

Prediktiv analys

FÖRELÄSNING 3

Dagens fråga

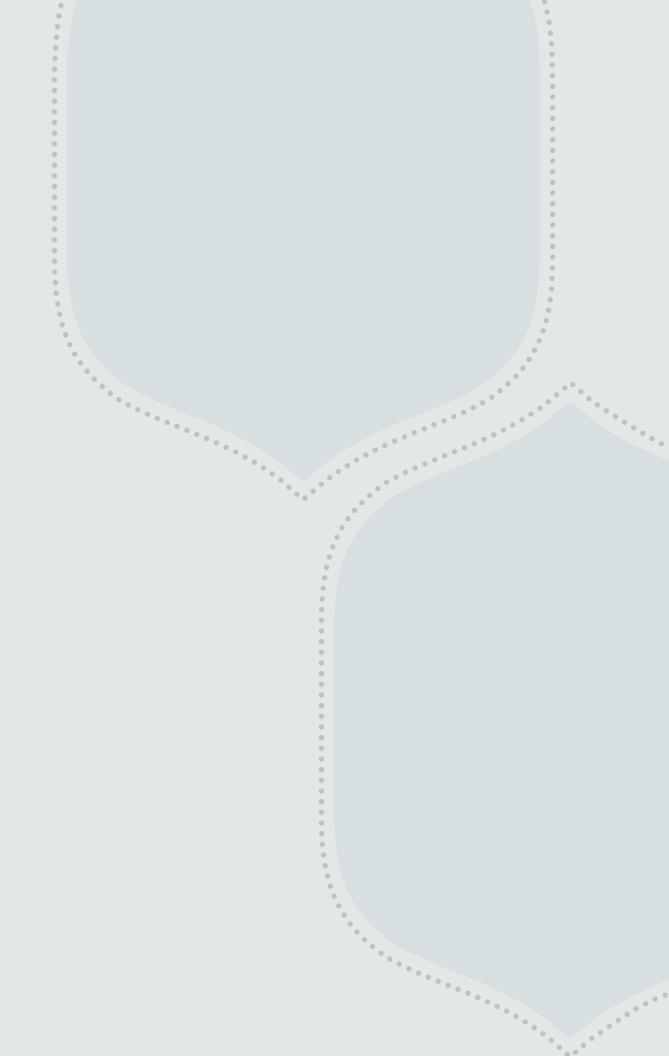
HAR DU NÅGRA FOBIER,
ISÅFALL VAD?

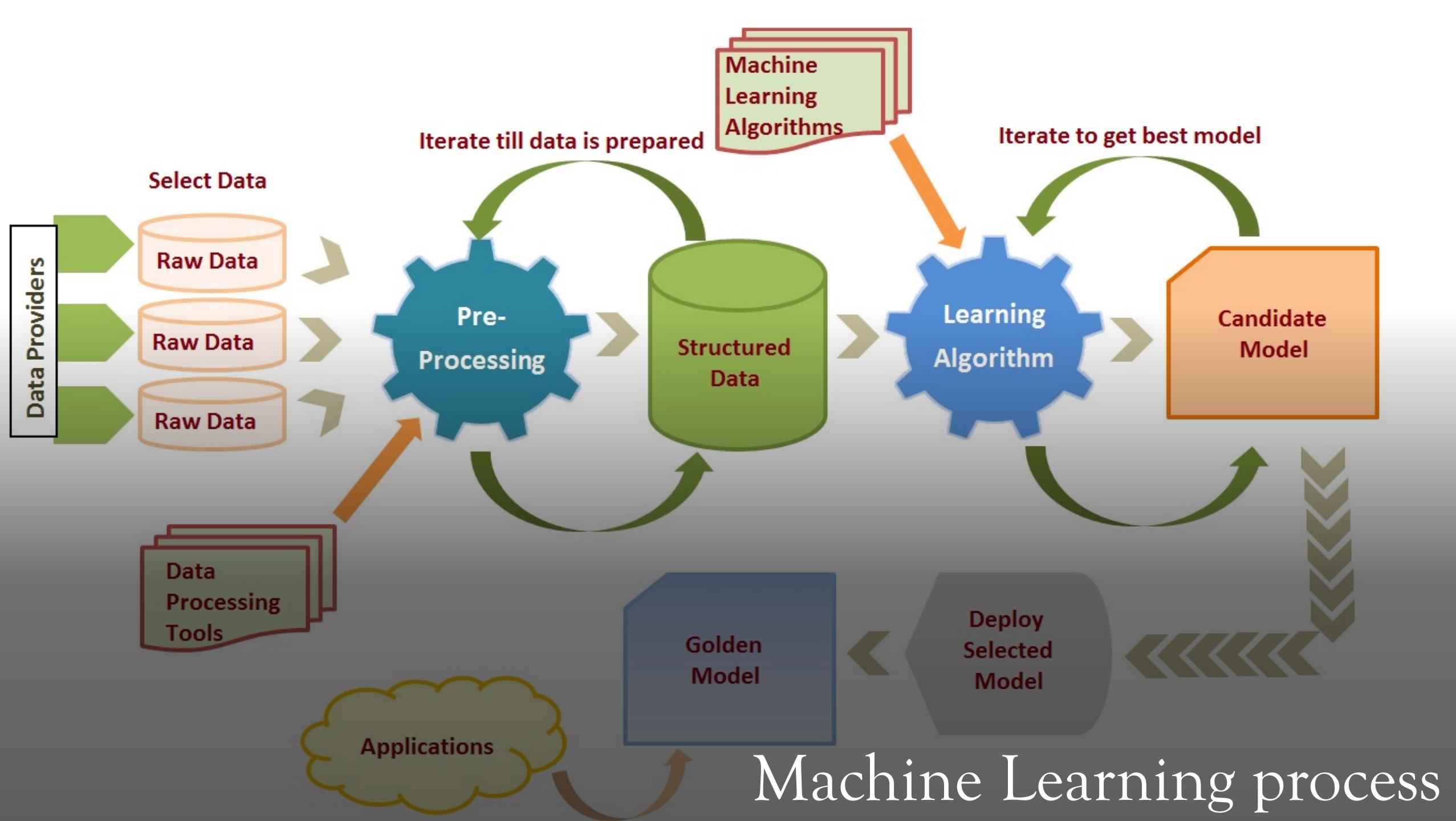


Dagens agenda

- Viktiga python bibliotek för prediktiv analys:
 - Numpy
 - Pandas
 - Matplotlib
 - Seaborn
 - Scikit-learn
- Prediktiv analys i python: viktigaste stegen
 1. Förbereda data
 2. Importera estimeringsobjektet
 3. Skapa instans av modellen
 4. Träna modellen
 5. Utvärdera modellen
 6. Prediktera

Förra föreläsning





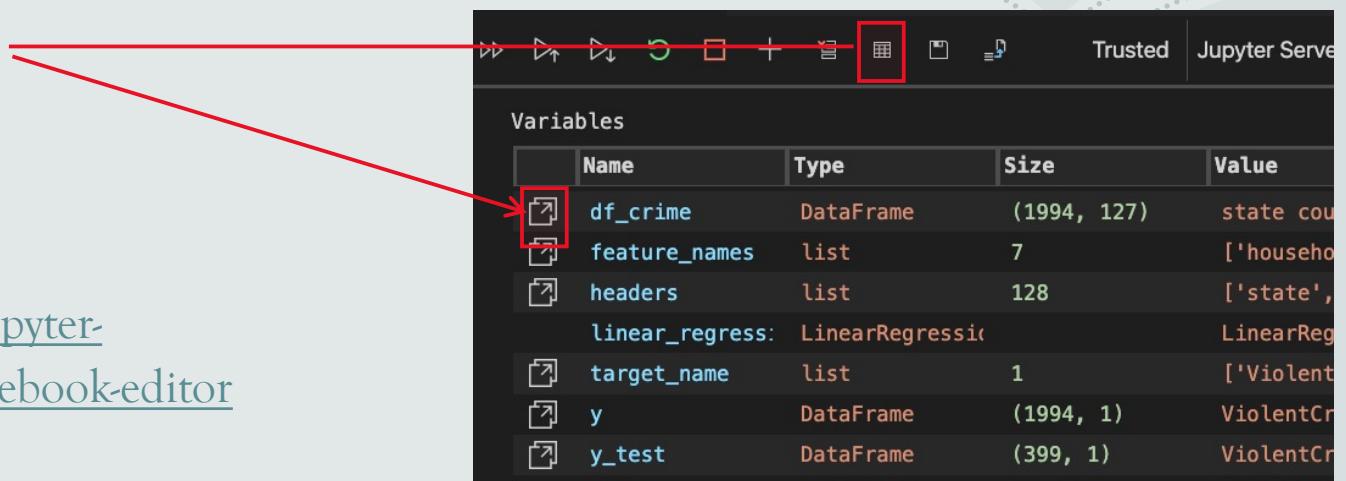
Vi följer specifika steg för att bygga en Machine learning modell



Detta är generella steg och man kan få hoppa fram och tillbaka flera gånger innan det blir rätt.

