18. Strojové učení s využítím regrese a klasifikace

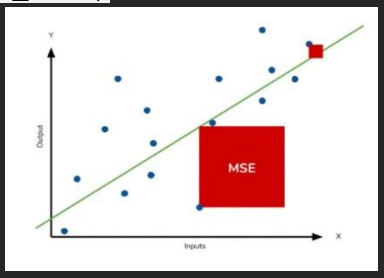
Regrese

Lineární regrese je metoda, díky které můžeme najít přímku tak, aby všechna data k ní byla co nejblíže.

Data musíme procházet ručně, počítač nedokáže rozeznat, co je ze začátku blbost a co není. Proto šílené a nereálné ceny musíme odebrat. Pokud jsou nějaká data opravdu vysoká nebo nízká, ignorujeme je, jelikož je rozptylují. Je důležité co nejméně zúžit rozptyl. Počítač podle grafu vytvoří funkci, která vrátí přímku, která prochází přibližně prostředkem dat.

Vývoj takového modelu spočívá v tom, že dáme našemu modelu 80% našich data a jejich správné hodnoty jako data testovací. On podle nich přibližně vytvoří lineární regresi a vytvoří funkci a přímku.

1.) Data jsou rozdělena na `X_train, X_test, y_train, y_test`, kde X_train jsou data trénovací a vstupní hodnoty, Y_train jsou následně správné výsledky pro tato testovací data. X_test jsou testovací data, tudíž těch 20 procent a Y_test jsou jejich správné výsledky.
2.) Model, v tomto případě lineárné regrese, se z 80% dat naučí a vytvoří z ní přímku. `model.fit(x_train,y_train)`
3.) Poté model vyzkoušíme pomocí `pred = model.predict(x_test)`
4.) Zjistíme MAE a MSE – mean_absolute_error a mean_squared_error. MAE je tedy průměrná hodnota rozdílů mezi predikovanou a reálnou hodnotou. MSE zjistí, jak daleko je jeho bod od přímky a poté podle délky vytvoří kostku (proto square)
5.) A teď máme víceméně funkční model, který se dá poté exportovat a využít k predikci. K tomu můžeme použít knihovnu pickle na serializaci.



Příkladem využití takovéto lineární regrese je například počítání ceny aut na základě určitých parametrů nebo výpočet ceny domů,

Random Forest Regression

Random Forest Regression funguje na principu vytváření rozhodovacích stromů. Každý strom se poté trénuje na jiných datech a predikuje data podle sebe, načež se nakonec všechny spojí. Výhoda rozhodovacích stromů je, že nemusí být vůbec lineární.

Klasifikace

Klasifikace spočívá v určování na základě vstupních dat. Máme například 10 hrušek a 10 jablek. Musíme počítači říci, co je jablko a co je hruška a řekneme mu například, že čím více červené, tím více je šance, že se jedná o jablko. Čím více zelené, je to zase hruška. Podle dat zjistíme, kde přibližně se nachází přechod mezi tím, kde je jablko a kde je hruška. Vždycky se stane, že některé jablka jsou zase zelená. Žádný algoritmus ovšem nemůžeme být naprosto přesný.

V programování se využívá předpřipravená knihovna pro klasifikace Irisů – květin. Podle jejich barev a podle délky listků se poté rozeznává konkrétní druh. V Pythonu toho docílíme pomocí `iris = datasets.load_iris()`.

Následně se data standartizují tím, že se nalezne střední hodnota a standartizovaná data budou následně obsahovat směrodatnou odchylku od průměrů

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]]
[[-0.90068117  1.01900435 -1.34022653 -1.3154443 ]
 [-1.14301691 -0.13197948 -1.34022653 -1.3154443 ]
 [-1.38535265  0.32841405 -1.39706395 -1.3154443 ]
 [-1.50652052  0.09821729 -1.2833891  -1.3154443 ]]
```

Data a standartizovaná data

Standartizovat můžeme pomocí

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(iris.data)
standard_iris_data = scaler.transform(iris.data)
```

Od této chvíle můžeme tedy zjistit průměr pomocí metody `.mean()` a směrodatnou odchylku pomocí `.std()`

Klasifikaci je možné dělat pomocí lineární regrese či pomocí rozhodovacích stromů. Teď se zaměříme na lineární regresi. Jako vždy je potřeba udělat model, následně modelu poskytnout data, data standartizovat a otestovat. V případě klasifikace můžeme poté získat pomocí `classification_report()`

1.) Precision - procentuální úspěšnost celkého odhadu
2.) Recall - procentuální úspěšnou správného určení (třeba že pacient je nemocný), kolik špatně jej nezajímá
3.) F1-score -Průměr mezi Precision a Recall, čím více jsou vyvážené tyto hodnoty, tím lepší je F1-score
4.) Support – Kolik bylo konkrétních příkladů na vstupu (Tedy v případě Irisů kolik bylo od každého druhu)

Je možné využít i rozhodovací stromy pomocí Random Forest Regression, ty se v případě klasifikace irisů hodí více a jsou přesnější a efektivnější.

Rozhodovací stromy

Pokud máme auto podle najetých kilometrů, značka, atd. Podle toho, jaké splňuje požadavky, se posunuje ve stromu. Nejdříve třeba podle modelu, potom podle značky, na konci podle kilometrů dá cenu.

Rozhodovací stromy mají problém s tím, že pokud máme záznam pro 4000 a 6000, tak v případě, že máme cenu pro 5000, nevědí.