Indice

1 Analisi dei dati di probabilità

Durante il periodo 20/05 - 24/05 mi sono occupata di analizzare la probabilità che ha un candidato di rispondere correttamente alle domande in fase di test; valutando le relazioni di dipendenza che possono esistere tra più domande e l'impatto che può assumere la fortuna.

1.1 Problema in esame

Test, sottoposto ad un candidato durante un colloquio, composto da *domande a tripla risposta multipla.*

Nel suddetta sezione vengono analizzate le relazioni che intercorrono tra due domande, denominate A e B, a seconda se il candidato risulta in grado di rispondervi correttamente o meno.

1.2 Caratteristiche degli eventi di coppia

Tipi di eventi trattati:

- **Eventi indipendenti**;
- **Eventi dipendenti**:
  - A e B sono strettamente dipendenti;
  - A implica B.
- **Evento conosciuto ed evento indovinato**.

Struttura usata per rappresentare la probabilità degli eventi di coppia:

$$AB$$

$$/\backslash$$

$$A\ B$$

$$\backslash/$$

$$Z$$

con:

- AB rappresenta la probabilità complessiva dell'evento che si verifica sempre;
- A rappresenta la probabilità che permette il verificarsi di A, ma non di B;

- B rappresenta la probabilità che permette il verificarsi di B, ma non di A;
- Z rappresenta la probabilità a zero, l'impossibilità del verificarsi dell'evento.

1.2.1 Eventi indipendenti

A e B sono due domande la quali risposte sono completamente scorrelate tra di loro.

$$P(A)P(B)$$

$$/\backslash$$

$$P(A)(1-P(B))\ (1-P(A))P(B)$$

$$\backslash/$$

$$(1-P(A))(1-P(B))$$

Considerazioni generali La probabilità complessiva nel caso di domande indipendenti A e B viene data da $P(A)$ per $P(B)$.
Se è conosciuta dal candidato la risposta alla domanda A ma non alla domanda B la probabilità di ottenere una risposta corretta è $P(A)$, mentre la probabilità di ottenere una risposta non corretta per B vale $1-P(B)$. Il ragionamento duale è svolto nel calcolo della probabilità per la riposta corretta alla domanda B ma non ad A.
La probabilità di non ottenere alcuna risposta corretta alle due domande viene calcolata prendendo in considerazione gli eventi contrari a quelli coinvolti. Dunque per A la probabilità che il candidato non conosca la soluzione è $1-P(B)$, dualmente per B la probabilità è $1-P(A)$.

1.2.2 Eventi dipendenti

A e B sono due domande fortemente correlate tra di loro se si risponde correttamente ad una delle due domande si risponde correttamente anche all'altra.

$$P(A)^2$$

$$/\backslash$$

$$0\ 0$$

$$\backslash/$$

$$(1-P(A))^2$$

Considerazioni generali La probabilità complessiva nel caso di domande dipendenti A e B viene data da $P(A)$ per $P(B)$; ma $P(A) = P(B)$ dunque $P(A)^2 = P(B)^2$.
Conseguentemente se il candidato non conosce la risposta alla domanda A non può conoscere la risposta alla domanda B percui la probabilità di conoscere uno dei due eventi è pari a 0.
In questo caso la probabilità a 0 è $(1 - P(A))(1 - P(B)) = (1 - P(A))^2$ essendo che A=B.

A implica B Se si sa rispondere alla domanda A di conseguenza si è in grado di rispondere anche alla domanda B.
Tuttavia non vale il ragionamento opposto, se si sa rispondere alla domanda B non significa che si è in grado di rispondere alla domanda A.

$$P(A)$$

$$/\backslash$$

$$0 \; P(B) - P(A)$$

$$\backslash/$$

$$1 - P(B)$$

Considerazioni generali La probabilità complessiva nel caso di domande dipendenti A e B viene data esclusivamente da $P(A)$ in quanto la conoscenza di sia di A che di B è possibile solo se si ha piena conoscenza di A.
Dunque la probabilità che si conosca la risposta alla domanda A ma non a B è impossibile (pari a 0); mentre se si ha conoscenza della domanda B ma non di A la probabilità si stanzia a $P(B) - P(A)$.
La probabilità a zero è $1 - P(B)$ indicatore dell'impossibilità di avere la risposta corretta per A.

1.3 Evento conosciuto ed evento indovinato

Durante un test il candidato deve saper scegliere la risposta, corretta o meno, alla domanda posta. Le variabili che entrano in gioco durante l'esecuzione dell'atto non riguardano esclusivamente la conoscenza personale del singolo. La probabilità di un evento A è data dalla formula:

$$P(A) = P(A_C) + P(A_I)$$

Le variabili in uso sono:

- $P(A_C)$: probabilità che il candidato sappia rispondere alla domanda A correttamente per sua conoscenza;
- $P(A_I)$: probabilità che il candidato sappia rispondere alla domanda A correttamente indovinando.

Per quanto appena definito sopra valgono le seguenti proprietà:

1. $P(B_C|A_C) = 1$
2. $P(B_C|A_I) = P(B_C)$
3. $P(B_I|A_C) = 0$
4. $P(B_I|A_I) = P(B_I)$

1.3.1 Probabilità di rispondere correttamente ad una domanda

Variabili coinvolte:

- $P(A)$: probabilità necessaria perchè si verifichi, per la domanda A, che il candidato dia la risposta corretta. Per la legge dei grandi numeri la frequenza porta alla probabilità.
- S_0: insieme dei casi in cui in una domanda non viene scartata alcuna risposta dal dominio delle risposte possibili;
- S_1: insieme dei casi in cui in una domanda viene scartata una risposta dal dominio delle risposte possibili;
- S_2: insieme dei casi in cui in una domanda vengono scartate due risposte dal dominio delle risposte possibili.
- $P(I)$: probabilità di dare la risposta corretta alla domanda A indovinando;
- $P(C)$: probabilità di dare la risposta corretta alla domanda A per conoscenza.

Sapendo che $P(I) = P(A) - P(C)$ logicamente vale anche $P(A) = P(I) + P(C)$.

Se un candidato non è in grado scartare alcuna risposta dalla domanda ha 1 possibilità su 3 di, indovinando, dare la risposta corretta. Se un candidato invece risulta in grado di scartare una risposta sbagliata alla domanda, rimane con 1 possibilità su 2 di poter dare la risposta corretta. Se invece, caso ottimo, il candidato ha piena conoscenza della domanda posta risulta in grado di scartare due risposte sbagliate lasciando un'unica risposta possibile, quella esatta. Il ragionamento sopra espresso può venire espresso con la seguente espressione:

$$P(A) = P(S_0)\frac{1}{3} + P(S_1)\frac{1}{2} + P(S_2)$$

Ora individuiamo quale è la probabilità effettiva per un candidato di dare la risposta corretta ad una domanda A.

$1 = S_0 + S_1 + S_2$
$S_0 = 1 - S_1 - S_2$

Sostituendo:

P(A)= $(1 - P(S_1) - P(S_2))\frac{1}{3} + P(S_1)\frac{1}{2} + P(S_2)$
$= \frac{1}{3} - \frac{1}{3}P(S_1) - \frac{1}{3}P(S_1) + \frac{1}{2}P(S_1) + P(S_2)$
$=\frac{1}{3} + \frac{1}{6}P(S_1) + \frac{2}{3}P(S_2)$

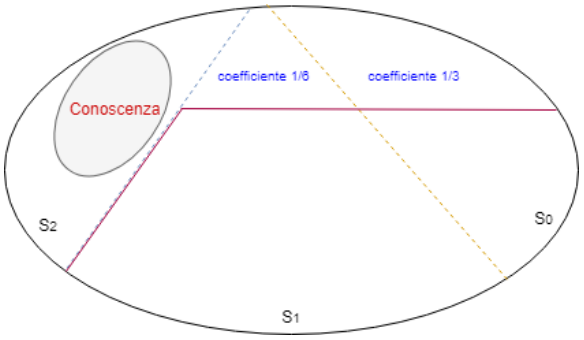

Figura 1: Rappresentazione insiemistica della probabilità di rispondere correttamente ad una domanda: P(A)

Considerazioni importanti

In conclusione P(A)=$\frac{1}{3} + \frac{1}{6}P(S_1) + \frac{2}{3}P(S_2)$. Ovvero la probabilità per un candidato di dare in una domanda A la risposta corretta dipende dai seguenti fattori:

- $\frac{1}{3}$: coefficiente che rappresenta la probabilità effettiva per chi non conosce la risposta alla domanda di dare la risposta corretta;
- $\frac{1}{2}P(S_1)$: coefficiente che rappresenta la probabilità effettiva di dare la risposta corretta quando il candidato è in grado di scartare una risposta sbagliata alla domanda;
- $\frac{2}{3}P(S_2)$: coefficiente che rappresenta la probabilità effettiva di dare la risposta corretta quando il candidato è in grado di scartare due risposte sbagliate alla domanda.

Dall'analisi della tipologia di eventi di coppia e dal calcolo della probabilità necessaria per poter rispondere correttamente ad una domanda, si giunge alla valenza dei seguenti assiomi:

1. Le coppie di domande A e B devono essere fra loro disgiunte, altrimenti si genererebbero situazioni di invalidità dei risultati;

2. Per rispondere correttamente ad una domanda non è necessario che il candidato abbia piena conoscenza di tutti gli argomenti richiesti dall'esame, ma bensì ne risultano sufficienti $n - 1$;

3. La probabilità di conoscere è contenuta all'interno di S_2, in quanto se un candidato conosce è conseguentemente in grado, da una domanda, di scartare due risposte sbagliate.

1.3.2 Il piano

La probabilità P(A) che un candidato ha in gioco nel momento in cui si approccia a rispondere ad una domanda può venire rappresentata in un piano.

Di seguito viene mostrata l'immagine di un modellino, rappresentativo di P(A), realizzato durante l'analisi.
TODO: foto modello

Ognuno dei tre assi cartesiani rappresenta l'insieme dei casi di scarto (S_0, S_1, S_2). L'intersezione tra i punti del piano indica la regione accettabile contenente il range di valori assumibili da P(A). Tale punto proiettato su ognuno dei tre assi permette l'individuazione esatta dei coefficienti delle variabili S_0, S_1, S_2.

Ogni porzione della regione del piano viene individuata con la seguente tecnica:

1. Per individuare ogni retta passante per S_0, S_1 e S_2 è necessario assumere che $S_0 + S_1 + S_2 = 1$;

2. La retta passante per S_0 è rappresentabile per mezzo delle seguenti equazioni:

$$S0 = 0 \text{ e } S_1 + S_2 = 1$$

   In questo modo l'asse S_0 è fissato a 0 e estrapolando S_1 e S_2 da $S_1 = -S_2 + 1$ assumono valori tra (0,1).

Figura 2: Rappresentazione della retta passante per $S_0 = 0$

3. Il medesimo ragionamento vale per le rette passanti per S_1 e S_2.

$$S_1 = 0 \text{ e } S_0 + S_2 = 1$$

l'asse S_1 è fissato a 0 e S_0 e S_2 assumono valori tra (0,1).

$$S_2 = 0 \text{ e } S_1 + S_0 = 1$$

l'asse S_2 è fissato a 0 e S_0 e S_2 assumono valori tra (0,1).

4. In questo modo l'unione di tutte le rette passanti per gli assi creano la regione accettabile dei valori di P(A).

Avendo rappresentato il piano si ottiene nei punti di intersezioni fra le tre rette la regione accettabile per P(A). Inoltre è possibile, ora, individuare il fascio di rette che tangenti il piano permettono di affermare se una specifica domanda è, in base alla sua frequenza, ha difficoltà bassa, media, alta per un candidato.

- Se una domanda ha una difficoltà bassa la retta si situa passante per i punti $0 < S_2 <= 1$ (molto vicino a 1) e $(S_0, S_1) < 0$ (tendenti a 0);
- Se una domanda ha una difficoltà alta la retta si situa passante per i punti $S_2 <= 0$ (molto vicino a 0), $S_1 < 1$ e $S_0 <= 1$ (tendente a non scartare alcuna risposta);

- Se una domanda ha una difficoltà media la retta si situa nella parte centrale della regione accettabile, passante per i punti 0 <= (S_0, S_1, S_2) <= 1.

Rappresentazione di P(A)

Vediamo alcuni casi di come le domande possono venire rappresentate sul piano:

La funzione di partenza è:

$$F = \frac{1}{3} + \frac{1}{6}S_1 + \frac{2}{3}S_2$$

Va esplicitato S_1, i passaggi utili da fare sono i seguenti:

$$\frac{-1}{6}S_1 = \frac{1}{3} + \frac{2}{3}S_2 - F \rightarrow S_1 = -4S_2 - 2 + 6F$$

Essendo che 0 <= S_2 <= 1 usando $S_1 = 1$ e $S_2 = 0$ si ottiene che $F = \frac{1}{2} = 0.5$

Quanto appena calcolato può venire rappresentato graficamente impiegando la retta $S_1 = 1 - S_2$ (responsabile di definire una porzione del piano in base alle variaibili coinvolte) e mediante la retta $S_1 = -4S_2 - 2 + 6F$ (che permette di calcolare il fascio di rette tangenti alla prima retta).

Figura 3: Rappresentazione di P(A) per una frequenza 0.5 proiettata su assi $S_0 = 0$, S_1 e S_2.

Nella figura sopra sono rappresentati i seguenti significati:

- La linea azzurra rappresenta $S_2 = 1 - S_1$;
- La linea rosa rappresenta la retta tangente $S_1 = -4S_2 + 1$;
- Punto A (blu):
  $S_1 = 0.5 = \frac{1}{2}$
  $\frac{1}{2} = -4S_2 + 1 \rightarrow 4S_2 = 1 - \frac{1}{2} \rightarrow S_2 = \frac{1}{8}$
  $S_0 = 1 - \frac{1}{2} - \frac{1}{8} = \frac{3}{8}$

  Ovvero metà dei candidati sottoposti alla domanda sa scartare una delle risposte, lo 0.16% sa dare la risposta corretta e lo 0.36% non sa scartare alcune delle risposte possibili.
- Punto B (verde):
  $S_1 = 0$
  $S_2 = \frac{1}{4}$
  $S_0 = 1 - \frac{1}{4} = \frac{3}{4}$

  Ovvero nessun dei candidati sottoposti alla domanda sa scartare una delle risposte, lo 0.25% sa dare la risposta corretta e lo 0.75% non sa scartare alcune delle risposte possibili.
- Punto C (fucsia):
  $S_1 = \frac{1}{4}$
  $S_2 = \frac{3}{16}$
  $S_0 = 1 - \frac{1}{4} - \frac{3}{16} = \frac{9}{16}$

  Ovvero lo 0.25% dei candidati sottoposti alla domanda sa scartare una delle risposte, lo 0.19% sa dare la risposta corretta e lo 0.56% non sa scartare alcune delle risposte possibili.
- Punto D (arancione):
  $S_1 = \frac{1}{3}$
  $S_2 = \frac{1}{3}$
  $S_0 = 1 - \frac{1}{3} - \frac{1}{3} = \frac{1}{3}$

  Osserviamo che il punto in esame fuoriesce dalla regione delimitata dalla retta tangente di frequenza 0.5 ($S_1 = -4S_2 + 1$). Conseguenza diretta data dall'impossibilità di ottenere una probabilità del 50% sulla domanda con $\frac{1}{3}$ di candidati che sa scartare 2 risposte, $\frac{1}{3}$ che ne sa scartare 1 e $\frac{1}{3}$ nessuna.

