UNIVERSITÀ DI PISA

Data Mining II: Advanced Topics and Applications

A report of the room occupancy dataset analysis

Ferri Lorenzo (607828)
email lorenzoferri1995@gmail.com

Ilic Ema (602796)
email e.ilic@studenti.unipi.it

Pappolla Roberta (534109)
email r.pappolla@studenti.unipi.it

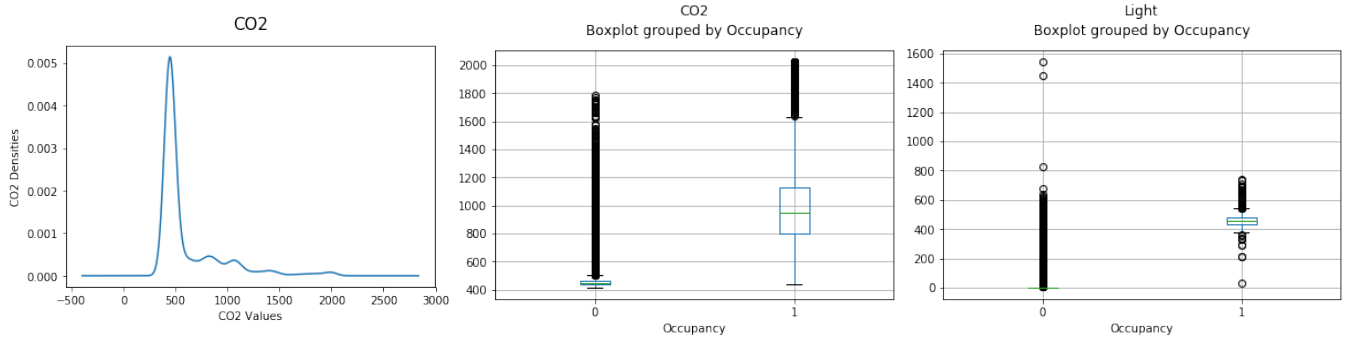
Data Mining (654AA), Anno accademico 2019/2020

Indice

1 Task 1 - Basic Classifiers and Evaluation

1.1 Data Understanding and Preparation

Il Training Set fornito è costituito da 8143 Oggetti e 7 Attributi. 'Occupancy' è l'attributo binario in output su cui deve essere eseguita la classificazione, che riguarda la previsione della presenza o meno di un soggetto (o più) in una stanza. 'Date' è il momento temporale in cui vengono rilevati i valori dei restanti attributi, che sono tutti numerici. La serie temporale del Training Set riguarda 6 giorni, dal 2015-02-04 alle 17:51:00 al 2015-02-10 alle 09:33:00, con una frequenza di rilevazione di un minuto. Il dataset non presenta alcun Missing Value e gli unici Zero Values presenti sono 5160 valori dell'attributo 'Light'. Questi non sembrano costituire un errore di rilevazione perché sono coerenti con il significato dell'attributo (luce nulla quando la stanza è chiusa e inutilizzata o nelle ore notturne). Tutti gli attributi numerici presentano un picco di frequenze sulla parte bassa dei valori, caratteristica questa che è decisamente accentuata negli attributi 'Light' e 'CO2'. E' possibile che 0 e 450 circa siano rispettivamente i valori di 'Light' e 'CO2' nella stanza quando nessuno è presente. Valori diversi invece potrebbero indicare una certa attività nella stanza. Tale ipotesi è avvalorata dai Box Plot dei due attributi con valori raggruppati per classe in output. Le due mediane dei valori di 'Light'e 'CO2' quando 'Occupancy' è 1 sono rispettivamente 450 e 950 circa, molto diversi dai picchi in distribuzione.

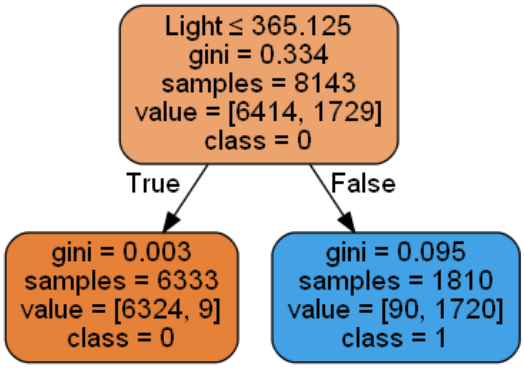


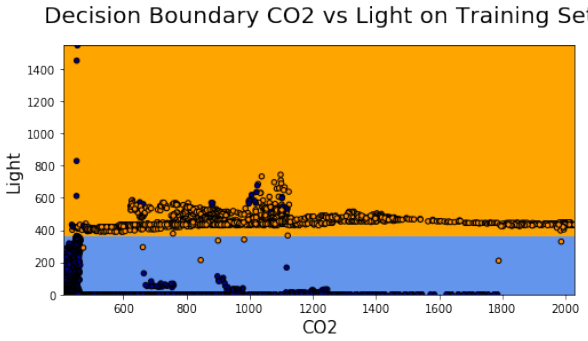
I due attributi con la maggior dipendenza lineare sono ovviamente risultati 'Humidity' ed 'Humidity Ratio'. Di fatto i due attributi esprimono lo stesso concetto, l'umidità, in due formati diversi (assoluta e relativa). Da un elementare esame degli Scatter Plot l'attributo che riesce a discriminare meglio tra le classi in output è 'Light'. L'attributo che ha mostrato la correlazione più elevata con 'Light' è stato 'Temperature': Pearson Corr = 0.65, Spearman Corr = 0.565, Kendall Corr = 0.415.

1.2 Basic Classification Methods

Nei modelli in cui risultasse necessario sono stati cercati i migliori Iper-Parametri mediante una Random Search che ottimizzasse il Positive F1 Score medio ricavato dalle 5 iterazioni compiute dalla Cross-Validation sul Training Set (Random Search CV con 5 Fold). La decisione dell'F1 come metrica da ottimizzare deriva dalla maggior attenzione riservata alle performance in termini di Precision e Recall per la classe 1 in output. I Decision Boundary sono stati rappresentati fittando il modello sulle due dimensioni rappresentate (non rappresenta dunque il reale Boundary su tutto il dataset ma ne fornisce un'approssimazione).

Decision Tree: La Random Search CV è stata eseguita più volte cercando la combinazione più efficace degli Iper-Parametri 'min samples split' e 'min samples leaf' in un dominio molto ampio per entrambi [1, 500]. Le performance migliori si hanno per valori alti degli Iper-Parametri, che portano ad alberi semplici. L'albero migliore è uno split sull'attributo 'Light' al valore 365.125: le istanze classificate positive sono quelle con 'Light' $>$ 365.125. L'attributo 'Light' quindi riveste un'importanza totale nella classificazione.



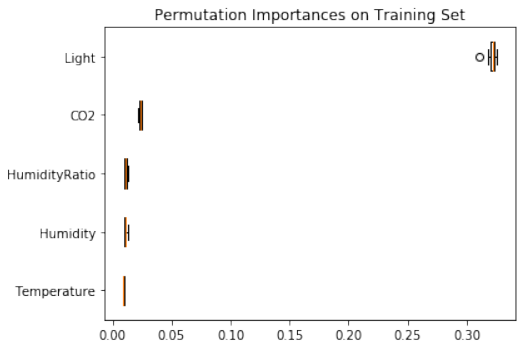


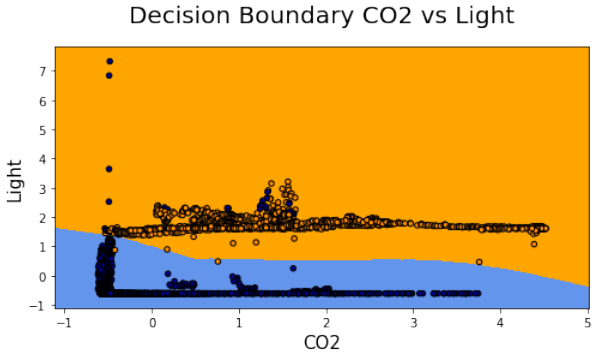
Gli oggetti misclassificati sul Training Set sono solo 99 (di cui 90 sulla classe positiva). La 5 Fold Cross-Validation sul Training Set ha restituito: Accuracy con confidenza del 95% = 0.98 (+/- 0.05); F1 Score medio pesato con confidenza del 95% = 0.98 (+/- 0.04). Un singolo Decision Boundary lineare è sufficiente a discriminare bene nelle classi in output e gli errori non sono aree densamente popolate. E' stato appurato che Decision Tree più complessi risultano overfittati sul Training Set e non performano bene sui Test Set.

	Test Set 1	Test Set 2
Accuracy:	0.9786	0.9931
F1-score:	[0.9829, 0.9714]	[0.9956, 0.9838]
Precision:	[1.00, 0.95]	[1.00, 0.97]
Recall:	[0.97, 1.00]	[0.99, 0.99]

L'AUC (Area sotto la ROC Curve) per i due Test Set è: AUC(Test 1)=0.983 e AUC(Test 2)=0.994.
L'ipotesi formulata inizialmente in fase di Data Understanding per l'attributo 'Light' sembra essere confermata. Esso è sufficiente a predire la classe in output, con un semplice Stump che divide i casi in cui la stanza è lievemente o per niente illuminata ('Light' <= 365.125), caso tipico di assenza di persone all'interno, dai casi in cui la stanza è apprezzabilmente illuminata, perché ad esempio le finestre sono state aperte o è stata accesa la luce da qualcuno. Nel Training Set sono presenti 5160 oggetti con valore dell'attributo 'Light' pari a 0, tutti questi hanno valore della classe 'Occupancy' pari a 0.

K-NN: Poiché il modello si basa sul calcolo delle distanze tra gli oggetti sia il Training che i Test Set sono stati Standardizzati (gli attributi sono stati riportati tutti a media 0 e varianza 1). La Random Search è stata eseguita cercando i migliori valori di 'n neighbors' nel dominio [1, 50] e del tipo di peso da dare a questi Neighbors per classificare l'istanza. La miglior combinazione di Iper-Parametri è 'weights'='uniform', 'n neighbors'=46.

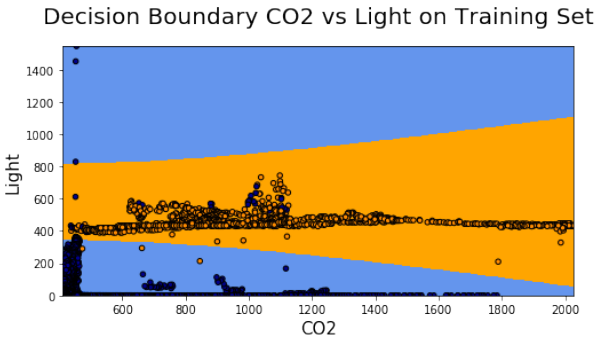




L'attributo più importante è sempre 'Light' ma anche 'CO2' riveste una seppur minima importanza. La Cross-Validation sul Training Set ha restituito una performance peggiore di quella del Decision Tree: Accuracy con confidenza del 95% = 0.95 (+/- 0.08); F1 Score medio pesato con confidenza del 95% = 0.95 (+/- 0.07). Anche la performance sui Test Set è peggiore, quindi apparentemente 'CO2' non aggiunge significatività alla classificazione.

	Test Set 1	Test Set 2
Accuracy:	0.9595	0.956
F1-score:	[0.9684, 0.9434]	[0.9721, 0.8957]
Precision:	[0.96, 0.96]	[0.97, 0.89]
Recall:	[0.98, 0.93]	[0.97, 0.90]
AUC:	0.99	0.988

Naive Bayes: Non è necessario il Tuning di nessun iper-parametro.
Come per il K-NN oltre a 'Light' anche la 'CO2' riveste una minima importanza e di nuovo i risultati in termini di Cross-Validation sul Training Set e di test sui Test Set sono lievemente peggiori dello Stump banale. Solo l'area sotto le ROC Curve è maggiore. Questo significa che usando il 50% come threshold di probabilità per la classificazione il modello ha una performance peggiore ma in genere è più stabile del Decision Tree perché i True Positive e False Positive risultano migliori se facciamo variare il threshold.

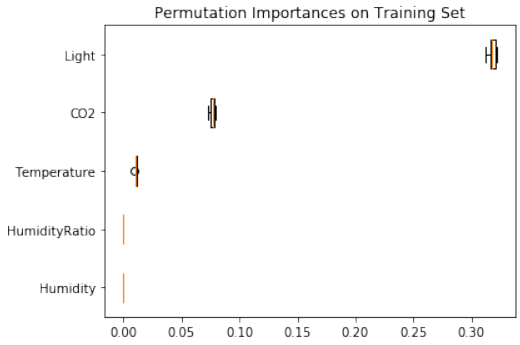


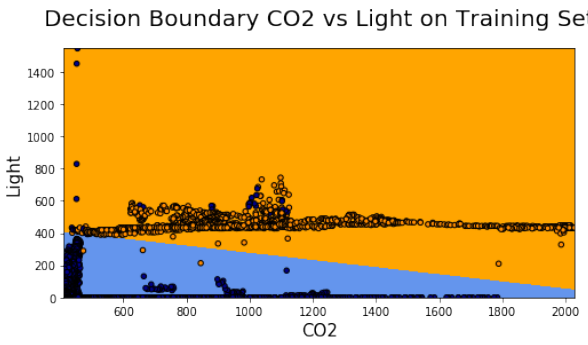
	Test Set 1	Test Set 2
Accuracy:	0.9774	0.9876
F1-score:	[0.9820, 0.9699]	[0.9921, 0.9711]
Precision:	[1.00, 0.95]	[1.00, 0.95]
Recall:	[0.97, 0.99]	[0.99, 0.99]
AUC:	0.989	0.996

Logistic Regression: Non è necessario il Tuning di nessun iper-parametro. Il modello ottenuto è il seguente:

$$P = \frac{1}{1 + \exp - (19.32 - 1.4 * \text{Temp} - 0.04 * \text{Humid} + 0.02 * \text{Light} + 0.01 * \text{CO2} - 0.1 * \text{HumidRatio})} \quad (1)$$

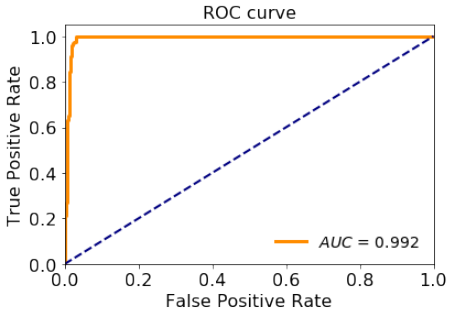
Si assiste ad un incremento ulteriore dell'importanza di 'CO2' e ad una minima importanza anche dell'attributo 'Temperature'.

