

CORSO DI LAUREA MAGISTRALE IN GEOFISICA DI ESPLORAZIONE E  
APPLICATA

TESI DI LAUREA

CONFRONTO DI ALGORITMI DI MACHINE LEARNING NELLA CLASSIFICAZIONE  
DI LITHO-FLUID FACIES DA DATI SISMICI A RIFLESSIONE

CANDIDATO:

**Carniani Davide**

RELATORE:

**Mazzotti Alfredo**

CORRELATORE:

**Aleardi Mattia**

CONTRORELATORE:

**Virgilio Michele**

ANNO ACCADEMICO 2018/2019

# Indice

<b>Indice</b>	<b>1</b>
<b>Lista delle Tabelle</b>	<b>7</b>
<b>Lista delle Figure</b>	<b>9</b>
<b>1 Riassunto</b>	<b>17</b>
<b>2 Introduzione</b>	<b>21</b>
<b>3 Teoria</b>	<b>25</b>
3.1 Il problema del Learning . . . . .	25
3.1.1 Problemi di classificazione e regressione . . . . .	26
3.1.2 La funzione di loss e la nozione di Rischio . . . . .	26
3.2 Trasformazioni dei dati . . . . .	27
3.2.1 Standardizzazione . . . . .	27
3.3 Tecniche di resampling . . . . .	28
3.3.1 K-fold Cross Validation . . . . .	29
3.3.2 Grid-Search Cross Validation . . . . .	30
3.3.3 Cross Validation più Test . . . . .	30
3.3.4 Formalizzazione . . . . .	31
3.4 Algoritmi di Supervised Learning . . . . .	33
3.4.1 Support Vector Classifier . . . . .	33
3.4.2 Logistic Regression . . . . .	37
3.4.3 Neural networks . . . . .	39
3.4.4 K Nearest Neighbors . . . . .	42
3.4.5 Random Forest . . . . .	45
3.5 Valutazione del modello . . . . .	48
3.5.1 Confusion Matrix . . . . .	48
3.5.2 Metriche ricavate dalla Confusion Matrix . . . . .	50

3.5.3	Metriche specifiche per le classi . . . . .	51
3.6	Inversione Bayesiana Amplitude Versus Offset . . . . .	52
3.6.1	Il modello a priori . . . . .	53
3.6.2	Il forward modeling sismico . . . . .	55
3.6.3	Il modello a posteriori . . . . .	56
3.7	Relazioni tra variabili elastiche e litho-fluidfacies . . . . .	57
<b>4</b>	<b>Indagine preliminare</b>	<b>61</b>
4.1	Comprensione del dato . . . . .	61
4.2	Effetto della Normalizzazione . . . . .	66
4.3	Iper-parametri . . . . .	69
4.3.1	Algoritmi Lineari . . . . .	70
4.3.2	Neural Network . . . . .	71
4.3.3	Random Forest . . . . .	72
4.3.4	K Nearest Neighbors . . . . .	73
<b>5</b>	<b>Classificazione con Training Set alla scala dei log</b>	<b>75</b>
5.1	Griglia degli iper-parametri . . . . .	78
5.2	Scelta del modello . . . . .	78
5.3	Classificazione su dati del pozzo blind . . . . .	83
5.4	Inversione e classificazione dati sismici sintetici ricavati dal pozzo blind . . . . .	86
5.4.1	Inversione parametrizzata nelle velocità e classificazione dati sismici sintetici ricavati dal pozzo blind . . . . .	87
5.4.2	Inversione AVO parametrizzata nelle impedenze e nella densità e classificazione dati sismici sintetici ricavati dal pozzo blind . . . . .	91
<b>6</b>	<b>Classificazione con Training Set alla scala sismica</b>	<b>95</b>
6.1	Scelta del modello . . . . .	97
6.2	Classificazione da risultato inversione AVO su dati sismici da pozzo blind . . . . .	102
<b>7</b>	<b>Classificazione del dato reale</b>	<b>105</b>
7.1	Scelta del modello . . . . .	105
7.2	Stima dei parametri elasticci . . . . .	109
7.2.1	Stima dell'ondina sorgente con il metodo statistico . . . . .	110
7.2.2	Inversione AVO . . . . .	112
7.3	Risultati . . . . .	114
<b>8</b>	<b>Conclusioni</b>	<b>117</b>
<b>9</b>	<b>Prospettive future</b>	<b>121</b>

<i>INDICE</i>	3
<b>A</b>	<b>123</b>
A.1 Glossario . . . . .	123
A.2 Metodo Statisico per la stima dell'ondina . . . . .	124
A.3 Bootstrap . . . . .	126
A.4 Programma Model Evaluation . . . . .	126
<b>Bibliografia</b>	<b>129</b>



# Liste degli Algoritmi

1	K fold Cross Validation . . . . .	30
2	Random Forest . . . . .	48



# Liste delle Tabelle

4.1	Parametri statiscici delle Shale . . . . .	64
4.2	Parametri statiscici delle Brine . . . . .	64
4.3	Parametri statiscici delle Gas . . . . .	64
4.4	Risultati di una K-fold Cross Validation effettuata su dati non normalizzati e normalizzati, con diversi algoritmi di ML. . . . .	68
4.5	Coefficienti per Logistic Regression per due valori di C. . . . .	71
4.6	Coefficienti per Linear SVC per due valori di C. . . . .	71
5.1	Proporzioni delle classi di litho-fluidfacies nei due set espressi in percentuale.	76
5.2	Griglia degli iper-parametri di tuning . . . . .	78
5.3	Valori degli iper-parametri per i migliori modelli degli algoritmi . . . . .	79
5.4	Valori di media e deviazione standard di Accuratezza Bilanciata calcolata sulle performance nelle folds singole per ogni modello di ML tunato con la Cross Validation . . . . .	82
5.5	Metriche per le singole classi di Precision, Recall e F1-score . . . . .	83
5.6	Metriche per le singole classi di Precision, Recall e F1-score . . . . .	85
5.7	Metriche per le singole classi di Precision, Recall e F1-score . . . . .	90
6.1	Valori degli iper-parametri per i migliori modelli degli algoritmi . . . . .	98
6.2	Valori di media e deviazione standard di Accuratezza Bilanciata calcolata sulle performance nelle folds singole per ogni modello di ML tunato con la Cross Validation . . . . .	100
6.3	Metriche per le singole classi di Precision, Recall e F1-score . . . . .	101
7.1	Valori degli iper-parametri per i migliori modelli degli algoritmi . . . . .	106
7.2	Valori di media e deviazione standard di Accuratezza Bilanciata calcolata sulle performance nelle folds singole per ogni modello di ML tunato con la Cross Validation . . . . .	108
7.3	Metriche per le singole classi di Precision, Recall e F1-score . . . . .	109



# Liste delle Figure

3.1	Divisione grafica dell'intero dataset utilizzando la Hold out. . . . .	28
3.2	Visualizzazione grafica della k-fold Cross Validation. L'intero set, diviso in 5 folds, viene utilizzato per allenare e testare l'algoritmo attraverso una serie di iterazioni pari al numero delle cartelle da utilizzare come validation. . . . .	29
3.3	Schema generale dell'applicazione del Cross Validation più test per un algoritmo di learning . . . . .	31
3.4	Un problema di classificazione binario in due dimensioni. A sinistra: Osservazioni di due classi differenti distinte dal colore che possono essere divisi senza errore da più iperpiani. A destra: l'iperpiano scelto dal classificatore con massimo margine . . . . .	33
3.5	Un problema di classificazione costituito da due classi rappresentate da differenti colori. $\mathbf{x}_1$ e $\mathbf{x}_2$ rappresentano le variabili utilizzate per predire la classe di output. I dati sono stati predetti da un classificatore con massimo margine. La linea continua è l'iperpiano che divide il risultato della predizione, mentre le linee tratteggiate rappresentano il margine. Si noti come il margine sia definito dalla distanza tra i support vectors e l'iperpiano. . . . .	34
3.6	In questo caso campioni appartenenti a due classi differenti distinte dal colore, rappresentati dai numeri nel piano definito dai parametri $\mathbf{x}_1$ e $\mathbf{x}_2$ , sono stati predetti utilizzando un SVC. Si noti che il problema non è linearmente risolvibile, infatti i campioni 12 e 11 sono classificati erroneamente. In questo caso i support vectors sono i campioni 12, 2, 1, 8, 11, 9 e 7. . . . .	34
3.7	Un SVC è stata allenata sui dati di training con differenti valori di C. In alto a sinistra i valori di C sono i più piccoli, mentre il valore più alto è usato in basso a destra. . . . .	35

3.8 Confronto tra probabilità stimate con LR e linear regression a partire da un esempio in cui si cerca di predire la probabilità di default di un debitore utilizzando come predittore il suo bilancio economico $Pr(Default = 1 Bilancio)$ . I campioni sono categorizzati come 0 (non default) e 1 (default avvenuto) giacciono nelle linee tratteggiate. Le linee blu rappresentano le stime di probabilità che un input appartenga rispettivamente alla classe 1 o 0. A sinistra è mostrata la stima della probabilità con l'utilizzo della linear regression, mentre a destra la stessa operazione è stata fatta con logistic regression. . . . .	37
3.9 Diagramma che descrive il funzionamento di un Multi Layer Perceptron. Le $\times$ indicano i prodotti tra tra matrici. La matrice di input NxP dove N è il numero dei campioni e P il numero di features, viene in primo luogo moltiplicata con la matrice dell'Hidden Layer. L è un numero usualmente definito dall'utente e rappresenta il numero di neuroni dell'hidden layer. Il risultato di questa moltiplicazione viene passato a una funzione $g(x)$ e passato all'output layer. Una volta applicata la funzione di output $f(x)$ si ottiene l'output del NN. Per un NN adibito a un problema multiclass, le colonne dell'Output Layer sono pari al numero di classi Q. . . . .	39
3.10 Forma della funzione di cross-Entropy per un problema binario $y = \{0, 1\}$ . Sull'asse delle ascisse P1 rappresenta la probabilità $P(Y = 1 X = x)$ , mentre su quello delle ordinate il valore della Cross Entropy. . . . .	41
3.11 L'approccio KNN usato con $K = 3$ su un problema di classificazione con due classi, sei arancioni e sei blu. A sinistra: il punto da predire è rappresentato da una croce. I tre punti più vicini all'osservazione di test sono identificati e la classe viene predetta come blu. A destra: I confini per questo esempio sono mostrati in nero. I grid blu e arancioni mostrano le regioni in cui è predetta la rispettiva classe. . . . .	42
3.12 Esempio di un DT che divide lo spazio delle features in 5 regioni. Si tratta di un problema in due dimensioni dove $\mathbf{x}_1$ e $\mathbf{x}_2$ compongono lo spazio delle variabili. La prima segmentazione coinvolge $\mathbf{x}_1$ e genera due regioni, le quali vengono divise ulteriormente formando le regioni $R_1, R_2, R_3, R_4, R_5$ , che sono anche chiamate anche foglie del DT. Ad ogni segmentazione si accompagna una regola, data dal parametro scelto e dal suo valore di segmentazione $t_i$ . . . . .	45
3.13 Spazio di $\mathbf{x}_1$ e $\mathbf{x}_2$ diviso dall'albero in Figura 3.13. I valori $t_i$ sono i valori da cui partono le linee che dividono il dataset nelle regioni $R_j$ . . . . .	46
3.14 La funzione di Gini per un problema in due classi. . . . .	47
3.15 Esempio di Confusion Matrix per un problema di classificazione binario. . . . .	49

3.16 Esempio di Confusion Matrix per un problema di classificazione binario con classi sbilanciate a favore della classe Positiva. . . . .	51
3.17 Valori di densità e $V_p$ in relazione alla saturazione in gas. . . . .	58
3.18 Variazione con la profondità della porosità per campioni di Sand e Shale pure. A sinistra gli stati diagenetici che si verificano a una certa profondità. Dal momento che entrambe le litologie possono variare notevolmente a causa della composizione, della texture, e dal gradiente di pressione non è stata assegnata alcuna scala agli assi. . . . .	59
4.1 In figura sono mostrati i dati relativi a un pozzo dell'area di studio. Sono mostrati i valori delle 3 proprietà elastiche e delle 3 petrofisiche. Queste sono associate a un asse d i profondità differente. . . . .	62
4.2 In figura è mostrato un grafico a barre che mostra il numero di campioni per ogni classe sull'intero dataset a disposizione. Le Shale contribuiscono con 5987 campioni, le Brine Sand con 736 campioni mentre le Gas Sand sono 324 campioni. . . . .	62
4.3 Visualizzazione in un piano $I_p$ $V_p/V_s$ dei campioni riguardanti tutti i log di pozzo disponibili . . . . .	63
4.4 Funzioni di densità di probabilità ottenuti con l'algoritmo kernel density per i valori normalizzati di $V_p$ , $\rho$ e $V_s$ di tutto il dataset proveniente dai parametri elasticci dei 5 pozzi nella zona di interesse. In blu è rappresentata la funzione di probabilità delle Shale, in rosso quella delle Brine Sand e in Verde quella delle Gas Sand. . . . .	65
4.5 Valori di importanza normalizzata con l'uso di un modello di RF allenato su tutto il dataset proveniente dai parametri elasticci dei 5 pozzi nella zona di interesse con 100 alberi . . . . .	66
4.6 Grafici a barre dei valori dei coefficienti relativi ai parametri usati nel training. A destra i valori dei coefficienti prima della normalizzazione, a destra dopo la normalizzazione. Partendo dall'alto al basso gli algoritmi utilizzati sono: LR con $C = 1$ , SVC con $C = 1$ e infine NN con $\alpha = 0.001$ . . . . .	68
4.7 Matrice di correlazione tra le variabili di partenza calcolate con il metodo di Pearson. . . . .	69
4.8 Boundaries per due modelli di Logistic Regression allenati con due differenti fattori inversi di regolarizzazione utilizzando come features $V_p$ e $\rho$ . . . . .	71
4.9 Boundaries per due modelli di NN allenati con su due differenti fattori di regolarizzazione utilizzando come funzione di attivazione Rectified Linear Unit e come features $V_p$ e $\rho$ . . . . .	72

4.10 Boundaries per tre modelli di Random Forest allenati al variare degli alberi e della loro profondità, utilizzando come features $V_p$ e $\rho$ . . . . .	73
4.11 Boundaries per due modelli di NN allenati con due differenti valori di vicini utilizzando come parametri $V_p$ e $\rho$ . . . . .	73
5.1 Flowchart delle operazioni svolte: Il dato di test è stato diviso dal dato di training attraverso stratificazione. Su quest'ultimo è stato svolto il training e il tuning degli algoritmi, da cui si deriverà un miglior modello di ML utilizzato per la predizione delle classi sul Test Set e su un pozzo blind. . . . .	76
5.2 Funzioni di densità di probabilità ottenute con l'algoritmo kernel density per valori normalizzati di $V_p$ , $\rho$ e $V_s$ differenziate per i dati di training e di test. Le distribuzioni sono normalizzate per gli stessi valori di media e deviazione standard. Le linee blu, rosse e verdi sono indicano rispettivamente Shale, Brine Sands e Gas Sands. . . . .	77
5.3 Confusion Matrices ricavate dalla somma delle performance su ogni fold della Cross Validation. All'interno sono mostrati i valori normalizzati per il numero dei campioni per ogni classe. Sopra a ogni CM è riportata la percentuale di campioni correttamente classificati. . . . .	80
5.4 Box-plot dei valori di Accuratezza Bilanciata derivati delle performance sulle 10 cartelle della Cross Validation per le migliori combinazioni di parametri di ogni algoritmo. Il Box-plot riporta in arancione la mediana, il box rappresenta la zona dal primo al terzo quartile, mentre le linee arrivano fino ai valori minimi e massimi. I cerchi rappresentano gli outliers. . . . .	81
5.5 Confusion Matrix ricavata dal Test Set, i cui valori sono normalizzati per il numero di campioni in ogni classe. . . . .	83
5.6 Funzioni di densità di probabilità ottenute con l'algoritmo kernel per i valori normalizzati di $V_p$ , $\rho$ e $V_s$ . I valori di media e deviazione standard sono calcolati sul dato di training e applicati al log di prova. Le linee blu, rosse e verdi sono indicano rispettivamente Shale, Brine e Gas Sand. . . . .	84
5.7 Confusion Matrix calcolata sul log usato come log blind, non presente nei dati di training. In alto è presente il numero di campioni riconosciuti sul totale .	85
5.8 Confronto tra dati osservati e predetti per i dati provenienti dal pozzo blind.	86
5.9 Risultati inversione Bayesiana AVO sul sismogramma generato dal pozzo blind. La curva nera indica i valori veri, quella rossa la Maximum A Posteriori(MAP) e quella verde il modello low-frequency. Inoltre le curve grigie sono altre realizzazioni della distribuzione a posteriori $p(m d)$ , mentre quelle azzurre sono le realizzazioni del modello a priori $p(m)$ . . . . .	88

5.10 Funzioni di densità di probabilità per i valori normalizzati di $V_p$ , $\rho$ e $V_s$ normalizzati utilizzando la deviazione standard e la media calcolati sul Training Set. La prima riga riguarda le funzioni di densità dei valori dei parametri elastici usati per allenare il modello di KNN, la seconda illustra quelle generate dalla MAP mentre l'ultima riga illustra le funzioni di densità di probabilità ottenute dal log originale. Le linee blu, rosse e verdi sono indicano rispettivamente Shale, Brine Sands e Gas Sands. . . . .	89
5.11 Confusion Matrix relativa alle predizioni sui parametri elastici ottenuti con la map dell'inversione AVO . . . . .	90
5.12 Confronto tra fluid-lithofacies vere, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(MAP), che su tre realizzazioni della distribuzione a posteriori $p(m d)$ . . . . .	91
5.13 Risultati inversione Bayesiana AVO parametrizzata nelle impedenze sul sismogramma ricavato dal pozzo blind. La curva nera indica i valori veri, quella rossa la Maximum A Posteriori(MAP) e quella verde il modello low-frequency. Inoltre le curve grigie sono altre realizzazioni della distribuzione a posteriori $p(m d)$ , mentre quelle azzurre sono le realizzazioni del modello a priori $p(m)$ . . . . .	92
5.14 Confusion Matrix relativa alle predizioni sui parametri elastici ottenuti con la map dell'inversione AVO parametrizzata nelle impedenze acustiche. . . . .	93
5.15 Confronto tra fluid-lithofacies vere, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(map), che su tre realizzazioni della distribuzione a posteriori $p(m d)$ ottenute con un inversione AVO parametrizzata nelle impedenze acustiche. . . . .	94
6.1 Visualizzazione in un piano $I_p$ $V_p/V_s$ dei campioni riguardanti tutti i pozzi, escluso il blind, ricampionati alla massima frequenza di campionamento, filtrati con un filtro passa basso a 250Hz e ricampionati a 2ms. . . . .	96
6.2 Funzioni di densità per i valori di $V_p$ , $\rho$ e $V_s$ normalizzati a partire da deviazione standard e media calcolati sul Training Set. In questo plot sono confrontate le funzioni di densità dei dati alla scala dei log, in alto, e quelli dei dati alla scala sismica dopo il ricampionamento. Le linee blu, rosse e verdi sono indicano rispettivamente Shale, Brine e Gas Sand. . . . .	97
6.3 Confusion Matrices ricavate dalla somma delle perfomance su ogni fold della Cross Validation. All'interno sono mostrati i valori normalizzati per il numero dei campioni per ogni classe. Sopra a ogni CM è riportata la percentuale di campioni correttamente classificati. . . . .	99

6.4	Box-plot dei valori di Accuratezza Bilanciata derivati delle performance sulle 10 cartelle della Cross Validation per le migliori combinazioni di parametri di ogni algoritmo. Il Box-plot riporta in arancione la mediana, il box rappresenta la zona dal primo al terzo quartile, mentre le linee arrivano fino ai valori minimi e massimi. I cerchi rappresentano gli outliers. . . . .	100
6.5	Confusion Matrix ricavata dal Test Set, i cui valori sono normalizzati per il numero di campioni in ogni classe. . . . .	101
6.6	Confronto tra dati osservati, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(MAP), che su tre realizzazioni della distribuzione a posteriori $p(m d)$ ottenute con un inversione AVO parametrizzata nelle velocità.	102
6.7	Confronto tra dati osservati, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(MAP), che su tre realizzazioni della distribuzione a posteriori $p(m d)$ ottenute con un inversione AVO parametrizzata nelle impedanze acustiche. . . . .	103
6.8	Confusion Matrices del modello di KNN allenato alla frequenza di log e a scala sismica, testato nella classificazione delle map di . . . . .	104
7.1	Confusion Matrices ricavate dalla somma delle performance su ogni fold della Cross Validation. All'interno sono mostrati i valori normalizzati per il numero dei campioni per ogni classe. Sopra a ogni CM è riportata la percentuale di campioni correttamente classificati. . . . .	107
7.2	Box-plot dei valori di Accuratezza Bilanciata derivati delle performance sulle 10 cartelle della Cross Validation per le migliori combinazioni di parametri di ogni algoritmo. Il Box-plot riporta in arancione la mediana, il box rappresenta la zona dal primo al terzo quartile, mentre le linee arrivano fino ai valori minimi e massimi. I cerchi rappresentano gli outliers. . . . .	108
7.3	Confusion Matrix ricavata dalla classificazione sul Test Set, i cui valori sono normalizzati per il numero di campioni in ogni classe. . . . .	109
7.4	Stack lungo la in-line invertita. Le due linee gialle rappresentano la posizione di due pozzi localizzati lungo la sezione in-line. . . . .	110
7.5	In alto a sinistra l'autocorrelazione di una traccia sismica presa a un'angolo di 15 gradi, in alto a sinistra la autocorrelazione dopo l'applicazione di una finestra di Bartlett. In basso gli spettri di ampiezza prima e dopo dell'applicazione della finestra. . . . .	111

7.6	A sinistra ondina ottenuta dalla trasformata inversa della radice quadrata dello spettro di ampiezza della traccia sismica presa a 15 gradi. A destra la stessa ondina dopo l'applicazione di un filtro passa banda con frequenze di taglio pari a 15/45Hz e di una finestra Gaussiana con deviazione standard uguale a 17 nell'asse dei campioni. . . . .	112
7.7	Illustrazione delle ordine stimate a seconda dell'angolo di incidenza del raggio sismico. . . . .	112
7.8	Visualizzazione grafica dei valori dei parametri elastici ricavati dalla map dell'inversione AVO sul dato reale. . . . .	113
7.9	Funzioni di densità dei parametri elastici ricavati dalla map dell'inversione AVO sul dato reale (In blu) confrontate con le funzioni di densità dei parametri elastici del dato di training (In rosso). . . . .	114
7.10	Immagine che illustra le classificazioni effettuate dal miglior modello di KNN sulla soluzione map dell'inversione AVO sul dato sismico del dato reale. Il reservoir è indicato da una freccia rossa. . . . .	114
7.11	Immagine che illustra le classificazioni effettuate dal miglior modello di KNN sulla soluzione map dell'inversione AVO sul dato sismico del dato reale. Il reservoir è indicato da una freccia rossa. . . . .	115
7.12	Confronto tra litho-lithofacies vere corrispondenti ai pozzi passanti per le Xline 177 e 198, confrontati con le rispettive litho-fluidfacies classificate dal modello di KNN dalla soluzione map dell'inversione AVO sul dato reale. . . . .	116
A.1	Visualizzazione grafica del Bootstrap: da un set di dati iniziali vengono estratti 3 set con lo stesso numero di campioni, ma alcuni di essi presenti nel dataset originale si ripetono. . . . .	126
A.2	Diagramma di flusso del programma Model Evaluation. . . . .	127
A.3	Struttura del foglio di testo degli input da cui il programma legge il dataset iniziale. . . . .	127



# Capitolo 1

## Riassunto

L’obiettivo di questo lavoro di tesi è quello di testare diversi algoritmi di Machine Learning (ML) nell’identificazione delle litho-fluidfacies da dati di pozzo e di sismica a riflessione. Gli algoritmi sono stati allenati sui valori dei parametri elastici di cinque log di pozzo acquisiti in una determinata area di studio, ed è stata verificata la loro applicabilità su dati sismici sintetici e reali. I parametri elastici ricavati dai log e usati per allenare gli algoritmi sono: la velocità delle onde compressionali  $V_p$ , la velocità delle onde di taglio  $V_s$  e la densità  $\rho$ . Gli algoritmi testati in questo lavoro di tesi sono cinque: Neural Network, Logistic Regression, Support Vector Classifier, K Nearest Neighbors e Random Forest. Facendo l’assunzione che esista una funzione che lega i parametri elastici con le litho-fluidfacies, gli algoritmi di ML cercano di riprodurre questa funzione; Logistic Regression e Support Vector Classifier la approssimano con una funzione lineare, mentre i restanti algoritmi sono in grado di approssimarla con una non lineare. Il lavoro di tesi inizia con la definizione delle litho-fluidfacies a partire dai valori delle proprietà petrofisiche contenute nei well-log, ovvero la porosità  $\phi$ , la saturazione in acqua  $S_w$  e l’argilosità  $Sh$ . I campioni dei log con valori di  $\phi$  uguali o inferiori a 0.1 e con valori di  $Sh$  maggiori o uguali di 0.5 saranno assegnati alla classe delle Shale, tutti i restanti saranno definiti in base a  $S_w$ : per valori di  $S_w$  minori o uguali di 0.5 i campioni apparterranno alla classe delle Gas Sands, altrimenti saranno classificati come Brine Sands.

Una volta che ogni campione dei log è stato assegnato a una classe si sono condotte tre indagini preliminari. La prima di esse mira a studiare in che modo variazioni sui parametri elastici influenzino la classificazione delle litho-fluidfacies. La seconda indagine si concentra sull’effetto della normalizzazione dei dati sul training degli algoritmi. In questo caso sono stati aggiunti nella fase di training degli algoritmi 6 parametri costituiti da valori random con distribuzione Gaussiana, ed è stata valutata l’importanza relativa di tutti i parametri

nella classificazione sia prima che dopo la normalizzazione dei dati. Infine la terza indagine studia l'impatto degli iper-parametri degli algoritmi sulle funzioni di classificazione che essi restituiscono. Gli iper-parametri sono quei valori che un algoritmo richiede come input per poter effettuare il training.

Nella seconda parte del lavoro di tesi i dati di quattro log di pozzo saranno divisi in due set differenti detti Training Set e Test Set. Il primo set è stato utilizzato per allenare gli algoritmi di ML utilizzando una tecnica chiamata K-Fold Cross Validation. Questo metodo consiste nel dividere il Training Set in  $K$  parti contenenti lo stesso numero di campioni, chiamate cartelle. A turno ciascuna di queste cartelle viene usata per validare il modello di ML prodotto dall'allenamento sulle altre  $K-1$  cartelle. La media delle performances di un modello di ML sulle  $K$  cartelle produce un valore più robusto rispetto a un'unica prova, e questo valore può essere confrontato con quello ricavato da differenti modelli per effettuare una comparazione. Come performance degli algoritmi si valuta la media delle predizioni corrette su ogni classe, chiamata Accuratezza Bilanciata. Durante questo procedimento è stato inoltre eseguito il tuning, ovvero la scelta degli iper-parametri degli algoritmi, con una tecnica chiamata Grid-Search. Il Grid-Search consiste nel testare su ogni cartella lo stesso algoritmo più volte, variando gli iper-parametri con cui si effettua il training e selezionando quelli che restituiscono un miglior valore delle performance sulla media di ogni cartella. Al termine della Cross Validation e del Grid-Search il modello di ML con il più alto valore di Accuratezza Bilanciata è stato utilizzato per predire le litho-fluidfacies dei campioni del Test Set e di un pozzo blind tenuto da parte. Il pozzo blind inoltre è stato usato per produrre un sismogramma sintetico, da cui sono state stimate le proprietà elastiche usando un' inversione Amplitude Versus Offset(AVO), che sfrutta la dipendenza dei coefficienti di riflessione al variare dell'angolo di incidenza di un raggio sismico con l'interfaccia riflettente. Il miglior modello di ML è stato utilizzato per predire le litho-fluidfacies delle proprietà elastiche ricavate dall'inversione AVO.

Nell'ultima parte del lavoro di tesi è stato ripetuto ancora il processo di selezione del modello di ML facendo uso questa volta dei dati provenienti da tutti e cinque i log di pozzo. Scelto il miglior modello di ML esso è stato utilizzato per predire le litho-fluidfacies di una sezione inline estratta da un volume sismico terrestre. In questo caso, per stimare le proprietà elastiche del dato sismico è stato necessario stimare l'ondina sorgente, la quale è stata utilizzata come input nell'inversione AVO. Per svolgere questo compito sono stati applicati due metodi: stima dell'ondina tramite Singular Value Decomposition(SVD) troncata e tramite autocorrelazione delle tracce sismiche. Tra le varie ondine stimate è stata selezionata quella che ha restituito una stima più credibile.

Dall'analisi svolta sui log nella prima fase del lavoro è emerso che esiste un overlap tra le classi nello spazio delle proprietà elastiche e che quindi gli algoritmi non saranno mai in grado di predire correttamente il 100% dei campioni. Si è osservato come la normalizzazione giochi un ruolo fondamentale nel corretto allenamento degli algoritmi e che gli iper-parametri di tuning regolano la complessità delle funzioni di classificazione restituite da essi. I vari test sui dati di log dimostreranno che le Brine Sands sono risultate la classe più difficile da identificare soprattutto per i modelli lineari, i quali non essendo in grado di produrre boundary complessi hanno sofferto dell'overlap esistente tra classi. Ad ogni Cross Validation sono state illustrate e commentate le Confusion Matrices dei modelli insieme ai valori ricavabili da esse. Le classificazioni sul pozzo blind hanno mostrato risultati differenti da quelli del Test Set a causa della differenza nelle funzioni di densità di probabilità delle proprietà elastiche tra i due set. Il risultato dell'inversione AVO sul sintetico prodotto dal pozzo blind sarà di più difficile classificazione rispetto al pozzo blind originale, a causa della perdita di risoluzione nello spazio dei parametri elastici causata dall'effetto filtro dell'onda sorgente. Durante tutte le prove svolte il rank dei migliori algoritmi non è variato, suggerendo una consistenza dei risultati: il miglior modello di ML stimato dalla Cross Validation è stato sempre KNN. Per quanto riguarda l'inversione del dato reale, nonostante il basso rapporto segnale rumore che lo caratterizza, il miglior modello di ML è stato comunque in grado di predire con buona affidabilità i livelli di interesse saturi a gas.



# Capitolo 2

## Introduzione

Negli ultimi anni il Machine Learning(ML) ha riscosso un successo notevole, trovando applicazione nei più svariati ambiti tra cui la genetica, la medicina, l'economia e le scienze sociali. Il ML è un settore dell'Intelligenza Artificiale che dà al computer l'abilità di imparare senza essere stato esplicitamente programmato [1]. La popolarità del ML risiede nell'aumento della potenza computazionale e nel sempre maggior numero di dati a disposizione. Inoltre gli algoritmi di learning sono in grado di estrapolare relazioni più o meno complesse dai dati senza la necessità di programmarle esplicitamente: non è sempre richiesta conoscenza sulle vere relazioni esistenti in un problema. Anche in geofisica questi metodi sono sempre più studiati [2] [3].

Uno dei problemi geofisici che ha giovato dei metodi di apprendimento automatico è il problema della classificazione delle litho-fluidfacies sismiche, dove questi algoritmi possono giocare un ruolo di supporto per il geologo/geofisico. L'analisi delle litho-fluidfacies è un lavoro indispensabile nella correlazione stratigrafica e nelle analisi delle facies sedimentarie, permette di delimitare il reservoir e di individuare le zone di interesse nell'area di studio. Una facies sismica è intesa come un'unità sedimentaria a scala sismica caratterizzata dalla sua litologia,configurazione, petrografia e proprietà sismiche [4]. È importante specificare che la sismica non è influenzata in modo diretto dalle facies, ma dalle proprietà elastiche della sottosuperficie. L'utilizzo di metodi di ML in questo ambito è giustificato dal numero di dati, dalla loro qualità e dalle informazioni a priori a disposizione. Per allenare un algoritmo di ML a svolgere questo compito è essenziale assegnare ad ogni campione descritto dai parametri elasticci una litho-fluidfacies. La fonte di informazione che permette di fare l'associazione tra variabili elastiche e litho-fluidfacies è il log di pozzo. Il well-logging è una pratica comune quando le caratteristiche degli strati del sottosuolo devono essere studiati; attraverso questa pratica è possibile ottenere una descrizione dettagliata delle formazioni grazie alla misura di un gran numero di proprietà della sottosuperficie. Naturalmente perchè il log fornisca

un'informazione affidabile questo deve essere rappresentativo dell'area di studio. Se così non fosse i modelli di ML non sarebbero in grado di portare a termine il compito di classificazione in modo corretto: essi funzionano al meglio quando i dati su cui sono allenati provengono dalla stessa distribuzione di probabilità del dato da classificare. I dati di log forniscono informazioni puntuale; per estendere la predizione lateralmente è necessario utilizzare quelle provenienti da un dato sismico a riflessione da cui, tramite un procedimento di inversione AVO, è possibile stimare la velocità delle onde P  $V_p$ , la velocità delle onde S  $V_s$  e la densità  $\rho$ . I dati di log saranno usati per creare un miglior modello di ML e questo potrà essere usato per la classificazione delle facies in un dato sismico producendo risultati di supporto per gli operatori.

Dal momento che esistono più algoritmi in grado di eseguire questo compito è spesso utile confrontarli per selezionare il migliore in base a una certa performance definita a seconda degli scopi del lavoro. Gli algoritmi confrontati sono: Logistic Regression (LR), Support Vector Classifier (SVC), K Nearest Neighbors (KNN), Neural Network (NN) e Random Forest(RF). Nel corso del lavoro di tesi è stato commentato il confronto tra i modelli lineari (LR e SVC) e i modelli non lineari.

Dal momento che i dati forniti dai log non sono comparabili a quelli di un problema *Big – Data*, si farà uso della K-fold Cross Validation per sfruttare al meglio il dato e produrre stime delle performance affidabili. La Cross Validation non permetterà soltanto di scegliere il modello di ML, ma anche di effettuare il tuning. Ogni algoritmo funziona con dei parametri di input, chiamati iper-parametri. Il tuning è quel processo che si occupa di selezionare quelli più adatti al problema. Per effettuare il tuning sarà usato un algoritmo di ricerca "brute-force" chiamato Grid-Search. L'intera pipeline di selezione dell'algoritmo e di misura dell'errore di generalizzazione è stata programmata in linguaggio python 3.6 con l'ausilio della libreria scikit-learn ed è stato prodotto un file eseguibile utile a produrre un best model su altri problemi.

Nel capitolo successivo sono introdotte le basi di un problema di learning e le principali tecniche di ML utilizzate, seguite dalla teoria sulla inversione AVO e sulle relazioni tra parametri elastici e lithofacies.

Nel capitolo 4 è introdotto il dato proveniente dai log di pozzo e sono svolte le indagini preliminari sugli algoritmi di ML.

Nel capitolo 5 è stata svolta la Cross Validation sui dati di 4 pozzi alla scala dei log, scelto il miglior modello sono commentate le performance sul Test Set e sul pozzo blind prima e dopo l'inversione AVO.

Nel capitolo 6 lo stesso procedimento è stato svolto sui dati di training convertiti alla scala

sismica, confrontando i risultati prima e dopo il filtraggio dei dati di training.

Nel capitolo 7 è ripetuta la Cross Validation utilizzando tutti e 5 i log di pozzo, è stata stimata l'ondina del dato reale e effettuata l'inversione AVO per ricavare i parametri elastici che saranno classificati dal miglior modello ricavato.

Nei capitoli 8 e 9 sono presenti le conclusioni e sono discussi i possibili miglioramenti effettuabili sul lavoro di tesi. Il lavoro è completato in appendice dove vengono spiegate le tecniche utilizzate per la stima dell'ondina e il bootstrap. È inoltre presente una sezione riguardante il software implementato per il lavoro di tesi.



# Capitolo 3

## Teoria

### 3.1 Il problema del Learning

Il Machine Learning è definito come una classe di programmi o sistemi che costruiscono un modello predittivo dai dati di input. Questi sistemi usano il modello appreso per fare predizioni utili su nuovi dati, estratti dalla stessa distribuzione di probabilità di quelli usati per eseguire il training del modello [5]. Il problema generale del learning può essere descritto attraverso tre componenti:

- Uno spazio dei campioni  $\mathcal{X}$  con vettori random  $\mathbf{x} \in \mathbb{R}$  ottenuti da una distribuzione sconosciuta.
- Un label  $y \in \mathcal{Y}$  per ogni vettore  $\mathbf{x}$ .
- Un algoritmo di learning  $A$  che può implementare un set di funzioni  $f$  da una classe di funzioni  $\mathcal{F}$  sullo spazio dei campioni.

L'insieme dei vettori  $\mathbf{x}$ , di lunghezza  $P$ , forma una matrice degli input  $\mathbf{X}$ , composta da  $P$  colonne e  $N$  righe, dove  $P$  è pari al numero di variabili o features che descrivono i campioni, mentre  $N$  è pari al numero dei campioni.

Date le tre componenti il problema del learning è quello di scegliere il classificatore migliore dal set di funzioni  $\mathcal{F}$  che possono approssimare gli output. Supponendo quindi di osservare una risposta  $y$  a un input  $\mathbf{x}$ , si assume che esista una qualche relazione  $f$  che lega gli input agli output. In forma generale si può scrivere [6].

$$y = f(\mathbf{x}) + \epsilon \tag{3.1}$$

dove la componente  $\epsilon$  è chiamata errore irriducibile e descrive tutte le componenti che non sono note, come per esempio il rumore.

Si cerca quindi di stimare  $f$  tramite un set di osservazioni chiamato *training set* che vengono

passate a un algoritmo di training che a sua volta produce un output  $\hat{f}(\mathbf{x})$ . Nella maggior parte dei casi, benché  $\hat{f}(\mathbf{x})$  sia uguale a  $f(\mathbf{x})$ ,  $y$  sarà differente da  $\hat{y}$  a causa di  $\epsilon$ .

