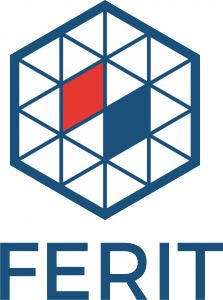
**Sveučilište Josipa Jurja Strossmayera u Osijeku**

**Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku**

**Računarstvo usluga i analiza podataka**

**Projektni zadatak**

**Klasifikacija životinja u zoološkom vrtu**

**Izradio :** Filip Ćuić,

Sveučilišni diplomski studij računarstva, modul DRD

**Mentori** : Prof.dr.sc. Goran Martinović,

Izv.prof.dr.sc. Alfonzo Baumgartner,

Dr.sc. Bruno Zorić,

Dr.sc. Dražen Bajer

Sadržaj rada

[1. Uvod 1](#_Toc18228770)

[2. Pregled područja i problematike 2](#_Toc18228771)

[2.1 Strojno učenje 2](#_Toc18228772)

[2.2.1 Nadzirano učenje 3](#_Toc18228773)

[2.2.2 Nenadzirano učenje 4](#_Toc18228774)

[2.2.3 Podržano učenje 5](#_Toc18228775)

[3. Opis problema 6](#_Toc18228776)

[3.1 Tijek rješavanja problema 6](#_Toc18228777)

[3.2 Korišteni podaci 7](#_Toc18228778)

[3.3 Korišteni postupci klasifikacije 9](#_Toc18228779)

[3.3.1 Multiclass Decision Forest 10](#_Toc18228780)

[3.3.2 Multiclass Decision Jungle 11](#_Toc18228781)

[3.3.3 Multiclass Logistic Regression 11](#_Toc18228782)

[3.3.4 Multiclass Neural Network 12](#_Toc18228783)

[3.4 Rezultati klasifikacije 14](#_Toc18228784)

[3.4.1 Testiranje funkcionalnosti modela 15](#_Toc18228785)

[4. Razvoj korisničkog programskog sučelja 16](#_Toc18228786)

[4.1 Opis programskog rješenja 16](#_Toc18228787)

[4.2 Konačni izgled eksperimenta 17](#_Toc18228788)

[4.3 Korištenje API-a 18](#_Toc18228789)

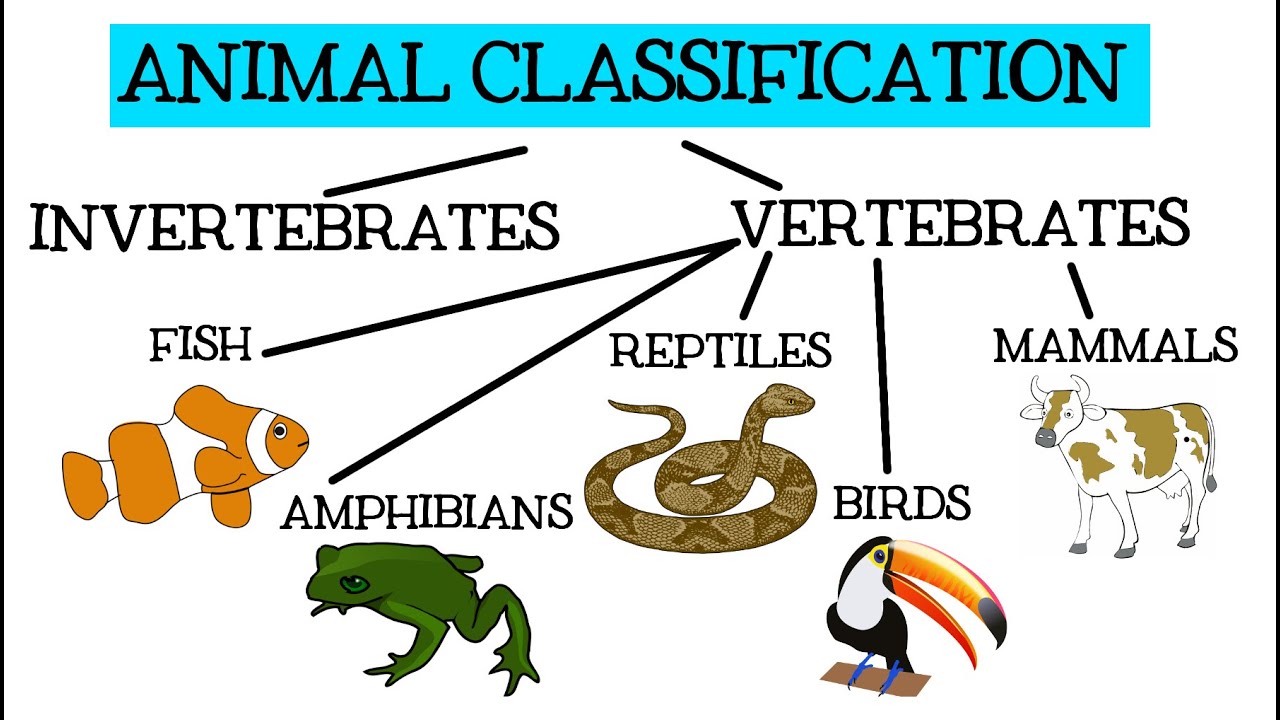
[5. Zaključak 20](#_Toc18228790)

[6. Literatura 21](#_Toc18228791)

[7. Prilozi 22](#_Toc18228792)

# Uvod

U ovom projektnom zadatku obrađuje se klasifikacija životinja u zoološkom vrtu na temelju gotovog skupa podataka s Kaggle stranice. Skup podataka sastoji se od 101 životinje koje su opisane sa 16 značajki (atributa). Prilikom klasifikacije koristit će se model strojnog učenja, to jest višeklasna neuronska mreža (*eng. Multiclass Neural Network*). Na temelju značajki, napravljeni model strojnog učenja pokušat će svrstati životinje u jednu od sljedećih klasa : sisavac, ptica, reptil, riba, vodozemac, kukac ili beskralježnjak. Projektni zadatak sastoji se od dva glavna dijela : izrade modela strojnog učenja na temelju gotovog skupa podataka preko web servisa Microsoft Azure Machine Learning Studio, te izrade korisničkog sučelja u Windows Forms aplikaciji. Cilj zadatka je predviđanje klase životinje koju korisnik unese. Korisnik će moći unijeti životinju koju želi, nasumično generirati životinju ili unijeti jednu od životinja iz zoološkog vrta koje će mu biti prikazane u aplikaciji i tako provjeriti ispravnost napravljenog modela. Način izrade modela strojnog učenja i web API-a, korisničke aplikacije, te način njihova povezivanja bit će opisan u nastavku.