Jupyter notebook

- För att se alla variabler man har skapat
- Hjälp om Notebook i VSC:
- https://code.visualstudio.com/docs/python/jupyter-support#_intellisense-support-in-the-jupyter-notebook-editor



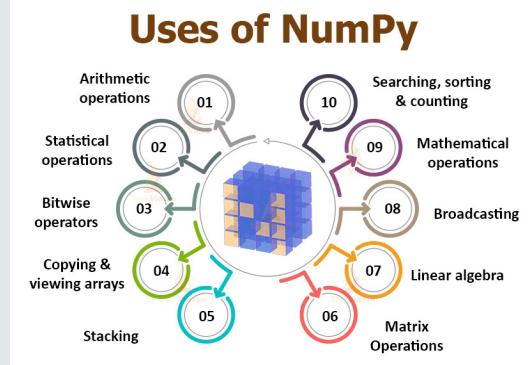
Viktiga python bibliotek för prediktiv analys

D

E

NumPy

- NumPy är ett bibliotek för Python-programmeringsspråket som ger stöd för stora, flerdimensionella matriser och matriser, tillsammans med en stor samling matematiska funktioner på hög nivå för att fungera på dessa matriser.
- NumPys huvudobjekt är den homogena flerdimensionella matrisen. Det är en tabell över element (vanligtvis siffror), alla av samma typ, indexerade med en tuple av positiva heltal.
- I NumPy kallas dimensioner axlar. Antalet axlar är rang eller dimension.
- Om vi skickar en lista med listor till np.array () skapas en tvådimensionell matris.
- Om du skickar en lista, med listor, med listor, skapas en tredimensionell array, och så vidare och så vidare.



```
distances = [10, 15, 17, 26, 20]
times = [0.3, 0.47, 0.55, 1.2, 1.0]
```

[8] ✓ 0.2s

```
> 
speeds = distances/times
print(speeds)
```

[9] ✘ 0.4s

Utan NumPy

```
...
-----
TypeError
/var/folders/82/t5bg_f9n6k9_ytn11s_ych740000gn/T/ipykernel_55368/2060879328
le>
----> 1 speeds = distances/times
      2 print(speeds)
```

Traceback (most recent call last)
/var/folders/82/t5bg_f9n6k9_ytn11s_ych740000gn/T/ipykernel_55368/2060879328

TypeError: unsupported operand type(s) for /: 'list' and 'list'

```
▷ ▾  
  distances = [10, 15, 17, 26, 20]  
  times = [0.3, 0.47, 0.55, 1.2, 1.0]
```

[8] ✓ 0.2s

Med NumPy

```
import numpy as np  
distances = np.array(distances)  
times = np.array(times)
```

[4] ✓ 0.3s

```
speeds = distances/times  
print(speeds)
```

[6] ✓ 0.3s

[...] [33.33333333 31.91489362 30.90909091 21.66666667 20.]

- Pandas är ett bibliotek för manipulation och analys av data.
- Den ger datastrukturer och operationer för att manipulera numeriska tabeller och tidsserier.
- Pandas används främst för dataanalys. Pandas tillåter import av data från olika filformat som CSV, JSON, SQL, Microsoft Excel.
- Pandas tillåter olika datahanteringsåtgärder som sammanslagning, omformning, val, samt datarengöring och datavridningsfunktioner.



- Series
 - 1-Dimensionell
 - Homogen – kan vara av alla typer (integers, strings, floats m.fl.)
 - Pandas Series objekt har alltid index
 - Supportar vektoriserade operationer
 - Automatisk datainriktning
- Dataframe
 - 2-Dimensionella
 - Potentiellt heterogen tabelldata
 - Har index för både radar och kolumner
 - Kan ses som "dict-likas" behållare för serier av objekt
 - Liknar SQL tabeller eller Excel tabeller



```
In [2]: import numpy as np
```

```
In [3]: # The importing convention  
import pandas as pd
```

```
In [4]: default = pd.read_csv('./datasets/credit_card_default.csv', index_col="ID")
```

```
In [5]: default.head()
```

```
Out[5]:
```

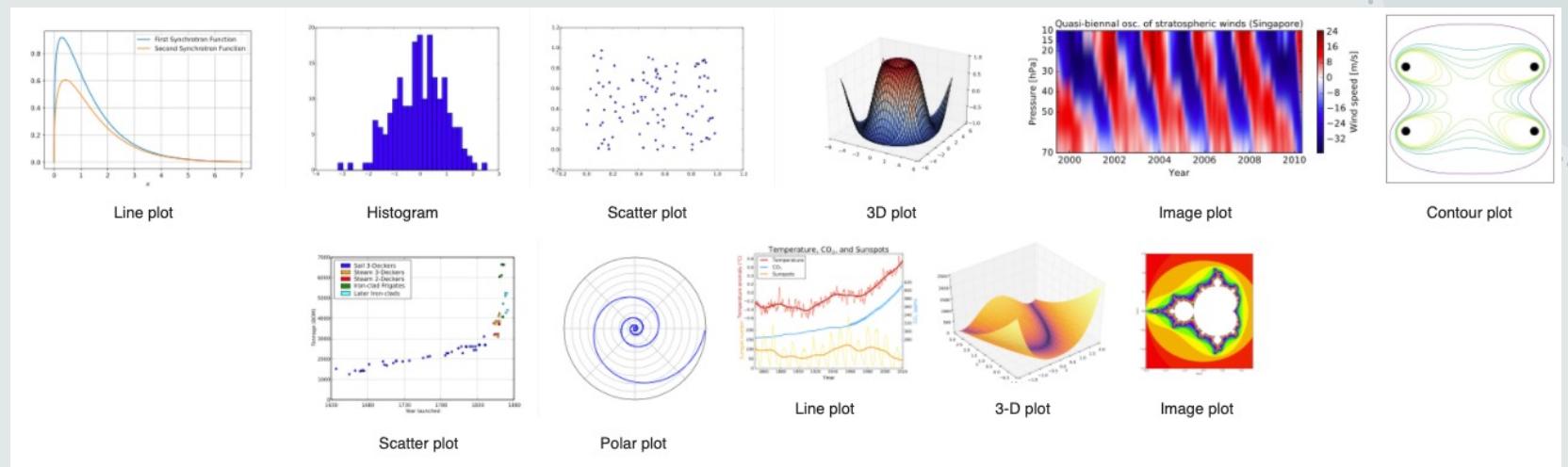
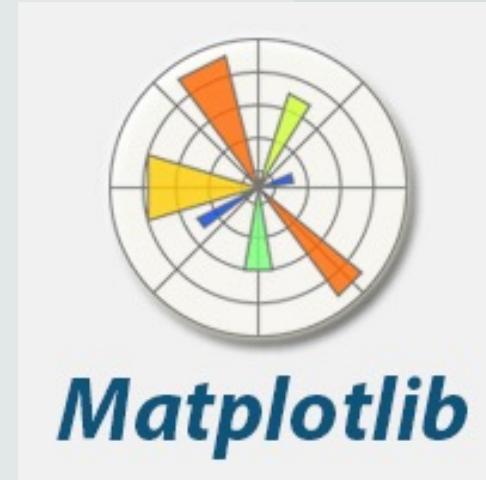
ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	PAY_5	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	...
1	20000	2	2	1	24	2	2	-1	-1	-2	...	0	0	0	0	689	...
2	120000	2	2	2	26	-1	2	0	0	0	...	3272	3455	3261	0	1000	...
3	90000	2	2	2	34	0	0	0	0	0	...	14331	14948	15549	1518	1500	...
4	50000	2	2	1	37	0	0	0	0	0	...	28314	28959	29547	2000	2019	...
5	50000	1	2	1	57	-1	0	-1	0	0	...	20940	19146	19131	2000	36681	...

