**RAČUNARSTVO USLUGA I ANALIZA PODATAKA**

SEMINARSKI RAD

**PREDVIĐANJE USPJEŠNOSTI TELEMARKETINGA BANKE**

Studenti:

Petar Bagarić

Tomislav Jukić

Sadržaj

[1.Uvod 3](#_Toc461839856)

[2. Opis problema 3](#_Toc461839857)

[2.1. Korišteni podaci 3](#_Toc461839858)

[3. Klasifikatori 4](#_Toc461839859)

[3.1. Two-Class Decision Jungle 4](#_Toc461839860)

[3.2. Two-Class Neural Network 4](#_Toc461839861)

[3.3. Two Class Averaged Perceptron 5](#_Toc461839862)

[3.4. Two Class Boosted Decision Tree 5](#_Toc461839863)

[4. Programsko rješenje 6](#_Toc461839864)

[4.1. Izrada eksperimenta u AzureML 6](#_Toc461839865)

[4.2. Usporedba modula 9](#_Toc461839866)

[4.3. Izrada desktop aplikacije u C# 11](#_Toc461839867)

[5. Zaključak 14](#_Toc461839868)

[6. Literatura 15](#_Toc461839869)

# 1.Uvod

Tema ovog seminarskog rada je predviđanje uspješnosti telemarketinga banke, odnosno cilj rada je izraditi aplikaciju u programskom jeziku *C#* koja na osnovu određenih parametara predviđa uspješnost prodajnog procesa banke. Za uspješnu realizaciju navedenog projekta koristili smo Microsoft Azure Machine Learning i Visual Studio. **Microsoft Azure Machine Learning** je cloud platforma  koja omogućuje**data scientistima** i **developerima** jednostavnu izradu prediktivnih modela kroz web sučelje. Microsoft Azure Machine Learning Studio je drag&drop alat koji koristimo za izradu, testiranje i implementaciju prediktivnih analitičkih rješenja te implementiranje istih unutar poslovnih aplikacija [1]. Navedenu platformu koristili smo za treniranje modela koji obavlja klasifikaciju. Što se tiče Visual Studio alata, on predstavlja seriju Microsoftovih alata za razvijanje softvera te je on iskorišten za izradu desktop aplikacije. Seminarski rad će prikazati opis problema projekta te detalje njegove realizacije.

# 2. Opis problema

Problem ovog rada predstavlja predviđanje uspješnosti telemarketinga banke prema klasifikaciji klijenata s obzirom na parametre prikupljenje tijekom procesa prodaje. Parametri su podijeljeni s obzirom na dob, bračni status, zaposlenje, stupanj obrazovanja, kreditno stanje i stanje bankovnog računa, hipoteku i privatna zaduženja, način, datum i trajanje ostvarenog kontakta s pojedinim klijentom, broj ostvarenih razgovora sa svakim pojedinim klijentom, razdoblje između ostvarenih razgovora te ostvarenje ciljeva prijašnjih razgovora. Svaki od navedenih parametara nosi određenu težinu za krajnji rezultat te će se rad na tu temu kasnije detaljnije usmjeriti.

## 2.1. Korišteni podaci

Za realizaciju ovog projekta dodjeljeni su nam podaci s linka <https://archive.ics.uci.edu/ml/datasets/Bank+Marketing>. Od ponuđenih skupova podataka, odabrali smo skup koji obuhvaća 4500 redaka pri čemu svaki redak predstavlja svakog pojedinog klijenta s njegovim bitnim parametrima tijekom ostvarenog, odnosno ostvarenih kontakata. Za obradu podataka koristili smo Online Converter iz XLS formata u CSV format.

# 3. Klasifikatori

Klasifikator predstavlja algoritam koji na temelju određenih značajki određuje klasu ulazne teksture te se odgovarajući klasifikator odabire na temelju količine korištenih značajki. Za ovaj projekt korištena je veća količina klasifikatora te smo kasnije izabrali četiri najpogodnija klasifikatora; Two-Class Decision Jungle, Two-Class Neural Network, Two-Class Averaged Perceptron i Two-Class Boosted Decision Tree koji su objašnjeni nadalje u radu.

