

Mašinsko učenje

Šta je ideja za narednih 6 časova?

- OSNOVNO razumijevanje?
- Softversko inženjerstvo - matematika je alat ne i suština
- Dodatno upoznavanje sa Pythonom i bibliotekama (biće malo čitanja dokumentacije)
- Knjige + vježbe, zapisivanje, članci i Jupyter Notebooks (zadaci za vježbu i primjeri)
- Cilj - da znamo koje su mogućnosti, ograničenja, problemi i potencijalni pravci za implementaciju jednostavnih problema
- Koncepti su bitni!

Kako Netflix predviđa šta ćemo gledati?

Kako definisati mašinsko učenje?

- Nije bitna definicija već koncept
- Matematička funkcija - imamo ulaz i želimo da "predvidimo izlaz"
- Kako predviđamo izlaz?
- Upotreba poznatih parova ulaza i izlaza kako bismo naučili obrazce unutar podataka (Supervised learning)
- Razlika u odnosu na tradicionalne pristupe - ne vrši se hard kodovanje obrazaca i pravila već se pravila "uče"

$$Y \cong f(X)$$

A sad malo podjela...

- Supervised - naš fokus, imamo poznate ulaze i izlaze koje koristimo kako bismo naučili obrasce koji se primjenjuju za parove ulaza i izlaza koje nismo prije vidjeli
- Labela - vrijednost koja je pridružena ulazu, vrijednost koju želimo da predvidimo (klasa kojoj ulaz pripada, brojevna vrijednost...)
- Šta ako ne znamo parove i ako trebamo pronaći neke pravilnosti u podacima koji nemaju labelu? - Unsupervised learning (grupisanje, klasterizacija podataka, pronalaženje anomalija)
- Reinforcement learning - treniranje pametnih agenata koji imaju interakciju sa sredinom (self-driving)

Dvije bitne klase problema

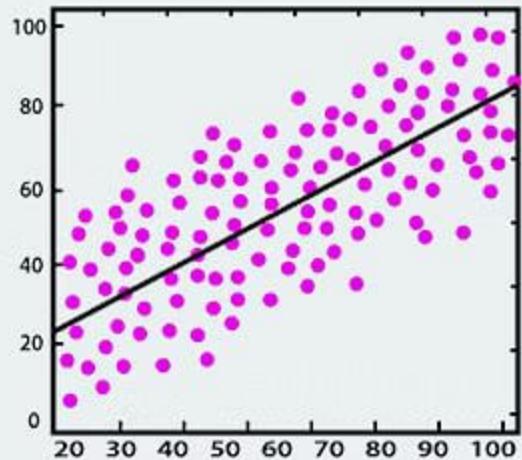
Supervised learning (nadgledano učenje):

1. Regresija

- Predviđamo kontinualnu vrijednost na osnovu ulaza
- Koja će biti cijena BitCoina za mjesec dana?

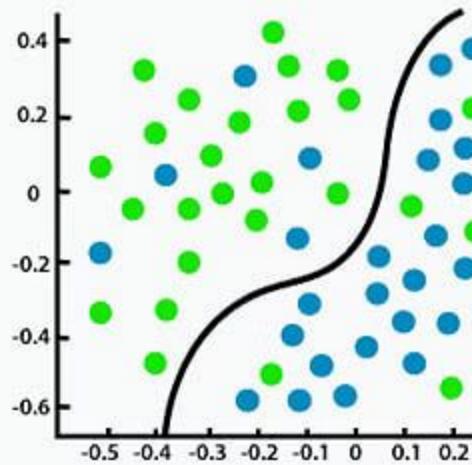
2. Klasifikacija

- Predviđamo diskretnu vrijednost iz konačnog skupa vrijednosti
- Da li je na slici pas ili mačka? Da li je email spam ili nije?