Vediamo ulteriori due esempi che permettono di valutare cosa accade nel piano nel caso di una frequenza:

1. Quasi in prossimità di 1;

2. Pari alla soglia minima dell'indovinato.

Il grafico è il seguente:

Figura 4: Rappresentazione di P(A) per una frequenza 0.33 e 0.8 proiettate su assi $S_0 = 0$, S_1 e S_2.

- La linea azzurra mostra la retta tangente con frequenza 0.80%. In questa abbiamo calcolato il punto E (giallo):
  $S_1 = 0$
  $S_2 = \frac{14}{20}$
  $S_0 = 0$

  Quasi la totalità dei candidati ha la conoscenza per poter scartare tutte le risposte sbagliate e dare la risposta giusta alla domanda.

- La linea viola mostra la retta tangente con frequenza 0.33%. In questa abbiamo calcolato il punto F (rosa):
  $S_1 = 0$
  $S_2 = 0$
  $S_0 = 1$

  Ovvero nessuno dei candidati ha la conoscenza per poter scartare nè una nè due risposte, percui l'unica possibilità per un candidato di rispondere alla domanda è indovinare. È evidente come se un candidato non sa la risposta ad una domanda ha una probabilità dello 0.33% di poter indovinare la risposta corretta.

2 Rete neurale

La libreria utilizzata per sviluppare la Rete neurale è stata *ConvNetJS*. L'aspetto positivo di tale scelta è stata la semplicità nell'utilizzo del linguaggio javascript; l'aspetto negativo ha riguardato la totale mancanza di mantenibilità della libreria stessa che comporta la scarsità di esempi applicativi, oltre alla documentazione ufficiale, che costringono lo sviluppatore ad una ricerca approfondita personale in un ambiente ove lo nozioni si presentano scarse e a continue prove per verificare la validità del codice prodotto.

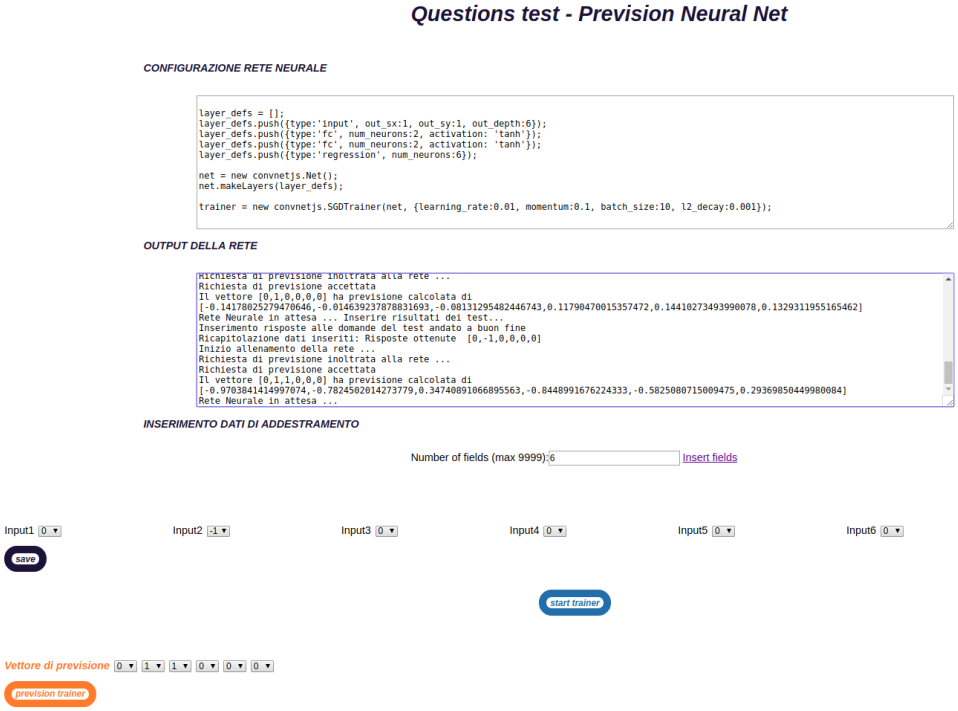

Figura 5: Interfaccia utente della Rete neurale di prova.

Durante il periodo 24/05 - 31/05 mi sono occupata dello sviluppo di una Rete neurale in grado di ricevere in input un training set di dimensione 6 e di restituire una previsione sui dati di apprendimento ricevuti. Il problema che la rete mira ad analizzare è quello discusso nel precedente capitolo *Analisi dei dati di probabilità*

Per agevolare l'apprendimento della rete, ed ottenere delle previsioni stabili mi sono occupata di implementare due metodi di generazione randomica di dati in modo da far apprendere massicciamente la stessa. Il dato prodotto consiste in un vettore di 6 elementi, composto da -1, 0 e 1 con il seguente criterio:

- **-1**: la domanda x è stata posta al candidato che ha risposto in maniera errata;
- **0**: la domanda x non è stata posta al candidato;
- **1**: la domanda x è stata posta al candidato che ha saputo rispondere correttamente.

Il primo metodo che ho sviluppato si occupa di generare un vettore di dati di apprendimento basandosi esclusivamente su come le domande sono interconnesse tra di loro (grazie all'uso di un grafo della conoscenza costruito ad hoc); il secondo metodo ripropone quanto perseguito dal primo metodo con il valore aggiunto di generazione di un profilo randomico di un candidato, che tiene conto della probabilità di risposta ad una domande seguendo la formula P(A)= $\frac{1}{3} + \frac{1}{6}P(S_1) + \frac{2}{3}P(S_2)$.

2.1 Test effettuati

Alcune decisioni che ho preso durante la scelta dell'architettura della rete riguardano i seguenti settori:

1. Una rete neurale non deve, per fornire dei dati attendibili, possedere un numero di neuroni troppo elevato rispetto al trainset effettuato; altrimenti la previsione ritornerebbe l'identità del vettore di input della stessa, come conseguenza diretta della capacità troppo elevata di immagazzinare dati.
2. I layers, ho deciso, di allenarli mediante tecnica di regressione, che permette l'inserimento in input di una funzione obiettivo e l'ottenimento di un risultato, in output, anche in virgola mobile e composto di tanti elementi quanti sono i neuroni di regressione dichiarati. Per la mia rete di prova è necessario dichiarare 6 neuroni in regressione perchè l'output, appunto, che ci si aspetta dal sistema è di 6 elementi.
3. Per costruire un dataset di dati consistente che permettesse alla rete di imparare qualcosa ho costruito un grafo della conoscenza con lo scopo di mettere in relazione degli argomenti che coinvolgono uno o più domande.

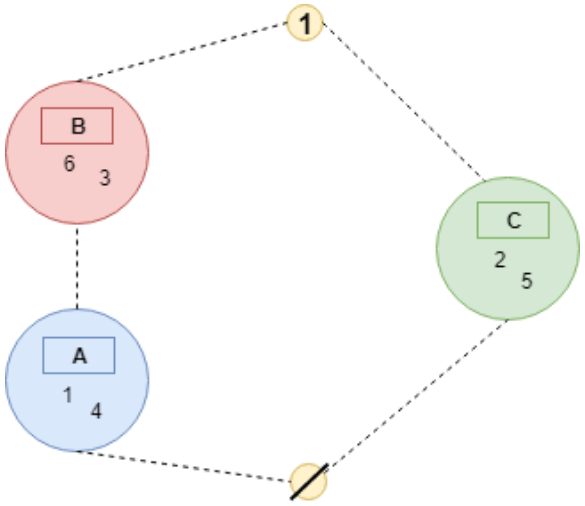

Figura 6: Grafo rappresentante le relazioni esistenti tra il set di domande di prova.

Per svolgere l'apprendimento ogni vettore, facente parte del dataset, viene dato in pasto alla rete che a sua volta provvede alla sua assimilazione come conoscenza mediante la tecnica dell'autoencoder, ovvero la rete impara il vettore riducendone lo spazio occupato.

4. Per creare il dataset ho ritenuto sufficiente generare *2000* vettori di risposta in modo da compiere in maniera esaustivo l'apprendimento della rete.

Il vettore passato in input per svolgere le previsioni è *[0,0,0,0,0,0]*,*[0,0,1,0,1,0]* e *[0,0,-1,0,0,0]*
Le aspettative riguardano la previsione di risposta di un candidato . Di seguito riporto quanto è stato rilevato in fase di test.

2.1.1 Architettura della rete: 4 neuroni per ciascuno dei 2 layers

Architettura della rete utilizzata:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:6});

net = new convnetjs.Net();
net.makeLayers(layer_defs);
```

```
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
 momentum:0.1, batch_size:10, l2_decay:0.001});
```

I layers utilizzati sono 2 e compositi da 4 neuroni.

Training set standard a 4 neuroni per ciascuno dei 2 layers

- `Il vettore [0,0,0,0,0,0] ha previsione calcolata di [-0.021598804903572744,-0.1372509042342871,0.06611969158456255, 0.018121335417653706,-0.11264571886853292,0.17520370837747462]`

  Appaiono in relazione le domande 1, 2, 5 e 3, 4, 6.
  Gli scostamenti tra le coppie 2, 5 e 3, 6 sono consistenti con quelle che sono le relazioni di dipendenza, invece 1, 4 ha una differenza di 0.016 circa che parte da qualche millesimo fino 0.5 Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4; nel test in analisi questo non viene rispettato da nessuna delle coppie in analisi per differenze che vanno da qualche millesimo fino a 0.018 circa.

  Osservazioni

  L'architettura testata si compone di 4 neuroni a layer su una base di 2000 test correndo il rischio di avere una rete che apprende troppo e come effetto negativo "veda" addirittura cose che non esistono. A prova di ciò sono i risultati non conformi alle attese. Dunque mi fermo qui con il test di tale architettura e riducendone il numero di neuroni presenti in ciascun layers e/o il numero di layers presenti.

  Le nuove architetture su cui ho effettuato i test sono esposte nei paragrafi seguenti.

2.1.2 Architettura della rete a 2 neuroni per ciascuno dei 2 layers

Architettura della rete utilizzata:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
```

```
layer_defs.push({type:'regression', num_neurons:6});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
 momentum:0.1, batch_size:10, l2_decay:0.001});
```

I layers utilizzati sono 2 compositi da 2 neuroni.

Training set standard su rete a 2 neuroni per ciascuno dei 2 layers

- `[Il vettore [0,0,0,0,0,0] ha previsione calcolata di [0.31232372051574936,0.7253754889487585,-0.5051208979797573, 0.32075742158673093,0.7324947496336937,-0.4348299972940168]`
  Appaiono in relazione le domande 1, 2, 4, 5 e 3, 6.
  Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4; in questo test la regola viene rispettata pienamente.
  Dai dati della previsione si nota come il candidato ha una buona probabilità di saper rispondere alla coppia 1 e 4, e ancora più elevata di saper rispondere correttamente alla coppia 2 e 5; molto bassa di saper rispondere correttamente alle 3 e 6 che sono, appunto, di una difficoltà maggiore rispetto alla coppia 1 e 4.
- `Il vettore [0,0,1,0,1,0] ha previsione calcolata di [0.5123144717131076,0.9123354449531641,0.2837937822420923, 0.46449868699771607,0.9029832167165894,0.3227303792035435]`
  Appaiono in relazione le domande 1, 2, 3 4, 5, 6.
  Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4; in questo test la regola viene rispettata pienamente.
  Dai dati della previsione si nota come il candidato ha un'ottima probabilità di saper rispondere alla coppia 2 e 5 (come imposto dal vettore previsione), buona di saper rispondere alla coppie 3 e 6 (come imposto dal vettore previsione) e più che buona di saper rispondere alle 1 e 4, che sono di una semplicità più elevata rispetto alla 3 e 4.

- `Il vettore [0,0,-1,0,0,0] ha previsione calcolata di [0.3698539826215957,0.288907514487717,-0.8504159455662308, 0.3663192502433841,0.2937448801761998,-0.7845589473185985]`

  Appaiono in relazione le domande 1, 2, 4, 5 e 3, 6.
  Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4; in questo test la regola viene rispettata pienamente.
  Dai dati della previsione si nota come il candidato ha una discreta probabilità di saper rispondere alla coppia 2 e 5, un pò meglio di saper rispondere alla coppie 1 e 4 e più di non saper saper rispondere alle 3 e 6 (come imposto dal vettore previsione).

Training set con generazione del profilo di un candidato e calcolo delle probabilità di risposta a 2 neuroni per ciascuno dei 2 layers

- `Il vettore [0,0,0,0,0,0] ha previsione calcolata di [0.057781303506280995,0.0513731100126314,-0.06600467867066256, 0.029940883111932555,-0.019564515397168573,-0.09570617900597932]`

  Appaiono in relazione le domande 1, 2, 4 e 3, 5, 6.
  Gli scostamenti tra la coppia 1, 4 e 3, 6 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande; invece per la coppia 2 e 5 i segni sono opposti con una differenza di 0.024. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4, la regola viene rispettata pienamente. Le anomalie riscontrate sono da ricondurre alla natura del vettore di training che si basa sul calcolo della probabilità di una risposta che sul grafo della conoscenza.

- `Il vettore [0,0,1,0,1,0] ha previsione calcolata di [0.19494624113789977,0.1712744021266377,0.577963304906936, 0.781098215373483,0.3774535909060714,0.03617314870307162]`

  Appaiono in relazione le domande 1, 2, 3, 4, 5, 6.
  Gli scostamenti tra le coppie 1 e 4 , 2 e 5, 3 e 6 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4, la regola non viene rispettata dalla domanda 1 in rapporto con la domanda per una differenza di 0.37 circa. Le anomalie riscontrate sono da ricondurre alla natura del vettore di training che si basa sul calcolo della probabilità di una risposta che sul grafo della conoscenza.

- `Il vettore [0,0,-1,0,0,0] ha previsione calcolata di [0.09845785763965222,0.015421380649956663,-0.5138068038427066, -0.4853190165287735,-0.22629262719814794,0.0008152164571250502]`

Appaiono in relazione le domande 1, 2, 6 e 3, 4, 5.
Gli scostamenti tra le coppie 1, 4 e 2, 5 e 3, 6 per una differenza tuttavia trascurabile che oscilla dallo 0.2 allo 0.5. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4, la regola non vale per la coppa 6 e 4. Le anomalie riscontrate sono da ricondurre alla natura del vettore di training che si basa sul calcolo della probabilità di una risposta che sul grafo della conoscenza.

Osservazioni

Confrontando i risultati ottenuti dalla rete con i layers impostati a 4 neuroni con quanto emerso dai dati risultanti dalla rete a 2 neuroni emerge come l'architettura a 2 neuroni a layers è sicuramente quella che da i risultati attesi.
Quanto emerso di discordate dal secondo training set è come da aspettative da associare alla natura stessa della creazione del set di dati.

