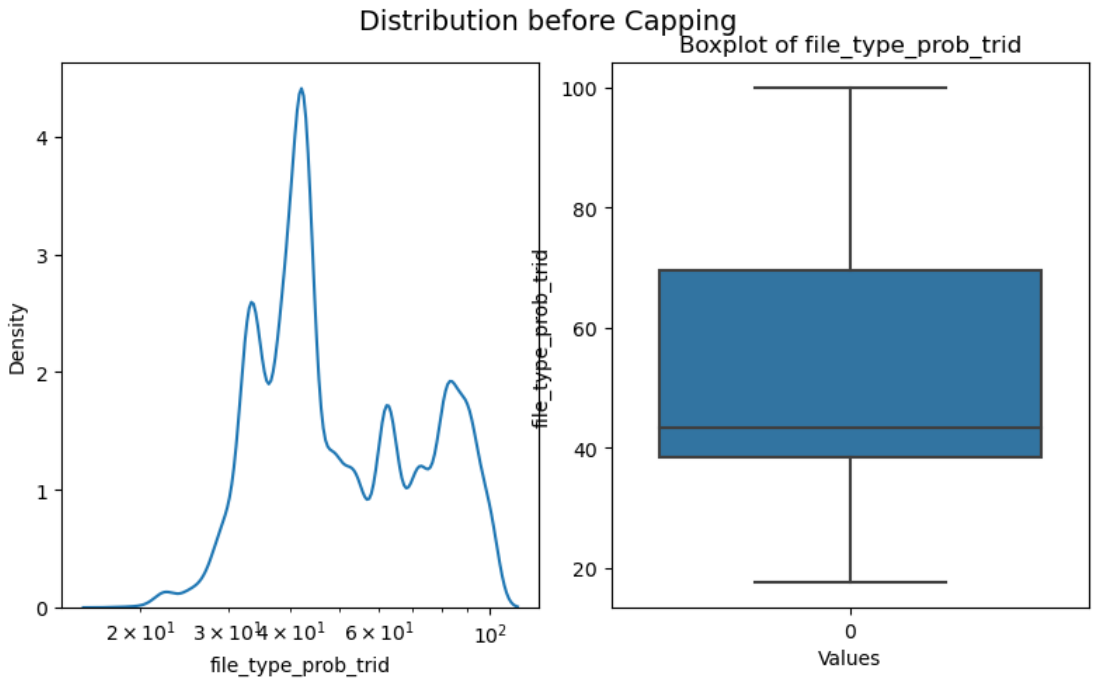
35)


Distribution before Capping
Density
A
Boxplot of A
A
Values

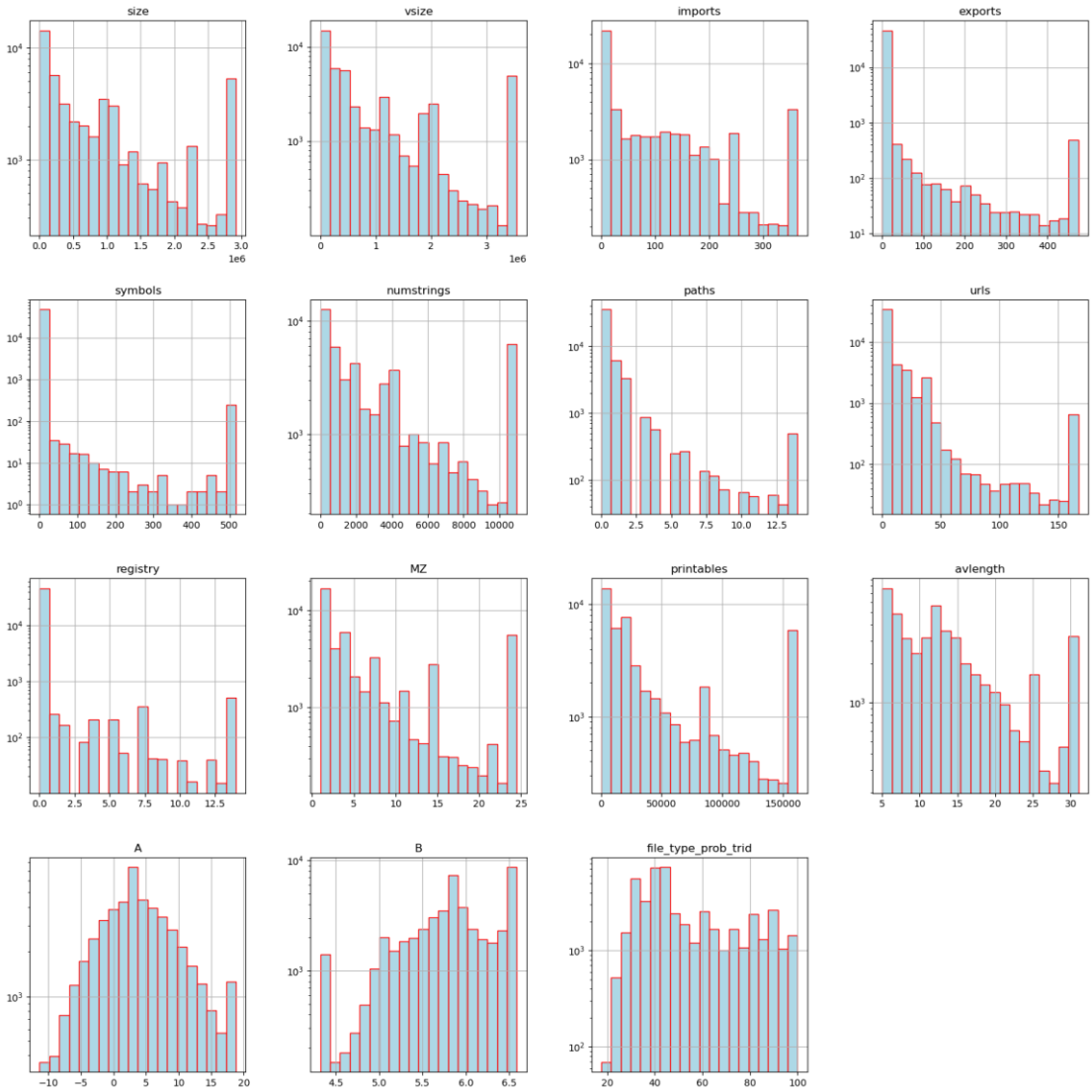
36)


Distribution before Capping
Density
B
Boxplot of B
B
Values

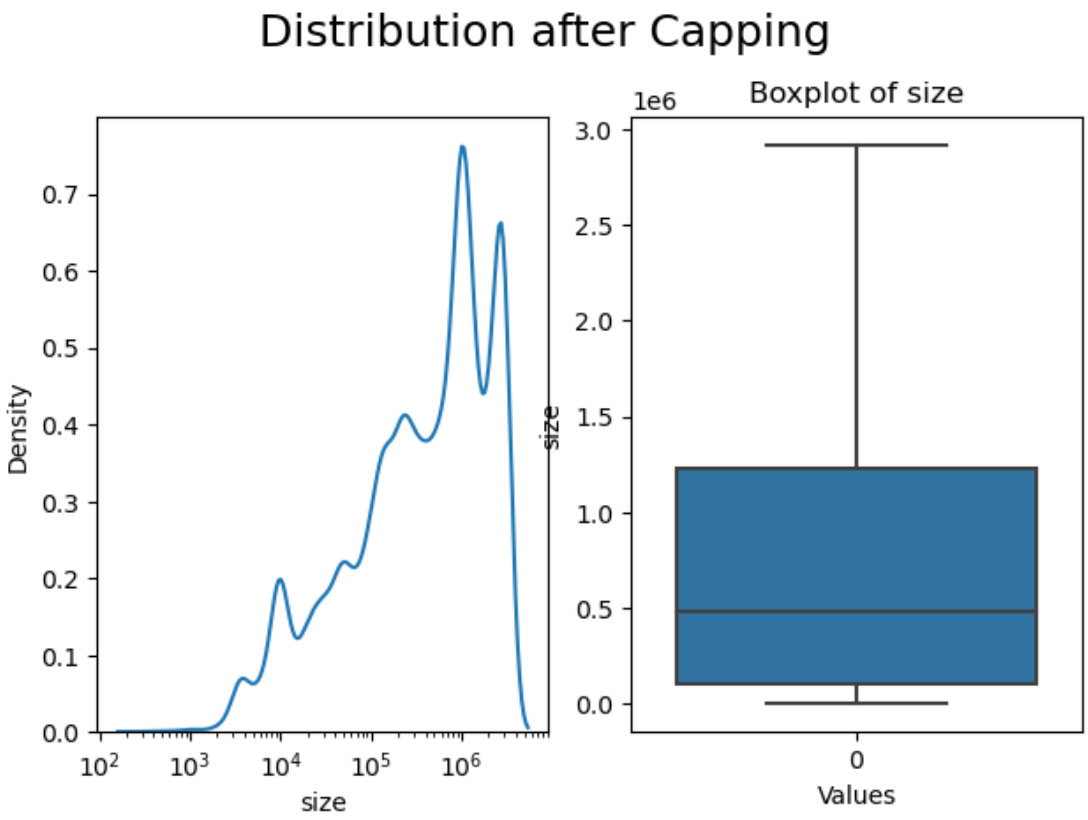
37)


Distribution before Capping
Density
file_type_prob_trid
Boxplot of file_type_prob_trid
file_type_prob_trid
Values

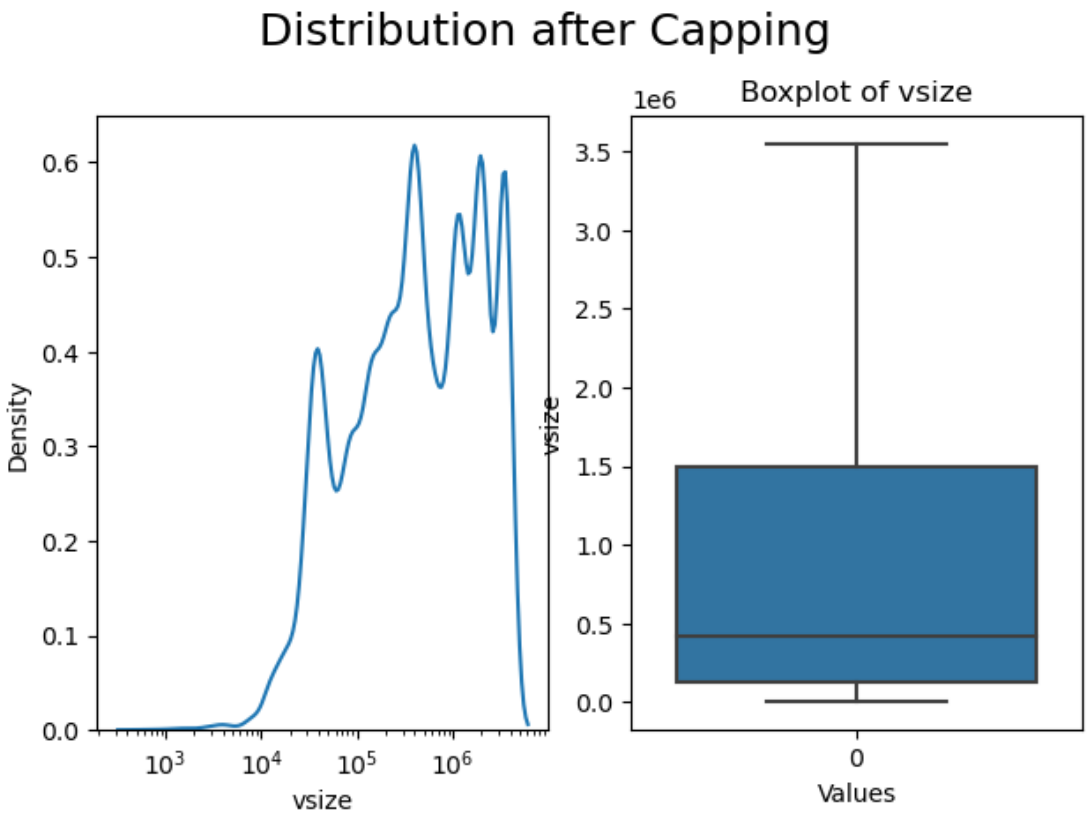
38)


size
vsize
imports
exports
symbols
numstrings
paths
urls
registry
MZ
printables
avlength
A
B
file_type_prob_trid

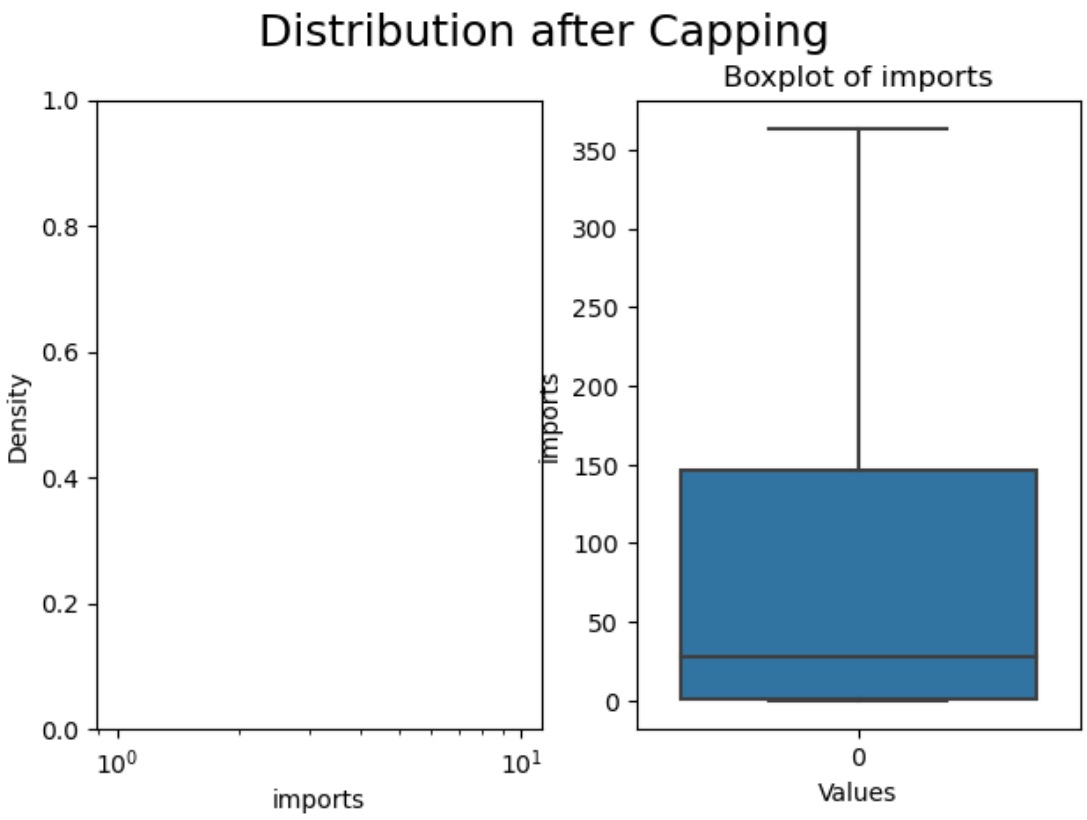
39)


Distribution after Capping
Density
size
Boxplot of size
1e6
size
Values

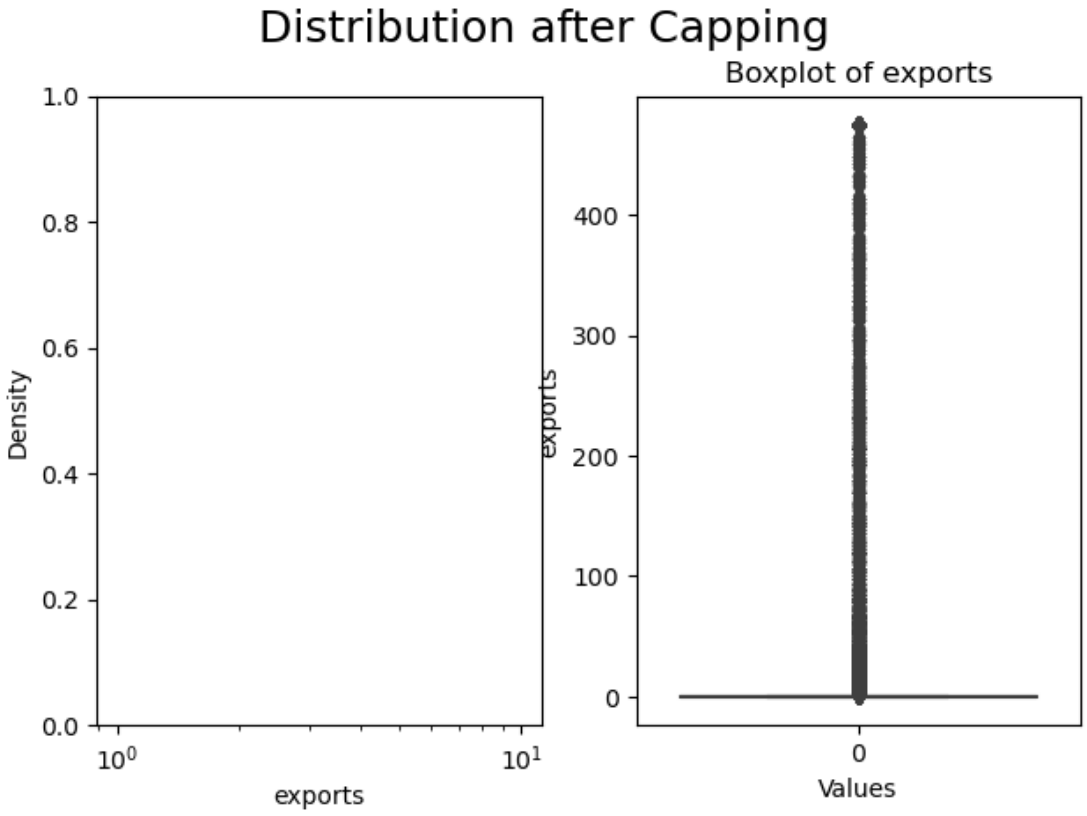
40)


