

Prediktiv analys

FÖRELÄSNING 9

Dagens fråga

- Vilken är den mest intressanta dokumentär du sett?

Dagens agenda



Data pre-processing



Hantera NULL värden



Feature scaling: normalisering, standarinsering, robust scaler



Outliers – upptäcka och hantera



Spam classifier data – Python

Förra föreläsning

- Classification trees
- Rule inference
- Imbalanced data
- Naive Bayes

Frågor till dagens lektion

Vilka fem steg ingår
i data pre-processing?

Vad är NaN värdet

Hur hanterar vi NaN
värdet?

Vad är feature
scaling och vad
använder vi det till?

Vilka vanliga feature
scaling metoder
finns och när ska vi
använda de?

Vad är en outlier?

Hur uppstår
outliers?

Hur upptäcka
outliers?

Varför är det viktigt
att upptäcka
outliers?

Hur kan vi undvika
outliers?

Vad är natural
language processing
(NLP) och vad
använder vi det till?

Vilka applikationer
är NLP bakom?

Vilka två tekniker
använder NLP?

En bra modell är inte tillräckligt



- Att använda Machine learning inkluderar processen å hitta den bästa modell som predikterar från ett givet dataset.
- Men vi behöver mer än detta! Vi måste också:
 1. **Transformera datan** – datan måste vara på ett format som modellen kan läsa av. Vi vill också få fram underliggande struktur i data som gör att modellerna fungerar bättre
 2. **Hyperparameter till modellen** – vi måste ta reda på vilka parameter modellen ska ha för att prediktera så bra som möjligt

Dokumentera alltid alla val du gör!

Data preprocessing

1. Handling Missing values (NaN)
2. Outliers
3. Feature scaling
4. Handling categorical features (One-hot encoding/Dummy variables)
5. Multicollinearity
6. Feature selection

Null värdet

- NULL värdet är när det saknas mätningar i datasetet. Det är inte det samma som 0
- I Python beteckas saknade NULL värdet med NaN (Not a Number)
- Varför saknas det värden?
 - Data kan skadas på grund av felaktigheter
 - Det kan finnas ett misslyckande i registreringen av värdena på grund av mänskliga- och systemfel
 - Användaren har inte angett värdet medvetet
- Vi vill alltid behålla så mycket information som möjligt, så vi vill ofta försöka undvika radera värden

Hantera Null värden



Brukar ofta saknas värden i verkliga dataset. Inga modeller kan hantera dessa NaN värden på egenhand. Vi måste fixa.

- Kolla om det finns NULL värden i datasetet
`df.isnull().sum()` *# Returns column names along with number of NaN values in that column*
- Ta bort rader eller kolumner med NULL värden. Är det stora delar av en kolumn som saknar värden är det bättre att ta bort hela kolumnen (ca 50%) Saknas bara enstaka värden kan man ta bort raden.
`df.dropna()` *# Takes various parameters: axis – 0 row, 1 columns.
inplace – True to change df*

Hantera Null värden

Är det ett stort dataset med flera hundra tusen datapunkter spelar det ingen roll om vi tar bort några rader. Har vi ett mindre dataset på några hundra punkter kan man förlora viktig information vid att ta bort data.

Exempel: Från en undersökning kan 20% av de frågade fått bli att svara på en fråga. Resterande data är värdefull så vi vill inte ta bort de raderna, men istället substituera.

- Imputation - substituera saknade värden i datasetet. Vi kan definiera vår egen funktion eller använda SimpleImputer från sklearn

```
from sklearn.impute import SimpleImputer
```

```
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')  
imputer = imputer.fit(df)  
df = imputer.transform(df)
```

- Man kan också använda metoden `fillna()`

Dokumentera alltid hur man hanterar NaN värden

Hantera Null värdet

- **Mean** – är vanligaste metoden. Numeriska värden
- **Median** – finns det många outliers vill man använda median
- **Mode** – flest förekomna värdet. Kategoriska värden
- **Konstant** – man kan ange det värdet man vill. I vissa fall kan 0 vara aktuellt
- **Forward/backward fill** – ge föregående eller nästa värde. Används för tidsberoende data
- **Interpolation** – använder värden från kringliggande punkter. Pandas metod *interpolate()* finns det många valmöjligheter på metod som linear, polynomial, quadratic

Outliers

- En **outlier** är en observation som är långt unna andra observationer
- En outlier kan bero på variation i mätningen eller indikera experimentell fel
- Outliers kan uppstå när datan samlas, felaktig inmatning, felrapportering, samplingsfel men också ett exceptionellt men sant värde
- Om möjligt kan outliers exkluderas från datasetet. Men vi kan också gå miste om viktig information för outlier kan vara bare ett väldigt stort värde
- Varför identifiera outliers? En del prediktiva modeller är sensitiva till attributens skala och distribution. Outliers kan förstöra och vilseleda träningen med att det tar längre tid att träna modellen, ge låg accuracy och i slutändan ge sämre resultat



To Drop or Not to Drop - Outliers

- De förstör modellen. Många statistiska metoder är väldigt sensitiva till outliers. Men man kan inte droppa en outlier *bara för* att det är en outlier. Därför måste man studera de innan.