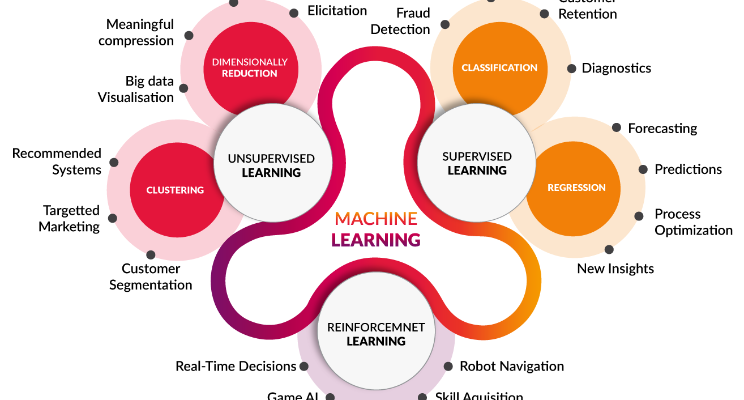
**Slika 1.** Klasifikacija životinja.

# Pregled područja i problematike

## Strojno učenje

Strojno učenje (*eng. machine learning*) je područje računalne znanosti koje je nastalo iz raspoznavanja uzoraka i računalne teorije učenja u području umjetne inteligencije. Strojno učenje je programiranje računala (sustava) tako da se optimizira neki kriterij uspješnosti na temelju prethodnog iskustva ili podatkovnih primjera. Glavni cilj strojnog učenja jest izgradnja modela koji je dobra i precizna aproksimacija podataka. Računalni sustavi koji koriste strojno učenje konstantno se prilagođavaju i uče iz podataka koji im se daju na raspolaganje.

Neke od primjena strojnog učenja su dubinska analiza podataka (*eng. data mining*) na ogromnim skupovima podataka (*eng. Big Data*), prepoznavanje objekata u području računalnog vida, optimizacija ponašanja robota u robotici, te prepoznavanje govora, pisanih znakova i sl. Najpreciznija i najkompleksnija primjena strojnog učenja su neuronske mreže koje se koriste kod rješavanja složenih problema nadziranog učenja (*eng. Supervised Learning*).



**Slika 2.1** Osnovna podjela strojnog učenja i neke od primjena strojnog učenja.

## 2.2.1 Nadzirano učenje

Nadzirano učenje (*eng. Supervised Learning*) je vrsta strojnog učenja gdje se pokušava odrediti nepoznata funkcionalna ovisnost između ulaznih veličina i izlaznih veličina. Kod skupa za učenje, svaki je ulaz označen svojom izlaznom veličinom, a algoritam koji se koristi pokušava pronaći vezu između ulaza i izlaza. Podatkovni primjeri su parovi sastavljeni od vektora ulaznih veličina i vrijednosti izlazne veličine **(x(i),y(i))**, a broj ulaznih veličina označava se s **n**, pa se vektor ulaznih veličina zapisuje u obliku : **x(i)=[x1(i),x2(i),...,xn(i)]T** .

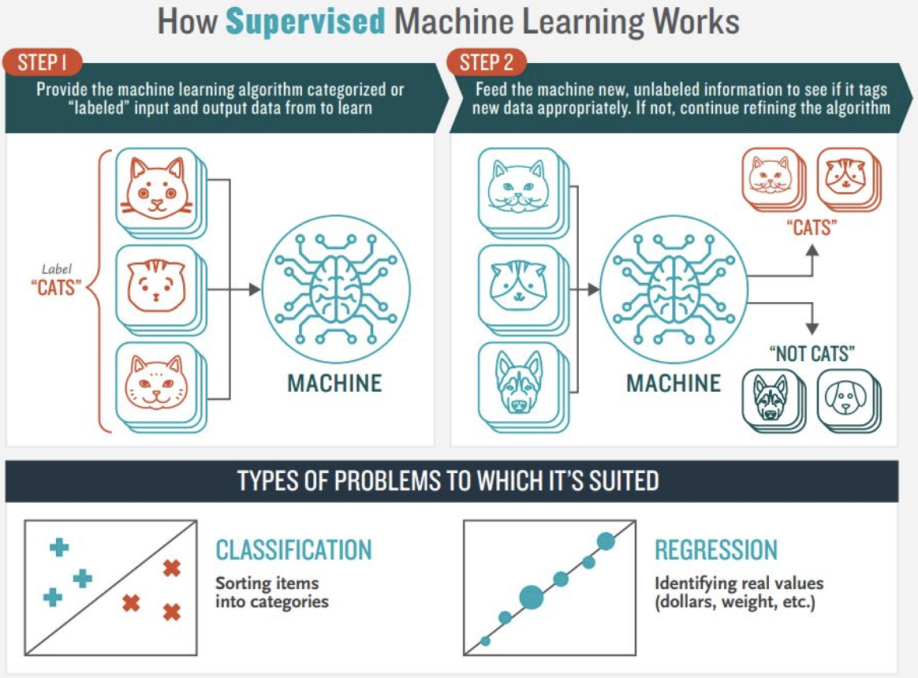
Skup za učenje je osnova nadziranog učenja, a obuhvaća raspoložive podatkovne primjere na temelju kojih se određuje nepoznata funkcionalna ovisnost primjenom jednog od algoritama strojnog učenja.

**Model dobiven nadziranim učenjem ima dvije osnovne namjene :**

**Predikciju** : - na temelju modela i vektora ulaznih veličina procijeniti vrijednost izlazne veličine

**Zaključivanje** : - odrediti kakav je oblik funkcionalne povezanosti između ulazne i izlazne veličine

Ako je izlazna veličina **y** prebrojiva, odnosno diskretna (kvantitativna) veličina, radi se o problemu klasifikacije, a ako je kontinuirana (neprebrojiva) veličina razmatra se problem regresije.