Distribution after Capping
Density
vsize
Boxplot of vsize
1e6
vsize
Values

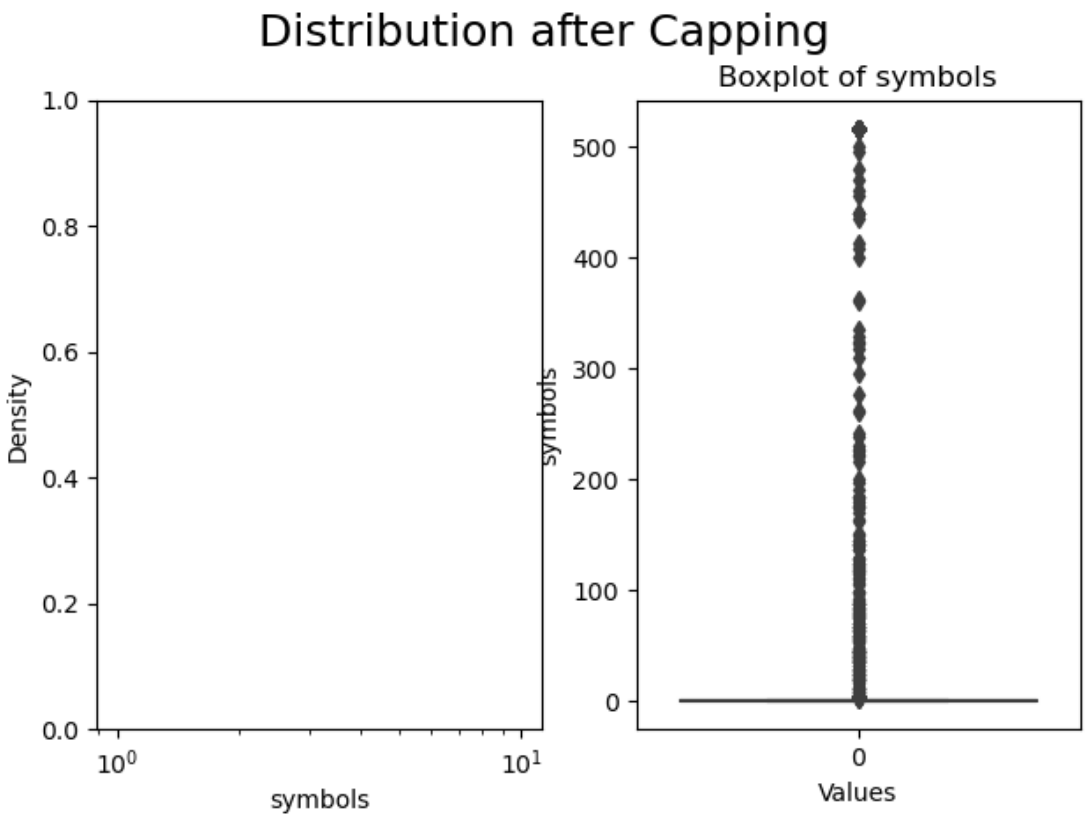
41)


Distribution after Capping
Boxplot of imports
Density
imports
Values
1.0
0.8
0.6
0.4
0.2
0.0
10^0
10^1
350
300
250
200
150
100
50
0
0

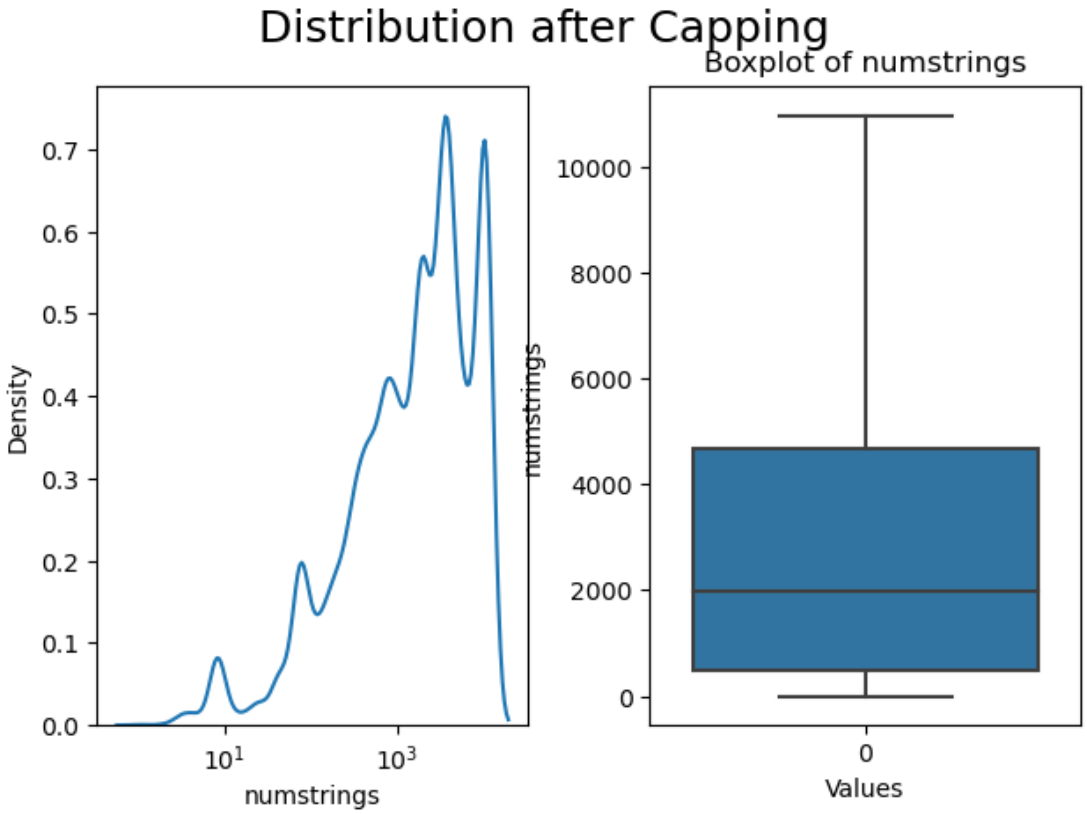
42)


Distribution after Capping
Boxplot of exports
Density
exports
Values
1.0
0.8
0.6
0.4
0.2
0.0
10^0
10^1
400
300
200
100
0
0

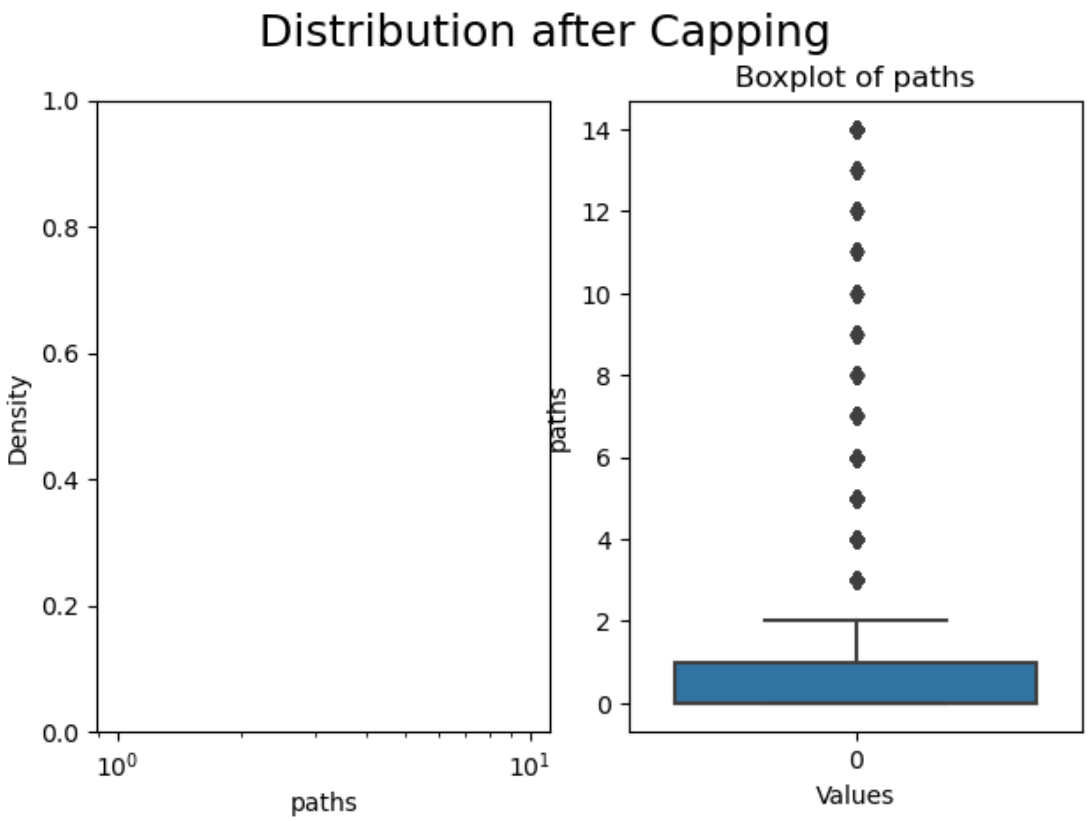
43)


Distribution after Capping
Boxplot of symbols
Density
symbols
symbols
Values

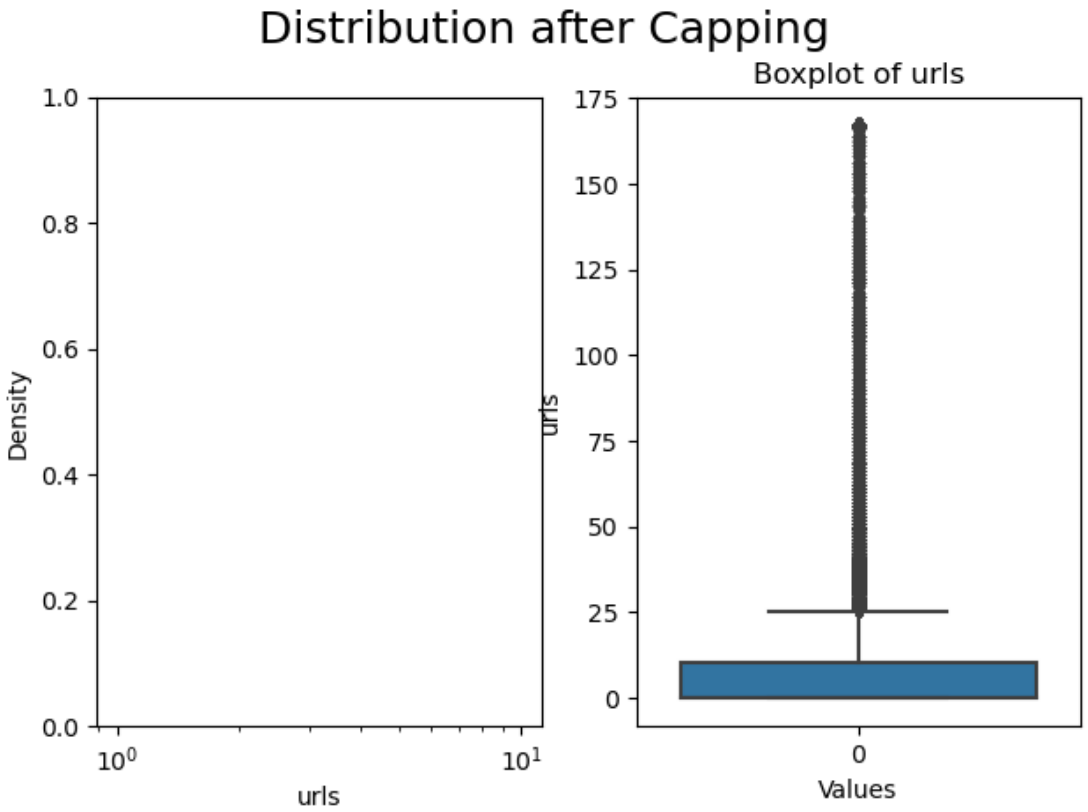
44)


Distribution after Capping
Boxplot of numstrings
Density
numstrings
numstrings
Values

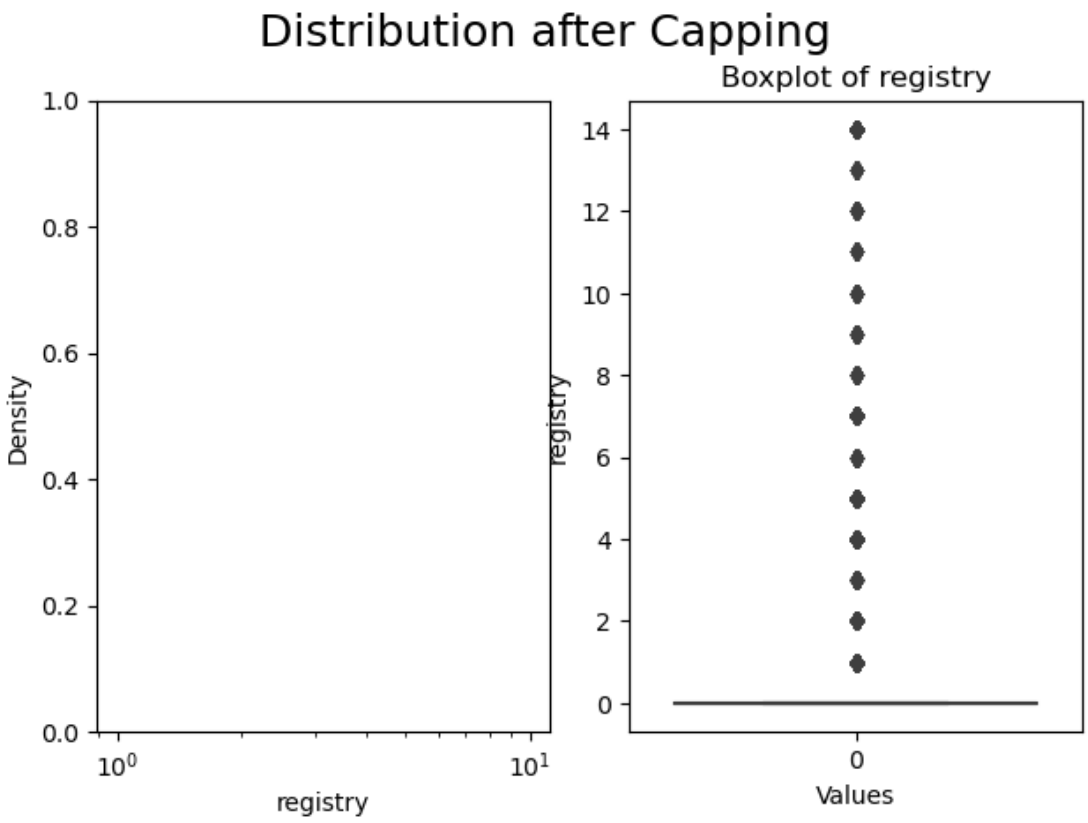
45)


Distribution after Capping
Boxplot of paths
Density
paths
Values

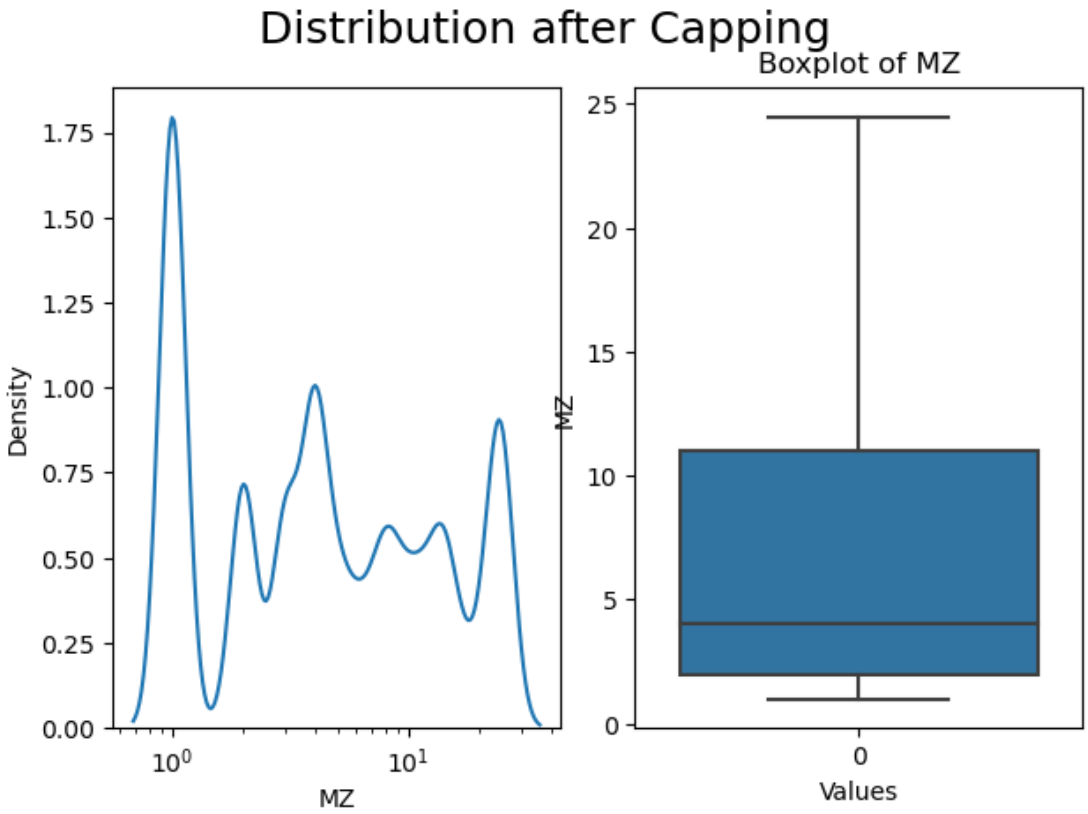
46)