5 rows × 24 columns

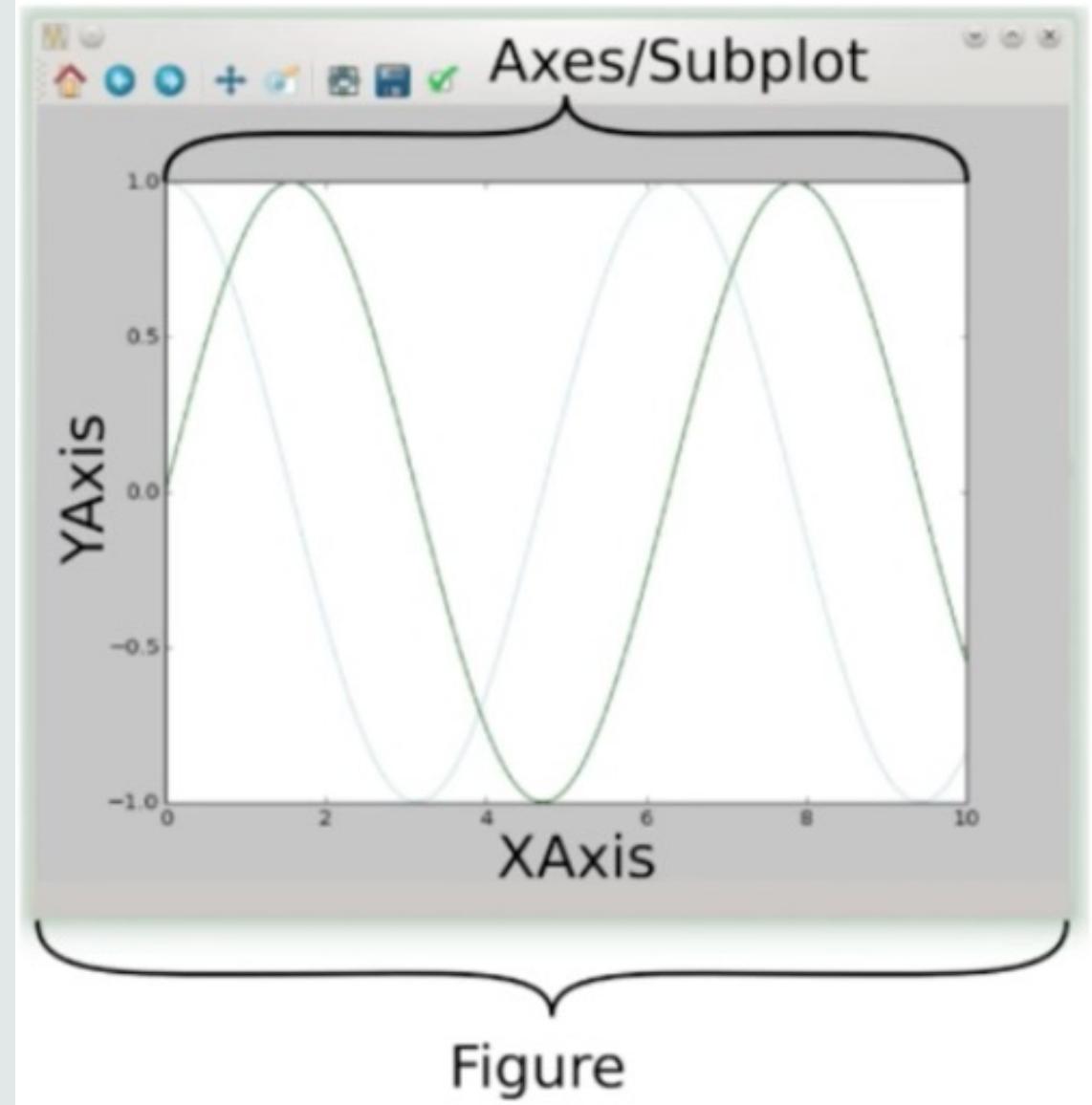
Pandas

Matplotlib

- Matplotlib är ett plotningsbibliotek för Python och dess numeriska matematikförlängning NumPy.



Interface



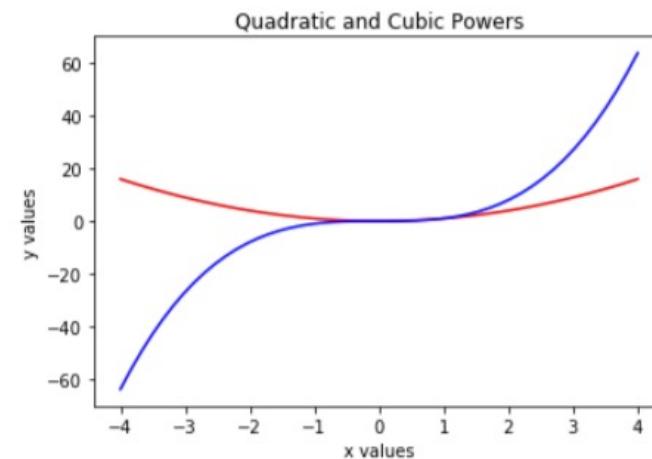
Interface

```
In [1]: import numpy as np
```

```
In [2]: import matplotlib.pyplot as plt  
%matplotlib inline
```

```
In [3]: x = np.linspace(-4,4)  
y1, y2 = x**2, x**3
```

```
In [4]: # Object Oriented Interface  
fig, ax = plt.subplots()  
ax.plot(x, y1, 'red')  
ax.plot(x, y2, 'blue')  
ax.set_title('Quadratic and Cubic Powers')  
ax.set_xlabel('x values')  
ax.set_ylabel('y values');
```



Visualisering med Pandas

- Kräver mindre kod än Matplotlib
- Använder Matplotlib under skalet

```
In [1]: import pandas as pd  
import matplotlib.pyplot as plt  
%matplotlib inline
```