**Slika 2.2.1** Prikaz načina rada nadziranog strojnog učenja na konkretnom primjeru i njegove podjele.

## 2.2.2 Nenadzirano učenje

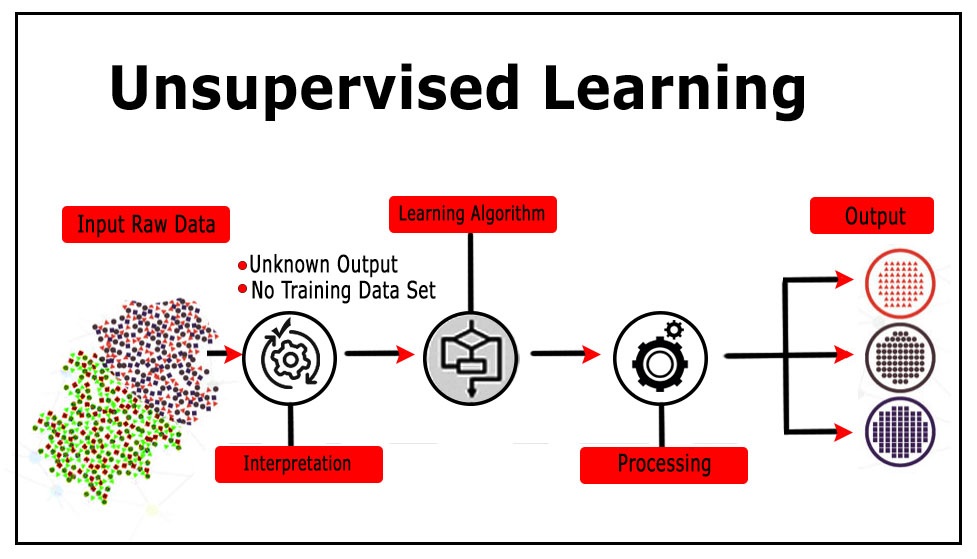
Kod nenadziranog učenja (*eng. Unsupervised learning*) podaci za treniranje sastoje se samo od ulaznih podataka, bez pripadajućih izlaza. Cilj nenadziranog učenja je pronaći određene pravilnosti u podacima, to jest otkriti strukture ili značajke podataka, te ih po njima grupirati u klastere (grupe), smanjiti dimenzionalnost podataka ili pronaći različite načine promatranja složenih podataka kako bi oni bili organiziraniji.

**Primjeri nenadziranog učenja :**

**Grupiranje podataka : -** grupiranje tržišta, grupiranje dokumenata po sličnosti, analiza društvenih mreža, kompresija slike itd.

**Smanjivanje dimenzionalnosti :** - kompresija i vizualizacija podataka

**Detekcija nepravilnosti podataka**



**Slika 2.2.2** Prikaz načina rada nenadziranog učenja.

## 2.2.3 Podržano učenje

Podržano učenje (*eng. Reinforcement learning*) je tip strojnog učenja gdje određeni programski agent poduzima akcije u nekom okruženju. Agent samostalno otkriva optimalno ponašanje metodom pokušaja i pogreški. On poduzima određenu akciju u cilju rješavanja problema, te nakon svake akcije dobiva povratnu informaciju (feedback) koja mu govori o napredovanju riješenosti problema, a u slučaju rješavanja problema dobiva nagradu. Nakon nekog broja pokušaja, agent bi trebao naučiti optimalan način odabira akcije u svakom koraku rješavanja problema, što je sekvenca koja maksimizira ukupnu nagradu.

**Primjeri podržanog učenja :**

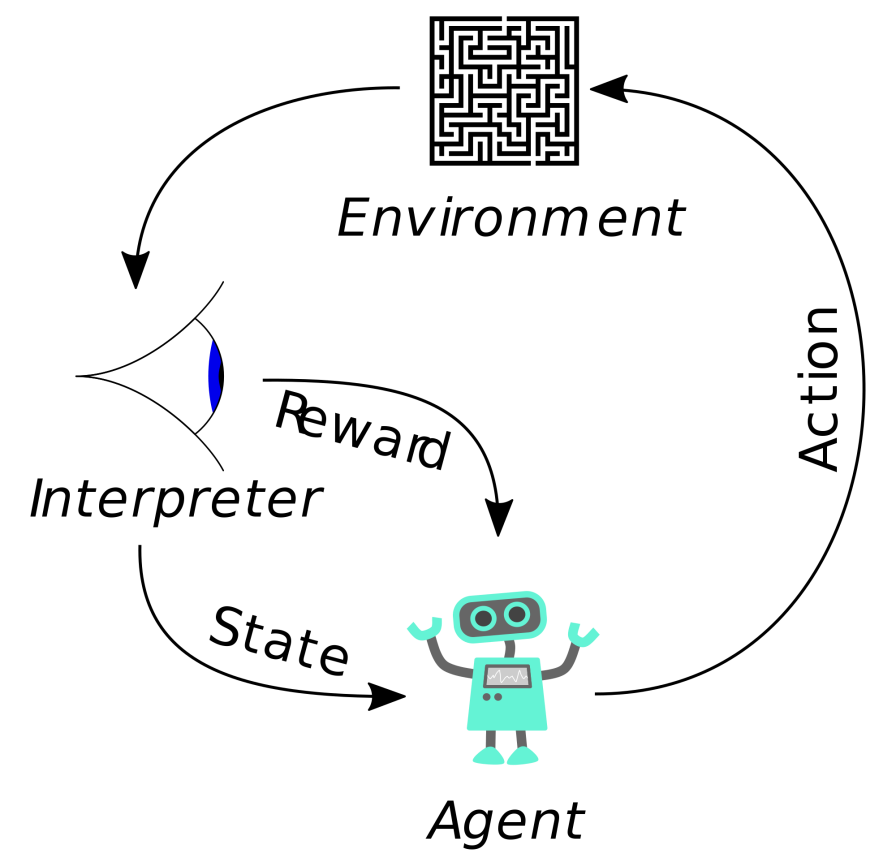
- robot koji uči slagati rublje

- roboti koji se uče kretati

- ponude i reklamiranje na webu

- umjetna inteligencija video igara

- robot koji igra šah



**Slika 2.2.3** Prikaz načina rada podržanog učenja.

# Opis problema

## 3.1 Tijek rješavanja problema

Zadani problem zahtjeva izgradnju modela strojnog učenja sa zoo.csv skupom podataka kao skupom za učenje. Model strojnog učenja kao cilj ima što precizniju procjenu klase unesene životinje na temelju njezinih značajki. Ulazne veličine bit će značajke životinja, a to su : dlaka, perje, jaja, mlijeko, zračna, vodena, grabežljivac, zubata, kralježnica, diše, otrovna, peraje, broj nogu, rep, domaća, veličina mačke, a izlazne veličina su klase životinja : sisavac, ptica, reptile, riba, vodozemac, kukac i beskralježnjak. Valja napomenuti kako su sve značajke skupa podataka osim klase životinja i broja nogu binarne vrijednosti, to jest, sve značajke osim navedenih mogu iznositi 0 i 1, što znači da životinja ima ili nema navedeno svojstvo. Model je treniran i testiran unutar Microsoft Azure Machine Learning Studia, točnije u eksperimentu pod nazivom ZOO Animal Classification. Nadalje, u eksperimentu su se odredile navedene ulazne veličine (značajke životinje) ,te izlazne veličine (klase) nakon čega se eksperiment razvio u predikcijski eksperiment. Za razvijanje predikcijskog eksperimenta nužno je odabrati jedan od četiri moguća algoritma strojnog učenja koji će se koristiti u predviđanju klase unesene životinje. Pošto se koristi višeklasna klasifikacija, moguća četiri algoritma su : šuma odluke (*eng. Decision Forest*), džungla odluke (*eng. Decision Jungle*), logistička regresija (*eng. Logistic regression*), te neuronska mreža (*eng. Neural Network*). Zbog stečenih znanja u području strojnog učenja, predikcijski eksperiment razvio se po višeklasnoj neuronskoj mreži jer je neuronska mreža najkompleksniji i najtočniji algoritam strojnog učenja. Rezultat je predikcijski eksperiment koji kao ulaze ima već navedene značajke životinja, a kao izlaz vjerojatnosti pripadanja svakoj od navedenih klasa i predviđenu klasu (rezultat) koja je ujedno i najveća dobivena vjerojatnost. Zbog povezivanja s korisničkim sučeljem, u Machine Learning Studiu nužno je napraviti web servis (Web API) koji će se pozivati svaki put kad korisnik unese ulazne podatke i kad izlazni podaci budu potrebni. Na korisničkom sučelju su se dodatno formatirali korisnički ulaz (ograničavanje ulaznih vrijednosti), odgovori API-a na dane zahtjeve, te izlazne vrijednosti aplikacije.