Distribution after Capping
Boxplot of urls
Density
urls
Values

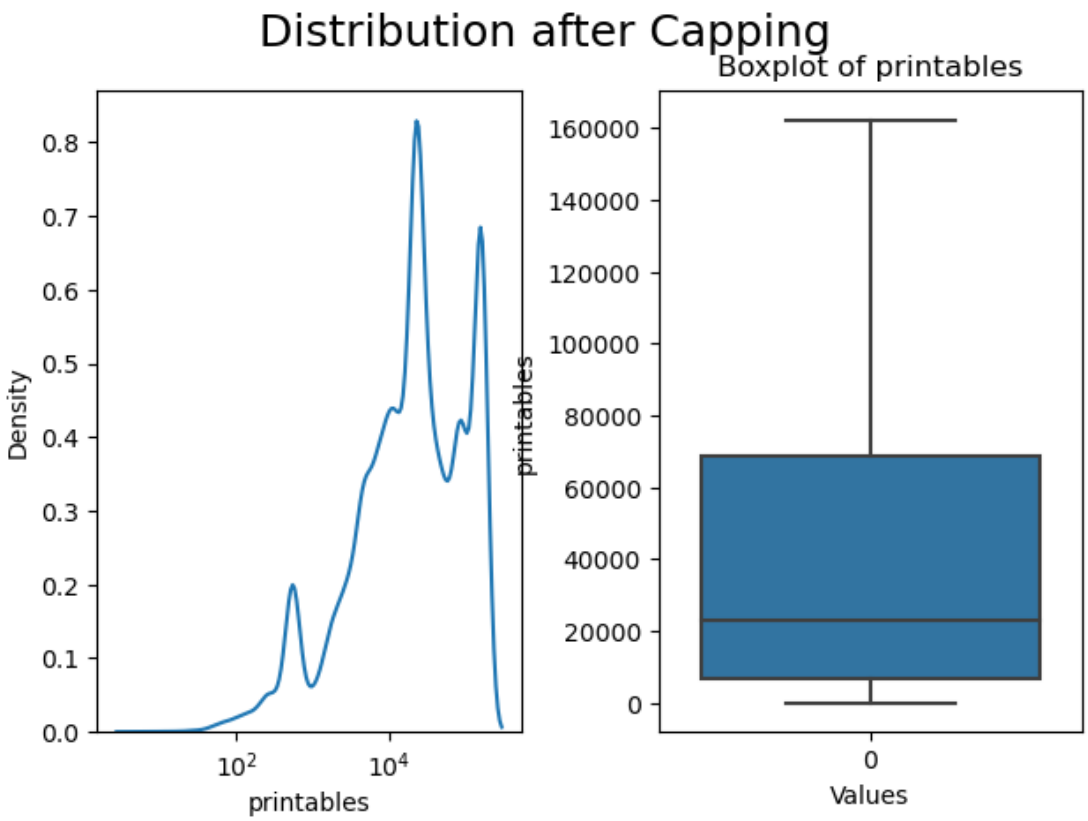
47)


Distribution after Capping
Boxplot of registry
Density
registry
registry
Values

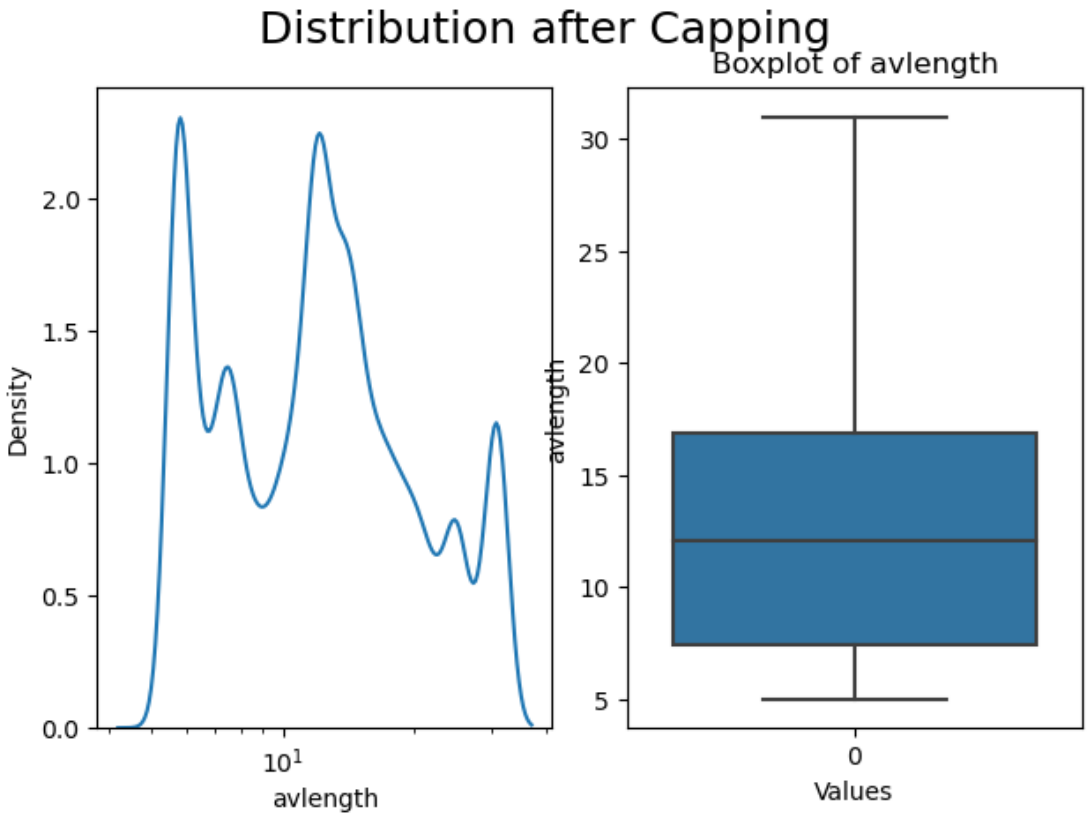
48)


Distribution after Capping
Boxplot of MZ
Density
MZ
MZ
Values

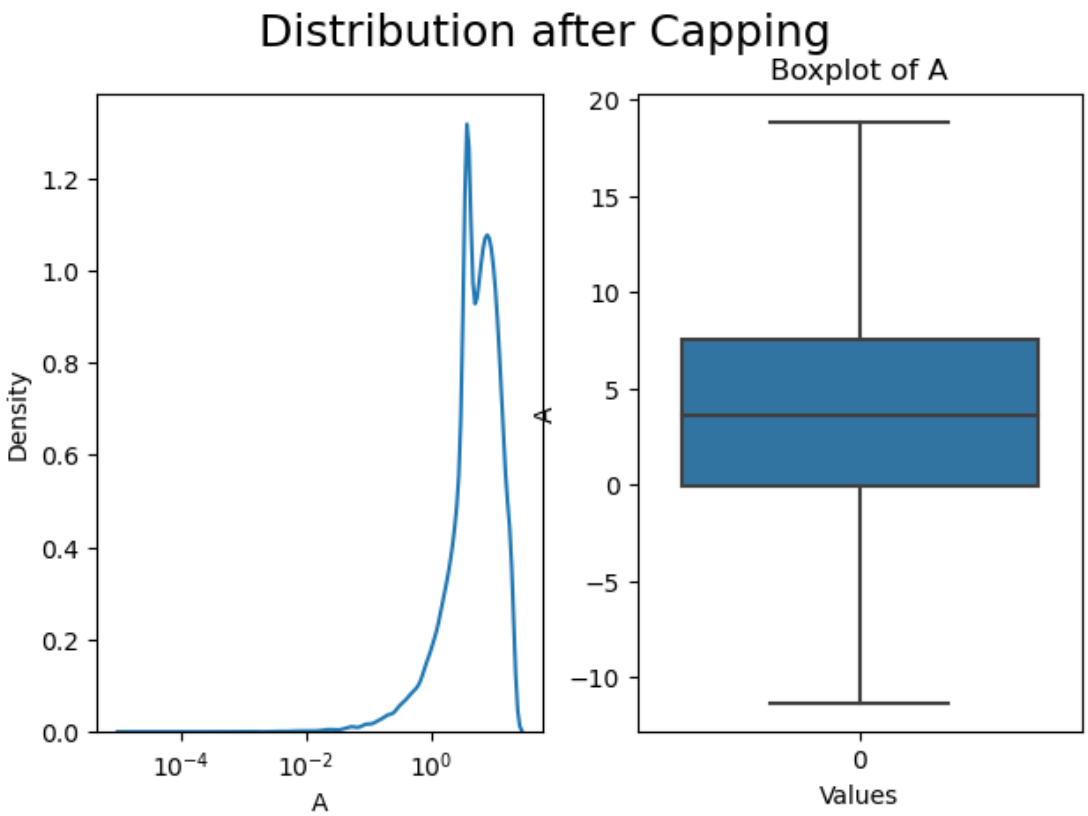
49)


Distribution after Capping
Boxplot of printables
Density
printables
Values

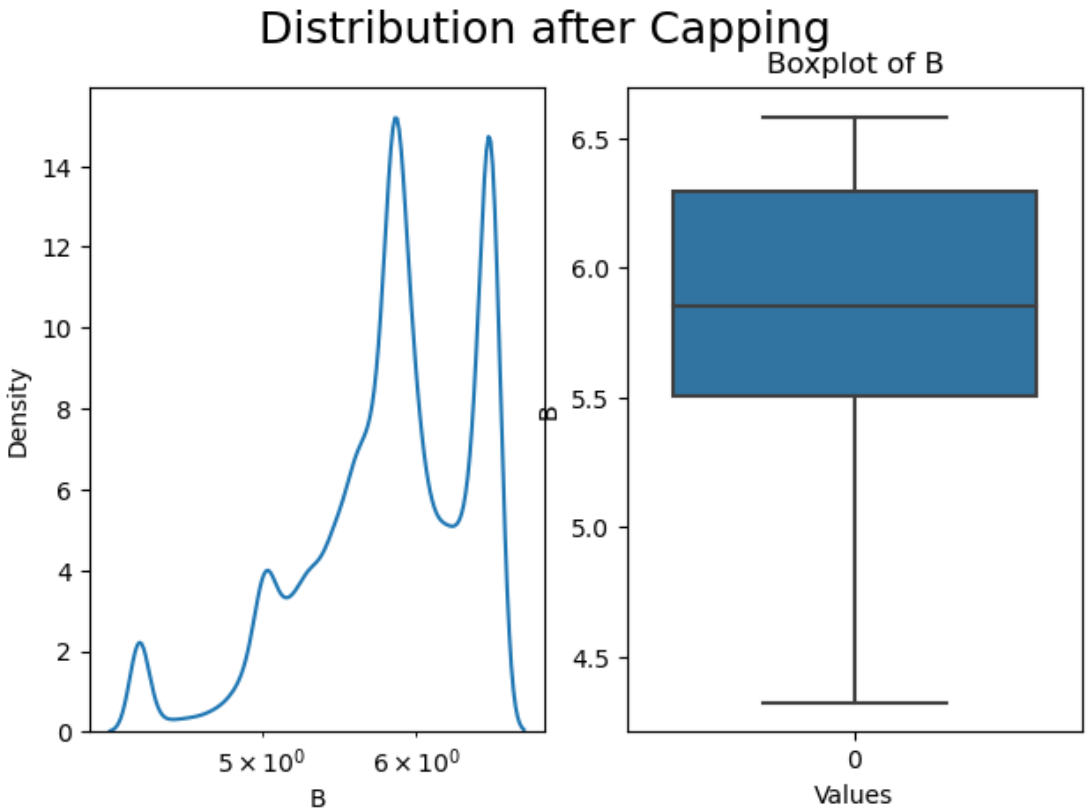
50)


Distribution after Capping
Boxplot of avlength
Density
avlength
Values

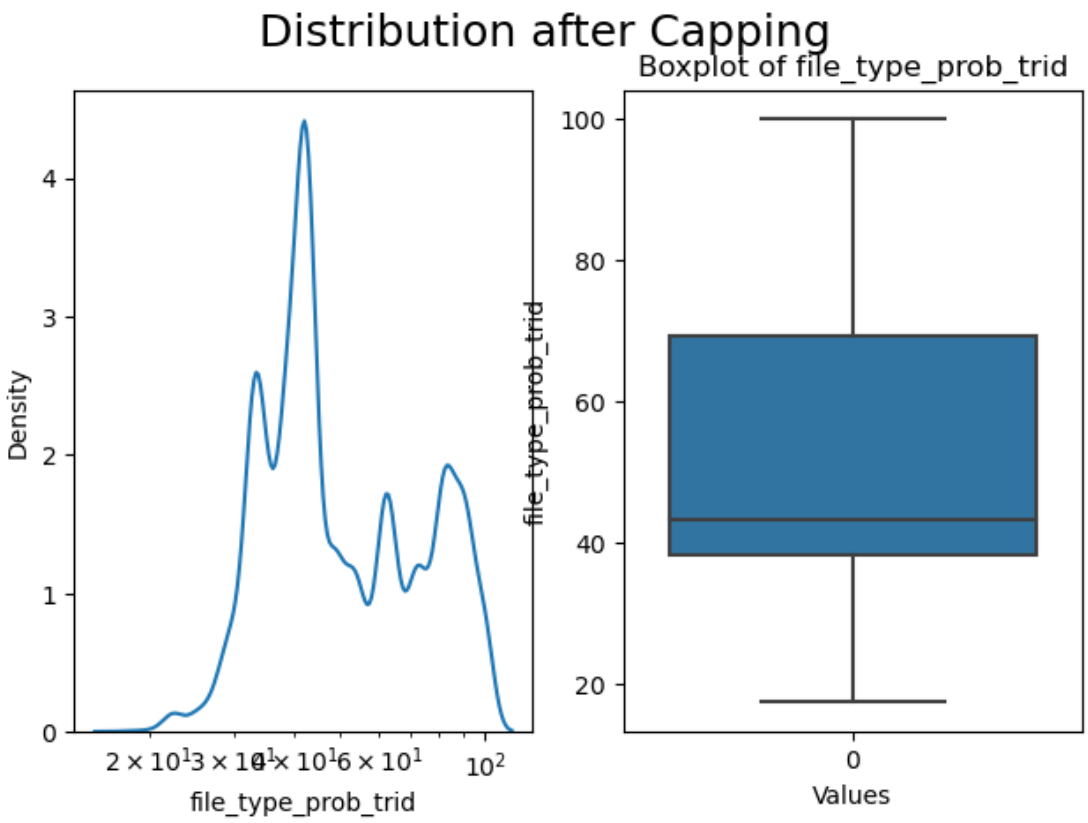
51)


Distribution after Capping
Boxplot of A
Density
A
Values

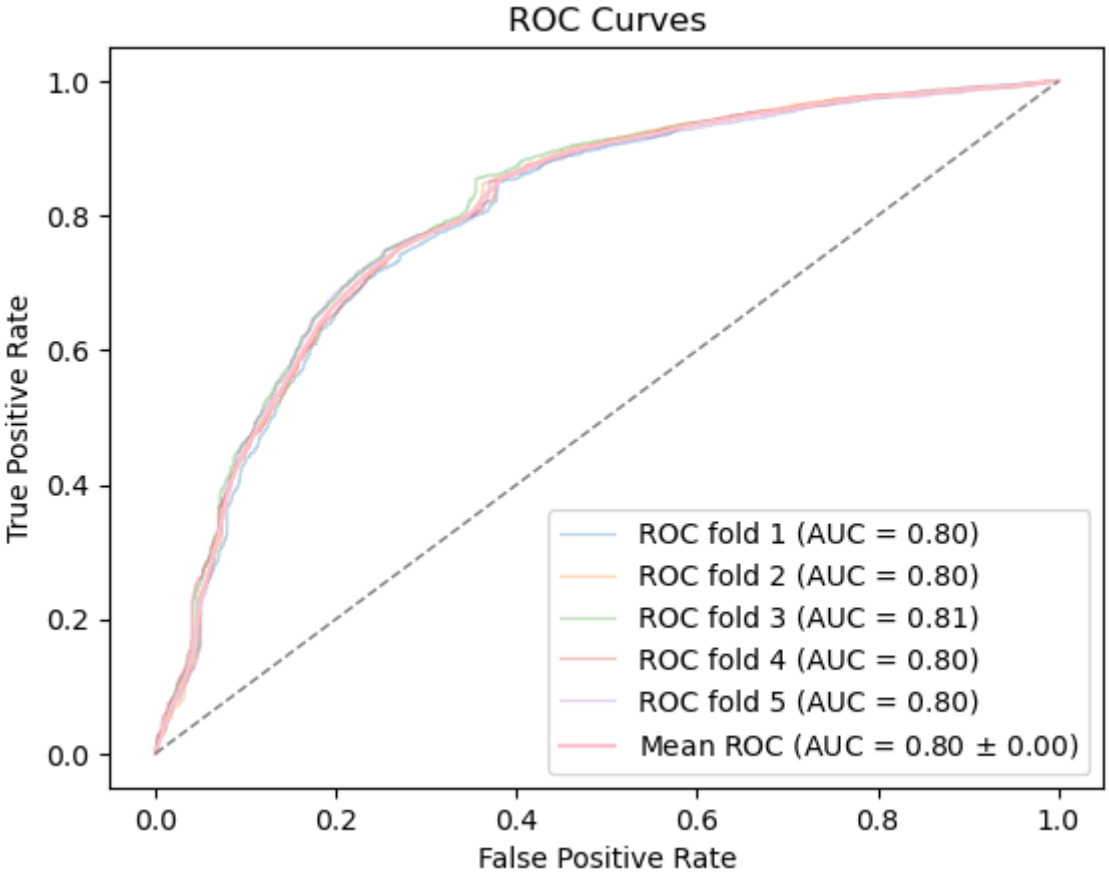
52)