## 3.1. Two-Class Decision Jungle

Two-Class Decision Jungle ili šuma odlučivanja je modul za stvaranje modela strojnog učenja koji se temelji na nadziranom algoritmu učenja , a sastoji se od više stabala odlučivanja. Zbog takvog načina izvedbe, šuma odlučivanja omogućuje dolazak do određenog lista stabla pomoću više puteva. Prednosti ovog modula su to što omogućuje spajanje grana stabala odlučivanja te se na takav način osiguravaju manji memorijski zahtjevi te se ostvaruju kvalitetnije generalizacijske izvedbe od stabla odlučivanja. Druga prednost modula je izvođenje integriranih značajki obilježja i klasifikacije uz veću otpornost na šumove i smetnje [2].

## 3.2. Two-Class Neural Network

Two-Class Neural Network ili neuronska mreža predstavlja modul za kreiranje neuronske internetske mreže koja se koristi za predviđanje ciljeva sa dvije vrijednosne varijable. Kao primjer navedenog, može se navesti predviđanje medicinskih scenarija poput određivanja bolesti koju određeni pacijent može ili ne mora imati. Klasifikacija pomoću neuronskih mreža je nadzirano strojno učenje što znači da podaci za treniranje uključuju željene izlaze, a obavlja se predviđanje izlaza za buduće ulaze. Budući da umjetna neuronska mreža predstavlja skup međusobno povezanih slojeva ulazi dovode do izlaza pomoću niza rubova i čvorova pri čemu svaki od njih ima svoju određenu težinu. Istraživanja su pokazala da neuronske mreže mogu biti vrlo učinkovite u rješavanju složenih zadataka kao što su slike i prepoznavanje govora budući da se u takvim zadacima uzastopni slojevi koriste za modeliranje odnosno povećavanje razina semantičkih dubina[2].

## 3.3. Two Class Averaged Perceptron

Two-Class Averaged Perceptron ili prosječni perceptron predstavlja modul za stvaranje modela strojnog učenja koji koristi prosječni perceptron algoritam, a klasifikacija modula kao nadzirana metoda učenja zahtjeva označeni skup podataka koji uključuje stupac oznaka. Trenirani model se može koristiti za predviđanje vrijednosti novih ulaznih primjera. Ulazi su klasificirani u nekoliko mogućih izlaza na temelju linearne funkcije, nakon čega se izvršava kombiniranje s nizom težina izvedenih od značajki vektora. Jednostavniji modeli perceptrona su pogodni za učenje linerano rastavljivih uzoraka, dok neuronske mreže, a osobito one duboke, mogu oblikovati složenije klasne granice. Međutim, perceptron je brži modul te budući da se proces izvodi serijski perceptron se može koristiti s kontinuiranim treningom [2].

