Proiect Final

Analiza Integrată a Datelor și Raportare

,

Abstract

În raportul de față vom prezenta o încercare de analiză unor seturi de date referitoare la cazurile de Covid-19, care au fost mediatizate, ratele șomajului, populația rezidentă și emigranții temporari. Folosind acestea ca variabile în studiul nostru, au fost create două modele pentru a explica variația numărului de emigranți temporari, în județele din România, pe parcursul anului 2020, prin ratele de cazuri mediatizate de confirmați, decedați și vindecați, ratele de șomaj și populația rezidentă a județelor amintite, în aceeași perioadă de timp. Dintre cele doua modele propuse, s-a ales unul singur: de regresie liniară. Cum a fost ales și procesul care a dus la construcția și alegerea acestuia vor fi detaliate în cadrul prezentului raport.

1. Crearea și popularea unei baze de date locale

Pentru buna funcționare a lucrului în echipă, în cadrul acestui proiect, au fost folosite: GitHub, DataSpell și MongoDB, codul fiind scris sub forma de Jupyter Notebook. Cu ajutorul GitHub a fost creat un repository în care să poată fii puse toate fișierele de output, care au rezultat în urma analizei desfășurate de către fiecare membru al echipei. Am găsit această metodă ca fiind foarte eficientă, atât în managementul echipei și a urmăririi progresului proiectului, dar și a ușurinței de conectare la următorul sift pe care l-am folosit: DataSpell, care permite clonarea repository-ului GitHub Classroom. Aici, fiecare membru al echipei și-a creat propriul Jupyter Notebook, în care și-a redactat codul potrivit rezolvării sarcinii aferente fiecăruia. Următor acestui pas, fiecare fișier sub forma „.csv” a fost încărcat într-o colecție corespunzătoare numelui fiecăreia în MongoDB, care odată conectat la DataSpell a făcut mult mai facilă conectarea la baza e date necesară. MongoDB este o bază de date cross-platformă și open-source orientată pentru documente, un fel de bază de date NoSQL, ce redă structura bazată pe tabela bazei de date relaționale pentru a adapta documente asemănătoare JSON care au scheme dinamice pe care le numește BSON. A fost folosită această variantă de încărcare a datelor dintr-o bază de date non-relațională pentru a imita cât mai bine o situație reală întâmpinată în momentul intrării pe piața muncii, situație în care datele vor fi stocate în Cloud.

Primul pas făcut în orice analiză este acela de a accesa datele și de le aduce sub un format cu care se poate lucra mai departe în analiza exploratorie și preprocesarea datelor. Prin urmare, am adus datele din API-ul asignat echipei noastre[1]. Acesta vine într-un format de tip .json, cuprinzând multiple dicționare în dicționare. Am verificat cărui subset de date îi aparțin datele, care pot fi desfășurate în coloane ale unui DataFrame și am descoperit două subseturi, pe care le-am salvat, iar mai apoi lipit, având acum un DataFrame cu patru coloane, dintre care una cuprinde un dicționar. Următorul pas a fost să desfășor coloana mai devreme menționată și să o lipesc la DataFrame-ul de mai înainte creând un nou set de date, din care mai apoi am eliminat coloana nedesfășurată, având acum informațiile separate pe coloane. Acestea fiind realizate, baza de date este acum gata de salvare sub formă „.csv", numit „case_relations_new.csv", o bază de date care cuprinde informații despre cazuri de persoane confirmate, decedate, sau vindecate, care au fost mediatizate.

Următorul pas a fost căutarea unui nou set de date dintr-o sursă oficială, care să ne ajute mai departe la construirea unui model pentru predicție. Am ales, în acest scop, ratele de șomaj pe județele din România în anul 2020 din setul de date pus la dispoziție de INSSE Tempo[2], deoarece datele din baza de date precedentă indicau același an. Datele au fost descărcate sub formă „.csv", însă numele coloanelor nu erau recunoscute, ci erau dispuse ca parte din baza de date, prin urmare le-am setat în comanda de citirea a .csv-ului, iar mai apoi cu ajutorul funcției .iloc am suprascris DataFrame-ul fără primul rînd, care cuprindea numele coloanelor care nu au fost recunoscute înainte, după care l-am salvat în folderul creat pentru bazele de date, sub numele: „someri.csv". Același proces l-am urmat și cu datele referitoare la populația rezidentă în județele din România[3], date alese cu scopul creării mai târziu a unor rate de confirmați Covid-19, decedați sau vindecați, mediatizați, din populația rezidentă a fiecărui județ. La fel am procedat și cu datele referitoare la emigranții temporari, în anul 2020, pe județe[4]. Cele două baze de date au fost salvate sub denumirile de: „pop_rezidenta.csv", respectiv „emigranti_temp.csv" în folderul „data", care cuprinde și celelalte baze de date. Odată create aceste .csv-uri cu fiecare bază de date

[1] API Case Relations Database: https://covid19.geo-spatial.org/api/statistics/getCaseRelations
[2] INSSE Tempo - Rate șomaj: http://statistici.insse.ro:8077/tempo-online/#/pages/tables/insse-table
[3] INSSE Tempo - Populația rezidentă: http://statistici.insse.ro:8077/tempo-online/#/pages/tables/insse-table
[4] INSSE Tempo - Emigranți temporari: http://statistici.insse.ro:8077/tempo-online/#/pages/tables/insse-table

le-am încărcat în MongoDB în câte o colecție cu același nume ca și bazele de date, iar acum datele sunt gata pentru a fi mai departe explorate.

2. Analiza exploratorie a datelor - EDA

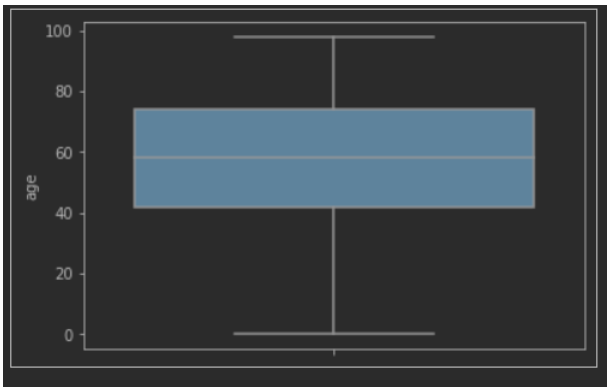
În cadrul acestui pas am explorat modul în care sunt distribuite datele, atat prin reprezentare vizuala, cât și prin examinarea principalelor moduri prin care pot sa observ cum distribuite datele în setul nostru de date. Astfel, pentru a putea incepe rezolvarea acestui pas a fost necesara accesarea celor patru baze de date pe care le-am utilizat in aceasta analiza statistica, acest lucru realizandu-l prin intermediul conectării la MongoDB, creand un cursor by default pentru a le putea accesa direct prin intermediul acesteia.

Mai departe a urmat sa vad cum este distribuit fiecare set de date, observand ca niciun set de date nu are același număr de coloane si randuri: “case_relations” avand 10630 de randuri si 19 coloane, cel de “șomeri” - 42 de rânduri și 6 coloane, “pop_rezidenta” - 84 de randuri si 7 coloane, iar ultima, cea de “emigranți” - 42 de rânduri și 7 coloane. A urmat sa vizualizez pentru fiecare coloana din seturile de date din analiza fiecare tip de date pe care îl au coloanele, iar unde a fost cazul le-am transformat în tipul de date specific pentru a putea vedea cum sunt mai departe distribuite acestea (de exemplu: din object in datetime). Am observat ca am avut diverse tipuri de date, cum ar fi: “object, datetime[ns], boolean, float64, int64”, iar pentru ca am observat ca sunt foarte puține de tip numeric am ajuns la concluzia ca mi-ar fi dificial ca in acest pas sa realizez o corelație ca sa înțeleg cum coloreaza variabilele între ele, seturile de date fiind și diferite, unirea lor având loc doar în următorul pas de preprocesare.

