

ESTE CURSO ES PARTE DE UNA ESPECIALIZACION DE
“Aprendizaje automático avanzado”

- 1-Introduction to Deep Learning
- 2-How to Win a Data Science Competition:
Learn from Top Kagglers
- 3-Bayesian Methods for Machine Learning
- 4-Practical Reinforcement Learning
- 5-Deep Learning in Computer Vision
- 6-Procesamiento de lenguajes naturales
- 7-Addressing Large Hadron Collider Challenges
by Machine

2-How to Win a Data Science Competition: Learn from Top Kagglers

(SEMANA 1)

CONCEPTOS PRINCIPALES EN UNA COMPETENCIA

1-DATA

2-MODEL

3-SUBMISSION

4-EVALUATION

5-LEADERBOARD

4-EVALUATION FUNCTION

ACCURACY

LOGISTIC LOSS

AUC

RMSE

MAE

5.1 PUBLIC TEST USING DURING COMPETITION

5.2 PRIVATE TEST USED FOR FINAL RANK.

OTROS SITIOS DE COMPETENCIAS

KAGGLE

DRIVENDATA

CROWDANALITYX

CODALAB

DATASCIENCECHALLENGE.NET

SINGLE-COMPETITION SITES (LIKE KDD, VIZDOOM)

OTRO CURSO IMPORTANTE

COMPETITIVE DATA SCIENCE - COURSERA

INGRESANDO A KAGGLE

COMO CREAR UNA CUENTA CARACTERISTICAS PRINCIPALES DEPENDE DE CADA CONCURSO

EJEMPLOS

COMO ESTÁ ORGANIZADO EL SITIO OPCIONES.
OVERVIEW, FORO, CODE ,ETC

ALGUNOS DATOS CLAVES
5 PERSONAS POR EQUIPO
NO SE PERMITEN DATOS EXTERNOS
ENTORNO KAGGLE COMO FUNCIONA
COMO COMPARTIR Y SUBIR UN ARCHIVO
DISPONIBILIDAD DE DATOS
TIEMPOS DE USOS.

INGRESANDO A KAGGLE

COMO CREAR UNA CUENTA CARACTERISTICAS PRINCIPALES DEPENDE DE CADA CONCURSO

REAL WORLD VS. COMPETITIONS

REAL WORLD

It's a complicated process included:

- Understanding of business problem
- Problem formalization
- Data collecting
- Data preprocessing
- Modelling
- Way to evaluate model in real life
- Way to deploy model

Aspect	Real Life	Competition
Problem formalization	Y	N
Choice of target metric	Y	N
Deployment issues	Y	N
Inference speed	Y	N
Data collecting	Y	N/Y
Model complexity	Y	N/Y
Target metric value	Y	Y

FAMILIES OF ML ALGORITHM

LINEAR MODEL

EXAMPLES LOGISTIC REGRESSION / SUPPORT VECTOR MACHINE

TREE-BASED: DECISION TREE, RANDOM FOREST, GBDT

KNN

NEURONAL NETWORKS

HARDWARE / SOFTWARE

Most of competitions (expect image-based) can be solved on:

- High-level laptop
- 16+ gb ram
- 4+ cores

Quite good setup:

- Tower PC
- 32+ gb ram
- 6+ cores

EQUIPO

RAM 64G O 128G

32 CORES

SSD

CLOUD RESOURCES

AMAZON AWS (*) / MICROSOFT AZURE / GOOGLE CLOUD

LENGUAJES

PYTHON / R.

RESOURCES

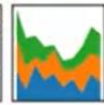
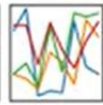
Most of competitors use Python data science software stack.



NumPy

pandas

$$y_{it} = \beta^T x_{it} + \mu_i + \epsilon_{it}$$



matplotlib

IP[y]:
IPython



dmlc
XGBoost

Microsoft / LightGBM

K Keras

danielfrg / tsne
forked from osdf/py_bh_tsne



VOWPAL WABBIT

srendle / libfm

guestwalk / libffm

baidu / fast_rgf

Conclusion

1. Numeric feature preprocessing is different for tree and non-tree models:
 - a. Tree-based models doesn't depend on scaling
 - b. Non-tree-based models hugely depend on scaling
2. Most often used preprocessings are:
 - a. MinMaxScaler - to $[0,1]$
 - b. StandardScaler - to $\text{mean}=0, \text{std}=1$
 - c. Rank - sets spaces between sorted values to be equal
 - d. $\text{np.log}(1+x)$ and $\text{np.sqrt}(1+x)$

1. Scaling and Rank for numeric features:
 - a. Tree-based models doesn't depend on them
 - b. Non-tree-based models hugely depend on them
2. Most often used preprocessings are:
 - a. MinMaxScaler - to $[0,1]$
 - b. StandardScaler - to $\text{mean}=0, \text{std}=1$
 - c. Rank - sets spaces between sorted values to be equal
 - d. $\text{np.log}(1+x)$ and $\text{np.sqrt}(1+x)$
3. Feature generation is powered by:
 - a. Prior knowledge
 - b. Exploratory data analysis

1. Values in ordinal features are sorted in some meaningful order
2. Label encoding maps categories to numbers
3. Frequency encoding maps categories to their frequencies
4. Label and Frequency encodings are often used for tree-based models
5. One-hot encoding is often used for non-tree-based models
6. Interactions of categorical features can help linear models and KNN

Datetime

- a. Periodicity
- b. Time since row-independent/dependent event
- c. Difference between dates

Coordinates

- a. Interesting places from train/test data or additional data
- b. Centers of clusters
- c. Aggregated statistics