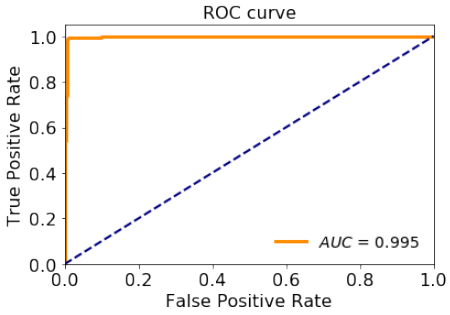




La Cross-Validation sul Training Set ha restituito valori leggermente migliori del Decision Tree, perché la varianza è minore: Accuracy con confidenza del 95%: 0.98 (+/- 0.03); F1 Score medio pesato con confidenza del 95%: 0.98 (+/- 0.03). Le performance sui Test Set sono ancora leggermente peggiori dello Stump ma l'AUC è risultata la più alta di tutti i modelli, rendendo di fatto la Logistic Regression il modello più stabile.

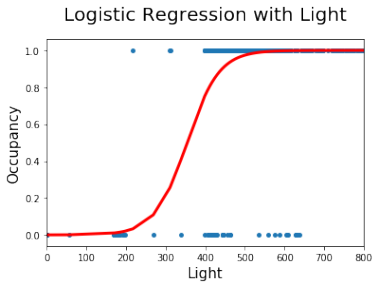
	Test Set 1	Test Set 2
Accuracy:	0.9764	0.9842
F1-score:	[0.9811, 0.9683]	[0.9900, 0.9619]
Precision:	[0.99, 0.95]	[0.99, 0.97]
Recall:	[0.97, 0.99]	[0.99, 0.95]





La regressione logistica è stata calibrata e testata anche con il solo attributo 'Light' in input. Il modello ottenuto è il seguente:

$$P = \frac{1}{1 + \exp - (0.025 - 8.752 * \text{Light})} \qquad (2)$$

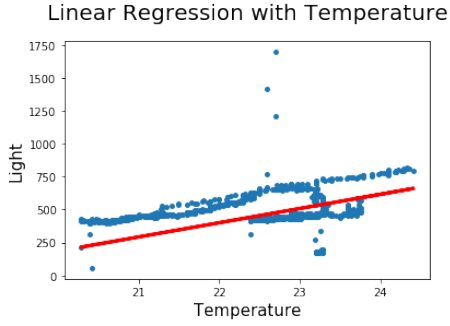


Le performance sui Test sono identiche a quelle del Decision Tree iniziale perché questo modello porta allo stesso Decision Boundary singolo.

1.3 Regression

Poiché 'Light' è risultato ampiamente l'attributo più importante è stato scelto per tentare di predirne i valori mediante le regressioni. E' stata calibrata una regressione lineare (in due dimensioni) con solo l'attributo 'Temperature' in input, che è quello risultato più correlato con 'Light' nella fase di Data Understanding. La regressione rappresentata in figura è sul Test Set e sono stati eliminati gli oggetti con valore di 'Light' pari a 0 (solo graficamente, non nel calcolo dei risultati, per focalizzare l'attenzione sulla parte importante della relazione lineare).

$$\text{Light} = -2447.03 + 124.47 * \text{Temp} \qquad (3)$$



La performance viene misurata con il Coefficiente di Determinazione, lo Scarto Quadratico Medio e lo Scarto Assoluto Medio su entrambi i Test Set.

	Test Set 1	Test Set 2
R2:	0.512	0.444
MSE:	30530.250	24109.752
MAE:	152.073	131.116

E' stata prodotta la regressione multipla aggiungendo tutti gli altri attributi numerici in input. Il modello è stato regolarizzato per tentare di ridurne la complessità, ponendo più vicini a zero i parametri che non migliorano sensibilmente la predizione dell'output. Il miglior risultato è stato raggiunto con la Ridge Regression, della quale si riportano la funzione e i risultati sui Test Set.

$$\text{Light} = -1176.14 + 0.35 * \text{CO2} - 5.78 * \text{Humid} - 15.83 * \text{HumidRatio} + 59.83 * \text{Temp} \tag{4}$$

	Test Set 1	Test Set 2
R2:	0.595	0.153
MSE:	25364.214	36724.966
MAE:	131.360	148.516

Rispetto alla regressione lineare semplice la performance migliora di poco sul primo Test Set ma peggiora di molto sul secondo. Quindi gli attributi diversi da 'Temperature' non aiutano a migliorare la predizione su 'Light'.

1.4 Dimensionality Reduction

Dimensionality Reduction Methods con Decision Tree: sono state usate le varie tecniche per ridurre le dimensioni del dataset ed è stato istruito un Decision Tree sui nuovi dataset. Il primo metodo testato è stato il Variance Threshold. Nella tabella quì sotto si può osservare come variano le dimensioni del dataset al variare del livello di varianza sotto al quale un attributo viene eliminato.

Dataset Shape	Integer Variance Threshold
(8143,5)	0
(8143,4)	1
(8143,3)	2
(8143,2)	31
(8143,1)	37926

I risultati delle classificazioni su questi nuovi dataset sono esattamente gli stessi della classificazione sul dataset intero quando la soglia è compresa tra 1 e 37926, cioè con dataset tra 2 e 4 dimensioni. Tuttavia i risultati peggiorano significativamente per un numero di dimensioni pari a una, cioè quando viene eliminato l'attributo 'Light', che è il secondo attributo con la varianza maggiore dopo 'CO2'. Li riportiamo quì per il Test Set 1.

Accuracy: 0.867167

	F1-score	Precision	Recall
Class 0	0.8911	0.93	0.86
Class 1	0.8298	0.78	0.89

Il metodo Univariate Feature Selection (UFS) seleziona 'Light' quando k=1; 'Light' e 'CO2' quando k=2; 'Light', 'CO2' e 'Temperature' con k = 3; 'Light', 'CO2', 'Temperature' e 'Humidity' con k=4. La Recursive Feature Elimination (RFE) ha individuato 'Light' come unico attributo da conservare. I risultati ottenuti dal test con UFS e RFE sono identici al caso del dataset non ridotto.

Sono state usate poi le due tecniche Principal Component Analysis (PCA) e Singular Value Decomposition (SVD). PCA e SVD restituiscono un dataset con lo stesso numero di dimensioni dell'originale ma che rappresentano in ordine le componenti principali che spiegano la maggior variabilità possibile nei dati.

Sono stati testati valori di k (numero delle prime k componenti principali da cui vogliamo che sia composto il nuovo dataset) da 1 a 4. I risultati di test in termini di Accuracy, Precision e Recall sono uguali per i diversi k, cambiano solo per tra i due metodi. Si riportano quì (sempre per il Test Set 1).

Accuracy: 0.97523

PCA	F1-score	Precision	Recall
Class 0	0.9802	0.99	0.97
Class 1	0.9668	0.95	0.99

Accuracy: 0.97036

SVD	F1-score	Precision	Recall
Class 0	0.9761	1.0	0.95
Class 1	0.9609	0.93	1.0

Sono stati testati anche altri classificatori semplici come KNN, Logistic Regression e Naïve Bayes e in questi casi i risultati cambiano in base al parametro k selezionato. Ad esempio, quando utilizziamo l'UFS con Logistic Regression o Naïve Bayes i risultati rimangono gli stessi per k = 1 e 2, ma cambiano per k = 3 e 4. In definitiva i risultati non vengono migliorati in nessun caso con qualsiasi tecnica selezionata. Tuttavia quando 'Light' e 'CO2' rimangono nel dataset (k=2 in UFS) o quando si usa una sola componente principale (k=1 in PCA e SVD) i risultati sono praticamente identici ai migliori ottenuti nella fase precedente. Questo ci porta a dire che gli altri attributi non sono importanti ai fini dell'Occupancy Detection e se il dataset fosse stato molto più complesso si sarebbero potuti eliminare per ridurne le dimensioni.

PCA a due componenti testata su diversi classificatori: il dataset risultato dall'analisi PCA con due componenti principali è stato usato per istruire altri classificatori oltre al Decision Tree: KNN, Naïve Bayes, Logistic Regression. I risultati di test dei tre modelli sono riportati quì sotto in ordine.

Accuracy: 0.97824

KNN	F1-score	Precision	Recall
Class 0	0.9826	1.0	0.97
Class 1	0.9709	0.95	1.0

Accuracy: 0.94822

Naïve Bayes	F1-score	Precision	Recall
Class 0	0.9576	1.0	0.92
Class 1	0.9336	0.88	1.0

Accuracy: 0.97824

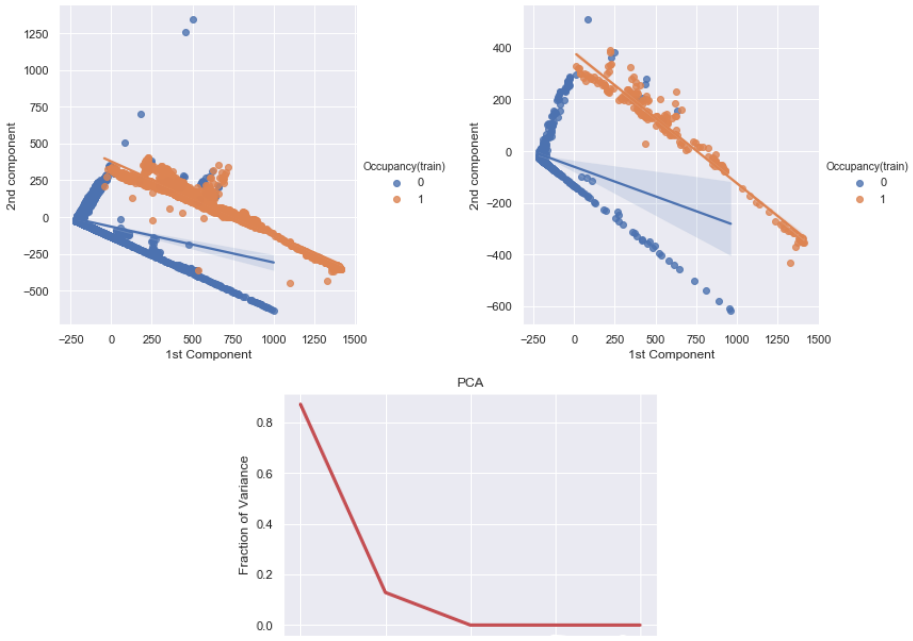
Logistic Reg	F1-score	Precision	Recall
Class 0	0.9826	1.0	0.97
Class 1	0.9709	0.95	1.0

Si confrontano i risultati di test sul Test Set 1 con l'intero dataset in input rispetto a quelli con il dataset ridotto a due componenti principali in input. Per il KNN la Precision è rimasta la stessa, l'F1 della classe 0 è leggermente migliorata e l'F1 della classe 1 è leggermente peggiorata. Per il Naïve Bayes i risultati sono tutti peggiorati. Per la Logistic Regression i risultati sono tutti migliorati. E' interessante notare che la Logistic Regression sul dataset ridotto offre risultati identici al Decision Tree sull'intero dataset (si riportano anche quelli sul Test Set 2).

Accuracy: 0.99313

Logistic Reg	F1-score	Precision	Recall
Class 0	0.9956	1.0	0.99
Class 1	0.9838	0.97	0.99

Si riporta infine lo Scatter Plot dell'intero dataset ridotto a due dimensioni con la PCA (a sinistra) e di un sample di 1000 oggetti (a destra). Sotto invece è stato riportato il grafico della frazione di variabilità catturata da ciascuna delle componenti principali in ordine.



1.5 Imbalanced Learning

La composizione della variabile di output, Occupancy, è la seguente: per il valore "0" vi sono 6414 record su 8143 (ovvero il 79%), invece per quanto riguarda il valore "1" vi sono 1729 record, il 21%. Possiamo quindi osservare un leggero sbilanciamento verso il valore "0". Abbiamo pertanto deciso di incrementare questo sbilanciamento fino ad ottenere uno squilibrio del 97% per il valore "0", e avere così solo il 3% di oggetti con valore "1". Dopo alcuni tentativi abbiamo deciso di creare le seguenti tre versioni sbilanciate del training set utilizzando diverse tecniche di undersampling e oversempling:

1. **Undersampled (che chiameremo *dtU*):** la prima versione di data training è stata modificata attraverso una variante del modello "Near Neighbor", ovvero "Near Miss", decrementando così il numero di oggetti con valore "1" da 1729 fino a 198 record e lasciando invariato il numero di record per il valore "0".

2. **Oversampled (che chiameremo *dtO*):** nel secondo abbiamo invece lasciato invariato il numero di oggetti con valore "1" ed incrementato il numero di record con valore "0" a 55904, in maniera tale che rappresentassero il 97% del data training, implementando il metodo "SMOTE".

3. **Undersampled e Oversampled (che chiameremo *dtUO*):** la composizione dell'ultimo è stata modificata completamente ottenendo il seguente training: 55031 oggetti per il valore "0" e 1702 per il valore "1" attraverso l'implementazione delle seguenti tecniche: "Random OverSampler" e "Neighbourhood Cleaning Rule".

Classificazioni base

Procediamo ora ad applicare diverse tecniche di classificazione ai tre training set modificati. Essendo questi sbilanciati in maniera considerevole, l'obiettivo è quello di ottenere un valore dell'accuracy pari almeno a classificare sempre con il valore "0", quindi del 97%.