A livello pratico gli obiettivi finali di un problema di learning sono due: [7]:

1. La *scelta del modello* di ML: stimare le performance di differenti modelli allo scopo di selezionare il più adatto
2. La *stima del rischio*: una volta selezionato il modello di ML finale, stimare il suo errore di generalizzazione, o rischio, in nuovi dati.

Il tipo di problema di learning appena descritto è chiamato *Supervised* dal momento che il vettore dei label  $\mathbf{y}$  è disponibile. Tuttavia non sempre questo elemento è conosciuto dall'operatore. In tal caso, si parla di un problema *Unsupervised*, non supervisionato, dove si cerca di estrarre un pattern dai dati, osservandone caratteristiche comuni senza avere la possibilità di verificare la correttezza dei risultati.

Questo lavoro di tesi si concentra sul problema Supervised.

### 3.1.1 Problemi di classificazione e regressione

Le variabili che descrivono i campioni possono essere caratterizzate da valori quantitativi o qualitativi. Le features quantitative sono composte da valori numerici, mentre per rappresentare parametri qualitativi si usa un set di valori discreti che rappresentano una categoria o classe. I problemi di regressione sono quei problemi per cui  $y$ , il target, è rappresentato da valori numerici, mentre i problemi di classificazione sono quelli in cui  $y$  è rappresentato da valori categorici. I problemi di classificazione con più di una classe sono detti problemi multi classe.

### 3.1.2 La funzione di loss e la nozione di Rischio

La scelta del miglior classificatore è spesso basata nella misura del rischio, che è un grado di disaccordo tra il label vero  $y$  di un vettore  $\mathbf{x}$  e quello assegnato dal classificatore  $f : \mathcal{X} \rightarrow \mathcal{Y}$ , denotato con  $f(\mathbf{x})$ .

Una funzione di *loss* è una misura quantitativa della differenza tra  $y$  e  $f(\mathbf{x})$  e si denota con  $L(y, f(\mathbf{x}))$ , che restituisce una misura quantitativa della differenza tra  $y$  e  $f(\mathbf{x})$ . Il *rischio atteso* per un classificatore  $f$  è definito come:

$$R(f) = \int L(y, f(\mathbf{x})) dD(\mathbf{x}, y) \quad (3.2)$$

Che rappresenta il valore della funzione  $L(y, f(\mathbf{x}))$  su tutta la distribuzione di probabilità reale  $D(\mathbf{x}, y)$ . La distribuzione di probabilità reale  $D(\mathbf{x}, y)$  è sconosciuta.

Utilizzando la uno-zero loss  $L(y, f(\mathbf{x})) = 1$  quando  $y \neq f(\mathbf{x})$  e 0 altrimenti, si può scrivere il rischio atteso come:

$$R(f) = Pr(y \neq f(\mathbf{x})) \quad (3.3)$$

Dove  $Pr$  rappresenta la probabilità che la classe predetta dal classificatore  $f$ ,  $f(\mathbf{x})$  sia diversa da quella vera  $y$ .

E' da specificare che la uno-zero loss non è l'unica misura per esprimere il rischio, e che questa è definita dall'utente.

In generale il rischio atteso per un classificatore non è stimabile senza la conoscenza della distribuzione di probabilità reale  $D(\mathbf{x}, y)$ . Si prova ad ovviare a questo problema attraverso il calcolo del *rischio empirico*  $R_s(f)$ . Per un classificatore  $f = A(S)$  indotto dall'algoritmo  $A$  in un Training Set  $S$  di dimensione  $N$  il rischio empirico si calcola come :

$$R_s(f) = \frac{1}{N} \sum_{i=1}^N L(y_i, f(\mathbf{x}_i)) \quad (3.4)$$

Ovvero si calcola come la media della sommatoria della loss su tutti gli  $N$  dati del Training Set  $N$ . Dal momento che il rischio empirico del classificatore è usato per approssimare il rischio vero, lo scopo della stima dell'errore è di ottenere una misura non biasata del rischio empirico. Il Training Set fornisce una stima ottimisticamente biasata del rischio e non è quindi adatto a questo scopo. In seguito si illustreranno una serie di metodi per stimare il rischio empirico.

## 3.2 Trasformazioni dei dati

### 3.2.1 Standardizzazione

La standardizzazione di un dataset è un processo di normalizzazione richiesto per alcuni algoritmi di Machine Learning implementati in scikit-learn; le performance di tali algoritmi potrebbero essere non ottimali se le features individuali non assomigliassero a dati con una distribuzione normale standard: Gaussiane con media zero e varianza unitaria. [8]

Nella pratica la forma della distribuzione è ignorata e i dati vengono solo centrati sottraendo la media e dividendo per la deviazione standard.

$$x_t = \frac{x - ave(x)}{\sigma(x)} \quad (3.5)$$

Dove la media e la varianza sono sempre calcolati sui dati utilizzati per il training.

Molti elementi usati nelle funzioni oggetto di un algoritmo di learning, come per esempio i termini di regolarizzazione dei modelli lineari, assumono che tutte le features siano centrate intorno lo zero ed abbiano varianza nello stesso ordine. Se una feature possiede una varianza maggiore in termini di magnitudo delle altre, potrebbe dominare la funzione oggetto e rendere gli algoritmi non capaci di convergere alla soluzione ideale.

In definitiva la normalizzazione è eseguita allo scopo di:

1. Eliminare i fattori di scala
2. Fare in modo che i fattori di regolarizzazione degli algoritmi parametrici agiscano nel modo corretto

### 3.3 Tecniche di resampling

Per portare a termine gli step di un problema di learning è necessario ottenere una stima non biasata del rischio empirico. Per ottenere questa stima non è possibile utilizzare l'errore sul Training Set, il quale fornisce un errore ottimisticamente biasato. Per ottenere una stima non biasata l'intero dato deve essere diviso in subset. Un metodo utilizzato quando il numero di dati è sufficiente per ottenere una stima dell'errore rilevante è l'Hold out. L'Hold out consiste nel selezionare i campioni in modo random e dividere il dataset in tre set:

1. Il *Training* set, utilizzato per allenare il modello di ML.
2. Il *Validation* set, utilizzato per effettuare la scelta del modello di ML.
3. Il *Test* set, utilizzato per la stima del rischio.

La Hold out è illustrata in figura 3.1, solitamente il 60% del dato viene utilizzato per il training, mentre il restante 40% è diviso tra validation e test. Il dato di Training e quello di Validation vengono chiamati congiuntamente design set, dal momento che sono utilizzati per produrre il modello finale.

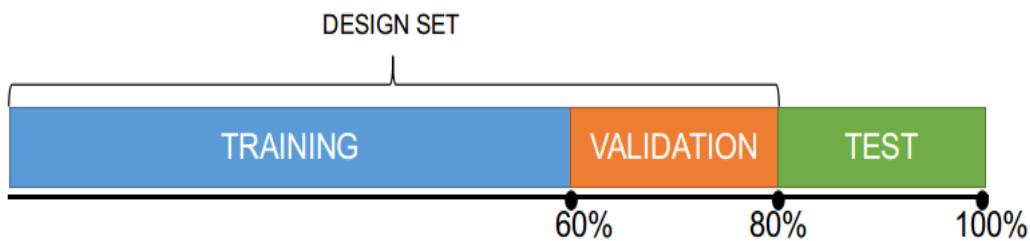


Figura 3.1: Divisione grafica dell'intero dataset utilizzando la Hold out.

Questo approccio richiede che le stime offerte dal dato di validation e di test siano statisticamente significative. Quando ciò non è possibile i metodi di resampling forniscono un buono strumento per avere una stima non biasata del rischio. Queste tecniche consistono nel selezionare ripetutamente un certo numero di campioni da un Training Set e allenare per ogni set un algoritmo [6].

Gli approcci di resampling possono essere computazionalmente costosi, dal momento che necessitano di allenare un algoritmo più di una volta.

### 3.3.1 K-fold Cross Validation

La K-fold Cross Validation si basa sulla divisione del dato di training in K gruppi, usualmente chiamati cartelle o folds. Ciclando su ciascuna delle folds, una di queste è usata per stimare l'errore di validation, mentre le altre K-1 folds sono usate per allenare l'algoritmo. Il processo è illustrato in Figura 3.2, dove la divisione in 5 cartelle necessita l'allenamento e la misura delle performance dello stesso algoritmo 5 volte. Si avranno dunque 5 differenti risultati che necessiteranno di essere mediati per produrre un risultato più robusto rispetto a una singola prova. La Cross Validation permette di sfruttare in modo ottimale i dati a disposizione, soprattutto quando essi non sono in un numero sufficiente per avere delle stime statisticamente rilevanti da un solo set di validation.

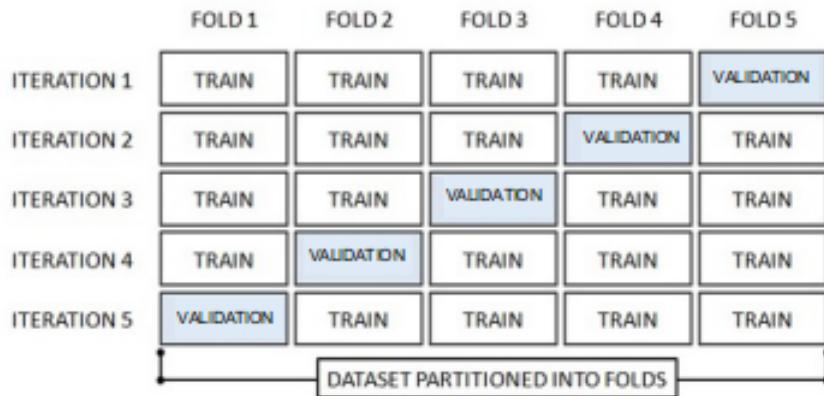


Figura 3.2: Visualizzazione grafica della k-fold Cross Validation. L'intero set, diviso in 5 folds, viene utilizzato per allenare e testare l'algoritmo attraverso una serie di iterazioni pari al numero delle cartelle da utilizzare come validation.

L'algoritmo 1 spiega le fasi del processo: la prima operazione consiste nello scegliere le cartelle. Spesso le cartelle sono composte prelevando campioni random dal dataset, ma esistono diversi criteri di scelta dei campioni.

---

**Algorithm 1** K fold Cross Validation

---

```

1: procedure
2:   Dividere il dato in K subset.
3:   for  $i$  in  $K$  do
4:     Train su  $S_{K-i}$ 
5:     Test su  $V_i$ 
6:      $Score_i \leftarrow L(V_i)$ 
7:   Calcolare  $std(Score)$ 
8:   Calcolare  $ave(Score)$ 

```

---

Dove  $S$  e  $V$  sono rispettivamente i dati di training e di validation. Con  $L(V_i)$  si intende la funzione di loss calcolata sul set di validation  $V_i$ , che viene assegnata alla variabile  $Score$ . Quando si utilizza la Cross Validation con un set di dati sbilanciati, ovvero con squilibri nelle dimensioni dei campioni appartenenti alle varie classi, si esegue la procedura di *stratification*. Infatti la ripartizione random dei campioni nelle folds potrebbe portare a folds non contenenti campioni di una classe poco rappresentata. La stratification consiste nel prendere nota delle rappresentazioni di ogni classe nell' intero dataset e assicurarsi che questa rappresentazione sia mantenuta sia nell folds e sia nel Test Set.

### 3.3.2 Grid-Search Cross Validation

Una parte importante del processo di selezione del modello di ML è il *tuning*, che riguarda l'ottimizzazione dei parametri  $\theta$  degli algoritmi, chiamati anche iper-parametri. Il tuning passa spesso per una procedura brute-force chiamata Grid-Search. Si tratta di un metodo empirico che, iterativamente, seleziona tutte le combinazioni di iper-parametri da un grid definito dall'operatore e misura le performance dell'algoritmo allenato con tali combinazioni di iper-parametri attraverso la Cross Validation. Su ogni cartella di test vengono misurate le performance utilizzando tutte le possibili combinazioni degli iper-parametri dell'algoritmo. Il termine performance si riferisce alla particolare metrica applicata dall'operatore per misurare la qualità delle predizioni del classificatore.

### 3.3.3 Cross Validation più Test

Il dato di training è utilizzato nella Cross Validation in combinazione con il Grid-Search per effettuare la selezione del modello di classificazione, mentre il Test Set è utilizzato per stima del rischio. I passi seguiti da questo metodo sono illustrati in figura 3.3. Per prima cosa si divide il dato nei set di Training e di Test[9]. Per un dato algoritmo di learning si esegue una procedura di tuning all'interno della Cross Validation attraverso il Grid-Search.

Una volta completato questo step si riallenna l'algoritmo con i migliori iper-parametri su tutto

il Training Set e si stima il rischio di questo modello di ML sul Test Set.

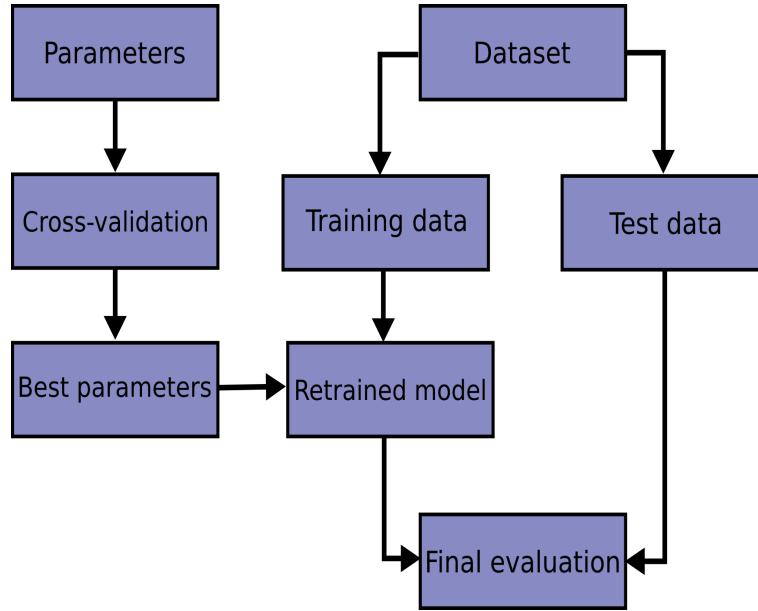


Figura 3.3: Schema generale dell'applicazione del Cross Validation più test per un algoritmo di learning

### 3.3.4 Formalizzazione

Richiamando il capitolo 3, le formule 3.2 e 3.4 sulla stima del rischio, per un dato numero di campioni distribuiti indipendentemente  $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) \dots (\mathbf{x}_n, y_n)$  appartenenti a un dato di training  $S$  con  $N$  campioni raccolto da una distribuzione  $D(\mathbf{x}, y)$ , il processo di learning cerca una funzione  $f$  nello spazio delle funzioni  $\mathcal{F}$ . Dopo il processo di training  $f$  sarà  $\hat{f}$  la migliore funzione che minimizzerà il rischio atteso calcolato come:

$$R(f) = \int L(y, f(\mathbf{x})) dD(\mathbf{x}, y) \quad (3.6)$$

Dove  $L$  è una funzione di loss. Per un problema di classificazione una tipica funzione di perdita è la zero-uno loss:

$$L((\mathbf{x}, y), f) = \begin{cases} 0 & \text{se } f(\mathbf{x}) = y \\ 1 & \text{Altrimenti} \end{cases} \quad (3.7)$$

Il problema principale è che  $D(\mathbf{x}, y)$  non è conosciuta. Si usa dunque il rischio empirico per approssimare il rischio stimato. Il rischio empirico è una stima unbiased del rischio, calcolata come

$$R_s(f) = \frac{1}{N} \sum_{i=1}^N L(y_i, f(\mathbf{x}_i)) \quad (3.8)$$

Il rischio è valutato sul Test Set. Di seguito una formulazione della Cross Validation e della stima del rischio sul Test Set:

1. Sono selezionate un set di funzioni con un set di iper-parametri  $\theta$ .
2. Il dataset  $S$  è diviso in dato di training  $S_{tr}$  e di test  $S_{te}$ .
3. Per ogni valore di  $\theta_m$  di  $\theta$ 
  - (a) stimare  $R_s(\hat{f}_{\theta_m})$  usando la Cross Validation
4. Selezionare  $\hat{\theta}_m = \min(R_s(\hat{f}_{\theta_m}(S_{tr})))$
5. Riallenare  $\hat{f}(S_{tr}) = \min(R_s(f, S_{tr}))$
6. Stimare  $R_s(\hat{f}(S_{tr}))$  con  $\frac{1}{N_{te}} \sum_{x_n \in S_{te}}^{N_{te}} L(x_n, \hat{f}(S_{tr}))$

## 3.4 Algoritmi di Supervised Learning

### 3.4.1 Support Vector Classifier

Support Vector Classifier (SVC) è un approccio per la classificazione sviluppato dalla community di computer science nel 1990, anno dal quale ha guadagnato progressivamente popolarità [6]. Si tratta di una generalizzazione di un classificatore semplice e intuitivo, chiamato *classificatore con massimo margine*, il quale ricerca tra gli infiniti iperpiani che possono dividere le classi, l'*iperpiano con massimo margine*, che è quello più lontano dalle osservazioni di training. Il problema è illustrato in Figura 3.4. In uno spazio p-dimensionale, un iperpiano è una superficie piatta di dimensione p-1.

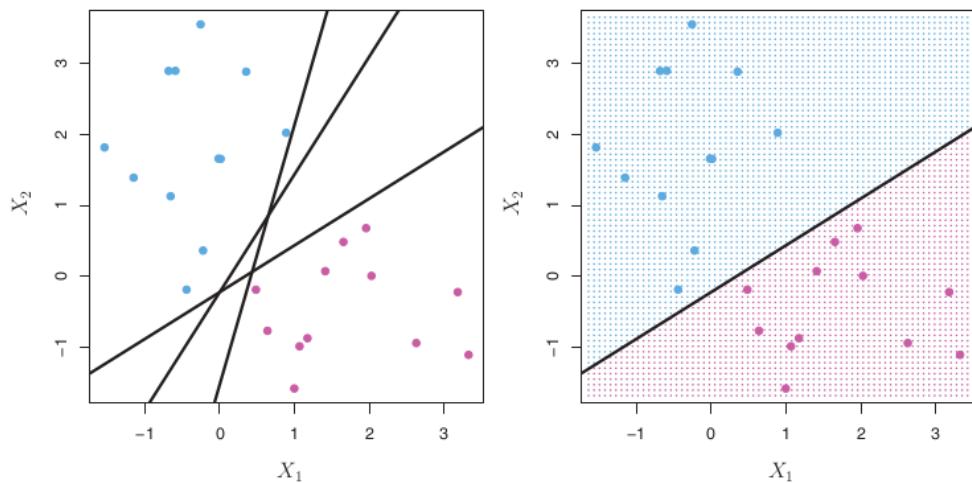


Figura 3.4: Un problema di classificazione binario in due dimensioni. A sinistra: Osservazioni di due classi differenti distinte dal colore che possono essere divisi senza errore da più iperpiani. A destra: l'iperpiano scelto dal classificatore con massimo margine

In Figura 3.5 si osserva come il margine sia la distanza tra la linea tratteggiata e l'iperpiano di divisione indicato dalle frecce. Le osservazioni che delimitano il margine sono chiamati *support vectors*, e sono gli unici campioni che hanno peso nel definire l'iperpiano finale.

Il classificatore con margine massimo non può risolvere problemi con dati non linearmente separabili, mentre la sua estensione, SVC risolve questo problema ed è adattata per trattare un range di applicazioni più ampio. Quando i dati non sono separabili da un iperpiano lineare, si lascia che il classificatore abbia un margine di errore, ovvero che classifichi erroneamente un certo numero di campioni. Il caso è mostrato in Figura 3.6, dove si osserva come all'interno del margine risiedano i campioni 1 e 8, mentre i campioni 11 e 12 siano classificati in modo scorretto.

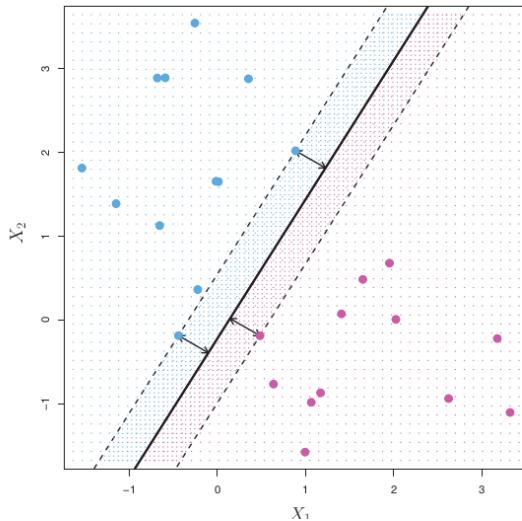


Figura 3.5: Un problema di classificazione costituito da due classi rappresentate da differenti colori.  $x_1$  e  $x_2$  rappresentano le variabili utilizzate per predire la classe di output. I dati sono stati predetti da un classificatore con massimo margine. La linea continua è l'iperpiano che divide il risultato della predizione, mentre le linee tratteggiate rappresentano il margine. Si noti come il margine sia definito dalla distanza tra i support vectors e l'iperpiano.

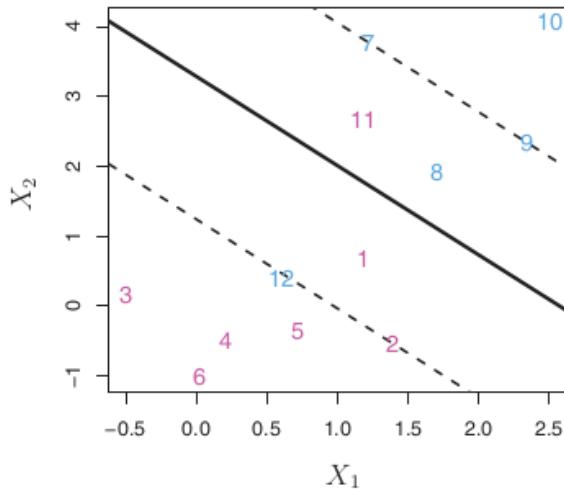


Figura 3.6: In questo caso campioni appartenenti a due classi differenti distinte dal colore, rappresentati dai numeri nel piano definito dai parametri  $x_1$  e  $x_2$ , sono stati predetti utilizzando un SVC. Si noti che il problema non è linearmente risolvibile, infatti i campioni 12 e 11 sono classificati erroneamente. In questo caso i support vectors sono i campioni 12, 2, 1, 8, 11, 9 e 7.

Dati dei vettori di training  $\mathbf{x}_i \in R^p, i = 1, 2, ..N$  vettore degli output  $\mathbf{y}$  nella forma  $y_i = \{1, -1\}$ , SVC genera un vettore colonna dei pesi  $\mathbf{w}$  che lega gli input agli output attraverso una funzione di decisione. SVC risolve il seguente problema di minimizzazione[10]:

$$\min_w \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N (\max(0, (1 - y_i \mathbf{x}_i \mathbf{w})) \quad (3.9)$$

dove  $\max(0, (1 - y_i \mathbf{x}_i \mathbf{w}))$  è chiamata funzione di Hinge. La funzione di Hinge è maggiore di 0 soltanto per le osservazioni che si trovano soltanto nel lato sbagliato del margine, ovvero quando  $y_i \mathbf{x}_i \mathbf{w} > 1$ . L'ampiezza del margine è regolata dal coefficiente di regolarizzazione. L'effetto del parametro  $C$ , inverso del fattore di regolarizzazione, è quello di penalizzare i support vectors. Maggiore sarà il valore di  $C$ , minore sarà l'ampiezza del margine. Un'illustrazione grafica dell'effetto di  $C$  è fornita in Figura 3.7, dove si osserva che aumentando il contributo di  $C$  il margine si restringe, andando a considerare sempre meno campioni nella soluzione finale.

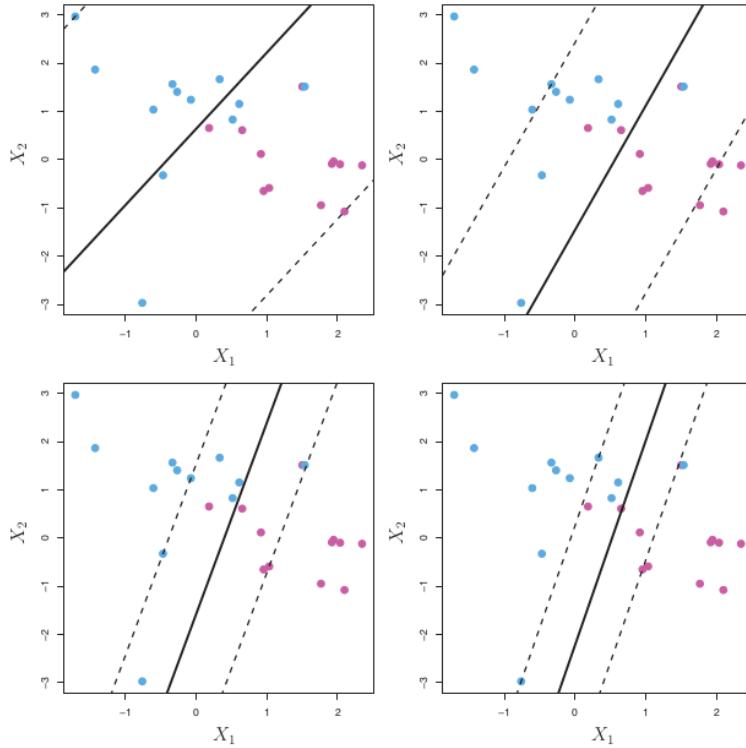


Figura 3.7: Un SVC è stata allenata sui dati di training con differenti valori di  $C$ . In alto a sinistra i valori di  $C$  sono i più piccoli, mentre il valore più alto è usato in basso a destra.

Allo scopo di ottenere vantaggi dal punto di vista computazionale, la funzione di Hinge nel problema di minimizzazione seguente è elevata al quadrato:

$$\min_w \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N (\max(0, (1 - y_i \mathbf{x}_i \mathbf{w}))^2 \quad (3.10)$$

Il fatto che SVC sia basata soltanto in un subset dei dati di training, fa sì che sia un

metodo robusto rispetto agli outliers. L'approccio usato per la risoluzione dei precedenti metodi è un coordinate descent method [11].

Una volta che l'algoritmo, per un campione  $\mathbf{x}$  con un label nella forma  $y = \{-1, 1\}$ , attraverso il training ha restituito dei valori del vettore dei pesi  $\mathbf{w}$ , la funzione di decisione, ovvero quella che decide la classe predetta  $\hat{y}$  come:

$$\hat{y} = sign(\mathbf{x}\mathbf{w}) \quad (3.11)$$

Essendo SVC un classificatore binario esso è adatto per la soluzione di problemi con due classi. In un problema multiclasse una classe alla volta è considerata come  $y=1$  e l'algoritmo restituisce una matrice dei pesi con tante righe quante sono le classi.

### 3.4.2 Logistic Regression

La LR è un algoritmo di classificazione binario lineare che nasce dal bisogno di calcolare la probabilità che un campione appartenga a una particolare classe. Per descrivere  $p(\mathbf{x}) = Pr(y = 1|\mathbf{x})$ , ovvero la probabilità condizionata di  $y = 1$  dato  $\mathbf{x}$ , una regressione lineare non è il metodo corretto [6]. Una regressione lineare modella  $p(\mathbf{x})$  come

$$p(X) = w_0 + w_1 \mathbf{x}_1 \quad (3.12)$$

dove  $w_0$  e  $w_1$  rappresentano rispettivamente l'intercetta e la pendenza associate al vettore degli input  $\mathbf{x}_1$  nel caso che  $X$  contenga una sola feature. In Figura 3.8 a sinistra, si osservano le stime di probabilità calcolate utilizzando un modello di regressione lineare rappresentate da una linea blu. Si nota come le probabilità assumano anche valori negativi, infatti una regressione lineare restituisce dei valori appartenenti all'intero insieme  $\mathbb{R}$  [12]. A destra la stima della probabilità è stata calcolata utilizzando la LR; in questo caso la stima della probabilità rimane tra 0 e 1.

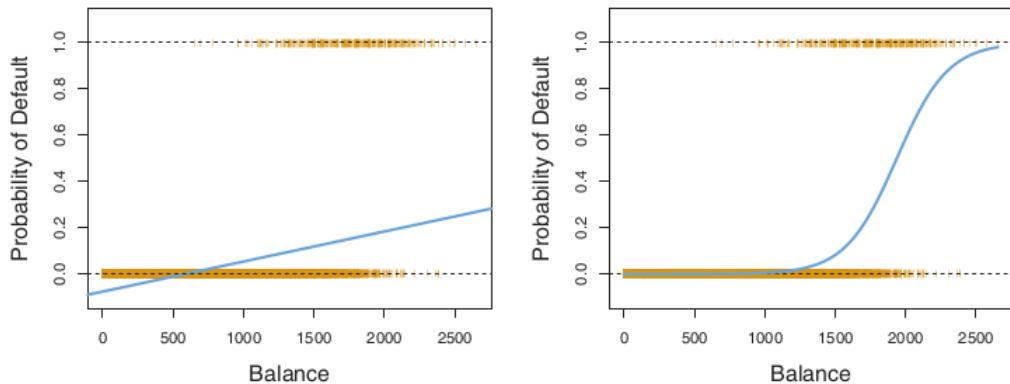


Figura 3.8: Confronto tra probabilità stimate con LR e linear regression a partire da un esempio in cui si cerca di predire la probabilità di default di un debitore utilizzando come predittore il suo bilancio economico  $Pr(Default = 1|Bilancio)$ . I campioni sono categorizzati come 0 (non default) e 1 (default avvenuto) giacciono nelle linee tratteggiate. Le linee blu rappresentano le stime di probabilità che un input appartenga rispettivamente alla classe 1 o 0. A sinistra è mostrata la stima della probabilità con l'utilizzo della linear regression, mentre a destra la stessa operazione è stata fatta con logistic regression.

La LR modella  $p(\mathbf{x})$  attraverso una funzione logistica, che come già visto ricopre tutti i range di probabilità per un determinato input.

$$p(\mathbf{x}) = \frac{e^{w_0 + w_1 \mathbf{x}_1}}{1 + e^{w_0 + w_1 \mathbf{x}_1}} \quad (3.13)$$

Si può anche notare che la variazione di un'unità dei valori di  $\mathbf{x}$ , non comporta sempre la

stessa variazione nei valori di  $p(\mathbf{x})$  come nella regressione lineare, ma che questa variazione è influenzata dai valori di  $\mathbf{x}$  stessi. Dopo il processo di training, con il vettore dei label nella forma  $y_i = \{-1, 1\}$  l'output finale è una probabilità di appartenenza  $p(\mathbf{x})$  e la classe :

$$\hat{y} = \begin{cases} 1 & \text{if } p(\mathbf{x}) > 0.5 \\ -1 & \text{Altrimenti} \end{cases} \quad (3.14)$$

Nell'implementazione utilizzata, LR è allenata utilizzando la Negative log-likelihood. Dati dei vettori di training  $\mathbf{x}_i \in R^p, i = 1, 2, ..N$  vettore degli output  $\mathbf{y} \in R^n$  nella forma  $y_i = \{1, -1\}$ , la funzione da minimizzare assume la forma [10]:

$$\min_w \quad \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^N (\log(1 + e^{-y_i \mathbf{x}_i^T \mathbf{w}})) \quad (3.15)$$

Il processo di training è realizzato attraverso un metodo di regione di confidenza di Newton [13]. Anche in questo caso C è un parametro inverso di regolarizzazione e decide il peso assunto dal fattore di regolarizzazione in norma 2.

Essendo LR un classificatore binario esso è adatto per la soluzione di problemi con due classi. In un problema multiclass uno classe alla volta è considerata come  $y = 1$  e l'algoritmo restituisce una matrice dei pesi con tante righe quante sono le classi.

### 3.4.3 Neural networks

Le Neural Networks(NN) sono algoritmi non lineari ispirati al funzionamento del cervello. Il termine Neural Network si è evoluto per comprendere una numerosa classe di modelli di learning. In questa sezione si descrive una delle più semplici applicazioni delle reti neurali, ovvero il single layer back propagation network anche chiamato Multi Layer Perceptron(MLP). Differentemente dagli algoritmi precedentemente descritti i NN coinvolgono una serie di trasformazioni lineari e non, prima che il dato di input possa essere trasformato in una classificazione. Proprio la combinazione di tali operazioni permette ai NN di approssimare funzioni molto complesse.

In Figura 3.9, è illustrato il processo a cui viene sottoposto il dato di input  $\mathbf{X}$  quando viene passato a un NN. La struttura di un NN è composta da due strutture chiamate layers: Un Hidden Layer e un Output Layer. L'Hidden Layer è formato da due elementi: una matrice dei pesi  $\mathbf{W}_1$  e una funzione detta funzione di attivazione  $g(x)$ .  $\mathbf{W}_1$  ha dimensione  $P \times L$ , dove  $P$  è il numero di features del dato di input  $\mathbf{X}$ , mentre  $L$  è la dimensione dell'Hidden Layer ed è un parametro scelto dall'utente che regola la complessità della funzione prodotta dal NN. La matrice dei dati di input  $\mathbf{X}$  viene moltiplicata con la matrice dei pesi dell'Hidden layer  $\mathbf{W}_1$  e passati alla funzione  $g(x)$ . Il risultato di questa operazione viene trasferito all'output Layer, costituito dalla matrice dei pesi  $\mathbf{W}_2$  di dimensione  $L \times Q$ , dove  $Q$  è il numero di classi, e dalla funzione di output  $f(x)$ . Generalmente ai layers si aggiunge una riga costituita dai pesi del bias  $\mathbf{w}_0$ .

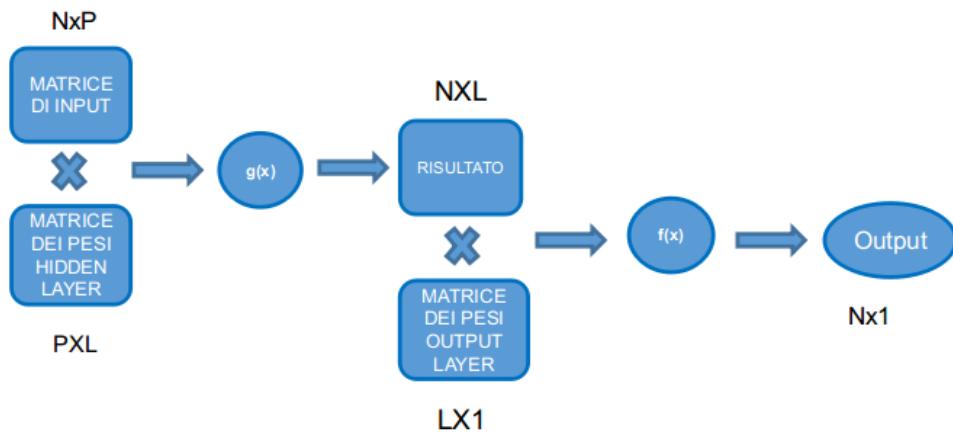


Figura 3.9: Diagramma che descrive il funzionamento di un Multi Layer Perceptron. Le  $\times$  indicano i prodotti tra matrici. La matrice di input  $N \times P$  dove  $N$  è il numero dei campioni e  $P$  il numero di features, viene in primo luogo moltiplicata con la matrice dell'Hidden Layer.  $L$  è un numero usualmente definito dall'utente e rappresenta il numero di neuroni dell'hidden layer. Il risultato di questa moltiplicazione viene passato a una funzione  $g(x)$  e passato all'output layer. Una volta applicata la funzione di output  $f(x)$  si ottiene l'output del NN. Per un NN adibito a un problema multiclassificazione, le colonne dell'Output Layer sono pari al numero di classi  $Q$ .

Considerando  $\mathbf{W}_1 \mathbf{W}_2$  i pesi associati con l'input e l'hidden layer,  $g(x)$  e  $f(x)$ , un NN per la classificazione è in grado di restituire una stima condizionata della probabilità  $P(\mathbf{y}|\mathbf{X})$  che gli input appartengano a un determinato output:

$$P(\mathbf{X}|\mathbf{y}) = f((g(\mathbf{X} \cdot \mathbf{W}_1)) \cdot \mathbf{W}_2) \quad (3.16)$$

dove  $\cdot$  indica il prodotto tra matrici. Esistono più funzioni di attivazione  $g(x)$ , quella utilizzata nel lavoro di tesi è la Rectified Linear Unit (ReLU), che ha il pregio di alleggerire il processo di training ponendo alcuni neuroni a 0 :

$$g(\mathbf{X} \cdot \mathbf{W}_1) = \max(0, \mathbf{X} \cdot \mathbf{W}_1) \quad (3.17)$$

La stima data da  $P(\mathbf{X}|\mathbf{y})$  è restituita da  $g$ , che per un problema multiclasse è di solito una funzione *SoftMax*, o esponenziale normalizzata, ovvero una generalizzazione della logistica. Se  $\mathbf{Z} = g(\mathbf{X} \cdot \mathbf{W}_1) \cdot \mathbf{W}_2$  è il risultato dell'output layer prima dell'applicazione di  $g$ , la softmax si esprime come:

$$f(\mathbf{z}) = \text{softmax}(x)_i = \frac{e^{\mathbf{z}_i}}{\sum_{q=1}^Q e^{\mathbf{z}_{iq}}} \quad (3.18)$$

La Softmax normalizza un vettore composto da  $Q$  elementi in valori compresi tra 0 e 1 la cui somma è 1.  $Q$  rappresenta il numero di classi. I valori calcolati dalla softmax sono utilizzati nel della funzione oggetto.

Per un NN adibito alla classificazione, una tipica funzione oggetto comprende la funzione di *cross entropy*(CE)  $H(\mathbf{p}_y)$ , dove  $\mathbf{p}_y$  è il vettore di lunghezza  $Q$ , dove  $Q$  è il numero delle classi, che descrive la probabilità che un campione sia associato ad ogni classe, ed è l'output della softmax.

Dati dei vettori di training nella forma  $\mathbf{x}_i \in R^P, i = 1, \dots, N$  per un caso con due classi, e un vettore  $\mathbf{y}$  nella forma  $y_i = \{0, 1\}$ , si cerca il minimo della funzione di CE più un fattore di regolarizzazione  $\alpha$  in norma 2, chiamato tipicamente *weight decay*, dal momento che penalizza valori alti nei pesi dei neuroni.

