Universitat Politècnica de Catalunya

Grau en Intel·ligència Artificial

PERCEPTRÓ MULTICAPA

Pràctica1 XNDL

Autors:

Jaume Mora Ladària, **Marta Nadal Par**

maig 2024

Índex

1 Introducció

La metodologia seguida en aquest treball consta d'una anàlisi exploratòria de les dades, després dissenyarem una estratègia de preprocessament, a més avaluarem un model lineal base de regressió. Finalment, ens enfocarem en millorar aquest model utilitzant una xarxa neuronal de perceptró multicapa (MLP), on iterativament diagnosticarem el model i proposarem millores.

Hem seleccionat la base de dades '*rain_data*', ja que és un conjunt de dades complex i amb diversitat, que ens permetrà aplicar diferents tècniques de processament. '*rain_data*' té més de 20 variables (combinant variables numèriques i categòriques) i més de 60.000 mostres, cosa que ens proporciona molta informació útil per realitzar una anàlisi exhaustiva i construir models robustos.

La variable que volem predir és la variable binària 'RainTomorrow', que indica si plourà o no demà. Creiem que aquesta és la variable més interessant de predir, ja que és una variable clau que pot tenir un impacte significatiu en diverses decisions quotidianes i professionals.

2 Anàlisi Exploratoria de Dades (EDA)

2.1 Estudi estadístic i visualització de les dades

Si fem una anàlisi estadística a partir dels gràfics podem observar com les temperatures mínimes i màximes es distribueixen de manera relativament normal. Per contra, les dades de precipitació mostren un clar esbiaixament cap a la dreta, amb la majoria de dies registrant poca o cap pluja i uns pocs registrant valors alts.

Les dades sobre la velocitat del vent i la pressió atmosfèrica també presenten patrons interessants. Per una banda, la velocitat del vent tendeix a valors més baixos. D'altra banda, la pressió atmosfèrica mostra una distribució més uniforme i estreta, típica d'una variable amb poca variabilitat diària.

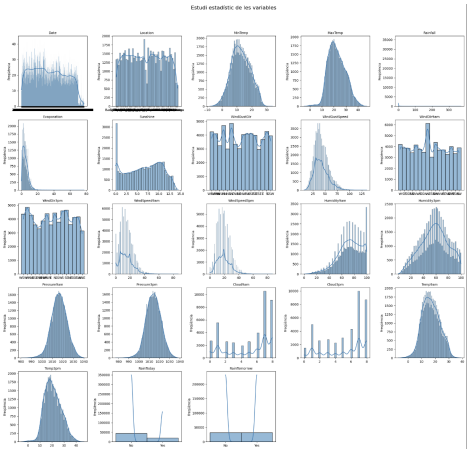

Figura 1: Estudi estadístic de les dades

2.2 Correlacions entre variables

Com podem observar amb la matriu de correlació de les variables numèriques, tenim diverses relacions interessants entre variables. Una de les correlacions més evidents és entre les temperatures màximes i mínimes (0.74). Com és evident, els dies més càlids tenen tant la màxima i la mínima més altres i al contrari pels dies freds. Aquest patró es repeteix per les temperatures registrades a les 9 del matí i les 3 de la tarda (amb una correlació altíssima del 0.86).

El mateix passa amb la pressió atmosfèrica que, tant a les 9 del matí com a les 3 de la tarda, mostra una correlació gairebé perfecta de 0.96, reflectint poca variabilitat diària en aquesta mesura.

Finalment veiem altres correlacions altes però menys significatives com les de la humitat o el vent per exemple.

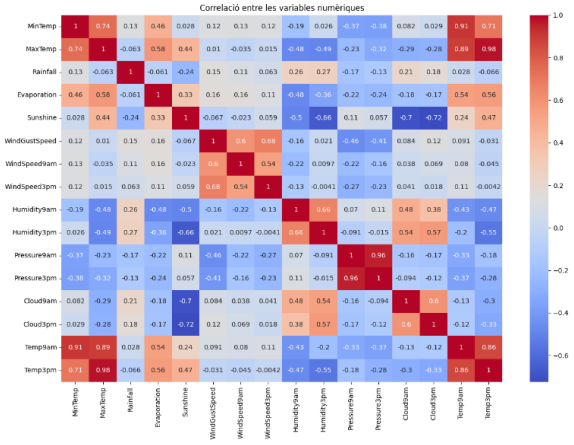

Figura 2: Correlació entre variables numèriques

3 Preprocessament

Un bon preprocessament és clau per tenir un bon funcionament dels models a entrenar més endavant. Per això, hem optat per dedicar-hi molt temps i per fer un preprocessament gairebé personalitzat per a cada una de les variables. A continuació oferim una descripció per a cada una de les variables preprocessades sobre la tècnica utilitzada.