A urmat să examinez valorile lipsa, iar la acest subpunct am văzut ca setul de date: “populatie”, “șomeri” și “emigranti” nu au coloane cu valori lipsa, ci doar setul de baza cu privire la COVID, adica “case_relations”, are valori lipsa. Acest lucru ne confirma ca cel mai ușor ar fi sa eliminăm coloanele cu valori lipsa pentru ca în cazul analizei din acest proiect n-ar fi fost importante ci doar ar fi influențat ca modelul sa nu iasa unul potrivit. De asemenea, chiar dacă nu au existat valori lipsa în celelalte seturi de date, tot se găsește relevata ideea de a elimina cateva coloane care pentru ca exista coloane cu randuri care au aceeași valoare, cum ar fi coloana cu anul, unde se repeta valoarea pe toate randurile.

Înainte de realizarea statisticilor descriptive pentru fiecare set de date ca să-mi fie mai ușor de vizualizat în forma tabelara am verificat și randurile duplicate și am eliminat cateva coloane pe care le-am considerat ca fiind irelevante in acest proces (toți pașii care țin de preprocesare au fost realizați doar în pasul respectiv, aici nu am salvat nicio baza de date în urma micilor modificări, considerand relevant ca fiecare activitate sa fie facuta in propriul pas pentru a se înțelege mai bine ce înseamnă fiecare proces).

FIG 1. Boxplot varsta



În pasul statisticilor descriptive, la fel și in graficul din Fig. 1, am putut în principiu sa vad cum se situează valoarea medie a variabilelor, realizand diverse filtre ca sa înțeleg mai bine datele. In setul de date cu COVID-19, avand doar o variabila numerică, varsta media a populației este în jurul varstei de 55 de ani, aceasta abatandu-se de la medie cu 22,98 de abateri standard care se referă la: cu cât punctele de date sunt mai îndepărtate de valoarea medie, cu atât este mai mare abaterea din setul de date, ceea ce reprezintă faptul că punctele de date sunt împrăștiate pe o gamă mai largă de valori și invers. Minimul varstei este reprezentat de 0 ani, iar maximul de 98. Cuantilele sunt puncte tăiate și împart intervalul unei distribuții de probabilitate în intervale continue cu probabilitate egala, cuantila de 25% fiind reprezentată de varsta de 42 de ani, cea de 50% de 58 de ani, iar cea de 75% de 74 de ani, cuantila de 50% reprezentând și mediana.

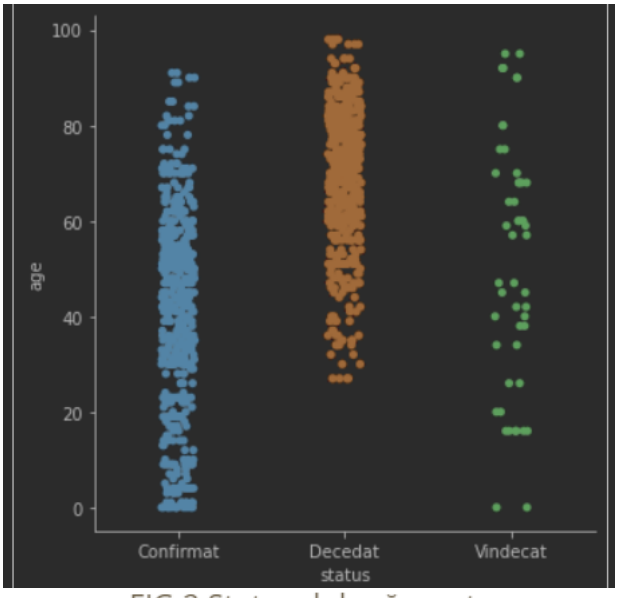


FIG 2 Statusul după varsta

Din graficul Fig. 2, comparand cu varsta statusul cazurilor, în cazul statusului "confirmat" varsta este una uniforma, si cazurile sunt distribuite la orice interval de varsta într-un număr mai mare cu accent la mijlocul liniei formată din puncte. Varsta celor decedați are o intensitate mai mare de la aproximativ 25 de ani în sus, pană la sfarsitul vietii, de la 60 de ani în sus linia fiind mai accentuata. Aici varsta pare sa influențeze

numărul celor decedați, însă nu poate sa fie un factor definitoriu, ci ar trebui sa consultam și alți factori pentru a putea vedea ce influențează cu adevărat acest status. Cazurile vindecate sunt distribuite aproximativ uniform, însă și în acest caz vorbim despre un număr redus de cazuri vindecate, punctele find puține, însă asta înseamnă ca vindecarea nu tine cont de varsta.

In setul de date al somerilor, media ratei de șomaj în cele 42 de județe este de 3,93, abătându-se de la medie cu 1,72. Valoarea cea mai mica a ratei de șomaj de 0.70 se inregistreaza în Ilfov, iar cea mai mare de 7,90 in Buzau, mediana avand o valoare de 3,65, iar de obicei nu este foarte diferită de medie. La fel aceste valori pot sa fie vizualizate și în graficul de mai jos (Fig, 3), unde valori mari ale ratei de șomaj (peste 4,2%)sunt înregistrate și în alte județe, cum ar fi: Salaj, Covasna, Harghita, Bacau, Neamt, Suceava Vaslui, Galati, Ialomita, Teleorman, Dolj, Mehedinti, Olt.