Considerando  $p_{yi1}$  come la probabilità che un campione appartenga alla classe  $q = 1$ , e  $y_i$  sia il label vero associato al campione  $\mathbf{x}$ , la funzione oggetto di un NN per la classificazione è espressa come:

$$\min_w \sum_{i=1}^N (y_i \ln(p_{yi1}) - (1 - y_i) \ln(1 - p_{yi1})) + \alpha \|\mathbf{W}\|_2^2 \quad (3.19)$$

dove  $\sum_{i=1}^N (-y_i \ln(p_{yi}) - (1-y_i) \ln(1-p_{yi}))$  si riferisce al caso binario, con due classi, della funzione di CE. In figura 3.11 è illustrata la forma della funzione di CE riferita al caso in cui la classe di interesse sia uno o zero. Si nota come per  $y=1$  descritto dalla linea blu il minimo della funzione è raggiunto per il 100% di probabilità che il campione appartenga a  $y=1$ .

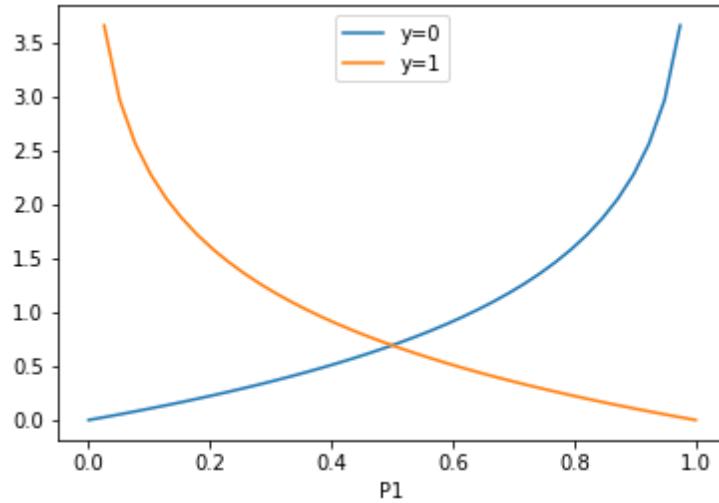


Figura 3.10: Forma della funzione di cross-Entropy per un problema binario  $y = \{0, 1\}$ . Sull'asse delle ascisse P1 rappresenta la probabilità  $P(Y = 1|X = x)$ , mentre su quello delle ordinate il valore della Cross Entropy.

Una generalizzazione del caso multi classe della cross-entropy è data dalla sommatoria, per ogni campione, del logaritmo della probabilità predetta dal NN associata al label vero:

$$\sum_{i=1}^N \mathbf{y}_i \ln(\mathbf{p}_{yi}) \quad (3.20)$$

In questo caso  $\mathbf{y}_i$  è il vettore dei label veri di zeri di lunghezza Q, dove Q è il numero delle classi, eccetto per l'indice associato alla classe vera.  $\mathbf{p}_{yi}$  è la probabilità predetta dal NN che un campione appartenga alla classe.

Un NN predice come classe di output quella associata alla probabilità maggiore. Il training di un NN restituisce i valori ottimali dei pesi contenuti in  $\mathbf{W}_1$  e in  $\mathbf{W}_2$ , ed è un problema di ottimizzazione non convesso, cioè ha più di un minimo. Per risolverlo in questo lavoro di tesi è stato adottato un algoritmo limited memory Broyden–Fletcher–Goldfarb–Shanno (LM-BFGS)[14]. Il gradiente della funzione oggetto è calcolato con un algoritmo di backpropagation. L'inizializzazione dei pesi di NN è perfromata utilizzando il metodo euristico proposto da Glorot et. al. [15].

### 3.4.4 K Nearest Neighbors

K Nearest Neighbors (KNN) è un algoritmo dalla concezione semplice che non fa assunzioni sulla forma della funzione di approssimare. Nonostante la sua semplicità KNN si è mostrato valido in un ampio numero di applicazioni, come il riconoscimento della scrittura e l'interpretazione delle immagini satellitari [6].

Considerando un problema di classificazione si suppone che un campione nello spazio delle features appartenga alla stessa classe dei suoi vicini. Si prova a dover classificare un campione di input  $\mathbf{x}_0$  utilizzando i  $K$  campioni di training  $\mathbf{x}(k), k = 1, \dots, K$  con la distanza minore da  $\mathbf{x}_0$ . Ognuno di questi campioni di training voterà la classe predetta  $\hat{y}$  di  $\mathbf{x}_0$  con la sua classe di appartenenza.

In Figura 3.11 viene mostrato un esempio di applicazione del KNN. Supponendo che  $K$  si sia uguale a 3, si osserva che i tre campioni di training più vicini al punto da predire consistono in 2 campioni blu e 1 arancione: il campione viene assegnato alla classe blu. Il classificatore, utilizzando per ogni punto nello spazio i  $K$  vicini, riesce a produrre delle linee di confine non lineari che predicono correttamente tutti i dati di training. Si tratta infatti di un algoritmo molto efficace nel fitting di funzioni fortemente non lineari, dove il parametro  $K$  scelto dall'operatore influenza in maniera decisiva l'efficacia del modello di ML. Quindi in sostanza classe che raggiunge la maggioranza dei voti è la classe predetta da KNN.

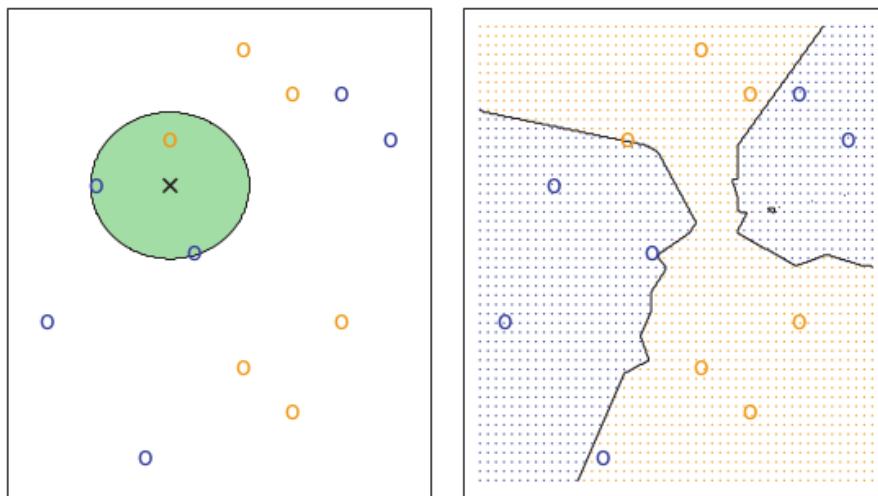


Figura 3.11: L'approccio KNN usato con  $K = 3$  su un problema di classificazione con due classi, sei arancioni e sei blu. A sinistra: il punto da predire è rappresentato da una croce. I tre punti più vicini all'osservazione di test sono identificati e la classe viene predetta come blu. A destra: I confini per questo esempio sono mostrati in nero. I grid blu e arancioni mostrano le regioni in cui è predetta la rispettiva classe.

Possiamo scrivere la regola del KNN per una classe  $q$  come:

$$\hat{y} = \max\left(\sum_{k=1}^K I(y_k)\right) \quad (3.21)$$

dove  $I()$  è la funzione indicatrice scritta come:

$$I(y_k) = \begin{cases} 1 & \text{se } y_k \text{ appartiene alla classe q} \\ 0 & \text{Altrimenti} \end{cases} \quad (3.22)$$

In alcuni casi è stato dimostrato che miglioramenti nelle performance di KNN possono essere raggiunti se la classe predetta è decisa da una somma pesata dalle distanze dei campioni, in modo che quelli più vicini influenzino maggiormente la sua scelta [16]

In questo modo si può riscrivere la regola di classificazione precedente:

$$\hat{y} = \max\left(\sum_{k=1}^K I(y_k)w_k\right) \quad (3.23)$$

Ogni voto è quindi pesato da un valore  $w_k$  dato da:

$$w_k = \frac{1}{d(\mathbf{x}_0, \mathbf{x}_k)} \quad (3.24)$$

Dove  $\mathbf{x}$  è il campione da predire. L'effetto di questa scelta è fortemente dipendente dai dati. Per esempio in un dataset sbilanciato, un valore di  $K$  troppo alto può rendere il modello di ML troppo sensibile alla classe di maggioranza. In questo caso pesare i dati per la propria distanza potrebbe aumentare la qualità della predizione. In genere due tipi di distanza sono considerati per  $w$ , quella Euclidea e quella di Manhattan, rispettivamente:

$$d(\mathbf{x}_0, \mathbf{x}_k) = \|\mathbf{x}_0 - \mathbf{x}_k\|_2 \quad (3.25)$$

$$d(\mathbf{x}_0, \mathbf{x}_k) = \|\mathbf{x}_0 - \mathbf{x}_k\|_1 \quad (3.26)$$

Il KNN offre tempi di training più corti rispetto a quelli del Neural Network, dal momento che l'unico suo compito è memorizzare il dataset per un successivo utilizzo. Per quanto riguarda la predizione dei nuovi campioni i tempi necessari per calcolare le distanze di ogni

punto dal dataset potrebbero essere proibitivi. Per ovviare a questo inconveniente, sono state create delle strutture che riducono drasticamente i tempi di calcolo. Una delle strutture più utilizzate è il KD-Tree [17].

### 3.4.5 Random Forest

Random Forest(RF) è un metodo che si basa sull’aggregazione di un certo numero di classificatori o regressori detti Decision Trees(DT).

Un DT segmenta ricorsivamente in modo binario lo spazio dei parametri in una serie di regioni  $R$  che comprendono un set di campioni. Per costruire un DT, sono necessari due steps [6]:

1. La divisione dello spazio dei parametri nel set di tutti i possibili valori per  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_P$  in  $J$  regioni distinte e non sovrapposte  $R_1, R_2, \dots, R_J$ .
2. Per ogni osservazione che fa parte di una regione  $R_j$ , si ha un solo output corrispondente alla classe più numerosa in  $R_j$

Un esempio di DT è illustrato in Figura 3.12, dove sono state effettuate 4 segmentazioni dello spazio dei parametri composto da  $\mathbf{x}_1$  e  $\mathbf{x}_2$ . Queste 4 partizioni hanno generato 5 regioni  $R_1, R_2, R_3, R_4, R_5$ . Ogni volta che avviene una segmentazione, o split, si deve scegliere un parametro  $\mathbf{x}_p$  tra  $\mathbf{x}_1$  e  $\mathbf{x}_2$  e un valore di questo parametro  $t_i$ . I campioni associati a valori minori di  $t_i$  sono assegnati a un ramo sinistro, altrimenti sono assegnati a un ramo destro. Ad ogni segmentazione si formano dunque due regioni che potranno essere soggette a nuove segmentazioni binarie.

Il primo split effettuato dall’albero è detto radice, mentre le regioni  $R_j$  sono anche dette foglie.

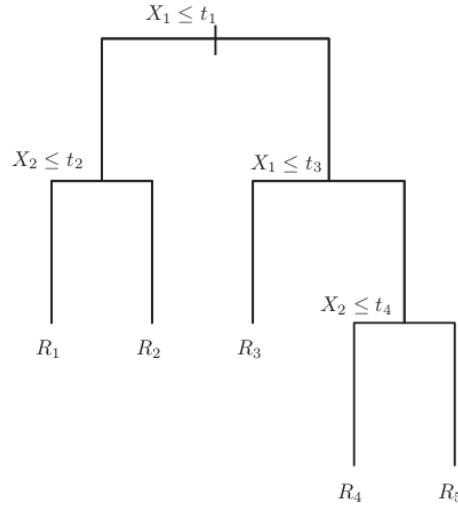


Figura 3.12: Esempio di un DT che divide lo spazio delle features in 5 regioni. Si tratta di un problema in due dimensioni dove  $\mathbf{x}_1$  e  $\mathbf{x}_2$  compongono lo spazio delle variabili. La prima segmentazione coinvolge  $\mathbf{x}_1$  e genera due regioni, le quali vengono divise ulteriormente formando le regioni  $R_1, R_2, R_3, R_4, R_5$ , che sono anche chiamate anche foglie del DT. Ad ogni segmentazione si accompagna una regola, data dal parametro scelto e dal suo valore di segmentazione  $t_i$ .

In Figura 3.13 si osservano le 5 regioni nello spazio in due dimensioni. Le segmentazioni hanno prodotto linee non sovrapposte che classificano i campioni al loro interno.

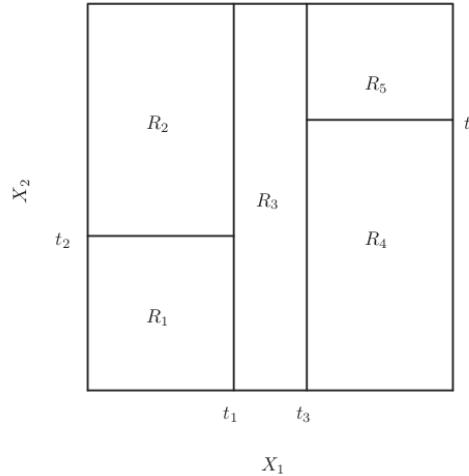


Figura 3.13: Spazio di  $\mathbf{x}_1$  e  $\mathbf{x}_2$  diviso dall'albero in Figura 3.13. I valori  $t_i$  sono i valori da cui partono le linee che dividono il dataset nelle regioni  $R_j$ .

Per selezionare la variabile  $\mathbf{x}_p$  e il suo valore  $t_i$  su cui eseguire lo split è necessaria una metrica che indichi quale combinazione di questi due elementi fornisca il risultato migliore. Una metrica comunemente utilizzata nei problemi di classificazione è la funzione di Gini o impurità  $H()$ . Se esistono  $Q$  classi associate al problema, si cerca ad ogni split  $m$  di minimizzare la funzione di Gini. Se lo split rappresenta una regione  $R_j$  con dati di training  $\mathbf{X}_j$  con numero di campioni  $N_j$ ,  $H(\mathbf{X}_j)$  è espressa come:

$$H(\mathbf{X}_j) = \sum_{q=1}^Q p_{jq}(1 - p_{jq}) \quad (3.27)$$

dove  $p_{jq}$  rappresenta la proporzione della classe  $q$  nella regione  $R_j$ :

$$p_{mk} = \frac{1}{N_j} \sum_{i=1}^{N_j} I(y_i) \quad (3.28)$$

dove  $I()$ , come già visto è la funzione indicatrice. La funzione di Gini rappresenta l'impurità del dataset contenuto in una regione. Come impurità si intende la presenza in parti più o meno uguali di classi differenti. La forma di questa funzione per due classi è mostrata in Figura 3.14. L'output più impuro è quello che comprende in parti uguali le due classi. In un dataset completamente puro  $H(\mathbf{X}) = 0$  e ci sono campioni appartenenti a una sola classe.

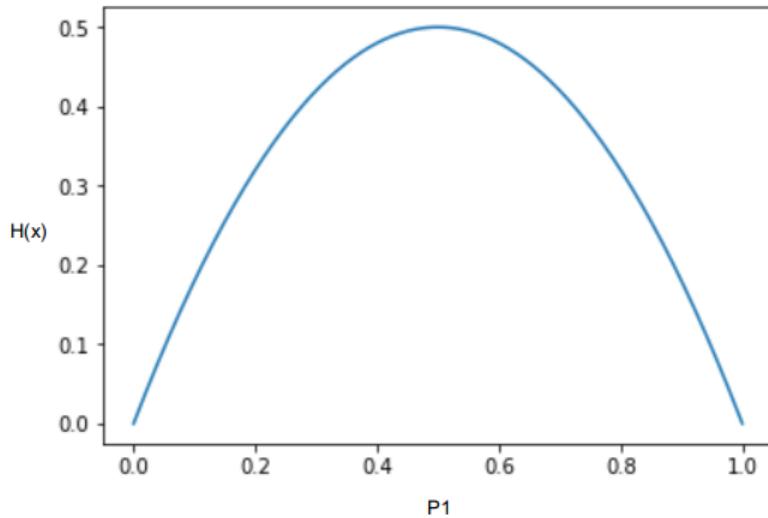


Figura 3.14: La funzione di Gini per un problema in due classi.

Quindi il processo di segmentazione dello spazio avviene selezionando ad ogni split il parametro  $\mathbf{x}_p$  e  $t_i$  che minimizzano la funzione di Gini per quella regione.

Questo processo continua finché non sono raggiunti due criteri:

1. L'impossibilità di diminuire l'impurità attraverso ulteriori segmentazioni
2. Il raggiungimento di una certa profondità( $D$ ) definita dall'utente

Come profondità ( $D$ ) si intende il numero massimo di split che ci sono tra la radice e le foglie. Una volta introdotti i DT, bisogna pensare che RF non è altro che un algoritmo costituito da un insieme di essi. RF è costituito da una serie di DT non correlati tra di loro. Ogni DT che appartiene a RF è allenato in set derivati dal Training Set con un metodo di resampling chiamato Bootstrap, descritto in appendice A.3.

Quando si costruiscono gli alberi che formano una RF, non vengono comparati tutti i parametri ad ogni split, bensì soltanto un campione random di essi. Sia il Bootstrap che la scelta random di parametri sono due metodi per cercare di aumentare la variabilità tra i DT che compongono RF.

Il numero di DT utilizzati in RF è un iper-parametro scelto dall'operatore, ed è indicato con  $N_{est}$ , numero di estimatori. In un RF un campione da predire è valutato da tutti i DT, ognuno dei quali restituisce una classe predetta. La classe più frequentemente predetta è associata al campione da predire. Un procedimento informale per RF è descritto nell'algoritmo

---

**Algorithm 2** Random Forest

---

```

1: procedure
2:   for  $i$  in  $N_{estimators}$  do
3:     Costruire un set dal Training Set utilizzando il Bootstrap.
4:     Allenare un DT sul set Bootstrap selezionando ad ogni split randomicamente
    $\sqrt{P}$  tra tutti quelli esistenti

```

---

Dove  $P$  è il numero di features. Talvolta le funzioni generate da un modello di RF possono essere troppo complesse per il problema. Per evitare questo problema si agisce fermando il processo di costruzione dei DT a una certa profondità.

## 3.5 Valutazione del modello

Il calcolo del rischio empirico avviene tramite l'utilizzo di una funzione di loss  $L$ . La definizione di  $L$  è fondamentale per comparare le performance degli algoritmi utilizzati; essa definisce qual'è l'aspetto rilevante della predizione, inoltre la valutazione di differenti misure della performance, da luogo a differenti risultati comparativi [18].

È quindi essenziale che le performance degli algoritmi siano misurate in base a uno o pochi dei più rilevanti aspetti del problema.

### 3.5.1 Confusion Matrix

La Confusion Matrix è uno strumento comodo per visualizzare le performance sul Cross Validation o sul Test Set di un algoritmo di classificazione. Una Confusion Matrix  $\mathbf{C}$ ,  $\mathbf{C} = \{c_{ij}\}, i, j \in \{1, 2, \dots, Q\}$  dove  $i$  indica il numero di righe e  $j$  il numero di colonne può essere scritta come  $\mathbf{C}(f)$  per un classificatore  $f$ .  $\mathbf{C}(f)$  è una matrice quadrata di dimensioni  $Q \times Q$  per un dataset con  $Q$  classi. Ogni elemento  $c_{ij}(f)$  della Confusion Matrix denota il numero di campioni che appartengono realmente alla classe  $i$ , ma che invece sono stati assegnati dal classificatore  $f$  alla classe  $j$ . Per esempio,  $c_{13}(f)$  indica il numero di campioni assegnati alla classe 3 da  $f$ , ma che in realtà appartengono alla classe 1.

In base a quanto detto si possono fare le seguenti considerazioni:

- $\sum_{j=1}^Q c_{ij}(f)$  denota il numero totale di campioni della classe  $i$  nel Test Set
- $\sum_{i=1}^Q c_{ij}(f)$  denota il numero totale di campioni assegnati alla classe  $j$  dal classificatore  $f$

- Tutte le caselle diagonali  $c_{ii}$  denota il numero di campioni classificati correttamente per la classe  $i$ . Quindi  $\sum_{i=1}^Q c_{ii}(f)$  denota il numero di campioni classificati correttamente dal classificatore  $f$
- Tutte le icaselle non diagonali indicano un errore di classificazione. Quindi  $\sum_{i,j:i \neq j} c_{ii}(f)$  denota il numero totale di errori di classificazione per  $f$ .

Un esempio di Confusion Matrix per un problema di classificazione binario è mostrato in figura 3.15. Utilizzando un problema con due classi, esse possono essere chiamate classe positiva(P) e Negativa(N).

Per semplificazione le metriche ricavate dalla Confusion Matrix si riferiranno a questo caso binario, benchè attraverso una notazione "uno contro tutti",cioè andando a considerare una classe alla volta come positiva, potranno essere applicate al caso multiclasse.

Come si osserva  $c_{11}$  indica gli elementi della prima riga e della prima colonna uguali al numero totale di esempi che appartengono alla classe positiva e sono assegnati a tale classe dal classificatore  $f$ . I valori assumbili all'interno di  $\mathbf{C}(f)$  sono quindi 4:

1. *Vero positivo*, la classe predetta P coincide con quella vera P
2. *Vero negativo*, la classe predetta N coincide con quella vera N
3. *Falso positivo*, la classe predetta P non coincide con quella vera N
4. *Falso negativo* , la classe predetta N non coincide con quella vera P

VERO POSITIVO $c\{11\}$	FALSO POSITIVO $c\{12\}$
FALSO NEGATIVO $c\{21\}$	VERO NEGATIVO $c\{22\}$

Figura 3.15: Esempio di Confusion Matrix per un problema di classificazione binario.

Quindi si deduce che il numero totale di positivi P e il numero totale di negativi N sia:

$$N = VN + FP \quad P = VP + FN \quad (3.29)$$

### 3.5.2 Metriche ricavate dalla Confusion Matrix

La metrica generalmente più utilizzata nella valutazione delle performance di un algoritmo è l'*Accuratezza*, ovvero la capacità del classificatore di riconoscere tanti più campioni possibili. Richiamando la definizione di rischio empirico  $R_s(f)$ , nell'equazione 3.4, di un classificatore  $f$  in esso può essere scritto in termini della Confusion Matrix:

$$R_T(f) = \frac{\sum_{i,j:i \neq j} c_{ii}(f)}{\sum_{i,j=1}^l c_{ij}(f)} = \frac{\sum_{i,j=1}^l c_{ij}(f) - \sum_{i=1}^l c_{ii}(f)}{\sum_{i,j=1}^l c_{ij}(f)} \quad (3.30)$$

il quale misura la frazione di campioni del Test Set erroneamente classificati dal classificatore.

Questa misura include i campioni di tutte le classi. La misura complementaria è l'*Accuratezza*, che misura il tasso di campioni correttamente classificati. La misura dell'*Accuratezza*  $Acc_T$  di un classificatore  $f$  nel test  $T$  è:

$$Acc_T(f) = \frac{1}{T} \sum_{i=1}^T I(y_i = f(\mathbf{x}_i)) \quad (3.31)$$

in termini di Confusion Matrix si ottiene:

$$Acc_T(f) = \frac{\sum_{i=1}^l c_{ii}(f)}{\sum_{i,j=1}^l c_{ij}(f)} \quad (3.32)$$

che per il caso di classificazione binario diventa

$$Acc_T(f) = \frac{c_{11}(f) + c_{22}(f)}{c_{11}(f) + c_{12}(f) + c_{21}(f) + c_{22}(f)} = \frac{VP + VN}{P + N} \quad (3.33)$$

Questa misura riassume la performance generale tenendo in considerazione tutte le classi, quindi non fornisce alcuna informazione su come è stata classificata una classe in particolare. Inoltre l'*Accuratezza* fornisce un metodo di misurazione efficace soltanto quando le proporzioni di campioni appartenenti alle classi differenti sono più o meno bilanciate. Sal momento che la *skew* di una distibuzione punta verso una determinata classe sarà quella a prevalere sulla quantità misurata, creando un bias nella stima.

Un modo per sorpassare questo problema è quello di pesare le predizioni corrette per ogni classe per il loro numero di campioni. In questo modo si ottiene la *Accuratezza Bilanciata*, che per il caso binario può essere scritto come:

$$BAcc_T = \frac{VP}{\frac{P}{2}} + \frac{VN}{\frac{N}{2}} \quad (3.34)$$

Il problema è illustrato in figura 3.16, dove in uno scenario di classificazione binaria la classe negativa, con soli 7 campioni rispetto a quella positiva con 100 campioni, non è mai stata predetta correttamente. In questo caso l'*Accuratezza* produce una stima biasata:

$Acc = \frac{90 + 0}{107} = 0.84$ , mentre l' *AccuratezzaBilanciata* rende esplicita la mancanza di predizioni corrette per N:  $BAcc = \left( \frac{90}{100} + \frac{0}{7} \right) \frac{1}{2} = 0.45$

		P	N
		90	10
P	P	90	10
	N	7	0

Figura 3.16: Esempio di Confusion Matrix per un problema di classificazione binario con classi sbilanciate a favore della classe Positiva.

### 3.5.3 Metriche specifiche per le classi

Un altro aspetto importante ricavabile dalla Confusion Matrix è la proporzione di campioni che appartengono realmente alla classe  $i$  tra tutti i campioni classificati come  $i$ . *Precision* e *Recall* misurano questa statistica, considerando  $i$  come la classe positiva.

In termini dei valori assumibili nella Confusion Matrix si può definire la *Precision* come:

$$Prec(f) = \frac{TP}{TP + FP} \quad (3.35)$$

Quindi questa metrica misura quanto "preciso" è stato l'algoritmo a identificare i campioni della classe positiva.

La controparte della "Precision" è il "Recall", il quale misura la stessa quantità per la classe negativa:

$$Rec(f) = \frac{TP}{TP + FN} \quad (3.36)$$

Queste statistiche forniscono informazione aggiuntiva quando non si è interessati soltanto alla proporzione di informazione identificata, ma si vuole indagare quanta informazione utile è stata riconosciuta.

Un problema che riguarda la valutazione attraverso queste metriche è che non riescono a

incapsulare tutti gli aspetti di interesse. Inoltre più di un valore deve essere riportato per valutare il classificatore di interesse, il chè rende più difficoltosa l'interpretazione del risultato.

Una metrica che tenta di combinare l'informazione di *Precision* e *Recall* è l' $F_1$  score. L'  $F_1$  score cerca di migliorare la qualità dell'informazione generando un solo valore di riferimento. Più specificatamente si tratta di una media armonica di *Precision* e *Recall*.

$$F_1 = \frac{2[Prec(f) \times Rec(f)]}{Prec(f) + Rec(f)} \quad (3.37)$$

ed è chiamata anche *misura F Bilanciata* dal momento che la formula generale è descritta come:

$$F_\alpha = \frac{(1 + \alpha)[Prec(f) \times Rec(f)]}{[\alpha \times Prec(f)] + Rec(f)} \quad (3.38)$$

dove  $\alpha$  è deciso in base al trade off tra *Precision* e *Recall*.

### 3.6 Inversione Bayesiana Amplitude Versus Offset

Una inversione Bayesiana risulta estremamente utile in geofisica dal momento che può sfruttare l'informazione fornita dai modelli a priori. In caso di mal-condizionamento, una situazione comune in abito geofisico, esisteranno più modelli in grado di spiegare il dato osservato. Mentre in molti altri approcci la riduzione dello spazio delle soluzioni avviene attraverso l'utilizzo di un fattore di regolarizzazione, l'inversione Bayesiana cerca di ridurre lo spazio delle soluzioni possibili attraverso l'incorporazione di un modello a priori che proviene da informazioni a priori pregresse, come per esempio interpretazioni geologiche o dati di pozzo riguardanti l'area di indagine. Questo tipo di informazione è espresso con una distribuzione di probabilità a priori  $p(m)$ , combinata poi con i dati sperimentali per ottenere una distribuzione di probabilità a posteriori  $p(m|d)$ . Il modello che corrisponde ai valori più alti di  $p(m|d)$  è chiamato modello Maximum a Posteriori (MAP) e spesso coincide con la soluzione di un'inversione ai minimi quadrati.

Un'inversione Amplitude Versus Offset(AVO) è una inversione sismica pre-stack che ha lo scopo di stimare i parametri elastici utilizzando le ampiezze degli eventi sismici riflessi a differenti offset/angoli [19]. L'inversione AVO sfrutta la dipendenza dei coefficienti di riflessione dall'angolo di incidenza dei raggi sismici con l'interfaccia riflettente; i coefficienti di riflessione a loro volta dipendono dai parametri elastici: la velocità delle onde P  $V_p$ , la velocità delle onde di taglio  $V_s$  e la densità  $\rho$ . Utilizzando un approccio Bayesiano nella risoluzione dell'inversione AVO sarà possibile caratterizzare l'incertezza associata ai parametri elastici.

Lo scopo dell'inversione è quindi, dato un sismogramma al variare dell'offset che rappresenta i dati osservati  $d_{obs}$  ottenere una distribuzione a posteriori dei parametri elastici che rappresentano il modello  $m$ . La relazione tra una traccia sismica e la sua variazione con l'offset è non lineare, ed è descritta dalle equazioni di Zoeppritz. Per realizzare l'inversione si utilizza il modello convoluzionale ed un'approssimazione lineare dell'equazione di Zoeppritz valida per piccoli angoli di incidenza( $< 40^\circ$ ) e piccoli contrasti elastici dell'interfaccia riflettente [20]. In questo caso l'equazione che lega i coefficienti di riflessione PP (Cpp) e i parametri elastici è:

$$Cpp(\theta) = a_\alpha(\theta) \frac{\Delta\alpha}{\bar{\alpha}} a_\beta(\theta) \frac{\Delta\beta}{\bar{\beta}} + a_\rho(\theta) \frac{\Delta\rho}{\bar{\rho}} \quad (3.39)$$

dove

$$a_\alpha(\theta) = \frac{1}{2}(1 + \tan^2\theta) \quad (3.40)$$

$$a_\beta(\theta) = -4 \frac{\bar{\beta}^2}{\bar{\alpha}^2} \sin^2\theta \quad (3.41)$$

$$a_\rho(\theta) = \frac{1}{2}(1 - 4 \frac{\bar{\beta}^2}{\bar{\alpha}^2} \sin^2\theta) \quad (3.42)$$

$\bar{\alpha}, \bar{\beta}$  e  $\bar{\rho}$  sono rispettivamente  $V_p$ ,  $V_s$  e  $\rho$  mediate sull'interfaccia riflettente;  $\Delta\alpha, \Delta\beta$  e  $\Delta\rho$  sono i corrispondenti contrasti e  $\theta$  è l'angolo di riflessione.

La relazione deve essere estesa per una funzione continua nel tempo:

$$Cpp(t, \theta) = a_\alpha(t, \theta) \frac{\partial}{\partial t} \ln \alpha(t) a_\beta(t, \theta) \frac{\partial}{\partial t} \ln \beta(t) a_\rho(t, \theta) \frac{\partial}{\partial t} \ln \rho(t) \quad (3.43)$$

qui  $a_\alpha(t, \theta), a_\beta(t, \theta)$  e  $a_\rho(t, \theta)$  sono generalizzazioni delle espressioni dipendenti solo dall'angolo, con velocità che dipendono dal tempo  $\bar{\alpha}(t), \bar{\beta}(t)$  rappresentate da un modello elastico a bassa frequenza che è indipendente dal dato sismico ma dipendente dall'area investigata.

### 3.6.1 Il modello a priori

Il modello a priori racchiude l'informazione indipendente dai dati a disposizione e deriva da informazioni a priori sullo spazio dei parametri elastici. Per garantire la linearità del problema,  $\alpha(t), \beta(t)$  e  $\rho$  sono assunti log-gaussiani. Il logaritmo dei parametri elastici consente di definire un campo vettoriale Gaussiano:

$$\mathbf{m}(t) = [\alpha(t), \beta(t), \rho(t)]^T \quad (3.44)$$

con valore atteso

$$E\{\mathbf{m}(t)\} = \mu(t) = [\mu_\alpha(t), \mu_\beta(t), \mu_\rho(t)]^T \quad (3.45)$$

bisogna definire i valori della matrice di covarianza a priori considerando la correlazione temporale. La covarianza di  $\mathbf{m}$  per i tempi  $t$  e  $s$  è

$$Cov\{\mathbf{m}(t), \mathbf{m}(s)\} = \Sigma(t, s) \quad (3.46)$$

dove

$$\Sigma(t, s) = \Sigma_0 \nu(\tau) \quad (3.47)$$

e  $\Sigma_0$  è la matrice di covarianza del modello a priori priori stazionario,  $\nu$  è la funzione di correlazione temporale e  $\tau$  è la differenza dei tempi  $|t - s|$ ,  $\nu$  lega i valori dei parametri elastici in base alla loro posizione nell'asse dei tempi.

$$\Sigma_0 = \begin{bmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \nu_{\alpha\beta} & \sigma_\alpha \sigma_\rho \nu_{\alpha\rho} \\ \sigma_\alpha \sigma_\beta \nu_{\alpha\beta} & \sigma_\beta^2 & \sigma_\beta \sigma_\rho \nu_{\beta\rho} \\ \sigma_\alpha \sigma_\rho \nu_{\alpha\rho} & \sigma_\beta \sigma_\rho \nu_{\beta\rho} & \sigma_\rho^2 \end{bmatrix} \quad (3.48)$$

dove gli elementi diagonali sono le varianze e  $\nu_{\alpha\beta}, \nu_{\alpha\rho}, \nu_{\beta\rho}$  sono le correlazioni tra  $\ln\alpha(t)$ ,  $\ln\beta(t)$  and  $\ln\rho(t)$ , rispettivamente. Gli elementi della diagonale di  $\Sigma_0$  sono le varianze di  $V_p$ ,  $V_s$  e  $\rho$ . La funzione di correlazione temporale  $\nu_t(\tau)$  deve essere una funzione positiva con valori nell'intervallo  $[-1, 1]$  e avere la proprietà che  $\nu_t(0) = 1$ . Una tale funzione di correlazione è la funzione esponenziale negativa:

$$\nu_t(\tau) = \exp\left[-\left(\frac{\tau}{d}\right)^2\right] \quad (3.49)$$

dove  $d$  è un parametro che caratterizza il range della dipendenza temporale. La derivata di  $\mathbf{m}(t)$  e la rispettiva matrice di covarianza  $\Sigma''(t, s)$  serviranno sia nel forward modeling che nell'inversione:

$$\mathbf{m}'(t) = \left[ \frac{\partial}{\partial t} \ln\alpha(t), \frac{\partial}{\partial t} \ln\beta(t), \frac{\partial}{\partial t} \ln\rho(t) \right]^T \quad (3.50)$$

Con valore atteso :

$$E\{\mathbf{m}'(t)\} = \frac{\partial}{\partial t}\mu(t) \quad (3.51)$$

E' associato a una covarianza che è la derivata seconda su t ed s della matrice di covarianza associata a  $\mathbf{m}$ :

$$Cov\{\mathbf{m}'(t), \mathbf{m}'(s)\} = \frac{\partial^2}{\partial t \partial s} \Sigma(t, s) = \Sigma''(t, s) \quad (3.52)$$

La cross-covarianza tra  $\mathbf{m}'(t)$  e  $\mathbf{m}(s)$  è

$$Cov\{\mathbf{m}'(t), \mathbf{m}(s)\} = \frac{\partial}{\partial t}\Sigma(t, s) = \Sigma'(t, s) \quad (3.53)$$

Nel software che implementa il forward modeling e l'inversione i campi di valori continui sono approssimati da vettori e matrici:

$$E\{m(t)\} = \boldsymbol{\mu}_m, \quad \Sigma(t, s) = \boldsymbol{\Sigma}_m \quad (3.54)$$

$$E\{m'(t)\} = \boldsymbol{\mu}'_m, \quad \Sigma'(t, s) = \boldsymbol{\Sigma}'_m, \quad \Sigma''(t, s) = \boldsymbol{\Sigma}''_m \quad (3.55)$$

### 3.6.2 Il forward modeling sismico

L'obiettivo del forward modeling sismico in questa inversione è quello di passare dai valori dei parametri elastici  $\mathbf{m}$  ai valori delle ampiezze del sismogramma al variare degli angoli di incidenza, che in questo caso sono i dati  $\mathbf{d}_{obs}$ .

Essendo il forward modeling lineare esso è quindi scrivibile in forma matriciale. Per realizzare il forward modeling sono necessari gli elementi del modello convoluzionale:

- Un vettore di riflettività  $\mathbf{c}$  al variare dei tempi  $t$  e degli angoli  $\theta$ , calcolato a partire dai valori dei parametri elastici.
- Una matrice  $\mathbf{S}$  che rappresenta l'onda del modello convoluzionale per ogni angolo di riflessione.

La moltiplicazione tra  $\mathbf{S}$  e  $\mathbf{c}$  restituisce il sismogramma  $\mathbf{d}_{obs}$

$$\mathbf{d}_{obs} = \mathbf{Sc} + \mathbf{e} \quad (3.56)$$

dove  $e$  rappresenta il vettore associato a rumore random. Per rappresentare la convoluzione attraverso una moltiplicazione tra matrici,  $S$  è una matrice a blocchi diagonali contenente l'onda al variare dei coefficienti di riflessione. Si può riscrivere l'equazione precedente nella forma espansa per esplicitare la forma degli operatori:

$$\begin{bmatrix} \mathbf{d}_{obs}(\theta_1) \\ \vdots \\ \mathbf{d}_{obs}(\theta_n) \end{bmatrix} = \begin{bmatrix} \mathbf{S}(\theta_1) & & \\ & \ddots & \\ & & \mathbf{S}(\theta_n) \end{bmatrix} \times \begin{bmatrix} \mathbf{c}(\theta_1) \\ \vdots \\ \mathbf{c}(\theta_n) \end{bmatrix} + \begin{bmatrix} \mathbf{e}(\theta_1) \\ \vdots \\ \mathbf{e}(\theta_n) \end{bmatrix} \quad (3.57)$$

dove  $\mathbf{d}_{obs}(\theta_i)$ ,  $\mathbf{c}_{obs}(\theta_i)$  esprimono rispettivamente la traccia sismica e la traccia di riflettività nel dominio temporale per l'angolo i,  $\mathbf{S}(\theta_i)$  è una matrice Toeplitz che rappresenta l'ondina S per l'angolo i.