Distribution after Capping
Boxplot of B
Density
B
Values

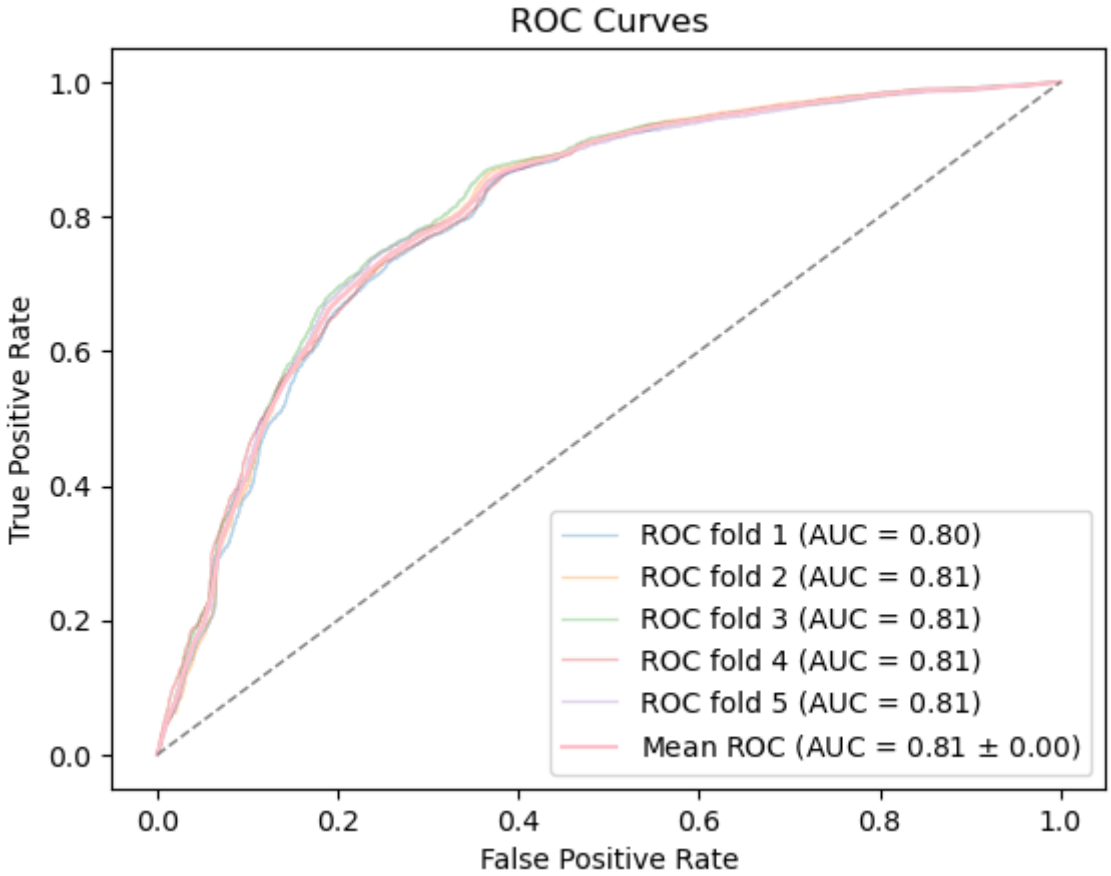
53)



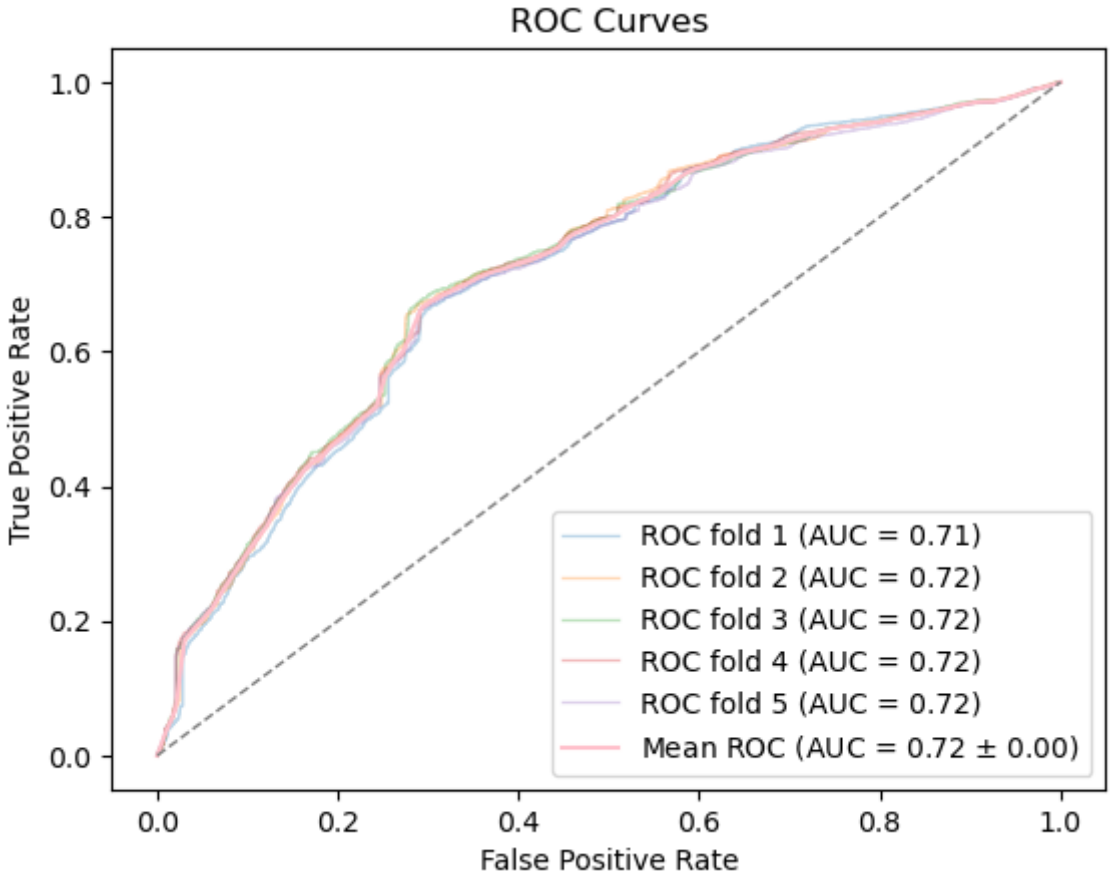
54)רגרסיה לוגיסטית 1



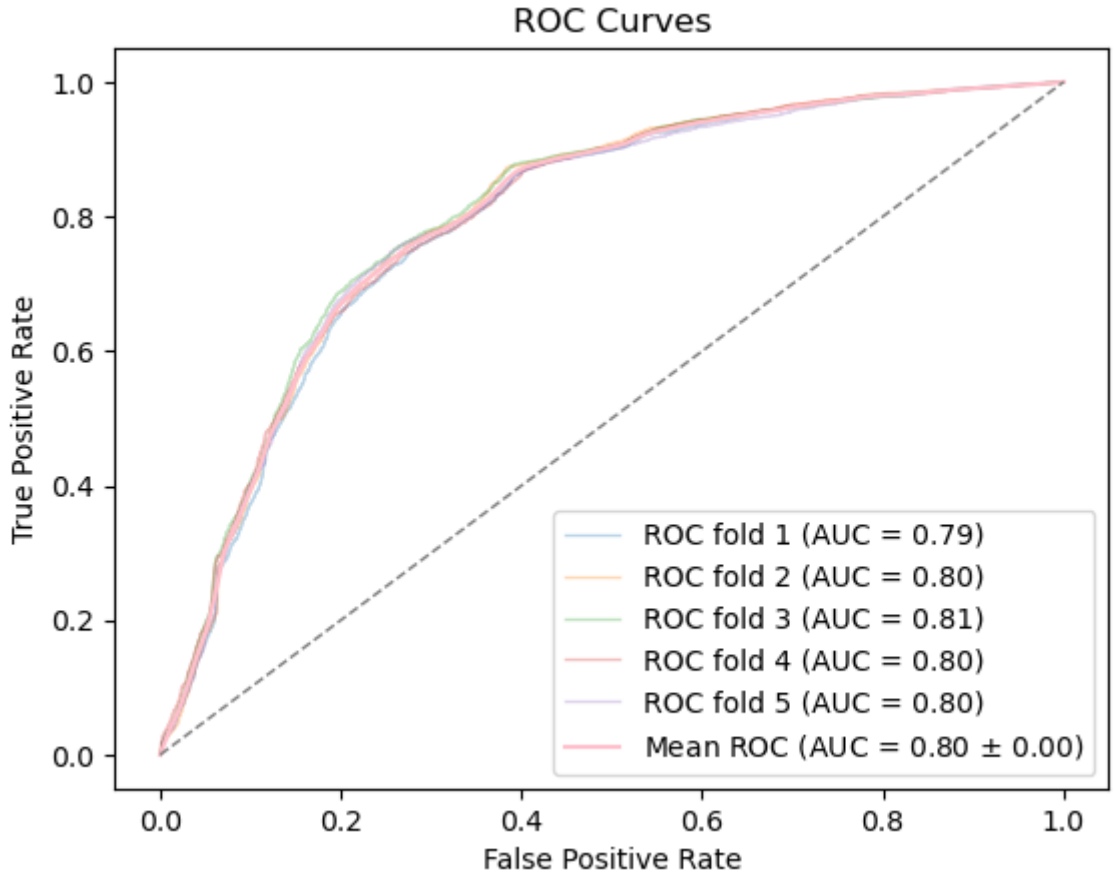
55) 2:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.80)
ROC fold 2 (AUC = 0.81)
ROC fold 3 (AUC = 0.81)
ROC fold 4 (AUC = 0.81)
ROC fold 5 (AUC = 0.81)
Mean ROC (AUC = 0.81 ± 0.00)

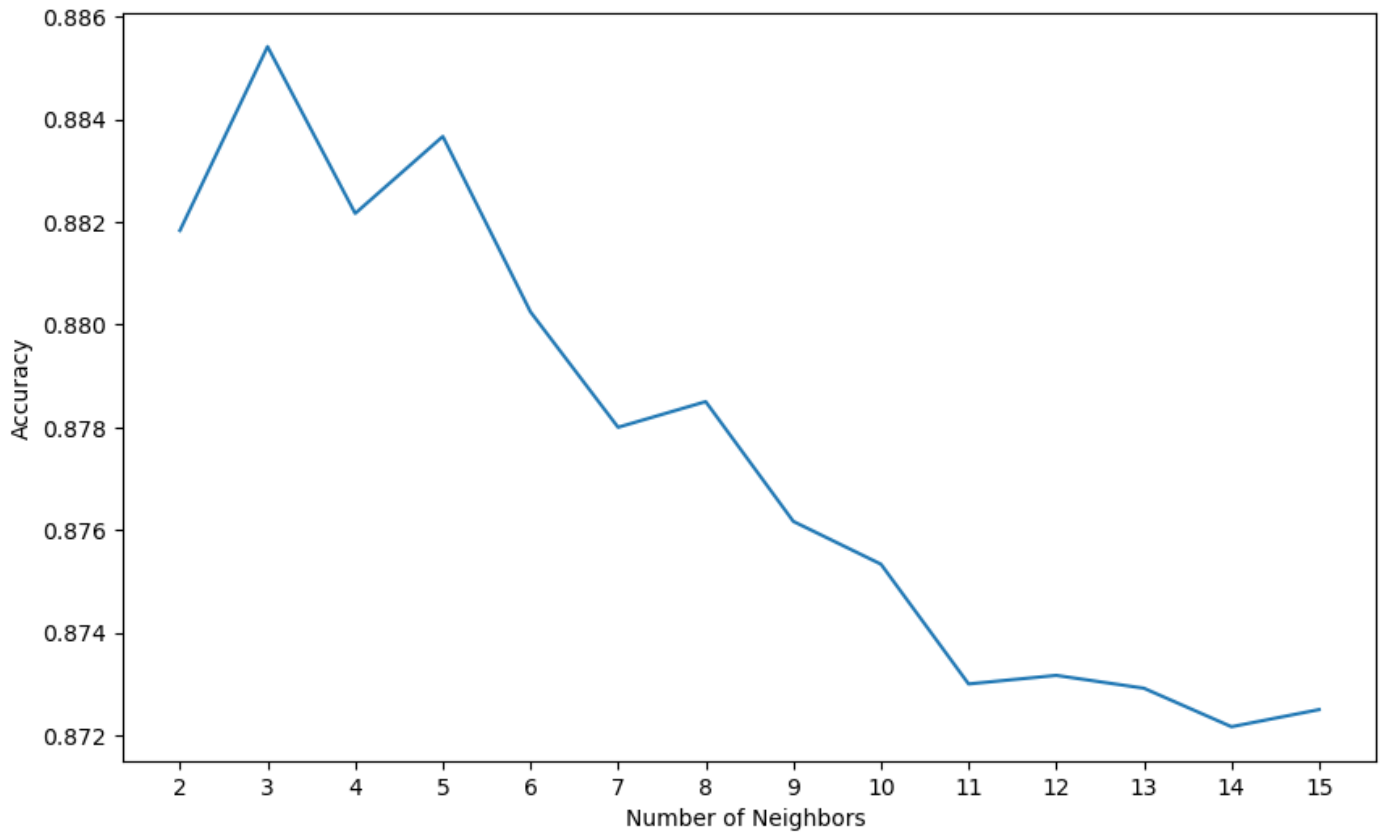
56: 3:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.71)
ROC fold 2 (AUC = 0.72)
ROC fold 3 (AUC = 0.72)
ROC fold 4 (AUC = 0.72)
ROC fold 5 (AUC = 0.72)
Mean ROC (AUC = 0.72 ± 0.00)

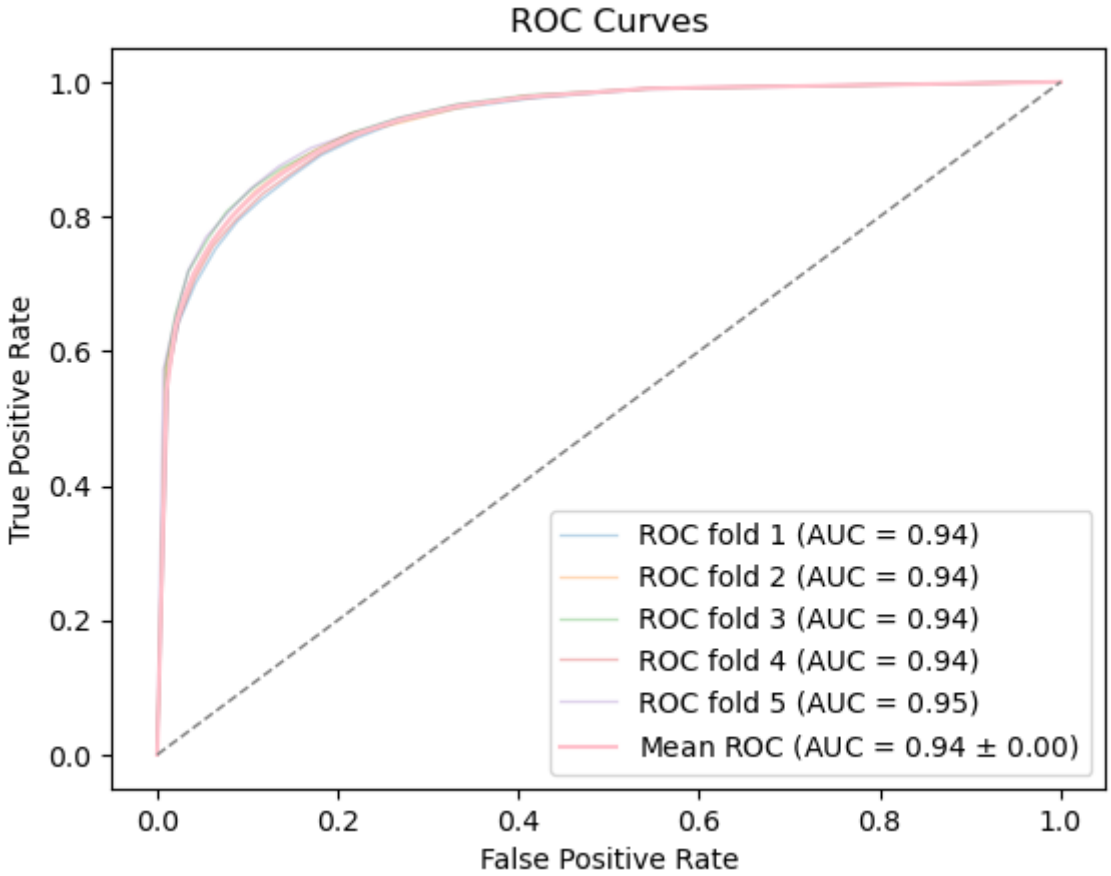
57: 4:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.79)
ROC fold 2 (AUC = 0.80)
ROC fold 3 (AUC = 0.81)
ROC fold 4 (AUC = 0.80)
ROC fold 5 (AUC = 0.80)
Mean ROC (AUC = 0.80 ± 0.00)