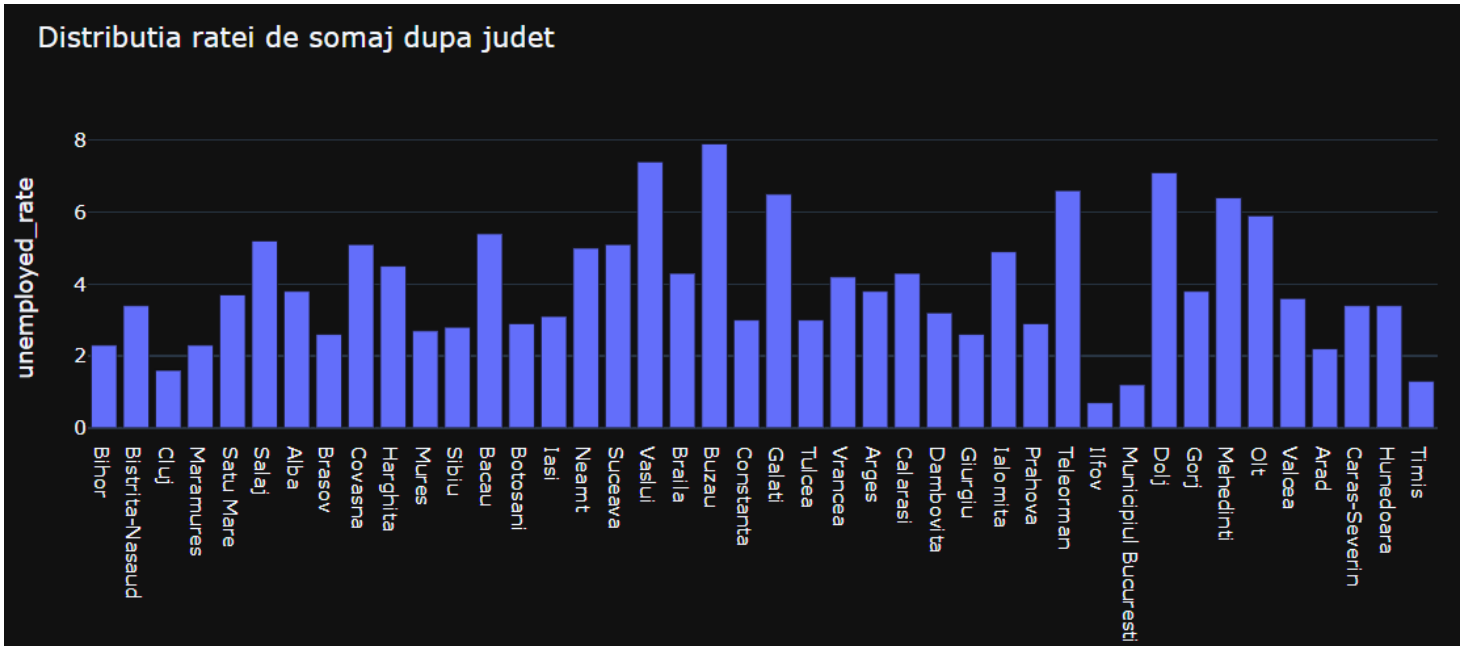


FIG 3. Rata somaj

Baza de date a populatiei, cât și a emigranților temporari este reprezentată tot pentru cele 42 de județe, media emigranților fiind de 5565 de indivizi, cu un număr maxim de 19593 și unul minim de 2526. Prin intermediul graficului putand sa observ ca cei mai multi emigranti sunt proveniți din București, iar cei mai puțini în Tulcea, Salaj sau Covasna.

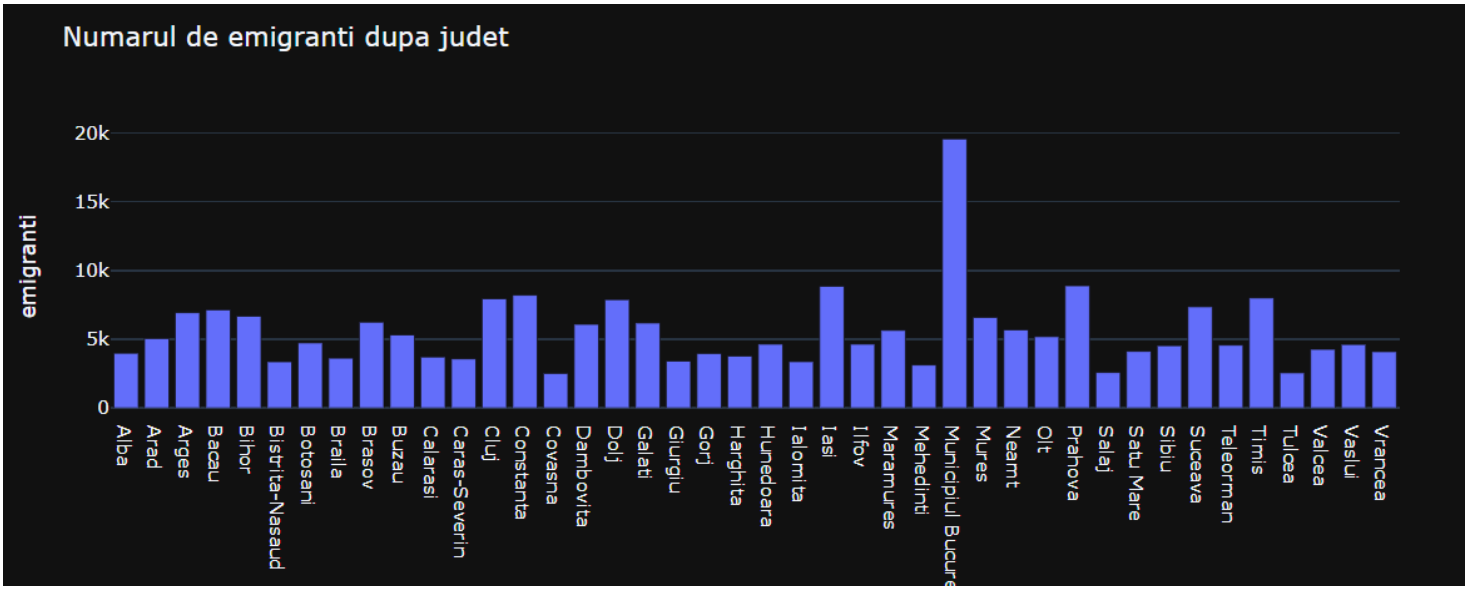


FIG 4. Numar emigranti temporari

Mai departe am dorit sa vad cati dintre cei care s-au infectat cu COVID sunt barbati, respectiv femei, astfel 565 au fost femei și 478 barbati, însă exista așa cum vedem și în grafic linia verde există valori lipsa, astfel pentru a fi mai ușor procesul o sa eliminam aceasta coloana in analiza finala. Barbatii reprezinta categoria care este reprezentata de un număr mai mare în cadrul fiecărui status, urmat ulterior de femei. Pe langa asta, pot sa vad ca si cei mai mulți sunt cei confirmați urmati de

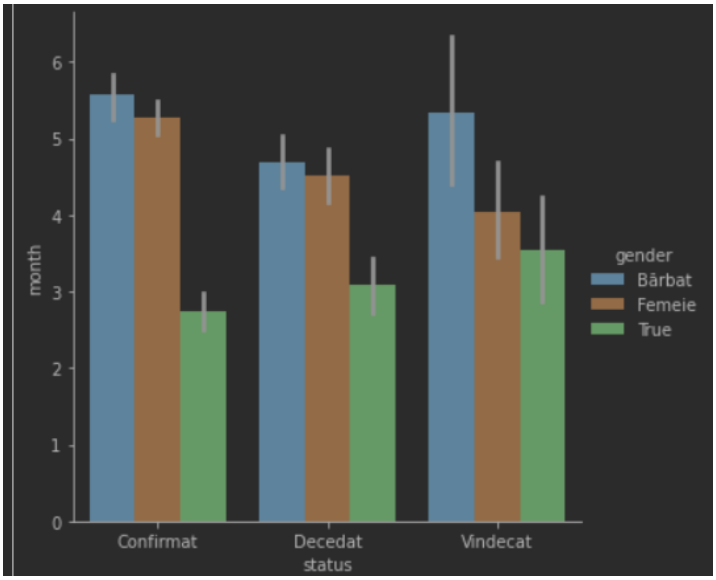


FIG 5. Distribuția cazurilor după gen

cei vindecati, decedati fiind mai putini. Referitor la țara de infecție, cei mai mulți sunt din România, urmata si de catre Italia, Fanta, Moldova, etc. În cazul acestea trebuie sa normalizam datele pentru ca România apare atat cu diacritice, cat si fara, iar acest lucru ar putea influența negativ analiza, alta optiune fiind la optarea eliminarii variabilei din analiza de față.

Înainte de reprezentarea grafica am filtrat cazurile de COVID-19 după status “vindecați”, “confirmati” si “decedati” in ordine ascendenta. Rezultatele arată ca în Suceava au fost cele mai multe cazuri confirmate (1932), dar si decedate, diferenta fiind foarte mare fata de alte orașe, cele mai puține cazuri in Salaj (14), insa exista si o zona categorizata ca fiind “necunoscuta”. In Caras-Severin, Bihor și Gorj s-au declarat cei mai mulți “decedați, iar în Ialomita cei mai mulți vindecați.

În final a urmat și reprezentarea grafică unde inițial am împărțit statusul în trei coloane pentru a putea realiza mai clară reprezentarea grafica, însă așa cum am zis mai sus -modificările pe baza de date s-au făcut în pasul specific. Chiar dacă as fi vrut sa realizez o harta până la urmă pentru ca voiam doar sa vad unde sunt cele mai multe cazuri am optat pentru tree map-ul și astfel am creat trei tree map-uri, unde am reprezentat numărul de persoane confirmate, vaccinate și decedate cu referire la COVID, așa cum le-am aratat inițial și sub forma de tabel.