Il calcolo della funzione di riflettività discreta  $C_{pp}(t, \theta)$  può essere effettuato attraverso il prodotto:

$$\mathbf{c} = \mathbf{A}\mathbf{m}' \quad (3.58)$$

la matrice  $\mathbf{A}$  è definita dai coefficienti  $a_\alpha(t, \theta)$ ,  $a_\beta(t, \theta)$  e  $a_\gamma(t, \theta)$  ed ha la forma:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_\alpha(\theta_1) & \mathbf{A}_\beta(\theta_1) & \mathbf{A}_\rho(\theta_1) \\ \vdots & \vdots & \vdots \\ \mathbf{A}_\alpha(\theta_n) & \mathbf{A}_\beta(\theta_n) & \mathbf{A}_\rho(\theta_n) \end{bmatrix} \quad (3.59)$$

$\mathbf{e}$  è assunto gaussiano a media zero

$$\mathbf{e} \sim \mathcal{N}(0, \Sigma_e) \quad (3.60)$$

e indipendente da  $\mathbf{m}$ . Anche  $\mathbf{d}_{obs}$  è Gaussiano

$$\mathbf{d}_{obs} \sim \mathcal{N}(\boldsymbol{\mu}_{d_{obs}}, \Sigma_{d_{obs}}) \quad (3.61)$$

con

$$\boldsymbol{\mu}_{d_{obs}} = S\mathbf{A}\boldsymbol{\mu}'_m \quad (3.62)$$

$$\Sigma_{d_{obs}} = S\mathbf{A}\Sigma''_m\mathbf{A}^T S^T + \Sigma_e \quad (3.63)$$

### 3.6.3 Il modello a posteriori

Assumendo la distribuzione a priori Gaussiana ed essendo il forward modeling lineare anche la distribuzione a posteriori sarà Gaussiana. La media di questa distribuzione  $\boldsymbol{\mu}_{m|d_{obs}}$  coincide pertanto con la soluzione map, che può essere calcolata come:

$$\boldsymbol{\mu}_{m|d_{obs}} = \boldsymbol{\mu}_m + (\mathbf{S}\mathbf{A}\Sigma'_m)^T \Sigma_{d_{obs}}^{-1} (\mathbf{d}_{obs} - \boldsymbol{\mu}_{d_{obs}}) \quad (3.64)$$

La matrice di covarianza a posteriori della soluzione map che descrive l'incertezza sulla soluzione ottenuta è:

$$\boldsymbol{\Sigma}_{m|d_{obs}} = \boldsymbol{\Sigma}_m - (\mathbf{S}\mathbf{A}\boldsymbol{\Sigma}'_m)^T \boldsymbol{\Sigma}_{d_{obs}}^{-1} \mathbf{S}\mathbf{A}\boldsymbol{\Sigma}'_m \quad (3.65)$$

La distribuzione a posteriori descrive completamente l'incertezza connessa alla stima delle proprietà di interesse. Un set di possibili soluzioni possono essere generate attraverso simulazioni stocastiche che estraggono stime plausibili dei parametri elastici dalla distribuzione a posteriori.

### 3.7 Relazioni tra variabili elastiche e litho-fluidfacies

La determinazione delle litho-fluidfacies in un dato sismico può avvenire attraverso l'utilizzo di tre variabili fondamentali: la velocità delle onde compressionali  $V_p$ , la velocità delle onde di taglio  $V_s$  e la densità  $\rho$ . Purtroppo queste tre proprietà da sole non sono sempre in grado di fornire un constraint sulla litologia della roccia: essi non dipendono direttamente dalle lithofacies ma da una serie di parametri non strettamente dipendenti dalla litologia, come la porosità, la saturazione, la cementazione, la profondità, la composizione mineralogica e il contenuto in argilla.

Tuttavia esiste una serie di relazioni fisiche che regola i rapporti tra i parametri petrofisici e quelli sismici. Tra le litho-fluidfacies di comune interesse in ambito di esplorazione ci sono le litologie clastiche, ovvero le Shale, le Brine Sands e le Gas Sands. Considerando  $K$  il modulo di bulk di un determinato materiale e  $\mu$  il suo modulo di taglio la  $V_p$  può essere definita come:

$$V_p = \sqrt{\frac{K + 4/3\mu}{\rho}} \quad (3.66)$$

mentre  $V_s$  non dipende dal modulo di bulk

$$V_s = \sqrt{\frac{\mu}{\rho}} \quad (3.67)$$

L'equazione di Gassman, valida per materiali isotropi con propagazione delle onde a bassa frequenza descrive come il modulo di bulk di una roccia possa essere scomposto in due fattori:

$$\frac{1}{K_r} = \frac{1}{K_m} + \frac{\phi}{K_\phi} \quad (3.68)$$

dove  $K_r$ ,  $K_m$  e  $K_\phi$  sono rispettivamente il modulo di bulk della roccia saturata, del minerale che compone la roccia e dello spazio saturato.

Si osserva come la velocità delle onde compressionali dipenda in modo indiretto da queste componenti del modulo di bulk. Dal momento che il modulo di bulk di una roccia contenente brine è generalmente più alto di quello di una roccia contenente gas, la  $V_p$  dipende dal tipo

di fluido che satura la roccia. In figura 3.17 è mostrata la dipendenza di  $V_p$  e  $\rho$  in funzione della saturazione in gas. Si osserva come un leggero aumento nella saturazione in gas porti a un netto decremento nella  $V_p$ , tuttavia dopo una saturazione del 20% questo trend si inverte e la  $V_p$  non è determinativa del contenuto in gas della roccia. La densità  $\rho$  mantiene un trend lineare inversamente proporzionale alla saturazione in gas.

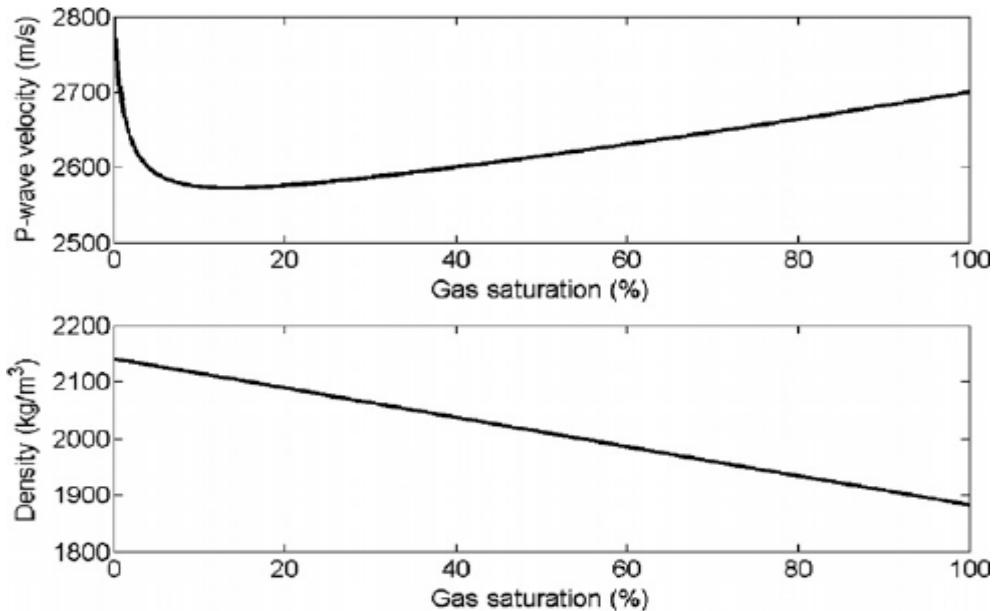


Figura 3.17: Valori di densità e  $V_p$  in relazione alla saturazione in gas.

Le stesse considerazioni non valgono per  $V_s$ , che non dipende dal modulo di bulk  $K$  e quindi non è sensibile a variazioni nel fluido di saturazione.

Un altro importante parametro nella determinazione della  $V_p$  e della  $V_s$  è la porosità  $\phi$ . La porosità può venire in aiuto nella distinzione tra Shale e Sand se lo stato diagenetico delle litologie è definito. In figura 3.18 si osserva come varia la porosità a seconda degli stati diagenetici per un campione di pure Shale e uno di clean sand. Durante la diogenesi i rapporti di porosità tra Shale e Sand si invertono più volte. Sebbene una misura quantitativa di questi rapporti sia difficoltosa si possono definire tre regole:

- Al momento della compattazione le Shale, a causa del loro fattore di forma tendono ad avere una porosità maggiore rispetto alle Sands.
- Le Shale perdono porosità molto più rapidamente durante la compattazione meccanica
- Le Sands perdono porosità più rapidamente durante la compattazione chimica

La  $V_p$  ha un andamento che in genere è inversamente proporzionale a  $\phi$ , quindi se si conosce lo stato diagenetico delle rocce nella zona di interesse la  $V_p$  potrà fornire uno strumento utile nella discriminazione tra le due facies. Tuttavia altri fattori complicano il

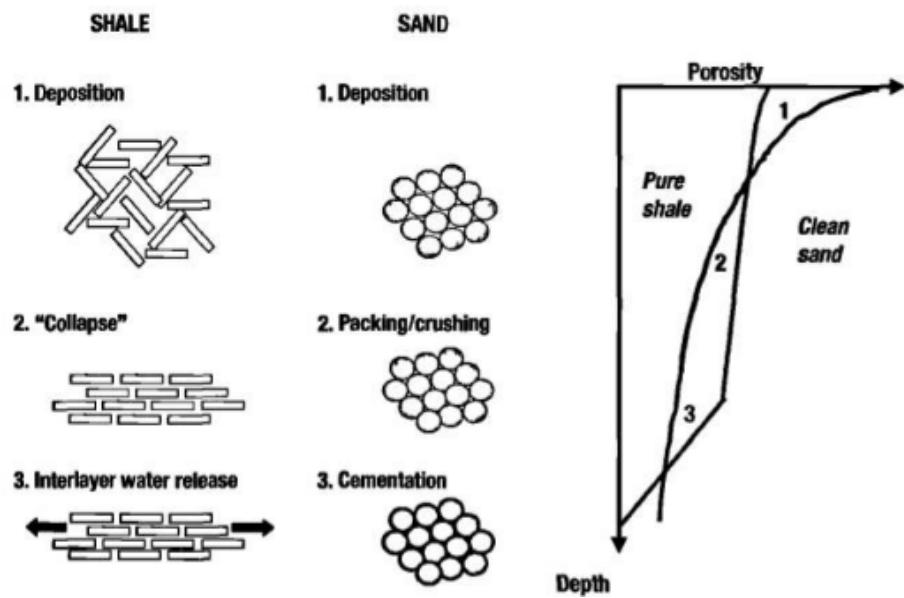


Figura 3.18: Variazione con la profondità della porosità per campioni di Sand e Shale pure. A sinistra gli stati diagenetici che si verificano a una certa profondità. Dal momento che entrambe le litologie possono variare notevolmente a causa della composizione, della texture, e dal gradiente di pressione non è stata assegnata alcuna scala agli assi.

problema. La presenza di grani di argilla all'interno delle sabbie diminuisce significativamente la porosità, senza però generare un aumento di  $V_p$  altrettanto significativo. In questo contesto la  $V_s$  può fornire informazione aggiuntiva: la forma appiattita dei minerali di argilla causa una un decremento della velocità delle onde di taglio, conseguentemente il rapporto tra  $V_p$  e  $V_s$  nelle Shale è generalmente più basso rispetto a quello nelle Sands.



# Capitolo 4

## Indagine preliminare

Prima di procedere alla scelta del modello di ML sono state effettuate delle indagini preliminari per valutare l'effetto di diversi fattori sul risultato finale. Dopo aver classificato le litho-fluidfacies di tutti e 5 i log attraverso l'uso di valori cut-off dei parametri petrofisici e aver associato queste classi ai corrispondenti valori dei parametri elastici si sono svolte tre indagini preliminari:

- Una fase di analisi del dato dei log di pozzo
- una valutazione dell'effetto della standardizzazione
- l'osservazione del comportamento degli algoritmi al variare dei loro parametri.

### 4.1 Comprensione del dato

I dati dei log di pozzo comprendono i 3 parametri elastici  $V_p$ ,  $V_s$  e  $\rho$  e le 3 proprietà petrofisiche: la saturazione in acqua  $Sw$ , l'argillosità  $Sh$  e la porosità  $\phi$ . Il primo step del lavoro comprende:

1. La classificazione dei campioni in tre litho-fluidfacies attraverso dei valori cut-off delle proprietà petrofisiche.
2. L'associazione delle classi ai valori dei parametri elastici.

Le litho-fluidfacies di interesse sono: Shale, Gas Sands e Brine Sand. Le Shale, sono associate a cut-off di argillosità  $Sh \geq 0.5$  e di porosità  $\phi \leq 0.1$ , tutti i campioni restanti sono classificati come Brine sands, che hanno valori di saturazione in acqua  $Sw \geq 0.5$  e Gas Sands con  $Sw < 0.5$ .

In figura 4.1 sono illustrati i valori in scala logaritmica dei parametri per un pozzo dell'area di studio. Si osserva come le proprietà petrofisiche siano state calcolate nel range di profondità

dove è presente un'evidente calo dei valori delle proprietà elastiche causato dalla presenza di un livello saturo a gas identificato come il reservoir.

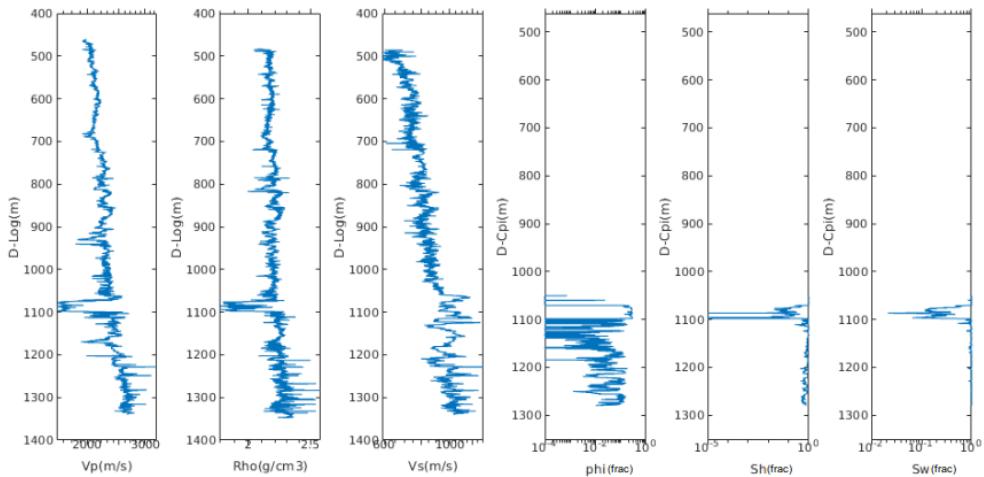


Figura 4.1: In figura sono mostrati i dati relativi a un pozzo dell'area di studio. Sono mostrati i valori delle 3 proprietà elastiche e delle 3 petrofisiche. Queste sono associate a un asse di profondità differente.

Il procedimento ha restituito un dataset formato da 7047 campioni i quali sono prevalentemente composti dalle Shale. La proporzione delle litho-fluidfacies è mostrata in un grafico a barre in Figura 4.2. Le Shale rappresentano all'incirca l'85% dei campioni, mentre il gas rappresenta solo il 4,6% dell'intero dataset. Si è quindi in presenza di un dataset sbilanciato.

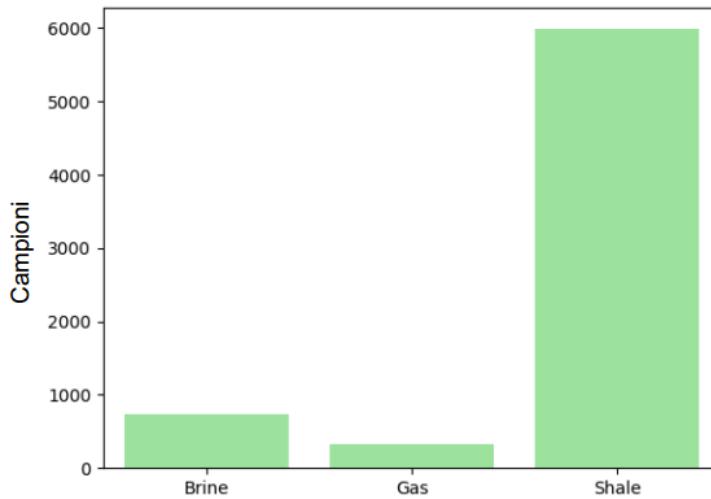


Figura 4.2: In figura è mostrato un grafico a barre che mostra il numero di campioni per ogni classe sull'intero dataset a disposizione. Le Shale contribuiscono con 5987 campioni, le Brine Sand con 736 campioni mentre le Gas Sand sono 324 campioni.

I campioni di log sono stati visualizzati in un grafico 2D, utilizzando l'impedenza  $P$  nell'asse delle ascisse e il rapporto  $V_p/V_s$  allo scopo visualizzare l'andamento dei parametri elasticici in funzione delle facies.

In Figura 4.3 si osservano i tre clusters che identificano le litho-fluidfacies.

Si osserva l'esistenza di una relazione tra i parametri e le litho-fluidfacies: i tre clusters sono concentrati in zone differenti. Come in genere accade, le Shale hanno un rapporto  $V_p/V_s$  più alto rispetto a quello delle Sands, dal momento che il modulo di taglio nelle Shale risente particolarmente della forma appiattita dei minerali delle argille.

Le Sands contenenti brine si mostrano in questo grafico come un cluster piuttosto sparso, con una concentrazione più importante tra valori di  $V_p/V_s$  compresi tra 1.9 e 2.25 e tra valori di  $I_p$  compresi tra  $4 \times 10^6$  e  $4,5 \times 10^6 \text{ m/s} \times \text{Kg/m}^3$ . La sparsità di tale litho-fluidfacies potrebbe essere dovuta al variare del contenuto in argilla, che ha la proprietà di riempire i vuoti nei grani di sabbia e di generare un aumento dei moduli elastici  $K$  e  $\mu$ . Il cluster del gas è più facilmente distinguibile e giace a valori bassi di proprietà elastiche. Il gas infatti ha la proprietà di ridurre notevolmente rispetto al brine il modulo di bulk  $K$  e la densità. Le tre classi sono sovrapposte nello spazio delle features soprattutto nell'interfaccia tra Brine Sand e Shale, ci si aspetta dunque che la maggioranza degli errori nella classificazione risieda nella zona che nello spazio dei parametri rappresenta l'interfaccia tra Brine Sand e Shale.

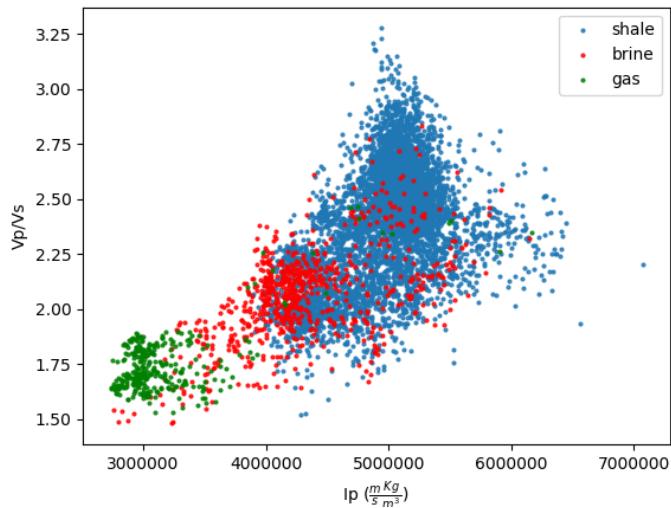


Figura 4.3: Visualizzazione in un piano  $I_p$   $V_p/V_s$  dei campioni riguardanti tutti i log di pozzo disponibili

Sono inoltre state calcolate le proprietà statistiche del dataset. Nelle Tabelle 4.1, 4.2 e 4.3 sono consultabili i parametri statistici  $V_p$ ,  $V_s$  e  $\rho$  differenziati in base alle tre litho-fluidfacies. Dalle tabelle si osserva come la media di  $V_p$  e  $\rho$  decresca sensibilmente passando da Shale a gas sand, mentre le tre classi mostrano un valore medio di  $V_s$  piuttosto simile. Brine Snads

mostrano dei valori di deviazione standard di  $V_p$  e  $\rho$  significativamente più alti, segno che i campioni sono spalmati su un range di valori ampio.

Tabella 4.1: Parametri statisci delle Shale

Shale	$V_p$ m/s	$V_s$ m/s	$\rho$ g/cm <sup>3</sup>
Media	2286.7	961.80	2.1790
Std	138.23	96.0484	0.0490
Min	1902.0	686.38	1.906
Max	2985.6	1486.14	2.4090

Tabella 4.2: Parametri statisci delle Brine

Brine	$V_p$ m/s	$V_s$ m/s	$\rho$ g/cm <sup>3</sup>
Media	2046.4	994.24	2.100
Std	220.6	89.30	0.0699
Min	1405.9	823.3	1.883
Max	2778.9	1328.0	2.329

Tabella 4.3: Parametri statisci delle Gas

Gas	$V_p$ m/s	$V_s$ m/s	$\rho$ g/cm <sup>3</sup>
Media	1661.6	947.11	1.9009
Std	184.92	62.94	0.0836
Min	1510.9	840.77	1.7619
Max	2849.3	1254.1	2.1790

Successivamente sono state stimate le funzioni di densità di probabilità attraverso l'utilizzo di un kernel Gaussiano.

Per realizzare questa operazione si è utilizzato l'algoritmo kernel density. Si è utilizzata una finestra di Silverman [21] che stima la larghezza di banda di ogni campione come:

$$h = 1.06 \times \sigma \times N^{-1/5} \quad (4.1)$$

Dove  $N$  è il numero totale di campioni e  $\sigma$  è la loro deviazione standard. Nella pratica, questa funzione assegna ad ogni campione presente nel dataset una finestra  $h$ , la quale è utilizzata come deviazione standard di una funzione gaussiana. Ogni dato sarà rappresentato da questa funzione di densità che avrà come media i valori dei campioni stessi. La somma di tutte le singole distribuzioni dei dati restituisce la stima finale.

I risultati della stima della densità sui parametri normalizzati per media e varianza sono

visualizzati in Figura 4.4. La funzione di densità è stata calcolata per ogni classe.

$V_p$  e  $\rho$  dividono i dati in tre distribuzioni distinte, dalle quali si intuisce una certa correlazione tra variabile e litho-fluidfacies. Per quanto riguarda  $V_s$  le tre distribuzioni hanno media e varianza molto simile, indice che tale variabile non fornisce una sostanziale differenziazione tra le classi.  $V_s$  infatti non dipende dal fluido che riempie la porosità delle rocce e inoltre la zona di interesse potrebbe essere situata a profondità e a stati di compattazione della roccia che non permettono una distinzione di  $V_s$  tra Shale e Sands.

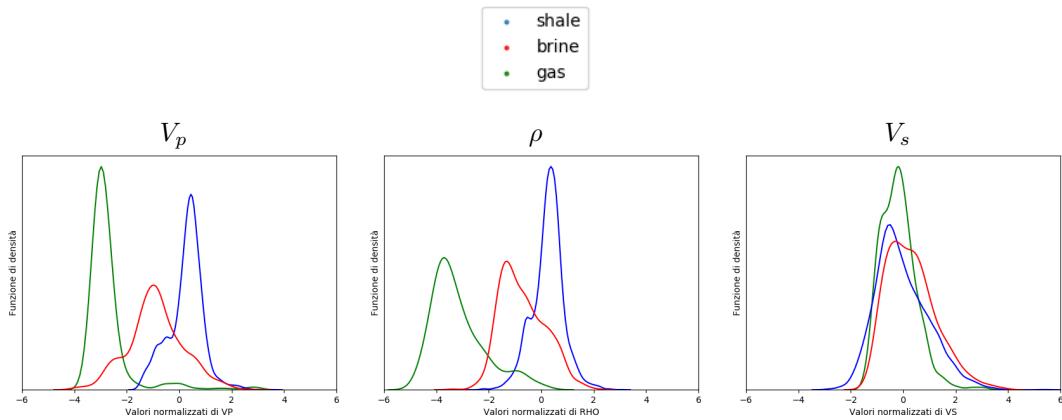


Figura 4.4: Funzioni di densità di probabilità ottenuti con l'algoritmo kernel density per i valori normalizzati di  $V_p$ ,  $\rho$  e  $V_s$  di tutto il dataset proveniente dai parametri elastici dei 5 pozzi nella zona di interesse. In blu è rappresentata la funzione di probabilità delle Shale, in rosso quella delle Brine Sand e in Verde quella delle Gas Sand.

RF fornisce un importante strumento di analisi del dato, la *Feature Importance*, che dà una misura di quanto ogni variabile sia capace di generare un set quanto più puro, ovvero con quanti più campioni appartenenti a una sola classe .

Per calcolare la Feature Importance è stato allenato un modello di RF con 100 alberi. Ad ogni split, ogni albero seleziona una variabile per dividere i dati in due zone. La feature scelta è quello che minimizza la funzione di Gini. Il numero di volte  $N_p$  che una variabile  $p$  viene scelta per dividere i dati è diviso per il numero  $N_{tot}$  di split eseguiti all'interno dell'albero. Questa operazione viene fatta sommando i valori da tutti gli alberi costruiti. La Feature Importance per una feature  $p$  sarà data da  $N_p/N_{tot}$ . I risultati sono illustrati in figura 4.5. Il grafico a barre mostra che  $V_p$  è stata la feature più scelta per la costruzione degli alberi.

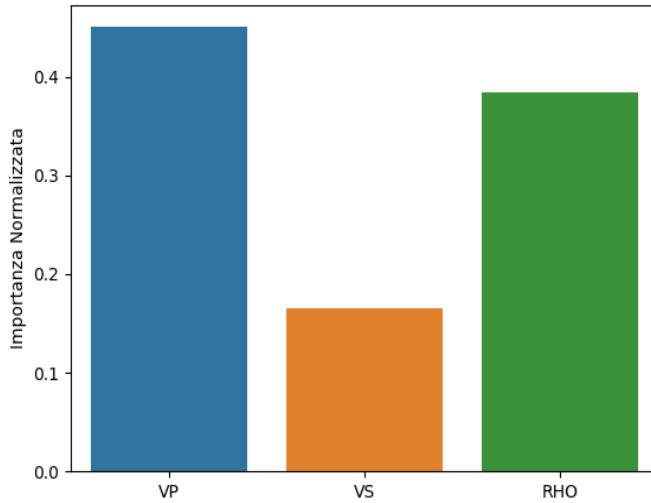


Figura 4.5: Valori di importanza normalizzata con l’uso di un modello di RF allenato su tutto il dataset proveniente dai parametri elastici dei 5 pozzi nella zona di interesse con 100 alberi

## 4.2 Effetto della Normalizzazione

Questa parte dell’indagine preliminare è volta a osservare il comportamento dei pesi  $w$  generati dagli algoritmi parametrici alla fine del processo di training con e senza la standardizzazione dei dati.

I coefficienti derivanti dal processo di training sono indice di quanto un determinato parametro influisce sulla predizione finale. Se si considerano una serie di variabili linearmente indipendenti  $x_1, x_2 \dots x_P$  appartenenti a un campione  $\mathbf{x}$  e dei coefficienti  $w_1, w_2 \dots w_P$  associati a una regressione lineare nella forma:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 \dots w_P x_P \quad (4.2)$$

I coefficienti indicheranno quanto la variazione di un’unità in  $x_p$  influenzerà il valore di  $\hat{y}$ .

Ci si aspetta quindi che variabili totalmente scorrelate dall’output saranno associate a pesi poco significativi, dal momento che hanno poca importanza nella predizione finale. I modelli di regressione lineare, come gli algoritmi qui utilizzati, aggiungono alla funzione oggetto un parametro di regolarizzazione che ha l’effetto di abbassare i valori dei coefficienti. Questi parametri di regolarizzazione si basano sull’assunzione di dati con distribuzione Gaussiana a media zero e varianza dello stesso ordine.

Per prima cosa si sono aggiunti ai valori di paratenza 6 variabili composte da rumore gaussiano random nel seguente modo:

- $N1 = \mathcal{N}(100, 20)$ .
- $N2 = \mathcal{N}(100, 35)$ .
- $N3 = \mathcal{N}(100, 40)$ .
- $N4 = \mathcal{N}(100, 55)$ .
- $N5 = \mathcal{N}(100, 60)$ .
- $N6 = \mathcal{N}(100, 80)$ .

Per verificare la variazione dei coefficienti dopo la normalizzazione Logistic Regression e Support Vector Classification sono stati allenati utilizzando tutti i dati di training abbinati alle variabili random e sono stati plottati i valori assoluti dei coefficienti.

I parametri di regolarizzazione per i tre algoritmi sono stati mantenuti di default rispettivamente  $C = 1$  per SVC e LR. Dopo l'allenamento SVC e LR restituiranno un array dei coefficienti per ogni classe nella forma:

$$\hat{y}_1 = w_{01} + w_{11}V_p + w_{21}V_s + w_{31}\rho + w_{41}N_1 + \dots + w_{91}N_6 \quad (4.3)$$

$$\hat{y}_2 = w_{02} + w_{12}V_p + w_{22}V_s + w_{32}\rho + w_{42}N_1 + \dots + w_{92}N_6 \quad (4.4)$$

$$\hat{y}_3 = w_{03} + w_{13}V_p + w_{23}V_s + w_{33}\rho + w_{43}N_1 + \dots + w_{93}N_6 \quad (4.5)$$

I valori assoluti di questi coefficienti sono stati sommati per ogni classe. In Figura 4.6 sono mostrati i valori dei pesi risultanti dalla somma dei loro valori assoluti per ogni classe. In alto a sinistra, si osservano i valori dei coefficienti di Logistic Regression per i dati non normalizzati.  $\rho$ , che è dato in  $g/cm^3$  e presenta valori nell'ordine di  $10^0$ , è associato a un coefficiente di circa 0.07, molto più alto rispetto a quello delle altre variabili. Questo effetto non è dato da una maggiore importanza di  $\rho$ , ma dalla differenza di 2-3 ordini di grandezza con gli altri parametri. La variabile random N1 ha un coefficiente maggiore rispetto alla  $V_p$ . È chiaro che questo grafico non rende una credibile interpretazione dei dati.

In alto a destra, l'allenamento dello stesso modello di ML è stato effettuato dopo la normalizzazione dei dati. In questo caso i coefficienti rappresentano meglio lo stato del problema. Le stesse premesse valgono per il modello di SVC con lo stesso valore di C. I valori relativi dei parametri dopo la normalizzazione sono comparabili a quelli di Logistic Regression.

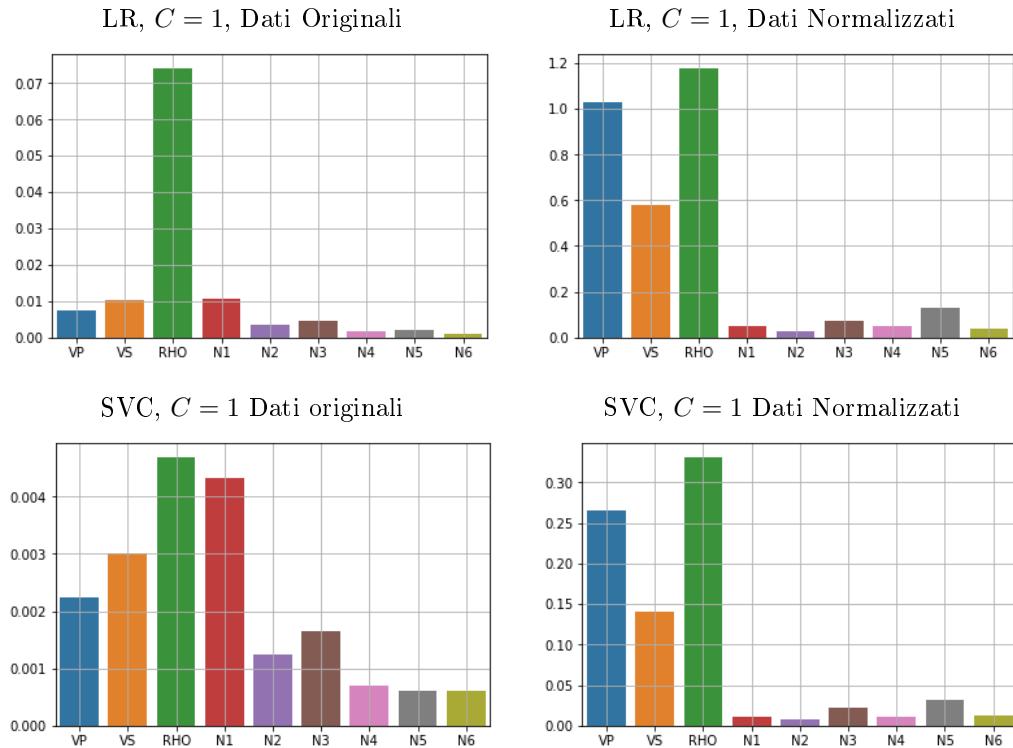


Figura 4.6: Grafici a barre dei valori dei coefficienti relativi ai parametri usati nel training. A destra i valori dei coefficienti prima della normalizzazione, a destra dopo la normalizzazione. Partendo dall'alto al basso gli algoritmi utilizzati sono: LR con  $C = 1$ , SVC con  $C = 1$  e infine NN con  $\alpha = 0.001$ .

La normalizzazione dei dati ha quindi permesso di ottenere risultati interpretabili dai modelli lineari,

La standardizzazione coinvolge anche le performance degli algoritmi. In tabella 4.4 sono illustrati i risultati della media dell'Accuratezza Bilanciata per ogni fold, per una K-fold Cross Validation con  $K = 10$ . I modelli utilizzati sono LR, SVC, KNN e NN.

Gli algoritmi parametrici su dati non normalizzati mostrano performance vicine alla predizione random 0.33, mentre sui dati normalizzati hanno performance molto più significative. Per quanto riguarda KNN l'Accuratezza Bilanciata non cala drasticamente come nei precedenti algoritmi. Esso non minimizza alcuna funzione oggetto e non fa uso di un parametro di regolarizzazione. Il calo delle performance nel suo caso è dato da bias nel calcolo delle distanze quando i dati non sono nello stesso ordine di grandezza.

Tabella 4.4: Risultati di una K-fold Cross Validation effettuata su dati non normalizzati e normalizzati, con diversi algoritmi di ML.

Algoritmo	Accuratezza Bilanciata (No norm.)	Accuratezza Bilanciata (Norm.)
LR	0.394	0.669
SVC	0.380	0.661
KNN	0.681	0.744
NN	0.333	0.734

Le stesse considerazioni non valgono per Random Forest. Questo tipo di algoritmo non è influenzato da variazioni di scala perché valuta volta per volta i parametri in modo univariato. I modelli lineari precedentemente allenati mostravano un valore del coefficiente di  $\rho$  più alto rispetto a quello di  $V_p$ , differentemente da quanto ricavato dalla FI in figura 4.5. Un paragone tra i due risultati non è un problema facile, dal momento che nei modelli parametrici la correlazione tra le variabili gioca un ruolo importante nel risultato finale. La correlazione tra due variabili è data da:

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y} \quad (4.6)$$

In figura 4.7 è mostrata la matrice di correlazione di Pearson tra  $V_s$ ,  $V_p$  e  $\rho$ ; questi ultimi parametri mostrano una proporzionalità diretta che può aver influito nel calcolo dei pesi nei modelli lineari.

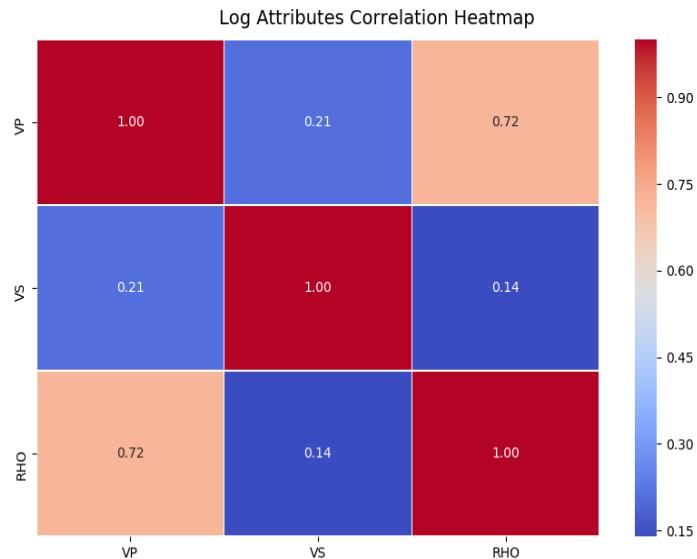


Figura 4.7: Matrice di correlazione tra le variabili di partenza calcolate con il metodo di Pearson.

### 4.3 Iper-parametri

In questa sezione si sono effettuate una serie di prove per osservare il comportamento degli algoritmi al variare dei loro iper-parametri di tuning. È stato eseguito il training dei cinque algoritmi su tutto il dataset di partenza, non includendo  $V_s$  per ottenere una visualizzazione

in 2D dei confini di decisione dei modelli al variare degli iper-parametri. Gli iper-parametri utilizzati sono:

- Logistic Regression , Linear SVC
  1. **C**, il parametro di regolarizzazione inverso in norma 2. Valori più alti di C daranno meno importanza al fattore di regolarizzazione.
- Neural Netowrk
  1.  **$\alpha$** , il parametro di regolarizzazione in norma 2 .
- K Nearest neighbors
  1. **K**, il numero di vicini utilizzati per predire la classe finale.
  2. **Weight**, booleano che indica se il voto dei vicini deve essere pesato per la loro distanza.
- Random Forest
  1. **N estimators**, il numero di alberi costruiti nel Random Forest.
  2. **Depth**, controlla la profondità degli alberi

### 4.3.1 Algoritmi Lineari

In figura 4.8 sono mostrati i boundaries dei modelli lineari. A sinistra per entrambi è stato scelto un C che assegna poca importanza al fattore di regolarizzazione, mentre a destra il peso assegnato alla funzione di Hinge e alla Log-likelihood è poco significativo. I modelli poco regolarizzati mostrano dissimilarità nei boundaries causate dalle differenti assunzioni di base delle funzioni utilizzate: La funzione di Hinge cerca di mantere l'iperpiano di separazione il più lontano possibile dai clusters mentre la log likelihood cerca i valori di  $V_p$  e  $\rho$  più probabili per ogni classe. Quando il fattore di regolarizzazione assume importanza entrambi gli algoritmi cercano il boundary più semplice in termine di distanza euclidea e i due modelli convergono a un unico classificatore. L'effetto della regolarizzazione infatti è quello di penalizzare valori alti dei coefficienti. Nelle tabelle 4.5 e 4.6 i coefficienti per un valore di  $10^{-8}$  sono di 5/6 ordini di grandezza inferiori rispetto ai corrispondenti per  $C = 10^8$  per entrambi i classificatori.