2.1.3 Architettura della rete a 4 neuroni per 1 layer

Architettura della rete utilizzata:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:6});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
 momentum:0.1, batch_size:10, l2_decay:0.001});
```

Viene utilizzato un unico layer da 4 neuroni.

Training set standard su rete a 4 neuroni per 1 layer

- `Il vettore [0,0,0,0,0,0] ha previsione calcolata di [0.12202628618565468,0.08221724740100582,0.02233631914718809, 0.09586625658118901,0.05558075220027264,0.13443779128784109]`

  Appaiono in relazione le domande 1, 2, 3, 4, 5, 6.
  Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4;

in questo test la regola non viene rispettata dalla domanda 6.
Dai dati della previsione si nota come il candidato non ha una buona probabilità di saper rispondere alle domande e la domanda 6 non si presenta conforme alle aspettative.

Osservazioni

Rispetto a quanto osservato nei casi precedenti, ancora l'architettura che rispetta le attese è quella con 2 neuroni per 2 layers.

Tale conclusione ha perfettamente senso in quanto il grafo della conoscenza che ho usato come base per costruire i vettori di apprendimento è composto da 3 nodi (A, B, C) indicanti 3 neuroni; il quarto può venire valutato come un nodo della rete utile per parametri in entrata e in uscita.

Per estendere maggiormente la mia conoscenza della rete, ho provveduto ad aumentare progressivamente il numero di neuroni a layers e osservarne le interazioni. Svolgendo ciò mi sono accorta che il risultato ottenuto dalla previsione era il più possibile vicino al vettore previsione; conseguenza diretta di un numero eccessivo di neuroni dati alla rete per l'apprendimento rispetto al training set svolto, generatrice di una situazione di overfitting e non attendibilità dei dati raccolti. L'architettura a 1 e 2 neuroni invece presenta una buona capacità di previsione in quasi tutti i casi, però tende ad andare in overfitting, come riporto di seguito:

```
Il vettore [0,0,0,0,0,1] ha previsione calcolata di
[0.5613347853884025,0.8310670629630683,-1.03049430206139,
0.5492731069379962,0.5679700877862532,-0.8637707232817535]
```

Il numero di neuroni non è sufficiente per memorizzare che la domanda 6 deve essere positiva, e comporta a cascata la correttezza anche delle domande 3, 4 e 1. La situazione si presenta simile se il layer con 1 neurone è posto al di sotto.

2.2 Sviluppo della rete delle domande nel database aziendale

2.2.1 Montaggio e configurazione della rete

Durante la settimana dal 03/06 al 07/06 la mia attività principale è stata il montaggio e la configurazione della Rete neurale inerente il database aziendale con dataset i colloqui ai candidati. Inoltre ho rivolto parte delle ore a modificare e ottimizzare quanto già implementato nella Rete di prova, in modo che ogni cosa implementata sulla rete del database è presenta anche in versione ridotta.

Per rendere più comprensibile le previsioni di probabilità ottenute, a seguito dell'addestramento della rete e della data in pasto del vettore previsione, ho realizzato un'immagine canvas in cui ogni domanda viene raffigurata con un quadrattino colorato in base alla previsione risultante (verde se a 1, bianco a 0, rosso a -1, gradazioni di bianco - verde e bianco - rosso per i valori intermedi.

Figura 7: Rete di prova dopo lo sviluppo del canvas per le previsioni.

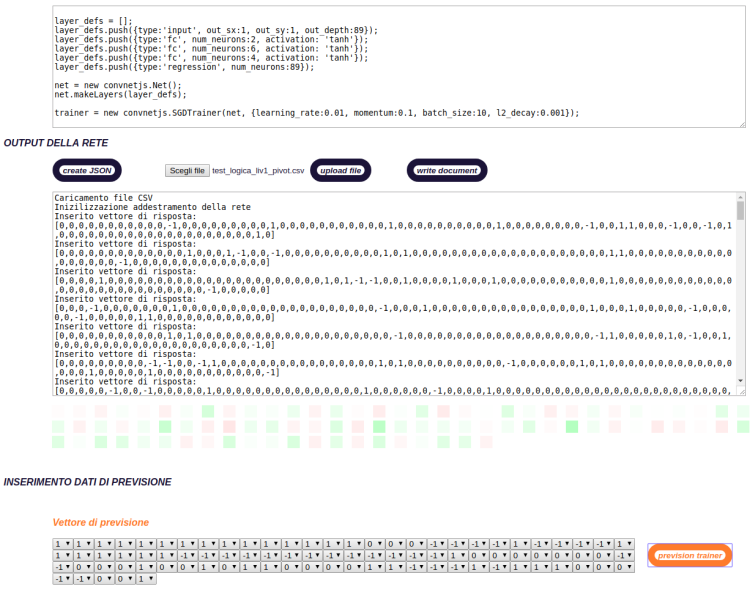

Figura 8: Rete neurale del database aziendale.

Una prima architettura su cui ho deciso di analizzare i risultati della rete, basandomi anche sul quanto appreso dalla rete neurale di prova e dal numero di vettori di test utilizzati (1245 vettori x 89), è stata la seguente:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

Ho aggiunto un layer e messo un numero di neuroni per layer in modo da formare un romboide. Devo verificare la bontà di questa mia scelta o se invece mi porta ad una situazione di overfitting.

2.2.2 Natura delle domande contenute nel database aziendale

Analizzando il training set dei vettori ho riscontrato tali correlazioni:

- Solo una piccola parte delle domande presenti nel database vengono svolte durante un colloquio con un candidato. In media una decina su 89 possibili.
- Dalla rete sembra che le domande abbiano qualche correlazione, tuttavia la configurazione attuale ne rende difficoltosa l'individuazione. Quello che mi sembra opportuno ricercare testando l'architettura di della rete è che si vadano a formare dei cluster.

2.2.3 Test e Documentazione

Durante la settimana dal 10/06 al 18/06 ho effettuato quanto definito al interno del Piano di Lavoro come "Test e Documentazione".

Test nella Rete di prova

Architettura della rete utilizzata:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:6});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
 momentum:0.1, batch_size:10, l2_decay:0.001});
```

- `Il vettore [1,1,1,1,1,1] ha previsione calcolata di [0.8521066399598267,0.898137375081856,0.9993098151218291,0.792190337086403, 0.811145866789799,0.9514731560722426]`

Figura 9: Risultato della rete di prova a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

Quanto mostrato dal dettaglio delle domande ha il seguente significato per un candidato:

- *se la domanda 1 è settata a 1 (corretta)*: la rete prevede che la domanda 1 e 4 abbiano una probabilità alta di essere risposte in modo corretto (verde); la 3 e 6 una probabilità non eccessiva di venire risposte in modo sbagliato (rosa attenuato), la 2 e la 5 di non venire nemmeno poste (bianco con qualche minima sfumatura di verde).
- *se la domanda 2 è settata a 1 (corretta)*: la rete prevede che la domanda 1 e 4 abbiano una probabilità non molto alta di essere risposte in modo corretto (bianco con qualche sfumatura di verde); la 3 e 6 una buona probabilità di venire risposte in modo sbagliato (rosa), la 2 e la 5 di venire date in modo corretto (verde).
- *se la domanda 3 è settata a 1 (corretta)*: la rete prevede che la domanda 3 e 6 abbiano una probabilità comunque bassa di essere risposte in modo corretto (bianco con qualche sfumatura di verde); la 1 e 4 con probabilità di venire risposte in modo corretto (verde) perchè più semplici delle domande 3 e 6, la 2 e la 5 di venire risposte correttamente (verde).
- *se la domanda 4 è settata a 1 (corretta)*: la rete prevede un risultato identico a quanto ottenuto dalla domanda 1.

- *se la domanda 5 è settata a 1 (corretta)*: la rete prevede un risultato similare a quanto ottenuto dalla domanda 2. Cambia solo quanto previsto dalle domande 3 e 6 che si presentano con un rosa un pò più intenso, in quanto con correlate alla coppia di domande 2 e 5.
- *se la domanda 6 è settata a 1 (corretta)*: la rete prevede un risultato simile a quanto ottenuto dalla domanda 3. La coppia 2 e 5 hanno una probabilità minore di essere date correttamente (bianco con sfumature di verde) ma perchè non correlate alle domande 3 e 6.

- `Il vettore [-1,-1,-1,-1,-1,-1] ha previsione calcolata di [0.3440856175367477,-0.5026946644729329,-1.284368009920025, 0.35883842020377565,-0.37844446052773495,-1.1717763012412878]`

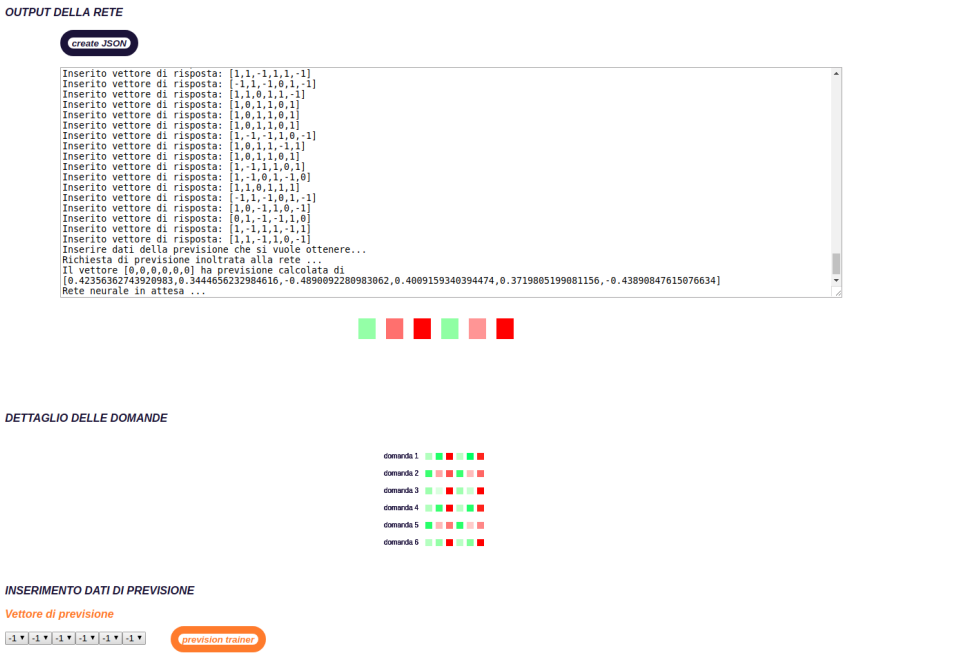

Figura 10: Risultato della rete di prova a seguito di un vettore di previsione [-1, -1, -1, -1, -1, -1] in input.

- *se la domanda 1 è settata a -1 (non corretta)*: la rete prevede che la domanda 1 e 4 non abbiano una probabilità alta di essere risposte in modo corretto (bianco con sfumature di verde); la 3 e 6 una probabilità molto alta di venire risposte in modo sbagliato (rosso), la 2 e la 5 di non venire nemmeno poste (verde con qualche sfumatura di bianco).

- *se la domanda 2 è settata a -1 (non corretta)*: la rete prevede che la domanda 1 e 4 abbiano una probabilità non molto alta di essere risposte in modo non corretto (verde con qualche sfumatura di bianco); la 3 e 6 una buona probabilità di venire risposte in modo sbagliato (rosa), la 2 e la 5 di venire date in modo non corretto (rosa molto atenuato).
- *se la domanda 3 è settata a -1 (non corretta)*: la rete prevede che la domanda 3 e 6 abbiano una probabilità comunque alta di essere risposte in modo non corretto (rosso); la 1 e 4 con bassa probabilità di venire risposte in modo corretto (bianco con qualche sfumatura di verde) perchè più semplici delle domande 3 e 6, la 2 e la 5 di non venire nemmeno poste o comunque basso di venire risposto correttamente(bianco con sfumature di verde).
- *se la domanda 4 è settata a -1 (non corretta)*: la rete prevede un risultato identico a quanto ottenuto dalla domanda 1.
- *se la domanda 5 è settata a -1 (non corretta)*: la rete prevede un risultato similare a quanto ottenuto dalla domanda 2. Cambia solo quanto previsto dalle domande 3 e 6 che si presentano con un rosa un pò meno intenso, in quanto non correlate alla coppia di domande 2 e 5.
- *se la domanda 6 è settata a -1 (non corretta)*: la rete prevede un risultato simile a quanto ottenuto dalla domanda 3. La coppia 2 e 5 hanno una probabilità maggiore di essere date correttamente (bianco con sfumature di verde) ma perchè non correlate alle domande 3 e 6.

Test nella Rete del database

Architetture testate

Durante tutto il periodo ho effettuato una serie di test su molteplici architettura della rete, con gradi di correlazione tra le domande pari al 100% o con uno differenza massima di 5 punti colore rispetto al canvas risultante per ogni domanda.
Tuttavia non sono, durante questo periodo, riuscita ad individuare un'architettura sufficientemente stabile per prevedere risultati attendibili; a causa della molteplicità di dati che hanno aumentato esponenzialmente la complessità di analisi rispetto alla Rete di prova. Tale complessità è rimasta costante anche successivamente allo studio del contenuto delle domande nel database aziendale. Architettura della rete utilizzata:

- `layer_defs = [];`

```
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

Figura 11: Risultato della rete del database a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

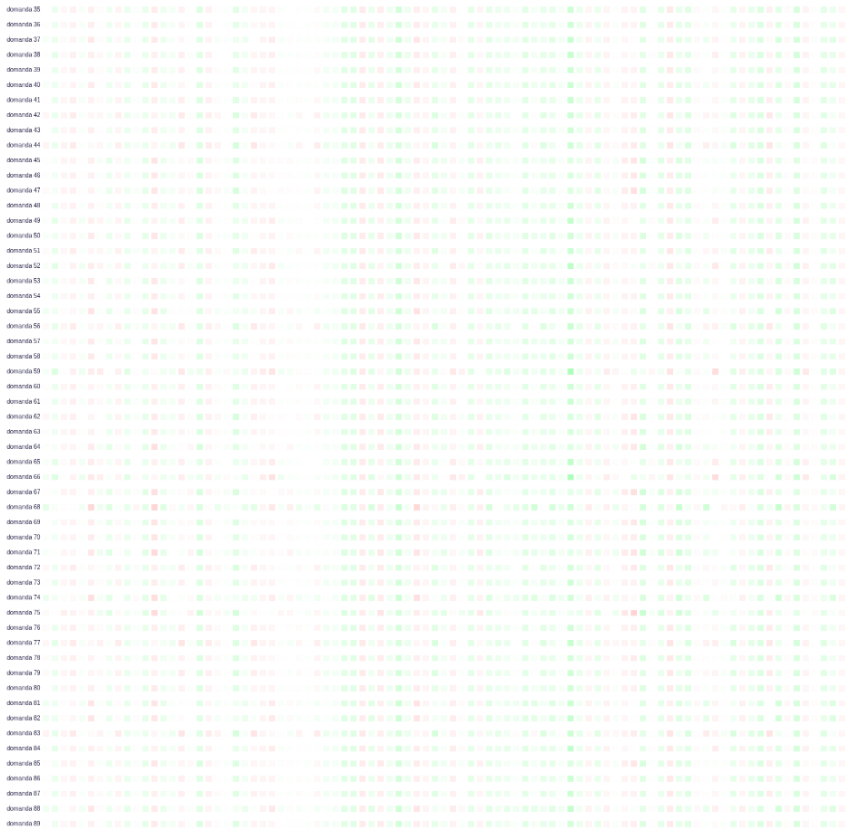

Figura 12: Risultato della rete del database a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

Dagli screen della rete riportati sopra appare come "sembrano" domande:

1. Analisi verticale:
   - *semplici* la 18, 22, 34, 35, 37, 39, 41, 44, 48, 50, 51, 52, 53, 54, 55, 56, 57, 59, 60, 67, 69, 71, 72, 77, 79, 80, 82, 84, 87 e 88. Inoltre di queste sembrano in relazione ancora più stretta le domande 18, 22, 40 e 59.
   - *difficili* la 3, 4, 6, 13, 16, 19, 24, 25, 26, 36, 38, 42, 43, 46, 49, 61, 63, 65, 66, 70, 78, 81, 85 e 89. Inoltre di queste sembrano in relazione ancora più stretta le domande 6, 13, 19, 36, 38, 42, 46 e 70.
2. Analisi orizzontale: Appaiono in relazione stretta le seguenti domande:
   - 2, 3, 4, 5;
   - 7, 8, 9;

- 14, 16;
- 20, 21;
- 26, 32;
- 29, 32
- 33, 34, 35, 36, 38;
- 39, 41, 43;
- 46, 48;
- 49, 52;
- 50, 53;
- 72, 79;
- 81, 82;
- 86, 87, 88.

Figura 13: Risultato della rete del database a seguito di un vettore di previsione [-1, -1, -1, -1, -1, -1] in input.

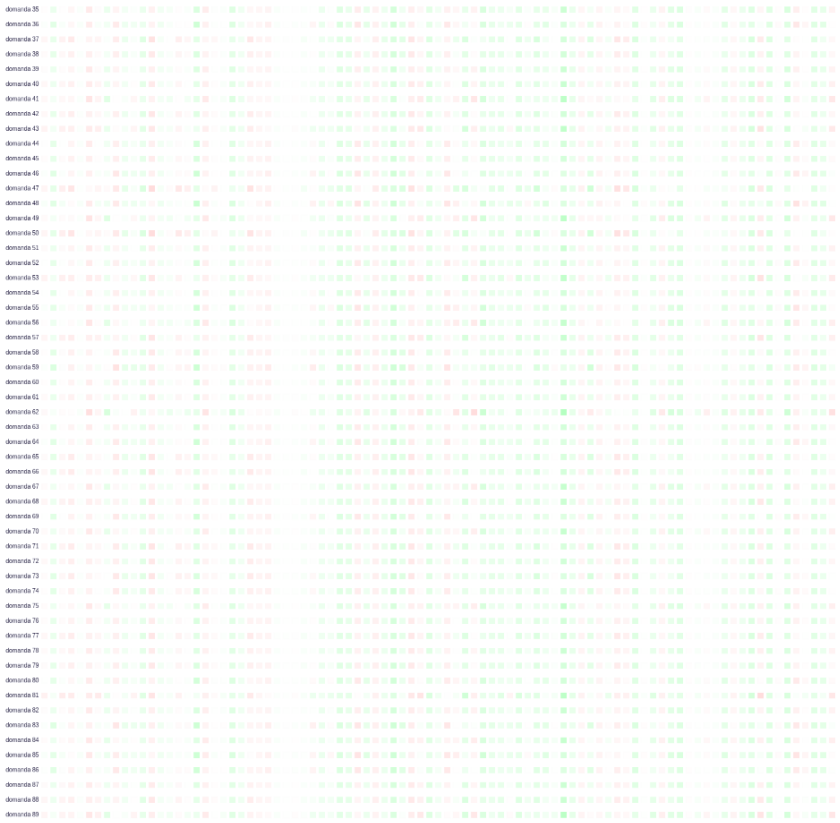

Figura 14: Risultato della rete del database a seguito di un vettore di previsione [-1, -1, -1, -1, -1, -1] in input.

Dagli screen della rete riportati sopra appare come "sembrano" domande:

1. Analisi verticale:
   - *semplici* la 18, 22, 34, 35, 37, 38, 39, 41, 44, 48, 50, 51, 52, 54, 55, 56, 67, 69, 71, 72, 77, 79, 80, 82, 84, 87 e 88. Inoltre di queste sembrano in relazione ancora più stretta le domande 18, 22, 38, 52 e 57.
   - *difficili* la 3, 4, 6, 13, 19, 27, 36, 38, 42, 43, 46, 61, 62, 65, 70. Inoltre di queste sembrano in relazione ancora più stretta le domande 6, 13, 19, 36, 38, 42, 46, 61 e 62.
2. Analisi orizzontale: Appaiono in relazione stretta le seguenti domande:
   - rimaste consistenti con il vettore [1, 1, 1, 1, 1, 1].

- `layer_defs = [];`

```
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:12, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01, momentum:0.1
, batch_size:10, l2_decay:0.001});
```

Noto che aumentando il numero di neuroni sull'unico layer esistente, il valore della domanda corrispondente al vettore della previsione sembra sempre più marcato, segno che la rete "impara troppo" e ricade nel restituire l'immagine stessa del vettore previsione.

Figura 15: Risultato della rete del database a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

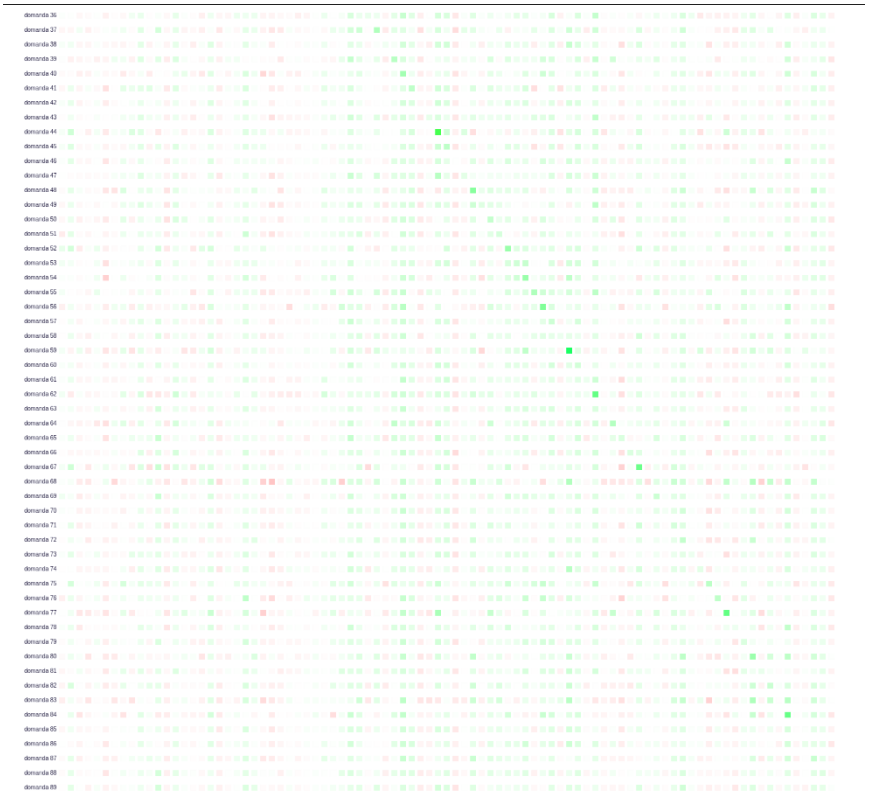

Figura 16: Risultato della rete del database a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

Dagli screen della rete riportati sopra appare come la situazione appare meno lineare del caso analizzato precedentemente. Le domande non vengono separate per linee rete; ma per aree di relazione.

Dagli screen della rete riportati sopra appare come "sembrano" domande:

1. Analisi verticale:
   - *semplici* la 18, 22, 34, 35, 37, 39, 41, 44, 48, 50, 51, 52, 53, 54, 55, 56, 57, 59, 60, 67, 69, 71, 72, 77, 79, 80, 82, 84, 87 e 88. Inoltre di queste sembrano in relazione ancora più stretta le domande 18, 22, 40 e 59.
   - *difficili* la 3, 4, 6, 13, 16, 19, 24, 25, 26, 36, 38, 42, 43, 46, 49, 61, 63, 65, 66, 70, 78, 81, 85 e 89. Inoltre di queste sembrano in relazione ancora più stretta le domande 6, 13, 19, 36, 38, 42, 46 e 70.
2. Analisi orizzontale: Appaiono in relazione stretta le seguenti domande:

– Vengono meno le relazioni individuate precedentemente.

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```
- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:12, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```
- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:12, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:8, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```
- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  ```

```
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001})
```

- ```
  layer_defs = [];
  layer_defs.push
  ({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  ```
  ```

```
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
   momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  ```

```
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  ```
  ```

```
momentum:0.1, batch_size:10, l2_decay:0.001});
```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:5, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  ```

```
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);
  ```
  ```

```
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
 momentum:0.1, batch_size:10, l2_decay:0.001});
```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```
  ```

- ```
  layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:8, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:8, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
  momentum:0.1, batch_size:10, l2_decay:0.001});
  ```

  L'obiettivo dei miei test è stato individuare una configurazione abbastanza "forte" da permettere il riscontro di correlazioni e dei possibili cluster esistenti con il solo ausilio della Rete. Tuttavia per proseguire il mio studio è indispensabile che mi avvalga di anche altri strumenti di supporto. Il prossimo passo è costruire un modello di dati mediante la tecnica di Principal **C**omponent **A**nalysis .
  ```

3 Principal Component Analysis

La **P**rincipal **C**omponent **A**nalysis (acronimo PCA) è una tecnica impiegata nell'ambito della statistica multivariata[1] usata per semplificare i dati d'origine.
Lo scopo che tale tecnica persegue è lo studio della relazione esistenti tra i campioni d'interesse, con la riduzione di un numero più o meno elevato di variabili. La riduzione dimensionale avviene tramite una trasformazione lineare delle variabili coinvolte con lo scopo di effettuare la proiezione di quelle originarie in un nuovo sistema cartesiano, in cui tutte le variabili vengono ordinate in maniera decrescente per ordine di varianza. Successivamente la variabile con maggiore varianza viene proiettata sul primo asse, la seconda sul secondo e via via sempre così per tutte le variabili sotto esame. La PCA è particolarmente utile quando la dimensionalità dello spazio delle misure è elevata (molte colonne); tuttavia i campioni si vengono a trovare in uno spazio di dimensioni significativamente ridotte.
Indispensabile per la buona riuscita della tecnica è, come già esposto sopra, la ricerca del numero di componenti principali significative, ovvero tutte le variabili coinvolte a meno di quelle legate al rumore, che concorrono a comporre la *dimensionalità intrinseca*. Il "rumore" è sempre concentrato nelle ultime variabili, non includerle nell'analisi dei dati porta a dati più puliti, con un rapporto segnale/rumore più alto.
Il calcolo di quali sono le componenti più significative si ottiene con la varianza. La prima componente principale spiega la massima percentuale della variabilità presente nei dati rappresentabili in una singola dimensione, in poche parole la direzione lungo cui si registra la massima dispersione dei dati (quanto il valore medio si discosta). La varianza porta il vantaggio di essere indipendente dal sistema di riferimento, difatti conseguentemente una rotazione degli assi mantiene immutata la varianza totale all'interno del sistema.

3.1 Metodologia applicata

L'analisi e l'implementazione della tecnica di **P**rincipal **C**omponent **A**nalysis l'ho svolta nel periodo 17 - 21 luglio con la seguente metodologia:

- Studio e configurazione di R e R-Studio;
- Studio del significato della tecnica e sua correlazione con le Rete neurali;
- Studio della metodologia di implementazione;
- Scelta del package più adeguato per svolgere l'applicazione della PCA sui dati di trainset;

[1] parte della statistica in cui l'oggetto dell'analisi è almeno composta da due elementi

- Implementazione del metodo della PCA e dei metodi di supporto ad analisi dei dati.
- Analisi dei risultati ottenuti dalla PCA;
- Analisi delle domande presenti nel database aziendale;
- Confronto dei risultati della PCA con quelli ottenuti dalla Rete neurale e confronto della loro validità con le domande del database.

Ogni implementazione e osservazione l'ho svolta prima nei dati del trainset di prova e solo successivamente gli traslati all'interno del trainset delle domande nel database aziendale. In modo ho avuto sempre ben la correttezza e le possibili correlazioni vigenti nel metodo in uso.

3.2 Sviluppo

La tecnica di PCA effettua analisi dei dati. I dati che ho impiegato sono i medesimi che ho utilizzato per effettuare il trainset della Rete neurale. Il formato utilizzato è sempre CSV e i dati contenuti sono strutturati per riga con i risultati di ogni test e per colonna con il risultato di una specifica domanda k.

1. Caricamento del package **factoextra**: la mia scelta è ricaduta su tale package e non in altri con la medesima funzione, per via della sua visualizzazione dei dati elegante basata su ggplot2;
2. **Caricamento del file CSV** in memoria per mezzo della trasposizione in ata frame;
3. *Standardizzazione* dei dati del trainset. Tale compito mi è risultato indispensabile perchè anche se di norma, la standardizzazione, viene usata per evitare situazioni erronee (alcune variabili X presentate con una variabilità molto maggiore rispetto ad altre) e io nei casi in analisi ero in possesso di dati già presentati con la medesima scala; avendo la necessità di individuare gli autovettori e applicare il criterio di Keiser nel trainset ho dovuto procedere ugualmente a standardizzare.
4. Calcolo della **PCA** con la seguente formula:

   ```
   prcomp(df_numeric, scale = FALSE)
   ```

   R offre due metodi per calcolare la PCA:

   - *prcomp(x, scale = FALSE)*: dove x rappresenta una matrice numerica o data frame e *scale* un valore logico che indica se le variabili devono essere ridimensionate/standardizzate;

- *princomp(x, cor = FALSE, scores = TRUE)*: dove x rappresenta una matrice numerica o data frame, *cor* valore logico che se a true ridimensiona e centra i dati prima di procedere all'analisi e *scores* valore logico che se a true calcola le coordinate su ciascun componente principale.
  Metodologia che ho utilizzato per passi:

Ho deciso di far uso della funzione *prcomp* perchè usa la decomposizione del valore singolare (SGV) che offre una precisione leggermente migliore rispetto all'uso del metodo princomp. Il metodo *prcomp* include nei propri elementi di output:

(a) sdev: deviazione standard delle componenti principali;

(b) rotation: la matrice dei carichi delle variabili ovvero le colonne degli autovettori;

(c) center: la media variabile, indica se le variabili devono essere spostate per essere centrate sullo zero;

(d) scale: deviazione standard delle variabili;

(e) x: coordinate degli individui sulle componenti principali.

5. Calcolo degli **autovalori della matrice di covarianza**. Mostra la percentuale di varianze di competenza di ciascun componente principale con

   ```
   get_eig(res.pca)
   ```

6. **Riepilogo** mediante il metodo summary dei risultati ottenuti dal calcolo della PCA. Effettua la standardizzazione dei risultati ottenuti e ne calcola gli autovalori di covarianza con individuazione di deviazione standard, proporzione della varianza e proporzione cumulata.

   ```
   summary(res.pca)
   ```

7. **Calcolo degli autovettori** con

   ```
   loadings(res.pca)
   ```

8. **Individuazione della matrice correlazione** dei dati con

   ```
   cor(df_numeric)
   ```

9. Creazione di un data frame contenete per ogni variabile principale quali sono le componenti ordinate in modo decrescente strettamente correlate.

3.3 Risultati ottenuti

I dati risultati dall'elaborazione dei dati inerenti il trainset delle domande nel database gli ho riportati, per motivi di spazio e comprensione, in modo parziale. Ogni metodo utilizzato viene di seguito descritto nel dettaglio.

3.3.1 Calcolo della Principal Component Analysis