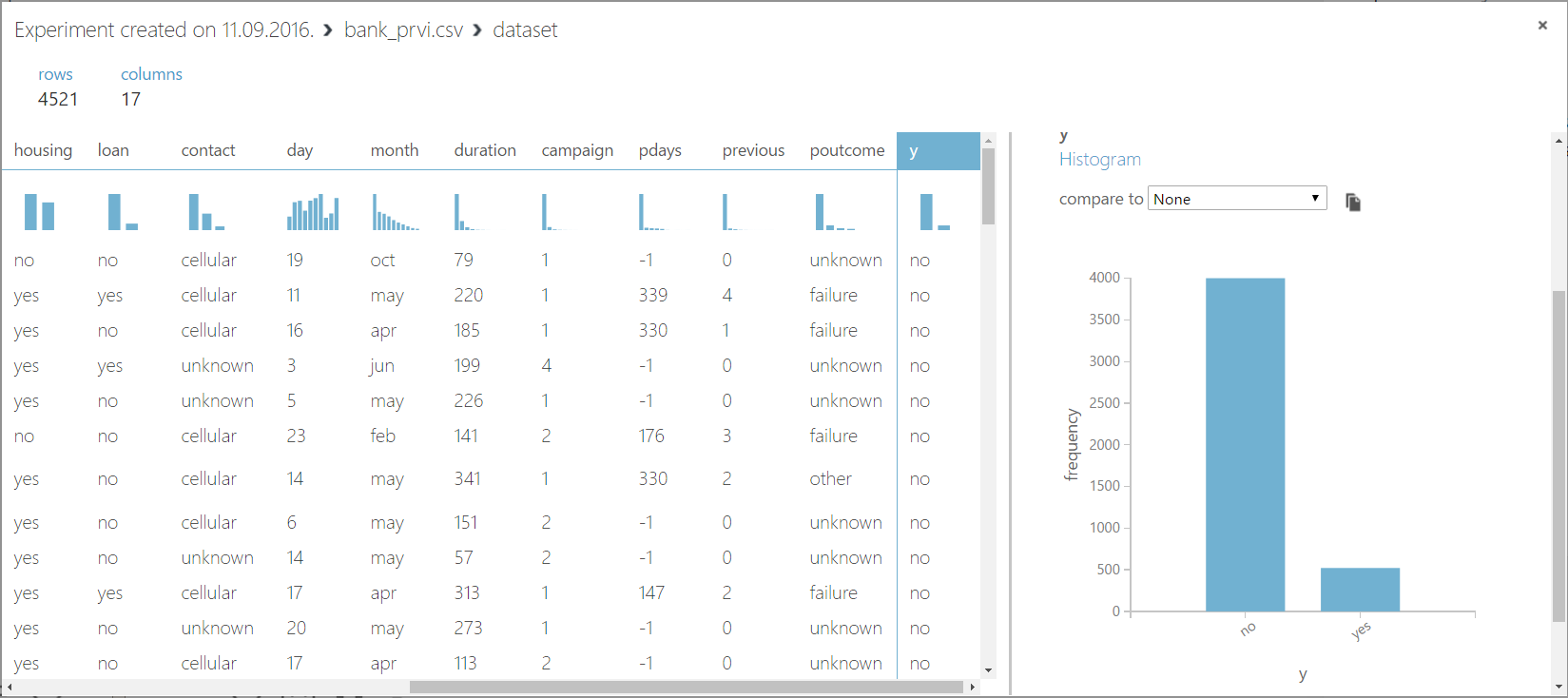
## 3.4. Two Class Boosted Decision Tree

Two-Class Boosted Decision Tree ili pojačano stablo odlučivanja je modul za kreiranje modela strojnog učenja koji se temelji na algoritmu pod nazivom pojačano stablo odlučivanja koji funkcionira na način da drugo stablo odlučivanja ispravlja pogreške prvog stabla, treće stablo ispravlja pogreške prvog i drugog stabla i tako dalje. Predviđanje je utemeljno na cijelom skupu stabala. Općenito, kada je podešena pravilna konfiguracija, pojačano stablo odlučivanja se očituje kao najlakša metoda koja osigurava vrhunske izvedbe na različitim zadacima strojnog učenja. Međutim, ovaj modul je jedan od modula s više memorijskih zahtjeva zato jer izvršavanje zadržava sve u memoriji te samim time pojačano stablo odlučivanja nema mogućnost obrade velikih skupova podataka [2].

# 4. Programsko rješenje

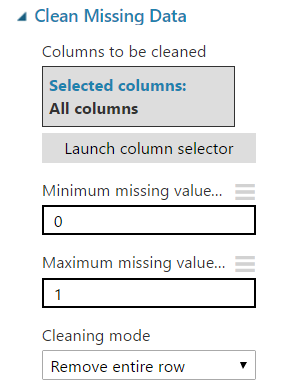
## 4.1. Izrada eksperimenta u AzureML

Izrada eksperimenta u AzureML je započela kreiranjem novog eksperimenta u navedenom cloud servisu te se nadalje unose podaci za uspješnost predviđanja telemarketinga banke u željenom formatu, u ovom slučaju formatu CSV kao što je prikazano na slici 4.1.