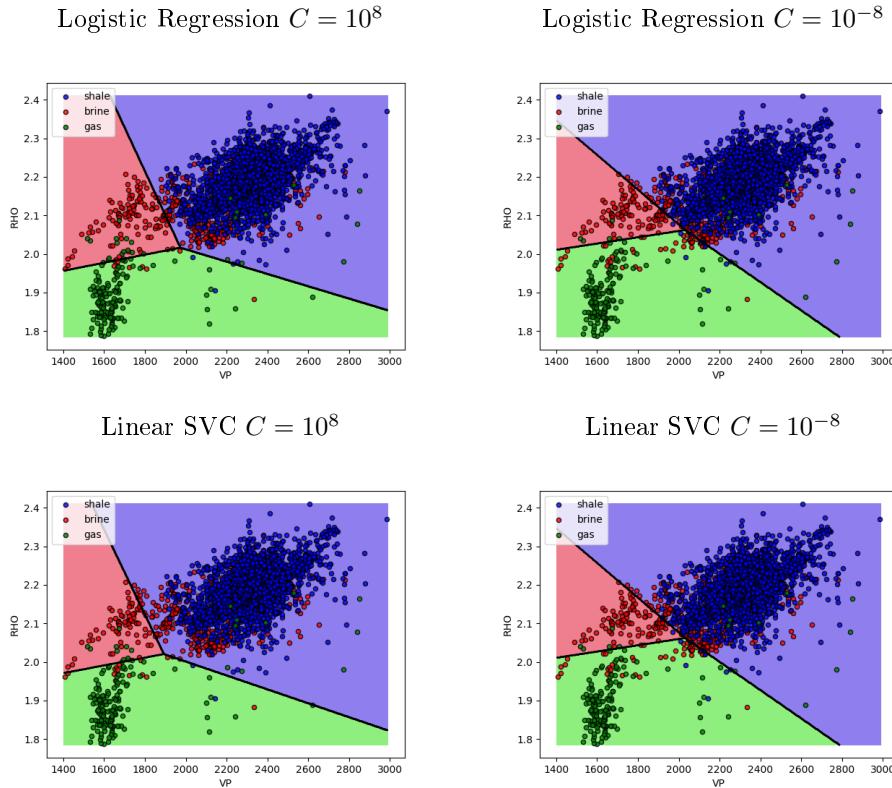


Figura 4.8: Boundaries per due modelli di Logistic Regression allenati con due differenti fattori inversi di regolarizzazione utilizzando come features  $V_p$  e  $\rho$ .

Tabella 4.5: Coefficienti per Logistic Regression per due valori di C.

	$10^8$	$V_p$	$\rho$	$10^{-8}$	$V_p$	$\rho$
Shale		1.239	1.052		1.08 $10^{-5}$	1.014 $10^{-5}$
Brine		-1.090	0.281		-3.768 $10^{-6}$	-5.729 $10^{-6}$
Gas		-2.577	-0.297		-6.378 $10^{-6}$	-5.165 $10^{-6}$

Tabella 4.6: Coefficienti per Linear SVC per due valori di C.

	$10^8$	$V_p$	$\rho$	$10^{-8}$	$V_p$	$\rho$
Shale		0.332	0.240		4.357 $10^{-5}$	4.058 $10^{-5}$
Brine		-0.256	0.385		-2.291 $10^{-5}$	-1.507 $10^{-5}$
Gas		-0.080	-0.623		-2.068 $10^{-5}$	-2.551 $10^{-5}$

### 4.3.2 Neural Network

In figura 4.9 due modelli di NN sono stati allineati tenendo fissa la funzione di attivazione, scelta come Rectified Linear Unit e il numero di Layers pari a 100. Al variare di  $\alpha$  si osserva un marcato effetto sulla complessità dei boundaries: quando il fattore di regolarizzazione è poco

importante il NN classifica i dati con una funzione articolata, soprattutto nei punti dove il dato è di difficile separazione. Quando il fattore di regolarizzazione è settato a valori alti i boundaries vengono linearizzati nel cercare una soluzione più generale al problema.

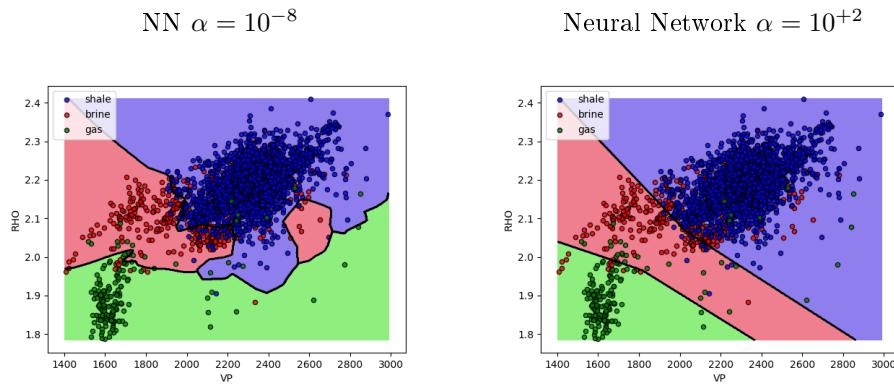


Figura 4.9: Boundaries per due modelli di NN allenati con su due differenti fattori di regolarizzazione utilizzando come funzione di attivazione Rectified Linear Unit e come features  $V_p$  e  $\rho$ .

### 4.3.3 Random Forest

Uno dei più importanti iper-parametri che riguardano Random Forest è la profondità massima degli alberi, che ha l'effetto di semplificare i boundaries dell'algoritmo. Tale iper-parametro controlla la complessità del modello di ML ed ha lo scopo di evitare basse performance nel dato di Cross Validation o di test. Questo effetto è illustrato In figura 4.10, dove a sinistra è stato allenato un modello di RF con 10 alberi, mentre al centro la profondità massima è stata impostata a 5. Il modello di RF con profondità 5 ha costruito delle zone nettamente distinte.

Un'altro iper-parametro che ha un significativo effetto nella predizione è il numero di alberi costruiti dal modello di RF. A destra RF è stato allenato con 200 alberi. Confrontando questo risultato con quello ottenuto con il modello di RF a 10 alberi si nota che nonostante entrambi i modelli mostrino boundaries fortemente irregolari, l'utilizzo di un numero maggiore di classificatori ha aumentato la risoluzione delle linee di boundary.

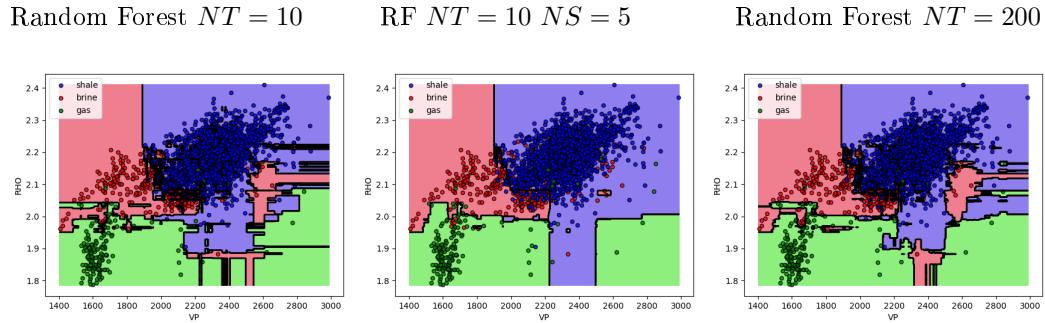


Figura 4.10: Boundaries per tre modelli di Random Forest allenati al variare degli alberi e della loro profondità, utilizzando come features  $V_p$  e  $\rho$ .

#### 4.3.4 K Nearest Neighbors

L'iper-parametro su cui si basano maggiormente i modelli di KNK è il numero di vicini K. Questo iper-parametro è in grado di cambiare radicalmente la funzione risultante da un'allenamento di un KNN. Generalmente un modello di KNN allenato con valori di K vicini a 1 restituisce boundaries molto irregolari e sharp che classificano molto bene i dati di training. Aumentando il numero di vicini la funzione tende a divenire più smooth a causa del contributo di campioni lontani dal punto da predire. Questo effetto è illustrato in figura 4.11 dove si osservano i risultati per due modelli allenati con  $K = 3$  e  $K = 30$ .

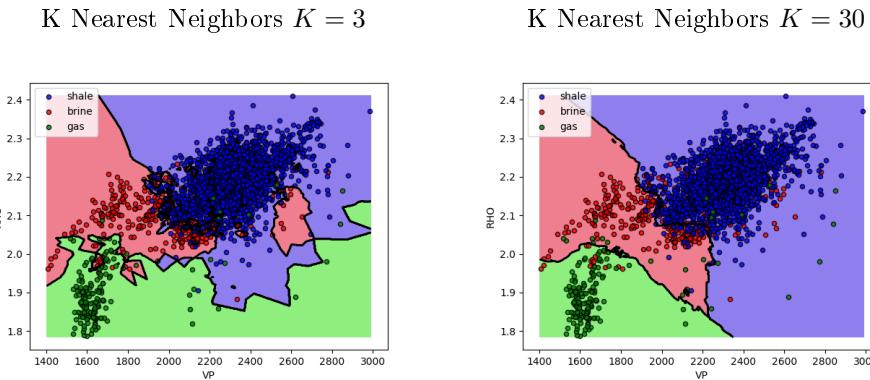


Figura 4.11: Boundaries per due modelli di NN allenati con due differenti valori di vicini utilizzando come parametri  $V_p$  e  $\rho$ .



## Capitolo 5

# Classificazione con Training Set alla scala dei log

In figura 5.1 è sintetizzato il workflow seguito in questo capitolo. Il dato dei 4 pozzi è stato diviso in Test Set(20%) e Training Set(80%) attraverso la stratification, ovvero mantenendo le proporzioni di campioni per ogni classe. Il programma python scritto per le classificazioni è descritto in appendice A.4. Per realizzare questi set, i valori dei campioni di tutti e 4 i pozzi sono stati riuniti in un'unica matrice, dalla quale in modo pseudo-random sono stati estratti i campioni da associare al Test Set.

Uno dei log di pozzo non è stato incluso nei dati usati nella Cross Validation per essere usato come blind test. La metrica su cui si basa la scelta del miglior modello di ML è l'Accuratezza Bilanciata, ovvero si cerca il modello di ML in grado di predire mediamente bene tutte e tre le litho-fluidfacies.

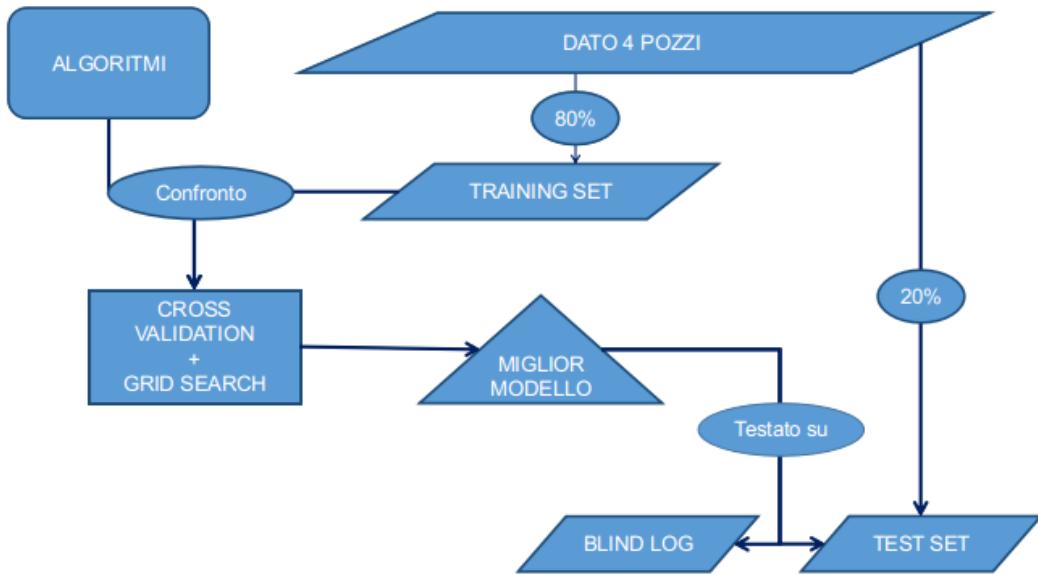


Figura 5.1: Flowchart delle operazioni svolte: Il dato di test è stato diviso dal dato di training attraverso stratificazione. Su quest'ultimo è stato svolto il training e il tuning degli algoritmi, da cui si deriverà un miglior modello di ML utilizzato per la predizione delle classi sul Test Set e su un pozzo blind.

Si assume che i parametri elastici dei due set provengano dalla stessa distribuzione di densità di probabilità. Per essere certi che questa assunzione sia rispettata sono state calcolate le distribuzioni di densità con l'algoritmo del kernel density adottato in precedenza. Training e test sono normalizzati con gli stessi valori di media e varianza calcolati sul Training Set. In figura 5.2 sono illustrate le funzioni di densità per ogni classe per le tre variabili. Da una visualizzazione grafica non si osservano sostanziali differenze in nessuna delle tre distribuzioni. Una sottile diffrenza risiede nell'irregolarità nella funzione di densità del Test Set a causa del più basso numero di campioni a disposizione.

Inoltre in tabella 5.1 è possibile osservare le percentuali di campioni per ogni classe nei dati di train e di test. La procedura di stratification ha mantenuto la proporzione dei campioni nei due dataset praticamente invariata.

Tabella 5.1: Proporzioni delle classi di litho-fluidfacies nei due set espressi in percentuale.

Set	Shale	Brine	Sand
Train	87.956%	9.042%	3.043%
Test	87.914%	9.066%	2.977%

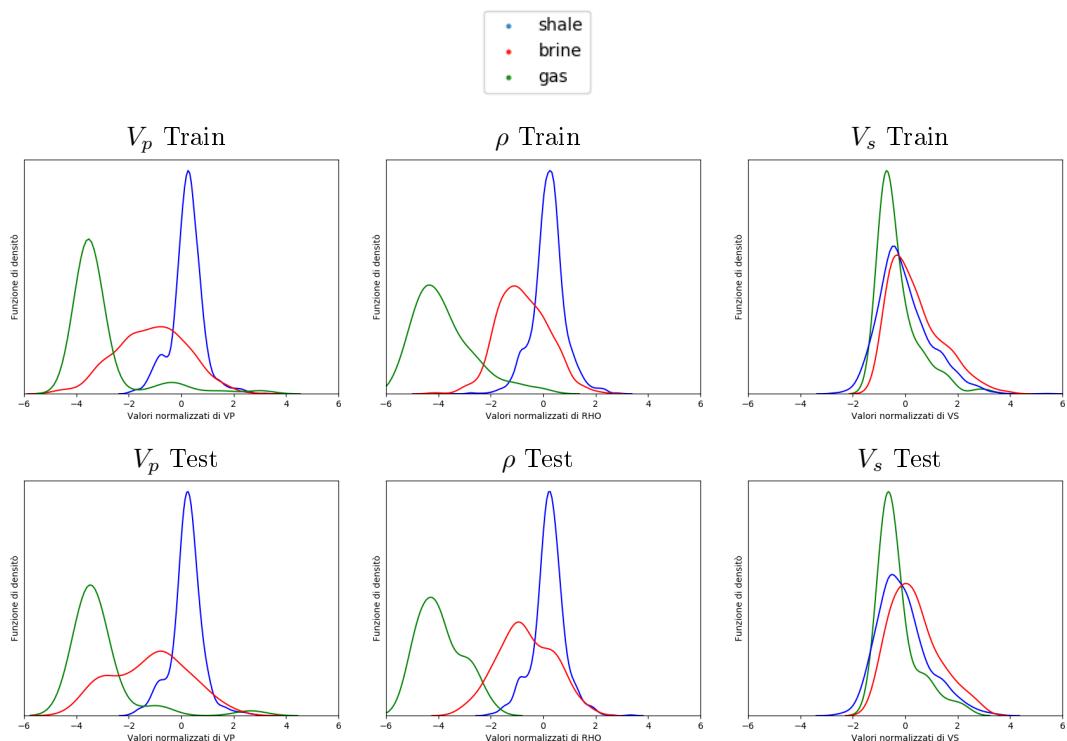


Figura 5.2: Funzioni di densità di probabilità ottenute con l'algoritmo kernel density per valori normalizzati di  $V_p$ ,  $\rho$  e  $V_s$  differenziate per i dati di training e di test. Le distribuzioni sono normalizzate per gli stessi valori di media e deviazione standard. Le linee blu, rosse e verdi sono indicate rispettivamente Shale, Brine Sands e Gas Sands.

## 5.1 Griglia degli iper-parametri

Per procedere con il Tuning degli algoritmi è stato necessario scegliere una griglia degli iper-parametri di partenza per il Grid-Search. In tabella 5.2 è mostrata la griglia scelta. Per NN il numero di Hidden Layers è stato mantenuto pari a 100 in tutte le prove. Questa scelta è derivata dal fatto che il numero dei neuroni e il parametro di regolarizzazione agiscono in modo inverso sulla complessità del modello di ML.  $\alpha$  non è stato esplorato oltre  $10^4$  perchè in base alla conoscenza acquisita da prove precedenti una regolarizzazione più importante non avrebbe portato a risultati migliori. Per quanto riguarda KNN dal momento che il tempo di training per questo algoritmo non è rilevante si sono utilizzati valori di  $K$  in un range da 1 a 50, con o senza l'utilizzo di pesi per le distanze che possono essere in norma 2 o in norma 1. Anche per i modelli lineari la ricerca è stata effettuata su un ampio range di valori di regolarizzazione. Il numero massimo di profondità  $D$  per RF è stato scelto in base alla massima lunghezza raggiunta senza imposizioni dai DT allenati nelle prove preliminari, che si è calcolata pari a 26. L'esplorazione del numero  $Nest$  di DT per un RF è stato impostato in un range da 10 a 250.

Tabella 5.2: Griglia degli iper-parametri di tuning

Algoritmo	P1	P2	P3
KNN	$K = \{1, 2, \dots, 50\}$	$W = [\text{Uniformi}, \text{Pesati}]$	Pesi=[N1,N2]
LR	$C = \{10^{-13}, 10^{-12}, \dots, 10^{13}\}$		
SVC	$C = \{10^{-13}, 10^{-12}, \dots, 10^{13}\}$		
NN	$\alpha = \{10^{-13}, 10^{-12}, \dots, 10^4\}$		
RF	$D = \{5, 10, \dots, 25, \text{None}\}$	$Nest = \{10, 30, \dots, 250\}$	

## 5.2 Scelta del modello

La K-fold Cross Validation è stata impostata con numero di cartelle pari a 10. È da specificare che ogni operazione di pre-processing del dato, che in questo caso comprende solo la normalizzazione, va effettuata ogni volta che usa una cartella differente come Validation Set. Attuare trasformazioni del dato su tutto il set prima della Cross Validation introdurrebbe un bias nel processo. Inoltre media e deviazione standard sono ricalcolate sul dato di training e applicate per trasformare i dati di validation e di test.

I risultati del tuning sono illustrati in tabella 7.3. Si osserva che entrambi i modelli lineari hanno ottenuto performance migliori con parametri di regolarizzazione alti. Come già visto

in precedenza ci si aspetta che i 2 modelli siano molto simili se non identici. La scelta di tali iper-parametri risiede nell'impossibilità dei modelli lineari di produrre boundaries affidabili sulla base delle funzioni oggetto dei rispettivi algoritmi. Per quanto riguarda il modello di KNN migliore utilizza un peso in norma 1 sui cast dei campioni, questo potrebbe aver mitigato l'effetto dello sbilanciamento del dataset.

Tabella 5.3: Valori degli iper-parametri per i migliori modelli degli algoritmi

Modello di ML	P1	P2	P3
KNN	$K = 4$	$W = \text{Pesati}$	N1
LR	$C = 10^{-6}$		
SVC	$C = 10^{-7}$		
NN	$\alpha = 10^{-2}$		
RF	$D = 20$	$N_{est} = 250$	

In figura 5.4 sono state calcolate le Confusion Matrices date dalla somma dei risultati ottenuti su ogni fold, normalizzata per il numero totale di campioni per ogni classe. Si tenga presente che i valori nella diagonale così normalizzati sono uguali al Recall.

Gli errori più comuni per tutti gli algoritmi risiedono nella confusione tra i campioni di Brine Sands e i campioni di Shale. Come già osservato precedentemente le funzioni di densità delle due classi si sovrappongono in modo evidente, spiegando i risultati ottenuti. Il basso Recall sulle Brine è dovuto al fatto che esse hanno una distribuzione molto ampia, con campioni che si sovrappongono anche al cluster dei campioni di Gas. Tutti i modelli sono stati molto efficaci nell'identificare i campioni di Gas Sands e di Shale. Dall'osservazione di tutte e 5 le Confusion Matrices, si possono distinguere i differenti risultati ottenuti dai modelli lineari e non lineari: i primi non sono in grado di produrre delle linee di boundary complesse e identificano solo il 20% delle Brine Sands, mentre i secondi sono in grado di restituire funzioni che riconoscono fino al 62% di esse. In quest'ultimo caso però l'aumento del Recall sulle Brine Sands è accompagnato da una diminuzione del Recall sulle altre due classi. Si osserva inoltre come LR e SVC condividono la stessa Confusion Matrix a causa del basso valore di C che aumenta il contributo del fattore di regolarizzazione. A livello di Accuratezza i risultati mostrano un'elevata capacità di tutti i modelli di ML di riconoscere un gran numero di campioni: si passa dal 92,1% dei campioni riconosciuti dai modelli lineari al 94,5% dei campioni riconosciuti dai modelli non lineari. Si noti infine come le confusion matrices dei vari modelli siano concordi nelle predizioni: essi dimostrano una consistenza nei risultati ottenuti.

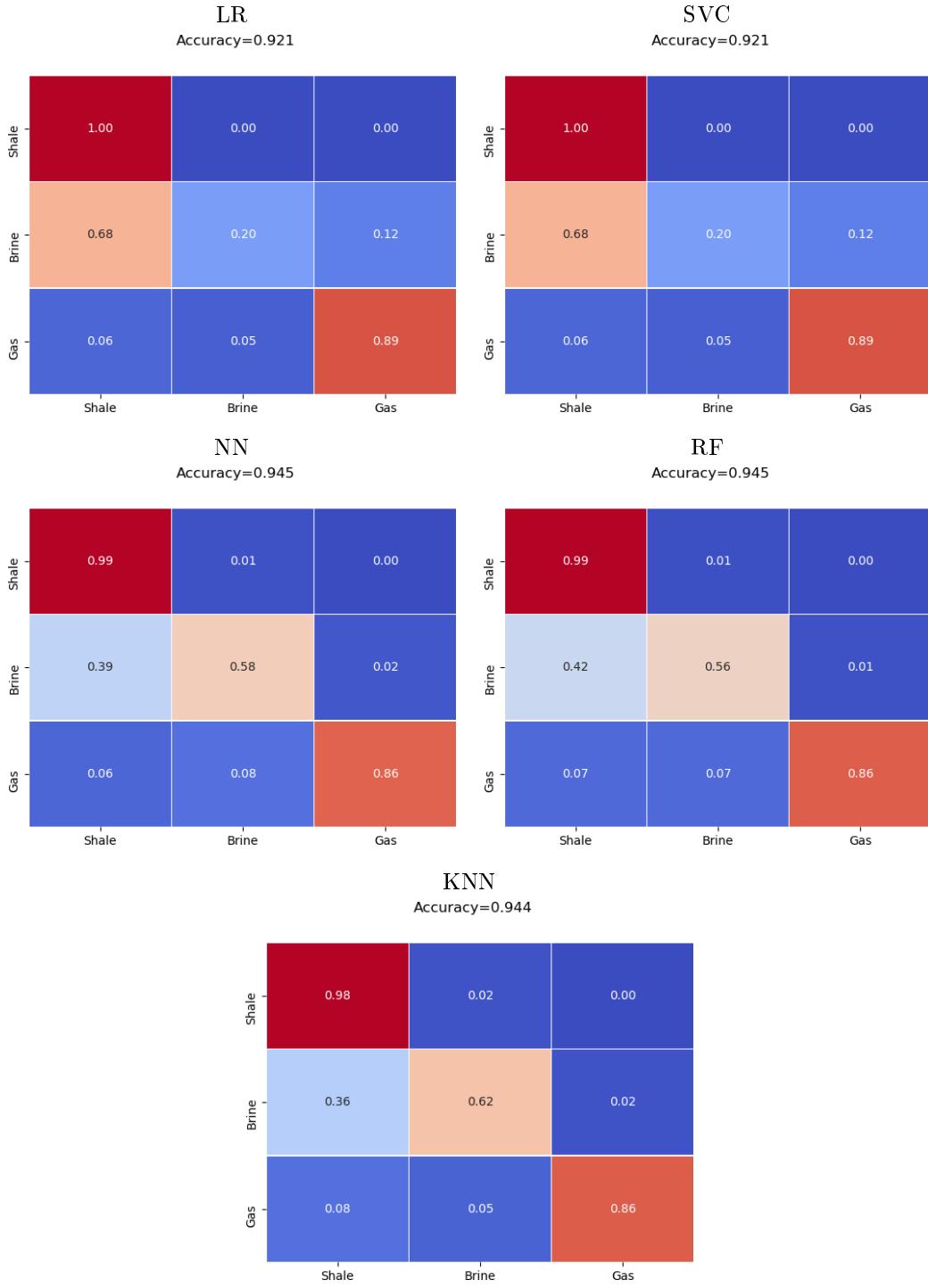


Figura 5.3: Confusion Matrices ricavate dalla somma delle performance su ogni fold della Cross Validation. All'interno sono mostrati i valori normalizzati per il numero dei campioni per ogni classe. Sopra a ogni CM è riportata la percentuale di campioni correttamente classificati.

Un confronto per i valori di Accuratezza Bilanciata per risultati della Cross Validation è mostrato nel Box-plot in figura 5.4. Si osserva il distacco tra i modelli di ML lineari e quelli non lineari. Questi ultimi hanno formato mediamente meglio sulle 3 classi, data la loro capacità di creare linee di boundary complesse nelle zone più incerte, soprattutto nella zona tra Brine Sands e Shale. NN e KNN in particolare hanno avuto le migliori performance.

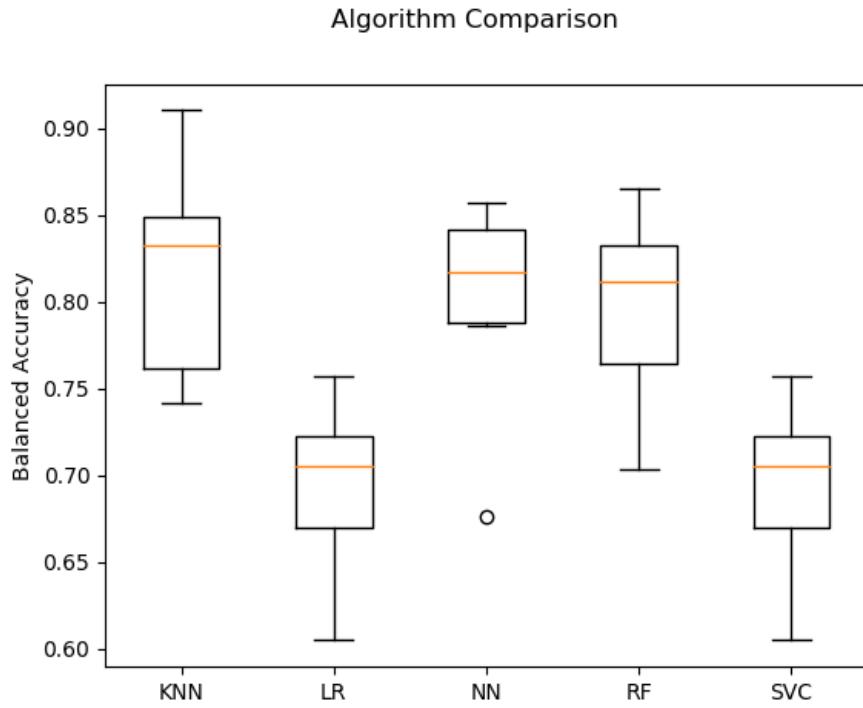


Figura 5.4: Box-plot dei valori di Accuratezza Bilanciata derivati delle performance sulle 10 cartelle della Cross Validation per le migliori combinazioni di parametri di ogni algoritmo. Il Box-plot riporta in arancione la mediana, il box rappresenta la zona dal primo al terzo quartile, mentre le linee arrivano fino ai valori minimi e massimi. I cerchi rappresentano gli outliers.

In tabella 7.2 sono specificati i valori di media e deviazione standard sui risultati delle 10 fold per ogni modello di ML. Il modello di KNN è quello che ha fornito le performance migliori ed è stato quindi scelto come best model. I risultati prodotti dai modelli lineari sono associati a una deviazione standard minore rispetto ai modelli non lineari, indice della robustezza delle loro predizioni.

Tabella 5.4: Valori di media e deviazione standard di Accuratezza Bilanciata calcolata sulle performance nelle folds singole per ogni modello di ML tunato con la Cross Validation

Modello di ML	Media	STD
KNN	0.815993	0.052965
LR	0.693366	0.041612
SVC	0.699700	0.041612
NN	0.806172	0.048123
RF	0.799264	0.041612

IL miglior modello di KNN è stato riallenato su tutte e 10 le fold ed è stato usato per eseguire una classificazione sul Test Set, il quale come già visto deriva dall'estrazione di campioni pseudo-random dai dati dei 4 pozzi e mantiene la stessa proporzione del numero di campioni di ogni classe. Lo scopo di questa classificazione è quello di misurare le capacità di generalizzazione del modello di ML prodotto.

I risultati sono esposti nella Confusion Matrix in figura 5.5 e nelle rispettive metriche in tabella 7.3. Si può affermare che: Gli elevati valori di F1-score per la classe delle Shale (96,3%) e del Gas (95,5%) indicano predizioni di alta qualità per queste due classi, ovvero che per esse non solo sono stati riconosciuti un gran numero di campioni, ma che le predizioni su questi campioni sono affidabili. Più precisamente, il modello di KNN è stato in grado di classificare correttamente il 97,4% dei campioni di Shale, ed il 95,2% che il modello di KNN ha predetto Shale, lo ha fatto in modo corretto.

In modo analogo vengono riconosciuti il 94% dei campioni di gas. La precisione molto elevata (96,9%) su questa classe indica che raramente un'altra classe verrà confusa per Gas. Consistentemente a quanto già visto, il modello di ML ha ottenuto performance inferiori per le Brine Sands. L'F1 score, che in questo caso è del 60%, indica come questa classe sia stata predetta in modo meno accurato.

Vengono riconosciute solo il 53,4% delle Brine del Test Set. Circa il 46% delle Brine Sands viene classificata come Shale, indice della forte sovrapposizione tra le due classi mantenuta nel Test Set. Ogni volta che il modello di ML predice Brine, il 68,3% delle volte fornisce una predizione corretta.

Questa Confusion Matrix è analoga a quella precedentemente vista nella Cross Validation: bisogna ricordare che entrambi i set provengono dalla stessa distribuzione di probabilità e che tutte le precedenti considerazioni valgono per predizioni su dati che anch'essi provengono dalla stessa distribuzione.

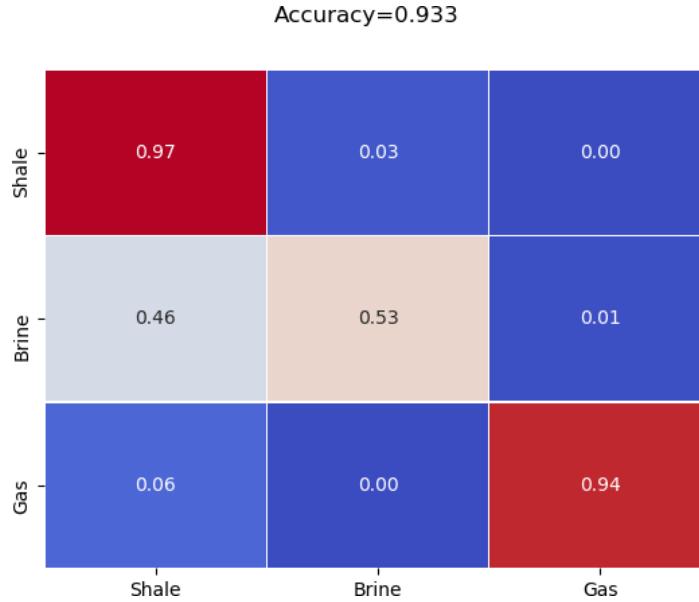


Figura 5.5: Confusion Matrix ricavata dal Test Set, i cui valori sono normalizzati per il numero di campioni in ogni classe.

Tabella 5.5: Metriche per le singole classi di Precision, Recall e F1-score

	Precision	Recall	F1-score
Shale	0.95224	0.97454	0.96326
Brine	0.68354	0.53465	0.60000
Gas	0.96970	0.94118	0.95522

Il valore di Accuratezza Bilanciata sul Test Set è dell' 81,673%, inferiore al valore di Accuratezza che ha restituito un valore superiore al 93%, indicando che sebbene il modello abbia predetto correttamente un gran numero campioni, questi campioni non sono distribuiti equamente su tutte le classi.

### 5.3 Classificazione su dati del pozzo blind

Prima di procedere con il test sul pozzo blind sono state calcolate le funzioni di densità di probabilità relative ai valori del pozzo blind. In figura 5.6 si osservano i risultati del calcolo delle funzioni di densità dove i dati del blind sono stati scalati per media e varianza calcolata sui dati di training.

Osservando la funzione di densità ogni classe si osserva che alcune di queste vengono preservate, mentre altre variano. In particolare la distribuzione delle Shale varia sensibilmente tra il Trainig Set su cui sono stati allenati i modelli di ML e il pozzo blind su cui si utilizzerà il modello di KNN precedentemente allenato. La distribuzione dei campioni di Gas invece ricade più o meno negli stessi valori del dato precedente. In  $V_s$  Log si nota che i campioni

di Brine Sand e Shale sono decisamente più sovrapposti lungo questa variabile. Le funzioni di densità delle Brine Sand in  $V_p$  e  $\rho$  per le Brine Sand ricadono in un range di valori più stretto. Inoltre tutte le distirbuizioni in  $V_p$  e  $\rho$  nel blind sono shiftate verso valori più bassi.

Si vogliono quindi confrontare i risultati ottenuti classificando il pozzo blind con quelli ottenuti sulla classificazione precedentemente effettuata sul Test Set. Dall'ossservazione grafica delle funzioni di densità ci si aspetta che le differenze osservate producano dei risultati diversi rispetto a quelli ottenuti in precedenza, causati da un bias nelle prorpietà elastiche.

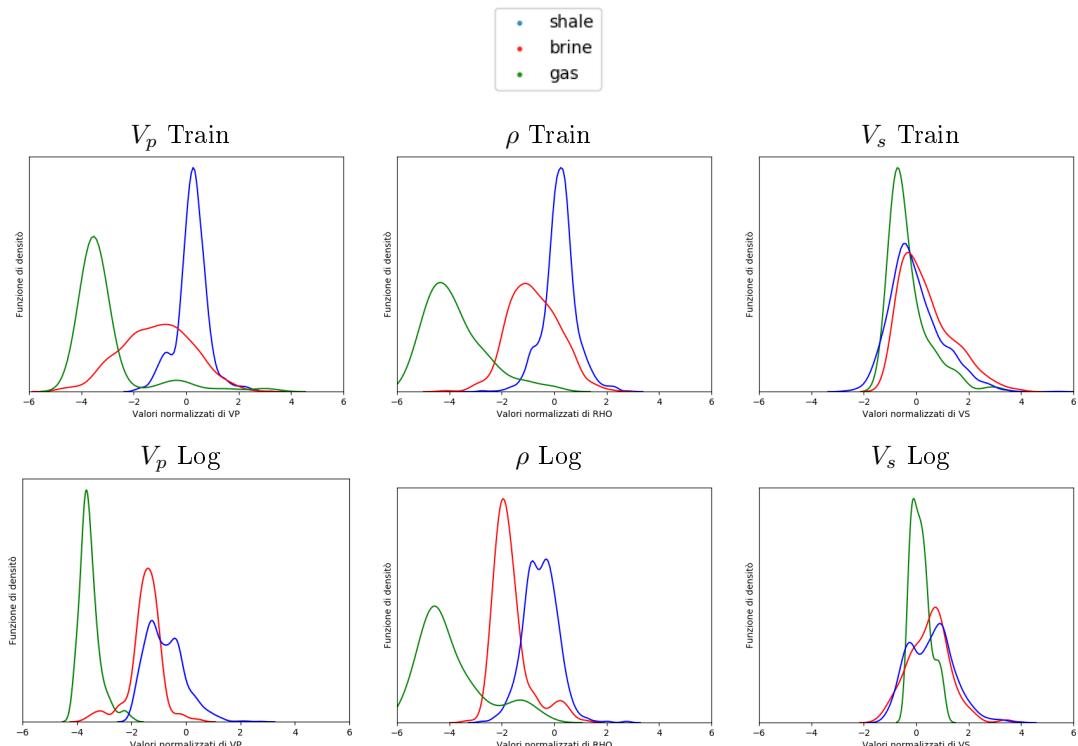


Figura 5.6: Funzioni di densità di probabilità ottenute con l'algoritmo kernel per i valori normalizzati di  $V_p$ ,  $\rho$  e  $V_s$ . I valori di media e deviazione standard sono calcolati sul dato di training e applicati al log di prova. Le linee blu, rosse e verdi sono indicano rispettivamente Shale, Brine e Gas Sand.