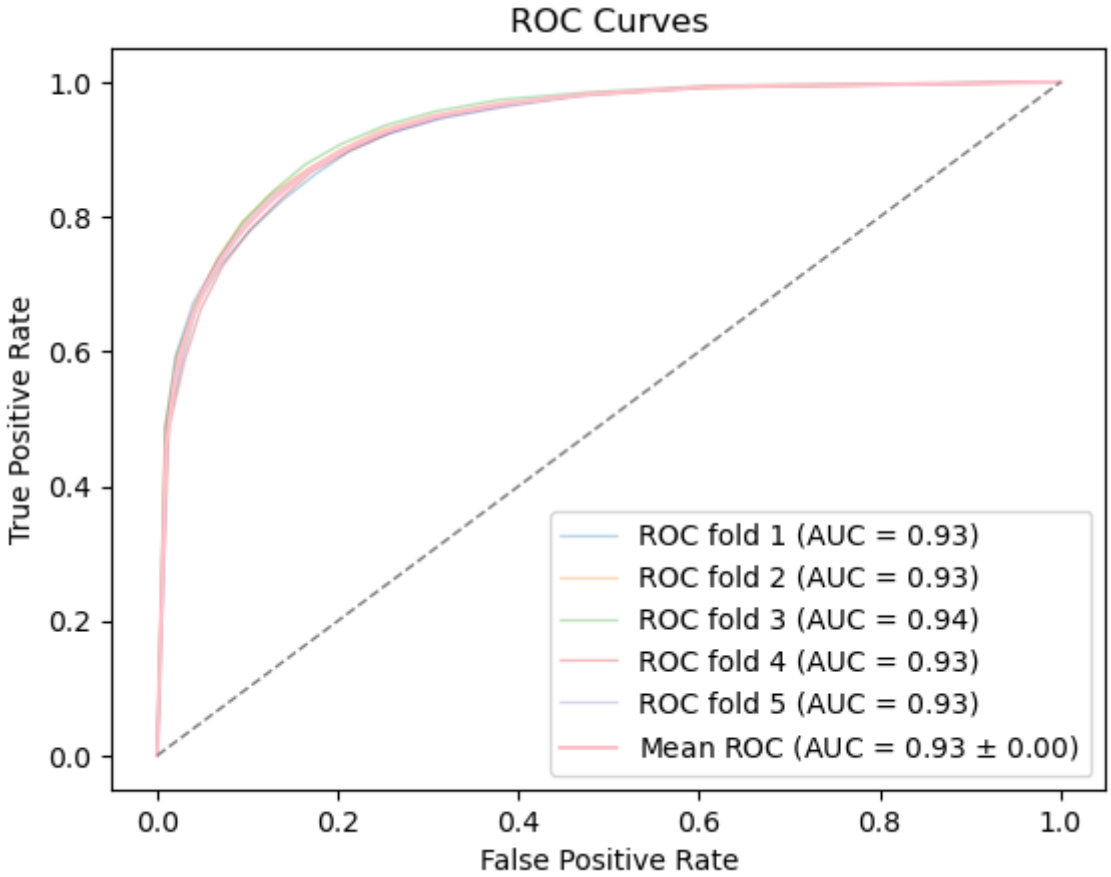
58: KNN 1:


Accuracy
Number of Neighbors
0.872
0.874
0.876
0.878
0.880
0.882
0.884
0.886
2
3
4
5
6
7
8
9
10
11
12
13
14
15

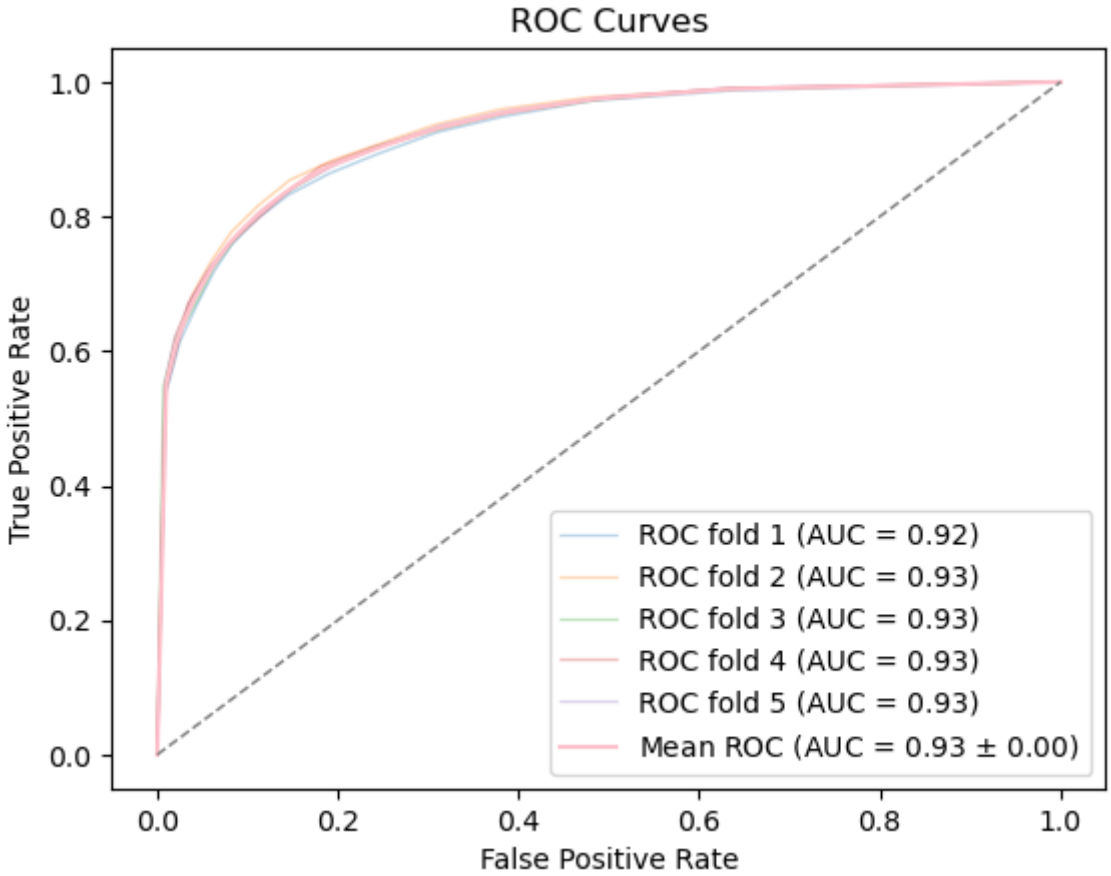
59: 1:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.94)
ROC fold 2 (AUC = 0.94)
ROC fold 3 (AUC = 0.94)
ROC fold 4 (AUC = 0.94)
ROC fold 5 (AUC = 0.95)
Mean ROC (AUC = 0.94 ± 0.00)

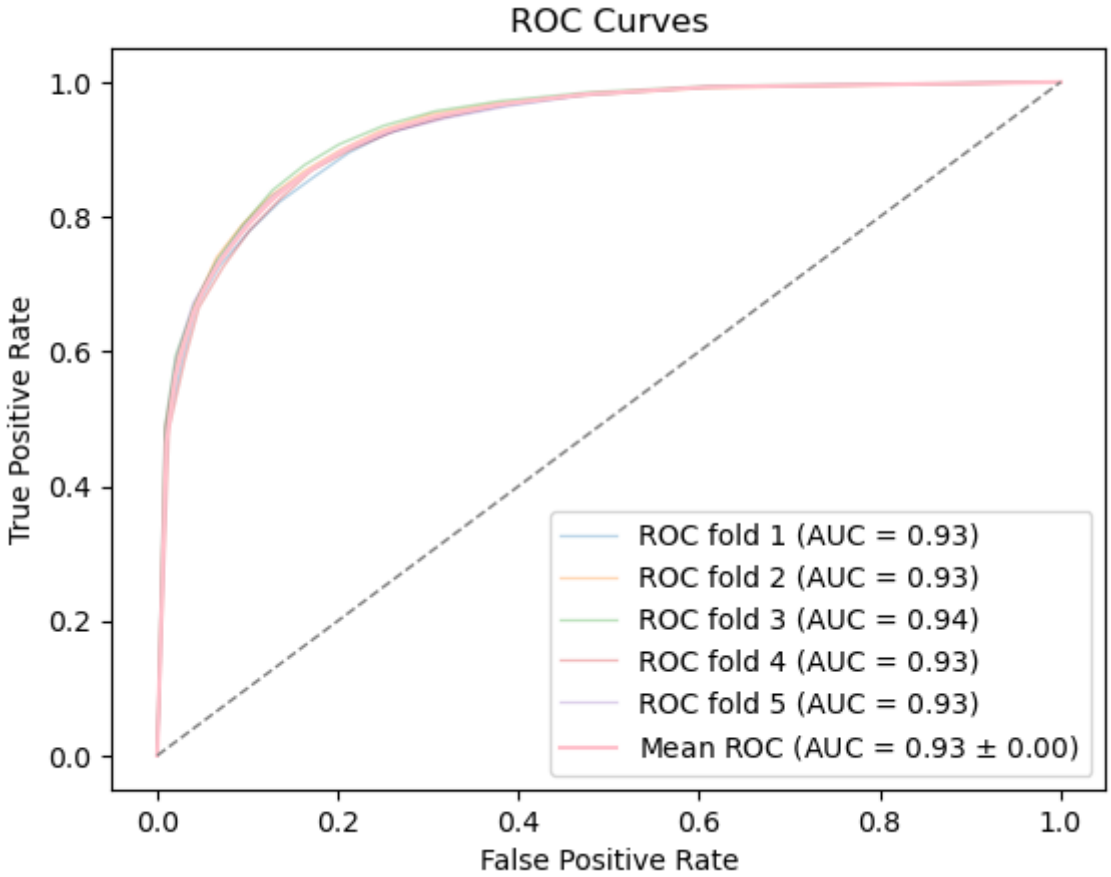
60: 2:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.93)
ROC fold 2 (AUC = 0.93)
ROC fold 3 (AUC = 0.94)
ROC fold 4 (AUC = 0.93)
ROC fold 5 (AUC = 0.93)
Mean ROC (AUC = 0.93 ± 0.00)

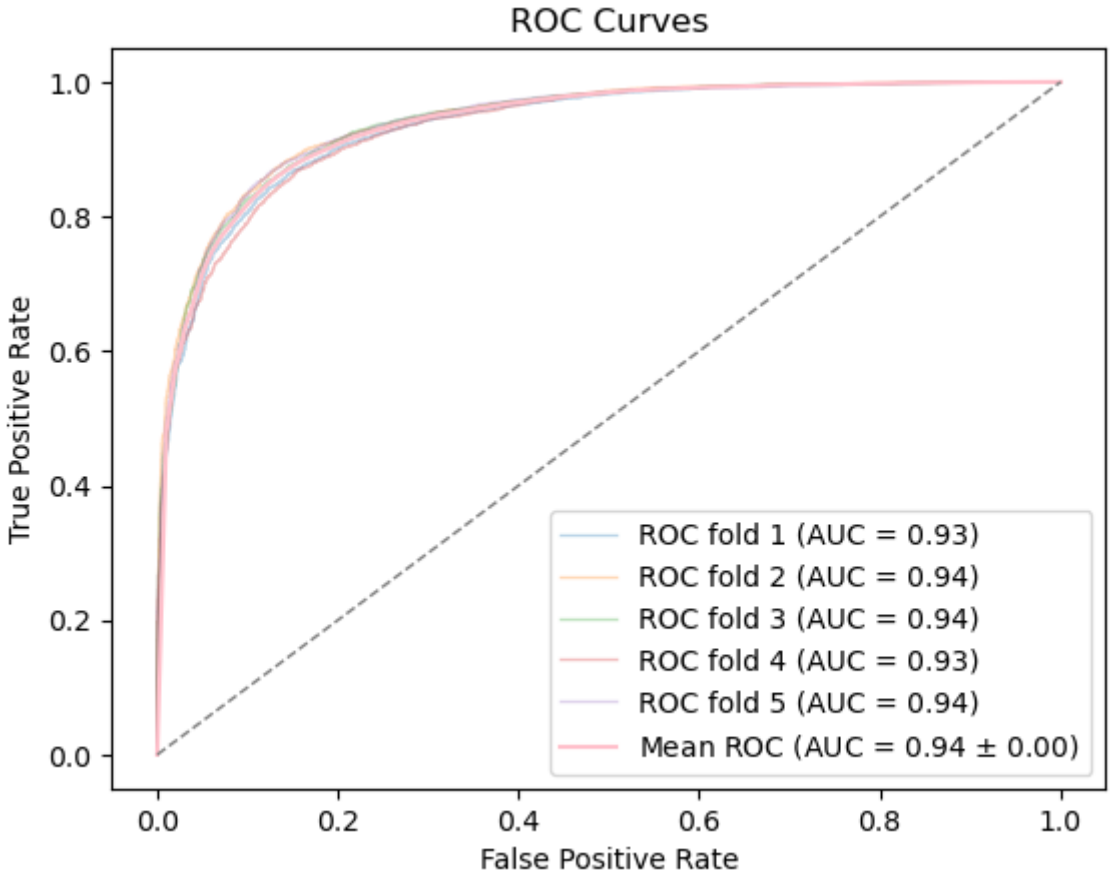
61: 3:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.92)
ROC fold 2 (AUC = 0.93)
ROC fold 3 (AUC = 0.93)
ROC fold 4 (AUC = 0.93)
ROC fold 5 (AUC = 0.93)
Mean ROC (AUC = 0.93 ± 0.00)

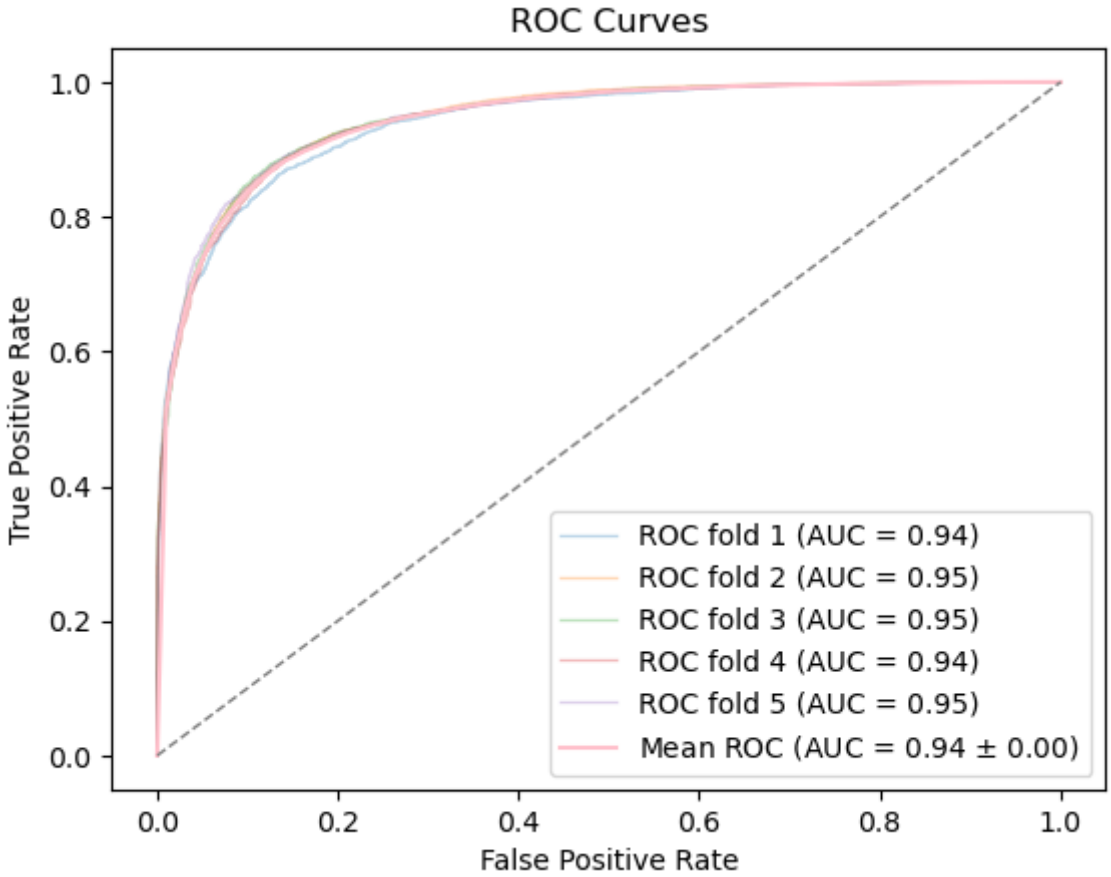
62: 4:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.93)
ROC fold 2 (AUC = 0.93)
ROC fold 3 (AUC = 0.94)
ROC fold 4 (AUC = 0.93)
ROC fold 5 (AUC = 0.93)
Mean ROC (AUC = 0.93 ± 0.00)

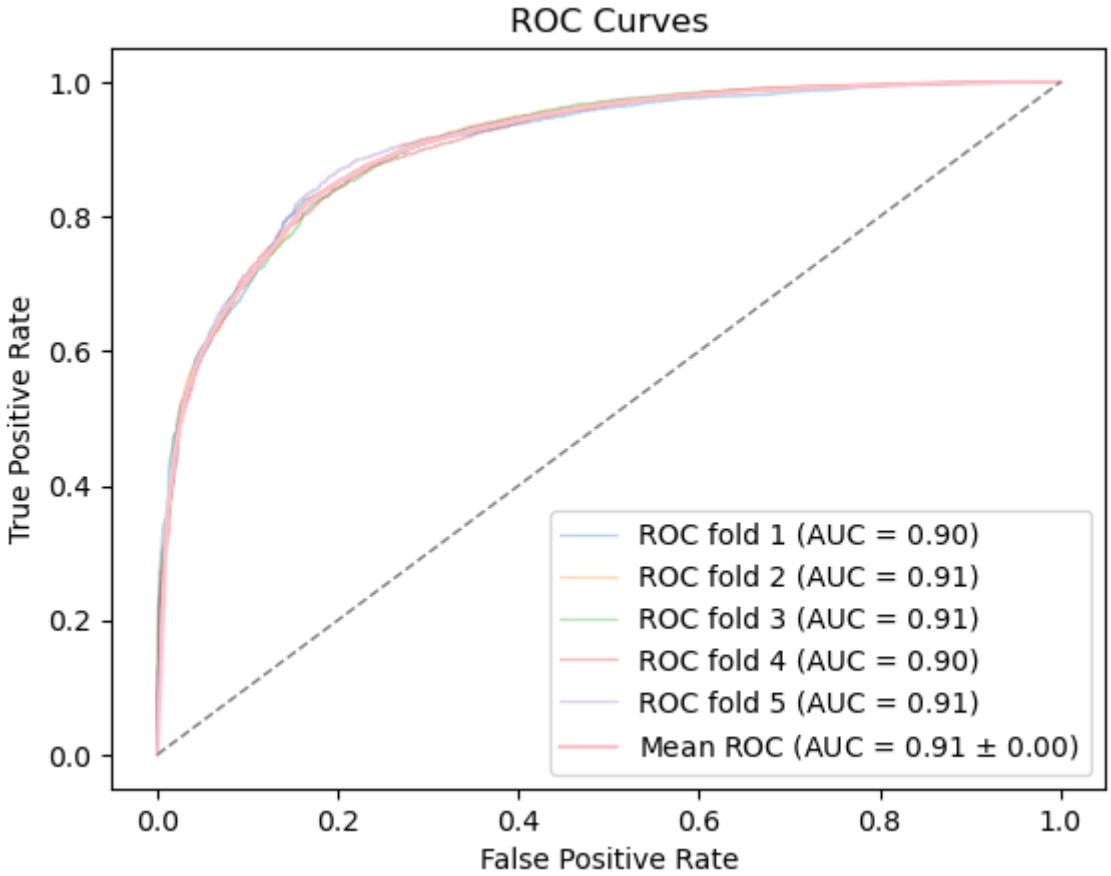
63: mlp 1


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.93)
ROC fold 2 (AUC = 0.94)
ROC fold 3 (AUC = 0.94)
ROC fold 4 (AUC = 0.93)
ROC fold 5 (AUC = 0.94)
Mean ROC (AUC = 0.94 ± 0.00)

