Titanic Survival Classification

Dragan Janjanin, RA86/2022 SAuSAU, Septembar 2025.

# Uvod i opis problema

Cilj projektong zadatka je rešavanje klasifikacionog problema preživljavanja putnika Titanika. Zahtevi za izradu projekta su uspešno rukovanje datim podacima iz skupa za treniranje, njihova obrada i manipulacija radi treniranja modela mašinskog učenja, izbor najboljeg modela i njegovih parametara, analiza samih rezultata algoritma. Za kraj, na skupu podataka za testiranje potrebno je primeniti isti model sa parametrima i zapisati predviđanja.

U podacima nam je za svakog putnika dat podskup identifikujućih podataka:

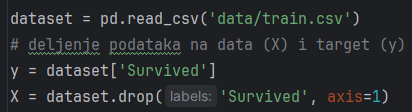
* **PassengerId** — jedinstveni identifikator svakog putnika, korišćen za praćenje.
* **Survived** — **ciljana promenljiva** označena vrednostima 0 (nije preživeo) i 1 (preživeo).
* **Pclass** — klasa karte (1 = prva, 2 = druga, 3 = treća), funkcioniše kao pokazatelj socio-ekonomske pozicije.
* **Name** — puno ime putnika, iz kojeg se može izvesti titula (npr. “Mr.”, “Mrs.”).
* **Sex** — pol putnika (male / female), značajan demografski faktor.
* **Age** — starost u godinama; ponekad izražena kao decimalna vrednost (npr. 0.5 za bebe).
* **SibSp** — broj braće/supružnika koji su bili na brodu (uz striktno definisane relacije).
* **Parch** — broj roditelja/dece na brodu (uz definisane kategorije kao što su majka, otac, dete).
* **Ticket** — broj karte, koji može sadržati informacije o tipu ili grupi putnika.
* **Fare** — cena karte, izražena u britanskim funtama, ukazuje na platežnu moć i položaj.
* **Cabin** — broj kabine, često nepotpun i korišćen kao potencijalni indikator dodate vrednosti (npr. blizina palube).
* **Embarked** — luka ukrcavanja (C = Cherbourg, Q = Queenstown, S = Southampton), može ukazivati na geografske i društvene obrasce.

# Postupak izrade

## Priprema podataka

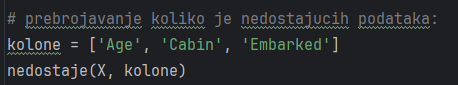
Potrebno je odabrati relevantne podatke iz zadatog skupa.

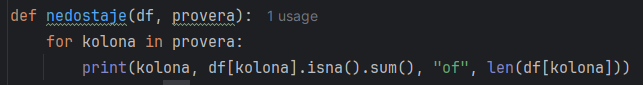
Prvenstveno, učitavam podatke iz date train.csv datoteke uz pomoć „pandas“ biblioteke.



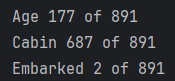
Učitan dataframe dalje delim na X i y. X predstavlja identifikujuće podatke, a y cilj. Odnosno na ovaj način postižem „X, y = dataset.data, dataset.target“ koji smo primenjvali na vežbama.

Sledeće potrebno je da prebrojim nedostajuće podatke.





Ovo daje izlaz:



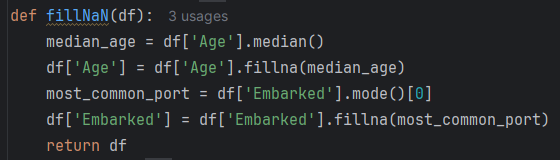
Age je faktor koji ne bi trebalo zanemariti, ipak je izreka „žene i deca prvo“, a odbaciti podatak koji direktno identifikuje mlade osobe bi nam narušilo treniranje modela. Problem je što fali dosta vrednosti u Age koloni i to se mora nekako popuniti.

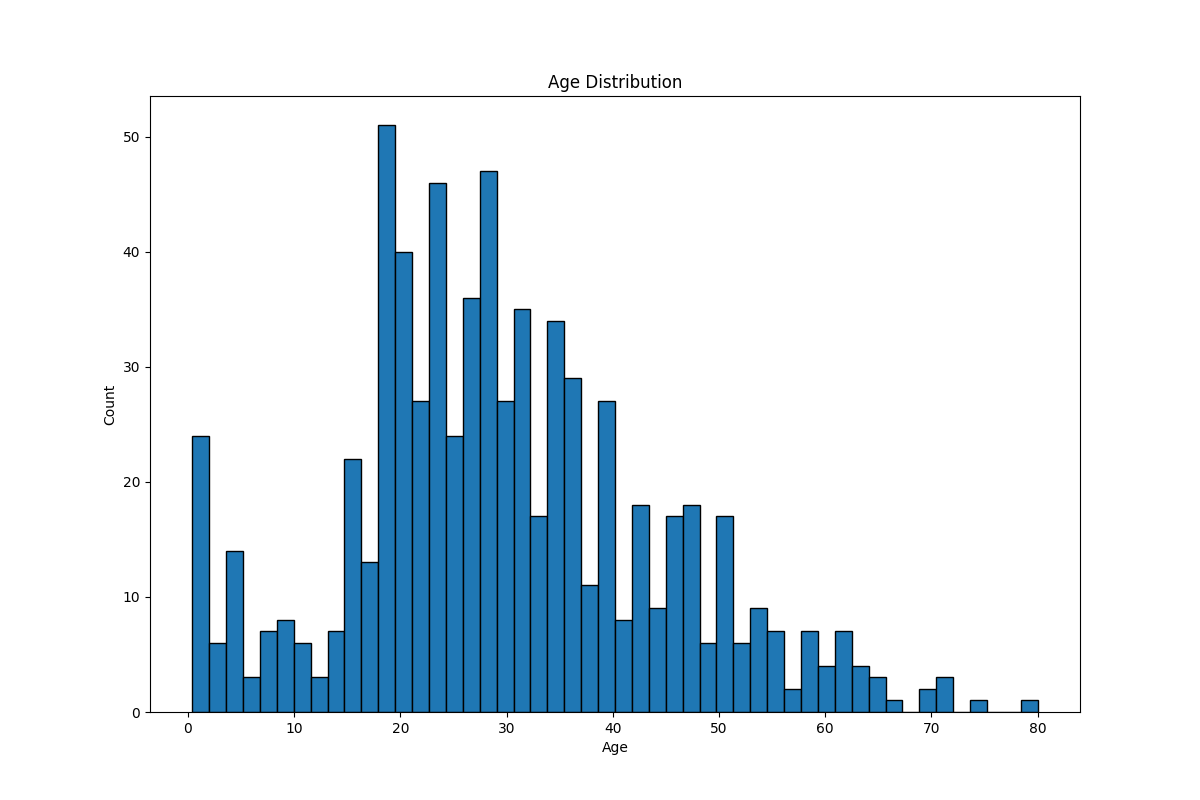
Dok je podatak o kabini potencijalno vrlo koristan, sama činjenica da fali 77,1% podataka iz te kolone čini ovaj deo skupa neiskoristljivim.