Drop outliers om:

1. Det är uppenbart den beror på felaktig angiven eller mätt data.

Ex: I ett dataset om en klass är det en student med ålder 500 år.

2. Om outlier inte ändrar resultaten men påverkar antaganden.

Att ta bort outliern i figuren påverkar inte regressionslinjen

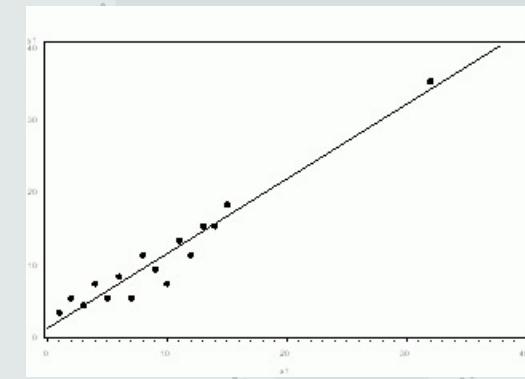
3. Om outlier påverkar båda resultaten och antaganden är det viktigt att testa och dokumentera hur den påverkar modellen

Att ta bort outlier kan ha en påverkan på modellen

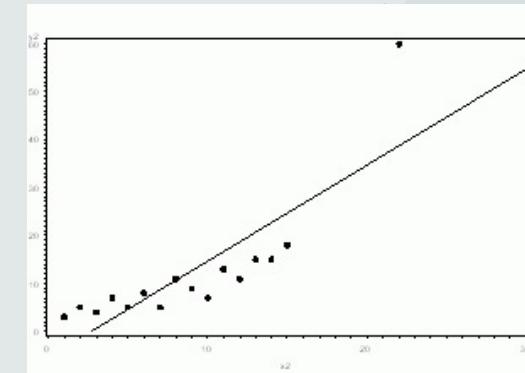
4. Om outlier skapar en relation där det faktiskt inte är ett samband.

Sambandet är skapat av outlieren. Utan den är det inget samband

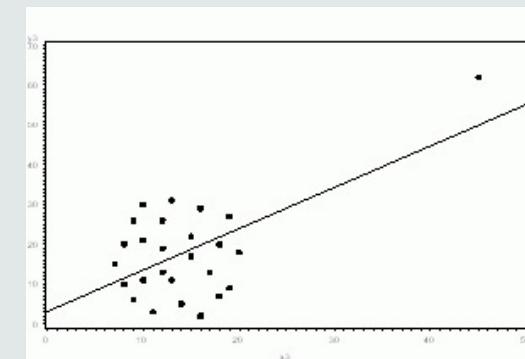
Dokumentera alltid alla borttagne outliers