```
> prcomp(df_numeric, scale = FALSE)
Standard deviations (1, .., p=6):
[1] 1.5905164 0.9270882 0.6549647 0.2877259 0.2567623 0.2267723

Rotation (n x k) = (6 x 6):
           PC1         PC2        PC3           PC4          PC5          PC6
V1 -0.3136368 -0.06351095  0.6506652 -0.0008078537  0.660023581  0.196472324
V2  0.3888043  0.53581153  0.2479470  0.7072146111 -0.008846399  0.005359089
V3 -0.5186977  0.45305827 -0.1486278  0.0011520653  0.150101946 -0.693591199
V4 -0.3009414 -0.07357741  0.6112341 -0.0012775235 -0.719969817 -0.109796655
V5  0.3876316  0.53818077  0.2452635 -0.7069936980 -0.008478550  0.006089356
V6 -0.4905017  0.45668194 -0.2435273  0.0019475491 -0.152746837  0.684257823
```

Figura 17: Visualizzazione del calcolo della PCA tramite prcomop per il trainset di prova.

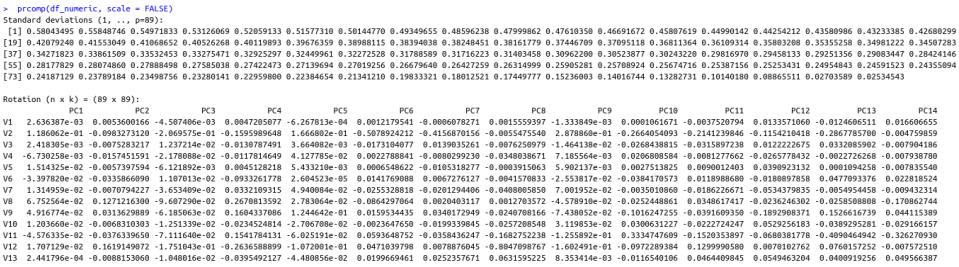

Figura 18: Visualizzazione del calcolo della PCA tramite prcomop per il trainset delle domande nel database.

Osservazioni Osservando esclusivamente la deviazione standard ottenuta dal trainset di prova, appare come i PC da prendere in considerazione per l'analisi sono i primi tre; inoltre nella rotazione appare chiaro ch le valutazioni ottenute da V1 sono in relazione con V4, V3 con V6 e V2 con V5. Tuttavia queste sono solo mere osservazioni senza ancora alcuna prova matematica completa a supporto.

Per quanto concerne il trainset delle domande nel database è molto difficile fare qualunque tipo di assunzione sulla natura dei dati causa la loro numerosità.

3.3.2 Calcolo degli autovettori ed individuazione di Summary

```
> summary(res.pca)
Importance of components:
                          Comp.1    Comp.2    Comp.3     Comp.4     Comp.5      Comp.6
Standard deviation     1.9297771 1.0581456 0.9099156 0.38288957 0.34464930 0.244856110
Proportion of Variance 0.6209837 0.1867054 0.1380601 0.02444629 0.01980709 0.009997418
Cumulative Proportion  0.6209837 0.8076891 0.9457492 0.97019549 0.99000258 1.000000000
> # fornisce gli autovalori della matrice di covarianza/varianza delle dimensioni principali
> get_eig(res.pca)
      eigenvalue variance.percent cumulative.variance.percent
Dim.1 3.72403947       62.0983737                    62.09837
Dim.2 1.11967213       18.6705374                    80.76891
Dim.3 0.82794633       13.8060084                    94.57492
Dim.4 0.14660442        2.4446293                    97.01955
Dim.5 0.11878314        1.9807094                    99.00026
Dim.6 0.05995451        0.9997418                   100.00000
```

Figura 19: Visualizzazione del calcolo del metodo summary ed individuazione degli autovalori per il trainset di prova.

```
> summary(res.pca)
Importance of components:
                          Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7     Comp.8     Comp.9    Comp.10    Comp.11    Comp.12    Comp.13    Comp.14    Comp.15    Comp.16
Standard deviation     1.43136144 1.30933960 1.27248819 1.25540778 1.24609579 1.22900589 1.22206570 1.21814257 1.20907848 1.19278209 1.1790983 1.17045380 1.16824099 1.15753299 1.15668352 1.14908663
Proportion of Variance 0.02303868 0.01927807 0.01820818 0.01772265 0.01746071 0.01698505 0.01679377 0.01668611 0.01643872 0.01599857 0.0156336 0.01540521 0.01534701 0.01506696 0.01504486 0.01484788
Cumulative Proportion  0.02303868 0.04231675 0.06052493 0.07824758 0.09570828 0.11269334 0.12948710 0.14617321 0.16261193 0.17861050 0.1942441 0.20964931 0.22499632 0.24006328 0.25510814 0.26995602
                          Comp.17    Comp.18    Comp.19    Comp.20    Comp.21    Comp.22    Comp.23    Comp.24    Comp.25    Comp.26    Comp.27    Comp.28    Comp.29    Comp.30    Comp.31    Comp.32
Standard deviation     1.13838196 1.12889068 1.12429911 1.11484336 1.10634870 1.10607053 1.09819751 1.09140282 1.0838406 1.07913609 1.07706163 1.07078478 1.06286985 1.05672587 1.05326482 1.05119497
Proportion of Variance 0.01457253 0.01433055 0.01421421 0.01397612 0.01376395 0.01375703 0.01356188 0.01339458 0.0132096 0.01309518 0.01304488 0.01289328 0.01270338 0.01255693 0.01247482 0.01242583
Cumulative Proportion  0.28452855 0.29885910 0.31307331 0.32704943 0.34081338 0.35457041 0.36813229 0.38152687 0.3947365 0.40783165 0.42087653 0.43376981 0.44647318 0.45903012 0.47150493 0.48393076
                          Comp.33    Comp.34    Comp.35    Comp.36    Comp.37    Comp.38    Comp.39    Comp.40    Comp.41    Comp.42    Comp.43    Comp.44    Comp.45    Comp.46    Comp.47    Comp.48
Standard deviation     1.04333725 1.04116605 1.02871449 1.02479858 1.02307950 1.01728386 1.01558054 1.00733966 1.00662110 0.99744297 0.99650091 0.99570402 0.98376459 0.98205540 0.97349201 0.97125204
Proportion of Variance 0.01224076 0.01218987 0.01190005 0.01180962 0.01177003 0.01163706 0.01159812 0.01141066 0.01139439 0.01118755 0.01116643 0.01114858 0.01088282 0.01084503 0.01065672 0.01060774
Cumulative Proportion  0.49617152 0.50836139 0.52026144 0.53207106 0.54384109 0.55547815 0.56707628 0.57848694 0.58988133 0.60106888 0.61223531 0.62338389 0.63426671 0.64511174 0.65576847 0.66637621
                          Comp.49    Comp.50    Comp.51    Comp.52    Comp.53    Comp.54    Comp.55    Comp.56    Comp.57    Comp.58    Comp.59    Comp.60    Comp.61    Comp.62    Comp.63
Standard deviation     0.96534887 0.95952978 0.95755890 0.95339737 0.94190469 3 0.938251598 0.932304719 0.931150066 0.92130027 0.918250971 0.91639097 0.907151736 0.901228206 0.894532147 0.893069712
Proportion of Variance 0.01047919 0.01035323 0.01031074 0.01022132 0.009976378 0.009899143 0.009774054 0.009749859 0.00954468 0.009481603 0.00944323 0.009253773 0.009133317 0.008998101 0.008968704
Cumulative Proportion  0.67685539 0.68720862 0.69751937 0.70774068 0.717717060 0.727616203 0.737390257 0.747140115 0.75668480 0.766166398 0.77560963 0.784863401 0.793996718 0.802994819 0.811963523
                          Comp.64    Comp.65    Comp.66    Comp.67    Comp.68    Comp.69    Comp.70    Comp.71    Comp.72    Comp.73    Comp.74    Comp.75    Comp.76    Comp.77    Comp.78
Standard deviation     0.887806012 0.880084930 0.877052619 0.871869207 0.865414738 0.859708087 0.856741253 0.847836174 0.846758159 0.837088957 0.829688003 0.823652930 0.817335935 0.812411088 0.808770040
Proportion of Variance 0.008863293 0.008709799 0.008649884 0.008547944 0.008421851 0.008311147 0.008253883 0.008083191 0.008062649 0.007879564 0.007740849 0.007628646 0.007512079 0.007421824 0.007355447
Cumulative Proportion  0.820826816 0.829536615 0.838186499 0.846734442 0.855156293 0.863467440 0.871721324 0.879804515 0.887867164 0.895746728 0.903487576 0.911116222 0.918628301 0.926050125 0.933405572
                          Comp.79    Comp.80    Comp.81    Comp.82    Comp.83    Comp.84    Comp.85    Comp.86    Comp.87    Comp.88    Comp.89
Standard deviation     0.79836916 0.796146604 0.782386719 0.772551940 0.759737073 0.755173064 0.743555034 0.728101546 0.717566860 0.693109862 0.464987600
Proportion of Variance 0.00716748 0.007127629 0.006883383 0.006711419 0.006490611 0.006412863 0.006217062 0.005961326 0.005790069 0.005402106 0.002430481
Cumulative Proportion  0.94057305 0.947700681 0.954584063 0.961295482 0.967786093 0.974198956 0.980416018 0.986377344 0.992167413 0.997569519 1.000000000
```

Figura 20: Visualizzazione del calcolo del metodo summary per il trainset delle domande nel database.