Fig 6. Tree Map a cazurilor cu statusul confirmati

Fig.7 Tree Map a cazurilor cu statusul decedati

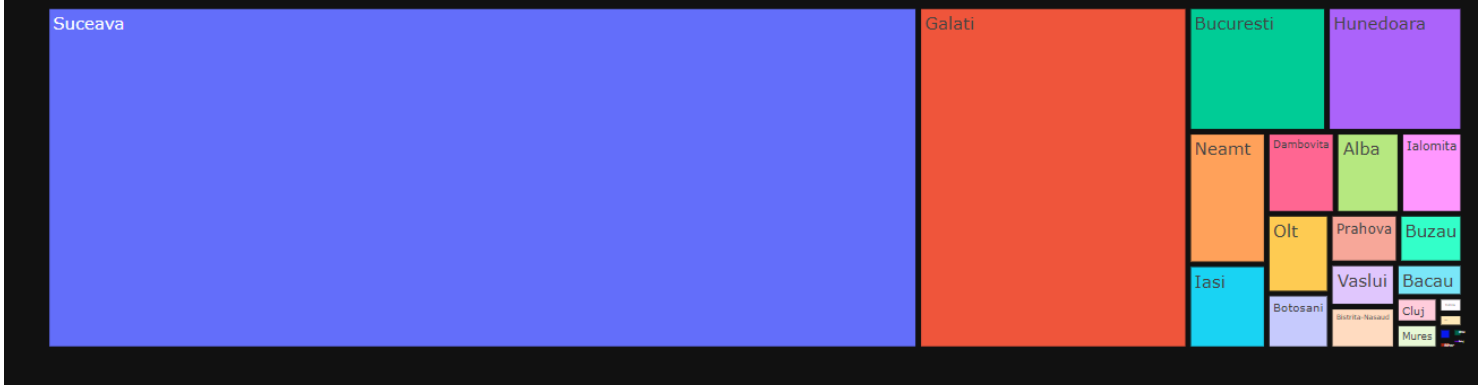


Fig 8. Tree Map a cazurilor cu statusul vindecati

Așadar din cele trei grafice de mai sus pot sa vad ca cele mai multe cazuri confirmate de COVID intradevar sunt în Suceava, la fel și cele cu statusul “decedați”, urmat în ambele cazuri de Galați si București. Insa, uitandu-ma la graficul cu cei vindecați pot sa observ ca Ialomița este județul care ia primul loc, ceea ce reprezinta ca cei mai multi oameni vindecati se regasesc in Ialomita, abia apoi fiind urmat de Suceava si Bucuresti.

La fel mă ajuta și graficul cu distribuția cazurilor realizat mai jos în cadrul caruia pot sa vad cu linia albastra ca sunt cei mai multe cazuri de “confirmați”, cei cu roșu de “decedați”, iar în final și “vindecați” despre care pot sa spun ca sunt cele mai puține cazuri. Uitandu-ma pe acest grafic (aici se vede puțin mai mic din cauza formatului pentru a se înțelege liniile) nu doar în Suceava sunt multe cazuri confirmate, ci si in orașe precum Galati, Iasi, Buzau, Braila, Dambovita, București. Deci pot concluziona din acesta ca in zona estica a tarii (predominat partea Moldovei) exista cam cel mai mare număr de cazuri confirmate. Numărul de cazuri vindecate este destul de mic și nu predomină într-o proportie prea mare aproape în niciun județ, iar cele de decedați au valori mai mari în Suceava, Galati, urmat de București. Totusi, comparativ cu cele două statusuri de “confirmat” și “decedat”, exista mai multe cazuri de “confirmați”.