## 3.2 Korišteni podaci

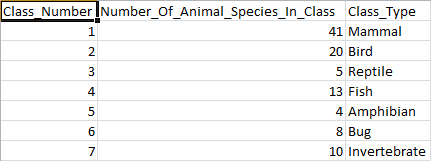
Korišteni podaci (zoo.csv skup podataka) pripada UCI Machine Learning repozitoriju, a preuzet je s Kaggle web stranice. Unutar skupa podataka postoji 101 životinja, 16 značajki i 7 mogućih klasa koje su navedene u poglavlju [1.](#_Uvod) i [3.1](#_3.1_Postupak_rješavanja) . Klase su označene brojkama od 1 do 7, zbog čega se unutar korisničkog sučelja trebalo raditi pretvaranje podataka za prikaz korisniku. Kod izgradnje modela strojnog učenja nije bila potrebna nikakva predobrada podataka, samo je bilo nužno raspodijeliti značajke skupa za učenje u ulazne i izlazne vrijednosti(opisano u [3.1](#_3.1_Postupak_rješavanja)).

**Slika 3.2** Prikaz učestalosti životinja pojedine klase unutar zoo.csv skupa za učenje modela.

Prije samog učenja modela, potrebno je dobro proučiti skup podataka na kojemu će se to napraviti. Kao što je i prikazano na [slici 3.2](#Učestalost), u skupu za učenje najviše je životinja koje pripadaju klasi sisavac, iz čega možemo zaključiti da će model točnije predviđati unesene životinje koje budu pripadnici te klase. Dodatne statistike i pojedinosti o skupu podataka dane su u prilogu zivotinje.xlsx.

**Slijedi pregled značajki životinja (s odgovarajućim formatom podataka) iz zoo.csv skupa za učenje :**

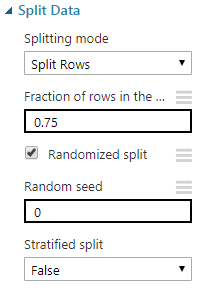
1. **Animal name** – ime životinje, niz znakova, jedinstveno za svaku instancu životinje
2. **Hair** – dlaka, binarna vrijednost, govori ima li životinja dlaku
3. **Feathers** – perje, binarna vrijednost, govori ima li životinja perje
4. **Eggs** – jaja, binarna vrijednost, govori nosi li životinja jaja
5. **Milk** – mlijeko, binarna vrijednost, govori proizvodi li životinja mlijeko
6. **Airborne** – zračna životinja, binarna vrijednost, govori je li životinja zračna
7. **Aquatic** – vodena životinja, binarna vrijednost, govori je li životinja vodena
8. **Predator** – grabežljivac, binarna vrijednost, govori je li životinja grabežljivac
9. **Toothed** – zubata životinja, binarna vrijednost, govori ima li životinja zube
10. **Backbone** – kralježnica, binarna vrijednost, govori ima li životinja kralježnicu
11. **Breathes** – disanje, binarna vrijednost, govori diše li životinja
12. **Venomous** – otrovna, binarna vrijednost, govori ima li životinja otrovne žlijezde
13. **Fins** – peraje, binarna vrijednost, govori ima li životinja peraje
14. **Legs** – broj nogu, cijeli broj koji može iznositi 0, 2, 4 i 8, govori koliko nogu ima životinja
15. **Tail** – rep, binarna vrijednosti, govori ima li životinja rep ili ne
16. **Domestic** – domaća životinja, binarna vrijednost, govori je li životinja domaća
17. **Catsize** – veličina mačke, binarna vrijednost, govori je li životinja veličine mačke
18. **Class\_type** – klasa, cijeli broj od [1,7], određuje pripadnost životinje određenoj klasi



**Slika 3.2.1** Prikaz vrijednosti atributa Class\_type i klase životinje na koju se cijeli broj odnosi, te broj životinja koje pripadaju svakoj klasi.

## 3.3 Korišteni postupci klasifikacije

Skup za učenje dijelio se na trening i testne podatke da bi se model naučio i testirao u cilju predviđanja klasa za unos nekih drugih vrijednosti. U praksi se kod učenja modela strojnog učenja preporučuju podjele podataka : (65%-35%), (70%-30%), (75%-25%) ili (80%-20%) posto podataka u korist skupa za treniranje u odnosu na skup za testiranje. U ovom projektnom zadatku koristila se podjela od 75%-25% podataka u korist skupa za treniranje jer se u praksi to pokazala kao najoptimalnija podjela. Tako podijeljeni skup zapravo znači da će se 75% instanci podataka cijelog ulaznog skupa koristiti za treniranje modela, a 25% za njegovo testiranje.

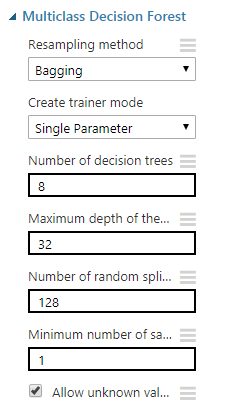


**Slika 3.3** Korištena podjela podataka kod kreiranja modela strojnog učenja u Microsoft Azure Machine Learning Studiu.

Nakon dijeljenja podataka, treniranja i testiranja modela, potrebno je napraviti višeklasnu klasifikaciju nekih drugih ulaznih podataka izvan skupa za učenje. Unutar Azure Machine Learning Studia postoje 4 moguća algoritma po kojima možemo napraviti klasifikaciju, a oni će biti opisani u nastavku.