Successivamente si è svolta la classificazione sul pozzo blind utilizzando il modello di KNN. In figura 5.7 è mostrata la Confusion Matrix relativa a questa classificazione. Questa volta si nota un sostanziale equilibrio nel Recall delle tre classi. In percentuale sono stati riconosciuti più campioni di Brine Sands ma meno campioni di Shale e di Gas: un buon numero dei campioni appartenenti a queste ultime due classi sono stati assegnati alla classe delle Brine Sands.

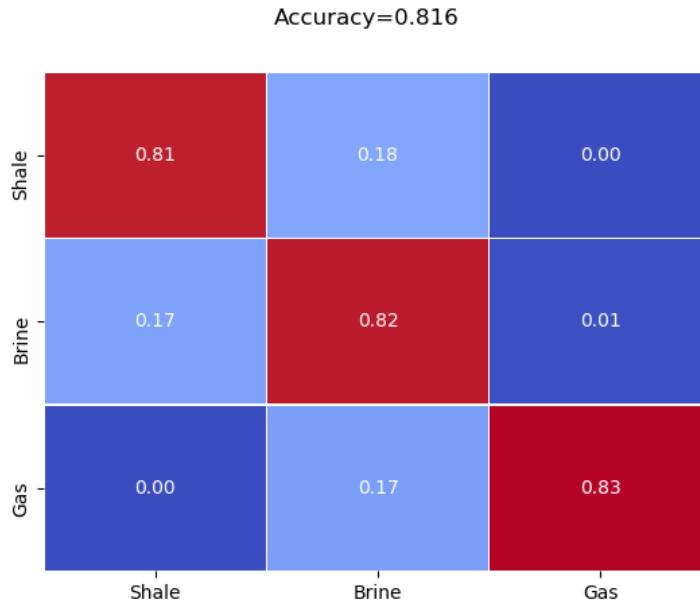


Figura 5.7: Confusion Matrix calcolata sul log usato come log blind, non presente nei dati di training. In alto è presente il numero di campioni riconosciuti sul totale

Ad ogni modo come indicato in tabella 5.6 le predizioni per le Brine Sand sono poco informative: sebbene il Recall indichi che l'82% delle Brine Sands sono state predette, esse sono associate a una Precision del 45%. L'F1 score confrontato con quelli osservati nel Test Set indica comunque un minore bilanciamento di Prec e Rec in tutte le classi.

Sempre richiamando un confronto con il Test Set, recuperato dalla stessa distribuzione del train, il modello di KNN ha riconosciuto meno campioni sul totale. Ad ogni modo si nota una robustezza nel valore di Accuratezza Bilanciata di 82,3038% leggermente superiore a quello calcolato sul Test Set. Un'illustrazione grafica della predizione si ritrova in figura 5.8.

Tabella 5.6: Metriche per le singole classi di Precision, Recall e F1-score

	Precision	Recall	F1-score
Shale	0.95633	0.81413	0.87952
Brine	0.45631	0.81739	0.58567
Gas	0.97037	0.83439	0.89726

In figura 5.12 il pozzo predetto dal modello di KNN e quello originale sono confrontati. Da queste immagini si comprende meglio come il modello di ML abbia riconosciuto gran parte delle Brine Sand, ma abbia predetto Brine Sand con troppa confidenza. Questa classificazione, sempre dovuta alla variazione della proprietà statistiche dei parametri elastici, non è concorde con le previsioni fatte sulla base del Test Set, dove lo stesso modello di KNN

ha riconosciuto circa il 53% delle Brine Sand, ma con una Precision del 68% che qui scende al 45%.

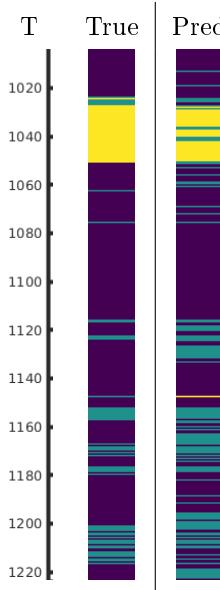


Figura 5.8: Confronto tra dati osservati e predetti per i dati provenienti dal pozzo blind.

Discutendo i risultati ottenuti, si puo affermare che il best model di KNN abbia mantenuto una media di score sulle tre classi comparabile con quella sul Test Set, ma gli errori sul pozzo di prova sono differenti da quelli sul Test Set: molti campioni di Shale e di Gas Sand sono stati classificati come Brine Sand. Inoltre si assiste a un peggioramento della qualità della predizione: i valori di F1 score per tutte e tre le classi sono inferiori a quelli sul Test Set. La perdita di predizioni corrette in Shale ha causato una netta diminuzione dell'Accuratezza, pari ora a all'81,6%.

## 5.4 Inversione e classificazione dati sismici sintetici ricavati dal pozzo blind

I parametri elastici del pozzo blind sono stati impiegati nel forward modeling sismico per produrre un sismogramma sintetico convoluzionale utilizzando un ondina di Ricker a 50Hz. Il sismogramma è stato utilizzato come dato osservato  $d_{obs}$  in un'inversione AVO che restituisce una distribuzione a posteriori  $p(m|d)$  dei parametri elastici. L'inversione è effettuata due volte: nella parametrizzazione  $V_p$ ,  $V_s$  e  $\rho$  e in quella in  $I_p$ ,  $I_s$  e  $\rho$ . La soluzione map di queste inversioni e alcune realizzazioni, estratte con metodo Montecarlo, sono classificate utilizzando il modello di KNN ricavato dall'allenamento sul Training Set alla scala dei log. A causa di diversi fattori, quali il mal-condizionamento del problema inverso, il rumore e la perdita di

risoluzione dovuta all'effetto filtro causato dall'ondina sorgente, non ci si aspetta un'elevata performance della classificazione rispetto alla classificazione sui parametri elastici del pozzo alla scala dei log.

Si vuole indagare se questo modello di KNN sia in grado di classificare correttamente almeno il livello saturo a gas derivante dai parametri elastici della  $p(m|d)$ .

#### 5.4.1 Inversione parametrizzata nelle velocità e classificazione dati sismici sintetici ricavati dal pozzo blind

Prima di procedere con l'inversione del log, sia i parametri elastici che quelli petrofisici sono stati convertiti a scala sismica. Per evitare perdita di informazione dovuta all'aliasing, il dato di log ha subito i seguenti step:

1. I dati di log in profondità sono riportati alla scala dei tempi. Questo procedimento è stato effettuato con l'ausilio delle velocità  $V_p$
2. sono stati ricampionati alla massima frequenza di campionamento
3. sono stati filtrati con un filtro passa basso di ordine 150 a una frequenza di taglio di  $250Hz$
4. È stata eseguita un'interpolazione tra il dato filtrato e un'asse dei tempi a 2ms.

Per classificare le litho-fluidfacies associate ai nuovi parametri elastici ricavati al termine del ricampionamento è stata eseguita nuovamente la classificazione con gli stessi valori cut-off usati in precedenza.

Successivamente è stato generato il low frequency model su cui si basa l'inversione utilizzando una finestra a media mobile di 25 campioni. Una volta calcolato il sismogramma attraverso il forward modeling si è aggiunto un rumore random gaussiano con deviazione standard di 0.02. Gli angoli scelti per ricavare il dato sismico osservato sono :  $0^\circ, 5^\circ, 10^\circ, 15^\circ, 20^\circ, 25^\circ$  e  $30^\circ$ . Il risultato delle inversioni è mostrato in figura 5.9.

Il modello vero, mostrato con una linea nera è stato approssimato particolarmente bene in  $V_p$  e  $\rho$ , dalla soluzione map, rappresentata dalla linea rossa mentre in  $V_s$  l'inversione ha restituito valori sempre vicini al low frequency model. I dati del sismogramma predetti a partire dalla soluzione map dei parametri elastici hanno ricostruito il sismogramma vero con una buona affidabilità. Intorno ai 1080 ms e ai 1120 si osserva come l'inversione sia guidata da  $V_p$ . Questo effetto è dovuto al malcondizionamento sul parametro  $\rho$ , il quale nell'inversione è il meno influente tra i parametri elastici, e per tale motivo il risultato seguirà le informazioni a priori ed in particolare le informazioni di correlazione tra  $V_p$  e  $\rho$  in  $\Sigma_0$ .

Si osserva inoltre in  $V_p$  e in  $\rho$  come la map non sia in grado di ricostruire adeguatamente l'informazione ad alta frequenza tra 1150 e 1220 ms.

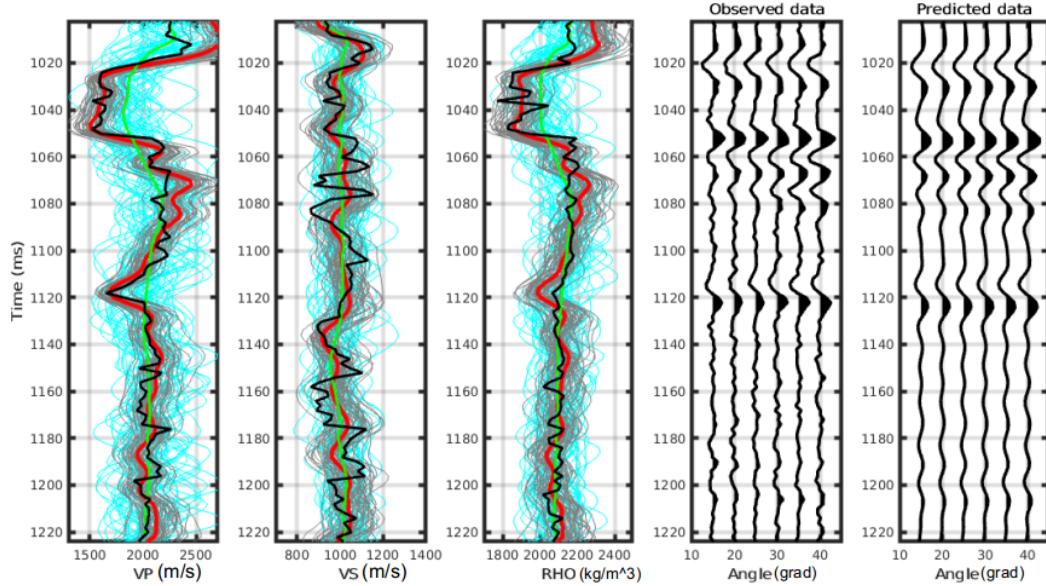


Figura 5.9: Risultati inversione Bayesiana AVO sul sismogramma generato dal pozzo blind. La curva nera indica i valori veri, quella rossa la Maximum A Posteriori(MAP) e quella verde il modello low-frequency. Inoltre le curve grigie sono altre realizzazioni della distribuzione a posteriori  $p(m|d)$ , mentre quelle azzurre sono le realizzazioni del modello a priori  $p(m)$ .

Il confronto tra le funzioni di densità di probabilità dei parametri elastici del dato di training, quelle dei parametri elastici della soluzione map e quelle dei parametri elastici del pozzo blind sono mostrati in figura 5.10.

La funzione di densità di probabilità delle variabili elastiche calcolate a partire dal dato sismico sintetico è differente rispetto a quella del log blind originale. In particolare le code delle distribuzioni nel dato invertito sono più corte e i valori di media e varianza delle funzioni risultano alterate. Si osserva che il parametro  $V_s$  risultato della map fornisce pochissima informazione sulla classificazione. Questo può essere dovuto ad insufficiente informazione del modello a priori e dalla poca determinazione che  $V_s$  ha nel riprodurre il dato sintetico.

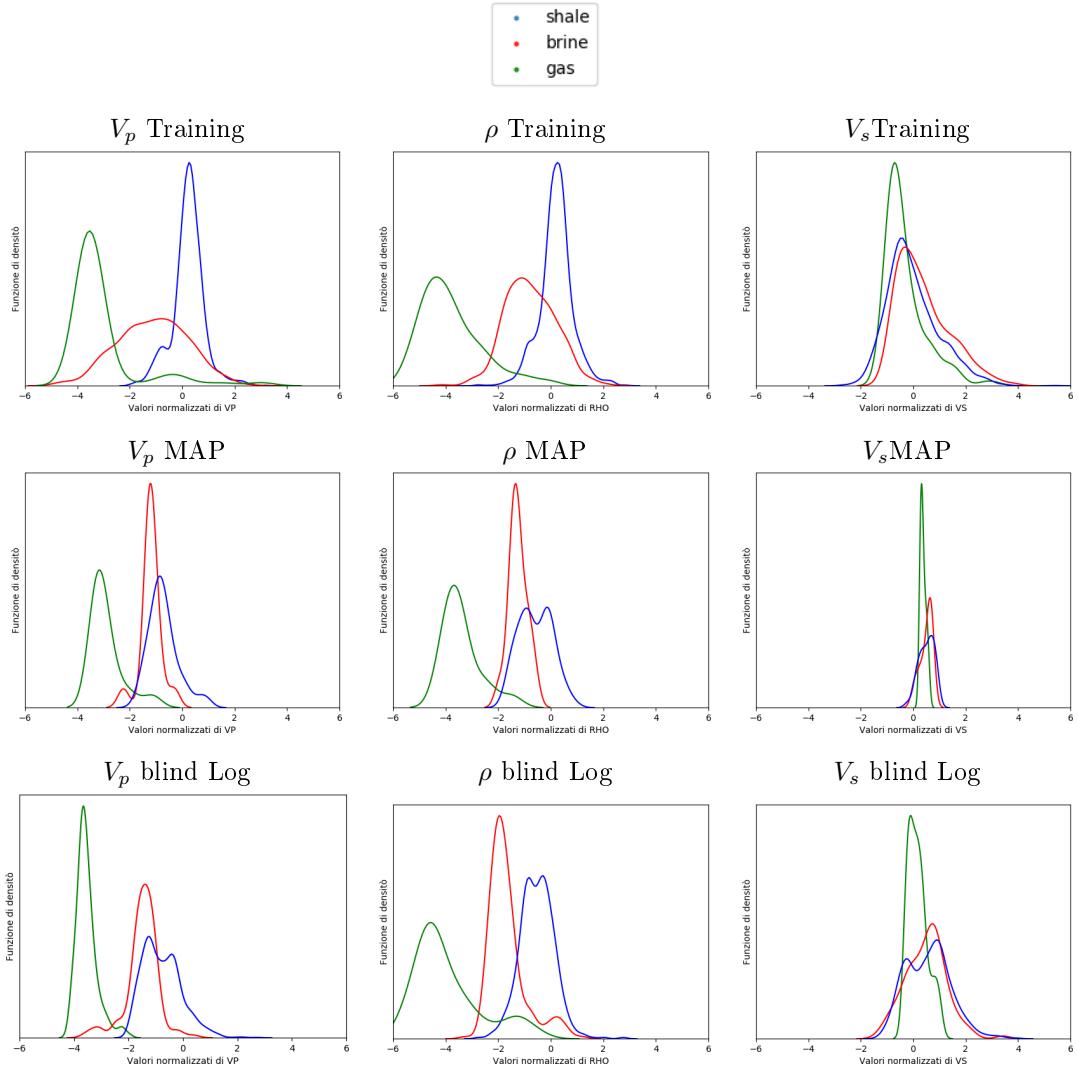


Figura 5.10: Funzioni di densità di probabilità per i valori normalizzati di  $V_p$ ,  $\rho$  e  $V_s$  normalizzati utilizzando la deviazione standard e la media calcolati sul Training Set. La prima riga riguarda le funzioni di densità dei valori dei parametri elastici usati per allenare il modello di KNN, la seconda illustra quelle generate dalla MAP mentre l'ultima riga illustra le funzioni di densità di probabilità ottenute dal log originale. Le linee blu, rosse e verdi sono indicate rispettivamente Shale, Brine Sands e Gas Sands.

La maximum a posteriori dell'inversione AVO è stato utilizzato per testare nuovamente il best model di KNN. Sono state inoltre estratte dalla distribuzione a posteriori  $p(m|d)$ , 3 simulazioni Montecarlo per testare il modello su parametri elastici ottenuti da modelli statisticamente possibili. Sono stati rimossi da tutte le soluzioni i primi 7 campioni, soggetti a un artefatto derivato dall'utilizzo della finestra a media mobile utilizzata per ricavare il low frequency model.

In figura 5.11 la Confusion Matrix riferita alla classificazione sulla soluzione map riporta

un'elevato numero di sbagli tra i campioni di Shale e quelli di Brine Sand: l'81% dei campioni di Brine Sands sono stati classificati come campioni di Shale, viceversa il 12% di Shale sono state classificate come Brine Sands. IL modello di KNN ha mantenuto una buona performance sui campioni di Gas Sands riconoscendone il 94%. Come visualizzabile in tabella 5.7 l'F1-score evidenzia la scarsa qualità delle predizioni sui campioni di Brine Sands rispetto alle altre due classi.

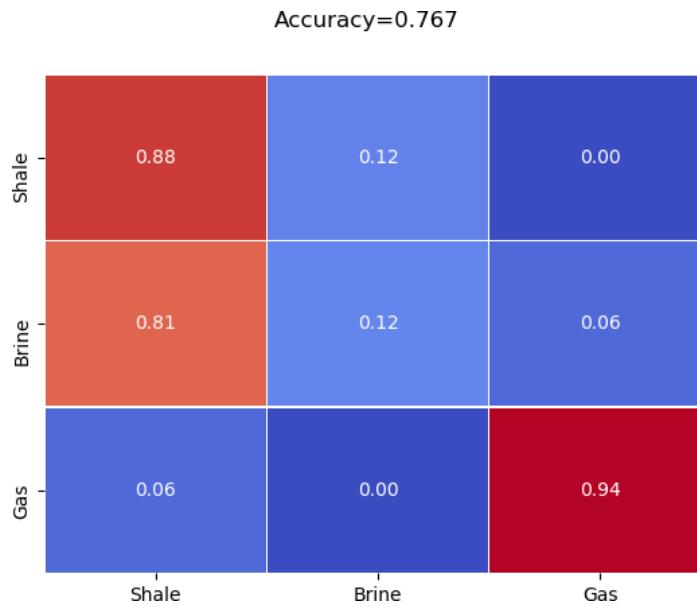


Figura 5.11: Confusion Matrix relativa alle predizioni sui parametri elastici ottenuti con la map dell'inversione AVO

Tabella 5.7: Metriche per le singole classi di Precision, Recall e F1-score

	Precision	Recall	F1-score
Shale	0.80822	0.88060	0.84286
Brine	0.20000	0.12500	0.15385
Gas	0.93750	0.93750	0.93750

Osservando il confronto tra classi predette e osservate in figura 5.12 si nota l'alta variabilità nella classificazione degli strati di Brine Sands: Le classificazioni sulle realizzazioni della distribuzione a posteriori variano soprattutto per la disposizione dei livelli sabbiosi saturi in acqua, mentre sono tutti più o meno concordi nella posizione dello strato di gas. Si nota come intorno ai 1120 ms, dove si ha un abbassamento della velocità delle onde P e della densità, il modello abbia erroneamente predetto in tutte le realizzazioni uno strato di gas. Infine rispetto alle predizioni sui parametri del log blind, una perdita di risoluzione nella classificazione dovuta al diverso range in frequenza del log e della scala sismica. In tutte le classificazioni lo strato di gas è stato sempre identificato.

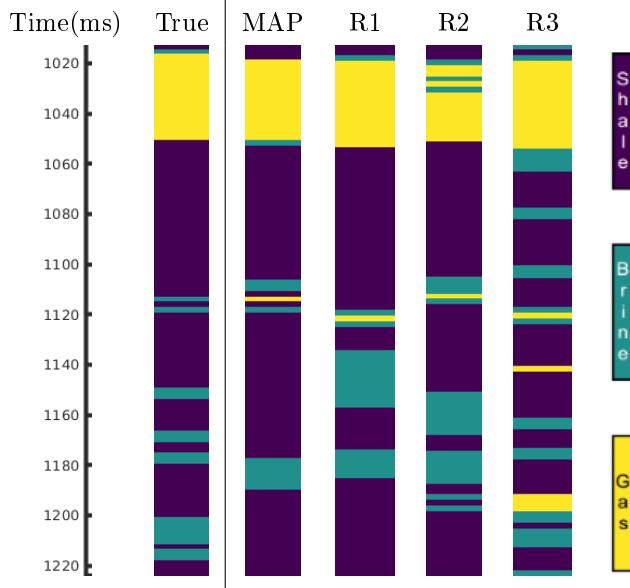


Figura 5.12: Confronto tra fluid-lithofacies vere, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(MAP), che su tre realizzazioni della distribuzione a posteriori  $p(m|d)$ .

Per quanto riguarda la classificazione su proprietà invertite si può concludere che:

- Le metriche relative alle Brine Sands indicano una scarsa performance, inoltre le classificazioni sulle realizzazioni a posteriori non sono concordi nella definizione degli strati di Brine Sands. Si deduce quindi che il modello di KNN non è in grado di classificare correttamente questa classe dai parametri elastici ricavati con l'inversione AVO.
- Il classificatore è stato in grado di predire correttamente il livello saturo a gas sia nella soluzione map, sia nelle plausibili soluzioni a posteriori estratte con metodo Montecarlo dalla  $p(m|d)$

#### 5.4.2 Inversione AVO parametrizzata nelle impedenze e nella densità e classificazione dati sismici sintetici ricavati dal pozzo blind

L'inversione è stata nuovamente eseguita con una parametrizzazione nell'impedenza acustica  $I_p$ , nell' impedenza  $S$   $I_s$  e nella densità  $\rho$ . In questo caso a livello teorico si possono avere risultati migliori:  $I_p$ ,  $I_s$  e  $\rho$  in questa parametrizzazione sono massimamente ortogonali [22].

I risultati relativi all'inversione sono mostrati in figura 5.13. Per effettuare questa inversione si è modificata l'equazione che lega il coefficiente di riflessione PP ai parametri elastici.

$$C_{pp}(\theta) = a_\alpha(\theta) \frac{\Delta\alpha}{\bar{\alpha}} + a_\beta(\theta) \frac{\Delta\beta}{\bar{\beta}} + a_\rho(\theta) \frac{\Delta\rho}{\bar{\rho}} \quad (5.1)$$

dove

$$+a_\alpha(\theta) = \frac{1}{2\cos^2(\theta)} \quad (5.2)$$

$$+a_\beta(\theta) = -4\frac{\bar{\beta}^2}{\bar{\alpha}^2}\sin^2\theta \quad (5.3)$$

$$+a_\rho(\theta) = \frac{1}{2} - \frac{1}{2\cos^2(\theta)} + 2\frac{\bar{\beta}^2}{\bar{\alpha}^2}\sin^2(\theta) \quad (5.4)$$

$\alpha$  e  $\beta$  rappresentano  $I_p$  e  $I_s$ . Questa volta si osserva come la map sia molto vicina ai valori di  $I_p$  successivi a 1120 ms. Le realizzazioni nello spazio di  $p(m|d)$  in  $I_p$  ricadono tutte molto vicine alla soluzione map, indice dell'affidabilità del risultato. In  $I_s$  la map ha approssimato sufficientemente bene soltanto la prima parte fino a 1060 ms, dopodiché non riesce a seguire l'andamento dei successivi campioni.

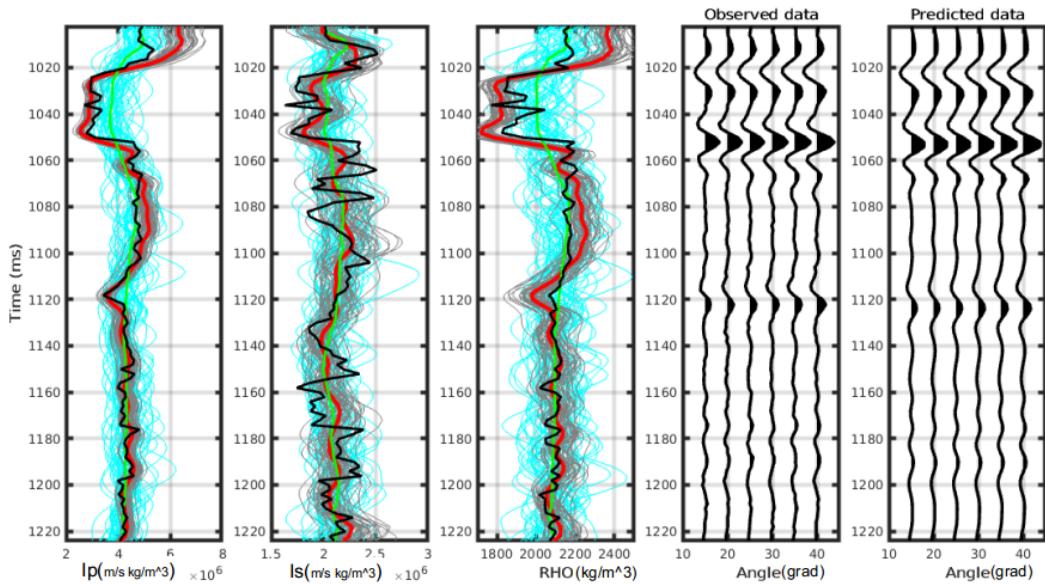


Figura 5.13: Risultati inversione Bayesiana AVO parametrizzata nelle impedenze sul sismogramma ricavato dal pozzo blind. La curva nera indica i valori veri, quella rossa la Maximum A Posteriori(MAP) e quella verde il modello low-frequency. Inoltre le curve grigie sono altre realizzazioni della distribuzione a posteriori  $p(m|d)$ , mentre quelle azzurre sono le realizzazioni del modello a priori  $p(m)$ .

In figura 5.14 è mostrata la Confusion Matrix relativa alla predizione sulla map. La classificazione con il modello di KNN questa volta ha restituito una maggiore affidabilità per le Brine Sand: Ne sono state riconosciute il 38% rispetto al 12% con la parametrizzazione

#### 5.4. INVERSIONE E CLASSIFICAZIONE DATI SISMICI SINTETICI RICAVATI DAL POZZO BLIND93

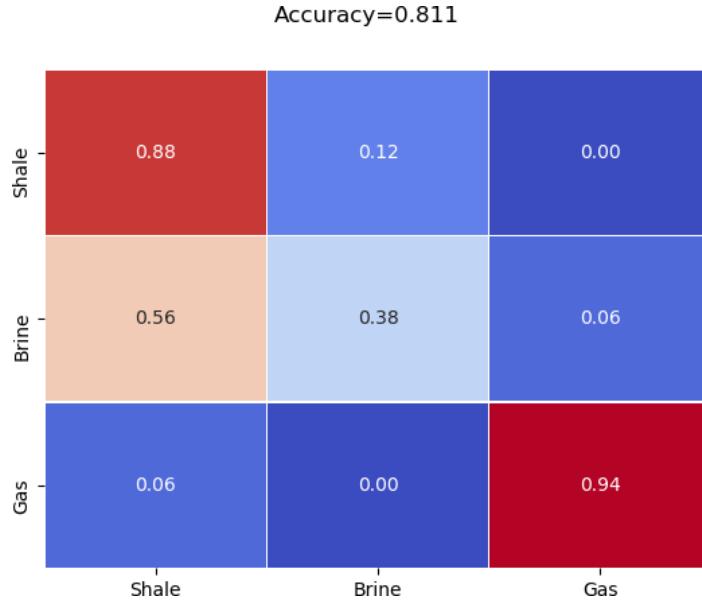


Figura 5.14: Confusion Matrix relativa alle predizioni sui parametri elasticici ottenuti con la map dell'inversione AVO parametrizzata nelle impedenze acustiche.

nelle velocità. Questo miglioramento è attribuibile a una migliore ricostruzione dei valori di  $I_p$  e  $\rho$  nei tempi superiori a 1120 ms, a sua volta dipendente dal tipo di parametrizzazione o dal miglior modello a priori. È aumentato anche il numero di campioni riconosciuti sul totale. Dalla classificazione sono stati esclusi i primi 7 campioni.

In figura 5.15 è mostrato il confronto tra le classi predette a partire dalla soluzione map e da 3 differenti realizzazioni estratte da  $p(m|d)$  e le classi osservate. Anche in questo caso tutte le realizzazioni sono concordi nella definizione del livello saturo a gas tra 1020 e 1050 ms. In generale però la maggior variabilità si ha nella classificazione negli strati di Brine Sand.

Sia nella parametrizzazione nelle velocità e in quelle nelle impedenze sismiche, la classe più sensibile ad effetti legati alla propagazione dell'errore dallo spazio dei dati a quello dei parametri è la Brine Sand a causa dell'overlap tra le proprietà elastiche di questa lithofacies e le Shale.

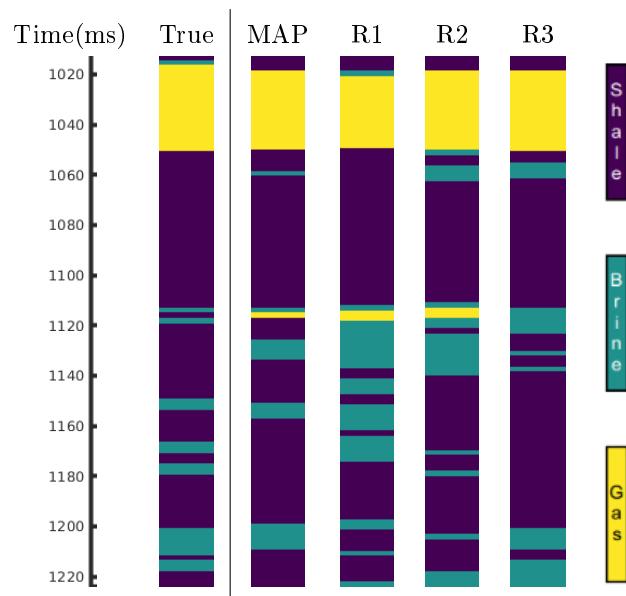


Figura 5.15: Confronto tra fluid-lithofacies vere, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(map), che su tre realizzazioni della distribuzione a posteriori  $p(m|d)$  ottenute con un inversione AVO parametrizzata nelle impedenze acustiche.

## Capitolo 6

# Classificazione con Training Set alla scala sismica

In questo capitolo si è ripetuta la Cross Validation seguita dalle classificazioni sui dati del pozzo blind derivati dall'inversione Bayesiana AVO. I dati utilizzati questa volta sono derivati dal filtraggio e dal ricampionamento dei dati dei 4 pozzi. Si vuole indagare l'effetto della perdita di informazione ad alta frequenza nel Training Set, quando i target della classificazione sono parametri elastici alla scala sismica. Le operazioni di filtraggio e ricampionamento si basano sugli stessi valori già adottati in precedenza per portare il blind log a scala sismica: ricampionamento alla massima frequenza sulla scala dei tempi, filtraggio passa basso con frequenza di taglio a 250 Hz e ricampionamento su asse dei tempi sismico a 2ms. Il numero di campioni dopo questo processo è sceso considerevolmente da 5584 a 381, con 324 campioni di Shale, 43 di Brine Sand e 14 di Gas Sand. Sarà interessante verificare gli effetti del calo del numero dei campioni sia sulla Cross Validation che sulle classificazioni successive. In figura 6.1 sono illustrati i campioni nel piano  $V_p/V_s$  e  $I_p$  dopo la conversione a scala sismica. In linea generale sono valide le stesse considerazioni fatte per i campioni alla frequenza dei log con la differenza che ci sono meno campioni a rappresentare le 3 classi.

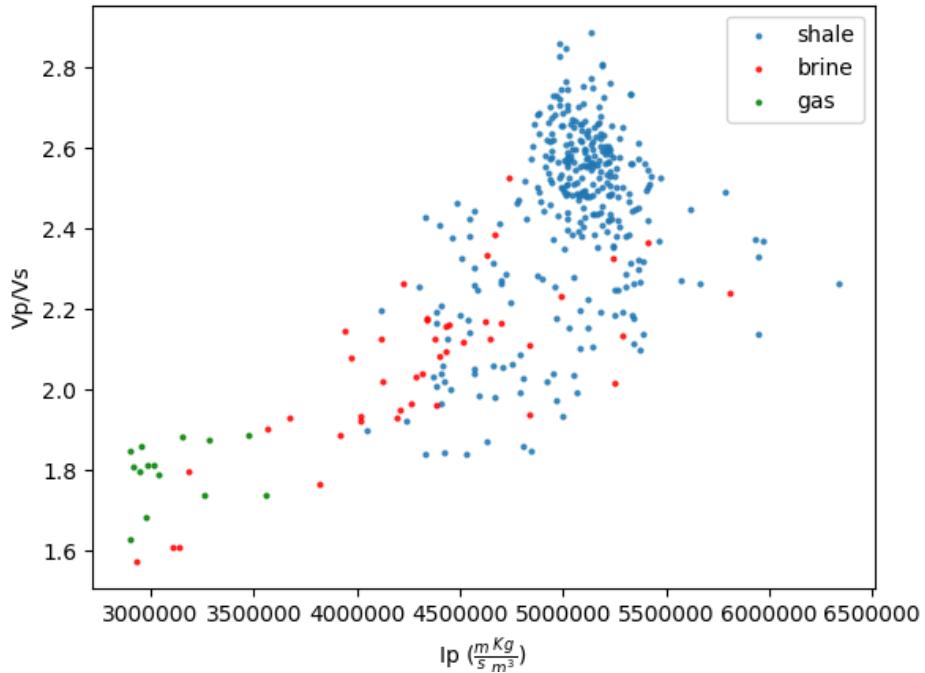


Figura 6.1: Visualizzazione in un piano  $I_p$   $V_p/V_s$  dei campioni riguardanti tutti i pozzi, escluso il blind, ricampionati alla massima frequenza di campionamento, filtrati con un filtro passa basso a 250Hz e ricampionati a 2ms.

Anche dal confronto con le funzioni di densità del dato a scala di log in figura 6.2 non si osservano grandi differenze tra le due distribuzioni, se non un accorciamento delle code causato dalla perdita dei valori estremi ad opera del filtraggio.

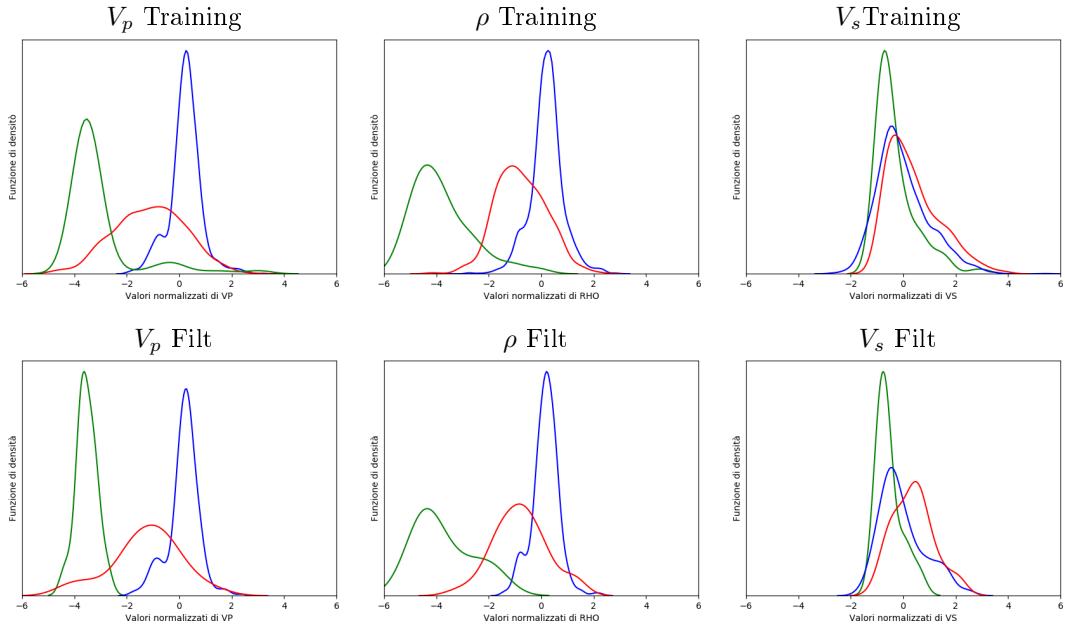


Figura 6.2: Funzioni di densità per i valori di  $V_p$ ,  $\rho$  e  $V_s$  normalizzati a partire da deviazione standard e media calcolati sul Training Set. In questo plot sono confrontate le funzioni di densità dei dati alla scala dei log, in alto, e quelli dei dati alla scala sismica dopo il ricampionamento. Le linee blu, rosse e verdi sono indicate rispettivamente Shale, Brine e Gas Sand.

## 6.1 Scelta del modello

La griglia degli iper-parametri per il Grid-Search è la stessa usata in tabella 5.2 per la prova a frequenza di log.

In tabella 6.1 sono illustrati i valori degli iper-parametri dei migliori modelli ottenuti sul Training Set alla scala sismica. Confrontando i valori degli iper-parametri attuali con quelli ottenuti dal training alla scala dei log, mostrati in tabella 5.3 nel capitolo 5, si osservano due fenomeni differenti per i modelli di ML lineari e non. I primi hanno performato meglio con coefficienti di regolarizzazione inversi di 3 ordini di grandezza superiori rispetto a quelli che hanno portato una migliore performance con il Training Set alla scala dei log. I modelli non lineari invece hanno tutti prodotto modelli più semplici: per NN il coefficiente di regolarizzazione  $\alpha$  è 3 ordini di grandezza più grande rispetto a quello che ha fornito migliori performance alla scala dei log, si veda anche il numero di vicini  $K$  per KNN pari a 5 contro 4 alla scala dei log e la profondità  $D$  e il numero di alberi  $N_{est}$  pari a 10 e a 120. Gli effetti degli iper-parametri in questione sono stati discussi nella sezione 3 del capitolo 4. Questa differenza nei valori dei migliori iper-parametri è direttamente collegabile a una diminuzione del numero dei campioni nel Training Set. In assenza di un numero sufficiente di dati i

modelli di ML non lienari non hanno sufficiente informazione per costruire linee di boundary complesse, mentre i modelli lineari funzionano bene con un numero di dati ridotto.