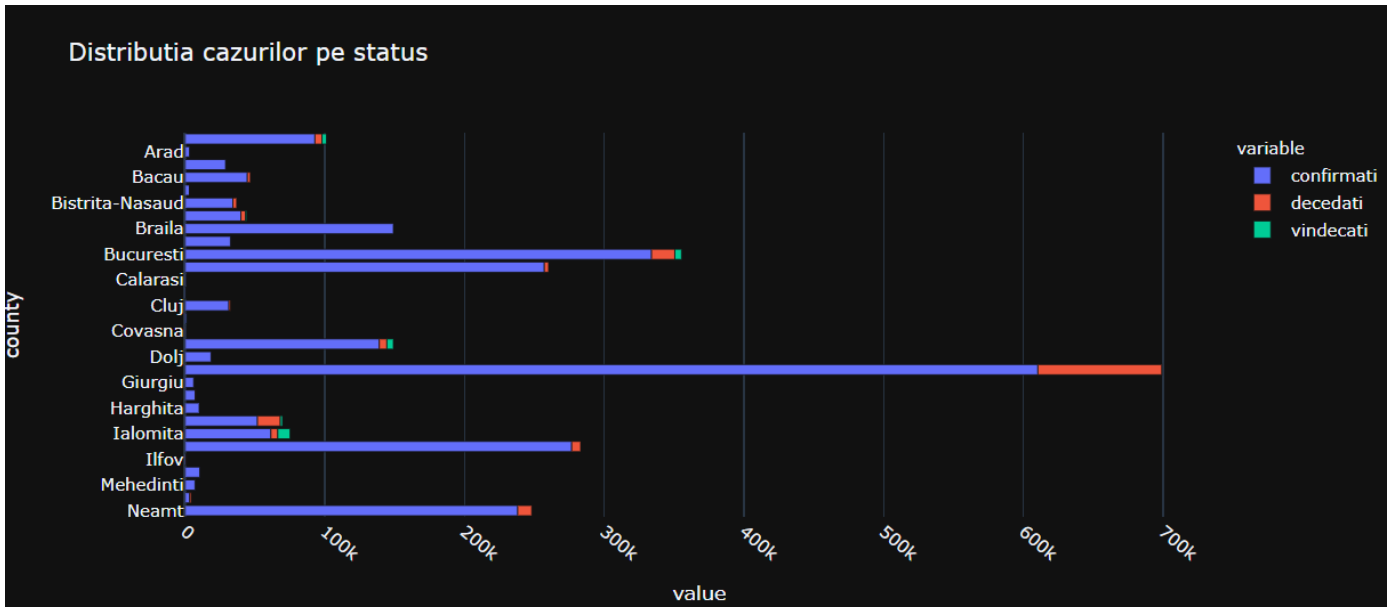


FIG 9 Distribuția cazurilor după status

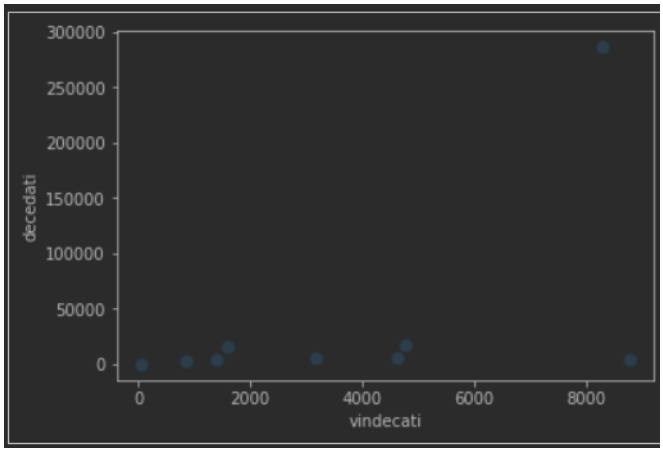


FIG 10. Scatter plot: decedati-vindecati

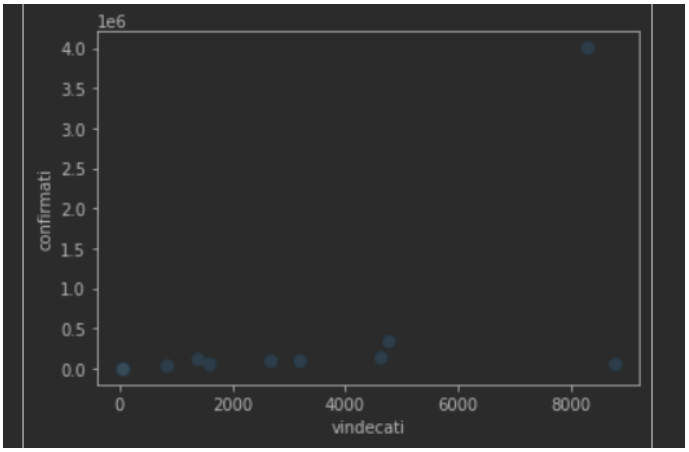


FIG 11. Scatter plot: confirmati-vindecati

Primul scatterplot reprezinta asocierea dintre “confirmatii” si “vindecatii”, iar al doilea “vindecatii” si “decedatii”, din care pot sa observ ca linia este oarecum orizontala si merge de la stanga la dreapta, existand o mica asociere, insa exista si outleri in ambele grafice, care pot sa fie văzute în partea dreapta (-->). Vizualizand cele prezentate mai sus exista posibilitatea destul de mare sa vorbim despre Suceava în ambele cazuri. Asadar, după ce am examinat datele și am încercat sa vad cum sunt distribuite acestea în vederea obținerii unei analize cat mai reușite a urmat preprocesarea în pasul următor, intrucat s-a observat prezent faptul ca exista coloane care necesita eliminarea din analiza, informații lipsa sau chiar date care nu se pliaza bine pe aceasta analiza.

3. Pre-procesarea datelor

În cadrul acestui pas am pregătit și modificat seturile de date pentru a putea fi folosite în analizele viitoare. Am început prin a importa cele patru seturi de date din MongoDB. După accesarea celor patru seturi de date, am început modificările primului set de date și anume „case_relations_new". În cazul acestui set de date am adus cele mai multe modificări, deoarece conținea foarte multe coloane și foarte multe rânduri, multe dintre ele având valori lipsă.

Am început cu eliminarea tuturor coloanelor care nu ne erau utile, rămânând în final cu doar două coloane, care reprezentau județele și statusul: „confirmat", „vindecat", „decedat". Pentru ca toate seturile noastre de date să aibă același număr de valori, am grupat datele după cele trei categorii de status, în funcție de fiecare județ, urmând ca datele grupate să le unesc într-un nou set de date.

După ce am obținut noul set de date, am verificat prezența valorilor lipsa. Am observat faptul erau foarte multe date lipsă, așa că am înlocuit toate acele valori cu 0. Am decis să înlocuim toate datele lipsă cu 0 deoarece, fiind date despre numărul persoanelor confirmate, vindecate sau decedate din cauza Covid-19, raportate în presa din România, am considerat faptul că datele care lipseau nu au fost raportate în presă.

Explorând datele am observat faptul că numele unui județ era trecut drept „Necunoscut". Din lista de județe lipsea județul „Satu-Mare", așadar, am considerat că județul „Necunoscut" este județul „Satu-Mare", ceea ce a dus la înlocuirea numelui în setul nostru de date. De asemenea, am mai mai modificat și județul „București" în „Municipiul București", pentru ca fi scris la fel în toate bazele de date pe care le-am folosit. Am mai adus de asemenea modificări în ceea ce privește tipul datelor pe care le-am utilizat. Am transformat datele din tip „float" în tip „int64" pentru a putea fi utilizate pe viitor în analize.

După modificarea primului set de date, a urmat modificarea seturilor de date de pe INS Tempo: „someri", „pop_rezidenta" și „emigranti". În cazul acestor seturi de date nu au fost necesare foarte multe modificări, deoarece, fiind date oficiale, nu au existat date lipsă pe care să fie nevoie să le înlocuiesc, sau nevoia de a grupa datele după anumite caracteristici. Modificările pe care le-am adus în cazul acestora a ținut de eliminarea datelor de care nu aveam nevoie. Pentru setul de date în care aveam șomerii am rămas doar cu județele și rata de șomaj din fiecare județ, în cazul populației rezidente am rămas doar cu

județele și cu numărul de persoane din fiecare județ, iar în rândul emigranților, am rămas de asemenea cu județele și numărul de emigranți temporari din fiecare județ.

După ce am ajuns la forma dorită și a datelor oficiale, am unit cele 4 seturi de date în unul singur. Pentru început, am unit populația rezidentă cu datele despre persoanele confirmate, decedate sau vindecate, cheia de legătură fiind variabila „county”. Am ales să unesc mai întâi aceste două seturi de date deoarece, datele din interiorul acestora mi-a permis să calculez ratele de persoane confirmate, vindecate și decedate în funcție de fiecare județ, rate care au fost utilizate ulterior în analize. Aceste noi date le-am unit mai apoi cu datele despre șomeri și emigranți, ajungând aproape de forma finală a bazei noastre de date.

Am vizualizat distribuția noului set de date printr-o serie de boxplot-uri și am observat faptul că în rândul fiecărei variabile existau outlieri. Așa cum am văzut și mai devreme în etapa de vizualizare a datelor, județul Suceava a avut cele mai mari valori, așadar, am hotărât să eliminăm acest județ din setul nostru de date, urmând ca analizele făcute în continuare să nu îl aibă în vedere.

Ultima modificare asupra datelor a fost eliminarea coloanelor în care apărea numărul de persoane confirmate, vindecate și decedate, deoarece aceste date nu vor mai avea nicio utilitate în continuare, fiind utile doar în calculare ratelor raportate la numărul populației.

După toate aceste modificări, setul nostru final de date este compus din 7 coloane, care reprezintă județul, rata persoanelor confirmate, rata persoanelor vindecate, rata persoanelor decedate, populația rezidentă, rata de șomaj și numărul emigranților temporari, și 42 de rânduri.

Următorul pas a fost vizualizarea noilor date. În această etapă, am reprodus câteva din graficele făcute de colega mea la pasul anterior pentru a vedea cum s-au modificat acestea în urma modificării datelor. Am început prin a crea trei treemap-uri cu ratele pe care le-am calculat.

Fig 12: Treemap cu rata persoanelor confirmate

Figura de mai sus reprezintă treemap-ul pentru rata persoanelor confirmate. Am observat faptul că după eliminarea județului Suceava, județul cu cea mai mare rată a persoanelor confirmate este județul Galați. În Figura 13 se poate observa că și în cazul persoanelor decedate, tot în județul Galați s-a înregistrat cea mai mare rată.