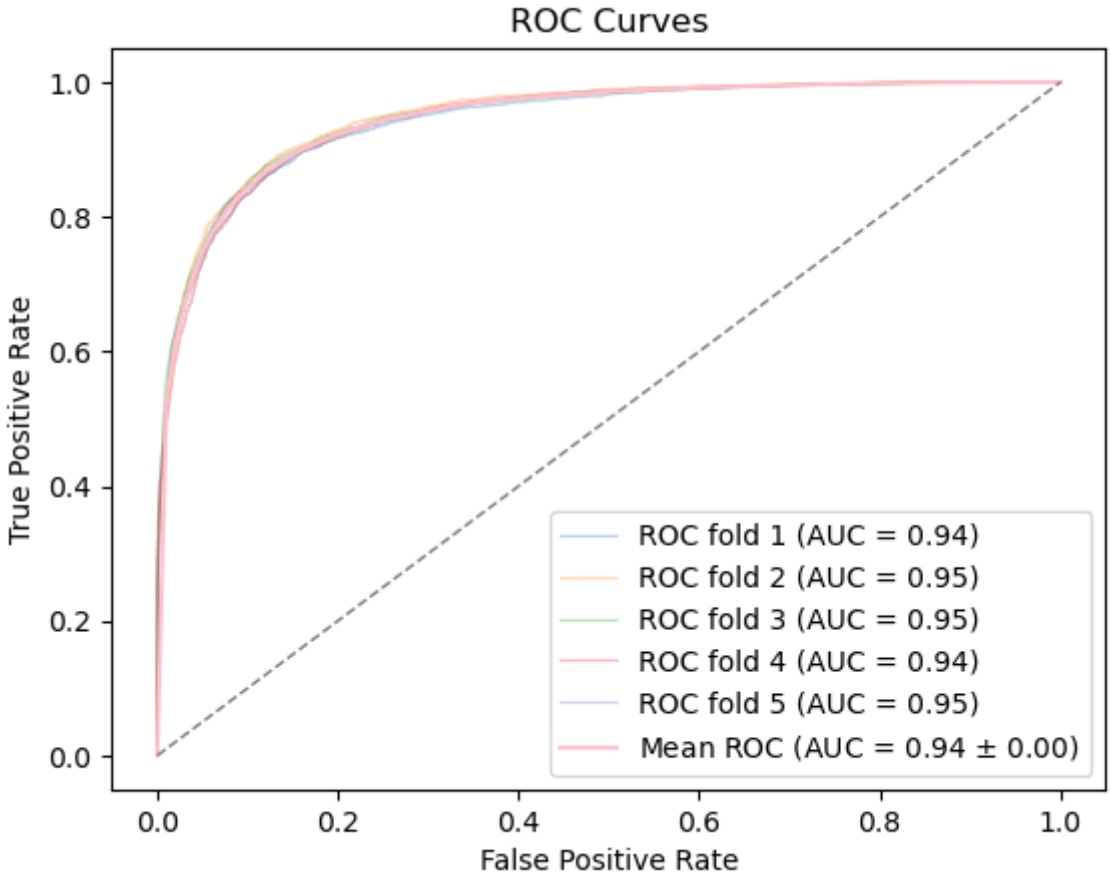
64: 2:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.94)
ROC fold 2 (AUC = 0.95)
ROC fold 3 (AUC = 0.95)
ROC fold 4 (AUC = 0.94)
ROC fold 5 (AUC = 0.95)
Mean ROC (AUC = 0.94 ± 0.00)

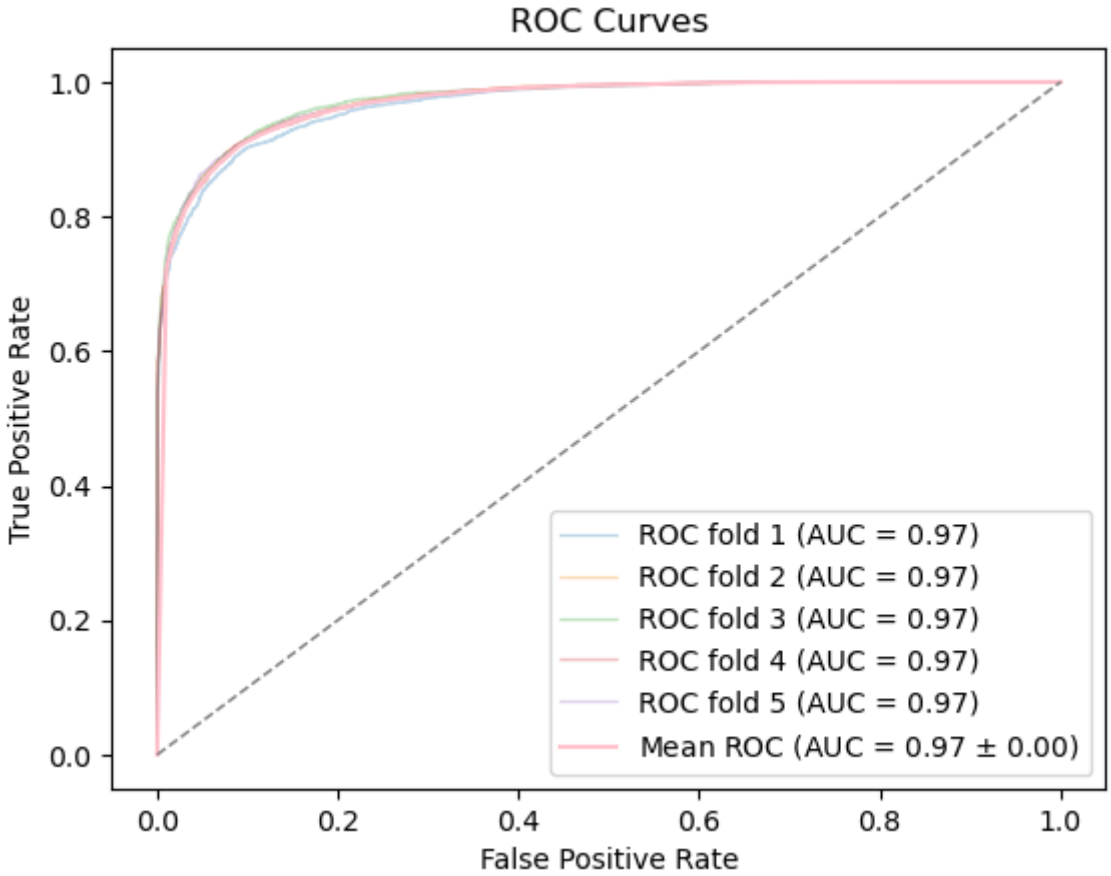
65: 3:



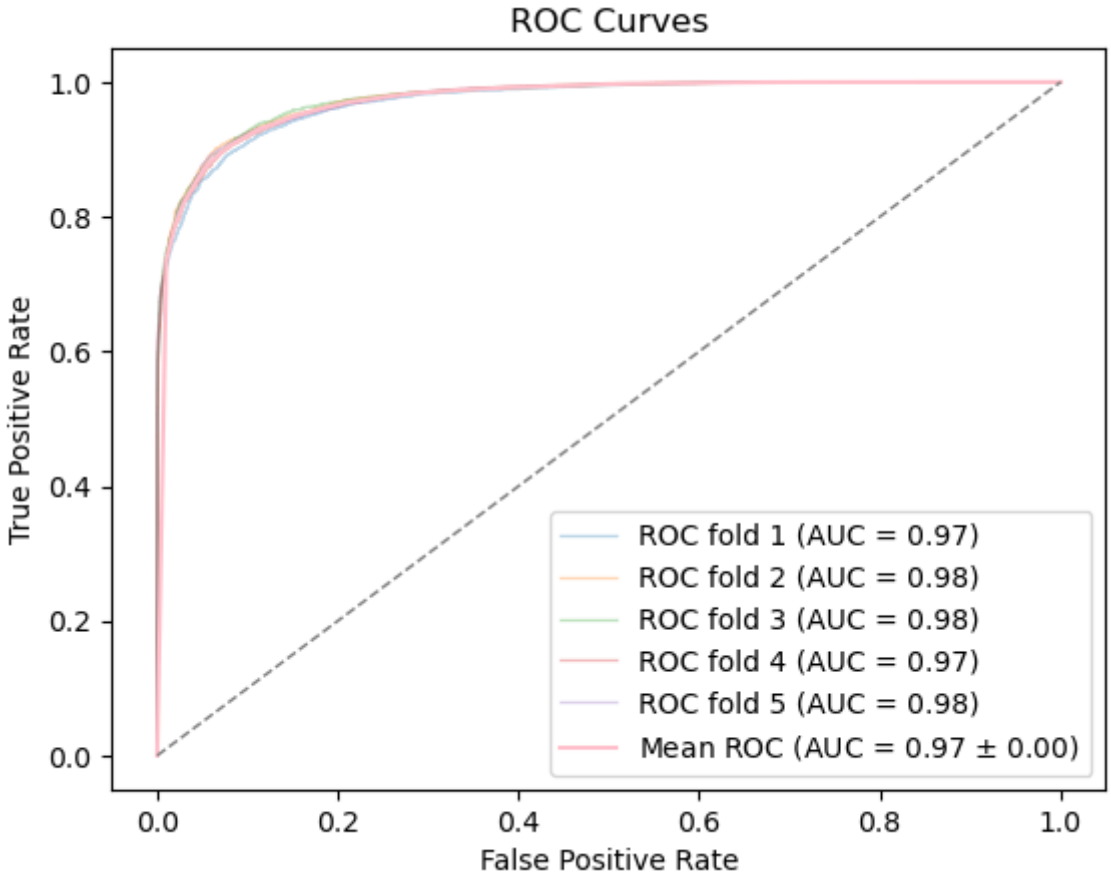
66: 4:



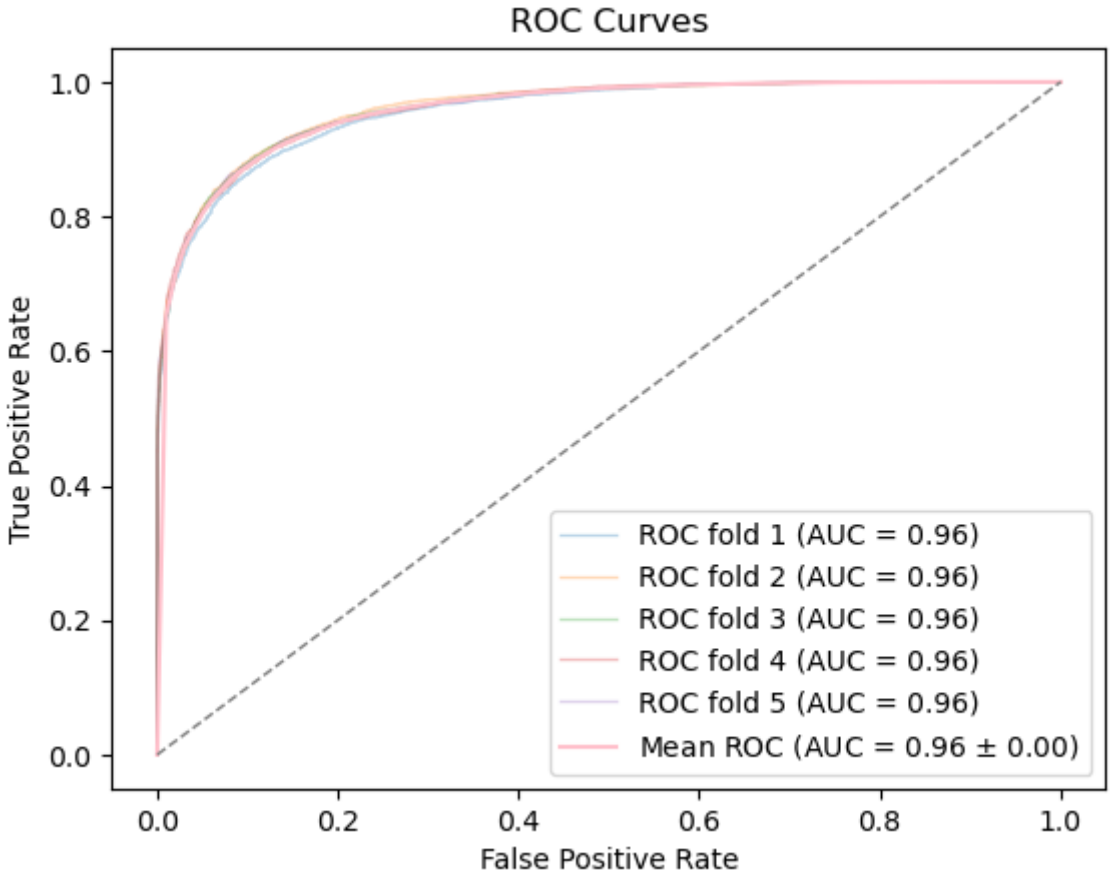
67: עץ רנדומלי 1:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.97)
ROC fold 2 (AUC = 0.97)
ROC fold 3 (AUC = 0.97)
ROC fold 4 (AUC = 0.97)
ROC fold 5 (AUC = 0.97)
Mean ROC (AUC = 0.97 ± 0.00)

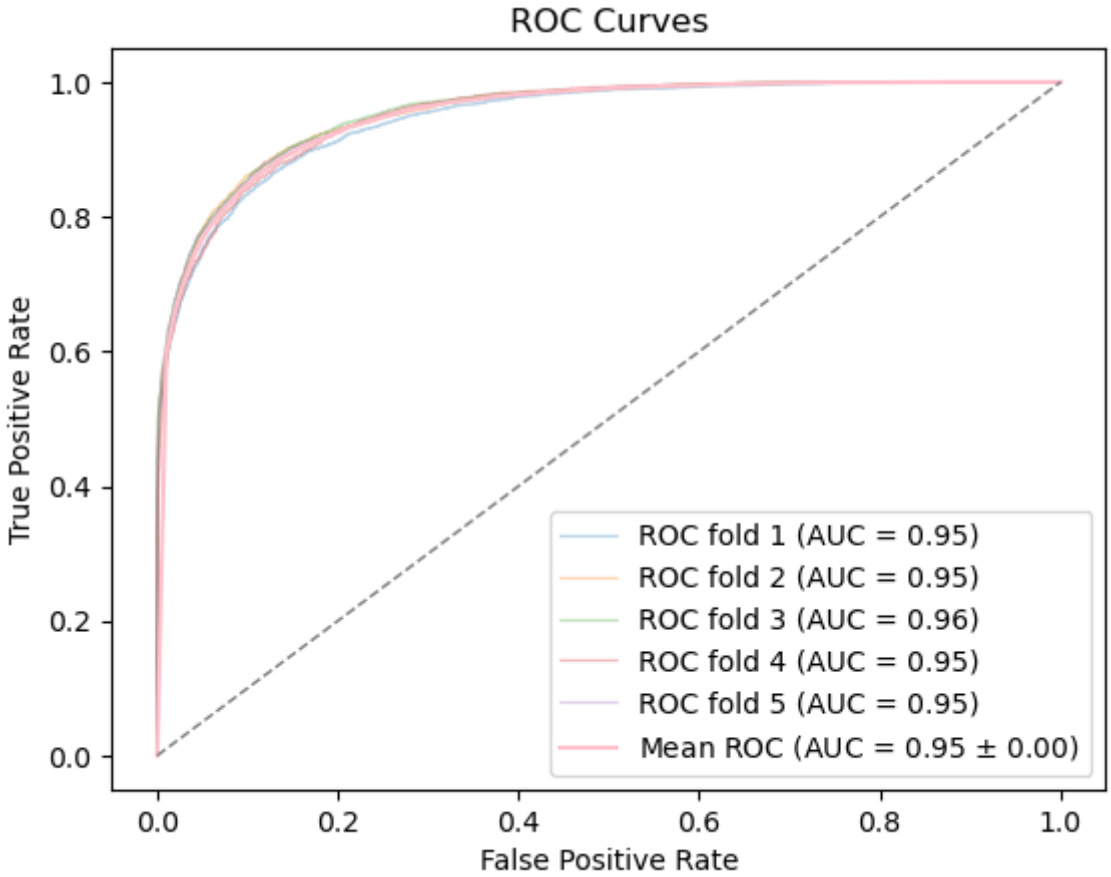
68: 2:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.97)
ROC fold 2 (AUC = 0.98)
ROC fold 3 (AUC = 0.98)
ROC fold 4 (AUC = 0.97)
ROC fold 5 (AUC = 0.98)
Mean ROC (AUC = 0.97 ± 0.00)