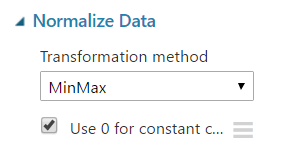
Slika 4.1.

Nakon unosa podataka dodaje se element Clean Missing Data te se odabire opcija Remove entire row za one podatke koji su nepotpuni kao što možete vidjeti na slici 4.2.



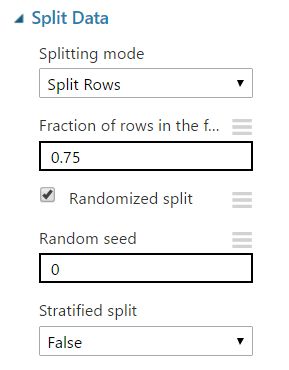
Slika 4.2.

Dodavanjem elementa Normalize data postiže se normalizacija podataka u ujednačene raspone što olakšava stvaranje modela, odnosno smanjuje potrebu za regularizacijom. Pri dodavanju navedenog elementa potrebno je kod opcije Transformation method odabrati MinMax (Slika 4.3.).



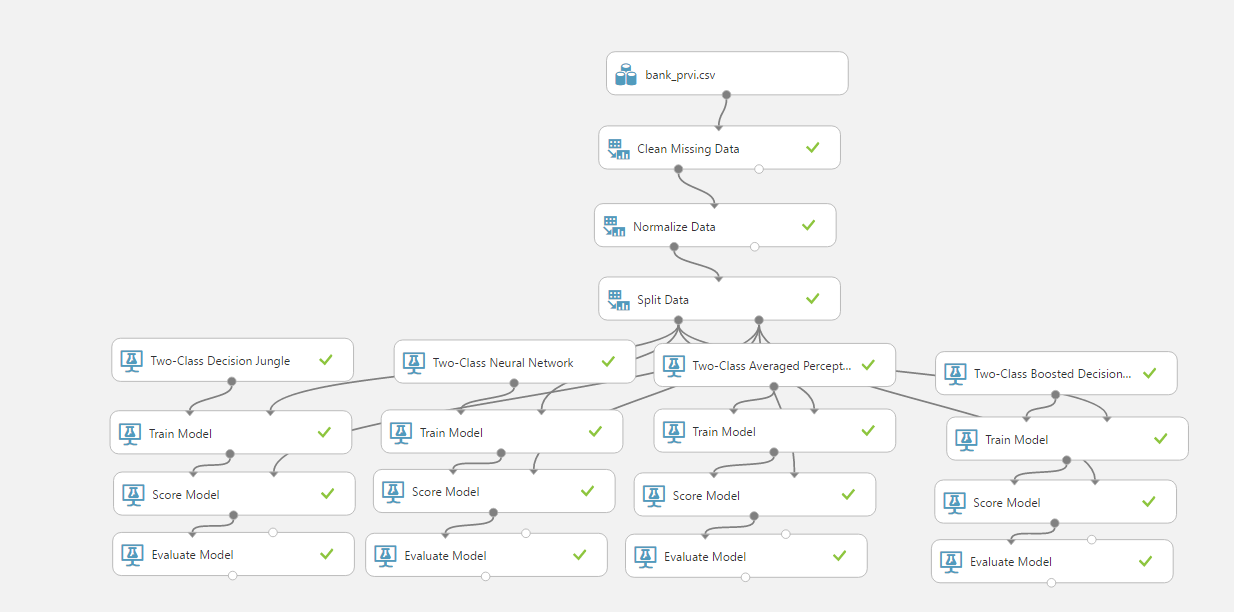
Slika 4.3.