Mesta ukrcavanja su skoro skroz potpuna.

Podatke PassengerId, Name, Ticket i Cabin sam odstranio iz dataframe-a. Prva tri ne utiču direktno na preživljavanje, dok je Cabin, kao što sam rekao ranije, neupotrebljiv podatak.

Popunjavanje nedostajućih vrednosti se postiže lako pozivom fillNan fukncije:



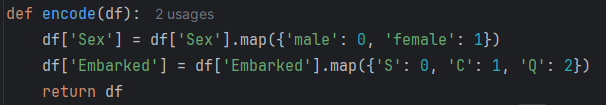
Odluka za popunjavanje Age kolone medijanom proizilazi iz analize raspodele uzrasta putnika.

Očevidno je da postoji neka normalna raspodela uzrasta uz velik broj ekstrema na stranu mlađih uzrasta. Samim tim popunjavanje je najpogodnije uraditi upisivanjem srednje vrednosti ili medijanje Age kolone. Takve vrednosti nam neće kvariti proračune modela, za razliku da smo preko svih NaN upisali 40 ili 10. Odlučio sam da upišem medijanu jer na srednju vrednost baš i utiču prisutni ekstremi. Medijana se dobija kao vrednost srednjeg elementa sortiranog seta sa neparnim brojem elemenata, ili srednja vrednost dva srednja elementa ako set ima paran broj elemenata.

Nedostajuća mesta ukrcavanja sam samo popunio sa najčešćom vrednošću.

Sledeći korak u pripremi podataka je enkodiranje. Potrebno je nenumeričke vrednosti pretvoriti u numeričke kako bi mogli koristiti modelu.

U trenutnom dataframe-u kolone Sex i Embarked sadrže stringove pa:

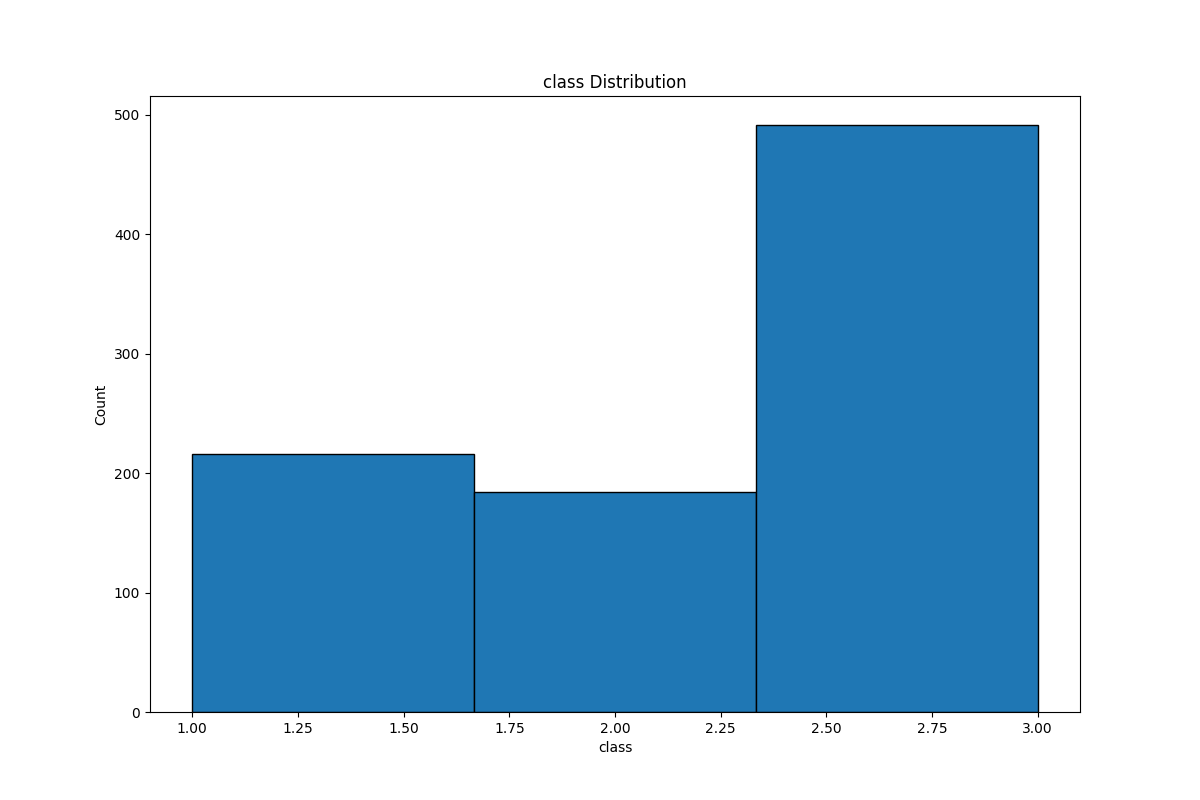


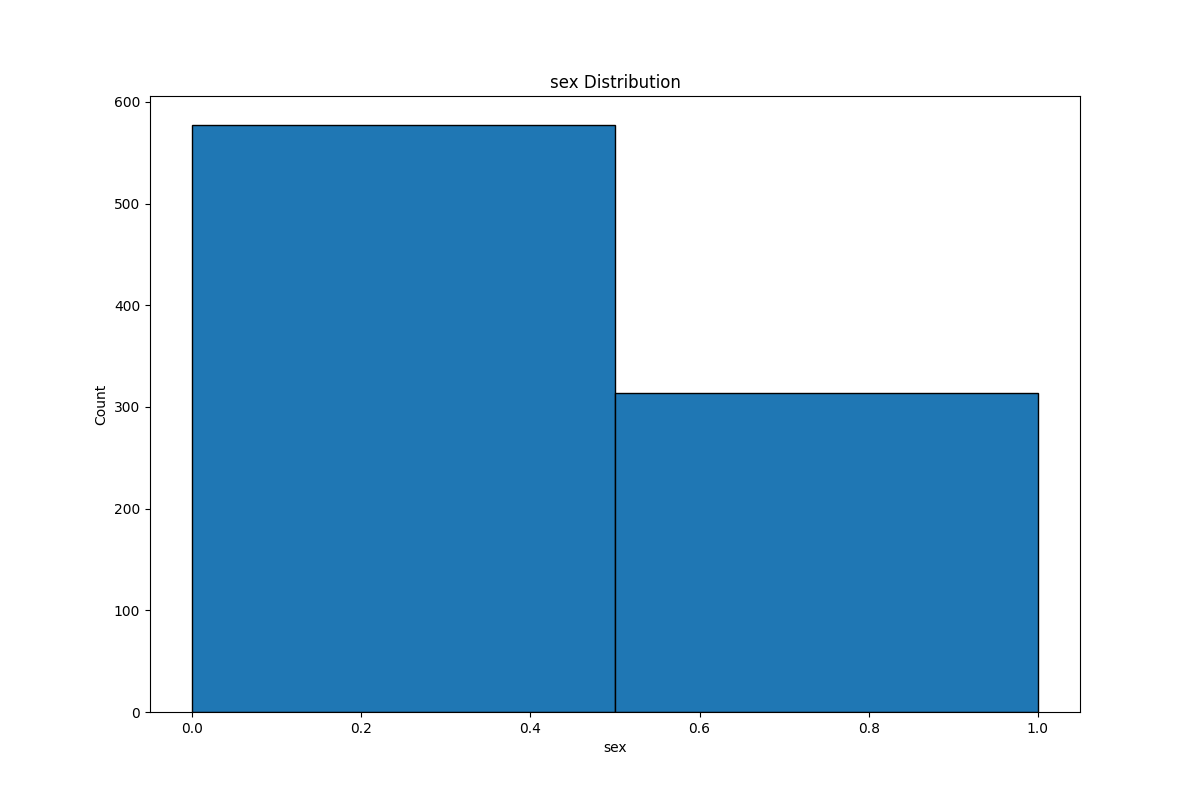
I sada spreman skup X upisujem u edited.csv radi prikaza.