versus

Regression



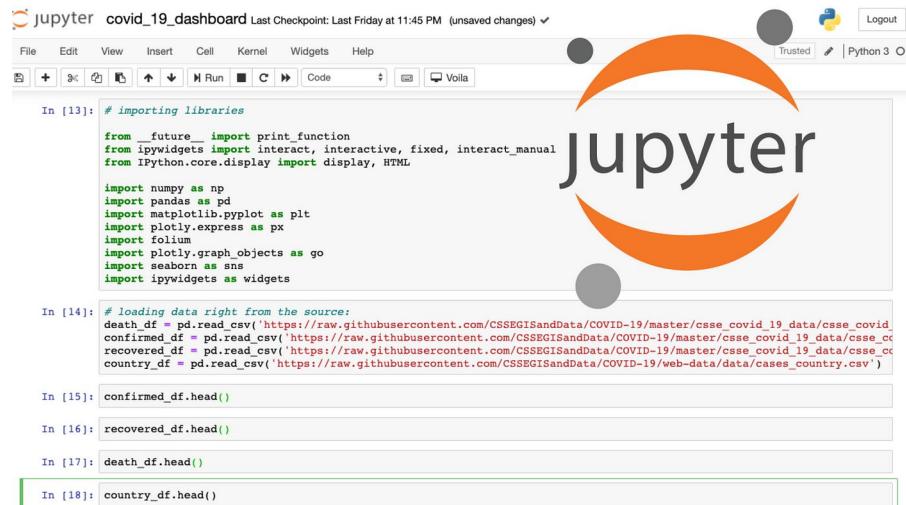
Classification

Bitne Python biblioteke

- Dolaze tek nakon što shvatimo suštinu
- Većina potrebnog koda će biti dosupnata sa komentarima u pratećim materijalima
- Biblioteke koje ćemo koristiti (industrijski standard):
 - NumPy
 - scikit-learn
 - Pandas
 - PyTorch

Prvo okruženje za razvoj

- Jupyter Notebook ili Jupyter Lab
- Interaktivni razvoj
- Brz proces iteracije i feedbacka
- I dalje isti Python kod samo ga ne morate pokretati iz terminala ili editora
- Instalacija



The screenshot shows a Jupyter Notebook interface with the title "jupyter covid_19_dashboard Last Checkpoint: Last Friday at 11:45 PM (unsaved changes)". The notebook has several code cells:

- In [13]:

```
# importing libraries
from __future__ import print_function
from ipywidgets import interact, interactive, fixed, interact_manual
from IPython.core.display import display, HTML

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
import folium
import plotly.graph_objects as go
import seaborn as sns
import ipywidgets as widgets
```
- In [14]:

```
# Loading data right from the source
death_df = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_deaths.csv')
confirmed_df = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_confirmed.csv')
recovered_df = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_recovered.csv')
country_df = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/web-data/data/cases_country.csv')
```
- In [15]:

```
confirmed_df.head()
```
- In [16]:

```
recovered_df.head()
```
- In [17]:

```
death_df.head()
```
- In [18]:

```
country_df.head()
```

Regresija u Pythonu

- Imamo ulaze, X, i izlaze, Y
- Potrebno pronaći parametre a i b tako da važi $Y = aX + b$
- Često nije moguće - podaci u stvarnosti nisu savršene funkcije, postoji šum
- Želimo da aproksimiramo najbolje rješenje?
- Bitna ideja - minimizacija greške
- Šta bi bila greška u slučaju regresije?
- Udaljenost linije od svake od tačaka
- Pronalazimo parametre tako da je udaljenost minimalna

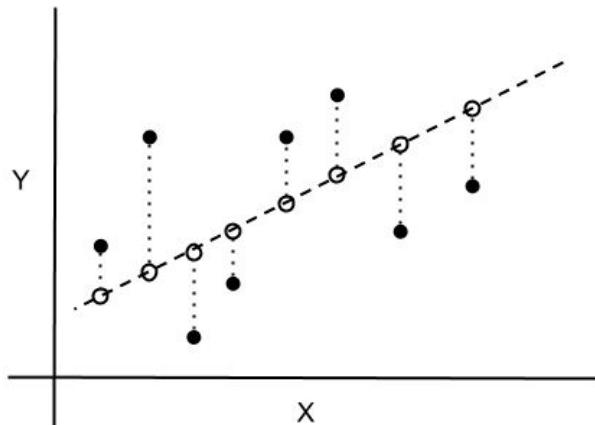
$$\hat{\alpha} = \bar{y} - (\hat{\beta} \bar{x}),$$