```
> get_eig(res.pca)
       eigenvalue variance.percent cumulative.variance.percent
Dim.1   2.0487956        2.3038680                    2.303868
Dim.2   1.7143702        1.9278071                    4.231675
Dim.3   1.6192262        1.8208178                    6.052493
Dim.4   1.5760487        1.7722647                    7.824758
Dim.5   1.5527547        1.7460707                    9.570828
Dim.6   1.5104555        1.6985053                   11.269334
Dim.7   1.4934446        1.6793765                   12.948710
Dim.8   1.4838713        1.6686114                   14.617321
Dim.9   1.4618708        1.6438718                   16.261193
Dim.10  1.4227291        1.5998571                   17.861050
Dim.11  1.3902728        1.5633600                   19.424410
Dim.12  1.3699621        1.5405206                   20.964931
Dim.13  1.3647870        1.5347012                   22.499632
Dim.14  1.3398826        1.5066963                   24.006328
Dim.15  1.3379168        1.5044857                   25.510814
Dim.16  1.3204001        1.4847882                   26.995602
Dim.17  1.2959135        1.4572531                   28.452855
Dim.18  1.2743942        1.4330546                   29.885910
Dim.19  1.2640485        1.4214209                   31.307331
Dim.20  1.2428757        1.3976122                   32.704943
Dim.21  1.2240074        1.3763948                   34.081338
Dim.22  1.2233920        1.3757027                   35.457041
Dim.23  1.2060378        1.3561879                   36.813229
Dim.24  1.1911601        1.3394580                   38.152687
Dim.25  1.1747105        1.3209604                   39.473647
Dim.26  1.1645347        1.3095178                   40.783165
Dim.27  1.1600618        1.3044880                   42.087653
Dim.28  1.1465801        1.2893278                   43.376981
Dim.29  1.1296923        1.2703376                   44.647318
Dim.30  1.1166696        1.2556935                   45.903012
Dim.31  1.1093668        1.2474815                   47.150493
Dim.32  1.1050109        1.2425833                   48.393076
Dim.33  1.0885526        1.2240760                   49.617152
```

Figura 21: Individuazione degli autovalori per il trainset delle domande nel database.

Osservazioni È essenziale per individuare il numero di componenti (PC) necessarie per effettuare un'analisi corretta basarsi sul calcolo della *variance.percent* o *Proportion of Variance*. A tale scopo è prendere le componenti principali che catturano la maggior parte di variabiltià dei dati.
Nel caso del trainset di prova basta le variabili PC1, PC2 e PC3 sono sufficienti a catturare il 93% della variabilità presente.
Quanto appena descritto si può riscontrare anche graficamente, come presentato di seguito.

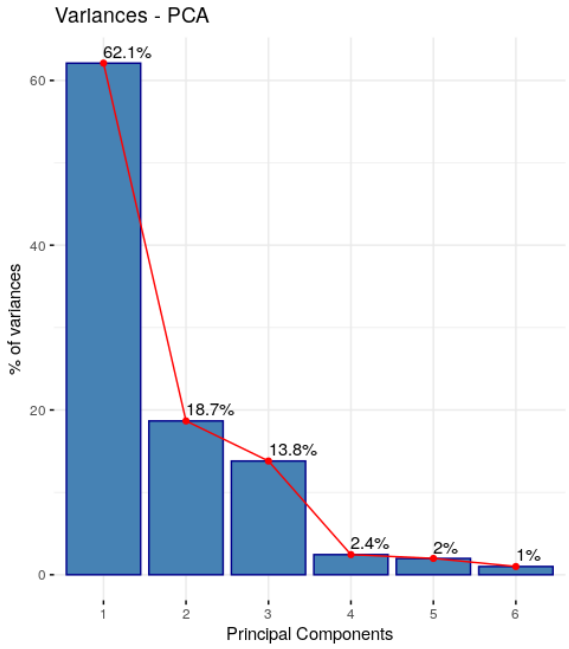

Figura 22: Plot della rappresentazione grafica della varianza sui PC del trainset di prova.

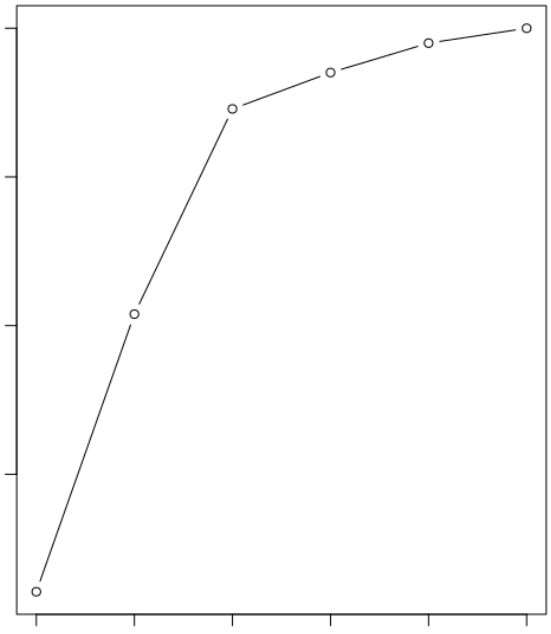

Figura 23: Plot della rappresentazione grafica della varianza sui PC del trainset di prova.

Le ultime PC4, PC5 e PC6 hanno una variabilità molto bassa, trascurabile.

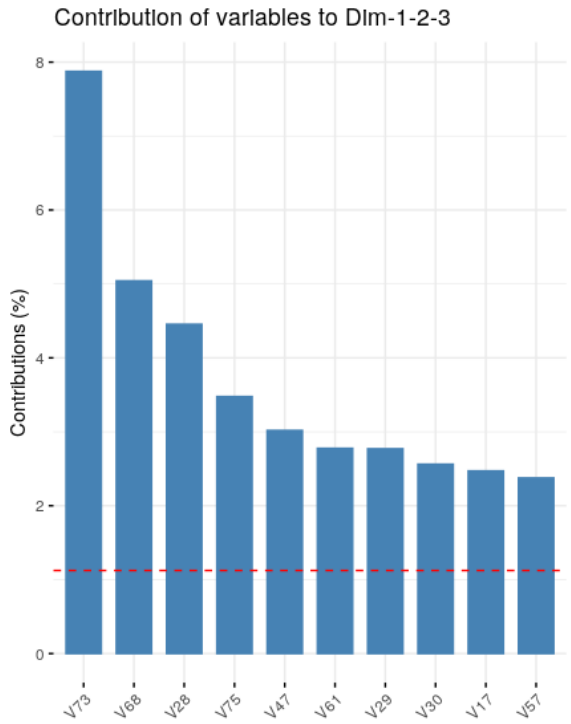

Figura 24: Rappresentazione grafica di come la varianza si distribuisce sulle PC individuate dal modello sul trainset di prova.

Tuttavia per quanto riguarda le domande nel database ogni conclusione "a occhio " risulta impossibile da effettuare sempre a causa della numerosità dei dati di trainset. L'utilizzo di un analisi dei risultati per mezzo di plot è l'unica via percorribile.

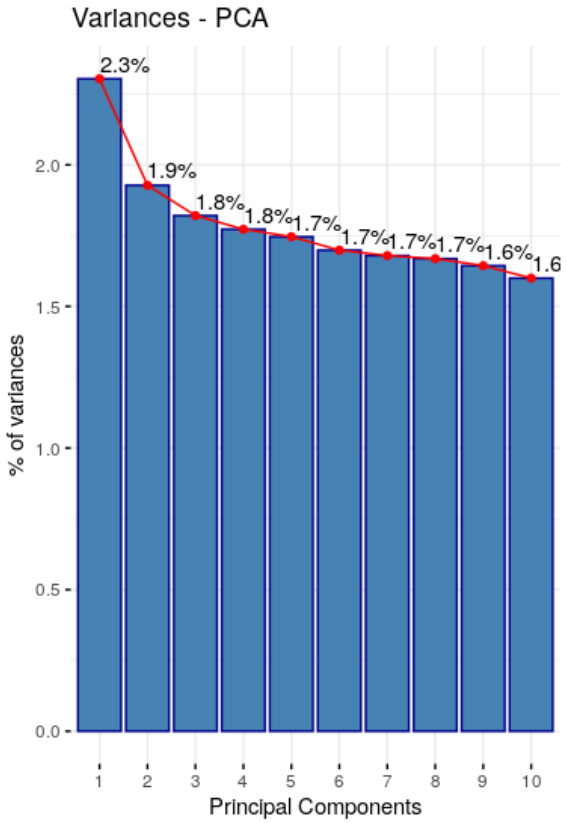

Figura 25: Plot della rappresentazione grafica della varianza dei primi dieci PC del trainset delle domande nel database.

Il plot mostra esclusivamente le prime dieci componenti; tutte si presentano con una varianza molto basse. A causa di ciò per poter affermare quante PC sono indispensabili per una valutazione oggettiva dei dati è indispensabile avere una visione totalitaria di tutte variabili coinvolte nel modello.

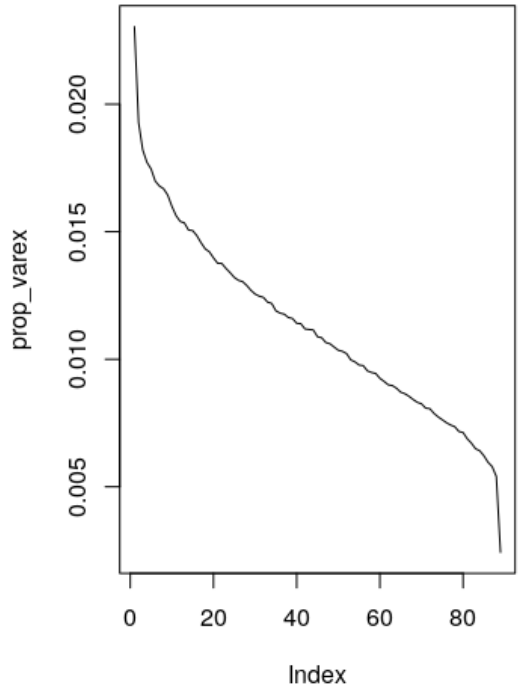

Figura 26: Plot della rappresentazione grafica della varianza di tutti i PC del trainset delle domande nel database.

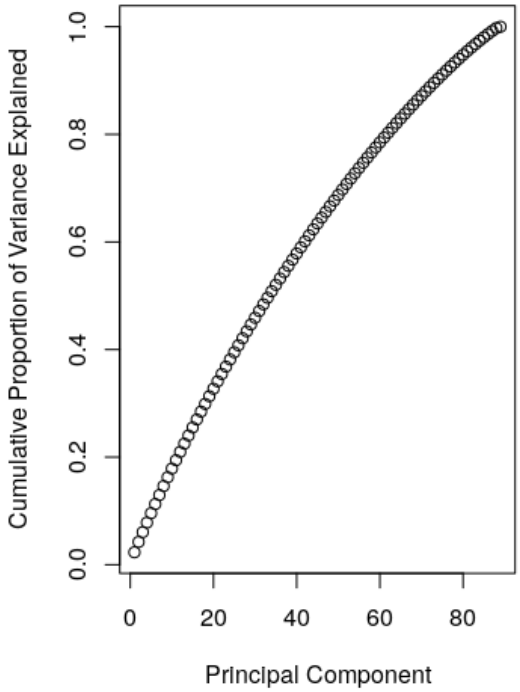

Figura 27: Plot della rappresentazione grafica della varianza di tutti i PC del trainset delle domande nel database.

Non risulta sufficiente l'andamento dei plot per riuscire a definire il numero adeguato di componenti principali da utilizzare. In questi casi, è buona norma, fare riferimento a tre criteri:

- *Quota della varianza totale*: si deve considerare un numero di CP tale che tenga conto di una percentuale sufficientemente elevata di varianza totale proporzionale al numero di variabili originarie (ovvero più è alto il numero di componenti del modello e più è accettata una percentuale minore di varianza spiegata).

- *Screen-graph*: fa uso dei plot degli autovalori in funzione al numero di CP. Gli autovalori sono decrescenti, per cui il grafico ha una buona possibilità, di assumerà la forma di una spezzata con pendenza negativa.

- *Eigenvaue one o Regola di Kaiser*: afferma di considerare tutte ed esclusivamente le CP con autovalore maggiore di 1.

Per soddisfare il *primo criterio* è sufficiente fare riferimento a *variance.percent* in get_eigen o *Proportion of Variance* in summary. Da questa asserzione ne consegue un quesito: quale è il numero di varianza accettabile avendo un numero di variabili molto elevato?. Per effettuare una delle vie utilizzate è procedere al soddisfacimento del *terzo criterio*. Gli autovalori di tutte le componenti coinvolte si possono vedere nella *Standard deviation* risultante dalla summary. Le PC del modello, con autovalore superiore a 1, sono le PC contenute nell'intervallo 1-41. Il *secondo criterio*, invece, in questo caso specifico non è stato in grado di dirmi molto, in quanto la diminuzione degli autovalori è graduale, senza salti evidenti.
In conclusione, ho considerato come una percentuale di copertura adeguata quella fornita dalla che 41 prime componenti principali.

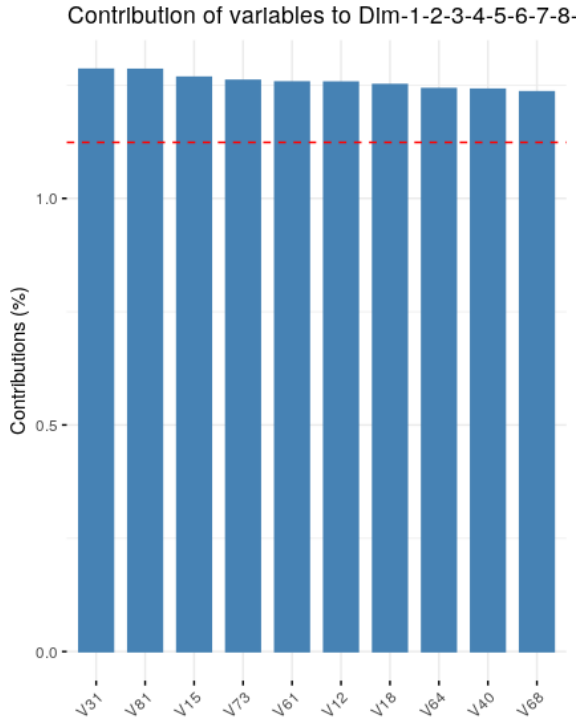

Figura 28: Rappresentazione grafica di come la varianza si distribuisce sulle PC individuate dal modello sul trainset delle domande nel database.

3.3.3 Calcolo degli autovettori

```
> loadings(res.pca)

Loadings:
   Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
V1  0.420         0.569  0.702
V2 -0.394  0.515  0.282        -0.707
V3  0.418  0.477 -0.278                -0.721
V4  0.421         0.565 -0.709
V5 -0.393  0.518  0.279         0.707
V6  0.402  0.489 -0.350                 0.689

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var  0.167  0.167  0.167  0.167  0.167  0.167
Cumulative Var  0.167  0.333  0.500  0.667  0.833  1.000
```

Figura 29: Autovettori del trainset di test.

Osservazioni L'analisi dei loadings sulle componenti principali permette di determinare il contributo delle variabili originarie al modello PC.
Per quanto riguarda l'analisi dei dati del trainset la prima variabile ha valori 0.420, 0.569 e 0.702, rispettivamente nelle componenti 1, 3 e 4 e sembra legarsi ai risultati nella variabile 4 (che riempie le medesime componenti oscillando di poco nei valori presentati), il medesimo match coinvolge le variabili 2 con 5 e 3 con 6.
Tali assunzioni sono molto più complesse da effettuare per il trainset dei dati delle domande nel databas e per fare ciò ho provveduto a calcolare la matrice correlazione.

3.3.4 Calcolo della matrice di correlazione

```
> cor.pca
          V1         V2         V3         V4         V5         V6
V1  1.0000000 -0.4725415  0.5354458  0.8534648 -0.4724833  0.4749618
V2 -0.4725415  1.0000000 -0.4025775 -0.4839185  0.8811501 -0.3892455
V3  0.5354458 -0.4025775  1.0000000  0.5237420 -0.3996431  0.9371832
V4  0.8534648 -0.4839185  0.5237420  1.0000000 -0.4838589  0.4736120
V5 -0.4724833  0.8811501 -0.3996431 -0.4838589  1.0000000 -0.3862259
V6  0.4749618 -0.3892455  0.9371832  0.4736120 -0.3862259  1.0000000
```

Figura 30: Autovettori del trainset di test.