Decision tree

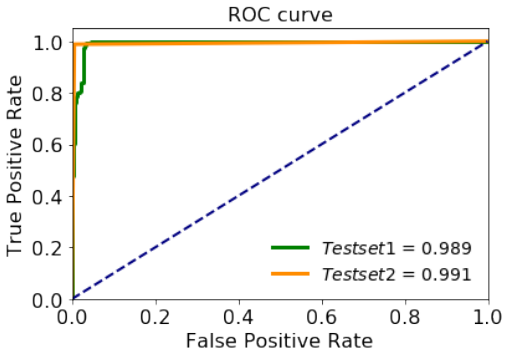
Anche qui si è cercato di ottenere la migliore classificazione attraverso l'uso della Randomized Search, cercando quindi la migliore combinazione degli Iper-Parametri "min samples split" e "min samples leaf". Abbiamo effettuato diverse prove con il numero di iterazioni in un range tra i 150 e 600. I parametri ottenuti sono i seguenti:

	Min samples split	**Min samples leaf**
Undersampled	308	2
Oversampled	432	8
Undersampled e Oversampled	372	90

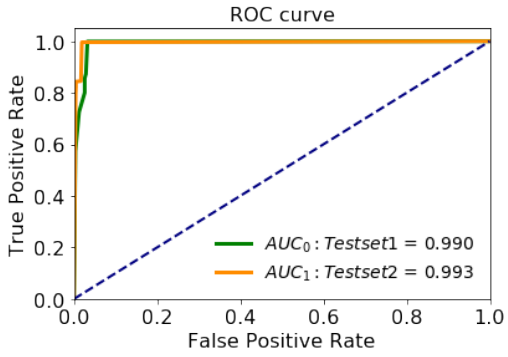
Tutte le classificazioni ottengono un'Accuracy maggiore del 97% nel training set (0.9868, 0.9859 e 0.9859). Vediamo ora quali ottengono risultati interessanti nei due test set. Le classificazioni basate sugli ultimi due training set (dtO e dtUO) hanno rispettivamente ottenuto: nel primo test set un'Accuracy di 0.90994 e 0.91219, e nel secondo di 0.9477 e 0.96021. Non si evince niente di particolarmente interessante, dato che abbiamo ottenuto valori migliori in precedenza. Osserviamo quindi i valori ottenuti con la prima classificazione, quella su dtU:

	Test Set 1	**Test Set 2**
Accuracy:	0.97673	0.99292
F1-score:	[0.98144, 0.96881]	[0.99551, 0.98324]
Weighted Avg F1-score	0.98	0.99
Precision:	[0.99, 0.95]	[1.00, 0.98]
Recall:	[0.97, 0.99]	[0.99, 0.99]

I valori ottenuti nel secondo test set sono estremamente positivi, tra i migliori; ma comunque inferiori a quelli ottenuti con l'utilizzo dello stesso modello sul training originale. Dai grafici sottostanti si può osservare che, nonostante l'Accuracy del terzo training set sia inferiore a quella del primo, quest'ultimo ha un'area sotto la curva maggiore in entrambi i test set.



(a) AUC dell'undersampled data training nei due test set.



(b) AUC dell'undersampled e oversampled data training nei due test set.

K Nearest Neighbor, Logistic Regression

Anche questi due modelli riescono ad ottenere un'Accuracy maggiore del minimo sopracitato (sempre intorno al valore 0.9858) usando tutti e tre i training modificati, ma non ottengono risultati particolarmente interessanti nei test set. Questi, infatti, hanno valori dell'Accuracy che partono da 0.8435 e arrivano ad un valore massimo di 0.9602. Per il test set 1 non si riesce ad ottenere un valore maggiore al 0.9126. La combinazione di PCA e Logistic Regression aveva ottenuto uno tra i migliori risultati; applicando, invece, lo stesso modello sul training sbilanciato con le tre tecniche differenti non si riescono ad ottenere risultati interessanti.

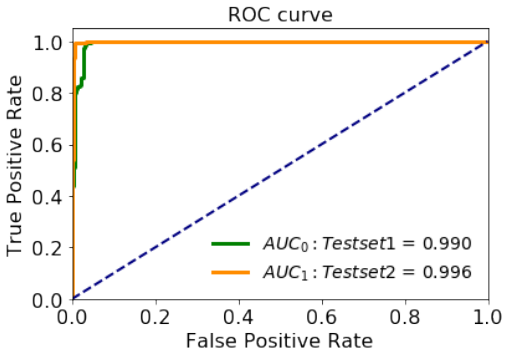
Naive Bayes

Risultati interessanti sono stati ottenuti con il classificatore "Naive Bayes". Questo modello non richiedeva nessun Iper-Parametro in input, perciò non si è ricorsi all'utilizzo del metodo Randomized Search. Nella tabella sottostante è possibile osservare i risultati ottenuti usando i tre training:

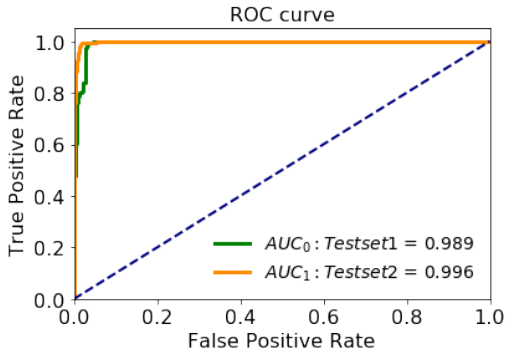
	Accuracy test set 1	**Accuracy test set 2**
Undersampled	0.91744	0.95888
Oversampled	0.97711	0.98985
Undersampled e Oversampled	0.97711	0.97898

E' curioso osservare come i dataset con cui si riesce ad ottenere risultati ottimali, utilizzando questo classificatore, siano i training maggiormente modificati (dtO e dtUO). Questi hanno ottenuto valori positivi dell'Accuracy nei precedenti modelli, ma non riuscivano a raggiungere il valore minimo accettabile. I due data training modificati ottengono i medesimi risultati nel primo test set; ma la classificazione ottenuta dal training dtO ha prodotto dei valori migliori nel secondo test set, oltre tutto uno dei modelli con la maggiore AUC per entrambi i test set.

	Oversampled		**Under e Oversampled**	
	Test Set 1	**Test Set 2**	**Test Set 1**	**Test Set 2**
Accuracy:	0.97711	0.98985	0.97711	0.97897
F1-score:	[0.98172, 0.96939]	[0.99354, 0.9762]	[0.98172, 0.96939]	[0.986746, 0.94922]
Weighted Avg F1-score	0.98	0.99	0.98	0.98
Precision:	[1.00, 0.95]	[1.00, 0.96]	[1.00, 0.95]	[0.98, 0.96]
Recall:	[0.99, 0.99]	[0.99, 0.99]	[0.97, 0.99]	[0.99, 0.94]



(c) AUC dell'oversampled data training nei due test set.



(d) AUC dell'undersampled e oversampled data training nei due test set.

2 Task 2 - Advanced Classifiers and Evaluation

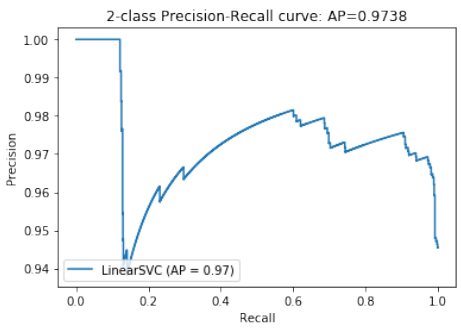
2.1 Support Vector Machines (SVM)

2.1.1 Linear SVM

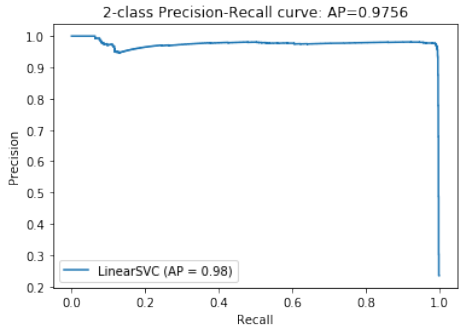
Per cominciare abbiamo fatto diverse prove assegnando diversi valori ai seguenti parametri: penalty, loss, multi_class, dual, fit_intercept, tol e C. Le migliori combinazioni di questi parametri sono state trovate mediante la Randomized Search. Da notare che quella con il miglior valore di Accuracy per il primo test set ottiene un risultato peggiore nel secondo: 1) 0.97936, miglior valore dell'Accuracy rispetto a quella ottenuta classificando con il Decision Tree; 2) 0.81593, con una Precision per il valore "1" di 0.53. Visto i risultati non proprio ottimali ottenuti nel secondo test set si è optato per un'altra combinazione dei parametri, più precisamente diverse combinazioni di parametri, che ottengono i seguenti risultati:

	Test Set 1	**Test Set 2**
Accuracy:	0.97899	0.98359
F1-score:	[0.98319, 0.97197]	[0.9895, 0.96232]
Weighted Avg F1-score	0.98	0.98
Precision:	[1.00, 0.95]	[1.00, 0.93]
Recall:	[0.97, 1.00]	[0.98, 1.00]

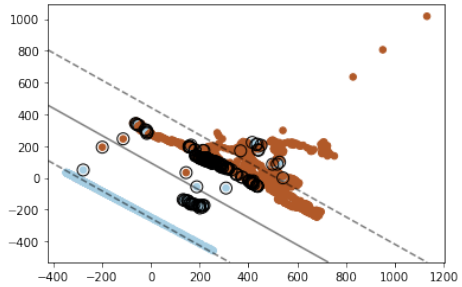
Qui di seguito vi è una rappresentazione grafica dei valori Precision e Recall che è possibile ottenere nei due test set variando le soglie ed il rispettivo valore medio ponderato con l'aumento del recall dalla precedente soglia utilizzata come peso, assunto dalla Precision (**AP**). Questo ci aiuta a comprendere la qualità della classificazione ottenuta. Come è possibile vedere nel primo test set la curva presenta diversi e profondi picchi, invece la seconda curva è più lineare. Possiamo affermare che i risultati ottenuti sono ottimi in quanto le accuracy ottenete nei due test set sono leggermente superiori al AP ($0.9789 > 0.9738$ e $0.98359 > 0.9756$) e abbiamo anche ottenuto alti valori di Precision e Recall, cosa non scontata nel primo testset con questo modello. Considerando che non sappiamo le combinazioni di risultati ottenibili tra i due testset siamo soddisfatti della classificazione ottenuta.



(e) P-R curve per il primo test set.



(f) P-R curve per il secondo test set.



(g) Soluzione grafica PCA.

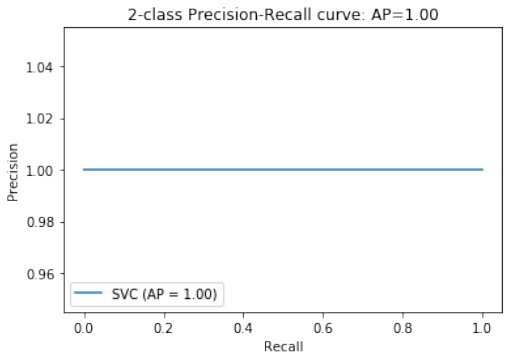
Per avere un'idea grafica della soluzione scelta si è deciso di applicare il medesimo modello al training trasformato attraverso la PCA, il quale otteneva risultati simili, che è possibile osservare qui sopra.

2.1.2 SVC

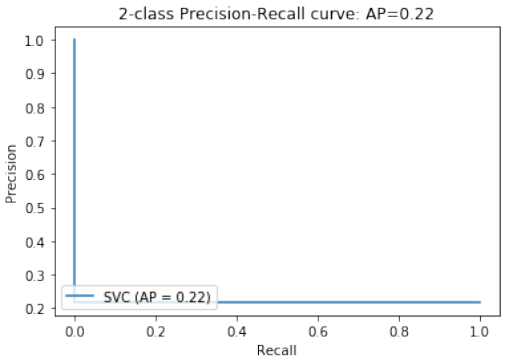
Per l'analisi effettuata attraverso la Non Linear SVC si è cercata la miglior combinazione tra le variabili C, kernel, tol, degree (per kernel='poly'), gamma e decision function shape. Nei risultati ottenuti con la classificazione NLSVM è molto accentuata la differenza di risultati dell'accuracy tra il primo test set e il secondo, è molto facile ottenere risultati estremamente positivi nel primo test set, e altrettanto negativi nel secondo (ES: 1)0.9973733583489681 2) 0.34290401968826906). Un risultato molto interessante è stato trovato assegnando i seguenti valori ai parametri: C=1, tol=0.01, gamma=100, class_weight='balanced', random_state=42, con questi parametri è possibile classificare perfettamente il primo test set, e sufficientemente bene il secondo, l'unico problema è che nel secondo test vi è una Precision per il valore "1" di 0, i risultati ottenuti sono i seguenti:

	Test Set 1	**Test Set 2**
Accuracy:	1	0.78989
F1-score:	[1.00, 1.00]	[0.88261, 0.00]
Weighted Avg F1-score	1.00	0.70
Precision:	[1.00,1.00]	[0.79, 0.00]
Recall:	[1.00, 1.00]	[1.00, 0.00]

Altrettanto interessanti sono le curve Precision-Recall derivanti da questa classificazione per i due test set:



(h) P-R curve per il primo test set.



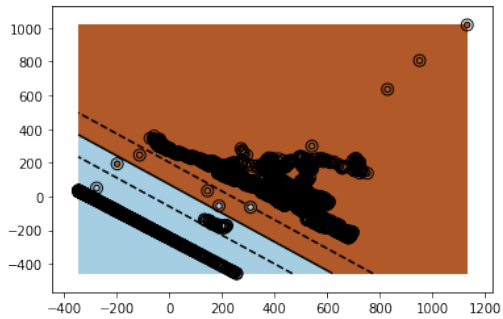
(i) P-R curve per il secondo test set.

Attraverso l'uso della Randomized search abbiamo trovato la miglior classificazione per il testset1 ma che non ottiene un buon risultato nel secondo. Il miglior risultato è stato ottenuto cambiando i parametri manualmente. Per prima cosa è interessante notare che con entrambi i classificatori (Linear e NonLine) non è possibile ottenere una accuracy maggiore di 0.9789868667917448 nel primo testset, che ottenga buoni risultati anche nel secondo. La miglior classificazione e stata ottenuta con i seguenti parametri: C=1.65, kernel= 'poly', tol= 0.3, degree=1,gamma='scale',decision_function_shape='ovr'. I risultati sono ottimi e ottengono una accuracy maggiorare dall'AP (= 0.969720 e 0.974440), e sono i seguenti:

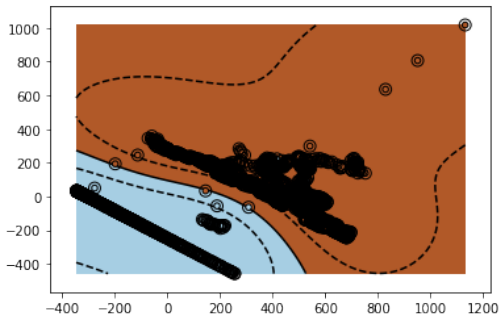
	Test Set 1	**Test Set 2**
Accuracy:	0.97899	0.98605
F1-score:	[0.98319 0.97197]	[0.99110 0.96774]
Weighted Avg F1-score	0.98	0.99
Precision:	[1.00, 0.95]	[1.00, 0.94]
Recall:	[0.97, 1.00]	[0.98, 1.00]

Le due Precision-Recall che si ottengono sono quasi uguali a quelle ottenute nella LSVM quindi è inutile mostrarle. I due classificatori individuati ottengono risultati molto simili: uguali nel primo testset ma si differenziano nel secondo in quanto con la NLSVM ottiene valori maggiori per tutti gli indici di performance.