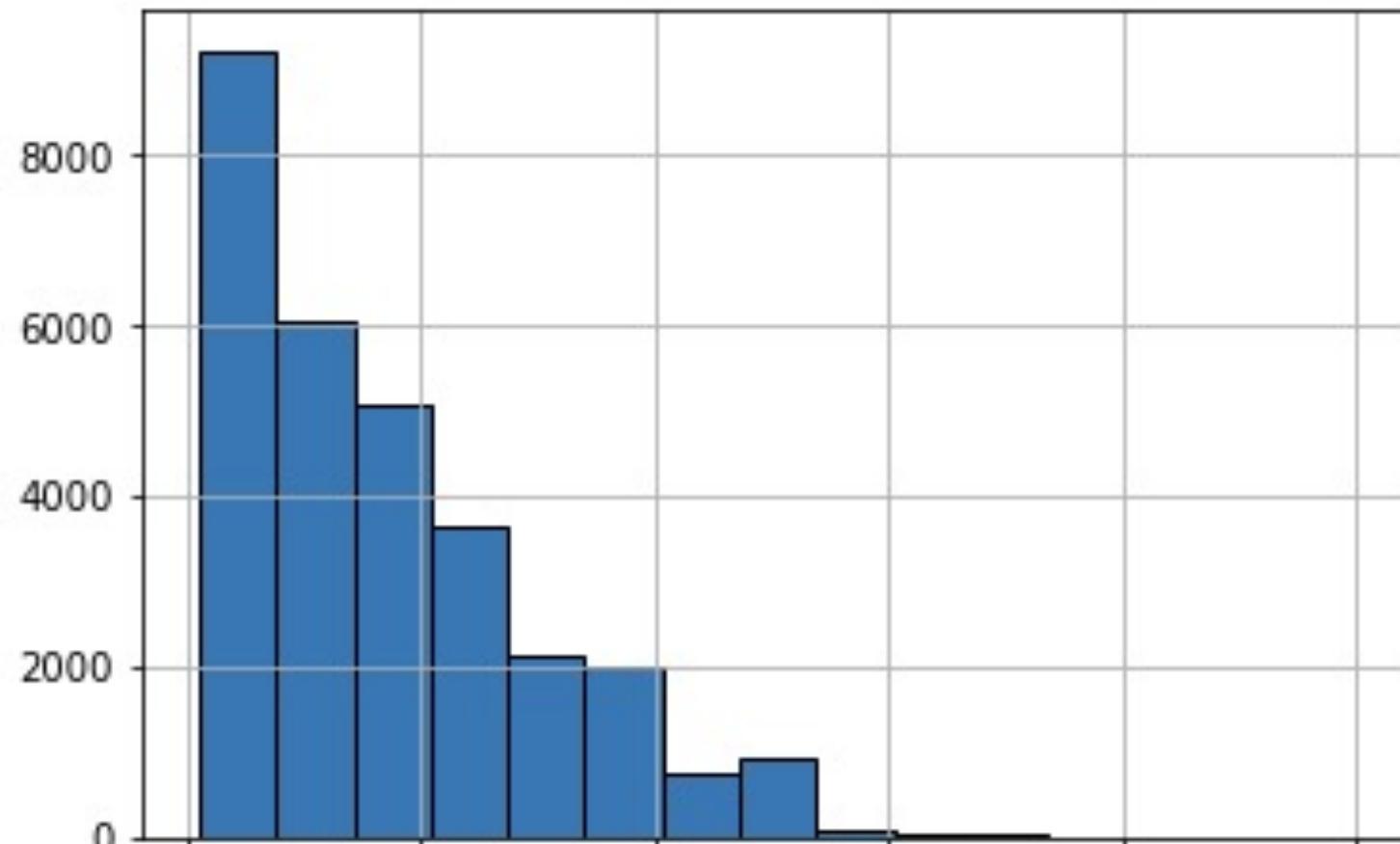
```
In [2]: default = pd.read_csv('../data/credit_card_default.csv', index_col="ID")
```



```
In [3]: default['LIMIT_BAL SEX EDUCATION MARRIAGE AGE PAY_0 PAY_2 PAY_3 PAY_4'.split()+'default payment next month'].head()
```

ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_0	PAY_2	PAY_3	PAY_4	default payment next month
1	20000	2	2	1	24	2	2	-1	-1	1
2	120000	2	2	2	26	-1	2	0	0	1
3	90000	2	2	2	34	0	0	0	0	0
4	50000	2	2	1	37	0	0	0	0	0
5	50000	1	2	1	57	-1	0	-1	0	0
6	50000	1	1	2	37	0	0	0	0	0
7	500000	1	1	2	29	0	0	0	0	0
8	100000	2	2	2	23	0	-1	-1	0	0
9	140000	2	3	1	28	0	0	2	0	0
10	20000	1	3	2	35	-2	-2	-2	-2	0
11	200000	2	3	2	34	0	0	2	0	0
12	260000	2	1	2	51	-1	-1	-1	-1	0
13	630000	2	2	2	41	-1	0	-1	-1	0
14	70000	1	2	2	30	1	2	2	0	1
15	250000	1	1	2	29	0	0	0	0	0
16	50000	2	3	3	23	1	2	0	0	0
17	20000	1	1	2	24	0	0	2	2	1
18	320000	1	1	1	49	0	0	0	-1	0
19	360000	2	1	1	49	1	-2	-2	-2	0
20	180000	2	1	2	29	1	-2	-2	-2	0

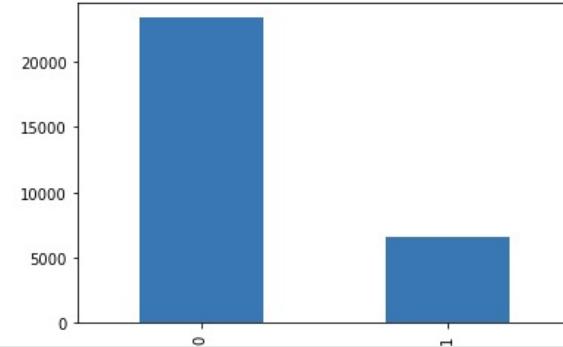
```
In [5]: default['LIMIT_BAL'].hist(edgecolor='black', bins=15);
```



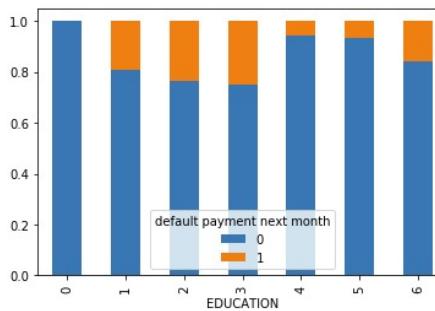
Histogram

Barplots

```
In [7]: default['default payment next month'].value_counts().plot(kind='bar');
```

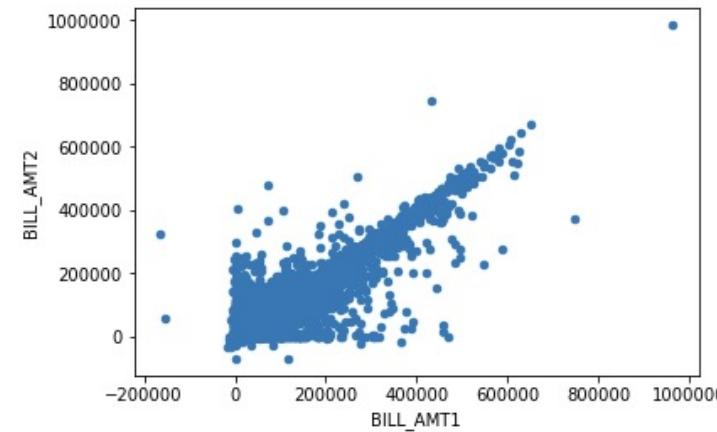


```
In [8]: default.groupby(['EDUCATION'])['default payment next month'].value_counts(normalize=True)\n.unstack().plot(kind='bar', stacked=True);
```

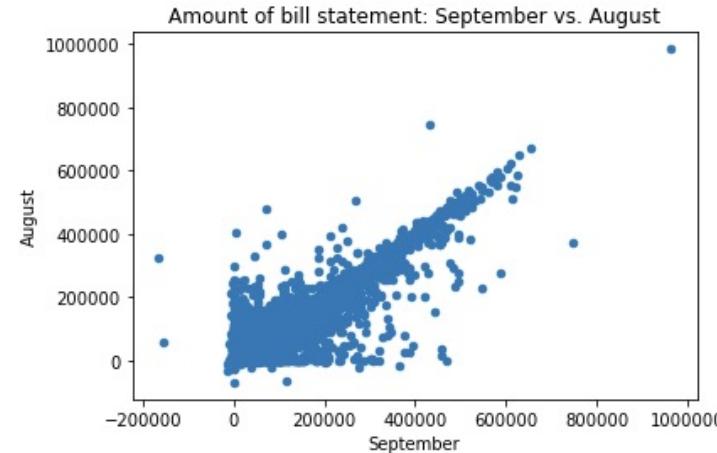


Scatter plots

```
In [9]: default.plot.scatter(x='BILL_AMT1', y='BILL_AMT2');
```



```
In [10]: fig, ax = plt.subplots()  
default.plot.scatter(x='BILL_AMT1', y='BILL_AMT2', ax=ax);  
ax.set_title("Amount of bill statement: September vs. August")  
ax.set_xlabel("September")  
ax.set_ylabel("August");
```