Prije određivanja klasifikacije projekta, skup podataka se dijeli u podskupove za trening i testiranje te se samim time ostvaruje neovisno vrednovanje modela na temelju podataka koji nisu bili viđeni tijekom samog treninga. Navedeni postupak se ostvaruje dodavanjem elementa Split data te se odabire postotak podataka koji će se iskoristiti za trening, dok se ostali podaci usmjeravaju na proces testiranja. Obično podskup podataka za trening dobiva veći postotak, kao što se dogodilo i u slučaju našeg projekta, a postotak je iznosio 75%, dok ostalih 25% predstavlja podatke za testiranje. Navedeno se može vidjeti na slici 4.4.



Slika 4.4.

Nakon toga izvršava se postupak odabira više modula u našem eksperimentu. Iako smo koristili više modula u prvoj inačici eksperimenta, u konačnoj inačici eksperimenta odabrali smo četiri najpogodnija modula. Svaki modul ima svoje elemente; Train Model i Score Model te Evaluate Model. U Train Model elementu je potrebno odabrati stupac koji predstavlja ovisnu varijablu, a ta varijabla je u našem slučaju varijabla y. Varijabla može kao rezultat imati yes ili no, odnosno yes predstavlja uspješnu prodaju, dok no predstavlja neuspješnu prodaju. Konačni izgled eksperimenta je prikazan na slici 4.5.



Slika 4.5.

Sljedeći korak podrazumijeva pokretanje samog eksperimenta pomoću opcije Run koja se nalazi na dnu prozora te je potrebno pričekati da završi trening modela. Dobiveni rezultati se mogu vidjeti na Score model elementu.

## 4.2. Usporedba modula

Nakon dobivenih rezultata klasifikacijskih modula, izvršava se njihova usporedba korištenjem Evaluate model elementa. Tablica 4.6. prikazuje dobivene rezultate za svaki pojedini modul te se na osnovu rezultata odabire najbolji modul.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **True Positive** | **False Negative** | **False Positive** | **True Negative** | **Recall** | **F1 Score** | **Precision** | **Accuracy** |
| **Two Class Decison Jungle** | 43 | 76 | 27 | 984 | 0.361 | 0.455 | 0.614 | 0.909 |
| **Two Class Neural Network** | 45 | 74 | 33 | 978 | 0.378 | 0.457 | 0.577 | 0.905 |
| **Two Class Averaged Perceptron** | 42 | 77 | 24 | 987 | 0.353 | 0.454 | 0.636 | 0.911 |
| **Two Class Boosted Decision Tree** | 70 | 49 | 59 | 952 | 0.588 | 0.565 | 0.543 | 0.904 |

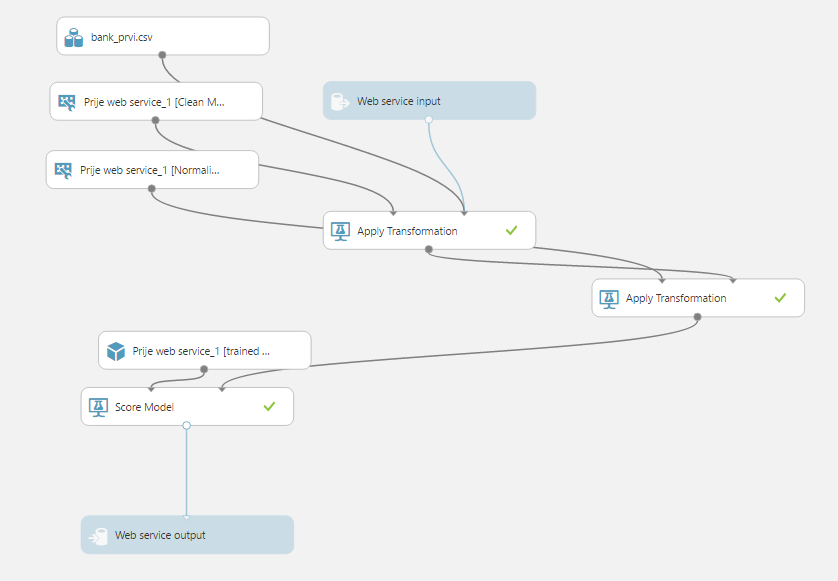
Tablica 4.6.