Treating values which do not present in train data

1. The choice of method to fill NaN depends on the situation
2. Usual way to deal with missing values is to replace them with -999, mean or median
3. Missing values already can be replaced with something by organizers
4. Binary feature "isnull" can be beneficial
5. In general, avoid filling nans before feature generation
6. Xgboost can handle NaN

CONCLUSION **Bag of words**

Pipeline of applying BOW

1. Preprocessing:

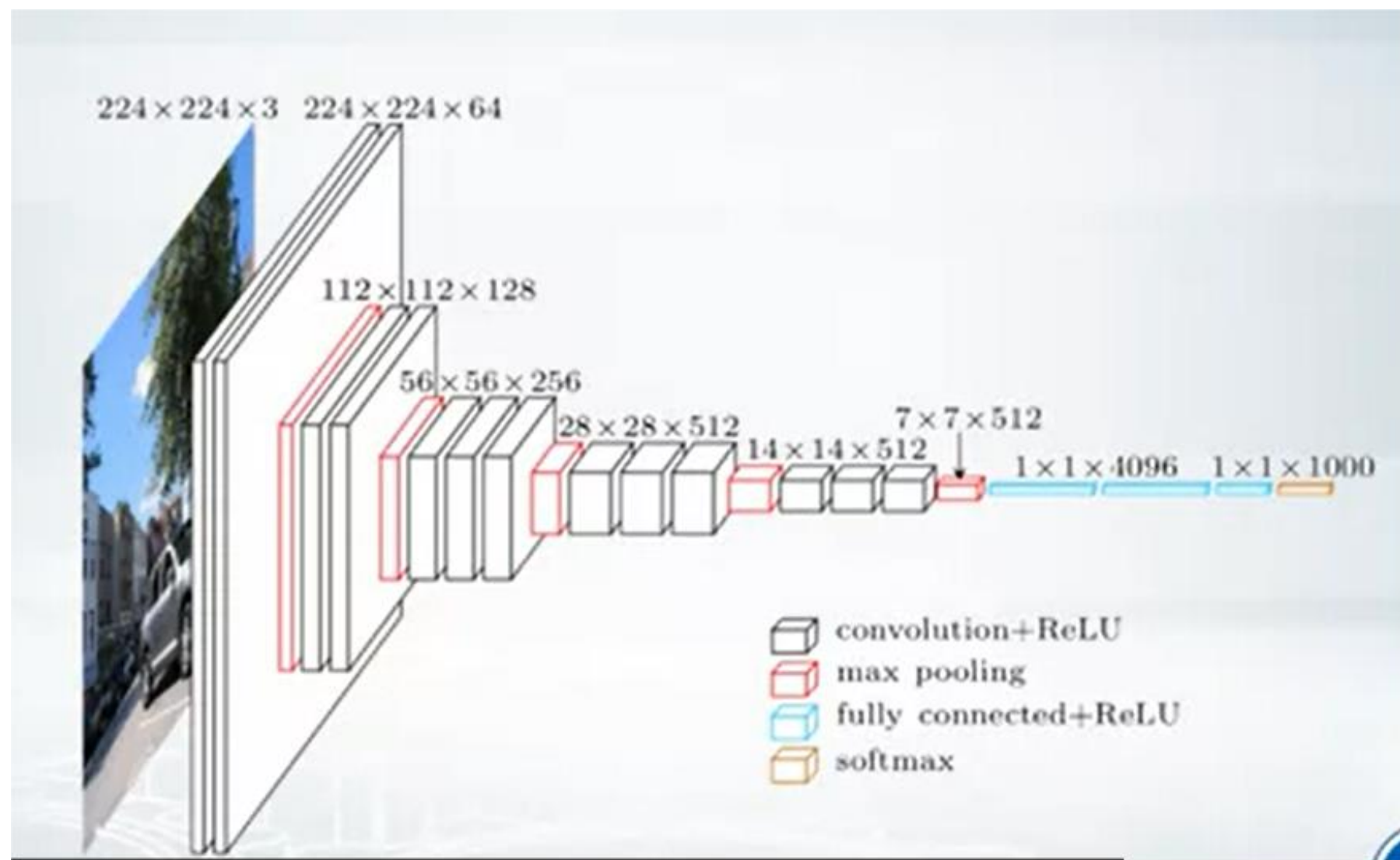
Lowercase, stemming, lemmatization, stopwords

2. Ngrams can help to use local context

3. Postprocessing: TFiDF

CONCLUSION **Word2vec**, CNN

1. Bag of words
 - a. Very large vectors
 - b. Meaning of each value in vector is known
2. Word2vec
 - a. Relatively small vectors
 - b. Values in vector can be interpreted only in some cases
 - c. The words with similar meaning often have similar embeddings



1. Texts

a. Preprocessing

- i. Lowercase, stemming, lemmatization, stopwords

b. Bag of words

- i. Huge vectors
- ii. Ngrams can help to use local context
- iii. TFIDF can be of use as postprocessing

c. Word2vec

- i. Relatively small vectors
- ii. Pretrained models

2. Images

- a. Features can be extracted from different layers
- b. Careful choosing of pretrained network can help
- c. Finetuning allows to refine pretrained models
- d. Data augmentation can improve the model

AL FINAL HAY UNA PARTICIIPACION A UNA COMPETENCIA EN KAGGLE
PERO PARA EL CURSO, PARECE INTERESANTE ES PARA APROBAR LA ESPECIALIZACIÓN

SI QUIEREN LOS CUESTIONARIOS LOS HACEMOS EN LA REUNIÓN EN UN RATITO
DE LA SEMANA 1 y hay un ejercicio básico con pandas.