## 3.3.1 Multiclass Decision Forest

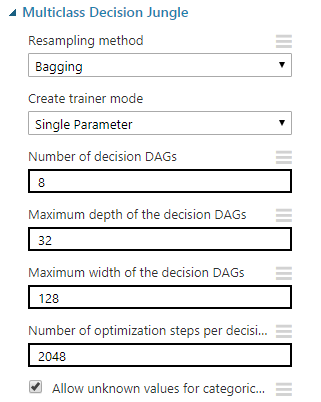
Stablo odluke (*eng. Decision Forest*) jedan je od najpoznatijih algoritama strojnog učenja koji se koristi kod klasifikacije. Stabla odluke radi na principu stabla kao strukture podataka. U takvim strukturama nalazimo čvorove, grane i listove. Čvorovi predstavljaju test na svakom atributu (npr. hoće li vrijednost atributa biti veća ili manja od neke vrijednosti), grane predstavljaju ishod testa (lijeva grana označava istinitost, a desna neistinitost, to jest u lijevu granu spadaju oni čvorovi za koje je promatrana tvrdnja točna, npr. ako ispitujemo je li voće crveno na jabukama i bananama lijevo podstablo će činiti jabuke, a desno banane), a listovi, posljednji čvorovi stabla koji nemaju nasljednike predstavljaju rezultirajuće klase koje smo dobili u jednom stablu.



**Slika 3.3.1** Prikaz svojstava korištenih kod kreiranja modela pomoću algoritma stabla odluke.

## 3.3.2 Multiclass Decision Jungle

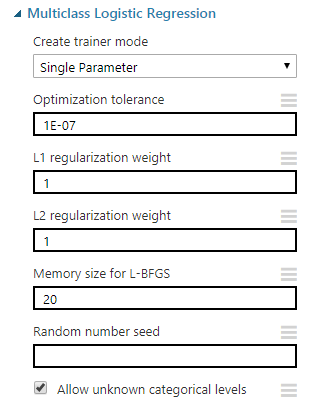
Džungla odluke (*eng. Decision Jungle*) algoritam jako je sličan stablu odluke, ali tijekom rada zauzima manje memorije i duže traje. Nadalje, omogućuju spajanje grana kod donošenja odluke što im daje bolju generalizacijsku sposobnost.



**Slika 3.3.2** Prikaz svojstava korištenih kod kreiranja modela pomoću algoritma džungle odluke.

## 3.3.3 Multiclass Logistic Regression

Logistička regresija (*eng. Logistic Regression*) je metoda u statistici koja se koristi za predviđanje rezultata, a posebno je pogodna za klasifikaciju. Algoritam predviđa vjerojatnost pojavljivanja događaja tako što pomoću trening podataka radi logističku funkciju. Prilikom korištenja ovog algoritma koristila se Single parameter metoda koja govori modelu da mi želimo odrediti parametre na temelju kojih radimo klasifikaciju. Svi ostali parametri ostali su predefinirani.



**Slika 3.3.3** Prikaz svojstava korištenih kod kreiranja modela pomoću algoritma logističke regresije.

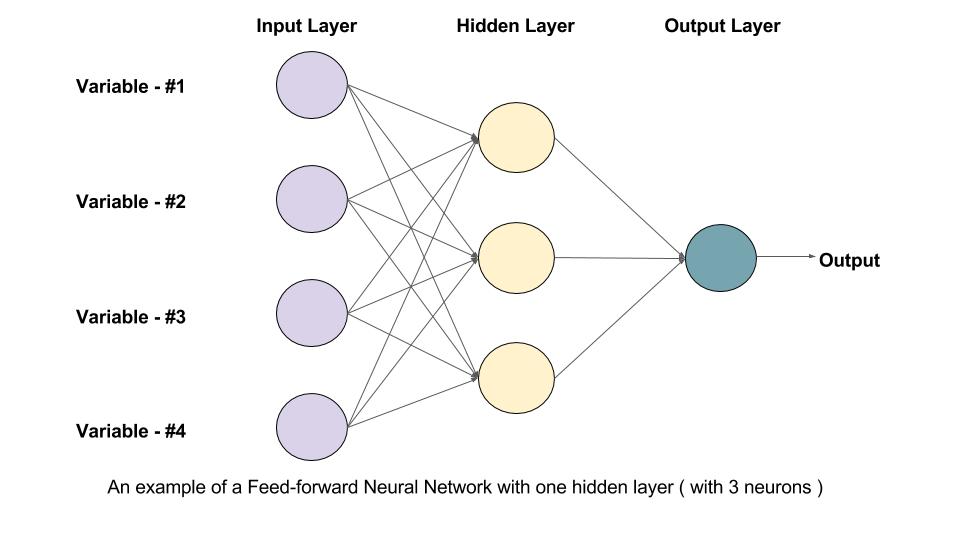
## 3.3.4 Multiclass Neural Network

Algoritam umjetnih neuronskih mreža (*eng. Artificial Neural Network*) glavni je model strojnog učenja za rješavanje složenih problema. Umjetna neuronska mreža sastoji se od osnovnih gradivnih jedinica neurona. Većina umjetnih neuronskih mreža podijeljene su na više slojeva, a najjednostavnija se sastoji od ulaznog i izlaznog sloja, te skrivenog sloja, gdje svaki sloj ima nekoliko neurona.

**Osnovni tipovi neuronskih mreža :**

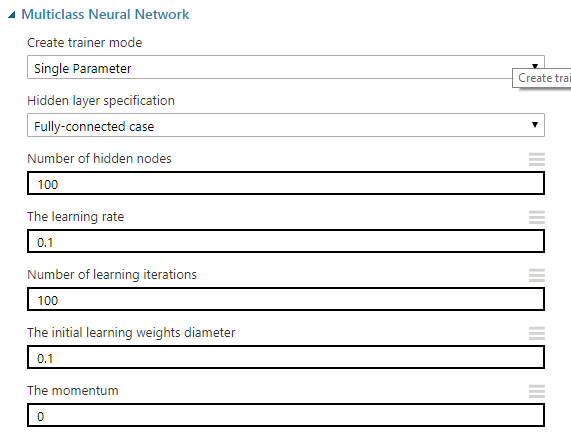
1. **Aciklička mreža** – struktura mreže opisuje se u obliku n1 x n2 x ... x nn, koja označava mrežu od n slojeva kod koje n1 neurona čini ulazni sloj, n2 prvi skriveni sloj, a nn izlazni sloj, nema povratnu vezu
2. **Mreža s povratnom vezom** – mreža koja u svojoj strukturi sadrži barem jednu povratnu vezu, koja omogućuje ponovno obilaženje čvora nakon konačnog broja koraka
3. **Lateralno povezana mreža**
4. **Hibridna mreža**

Neuronske mreže su najrazvijeniji i najsloženiji algoritam strojnog učenja i koriste se u svim područjima gdje se koristi strojno učenje : prepoznavanje govora u zvuku, osoba sa slike, predviđanje vrijednosti, raspoznavanju uzoraka, medicinskoj dijagnostici itd.



**Slika 3.3.4** Prikaz neuronske mreže s 3 sloja (ulazni, jedan skriveni i izlazni). Ulazi predstavljaju varijable na temelju kojih se radi predikcija. Linije koje povezuju slojeve nazivaju se težinama neuronske mreže.