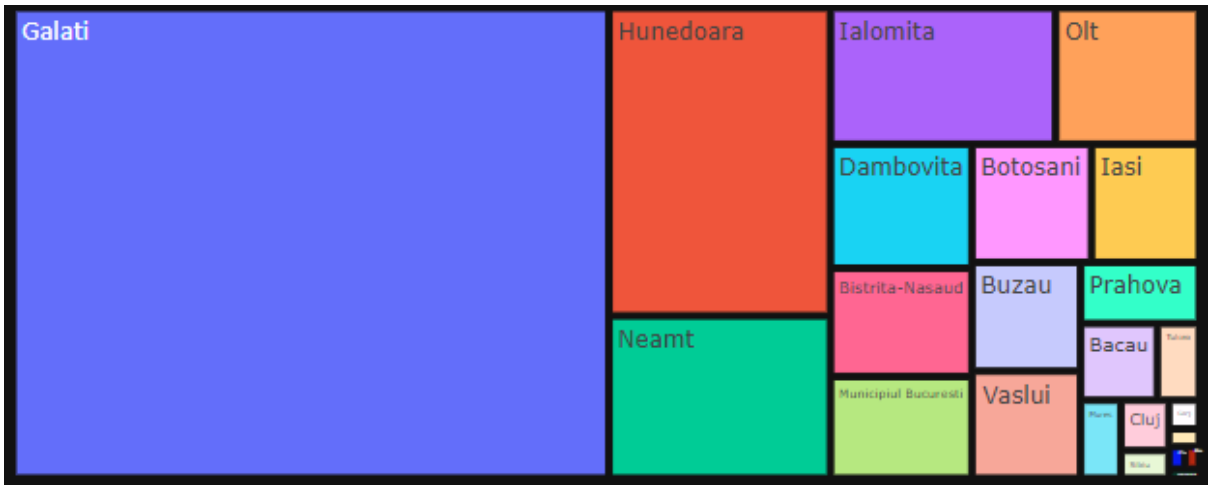


Fig 13: Treemap cu rata persoanelor decedate

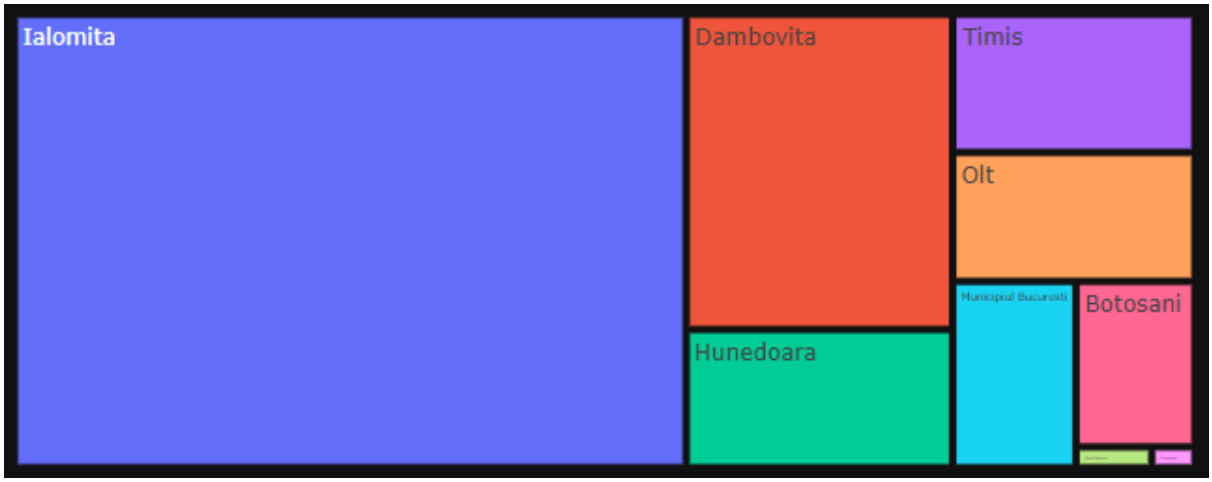


Fig 14: Treemap cu rata persoanelor vindecate

În cazul ratei persoanelor vindecate, am observat faptul că și după toate modificările pe care le-am adus setului de date, situația nu s-a schimbat, județul Ialomița având în continuare cel mai mare raport de persoane vindecate de Covid-19.

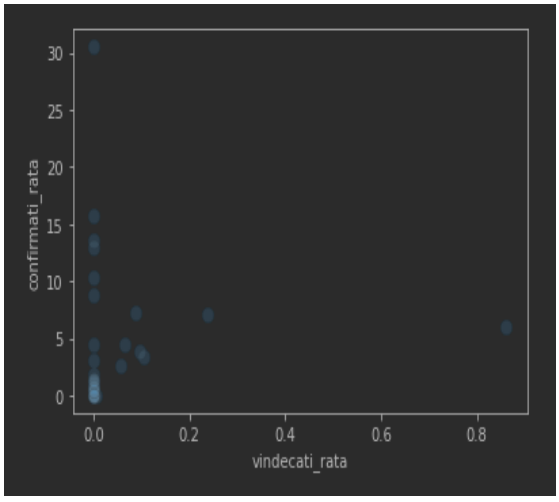


Fig 15: Scatterplot vindecați-confirmați

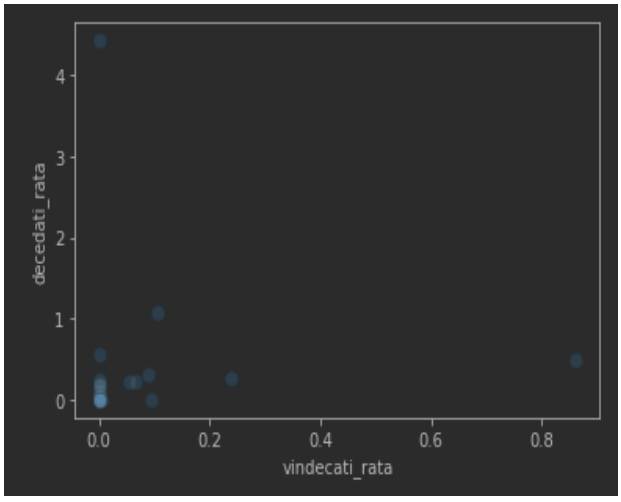


Fig 16: Scatterplot vindecați-decedați

În urma realizării unui scatterplot în care am inclus rata persoanelor vindecate și rata persoanelor confirmate (Figura 15), am observat faptul că acum datele noastre nu mai sunt reprezentate sub forma unei linii orizontale, ci sunt mai degrabă așezate sub forma unei linii verticale, existând și în acest caz outlieri, atât în ceea ce privește rata persoanelor vindecate, cât și în rândul ratei persoanelor confirmate. Analizând Figura 16, am observat faptul că în cazul persoanelor vindecate și a celor decedate, distribuția tinde să se

conglomereze într-un anumit punct și nu se mai observă forma unei linii. De asemenea, există outlieri în rândul persoanelor vindecate și a celor decedate.

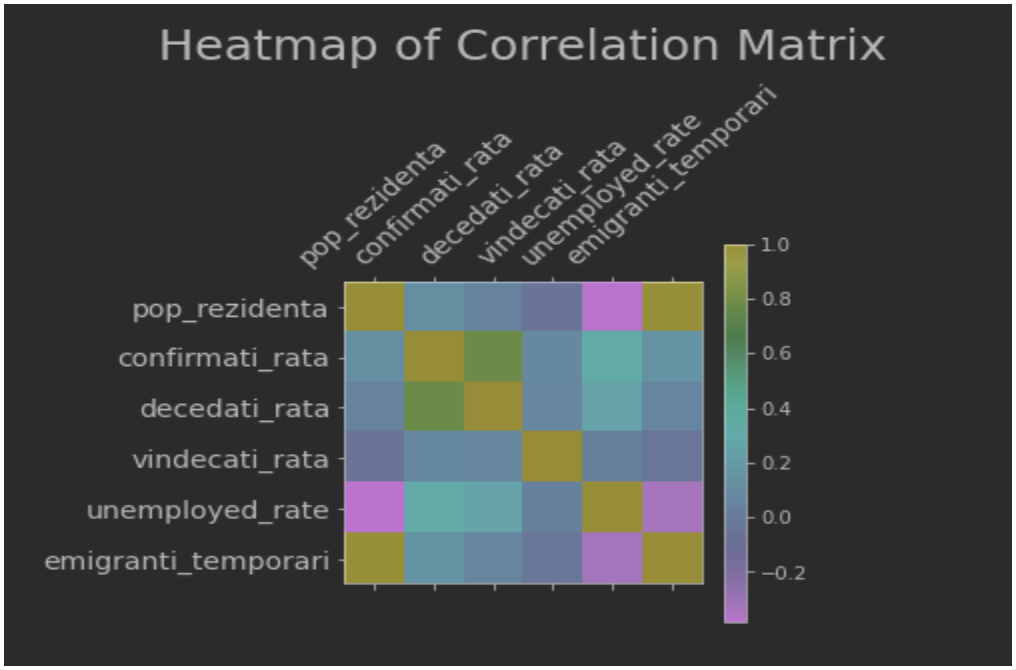


Fig 17: Heatmap-ul corelațiilor dintre variabile

Pentru realizarea graficului de mai sus, a fost nevoie să elimin variabila „județ” din setul nostru de date, deoarece nu reprezenta o variabilă numerică, așadar nu putea fi inclusă într-o analiză de corelație. Eliminarea variabilei a fost făcută doar pentru acest grafic, aceasta făcând parte din setul de date final. În urma realizării graficului am observat faptul că există o corelație puternică între rata persoanelor confirmate și rata persoanelor decedate, de asemenea o corelație puternică existând și între numărul de emigranți temporari și populația rezidentă.