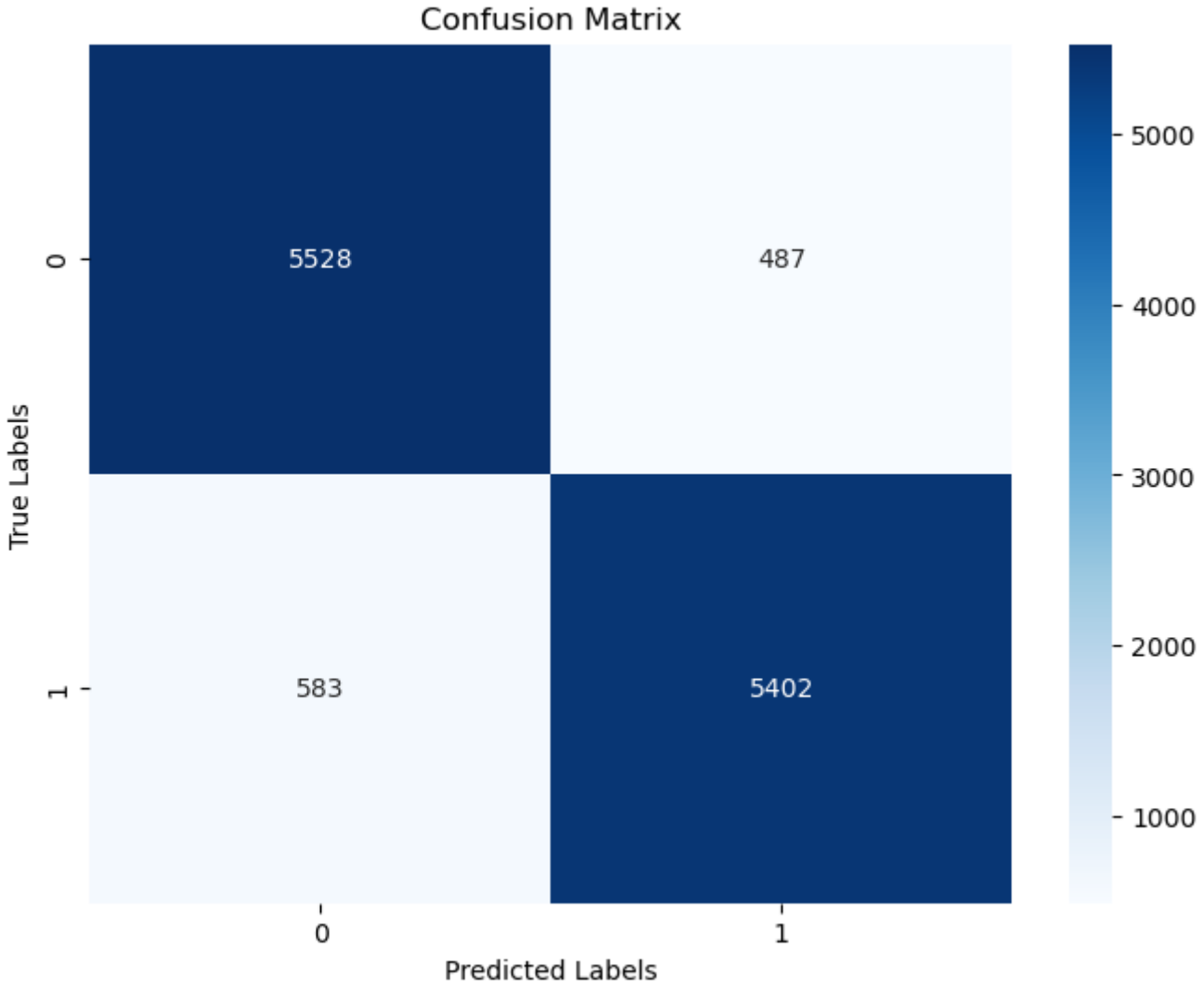
69: 3:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.96)
ROC fold 2 (AUC = 0.96)
ROC fold 3 (AUC = 0.96)
ROC fold 4 (AUC = 0.96)
ROC fold 5 (AUC = 0.96)
Mean ROC (AUC = 0.96 ± 0.00)

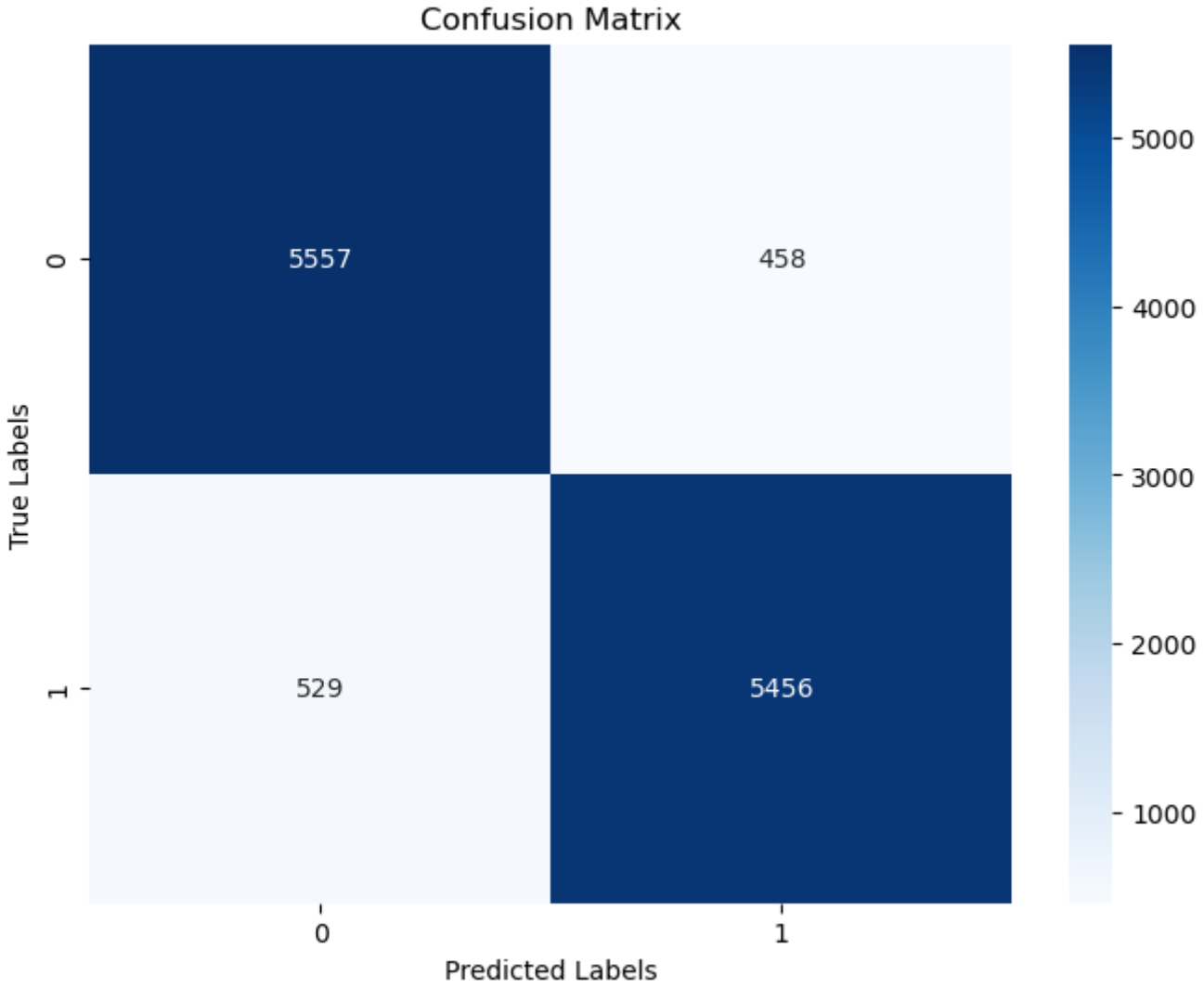
70: 4:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.95)
ROC fold 2 (AUC = 0.95)
ROC fold 3 (AUC = 0.96)
ROC fold 4 (AUC = 0.95)
ROC fold 5 (AUC = 0.95)
Mean ROC (AUC = 0.95 ± 0.00)

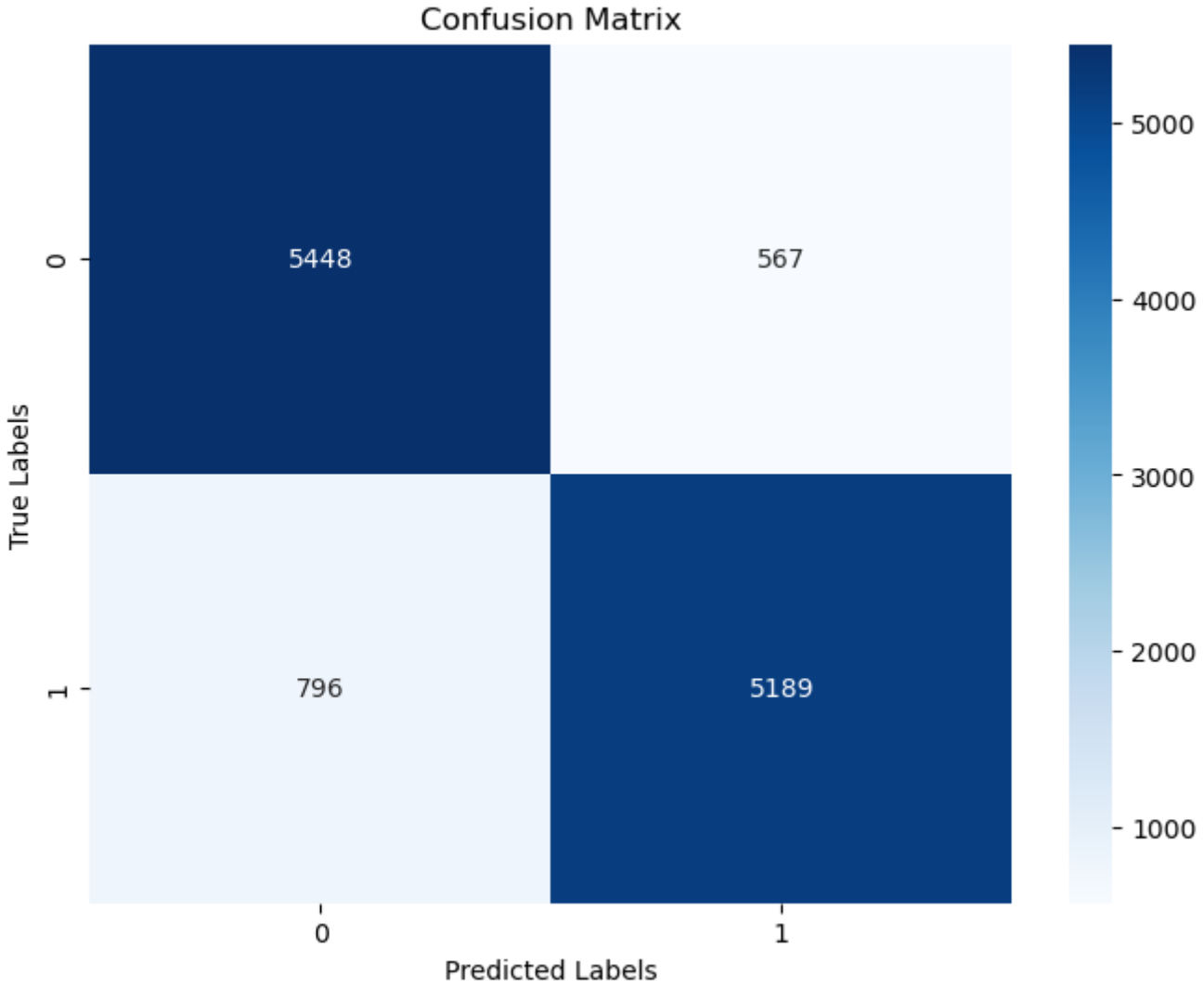
71:


Confusion Matrix
5528
487
583
5402
True Labels
Predicted Labels
0
1
0
1
5000
4000
3000
2000
1000

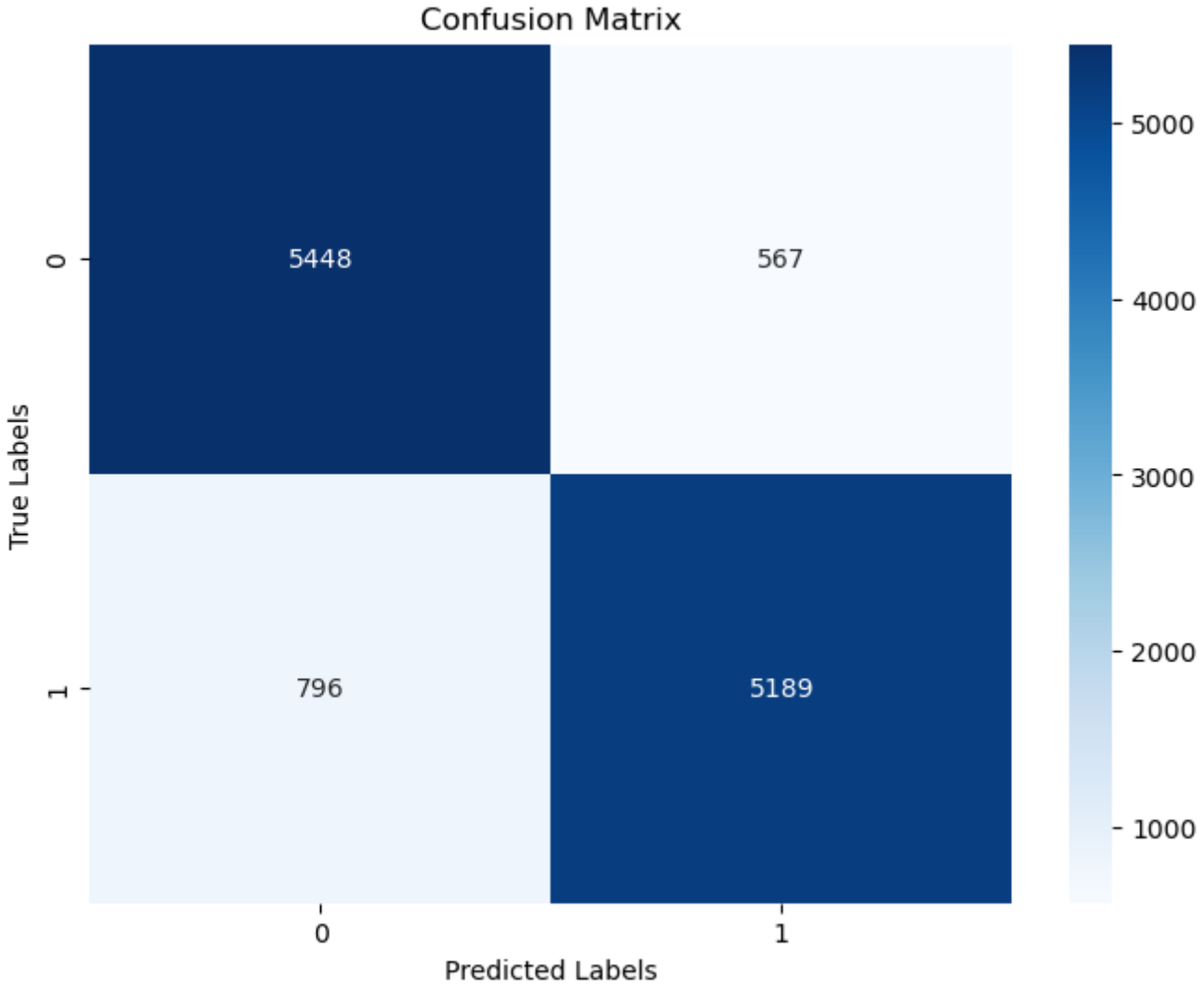
72:


Confusion Matrix
5557
458
529
5456
True Labels
0
1
Predicted Labels
0
1
5000
4000
3000
2000
1000