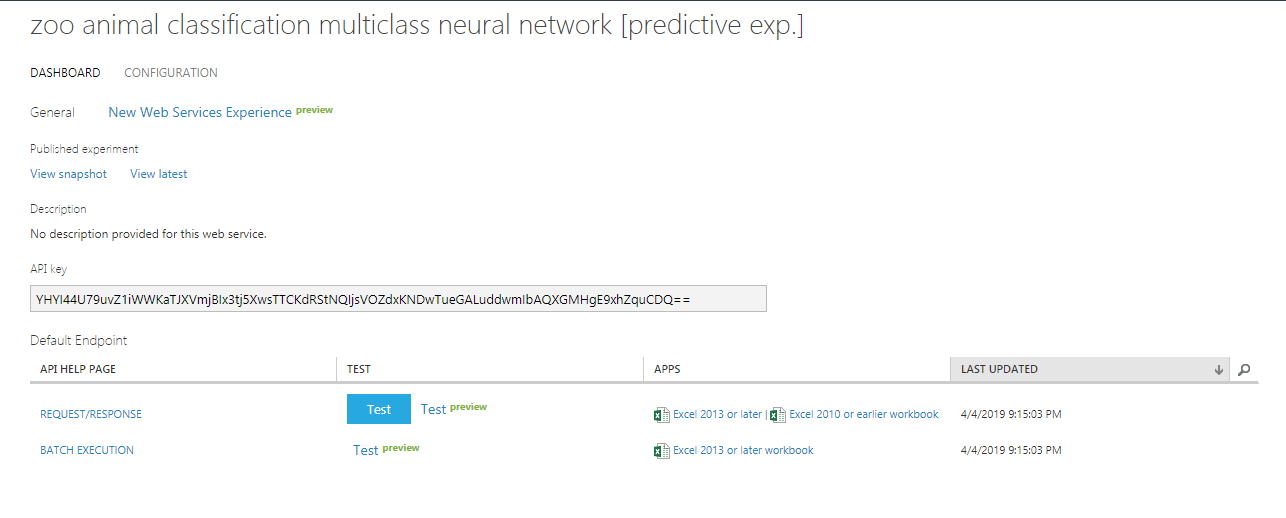
Zbog navedenih svojstava neuronske mreže, iz neuronske mreže će se razviti predikcijski eksperiment koji će se postaviti kao web API (*eng. Application Programming Interface*) čija će se funkcionalnost koristiti na korisničkom sučelju. Kod procesa treniranja modela, najvažniji parametar neuronske mreže je stopa učenja (*eng. learning rate*) koji upravlja promjenom modela na procijenjenu grešku svaki put kad su vrijednosti težina neuronske mreže promijenjene. Odabir idealne stope učenja može biti zahtjevno – premala vrijednost rezultira dugim ili prekinutim procesom treniranja, a prevelika vrijednost neoptimalnim skupom vrijednosti grana (preveliki koraci optimizacije).



**Slika 3.3.4.1** Korišeni parametri kod kreiranja modela neuronske mreže.Osim navedeih parametra koristila se i Min-Max normalizacija parametara (skaliranje vrijednosti parametara na raspon[0,1] što i nije bilo potrebno osim za značajku broj nogu).

## 3.4 Rezultati klasifikacije

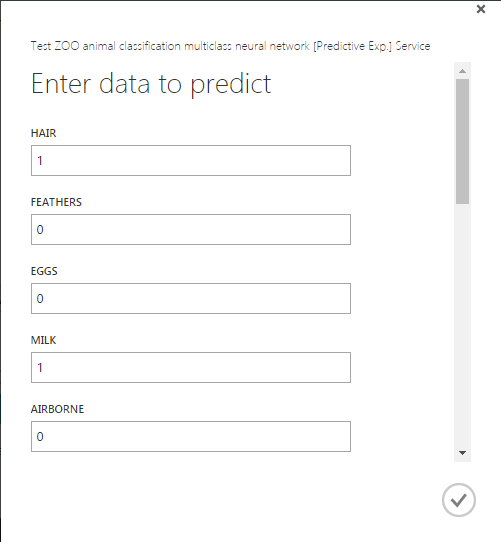
Nakon kreiranja predikcijskog eksperimenta, iz njega je bilo potrebno razviti web servis (web API) pomoću kojeg ćemo testirati rad modela. Osim testiranja, pomoću njega ćemo biti u stanju pozivati model u bilo kojem aplikacijskom sučelju. Razvijeni web servis ima svoj jedinstveni API ključ preko kojeg možemo pristupati njegovoj funkcionalnosti iz različitih aplikacijskih okruženja (Python, Angular, R, Visual Studio itd.). Korištenjem posebnih funkcija koje propisuje korišteno okruženje, možemo unijeti podatke (odrediti ulaz) preko POST metode, dobiti rezultate kreiranog programskog modela pomoću GET metode, obrisati podatke poslane na servis preko DELETE metode itd.



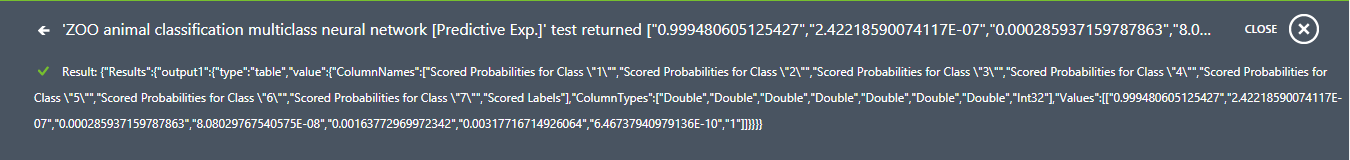
**Slika 3.4** Prikaz sučelja razvijenog Web API-a u MS Azureu koji će biti programsko sučelje korisničke aplikacije i treniranog modela koji radi klasifikaciju.

## 3.4.1 Testiranje funkcionalnosti modela

Testiranje i provjera rada modela odvija se na sučelju razvijenog API-a. U svrhu testiranja, bit će unesena instanca čovjeka s njegovim prikladnim vrijednostima traženih značajki. Znamo da je čovjek sisavac (izlazna veličina koju bi trebali dobiti) i da kao instanca nije bio dio skupa za treniranje i testiranje.