$$\hat{\beta} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

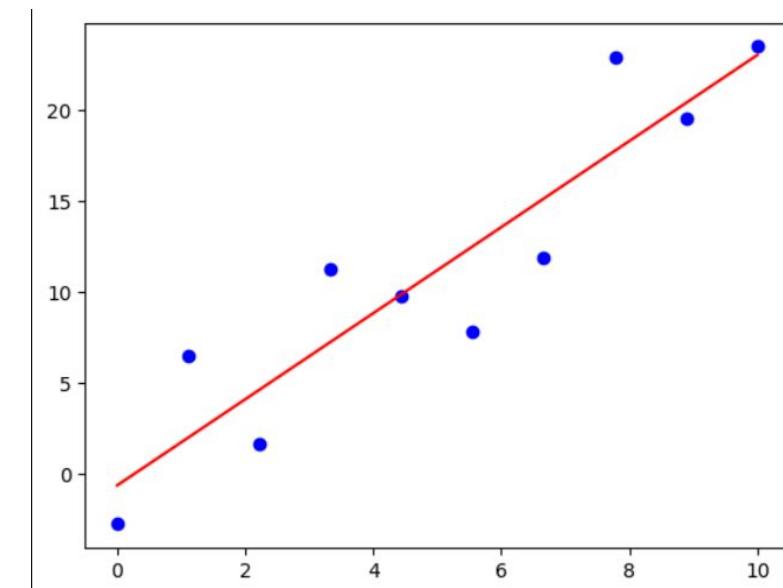
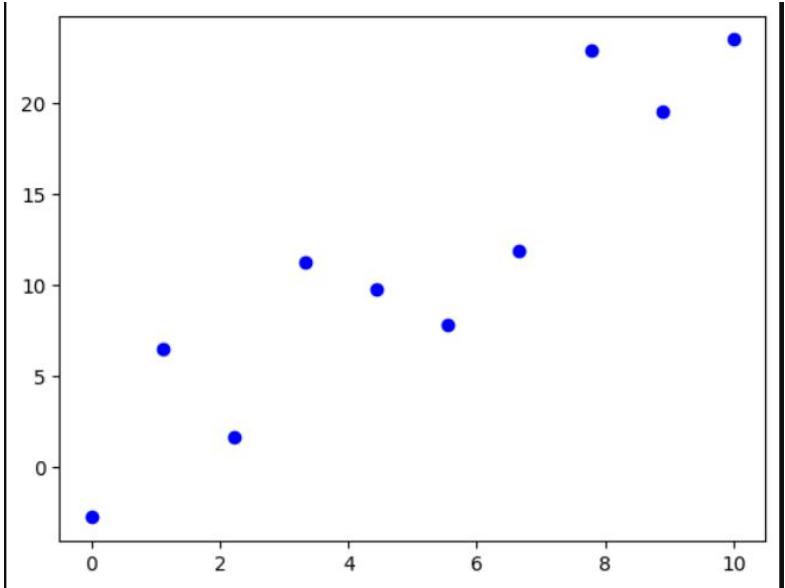
$$= \frac{s_{x,y}}{s_x^2}$$

$$= r_{xy} \frac{s_y}{s_x}.$$

- Observed Values
- Predicted Values
- - - Regression Line
- Y Scale Difference Between Observed and Predicted Values



```
● ● ●  
coefficients = np.polyfit(X, Y, 1)  
polynomial = np.poly1d(coefficients)
```

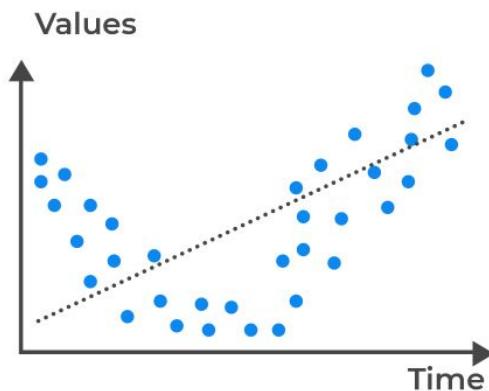


Da li imamo dobar model?

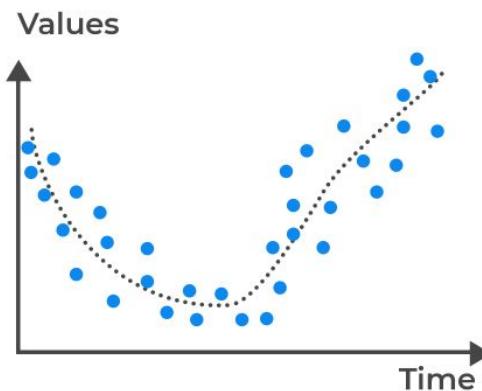
- Nisu svi podaci linearни - ne možemo linearnom funkcijom da aproksimiramo kvadratnu i obrnuto
 - Underfitting - naš model nije dovoljno kompleksan za dati problem (pokušavamo linearnom funkcijom aproksimirati kvadratnu)
 - Overfitting - naš model je prekompleksan, previše fittuje podatke koje imamo, ne može da generalizuje (koristimo polinom n-tog stepena da aproksimiramo $aX + b$)