3.1 Tractament variable a variable

- Variable 'Date': Per la variable 'Date' l'hem convertit en 'Season', és a dir, hem dividit les dades en quatre grups diferents depenent de l'estació de l'any (primavera, estiu, tardor o hivern), a cada estació li hem assignat un número (hivern 0, primavera 0.33, estiu 0.66, tardor 1), d'aquesta manera tenim una variable numèrica però conservant la informació categòrica. Creiem que és important aquest canvi, ja que l'estació de l'any pot tenir un efecte significatiu en els patrons climàtics i, per tant, en la quantitat de pluja registrada.

- Variables 'RainToday' i 'RainTomorrow': En el cas de les variables binàries 'RainToday' i 'RainTomorrow', hem convertit la classe 'Yes' en 1 i 'No' en 0. A més hem optat per eliminar les files que tenien valors mancants en 'RainTomorrow', ja que com aquesta base de dades conté una gran quantitat de dades, preferim no haver d'imputar els missings de la que serà la variable objectiu.

- Variable 'Location': Per la modificació de la variable 'Location' hem agrupat les ciutats en estats d'Austràlia. Així hem passat de tenir 49 ciutats a 8 estats i hem pogut reduir el cost del OneHotEncoding que hem aplicat després.

- Variables de direcció: Per les variables categòriques WindGustDir, WindDir9am i WindDir3pm hem decidit separar cada una d'elles en dues variables noves, Nord-Sud i Est-Oest, i en aquestes dues noves variables hem assignat valors diferents (1, 0.5 i 0) per representar la direcció del vent. D'aquesta manera simplifiquem i estructurem millor les dades, convertint la informació en un format més fàcil d'analitzar i manejar.

- Variable 'Rainfall': En el cas de la variable 'Rainfall' obervant la seva distribució en la imatge següent, veiem que no ens aporta gaire informació útil pel nostre estudi, per tant decidim eliminar la variable.

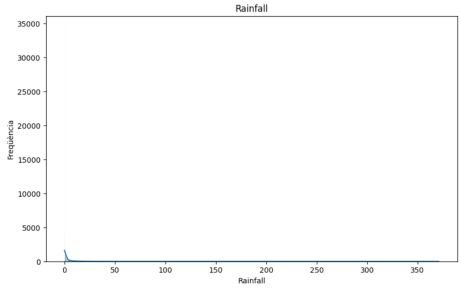

Figura 3: Gràfic 'Rainfall'

3.2 Tractament d'outliers i gestió de missings:

Abans de tractar els outilers, observem que algunes variables contenen una gran quantitat de valors buits. Tenint en compte que la nostra base de dades té 60.000 mostres, hem decidit eliminar les variables que tenen més de 25.000 missings, ja que més de la meitat de les seves mostres tindrien missings, la qual cosa no aportaria informació significativa. Per tant, hem decidit eliminar les següents variables: 'Rainfall', 'Evaporation', 'Sunshine', 'Cloud9am', 'Cloud3pm'.

Una vegada eliminades aquestes variables, fem un estudi dels outliers de cada variable, en la següent taula podem observar la quantitat d'outliers de cada variable:

Variable	Valor
MinTemp	40
MaxTemp	290
WindGustSpeed	1167
WindSpeed9am	786
WindSpeed3pm	1422
Humidity9am	722
Humidity3pm	0
Pressure9am	643
Pressure3pm	509
Temp9am	128
Temp3pm	373

Taula 1: Outliers de les variables

Observem que la variable que conté més outliers és WindSpeed3pm, amb 1422 outliers en la nostre base de dades de 60.000 mostres. Això representa menys del 5% de les mostres, la qual cosa considerem una proporció relativament baixa. Tot i així, els outliers poden afectar significativament els nostres resultats, per tat preferim eliminar-los per garantir que les nostres anàlisis estiguin basades en dades més fiables i representatives.

Com podem observar a continuació comparant les distribucions de les variables abans i després de l'eliminació dels outliers, la majoria de les variables no canvien i les que canvien ho fan de tal manera que es normalitzen correctament. Si ens fixem en les dues distribucions amb l'abans i el després veiem com, en general teníem ben pocs outliers. Al tenir tantes dades hem optat per eliminar directament les files amb outliers en comptes de voler-les convertir a missings i finalment imputar-les.

