<k/>

PANDAS AND MACHINE LEARNING

From a CSV to a first model

THEKHANSA.COM

LOAD AND LOOK

```
import pandas as pd
df = pd.read_csv("data.csv")
df.head()   df.tail()
df.shape   df.columns
df.info()   df.describe()
df["col"].value_counts()
```

SELECT AND FILTER

```
df["a"]          df[["a", "b"]]
df.loc[rows, "a"]   # by label
df.iloc[0:5, 0:2]    # by position
df[df["age"] > 30]
df[(df.a > 1) & (df.b == "x")]
df.query("a > 1 and b == 'x'")
```

CLEAN

```
df.isna().sum()
df.dropna()   df.fillna(0)
df.drop_duplicates()
df["a"] = df["a"].astype(int)
df["d"] = pd.to_datetime(df["d"])
df.rename(columns={"old": "new"})
```

GROUP AND COMBINE

```
df.groupby("city")["sales"].sum()
df.groupby("city").agg(
    total=("sales", "sum"),
    avg=("sales", "mean"))
pd.merge(a, b, on="id", how="left")
pd.concat([a, b])
df.pivot_table(index="a", columns="b",
    values="v", aggfunc="sum")
```

SORT, NEW COLUMNS, SAVE

```
df.sort_values("a", ascending=False)
df["total"] = df["qty"] * df["price"]
df["x"] = df["a"].apply(func)
df.to_csv("out.csv", index=False)
```

PREPARE FOR A MODEL

```
from sklearn.model_selection import train_test_split
X = df.drop(columns="target")
y = df["target"]
X_tr, X_te, y_tr, y_te = train_test_split(
    X, y, test_size=0.2, random_state=42)
pd.get_dummies(X)       # text to numbers
```

TRAIN AND PREDICT

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(random_state=42)
model.fit(X_tr, y_tr)
pred = model.predict(X_te)
# regression: LinearRegression, RandomForestRegressor
```

CHECK THE RESULT

```
from sklearn.metrics import (
    accuracy_score, classification_report,
    mean_squared_error, r2_score)
accuracy_score(y_te, pred)
classification_report(y_te, pred)
from sklearn.model_selection import cross_val_score
cross_val_score(model, X, y, cv=5)
```

WHICH METRIC?

```
Balanced classes:  accuracy
Rare positives:  precision, recall, F1
Regression:  RMSE (same unit as target), R squared
Time series:  split by time, never shuffle
```

Good to know: Fit scalers and encoders on the training set only, then apply them to the test set. If the training score is far better than the test score, the model is overfitting.