Tabella 6.1: Valori degli iper-parametri per i migliori modelli degli algoritmi

Modello di ML	P1	P2	P3
KNN	$K = 5$	$W = \text{Pesati}$	N1
LR	$C = 10^{-3}$		
SVC	$C = 10^{-4}$		
NN	$\alpha = 1$		
RF	$D = 10$	$N_{est} = 120$	

In figura 6.3 sono illustrate le confusion matrices ottenute dopo la Cross Validation. Tutte le Confusion Matrices riportano classificazioni di buona qualità dal punto di vista del numero di campioni predetti, con valori di Accuratezza sempre superiori al 90%. Nonostante i modelli lineari siano utilizzati siano contraddistinti da coefficienti di regolarizzazione inversi più alti rispetto alla classificazione con Training Set alla scala dei log, essi hanno prodotto linee di boundary che hanno classificato in ugual modo i campioni del Validation Set, con alti valori di Recall per i campioni di Shale e Gas Sand al prezzo di una scarsa performance per i campioni di Brine Sand. I modelli non lineari invece hanno saputo riconoscere meglio i campioni di Brine Sand. L'altra faccia della medaglia è un calo del Recall associato ai campioni di Gas Sand.

Nel box plot in figura 6.4 sono mostrati mediana, quartili e valori minimi e massimi per i valori di Accuratezza Bilanciata ottenuti da ogni cartella, mentre nella tabella 6.2 si riportano i valori numerici di media e deviazione standard ottenuti dalle performance sulle fold per ogni modello di ML. Con i campioni del Training Set filtrato a bassa frequenza RF ha restituito un Box molto allungato con una deviazione standard notevole. KNN e NN in una fold hanno ottenuto risultati prossimi alla predizione random. Sebbene le mediane dei modelli non lineari siano piazzate più in alto rispetto a quelle dei modelli lineari, questi ultimi si presentano con box compatti ancora una volta indice dell'affidabilità delle loro predizioni. Per quanto riguarda i valori di media dell' Accuratezza Bilanciata sui risultati sulle fold, sorprendentemente LR e SVC hanno in media performato meglio di RF e ottenuto valori di std sensibilmente più bassi. Ad ogni modo KNN si conferma ancora una volta il miglior modello di ML per il valore di media di Accuratezza Bilanciata.

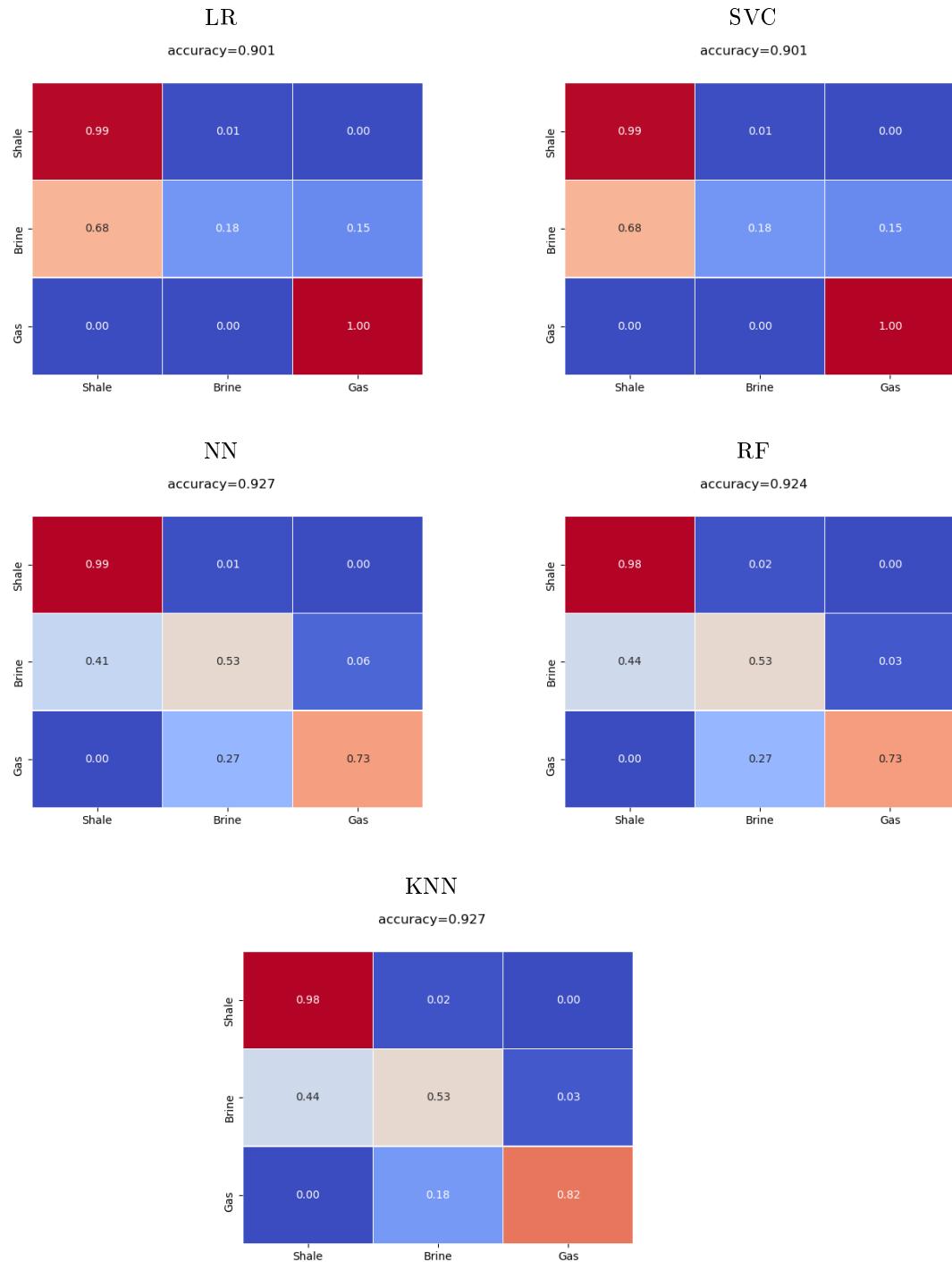


Figura 6.3: Confusion Matrices ricavate dalla somma delle performance su ogni fold della Cross Validation. All'interno sono mostrati i valori normalizzati per il numero dei campioni per ogni classe. Sopra a ogni CM è riportata la percentuale di campioni correttamente classificati.

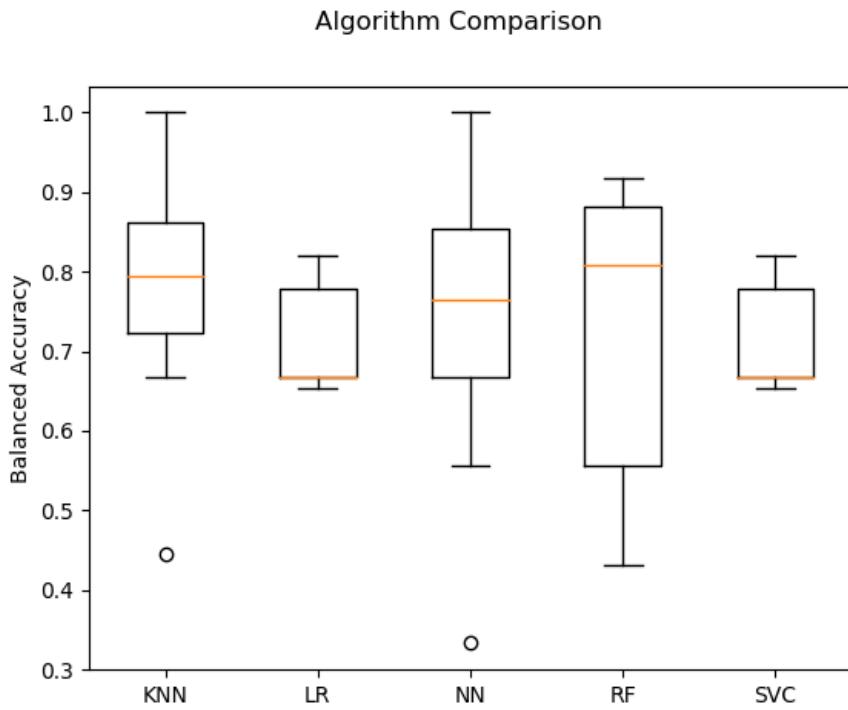


Figura 6.4: Box-plot dei valori di Accuratezza Bilanciata derivati delle performance sulle 10 cartelle della Cross Validation per le migliori combinazioni di parametri di ogni algoritmo. Il Box-plot riporta in arancione la media, il box rappresenta la zona dal primo al terzo quartile, mentre le linee arrivano fino ai valori minimi e massimi. I cerchi rappresentano gli outliers.

Tabella 6.2: Valori di media e deviazione standard di Accuratezza Bilanciata calcolata sulle performance nelle folds singole per ogni modello di ML tunato con la Cross Validation

Modello di ML	Media	STD
KNN	0.778166	0.139831
LR	0.717366	0.059954
SVC	0.717366	0.059954
NN	0.745338	0.176739
RF	0.713869	0.187845

Il modello di KNN generato dalla Cross Validation è stato riallentato su tutte e 10 le fold e utilizzato per una predizione sul Test Set. La Confusion Matrix su questa classificazione è mostrata in figura 6.5. Il modello riuscito a predire correttamente tutti i campioni di Gas Sand, anche se questo risultato possibilmente deriva dal basso numero di campioni contenuti nel Test Set. Ancora una volta un gran numero di campioni delle Brine Sands è stato assegnato alla classe delle Shale. Le Brine Sands, come indicato dal valore di F1-score in tabella 6.3 pari a 0.571 sono la classe associata a una qualità della predizione più scarsa, mentre le Gas hanno un F1-score pari a 1: esse sono state classificate tutte correttamente

e nessun'altra classe è stata confusa per Gas Sand. Ad ogni modo, dal momento che questi valori provengono da un set di campioni non molto grande, essi sono associati a una minore significatività.

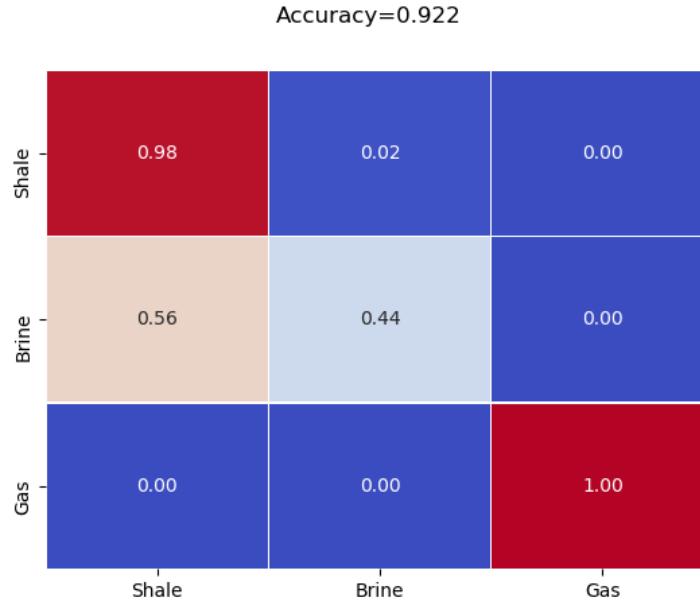


Figura 6.5: Confusion Matrix ricavata dal Test Set, i cui valori sono normalizzati per il numero di campioni in ogni classe.

Tabella 6.3: Metriche per le singole classi di Precision, Recall e F1-score

	Precision	Recall	F1-score
Shale	0.92754	0.98462	0.95522
Brine	0.80000	0.44444	0.57143
Gas	1.00000	1.00000	1.00000

Dai risultati ottenuti si può quindi affermare che il training dei modelli nei dati a frequenza sismica ha prodotto:

- Un abbassamento delle performance dei modelli non lineari dovuto probabilmente alla diminuzione di campioni disponibili
- Un modello di KNN simile a quello ricavato dal training con Training Set alla scala dei log.
- Un aumento dell'incertezza sui risultati della Cross Validation, ovvero deviazioni standard più alte.

## 6.2 Classificazione da risultato inversione AVO su dati sismici da pozzo blind

I risultati delle inversioni AVO sul sismogramma sintetico prodotto dai parametri elastici del pozzo blind precedentemente calcolati sono stati classificati con il modello di KNN ottenuto dal training a scala sismica. Sono state classificate sia le parametrizzazioni in  $V_p$  e  $V_s$  sia quelle in  $I_p$  e  $I_s$ . Nella classificazione sono stati coinvolte sia la map che le stesse realizzazioni dalla distribuzione a posteriori. I risultati di queste classificazioni sono illustrati nelle figure 6.6 e 6.7. Attraverso l'osservazione dei risultati non si deducono particolari differenze dalla classificazione fatta sul modello allenato sui dati alla scala di log: anche in questo caso le classificazioni sulle soluzioni a posteriori esaminate sono concordi nel definire il livello saturo a gas e si distinguono nelle classificazioni tra Brine Gas e Shale.

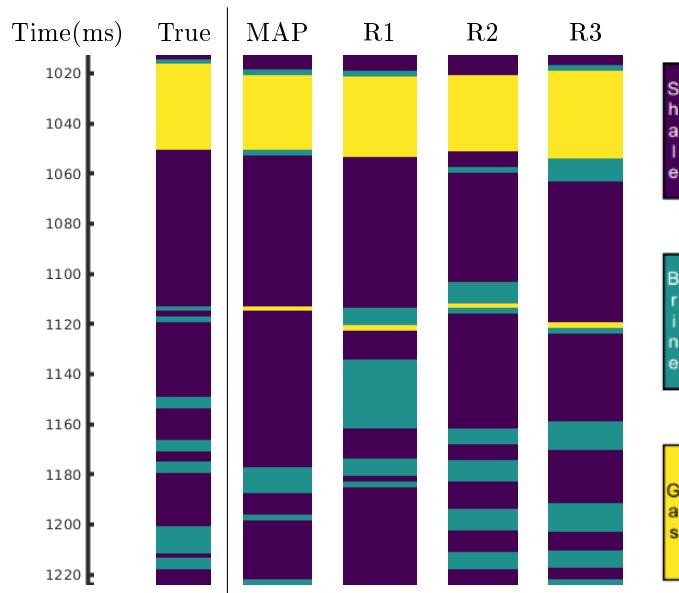


Figura 6.6: Confronto tra dati osservati, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(MAP), che su tre realizzazioni della distribuzione a posteriori  $p(m|d)$  ottenute con un inversione AVO parametrizzata nelle velocità.

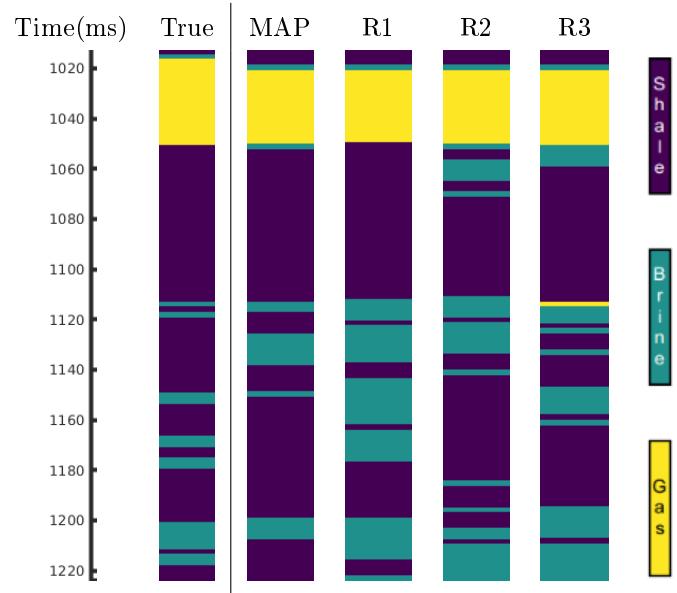


Figura 6.7: Confronto tra dati osservati, a sinistra, e le predizioni effettuate sia sulla maximum a posteriori(MAP), che su tre realizzazioni della distribuzione a posteriori  $p(m|d)$  ottenute con un inversione AVO parametrizzata nelle impedenze acustiche.

Tuttavia dall’analisi delle Confusion Matrices in figura 6.8 si osserva che il modello di KNN allenato alla scala dei log e testato sulle map risultanti dall’inversione del dato sintetico con le parametrizzazioni in  $I_p$  e in  $V_p$  ha ottenuto migliori risultati rispetto al modello di KNN allenato alla scala sismica, sia per quanto riguarda il numero di campioni riconosciuti sia nell’Accuratezza Bilanciata.

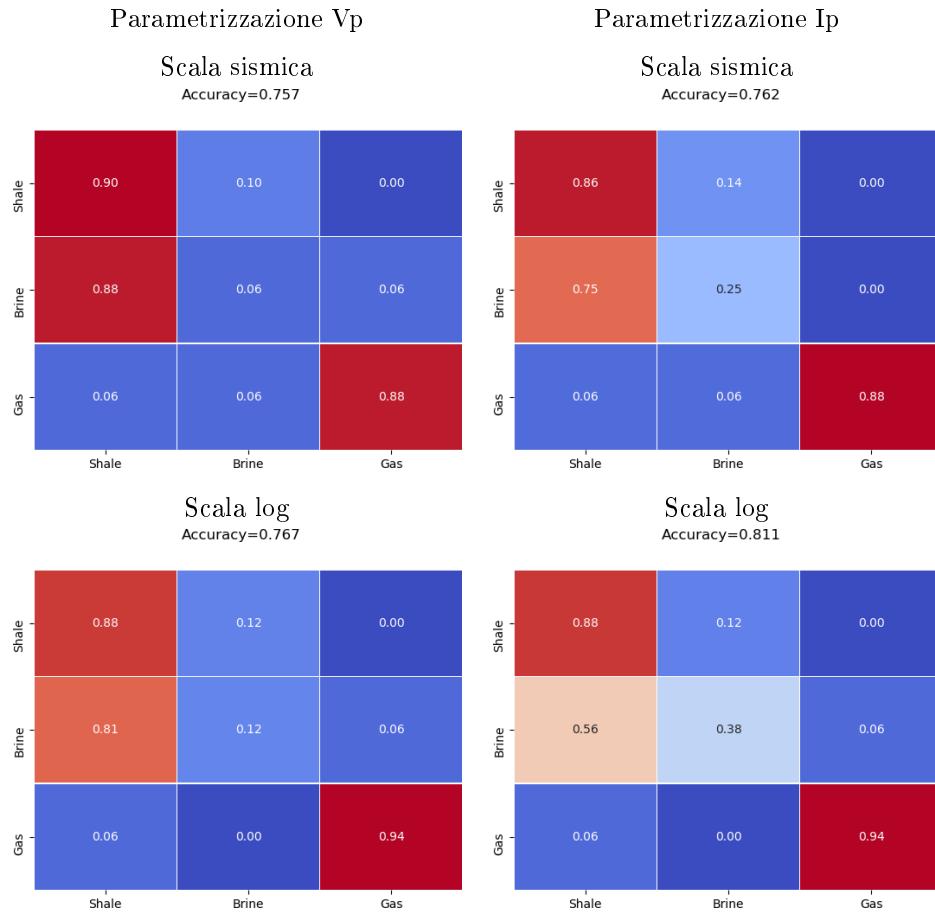


Figura 6.8: Confusion Matrices del modello di KNN allenato alla frequenza di log e a scala sismica, testato nella classificazione delle map di

In conclusione l’allenamento a scala sismica non ha prodotto differenze significative rispetto all’allenamento a scala dei log se non una lieve diminuzione delle performance sia nel Test set che nella classificazione sulle soluzioni map ottenuti dall’inversione AVO del sismogramma sintetico prodotto a partire dal pozzo blind nelle differenti parametrizzazioni. Per questo motivo si è deciso di allenare il modello per la classificazione sul dato reale utilizzando i dati alla scala dei log.

# Capitolo 7

## Classificazione del dato reale

In questa parte del lavoro di tesi tutti i dati di pozzo dei 5 pozzi alla scala log sono stati utilizzati per generare un set di esempi utile ad allenare il miglior modello di ML per predire le litho-fluidfacies appartenenti a una sezione sismica estratta da un volume 3D. L'obiettivo è quello di delineare lateralmente l'estensione del reservoir a Gas. Dal momento che il pozzo blind come già visto ha una distribuzione di densità dei parametri elastici differente rispetto a quella del Training Set, la sua aggiunta nel Training Set arricchisce il dataset rendendo il dato su cui si allenano gli algoritmi più rappresentativo delle vere proprietà statistiche dei parametri elastici.

### 7.1 Scelta del modello

Il dato composto dai parametri elastici provenienti dai 5 log di pozzo è stato diviso in Training Set(80%) e in Test Set(20%) attuando la procedura di stratification. Sul Training Set è stata eseguita la Cross Validation con 10 folds per tutti i modelli di ML. Si sceglierà in questo modo il modello di ML per la successiva classificazione sul Test Set e sui parametri elastici stimati dal dato reale. Al termine della Cross Validation si sono stimati i migliori iper-parametri per ogni algoritmo. La griglia degli iper-parametri scelta per il tuning è la stessa in tabella 5.2. In tabella 7.1 sono riportati i valori dei migliori iper-parametri per ogni modello di ML allenato. L'aggiunta del pozzo blind alla classificazione non ha causato una forte variazione nei migliori iper-parametri dei modelli di ML.

Tabella 7.1: Valori degli iper-parametri per i migliori modelli degli algoritmi

Modello di ML	P1	P2	P3
KNN	$K = 3$	$W = \text{Pesati}$	N1
LR	$C = 10^{-5}$		
SVC	$C = 10^{-6}$		
NN	$\alpha = 10^{-2}$		
RF	$D = 20$	$N_{est} = 250$	

In figura 7.1 sono mostrate le Confusion Matrices dei modelli allenati sulla Cross Validation, dati dalla somma dei punteggi su ogni cartella. I modelli lineari, che come già in precedenza osservato convergono ad un unico classificatore, non hanno mai predetto la classe delle Brine Sands. In condizioni di forte incertezza causate dall'aggiunta dei dati provenienti dal pozzo blind, questi classificatori hanno avuto migliori performance ignorando la classe delle Brine Sands. I modelli non lineari invece hanno saputo classificare con buoni risultati i dati delle contenuti nelle fold. Anche in questo caso essi riconoscono una buona percentuale di campioni di Brine Sands nonostante l'overlap presente tra le classi.

Osservando i box plot dei valori di Accuratezza Bilanciata ottenuti dalle folds della Cross Validation in figura 7.2 si osserva che i modelli di ML lineari hanno avuto un calo di Accuratezza Bilanciata, rispetto al training effettuato su 4 log di pozzo, di circa il 4%. Il forte abbassamento di deviazione standard osservato per i modelli lineari è con molta probabilità attribuibile al fatto che le classificazioni sulle 10 fold effettuate da questi modelli di ML, non predicendo mai la classe delle Brine Sands, variano soltanto nella classificazione di Shale e Gas Sand. I modelli non lineari hanno mediamente predetto meglio le classi e l'aggiunta del pozzo blind alla classificazione non ha abbassato le performance sulle fold.

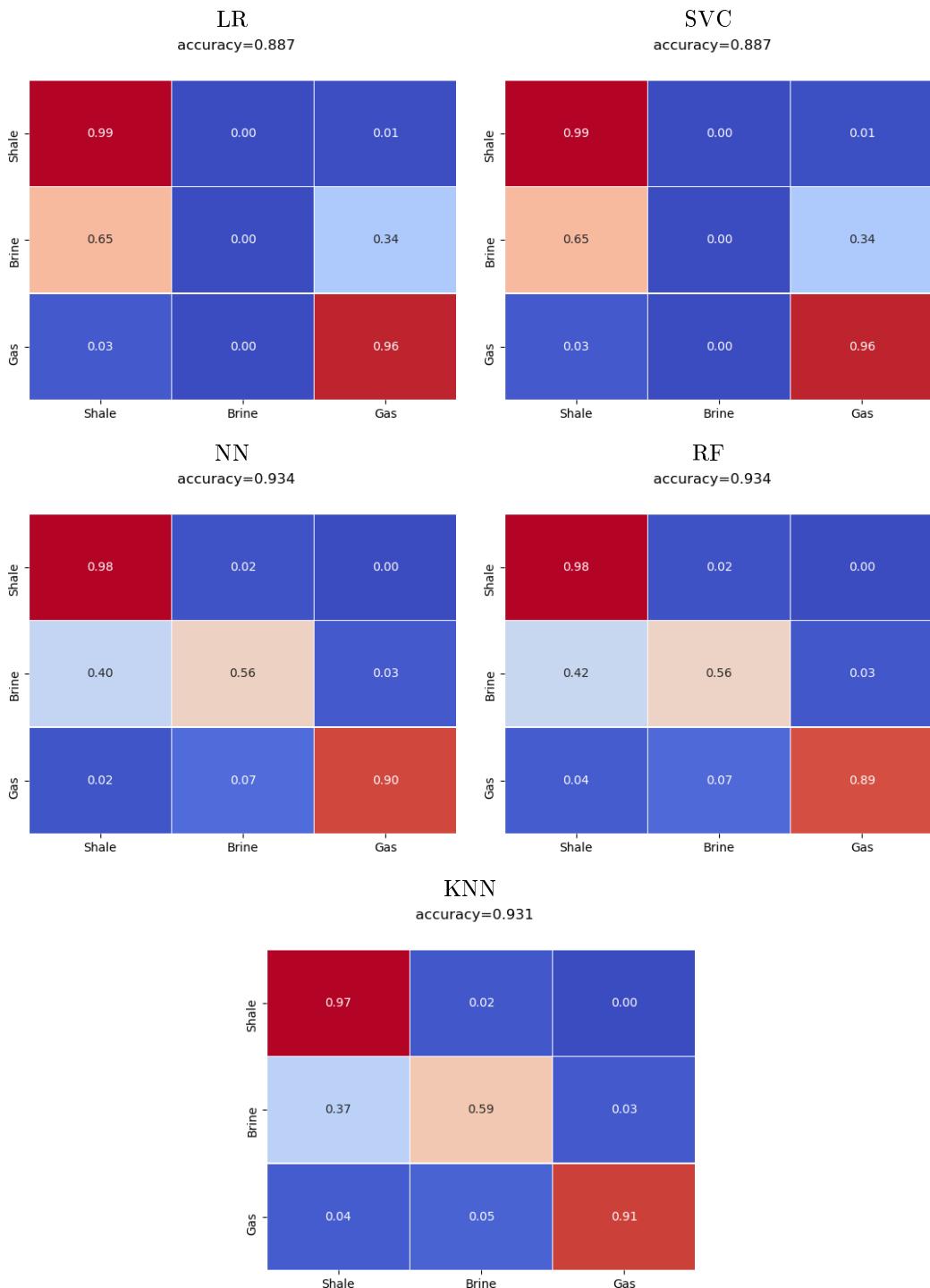


Figura 7.1: Confusion Matrices ricavate dalla somma delle performance su ogni fold della Cross Validation. All'interno sono mostrati i valori normalizzati per il numero dei campioni per ogni classe. Sopra a ogni CM è riportata la percentuale di campioni correttamente classificati.

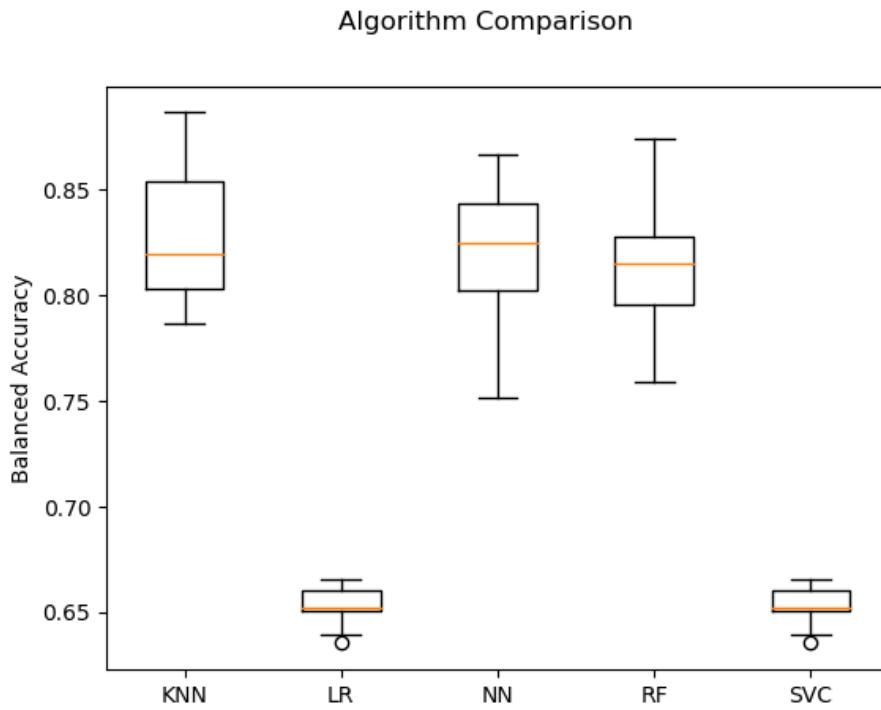


Figura 7.2: Box-plot dei valori di Accuratezza Bilanciata derivati delle performance sulle 10 cartelle della Cross Validation per le migliori combinazioni di parametri di ogni algoritmo. Il Box-plot riporta in arancione la mediana, il box rappresenta la zona dal primo al terzo quartile, mentre le linee arrivano fino ai valori minimi e massimi. I cerchi rappresentano gli outliers.

In tabella 7.2, dai valori di media dell'Accuratezza Bilanciata su ogni fold, KNN è stimato ancora una volta come miglior modello di ML. I valori di Accuratezza Bilanciata per KNN, NN e RF indicano predizioni bilanciate nello spazio delle tre classi, mentre i modelli lineari non hanno avuto performance comparabili.

Tabella 7.2: Valori di media e deviazione standard di Accuratezza Bilanciata calcolata sulle performance nelle folds singole per ogni modello di ML tunato con la Cross Validation

Modello di ML	Media	STD
KNN	0.830792	0.052965
LR	0.653587	0.041612
SVC	0.653587	0.041612
NN	0.820303	0.048123
RF	0.816565	0.041612

In figura 7.3 è illustrata la Confusion Matrix ricavata dalla classificazione del modello di KNN sul dato di Test e sono riportate le relative metriche in tabella 7.3. I valori di questa

Confusion Matrix sono molto simili a quelli ricavati dal Test Set effettuato con 4 log. Come nelle classificazioni sui precedenti Test Set, il modello ha archiviato discrete performance nella classificazione di dati provenienti dalla stessa distribuzione. L'aggiunta del pozzo utilizzato per le prove blind ha comunque causato un lieve peggioramento delle performance per le classi di Gas e Shale, come osservabile dai rispettivi valori di F1-score.

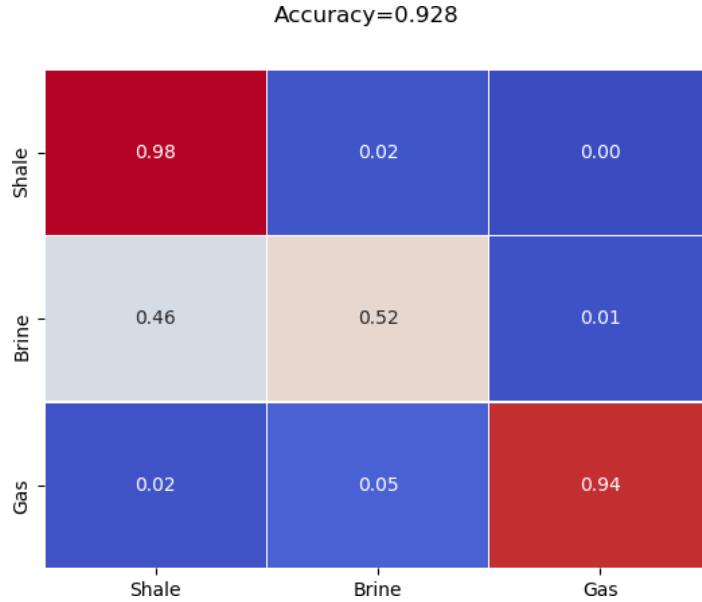


Figura 7.3: Confusion Matrix ricavata dalla classificazione sul Test Set, i cui valori sono normalizzati per il numero di campioni in ogni classe.

Tabella 7.3: Metriche per le singole classi di Precision, Recall e F1-score

	Precision	Recall	F1-score
Shale	0.94435	0.97746	0.96062
Brine	0.71963	0.52381	0.60630
Gas	0.96825	0.93846	0.95312

Da questo risultato ci si aspetta che per dati provenienti da una distribuzione di probabilità quantomeno simile a quella utilizzata per allenare i dati di partenza si identificheranno un'alta percentuale dei campioni di Gas e Shale.

## 7.2 Stima dei parametri elastici

In figura 7.4 si osserva una sezione stack campionata a 4ms dell'inline da cui si ricaveranno i parametri elastici da usare per la classificazione con il modello di KNN ricavato precedentemente. Il dato sismico in questione proviene da un'acquisizione terrestre che investiga un reservoir clastico saturato a gas e localizzato all'interno di depositi torbiditici. Il reservoir è localizzato nell'intervallo tra 0.92 ms 1.025 ms. Il dato pre-stack utilizzato nell'inversione contiene angoli

di incidenza di 15, 20, 25, 30, 35 e 40 gradi. Lungo questa sezione passano 2 pozzi con informazioni che saranno utilizzate anche per validare il risultato della classificazione

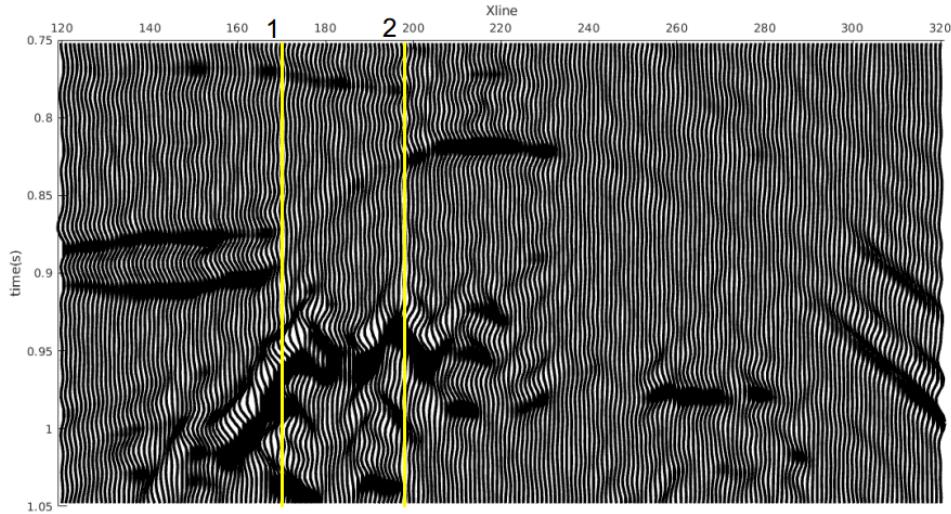


Figura 7.4: Stack lungo la in-line invertita. Le due linee gialle rappresentano la posizione di due pozzi localizzati lungo la sezione in-line.

Per stimare le proprietà elastiche dal dato reale è stato necessario stimare l'ondina sorgente da inserire nel forward modeling dell'inversione. Le precedenti fasi di processing hanno riportato tale ondina a fase zero.

Esistono una serie di metodi per stimare l'ondina sorgente, in questo lavoro di tesi sono state applicate la stima dell'ondina ai minimi quadrati e il metodo statistico.

Ogni metodo ha le sue particolari caratteristiche: il metodo ai minimi quadrati è utile quando è presente una traccia di riflettività calcolabile dai dati di pozzo. L'autocorrelazione è un metodo efficace per calcolare l'ampiezza dell'ondina, ma non da alcuna informazione sulla fase. In questo caso però la fase assunta dell'ondina sorgente è zero e quindi l'autocorrelazione della traccia dovrebbe restituire la corretta stima dell'ondina. Di seguito è mostrato il metodo che ha consentito di stimare l'ondina con una buona affidabilità.

### 7.2.1 Stima dell'ondina sorgente con il metodo statistico

Il metodo statistico consente di calcolare l'ondina sorgente del sismogramma attraverso la sola traccia sismica, perdendo l'informazione sullo spettro di fase. Un approfondimento teorico di questo metodo è presente in appendice A.2. L'ondina utilizzata per il forward modeling sismico in questo caso è a fase zero, quindi una stima coerente può essere effettuata attraverso l'autocorrelazione.

La stima con il metodo statistico è stata effettuata con le tracce ricavate dalla media dei CMP intorno al punto in cui passano i pozzi.

Per prima cosa è stata calcolata l'autocorrelazione, scelta in una finestra di 82 campioni, di tutte le tracce al variare dell'angolo, ed è stato calcolato il relativo spettro di ampiezza. Per rendere gli spettri più smooth alla funzione di autocorrelazione è stata applicata una finestra di Bartlett con lo stesso numero di campioni dell'autocorrelazione. In figura 7.5 sono mostrate le funzioni di autocorrelazione di una traccia sismica a 15 gradi e i relativi spettri di ampiezza prima e dopo l'applicazione della finestra di Bartlett. Si nota che l'utilizzo della finestra ha sortito l'effetto desiderato, abbassando i valori vicini alle code della funzione e rendendo lo spettro di ampiezza più uniforme.