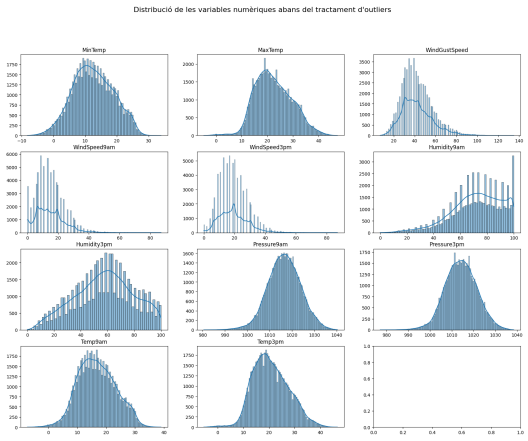

Figura 4: Distribucions amb outliers

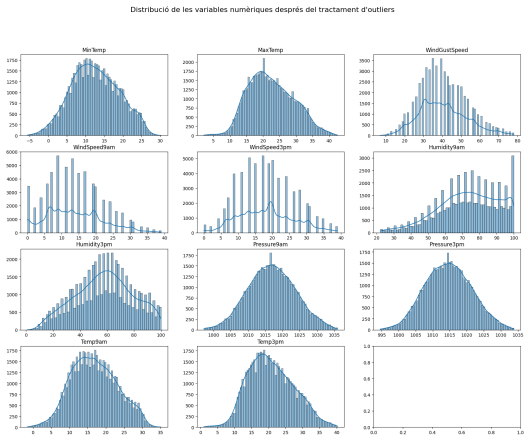

Figura 5: Distribucions sense outliers

Finalment, pel que fa al tractament de missings, a part del que hem comentat anteriorment sobre les variables amb més de 25000 missings, hem pensat que el millor era imputar-los utilitzant un **KNNImputer** amb una K=5, ja que permet estimar els missings basant-se en les 5 observacions més properes, proporcionant una bona aproximació de les característiques locals de les dades. A més a més, era el que ens donava millors resultats.

Un cop hem completat el tractament d'outliers i de missings, hem normalitzat les dades per equilibrar les variables. Per a això, hem utilitzat **MinMaxScaler**, que és una bona opció, ja que manté la forma de la distribució, cosa que és útil per mantenir les proporcions relatives entre els punts de dades. A més, els valors resultants són fàcils d'interpretar, al estar en una escala comprensible.

4 Remostreig

4.1 Divisió train i test

Per fer el particionat del dataset, hem observat que la millor distribució seria dedicar **un 70% de les dades per a l'entrenament i el 30% restant per a la prova**. Aquesta proporció és força habitual i ofereix un bon equilibri entre disposar de prou dades per a l'entrenament del model, i alhora conservar una quantitat substancial per a comprovar la seva eficàcia en dades noves. Aquest particionat s'ha realitzat abans de la imputació dels valors mancants.

4.2 Validació creuada

Per al model de regressió logística hem utilitzat un Cross Validation amb 5 particions per a poder avaluar robustament el rendiment del model. A més a més, per a trobar els millors hiperparàmetres del model també utilitzem un GridSearchCV.

5 Model lineal base

5.1 Regressió

En aquesta secció, hem entrenat i avaluat un model de regressió logística per predir la variable 'RainTomorrow' que indica si plourà o no demà. Hem utilitzat la tècnica de validació creuada amb GridSearchCV per trobar els millors hiperparàmetres per al model. Els hiperparàmetres en la regressió logística són:

- C: Controla la força de regularització (evitar sobreajustament), un valor més gran indica una regularització més feble, mentre que un valor més petit indica una regularització més forta. En aquest cas hem trobat que el millor valor és $\mathbf{C = 100}$, el que suggereix que el model té més flexibilitat per adaptar-se a les dades d'entrenament.

- Penalty: Especifica el tipus de regularització que s'aplica al model. 'l1' (lasso), que penalitza els coeficients amb valors absoluts,'l2' (ridge), que penalitza els quadrats dels coeficients. Hem trobat que pel nostre model funciona millor **penalty = 'l1'**.

- Solver: Especifica l'algorisme utilitzat per optimitzar la funció de cost en el model. Hem trobat que la millor opció és **solver = 'liblinear'** ja que és un solver eficient per problemes binàris.

5.2 Interpretació dels resultats

Per tal d'avaluar el nostre model de regressió logística entrenat amb els millors hiperparàmetres trobats ('C': 100, 'penalty': 'l1', 'solver': 'liblinear'), hem utilitzat la validació creuada, amb les mètriques d'accuracy, precisió i F1-score, ja que ens ofereixen una comprensió completa del rendiment. A la següent taula podem observar els resultats obtinguts:

Mètrica	**Mitjana**
Accuracy en validació creuada	0.77494
Precisió en validació creuada	0.77521
F1-score en validació creuada	0.77479

Taula 2: Resultats de la validació creuada

Els resultats mostren que el model de regressió logística té un rendiment general consistent en el conjunt de train, amb una accuracy, precisió i f1-score al voltant del 77.5%. Això suggereix que el model és capaç de generalitzar adequadament les noves dades.