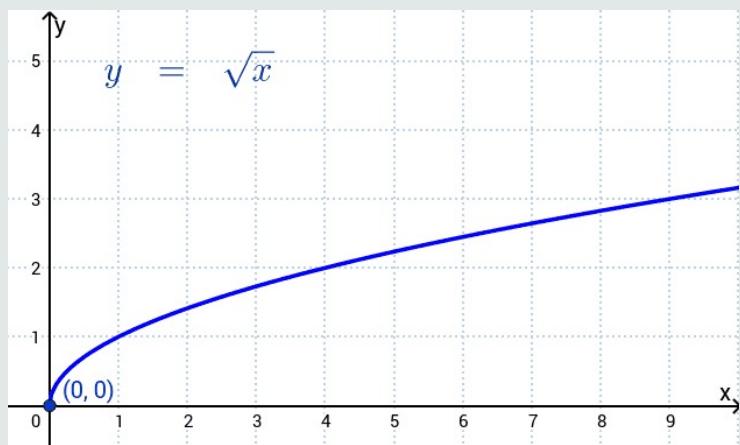
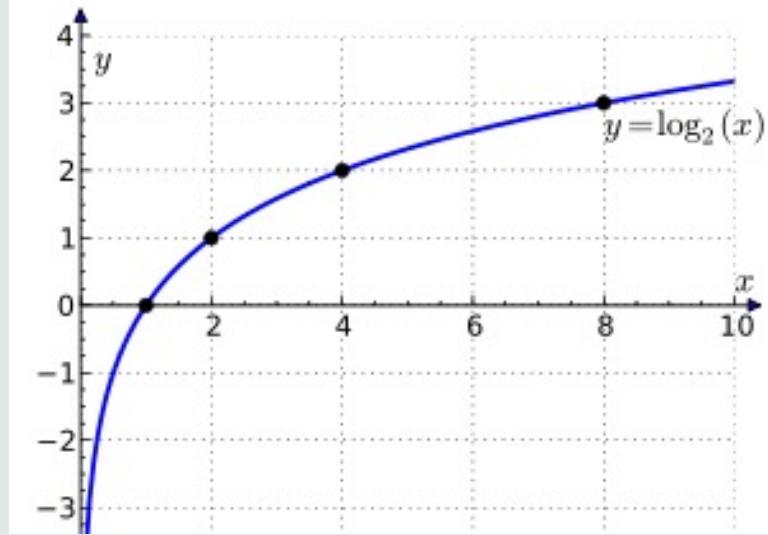
2.



3.



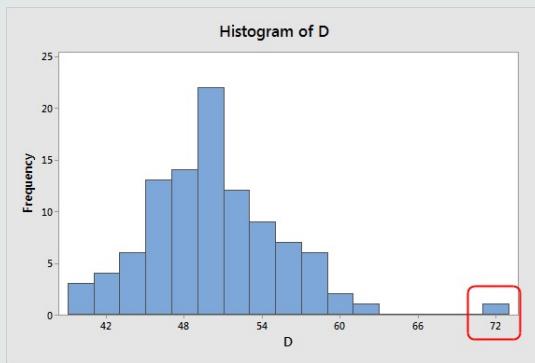
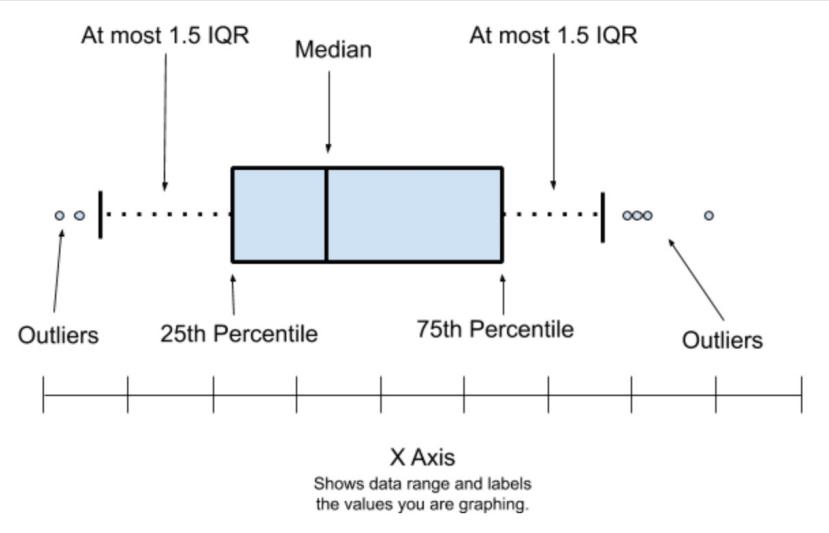
4.



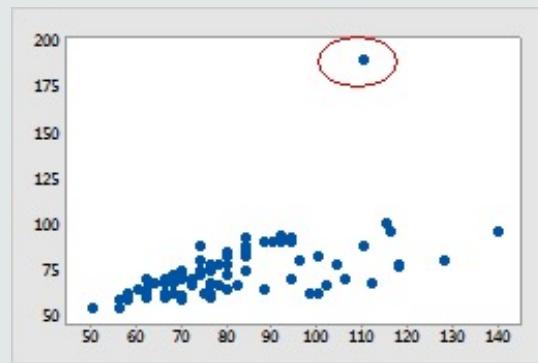
To Drop or Not to Drop - Outliers

- Vad med när man **inte** borde droppa outliers, vad gör man då?
- Transformera datan med kvadratroten \sqrt{x} eller logaritmen $\log(x)$. Dessa gör höga siffror mindre. Se figurerna hur y inte växer särskilt när x blir större och större.
- Transformera då alla datapunkter till en feature (kolumn).
- Använda *feature scaling* med robust scaler till exempel (mer om detta snart)