Decision Boundary Abbiamo ottenuto i seguenti grafici utilizzando il training set modificato attraverso la tecninca della PCA. Per individuare la miglior Decision Boundary ricavata bisogna verificare quale ha il margine maggiore. Siamo in presenza di un caso non linearmente separabile in quanto l'iperpiano non separa perfettamente le due classi, quindi oltre al margine c'è da considerare anche la variabile di allentamente (slack variables). I margini sono visibili nel grafico, sono rappresentati dalla linea tratteggiata, per questi motivi pensiamo che la miglior soluzione grafica si ottiene con il modello NonLinear.



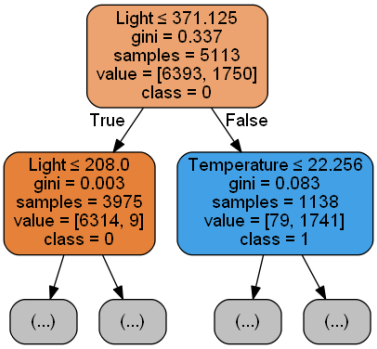
(j) Decision Boundary Linear SVM per testset 1.

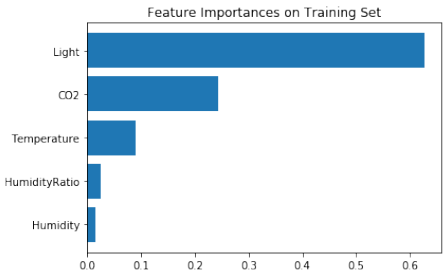


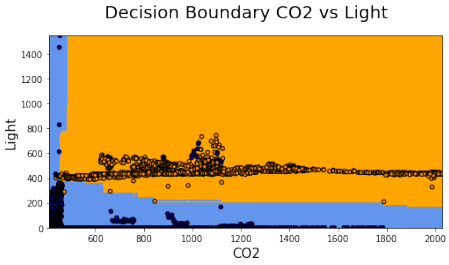
(k) Decision Boundary Non Linear SVM per testset1.

2.2 Ensemble Learning

Random Forest: I Decision Boundary tipici che hanno restituito i risultati più stabili, apprezzabili dal valore degli indici di validazione sul Training Set e dell'area AUC sul Test Set, sono stati quelli della Logistic Regression e Naive Bayes. Tramite la Random Forest è stato posto l'obiettivo di raggiungere quella stabilità, tramite un Decision Boundary analogo, insieme ad una performance sui Test Set pari a quella ottenuta con il Decision Tree. Lo scopo è stato raggiunto con un modello che ha per Iper-Parametri: 'max features'='log2', 'random state'=0, 'n estimators'=100', 'max depth'=3 e i valori di default per gli altri. Il valore migliore della 'max depth' è stato trovato mediante la Random Search cercando nel dominio [1,100] e ottimizzando per l'F1 Score. Il fatto che il 'max depth' ottimale sia un valore basso indica che sono sufficienti singoli alberi semplici non overfittati. Le feature diverse da 'Light' aumentano la loro importanza. Il Decision Boundary è più preciso per valori bassi di 'CO2'. Quando 'CO2' è basso 'Light' viene splittato ad un valore leggermente superiore a 365.125, come si nota dall'albero proposto quì (split a 371.125). In questo modo la zona di maggior contatto tra gli addensamenti delle due classi in output viene separata con più precisione.

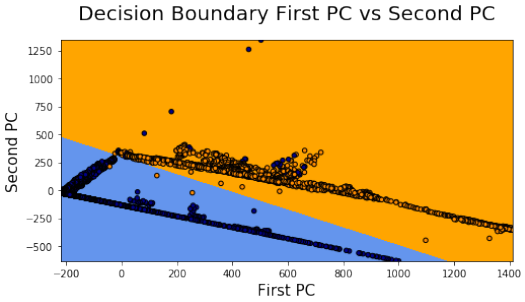




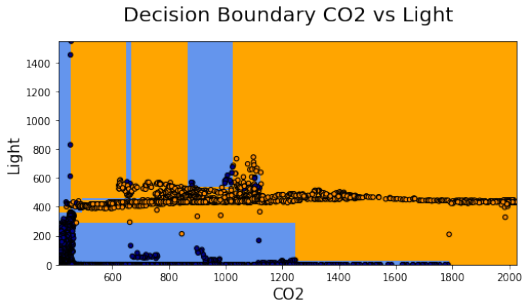


La 5 Fold Cross-Validation sul Training Set ha restituito: Accuracy con confidenza del 95% = 0.98 (+/- 0.03); F1 Score medio pesato con confidenza del 95% = 0.98 (+/- 0.03). Con una varianza minore del Decision Tree semplice il modello risulta più stabile sul Training Set. Sui Test Set i risultati delle metriche di valutazione sono identici a quelli del Decision Tree iniziale. Migliora però l'AUC per entrambi i Test Set: AUC(Test 1)=0.986 AUC(Test 2)=0.996. Dnque il modello è più stabile anche sui Test Set e l'obiettivo è stato raggiunto (questo modello è leggermente migliore dello Stump sul dataset iniziale non trasformato).

Bagging: Il Bagging è stato eseguito con lo stesso scopo della Random Forest, con classificatori diversi dal Decision Tree: Support Vector Machine, Naive Bayes, Logistic regression. Particolare attenzione è stata riservata a quest'ultimo perché era stato il modello più stabile nella fase di classificazione di base. I risultati migliori in assoluto fino ad ora infatti vengono ottenuti con un Bagging composto da 50 Logistic Regression (50 è il numero di base classifier minimo per raggiungere il risultato migliore).
A fronte degli stessi risultati della Random Forest sia per la Cross-Validation che per i test abbiamo quì i migliori valori dell'AUC: AUC(Test 1)=0.992 AUC(Test 2)=0.996. I risultati con gli altri tipi di classificatori base sono tutti peggiori. La Regressione Logistica è stato il modello migliore anche dopo aver ridotto il Dataset a due dimensioni con la PCA, dunque è stato fatto un tentativo di classificazione con Bagging sulle Logistic Regressions anche con il dataset bidimensionale uscito dalla PCA. I risultati sono stati lievemente peggiori di quelli discussi quì sopra ma si riporta comunque il Decision Boundary nelle prime due componenti principali (PC).



Boosting: Come è noto l'AdaBoost Classifier è un Ensamble method che si basa sulla maggior attenzione riservata agli oggetti misclassificati nelle iterazioni precedenti. A questo modello quindi è stato riservato lo scopo di produrre un Decision Boundary che individuasse l'area centrale dello Scatterplot dove è presente un piccolo addensamento di oggetti con label in outpu 0 che sono sempre stati classificati male. Essi potrebbero essere i False Positive che hanno tenuto lievemente più bassa la Precision in tutti i classificatori di base. Lo scopo è stato raggiunto con un modello che usa il Decision Tree come Base Classifier e che ha per Iper-Parametri: 'n estimators'=100, 'random state'=0 e 'learning rate'=0.2, quest'ultimo trovato ottimizzando l'F1 Score tramite una Grid Search nel dominio [0,1] con step 0.1.



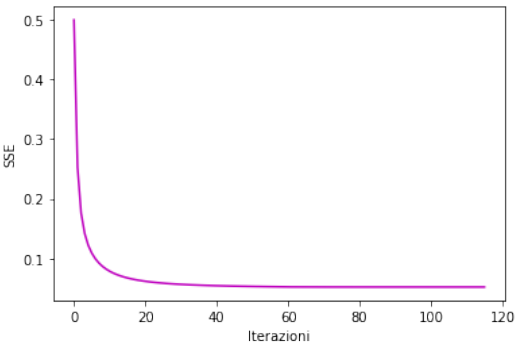
Come si può notare dal Boundary vengono individuati in pratica tutti gli oggetti precedentemente misclassificati sul Training Set. L'esito sia in termini di Cross-Validation sul Training Set che in termini di Test peggiora sensibilmente. Per quanto riguarda la validazione Accuracy ed F1 Score medio pesato passano entrambi ad un valore di 0.93 con due deviazioni standard (95% di confidenza) di 0.12 circa. Sui due Test Set l'Accuracy passa a 0.922 e 0.938 rispettivamente e la Positive Precision peggiora passando a 0.94 e 0.85 rispettivamente. Le AUC calano a 0.905 e 0.91. Il modello è evidentemente overfittato sul Training Set e questo dimostra che gli oggetti al centro dello ScatterPlot, con 'Light' superiore a 600 e 'CO2' intorno a 1000 sono più un caso specifico presente nel Training Set fornito piuttosto che una relazione da cui poter derivare conclusioni generali.

L'AdaBoost è stato utilizzato anche con un Naive Bayes Classifier che ha permesso di individuare come zona della classe 0 anche quella con 'CO2'>1200 e 'Light'>600. I risultati però sono stati altrettanto pessimi.

2.3 Neural Networks and Deep Learning

Single Layer Perceptron: Gli iper-parametri usati per ottenere i risultati migliori con la funzione Perceptron sono stati scelti usando la CrossValidationCV, massimizzando per l'F1. Il miglior modello ha alpha uguale a 0.0281 e 'Penalty' uguale a 'L1' (regressione lasso), ma i risultati ottenuti sono pessimi: Accuracy=0.756; F1-Score Weighted Average=0.72.

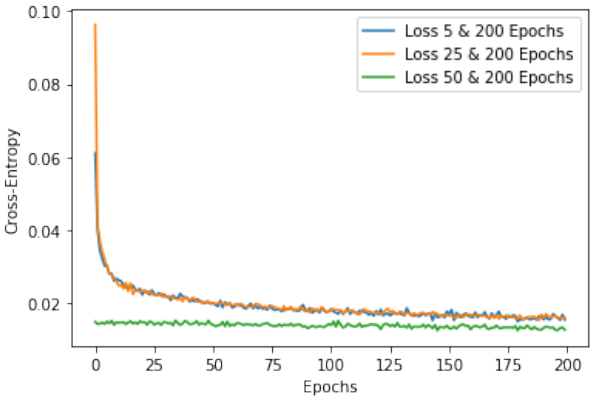
Multi Layer Perceptron: sono state eseguite molteplici prove con i vari iper-parametri per il Multi Layer Perceptron e si è assistito ad alcuni esiti interessanti. Ad esempio, con due hidden layer i risultati non variano molto se il primo layer contiene da 100 nodi in su o se il secondo ne contiene da 50 in su. Questo ha portato a preferire l'uso di 100 e 50 nodi rispettivamente nel primo e secondo layer per evitare modelli complessi (possibilmente Overfittati) che non portano a risultati apprezzabilmente migliori.



In una delle prime prove con il 'MLP' dunque si è settato il primo layer a 100, il secondo a 50, ed anche un terzo layer a 25, scegliendo Adam come solver, l'alpha di 0.0001, il learning rate 'Adattivo', la funzione di attivazione 'hyperbolic tangent' ed un 'Momentum' di 0.9. La Loss è risultata bassa (0.01868) ed Accuracy e F1 sono migliorati rispetto al Single Layer Perceptron (0.92908 e [0.9457, 0.89778], rispettivamente), ma ancora non raggiungono i risultati migliori ottenuti con altri classificatori. Successivamente ad altre prove i risultati si sono migliorati scegliendo Stochastic Gradient Descent come solver al posto di adam e mantenendo soltanto i primi due layers da 100 e 50 nodi rispettivamente per evitare l'Overfitting (il terzo layer non apporta miglioramenti significativi). L'alpha è stato

aumentato a 0.01 sempre allo scopo di penalizzare modelli complessi. Di nuovo è stato scelto L2 come 'Penalty' (lasso regression). Nonostante un lieve aumento della Loss a 0.05 rispetto al modello precedente (il cui grafico può essere osservato sopra), l'Accuracy è aumentata a 0.9786 e l'F1 a [0.9829, 0.97124], rispettivamente per i valori in output 0 e 1.

Deep Learning: per approfondire le analisi sui Deep Neural Networks è stata usata anche la libreria Keras. E' stato istruito un modello con 2 Hidden Layer da 100 e 50 nodi rispettivamente, entrambi con funzione di attivazione 'relu', e un Output Layer con funzione di attivazione 'sigmoid'. Essendo un problema binario si è usato la Cross-Entropy. Come modello per l'ottimizzazione si è usato 'Adam', massimizzando per l'Accuracy. Il numero delle epoche è stato settato a 200. Questo DNN è stato testato con un numero di Batch pari a 5, 25 e 50, ottenendo i risultati in figura.
Si può osservare che le curve delle Loss per 5 e 25 Batches seguono pressoché lo stesso andamento (l'unica differenza rilevante è che la prima parte più in basso). La curva della Loss per 50 Batches invece parte già da un valore di Loss molto basso e rimane sempre abbastanza stabile.



Il training set è stato suddiviso in Training e Validation Set. Si è costruito un modello che fa uso dell'Early Stopping con 'Patience' di 5 iterazioni, calcolando la Loss sul Validation Set, sempre con 200 Epoche. Questi sono i risultati riguardanti Loss e Accuracy di tutti e quattro i modelli costruiti: 5 Batch: Loss=0.252636, Accuarcy=0.951219 (Accuracy più alta); 10 Batch: Loss=0.183976 (Loss più basso), Accuarcy=0.922702; 25 Batch: Loss=0.553305, Accuarcy=0.923452; 50 Batch: Loss=0.737575, Accuarcy=0.231895. Successivamente è stata introdotta anche la regolarizzazione L2 (lasso) per cercare di evitare l'Overfitting. sono stati usati 3 Hidden Layer, mantenendo invariato il resto degli Iper-Parametri. Il modello è stato eseguito con Batch Size di 10 e numero di epoche pari a 100. Il risultato comunque non è stato soddisfacente, con Accuracy di 0.64 e Loss di 5.63. In fine, si è usata la RandomizedSearchCV per ottenere i migliori Iper-Parametri, cercando nei domini: numero di Layers da uno a sei; numero di Hidden Layers da 2 a 10 e 25, 50 e 100; funzione di attivazione 'relu' e 'Hyperbolic Tangent'; modelli di ottimizzazione 'adam' e 'adagard'. Il miglior risultato si è ottenuto con: 'optimizer'='adagrad'; 'n layers'=5, 'h dim'=7; 'activation'='tanh'. I risultati sul Test Set 1 sono stati: Accuracy=0.97826; Loss=0.156431. E sul Test Set 2: Accuracy=0.926887; Loss=0.217802.