Kao što prikazuje tablica 4.6. najbolji modul je Two Class Averaged Perceptron s točnošću od 91.1%. Na njegovom Train model elementu imamo opciju pomoću koje vidimo težine parametara kako je sam modul odredio prema prijašnjim postupcima (Slika 4.7).



Slika 4.7.

Kao što možemo vidjeti daleko najbitniji parametar je sama dužina razgovora s klijentom kao što smo mogli i predvidjeti jer je logično ako je vremenski interval razgovora veći to je veća i zainteresiranost klijenta, a samim time i veća mogućnost realizacije uspješnosti prodaje. Također, iz tablice je vidljivo da postoji razlika između parametra nezaposlenosti i parametra mirovine, odnosno nezaposlenost klijenta ne doprinosti uspješnosti prodaje, dok s druge strane umirovljenički status klijenta pogoduje ostvarenju same prodaje. Nadalje, odabiremo Train model najtočnijeg modula i opciju Deploy web service iz alatne trake. Rezultat navedenog postupka prikazuje slika 4.8. te sam web servis ima mogućnost testiranja odabirom opcije Test. Također postoji i opcija Request/Response koja omogućuje dobivanje potpunog opisa API-ja koji je iskorišten za izradu desktop aplikacije.

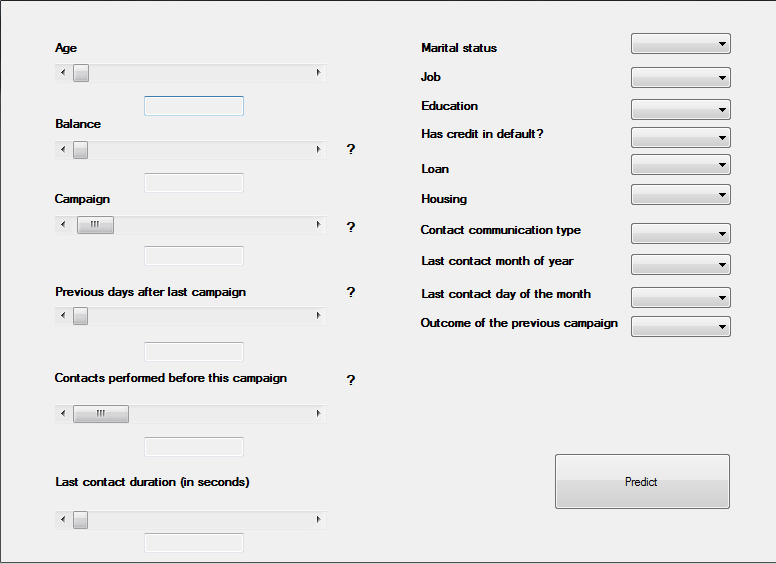


Slika 4.8.