**Slika 3.4.1** Testiranje razvijenog predikcijskog modela unutar MS Azure okruženja.

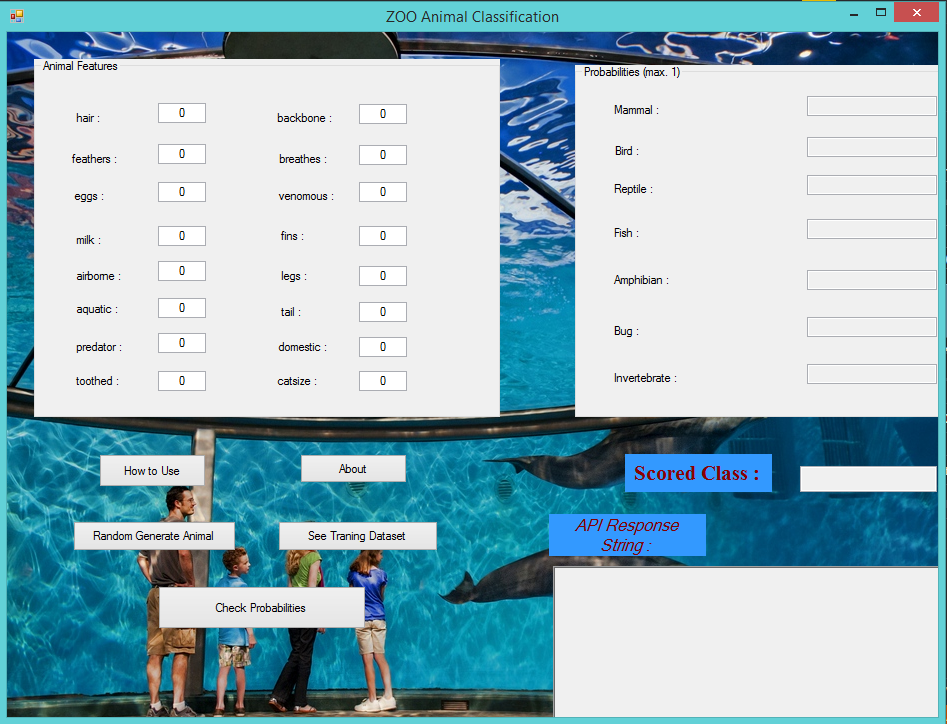


**Slika 3.4.2** Prikaz odgovora razvijenog API-a na unesenu instancu čovjek. Iako je dobiveni niz znakova dosta nepregledan, model je ispravno i jako precizno procjenio vjerojatnost pripadnosti čovjeka klasi sisavac (99.94%).

# 4. Razvoj korisničkog programskog sučelja

## Opis programskog rješenja

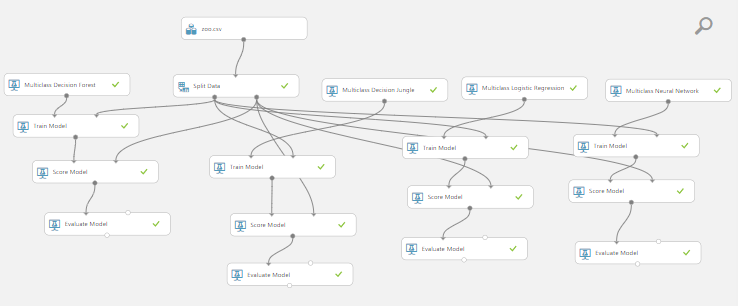
Aplikacija omogućuje unos značajki životinje kojoj korisnik želi provjeriti pripadnost. Na temelju tih podataka aplikacija šalje zahtjev (ulazne vrijednosti) na razvijeni Web API opisan u [3.4](#_3.4_Rezultati_klasifikacije). Zatim, Web API vraća odgovor kao niz znakova iz kojeg je potrebno izvući bitne dobivene podatke (procijenjenu klasu i vjerojatnosti pripadnosti svim mogućim klasama). Korisničko sučelje napravljeno je u Visual Studio okruženju, točnije u Windows Forms aplikaciji napisanoj u programskom jeziku C#. Funkcionalnost aplikacije i njezinih stavki bit će opisana u nastavku.



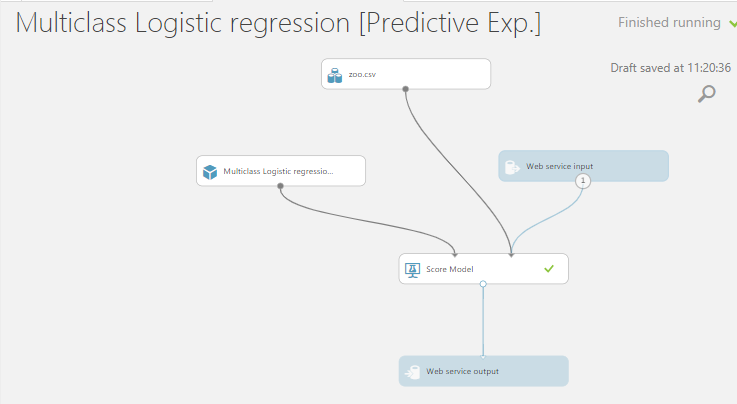
**Slika 4.1** Prikaz izgleda aplikacije prilikom njezina pokretanja.

## Konačni izgled eksperimenta

Prilikom izrade aplikacije isprobalo se 4 modela strojnog učenja kako bi se riješio dani problem. Pošto je skup podataka dosta jednostavan (mal raspon vrijednosti većine ulaznih podataka) i ne zahtjeva veliku odradbenu moć, nije bilo toliko bitno koji će se model izabrati za klasifikaciju životinja. Donešena je odluka da će se u korisničkoj aplikaciji koristiti model umjetne neuronske mreže, za koje je poznato da je najprecizniji postojeći model strojnog učenja. Detaljni postupci rađenja eksperimenta opisani su u poglavlju [3.3](#_3.3_Korišteni_postupci).