Exempelkod



Med pandas samt
matplotlib ihop
med pandas

Pandas.ipynb och
VisualizationsWith
Pandas.ipynb

Statistisk visualisering med Seaborn

- Ger ett hög-nivå gränssnitt för att rita attraktiv statistisk grafik
- Bygger på Matplotlib och är tight integrerad med Python's Data Science stack
- Support för NumPy och Pandas data strukturer och statistika rutiner från SciPy och statsmodels



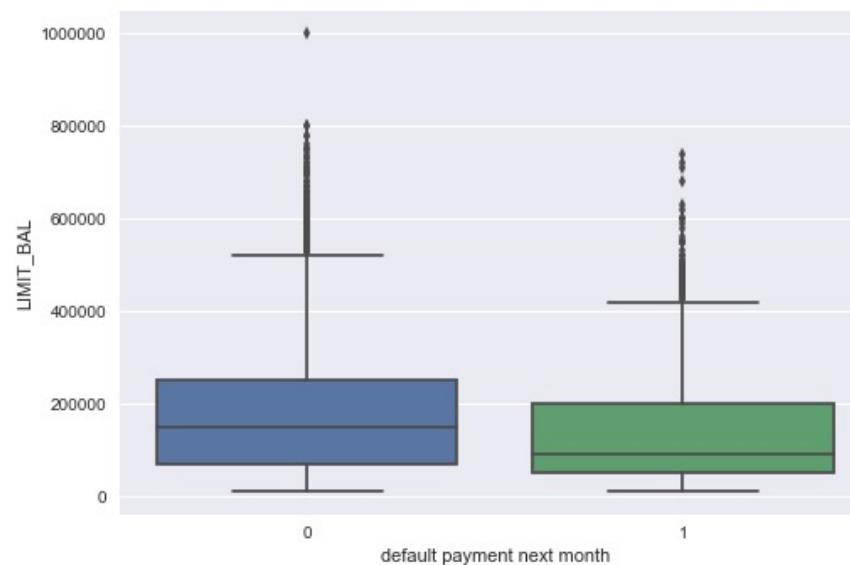
seaborn

```
In [1]: import pandas as pd  
import matplotlib.pyplot as plt  
%matplotlib inline
```

```
In [2]: import seaborn as sns
```

Boxplots

```
In [4]: # Boxplots to compare distributions and look at possible outliers  
sns.boxplot(x='default payment next month', y='LIMIT_BAL', data=default);
```



```
In [3]: default = pd.read_csv('../data/credit_card_default.csv', index_col="ID")
```

Factor plots

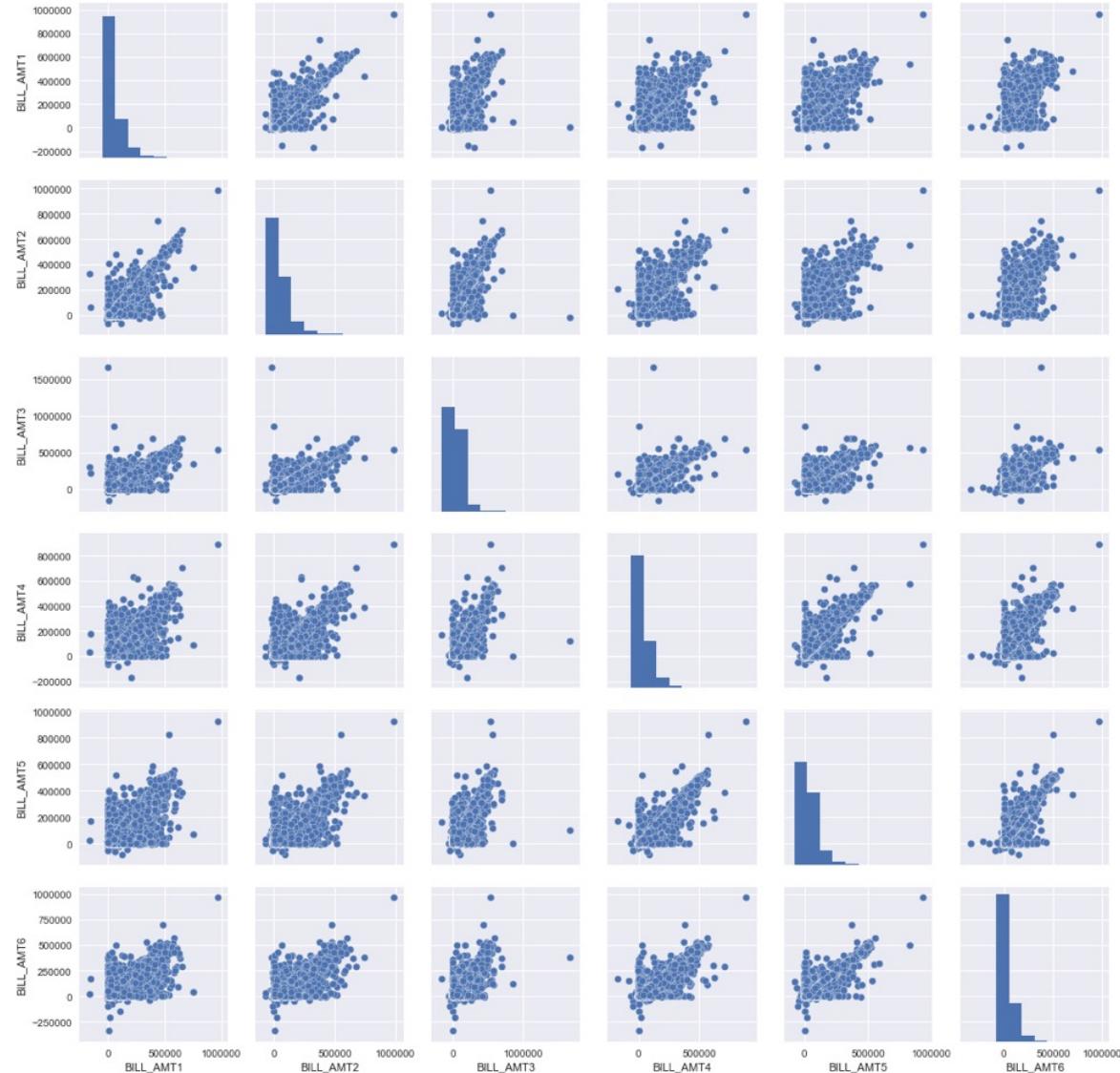
```
In [5]: # Factor plots to visualize differences in means  
sns.factorplot(x='default payment next month', y='LIMIT_BAL', data=default, aspect=2);
```



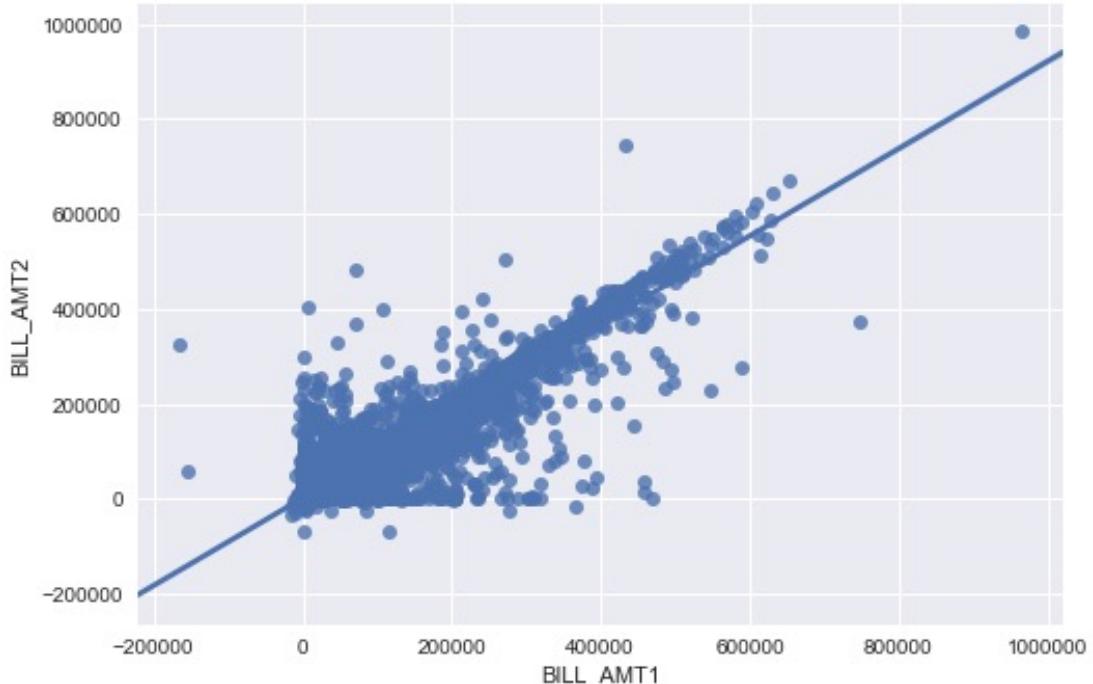
Factor plots forts.

```
In [6]: bill_amount_vars = default[['BILL_AMT'+str(i) for i in range(1,7)]]
```

```
In [7]: sns.pairplot(bill_amount_vars);
```



```
In [8]: sns.regplot(x='BILL_AMT1', y='BILL_AMT2', data=default);
```