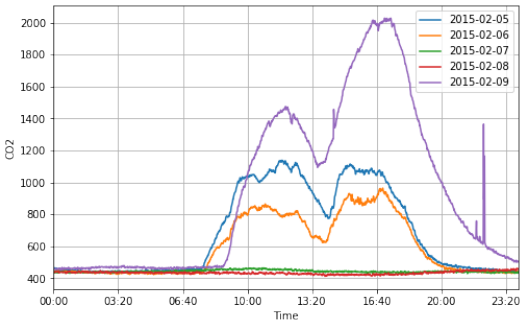
3 Task 3 - Time Series Analysis and Forecasting/Classification

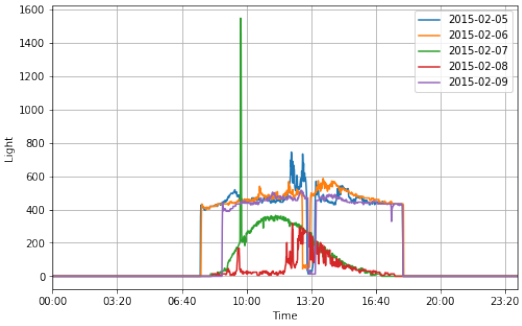
L'attributo 'date' è costituito da valori del tempo rilevati ogni minuto ma sono presenti diversi valori dove anziché all'istante del minuto preciso è stato osservato il valore un secondo prima (es. 17:51:59 anziché 17:52:00). Si tratta di un piccolo chiaro errore di rilevazione che potrebbe inficiare le analisi e quindi tutti i valori con 59 secondi sono stati corretti mediante il loro Offset di 1 secondo in avanti, sia nel Training che nei due Test Set. Il primo Test Set contiene rilevazioni temporalmente precedenti al Training ed il secondo Test Set contiene rilevazioni successive. Il termine del primo Test Set non coincide con l'inizio del Training e la fine di quest'ultimo non coincide con l'inizio del secondo Test (ci sono diverse ore di distanza tra i tre).

3.1 Similarities

Il training set parte dalla data "2015-02-04 17:51:00" e termina con "2015-02-10 09:33:00". Abbiamo deciso di analizzare e confrontare le diverse giornate che sono contenute nel training, ma solo quelle i cui dati sono stati rilevati per l'intera giornata, per questo abbiamo escluso la prima e l'ultima giornata. I 5 giorni analizzati partono dal giovedì e arrivano al lunedì successivo. Possiamo notare che nelle giornate di sabato

(7-02) e domenica (8-02) per le variabili "CO2" e "Light" vi sono valori e andamenti differenti. Per la prima in entrambe le giornate i valori sono costanti per tutto l'arco temporale, nella seconda l'andamento tra i giorni infrasettimanali, il sabato e la domenica sono completamente differenti. In quest'ultima possiamo notare un picco minimo nella fascia oraria 13:20, quindi orario di pranzo. Questo dipende dal fatto che probabilmente le stanze in questione sono uffici e il fine settimana questi sono chiusi. Questa teoria è confermata dal fatto che in queste giornate l'Occupancy assume solo il valore 0.

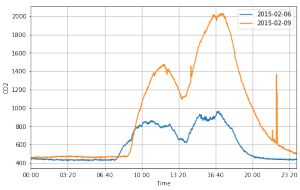


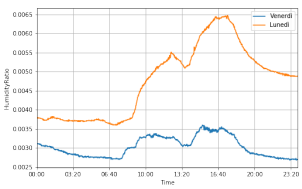


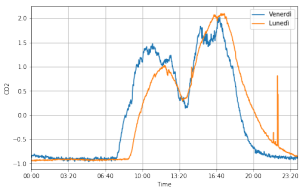
Le altre variabili non presentano questa grossa differenza tra i giorni infrasettimanali e quelli del fine settimana perché dipendono maggiormente dall'ambiente esterno. Proprio per questa ragione abbiamo deciso, dopo svariate prove, di selezionare le giornate "2015-02-06", ovvero Venerdì, e "2015-02-09", Lunedì, per verificare la similarità fra il primo giorno feriale e l'ultimo. La differenza tra giorno feriale e festivo sarebbe eccessiva, e quindi può essere più interessante questo tipo di analisi.
Distances: Abbiamo calcolato le distanze delle diverse variabili dei due giorni con i tre metodi, abbiamo ottenuto che:

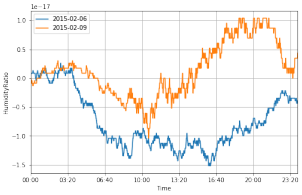
Variabile/Distanza	**Euclidean**	**Manhattan**	**Dynamic Time Warp.**
Light	4413.853369254327	61127.71666666666	641.4804101971212
CO2	18.015850971010515	467.43745555758085	3.9717534527948914
HumidityRatio	4.1040373447994576e-16	1.3067671944533288e-14	2.1525517407203092e-16

Le distanze sono decisamente maggiori con la Manhattan Distance ma il risultato non cambia. La serie che presenta maggiore differenza fra i due giorni (primo e ultimo feriale) è quella che riguarda la variabile delle luce. Ma non possiamo escludere che questa grossa differenza non dipenda semplicemente da una giornata particolarmente nuvolosa, considerando che comunque i dati sono stati rilevati nel mese di febbraio.
Transformations: dopo aver applicato le diverse tecniche di trasformazione alle giornate precedentemente selezionate e alle diverse variabili si è deciso di utilizzare le seguenti trasformazioni e attributi : 1) "Light", non trasformata; 2) "CO2" -> Amplitude Scaling; 3) "HumidityRatio"-> Trend removal, con valore del parametro "window" pari a 1. Qui sotto è possibile vedere le due variabili, in ordine CO2 e HumidyRatio, prima della trasformazione (prime due immagini) e dopo (ultime due).





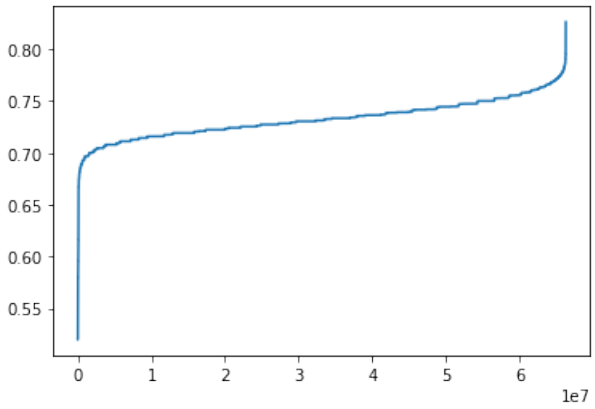




Time Series Approximation: Abbiamo applicato tre diverse tecniche di approssimazione (Piecewise Aggregate Approximation, Symbolic Aggregate Approximation e 1d-SAX) alle serie temporali ottenute, modificando i parametri cercando di ottenere delle approssimazioni il più simili possibile alla serie originale. La SAX approximation non riesce ad approssimare in maniera ottimale le serie, soprattutto per quanto

riguarda la variabile luce, anche utilizzando alti valori del parametro "n_sax_symbols". Invece quella che riesce ad approssimarle al meglio è la PAA.

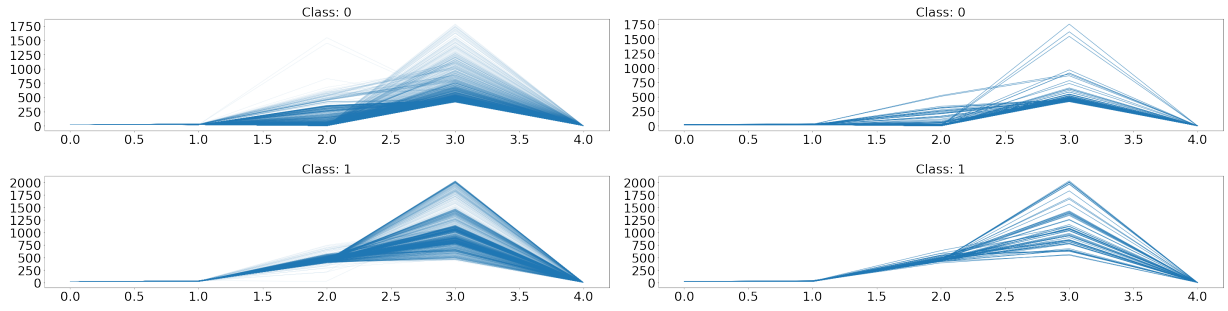
Clustering: Per quanto concerne la ricerca dei cluster abbiamo utilizzato le tecniche "Shape-based Clustering" e "Compression based Clustering". Per la prima abbiamo utilizzato "TimeSeriesKMeans" e si sono testati diversi valori degli iper-parametri "k" [per valori da 2 a 10] e "metric" [“euclidean”, “dtw” e “softdtw”], ma per ogni valore di "k" le tre differenti metriche restituivano lo stesso risultato. Per la seconda abbiamo prima di tutto trasformato gli elementi in stringe, attraverso la funzione "cdm_dist", insieme alla compressione e al calcolo delle distanze fra di loro usando "Pairwise distances". Coppia per coppia, tutte le serie temporali sono state trasformate nelle stringe. La distanza fra di loro è stata calcolata, ottenendo la matrice di 8143x8143 valori. La matrice delle distanze è stata trasformata in un’array di una dimensione sola di lunghezza 66308449, e la successione ordinata di queste distanze può essere osservata nella figura riportata sotto.



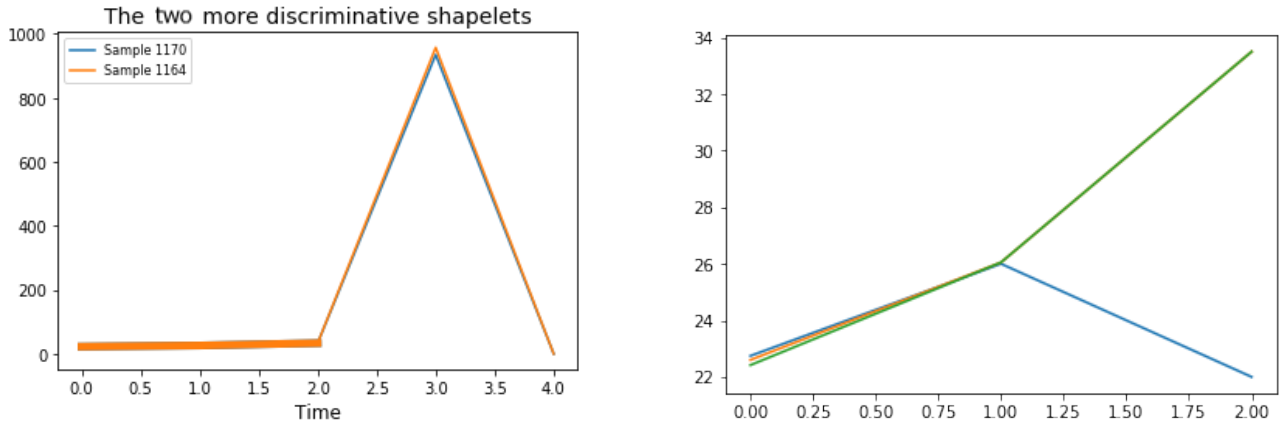
Poi, è stato usato il DBSCAN per ricalcolare la matrice, settando come epsilon il valore di 0.72, ottenuto dalla osservazione del gomito sul grafico precedente. Come valore del parametro metric abbiamo utilizzato ‘precomputed’ in modo tale da poter fittare la matrice calcolata precedente. Il risultato è stato un’array di 8143 valori, ognuno di valore 0 o -1, dove 0 segna il valore appartenente al cluster mentre -1 l’oggetto indicato come noise.

3.2 Shapelet and Motif Discovery

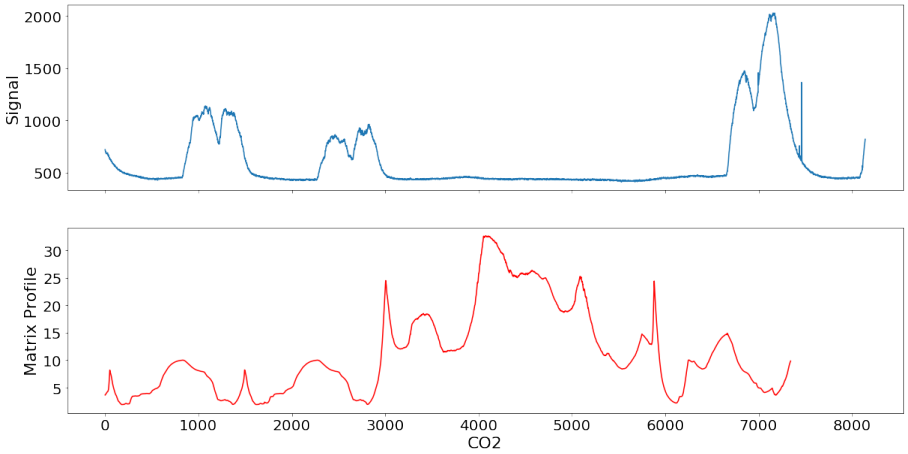
Questo task è stato svolto sul Training Set, che è stato rappresentato nei quattro grafici quì sotto ponendo sulle ascisse i 5 attributi presenti (in ordine da 0 a 4: ’Temperature’=0.0, ’Humidity’=1.0, ’Light’=2.0, ’CO2’=3.0, ’Humidity Ratio’=0.4) e sulle ordinate il range di valori per tutti questi attributi. Si sono rappresentate distintamente le istanze appartenenti alle due classi in output. Ogni istanza temporale è una linea nei grafici quì sotto. Nei due grafici di sinistra sono stati rappresentati tutti gli oggetti usando uno spessore piccolo per le linee, mentre nei due grafici di destra si è rappresentato un sample di 300 oggetti aumentando lo spessore delle linee.