4. Selectare model

După ce am primit datele preprocesate și implicit varianta finală a setului nostru de date, am trecut mai departe la aplicarea unor modele de predicție, deocamdată fiind la nivel de testare. Primul pas a fost acela de a importa datele actualizate în MongoDB, iar mai apoi conectate la DataSpell. Am transformat setul de date într-un DataFrame și l-am folosit mai departe sub numele de „df”. Înainte de a putea începe efectiv analiza, a fost nevoie să

șterg coloanele “_id” și “county” deoarece nu am avut nevoie de ele în analiză, ba chiar ar fi putut afecta rezultatele. După ce am eliminat aceste coloane, am calculat corelația variabilelor.

Următorul pas a fost să împart setul de date în unul de antrenare și unul de testare. Această împărțire are un rol foarte important deoarece, setul de antrenare (train), este necesar pentru a învăța algoritmul să facă predicții mai bune. Pe setul de date train aplicăm toate analizele și metodele pentru a îmbunătății cât mai mult modelul, pe când, setul de testare (test), este necesar pentru a valida progresul algoritmilor, adică cât de bine a mers antrenarea. Sunt mai multe metode de a împărți setul de date în antrenare și testare, dar cel mai comun este 80:20. Setul de antrenare va fi întotdeauna cel mai mare pentru o mai bună analiză. Este recomandat ca împărțirea să fie aleatoare pentru ca rezultatele să fie mai bune. După această împărțire, am ales ca variabila dependentă să fie emigranții temporari “emigranti”, iar variabilele independente să fie populația rezidentă “pop_rezidenta”, rata de cazuri confirmate “confirmati_rata”, rata deceselor “decedati_rata”, rata vindecărilor “vindecati_rata”, respectiv rata șomajului “unemployed_rate”. S-a încercat explicarea influenței cazurilor de Coronavirus și a ratei șomajului asupra emigrării temporare folosind ca model de analiză, regresia.

Regresia este parte a Învățării Automate supervizate și este un instrument util pentru a prezice valori cantitative. Regresia liniară poate fi de două tipuri. Atunci când încercăm să prezicem o variabilă dependentă sau de output pe baza unei variabile independente sau de input, atunci regresia liniară este simplă. Dacă încercăm să prezicem o variabilă dependentă sau de output pe baza a mai multor variabile independente sau de input, atunci vorbim despre o regresie liniară multiplă sau multiliniară. Regresia este ca model, o linie. Ceea ce presupune modelul este ca regresia liniară să găsească cea mai bună linie dintre variabila de input și cea de output, astfel încât predicția să fie cât mai corectă, iar erorile cât mai minimizate. Erorile reprezintă diferența dintre valoarea reală și cea prezisă de model. Scopul este ca diferența să fie cât mai mică. Pentru ca prezicerea să fie cât mai corectă, relația trebuie să fie una cât mai liniară, să existe corelație între variabila de output și cea sau cele de input. De asemenea, distribuția trebuie să fie una normală.

Așadar, pentru a aplica modelul de regresie liniară, prima dată am normalizat datele calculând scorul Z. Mai apoi, am ajustat modelul de regresie liniară pe subsetul de date train și am calculat coeficienții modelului (interceptul și panta dreptei de regresie). După acest pas, am mai calculat și predicțiile modelului pe subsetul de test. Nu am mai continuat să calculez ceea ce ține de partea de validare a modelului, deoarece acestea nu fac parte din sarcina mea. Pentru a avea termen de comparație, am mai aplicat și metoda Stochastic Gradient Descend. Gradient Descent este atunci când fiecare pas se aplică după recalcularea parametrilor pentru un singur sample de date. Gradient Descent se folosește ca să optimizeze acel criteriu care este minimizarea funcției de cost. De asemenea, se mai utilizează atunci când parametrii nu pot fi calculați analitic și se caută un algoritm de optimizare. Modul de a folosi Gradient Descent este de a se încerca diferite valori ale coeficienților, evaluarea costurilor și selectarea unor noi coeficienți care au un cost mai bun, adică mai mic. Diferența dintre Gradient Descent și Regresia Liniară este că regresia liniară folosește acea formă/soluție explicită ca să calculeze beta-urile, pe când Gradient Descent nu folosește o soluție explicită ca să învețe din date, ci folosește minimizarea funcției de cost.

La fel ca și în cazul regresiei liniare, am ajustat modelul pe setul de date train, am calculat coeficienții și mai apoi predicțiile pe setul de date test. La final, după aplicarea celor două modele, am transformat modelele în două fișiere distincte de tip pickle (.pkl), și le-am încărcat împreună cu notebook-ul pe GitHub gata pentru partea de validare.

5. Validare model

În cadrul acestui pas, am importat datele finale în MongoDB și am făcut legătura cu DataSpell pentru a putea începe urmărirea parcursului pașilor anteriori. Astfel, pentru fiecare model de regresie ales de Camelia prin selectarea modelelor, am decis să calculez o serie de erori (mean square error, mean absolute error și root mean square error), coeficientul regresional R2 și procentele de acuratețe ale antrenării și testării, toate acestea fiind calculate pentru ambele modele (modelul de regresie liniară și modelul Stochastic Gradient Descend). Am ales acești indicatori deoarece sunt relevanți în alegerea celui mai bun model, ei putând fi comparați ușor.

Definirea indicatorilor folosiți în alegerea celui mai bun model, vor fi de folos în înțelegerea deciziei luate. Astfel că: „mean square error" ne spune cât de aproape față de dreapta de regresie este setul nostru de puncte prin măsurarea distanței de la punct la dreaptă. Toate acestea sunt ridicate la pătrat pentru a evita valorile negative și de a da greutate valorilor mai mari; „mean absolute error" este media erorilor absolute, unde erorile absolute sunt diferențele dintre erorile valorilor măsurate și erorile valorilor reale; „root mean square error" ne arată abaterile standard ale rezidurilor, adică măsoară cât de împrăștiate sunt rezidurile. Coeficientul de determinare R2 este folosit pentru a analiza cum diferențele dintr-o variabilă pot fi explicate de diferențele dintr-o altă variabilă, sau în cazul nostru variabile. Mai exact, cât la sută din varianța variabilei dependente (emigranți temporari) este explicată de varianța variabilelor independente (populația rezidentă, rata de cazuri confirmate, rata deceselor, rata vindecărilor, respectiv rata șomajului).