6 Procés iteratiu - Perceptró Multicapa (MLP)

A continuació presentem els nostres models pels quals fem un diagnòstic i, en cas de que sigui necessari una proposta de millora per a poder comparar si el model següent el millora. Cal especificar que hem fet els models en ordre intentant sempre millorar el model anterior.

Alguns dels hiperparàmetres dels models han sigut fixes per tots els models perquè no hem trobat en cap cas uns hiperparàmetres que ens donessin millors resultats. És el cas de **l'optimitzador Adam**, un **learning_rate = 0.001** i una funció de pèrdua anomenada **sparce categorical crossentropy**. A més a més, hem utilitzat la funció d'Early Stopping per a cada model, monitoritzant la pèrdua al conjunt de validació amb una paciència de 10 epochs.

Aquí al report hem afegit les gràfiques amb el val_loss i el val_accuracy però per qüestions d'espai no hem pogut afegir l'AUC-ROC score ni un classification report sencer, que sí que són al notebook.

6.1 Model 1. Hiperparàmetres, diagnòstic, avaluació i millora proposada.

Veiem que, així com se'ns demanava a l'enunciat, el nostre primer model consta d'una sola capa amb 100 neurones i una funció d'activació ReLU.

Hiperparàmetre	**Valor**
Número de capes	1
Neurones a la capa oculta	1000
Funció d'activació	ReLU

Taula 3: Hiperparàmetres del Model 1

A continuació, tenim els resultats del nostre primer model que, per sorpresa semblen bastant bons. Com ens deia l'enunciat, hem començat amb una xarxa neuronal molt bàsica que contenia una sola

capa amb 100 neurones, però sembla que aquesta ha donat uns resultats que costaran de millorar.

Si ens fixem n'ha tingut prou amb 16 èpoques per aconseguir un accuracy final de 0.7525 al test i un loss de 0.5221. A més a més, sembla que tenim un model que no fa sobreajustament (overfitting) ni tampoc sembla inestable. Són molt bons resultat per a un primer model tan bàsic.

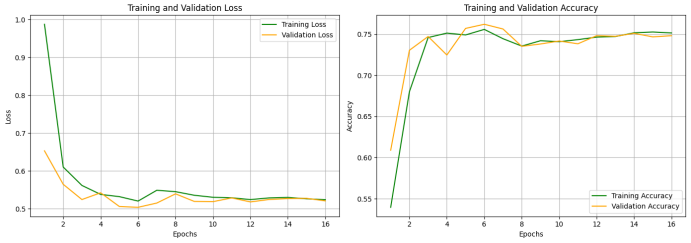

Figura 6: Loss i Accuracy Model 1

Tot i aquests resultats gairebé excel·lents, proposem intentar dividir les 100 neurones en dues capes de 50 pel següent model i veurem si millora.

6.2 Model 2. Hiperparàmetres, diagnòstic, avaluació i millora proposada.

Pel segon model, agafant la proposta del model anterior, hem decidit dividir la capa anterior en dues, alhora mantenint la funció d'activació ReLU, i intentar millorar els resultats.

Hiperparàmetre	**Valor**
Número de capes	2
Neurones a la capa oculta	50 a cada capa
Funció d'activació	ReLU

Taula 4: Hiperparàmetres del Model 2

En quant a les mètriques d'avaluació que tenim a continuació, veiem que aquest model millora lleugerament a l'anterior, aconseguint un accuracy de test de 0.7740 en aquest cas. Ara bé, si ens fixem en les gràfiques, veiem com és molt inestable ja que constantment tenim pics tant màxims com mínims en l'accuracy com en el loss.

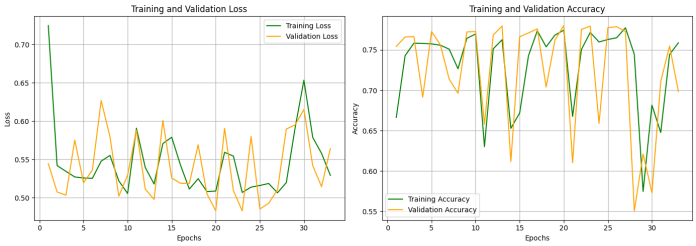

Figura 7: Loss i Accuracy Model 2

Creiem que, creant tres capes amb diferents nombres de neurones a cada capa serem capaços de millorar els resultats obtinguts i evitarem l'inestabilitat.