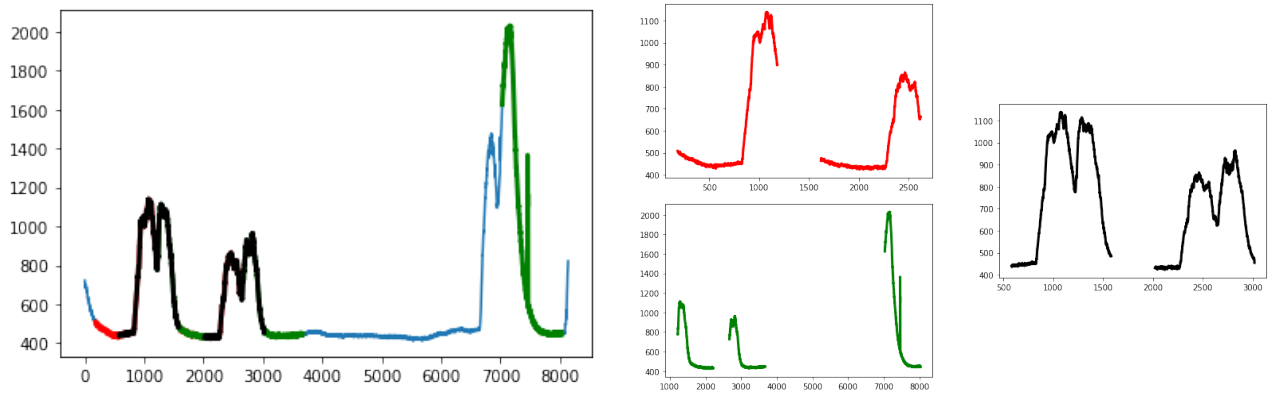
Shapelets. I grafici sopra ci permettono già di apprezzare la 'Shape' tipica dei valori del dataset per le due classi in output. I valori più elevati di 'Light' (2.0) e 'CO2' (3.0) sono presenti per la maggior parte nei grafici riferiti alla classe 'Occupancy'=1. Le Shapelet sono state estratte usando l'algoritmo 'Shapelet Transform', testando la 'Window size' da 2 a 5 (come criterio per il punteggio è stato scelto 'mutual information'). Come si nota dalla figura in basso a sinistra Le Shapelets risultanti, cioè le due sotto-sequenze temporali più rappresentative delle classi del dataset, partono dagli indici di riga 1170 e 1164. La parte di grafico più spessa rappresenta quella che più discrimina tra le classi in output. Nel grafico di sinistra si riporta dunque un dettaglio del medesimo grafico, riferito solo ai primi tre attributi. E' chiaro che è l'attributo 'Light' quello che apporta il maggior grado di differenza tra le Shapelets delle classi in output.



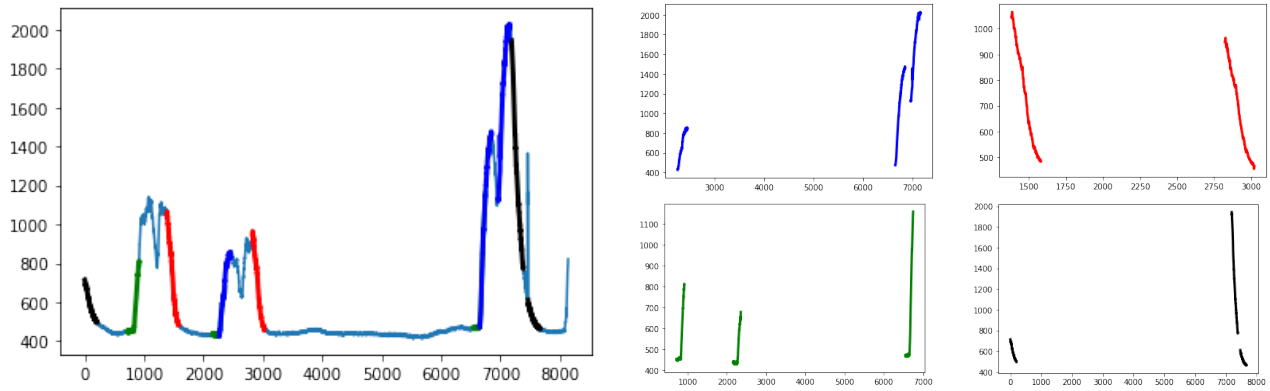
Motifs. Come attributo da analizzare è stato scelto 'CO2'. Sono stati generati i grafici della serie temporale (Signal) e Matrix Profile.



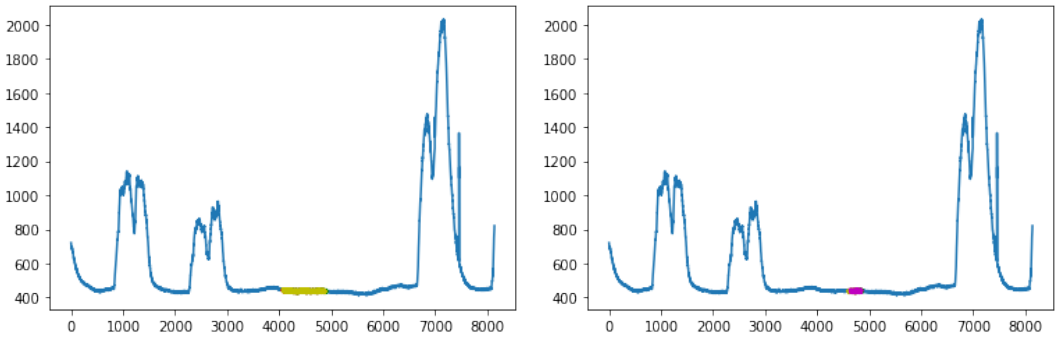
Dopo alcuni test un valore efficiente per l'Iper-Parametro 'Window Size' è stato 800. Questo valore permette di catturare solo le Anomalie e i Motifs interessanti, escludendo i dettagli irrilevanti (white noise). I picchi nella matrix profile coincidono con le anomalie mentre i valori vicini a 0 coincidono con i Motifs. Successivamente sono stati cercati i Motifs più frequenti, che si riportano nei grafici quì sotto.



Per confronto, si riportano anche i risultati ottenuti con una 'Window Size' di 200, che sono comunque interessanti.



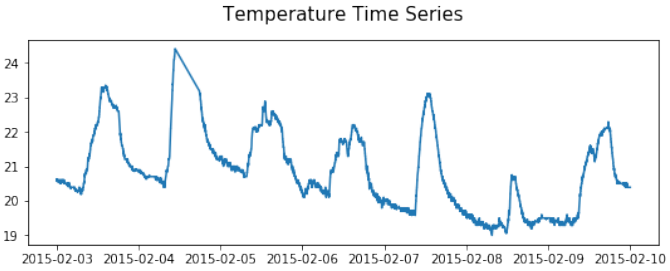
Le anomalie sono state estratte usando una 'Window Size' di 800 (figura in basso a sinistra) e una 'Window Size' di 200 (figura in basso a destra). È interessante osservare che l'area rimane la stessa (si contrae soltanto).



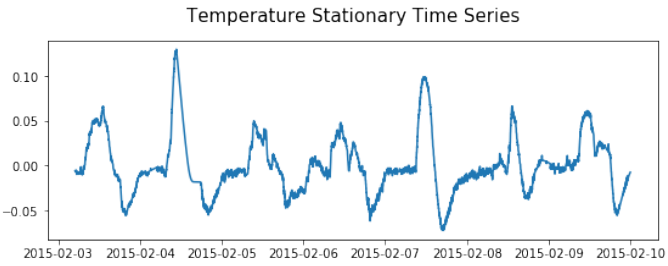
3.3 Forecasting

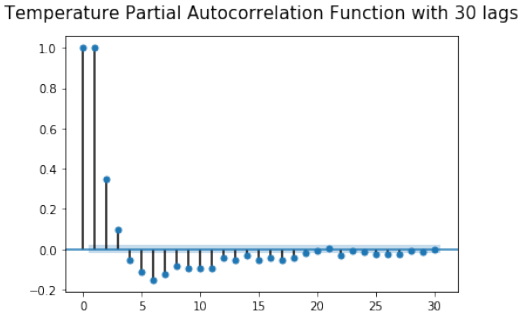
Questo task è stato eseguito sull'attributo 'Temperature'. La Time Series usata è stata generata unendo i valori dell'attributo nel Test Set 1 a quelli del Training Set (in quest'ordine per rispettare le precedenze temporali). Tra i due dataset intercorrono alcune ore, questi Missing Values sono stati riempiti con un'interpolazione (retta che collega l'ultimo valore del Test Set 1 al primo valore del Training Set). Inoltre sono stati eliminati i valori iniziali e finali in modo che la serie cominciasse e terminasse esattamente a mezzanotte (per 7 giorni, dal 2015-02-03 al 2015-02-09 compreso). La serie temporale presenta un leggero trend discendente dall'inizio fino al giorno 2015-02-08. Successivamente sembra stabilizzarsi o addirittura cominciare un'inversione di trend. E' presente un'evidente stagionalità che si ripete ogni 1440 minuti, cioè

esattamente ogni giorno. Ciò è dovuto all'escursione termica tra il giorno e la notte (cosa che è evidente anche dal grafico).



La varianza della serie rimane abbastanza stabile al variare del tempo, ma per alcune giornate la differenza termica è importante, ad esempio tra il 2015-02-07 ed il 2015-02-08. Il Dickey-Fuller Test per la verifica della stazionarietà indica che la serie non è propriamente stazionaria: Test Statistic = -2.783; Critical Value (1%) = -3.43; Critical Value (5%) = -2.86; Critical Value (10%) = -2.57. E' stato quindi deciso di generare un'altra serie rendendo stazionaria questa sopra, al fine di usarla per i modelli che necessitino della stazionarietà. La varianza è stata ridotta mediante una trasformazione logaritmica della Time Series (ciascun valore diventa il logaritmo naturale del valore originale) ed il trend è stato eliminato sottraendo a ciascun valore la media mobile a 300 periodi (cioè il valore medio della temperatura nelle 5 ore precedenti). La nuova serie è stazionaria: il Test Statistics del Dickey-Fuller Test passa a -4.617, inferiore a tutti i Critical Value per ogni confidenza. Il grafico della Autocorrelation cala esponenzialmente mentre quello della Partial Autocorrelation ha i primi 4 valori rilevanti e poi si inverte. Da ciò si deduce che la Time Series potrebbe essere approssimata da un modello autoregressivo (AR) di ordine 4.

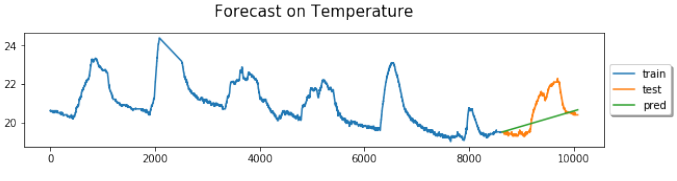




Per effettuare il forecasting la Time Series è stata suddivisa usando i valori dell'ultimo giorno (2015-02-09) come Test per i modelli applicati alla serie originale non stazionaria, e usando i valori dalle 10:00 dell'ultimo giorno fino alla fine come Test per i modelli applicati alla serie resa stazionaria.

Simple Exponential Smoothing: Essendo presente una chiara stagionalità il SES non può produrre una buona predizione. In ogni caso il modello è stato applicato sulla serie resa stazionaria facendo variare il valore dell'Iper-Parametro Alpha (lo Smoothing Level) nel range [0,1]. I risultati non variano molto, in generale la predizione graficamente risulta essere una semplice retta parallela, molto vicina alla media storica della TS. Infatti i Coefficienti di Determinazione sono vicini a 0 per ogni tentativo (un valore di R2 pari a 0 indica che il modello ha una Performance esattamente pari alla media).

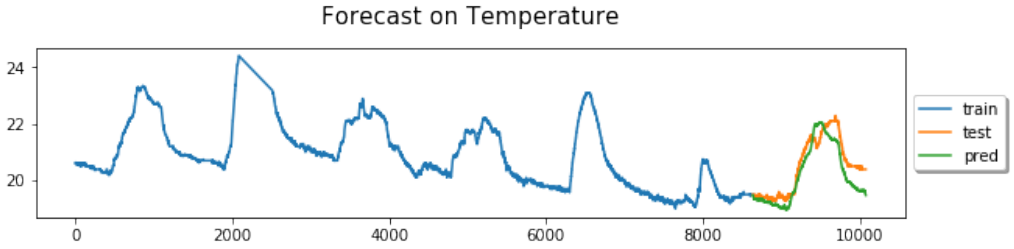
Holt's Linear Trend method:



Anche questo metodo non produce buone predizioni su dati con stagionalità ma permette Forecast su dati con trend, quindi è stato testato sulla Time Series originale non stazionaria. Con valori degli Iper-Parametri Smoothing Level = 0.01 e Smoothing Slope = 0.1 riesce a catturare l'inversione di trend nella parte finale della TS. Il coefficiente R2 è pari a 0.2786 ed il Mean Absolute Percentage Error (MAPE) è 0.0339.

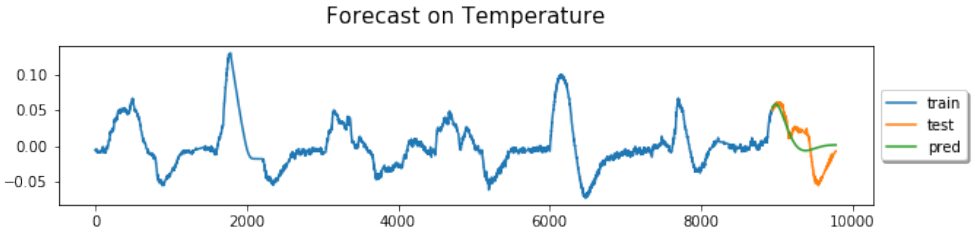
Holt Winter's Exponential Smoothing: Data l'accentuata stagionalità nei dati, con periodi giornalieri, è stato prodotto un modello basato su 1440 periodi (1 giorno in minuti), in modo da catturare il pattern tipico della temperatura giornaliera. I risultati dell'HWES sono i migliori tra i modelli testati sia sulla serie stazionaria che non (riportiamo quì quelli sulla serie originale, perché l'HWES è un modello che da solo riesce a catturare Trend e Seasonality).