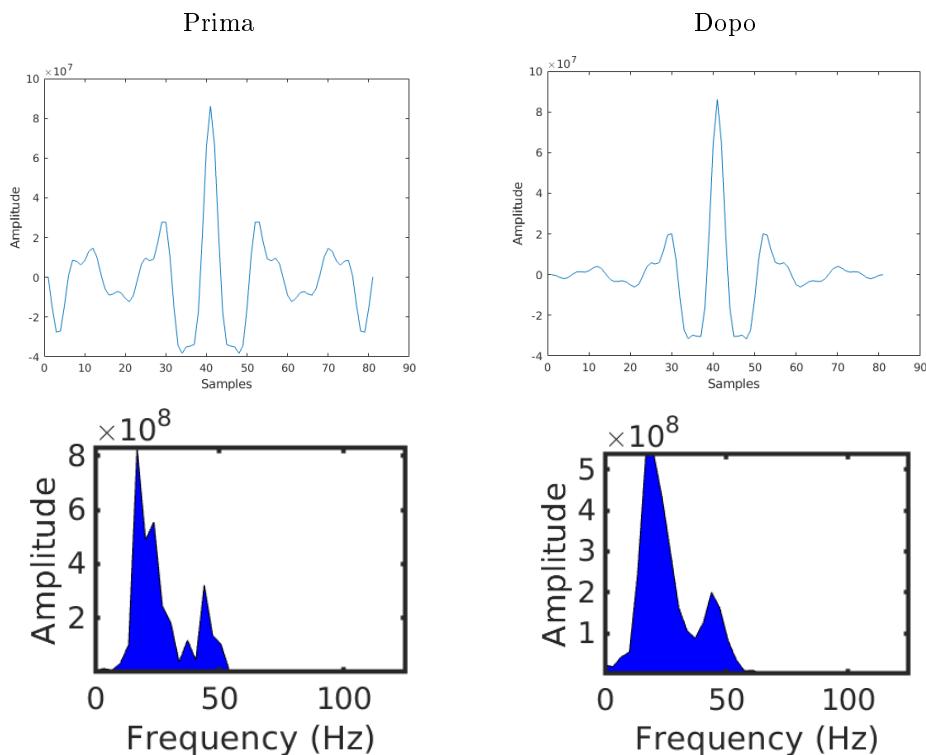


Figura 7.5: In alto a sinistra l'autocorrelazione di una traccia sismica presa a un'angolo di 15 gradi, in alto a sinistra la autocorrelazione dopo l'applicazione di una finestra di Bartlett. In basso gli spettri di ampiezza prima e dopo dell'applicazione della finestra.

Successivamente è stata calcolata la radice quadrata dello spettro di ampiezza della funzione di autocorrelazione dopo l'applicazione della finestra per ogni traccia e attraverso l'uso di una trasformata inversa si è ottenuta una prima stima dell'ondina sorgente, visualizzabile a sinistra in figura 7.6. Per eliminare i contributi rumorosi è stato applicato un filtro passa banda con frequenze di taglio a 15/45 Hz. Per non introdurre un effetto di Gibbs

è stato effettuato uno zero padding sul filtro nel dominio dei tempi per generare delle rampe smooth nel dominio delle frequenze. Come ultimo step per aumentare la qualità dell'ondina e mandare a zero le code dell'ondina è stata applicata una finestra Gaussiana con deviazione standard pari a 17 nell'asse dei campioni. Questi step hanno prodotto il risultato illustrato in figura 7.6 a destra. L'ondina finale è stata ritagliata in un range di 75 campioni.

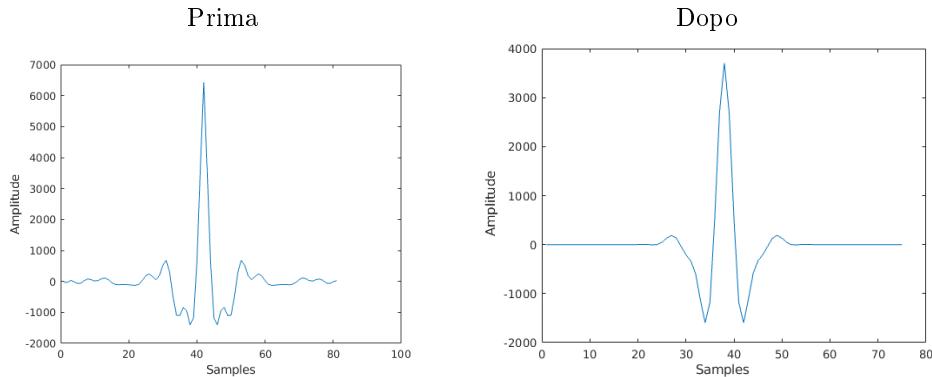


Figura 7.6: A sinistra ondina ottenuta dalla trasformata inversa della radice quadrata dello spettro di ampiezza della traccia sismica presa a 15 gradi. A destra la stessa ondina dopo l'applicazione di un filtro passa banda con frequenze di taglio pari a 15/45Hz e di una finestra Gaussiana con deviazione standard uguale a 17 nell'asse dei campioni.

Questo processo è stato eseguito su tutti gli angoli. Il risultato della sitma è mostrato in figura 7.7. Si nota che l'ampiezza delle ondine è dipendente dall'angolo e in particolare, tolta l'ondina a 20 gradi esse aumentano in ampiezza con l'aumentare dell'offset.

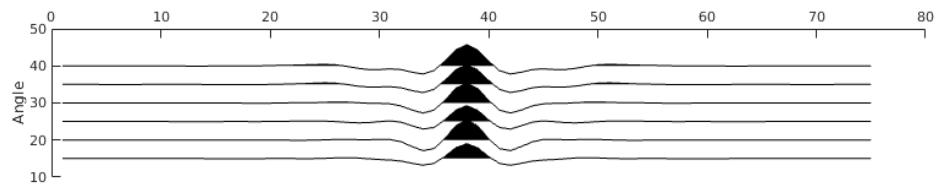


Figura 7.7: Illustrazione delle ondine stimate a seconda dell'angolo di incidenza del raggio sismico.

### 7.2.2 Inversione AVO

Una volta ottenuta la stima dell'ondina al variare degli angoli di incidenza essa è stata utilizzata nell'inversione Bayesiana AVO per produrre una stima dei parametri elastici del dato reale. Come modello a bassa frequenza a differenza delle precedenti inversioni si è

utilizzato un valore unico pari alla media dei parametri elastici ricavato dai log di pozzo. La soluzione map di questa inversione, eseguita con una parametrizzazione nelle impedenze acustiche è mostrata in figura 7.8. Si osservano bassi valori di impedenza acustica  $I_p$  e densità  $\rho$  in prossimità del reservoir e delle principali riflessioni. La stima map sarà utilizzata per una classificazione effettuata con il modello di KNN generato a partire dai dati di training. Non avendo a disposizione una corretta classificazione del dato reale non sarà possibile verificare la correttezza della classificazione, tuttavia il livello saturo a gas è presente nella zona tra 0.95 e 1 ms e tra le xline 150 e 200 e ci si aspetta che questo livello venga riconosciuto dal modello di ML.

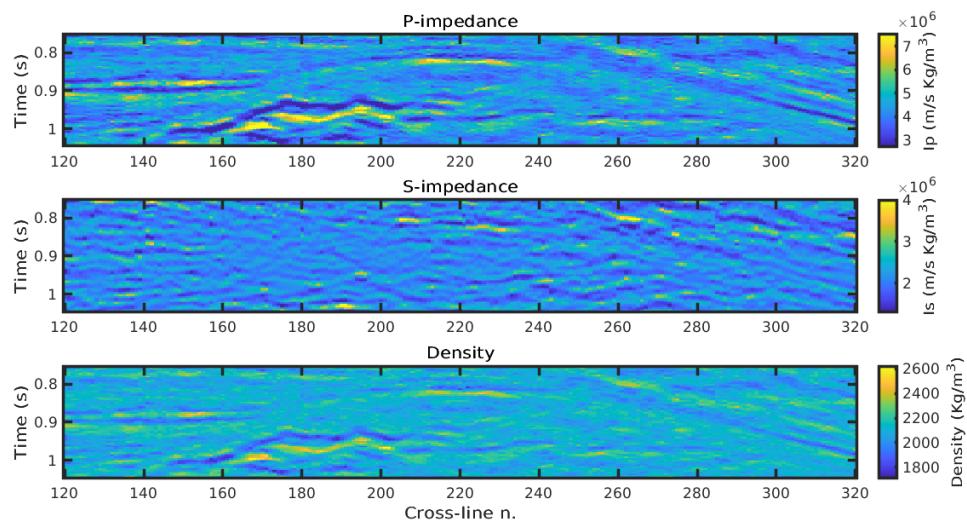


Figura 7.8: Visualizzazione grafica dei valori dei parametri elastici ricavati dalla map dell'inversione AVO sul dato reale.

In figura 7.9 le funzioni di densità dei parametri elastici ottenute dai campioni ricavati con l'inversione AVO sono state messe a confronto con quelle ricavate dal Training Set per verificare quanto i parametri statistici nei due set coincidano. Si osserva che le distribuzioni hanno un valore di media molto simile, ma differiscono nella forma. In  $I_p$  e  $\rho$  nel dato di training sono presenti curvature nella parte sinistra della distribuzione che identificano le distribuzioni delle Brine Sand e delle Gas. Vi sono quindi differenze sicuramente non trascurabili tra i parametri elastici di training e quelli del dato reale.

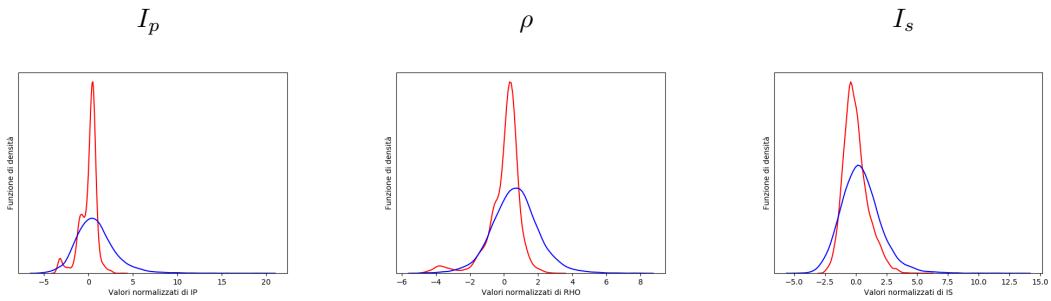


Figura 7.9: Funzioni di densità dei parametri elastici ricavati dalla map dell'inversione AVO sul dato reale (In blu) confrontate con le funzioni di densità dei parametri elastici del dato di trainning (In rosso).

### 7.3 Risultati

I parametri elastici calcolati con l'inversione AVO sono stati classificati con il miglior modello di KNN calcolato sui dati derivati dai 5 log di pozzo. In figura 7.10 Si osserva, in primo luogo, la visibile sovrappredizione sui livelli saturi a gas; il modello di ML è comunque riuscito in modo soddisfacente a delinare il reservoir, indicato da una freccia rossa, e a riprodurre le principali geometrie che caratterizzano il dato sismico. Questo risultato è stato ottenuto nonostante due problemi: il basso rapporto S/N del dato sismico di partenza e le differenze nelle proprietà statistiche con il dato su cui è stato effettuato il training. Inoltre si ricorda che negli algoritmi di ML utilizzati non esiste alcun vincolo laterale che correli spazialmente classificazioni su campioni adiacenti.

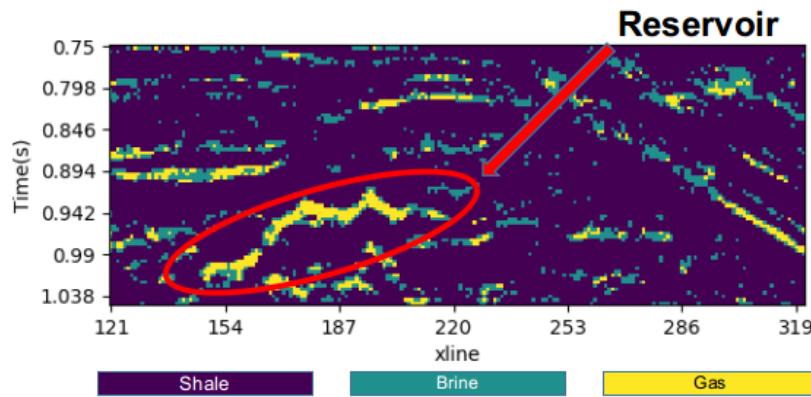


Figura 7.10: Immagine che illustra le classificazioni effettuate dal miglior modello di KNN sulla soluzione map dell'inversione AVO sul dato sismico del dato reale. Il reservoir è indicato da una freccia rossa.

Anche in questo caso le metriche ricavate dalla classificazione sul Test set non sono estendibili alle classificazioni sul dato reale: la Precision per i campioni di Gas Sand nel Test Set era pari al 95% e se il dato reale avesse ricalcato la distribuzione del dato di training non si sarebbe verificata questa sovrappredizione sulla classe delle Gas Sands. Questo naturalmente è riconducibile al fatto che i pozzi non rappresentano tutta l'area di studio, ma in modo specifico quella in prossimità del reservoir.

La figura 7.11 indica la posizione dei due log di pozzo passanti per il dato reale, attraverso la sovrapposizione delle le classi vere calcolate a partire dalle proprietà petrofisiche dei due log passanti per il dato reale ricampionati a 4 ms nelle xline di riferimento. Questi pozzi possono essere usati per validare il risultato ottenuto sul dato reale.

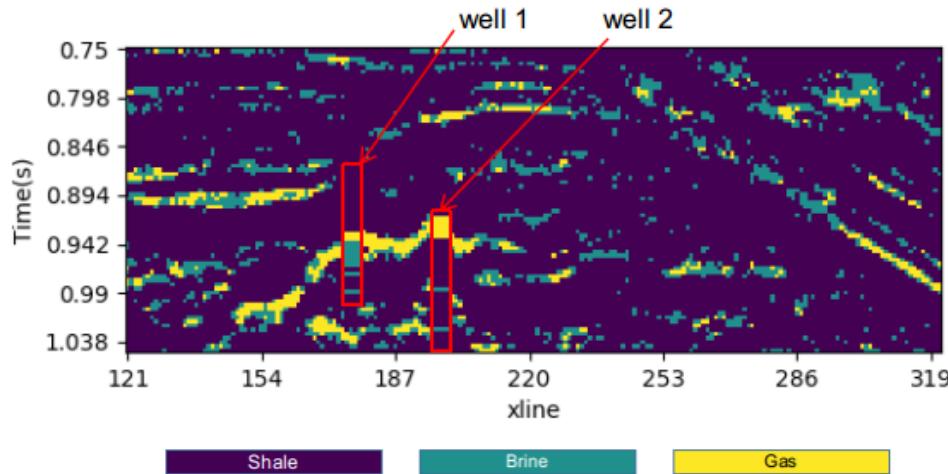


Figura 7.11: Immagine che illustra le classificazioni effettuate dal miglior modello di KNN sulla soluzione map dell'inversione AVO sul dato sismico del dato reale. Il reservoir è indicato da una freccia rossa.

Molto probabilmente le interfacce saturate a gas individuate nelle xline sono da associare a livelli saturi a brine. In figura 7.12 è illustrato il confronto tra le classi vere dei pozzi passanti per le Xline 177 e 198, ricampionati a 4ms, e alla classificazione sulla soluzione map dell'inversione AVO sul dato reale. Si osserva da entrambi i confronti la fortissima differenza nella definizione degli strati di sand saturi a brine. Tuttavia per quanto riguarda i livelli di Shale e di Gas la classificazione sembra essere robusta.

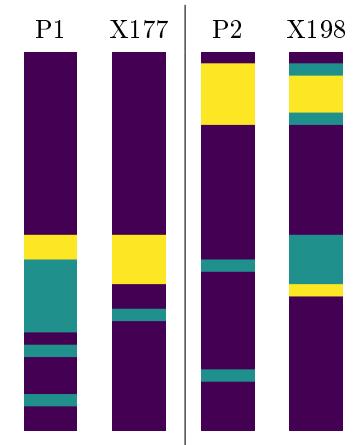


Figura 7.12: Confronto tra litho-lithofacies vere corrispondenti ai pozzi passanti per le Xline 177 e 198, confrontati con le rispettive litho-fluidfacies classificate dal modello di KNN dalla soluzione map dell'inversione AVO sul dato reale.

Si presentano questi risultati in via del tutto preliminare, con ampio margine di miglioramento perseguibile in studi futuri.

# Capitolo 8

## Conclusioni

In questo lavoro di tesi, si è affrontato un problema ricorrente nel campo della geofisica di esplorazione, ovvero la classificazione delle fluid-lithofacies. Questo problema è stato affrontato con l'utilizzo di metodi e tecniche dell'ambito del Machine Learning.

La presenza dei parametri petrofisici ( $\phi$ ,  $Sw$  e  $Sh$ ) nei log di pozzo ha reso possibile la classificazione delle litho-fluidfacies a seconda di valori di cut-off. Questa classificazione è stata associata ai valori dei parametri elastici per generare un dato di input per un problema di classificazione supervised.

Da un'analisi grafica dei campioni effettuata con i parametri  $I_p$  e  $V_p/V_S$ , si è dedotta l'esistenza di tre cluster differenti associati alle tre lithofacies. Il dato di partenza presenta una zona di overlap tra Shale e Brine Sand che coinvolge soprattutto quest'ultima classe, rivelatasi quella più difficile da riconoscere.

L'analisi delle funzioni di densità di probabilità calcolate con il kernel density, della feature importance ricavata da RF e dei coefficienti dei modelli lineari hanno rilevato che le due variabili più informative nella discriminazione delle litho-fluidfacies sono  $V_p$  e  $\rho$ , mentre  $V_s$  non si rivela in ugual modo determinante. Questo è stato attribuito da un lato all'insensibilità della velocità delle onde di taglio alla presenza di fluidi nello spazio poroso e dall'altro alle caratteristiche particolari di Shale e Sand che in questa area si trovano a una profondità che non permette una distinzione in base alla  $V_s$ .

Dall'analisi dei valori relativi dei coefficienti dei modelli lineari e dalle performance ottenute con l'allenamento su dati normalizzati e non, si è osservato che la normalizzazione gioca un ruolo fondamentale nelle performance degli algoritmi di learning.

La visualizzazione grafica delle linee di boundary al variare degli iper-parametri degli algoritmi, ha mostrato come questi iper-parametri abbiano la funzione di regolare la complessità delle funzioni che essi producono per la classificazione.

Nella seconda parte del lavoro di tesi i 5 modelli sono stati comparati sulle performance della K-fold Cross Validation con 10 folds effettuata sui parametri elastici del Training Set derivato da 4 log di pozzo. Dalle Confusion Matrices derivanti dalla Cross Validation si è osservato come gli algoritmi lineari abbiano difficoltà a classificare correttamente i campioni di Brine Sands ed in generale ricadono nei modelli con il più basso valore di Accuratezza Bilanciata, sebbene questi valori siano associati a una deviazione standard più bassa rispetto ai modelli non lineari. Inoltre SVC e LR hanno prodotto modelli con performance identiche dato l'alto valore del coefficiente di regolarizzazione che ha fatto ottenere agli algoritmi un valore di Accuratezza Bilanciata maggiore. I modelli non lineari hanno classificato meglio le Brine Sand e in generale ottenuto una classificazione di qualità migliore. Il miglior modello di ML è risultato KNN il quale, sebbene sia un algoritmo di concezione semplice, ha dimostrato di comportarsi in modo ottimale nell'interfaccia Brine Sand-Shale nello spazio dei parametri elastici. Nella classificazione sul Test set il modello di KNN ha ottenuto ottimi risultati su Brine Sand e Gas Sand, dimostrando performance notevoli su dati estratti dalla stessa distribuzione dei dati di training. La classificazione sul pozzo blind ha invece restituito performance non in linea con quanto ottenuto sul test set a causa della variabilità statistica nelle proprietà elastiche del pozzo blind. Benchè l'Accuratezza Bilanciata sul pozzo blind sia analoga a quella avuta sul test set, le predizioni sulle varie classi, come indicato dai valori di F1 score, sono state di qualità inferiore.

Le classificazioni, effettuate con il modello di KNN, sui parametri elastici delle soluzioni map e da alcune possibili realizzazioni a posteriori delle inversioni AVO, hanno riportato correttamente il livello saturo a gas, ma non hanno saputo definire con sufficiente precisione le interfacce di Brine Sand. Con l'AVO parametrizzata nelle impedenze si sono ottenuti risultati leggermente migliori: in questo caso i tre parametri sono massimamente ortogonali. Lo stesso procedimento eseguito sui dati filtrati e ricampionati a scala sismica ha mostrato come le performance degli algoritmi lineari si sono avvicinate a quelle degli algoritmi non lineari. Il numero di campioni ha giocato un ruolo determinante nelle performance degli algoritmi: nella Cross Validation con training set alla scala dei log gli algoritmi non lineari archiviano valori di Accuratezza Bilanciata maggiori rispetto alla Cross Validation con training set a scala sismica; viceversa i modelli lineari hanno performato meglio con un numero di campioni più ristretto. Questo suggerisce di porre maggiore attenzione ai modelli lineari quando il numero di campioni disponibili per il training non è elevato. Ad ogni modo il miglior algoritmo in questo caso è risultato sempre KNN.

KNN allenato con training set alla scala sismica impiegato nella classificazione della soluzione map e sulle altre realizzazioni delle distribuzioni a posteriori per i risultati delle inversioni AVO parametrizzate nelle velocità e nelle impedenze, ha restituito performance inferiori

rispetto al modello di KNN allenato sui dati alla frequenza di log. È stato quindi deciso di utilizzare i dati alla frequenza di log per creare il modello utile a classificare i parametri elasticci ricavati dai dati sismici reali.

La Cross Validation è stata eseguita per produrre il miglior modello da usare nella classificazione del dato reale. Il training set questa volta è composto da campioni provenienti da 5 log di pozzo. L'aggiunta del pozzo utilizzato come blind in precedenza con una differente distribuzione arricchirebbe il dataset rendendolo più rappresentativo della realtà. Dai risultati della Cross Validation il miglior modello è rimasto KNN.

Il fatto che questo algoritmo abbia archiviato performance migliori in tutte e tre le Cross Validation può essere spiegato con il fatto che KNN lavora in modo ottimale in poche dimensioni.

Inoltre tutte le volte che è stata effettuata la Cross Validation i 5 modelli di ML hanno restituito Confusion Matrices tra di loro simili, suggerendo una robustezza nei risultati ottenuti. L'unica variazione significativa nelle Confusion Matrices prodotte si è verificata tra modelli lineari e non lineari.

Le performance di KNN sul Test Set sono state di buona qualità ad eccezione per le classificazioni sulle Brine Sands.

Attraverso l'autocorrelazione di alcune tracce sismiche estratte dal dato reale e diverse fasi di processing dell'autocorrelazione, è stato possibile recuperare un ondina a fase zero per ogni angolo di incidenza. Questa ondina è stata in seguito utilizzata per stimare i parametri elasticci oggetto della classificazione con il modello di KNN. Il modello di KNN utilizzato per la classificazione sulla map ha saputo fornire dei risultati preliminari nei quali sono stati individuati correttamente i livelli saturi a gas, il top e il bottom del reservoir. La principale conclusione è che a ogni classificazione prodotta da un modello di ML vada associata la supervisione di un operatore con conoscenze geologiche/geofisiche in grado di interpretare correttamente i risultati finali.



## Capitolo 9

# Prospettive future

In questo lavoro di tesi solo alcuni degli algoritmi e dei metodi di ML sono stati trattati. Una prospettiva interessante potrebbe essere l'aggiunta di un kernel per SVC, tecnica che permette di espandere lo spazio delle features per i dati di input e generare un classificatore non lineare nello spazio delle features di partenza. Questo tipo di classificatore è ampiamente usato in ambito scientifico e industriale grazie alla sua robustezza: esso ha un problema di ottimizzazione convesso, ma riesce a riprodurre funzioni fortemente non lineari.

Un'ulteriore sviluppo di questo lavoro di tesi vede l'impiego di metodi di unsupervised learning, come per esempio metodi di clustering sul dato reale. Queste tecniche potrebbero fornire uno strumento di analisi che, attraverso la ricerca di pattern all'interno del dataset composto dai parametri elastici stimati dall'inversione AVO, potrebbe essere in grado di effettuare una prima classificazione delle facies da confrontare con la classificazione effettuata con il modello di ML. Infine potrebbero essere performate delle fasi di feature selection, ovvero potrebbero essere selezionate tra le variabili che descrivono un campione soltanto quelle che meglio permettono di separare le classi. A tal fine potrebbero essere calcolate dai parametri elastici di partenza ulteriori variabili tra le quali il modulo bulk  $K$ , il modulo di taglio  $\mu$  e il coefficiente di possion  $\nu$ . Tale operazione può passare per esempio attraverso metodi stepwise, i quali andrebbero performati all'interno della Cross Validation.

Un'altro possibile sviluppo per conoscere meglio l'incertezza associata alle classificazioni campione per campione, da un risultato di un'inversione AVO, potrebbe essere l'impiego di metodi Montecarlo per estrarre tante più possibili realizzazioni a posteriori dei parametri elastici, classificarli con un metodo di ML, e osservare quante volte un campione mantiene invariata la sua classificazione nelle varie simulazioni. Un'altra possibile soluzione per conoscere meglio l'incertezza associata alle classificazioni potrebbe essere quella di adottare metodi di ML con un'impronta fortemente probabilistica, come per esempio i Bayesian Neural Network. Infine, dati i limiti riscontrati dall'Accuratezza Bilanciata, la quale non sempre è in grado di

descrivere correttamente la qualità della predizione, un'altra metrica o una combinazione di metriche potrebbe essere utilizzata per definire le performances del modello.

# Appendice A

## A.1 Glossario

Per aiutare il lettore a comprendere termini comuni del Machine Learning sono fornite le definizioni di alcuni dei termini più usati. [23]

### Attributo (Parametro,Feature)

Una quantità che descrive un esempio. Un attributo ha un dominio definito dal tipo di attributo che denota i valori che possono essere presi dall'attributo. Una feature è una la specificazione di un attributo e del suo valore. Per esempio, se  $V_p$  è un attributo  $V_p=2000$  m/s è una feature per un campione. Alcuni autori usano feature come sinonimo di attributo.

### Algoritmo

Un algoritmo di Machine Learning è un set di ipotesi inteso prima del processo di training. Quando per esempio si parla di algoritmo di regressione lineare, ci si riferisce un set di funzioni che hanno caratteristiche simili.

### Set, dato

Un insieme di campioni indipendenti e appartenenti alla stessa distribuzione di probabilità. Il dato sismico invece è il dato osservato/predetto in termini di inversione Bayesiana AVO

### Classificatore

Un classificatore mappa esempi non etichettati in classi discrete.

### Dimensione, Spazio

Un attributo o una serie di attributi che insieme descrivono una proprietà. Per esempio una dimensione geografica potrebbe consistere in tre attributi: Posizione in x,y e z.

### Campione, Esempio

Un singolo oggetto su cui sarà usato un modello di ML, per esempio usato per fare una predizione. Nella maggior parte dei lavori di Machine Learning, gli esempi sono descritti da vettori di features.

Modello di ML/ di KNN/SVC/LR/NN/RF)

Un modello di ML può essere una rappresentazione matematica di un processo del mondo reale, che viene prodotto dalla scelta degli iper-parametri e dall'allenamento dell'algoritmo.

Modello

Il modello come termine unico descrive per l'inversione Bayesiana la matrice  $m$  composta dai valori dei parametri elastici.

Regressore

Un regressore mappa esempi non etichettate a un valore definito da un dominio di valori

Training

Il processo di determinazione dei pesi  $w$  ideali compresi in un modello.

## A.2 Metodo Statisico per la stima dell'ondina

Il metodo statistico stima l'ondina dall'autocorrelazione del sismogramma [24]. Calcolando l'autocorrelazione della traccia sismica  $\phi_s(t)$  si ottiene:

$$\phi_s(t) = [w(-t) \bullet r(-t) + n(-t)] \bullet [w(t) \bullet r(t) + n(t)] \quad (\text{A.1})$$

$$= w(-t) \bullet r(-t) \bullet w(t) \bullet r(t) + w(-t) \bullet r(-t) \bullet n(t) + n(-t) \bullet w(t) \bullet r(t) + n(-t)n(t) \quad (\text{A.2})$$

$$= \phi_w(t) \bullet \phi_r(t) + w(-t) \bullet \phi_{rn}(t) + w(t) \bullet \phi_{nr}(t) + \phi_n(t) \quad (\text{A.3})$$

dove  $\bullet$  indica la convoluzione,  $\phi_s(t)$ ,  $\phi_w(t)$ ,  $\phi_r(t)$  e  $\phi_n(t)$  sono rispettivamente l'autocorrelazione di  $s(t)$ ,  $w(t)$ ,  $r(t)$ .  $\phi_{rn}(t)$  e  $\phi_{nr}(t)$  sono le cross-correlazioni tra  $r(t)$  e  $n(t)$ . Se si assume che  $r(t)$  e  $n(t)$  siano due serie random di lunghezza infinita allora la loro autocorrelazione potrà essere scritta in termini di un delta di Dirac  $\delta(t)$ :

$$\phi_r(t) = P_r \delta(t) \quad (\text{A.4})$$

$$\phi_n(t) = P_n \delta(t) \quad (\text{A.5})$$

con  $P_r$  e  $P_n$  le rispettive potenze. la cross-correlazione tra  $n$  e  $r$  sarà uguale a 0:

$$\phi_{rn}(t) = \phi_{nr}(t) = 0 \quad (\text{A.6})$$

Sostituendo nell'equazione iniziale si ottiene:

$$\phi_s(t) = P_r \phi_w(t) + P_n \partial(t) \approx a \phi_w(t) \quad (\text{A.7})$$

dove  $a$  è una costante. Nel dominio delle frequenze si ottiene

$$|S(f)|^2 \approx a|W(f)|^2 \quad (\text{A.8})$$

dove  $S(f)$  e  $W(f)$  sono le trasformate di Fourier di  $s(t)$  e  $w(t)$ . Quindi lo spettro di ampiezza dell'ondina sismica può essere stimato dalla funzione di autocorrelazione della traccia sismica, perdendo però l'informazione sullo spettro di fase.

### A.3 Bootstrap

Il Bootstrap è un metodo di resampling che genera subset derivati dal training set di partenza estraendo da esso campioni random senza escluderli da un nuovo ripescaggio: quando un campione è selezionato per essere parte di un subset esso sarà ancora disponibile per un'ulteriore selezione. Come risultato di questo processo alcuni campioni saranno rappresentati nel dataset di bootstrap più volte, mentre altri, riferiti come campioni *out of bag*, sono lasciati fuori e possono essere utilizzati per validare le performance del modello di ML. Questo non si è verificato nell'allenamento di Random Forest in quanto le sue performance si sono valutate sulla Cross Validation e il bootstrap è stato utilizzato per generare diversi dataset su cui allenare i DT. In figura A.1 sono mostrati tre subset derivati dal bootstrap: 10 campioni sono presenti nel dataset originale ed il primo subset generato contiene due volte il campione 10, il campione 5 e il campione due, quindi tre campioni sono lasciati fuori e possono essere utilizzati per la validazione del modello.

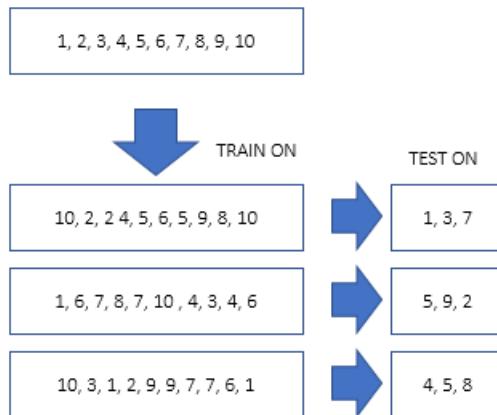


Figura A.1: Visualizzazione grafica del Bootstrap: da un set di dati iniziali vengono estratti 3 set con lo stesso numero di campioni, ma alcuni di essi presenti nel dataset originale si ripetono.

### A.4 Programma Model Evaluation

L'intera pipeline di lavoro per la scelta del modello si basa su un'unico programma in grado di prendere in input una matrice contenuta in un foglio di testo. Il programma è stato realizzato in python 3.6 e convertito in eseguibile con l'ausilio del pacchetto cx-Freeze (6.0b1) (<https://cx-freeze.readthedocs.io>) ed è compatibile con SO Windows, ed è in grado di allenare i 5 algoritmi utilizzati nel lavoro di tesi e restituire il migliore in un formato leggibile dalla libreria Python 'Joblib'. In figura A.2 è visualizzato il diagramma di flusso del programma. Per prima cosa sono caricati i parametri di input e il dataset. Il programma dopo il processo di Cross Validation restituisce le Confusion Matrices di ogni modello in 5 immagini in formato

.png, e in maniera analoga la confusion matrix del miglior modello di ML sul test set. Inoltre restituisce le performance sul test set in formato .txt, contenenti Precision, Recall e F1 score per ogni classe.

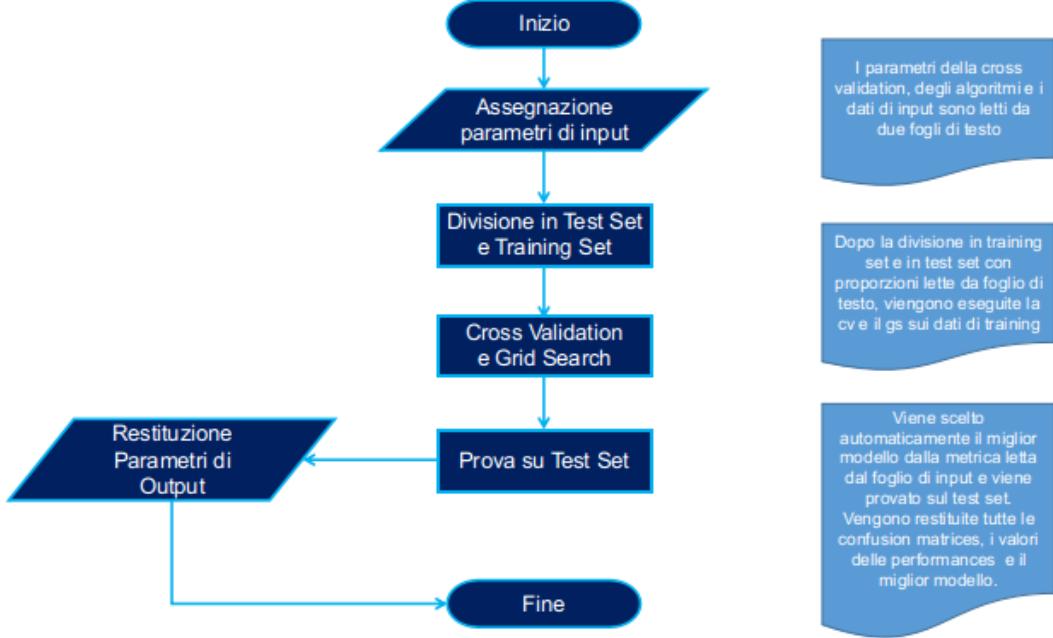


Figura A.2: Diagramma di flusso del programma Model Evaluation.

Il dataset deve essere presente in un foglio di testo, come in figura A.3: le colonne devono essere separate da un carattere ';' e la prima riga deve contenere i nomi delle colonne. È importante che la colonna dei label sia nominata 'Y' e che sia nominato X.txt. Il foglio dei parametri, 'Parameters.txt' deve contenere nella prima riga il numero di Folds per la cross validation, nella seconda la proporzione del Test Set e nella terza riga la metrica su cui si valuta la bontà del modello di ML, tra quelle per la classificazione presenti in [https://scikit-learn.org/stable/modules/model\\_evaluation.html](https://scikit-learn.org/stable/modules/model_evaluation.html).

---

```

VP;VS;RHO;Y
2551.60589643827;1108.12203644923;2258.78390659771;1
2373.31365284184;1106.26621926347;2201.71670167433;2
2111.99685823507;1083.86279056974;2108.86663804490;3
1864.28528552779;1052.03520454825;2012.00968676564;3
1687.90466515248;1014.11726291987;1931.69862449060;3
1599.75199281652;980.476128227330;1888.85297496685;3
1596.70692716915;968.849628885194;1887.11702781503;3
1625.16757831146;970.695824294500;1899.12615124218;3
1644.01949799450;984.318907122628;1909.31406674909;3

```

Figura A.3: Struttura del foglio di testo degli input da cui il programma legge il dataset iniziale.



# Bibliografia

- [1] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.
- [2] F. Aminzadeh and P. De Groot, *Neural networks and other soft computing techniques with applications in the oil industry*. Eage Publications, 2006.
- [3] T. Zhao, V. Jayaram, A. Roy, and K. J. Marfurt, “A comparison of classification techniques for seismic facies recognition,” *Interpretation*, vol. 3, no. 4, pp. SAE29–SAE58, 2015.
- [4] P. Avseth, T. Mukerji, and G. Mavko, *Quantitative Seismic Interpretation*. 2005.
- [5] *Google Machine Learning Glossary*.
- [6] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*. 2013.
- [7] H. Trevor, T. Robert, and F. JH, “The elements of statistical learning: data mining, inference, and prediction,” 2009.
- [8] S.-l. Developers, *Scikit-learn User Guide*.
- [9] S. Bengio, “An introduction to statistical machine learning - theoretical aspects -,” June 2003.
- [10] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “Liblinear: A library for large linear classification,” *Journal of Machine Learning Research* 9, 2008.
- [11] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan, “A dual coordinate descent method for large-scale linear svm,” in *Proceedings of the 25th international conference on Machine learning*, pp. 408–415, ACM, 2008.
- [12] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*, vol. 398. John Wiley & Sons, 2013.
- [13] C.-J. Lin, R. C. Weng, and S. S. Keerthi, “Trust region newton method for logistic regression,” *Journal of Machine Learning Research*, vol. 9, no. Apr, pp. 627–650, 2008.

- [14] M. S. Apostolopoulou, D. G. Sotiropoulos, I. E. Livieris, and P. Pintelas, “A memoryless bfgs neural network training algorithm,” in *2009 7th IEEE International Conference on Industrial Informatics*, pp. 216–221, IEEE, 2009.
- [15] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.
- [16] M. Bicego and M. Loog, “Weighted k-nearest neighbor revisited,” in *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 1642–1647, IEEE, 2016.
- [17] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.
- [18] N. Japkowicz and M. Shah, *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.
- [19] A. Buland and H. Omre, “Bayesian linearized avo inversion,” *Geophysics*, vol. 68, no. 1, pp. 185–198, 2003.
- [20] K. Aki and P. Richards, “Quantitive seismology: Theory and methods. 1a. ed,” *San Francisco (EEUU): Freeman and Company*, vol. 1, p. 557, 1980.
- [21] B. W. Silverman, *Density estimation for statistics and Data Analysis*. Chapman and Hall, 1986.
- [22] A. de Nicolao, G. Drufuca, and F. Rocca, “Eigenvalues and eigenvectors of linearized elastic inversion,” *Geophysics*, vol. 58, no. 5, pp. 670–679, 1993.
- [23] R. Kohavi and F. Provost, “Special issue on applications of machine learning and the knowledge discovery process,”
- [24] T. Cui and G. F. Margrave, “Seismic wavelet estimation,” *the 26th Annual Report of the CREWES Project*, 2014.