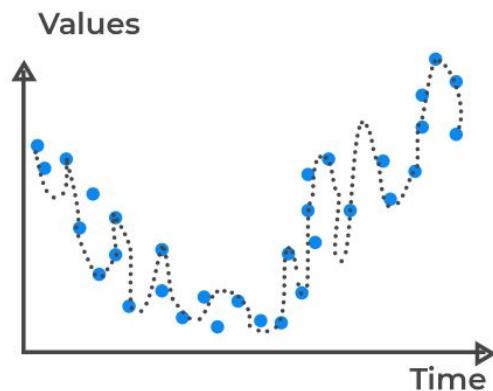
Generalization and Overfitting



Underfitted
(High bias error)



Good Fit/Robust
(Balance between
bias and variance)



Overfitted
(High variance error)

Trening i testni skup

- Model obučavamo na trening skupu - poznati parovi ulaza i izlaza (x koordinate koje imaju pridruženu y koordinatu) - koristimo da pronađemo konstante a i b
- Tesk skup koristimo za provjeru - kako model generalizuje na podakte koje nije video (neki vid testiranja rada u produkciji)
- Cilj ocijeniti - da li je naš model "pametan" ili je "učio napamet"
- Tipično:
 - 70-80% trening
 - 20-30% test
- Testni skup se ne smije koristiti za trening!

Klasifikacija u Pythonu

- Imamo skup ulaznih vektora, X , i svakom vektoru je pridružena neka klasa (labela) koje čine izlazni vektor, Y
- Potrebno je pronaći funkciju koja će novi vektor ulzani vektor svrstatи u odgovarajuću klasu
- Većina algoritama zasnovana na gradijentnom spustu, biće obrađeni na narednim predavanjima
- Jednostavniji pristup - K-Nearest Neighbors

K-Nearest Neighbors

- Za određeni broj podataka već znamo u kojim su klasama
- Za novi ulazni podatak za koji ne znamo klasu pronađemo distancu između svih vektora unutar skupa X
- Sortiramo distance i uzmemmo K najbližih vektora i provjerimo njihove klase
- Klasa koja se najčešće pojavljuje je predviđena klasa
- Broj susjeda koje gledamo bitan faktor - definiše granicu odlučivanja
- Nedostatak - brzina i memorija - moramo čuvati sve podatke koji su poznati i izračunati sve parove distanci
- Možemo dodijeliti težinu nekoj tački u zavisnosti od udaljenosti

Scikit-learn biblioteka

- Jedan od Vaših najboljih prijatelja u mašinskom učenju
- Gotove implementacije većine algoritama za regresiju, klasifikaciju i brojne druge probleme i zadatke
- Uvijek prvo pogledati da li postoji već gotova implementacija u scikit-learn biblioteci prije razvijanja sopstvenog algoritma!
- Gotove klase i unificiran interfejs - često se korišteni algoritam može promijeniti sa promjenom naziva klase koja se koristi
- Sadrži i veliki broj setova podataka na kojima se može učiti mašinsko učenje



```
from sklearn.neighbors import KNeighborsClassifier  
  
knn = KNeighborsClassifier(n_neighbors=3)  
  
knn.fit(X_train, y_train)  
  
y_pred = knn.predict(X_test)
```

Kako da znamo je li model dobar?

- METRIKE!
- Regresija jednostavne - MSE ili MAE
- Klasifikacija malo kompleksnija
 - Naivno bi bilo gledati tačnost (koliko smo klase dobro pogodili)
 - Podaci često nisu balansirani
 - Recimo radimo detekciju raka pluća - 99% stanovništva ga nema, ali nam je jako bitno da detektujemo onih 1% koji ga imaju, tako da nam tačnost od 99% ne znači puno

Zadaci za vježbu/razmišljanje/istraživanje

- Kako prilagoditi kNN algoritam da radi regresiju umjesto klasifikacije?
- Koje metrike bismo mogli još uvesti umjesto tačnosti za ocjenjivanje klasifikacionih modela?

Zaključak

- ML - skup algoritama, učenje obrazaca
- Supervised, unsupervised i RL
- Regresija = kontinualne vrijednosti
- Klasifikacija = diskretna vrijednost iz predefinisanog skupa
- Trening vs test skup
- Regresija i numpy.polyfit
- Overfitting vs underfitting
- Klasifikacija i kNN sa KNeighborsClassifier iz sklearn
- Evaluacija i zašto tačnost nije dobra?

Hvala na pažnji!