6.3 Model 3. Hiperparàmetres, diagnòstic, avaluació i millora proposada.

Un cop més, creem seguint les propostes anteriors i tenim un model amb dues capes ocultes de 128 i 64 neurones amb una funció d'activació ReLU i una capa de sortida amb una funció d'activació Softmax.

Hiperparàmetre	**Valor**
Número de capes	3
Neurones a les capes ocultes	128, 64
Neurones a la capa de sortida	10
Funció d'activació capa d'entrada	ReLU
Funció d'activació capa de sortida	Softmax

Taula 5: Hiperparàmetres del Model 3

En aquest cas, aconseguim els millors resultats en les mètriques amb un 0.8005 d'accuracy (ja veurem com els del model 4 no seran millors) però, mirant el gràfic següent veiem com aquest model fa sobreajustament (overfitting) i és que, mentres la corba de validation accuracy és manté bastant estable, la de train accuracy creix constantment (arribant a més de 0.81).

Figura 8: Loss i Accuracy Model 3

Per això, i vist que el model que ens ha funcionat millor és el primer que té només una capa, proposem per l'últim model fer un model amb una sola capa però reduïnt considerablement el nombre de neurones de la capa per tenir una xarxa neuronal més eficient.

6.4 Model 4. Hiperparàmetres, diagnòstic i avaluació.

En l'últim model, seguint les recomanacions anteriors i veient els resultats dels models anteriors, crearem una xarxa neuronal amb una sola capa i 10 neurones. Així reduïm considerablement el cost de la xarxa i, paral·lelament obtenim uns molt bons resultats. Mantenim la funció ReLU que ha funcionat sempre.

Hiperparàmetre	**Valor**
Número de capes	1
Neurones a la capa oculta	10
Funció d'activació	ReLU

Taula 6: Hiperparàmetres del Model 4

A les corbes de loss i training veiem que obtenim els resultats més estables i coherents fins ara. Veiem que les dues corbes de validació i train van sempre molt juntes. Podem descartar, per tant, que el model faci overfitting i sembla també que és un model molt estable sense pics sobtats.

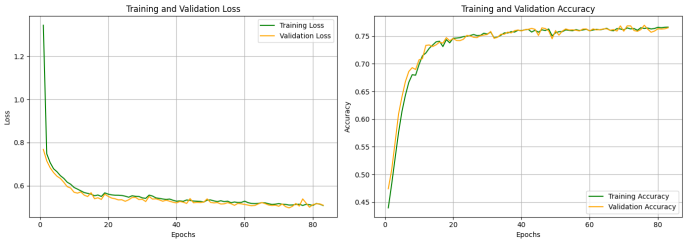

Figura 9: Loss i Accuracy Model 4

En aquest cas no tenim propostes de millora al ser l'últim model entrenat.

7 Model guanyador i conclusions

Com ja hem mencionat, dels models discutits anteriorment, el model 4 és el que considerem com a guanyador, ja que no hem aconseguit cap canvi que millorés el model. Aquest model consta d'una única capa neuronal amb 10 neurones a la capa oculta, utilitzant la funció d'activació ReLu.

A continuació, podem observar la matriu de confusió, que ens mostra la relació entre les prediccions del model i les classes reals. Aquest model té una accuracy de 0.75, que indica la proporció d'instàncies classificades correctament. Tenint en compte aquests dos mètodes d'avaluació podem concloure que el model pot tenir dificultats per a distingir entre les dues classes.

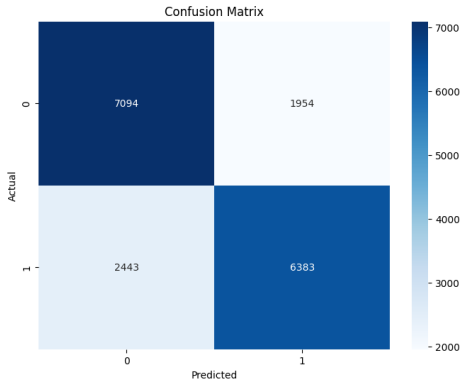

Figura 10: Mariu de confusió del model guanyador

En conclusió, ens sorprèn bastant que el millor model MLP hagi sigut un model que utilitza una sola capa neuronal, ja que sovint se solen utilitzar arquitectures més complexes amb diverses capes ocultes per aconseguir millors resultats i tasques de classificació. Això ens suggereix que les dades poden ser bastant simples o que la capacitat de generalització amb una sola capa és suficient.