```
> cor.pca
```

	V1	V2	V3	V4	V5	V6	V7
V1	1.000000000	0.0200272106	-0.0045634941	-0.0090382940	0.0324932263	-1.222665e-02	1.449970e-02
V2	0.020027211	1.0000000000	0.0169844299	0.0277904712	0.0022019004	-1.355124e-02	7.014736e-02
V3	-0.004563494	0.0169844299	1.0000000000	0.0090900228	0.0035848287	-5.824745e-03	-1.744317e-02
V4	-0.009038294	0.0277904712	0.0090900228	1.0000000000	0.0070999842	-2.448739e-04	5.624869e-02
V5	0.032493226	0.0022019004	0.0035848287	0.0070999842	1.0000000000	9.604583e-03	1.852404e-02
V6	-0.012226653	-0.0135512363	-0.0058247451	-0.0002448739	0.0096045826	1.000000e+00	-9.605035e-03
V7	0.014499703	0.0701473627	-0.0174431726	0.0562486863	0.0185240363	-9.605035e-03	1.000000e+00
V8	0.007075831	0.0149970844	0.0096161144	-0.0022587458	-0.0055583816	-2.443326e-02	5.818110e-02
V9	-0.064951922	-0.0153076421	0.0290833538	0.0365071171	0.0257313911	8.692034e-03	1.854900e-02
V10	0.009635273	0.0310284650	-0.0186844323	0.0152464032	-0.0075689377	-8.088067e-03	1.563069e-02
V11	0.002557398	-0.0327089285	-0.0499445853	0.0338330601	-0.0020089503	-6.552641e-02	2.313817e-02
V12	0.008687118	-0.0105609686	0.0118058675	0.0355881260	-0.0068241198	3.779342e-02	-3.108653e-03
V13	-0.012838870	-0.0556537880	0.0042965374	-0.0126175020	0.0100855066	8.639492e-03	-5.174636e-02
V14	0.006179775	-0.0187787934	-0.0123475985	0.0236108359	-0.0048544895	-6.909373e-02	3.624605e-02
V15	-0.074558925	-0.0100399765	0.0264158927	-0.0093046234	-0.0238124172	-3.089299e-02	1.114677e-02
V16	-0.026626358	-0.0111545265	0.0565710729	-0.0469258898	0.0042767939	1.902076e-03	1.274478e-03
V17	0.022553887	0.0198581396	0.0232952095	0.0151158566	0.0135916823	-6.883897e-02	-1.720176e-02
V18	-0.031959719	0.0192947677	0.0154470300	-0.0282878561	0.0568743229	-3.990590e-03	4.406076e-02
V19	0.079722819	0.0616647697	0.0205788512	-0.0260978417	0.0235808600	1.055257e-02	5.602897e-03
V20	-0.104791476	0.0498297021	-0.0033508826	-0.0066366388	-0.1442112569	-8.977787e-03	-5.282246e-02
V21	0.001301225	0.0316287927	-0.0186170659	0.0240706152	-0.0010221703	-1.603207e-02	6.974488e-02
V22	0.045962996	-0.0423900226	0.0268650232	0.0698367467	0.0006269681	-4.037566e-02	7.071889e-03
V23	0.009656707	-0.0113444095	0.0444239593	-0.0161160728	-0.0075857749	5.642155e-02	-4.114964e-03
V24	0.020861554	0.0024512277	-0.0078510543	0.0115072959	0.0140498602	-5.518751e-02	1.604833e-02
V25	0.013275958	-0.0355132590	-0.0423664687	-0.0102464149	0.0061111783	-4.992188e-02	2.756445e-02
V26	-0.080976981	0.0437900955	-0.0284118478	-0.0195245706	0.0056982468	-2.641208e-02	-3.814904e-02
V27	0.001958346	0.0052576995	0.0289586575	-0.0389506607	-0.0015383680	-3.752493e-02	-4.667907e-02
V28	0.060859805	0.0056544438	0.0016281298	0.0032246160	0.0842716139	4.362135e-03	1.617665e-02
V29	-0.068680869	0.0040028793	0.0035001708	0.0069323137	0.0234200271	9.377764e-03	-1.282110e-02
V30	0.081173322	0.0242603493	0.0050719649	0.0100453533	-0.0582349537	1.358896e-02	-4.936974e-02
V31	0.015643566	-0.0185121426	-0.0145959643	-0.0217811370	-0.0112063393	-9.150019e-02	1.525326e-02
V32	0.024600801	-0.0838273079	-0.0108354505	-0.0380029342	-0.0072042931	-3.782271e-02	2.223962e-02
V33	0.004074525	-0.0149726427	0.0328942810	0.0661715488	-0.0032007214	-4.091023e-02	6.711906e-02
V34	0.016626451	-0.0446870482	-0.0024101855	0.0027019968	-0.0130608211	-1.589349e-02	1.116960e-02
V35	-0.040844583	-0.0506299348	-0.0097860230	0.0067496206	0.0282693454	1.843719e-02	-6.455991e-02
V36	-0.010443892	-0.0785602870	-0.0034643120	-0.0172855893	0.0082041441	-3.802722e-02	1.356867e-02
V37	0.009650804	-0.0150058025	0.0635131458	-0.0176091642	-0.0075811384	-2.354418e-02	-4.576715e-02
V38	-0.009846157	0.0236583883	-0.0350306022	0.0281073376	0.0230927681	1.929406e-02	1.481069e-02
V39	0.012152122	-0.0291991701	-0.0027640867	-0.0548243741	-0.0095460355	4.961918e-03	1.438521e-03
V40	0.049559619	-0.0272532676	-0.0171460800	-0.0206636799	-0.0368280446	-2.305003e-03	-3.061603e-02
V41	0.011096954	-0.0374910888	-0.0134970214	-0.0374110271	-0.0087171539	1.582526e-03	8.971930e-03

	V8	V9	V10	V11	V12	V13	V14
V1	0.0070758308	-0.0649519219	0.0096352727	0.0025573977	0.0086871180	-1.283887e-02	0.0061797747
V2	0.0149970844	-0.0153076421	0.0310284650	-0.0327089285	-0.0105609686	-5.565379e-02	-0.0187787934
V3	0.0096161144	0.0290833538	-0.0186844323	-0.0499445853	0.0118058675	4.296537e-03	-0.0123475985
V4	-0.0022587458	0.0365071171	0.0152464032	0.0338330601	0.0355881260	-1.261750e-02	0.0236108359
V5	-0.0055583816	0.0257313911	-0.0075689377	-0.0020089503	-0.0068241198	1.008551e-02	-0.0048544895
V6	-0.0244332634	0.0086920338	-0.0080880666	-0.0655264062	0.0377934184	8.639492e-03	-0.0690937292
V7	0.0581811046	0.0185490010	0.0156306903	0.0231381746	-0.0031086528	-5.174636e-02	0.0362460523
V8	1.0000000000	0.0766508911	0.0078533144	0.0256721371	-0.0263441964	-2.351978e-02	-0.0084265948
V9	0.0766508911	1.0000000000	-0.0349939777	-0.0142172292	-0.0101187252	-2.265859e-02	-0.0084913600
V10	0.0078533144	-0.0349939777	1.0000000000	0.0527671384	0.0174195214	-1.752897e-02	-0.0315645566
V11	0.0256721371	-0.0142172292	0.0527671384	1.0000000000	0.0086376189	-9.989410e-02	0.0476110206
V12	-0.0263441964	-0.0101187252	0.0174195214	0.0086376189	1.0000000000	-2.887564e-02	-0.0436317988
V13	-0.0235197807	-0.0226585933	-0.0175289660	-0.0998940992	-0.0288756354	1.000000e+00	-0.0118652385
V14	-0.0084265948	-0.0084913600	-0.0315645566	0.0476110206	-0.0436317988	-1.186524e-02	1.0000000000
V15	0.0207768267	-0.0392901147	-0.0222212880	-0.0017877176	0.0381403393	-6.254951e-02	-0.0093746618
V16	0.0114722747	-0.0170850337	-0.0376931682	-0.0025752982	0.0019073041	-4.270461e-02	-0.0735133872
V17	0.0732774112	0.0398040762	-0.0153210876	0.0053478215	-0.0043380998	-9.066885e-02	0.0243573872
V18	-0.0312988401	0.0308294586	-0.0086538392	0.0232946240	-0.0215117795	7.200024e-03	0.0003392265
V19	-0.0094769428	-0.0013309297	0.0165682252	-0.0138970684	-0.0263225378	6.687972e-02	0.0178439408
V20	0.0051956413	-0.0043515419	0.0070749883	0.0018778461	0.0063787773	-9.427326e-03	0.0045376852
V21	0.0514436061	0.0165919084	-0.0139277357	0.0504173164	0.0490217371	-5.487763e-03	0.0508444448
V22	0.0407826922	0.0130334834	-0.0072728103	-0.0258236338	-0.0311185464	-3.023208e-02	-0.0307062098
V23	0.0143174178	-0.0122236390	-0.0277088011	-0.0205101241	-0.0428572731	-3.804852e-02	-0.0450199111
V24	0.0522678520	0.0180121917	0.0098716182	-0.0209599568	-0.0309977718	-1.442688e-03	0.0368945664
V25	0.0375204283	-0.0062977412	-0.0200749214	-0.0208013582	0.0080211762	-1.886551e-02	-0.0064415288
V26	0.0091206432	0.0847803298	0.0022580353	0.0172217957	-0.0024255320	-3.725737e-02	-0.0108781581
V27	0.0657728135	-0.0641613837	0.0119144912	0.0206141402	-0.0150782337	-1.950671e-02	0.0192888389
V28	-0.0025244628	0.0231730686	-0.0034376016	-0.0009124095	-0.0030993260	4.580558e-03	-0.0022047745
V29	-0.0054271170	0.0045454114	-0.0073901925	-0.0019615077	-0.0066629640	9.847331e-03	-0.0047398477
V30	-0.0078642298	-0.0754163816	-0.0107088482	-0.0028423465	-0.0096550488	1.426939e-02	-0.0068683339
V31	-0.0111650752	0.0329582484	-0.0395465069	-0.0292287789	-0.0230433864	-5.295572e-02	0.0140466499
V32	0.0735698278	0.0488627662	-0.0496173499	0.0028080101	0.0206264840	-4.796123e-03	0.0135462876
V33	0.0580707391	0.0647437596	-0.0299316827	0.0313914808	-0.0001254868	2.493929e-02	0.0401322686
V34	-0.0904235780	-0.0174180527	-0.0393713876	0.0083573090	0.0045880559	-3.054788e-02	-0.0469246356
V35	-0.0444008329	-0.0469368450	-0.0450905890	-0.0265444157	-0.0325695266	4.813517e-02	-0.0173156735
V36	0.0077480409	0.0267110484	0.0514283284	0.0553030847	0.0331457637	-1.790453e-02	0.0192202794
V37	0.0035853161	0.0220809750	-0.0492936022	0.0062671749	-0.0290784657	2.211805e-02	-0.0553663615
V38	-0.0224116215	0.0585327987	0.0066000548	-0.0129748364	0.0131085226	6.800904e-03	-0.0242852181
V39	-0.0127956515	-0.0191118988	-0.0348691071	0.0028995006	-0.0039131474	-3.012201e-03	0.0153979974
V40	-0.0423706369	-0.0372822005	-0.0280445639	-0.0526978338	-0.0223886353	6.927160e-02	-0.0273261579
V41	-0.0360435448	-0.0271766879	0.0253216236	0.0275827018	-0.0069476185	2.287136e-02	-0.0266416937

Figura 31: Autovettori del trainset delle domande nel database.

I CSV generati di correlazione sono i seguenti:

	A
1	V1" "V2" "V3" "V4" "V5" "V6
2	1" 1 4 3 6 5 2 "2" 2 5 6 3 1 4 "3" 3 6 1 4 5 2 "4" 4 1 3 6 5 2 "5" 5 2 6 3 1 4 "6" 6 3 1 4 5 2

Figura 32: CSV generato a partire dalla matrice correlazione del trainset della rete di prova.

```
"34" 34 60 52 38 26 54 36 48 65 69 39 84 16 18 1 73 46 20 59 40 7 64 32 72 11 83 68 75 82 88 57 12 19 89 4 86 81 58 43 56 71 66 42 3 70 28 21 61 41 74 29 85 5 23 87 6 9 30 47 78 13 77 50 67 76 22 27 79 10 49 2 51 24 14 80 17 45 37 63 35 31 25 55 62 44 33 15 53 8
"35" 35 68 57 78 84 13 56 48 33 36 5 72 66 16 17 6 46 24 74 50 25 61 23 19 4 40 83 29 39 69 30 88 27 85 81 15 22 70 63 60 38 49 58 3 47 26 76 51 21 87 42 77 82 79 14 54 44 31 65 71 59 18 62 41 11 89 52 53 12 80 55 37 43 75 64 1 8 10 9 86 2 32 34 28 20 67 7 45 73
"36" 36 54 85 11 10 23 75 84 74 55 12 35 64 9 34 59 57 14 50 44 7 32 18 60 30 49 76 40 31 39 5 25 29 8 47 41 79 28 65 72 52 82 71 15 69 70 3 78 48 68 83 86 58 21 46 43 20 62 38 73 51 1 24 87 45 26 66 17 4 13 19 67 81 88 80 61 89 56 63 77 6 53 16 33 42 37 22 27 2
"37" 37 53 3 58 24 75 65 17 48 59 89 13 9 61 49 43 46 67 80 1 55 73 76 20 33 11 38 47 68 83 31 8 45 28 60 66 42 70 29 5 72 69 71 21 30 74 16 25 81 2 62 86 15 4 79 19 82 85 51 18 40 6 78 12 64 88 77 27 26 63 57 35 52 22 7 54 56 50 10 32 34 39 14 23 84 36 41 44 87
"38" 38 85 9 39 40 18 34 72 71 65 4 29 82 50 2 5 6 23 69 51 7 62 12 56 33 61 45 22 13 10 77 37 46 59 21 28 41 80 43 49 55 64 60 26 67 68 83 42 79 52 74 20 36 35 73 1 63 27 11 88 87 54 19 48 86 31 8 66 14 53 16 47 30 84 25 3 44 24 81 76 78 58 70 57 75 17 32 89 15
"39" 39 38 66 82 79 62 74 24 26 81 34 16 31 63 61 52 14 19 17 44 1 73 64 20 36 78 60 49 68 83 75 85 6 65 43 35 76 23 11 32 7 87 86 3 13 12 28 53 15 29 5 25 8 30 46 58 18 71 77 56 9 88 21 48 59 22 69 2 41 70 57 10 80 27 55 72 47 54 40 50 67 42 84 51 37 4 33 45 89
"40" 40 73 13 29 30 1 38 26 70 87 56 41 55 89 64 80 28 50 17 34 16 36 83 22 84 35 88 19 69 6 43 74 25 32 23 86 78 24 76 85 31 72 57 46 42 62 79 47 65 63 75 77 61 3 51 4 58 71 12 37 82 53 27 49 66 60 2 14 10 18 59 44 7 54 81 5 9 8 39 33 11 52 20 21 45 68 15 67 48
"41" 41 70 50 81 56 82 11 10 61 13 40 76 63 22 1 73 60 7 16 20 25 23 36 83 68 32 17 38 77 19 6 27 52 26 28 89 65 69 12 29 5 55 34 79 24 84 54 30 3 44 31 46 87 78 15 62 21 47 49 35 14 9 43 72 53 85 66 86 80 39 51 74 64 8 4 2 45 48 71 42 18 59 57 33 75 67 58 37 88
"42" 42 52 56 44 4 84 51 59 47 33 63 54 22 80 72 87 12 53 30 3 57 76 5 29 18 69 64 62 19 2 9 8 28 67 14 89 60 71 25 26 82 58 34 46 10 38 37 13 68 83 20 88 16 40 73 1 31 35 48 50 85 6 78 27 7 15 45 24 65 79 70 77 23 21 49 75 55 32 66 11 81 17 39 41 43 36 86 61 74
"43" 43 75 7 24 9 17 63 31 6 15 47 65 8 32 50 28 66 37 71 78 56 77 89 49 53 26 14 16 18 55 10 60 5 29 39 46 38 72 34 54 11 76 23 68 83 87 88 25 84 40 22 4 12 2 36 69 48 33 85 59 51 79 13 80 44 3 19 27 73 41 86 70 45 81 52 35 67 21 1 74 30 62 20 61 64 57 42 58 82
"44" 44 25 57 28 42 69 55 3 30 73 63 58 74 20 85 86 51 14 47 36 39 32 84 5 64 60 53 83 77 2 21 82 46 67 9 6 62 66 15 65 17 27 61 56 24 54 16 1 78 22 10 41 76 7 4 79 23 71 35 43 50 49 13 19 31 40 33 70 29 26 8 88 38 75 89 72 12 81 11 59 68 87 18 52 34 80 48 45 37
"45" 45 67 46 14 4 63 25 61 72 19 21 11 7 64 86 66 81 82 16 38 70 58 1 73 47 20 2 68 83 13 37 6 28 49 74 24 29 5 79 30 80 60 9 65 52 18 36 53 78 55 48 33 15 27 76 42 88 89 85 69 10 43 17 50 31 51 56 3 57 75 41 26 87 32 34 23 62 71 22 59 40 84 12 35 39 44 8 54 77
"46" 46 45 69 82 61 76 50 56 23 59 51 89 49 81 86 37 57 14 26 35 87 34 30 6 5 29 11 4 16 10 38 28 43 44 80 25 27 65 42 68 83 67 32 20 36 47 2 52 73 24 1 18 63 40 72 84 75 39 88 19 53 54 41 22 48 66 3 78 17 55 62 71 33 70 7 64 85 79 31 13 58 21 8 60 12 74 15 9 77
```

Figura 33: CSV generato a partire dalla matrice correlazione del trainset delle domande nel database.

I file CSV gli ho generati prendendo la matrice correlazione, che mostra il grado di correlazione di ogni componente con ogni variabile del modello, creandovi un data frame che mostra per ogni variabile quale è la componente a cui si collega in ordine decrescente (da quella che si correla di più a quella che si correla di meno).
Da tale distribuzione dei dati, per il trainset di prova emerge quello che mi aspettavo. Per esempio la variabile 1 si correla con grado massimo con se stessa, successivamente con la componente 4 (che è la sua domanda sorella) e successivamente con le domande 3 e 6 (i suoi genitori) e alla fine con le domande 2 e 5 (nella realtà queste ultime due non hanno alcuna correlazione

con la domanda 1). Tale ragionamento vale per tutte le altre 5 componenti, i cui risultati rimangono coerenti con le aspettative e con ciò che viene dichiarato nel Grafo della Conoscenza, rappresentato nella figura 6 nella sezione § 2.1.

Pensavo di poter effettuare il medesimo ragionamento anche per analisi dei dati del trainset del database, tuttavia questo non è stato possibile, a causa delle seguenti motivazioni:

- In primo luogo, se una variabile nella matrice correlazione è dichiarata in correlazione con una componente, questa poi non risulta in correlazione con la variabile;
- Analizzando il testo delle domande, contenute nel database aziendale ho riscontrato come il modello prodotto crea relazioni strette fra domande appartenenti trivialmente a categorie diverse, (ad esempio domande che trattano relazioni con altre di serie numeriche) e meno con domande che parlano dello stesso tema.
- Dal pot delle variabili viene mostrato come le stesse indicate come correlate risultano, invece, sparse impedendo la formazione dei cluster. Su un numero n grande di variabili (come le 89 domande nel database) tale fenomeno verrebbe espanso impedendo a chi interpreta il modello di individuare una correlazione tra il grafico delle variabili e la matrice di correlazione.

A questo punto ho iniziato a farmi delle domande. Escludendo un mio errore di codifica (dopo aver opportunamente effettuato un accurato controllo del codice da me prodotto) e trovandomi di fronte ad una situazione dove i risultati inerenti ai dati del trainset di prova soddisfano appieno le attese (infatti non solo le correlazioni matchano perfettamente con quanto viene indicato dal grafo della conoscenza; ma anche il plot che viene generato dalla PCA sulle variabili presenta una coerenza stretta con tali assunzioni) ho iniziato a pensare che centrasse la possibilità che ha un candidato di indovinare correttamente una risposta ad una domanda.

Ho rifatto, perciò, tutta l'analisi su un nuovo modello, adoperando il trainset di prova generato da n input sottoposti alla probabilità di indovinare. Come viene illustrato dai plots nella sezione seguente, i dati individuati dal trainset di prova puro vengono totalmente falsati quando si tiene conto di tale eventualità rispetto alle aspettative imposte dal Grafo della conoscenza.

3.4 Conclusione dell'analisi

In conclusione, un test con domande a triplice risposta multipla concede ad un candidato l'elevata possibilità di indovinare le domande che non sa; e il

modello creato dalla Principal **C**omponent **A**nalysis con i dati delle risposte alle domande nel database ne fornisce la prova.

Infatti se la possibilità di indovinare fosse bassa i risultati del modello in esame ne verrebbero appena "sporcati", e invece invalidano completamente la possibilità di avere un risultato attendibile e coerente con la realtà.
Le figure seguenti mostrano proprio tale fenomeno: nel caso dell'indovinato le domande con una maggiore correlazione non creano alcun cluster.

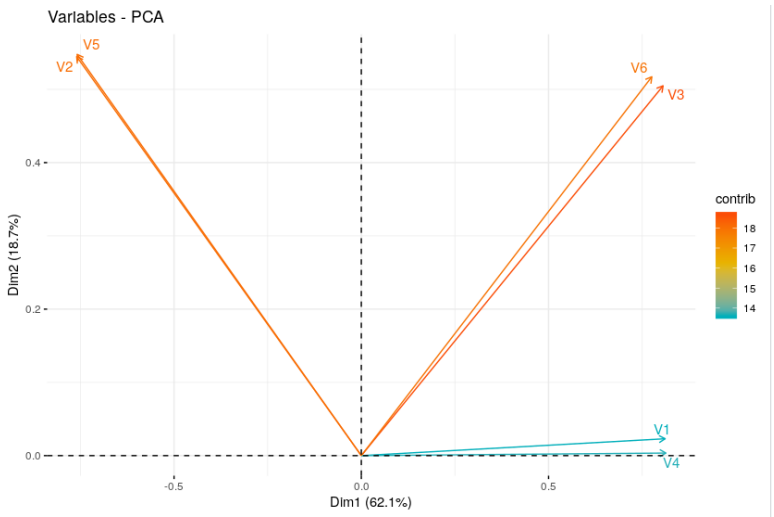

Figura 34: Rappresentazione per mezzo di plot di come le variabili si presentano nelle due componenti principali con il calcolo della PCA - utilizzo di trainset di prova puro sul grafo della conoscenza.