## Vizuelizacije podataka

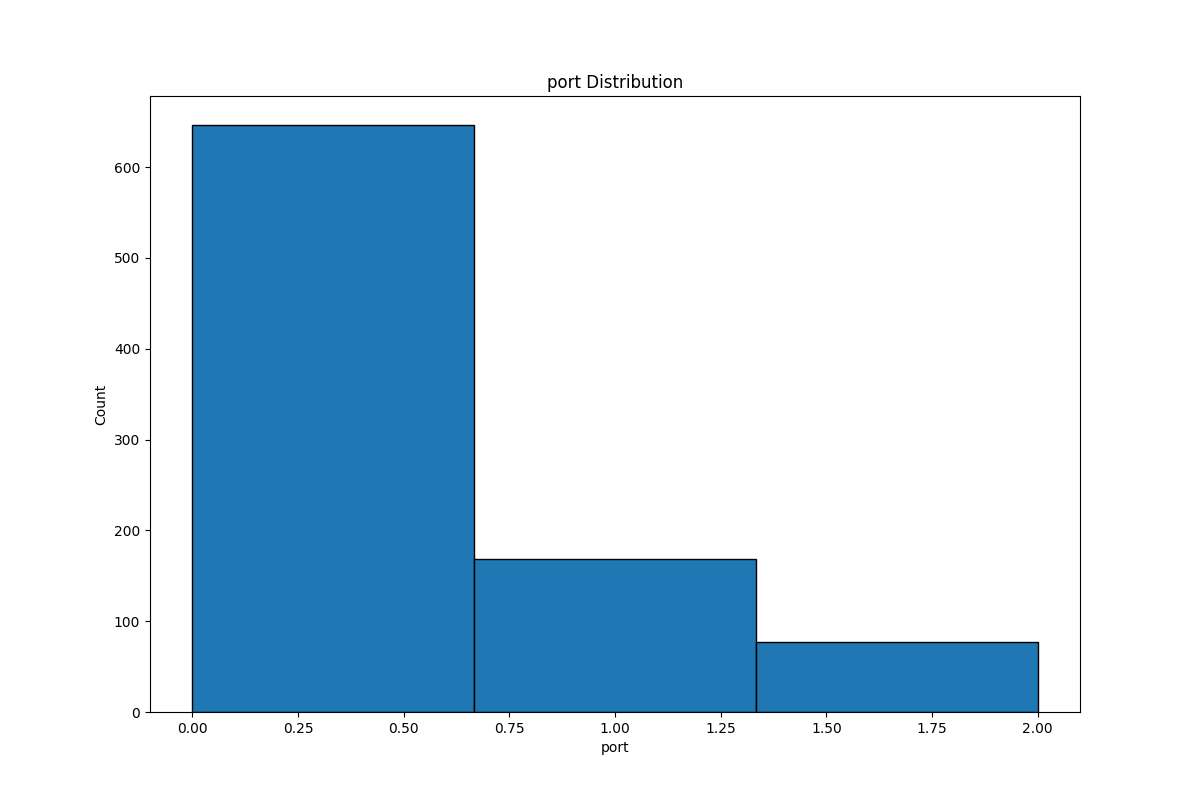
### Distribucije po vrednostima kolona

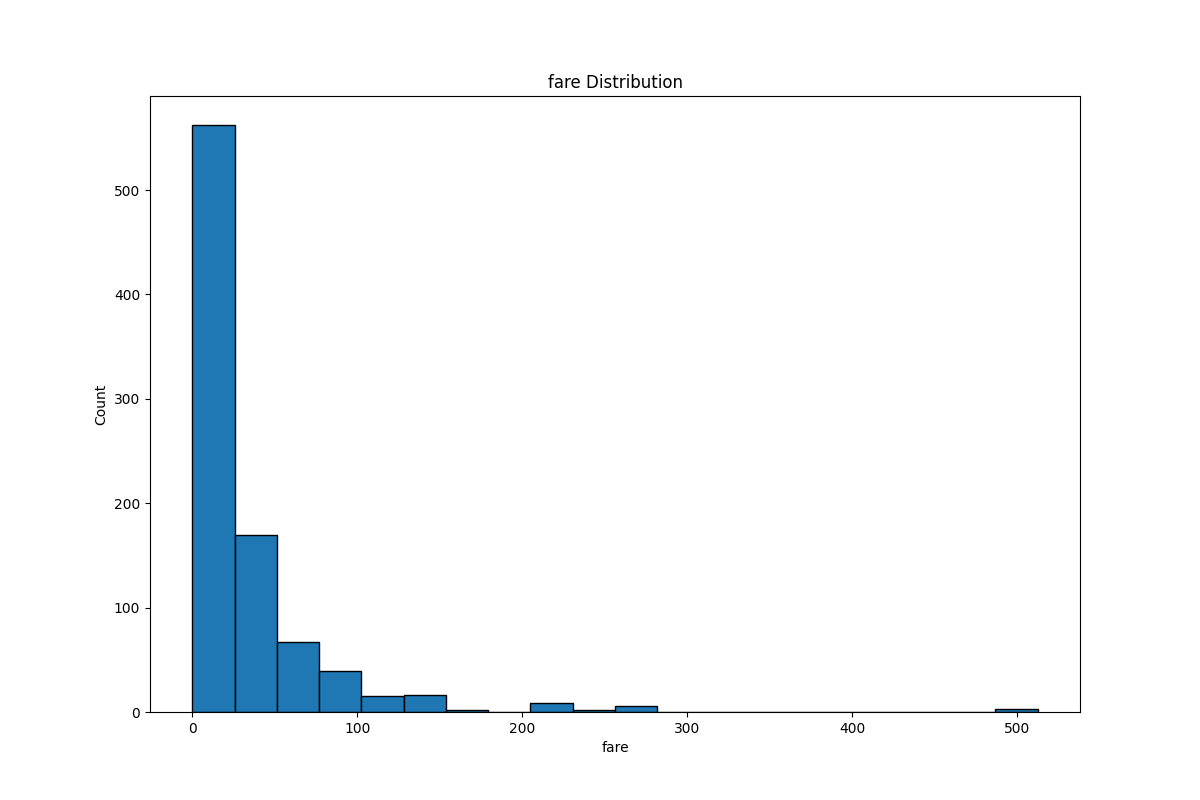
Raspodela putnika po klasama:

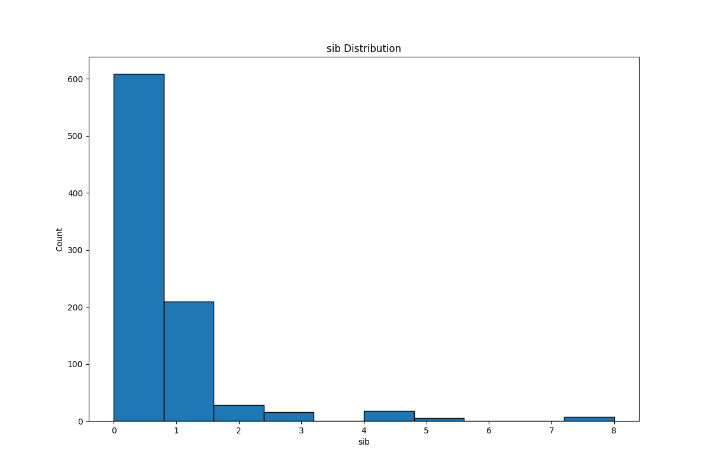
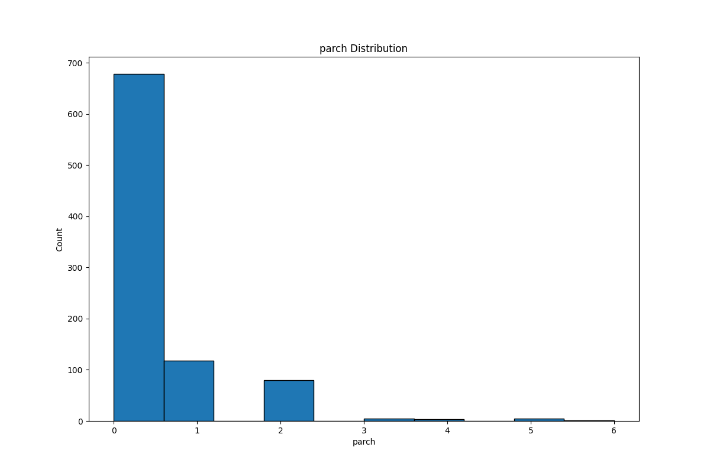


Raspodela putnika po polovima:

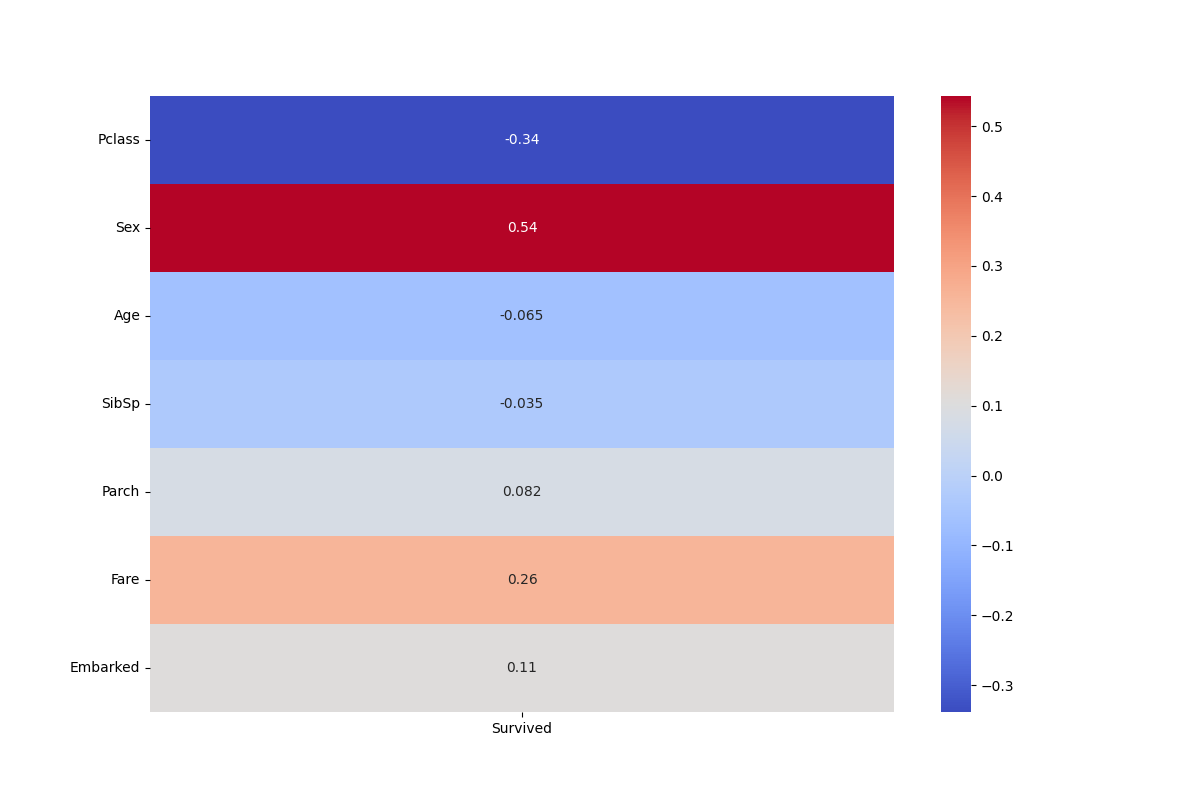
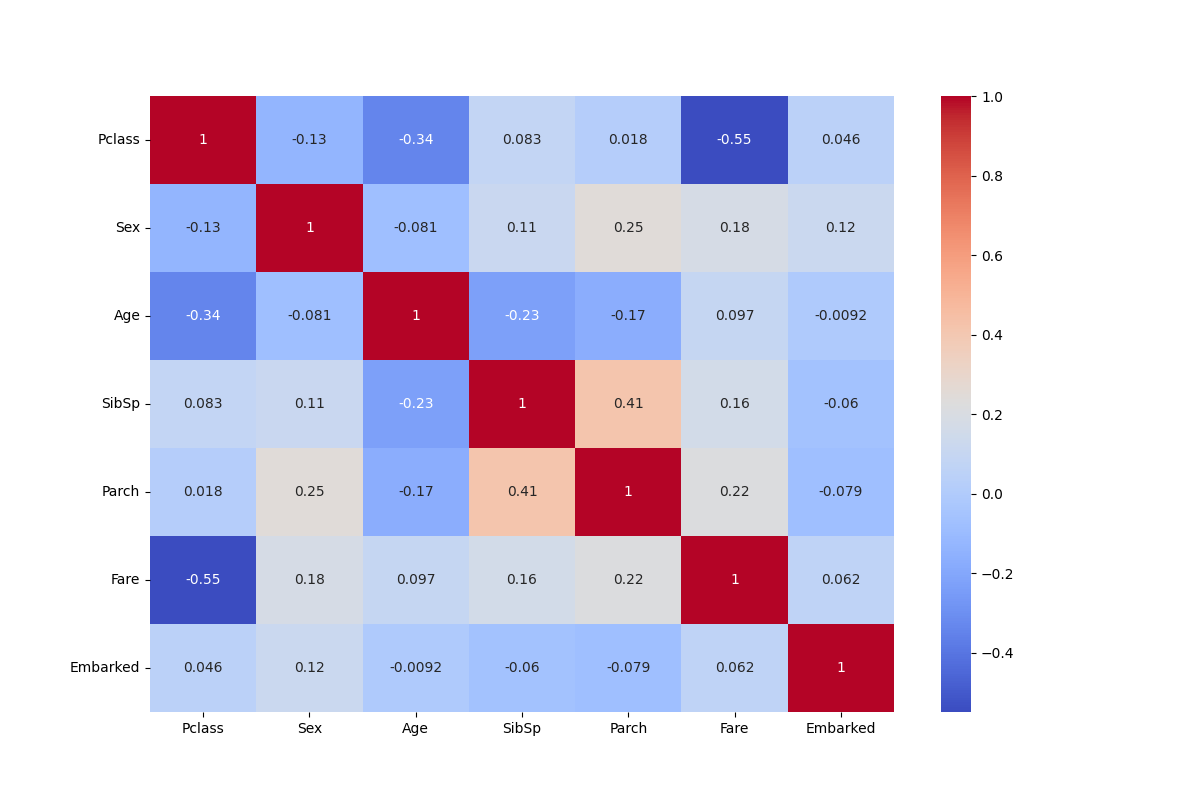
Raspodela putnika po mestu ukrcavanja:



Raspodela putnika po ceni karte:

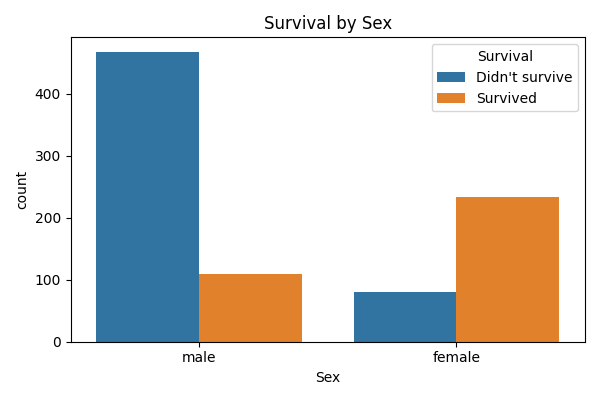
Raspodele putnika po broju prisutnih osoba sa relacijama:

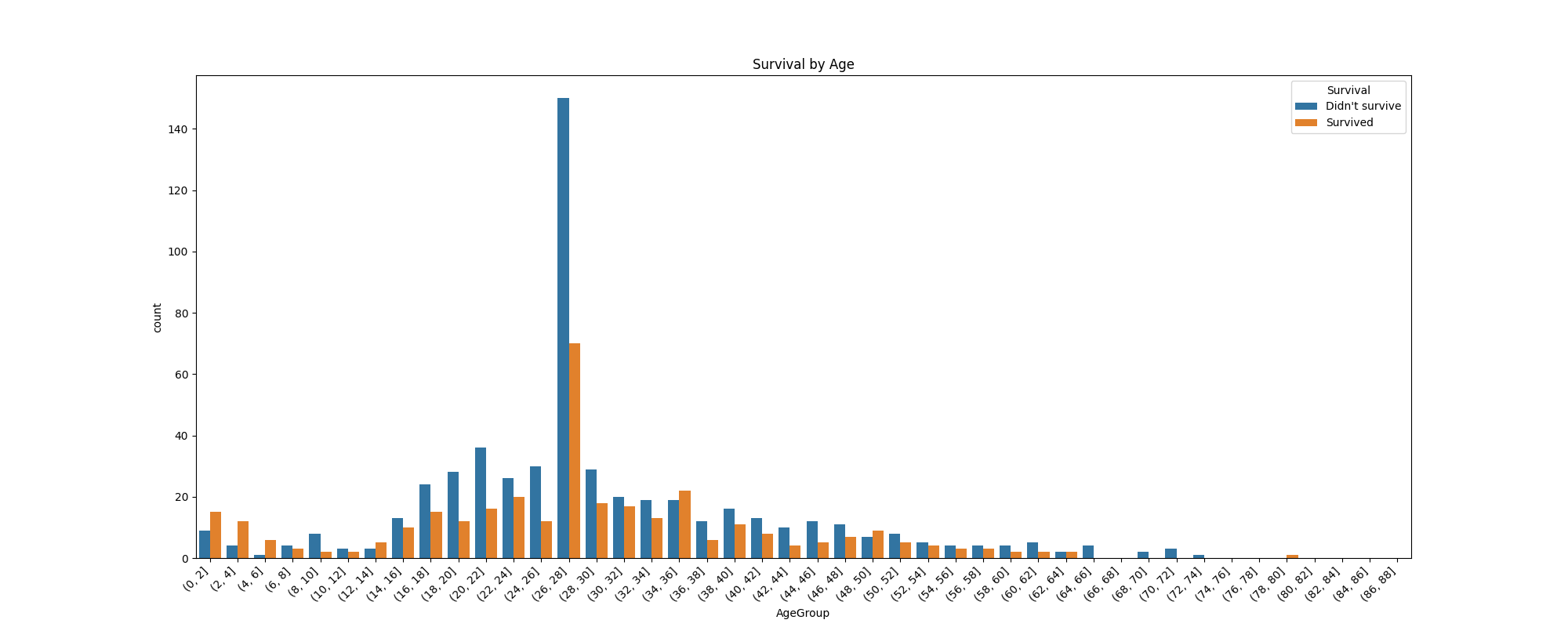
### Korelacija podataka međusobno i sa ciljem



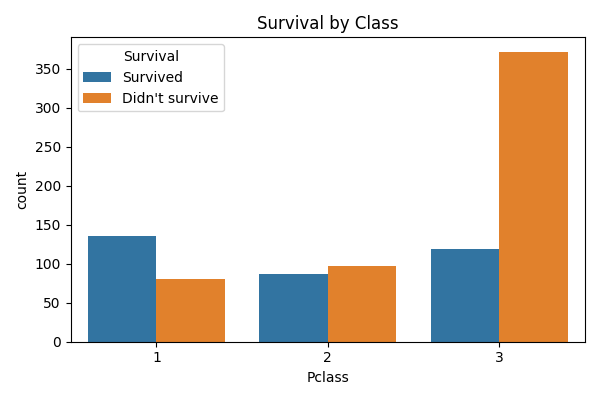
### Odnosi atributa i cilja

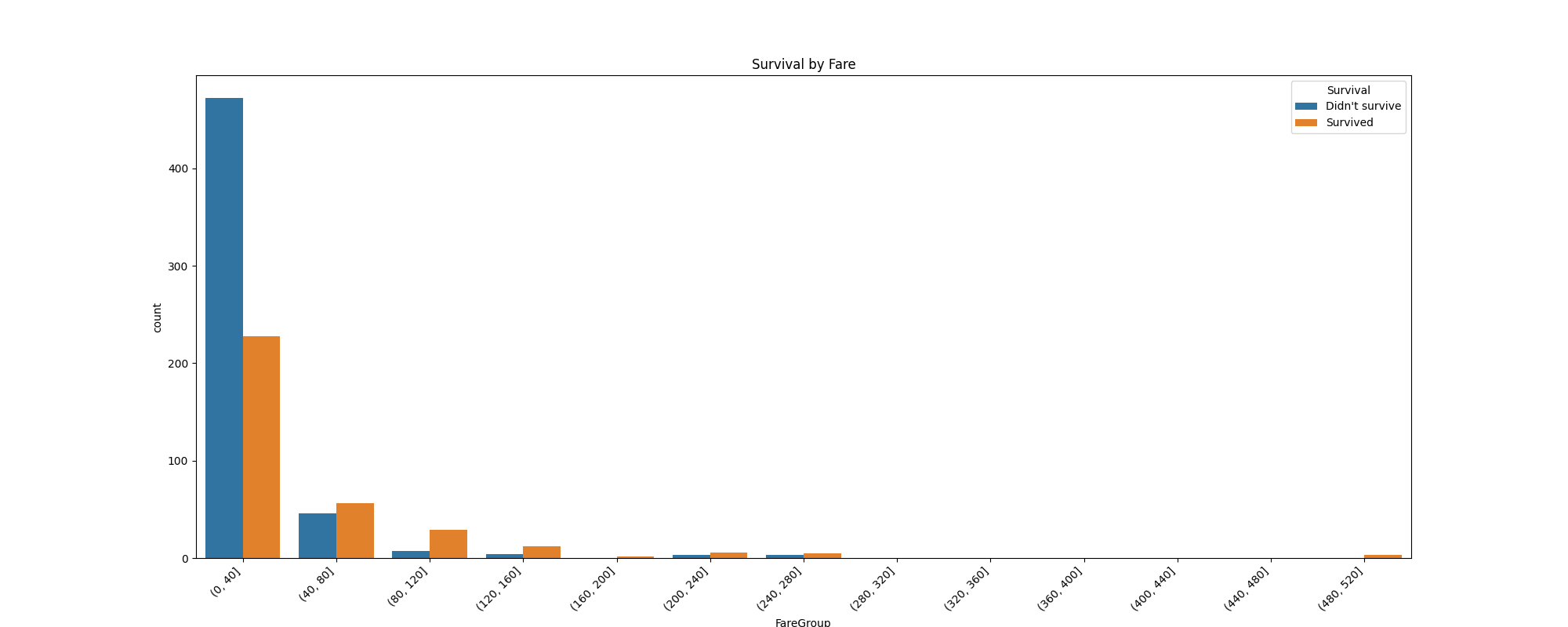
Preživeli po polu:



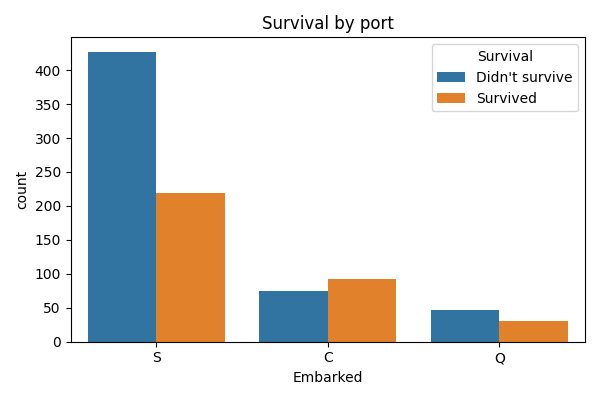
Preživeli po uzrastu:

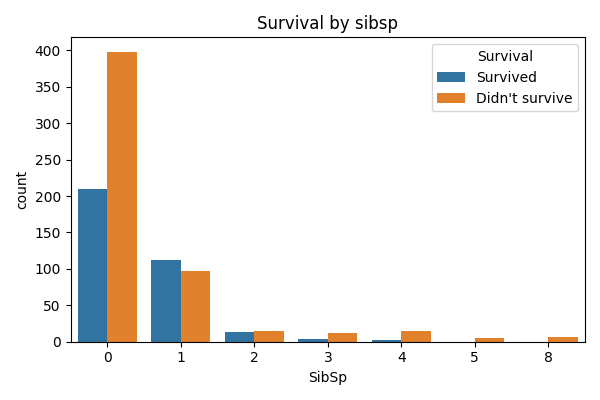
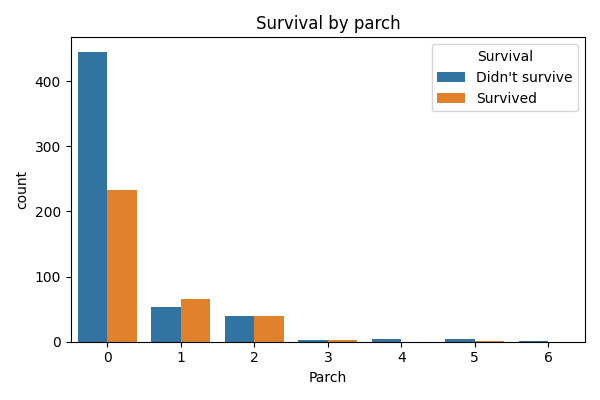
Preživeli po klasi:



Preživeli po ceni karte:

Preživeli po mestu ukrcavanja:



Preživeli po broju relacija:

## Odabir modela i njihovih parametara

Problem koji se rešava je klasifikacioni. Modeli za klasifikaciju su Logistic Regression, Decision Tree Classifier, Random Forest Classifier i KNeighbors Classifier.

Funkcija findBestModelAndTestSize za parametre uzima dict koji sadrži modele, nase podatke podeljenje na X (data) i y (target), niz test\_sizes sa raznim vrednostima za podelu u train\_test\_split. Vraća najbolji model i test\_size za koji najbolje radi.

U funkciji se iterira kroz sva četiri modela i probavaju se za svaki od vrednosti u test\_sizes. Za svaki model i test\_size se pronalaze tačnost, preciznost, odziv i f1 skor. Nadalje iz tih vrednosti pronalazi weighted\_score koji daje okvirni pregled sposobnosti modela. Koristi takvu metriku kako bi se dalo više vrednosti odzivu i f1 skoru, bas zato što su nam bitniji za ovakav problem. Tačnost može davati lažnu sliku sposobnosti modela, ova ocena je korisna kada su nam podaci balansirani, a za ovaj problem nisu (više ljudi je preminulo nego preživelo). Odziv govori koliko je tačno pogođenih vrednosti i koristan je kada treba odrediti što više preživelih, a f1 balansira odziv i preciznost, pa je jedna od korisnijih ocena za ovaj model.

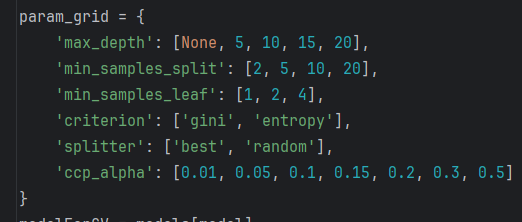
Funkcija vraća Decision Tree sa test\_size=0.2 kao najbolji model.

### Unakrsna validacija i hiperparametrizacija

Sa poznatim modelom sledeći korak je podesiti parametre kako bi odredili realnije vrednosti traženih procena. Unakrsnom validacijom postižemo odabir hiperparametara kao i evaluaciju generalizacije modela na neviđenim podacima. Podatke delimo na cv=10 delova i svake iteracije koristimo različit deo kao test set. Funkcija uzima X, y, Decision Tree (model), test\_size koji je unapred određen i parametre za hiperparametrizaciju.