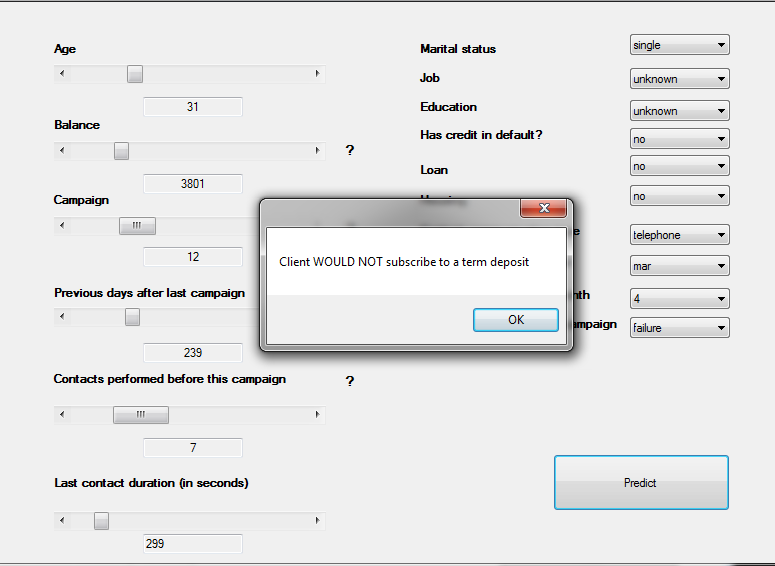
## 4.3. Izrada desktop aplikacije u C#

C# aplikacija se sastoji od forme u kojoj se mogu unijeti podaci za svakog korisnika. Nakon unošenja podataka klikom na gumb „Predict“ dobije se predviđanje o tome da li bi taj korisnik pristao na oročenu štednju neke banke. Na mjesta gdje se unose numeričke vrijednosti su korišteni slideri umjesto klasičnih textbox-ova radi lakšeg kontroliranja iznimaka.

Nakon pritiska gumba „Predict“ poziva se funkcija *InvokeRequestResponseService(data)* unutar klase *RequestResponse* koja služi za komunikaciju sa servisom. Ona šalje serveru naše podatke i čeka odgovor, a server nakon što obradi zahtjev šalje rezultat, koji je u našem slučaju yes/no.



Slika 4.9. Sučelje korisničke aplikacije



Slika 5.0. Rezultat predviđanja

Server kojemu šaljemo podatke radi s matricama, te je nužno naše podatke kopirati u string matricu. To je opisano u idućem dijelu koda.

|  |  |
| --- | --- |
| Linija | Kod |
| 1: | string[,] dataTosend = new string[1, data.Length]; |
| 2: | int i = 0; |
| 3: | foreach (string temp in data) |
| 4: | { |
| 5: | dataTosend[0, i] = temp; |
| 6: | i++; |
| 7: | } |

Kod koji vraća grešku u slučaju ne popunjavanja svih polja:

|  |  |
| --- | --- |
| Linija | Kod |
| 1: | foreach (Control c in this.Controls) |
| 2: | { |
| 3: | if (c is TextBox) |
| 4: | { |
| 5: | TextBox textBox = c as TextBox; |
| 6: | if (textBox.Text == string.Empty && flag == false) |
| 7: | { MessageBox.Show("Fill in all the answers"); |
| 8: | flag = true; |
| 9: | } } |
| 10: | flag = false; |

Kod koji služi za pronalaženje stringa koji prikazuje traženi rezultat :

|  |  |
| --- | --- |
| Linija | Kod |
| 1: | private string getY(string output) |
| 2: | { |
| 3: | int n1 = output.Length - 1; |
| 4: | while (output[n1] != '"') n1--; |
| 5: | int n2 = n1 - 1; |
| 6: | while (output[n2] != '"') n2--; |
| 7: | n1 = n2 - 1; |
| 8: | while (output[n1] != '"') n1--; |
| 9: | n2 = n1 - 1; |
| 10: | while (output[n2] != '"') n2--; |
| 11: | string predictedClass = output.Substring(n2 + 1, n1 - n2 - 1); |
| 12: | return predictedClass; |
| 13: | } |

# 5. Zaključak

Prilikom izrade ovog rada korišteno je više tehnologija. Klasifikacija se odvija pomoću neuronske mreže u Azure-ovom studiju za strojno učenje, što se poslije koristi u C# aplikaciji koja predviđa da li bi neka osoba pristala na oročenu štednju određene banke u ovisnosti o unesnim podacima. Iako rezultati nisu apsolutno točni, daju određenu predodžbu bankama o tome koliko je uspješan njihov telemarketing.

Link na github projekta: <https://github.com/pbagaric1/Klasifikacija-uspjesnosti-telemarketinga-banke-Bagaric-Jukic>

# 6. Literatura

[1] <http://blog.inteligencija.com/prediktivna-analitika-upogonjena-microsoft-azure-ml-platformom/>

[2] <https://msdn.microsoft.com/en-us/library/azure/dn905976.aspx>

[3] <https://msdn.microsoft.com/en-us/library/azure/dn905947.aspx>