Exempelkod

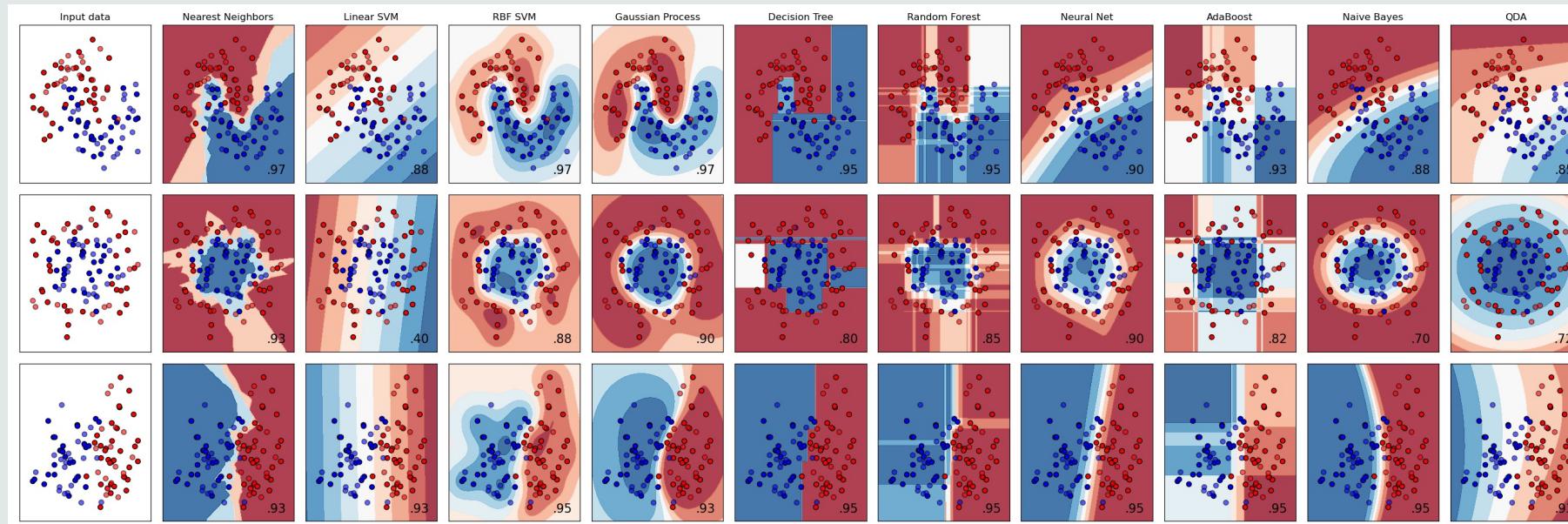
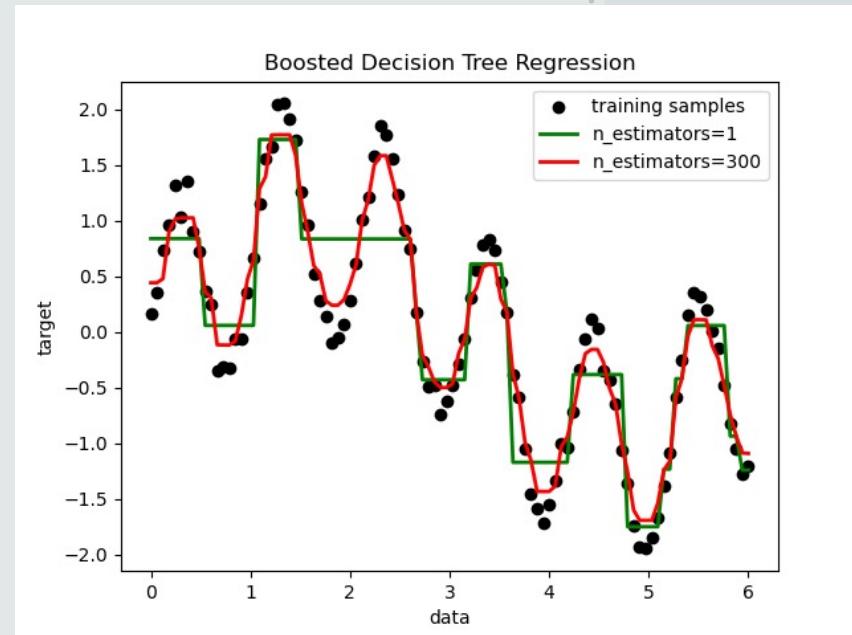


Scikit-learn

- Populärt Machine learning i Python.
- Enkla och effektiva verktyg för att skapa prediktiv analys modeller
- Bygd på NumPy, SciPy och matplotlib
- Skapa modeller för regression, klassificering, clustering, reduktion av dimensioner, model selection och preprocessing



Scikit-learn

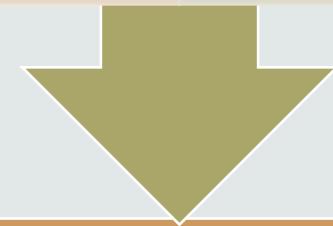


Övning

I Lecture03 finns notebooks:

Numpy.ipynb

Matplotlib.ipynb



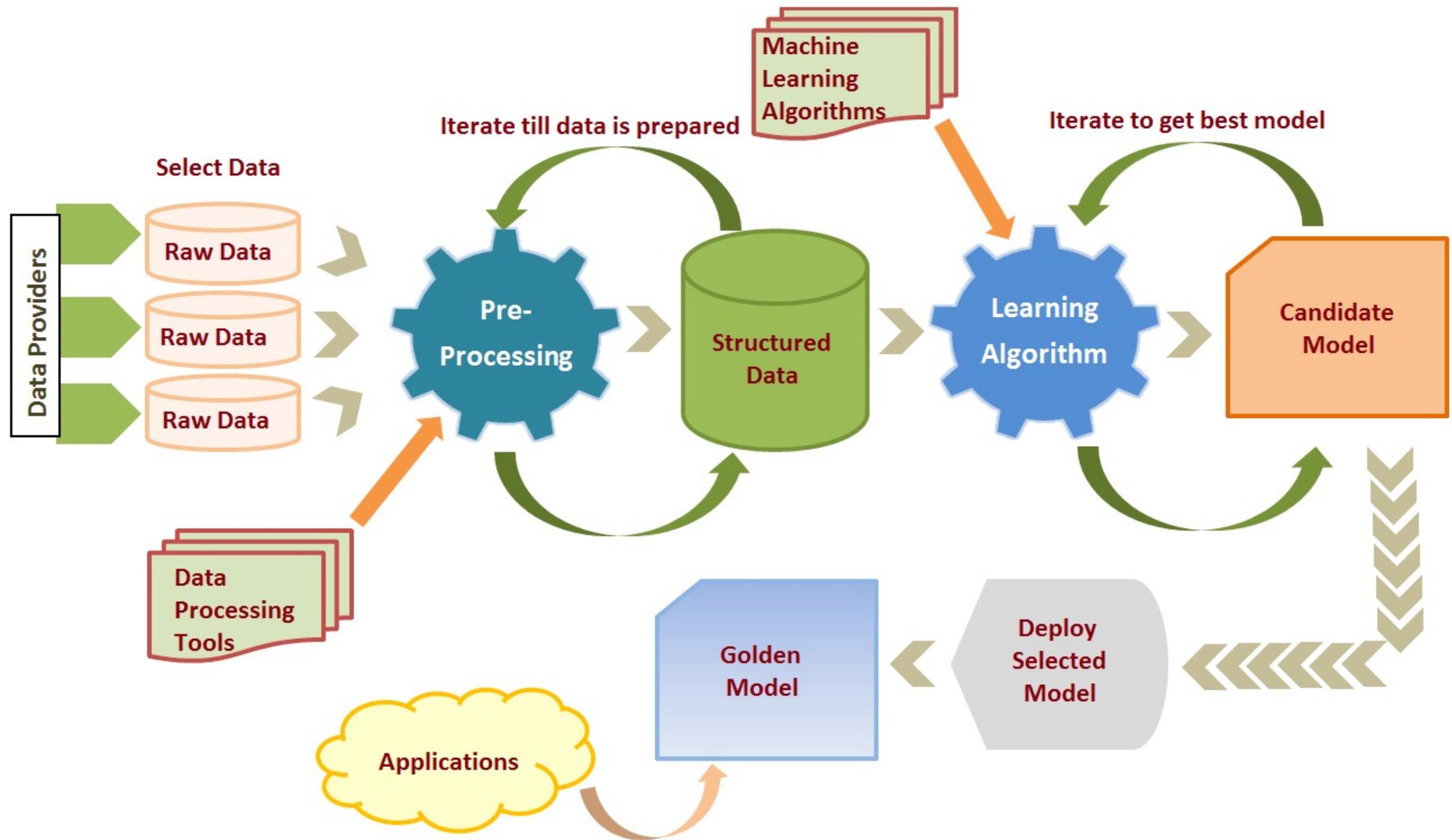
Gå igenom dessa

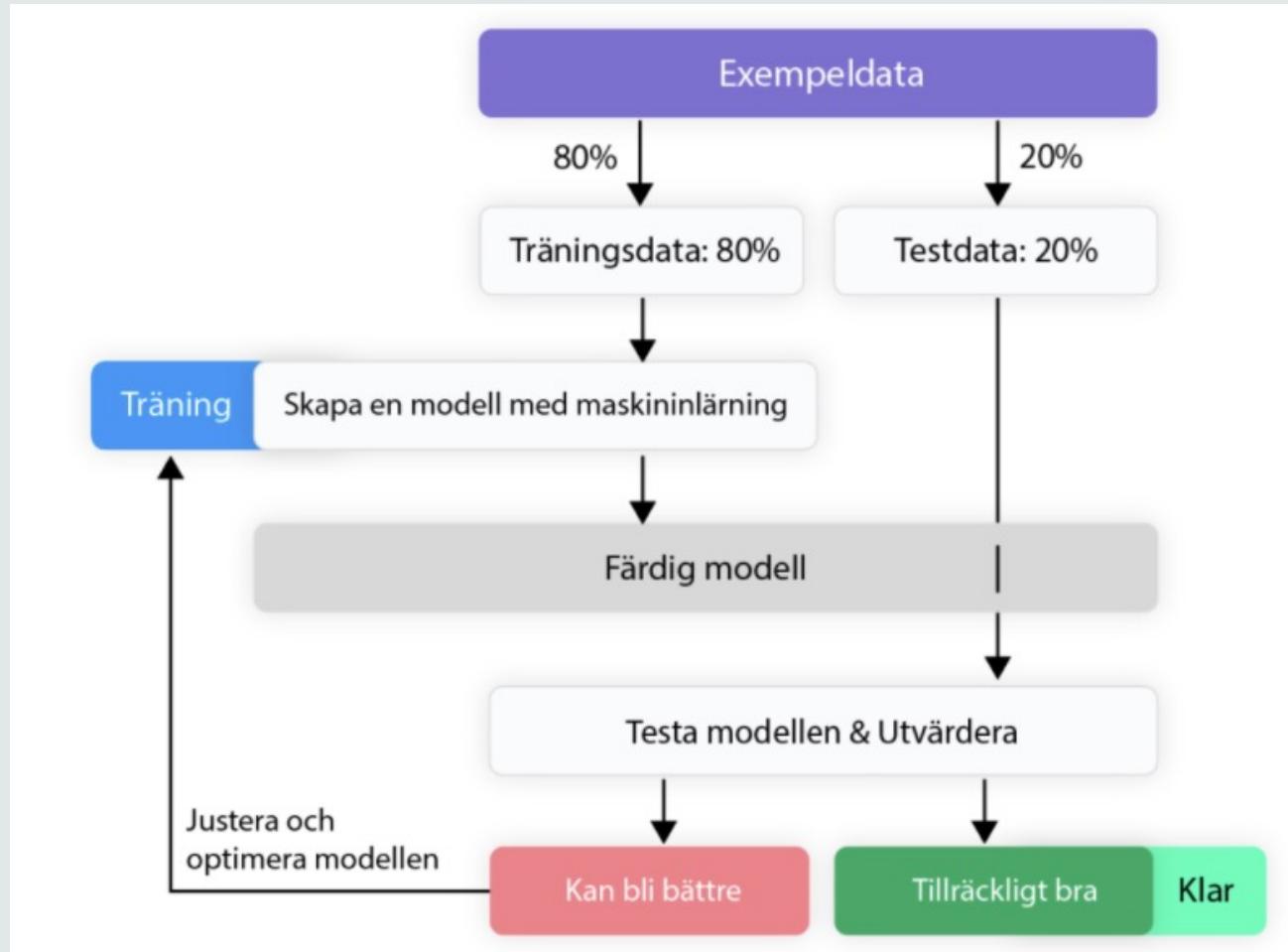
Övning

- Ni ska använda iris blomman datasetet, iris.csv. Finns i data i Lecture03. Hitta först information on datasetet (kaggle) och skriva in det i början av notebooken. Ni ska utforska datan med boxplot, histogram och scatter plot. Vad lär ni er om datasetet när ni kollar på alla dessa plottar? Anteckna i notebook med markdown.
- Vill ni träna mer på Numpy? <https://www.machinelearningplus.com/python/101-numpy-exercises-python/>
- Vill ni träna mer på Pandas? <https://www.machinelearningplus.com/python/101-pandas-exercises-python/>



Prediktiv analys i Python





Vi följer specifika steg för att bygga en Machine learning modell



Detta är generella steg och man kan få hoppa fram och tillbaka flera gånger innan det blir rätt.