R2	0.8012
MAPE	0.02
MAXAPE	0.047
TAPE	28.88
RMSE	0.477
MAE	0.407
MAD	0.375



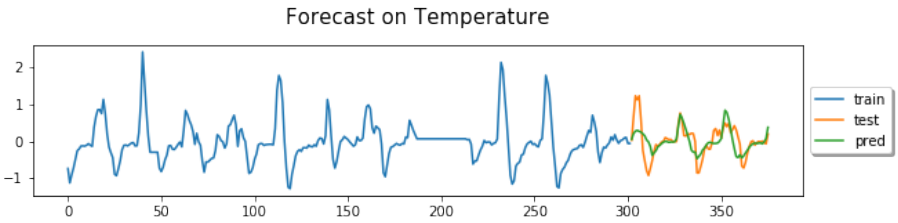
ARIMA: Come suggerito dal grafico della PACF il miglior modello ARIMA per i dati ha Autoregressione di ordine 4 e Media Mobile di ordine 1: ARIMA(4, 0, 1). Il modello ARIMA funziona solo su serie stazionarie (su cui appunto è stato testato) e non predice la stagionalità. In ogni caso esso predice discretamente l'inversione della temperatura dall'ora più calda dell'ultima giornata (le 10:00) in poi. In particolare il modello predice dove si sposterà la media della serie e con che velocità lo farà.

R2	0.5144
MAPE	35.814
MAXAPE	5569.45
TAPE	30083.50



SARIMAX: il SARIMAX non riesce a fare forecast sulle Time Series usate fin'ora perché implica una computazione troppo dispendiosa. Quindi invece di usare la Time Series con la frequenza originale (minuti) si è selezionato una frequenza di ore. Per ovviare alla presenza di poche istanze temporali si è unito non solo il primo Test Set prima del Training ma anche il secondo Test Set dopo il Training (sempre per rispettare le precedenze temporali), interpolandone i valori mancanti come fatto nel primo caso. Alla nuova Time Series è stato rimosso il Trend sottraendo a ciascun valore la media mobile a 5 periodi. Il Dickey-Fuller Test ne ha confermato la stazionarietà. E' stato usato un modello SARIMAX(4, 0, 1, 24), riprendendo cioè gli stessi Iper-Parametri del modello ARIMA e aggiungendovi una Seasonality di 24 periodi (dato che la frequenza adesso è di 1 ora la stagionalità giornalierà della temperatura si ripete ogni 24 ore). Riportiamo quì i risultati del test del modello sugli ultimi tre giorni esatti della time series.

R2	0.3692
MAPE	4.338
MAXAPE	73.911
TAPE	321.009
RMSE	0.3528



3.4 Time Series Classification

3.4.1 Shape and Structural based Classifiers

La Classification basata su Shaplet e Features è stata condotta sia con il dataset normalizzato (con la Min Max Normalization) che con quello originale. I risultati migliori sono stati ottenuti senza normalizzazione, quindi vengono discussi quelli. I risultati sul Test Set 1 vengono riportati tutti nella tabella in basso.

Shapelet based Classifier: questo metodo esegue la classificazione basata sulla distanza tra le Time Series del dataset e le Shapelet che meglio discriminano nelle classi in Output. Il metodo 'grabocka' ha suggerito un numero di 4 dimensioni per eseguire la classificazione. Come 'optimizer' è stato usato Adagard, con 'learning rate'=0.01 e regolarizzione L2 di 0.1.

Shapelet Distances based Classifier: questo metodo invece trasforma Training e Test Set in array che hanno per valori le distanze tra le Time Series e le Shapelets. Il dataset così costruito poi è stato testato su tre diversi Classifier: Decision Tree (DT), KNN e Logistic Regression (LR).

Feature Based Classifier: Sono state calcolate diverse statistiche per ogni attributo, tipo Curtosi, Deviazione Standard, Media, Varianza, ecc. Queste statistiche sono state vettorizzate e usate come input per gli stessi tre classifier: Decision Tree, KNN e Logistic Regression.

Classifier/Metrics	Accuracy	F1 class 0	F1 class 1
Shapelet	0.72	0.82	0.40
Shapelet Distances based KNN	0.94	0.95	0.92
Shapelet Distanecs Based DT	0.91	0.93	0.86
Shapelet Distanecs Based LR	0.93	0.95	0.91
Structure (Feature) Based KNN	0.87	0.90	0.84
Structure (Feature) Based DT	0.98	0.98	0.97
Structure (Feature) Based LR	0.98	0.98	0.97

I risultati migliori sono quelli dei modelli Feature Based, ma comunque non superano le performance viste nei capitoli precedenti.

3.4.2 Representation based Classifiers

Recurrent Neural Network: l'architettura della LSTM utilizzata è stata la stessa fornita: in input un LSTM con due nodi, 5 RNN di tipo LSTM, 6 Fully Connected Hidden Layers e un Output Layer di due nodi (altre architetture sono state testate ma hanno generato tutte problemi). Il Layer in Output ha funzione di attivazione 'Sigmoid' mentre gli altri hanno tutti funzione di attivazione 'ReLU'. L'ottimizzatore usato è 'adam' e il parametro ottimizzato è l'Accuarcy, riportando anche la 'sparse categorical crossentropy' come Loss. Come dataset per il training è stato usato il Training Set senza l'attributo 'Humidity Ratio' e aggiungendo un nuovo attributo relativo all'ora, estrapolato dalla colonna delle date. Questo perché come si è notato dall'analisi delle serie temporali l'occupazione della stanza dipende in grossa misura dall'ora della giornata (tendenzialmente la stanza è occupata nelle ore diurne), quindi può essere utile aggiungere l'attributo 'hours' per facilitare l'individuazione di questa relazione da parte del classificatore. In ogni caso la LSTM dovrebbe riuscire da sola a discriminare per le classi in output anche sulla base dal trascorrere del tempo, perché lo State interno al modello contiene la memoria dei pesi delle iterazioni precedenti (questa è la differenza fondamentale per cui questi modelli costruiti appositamente per classificare le Time Series dovrebbero funzionare meglio dei classificatori tradizionali). La Batch Size selezionata è stata 50 ed il modello è stato istruito per 100 epoche, testato su entrambi i Test Set e validato in ogni epoca su un Validation Set ottenuto con il 30% del Training Set. I risultati sono in pratica gli stessi del miglior modello in assoluto ottenuto fin'ora sia in termini di metriche sui test che di AUC delle ROC Curve.

	Test Set 1	Test Set 2
Accuracy:	0.9782	0.9922
F1-score:	[0.9826, 0.9709]	[0.995, 0.9817]
Precision:	[1.00, 0.95]	[1.00, 0.97]
Recall:	[0.97, 1.00]	[0.99, 0.99]
AUC:	0.992	0.996

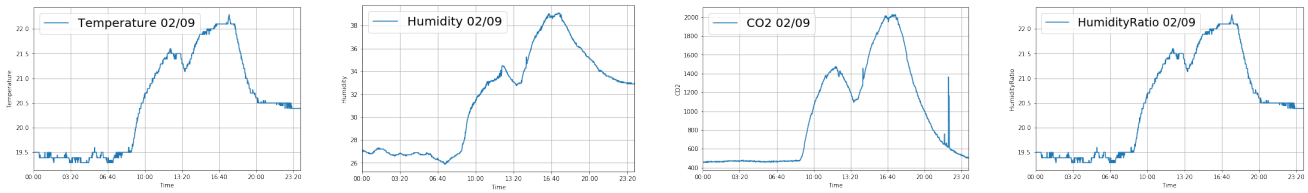
Lo stesso setup è stato eseguito anche istruendo il modello sul Training Set senza l'attributo 'Light'. Nonostante i risultati siano sensibilmente peggiorati, Accuracy(Test 1)=0.8964 e Accuracy(Test 2)=0.691, è utile tenere in considerazione il modello così istruito. Questo perché per tutti i modelli 'Light' è stato l'attributo che ha catalizzato un'importanza quasi totale nel predire la classe in output. Da un punto di vista di spiegazione dei risultati però ciò potrebbe essere fuorviante: l'andamento del valore delle variabili è per tutte molto simile e dunque sebbene 'Light' sia sicuramente quello che discrimina meglio per le classi in output è possibile che siano altri in realtà gli attributi che spiegano logicamente meglio la presenza di qualcuno nella stanza (es. le variazioni di 'CO2'). Infatti la luce è un attributo che, a differenza degli altri, varia normalmente durante la giornata a causa della luce esterna diurna (in concomitanza dell'occupazione della stanza) e non necessariamente perché qualcuno ne ha accesa una artificiale. Nelle maggior parte dei casi del dataset però questo ragionamento non vale perché l'intensità della luce varia velocemente e di un'entità rilevante, cosa che esclude che si possa trattare di luce naturale.

4 Task 4 - Sequential Pattern Mining

Per applicare il SPM abbiamo deciso di così modificare il traning set:

- Nel primo, abbiamo diviso il training per le diverse variabili (Temperature, Humidity...)e applicato la sax transformation scegliendo i valori dei parametri in maniera tale da ottenere una trasformazione il più fedele possibile alla serie temporale originale. Per poi creare le rispettive mappe dei simboli e ricavare le sequenze. Al termine abbiamo unito le 5 sequenze per ottenere un unico array.
- Nel secondo, abbiamo diviso il training per le giornate e anche questa volta abbiamo tenuto in considerazione solo le giornate i cui dati sono stati rilevati per l'intero arco. Successivamente abbiamo diviso le giornate per le variabili e applicato le dovute trasformazioni, per poi raggrupparle insieme ricreando le "giornate" iniziali, ottenendo così 5 "databases" formati da 5 sequenze.

Osservazioni: possiamo notare che in tutti i giorni l'andamento relativo all'Humidity e all'HumidityRatio sono molto simili, questo è naturale poiché il secondo è ricavato dall'umidità stessa. E' interessante notare un cosa strana che accade nel giorno 5, che corrisponde al lunedì, tutte le variabili tranne "Light" assumono un andamento davvero molto simile, come è possibile osservare nei grafici sottostanti. Presentano tutti un andamento quasi costante fino alle 9, poi cambia e diventa crescente, con un picco all'ora di pranzo (13:30). Questa somiglianza è possibile notarla in maniera così evidente solo il giorno 5. Questo ci dice che l'occupazione della stanza influenza fortemente tutte le variabili soprattutto il lunedì.



Passiamo ora all'analisi dei pattern. Abbiamo prima di tutto stabilito come frequenza minima 3. Per quanto riguarda la prima trasformazione le singole sequenze sono composte da 8143 elementi, per questa ragione, e per limitazioni dovute alla potenza dell'elaboratore, abbiamo deciso di analizzare a campioni di 100 o 1000 alla volte le diverse sequenze (Es: primi/ultimi e centrali 100/1000, o anche nel mezzo, selezionando i medesimi elementi per tutte). Alcune di queste combinazioni non ottenevano nessun pattern con

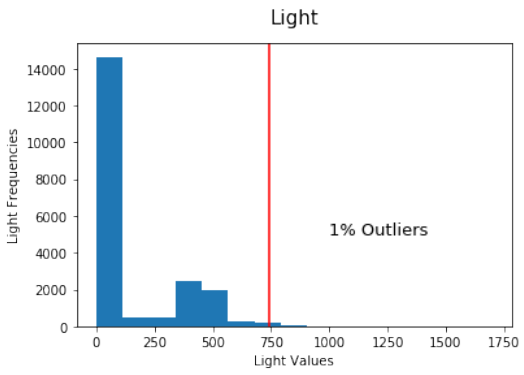
frequenza 3, alcuni neanche con frequenza 2, ma soprattutto nessuna di queste ha una frequenza maggiore a 3.
Essendo le sequenze formate da numeri ripetuti in successione, che può essere uno solo o più (es: 2222...000)a seconda di quanto è grande il campione selezionato, comporta che i pattern ricavati sono dello stesso tipo, in questo caso specifico sono una successione di un unico numero, generalmente composta fino a 25 elementi, quindi sono accettabili anche tutti i pattern che da esso contenuti. Non si è trovato nulla di interessante all'interno dei pattern.
Per quanto riguarda la seconda trasformazione invece, la singola sequenza è composta da 1440 elementi quindi abbiamo cercato le frequenze nei primi 500 elementi, i 500 successivi e i restanti 440. Qui riusciamo a trovare pattern con frequenze pari a 5 e anche composti dalla successione di non solo un numero ma di due o più numeri (ad esempio questo accade il giorno 4 nella selezione degli elementi [500:1000] nella quale una parte dei pattern riguarda la sequenza di 3 numeri: 5,2 e 0), cosa che non accadeva nelle prove precedenti. Visto che tutte le parti dei 5 giorni producono un grosso numero di pattern abbiamo deciso di alzare la frequenza minima a 4. Anche in questo caso il numero di pattern è elevato e solo in tre giornate abbiamo sezioni che non producono risultati(parte 1 dei giorni 1, 3 e 4, parte 3 giorno 3), qui sono ancora presenti pattern composti dalla successione di più numeri. Abbiamo deciso di incrementare la frequenza a 5. Se non contiamo i pattern contenuti da quello più grande possiamo dire si riducono a soli 6. Con una piccola tabella analizziamo i risultati ottenuti con questa min. freq., all'interno segneremo "x" nel caso in cui non producessero pattern, nel caso contrario abbiamo segnato il numero che viene ripetuto, fino anche ad ottenere una successione formata da più di 60 elementi.

Parte/**Giorno**	**G1**	**G2**	**G3**	**G4**	**G5**
0:500	x	1	x	x	0
501:1000	x	5	x	x	2
1001:1440	0	2 e 0	x	x	x

Possiamo notare che l'unico pattern che ricorre tra le giornate è quello composto dal numero 0. Per curiosità siamo andati a confrontarli con quelli ottenuti con la prima divisione. Troviamo ricorrente il pattern composto dalla successione di 1 sia in quelli ricercati con gli elementi [7143:8143] che [6000:7000]. Un altro che ritroviamo è il pattern composto dalla sequenza di 0, all'interno degli elementi [5000:5100]. In conclusione a seconda di come si modifica la serie temporale e a seconda degli elementi che si tengono in considerazione si possono perdere informazioni, come nella prima modifica effettuata nella quale non otteniamo pattern composti da più di un numero. I risultati ottenuti di per sé sono poco interessanti.