```
> correlazione
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    4    3    6    5    2
[2,]    2    5    6    3    1    4
[3,]    3    6    1    4    5    2
[4,]    4    1    3    6    5    2
[5,]    5    2    6    3    1    4
[6,]    6    3    1    4    5    2
```

Figura 35: Le domande in correlazione con la variabili principali in ordine decrescente sui dati di prova.

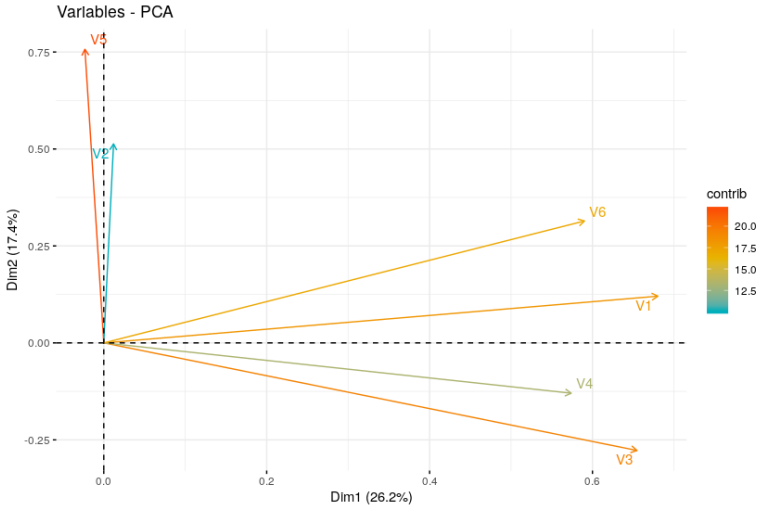

Figura 36: Rappresentazione per mezzo di plot di come le variabili si presentano nelle due componenti principali con il calcolo della PCA - utilizzo di trainset di prova spurio con la probabilità di indovinare.

```
> correlazione
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    6    3    4    5    2
[2,]    2    6    5    4    1    3
[3,]    3    1    4    6    2    5
[4,]    4    3    6    1    2    5
[5,]    5    6    2    1    4    3
[6,]    6    1    4    3    5    2
```

Figura 37: Le domande in correlazione con la variabili principali in ordine decrescente sui dati di prova soggetti alla probabilità di indovinare.

L'analisi del modello ha evidenziato come il Reticolo della Conoscenza costruito dai dati delle domande nel database, una volta costruito, risentirà inevitabilmente del fattore di rischio "domande indovinate". Mi aspetto, per cui, che in rapporto al contenuto delle stesse gli argomenti e la difficoltà delle domande appartenenti al medesimo cluster possano non essere coincidenti.
L'unica possibilità per continuare la costruzione del reticolo è rivolgere nuovamente l'attenzione alla Rete neurale e studiare i cluster possibili per mezzo delle previsioni assunte da ogni variabile sulla base della previsione standard[2].

[2]Vettore previsione tutto a zero

4 Costruzione del Reticolo della Conoscenza

Dal 26/07 al 28/07 ho iniziato la Costruzione del Reticolo della Conoscenza. Per realizzarlo ho utilizzato un'applicativo sviluppato durante un precedente stage dell'azienda, che fa largo uso della libreria d3.js; effettuando elaborazione di dati presentandoli in *Cluster Based* o *Force Based*, a discrezione delle esigenze dell'utente.

4.1 Descrizione del sistema

TODO: FOTO DELL'APPLICATIVO (magari con qualche tendina aperta per far vedere le opzioni) L'applicazione accetta in import file di estensione CSV. Ogni colonna di quest'ultimo viene interpretata dal sistema come un parametro da elaborare; per questo la prima riga del file deve essere o preceduta da una dichiarazione di variabili, tante quante sono le colonne da parametrizzare, oppure è questa prima riga che viene interpretata come una dichiarazione e conseguentemente non rappresentata all'interno del Reticolo. Nel sistema possono venire settati i seguenti aspetti:

- La *tipologie* di Reticolo:
  - Cluster Based: raggruppa un insieme di oggetti in modo tale che gli tutti gli elementi contenuti nel medesimo cluster sono più simili l'uno all'altro rispetto a quelli contenuti in altri gruppi;
  - Force Based: in base alla forza di ogni nodo viene rappresentata come unica regione compatta le istanze appartenenti alla medesima classe in cui vengono visivamente identificati i percorsi di differenziazione. Nel layout le celle differenzianti sono poste in prossimità della classe più fortemente correlata.
- *Normalizzazione* dei dati in input:
  - No: non viene applicata alcuna tecnica di normalizzazione dei dati;
  - MinMax: i dati vengono ridimensionati su un intervallo specifico (min, max), tuttavia tale tecnica non è in grado di gestire i valori anomali;
  - Gaussian: o normale in cui i dati vengono normalizzati in una curva in cui i valori della stessa grandezza sono soggetti ad approssimazione;
  - Interquartile: si occupa di standardizzare i dati in modo da quantificare l'estensione del 50% della distribuzione del carattere che si trovano attorno alla mediana;
- Tipologia di *distanza* applicabili ai punti:

  - Euclidea: tiene conto della distanza tra i punti;
  - Camberra: tiene conto della distanza tra le coppie in uno spazio vettoriale;
  - Pearson: distanza di correlazione che misura il grado di correlazione tra due punti. Valuta la covarianza tra due variabili in rapporto al prodotto della deviazione standard. Non è vantaggiosa su dati semplici.

- *Metodo* di associazione dei punti:
  - Single: "vicino al prossimo", la distanza fra i gruppi è posta al pari della più piccola delle distanze calcolabili a due a due tra tutti gli elementi del gruppo. Accentua tutte le somiglianze tra i gruppi a discapito della loro differenziazione netta.
  - Complete: "vicino più lontano", viene considerata la maggiore tra le distanze calcolate a due a due tra gli elementi di due gruppi. Privilegia la differenziazione tra i gruppi a discapito dell'omogeneità degli elementi in essi contenuti. In questo caso i punti vengono rappresentati come meno compatti e diluiti.
  - Average: viene considerata come distanza fra due gruppi la media fra tutte le distanze calcolate a due a due tra gli elementi dei due gruppi. I risultati ottenibili sono i più attendibili (essendo basato sulla media delle distanze), i gruppi risultano più omogenei e differenziati tra di loro.

Un'ulteriore funzionalità permette all'utente di decidere se si desidera procedere con una rappresentazione del Reticolo manuale o automatica progressiva dei dati.

4.1.1 Configurazione

Analizzando l'applicativo in base al carattere dei dati in ingresso e alle aspettative sull'output del modello, ho riscontrato che la configurazione necessaria per la formazione del Reticolo della Conoscenza è vincolata alla dischiarazione delle seguenti proprietà:

- *Redistance*: No;
- *Normalize*: No;
- *Distance-Type*: Euclidea;
- *Method*: Single.

TODO: SCREEN DELLA CONFIGURAZIONE IN USO PER LA CREAZIONE DEL RETICOLO

5 Creazione dei file CSV

Come già accentato all'interno della sezione §4.1 prima di procedere alla creazione del Reticolo ho dovuto preparare i dati di previsione prima di poterli dare in pasto al sistema. Questo è stato reso più agevole grazie alla creazione, da parte mia, di un metodo che ha il compito, una volta messa in funzione la Rete neurale oggetto di studio (di prova o del database), di calcolare:

1. Le previsioni ottenibili da un vettore di previsione settato a 1 o -1 per ogni singolo elemento;

2. Sui dati del punto (1) un vettore delle differenze dove viene calcolato il delta in rapporto al vettore di standard [3].

Ho fatto in modo che il vettore delle differenze venga stampato su console del browser [4], in modo che ne basti prelevare il contenuto e inserirlo su un file CSV. Ogni elemento per poter funzionare all'interno dell'applicativo deve essere separato da un ; e ogni riga di previsione deve essere preceduta dal codice della domande in modo da rendere più agevole l'interpretazione del Reticolo. È a discrezione dell'utente l'inserimento di un'ulteriore riga di dichiarazione dei parametri.

6 Creazione del Reticolo della Conoscenza per sui dati di Prova

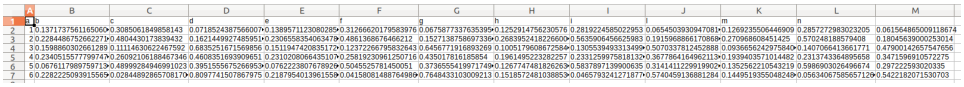

Figura 38: file CSV generato per la creazione del Reticolo della Conoscenza sui dati di prova.

TODO: SCREEN DEI VARI PASSAGGI DI FORMAZIONE DEL RETICOLO PER LA RETE DI PROVA

6.0.1 Osservazioni

6.0.2 Osservazioni Reticolo dati di prova

Il Reticolo della Conoscenza è perfettamente coerente con le aspettative; ricalca fedelmente quanto evidenziato nella figura 6 nella sezione § 2.1 Inoltre

[3] Vettore tutto a zero

[4] unica alternativa facendo uso di solo codice javascript

gli accoppiamenti tra i punti ricalcano quanto dichiarato dalla matrice correlazione ottenuta dal calcolo del modello effettuato con la PCA e visibile dall'immagine 32. Tuttavia i risultati della PCA non devo essere presi come assioma in quanto la Rete neurale, può cogliere oscillazioni che il modello matematico non è in grado.

7 Creazione del Reticolo della Conoscenza per sui dati delle domande nel database

Figura 39: Porzione di esempio file CSV generato per la creazione del Reticolo della Conoscenza sui dati del database.

La creazione del Reticolo della Conoscenza sui dati delle domande nel database ha richiesto la creazione di file CSV su un architettura a 2 layer della rete con 6, 8, 10, 12 neuroni ciascuno. Questo a causa della non conoscenza a priori del numero di cluster che comporranno il Reticolo, come invece accade per i dati di prova.
Tale differenziazione ha permesso durante la configurazione del sistema e la creazione del Reticolo che mi accorgersi dell'estrema variabilità dei cluster e correlazioni a seconda dell'oscillazione del numero di neuroni per layers, come viene evidenziato dalle immagini seguenti.

TODO: INSERIMENTO DEI SCREEN DEL RETICOLO SUI CASI A 6, 8, 10, 12 NEURONI A CLUSTER E A FORCE BASED.

7.1 Osservazioni

7.1.1 Osservazioni Reticolo dati del database

La variazione dei Reticoli ottenuti dalle diverse architetture della Rete neurale indica come sia necessario approfondire la tematica e individuare una

strategia unica che permetta di individuare in modo univoco l'architettura necessaria per ottenere un Reticolo della Conoscenza valido.