01. Förbereda datan (Kriminalitet)

- Importera data
- Se till att dataframen är på korrekt format
- Hantera NaN värden

```
# Import data
crime = pd.read_csv('./data/crime.csv', na_values='?', index_col='communityname')
```

communityname	state	county	community	fold	population	householdsize	agePct12t21
Lakewoodcity	8	NaN	NaN	1	0.19	0.33	0.34
Tukwilacity	53	NaN	NaN	1	0.00	0.16	0.26
Aberdeentown	24	NaN	NaN	1	0.00	0.42	0.39
Willingborotownship	34	5.0	81440.0	1	0.04	0.77	0.51
Bethlehemtownship	42	95.0	6096.0	1	0.01	0.55	0.38

01. Förbereda datan (forts.)

- Skapar två objekt av datan:
 - x (egenskaper, features)
 - y (mål, target)

```
feature_names = ['households', 'pctUrban', 'medIncome', 'PctKids2Par', 'PctIlleg', 'PctFam2Par']
target_name = 'ViolentCrimesPerPop'
```

✓ 0.7s

```
# Getting the features and the target, sklearn can work directly with pandas dataframes
x = crime[feature_names]
y = crime[target_name]
```

✓ 0.4s

01. Förbereda datan (forts.)

- Dela upp datan i träningsdata och testningsdata

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=123)
✓ 0.3s
```

02. Importera estimeringsobjektet (modellen)

```
from sklearn.linear_model import LinearRegression
```

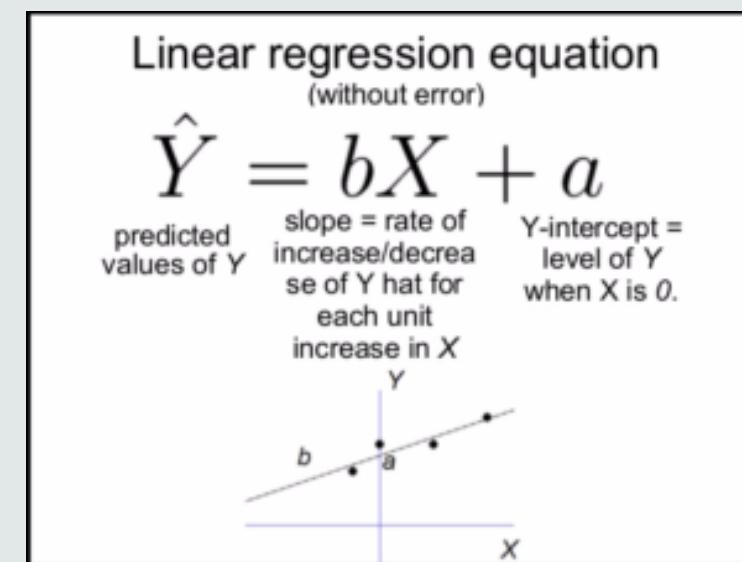
✓ 0.3s

03. Skapa en instans av modellen

- Här kan vi skicka in extra värden vid behov.
- Vi använder bara standardvärden
- Refereras till som "hyper parameters"

```
linear_regression_model = LinearRegression() # Pass any parameters
```

✓ 0.3s



04. Träna modellen med träningsdata

- Så här skickar vi in träningsdata
- x (egenskaper, features)
- y (mål, target)

"fit" används alltid



```
linear_regression_model.fit(x_train, y_train)
```

✓ 0.3s

Nu är modellen tränad

05. Utvärdera modellen

- Hämta testningsdata som modellen hittills inte har sett

```
from sklearn.metrics import mean_squared_error
# Get the predictions of the model for the data it has not seen (testing)
y_pred_test = linear_regression_model.predict(x_test)
# All the metrics compare in some way how close are the predicted vs. the actual values
error_metric = mean_squared_error(y_pred=y_pred_test, y_true=y_test)
print('The Mean Square Error of this model is: ', error_metric)
```

✓ 0.3s

Evaluating metric

Mean squared error

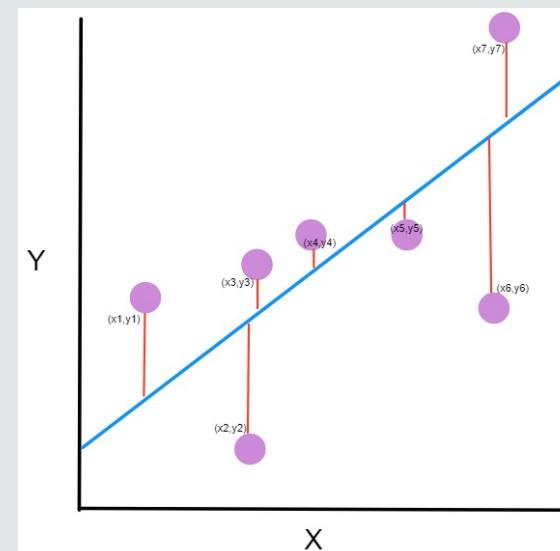
$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

MSE = mean squared error

n = number of data points

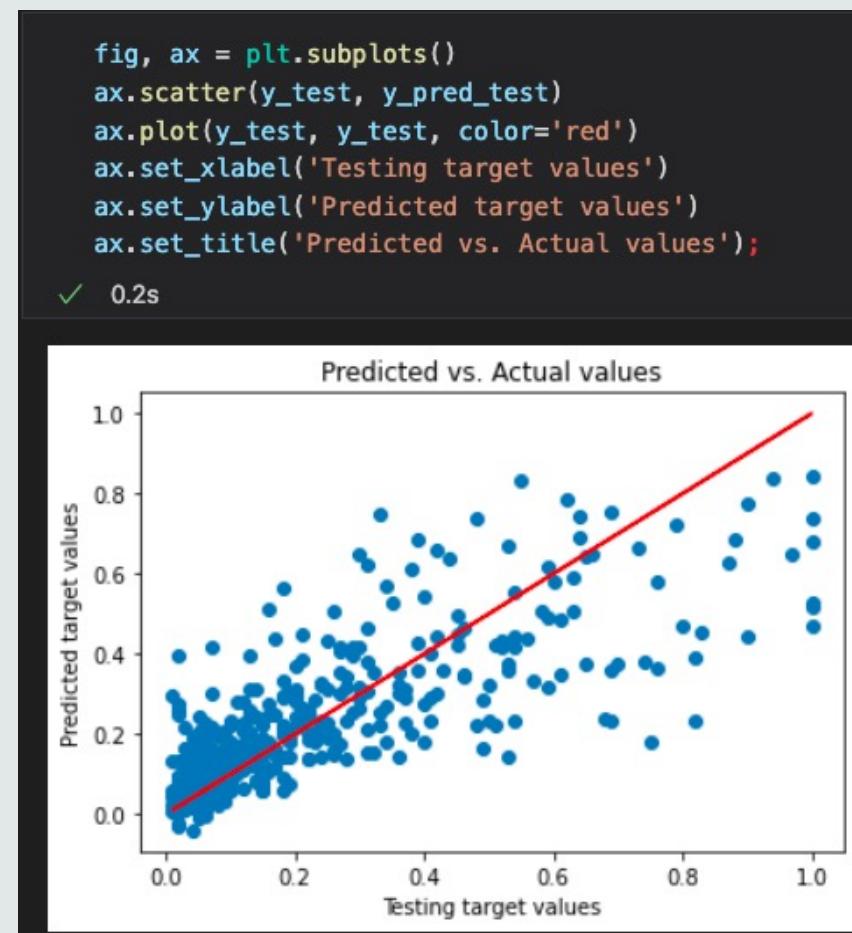
Y_i = observed values

\hat{Y}_i = predicted values



05. Utvärdera modellen (forts)

- Visualisera hur nära predikteringen är de verkliga värdena
- Man kan se att några av värden är lite långt bort
- Om vi är nöjda med modellen så går vi vidare till steg 06



06. Gör predikteringar

- In [12] objektet används för att göra predikteringen
- ”new_data” är ett nytt område/samhälle som vi vill använda vår modell på för att beräkna kriminalitet för.

```
new_data = OrderedDict([
    ('households', 0.5),
    ('pctUrban', 0.6),
    ('medIncome', 0.1),
    ('PctKids2Par', 0.5),
    ('PctIlleg', 0.2),
    ('PctFam2Par', 0.3)])
# .values.reshape(1, -1) because it must be 2-dim, because we passed only one new observation
new_data = pd.Series(new_data).values.reshape(1,-1)
# Use the model to make predictions
linear_regression_model.predict(new_data)
✓ 0.6s
```

För ett samhälle med karaktäristiken ovan så säger vår modell att mål värdet (target value) är 0.17574784

Vad har vi gjort idag?

- Viktiga python bibliotek för prediktiv analys:
 - Numpy
 - Pandas
 - Matplotlib
 - Seaborn
 - Scikit-learn
- Prediktiv analys i python: viktigaste stegen
 1. Förbereda data
 2. Importera estimeringsobjektet
 3. Skapa instans av modellen
 4. Träna modellen
 5. Utvärdera modellen
 6. Prediktera

Nästa lektion

- Regression
- The multiple regression model
- K-nearest neighbor
- Lasso regression
- Model evaluation for regression