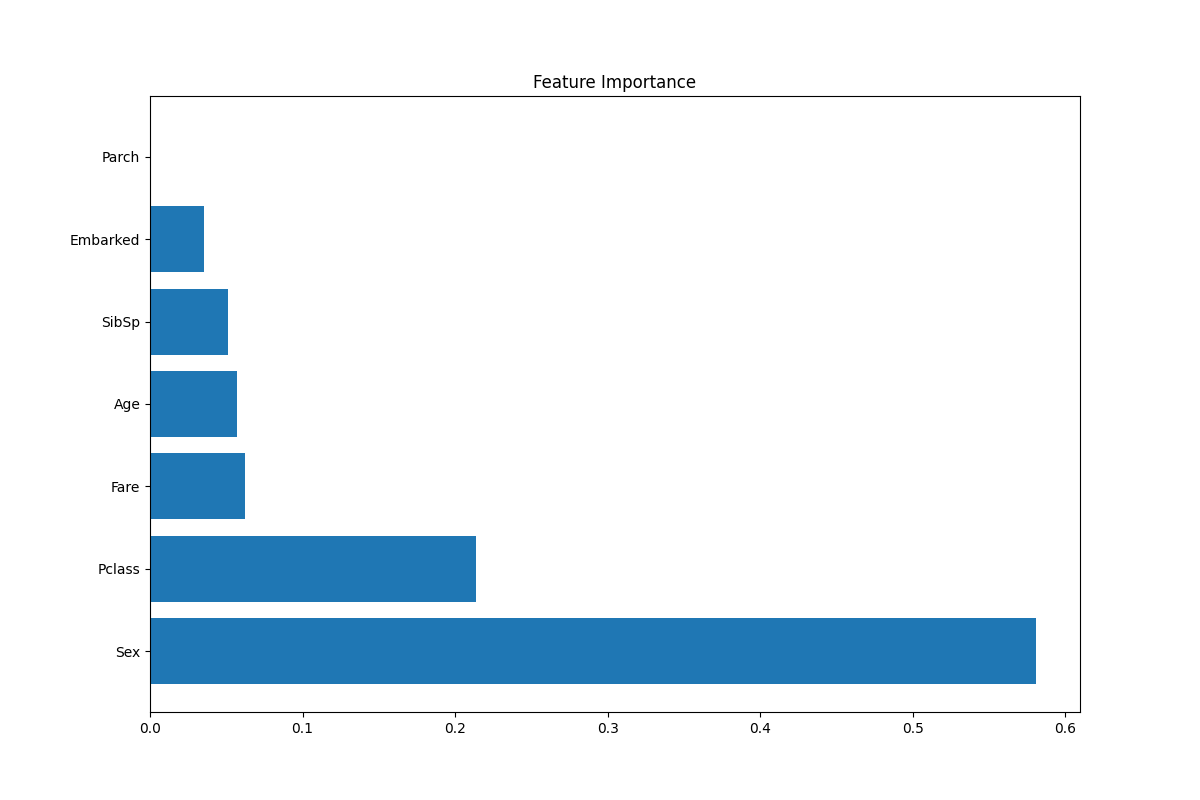
Za skup parametara:



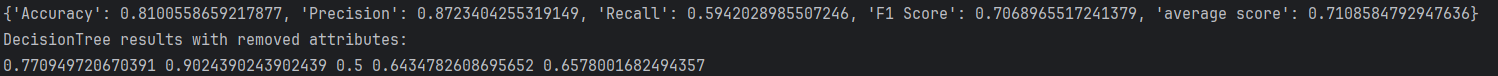
Ove vrednosti sam odabrao otprilike, uključuju i defultne vrednosti za te parametre.

Vraćena vrednost je Decision Tree model sa svim odabranim hiperparametrima: DecisionTreeClassifier(ccp\_alpha=0.01, criterion='entropy', max\_depth=10, min\_samples\_split=20, random\_state=0, splitter='random')

### Najuticajniji atributi



Za odabrani model i hiperparametre, može se odrediti feature importance, tačno iz toga se vidi koji atributi su i najznačajniji. Uoči toga iz našeg X možemo odbaciti manje bitne kolone, Embarked, SibSp i Parch, i opet trenirati model.

Ovo daje malo lošiji model što se tiče unazad već korišćenih ocena. Maltene sve opadne osim preciznosti koja poraste.

# Zaključak

Na zadati problem se kroz ovaj projekat odabrao najpogodniji model sa parametrima za predviđanje preživelih. Nakon obrade trening skupa, isti model primenjen je i na testni skup test.csv. Procene su upisane pored zadatih podataka i izvezene u vidu test\_set\_predictions.csv.

Nakon inicijalnog treninga izabran je DecisionTreeClassifier sa test\_size 0.2, ocenjen je sa 77.36% ispravnom procenom. Potom nakon unakrsne validacije i ispravljanja hiperparametara, over-fitting-a model leži na pouzdanijih i realnijih 71.08% ispravne procene. Konačno, nakon uklanjanja manje bitnih atributa, određenih kroz feature importance, redukovani model ima 65.78% ispravnu procenu.

### Moguća unapređenja

U samom projektu pojedini delovi se mogu doraditi, naime širi odabir vrednosti test\_sizes, parametara u param\_grid i najbitnije koeficijenti u weighted\_score. Sve ove vrednosti su izabrane po uzoru par primera kojih imam sa vežbi i koji se generalno koriste, verujem doduše da bi se dozvoljavanjem da program odradi svoje kroz više iteracija treniranja i unakrsne validacije uz veći skup parametara mogao postići bolji rezultat. Sam odabir modela je bio mrtva trka između random forest i decision tree, ali je prevagnulo na jednu stranu baš zbog prioritizovanja odziva i f1 skora.

Dalja unapređenja podrazumevaju upotrebu titula iz imena kao i kabina u kojima su putnici bili smešteni, ovog puta je to izostavljeno jer bi zahtevalo mnogo više posla nego što bi doprinelo samom treniranju modela.