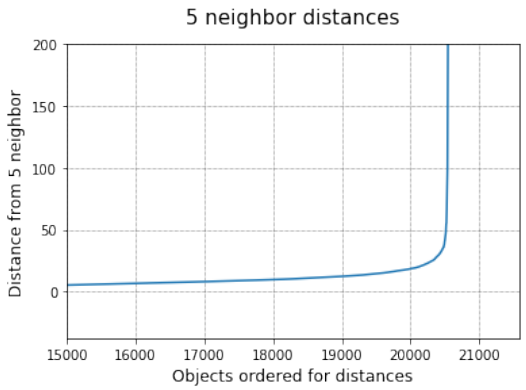
5 Task 5 - Outlier Detection

Sono stati uniti i due dataset di test al training set per generare un DataFrame più grande su cui trovare gli Outliers. Il nuovo dataset è costituito da 20560 oggetti.
BoxPlot:
I BoxPlot degli attributi 'Light' e 'CO2' sono già stati rappresentati nella fase iniziale di Data Understanding. Per l'attributo 'Light' il 99-esimo percentile è il valore 744. Cioè l'1% delle istanze del dataset, che corrispondono a 205 oggetti, ha un valore dell'attributo 'Light' superiore a 744 (questi sono l'1% degli Outliers). Non esistono invece Outliers per i valori bassi dell'attributo perché il valore più basso, che è 0, costituisce da solo il 62% dei valori dell'attributo. Questi Outliers discriminano molto bene nella classe in output: 202 di essi hanno valore di 'Occupancy' pari a 1 e solo 3 oggetti hanno un valore di 0. Questi 3 oggetti sono anomalie del datset: a fronte di valori di 'Light' intorno a 800 e

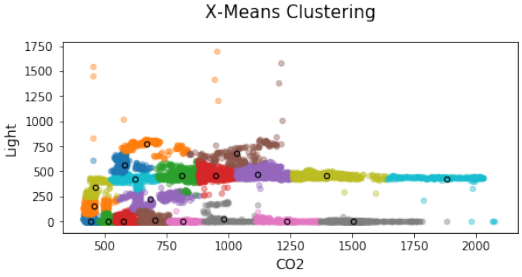


1500, che farebbero presupporre l'occupazione della stanza, questa in effetti non è occupata. Questi 3 oggetti si riferiscono a tre minuti consecutivi dalle 09:42:00 alle 09:44:00 del giorno 2015-02-07. I valori degli altri attributi sono tutti nella norma. Essendo pochi oggetti è possibile che si tratti di errori di rilevazione ma si è avanzata anche l'ipotesi che il soggetto si fosse temporaneamente assentato dalla stanza.

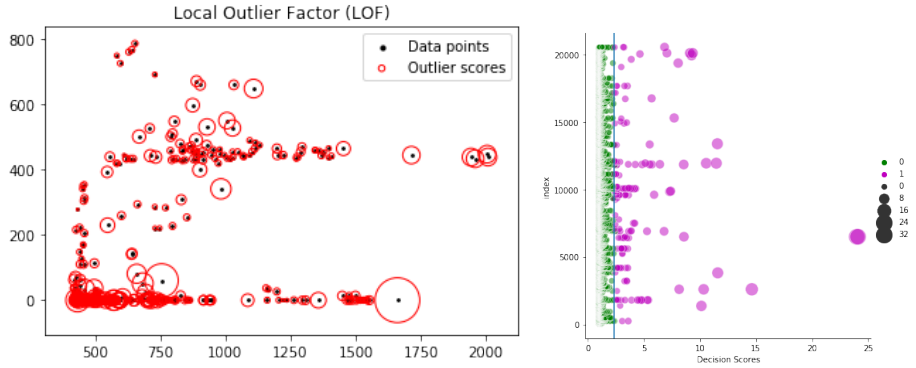
DBSCAN: Sono state calcolate le distanze di Manhattan di ciascun oggetto dal suo quinto punto più vicino ed è stato rappresentato il grafico di queste distanze ordinate. Come si nota tutti gli oggetti con valori canonici si trovano ad una distanza tra 0 e 25 dal quinto oggetto più vicino, mentre quelli che si presuppone siano Outliers hanno una distanza maggiore.
Dunque il DBSCAN è stato eseguito con Iper-Parametri: 'min samples' = 5 ed 'eps' = 25. Gli Outliers sono 52 oggetti a cui è stata associata la label -1 dal modello. Di questi solo 12 hanno il valore di 'Light' maggiore di 744, che è pari in media ad un valore molto più alto, 1230. Questo significa che 193 dei 205 Outliers specifici del singolo attributo 'Light' in realtà non sono Outliers di tutto il dataset, perché gli Outliers effettivi sono quelli che hanno congiuntamente 'CO2' e 'Light' elevati. Infatti i 52 Outliers ottenuti con il DBSCAN hanno valori medi nella norma per tutti gli attributi tranne che per 'Light' e soprattutto per 'CO2', pari rispettivamente a 554.25 e 1026. Questo risultato può essere dedotto anche dalle distribuzioni degli attributi: 'CO2' e 'Light' sono i due attributi che più di tutti presentano un marcato addensamento di frequenze nella parte bassa delle distribuzioni e pochissimi oggetti per i valori alti, che vanno a costituire appunto gli Outliers. Questi Outliers non discriminano tra le classi in output, 27 hanno classe 0 e 25 hanno classe 1.



X-Means:
Sono state eseguite diverse prove dell'algoritmo X-Means (tecnica di Clustering avanzato e non di Outlier Detection). I risultati sono diversi ad ogni ogni prova a causa della caratteristica randomica dell'algoritmo, ma il risultato in generale è quello riportato in figura. Questo metodo non rileva prettamente gli Outliers. Vengono solo associati cluster diversi da quelli principali agli oggetti con valori elevati di 'Light' e 'CO2'.

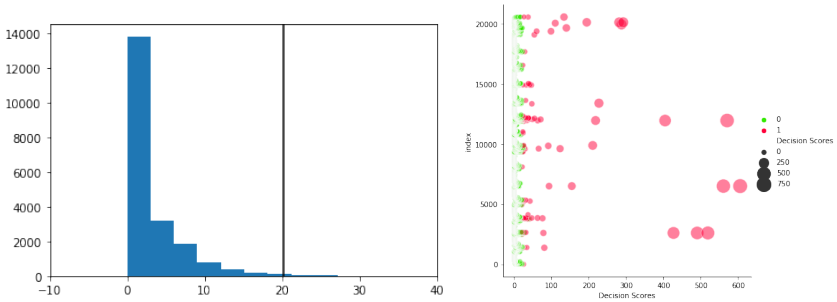


LOF: Il metodo LOF è stato applicato con entrambe le librerie Scikit Learn e PyOD. Con SKLearn è stato settato l'Iper-Parametro 'n neighbors' pari a 5. Il metodo restituisce 1871 Outliers. L'Outlierness Score in questo caso è il 'negative outlier factor' cioè il LOF in negativo, quindi più basso è questo Score e maggiore è l'Outlierness del punto. Infatti la media del Negative LOF Score degli Outliers è circa 11 volte inferiore alla media del Negative LOF Score di tutto il dataset. Nella figura in basso a sinistra è stato rappresentato lo scatterplot degli attributi 'Light' e 'CO2' di un sample di 500 oggetti del datset, accerchiati da una circonferenza con raggio proporzionale al LOF Score assegnato dal modello. E' interessante notare, come ovviamente ci si attende, che anche punti vicini ad altri punti in realtà possono avere un Outlierness elevata se le densità intorno ai vicini è molto superiore alla densità intorno al punto considerato. Con la libreria PyOD il livello di LOF che discrimina tra Inliers ed Outliers è 2.344 (gli Outliers hanno un LOF superiore a questo Thershold). Il LOF più basso ottenuto è 0.9229 e quello più alto è 24.1017. La linea verticale nella figura a destra marchia il confine fra Outliers e Inliers.

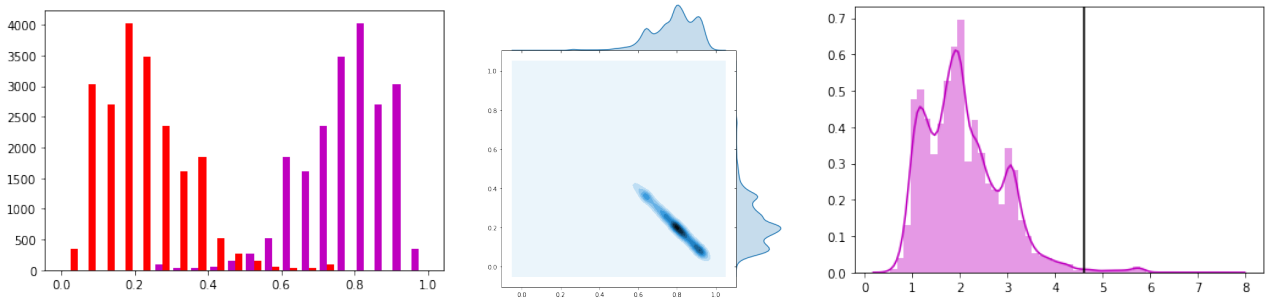


ABOD. Sono stati fatti diversi esperimenti e alla fine è stato scelto 20 come valore dell'Iper-Parametro 'neighbor'. L'algoritmo ha identificato 20365 Inlier e 195 Outlier, assegnando un punteggio di Outlierness ad ogni punto (oggetti con punteggi alti sono considerati Outliers). Il punteggio assegnato agli oggetti del dataset varia nell'intervallo fra -8433405.81 e -1.987e-12 ed il punteggio sopra cui i valori sono considerati Outliers è -2.7557e-06.

KNN. Come per il DBSCAN sono state calcolate le distanze di ciascun punto dal proprio 5-NN. Gli Outliers sono l'1% dei punti con la distanza più elevata dal proprio vicino. La maggioranza dei punti (99,99%) ha un valore della distanza vicino a 0 mentre la minoranza ha distanze più alte di 600. Il dataset è stato normalizzato per facilitare l'interpretazione visuale nella figura in basso a destra. Quì gli Outliers sono le frequenze a destra della linea verticale. Nell'altra figura riportiamo una rappresentazione più chiara che individua il valore 20.20 come threshold di distanza dal 5-NN sopra il quale un oggetto viene considerato Outlier (punti rossi). Nell'asse y abbiamo i valori degli attributi e la larghezza dei cerchi rappresenta il punteggio di ciascun punto.

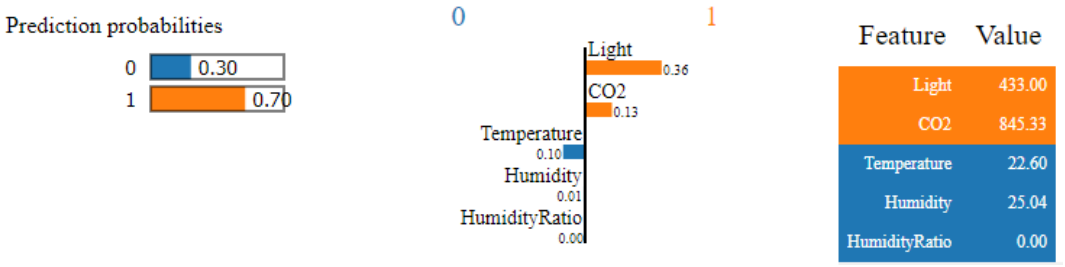


AutoEncoder. E' stato costruito un modello con 5 Hidden Layer con numero di nodi [5, 3, 2, 3, 5] e funzione di attivazione 'relu'. per l'Output Layer è stata usata una funzione di attivazione 'sigmoid'. Il modello è stato istruito con 10 epoche. Nel grafico quì sotto a sinistra sono state rappresentate le distribuzioni di frequenza della probabilità che ciascun punto sia un Outlier (istogramma viola) e la probabilità inversa, cioè quella che un certo oggetto sia un Inlier (istogramma rosso). Quindi ad esempio la frequenza più alta, circa 4000, in corrispondenza del valore 0.2 significa che 4000 oggetti circa hanno una probabilità del 20% di essere Outlier. E' interessante notare lo Spike di 2000 oggetti con probabilità 40% di essere Outliers. Le due distribuzioni sono ovviamente correlate (come si vede dalla figura centrale). La figura a destra invece è la distribuzione di probabilità del punteggio di Outlierness assegnato dal modello. I due grafici sono ovviamente simili perché la probabilità di essere Outlier dipende direttamente dal punteggio di Outlierness assegnato dal modello.



6 Explainability and Conclusions

E' stata usato il metodo 'LIME' per spiegare la classificazione effettuata su un oggetto casuale del dataset (quello all'indice 190). Come classificatore è stata scelta la logistic regression, in quanto modello più stabile.



In figura possiamo osservare i risultati del metodo. A destra abbiamo le probabilità che l'oggetto sia di classe 0 (30%) o di classe sia 1 (70%). Al centro abbiamo la divisione degli attributi tra quelli che prevedono una classe 1 per l'oggetto ('Light' e 'CO2') e gli altri che prevedono una classe 0. Quì ad ogni attributo è anche associato il valore della sua Feature Importance nella classificazione. In fine, l'ultima figura rappresenta i valori che assume l'istanza in questione per ogni attributo.

Possiamo concludere che è quasi certo che la stanza in questione sia un ufficio di lavoro. Il soggetto (o anche soggetti) vi è presente solo nelle ore diurne. Entra verso le 07:40 nei primi giorni dal 2015-02-02 al 2015-02-06 e verso le 08:40 nei giorni seguenti fino al 2015-02-17. nei giorni 2015-02-07, 2015-02-08, 2015-02-14 e 2015-02-15 nessuno è presente mai nella stanza perché si tratta di Weekend e l'ufficio è ovviamente chiuso. Anche nei fine settimana quando nessuno è presente c'è una minima quantità di luce nella stanza (arriva in genere ad un massimo intorno a 300 nel mezzo del giorno) e questo fa supporre che sia presente una finestra da cui entra un po' di luce esterna, cosa che comunque è stata rilevata dagli algoritmi di Machine Learning che hanno imposto livelli superiori di luce come indicatore di occupazione della stanza. All'incirca dalle 13:00 alle 13:40 di tutti i giorni lavorativi i valori dei vari attributi hanno un brusco calo (la luce in particolare) cosa che può indicare che il soggetto stacca da lavoro per la pausa pranzo. Infine quasi sempre alle 18:05 esatte dei giorni lavorativi il soggetto finisce di lavorare. In tutti i momenti iniziali e finali della giornata, quando il soggetto entra ed esce definitivamente dalla stanza, la luce passa immediatamente da 0 ad un valore introno a 420 circa, questo fa supporre che la luce artificiale della stanza da sola, quando fuori è buio e la luce esterna non contribuisce, porti a questo valore della luce, successivamente durante la giornata la luce si intensifica.