Tabel 1: Metode de validare modele

	MAE	**MSE**	**RMSE**	**R^2**	**Training Accuracy**	**Testing Accuracy**
LINEAR REGRESSION	270.57	105084.59	324.17	0.96	99.24	97.28
SGD REGRESSION	378.18	208314.74	456.41	0.92	95.96	92.00

Toate cele de mai sus au fost calculate pe ambele modele, iar mai apoi comparate rezultatele. Modelul final ales ramâne regresia liniară, această alegere fiind explicată prin valoarea coeficientului R2 și prin diferențierea acurateții modelelor. Varianța variabilei dependente este explicată în proporție de 95,9% de varianța variabilelor independente în cazul regresiei liniare, pe când în cazul regresiei SGD varianța variabilei dependente în funcție de varianța variabilelor independente este explicată în proporție de 92%. Astfel, modelul regresiei liniare față de modelul regresiei SGD, pe baza coeficientului R2, este mai bun cu 4%. Totodată, toate erorile explicate mai sus, se remarcă a fii mai mici la regresia liniară: mean absolute error este de 270,6 în cazul regresiei liniare, mai mic cu 108 față de regresia SGD, mean square error este cu 103.230 mai aproape de 0 în cazul regresiei liniare

față de cea SGD. Și în cazul RMSE diferențele se arată a fii la fel, regresia liniară având un coeficient mai bun. Procentele de acuratețe se remarcă din nou, a fi mai mari în cazul regresiei liniare. Aici, diferentele sunt de 1,96% în cazul antrenării și 3,96 în cazul testării.Iar diferențele dintre antrenare si testare, sunt mai mici în cazul regresiei liniare, acestea fiind egale cu 3,28%, față de 5,28.

6. Creare pipeline

În cadrul acestei etape, după introducerea celor necesare pentru conectarea la echipă prin Git clone, dar și a introducerii prin MongoDB a csv-ului ca final_data, s-a început crearea propriu zisă a ML Pipeline-ului pentru prezicerea modelului. Am importat un fișier de tip pickle**,** ce consta în modelul nostru valid creat anterior. După importarea librăriilor necesare am folosit doi pași, și anume Strandard Scaler pentru standardizarea datelor și normalizarea valorilor, lucrând pe datele nenormalizate de Camelia și Linear Regression, adică modelul ales ca fiind valid. După acest pas, a urmat verificarea acurateții Pipeline-ului construit prin scorul acesteia, care are o valoare mare de 0,966, lucru care ne determină să înaintăm cu analiza mai departe.

În următorul pas am realizat vizualizarea pipeline-ului în care am afișat și folosit textul diagramei prin ***set_config*** care mi-a descris ce curpinde acest Pipeline. După vizualizarea textului, am apelat și la o diagramă pentru o vizualizare mai bună a pașilor mei.

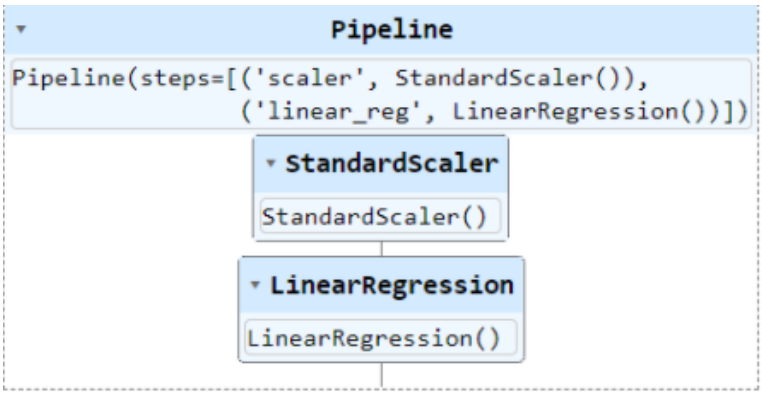


Fig 18: Diagrama Pipeline

Ultimul pas a fost stocarea pipeline-ului creat sub forma unui fișier de tip ***.joblib***, într-un folder denumit „***Pipeline***" pe care l-am creat în folderul echipei noastre. Acum

pipeline-ul cuprinde un model de regresie liniară, care va putea fi distribuit pentru a fi folosit oricând vom mai avea nevoie de acesta.

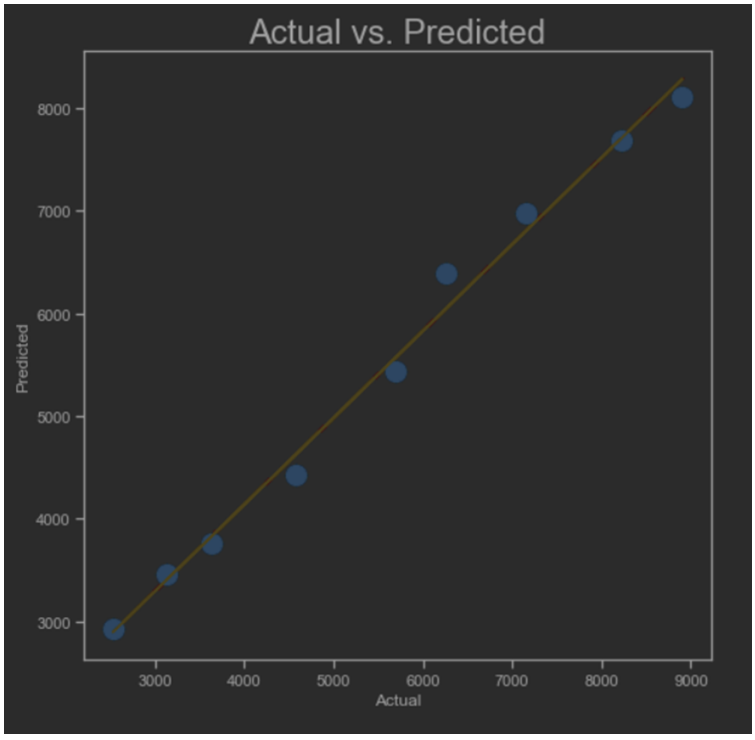
7. Predicție cu pipeline

În cadrul acestui pas, am importat baza de date actualizată în MongoDB și apoi am făcut conexiunea cu DataSpell. Am importat pipeline-ul creat la pasul anterior și am ajustat modelul valid pe setul de date de antrenare. Am folosit pipeline-ul pentru a face predicția variabilei dependente "emigranți" pe setul de date de testare. Predicția am transformat-o într-un DataFrame pentru a putea vizualiza datele actuale și cele prezise într-un tabel. Pentru o vizualizare mai bună a datelor, am importat librăria matplotlib și am făcut un scatter plot. Din acesta putem observa că datele noastre prezise urmează același curs cu cele reale și că valorile prezise sunt apropiate de cele reale. Pentru datele prezise și cele reale am creat un .csv pe care l-am incarcat în folderul cu date.

Tabel 2: Valori actuale și prezise ale emigranților temporari

Actual Value	**Predicted Value**
5695	5431.894983
2526	2927.042047
8220	7686.048766
7153	6979.808199
3137	3461.038175
8907	8114.534781
6255	6390.847706
4583	4432.939655
3636	3755.374663

Fig 19: Scatterplot valori actuale vs. valori prezise



8. Concluzii

În urma analizei pe care am făcut-o seturilor de date menționate în prima parte a acestui raport, putem afirma faptul că există o relație liniară între variabilele explicative și cea explicată. Astfel, prezența cazurilor mediatizate de Covid-19, reprezentate de ratele de confirmați, decedați și vindecați, împreună cu ratele de șomaj și populația rezidentă pot explica semnificativ variația numărului de emigranți temporari, în cadrul județelor din România. Această generalizare nu o putem face, însă, în cazul județului Suceava, deoarece acesta a fost exclus din analiză, fiind un outlier care influența mult calitatea modelului. Propunerea noastră ar fi ca o analiză următoare să fie dedicată exclusiv acestui județ, pentru a testa și analiza fenomenul, sau anomaliile, care au dus la identificarea acestuia ca un outlier. De asemenea, se va putea testa dacă modelul propus și validat în cadrul acestui proiect este în continuare de actualitate și calitate, când vine vorba de creșterile enorme detectate în numărul de cazuri de Covid-19, care au fost mediatizate. Aceasta o vom considera ca o provocare pentru un viitor focus de studiu.