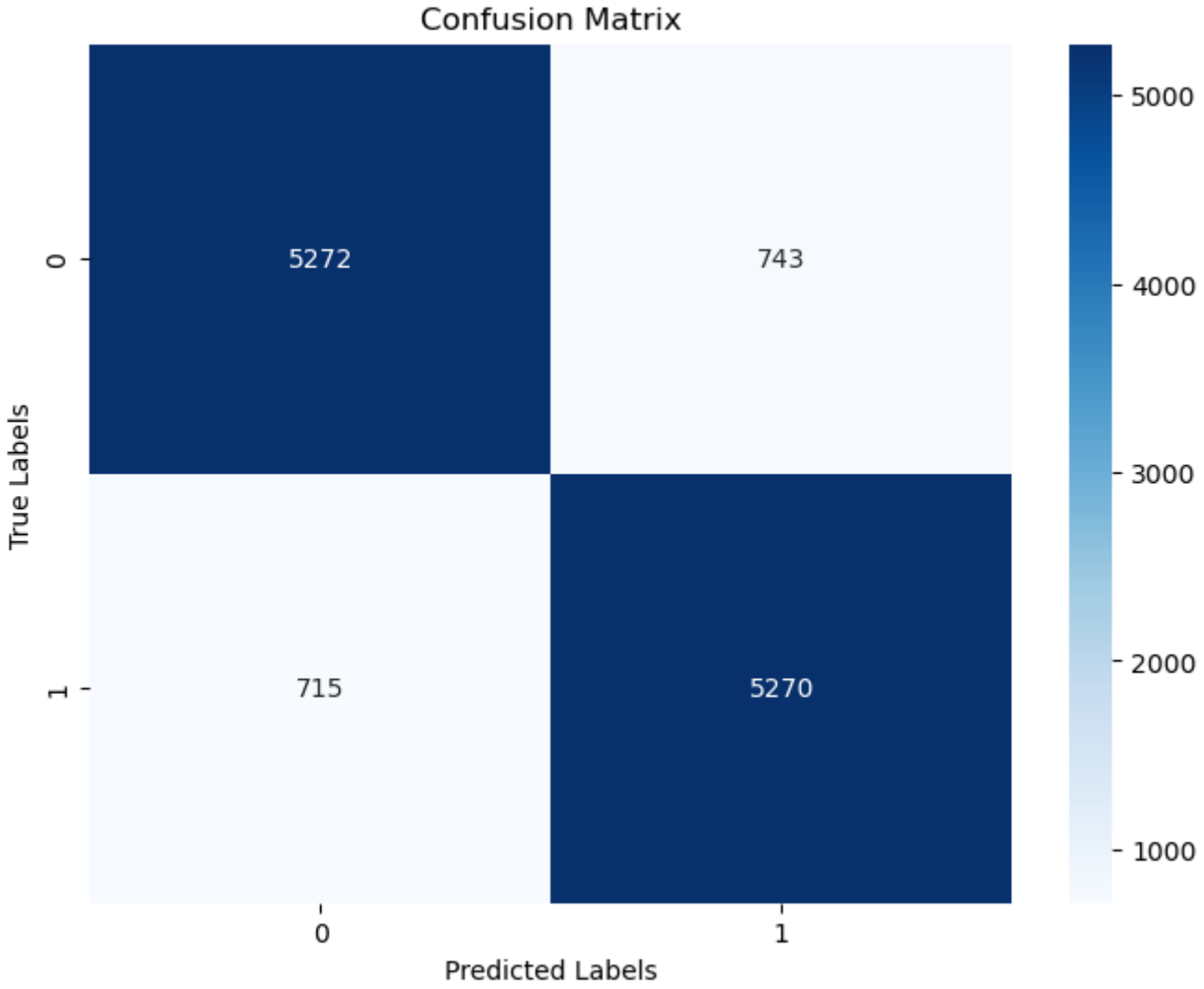
73:


Confusion Matrix
5448
567
796
5189
True Labels
Predicted Labels
0
1
0
1
5000
4000
3000
2000
1000

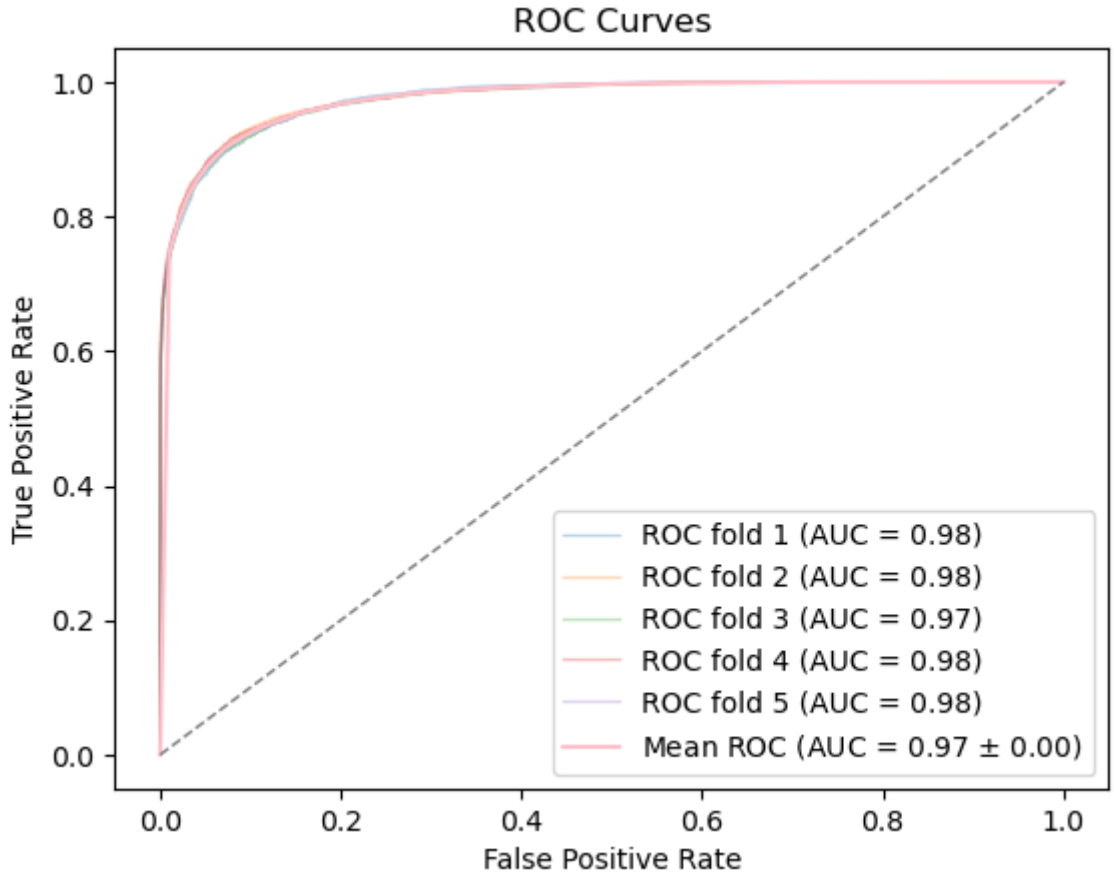
74:


Confusion Matrix
5448
567
796
5189
True Labels
Predicted Labels
0
1
0
1
5000
4000
3000
2000
1000

75:


Confusion Matrix
5272
743
715
5270
True Labels
Predicted Labels
0
1
0
1
5000
4000
3000
2000
1000

76: פייפליין:



---

77)*לכל המודלים הוספנו את GridSearchCV על מנת למצוא את הקומבינציה הכי טובה של היפר פרמטרים לכל מודל. בכולם בחרנו את scoring של roc_auc והגדרנו refit=True על מנת שהפרמטרים שמצאנו כההיפר פרמטרים הטובים ביותר למודל, ייושמו על סט האימון ובנוסף שהערכת סט הולידציה תהיה על בסיס סט האימון המשודרג הנ"ל. בנוסף הn_jobs=-1 מייצג את העבודה המקבילית שתחסוך זמני ריצה, הcv=5 זה הברירת מחדל. הCV זה הcross validation שבו נשתמש במהלך ה GridSearchCVוהוא מאפשר לנו להחליט איך לפצל לסט אימון וסט ולידציה לכל קומבינציה של ההיפר פרמטרים.

78)*מודל נוסף שבדקנו ובסוף לא השתמשנו בו הוא מודל **Naïve bayes** אך בסוף לא השתמשנו בו עקב תוצאות לא טובות. הפרמטרים היחידים שניתן להוסיף הם' var_smoothing **,** 'priors ' .השתמשנו בברירת המחדל מכיוון ש'**priors** ' מייצג את ההסתברות הפריורית לכל מחלקה, ומכיוון ואין לנו ידע מקדים על הדאטה שיתמוך בערכים מסוימים שנשים ל'**priors** ', העדפנו להשתמש בברירת המחדל בה האלגוריתם מחשב את ההסתברויות המקדימות של כל קבוצה על פי סט האימון, מהנחה שסט האימון מספיק גדול ויחסית מייצג את כל האוכלוסייה. הגדרת ה'prior' יכולה להשפיע על ההטייה. אם לא היינו בוחרים בברירת המחדל, האלגוריתם היה מתייחס להגדרת הprior שנתנו לו ונותן משקל רב יותר למחלקה המתאימה ובכך יכולנו לשנות את הbias. מההנחה שסט האימון מייצג נכונה את האוכלוסייה אנחנו מקטינים את הbias שיכל להיווצר אם היינו מגדירים אקראית את ההסתברויות הפריוריות.

בנוסף השתמשנו בברירת המחדל ב'**var_smoothing** ' פרמטר זה שולט ברמת ההחלקה של השונות. ככל שנגדיל את הערך של **var_smoothing** כך תגדל השונות (ותקטן ההטייה בהתאם) מכיוון שזה מוסיף ערך קבוע לשונות הפיצ'רים. כדי להימנע מoverfitting או הטייה גדולה מדי, העדפנו להשתמש בברירת המחדל.

Confusion matrix(79

FP	TP

TN	FN

ניראה ככה:

באופן כללי מדד הaccuracy מסביר כמה חוזו בצורה נכונה(correctly predicted) מתוך כלל הדוגמאות. כלומר בהינתן דוגמה רנדומלית , מה הסיכויים שחזה בצורה נכונה?

כאשר התשובות מוטות לצד אחד עדיף להשתמש במדד ה precision או מדד ה recall ובשמו הנוסף sensitivity.

מדד הprecision מייצג: כשחוזים שמשהו יקרה, מה הסיכויים שאפשר להיות בטוחים בשזה באמת קרה?

מדד ה recall מייצג: מתוך הדוגמאות הנכונות(True), מה הסיכויים שצדקתי?

מדרך 1 קיבלנו:

```
True Positives (TP): 5503
False Positives (FP): 512
True Negatives (TN): 5369
False Negatives (FN): 616
recall\sensitivity: 0.899329955875143
precision: 0.914879467996675
accuracy: 0.906
```

והתצוגת confusion matrix ב89

מדרך 2:

```
True Positives (TP): 5299
False Positives (FP): 716
True Negatives (TN): 5278
False Negatives (FN): 707
recall\sensitivity: 0.8822843822843823
precision: 0.8809642560266001
accuracy: 0.8814166666666666
```

והתצוגת confusion matrix ב92

מדרך 3:

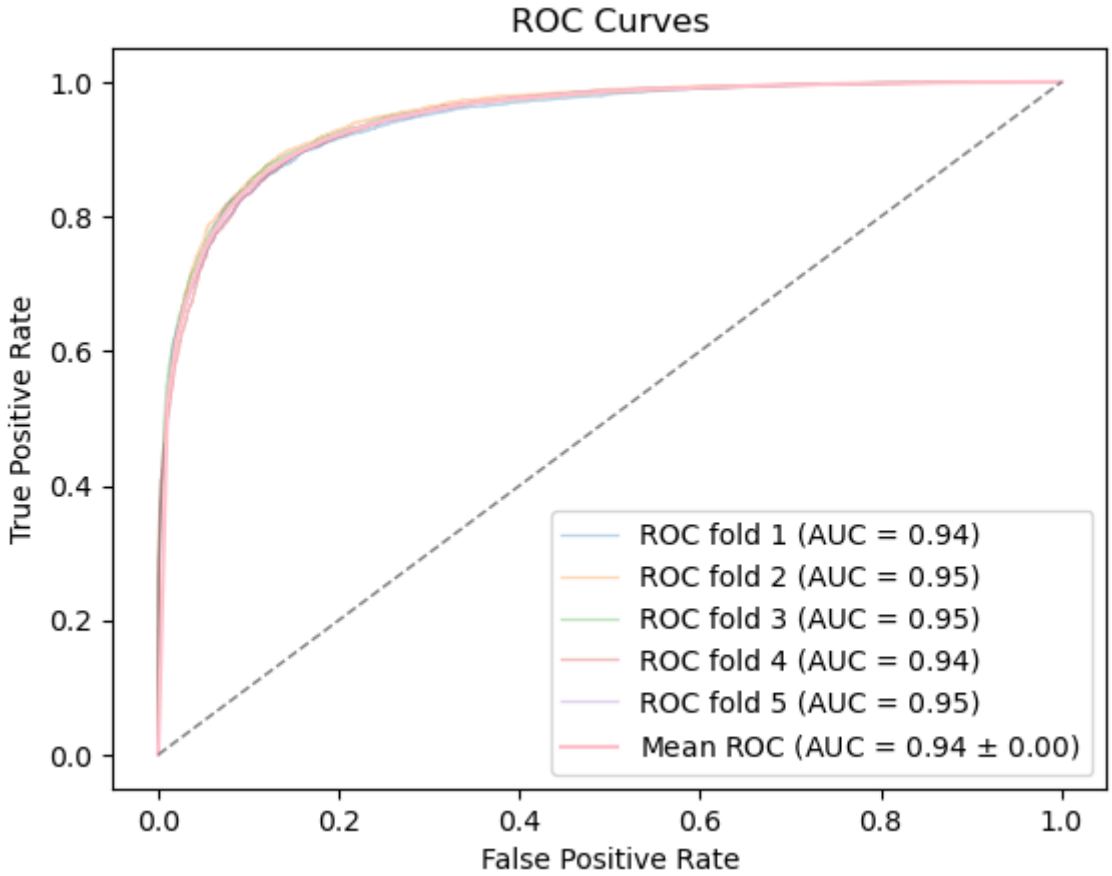
```
True Positives (TP): 5333
False Positives (FP): 682
True Negatives (TN): 5313
False Negatives (FN): 672
recall\sensitivity: 0.8880932556203164
precision: 0.886616791354946
accuracy: 0.8871666666666667
```

והתצוגת confusion matrix ב95

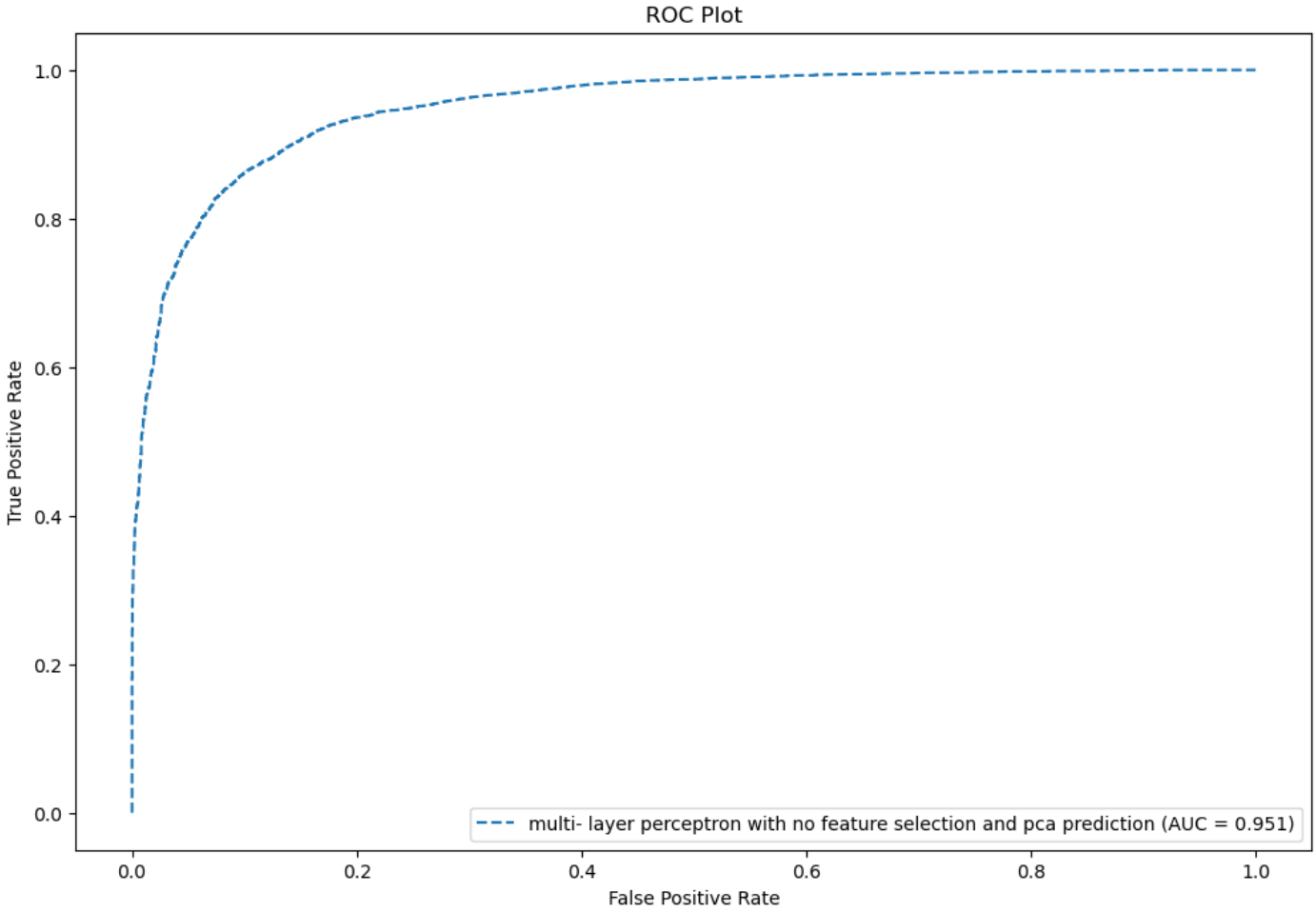
80) וקובץ הCSV נראה ככה: (רואים פה רק חלק):

B	A	
proba	sha256	1
0.994732	023928c14	2
0.229575	6436083d4	3
0.485881	038e71f41	4
0.008172	be913ef29	5
0.37193	e78f83f0af	6
0.997935	c290de562	7
0.007651	15c686509	8
0.990066	633e2d2c1	9
0.16656	e80ed8f43	10
0.235311	a3071ca37	11
0.975499	1de60dd8a	12
0.320449	5349c2324	13
0.427857	3f9b01ac4	14
0.994283	e2c2d607d	15
0.172923	4967f1a3b	16
0.317981	540eb4eb6	17
0.325025	511312bd5	18
0.845333	7e46eac60	19

81) הבדלים לדוגמה:


ROC Curves
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
ROC fold 1 (AUC = 0.94)
ROC fold 2 (AUC = 0.95)
ROC fold 3 (AUC = 0.95)
ROC fold 4 (AUC = 0.94)
ROC fold 5 (AUC = 0.95)
Mean ROC (AUC = 0.94 ± 0.00)

(82


ROC Plot
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
multi- layer perceptron with no feature selection and pca prediction (AUC = 0.951)