**Slika 4.2.1** Prikaz konačnog izgleda eksperimenta u Microsoft Azure-u.



**Slika 4.2.2** Prikaz predikcijskog eksperimenta koji se koristio za razvijanje Web API-a.

## Korištenje API-a

Za pozivanje kreiranog razvijenog API-a u programskom jeziku C# potrebno je odrediti njegov URI (*eng. Uniform Resource Identifier*), tip zahtjeva koji se šalje (POST), te dodatne opcije među kojima su jedinstveni API ključ i format podataka preko kojeg se podaci šalju API-u (najčešće JSON format) kao i sami ulazni podaci. Ulazni podaci šalju se preko funkcije AddParameter koja prima niz znakova u posebnom formatu koji je preuzet iz uputa o korištenju API-a na MS Azureu. Kako bi se mogli slati zahtjevi na API, potrebno je koristiti RestSharp C# biblioteku. Za spremanje odgovora API-a koji predstavlja izlazne podatke i rezultat klasifikacije razvijenog modela koristi se posebna varijabla tipa IRestResponse koja će predstavljati odgovor API-a na određeni zahtjev koji se poslao serveru. Da bi se dobivene izlazne vrijednosti iskoristile, spremile u objekt instance životinja, te naposljetku prikazale korisniku, potrebno je koristiti deserijalizaciju JSON niza znakova. Proces deserijalizacije je dodjeljivanje vrijednosti objektu iz odreženog JSON niza znakova, pri čemu imena atributa objekta (instanciranog iz klase) moraju biti jednaka imenima vrijednosti u JSON formatu (koji je objektni model). Zbog toga su se dijelovi niza znakova API-evog odgovora morali zamijeniti imenima atributa (varijabli) objekta kako bi spremanje vrijednosti u objekt bilo moguće. Deserijalizacijom smo dobili vrijednosti objekta životinja iz kojeg jednostavno možemo izvući dobivene vrijednosti i prikazati ih korisniku. Izlazna vrijednost koja predstavlja klasu unesene životinje je broj od 1 do 7, a u korisničkom sučelju su se te vrijednosti pridodale odgovarajućem imenu klase (sisavac za 1, 2 ptica za 2 itd.). U aplikaciji je ograničen unos vrijednosti koje nisu u domeni značajki životinja (sve vrijednosti mogu sadržavati samo 0 i 1, a broj nogu 0, 2, 4 i 8), te se korisnik upozorava prilikom krivog unosa. Korisniku je omogućeno generiranje nasumične (izmišljene) životinje kako bi se dodatno provjerila funkcionalnost treniranog modela. Nadalje, korisniku je omogućen pregled skupa za učenje, kako bi mogao unijeti životinju iz zoološkog vrta i provjeriti radi li napravljeni model. Korisničko sučelje također nudi opis kako se aplikacija koristi, te podatke o razvijenoj aplikaciji.

1. **public** **class** Animal
2. {
3. **private** **int** hair, feathers, eggs, milk, airborne, aquatic, predator,
4. toothed, backbone, breathes, venomous, fins, tail, domestic, catsize;
5. **private** **int** legs;
6. **public** **string** probabilityMammal;
7. **public** **string** probabilityBird;
8. **public** **string** probabilityReptile;
9. **public** **string** probabilityFish;
10. **public** **string** probabilityAmphibian;
11. **public** **string** probabilityBug;
12. **public** **string** probabilityInvertebrate;
13. **public** **string** score;
14. ...
15. }

**Kod 4.3.1** C# klasa koja se koristi za instanciranje objekta životinja. Kao atribute sadržava ulazne i izlazne vrijednosti modela strojnog učenja.

1. var client = **new** RestClient("https://ussouthcentral.services.azureml.net/workspaces/c8f8d54293054df997070fac57ee9366/services/da2db6f9f6d947458210709e2da232bb/execute?api-version=2.0&format=swagger"); //poveznica servisa na koji šaljemo zahtjev
2. var request = **new** RestRequest(Method.POST);
3. request.AddHeader("Postman-Token", "e8414b95-b5c7-4d92-9fa6-16c0c942414d");
4. request.AddHeader("cache-control", "no-cache");
5. request.AddHeader("Authorization", "Bearer YHYI44U79uvZ1iWWKaTJXVmjBIx3tj5XwsTTCKdRStNQIjsVOZdxKNDwTueGALuddwmIbAQXGMHgE9xhZquCDQ==");//API ključ
6. request.AddHeader("Content-Type", "application/json");
7. request.AddParameter("undefined", "{\r\n\"Inputs\": {\r\n\"input1\":\r\n[\r\n{\r\n'hair': \""
8. + animal.GetHair() + "\",   \r\n'feathers': \""
9. + animal.GetFeathers() + "\",\r\n'eggs': \""
10. + animal.GetEggs() + "\",\r\n'milk': \""
11. + animal.GetMilk() + "\",\r\n'airborne': \""
12. + animal.GetAirborne() + "\",\r\n'aquatic': \""
13. + animal.GetAquatic() + "\",\r\n'predator': \""
14. + animal.GetPredator() + "\",\r\n'toothed': \""
15. + animal.GetToothed() + "\",\r\n'backbone': \""
16. + animal.GetBackbone() + "\",\r\n'breathes': \""
17. + animal.GetBreathes() + "\",\r\n'venomous': \""
18. + animal.GetVenomous() + "\",\r\n'fins': \""
19. + animal.GetFins() + "\",\r\n'legs': \""
20. + animal.GetLegs() + "\",\r\n'tail': \""
21. + animal.GetTail() + "\",\r\n'domestic': \""
22. + animal.GetDomestic() + "\",\r\n'catsize': \""
23. + animal.GetCatsize() + "\",\r\n}\r\n],\r\n},\r\n\"GlobalParameters\":  {\r\n}\r\n}\r\n", ParameterType.RequestBody); //ulazne vrijednosti
24. IRestResponse response = client.Execute(request); //odgovor na ulazne vrijednosti
25. **string** ResponseContentString = response.Content.ToString();
26. **string** json = CreateJson(ResponseContentString);
27. Animal animaldeserialized = JsonConvert.DeserializeObject<Animal>(json);  //deserijalizacija niza znakova
28. richTextBox2.Text = json;
29. Display(animaldeserialized);

**Kod 4.3.2** Dio koda u kojem se šalje zahtjev i prima traženi odgovor sa servera.

# Zaključak

U ovom projektu susrelo se sa strojnim učenjem, njegovim podvrstama i jednom od njegovih primjena. Algoritmi strojnog učenja uvelike su pomogli čovječanstvu u rješavanju složenih problema, ponajviše zbog svoje preciznosti i mogućnosti samozaključivanja na temelju prethodnih grešaka i podataka. Oni se koriste u visokom stupnju osiguranja, predviđanju katastrofa, te u svim mogućim granama znanosti. Obrađeni problem klasifikacije životinja u zoološkom vrtu uvelike bi pomogao službenicima zoološkog vrta u odjeljivanju životinja u grupe po njihovim klasama, pri čemu bi se odvojeno mogli posvetiti potrebama životinja svake klase. Klasifikacija životinja ključna je u ljudskom razumijevanju živog svijeta, a prisutna je još od 18. stoljeća. Napravljeni klasifikacijski model može pomoći u klasifikaciji novootkrivenih životinja s kojima se čovječanstvo još nije susrelo.

# Literatura

[1] Neuronske mreže, način rada i parametri : <https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/>.

[2] Nadzirano i nenadzirano učenje : <https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/>.

[3] Podržano učenje : <https://skymind.ai/wiki/deep-reinforcement-learning>.

[4] Načini rada i parametri algoritama strojnog učenja : Šuma odluke, Džungla odluke, Logistička regresija, Neuronska mreža <https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/machine-learning-initialize-model-classification>.

[5] Skup za učenje preuzet sa : <https://www.kaggle.com/uciml/zoo-animal-classification>.

# Prilozi

[1] Programsko rješenje (branch Develop) : <https://github.com/fcuic/Zoo-Animal-Classification/tree/Develop> .