Box plot



Histogram



Scatter plot

Upptäcka Outliers - Visualiseringstekniker

- **Box plot** – grafisk sätt att visa distribution till data. Outliers plottas med egna symboler (cirkler)
- **Histogram** – några observationer bortanför huvudgruppen
- **Scatter plot** – mellan två variabler kan man se om en outlier är långt unna det ”vanliga” sambandet till variablerna

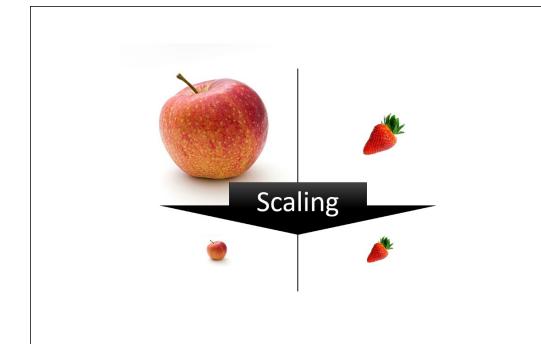


Outliers – hur undvika?

- Hur undvika och hantera outliers?
- Stort dataset gör att varje datapunkt inte har lika stor ”vikt”
- Hur stor är för stor outlier? Svårt att säga. Men tio gånger så stor som övre gräns gör mer skada än en som är dubbelt så stor.
- Ha bra ”domänkunskap” – kunna väl sin data.
- Ta bort hela raden där outlieren finns (försöka undvika)
- Justera ner värdet om man ser att den beter sig likt som andra ”övriga värden”.
- Ge nytt värde om det visar sig vara mätfel. Genomsnittet av de andra datapunkterna till exempel.
- Transformera datan
- Det finns algoritmer kan hantera outliers. Brukar ha hög ”computational cost”.

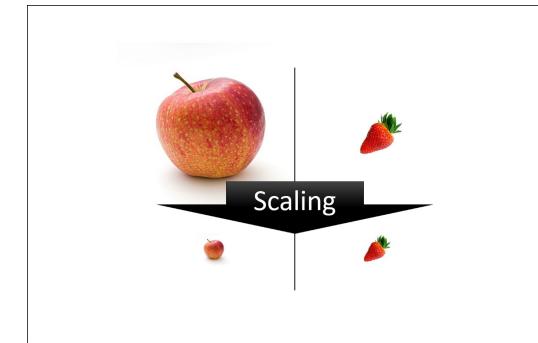
Feature Scaling

- **Standardisering** - process för att göra datan jämförbar med annan data. Olika variabler blir satt i samma skala så att de kan jämföras.
- **Feature scaling** är en teknik för att standardisera oberoende features till ett fast intervall.
- **Avståndet** mellan alla datapunkterna förblir det samma.
Det är värdet de har som ändras
- **Feature scaling** görs under pre-processing för att hantera mycket varierande storlekar på datan. Om det inte görs tenderar algoritmen att vikta stora värden ännu högre och betrakta mindre värden ännu mindre.



Feature Scaling

- Det är alltså input variabler x som skalas, INTE target output y .
- Skalare används för kontinuerliga features.
- Avsikten med scaling är att minska variansen i datan så att de flesta prediktionerna hamnar i området med mest data.
- Användning av scaling kan öka noggrannheten (accuracy).
- *Exempel:* Om vi inte använder feature scaling kan värdet 3000 meter värderas större än 5 km. Detta stämmer inte och vi får fel prediktion. Använder vi feature scaling får vi alla värden till samma storlek och tar itu med problemet.



Feature scaling

De två vanligaste feature scaling metoderna är normalization och standardization.

- Normalization eller **Min-Max Scaling** – Skalar värden till en skala mellan 0 och 1

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Exempel

$$data = [76, 84, 69]$$

$$x_{new} = \frac{76 - 69}{84 - 69} = 0.47$$

Feature scaling

- **Standardization** – Skalar värden så de har en distribution med mean=0 och varians=1, så koncentrerad kring 0.

$$x_{new} = \frac{x - \text{mean}(data)}{\text{std}(data)} = \frac{x - \mu}{\sigma}$$

Std = standard deviation. Berättar som spridning i av värden till features

Exempel:

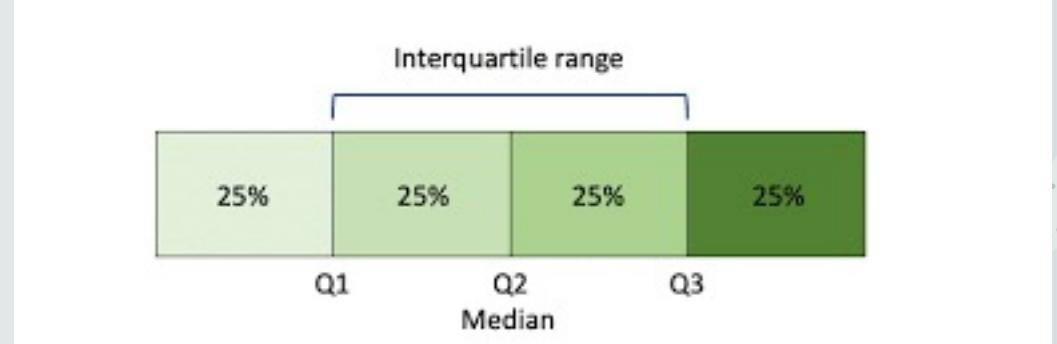
$$data = [76, 84, 69]$$

$$\text{mean} = \mu = \frac{\sum x_i}{n} = \frac{76 + 84 + 69}{3} = 76.3$$

$$\text{std} = \sigma = \sqrt{\frac{\sum(x - \mu)^2}{n - 1}} = \sqrt{\frac{(76 - 76.3)^2 + (84 - 76.3)^2 + (69 - 76.3)^2}{2}} = \sqrt{\frac{112.7}{2}} = 7.5$$

$$x_{new} = \frac{76 - 76.3}{7.5} = -0.04$$

Feature scaling



- Robust data scaling – median och interquartile range (IQR) används så skaleringen inte är känslig för outliers
- IQR – de 50% mellersta datapunkter när de sorteras från lägsta till högsta värde. Alltså skillnaden mellan 75th och 25th percentiles eller Q3-Q1. "Boxen" i en box-plot.
- För att den delar alla datapunkter med intervallet mest data är och inte hela skalan, undviker vi effekten av outliers.

$$x_{new} = \frac{x - \text{median}(data)}{Q_3 - Q_1}$$

Feature scaling - Python

Scikit-learn: Sklearn.preprocessing

- StandardScaler() – standardization
- MinMaxScaler() – normalization
- RobustScaler() – Skalarer med en metod som är robust mot outliers

..osv finns **många** skaleringsmetoder



- **Min-max normalization** – kanske inte så användbar inom machine-learning. Bra för utforskning av data dock! Inte alltid så bra för att öka accuracy. Känslig för outliers.
- **Standardization** – väldigt bra ”go-to” scaler för kontinuerlig data. Bra för att öka accuracy. Känslig för outliers.
- **RobustScaler** – Robust mot outliers. Bra ”go-to” scaler.

Feature scaling – hur välja rätt?

- Vilka fem steg ingår i data pre-processing?
- Vad är NaN värden
- Hur hanterar vi NaN värden?
- Vad är feature scaling och vad använder vi det till?
- Vilka vanliga feature scaling metoder finns och när ska vi använda de?
- Vad är en outlier?
- Hur uppstår outliers?
- Hur upptäcka outliers?
- Varför är det viktigt att upptäcka outliers?
- Hur kan vi undvika outliers?
- Vad är natural language processing (NLP) och vad använder vi det till?
- Vilka applikationer är NLP bakom?
- Vilka två tekniker använder NLP?

Frågor för dagens lektion

Predicting Post Popularity

- Data Set Information:

This dataset summarizes a heterogeneous set of features about articles published by Mashable in a period of two years. The goal is to predict the number of shares in social networks (popularity).

- Features description

Number of Attributes: 61 (58 predictive attributes, 2 non-predictive, 1 target)

Predicting Post Popularity

Hands on



Improving Accuracy

- Som exemplet visar är det inte alltid helt lätt att göra prediktiva modller. Det finns mange strategier för att förbättra en modell:
 - Collect more and/or better data
 - Feature engineering
 - Pre-processing: Deal with outliers, missing data and scale of features
 - Feature selection methods
 - Regularization
 - Parameter tuning
 - Use more complex models
- Trots detta går det inte alltid att prediktera trots stora mängder data, komplexa modller och bästa metoder..

Vad har vi gjort idag?

- Data preprocessing
- Hantera NULL värden
- Outliers
- Feature scaling

Nästa lektion

- Optimering av prediktionsmodeller
- Endsamble methods
 - Random forest
 - Bagging
 - Bossting och AdaBoost
- Natural Langugae Processing NLP